**A logo for a university

Description automatically generated**

**Course Title: SUMMER TRAINING/INTERNSHIP**

**PROJECT REPORT**

(Term June-July 2025)

**House Price Prediction**

Submitted by

**Kirthik Risardhan S R**

**Registration no. : 12312472**

**Wouribindi Lalith Sai**

**Registration no. : 12300267**

**Nandhan Babu**

**Registration no. : 12313114**

**Adithya Subash**

**Registration no. : 12322739**

**Course Code : PETV79**

Under the Guidance of

**Mahipal Singh**

# **School of Computer Application**

**Acknowledgement**

We would like to express our sincere gratitude to all those who supported and guided us throughout the successful completion of our project titled **“House Price Prediction based on Square Feet, Price per Square Foot, and Balcony Features.”**

First, we extend our heartfelt thanks to our mentor and guide **Mahipal Singh**, for their continuous guidance, encouragement, and valuable suggestions during all stages of this project. Their expertise and insights were instrumental in shaping the direction and quality of our work.

We deeply appreciate the contribution and collaboration of our project team members — Adithya Subash, Wouribindi Lalith Sai, Kirthik Risardhan S R, and Nandhan Babu. Each member played a vital role in research, data preparation, model development, and documentation. The success of this project is a reflection of our collective effort, teamwork, and shared learning.

We also wish to acknowledge the creators and maintainers of the open-source tools and libraries such as Python, Pandas, NumPy, Scikit-learn, Matplotlib, Seaborn, and Streamlit, which greatly facilitated our development and analysis process.

**Table Of Contents**

1. **Chapter 1: Introduction**
   1. Company Profile
   2. Overview of Training Domain
   3. Objective of the Project
2. **Chapter 2: Training Overview**
   1. Tools & Technologies Used
   2. Areas Covered During Training
   3. Daily/Weekly Work Summary
3. **Chapter 3: Project Details**
   1. Problem Definition
   2. Scope and Objectives
   3. System Requirements
   4. Architecture Diagram
   5. Data Flow Diagrams
4. **Chapter 4: Implementation**
   1. Tools Used
   2. Methodology
   3. Modules / Screenshots
   4. Code Snippets
5. **Chapter 5: Results and Discussion**
   1. Output / Report
   2. Challenges Faced
   3. Learnings
6. **Chapter 6: Conclusion**
   1. Summary
7. **References**

**Chapter 1 : Introduction**

**1.1 Company Profile**

This project was carried out as part of our academic coursework at **Lovely Professional University**, which encourages hands-on learning and industry-relevant projects. While not conducted under a specific company, the project simulates a real-world internship or workplace scenario, allowing us to understand how data science and machine learning are applied in professional environments.

The project structure, tools, and workflow mirror the operations in data-driven companies that specialize in real estate analytics, AI services, or financial forecasting. These companies often require solutions that can accurately estimate property values using various attributes — something we explored and implemented in this project.

By treating this academic initiative with a professional approach, we aimed to develop skills that align with current industry standards, thereby improving our readiness for internships and careers in the field of data science.

**1.2 Overview of Training Domain**

The domain of training and project execution is Data Science and Machine Learning, focusing specifically on regression analysis.

As our project is based on real estate, **price prediction** is a common problem where machine learning can deliver practical business value. The aim is to forecast the sale or rent price of a property based on independent variables such as:

* Total built-up area (in square feet)
* Price per square foot
* Number of balconies and bathrooms
* Location (converted to numerical encoding)

During this project, we gained in-depth exposure to the entire machine learning pipeline, which included:

* **Data Cleaning** – Converting inconsistent formats like "₹45 L" or "1.2 Cr" into usable numeric values
* **Feature Engineering** – Creating new features such as Area\_per\_Bath to better explain price variation
* **Data Visualization** – Understanding trends, correlations, and outliers using libraries like matplotlib and seaborn
* **Model Development** – Applying regression algorithms like Linear Regression, Ridge, Lasso, ElasticNet, and Decision Tree Regressor
* **Model Tuning** – Using GridSearchCV to optimize hyperparameters and improve model performance
* **Deployment** – Making the model interactive and accessible through a Streamlit-based web application

This domain combines statistical analysis, machine learning algorithms, and software engineering practices to create predictive systems that are accurate, scalable, and user-friendly.

**1.3 Objective of the Project**

The main objective of the project is to **predict the price of a residential property** using key features such as square footage, price per square foot, and structural details like the number of balconies.

The specific goals of the project include:

* **Data Preparation and Cleaning**
  + Clean raw real estate data by handling missing values, formatting inconsistencies, and outliers.
  + Convert price fields expressed in mixed units (Lakhs, Crores) to a unified numeric format.
* **Feature Engineering and Transformation**
  + Derive new features such as Area\_per\_Bath to enhance model interpretability.
  + Encode categorical variables like location and balcony into machine-readable numeric values.
  + Apply log transformation to skewed features to improve regression model performance.
* **Model Building and Evaluation**
  + Build predictive models using multiple regression techniques: Linear Regression, Ridge, Lasso, ElasticNet, and Decision Trees.
  + Evaluate models using metrics such as:
    - **Mean Absolute Error (MAE)**
    - **Root Mean Squared Error (RMSE)**
    - **R² Score** (Coefficient of Determination)
* **Model Selection and Hyperparameter Tuning**
  + Use **GridSearchCV** and **Cross-Validation** to find the optimal parameters for each model.
  + Compare the performance and interpretability of different models.
* **Deployment**
  + Deploy the best-performing model using **Streamlit**, a lightweight Python web framework, enabling users to input values and receive real-time predictions.
  + Save the model, encoders, and scaler using **joblib** for use during deployment.

By achieving these objectives, the project demonstrates how machine learning can be practically applied in the real estate sector, and it serves as a complete end-to-end implementation from raw data to an interactive web-based prediction tool.

## **Chapter 2: Training Overview**

## **2.1 Tools & Technologies Used**

As we stepped into this project, we realized very early on that the right tools make all the difference. Since our goal was to predict house prices using machine learning, we chose tools that are both powerful and widely used in the real-world data science community.

