REAL ESTATE MARKET ANALYSIS AND PREDICTION USING MACHINE LEARNING

Supervisor: Mohamad Fneich

Done by Mohamad Naji

2023-2024

A Project Report on

REAL ESTATE MARKET ANALYSIS AND PREDICTION USING MACHINE LEARNING

Submitted in final project for the degree of

Masters

in

GIS and Data Science

Under the guidance of

DR. Ahmad Faour

Lebanese University

Department of Applied Mathematic

FACULTY OF SCIENCE

Table of Contents

Abstract

The real estate market, characterized by its complexity and dynamic nature, presents significant challenges for accurate property valuation and market analysis. This project leverages machine learning techniques to develop a robust system for predicting real estate prices in Lebanon. By integrating diverse datasets from web scraping, spatial analysis, and demographic sources, the study identifies key factors influencing property prices, such as location, size, and amenities. Advanced data preprocessing methods and machine learning models, including Random Forest, Gradient Boosting, and Stacking Regressor, were employed to achieve reliable predictions, with the Stacking Regressor yielding the highest accuracy. The project also introduces an interactive web platform that allows users to input property details and obtain price forecasts, facilitating data-driven decision-making for buyers, sellers, and investors. This report outlines the project's methodologies, challenges, and findings while proposing future enhancements, such as personalized recommendations and advanced analytics, to further improve usability and accuracy.

Introduction

Project Overview:

The rapid growth of the real estate sector has intensified competition, making it essential for stakeholders to gain a deeper understanding of market dynamics. Achieving success in this domain requires data-driven insights into property values, and location-specific attributes. This project aims to create a powerful analysis and prediction system that leverages machine learning to analyze and forecast real estate prices. By integrating vast datasets covering property characteristics, demographic information, spatial attributes, and by using machine learning the system provides actionable insights for buyers, sellers, and investors.

The core objective of this research is to develop a tool that automates the creation of a real estate dataset, an application capable of accurate property price predictions and offering a user-friendly interface for real estate exploration. By employing advanced machine learning algorithms, the project identifies key factors influencing property prices, helping stakeholders make informed decisions. This research also emphasizes data preprocessing, exploratory analysis, and model evaluation to ensure high predictive accuracy and usability. The results are presented through an interactive web platform, designed to support strategic decision-making and improve user engagement.

Problem statement

The abundance of available property data and intricate market dynamics make it difficult for investors, buyers, and sellers to make well-informed judgments in the real estate market. Without sophisticated tools and insights, forecasting property prices, and comprehending market circumstances can be challenging. This project addresses these gaps by building a tool to create a dataset for real estate in Lebanon and a machine learning-powered system that simplifies property evaluation and enhances decision-making processes.

Objective

This project is designed to achieve the following goals:

1. **Build an automation process tool to build real estate datasets:** create a tool to scrape real estate data from various sources and merge them together in a standard structure.
2. **Develop a Reliable Prediction System**: Build machine learning models to accurately predict real estate prices based on various property attributes and market conditions.
3. **Provide Actionable Insights**: Analyze key factors influencing property prices and present insights to users through a user-friendly platform.
4. **Enhance Stakeholder Decision-Making**: Support buyers, sellers, and investors by offering reliable data-driven tools for real estate market analysis.
5. **Introduce a Scalable Platform**: Create a system that can be expanded to include additional regions, features, and functionalities in the future.

Significance of Study

This project addresses critical challenges in the real estate sector by automating dataset creation and leveraging machine learning for accurate price predictions. It empowers stakeholders with data-driven insights, enabling informed decision-making, improved market understanding, and enhanced operational efficiency. By providing a scalable and adaptable platform, the system supports future expansion and offers businesses a competitive edge in navigating market trends and property valuations effectively.

Literature review

One significant challenge faced in this study was the absence of publicly available, comprehensive datasets on real estate prices in Lebanon. Despite extensive efforts to locate reliable data from various public and private sources, most datasets were incomplete, outdated, or lacked key features that are crucial for machine learning models. As a result, we had to collect and compile a dataset from scratch, incorporating data from multiple sources to ensure its relevance and accuracy. This gap in readily available datasets has also been highlighted in prior research, which often relies on datasets from more developed markets (e.g., U.S. or Europe) where such information is more accessible (Smith et al., 2018).

Furthermore, there is a notable lack of tools or models specifically designed for real estate price prediction in Lebanon. While there are various machine learning-based models available for real estate price prediction in countries like the U.S., Canada, and the U.K. (Cheng et al., 2020), no such models have been widely adopted or tailored to the Lebanese context. This underscores the necessity of developing localized models that account for Lebanon's unique socio-economic and political factors.

In contrast, numerous studies and commercial tools have been developed in other regions that successfully utilize machine learning techniques to predict real estate prices (Kim et al., 2019; Zhang et al., 2021). These models typically rely on rich datasets and well-established market dynamics, which are often absent or different in emerging markets like Lebanon. This research aims to fill the void by proposing a tailored machine learning approach that incorporates the nuances of the Lebanese real estate market.

Methodology

Possible Data Sources

To build a comprehensive real estate dataset for this project, data was gathered from multiple sources:

1. **Web Scraping**: Property details were extracted from websites such as Dubizzle, PBM, OSE Properties, and Real Estate Lebanon.

2. **Surveys**: A structured survey collected insights from property owners, real estate agents, and buyers about property attributes and market trends.

3. **Official Sources**: Municipalities and government agencies were contacted for data on property registration, land use, but because of the current situation and the lack of data we didn’t find any data.

4. **Demographic and Spatial Data**: External providers like ESRI Lebanon and using Geoapify supplied demographic, population density, and locational data.

These diverse sources ensured a robust dataset that integrates property details, and spatial attributes for accurate price predictions.

Data Collection

To create a robust real estate dataset covering Lebanon, data was scraped from multiple websites due to the lack of an existing dataset. Key sources included **Dubizzle** (https://www.dubizzle.com.lb/), **PBM** (https://pbm-leb.com/en/), **Real Estate** (https://www.realestate.com.lb/), and **OSE Properties** (https://ose-properties.com/). We enriched this dataset with demographic and spatial layers obtained from **ESRI Lebanon** and conducted spatial analyses in **ArcGIS**. An automated pipeline was implemented to standardize and merge the data, utilizing **Geoapify** for geolocation tasks. The resulting dataset is now prepared for cleaning and preprocessing to support predictive modeling efforts.

Automated process

An automated data scraping process was developed using Python to gather real estate data from multiple sources. The process was deployed on a Contabo server and configured to run monthly at a specified date and time. The steps followed were:

1. **Website Scraping**: For each real estate website, the URLs of listings were scraped by iterating through all the pages on the homepage. The collected URLs were saved in an Excel file on the server.

2. **Real Estate Details Scraping**: We then iterated through the previously scraped URLs to extract detailed information for each property, which was saved in an Excel file on the server.

3. **Data Merging**: Once data from all resources is collected, the information is merged into a single Excel file for easier analysis.

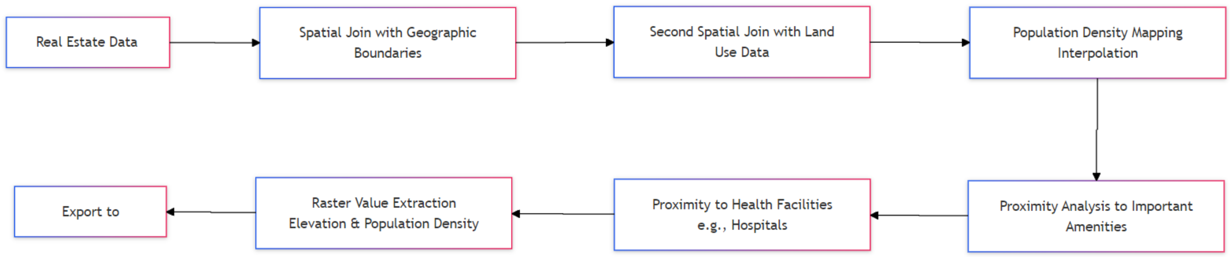
4. **Service Categorization**: As the data came from multiple sources, it was essential to standardize and categorize the services into consistent groups for uniformity.

5. **Geolocation Processing**: Finally, we reviewed the data to ensure each entry had valid coordinates. For those missing coordinates, we utilized the **Geoapify API** to obtain geolocation data based on the property's location name, ensuring accurate mapping.

Spatial Analysis Steps

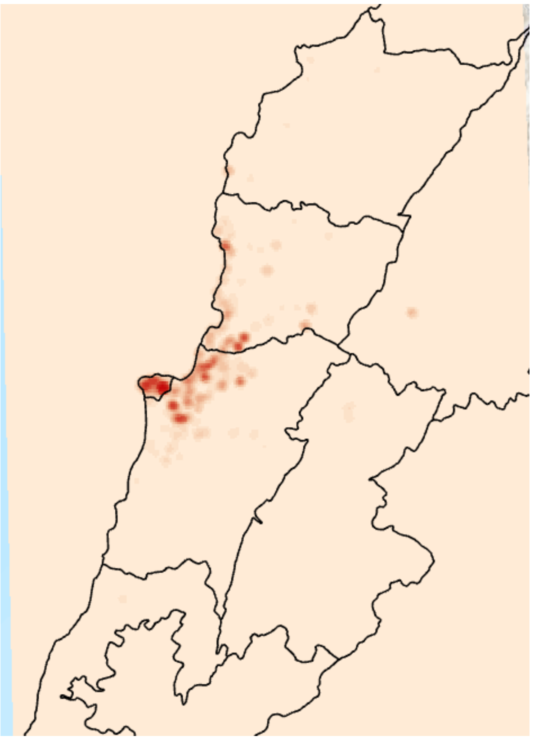
These steps involve joining multiple datasets, spatially analyzing property proximity to amenities, and incorporating raster data to enrich the real estate dataset. The result is a comprehensive dataset ready for further analysis, predictive modeling, or reporting.

1. **First Spatial Join**: The real estate data is joined with geographic boundary data to associate each property with its corresponding location. This operation combines relevant property details with geographic boundaries.
2. **Second Spatial Join**: The real estate data is then joined with land use data to add information about the land's characteristics, enriching the dataset with details about property location and land classification.
3. **Population Density Mapping**: A natural neighbor interpolation is used to generate a population density map based on existing point data, which is later applied to the real estate properties to add population-related information.
4. **Proximity Analysis**: The proximity of each property to important amenities (e.g., universities, hospitals, schools) is calculated. This step measures how far each property is from key facilities to provide insights on accessibility.
5. **Filtering and Proximity to Health Facilities**: Specific health facility types (e.g., hospitals, nurseries) are selected, and their distances from the properties are measured to assess how close the properties are to health services.
6. **Raster Value Extraction**: Raster values, such as elevation and population density, are extracted to the property locations, adding environmental and demographic data to the real estate dataset.
7. **Export to Excel**: After processing the spatial data, the final dataset is exported to an Excel file for further analysis or reporting.


