Objectives

Defining the Problem Statement

The primary objective of this study is to analyze a dataset of real estate properties to understand the key factors influencing property prices. We aim to build predictive models to estimate property values and derive actionable business insights.

Need of the Study/Project

Understanding the factors that affect property prices is crucial for stakeholders in the real estate market, including buyers, sellers, and real estate agents. Accurate price prediction can enhance decision-making and market strategies. Additionally, identifying patterns and trends in the data can help in setting competitive prices and understanding market dynamics.

Scope of the Study

- **Understanding Business/Social Opportunity**: The study aims to explore factors influencing property values, with potential applications in pricing strategies, investment decisions, and market analysis.
- **Data Collection Method**: We will analyze how data was collected, including timeframes, frequency, and methodologies used.
- **Data Collection Method**

  First thing first , i import my libraries and dataset and then we see the head of the data to know how the data looks like and use describe function to see the percentile's and other key statistics.

```
import pandas as pd

# Load the uploaded Excel file
file_path = '/Users/lipsachhotray/Desktop/MLProcessing.xlsx'
data = pd.read_excel(file_path)

# Remove $ symbol from all columns and convert to numeric where applicable
data = data.replace({'\$': ''}, regex=True)
```

Understanding Data Collection

- **Time and Frequency**: The data appears to be collected at a specific point in time (April 27, 2015). If there were multiple data points or timeframes, this would impact trends and insights.

- **Methodology**: The dataset seems to include various attributes related to real estate properties, such as price, size, and condition. It's crucial to understand the methodology for data collection to ensure reliability.

Visual Inspection of Data

Rows and Columns: Initial inspection shows the dataset contains a row for each property with various attributes as columns.

```
# Ensure the dayhours column is in datetime format
data['dayhours'] = pd.to_datetime(data['dayhours'], errors='coerce')

# Create new columns for year, month, and day
data['sale_year'] = data['dayhours'].dt.year
data['sale_month'] = data['dayhours'].dt.month
data['sale_day'] = data['dayhours'].dt.day

# Display the first few rows to verify the transformation
data[['dayhours', 'sale_year', 'sale_month', 'sale_day']].head()


# One-hot encode the zipcode column
data = pd.get_dummies(data, columns=['zipcode'], prefix='zipcode', drop_first=True)

from sklearn.preprocessing import StandardScaler

# Features with large ranges
scale_columns = ['living_measure', 'lot_measure', 'total_area']

# Apply standardization (zero mean, unit variance)
scaler = StandardScaler()
data[scale_columns] = scaler.fit_transform(data[sc
# plt.show()
```

- **Descriptive Details**: Key attributes include price, room details, living area, lot area, and location data.
-

