Probability and Stats for AI

Real Estate Evaluation

AAI 500 Final Project - Yash Chaturvedi

Introduction

This project is a part of AAI 500 Probability and Statistics for Artificial Intelligence. In this project, we will work with a dataset of real estate transactions in Taiwan to predict the house price of a unit area based on several features, such as the transaction date, house age, distance to the nearest MRT station, number of convenience stores, latitude, and longitude. The main objective is to find the best model for house price per local unit area after a thorough data analysis.

Data Description

The dataset used in the analysis of this project was adapted from the UCI machine learning repository.

Identification of Predictor Variables & Response Variables

Predictor Variable Descriptions -

X1=the transaction date (for example, 2013.250=2013 March, 2013.500=2013 June, etc.)

X2=the house age (unit: year)

X3=the distance to the nearest MRT station (unit: meter)

X4=the number of convenience stores in the living circle on foot (integer)

X5=the geographic coordinate, latitude. (unit: degree)

X6=the geographic coordinate, longitude. (unit: degree)

Response

y = house price per local unit area (NewTaipei Dollar $10000 per 3.3 square meters)

1. Data Cleaning/Preparation

A preliminary analysis of the data was performed before any model selection or diagnostics were taken. First, each of the variables was analyzed for abnormalities including looking for null, NaN and missing values. By checking for assumptions on each variable, one can safely and accurately use various regression analysis later in the project to choose the best model of housing prices. If any variables were found to possess any of the previously mentioned issues, a transformation was applied to that variable to remedy the problem. Our dataset didn't have any of the above mentioned issues.

Looking at the independent variables we found that the dataset contained a column "No" which was not relevant for our analysis and was just a count for the observations. Therefore,

we drop this from our dataset. Our column names as mentioned above in (**Table 1**) also contained spaces, and these were altered to replace the spaces with underscores to maintain consistency.

2. Exploratory Data Analysis

A frequency distribution is a way of organizing and displaying data in a tabular form to show the number of times a particular value or range of values occurs in a dataset. In other words, it represents how often each value or range of values appears in a dataset. We also construct a frequency distribution of the 'house price of unit area' variable (**Figure 2.1**). We also construct the same tables but instead showing the percentage of houses that fell in the specified price range (**Figure 2.2**).

A histogram is a graphical representation of a frequency distribution, showing the frequency or count of houses within specified price intervals. We construct a histogram of the 'house price of unit area' variable (**Figure 2.3**). As we can see from the figure, the histogram is positively skewed with a few outliers on the upper hand. The positive skewness would also mean that the mean would be greater than the median. The histogram of the data also resembles a bell-shaped curve. A normal distribution typically has a symmetric, unimodal distribution with the majority of the data clustered around the mean which we can see here apart from the existing outliers.

We also calculate the mean, median and standard deviation for the response variable. Mean = 37.98 means that the house price of unit area, in Taiwan Dollar/Ping across New Taipei City, Taiwan has an average of 37.98 in tens of thousand taiwan dollars. Median = 38.45 is the middle value. Therefore, we can say that houses with their house prices per unit area = 38.45 ten thousand taiwan dollars have their prices more than that of 50% houses from the observations and have less than 50% of the houses from the observations. Standard Deviation = 13.59 is the typical difference in the house price of unit area from the mean house prices per unit area. It is a measure of the variability. Approximately 68% of the house prices would fall within one standard deviation of the mean, which would be in the range of mean ± standard deviation, i.e., 37.98 ± 13.59. This range would be from 24.39 to 51.57. Approximately 95% of the house prices would fall within two standard deviations of the mean, which would be in the

range of mean ± 2 * standard deviation, i.e., 37.98 ± 2 * 13.59. This range would be from 10.80 to 65.16. Almost all of the house prices would fall within three standard deviations of the mean, which would be in the range of mean ± 3 * standard deviation, i.e., 37.98 ± 3 * 13.59. This range would be from -2.79 to 78.75. As we can also see the median is greater than the mean, so we can say that the distribution is right-skewed or positively skewed. In a right-skewed distribution, the majority of the data points are located on the left-hand side of the distribution, with a long tail on the right-hand side. This means that there are some extreme values or outliers towards the higher end of the distribution, which pull the mean towards that direction, while the median remains closer to the center of the distribution.

The explanatory variable 'Y_house_price_of_unit_area' was found to possess outlier/s as well as displaying behavior of a non-normally distributed variable (right skewed)(**Figure 2.3**). Because of this, a Box-Plot was constructed which seemed to be slightly negatively skewed (**Figure 2.4**). A box plot, also known as a box-and-whisker plot, is a graphical representation of a dataset that displays the distribution of the data along with its median, quartiles, and any outliers. The min was found to be closer to the lower quartile, then the max which was a bit farther away than the upper quartile. There seemed to be 2 outliers. We will check for outliers later in the analysis as well, and will propose methods to detect and remove them. Apart from the techniques we discuss in the project, here are some other ways to highlight observations that fall far from the model fit -

- We can plot residuals to detect observations that fall far from the trend generated by the model.
- The leverage is a measure of an observation's potential influence on the fit. Observations for which explanatory variables are far from their means have greater potential influence on the least squares estimates.
- For an observation to actually be influential, it must have both a relatively large leverage and a relatively large residual.

We use Z-scores and IQR (Interquartile Range) to detect the outliers. Z-score can be calculated using the formula: (data point - mean) / standard deviation. The z-score method is sensitive to extreme values and outliers, and can be more appropriate for normally distributed data. However it may classify data points as outliers even if they are not truly unusual, especially if the data is skewed or has heavy tails. IQR stands for interquartile range, which is a measure of

variability or spread in a dataset. It represents the range of the middle 50% of the data, or the distance between the 25th percentile (Q1) and the 75th percentile (Q3). The IQR method is robust to extreme values and outliers, and can be more appropriate for skewed data or data with heavy tails. However it may not detect outliers in a symmetrical distribution or in a distribution with multiple modes.

Using the Z-scores we found one outlier, while through the IQR we found three of them. We proceed to remove the outliers pointed out by the IQR. After removing, we again construct a histogram to check the distribution of the data, and as we can see (**Figure 2.5**) the distribution is normal now. We also calculate the five number summary for the response variable which can be seen in **Table 2**. The lower quartile was 27.70, so around 25% of the observations have house prices per unit area under $277000. The upper quartile was 46.60, so around 75% of the observations have house prices per unit area under $466000.