Real Estate Data
Spatial Join with Geographic Boundaries
Second Spatial Join with Land Use Data
Population Density Mapping Interpolation
Proximity Analysis to Important Amenities
Proximity to Health Facilities e.g., Hospitals
Raster Value Extraction Elevation & Population Density
Export to

Study area

After mapping the real estate data, we observed that the data is concentrated in three main governorates. As a result, we decided to limit the scope of the study to these regions to ensure a more focused and relevant analysis. This targeted approach allows us to examine the factors influencing real estate trends in these specific areas, where the majority of the data points are located.



The governorates are:

- Beirut
- Mont-Lebanon
- Keserwan

It is important to note that, officially, Mont-Lebanon and Keserwan are part of the same governorate; however, they are often treated separately in some sources. Therefore, we chose to separate them for greater accuracy in our analysis.

Data preprocessing

Several data preparation stages were conducted after combining datasets from various categories

Initial Handling

- Categorize real estate services.
- Filled certain columns with default values to ensure dataset consistency.
- Removed rows with missing price values to maintain data completeness and reliability.
- Removed columns with many missing values.
- Detected and removed duplicate rows to ensure data integrity and eliminate bias in subsequent analysis.

Data Standardization

- Processed the “Price” column by removing currency symbols (e.g., "$") and converting the values to the float data type.
- Processed the “Size” column by removing size symbols (e.g., "m^2") and converting the values to the float data type.

Data augmentation

We have used data augmentation to add records to the data where price is higher than 600K$, at the end of the process we have added 1268 new records.

Label Encoding

- Converted categorical columns into numerical formats using label encoding.
- Columns encoded include Property Type, Ownership, Payment method, Condition, land use land cover (LULC), governorate.

Handling Missing Values

- Applied an “Iterative Imputer” to intelligently fill remaining missing values, preserving the dataset's structural integrity.

Column	**Non-Null Count**	**Data type**
Property Type	8197	int64
Ownership	8197	int64
Bedrooms	8197	int64
Bathrooms	8197	int64
Size (m^2)	8197	float64
Payment method	8197	int64
Condition	8197	int64
X	8197	float64
Y	8197	float64
Near Amenities	8197	float64
Heating and Cooling	8197	float64
Outdoor and Landscaping	8197	float64
Security Features	8197	float64
Storage and Space	8197	float64
Views	8197	float64
Technology and Utilities	8197	float64
Luxury and Convenience	8197	float64
Fitness and Recreation	8197	float64
Pet-Friendly	8197	float64
Child-Friendly	8197	float64
LULC	8197	int64
governorate	8197	int64
NEAR_PRV_UNI_DIST	8197	float64
NEAR_PUB_UNI_DIST	8197	float64
NEAR_HOSPITAL_DIST	8197	float64
NEAR_NURSERY_DIST	8197	float64
NEAR_PHARMACY_DIST	8197	float64
NEAR_SCHOOL_DIST	8197	float64
DEM	8197	float64
Population	8197	float64

Price	8197	float64

Outlier Removal

Used the “Isolation Forest algorithm” to detect and remove outliers, ensuring the dataset contained only valid and representative data points.

After removing the outliers, the number of records has been decreased to 6967 records.

Feature Selection

- Calculated correlations between all features and the “Price” column.
- Selected the top 19 features for model training, including Size (m²), Bathrooms, governorate, Bedrooms, NEAR_PUB_UNI_DIST, NEAR_PRV_UNI_DIST, NEAR_HOSPITAL_DIST, NEAR_NURSERY_DIST, NEAR_PHARMACY_DIST, Population, LULC, Storage and Space, DEM, Luxury and Convenience, Security Features, Heating and Cooling, Fitness and Recreation, Technology and Utilities, NEAR_SCHOOL_DIST.

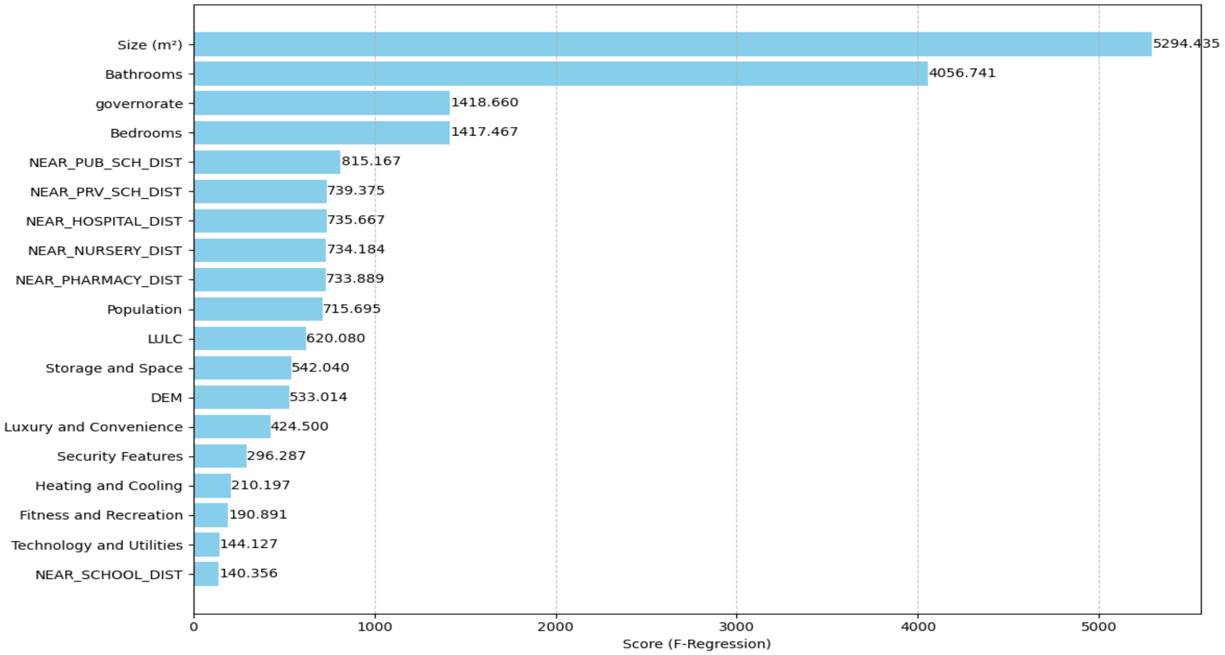


Figure 1 - Correlation between price and each selected field

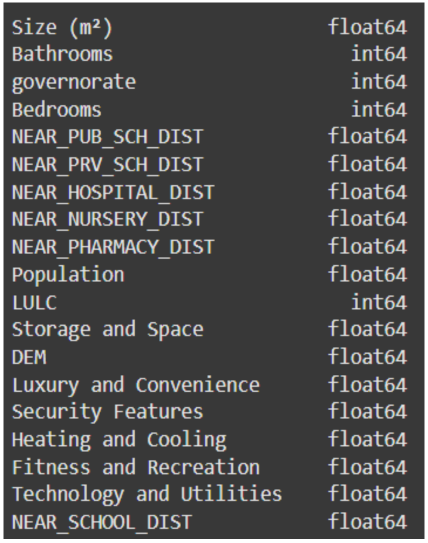
Final Dataset Shape

- After preprocessing, the dataset was reduced to a clean and optimized shape of (6988, 19), ready for machine learning applications.

Data analysis

Understanding the data, spotting any problems, and exploiting insights for project development are all important steps in the data science cycle. It is critical in resolving challenges and ensuring the smooth execution of projects. Python packages such as Matplotlib and Seaborn were used in this investigation, along with Power BI for better visualizations. Let's go into the details of our data analysis procedure:

Columns and their datatypes:



```
Size (m²)                      float64
Bathrooms                        int64
governorate                      int64
Bedrooms                         int64
NEAR_PUB_SCH_DIST              float64
NEAR_PRV_SCH_DIST              float64
NEAR_HOSPITAL_DIST             float64
NEAR_NURSERY_DIST              float64
NEAR_PHARMACY_DIST             float64
Population                     float64
LULC                             int64
Storage and Space              float64
DEM                            float64
Luxury and Convenience         float64
Security Features              float64
Heating and Cooling            float64
Fitness and Recreation         float64
Technology and Utilities       float64
NEAR_SCHOOL_DIST               float64
```

Number of missing values

Let us check the number of missing values in the data columns:

