d208

April 11, 2024

0.1 A

0.1.1 1

How can the organization best allocate resources to direct sales, improve service provision, and or client facing services in order to maximize monthly revenue or 'MonthlyCharge' ?

0.1.2 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 'MonthlyCharge'.

0.2 B.

0.2.1 1.

Linear Relationship: The core premise of multiple linear regression is the existence of a linear relationship between the dependent (outcome) variable and the independent variables. This linearity can be visually inspected using scatterplots, which should reveal a straight-line relationship rather than a curvilinear one.

Multivariate Normality: The analysis assumes that the residuals (the differences between observed and predicted values) are normally distributed. This assumption can be assessed by examining histograms or Q-Q plots of the residuals, or through statistical tests such as the Kolmogorov-Smirnov test.

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.

Variance Inflation Factor (VIF), with VIF values above 10 indicating problematic multicollinearity. Solutions may include centering the data (subtracting the mean score from each observation) or removing the variables causing multicollinearity.

(Assumptions of multiple linear regression 2024) ### 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 pyhon 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 adjusted R squared, and the p values of coefficientst

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

0.3 C.

0.3.1 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.

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

[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
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')]
```

[2]:
```
# 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('MonthlyCharge')
    plt.show()

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

0.4.1 identify duplicate rows by 'CaseOrder'

[3]:
```
# Find duplicate rows
duplicate_rows = df.duplicated(["CaseOrder"]).sum()

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

```
0
```

0.4.2 identify missing values

[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
```

0.4.3 Check for outliers

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

[5]:
```
         CaseOrder           Zip           Lat           Lng     Population  \
count  10000.00000  10000.000000  10000.000000  10000.000000   10000.000000
mean    5000.50000  49153.319600     38.757567    -90.782536    9756.562400
std     2886.89568  27532.196108      5.437389     15.156142   14432.698671
min        1.00000    601.000000     17.966120   -171.688150       0.000000
25%     2500.75000  26292.500000     35.341828    -97.082812     738.000000
50%     5000.50000  48869.500000     39.395800    -87.918800    2910.500000
75%     7500.25000  71866.500000     42.106908    -80.088745   13168.000000
max    10000.00000  99929.000000     70.640660    -65.667850  111850.000000

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

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

       MonthlyCharge  Bandwidth_GB_Year         Item1         Item2  \
count   10000.000000       10000.000000  10000.000000  10000.000000
mean      172.624816        3392.341550      3.490800      3.505100
std        42.943094        2185.294852      1.037797      1.034641
min        79.978860         155.506715      1.000000      1.000000
25%       139.979239        1236.470827      3.000000      3.000000
50%       167.484700        3279.536903      3.000000      4.000000
75%       200.734725        5586.141370      4.000000      4.000000
max       290.160419        7158.981530      7.000000      7.000000

              Item3         Item4         Item5         Item6         Item7  \
count  10000.000000  10000.000000  10000.000000  10000.000000  10000.000000
```

```
mean      3.487000     3.497500     3.492900     3.497300     3.509500
std       1.027977     1.025816     1.024819     1.033586     1.028502
min       1.000000     1.000000     1.000000     1.000000     1.000000
25%       3.000000     3.000000     3.000000     3.000000     3.000000
50%       3.000000     3.000000     3.000000     3.000000     4.000000
75%       4.000000     4.000000     4.000000     4.000000     4.000000
max       8.000000     7.000000     7.000000     8.000000     7.000000

              Item8
count  10000.000000
mean       3.495600
std        1.028633
min        1.000000
25%        3.000000
50%        3.000000
75%        4.000000
max        8.000000
```

0.5 2. Describe dependent and independent vatiables

[6]:
```
## dependent variable

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

[6]:
```
count    10000.000000
mean       172.624816
std         42.943094
min         79.978860
25%        139.979239
50%        167.484700
75%        200.734725
max        290.160419
Name: MonthlyCharge, dtype: float64
```

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

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

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

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

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

[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
```

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

[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
```

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

[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
```

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

[12]:
```
count           10000
unique              3
top       Fiber Optic
```

```
freq              4408
Name: InternetService, dtype: object
```

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

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

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

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

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

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

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

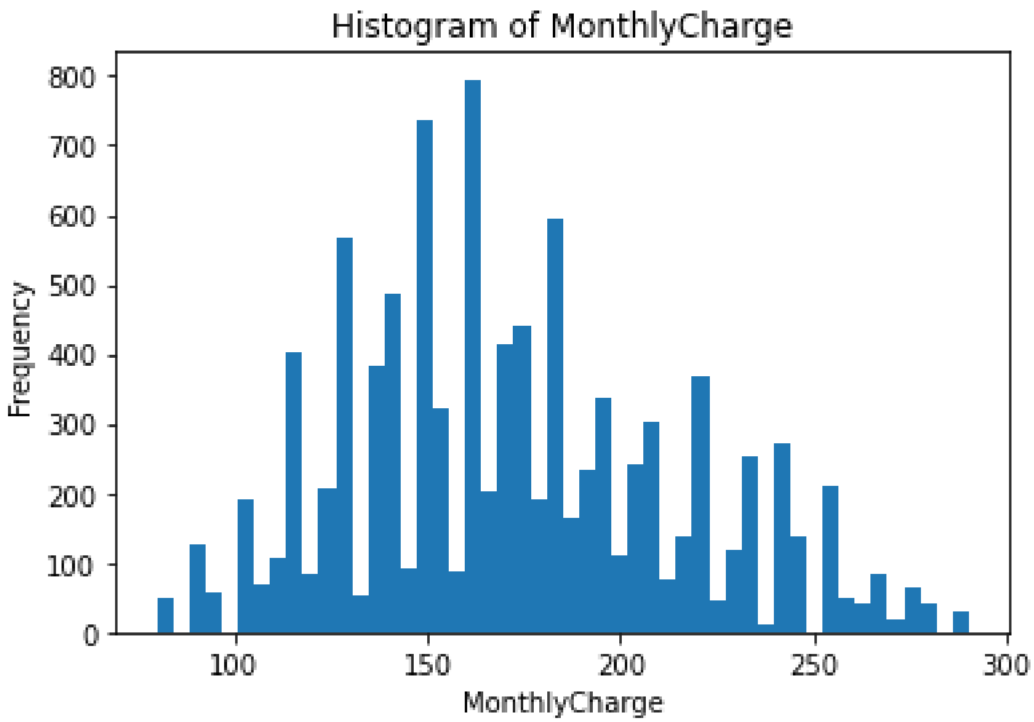
[16]:
```
count     10000
unique        2
top          No
freq       5110
Name: StreamingMovies, dtype: object
```

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

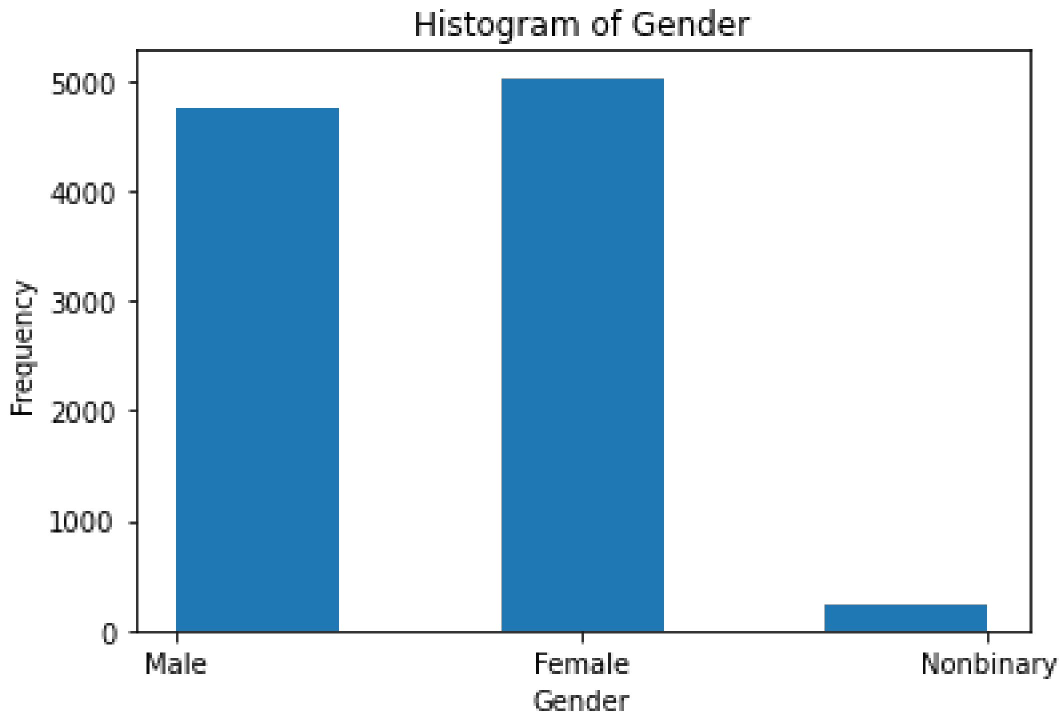
[17]:
```
count     10000
unique        2
top          No
freq       5494
Name: OnlineBackup, dtype: object
```

