Task 1:

Linear Regression Analysis

Keniyah Chestnut

Statistical Data Mining - D600

SID:012601305

A: GITLAB REPOSITORY

The GitLab repository was created, cloned into the IDE, and all commits were pushed as each section was completed. A link to the repository and commit history has been submitted via the "Comments to Evaluator" section.

B1: PROPOSAL OF QUESTION

My research question is: Do square footage and crime rate affect the value of a home, and if so, how much? This analysis will help identify how each of these variables contributes to overall home price trends.

B2: DEFINED GOAL

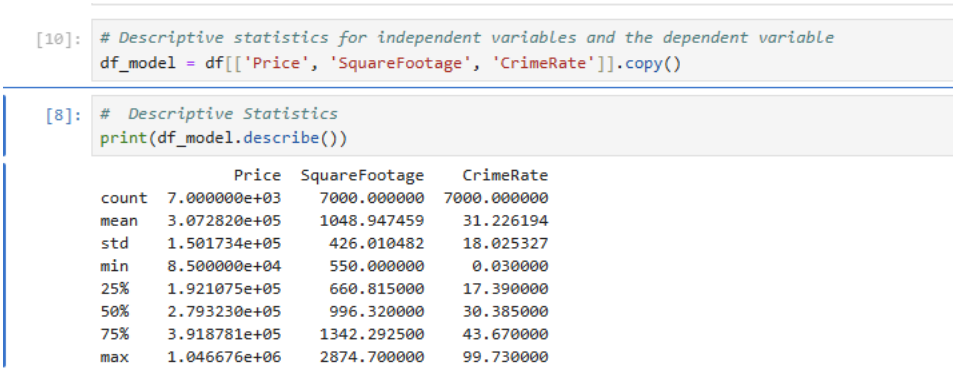
The goal of this data analysis is to use a multiple linear regression model to determine how much square footage and crime rate individually influence housing prices. A real estate firm could use this model to estimate prices for future development or appraisals.

C1: VARIABLE IDENTIFICATION

In this regression analysis, the independent variables are SquareFootage and CrimeRate, and the dependent variable is Price. The goal is to examine how changes in square footage and crime rate affect the price of a house. Since Price responds to changes in the other two variables, it is considered the dependent variable in this model.

C2: DESCRIPTIVE STATISTICS

Descriptive statistics for the selected variables are as follows:



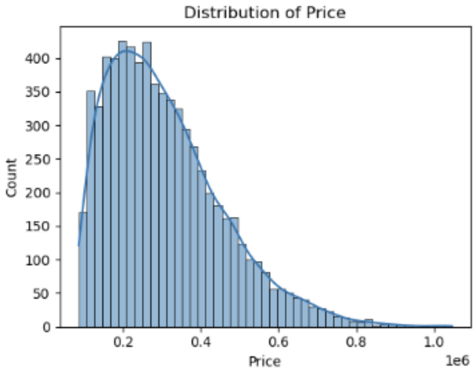
```
[10]: # Descriptive statistics for independent variables and the dependent variable
      df_model = df[['Price', 'SquareFootage', 'CrimeRate']].copy()
```

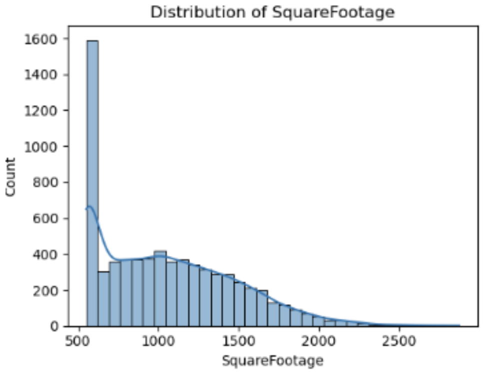
```
[8]: #  Descriptive Statistics
     print(df_model.describe())
```