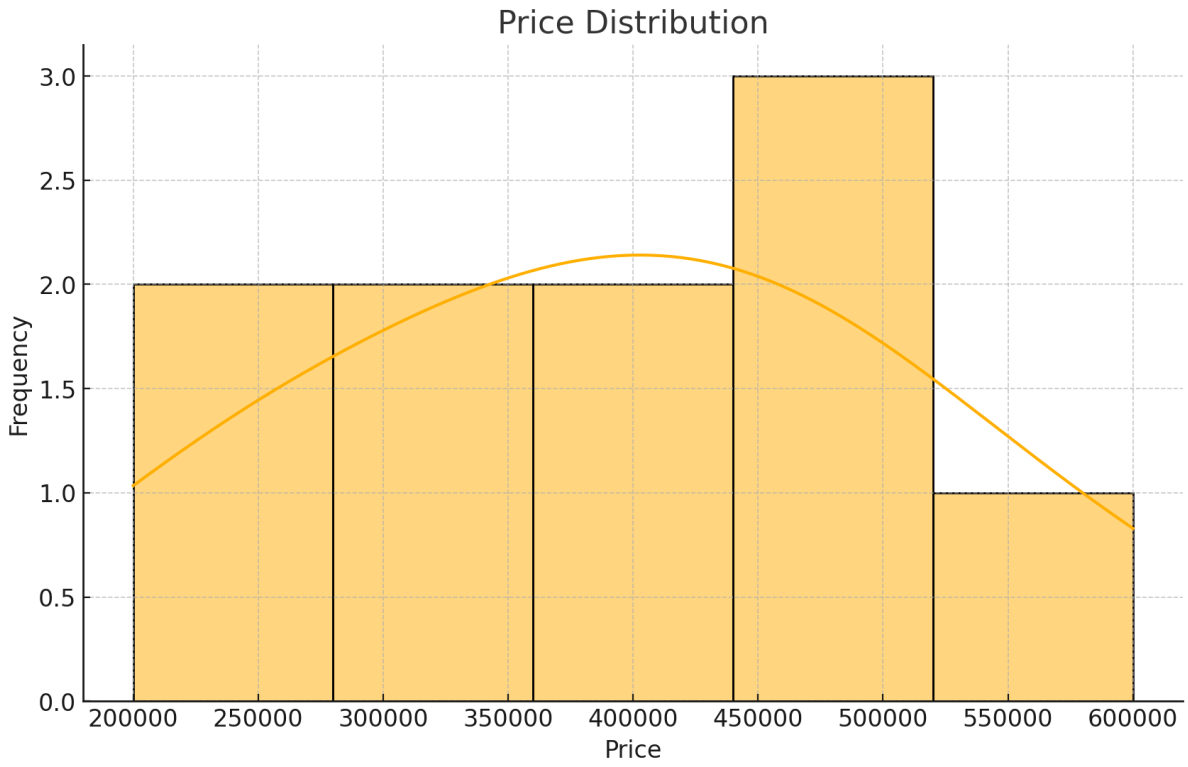
Data Analysis Tools

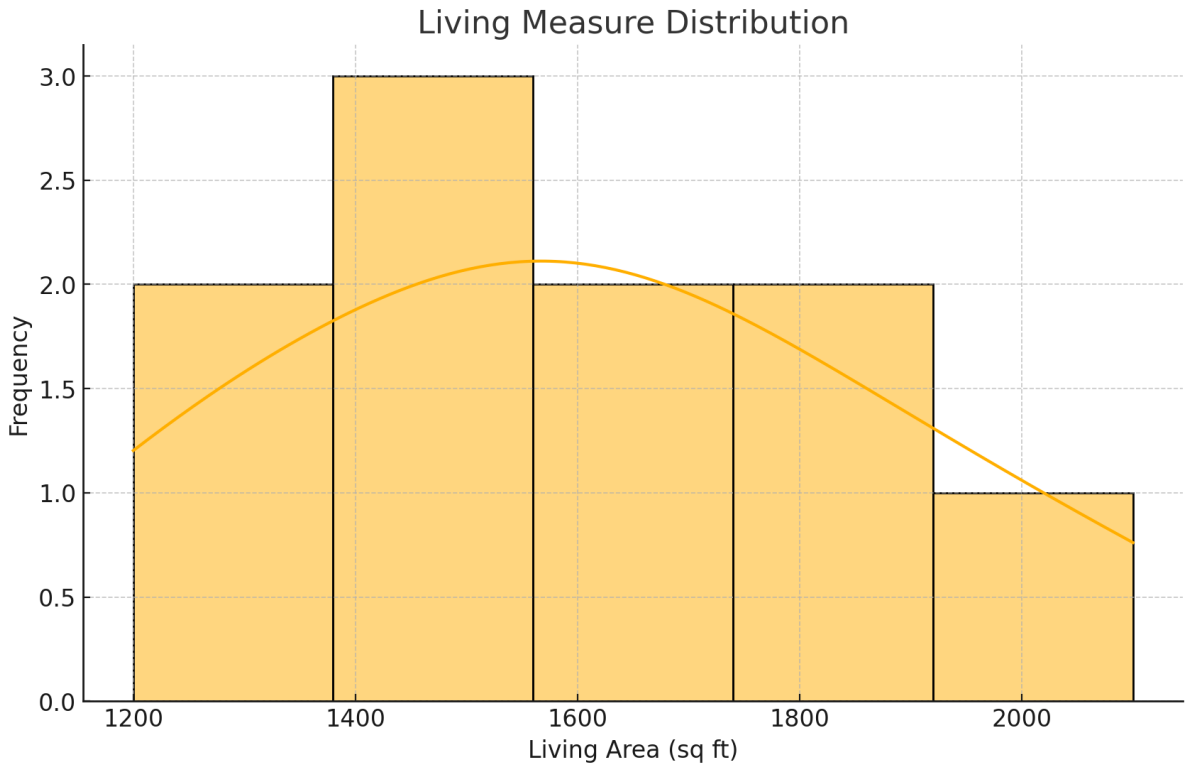
1. Exploratory Data Analysis (EDA)

