Statistical Data Mining

Task 1

Nathan Hefner

B. Describe the purpose of this data analysis by doing the following:

1. Propose one research question that is relevant to a real-world organizational situation captured in the provided dataset that you will answer using multiple linear regression in the initial model.

For multiple linear regression, we need one dependent variable and two or more independent variables. Our dependent variable will be the price of the home, and our six independent variables, the values that will help us predict the cost of the house, will be Square Footage, Backyard Space, Number of Bedrooms, Floors, Age Of Home, and Crime Rate. Therefore, our research question will be: How much does the property size, backyard space, number of bathrooms, number of floors, age of home, and the local crime rate affect the total house price?

2. Define one goal of the data analysis. Ensure that your goal is reasonable within the scope of the scenario and is represented in the available data.

This data analysis will aim to discover how these six different variables affect the price. By doing so, we can maximize return on investment by finding the best size for construction and area to build. This will help keep construction costs as low as possible to get the best possible returns. This will also help us find the point of diminishing returns as the more significant the house size or backyard space may not give the best possible investment returns.

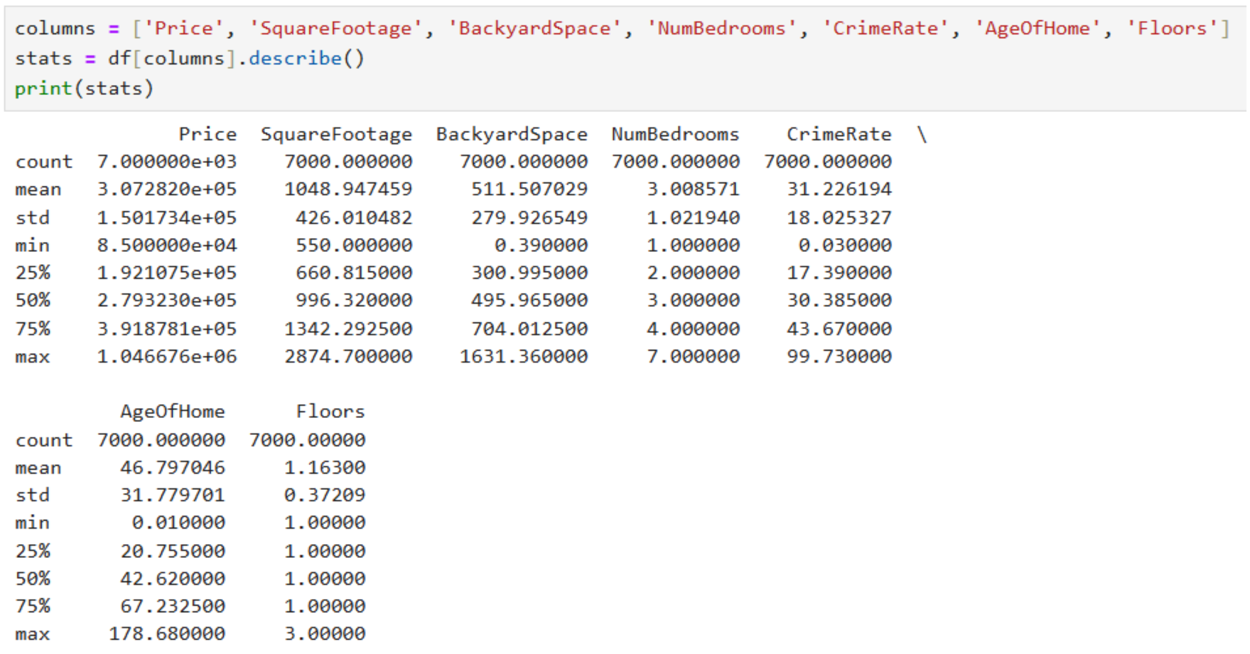
C. Summarize the data preparation process for multiple linear regression analysis by doing the following:

1. Identify the dependent and all independent variables that are required to answer the research question and justify your selection of variables.

The dependent variable is the price of the property because this is the variable that will change because of the other variables, therefore being dependent. Our six independent variables will be SquareFootage, Backyard Space, Number of Bedrooms, Floors, Age of Home, and Crime Rate because I believe the larger the property size and the lower the crime rate, the bigger the price will be. Therefore, being the independent variables.

2. Describe the dependent variable and all independent variables from part C1 using descriptive statistics (counts, means, modes, ranges, min/max), including a screenshot of the descriptive statistics output for each of these variables.

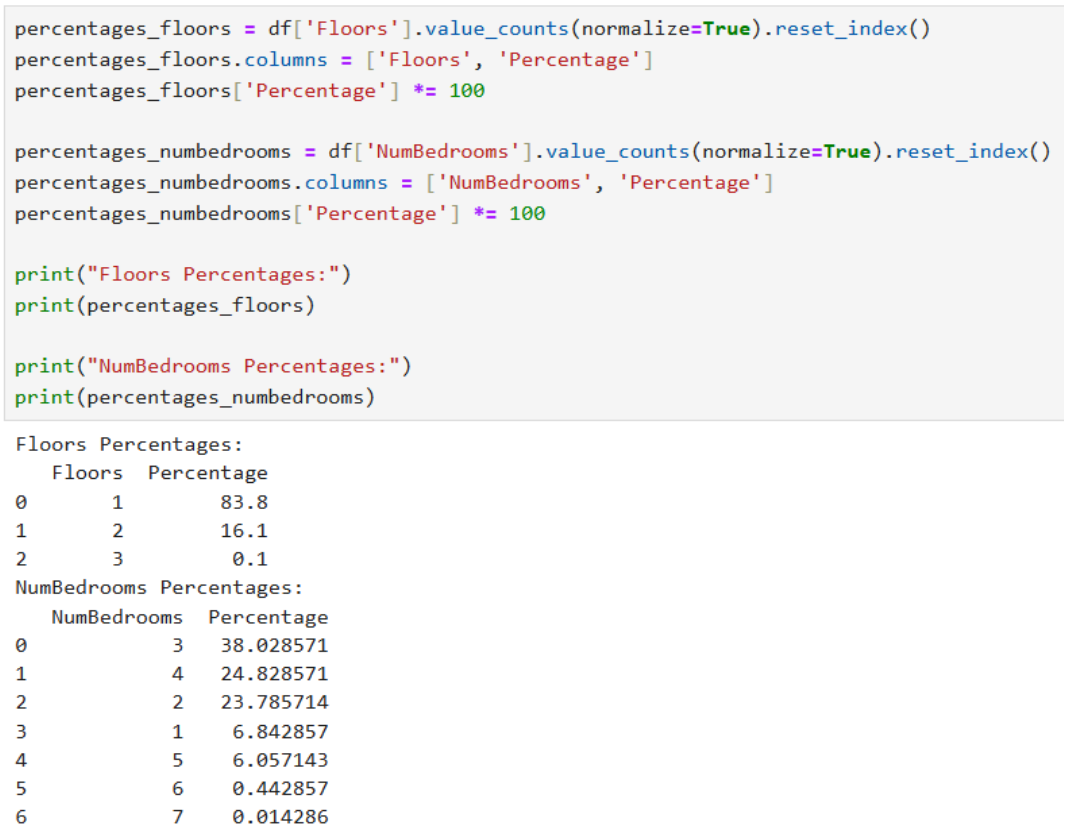
Below is the code for the descriptive statistics for Price, Square Footage, Backyard Space, Number of Bedrooms, Crime rate, and Age of Home.



```
columns = ['Price', 'SquareFootage', 'BackyardSpace', 'NumBedrooms', 'CrimeRate', 'AgeOfHome', 'Floors']
stats = df[columns].describe()
print(stats)
```

```
              Price  SquareFootage  BackyardSpace  NumBedrooms    CrimeRate  \
count  7.000000e+03    7000.000000    7000.000000  7000.000000  7000.000000
mean   3.072820e+05    1048.947459     511.507029     3.008571    31.226194
std    1.501734e+05     426.010482     279.926549     1.021940    18.025327
min    8.500000e+04     550.000000       0.390000     1.000000     0.030000
25%    1.921075e+05     660.815000     300.995000     2.000000    17.390000
50%    2.793230e+05     996.320000     495.965000     3.000000    30.385000
75%    3.918781e+05    1342.292500     704.012500     4.000000    43.670000
max    1.046676e+06    2874.700000    1631.360000     7.000000    99.730000

         AgeOfHome      Floors
count  7000.000000  7000.00000
mean     46.797046     1.16300
std      31.779701     0.37209
min       0.010000     1.00000
25%      20.755000     1.00000
50%      42.620000     1.00000
75%      67.232500     1.00000
max     178.680000     3.00000
```

Because the number of floors and Number of Bedrooms have limited unique values I decided to add the percentages of each to understand them better.

