Advanced Analytics in Real Estate: Predicting Property Prices with Machine Learning

Group Members:

- Michelle LEE

- Gagan Singh

- Kaustubh Pandit

- JEN- HSIANG YANG

1.1 Background, Motivation, & Business Question

In the fast-paced real estate market, accurate price predictions are invaluable. This project is driven by the challenge faced by both individual and institutional investors in predicting property prices in a volatile market. The overarching business question is: How can we leverage machine learning to predict property prices based on various features, enhancing market understanding and investment decisions? This question is pivotal for clients aiming to maximize investment returns, optimize property listings, and navigate market fluctuations effectively.

1.2 Data and Statistical Question(s)

1.2.1 Data Description

Our dataset[1] presents a comprehensive snapshot of Mumbai's dynamic real estate market, featuring many essential attributes that characterize a diverse range of residential properties, including apartments and houses. It has undergone meticulous preprocessing to address missing values, data type inconsistencies, and outliers, ensuring a solid foundation for insightful analysis. Encompassing crucial details like possession status, electricity and water availability, city location, number of floors, balconies, property lifespan, covered and carpet areas, and prices, the dataset paints a vivid picture of the housing landscape in the bustling metropolis. These key attributes, ranging from the number of bedrooms and property type to locality and size, collectively unveil the narrative of Mumbai's vibrant and ever-evolving housing market.

[1] https://www.kaggle.com/datasets/shudhanshusingh/real-estate-properties-dataset

1.2.2 Statistical Question

The core statistical question we aim to address is: What are the key determinants of property prices, and how can we model these relationships to predict prices accurately? This question guides our exploratory data analysis, feature engineering, and modeling efforts.

Utility of the Statistical Question

- *Direct Business Impact:* By identifying and quantifying the relationship between property features and their prices, our analysis provides actionable insights for pricing strategies, market positioning, and investment evaluations.

- *Market Understanding:* This question facilitates a deeper understanding of market dynamics, helping stakeholders anticipate price movements and assess property value accurately.

Limitations

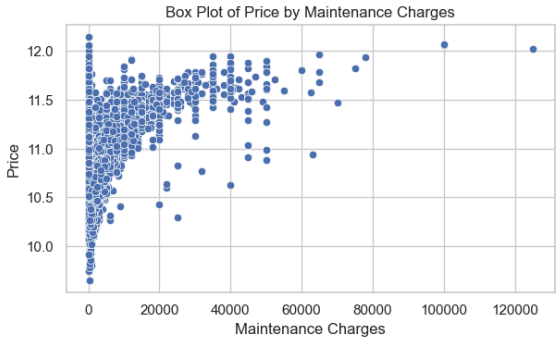
- *Market Variability:* The question may not fully encompass the nuanced effects of macroeconomic indicators, local market conditions, or non-quantifiable factors like property aesthetics.

- *Data Representativeness:* The predictive model's accuracy is contingent on the representativeness of the dataset. Limited or biased data could lead to skewed predictions, affecting decision-making.

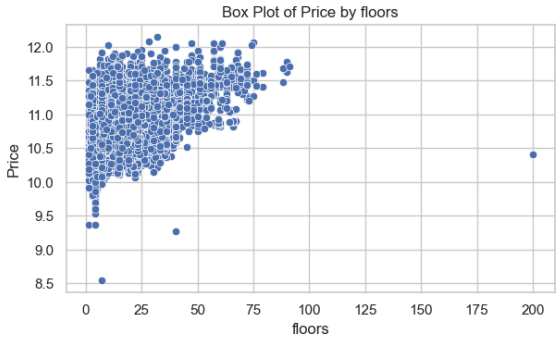
2. Exploratory Data Analysis

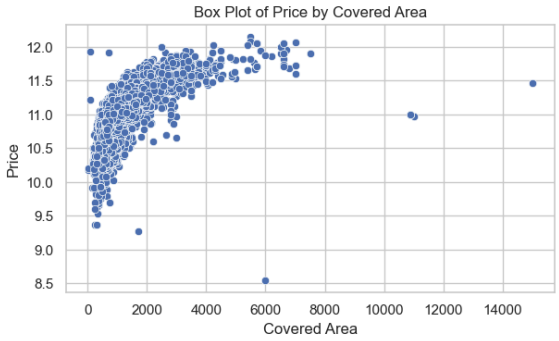
To explore the relationship between the target variable 'Price' and each independent variable, scatter plots are created for numerical variables and box plots are created for categorical variables.

