**House Price Prediction**

**Project Design Report**



**Supervisor**

**Dr. Ali Samad**

**Submitted by**

**1st Zohaib Nizami**

**{S21BDATS1M02021}**

**2nd Madni akhthar**

**{S21BDATS1M02046}**

**Department of Data Science**

**The Islamia University of Bahawalpur**

**Submission Certificate**

This is to certify that, we are students at the Department of Data Science, The Islamia University of Bahawalpur. We declare that our final year project design report is checked and approved by our supervisor for submission and the whole text has been written by us and not directly copied from any source.

**Students**

Madni Akhtar , signature \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Zohaib Nizami , signature \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Supervisor** Sir. Ali Samad

**Designation** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Signature** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Date**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Table of Contents**

[Chapter 01 1](#_Toc197166118)

[Introduction 1](#_Toc197166119)

[1.1 Background 1](#_Toc197166120)

[1.2 Problem Description 2](#_Toc197166121)

[1.3 Project Objectives 3](#_Toc197166122)

[1.4 Project Scope 4](#_Toc197166123)

[Chapter 02 5](#_Toc197166124)

[Literature Review 5](#_Toc197166125)

[2.1 Existing Solution 6](#_Toc197166126)

[2.2 Limitations and Gaps 6](#_Toc197166127)

[2.3 Proposed Solution 7](#_Toc197166128)

[Chapter 03 8](#_Toc197166129)

[Methodology 8](#_Toc197166130)

[3.1 Data Collection 9](#_Toc197166131)

[3.2 Data Preparation 9](#_Toc197166132)

[3.3 Data Analysis 10](#_Toc197166133)

[3.4 Modeling 11](#_Toc197166134)

[3.5 Deployment 13](#_Toc197166135)

[Chapter 04 14](#_Toc197166136)

[System Design 14](#_Toc197166137)

[4.1 Existing System 16](#_Toc197166138)

[4.2 Proposed System 16](#_Toc197166139)

[4.3 Requirement Specification 17](#_Toc197166140)

[4.4 Use Cases and UML 18](#_Toc197166141)

[4.5 Database Design 18](#_Toc197166142)

[4.6 User Interface Design 19](#_Toc197166143)

[Chapter 05 20](#_Toc197166144)

[Project Management 20](#_Toc197166145)

[5.1 System Implementation 21](#_Toc197166146)

[5.2 Testing and Evaluation 22](#_Toc197166147)

[5.3 Milestones 24](#_Toc197166148)

[5.4 Risks 24](#_Toc197166149)

[5.5 Timeline 25](#_Toc197166150)

[References 27](#_Toc197166151)

# Chapter 01

# Introduction

The real estate market is an ever-evolving industry where accurate house price prediction plays a crucial role in facilitating informed decisions for buyers, sellers, investors, and policymakers. Traditional methods for estimating property values, such as manual assessments or linear regression models, often fall short in capturing the complex interactions between various factors like location, size, amenities, and market trends. With the advancements in machine learning, it has become possible to develop predictive models that can analyze large datasets, uncover hidden patterns, and provide reliable forecasts. This project leverages a Polynomial Features model, a powerful approach capable of modeling non-linear relationships, to predict house prices based on diverse property attributes. Additionally, to enhance user experience and accessibility, the model has been deployed using a Streamlit-based frontend, allowing users to interact with the system seamlessly and obtain real-time predictions. By integrating advanced machine learning techniques with user-friendly deployment, this project aims to bridge the gap between technical solutions and practical usability in the real estate sector.

## 1.1 Background

The real estate industry is one of the most dynamic and influential sectors in the global economy, driven by fluctuating demand, evolving customer preferences, and varying market trends. Traditionally, property valuation relied heavily on manual assessments or simplistic models that focused on a limited set of variables such as the property's size and location. However, these methods often failed to capture the complex relationships between the multitude of factors that influence house prices. The advent of machine learning has transformed this scenario by enabling predictive models to analyze and synthesize diverse datasets with unprecedented precision.

In this project, the goal is to predict house prices using machine learning techniques that go beyond basic linear assumptions. The dataset contains multiple features, including geographical, structural, and market-related attributes. Leveraging these features, the project employs a Polynomial Features model that identifies non-linear patterns in the data, ensuring robust and accurate predictions. By implementing this model alongside a Streamlit-based user interface, the project aims to make house price prediction accessible and practical for real-world use.

## 1.2 Problem Description

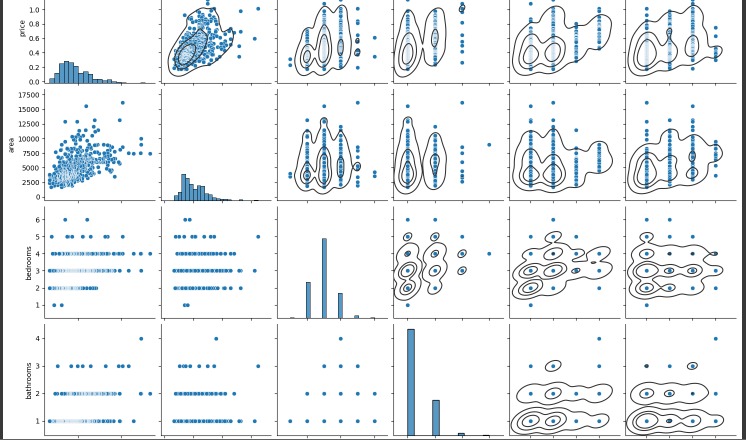
Accurate prediction of house prices is a significant challenge due to the interplay of numerous influencing factors. Features like the size of the house, number of rooms, proximity to amenities, and even economic conditions can affect property values. Many of these relationships are non-linear, which traditional statistical methods struggle to accommodate. Additionally, data quality issues, such as missing values and outliers, further complicate the modeling process, demanding robust preprocessing techniques.

This project addresses these challenges by developing a machine learning model capable of analyzing complex, non-linear interactions between variables. A Polynomial Features approach is utilized to transform the input data, enabling the model to learn higher-order relationships effectively. Furthermore, the deployment of this model using a Streamlit interface ensures that predictions are not only accurate but also easily accessible to end users. This real-time prediction system bridges the gap between machine learning advancements and practical application in the real estate domain.

## 1.3 Project Objectives

The primary objective of this project is to create a robust, accurate, and user-friendly system for predicting house prices. This involves several sub-objectives:

1. **Exploratory Data Analysis (EDA)**: Conduct a comprehensive analysis of the dataset to uncover trends, correlations, and patterns that inform feature selection and model design.



1. **Machine Learning Modeling**: Implement a Polynomial Features model to capture the non-linear relationships in the dataset and evaluate its performance using standard metrics such as R2 Score and Mean Absolute Error.
2. **Deployment**: Develop a Streamlit-based web application that allows users to input property features and obtain real-time price predictions.

By achieving these objectives, the project provides a reliable tool for real estate professionals, buyers, and sellers to make informed decisions based on data-driven insights.

## 1.4 Project Scope

This project’s scope encompasses the entire lifecycle of machine learning implementation, from data collection to deployment. Initially, the dataset is obtained from Kaggle, a trusted source for high-quality datasets. The data is then cleaned, processed, and analyzed to ensure its readiness for modeling. The modeling phase focuses on implementing the Polynomial Features approach, which is well-suited for capturing complex relationships between variables.

The scope extends to deploying the trained model using Streamlit, a Python-based framework for creating interactive web applications. The resulting system allows users to input property details such as size, location, and number of rooms and receive instant predictions. This practical deployment ensures that the project’s outcomes are not confined to theoretical evaluations but have tangible, real-world applications.

# Chapter 02

# Literature Review

House price prediction has been an extensively studied problem in the fields of economics, real estate, and machine learning. Traditional methods relied heavily on linear regression models, which assume a direct relationship between property features and prices. These models provided a foundational understanding of how key variables such as location, square footage, and number of rooms impact house prices. However, they often fell short when addressing complex, non-linear interactions between features. For instance, the impact of neighborhood quality might differ based on the property’s size or proximity to amenities, a nuance that linear models cannot effectively capture. This limitation has driven researchers toward more sophisticated techniques.

In recent years, machine learning approaches have gained prominence due to their ability to model complex relationships in data. Models such as decision trees, random forests, and support vector machines have been applied to house price prediction with varying degrees of success. These models offer improved accuracy by capturing non-linear interactions and accommodating high-dimensional data. However, they also come with challenges, including the need for extensive computational resources and a lack of interpretability. Ensemble methods like Gradient Boosting Machines and XGBoost have further pushed the boundaries, delivering state-of-the-art performance. Despite these advancements, the accessibility of such models for non-technical users remains limited, creating a gap between technical innovation and practical usability.

This project addresses the gaps identified in the literature by leveraging a Polynomial Features model to capture non-linear relationships between property features and prices. Unlike black-box models, this approach provides a balance between performance and interpretability, making it suitable for deployment in practical scenarios. Additionally, by deploying the model through a user-friendly Streamlit interface, this project ensures that the predictive power of machine learning is accessible to real estate professionals, buyers, and sellers. This integration of advanced modeling techniques with an intuitive interface bridges the gap between technical advancements and real-world applications, offering a comprehensive solution to house price prediction.

## 2.1 Existing Solution

Existing solutions for house price prediction have traditionally relied on statistical models, such as linear regression, which establish a direct relationship between independent variables (features) and the dependent variable (price). These models are easy to interpret and implement, making them widely used in real estate valuation. Some solutions have incorporated more complex approaches, such as decision trees and random forests, to capture non-linear interactions and improve accuracy. Tools like Zillow and Redfin, which use proprietary algorithms, have also emerged as benchmarks in the industry, offering price estimates based on publicly available and user-provided data.

Despite advancements, these solutions have their limitations. Statistical models are often too simplistic to account for the diverse and interdependent variables that affect house prices, such as socioeconomic factors and market volatility. On the other hand, advanced machine learning models like ensemble methods or neural networks, while accurate, are not easily interpretable or accessible to end-users without technical expertise. Furthermore, many tools lack an interactive platform that enables users to input specific features and receive personalized predictions in real-time.

## 2.2 Limitations and Gaps

The primary limitation of existing solutions lies in their inability to balance complexity and usability. Simpler models fail to capture non-linear relationships, leading to less accurate predictions. For example, the value added by a second bathroom may depend on the property’s location, a nuance missed by linear models. Conversely, while machine learning models like random forests or XGBoost can address these complexities, they often operate as "black boxes," providing limited insight into how predictions are made, which can erode user trust.

Existing solutions struggle to balance **complexity and usability**:

* Simpler models fail to capture **non-linear relationships**, leading to inaccurate predictions (e.g., the value of a second bathroom depending on location).
* Advanced models like random forests or XGBoost handle complexity but act as **"black boxes"**, limiting transparency and trust.

**Lack of accessibility and deployment options**:

* Advanced models are often restricted to **research or proprietary platforms**, limiting their practical use for real estate professionals and consumers.

**User experience limitations**:

* Many solutions lack **intuitive interfaces**, making them challenging for users with limited technical expertise.

There is a need for a model that is **both powerful and interpretable**, with an **easy-to-use deployment framework**.

## 2.3 Proposed Solution

This project proposes a solution that combines the power of a Polynomial Features model with the accessibility of a Streamlit-based deployment. The Polynomial Features model enhances prediction accuracy by transforming input features to capture complex non-linear relationships that traditional models often miss. This approach balances complexity and interpretability, offering users insights into how different features influence price predictions.

To address the usability gap, the model is integrated into a Streamlit interface, enabling users to interact with the system seamlessly. Users can input property details such as size, location, and amenities and receive real-time predictions without requiring any technical knowledge. This proposed solution bridges the gap between technical sophistication and practical application, ensuring that the power of machine learning is both accessible and actionable for real-world use cases. By focusing on accuracy, interpretability, and usability, the project sets a new standard for house price prediction tools.

# Chapter 03

# Methodology

The methodology of this project follows a structured approach, starting with data collection and preprocessing, followed by model development, evaluation, and deployment. The dataset for this project was sourced from Kaggle, containing various features related to house prices, such as square footage, location, number of rooms, and proximity to amenities. The initial step involved a thorough examination of the dataset to understand its structure and identify potential issues, such as missing values, outliers, or inconsistencies. Exploratory Data Analysis (EDA) was conducted to uncover relationships between features, trends, and patterns that could inform the model design. Visualizations using tools like Seaborn and Matplotlib were crucial in understanding these relationships and identifying the need for a non-linear model.

Data preparation was a key phase in the methodology, ensuring the dataset was suitable for training a robust machine learning model. Missing values were handled using imputation techniques, while outliers were addressed to minimize their impact on the model. Features were scaled and normalized to ensure compatibility with the Polynomial Features transformation, which generates higher-order feature interactions. Additionally, feature selection techniques, such as Recursive Feature Elimination (RFE), were employed to identify the most influential predictors, reducing noise and improving model efficiency. These preprocessing steps ensured the data was clean, consistent, and optimized for training.

Modeling was carried out using the Polynomial Features approach, chosen for its ability to capture complex, non-linear interactions between variables. The model transformed input features into higher-degree combinations, allowing it to account for relationships that linear models could not. The training process involved splitting the data into training and testing sets, ensuring the model was evaluated on unseen data to assess its generalizability. Performance metrics, including R2 Score and Mean Absolute Error (MAE), were used to evaluate the model's accuracy and reliability. Once trained and validated, the model was deployed using Streamlit, a Python-based framework for building interactive web applications. The deployment phase included designing an intuitive interface that allows users to input property details and receive real-time predictions, making the solution both powerful and accessible.

## 3.1 Data Collection

The data used in this project was sourced from Kaggle, a platform known for providing high-quality datasets. The dataset contains a variety of features that are crucial for house price prediction, such as property size, number of rooms, location, proximity to schools, and amenities. Additionally, it includes financial and market-related features like historical price trends, which offer insights into market dynamics. The dataset’s comprehensive nature ensures that it can capture the multifaceted factors influencing house prices, providing a robust foundation for model development.

To ensure the dataset was suitable for analysis, a thorough inspection was performed to identify any missing values, inconsistencies, or errors. Metadata documentation was reviewed to understand the definitions of each feature, and exploratory checks were conducted to ensure data completeness. Data collection was guided by the principle of inclusivity, ensuring that all relevant variables contributing to house prices were captured. The dataset was then imported into the Python environment using libraries such as Pandas, allowing for seamless manipulation and analysis.

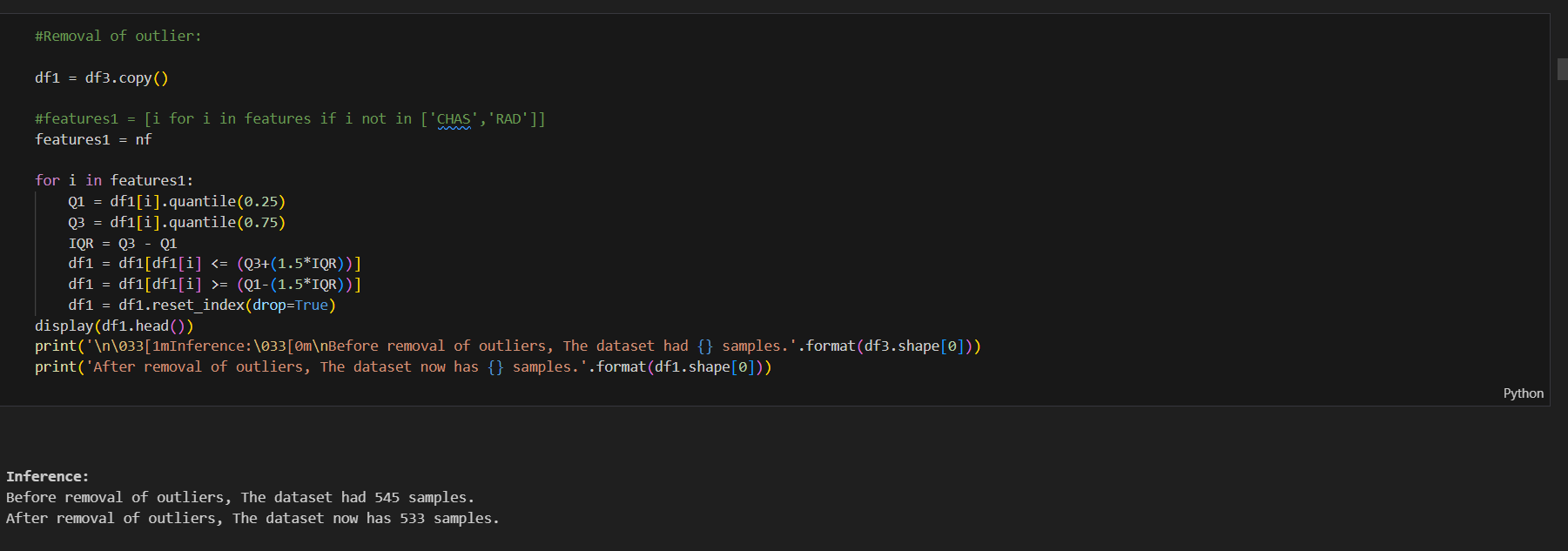
Furthermore, to address potential biases, the data was checked for representativeness across different categories, such as property types and geographical locations. Any skewness in the dataset was flagged for correction during the preprocessing stage. By starting with a well-curated and comprehensive dataset, the project laid a solid groundwork for accurate and reliable predictions.

## 3.2 Data Preparation

Data preparation was a critical step in ensuring the dataset was ready for machine learning. Missing values were a significant concern, as they could distort model performance. These were handled using imputation techniques, such as mean imputation for numerical features and mode imputation for categorical features. Outliers, which could potentially skew predictions, were detected using statistical methods like the interquartile range (IQR) and were addressed by capping or removing extreme values.

Feature selection was another crucial part of data preparation. Recursive Feature Elimination (RFE) was employed to identify the most important features that contributed to house price predictions. This step helped reduce the dimensionality of the dataset, ensuring that only the most relevant variables were included in the model. Additionally, all numerical features were scaled using StandardScaler to ensure uniformity and compatibility with the Polynomial Features transformation, which generates higher-order terms from the input features.

Categorical variables were encoded using one-hot encoding to convert them into numerical format while preserving the underlying information. This transformation allowed the model to understand and utilize categorical data effectively. By the end of the preparation phase, the dataset was clean, well-structured, and optimized for model training, ensuring a smooth transition to the modeling stage.

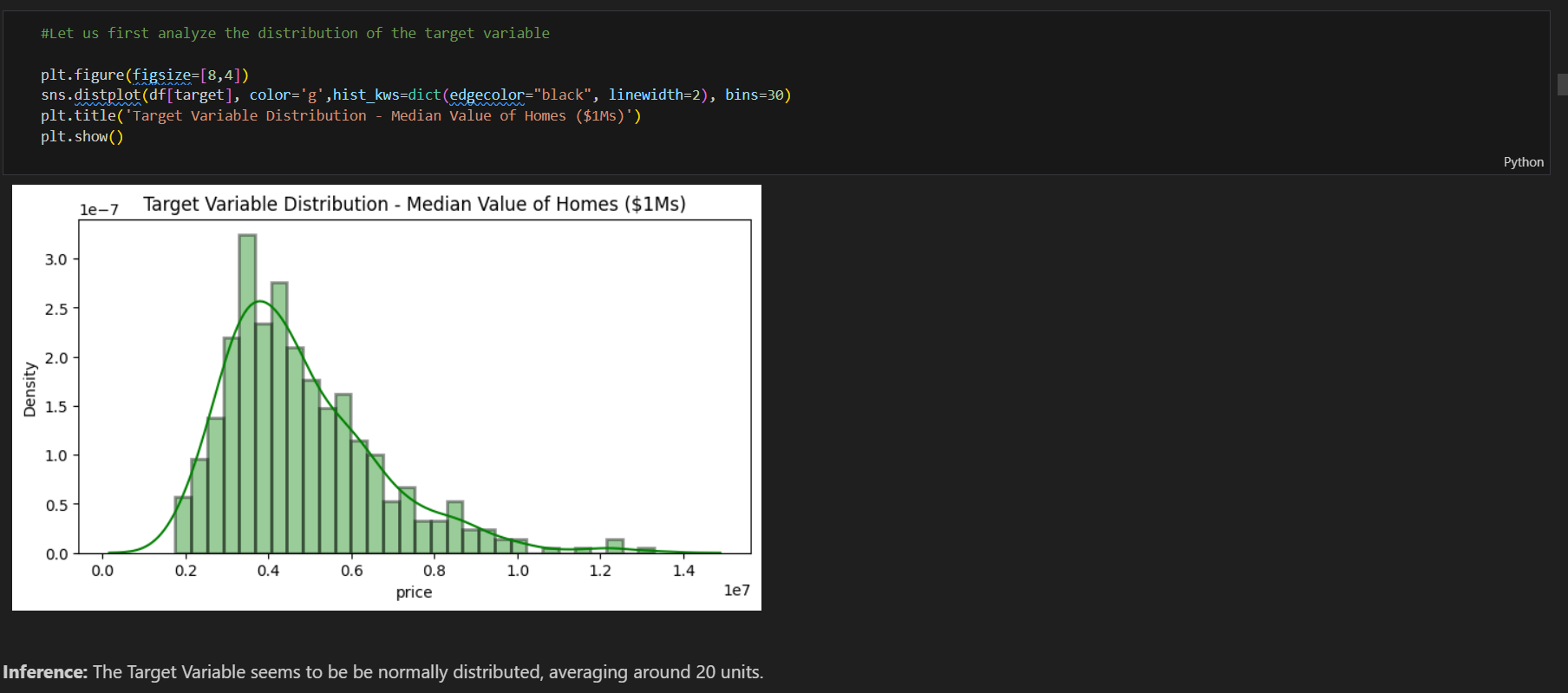


## 3.3 Data Analysis

Exploratory Data Analysis (EDA) was performed to uncover insights and patterns within the dataset. This involved visualizing relationships between features and identifying trends that could influence house prices. For example, scatter plots were used to analyze the correlation between house size and price, while heatmaps revealed interactions among multiple variables. EDA provided a deeper understanding of the dataset, highlighting the need for a non-linear modeling approach to capture complex relationships.

Descriptive statistics such as mean, median, and standard deviation were calculated to summarize the dataset. Distribution plots were generated to identify skewness and outliers in key features, such as price and square footage. These visualizations helped inform decisions about feature transformations and adjustments. For example, log transformations were considered for variables with significant skewness to normalize their distribution.

EDA also highlighted the presence of multicollinearity, where certain features were highly correlated with each other. This insight guided the feature engineering process, ensuring redundant variables were removed to improve model efficiency. The findings from EDA not only validated the choice of using a Polynomial Features model but also shaped the subsequent steps in the modeling process.

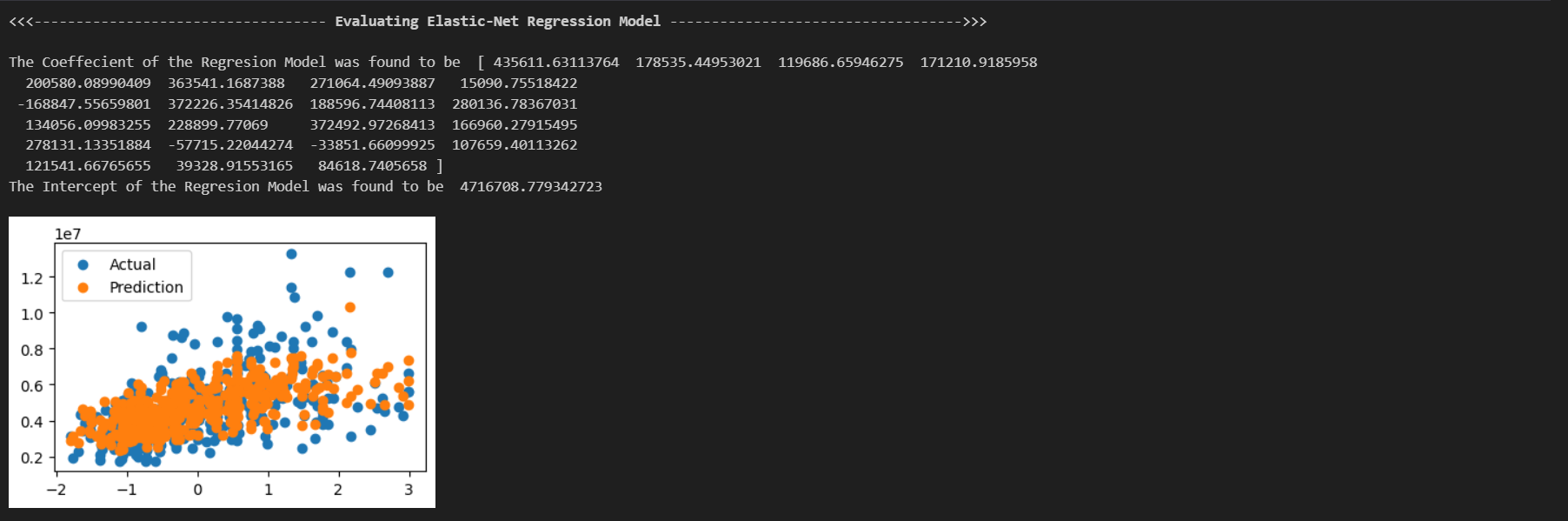


## 3.4 Modeling

The core of this project was the implementation of a Polynomial Features model, which transforms input data to include higher-degree combinations of features. This approach was chosen to capture the non-linear interactions that are common in real estate markets, such as the compounded effect of location and size on house prices. The model training process began by splitting the dataset into training and testing sets, ensuring that the model’s performance was evaluated on unseen data.

Once the data was prepared, the Polynomial Features transformation was applied to generate additional terms from the original features. These transformed features provided the model with the ability to learn complex relationships, such as quadratic or cubic dependencies. The transformed data was then fed into a linear regression model, which leveraged these higher-order terms to make predictions. Hyperparameter tuning was conducted to optimize the model, ensuring a balance between accuracy and computational efficiency.

The model’s performance was evaluated using metrics such as R2 Score, Mean Absolute Error (MAE), and Mean Squared Error (MSE). These metrics provided a comprehensive understanding of the model’s predictive accuracy and reliability. The Polynomial Features model demonstrated its ability to outperform traditional linear models, validating its suitability for the task. This phase concluded with a robust and well-validated model ready for deployment.



## 3.5 Deployment

To make the model accessible and user-friendly, it was deployed using Streamlit, a Python-based framework for building interactive web applications. The deployment process involved designing a graphical user interface (GUI) that allows users to input property features and obtain price predictions in real-time. This interface includes fields for entering details such as house size, number of rooms, and location, as well as a button to generate predictions.

Streamlit was chosen for its simplicity and effectiveness in creating intuitive interfaces. The deployment process involved integrating the trained model with the Streamlit application, ensuring seamless interaction between the backend model and the frontend interface. Real-time processing capabilities were implemented to ensure users receive instant feedback, enhancing the usability of the application.

The deployed application provides a practical tool for real estate professionals, buyers, and sellers, bridging the gap between machine learning technology and everyday use. By making the predictions accessible through a simple and interactive platform, the project achieves its goal of providing a powerful yet user-friendly solution for house price prediction.

# Chapter 04

# System Design

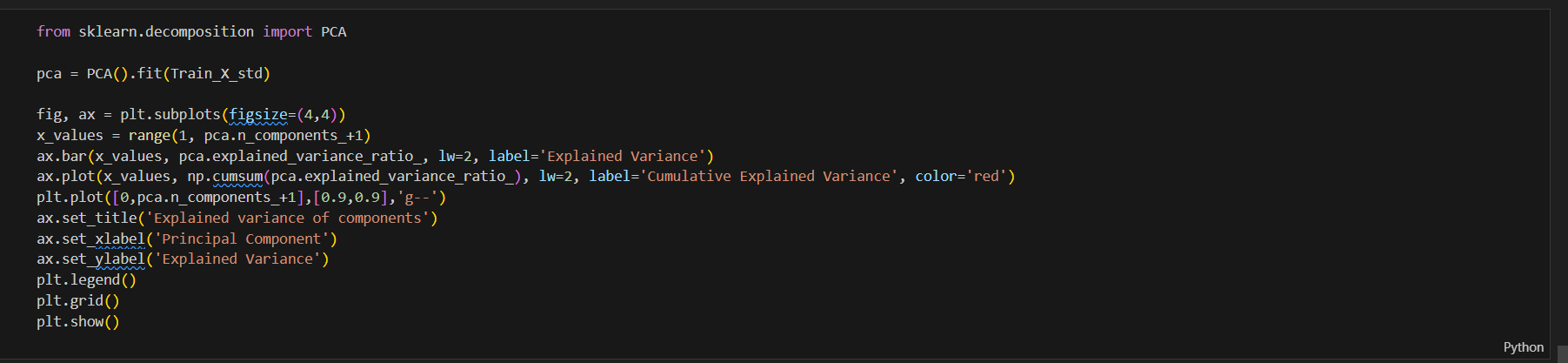
The system design for this house price prediction project is structured to seamlessly integrate advanced machine learning techniques with a user-friendly deployment interface. It involves multiple components, including the data processing pipeline, the modeling framework, and the deployment mechanism. Each component is meticulously designed to ensure robustness, scalability, and ease of use, allowing end-users to interact with the system efficiently while benefiting from accurate predictions.

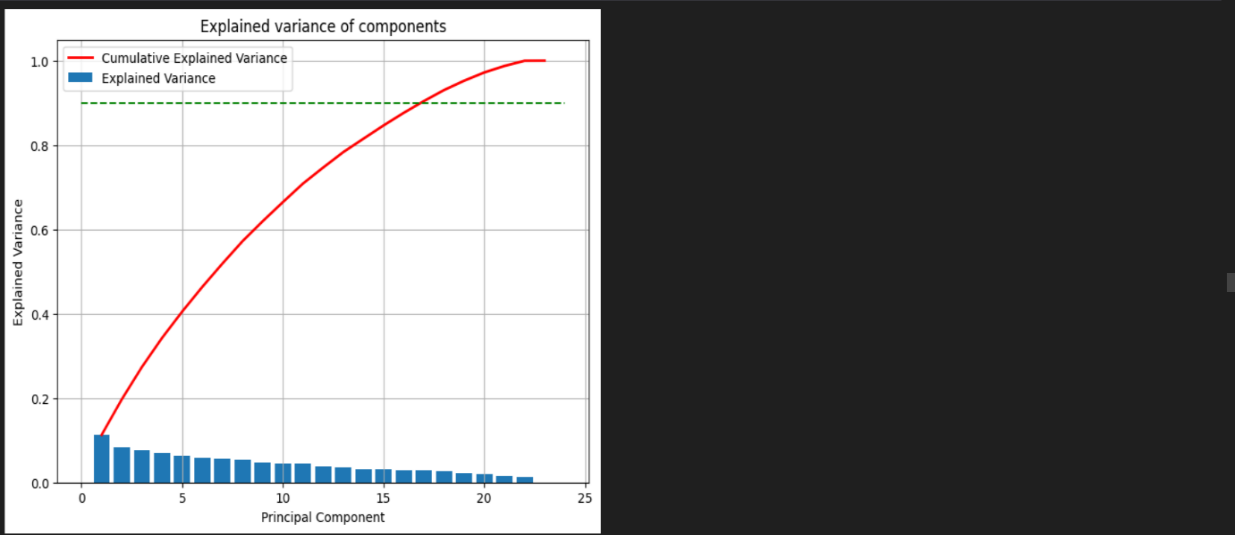
The core of the system design revolves around the implementation of a **Polynomial Features model**, which transforms the input features into higher-order combinations to capture complex, non-linear relationships. This process begins with a well-structured data pipeline that preprocesses raw data by cleaning, normalizing, and encoding it into a format suitable for machine learning. The data pipeline ensures that missing values are handled appropriately, outliers are addressed, and all features are scaled or encoded for compatibility with the model. Feature selection techniques, such as Recursive Feature Elimination (RFE), further refine the input data by identifying the most relevant predictors, reducing noise and improving model performance. The preprocessing steps are modular, enabling future updates or adjustments to the dataset without affecting the downstream components.

The system is designed to optimize the interaction between data and the machine learning model. The Polynomial Features transformation is applied as part of the modeling framework to generate higher-order terms, enabling the linear regression model to learn complex patterns. The transformed data is fed into the regression model, which has been tuned for optimal performance using hyperparameter adjustment techniques. To ensure that the system can generalize well to new data, the model is evaluated using metrics such as R2 Score, Mean Absolute Error (MAE), and Mean Squared Error (MSE) on a test dataset. This evaluation step is integrated into the design to provide continuous feedback on the model’s performance, enabling iterative improvements and ensuring reliability.

Deployment is a critical aspect of the system design, ensuring that the machine learning model is accessible to end-users. The system leverages **Streamlit**, a Python-based framework, to create a web-based interface that allows users to input property details and receive price predictions in real-time. The interface is designed with simplicity and intuitiveness in mind, featuring input fields for key property attributes, a submit button to process predictions, and visual feedback displaying the results. The backend integrates the trained machine learning model with the Streamlit application, handling data processing and prediction generation seamlessly. The deployment architecture is lightweight and scalable, allowing it to be hosted on cloud platforms or local servers based on user requirements.

Furthermore, the system is designed to accommodate future enhancements, such as integrating additional models or expanding the input features to include external data sources like market trends or economic indicators. This flexibility is achieved through a modular architecture, where each component operates independently but communicates effectively with others. For example, the preprocessing pipeline, model training framework, and deployment interface are interconnected through well-defined APIs, ensuring smooth data flow and compatibility. The overall system design emphasizes not only high performance and accuracy but also accessibility and adaptability, making it a comprehensive solution for house price prediction in real-world scenarios.





## 4.1 Existing System

Existing house price prediction systems largely rely on traditional statistical models or basic machine learning algorithms that fail to fully capture the intricacies of the real estate market. Linear regression models, for instance, assume a linear relationship between features and house prices, which may oversimplify the problem. These systems often depend on a limited set of features, such as square footage, location, and number of rooms, ignoring other influential factors like neighborhood quality, proximity to amenities, and market trends. While some advanced machine learning systems, such as those employed by real estate platforms like Zillow and Redfin, utilize proprietary algorithms to improve prediction accuracy, they are typically inaccessible to the general public or require subscription-based services.

A significant limitation of the existing systems is their inability to capture non-linear relationships effectively. For example, the impact of location on house prices might vary depending on other features, such as property size or proximity to transport hubs. Additionally, these systems often lack transparency, as they do not provide users with insights into how predictions are made. Usability is another challenge; most systems are not interactive or user-friendly, requiring technical expertise to operate. This creates a gap between the advanced capabilities of machine learning algorithms and the practical needs of end-users like real estate agents, buyers, and sellers.

## 4.2 Proposed System

The proposed system aims to address the limitations of existing solutions by integrating a powerful Polynomial Features model with an interactive and user-friendly deployment interface. Unlike traditional models, the Polynomial Features model transforms input data into higher-order terms, allowing the system to capture complex, non-linear interactions between features. For instance, the combined influence of property size and neighborhood quality on price can be effectively modeled through these transformations, resulting in significantly improved prediction accuracy.

To make this advanced system accessible, it is deployed using Streamlit, a Python-based framework for building web applications. The interface allows users to input key property attributes, such as size, location, and amenities, and receive real-time predictions. Additionally, the system provides visual feedback, such as charts or graphs, to explain how each feature contributes to the predicted price, enhancing transparency and user trust. The modular architecture of the proposed system ensures scalability, enabling the integration of additional features or models as required. For example, external data sources, such as economic indicators or regional market trends, can be incorporated in future iterations to further enhance prediction accuracy.

## 4.3 Requirement Specification

The system requires a combination of hardware and software components to function efficiently. On the hardware side, the system should run on a machine with at least 8GB of RAM and a modern multi-core processor to handle the computational demands of data preprocessing, model training, and prediction generation. Storage requirements depend on the dataset size but should accommodate large datasets for scalability. An internet connection is recommended for hosting the Streamlit application and enabling seamless access.

From a software perspective, the system is built using Python 3.8 or above, leveraging libraries such as Pandas for data manipulation, Scikit-learn for machine learning, and Streamlit for deployment. Visualization libraries like Matplotlib and Seaborn are used for data analysis and to provide insights within the user interface. Additionally, the system requires a virtual environment to manage dependencies and ensure compatibility across different setups. The modular architecture ensures that each component, such as data preprocessing, modeling, and deployment, operates independently while seamlessly integrating into the larger framework.

## 4.4 Use Cases and UML

The system's primary use case is to provide real-time house price predictions based on user input. The workflow begins with the user entering property details, such as square footage, location, and number of bedrooms, into the Streamlit interface. The system preprocesses this input, applies the trained Polynomial Features model, and returns the predicted price alongside visual feedback. This functionality is particularly useful for real estate agents and buyers, enabling them to assess property values quickly and make informed decisions.

Secondary use cases include the ability to expand the system for batch predictions, where multiple property details can be uploaded via a CSV file for bulk processing. This functionality is valuable for real estate firms or researchers analyzing market trends. A UML diagram illustrating the workflow would include the user, Streamlit interface, preprocessing module, prediction model, and output display. The interaction between these components ensures a seamless and efficient user experience. (Add UML diagram here)

## 4.5 Database Design

The database design for the system is intentionally kept simple to ensure scalability and efficiency. The primary data source is the Kaggle dataset, stored in CSV format for ease of manipulation and access. The dataset includes features such as property size, location, number of rooms, and historical price trends. During the preprocessing phase, additional columns may be generated, such as transformed features from the Polynomial Features model, which are temporarily stored in memory to reduce storage overhead.

The database schema is designed to support extensibility, allowing for the integration of new datasets or features as needed. For example, columns for external data, such as economic indicators or real estate market trends, can be added without restructuring the database. The system operates on a load-on-demand principle, ensuring that only the necessary data is processed and reducing computational costs. This lightweight and flexible database design aligns with the overall goal of creating an efficient and scalable system.

## 4.6 User Interface Design

The user interface is a critical component of the system, as it bridges the gap between the complex backend processes and the end-users. Designed using Streamlit, the interface is intuitive and interactive, enabling users to input property details and receive predictions seamlessly. The layout is simple yet functional, with clearly labeled fields for inputs such as house size, number of bedrooms, location, and proximity to amenities. A prominently displayed “Predict” button triggers the backend model to generate results, which are displayed in both numerical and graphical formats.

To enhance the user experience, the interface includes features like real-time validation of inputs, ensuring users enter valid and meaningful data. For instance, the system may prompt users to re-enter a value if it detects a non-numerical input in a numerical field. Visual feedback, such as bar charts or pie charts, is integrated to explain how each feature contributes to the final prediction, improving transparency and user trust. The interface is designed to be responsive and lightweight, ensuring smooth performance on various devices, from desktops to tablets. (Add screenshots of the Streamlit interface here)

# Chapter 05

# Project Management

Managing the development of the house price prediction system required a structured and phased approach to ensure that all aspects of the project were completed efficiently and effectively. The project was divided into distinct phases, including planning, data preparation, model development, deployment, and evaluation. Each phase was carefully planned, with specific goals, milestones, and deliverables. This structured approach ensured that the project stayed on track and that resources were allocated effectively, minimizing the risk of delays or bottlenecks.

The planning phase focused on understanding the project requirements and defining the objectives. This involved identifying the dataset from Kaggle, analyzing its suitability for the task, and determining the machine learning techniques to be used. A timeline was established, dividing the project into weekly tasks to maintain consistent progress. Tools such as Gantt charts were used to visualize the timeline and ensure that dependencies between tasks were identified and managed. The planning phase also included risk assessment, where potential challenges, such as data quality issues or deployment difficulties, were identified, and mitigation strategies were devised.

During the execution phase, the tasks outlined in the planning phase were implemented. Data preprocessing and feature engineering were the initial focus, ensuring that the dataset was clean, consistent, and optimized for modeling. This required close monitoring to ensure that tasks such as imputation, outlier handling, and feature scaling were completed within the allocated timeframes. The modeling phase involved experimenting with the Polynomial Features transformation and tuning the regression model for optimal performance. This phase required frequent evaluations using metrics such as R2 Score and Mean Absolute Error to ensure the model met the project's accuracy requirements.

The deployment and evaluation phase emphasized the practical application of the system. The trained model was integrated into a Streamlit interface, creating an intuitive and user-friendly platform for real-time predictions. Rigorous testing was conducted to identify and fix any issues related to the user interface, backend integration, or prediction accuracy. Feedback loops were established during this phase, allowing for refinements based on user testing and evaluation results. Continuous monitoring of the deployed system ensured it met performance standards and user expectations. Overall, project management played a critical role in delivering a robust, accurate, and accessible house price prediction system within the defined timeline and resource constraints.

## 5.1 System Implementation

The implementation of the house price prediction system followed a carefully designed roadmap, integrating various components to create a seamless and functional workflow. The first step in implementation was data preprocessing, which involved cleaning and preparing the dataset for analysis. Missing values were handled using statistical imputation techniques, while outliers were treated using methods like capping or removal to minimize their impact on the model. Feature selection and scaling were performed to ensure that the data was optimized for the Polynomial Features transformation, which played a central role in capturing non-linear relationships.

Once the data was prepared, the next phase focused on model development. The Polynomial Features transformation was applied to generate higher-order terms, creating new combinations of input variables that could better capture the complexities of house price determinants. These transformed features were fed into a linear regression model, and hyperparameter tuning was conducted to enhance its performance. Various evaluation metrics, such as R2 Score and Mean Absolute Error (MAE), were calculated to assess the model’s accuracy and reliability, ensuring it met the predefined performance benchmarks.

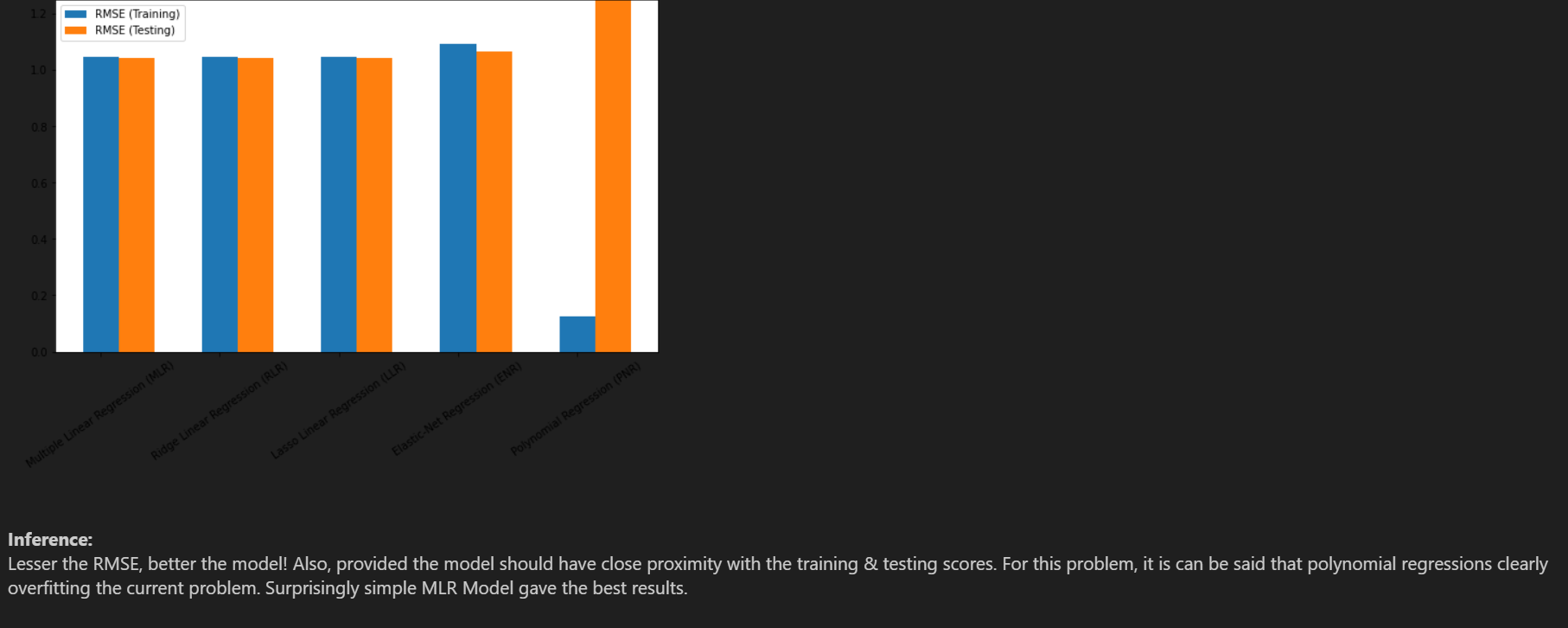
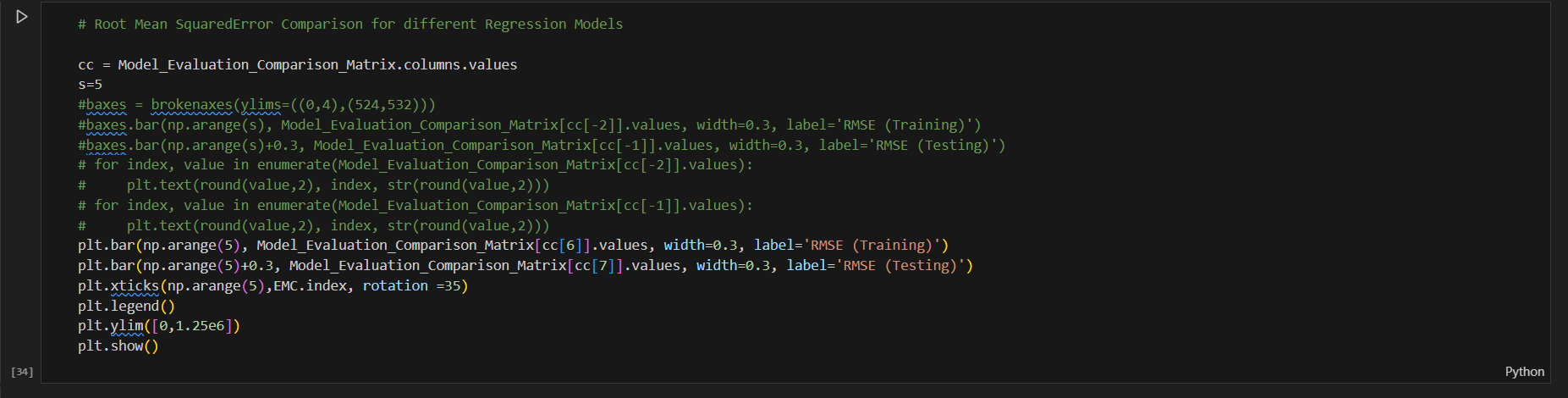
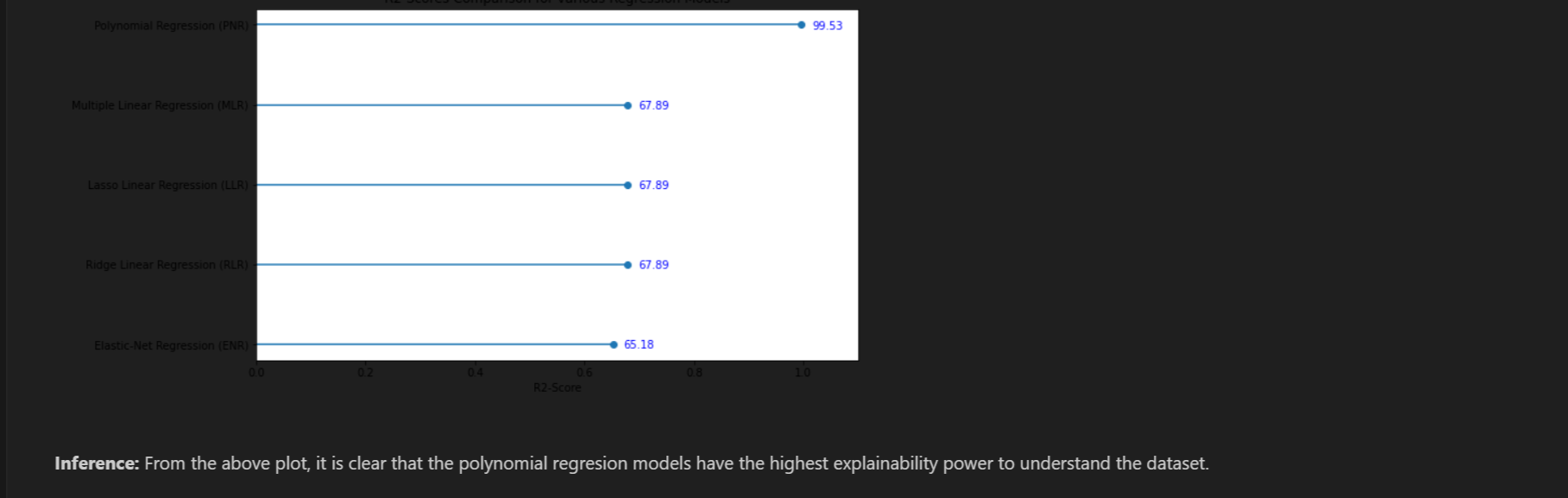
The final stage of implementation involved deploying the trained model using Streamlit, a Python-based framework for web applications. The Streamlit application was developed to allow users to input house attributes, such as size, location, and number of rooms, and receive predictions in real-time. The deployment process required integrating the backend model with a user-friendly frontend, ensuring smooth interaction and efficient processing. The system was also tested for responsiveness and scalability, making it capable of handling a wide range of inputs without performance degradation.

## 5.2 Testing and Evaluation

The testing and evaluation phase was crucial in ensuring the system's reliability, accuracy, and usability. The trained model was tested on a separate testing dataset to evaluate its generalization capabilities and identify potential weaknesses. The testing process involved calculating key metrics such as R2 Score, which measures how well the model's predictions align with actual values, and Mean Absolute Error (MAE), which quantifies the average difference between predicted and actual prices. These metrics provided insights into the model's strengths and areas for improvement.

In addition to model evaluation, the Streamlit application underwent extensive user testing to ensure that the interface was intuitive and functional. Beta testing sessions were conducted with a sample group of users, including real estate professionals and non-technical individuals, to gather feedback on the user experience. Issues such as slow response times, input validation errors, and unclear outputs were identified and resolved during this phase. The feedback loop established during testing allowed for iterative improvements, ensuring that the system met the needs of its intended users.

Furthermore, edge cases and extreme inputs were tested to ensure the system's robustness. For example, the model was tested with unusually high or low values for property size and location quality to verify its handling of outliers and anomalies. Stress testing was also conducted to assess the system's performance under high workloads, ensuring scalability and reliability in real-world scenarios. By the end of the evaluation phase, the system was refined to deliver accurate predictions with a user-friendly experience.



## 5.3 Milestones

The project was divided into several milestones to ensure steady progress and timely completion. The first milestone focused on data collection and preprocessing, which included obtaining the dataset from Kaggle, cleaning it, and performing exploratory data analysis (EDA). This phase was completed in the first two weeks and set the foundation for subsequent tasks by ensuring that the data was ready for modeling.

The second milestone involved model development, where the Polynomial Features transformation was applied, and the regression model was trained and tuned. This phase lasted two weeks and included extensive experimentation with hyperparameters to optimize the model's performance. Evaluation metrics were calculated to validate the model's effectiveness, and any underperforming configurations were iteratively improved. The completion of this milestone marked a significant achievement in developing a robust predictive model.

The third milestone focused on deployment and user interface development. The Streamlit application was built and integrated with the trained model, creating a platform for real-time predictions. This phase required collaboration between backend and frontend components, ensuring seamless interaction. The final milestone, testing and evaluation, involved validating the system's accuracy, usability, and scalability. This phase lasted one week and concluded with a stable and functional system ready for real-world use. These milestones ensured the project stayed on track and delivered results within the specified timeline.

## 5.4 Risks

Several risks were identified during the project, ranging from data-related issues to deployment challenges. One of the primary risks was overfitting due to the complexity of the Polynomial Features transformation, which creates numerous higher-order terms. Overfitting could lead to a model that performs well on the training data but fails to generalize to new data. To mitigate this risk, regularization techniques were applied, and cross-validation was used during model training to evaluate its performance on unseen data.

Another significant risk was data quality. Missing values, outliers, and inconsistencies in the dataset could compromise the model's accuracy. This risk was addressed by implementing robust preprocessing steps, such as imputation and outlier handling, to ensure data integrity. Additionally, multicollinearity among features was identified as a potential issue, which was mitigated by performing feature selection and analyzing the Variance Inflation Factor (VIF) to remove redundant predictors.

Deployment posed its own set of challenges, particularly in ensuring the Streamlit application was scalable and responsive under varying workloads. Risks such as slow response times and compatibility issues with different devices were addressed through extensive testing and optimization. Finally, user adoption risk was considered, as non-technical users might find the system difficult to use. To mitigate this, the user interface was designed to be intuitive and user-friendly, with clear instructions and real-time feedback for input validation.

## 5.5 Timeline

The timeline for the project was carefully structured, with each phase allocated a specific duration to ensure timely completion. The project began with a one-week planning phase, during which objectives, tools, and datasets were finalized. This was followed by a two-week data collection and preprocessing phase, which involved cleaning the dataset, performing exploratory data analysis, and preparing it for modeling. These tasks ensured that the dataset was ready for the model training phase.

The third phase, spanning two weeks, focused on model development and evaluation. During this time, the Polynomial Features transformation was applied, and the regression model was trained and optimized. Evaluation metrics such as R2 Score and Mean Absolute Error were calculated to assess the model’s performance, and adjustments were made to improve accuracy. This phase also included initial testing to validate the model on unseen data.

The final two weeks of the project were dedicated to deployment and user testing. The Streamlit application was developed and integrated with the trained model, creating a platform for real-time predictions. Extensive testing was conducted during this phase to ensure that the system was user-friendly and responsive. By the end of this timeline, the project was fully implemented, tested, and ready for deployment, adhering to the predefined schedule and milestones.

# References

Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5-32. <https://doi.org/10.1023/A:1010933404324>

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. The Annals of Statistics, 29(5), 1189-1232. <https://doi.org/10.1214/aos/1013203451>

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825-2830. <https://jmlr.org/papers/v12/pedregosa11a.html>

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, 65(6), 386. <https://doi.org/10.1037/h0042519>

Zhou, Z. H. (2021). Machine Learning. Springer Nature. <https://link.springer.com/book/10.1007/978-981-15-5510-3>

Kaggle. (n.d.). Ames Housing Dataset. Retrieved from Kaggle. <https://www.kaggle.com/datasets/prevek18/ames-housing-dataset>

Montgomery, D. C., Peck, E. A., & Vining, G. G. (2012). Introduction to Linear Regression Analysis. John Wiley & Sons. <https://doi.org/10.1002/9781118386088>

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785-794). <https://doi.org/10.1145/2939672.2939785>

Zhang, D., & Zhou, Z. H. (2007). Multilabel neural networks with applications to functional genomics and text categorization. IEEE Transactions on Knowledge and Data Engineering, 18(10), 1338-1351. <https://doi.org/10.1109/TKDE.2007.190746>

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press. [http://www.deeplearningbook.org](http://www.deeplearningbook.org/)

Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer. <https://doi.org/10.1007/978-0-387-84858-7>

Streamlit. (n.d.). The fastest way to build and share data apps. Retrieved from Streamlit. <https://streamlit.io/>

Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer. <https://doi.org/10.1007/978-0-387-45528-0>

Geurts, P., Ernst, D., & Wehenkel, L. (2006). Extremely randomized trees. Machine Learning, 63(1), 3-42. <https://doi.org/10.1007/s10994-006-6226-1>

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980. <https://arxiv.org/abs/1412.6980>