Median	38.40
Lower Quartile	27.50
Upper Quartile	46.30
Min Value	73.60
Max Value	7.60

Table 2: Five Number Summary for 'Y_house_price_of_unit_area'

Scatterplots demonstrate the relationships between the variables and their distributions. We also construct a pairwise scatter plot which can be seen in (**Figure 2.6**). The output shows that the house price of a unit area was positively correlated with the number of convenience stores, but negatively correlated with the distance to the nearest MRT station and the house age. There was also a moderate positive correlation with latitude and a moderate positive correlation with longitude.

Correlation refers to the statistical association or relationship between two variables. It measures the degree to which the values of one variable are related to the values of another variable. We find the correlation between the dependent variable and the predictors, and also between all the predictors. A correlation coefficient of +1 indicates a perfect positive

correlation, meaning that as one variable increases, the other variable also increases at a constant rate. A correlation coefficient of -1 indicates a perfect negative correlation, meaning that as one variable increases, the other variable decreases at a constant rate. A correlation coefficient of 0 indicates no correlation, meaning that there is no linear relationship between the two variables. We construct a heatmap to portray the pairwise correlations (**Figure 2.7**) and notice that there is no strong correlation between any pair of variables except the following. We look for any values where the absolute value is .7 or greater for a strong correlation.

- X3_distance_to_the_nearest_MRT_station has a strong correlation with X6_longitude.
- Y_house_price_of_unit_area and X3_distance_to_the_nearest_MRT_station have a correlation of 0.7 which can be considered a strong correlation.

As we know X3_distance_to_the_nearest_MRT_station has a strong correlation with X6_longitude, this will interfere with the assumptions of independence for regression models, and as we can see that X3_distance_to_the_nearest_MRT_station has a strong correlation with our dependent variable too, we will drop X6_longitude from our analysis. After removing this, we construct a heat map again to check the correlation, and now they are looking better (**Figure 2.8**). Removing this column didn't affect our other variables too much.

3. Model Selection

We fit models to data in order to make predictions or inferences about the underlying relationship between the variables of interest. We tried to fit different models to the data, and later we discussed various methods that can be used to select a model. The models that we fit were:

- **Normal GLM** using identity link function (explicitly models a normally distributed response variable). The Generalized Linear Model Regression Results for this model can be seen in **Figure 3.1.1.**
- **Gamma GLM** using identity link function (assumes that the response variable has a gamma distribution which is commonly used to model non-negative continuous

data that is right-skewed which we already know is the case.). The Generalized Linear Model Regression Results for this model can be seen in **Figure 3.1.2.**

- **Multiple Linear Regression** used to model the linear relationship between a dependent variable and two or more independent variables. The Generalized Linear Model Regression Results for this model can be seen in **Figure 3.1.3.**
- **Multiple Linear Regression with log transformed response variable.** We take the logarithmic of the response variable value and try to fit the model. The Generalized Linear Model Regression Results for this model can be seen in **Figure 3.1.4.**
- **Gamma GLM with log transformed response variable.** We take the logarithmic of the response variable value and try to fit the model. The Generalized Linear Model Regression Results for this model can be seen in **Figure 3.1.5.**

To compare different models, we can use a combination of various evaluation metrics and techniques. Here are a few methods we can use to compare the above mentioned methods:

- **Mean Squared Error (MSE):** MSE measures the average of the squared differences between the predicted and actual values. A model with a lower MSE is better.
- **R-squared (R2):** R-squared measures the proportion of variance in the dependent variable that is explained by the independent variables. A model with a higher R2 is better.
- **Akaike Information Criterion (AIC):** AIC is a measure of the trade-off between the goodness of fit of a model and its complexity. A model with a lower AIC is preferred over a model with a higher AIC.
- **Bayesian Information Criterion (BIC):** BIC is similar to AIC but has a stronger penalty for model complexity. A model with a lower BIC is preferred over a model with a higher BIC.

While fitting our models we note down the AIC and BIC values for each model which can be shown below in **Table 3:**

Models	AIC	BIC
Normal GLM	2930.39	26884.50
Gamma GLM	2933.45	-2419.91
Multiple Linear Regression	2930.39	2950.49
Multiple Linear Regression with log transformed Y	-94.93	-74.84
Gamma GLM with log transformed Y	-44.25	-2441.87

Table 3: Models and their AIC and BIC

As we can see the linear regression model with logarithmic response variable has the lowest AIC of -94.93, therefore we will be choosing to work with this model.

4. Model analysis

Model analysis involves evaluating the performance and reliability of a statistical model, in order to assess its fit to the data and its suitability for making predictions or inferences. Our data does follow the assumptions of the Linear Regression Model - linearity, independence, homoscedasticity, normality, no multicollinearity, and no influential outliers.

We fit the linear regression model with a logarithmic response variable and divide our dataset in a ratio of 30% for testing and 70% for training the model. Fitting the model, we get the model coefficients which can be seen in **Figure 4.1.**

Thus our prediction equation was:

> ***log_Y_house_price_of_unit_area = -405.17 + 0.104 * X1_transaction_date - 0.007 * X2_house_age - 0.0002 * X3_distance_to_the_nearest_MRT_station + 0.026 * X4_number_of_convenience_stores + 8.004 * X5_latitude***

We also find the R-squared and Adj R-squared values for the model. R-squared and Adjusted R-squared are used to evaluate the goodness-of-fit of a regression model.

R-squared is a statistical measure that represents the proportion of the variance for a dependent variable that's explained by an independent variable or variables in a regression model. It takes a value between 0 and 1, with 1 indicating a perfect fit. In this case, the R-squared for the training data is 0.77, indicating that the model explains 77% of the variance in the dependent variable.

Adjusted R-squared is a modified version of R-squared that adjusts for the number of predictors in a model. As the number of predictors increases, R-squared will always increase, even if the new predictors are not relevant. Adjusted R-squared, on the other hand, penalizes the addition of irrelevant predictors by adjusting for the number of predictors in the model. In this case, the Adjusted R-squared for the training data is also 0.77, indicating that the model fits well even after accounting for the number of predictors.