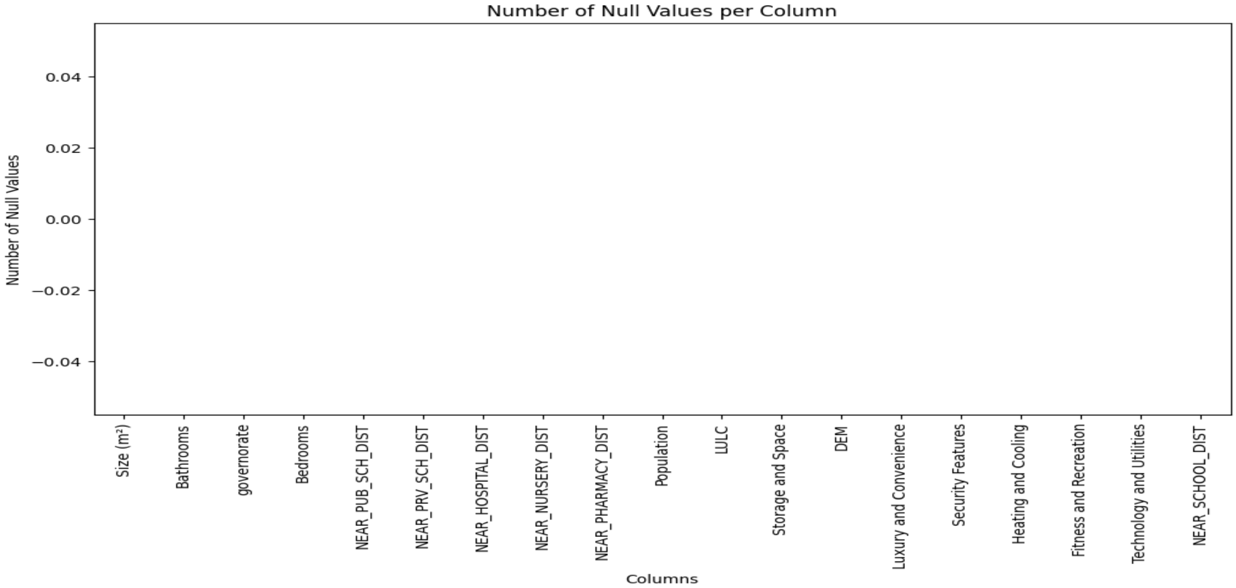
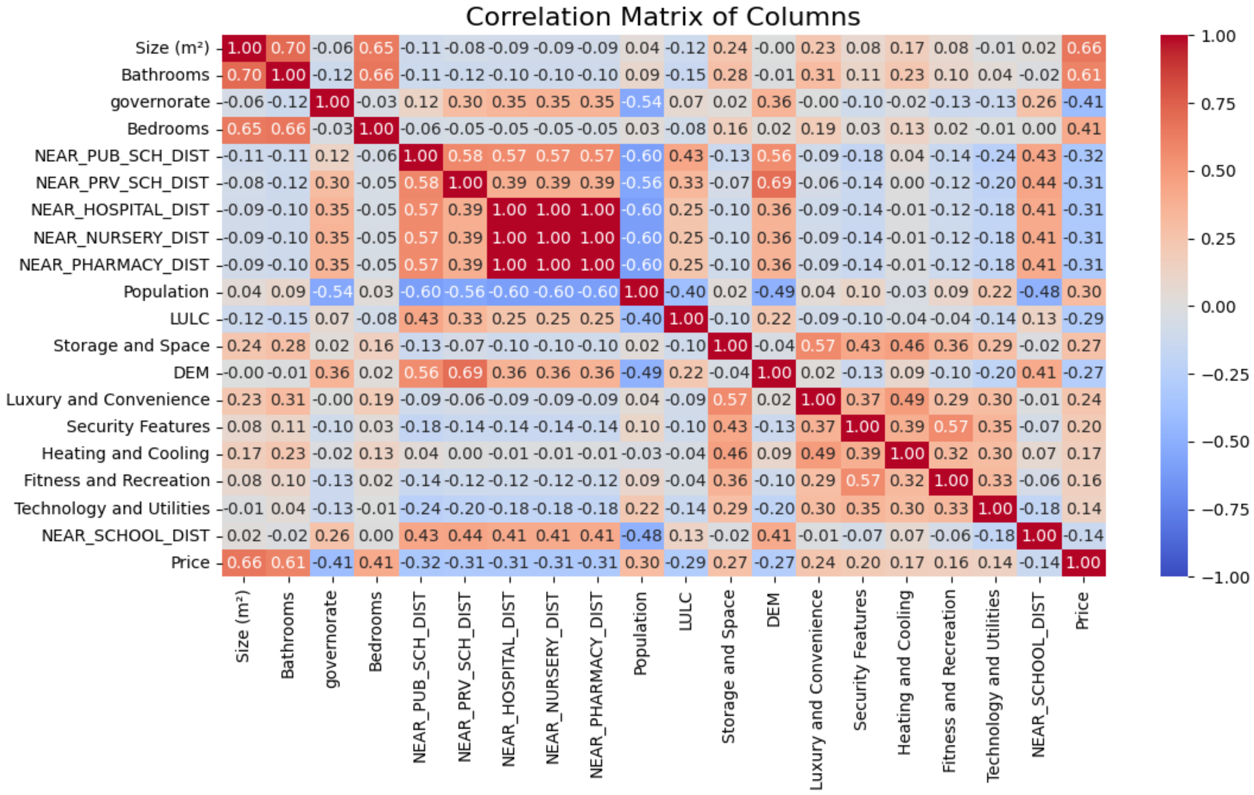
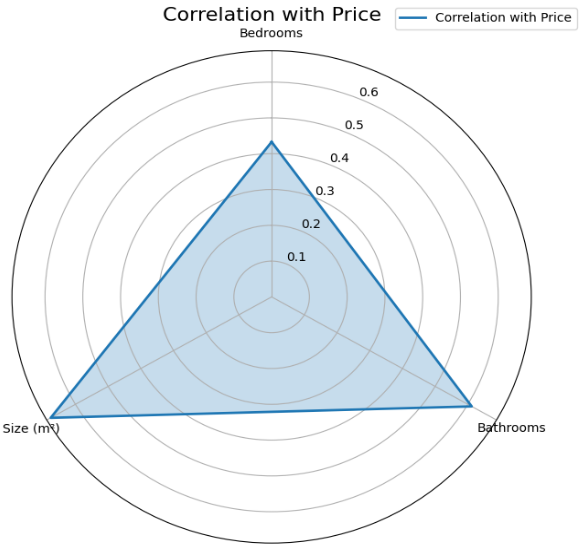


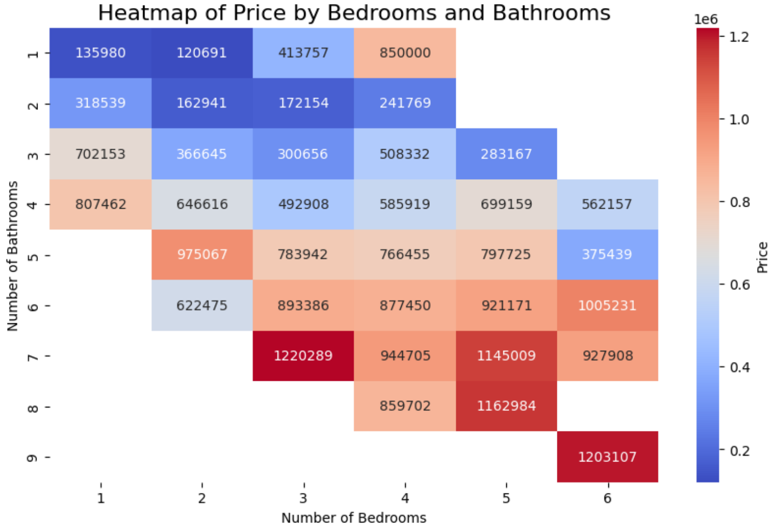
Figure 2 - This shows that removing missing values from the dataset was done correctly.

Correlation between columns



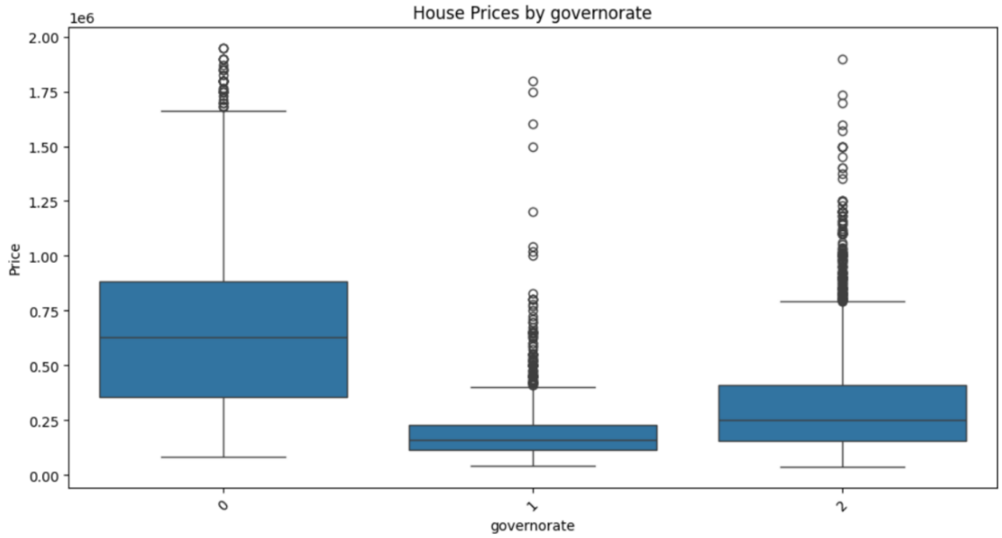
As we can see, the price is strongly correlated with the columns of bedrooms, bathrooms, and size. The graph below will demonstrate this finding.





The radar chart shows the relationship between the price of a home and its size, number of bedrooms, and number of bathrooms. The strongest association is seen in property size, highlighting how important it is in setting pricing. Additionally, bathrooms are important, indicating that purchasers like the ease and comfort they offer. Although they have a positive correlation, bedrooms have a comparatively smaller impact, suggesting that purchasers place more value on a property's total size and the availability of bathrooms than on the number of bedrooms. These results can help determine how much to charge for real estate costs and what aspects to highlight in advertising campaigns.

Governorate vs price



Index:

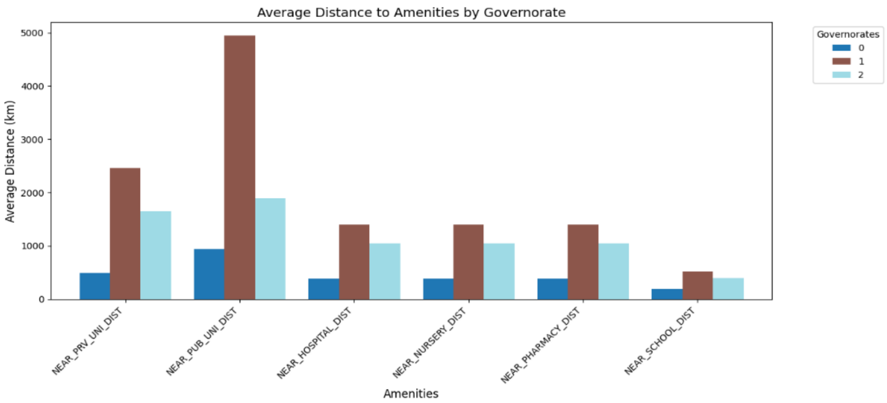
- 0: Beyrouth
- 1: Keserwan-Jbeil
- 2: Mont-Liban

House prices vary significantly across the governorates, with Governorate 0 being the most affluent, Governorate 2 moderately priced, and Governorate 1 being the least affluent but with potential high-value properties in certain areas. This distribution indicates differing economic statuses and potential investment opportunities in each governorate.

Average Distance to Amenities by governorates:

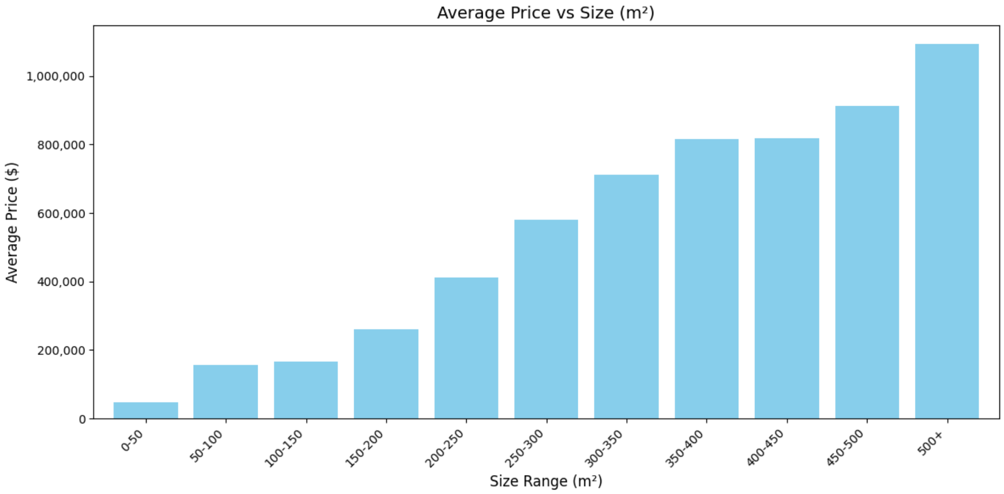
It refers to the average of the distances to amenities for a group of properties, typically calculated at the level of a larger geographic unit, governorate in our case.

This is the means of individual distances to amenities for all properties within a specific area.



Governorate 0:
has the closest access to schools, pharmacies, nurseries, hospitals, and universities. This implies that it has easier access to facilities, which can be indicative of a more developed or urbanized area.
Governorate 1:
demonstrates the greatest distances to all facilities, particularly public hospitals and universities, suggesting restricted access to necessary services. This could indicate a less developed or more rural area.
Governorate 2:
is better than Governorate 1 but still more distant from facilities than Governorate 0. This may benefit from additional improvement and indicates a modest level of service accessibility.
In summary, Governorate 0 offers the greatest access to facilities, but Governorate 1 faces major accessibility issues. While Governorate 2's access to amenities is greater than Governorate 1's, it is still not ideal.

