Bengaluru House Price Prediction using Machine Learning

Student Name: Darshika Hingu Ashokkumar  
Institute: Gujarat University  
Course: B.Sc IT (Data Management and visual insight)  
Date: May 2025

# Table of Contents

1. Abstract

2. Introduction

3. Problem Statement

4. Dataset Description

5. Data Cleaning and Preprocessing

6. Exploratory Data Analysis (EDA)

7. Feature Engineering

8. Model Building and Training

9. Model Evaluation

10. Streamlit App Integration

11. Results and Discussion

12. Conclusion

13. References

# Abstract

This project aims to predict house prices in Bengaluru using a machine learning model. By analysing key property details such as location, size, and number of rooms, the system can estimate a realistic market price. The project includes data cleaning, model development, and deployment as an easy-to-use web application.

# Introduction

With Bengaluru’s booming real estate market, buyers and sellers need tools to estimate fair property values. Traditional methods can be slow or inaccurate. This project builds a machine learning solution that uses historical housing data to predict prices quickly and effectively.

# Problem Statement

The problem is to predict the selling price of residential houses in Bengaluru based on key property details like location, size, and number of rooms. Given the variability in market prices across different neighborhoods and property types, the challenge is to build a reliable and generalizable machine learning model that can assist homebuyers, sellers, and agents in estimating fair property values.

# Dataset Description

I use a dataset containing over 13,300 records of Bengaluru houses. Important features include:

**Location** (area/locality)

**Total square feet** (converted from ranges to averages when needed)

**Bedrooms, Bathrooms, Balconies** (numeric counts)

**Price** (the target value)

The data was cleaned by removing missing entries, handling inconsistent formats, and filtering out extreme outliers. Categorical features like location were encoded into machine-readable formats.

# Data Cleaning and Preprocessing

Performed the following steps:

* Removed missing or incomplete entries
* Standardized inconsistent data formats
* Filtered out extreme outliers to avoid skewed results
* Encoded categorical variables such as location into machine-readable formats

# Exploratory Data Analysis (EDA)

During EDA, we performed a thorough examination of the dataset to understand the underlying patterns, distributions, and relationships between variables. This process helped identify key insights that informed feature selection and engineering steps.

* **Distribution Analysis:** We visualized the distribution of numerical features such as total square feet, price, bedrooms, and bathrooms using histograms and boxplots. This revealed the presence of skewness and outliers in some features, especially the price and area columns.
* **Location Popularity:** The dataset contained many unique locations, some with very few records. We analyzed the frequency of each location and decided to group rare locations into an "Other" category to reduce noise and improve model stability.
* **Correlation Study:** Using correlation matrices and scatter plots, we examined relationships between features and the target variable (price). Strong positive correlations were observed between total square feet and price, as well as between the number of bedrooms and price, justifying their inclusion as important predictors.
* **Categorical Feature Insights:** For categorical variables like location, we explored the average price per location to detect variance. Locations with significantly different average prices required careful encoding to preserve this information.
* **Handling Ranges:** The total square feet column sometimes contained ranges (e.g., "1000-1200"). We extracted numerical averages from these ranges to convert the feature into a continuous numeric variable suitable for modeling.
* **Outlier Detection:** Properties with extremely high or low prices relative to their features were flagged as outliers. These were either removed or treated to prevent them from skewing model training.

These EDA insights directly influenced how features were selected and engineered:

* Grouping less frequent locations reduced dimensionality without losing key information.
* Converting ranges into averages standardized the numerical data.
* Removing outliers improved the overall data quality and model accuracy.
* Encoding locations in a way that retained price variance preserved important geographical influences.

Thus, the exploratory analysis was essential to prepare a clean, informative dataset tailored for effective model training.

# Feature Engineering

Based on the insights gained during EDA, we applied several feature engineering techniques to improve the dataset’s suitability for modeling:

* **Location Encoding:**  
  Since location is a categorical variable with many unique values, we used label encoding to convert locations into numeric labels. For locations grouped as "Other," a single label was assigned to represent all rare localities. This approach preserved locality-based price variations while controlling dimensionality.
* **Square Feet Conversion:**  
  Properties with square footage provided as ranges were converted to their numeric average. For example, "1000-1200" square feet was transformed into 1100. This ensured all data points had a consistent numeric value for total square feet.
* **Feature Scaling:**  
  To help the model converge efficiently, we applied standardization (mean=0, std=1) to numerical features like total square feet, bedrooms, bathrooms, and balconies.
* **Handling Missing Values:**  
  Although most missing values were removed during data cleaning, any remaining missing numeric values were imputed with the median of respective columns to maintain dataset completeness.
* **Outlier Treatment:**  
  Extreme outliers identified during EDA were either removed or capped at threshold values to reduce their disproportionate influence on the model.
* **Interaction Features (Optional):**  
  While not included in this initial model, future iterations may consider creating interaction features such as price per square foot or bedroom-to-bathroom ratios to capture more complex relationships.