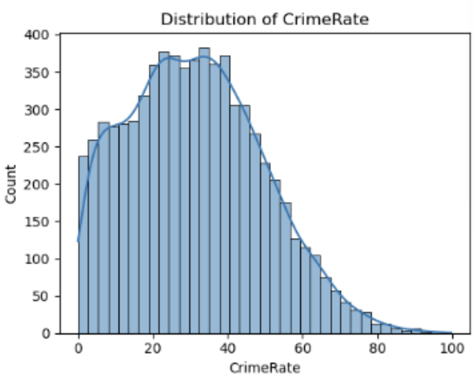
```
              Price  SquareFootage    CrimeRate
count  7.000000e+03    7000.000000  7000.000000
mean   3.072820e+05    1048.947459    31.226194
std    1.501734e+05     426.010482    18.025327
min    8.500000e+04     550.000000     0.030000
25%    1.921075e+05     660.815000    17.390000
50%    2.793230e+05     996.320000    30.385000
75%    3.918781e+05    1342.292500    43.670000
max    1.046676e+06    2874.700000    99.730000
```

C3: VISUALIZATIONS

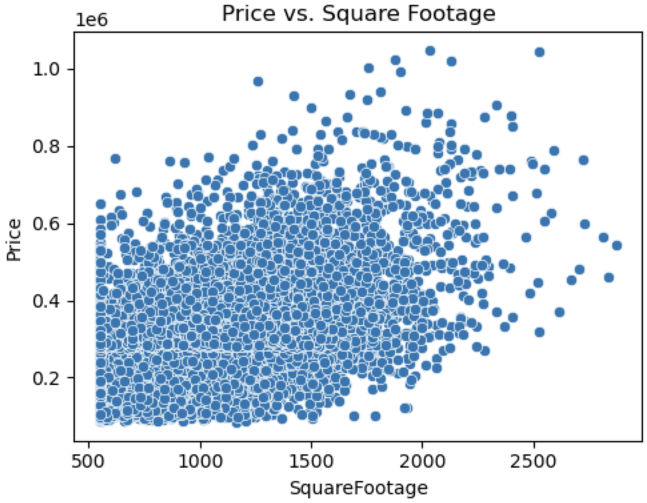
The univariate visualizations show the distribution of Price, SquareFootage, and CrimeRate

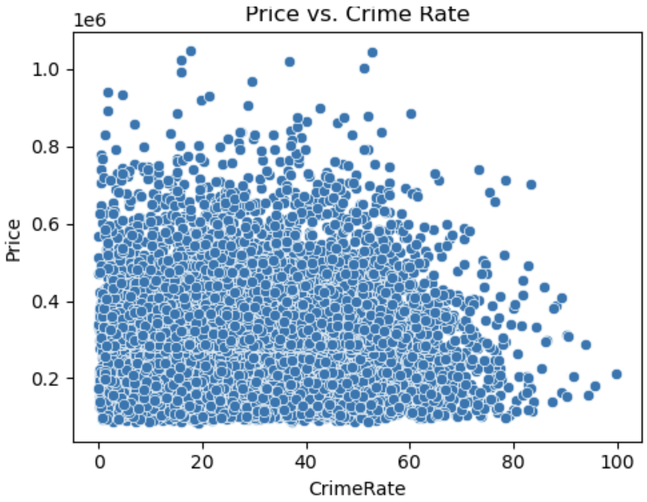






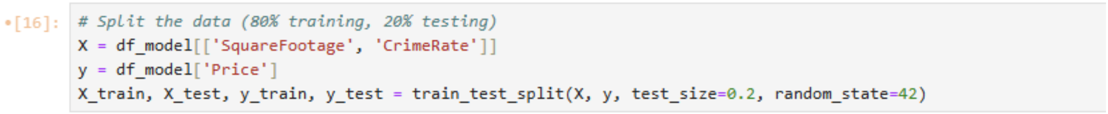
Here are the bivariate visualizations showing the relationship between Price and each independent variable, Square Footage and Crime Rate.





