



SPURTHY COLLEGE OF SCIENCE AND MANAGEMENT STUDIES

(Affiliated to Bangalore University)

**#328, Marsur Gate, Anekal Main Road, Marsur Post, Anekal
Taluk, Bangalore – 562106**

A Project Report on

REAL ESTATE PRICE PREDICTION

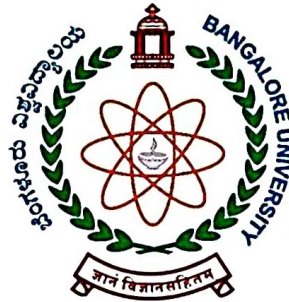
Submitted in partial fulfilment of requirement for the award of the degree

BACHELOR OF COMPUTER APPLICATIONS

For the Academic year 2023-2024

Of

Bangalore University



BY

ADHIL N A

SHAMEEM M A

U03KW21S0042

U03KW21S0058

Under the Guidance of
Mrs. Roja K
Assistant Professor, Department of BCA



SPURTHY COLLEGE OF SCIENCE AND MANAGEMENT STUDIES
(Affiliated to Bangalore University)
#328, Marsur Gate, Anekal Main Road, Marsur Post, Anekal
Taluk, Bangalore – 562106

BONAFIDE CERTIFICATE

This is to certify that the project entitled "**REAL ESTATE PRICE PREDICTION**" Is a bonafide work carried out By **Adhil N A (U03KW21S0042)** and **Shameem M A (U03KW21S0058)** who carried out the project work under my supervision, as part of her sixth semester project. Submitted in partial fulfilment of the requirement of sixth semester Course work of BCA during the academic session 2023-2024.

Mrs. Roja K
Assistant professor

**Spurthy College of Science and
Management Studies**

Mrs. K Sandhya
HOD of BCA department
**Spurthy College of Science and
Management Studies**

Mr. PRAKASH V
Principal
**Spurthy College of Science And
Management Studies**



SPURTHY COLLEGE OF SCIENCE AND MANAGEMENT STUDIES

(Affiliated to Bangalore University)

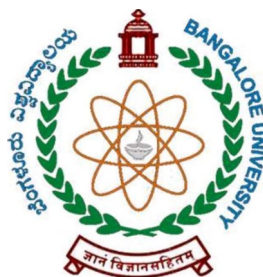
**#328, Marsur Gate, Anekal Main Road, Marsur Post, Anekal
Taluk, Bangalore— 562106**

GUIDE CERTIFICATE

This is to certify that the project report certified "**Real Estate Price Prediction**" under the guidance of **Mrs.Roja K ,Assistant Professor**. Submitted in the partial fulfilment of the requirement for Bachelor of Computer Applications.

To

Bangalore University



BY

ADHIL N A

U03KW21S0042

SHAMEEM M A

U03KW21S0058

Has worked under my guidance and that no part of this report has been submitted for the reward of any other degree and has not been published in any journal or magazine.

Date:

Place:

Name of the faculty: **Roja K**

Signature
(Academic Guide)



SPURTHY COLLEGE OF SCIENCE AND MANAGEMENT STUDIES

(Affiliated to Bangalore University)

**#328, Marsur Gate, Anekal Main Road, Marsur Post, Anekal
Taluk, Bangalore — 562106**

CERTIFICATE OF EVALUATION

**COLLEGE NAME : SPURTHY COLLEGE OF SCIENCE AND
MANAGEMENT STUDIES.**

BRANCH : BACHELOR OF COMPUTER APPLICATIONS.

NAME OF STUDENTS : ADHIL NA & SHAMEEM MA

PROJECT TITLE : REAL ESTATE PRICE PREDICTION

This project work report submitted in partial fulfillment for the award of BCA degree in BANGALORE UNIVERSITY, BANGALORE, is evaluated and confirmed to be report of the work done by the above student.

Submission for the project viva voice held on

INTERNAL EXAMINER

EXTERNAL EXAMINER



SPURTHY COLLEGE OF SCIENCE AND MANAGEMENT STUDIES

(Affiliated to Bangalore University)

**#328, Marsur Gate, Anekal Main Road, Marsur Post, Anekal
Taluk, Bangalore — 562106**

DECLARATION

I hereby declare that the project work entitled " **REAL ESTATE PRICE PREDICTION** " is a genuine work carried out by me under guidance of **Mrs. Roja K**, of the department of BCA, in partial fulfillment for the award of the degree of "**Bachelor of Computer Applications**" of Bangalore University, Bangalore.

ADHIL N A
U03KW21S0042

SHAMEEM M A
U03KW21S0058

TABLE OF CONTENTS

1. INTRODUCTION	4
1.1 BACKGROUND	4
1.2 GOAL AND OBJECTIVE	7
1.3 PROPOSED PROBLEM AND SOLUTION	9-11
2. METHODOLOGY	12
2.1 SYSTEM REQUIREMENTS	13
2.1.1 HARDWARE REQUIREMENTS	13
2.1.2 SOFTWARE REQUIREMENTS	
2.1.3 EXISTING SYSTEM	13
2.1.4 LANGUAGES USED	14
2.2 CSV DATASET PROCESSING METHOD	19
2.2.1 IMPLEMENTATION	19
2.3 MACHINE LEARNING	21
2.3.1 ALGORITHMS	33
2.4 PSEUDOCODE	37
3. RESULTS	40
3.1 DATASET	40
3.2 CSV DATASET PROCESSING	41
3.2.1 PREPROCESSING TECHNIQUES	41
3.2.2 FEATURE EXTRACTION	52
3.2.3 MACHINE LEARNING TECHNIQUES	54
4. APPENDICES	56
4.1 SOURCE CODE	56
5. SUMMARY AND CONCLUSION	72
6. SNAPSHOTS	74
7 CONCLUSION	75
8. BIBLIOGRAPHY	76

LIST OF CHARTS

Chart no:	Particular	Page. No:
01	CSV Dataset Processing method	19
02	Feature extraction	52

LIST OF IMAGES

Image no	Particular	Page no
01	PSUEDOCODE	37
02	RESULT	40
03	FEATURE EXTRACTION	52
04	MACHINE LEARNING TECHNIQUES	54
05	APPENDIX	56

LIST OF ABBREVIATIONS

API	Application Programme Interface
APP	Application
FLRS	FLRS (Fault Location, Isolation, and Service Restoration)
ANN	Artificial Neural Network
FIS	Fuzzy Inference System
CSS	Cascading Style Sheet
CSV	Comma Separated Values
GUI	Graphical User Interface
HTML	Hypertext Markup Language
JS	Java-script
ML	Machine Learning
UI	User Interface

1. INTRODUCTION

1.1 BACKGROUND

The traditional approach of setting sales and marketing goals has become increasingly inadequate for companies trying to keep pace with the competitive market. Historically, these goals were often formulated without deep insights into customer purchasing patterns, leading to a gap between the strategies employed and the actual market dynamics. This disconnect is becoming more pronounced as consumer behaviors evolve rapidly and markets become more saturated. In this context, companies are finding it challenging to maintain their competitive edge using conventional methods.

One of the significant transformations in the sales and marketing domain is the advent of Machine Learning (ML). ML advancements have revolutionized how businesses approach their markets, providing tools to analyze vast amounts of data to extract meaningful insights. These insights are pivotal in understanding consumer behavior, predicting future trends, and tailoring marketing strategies accordingly. For instance, ML can help identify subtle patterns in purchasing behavior that might be invisible to human analysts, allowing companies to anticipate customer needs more accurately and personalize their offerings.

Among the critical aspects where ML has made a substantial impact are consumers' purchase patterns, target audience identification, and sales forecasting. By analyzing historical data, ML models can predict future sales trends, helping businesses plan more effectively. For example, retail companies use ML algorithms to forecast product demand during different seasons, ensuring optimal inventory levels and reducing the risk of overstocking or stockouts. Similarly, in the real estate sector, understanding and predicting house prices have become more sophisticated with ML, aiding buyers, sellers, and investors in making informed decisions.

This paper focuses on predicting the prices of real estate properties, specifically houses, in India using various Machine Learning algorithms. The real estate market in India is vast and diverse, with prices influenced by numerous factors beyond just the size of the property. These factors include location, proximity to amenities, socio-economic conditions, and market trends, among others. Traditionally, real estate pricing relied heavily on human judgment and simple statistical methods, which could not adequately capture the complex interplay of these factors.

In recent years, the application of ML in real estate has shown promising results. By leveraging algorithms such as Linear Regression and Random Forest Regression, we can analyze large datasets to uncover the underlying factors that influence property prices. Linear Regression helps in understanding the linear relationship between dependent and independent variables, making it suitable for initial price predictions. On the other hand, Random Forest Regression, a more sophisticated ensemble learning method, can handle non-linear relationships and interactions between variables, often resulting in more accurate predictions.

The significance of accurate real estate price predictions cannot be overstated. Real estate is not only a basic necessity but also a significant investment representing wealth and prestige. Accurate price predictions benefit various stakeholders, including household investors, bankers, policymakers, and real estate developers. Investors rely on these predictions to make informed decisions, minimizing risks and maximizing returns. Banks and financial institutions use property valuations to assess mortgage risks and manage their lending portfolios. Policymakers can utilize these insights to implement housing policies that promote affordability and economic stability.

Moreover, sales forecasting has always been a critical area for business organizations. Accurate forecasts enable companies to manage their resources efficiently, plan their production schedules, and devise effective marketing strategies. For instance, an accurate forecast of housing demand can help construction companies plan their projects better, ensuring timely delivery and cost management. Similarly, real estate agencies can optimize their marketing efforts by targeting potential buyers more effectively, reducing marketing costs, and increasing conversion rates.

In contrast, manual predictions of house prices are prone to errors and inefficiencies. Human analysts may overlook critical factors or fail to identify emerging trends, leading to inaccurate forecasts. This can result in poor decision-making, financial losses, and missed opportunities. Additionally, manual analysis is time-consuming, which is undesirable in today's fast-paced world where timely decisions are crucial for maintaining a competitive edge.

The global economy heavily depends on the business sector's ability to meet market demands. Sales forecasting plays a crucial role in this regard by predicting future demand for products or services. For example, in the context of real estate, forecasting the demand for housing in specific regions helps developers allocate resources and plan construction projects more effectively. It also assists policymakers in understanding housing needs and implementing appropriate measures to ensure housing availability and affordability.

In this paper, we aim to predict the price of houses in India using three different Machine Learning algorithms: Linear Regression, Random Forest Regression, and another suitable algorithm. By comparing their performance, we seek to determine which algorithm provides the most accurate predictions based on key features extracted from the raw data. This comprehensive study involves not only the application of these algorithms but also a detailed analysis and exploration of the collected data to gain a complete understanding of the factors influencing house prices.

Accurately predicting house prices is a complex task due to the myriad of factors involved. However, with the advancements in ML, we can handle this complexity more effectively. The insights gained from this analysis will help business organizations make probabilistic decisions at each stage of their marketing strategy, enhancing their ability to respond to market dynamics and achieve their business objectives.

In conclusion, the integration of Machine Learning in sales and marketing represents a significant shift from traditional approaches. By leveraging advanced algorithms to analyze consumer behavior and market trends, businesses can develop more effective strategies, improve their decision-making processes, and ultimately gain a competitive edge in the market. The application of ML in predicting real estate prices in India is a testament to the potential of these technologies to transform industries and drive growth in the modern economy.

Key Words: Random Forest Regression, Linear Regression, Algorithms, Real Estate, Price Prediction, Data mining, Machine Learning.

1.2 GOAL AND OBJECTIVE

The main aim of the project is to help people who plan to buy a house so they can know the price range in the future, then they can plan their finances well.

- To help all the users that are trying to find out the cost of any real estate property.
- To predict the market value of a real estate property.
- Help to find a starting price for a property based on the geographical variables.
- Anticipate the future costs by breaking down past market patterns and value ranges, and coming advancements.
- To make it user friendly and free of cost for the users.

- **Enhance Accuracy in Real Estate Pricing:**

- Leverage Machine Learning algorithms to improve the precision of real estate price predictions in India, surpassing traditional estimation methods.

- **Facilitate Informed Decision-Making:**

- Provide stakeholders (investors, developers, banks, policymakers) with reliable data-driven insights to make informed decisions regarding real estate investments, lending, and policy formulation.

- **Optimize Resource Allocation:**

- Assist real estate developers and agencies in planning and allocating resources efficiently by predicting housing demand and market trends.

- **Promote Technological Adoption:**

- Encourage the adoption of advanced Machine Learning techniques in the real estate sector, showcasing their potential benefits and applications.

