House Price Predictor Using Machine Learning

1st Aryan Malhotra
dept. SCOPE - 22BDS0207
Vellore Institute of Technology
Vellore, India
aryan.malhotra2022@vitstudent.ac.in

2nd Krishang Ratra
dept. SCOPE - 22BCE0904
Vellore Institute of Technology
Vellore, India
krishang.ratra2022@vitstudent.ac.in

***Abstract*—The house price predictor project leverages machine learning to estimate property prices in Mumbai, providing a practical solution for potential buyers, sellers, and real estate professionals. The dataset, sourced from Kaggle, comprises 76,038 entries with nine attributes, including BHK, property type, locality, area, price, region, construction status, and property age. The model focuses on three primary parameters—region, area in square feet, and BHK to deliver accurate price predictions.**

The project integrates a Flask-based back-end and a user-friendly front-end interface to enable seamless interaction. Users can input property details and instantly obtain predicted prices. The system was hosted using Postman, ensuring accessibility and ease of use. This streamlined and data-driven approach offers a robust tool for real estate price estimation, bridging the gap between market trends and user expectations.

I. Introduction

Accurate real estate price estimation is crucial for both buyers and sellers in the highly dynamic housing market. With property prices varying widely based on location, size, type, and several other factors, it becomes essential to have a reliable system that can predict housing prices effectively. This project addresses this challenge by developing a house price prediction model tailored specifically to properties in Mumbai, India.

The dataset for this project, sourced from Kaggle, includes 76,038 records with nine key parameters such as BHK, property type, locality, area, price, region, construction status, and property age. By analyzing these factors, the project identifies region, area in square feet, and BHK as the most significant predictors of price, simplifying the prediction process while maintaining accuracy.

The system is built using machine learning algorithms and is supported by a Flask backend, which handles computations and predictions. A user-friendly frontend interface allows users to input their property details and receive instant predictions. By hosting the application with Postman, the project ensures accessibility for diverse users.

This innovative approach not only simplifies the price estimation process but also serves as a valuable tool for real estate professionals, homeowners, and potential buyers seeking data-driven insights into the housing market.

II. About the Dataset

The dataset used in this project was sourced from Kaggle and focused on house pricing data in Mumbai. It consists of 76,038 rows (data points) and 9 key parameters that describe various aspects of properties. The parameters include BHK (number of bedrooms), type (such as apartment, villa, etc.), locality (specific area within Mumbai), area (property size in square feet), price (price of the property), price-unit (unit of price, either Lakhs or Crores), region (geographical region within Mumbai), status (whether the property is ready to move or not), and age (whether the property is new or old).

The dataset provides a detailed representation of property characteristics, allowing for predictions based on the relationship between these attributes. The target variable for the model is the price, with the primary predictors being region, area, and BHK, as these factors are most indicative of property value. The dataset is comprehensive, covering a wide range of properties across different regions, which helped in building a robust model capable of predicting house prices with high accuracy.

TABLE I
Data Table

bhk	area	price	region
2	650	262.0	Agripada
3	685	250.0	Andheri West
2	876	59.98	Panvel

Parameters that affect the price

III. Implementation

A. Data Pre-processing

The dataset was initially grouped by the 'type' parameter to analyze property distribution and gain insights. To simplify the data, unnecessary columns such as 'locality', 'status', and 'age' were dropped. Rows containing missing values were removed to maintain data consistency, ensuring a cleaner dataset for analysis. Additionally, all property prices were standardized to a single unit (Lakhs) for uniformity and easier comparison.

A new feature, 'price-per-sqft', was introduced, calculated as

$$(price * 100000)/area$$

to provide a standardized measure of property value. The 'region' values were also cleaned and standardized by trimming unnecessary whitespace and consolidating less frequent regions into a single category labeled as 'other'.