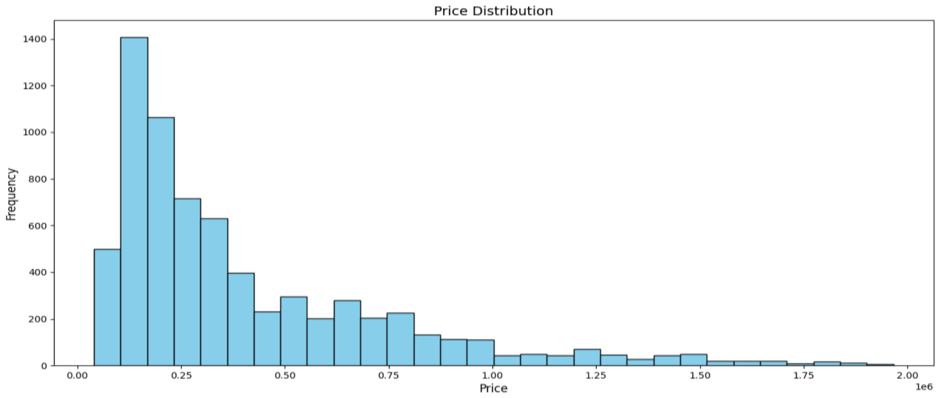
Price vs Size



The average price of real estate over a range of sizes shows a distinct upward trend, with prices rising as property size increases. With an average price of 48,500$, smaller houses in the 0–50 m² segment are the most reasonably priced. Mid-sized houses, including those in the 150–200 m² range, have gradually rising prices, with an average of $259,867$. Because of their superior value, larger properties—especially those larger than 500 m²—command far higher prices, with an average of $1,092,666$.

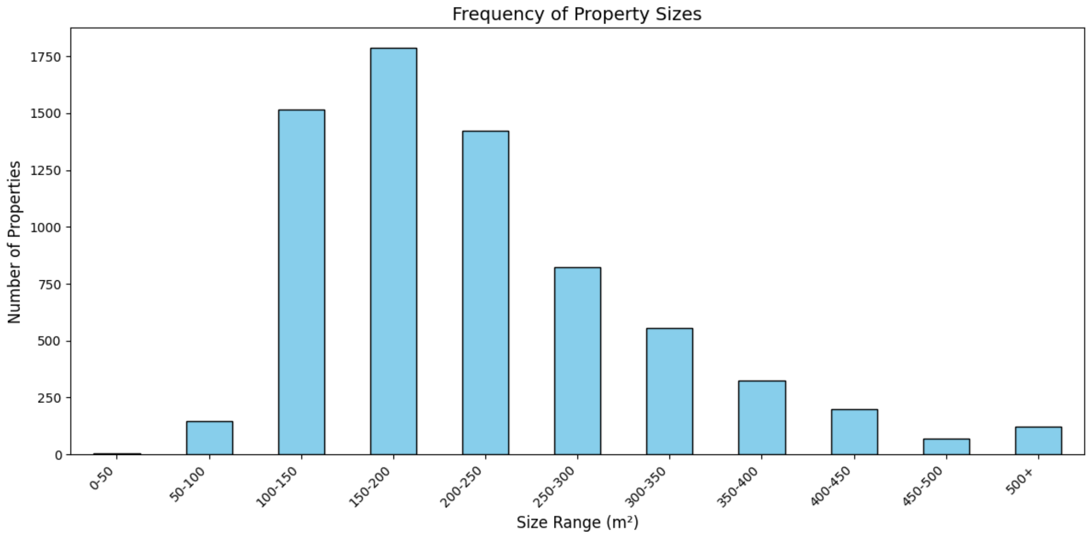
Interestingly, the price growth for the larger homes plateaus and then rises again between 400 and 500 m². This trend identifies different market niches, ranging from high-end luxury residences to more affordable smaller dwellings.

Price distribution:



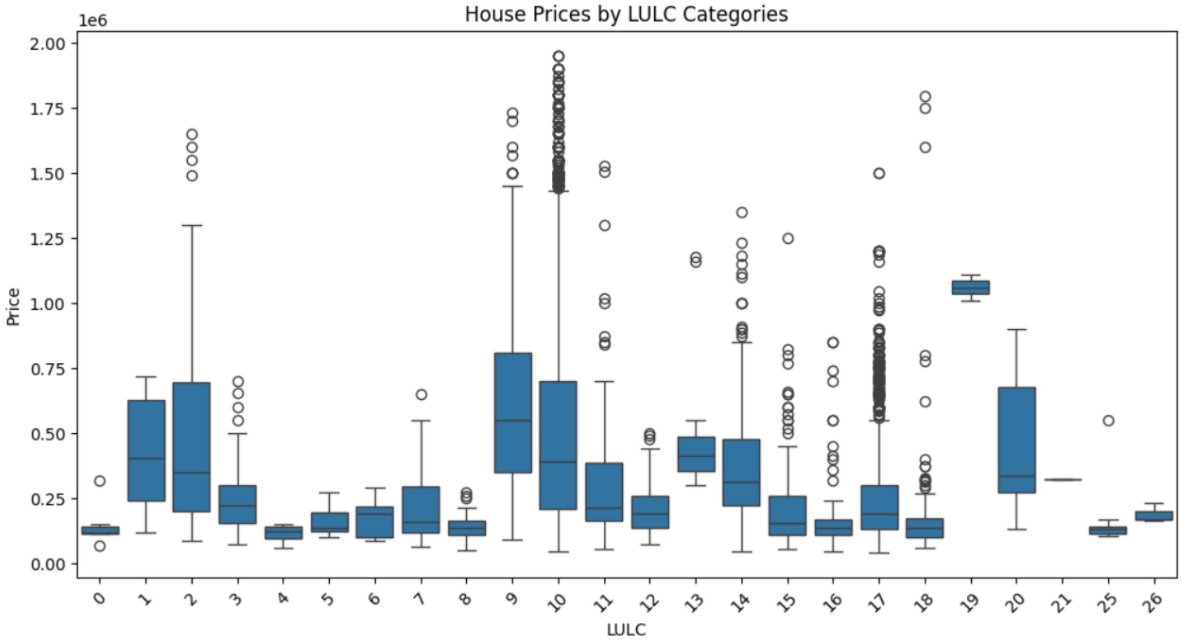
There is a lot of variety in the dataset's property prices. With a standard deviation of 344,892 and a mean price of 405,078, the sample's real estate prices appear to vary widely. There are properties with modest and high prices, as seen by the minimum price of 40,000 and the maximum price of 1,965,996. The bulk of properties are in the mid-range price bracket, as evidenced by the first quartile (25%) being 160,000, the median (50%) being 275,000, and the third quartile (75%) being 550,000. These findings demonstrate the wide range of property prices, including some extremely high outliers.

Frequency of Property Sizes



The distribution of property sizes reveals a concentration of properties in the mid-sized ranges. The most common size ranges are 150-200 m² (1,786 properties), 100-150 m² (1,517 properties), and 200-250 m² (1,423 properties), representing a majority of the dataset. In contrast, there are fewer properties in the smaller (0-50 m²) and larger (500+ m²) size ranges, with only 4 and 122 properties respectively. This indicates that mid-sized properties are more prevalent in the market, while both very small and very large properties are relatively rare.

LULC categories vs Price

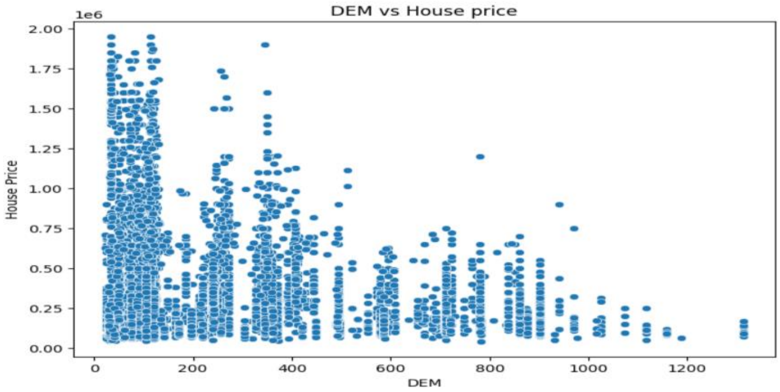


Category-Specific Variation: Certain LULC categories, such as Dense Urban Fabric (10) and Medium Density Urban Fabric (17), show a broad range of house prices, indicating varied property values within these urban areas.

Low Prices in Natural Areas: Categories such as Bare Rocks (1), Banana (0), and Rocky Outcrops (23) generally have lower house prices, likely due to their limited development potential or lower desirability for residential purposes.

High Prices in Specific Urban Areas: Some categories like Port Areas (20) and Dense Urban Fabric (10) show higher median prices, suggesting that proximity to urban centers and infrastructure increases property value.

DEM vs price



1. **Higher House Prices Cluster at Lower DEM Values**:
   - A significant number of houses with higher prices (e.g., around $1,000,000 or more) are concentrated where DEM values are relatively low (close to 0-200).
   - This suggests that areas with lower elevation tend to have more expensive houses.
2. **Lower House Prices Spread Across Higher DEM Values**:
   - As DEM increases (e.g., above 400), the house prices are generally lower and tend to scatter with less clustering.
3. **Inverse Relationship**:
   - There appears to be a general inverse correlation between DEM and house prices. In other words, as elevation increases, house prices tend to decrease.

Models and Techniques

Evaluation Metrics

Evaluation metrics play a crucial role in assessing the performance of machine learning models. They provide a quantitative measure of how well a model predicts outcomes, enabling comparisons between different algorithms and fine-tuning for improved accuracy. In the context of real estate price prediction, these metrics are essential for ensuring that the models produce reliable and actionable insights.

Mean Square Error:

The mean squared error (MSE) or mean squared deviation (MSD) of an estimator (or a process for estimating an unobserved quantity) is a statistical metric that calculates the average squared difference between the estimated and actual values. The expected value of the squared error loss is represented by the risk function known as MSE.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$$

Where: MSE = mean squared error

n = number of data points

Y_i = observed values

$Y_i^{\wedge}$= predicted values

How it works:

1- Calculate the error for each data point by subtracting the predicted value from the actual value.
2- Square each error to ensure positivity and emphasize larger deviations.
3- Sum all squared errors and divide by the total number of data points to find the average, yielding the MSE.

R Squared:

R-squared (R2) is defined as a number that tells you how well the independent variable(s) in a statistical model explains the variation in the dependent variable. It ranges from 0 to 1, where 1 indicates a perfect fit of the model to the data.

$$R^2 = 1 - \frac{RSS}{TSS}$$

Where: R2 = coefficient of determination

RSS = sum of squares of residuals

TSS = total sum of squares

How it works:

1- **Calculate the Total Variance**: Find the total variance of the dependent variable (how much it varies from the mean).
2- **Measure Explained Variance**: Compute how much variance is explained by the independent variables (the model's predictions).

3- **Compute Residual Variance**: Calculate the variance of the residuals (differences between actual and predicted values).

4- **Ratio of Explained to Total Variance**: Divide the explained variance by the total variance to get the R-squared value.

5- **Range**: The value ranges from 0 (no explanatory power) to 1 (perfect fit).

