A.

1.

How can the organization best allocate resources to direct sales, improve service provision, and or client facing services in order to minimize 'Churn' ?

2.

The goals of this data analysis are to indentify correlations and relationships in the data set that are actionable and have a positive correlation with 'Churn'.

B.

1.

Observations are independent:

Logistic regression assumes the observations to be independent of each other and independent of repetitive measurement. Any individual should not be measured more than once and neither should it be taken in for the model.

No Multicollinearity: It is essential that the independent variables are not too highly correlated with each other, a condition known as multicollinearity. This can be checked using: Correlation matrices, where correlation coefficients should ideally be below 0.80.

No extreme outliers: Logistic regression assumes that there are no extreme outliers or any external observations that influence the data that goes into the model. Cook's distance is an effective way to rule out the outliers and external observations from a dataset. You can choose to eradicate those from the data or decide to replace them with a mean or median. You can also let the outliers be, but remember to report those in the regression results.

Sample size: The logistic regression assumes that the sample size from which the observations are drawn is large enough to give reliable conclusions for the regression model.

https://www.voxco.com/blog/logistic-regression-assumptions/

2.

One benefit of python is that it is an interpreted language. There is no compile time, so it is much quicker for iterative processes such as the backward elimination process when we are reducing the regression model and reducing independent variables.

Another benefit of python language is that it has many libraries and packages that can automate the regression model creation process and simplify it to just a few lines of code. When it is time to compare the reduced model, the python packages can help us quickly compare the models by showing us important regression model metrics such as the p values of coefficients

3 .

Logistic regression is an appropriate technique to use for analyzing the research question in part 1 because the question we are answering involves predicting a binary categorical variable 'Churn'. Another reason Logistic regression is an appropriate technique is because part of the question involves identifying correlations between multiple predictor variables and one categorical dependent variable.

C.

1.

My data cleaning goals are as follows:

Identify any duplicate rows and remove them. I will do this by comparing rows by 'CaseOrder'. If there are any duplicates I will drop one of the duplicate rows.

Identify any missing values. I will use the df.isna() function to list columns with missing values. I will impute the values with different techniques depending on the data type and context of each column.

Identify any outliers. I will use z-scores, IQR tests and the describe() method to identify outliers. I will first use the describe() function to get an overview, and if further analysis is needed I can use z-scores and IQR tests to further identify outliers. If a value is clearly an outlier, it can be imputed from other values or the row dropped.

See cells below for further explanation of each step and annotated code.

In [1]:
```
#import libraries and read in the data from file.
import pandas as pd
from scipy.stats import zscore
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
# Assuming your CSV file is named 'data.csv', adjust the file path as needed
```

```python
file_path = '/home/dj/skewl/d208/churn_clean.csv'
pd.set_option('display.max_columns', None)
# Read the data from the CSV file into a DataFrame
df = pd.read_csv(file_path)
#drop index column
df = df.loc[:, ~df.columns.str.contains('Unnamed')]
```

In [2]:
```python
# helper functions

#function to plot histogram univariate
def plot_hist(col_name, num_bins, do_rotate=False):
    plt.hist(df[col_name], bins=num_bins)
    plt.xlabel(col_name)
    plt.ylabel('Frequency')
    plt.title(f'Histogram of {col_name}')
    if do_rotate:
        plt.xticks(rotation=90)
    plt.show()

def line_plot(indep):
    # hexbin plot for continuous variables
    plt.hexbin(df[indep], df['MonthlyCharge'], gridsize=10)
    plt.colorbar()
    plt.title('Hexbin Plot')
    plt.xlabel(indep)
    plt.ylabel('Churn')
    plt.show()

def cross_tab(col2):
    # Create a cross-tabulation of the two categorical variables
    cross_tab = pd.crosstab(df['Churn'], df[col2])
    # Plot the heatmap
    sns.heatmap(cross_tab, annot=True, cmap='Blues')
    plt.title(f'Heatmap of Churn and {col2}')
    plt.xlabel(col2)
    plt.ylabel('Churn')
    plt.show()

def box_plot(indep):
    # Box plot for categorical predictor and continuous outcome variable
    df.boxplot(column=indep, by='Churn')
    plt.title('Box Plot',y=.5)
    plt.xlabel("Churn")
    plt.ylabel(indep)
    plt.show()

def stacked_tab(col1):
```

```
    # Create a cross-tabulation of the two categorical variables
    cross_tab = pd.crosstab(df['Churn'], df[col1])
    # Plot the stacked bar plot
    cross_tab.plot.bar(stacked=True)
    plt.figure(figsize=(10, 6))
    plt.title(f'Stacked Bar Plot of Chrun and {col1}')
    plt.xlabel('Churn')
    plt.ylabel('Frequency')
    plt.show()
```

identify duplicate rows by 'CaseOrder'

In [3]:

```
# Find duplicate rows
duplicate_rows = df.duplicated(["CaseOrder"]).sum()

# Print duplicate rows   # found NO duplicate rows here!
print(duplicate_rows)
```

```
0
```

identify missing values

In [4]:

```
# Identify missing values using isna() method
missing_values = df.isna().sum()
# Print DataFrame with True for missing values and False for non-missing values
print(missing_values)

# no missing values here!
```

```
CaseOrder               0
Customer_id             0
Interaction             0
UID                     0
City                    0
State                   0
County                  0
Zip                     0
Lat                     0
Lng                     0
Population              0
Area                    0
TimeZone                0
Job                     0
Children                0
Age                     0
Income                  0
Marital                 0
Gender                  0
Churn                   0
Outage_sec_perweek      0
Email                   0
Contacts                0
Yearly_equip_failure    0
Techie                  0
Contract                0
Port_modem              0
Tablet                  0
InternetService         0
Phone                   0
Multiple                0
OnlineSecurity          0
OnlineBackup            0
DeviceProtection        0
TechSupport             0
StreamingTV             0
StreamingMovies         0
PaperlessBilling        0
PaymentMethod           0
Tenure                  0
MonthlyCharge           0
Bandwidth_GB_Year       0
Item1                   0
Item2                   0
Item3                   0
Item4                   0
Item5                   0
Item6                   0
```

```
Item7                0
Item8                0
dtype: int64
```

Check for outliers

In [5]:
```
# check for outliers. Doesn't seem to be any outliers.
df.describe()
```

Out[5]:

	CaseOrder	Zip	Lat	Lng	Population	Children	Age	Income	Outage_sec_pe
count	10000.00000	10000.000000	10000.000000	10000.000000	10000.000000	10000.0000	10000.000000	10000.000000	10000.0
mean	5000.50000	49153.319600	38.757567	-90.782536	9756.562400	2.0877	53.078400	39806.926771	10.0
std	2886.89568	27532.196108	5.437389	15.156142	14432.698671	2.1472	20.698882	28199.916702	2.9
min	1.00000	601.000000	17.966120	-171.688150	0.000000	0.0000	18.000000	348.670000	0.0
25%	2500.75000	26292.500000	35.341828	-97.082812	738.000000	0.0000	35.000000	19224.717500	8.0
50%	5000.50000	48869.500000	39.395800	-87.918800	2910.500000	1.0000	53.000000	33170.605000	10.0
75%	7500.25000	71866.500000	42.106908	-80.088745	13168.000000	3.0000	71.000000	53246.170000	11.9
max	10000.00000	99929.000000	70.640660	-65.667850	111850.000000	10.0000	89.000000	258900.700000	21.2

2. Describe dependent and independent variables

In [6]:
```
## dependent variable

df['Churn'].describe()
```

Out[6]:
```
count     10000
unique        2
top          No
freq       7350
Name: Churn, dtype: object
```

In [7]:
```
# independent variable
df['Gender'].describe()
```

Out[7]:
```
count      10000
unique         3
top       Female
freq        5025
Name: Gender, dtype: object
```

In [8]:
```
df['Area'].describe()
```

Out[8]:
```
count        10000
unique           3
top       Suburban
freq          3346
Name: Area, dtype: object
```

In [9]:
```
df['Age'].describe()
```

Out[9]:
```
count    10000.000000
mean        53.078400
std         20.698882
min         18.000000
25%         35.000000
50%         53.000000
75%         71.000000
max         89.000000
Name: Age, dtype: float64
```

In [10]:
```
df['Income'].describe()
```

Out[10]:
```
count     10000.000000
mean      39806.926771
std       28199.916702
min         348.670000
25%       19224.717500
50%       33170.605000
75%       53246.170000
max      258900.700000
Name: Income, dtype: float64
```

In [11]:
```
df['Outage_sec_perweek'].describe()
```

Out[11]:
```
count    10000.000000
mean        10.001848
std          2.976019
min          0.099747
25%          8.018214
50%         10.018560
75%         11.969485
max         21.207230
Name: Outage_sec_perweek, dtype: float64
```

In [12]:
```
df['InternetService'].describe()
```

Out[12]:
```
count           10000
unique              3
top       Fiber Optic
freq             4408
Name: InternetService, dtype: object
```

In [13]:
```
df['Phone'].describe()
```

Out[13]:
```
count     10000
unique        2
top         Yes
freq       9067
Name: Phone, dtype: object
```

In [14]:
```
df['OnlineSecurity'].describe()
```

Out[14]:
```
count     10000
unique        2
top          No
freq       6424
Name: OnlineSecurity, dtype: object
```

In [15]:
```
df['DeviceProtection'].describe()
```

Out[15]:
```
count     10000
unique        2
top          No
freq       5614
Name: DeviceProtection, dtype: object
```