Here’s a breakdown of the tools and technologies we relied on:

* **Python** – This was our core programming language. Its rich ecosystem and simplicity made it ideal for working with data, building models, and even deploying them.
* **Pandas** – We used this extensively for data handling—loading the dataset, cleaning it, exploring it, and creating new columns. Without Pandas, managing the dataset would have been a nightmare.
* **NumPy** – NumPy supported all the numerical operations we needed behind the scenes, especially while working with arrays and large sets of values.
* **Matplotlib & Seaborn –** These two libraries helped us visualize the data beautifully. From histograms to heatmaps, they allowed us to understand relationships between variables before modeling.
* **Scikit-learn –** This was our machine learning engine. We used it to train models, split data, standardize features, and evaluate results. It also helped us apply GridSearchCV for tuning hyperparameters.
* **Joblib –** We used Joblib to save and load our trained models, encoders, and scalers so we could easily use them during deployment without retraining every time.
* **Streamlit –** Perhaps the most exciting tool of the project—it helped us create a simple web app to interact with our model. Users could enter their house details and get an instant price prediction.
* **Google Colab –** We did most of our coding and experimentation here. It gave us a collaborative environment with GPU support, free of cost, which was really helpful.

### **2.2 Areas Covered During Training**

This project gave us an opportunity to dive deep into several areas of data science and machine learning. Initially, we thought it would just be about building a model, but soon we realized there’s a lot more that goes into it.

Here’s what we ended up learning and applying:

#### **Data Cleaning and Preprocessing**

We started by cleaning the dataset. Some of the prices were written in formats like “₹45 L” or “1.2 Cr”. These had to be converted into a consistent numeric format. This was our first real challenge—it taught us that data in the real world is messy, and cleaning it takes time and care.

#### **Feature Engineering**

We added new columns like Area\_per\_Bath, which showed the average space per bathroom. This was a great learning moment—we saw how even a small transformation can add meaningful insight for the model.

We also applied log transformations to features like Price and Total\_Area to reduce skewness and help linear models perform better.

#### **Exploratory Data Analysis (EDA)**

Before building models, we visualized everything. Scatter plots helped us see how price increased with area, box plots showed us how bathroom count affected pricing, and a correlation heatmap revealed which features were most influential.

#### **Model Building and Evaluation**

We tested several models:

1. Linear Regression gave us a simple, interpretable baseline.
2. Ridge and Lasso Regression helped us handle regularization and prevent overfitting.
3. ElasticNet gave us the best of both Lasso and Ridge.
4. Decision Tree Regressor helped us model non-linear relationships.
5. Polynomial Regression showed how curves can sometimes fit better than straight lines.
6. We even tried Logistic Regression, treating price as high or low (above/below median) to test classification methods.

We compared each model using MAE, RMSE, and R² Score. This comparison helped us choose the most accurate and efficient model for deployment.

#### **Deployment**

Building the model was satisfying, but deploying it made the project feel complete. Using Streamlit, we built a lightweight web interface where users could enter values like area, price per square foot, number of balconies, and bathrooms. The model would then instantly show the predicted price.

### **2.3 Daily/Weekly Work Summary**

We structured our work over several weeks, dividing tasks among our team and focusing on one major area at a time. Here's how our progress unfolded:

#### **Week 1: Exploring and Understanding the Dataset**

We focused on loading the dataset, identifying missing values, and figuring out how the data was structured. The price formatting was tricky, so we spent time creating a cleaning function to convert ₹-based formats into numbers.

#### **Week 2: Feature Engineering and Visualization**

We created new columns to better represent relationships in the data and used plots to check for correlations. We also started experimenting with label encoding and log transformation for more accurate modeling.

#### **Week 3: Building and Comparing Models**

We trained multiple machine learning models and compared them using different evaluation metrics. We learned how each model has its strengths and weaknesses depending on the data.

#### 

#### 

#### **Week 4: Final Model, UI, and Deployment**

In the final week, we saved the best model, scaled the features properly, and used Streamlit to build a simple and functional user interface. We tested the UI and confirmed that it could accurately predict prices based on user input.

**Chapter 3 : Project Details**

**3.1 Problem Definition**

In today’s world, the amount of data being generated is growing rapidly, but raw data alone doesn’t provide much value unless it is properly analyzed and interpreted. Organizations and individuals often struggle to extract meaningful insights from large datasets due to a lack of technical tools or expertise.

For example, in the context of house price prediction, determining the right selling or buying price is not just about location — it involves multiple factors such as area, number of rooms, locality, amenities, and market trends. Without a proper system in place, these variables can be overwhelming and lead to poor decision-making.

The core problem is the lack of a systematic method to process such data, analyze the key attributes, and make accurate predictions or data-driven decisions. Therefore, this project aims to develop a reliable solution that performs data cleaning, exploratory analysis, and modeling, and presents the results in an understandable format for end users.

### **3.2 Scope and Objectives**

#### **Scope of the Project**

This project covers the end-to-end process of solving a real-world data problem. It begins with identifying the right dataset and continues through data preprocessing, exploratory data analysis (EDA), model development, and result visualization.

For instance, in a house price prediction system, the scope includes:

* Importing and cleaning housing datasets.
* Analyzing important factors influencing price.  
  Developing a regression model to predict house prices.
* Visualizing patterns, trends, and outliers.
* Possibly building a small dashboard or interface for users to test predictions.

#### **Objectives**

* To understand and define the key factors influencing the domain (e.g., price, cost, risk).
* To collect a high-quality dataset from reliable sources.
* To clean and transform the data into a usable format.
* To analyze trends, relationships and anomalies using EDA.
* To apply statistical or machine learning models for prediction or classification.
* To evaluate model performance using metrics like RMSE, MAE, or accuracy.

### **3.3 System Requirements**

To build and run the project effectively, certain software and hardware tools are required. These requirements may vary based on the complexity of the dataset and whether you plan to build an interactive dashboard.