R-squared and Adjusted R-squared can also be computed for the test data to evaluate how well the model generalizes to new data. In this case, the R-squared for the test data is 0.53, indicating that the model explains 53% of the variance in the dependent variable for the test data. The Adjusted R-squared for the test data is 0.52, which is slightly lower than the training data, indicating that the model may be overfitting to the training data.

Finally after predicting the values based on the model that we trained on the 70% of the data, we test it on the remaining 30% and plot the residuals (**Figure 4.2**). Residual analysis is plotting the residuals (the difference between the actual and predicted values) against the predicted values. The plot should show no obvious patterns, indicating that the model is capturing all of the relevant information in the data.

5. Conclusions and Recommendations

Based on the analysis conducted, we can conclude that the linear regression model with a logarithmic response variable is the most suitable for predicting house prices in the real estate market. The model showed good performance on the training and testing datasets and satisfied the assumptions of the linear regression model.

The model coefficients showed that the transaction date and number of convenience stores had a positive impact on house prices, while the age of the house and distance to the nearest MRT station had a negative impact on house prices.

As recommendations, we suggest that real estate agents and homeowners consider the factors that affect house prices in the model when making decisions related to the buying and selling of properties. It is also important to keep in mind that the model is only a representation of the underlying relationship between the variables, and there may be other factors that affect house prices that are not included in the model. Further analysis and research can be done to explore other factors that may influence house prices in the real estate market.

Appendix A: Python Code Output

```
(-0.001, 5.0]      0
(5.0, 10.0]        1
(10.0, 15.0]      14
(15.0, 20.0]      21
(20.0, 25.0]      44
(25.0, 30.0]      46
(30.0, 35.0]      41
(35.0, 40.0]      57
(40.0, 45.0]      70
(45.0, 50.0]      47
(50.0, 55.0]      34
(55.0, 60.0]      20
(60.0, 65.0]      11
(65.0, 70.0]       2
(70.0, 75.0]       3
(75.0, 80.0]       0
(80.0, 85.0]       0
(85.0, 90.0]       0
(90.0, 95.0]       0
(95.0, 100.0]      0
```

Figure 2.1: Frequency distribution of the 'house price of unit area' variable

```
(-0.001, 5.0]      0.00
(5.0, 10.0]        0.24
(10.0, 15.0]       3.41
(15.0, 20.0]       5.11
(20.0, 25.0]      10.71
(25.0, 30.0]      11.19
(30.0, 35.0]       9.98
(35.0, 40.0]      13.87
(40.0, 45.0]      17.03
(45.0, 50.0]      11.44
(50.0, 55.0]       8.27
(55.0, 60.0]       4.87
(60.0, 65.0]       2.68
(65.0, 70.0]       0.49
(70.0, 75.0]       0.73
(75.0, 80.0]       0.00
(80.0, 85.0]       0.00
(85.0, 90.0]       0.00
(90.0, 95.0]       0.00
(95.0, 100.0]      0.00
```

Figure 2.2: Frequency distribution of the 'house price of unit area' variable shown as percentages