In [16]:
```
df['StreamingMovies'].describe()
```

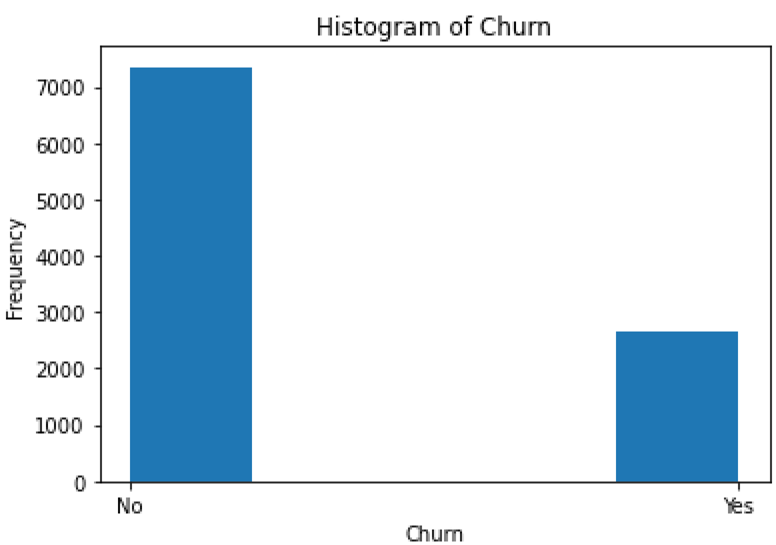
Out[16]:
```
count      10000
unique         2
top           No
freq        5110
Name: StreamingMovies, dtype: object
```

In [17]:
```
df['OnlineBackup'].describe()
```

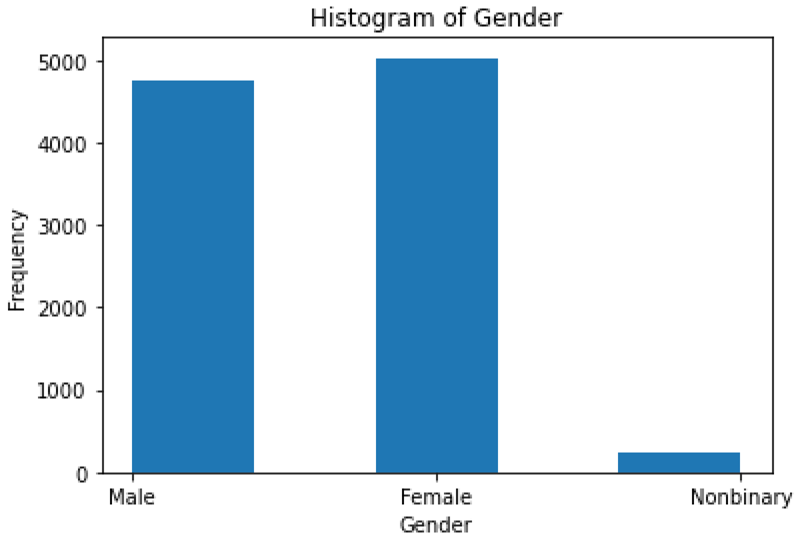
Out[17]:
```
count      10000
unique         2
top           No
freq        5494
Name: OnlineBackup, dtype: object
```

3. Generate univariate and bivariate visualizations of the distributions of the dependent and independent variables, including the dependent variable in your bivariate visualizations.