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

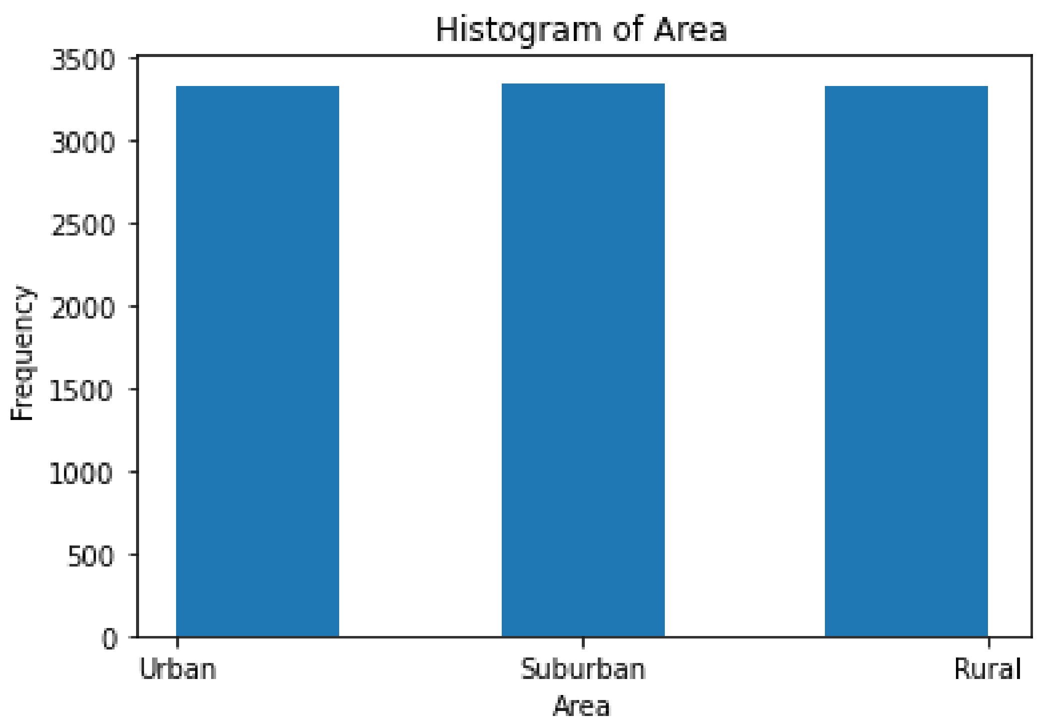
[18]:
```
plot_hist('MonthlyCharge',50)
```



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

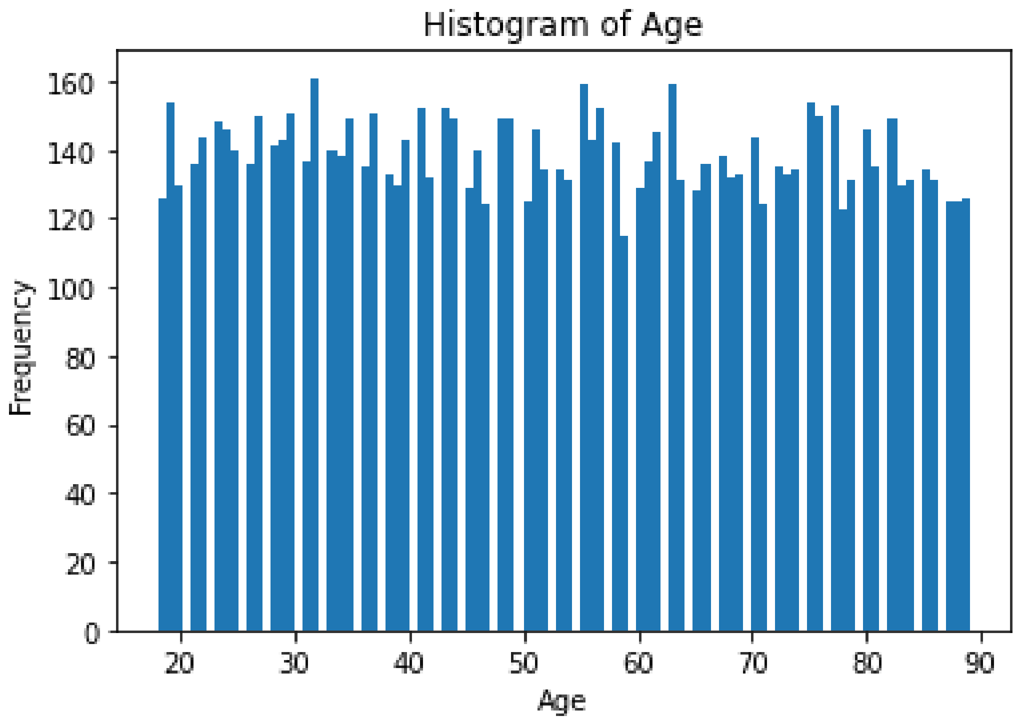


[20]:
```
plot_hist('Area',5)
```



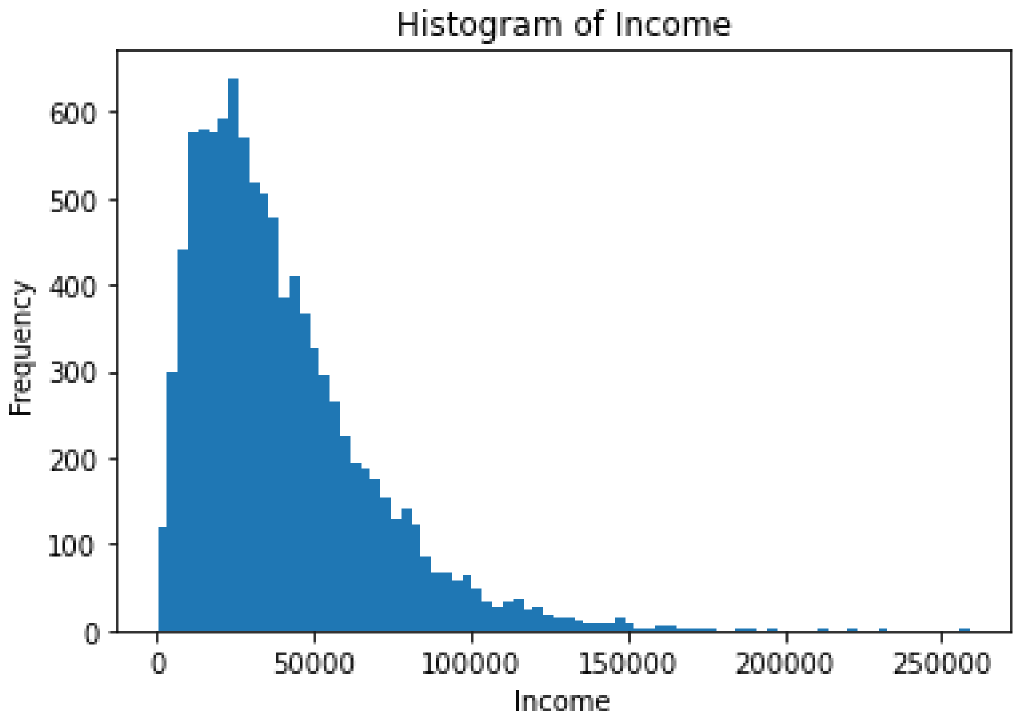
[21]:

```
plot_hist('Age',100)
```

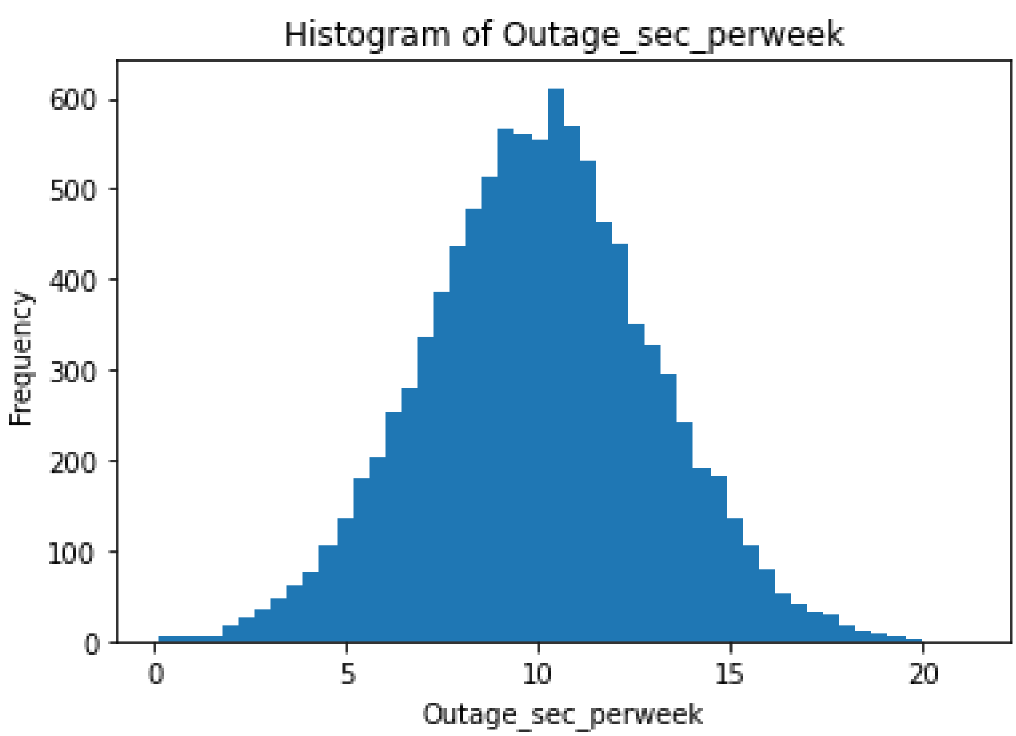


[22]:

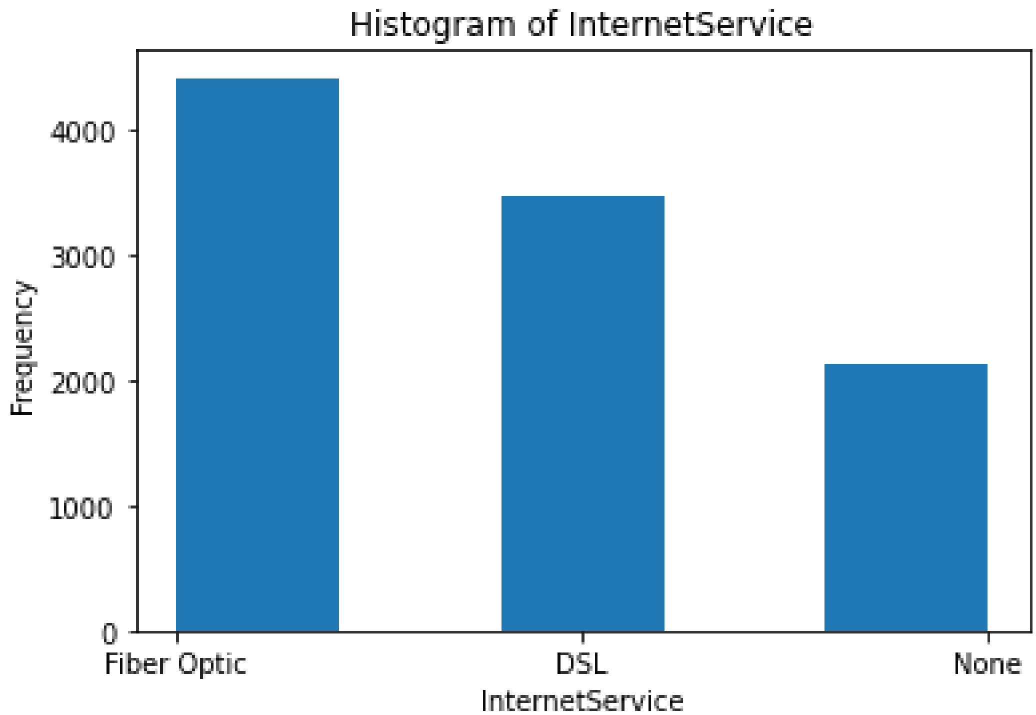
```
plot_hist('Income',80)
```



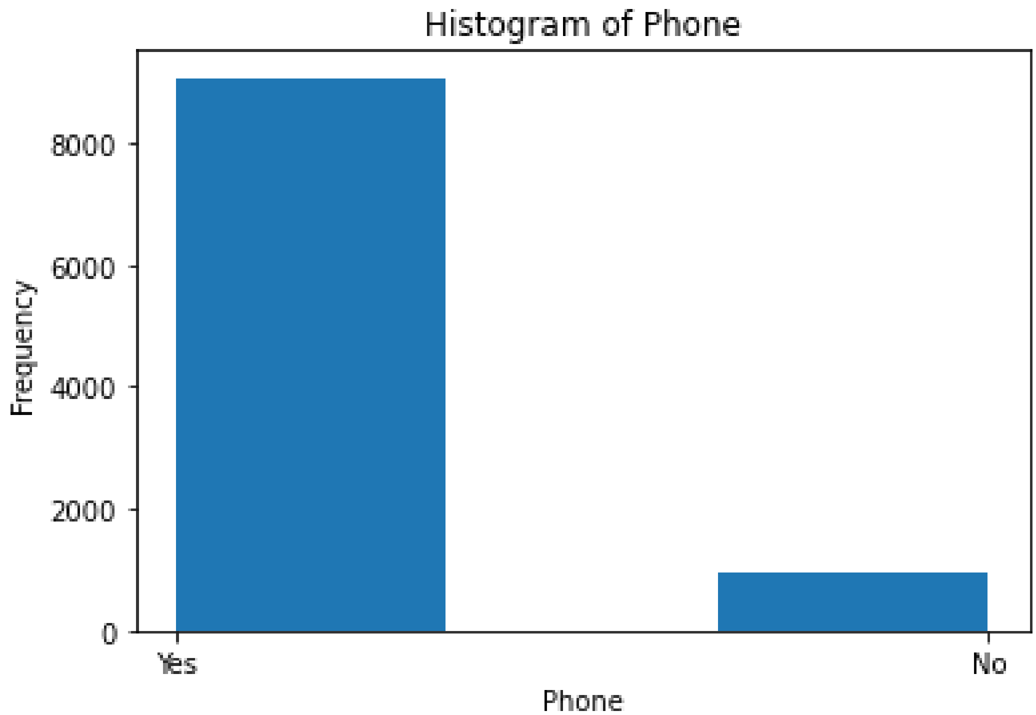

```
[23]: plot_hist('Outage_sec_perweek',50)
```



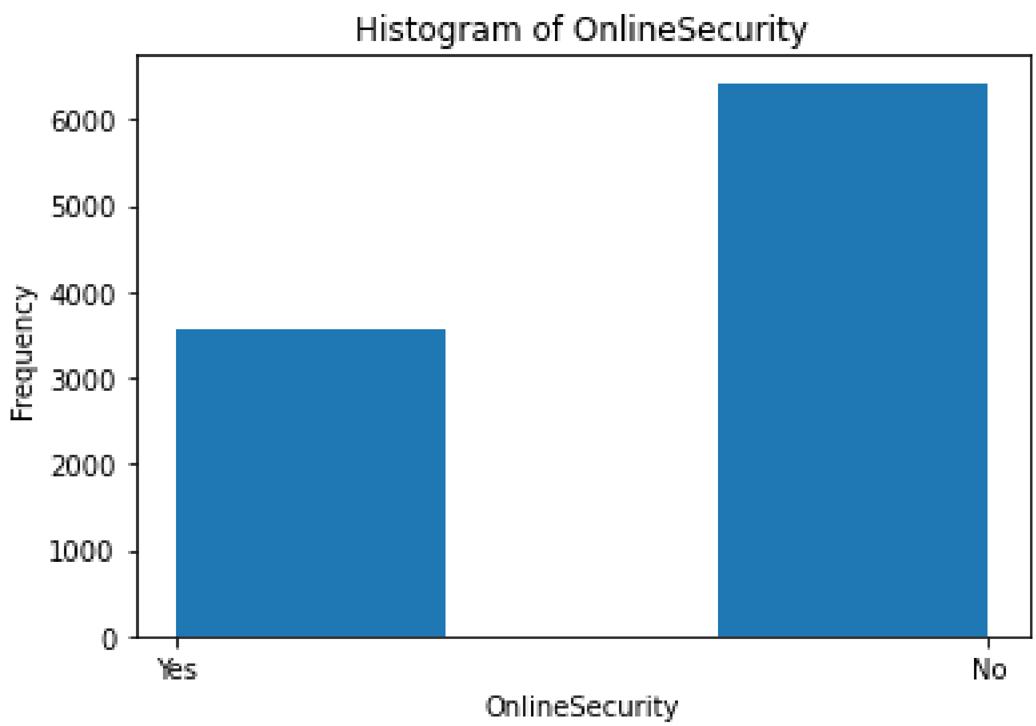
[24]:
```
plot_hist('InternetService',5)
```



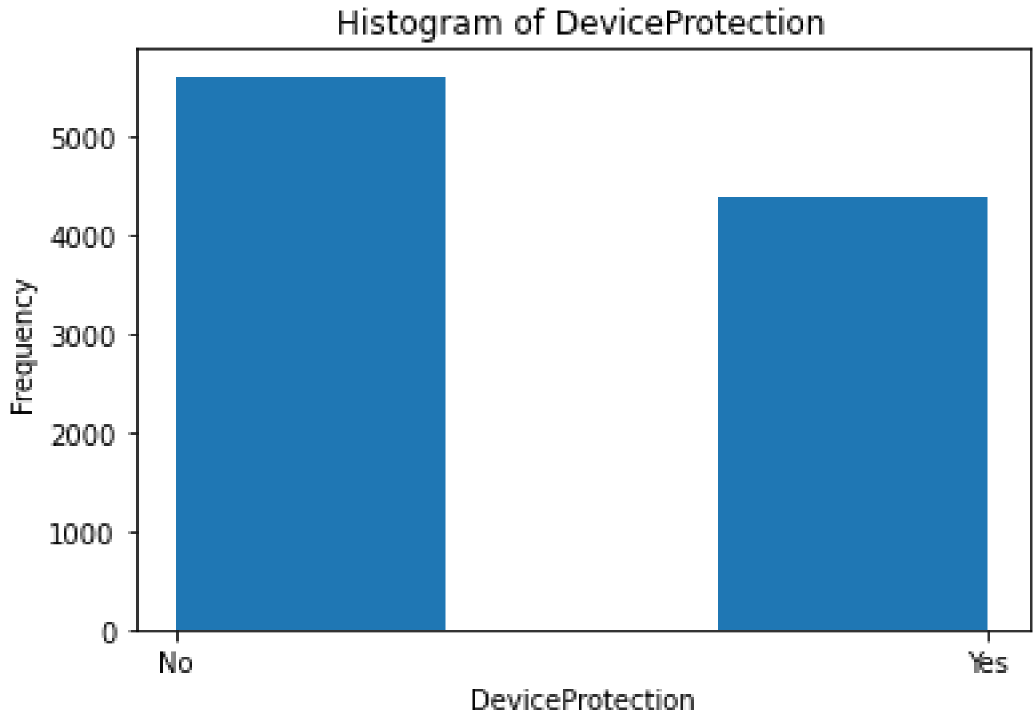
[25]:
```
plot_hist('Phone',3)
```



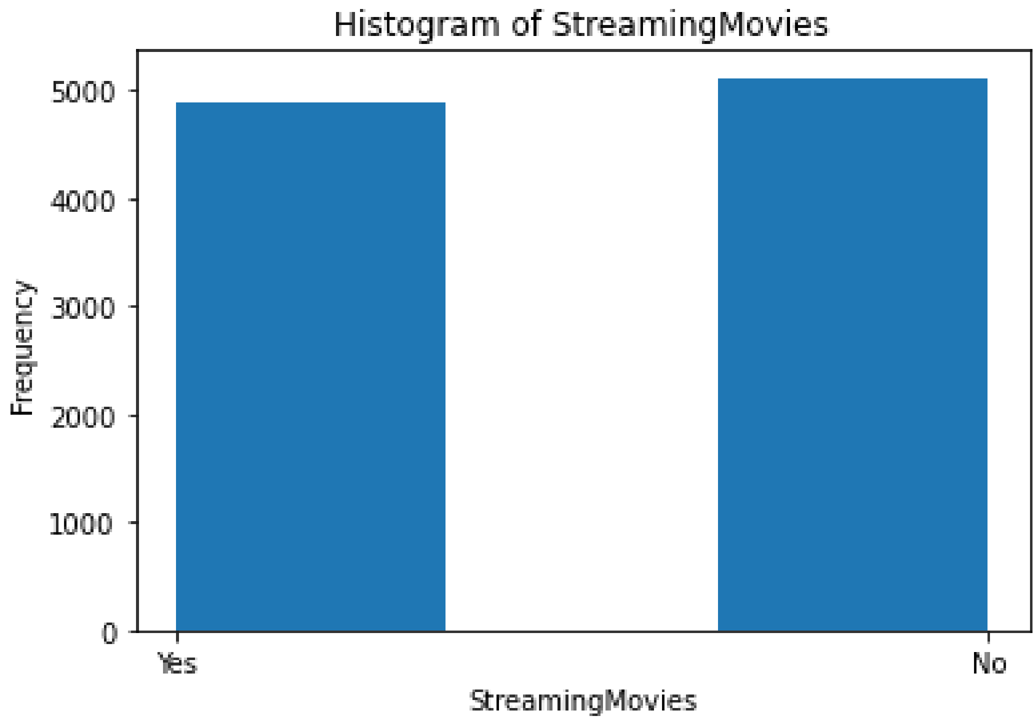
[26]:
```
plot_hist('OnlineSecurity',3)
```