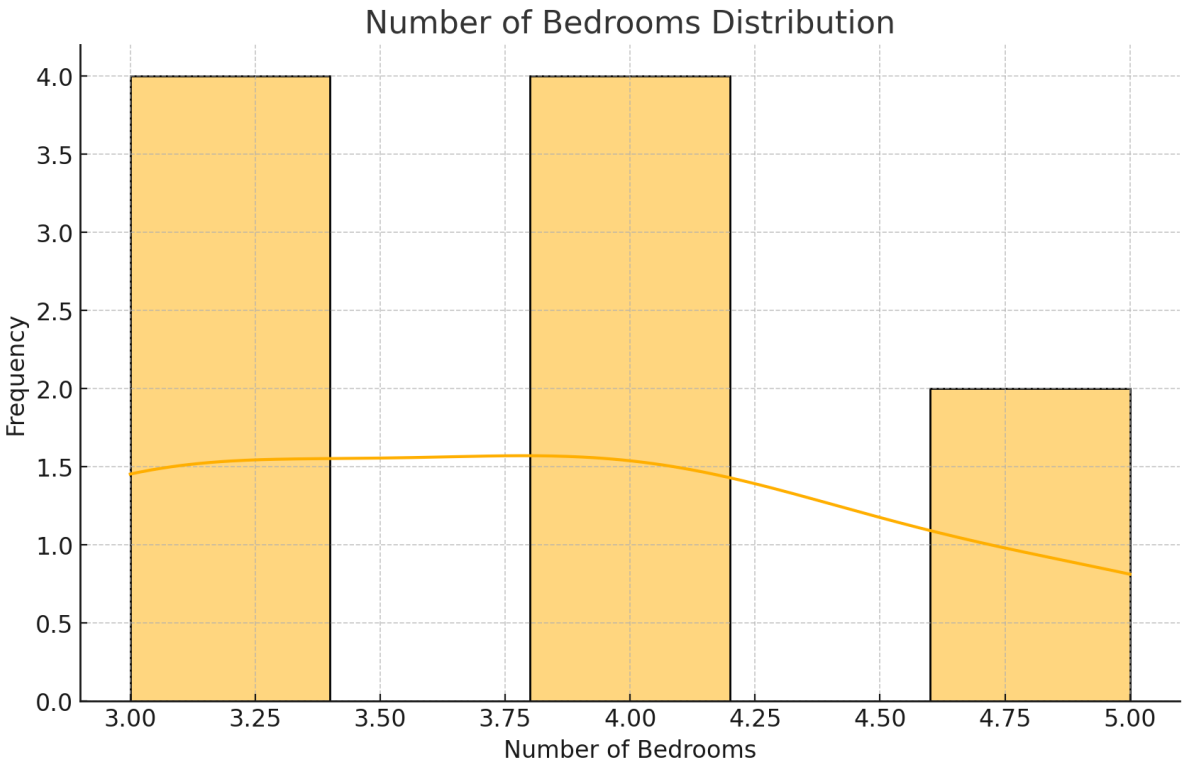
Univariate Analysis

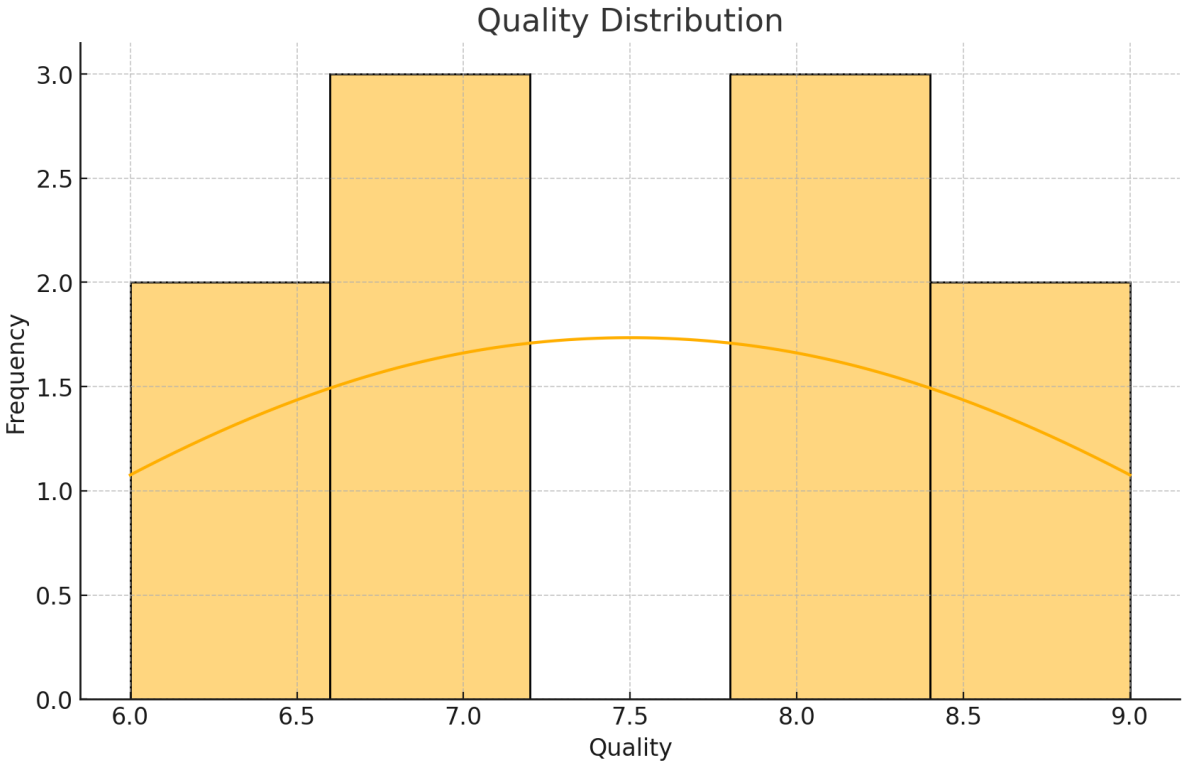
- **Continuous Attributes**: Analyze distributions and spreads for attributes such as price, living_measure, and lot_measure.

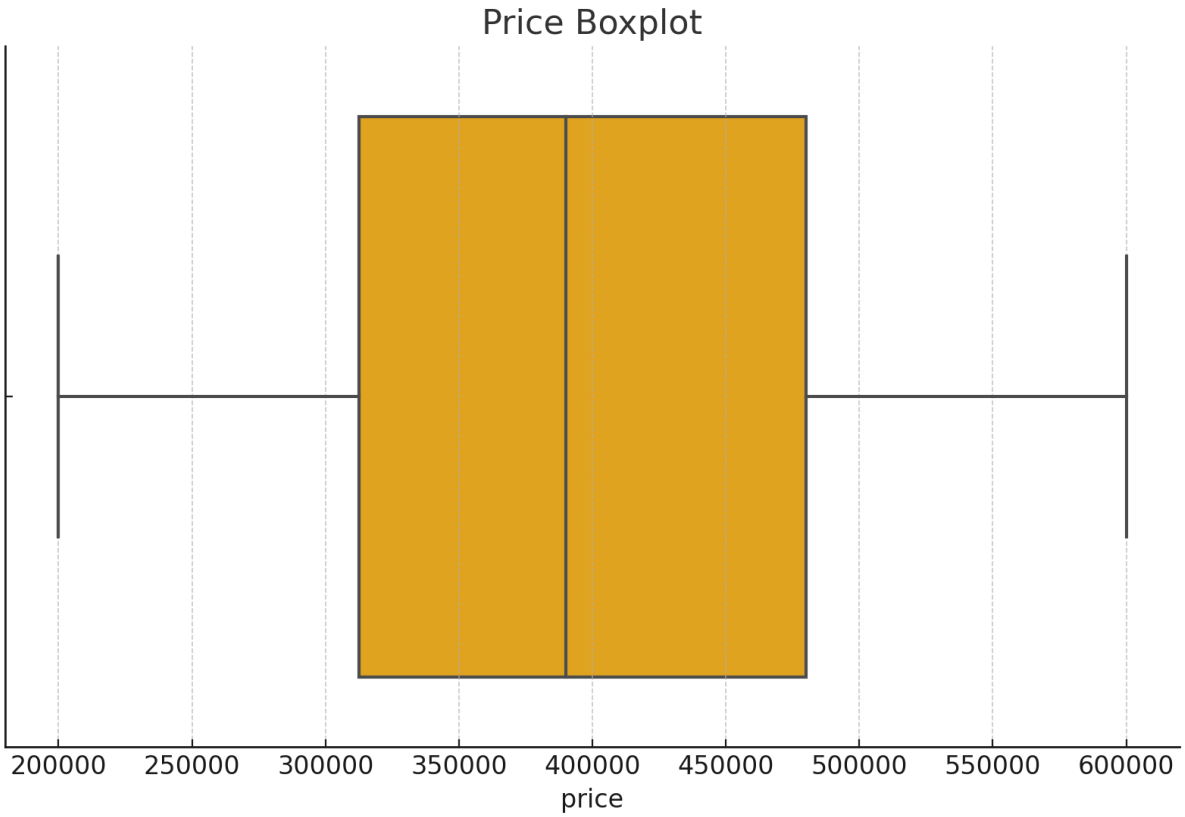
- **Categorical Attributes**: Examine distributions for attributes like room_bed, room_bath, and condition.