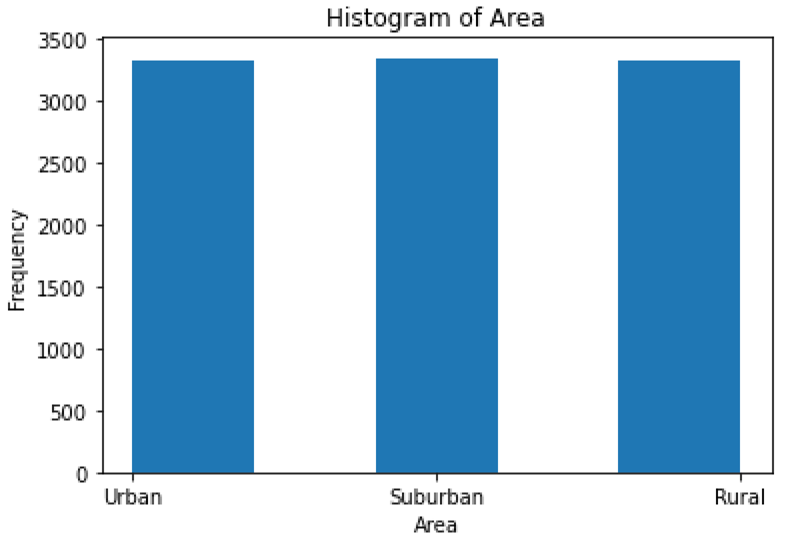
In [18]:
```
plot_hist('Churn',5)
```



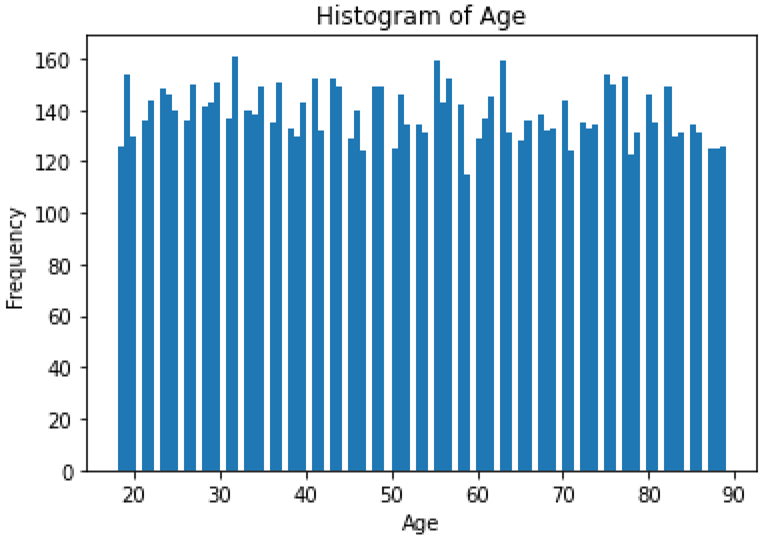
In [19]:
```
plot_hist('Gender',5)
```



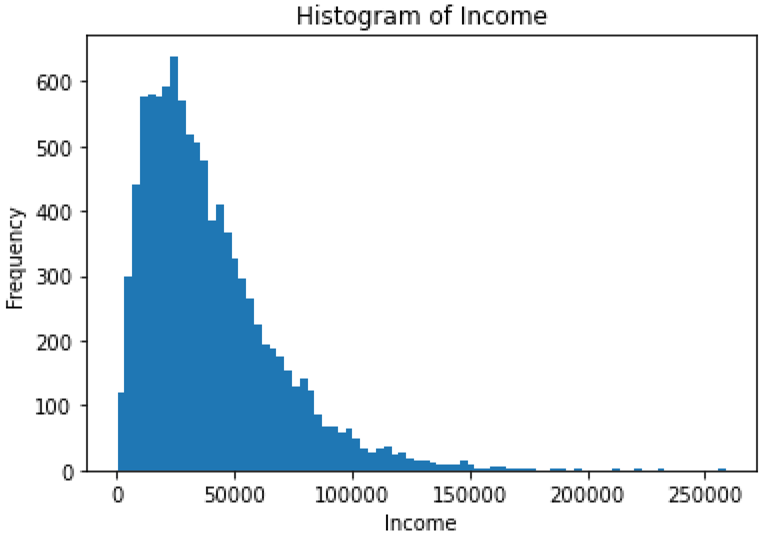
In [20]:
```
plot_hist('Area',5)
```



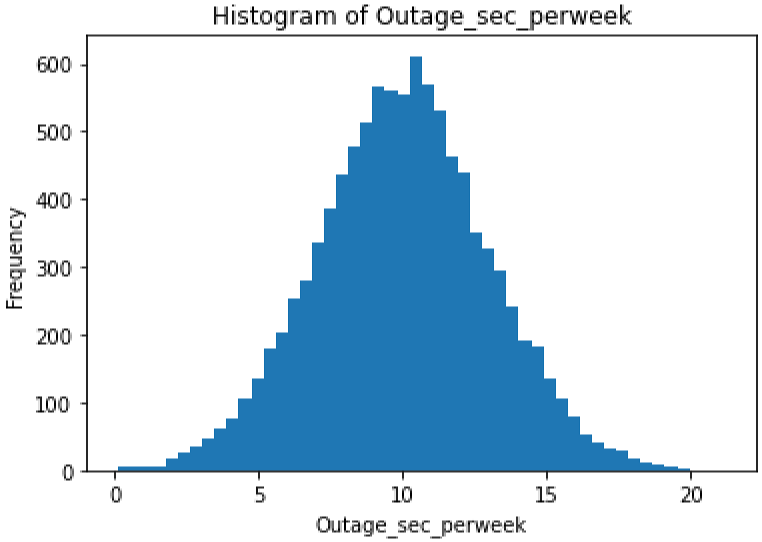
In [21]:
```
plot_hist('Age',100)
```



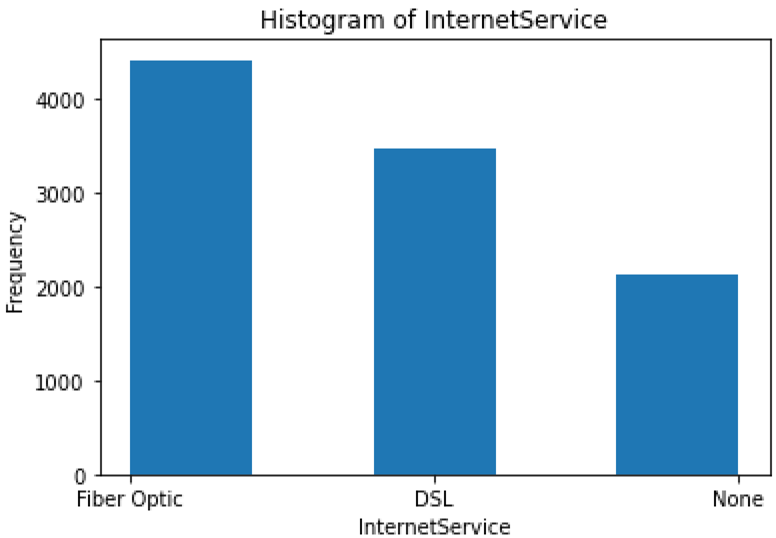
In [22]:
```
plot_hist('Income',80)
```



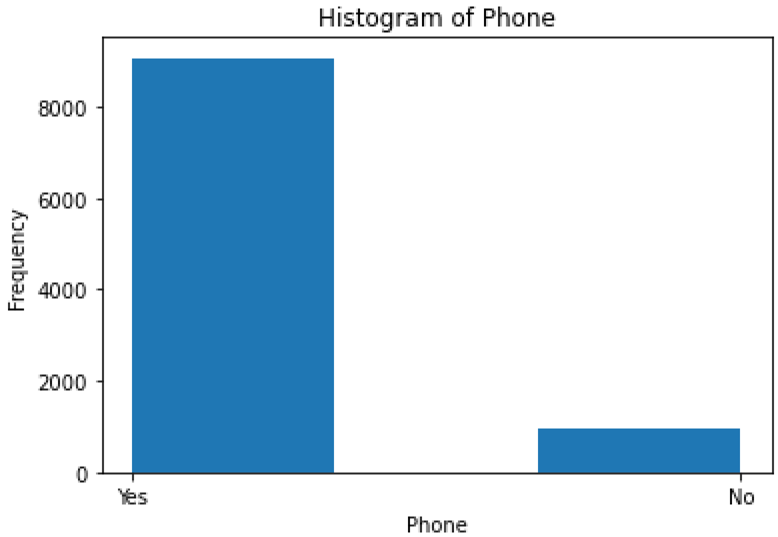
In [23]:
```
plot_hist('Outage_sec_perweek',50)
```



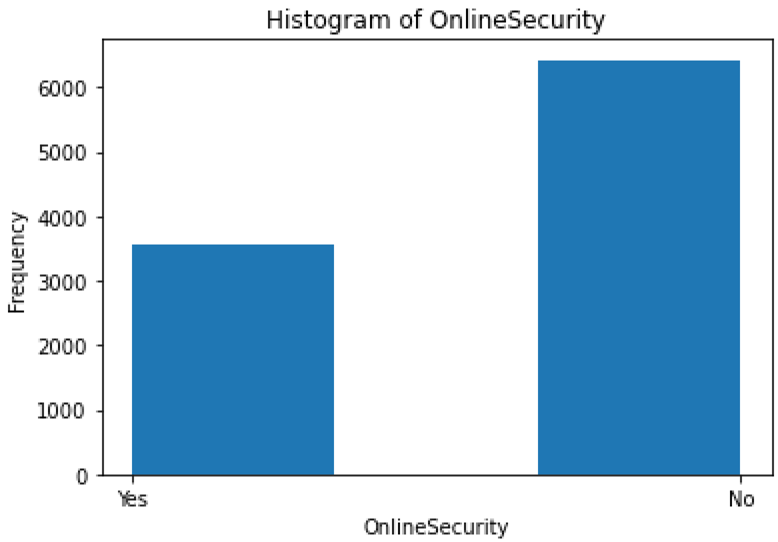
In [24]:
```
plot_hist('InternetService',5)
```



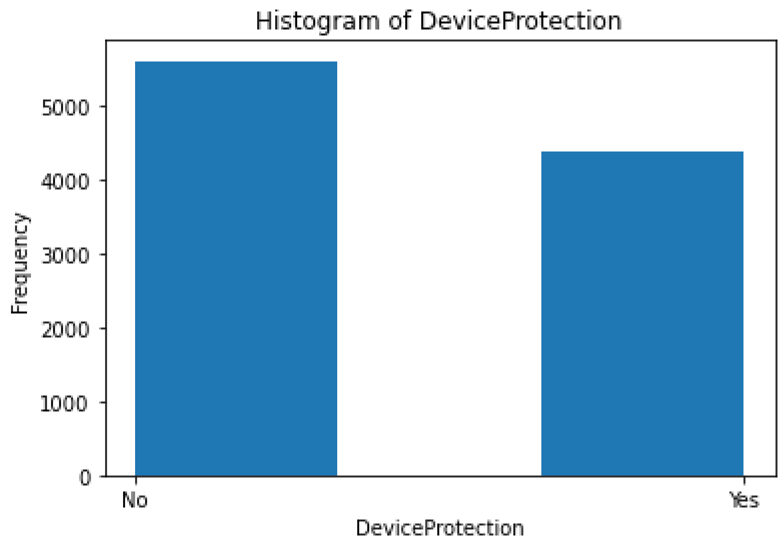
In [25]:
```
plot_hist('Phone',3)
```



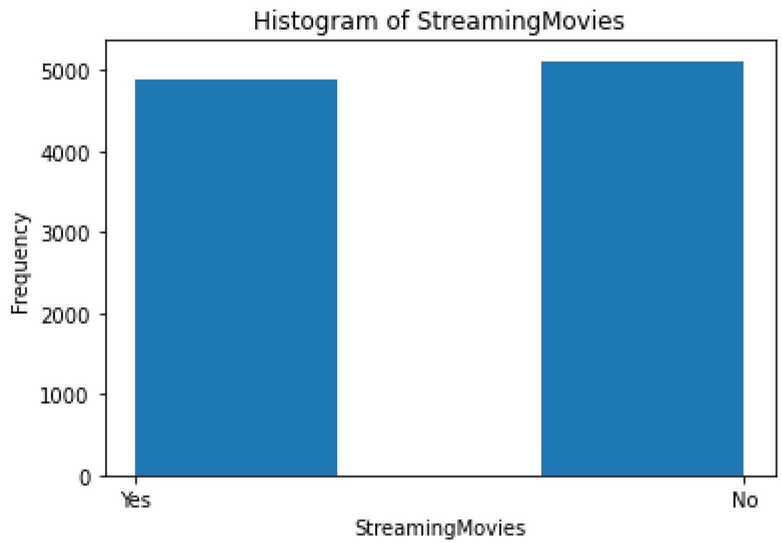
In [26]:
```
plot_hist('OnlineSecurity',3)
```



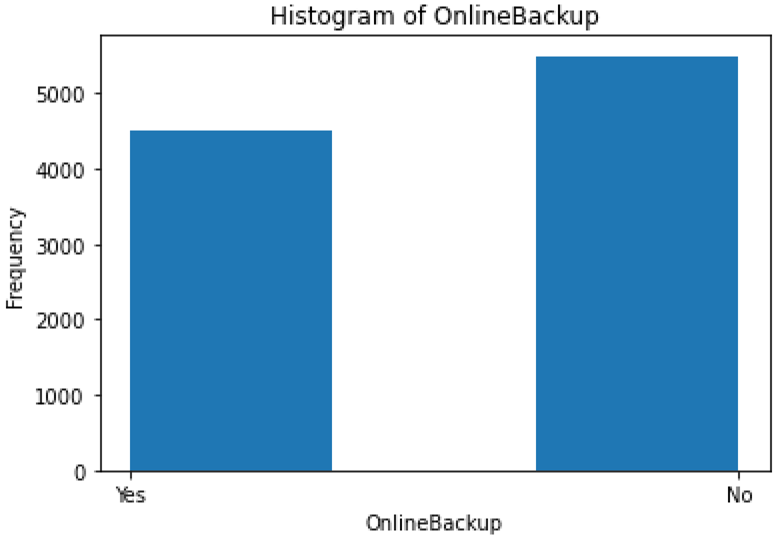
In [27]:
```
plot_hist('DeviceProtection',3)
```