Root Mean Squared Error (RMSE):

Is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit. Root mean square error is commonly used in climatology, forecasting, and regression analysis to verify experimental results.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Where: y_i= Actual value for the i^{th} observation.

$Y_i^{\wedge}$= Predicted value for i^{th} observation.

n = Total number of observations.

How it works:

1. **Calculate the Squared Error for Each Observation**: For each data point, find the difference between the actual value and the predicted value, then **square** this difference.
2. **Sum of Squared Errors**: Add up all the squared errors for each data point to calculate the **total squared error**.
3. **Calculate the Mean of Squared Errors**: Divide the total squared error by the number of observations to get the **mean squared error (MSE)**.
4. **Take the Square Root of MSE**: To return the error to the same scale as the original data, take the square root of the MSE.
5. **Interpret the RMSE Value**:
   - The resulting RMSE value represents the **average magnitude** of the errors in the model’s predictions, but it is more sensitive to large errors than MAE because of the squaring operation.
   - A lower RMSE means better model performance, as the predictions are closer to the actual values.
6. **Range**: RMSE ranges from 0 to infinity:
   - **0** means perfect predictions (no error).
   - **Higher values** indicate larger errors, meaning the model is less accurate.

Mean Absolute Error (MAE)

Mean Absolute Error (MAE) is a measure of the average size of the mistakes in a collection of predictions, without taking their direction into account. It is measured as the average absolute difference between the predicted values and the actual values and is used to assess the effectiveness of a regression model.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i|$$

Where: y_i = Actual value for the i^{th} observation.

$y_i^{\wedge}$ = Predicted value for the i^{th} observation.

n = Total number of observations.

How it works:

1. **Calculate the Absolute Error for Each Observation:** For each data point, find the difference between the actual value and the predicted value, then take the absolute value of this difference.

   Absolute Errori = |yi – yi^ |

2. **Sum of Absolute Errors:** Add up all the absolute errors for each data point to calculate the total absolute error.

   $$\text{Total Absolute Error} = \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

   Where n is the total number of observations in the dataset.

3. **Calculate the Mean of Absolute Errors:** Divide the total absolute error by the number of observations to get the average of these absolute errors.

   $$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i|$$

4. **Interpret the MAE Value:** The resulting MAE value represents the average magnitude of errors in the model's predictions. A lower MAE means better model performance, as the predictions are closer to the actual values.
5. **Range:** MAE ranges from 0 to infinity:
   - 0 means perfect predictions (no error).
   - Higher values indicate larger errors, meaning the model is less accurate.

Models

To choose the optimal model, we compared the outcomes of several models.

Model Selection

Several regression models were implemented to predict house prices, including:

- Random Forest
- Extra Trees
- Gradient Boosting
- AdaBoost
- Ridge Regression
- Lasso Regression
- ElasticNet
- K-Nearest Neighbors
- Decision Tree

These models were chosen to encompass a variety of tree-based, linear, and instance-based algorithms suitable for regression tasks.

Hyperparameter Optimization Using Optuna

To improve model performance, **Optuna**, a powerful hyperparameter optimization library, was utilized. For each model, an **objective function** was defined to minimize the Mean Squared Error (MSE) on the test set. The hyperparameters optimized for each model are as follows:

- **Random Forest**: Number of trees, maximum depth, minimum samples split, and minimum samples per leaf.
- **Extra Trees**: Similar parameters as Random Forest.
- **Gradient Boosting**: Number of trees, learning rate, maximum depth, and minimum samples split.
- **AdaBoost**: Number of estimators and learning rate.
- **Ridge, Lasso, and ElasticNet**: Regularization strength (alpha) and L1/L2 ratio (for ElasticNet).
- **K-Nearest Neighbors**: Number of neighbors, weight function, and distance metric.
- **Decision Tree**: Maximum depth, minimum samples split, and minimum samples per leaf.

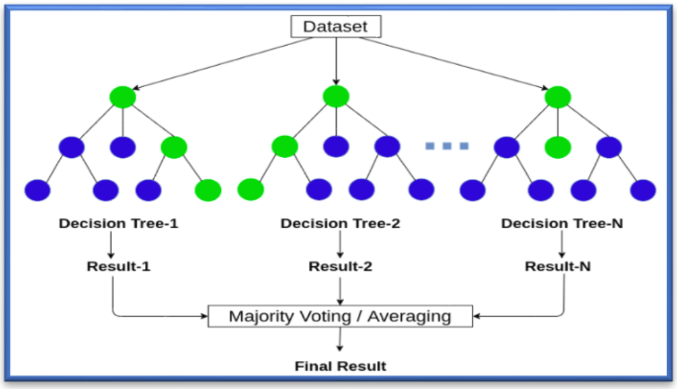
Each model was trained using the train_test_split method, with 70% of the data for training and 30% for testing. Optuna was configured to run 50 optimization trials per model, exploring the hyperparameter space effectively.

Model details

Breakdown of how each of these models works:

Random Forest:

It is an ensemble learning method that constructs multiple decision trees during training. Each tree is trained on a random subset of the data, and the final prediction is made by aggregating the predictions of all the trees (using majority voting for classification or averaging for regression). This approach reduces overfitting compared to a single decision tree and increases robustness, as each individual tree's prediction is less likely to be overly influenced by outliers or noise.



How does it work?

1. **Bootstrap Data**: Multiple datasets are created through random sampling with replacement, with each dataset used to train one decision tree.
2. **Build Decision Trees**: Each decision tree is trained on a different subset of the data, with random feature selection at each split to ensure diversity among the trees.
3. **Splitting Nodes**: Each tree is grown by selecting the best split at each node based on a certain criterion (e.g., Gini impurity or Information Gain).
4. **Aggregate Predictions**: Once all trees are trained, each tree makes a prediction. For regression tasks, the final output is the average of all individual tree predictions; for classification tasks, it is the majority vote across all trees.

Why is it used?

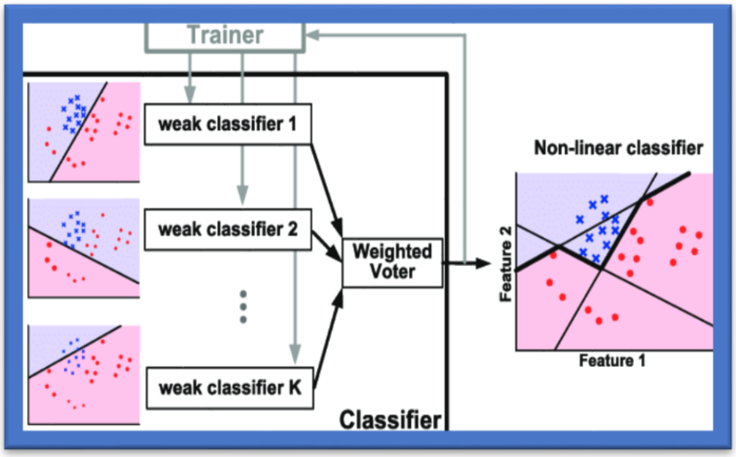
It can represent intricate correlations between features like location, size, and market movements and their effects on property prices, it is very helpful in real estate market analysis and prediction. Even when dealing with noisy or incomplete data prevalent problems in real-world datasets—its ensemble technique guarantees resilient performance. Additionally, its feature importance capabilities provide stakeholders with actionable insights by assisting in the identification of important elements driving prices. The model is perfect for properly predicting property prices across a variety of datasets in the ever-changing real estate market because of its scalability and resistance to overfitting.

Performance:

MSE 18641790000.00	RMSE 136534.9401	MAE 84586.28011	R-SQUARED 0.853969

AdaBoost (Adaptive Boosting):

It works by combining weak learners (often decision trees) into a stronger model. The algorithm focuses on the mistakes made by previous models. Initially, all data points have equal weight, but misclassified points are given higher weight in the next iteration. This helps the model focus on harder-to-classify examples, improving performance with each iteration. The final prediction is typically a weighted average of the predictions of all models.



How does it work?

1. **Initialize Weights**: Assign equal weights to all data points in the training set. These weights determine the importance of each data point in the model training.
2. **Train Weak Learner**: Train a weak learner (typically a decision tree with only one split, called a decision stump) on the weighted data.
3. **Compute Error Rate**: Calculate the error rate of the weak model. If the model misclassifies a point, the weight of that point is increased so the next model focuses on correcting these errors.
4. **Update Weights**: After each weak learner is trained, update the weights of misclassified points, increasing their importance for the next model.
5. **Combine Weak Learners**: The final prediction is the weighted sum of predictions from all weak learners, where more accurate learners are given higher weight.
6. **Repeat**: This process is repeated for a predefined number of iterations, with each new learner trying to correct the errors of the previous ones.

Why is it used?

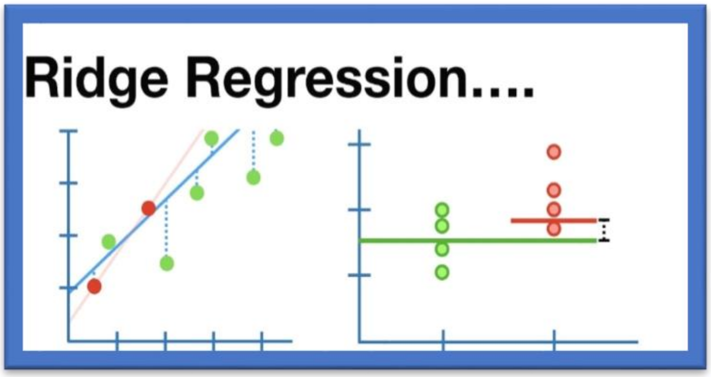
It's useful for predicting real estate prices since it can increase model accuracy by concentrating on properties that are more difficult to predict. By iteratively improving forecasts, it performs well with complicated information and is useful for managing a variety of market data. Better generalization is ensured by its boosting method, especially when handling outliers or challenging pricing trend patterns.

Performance:

MSE: 35692970000.00	RMSE:188925.8423	MAE: 138041.3569	R-SQUARED: 0.720398

Ridge:

It is a linear regression method that adds an L2 regularization term (penalty) to the loss function. This term discourages large coefficients in the model, thereby reducing overfitting. The result is a more generalizable model, as the regularization helps control the complexity of the model, ensuring that it does not fit the noise in the data.



How does it work?

1- **Fit Linear Model**: Fit a linear regression model to the dataset using the least squares method.
2- **Add Regularization**: Add an L2 regularization term to the loss function. The penalty is proportional to the square of the magnitude of the coefficients.
3- **Minimize Loss Function**: Minimize the sum of the residual sum of squares (RSS) and the penalty term to find the optimal coefficients. This prevents overfitting by discouraging large coefficient values.
4- **Output Prediction**: Use the trained model with regularized coefficients to make predictions on new data.

Why is it used?