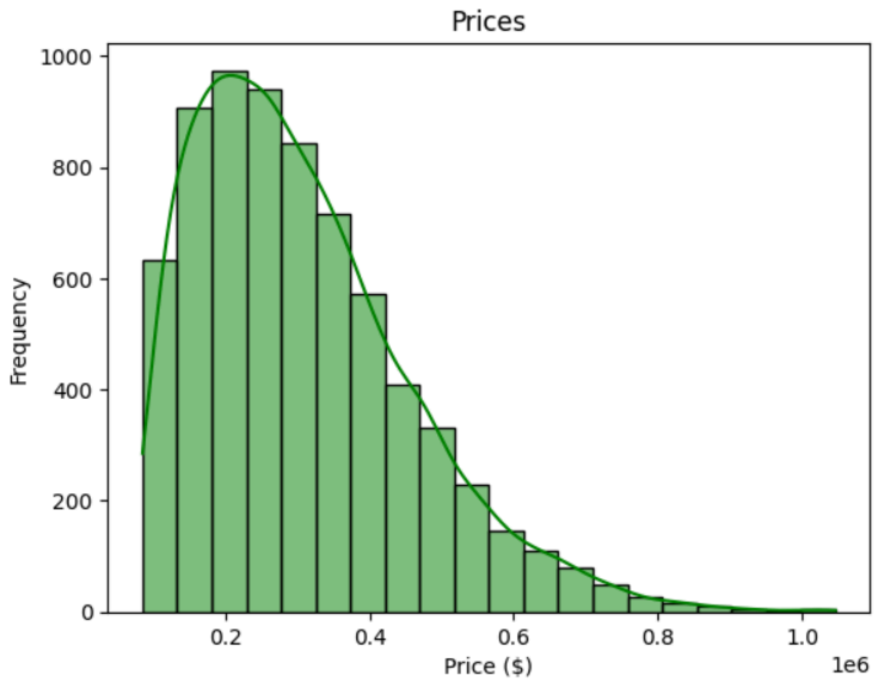


```
percentages_floors = df['Floors'].value_counts(normalize=True).reset_index()
percentages_floors.columns = ['Floors', 'Percentage']
percentages_floors['Percentage'] *= 100

percentages_numbedrooms = df['NumBedrooms'].value_counts(normalize=True).reset_index()
percentages_numbedrooms.columns = ['NumBedrooms', 'Percentage']
percentages_numbedrooms['Percentage'] *= 100

print("Floors Percentages:")
print(percentages_floors)

print("NumBedrooms Percentages:")
print(percentages_numbedrooms)
```

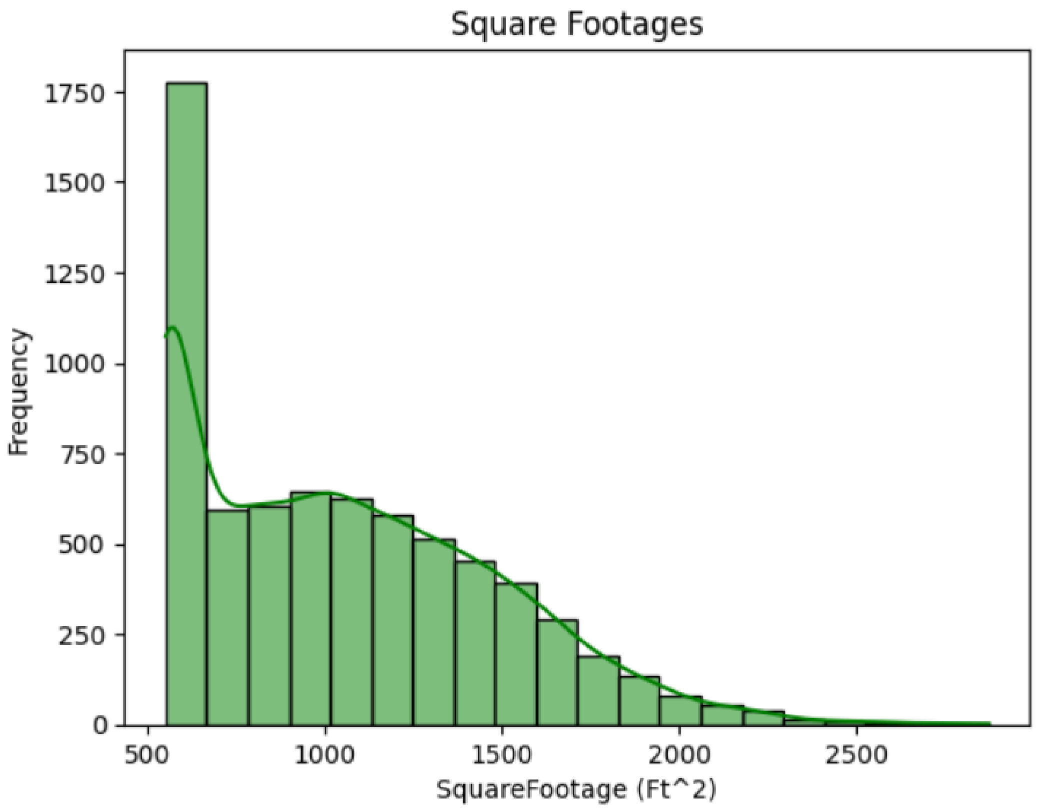
```
Floors Percentages:
   Floors  Percentage
0       1        83.8
1       2        16.1
2       3         0.1
NumBedrooms Percentages:
   NumBedrooms  Percentage
0            3   38.028571
1            4   24.828571
2            2   23.785714
3            1    6.842857
4            5    6.057143
5            6    0.442857
6            7    0.014286
```

3. Generate univariate and bivariate visualizations of the distributions of the dependent and independent variables from part C1, including the dependent variable in the bivariate visualizations.

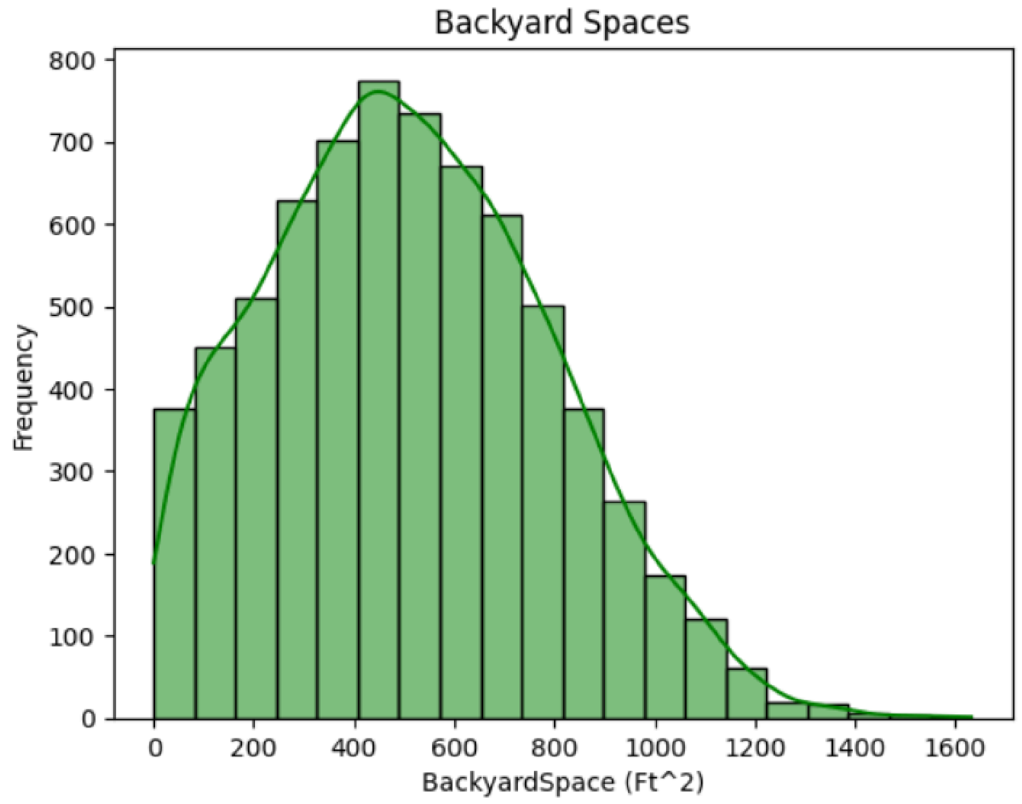
Below is my univariate visualization for the price variable.