In [28]:
```
plot_hist('StreamingMovies',3)
```

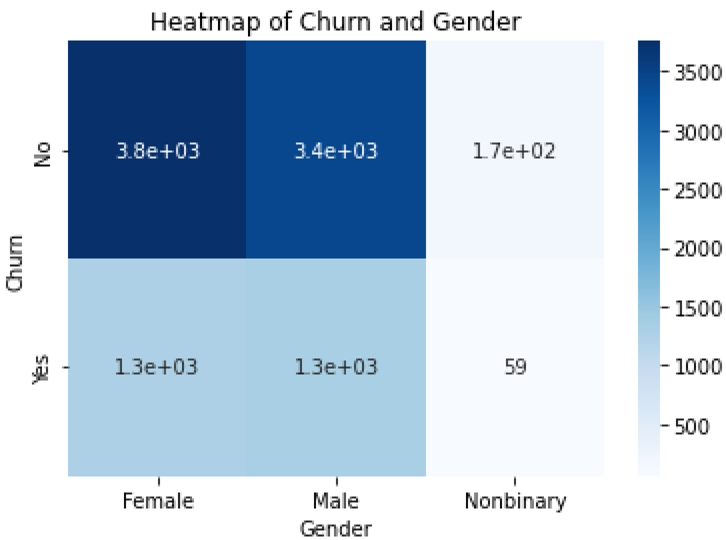


In [29]:
```
plot_hist('OnlineBackup',3)
```

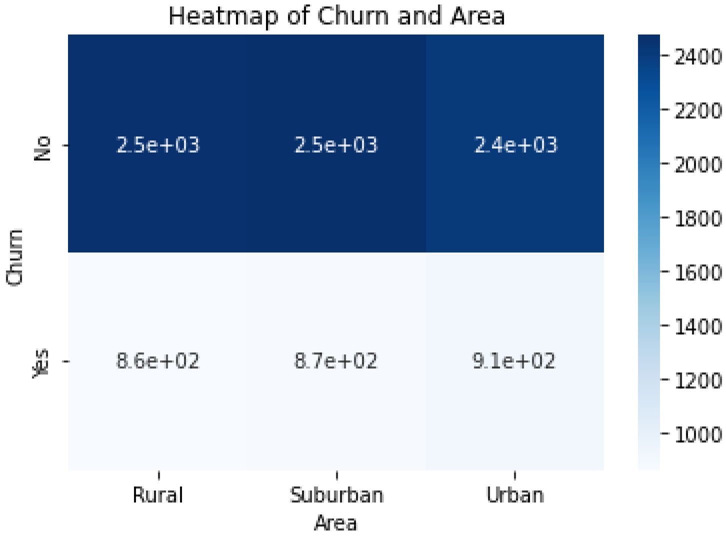


bivariate - graphing against the dependent variable

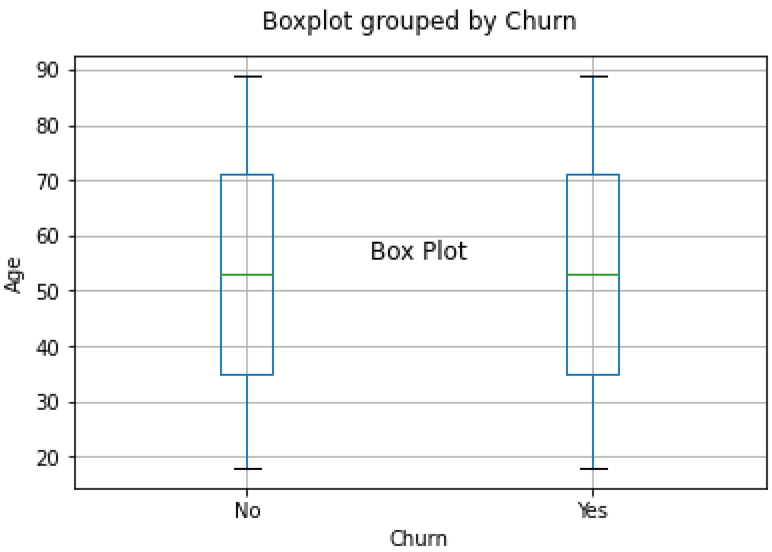
In [30]:
```
cross_tab('Gender')
```



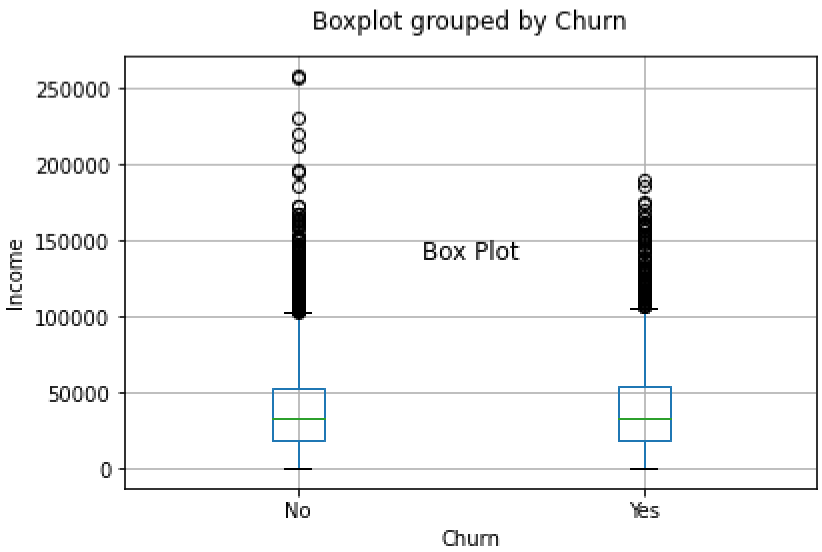
In [31]:
```
cross_tab('Area')
```



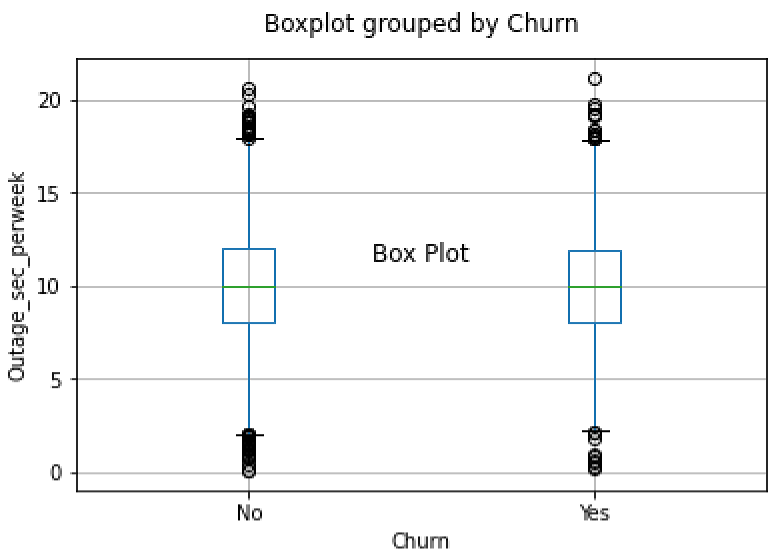
In [32]:
```
box_plot('Age')
```



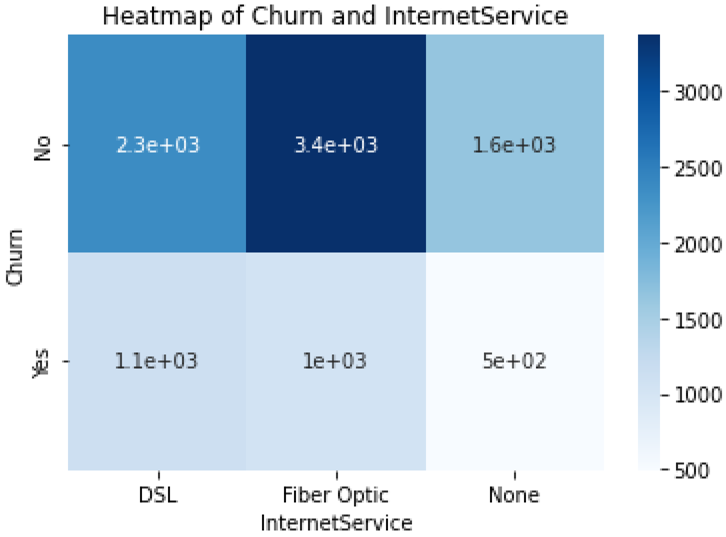
In [33]:
```
box_plot('Income')
```



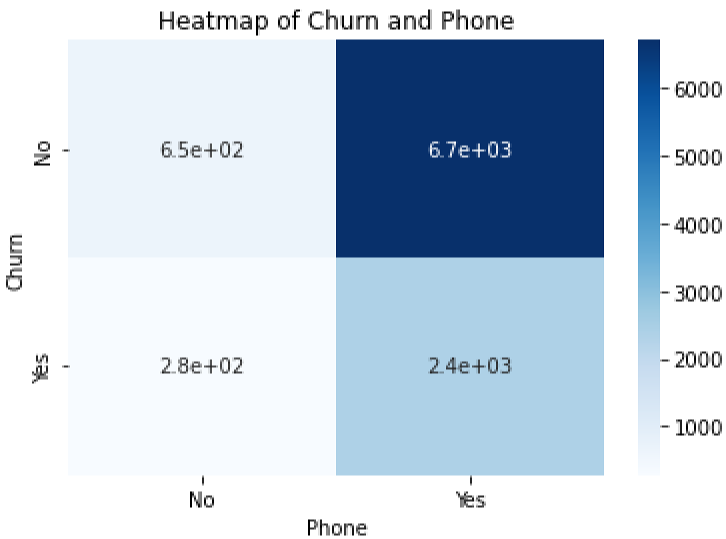
In [34]:
```
box_plot('Outage_sec_perweek')
```



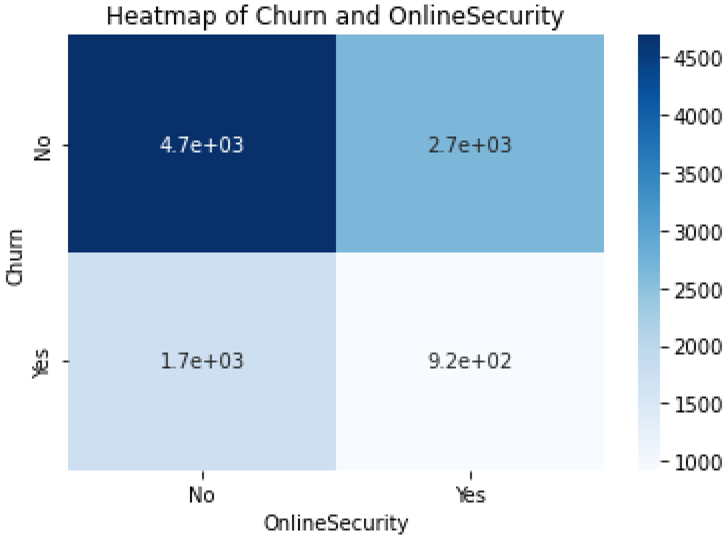
In [35]:
```
cross_tab('InternetService')
```