- **Contribute to Economic Stability:**

- Support the stability and growth of the real estate market by providing accurate pricing forecasts, thereby fostering investor confidence and market efficiency

- **Data Collection and Preparation:**

- Gather extensive data from various property aggregators across India, including key features such as location, size, amenities, and market conditions.
- Clean, preprocess, and analyze the collected data to ensure it is suitable for Machine Learning modeling.
- **Algorithm Implementation:**
 - Implement three different Machine Learning algorithms (e.g., Linear Regression, Random Forest Regression, and another suitable model) to predict real estate prices.
 - Train and test these models on the prepared dataset to evaluate their performance.
- **Performance Comparison:**
 - Compare the accuracy and reliability of the different Machine Learning models to determine which algorithm provides the most accurate price predictions.
 - Use metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared to assess model performance.
- **Insight Generation:**
 - Analyze the results to identify key factors influencing real estate prices in India.
 - Provide actionable insights based on the model outputs to help stakeholders understand market dynamics and make informed decisions.
- **Visualization and Reporting:**

- Create visualizations (graphs, charts, and maps) to represent the data analysis and prediction results clearly and effectively.
- Compile a comprehensive report detailing the methodology, findings, and recommendations for future applications and research.

- **Scalability and Future Research:**

- Explore the scalability of the implemented models to other regions and types of real estate properties.
- Identify areas for further research and potential improvements in predictive modeling techniques for real estate pricing.

1.3 PROPOSED PROBLEM AND SOLUTION

1.3.1 PROPOSED PROBLEM

Housing prices in any city are an important reflection of the economy, and housing price ranges are of great interest for both buyers and sellers. The fluctuations and trends in housing prices can provide critical insights into the economic conditions of a region, influencing decisions made by individual buyers, real estate investors, and policymakers alike.

Ask any home buyer to describe their dream house, and they probably won't begin with the height of the basement ceiling or the proximity to an east-west railroad. Most buyers envision their ideal home by thinking of features such as the number of bedrooms, the presence of a garden or a white-picket fence, and the overall aesthetic appeal of the property. However, in reality, there is much more that influences price negotiations than these superficial characteristics.

Factors such as the location of the property, proximity to essential services like schools and hospitals, the quality of the local infrastructure, and the economic status of the neighborhood play a crucial role in determining the price of a house. Moreover, market dynamics, interest rates, and broader economic indicators significantly impact real estate prices. Prices of real estate properties are sophisticatedly linked with our economy, as they reflect not just the value of the physical property, but also the economic health and growth potential of the area.

Despite all of this, we do not have accurate measures of housing prices based on the vast amount of data available. The complexity and variability of the factors involved make it challenging to derive precise estimates. Traditional methods of estimating housing prices often fall short in capturing the intricate relationships between various economic indicators and property values. This gap highlights the need for more advanced analytical tools and techniques.

Simulation results show that the Fuzzy Logic System Regression (FLSR) provides a superior prediction function compared to Artificial Neural Networks (ANN) and Fuzzy Inference Systems (FIS) in capturing the functional relationship between dependent and independent variables and has the lowest computational complexity. FLSR's ability to model non-linear relationships and handle uncertainty makes it a powerful tool for real estate price prediction. Its comparative advantage lies in its efficiency and accuracy, which are crucial for making reliable forecasts in a data-rich environment.

Therefore, the goal of this project is to use machine learning algorithms to predict the selling prices of houses based on many economic factors. By leveraging advanced computational techniques, we aim to develop models that can analyze large datasets, identify patterns, and provide accurate price predictions. These predictions will help various stakeholders make informed decisions, whether they are buying a home, investing in real estate, or formulating economic policies. This project will explore the use of different machine learning approaches, compare their performance, and determine the most effective method for predicting housing prices in diverse economic conditions.

1.3.2 SOLUTION

Our approach to predicting real estate prices in India using machine learning involves several key steps. The process begins with meticulously collecting and preprocessing data from various property aggregators across the country. This step is crucial as it ensures that we have a comprehensive and clean dataset that is ready for detailed analysis. The data collection phase involves aggregating a wide range of information, including property details, geographical data, and economic indicators. Preprocessing includes cleaning the data to remove any inconsistencies or errors, handling missing values, and standardizing the data formats to ensure uniformity.

Next, we focus on feature selection and engineering, which is a critical step in the machine learning pipeline. During this phase, we identify and select key factors that significantly influence house prices. These factors include, but are not limited to, the location of the property, the presence and quality of amenities (such as schools, hospitals, and shopping centers), and various economic indicators (such as interest rates, employment rates, and market trends). Feature engineering may also involve creating new features from existing data to better capture the underlying patterns and relationships.

With the dataset prepared and the features selected, we then experiment with different machine learning algorithms. Initially, we employ algorithms like Linear Regression and Random Forest Regression to train models on the data. Linear Regression is used to understand the linear relationships between the dependent variable (house prices) and the independent variables (features), providing a straightforward and interpretable model. On the other hand, Random Forest Regression, an ensemble learning method, is utilized to capture more complex, non-linear relationships in the data. By using these different algorithms, we can assess which model best fits the data and provides the most accurate predictions.

After training the models, we move on to the evaluation and comparison phase. This involves using various metrics to assess the performance of each model. Metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared are used to evaluate how well the models predict house prices. We also perform cross-validation to ensure the models generalize well to unseen data. Through thorough evaluation, we identify the most accurate and reliable model.

Once the most accurate model is selected, we deploy it into production. This deployment involves integrating the model into a real-world application where it can be used to provide predictions on new data. We ensure that the model is robust and can handle real-time data inputs. Additionally, we set up

mechanisms for continuous monitoring and updating of the model to adapt to changing market trends. This involves regularly retraining the model with new data to maintain its accuracy and relevance.

Through this comprehensive approach, we aim to provide stakeholders with reliable insights for informed decision-making in the dynamic real estate market. By leveraging machine learning, we can uncover hidden patterns and relationships in the data, enabling more accurate predictions of house prices. These insights can help buyers, sellers, investors, and policymakers make better decisions, ultimately contributing to a more efficient and transparent real estate market in India

2. METHODOLOGY

2.1 SYSTEM REQUIREMENTS

2.1.1 HARDWARE REQUIREMENTS

The hardware requirements for running the proposed solution depend on the scale of data and complexity of machine learning algorithms used. However, a basic setup would typically include:

- Processor: Ryzen 5 or equivalent
- Memory (RAM): 8 GB RAM or higher
- Storage: Sufficient storage space for storing datasets, models, and intermediate results.
- Graphics Processing Unit (GPU): Optional but recommended for accelerating model training •

Network: Ethernet/wifi for internet connectivity

2.1.2 SOFTWARE REQUIREMENTS

The software requirements for implementing the proposed solution are as follows:

- Operating System: windows 10 or ubuntu 20.04 LTS or higher.
- Web browser: Google Chrome or Brave etc.
- Integrated Development Environment (IDE): Visual Studio Code, python 3.8 or higher.

2.1.3 EXISTING SYSTEM

The existing system for real estate price prediction serves as a basic tool that provides rough estimates based on traditional statistical methods and domain knowledge. While it offers some functionalities that can help users make decisions, its limitations in data handling, model sophistication, and adaptability reduce its effectiveness in a dynamic and data-rich market. There is a significant opportunity to enhance these systems by integrating advanced machine learning techniques, improving data preprocessing and feature engineering, and developing more user-friendly and adaptive interfaces.

- **Price Estimation:**

User Input: Users input property details (e.g., size, location, number of bedrooms).

Prediction: The system provides a price estimate based on the input data and the pre-built model.

- **Comparative Market Analysis:**

Historical Data: Users can view historical price trends and averages for specific locations.

Current Listings: The system may offer comparisons with current listings to help users gauge market conditions.

- **Report Generation:**

Summarized Data: The system can generate reports summarizing price trends and predictions for given areas.

Limitations

1. **Data Quality and Volume**
2. **Sophistication of Models**
3. **Feature Engineering**
4. **Model Updating and Adaptability**
5. **User Interaction and Accessibility**

2.1.4 LANGUAGES USED

- Front-end: HTML, CSS, Javascript
- Back-end: python, flask

1) HTML



HyperText Markup Language (HTML) is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript. HTML is the abbreviation for Hyper Text Markup Language. It's a markup language and is used to construct web pages. HTML is a markup language that integrates hypertext & markup. The term "hypertext" refers to the connection between online pages. The text document within the tags that specifies the structure of web pages is produced using a markup language. This language is used to annotate (add notes to) text so that a computer can comprehend it and alter it appropriately. The majority of markup languages (such as HTML) are understandable by humans. Tags are used in the language to specify what type of text processing is required

2)CSS



Cascading Style Sheets (CSS) is a style sheet language used for specifying the presentation and styling of a document written in a markup language such as HTML or XML (including XML dialects such as SVG, MathML or XHTML). CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS (Cascading Style Sheets) is a language designed to simplify the process of making web pages presentable. It allows you to apply styles to HTML documents, describing how a webpage should look by prescribing colors, fonts, spacing, and positioning. CSS provides developers and designers with powerful control over the presentation of HTML elements.

HTML uses tags and CSS uses rulesets. CSS styles are applied to the HTML element using selectors. CSS is easy to learn and understand, but it provides powerful control over the presentation of an HTML document.

Why CSS?

Saves Time: Write CSS once and reuse it across multiple HTML pages.

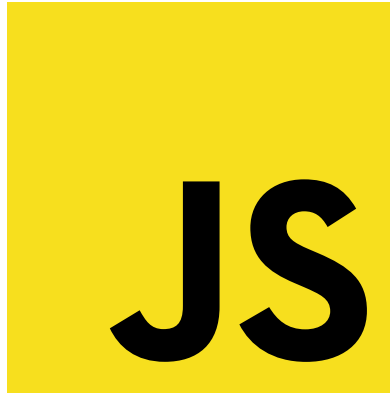
Easy Maintenance: Change the style globally with a single modification.

Search Engine Friendly: Clean coding technique that improves readability for search engines.

Superior Styles: Offers a wider array of attributes compared to HTML.

Offline Browsing: CSS can store web applications locally using offline cache, allowing offline viewing.

3)JAVASCRIPT



JavaScript is a high-level programming language that follows the ECMAScript standard. It was originally designed as a scripting language for websites but became widely adopted as a general-purpose programming language, and is currently the most popular programming language in use. JavaScript is usually found running in a web browser as interactive or automated content, ranging from popup messages and live clocks to large web applications. JavaScript is also commonly used in server-side programming through platforms like Node.js, or "embedded" in non-JavaScript applications where the base programming language lacks the high-level functionality that JavaScript offers.

JavaScript was designed to add interactivity to HTML pages

- ☐ JavaScript is a scripting language
- ☐ A scripting language is a lightweight programming language
- ☐ JavaScript is usually embedded directly into HTML pages
- ☐ JavaScript is an interpreted language (means that scripts execute without preliminary compilation)
- ☐ Everyone can use JavaScript without purchasing a license

4)PYTHON



Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

Python consistently ranks as one of the most popular programming languages, and has gained widespread use in the machine learning community

Python, with its comprehensive suite of libraries and tools, is well-suited for the backend of a real estate price prediction project. It allows for efficient data collection, preprocessing, feature selection, model training, evaluation, and deployment. By leveraging Python's capabilities, the project can deliver accurate and reliable real estate price predictions, providing valuable insights for stakeholders in the dynamic real estate market.

Python is an excellent choice for the backend of a real estate price prediction project due to its extensive libraries, ease of use, and strong community support. Below, we will explore in detail how Python can be utilized effectively in each stage of the project, from data collection to model deployment.

1. Data Collection

Python provides several libraries to facilitate data collection from various sources

2. Data Preprocessing

Python's pandas library is a powerful tool for data manipulation and preprocessing

3. Feature Selection and Engineering

Using libraries like scikit-learn, Python facilitates feature selection through techniques like correlation matrices, feature importance, and recursive feature elimination.

5. Model Deployment

For deploying the machine learning model into a production environment, Python offers several frameworks

5) FLASK



Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

Benefits of Using Flask

1. **Simplicity:** Flask's minimalistic nature allows for a simple and clean setup, making it easy to build and maintain.
2. **Flexibility:** Flask's modularity allows developers to choose their components, offering flexibility in terms of database connectors, ORM, and other integrations.
3. **Ease of Integration:** Flask can be easily integrated with various machine learning libraries and tools, facilitating the deployment of ML models.
4. **Scalability:** Although lightweight, Flask can be scaled for larger applications by integrating with other tools and frameworks.
5. **Community and Documentation:** Flask has extensive documentation and a strong community, making it easy to find resources and support.