D1: SPLITTING THE DATA

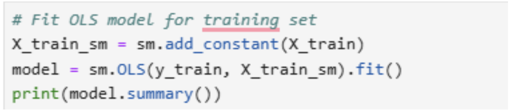
he dataset was split into 80 percent training data (5,600 rows) and 20 percent testing data (1,400 rows). Only the selected columns (SquareFootage, CrimeRate, and Price) were included in the split. This helps test how well the model works on new data.



```
•[16]: # Split the data (80% training, 20% testing)
       X = df_model[['SquareFootage', 'CrimeRate']]
       y = df_model['Price']
       X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

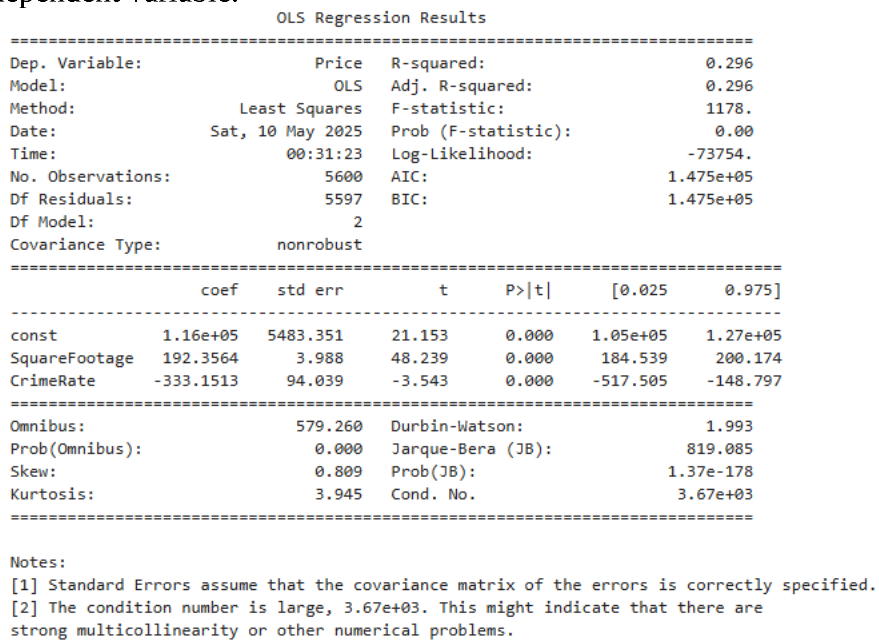
D2: MODEL OPTIMIZATION

I used Ordinary Least Squares (OLS) regression to model housing prices based on square footage and crime rate. The model was fit using the standard OLS method, which minimizes the residual sum of squares.



```
# Fit OLS model for training set
X_train_sm = sm.add_constant(X_train)
model = sm.OLS(y_train, X_train_sm).fit()
print(model.summary())
```

Here are the results after optimization. The output includes the adjusted R-squared, R-squared, F-statistic, probability of the F-statistic, coefficient estimates, and the p-values for each independent variable.

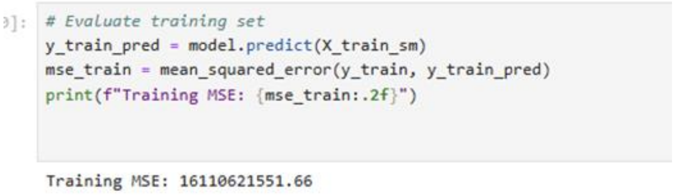


```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Price   R-squared:                       0.296
Model:                            OLS   Adj. R-squared:                  0.296
Method:                 Least Squares   F-statistic:                     1178.
Date:                Sat, 10 May 2025   Prob (F-statistic):               0.00
Time:                        00:31:23   Log-Likelihood:                -73754.
No. Observations:                5600   AIC:                         1.475e+05
Df Residuals:                    5597   BIC:                         1.475e+05
Df Model:                           2
Covariance Type:            nonrobust
===============================================================================
                  coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------
const         1.16e+05   5483.351     21.153      0.000    1.05e+05    1.27e+05
SquareFootage  192.3564     3.988     48.239      0.000     184.539     200.174
CrimeRate     -333.1513    94.039     -3.543      0.000    -517.505    -148.797
==============================================================================
Omnibus:                      579.260   Durbin-Watson:                   1.993
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              819.085
Skew:                           0.809   Prob(JB):                    1.37e-178
Kurtosis:                       3.945   Cond. No.                     3.67e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 3.67e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