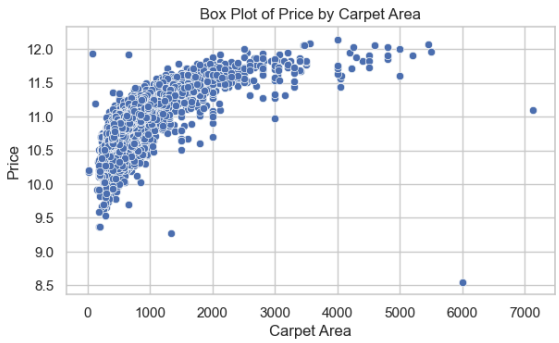
Numerical variables - Based on scatter plots, the variables 'Maintenance Charges', 'Bathroom', 'Floors', 'Covered Area', 'Carpet Area', and ‘bedroom’ have relationships with 'Price'.Note that the relationship between the transformed 'Price' and the variables 'Covered Area' and 'Carpet Area' is not linear, so a more complex model other than linear regression may be better suited for the dataset.

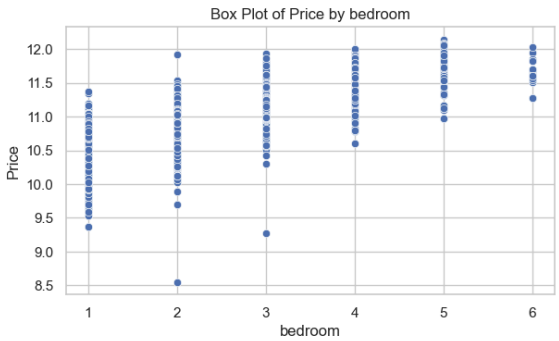

Box Plot of Price by Maintenance Charges
Price
Maintenance Charges
12.0
11.5
11.0
10.5
10.0
0
20000
40000
60000
80000
100000
120000


Box Plot of Price by Bathroom
Price
Bathroom
12.0
11.5
11.0
10.5
10.0
9.5
9.0
8.5
2
4
6
8
10


Box Plot of Price by floors
Price
floors
12.0
11.5
11.0
10.5
10.0
9.5
9.0
8.5
0
25
50
75
100
125
150
175
200

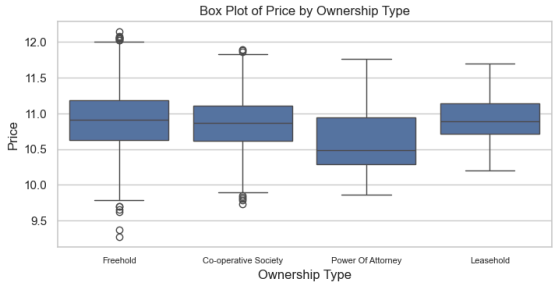

Box Plot of Price by Covered Area
Price
Covered Area
12.0
11.5
11.0
10.5
10.0
9.5
9.0
8.5
0
2000
4000
6000
8000
10000
12000
14000

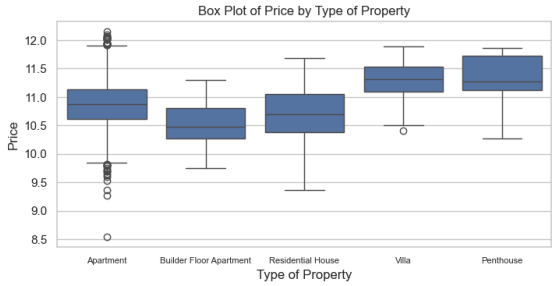

Box Plot of Price by Carpet Area
Price
Carpet Area
12.0
11.5
11.0
10.5
10.0
9.5
9.0
8.5
0
1000
2000
3000
4000
5000
6000
7000