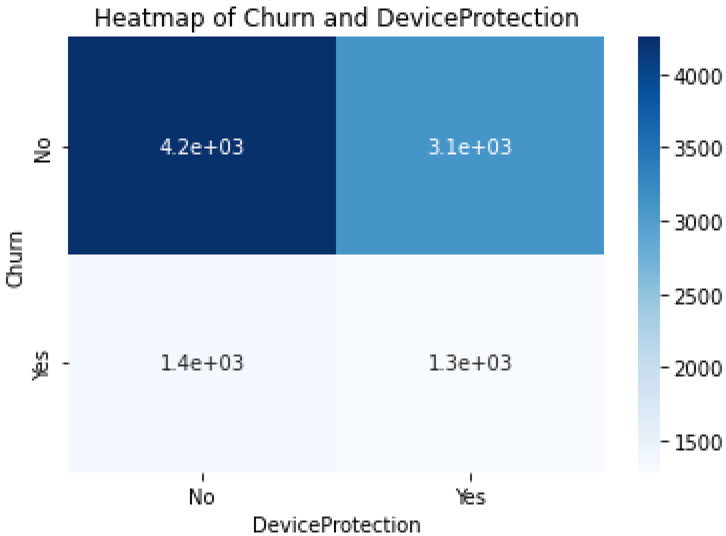
In [36]:
```
cross_tab('Phone')
```



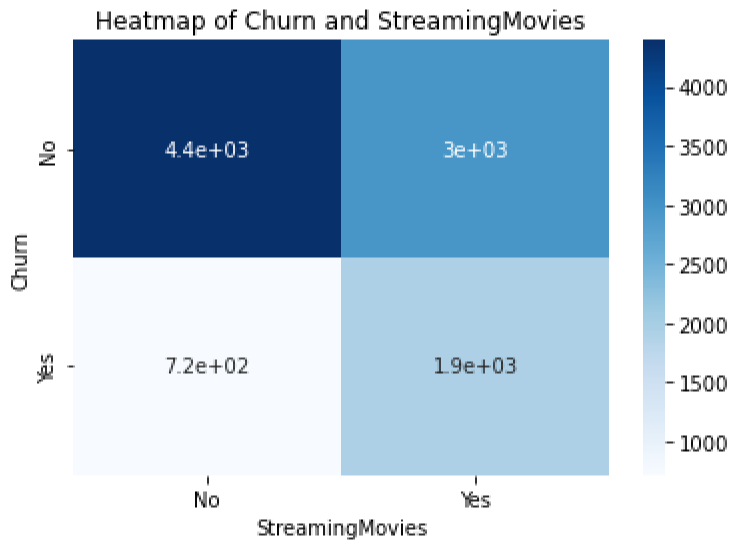
In [37]:
```
cross_tab('OnlineSecurity')
```



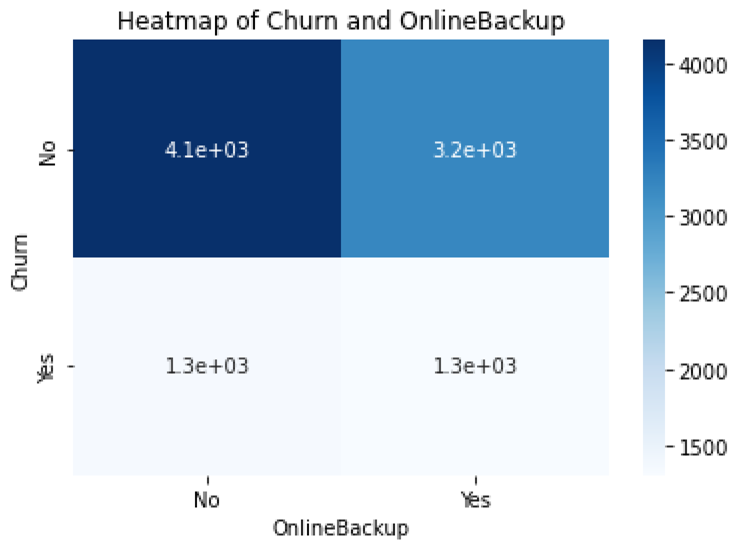
In [38]:
```
cross_tab('DeviceProtection')
```



In [39]:
```
cross_tab('StreamingMovies')
```




```
In [40]: cross_tab('OnlineBackup')
```



4)

My goals for data transformation are to one-hot encode the categorical variables. The dependent variable 'churn' will be mapped to binary values as well.

```
In [41]: import statsmodels.api as sm
from sklearn.model_selection import train_test_split
#split continuous and categorical variables into separate dataframes
dfcon = df[['Age','Income','Outage_sec_perweek']]
```

```
dfcat = df[['Gender','Area','InternetService','Phone','OnlineSecurity','DeviceProtection','StreamingMovies','OnlineBack
#one-hot encode categorical data and drop first level of each
dfcat_encoded = pd.get_dummies(dfcat,drop_first=True)
#concatenate the columns
data = pd.concat([dfcon, dfcat_encoded], axis=1)
# Convert categorical dependent variable to binary (0/1)
churn_binary=df['Churn'].map({'No': 0, 'Yes': 1})
data['Churn']= churn_binary
#write the prepared data to .csv file
data.to_csv('prepared-data2.csv', index=False)
#remove independent var from data set
del data['Churn']
independent_vars = sm.add_constant(data)
x_train, x_test, y_train, y_test = train_test_split(independent_vars, churn_binary, test_size=0.2, random_state=42)
```

D. Compare an initial and a reduced linear regression model

1. Construct an initial multiple linear regression model from all independent variables that were identified in part C2.

In [42]:

```
#Initial Model
independent_vars = sm.add_constant(data)
x_train, x_test, y_train, y_test = train_test_split(independent_vars, churn_binary, test_size=0.2, random_state=42)
model = sm.Logit(y_train,  x_train).fit()
print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.526410
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                  Churn   No. Observations:                 8000
Model:                          Logit   Df Residuals:                     7985
Method:                           MLE   Df Model:                           14
Date:                Sun, 14 Apr 2024   Pseudo R-squ.:                 0.08677
Time:                        12:35:17   Log-Likelihood:                -4211.3
converged:                       True   LL-Null:                       -4611.4
Covariance Type:            nonrobust   LLR p-value:                 9.510e-162
===============================================================================================
                                  coef    std err          z      P>|z|      [0.025      0.975]
-----------------------------------------------------------------------------------------------
const                          -1.5780      0.166     -9.483      0.000      -1.904      -1.252
Age                            -0.0003      0.001     -0.263      0.793      -0.003       0.002
Income                       8.264e-07   9.33e-07      0.886      0.376      -1e-06    2.65e-06
Outage_sec_perweek             -0.0041      0.009     -0.455      0.649      -0.022       0.013
Gender_Male                     0.1173      0.054      2.167      0.030       0.011       0.223
Gender_Nonbinary               -0.0104      0.186     -0.056      0.955      -0.375       0.354
Area_Suburban                   0.0115      0.065      0.176      0.861      -0.117       0.140
Area_Urban                      0.0260      0.066      0.396      0.692      -0.103       0.155
InternetService_Fiber Optic    -0.4363      0.060     -7.275      0.000      -0.554      -0.319
InternetService_None           -0.4694      0.074     -6.364      0.000      -0.614      -0.325
Phone_Yes                      -0.2390      0.088     -2.703      0.007      -0.412      -0.066
OnlineSecurity_Yes             -0.0422      0.056     -0.754      0.451      -0.152       0.067
DeviceProtection_Yes            0.2359      0.054      4.394      0.000       0.131       0.341
StreamingMovies_Yes             1.4019      0.056     24.895      0.000       1.292       1.512
OnlineBackup_Yes                0.2425      0.054      4.519      0.000       0.137       0.348
===============================================================================================
```

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

I have chosen to use backward elimination of predictor variables as my feature selection procedure. This is so I can iteratively choose which predictor variables I want to keep based on p values. I can also see how removing each predictor variable one at a time will effect the model metrics such as psuedo R squared and log-likelihood.

I have chosen to use the log-likelihood for a model evaluation metric because it measures the 'goodness of fit' of the model, which means the likelihood that the model will predict the correct outcome.

With a model that includes predictor variables with low p values, and a high log-likehood measurement we can more accurately predict 'Churn' and understand how each variable is correlated with 'Churn'.

3. Provide a reduced linear logistic regression model that follows the feature selection or model evaluation process in part D2, including a screenshot of the output for each model.

