Business Report

Capstone Project

House Price Prediction

PGPDSBA

Chithira Raj

Table of Contents

List of Tables

List of Figures

1. Introduction of the business problem

1.1 Problem Statement

Estimating the value of a house goes beyond just its location and square footage—many factors contribute to its market price. Homeowners looking to sell their property often face challenges in determining an appropriate asking price. Setting the price too low may result in a financial loss, while pricing it too high could drive away potential buyers.

Traditionally, homeowners and real estate agents compare similar properties in the neighborhood to estimate a house's price. However, this approach may be subjective and inconsistent, as it does not account for a comprehensive set of factors such as house condition, renovations, number of rooms, scenic views, and market trends.

The dataset used in this study pertains to residential house sales in King County, Washington, which includes Seattle and its surrounding areas, during the period of 2014–2015. During this time, the Seattle housing market was experiencing a notable surge in both demand and property values, driven by rapid urban development, population growth, and a booming tech industry. As part of the larger U.S. housing market, which saw a 34% increase in home sales over the past decade—reaching a record high of 5.51 million units—the Seattle area mirrored this upward trend. With rising incomes and improved living standards, the demand for residential housing in metropolitan areas like Seattle significantly increased.

Given this context, house price prediction has attracted widespread attention because predictive insights can help various stakeholders make informed decisions. The objective of this study is to leverage data-driven techniques to analyze all available house features and develop a predictive model for house prices. By doing so, we aim to provide a more accurate, transparent, and efficient pricing strategy for homeowners, buyers, and real estate professionals.

1.2 Need of the study/project

Accurate house price estimation is crucial for various stakeholders, including homeowners, buyers, real estate agents, and financial institutions. Traditionally, property pricing relies on comparing similar houses in the same neighborhood. However, this approach fails to consider multiple influencing factors, leading to inaccurate pricing and potential financial losses.

1. For Homeowners & Sellers

   - Helps set a competitive and fair asking price.
   - Reduces the risk of overpricing, which can delay sales, or underpricing, which leads to financial loss.

2. For Buyers & Investors

   - Allows buyers to assess the true value of a property before making an offer.
   - Helps investors evaluate potential appreciation and return on investment.

3. For Real Estate Agents & Market Analysts

   - Provides data-driven insights for better pricing recommendations.
   - Helps identify market trends and price fluctuations based on property attributes.

4. For Banks & Financial Institutions

   - Enables better risk assessment for mortgage approvals.
   - Helps banks determine accurate collateral values for home loans.

1.3 Understanding business/social opportunity

1. Business Opportunities

- Improved Market Efficiency → Reduces pricing discrepancies by providing accurate property valuations.
- Better Investment Decisions → Investors can analyze historical price trends and identify high-growth areas.
- Optimized Real Estate Listings → Agents and sellers can set realistic prices, improving the sales cycle.
- Enhanced Mortgage Risk Assessment → Banks can accurately evaluate a property's worth for loan approvals.

2. Social Impact

- Increased Housing Affordability Awareness → Buyers can identify affordable neighborhoods and make informed decisions.
- Reduced Market Speculation → Prevents property overvaluation and real estate bubbles.
- Fair Pricing and Transparency → Promotes ethical real estate transactions by reducing pricing manipulation.
- Encourages Sustainable Development → Helps policymakers understand housing demand and plan better infrastructure.

2. Data Report

2.1. Data Dictionary

S.No.	Variables	Description
1	cid	a notation for a house
2	dayhours	Date house was sold
3	price	Price is prediction target
4	room_bed	Number of Bedrooms/House
5	room_bath	Number of bathrooms/bedrooms
6	living_measure	square footage of the home
7	lot_measure	square footage of the lot
8	ceil	Total floors (levels) in house
9	coast	House which has a view to a waterfront
10	sight	Has been viewed
11	condition	How good the condition is (Overall)
12	quality	grade given to the housing unit, based on grading system
13	ceil_measure	square footage of house apart from basement
14	basement	square footage of the basement
15	yr_built	Built Year
16	yr_renovated	Year when house was renovated
17	zipcode	zip
18	lat	Latitude coordinate
19	long	Longitude coordinate
20	living_measure15	Living room area in 2015(implies-- some renovations) This might or might not have affected the lotsize area
21	lot_measure15	lotSize area in 2015(implies-- some renovations)
22	furnished	Based on the quality of room
23	total_area	Measure of both living and lot

Table 1: Data Dictionary

2.2. Understanding how data was collected in terms of time, frequency, and methodology

- Timeframe of Data Collection

  The dataset spans from May 2, 2014, to May 27, 2015 (about 13 months).

- Frequency of Data Collection

  The number of house sales varies by month:

  - May 2014: 1,768 sales
  - June 2014: 2,180 sales
  - July 2014: 2,211 sales
  - August 2014: 1,940 sales
  - September 2014: 1,774 sales

  This suggests that data was recorded daily but aggregated on a monthly basis.

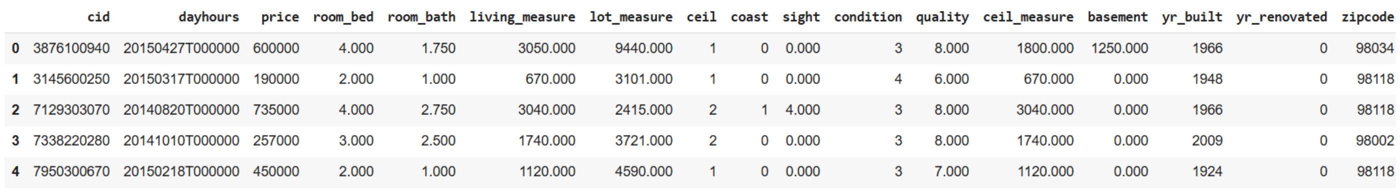
- Methodology of Data Collection

  Likely sourced from real estate listings, government property records, or MLS (Multiple Listing Service).

  Includes transaction history with details on property characteristics and geographic location.

2.3. Visual inspection of data (rows, columns, descriptive details)

2.3.1. Import libraries and load the data



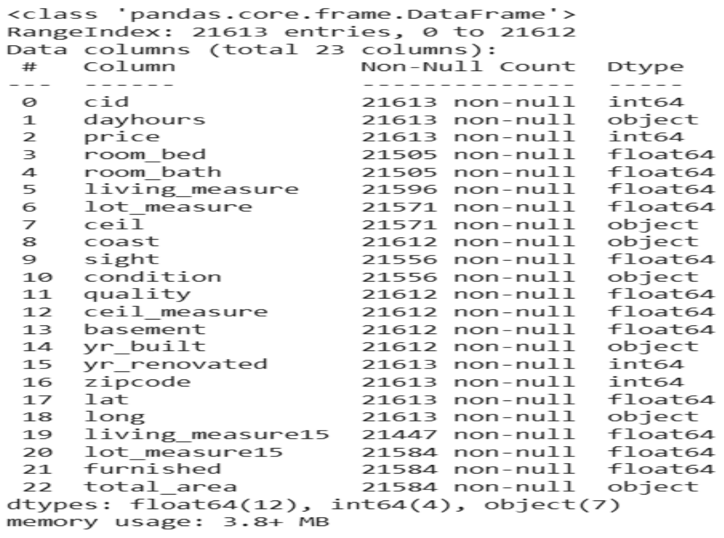
	cid	dayhours	price	room_bed	room_bath	living_measure	lot_measure	ceil	coast	sight	condition	quality	ceil_measure	basement	yr_built	yr_renovated	zipcode
0	3876100940	20150427T000000	600000	4.000	1.750	3050.000	9440.000	1	0	0.000	3	8.000	1800.000	1250.000	1966	0	98034
1	3145600250	20150317T000000	190000	2.000	1.000	670.000	3101.000	1	0	0.000	4	6.000	670.000	0.000	1948	0	98118
2	7129303070	20140820T000000	735000	4.000	2.750	3040.000	2415.000	2	1	4.000	3	8.000	3040.000	0.000	1966	0	98118
3	7338220280	20141010T000000	257000	3.000	2.500	1740.000	3721.000	2	0	0.000	3	8.000	1740.000	0.000	2009	0	98002
4	7950300670	20150218T000000	450000	2.000	1.000	1120.000	4590.000	1	0	0.000	3	7.000	1120.000	0.000	1924	0	98118

Figure 1: Data Overview

2.3.2. Check the structure of data

Shape of the dataset: 21613 rows and 13 columns

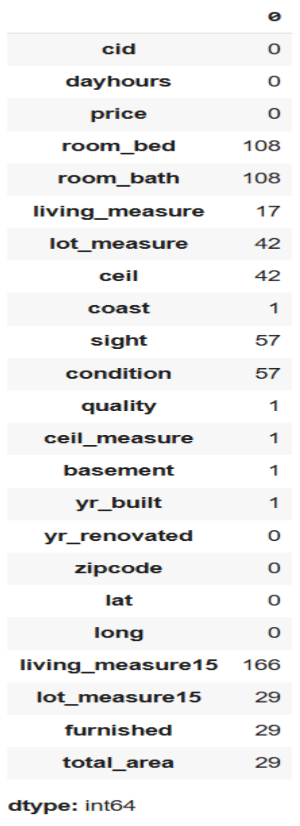
2.3.3. Check the types of the data



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21613 entries, 0 to 21612
Data columns (total 23 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   cid               21613 non-null  int64
 1   dayhours          21613 non-null  object
 2   price             21613 non-null  int64
 3   room_bed          21505 non-null  float64
 4   room_bath         21505 non-null  float64
 5   living_measure    21596 non-null  float64
 6   lot_measure       21571 non-null  float64
 7   ceil              21571 non-null  object
 8   coast             21612 non-null  object
 9   sight             21556 non-null  float64
 10  condition         21556 non-null  object
 11  quality           21612 non-null  float64
 12  ceil_measure      21612 non-null  float64
 13  basement          21612 non-null  float64
 14  yr_built          21612 non-null  object
 15  yr_renovated      21613 non-null  int64
 16  zipcode           21613 non-null  int64
 17  lat               21613 non-null  float64
 18  long              21613 non-null  object
 19  living_measure15  21447 non-null  float64
 20  lot_measure15     21584 non-null  float64
 21  furnished         21584 non-null  float64
 22  total_area        21584 non-null  object
dtypes: float64(12), int64(4), object(7)
memory usage: 3.8+ MB
```

Figure 2: Datatypes

2.3.4. Missing Values



	0
cid	0
dayhours	0
price	0
room_bed	108
room_bath	108
living_measure	17
lot_measure	42
ceil	42
coast	1
sight	57
condition	57
quality	1
ceil_measure	1
basement	1
yr_built	1
yr_renovated	0
zipcode	0
lat	0
long	0
living_measure15	166
lot_measure15	29
furnished	29
total_area	29

dtype: int64

Figure 3: Missing values check

2.3.5. Data Duplicates

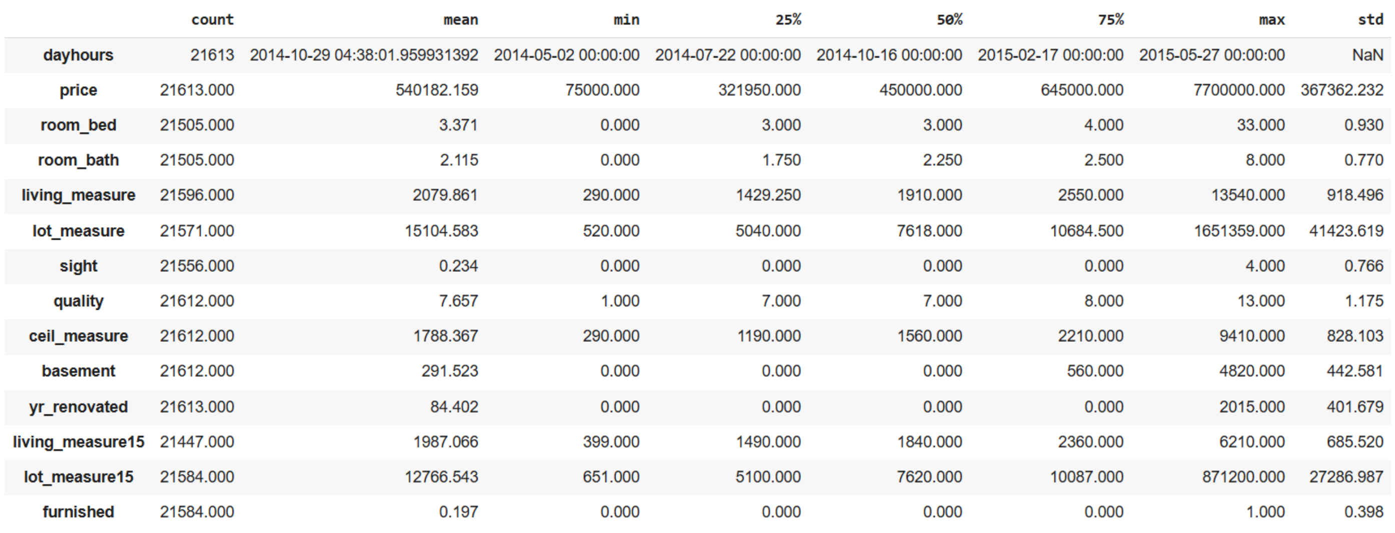
There are no duplicate rows.

2.3.6. Eliminate Irrelevant columns

The cid column consists of system-generated IDs, making it irrelevant for a prediction model.

Similarly, zipcode, latitude, and longitude represent spatial characteristics, which are not useful for a regression model.

2.3.7. Statistical Summary



	count	mean	min	25%	50%	75%	max	std
dayhours	21613	2014-10-29 04:38:01.959931392	2014-05-02 00:00:00	2014-07-22 00:00:00	2014-10-16 00:00:00	2015-02-17 00:00:00	2015-05-27 00:00:00	NaN
price	21613.000	540182.159	75000.000	321950.000	450000.000	645000.000	7700000.000	367362.232
room_bed	21505.000	3.371	0.000	3.000	3.000	4.000	33.000	0.930
room_bath	21505.000	2.115	0.000	1.750	2.250	2.500	8.000	0.770
living_measure	21596.000	2079.861	290.000	1429.250	1910.000	2550.000	13540.000	918.496
lot_measure	21571.000	15104.583	520.000	5040.000	7618.000	10684.500	1651359.000	41423.619
sight	21556.000	0.234	0.000	0.000	0.000	0.000	4.000	0.766
quality	21612.000	7.657	1.000	7.000	7.000	8.000	13.000	1.175
ceil_measure	21612.000	1788.367	290.000	1190.000	1560.000	2210.000	9410.000	828.103
basement	21612.000	291.523	0.000	0.000	0.000	560.000	4820.000	442.581
yr_renovated	21613.000	84.402	0.000	0.000	0.000	0.000	2015.000	401.679
living_measure15	21447.000	1987.066	399.000	1490.000	1840.000	2360.000	6210.000	685.520
lot_measure15	21584.000	12766.543	651.000	5100.000	7620.000	10087.000	871200.000	27286.987
furnished	21584.000	0.197	0.000	0.000	0.000	0.000	1.000	0.398

Figure 4: Statistical Summary

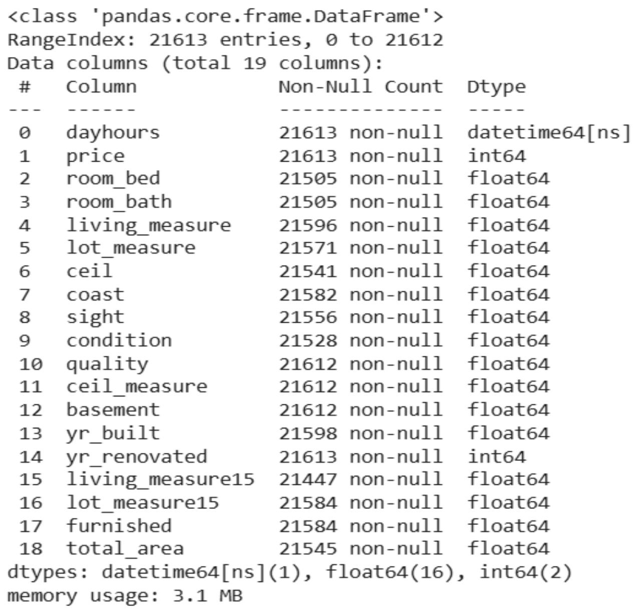
Insights

- The dayhours column contains datetime values, which might require conversion for analysis.
- Missing values are present as the count of observations varies across columns.
- price has a high standard deviation (~367,362), indicating a wide range of values and possible outliers.
- lot_measure and lot_measure15 have large maximum values compared to their 75th percentile, suggesting potential extreme values.
- The furnished column has a mean of 0.197, indicating that a small proportion of properties are furnished.

2.4. Data Preprocessing

2.4.1. Data Cleaning

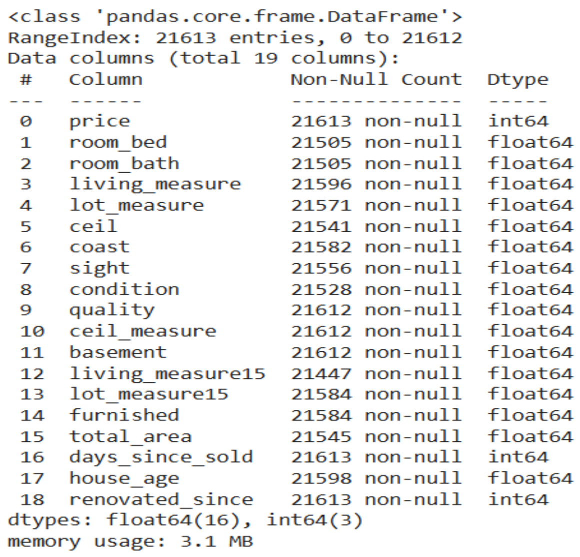
- Some columns contain unknown values, such as $. We can initially replace these with null values and then apply appropriate missing value treatment.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21613 entries, 0 to 21612
Data columns (total 19 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   dayhours         21613 non-null  datetime64[ns]
 1   price            21613 non-null  int64
 2   room_bed         21505 non-null  float64
 3   room_bath        21505 non-null  float64
 4   living_measure   21596 non-null  float64
 5   lot_measure      21571 non-null  float64
 6   ceil             21541 non-null  float64
 7   coast            21582 non-null  float64
 8   sight            21556 non-null  float64
 9   condition        21528 non-null  float64
 10  quality          21612 non-null  float64
 11  ceil_measure     21612 non-null  float64
 12  basement         21612 non-null  float64
 13  yr_built         21598 non-null  float64
 14  yr_renovated     21613 non-null  int64
 15  living_measure15 21447 non-null  float64
 16  lot_measure15    21584 non-null  float64
 17  furnished        21584 non-null  float64
 18  total_area       21545 non-null  float64
dtypes: datetime64[ns](1), float64(16), int64(2)
memory usage: 3.1 MB
```

Figure 5: Dataframe info after removing values

- Convert date/year columns such as dayhours, yr_built, and yr_renovated into the number of days or years passed since today. After deriving these new columns, drop the original ones.