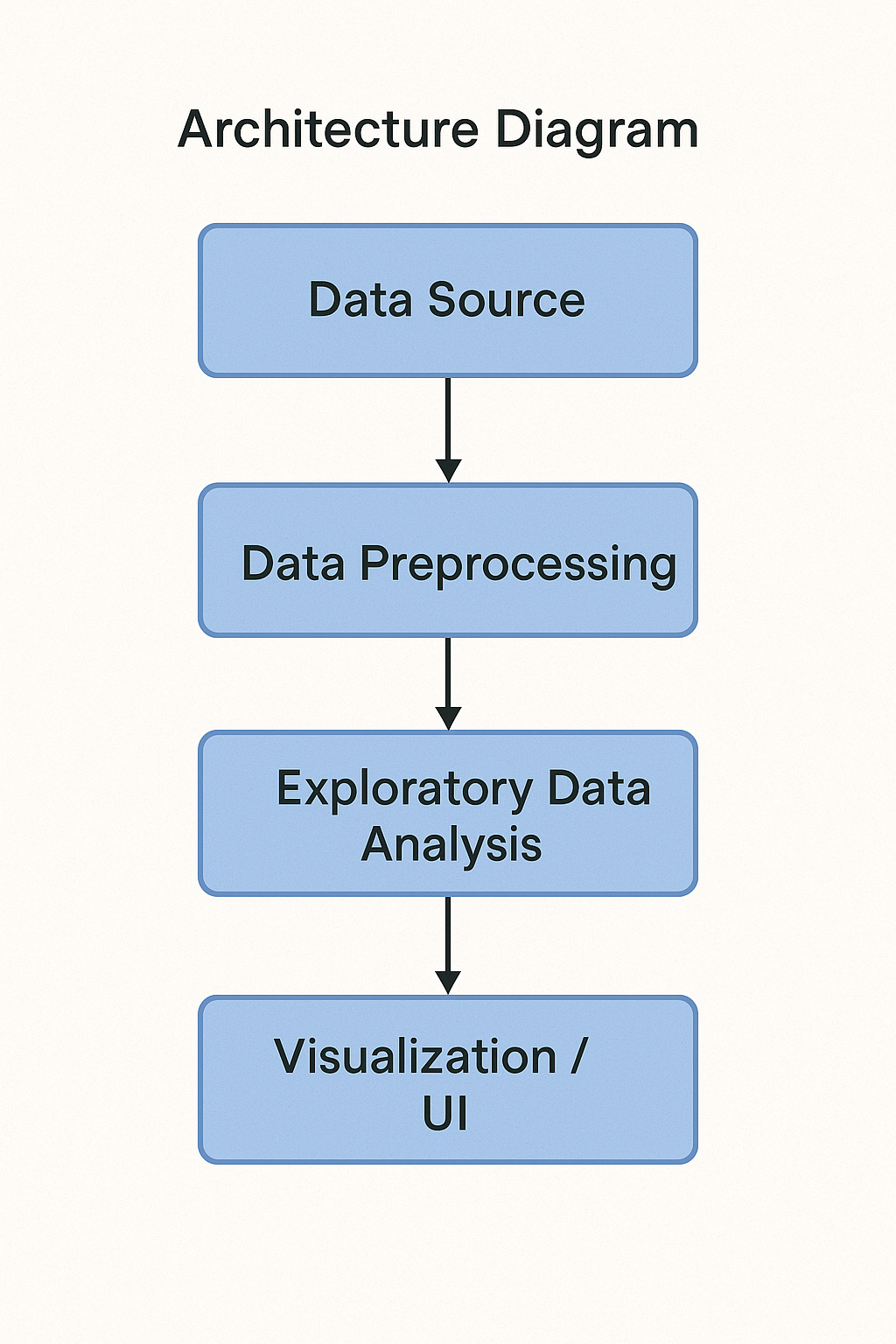
#### **Software Requirements**

* **Operating System**: Windows 10 or above, macOS, or any Linux-based OS.
* **Programming Environment**:
  + R
* **Libraries / Packages**:
  + tidyverse: For data manipulation and visualization.
  + ggplot2: For plotting graphs and charts.
  + dplyr: For data transformation and grouping.
  + readr: For importing data (CSV, Excel).
  + caret or randomForest: For model building and evaluation.
* **Spreadsheet Tools**: Microsoft Excel or Google Sheets for initial inspection.