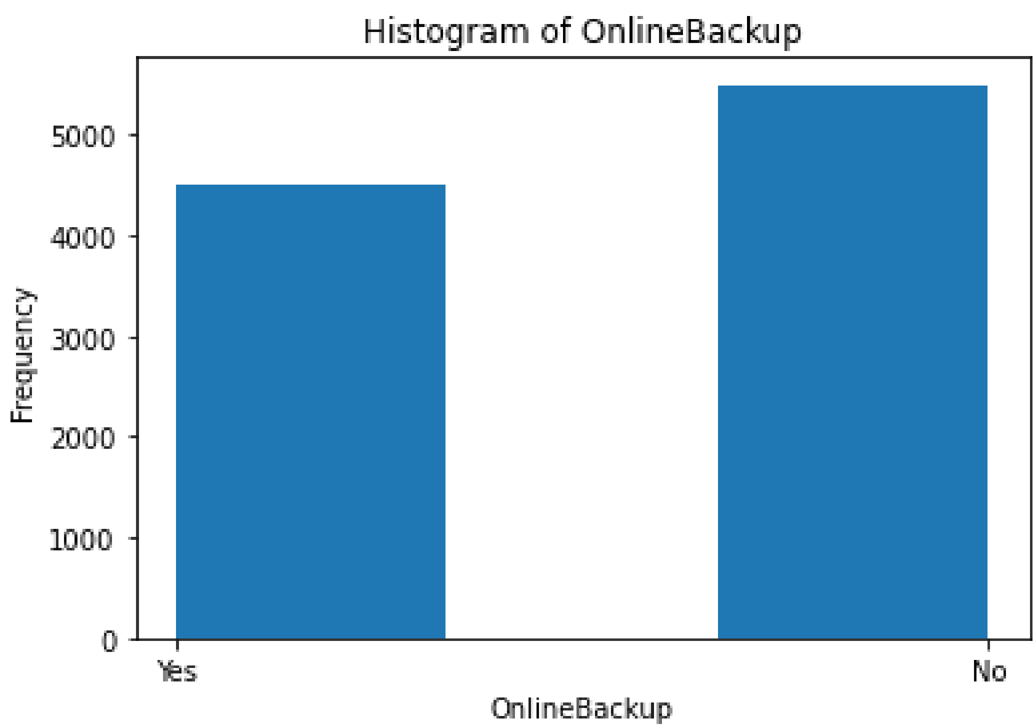
[27]:
```
plot_hist('DeviceProtection',3)
```