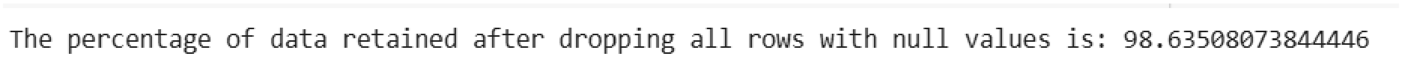


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21613 entries, 0 to 21612
Data columns (total 19 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   price             21613 non-null  int64
 1   room_bed          21505 non-null  float64
 2   room_bath         21505 non-null  float64
 3   living_measure    21596 non-null  float64
 4   lot_measure       21571 non-null  float64
 5   ceil              21541 non-null  float64
 6   coast             21582 non-null  float64
 7   sight             21556 non-null  float64
 8   condition         21528 non-null  float64
 9   quality           21612 non-null  float64
 10  ceil_measure      21612 non-null  float64
 11  basement          21612 non-null  float64
 12  living_measure15  21447 non-null  float64
 13  lot_measure15     21584 non-null  float64
 14  furnished         21584 non-null  float64
 15  total_area        21545 non-null  float64
 16  days_since_sold   21613 non-null  int64
 17  house_age         21598 non-null  float64
 18  renovated_since   21613 non-null  int64
dtypes: float64(16), int64(3)
memory usage: 3.1 MB
```

Figure 6: Dataframe info after deriving columns

2.4.2. Missing Value Treatment

- Remove all rows containing null values.

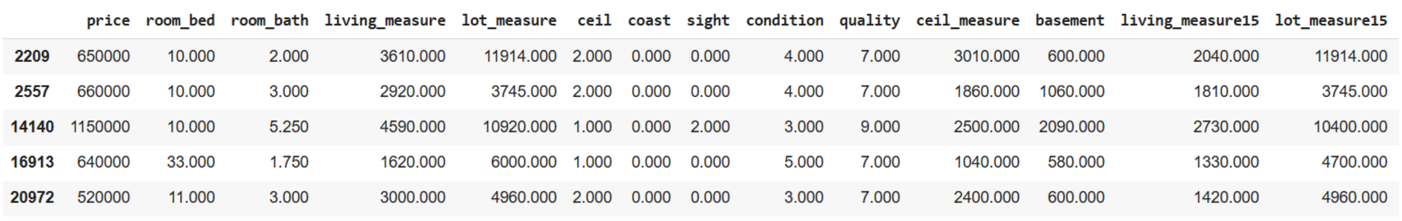