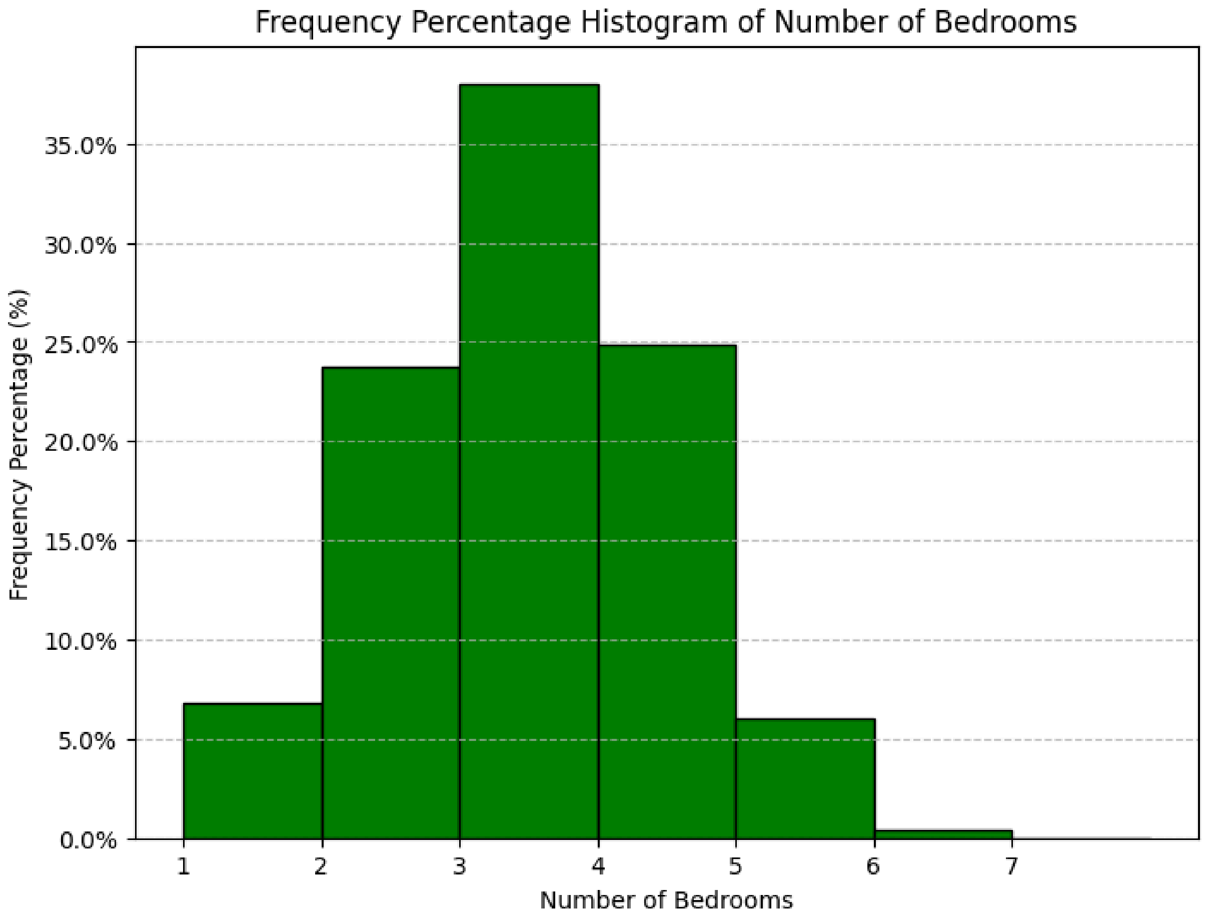
Below is my univariate visualization for the square footage variable.



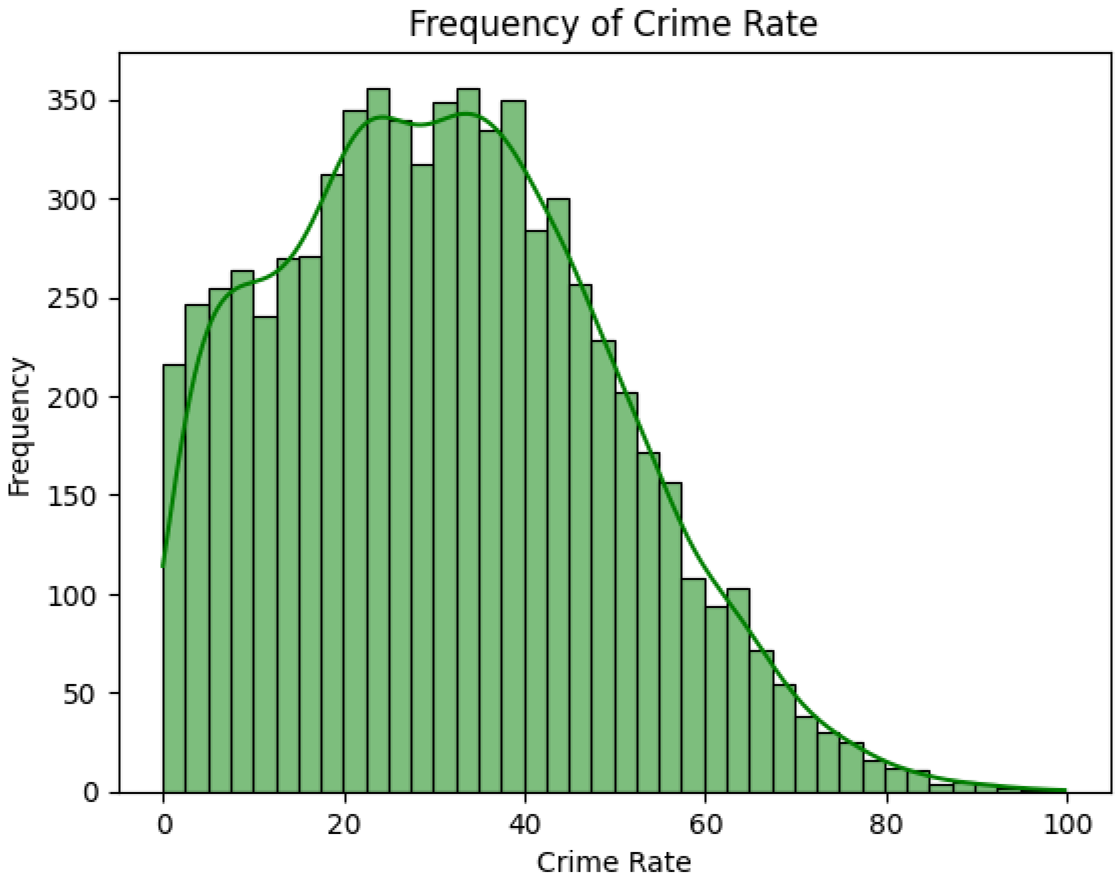
Below is my univariate visualization for the backyard space variable.



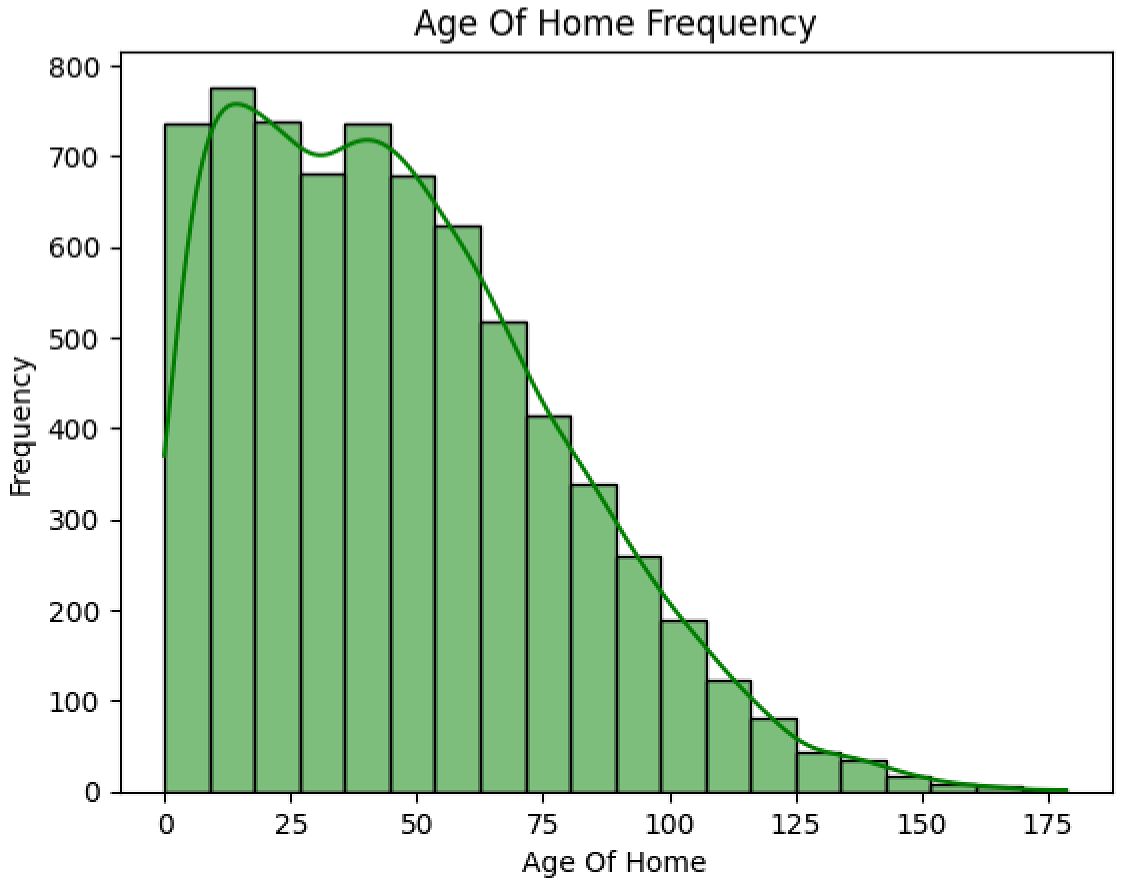
Below is my univariate visualization for the Number of Bedrooms variable.