2.2 CSV DATASET PROCESSING METHODS

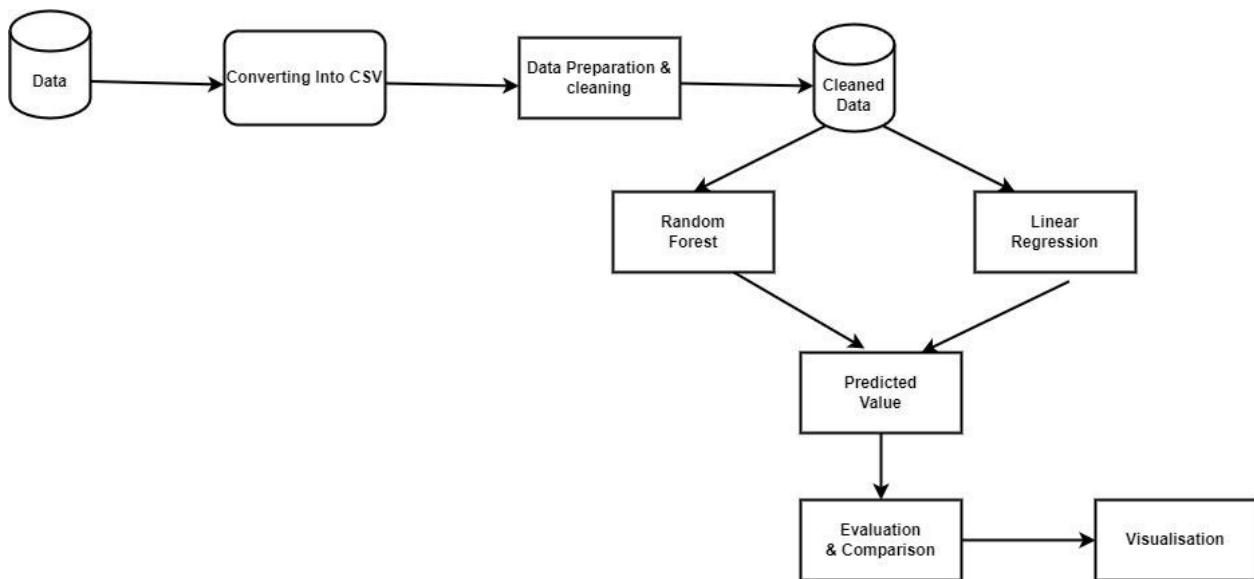


Fig:1 System Architecture

The software development cycle for the real estate price prediction project ensures a structured approach to building a robust and scalable application. By following these stages—requirement analysis, system design, implementation, testing, deployment, and maintenance—the project can effectively deliver accurate and reliable price predictions, meeting the needs of stakeholders in the dynamic real estate market.

2.2.1 IMPLEMENTATION

- The implementation process of the project begins with importing the dataset, which is then meticulously preprocessed to ensure it is ready for analysis. Data preprocessing is a critical step that involves several important tasks. First, the data is converted into an understandable and usable form, which may include parsing dates, normalizing numerical values, and encoding categorical variables. This transformation ensures that the data is in a consistent format suitable for machine learning algorithms.
- Next, the data cleaning process is undertaken. This involves identifying and handling missing values, which could be filled in with mean, median, or mode values, or through more advanced imputation techniques. Additionally, inconsistencies within the data are corrected to maintain uniformity and reliability. During this stage, we also detect and eliminate outliers. Outliers

are data points that exhibit abnormal behavior compared to the rest of the dataset. They can significantly skew the results of the analysis if not addressed properly. Techniques such as z-score, IQR (Interquartile Range), or visualization methods are employed to identify these outliers, which are then either removed or adjusted to ensure the data's integrity.

- With the cleaned dataset, we proceed to apply the regression techniques chosen for this project: Random Forest and Linear Regression. These algorithms are used to train models on the preprocessed data. Linear Regression, a simple yet powerful algorithm, helps in understanding the linear relationships between the dependent and independent variables. Random Forest Regression, on the other hand, is an ensemble learning method that builds multiple decision trees and merges them to get a more accurate and stable prediction. This method is particularly useful for capturing complex, non-linear relationships in the data.
- After training the models using these regression techniques, we obtain outputs from each algorithm. These outputs represent the predicted house prices based on the respective models. To determine which model performs the best, we compare the outputs using various evaluation metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared. These metrics help in assessing the accuracy and reliability of the predictions made by each model.
- The comparison of the models involves analyzing their performance on a validation dataset to ensure they generalize well to new, unseen data. Cross-validation techniques might also be employed to validate the robustness of the models further. Based on this thorough evaluation, the model that demonstrates the highest accuracy and reliability is selected.
- Finally, the most accurate algorithm is chosen for the project. This selected model is then deployed for practical use, providing stakeholders with reliable predictions for informed decision-making in the real estate market. The implementation process does not end here; continuous monitoring and updating of the model are essential to adapt to new data and changing market conditions, ensuring sustained accuracy and relevance of the predictions.

- **Data Splitting:**You split your dataset into training and testing sets using `train_test_split()` method from `sklearn.model_selection`.
- **Linear Regression Training:**You instantiated and trained a Linear Regression model using `LinearRegression()` from `sklearn.linear_model`. Saved the trained model using `pickle.dump()`.
- **Making Predictions with Linear Regression:**Used the trained Linear Regression model to make predictions on the test set (`X_test`).

- **Ridge Regression Training:** You instantiated and trained a Ridge Regression model using `Ridge()` from `sklearn.linear_model`.
- **Cross-Validation:** Utilized cross-validation with `ShuffleSplit` and `cross_val_score()` from `sklearn.model_selection` to assess the performance of the Linear Regression model.
- **XGBoost Model Training:** You imported and trained an XGBoost model using `rgb.XGBRegressor()` from `xgboost`.
- **Saving XGBoost Model:** Saved the trained XGBoost model using `pickle.dump()`.
- **Building Predictive System:** Defined a function `predict_price()` to make predictions using the trained XGBoost model. Reshaped input data into a numpy array and used the XGBoost model to make predictions.
- **Predicting Prices:** Provided sample input data (`input_data_1` and `input_data_2`) and obtained predictions using the `predict_price()` function. Printed the predictions for both sets of input data.

2.3 MACHINE LEARNING

Machine learning is a branch of artificial intelligence (AI) that focuses on the development of algorithms and statistical models that enable computers to learn and make predictions or decisions without being explicitly programmed. It involves the use of data to train models, allowing them to identify patterns, extract insights, and make predictions or decisions based on new data. Machine learning algorithms are categorized into supervised, unsupervised, and reinforcement learning, each serving different purposes in tasks such as classification, regression, clustering, and reinforcement learning.

Machine learning (ML) is a discipline of artificial intelligence (AI) that provides machines with the ability to automatically learn from data and past experiences while identifying patterns to make predictions with minimal human intervention.

Machine learning methods enable computers to operate autonomously without explicit programming. ML applications are fed with new data, and they can independently learn, grow, develop, and adapt.

Machine learning derives insightful information from large volumes of data by leveraging algorithms to identify patterns and learn in an iterative process. ML algorithms use computation methods to learn directly from data instead of relying on any predetermined equation that may serve as a model.

The performance of ML algorithms adaptively improves with an increase in the number of available samples during the 'learning' processes. For example deep learning is a sub-domain of machine learning that trains computers to imitate natural human traits like learning from examples. It offers better performance parameters than conventional ML algorithms.

While machine learning is not a new concept – dating back to World War II when the Enigma Machine was used – the ability to apply complex mathematical calculations automatically to growing volumes and varieties of available data is a relatively recent development.

Importance of Machine Learning in Real Estate Price Prediction

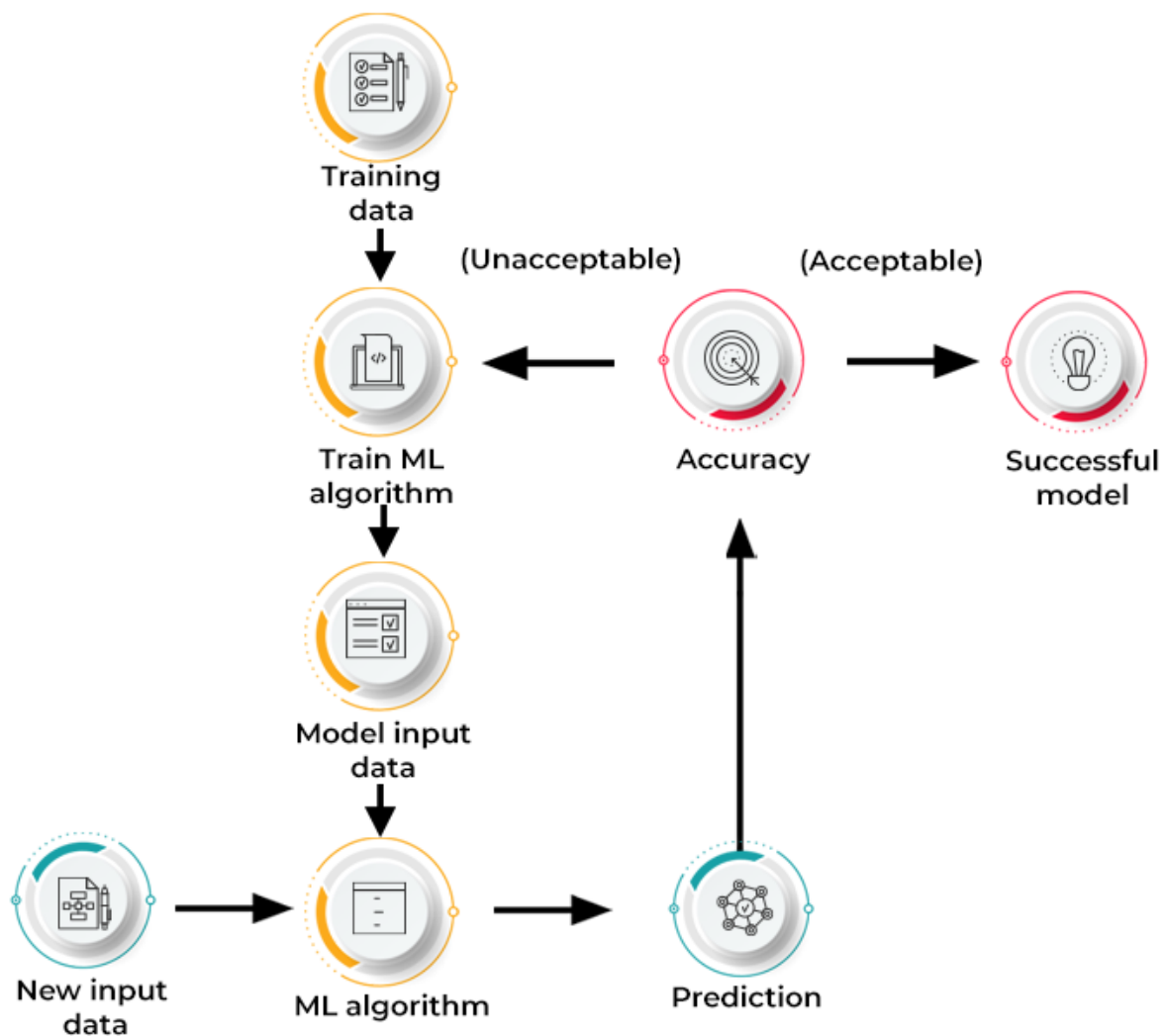
1. **Accuracy:** ML models can handle complex and non-linear relationships between various factors affecting house prices, leading to more accurate predictions.
2. **Efficiency:** Automating the prediction process saves time and resources compared to manual analysis.
3. **Scalability:** ML models can process large datasets quickly, making it feasible to predict prices for numerous properties simultaneously.
4. **Insights:** ML can uncover hidden patterns and correlations in the data, providing valuable insights into the real estate market.

Key Steps in Applying Machine Learning to Real Estate Price Prediction

1. **Data Collection:** Gather data from various sources, such as property aggregators, government databases, and real estate websites. The data typically includes features like location, size, number of bedrooms, age of the property, and other amenities.
2. **Data Preprocessing:** Clean the data to handle missing values, remove outliers, and normalize features. This step ensures the quality and consistency of the data, which is crucial for building accurate models.
3. **Feature Engineering:** Create new features that better represent the underlying patterns in the data. This might include calculating the price per square foot, categorizing properties based on their age, or encoding categorical variables.
4. **Model Selection:** Choose appropriate machine learning algorithms for the prediction task. Common algorithms used in real estate price prediction include:
 - **Linear Regression:** A simple and interpretable model that assumes a linear relationship between the features and the target variable (price).
 - **Random Forest Regression:** An ensemble learning method that builds multiple decision trees and combines their predictions for improved accuracy and robustness.
 - **Gradient Boosting Machines (GBM):** An advanced ensemble technique that builds trees sequentially, each one correcting the errors of its predecessor.
 - **Support Vector Regression (SVR):** A model that uses support vector machines to perform regression tasks, effective for high-dimensional datasets.

5. **Model Training:** Split the data into training and testing sets. Train the selected models on the training set, ensuring they learn the patterns in the data.
6. **Model Evaluation:** Evaluate the models' performance using metrics like Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared (R^2). This step helps in comparing the models and selecting the best one based on their accuracy.
7. **Model Deployment:** Deploy the best-performing model into a production environment where it can make real-time predictions for new data. This involves integrating the model into a web application or API using frameworks like Flask.
8. **Monitoring and Maintenance:** Continuously monitor the model's performance and update it as new data becomes available. This ensures that the model remains accurate and relevant over time.