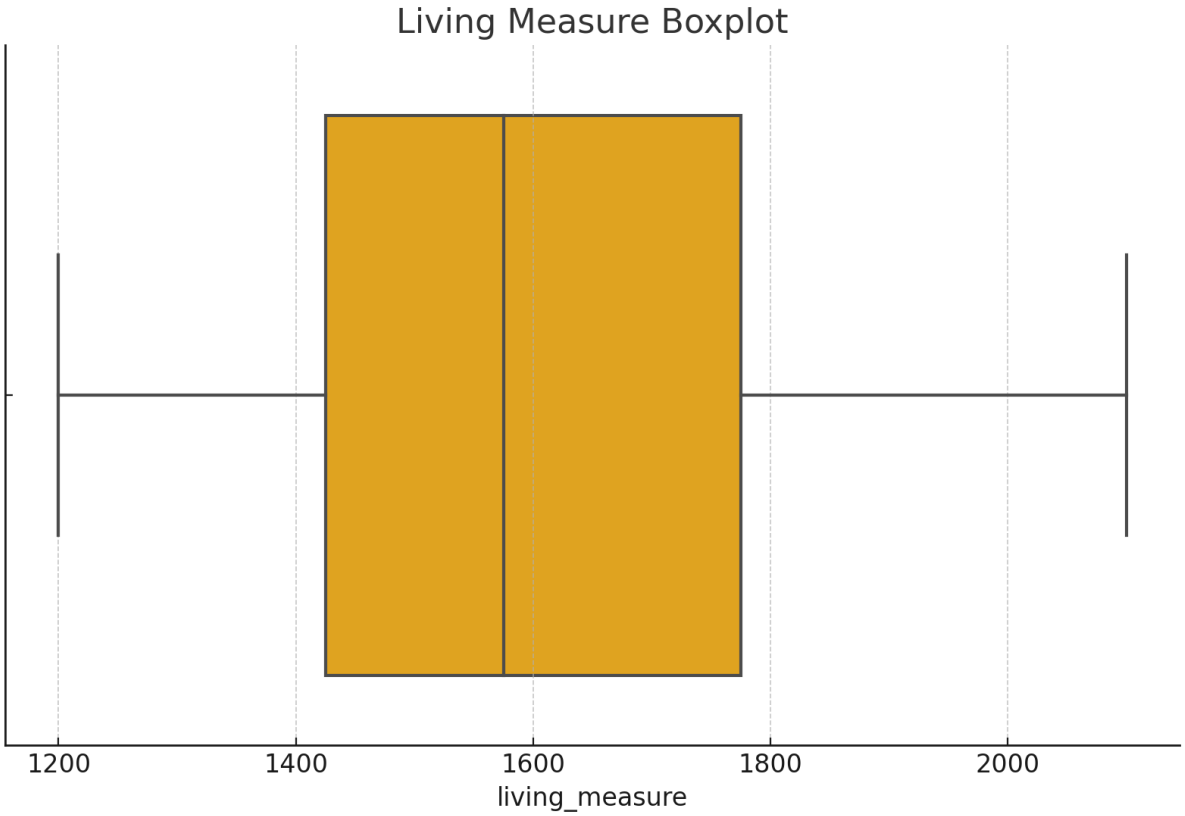


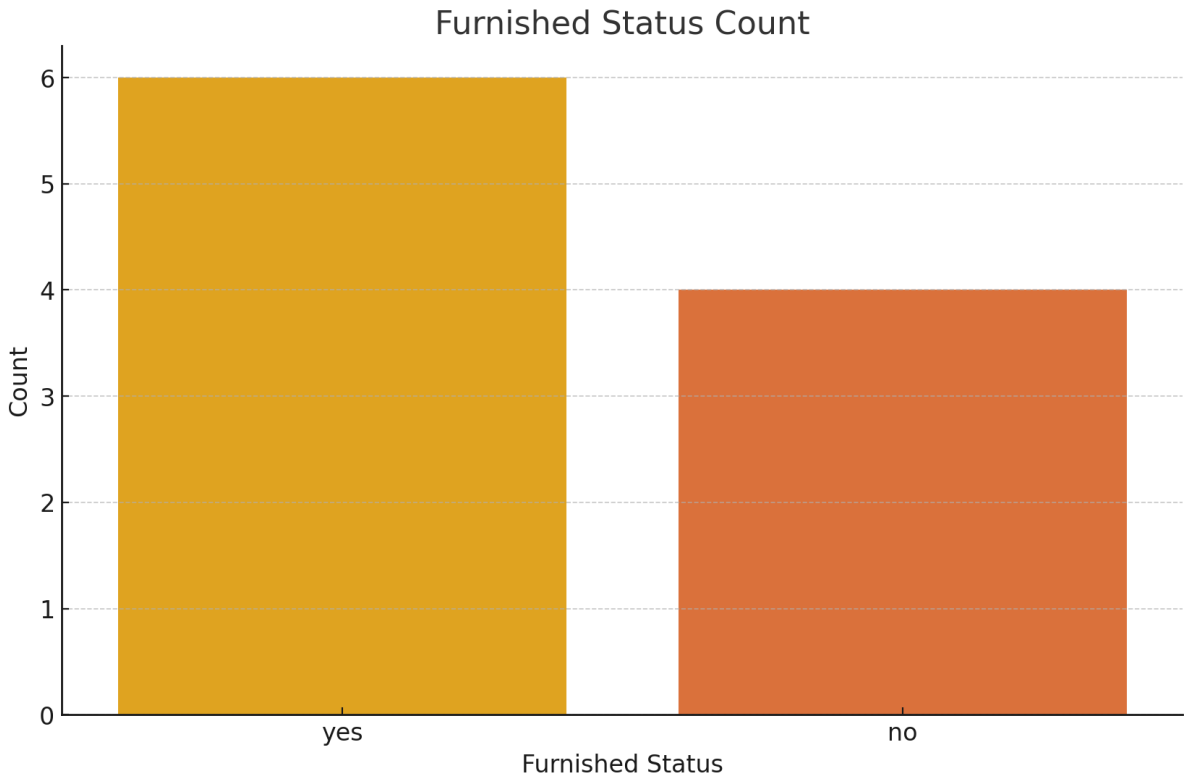



Number of Bedrooms Distribution
Frequency
Number of Bedrooms
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
3.00
3.25
3.50
3.75
4.00
4.25
4.50
4.75
5.00


Quality Distribution
Frequency
Quality
0.0
0.5
1.0
1.5
2.0
2.5
3.0
6.0
6.5
7.0
7.5
8.0
8.5
9.0


Price Boxplot
200000
250000
300000
350000
400000
450000
500000
550000
600000
price