In [43]:
```
#original model
independent_vars = sm.add_constant(data)
x_train, x_test, y_train, y_test = train_test_split(independent_vars, churn_binary, test_size=0.2, random_state=42)
model = sm.Logit(y_train, x_train).fit()
print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.526410
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                  Churn   No. Observations:                 8000
Model:                          Logit   Df Residuals:                     7985
Method:                           MLE   Df Model:                           14
Date:                Sun, 14 Apr 2024   Pseudo R-squ.:                 0.08677
Time:                        12:35:17   Log-Likelihood:                -4211.3
converged:                       True   LL-Null:                       -4611.4
Covariance Type:            nonrobust   LLR p-value:                9.510e-162
==================================================================================================
                                   coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------------------------
const                           -1.5780      0.166     -9.483      0.000      -1.904      -1.252
Age                             -0.0003      0.001     -0.263      0.793      -0.003       0.002
Income                        8.264e-07   9.33e-07      0.886      0.376      -1e-06    2.65e-06
Outage_sec_perweek              -0.0041      0.009     -0.455      0.649      -0.022       0.013
Gender_Male                      0.1173      0.054      2.167      0.030       0.011       0.223
Gender_Nonbinary                -0.0104      0.186     -0.056      0.955      -0.375       0.354
Area_Suburban                    0.0115      0.065      0.176      0.861      -0.117       0.140
Area_Urban                       0.0260      0.066      0.396      0.692      -0.103       0.155
InternetService_Fiber Optic     -0.4363      0.060     -7.275      0.000      -0.554      -0.319
InternetService_None            -0.4694      0.074     -6.364      0.000      -0.614      -0.325
Phone_Yes                       -0.2390      0.088     -2.703      0.007      -0.412      -0.066
OnlineSecurity_Yes              -0.0422      0.056     -0.754      0.451      -0.152       0.067
DeviceProtection_Yes             0.2359      0.054      4.394      0.000       0.131       0.341
StreamingMovies_Yes              1.4019      0.056     24.895      0.000       1.292       1.512
OnlineBackup_Yes                 0.2425      0.054      4.519      0.000       0.137       0.348
==================================================================================================
```

Reduced model

In [44]:
```
#Reduced model
df_reduced =  independent_vars.copy()
del df_reduced['Area_Urban']
del df_reduced['Area_Suburban']
del df_reduced['Age']
del df_reduced['Gender_Male']
del df_reduced['Outage_sec_perweek']
del df_reduced['Gender_Nonbinary']
del df_reduced['OnlineSecurity_Yes']
del df_reduced['Income']
x_train, x_test, y_train, y_test = train_test_split(df_reduced, churn_binary, test_size=0.2, random_state=42)
model = sm.Logit(y_train, x_train).fit()
print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.526815
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                  Churn   No. Observations:                 8000
Model:                          Logit   Df Residuals:                     7993
Method:                           MLE   Df Model:                            6
Date:                Sun, 14 Apr 2024   Pseudo R-squ.:                 0.08607
Time:                        12:35:17   Log-Likelihood:                -4214.5
converged:                       True   LL-Null:                       -4611.4
Covariance Type:            nonrobust   LLR p-value:                 3.331e-168
===============================================================================================
                                  coef    std err          z      P>|z|      [0.025      0.975]
-----------------------------------------------------------------------------------------------
const                          -1.5472      0.104    -14.942      0.000      -1.750      -1.344
InternetService_Fiber Optic    -0.4379      0.060     -7.307      0.000      -0.555      -0.320
InternetService_None           -0.4677      0.074     -6.348      0.000      -0.612      -0.323
Phone_Yes                      -0.2398      0.088     -2.715      0.007      -0.413      -0.067
DeviceProtection_Yes            0.2377      0.054      4.434      0.000       0.133       0.343
StreamingMovies_Yes             1.4007      0.056     24.896      0.000       1.290       1.511
OnlineBackup_Yes                0.2391      0.054      4.464      0.000       0.134       0.344
===============================================================================================
```

E.

1.Explain your data analysis process by comparing the initial logistic regression model and reduced logistic regression model

My model evaluation metrics are log-likelihood and psuedo r squared. I used backwards elimination by P value to reduce the model. Since I had predictor variables that had large coefficients, the psuedo R squared value was about the same in both models. Log-likelihood didn't change much either. This is because the predictor variables with the largest coefficients and smallest P values were not removed.

Original log-likelihood = -4211.3

Reduced model log-likelihood = -4214.5

Original psuedo R squared = .08677

Reduced model psuedo R squared = .08607

The reduced model is just slightly worse in terms of log-likelihood and psuedo R squared metric. The difference is very small and the model has much fewer independent variables, so this may be a worthwhile trade off.

2. Provide the output and all calculations of the analysis you performed, including the following elements for your reduced logistic regression model

In [45]:

```
#calculations to reduce original model
df_reduced = independent_vars.copy()
df_reduced = sm.add_constant(df_reduced)
x_train, x_test, y_train, y_test = train_test_split(df_reduced, churn_binary, test_size=0.2, random_state=42)
model = sm.Logit(y_train, x_train).fit()
print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.526410
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                  Churn   No. Observations:                 8000
Model:                          Logit   Df Residuals:                     7985
Method:                           MLE   Df Model:                           14
Date:                Sun, 14 Apr 2024   Pseudo R-squ.:                 0.08677
Time:                        12:35:17   Log-Likelihood:                -4211.3
converged:                       True   LL-Null:                       -4611.4
Covariance Type:            nonrobust   LLR p-value:                 9.510e-162
================================================================================================
                                   coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------------------------
const                           -1.5780      0.166     -9.483      0.000      -1.904      -1.252
Age                             -0.0003      0.001     -0.263      0.793      -0.003       0.002
Income                        8.264e-07   9.33e-07      0.886      0.376      -1e-06    2.65e-06
Outage_sec_perweek              -0.0041      0.009     -0.455      0.649      -0.022       0.013
Gender_Male                      0.1173      0.054      2.167      0.030       0.011       0.223
Gender_Nonbinary                -0.0104      0.186     -0.056      0.955      -0.375       0.354
Area_Suburban                    0.0115      0.065      0.176      0.861      -0.117       0.140
Area_Urban                       0.0260      0.066      0.396      0.692      -0.103       0.155
InternetService_Fiber Optic     -0.4363      0.060     -7.275      0.000      -0.554      -0.319
InternetService_None            -0.4694      0.074     -6.364      0.000      -0.614      -0.325
Phone_Yes                       -0.2390      0.088     -2.703      0.007      -0.412      -0.066
OnlineSecurity_Yes              -0.0422      0.056     -0.754      0.451      -0.152       0.067
DeviceProtection_Yes             0.2359      0.054      4.394      0.000       0.131       0.341
StreamingMovies_Yes              1.4019      0.056     24.895      0.000       1.292       1.512
OnlineBackup_Yes                 0.2425      0.054      4.519      0.000       0.137       0.348
================================================================================================
```

Area_Urban   P 0.692   > .05

In [46]:

```python
#calculations to reduce original model
df_reduced = independent_vars.copy()
del df_reduced['Area_Urban']
df_reduced = sm.add_constant(df_reduced)
x_train, x_test, y_train, y_test = train_test_split(df_reduced, churn_binary, test_size=0.2, random_state=42)
model = sm.Logit(y_train, x_train).fit()
print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.526420
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                  Churn   No. Observations:                 8000
Model:                          Logit   Df Residuals:                     7986
Method:                           MLE   Df Model:                           13
Date:                Sun, 14 Apr 2024   Pseudo R-squ.:                 0.08676
Time:                        12:35:17   Log-Likelihood:                -4211.4
converged:                       True   LL-Null:                       -4611.4
Covariance Type:            nonrobust   LLR p-value:                1.283e-162
===============================================================================================
                                  coef    std err          z      P>|z|      [0.025      0.975]
-----------------------------------------------------------------------------------------------
const                          -1.5641      0.163     -9.614      0.000      -1.883      -1.245
Age                            -0.0003      0.001     -0.265      0.791      -0.003       0.002
Income                       8.263e-07   9.33e-07      0.886      0.376      -1e-06    2.65e-06
Outage_sec_perweek             -0.0041      0.009     -0.453      0.651      -0.022       0.014
Gender_Male                     0.1174      0.054      2.169      0.030       0.011       0.223
Gender_Nonbinary               -0.0113      0.186     -0.061      0.951      -0.376       0.353
Area_Suburban                  -0.0015      0.057     -0.027      0.979      -0.112       0.109
InternetService_Fiber Optic    -0.4362      0.060     -7.274      0.000      -0.554      -0.319
InternetService_None           -0.4692      0.074     -6.363      0.000      -0.614      -0.325
Phone_Yes                      -0.2399      0.088     -2.714      0.007      -0.413      -0.067
OnlineSecurity_Yes             -0.0423      0.056     -0.755      0.450      -0.152       0.067
DeviceProtection_Yes            0.2361      0.054      4.397      0.000       0.131       0.341
StreamingMovies_Yes             1.4018      0.056     24.893      0.000       1.291       1.512
OnlineBackup_Yes                0.2425      0.054      4.518      0.000       0.137       0.348
===============================================================================================
```