```
The percentage of data retained after dropping all rows with null values is: 98.63508073844446
```

Figure 7: data after dropping null values

- Only 1.36% of the data is lost after dropping rows with missing values, it is an acceptable trade-off to ensure data quality and consistency.
- Since the loss is minimal (only ~1.36%), it is reasonable to drop the rows instead of imputing missing values, which might introduce bias or incorrect assumptions.

2.4.3. Outlier Check



	price	room_bed	room_bath	living_measure	lot_measure	ceil	coast	sight	condition	quality	ceil_measure	basement	living_measure15	lot_measure15
2209	650000	10.000	2.000	3610.000	11914.000	2.000	0.000	0.000	4.000	7.000	3010.000	600.000	2040.000	11914.000
2557	660000	10.000	3.000	2920.000	3745.000	2.000	0.000	0.000	4.000	7.000	1860.000	1060.000	1810.000	3745.000
14140	1150000	10.000	5.250	4590.000	10920.000	1.000	0.000	2.000	3.000	9.000	2500.000	2090.000	2730.000	10400.000
16913	640000	33.000	1.750	1620.000	6000.000	1.000	0.000	0.000	5.000	7.000	1040.000	580.000	1330.000	4700.000
20972	520000	11.000	3.000	3000.000	4960.000	2.000	0.000	0.000	3.000	7.000	2400.000	600.000	1420.000	4960.000

Figure 8: Data Rows with bedrooms >=10

- Most residential houses have 3 to 5 bedrooms.
- A house with 10+ bedrooms is extremely rare and likely a data entry error.
- Removing them ensures a cleaner dataset for training.

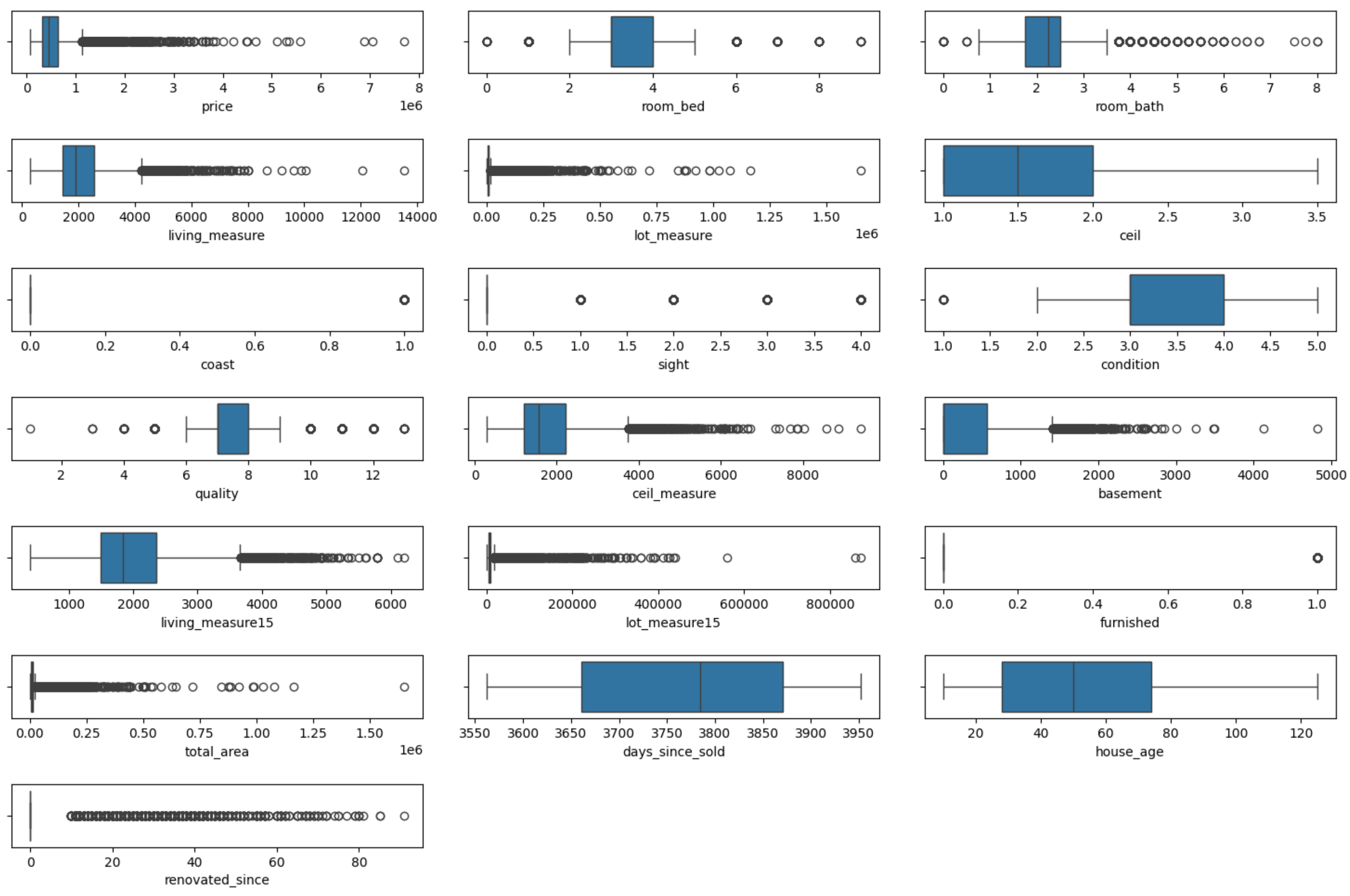
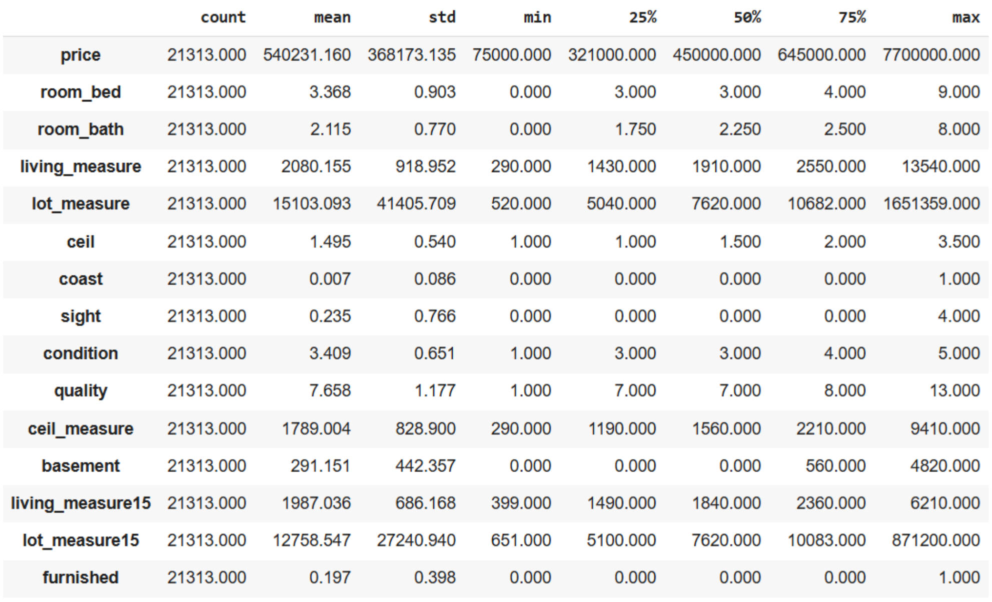


Figure 9: Boxplots with Outliers

However, not all outliers should be removed because some might represent actual, valid data points rather than errors.

2.4.4. Statistical Summary



	count	mean	std	min	25%	50%	75%	max
price	21313.000	540231.160	368173.135	75000.000	321000.000	450000.000	645000.000	7700000.000
room_bed	21313.000	3.368	0.903	0.000	3.000	3.000	4.000	9.000
room_bath	21313.000	2.115	0.770	0.000	1.750	2.250	2.500	8.000
living_measure	21313.000	2080.155	918.952	290.000	1430.000	1910.000	2550.000	13540.000
lot_measure	21313.000	15103.093	41405.709	520.000	5040.000	7620.000	10682.000	1651359.000
ceil	21313.000	1.495	0.540	1.000	1.000	1.500	2.000	3.500
coast	21313.000	0.007	0.086	0.000	0.000	0.000	0.000	1.000
sight	21313.000	0.235	0.766	0.000	0.000	0.000	0.000	4.000
condition	21313.000	3.409	0.651	1.000	3.000	3.000	4.000	5.000
quality	21313.000	7.658	1.177	1.000	7.000	7.000	8.000	13.000
ceil_measure	21313.000	1789.004	828.900	290.000	1190.000	1560.000	2210.000	9410.000
basement	21313.000	291.151	442.357	0.000	0.000	0.000	560.000	4820.000
living_measure15	21313.000	1987.036	686.168	399.000	1490.000	1840.000	2360.000	6210.000
lot_measure15	21313.000	12758.547	27240.940	651.000	5100.000	7620.000	10083.000	871200.000
furnished	21313.000	0.197	0.398	0.000	0.000	0.000	0.000	1.000

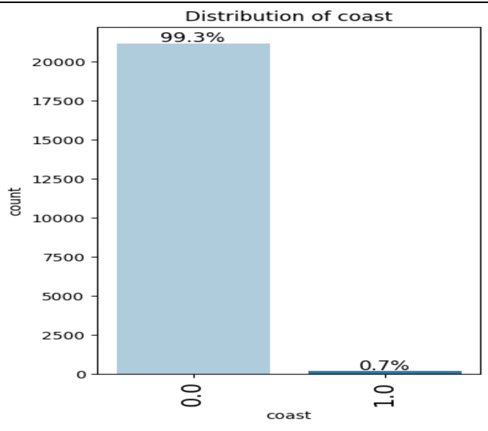
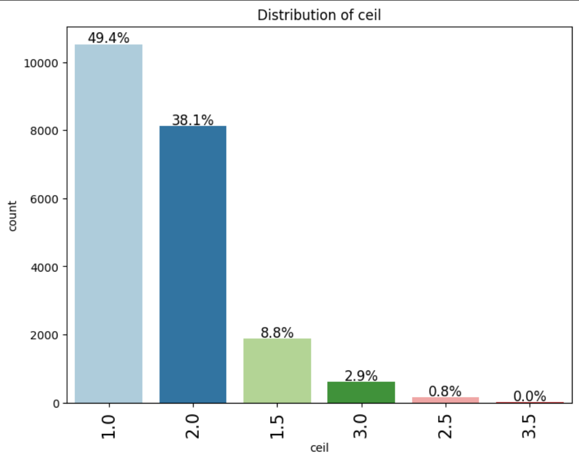
Figure 10: Stats summary after data cleaning

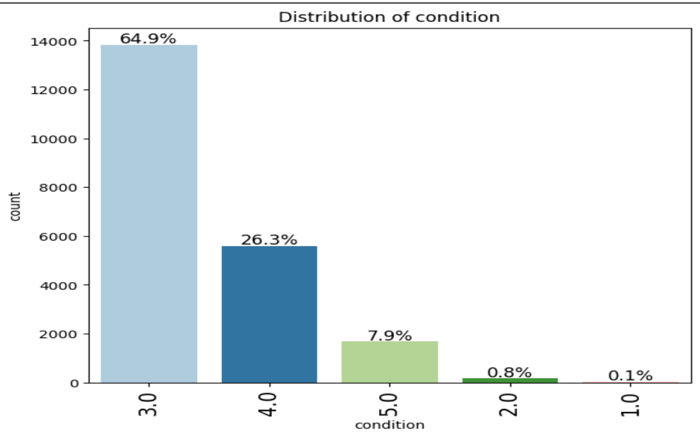
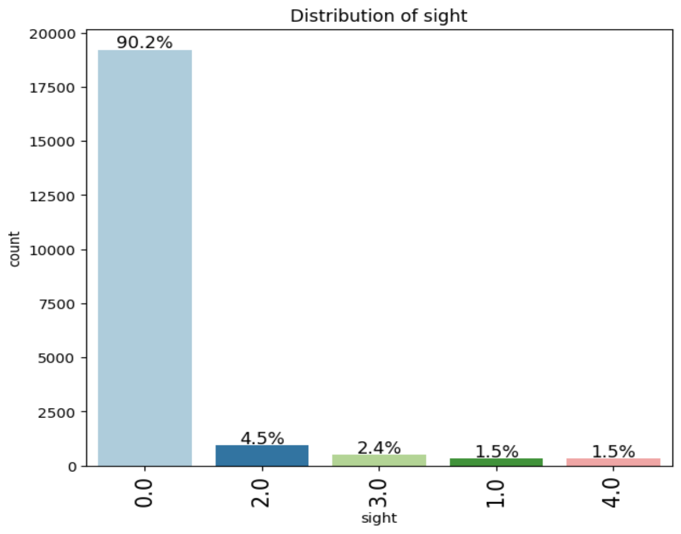
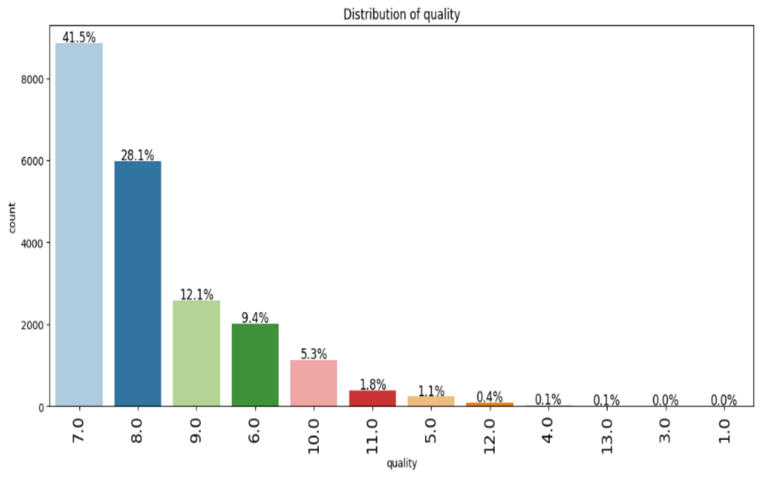
Insights

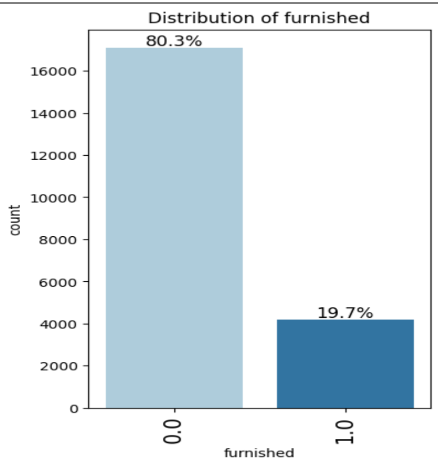
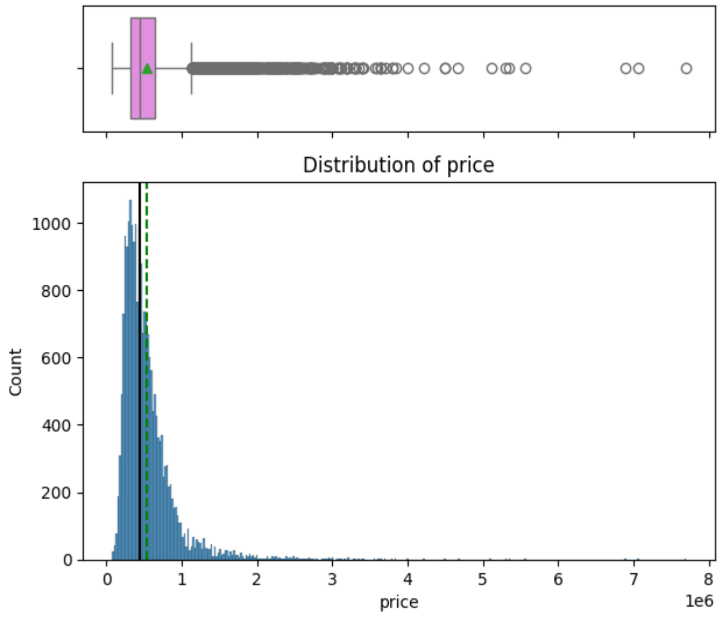
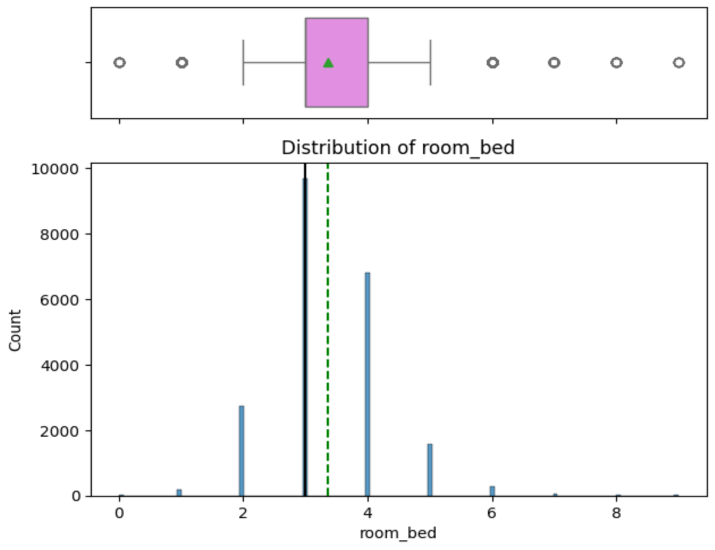
- The dataset now has a uniform count across all columns, suggesting missing values were handled.
- Standard deviations for features like lot_measure15 and price remain high, indicating variability in property sizes and prices.
- The coast feature has a very low mean (~0.007), indicating that only a small number of properties are located near the coast.
- quality has a mean of 7.658 and a relatively small standard deviation (1.177), suggesting most properties have similar quality ratings.
- basement has a mean of 291.151 but a minimum of 0, meaning many properties do not have a basement.

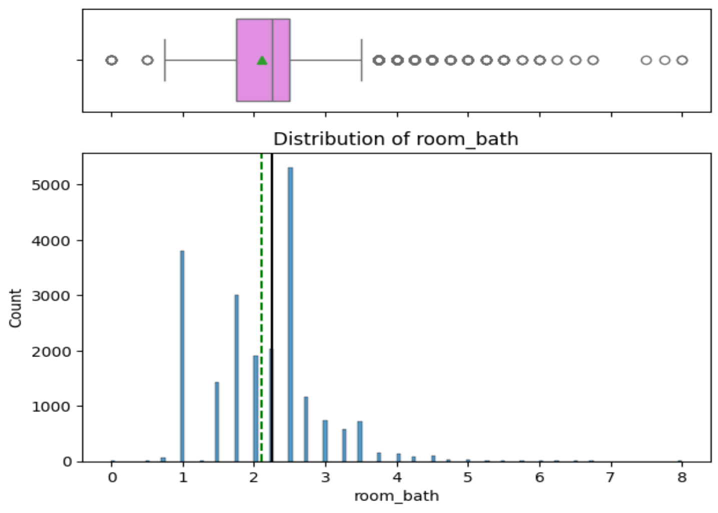
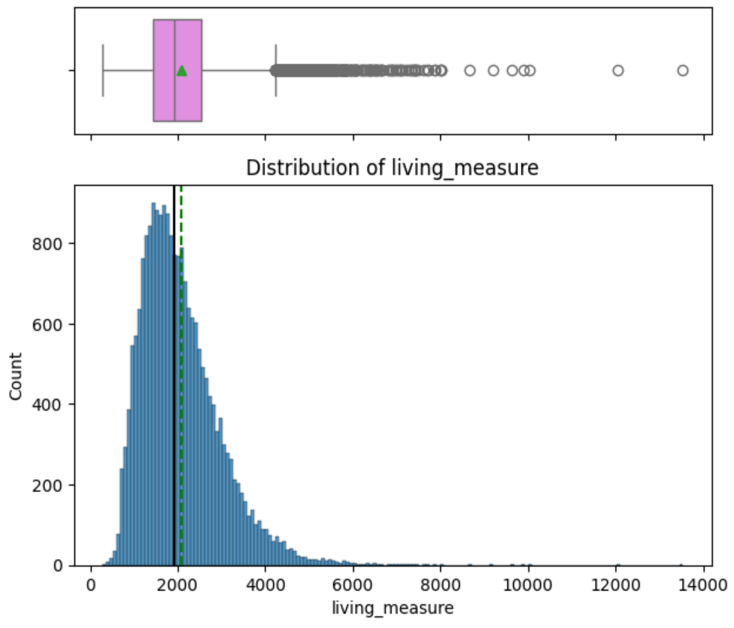
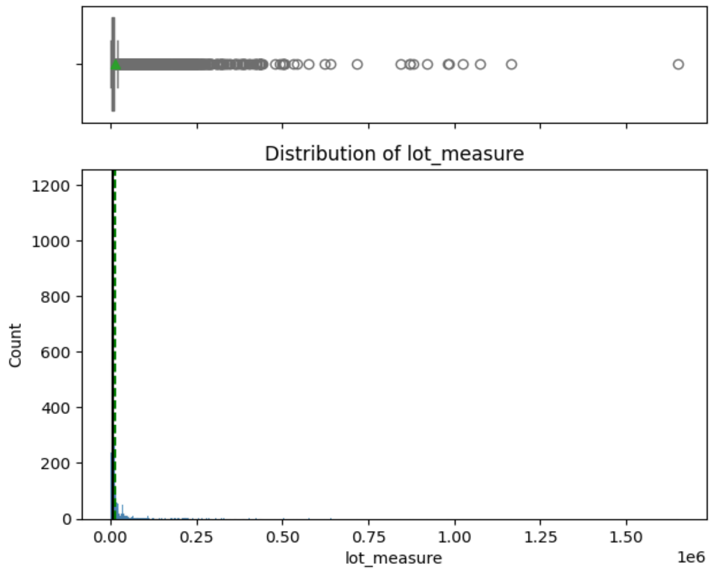
3. Exploratory Data Analysis

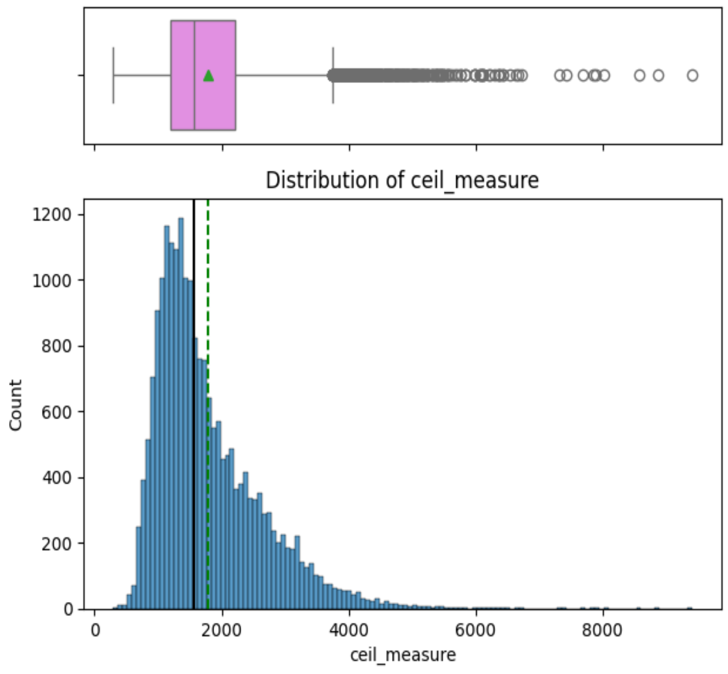
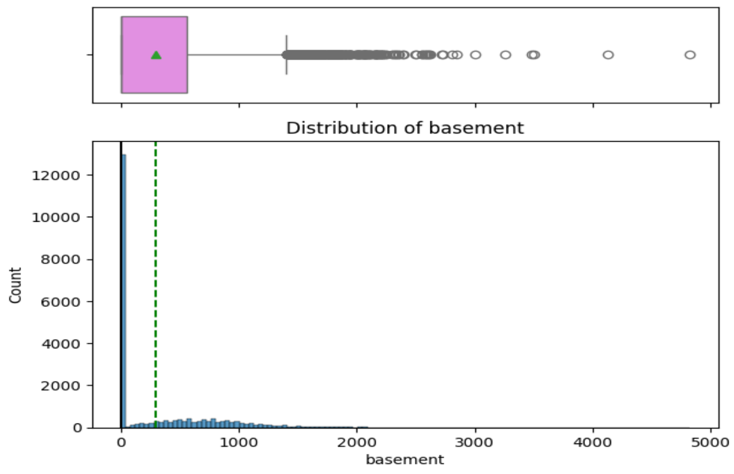
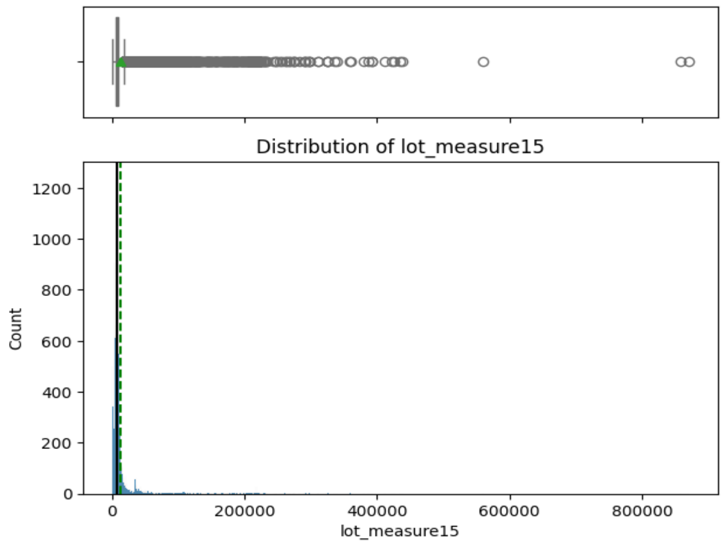
5.1. Univariate Analysis

Plot	Variable	Key Insight
 *Figure 11: coast*	coast	Only 0.7% of properties are waterfront, indicating rarity.
 *Figure 12: ceil*	ceil	Almost half (49.4%) are single-story; two-story homes make up 38.1%.

 Figure 13: condition	condition	64.9% of homes are in average condition; only 7.9% are in excellent condition.
 Figure 14: sight	sight	90.2% have no view; premium views are very rare (3.9%).
 Figure 15: quality	quality	Majority (69.9%) rated 7–8; luxury ratings (11–13) are only 2.0%.

 *Figure 16: furnished*	furnished	Only 19.7% of homes are sold furnished.
 *Figure 17: price*	price	Price is heavily right-skewed; most homes under $1M with long-tail outliers.
 *Figure 18: room_bed*	room_bed	Most homes have 3–4 bedrooms, with a peak at 3.

 Figure 19: room_bath	room_bath	Bimodal with peaks at 2 and 2.5 baths; right-skewed.
 Figure 20: living_measure	living_measure	Concentrated between 1000–3000 sq ft; long tail for larger homes.
 Figure 21: lot_measure	lot_measure	Highly right-skewed; most lots small, few large outliers.

 *Figure 22: ceil_measure*	ceil_measure	Most fall between 1000–2500 sq ft; long tail toward larger homes.
 *Figure 23: basement*	basement	Concentrated at/near zero; many have no basement; some large outliers.
 *Figure 24: lot_measure15*	lot_measure15	Highly skewed distribution; few very large lots exist.

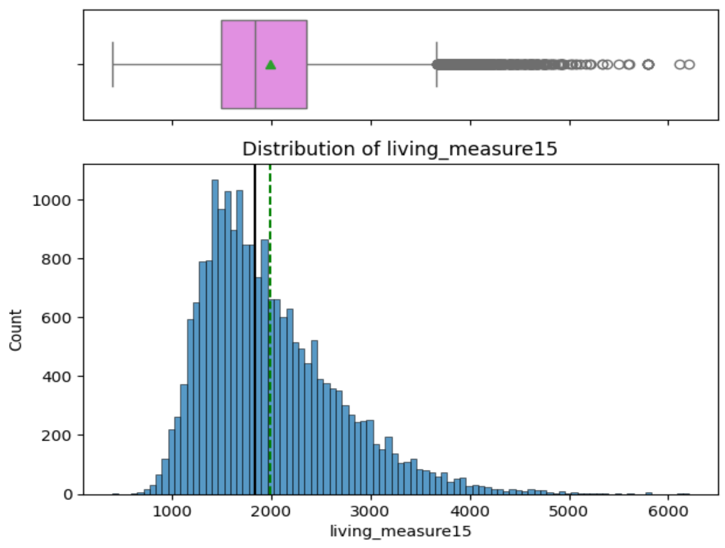
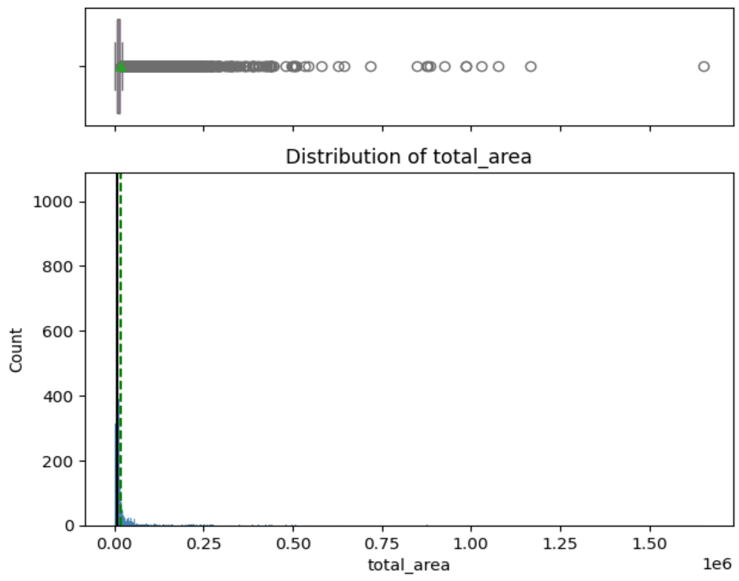
 *Figure 25: living_measure15*	living_measure15	Similar to living_measure; concentrated in 1000–3000 sq ft range.
 *Figure 26: total_area*	total_area	Most homes have small total areas; highly right-skewed with notable outliers.

Table 2: Univariate Analysis

Insights

While analyzing continuous variables, we evaluated not only their central tendency (mean, median) but also skewness. Skewness impacts the visibility and strength of the relationship between predictors and the target variable (Price). For instance, heavily right-skewed variables like plot size and living area may exaggerate or mask trends due to outliers, potentially weakening model performance if not transformed or properly binned. Hence, variables with less skewness and clearer separation in central tendencies (like Quality Grade) are more reliable predictors of house price in their raw form.

5.2. Bivariate Analysis

Correlation Check

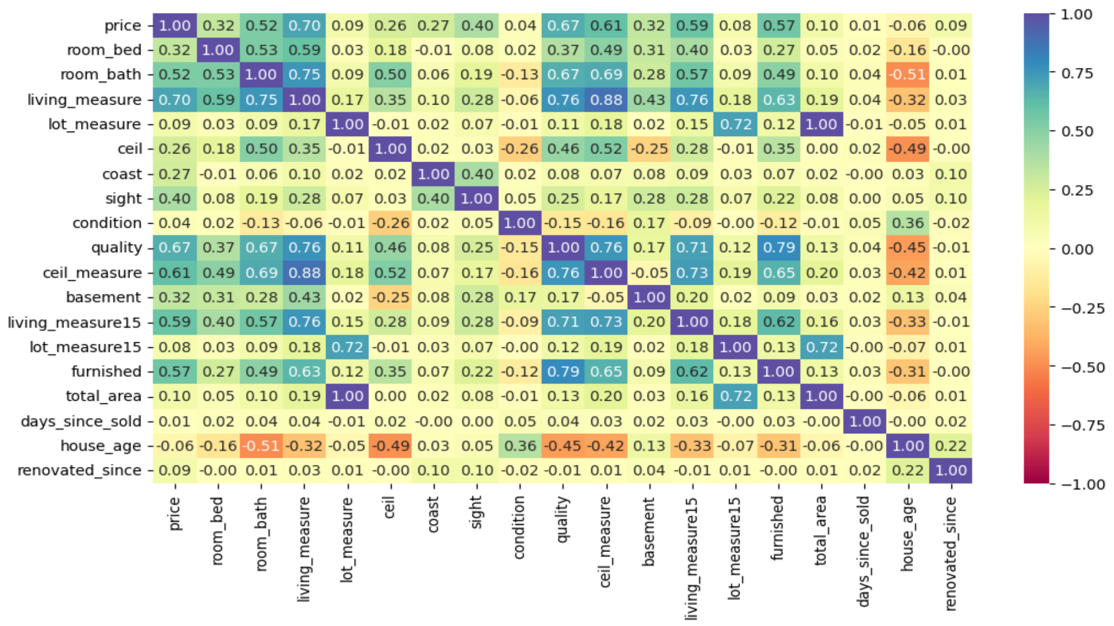


Figure 27: Heatmap

Insights

Strong Positive Correlations

- Price vs. Living Measure (0.70) & Quality (0.67) Larger living spaces and better quality homes tend to have higher prices.
- Living Measure vs. Living Measure 15 (0.76) Homes with larger interior spaces often exist in neighborhoods with similarly large houses.
- Furnished vs. Living Measure (0.63) Larger homes are more likely to be furnished.
- Ceil Measure vs. Living Measure (0.88) Homes with higher ceilings tend to have larger living areas.

Moderate Positive Correlations

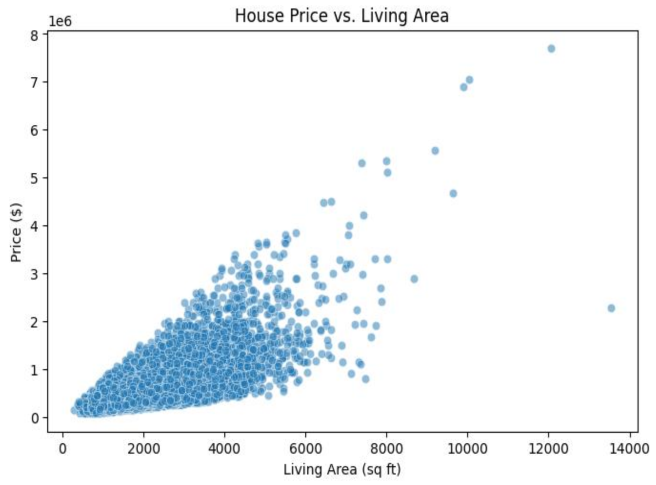
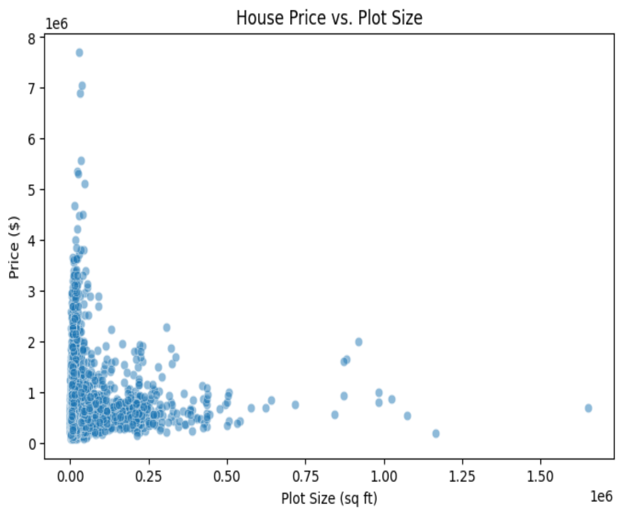
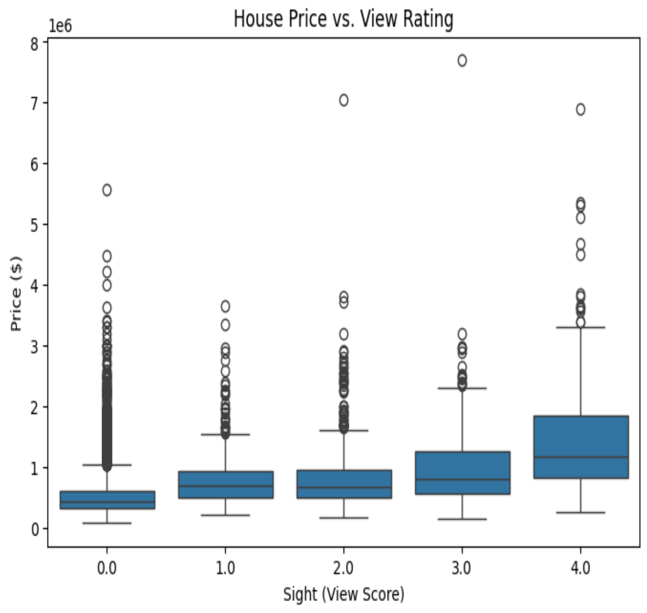
- Price vs. Room Bath (0.52) & Room Bed (0.32) More bedrooms and bathrooms generally increase the house price.
- Basement vs. Living Measure (0.43) Homes with basements tend to have more living space.
- Lot Measure vs. Lot Measure 15 (0.72) Large lots are typically found in neighborhoods with similarly large lots.

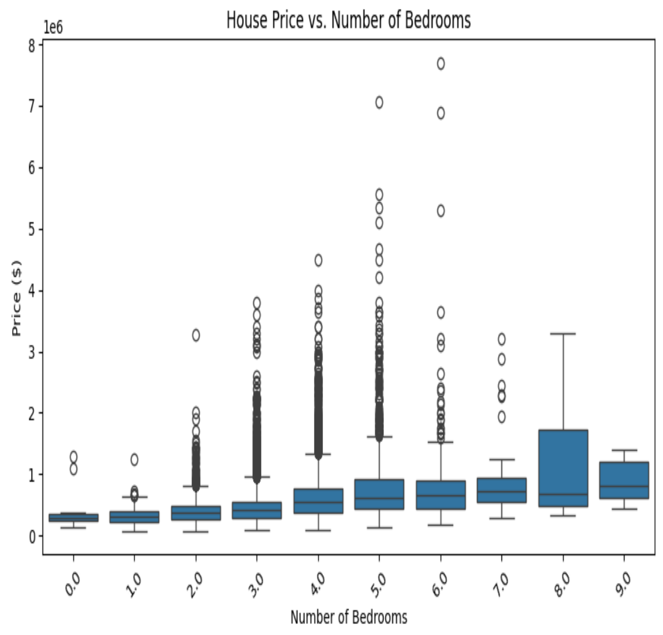
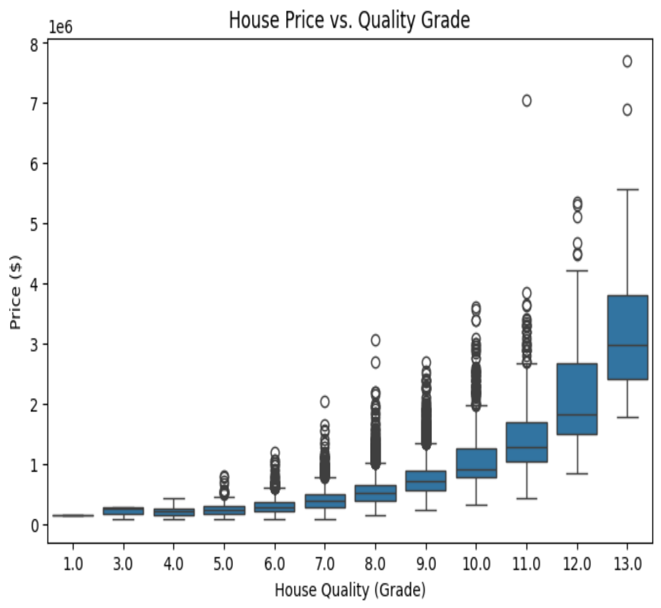
Negative Correlations

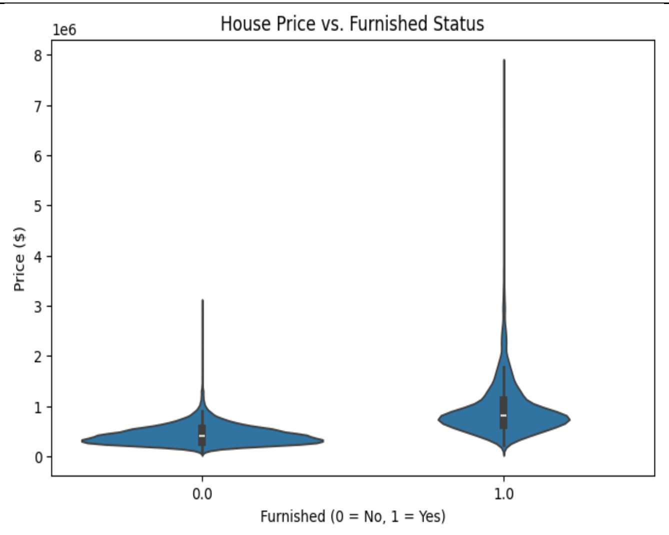
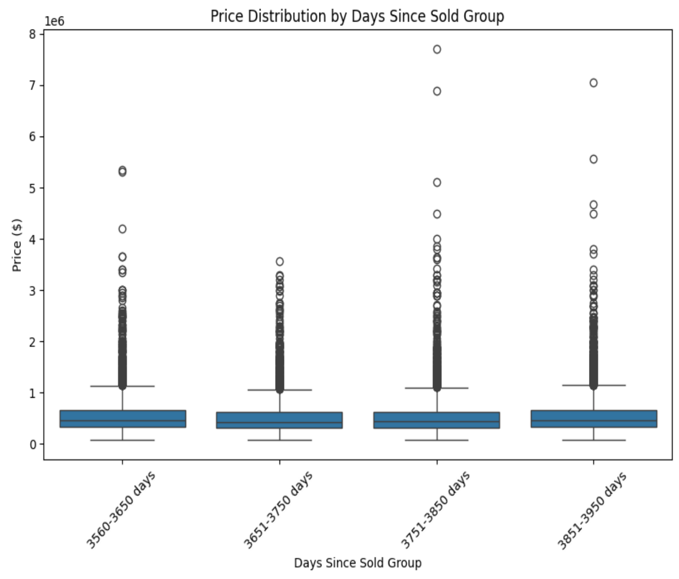
- House Age vs. Room Bath (-0.51) Older homes tend to have fewer bathrooms.
- Ceil Measure vs. House Age (-0.42) Older homes typically have lower ceilings.
- Quality vs. House Age (-0.45) Newer homes tend to have higher quality ratings.

Weak Correlations

- Condition vs. Price (0.04) Property condition does not have a strong impact on price.
- Sight vs. Price (0.40) Having a scenic view has a mild impact on house price.
- Renovated Since vs. Price (0.09) Recent renovations do not significantly impact price.

Plot	Variables	Correlation	Insights
 *Figure 28: House price vs living area*	price vs living_measure	Strong Positive	Larger homes tend to have higher prices; triangular distribution shows increasing variance.
 *Figure 29: House price vs Plot Size*	price vs lot_measure	Weak	Unusual pattern; some of the largest plots have lower prices, likely due to location.
 *Figure 30: House price vs Sight*	price vs sight	Moderate Positive (0.40)	Higher view ratings significantly raise median prices; premium views are rare.

 *Figure 31: House price vs Number of Bedrooms*	price vs room_bed	Moderate Positive (0.32)	Prices rise with bedroom count; peak value at 5 bedrooms; 3–4 bedrooms most common.
 *Figure 32: House price vs Quality*	price vs quality	Strong Positive (0.67)	Clear step-wise increase in median price with higher quality ratings.

 *Figure 33: House price vs Furnished*	price vs furnished	Moderate Positive (0.63)	Furnished homes skew toward higher prices and show more extreme outliers.
 *Figure 34: House Price vs Days since sold v1*	price vs days_since_sold	Very Weak	No meaningful trend; other factors dominate.

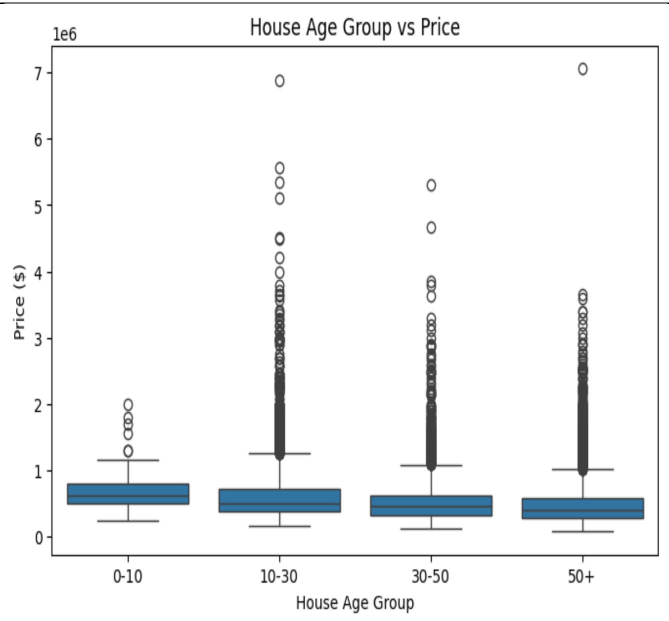
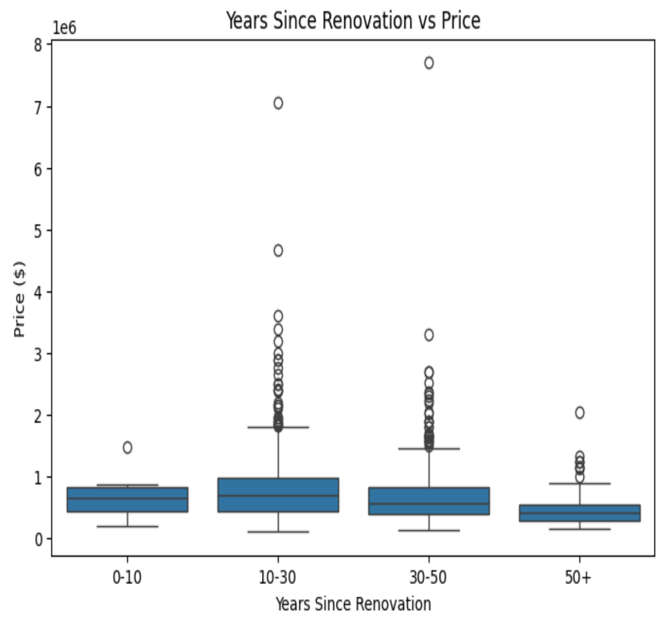
 *Figure 35: House Price vs House age v2*	price vs house_age	Weak Negative (-0.45)	Price remains relatively stable across ages; newer homes slightly more expensive.
 *Figure 36: House price vs years since renovation*	price vs years_since_renovated	Slight Negative	Renovations 10–30 years ago show better prices than older ones; recent renovations show consistency but fewer outliers.

Table 3: Bivariate Analysis

Insights

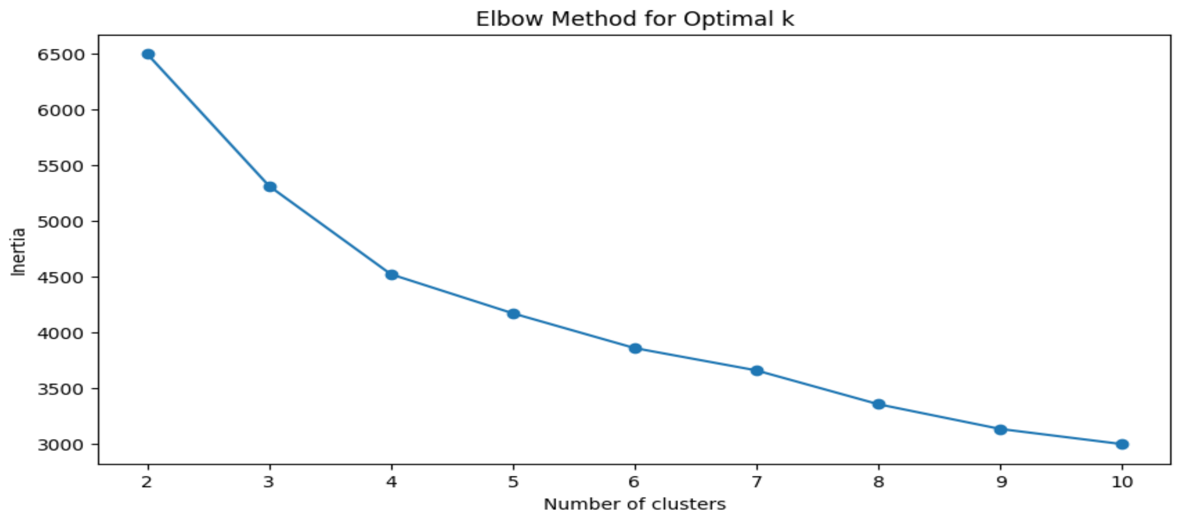
- Living Area has a strong positive correlation with price; larger homes cost more, with increasing price variability.
- Plot Size shows a weak or inverse relationship; many high-priced homes have smaller plots.
- View Quality significantly boosts price; higher-rated views lead to higher median prices.
- Bedrooms correlate with price, especially 5-bedroom homes which show the highest values.
- Furnished Homes tend to be priced higher with more premium outliers.
- Quality Grade is the strongest predictor of price, with higher grades commanding significantly higher prices.
- House Age and Days Since Sold have minimal impact on price.
- Renovation Timing matters; homes renovated 10–30 years ago tend to be more valuable than very recent or very old renovations.

4. Clustering

4.1. KMeans Clustering

- Data Preprocessing: The dataset was normalized using MinMaxScaler after removing the 'price' column to ensure all features are on a similar scale.
- Clustering Analysis: K-Means clustering was applied to identify patterns in the data, with the number of clusters (k) ranging from 2 to 10.
- Elbow Method: The inertia (sum of squared distances from points to cluster centers) was plotted against different values of k to determine the optimal number of clusters. A noticeable "elbow" in the curve indicates the ideal k.
- Silhouette Score Analysis: The silhouette score, which measures clustering quality, was calculated for each k to evaluate the cohesion and separation of clusters.

Elbow Method



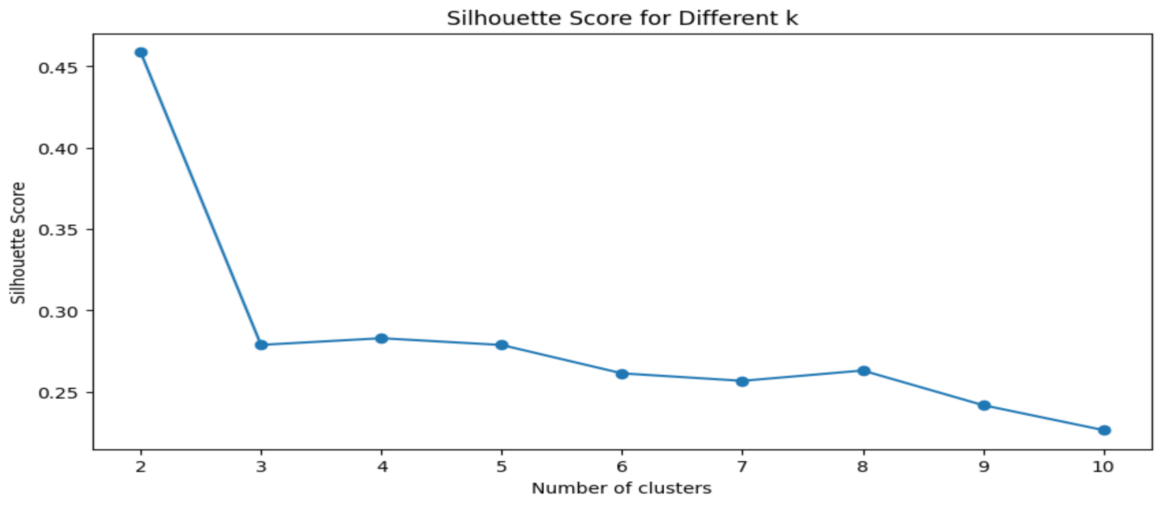
Key Observations:

- Sharp drop from k=2 to k=4: A significant decrease in inertia, meaning clusters are becoming well-separated.
- Gradual decrease from k=5 onwards: The improvement diminishes after k=4, indicating a diminishing return in adding more clusters.
- Elbow point likely around k=4 or k=5: This suggests that using 4 or 5 clusters balances compactness and interpretability.

Conclusion:

- The optimal number of clusters appears to be k=4 or k=5.
- We can validate this further using the Silhouette Score to assess cluster quality.

Silhouette Score



Key Observations:

- Highest silhouette score at k=2 (~0.46): This suggests that 2 clusters are the most well-defined.
- Slight improvement around k=4: The silhouette score stabilizes slightly at k=4, aligning with the elbow method's suggestion.
- Declining trend for k > 5: Clusters become less well-separated, meaning over-segmentation might be happening.

Conclusion:

- k=2 gives the best-defined clusters, but it may be too simplistic.
- k=4 could be a balanced choice, offering segmentation without too much overlap.
- If interpretability matters, k=4 or k=5 is a reasonable choice.

For K-Means, we can set the number of clusters to **4** and generate the clusters accordingly.

Cluster Profiling

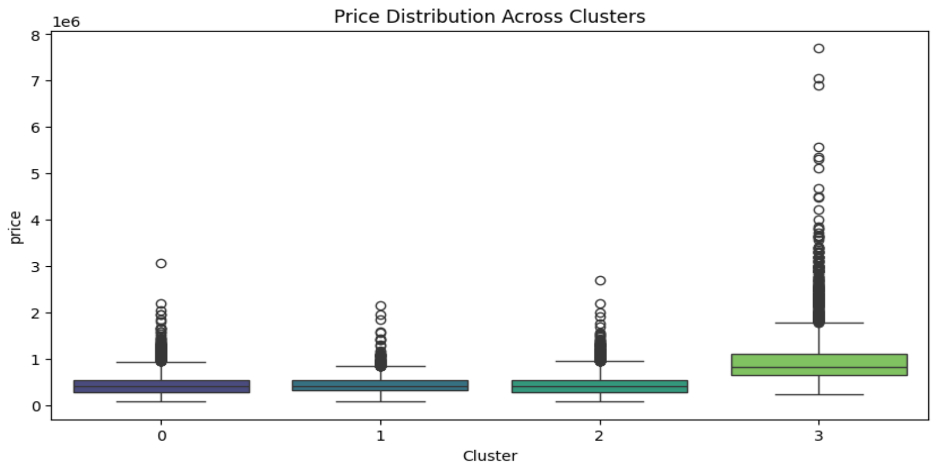


Figure 37: Price distribution across clusters

- Cluster 0: Older, Mid-Range Homes
  Affordable homes (~$437K), older (69 yrs), moderate condition & quality, decent space with large lots. Rarely renovated, may need updates. Likely in suburban/rural areas.

- Cluster 1: Newer, Compact, Slightly Expensive Homes
  Newer (25 yrs), slightly pricier (~$439K), modern 2-floor layout, smaller lots but efficient space. Low renovation needs, family-friendly with more bathrooms.

- Cluster 2: Older, More Deteriorated Homes
  Very similar to Cluster 0 but older (72 yrs) and in slightly worse condition. Still affordable, with large lots but low renovation frequency. May require more maintenance.

- Cluster 3: Luxury Homes
  High-end homes (~$961K), spacious with largest living & lot areas. Best quality, frequent renovations, and some coastal views. Ideal for premium buyers, though moderately aged (35 yrs).

4.2. Hierarchical Clustering

- Hierarchical Clustering Analysis: Applied Agglomerative Clustering to group data points based on similarity without predefining cluster numbers.
- Dendrogram Visualization: A dendrogram was generated using Ward's method to determine the optimal number of clusters by identifying natural splits in the data.
- Clustering Implementation: Agglomerative Clustering was performed with four clusters, using Euclidean distance as the metric and Ward's linkage to minimize variance within clusters.

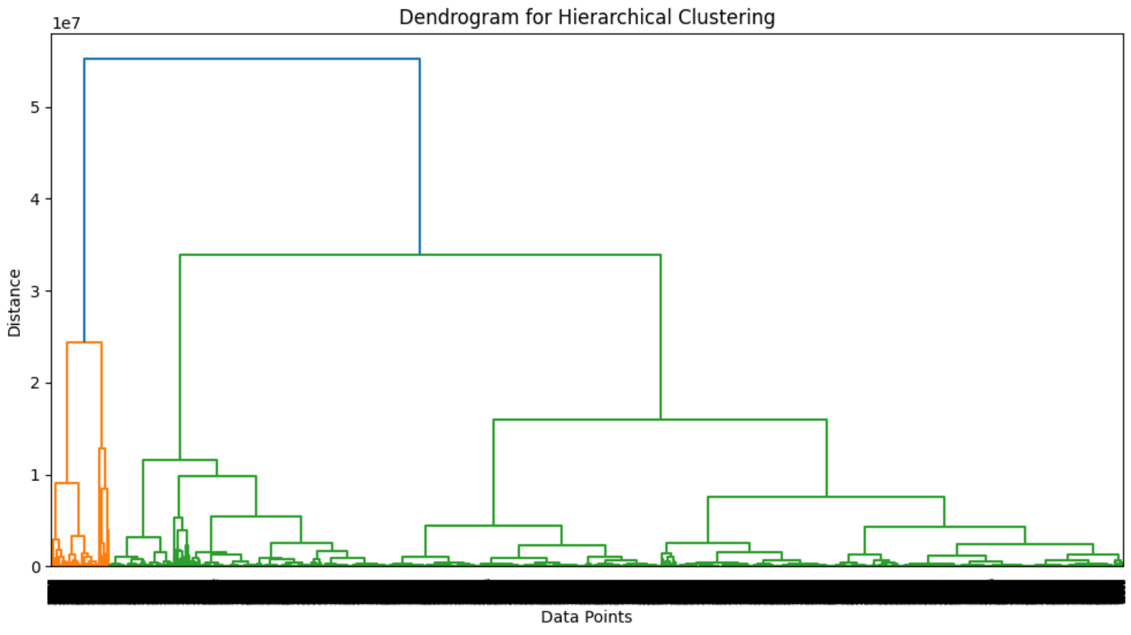


Figure 38: Dendrogram

Insights

1. Number of Clusters
   - The large vertical blue line at the top suggests that the data naturally splits into two primary clusters at a high level.
   - As you move down, the tree branches into smaller subclusters, indicating more fine-grained groupings.
2. Cluster Separation
   - The height of the merges (vertical lines) represents the distance between clusters.
   - The biggest vertical jumps (especially in blue and green) show where significant differences exist between clusters.
   - The orange section on the left suggests closely related data points with smaller distances, meaning those points are more similar to each other.
3. Best Cut-Off for Clusters
   - If you want a higher-level segmentation, cutting around the large blue split (at ~50 million distance) would give 2 clusters.
   - If you need more detailed clustering, cutting at a lower level (around 10-15 million) could result in 4-6 clusters.
4. Cluster Density
   - The left side of the dendrogram has many short branches, indicating dense, highly similar data points.
   - The right side has longer branches, meaning the data points are more dispersed and dissimilar.

Cluster Profiling

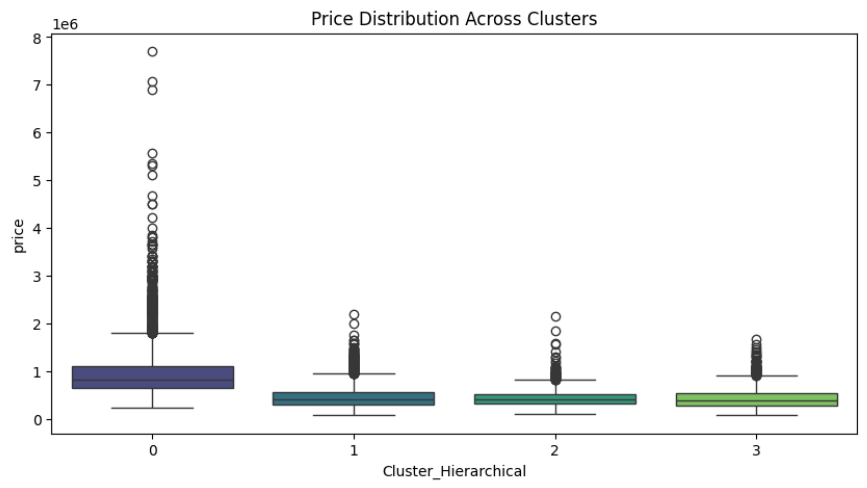


Figure 39: Price distribution across clusters

- Cluster 0: Luxury Estates
  High-end homes (~$961K), spacious with premium quality (9.5), mostly furnished (98%), and recently renovated. Ideal for luxury buyers seeking move-in-ready properties.
- Cluster 1: Standard Family Homes
  Mid-priced (~$439K), oldest but in best condition. Large lots, moderate living area, and unfurnished—offering room for personalization.
- Cluster 2: Modern Mid-Sized Homes
  Newer homes (~$435K) with highest ceilings and a modern layout. Not furnished, minimally renovated, likely with original features.
- Cluster 3: Budget-Friendly Homes
  Most affordable (~$424K), smaller in size with lower ceilings. Older and minimally renovated, ideal for first-time buyers on a budget.

4.3. Clusters

Feature	**Cluster 0: Luxury Estates**	**Cluster 1: Standard Family Homes**	**Cluster 2: Mid-Sized Modern Homes**	**Cluster 3: Budget-Friendly Homes**
Price (Mean, Median)	$961K, $810K	$439K, $400K	$435K, $410K	$424K, $390K
Bedrooms / Bathrooms	3.8 / 2.9	3.2 / 1.8	3.4 / 2.5	3.2 / 1.7
Living Area	3,237 sq. ft.	1,742 sq. ft.	2,018 sq. ft.	1,623 sq. ft.
Lot Size	25,097 sq. ft.	14,108 sq. ft.	10,404 sq. ft.	11,643 sq. ft.
Ceiling Height	1.87 floors	1.17 floors	2.1 floors (tallest)	1.1 floors (lowest)
Condition / Quality	3.25 / 9.5 (best quality)	3.61 (best condition) / 7.1	3.08 / 7.6	3.51 / 7.0
Basement Size	375 sq. ft.	353 sq. ft.	73 sq. ft. (smallest)	314 sq. ft.
Home Age	36 years	70 years (oldest)	26 years (youngest)	70 years
Renovation (Years Since)	1.36 years (recently renovated)	2 years (moderate)	0.1 years (least renovated)	0.6 years (minimal updates)

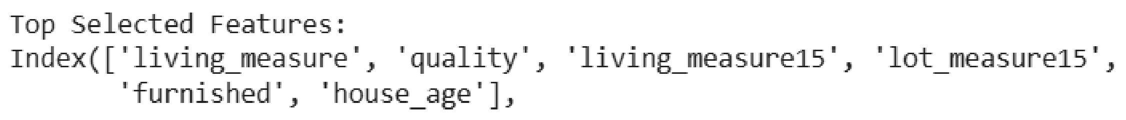
Furnishing	Mostly furnished (98%)	Not furnished (0%)	Not furnished (0%)	Not furnished (0%)
Coastal View (%)	3.7% (highest)	0%	0.2% (extremely rare)	0%

Table 4: Clusters

5. Recursive Feature Elimination

Feature selection is a crucial step in building predictive models, as it helps improve model performance by eliminating irrelevant or redundant features. One effective method for feature selection is Recursive Feature Elimination (RFE), which iteratively removes the least important features based on model performance until the optimal number of features is selected.

We employed a Random Forest Regressor model within RFE, which recursively fits the model while eliminating the least significant features based on coefficients.