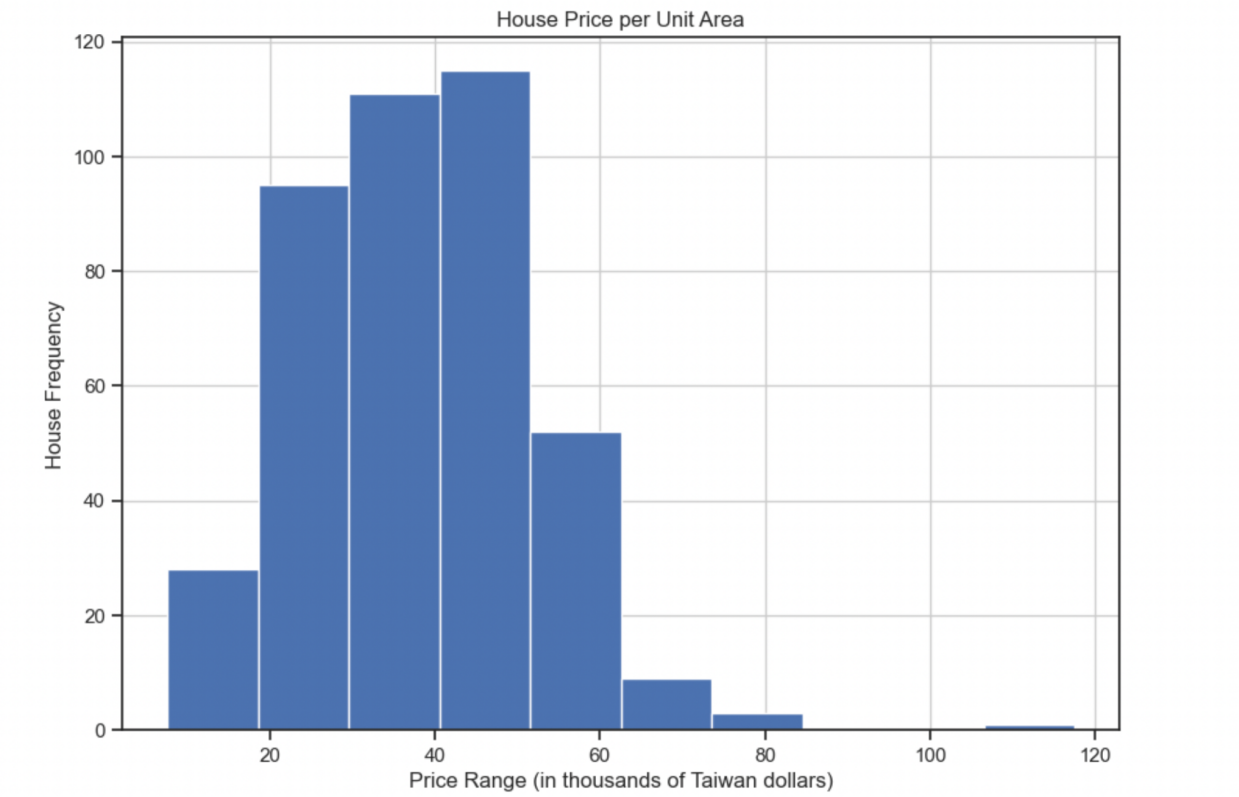

Figure 2.3: Histogram of the 'house price of unit area' variable

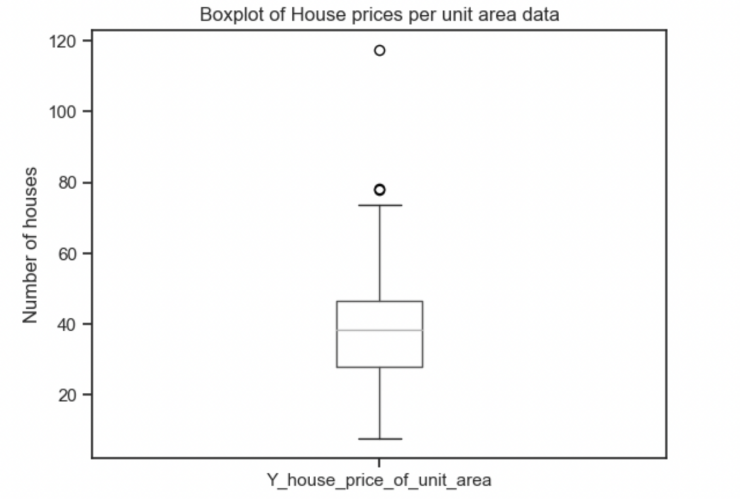

Figure 2.4: Boxplot of House prices per unit area data

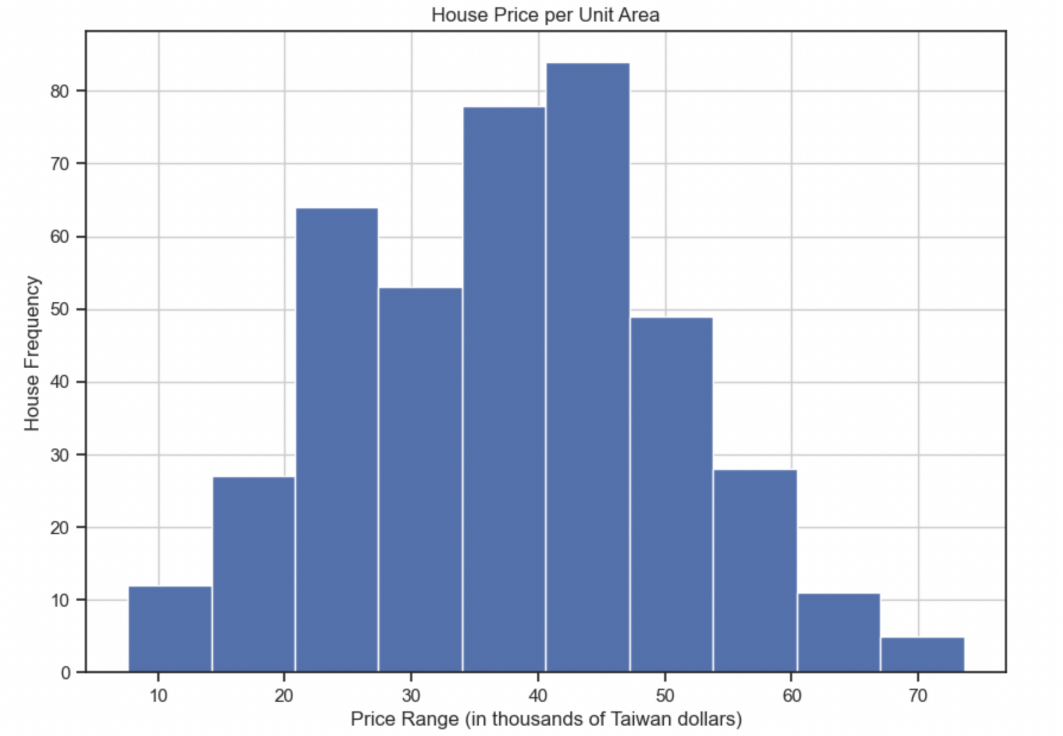

Figure 2.5: Histogram of the 'house price of unit area' variable after removing outliers