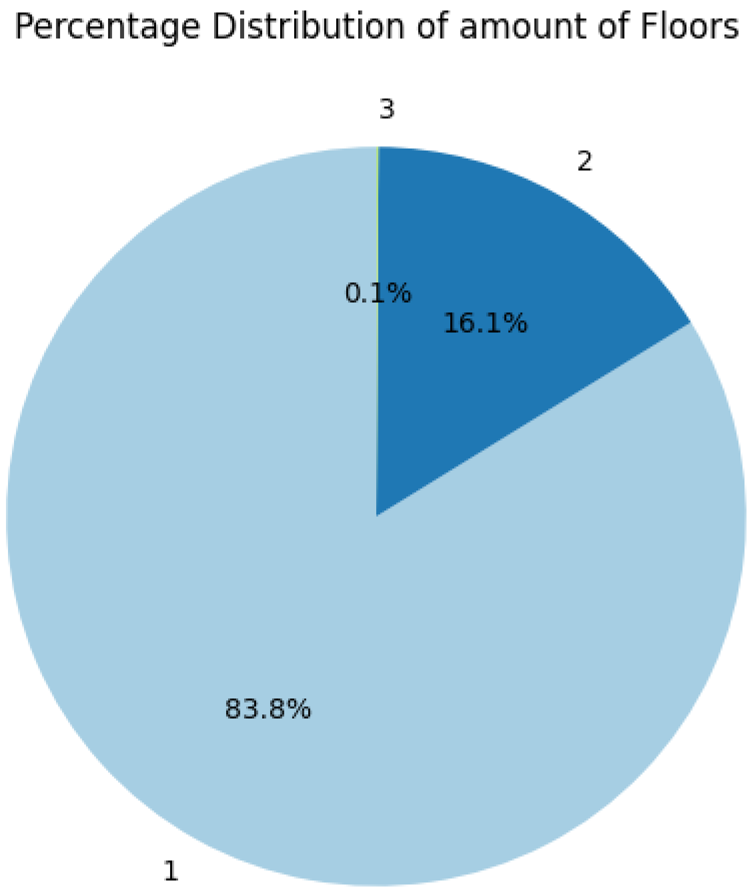
Below is my univariate visualization for the Crime Rate variable.



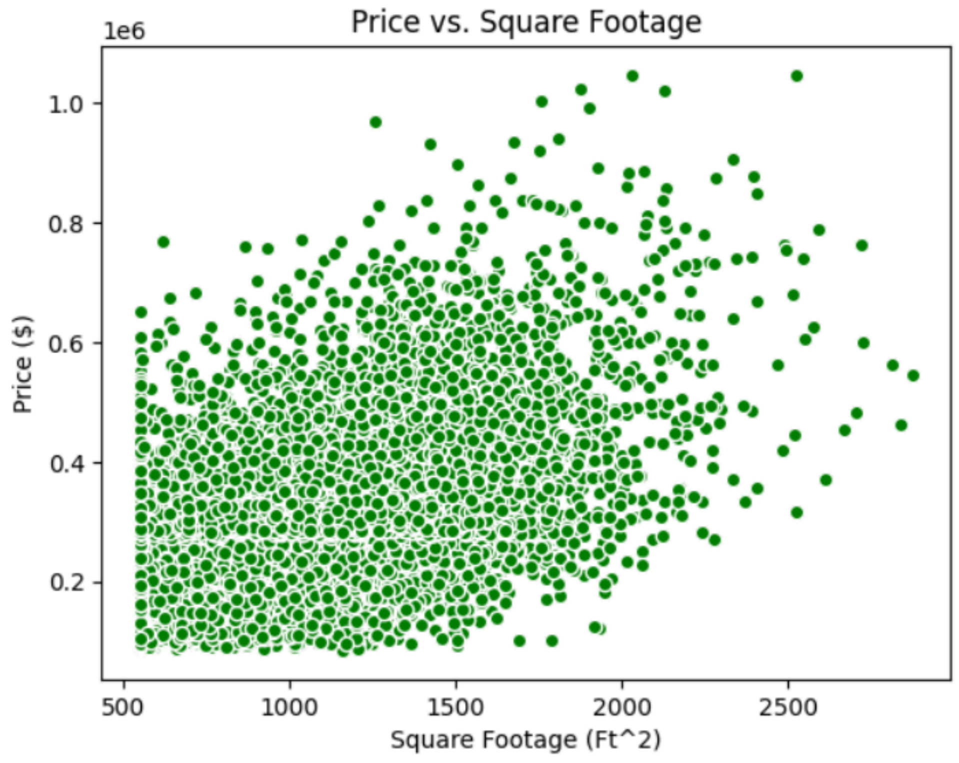
Below is my univariate visualization for the Age of Home variable.



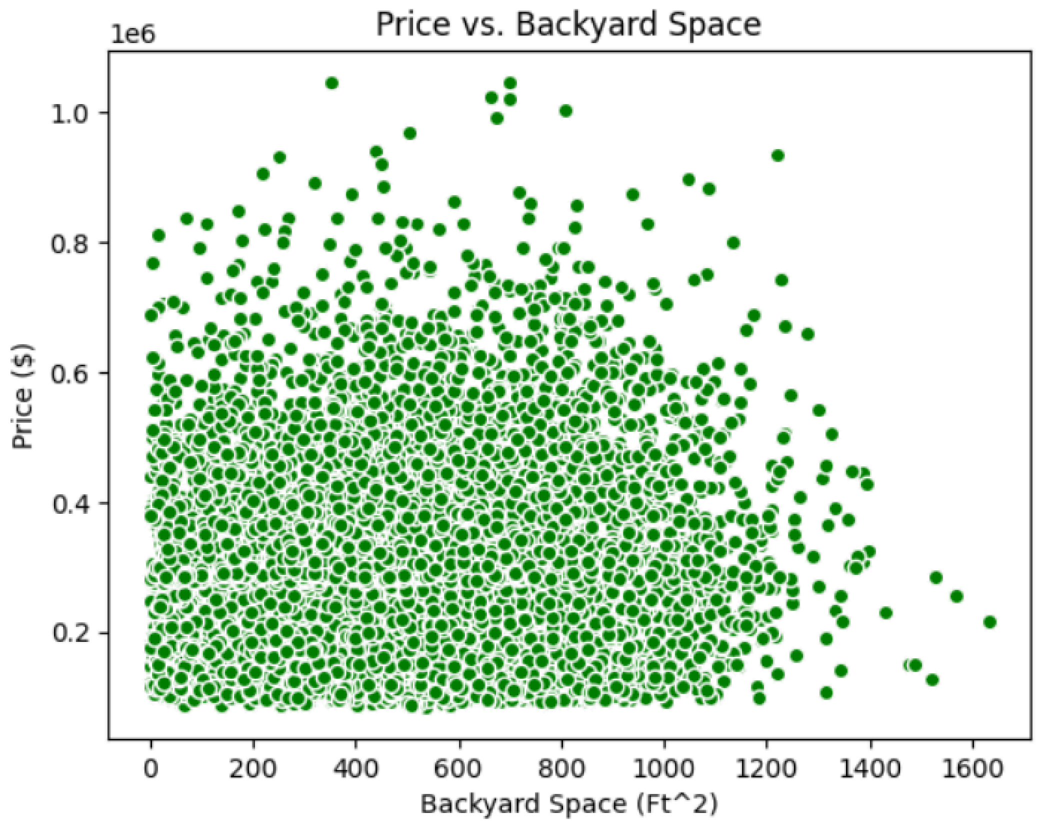
Below is my univariate visualization for the Floors variable.