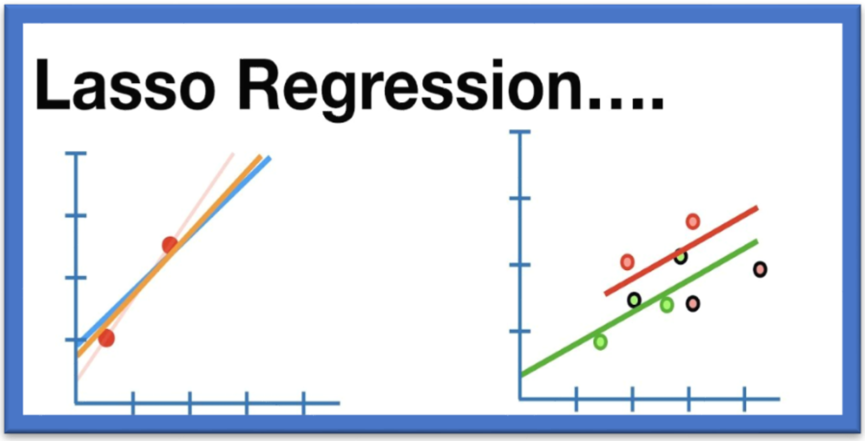
It's used in real estate to address multicollinearity, which is common when features like location, square footage, and amenities are interrelated. By controlling feature influence, it enhances model stability and ensures accurate price predictions across varying property attributes.

Performance:

MSE: 38518920000.00	RMSE: 196262.3716	MAE: 139793.8252	R-SQUARED: 0.698261

Lasso Regression

It is another linear regression method but uses L1 regularization. The L1 penalty forces some coefficients to become exactly zero, effectively selecting a subset of important features while discarding the irrelevant ones. This makes Lasso useful for feature selection in high-dimensional datasets.



How does it work?

1- **Fit Linear Model**: Fit a linear regression model to the dataset using the least squares method.
2- **Add Regularization**: Add an L1 penalty (Lasso) to the loss function, which encourages sparse coefficients. Some coefficients are driven to zero, effectively performing feature selection.
3- **Minimize Loss**: Minimize the sum of the residual sum of squares (RSS) and the L1 penalty term to find the optimal coefficients.
4- **Output Prediction**: Use the trained model with sparse coefficients to make predictions on new data.

Why is it used?

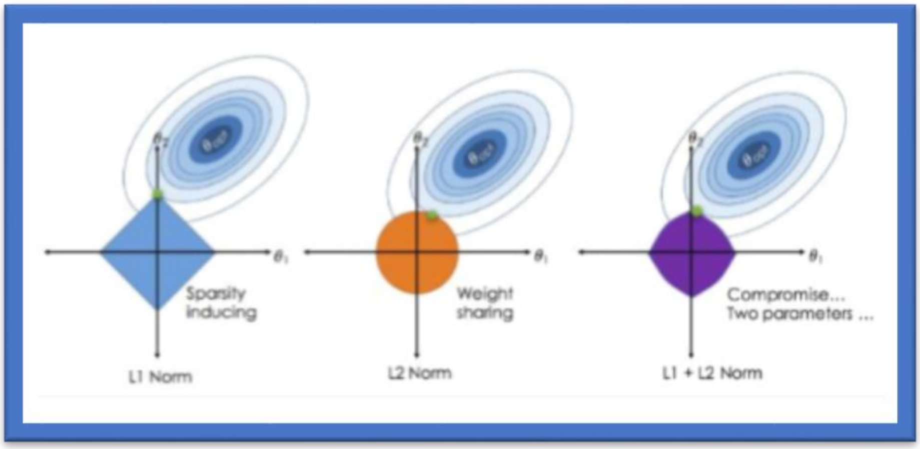
It's ideal for identifying the most significant factors influencing real estate prices. Its feature selection capability simplifies models by focusing only on the most relevant predictors, which is particularly useful when dealing with high-dimensional data.

Performance:

MSE: 38538610000.00	RMSE:196312.525	MAE: 139919.0933	R-SQUARED: 0.698106

Elastic Net:

It combines Ridge and Lasso by incorporating both L1 and L2 regularization terms. It is particularly useful when there are correlations between the features in the dataset. ElasticNet can retain some of the feature selection properties of Lasso while also stabilizing the solution, like Ridge, when features are highly correlated.



How does it work?

1- **Fit Linear Model**: Fit a linear regression model to the dataset using the least squares method.
2- **Add Regularization**: Combine both L1 (Lasso) and L2 (Ridge) penalties in the loss function. The regularization term encourages sparsity (from Lasso) and smoothness (from Ridge).
3- **Control Regularization**: Use a parameter to control the balance between the L1 and L2 penalties.
4- **Minimize Loss**: Minimize the sum of the residual sum of squares (RSS) along with the combined L1 and L2 penalties to get the optimal coefficients.
5- **Output Prediction**: Make predictions using the trained model with both L1 and L2 regularized coefficients.

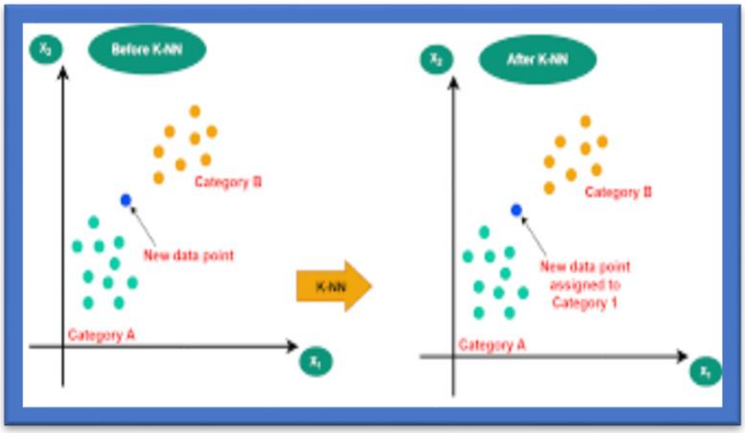
Why is it used?
It's used in real estate when there are many correlated features, such as proximity to amenities or property size and number of bedrooms. By combining the strengths of Ridge and Lasso, it ensures accurate price predictions while maintaining a balanced and interpretable model.

Performance:

MSE:38522300000.00	RMSE: 196270.9963	MAE:139812.2547	R-SQUARED: 0.698234

K-Nearest Neighbors (KNN):

It is a simple, non-psed for both classification and regression. It makes predictions based on the majority class (for classification) or the average of the values (for regression) of the k-nearest data points to the point being predicted. The "nearness" is determined by a distance metric, typically Euclidean distance, but other metrics can be used depending on the problem.



How does it work?

1. **Calculate Distance**: For each data point (test data), the distance to all other data points in the training set is calculated using a distance metric like Euclidean distance.
2. **Identify Neighbors**: The K nearest neighbors to the test point are identified by selecting the K points with the smallest distance to the test point.
3. **Make Prediction**:
   a. For regression: The predicted value is the average of the target values of the K nearest neighbors.
   b. For classification: The predicted class is the most frequent class label among the K nearest neighbors.
4. **Output Prediction**: The prediction is the class label or target value based on the majority vote or average.

Why is it used?

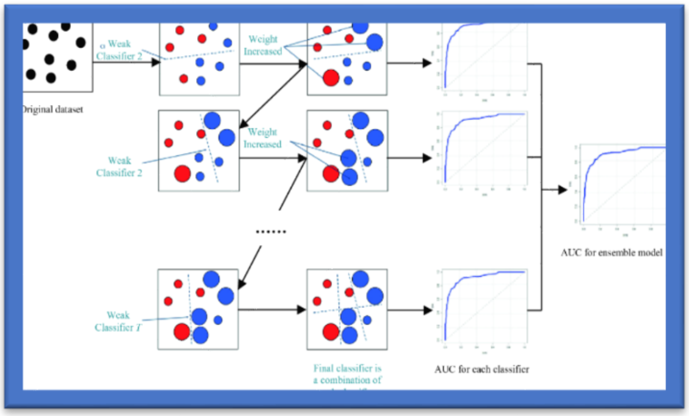
It’s effective for real estate because it captures local property trends, such as neighborhood-specific price variations. By considering nearby properties with similar features, it provides localized and highly accurate predictions.

Performance:

MSE: 23753880000.00	RMSE: 154122.9303	MAE: 94840.92408	R-SQUARED: 0.813923

Gradient Boosting:

Gradient Boosting is an ensemble learning technique that builds a strong model by combining multiple weak learners, typically decision trees. It starts with an initial prediction (e.g., the mean of the target) and iteratively trains new trees to predict the residuals (errors) of the current model. Each tree's predictions are added to the overall model, scaled by a learning rate to control their contribution. This sequential process continues until the residuals are minimized. Regularization techniques, like limiting tree depth or the number of trees, help prevent overfitting, making Gradient Boosting highly effective for regression and classification tasks.



How does it work?

1. **Initialize Model**: Start with a weak model (usually a constant prediction, like the mean of the target variable).
2. **Compute Residuals**: After the initial model is created, calculate the residuals for each data point. The residual is the difference between the actual value and the predicted value of the current model.
3. **Fit New Model**: A new weak learner (typically a small decision tree) fits these residuals. The model attempts to predict how much the previous model's errors can be corrected.
4. **Update Predictions**: The predictions from the new model are added to the previous model's predictions, which helps to minimize the residuals iteratively.
5. **Repeat**: This process is repeated for several iterations. Each subsequent model tries to correct the errors made by the previous model. The final prediction is the sum of all individual predictions.

Why is it used?

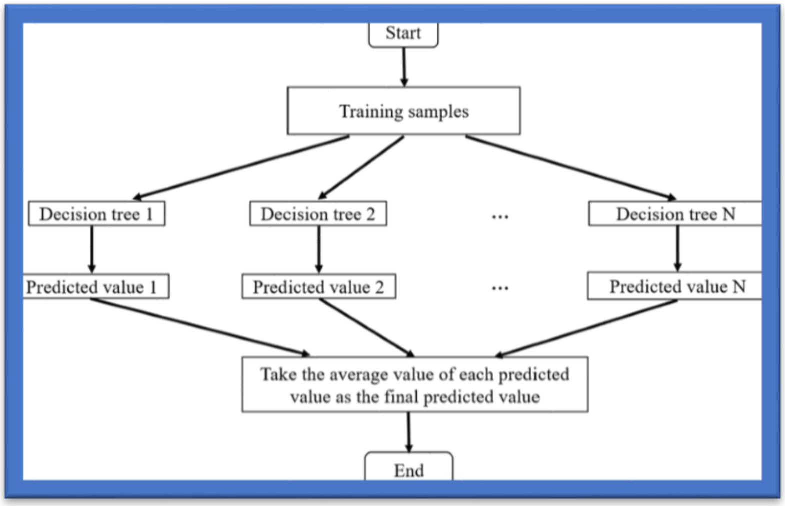
It's widely used in real estate for its ability to capture non-linear relationships between features and prices. Its iterative error correction ensures high accuracy, making it suitable for dynamic and heterogeneous property markets.

Performance:

MSE: 18143970000.00	RMSE: 134699.5434	MAE:85175.70733	R-SQUARED: 0.857869

Extra Trees (Extremely Randomized Trees)

It is like Random Forest, but the best split is selected based on some criterion, while Extra Trees selects splits randomly. This increases the diversity among the trees and can help improve generalization, often resulting in a faster and less prone-to-overfitting model compared to Random Forest.



How does it work?