HOW DOES MACHINE LEARNING WORK?



Objectives, consideration and goal of the project :

1. Forecast Accuracy:

- Objective: Develop machine learning models that accurately predict future prices.
- Considerations: Aim for high accuracy metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), or percentage error.
- Goals: Achieve a prediction accuracy that meets or exceeds industry standards or benchmarks. Continuously refine models to improve accuracy over time.

2. Model Comparison:

- Objective: Evaluate and compare the performance of different predictive models.
- Considerations: Compare regression models (e.g., linear regression, random forest, neural networks) based on metrics like accuracy, interpretability, and computational efficiency.
- Goals: Identify the model(s) that provide the most accurate and reliable predictions for your specific price prediction problem.

3. Feature Importance:

- Objective: Determine which features (variables) have the most significant impact on price predictions.
- Considerations: Use feature selection techniques (e.g., correlation analysis, feature importance from tree-based models) to identify critical factors influencing prices.
- Goals: Understand the drivers of price changes and potentially refine models by focusing on the most impactful features.

4. Real-time Prediction:

- Objective: Develop models capable of making price predictions in real-time or near-real-time.
- Considerations: Implement algorithms that can process incoming data quickly and update predictions as new information becomes available.
- Goals: Enable timely decision-making based on the latest market conditions or data updates.

5. Scalability:

- Objective: Ensure that the prediction models can handle large volumes of data efficiently.
- Considerations: Optimize model performance and computational efficiency to scale with increasing data sizes or transaction volumes.
- Goals: Maintain prediction accuracy while accommodating growth in data volume and processing demands.

6. User Interface (UI):

- Objective: Design an intuitive interface for stakeholders to interact with and understand price predictions.
- Considerations: Incorporate visualization tools, dashboards, and user-friendly features to present predictions and insights effectively.
- Goals: Facilitate easy interpretation of predictions by non-technical stakeholders, enhancing usability and decision-making.

7. Risk Assessment:

- Objective: Evaluate and mitigate risks associated with price predictions.
- Considerations: Identify potential sources of error or bias in predictions and implement measures to mitigate these risks.

- Goals: Provide stakeholders with transparent assessments of prediction reliability and potential uncertainties.

8. Documentation and Reporting:

- Objective: Provide comprehensive documentation and regular reports on prediction model performance.

- Considerations: Document model development processes, validation techniques, and performance metrics.

- Goals: Enable stakeholders to review and understand prediction results, fostering trust and transparency in the prediction process.

9. Integration:

- Objective: Integrate prediction models into existing systems or workflows.

- Considerations: Ensure compatibility with existing software and data infrastructure.

- Goals: Seamlessly incorporate price predictions into decision-making processes and operational workflows.

10. Continuous Improvement:

- Objective: Establish mechanisms for ongoing model refinement and improvement.

- Considerations: Monitor prediction performance over time and update models with new data and improved techniques.

- Goals: Continuously enhance prediction accuracy and relevance to adapt to changing market conditions or business requirements.

Each objective should align with the overall project goals and contribute to the successful implementation and utilization of price prediction models within your organization or project scope.

Scope of the Real Estate Price Prediction Project :

1. *Objective*:

- Develop predictive models to estimate future property prices accurately based on historical data and relevant factors.

2. *Target Audience*:

- Real estate investors, developers, agents, and financial institutions seeking reliable forecasts to make informed decisions regarding property investments, pricing strategies, or portfolio management.

3. *Geographic Focus*:

- Define the geographic area(s) of interest, such as specific cities, neighborhoods, or regions where price predictions will be applicable.

4. *Property Types*:

- Specify the types of properties covered by the prediction models (e.g., residential homes, commercial real estate, luxury properties) based on stakeholder needs and market dynamics.

5. *Data Sources*:

- Collect and integrate diverse datasets including historical sales prices, property features (e.g., size, location, amenities), economic indicators (e.g., interest rates, GDP growth), demographic trends, and local market conditions.

6. *Methodology*:

- Utilize advanced statistical methods and machine learning algorithms tailored for real estate forecasting, such as hedonic pricing models, spatial analysis, time series forecasting (e.g., ARIMA, VAR), or ensemble methods (e.g., random forest, gradient boosting).

7. *Feature Engineering*:

- Identify and preprocess relevant features that significantly impact property prices, including quantitative variables (e.g., square footage, number of bedrooms) and qualitative factors (e.g., neighborhood quality, proximity to amenities).

8. *Model Validation*:

- Validate models through rigorous testing procedures, including cross-validation, out-of-sample testing, and sensitivity analysis, to ensure robustness and reliability of predictions.

9. *Performance Metrics*:

- Evaluate prediction accuracy using metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), or coefficient of determination (R-squared) against actual sale prices to quantify model effectiveness.

10. *Interpretability and Transparency*:

- Ensure models provide interpretable insights into the factors influencing property price predictions, enabling stakeholders to understand and trust the forecasting outcomes.

11. ***Implementation*:**

- Integrate prediction models into decision support systems, real estate platforms, or investment tools to facilitate access and utilization of predictions by stakeholders.

12. ***Risk Management*:**

- Address risks associated with predictions, such as model uncertainties, market volatility, regulatory changes, and external economic factors impacting real estate markets.

13. ***Documentation and Reporting*:**

- Provide comprehensive documentation of model development processes, validation results, performance metrics, and assumptions to stakeholders. Regularly update and report on prediction accuracy and model refinements.

14. ***Ethical Considerations*:**

- Address ethical implications related to data privacy, fairness in predictions (e.g., avoiding bias in model outcomes), and potential impacts of prediction decisions on diverse stakeholders.

15. ***Constraints and Limitations*:**

- Acknowledge limitations such as data availability, quality, and timeliness; model assumptions; and uncertainties inherent in predicting real estate prices, ensuring stakeholders understand the scope of prediction reliability.

16. ***Scalability*:**

- Design models that can scale with increasing data volumes, market complexity, and computational demands to maintain effectiveness and relevance over time.

17. Continuous Improvement:

- Establish mechanisms for ongoing model enhancement, incorporating feedback, new data sources, and advances in prediction techniques to improve accuracy and adaptability to changing real estate market dynamics.

Here are key risks and corresponding mitigation approaches:

Risks in Real Estate Price Prediction Project :

1. *Data Quality Issues*:

- *Risk*: Inaccurate or incomplete data can lead to biased models and unreliable predictions.
- *Mitigation*:
 - Conduct thorough data validation and cleaning processes.
 - Use multiple data sources to enhance data quality and completeness.
 - Implement data quality checks and corrections throughout the project lifecycle.

2. *Model Overfitting*:

- *Risk*: Models may become overly complex and fit too closely to training data, leading to poor generalization on new data.
- *Mitigation*:

- Use regularization techniques (e.g., Lasso, Ridge) to penalize overly complex models.
- Employ cross-validation to assess model performance on unseen data.
- Simplify models and feature selection to reduce overfitting risks.

3. *Market Volatility and External Factors*:

- *Risk*: Real estate markets are influenced by economic fluctuations, regulatory changes, and geopolitical events, impacting price trends unpredictably.
- *Mitigation*:
 - Incorporate economic indicators (e.g., GDP growth, interest rates) and market sentiment analysis into predictive models.
 - Monitor external factors and update models accordingly to reflect changing market conditions.
 - Use ensemble methods or scenario analysis to account for uncertainty in external factors.

4. *Model Assumptions and Biases*:

- *Risk*: Models may be based on assumptions that do not hold over time or contain inherent biases (e.g., sampling bias).
- *Mitigation*:
 - Document and validate assumptions underlying the predictive models.
 - Regularly review and update models with new data and improved methodologies.
 - Conduct sensitivity analysis to assess the impact of different assumptions on model outputs.

5. *Lack of Transparency and Interpretability*:

- ***Risk***: Stakeholders may struggle to understand how predictions are generated, leading to mistrust or misinterpretation of results.

- ***Mitigation***:

- Use interpretable models whenever possible (e.g., linear regression, decision trees).

- Provide clear explanations of model inputs, outputs, and underlying methodologies.

- Use visualization techniques to enhance understanding of prediction factors and model insights.

6. ***Ethical Considerations***:

- ***Risk***: Predictions may inadvertently reinforce or exacerbate social inequalities, especially in diverse or underserved communities.

- ***Mitigation***:

- Implement fairness-aware techniques to detect and mitigate biases in predictions.

- Ensure compliance with ethical guidelines and regulations (e.g., fairness metrics, bias detection algorithms).

- Consider the broader societal impacts of prediction outcomes and adjust models accordingly.

7. ***Operational Risks***:

- ***Risk***: Technical failures, system downtime, or integration issues may disrupt prediction delivery or decision-making processes.

- ***Mitigation***:

- Establish robust backup and contingency plans for system failures.

- Conduct regular maintenance and testing of predictive models and infrastructure.

- Provide technical support and training to stakeholders on using prediction outputs effectively.

Conclusion

By addressing these risks proactively and implementing appropriate mitigation strategies, real estate price prediction projects can enhance the accuracy, reliability, and ethical integrity of their predictions. Continuous monitoring, adaptation to changing market conditions, and transparency in methodologies are key to ensuring successful outcomes and informed decision-making for stakeholders involved in real estate investment and management.

2.3.1 ALGORITHMS

- **Random Forest** is a popular ensemble learning technique in machine learning, particularly for classification and regression tasks. It operates by constructing multiple decision trees during training and outputting the mode (classification) or mean (regression) of the individual trees. Each decision tree is trained on a random subset of the training data and a random subset of features, reducing the risk of overfitting and increasing robustness. Random Forests are known for their versatility, scalability, and ability to handle high-dimensional data, making them widely used in various domains, including finance, healthcare, and marketing.

Why Random Forest for Real Estate Price Prediction?

1. **High Accuracy:** By combining the predictions of multiple decision trees, Random Forest often achieves higher accuracy than individual decision trees.
2. **Robustness:** Random Forest is less likely to overfit compared to a single decision tree because it averages multiple trees.
3. **Handling of Missing Values:** Random Forest can handle missing values in the dataset effectively.
4. **Feature Importance:** It provides insights into the importance of each feature, helping to understand which factors are most influential in determining house prices.
5. **Scalability:** It can handle large datasets and high-dimensional data, making it suitable for complex real estate datasets with many features.

How Random Forest Works

1. **Bootstrap Sampling:** Random Forest uses bootstrap sampling to create multiple subsets of the training data. Each subset is used to train a different decision tree.

2. **Random Feature Selection:** When splitting nodes, each tree considers a random subset of features, which ensures diversity among the trees and reduces the correlation between them.
3. **Building Trees:** Each decision tree is built to its full depth without pruning, creating a collection of deep trees.
4. **Aggregating Predictions:** For regression tasks, the final prediction is the average of the predictions from all the individual trees.

Steps to Implement Random Forest for Real Estate Price Prediction

1. **Data Collection:** Gather historical real estate data, including features like location, size, number of bedrooms, age of the property, and other amenities.
2. **Data Preprocessing:** Clean the data, handle missing values, and normalize the features to ensure the data is ready for model training.
3. **Feature Selection and Engineering:** Identify and create relevant features that can improve the model's performance.
4. **Model Training:** Train the Random Forest model on the preprocessed data.
5. **Model Evaluation:** Evaluate the model's performance using metrics like Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared (R^2).
6. **Model Deployment:** Deploy the trained model to a production environment where it can make real-time predictions.
7. **Monitoring and Maintenance:** Continuously monitor the model's performance and update it with new data to ensure it remains accurate.

Model Deployment (using Flask):

```
from flask import Flask, request, jsonify
import joblib

app = Flask(__name__)

# Load the trained model
model = joblib.load('random_forest_model.pkl')

@app.route('/predict', methods=['POST'])
```

```
def predict():
    data = request.get_json(force=True)
    features = scaler.transform([data['features']])
    prediction = model.predict(features)
    return jsonify({'prediction': prediction[0]})

if __name__ == '__main__':
    app.run(debug=True)
```

- **Linear Regression** is a fundamental statistical method used for modeling the relationship between a dependent variable and one or more independent variables. It assumes a linear relationship between the variables, where changes in the independent variables lead to proportional changes in the dependent variable. The goal of linear regression is to find the best-fitting line (or hyperplane in higher dimensions) that minimizes the difference between the observed and predicted values of the dependent variable. This is achieved by estimating the coefficients of the linear equation through techniques like Ordinary Least Squares (OLS) or gradient descent. Linear Regression is widely used for prediction and inference tasks in fields such as economics, finance, and social sciences.

Steps to Implement Linear Regression for Real Estate Price Prediction

Data Collection: Gather historical real estate data, including features like location, size, number of bedrooms, age of the property, and other amenities.

Data Preprocessing: Clean the data, handle missing values, and normalize the features to ensure the data is ready for model training.