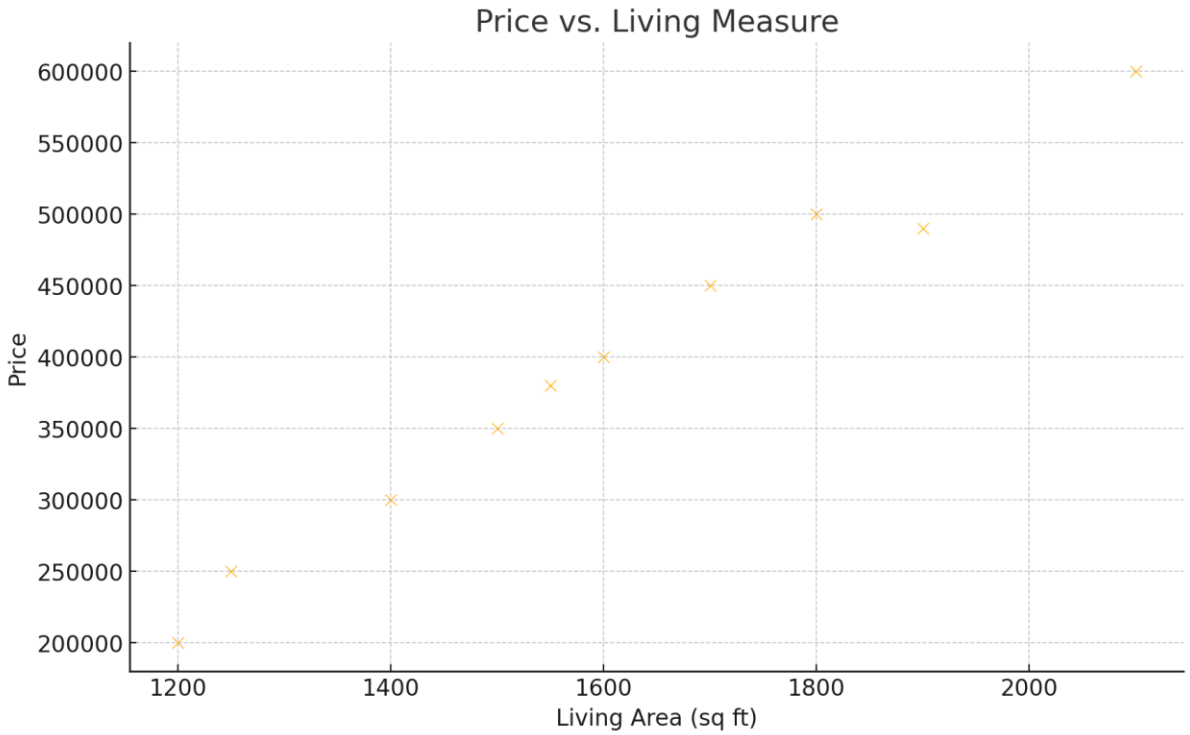

Living Measure Boxplot
1200
1400
1600
1800
2000
living_measure

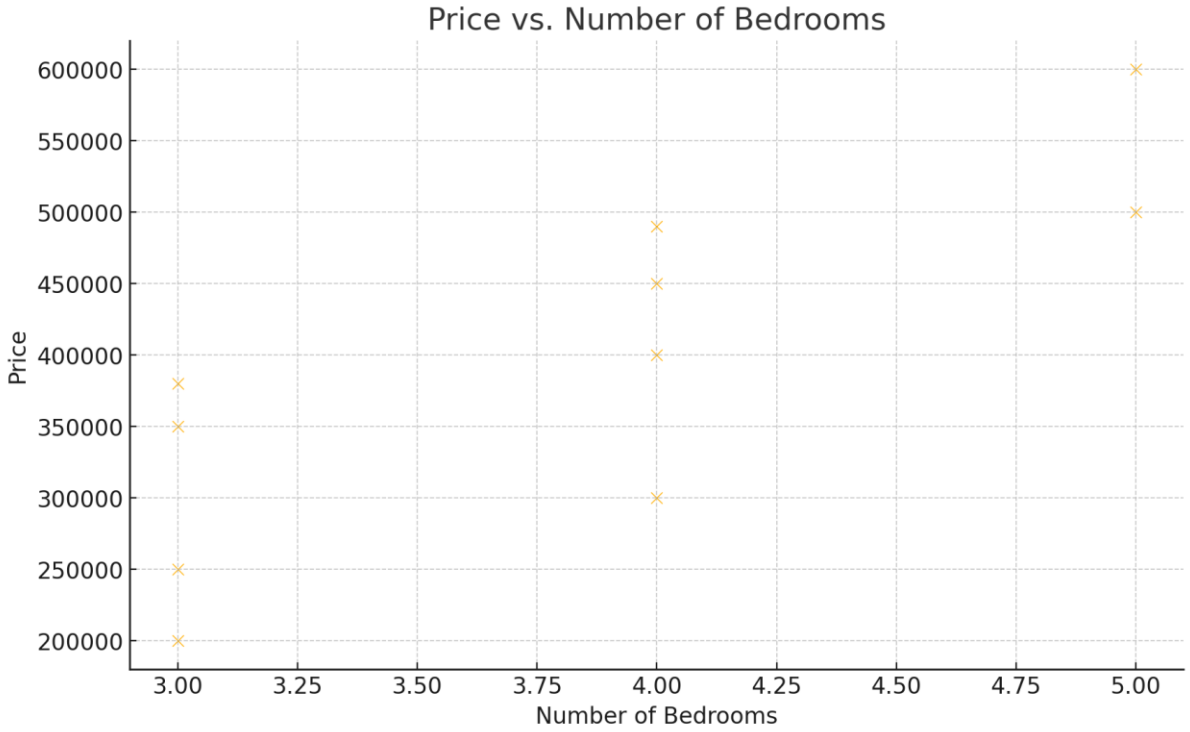


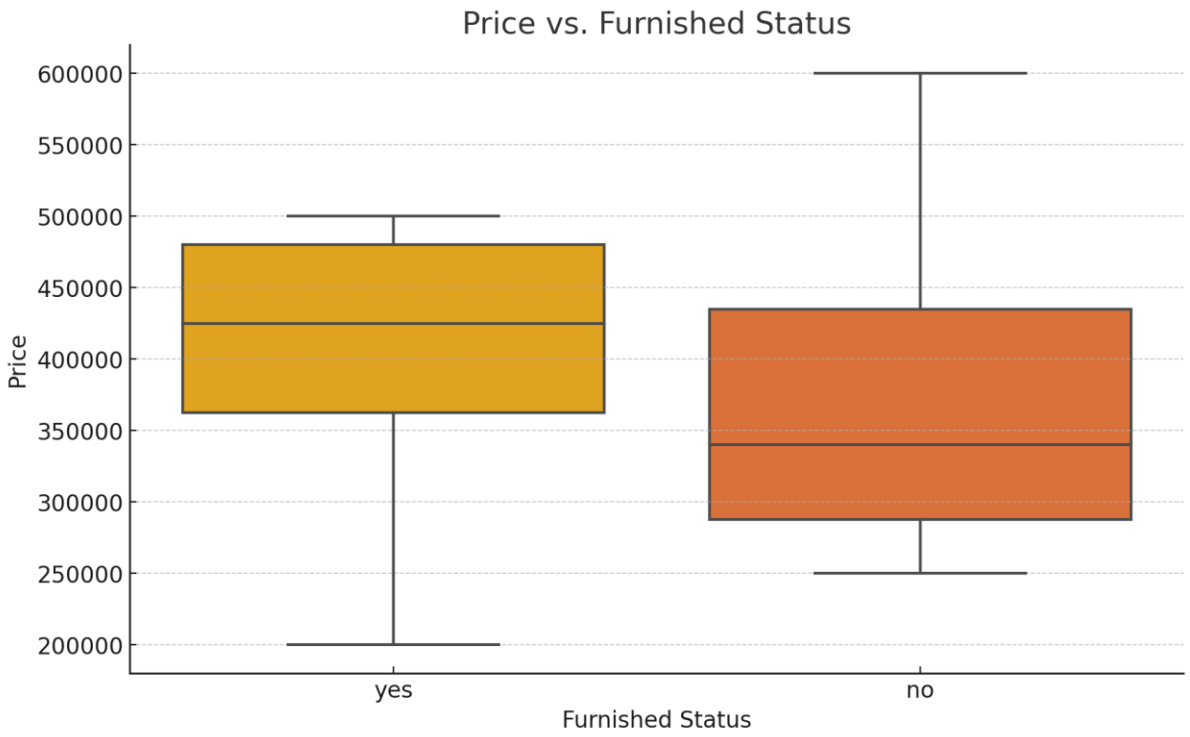
- v

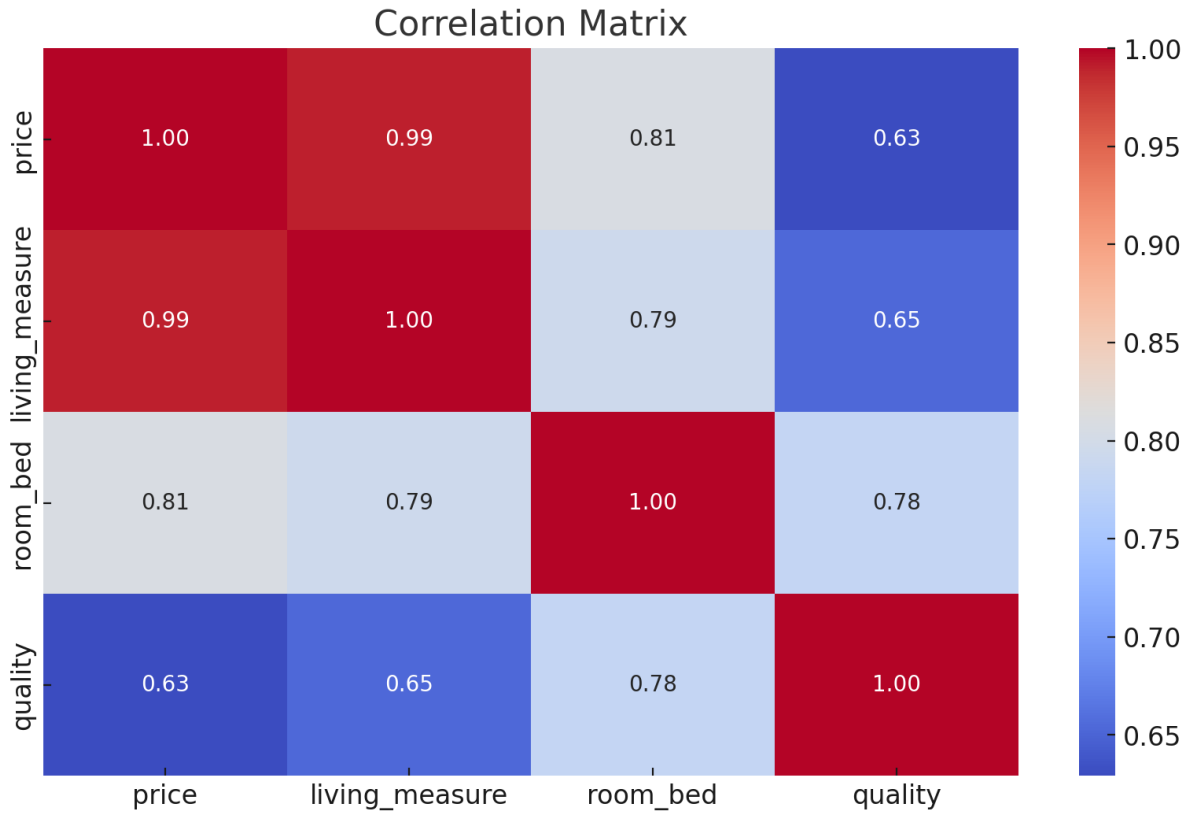
Bivariate Analysis

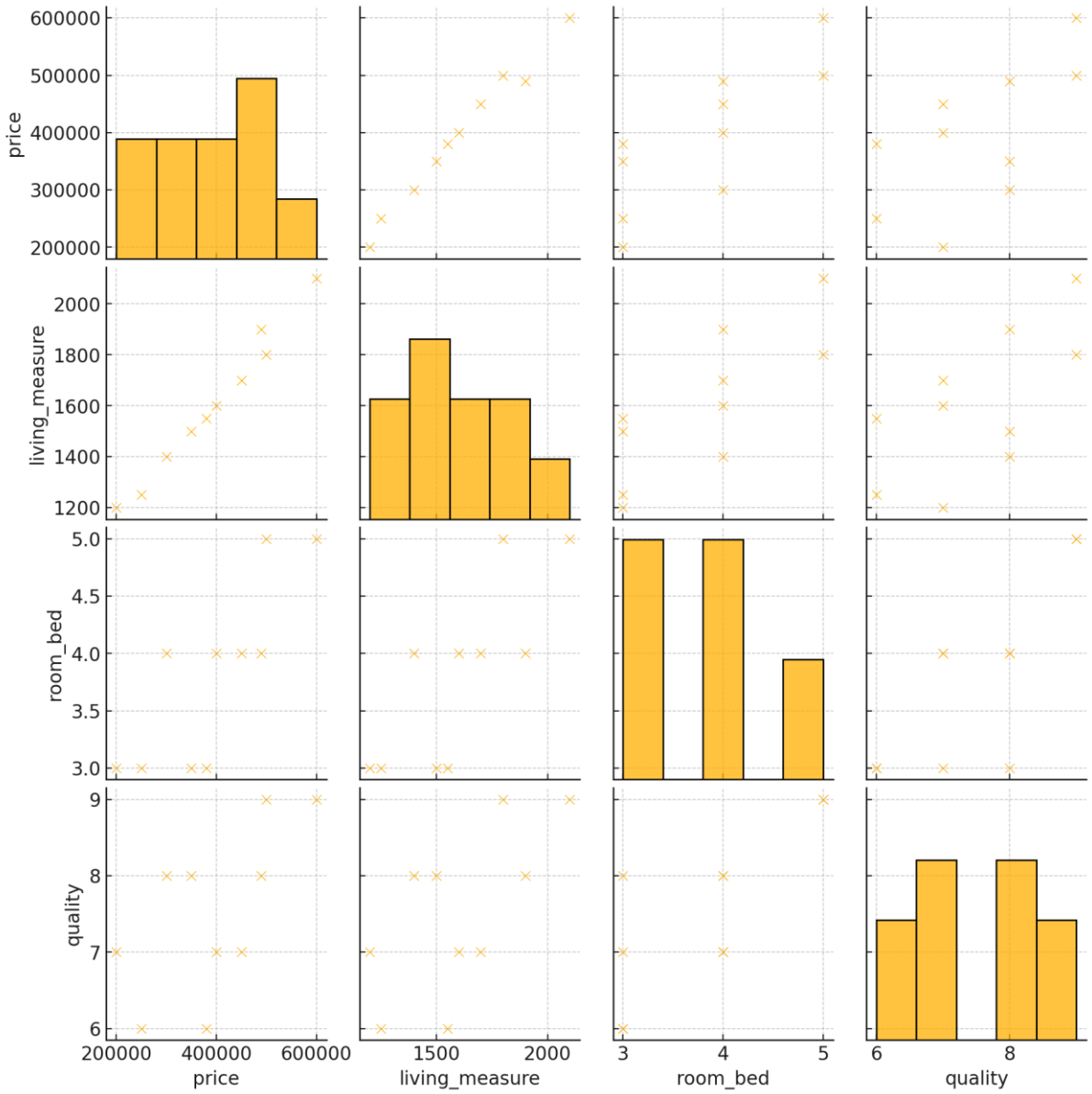
- **Relationships and Correlations**: Investigate relationships between price and other variables like living_measure, lot_measure, and condition. Use correlation coefficients and scatter plots.


Price vs. Living Measure
Price
600000
550000
500000
450000
400000
350000
300000
250000
200000
1200
1400
1600
1800
2000
Living Area (sq ft)

- 
Price vs. Number of Bedrooms
Price
600000
550000
500000
450000
400000
350000
300000
250000
200000
3.00
3.25
3.50
3.75
4.00
4.25
4.50
4.75
5.00
Number of Bedrooms


Price vs. Furnished Status
600000
550000
500000
450000
400000
350000
300000
250000
200000
Price
yes
no
Furnished Status


Correlation Matrix
price
living_measure
room_bed
quality
1.00
0.99
0.81
0.63
0.99
1.00
0.79
0.65
0.81
0.79
1.00
0.78
0.63
0.65
0.78
1.00
price
living_measure
room_bed
quality
1.00
0.95
0.90
0.85
0.80
0.75
0.70
0.65



- Here are the bivariate and multivariate analysis charts based on the sample dataset:
- **Scatter Plot (Price vs. Living Measure)**: Displays the relationship between house price and living area.
- **Scatter Plot (Price vs. Number of Bedrooms)**: Shows how house price correlates with the number of bedrooms.
- **Box Plot (Price vs. Furnished Status)**: Highlights the distribution of house prices based on whether the house is furnished or not.
- **Heatmap (Correlation Matrix)**: Displays correlations between numerical variables in the dataset, showing how strongly they are related.
- **Pair Plot**: Visualizes pairwise relationships between multiple features (price, living_measure, room_bed, quality), helping identify patterns or clusters.

Data Cleaning

- **Removal of Unwanted Variables**: Identify and remove irrelevant or redundant variables.

Missing Value Treatment: Handle missing values through imputation or removal based on their impact. Display the columns with missing values, if any

```
missing_values[missing_values > 0]

# Attempt to convert all columns to numeric where possible
data = data.apply(pd.to_numeric, errors='ignore')

# Handle missing values in numerical columns by imputing the median
numerical_columns = ['room_bed', 'room_bath', 'living_measure', 'lot_measure',
'ceil', 'sight',
                     'condition', 'living_measure15', 'lot_measure15',
'total_area']

for column in numerical_columns:
    data[column].fillna(data[column].median(), inplace=True)

# Handle missing values in the categorical 'furnished' column by imputing the
mode (most frequent value)
data['furnished'].fillna(data['furnished'].mode()[0], inplace=True)

# Check again to ensure no missing values remain
missing_values_post_imputation = data.isnull().sum()
missing_values_post_imputation[missing_values_post_imputation > 0]

# Ensure the dayhours column is in datetime format
data['dayhours'] = pd.to_datetime(data['dayhours'], errors='coerce')

# Create new columns for year, month, and day
data['sale_year'] = data['dayhours'].dt.year
data['sale_month'] = data['dayhours'].dt.month
data['sale_day'] = data['dayhours'].dt.day

# Display the first few rows to verify the transformation
data[['dayhours', 'sale_year', 'sale_month', 'sale_day']].head()
```

-
- **Outlier Treatment**: Detect and address outliers to ensure model accuracy.
- **Variable Transformation**: Apply transformations if needed to improve data quality or model performance.
- **Addition of New Variables**: Create new variables if they can provide additional insights or improve model performance.