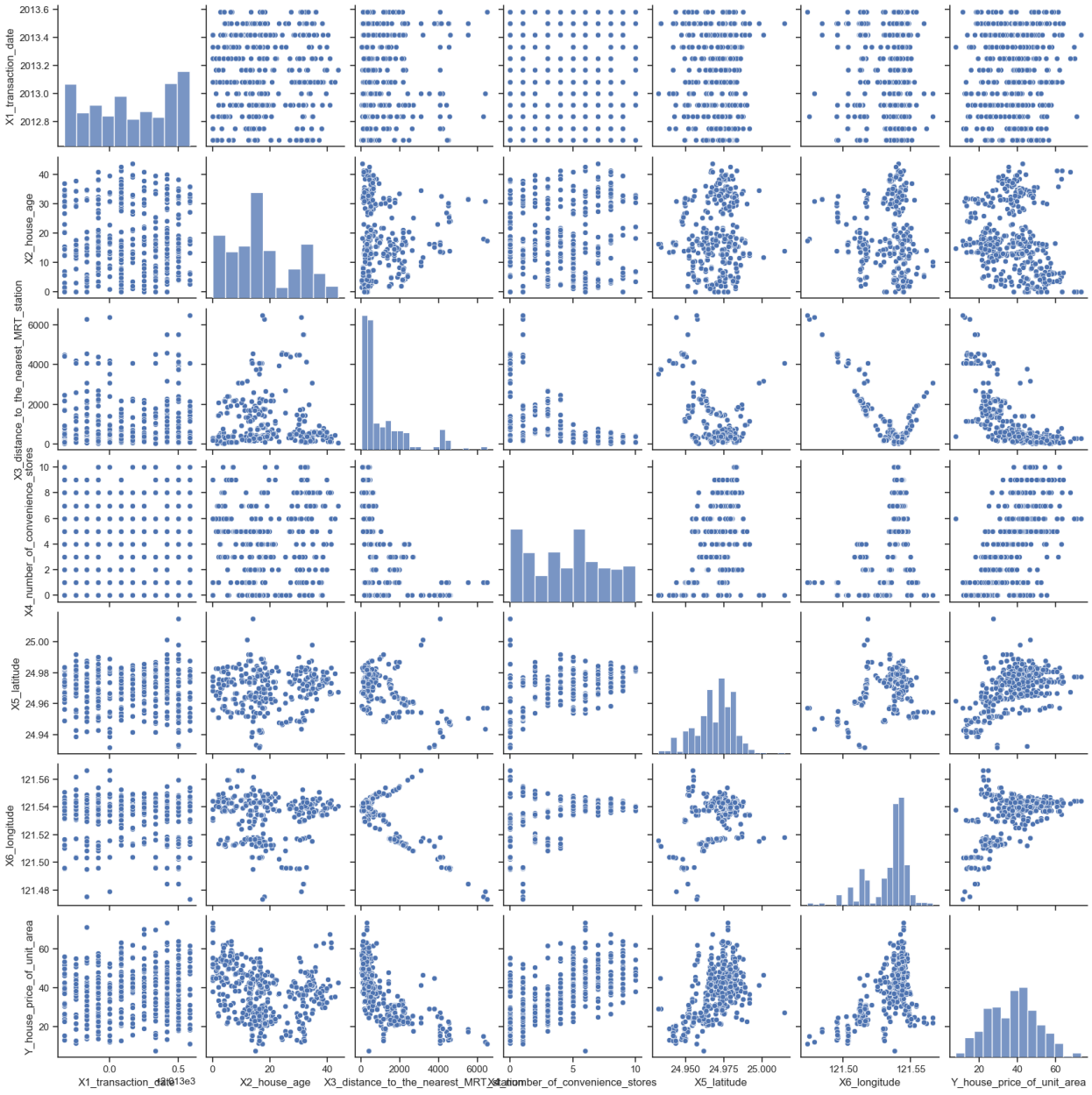

Figure 2.6: Pairwise Scatterplots

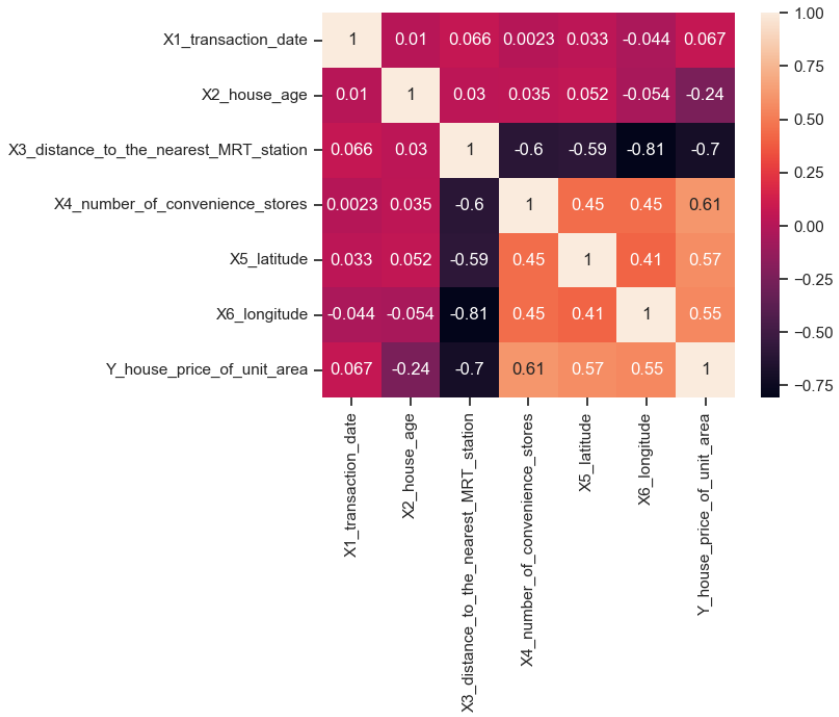

Figure 2.7: Correlation Heatmap

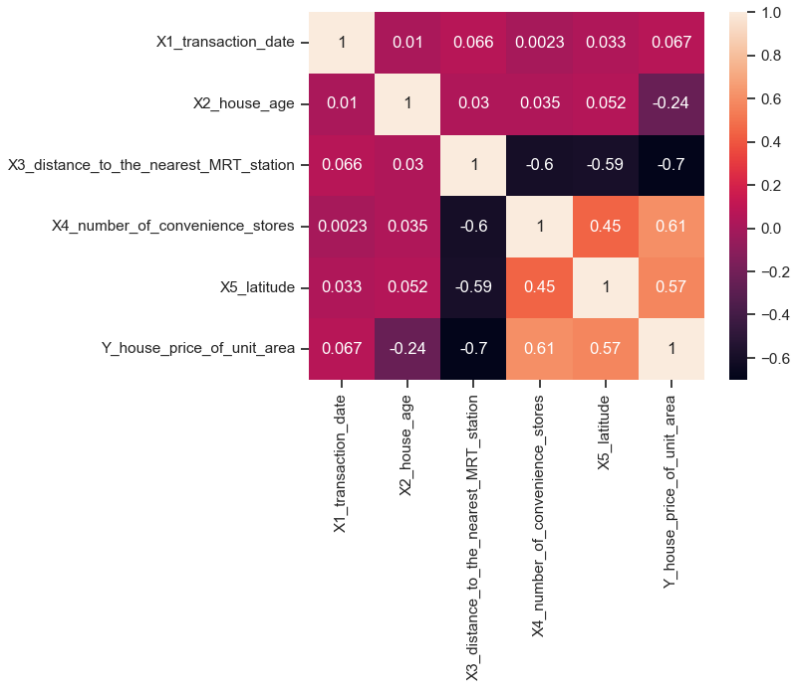

Figure 2.8: Correlation Heatmap after removing column 'X6_longitude'

```
Intercept                        -14989.976129
X1_transaction_date                  2.538906
X2_house_age                        -0.314117
X4_number_of_convenience_stores      1.928892
X5_latitude                        397.054528
dtype: float64
```