```
Top Selected Features:
Index(['living_measure', 'quality', 'living_measure15', 'lot_measure15',
       'furnished', 'house_age'],
```

Figure 40: RFE Selected Features

6. Model Building and Tuning

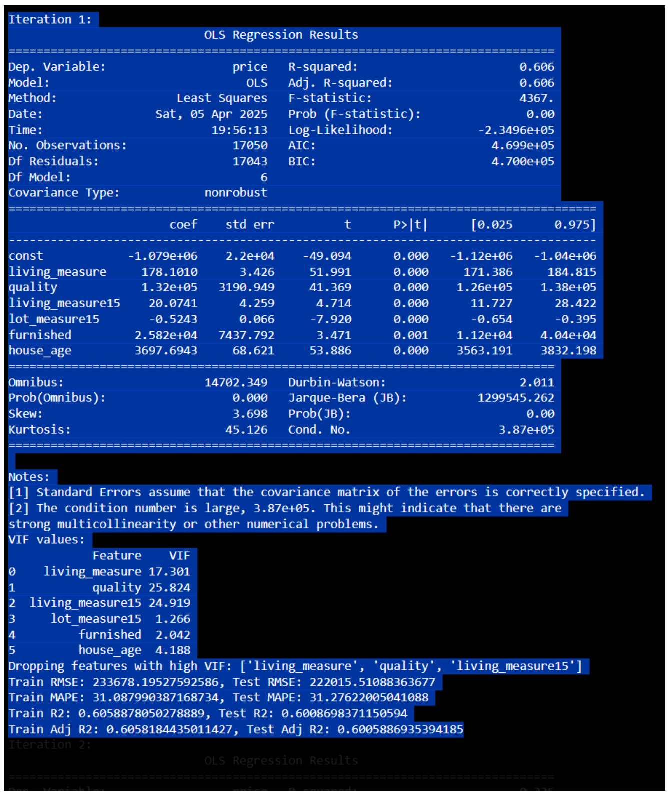
6.1. Parametric Models

6.1.1. OLS Regression

Ordinary Least Squares (OLS) is a method for estimating the relationships between a dependent variable and one or more independent variables.

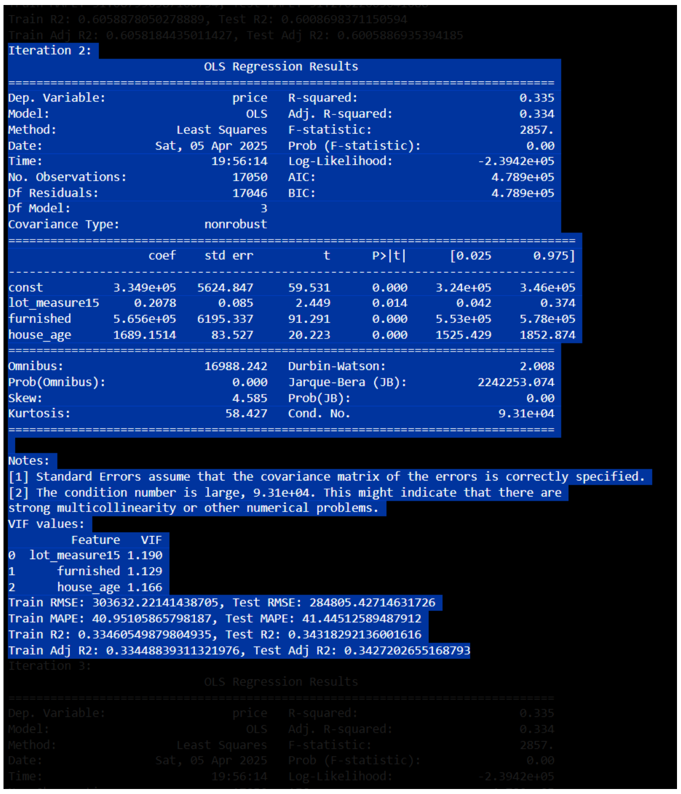
It minimizes the sum of squared residuals (differences between observed and predicted values).

6.1.1.1 Various Iterations of MLR



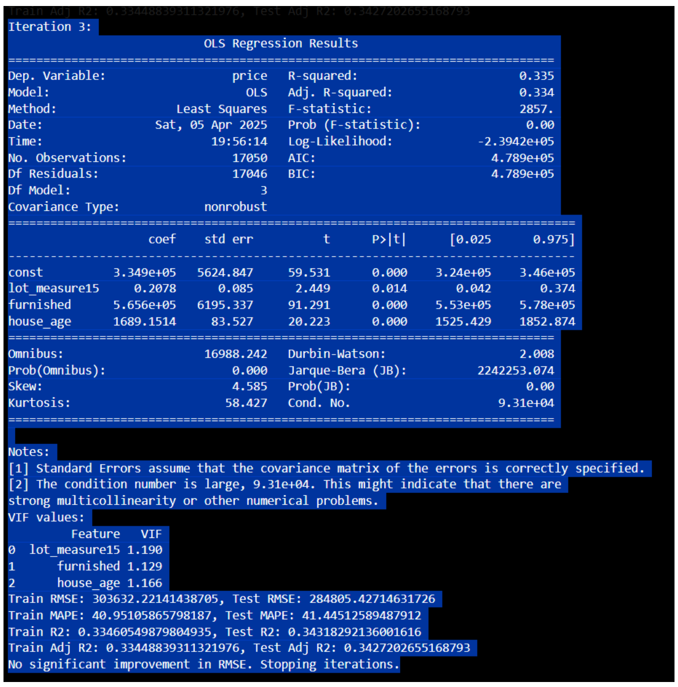
```
Iteration 1:
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.606
Model:                            OLS   Adj. R-squared:                  0.606
Method:                 Least Squares   F-statistic:                     4367.
Date:                Sat, 05 Apr 2025   Prob (F-statistic):               0.00
Time:                        19:56:13   Log-Likelihood:            -2.3496e+05
No. Observations:               17050   AIC:                         4.699e+05
Df Residuals:                   17043   BIC:                         4.700e+05
Df Model:                           6
Covariance Type:            nonrobust
===================================================================================
                      coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------
const            -1.079e+06    2.2e+04    -49.094      0.000   -1.12e+06   -1.04e+06
living_measure     178.1010      3.426     51.991      0.000     171.386     184.815
quality            1.32e+05   3190.949     41.369      0.000    1.26e+05    1.38e+05
living_measure15    20.0741      4.259      4.714      0.000      11.727      28.422
lot_measure15       -0.5243      0.066     -7.920      0.000      -0.654      -0.395
furnished         2.582e+04   7437.792      3.471      0.001    1.12e+04    4.04e+04
house_age         3697.6943     68.621     53.886      0.000    3563.191    3832.198
==============================================================================
Omnibus:                    14702.349   Durbin-Watson:                   2.011
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          1299545.262
Skew:                           3.698   Prob(JB):                         0.00
Kurtosis:                      45.126   Cond. No.                     3.87e+05
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 3.87e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
VIF values:
            Feature     VIF
0    living_measure 17.301
1           quality 25.824
2  living_measure15 24.919
3     lot_measure15  1.266
4         furnished  2.042
5         house_age  4.188
Dropping features with high VIF: ['living_measure', 'quality', 'living_measure15']
Train RMSE: 233678.19527592586, Test RMSE: 222015.51088363677
Train MAPE: 31.087990387168734, Test MAPE: 31.27622005041088
Train R2: 0.6058878050278889, Test R2: 0.6008698371150594
Train Adj R2: 0.6058184435011427, Test Adj R2: 0.6005886935394185
Iteration 2:
                            OLS Regression Results
==============================================================================
```