Feature Selection and Engineering: Identify and create relevant features that can improve the model's performance.

Model Training: Train the Linear Regression model on the preprocessed data.

Model Evaluation: Evaluate the model's performance using metrics like Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared (R^2).

Model Deployment: Deploy the trained model to a production environment where it can make real-time predictions.

Monitoring and Maintenance: Continuously monitor the model's performance and update it with new data to ensure it remains accurate.

Model Deployment (using Flask):

python

Copy code

```
from flask import Flask, request, jsonify
import joblib

app = Flask(__name__)

# Load the trained model
model = joblib.load('linear_regression_model.pkl')

@app.route('/predict', methods=['POST'])
def predict():
    data = request.get_json(force=True)
    features = scaler.transform([data['features']])
    prediction = model.predict(features)
    return jsonify({'prediction': prediction[0]})

if __name__ == '__main__':
    app.run(debug=True)
```

2.4 PSEUDOCODE

A) TRAINING

```
X_train,X_test,y_train,y_test=train_test_split  
(X,y,test_size=0.20,random_state=0)
```

```
print(X_train.shape)  
print(X_test.shape)  
print(y_train.shape)  
print(y_test.SHAPE)
```

B) APPLYING LINEAR REGRESSION

```
lr=LinearRegression()  
import pickle
```

```
lr.fit(X_train,y_train)  
with  
open("model.pkl",  
"wb") as f:  
    pickle.dump(lr, f)
```

```
y_pred=lr.predict(X_test)
```

```
y_test
```

```

3777    77
5462    72
2860    50
3265    70
3430    39
...
4453    46
6101    65
1364    41
2865    68
2434   120
Name: price, Length: 1380, dtype: int32

```

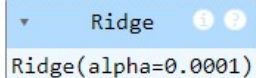
```
[ ] print("R2 score of Lasso model is: ", r2_score(y_test,y_pred_lasso))
```

```
R2 score of Lasso model is: 0.8026679284159816
```

C) APPLYING RIDGE

```
ridge=Ridge(alpha=0.0001)
```

```
ridge.fit(X_train,y_train)
```



```
Ridge(alpha=0.0001)
```

```

from sklearn.model_selection import
ShuffleSplit
from
sklearn.model_selection import
cross_val_score
cv = ShuffleSplit(n_splits=5, test_size=0.2,
random_state=0)

cross_val_score(LinearRegression(),
X,y, cv=cv)

```

D) LETS PREDICT THE PRICE

	location	size	total_sqft	bath
1	0	3	1875.0	3
2	0	5	1500.0	5
3	0	3	2065.0	4
5	0	3	2059.0	3
6	0	2	1394.0	2
...
10329	144	1	812.0	1
10332	144	2	1200.0	2
10333	144	2	1015.0	2
10335	144	3	1805.0	3
10336	144	4	3600.0	5

6896 rows × 4 columns

```
pip install
xgboost import
xgboost as xgb
```

```
xgb.fit(X_train,y_train) with
open("xgbmodel.pkl",
"wb") as f:
    pickle.dump(xgb,
f)
```

```
XGBRegressor
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=None, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             multi_strategy=None, n_estimators=None, n_jobs=None,
             num_parallel_tree=None, random_state=None, ...)
```

E) BUILDING A PREDICTIVE SYSTEM

```
input_data=(144,4,1900,2)

# changing input data to numpy array
input_data_as_numpy_array=np.asarray(input_data)

# Reshaping the array input_data_resaped=
input_data_as_numpy_array.reshape(1,-1)

prediction=xgb.predict(input_data_resaped)
print(prediction)
```

OUTPUT:

[175.61967]

```
input_data=(142,3,2100,3)

input_data_as_numpy_array=np.asarray(input_data)
input_data_resaped= input_data_as_numpy_array.reshape(1,-1)
prediction=xgb.predict(input_data_resaped)
print(prediction)
```

OUTPUT:

[145.5665]

3. RESULTS

3.1 DATASET

The dataset provided contains information on house prices in Bengaluru (Bangalore), India. It includes several attributes such as the area type, availability, location, size, society, total square footage, number of bathrooms, number of balconies, and the corresponding price of the houses.

Dataset: [linkhttps://github.com/1-apex/Real-Estate-PricePrediction/blob/main/Bengaluru_House_Data.csv](https://github.com/1-apex/Real-Estate-PricePrediction/blob/main/Bengaluru_House_Data.csv)

3.2 CSV DATASET PROCESSING

3.2.1 PREPROCESSING TECHNIQUES

A) IMPORTING DATASET

```
df=pd.read_csv('D:/Real-Estate-Price-Prediction-main/Bengaluru_House_Data.csv')
```

OUTPUT:



	area_type	availability	location	size	society	total_sqft	bath	balcony	price
0	Super built-up Area	19-Dec	Electronic City Phase II	2 BHK	Coomee	1056	2.0	1.0	39.07
1	Plot Area	Ready To Move	Chikka Tirupathi	4 Bedroom	Theanmp	2600	5.0	3.0	120.00
2	Built-up Area	Ready To Move	Uttarahalli	3 BHK	NaN	1440	2.0	3.0	62.00
3	Super built-up Area	Ready To Move	Lingadheeranahalli	3 BHK	Soiewre	1521	3.0	1.0	95.00
4	Super built-up Area	Ready To Move	Kothanur	2 BHK	NaN	1200	2.0	1.0	51.00

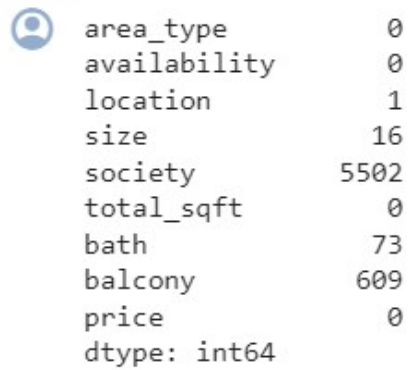
B) LETS CHECK THE VALUE COUNTS OF EACH COLUMN TO HAVE A BETTER IDEA ABOUT THE DATA

```
for i in df.columns:  
    print(df[i].value_counts())  
    print('-'*30)
```

C) CHECKING AND TREATING NULL VALUES


```
df.isna().sum()
```

OUTPUT:

A screenshot of a Jupyter Notebook cell showing the output of the command df.isna().sum(). The output is a list of columns and their corresponding counts of missing values. The columns are area_type (0), availability (0), location (1), size (16), society (5502), total_sqft (0), bath (73), balcony (609), and price (0). The dtype is int64.

area_type	0
availability	0
location	1
size	16
society	5502
total_sqft	0
bath	73
balcony	609
price	0

dtype: int64

D) AFTER EXPLORING THE DATASET, I FOUND THAT THE COLUMN AREA-TYPE, SOCIETY ,AVAILABILITY AND BALCONY IS NOT MUCH CONTRIBUTING TO THE HOUSE PRICE. SO LETS DROP THOSE COLUMN

```
df.drop(['area_type','availability','society','balcony'],axis=1,inplace=True) df.head(2) OUTPUT:
```

	location	size	total_sqft	bath	price
0	Electronic City Phase II	2 BHK	1056	2.0	39.07
1	Chikka Tirupathi	4 Bedroom	2600	5.0	120.00

E) ALL THE MISSING VALUES WERE TREATED AND NOW THERE ARE NO MISSING VALUES IN THE DATASET

```
df.info()
```

OUTPUT:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13320 entries, 0 to 13319
Data columns (total 5 columns):
#   Column      Non-Null Count  Dtype
---  -
0   location    13320 non-null  object
1   size        13320 non-null  object
2   total_sqft  13320 non-null  object
3   bath        13320 non-null  float64
4   price       13320 non-null  float64
dtypes: float64(2), object(3)
memory usage: 520.4+ KB
```

F) THERE ARE 1053 LOCATION WHICH OCCURRED LESS THAN OR EQUAL TO 10 TIMES IN THE DATASET. SO LETS REPLACE THOSE RECORD AS OTHER

```
df['location']=df['location'].apply(lambda x: 'other' if x
in location_count_less_than_20 else x)
```

OUTPUT:

```
location
other          4288
Whitefield     542
Sarjapur Road  399
Electronic City 304
Kanakpura Road 273
...
Domlur         22
Hoskote        22
Binny Pete     21
Basaveshwara Nagar 21
Ulsoor         21
Name: count, Length: 145, dtype: int64
```

G) ALL THE FEATURES NOW HAS BEEN CONVERTED TO NUMERICAL

```
df.head()
```

OUTPUT:

	location	size	total_sqft	bath	price	price_per_sqft
0	Electronic City Phase II	2	1056.0	2	39	3693.181818
1	other	4	2600.0	5	120	4615.384615
2	Uttarahalli	3	1440.0	2	62	4305.555556
3	Lingadheeranahalli	3	1521.0	3	95	6245.890861
4	Kothanur	2	1200.0	2	51	4250.000000

H) DETECTING AND REMOVING OUTLIERS

```
# Lets check which columns has outliers
```

```
df.describe()
```

	size	total_sqft	bath	price	price_per_sqft
count	13320.000000	13073.000000	13320.000000	13320.000000	1.307300e+04
mean	2.802778	1554.942029	2.688814	112.462538	7.940828e+03
std	1.294496	1238.458773	1.338754	149.012102	1.072442e+05
min	1.000000	1.000000	1.000000	8.000000	2.678298e+02
25%	2.000000	1100.000000	2.000000	50.000000	4.255319e+03
50%	3.000000	1275.000000	2.000000	72.000000	5.440000e+03
75%	3.000000	1670.000000	3.000000	120.000000	7.335542e+03
max	43.000000	52272.000000	40.000000	3600.000000	1.200000e+07

```
# From above, it can be seen that for total_sqft column min sqft is 1 which is surely an outlier. So lets filter the data where total_sqft is atleast 300  
(df['total_sqft']/df['size']).describe()
```

count	13073.000000
mean	573.254923
std	389.887823
min	0.250000
25%	472.000000

Lets filter our data where total_sqft/size of the house is greater than or equal to 300

```
df=df[(df['total_sqft']/df['size'])  
>=300]
```

```
df.describe()
```

OUTPUT:

	size	total_sqft	bath	price	price_per_sqft
count	12329.000000	12329.000000	12329.000000	12329.000000	12329.000000
mean	2.651472	1590.166773	2.561441	111.340417	6313.256792
std	0.973754	1261.827604	1.072551	152.799538	4191.833785
min	1.000000	300.000000	1.000000	8.000000	267.829813
25%	2.000000	1118.000000	2.000000	49.000000	4197.761194
50%	3.000000	1300.000000	2.000000	70.000000	5291.828794
75%	3.000000	1700.000000	3.000000	115.000000	6933.333333
max	16.000000	52272.000000	16.000000	3600.000000	176470.588235

I) FROM ABOVE, IT CAN BE CLEARLY SEEN THAT ALL THE COLUMNS HAS OUTLIERS,

SO LETS DETECT AND REMOVE THEM

```
# Also lets visualize the outliers

fig,

axes=plt.subplots(2,2,figsize=(15,10))

axes[0,0].boxplot(df['size'])

axes[0,0].set_title('Boxplot of Size')

axes[0,1].boxplot(df['total_sqft'])

axes[0,1].set_title('Boxplot of

total_sqft')

axes[1,0].boxplot(df['bath'])

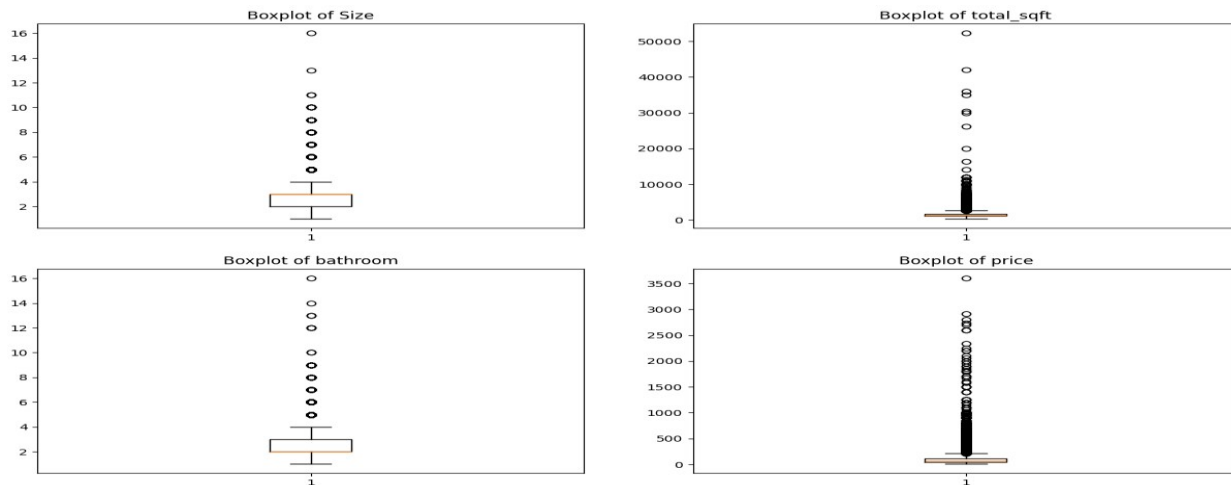
axes[1,0].set_title('Boxplot of

bathroom')

axes[1,1].boxplot(df['price'])

axes[1,1].set_title('Boxplot of price')
```

OUTPUT:Text(0.5, 1.0, 'Boxplot of price')



Here we find that min price per sqft is 267 rs/sqft whereas max is 176470 rs/sqft, this shows a wide variation in property prices. We should remove outliers per location using mean and one standard deviation

J) BUILDING AND TRAINING REGRESSION MODEL

