IDENTIFICATION OF METHODOLOGY USED IN REAL ESTATE PROPERTY VALUATION

AN INDUSTRY ORIENTED MINI REPORT

Submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING(AI&ML)

Submitted By

SRIKANTH KUDTHALA	**21UK1A05Q1**
DAVID POTHURAJU	**21UK1A05M3**
NAGARAJU KARRE	**21UK1A05M8**
ABHIRAM MAMIDI	**21UK1A05Q6**

Under the guidance of

Mr. G. Satish chander

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VAAGDEVI ENGINEERING COLLEGE

Affiliated to JNTUH, HYDERABAD

BOLLIKUNTA, WARANGAL (T.S) – 506005

DEPARTMENT OF

COMPUTER SCIENCE AND ENGINEERING

VAAGDEVI ENGINEERING COLLEGE(WARANGAL)

CERTIFICATE OF COMPLETION
INDUSTRY ORIENTED MINI PROJECT

This is to certify that the UG Project Phase-1 entitled "Identification Of Methodology Used In RealEstate property"being submitted by SRIKANTH KUDUTHALA(21UK1A05Q1),DAVID POTHURAJU(21UK1A05M3),NAGARAJU KARRE(21UK1A05M8),ABHIRAM MAMIDI(21UK1A05Q6) in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering to Jawaharlal Nehru Technological University Hyderabad during the academic year 2024-2025.

Project Guide	**HOD**
Mr.G.Sathish Chander	**DR.R.Naveen Kumar**
Assistant Professor)	(Professor)

External

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr.Syed Musthak Ahmed,** Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this UG Project Phase-1 in the institute.

We extend our heartfelt thanks to **Dr.R.Naveen Kumar**, Head of the Department of CSE, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the UG Project Phase-1.

We express heartfelt thanks to Smart Bridge Educational Services Private Limited, for their constant supervision as well as for providing necessary information regarding the UG Project Phase-1 and for their support in completing the UG Project Phase-1.

We express heartfelt thanks to the guide, **T.SATISH CHANDER ,** Assistant professor, Department of CSE for his constant support and giving necessary guidance for completion of this UG Project Phase-1.

Finally, we express our sincere thanks and gratitude to my family members, friends for their encouragement and outpouring their knowledge and experience throughout the thesis.

SRIKANTH KUDTHALA **(21UK1A05Q1)**
DAVID POTHURAJU **(21UK1A05M3)**
NAGARAJU KARRE **(21UK1A05M8)**
ABHIRAM MAMIDI **(21UK1A05Q6)**

ABSTRACT

The "Identification of Methodology Used in Real Estate Property Valuation" project aims to analyze and determine the methodologies commonly employed in valuing real estate properties. It involves studying various appraisal techniques, market analysis methods, and data-driven approaches used by real estate professionals, appraisers, and property valuation experts. By understanding the different methodologies, the project seeks to enhance accuracy, transparency, and reliability in real estate property valuations.

Scenario 1: Comparative Market Analysis (CMA)

Real estate agents and brokers use comparative market analysis (CMA) to determine the value of a property by comparing it to similar properties recently sold in the same area. This scenario explores how CMA methodologies vary based on factors such as property type, location, market trends, and comparable property selection criteria.

Scenario 2: Income Approach

Investors and appraisers use the income approach to value income-generating properties such as rental properties, commercial buildings, or investment properties. This scenario delves into the methodologies involved in estimating property value based on potential income streams, cash flow projections, occupancy rates, and market rental rates.

Scenario 3: Cost Approach

The cost approach involves estimating the value of a property by considering the cost of replacing or reproducing it minus depreciation. This scenario examines the methodologies used to assess construction costs, land value, depreciation factors, and adjustments for improvements or renovations when valuing real estate properties.

TABLE OF CONTENTS:-

1.INTRODUCTION 5

1.1 OVERVIEW… 5

1.2 PURPOSE 5

2.LITERATURE SURVEY 8

2.1 EXISTING PROBLEM 8

2.2 PROPOSED SOLUTION.....8-9

3.THEORITICAL ANALYSIS… 10

3.1 BLOCK DIAGRAM..... 10

3.2 HARDWARE /SOFTWARE DESIGNING 10-11

4.EXPERIMENTAL INVESTIGATIONS..... 12-13

5.FLOWCHART…..... 14

6.RESULTS….....15-18

7.ADVANTAGES AND DISADVANTAGES…..... 19

8.APPLICATIONS 20

9.CONCLUSION 20

10. FUTURE SCOPE… 21

11. BIBILOGRAPHY.....22-23

12. APPENDIX (SOURCE CODE)&CODE SNIPPETS....27-

1.INTRODUCTION

1.1.OVERVIEW

The real estate market is integral to economic stability, where precise property valuation is crucial. Traditional appraisal methods, while valuable, often suffer from subjectivity and inefficiency. This project leverages machine learning (ML) to improve the accuracy and efficiency of property valuations.

The primary goal is to develop machine learning models that can predict real estate property values more accurately than traditional methods, using various relevant features.

1.2. PURPOSE

The primary purpose of this project is to develop a robust and accurate methodology for real estate property valuation using machine learning (ML) techniques. This aims to address the limitations of traditional valuation methods, which are often subjective, time-consuming, and prone to inconsistencies.

Specifically, the project aims to:

1.Enhance Accuracy:

Improve the precision of property valuations by utilizing advanced ML models that can capture complex patterns and relationships within real estate data.

2.Increase Efficiency:

Streamline the valuation process, reducing the time and effort required to assess property values compared to manual appraisal methods.

3.Provide Objectivity :

Offer a data-driven approach to property valuation, minimizing human bias and subjectivity inherent in traditional appraisals.

4.Facilitate Informed Decision-Making:

Empower buyers, sellers, investors, and policymakers with reliable and consistent property valuations, aiding in more informed and confident decision-making.

5.Integrate Technology:

Develop a user-friendly web-based application that utilizes the trained ML models, allowing users to easily obtain property valuations by inputting relevant details.

6.Demonstrate ML Potential:

Showcase the potential of machine learning in transforming traditional industries, highlighting its applicability and benefits in real estate property valuation. By achieving these goals, the project seeks to revolutionize the real estate valuation process, making it more accurate, efficient, and accessible to a wide range of stakeholders.

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

-The current practices in real estate property valuation face several significant challenges, which this project aims to address using machine learning methodologies. These problems include:

-Subjectivity and Bias:Human Judgment: Traditional property valuation methods heavily rely on human appraisers whose judgments can vary based on personal experience, interpretation, and potential biases.

-Inconsistencies: Different appraisers might produce varying valuations for

the same property due to subjective assessments, leading inconsistencies.

-Time-Consuming Process:

Manual Analysis: Traditional methods require extensive manual data collection

and analysis, which can be time-consuming and labor-intensive. Delayed

Results: The reliance on human appraisers and manual processes often

results in delays, which can be critical in fast-moving real estate markets.

-Limited Data Utilization:

Data Constraints: Traditional methods often do not fully utilize the vast amounts of data available, such as real-time market trends, economic indicators, and detailed property characteristics.

-Feature Limitations: Key features influencing property values may be overlooked or underutilized, reducing the accuracy of valuations.

Scalability Issues:

-Resource Intensive: Scaling traditional appraisal methods to handle large volumes of properties can be challenging due to the need for extensive human resources.

-Limitations: Appraisers may have limited local knowledge, affecting the accuracy of valuations in diverse geographic locations.

Economic Fluctuations:

-Market Volatility: Rapid changes in market conditions, such as economic downturns or booms, can quickly render traditional valuations outdated.

Lagging Indicators: Traditional methods might rely on historical data that do not accurately reflect current market dynamics.

-Lack of Standardization:

Variable Standards: Different regions and organizations may followvarying standards and methodologies for property valuation, leading to inconsistent results Regulatory Challenges: Compliance with diverse regulatory requirements can complicate the valuation process.

-Limited Predictive Power:

Simplistic Models: Traditional statistical models, while useful, often lack the complexity to capture intricate relationships between property features and their values.

Overfitting/Underfitting: Basic models might overfit to specific datasets or underfit the real-world complexity, leading to less reliable predictions.

-Accessibility:

Professional Dependency: Property owners and potential buyers typically rely on professional appraisers, which can be costly and limit accessibility.

User Unfriendliness: Traditional appraisal reports might be difficult for non-experts to interpret, reducing their practical usability for average users.

-By addressing these existing problems through the application of machine learning, the project aims to provide a more accurate, efficient, and scalable solution for real estate property valuation.

2.2 PROPOSED SOLUTION

Identifying an effective methodology for real estate property valuation involves several key considerations to ensure accuracy and reliability. Here’s a proposed solution that incorporates widely accepted approaches:

1. Market Approach:

- Comparable Sales Method: This method evaluates the property by comparing it to recently sold properties with similar characteristics (location, size, condition). Adjustments are made for differences to arrive at an estimated value.
- Income Approach: Particularly useful for income-generating properties (like rental units or commercial spaces), this method calculates value based on the income it can generate. Capitalization rates are applied to estimate the property's current value based on projected future income.

2. Cost Approach:

- **Replacement Cost Method:** This method estimates the cost to replace the property with a similar one, adjusted for depreciation and obsolescence. It's useful for newer properties or unique buildings.
- **Depreciated Cost Method:** This evaluates the property's current value by subtracting accrued depreciation from its original cost. It's effective for older properties.

3. Valuation Process:

- **Data Collection:** Gather relevant data including property details, recent sales in the area, income potential (if applicable), and construction costs.
- **Analysis:** Apply the chosen valuation methods. For the Market Approach, identify comparable properties and make adjustments. For the Cost Approach, calculate replacement or depreciated costs. For the Income Approach, determine potential income and apply appropriate capitalization rates.
- **Final Valuation:** Synthesize findings from all approaches to arrive at a final valuation. Weight each approach based on relevance and reliability given the property type and market conditions.

4. Considerations:

- **Market Trends:** Account for current market conditions, including supply and demand dynamics, interest rates, and economic trends.
- **Property-Specific Factors:** Assess unique features like location, zoning regulations, environmental factors, and potential for future development or renovation.

5. Validation and Reporting:

- **Validation:** Cross-check results with local real estate experts, recent appraisals, and market indicators.
- **Reporting:** Present findings clearly and comprehensively, detailing methodology, assumptions made, and any limitations or risks associated with the valuation.

6. Technology Integration:

- Utilize real estate valuation software and databases for accurate data collection, analysis, and reporting.
- Implement geographic information systems (GIS) for spatial analysis and mapping of property attributes and market trends.

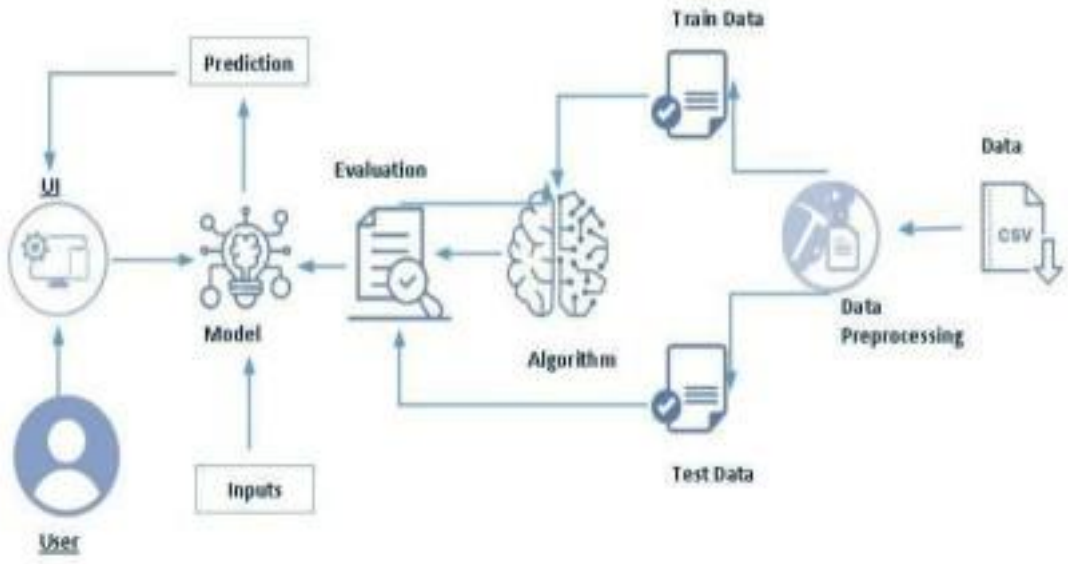
7. Regulatory Compliance:

- Ensure compliance with local regulations, appraisal standards (e.g., USPAP in the United States), and industry best practices.

- By integrating these methodologies and considerations, real estate professionals can develop a robust valuation approach that enhances accuracy, reliability, and confidence in property assessments.

3. THEORETICAL ANALYSIS

3.1. BLOCK DIAGRAM



3.2. SOFTWARE DESIGNING

The following software tools and environments are required to complete the real estate property valuation project using machine learning:

1. Google Colab:

Google Collab will serve as the development and execution environment for predictive modeling, data preprocessing, and model training tasks. It provides a cloud-based Jupyter Notebook environment with access to Python libraries and hardware acceleration.

2. Dataset (CSV File):

The dataset in CSV format is essential for training and testing the predictive model. It should include historical real estate data with

features such as transaction date, house age, distance to the nearest MRT station, number of convenience stores, latitude, longitude, and property prices.

3. Data Preprocessing Tools:

Python libraries like NumPy, Pandas, and Scikit-learn will be used to preprocess the dataset. This includes handling missing data, feature scaling, and data cleaning to ensure the quality of the data used for model training.

4. Feature Selection/Drop:

Feature selection or dropping unnecessary features from the dataset can be done using Scikit-learn or custom Python code to enhance the model's efficiency. This step ensures that only relevant features are used in the predictive modeling process.

5. Model Training Tools:

Machine learning libraries such as Scikit-learn, TensorFlow, or PyTorch will be used to develop, train, and fine-tune the predictive model. Various regression models, including linear regression, decision trees, random forests, and neural networks, can be considered depending on the dataset and project requirements.

6. Model Accuracy Evaluation:

After model training, accuracy and performance evaluation tools, such as Scikit-learn metrics or custom validation scripts, will assess the model's predictive capabilities. Metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared (R^2) will be used to measure the model's performance in predicting property values.

7. UI Based on Flask Environment:

Flask, a Python web framework, will be used to develop the user interface (UI) for the system. The Flask application will provide a user-friendly platform for users to input property details and view predicted valuations.

Integration and Workflow:

Google Colab: The central hub for model development and training. It provides a collaborative environment to write and execute code, handle data preprocessing, and train machine learning models.

Dataset and Data Preprocessing: The dataset will be processed using NumPy, Pandas, and Scikit-learn to clean the data, handle missing values, scale features, and select relevant features for the model.

Model Training: Scikit-learn, TensorFlow, or PyTorch will be used to train the predictive models on the processed dataset. Model hyperparameters will be tuned to optimize performance.

Model Evaluation: The trained models will be evaluated using various performance metrics to ensure their accuracy and reliability in predicting property values.

Flask UI: The Flask web application will facilitate user interaction, allowing users to input property details and view valuation predictions. The Flask environment will communicate with the trained models to generate predictions based on user inputs.

4.EXPERIMENTAL INVESTIGATION

In this project, we have used a Real Estate Property Dataset. This dataset is a CSV file consisting of labeled data and having the following columns:

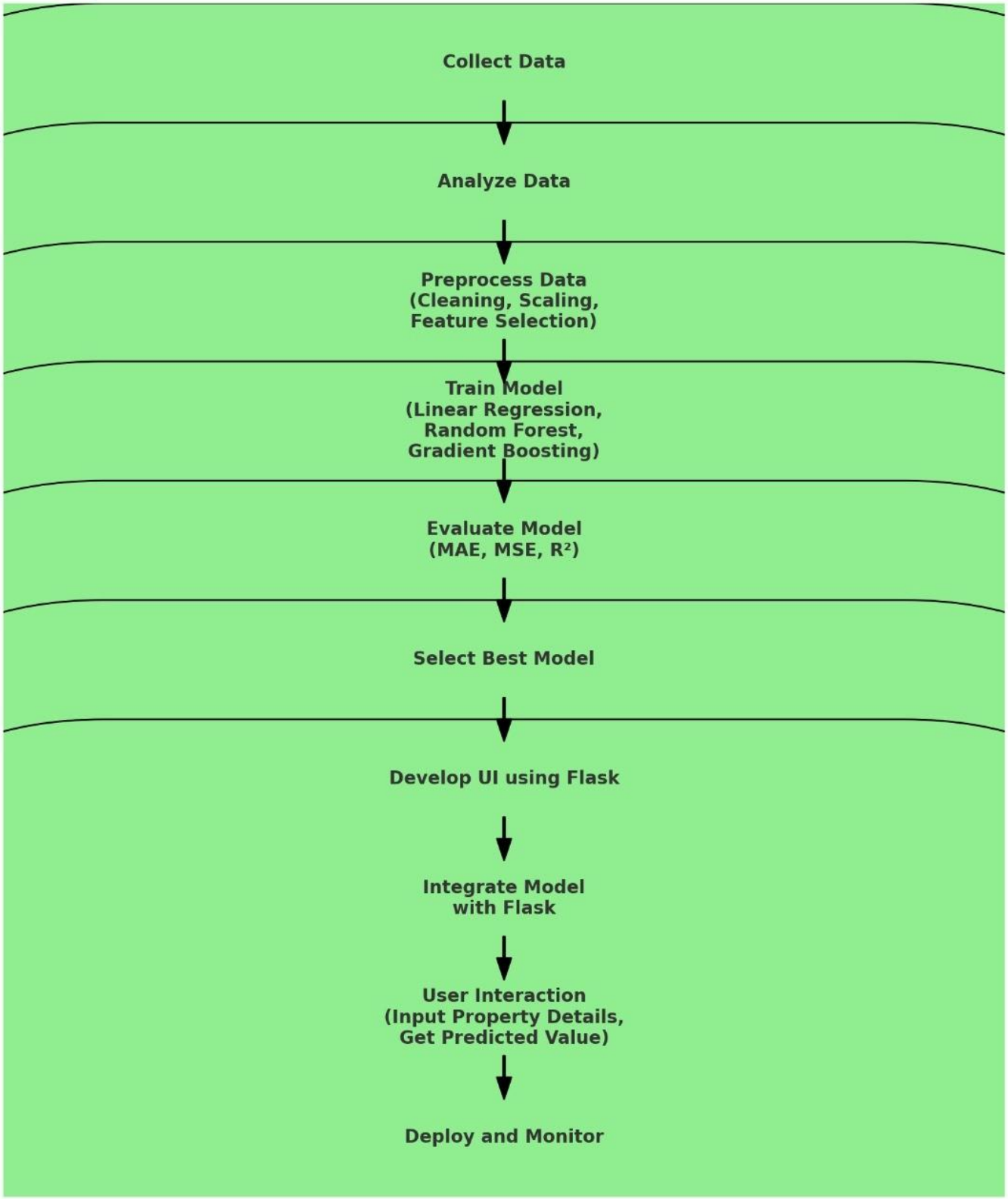
-X1_transaction_date: The date on which the property transaction occurred.

-X2_house_age: The age of the house in years.

-X3_distance_to_the_nearest_MRT_station: The distance from the property to the nearest Mass Rapid Transit (MRT) station.

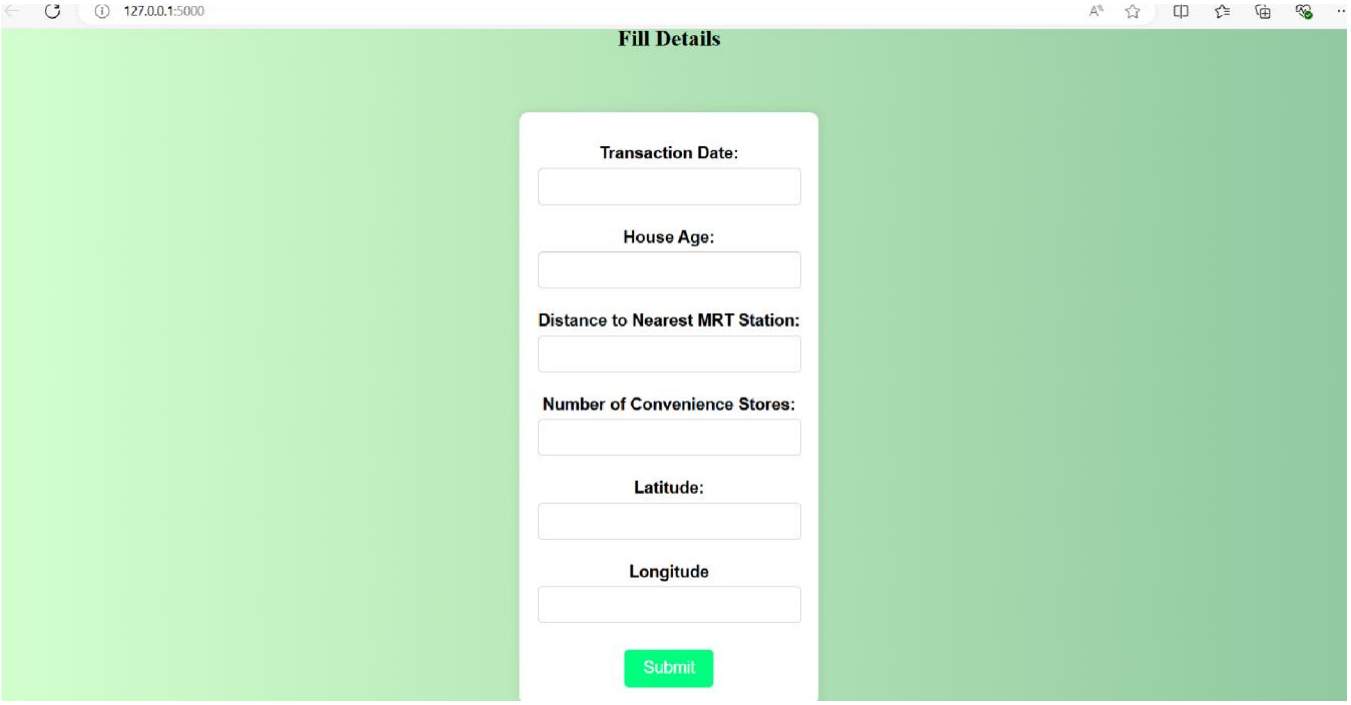
-X4_number_of_convenience_stores: The number of convenience stores within a certain distance from the property.
-X5_latitude: The geographical latitude of the property.
-X6_longitude: The geographical longitude of the property.
Price: The price of the property, which is the target variable for prediction. For the dataset we selected, it consists of more than the columns we want to predict So, we have chosen the feature drop it contains the columns that we are going to predict the House Price value

5.FLOW CHART

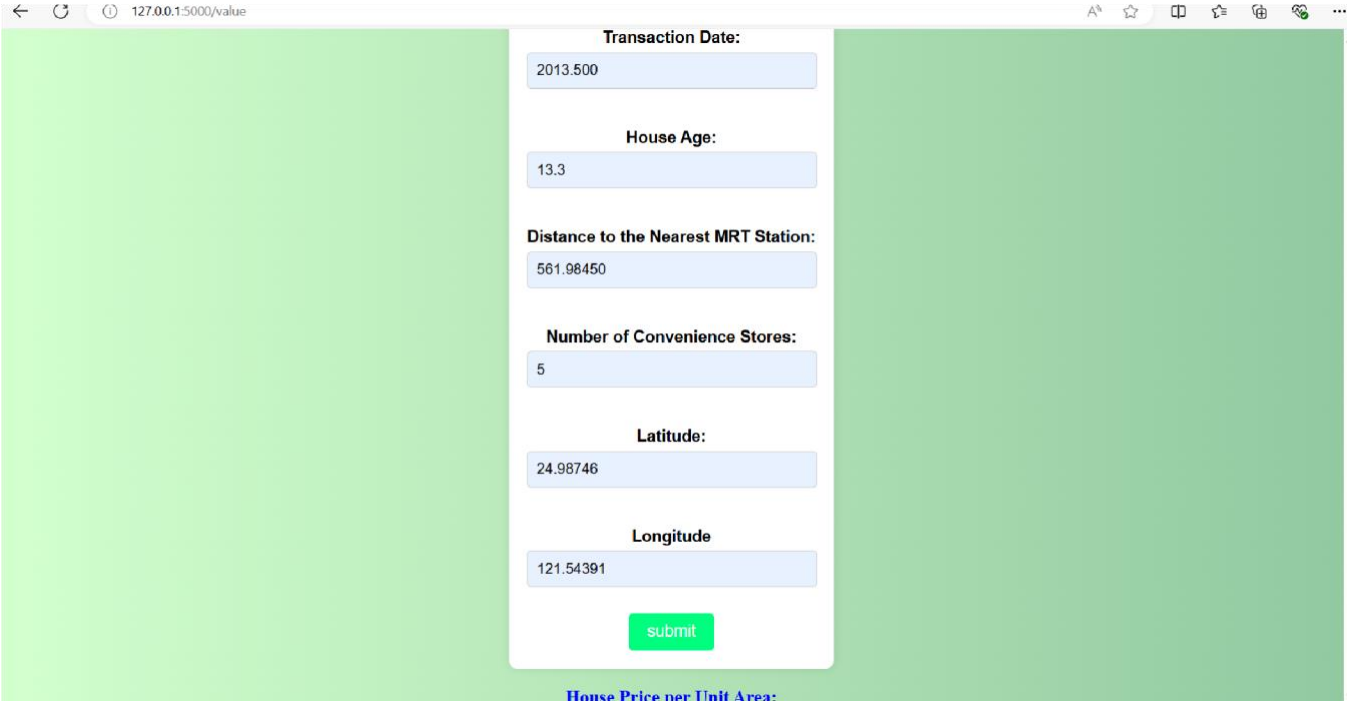


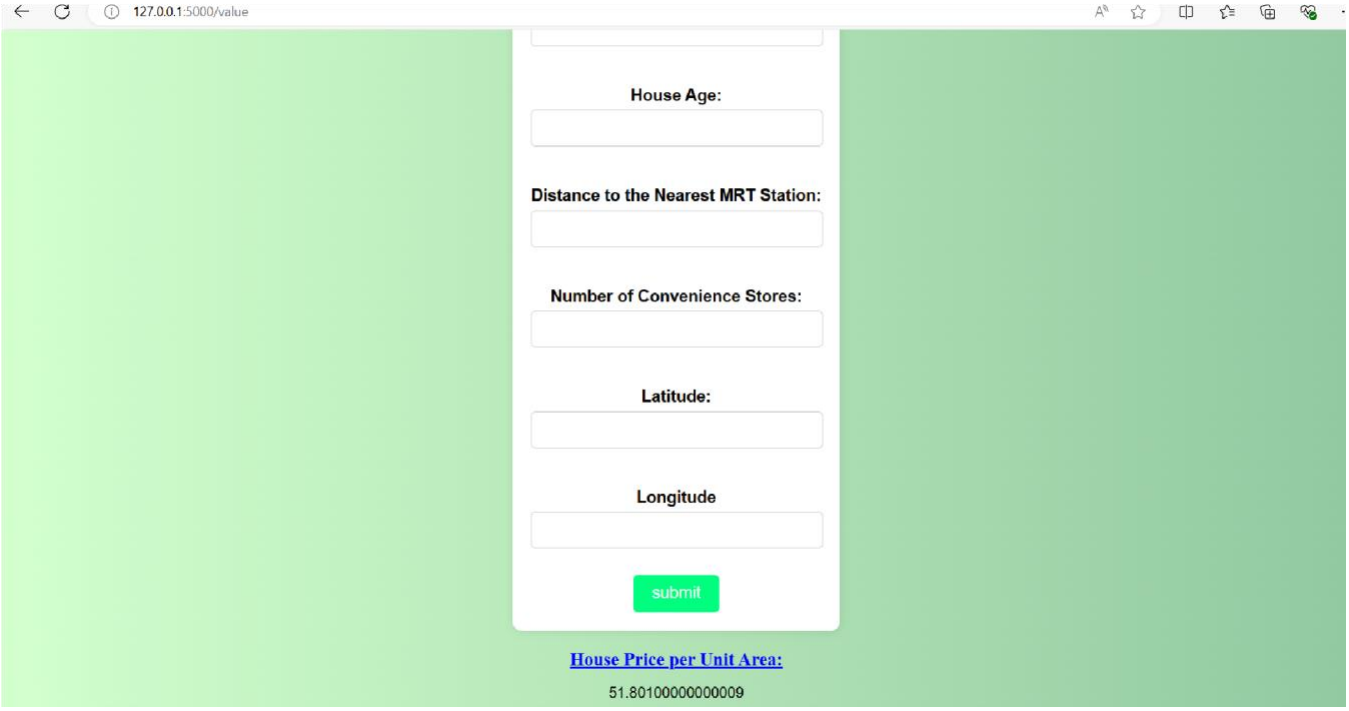
6.RESULT

<u>HOMEPAGE</u>



PREDICTIONS





7.ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

1. **Accuracy:** Machine learning models can analyze large datasets to identify complex patterns and relationships, leading to more accurate property valuations compared to traditional methods.
2. **Efficiency:** Once trained, ML models can quickly process new data and provide valuations, saving time compared to manual appraisals.
3. **Consistency:** ML models apply the same criteria to all properties, ensuring consistent valuations without human bias.
4. **Scalability:** ML models can handle a large number of properties simultaneously, making them suitable for large-scale real estate firms or platforms.
5. **Adaptability:** ML models can be updated with new data, allowing them to adapt to changing market conditions and improve over time.

DISADVANTAGES:

1. **Data Dependency:** The accuracy of ML models depends on the quality and quantity of the data. Incomplete or biased data can lead to inaccurate valuations.
2. **Complexity:** Developing and maintaining ML models requires specialized knowledge in data science and machine learning, which can be a barrier for some organizations.
3. **Transparency:** ML models, especially complex ones, can act as "black boxes," making it difficult to understand how they arrive at specific valuations, which can be a problem for stakeholders who need to understand the decision-making process.
4. **Overfitting:** ML models can overfit to training data, leading to poor performance on new, unseen data if not properly managed.
5. **Regulatory and Ethical Concerns:** There can be regulatory challenges and ethical concerns, particularly if the model's decisions impact significant financial transactions or involve sensitive data.

8.APPLICATIONS

1.Automated Valuation Models (AVMs)

- **Application:** Develop AVMs for instant property valuation.
- **Benefits:** Provides quick, reliable estimates for buyers, sellers, and real estate professionals. Reduces reliance on time-consuming manual appraisals.

2. Property Investment Analysis

- **Application:** Analyze and predict future property values for investment purposes.
- **Benefits:** Helps investors identify profitable investment opportunities by predicting price appreciation and rental income potential.

3. Market Analysis and Trends

- **Application:** Analyze market trends and forecast future property prices.
- **Benefits:** Assists real estate professionals and developers in making informed decisions about property development, marketing strategies, and timing of sales.

4. Risk Assessment for Lenders

- **Application:** Assess the risk associated with mortgage lending by evaluating property values.
- **Benefits:** Helps financial institutions and lenders determine loan-to-value ratios and make informed lending decisions.

5. Personalized Real Estate Services

- **Application:** Provide personalized property recommendations to buyers based on their preferences and budget.
- **Benefits:** Enhances user experience on real estate platforms, leading to higher customer satisfaction and engagement.

6. Property Tax Assessment

- **Application:** Automate the process of property tax assessment by accurately valuing properties.
- **Benefits:** Ensures fair and consistent tax assessments, reduces disputes, and improves efficiency for tax authorities.

7. Urban Planning and Development

- **Application:** Use property valuation models to assist in urban planning and development projects.
- **Benefits:** Helps urban planners and government agencies make data-driven decisions regarding zoning, infrastructure development, and resource allocation.

8. Insurance Underwriting

- **Application:** Determine property values for insurance underwriting and risk assessment.
- **Benefits:** Allows insurance companies to accurately assess property risks and set appropriate premiums.

9. Portfolio Management for Real Estate Firms

- **Application:** Manage and optimize real estate portfolios by predicting property values and market trends.
- **Benefits:** Enables real estate firms to make strategic decisions about property acquisitions, sales, and asset management.

10. Consumer Education and Transparency

- **Application:** Provide consumers with tools to understand property values and market dynamics.
- **Benefits:** Increases transparency in the real estate market, empowering buyers and sellers with accurate information to make informed decisions.

9.CONCLUSION

Incorporating machine learning into real estate property valuation brings numerous advantages that significantly enhance the accuracy, efficiency, and scalability of property assessments. By leveraging data-driven insights, machine learning models can provide more precise and consistent valuations compared to traditional methods. This technological advancement allows for quick processing of large datasets, making it ideal for large-scale applications and providing valuable benefits to a wide range of stakeholders, including buyers, sellers, investors, lenders, and real estate professionals.

However, the implementation of machine learning in this field also presents challenges. The dependency on high-quality data, the complexity of model development, and issues related to transparency and regulatory compliance must be carefully managed to ensure the reliability and acceptance of these models.

The applications of machine learning in real estate are vast and varied, from automated valuation models and investment analysis to urban planning and insurance underwriting. Each application not only enhances decision-making processes but also contributes to a more transparent and efficient real estate market.

10.FUTURE SCOPE

1. Enhanced Data Integration

- **Incorporation of Diverse Data Sources:** Future models will integrate more diverse and rich data sources, including satellite imagery, social media trends, environmental data, and economic indicators, to provide more comprehensive valuations.

- **Real-Time Data Updates:** Models will leverage real-time data from IoT devices, smart homes, and connected city infrastructure to provide up-to-date property valuations reflecting current market conditions.

2. Advanced Modeling Techniques

- **Deep Learning and Neural Networks:** The use of advanced deep learning techniques will enable models to capture even more complex patterns and relationships in the data, leading to higher accuracy.
- **Geospatial Analysis:** Improved geospatial analysis techniques will allow models to better account for the influence of location-specific factors on property values.

3. Personalized Valuations

- **Customization for Individual Users:** Models will offer more personalized property valuations based on individual user preferences, search history, and behavior, enhancing user experience on real estate platforms.
- **Context-Aware Valuations:** Context-aware models will adjust valuations based on specific scenarios, such as future urban development plans or changes in local amenities.

4. Explainable AI and Transparency

- **Improved Model Interpretability:** Future developments will focus on making machine learning models more interpretable and transparent, allowing users to understand the factors driving property valuations.
- **Regulatory Compliance:** Enhanced transparency will help models meet regulatory requirements and gain wider acceptance among stakeholders.

5. Integration with Augmented Reality (AR) and Virtual Reality (VR)

- **AR/VR for Property Viewing:** Integration with AR and VR technologies will allow users to virtually tour properties and receive real-time valuation insights, providing a more immersive and informative experience.
- **Simulations and Scenario Analysis: AR/VR for Property Viewing:** Integration with AR and VR technologies will allow users to virtually tour properties and receive real-time valuation insights, providing a more immersive and informative experience.

6. Automated Decision Support Systems

- **End-to-End Automation:** Future systems will offer end-to-end automation of the property valuation process, from data collection and preprocessing to model training and deployment.
- **Decision Support for Stakeholders:** These systems will provide actionable insights and recommendations for buyers, sellers, investors, and policymakers, enhancing decision-making processes.

7. Sustainable and Ethical AI

- **Sustainability Metrics:** Incorporating sustainability metrics into property valuations, such as energy efficiency and carbon footprint, to promote environmentally responsible real estate practices.
- **Ethical Considerations:** Ensuring models adhere to ethical guidelines, avoid biases, and promote fairness in property valuations.

8. Blockchain Integration

- **Secure and Transparent Transactions:** Integration with blockchain technology will ensure secure, transparent, and tamper-proof property transactions, enhancing trust in the valuation process.
- **Smart Contracts:** Utilizing smart contracts to automate and enforce terms of real estate transactions based on AI-generated valuations.

9. Global Expansion and Accessibility

- **Adaptation to Global Markets:** Models will be adapted to account for regional differences in real estate markets, making them applicable worldwide.
- **Accessibility for All Users:** Ensuring that advanced valuation tools are accessible to all users, including small real estate firms and individual homeowners.

11.BIBILOGRAPHY

Books

1. **James, G., Witten, D., Hastie, T., & Tibshirani, R.** (2013). *An Introduction to Statistical Learning with Applications in R*. Springer.
2. **Murphy, K. P.** (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.
3. **Goodfellow, I., Bengio, Y., & Courville, A.** (2016). *Deep Learning*. MIT Press.

Research Papers and Articles

1. **Bajari, P., Fruehwirth, J. C., Kim, K. I., & Timmins, C.** (2012). *A Rational Expectations Approach to Hedonic Price Regressions with Time-Varying Unobserved Product Attributes: The Price of Pollution*. American Economic Review, 102(5), 1898-1926.
2. **Kok, N., Monkkonen, P., & Quigley, J. M.** (2011). *Economic Geography, Jobs, and Regulations: The Value of Land and Housing*. Regional Science and Urban Economics, 41(2), 122-133.

3. **Poursaeed, O., Matera, T., & Belongie, S.** (2018). *Vision-Based Real Estate Price Estimation*. Machine Learning for Computer Vision Workshop at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Online Resources

1. **Scikit-learn Documentation**. *Feature Engineering and Model Evaluation*. https://scikit-learn.org/stable/documentation.html
2. **Flask Documentation**. *Web Development with Flask*. https://flask.palletsprojects.com/en/2.0.x/
3. **Kaggle**. *House Prices - Advanced Regression Techniques*. https://www.kaggle.com/c/house-prices-advanced-regression-techniques

Websites and Blogs

1. **Towards Data Science**. *Machine Learning in Real Estate: Predicting Housing Prices*. https://towardsdatascience.com/machine-learning-in-real-estate-predicting-housing-prices-29dc8a39d544
2. **Analytics Vidhya**. *A Comprehensive Guide to Data Preprocessing in Machine Learning*. https://www.analyticsvidhya.com/blog/2020/10/data-preprocessing-techniques/
3. **Medium**. *The Future of Real Estate: How AI and Big Data are Changing the Industry*. https://medium.com/@therealestateai/the-future-of-real-estate-how-ai-and-big-data-are-changingthe-industry-6e8c1e0e7df4

Conferences and Workshops

1. **IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. *Machine Learning for Computer Vision Workshop*.
2. **International Conference on Machine Learning (ICML)**. *Advances in Machine Learning for Real Estate Valuatio*

12.APPENDIX

Model building :

1)Dataset
2)Google colab and VS code Application Building
   1. HTML file (Index file, Predict file)
   1. CSS file
   2. Models in pickle format

SOURCE CODE:

INDEX.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>House Price Prediction</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
    <div class="container">
        <h1>Fill Details</h1>
```

```
        <form action="/value" method="POST">

            <label for="X1_transaction_date">Transaction Date:</label>

            <input type="text" id="X1_transaction_date"
name="X1_transaction_date"><br>
```

```
            <label for="X2_house_age">House Age:</label>

            <input type="text" id="X2_house_age" name="X2_house_age"><br>



            <label for="X3_distance_to_the_nearest_MRT_station">Distance to
Nearest MRT Station:</label>

            <input type="text" id="X3_distance_to_the_nearest_MRT_station"
name="X3_distance_to_the_nearest_MRT_station"><br>



            <label for="X4_number_of_convenience_stores">Number of Convenience
Stores:</label>

            <input type="text" id="X4_number_of_convenience_stores"
name="X4_number_of_convenience_stores"><br>



            <label for="X5_latitude">Latitude:</label>

            <input type="text" id="X5_latitude" name="X5_latitude"><br>
```

```
            <label for="X6_longitude">longitude</label>

            <input type="text" id="X6_longitude" name="X6_longitude"
required><br><br>





            <input type="submit" value="Submit">

        </form>



        {% if y is not none %}

            <div class="result">

                <h2>House Price per Unit Area:</h2>

                <p>{{ y }}</p>

            </div>

        {% endif %}

    </div>

</body>

</html>
```

PREDICT.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>House Price Prediction</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
    <div class="container">
        <h1>Fill Details</h1>
        <form action="/value" method="POST">
```

```
            <label for="X1_transaction_date">Transaction Date:</label>
            <input type="text" id="X1_transaction_date"
name="X1_transaction_date" required><br><br>

            <label for="X2_house_age">House Age:</label>
            <input type="text" id="X2_house_age" name="X2_house_age"
required><br><br>

            <label for="X3_distance_to_the_nearest_MRT_station">Distance to the
Nearest MRT Station:</label>
            <input type="text" id="X3_distance_to_the_nearest_MRT_station"
name="X3_distance_to_the_nearest_MRT_station" required><br><br>

            <label for="X4_number_of_convenience_stores">Number of Convenience
Stores:</label>
            <input type="text" id="X4_number_of_convenience_stores"
name="X4_number_of_convenience_stores" required><br><br>

            <label for="X5_latitude">Latitude:</label>
            <input type="text" id="X5_latitude" name="X5_latitude"
required><br><br>
```

```
            <label for="X6_longitude">Longitude</label>
            <input type="text" id="X6_longitude" name="X6_longitude"
required><br><br>


            <input type="submit" value="submit">
        </form>


        {% if y is not none %}
            <div class="result">
                <h2>House Price per Unit Area:</h2>
                <p>{{ y }}</p>
            </div>
        {% endif %}
    </div>
</body>
</html>
```

APP.PY

```
import os
from flask import Flask, render_template, request
import pickle
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
# Load the saved models
md = pickle.load(open("price.pkl", "rb"))
sc = pickle.load(open("scale.pkl", "rb"))



# Initialize the Flask application
app = Flask(__name__)


@app.route('/')
def index():
```

```
    return render_template('index.html')


@app.route('/value', methods=['POST'])
def predict():
    # Retrieve values from the form in the HTML page
    transaction_date = float(request.form['X1_transaction_date'])
    house_age = float(request.form['X2_house_age'])
    distance_to_mrt =
float(request.form['X3_distance_to_the_nearest_MRT_station'])
    convenience_stores = float(request.form['X4_number_of_convenience_stores'])
    latitude = float(request.form['X5_latitude'])
    longitude = float(request.form['X6_longitude'])
    # Create a 2D array for the input data
    data = [[transaction_date, house_age, distance_to_mrt, convenience_stores,
latitude,longitude]]


    # Transform the input data using the loaded scaler
    scaled_data = sc.transform(data)


    # Predict using the loaded model
    prediction = md.predict(scaled_data)


    # Render the template with the prediction result return
    render_template('predict.html', y=prediction[0])


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

CODE SNIPPETS

MODEL BUILDING

```
[1] import pandas as pd
    import numpy as np
    import matplotlib.pyplot as plt
    import seaborn as sns
    from sklearn.model_selection import train_test_split
    from sklearn.ensemble import RandomForestClassifier,GradientBoostingClassifier
    from sklearn.tree import DecisionTreeClassifier
    from sklearn.neighbors import KNeighborsClassifier
    from sklearn.metrics import f1_score
    from sklearn.metrics import classification_report,confusion_matrix
    import warnings
    import pickle
    from scipy import stats
    warnings.filterwarnings('ignore')
    plt.style.use('fivethirtyeight')
```

```
[2] dt=pd.read_csv('/content/drive/MyDrive/ dataset/real estate valuation data set.csv')
```

```
[3] dt.head()
```

	No	X1 transaction date	X2 house age	X3 distance to the nearest MRT station	X4 number of convenience stores	X5 latitude	X6 longitude	Y house price of unit area
0	1	2012.917	32.0	84.87882	10	24.98298	121.54024	37.9
1	2	2012.917	19.5	306.59470	9	24.98034	121.53951	42.2
2	3	2013.583	13.3	561.98450	5	24.98746	121.54391	47.3
3	4	2013.500	13.3	561.98450	5	24.98746	121.54391	54.8
4	5	2012.833	5.0	390.56840	5	24.97937	121.54245	43.1

✓ 0s completed at 2:52 PM

Next steps: Generate code with dt | View recommended plots

```
dt
```

	No	X1 transaction date	X2 house age	X3 distance to the nearest MRT station	X4 number of convenience stores	X5 latitude	X6 longitude	Y house price of unit area
0	1	2012.917	32.0	84.87882	10	24.98298	121.54024	37.9
1	2	2012.917	19.5	306.59470	9	24.98034	121.53951	42.2
2	3	2013.583	13.3	561.98450	5	24.98746	121.54391	47.3
3	4	2013.500	13.3	561.98450	5	24.98746	121.54391	54.8
4	5	2012.833	5.0	390.56840	5	24.97937	121.54245	43.1
...	...	...	...	...	...	...	...	...
409	410	2013.000	13.7	4082.01500	0	24.94155	121.50381	15.4
410	411	2012.667	5.6	90.45606	9	24.97433	121.54310	50.0
411	412	2013.250	18.8	390.96960	7	24.97923	121.53986	40.6
412	413	2013.000	8.1	104.81010	5	24.96674	121.54067	52.5
413	414	2013.500	6.5	90.45606	9	24.97433	121.54310	63.9

414 rows × 8 columns

Next steps: Generate code with dt | View recommended plots

House Price Pred

File Edit View Insert Runtime Tools Help All changes saved

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
[6] dt.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 414 entries, 0 to 413
Data columns (total 8 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   No                                      414 non-null    int64
 1   X1 transaction date                     414 non-null    float64
 2   X2 house age                            414 non-null    float64
 3   X3 distance to the nearest MRT station  414 non-null    float64
 4   X4 number of convenience stores         414 non-null    int64
 5   X5 latitude                             414 non-null    float64
 6   X6 longitude                            414 non-null    float64
 7   Y house price of unit area              414 non-null    float64
dtypes: float64(6), int64(2)
memory usage: 26.0 KB
```

```
[7] dt.dropna(inplace=True)
```

```
[8] dt.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 414 entries, 0 to 413
Data columns (total 8 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   No                                      414 non-null    int64
 1   X1 transaction date                     414 non-null    float64
 2   X2 house age                            414 non-null    float64
 3   X3 distance to the nearest MRT station  414 non-null    float64
 4   X4 number of convenience stores         414 non-null    int64
 5   X5 latitude                             414 non-null    float64
 6   X6 longitude                            414 non-null    float64
 7   Y house price of unit area              414 non-null    float64
dtypes: float64(6), int64(2)
memory usage: 26.0 KB
```

```
[9] dt.isnull().any()
```

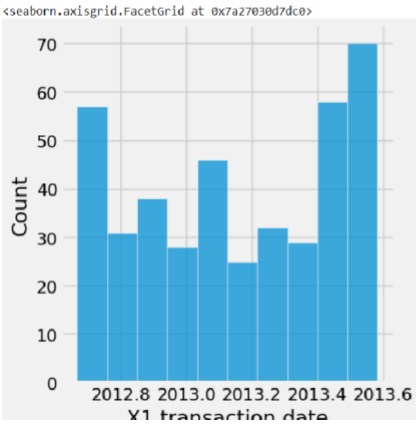
```
No                                      False
X1 transaction date                     False
X2 house age                            False
X3 distance to the nearest MRT station  False
```

House Price Pred

File Edit View Insert Runtime Tools Help All changes saved

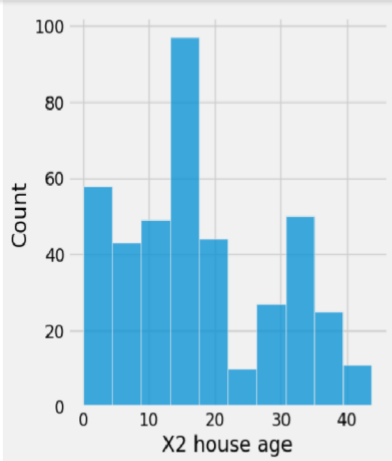
```
sns.displot(dt['X1 transaction date'])
sns.displot(dt['X2 house age'])
```

```
<seaborn.axisgrid.FacetGrid at 0x7a27030d7dc0>
```



House Price Pred

File Edit View Insert Runtime Tools Help All changes saved




```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
train_dt=x_train.join(y_train)
```

House Price Pred

File Edit View Insert Runtime Tools Help All changes saved

```
train_dt
```

	No	X1 transaction date	X2 house age	X3 distance to the nearest MRT station	X4 number of convenience stores	X5 latitude	X6 longitude	Y house price of unit area
192	193	2013.167	43.8	57.58945	7	24.96750	121.54069	42.7
234	235	2013.250	8.0	2216.61200	4	24.96007	121.51361	23.9
5	6	2012.667	7.1	2175.03000	3	24.96305	121.51254	32.1
45	46	2013.083	36.6	488.81930	8	24.97015	121.54494	38.3
245	246	2013.417	7.5	639.61980	5	24.97258	121.54814	40.8
...	...	...	...	...	...	...	...	...
71	72	2013.083	35.5	640.73910	3	24.97563	121.53715	40.8
106	107	2013.083	17.2	189.51810	8	24.97707	121.54308	47.1
270	271	2013.333	10.8	252.58220	1	24.97460	121.53046	117.5
348	349	2012.833	4.6	259.66070	6	24.97585	121.54516	53.7
102	103	2013.083	1.1	193.58450	6	24.96571	121.54089	54.4

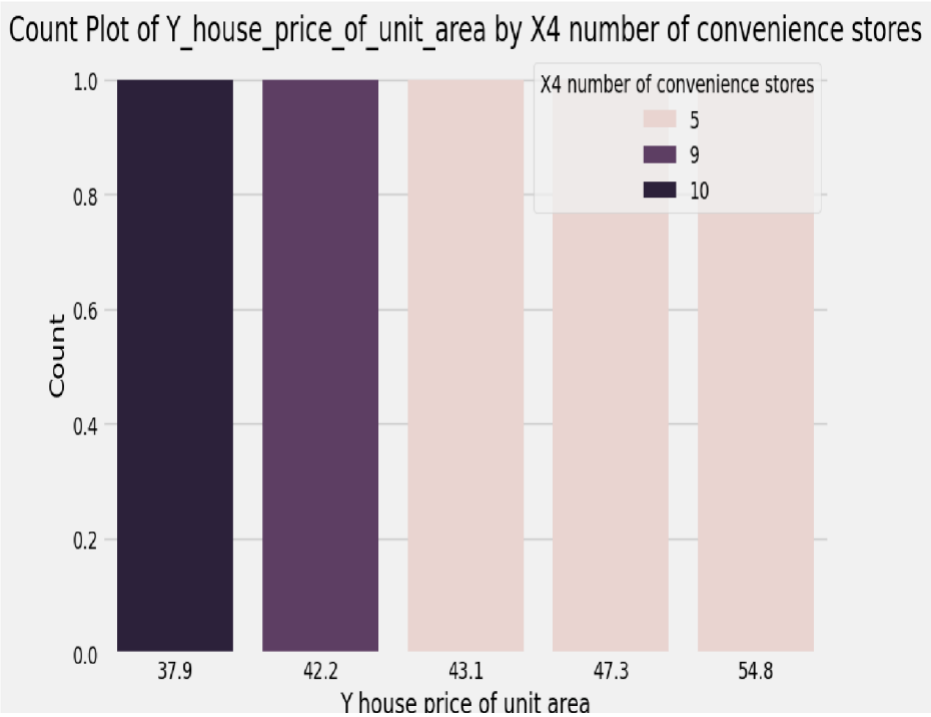
331 rows × 8 columns

```
[ ] plt.figure(figsize=(15, 8))
    sns.scatterplot(x='X3 distance to the nearest MRT station',
    y='X4 number of convenience stores',
```

House Price Pred

File Edit View Insert Runtime Tools Help All changes saved

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Sample data
data = {
    "X1 transaction date": [2012.917, 2012.917, 2013.583, 2013.500, 2012.833],
    "X2 house age": [32.0, 19.5, 13.3, 13.3, 5.0],
    "X3 distance to the nearest MRT station": [84.87882, 306.59470, 561.98450, 561.98450, 390.56840],
    "X4 number of convenience stores": [10, 9, 5, 5, 5],
    "X5 latitude": [24.98298, 24.98034, 24.98746, 24.98746, 24.97937],
    "X6 longitude": [121.54024, 121.53951, 121.54391, 121.54391, 121.54245],
    "Y_house_price_of_unit_area": [37.9, 42.2, 47.3, 54.8, 43.1]
}
dt = pd.DataFrame(data)
plt.figure(figsize=(10, 6))
sns.countplot(x='Y_house_price_of_unit_area', hue='X4 number of convenience stores', data=dt)
plt.title('Count Plot of Y_house_price_of_unit_area by X4 number of convenience stores')
plt.xlabel('Y house price of unit area')
plt.ylabel('Count')
plt.legend(title='X4 number of convenience stores')
plt.show()
```




```
[97] dt.describe()
```

	X1 transaction date	X2 house age	X3 distance to the nearest MRT station	X4 number of convenience stores	X5 latitude	X6 longitude	Y_house_price_of_unit_area
count	5.000000	5.000000	5.000000	5.00000	5.000000	5.000000	5.00000
mean	2013.150000	16.620000	381.202184	6.80000	24.983522	121.542004	45.06000
std	0.360228	10.023822	199.264918	2.48998	0.003830	0.002049	6.38694
min	2012.833000	5.000000	84.878820	5.00000	24.979970	121.539510	37.90000
25%	2012.917000	13.300000	306.594700	5.00000	24.980340	121.540240	42.23000
50%	2012.917000	13.300000	390.568400	5.00000	24.982980	121.542450	43.10000
75%	2013.500000	19.500000	561.984500	9.00000	24.987460	121.543910	47.30000
max	2013.583000	32.000000	561.984500	10.00000	24.987460	121.543910	54.80000

```
dt.isnull().any()
```

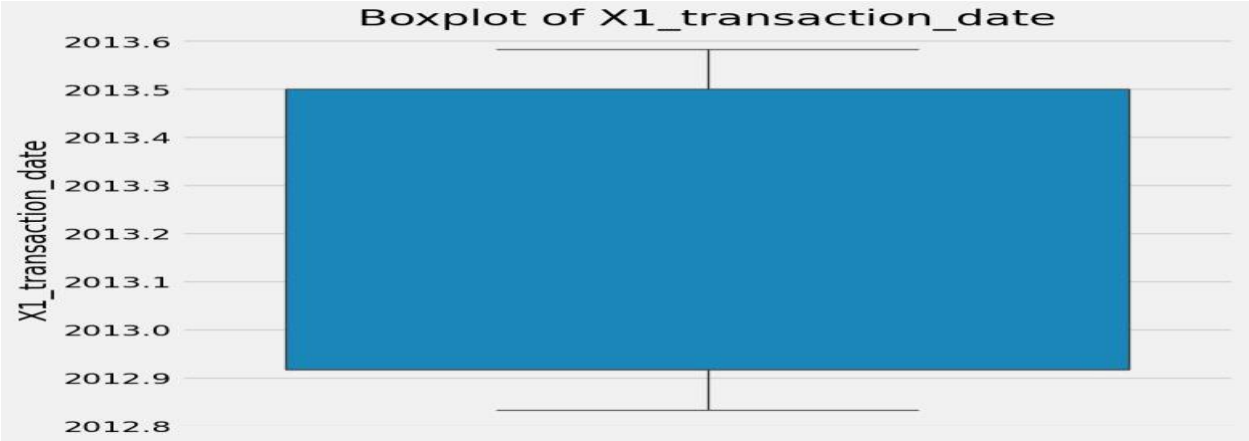
```
X1 transaction date                  False
X2 house age                         False
X3 distance to the nearest MRT station  False
X4 number of convenience stores      False
X5 latitude                          False
X6 longitude                         False
Y_house_price_of_unit_area           False
dtype: bool
```

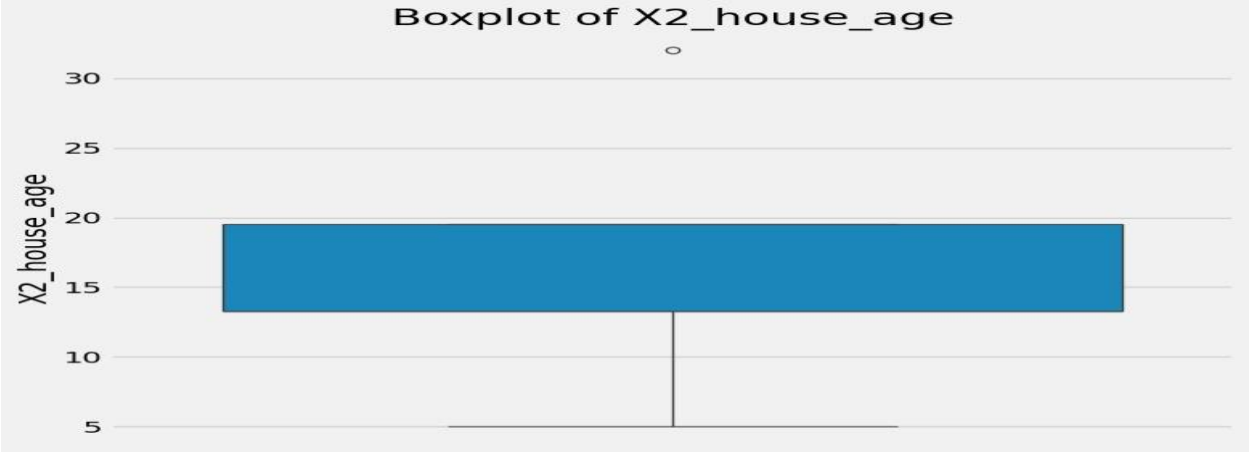
House Price Pred

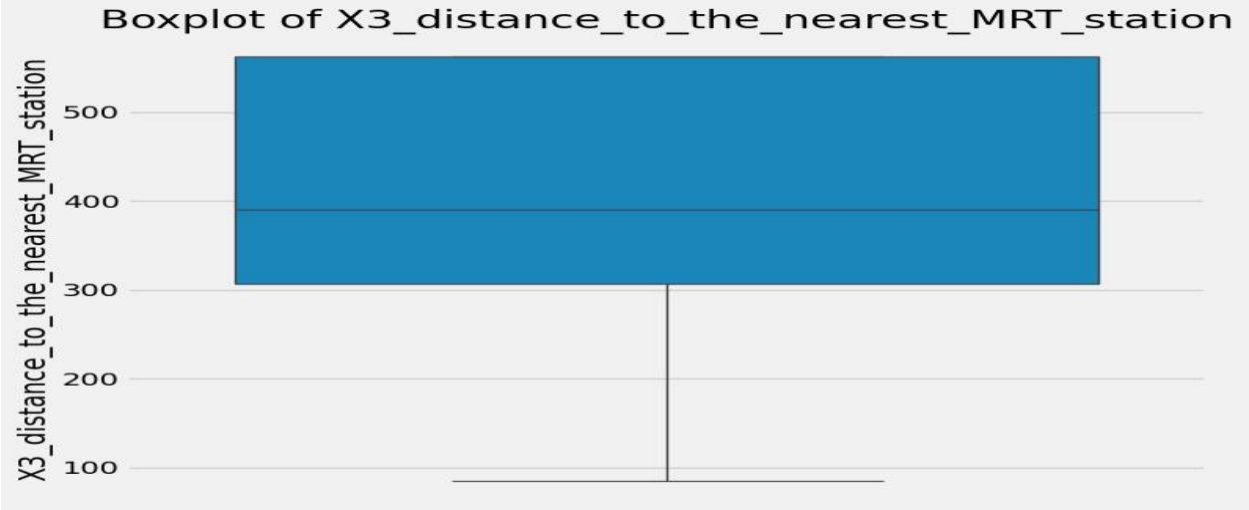
```
dt.columns = dt.columns.str.replace(' ', '_')
column_to_check=['X1_transaction_date','X2_house_age','X3_distance_to_the_nearest_MRT_station','X4_number_of_convenience_stores','X5_latitude','X6_longitude','Y_house_price_of_unit_area']
outliers_dt=pd.DataFrame()
for column_name in column_to_check:
  Q1=dt['X3_distance_to_the_nearest_MRT_station'].quantile(0.05)
  Q3=dt['X3_distance_to_the_nearest_MRT_station'].quantile(0.95)
  IQR=Q3-Q1
  lower_limit=Q1-1.5*IQR
  upper_limit=Q3+1.5*IQR
  column_outliers=dt[(dt[column_name]<lower_limit)|(dt[column_name]>upper_limit)] # Now this line should work without error.
  column_outliers['Outlier_Column']=column_name
  outliers_dt=pd.concat([outliers_dt,column_outliers])
  outliers_dt.reset_index(drop=True,inplace=True)
  print("outliers in the column '{}':".format(column_name))
  print("All Outliers:")
  print(outliers_dt)
  plt.figure(figsize=(8,6))
  sns.boxplot(dt[column_name])
  plt.title('Boxplot of {}'.format(column_name))
  plt.tight_layout()
  plt.show()
```

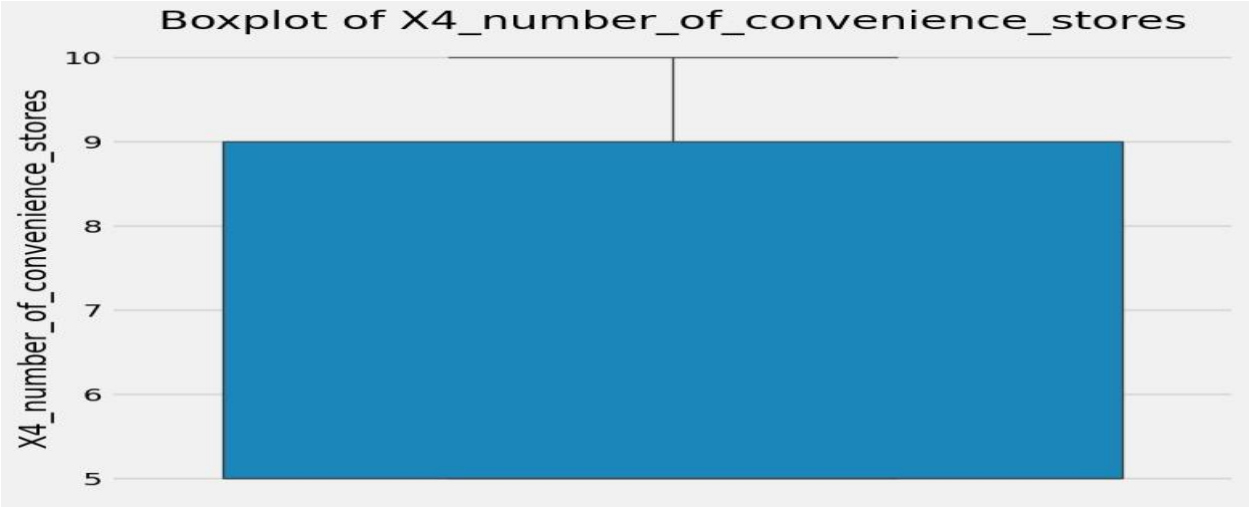
```
outliers in the column 'Y_house_price_of_unit_area':
All Outliers:
   X1_transaction_date  X2_house_age  X3_distance_to_the_nearest_MRT_station  \
0             2012.917          32.0                               84.87882
1             2012.917          19.5                              306.59470
2             2013.583          13.3                              561.98450
3             2013.500          13.3                              561.98450
4             2012.833           5.0                              390.56840

   X4_number_of_convenience_stores  X5_latitude  X6_longitude  \
0                               10     24.98298     121.54024
1                                9     24.98034     121.53951
2                                5     24.98746     121.54391
3                                5     24.98746     121.54391
```

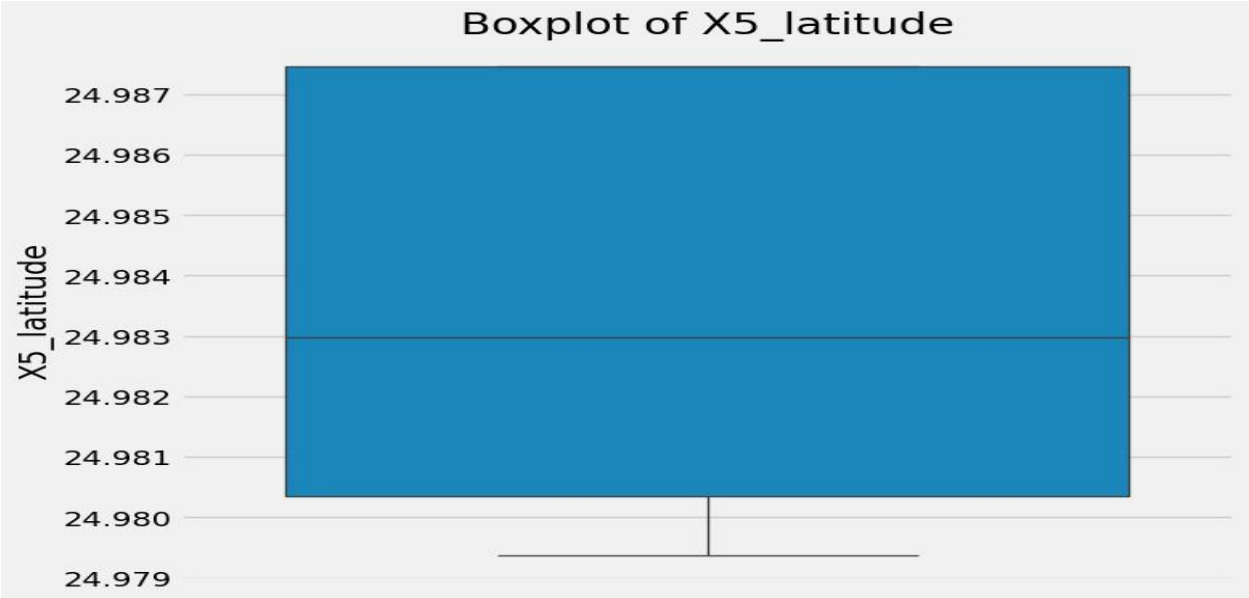
Connected to Python 3 Google Compute Engine backend

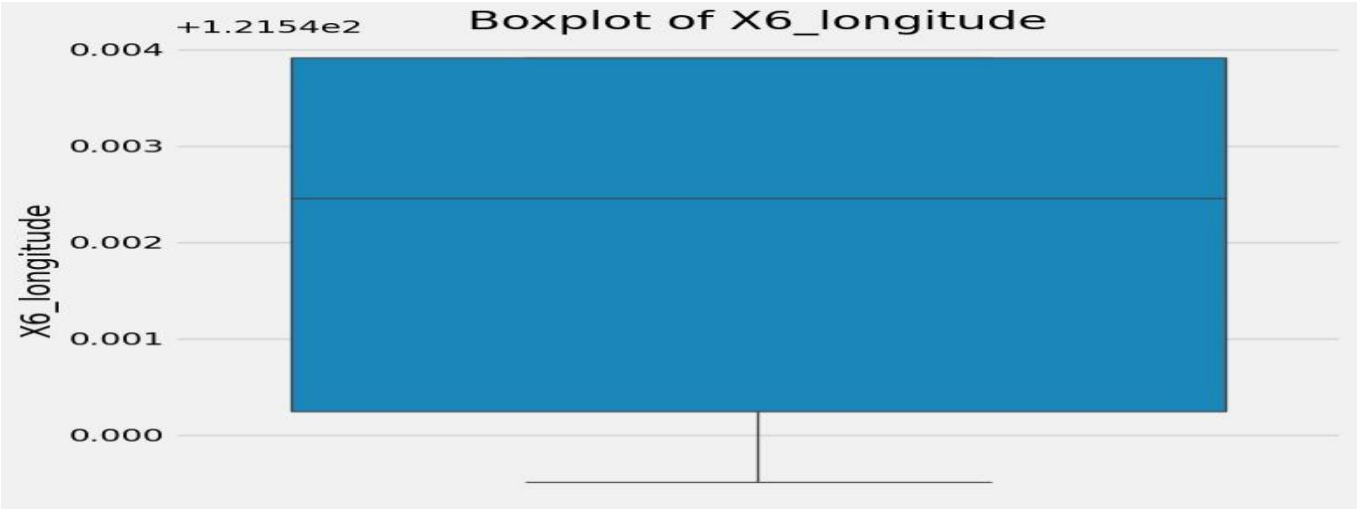

Boxplot of X1_transaction_date
X1_transaction_date
2013.6
2013.5
2013.4
2013.3
2013.2
2013.1
2013.0
2012.9
2012.8

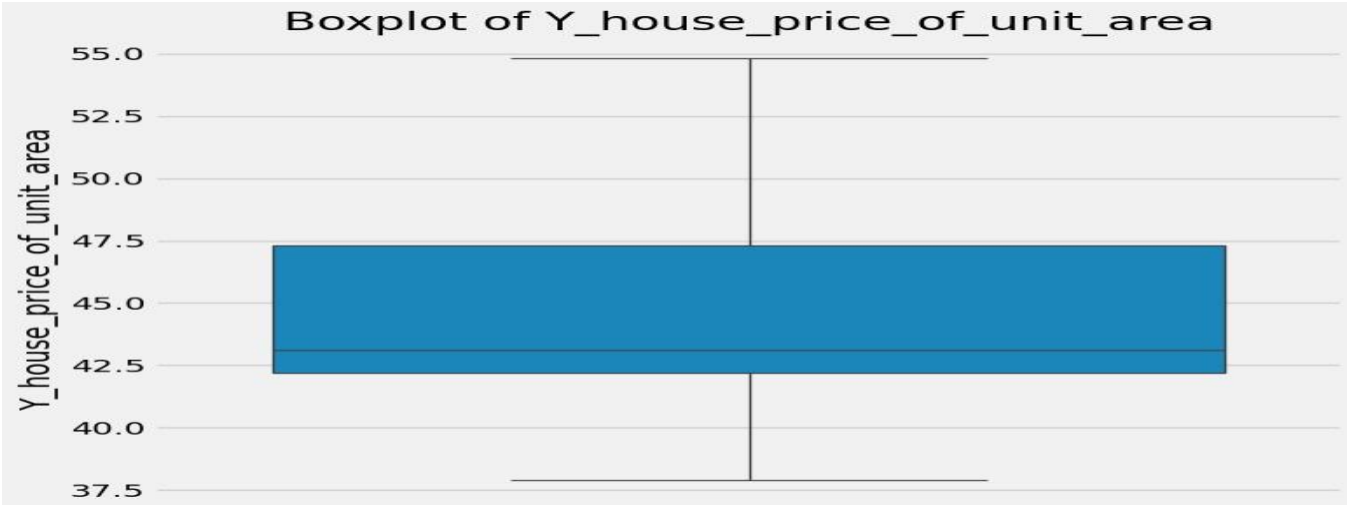

Boxplot of X2_house_age
X2_house_age
30
25
20
15
10
5


Boxplot of X3_distance_to_the_nearest_MRT_station
X3_distance_to_the_nearest_MRT_station
500
400
300
200
100


Boxplot of X4_number_of_convenience_stores
X4_number_of_convenience_stores
10
9
8
7
6
5


Boxplot of X5_latitude
X5_latitude
24.987
24.986
24.985
24.984
24.983
24.982
24.981
24.980
24.979


Boxplot of X6_longitude
+1.2154e2
X6_longitude
0.004
0.003
0.002
0.001
0.000



House Price Pred - Colab

https://colab.research.google.com/drive/1L3qxtUEUDg8MbgRM3hPXoB20xtKXjpqa#scrollTo=IgXfJPxBfqDh

House Price Pred

File Edit View Insert Runtime Tools Help Save failed

+ Code + Text

```
data = {
        "X1 transaction date": [2012.917, 2012.917, 2013.583, 2013.500, 2012.833],
        "X2 house age": [32.0, 19.5, 13.3, 13.3, 5.0],
        "X3 distance to the nearest MRT station": [84.87882, 306.59470, 561.98450, 561.98450, 390.56840],
        "X4 number of convenience stores": [10, 9, 5, 5, 5],
        "X5 latitude": [24.98298, 24.98034, 24.98746, 24.98746, 24.97937],
        "X6 longitude": [121.54024, 121.53951, 121.54391, 121.54391, 121.54245],
        "Y house price of unit area": [37.9, 42.2, 47.3, 54.8, 43.1]
    }
    dt = pd.DataFrame(data)

    # Define a function to remove outliers based on IQR
    def remove_outliers_iqr(df):
        Q1 = df.quantile(0.25)
        Q3 = df.quantile(0.75)
        IQR = Q3 - Q1
        return df[~((df < (Q1 - 1.5 * IQR)) | (df > (Q3 + 1.5 * IQR))).any(axis=1)]

    # Remove outliers
    dt_no_outliers = remove_outliers_iqr(dt)

    print("Original dataset size:", dt.shape)
    print("Dataset size after removing outliers:", dt_no_outliers.shape)
    print("Number of rows removed:", dt.shape[0] - dt_no_outliers.shape[0])
```

```
Original dataset size: (5, 7)
Dataset size after removing outliers: (4, 7)
Number of rows removed: 1
```

```
[333] sns.boxplot(data=dt,x='X3 distance to the nearest MRT station')
```

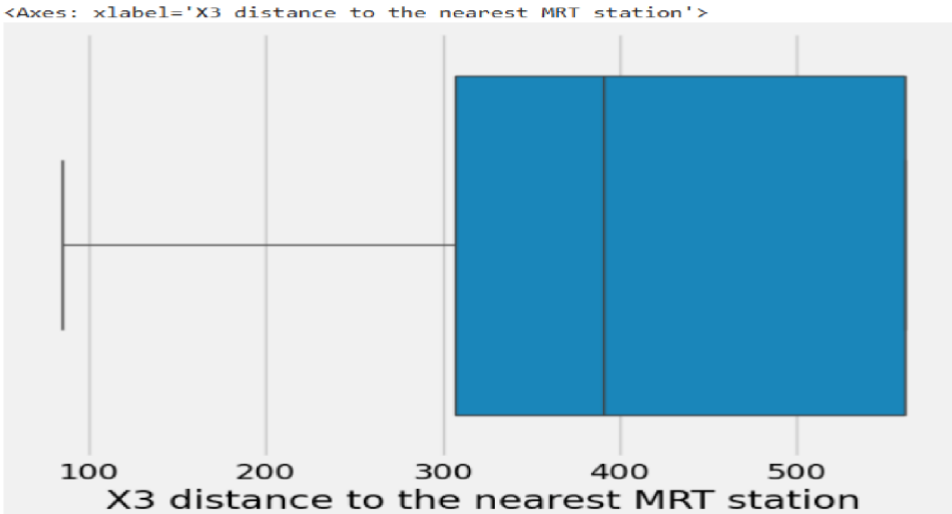
```
<Axes: xlabel='X3 distance to the nearest MRT station'>
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

Connected to Python 3 Google Compute Engine backend

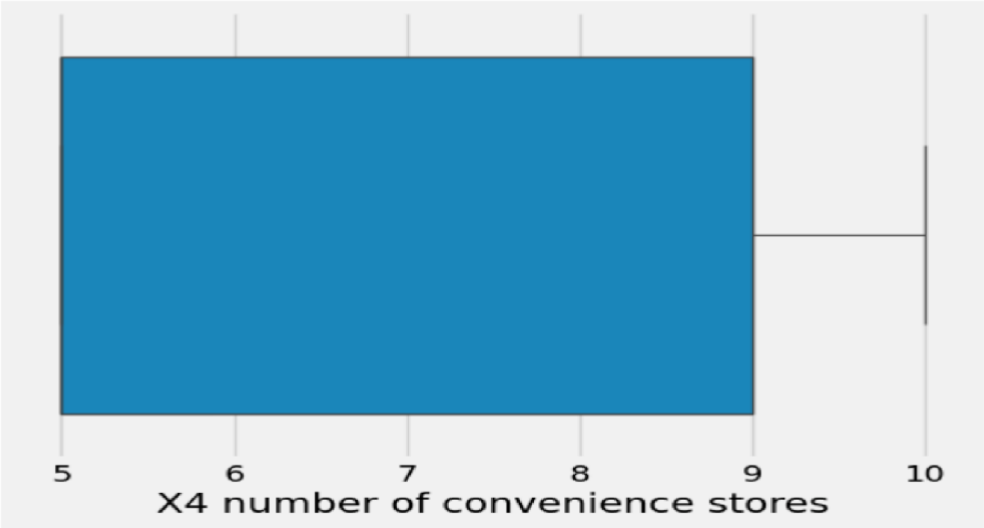
```
sns.boxplot(data=dt,x='X3 distance to the nearest MRT station')
```

```
<Axes: xlabel='X3 distance to the nearest MRT station'>
```



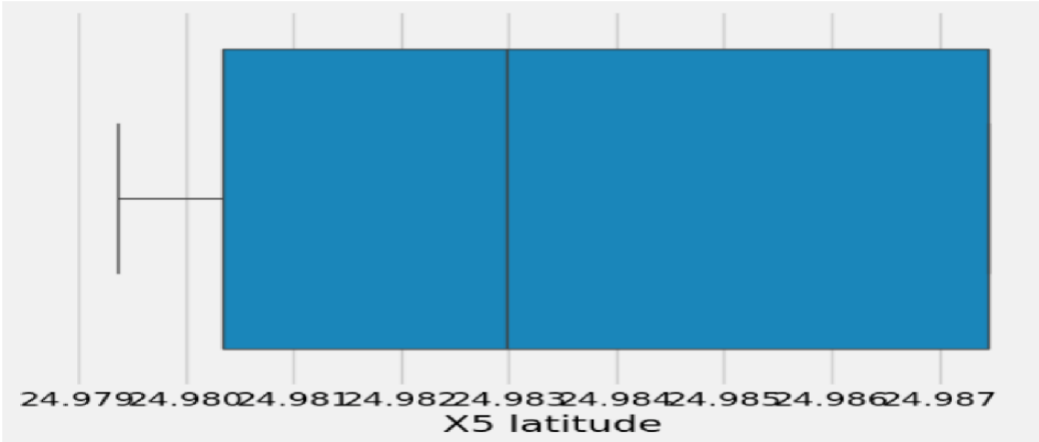
```
sns.boxplot(data=dt,x='X4 number of convenience stores')
```

```
<Axes: xlabel='X4 number of convenience stores'>
```



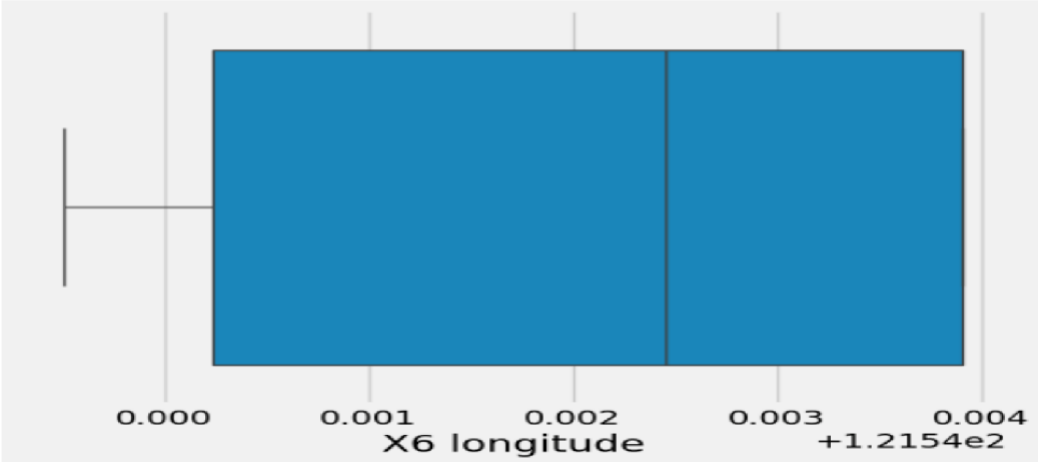
```
sns.boxplot(data=dt,x='X5 latitude')
```

```
<Axes: xlabel='X5 latitude'>
```



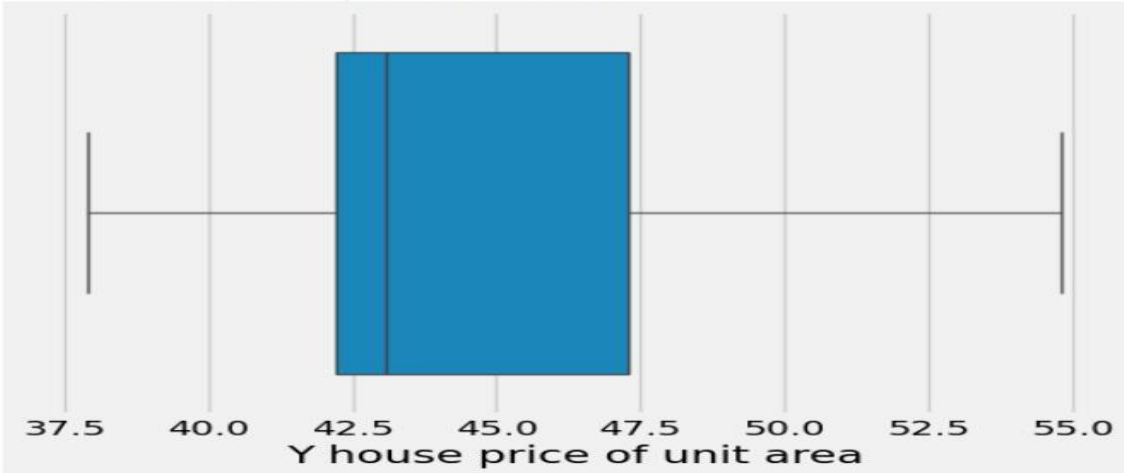

```
sns.boxplot(data=dt,x='X6 longitude')
```

```
<Axes: xlabel='X6 longitude'>
```



```
sns.boxplot(data=dt,x='Y house price of unit area')
```

```
<Axes: xlabel='Y house price of unit area'>
```




```
[417] X=dt.drop('Y house price of unit area',axis=1)
      Y=dt['Y house price of unit area']
```

```
[341] x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```

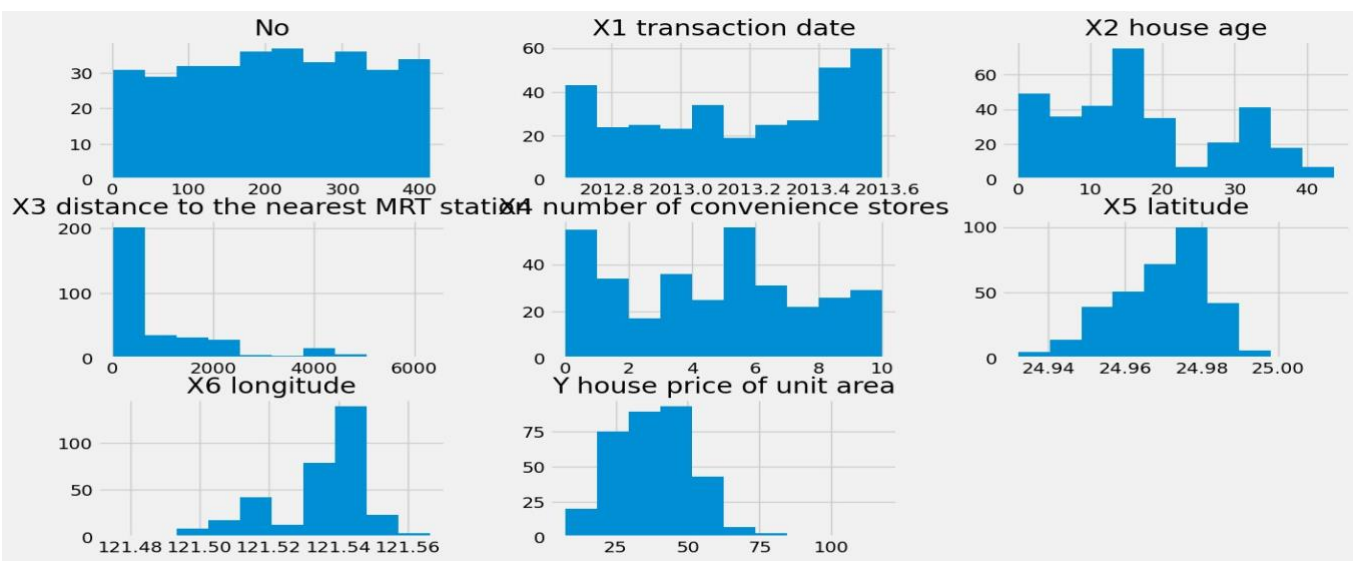
```
[342] train_dt=x_train.join(y_train)
      train_dt
```

	No	X1 transaction date	X2 house age	X3 distance to the nearest MRT station	X4 number of convenience stores	X5 latitude	X6 longitude	Y house price of unit area
192	193	2013.167	43.8	57.58945	7	24.96750	121.54069	42.7
234	235	2013.250	8.0	2216.61200	4	24.96007	121.51361	23.9
5	6	2012.667	7.1	2175.03000	3	24.96305	121.51254	32.1
45	46	2013.083	36.6	488.81930	8	24.97015	121.54494	38.3
245	246	2013.417	7.5	639.61980	5	24.97258	121.54814	40.8
...	...	...	...	...	...	...	...	...
71	72	2013.083	35.5	640.73910	3	24.97563	121.53715	40.8
106	107	2013.083	17.2	189.51810	8	24.97707	121.54308	47.1
270	271	2013.333	10.8	252.58220	1	24.97460	121.53046	117.5
348	349	2012.833	4.6	259.66070	6	24.97585	121.54516	53.7
102	103	2013.083	1.1	193.58450	6	24.96571	121.54089	54.4

331 rows × 8 columns

```
[418] train_dt.hist(figsize=(15,8))
      plt.show()
```

Connected to Python 3 Google Compute Engine backend



```
[429] x_train = x_train.dropna()
      y_train = y_train.loc[x_train.index]
```

```
[430] dt.dropna(inplace=True)
```

```
X1_transaction_date=['2012.917','2012.917','2013.583','2013.500','2012.833']
X2_house_age=['32.0','19.5','13.3','13.3','5.0']
X3_distance_to_the_nearest_MRT_station=['84.87882','306.59470','561.98450','561.98450','390.56840']
X4_number_of_convenience_stores=['10','9','5','5','5']
X5_latitude=['24.98298','24.98034','24.98746','24.98746','24.97937']
X6_longitude=['121.54024','121.53951','121.54391','121.54391','121.54245']
dt = pd.DataFrame({
    'X1_transaction_date': X1_transaction_date,
    'X2_house_age': X2_house_age,
    'X3_distance_to_the_nearest_MRT_station': X3_distance_to_the_nearest_MRT_station,
    'X4_number_of_convenience_stores': X4_number_of_convenience_stores,
    'X5_latitude': X5_latitude,
    'X6_longitude': X6_longitude
})
numerical_features=dt[['X1_transaction_date','X2_house_age','X3_distance_to_the_nearest_MRT_station','X4_number_of_convenience_stores','X5_latitude','X6_longitude']]
X=dt.values
print("feature matrix")
print(X)
```

```
feature matrix
[['2012.917' '32.0' '84.87882' '10' '24.98298' '121.54024']
 ['2012.917' '19.5' '306.59470' '9' '24.98034' '121.53951']
 ['2013.583' '13.3' '561.98450' '5' '24.98746' '121.54391']
 ['2013.500' '13.3' '561.98450' '5' '24.98746' '121.54391']
 ['2012.833' '5.0' '390.56840' '5' '24.97937' '121.54245']]
```

```
32] mean=train_dt['X3 distance to the nearest MRT station'].mean()
    std=train_dt['X3 distance to the nearest MRT station'].std()
    z_scores=[(x-mean)/std for x in train_dt['X3 distance to the nearest MRT station']]
    print(z_scores)
```

```
[-1.4870824572785777, -0.37441354316594566, 0.9072460810406845, 0.9072460810406845, 0.04700383836315336]
```

```
33] x_train,y_train=train_dt.drop('Y house price of unit area',axis=1),train_dt['Y house price of unit area']
```

```
34] from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=20)
```

```
36] from sklearn.linear_model import LinearRegression
    from sklearn.preprocessing import StandardScaler

    # Assuming x_train, x_test, y_train are defined earlier
    scaler = StandardScaler()
    x_train_scaled = scaler.fit_transform(x_train)
    x_test_scaled = scaler.transform(x_test)

    # Initialize and fit the Linear Regression model
    lr = LinearRegression()
    lr.fit(x_train_scaled, y_train)

    # Predict using the fitted model
    y_pred_lr = lr.predict(x_test_scaled)
```

```
37] y_pred_lr
```

```
array([44.30082882, 51.92008849, 27.44109342, 43.67836072, 43.71576816,
```

House Price Pred - Colab | New tab

https://colab.research.google.com/drive/1L3qxtUEUDg8MbgRM3hPXoB20xtKXjpqa#scrollTo=CtlLcBLYazOi

House Price Pred

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

```
[438] # Import necessary libraries
      import numpy as np
      import pandas as pd
      from sklearn.model_selection import train_test_split
      from sklearn.linear_model import LinearRegression
      from sklearn.metrics import mean_squared_error, r2_score

      # Assuming x and y are your feature matrix and target vector respectively
      # Split the data into training and testing sets
      x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=20)

      # Initialize the Multiple Linear Regression model
      mlr = LinearRegression()

      # Train the model
      mlr.fit(x_train, y_train)

      # Make predictions on the test set
      y_pred_mlr = mlr.predict(x_test)
```

```
from sklearn.tree import DecisionTreeRegressor
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=20)
```

```
[440] dtr=DecisionTreeRegressor(random_state=20)
      dtr.fit(x_train,y_train)
```

```
DecisionTreeRegressor
DecisionTreeRegressor(random_state=20)
```

```
[441] y_pred_dtr=dtr.predict(x_test)
```

```
[442] from sklearn.ensemble import RandomForestRegressor
```

Connected to Python 3 Google Compute Engine backend

House Price Pred - Colab | New tab

https://colab.research.google.com/drive/1L3qxtUEUDg8MbgRM3hPXoB20xtKXjpqa#scrollTo=CtILcBLYazOi

House Price Pred

File Edit View Insert Runtime Tools Help All changes saved

```
[442] from sklearn.ensemble import RandomForestRegressor
      x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=20)
      rf=RandomForestRegressor(n_estimators=20,random_state=80)
      rf.fit(x_train,y_train)
```

```
RandomForestRegressor
RandomForestRegressor(n_estimators=20, random_state=80)
```

```
[443] y_pred_rf=rf.predict(x_test)
```

```
import xgboost as xg
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=20)
xg=xg.XGBRegressor(objective='reg:linear',n_estimators=50,seed=23)
```

```
[445] xg.fit(x_train,y_train)
```

```
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=50, n_jobs=None,
             num_parallel_tree=None, objective='reg:linear', ...)
```

```
[446] y_pred_xg=xg.predict(x_test)
```

Connected to Python 3 Google Compute Engine backend

House Price Pred - Colab | New tab

https://colab.research.google.com/drive/1L3qxtUEUDg8MbgRM3hPXoB20xtKXjpqa#scrollTo=CtILcBLYazOi

House Price Pred

File Edit View Insert Runtime Tools Help All changes saved

```
from sklearn.ensemble import GradientBoostingRegressor
```

```
[448] gbr=GradientBoostingRegressor(n_estimators=18,max_depth=3,learning_rate=1)
```

```
[449] x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=20)
```

```
[450] gbr.fit(x_train,y_train)
```

```
GradientBoostingRegressor
GradientBoostingRegressor(learning_rate=1, n_estimators=18)
```

```
[451] y_pred_gbr=gbr.predict(x_test)
```

```
from sklearn.ensemble import AdaBoostRegressor
```

```
[453] adr=AdaBoostRegressor(n_estimators=10,learning_rate=1,random_state=20)
      x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=20)
```

```
[454] adr.fit(x_train,y_train)
```

```
AdaBoostRegressor
AdaBoostRegressor(learning_rate=1, n_estimators=10, random_state=20)
```

```
[455] y_pred_adr=adr.predict(x_test)
```

```
[456] print("The accuracy of Linear Regression:",r2_score(y_pred_lr,y_test))
      print("The accuracy of multilinear regression:",r2_score(y_pred_mlr,y_test))
      print("The accuracy of Decision Tree Regression:",r2_score(y_pred_dtr,y_test))
```

Connected to Python 3 Google Compute Engine backend

House Price Pred - Colab | New tab

https://colab.research.google.com/drive/1L3qxtUEUDg8MbgRM3hPXoB20xtKXjpqa#scrollTo=CtILcBLYazOi

House Price Pred

File Edit View Insert Runtime Tools Help All changes saved

```
print("The accuracy of Linear Regression:",r2_score(y_pred_lr,y_test))
print("The accuracy of multilinear regression:",r2_score(y_pred_mlr,y_test))
print("The accuracy of Decision Tree Regression:",r2_score(y_pred_dtr,y_test))
print("The accuracy of Random Forest Regression",r2_score(y_pred_rf,y_test))
print("The accuracy of XGBoost Regression",r2_score(y_pred_xg,y_test))
print("The accuracy of Gradient Boosting Regression",r2_score(y_pred_gbr,y_test))
print("The accuracy of Adaboost Regression",r2_score(y_pred_adr,y_test))
```

```
The accuracy of Linear Regression: 0.4180466128748409
The accuracy of multilinear regression: 0.41804661287475964
The accuracy of Decision Tree Regression: 0.6970919103505941
The accuracy of Random Forest Regression 0.7882737310455957
The accuracy of XGBoost Regression 0.7337036671657219
The accuracy of Gradient Boosting Regression 0.5931610408263717
The accuracy of Adaboost Regression 0.689363473776685
```

```
[457] from sklearn.ensemble import RandomForestRegressor
      x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
      rf=RandomForestRegressor(n_estimators=20,random_state=80)
```

```
[458] rf.fit(x_train,y_train)
```

```
RandomForestRegressor
RandomForestRegressor(n_estimators=20, random_state=80)
```

```
[459] y_pred_rf=rf.predict(x_test)
```

```
[460] from sklearn.metrics import mean_squared_error, r2_score
      import numpy as np
      ac = r2_score(y_pred_rf, y_test)
      rms = np.sqrt(mean_squared_error(y_pred_rf, y_test))
      ms = mean_squared_error(y_pred_rf, y_test)
```

Connected to Python 3 Google Compute Engine backend

House Price Pred - Colab | New tab

https://colab.research.google.com/drive/1L3qxtUEUDg8MbgRM3hPXoB20xtKXjpqa#scrollTo=CtILcBLYazOi

House Price Pred

File Edit View Insert Runtime Tools Help All changes saved

```
from sklearn.metrics import mean_squared_error, r2_score
import numpy as np
ac = r2_score(y_pred_rf, y_test)
rms = np.sqrt(mean_squared_error(y_pred_rf, y_test))
ms = mean_squared_error(y_pred_rf, y_test)
ac,rms,ms
```

```
(0.7614396134445647, 6.036898364845136, 36.44414186746988)
```

```
[461] from sklearn.preprocessing import StandardScaler

      # Assuming you have your training data `train_data`
      sc = StandardScaler()
      sc.fit(train_dt)  # Fit the scaler to your training data
```

```
StandardScaler
StandardScaler()
```

```
[462] import pickle
      from sklearn.preprocessing import StandardScaler
      from sklearn.ensemble import RandomForestRegressor
      rf_model = RandomForestRegressor()
      scaler = StandardScaler()
      with open('price.pkl', 'wb') as f:
          pickle.dump(rf_model, f)
      with open('scale.pkl', 'wb') as f:
          pickle.dump(scaler, f)
```

```
[463] from google.colab import files
      files.download('price.pkl')
```

Connected to Python 3 Google Compute Engine backend