```
Train R2: 0.6058878050278889, Test R2: 0.6008698371150594
Train Adj R2: 0.6058184435011427, Test Adj R2: 0.6005886935394185
Iteration 2:
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.335
Model:                            OLS   Adj. R-squared:                  0.334
Method:                 Least Squares   F-statistic:                     2857.
Date:                Sat, 05 Apr 2025   Prob (F-statistic):               0.00
Time:                        19:56:14   Log-Likelihood:            -2.3942e+05
No. Observations:               17050   AIC:                         4.789e+05
Df Residuals:                   17046   BIC:                         4.789e+05
Df Model:                           3
Covariance Type:            nonrobust
=================================================================================
                    coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------
const          3.349e+05   5624.847     59.531      0.000    3.24e+05    3.46e+05
lot_measure15     0.2078      0.085      2.449      0.014       0.042       0.374
furnished      5.656e+05   6195.337     91.291      0.000    5.53e+05    5.78e+05
house_age      1689.1514     83.527     20.223      0.000    1525.429    1852.874
==============================================================================
Omnibus:                    16988.242   Durbin-Watson:                   2.008
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          2242253.074
Skew:                           4.585   Prob(JB):                         0.00
Kurtosis:                      58.427   Cond. No.                     9.31e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 9.31e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
VIF values:
         Feature    VIF
0  lot_measure15 1.190
1      furnished 1.129
2      house_age 1.166
Train RMSE: 303632.22141438705, Test RMSE: 284805.42714631726
Train MAPE: 40.95105865798187, Test MAPE: 41.44512589487912
Train R2: 0.33460549879804935, Test R2: 0.34318292136001616
Train Adj R2: 0.33448839311321976, Test Adj R2: 0.3427202655168793
Iteration 3:
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.335
Model:                            OLS   Adj. R-squared:                  0.334
Method:                 Least Squares   F-statistic:                     2857.
Date:                Sat, 05 Apr 2025   Prob (F-statistic):               0.00
Time:                        19:56:14   Log-Likelihood:            -2.3942e+05
```



```
Train Adj R2: 0.33448839311321976, Test Adj R2: 0.3427202655168793
Iteration 3:
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.335
Model:                            OLS   Adj. R-squared:                  0.334
Method:                 Least Squares   F-statistic:                     2857.
Date:                Sat, 05 Apr 2025   Prob (F-statistic):               0.00
Time:                        19:56:14   Log-Likelihood:            -2.3942e+05
No. Observations:               17050   AIC:                         4.789e+05
Df Residuals:                   17046   BIC:                         4.789e+05
Df Model:                           3
Covariance Type:            nonrobust
=================================================================================
                    coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------
const          3.349e+05   5624.847     59.531      0.000    3.24e+05    3.46e+05
lot_measure15     0.2078      0.085      2.449      0.014       0.042       0.374
furnished      5.656e+05   6195.337     91.291      0.000    5.53e+05    5.78e+05
house_age      1689.1514     83.527     20.223      0.000    1525.429    1852.874
==============================================================================
Omnibus:                    16988.242   Durbin-Watson:                   2.008
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          2242253.074
Skew:                           4.585   Prob(JB):                         0.00
Kurtosis:                      58.427   Cond. No.                     9.31e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 9.31e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
VIF values:
         Feature    VIF
0  lot_measure15 1.190
1      furnished 1.129
2      house_age 1.166
Train RMSE: 303632.22141438705, Test RMSE: 284805.42714631726
Train MAPE: 40.95105865798187, Test MAPE: 41.44512589487912
Train R2: 0.33460549879804935, Test R2: 0.34318292136001616
Train Adj R2: 0.33448839311321976, Test Adj R2: 0.3427202655168793
No significant improvement in RMSE. Stopping iterations.
```

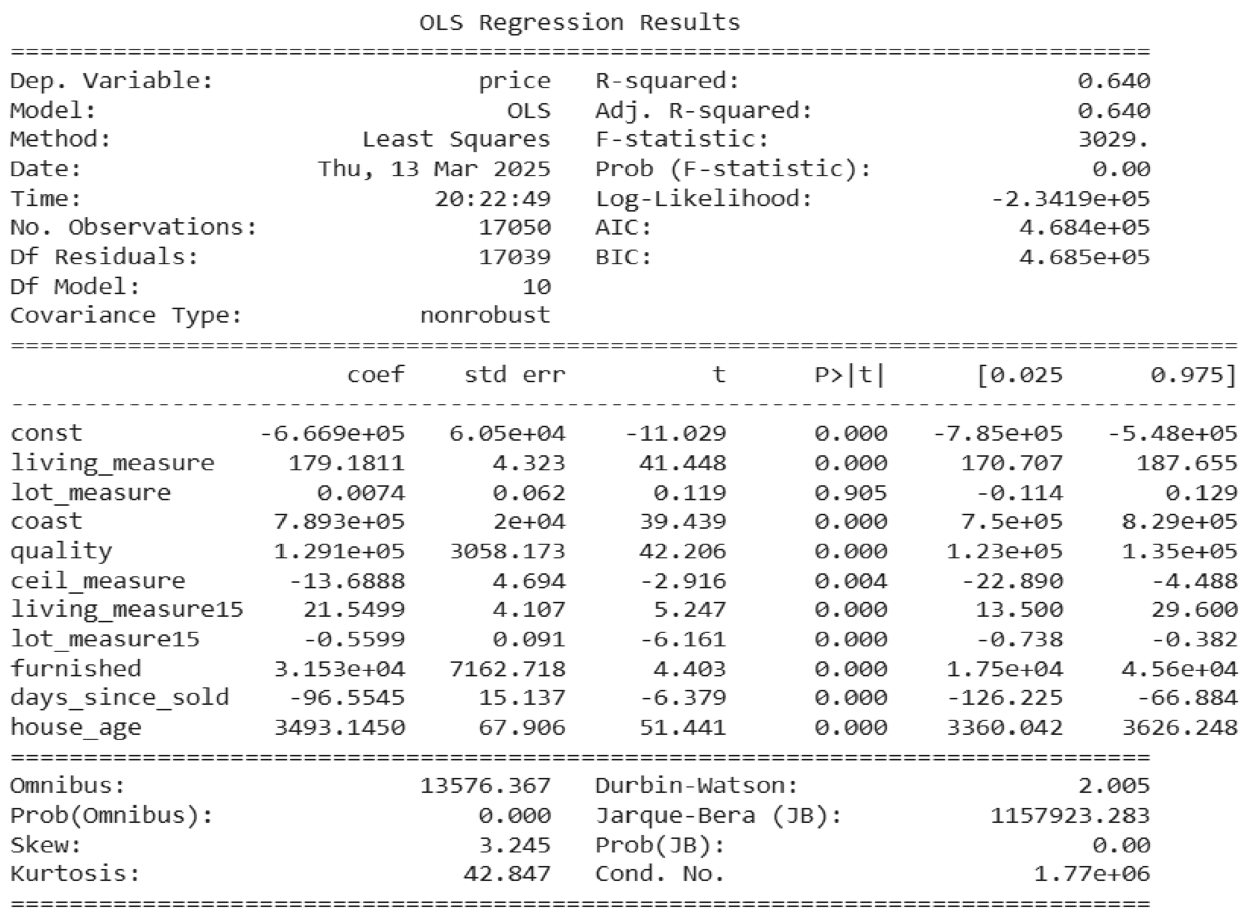
Figure 41: MLR Iterations

The process involves performing multiple linear regression (MLR) with iterative feature selection to improve model performance.

1. **Iteration 1:** The model is trained with all features. High multicollinearity is detected in features like living_measure, quality, and living_measure15, which have high VIFs. These features are dropped, and the model's performance is evaluated, with an R-squared of 0.606 and reasonable RMSE.
2. **Iteration 2:** The model is retrained with fewer features, resulting in a significant drop in R-squared to 0.335 and an increase in RMSE, indicating reduced predictive power.

3. **Iteration 3:** No improvement in performance is observed compared to the previous iteration, so the process is stopped based on the lack of significant improvement in RMSE.

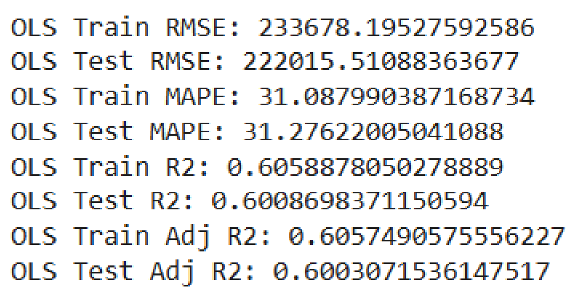
In conclusion, while multicollinearity was addressed, the reduction in features led to worse model performance, prompting an early stop to further iterations.



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.640
Model:                            OLS   Adj. R-squared:                  0.640
Method:                 Least Squares   F-statistic:                     3029.
Date:                Thu, 13 Mar 2025   Prob (F-statistic):               0.00
Time:                        20:22:49   Log-Likelihood:            -2.3419e+05
No. Observations:               17050   AIC:                         4.684e+05
Df Residuals:                   17039   BIC:                         4.685e+05
Df Model:                          10
Covariance Type:            nonrobust
====================================================================================
                       coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------
const            -6.669e+05   6.05e+04    -11.029      0.000   -7.85e+05   -5.48e+05
living_measure     179.1811      4.323     41.448      0.000     170.707     187.655
lot_measure          0.0074      0.062      0.119      0.905      -0.114       0.129
coast             7.893e+05      2e+04     39.439      0.000     7.5e+05    8.29e+05
quality           1.291e+05   3058.173     42.206      0.000    1.23e+05    1.35e+05
ceil_measure       -13.6888      4.694     -2.916      0.004     -22.890      -4.488
living_measure15    21.5499      4.107      5.247      0.000      13.500      29.600
lot_measure15       -0.5599      0.091     -6.161      0.000      -0.738      -0.382
furnished         3.153e+04   7162.718      4.403      0.000    1.75e+04    4.56e+04
days_since_sold    -96.5545     15.137     -6.379      0.000    -126.225     -66.884
house_age         3493.1450     67.906     51.441      0.000    3360.042    3626.248
==============================================================================
Omnibus:                    13576.367   Durbin-Watson:                   2.005
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          1157923.283
Skew:                           3.245   Prob(JB):                         0.00
Kurtosis:                      42.847   Cond. No.                     1.77e+06
==============================================================================
```