D3 & D4: Mean Squared Error and Model Accuracy

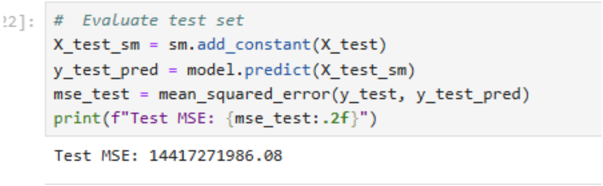
The Mean Squared Error (MSE) was calculated for both the training and test datasets using the original Ordinary Least Squares (OLS) regression model:



```
# Evaluate training set
y_train_pred = model.predict(X_train_sm)
mse_train = mean_squared_error(y_train, y_train_pred)
print(f"Training MSE: {mse_train:.2f}")
```

```
Training MSE: 16110621551.66
```

Training MSE: 16,110,621,551.66



```
#  Evaluate test set
X_test_sm = sm.add_constant(X_test)
y_test_pred = model.predict(X_test_sm)
mse_test = mean_squared_error(y_test, y_test_pred)
print(f"Test MSE: {mse_test:.2f}")
```

```
Test MSE: 14417271986.08
```

Test MSE: 14,417,271,986.08

Taking the square root of these values to get the Root Mean Squared Error (RMSE) for easier interpretation:

- Training RMSE: $126,914.31

- Test RMSE: $120,066.86

These results suggest that, on average, the model's predictions are off by about $126K on the training data and $120K on unseen test data. The relatively close values between the training and testing RMSEs indicate that the model generalizes reasonably well and is not overfitting.

E1: PACKAGES OR LIBRARIES LIST

Below is a list of the libraries I imported and the purpose each served in the analysis:

- pandas: Used for loading and manipulating the dataset in DataFrame format. It made it easy to clean, filter, and structure the data for modeling.

- matplotlib.pyplot: Essential for plotting univariate and bivariate distributions. I used it in combination with Seaborn to create visual representations of the data.

- seaborn: Provided enhanced visualization tools for plotting distributions and relationships, such as histograms and scatterplots with trend lines.

- sklearn.model_selection.train_test_split: Used to split the data into training and testing sets, which is critical for evaluating how well the model generalizes to unseen data.

- sklearn.linear_model.LinearRegression: Although included, this was not ultimately used in the final model. The statsmodels OLS method was preferred for its detailed statistical output.

- sklearn.metrics.mean_squared_error: Calculated the model’s prediction error (MSE), which helps assess the model’s accuracy.

- statsmodels.api: Used to fit the Ordinary Least Squares (OLS) regression model and provide a detailed summary of model statistics such as coefficients and p-values.

- statsmodels.stats.outliers_influence.variance_inflation_factor: Used to calculate the Variance Inflation Factor (VIF) for multicollinearity diagnostics, ensuring that independent variables are not too highly correlated.

E2: METHOD JUSTIFICATION

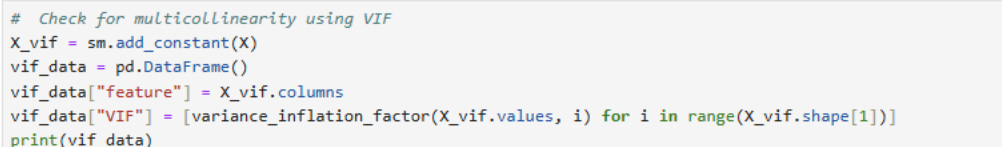
Ordinary Least Squares (OLS) regression was used to build the predictive model. This method is appropriate because it estimates the relationship between the independent variables (SquareFootage and CrimeRate) and the dependent variable (Price) by minimizing the sum of squared residuals. OLS is a widely accepted technique for linear regression analysis and provides detailed statistical outputs, such as p-values and confidence intervals, which help assess the significance of each predictor.