Generalized Linear Model Regression Results

Dep. Variable:	Y_house_price_of_unit_area	No. Observations:	411
Model:	GLM	Df Residuals:	406
Model Family:	Gaussian	Df Model:	4
Link Function:	identity	Scale:	72.237
Method:	IRLS	Log-Likelihood:	-1460.2
Date:	Mon, 27 Feb 2023	Deviance:	29328.
Time:	22:23:15	Pearson chi2:	2.93e+04
No. Iterations:	3	Pseudo R-squ. (CS):	0.7174
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-1.499e+04	3112.065	-4.817	0.000	-2.11e+04	-8890.441
X1_transaction_date	2.5389	1.490	1.704	0.088	-0.381	5.459
X2_house_age	-0.3141	0.037	-8.485	0.000	-0.387	-0.242
X4_number_of_convenience_stores	1.9289	0.160	12.058	0.000	1.615	2.242
X5_latitude	397.0545	37.737	10.521	0.000	323.090	471.019

Figure 3.1.1: Normal Model Results - identity link

```
Intercept                        -19199.534823
X1_transaction_date                  4.490857
X2_house_age                        -0.268576
X4_number_of_convenience_stores      1.901224
X5_latitude                        408.239881
dtype: float64
```

Generalized Linear Model Regression Results

Dep. Variable:	Y_house_price_of_unit_area	No. Observations:	411
Model:	GLM	Df Residuals:	406
Model Family:	Gamma	Df Model:	4
Link Function:	identity	Scale:	0.060182
Method:	IRLS	Log-Likelihood:	-1461.7
Date:	Mon, 27 Feb 2023	Deviance:	23.638
Time:	22:23:24	Pearson chi2:	24.4
No. Iterations:	21	Pseudo R-squ. (CS):	0.7180
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-1.92e+04	2949.291	-6.510	0.000	-2.5e+04	-1.34e+04
X1_transaction_date	4.4909	1.403	3.200	0.001	1.740	7.241
X2_house_age	-0.2686	0.038	-7.127	0.000	-0.342	-0.195
X4_number_of_convenience_stores	1.9012	0.179	10.633	0.000	1.551	2.252
X5_latitude	408.2399	34.433	11.856	0.000	340.753	475.726

Figure 3.1.2: Gamma Model Results - identity link

```
                            OLS Regression Results
==============================================================================
Dep. Variable:     Y_house_price_of_unit_area   R-squared:                       0.561
Model:                                    OLS   Adj. R-squared:                  0.557
Method:                         Least Squares   F-statistic:                     129.9
Date:                        Mon, 27 Feb 2023   Prob (F-statistic):           2.66e-71
Time:                                22:23:32   Log-Likelihood:                -1460.2
No. Observations:                         411   AIC:                             2930.
Df Residuals:                             406   BIC:                             2950.
Df Model:                                   4
Covariance Type:                    nonrobust
=================================================================================================
                                    coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------------------------
Intercept                     -1.499e+04   3112.065     -4.817      0.000   -2.11e+04   -8872.203
X1_transaction_date               2.5389      1.490      1.704      0.089      -0.390       5.468
X2_house_age                     -0.3141      0.037     -8.485      0.000      -0.387      -0.241
X4_number_of_convenience_stores   1.9289      0.160     12.058      0.000       1.614       2.243
X5_latitude                     397.0545     37.737     10.521      0.000     322.869     471.240
==============================================================================
Omnibus:                       30.010   Durbin-Watson:                   2.031
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               49.631
Skew:                           0.487   Prob(JB):                     1.67e-11
Kurtosis:                       4.396   Cond. No.                     1.49e+07
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.49e+07. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Figure 3.1.3: Multiple Linear Regression

```
                            OLS Regression Results
==============================================================================
Dep. Variable:     log_Y_house_price_of_unit_area   R-squared:                       0.694
Model:                                    OLS   Adj. R-squared:                  0.691
Method:                         Least Squares   F-statistic:                     230.1
Date:                        Mon, 27 Feb 2023   Prob (F-statistic):          6.01e-103
Time:                                22:24:01   Log-Likelihood:                 52.465
No. Observations:                         411   AIC:                            -94.93
Df Residuals:                             406   BIC:                            -74.84
Df Model:                                   4
Covariance Type:                    nonrobust
=================================================================================================
                                    coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------------------------
Intercept                      -201.1918     26.646     -7.551      0.000    -253.572    -148.811
X2_house_age                     -0.0073      0.001     -7.755      0.000      -0.009      -0.005
X3_distance_to_the_nearest_MRT_station    -0.0001   1.18e-05    -11.982      0.000      -0.000      -0.000
X4_number_of_convenience_stores   0.0285      0.005      6.212      0.000       0.019       0.037
X5_latitude                       8.2069      1.067      7.691      0.000       6.109      10.305
==============================================================================
Omnibus:                      116.415   Durbin-Watson:                   2.038
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1640.884
Skew:                          -0.773   Prob(JB):                         0.00
Kurtosis:                      12.666   Cond. No.                     4.21e+06
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 4.21e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Figure 3.1.4: Multiple Linear Regression with log transformed response variable

Generalized Linear Model Regression Results

Dep. Variable:	log_Y_house_price_of_unit_area	No. Observations:	411
Model:	GLM	Df Residuals:	406
Model Family:	Gamma	Df Model:	4
Link Function:	identity	Scale:	0.0039729
Method:	IRLS	Log-Likelihood:	27.123
Date:	Mon, 27 Feb 2023	Deviance:	1.6834
Time:	22:24:38	Pearson chi2:	1.61
No. Iterations:	10	Pseudo R-squ. (CS):	0.8845
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-200.1797	27.199	-7.360	0.000	-253.488	-146.871
X2_house_age	-0.0069	0.001	-7.010	0.000	-0.009	-0.005
X3_distance_to_the_nearest_MRT_station	-0.0001	1.1e-05	-11.902	0.000	-0.000	-0.000
X4_number_of_convenience_stores	0.0305	0.005	6.301	0.000	0.021	0.040
X5_latitude	8.1653	1.089	7.496	0.000	6.030	10.300

Figure 3.1.5: Gamma GLM with log transformed response variable

	Columns	Coefficients
0	X1_transaction_date	0.103859
1	X2_house_age	-0.007510
2	X3_distance_to_the_nearest_MRT_station	-0.000166
3	X4_number_of_convenience_stores	0.025843
4	X5_latitude	8.003622