1. **Bootstrap Data**: Create multiple subsets of the data by random sampling with replacement. This allows for diversity in the trees.
2. **Random Feature Selection**: At each split in the decision tree, a random subset of features is chosen, rather than considering all features. This introduces randomness to the tree building process, making it less likely to overfit.
3. **Split Nodes**: At each node of the tree, the algorithm selects the best random threshold for splitting the data. This randomness helps to avoid overfitting and reduces variance in predictions.
4. **Aggregate Predictions**: Once all trees are built, each tree makes a prediction. For regression tasks, the final prediction is the average of all individual tree predictions; for classification tasks, it's the majority vote.

Why is it used?

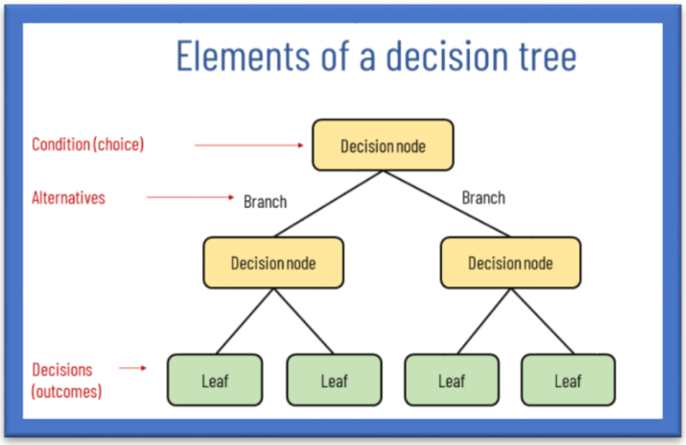
It's useful in real estate for its ability to generalize well across diverse datasets. Its random splitting reduces the risk of overfitting, ensuring stable and accurate predictions in markets with varied property types and trends.

Performance:

MSE: 16844100000.00	RMSE:129784.8073	MAE: 79642.89036	R-SQUARED: 0.868051

Decision Tree

It is a model that splits the dataset into subsets based on feature values. Each of the tree represents a decision based on a feature, and each branch represents the outcome of that decision. This process continues recursively until the tree reaches a terminal node (leaf) where the final prediction is made. Decision trees are simple to interpret but can easily be overfit, so they are often pruned or used in ensembles like Random Forests and Gradient Boosting.



How does it work?

1- **Select a Feature to Split**: At the root of the tree, the algorithm selects the feature that best splits the data, typically using metrics like Gini Impurity (for classification) or Mean Squared Error (for regression).

2- **Split Data**: The data is divided into subsets based on the chosen feature's value. This process is repeated recursively to build the tree.

3- **Repeat Splitting**: At each subsequent node, the feature that minimizes the split impurity is selected. This continues until the tree reaches a specified depth or the data can no longer be split meaningfully.

4- **Leaf Nodes**: Once the tree is built, each leaf node contains the final prediction (either a class label or a continuous value, depending on the problem).

5- **Make Prediction**: To make a prediction, the model traverses the tree from the root to a leaf node based on the input data's feature values.

Why is it used?

Are intuitive and interpretable, making them useful for understanding how different features, such as property size or location, influence prices. They can handle non-linear relationships, offering clear insights into property valuation.

Performance:

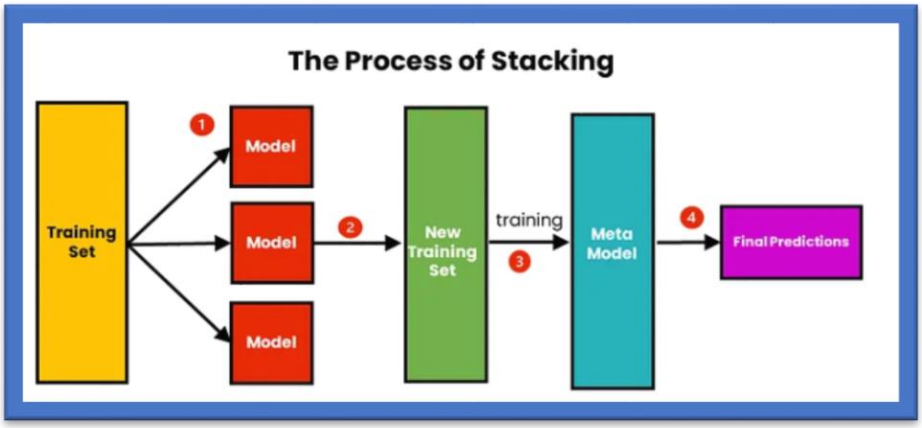
MSE: 26218550000.00	RMSE: 161921.4281	MAE: 99446.92799	R-SQUARED: 0.794616

Ensemble Methods: Stacking and Voting Regressors

To further enhance the prediction accuracy, ensemble techniques—**Stacking Regressor** and **Voting Regressor**—were implemented in the final stage of training. These methods leverage the strengths of multiple models to produce a robust prediction.

Stacking Regressor

It is an ensemble learning technique that combines the predictions of multiple machine learning models to improve overall performance. It typically involves a **meta-model** (often called the **stacking regressor** in regression tasks) that learns how to combine the predictions of several base models to achieve better results.



How it works:

1. **Train Base Models**: Several different regression models (e.g., Linear Regression, Decision Tree, Random Forest) are trained independently on the same dataset.
2. **Generate Predictions**: Once trained, these base models make predictions on the dataset. These predictions are used as input for a new model.
3. **Create Meta-Model**: A meta-model (also called a level 1 model) is trained using the predictions from the base models as new features. The goal is for the meta-model to learn how to combine the predictions from the base models to improve overall performance.
4. **Final Prediction**: The meta-model generates a final prediction by weighing the outputs of the base models. This model can outperform individual base models because it leverages diversity in the base models.

Why is it used?

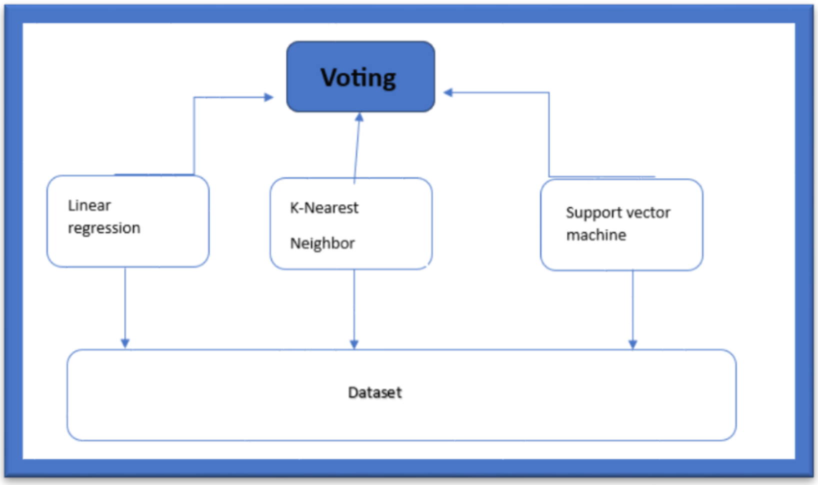
The real estate market is highly complex, with a wide range of factors affecting property prices. A **Stacking Regressor** combines multiple base models to improve prediction accuracy, reduce the likelihood of overfitting, and handle various complexities and relationships in the data. By using a stacking approach, it's possible to build more accurate and robust models that generalize better, making it ideal for real estate price prediction tasks.

Performance:

MSE:16125130000.00	RMSE: 126984.7737	MAE: 78406.78586	R-SQUARED: 0.873683

Voting regressor:

A voting regressor can be defined as a special method that combines or 'ensembles' multiple regression models and overperforms the individual models present as their estimators. The mathematical concept of a voting regressor is quite easy and very similar to that of a voting classifier. If we consider a crowd of machine learning models as M_1, M_2,\dots , M_x then M_n each model will produce a prediction P_n for a given input data I . Now if we pass it through Voting Regressor then the final prediction will be P_{voting} . Now, we can choose simple average mode which uniformly distributes the total weight to all the models, or we can choose custom-specified weights for each model which is called Weighted averaging.



How does it work?

1. **Train Multiple Models**: Train several different base regression models (e.g., Linear Regression, Random Forest, Ridge Regression) on the data.
2. **Generate Predictions**: Each base model makes a prediction based on the input data.
3. **Combine Predictions**: The final prediction is computed by averaging the predictions from all models (for regression tasks). This helps to reduce the variance and overfitting of individual models.
4. **Output Prediction**: The aggregated prediction is the output of the Voting Regressor.

Why is it used?

The **Voting Regressor** is an efficient and interpretable ensemble learning method for real estate price prediction. It ensures robust, accurate, and reliable predictions by aggregating outputs from multiple models, leveraging their unique strengths, and mitigating their weaknesses. This makes it particularly valuable for navigating the complex and dynamic nature of real estate markets.

Performance:

MSE: 20890680000.00	RMSE:144536.094	MAE: 95783.82816	R-SQUARED: 0.836352

Summary of Results

The table below summarizes the model's result:

Model	MSE	RMSE	MAE	R-squared
Stacking-Regressor	16125130000.00	126984.7737	78406.78586	0.873683
Extra-Trees	16844100000.00	129784.8073	79642.89036	0.868051
Gradient-Boosting	18143970000.00	134699.5434	85175.70733	0.857869
Random-Forest	18641790000.00	136534.9401	84586.28011	0.853969
Voting-Regressor	20890680000.00	144536.094	95783.82816	0.836352
K-Nearest-Neighbors	23753880000.00	154122.9303	94840.92408	0.813923
Decision-Tree	26218550000.00	161921.4281	99446.92799	0.794616
AdaBoost	35692970000.00	188925.8423	138041.3569	0.720398
Ridge	38518920000.00	196262.3716	139793.8252	0.698261
ElasticNet	38522300000.00	196270.9963	139812.2547	0.698234
Lasso	38538610000.00	196312.525	139919.0933	0.698106

To conclude:

- **Tree-Based Ensembles (Extra-Trees, Gradient Boosting, and Random Forest)** emerge as reliable alternatives to the Stacking Regressor, providing strong performance and low errors.
- Models like **Ridge, ElasticNet, and Lasso** do not capture the non-linear complexities of real estate data, making them less suitable.

- **KNN and AdaBoost** underperform, likely due to their inherent limitations in handling this type of data.
- The **Stacking Regressor** is the clear winner, achieving the lowest errors (MSE, RMSE, and MAE) and the highest R-squared value. This indicates its ability to leverage the strengths of multiple models, making it highly effective for real estate price prediction.

Python packages and modules used in this project

Here's a brief explanation of the packages we used in our models:

1. **TensorFlow:** A comprehensive open-source library for machine learning and deep learning. It allows users to build and train machine learning models efficiently, including neural networks, using GPU acceleration.
2. **Matplotlib:** A Python plotting library used to create static, animated, and interactive visualizations. The pyplot module simplifies creating basic charts like line graphs and histograms.
3. **Seaborn:** A statistical data visualization library based on Matplotlib. It provides high-level APIs for creating visually appealing and informative charts such as heatmaps and pair plots.
4. **IterativeImputer:** A scikit-learn experimental tool that imputes missing values by iteratively modeling each feature as a function of the others.
5. **train_test_split:** A utility from scikit-learn that splits datasets into training and testing subsets for proper evaluation of machine learning models.
6. **OneHotEncoder:** A preprocessing tool from scikit-learn that converts categorical data into one-hot numeric arrays for machine learning.
7. **SVR (Support Vector Regressor):** A regression technique in scikit-learn that finds a hyperplane that best fits the data while maximizing margin tolerance.
8. **Requests:** A popular Python library for making HTTP requests. It simplifies the process of sending HTTP requests and handling responses.
9. **Pandas** A powerful data manipulation and analysis library for Python. It provides data structures and functions to efficiently manipulate and analyze structured data.

