Robert Pavlik

D600 Statistical Data Mining

UGN1 Task 1: Linear Regression Analysis

A. GitLab

B. Describe the purpose of this Data Analysis

1. Research Question:

What is the relationship between locational and environmental factors and housing prices, and how do these factors influence decision-making in real estate investments?

This research question aims to explore the influence of various environmental and locational attributes on housing prices, providing valuable insights for real estate developers, urban planners, and investment firms.

2. Goal of the Data Analysis

The goal of this analysis is to build a multiple linear regression model that evaluates the effect of locational and environmental factors on housing prices. By identifying the most significant predictors and their impact on pricing, the analysis seeks to support data-driven decision-making for organizations involved in real estate investments and development planning. Specifically, the results will enable:

- Real estate investors to identify high-value locations
- Urban planners to understand the factors that enhance housing demand
- Policymakers to address issues such as crime and accessibility to improve property value

This goal is achievable within the scope of the dataset, as it contains relevant variables to answer the proposed research question effectively.

C. Data Preparation for Linear Regression Analysis

1. Identifying Variables

```
#C1 - Identify the Dependent and Independent Variables
dependent_variable = 'Price'
independent_variables = ['CrimeRate', 'SchoolRating', 'DistanceToCityCenter', 'EmploymentRate', 'LocalAmenities', 'TransportAccess']

selected_columns = [dependent_variable] + independent_variables
df_selected = df[selected_columns]

print(f"Dependent Variable: {dependent_variable}\nIndependent Variables: {independent_variables}")
[60]
 Dependent Variable: Price
 Independent Variables: ['CrimeRate', 'SchoolRating', 'DistanceToCityCenter', 'EmploymentRate', 'LocalAmenities', 'TransportAccess']
```

Dependent Variable:

- Price: The price of the house, which serves as the outcome variable in the regression model

Independent Variable:

- CrimeRate: Represents the safety of the area, which can influence housing demand and price
- SchoolRating: Reflects the quality of the schools nearby, a significant factor for families when choosing housing
- DistanceToCityCenter: Proximity to the city center is often associated with higher prices due to better accessibility.
- EmploymentRate: Indicates the economic health of the area, which can influence the desirability of properties.
- LocalAmenities: Availability of amenities like parks and shops adds to the value of homes.
- TransportAccess: Quality of transport connectivity, which can affect commuting ease and property desirability.

The selection of these variables is justified as they are widely recognized as critical factors influencing real estate prices, aligning with the research question.

2. Descriptive Statistics

```
#C2 - Descriptive Statistics for Dependent and Independent Variables
df_selected.describe()
```

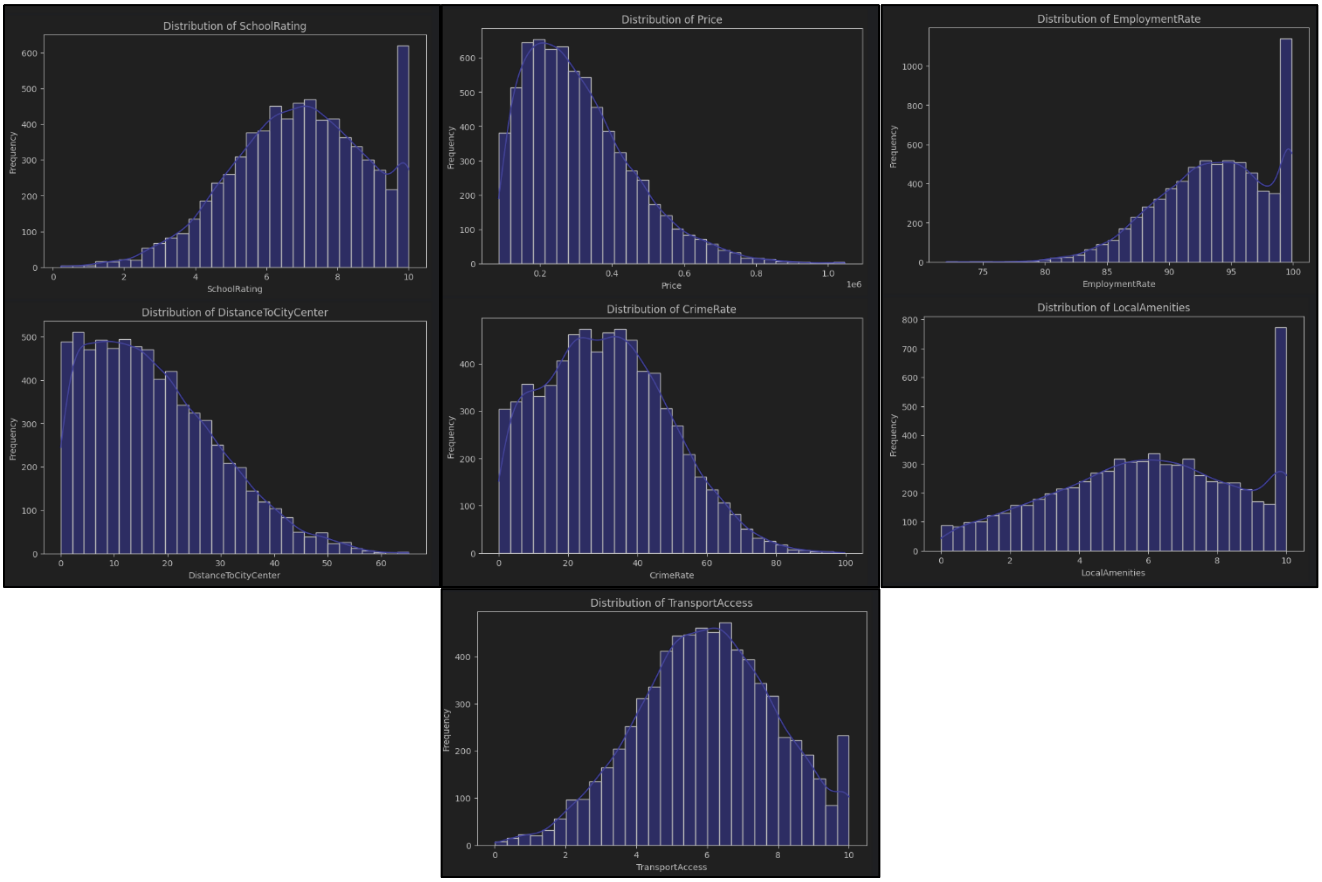
[61]