```
X = df.iloc[:,0:4]
X.head(3)
```

	location	size	total_sqft	bath
1	0	3	1875.0	3
2	0	5	1500.0	5
3	0	3	2065.0	4


```
y=df['price']
y
```

1	167
2	85
3	210
5	225
6	100
...	...
10329	26
10332	140
10333	60
10335	134
10336	400

Name: price, Length: 6896, dtype: int32

K) IMPORTING LIBRARIES

```
from sklearn.model_selection import
train_test_split from
sklearn.linear_model import
LinearRegression,Lasso,Ridge from
sklearn.metrics import
```

```

mean_absolute_error,
mean_squared_error,r2_score # Lets
scale the independent variable using
StandardScaler before fitting it into
the model from sklearn.preprocessing
import StandardScaler
ss=StandardScaler()
X_scaled

```

```

array([[ -2.01321953e+00,  5.33805942e-01,  4.40910504e-01,
         5.27810819e-01],
       [ -2.01321953e+00,  2.70684359e+00,  2.10705407e-03,
         2.50005867e+00],
       [ -2.01321953e+00,  5.33805942e-01,  6.63237585e-01,
         1.51393474e+00],
       ...,
       [  1.08689229e+00, -5.52712883e-01, -5.65412074e-01,
        -4.58313106e-01],
       [  1.08689229e+00,  5.33805942e-01,  3.59000527e-01,
         5.27810819e-01],
       [  1.08689229e+00,  1.62032477e+00,  2.45940637e+00,
         2.50005867e+00]])

```

```

pd.DataFrame(X_scaled)

```

OUTPUT:

	0	1	2	3
0	-2.013220	0.533806	0.440911	0.527811
1	-2.013220	2.706844	0.002107	2.500059
2	-2.013220	0.533806	0.663238	1.513935
3	-2.013220	0.533806	0.656217	0.527811
4	-2.013220	-0.552713	-0.121928	-0.458313
...
6891	1.086892	-1.639232	-0.802951	-1.444437
6892	1.086892	-0.552713	-0.348936	-0.458313
6893	1.086892	-0.552713	-0.565412	-0.458313
6894	1.086892	0.533806	0.359001	0.527811
6895	1.086892	1.620325	2.459406	2.500059

6896 rows × 4 columns

L) TRAINING

```
X_train,X_test,y_train,y_test=train_test_split
(X,y,test_size=0.20,random_state=0)
```

```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.SHAPE)
```

M) APPLYING LINEAR REGRESSION

```
lr=LinearRegression()
import pickle
```

```
lr.fit(X_train,y_train)
with open("model.pkl",
"wb") as f:
    pickle.dump(lr, f)
```



```
y_pred=lr.predict(X_test)
```

```
y_test
```

```
3777    77
5462    72
2860    50
3265    70
3430    39
...
4453    46
6101    65
1364    41
2865    68
2434   120
Name: price, Length: 1380, dtype: int32
```

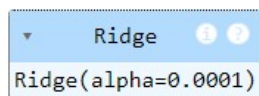
```
[ ] print("R2 score of Lasso model is: ", r2_score(y_test,y_pred_lasso))
```

```
R2 score of Lasso model is: 0.8026679284159816
```

N) APPLYING RIDGE

```
ridge=Ridge(alpha=0.0001)
```

```
ridge.fit(X_train,y_train)
```



```
Ridge(alpha=0.0001)
```

```
from sklearn.model_selection import
ShuffleSplit from
sklearn.model_selection import
cross_val_score
cv = ShuffleSplit(n_splits=5, test_size=0.2,
random_state=0)

cross_val_score(LinearRegression(),
X,y, cv=cv)
```

O) LETS PREDICT THE PRICE

	location	size	total_sqft	bath
1	0	3	1875.0	3
2	0	5	1500.0	5
3	0	3	2065.0	4
5	0	3	2059.0	3
6	0	2	1394.0	2
...
10329	144	1	812.0	1
10332	144	2	1200.0	2
10333	144	2	1015.0	2
10335	144	3	1805.0	3
10336	144	4	3600.0	5

6896 rows × 4 columns

```
pip install
xgboost import
xgboost as xgb
```

```
xgb.fit(X_train,y_train) with
open("xgbmodel.pkl",
"wb") as f:
    pickle.dump(xgb,
f)
```

```
XGBRegressor
XGBRegressor(base_score=None, booster=None, callbacks=None,
               colsample_bylevel=None, colsample_bynode=None,
               colsample_bytree=None, device=None, early_stopping_rounds=None,
               enable_categorical=False, eval_metric=None, feature_types=None,
               gamma=None, grow_policy=None, importance_type=None,
               interaction_constraints=None, learning_rate=None, max_bin=None,
               max_cat_threshold=None, max_cat_to_onehot=None,
               max_delta_step=None, max_depth=None, max_leaves=None,
               min_child_weight=None, missing=nan, monotone_constraints=None,
               multi_strategy=None, n_estimators=None, n_jobs=None,
               num_parallel_tree=None, random_state=None, ...)
```

P) BUILDING A PREDICTIVE SYSTEM

```
input_data=(144,4,1900,2)

# changing input data to numpy array
input_data_as_numpy_array=np.asarray(input_data)

# Reshaping the array input_data_resaped=
input_data_as_numpy_array.reshape(1,-1)

prediction=xgb.predict(input_data_resaped)
print(prediction)
```

OUTPUT:

[175.61967]

```
input_data=(142,3,2100,3)

input_data_as_numpy_array=np.asarray(input_data)
input_data_resaped= input_data_as_numpy_array.reshape(1,-1)
prediction=xgb.predict(input_data_resaped)
print(prediction)
```

OUTPUT:

[145.5665]

3.2.2 FEATURE EXTRACTION

Feature extraction is a crucial step in preparing data for machine learning models, involving the selection and transformation of raw data into a format that is more conducive to model training. It begins with identifying the features within the dataset that are expected to have predictive value for the target variable. Feature selection techniques, such as correlation analysis or feature importance ranking, help in determining the most relevant features. Following this, feature transformation

methods, like normalization or scaling, ensure that all features are on a similar scale, thus preventing any bias towards certain features during model training. Handling categorical features may involve encoding them into numerical values using techniques like one-hot encoding or label encoding. Additionally, feature engineering allows for the creation of new features from existing ones, potentially enriching the dataset with additional predictive information. Dimensionality reduction techniques such as Principal Component Analysis (PCA) or t-distributed Stochastic Neighbor Embedding (t-SNE) may be employed to address high-dimensional data, reducing complexity while preserving relevant information. Feature extraction is an iterative process, often requiring experimentation and validation through techniques like cross-validation to ensure the effectiveness and generalization of the feature extraction pipeline without introducing biases or information leakage. Through careful feature extraction, the performance and interpretability of machine learning models can be significantly improved, leading to more accurate predictions and insights from the data.

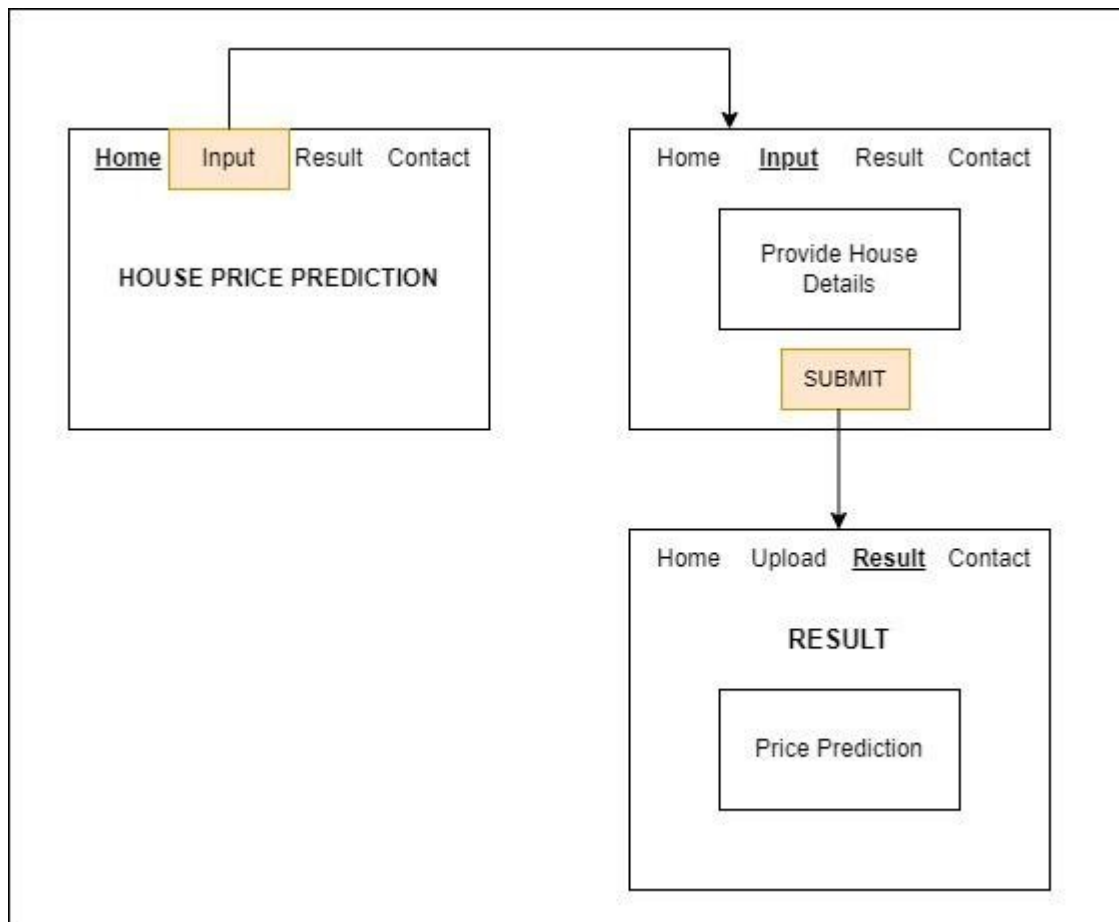


Fig 2:Working of project

3.2.3 MACHINE LEARNING TECHNIQUES

A) APPLYING LINEAR REGRESSION

The provided code segment applies Linear Regression to a dataset. It begins by initializing a Linear Regression model using the `LinearRegression()` function from the `sklearn.linear_model` module. Subsequently, the model is trained using the `fit()` method with the training data (`X_train`) and corresponding target labels (`y_train`). Following training, the trained model is serialized and saved into a file named "model.pkl" using Python's `pickle` module. Afterward, predictions are generated on the test data (`X_test`) using the trained model, and the resulting predictions are stored in the variable `y_pred`. However, it seems there's a line `y_test` which may have been intended to print the actual test labels, but it's not part of the logic for model training or prediction. If the intention is to print the test labels, it would require a print statement like `print(y_test)` after the model predictions. Overall, this code snippet showcases the typical workflow of training a Linear Regression model, saving it for future use, and utilizing it to make predictions on new data.

```
lr=LinearRegression()  
import pickle
```

```
lr.fit(X_train,y_train  
) with  
open("model.pkl",  
"wb") as f:  
    pickle.dump(lr, f)
```

```
y_pred=lr.predic  
t(X_test)
```

```
y_test
```

```

3777    77
5462    72
2860    50
3265    70
3430    39
...
4453    46
6101    65
1364    41
2865    68
2434   120
Name: price, Length: 1380, dtype: int32

```

```
[ ] print("R2 score of Lasso model is: ", r2_score(y_test,y_pred_lasso))
```

```
R2 score of Lasso model is: 0.8026679284159816
```

B) APPLYING RIDGE

The provided code segment applies Ridge Regression to a dataset. It begins by instantiating a Ridge Regression model with a specified regularization parameter ('alpha') of 0.0001. Subsequently, the model is trained using the 'fit()' method with the training data ('X_train') and corresponding target labels ('y_train'). Following training, cross-validation is performed using the 'ShuffleSplit' method to create five splits of the data, with 20% reserved for testing in each split. The 'cross_val_score()' function is then used to evaluate the performance of a Linear Regression model ('LinearRegression()') using the specified cross-validation strategy. However, it's worth noting that in this context, Ridge Regression is being used for training, but the cross-validation is being performed with a Linear Regression model. It might be more appropriate to use

Ridge Regression for cross-validation as well, especially if it's the model of interest. Overall, this code snippet demonstrates the application of Ridge Regression and cross-validation techniques to assess model performance.