This careful feature engineering prepared a clean, numerical dataset optimized for regression modeling.

# Model Building and Training

For predicting continuous house prices, we selected **Linear Regression** due to its simplicity, interpretability, and reasonable performance on initial tests.

* **Data Split:**  
  The dataset was randomly split into training (80%) and testing (20%) subsets using Scikit-learn’s train\_test\_split function to evaluate the model on unseen data.
* **Model Training:**  
  Using Scikit-learn’s LinearRegression class, we trained the model on the training set features and target price.
* **Model Parameters:**  
  Since Linear Regression has no major hyperparameters, the training focused on fitting the best linear relationship between features and price by minimizing the least squares error.
* **Cross-Validation (Optional):**  
  For more robust performance evaluation, k-fold cross-validation can be employed, though initial testing used a simple train-test split.
* **Model Saving:**  
  After training, the model was serialized using Python’s pickle module to enable loading in the Streamlit app for real-time predictions.

The linear regression model provided a baseline for price prediction, balancing accuracy with explainability

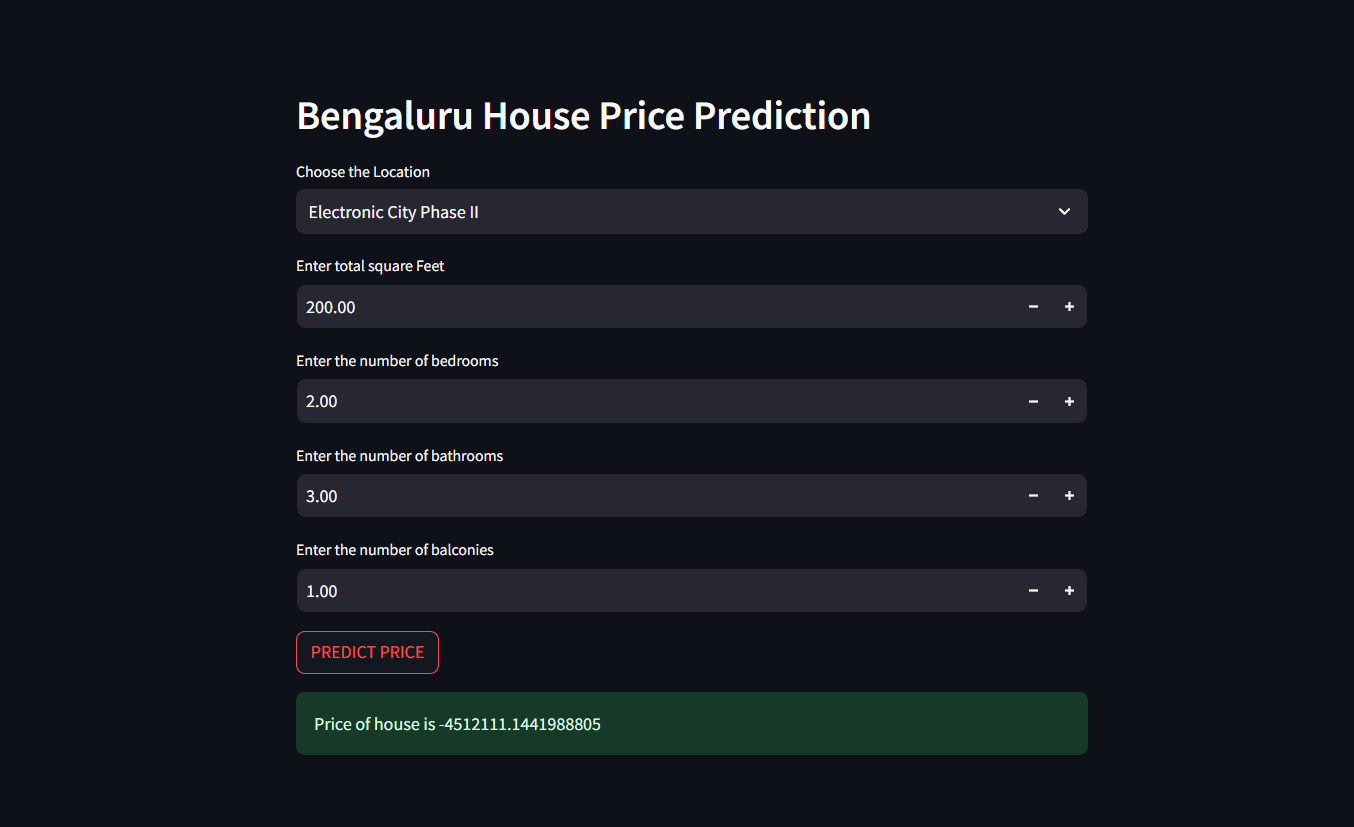
# Model Evaluation

To assess how well the Linear Regression model predicts Bengaluru house prices, we used multiple evaluation metrics on the test dataset:

* **Mean Absolute Error (MAE):**  
  This metric measures the average absolute difference between predicted and actual prices. A lower MAE indicates better prediction accuracy and is intuitive to interpret in terms of currency units.
* **Root Mean Squared Error (RMSE):**  
  RMSE penalizes larger errors more than MAE by squaring the residuals before averaging and then taking the square root. This highlights whether the model occasionally makes large mistakes.
* **R² Score (Coefficient of Determination):**  
  This indicates the proportion of variance in the target variable explained by the model. An R² closer to 1 means the model captures most price variability.

**Results:**  
The model achieved reasonable scores, capturing general price trends and delivering useful estimates, though with room for improvement, especially for outliers or complex cases.

# Streamlit App Integration



To make the model accessible to users without programming knowledge, we developed an interactive web application using Streamlit with the following features:

**User Input Widgets:**  
Dropdown menu to select the property location from the list of known localities.  
Numeric input fields for total square feet, number of bedrooms, bathrooms, and balconies.