Figure 4.1: Multiple Linear Regression Model Coefficients

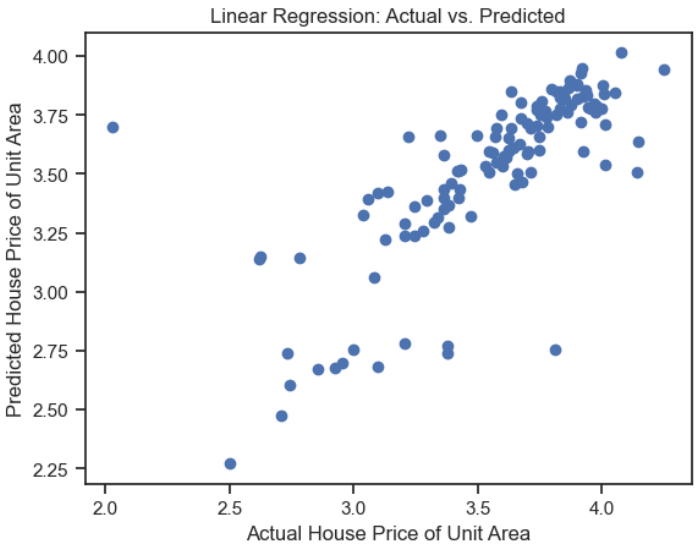

Figure 4.2: Predicted values against the actual values

Appendix B: Python Code

```
# import libraries

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

import statsmodels.formula.api as smf

from sklearn.linear_model import LinearRegression

from sklearn.model_selection import train_test_split



# Load the Excel file into a Pandas DataFrame

real_estate = pd.read_excel('Real_estate_valuation_data_set.xlsx')

df_real_estate_data_file = pd.DataFrame(real_estate)

real_estate



# Check for NaN values, null or missing values

print('The number of NaN values in each column of the dataset:\n', df_real_estate_data_file.isnull().sum())

print()



# check for null values

print('The number of null values in each column of the dataset:\n', df_real_estate_data_file.isna().sum())

print()



# check for missing values

print('Total number of missing values in the entire dataset =\n', df_real_estate_data_file.isnull().sum().sum())
```

```
# The 'No' column is not relevant for our analysis so we can drop it.

df_real_estate_data_file.drop('No', axis=1, inplace=True)


list(df_real_estate_data_file.columns)


# Rename the columns to remove the prefix

df_real_estate_data_file.columns = df_real_estate_data_file.columns.str.replace(' ', '_')


# Print the columns of the DataFrame

print(list(df_real_estate_data_file.columns))


# Construct a frequency distribution of the 'house price of unit area' variable

bins = [0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100]


freq_dis_real_estate = df_real_estate_data_file['Y_house_price_of_unit_area'].value_counts(bins=bins).sort_index()

freq_dis_real_estate_data_percentages = round((df_real_estate_data_file['Y_house_price_of_unit_area'].value_counts(bins=bins,
normalize=True).sort_index()*100), 2)


# Print the frequency distribution

print(freq_dis_real_estate)

print()

print(freq_dis_real_estate_data_percentages)



# Construct a histogram of the 'house price of unit area' variable

fig, ax = plt.subplots(figsize = (10, 7))
```

```
# Set the grid = True, to get a better sense of the values.

df_real_estate_data_file['Y_house_price_of_unit_area'].hist(grid = True)

plt.title('House Price per Unit Area')

plt.xlabel('Price Range (in thousands of Taiwan dollars)')

plt.ylabel('House Frequency')

plt.show()




import numpy as np

# Mean, Median, Standard Deviation

mean = np.mean(df_real_estate_data_file['Y_house_price_of_unit_area'])

print('Mean = %.2f'% mean)


median = np.median(df_real_estate_data_file['Y_house_price_of_unit_area'])

print('Median = %.2f'% median)


standard_deviation = np.std(df_real_estate_data_file['Y_house_price_of_unit_area'])

print('Standard_deviation = %.2f' % standard_deviation)


# Box plot construction

real_estate_bp = pd.DataFrame(df_real_estate_data_file['Y_house_price_of_unit_area'])

real_estate_bp.boxplot(grid = False)

# plot title

plt.title ('Boxplot of House prices per unit area data')

# y-axis label

plt.ylabel('Number of houses')
```

```
plt.show()


# Calculate z-scores for each data point

df_real_estate_data_file['z_score'] = (df_real_estate_data_file['Y_house_price_of_unit_area'] - mean) / standard_deviation


# Identify outliers with z-score greater than 3 or -3

outliers_z = df_real_estate_data_file.loc[(df_real_estate_data_file['z_score'] > 3) | (df_real_estate_data_file['z_score'] < -3)]


print('Outliers detected by z-score method:\n')

print(outliers_z)


df_real_estate_data_file = df_real_estate_data_file.drop('z_score', axis = 1)


lower_quantile = np.quantile(df_real_estate_data_file['Y_house_price_of_unit_area'], .25)

upper_quantile = np.quantile(df_real_estate_data_file['Y_house_price_of_unit_area'], .75)


iqr = upper_quantile - lower_quantile


# Identify outliers using IQR method

outliers_iqr = df_real_estate_data_file.loc[(df_real_estate_data_file['Y_house_price_of_unit_area'] < lower_quantile - 1.5 * iqr) |
(df_real_estate_data_file['Y_house_price_of_unit_area'] > upper_quantile + 1.5 * iqr)]

print('Outliers detected by IQR method:')

print(outliers_iqr)


# Remove outliers which are not in the IQR

df_real_estate_data_file =
df_real_estate_data_file.loc[~df_real_estate_data_file['Y_house_price_of_unit_area'].isin(outliers_iqr['Y_house_price_of_unit_area'])]
```

```
# Construct a histogram of the 'house price of unit area' variable after removing outliers.

fig, ax = plt.subplots(figsize = (10, 7))

# Set the grid = True, to get a better sense of the values.

df_real_estate_data_file['Y_house_price_of_unit_area'].hist(grid = True)

plt.title('House Price per Unit Area')

plt.xlabel('Price Range (in thousands of Taiwan dollars)')

plt.ylabel('House Frequency')

plt.show()


# The five number summary includes the median, the upper and lower quartiles, and the maximum and minimum values.

# Below we can see all of the above concluding the five number summary.

median = np.median(df_real_estate_data_file['Y_house_price_of_unit_area'])

lower_quantile = np.quantile(df_real_estate_data_file['Y_house_price_of_unit_area'], .25)

upper_quantile = np.quantile(df_real_estate_data_file['Y_house_price_of_unit_area'], .75)

max_value = df_real_estate_data_file['Y_house_price_of_unit_area'].max()