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

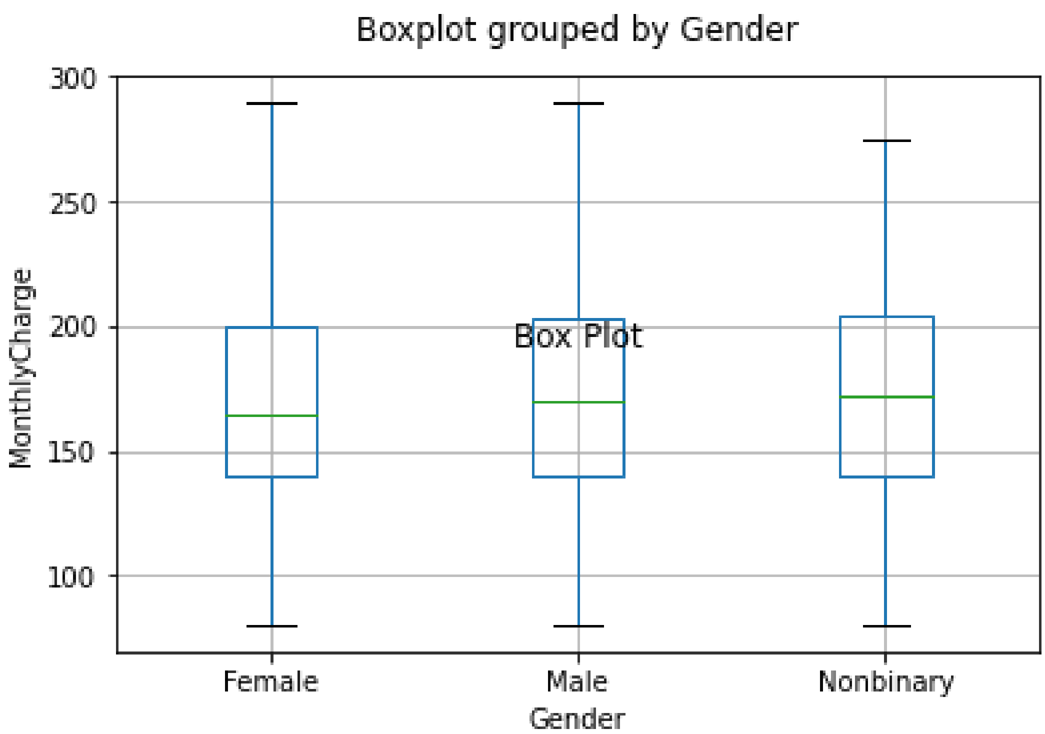


[29]:
```
plot_hist('OnlineBackup',3)
```

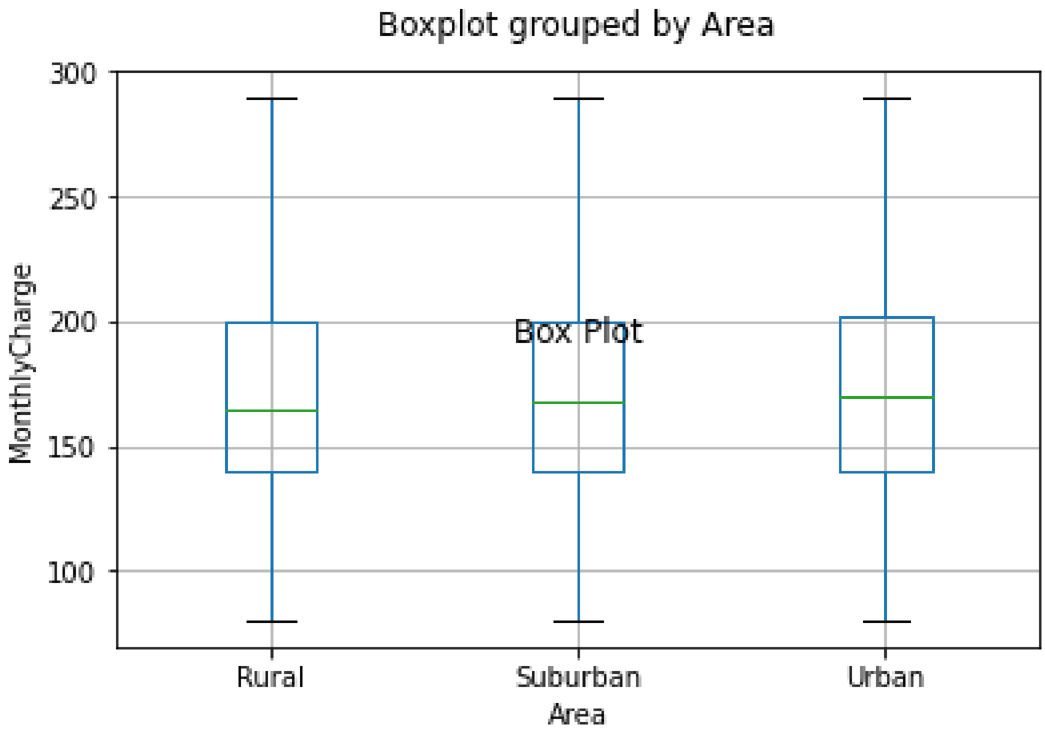


0.7 bivariate - graphing against the dependent variable

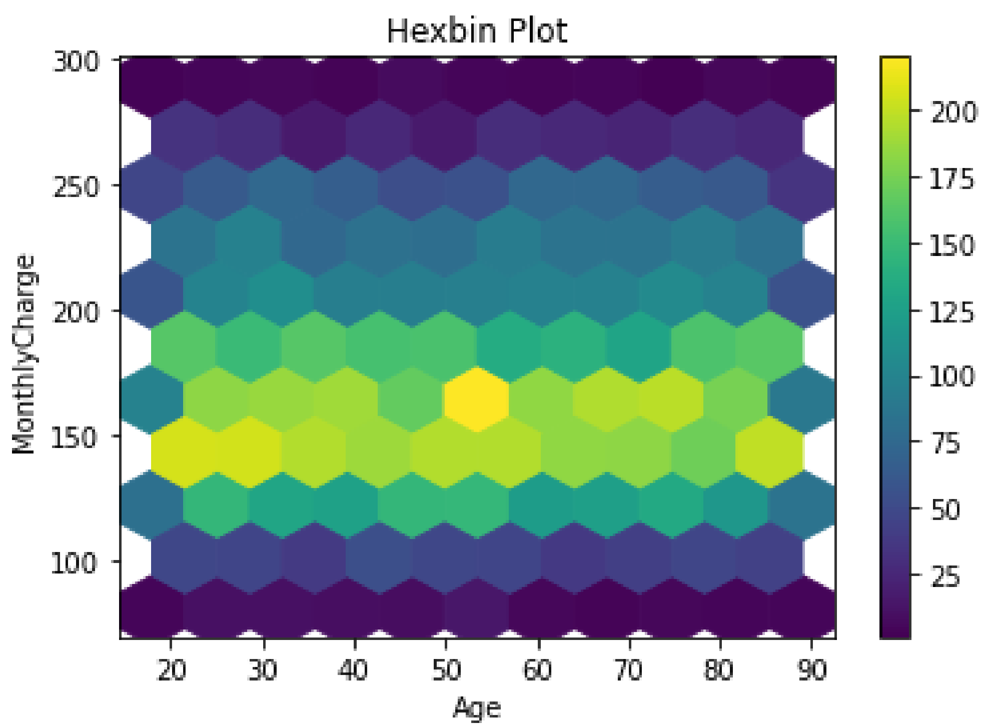
[30]:
```
box_plot('Gender')
```



[31]:
```
box_plot('Area')
```

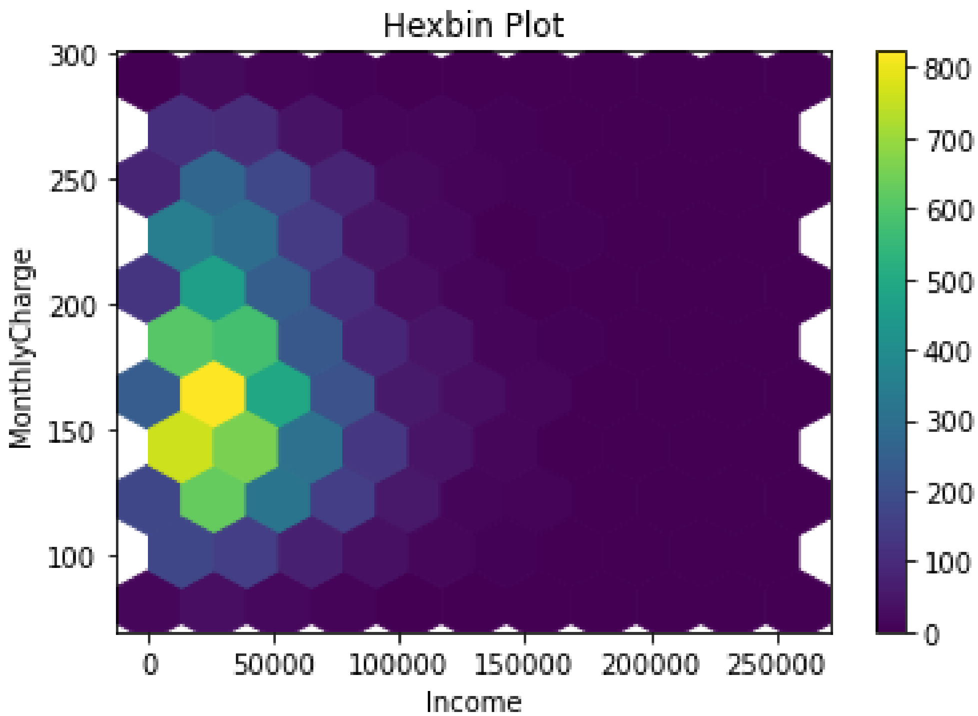


[32]:
```
line_plot('Age')
```



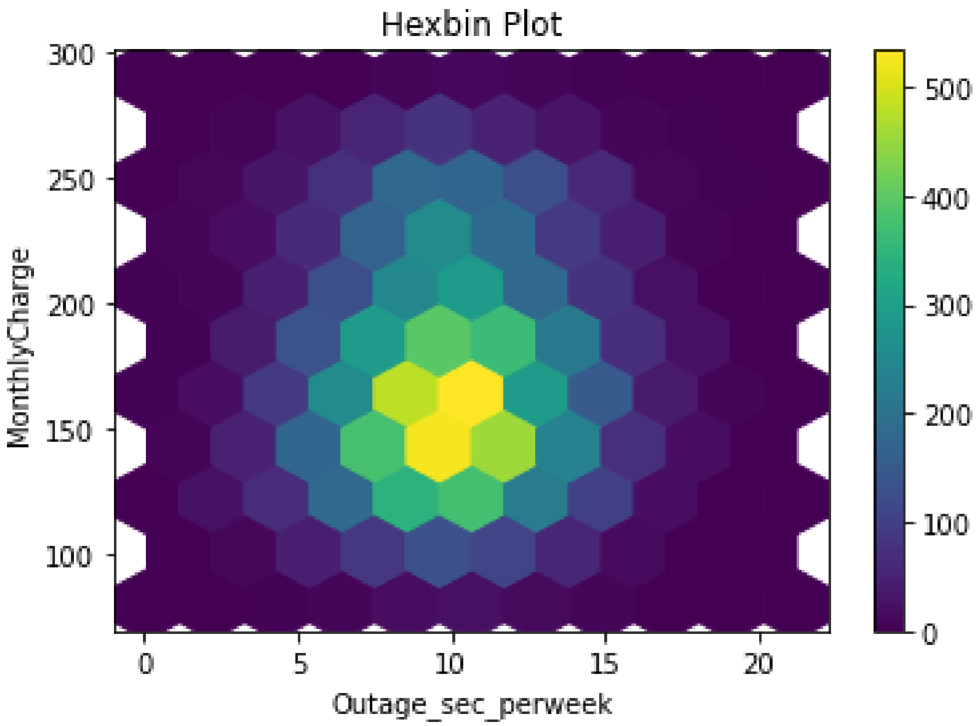
[33]:

```
line_plot('Income')
```

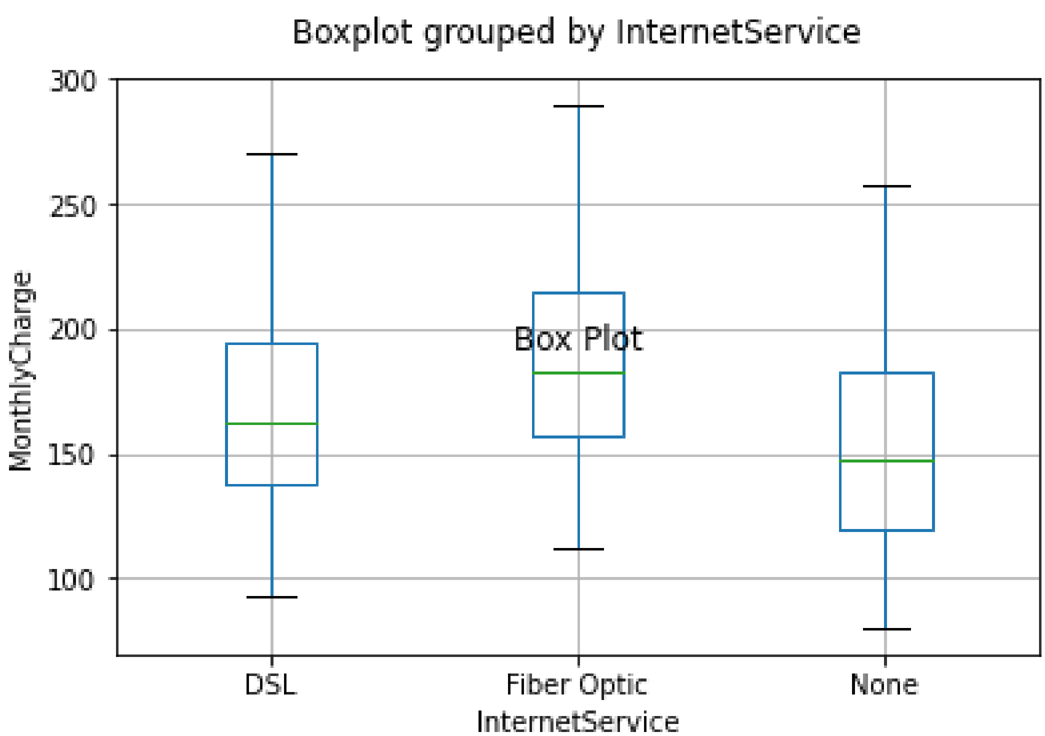


[34]:

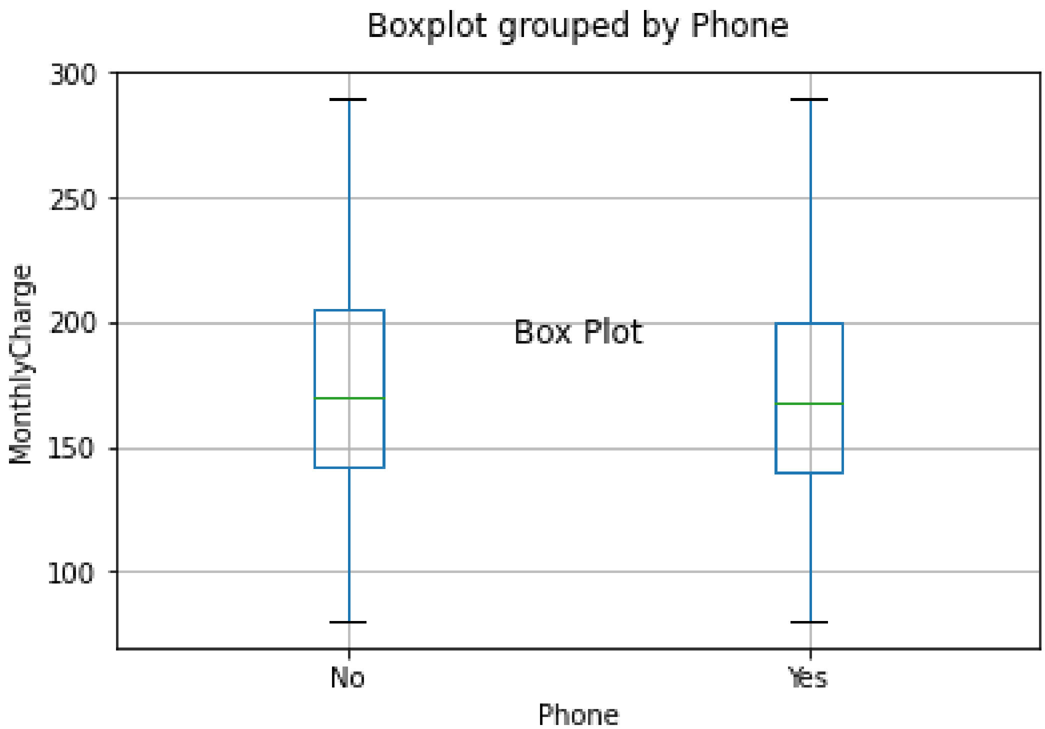
```
line_plot('Outage_sec_perweek')
```



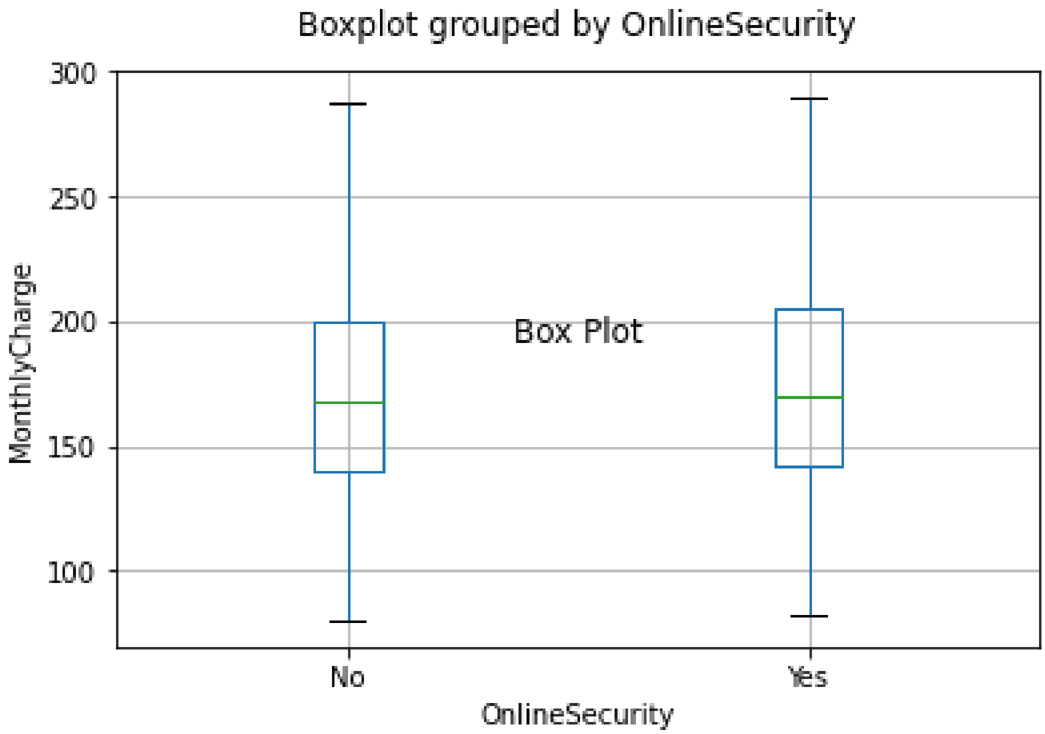

```
[35]: box_plot('InternetService')
```



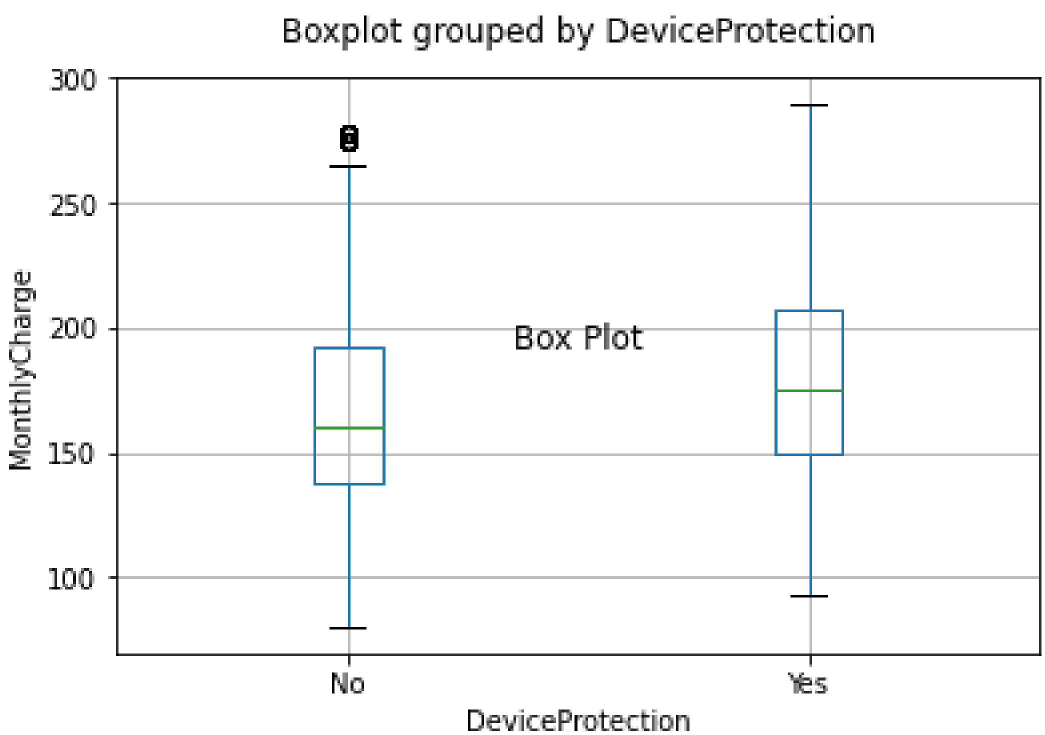
[36]:
```
box_plot('Phone')
```



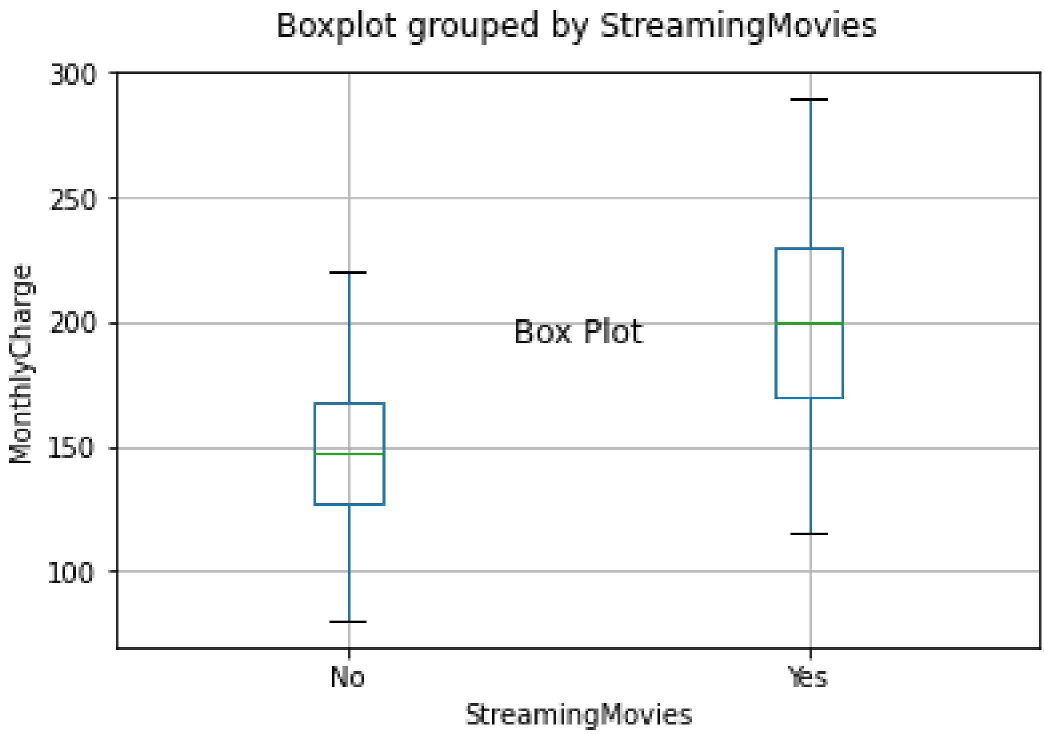
[37]:
```
box_plot('OnlineSecurity')
```



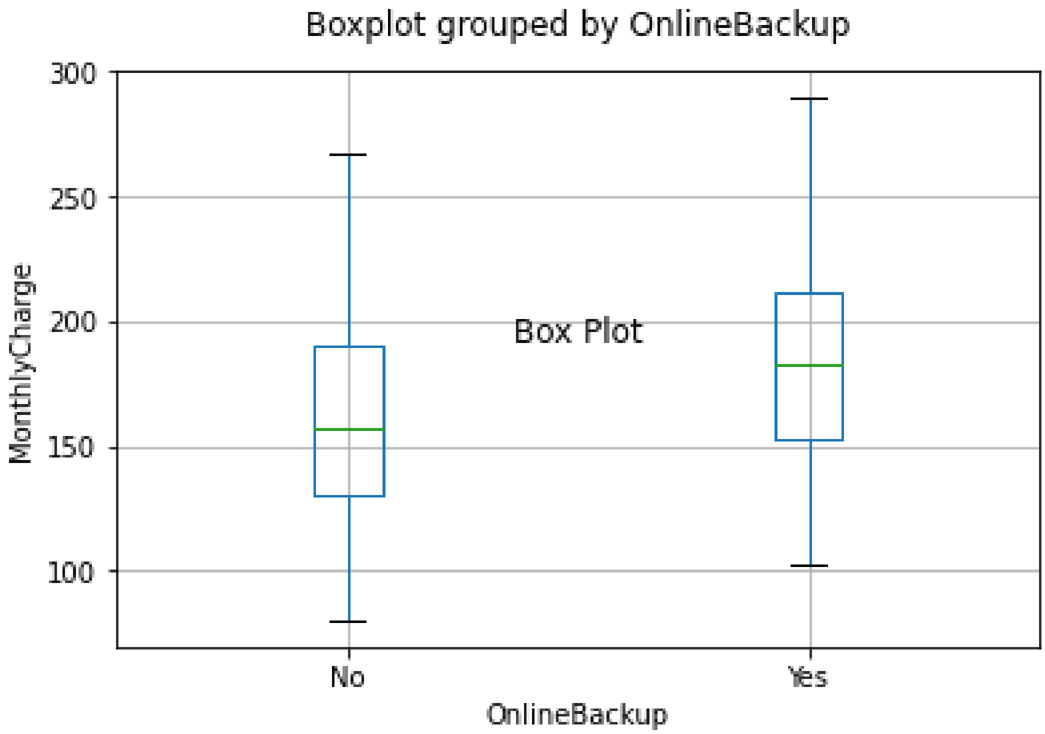
[38]:
```
box_plot('DeviceProtection')
```



[39]:
```
box_plot('StreamingMovies')
```



[40]:
```
box_plot('OnlineBackup')
```



0.7.1 4)

My goals for data transformation are to one-hot encode the categorical variables and then normalize all values.