This method was chosen because the relationship between the features and the target variable appeared to be linear based on exploratory data analysis, and OLS provides interpretability and diagnostic tools necessary for evaluating model quality.

E3: VERIFICATION OF ASSUMPTIONS

Before running the linear regression model, it was important to verify that the assumptions of linear regression were met. particularly the assumption that there is no multicollinearity among the predictor variables.

To test this, I calculated the Variance Inflation Factor (VIF) for the independent variables, SquareFootage and CrimeRate. VIF measures how much the variance of a regression coefficient is inflated due to multicollinearity.



```
#  Check for multicollinearity using VIF
X_vif = sm.add_constant(X)
vif_data = pd.DataFrame()
vif_data["feature"] = X_vif.columns
vif_data["VIF"] = [variance_inflation_factor(X_vif.values, i) for i in range(X_vif.shape[1])]
print(vif_data)
```

A VIF value greater than 5 (and especially over 10) typically indicates high multicollinearity. In this case, the VIF values were low for both predictors, suggesting no significant multicollinearity. This confirms that the assumption of independent predictors was met, and the model is valid for interpretation.

E4: EQUATION

$$\boldsymbol{Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \epsilon}$$

Where:

$$Y$$

Y is the predicted Price of a home

$$\boldsymbol{\beta_0}$$

is the intercept, or baseline price when all predictors are 0

$\boldsymbol{\beta_1 x_1}$

is the coefficient for Square Footage

$\boldsymbol{\beta_2 x_2}$

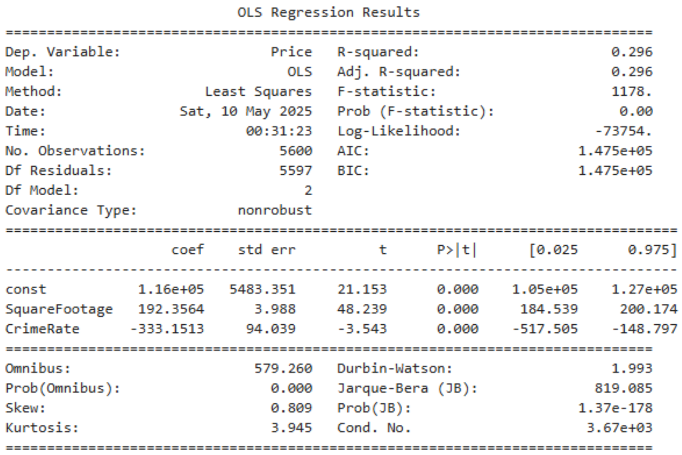
is the coefficient for Crime Rate

ε

ε is the error term, representing unexplained variability

Based on the model results, the equation is:

$$\text{Price} = 115{,}989.40 + 192.36 \times (\text{SquareFootage}) - 333.15 \times \text{CrimeRate}$$