B. Outlier Removal

To improve the dataset quality, several outlier removal techniques were applied. Entries where the area per BHK was less than 300 sqft were excluded, as they were deemed unrealistic. Outliers in 'price-per-sqft' were identified and removed using the mean and standard deviation for each region, ensuring a more consistent price distribution. Additionally, for each region, 2+ BHK properties with prices significantly lower than the mean price per sqft of 1 BHK properties were filtered out, further refining the dataset for accurate price predictions.

C. Exploratory Data Analysis(EDA)

The distribution of 'price-per-sqft' across different properties was visualized to gain insights into the spread of property prices. Additionally, scatter charts were plotted for specific regions, such as Chembur, to analyze price trends across varying property sizes. These visualizations helped identify patterns and relationships between property area, price, and region, providing a clearer understanding of the data before model training.

D. Model Development and Training

For feature transformation, categorical variables, particularly 'region', were converted into one-hot encoded features using 'pd.get-dummies', ensuring compatibility with machine learning algorithms. The dataset was then consolidated into a numerical format to prepare it for model training. To ensure accurate evaluation, the data was split into training and testing sets using an 80:20 ratio, which allowed the models to be trained and tested effectively.
Several models were implemented, including Linear Regression as the baseline model, Lasso Regression to mitigate over-fitting, and Decision Tree Regressor to capture non-linear relationships in the data. Cross-validation was performed using 'ShuffleSplit' and 'GridSearchCV' to fine-tune hyper-parameters and assess model performance. After evaluating the models, the Decision Tree Regressor was selected as the final model, based on its superior performance in predicting house prices.

E. Prediction and Deployment

A prediction function, 'predict-price', was developed to allow users to input property details, such as region, area, and BHK, and receive predicted house prices. The function encodes the user inputs into a format compatible with the trained model and returns the predicted price, ensuring a seamless interaction between the model and the user.
To ensure the model could be reused in the future, it was serialized using the pickle library. This allowed the trained model to be saved and loaded when needed. Additionally, a JSON file (columns.json) was created to store the feature names, ensuring consistency when handling user inputs during deployment and making the process more robust and reliable.
The back-end was integrated into a Flask server, which processed user inputs and used the trained model to generate predictions. A user-friendly front-end interface was also built, enabling users to easily input property details. The application was hosted using Postman for testing and accessibility, allowing for real-time interaction and further validation of the system's performance.

IV. Machine Learning Models

A. Linear Regression

A basic statistical model called linear regression shows that the dependent variable (home price) and the independent variables (features) have a linear relationship. The model makes the assumption that a linear equation can be used to forecast home prices:

$$price = (w1 * area) + (w2 * bhk) + (w3 * region) + b$$

where w1, w2 and w3 are coefficients(weights), and b is the bias term.
Advantages:

- Minimal training time and computationally efficient.
- Analysing and interpreting feature importance is simple.
- Ideal for datasets with roughly linear connections between variables.

Disadvantages:

- Makes the frequently implausible assumption that features and home prices have a linear connection.
- Poor performance on non-linear and high variance datasets.
- Sensitive to outliers and multi-collinearity.

B. Polynomial Regression

By adding higher-degree terms to model non-linearity, Polynomial Regression expands on Linear Regression:

$$price = (w1 * area) + (w2 * bhk^2) + (w3 * region^3) + b$$

Advantages:

- Outperforms linear regression in capturing non-linear relationships.
- More adaptable when fitting intricate datasets.

Disadvantages:

- Overfitting may occur while dealing with high-degree polynomials.
- More complicated to calculate than linear regression.

C. Lasso Regression

Lasso (Least Absolute Shrinkage and Selection Operator) is a regularised linear regression model that selects features by applying an L1 penalty to the loss function. The mathematical formula for Lasso regression is:

$$\hat{\beta} = \arg\min_{\beta} \sum_{i=1}^{n} (y_i - X_i\beta)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

where,

- y_i is the actual house price,
- X_i represents the input features (e.g., BHK, area in sqft, region),
- β are the model coefficients,
- λ is the regularization parameter, which controls feature selection.