[41]:
```
#split continuous and categorical variables into separate dataframes
dfcon = df[['Age','Income','Outage_sec_perweek']]
dfcat =␣
 ↪df[['Gender','Area','InternetService','Phone','OnlineSecurity','DeviceProtection','Streamin
#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)
#normalize the data
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
df_normalized = pd.DataFrame(scaler.fit_transform(data), columns=data.columns)
#write the prepared data to .csv file
df_normalized.to_csv('prepared-data.csv', index=False)
```

0.8 D. Compare an initial and a reduced linear regression model

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

[42]:
```python
#Initial Model

import statsmodels.api as sm
dependent_vars = sm.add_constant(df_normalized)
model = sm.OLS(df['MonthlyCharge'], dependent_vars).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:          MonthlyCharge   R-squared:                       0.554
Model:                            OLS   Adj. R-squared:                  0.554
Method:                 Least Squares   F-statistic:                     887.1
Date:                Thu, 11 Apr 2024   Prob (F-statistic):               0.00
Time:                        18:16:57   Log-Likelihood:                -47747.
No. Observations:               10000   AIC:                         9.552e+04
Df Residuals:                    9985   BIC:                         9.563e+04
Df Model:                          14
Covariance Type:            nonrobust
================================================================================
===============
                                  coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
---------------
const                         125.2258      1.688     74.185      0.000
121.917     128.535
Age                             0.3563      0.985      0.362      0.717
-1.574       2.286
Income                          0.7141      2.632      0.271      0.786
-4.446       5.874
Outage_sec_perweek              1.9076      2.036      0.937      0.349
-2.084       5.899
Gender_Male                     0.2567      0.581      0.442      0.659
-0.883       1.396
Gender_Nonbinary                0.8091      1.932      0.419      0.675
-2.978       4.596
Area_Suburban                  -0.0939      0.703     -0.134      0.894
-1.471       1.283
Area_Urban                     -0.0938      0.704     -0.133      0.894
-1.473       1.286
InternetService_Fiber Optic    19.1922      0.652     29.449      0.000
17.915      20.470
InternetService_None          -13.9434      0.790    -17.642      0.000
```

```
-15.493     -12.394
Phone_Yes                     -1.3398      0.987     -1.357      0.175
-3.275       0.595
OnlineSecurity_Yes             2.7922      0.599      4.661      0.000
1.618       3.966
DeviceProtection_Yes          12.6749      0.579     21.888      0.000
11.540      13.810
StreamingMovies_Yes           51.8440      0.574     90.284      0.000
50.718      52.970
OnlineBackup_Yes              22.0936      0.577     38.288      0.000
20.962      23.225
==============================================================================
Omnibus:                      901.877   Durbin-Watson:                   1.995
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              280.582
Skew:                           0.059   Prob(JB):                     1.18e-61
Kurtosis:                       2.188   Cond. No.                         18.6
==============================================================================

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

0.9 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. This is an effective way to reduce the model because I may choose to keep some predictor variables that may not necessarily meet standard thresholds of p < .05. This will enable me to more precisely answer the research question by identifying the effect of these predictor variables on the outcome variable even though they may not meet the p > .05 criteria. So even though the predictors may have a slightly larger p value we can still answer questions about how a variable correlates to 'MonthlyCharge. We not only want to predict future values of 'MonthlyCharge' but also know how a given predictor variable will correlate with things like the magnitude and sign of the coefficient so it may be wise to include them in the model. Also a predictor variable may have a good p value but won't be practically significant. With this feature selection method I have more control to actually get meaningful information about what the correlations are to 'monthlyCharge'.

I have chosen to use the adjusted r squared value as an evaluation metric. I have chose this one in particular because it will penalize for overfitting the model. It will more accurately predict goodness of fit with models with large numbers of predictor variables such as this one. Since it takes into account overfitting, I am less likely to create a model that uses redundant data and inaccurately defines the correlations of each predictor variable leading to false information about correlations to 'MonthlyCharge.'

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

[43]:
```
#original model
df_normalized = pd.DataFrame(scaler.fit_transform(data), columns=data.columns)
dependent_vars = sm.add_constant(df_normalized)
model = sm.OLS(df['MonthlyCharge'], dependent_vars).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:          MonthlyCharge   R-squared:                       0.554
Model:                            OLS   Adj. R-squared:                  0.554
Method:                 Least Squares   F-statistic:                     887.1
Date:                Thu, 11 Apr 2024   Prob (F-statistic):               0.00
Time:                        18:16:58   Log-Likelihood:                -47747.
No. Observations:               10000   AIC:                         9.552e+04
Df Residuals:                    9985   BIC:                         9.563e+04
Df Model:                          14
Covariance Type:            nonrobust
================================================================================
===============
                                  coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
---------------
const                         125.2258      1.688     74.185      0.000
121.917     128.535
Age                             0.3563      0.985      0.362      0.717
-1.574       2.286
Income                          0.7141      2.632      0.271      0.786
-4.446       5.874
Outage_sec_perweek              1.9076      2.036      0.937      0.349
-2.084       5.899
Gender_Male                     0.2567      0.581      0.442      0.659
-0.883       1.396
Gender_Nonbinary                0.8091      1.932      0.419      0.675
-2.978       4.596
Area_Suburban                  -0.0939      0.703     -0.134      0.894
-1.471       1.283
Area_Urban                     -0.0938      0.704     -0.133      0.894
-1.473       1.286
InternetService_Fiber Optic    19.1922      0.652     29.449      0.000
17.915      20.470
InternetService_None          -13.9434      0.790    -17.642      0.000
-15.493     -12.394
Phone_Yes                      -1.3398      0.987     -1.357      0.175
```

```
-3.275       0.595
OnlineSecurity_Yes              2.7922      0.599      4.661      0.000
1.618       3.966
DeviceProtection_Yes           12.6749      0.579     21.888      0.000
11.540      13.810
StreamingMovies_Yes            51.8440      0.574     90.284      0.000
50.718      52.970
OnlineBackup_Yes               22.0936      0.577     38.288      0.000
20.962      23.225
==============================================================================
Omnibus:                      901.877   Durbin-Watson:                   1.995
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              280.582
Skew:                           0.059   Prob(JB):                     1.18e-61
Kurtosis:                       2.188   Cond. No.                         18.6
==============================================================================