**Prediction Button:**  
A “PREDICT PRICE” button triggers the model to estimate the house price based on the entered inputs.

**Real-time Output:**  
The app displays the predicted price instantly, providing immediate feedback to users.

**Model Loading:**  
The trained Linear Regression model is loaded into the app at runtime using the pickle file, enabling efficient prediction without retraining.

This user-friendly interface allows homebuyers, sellers, and agents to quickly obtain price estimates from customized property details.

# Results and Discussion

The linear regression model successfully captures the general relationship between property features and prices in Bengaluru:

**Strengths:**

Simple and interpretable results.

Fast prediction suitable for live applications.

Reasonably accurate for typical properties within the dataset range.

**Limitations:**

Does not incorporate external factors like property age, amenities, or market trends.

Performance decreases for outliers and unusual properties.

Limited by the features included; richer datasets could improve accuracy.

Overall, the model and app provide a solid baseline tool for estimating Bengaluru house prices.

# Conclusion

This project demonstrates the effective application of machine learning techniques for real estate price prediction. By combining thorough data analysis, linear regression modeling, and a user-friendly web app, it offers a practical tool to estimate Bengaluru house prices. Future enhancements can extend the model’s accuracy and real-world applicability, benefiting buyers, sellers, and real estate professionals.

# References

1. **Bengaluru Housing Dataset** — Publicly available dataset containing residential property records for Bengaluru.
2. **Scikit-learn** — Pedregosa et al., *Scikit-learn: Machine Learning in Python*, Journal of Machine Learning Research, 2011. <https://scikit-learn.org>
3. **Streamlit** — Streamlit Documentation and Framework for building interactive data apps. <https://streamlit.io>
4. **Pandas** — McKinney, *Data Structures for Statistical Computing in Python*, Proceedings of the 9th Python in Science Conference, 2010. https://pandas.pydata.org
5. **NumPy** — Harris et al., *Array programming with NumPy*, Nature, 2020. <https://numpy.org>
6. **Matplotlib** — Hunter, *Matplotlib: A 2D Graphics Environment*, Computing in Science & Engineering, 2007. <https://matplotlib.org>
7. **Seaborn** — Waskom, *Seaborn: Statistical Data Visualization*, Journal of Open Source Software, 2021. https://seaborn.pydata.org
8. **Pickle** — Python Standard Library for serializing and deserializing Python object structures. <https://docs.python.org/3/library/pickle.html>