Lasso regression reduces some feature coefficients to zero, thereby removing less important variables, making it suitable for models with high-dimensional data and associated features. Lasso regression was applied to predict real estate prices in Mumbai based on various factors such as region, area in sqft., BHK.
Using GridSearchCV, the best hyperparameters for Lasso were:

$$\alpha = 1, selection = random$$

The model achieved a best score (R^2) of 0.7193, which indicates moderate predictive accuracy but was inferior to Decision Tree Regression.

D. Decision Tree Regressor

Decision Tree Regression is a non-linear supervised learning approach that recursively splits data into homogenous subsets to predict the connection between input features and target variables. It has a tree-like structure, with decisions made based on threshold conditions on feature values.
The Decision Tree model's mathematical representation is built on minimising impurities at each split using a criterion such as Mean Squared Error (MSE):

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2$$

where,

- y_i is the actual house price,
- $\hat{y}$ is the predicted house price,
- N is the number of samples in the node.

V. Model Selection

A pipeline was developed utilising GridSearchCV to systematically select the best-performing model by evaluating multiple machine learning methods and hyperparameters.

A. Grid Search with Cross-Validation (GridSearchCV)

GridSearchCV does an exhaustive search over a hyperparameter grid for each model. It uses k-fold cross-validation to measure generalisation performance while avoiding overfitting. The important steps are:
1. Defining numerous models and their hyperparameter grids.
2. Using 5-fold cross-validation to assess model performance.
3. Choose the optimal hyperparameter setup for each method.
4. Comparing models based on validation ratings.
Mathematically, for a given model M with hyperparameter set Θ, Grid Search optimization is defined as:

$$M^* = \arg\min_{M\in\mathcal{M}} \frac{1}{k}\sum_{i=1}^{k} L(M_\Theta, X_\text{train}^{(i)}, X_\text{val}^{(i)})$$

Where,

- M^* is the best model selected,
- M represents all candidate models,
- k is the number of folds in cross-validation,
- $L(M_\Theta, X_\text{train}^{(i)}, X_\text{val}^{(i)})$ is the loss function (Mean Squared Error, MSE),
- $X_\text{train}^{(i)}$ and $X_\text{val}^{(i)}$ are the training and validation sets in the i^th fold.

Model	**Best Score** (R^2)
Linear Regression	0.8778
Lasso Regression	0.7193
Decision Tree	0.9523

TABLE II
Model Comparison with Grid Search Optimization

VI. Conclusion

The house price predictor model achieved an impressive accuracy of 95%, making it a reliable tool for predicting property prices in Mumbai based on key factors such as region, area, and number of bedrooms (BHK). Through effective data pre-processing, feature engineering, and the use of a Decision Tree Regressor, the project demonstrated the successful application of machine learning techniques in the real estate domain. Additionally, the integration of the predictive model with a user-friendly front-end and a Flask-based back-end ensures seamless user interaction, providing a quick and efficient solution for estimating house prices. This project highlights the potential of machine learning to address real-world challenges with high accuracy and practical usability.

VII. Future Work

1. Incorporating Additional Features: Future iterations could consider adding more detailed parameters, such as proximity to amenities, transportation facilities, or crime rates, to improve prediction accuracy further.
2. Expanding the Dataset: Expanding the dataset to include other cities or regions would enable the model to generalize better and provide insights into broader real estate trends.
3. Advanced Modeling Techniques: Exploring more sophisticated models, such as ensemble methods (e.g., Random Forest, XGBoost) or deep learning, could improve the system's performance and handle complex relationships in the data.
4. Dynamic Pricing Updates: Integrating real-time data from property listing websites or APIs could keep the system up-to-date with the latest market trends.
5. Deployment on Cloud Platforms: Hosting the application on a cloud platform, such as AWS, Azure, or Google Cloud,

could make it more scalable and accessible to a larger audience.
6. Mobile Compatibility: Developing a mobile app version of the system would enhance user accessibility and engagement.

VIII. References

1. https://www.kaggle.com/datasets/dravidvaishnav/mumbai-house-prices/data