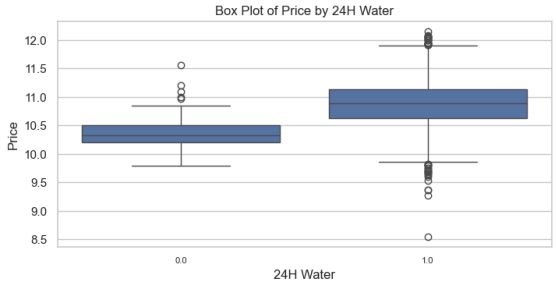

Box Plot of Price by bedroom
Price
bedroom
12.0
11.5
11.0
10.5
10.0
9.5
9.0
8.5
1
2
3
4
5
6

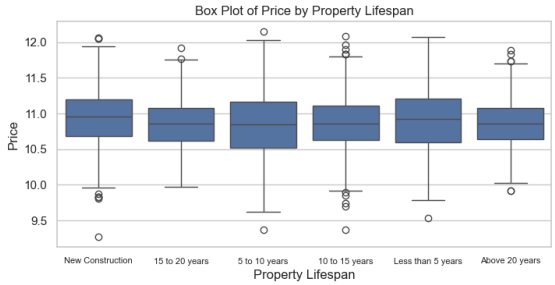
Categorical variables

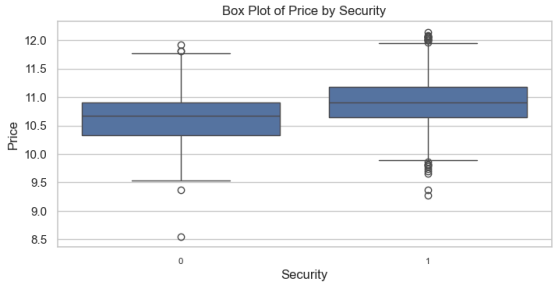
Based on box plots, the variables 'Property Lifespan', ‘'Type of Property'’, ‘Society, 24H Water’, 'Property Lifespan', and 'Piped Gas’, have relationships with 'Price'.

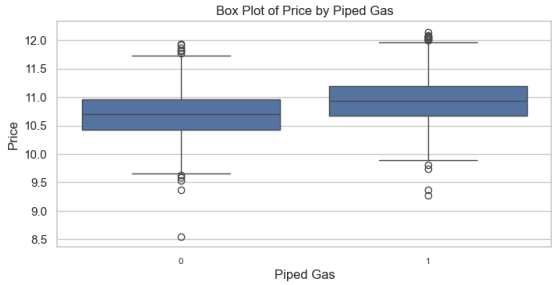












3. Method & Results

3.1 Feature Engineering and Data Preprocessing

To refine our dataset for modeling, we undertook several steps:

- *Initial Data Cleaning:* We removed irrelevant features such as IDs and landmarks that do not contribute to price prediction. This step simplifies the model and focuses on relevant predictors.

- *Handling Missing Values:* Missing values were addressed either by removal or imputation, depending on their significance and the volume of missing data. For

instance, properties with missing price information were excluded, while missing numerical features were imputed with median values.

- *Categorical Data Transformation:* We encode categorical variables (e.g., city, possession status) using one-hot encoding to convert them into a machine-readable format, facilitating their use in predictive modeling.

- *Numerical Feature Scaling:* Features like covered area and carpet area were normalized to ensure that all variables contribute equally to the model's predictive capability, preventing any single feature from disproportionately influencing the model due to its scale.

- *Feature Selection:* Based on exploratory data analysis, features with little to no variance or those highly correlated with others were identified for potential removal to improve model performance and interpretability.

Individual Data Transformations

For some column values based on some conditions, several different transformation functions and approaches were discussed across various contexts. Each function or method created for such transformations and details are as below:

1.Changing 'Possession Status' Based on Specific Values:

- Function/Approach: Used boolean indexing and the .apply() method with a lambda function.

- Description: This approach filters and modifies the 'Possession Status' column by changing specific string values ('Immediately', 'Ready to Move') to 'Ready to Move', and all other values to 'Under Construction', except for null values which are kept unchanged.

2. Modifying 'Floor Number' to Numeric Values Including Basements and Ground:

- Function/Approach: Utilized the .apply() method with a custom function.