Figure 42: OLS

- R-squared = 0.606 → Model explains 60.6% of the variance in price.
- Significant predictors: living_measure, quality, living_measure15, lot_measure15, furnished, house_age (all with p-values < 0.05).
- Insignificant predictors: None explicitly stated, but potential multicollinearity might have affected the significance of some variables.
- Negative impact: lot_measure15 has a negative coefficient, suggesting a negative relationship with price.
- Durbin-Watson = 2.011 → No severe autocorrelation.
- High Condition Number (3.87e+05) → Possible multicollinearity issues.
- Residuals are non-normal (Omnibus & Jarque-Bera p = 0.00), indicating that the residuals are skewed and exhibit high kurtosis, suggesting a deviation from normality.



```
OLS Train RMSE: 233678.19527592586
OLS Test RMSE: 222015.51088363677
OLS Train MAPE: 31.087990387168734
OLS Test MAPE: 31.27622005041088
OLS Train R2: 0.6058878050278889
OLS Test R2: 0.6008698371150594
OLS Train Adj R2: 0.6057490575556227
OLS Test Adj R2: 0.6003071536147517
```

Figure 43: OLS model metrics

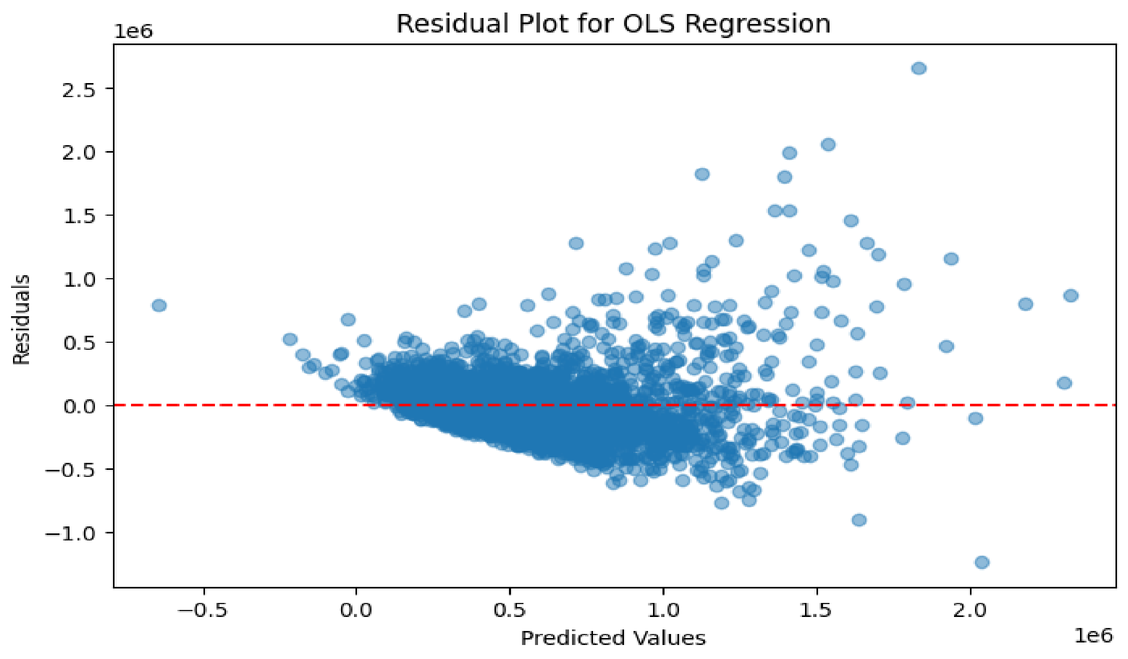


Figure 44: OLS Residual Plot

6.1.2. Lasso Regression - Tuned

Lasso (Least Absolute Shrinkage and Selection Operator) is a type of regression that adds L1 regularization to the OLS model. It helps in feature selection by shrinking some coefficients to exactly zero, effectively removing less important variables.

To determine the optimal regularization strength for Lasso regression, we performed hyperparameter tuning using GridSearchCV. The parameter grid included different values for the alpha parameter [0.01,0.1,1,10,100][0.01, 0.1, 1, 10, 100][0.01,0.1,1,10,100], which controls the amount of regularization applied to the model. The tuning process used 5-fold cross-validation and evaluated model performance based on negative mean squared error (MSE).

Following the hyperparameter tuning, we trained a Lasso regression model with an alpha value of 10. These predictions can be further evaluated to assess the model's performance in minimizing error and improving generalization.

```
Lasso Train RMSE: 233678.197846037
Lasso Test RMSE: 222015.56882892945
Lasso Train MAPE: 31.088625070810888
Lasso Test MAPE: 31.276601890633742
Lasso Train R2: 0.6058877963585978
Lasso Test R2: 0.600869628771804
Lasso Train Adj R2: 0.6057490488832795
Lasso Test Adj R2: 0.6003069449777794
```

Figure 45: Lasso model metrics

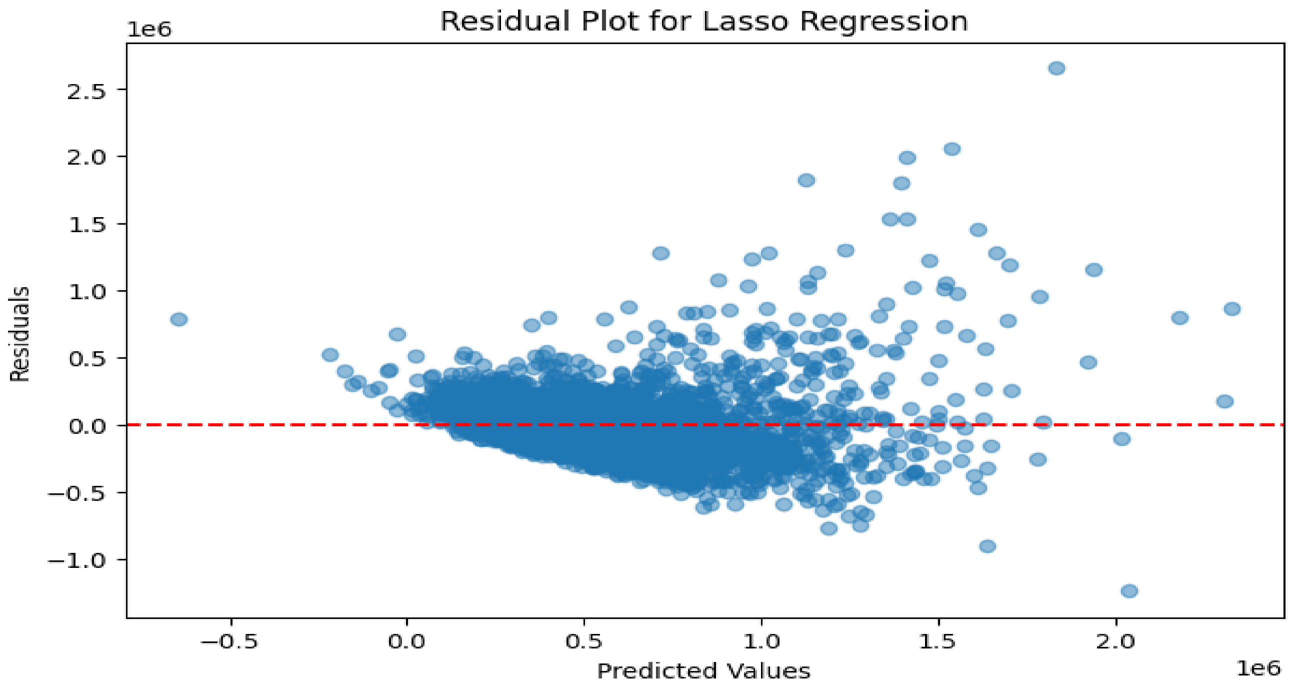


Figure 46: Lasso Residual plot

6.1.3. Ridge Regression - Tuned

Ridge Regression is a type of linear regression that applies L2 regularization to reduce overfitting. It adds a penalty term (alpha * sum of squared coefficients) to the OLS loss function. Unlike Lasso, Ridge shrinks coefficients but does not set them to zero, meaning all features contribute to the model.

To determine the optimal regularization strength for Ridge regression, GridSearchCV was used to tune the alpha parameter. The parameter grid included different values of alpha [0.01,0.1,1,10,100][0.01, 0.1, 1, 10, 100][0.01,0.1,1,10,100], which controls the degree of penalty applied to the regression coefficients. The tuning process used 5-fold cross-validation and evaluated performance based on negative mean squared error (MSE).

Following the hyperparameter tuning, a Ridge regression model was trained using an alpha value of 1.0.

```
Ridge Train RMSE: 233678.19555652916
Ridge Test RMSE: 222015.3984816356
Ridge Train MAPE: 31.087827362829824
Ridge Test MAPE: 31.276072586987862
Ridge Train R2: 0.6058878040813804
Ridge Test R2: 0.6008702412581528
Ridge Train Adj R2: 0.6057490566087811
Ridge Test Adj R2: 0.6003075583275956
```

Figure 47: Ridge model metrics

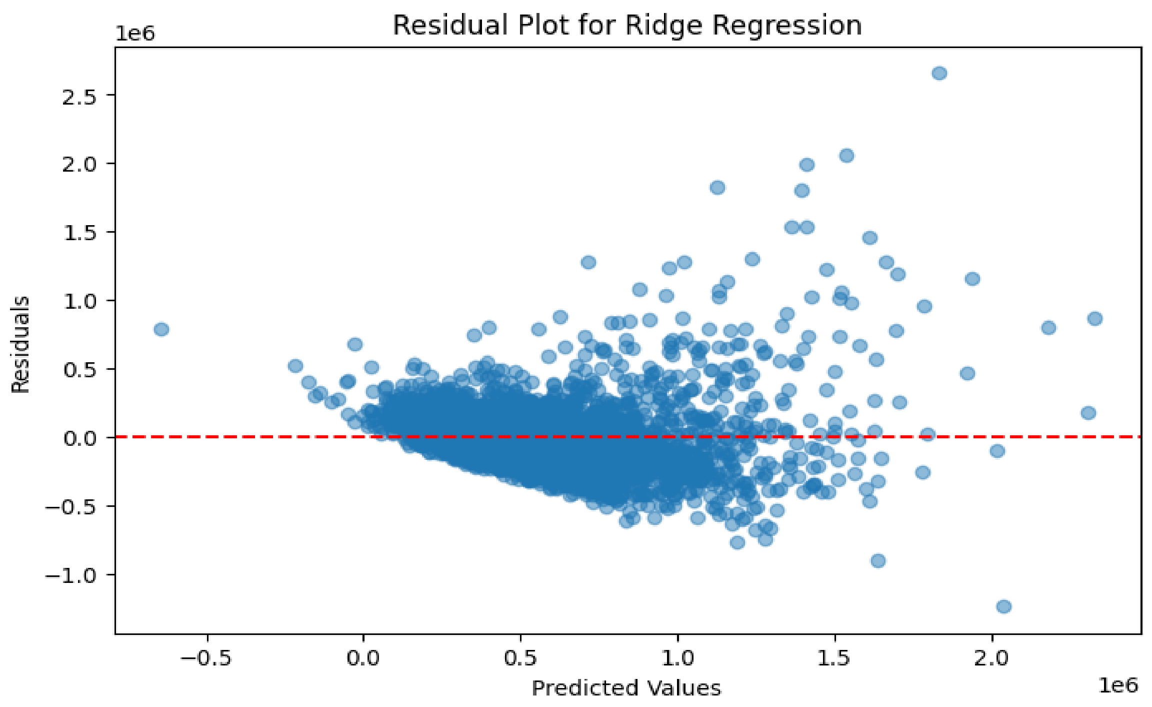


Figure 48: Ridge Residual Plot

Residual Plot Observation

OLS, Lasso, and Ridge Regression:

- The residuals are somewhat symmetrically distributed around zero.
- However, there's a funnel shape, indicating heteroscedasticity (variance of residuals increases with predicted values).
- Presence of outliers suggests that the model struggles with extreme values.

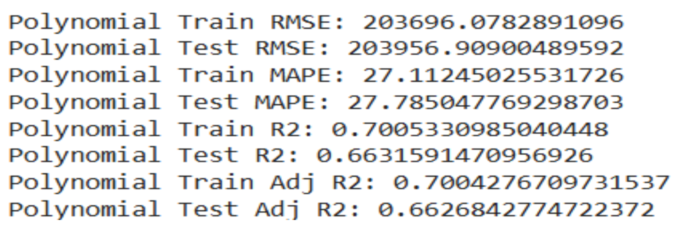
6.1.4. Polynomial Regression - Tuned

Polynomial Regression is an extension of linear regression where the relationship between the dependent and independent variable is modeled as an n-degree polynomial. It transforms input features by adding higher-degree terms (e.g., x^2, x^3, etc.) to capture nonlinear relationships.

To determine the optimal degree for Polynomial Regression, we tested different polynomial degrees ranging from 1 to 5. For each degree:

1. Polynomial features were generated by transforming the input data.
2. A Linear Regression model was trained using the transformed training data.
3. The model's performance was evaluated using the coefficient of determination (R2R^2R2) on the test set.
4. The degree with the highest test score was selected as the best degree.

After identifying the optimal degree, we trained a Polynomial Regression model with a degree of 2.



```
Polynomial Train RMSE: 203696.0782891096
Polynomial Test RMSE: 203956.90900489592
Polynomial Train MAPE: 27.11245025531726
Polynomial Test MAPE: 27.785047769298703
Polynomial Train R2: 0.7005330985040448
Polynomial Test R2: 0.6631591470956926
Polynomial Train Adj R2: 0.7004276709731537
Polynomial Test Adj R2: 0.6626842774722372
```

Figure 49: Polynomial Model metrics

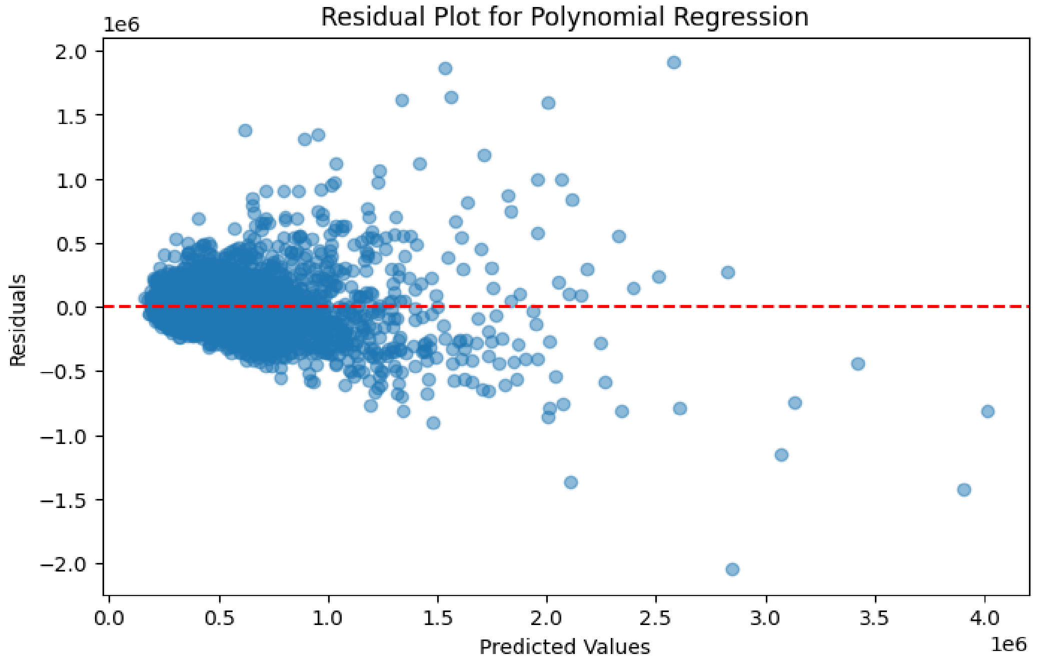


Figure 50: Polynomial Residual Plot

Residual Plot Observation

Polynomial Regression:

- The residuals exhibit a curved pattern, indicating that a linear model might not be capturing the relationships well.
- There's a wider spread of residuals at higher predicted values.

6.2. Non-Parametric Models

6.2.1. Random Forest Regression - Tuned

Random Forest Regression is an ensemble learning method that combines multiple decision trees to improve prediction accuracy and reduce overfitting. Uses bagging (Bootstrap Aggregating) to train each tree on a random subset of data and averages their predictions. Handles nonlinear relationships and high-dimensional data effectively.

To optimize the Random Forest Regressor, we used RandomizedSearchCV to tune hyperparameters. The search space included:

- Number of estimators between 100 and 500
- Maximum depth as one of [None, 10, 20, 30]
- Minimum samples required to split a node between 2 and 20
- Minimum samples required per leaf between 1 and 10

- Maximum number of features as either "sqrt" or "log2"

The RandomizedSearchCV performed 10 iterations with 3-fold cross-validation, optimizing for the R2 score. After training, the best hyperparameter combination was identified.

Using the best-found parameters (max_depth=20,max_features='log2',min_samples_leaf=5, min_samples_split=20,n_estimators=500, random_state=42, bootstrap=True), we trained a final Random Forest model.