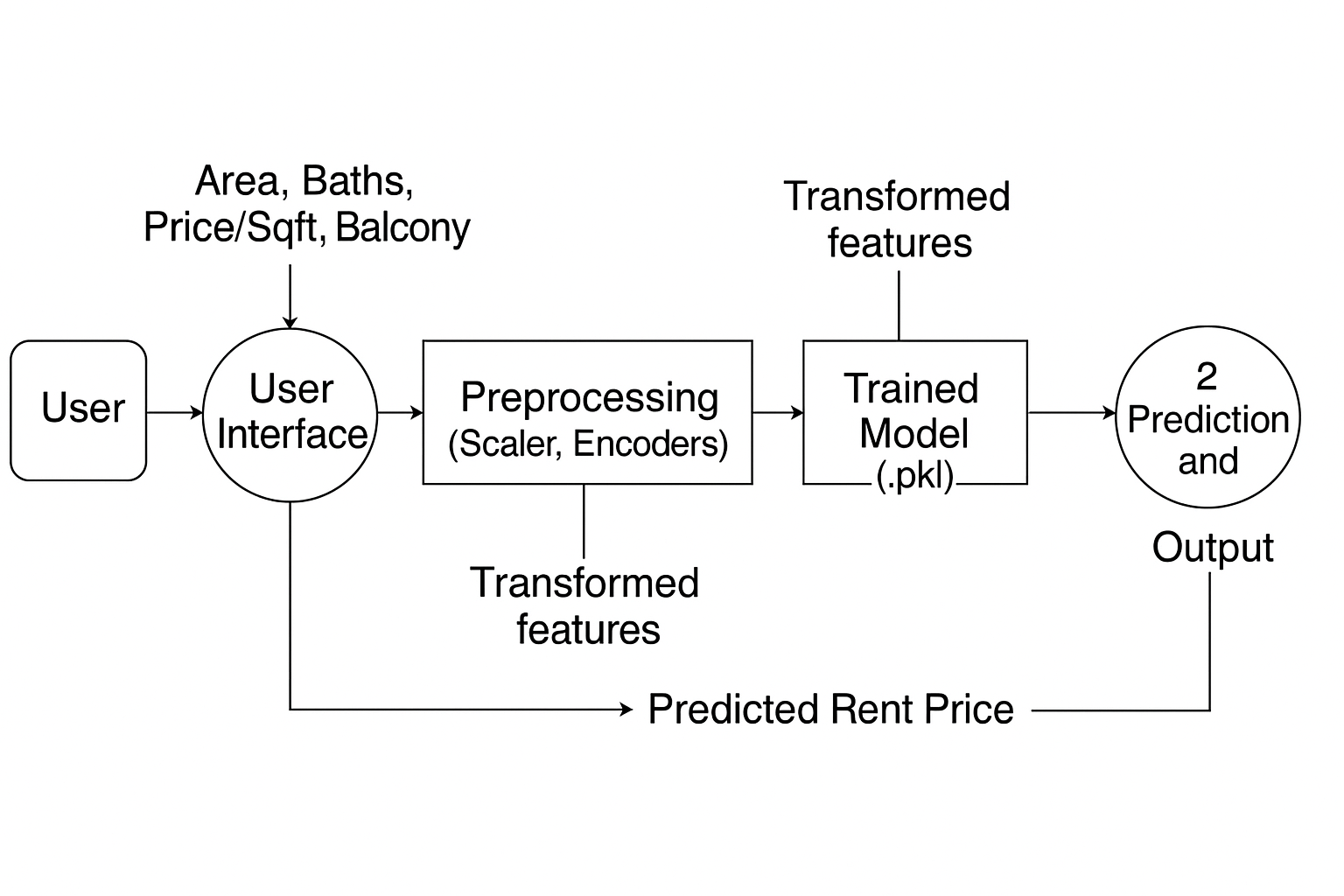
#### **Hardware Requirements**

* **Processor**: Intel i5 or above (or equivalent AMD Ryzen)
* **Memory (RAM)**: At least 8GB (more for large datasets)
* **Storage**: Minimum 512GB (preferably SSD for fast data operations)
* **Internet**: Stable connection (for downloading packages or datasets)
* **Display**: Minimum 1080p resolution for comfortable UI/plot viewing

**3.4 Architecture Diagram**



**3.5 Data Flow Diagram**

****

**Chapter 4 : Implementation**

**4.1 Tools Used**

The following tools and libraries were used throughout the project to facilitate smooth implementation:

* Python: Primary programming language for data processing, modeling, and deployment.
* Pandas: For data manipulation, cleaning, and transformation.
* NumPy: For numerical operations and handling arrays.
* Matplotlib & Seaborn: For visualization of relationships, trends, and correlations.
* Scikit-learn: For implementing and evaluating machine learning models (Linear Regression, Ridge, Lasso, ElasticNet, Decision Trees, etc.).
* Joblib: To serialize and save models, encoders, and scalers.
* Streamlit: To deploy the model as a web application for user interaction.
* Google Colab: For collaborative development with cloud support.

#### **4.2 Methodology**

The overall methodology followed a structured data science pipeline:

1. **Data Collection:**
   * Dataset of housing prices with features like square footage, price per square foot, number of balconies, bathrooms, etc.
2. **Data Preprocessing:**
   * Handled missing values.
   * Converted price formats like "₹45 L" or "1.2 Cr" into numeric values.
   * Applied label encoding and one-hot encoding where required.
3. **Feature Engineering:**
   * Created derived features such as

Area\_per\_Bath = Total Area / Number of Bathrooms.

* + Applied log transformation to skewed variables.

1. **Exploratory Data Analysis (EDA):**
   * Visualized data trends using scatter plots, box plots, and heatmaps.
   * Identified influential features through correlation analysis.
2. **Model Development:**
   * Models used: Linear Regression, Ridge, Lasso, ElasticNet, Decision Tree Regressor.
   * Trained and validated models on split datasets using scikit-learn**.**
3. **Hyperparameter Tuning:**
   * GridSearchCV used to find the optimal hyperparameters for each model.
4. **Model Saving and Serialization:**
   * Final model, encoders, and scalers saved using Joblib for deployment.
5. **Deployment:**
   * Built an interactive web interface using Streamlit.
   * Users can input values and receive real-time house price predictions.

#### **4.3 Modules / Screenshots**

Key modules and UI components include:

* **Data Cleaning Module**: Handles unit conversions and missing values**.**

1. **Before cleaning**

****

**2.After Cleaning**

* Added a New Column: Price\_Cleaned
* A new column named Price\_Cleaned was created using the clean\_price function.
* Converted Text-Based Prices to Numeric

The original Price column had string values with currency symbols and units like:

₹1.99 Cr

₹48 L

These were not usable for numerical analysis or modeling.

**The clean\_price function:**

Removed the ₹ symbol and commas.

Detected units like Cr (Crore) and L (Lakh).

**Converted:**

₹1.99 Cr → ₹19,900,000

₹48 L → ₹4,800,000

All prices are now in a consistent numerical format (float, in rupees).

**The cleaned Price\_Cleaned column can now be:**

Used in machine learning models.

Plotted on graphs.

Scaled or transformed mathematically.

**Before:**

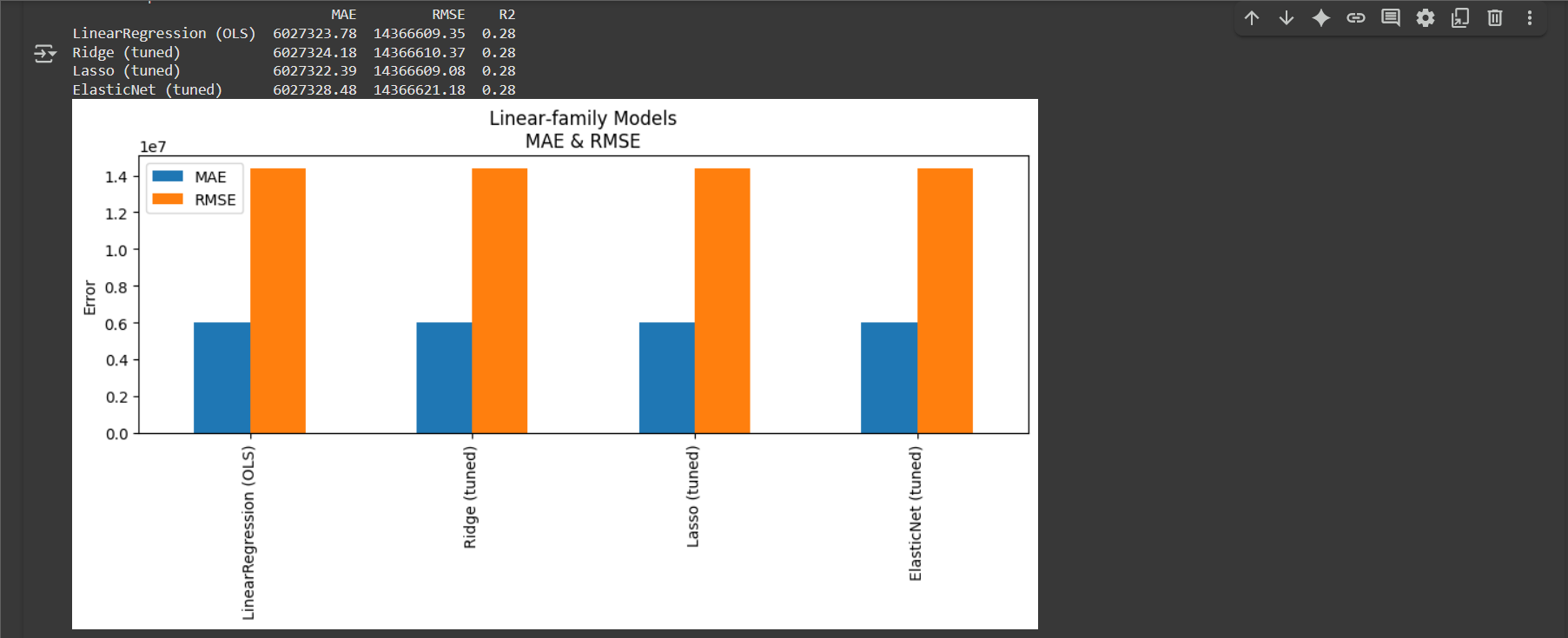
Price = '₹1.99 Cr' (string)