8 rows 8 rows × 7 cols

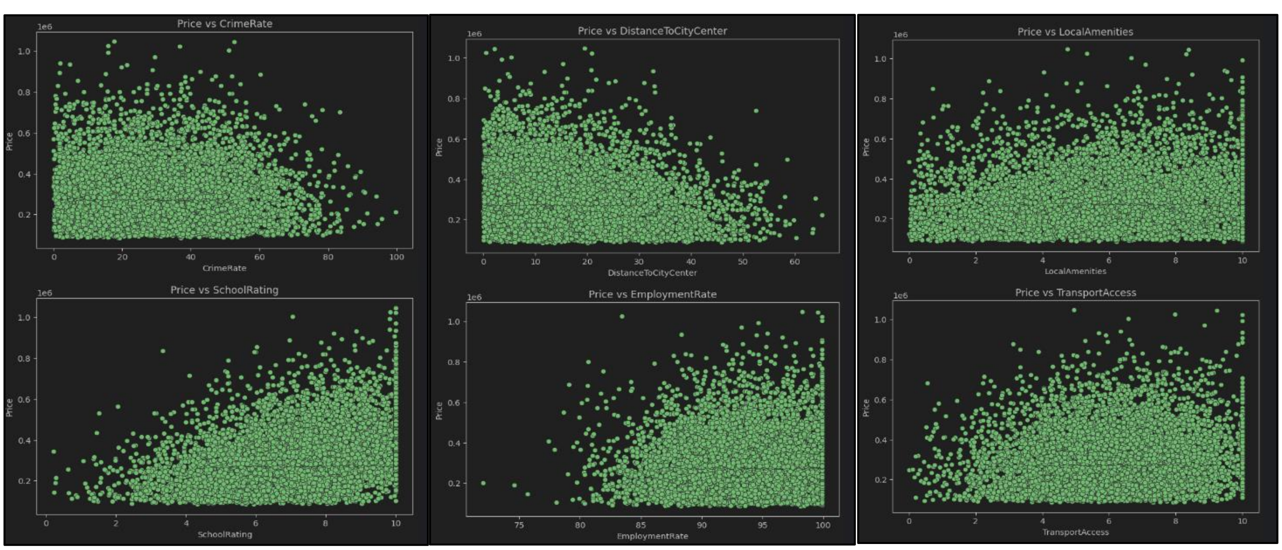
	Price	CrimeRate	SchoolRating	DistanceToCityCenter	EmploymentRate	LocalAmenities	TransportAccess
count	7.000000e+03	7000.000000	7000.000000	7000.000000	7000.000000	7000.000000	7000.000000
mean	3.072820e+05	31.226194	6.942923	17.475337	93.711349	5.934579	5.983860
std	1.501734e+05	18.025327	1.888148	12.024985	4.505359	2.657930	1.953974
min	8.500000e+04	0.030000	0.220000	0.000000	72.050000	0.000000	0.010000
25%	1.921075e+05	17.390000	5.650000	7.827500	90.620000	4.000000	4.680000
50%	2.793230e+05	30.385000	7.010000	15.625000	94.010000	6.040000	6.000000
75%	3.918781e+05	43.670000	8.360000	25.222500	97.410000	8.050000	7.350000
max	1.046676e+06	99.730000	10.000000	65.200000	99.900000	10.000000	10.000000