- Description: Transformed the 'Floor Number' column, where specific string values ('Upper Basement', 'Lower Basement', 'Ground') are mapped to corresponding numeric values (-1, -2, 0 respectively), while preserving other numeric values and nulls as they are.

3. Updating 'Electricity Status' and Renaming to '24H Elec':

- Function/Approach: Applied .apply() method for value transformation and .rename() for column renaming.

- Description: The 'Electricity Status' column values were updated such that 'No/Rare Powercut' is mapped to 1, and all other values are mapped to 0, with null values remaining unchanged. Additionally, the column was renamed to '24H Elec' to reflect the new data semantics.

4. Filtering 'Maintenance Type' to Keep Only 'Monthly' and Nulls:

- Function/Approach: Boolean indexing for filtering.

- Description: Selected and retained only the rows where 'Maintenance Type' is either 'Monthly' or null. This was achieved by applying a condition directly on the DataFrame to filter rows accordingly.

5. Changing 'Property Lifespan' to Reflect Numeric Ranges:

- Function/Approach: Not explicitly done but discussed as a hypothetical case for category encoding.

- Description: Though a specific function wasn't created in the answers, the discussion involved encoding categorical data (like 'Property Lifespan' ranges) into numerical values or applying one-hot encoding for model training purposes. The approach would typically use pandas' .map() function or scikit-learn's encoders.

6. Transforming 'Tenants Preference' into 'Bachelors Allowed':

- Function/Approach: Used .apply() with a transformation function.

- Description: The transformation involved mapping 'Bachelors' to 1 and all other values to 0 in the 'Tenants Preference' column, which was then renamed to 'Bachelors Allowed'. This binary encoding simplifies the data for analyses focusing on availability for bachelors.

General Approach for Conditional Transformation:

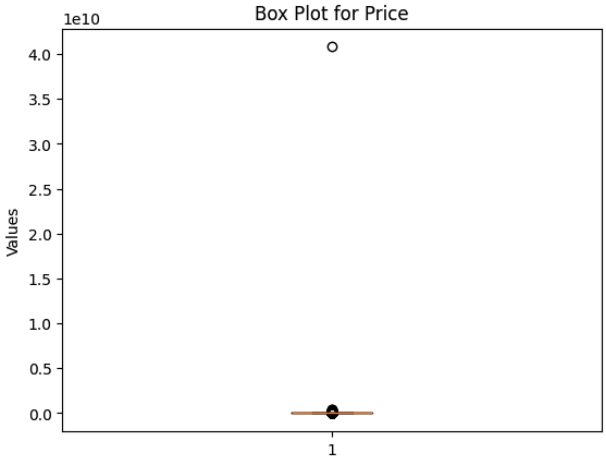
- The .apply() method was frequently used, combined with lambda functions or defined functions, to apply custom logic across a column. This method is versatile and can accommodate complex logic, making it suitable for a wide range of conditional data transformations.

- Boolean indexing was another common approach, particularly for filtering data based on specific conditions without explicitly iterating over rows.

Each transformation function or approach tailored the DataFrame to meet specific analytical needs or prepare the data for machine learning models, demonstrating the flexibility and power of pandas for data manipulation.

7. Removing Outliers

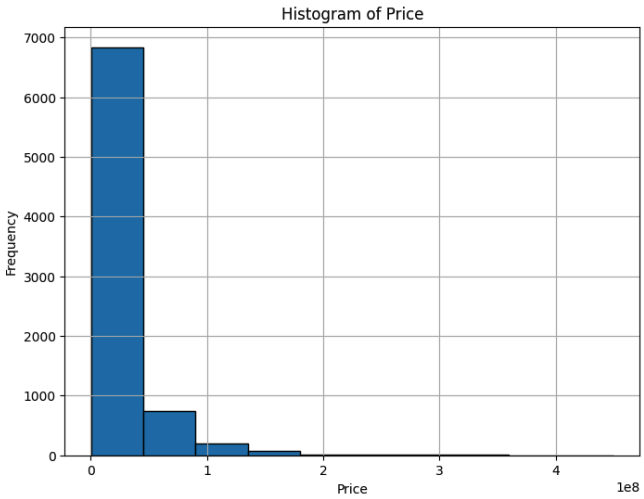
- Description: There is an outlier in the 'Price' column, so it is removed using the Pandas filter function.