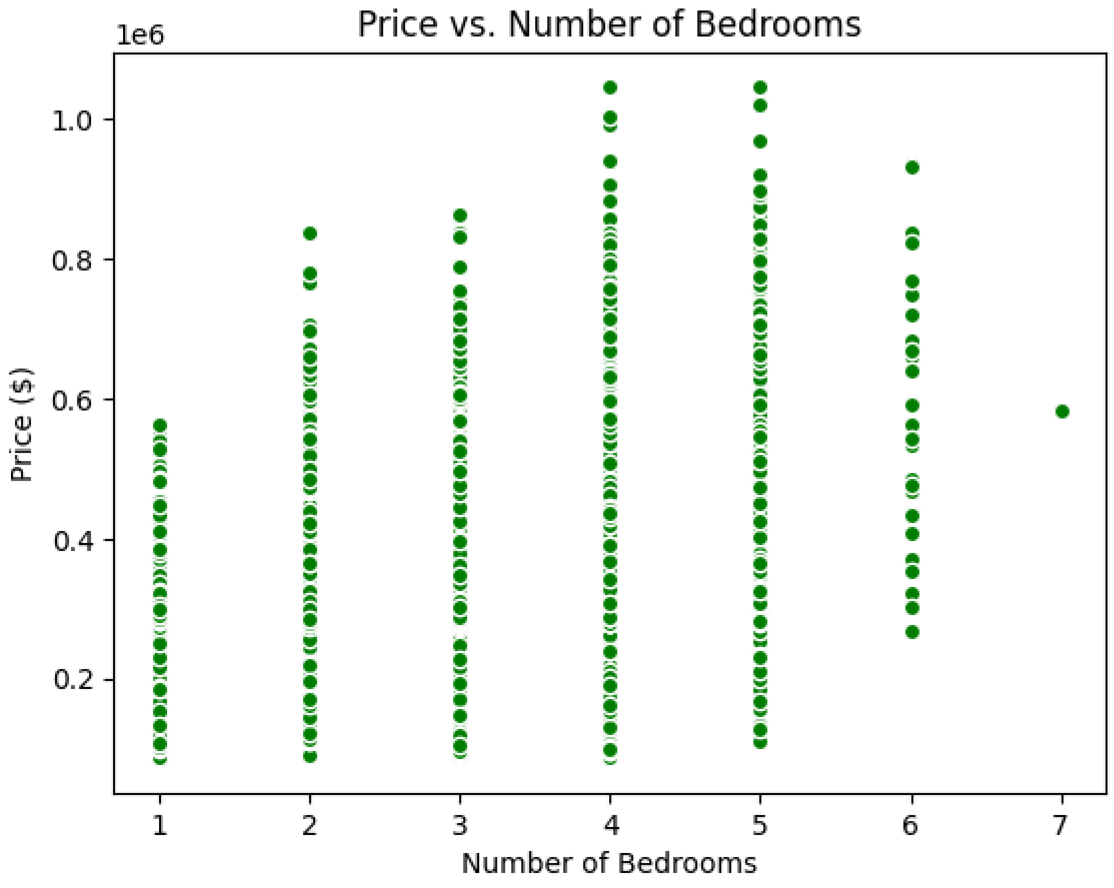
Below is my bivariate graph for Price versus Square Footage.



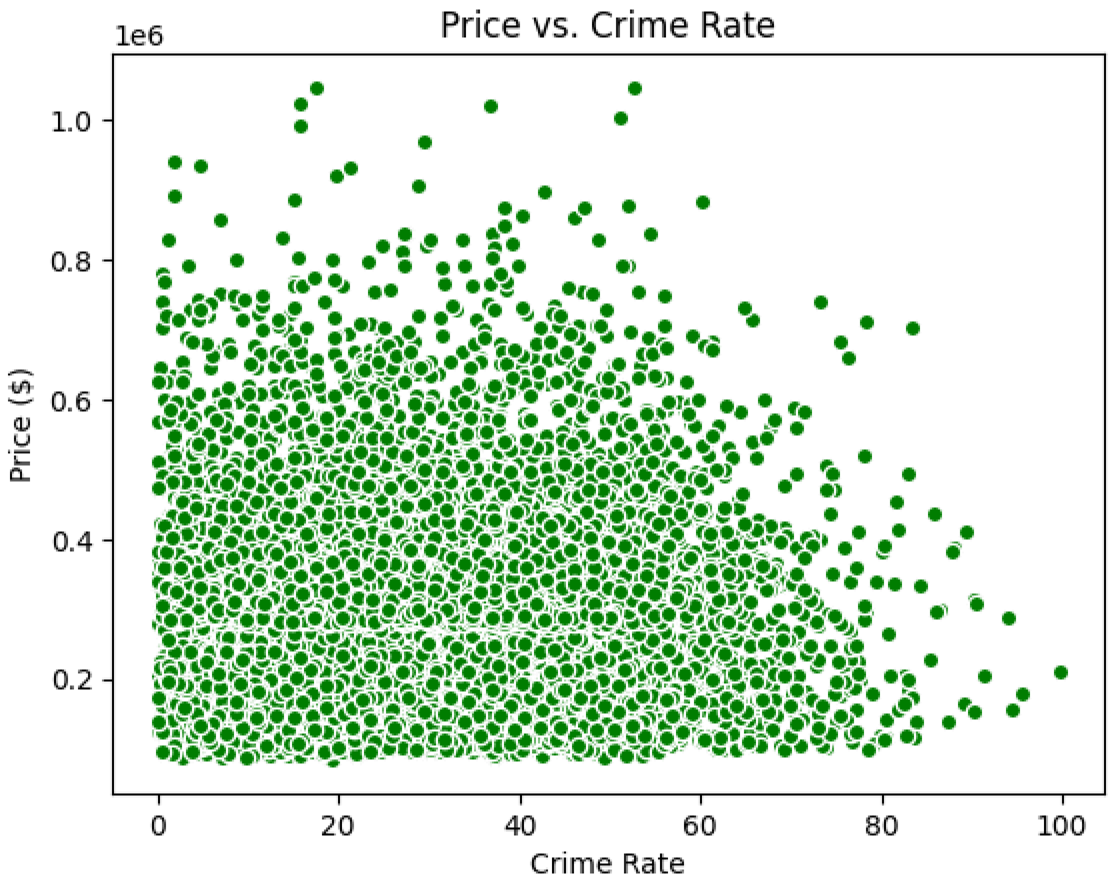
Below is my bivariate graph for Price versus Backyard Space.



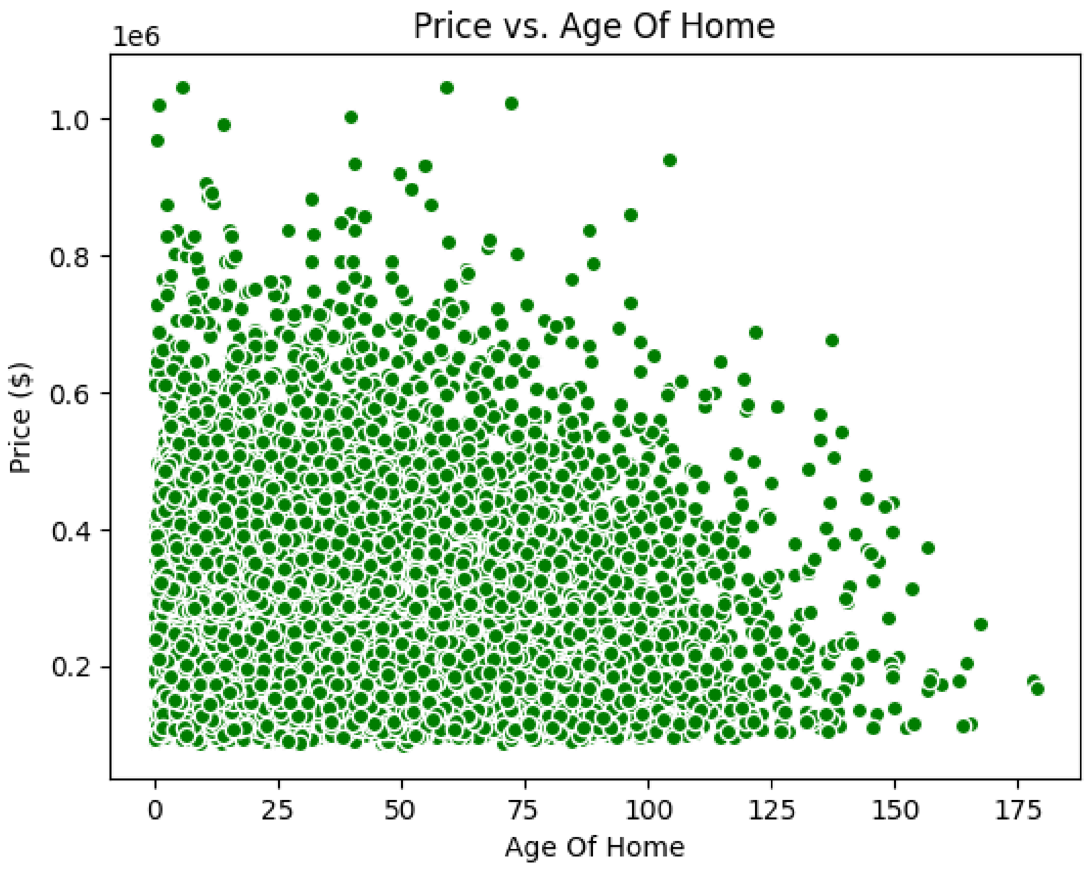
Below is my bivariate graph for Price versus Number of Bedrooms.