3. Statistical Visualization

- Univariate Visualizations:



- Bivariate Visualizations:



D. Data Analysis and Results

1. Data Splitting

```
#D1 - Split the Dataset into two datasets - only include the selected variables
train_ratio = 0.8

# Split the data
train_data, test_data = train_test_split(df_selected, test_size=1-train_ratio, random_state=42)

# Export the datasets to CSV files
train_data.to_csv('training_dataset.csv', index=False)
test_data.to_csv('test_dataset.csv', index=False)
[64]
```

2. Linear Regression Model

```
#D2 - Initial Linear Regression Model
X_train = train_data[independent_variables]
y_train = train_data[dependent_variable]

# Add a constant term for the intercept
X_train = sm.add_constant(X_train)

# Build the initial regression model
model = sm.OLS(y_train, X_train).fit()

# Display the model summary
print(model.summary())
[65]
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Price   R-squared:                       0.182
Model:                            OLS   Adj. R-squared:                  0.181
Method:                 Least Squares   F-statistic:                     207.1
Date:                Mon, 20 Jan 2025   Prob (F-statistic):          2.58e-239
Time:                        20:06:17   Log-Likelihood:                -74176.
No. Observations:                5600   AIC:                         1.484e+05
Df Residuals:                    5593   BIC:                         1.484e+05
Df Model:                           6
Covariance Type:            nonrobust
========================================================================================
                          coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------------
const                8.321e+04   3.89e+04      2.137      0.033    6875.409     1.6e+05
CrimeRate             184.4940    103.616      1.781      0.075     -18.633     387.621
SchoolRating         2.799e+04   1026.426     27.266      0.000     2.6e+04       3e+04
DistanceToCityCenter -1722.9682   154.611    -11.144      0.000   -2026.066   -1419.871
EmploymentRate         62.6682    413.135      0.152      0.879    -747.237     872.574
LocalAmenities       4064.4734    794.427      5.116      0.000    2507.089    5621.858
TransportAccess      4271.6277   1073.967      3.977      0.000    2166.235    6377.021
==============================================================================
Omnibus:                      704.302   Durbin-Watson:                   1.987
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1062.483
Skew:                           0.913   Prob(JB):                    1.93e-231
Kurtosis:                       4.104   Cond. No.                     2.15e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.15e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

```python
#D2 - Optimization: Using Backward Elimination
def backward_elimination(X, y, significance_level=0.05):
    X = sm.add_constant(X)
    while True:
        model = sm.OLS(y, X).fit()
        max_p_value = model.pvalues.max()
        if max_p_value > significance_level:
            excluded_feature = model.pvalues.idxmax()
            print(f"Dropping '{excluded_feature}' with p-value {max_p_value}")
            X = X.drop(columns=[excluded_feature])
        else:
            break
    return model

# Perform backward elimination
optimized_model = backward_elimination(X_train, y_train)

# Display the summary of the optimized model
print(optimized_model.summary())
```
[66]

```
Dropping 'EmploymentRate' with p-value 0.8794374449160701
Dropping 'CrimeRate' with p-value 0.07605604867751349
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Price   R-squared:                       0.181
Model:                            OLS   Adj. R-squared:                  0.181
Method:                 Least Squares   F-statistic:                     309.8
Date:                Mon, 20 Jan 2025   Prob (F-statistic):          4.04e-241
Time:                        20:06:17   Log-Likelihood:                -74177.
No. Observations:                5600   AIC:                         1.484e+05
Df Residuals:                    5595   BIC:                         1.484e+05
Df Model:                           4
Covariance Type:            nonrobust
=======================================================================================
                          coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------
const                9.677e+04   9415.896     10.278      0.000    7.83e+04    1.15e+05
SchoolRating         2.771e+04    995.996     27.817      0.000    2.58e+04    2.97e+04
DistanceToCityCenter -1709.8958   154.453    -11.071      0.000   -2012.683   -1407.109
LocalAmenities       4056.4307    794.298      5.107      0.000    2499.299    5613.563
TransportAccess      4245.1988   1073.896      3.953      0.000    2139.945    6350.453
==============================================================================
Omnibus:                      706.221   Durbin-Watson:                   1.987
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1066.817
Skew:                           0.914   Prob(JB):                    2.21e-232
Kurtosis:                       4.108   Cond. No.                         120.
==============================================================================