Area_Suburban P 0.979 > .05

In [47]:

```python
#calculations to reduce original model
df_reduced = independent_vars.copy()
del df_reduced['Area_Urban']
del df_reduced['Area_Suburban']
df_reduced = sm.add_constant(df_reduced)
x_train, x_test, y_train, y_test = train_test_split(df_reduced, churn_binary, test_size=0.2, random_state=42)
model = sm.Logit(y_train, x_train).fit()
print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.526420
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                  Churn   No. Observations:                 8000
Model:                          Logit   Df Residuals:                     7987
Method:                           MLE   Df Model:                           12
Date:                Sun, 14 Apr 2024   Pseudo R-squ.:                 0.08676
Time:                        12:35:17   Log-Likelihood:                -4211.4
converged:                       True   LL-Null:                       -4611.4
Covariance Type:            nonrobust   LLR p-value:                 1.538e-163
================================================================================================
                                  coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------------------------
const                          -1.5647      0.161     -9.695      0.000      -1.881      -1.248
Age                            -0.0003      0.001     -0.265      0.791      -0.003       0.002
Income                       8.263e-07   9.33e-07      0.886      0.376      -1e-06    2.65e-06
Outage_sec_perweek             -0.0041      0.009     -0.453      0.651      -0.022       0.014
Gender_Male                     0.1174      0.054      2.170      0.030       0.011       0.223
Gender_Nonbinary               -0.0114      0.186     -0.061      0.951      -0.376       0.353
InternetService_Fiber Optic    -0.4362      0.060     -7.274      0.000      -0.554      -0.319
InternetService_None           -0.4692      0.074     -6.363      0.000      -0.614      -0.325
Phone_Yes                      -0.2399      0.088     -2.714      0.007      -0.413      -0.067
OnlineSecurity_Yes             -0.0423      0.056     -0.756      0.450      -0.152       0.067
DeviceProtection_Yes            0.2361      0.054      4.398      0.000       0.131       0.341
StreamingMovies_Yes             1.4017      0.056     24.894      0.000       1.291       1.512
OnlineBackup_Yes                0.2425      0.054      4.519      0.000       0.137       0.348
================================================================================================
```

Age P 0.791 > .05

In [48]:
```python
#calculations to reduce original model
df_reduced = independent_vars.copy()
del df_reduced['Area_Urban']
del df_reduced['Area_Suburban']
del df_reduced['Age']
df_reduced = sm.add_constant(df_reduced)
x_train, x_test, y_train, y_test = train_test_split(df_reduced, churn_binary, test_size=0.2, random_state=42)
model = sm.Logit(y_train, x_train).fit()
print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.526424
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                  Churn   No. Observations:                 8000
Model:                          Logit   Df Residuals:                     7988
Method:                           MLE   Df Model:                           11
Date:                Sun, 14 Apr 2024   Pseudo R-squ.:                 0.08675
Time:                        12:35:18   Log-Likelihood:                -4211.4
converged:                       True   LL-Null:                       -4611.4
Covariance Type:            nonrobust   LLR p-value:                1.823e-164
===============================================================================================
                                  coef    std err          z      P>|z|      [0.025      0.975]
-----------------------------------------------------------------------------------------------
const                          -1.5829      0.146    -10.831      0.000      -1.869      -1.296
Income                       8.253e-07   9.33e-07      0.885      0.376      -1e-06    2.65e-06
Outage_sec_perweek             -0.0040      0.009     -0.451      0.652      -0.022       0.014
Gender_Male                     0.1175      0.054      2.173      0.030       0.012       0.224
Gender_Nonbinary               -0.0105      0.186     -0.056      0.955      -0.375       0.354
InternetService_Fiber Optic    -0.4363      0.060     -7.275      0.000      -0.554      -0.319
InternetService_None           -0.4691      0.074     -6.362      0.000      -0.614      -0.325
Phone_Yes                      -0.2400      0.088     -2.715      0.007      -0.413      -0.067
OnlineSecurity_Yes             -0.0421      0.056     -0.753      0.452      -0.152       0.068
DeviceProtection_Yes            0.2360      0.054      4.396      0.000       0.131       0.341
StreamingMovies_Yes             1.4016      0.056     24.893      0.000       1.291       1.512
OnlineBackup_Yes                0.2424      0.054      4.517      0.000       0.137       0.348
===============================================================================================
```

Gender_Male P 0.030   > .05

In [49]:
```
#calculations to reduce original model
df_reduced = independent_vars.copy()
del df_reduced['Area_Urban']
del df_reduced['Area_Suburban']
del df_reduced['Age']
del df_reduced['Gender_Male']
df_reduced = sm.add_constant(df_reduced)
x_train, x_test, y_train, y_test = train_test_split(df_reduced, churn_binary, test_size=0.2, random_state=42)
model = sm.Logit(y_train, x_train).fit()
print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.526719
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                  Churn   No. Observations:                 8000
Model:                          Logit   Df Residuals:                     7989
Method:                           MLE   Df Model:                           10
Date:                Sun, 14 Apr 2024   Pseudo R-squ.:                 0.08624
Time:                        12:35:18   Log-Likelihood:                -4213.8
converged:                       True   LL-Null:                       -4611.4
Covariance Type:            nonrobust   LLR p-value:                2.056e-164
===============================================================================================
                                  coef    std err          z      P>|z|      [0.025      0.975]
-----------------------------------------------------------------------------------------------
const                          -1.5236      0.143    -10.621      0.000      -1.805      -1.242
Income                        7.64e-07   9.32e-07      0.820      0.412   -1.06e-06    2.59e-06
Outage_sec_perweek             -0.0040      0.009     -0.445      0.656      -0.022       0.014
Gender_Nonbinary               -0.0681      0.184     -0.370      0.711      -0.429       0.293
InternetService_Fiber Optic    -0.4368      0.060     -7.286      0.000      -0.554      -0.319
InternetService_None           -0.4683      0.074     -6.354      0.000      -0.613      -0.324
Phone_Yes                      -0.2403      0.088     -2.720      0.007      -0.413      -0.067
OnlineSecurity_Yes             -0.0405      0.056     -0.724      0.469      -0.150       0.069
DeviceProtection_Yes            0.2380      0.054      4.436      0.000       0.133       0.343
StreamingMovies_Yes             1.4017      0.056     24.903      0.000       1.291       1.512
OnlineBackup_Yes                0.2413      0.054      4.499      0.000       0.136       0.346
===============================================================================================
```

Gender_Nonbinary P 0.955 > .05

In [50]:

```
#calculations to reduce original model
df_reduced = independent_vars.copy()
del df_reduced['Area_Urban']
del df_reduced['Area_Suburban']
del df_reduced['Age']
del df_reduced['Gender_Male']
del df_reduced['Gender_Nonbinary']
df_reduced = sm.add_constant(df_reduced)
x_train, x_test, y_train, y_test = train_test_split(df_reduced, churn_binary, test_size=0.2, random_state=42)
model = sm.Logit(y_train, x_train).fit()
print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.526728
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                  Churn   No. Observations:                 8000
Model:                          Logit   Df Residuals:                     7990
Method:                           MLE   Df Model:                            9
Date:                Sun, 14 Apr 2024   Pseudo R-squ.:                 0.08622
Time:                        12:35:18   Log-Likelihood:                -4213.8
converged:                       True   LL-Null:                       -4611.4
Covariance Type:            nonrobust   LLR p-value:                2.276e-165
===============================================================================================
                                coef    std err          z      P>|z|      [0.025      0.975]
-----------------------------------------------------------------------------------------------
const                        -1.5247      0.143    -10.631      0.000      -1.806      -1.244
Income                     7.608e-07   9.32e-07      0.816      0.414   -1.07e-06    2.59e-06
Outage_sec_perweek           -0.0040      0.009     -0.446      0.656      -0.022       0.014
InternetService_Fiber Optic  -0.4365      0.060     -7.281      0.000      -0.554      -0.319
InternetService_None         -0.4681      0.074     -6.351      0.000      -0.613      -0.324
Phone_Yes                    -0.2406      0.088     -2.724      0.006      -0.414      -0.067
OnlineSecurity_Yes           -0.0403      0.056     -0.721      0.471      -0.150       0.069
DeviceProtection_Yes          0.2384      0.054      4.445      0.000       0.133       0.344
StreamingMovies_Yes           1.4018      0.056     24.905      0.000       1.291       1.512
OnlineBackup_Yes              0.2405      0.054      4.488      0.000       0.135       0.346
===============================================================================================
```

OnlineSecurity_Yes P 0.471 > .05

In [51]:

```
#calculations to reduce original model
df_reduced = independent_vars.copy()
del df_reduced['Area_Urban']
del df_reduced['Area_Suburban']
del df_reduced['Age']
del df_reduced['Gender_Male']
del df_reduced['Gender_Nonbinary']
del df_reduced['OnlineSecurity_Yes']
df_reduced = sm.add_constant(df_reduced)
x_train, x_test, y_train, y_test = train_test_split(df_reduced, churn_binary, test_size=0.2, random_state=42)
model = sm.Logit(y_train, x_train).fit()
print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.526760
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                  Churn   No. Observations:                 8000
Model:                          Logit   Df Residuals:                     7991
Method:                           MLE   Df Model:                            8
Date:                Sun, 14 Apr 2024   Pseudo R-squ.:                 0.08617
Time:                        12:35:18   Log-Likelihood:                -4214.1
converged:                       True   LL-Null:                       -4611.4
Covariance Type:            nonrobust   LLR p-value:                2.860e-166
===============================================================================================
                                  coef    std err          z      P>|z|      [0.025      0.975]
-----------------------------------------------------------------------------------------------
const                          -1.5392      0.142    -10.838      0.000      -1.818      -1.261
Income                        7.65e-07   9.32e-07      0.821      0.412   -1.06e-06    2.59e-06
Outage_sec_perweek             -0.0040      0.009     -0.442      0.659      -0.022       0.014
InternetService_Fiber Optic    -0.4367      0.060     -7.286      0.000      -0.554      -0.319
InternetService_None           -0.4673      0.074     -6.342      0.000      -0.612      -0.323
Phone_Yes                      -0.2401      0.088     -2.718      0.007      -0.413      -0.067
DeviceProtection_Yes            0.2377      0.054      4.432      0.000       0.133       0.343
StreamingMovies_Yes             1.4011      0.056     24.898      0.000       1.291       1.511
OnlineBackup_Yes                0.2402      0.054      4.483      0.000       0.135       0.345
===============================================================================================
```