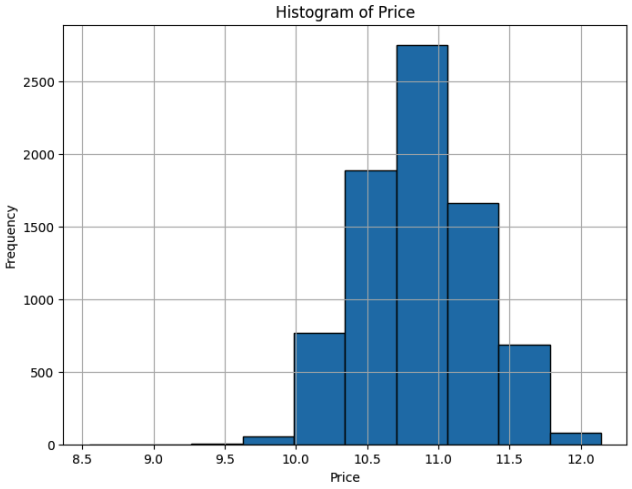
8. Transforming 'Price'

- function/Approach: Used stats.boxcox to transform.

- Description: From the barchart below, it is obvious that the price is not normal distributed data. it may cause a negative impact on regression models .



After Box-cox transformation, the transformed price data is normal distributed and can be used for regression model.



Data Pipeline

The data preprocessing step in a machine learning pipeline, involves setting up transformations for numeric features, categorical features, and binary categorical features. Let's break down each part:

The code categorizes the features (or columns) of a dataset into three distinct types based on their nature and the preprocessing steps applied to each. Here's a detailed description of each type:

1. Numeric Features:

- Columns: 'Units Available', 'Floor No', 'Maintenance Charges', 'Bathroom', 'floors', 'balconies', 'Covered Area', 'Carpet Area', 'bedroom'.

- Nature: These features are quantitative and represent measurable quantities. They can be integers or floating-point numbers. Examples include counts (e.g., 'Units Available'), dimensions (e.g., 'Covered Area', 'Carpet Area'), and other numeric specifications (e.g., 'Maintenance Charges', 'Bathroom' count).

- Preprocessing:

  - Scaling: Numeric features are scaled using `StandardScaler` to have a mean of 0 and a standard deviation of 1. Scaling is crucial for models that are sensitive to the magnitude of inputs, like SVMs, k-nearest neighbors, and gradient descent-based algorithms.

  - Imputation: Missing values in numeric features are imputed using the "most frequent" strategy, which replaces missing values with the most frequently occurring value in each column. This choice is somewhat unconventional for

numeric data, where the median or mean might typically be used, suggesting a specific rationale based on the data distribution or to maintain integer values.

2. Categorical Features:

- Columns: 'Possession Status', 'Ownership Type', 'furnished Type', 'Facing', 'Property Lifespan', 'Transaction Type', 'Type of Property'.

- Nature: These features are qualitative and describe categories or types. They are typically non-numeric and represent properties or attributes with two or more categories (e.g., 'Ownership Type', 'Facing').

- Preprocessing:

  - Imputation: Missing values are imputed using the "most frequent" strategy, filling in missing data with the most common category within each feature.

  - Encoding: Categorical features are encoded using `OneHotEncoder`, which converts categories into a binary matrix. This is necessary for models that require numerical input. The `handle_unknown='ignore'` parameter ensures that the model can handle new, unseen categories by ignoring them.

3. Binary Categorical Features:

- Columns: Include features like 'Commercial', 'Society', '24H Water', and many others related to amenities and property characteristics.

- Nature: These are special cases of categorical features with only two categories (binary). They indicate the presence or absence of a property or characteristic (e.g., 'Swimming Pool', 'Gymnasium').

- Preprocessing:

  - Encoding: Binary categorical features are also encoded using `OneHotEncoder` but with the `drop='if_binary'` option, which drops one of the binary columns to avoid redundancy and multicollinearity. This results in a single binary column for each feature, where 1 and 0 represent the presence or absence of the characteristic.