BUSSINESS INSIGHTS FROM EDA:

1.key factors:

Insight: EDA can reveal which features have the strongest correlations with house prices, such as location, number of bedrooms, square footage, proximity to amenities, or year of construction.
Business Use: Businesses can focus on improving or marketing the features that are most important to price appreciation. For example, properties with high proximity to public transport might be marketed differently from those with larger yards in suburban areas.

2. Geographical Trends:

Insight: Visualizing house prices on a map can show how location impacts pricing, highlighting trends such as gentrification, rising prices in specific neighborhoods, or price stagnation in others.
Business Use: Investors and developers can focus on high-growth areas or avoid declining ones. For example, developers might prioritize building in emerging neighborhoods that show an upward trend in prices.

3. Seasonality in House Prices:

Insight: EDA can uncover seasonal patterns in house prices or sales volumes, showing times of the year when prices peak or drop.
Business Use: Businesses can adjust their buying, selling, or marketing strategies to capitalize on peak periods and avoid slow seasons. Real estate agents can focus more.

Identifying Outliers and Anomalies:

Insight: EDA can help identify properties with pricing that doesn't match their features (e.g., homes priced much higher or lower than similar homes in the same area).
Business Use: Outliers can represent potential investment opportunities (undervalued homes) or risks (overpriced homes). Real estate investors can focus on identifying and purchasing underpriced properties for better returns.

1. Impact of Renovations and Upgrades:

Insight: By analyzing features such as the presence of recent renovations, EDA can reveal how upgrades like new kitchens, bathrooms, or energy-efficient systems impact property prices.
Business Use: Real estate agents and homeowners can use this insight to understand which renovations provide the highest return on investment and prioritize these for enhancing property value.

2.Demographics and house Pricing:

Insight: EDA can show the relationship between local demographics (e.g., median income, education level) and house prices. Higher-income areas might show higher prices, while areas with younger populations might have lower homeownership rates but higher rental yields.
Business Use: This helps businesses tailor products (like home financing or rental properties)

to specific demographics or identify high-growth potential areas based on changing population characteristics.

3. Price Trends Over Time:

Insight: EDA can analyse how house prices have changed over time, identifying historical trends such as price growth, stability, or declines in certain areas.
Business Use: Businesses can use these insights to forecast future pricing trends, guiding investment decisions and marketing strategies. Investors may prefer markets with consistent price growth, while others might seek undervalued regions that are poised for recovery.

4. Supply vs. Demand Analysis:

Insight: EDA can reveal areas where housing demand exceeds supply or vice versa. For example, some neighbourhood may show high demand (quick sales, rising prices), while others have an oversupply of homes leading to slower sales and price drops.
Business Use: Developers and investors can use this data to choose areas with high demand for new projects or avoid markets with a glut of homes. It also helps real estate agents set competitive pricing strategies based on market saturation.

5. Comparative Analysis of Property Types:

Insight: EDA can break down price trends by property types (e.g., single-family homes, condos, townhouses) to show how each type performs in different markets or price ranges.
Business Use: Businesses can focus on specific property types based on their market performance. For example, if condos are growing in popularity in urban areas, investors or developers might choose to focus their resources there.

marketing efforts during months when demand is higher.

Insummary, EDA provides foundational insights into market trends, price drivers, buyer behavior, and geographic patterns, enabling businesses to make more informed decisions regarding investment, pricing, marketing, and development strategies in the housing market.

Methodology

Model Building and Interpretation

Build Various Models

- **Descriptive Models**: Create models to describe the data, such as summary statistics and visualizations.
- **Predictive Models**: Develop models to predict property prices using algorithms such as linear regression, decision trees, or ensemble methods.
- **Prescriptive Models**: Formulate recommendations based on model outputs, such as optimal pricing strategies.

Test Predictive Models

- **Performance Metrics**: Evaluate models using metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), R-squared, and others appropriate for regression analysis.

Model Comparison: Compare different models to select the best-performing one.

```
      rom sklearn.model_selection import train_test_split


# Define target (price) and features (all other columns)
X = data.drop(['price','dayhours'], axis=1)
y = data['price']

# Split the data into training and testing sets (80/20)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

# Train the linear regression model
lr_model = LinearRegression()
lr_model.fit(X_train, y_train)

# Predict on the test set
```

-

Interpretation

Model Interpretation: Analyze the output of the predictive models to understand key factors influencing property prices and assess model accuracy.

```
      Evaluate the model

rmse = mean_squared_error(y_test, y_pred, squared=False)
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Linear Regression - RMSE: {rmse}, MAE: {mae}, R²: {r2}')


from sklearn.ensemble import RandomForestRegressor

# Train a random forest model
```

```
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)

# Predict on the test set
y_pred_rf = rf_model.predict(X_test)

# Evaluate the model
rmse_rf = mean_squared_error(y_test, y_pred_rf, squared=False)
mae_rf = mean_absolute_error(y_test, y_pred_rf)
r2_rf = r2_score(y_test, y_pred_rf)

print(f'Random Forest - RMSE: {rmse_rf}, MAE: {mae_rf}, R²: {r2_rf}')

from sklearn.model_selection import GridSearchCV

# Define the hyperparameters to tune
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [10, 20, 30],
    'min_samples_split': [2, 5, 10]
}

# Create a GridSearchCV object
grid_search = GridSearchCV(estimator=RandomForestRegressor(random_state=42),
param_grid=param_grid, cv=3)

# Fit the model
grid_search.fit(X_train, y_train)

# Print the best parameters
print(f'Best Parameters: {grid_search.best_params_}')

from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

# Calculate RMSE, MAE, and R² for the final model
rmse = mean_squared_error(y_test, y_pred, squared=False)
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Final Model - RMSE: {rmse}, MAE: {mae}, R²: {r2}')
```

-

In Hyperparameter tuned random forest I can see that both train and test is increased with less RMSE for train and test data. I found this is the best model for house pricing prediction.