10. **BeautifulSoup** is a Python library used for web scraping purposes. It allows you to extract and parse data from HTML or XML documents.
11. **Selenium** is a powerful tool used for automating web browsers. It is often employed for tasks such as web scraping, testing web applications, or performing repetitive browser actions.
12. **Linux Contabo server** is a virtual private server (VPS) or dedicated server provided by Contabo, a web hosting company. These servers run on a Linux operating system, which is popular for hosting websites, applications, and databases due to its stability, performance, and open-source nature.
13. **ArcPy:** it is designed to provide powerful scripting capabilities for geographic data analysis, management, and visualization. It is primarily used with **ArcGIS Pro** and **ArcMap**, Esri's flagship GIS software, and serves as a bridge for automating GIS workflows and extending the functionality of ArcGIS tools.

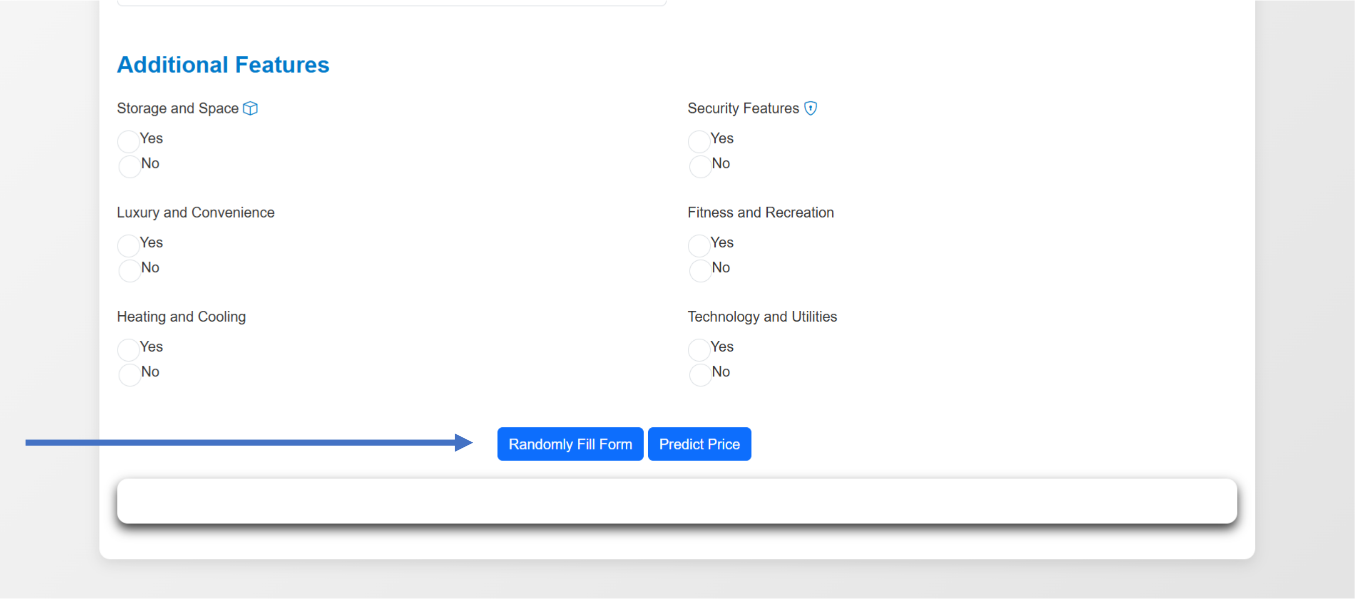
These packages are widely used in GIS, Data Science, data analysis, machine learning, and related fields.

Sneak peek on the website

Leveraging location-based variables, user-provided property details, and additional features, our website employs machine learning to predict real estate prices with precision. Designed to simplify property valuation, this tool offers an intuitive interface tailored to meet the unique needs of each user, enabling them to explore potential prices effortlessly.

1. **Property Details Input**:
    - Users can input essential property information, including:
        - **Size (m²)**: A mandatory field to specify the property's area.
        - **Number of Bathrooms** and **Bedrooms**: Customize the details to reflect the property's amenities.
        - **Governorate**: Select the property location for more accurate predictions.
2. **Proximity to Essential Facilities** (Distances in meters):
    - Users can input distances from critical infrastructure:
        - **Private and Public Universities**
        - **Hospitals, Nurseries, and Pharmacies**
        - **Schools**
3. **Additional Features**:
    - Users can enhance the prediction by providing:
        - **Elevation**: Specify the property's elevation using Digital Elevation Models (DEM).
        - **Land Use and Land Cover (LULC)**: Choose from predefined LULC categories.
        - **Population**: Enter the local population density.
    - Feature-specific options for improved predictions:

- **Storage and Space**: Indicate the availability of storage.
- **Security Features**: Highlight safety-focused aspects.
- **Luxury and Convenience**: Specify luxury elements such as premium finishes.
- **Fitness and Recreation**: Include fitness facilities or recreational spaces.
- **Heating and Cooling**: Consider advanced HVAC systems.
- **Technology and Utilities**: Add details about smart technologies or modern utilities.



Additional Features

Storage and Space
○ Yes
○ No

Security Features
○ Yes
○ No

Luxury and Convenience
○ Yes
○ No

Fitness and Recreation
○ Yes
○ No

Heating and Cooling
○ Yes
○ No

Technology and Utilities
○ Yes
○ No

To enhance user experience and facilitate easy exploration of the platform, the website includes a **"Randomly Fill Form"** button. This feature allows users to populate the input fields with randomly generated sample data to test the system's capabilities and predictions.

The **"Randomly Fill Form"** button is particularly helpful for users who want to quickly understand how the tool works without manually inputting data. By using this feature, users can observe how the machine learning models analyze various attributes (such as location, property type, and size) to generate predictions for real estate prices. This functionality ensures the platform remains accessible and user-friendly, catering to individuals with diverse levels of technical expertise.

Challenges

1. Lack of existing real estate datasets in Lebanon, necessitating the creation of a custom dataset.
2. Variability in data structure, format, and quality across multiple web scraping sources.
3. Most of sources doesn’t have a precise coordinate of the real estate.
4. Difficulties in geolocation and associating property data with accurate spatial coordinates.
5. Identifying the most suitable machine learning model among various tested options.

6. Handling computationally intensive tasks like data preprocessing, spatial analyses, and model training.
7. Integrating ArcGIS pro services into the full data processing.

Future work

Future improvements to the project could include:

Data collection and automation process

- **Expanding Data Sources**: Integrate additional real estate data sources to ensure comprehensive coverage.
- **User Interface Development**: Create a user-friendly interface to manage the scraping and processing pipeline efficiently.
- **Code Optimization**: Refactor the codebase to improve readability, maintainability, and scalability.
- **Data Versioning and Historical Analysis**: Implement a data history management system to store multiple versions of real estate data, enabling the analysis of historical price trends.
- **Integrating Comprehensive Data**: Enrich the dataset with additional features, including economic indicators, historical price trends, and user-generated content, to provide more granular and accurate forecasts.

These enhancements aim to streamline data collection and analysis while providing valuable insights into real estate trends for future projects.

Spatial analysis

To further enrich my dataset and improve property valuation insights, future enhancements to spatial analysis include:

- **Integration of Additional Spatial Features**: Incorporate key location-based attributes such as proximity to parks, crime hotspots, and traffic congestion patterns to provide more detailed insights into property values.
- **Inclusion of Economic Indicators**: Enhance predictive accuracy by integrating external economic factors, including interest rates, inflation trends, and employment statistics, to account for broader market influences on real estate prices.

These advancements aim to deepen the analytical scope and improve the precision of real estate market predictions.

Advanced Machine Learning and Deep Learning Models

Building on the solid foundation of this project, future developments will focus on incorporating advanced machine learning algorithms to enhance prediction accuracy:

- **Adopting XGBoost**: Utilize XGBoost for its built-in regularization, scalability, and ability to handle large, high-dimensional datasets, making it particularly effective for capturing the complexities of the real estate market.
- **Exploring Deep Learning Techniques**: Leverage advanced models such as Recurrent Neural Networks (RNNs) and Multi-Layer Perceptron (MLPs) to better capture intricate temporal patterns and property-related dependencies.

By combining these advanced methodologies with an extended feature set, this project can achieve significantly improved predictive accuracy and unlock deeper insights into the dynamics of the real estate market.

User Experience

Enhancing User Experience with Recommendations and Functionality

- **Personalized Recommendation System**:
  Integrate a recommendation system to deliver tailored property suggestions based on user preferences and interactions. A hybrid approach, combining content-based and collaborative filtering techniques, can ensure precise and relevant recommendations. By leveraging insights from the price prediction model, the system can highlight value-for-money properties or those aligned with current market trends, boosting user engagement and decision-making.
- **Improving Usability and Functionality**:
  - **Interactive Visualization Dashboards**: Develop comprehensive dashboards featuring charts, graphs, and maps to offer users an intuitive, interactive view of property data and predictions.
  - **Mobile Application Development**: Create a mobile app to provide a seamless and user-friendly experience, enabling users to access property predictions and related data conveniently anytime, anywhere.

These enhancements aim to improve user interaction, accessibility, and satisfaction while delivering actionable insights for informed real estate decisions.

Conclusion

The machine learning-based real estate price prediction project is the first step in incorporating data-driven insights into property appraisal. The project has effectively developed a model that uses key characteristics to forecast real estate values and presents the findings via an easy-to-use online interface, in addition to an effective automation tool for web scraping. The model offers dependable forecasts that can help users comprehend market trends and property prices by utilizing pertinent features and applying strong preprocessing techniques.

Although the current solution provides a useful platform, it is still a simple tool devoid of sophisticated features like interactive elements or a recommendation system. Future improvements, such as the addition of interactive dashboards, personalized recommendations, and a mobile-friendly design to increase user engagement and functionality, are made possible by this simplicity.

In summary, the project establishes a strong foundation for future advancement and creativity in real estate analytics, offering a clear route for future research to increase its potential and influence.

References

1. https://www.geeksforgeeks.org/voting-regressor/
2. https://medium.com/@brijesh_soni/stacking-to-improve-model-performance-a-comprehensive-guide-on-ensemble-learning-in-python-9ed53c93ce28
3. https://statisticsbyjim.com/regression/root-mean-square-error-rmse/
4. https://www.deepchecks.com/glossary/mean-absolute-error/
5. https://analyticsarora.com/the-most-important-things-you-need-to-know-about-elastic-net/
6. https://why-change.com/2021/11/13/how-to-create-decision-trees-for-business-rules-analysis/
7. https://www.spotfire.com/glossary/what-is-a-random-forest
8. Guide to Build Real Estate Price Prediction Model using ML algorithms | by Binisha Banjara | Medium
9. Search-CNKI
10. (PDF) House price prediction based on different models of machine learning