Additionally, to handle zeros in the target variable, we applied a log transformation to both the training and test labels, which adds 1 to the values before applying the natural logarithm. After model training and predictions, we reversed the log transformation on the predicted values, ensuring that predictions were in the original scale of the target variable.

```
Random Forest Train RMSE: 176082.06976945815
Random Forest Test RMSE: 188996.7196053877
Random Forest Train MAPE: 18.981092549040188
Random Forest Test MAPE: 23.33004181762581
Random Forest Train R2: 0.7762238680195349
Random Forest Test R2: 0.7107612688215719
Random Forest Train Adj R2: 0.7761450874766797
Random Forest Test Adj R2: 0.7103535074524294
```

Figure 51: Random Forest metrics

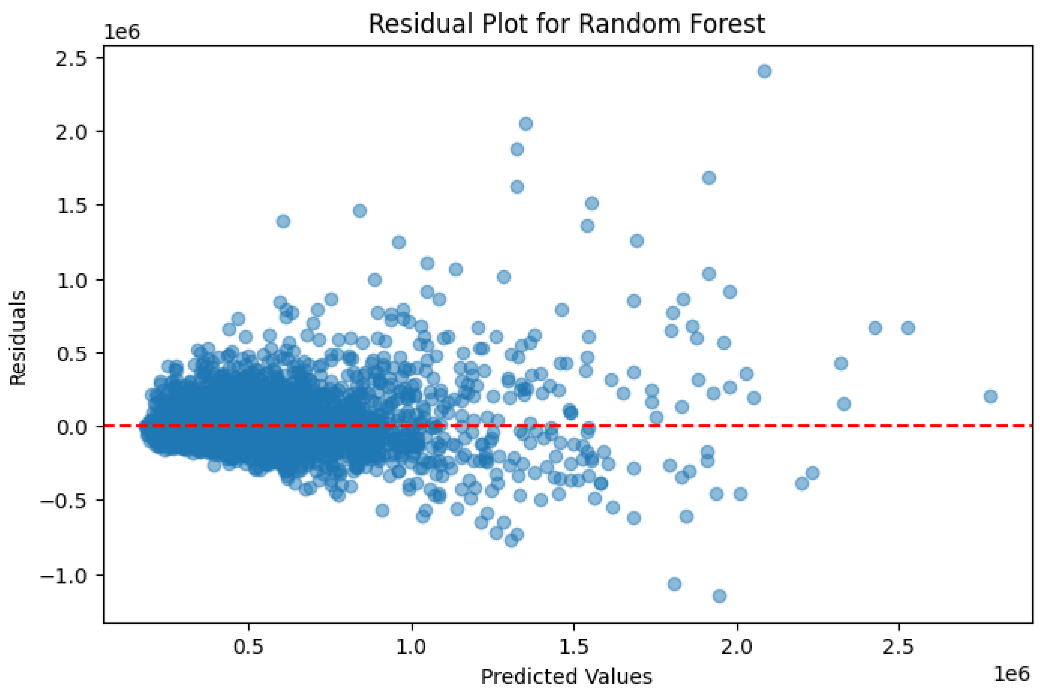


Figure 52: Random Forest Residual Plot

Residual Plot Observation

Random Forest:

- The residuals are more spread out but do not show a strong pattern.
- Still, there are some extreme residuals indicating the model might be overfitting certain data points.

6.2.2. Adaboost Regression

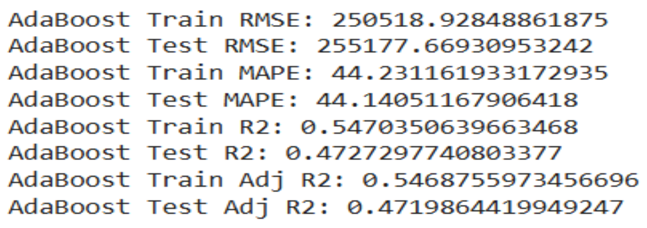
AdaBoost Regression (Adaptive Boosting) is an ensemble method that combines multiple weak learners (typically decision trees) to improve prediction accuracy. Assigns higher weights to incorrectly predicted samples, forcing the model to focus on harder cases in subsequent iterations. Aggregates predictions using a weighted sum of weak learners.

To optimize the AdaBoost Regressor, we used RandomizedSearchCV to tune hyperparameters. The search space included:

- Number of estimators between 50 and 500
- Learning rate sampled from a uniform distribution between 0.01 and 1.0
- Loss function selected from ["linear", "square", "exponential"]
- Base estimator chosen as either a Decision Tree Regressor with a maximum depth of 1 or 3

The RandomizedSearchCV performed 10 iterations with 3-fold cross-validation, optimizing for the R2 score. After training, the best hyperparameter combination was identified.

Using the best-found parameters (n_estimators=210, estimator=DecisionTreeRegressor(max_depth=3), learning_rate=0.1918, loss='linear'), we trained a final AdaBoost model.



```
AdaBoost Train RMSE: 250518.92848861875
AdaBoost Test RMSE: 255177.66930953242
AdaBoost Train MAPE: 44.231161933172935
AdaBoost Test MAPE: 44.14051167906418
AdaBoost Train R2: 0.5470350639663468
AdaBoost Test R2: 0.4727297740803377
AdaBoost Train Adj R2: 0.5468755973456696
AdaBoost Test Adj R2: 0.4719864419949247
```

Figure 53: Adaboost metrics

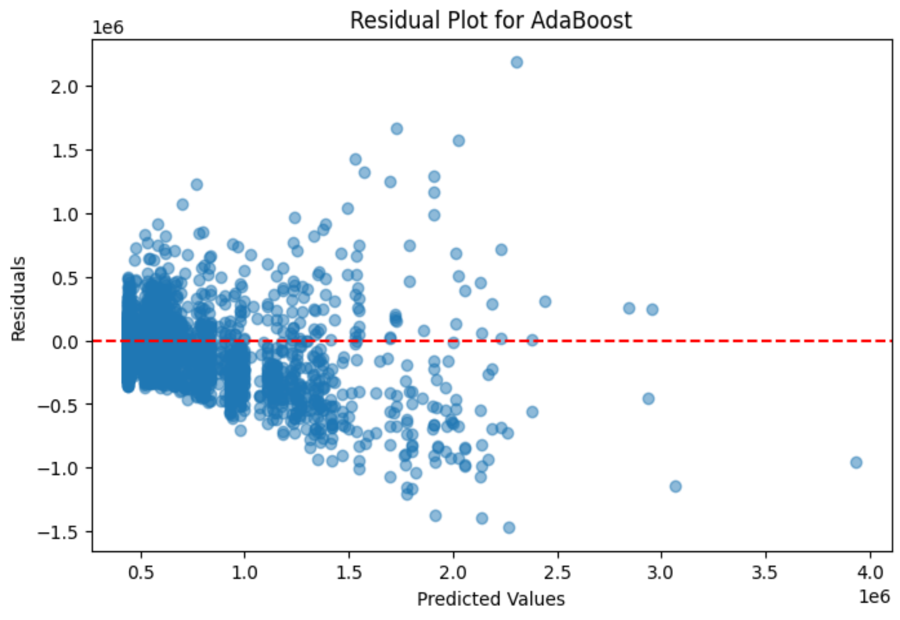


Figure 54: Adaboost Residual Plot

Residual Plot Observation

AdaBoost:

- Residuals appear skewed, with a significant number of negative residuals.
- It seems to struggle with extreme values, possibly due to the boosting method aggressively minimizing error on previous mistakes.

6.2.3. XGBoost Regression

XGBoost (Extreme Gradient Boosting) is an optimized, fast, and scalable gradient boosting algorithm. Uses boosted decision trees with advanced regularization (L1 & L2) to prevent overfitting. Employs parallel processing, making it much faster than traditional boosting methods like AdaBoost or standard Gradient Boosting.

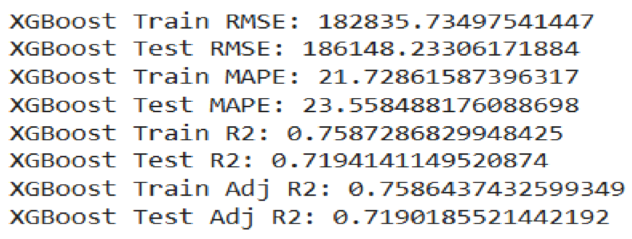
To optimize the XGBoost Regressor, we used RandomizedSearchCV to tune hyperparameters. The search space included:

- Number of estimators between 100 and 500
- Learning rate sampled from a uniform distribution between 0.01 and 0.3
- Maximum depth between 3 and 10
- Subsample ratio sampled from a uniform distribution between 0.5 and 1.0
- Column sampling ratio sampled from a uniform distribution between 0.5 and 1.0
- L2 regularization term sampled from a uniform distribution between 0 and 10

The RandomizedSearchCV performed 10 iterations with 3-fold cross-validation, optimizing for the R2 score. After training, the best hyperparameter combination was identified.

Using the best-found parameters (colsample_bytree=0.8, learning_rate=0.03, max_depth=5, n_estimators=500, reg_lambda=10, reg_alpha=2, subsample=0.8, random_state=42, min_child_weight=5), we trained a final XGBoost model.

Additionally, to handle zeros in the target variable, we applied a log transformation to both the training and test labels, which adds 1 to the values before applying the natural logarithm. After model training and predictions, we reversed the log transformation on the predicted values, ensuring that predictions were in the original scale of the target variable.



```
XGBoost Train RMSE: 182835.73497541447
XGBoost Test RMSE: 186148.23306171884
XGBoost Train MAPE: 21.72861587396317
XGBoost Test MAPE: 23.558488176088698
XGBoost Train R2: 0.7587286829948425
XGBoost Test R2: 0.7194141149520874
XGBoost Train Adj R2: 0.7586437432599349
XGBoost Test Adj R2: 0.7190185521442192
```

Figure 55: XGBoost Metrics

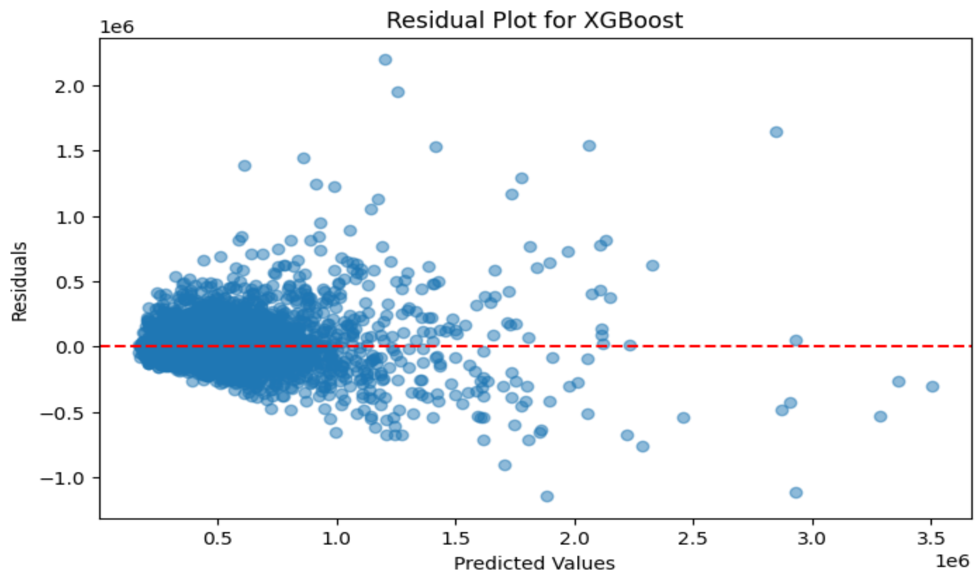


Figure 56: XGBoost Residual Plot

Residual Plot Observation

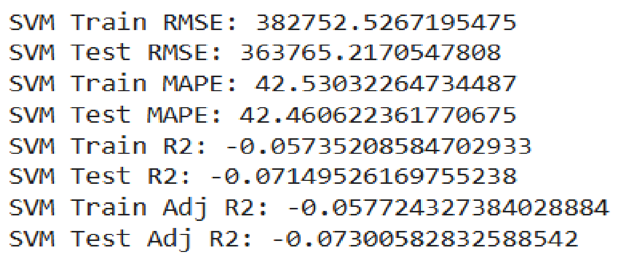
XGBoost:

- The spread of residuals is less pronounced compared to the linear models.
- However, some extreme values are still present, indicating that it might be slightly overfitting.

6.2.4. SVM Regression

Support Vector Machine (SVM) Regression finds the best hyperplane that fits the data while minimizing errors within a margin of tolerance (ε-insensitive loss function). Uses kernel tricks (linear, polynomial, RBF, etc.) to capture nonlinear relationships in data.

A Support Vector Machine (SVM) Regressor was trained using the default settings of the SVR() class. Further performance improvements could be achieved by tuning hyperparameters such as kernel type, regularization (C), epsilon, and gamma through GridSearchCV or RandomizedSearchCV.



```
SVM Train RMSE: 382752.5267195475
SVM Test RMSE: 363765.2170547808
SVM Train MAPE: 42.53032264734487
SVM Test MAPE: 42.460622361770675
SVM Train R2: -0.05735208584702933
SVM Test R2: -0.07149526169755238
SVM Train Adj R2: -0.057724327384028884
SVM Test Adj R2: -0.07300582832588542
```

Figure 57: SVM Metrics

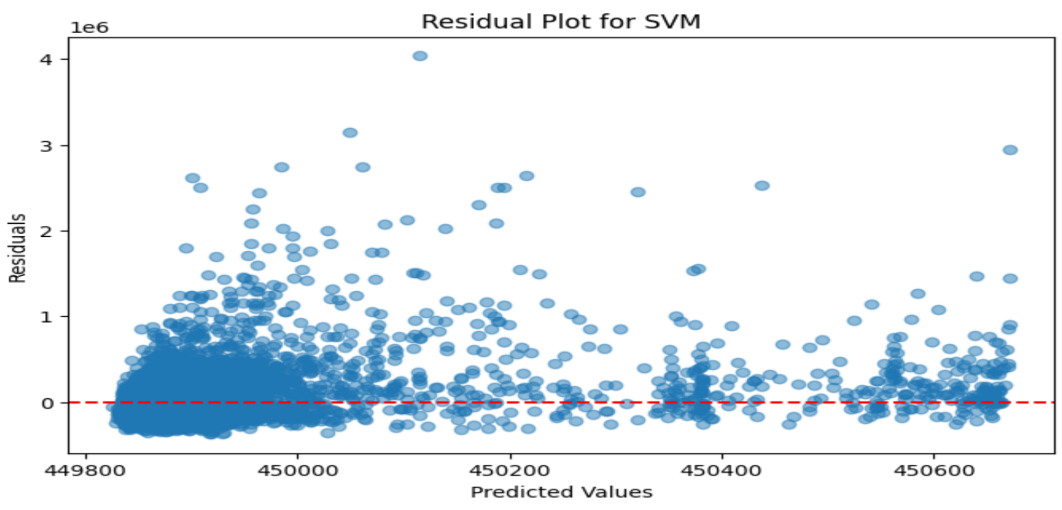


Figure 58: SVM Residual Plot

Residual Plot Observation

SVM:

- The residuals are highly concentrated in a very narrow range.
- This suggests the model is making predictions within a small range, which could indicate underfitting.
- There's a systematic pattern in the residuals, meaning the model is not capturing some nonlinear relationships effectively.

7. Model Comparison

Model	Train RMSE	Test RMSE	Train MAPE	Test MAPE	Train R²	Test R²	Train Adj R²	Test Adj R²	Residual Plot Observations
OLS	233678.2	222015.5	31.088	31.276	0.606	0.601	0.606	0.6	Biased
Lasso	233678.2	222015.6	31.089	31.277	0.606	0.601	0.606	0.6	Biased
Ridge	233678.2	222015.4	31.088	31.276	0.606	0.601	0.606	0.6	Biased
Polynomial	203696.1	203956.9	27.112	27.785	0.701	0.663	0.7	0.663	Biased
Random Forest	176082.1	188996.7	18.981	23.33	0.776	0.711	0.776	0.71	Not Biased
Adaboost	250518.9	255177.7	44.231	44.141	0.547	0.473	0.547	0.472	Biased
XGBoost	182835.7	186148.2	21.729	23.558	0.759	0.719	0.759	0.719	Not Biased
SVM	382752.5	363765.2	42.53	42.461	-0.057	-0.071	-0.058	-0.073	Biased

Table 5: Model Comparison

Observations

- **Random Forest** performed the best, achieving the lowest RMSE on the test data and demonstrating a relatively well-distributed residual pattern. However, there are slight signs of overfitting.
- XGBoost was the second-best model, delivering a competitive test RMSE while generalizing better than Random Forest.
- Polynomial Regression exhibited significant overfitting.
- SVM and AdaBoost underperformed, showing clear indications of either underfitting or high variance.

8. Random Forest – Model Interpretation

After selecting Random Forest as the best-performing model, we conducted a feature importance analysis to identify the most influential variables in predicting house prices.

Using the feature_importances_ attribute of the trained Random Forest model, we ranked features based on their contribution to price prediction. The results were visualized in a horizontal bar chart, making it easier to interpret the most impactful predictors. The top-ranked features included:

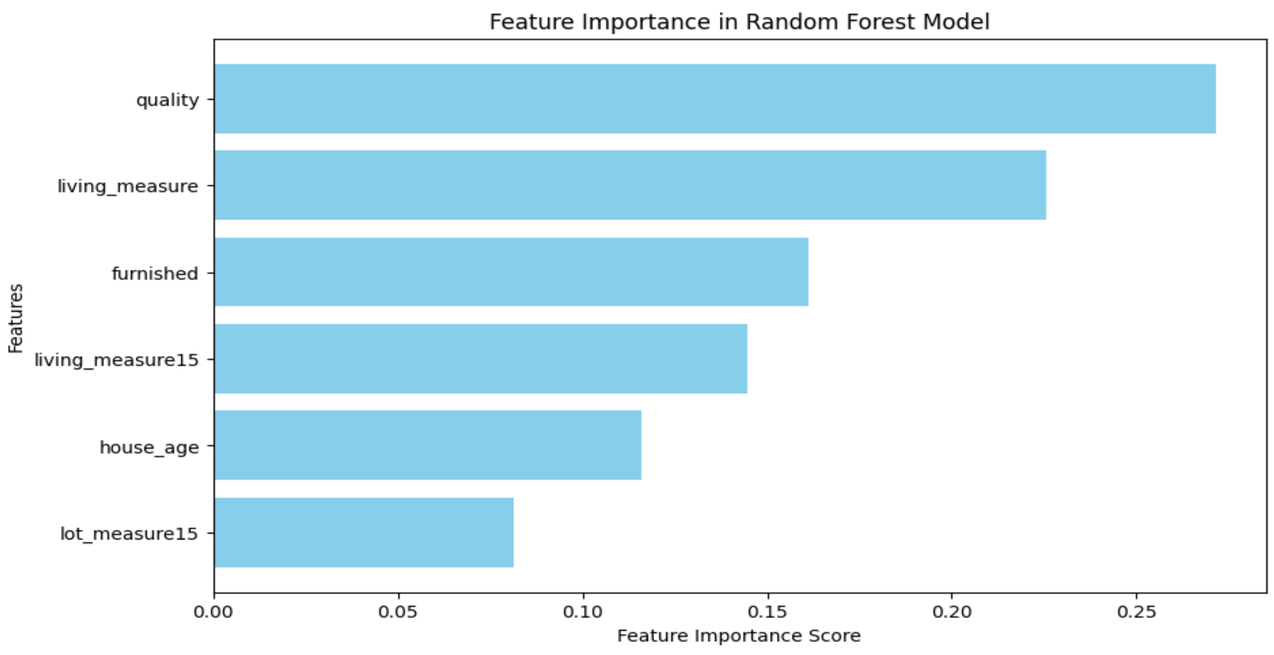
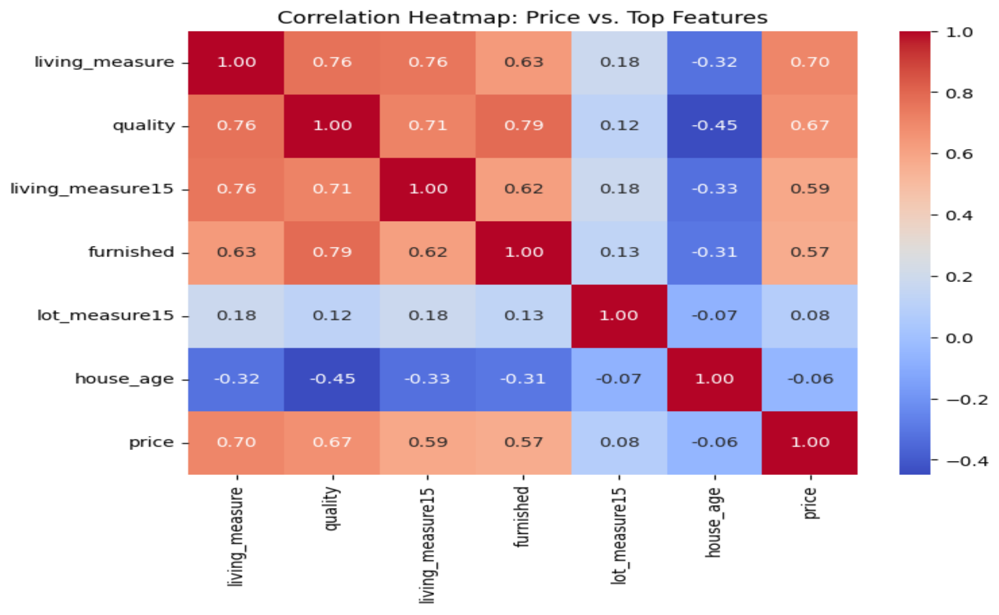


Figure 59: Random Forest Feature Importance

To better understand the relationship between the target variable (price) and the most important features, we performed bivariate analysis using correlation analysis, scatter plots (for continuous variables), and box plots (for categorical variables).