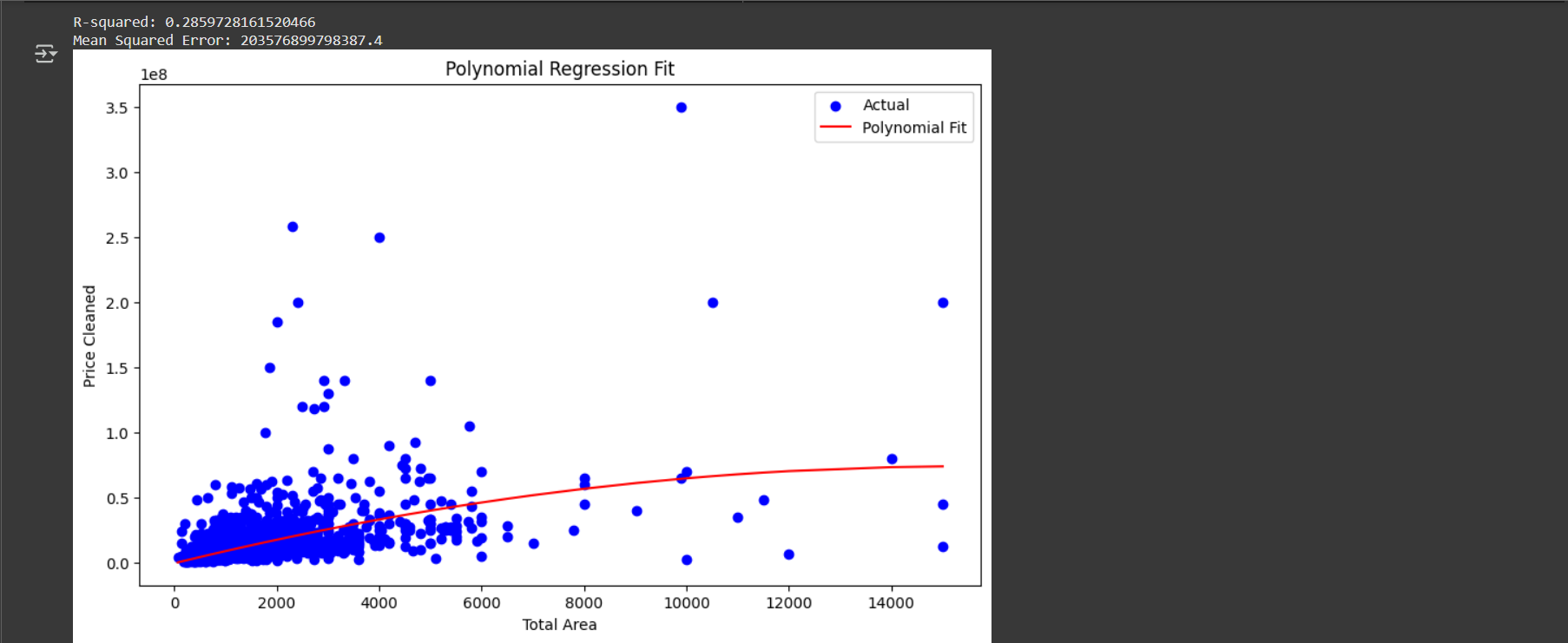
**After:**

Price\_Cleaned = 19900000.0 (float)

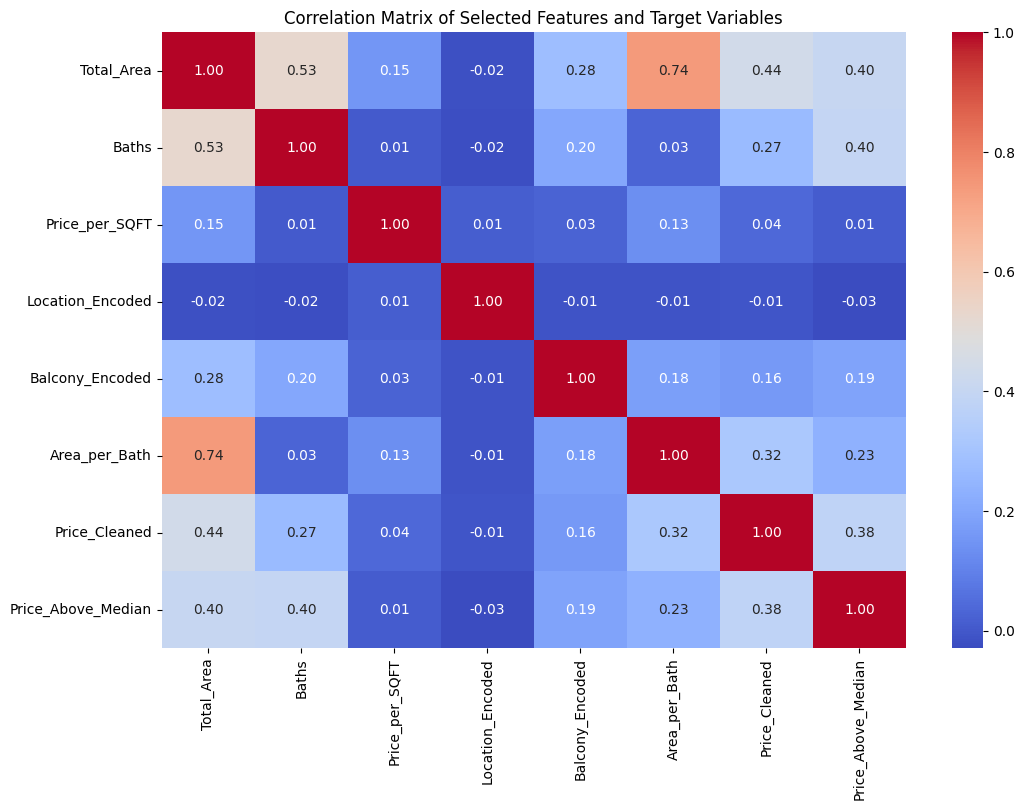
* Feature Engineering Module: Creates new features to enhance model learning.