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

0.10 Reduced model

[44]:

```python
#Reduced model
df_normalized = pd.DataFrame(scaler.fit_transform(data), columns=data.columns)
del df_normalized['Area_Urban']
del df_normalized['Area_Suburban']
del df_normalized['Age']
del df_normalized['Outage_sec_perweek']
del df_normalized['Gender_Male']
del df_normalized['Gender_Nonbinary']
del df_normalized['Income']
del df_normalized['Phone_Yes']
dependent_vars = sm.add_constant(df_normalized)
model = sm.OLS(df['MonthlyCharge'], dependent_vars).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:          MonthlyCharge   R-squared:                       0.554
Model:                            OLS   Adj. R-squared:                  0.554
Method:                 Least Squares   F-statistic:                     2070.
Date:                Thu, 11 Apr 2024   Prob (F-statistic):               0.00
Time:                        18:16:58   Log-Likelihood:                -47749.
No. Observations:               10000   AIC:                         9.551e+04
Df Residuals:                    9993   BIC:                         9.556e+04
Df Model:                           6
Covariance Type:            nonrobust
================================================================================
```

```
===============
                                    coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
---------------
const                           125.2414      0.696    179.964      0.000
123.877     126.606
InternetService_Fiber Optic      19.1959      0.651     29.472      0.000
17.919      20.473
InternetService_None            -13.9448      0.790    -17.652      0.000
-15.493     -12.396
OnlineSecurity_Yes                2.7878      0.599      4.657      0.000
1.614       3.961
DeviceProtection_Yes             12.7159      0.578     21.991      0.000
11.582      13.849
StreamingMovies_Yes              51.8573      0.574     90.356      0.000
50.732      52.982
OnlineBackup_Yes                 22.1003      0.577     38.334      0.000
20.970      23.230
==============================================================================
Omnibus:                      905.812   Durbin-Watson:                   1.995
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              281.087
Skew:                           0.059   Prob(JB):                     9.18e-62
Kurtosis:                       2.187   Cond. No.                         5.17
==============================================================================

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

0.11 E.

0.11.1 1.Explain your data analysis process by comparing the initial multiple linear regression model and reduced linear regression model

My model evaluation metric had originally been adjusted R squared. I have decided to change this to the F statistic metric. Since I had predictor variables that had large coefficients the R squared value was about the same in both models. This is because the predictor variables with the largest coefficients and smallest P values were not removed.

I think a better metric to compare these two models is the F statistic. This measures the overall statistical significance of the model. The value inceased as I removed variables with high p values. This indicates that the reduced model has less variables that are not statistically significant included.

Original F statistic = 887.1

Reduced model F statistic = 2070

Original R squared = .554

Reduced model R squared = .554

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

[45]:
```python
#calculations to reduce original model
df_normalized = pd.DataFrame(scaler.fit_transform(data), columns=data.columns)
del df_normalized['Area_Urban']
del df_normalized['Area_Suburban']
del df_normalized['Age']
del df_normalized['Outage_sec_perweek']
del df_normalized['Gender_Male']
del df_normalized['Gender_Nonbinary']
del df_normalized['Income']
del df_normalized['Phone_Yes']
dependent_vars = sm.add_constant(df_normalized)
model = sm.OLS(df['MonthlyCharge'], dependent_vars).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:          MonthlyCharge   R-squared:                       0.554
Model:                            OLS   Adj. R-squared:                  0.554
Method:                 Least Squares   F-statistic:                     2070.
Date:                Thu, 11 Apr 2024   Prob (F-statistic):               0.00
Time:                        18:16:58   Log-Likelihood:                -47749.
No. Observations:               10000   AIC:                         9.551e+04
Df Residuals:                    9993   BIC:                         9.556e+04
Df Model:                           6
Covariance Type:            nonrobust
================================================================================
===============
                                  coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
---------------
const                         125.2414      0.696    179.964      0.000
123.877     126.606
InternetService_Fiber Optic    19.1959      0.651     29.472      0.000
17.919      20.473
InternetService_None          -13.9448      0.790    -17.652      0.000
-15.493     -12.396
OnlineSecurity_Yes              2.7878      0.599      4.657      0.000
1.614       3.961
DeviceProtection_Yes           12.7159      0.578     21.991      0.000
11.582      13.849
StreamingMovies_Yes            51.8573      0.574     90.356      0.000
50.732      52.982
```

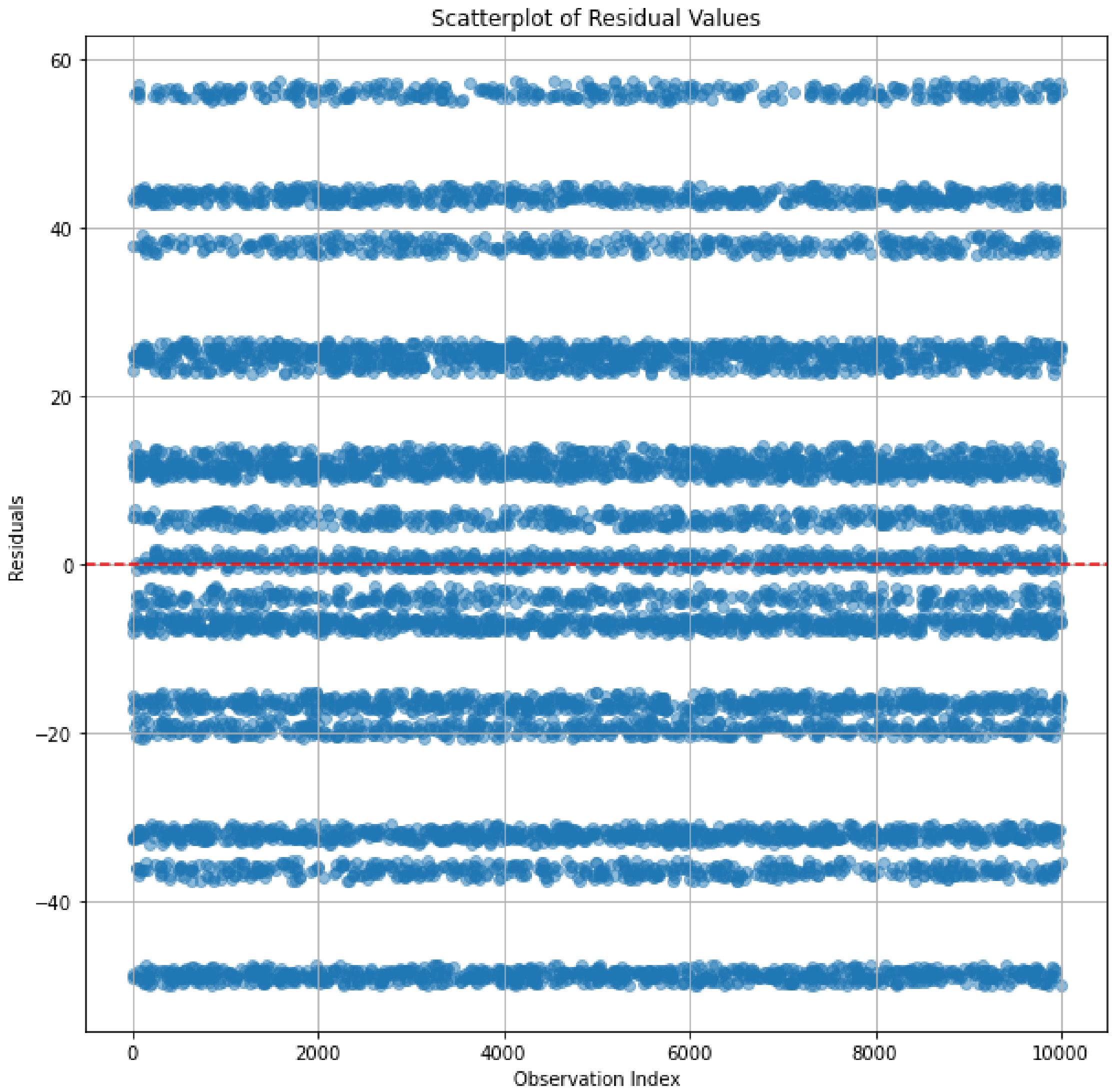
```
OnlineBackup_Yes               22.1003      0.577     38.334      0.000
20.970      23.230
==============================================================================
Omnibus:                      905.812   Durbin-Watson:                   1.995
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              281.087
Skew:                           0.059   Prob(JB):                     9.18e-62
Kurtosis:                       2.187   Cond. No.                         5.17
==============================================================================

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

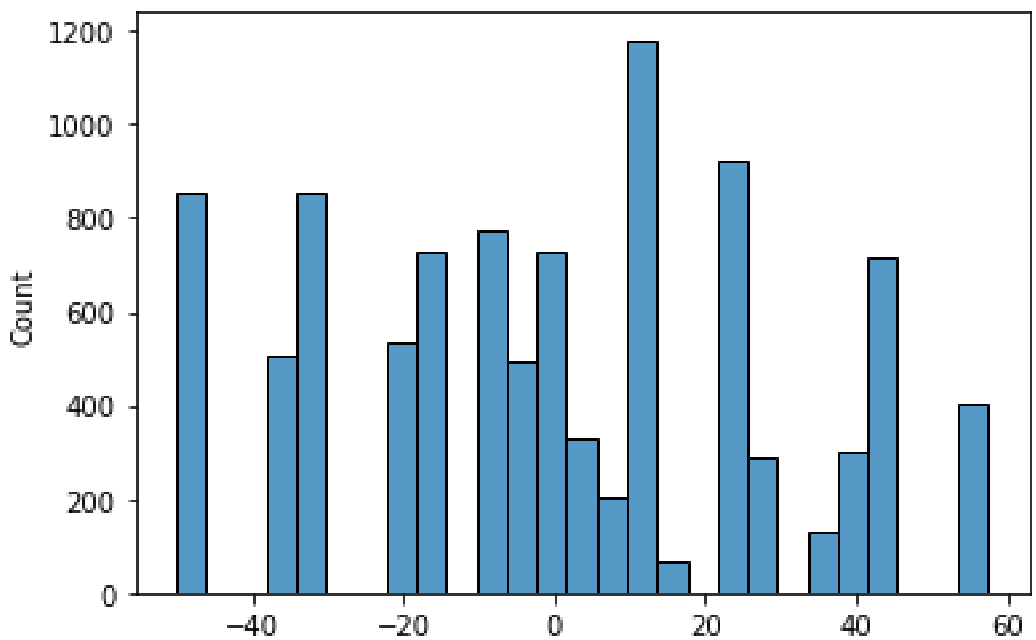
0.13 residual plot

[46]:

```python
# Create a scatterplot of residual values
residuals = model.resid
plt.figure(figsize=(10, 10))
plt.scatter(range(len(residuals)), residuals, alpha=0.5)
plt.axhline(y=0, color='r', linestyle='--')  # Add a horizontal line at y=0
plt.title('Scatterplot of Residual Values')
plt.xlabel('Observation Index')
plt.ylabel('Residuals')
plt.grid(True)
plt.show()
# Create a histogram of residual values
sns.histplot(residuals);
```



Scatterplot of Residual Values
Residuals
Observation Index
60
40
20
0
−20
−40
0
2000
4000
6000
8000
10000



0.14 residual standard error

[47]:
```
np.sqrt(np.sum(model.resid**2)/model.df_resid)
```

[47]: 28.68192657650494

0.15 3. code will be submitted with assignment.

0.16 F.

0.16.1 1. Discuss the results of your data analysis

regression equation : Y = 125.2414 + 19.959(X) + -13.9448(X) + 2.7878(x) + 12.7159((x) + 51.8573(X) + 22.1003(X) #### Interpretation of coefficients:

```
InternetService_Fiber Optic    19.1959   is the magnitude and it has a positive correlation wi
InternetService_None          -13.9448   is the magnitude and it has a negative correlation wi
OnlineSecurity_Yes              2.7878   is the magnitude and it has a positive correlation wi
DeviceProtection_Yes           12.7159   is the magnitude and it has a positive correlation wi
StreamingMovies_Yes            51.8573   is the magnitude and it has a positive correlation wi
OnlineBackup_Yes               22.1003   is the magnitude and it has a positive correlation wi

all coefficients have a p value of < .05 so they are statistically significant.
```

0.16.2 significance

I think that the practical significance of this reduced model is not that great. That is because it basically shows us some common sense things that we could just guess. Such as if a person subscribes to more services the monthly charge would be greater.

The statistical significance here is good because the coefficients show what we could guess with common sense. So with a different data set this could be very useful.

Limitations. Some of the limitations of this analysis are that the model works better with normally distributed variables that have a linear correlation with the outcome variable. Another limitation is that the standard error can be pretty large. A third limitation is that this only works for a continuous variables.

0.16.3 2.

My recommendations based on this analysis are that the organization should allocate resources to the sales team to upsell more services to increase the 'MonthlyCharge' for each customer. We could have guessed that maybe, but the data is here to confirm that and remove any doubt.

0.16.4 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).

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).

[48]:
```
import sys
print(sys.version)
```

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

[ ]:
```

```