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Price   R-squared:                       0.296
Model:                            OLS   Adj. R-squared:                  0.296
Method:                 Least Squares   F-statistic:                     1178.
Date:                Sat, 10 May 2025   Prob (F-statistic):               0.00
Time:                        00:31:23   Log-Likelihood:                -73754.
No. Observations:                5600   AIC:                         1.475e+05
Df Residuals:                    5597   BIC:                         1.475e+05
Df Model:                           2
Covariance Type:            nonrobust
=================================================================================
                    coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------
const           1.16e+05   5483.351     21.153      0.000    1.05e+05    1.27e+05
SquareFootage   192.3564      3.988     48.239      0.000     184.539     200.174
CrimeRate      -333.1513     94.039     -3.543      0.000    -517.505    -148.797
==============================================================================
Omnibus:                      579.260   Durbin-Watson:                   1.993
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              819.085
Skew:                           0.809   Prob(JB):                    1.37e-178
Kurtosis:                       3.945   Cond. No.                     3.67e+03
==============================================================================
```

Interpretation of coefficients:

Intercept ($115,989.40): Represents the estimated base price of a home when both square footage and crime rate are zero.

SquareFootage ($192.36): For every additional square foot, the price of the house increases by approximately $192.36, assuming crime rate is constant.

CrimeRate (-$333.15): For every one-unit increase in crime rate, the price of the house decreases by approximately $333.15, assuming square footage is constant.

This equation allows us to predict house prices based on square footage and neighborhood crime rate, providing insight into how each factor impacts value.

E5: MODEL METRICS

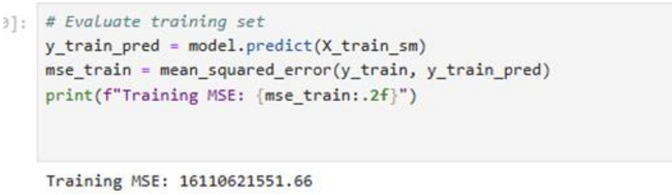
From the regression summary:

- R-squared = 0.296
- Adjusted R-squared = 0.296

The R-squared value indicates that about 29.6% of the variation in housing prices can be explained by the two predictors: SquareFootage and CrimeRate. While this shows a relationship, it also suggests that 70.4% of the price variation is due to other unaccounted factors such as location, number of bedrooms, property condition, etc.

Because only two variables were used, the Adjusted R-squared remains the same (0.296), confirming that the model is not penalized for including unnecessary predictors.

A comparison of the Mean Squared Error (MSE) for the training and test sets is provided below for the original model. No optimization technique (such as backward elimination) was applied in this version of the model.

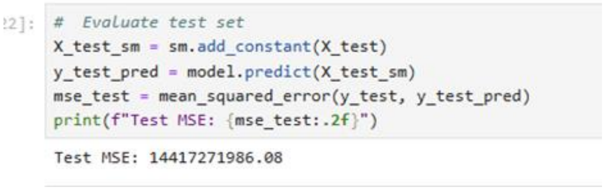


```
]: # Evaluate training set
   y_train_pred = model.predict(X_train_sm)
   mse_train = mean_squared_error(y_train, y_train_pred)
   print(f"Training MSE: {mse_train:.2f}")

   Training MSE: 16110621551.66
```

Training MSE: 16,110,621,551.66

Training RMSE: $126,914.31



```
2]: #  Evaluate test set
    X_test_sm = sm.add_constant(X_test)
    y_test_pred = model.predict(X_test_sm)
    mse_test = mean_squared_error(y_test, y_test_pred)
    print(f"Test MSE: {mse_test:.2f}")

    Test MSE: 14417271986.08
```

Test MSE: 14,417,271,986.08

Test RMSE: $120,066.86

These values show that, on average, the model's price predictions are off by about $126K on the training data and $120K on the test data. The relatively small gap between the two suggests that the model generalizes fairly well without overfitting, although there is still significant error, likely due to the simplicity of the model and missing important predictors such as location, age of home, or number of bedrooms.

E6–E7: Results, Implications, and Recommended Course of Action

The results of the regression model show that two key factors influence housing prices:

- Square footage is positively associated with price. Larger homes tend to be more valuable.
- Crime rate is negatively associated with price. Higher crime rates reduce home value.

However, the model's Root Mean Squared Error (RMSE) was approximately $123,173.41, and the R-squared value was 0.296. This means the model explains only 29.6% of the variance in housing prices, suggesting that 70.4% of price variability is influenced by other factors not captured in this analysis. While square footage and crime rate are relevant, they are not sufficient on their own to accurately predict home value.

Despite this limitation, the findings offer useful insights. Builders and developers could focus on constructing larger homes in low-crime areas to increase property value. To improve the model's accuracy and usefulness, the following steps are recommended:

1. Expand the model by including more variables such as location, number of bedrooms, year built, and school quality.
2. Rerun the multiple linear regression with the updated dataset to produce more accurate predictions and support stronger decision-making.

With a more complete model, stakeholders in real estate and development can better understand price drivers and make data-informed investment choices.

References

No outside sources were used other than official WGU course materials.