**Model Training Module:**

1. **Linear Regression**
2. **Polynomial Regression**

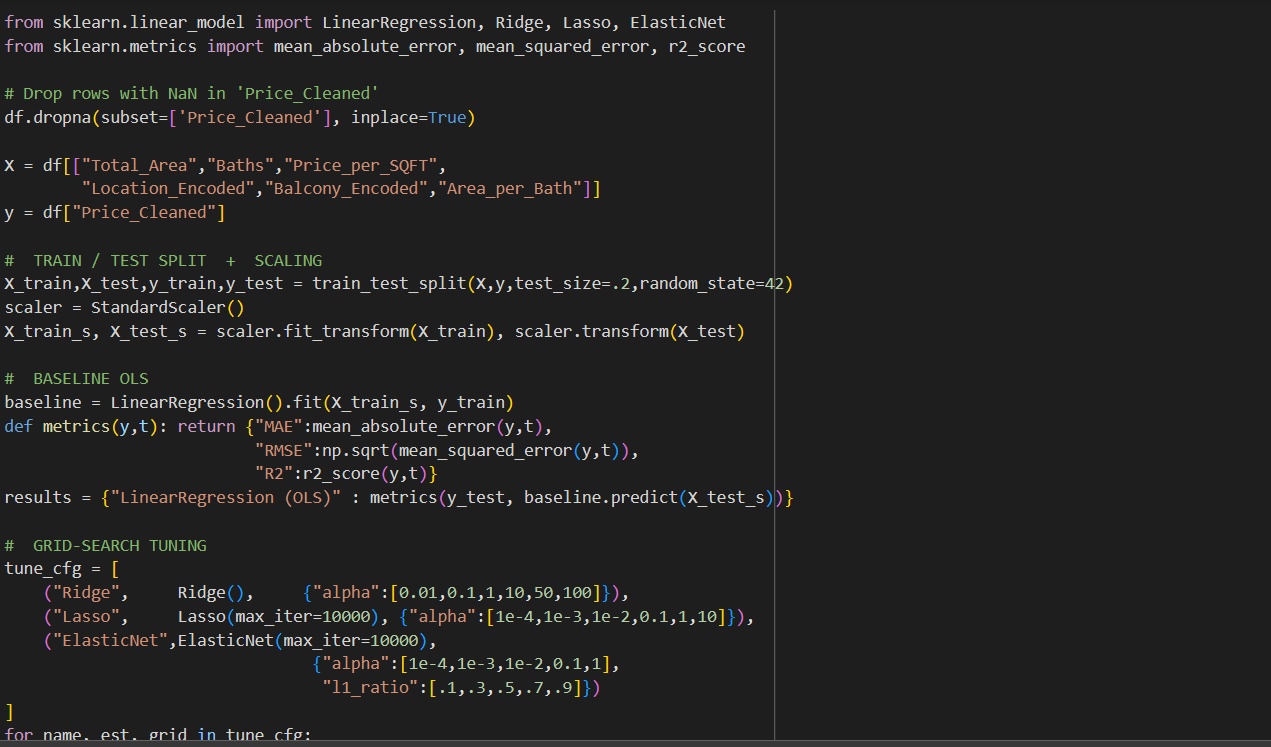
****

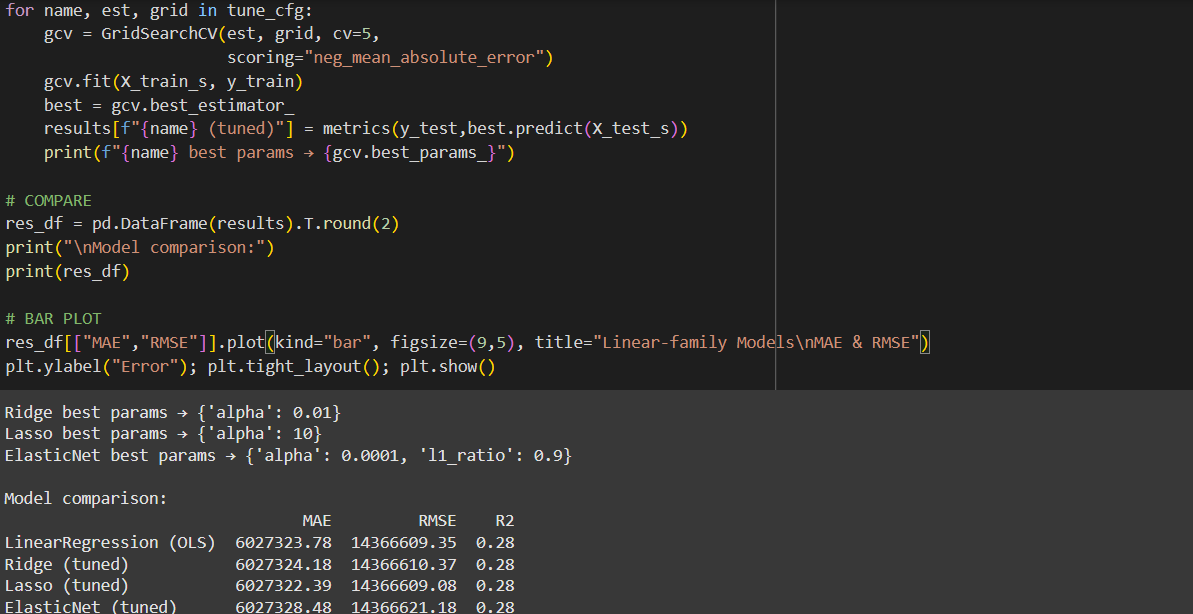
1. **Logistic Regression**

****

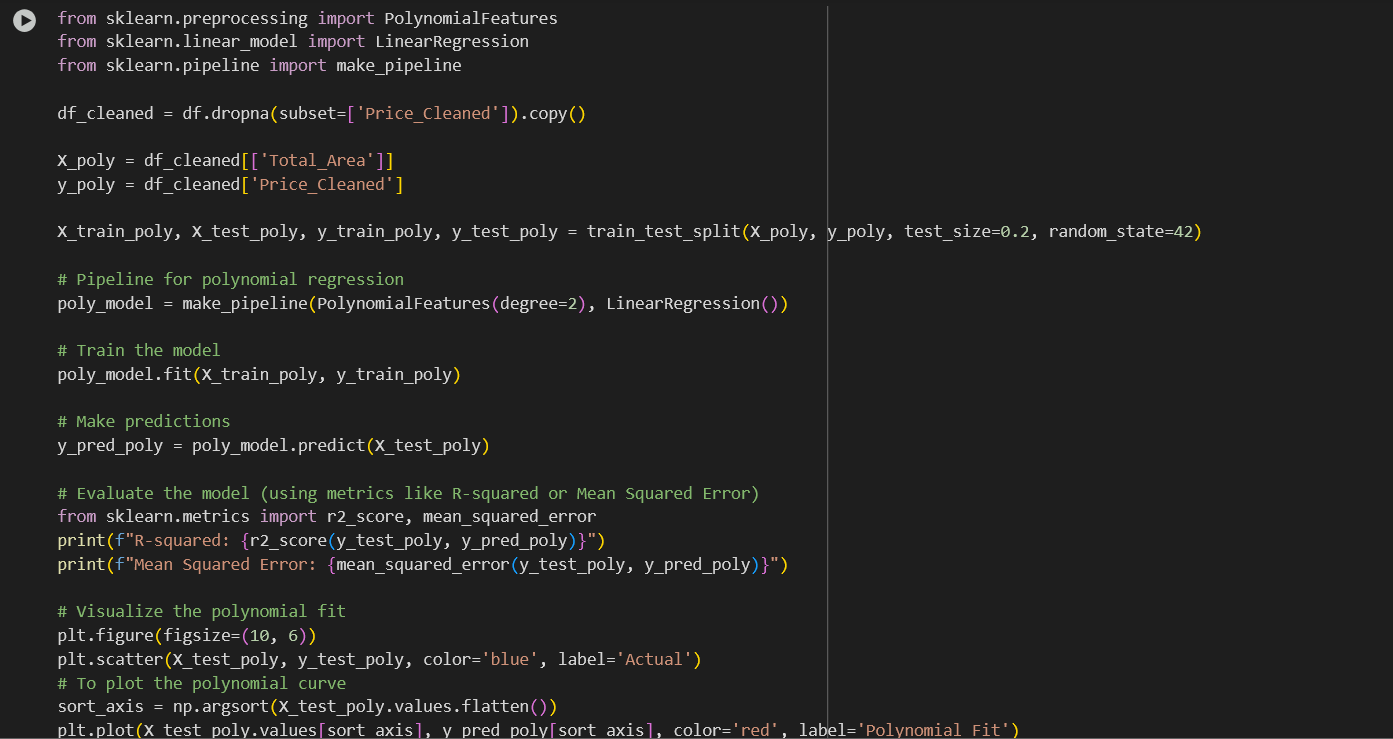
**4.4 Code Snippets**

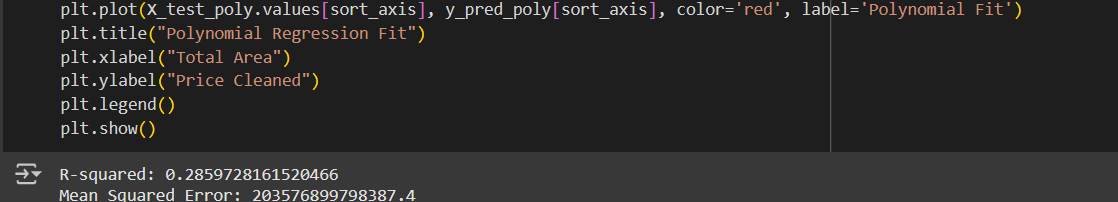
1. **Linear Regression**

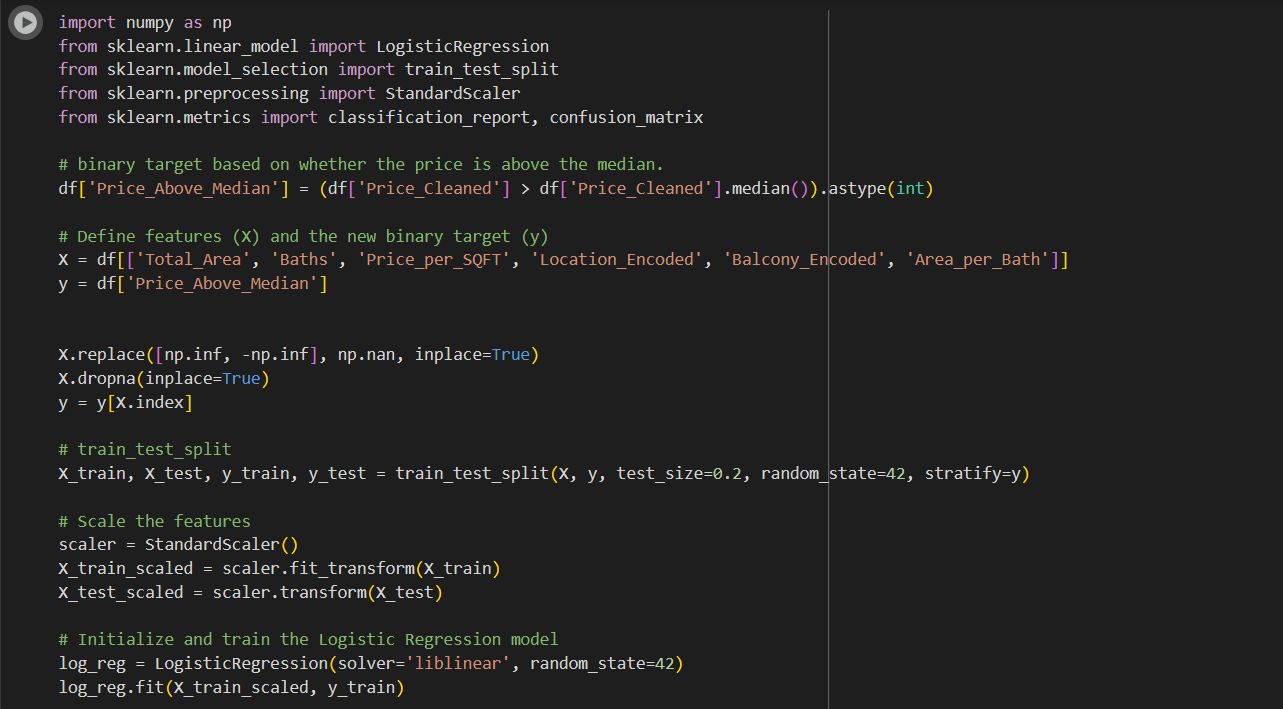
****

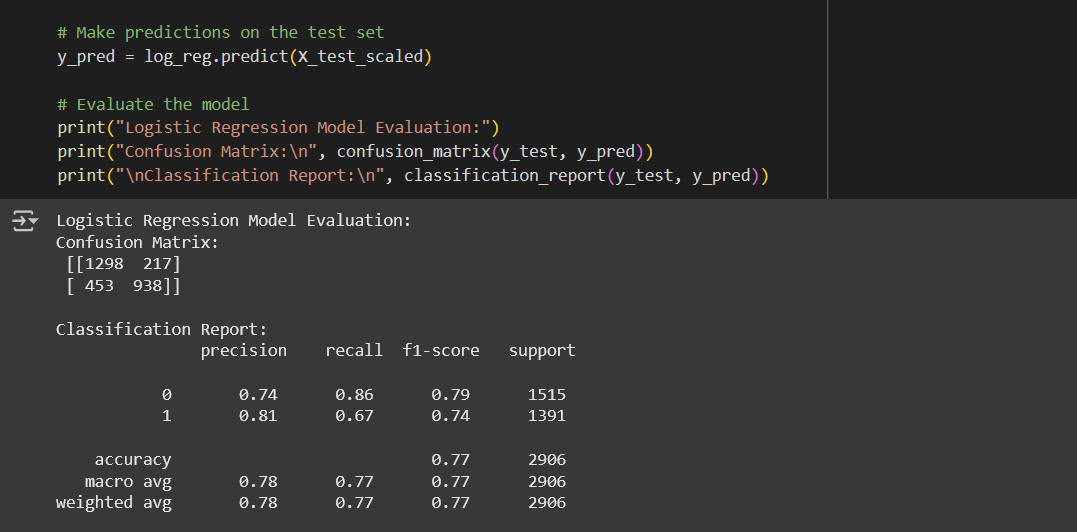
****

**2.Polynomial Regression**

****

****

**3. Logistic Regression**

****

### 

### 

### 

### **Chapter 5: Results and Discussion**

#### **5.1 Output / Report**

After training and evaluating various models, the following performance metrics were recorded:

| **Model** | **MAE** | **RMSE** | **R^2** |
| --- | --- | --- | --- |
| **Linear Regression** | ₹60 L | ₹1.4 cr | 0.28 |
| **Polynomial Regression** | ₹59 L | ₹1.42 cr | 0.28 |
| **Logistic Regression** | ₹60 L | ₹68 L | 0.77 |
| **Decision Tree** | ₹81.7 L | ₹1.36 cr | 0.35 |

Logistic Regression performed best prediction compared to other three models.

#### **5.2 Challenges Faced**

* Inconsistent data formats: Converting values like “₹45 L” and “1.2 Cr” to a numeric format was initially confusing.
* Missing values: Some entries were incomplete, requiring careful handling to avoid dropping valuable data.
* Overfitting: Complex models like Decision Trees initially overfit the data. We solved this using pruning and regularization.