```
ridge=Ridge(alpha=0.0001)
```

```
ridge.fit(X_train,y_train)
```

```

Ridge
Ridge(alpha=0.0001)

```

```
from sklearn.model_selection import
ShuffleSplit from
sklearn.model_selection import
cross_val_score
    cv = ShuffleSplit(n_splits=5, test_size=0.2,
random_state=0)

cross_val_score(LinearRegression(),
X,y, cv=cv)
```

4.APPENDICES

4.1 – SOURCE CODE

HOME PAGE :

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Real Estate Price Prediction</title>
</head>
<body>
    <h1>Real Estate Price Prediction</h1>
    <form action="/predict" method="post">
        <label for="location">Location:</label>
        <select id="location" name="location">
            {% for loc in locations %}
            <option value="{{ loc }}">{{ loc }}</option>
            {% endfor %}
        </select><br><br>
        <label for="size">Size (in BHK):</label>
        <input type="number" id="size" name="size"
required><br><br>
        <label for="total_sqft">Total Square Feet:</label>
        <input type="number" id="total_sqft" name="total_sqft"
required><br><br>
        <label for="bath">Number of Bathrooms:</label>
        <input type="number" id="bath" name="bath"
required><br><br>
        <button type="submit">Predict</button>
    </form>
</body>
</html>
```

ABOUT PAGE :

```
<!DOCTYPE html>
<html lang="en">
```



```

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
<div class="form-group">
  <label class="pb-2" for="Type">Keyword</label>
  <input type="text" class="form-control form-control-
lg form-control-a" placeholder="Keyword">
</div>
</div>
<div class="col-md-6 mb-2">
  <div class="form-group mt-3">
    <label class="pb-2" for="Type">Type</label>
    <select class="form-control form-select form-control-
a" id="Type">
      <option>All Type</option>
      <option>For Rent</option>
      <option>For Sale</option>
      <option>Open House</option>
    </select>
  </div>
</div>
<div class="col-md-6 mb-2">
  <div class="form-group mt-3">
    <label class="pb-2" for="city">City</label>
    <select class="form-control form-select form-control-
a" id="city">
      <option>All City</option>
      <option>Alabama</option>
      <option>Arizona</option>
      <option>California</option>
      <option>Colorado</option>
    </select>
  </div>
</div>

```

```

        </div>
    </div>
    <div class="col-md-6 mb-2">
        <div class="form-group mt-3">
            <label class="pb-2" for="bedrooms">Bedrooms</label>
            <select class="form-control form-select form-control-
a" id="bedrooms">
                <option>Any</option>
                <option>01</option>
                <option>02</option>
                <option>03</option>
            </select>
        </div>
    </div>
    <div class="col-md-6 mb-2">
        <div class="form-group mt-3">
            <label class="pb-2" for="garages">Garages</label>
            <select class="form-control form-select form-control-
a" id="garages">
                <option>Any</option>
                <option>01</option>
                <option>02</option>
                <option>03</option>
                <option>04</option>
            </select>
        </div>
    </div>
    <div class="col-md-6 mb-2">
        <div class="form-group mt-3">
            <label class="pb-2" for="bathrooms">Bathrooms</label>
            <select class="form-control form-select form-control-
a" id="bathrooms">
                <option>Any</option>

```

```

        <option>01</option>
        <option>02</option>
        <option>03</option>
    </select>
</div>
</div>
<div class="col-md-6 mb-2">
    <div class="form-group mt-3">
        <label class="pb-2" for="price">Min Price</label>
        <select class="form-control form-select form-control-
a" id="price">
            <option>Unlimite</option>
            <option>$50,000</option>
            <option>$100,000</option>
            <option>$150,000</option>
            <option>$200,000</option>
        </select>
    </div>
</div>
<div class="col-md-12">
    <button type="submit" class="btn btn-b">Search
Property</button>
</div>
</div>
</form>
</div>
</div><!-- End Property Search Section -->>

<!-- ===== Header/Navbar ===== -->
<nav class="navbar navbar-default navbar-trans navbar-expand-lg
fixed-top">
    <div class="container">

```

```

        <button class="navbar-toggler collapsed" type="button" data-
bs-toggle="collapse" data-bs-target="#navbarDefault" aria-
controls="navbarDefault" aria-expanded="false" aria-label="Toggle
navigation">

            <span></span>

            <span></span>

            <span></span>

        </button>

        <a class="navbar-brand text-brand" href="index.html">Own<span
class="color-b">Home</span></a>

        <div class="navbar-collapse collapse justify-content-center"
id="navbarDefault">

            <ul class="navbar-nav">

                <li class="nav-item">

                    <a class="nav-link active" href="/">Home</a>

                </li>

                <li class="nav-item">

                    <a class="nav-link " href="/about">About</a>

                </li>

                <li class="nav-item">

                    <a class="nav-link " href="/property">Property</a>

                </li>

                <li class="nav-item">

                    <a class="nav-link " href="/contact">Contact</a>

                </li>

            </ul>

        </div>

<!-- Template Main JS File -->

```

```

        <script src="../../static/assets/js/main.js"></script>

</body>

</html>

```

CONTACT PAGE :

```

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="utf-8">
    <meta content="width=device-width, initial-scale=1.0"
name="viewport">

    <title>OWNHOME</title>
    <meta content="" name="description">
    <meta content="" name="keywords">
</head>

<body>

    <!-- ===== Property Search Section ===== -->
    <div class="click-closed"></div>
    <!--/ Form Search Star /-->
    <div class="box-collapse">
        <div class="title-box-d">
            <h3 class="title-d">Search Property</h3>
        </div>
        <span class="close-box-collapse right-boxed bi bi-x"></span>
        <div class="box-collapse-wrap form">

```

```

<form class="form-a">
  <div class="row">
    <div class="col-md-12 mb-2">
      <div class="form-group">
        <label class="pb-2" for="Type">Keyword</label>
        <input type="text" class="form-control form-control-
lg form-control-a" placeholder="Keyword">
      </div>
    </div>
    <div class="col-md-6 mb-2">
      <div class="form-group mt-3">
        <label class="pb-2" for="Type">Type</label>
        <select class="form-control form-select form-control-
a" id="Type">
          <option>All Type</option>
          <option>For Rent</option>
          <option>For Sale</option>
          <option>Open House</option>
        </select>
      </div>
    </div>
    <div class="col-md-6 mb-2">
      <div class="form-group mt-3">
        <label class="pb-2" for="city">City</label>
        <select class="form-control form-select form-control-
a" id="city">
          <option>All City</option>
          <option>Alabama</option>
          <option>Arizona</option>
          <option>California</option>
          <option>Colorado</option>
        </select>
      </div>
    </div>
  </div>
</form>

```

```

        </div>

<div id="preloader"></div>

    <a href="#" class="back-to-top d-flex align-items-center justify-
content-center"><i class="bi bi-arrow-up-short"></i></a>

    <!-- Vendor JS Files -->

    <script
src="../../static/assets/vendor/bootstrap/js/bootstrap.bundle.min.js">
</script>

    <script src="../../static/assets/vendor/swiper/swiper-
bundle.min.js"></script>

    <script src="../../static/assets/vendor/php-email-
form/validate.js"></script>

    <!-- Template Main JS File -->

    <script src="../../static/assets/js/main.js"></script>

</body>

</html>

```

INDEX PAGE :

```

<!DOCTYPE html>
<html lang="en">

<head>

    <meta charset="utf-8">

    <meta content="width=device-width, initial-scale=1.0"
name="viewport">

    <title>OWNHOME</title>

    <meta content="" name="description">

```

```

    <meta content="" name="keywords">
</head>

<body>

    <!-- ===== Property Search Section ===== -->
    <div class="click-closed"></div>
    <!--/ Form Search Star /-->
    <div class="box-collapse">
        <div class="title-box-d">
            <h3 class="title-d">Search Property</h3>
        </div>
        <span class="close-box-collapse right-boxed bi bi-x"></span>
        <div class="box-collapse-wrap form">
            <form class="form-a">
                <div class="row">
                    <div class="col-md-12 mb-2">
                        <div class="form-group">
                            <label class="pb-2" for="Type">Keyword</label>
                            <input type="text" class="form-control form-control-
lg form-control-a" placeholder="Keyword">
                        </div>
                    </div>
                    <div class="col-md-6 mb-2">
                        <div class="form-group mt-3">
                            <label class="pb-2" for="Type">Type</label>
                            <select class="form-control form-select form-control-
a" id="Type">
                                <option>All Type</option>
                                <option>For Rent</option>
                                <option>For Sale</option>
                                <option>Open House</option>
                            </select>

```



```

        </div>
    </div>
    <div class="col-md-6 mb-2">
        <div class="form-group mt-3">
            <label class="pb-2" for="city">City</label>
            <select class="form-control form-select form-control-l
a" id="city">
                <option>All City</option>
                <option>Alabama</option>
                <option>Arizona</option>
                <option>California</option>
                <option>Colorado</option>
            </select>
        </div>
    </div>
    <div class="col-md-6 mb-2">
        <div class="form-group mt-3">
            <label class="pb-2" for="bedrooms">Bedrooms</label>
            <select class="form-control form-select form-control-l
a" id="bedrooms">
                <option>Any</option>
                <option>01</option>
                <option>02</option>
                <option>03</option>
            </select>
        </div>
    </div>
    <div class="col-md-6 mb-2">
        <div class="form-group mt-3">
            <label class="pb-2" for="garages">Garages</label>
            <select class="form-control form-select form-control-l
a" id="garages">
                <option>Any</option>

```

```

        <option>01</option>
        <option>02</option>
        <option>03</option>
        <option>04</option>
    </select>
</div>
</div>
<div class="col-md-6 mb-2">
    <div class="form-group mt-3">
        <label class="pb-2" for="bathrooms">Bathrooms</label>
        <select class="form-control form-select form-control-
a" id="bathrooms">
            <option>Any</option>
            <option>01</option>
            <option>02</option>
            <option>03</option>
        </select>
    </div>
</div>
<div class="col-md-6 mb-2">
    <div class="form-group mt-3">
        <label class="pb-2" for="price">Min Price</label>
        <select class="form-control form-select form-control-
a" id="price">
            <option>Unlimite</option>
            <option>$50,000</option>
            <option>$100,000</option>
            <option>$150,000</option>
            <option>$200,000</option>
        </select>
    </div>
</div>
<div class="col-md-12">

```

```

        <button type="submit" class="btn btn-b">Search
Property</button>

    </div>

</div>

</form>

</div>

</div><!-- End Property Search Section -->


<!-- ===== Header/Navbar ===== -->

<nav class="navbar navbar-default navbar-trans navbar-expand-lg
fixed-top">

    <div class="container">

        <button class="navbar-toggler collapsed" type="button" data-
bs-toggle="collapse" data-bs-target="#navbarDefault" aria-
controls="navbarDefault" aria-expanded="false" aria-label="Toggle
navigation">

            <span></span>

            <span></span>

            <span></span>

        </button>

        <a class="navbar-brand text-brand" href="index.html">Own<span
class="color-b">Home</span></a>

        <div class="navbar-collapse collapse justify-content-center"
id="navbarDefault">

            <ul class="navbar-nav">

                <li class="nav-item">

                    <a class="nav-link active" href="/">Home</a>

                </li>

                <li class="nav-item">

                    <a class="nav-link " href="/about">About</a>

                </li>

```

```

        <li class="nav-item">
            <a class="nav-link " href="/property">Property</a>
        </li>

        <li class="nav-item">
            <a class="nav-link " href="/contact">Contact</a>
        </li>
    </ul>
</div>

    <button type="button" class="btn btn-b-n navbar-toggle-box
navbar-toggle-box-collapse" data-bs-toggle="collapse" data-bs-
target="#navbarTogglerDemo01">
        <i class="bi bi-search"></i>
    </button>

</div>
</nav><!-- End Header/Navbar -->

<!-- ===== Intro Section ===== -->
<div class="intro intro-carousel swiper position-relative">

    <div class="swiper-wrapper">

        <div class="swiper-slide carousel-item-a intro-item bg-image"
style="background-image: url(../static/assets/img/slide-1.jpg)">
            <div class="overlay overlay-a"></div>
            <div class="intro-content display-table">
                <div class="table-cell">
                    <div class="container">
                        <div class="row">

```



```
<!-- Template Main JS File -->
<script src="../../static/assets/js/main.js"></script>

</body>

</html>
```

PRICE PREDICTION RESULT :

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Result</title>
    <!-- Add Google Maps API script -->