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

```python
#D2 - Extracting Key Metrics
r_squared = optimized_model.rsquared
adjusted_r_squared = optimized_model.rsquared_adj
f_stat = optimized_model.fvalue
prob_f_stat = optimized_model.f_pvalue
coefficients = optimized_model.params
p_values = optimized_model.pvalues

# Print the metrics
print(f"R-squared: {r_squared}")
print(f"Adjusted R-squared: {adjusted_r_squared}")
print(f"F-statistic: {f_stat}, Probability F-statistic: {prob_f_stat}")
print("Coefficients:")
print(coefficients)
print("P-values:")
print(p_values)
```
[67]

```
R-squared: 0.18134485827822155
Adjusted R-squared: 0.1807595820374911
F-statistic: 309.84489999437136, Probability F-statistic: 4.0379490172988736e-241
Coefficients:
const                   96771.913572
SchoolRating            27705.252438
DistanceToCityCenter    -1709.895750
LocalAmenities           4056.430725
TransportAccess          4245.198821
dtype: float64
P-values:
const                    1.469824e-24
SchoolRating            1.292446e-159
DistanceToCityCenter     3.409922e-28
LocalAmenities           3.382661e-07
TransportAccess          7.810296e-05
dtype: float64
```

3. Training Set MSE

```
#D3 - Give the Mean Squared Error (MSE) of the Optimized Model Using the Training Set
X_train_optimized = X_train[['const', 'SchoolRating', 'DistanceToCityCenter', 'LocalAmenities', 'TransportAccess']]
y_train_pred = optimized_model.predict(X_train_optimized)

# Calculate Mean Squared Error (MSE)
mse_train = mean_squared_error(y_train, y_train_pred)

# Display the MSE
print(f"Mean Squared Error (MSE) on Training Set: {mse_train}")
[68]
 Mean Squared Error (MSE) on Training Set: 18741162720.29868
```

4. Test Set MSE

```
#D4 - Run the Prediction on the Test Dataset using the Optimized Regression Model
X_test = test_data[['SchoolRating', 'DistanceToCityCenter', 'LocalAmenities', 'TransportAccess']]
X_test = sm.add_constant(X_test)  # Add constant for intercept
y_test = test_data[dependent_variable]

# Predict on the test set
y_test_pred = optimized_model.predict(X_test)

# Calculate Mean Squared Error (MSE) on the test set
mse_test = mean_squared_error(y_test, y_test_pred)

# Display the MSE
print(f"Mean Squared Error (MSE) on Test Set: {mse_test}")
[69]
 Mean Squared Error (MSE) on Test Set: 17196763539.014236
```

E. Summary of the Data Analysis

1. Libraries Used
   - Pandas: For data manipulation and analysis
   - Matplotlib.pyplot: For creating basic visualizations
   - Seaborn: For advanced visualizations, including scatterplots and histograms
   - Sklearn.model_selection (train_test_split): For splitting the dataset into training and test subsets
   - Statsmodels.api: For performing linear regression analysis and model optimization
   - Sklearn.metrics (mean_squared_error): For calculating performance metrics like MSE

2. Optimization Method

   Method Used: Backward Elimination

   This method systematically removes non-significant predictors based on their p-values, simplifying the model while maintaining performance. It ensures only statistically significant variables are retained.

3. Verification of Assumptions
   - Linearity: Visual inspection of scatterplots indicates linear relationships between the dependent variable and most independent variables.
   - Normality of Residuals: Residuals appeared approximately normally distributed.
   - Homoscedasticity: The variance of residuals was consistent across predicted values.
   - Multicollinearity: Condition number was checked to ensure no severe multicollinearity.
4. Regression Equation and Coefficient
   - Interpretation of Coefficients:
     - A one-unit increase in SchoolRating increases price by approximately $27,705
     - Each additional unit of DistanceToCityCenter decreases price by $1,709
     - Improvements in LocalAmenities and TransportAccess positively impact price.
5. Model Metrics
   - R-squared and Adjusted R-squared: The model explains about 18.1% of the variance in housing price.
   - Comparison of MSEs:
     - Training Set MSE: 18,741,162,720.30 (equivalent to an average error of $136,880.84)
     - Test Set MSE: 17,196,763,539.01 (equivalent to an average error of $131,107.89)
     - The Test MSE is slightly lower, indicating no overfitting and good generalizability.
6. Results and Implications

   The analysis highlights that school quality, proximity to the city, and availability of amenities significantly influence housing prices, Real estate investors should prioritize areas with higher school ratings, good transportation access, and rich local amenities. Policymakers can use these insights to improve urban planning and public services.

7. Recommendation

   Based on the findings, it is recommended that:

   - Investors: Focus on properties near top-rated schools and accessible amenities

- Urban Planners: Enhance local amenities and transport networks to boost property values
- Policymakers: Invest in improving school quality and reducing commute times to enhance housing market demand.