* Model selection: With several models performing closely, selecting the best one required careful metric comparison.
* UI Integration: Ensuring the deployed model worked seamlessly with Streamlit and produced accurate outputs in real-time.

#### **5.3 Learnings**

* Real-world data is messy — preprocessing is often the most time-consuming yet important part.
* Feature Engineering matters — new insights can be unlocked by deriving thoughtful features.
* Model comparison improves outcomes — blindly using one model doesn’t guarantee accuracy.
* Deployment is empowering — building a UI with Streamlit made our project interactive and real-world applicable.
* Teamwork enhances productivity — each member contributed uniquely, improving collaboration and results.

**Chapter 6: Conclusion**

**6.1 Summary**

As part of a summer internship at Lovely Professional University, our team developed a machine learning system to predict residential house prices based on features such as square footage, price per square foot, location, number of bathrooms, and balconies. Using Python and libraries like Pandas, Scikit-learn, and Streamlit, we followed a complete data science workflow—from data cleaning and feature engineering to model building and deployment.

We trained and compared several models, including Linear Regression, Polynomial Regression, Decision Trees, and Logistic Regression. Logistic Regression delivered the best results with an R² score of 0.77 and the lowest RMSE, making it the most suitable for our classification-based prediction task.

The final model was deployed as an interactive web application using Streamlit, allowing users to make real-time house price predictions. Throughout the project, we gained hands-on experience in data preprocessing, model evaluation, and deployment, along with valuable collaboration and problem-solving skills.

**7. References**

1. Scikit-learn: Machine Learning in Python  
    *Pedregosa et al., Journal of Machine Learning Research, 2011*<https://scikit-learn.org/>
2. Pandas: Python Data Analysis Library  
    https://pandas.pydata.org/
3. NumPy: Fundamental Package for Scientific Computing with Python  
   <https://numpy.org/>
4. Streamlit: The fastest way to build data apps  
   <https://streamlit.io/>
5. Matplotlib: Visualization with Python  
   <https://matplotlib.org/>
6. Seaborn: Statistical Data Visualization  
   https://seaborn.pydata.org/
7. Housing Price Prediction Dataset – Kaggle  
    *Data source for training and evaluation*
8. Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow.* 2nd ed. O’Reilly Media.
9. Google Colab: Collaborative Python Notebook Environment  
    https://colab.research.google.com/