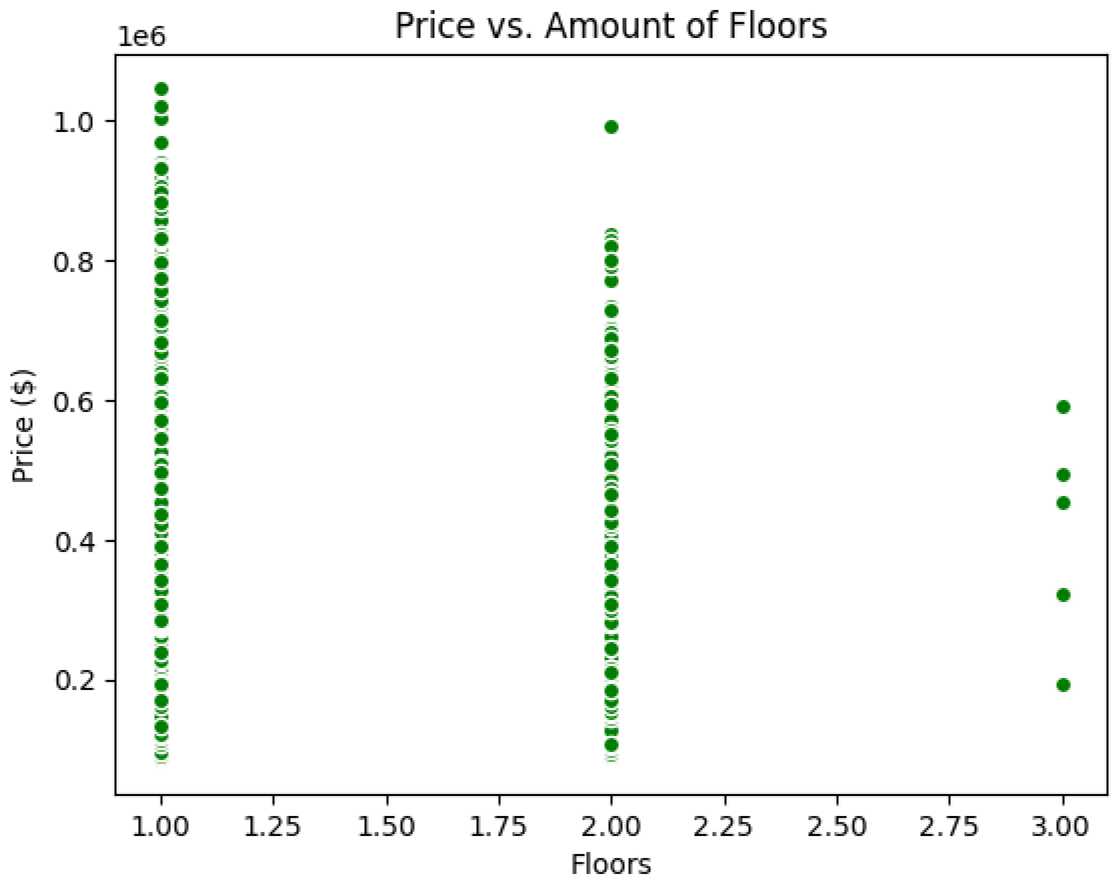
Below is my bivariate graph for Price versus Crime Rate.



Below is my bivariate graph for Price versus Age of Home.



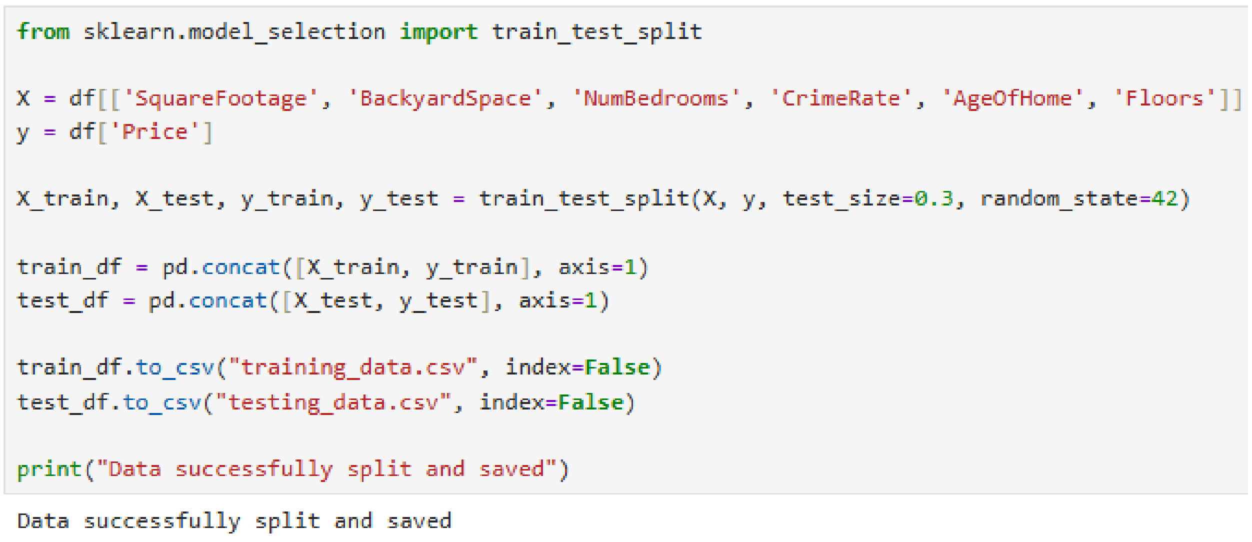
Below is my bivariate graph for Price versus Floors.



D. Perform the data analysis and report on the results by doing the following:

1. Split the data into two datasets, with a larger percentage assigned to the training dataset and a smaller percentage assigned to the test data set. Provide the files.

Below is the code I used to split the data. I did a 70/30 split with the larger portion assigned to the training dataset.