The distinction between general categorical features and binary categorical features in preprocessing highlights an effort to optimize the dataset for machine learning models, ensuring that each type of feature is treated in a manner that preserves its informational content while making it suitable for numerical algorithms.

3.2 Feature Selection

Given that there are more than 100 variables in the dataset, the RFE function is used for feature selection to reduce dimensionality and improve interpretability. Using

RFE with a maximum of 15 features, features ['numeric__Units Available', 'numeric__Floor No','numeric__Maintenance Charges', 'numeric__Bathroom', 'numeric__floors', 'numeric__balconies', 'numeric__Covered Area', 'numeric__Carpet Area', 'numeric__bedroom', 'categorical__furnished Type_Furnished', 'categorical__Property Lifespan_5 to 10 years', 'categorical__Property Lifespan_Above 20 years', 'categorical__Society_Y', 'categorical__24H Water_1.0', 'categorical__isPrimeLocationProperty_Y'] are selected as they are the most important features.

3.3 Reflection on selected performance metrics

R-squared is selected to be used as the metric to evaluate the model performance. This is because the primary objective of the model is to understand what features affect the variation in housing prices, and to what extent. Since R-squared measures how well can the selected features of the model explain the variations in housing prices, it is our performance metrics. This can help investors better understand what factors of a housing unit they should look for when making investment decision, and for developers to understand their market position. One potential ethical issues is that this information might be exploited and casue certain type of housing to be overpriced, leading to unaffordable housing to the public. This is an issue that we should keep a closer eye on to prevent our prediction to be utilized in a way that will create burdent to the general public.

Note that MSE, RMSE and MAPE information is also retrieved to get additional information and a more comprehensive view of the model's performance, yet we are still using R-squared to measure the model's performance. The result is that out of all the models, Random Forrest Regressor has the best R-squared score. Hence this model is selected for further hyperparameter tuning.

3.4 Hyperparameter tuning

A randomized hyperparameter search is used to enhance the model performance. The tuning is performed on the random forest regressor since we want to keep the number of features to be selected by the RFE at 15 to maintain interpretability of the model. Specifically, two hyperparameters are being tuned - the number of estimators (number of trees in the forest) and the maximum depth of the tree. We tune the first hyperparameter to see if increasing/decreasing the number of trees can help improve the predicting performance, and the latter to reduce the chances of overfitting as shown by the cross validation stage, there is quite a large gap between the train and test score.

The search result presents that the best hyperparameters for the regressor are having 115 trees in the forest, with a maximum depth of 12. This gives a training score of 0.818.

3.5 Model selection

The final model selected is Random Forest Regressor with the number of estimators of 115 and a max_depth of 12. Scoring it with the test set, it gives a score of 0.811, which is similar to the training score. It means that there is no overfitting nor underfitting problem with this model. The score used here is R-squared, aligning with the performance metrics that we chose. The model is also interpretable as we can find out how important each feature is by using .feature_importance_. This can help the stakeholders understand to what extent does a feature such as number of units available impact the listing price.

4. Summary

Leveraging the available information, we develop a predictive model to allow users make better informed decision in housing investments, and understand the real estate market better. Utilizing a Random Forest Regressor, this model predicts property prices with a good degree of accuracy. This is done so by predicting the prices based on 15 most important features that influence property prices the most. Namely, the number of bathrooms, bedrooms and balconies in a unit; the floor level of the unit and the number floors there are in the property; amount of maintenance charges; areas surrounded by the building; area in the building that is carpeted.; units available in the building; the property is society's; the unit is furnished; water is available 24/7; the property is at a prime location; and the property's lifespan. Using these factors, the model is able to explain approximately 81% of the variability in property prices.

Knowing the main influencing factors and the predictive power of the model means that investors can now make strategic investment decisions with higher confidence. It can also help them identify undervalued properties and predict the value of their properties investments. For developers, it means that they have a better understanding in the market position and appeal of their properties. They can utilize this information to enhancing their marketing and pricing strategies. Furthermore, the interpretability of this model indicates that it can be adapted continuously for the stakeholders to keep up with the market trend.