Performance Assessment for

D208: Predictive Modeling

Task 1 Revision 1

Drew Mendez

MSDA Western Governors University

D208: Predictive Modeling

Dr. Taylor Jensen

July 4, 2024

D208_PA_MendezD_Task1_Revision1

July 4, 2024

1 Part I: Research Question

1.1 A. Purpose of the Data Analysis

1.1.1 A1. Research Question

The data set selected for this performance assessment is the **churn** data set. The research question for this assessment is:

What variables contribute to a customer's income?

1.1.2 A2. Goals of the Data Analysis

The goal of this data analysis is to determine influential explanatory variables to develop a multiple regression model that can be used by stakeholders to predict the target variable `Income`, a continuous variable. This model could then be used by stakeholders in their marketing strategies, perhaps with ad campaigns for high-income customers offering premium services, or to market budget-friendly services to low-income customers.

2 Part II: Method Justification

2.1 B. Multiple Linear Regression Methods

2.1.1 B1. Four Assumptions of Multiple Linear Regression

The four assumptions of multiple linear regression are:

- Linearity: there exists a linear relationship between the target and each explanatory variable.
- No Multicollinearity: none of the explanatory variables are highly correlated with each other.
- Homoscedasticity: the residuals have constant variance
- Multivariate Normality: the residuals of the model are normally distributed

(Karir, 2022)

2.1.2 B2. Benefits of using Python

Python was chosen for the functionality of the many packages available and for the ability to write functions that can be reused throughout the project. The libraries and packages that are essential to this analysis are:

- From **Pandas**, the `.isnull()`, `.duplicated()`, and `.sum()` methods provide some important basic functionality. Additionally, `.quantile()` is used in the detection of outliers and `.value_counts()` is used to count unique elements in data frames.
- From **Matplotlib**, `pyplot` is used to generate histograms and boxplots of variables to observe their distributions and outliers.
- From **Statsmodels**, `OLS()`, `.fit()`, and `.add_constant()` are necessary to produce the multiple regression model.

2.1.3 B3. Why Multiple Linear Regression is Appropriate

The research question posed here has a continuous variable as the target variable. Since the goal of this analysis is to construct a model using multiple explanatory variables to predict a continuous response variable, a multiple linear regression model is appropriate.

3 Part III: Data Preparation

3.1 C. Summary of the Data Preparation Process

3.1.1 C1. Data Cleaning Goals

Before constructing the model, first the data set will be cleaned and treated. The plan to clean the data set involves detecting and treating duplicates, missing values, and outliers, and the re-expression of categorical variables. The steps and techniques necessary to perform these tasks is given:

- Duplicates:
  - Duplicates are detected below by chaining the `.duplicated()` and `.sum()` methods from the `Pandas` library and calling them on the data frame, returning the total count of duplicate observations. It is shown below that there are no duplicate rows.
- Missing Values:
  - Missing values are detected below by chaining the `.isnull()` and `.sum()` methods from the `Pandas` library and calling them on the data frame, returning the total count of missing values for each variable. Since the `InternetService` variable has `None` as one of its options, the 2129 erroneously identified null values are imputed with `None` to avoid being interpreted as nulls. It is shown below that there are no other missing values.
- Outliers:
  - The outliers of the following thirteen **quantitative variables** are identified using the `boxplot()` function from the `matplotlib` library and counted below using a user-defined function. It is shown below that there are no unacceptable/unreasonable outliers, so all outliers shown here will be retained.
    - Income: Annual income of customer (**continuous numeric data**)
    - Lat: GPS coordinates of the latitude of the customer residence (**continuous numeric data**)
    - Lng: GPS coordinates of the longitude of the customer residence (**continuous numeric data**)
    - Population: Population within a mile radius of customer (**discrete numeric data**)
    - Children: Number of children in customer's household (**discrete numeric data**)
    - Age: Age of customer (**continuous numeric data**)

    * Outage_sec_perweek: Average number of seconds per week of system outages in the customer's neighborhood (**continuous numeric data**)
    * Email: Number of emails sent to the customer in the last year (marketing or correspondence) (**discrete numeric data**)
    * Contacts: Number of times customer contacted technical support (**discrete numeric data**)
    * Yearly_equip_failure: The number of times customer's equipment failed and had to be reset/replaced in the past year (**discrete numeric data**)
    * Tenure: Number of months the customer has stayed with the provider (**continuous numeric data**)
    * MonthlyCharge: The amount charged, on average, per customer monthly (**continuous numeric data**)
    * Bandwidth_GB_Year: The average amount of data used, in GB, in a year by the customer (**continuous numeric data**)
- Re-expression of Categorical Variables:
  - The following thirteen **binary nominal categorical variables** are re-expressed below using binary encoding:
    * `Churn`: Whether the customer discontinued service within the last month (yes, no)
    * `Techie`: Whether the customer considers themselves technically inclined (yes, no)
    * `Port_modem`: Whether the customer has a portable modem (yes, no)
    * `Tablet`: Whether the customer owns a tablet such as iPad, Surface, etc. (yes, no)
    * `Phone`: Whether the customer has a phone service (yes, no)
    * `Multiple`: Whether the customer has multiple lines (yes, no)
    * `OnlineSecurity`: Whether the customer has an online security add-on (yes, no)
    * `OnlineBackup`: Whether the customer has an online backup add-on (yes, no)
    * `DeviceProtection`: Whether the customer has device protection add-on (yes, no)
    * `TechSupport`: Whether the customer has a technical support add-on (yes, no)
    * `StreamingTV`: Whether the customer has streaming TV (yes, no)
    * `StreamingMovies`: Whether the customer has streaming movies (yes, no)
    * `PaperlessBilling`: Whether the customer has paperless billing (yes, no)
  - The following six **nominal categorical variables** are re-expressed below using one-hot encoding:
    * `Area`: Area type (rural, urban, suburban)
    * `Marital`: Marital status of customer
    * `Gender`: Customer self-identification as male, female, or nonbinary
    * `Contract`: The contract term of the customer (month-to-month, one year, two year)
    * `InternetService`: Customer's internet service provider (DSL, fiber optic, None)
    * `PaymentMethod`: The customer's payment method (electronic check, mailed check, bank (automatic bank transfer), credit card (automatic))
  - If variables have too many unique values, re-expressing them will increase the dimensionality of the model (Middleton, 2022). For this reason, the following categorical variables will be omitted from the model:
    * `City`: Customer city of residence
    * `State`: Customer state of residence
    * `County`: Customer county of residence
    * `Zip`: Customer zip code of residence
    * `TimeZone`: Time zone of customer residence based on customer sign-up information
    * `Job`: Job of the customer/invoiced person

[1]:
```python
## C1 The following cells include the annotated code used to clean the data.
# See code attached, in D208_PA_MendezD_Task1_Revision1.ipynb



# Import the Pandas library, then load the data into a data frame with Pandas' .
 ↪read_csv() function
import pandas as pd
df = pd.read_csv('/Users/drewmendez/Documents/WGU/D208/churn_d208/churn_clean.
 ↪csv')

def printDupesNulls(data_frame):
# Detect duplicates with Pandas' .duplicated method chained with .sum() method.
# Identify missing values in the data frame with Pandas' .isnull() method,
# then sum the resulting series with the .sum() method

    duplicate_count = data_frame.duplicated().sum()
    missing_values_count = data_frame.isnull().sum()
    print('Number of duplicate rows:', duplicate_count)
    print("Number of missing values per variable:")
    print(missing_values_count)

printDupesNulls(df)
```

```
Number of duplicate rows: 0
Number of missing values per variable:
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         2129
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
```

[2]:
```
## C1 Treatment of NAs

# Since the 'InternetService' variable has 'None' as one of its options,
# it is necessary to impute 'None'

df['InternetService'].fillna('None', inplace=True)

# Verify that 'None' no longer appears as 'Null'
print('Number of `InternetService` nulls:', df['Tenure'].isnull().sum())
```

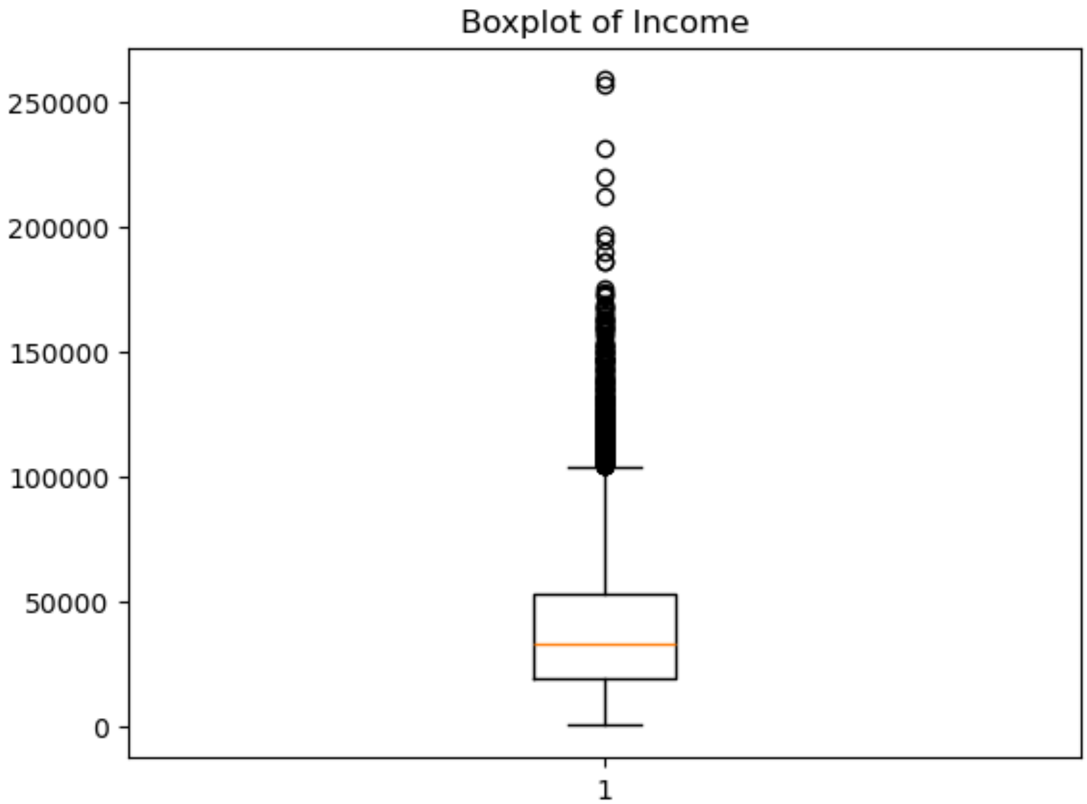
```
Number of `InternetService` nulls: 0
```

[3]:
```
## C1 Detect and Count Outliers of Numeric Variables

import matplotlib.pyplot as plt

def boxplotOutliers(data_frame, col_name):
# Visualize outliers using boxplot() from matplotlib
# First and third quartiles, Q1 and Q3, are found using .quantile() from Pandas,
```

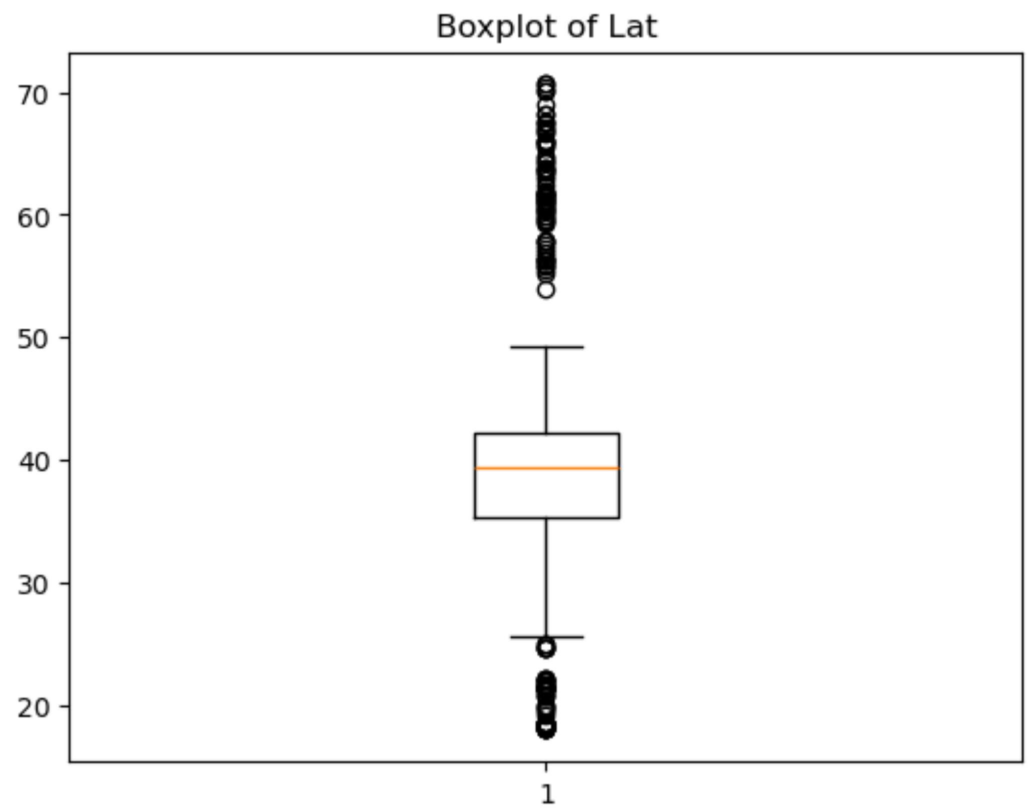
```
# then the interquartile range is found using IQR = Q3 - Q1.
# The upper whisker of the boxplot is found using max = Q3 + 1.5 * IQR.
# The lower whisker of the boxplot is found using min = Q1 - 1.5 * IQR.
# The .sum() method returns the count of observations greater than the max or␣
↪less than the min.
# The .round() method rounds the outlier count to two decimals.
# If loop to print corresponding outputs

    plt.boxplot(data_frame[col_name])
    plt.title(f'Boxplot of {col_name}')
    plt.show()

    Q1 = data_frame[col_name].quantile(0.25)
    Q3 = data_frame[col_name].quantile(0.75)
    IQR = Q3 - Q1
    maximum = round(Q3 + 1.5 * IQR, 2)
    minimum = round(Q1 - 1.5 * IQR, 2)
    outlier_count_up = (data_frame[col_name] > maximum).sum()
    outlier_count_low = (data_frame[col_name] < minimum).sum()


    if outlier_count_up > 0:
        if outlier_count_low > 0:
            print(f'For the `{col_name}` variable, all observations greater␣
↪than {maximum} or less than {minimum} are considered outliers.')
            print(f'The count of observations greater than {maximum} is␣
↪{outlier_count_up}.')
            print(f'The count of observations less than {minimum} is␣
↪{outlier_count_low}.')
        if outlier_count_low == 0:
            print(f'For the `{col_name}` variable, all observations greater␣
↪than {maximum} are considered outliers.')
            print(f'The count of observations greater than {maximum} is␣
↪{outlier_count_up}.')
    if outlier_count_up == 0:
        if outlier_count_low > 0:
            print(f'For the `{col_name}` variable, all observations less than␣
↪{minimum} are considered outliers.')
            print(f'The count of observations less than {minimum} is␣
↪{outlier_count_low}.')
        if outlier_count_low == 0:
            print(f'There are no outliers for the `{col_name}` variable.')
```

[4]:
```
# C1 Detection of Outliers for 13 Numeric Variables
```

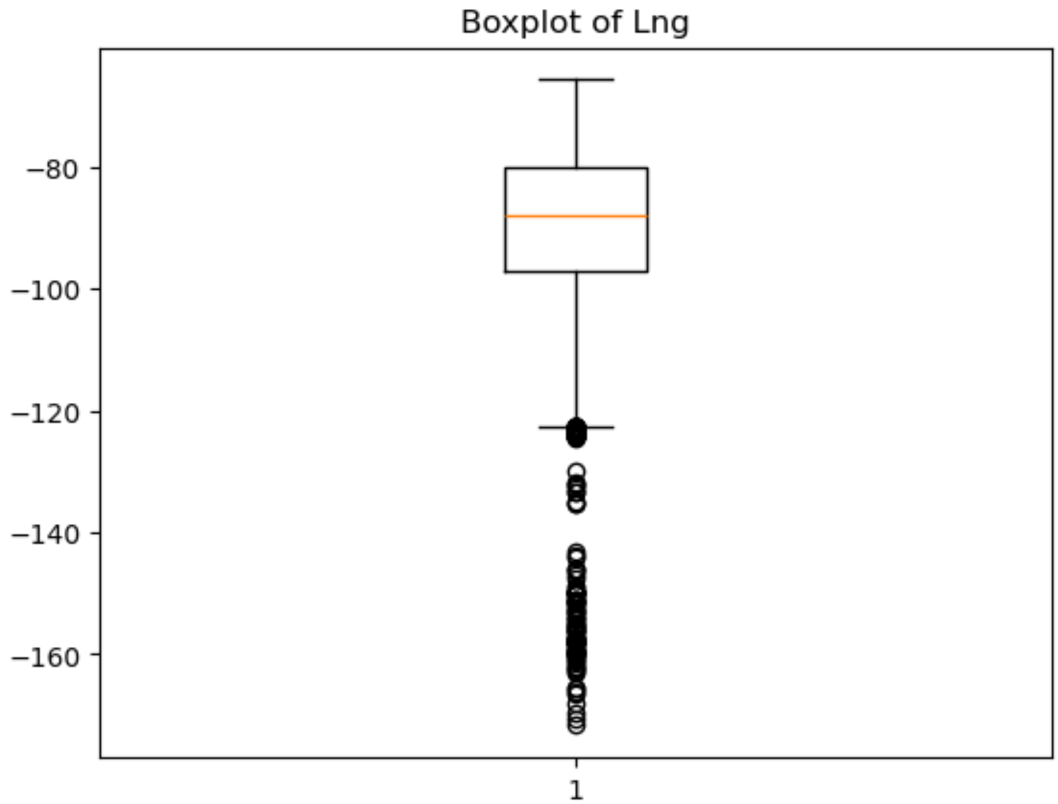
```
boxplotOutliers(df, 'Income')

boxplotOutliers(df, 'Lat')

boxplotOutliers(df, 'Lng')

boxplotOutliers(df, 'Population')

boxplotOutliers(df, 'Children')

boxplotOutliers(df, 'Age')

boxplotOutliers(df, 'Outage_sec_perweek')

boxplotOutliers(df, 'Email')

boxplotOutliers(df, 'Contacts')

boxplotOutliers(df, 'Yearly_equip_failure')

boxplotOutliers(df, 'Tenure')

boxplotOutliers(df, 'MonthlyCharge')

boxplotOutliers(df, 'Bandwidth_GB_Year')
```



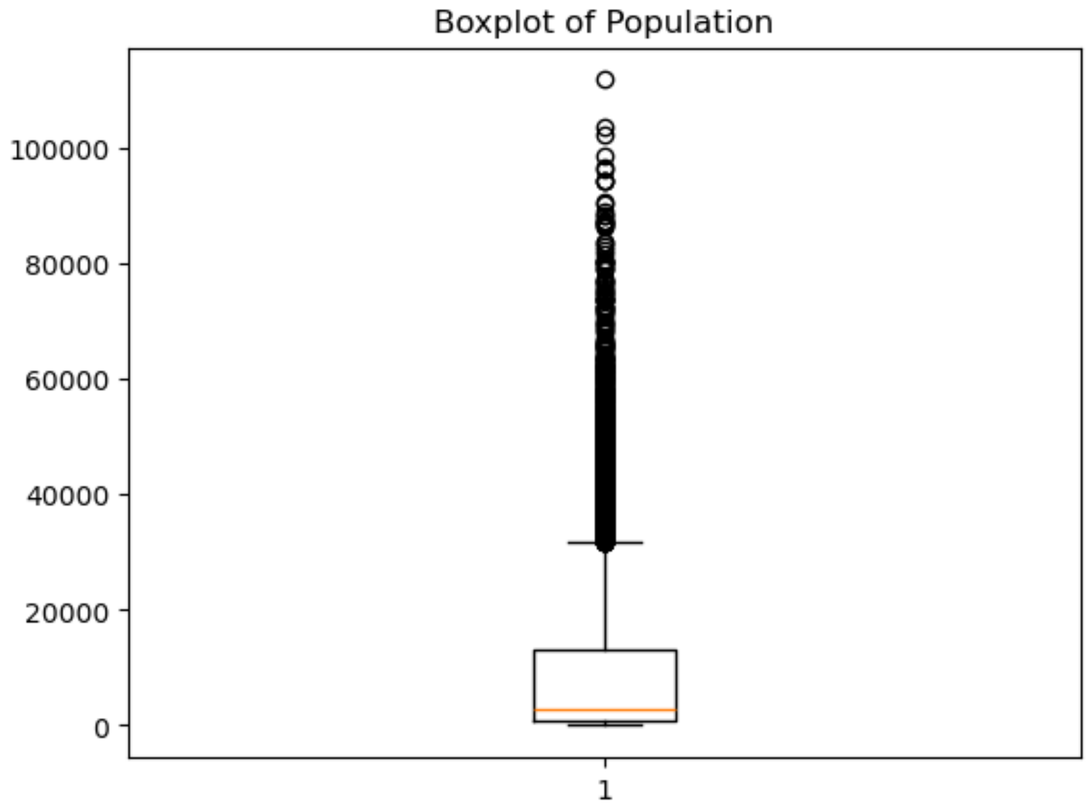

```
For the `Income` variable, all observations greater than 104278.35 are
considered outliers.
The count of observations greater than 104278.35 is 336.
```



For the `Lat` variable, all observations greater than 52.25 or less than 25.19 are considered outliers.
The count of observations greater than 52.25 is 77.
The count of observations less than 25.19 is 81.

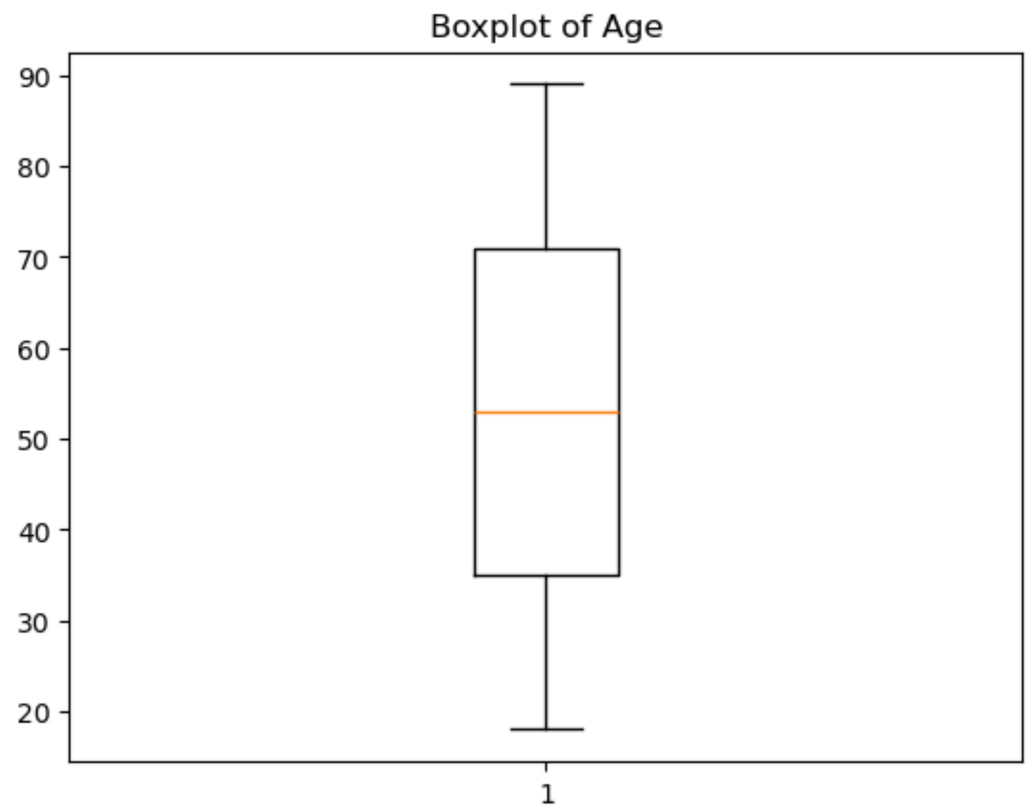


For the `Lng` variable, all observations less than -122.57 are considered outliers.
The count of observations less than -122.57 is 273.

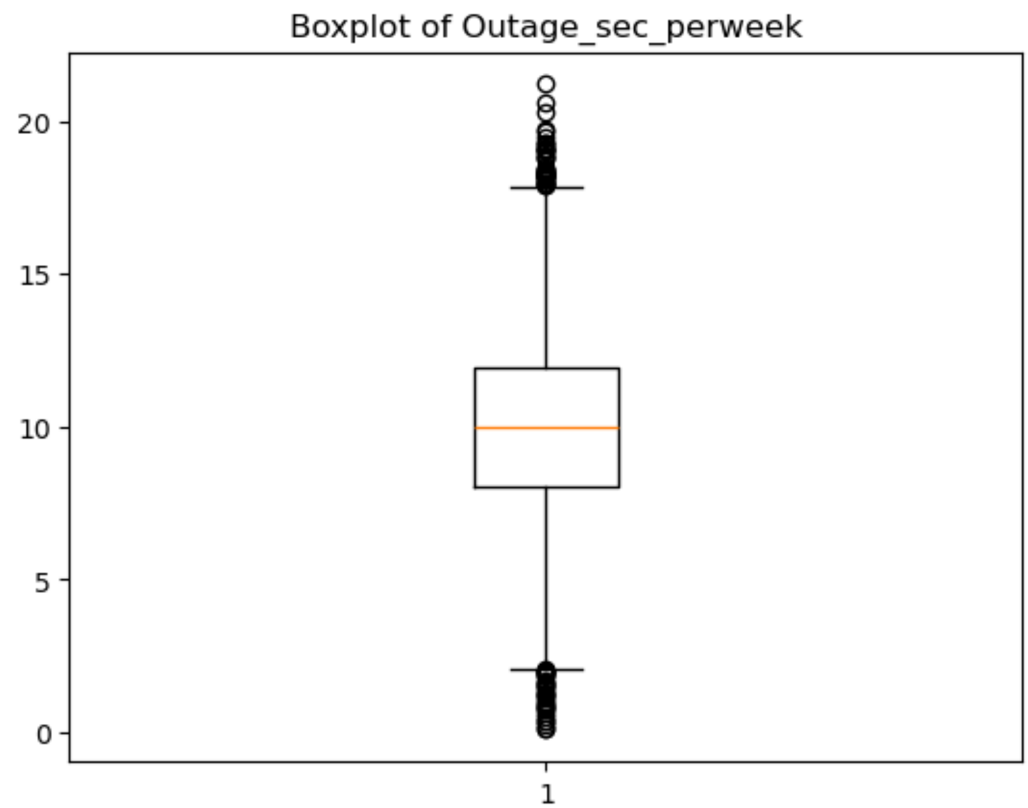


For the `Population` variable, all observations greater than 31813.0 are considered outliers.
The count of observations greater than 31813.0 is 937.

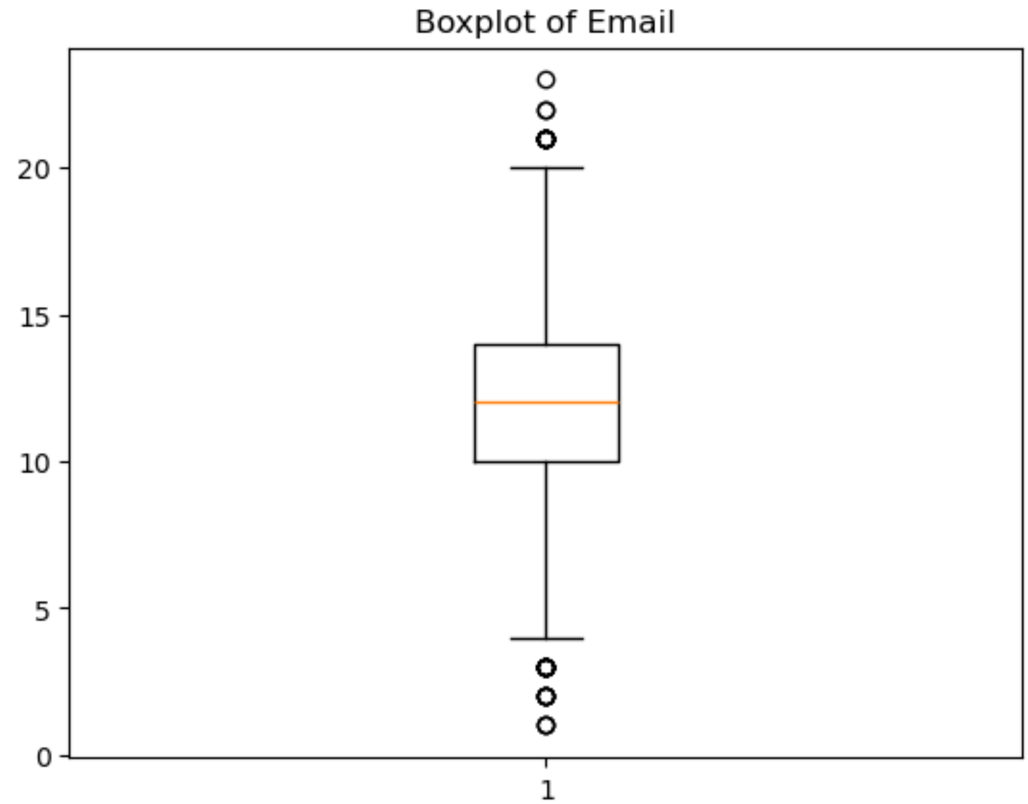
For the `Children` variable, all observations greater than 7.5 are considered outliers.
The count of observations greater than 7.5 is 401.



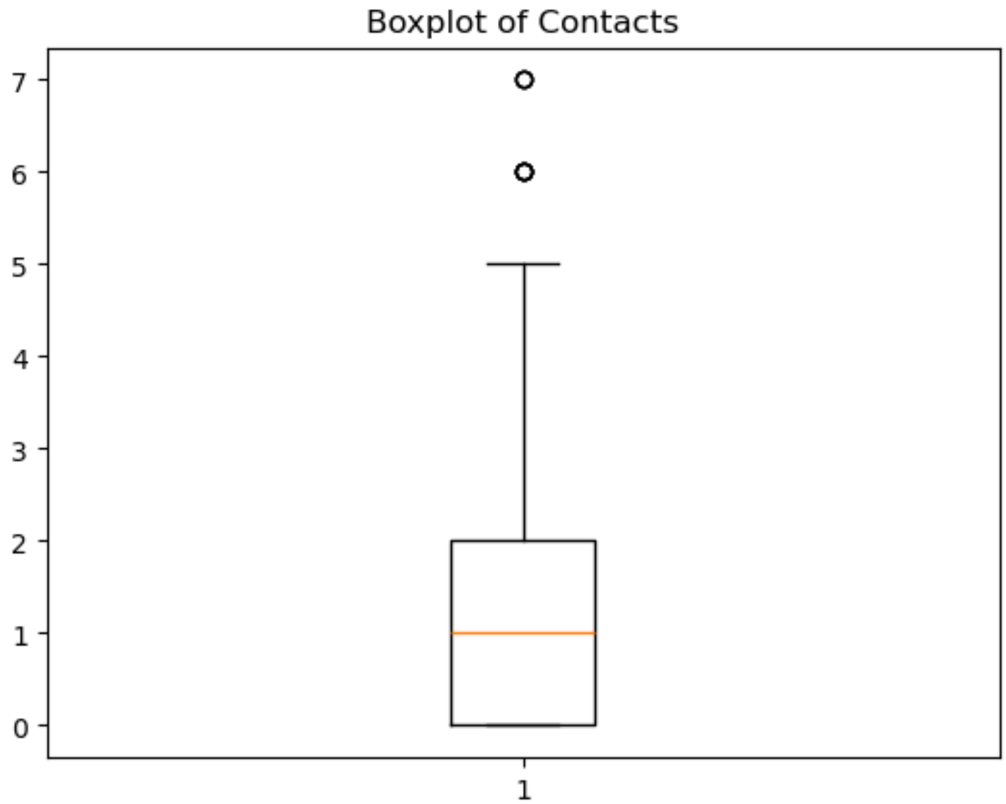
There are no outliers for the `Age` variable.



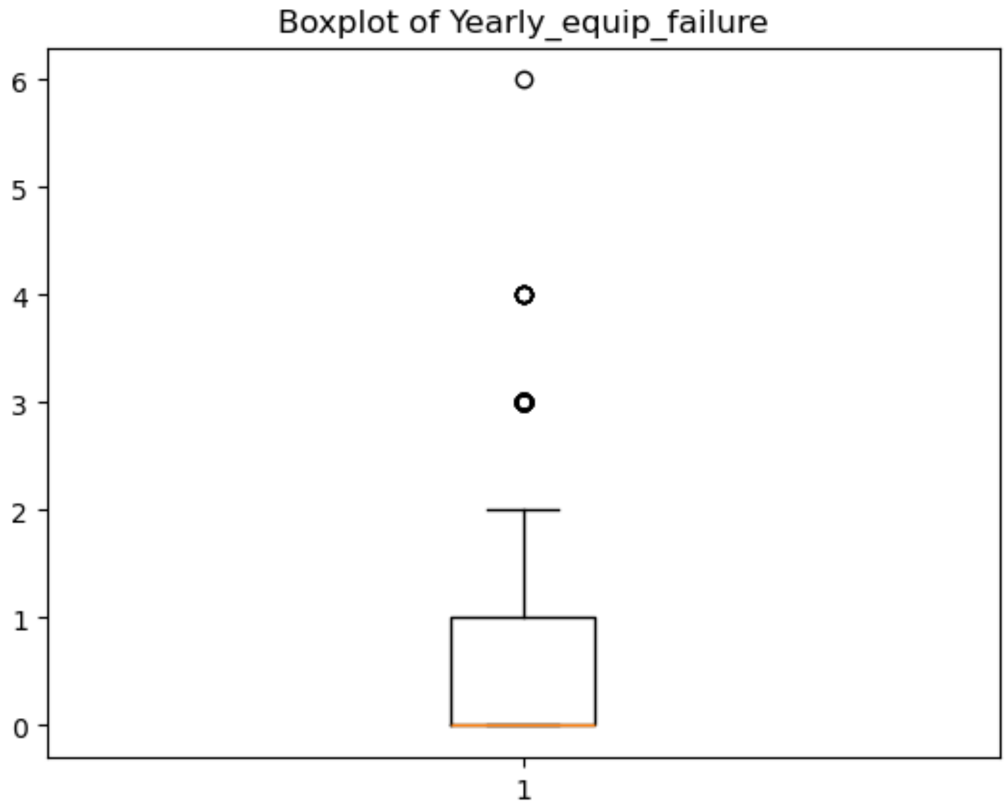
For the `Outage_sec_perweek` variable, all observations greater than 17.9 or less than 2.09 are considered outliers.
The count of observations greater than 17.9 is 43.
The count of observations less than 2.09 is 33.



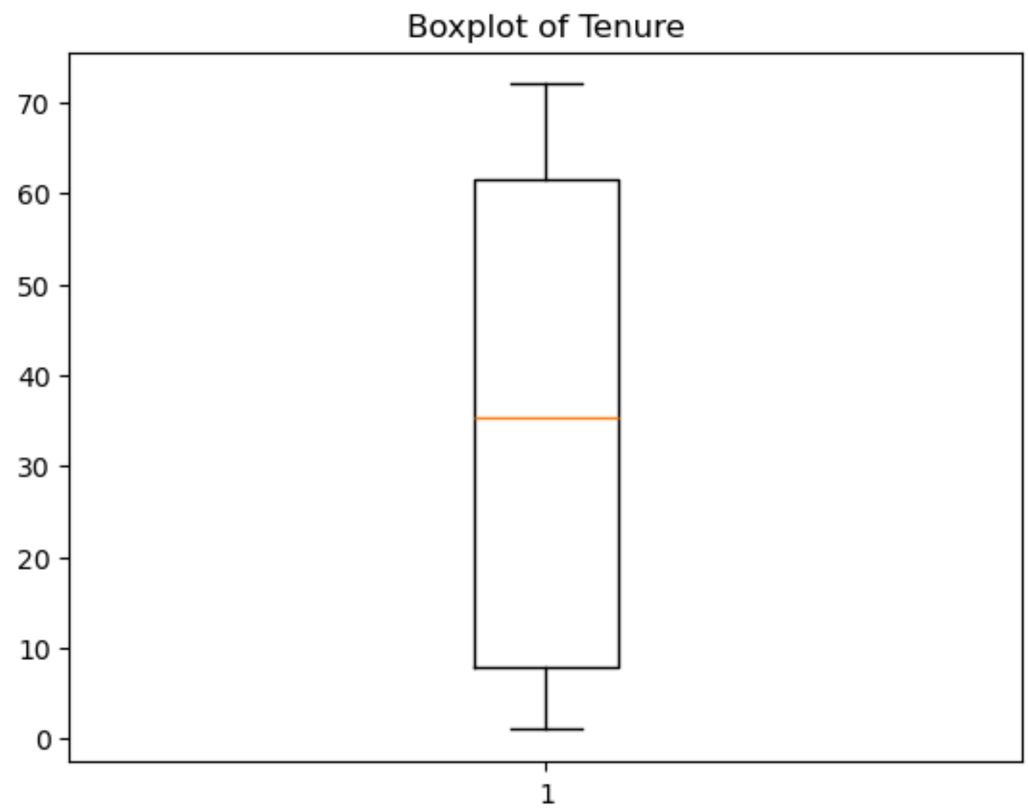

```
For the `Email` variable, all observations greater than 20.0 or less than 4.0
are considered outliers.
The count of observations greater than 20.0 is 15.
The count of observations less than 4.0 is 23.
```



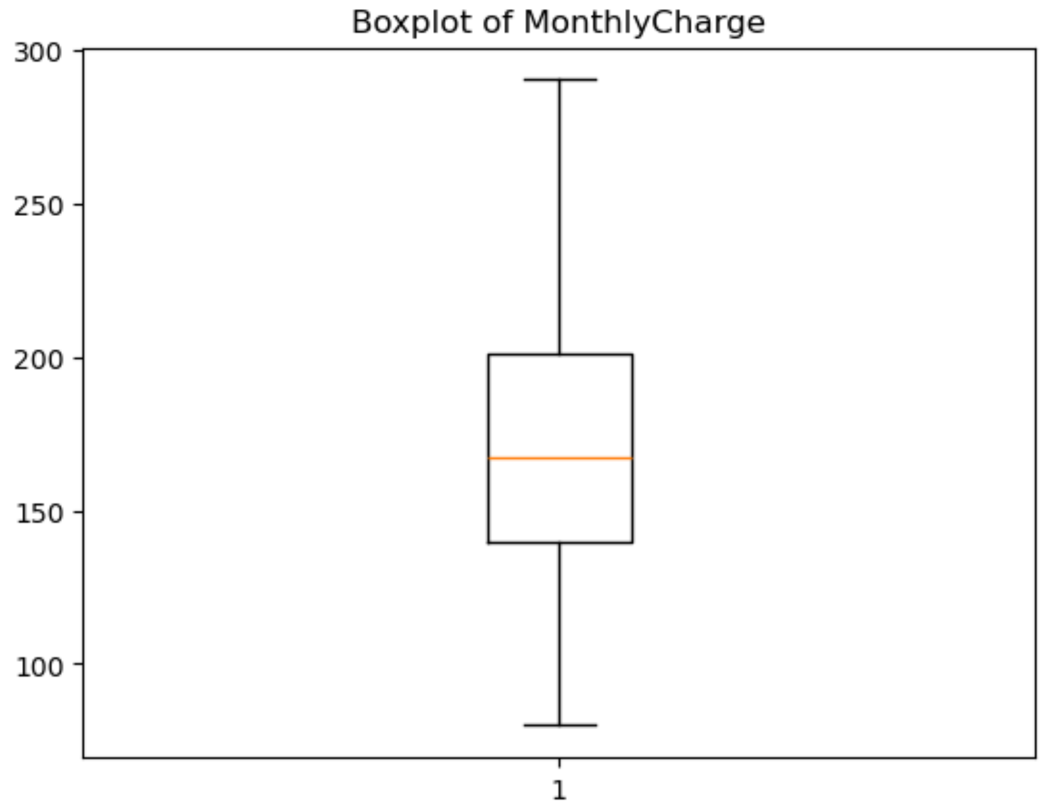
For the `Contacts` variable, all observations greater than 5.0 are considered outliers.
The count of observations greater than 5.0 is 8.