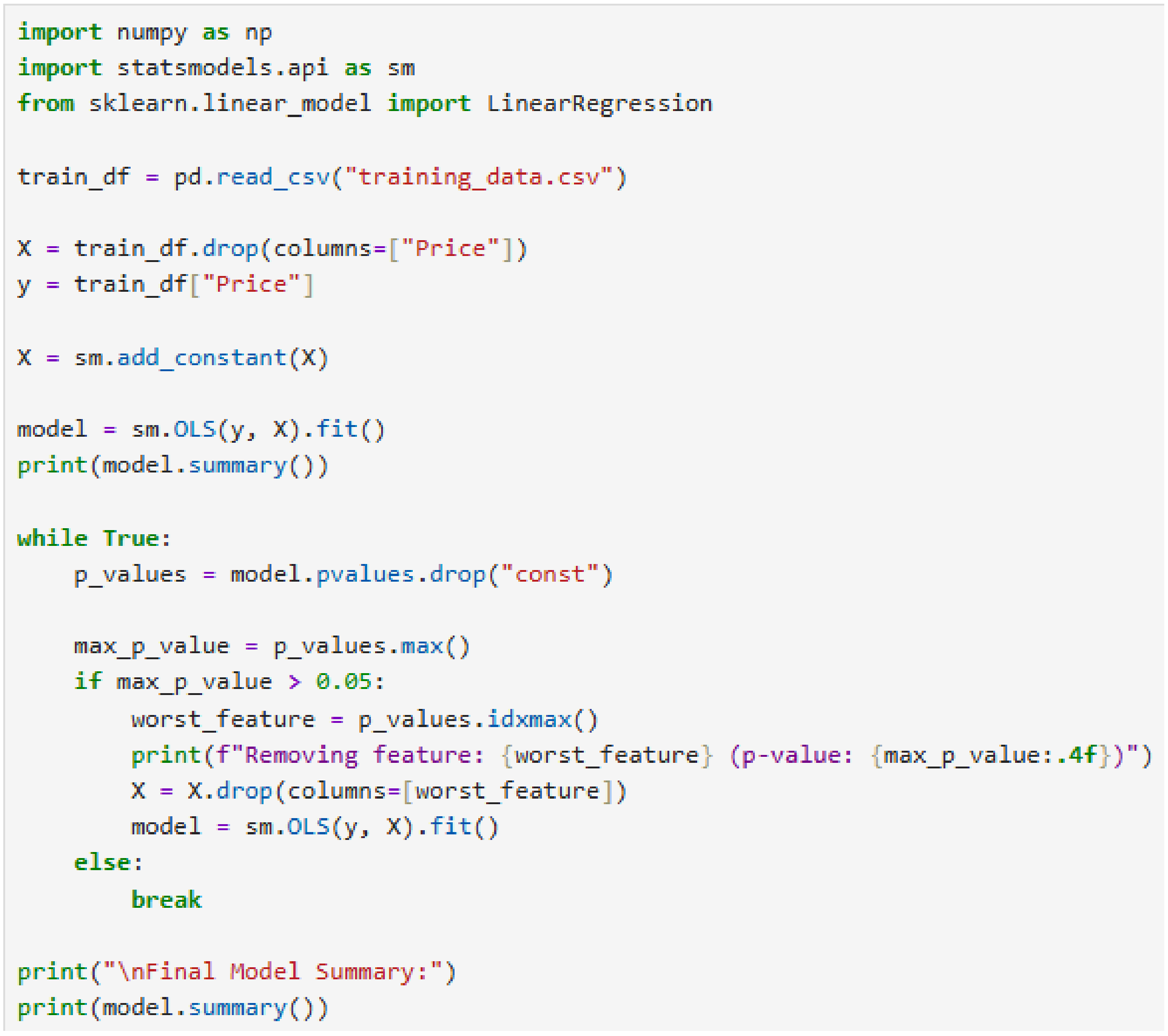


```
from sklearn.model_selection import train_test_split

X = df[['SquareFootage', 'BackyardSpace', 'NumBedrooms', 'CrimeRate', 'AgeOfHome', 'Floors']]
y = df['Price']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

train_df = pd.concat([X_train, y_train], axis=1)
test_df = pd.concat([X_test, y_test], axis=1)

train_df.to_csv("training_data.csv", index=False)
test_df.to_csv("testing_data.csv", index=False)

print("Data successfully split and saved")
```

```
Data successfully split and saved
```

2. Use the training dataset to create and perform a regression model using regression as a statistical method. Optimize the regression model using a process of your selection, including but not limited to, forward stepwise selection, backward stepwise elimination, and recursive selection. Provide a screenshot of the summary of the optimized model.

Below is my code for performing a regression model using regression as a statistical method along with optimizing the model using Backward Stepwise Elimination to remove the least significant features from the dataset.



```python
import numpy as np
import statsmodels.api as sm
from sklearn.linear_model import LinearRegression

train_df = pd.read_csv("training_data.csv")

X = train_df.drop(columns=["Price"])
y = train_df["Price"]

X = sm.add_constant(X)

model = sm.OLS(y, X).fit()
print(model.summary())

while True:
    p_values = model.pvalues.drop("const")

    max_p_value = p_values.max()
    if max_p_value > 0.05:
        worst_feature = p_values.idxmax()
        print(f"Removing feature: {worst_feature} (p-value: {max_p_value:.4f})")
        X = X.drop(columns=[worst_feature])
        model = sm.OLS(y, X).fit()
    else:
        break

print("\nFinal Model Summary:")
print(model.summary())
```

```
Final Model Summary:
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Price   R-squared:                       0.472
Model:                            OLS   Adj. R-squared:                  0.472
Method:                 Least Squares   F-statistic:                     1095.
Date:                Sun, 23 Mar 2025   Prob (F-statistic):               0.00
Time:                        14:03:54   Log-Likelihood:                -63822.
No. Observations:                4900   AIC:                         1.277e+05
Df Residuals:                    4895   BIC:                         1.277e+05
Df Model:                           4
Covariance Type:            nonrobust
=================================================================================
                    coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------
const         -3.911e+04   7291.307     -5.363      0.000   -5.34e+04   -2.48e+04
SquareFootage   176.4240      3.711     47.534      0.000     169.148     183.700
NumBedrooms    6.081e+04   1540.421     39.475      0.000    5.78e+04    6.38e+04
CrimeRate      -248.0556     87.508     -2.835      0.005    -419.611     -76.500
AgeOfHome      -272.7402     50.181     -5.435      0.000    -371.117    -174.363
==============================================================================
Omnibus:                      416.016   Durbin-Watson:                   1.991
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              549.266
Skew:                           0.727   Prob(JB):                    5.35e-120
Kurtosis:                       3.760   Cond. No.                     5.33e+03
==============================================================================
```

3. Give the mean squared error (MSE) of the optimized model used on the training set.

The equation for Mean Squared Error is as follows:

$$\text{MSE} = \frac{1}{n}\sum(y_{\text{actual}} - y_{\text{predicted}})^2$$

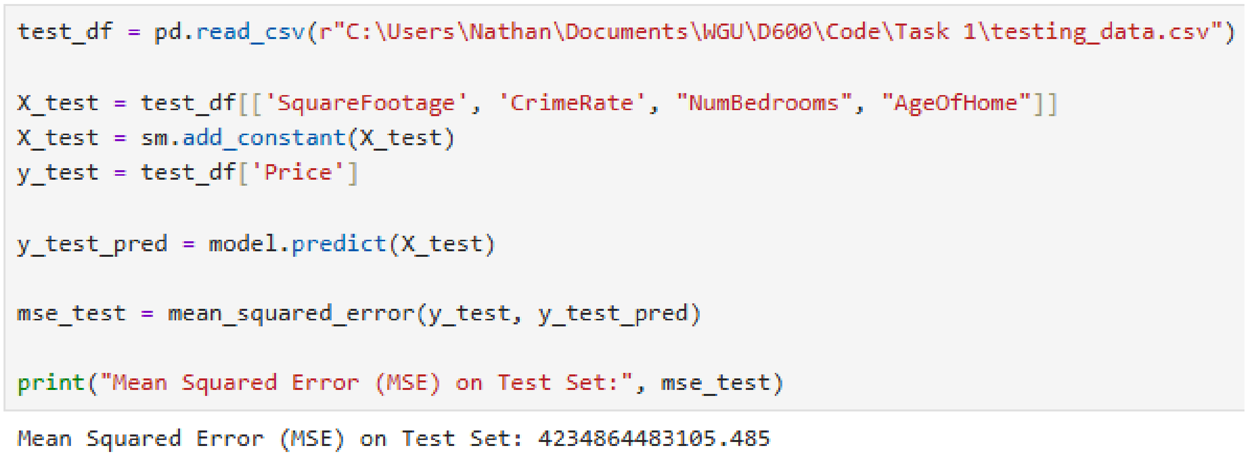
Below is my code to calculate this value using the optimized model.

```
from sklearn.metrics import mean_squared_error
import math

y_train_pred = model.predict(X)

mse_train = mean_squared_error(y, y_train_pred)

print("Mean Squared Error (MSE) on Training Set:", mse_train)
```

```
Mean Squared Error (MSE) on Training Set: 12043320000.380249
```

4. Run the prediction on the test dataset using the optimized regression model from part D2 to give the accuracy of the prediction model based on the mean squared error (MSE).

Below is my code and results after running the prediction on the test dataset



```
test_df = pd.read_csv(r"C:\Users\Nathan\Documents\WGU\D600\Code\Task 1\testing_data.csv")

X_test = test_df[['SquareFootage', 'CrimeRate', "NumBedrooms", "AgeOfHome"]]
X_test = sm.add_constant(X_test)
y_test = test_df['Price']

y_test_pred = model.predict(X_test)

mse_test = mean_squared_error(y_test, y_test_pred)

print("Mean Squared Error (MSE) on Test Set:", mse_test)
```

```
Mean Squared Error (MSE) on Test Set: 4234864483105.485
```

E. Summarize your data analysis by doing the following:

1. List the packages or libraries you have chosen for Python or R and justify how each item on the list supports the analysis.

Here are all the python packages I used and the reason for using them.