</head>
<body>
    <h1>Result</h1>
    <p>Predicted Price: {{ prediction }}</p>

</body>
</html>
```

5.SUMMARY AND CONCLUSION

The paper presents an innovative approach to revolutionize sales and marketing strategies by integrating Machine Learning (ML) techniques to predict real estate prices in India. In a landscape characterized by rapid market fluctuations and evolving consumer preferences, traditional methods often fall short in providing actionable insights. ML algorithms, including Linear Regression and Random Forest Regression, offer a promising solution by leveraging data-driven analysis to forecast property prices accurately. Through meticulous data collection and preprocessing, coupled with advanced feature engineering, the model aims to capture the multifaceted dynamics influencing real estate values, ranging from geographical factors to economic indicators. Furthermore, the incorporation of visualization tools facilitates a deeper understanding of data patterns and correlations, empowering stakeholders to make informed decisions amidst the complexities of the real estate market. By providing a comprehensive framework for price prediction, the paper not only addresses the pressing need for predictive analytics but also promises to transform the way businesses formulate sales strategies and navigate the competitive landscape. With its potential to enhance decision-making processes and optimize resource allocation, this approach heralds a new era of agility and efficiency in the real estate industry. In summary, the provided code snippets illustrate the application of linear regression techniques, including ordinary least squares regression and ridge regression, to a given dataset. The process begins with data preprocessing, which involves cleaning the data and handling outliers. Following this, the dataset is split into training and testing sets using the `'train_test_split()'` function from the `'sklearn.model_selection'` module. The first code segment applies ordinary least squares linear regression by instantiating a `'LinearRegression()'` model, fitting it to the training data, and saving the trained model using pickle serialization. Predictions are then made on the test data using the trained model. However, it seems there's a line intended to print the actual test labels (`'y_test'`), which should be replaced with a print statement. The second code segment applies ridge regression, where a Ridge Regression model with a specified regularization parameter is instantiated and trained using the training data. Cross-validation is performed using the `'ShuffleSplit'` method to evaluate the performance of a Linear Regression model, rather than the Ridge Regression model that was trained. In conclusion, the provided code demonstrates the application of linear regression techniques to predict outcomes based on input features. However, there are some areas for improvement, such as ensuring consistency in the choice of algorithms for training and evaluation, as well as implementing proper validation and evaluation techniques to assess

model performance accurately. Additionally, the integration of error handling and model evaluation metrics would enhance the robustness and reliability of the machine learning pipeline.

8.SNAPSHOTS

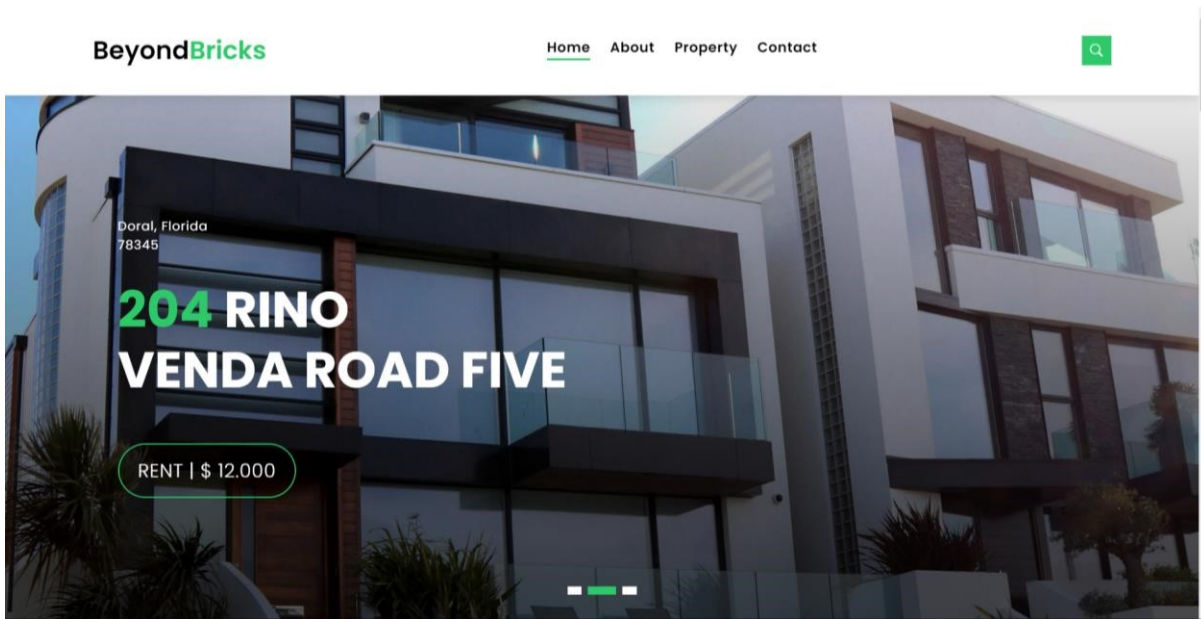


Fig 1: Home Page

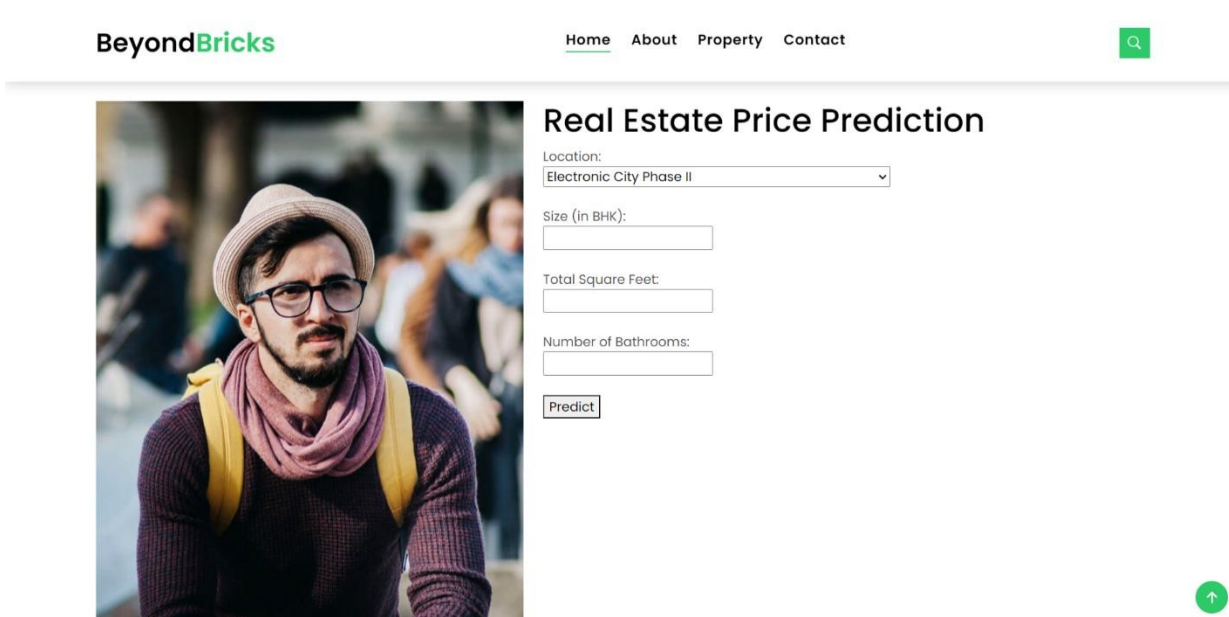


Fig 2: Price Prediction Page

1) SNAPSHOT 3:

BeyondBricks

[Home](#) [About](#) [Property](#) [Contact](#)



Predicted House Price

[175.94868841]

All

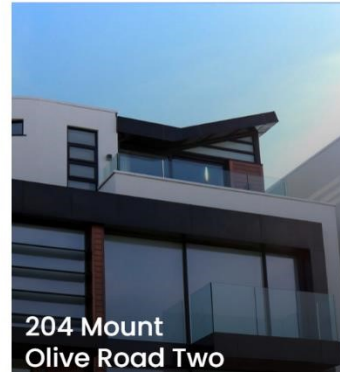
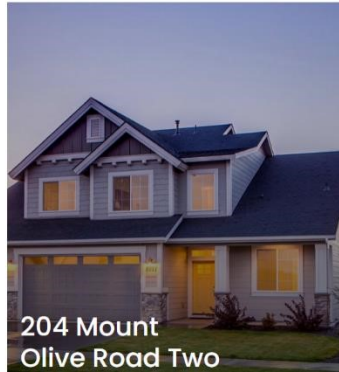
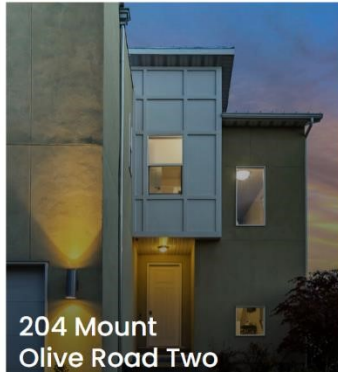


Fig 3: Price Prediction result Page

2) SNAPSHOT 4:

BeyondBricks

[Home](#) [About](#) [Property](#) [Contact](#)



fugit consectetur quo. Et ipsum eveniet laboriosam voluptas beatae possimus qui ducimus. Et voluptatem deleniti. Voluptatum voluptatibus amet. Et esse sed omnis inventore hic culpa.



Fig 4: Location page

3) SNAPSHOT 5:

BeyondBricks

[Home](#) [About](#) [Property](#) [Contact](#)


Your Name

Your Email


Subject

Message


Send Message

 Say Hello

Email: contact@example.com
Phone: +54 356 945234

 Find us in

Manhattan, Nueva York 10036,
EE. UU.

 Social networks





   

Fig 5: Contact Us page

6.FUTURE ENHANCEMENT

Moving forward, several avenues for enhancing the project predicting real estate prices using Machine Learning algorithms can be explored. Firstly, expanding the dataset by incorporating diverse sources such as economic indicators, demographic trends, and urban development plans could provide a more holistic understanding of market dynamics. Advanced feature engineering techniques, including spatial and temporal analysis, could extract nuanced features to capture subtle correlations affecting property prices. Experimentation with more sophisticated ML models like Gradient Boosting Machines or Neural Networks could improve prediction accuracy, while techniques to enhance model interpretability such as feature importance analysis and SHAP values could provide deeper insights into predictions. Implementing dynamic model updating mechanisms would ensure the model remains relevant amidst changing market conditions. Integration of geospatial analysis methods could account for spatial dependencies in real estate markets, while user-friendly interfaces and visualization tools would facilitate easy interpretation of predicted prices. Personalization techniques based on market segmentation could enable targeted strategies, while measures to mitigate biases and ensure fairness in decision-making processes are imperative. Collaboration with domain experts could enrich the analysis with specialized insights, fostering interdisciplinary approaches for more impactful solutions in the real estate domain. For future enhancements, several avenues can be explored to elevate the effectiveness and efficiency of the machine learning pipeline. First and foremost, implementing robust error handling mechanisms would fortify the pipeline against unforeseen issues like missing data or model convergence problems, ensuring smoother execution and user experience. Moreover, delving into hyperparameter tuning techniques such as grid search or random search can optimize model performance by identifying the most effective combination of hyperparameters. Additionally, expanding feature engineering efforts to encompass more sophisticated transformations and combinations can uncover deeper insights within the data, potentially enhancing predictive capabilities. Exploring diverse regularization techniques beyond Ridge Regression, such as Lasso Regression or Elastic Net, can provide better control over model complexity and prevent overfitting. Embracing ensemble methods like stacking or blending, along with fine-tuning cross-validation strategies, can further boost model robustness and generalization. Furthermore, optimizing the machine learning workflow through scikit-learn's 'Pipeline' class can streamline processes, improve reproducibility, and facilitate deployment in real-world scenarios. By pursuing these avenues, the machine learning pipeline can evolve to deliver more accurate, reliable, and scalable solutions across various domains and applications.

7. BIBLIOGRAPHY

- [1] Predicting House Prices Using Multiple Linear Regression Model" Authors: Siti Noor Ismail, Nurfatin
Afifah Zulkifli, Nurul Ain Mohd Yassin Published in: IOP Conference Series: Materials Science and Engineering Year: 2020
- [2] House Price Prediction: Exploratory Data Analysis and Machine Learning Approaches" Authors: Danxia Kong, Ruixue Liu, Yijia Huang Published in: 2019 IEEE International Conference on Big Data (Big Data) Year: 2019
- [3] House Price Prediction: A Deep Learning Approach" Authors: David Henriques, Luis Paulo Reis Published in: 2019 International Conference on Data Mining Workshops (ICDMW) Year: 2019
- [4] House Price Prediction Using a Random Forest Regressor: An Empirical Investigation" Authors: Abhishek M Singh, Shashank Khurana, Ankit Bhola, Anubhav Madhav Published in: 2019 International Conference on Machine Learning and Data Science (MLDS) Year: 2019