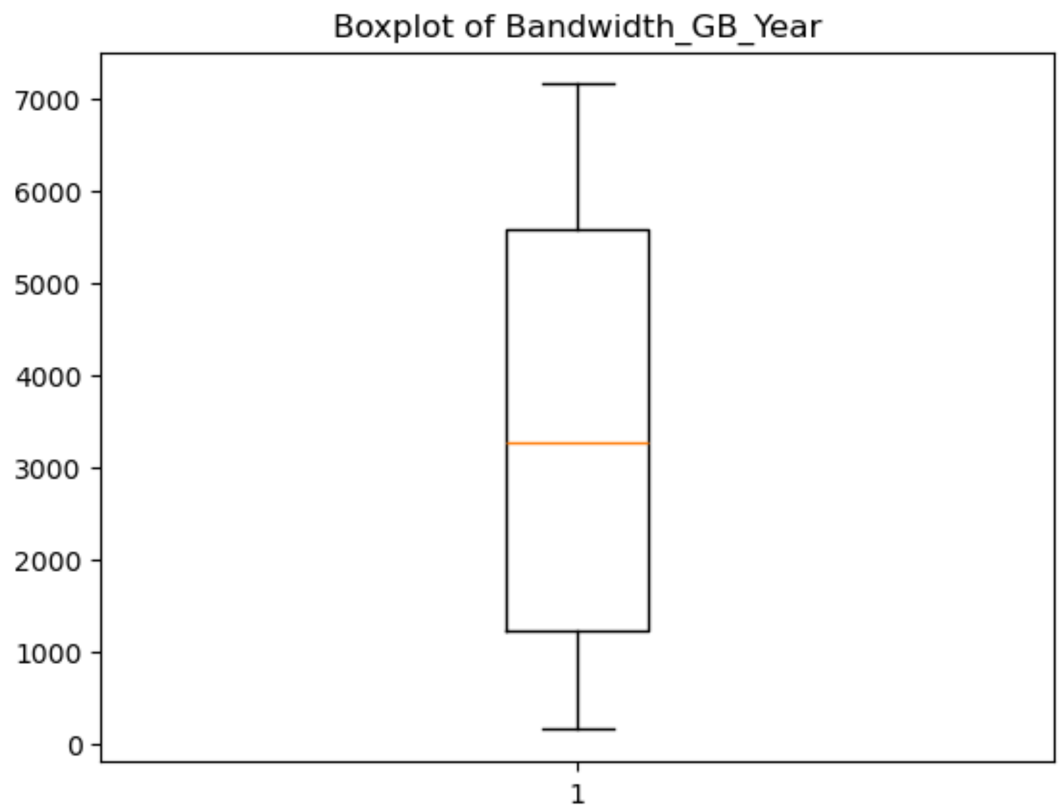
For the `Yearly_equip_failure` variable, all observations greater than 2.5 are considered outliers.
The count of observations greater than 2.5 is 94.



There are no outliers for the `Tenure` variable.



There are no outliers for the `MonthlyCharge` variable.



There are no outliers for the `Bandwidth_GB_Year` variable.

[5]:
```
## C1 Binary Encoding Re-expression of the 12 Binary Variables

# Create a list of the columns that will be encoded
binaryList = ['Churn', 'Techie', 'Port_modem', 'Tablet', 'Phone',
                 'Multiple', 'OnlineSecurity', 'OnlineBackup',␣
 ↪'DeviceProtection',
                 'TechSupport', 'StreamingTV', 'StreamingMovies',␣
 ↪'PaperlessBilling']

# Run a loop that replaces all 'Yes' with 1 and 'No' with 0 for each column in␣
 ↪the list above
for column in binaryList:
    df[column] = df[column].replace({'Yes': 1, 'No': 0})

binaryVars = df[['Churn', 'Techie', 'Port_modem', 'Tablet', 'Phone',
                 'Multiple', 'OnlineSecurity', 'OnlineBackup',␣
 ↪'DeviceProtection',
                 'TechSupport', 'StreamingTV', 'StreamingMovies',␣
 ↪'PaperlessBilling']]
```

[6]:
```
## C1 One-Hot Encoding Re-expression of the Six Categorical Variables

# Create additional data frame from variables being re-expressed
oneHotVars = df[['Area', 'Marital', 'Gender', 'Contract', 'InternetService',␣
↪'PaymentMethod']]

# Apply one-hot encoding, dropping first column to avoid multi-collinearity in␣
↪the model
oneHotVars = pd.get_dummies(oneHotVars, drop_first = True, dtype = int)

# Concatenate one-hot data frame with original data frame
# retain original variables for summaries
df = pd.concat([df, oneHotVars], axis=1)
```

3.1.2 C2. Summary Statistics of Dependent Variable and Independent Variables

The dependent variable for this multiple regression model will be the continuous variable `Income`.

The independent variables for the initial model will be:

- all twelve numeric variables
  - `Lat`, `Lng`, `Population`, `Children`, `Age`, `Outage_sec_perweek`, `Email`, `Contacts`, `Yearly_equip_failure`, `Tenure`, `MonthlyCharge`, `Bandwidth_GB_Year`
- all thirteen re-expressed binary variables
  - `Churn`, `Techie`, `Port_modem`, `Tablet`, `Phone`, `Multiple`, `OnlineSecurity`, `OnlineBackup`, `DeviceProtection`, `TechSupport`, `StreamingTV`, `StreamingMovies`, `PaperlessBilling`
- all six categorical variables re-expressed with one-hot encoding
  - `Area`, `Marital`, `Gender`, `Contract`, `InternetService`, `PaymentMethod`
- all eight ordinal variables
  - `Item1`, `Item2`, `Item3`, `Item4`, `Item5`, `Item6`, `Item7`, `Item8`

The written descriptions of the summary statistics for these variables are shown below.

[7]:
```
## C2 Summaries of Numeric and Ordinal Variables

print('Summary of Dependent Variable `Income`')
print(df['Income'].describe())

numericVars = df[['Lat', 'Lng', 'Population', 'Children', 'Age',␣
↪'Outage_sec_perweek', 'Email',
          'Contacts', 'Yearly_equip_failure', 'Tenure', 'MonthlyCharge',␣
↪'Bandwidth_GB_Year']]

ordinalVars = df[['Item1', 'Item2', 'Item3', 'Item4', 'Item5', 'Item6',␣
↪'Item7', 'Item8']]

print('Summaries of Independent Variables')
print(numericVars.describe())
```

```python
print('Summaries of Ordinal Variables')
print(ordinalVars.describe())
```

```
Summary of Dependent Variable `Income`
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
Summaries of Independent Variables
               Lat           Lng     Population    Children           Age  \
count  10000.000000  10000.000000   10000.000000  10000.0000  10000.000000
mean      38.757567    -90.782536    9756.562400      2.0877     53.078400
std        5.437389     15.156142   14432.698671      2.1472     20.698882
min       17.966120   -171.688150       0.000000      0.0000     18.000000
25%       35.341828    -97.082812     738.000000      0.0000     35.000000
50%       39.395800    -87.918800    2910.500000      1.0000     53.000000
75%       42.106908    -80.088745   13168.000000      3.0000     71.000000
max       70.640660    -65.667850  111850.000000     10.0000     89.000000

       Outage_sec_perweek         Email      Contacts  Yearly_equip_failure  \
count        10000.000000  10000.000000  10000.000000          10000.000000
mean            10.001848     12.016000      0.994200              0.398000
std              2.976019      3.025898      0.988466              0.635953
min              0.099747      1.000000      0.000000              0.000000
25%              8.018214     10.000000      0.000000              0.000000
50%             10.018560     12.000000      1.000000              0.000000
75%             11.969485     14.000000      2.000000              1.000000
max             21.207230     23.000000      7.000000              6.000000

             Tenure  MonthlyCharge  Bandwidth_GB_Year
count  10000.000000   10000.000000       10000.000000
mean      34.526188     172.624816        3392.341550
std       26.443063      42.943094        2185.294852
min        1.000259      79.978860         155.506715
25%        7.917694     139.979239        1236.470827
50%       35.430507     167.484700        3279.536903
75%       61.479795     200.734725        5586.141370
max       71.999280     290.160419        7158.981530
Summaries of Ordinal Variables
              Item1         Item2         Item3         Item4         Item5  \
count  10000.000000  10000.000000  10000.000000  10000.000000  10000.000000
mean       3.490800      3.505100      3.487000      3.497500      3.492900
```

```
std        1.037797      1.034641      1.027977      1.025816      1.024819
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      4.000000      3.000000      3.000000      3.000000
75%        4.000000      4.000000      4.000000      4.000000      4.000000
max        7.000000      7.000000      8.000000      7.000000      7.000000

              Item6         Item7         Item8
count  10000.000000  10000.000000  10000.000000
mean       3.497300      3.509500      3.495600
std        1.033586      1.028502      1.028633
min        1.000000      1.000000      1.000000
25%        3.000000      3.000000      3.000000
50%        3.000000      4.000000      3.000000
75%        4.000000      4.000000      4.000000
max        8.000000      7.000000      8.000000
```

[8]:

```
## C2 Summaries of Numeric Variables

def quantDesc(data_frame, col_name):
# Provide written description of the statistical summary output of .describe()

    count, mean, std, minimum, quarter, half, seventyfive, maximum =
 ↪data_frame[col_name].describe()

    print(f'For the variable `{col_name}`:')
    print(f'There are {count} observations.')
    print(f'On average, the data tends towards the mean, which is {round(mean,
 ↪2)}.')
    print(f'The standard deviation {round(std, 2)} is the amount of variation,
 ↪or how much the data differs from the mean.')
    print(f'The smallest observation is {round(minimum, 2)}.')
    print(f'25% of the data falls below {round(quarter, 2)}.')
    print(f'50% of the data falls below {round(half, 2)}.')
    print(f'75% of the data falls below {round(seventyfive, 2)}.')
    print(f'100% of the data falls below the largest observation,
 ↪{round(maximum, 2)}.')
    print('\n')
```

[9]:

```
## C2 Summaries of 13 Numeric Variables

quantDesc(df, 'Income')

quantDesc(df, 'Population')

quantDesc(df, 'Children')
```

```
quantDesc(df, 'Age')

quantDesc(df, 'Outage_sec_perweek')

quantDesc(df, 'Email')

quantDesc(df, 'Contacts')

quantDesc(df, 'Yearly_equip_failure')

quantDesc(df, 'Tenure')

quantDesc(df, 'MonthlyCharge')

quantDesc(df, 'Bandwidth_GB_Year')

quantDesc(df, 'Lat')

quantDesc(df, 'Lng')
```

```
For the variable `Income`:
There are 10000.0 observations.
On average, the data tends towards the mean, which is 39806.93.
The standard deviation 28199.92 is the amount of variation, or how much the data
differs from the mean.
The smallest observation is 348.67.
25% of the data falls below 19224.72.
50% of the data falls below 33170.6.
75% of the data falls below 53246.17.
100% of the data falls below the largest observation, 258900.7.


For the variable `Population`:
There are 10000.0 observations.
On average, the data tends towards the mean, which is 9756.56.
The standard deviation 14432.7 is the amount of variation, or how much the data
differs from the mean.
The smallest observation is 0.0.
25% of the data falls below 738.0.
50% of the data falls below 2910.5.
75% of the data falls below 13168.0.
100% of the data falls below the largest observation, 111850.0.


For the variable `Children`:
There are 10000.0 observations.
On average, the data tends towards the mean, which is 2.09.
The standard deviation 2.15 is the amount of variation, or how much the data
```

differs from the mean.
The smallest observation is 0.0.
25% of the data falls below 0.0.
50% of the data falls below 1.0.
75% of the data falls below 3.0.
100% of the data falls below the largest observation, 10.0.

For the variable `Age`:
There are 10000.0 observations.
On average, the data tends towards the mean, which is 53.08.
The standard deviation 20.7 is the amount of variation, or how much the data differs from the mean.
The smallest observation is 18.0.
25% of the data falls below 35.0.
50% of the data falls below 53.0.
75% of the data falls below 71.0.
100% of the data falls below the largest observation, 89.0.

For the variable `Outage_sec_perweek`:
There are 10000.0 observations.
On average, the data tends towards the mean, which is 10.0.
The standard deviation 2.98 is the amount of variation, or how much the data differs from the mean.
The smallest observation is 0.1.
25% of the data falls below 8.02.
50% of the data falls below 10.02.
75% of the data falls below 11.97.
100% of the data falls below the largest observation, 21.21.

For the variable `Email`:
There are 10000.0 observations.
On average, the data tends towards the mean, which is 12.02.
The standard deviation 3.03 is the amount of variation, or how much the data differs from the mean.
The smallest observation is 1.0.
25% of the data falls below 10.0.
50% of the data falls below 12.0.
75% of the data falls below 14.0.
100% of the data falls below the largest observation, 23.0.

For the variable `Contacts`:
There are 10000.0 observations.
On average, the data tends towards the mean, which is 0.99.
The standard deviation 0.99 is the amount of variation, or how much the data

differs from the mean.
The smallest observation is 0.0.
25% of the data falls below 0.0.
50% of the data falls below 1.0.
75% of the data falls below 2.0.
100% of the data falls below the largest observation, 7.0.

For the variable `Yearly_equip_failure`:
There are 10000.0 observations.
On average, the data tends towards the mean, which is 0.4.
The standard deviation 0.64 is the amount of variation, or how much the data differs from the mean.
The smallest observation is 0.0.
25% of the data falls below 0.0.
50% of the data falls below 0.0.
75% of the data falls below 1.0.
100% of the data falls below the largest observation, 6.0.

For the variable `Tenure`:
There are 10000.0 observations.
On average, the data tends towards the mean, which is 34.53.
The standard deviation 26.44 is the amount of variation, or how much the data differs from the mean.
The smallest observation is 1.0.
25% of the data falls below 7.92.
50% of the data falls below 35.43.
75% of the data falls below 61.48.
100% of the data falls below the largest observation, 72.0.

For the variable `MonthlyCharge`:
There are 10000.0 observations.
On average, the data tends towards the mean, which is 172.62.
The standard deviation 42.94 is the amount of variation, or how much the data differs from the mean.
The smallest observation is 79.98.
25% of the data falls below 139.98.
50% of the data falls below 167.48.
75% of the data falls below 200.73.
100% of the data falls below the largest observation, 290.16.

For the variable `Bandwidth_GB_Year`:
There are 10000.0 observations.
On average, the data tends towards the mean, which is 3392.34.
The standard deviation 2185.29 is the amount of variation, or how much the data

```
differs from the mean.
The smallest observation is 155.51.
25% of the data falls below 1236.47.
50% of the data falls below 3279.54.
75% of the data falls below 5586.14.
100% of the data falls below the largest observation, 7158.98.


For the variable `Lat`:
There are 10000.0 observations.
On average, the data tends towards the mean, which is 38.76.
The standard deviation 5.44 is the amount of variation, or how much the data
differs from the mean.
The smallest observation is 17.97.
25% of the data falls below 35.34.
50% of the data falls below 39.4.
75% of the data falls below 42.11.
100% of the data falls below the largest observation, 70.64.


For the variable `Lng`:
There are 10000.0 observations.
On average, the data tends towards the mean, which is -90.78.
The standard deviation 15.16 is the amount of variation, or how much the data
differs from the mean.
The smallest observation is -171.69.
25% of the data falls below -97.08.
50% of the data falls below -87.92.
75% of the data falls below -80.09.
100% of the data falls below the largest observation, -65.67.
```

[10]:
```
## C2 Summaries of Binary Variables

def binarySummary(data_frame, col_name):
# Get the counts, convert counts to percentages,
# and calculate and display summary statistics

    counts = data_frame[col_name].value_counts()
    percentages = counts / counts.sum() * 100

    summary_stats = pd.DataFrame({'Count': counts, 'Percentage': percentages})
    summary_stats = summary_stats.rename(index={0: 'no', 1: 'yes'})

    print(f'Summary of `{col_name}`')
    print(summary_stats)
```

```
    print('\n')
```

[11]:
```
## C2 Summaries of 13 Binary Variables

binarySummary(df, 'Churn')

binarySummary(df, 'Techie')

binarySummary(df, 'Port_modem')

binarySummary(df, 'Tablet')

binarySummary(df, 'Phone')

binarySummary(df, 'Multiple')

binarySummary(df, 'OnlineSecurity')

binarySummary(df, 'OnlineBackup')

binarySummary(df, 'DeviceProtection')

binarySummary(df, 'TechSupport')

binarySummary(df, 'StreamingTV')

binarySummary(df, 'StreamingMovies')

binarySummary(df, 'PaperlessBilling')
```

```
Summary of `Churn`
       Count  Percentage
Churn
no      7350        73.5
yes     2650        26.5


Summary of `Techie`
        Count  Percentage
Techie
no       8321       83.21
yes      1679       16.79


Summary of `Port_modem`
            Count  Percentage
Port_modem
no           5166       51.66
```

```
yes          4834       48.34


Summary of `Tablet`
        Count  Percentage
Tablet
no       7009       70.09
yes      2991       29.91


Summary of `Phone`
       Count  Percentage
Phone
yes     9067       90.67
no       933        9.33


Summary of `Multiple`
          Count  Percentage
Multiple
no         5392       53.92
yes        4608       46.08


Summary of `OnlineSecurity`
                Count  Percentage
OnlineSecurity
no               6424       64.24
yes              3576       35.76


Summary of `OnlineBackup`
              Count  Percentage
OnlineBackup
no             5494       54.94
yes            4506       45.06


Summary of `DeviceProtection`
                  Count  Percentage
DeviceProtection
no                 5614       56.14
yes                4386       43.86


Summary of `TechSupport`
             Count  Percentage
TechSupport
```

```
no              6250        62.5
yes             3750        37.5


Summary of `StreamingTV`
             Count  Percentage
StreamingTV
no              5071       50.71
yes             4929       49.29


Summary of `StreamingMovies`
                 Count  Percentage
StreamingMovies
no                  5110        51.1
yes                 4890        48.9


Summary of `PaperlessBilling`
                  Count  Percentage
PaperlessBilling
yes                  5882       58.82
no                   4118       41.18
```

[12]:

```
## C2 Summaries of Categorical Variables

def catSummary(data_frame, col_name):
# Get the counts, convert counts to percentages,
# and calculate and display summary statistics

    counts = data_frame[col_name].value_counts()
    percentages = counts / counts.sum() * 100

    summary_stats = pd.DataFrame({'Count': counts, 'Percentage': percentages})

    print(f'Summary of `{col_name}`')
    print(summary_stats)
    print('\n')
```

[13]:

```
## C2 Summaries of Six Categorical Variables and Eight Ordinal Variables

catSummary(df, 'Area')

catSummary(df, 'Marital')
```

```
catSummary(df, 'Gender')

catSummary(df, 'Contract')

catSummary(df, 'InternetService')

catSummary(df, 'PaymentMethod')


catSummary(df, 'Item1')

catSummary(df, 'Item2')

catSummary(df, 'Item3')

catSummary(df, 'Item4')

catSummary(df, 'Item5')

catSummary(df, 'Item6')

catSummary(df, 'Item7')

catSummary(df, 'Item8')
```

```
Summary of `Area`
          Count  Percentage
Area
Suburban   3346       33.46
Urban      3327       33.27
Rural      3327       33.27


Summary of `Marital`
               Count  Percentage
Marital
Divorced        2092       20.92
Widowed         2027       20.27
Separated       2014       20.14
Never Married   1956       19.56
Married         1911       19.11


Summary of `Gender`
           Count  Percentage
Gender
Female      5025       50.25
Male        4744       47.44
```

Nonbinary	231	2.31

Summary of `Contract`

	Count	Percentage
Contract		
Month-to-month	5456	54.56
Two Year	2442	24.42
One year	2102	21.02

Summary of `InternetService`

	Count	Percentage
InternetService		
Fiber Optic	4408	44.08
DSL	3463	34.63
None	2129	21.29

Summary of `PaymentMethod`

	Count	Percentage
PaymentMethod		
Electronic Check	3398	33.98
Mailed Check	2290	22.90
Bank Transfer(automatic)	2229	22.29
Credit Card (automatic)	2083	20.83

Summary of `Item1`

	Count	Percentage
Item1		
3	3448	34.48
4	3358	33.58
2	1393	13.93
5	1359	13.59
1	224	2.24
6	199	1.99
7	19	0.19

Summary of `Item2`

	Count	Percentage
Item2		
3	3415	34.15
4	3412	34.12
5	1368	13.68
2	1360	13.60
1	217	2.17

```
6        215        2.15
7         13        0.13


Summary of `Item3`
       Count  Percentage
Item3
3       3435       34.35
4       3410       34.10
2       1424       14.24
5       1313       13.13
6        203        2.03
1        202        2.02
7         12        0.12
8          1        0.01


Summary of `Item4`
       Count  Percentage
Item4
4       3452       34.52
3       3430       34.30
2       1350       13.50
5       1335       13.35
1        221        2.21
6        203        2.03
7          9        0.09


Summary of `Item5`
       Count  Percentage
Item5
3       3462       34.62
4       3417       34.17
2       1378       13.78
5       1321       13.21
1        206        2.06
6        204        2.04
7         12        0.12


Summary of `Item6`
       Count  Percentage
Item6
3       3445       34.45
4       3333       33.33
2       1427       14.27
5       1382       13.82
```

```
6          210          2.10
1          190          1.90
7           12          0.12
8            1          0.01


Summary of `Item7`
       Count  Percentage
Item7
4       3456       34.56
3       3446       34.46
5       1335       13.35
2       1309       13.09
6        224        2.24
1        219        2.19
7         11        0.11


Summary of `Item8`
       Count  Percentage
Item8
3       3461       34.61
4       3400       34.00
2       1378       13.78
5       1335       13.35
1        206        2.06
6        205        2.05
7         14        0.14
8          1        0.01
```

3.1.3 C3. Univariate and Bivariate Visualizations

The code below generates the univariate and bivariate visualizations of the distributions of the dependent and independent variables. The bivariate visualizations consider `Income` as the dependent variable.

[14]:
```
## C3 Univariate Visualizations

import matplotlib.pyplot as plt


def plot_histobox(data_frame, col_name):
# Overlays a transparent boxplot over a histogram

    data = data_frame[col_name].values
    fig, ax1 = plt.subplots()
```

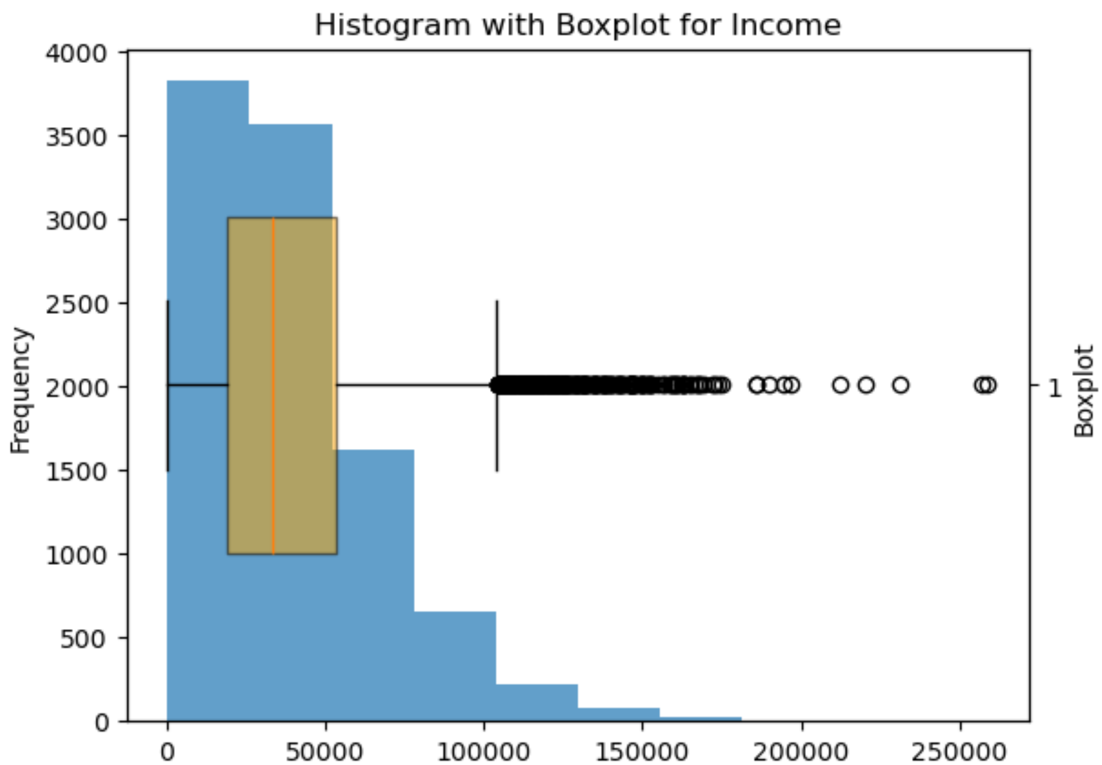
```
    ax1.hist(data, bins = 10, alpha = 0.7, label = 'Histogram')
    ax2 = ax1.twinx()
    ax2.boxplot(data, vert = False, widths = 0.5, patch_artist = True,
                boxprops = dict(facecolor = 'orange', alpha = 0.5))

    ax1.set_ylabel('Frequency')
    ax2.set_ylabel('Boxplot')

    plt.title(f'Histogram with Boxplot for {col_name}')
    plt.show()


def plot_binaryhist(data_frame, col_name):
# Histogram of a binary variable

    catCounts = data_frame[col_name].value_counts()
    cats = catCounts.index.tolist()
    counts = catCounts.values.tolist()

    plt.bar(cats, counts)
    plt.xticks(ticks = [0, 1], labels = ['No', 'Yes'])
    plt.title(f'Histogram of {col_name}')
    plt.xlabel('Category')
    plt.ylabel('Frequency')
    plt.show()


def plot_cathist(data_frame, col_name):
# Histogram of a categorical variable

    catCounts = data_frame[col_name].value_counts()
    cats = catCounts.index.tolist()
    counts = catCounts.values.tolist()

    plt.bar(cats, counts)
    plt.title(f'Histogram of {col_name}')
    plt.xlabel('Category')
    plt.ylabel('Frequency')
    plt.show()
```

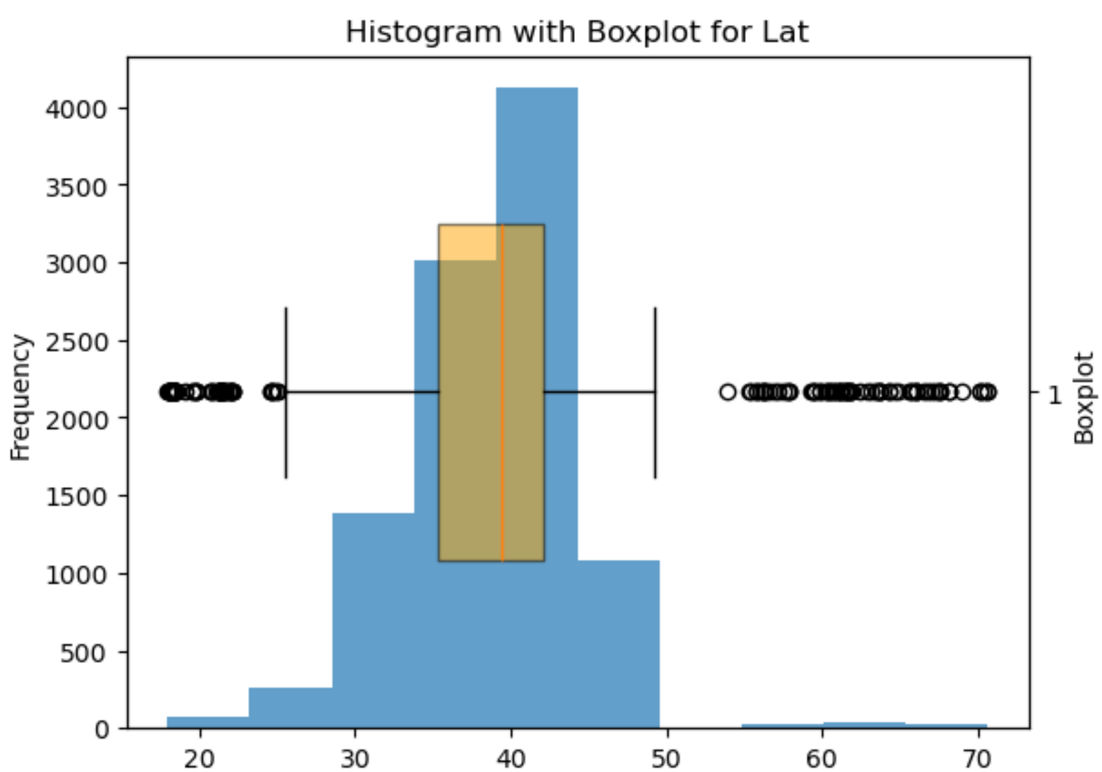
[15]:

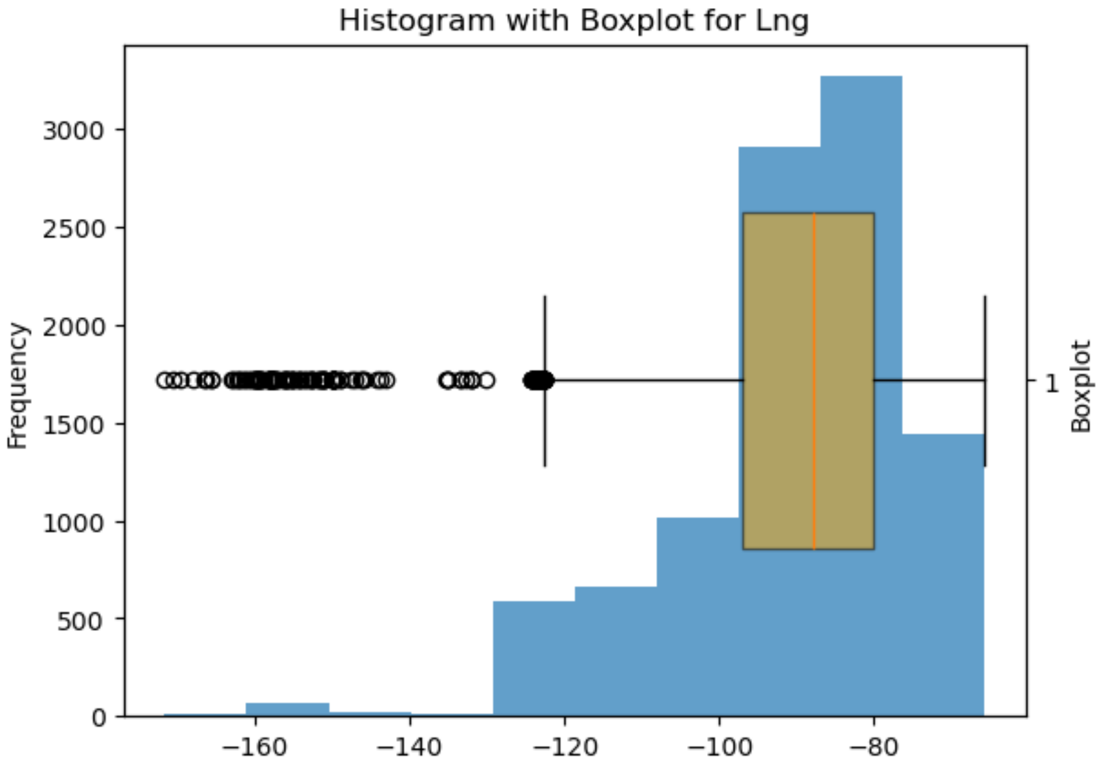
```
## C3 Univariate Visualizations
# Histograms + Boxplots for the dependent variable and 39 all independent␣
↪variables:



plot_histobox(df, 'Income')
```

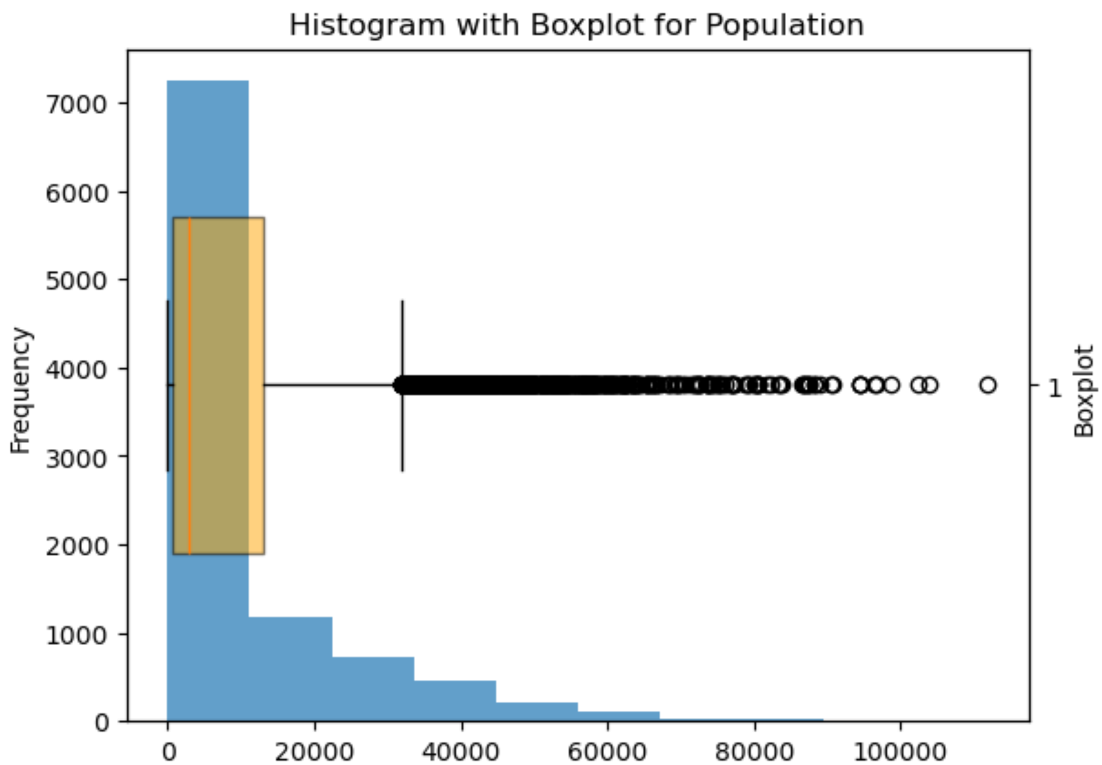
```python
plot_histobox(df, 'Lat')
plot_histobox(df, 'Lng')
plot_histobox(df, 'Population')
plot_histobox(df, 'Children')
plot_histobox(df, 'Age')
plot_histobox(df, 'Outage_sec_perweek')
plot_histobox(df, 'Email')
plot_histobox(df, 'Contacts')
plot_histobox(df, 'Yearly_equip_failure')
plot_histobox(df, 'Tenure')
plot_histobox(df, 'MonthlyCharge')
plot_histobox(df, 'Bandwidth_GB_Year')

plot_binaryhist(df, 'Churn')
plot_binaryhist(df, 'Techie')
plot_binaryhist(df, 'Port_modem')
plot_binaryhist(df, 'Tablet')
plot_binaryhist(df, 'Phone')
plot_binaryhist(df, 'Multiple')
plot_binaryhist(df, 'OnlineSecurity')
plot_binaryhist(df, 'OnlineBackup')
plot_binaryhist(df, 'DeviceProtection')
plot_binaryhist(df, 'TechSupport')
```

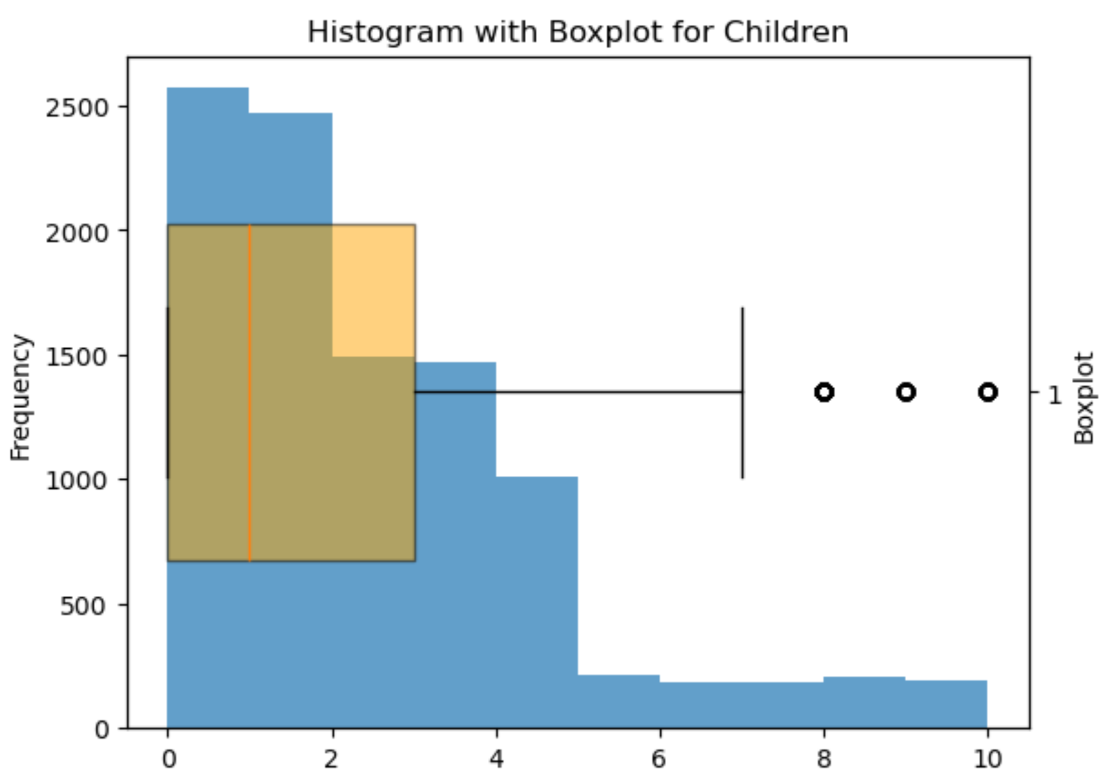
```
plot_binaryhist(df, 'StreamingTV')

plot_binaryhist(df, 'StreamingMovies')

plot_binaryhist(df, 'PaperlessBilling')


plot_cathist(df, 'Area')

plot_cathist(df, 'Marital')

plot_cathist(df, 'Gender')

plot_cathist(df, 'Contract')

plot_cathist(df, 'InternetService')

plot_cathist(df, 'PaymentMethod')


plot_cathist(df, 'Item1')

plot_cathist(df, 'Item2')

plot_cathist(df, 'Item3')

plot_cathist(df, 'Item4')

plot_cathist(df, 'Item5')

plot_cathist(df, 'Item6')

plot_cathist(df, 'Item7')

plot_cathist(df, 'Item8')
```

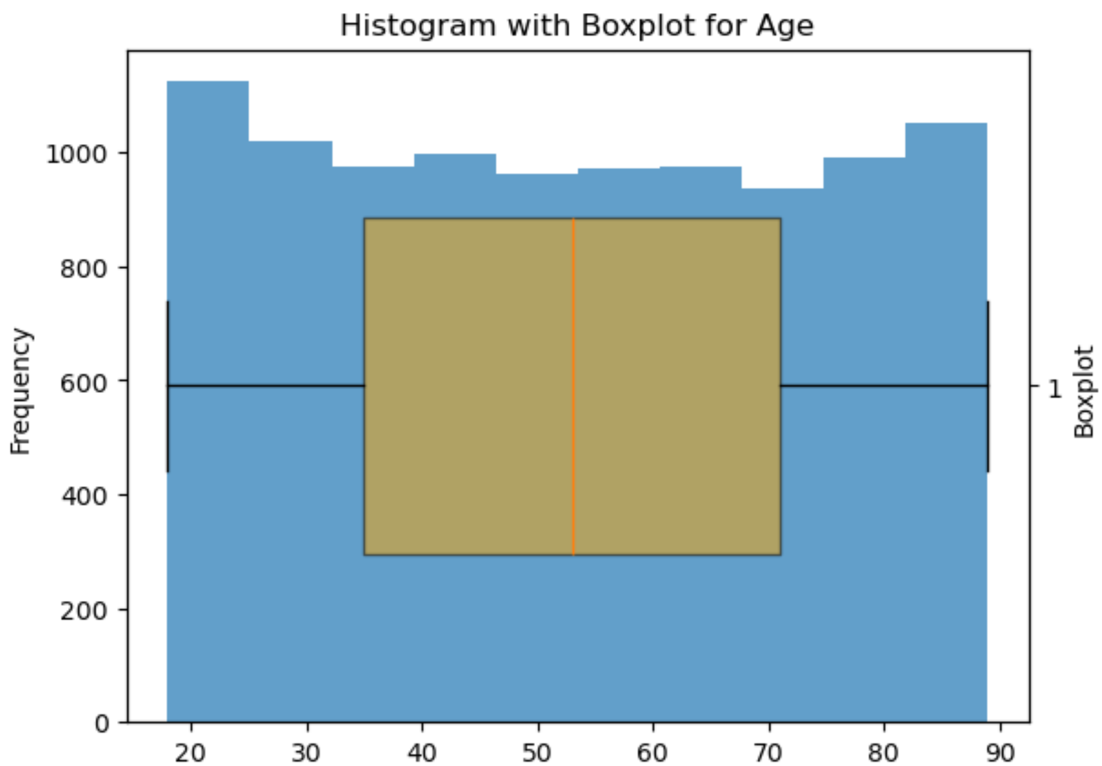

Histogram with Boxplot for Income
Frequency
Boxplot
4000
3500
3000
2500
2000
1500
1000
500
0
1
0
50000
100000
150000
200000
250000

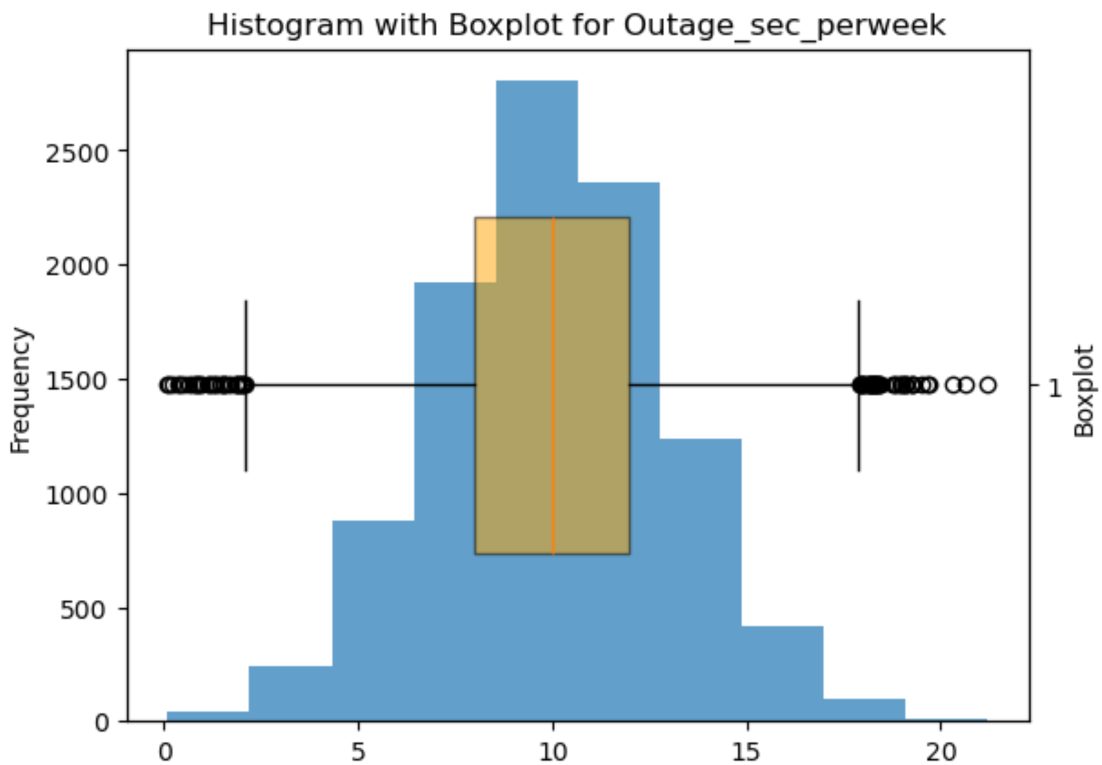

Histogram with Boxplot for Lat
Frequency
Boxplot
4000
3500
3000
2500
2000
1500
1000
500
0
1
20
30
40
50
60
70

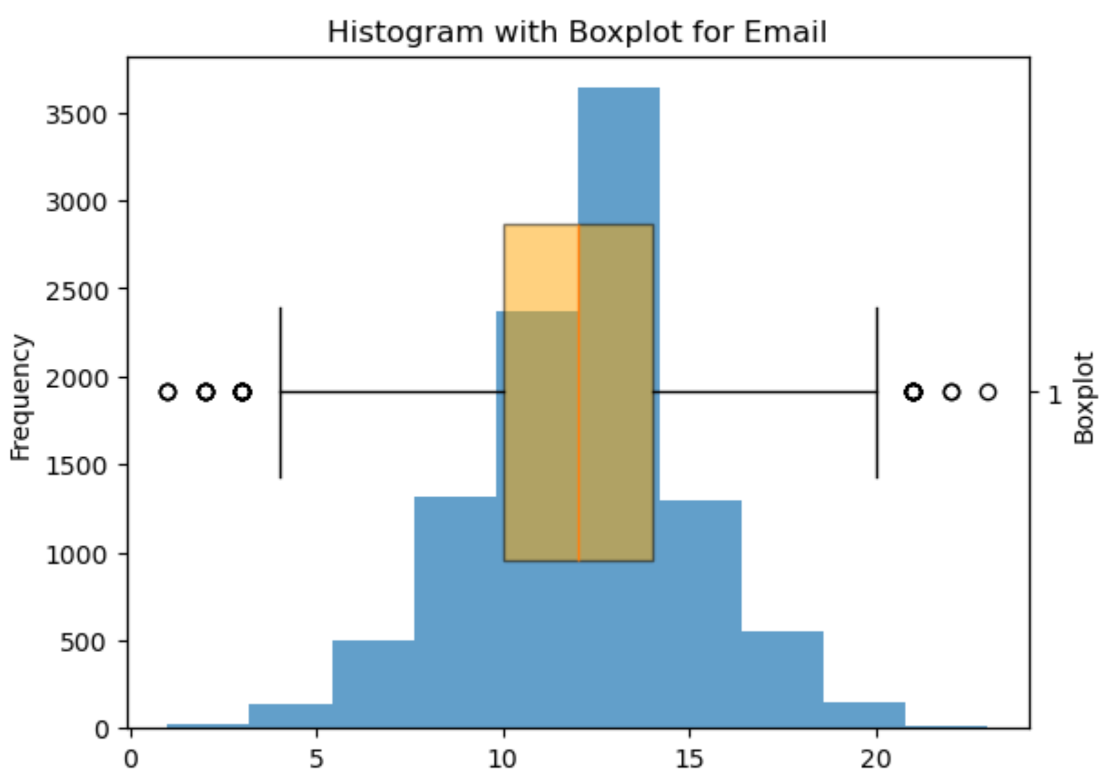

Histogram with Boxplot for Lng
Frequency
Boxplot
0
500
1000
1500
2000
2500
3000
−160
−140
−120
−100
−80
1

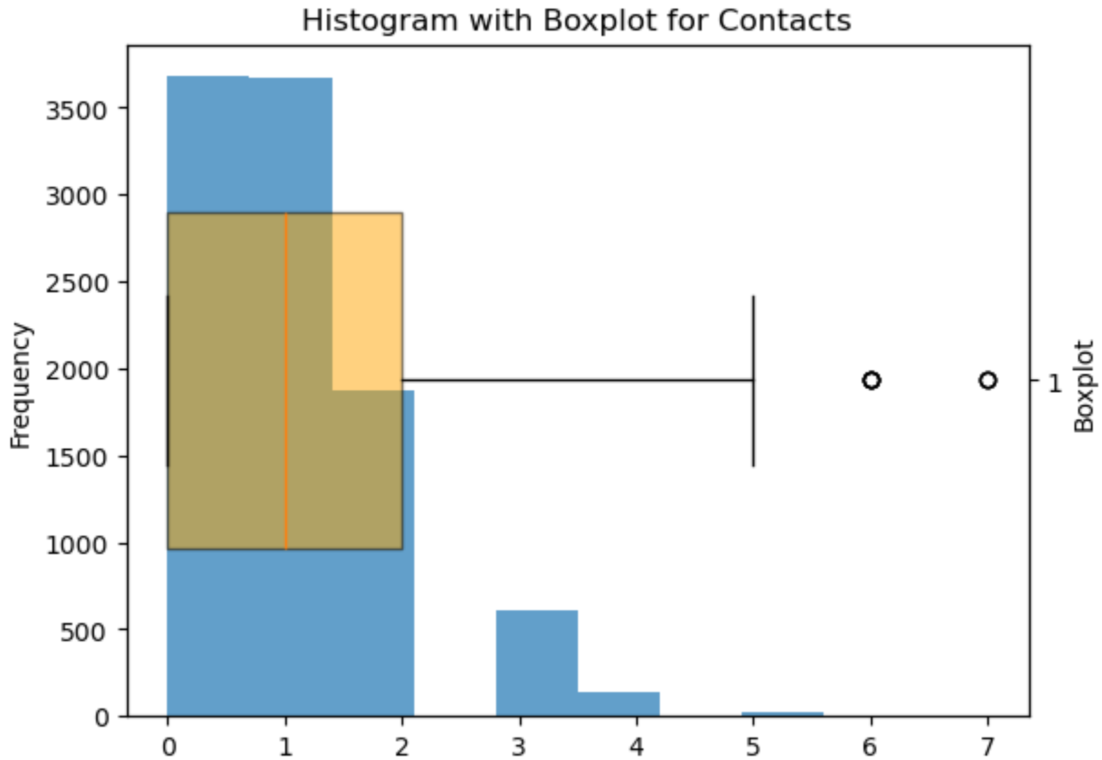

Histogram with Boxplot for Population
Frequency
Boxplot
7000
6000
5000
4000
3000
2000
1000
0
20000
40000
60000
80000
100000
1

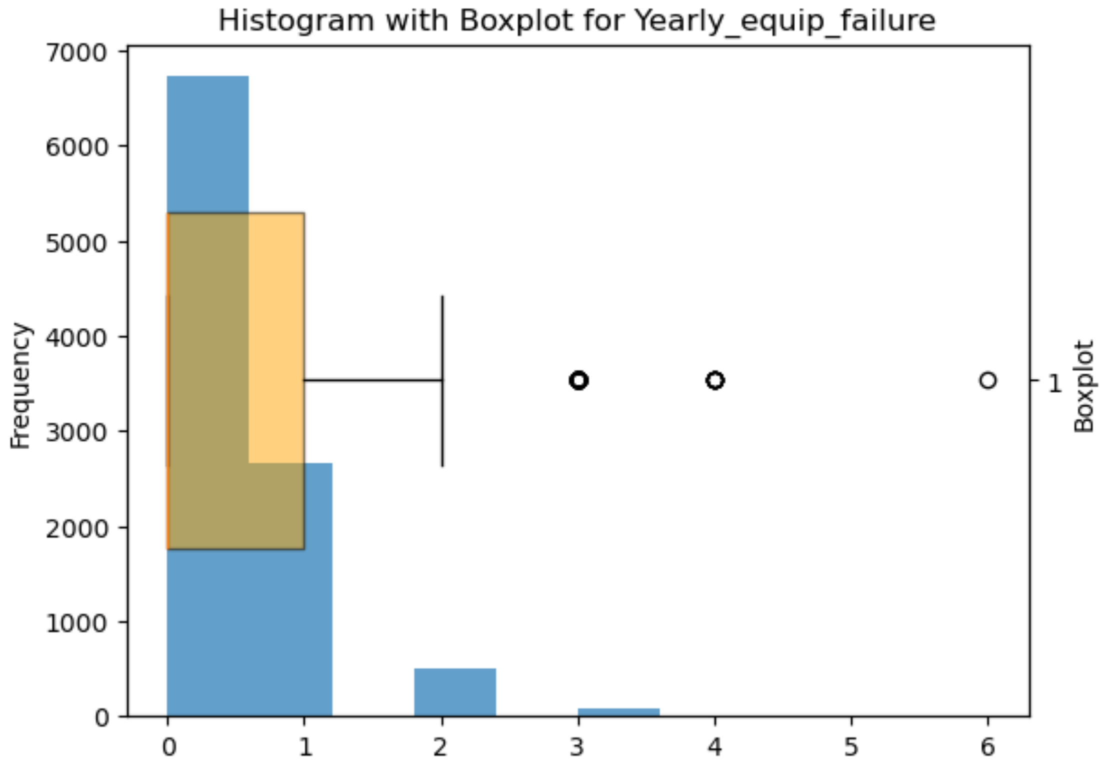

Histogram with Boxplot for Children
Frequency
Boxplot
2500
2000
1500
1000
500
0
2
4
6
8
10
1

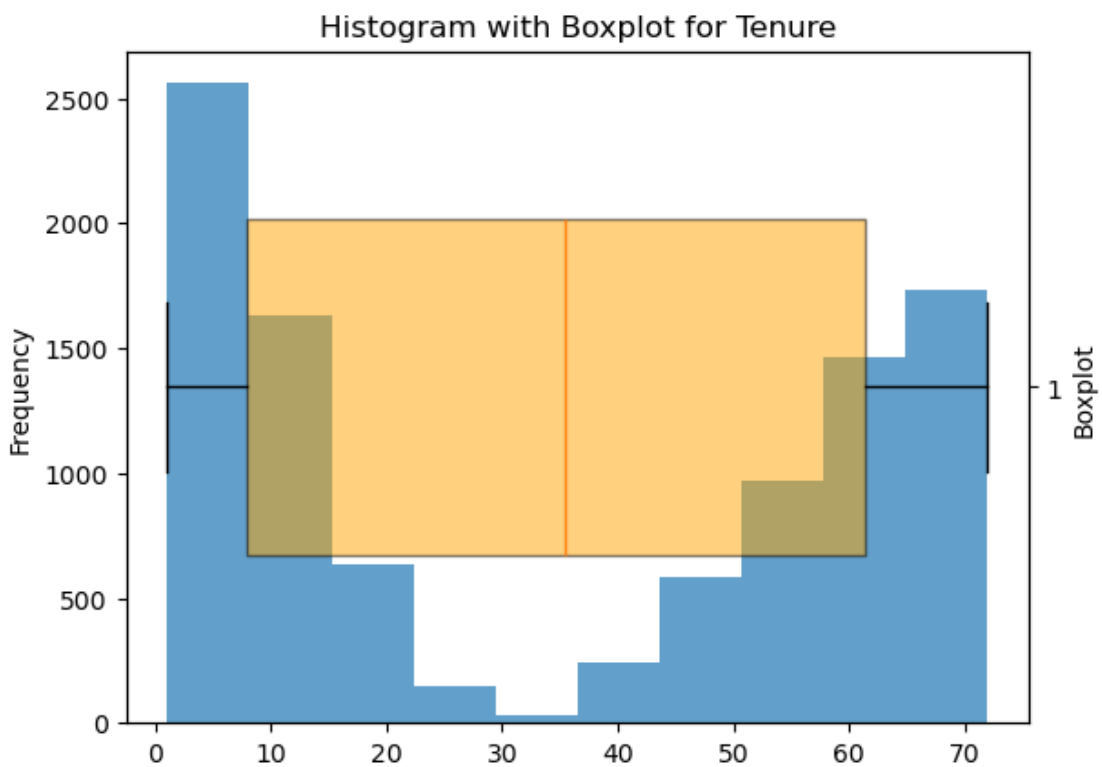

Histogram with Boxplot for Age
Frequency
Boxplot
0
200
400
600
800
1000
1
20
30
40
50
60
70
80
90

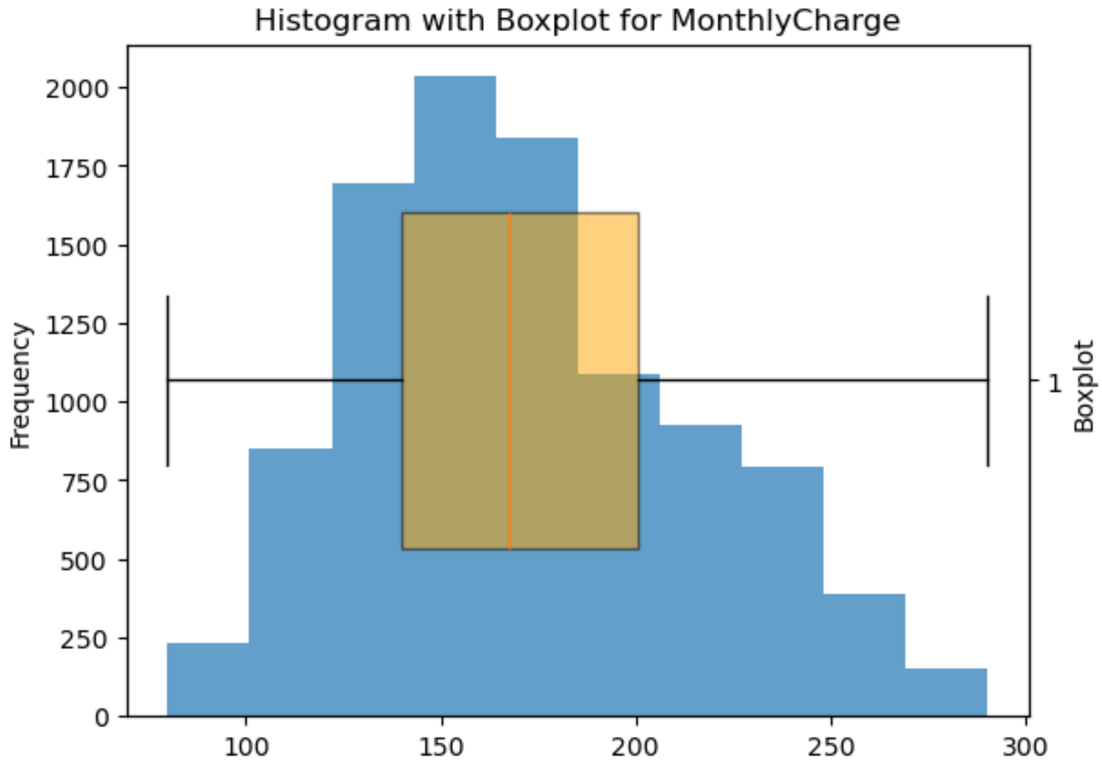

Histogram with Boxplot for Outage_sec_perweek
Frequency
Boxplot
2500
2000
1500
1000
500
0
0
5
10
15
20
1

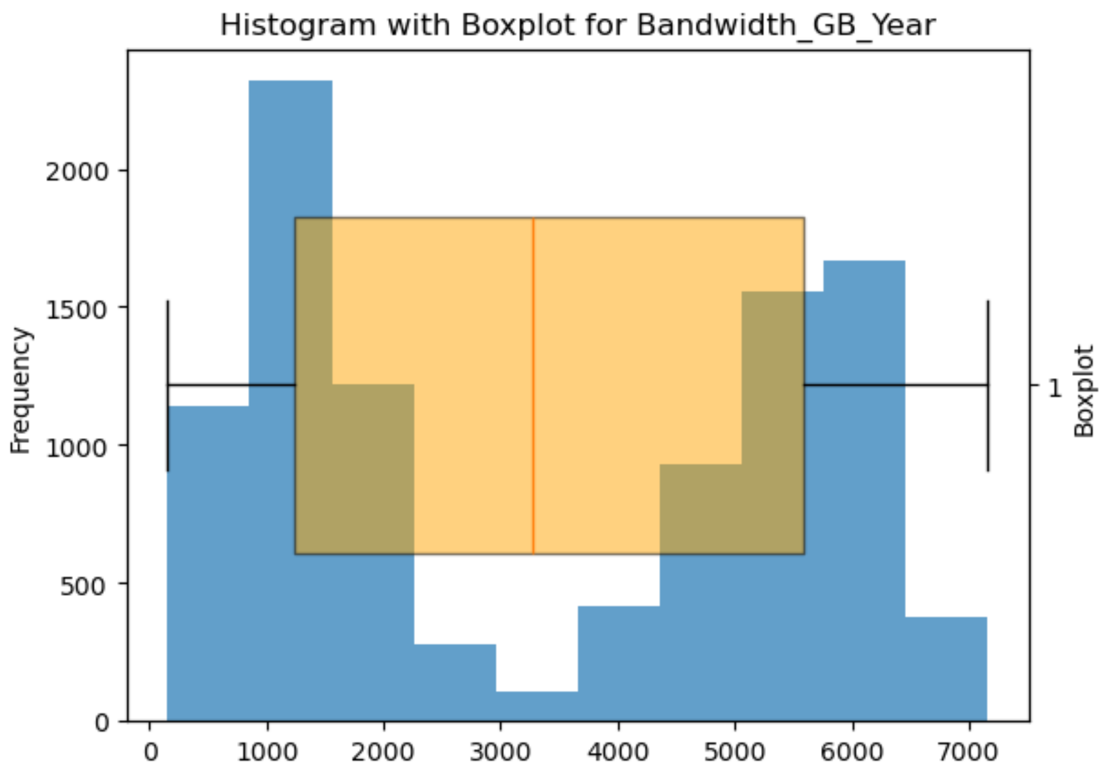

Histogram with Boxplot for Email
Frequency
Boxplot
3500
3000
2500
2000
1500
1000
500
0
0
5
10
15
20
1

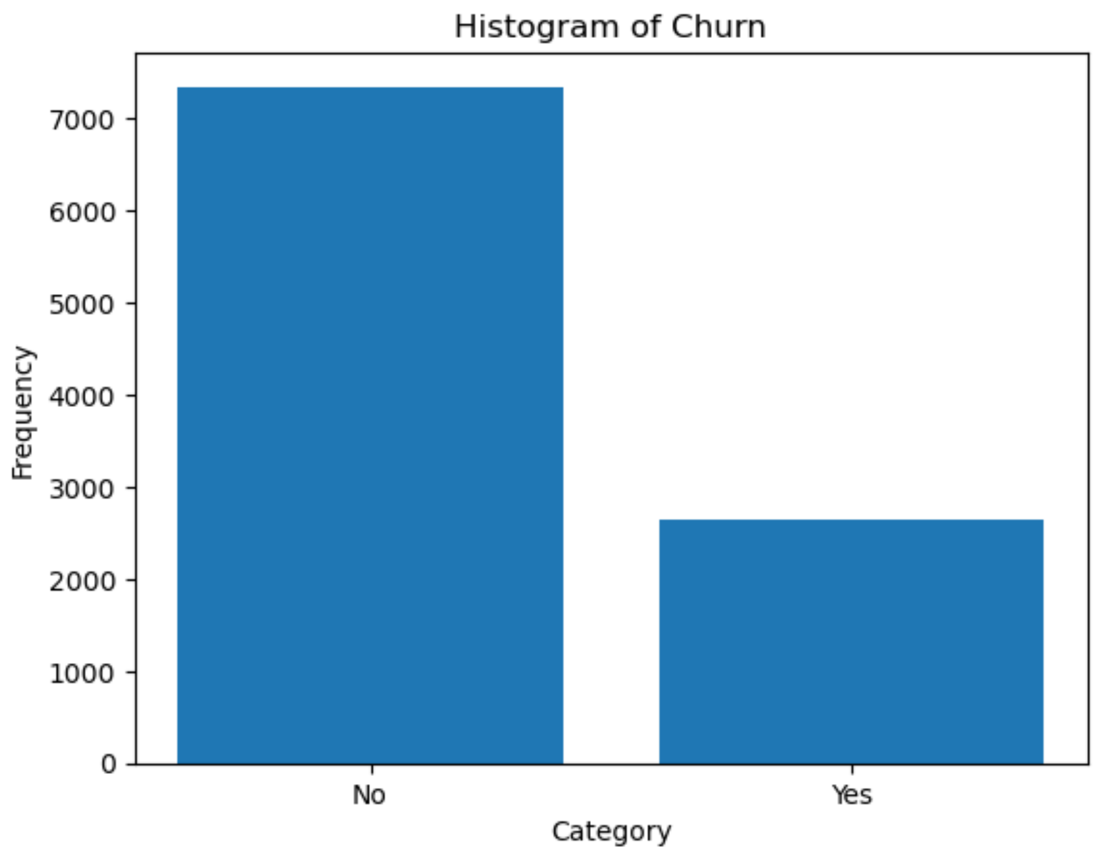

Histogram with Boxplot for Contacts
Frequency
3500
3000
2500
2000
1500
1000
500
0
0
1
2
3
4
5
6
7
1
Boxplot

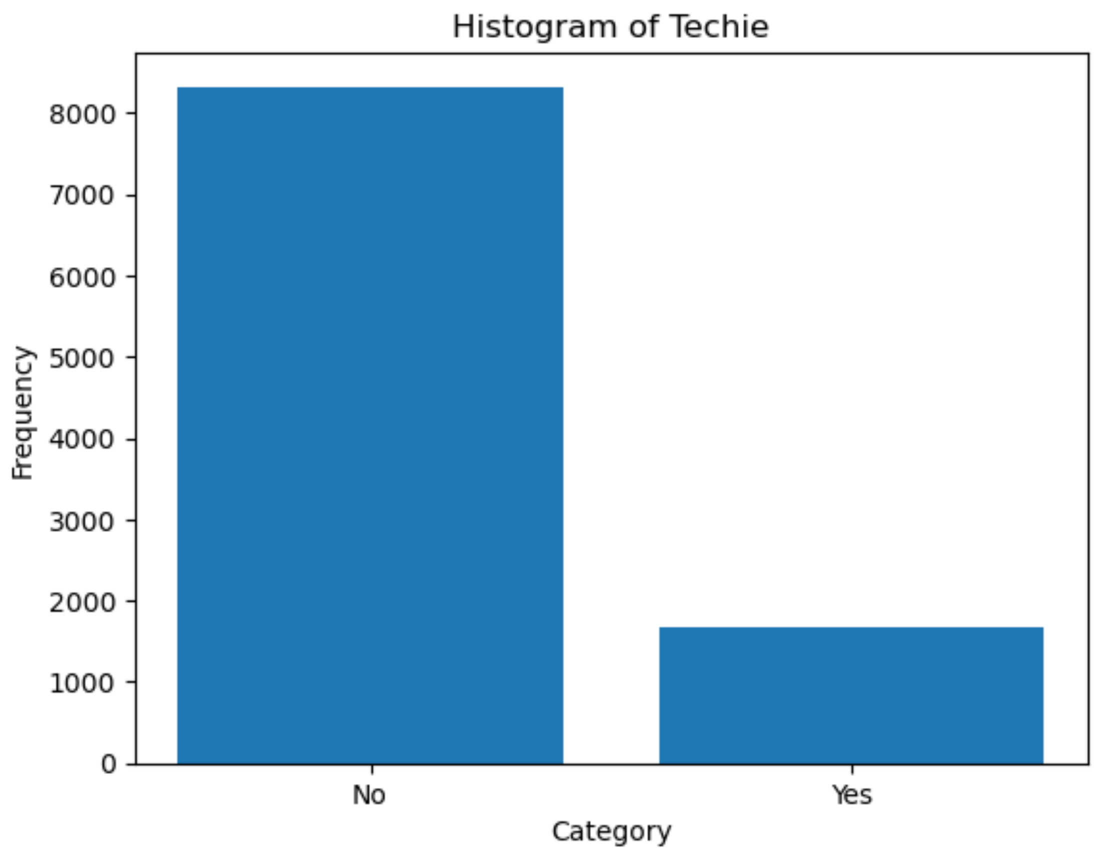

Histogram with Boxplot for Yearly_equip_failure
Frequency
Boxplot
0
1000
2000
3000
4000
5000
6000
7000
0
1
2
3
4
5
6
1

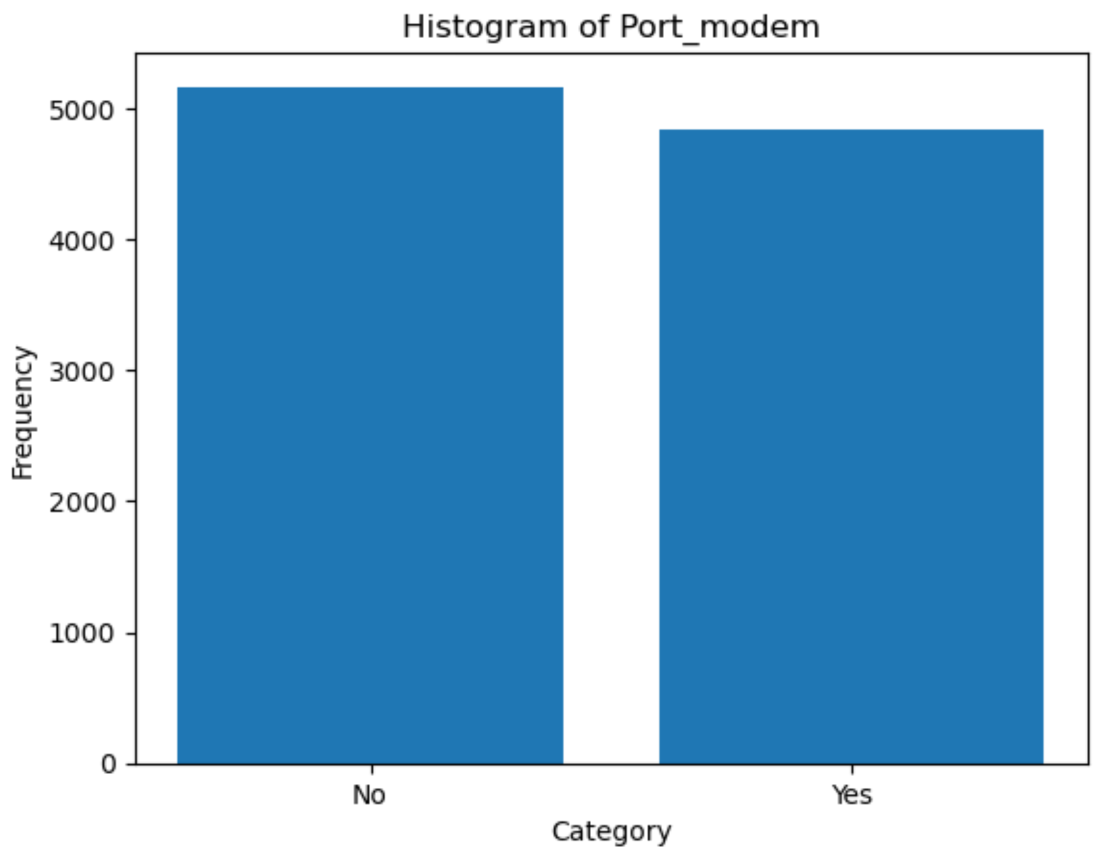

Histogram with Boxplot for Tenure
Frequency
Boxplot
0
500
1000
1500
2000
2500
0
10
20
30
40
50
60
70
1

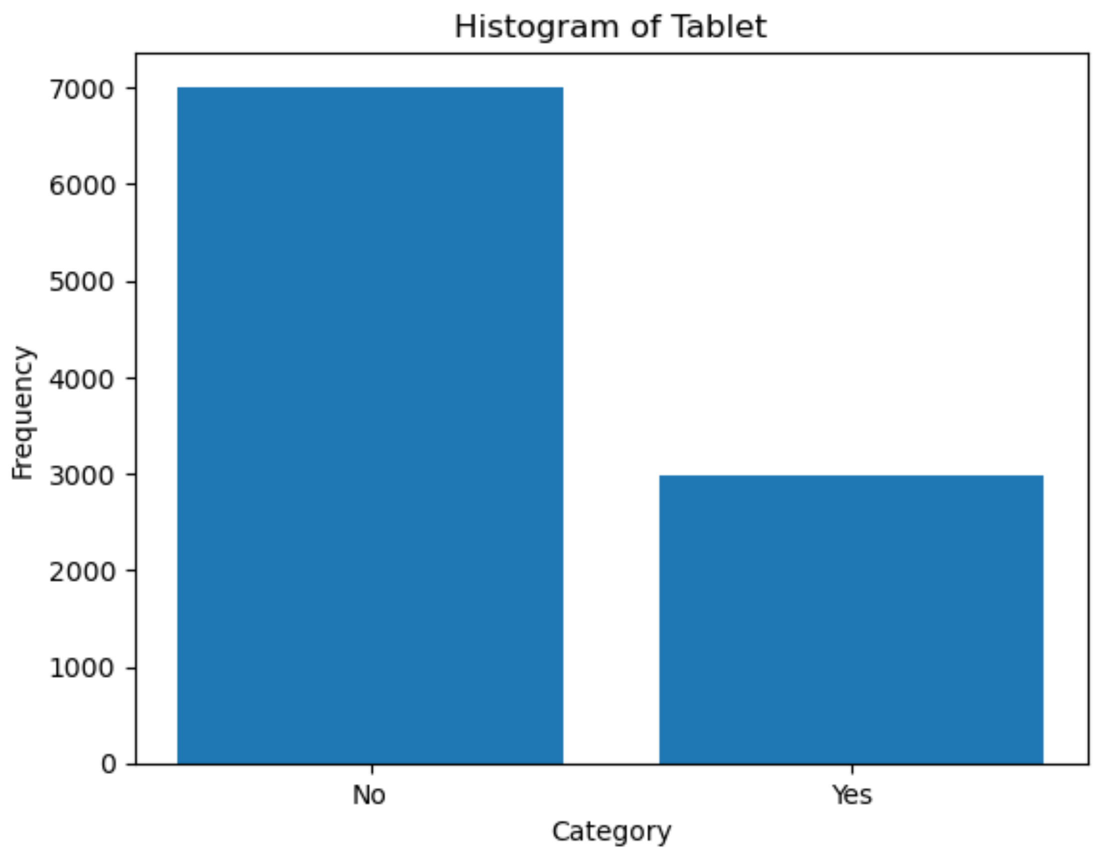

Histogram with Boxplot for MonthlyCharge
Frequency
Boxplot
0
250
500
750
1000
1250
1500
1750
2000
100
150
200
250
300
1

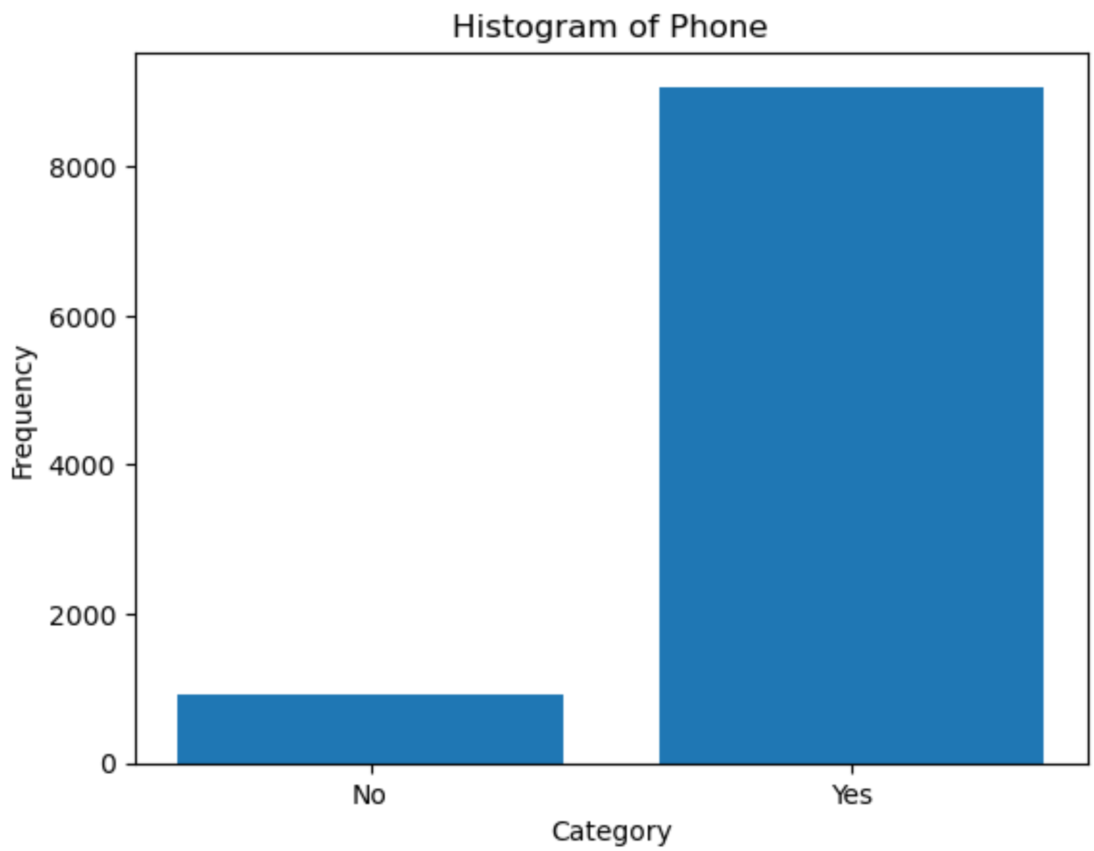

Histogram with Boxplot for Bandwidth_GB_Year
Frequency
Boxplot
0
500
1000
1500
2000
1
0
1000
2000
3000
4000
5000
6000
7000

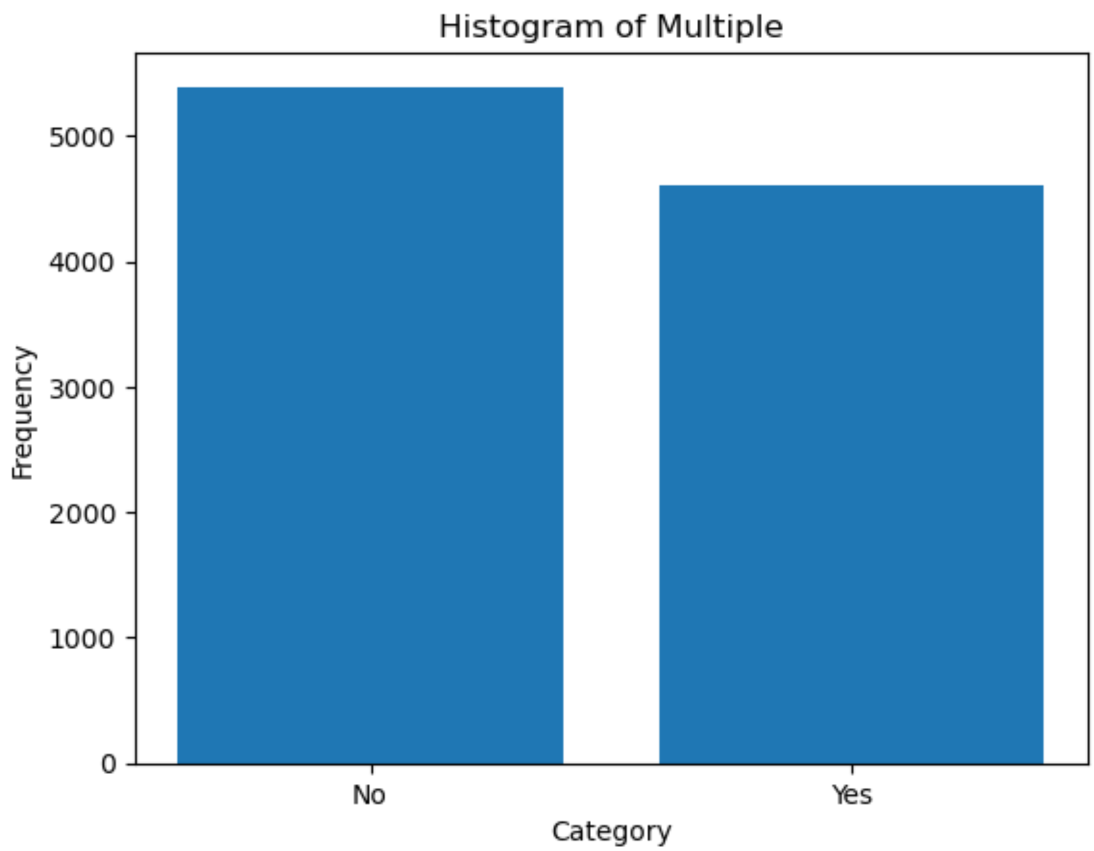

Histogram of Churn
Frequency
7000
6000
5000
4000
3000
2000
1000
0
No
Yes
Category

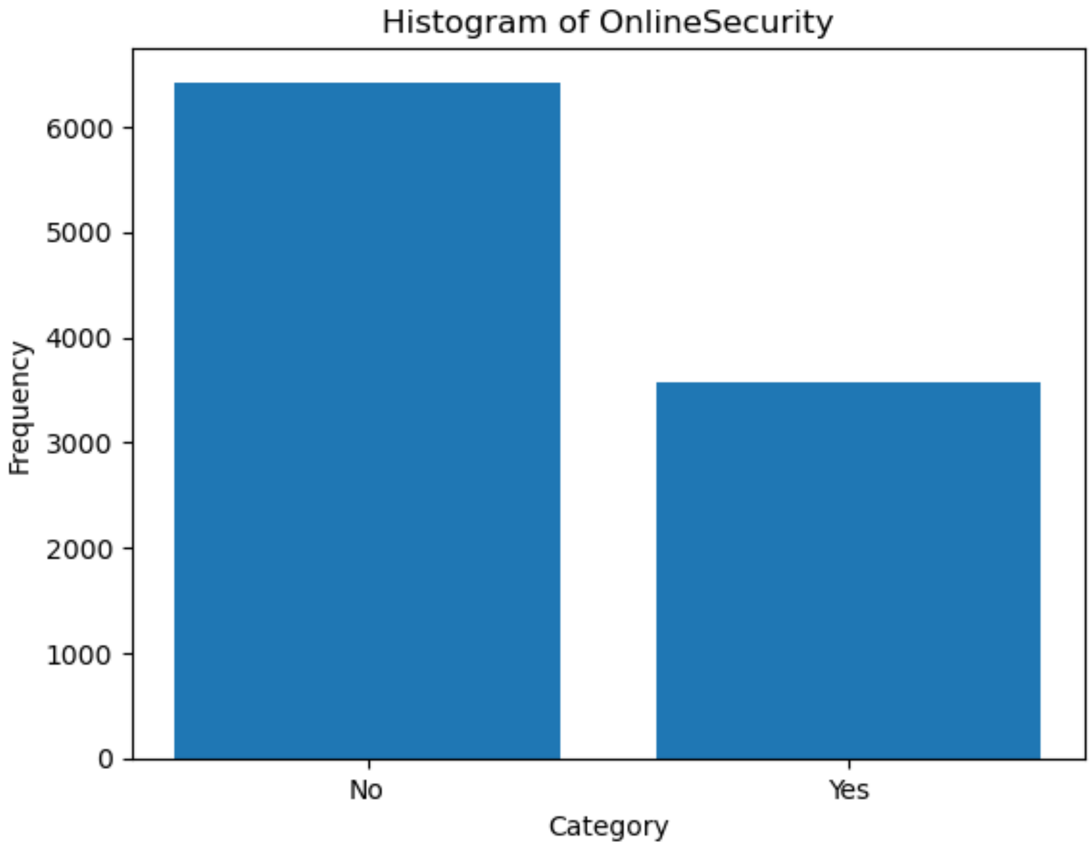

Histogram of Techie
8000
7000
6000
5000
4000
3000
2000
1000
0
Frequency
No
Yes
Category

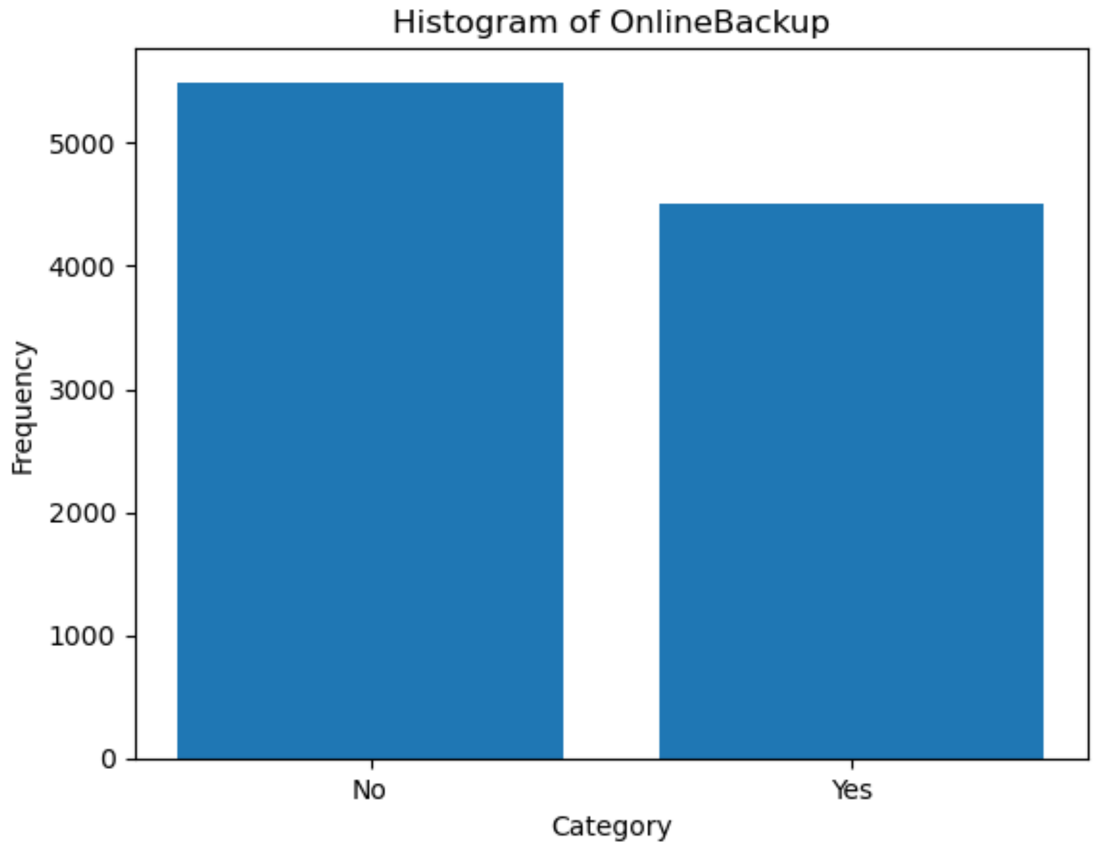

Histogram of Port_modem
5000
4000
3000
2000
1000
0
Frequency
No
Yes
Category

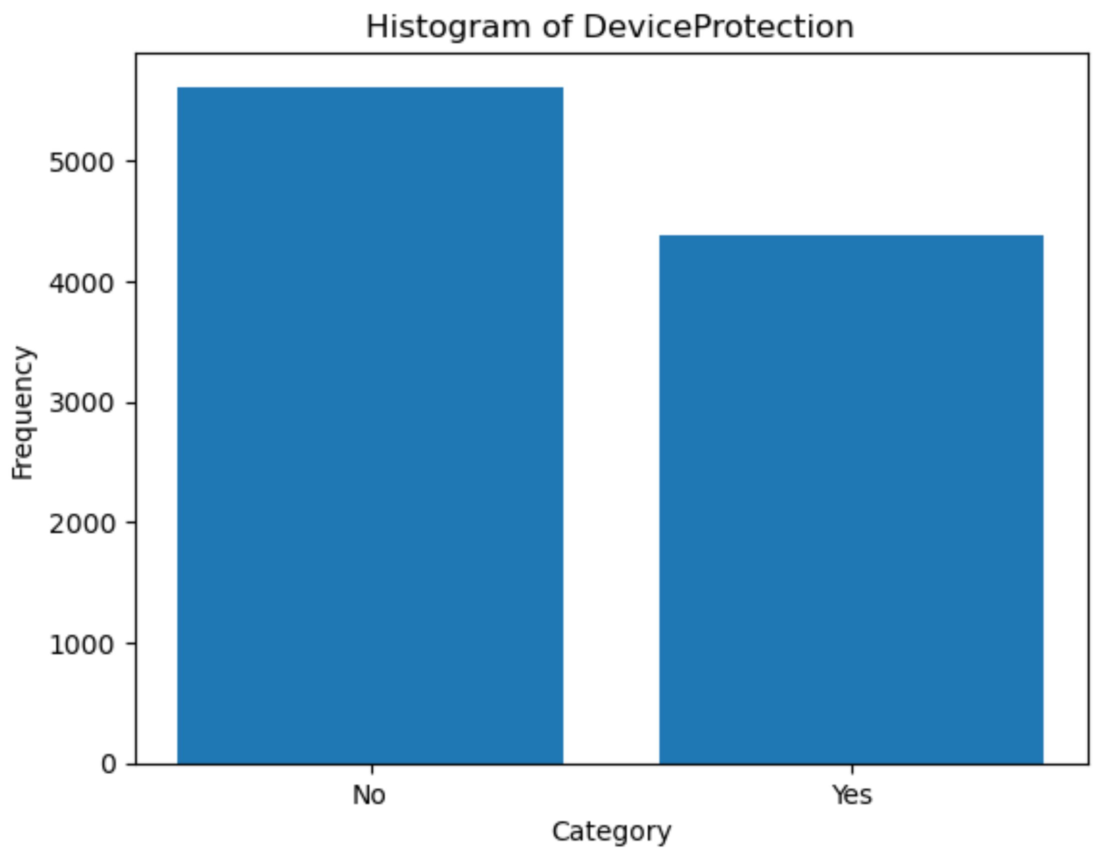

Histogram of Tablet
Frequency
0
1000
2000
3000
4000
5000
6000
7000
No
Yes
Category

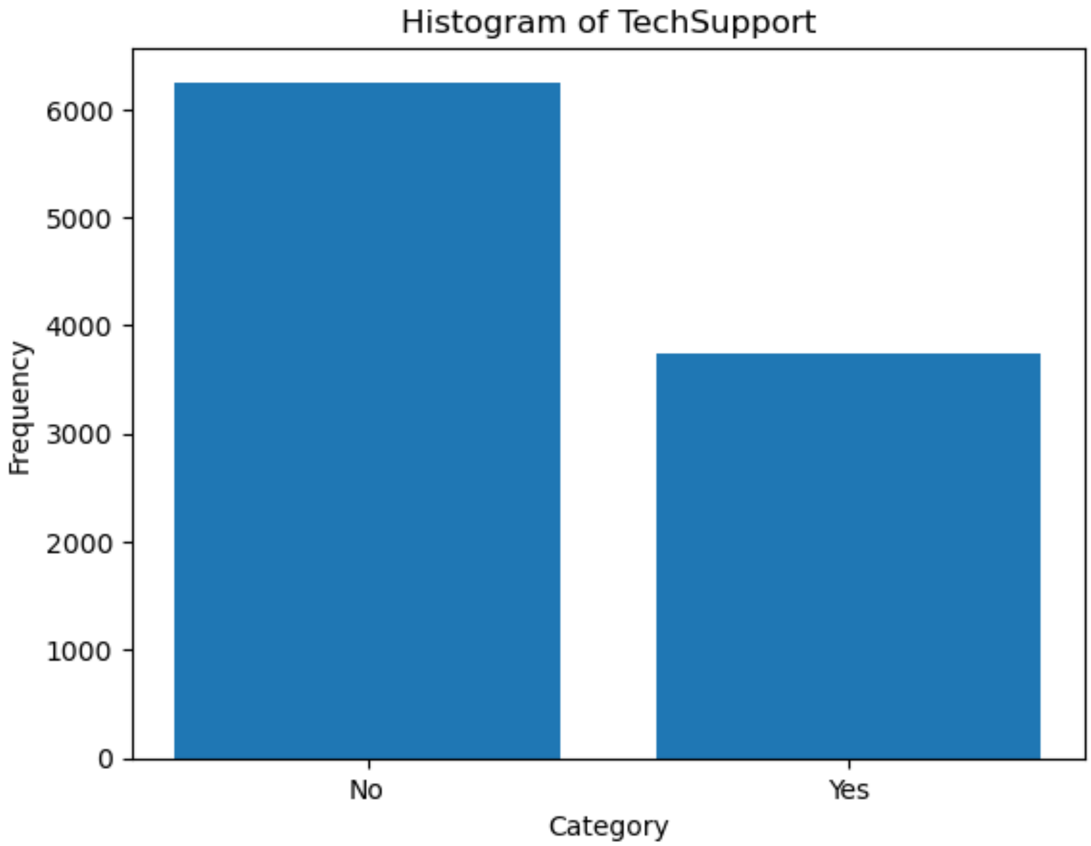

Histogram of Phone
Frequency
8000
6000
4000
2000
0
No
Yes
Category

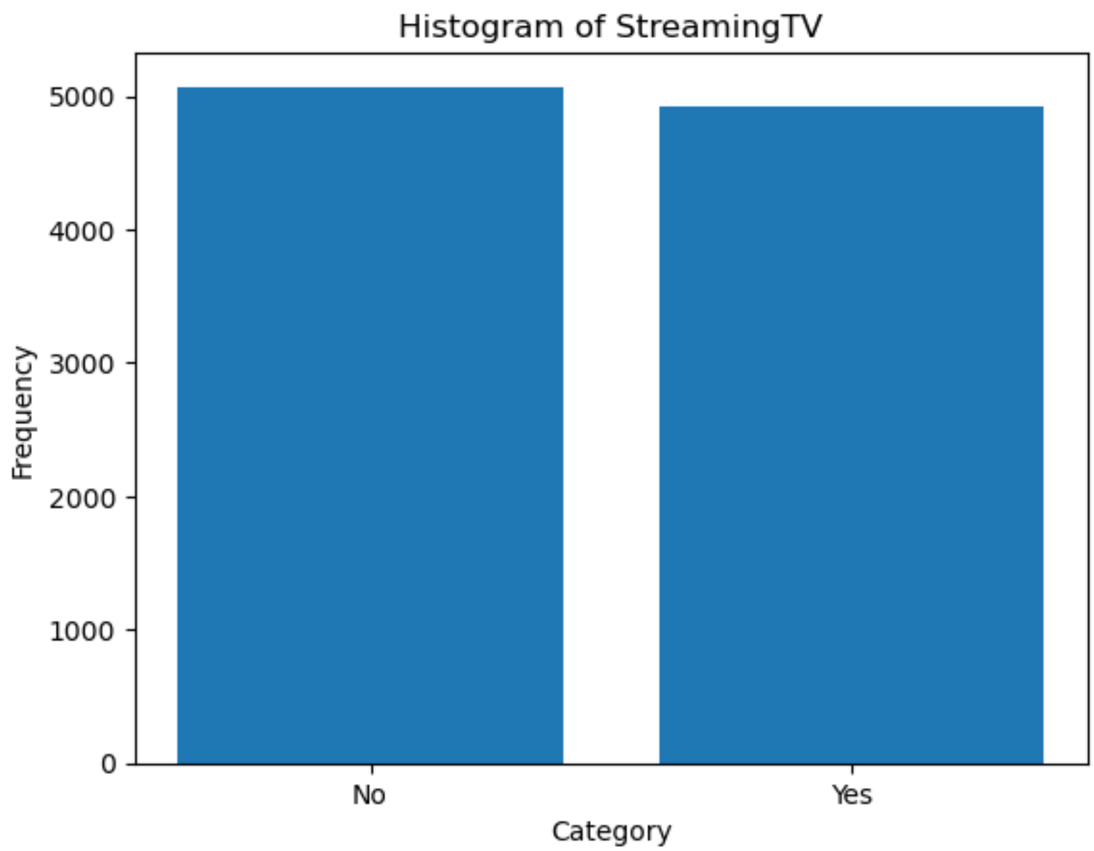

Histogram of Multiple
5000
4000
3000
2000
1000
0
Frequency
No
Yes
Category

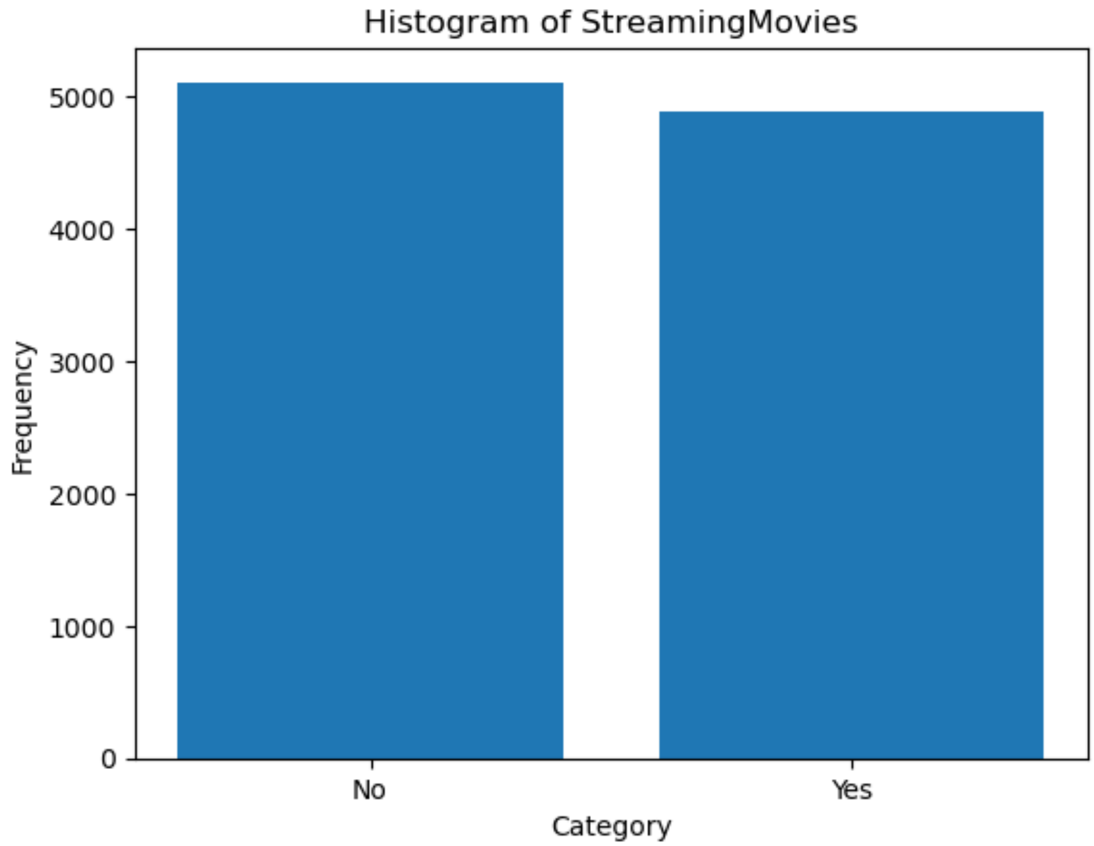

Histogram of OnlineSecurity
Frequency
6000
5000
4000
3000
2000
1000
0
No
Yes
Category

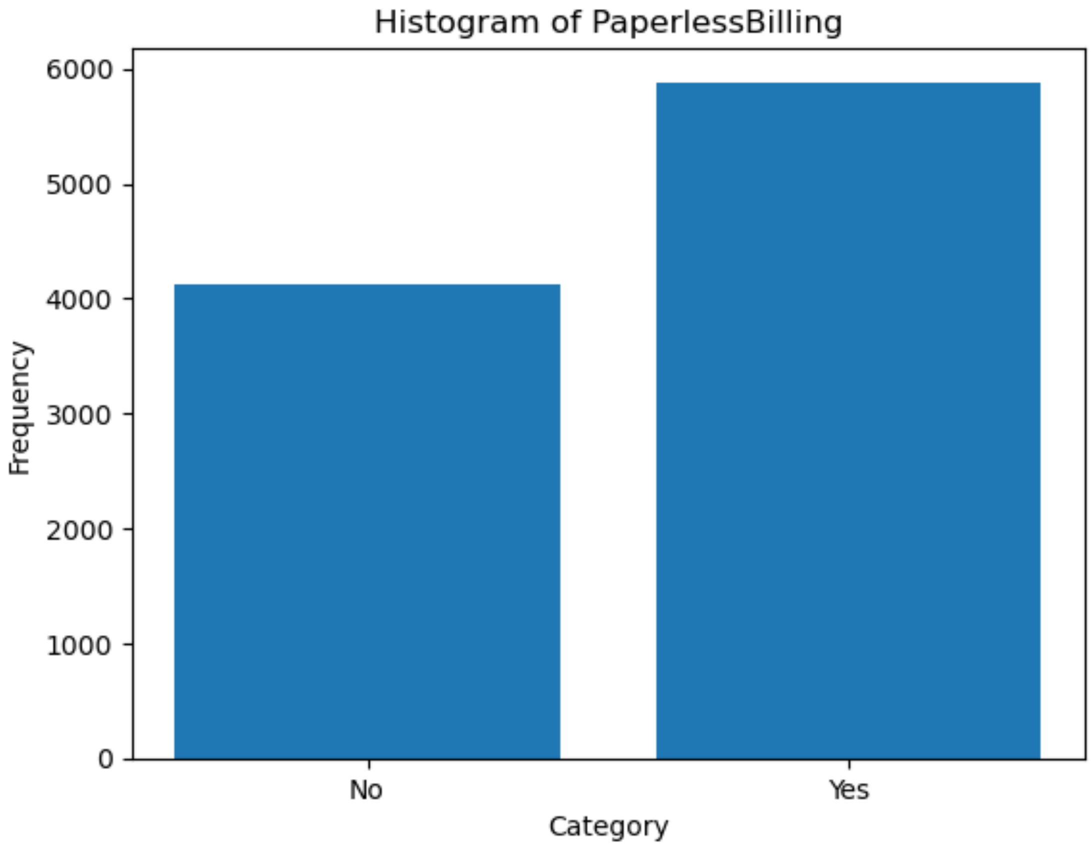

Histogram of OnlineBackup
5000
4000
3000
2000
1000
0
Frequency
No
Yes
Category

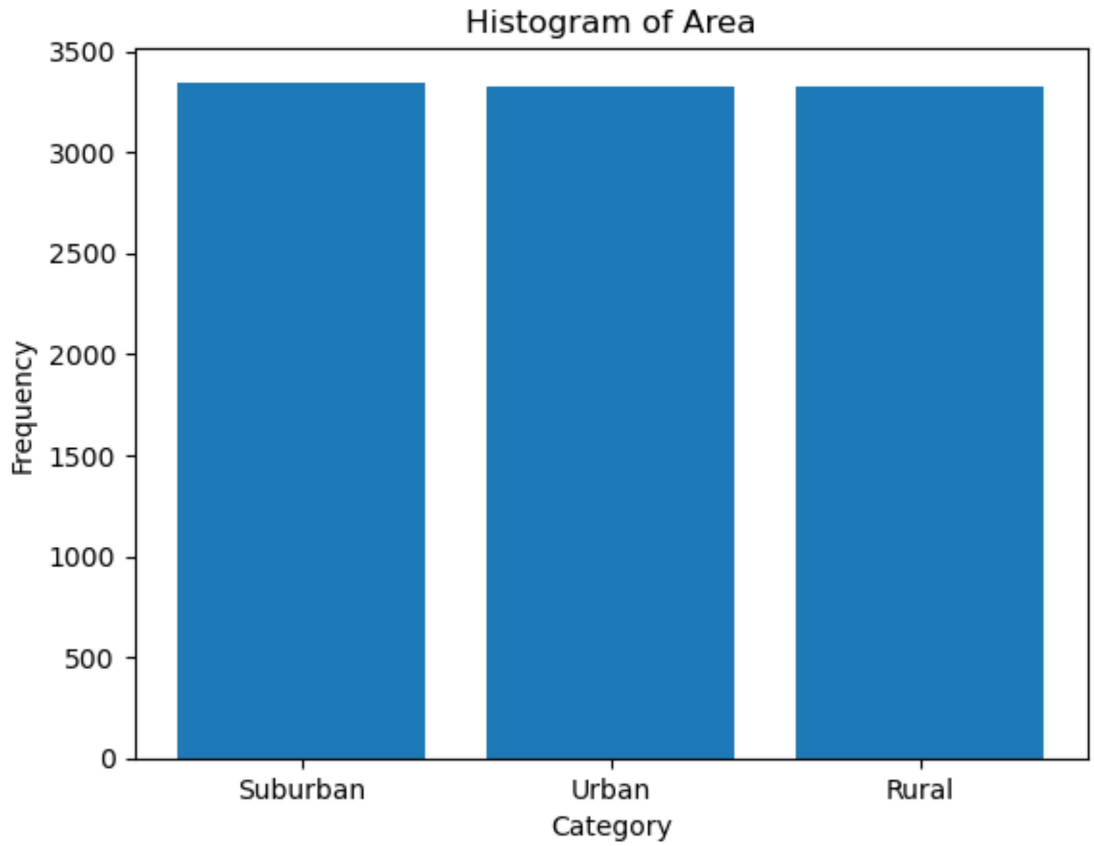

Histogram of DeviceProtection
5000
4000
3000
2000
1000
0
Frequency
No
Yes
Category

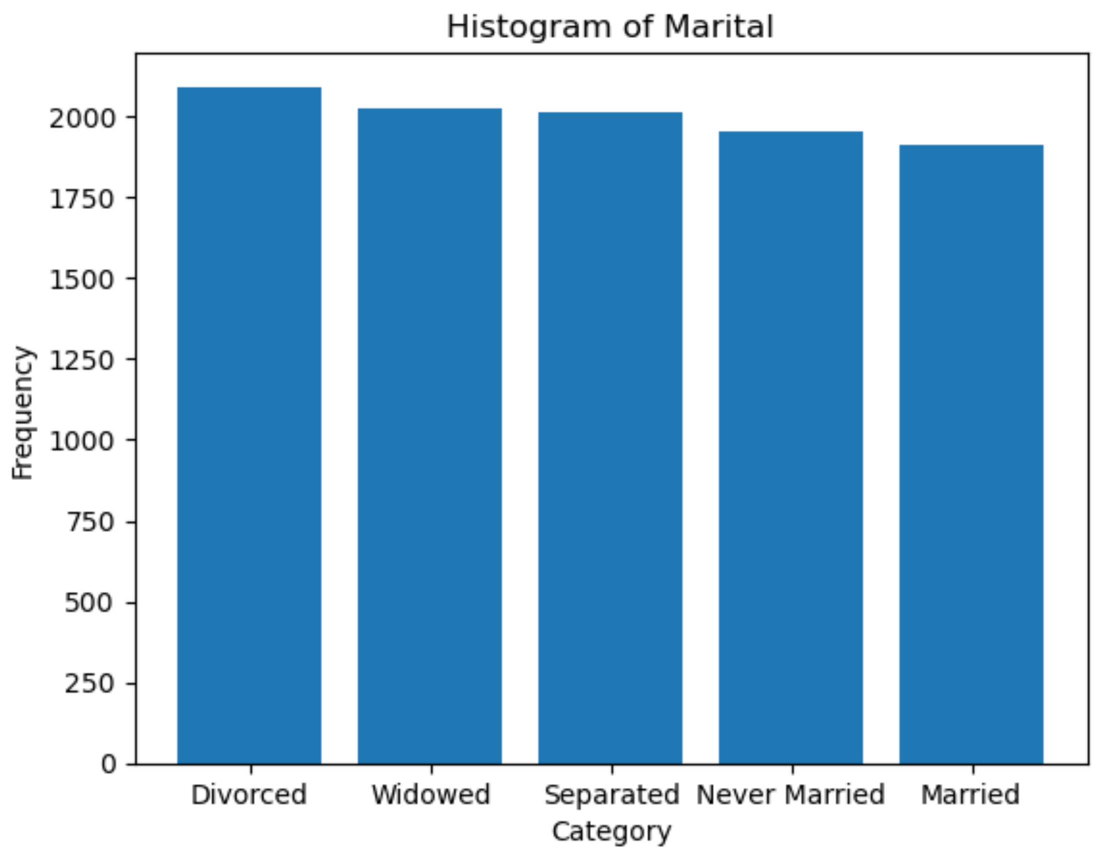

Histogram of TechSupport
6000
5000
4000
3000
2000
1000
0
Frequency
No
Yes
Category

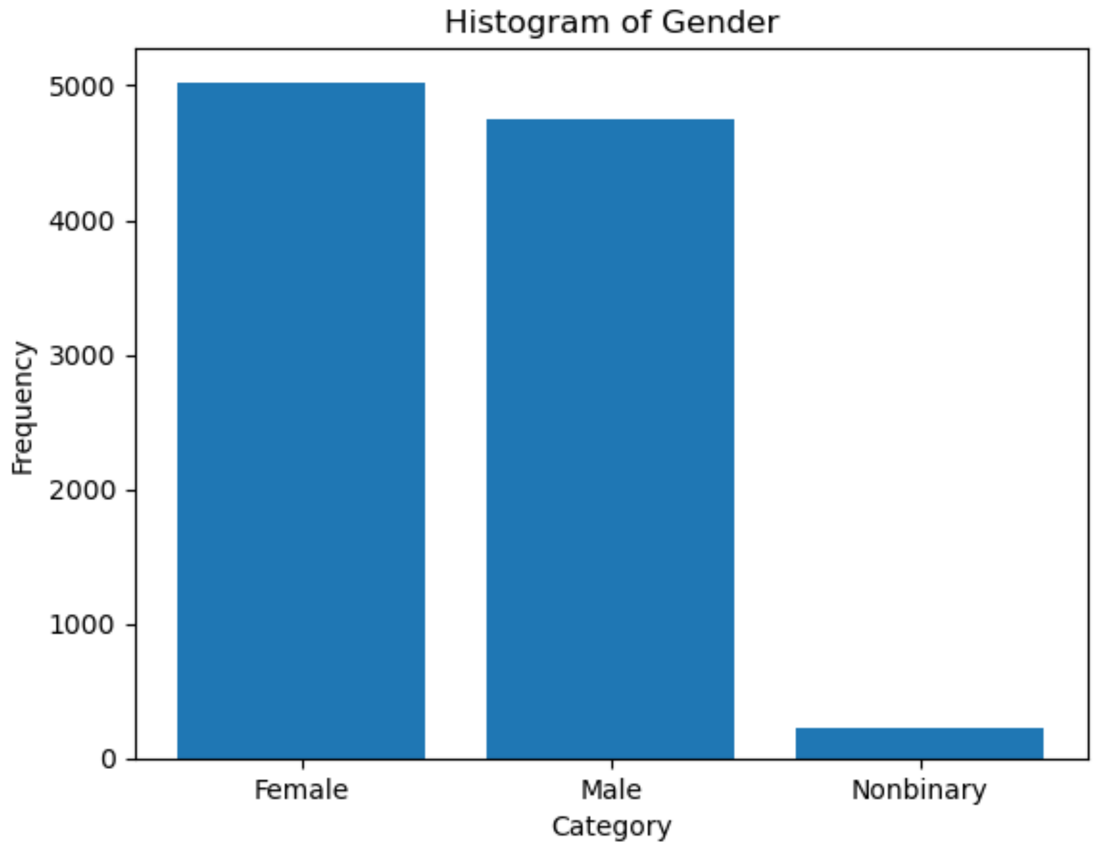

Histogram of StreamingTV
Frequency
Category
No
Yes
0
1000
2000
3000
4000
5000

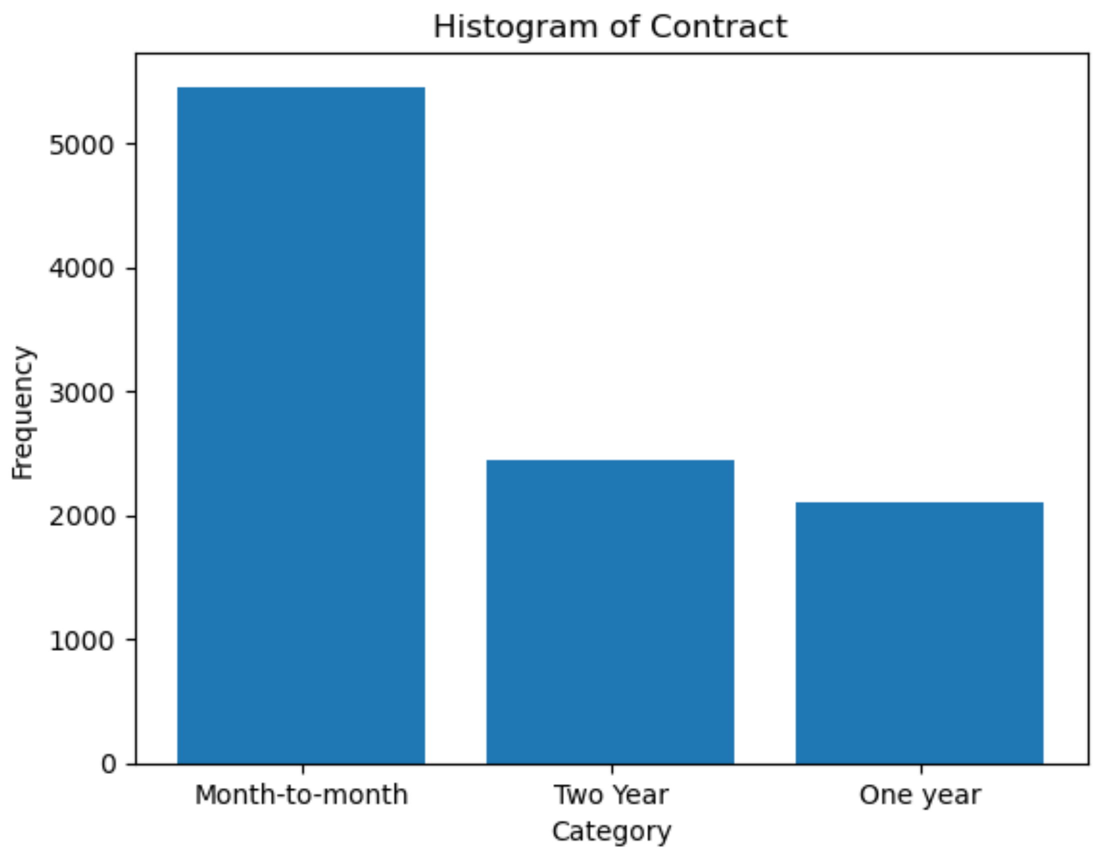

Histogram of StreamingMovies
5000
4000
3000
2000
1000
0
Frequency
No
Yes
Category

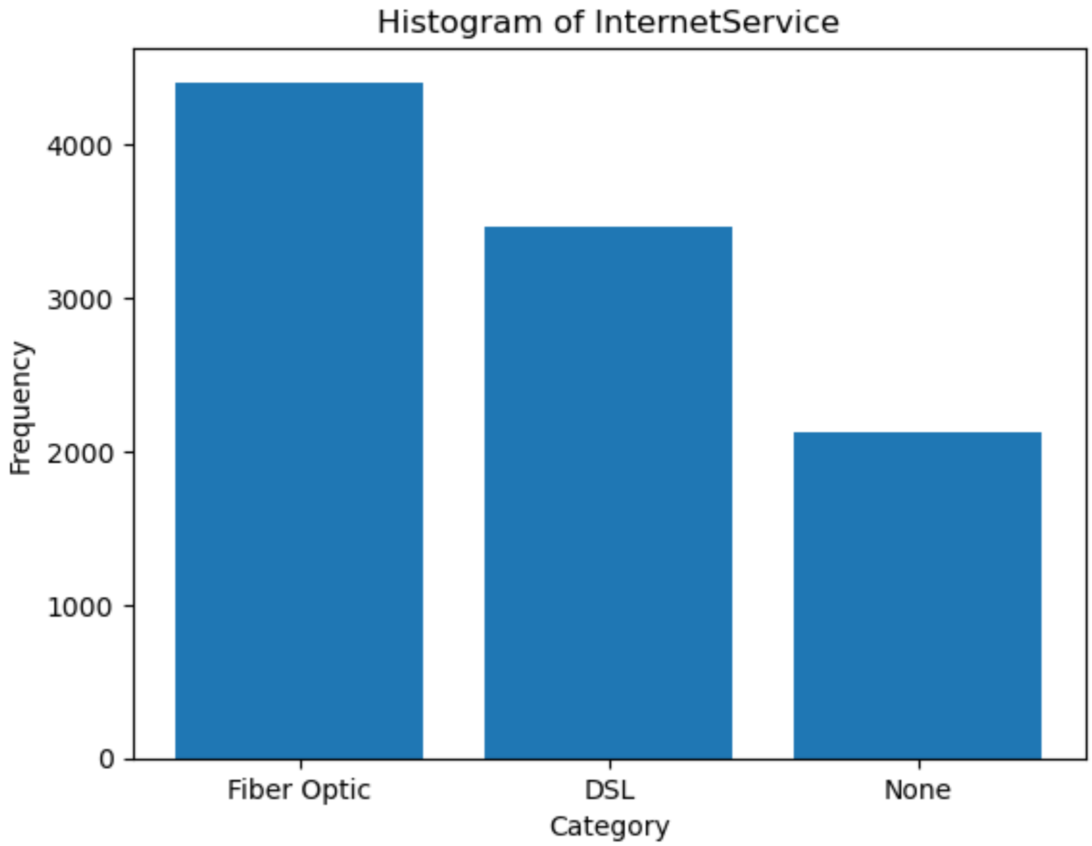

Histogram of PaperlessBilling
6000
5000
4000
3000
2000
1000
0
Frequency
No
Yes
Category


Histogram of Area
3500
3000
2500
2000
1500
1000
500
0
Frequency
Suburban
Urban
Rural
Category

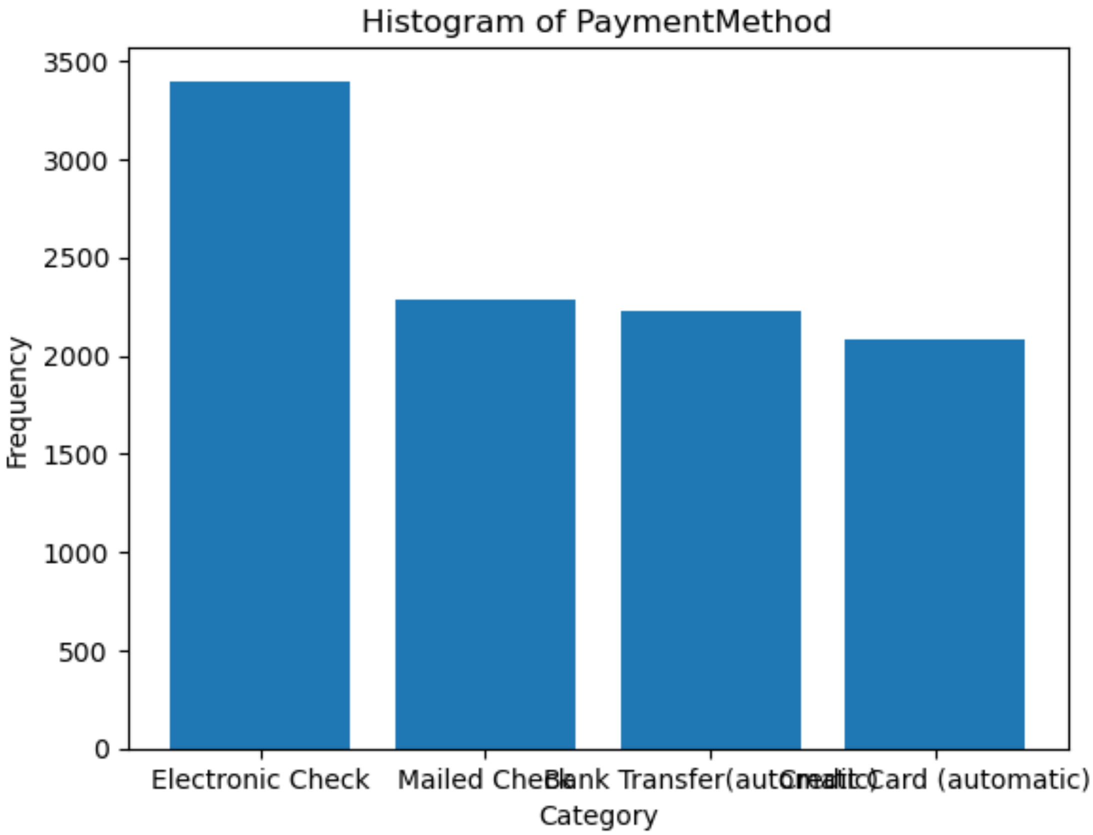

Histogram of Marital
Frequency
0
250
500
750
1000
1250
1500
1750
2000
Divorced
Widowed
Separated
Never Married
Married
Category

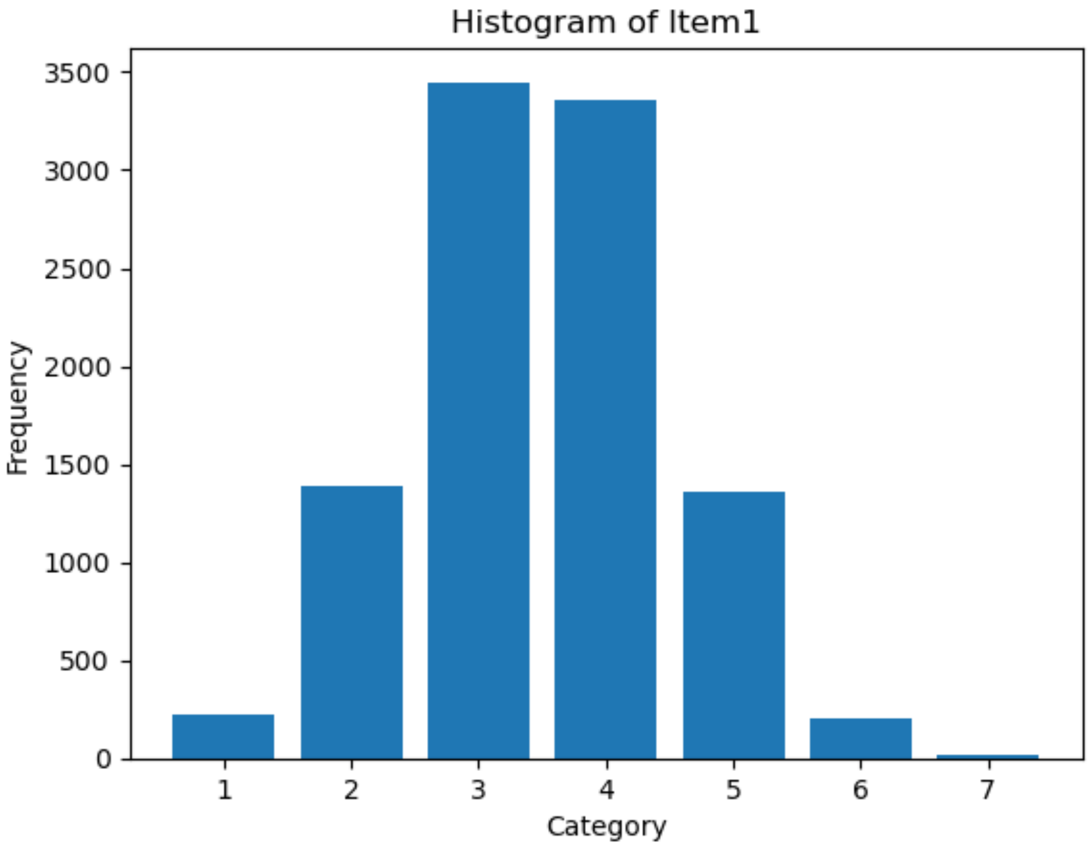

Histogram of Gender
5000
4000
3000
2000
1000
0
Frequency
Female
Male
Nonbinary
Category

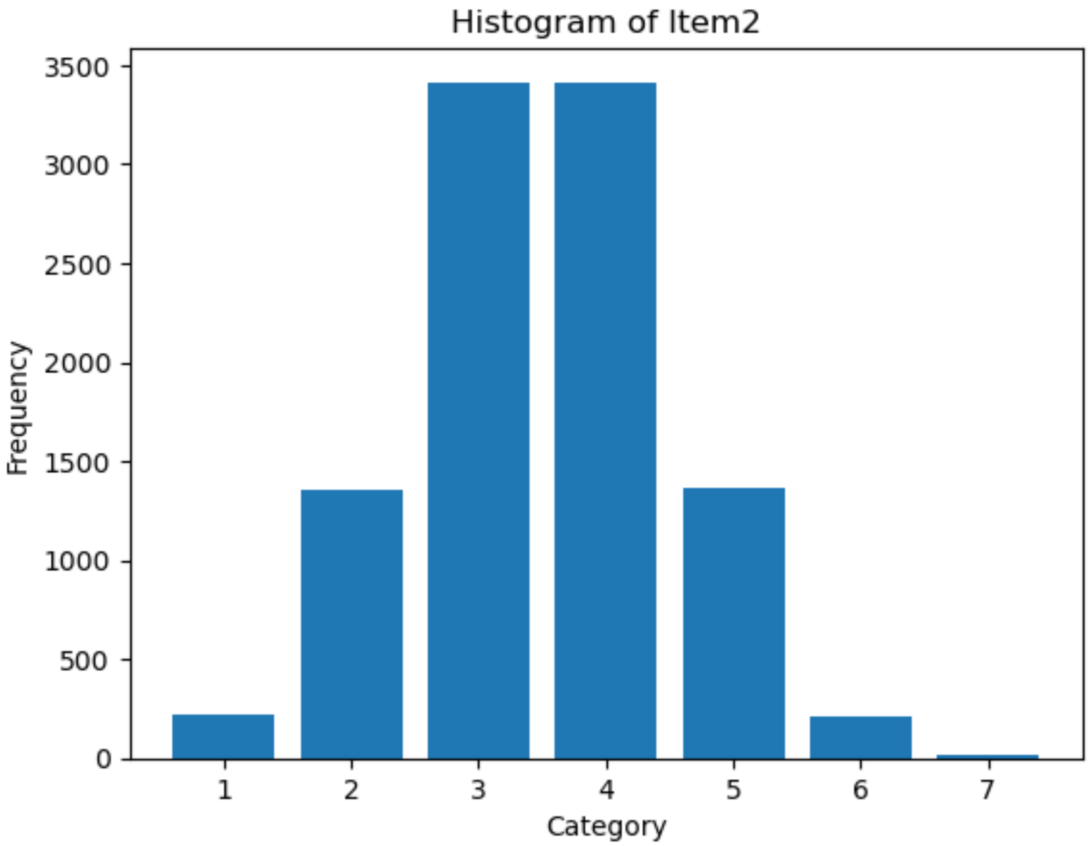

Histogram of Contract
Frequency
5000
4000
3000
2000
1000
0
Month-to-month
Two Year
One year
Category



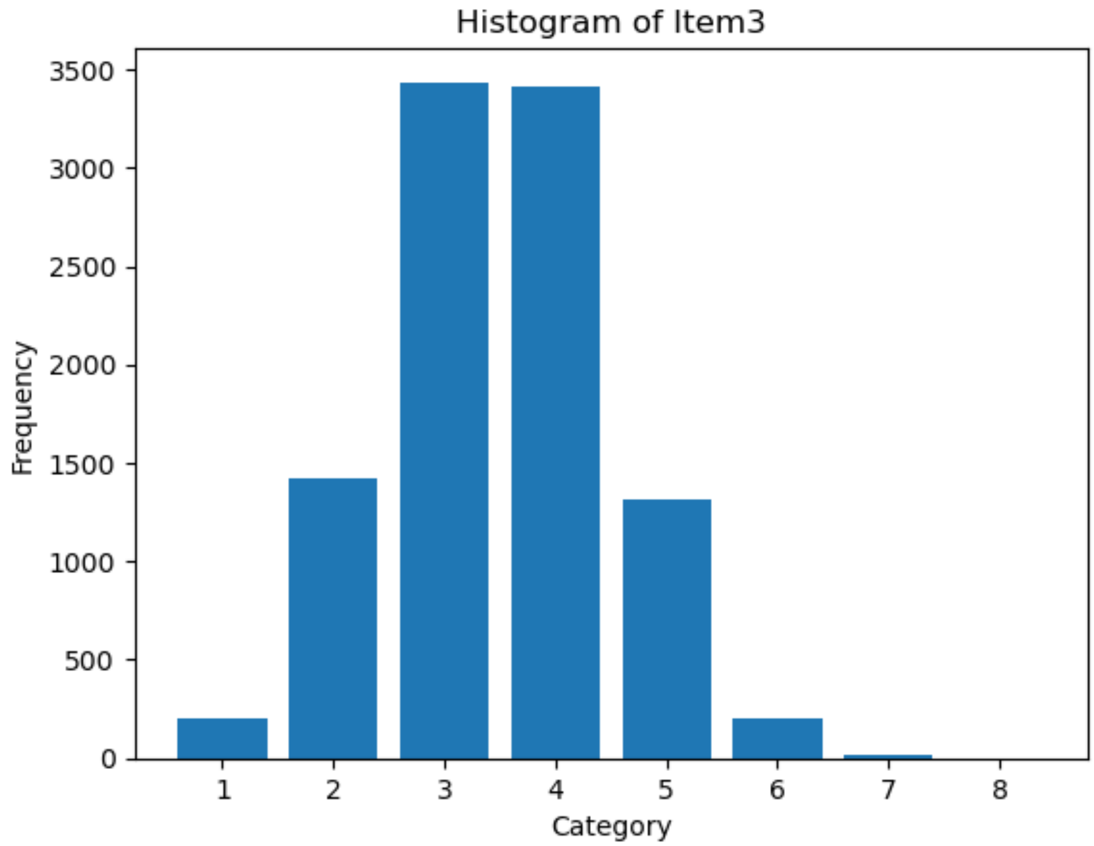
Histogram of InternetService
Frequency
4000
3000
2000
1000
0
Fiber Optic
DSL
None
Category

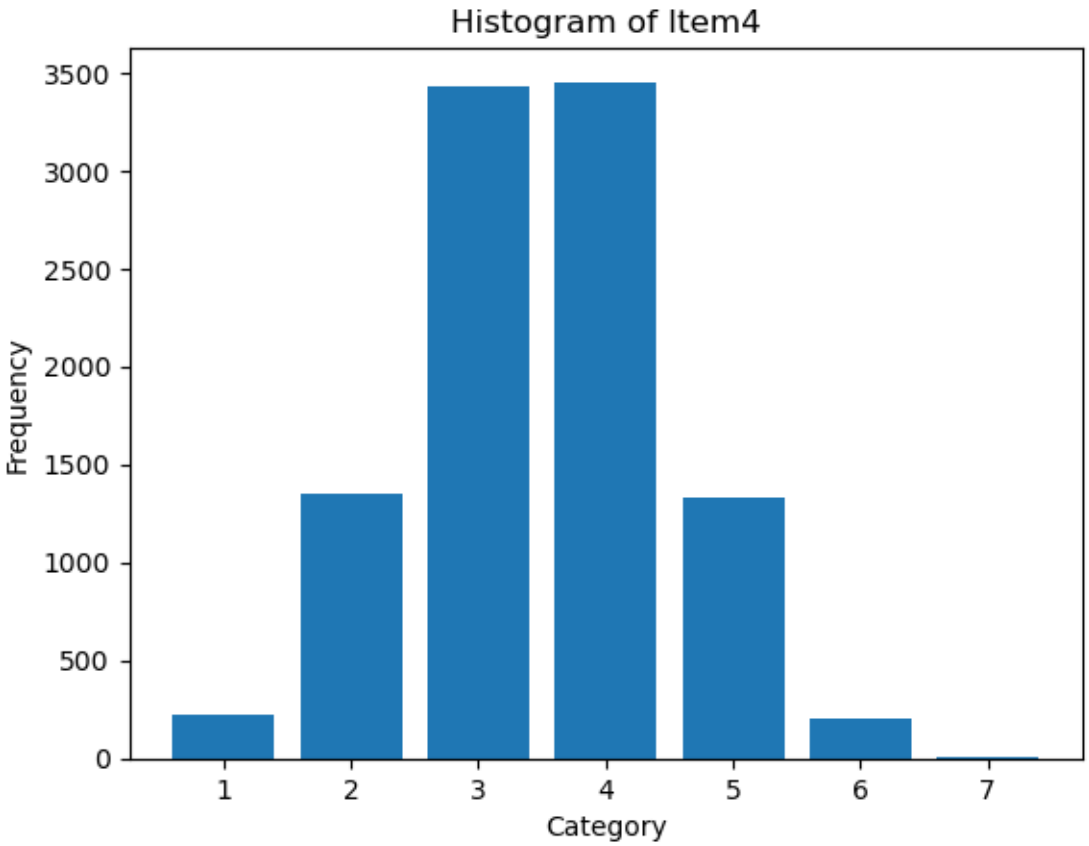

Histogram of PaymentMethod
Frequency
3500
3000
2500
2000
1500
1000
500
0
Electronic Check
Mailed Check
Bank Transfer(automatic)
Credit Card (automatic)
Category

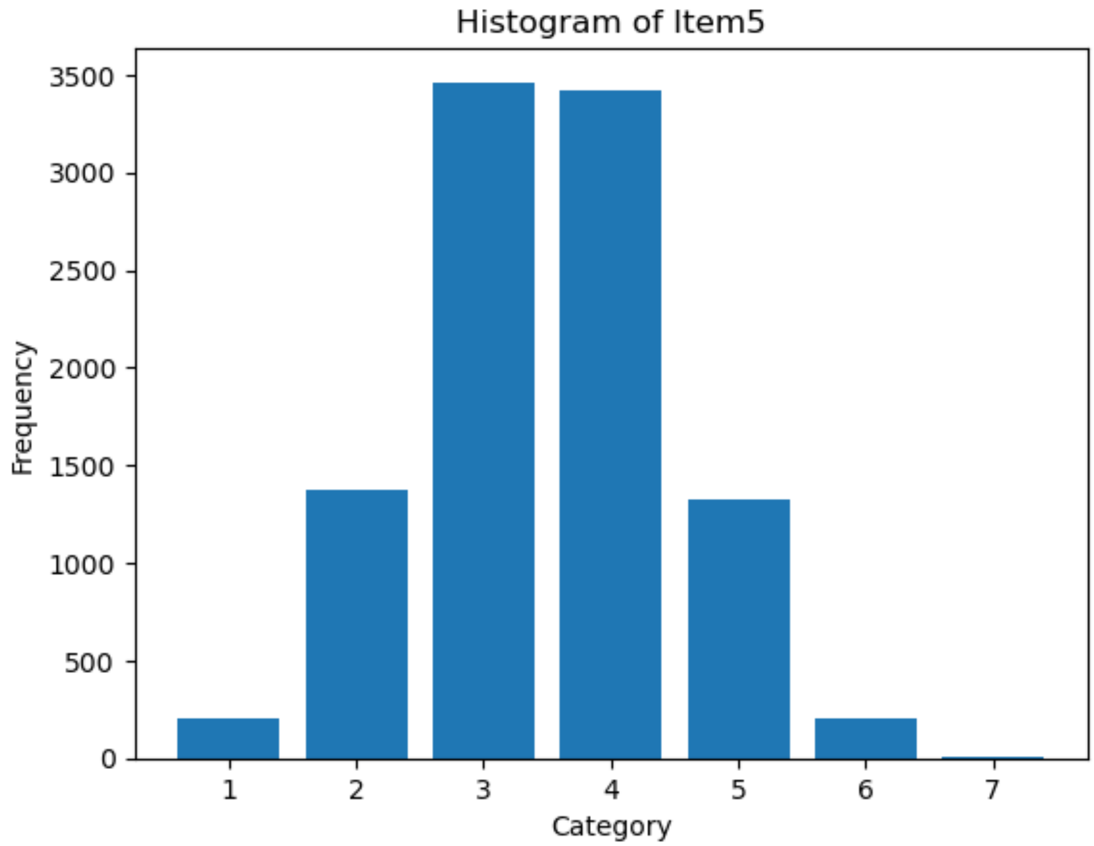

Histogram of Item1
3500
3000
2500
2000
1500
1000
500
0
Frequency
1
2
3
4
5
6
7
Category

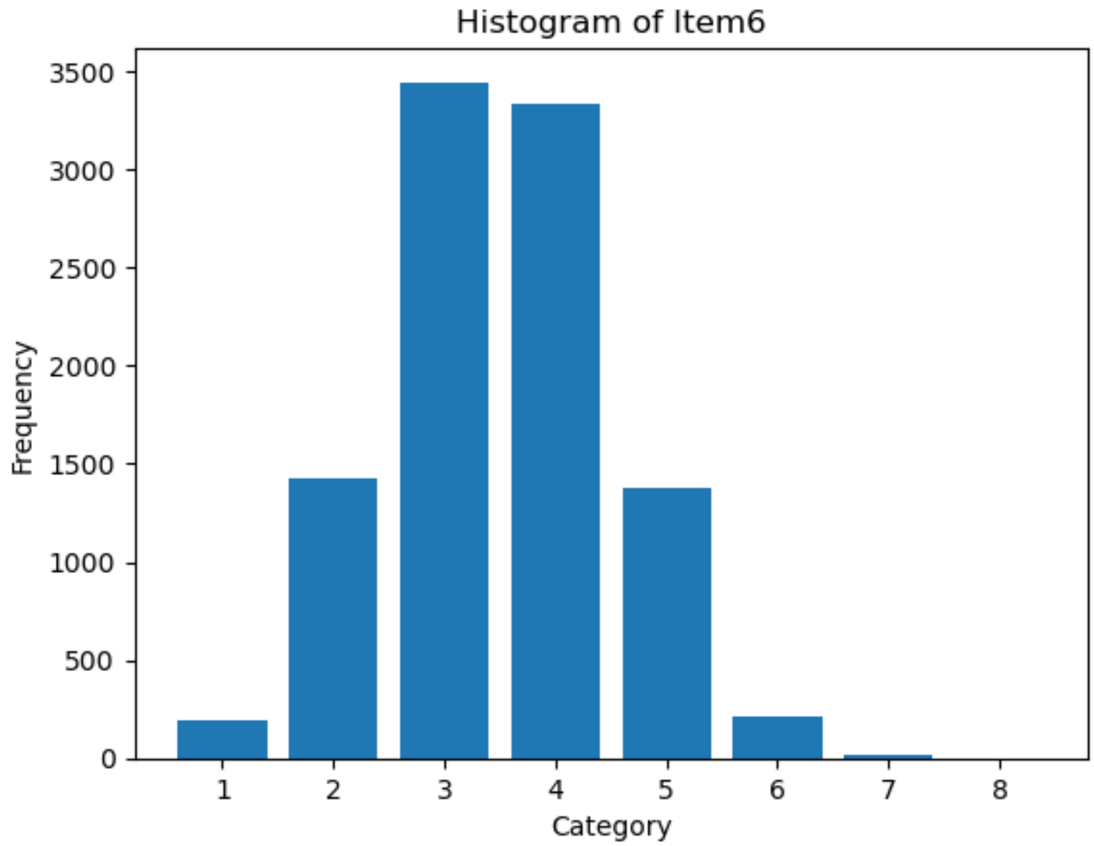


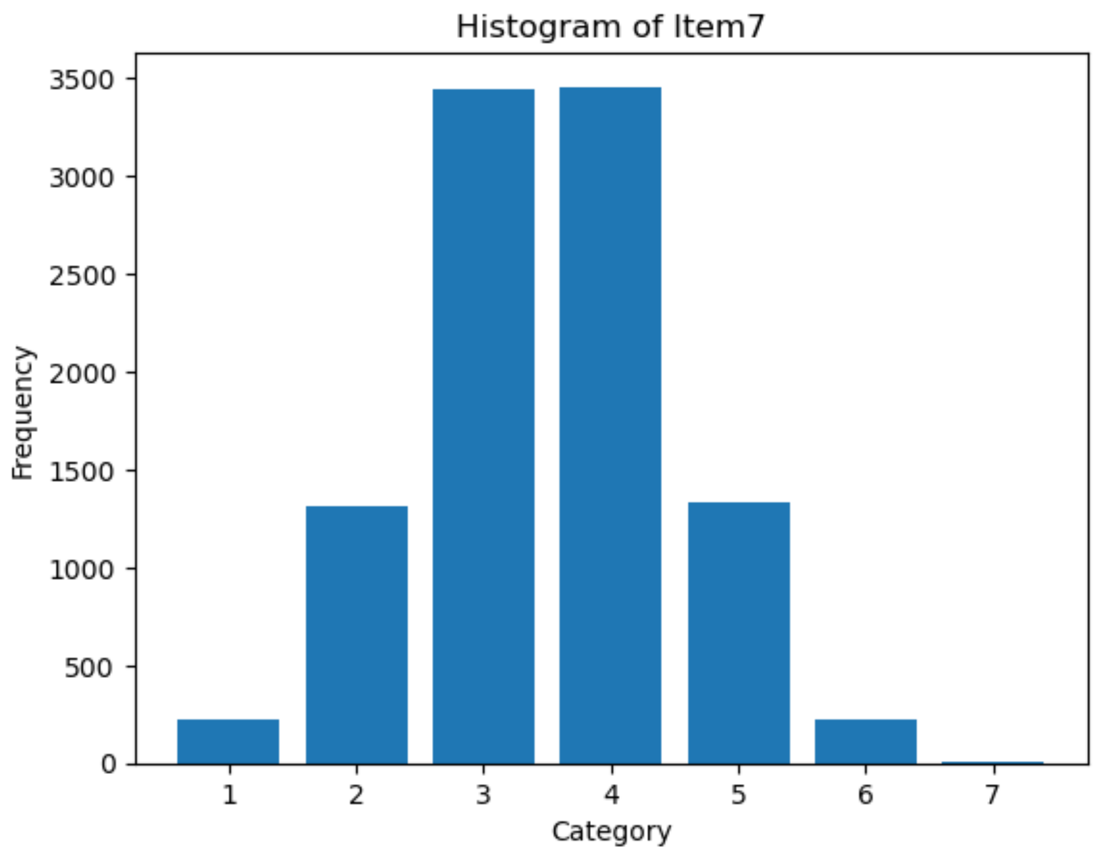
Histogram of Item2
Frequency
Category
0
500
1000
1500
2000
2500
3000
3500
1
2
3
4
5
6
7

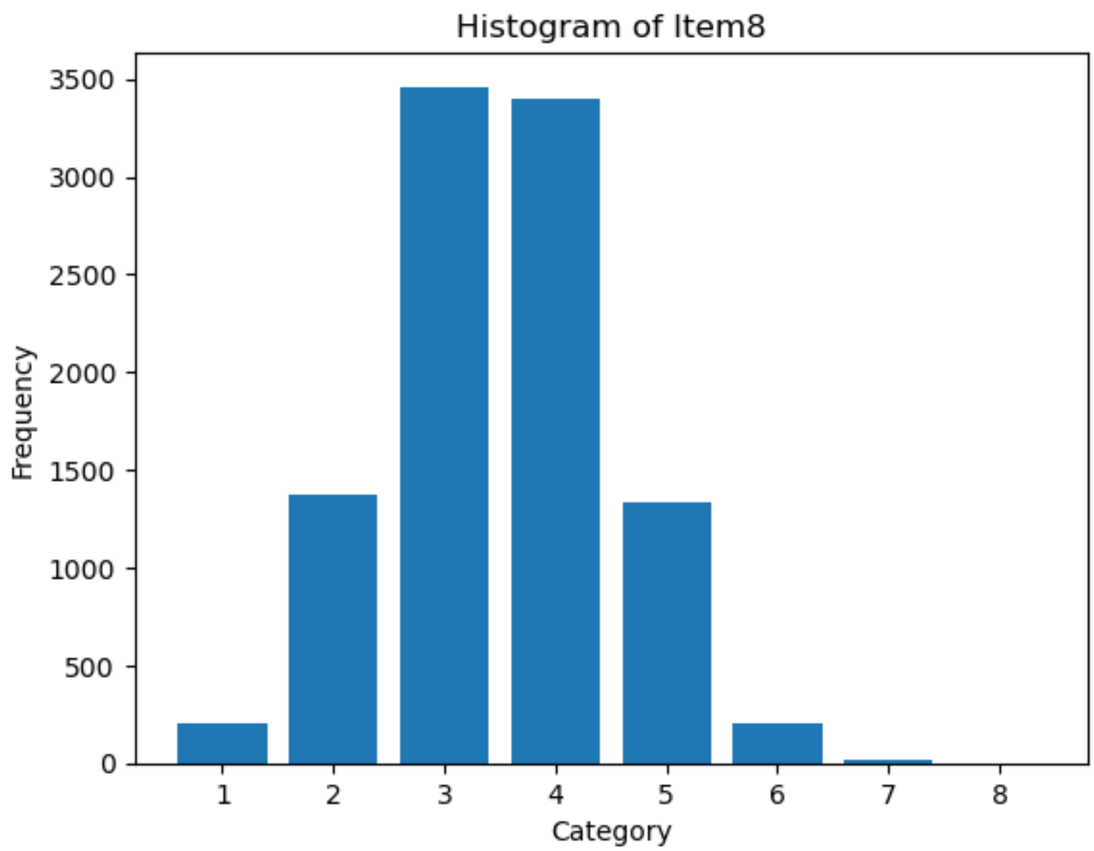

Histogram of Item3
3500
3000
2500
2000
1500
1000
500
0
Frequency
1
2
3
4
5
6
7
8
Category


Histogram of Item4
Frequency
Category
0
500
1000
1500
2000
2500
3000
3500
1
2
3
4
5
6
7


Histogram of Item5
Frequency
Category
0
500
1000
1500
2000
2500
3000
3500
1
2
3
4
5
6
7


Histogram of Item6
Frequency
3500
3000
2500
2000
1500
1000
500
0
1
2
3
4
5
6
7
8
Category

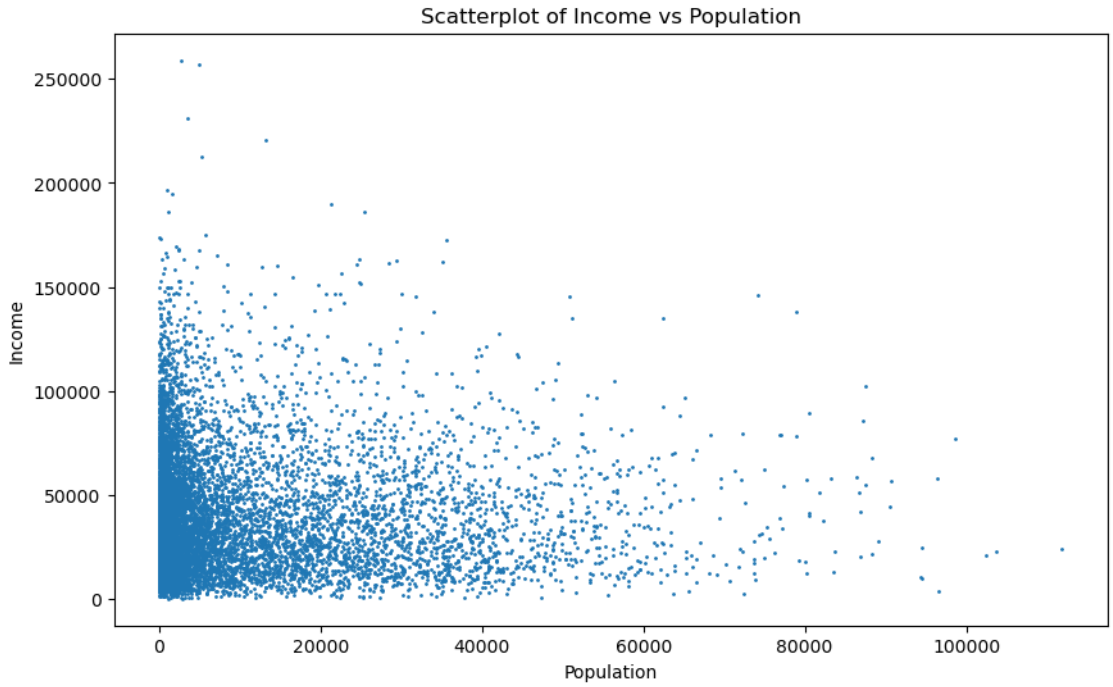

Histogram of Item7
Frequency
Category
0
500
1000
1500
2000
2500
3000
3500
1
2
3
4
5
6
7

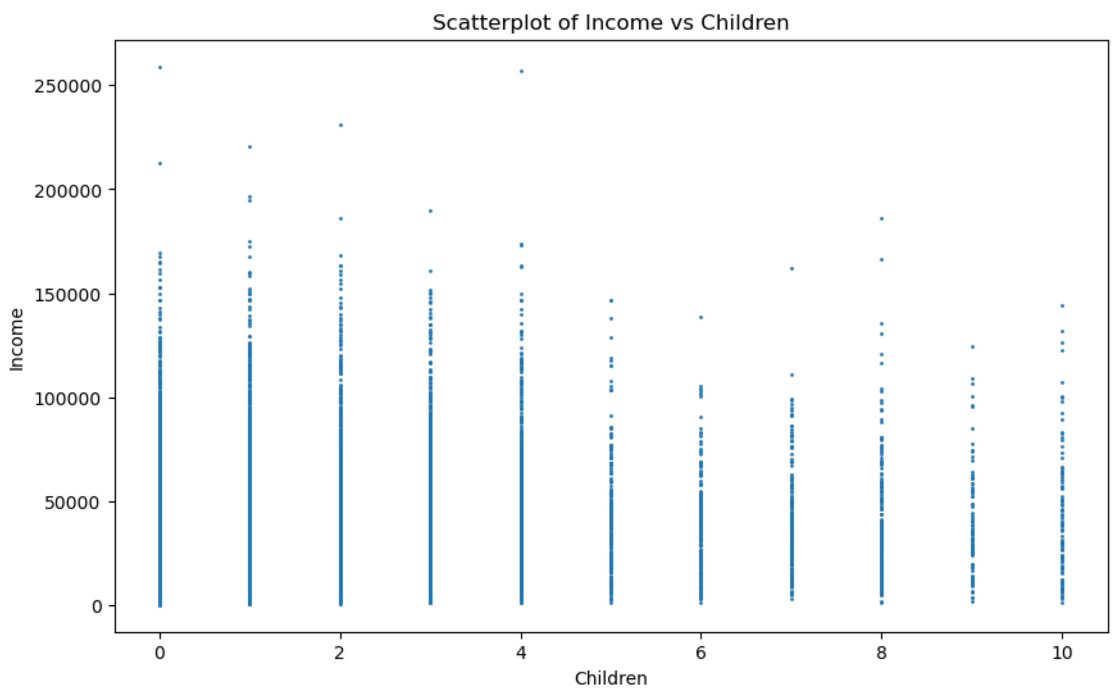


[16]:
```
## C3 Bivariate Visualizations

def plot_bivarcont(data_frame, col1, col2):
# Bivariate scatterplot of col2 vs col1

    plt.figure(figsize = (10,6))
    plt.scatter(data_frame[col1], data_frame[col2], s = 1)

    plt.xlabel(col1)
    plt.ylabel(col2)
    plt.title(f'Scatterplot of {col2} vs {col1}')
    plt.show()


def plot_bivarcat(data_frame, cont_col, cat_col):
# Bivariate boxplot of continuous vs categorical

    data_frame.boxplot(column = cont_col, by = cat_col, grid = False)

    plt.title(f'Box Plot of {cont_col} by {cat_col}')
```

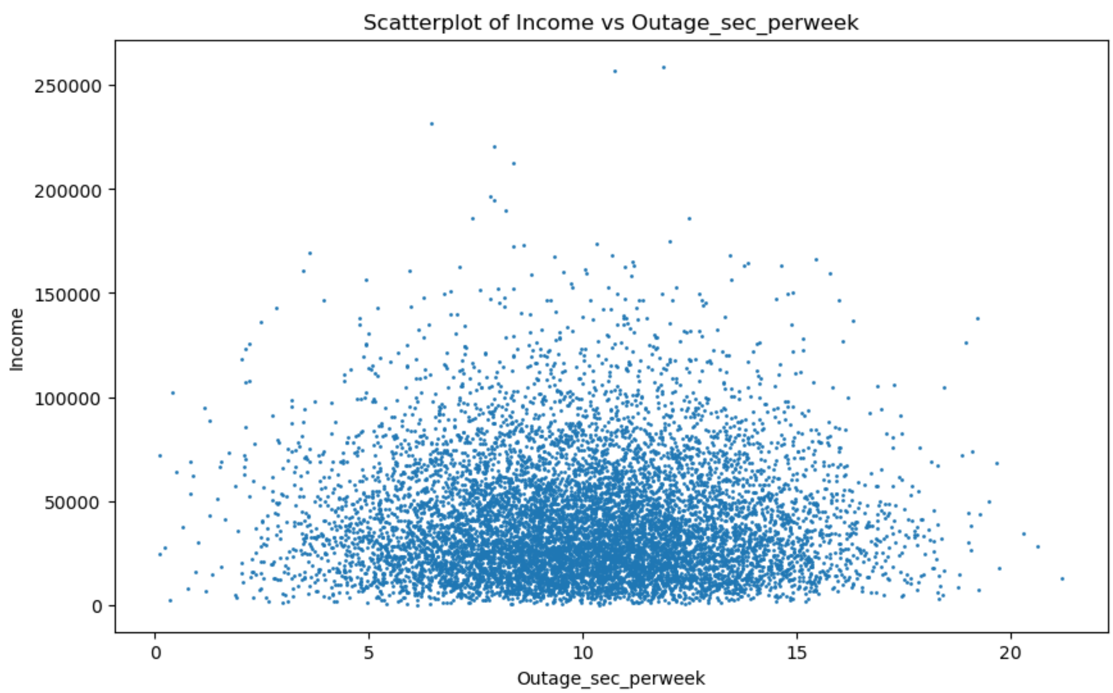
```
    plt.suptitle('')
    plt.xlabel(cat_col)
    plt.ylabel(cont_col)
    plt.show()
```

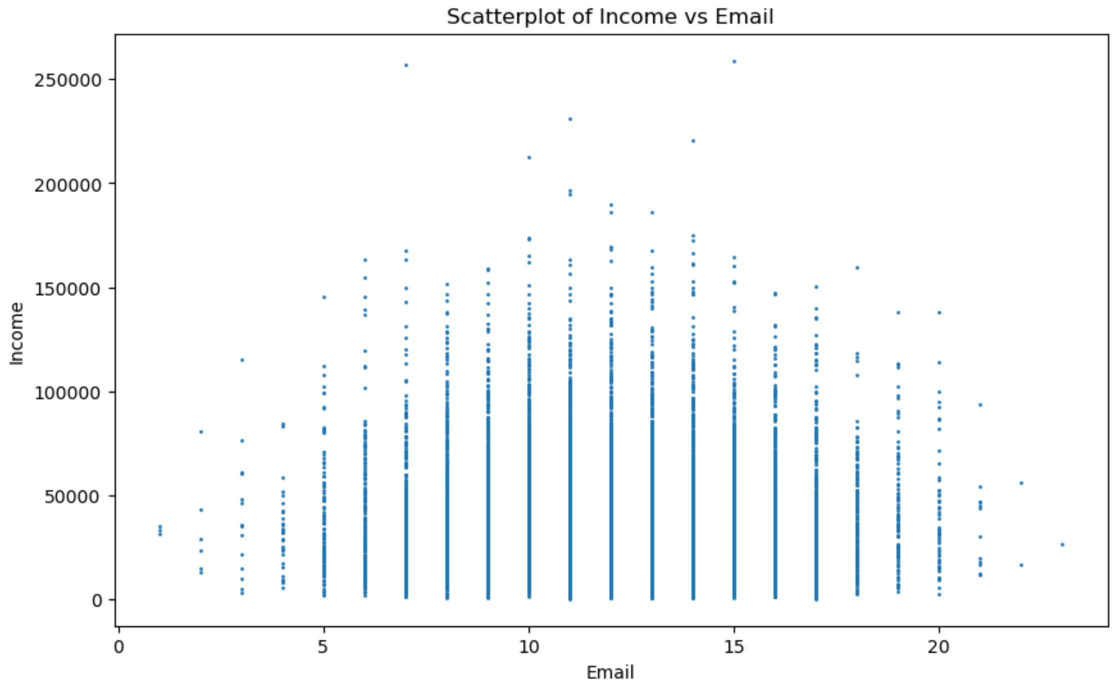
[17]:
```
## C3 Bivariate Visualizations
# Scatterplots of Numeric Variables

plot_bivarcont(df, 'Population', 'Income')

plot_bivarcont(df, 'Children', 'Income')

plot_bivarcont(df, 'Age', 'Income')

plot_bivarcont(df, 'Outage_sec_perweek', 'Income')

plot_bivarcont(df, 'Email', 'Income')

plot_bivarcont(df, 'Contacts', 'Income')

plot_bivarcont(df, 'Yearly_equip_failure', 'Income')

plot_bivarcont(df, 'Tenure', 'Income')

plot_bivarcont(df, 'MonthlyCharge', 'Income')

plot_bivarcont(df, 'Bandwidth_GB_Year', 'Income')

plot_bivarcont(df, 'Lat', 'Income')

plot_bivarcont(df, 'Lng', 'Income')
```


Scatterplot of Income vs Population
Income
Population
0
50000
100000
150000
200000
250000
0
20000
40000
60000
80000
100000

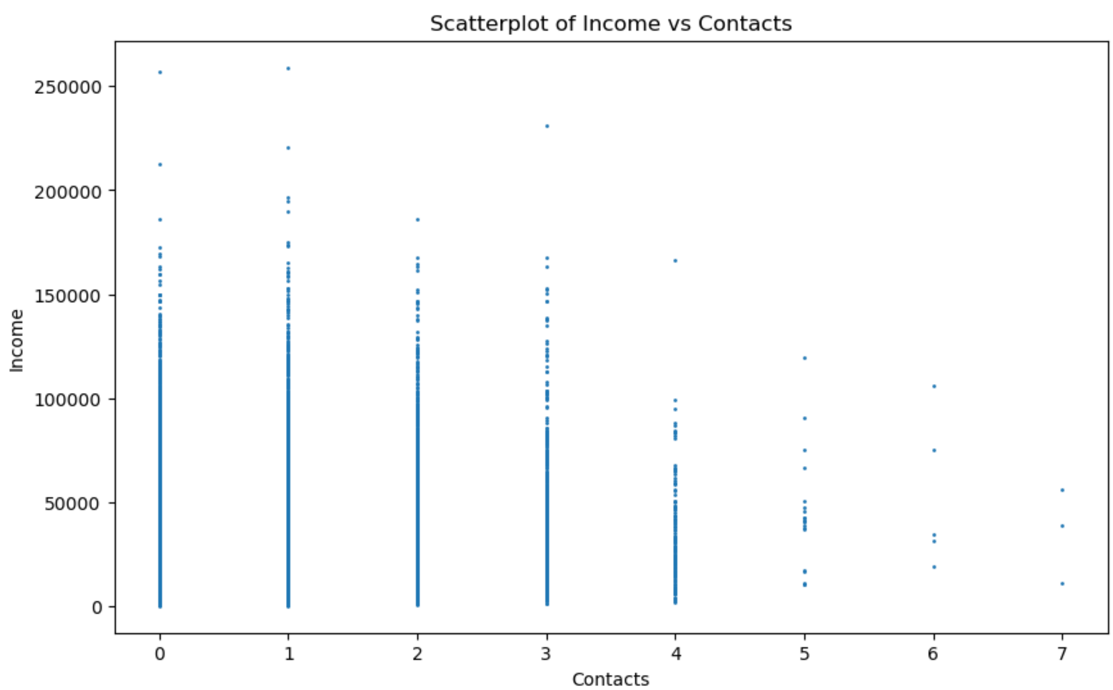

Scatterplot of Income vs Children
Income
Children
0
50000
100000
150000
200000
250000
0
2
4
6
8
10


Scatterplot of Income vs Age
Income
Age
250000
200000
150000
100000
50000
0
20
30
40
50
60
70
80
90

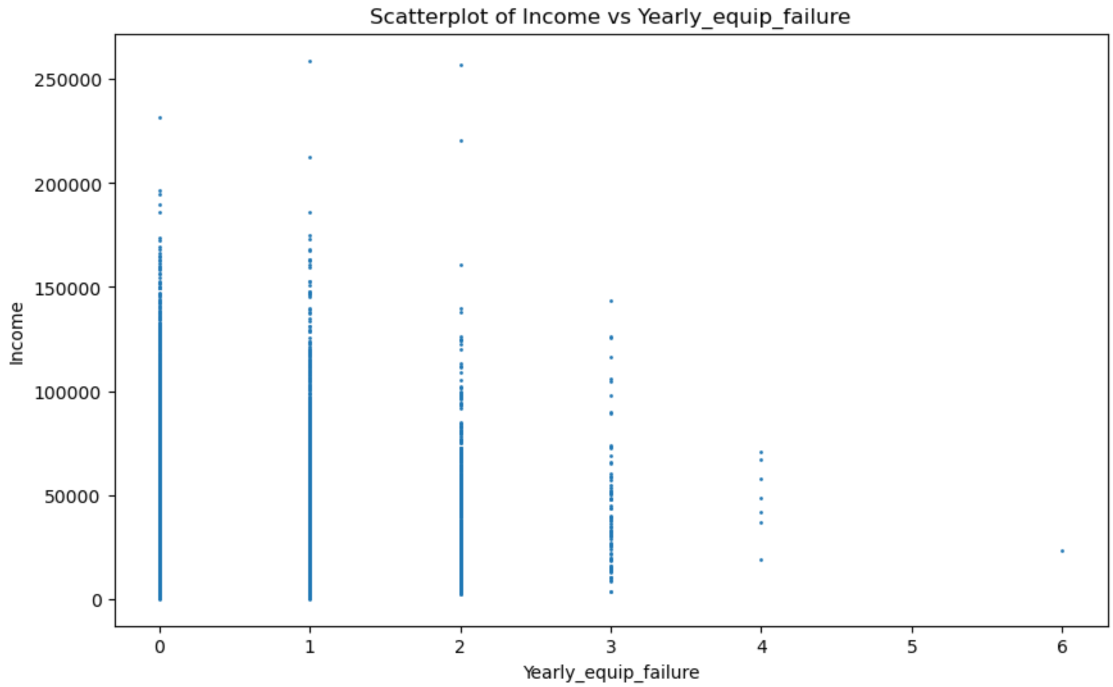

Scatterplot of Income vs Outage_sec_perweek
Income
Outage_sec_perweek
250000
200000
150000
100000
50000
0
0
5
10
15
20



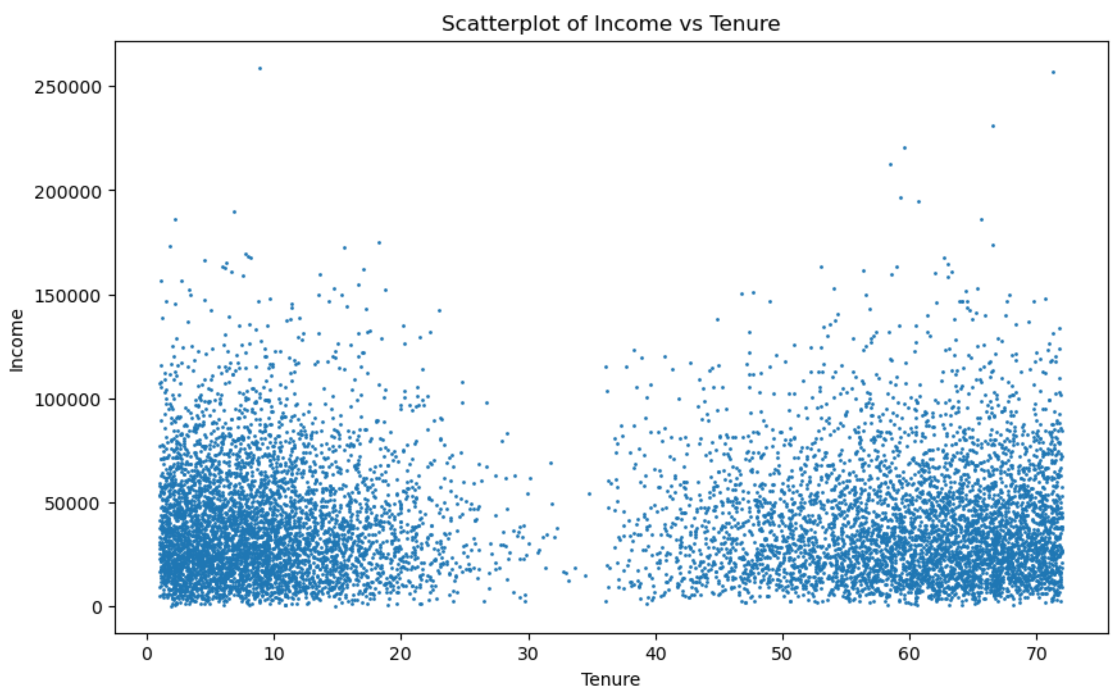
Scatterplot of Income vs Email
Income
Email
0
50000
100000
150000
200000
250000
5
10
15
20



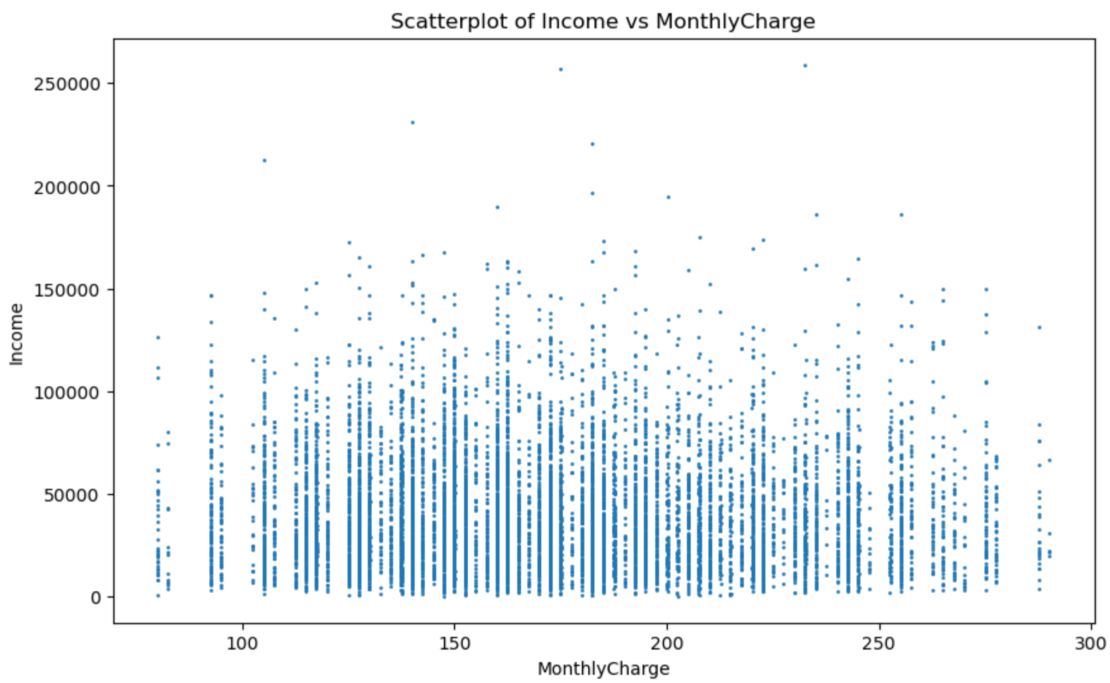
Scatterplot of Income vs Contacts
Income
Contacts
0
50000
100000
150000
200000
250000
1
2
3
4
5
6
7



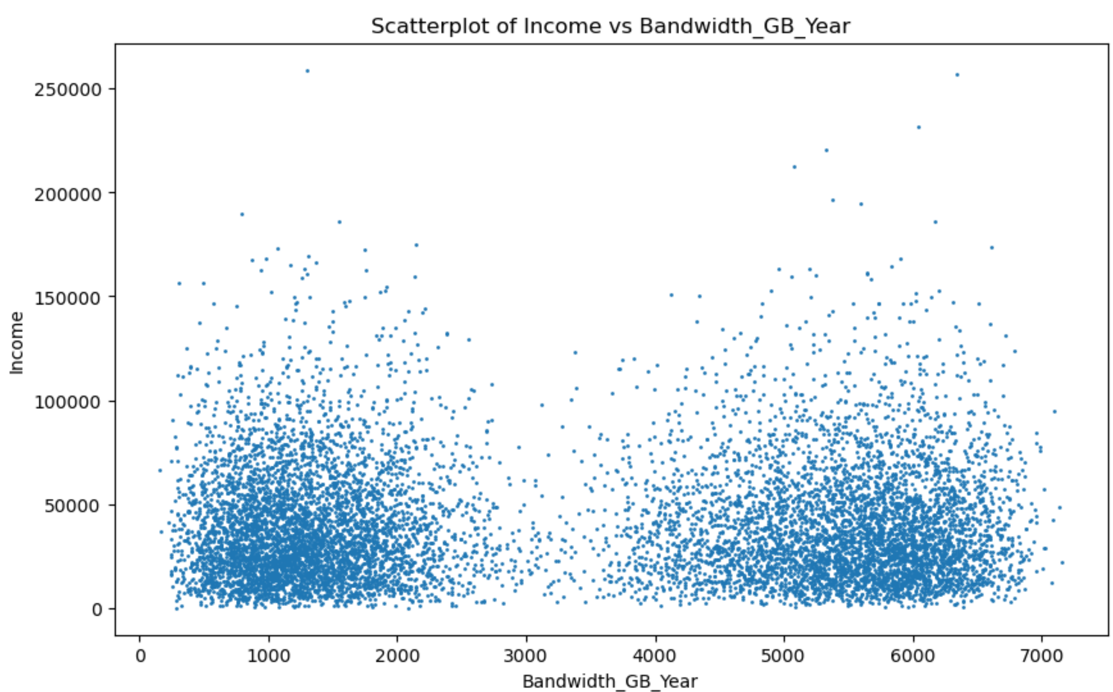
Scatterplot of Income vs Yearly_equip_failure
Income
Yearly_equip_failure



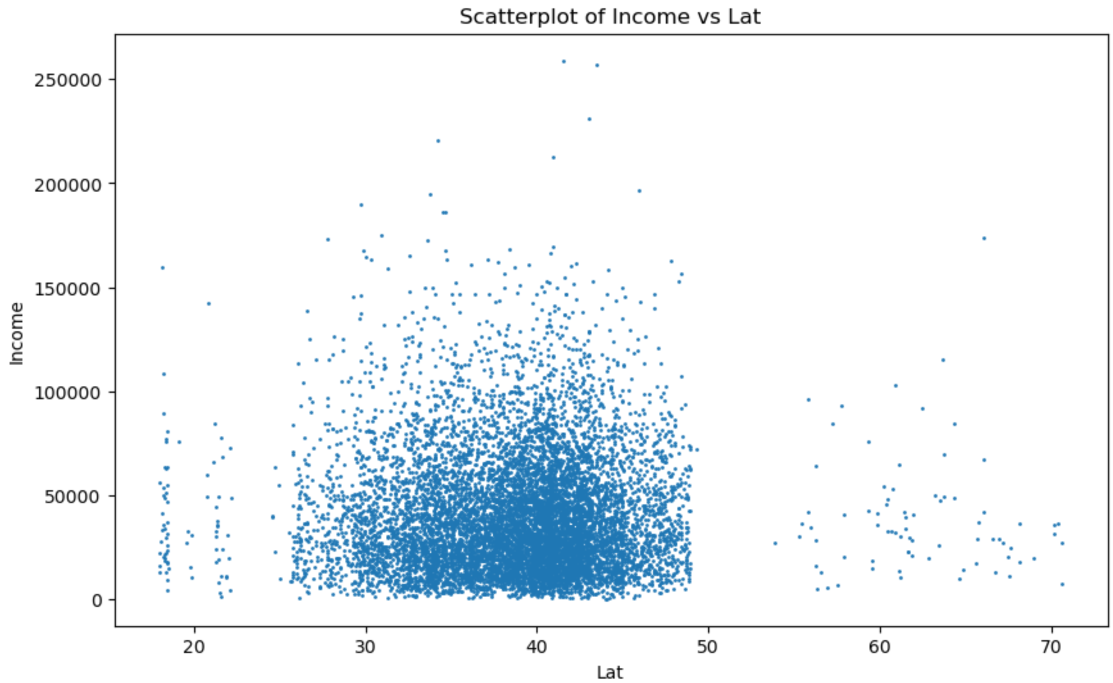
Scatterplot of Income vs Tenure
Income
Tenure

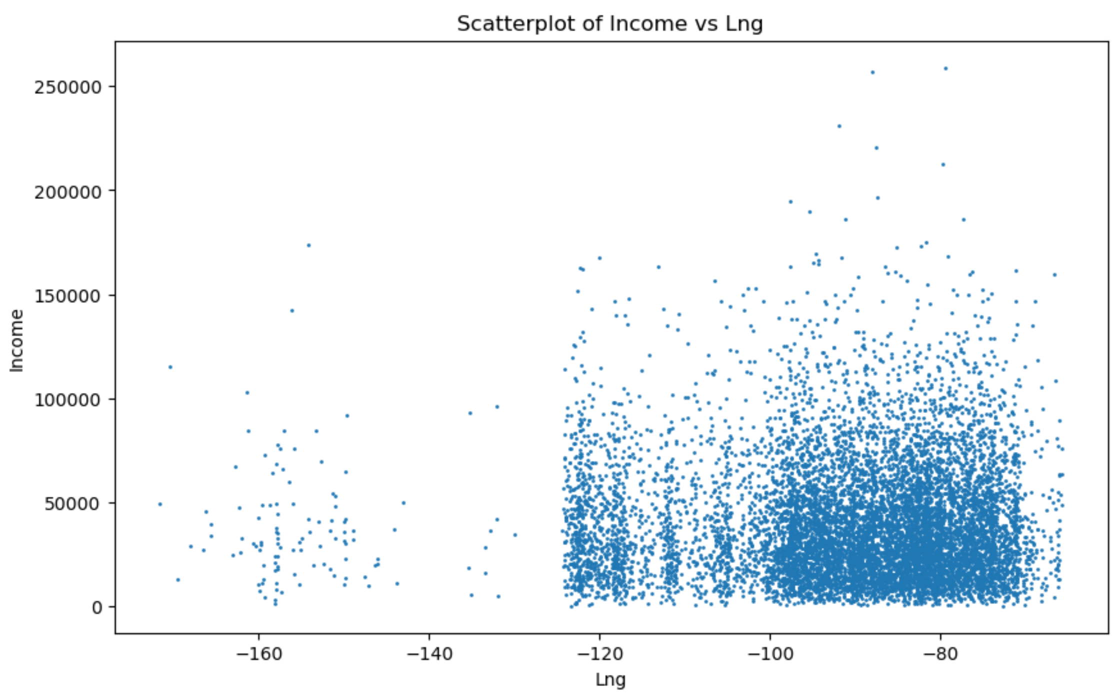


Scatterplot of Income vs MonthlyCharge
Income
MonthlyCharge
0
50000
100000
150000
200000
250000
100
150
200
250
300



Scatterplot of Income vs Bandwidth_GB_Year
Income
Bandwidth_GB_Year
0
50000
100000
150000
200000
250000
0
1000
2000
3000
4000
5000
6000
7000

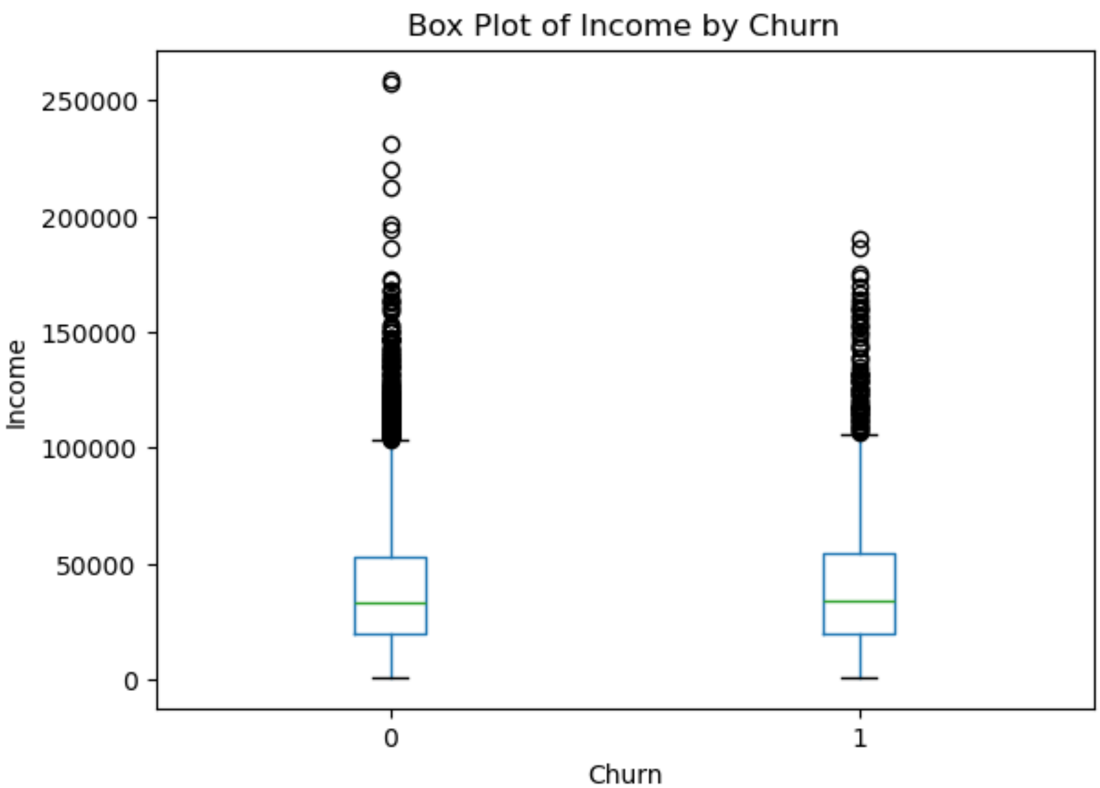


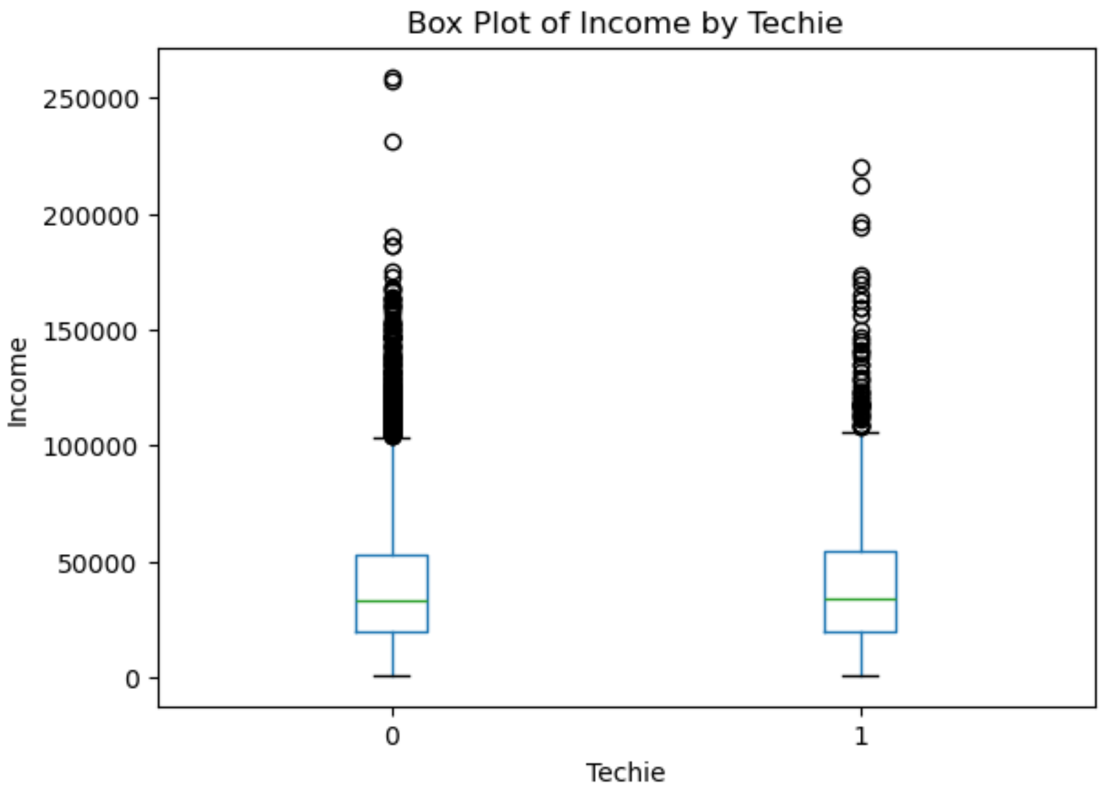


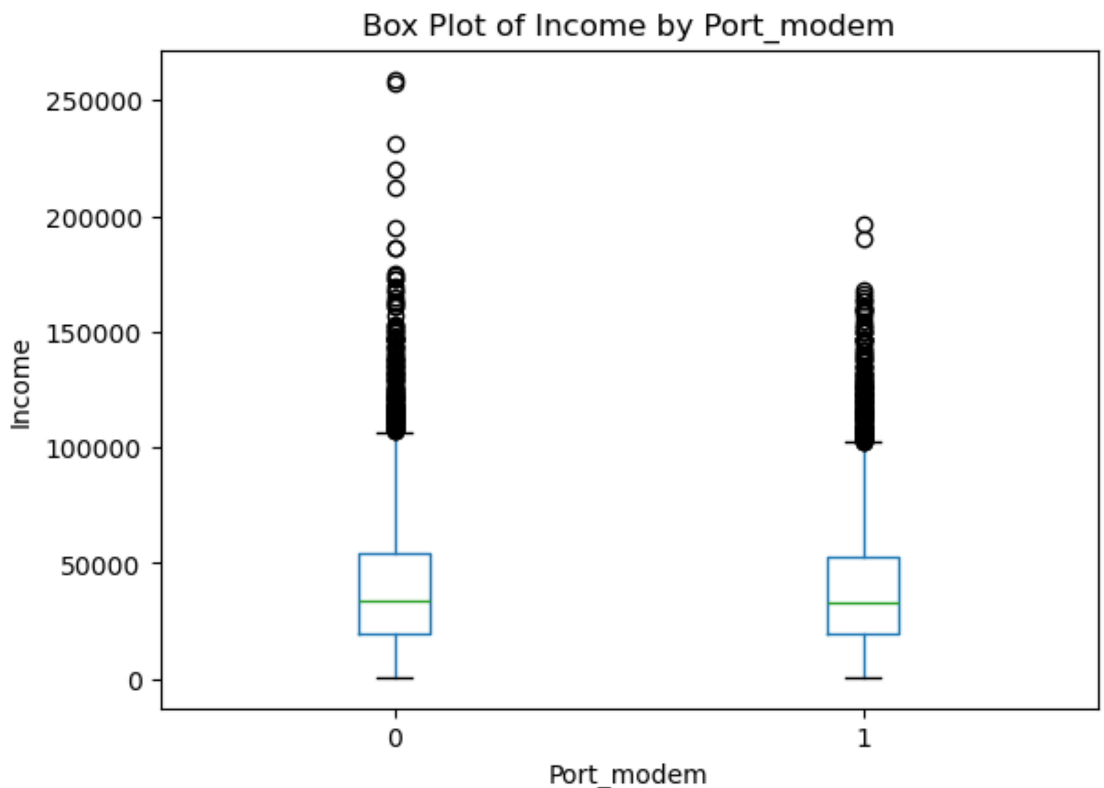
[18]:
```
## C3 Bivariate Visualizations
# Boxplots of Dependent Variable vs Categorical Variables
```

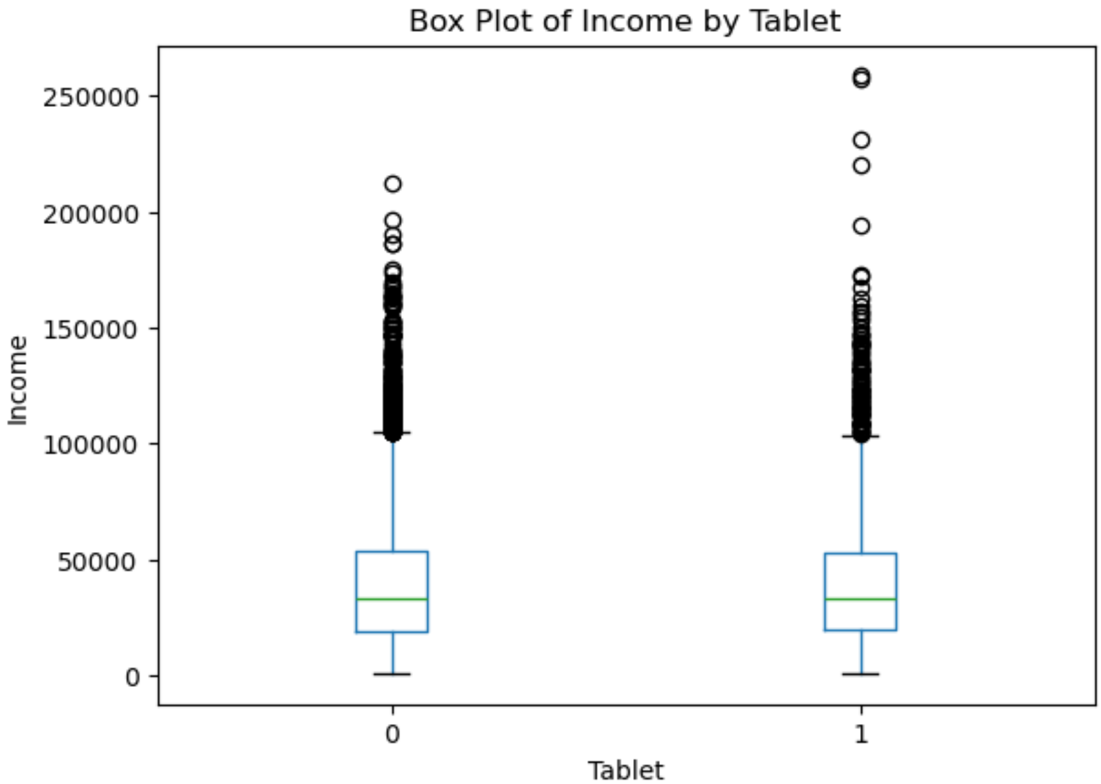
```
plot_bivarcat(df, 'Income', 'Churn')

plot_bivarcat(df, 'Income', 'Techie')

plot_bivarcat(df, 'Income', 'Port_modem')

plot_bivarcat(df, 'Income', 'Tablet')

plot_bivarcat(df, 'Income', 'Phone')

plot_bivarcat(df, 'Income', 'Multiple')

plot_bivarcat(df, 'Income', 'OnlineSecurity')

plot_bivarcat(df, 'Income', 'OnlineBackup')

plot_bivarcat(df, 'Income', 'DeviceProtection')

plot_bivarcat(df, 'Income', 'TechSupport')

plot_bivarcat(df, 'Income', 'StreamingTV')

plot_bivarcat(df, 'Income', 'StreamingMovies')

plot_bivarcat(df, 'Income', 'PaperlessBilling')


plot_bivarcat(df, 'Income', 'Area')

plot_bivarcat(df, 'Income', 'Marital')

plot_bivarcat(df, 'Income', 'Gender')

plot_bivarcat(df, 'Income', 'Contract')

plot_bivarcat(df, 'Income', 'InternetService')

plot_bivarcat(df, 'Income', 'PaymentMethod')


plot_bivarcat(df, 'Income', 'Item1')

plot_bivarcat(df, 'Income', 'Item2')

plot_bivarcat(df, 'Income', 'Item3')
```

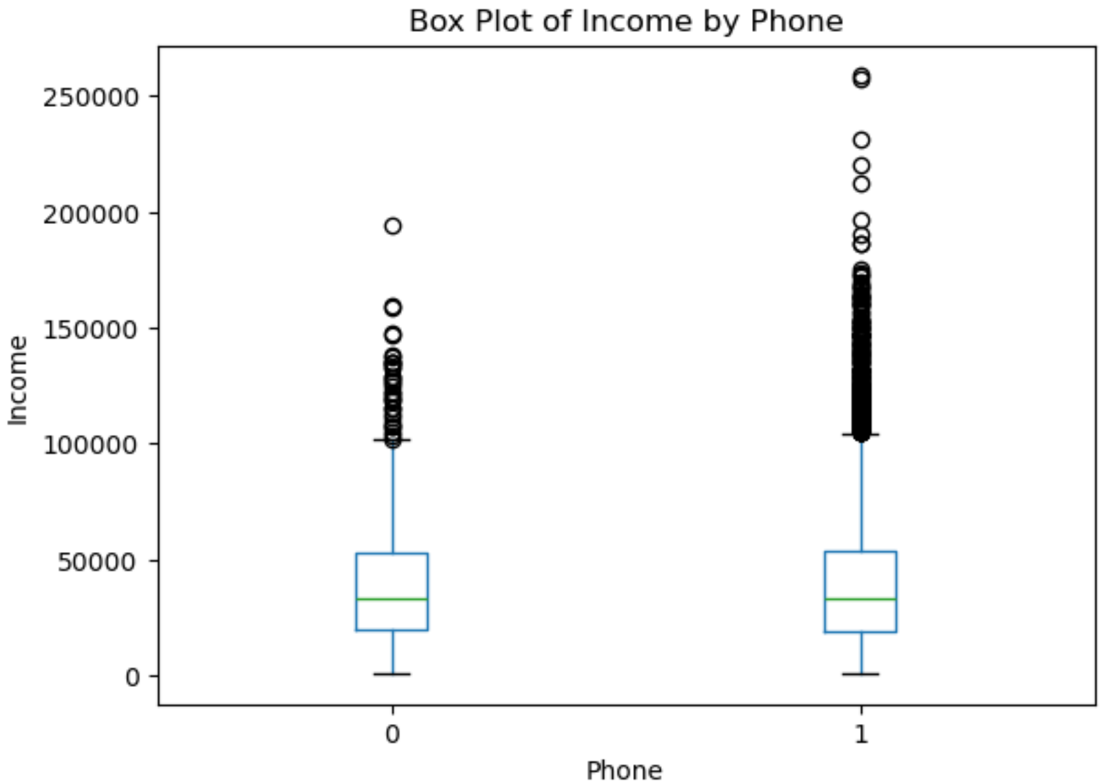
```
plot_bivarcat(df, 'Income', 'Item4')

plot_bivarcat(df, 'Income', 'Item5')

plot_bivarcat(df, 'Income', 'Item6')

plot_bivarcat(df, 'Income', 'Item7')

plot_bivarcat(df, 'Income', 'Item8')
```

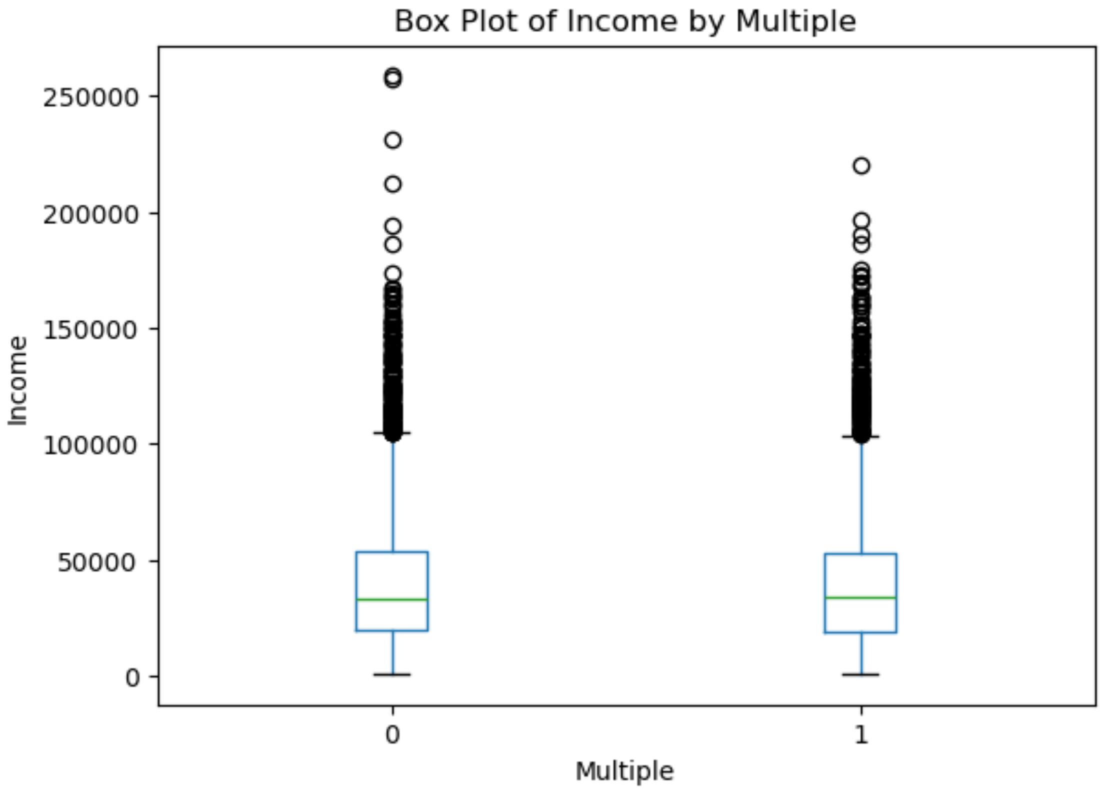


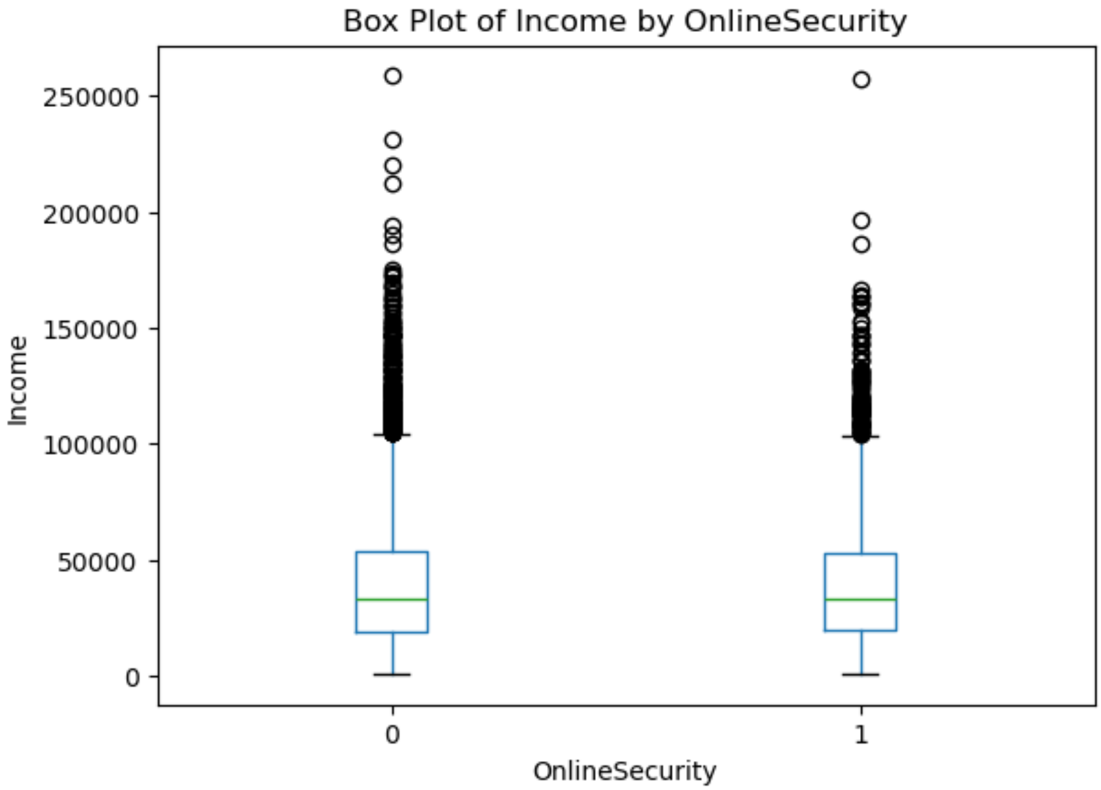

Box Plot of Income by Techie
250000
200000
150000
100000
50000
0
Income
0
1
Techie

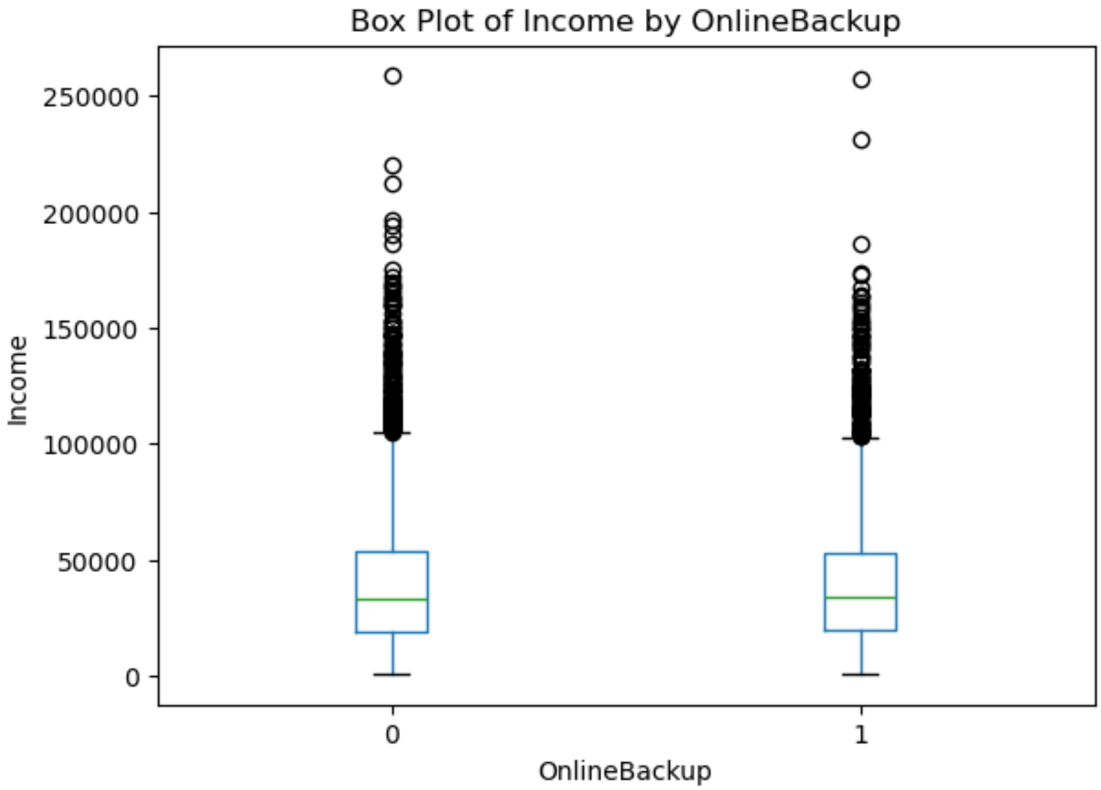

Box Plot of Income by Port_modem
250000
200000
150000
100000
50000
0
Income
0
1
Port_modem

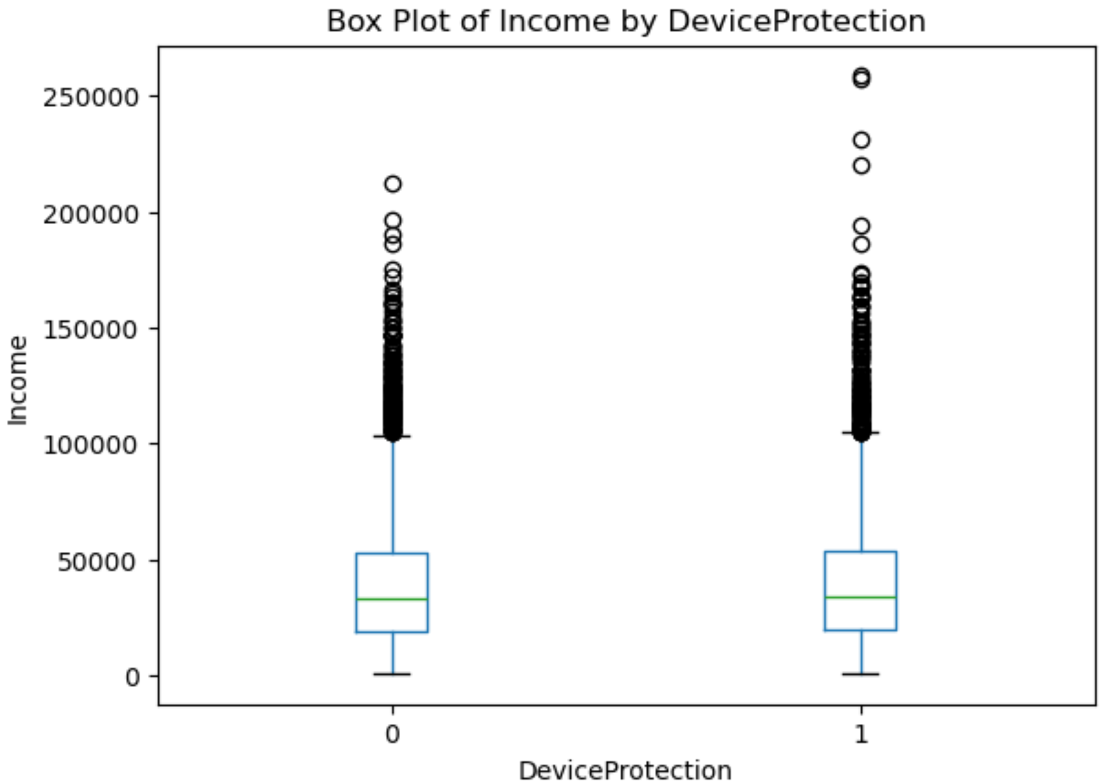

Box Plot of Income by Tablet
250000
200000
150000
100000
50000
0
Income
0
1
Tablet

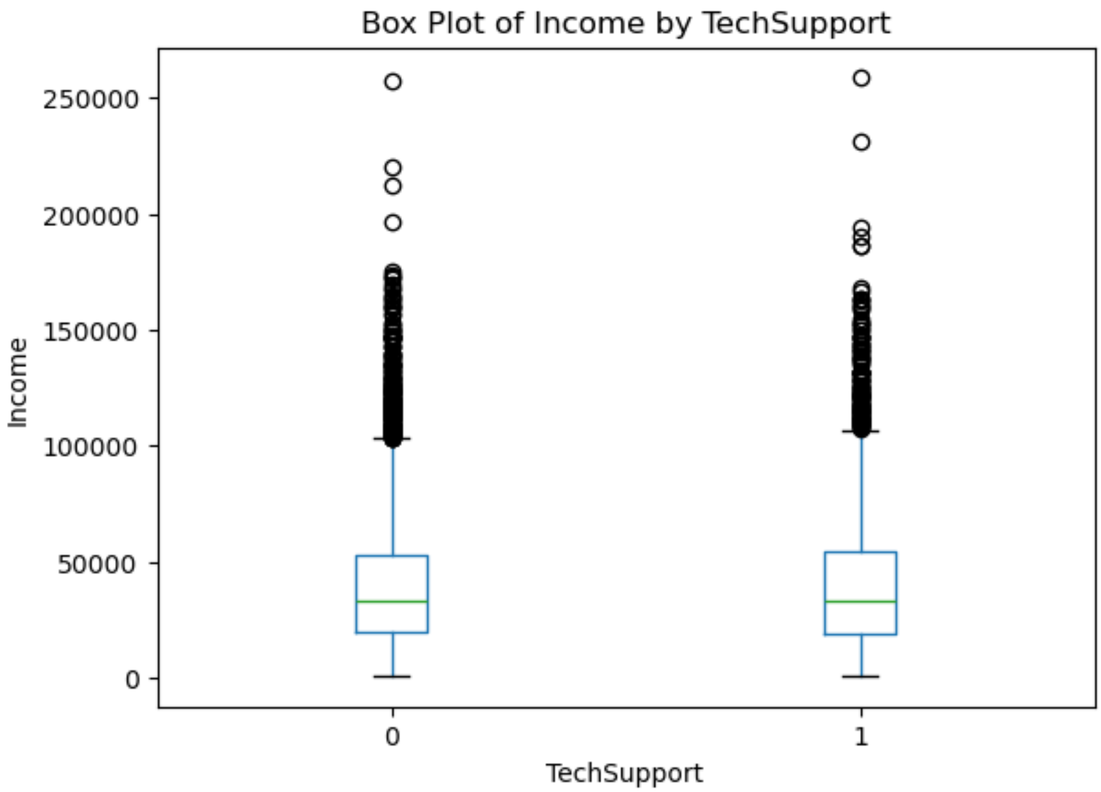

Box Plot of Income by Phone
Income
250000
200000
150000
100000
50000
0
0
1
Phone

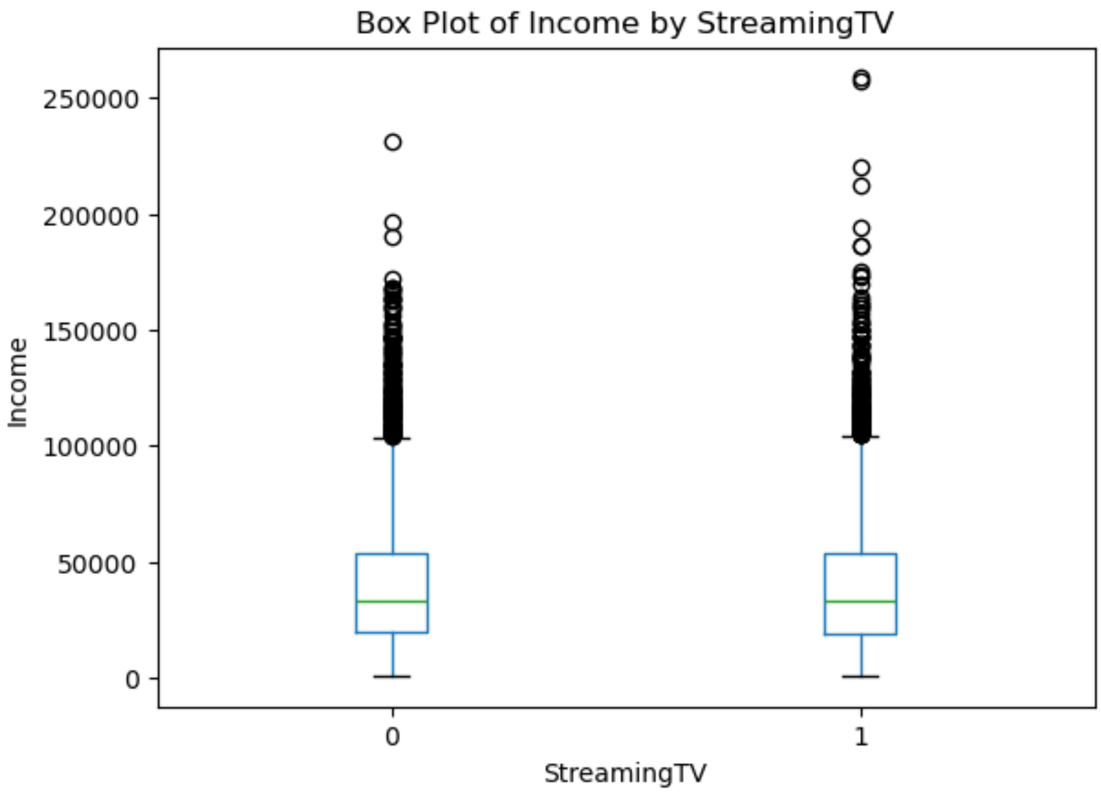

Box Plot of Income by Multiple
Income
250000
200000
150000
100000
50000
0
0
1
Multiple


Box Plot of Income by OnlineSecurity
250000
200000
150000
100000
50000
0
Income
0
1
OnlineSecurity

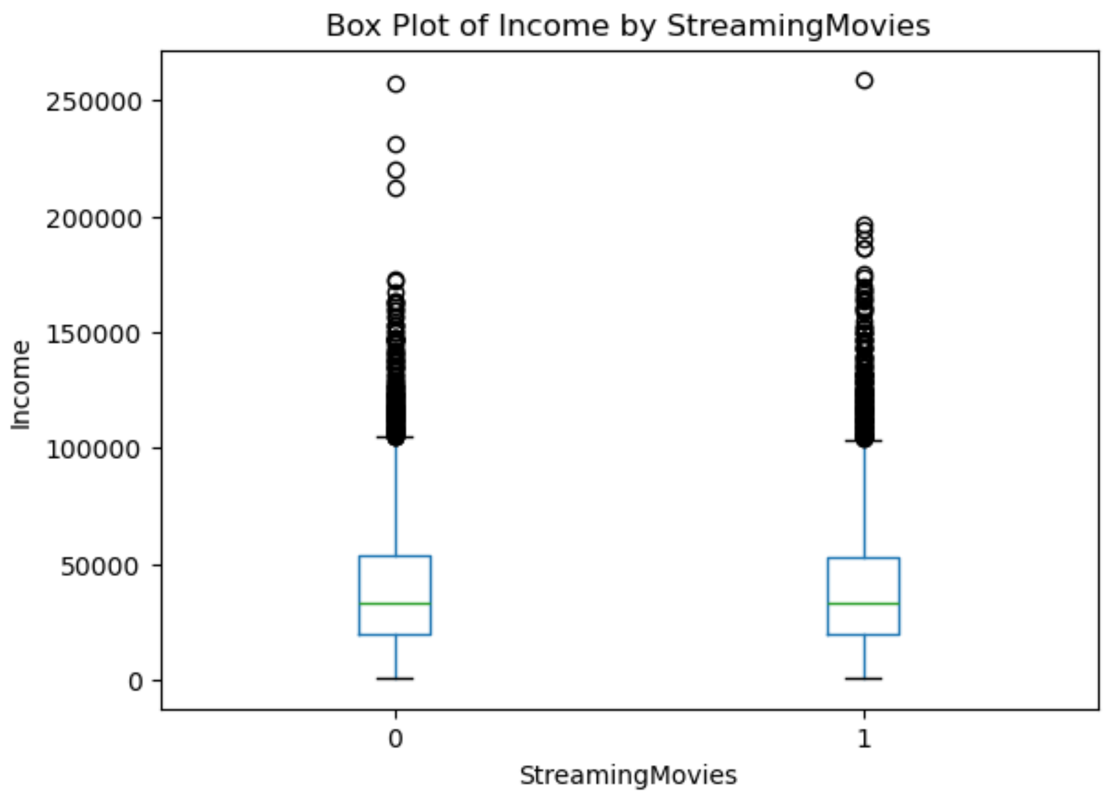

Box Plot of Income by OnlineBackup
250000
200000
150000
100000
50000
0
Income
0
1
OnlineBackup

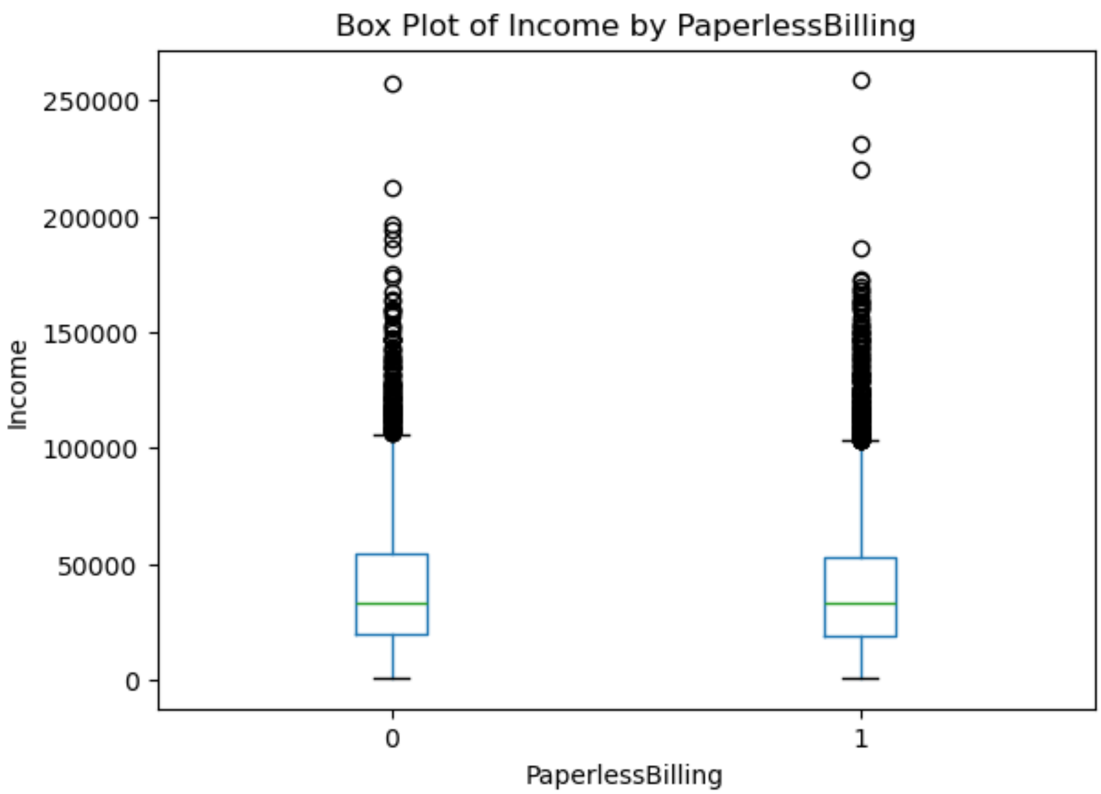

Box Plot of Income by DeviceProtection
Income
250000
200000
150000
100000
50000
0
0
1
DeviceProtection

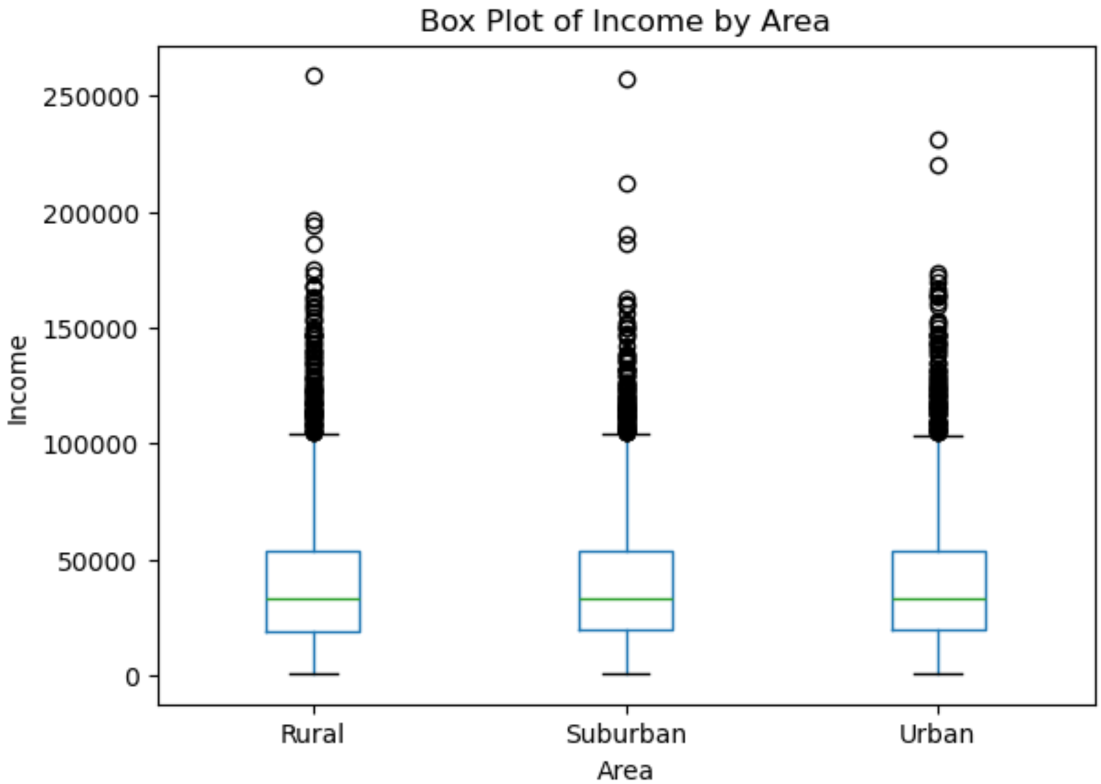

Box Plot of Income by TechSupport
250000
200000
150000
100000
50000
0
Income
0
1
TechSupport

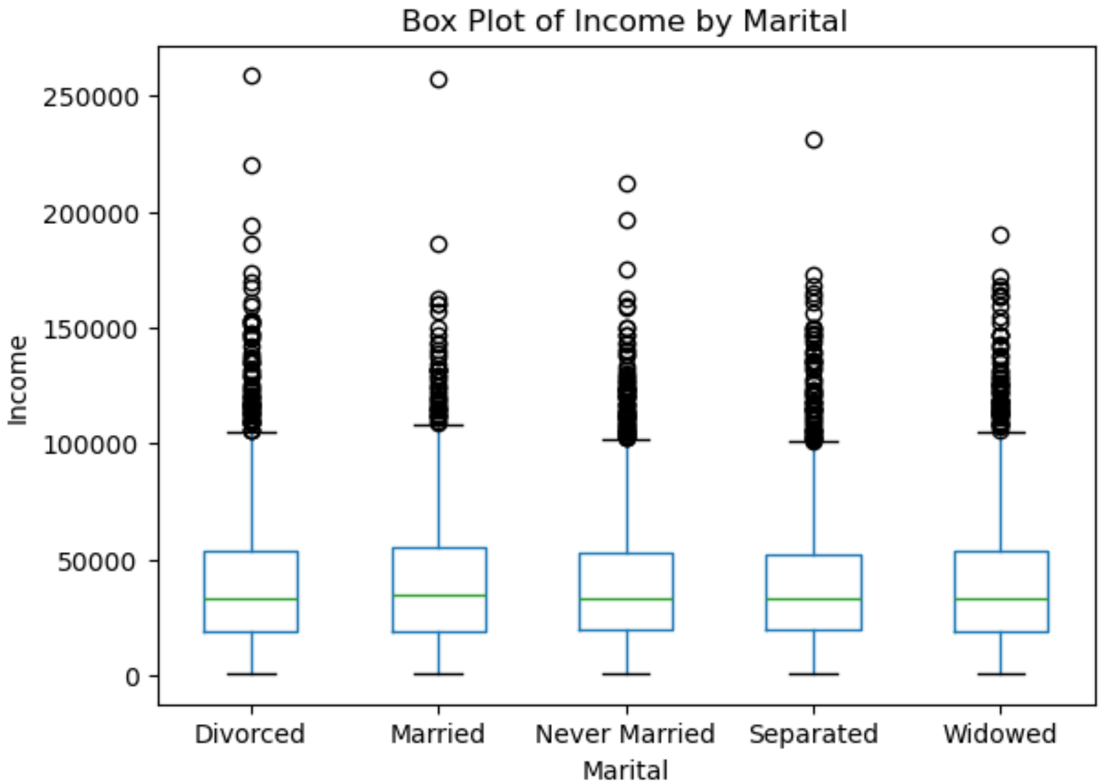


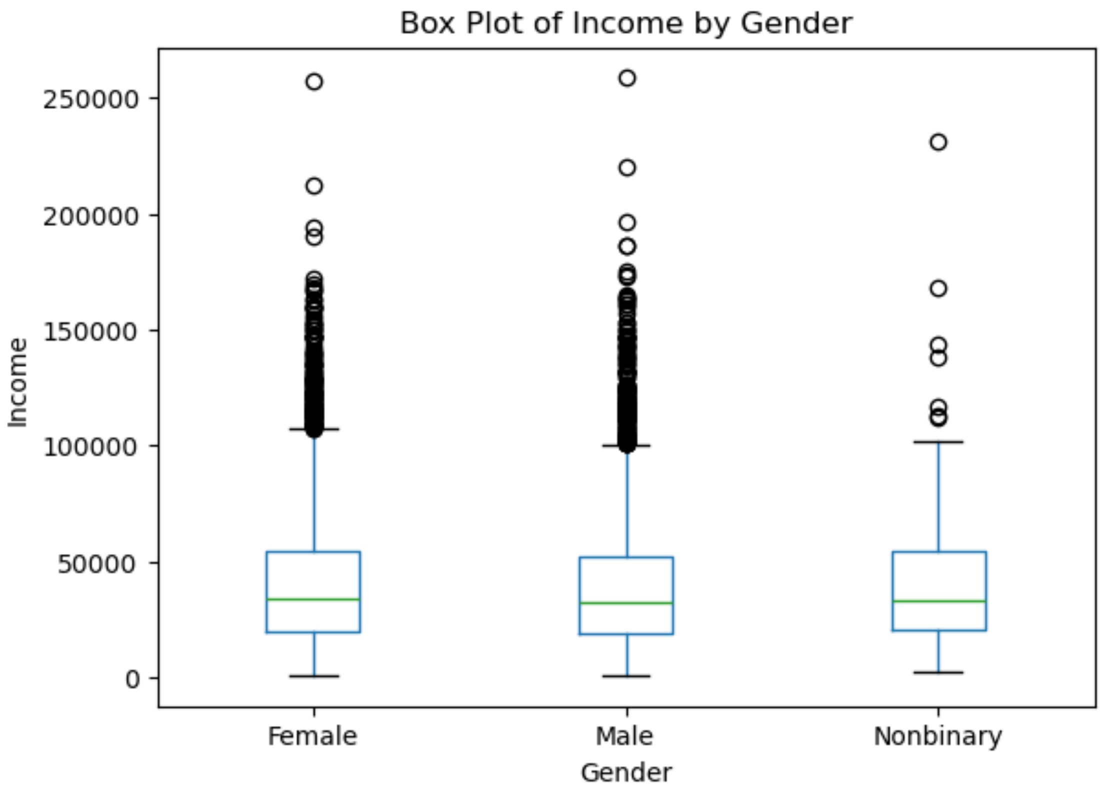
Box Plot of Income by StreamingTV
250000
200000
150000
100000
50000
0
Income
0
1
StreamingTV

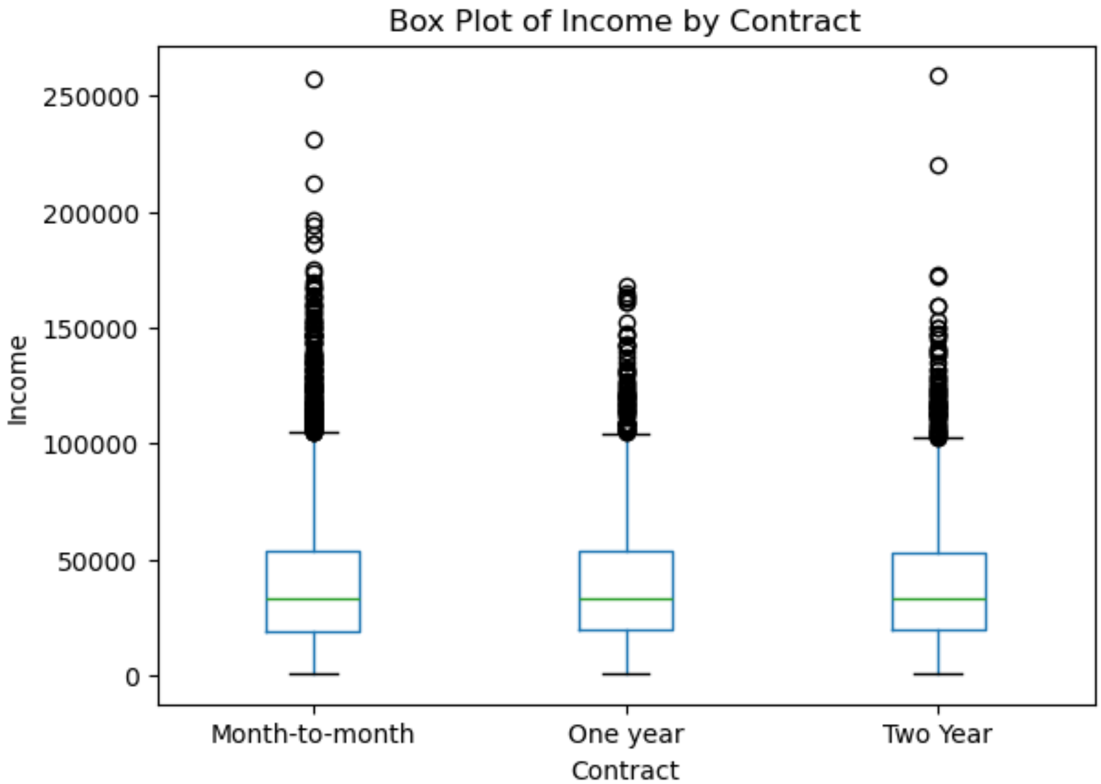

Box Plot of Income by StreamingMovies
Income
250000
200000
150000
100000
50000
0
0
1
StreamingMovies

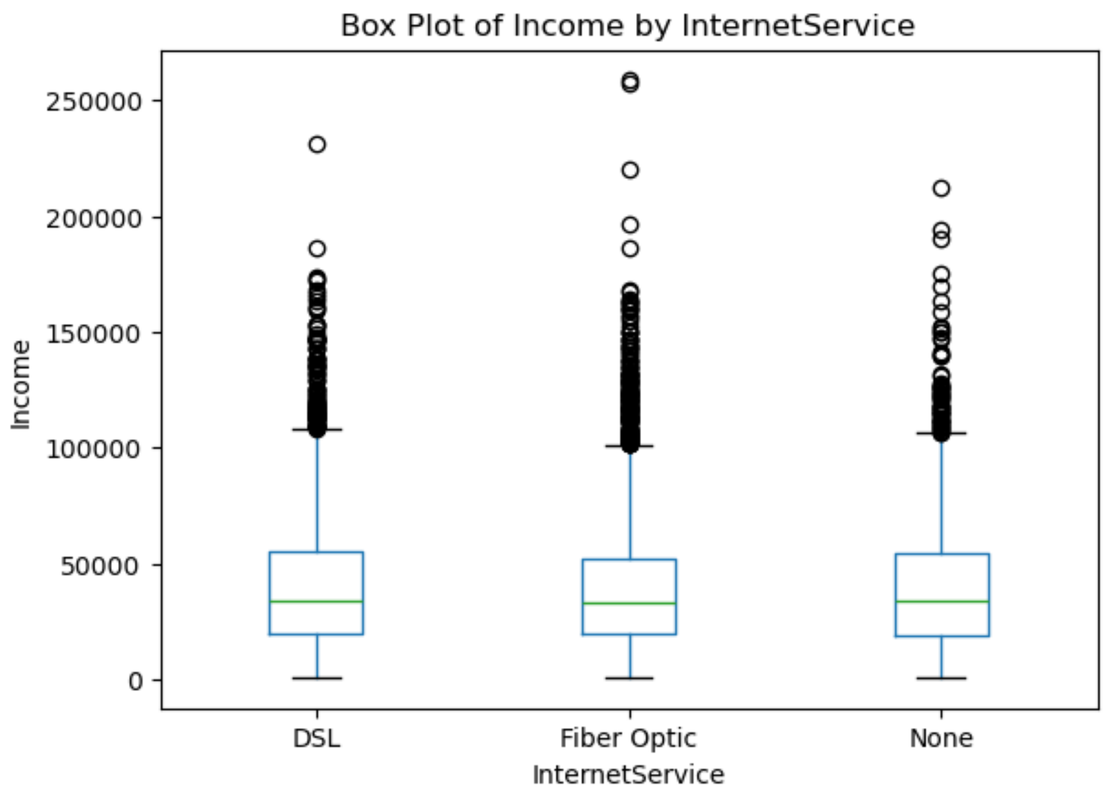

Box Plot of Income by PaperlessBilling
Income
250000
200000
150000
100000
50000
0
0
1
PaperlessBilling

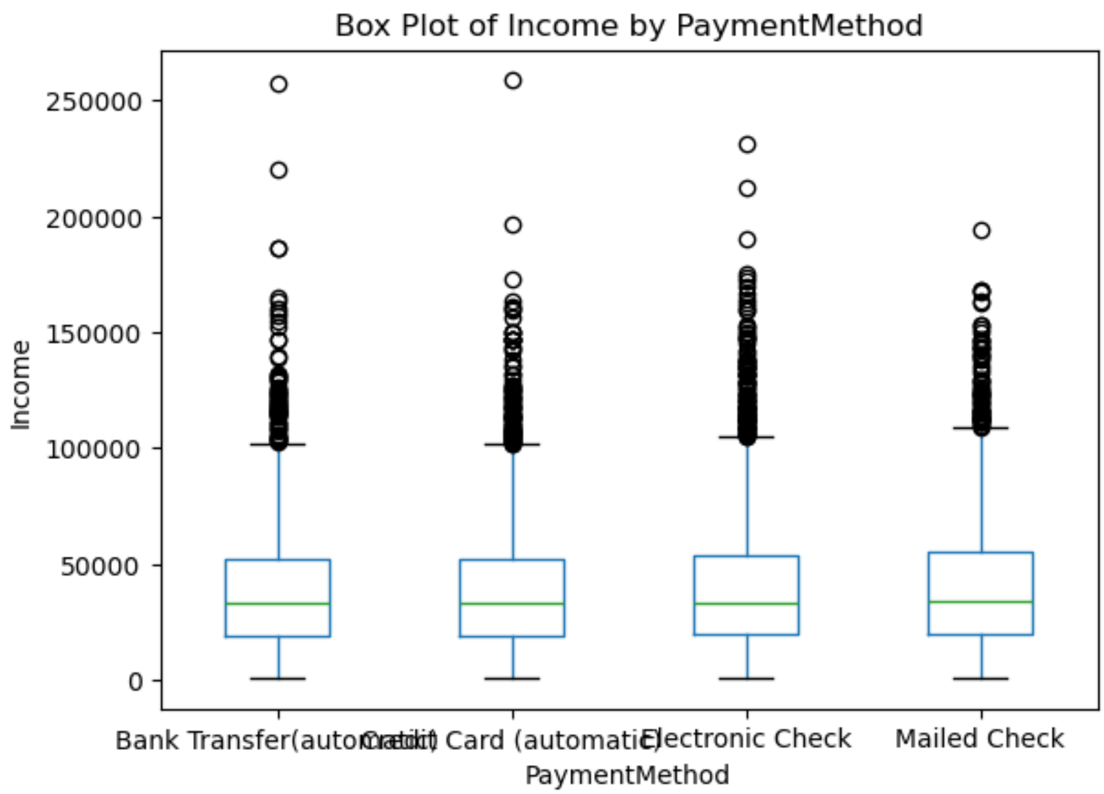

Box Plot of Income by Area
Income
250000
200000
150000
100000
50000
0
Rural
Suburban
Urban
Area


Box Plot of Income by Marital
250000
200000
150000
100000
50000
0
Income
Divorced
Married
Never Married
Separated
Widowed
Marital

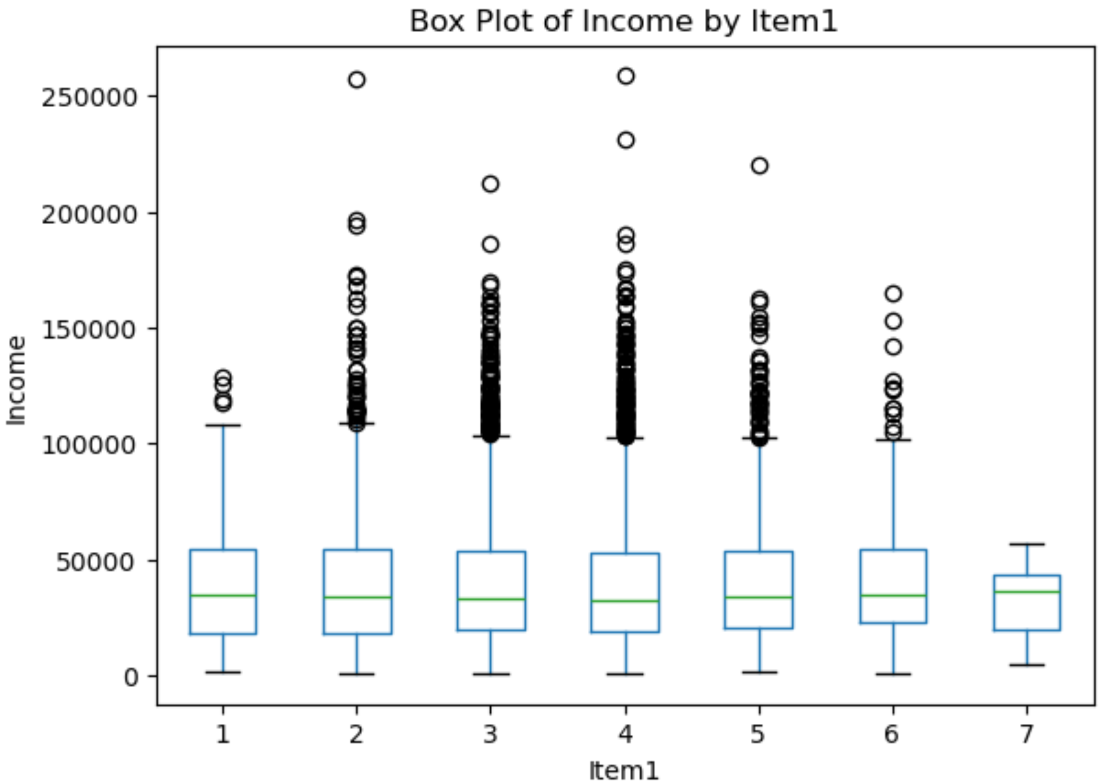

Box Plot of Income by Gender
250000
200000
150000
100000
50000
0
Income
Female
Male
Nonbinary
Gender

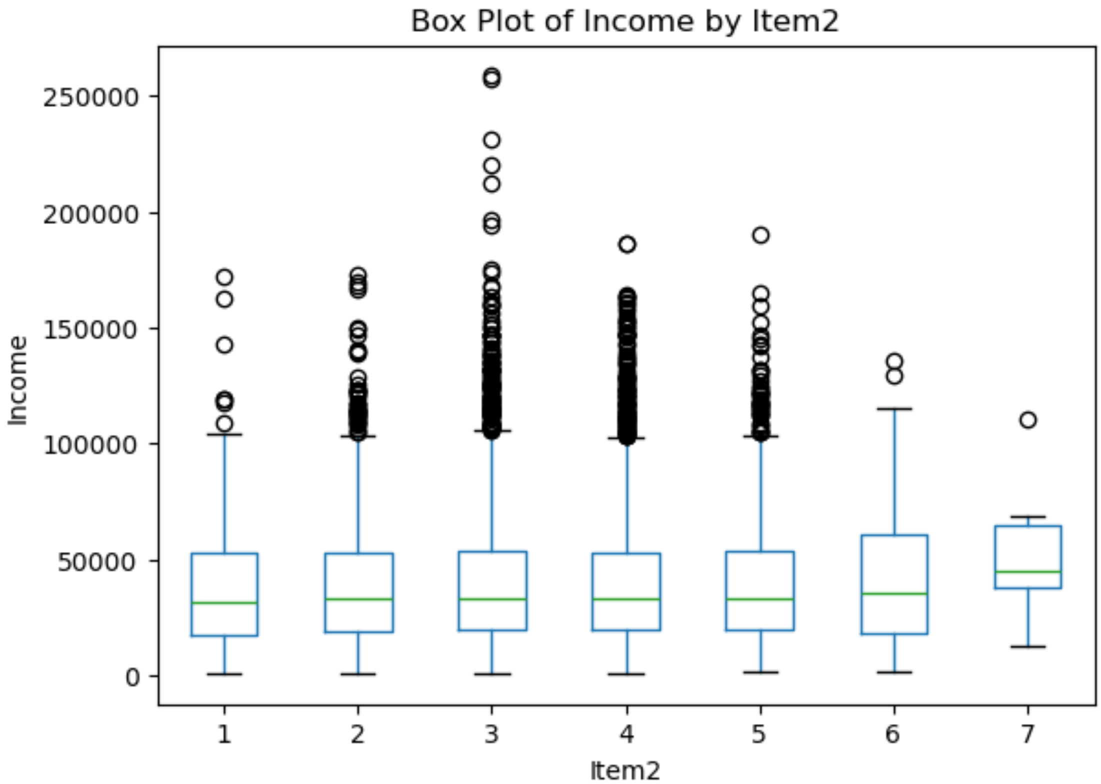

Box Plot of Income by Contract
Income
250000
200000
150000
100000
50000
0
Month-to-month
One year
Two Year
Contract

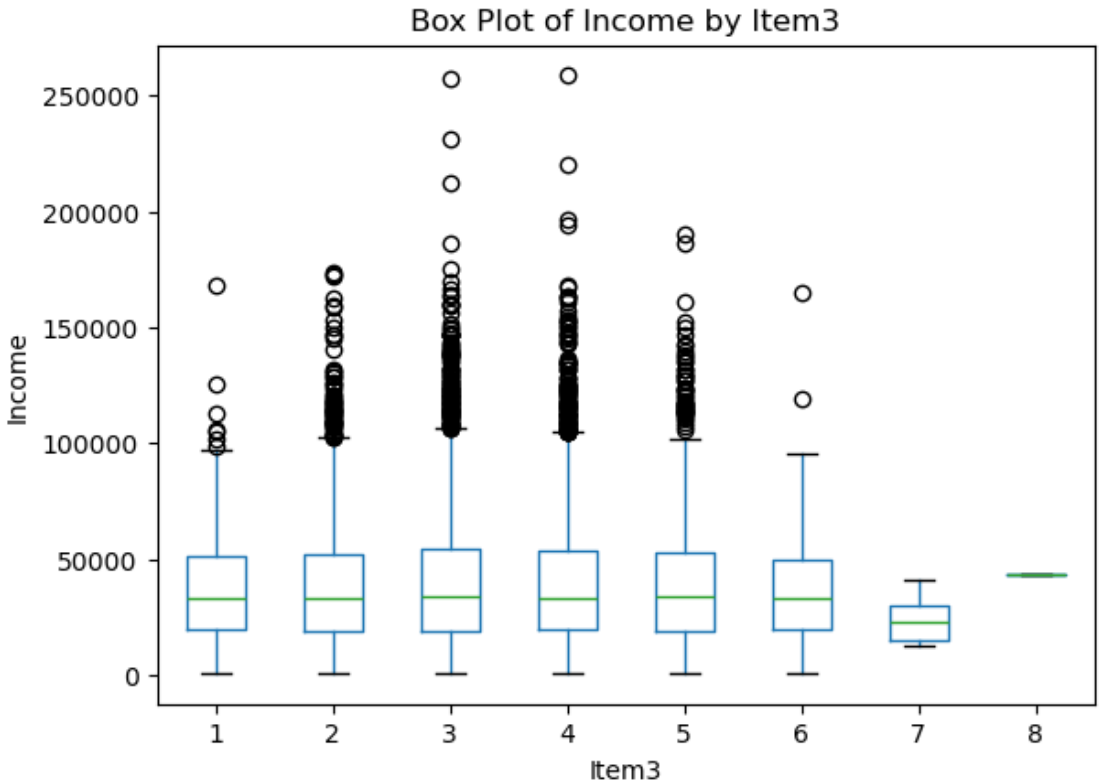

Box Plot of Income by InternetService
250000
200000
150000
100000
50000
0
Income
DSL
Fiber Optic
None
InternetService

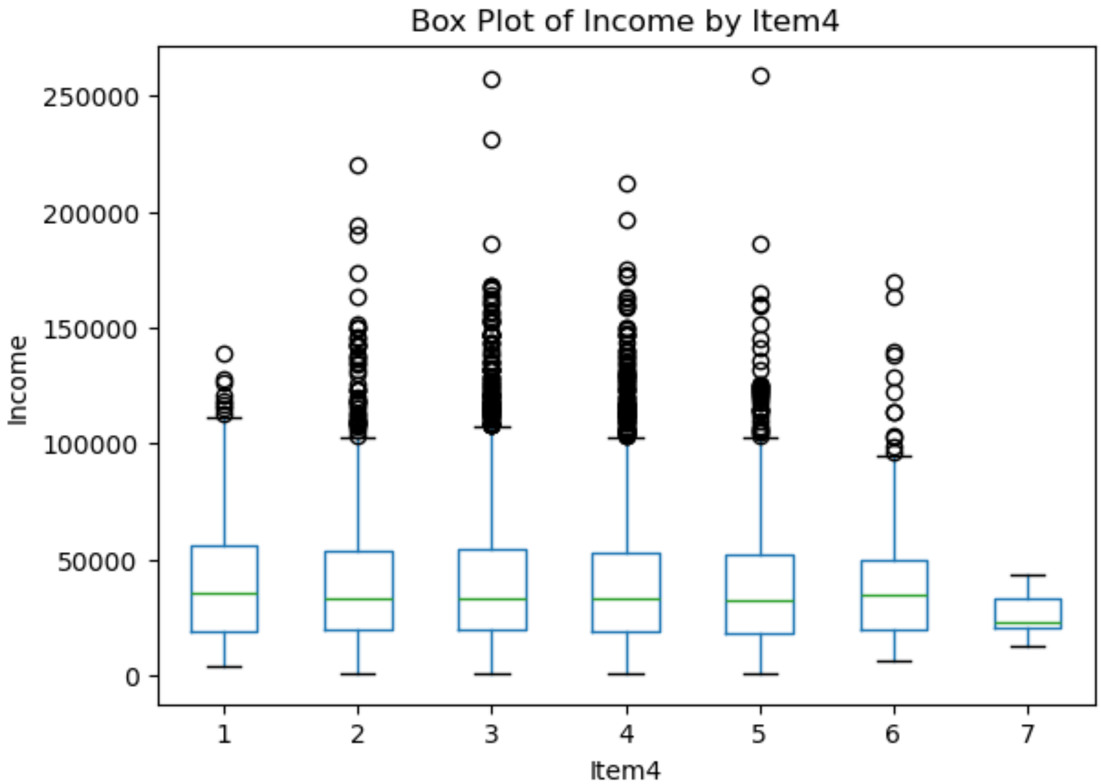


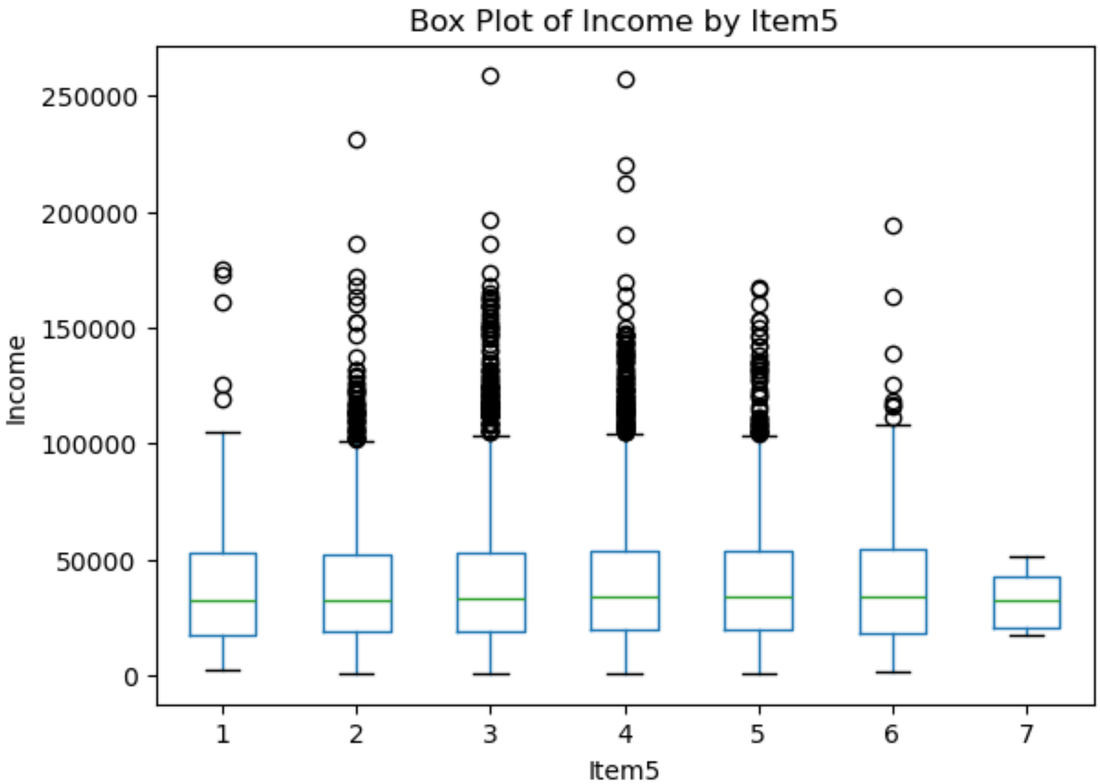
Box Plot of Income by PaymentMethod
250000
200000
150000
100000
50000
0
Income
Bank Transfer(automatic)
Credit Card (automatic)
Electronic Check
Mailed Check
PaymentMethod

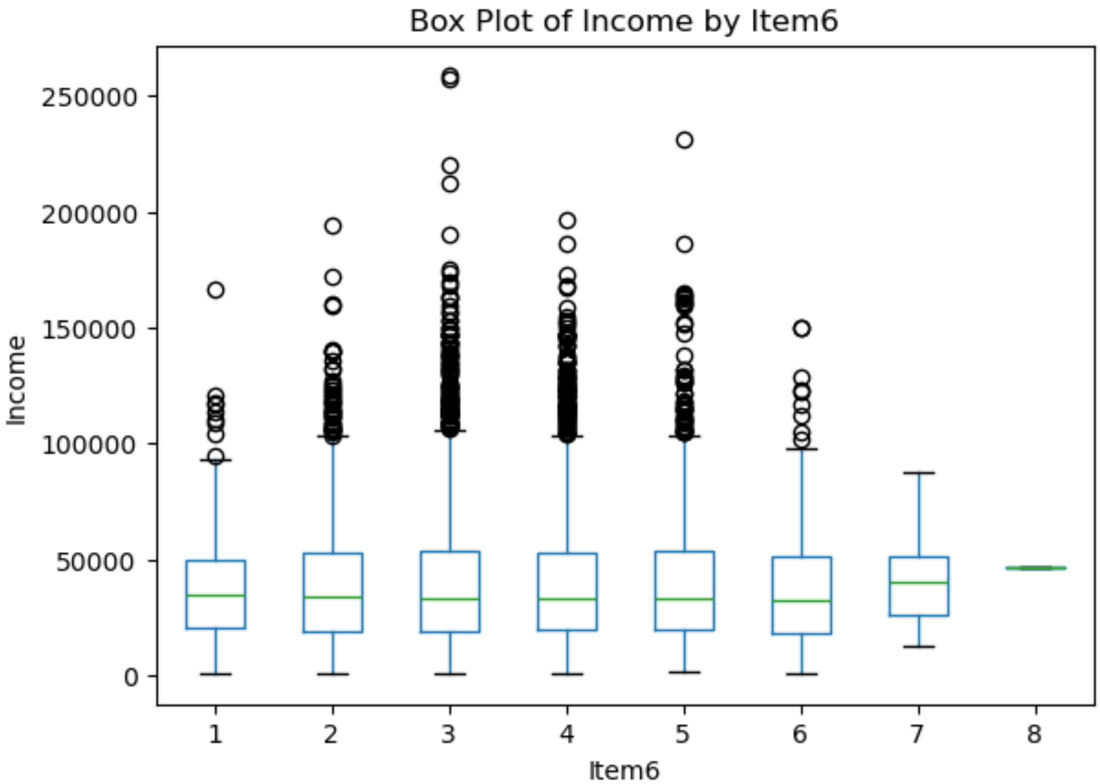

Box Plot of Income by Item1
250000
200000
150000
100000
50000
0
Income
1
2
3
4
5
6
7
Item1

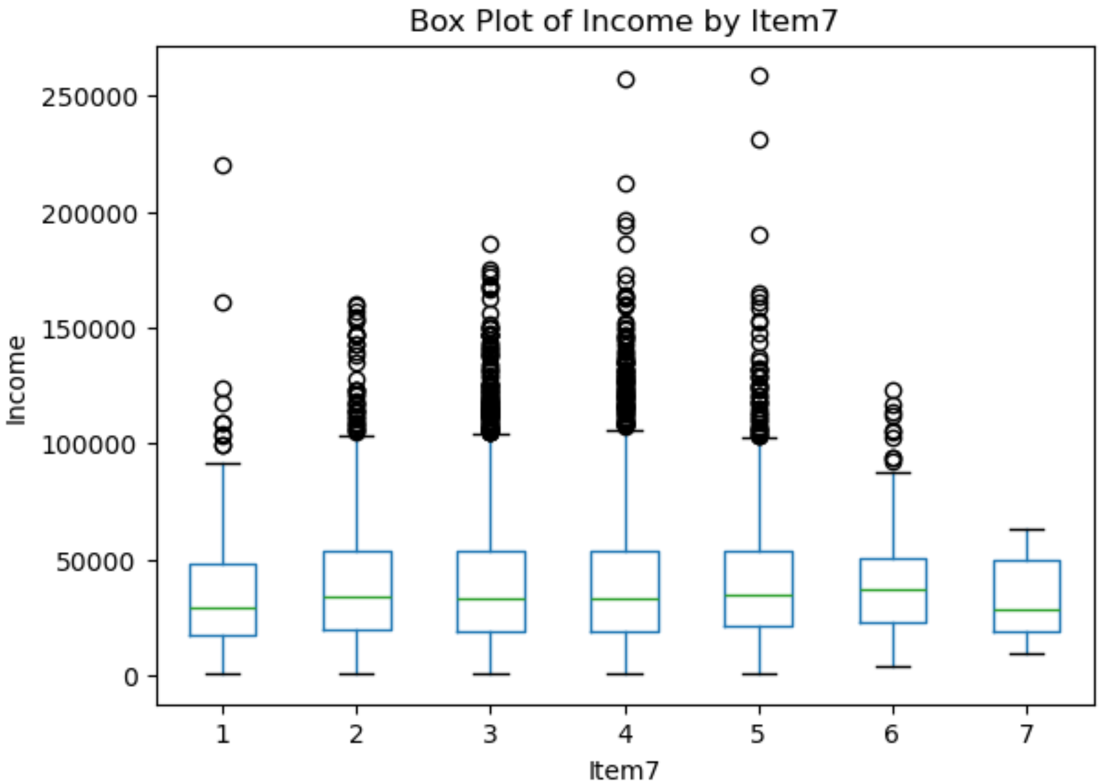

Box Plot of Income by Item2
Income
0
50000
100000
150000
200000
250000
Item2
1
2
3
4
5
6
7


Box Plot of Income by Item3
Income
250000
200000
150000
100000
50000
0
1
2
3
4
5
6
7
8
Item3

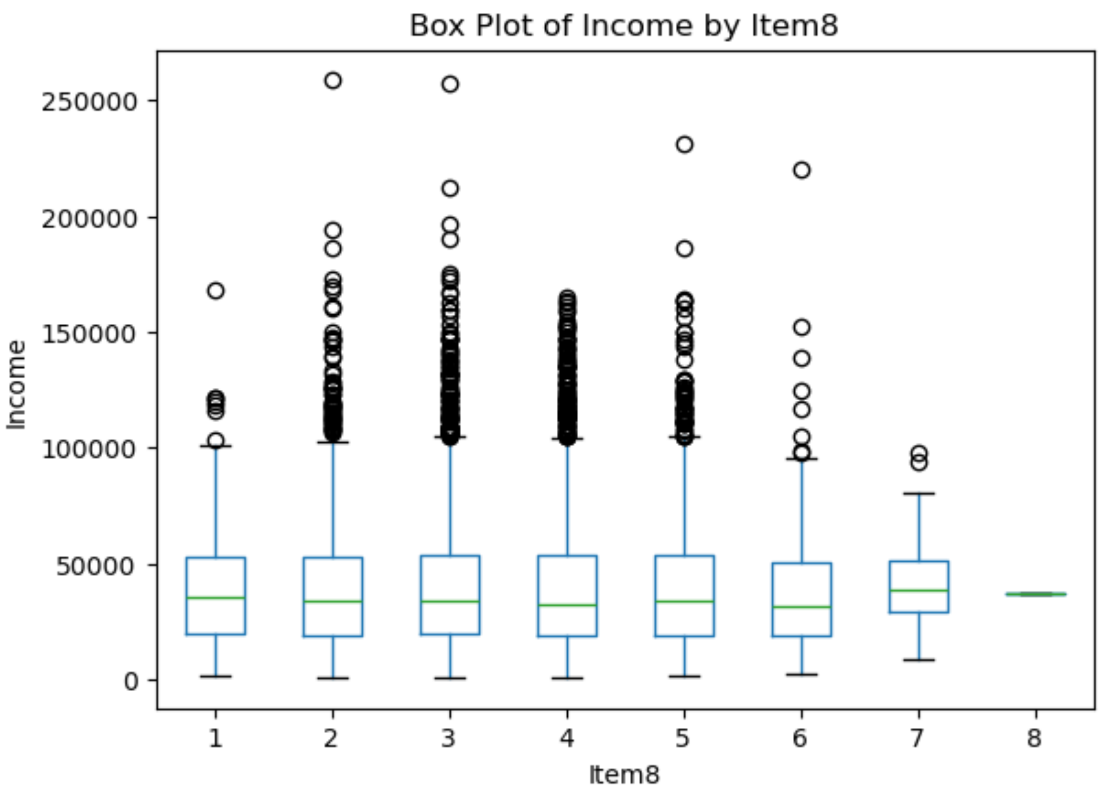

Box Plot of Income by Item4
Income
250000
200000
150000
100000
50000
0
1
2
3
4
5
6
7
Item4


Box Plot of Income by Item5
Income
250000
200000
150000
100000
50000
0
1
2
3
4
5
6
7
Item5


Box Plot of Income by Item6
Income
250000
200000
150000
100000
50000
0
1
2
3
4
5
6
7
8
Item6



Box Plot of Income by Item7
Income
250000
200000
150000
100000
50000
0
1
2
3
4
5
6
7
Item7



3.1.4 C4. Description of Data Transformation Goals and Steps to Achieve Goals

The data wrangling performed on the data set consisted of the following:

- Re-expression of the thirteen binary variables by encoding True and False as 1 and 0, respectively. This was done using a for loop that applies `.replace()` to specific columns using a dictionary. The thirteen re-expressed binary variables:
  - `Churn`, `Techie`, `Port_modem`, `Tablet`, `Phone`, `Multiple`, `OnlineSecurity`, `OnlineBackup`, `DeviceProtection`, `TechSupport`, `StreamingTV`, `StreamingMovies`, `PaperlessBilling`
- Six categorical variable were re-expressed using one-hot encoding with `.get_dummies()` from Pandas. The six variables re-expressed variables:
  - `Area`, `Marital`, `Gender`, `Contract`, `InternetService`, `PaymentMethod`

See code attached, in D208_PA_MendezD_Task1_Revision1.ipynb.

3.1.5 C5. Prepared Data Set as CSV file

[19]:
```
## C5 CSV Output

# Create explanatory variable data frame
expVars = pd.concat([numericVars, binaryVars, oneHotVars, ordinalVars], axis =
    1)
```

```
preparedData = pd.concat([df['Income'], expVars], axis = 1)

preparedData.to_csv('D208_PA_MendezD_Task1.csv', sep=',', encoding='utf-8',␣
↪index=False)
```

4 Part IV: Model Comparison and Analysis

4.1 D. Comparison of Initial and Reduced Linear Regression Models

4.1.1 D1. Initial Multiple Linear Regression Model

The following cells contain the code necessary to construct the initial MLR model with all 39 variables identified in C2. The summary of the initial model is printed below.

[20]:
```
## D1 Initial MLR Model with 39 independent variables

import statsmodels.api as sm

# Create explanatory variable data frame
expVars = pd.concat([numericVars, binaryVars, oneHotVars, ordinalVars], axis =␣
↪1)

# Add intercept to the model
expVars_intercept = sm.add_constant(expVars)

# Assign 'Income' as the dependent variable
depVar = df['Income']

# Fit the intercept model
model_intercept = sm.OLS(depVar, expVars_intercept).fit()

# Print the model summary
print(model_intercept.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 Income   R-squared:                       0.004
Model:                            OLS   Adj. R-squared:                 -0.000
Method:                 Least Squares   F-statistic:                    0.9029
Date:                Thu, 04 Jul 2024   Prob (F-statistic):              0.664
Time:                        11:03:22   Log-Likelihood:            -1.1664e+05
No. Observations:               10000   AIC:                         2.334e+05
Df Residuals:                    9951   BIC:                         2.337e+05
Df Model:                          48
Covariance Type:            nonrobust
================================================================================
=========================
```

```
                                              coef    std err          t
P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
-------------------------
const                                   4.278e+04   1.13e+04      3.783
0.000    2.06e+04    6.49e+04
Lat                                       24.8615     53.697      0.463
0.643     -80.396     130.119
Lng                                       -2.6922     18.815     -0.143
0.886     -39.574      34.190
Population                                -0.0135      0.020     -0.673
0.501      -0.053       0.026
Children                                 532.3464    996.776      0.534
0.593   -1421.536    2486.229
Age                                      -48.0417    106.087     -0.453
0.651    -255.993     159.910
Outage_sec_perweek                       -92.9173     95.029     -0.978
0.328    -279.194      93.360
Email                                    -83.7447     93.464     -0.896
0.370    -266.954      99.464
Contacts                                  56.3660    285.984      0.197
0.844    -504.220     616.952
Yearly_equip_failure                     248.3478    444.577      0.559
0.576    -623.112    1119.808
Tenure                                  1079.6633   2630.967      0.410
0.682   -4077.564    6236.891
MonthlyCharge                             70.2881     98.150      0.716
0.474    -122.107     262.683
Bandwidth_GB_Year                        -13.0926     32.111     -0.408
0.683     -76.037      49.852
Churn                                    601.5631    896.185      0.671
0.502   -1155.140    2358.267
Techie                                   272.5809    758.681      0.359
0.719   -1214.588    1759.750
Port_modem                              -798.2235    565.175     -1.412
0.158   -1906.080     309.633
Tablet                                   406.6171    618.026      0.658
0.511    -804.839    1618.074
Phone                                   -208.2915    972.793     -0.214
0.830   -2115.162    1698.579
Multiple                               -1397.6905   1385.296     -1.009
0.313   -4113.150    1317.769
OnlineSecurity                           245.1205   2266.890      0.108
0.914   -4198.443    4688.683
OnlineBackup                            -780.7626   1312.695     -0.595
0.552   -3353.910    1792.385
DeviceProtection                        1104.6897   1718.823      0.643
0.520   -2264.551    4473.931
```

```
TechSupport                              -318.2031   1237.532     -0.257
0.797   -2744.016    2107.610
StreamingTV                              -251.4161   3718.636     -0.068
0.946   -7540.696    7037.864
StreamingMovies                         -1180.5633   2600.316     -0.454
0.650   -6277.709    3916.582
PaperlessBilling                         -722.2859    574.760     -1.257
0.209   -1848.932     404.360
Area_Suburban                             254.6622    691.563      0.368
0.713   -1100.941    1610.266
Area_Urban                                168.5279    692.898      0.243
0.808   -1189.692    1526.748
Marital_Married                           563.8812    894.294      0.631
0.528   -1189.117    2316.879
Marital_Never Married                     324.6850    889.006      0.365
0.715   -1417.946    2067.317
Marital_Separated                        -577.5546    882.012     -0.655
0.513   -2306.477    1151.368
Marital_Widowed                           -35.7687    881.972     -0.041
0.968   -1764.612    1693.074
Gender_Male                              -479.3286   2162.096     -0.222
0.825   -4717.474    3758.817
Gender_Nonbinary                          351.1810   2022.443      0.174
0.862   -3613.216    4315.578
Contract_One year                         137.5288    755.737      0.182
0.856   -1343.868    1618.926
Contract_Two Year                         -56.8375    720.707     -0.079
0.937   -1469.568    1355.893
InternetService_Fiber Optic             -7969.6806   1.52e+04     -0.525
0.599   -3.77e+04    2.18e+04
InternetService_None                    -5241.8774   1.21e+04     -0.432
0.666    -2.9e+04    1.85e+04
PaymentMethod_Credit Card (automatic)     431.0365    861.572      0.500
0.617   -1257.818    2119.891
PaymentMethod_Electronic Check            829.1623    770.939      1.076
0.282    -682.035    2340.360
PaymentMethod_Mailed Check               1432.1044    841.615      1.702
0.089    -217.631    3081.840
Item1                                    -600.8701    405.091     -1.483
0.138   -1394.929     193.189
Item2                                     409.1144    379.412      1.078
0.281    -334.610    1152.839
Item3                                    -200.6647    348.049     -0.577
0.564    -882.910     481.581
Item4                                    -733.3633    311.053     -2.358
0.018   -1343.089    -123.637
Item5                                     209.1172    322.945      0.648
0.517    -423.919     842.154
```

```
Item6                                  146.8736    332.597      0.442
0.659    -505.084     798.832
Item7                                  711.3222    314.478      2.262
0.024      94.881    1327.763
Item8                                   64.5028    299.469      0.215
0.829    -522.517     651.523
==============================================================================
Omnibus:                     2774.122   Durbin-Watson:                   1.983
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             7994.598
Skew:                           1.455   Prob(JB):                         0.00
Kurtosis:                       6.274   Cond. No.                     1.42e+06
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 1.42e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
```

4.1.2 D2. Model Reduction Method and Justification

The model reduction method utilized below is backward stepwise elimination. Beginning with all explanatory variables, backwards elimination removes the least significant variable, as determined by p-value, at each iteration of the loop. This is iterated until no improvement is observed, or rather, no variables are observed to have $p > 0.05$ (Middleton, 2022).

[21]:
```
# D2 Model Reduction Method

def backward_elimination(data, target, siglevel):
# Function sourced from AnalyticsVidhya to perform Backwards Elimination

    features = data.columns.tolist()
    while(len(features) > 0):
        features_with_constant = sm.add_constant(data[features])
        p_values = sm.OLS(target, features_with_constant).fit().pvalues[1:]
        max_p_value = p_values.max()
        if(max_p_value >= siglevel):
            excluded_feature = p_values.idxmax()
            features.remove(excluded_feature)
        else:
            break
    return features
```

[22]:
```
# D2 Model Reduction Method

backward_elimination(expVars, depVar, 0.05)
```

[22]: ['Item4', 'Item7']

4.1.3 D3. Reduced Model

Upon applying Backwards Elimination, the explanatory variables of the reduced model are:

- Item4
- Item7

Below is the code that constructs the reduced model with intercept and prints its summary.

[23]:
```
# D3 Reduced Model

expVarsReduced = df[['Item4', 'Item7']]

expVarsReduced_intercept = sm.add_constant(expVarsReduced)

model_reduced = sm.OLS(depVar, expVarsReduced_intercept).fit()

print(model_reduced.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 Income   R-squared:                       0.001
Model:                            OLS   Adj. R-squared:                  0.001
Method:                 Least Squares   F-statistic:                     4.895
Date:                Thu, 04 Jul 2024   Prob (F-statistic):            0.00750
Time:                        11:03:26   Log-Likelihood:            -1.1665e+05
No. Observations:               10000   AIC:                         2.333e+05
Df Residuals:                    9997   BIC:                         2.333e+05
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        4.03e+04   1275.080     31.606      0.000    3.78e+04    4.28e+04
Item4       -743.2030    280.231     -2.652      0.008   -1292.513    -193.893
Item7        600.2211    279.500      2.147      0.032      52.346    1148.097
==============================================================================
Omnibus:                     2791.309   Durbin-Watson:                   1.983
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             8071.369
Skew:                           1.463   Prob(JB):                         0.00
Kurtosis:                       6.288   Cond. No.                         23.8
==============================================================================

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

4.2 E. Analysis Using Reduced Linear Regression Model

4.2.1 E1. Model Comparison using Model Evaluation Metrics

The initial model can be compared with the reduced model using AIC and the F statistic. The following comparisons can be drawn using the output of the code below.

- Adjusted R^2: Although very low, it increased from the initial model to the reduced model, so the variables add explanatory power to the regression (Middleton, 2022).
- AIC: Since the reduced model has a lower AIC, it has a better fit relative to the initial model.

[24]:
```
## E1 Model Comparison

def modelCompare(initialModel, reducedModel):
# Prints a data frame as a table that compares model evaluation metrics for two␣
↪regression models

    evalMetrics = {
        'Criteria': ['Adj R2', 'AIC', 'F stat', 'Prob (F-stat)'],
        'Initial': [initialModel.rsquared_adj, initialModel.aic, initialModel.
↪fvalue, initialModel.f_pvalue],
        'Reduced': [reducedModel.rsquared_adj, reducedModel.aic, reducedModel.
↪fvalue, reducedModel.f_pvalue]
    }

    em = pd.DataFrame(evalMetrics)
    print(em)
```

[25]:
```
## E1 Model Comparison

modelCompare(model_intercept, model_reduced)
```

```
        Criteria        Initial        Reduced
0         Adj R2      -0.000466       0.000779
1            AIC  233373.797638  233315.467874
2         F stat       0.902921       4.895328
3  Prob (F-stat)       0.663710       0.007499
```

4.2.2 E2. Residual Plots and Model's Residual Standard Error

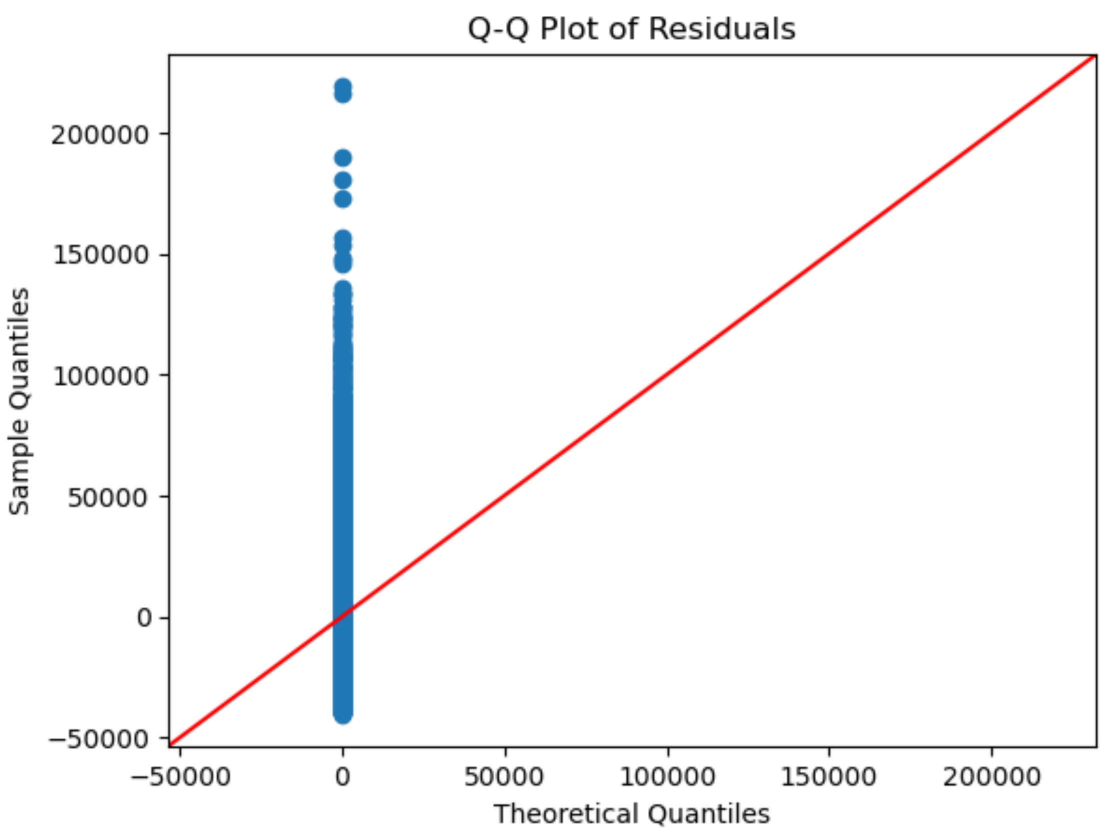
Below is the code that generate the Q-Q plot, the histogram of the model's residuals, as well as the model's residual standard error.

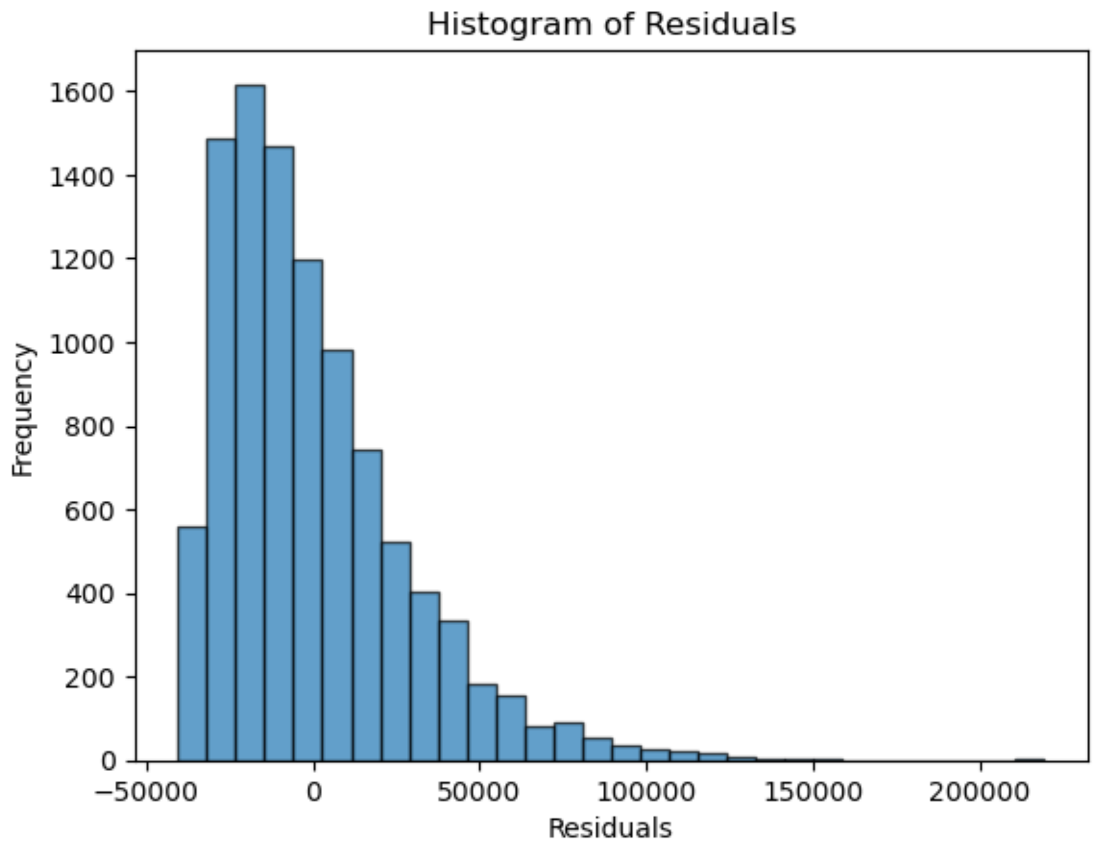
The residual standard error is shown below to be 28187.53.

[26]:
```
## E2 Residual Plots and RSE

import numpy as np

def residualPlots(model):
```

```
    # Takes the model and returns the Q-Q plot, the histogram of the residuals,␣
↪and the RSE.

    residuals = model.resid

    sm.qqplot(residuals, line = '45')
    plt.title('Q-Q Plot of Residuals')
    plt.show()

    plt.hist(residuals, bins = 30, edgecolor = 'k', alpha = 0.7)
    plt.title('Histogram of Residuals')
    plt.xlabel('Residuals')
    plt.ylabel('Frequency')
    plt.show()

    rse = np.sqrt(np.sum(residuals**2) / (len(residuals) - 2))
    print(f'The RSE is {rse}')
```

[27]:
```
## E2 Residual Plots and RSE

residualPlots(model_reduced)
```






```
The RSE is 28187.527461290818
```

4.2.3 E3. Code

See code attached, in D208_PA_MendezD_Task1_Revision1.ipynb

5 Part V: Data Summary and Implications

5.1 F. Summary of Findings

5.1.1 F1. Results of Data Analysis

- Regression Equation for the Reduced Model

$$Income = 40299.8 - 743.2(Item4) + 600.22(Item7)$$

- Interpretation of the Coefficients of the Reduced Model
  - All else constant, a customer's response to `Item4` is associated with an average decrease of \$743.20 in income. The higher the value of the survey response, the more income will decrease.

- All else constant, a customer's response to `Item7` is associated with an average increase of $600.22 in income. The higher the value of the survey response, the more income will increase.
- All else constant, the model assumes a customer begins with $40299.80 of income.

- Statistical and Practical Significance of the Reduced Model
  - As shown in E1, the F statistic is small, but with a p-value of $p = 0.007499$, at $\alpha = 0.05$, there is evidence to conclude that the model is statistically significant. However, the model may lack practical significance, because it is unlikely that two ordinal variables alone will be accurate predictors of a customer's income.
- Limitations of the Data Analysis
  - A major limitation of this data analysis is the assumption that any of these explanatory variables have a causal relationship between them and the dependent variable. Another issue is that the residuals do not appear to be normally distributed, as they are skewed right. Additionally, with only two ordinal explanatory variables, the model likely has poor predictive performance.

[28]:
```python
## F1 Regression Equation Coefficients

coefficients = model_reduced.params
print('Coefficients:')
print(coefficients)
```

```
Coefficients:
const    40299.803319
Item4     -743.203011
Item7      600.221109
dtype: float64
```

5.1.2 F2. Course of Action

Since the model has much room for improvement, my recommended course of action would be to seek alternative customer information that might be associated with income.

6 Part VI: Demonstration

6.1 G. Panopto Video

https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=c37f6c20-35a4-4b1f-b593-b1a1001680f1

6.2 H. Acknowledgement of Web Sources

pandas.get_dummies — pandas 2.2.2 documentation. (n.d.). https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.get_dummies.html

Getting started - statsmodels 0.15.0 (+73). (n.d.). https://www.statsmodels.org/devel/gettingstarted.html

Verma, V. (2020, October 24). Feature Selection using Wrapper Method - Python Implementation. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/10/a-comprehensive-guide-to-feature-selection-using-wrapper-methods-in-python/

6.3 I. Acknowledgement of Sources

Karir, R. (2022, March 16). 4 main assumptions in Multi-Linear Regression. LinkedIn. https://www.linkedin.com/pulse/4-main-assumptions-multi-linear-regression-ritik-karir/

Middleton, K. (2022, November). D208 - Webinar Getting Started with D208 Part I. WGU. https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=15e09c73-c5aa-439d-852f-af47001b8970

Middleton, K. (2022, November). D208 - Webinar Getting Started with D208 Part II. WGU. https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=39bbe2db-de7d-4bf5-913b-af5c0003da9d