- Pandas - Allows us to import, read and change datasets
- Sklearn.model_selection - Allows us to split data into separate training and test sets
- Statsmodels.api - Allows us to perform regression and extract the model parameters
- Sklearn.feature_selection - Allows us to optimize the model using Recursive Feature Elimination
- Sklearn.metrics - Allows us to calculate the MSE values
- Sklearn.linear_model - Allows us to use linear regression models
- Seaborn/Matplotlib.pyplot - Allows us to use graph and visual different data relationships
- Math - Allows us to use more complex functions in the code

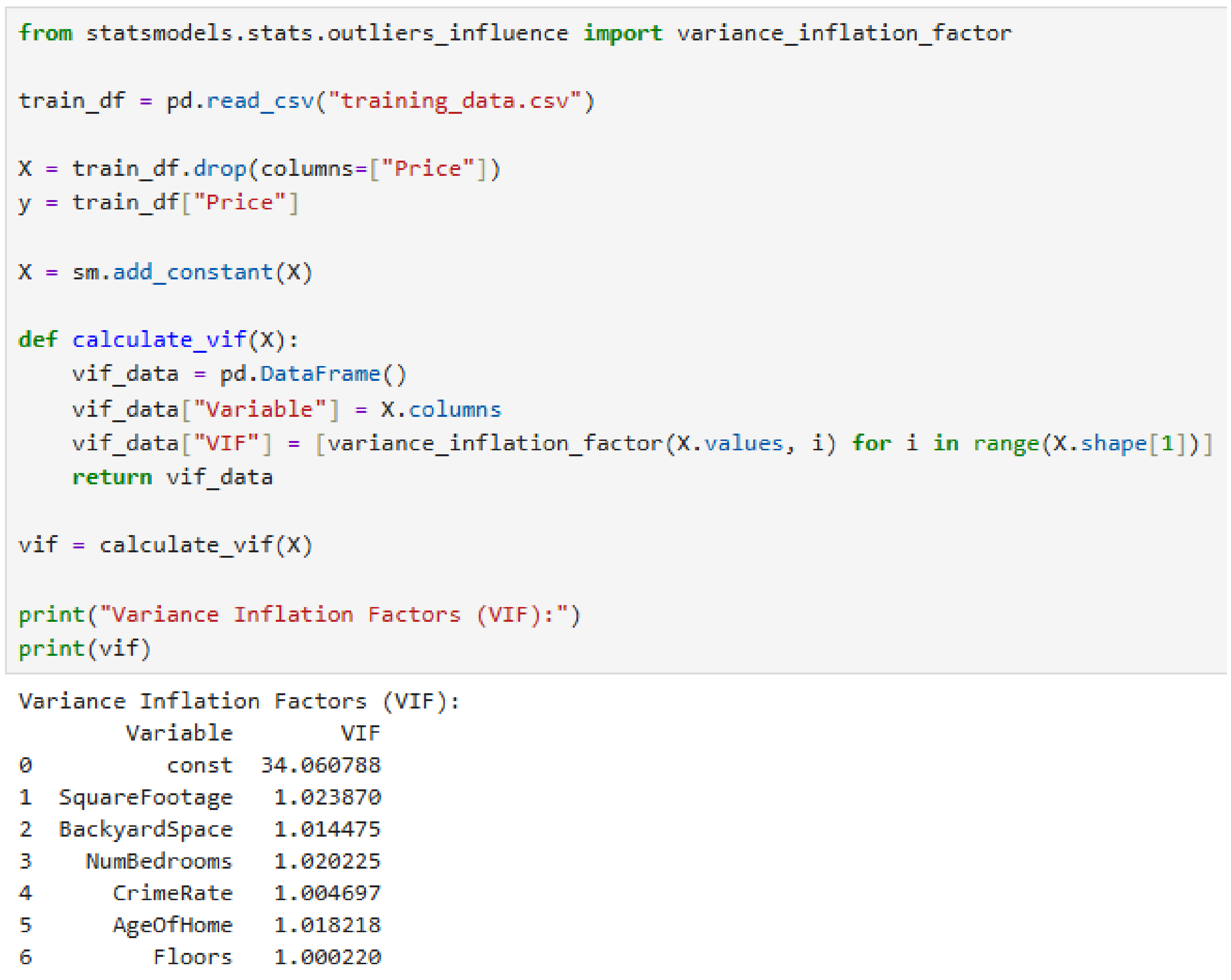
2. Discuss the method used to optimize the model and justification for the approach.

We used the Backward Stepwise Elimination method to optimize the dataset, which helps remove the least important variables. This helps improve the model performance and prevent overfitting by eliminating unnecessary predictors. It does so by relying on p-values to determine which insignificant predictors to remove. In our case any p-value greater than 0.05 will be dropped, thus simplifying the model by keeping the most significant features. We used BSE because our dataset is large enough, and variables can be removed without significantly harming the integrity of the dataset.

3. Discuss the verification of assumptions used to create the optimized model.

When building a linear regression model, we assume certain conditions are met so that the model may perform adequately and hold the results valid. If these assumptions are not met, the values produced by the model may be biased or skewed. To verify our model, we will check the dataset for multicollinearity.

To do so, I will first check for multicollinearity with the code below. We do this because the independent variables should not be highly correlated with each other.



```python
from statsmodels.stats.outliers_influence import variance_inflation_factor

train_df = pd.read_csv("training_data.csv")

X = train_df.drop(columns=["Price"])
y = train_df["Price"]

X = sm.add_constant(X)

def calculate_vif(X):
    vif_data = pd.DataFrame()
    vif_data["Variable"] = X.columns
    vif_data["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
    return vif_data

vif = calculate_vif(X)

print("Variance Inflation Factors (VIF):")
print(vif)
```

```
Variance Inflation Factors (VIF):
        Variable        VIF
0          const  34.060788
1  SquareFootage   1.023870
2  BackyardSpace   1.014475
3    NumBedrooms   1.020225
4      CrimeRate   1.004697
5      AgeOfHome   1.018218
6         Floors   1.000220
```

With the VIF value being close to 1 and under the safe threshold of 5, we can safely determine that none of the variables are multicollinear. Therefore, we can safely verify the assumptions are met.

4. **Provide the regression equation and discuss the coefficient estimates.**

Below is the regression equation.

$$y = \beta_0 + \beta_n x_n + \epsilon$$

Where:

- y = Dependent variable (Price in this case)
- x_n=Independent variable (For our case we will have 4)
- β_0 = The intercept when $x = 0$ (Constant Coefficient)
- β_n=The slope
- ϵ=Error in the model (Allows for more variance)

With our values calculated from the optimized dataset, we can plug the values in and find the equation for this dataset.

The constant coefficient, β_0, equals -39,110. This value will be our intercept.

The Square Footage coefficient, β_1, equals 176.424. This means that with every square foot added to the property, the price increases by $176.424.

The Number of Bedrooms coefficient, β_2, equals 60,810. This means that with every bedroom added, the price increases by $60,810.

The Crime Rate coefficient, β_3, equals -248.0556. This means that the higher the crime rate, the price decreases by $248.06 concurrently.

The Age of Home coefficient, β_4, equals -272.74. This means that the older the house, the price decreases by $272.74.

The error, ϵ, equals -24,800. This is a constant to our equation as it will consistently subtract $24,800 from the property's price. To help improve variance.

Subbing in variables where needed this will be our final equation

$$\textit{Price} = -39110 + (191.183 \times \textit{Square Footage}) + (60810 \times \textit{Number of Bedrooms}) + (-248.0556 \times \textit{Crime Rate}) + (272.74 \times \textit{Age of Home}) - 24800$$

5. **Discuss the model metrics by addressing each of the following:**
   - **the R2 and adjusted R2 of the training set**
   - **the comparison of the MSE for the training set to the MSE of the test set**

The R squared value is 0.472, and the adjusted R squared value is 0.472 as well. This means that the model can explain 47% of the variance for the price value but leaves 53% unexplained.

Using the MSE values from earlier, the MSE for the training set is 12043320000.380249, and the MSE for the test set is 4234864483105.485. The difference between these values is large and unexpected. Still, it could prove that the model is more complex because of the number of features, decreasing bias, and increasing variance. Still, overall, the model is overfitting and passed the optimal bias-variance trade-off point, thus making the difference between the mean squared error values large (MetricProf *YouTube*).

6. **Discuss the results and implications of your prediction analysis.**

After collecting our data points from our model, we now see that increasing the property's square footage and the number of bedrooms both increase the home's price, with the property's number of bedrooms having a larger magnitude of increasing property price. At the same time, the older the home is and the higher the crime rat,e both decrease the value of the home. However, the R squared value is 47.2%, and our MSE values for the training and testing sets differ drastically. Both of these values point to the data model needing more variables to help better predict the price of the home.

7. **Recommend a course of action for the real-world organizational situation from part B1 based on your results and implications discussed in part E6.**

A course of action based on the results is if we want to increase the price of a property, ensure that the square footage is significant, that the crime rate is low, and that we have room for multiple bedrooms. However, I recommend adding different variables and removing other variables to our model to increase the R squared value and lower the error amount. This will help our model better predict property prices and account for better variance by decreasing overfitting and removing any bias, thus making a more accurate data model.

Sources

MetricsProf. *YouTube*, YouTube, www.youtube.com/watch?v=UJ38S9N4C_U&t=794s. Accessed 27 Mar. 2025.