Heatmap



	living_measure	quality	living_measure15	furnished	lot_measure15	house_age	price
living_measure	1.00	0.76	0.76	0.63	0.18	-0.32	0.70
quality	0.76	1.00	0.71	0.79	0.12	-0.45	0.67
living_measure15	0.76	0.71	1.00	0.62	0.18	-0.33	0.59
furnished	0.63	0.79	0.62	1.00	0.13	-0.31	0.57
lot_measure15	0.18	0.12	0.18	0.13	1.00	-0.07	0.08
house_age	-0.32	-0.45	-0.33	-0.31	-0.07	1.00	-0.06
price	0.70	0.67	0.59	0.57	0.08	-0.06	1.00

Figure 60: Heatmap

Price vs Living Measure

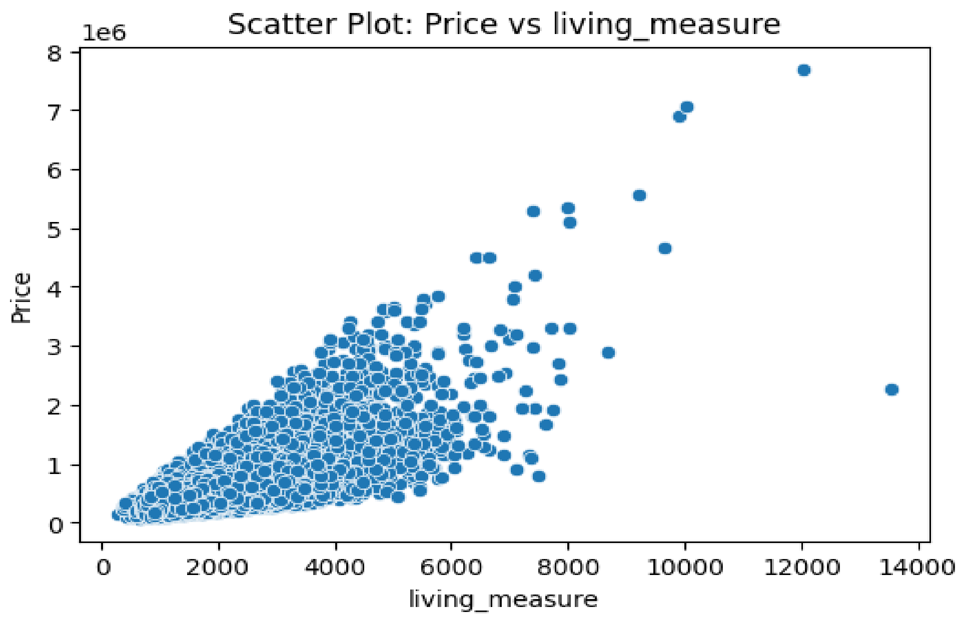


Figure 61: Price vs living measure

- Observation: There is a strong positive correlation between living_measure and price. Larger homes tend to have higher prices.
- Recommendation: Focus on marketing larger properties as premium homes. If applicable, suggest home extensions to increase property value.

Price vs House Age

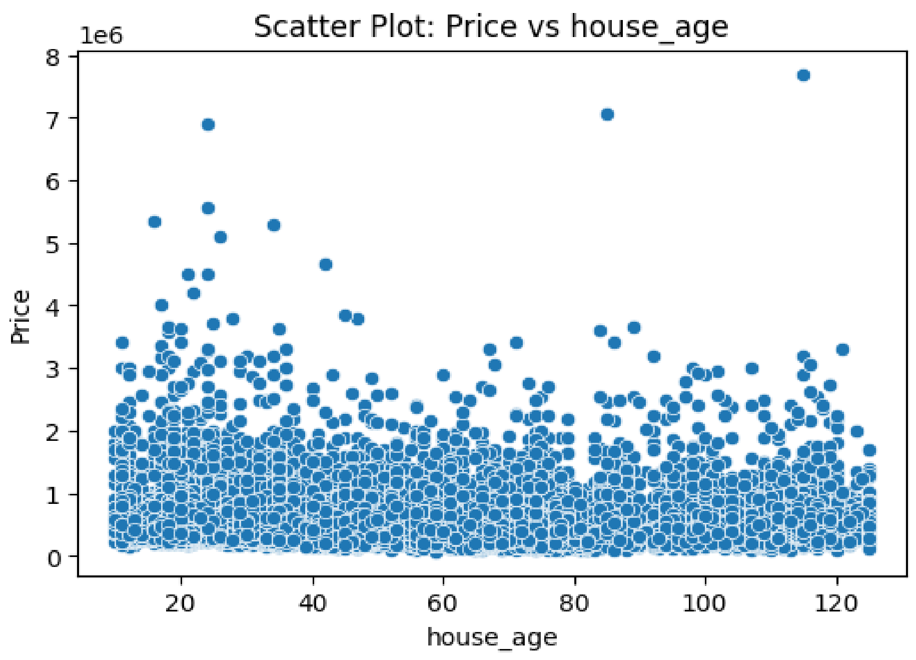


Figure 62: Price vs House age

- Observation: There is no clear trend between house_age and price. Some older homes still fetch high prices.
- Recommendation: Focus on marketing well-maintained older homes with renovation histories. Consider modernizing aging properties to maintain their value.

Price vs lot measure(2015)

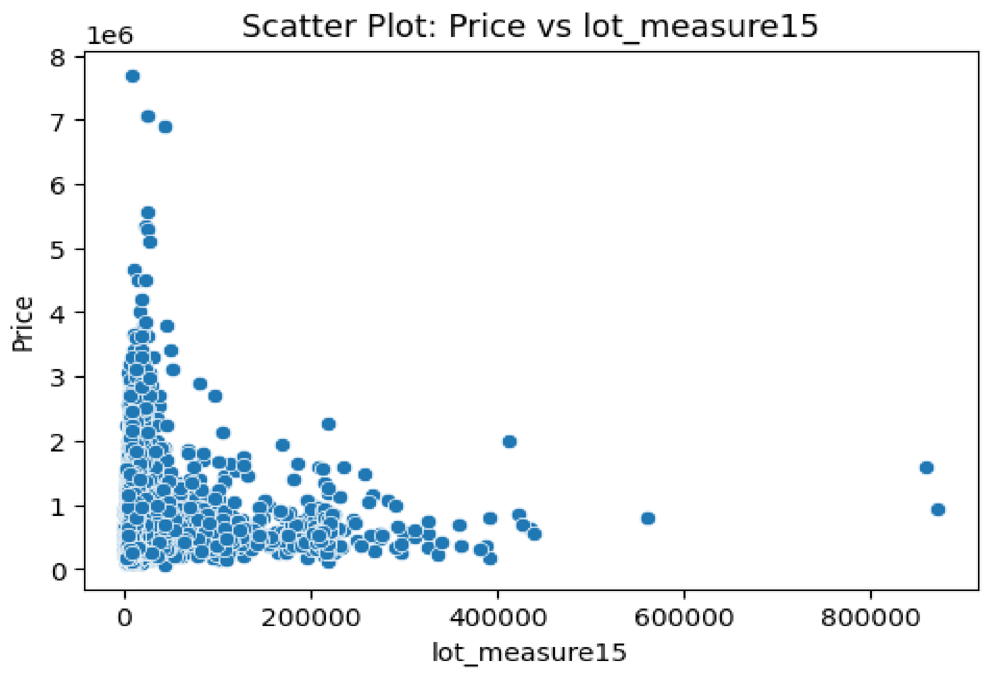


Figure 63: Price vs lot measure

- Observation: The correlation between lot size and price is weak (0.08), indicating that lot size has little impact on property price. Higher lot sizes generally do not correspond to higher property prices, and in fact, many high-value properties have relatively small lot sizes. There are a few exceptions where very large lots correspond to high prices, which could indicate unique property features such as commercial zoning, estate properties, or land development potential.

- Recommendation: Focus on location and property features rather than lot size alone when valuing properties. If large lots are undervalued, explore opportunities for subdivision or development to maximize returns. Highlight premium properties that balance lot size with desirable features such as location, amenities, or architectural quality.

Price vs Living measure(2015)

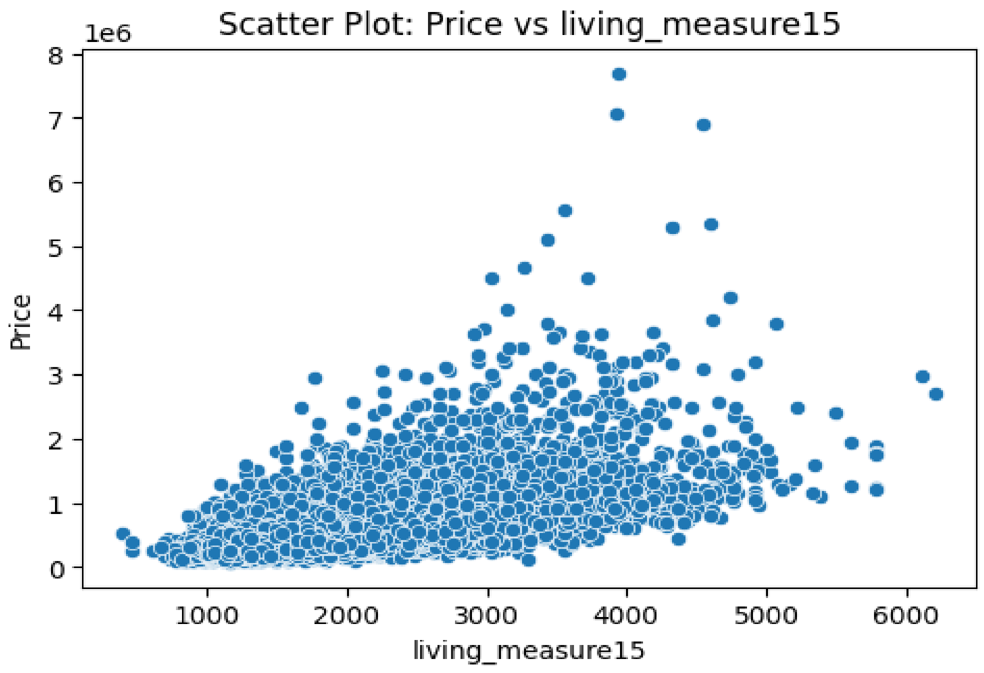


Figure 64: Price vs Living measure - 2015

- Observation: Similar to living_measure, a larger living_measure15 also correlates with higher prices.
- Recommendation: If applicable, ensure that properties maintain expansion potential for future value appreciation.

Price vs Quality

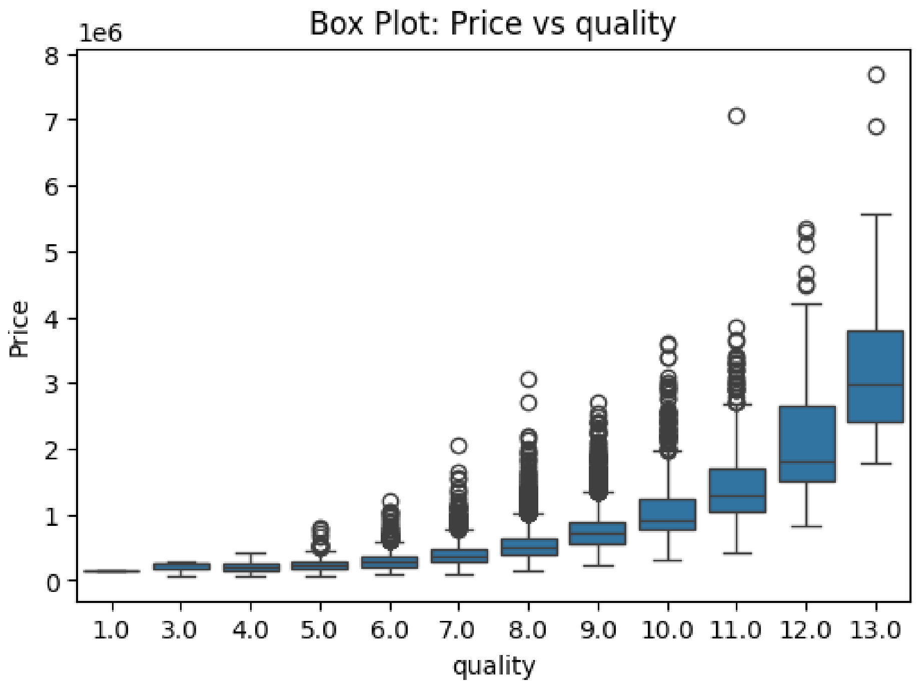


Figure 65: Price vs Quality

- Observation: As quality increases, price rises significantly. Higher-quality homes command premium pricing.

- Recommendation: Highlight high-quality materials and construction in listings. Invest in home improvements that increase quality scores (e.g., better finishes, energy-efficient upgrades).

Price vs Furnished

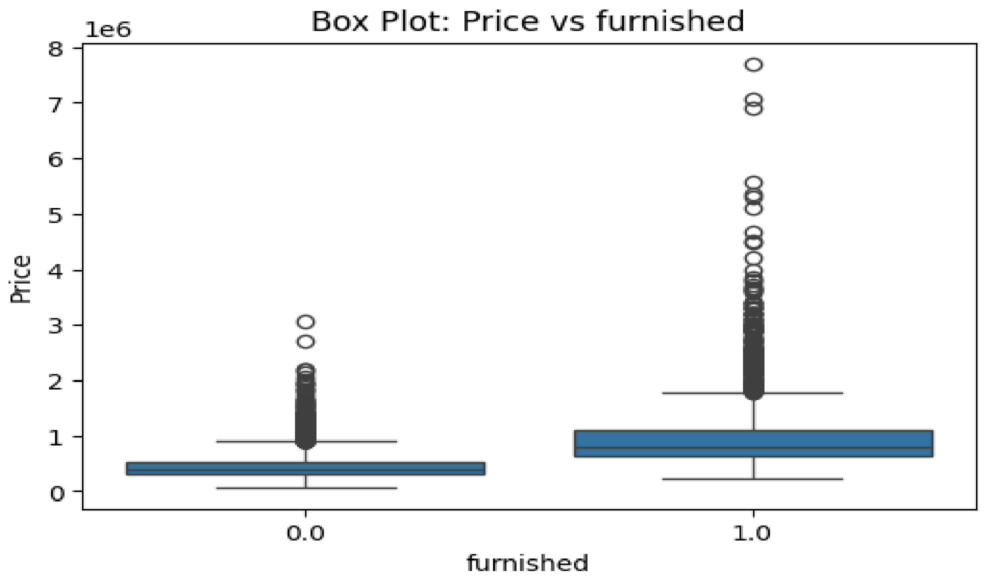


Figure 66: Price vs Furnished

- Observation: Furnished homes generally have higher median prices, though there are some high-priced unfurnished homes as well.
- Recommendation: Offering furnished options can be a great value-add. Staging properties with modern interiors can help boost perceived value and demand.

9. Feature Impact on Price using Linear Regression

The unit change impact on the target variable price, for each feature is derived from the coefficients of a linear regression model.

Feature	**Coefficient**	**Unit Change Impact on Price**
quality	1,32,006.82	A 1 unit increase in quality leads to an increase in price by 132,006.82 units (e.g., dollars).
furnished	25,819.00	A 1 unit increase in furnished (likely binary: 0 or 1) leads to an increase in price by 25,818.99 units.
house_age	3,697.69	A 1 unit increase in house_age (likely 1 year) leads to an increase in price by 3,697.69 units.
living_measure	178.101	A 1 unit increase in living_measure leads to an increase in price by 178.10 units.
living_measure15	20.074	A 1 unit increase in living_measure15 leads to an increase in price by 20.07 units.
lot_measure15	-0.524	A 1 unit increase in lot_measure15 leads to a decrease in price by 0.52 units.

10. Recommendations

- Key Price-Driving Features

The strongest factors influencing house prices include:

1. Living Area (Living Measure & Living Measure15)
   - Direct positive correlation with price—larger homes command higher prices.
2. Quality of Construction (Quality)
   - Homes with higher quality ratings (8-12) significantly increase in price.
   - As shown in the box plot, quality is a strong differentiator in price.
3. Furnishing Status (Furnished)
   - Furnished homes tend to have a higher median price than unfurnished ones.
4. House Age (House Age)
   - Minimal impact on price—older homes can maintain value if well-maintained.
5. Lot Area (LotMeasure15)
   - Larger lots don't always mean higher prices. Focus on location and unique property features.

For Sellers:

1. Price Adjustments Based on Key Features

1. Quality Adjustments
   - High Quality (8+ rating): Increase price by 10-20%.
   - Low Quality (below 6 rating): Reduce price by 5-15%.
2. Furnishing Adjustments
   - Fully furnished homes sell for 5-10% more than unfurnished ones.
3. Living Space (Neighborhood Comparison – Living Measure15)
   - If neighboring homes are larger: Adjust downward to remain competitive.
   - If neighboring homes are smaller: A larger house justifies a premium price.
4. House Age Adjustments
   - Older homes (50+ years): Reduce price by 5-10%, unless well-maintained.
   - Recent renovations (new kitchen, flooring, etc.): Justify a 5-15% price increase.
5. Ceiling Area Adjustments
   - High ceilings (spacious design): Increase price by 5-10% due to luxury appeal.
   - Lower ceiling height compared to nearby homes: Adjust price to remain competitive.

2. Using Data & Market Trends for Pricing

- Benchmark against recently sold properties in the area.
- Adjust based on demand:
  - If demand is high, list at the upper price range.

    - If competition is strong, price competitively to attract buyers.

3. Monitoring Market Response & Adjusting Price

- If there's no buyer interest within 2-3 weeks, consider small price reductions.
- Highlight key features (quality, space, furnishing) in marketing to attract the right buyers.

For Buyers:

1. Prioritize High-Value Homes:

- High-Quality Homes (8+ rating): These homes are worth the premium price. Investing in homes with superior construction quality ensures long-term value retention and resale potential.
- Low-Quality Homes (<6 rating): Negotiate for a 5-15% discount. Homes with lower ratings may need significant repairs or upgrades, so it's essential to factor in those costs when negotiating.
- Larger Homes: They tend to have higher resale value. Focus on homes with more space or larger living areas (Living Measure15) as these will be easier to sell in the future, especially if the neighborhood is desirable.

2. Look for Future Growth Potential:

- Bigger Homes in Expanding Areas: Invest in larger homes in areas that are experiencing growth. These properties will likely appreciate in value over time as the area becomes more developed.
- Large Lots: Consider properties with larger lots as they offer potential for subdivision or conversion into commercial use, which could significantly increase the property's future value.

3. Smart Negotiation & Market Awareness:

- Older, Renovated Homes: Homes that are older but have been recently renovated often provide better value than new builds. Pay attention to updates such as new kitchens, flooring, and other significant renovations. These homes can offer a lot of value for a reasonable price.
- Low Demand: If demand is low in a particular area or for a specific property, this is a prime opportunity to negotiate a better price or request additional upgrades, such as new appliances or repairs.
- Recent Price Drops: If a home has had a recent price drop, it may indicate seller flexibility. Use this to your advantage by negotiating for repairs or asking the seller to cover closing costs. It could be a great opportunity to secure a deal below market value.