min_value = df_real_estate_data_file['Y_house_price_of_unit_area'].min()


print('Median = %.2f'% median)

print('Lower_quantile = %.2f'% lower_quantile)

print('Upper_quantile = %.2f'% upper_quantile)

print('Max_value = %.2f'% max_value)

print('Min_value = %.2f'% min_value)


# Scatterplot

sns.set(style = "ticks")
```

```
sns.pairplot(df_real_estate_data_file)


df_real_estate_data_file.corr()


# Correlation
corr = df_real_estate_data_file.corr()
sns.heatmap(corr, annot=True)
plt.show()


# Dropping X6_longitude as it is highly correlated with other predictors.
df_real_estate_data_file = df_real_estate_data_file.drop('X6_longitude', axis = 1)


# Checking the correlation again using heatmap
corr = df_real_estate_data_file.corr()
sns.heatmap(corr, annot=True)
plt.show()


# GLM assuming normal response, identity link
import statsmodels.api as sm
import statsmodels.formula.api as smf
formula = 'Y_house_price_of_unit_area ~ X1_transaction_date + X2_house_age  + X4_number_of_convenience_stores + X5_latitude'
fit_real_estate_normal = smf.glm(formula = formula, data = df_real_estate_data_file, family = sm.families.Gaussian()).fit()
print(fit_real_estate_normal.params)
print(fit_real_estate_normal.summary())
# Normal GLM
print("Normal model AIC = %.2f" %fit_real_estate_normal.aic)
```

```
print("Normal model BIC = %.2f" %fit_real_estate_normal.bic)


# GLM assuming gamma response, identity link

identity_link = sm.families.links.identity()

gamma_dist = sm.families.Gamma(link = identity_link)

gamma_mod = smf.glm(formula = formula, data = df_real_estate_data_file, family = gamma_dist)

fit_real_estate_gamma = gamma_mod.fit()

print(fit_real_estate_gamma.params)

print(fit_real_estate_gamma.summary())

# Gamma GLM

print("Gamma model AIC = %.2f" %fit_real_estate_gamma.aic)

print("Gamma model BIC = %.2f" %fit_real_estate_gamma.bic)


# Fitting the linear model using the explanatory variables

fit_real_estate = smf.ols(formula = formula, data = df_real_estate_data_file).fit()

print(fit_real_estate.summary())

print("Linear regression model AIC = %.2f" %fit_real_estate.aic)

print("Linear regression model BIC = %.2f" %fit_real_estate.bic)


df_real_estate_data_file['log_Y_house_price_of_unit_area'] = np.log(df_real_estate_data_file['Y_house_price_of_unit_area'])


# Define the formula for the linear model using the log-transformed variable

formula_log = 'log_Y_house_price_of_unit_area ~ X2_house_age + X3_distance_to_the_nearest_MRT_station +
X4_number_of_convenience_stores + X5_latitude'


# Fit the linear model using the log-transformed variable
```

```
fit_real_estate_log = smf.ols(formula=formula_log, data=df_real_estate_data_file).fit()


# Print the summary of the fitted model

print(fit_real_estate_log.summary())

print("Linear regression model (log transformed) AIC = %.2f" %fit_real_estate_log.aic)

print("Linear regression model (log transformed) BIC = %.2f" %fit_real_estate_log.bic)


# GLM assuming gamma response, identity link

identity_link = sm.families.links.identity()

gamma_dist = sm.families.Gamma(link = identity_link)

gamma_mod = smf.glm(formula = formula_log, data = df_real_estate_data_file, family = gamma_dist)

fit_real_estate_gamma_log = gamma_mod.fit()


print(fit_real_estate_gamma.summary())

print("Gamma model (log transformed) AIC = %.2f" %fit_real_estate_gamma_log.aic)

print("Gamma model (log transformed) BIC = %.2f" %fit_real_estate_gamma_log.bic)


df_real_estate_data_file_log = df_real_estate_data_file.drop('Y_house_price_of_unit_area', axis = 1)

df_real_estate_data_file_no_log = df_real_estate_data_file.drop('log_Y_house_price_of_unit_area', axis = 1)

# assign independent variables, dependent variable, respectively

x = df_real_estate_data_file_log.loc[:, df_real_estate_data_file_log.columns != 'log_Y_house_price_of_unit_area']

# define the target

y = pd.DataFrame(df_real_estate_data_file_log['log_Y_house_price_of_unit_area'])

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3,

                                    random_state=42)

sk_model = LinearRegression()
```

```
sk_model.fit(x_train, y_train)

print('Model Coefficients:', sk_model.coef_)

print()

print('Intercept:', sk_model.intercept_)

coef = pd.DataFrame(sk_model.coef_.T, columns=['Coefficients'])

cols = pd.DataFrame(x.columns, columns=['Columns'])

joined = pd.concat([cols['Columns'], coef['Coefficients']], axis=1)

joined

predictions = sk_model.predict(x_test)

pd.DataFrame(predictions, columns=['Predictions'])

train_score = sk_model.score(x_train, y_train)

test_score = sk_model.score(x_test, y_test)


n = len(y_train)

p = len(x_train.columns)


adj_train_score = 1 - (1 - train_score) * ((n - 1) / (n - p - 1))

adj_test_score = 1 - (1 - test_score) * ((n - 1) / (n - p - 1))


print('R-squared (train): %.2f' %train_score)

print('Adjusted R-squared (train): %.2f' %adj_train_score)

print('R-squared (test): %.2f' %test_score)

print('Adjusted R-squared (test): %.2f' %adj_test_score)

# Plot the predicted values against the actual values

plt.scatter(y_test, predictions)

plt.xlabel('Actual House Price of Unit Area')
```

```
plt.ylabel('Predicted House Price of Unit Area')

plt.title('Linear Regression: Actual vs. Predicted')

plt.show()
```