Outage_sec_perweek P 0.659   > .05

In [52]:

```python
#calculations to reduce original model
df_reduced = independent_vars.copy()
del df_reduced['Area_Urban']
del df_reduced['Area_Suburban']
del df_reduced['Age']
del df_reduced['Gender_Male']
del df_reduced['Gender_Nonbinary']
del df_reduced['OnlineSecurity_Yes']
del df_reduced['Outage_sec_perweek']
df_reduced = sm.add_constant(df_reduced)
x_train, x_test, y_train, y_test = train_test_split(df_reduced, churn_binary, test_size=0.2, random_state=42)
model = sm.Logit(y_train, x_train).fit()
print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.526773
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                  Churn   No. Observations:                 8000
Model:                          Logit   Df Residuals:                     7992
Method:                           MLE   Df Model:                            7
Date:                Sun, 14 Apr 2024   Pseudo R-squ.:                 0.08614
Time:                        12:35:18   Log-Likelihood:                -4214.2
converged:                       True   LL-Null:                       -4611.4
Covariance Type:            nonrobust   LLR p-value:                2.851e-167
===============================================================================================
                                coef    std err          z      P>|z|      [0.025      0.975]
-----------------------------------------------------------------------------------------------
const                        -1.5787      0.110    -14.298      0.000      -1.795      -1.362
Income                     7.705e-07   9.32e-07      0.827      0.408   -1.06e-06     2.6e-06
InternetService_Fiber Optic  -0.4370      0.060     -7.291      0.000      -0.554      -0.320
InternetService_None         -0.4675      0.074     -6.345      0.000      -0.612      -0.323
Phone_Yes                    -0.2397      0.088     -2.713      0.007      -0.413      -0.067
DeviceProtection_Yes          0.2372      0.054      4.423      0.000       0.132       0.342
StreamingMovies_Yes           1.4008      0.056     24.895      0.000       1.290       1.511
OnlineBackup_Yes              0.2401      0.054      4.481      0.000       0.135       0.345
===============================================================================================
```

Income P 0.408 > .05 Done. Final reduced model

In [53]:
```python
#calculations to reduce original model
df_reduced = independent_vars.copy()
del df_reduced['Area_Urban']
del df_reduced['Area_Suburban']   #final reduced model
del df_reduced['Age']
del df_reduced['Gender_Male']
del df_reduced['Gender_Nonbinary']
del df_reduced['OnlineSecurity_Yes']
del df_reduced['Outage_sec_perweek']
del df_reduced['Income']
df_reduced = sm.add_constant(df_reduced)
x_train, x_test, y_train, y_test = train_test_split(df_reduced, churn_binary, test_size=0.2, random_state=42)
model = sm.Logit(y_train, x_train).fit()
print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.526815
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                  Churn   No. Observations:                 8000
Model:                          Logit   Df Residuals:                     7993
Method:                           MLE   Df Model:                            6
Date:                Sun, 14 Apr 2024   Pseudo R-squ.:                 0.08607
Time:                        12:35:18   Log-Likelihood:                -4214.5
converged:                       True   LL-Null:                       -4611.4
Covariance Type:            nonrobust   LLR p-value:                3.331e-168
================================================================================================
                                  coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------------------------
const                          -1.5472      0.104    -14.942      0.000      -1.750      -1.344
InternetService_Fiber Optic    -0.4379      0.060     -7.307      0.000      -0.555      -0.320
InternetService_None           -0.4677      0.074     -6.348      0.000      -0.612      -0.323
Phone_Yes                      -0.2398      0.088     -2.715      0.007      -0.413      -0.067
DeviceProtection_Yes            0.2377      0.054      4.434      0.000       0.133       0.343
StreamingMovies_Yes             1.4007      0.056     24.896      0.000       1.290       1.511
OnlineBackup_Yes                0.2391      0.054      4.464      0.000       0.134       0.344
================================================================================================
```

confusion matrix and accuracy calculation

In [54]:

```python
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
#get predictions
y_pred = model.predict(x_test)

# Compute confusion matrix
y_pred = (y_pred >= .5).astype(int)
conf_matrix = confusion_matrix(y_test, y_pred)

print("Confusion Matrix:")
print(conf_matrix)
print('=======================')
print(f"accuracy=  {accuracy_score(y_test, y_pred, normalize=True)}")
```

```
Confusion Matrix:
[[1415   41]
 [ 497   47]]
=======================
accuracy=  0.731
```

3. code will be submitted with assignment.

F.

1. Discuss the results of your data analysis

regression equation :

$\log(p/1\text{-}p) = -1.7528 + (-0.4358)(X) + (-0.4681)(X) + (-0.0393)(x) + 0.2418(x) + 1.4010(X) + 0.2402(X)$

Interpretation of coefficients:

The coefficient itself is the magnitude which represents the strength of the relationship.

The sign tells us if the relationship is negative or positive to the log odds of the dependent variable.

All these coefficients have p values < .05 so they are statistically significant.

	coef	std err	z	P>|z|	[0.025	0.975]
const	-1.5785	0.110	-14.310	0.000	-1.795	-1.362
InternetService_Fiber Optic	-0.4370	0.060	-7.291	0.000	-0.554	-0.320
InternetService_None	-0.4675	0.074	-6.345	0.000	-0.612	-0.323
Phone_Yes	-0.2397	0.088	-2.713	0.007	-0.413	-0.067
DeviceProtection_Yes	0.2372	0.054	4.423	0.000	0.132	0.342
StreamingMovies_Yes	1.4008	0.056	24.895	0.000	1.290	1.511
OnlineBackup_Yes	0.2401	0.054	4.481	0.000	0.135	0.345

All other predictors must be constant for these rules to work.

For continuous predictors:

A one-unit increase in the independent variable is associated with a change in the log odds of the outcome equal to the coefficient value.

For categorical predictors (dummy variables):

The coefficient represents the difference in log odds between the reference category (usually the category with the value of 0) and the category represented by the dummy variable.

```
const is the y intercept.


Observing 'InternetService_Fiber_Optic' True will result in the difference in the log odds of it's
coefficient and the reference category coefficient being applied to the log odds of the dependent
variable.

 Observing 'InternetService_None' True will result in the difference in the log odds of it's
coefficient and the reference category coefficient being applied to the log odds of the dependent
variable.

Observing 'DeviceProtection_yes' True will result in the difference in the log odds of it's
coefficient and the reference category coefficient being applied to the log odds of the dependent
variable.

 Observing 'Streaming_Movies_Yes' True will result in the difference in the log odds of it's
coefficient and the reference category coefficient being applied to the log odds of the dependent
variable.

 Observing 'Online_Backup_Yes' True will result in the difference in the log odds of it's coefficient
and the reference category coefficient being applied to the log odds of the dependent variable.

Observing 'Phone_Yes' True will result in the difference in the log odds of it's coefficient and the
reference category coefficient being applied to the log odds of the dependent variable.
```

significance

I think that the practical significance of this reduced model is moderate. By reading the coefficients in the regression equation we can identify some factors that correlate to a higher probability of churn. We can also see a few factors that correlate to a lower rate of churn. The accuracy is .73 so the model is predicting outcomes correctly in the test set.

The statistical significance here appears good by looking at psuedo R squared and log-likelihood. However I don't think this is a very good model because if you look at the confusion matrix, you can see that it almost had as many false positives as it predicted true positives. I think this may have been caused by the data set it self being skewed. The 'Churn' variable in this data set is mostly false. This looks like it is causing the model to have difficulty predicting 'churn'. I think this could be done better maybe with more data or some different variables.

I really don't like the confusion matrix. The accuracy is .73 but it is misleading because those are all true negatives because most of the 'churn' variable is set to false in this data set.

Limitations.

Some of the limitations of this analysis are that the model is only predicting true negatives with with any accuracy. The ability to predict true positives is almost less than half the predictions. Also I think there is not a clear linear relationship between the predictors and the outcome. That makes the model have lower accuracy. I think that is reflected in the confusion matrix.

2.

My recommendations based on this analysis are that the organization should allocate resources to the sales team to upsell more fiber optic Internet, Online security and focus less on device protection and streaming movies. This is based on the coefficients of the logistic regression equation.

Citations

Assumptions of multiple linear regression (2024) Statistics Solutions. Available at: https://www.statisticssolutions.com/free-resources/directory-of-statistical-analyses/assumptions-of-multiple-linear-regression/ (Accessed: 11 April 2024).

Dansbecker (2018) Using categorical data with one hot encoding, Kaggle. Available at: https://www.kaggle.com/code/dansbecker/using-categorical-data-with-one-hot-encoding (Accessed: 11 April 2024).

Sklearn.metrics.confusion_matrix (no date) scikit. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html#sklearn.metrics.confusion_matrix (Accessed: 12 April 2024).

Explore the core of logistic regression assumptions (2023) Voxco. Available at: https://www.voxco.com/blog/logistic-regression-assumptions/ (Accessed: 11 April 2024).

How to replace column values in a pandas DataFrame (2023) Saturn Cloud Blog. Available at: https://saturncloud.io/blog/how-to-replace-column-values-in-a-pandas-dataframe/ (Accessed: 06 April 2024).

In [55]:
```
import sys
print(sys.version)
```

```
3.10.12 (main, Nov 20 2023, 15:14:05) [GCC 11.4.0]
```

In [ ]: