University of Petra

Faculty of Administrative and Financial Sciences

Department of Business Intelligence and Data Analytics

Real Estate Price Analysis: Exploratory Data Analysis & Prediction

MHD Besher Al Ashkar

Semester: First Semester

2024/2025

Date:

STUDENTS' PROPERTY RIGHT DECLARATION AND ANTI-PLAGIARISM STATEMENT

We hereby certify that this graduation project at University of Petra is our original work, except for quotations and summaries that have been appropriately cited. This project has not been accepted for any degree nor is it currently being submitted for any other degree. It remains the exclusive property of University of Petra and is safeguarded under intellectual property laws and conventions.

We affirm that this report is our own work, excluding properly referenced quotations, and contains no plagiarism. We have read and understood the university's policies on assessment offenses, as outlined in the University of Petra Handbook.

Student:

Name: **Besher Alashkar** Signature: **Besher Alashkar** Date:

Contents

ABSTRACT

In the real estate market, accurately predicting property prices is crucial for investors, buyers, sellers, and policymakers to make informed decisions. This project aims to analyze and forecast real estate prices using a robust data-driven approach, addressing the growing need for accurate price predictions in the housing sector.
The primary goal is to develop a predictive model that estimates housing prices with high precision, offering valuable insights into the dynamics of the real estate market. This project utilizes a publicly available dataset from Kaggle, which contains detailed information on California housing prices, including factors like median income, population, and house characteristics.

The process begins with an extensive Exploratory Data Analysis (EDA) to understand the dataset's structure, identify influential features, and handle any data quality issues such as outliers or missing values. Statistical methods, including Pearson correlation for numerical variables and variance analysis, are employed to identify the most impactful predictors. Additionally, data visualization techniques are used to uncover hidden patterns and trends within the dataset.

Following the EDA, several machine learning models are implemented, including Linear Regression, Random Forest, Gradient Boosted Trees, and Support Vector Regression (SVR). These models are evaluated based on metrics like Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared to ensure the accuracy and reliability of the predictions.

In evaluating the performance of the models, Gradient Boosted Trees emerged as the best-performing model, achieving the lowest RMSE and the highest R-squared, indicating superior predictive accuracy. Random Forest and Neural Networks also demonstrated strong performance, particularly in capturing non-linear relationships within the data. Linear Regression, while simple and interpretable, showed limitations in handling complex interactions among variables, ranking lower in predictive accuracy. Support Vector Regression, though effective in some scenarios, struggled with the scale of the dataset and ranked lowest overall.

This project encompasses several stages, from data preprocessing and feature selection to model development and evaluation. The findings aim to assist stakeholders in making better decisions related to property investment and policy-making, ultimately contributing to a more transparent and efficient real estate market.

Keywords- Real Estate Analysis, Housing Prices, Machine Learning, EDA.

CHAPTER 1: INTRODUCTION

Introduction to the Topic

In today's competitive business environment, effectively utilizing data is essential for gaining an edge and driving growth. Business Intelligence (BI) and Data Analytics play a vital role in transforming raw data into actionable insights. BI encompasses the tools and techniques used to collect, process, and analyze business data, enabling companies to make well-informed decisions based on historical and real-time information. On the other hand, Data Analytics involves examining data to identify patterns and trends that can influence future strategies and decision-making.

This project is important because it addresses the challenge of accurately analyzing and predicting real estate prices, a critical issue in the real estate market. Inaccurate price assessments can lead to poor investment decisions and financial losses. Traditional methods for evaluating property prices are often limited and lack precision, leading to inefficiencies. By developing a more accurate model for price prediction, this project aims to equip stakeholders with better tools for analyzing market trends and forecasting property values, ultimately guiding more informed decision-making in the real estate sector.

The focus of this project on conducting a comprehensive real estate price analysis emphasizes the increasing demand for advanced tools that deliver clear insights into property market trends. By utilizing machine learning algorithms, the accuracy of price predictions will be significantly improved, offering a more reliable solution for evaluating property values. Overall, this project tackles a critical issue in the real estate industry and promotes data-driven approaches to support better decision-making for investors, buyers, and real estate professionals.

Problem Statement

Despite advancements in Business Intelligence (BI) and data analytics, many organizations still struggle to accurately predict real estate prices. This project aims to tackle these challenges by improving the accuracy and effectiveness of price prediction methods. Enhancing these techniques is crucial for developing data-driven strategies that can guide investment decisions, optimize property valuations, and better understand market dynamics in the real estate sector.

Addressing this issue is crucial because it directly influences investment decisions, market stability, and overall business performance in the real estate sector. Accurate price prediction allows stakeholders to make informed choices, optimize property investments, and better navigate market fluctuations. By tackling this problem, the project will contribute to more effective business practices, benefiting the real estate industry and its stakeholders as a whole.

Objectives and Goals

The primary objectives of this graduation project are to conduct a comprehensive analysis, including Exploratory Data Analysis (EDA), and to develop price prediction models using advanced machine learning algorithms. This project aims to refine real estate price prediction techniques and support informed decision-making in the real estate market. By focusing on both in-depth data analysis and predictive modeling, the project seeks to provide valuable insights into property price dynamics, assisting buyers, investors, and other stakeholders in making data-driven decisions in the real estate sector.

Project Scope and Limitations

This project is focused on analyzing the telecom customer churn dataset through Exploratory Data Analysis (EDA) and developing predictive models. The main tasks include visualizing customer data, identifying key features, and building churn prediction models using advanced machine learning techniques. The scope does not include the development of a practical, deployable tool but rather centers on analysis and model development.

The project is primarily reliant on the data provided, and any insights or predictions are contingent upon the accuracy and completeness of this dataset. The project is focused on analysis and model development rather than the creation of a deployable tool, so it does not address real-time implementation or practical application aspects. Computational resources also limit the ability to test more complex models or run extensive simulations.

Methodology Overview

To achieve the objectives of this graduation project, we employed a range of tools and techniques. Data cleaning, engineering, and statistical analyses, including hypothesis testing and correlation analysis, were conducted using Python. For data visualization, we utilized Python libraries such as Matplotlib, Seaborn, in addition to Power BI for interactive dashboards and in-depth visual insights. Machine learning models for price prediction were developed and evaluated in Python, focusing on metrics like Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). This comprehensive approach allowed for detailed analysis and clear presentation of findings, contributing to an accurate and insightful exploration of real estate price dynamics.

Stakeholders and Impact

This project primarily targets stakeholders in the real estate industry, including property investors, real estate agents, analysts, and market researchers, who rely on data-driven insights for strategic decision-making. It is also relevant for academic researchers in data science, machine learning, and economic analysis focused on property valuation. The predictive model developed in this project provides a valuable tool for estimating property prices with accuracy, enabling stakeholders to make informed choices about buying, selling, and investing in real estate. By offering these insights, the project not only supports market transparency and confidence but also contributes to the financial decision-making processes in the real estate sector. Furthermore, it enriches the academic community's understanding of predictive analytics applications in real estate, providing a foundation for future research and advancements in the field.

Project Timeline and Milestones

The project timeline covers each phase from initial data exploration through to the final model evaluation, with carefully planned milestones. The project begins with data collection and cleaning, followed by exploratory data analysis and feature selection. The next key milestone involves applying machine learning models for price prediction. The project will conclude with a detailed evaluation of model performance, with specific deadlines set for each phase to ensure steady progress and successful completion.

Provide a high-level figure of your solution

The solution for predicting real estate prices is organized into a structured pipeline, as shown in the high-level figure. The pipeline consists of several key stages: data collection, preprocessing, transformation, model building, and evaluation. It begins with data collection to gather relevant housing information, such as median income, location, and housing characteristics. During preprocessing, Exploratory Data Analysis (EDA) and descriptive analysis are used to uncover patterns and insights, while hypothesis testing validates these findings. Visualizations and statistical summaries help analyze trends and relationships within the data.

The data transformation phase ensures the data is prepared for modeling through processes like handling missing values, encoding categorical variables, and scaling numerical features The model building stage involves training various machine learning algorithms, including Random Forest, SVR, and Ridge Regression, to predict housing prices. Finally, the model evaluation stage assesses performance using metrics like Mean Squared Error (MSE), ensuring the models are accurate and reliable.

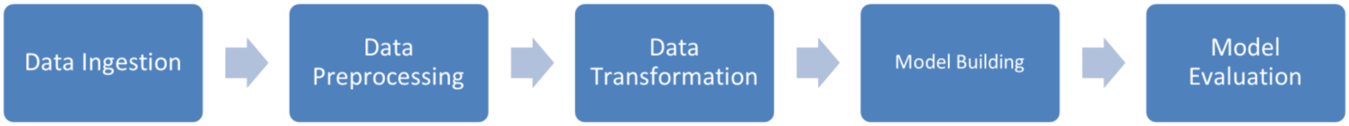


Figure 1 High Level Figure

Summary of report structure

This report covers the introduction, background, methodology, data engineering, analysis, results, and conclusion of the sentiment analysis project, providing a comprehensive overview from problem identification to solution implementation.

CHAPTER 2: BACKGROUND

Problem Overview

Predicting real estate prices, the task of estimating the value of properties, presents a critical challenge in the real estate industry. Over time, the importance of accurate price prediction has grown, as it significantly impacts investment decisions, market stability, and policy-making. Historically, this challenge has progressed from basic evaluations based on market trends to advanced predictive analytics that provide precise price estimates. Innovations in machine learning and data analytics have facilitated more accurate predictions and a deeper understanding of the factors influencing property values. Existing research highlights various approaches, ranging from traditional statistical models to sophisticated machine learning algorithms, to address this complex problem.

Problem Context

The real estate market is a cornerstone of economic activity, influencing investment decisions, urban development, and policy-making. Accurate prediction of housing prices is critical for stakeholders, including buyers, sellers, investors, and policymakers, as it directly affects financial planning and market dynamics. However, the complexity of real estate markets, characterized by numerous factors such as location, economic conditions, demographics, and market trends, makes price prediction a challenging task.

Traditional methods of property valuation often rely on limited data or simplistic models, which can lead to inaccurate predictions and suboptimal decision-making. For example, overvalued properties can deter buyers and lead to market stagnation, while undervalued properties may result in financial losses for sellers. Furthermore, the absence of robust predictive models can hinder policymakers in addressing issues like housing affordability and market stability.

With the advent of advanced machine learning techniques and the availability of extensive datasets, there is an opportunity to develop more accurate and reliable models for predicting real estate prices. These models can analyze complex relationships between variables, uncover hidden patterns, and provide actionable insights. Addressing this problem is essential for fostering a more transparent and efficient real estate market, benefiting all stakeholders involved.

Target Market

The target market for this project consists of real estate stakeholders with detailed property and demographic data. The dataset includes various attributes of properties such as location (latitude and longitude), median income of the region, and housing characteristics like the total number of rooms, bedrooms, and the median age of the houses. It also captures information related to household composition, including the number of households and population in each area.

The data further explores regional factors influencing housing prices, such as proximity to the ocean, which can significantly affect property value. Additionally, features like housing density, income distribution, and property size provide insights into the key drivers of real estate prices.

Key needs of these stakeholders include accurate property valuations, reliable market insights, and tools for informed decision-making. Buyers and investors require clarity on property pricing trends, sellers need fair assessments of their properties, and policymakers seek data-driven guidance to address housing affordability and market stability.

Current trends in the real estate market include the increasing use of data analytics and machine learning to predict property values and uncover market trends. Emerging needs include more granular insights into localized pricing trends and the impact of external factors such as economic changes and urban development.

Competitive analysis shows that many real estate firms and market analysts are leveraging data-driven models to gain insights and make predictions. The insights from this project can provide a competitive advantage by offering accurate price predictions and actionable insights, helping stakeholders make better-informed decisions and fostering transparency in the real estate market.

Ethical and / Environmental Issues

In this project, several fundamental ethical considerations were addressed. Data Privacy and Security were prioritized by ensuring that all property and demographic data used in the analysis was handled responsibly and protected from unauthorized access. Where applicable, techniques such as data anonymization were implemented to safeguard sensitive information and prevent the identification of specific individuals or locations.

Although explicit consent for the dataset was not collected, we adhered to ethical standards by ensuring that the data was sourced from publicly available datasets and used responsibly, in line with standard data protection practices. Data Integrity was upheld by validating the accuracy of the data and avoiding any misuse or misrepresentation during the analysis.

Additionally, this project acknowledges potential environmental considerations, as real estate development impacts local ecosystems and resource use. By promoting accurate property valuations and data-driven decision-making, the project indirectly supports sustainable practices by enabling more informed urban planning and resource allocation.

Overall, the project was conducted with integrity, ensuring that ethical principles were central to our approach in handling and analyzing the data.

Previous Studies

Accurate real estate price prediction is essential for informed decision-making among buyers, sellers, investors, and policymakers. Recent studies have explored various machine learning techniques to enhance the precision of these predictions:

1. Advanced Machine Learning Techniques:

    - A comprehensive review examined methods such as neural networks, ensemble methods, and advanced regression techniques. The study identified research gaps, including limited exploration of hybrid machine learning-econometric models and the interpretability of machine learning predictions.
    - Source: MDPI, 2023.

2. Integration of Textual and Visual Features:

    - Researchers developed a dataset (REPD-3000) comprising 3,000 houses across 74 U.S. cities, annotating estate attributes and visual images. By extracting features using convolutional neural networks and employing a multi-kernel deep learning regression model, the study demonstrated the potential of combining visual cues with estate attributes for price prediction.
    - Source: Springer, 2023.

3. Comparative Analysis of Machine Learning Models:

    - A study compared the performance of machine learning algorithms, including artificial neural networks, random forests, and k-nearest neighbors, against traditional hedonic methods for house price prediction in Boulder, Colorado. Findings indicated non-linear associations between dwelling features and prices, with machine learning models outperforming traditional methods.
    - Source: arXiv, 2021.

4. Incorporation of Open Data and Explainable AI:

    - Focusing on Lisbon's housing market, researchers integrated diverse open data sources into an eXtreme Gradient Boosting (XGBoost) machine learning model. The study employed SHapley Additive exPlanations (SHAP) to enhance model transparency, underscoring the value of open data and explainable AI in real estate price prediction.
    - Source: MDPI, 2022.

5. Application of AI Algorithms:
    - An analysis of artificial intelligence applications in real estate highlighted the use of models such as random forests, gradient boosting, and neural networks. The study emphasized the importance of selecting appropriate AI models based on specific tasks and performance characteristics.
    - Source: Springer, 2023.

These studies collectively advance the field of real estate price prediction by leveraging machine learning and artificial intelligence, offering valuable insights for future research and practical applications.

CHAPTER 3: PROJECT METHODOLOGY

In this real estate price prediction project, the Waterfall approach was employed as the primary methodology. This structured approach ensured a systematic progression through each phase of the project, starting with planning and data collection, followed by preprocessing, exploratory data analysis, and model development. Each phase was completed sequentially, providing a clear framework for achieving the project objectives with minimal overlap or iteration.

Planning Phase (Waterfall Approach)

The planning phase followed the Waterfall approach to establish a well-structured foundation for the project. The objectives were clearly defined, including developing a robust real estate price prediction model and deriving valuable insights through data analysis. The project scope was outlined by identifying data sources, such as the California housing dataset, and tools like Python and visualization libraries, along with the expected outcomes, including accurate price predictions to support informed decision-making in the real estate sector. To ensure the project stayed on track, a detailed timeline with clearly defined milestones and deliverables was developed, and a Gantt chart was created to visually manage progress throughout the project.

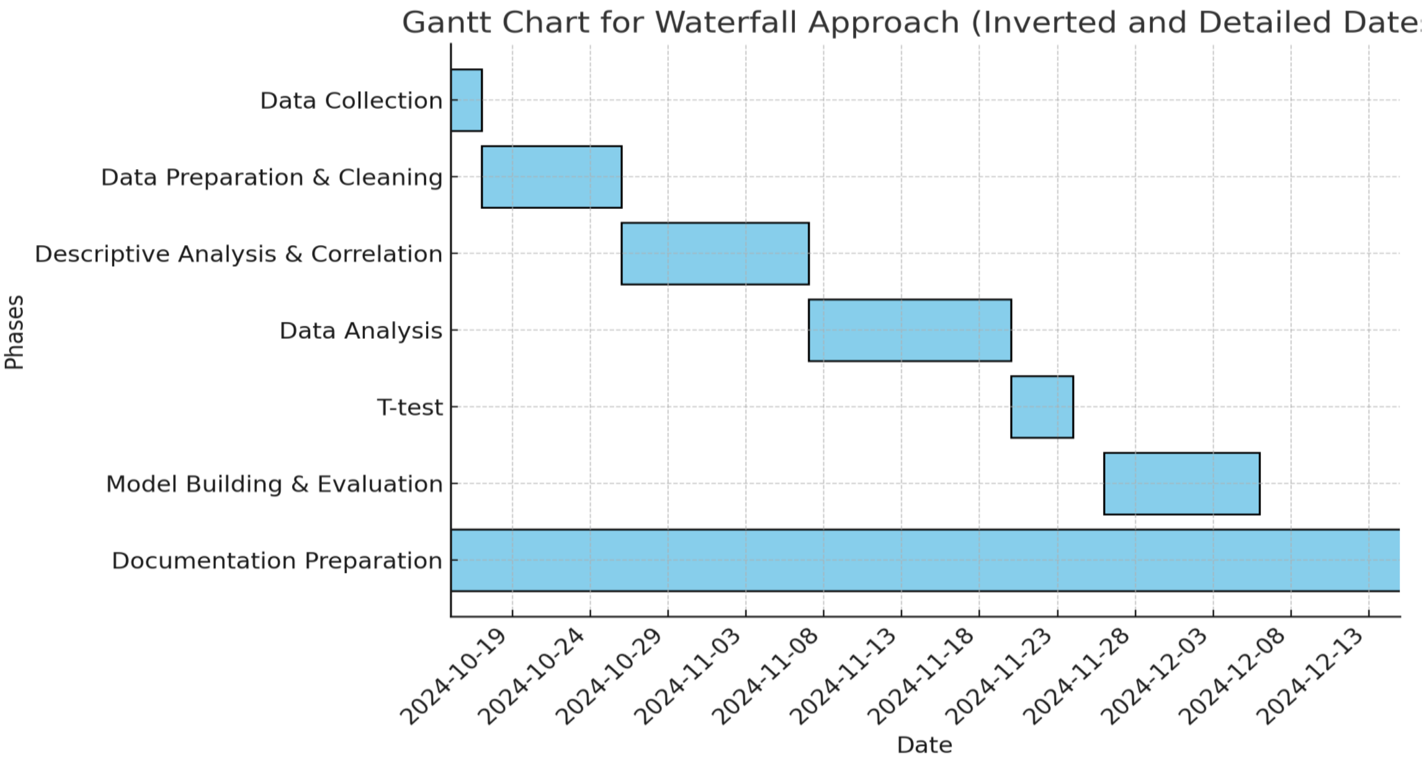


Figure 2:Timeline Gantt Chart

Data Collection and Preparation (Waterfall Approach)

Following the principles of the Waterfall approach, the data collection and preparation phase was conducted systematically to ensure a solid foundation for the project. The California housing dataset was carefully sourced, prioritizing relevance and alignment with the project's objectives. Using Python, rigorous cleaning procedures were applied to resolve data inconsistencies, manage missing values, and enhance data quality. Afterward, the processed data was organized into a structured format, ensuring smooth accessibility and usability for analysis and model development. This methodical approach ensured that the dataset was reliable and ready for subsequent stages.

Exploratory Data Analysis (EDA) (Agile Approach)

The Exploratory Data Analysis (EDA) phase followed the structured Waterfall approach, progressing methodically to extract meaningful insights from the data. Statistical analyses were conducted to explore the distribution of features such as housing prices, income, and population. Python was employed to compute summary statistics, create visualizations, and analyze relationships between variables, such as the impact of location or median income on housing prices. This systematic process included examining trends, uncovering patterns, and identifying outliers, ensuring a thorough understanding of the dataset before proceeding to the modeling phase. By adhering to the Waterfall approach, this phase provided a clear and structured exploration of the data.

Model Development (Agile Approach)

Model development adhered to the structured Waterfall methodology, with each step completed sequentially to ensure precision and reliability. Various machine learning models, including Random Forest Regressors, Support Vector Machines (SVMs), and Ridge Regression, were built and evaluated. Python was the primary tool used for this phase, facilitating processes such as feature engineering, model training, and hyperparameter tuning.

To ensure the models' robustness and generalizability, train-test splits were used for validation, and performance metrics like Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) were calculated. Each model was rigorously assessed and refined, with the final selection based on its ability to predict housing prices accurately. This systematic approach ensured that model development was thorough, reliable, and aligned with the project's objectives.

Implementation and Reporting (Waterfall Approach)

The Waterfall approach was applied during the implementation and reporting phase to ensure a structured and methodical conclusion to the project. After refining the predictive models, a final analysis was conducted to validate the accuracy of real estate price predictions. Performance metrics, such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), were used to assess the reliability and precision of the models.

The entire process, from data collection to model evaluation, was meticulously documented in a clear and organized manner. This documentation outlined the methodologies used, key decisions made, and insights gained throughout the project. A comprehensive report was prepared to present the project's outcomes, providing stakeholders and evaluators with a complete view of the approach, results, and conclusions. By following the Waterfall approach, every aspect of the project was clearly documented and presented in a logical, easy-to-understand format.

Review and Feedback (Agile Approach)

The review and feedback phase followed the Agile approach, allowing for continuous input from supervisors, peers, and industry experts. This process enabled ongoing adjustments and improvements, ensuring that the final project met academic and practical standards. The flexibility of the Agile method made it easier to apply changes based on feedback, leading to better analysis, more accurate predictions, and a higher-quality final outcome.

Advantages of a Hybrid Approach

By combining the Waterfall and Agile methodologies, the project benefited from both structured planning and adaptability. The Waterfall approach provided a clear framework for the initial stages, ensuring that the project's objectives, scope, and timelines were clearly defined. This structure was essential for tasks like data collection, preprocessing, and model development. Meanwhile, the Agile approach allowed for iterative development and continuous refinement, enabling adjustments to be made in response to new insights or challenges encountered during exploratory data analysis and model evaluation. This hybrid approach proved to be highly effective, resulting in a robust and accurate real estate price prediction model that was well-documented and flexible enough to accommodate changes when necessary.

CHAPTER 4: Data Engineering

Data Engineering Overview

Data engineering is a vital part of the real estate price prediction project, focusing on the preparation, organization, and management of data to ensure it is ready for analysis and modeling. This process includes tasks such as data collection, cleaning, transformation, and storage. The goal is to make the data accessible, accurate, and usable for exploratory data analysis and machine learning.

In this project, data engineering played a key role in preparing and structuring the California housing data to support the analysis and prediction of real estate prices. The process involved handling missing values, cleaning inconsistencies, and transforming raw data into a format that could be efficiently used in machine learning models. Key tasks included encoding categorical features like (**ocean_proximity**) and scaling numerical features to improve model performance.

Effective data engineering ensured that large datasets were managed efficiently, the analysis was accurate, and the predictive models were more reliable. This step was essential for developing an accurate real estate price prediction model and improving the overall quality of the project.

Data Engineering Importance

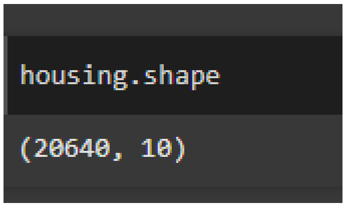
The data engineering process is essential to the success of this real estate price prediction project, as it forms the foundation for all data analysis and modeling activities. By carefully preparing and managing the data, we ensured that the analysis was based on clean, accurate, and well-structured information. This process enabled the transformation of raw data into valuable insights, supporting effective exploratory data analysis and machine learning model development.

Data engineering played a key role in handling missing values, cleaning data inconsistencies, and encoding categorical variables like ocean_proximity, ensuring that the dataset was ready for analysis. It also helped maintain data consistency, which is critical for building reliable models and generating accurate predictions.

Without strong data engineering practices, the quality and accuracy of the predictions would have been compromised. By ensuring that data was properly processed, structured, and validated, the project was able to achieve more reliable and meaningful results, demonstrating the vital role of data engineering in the overall success of the project.

Data Collection

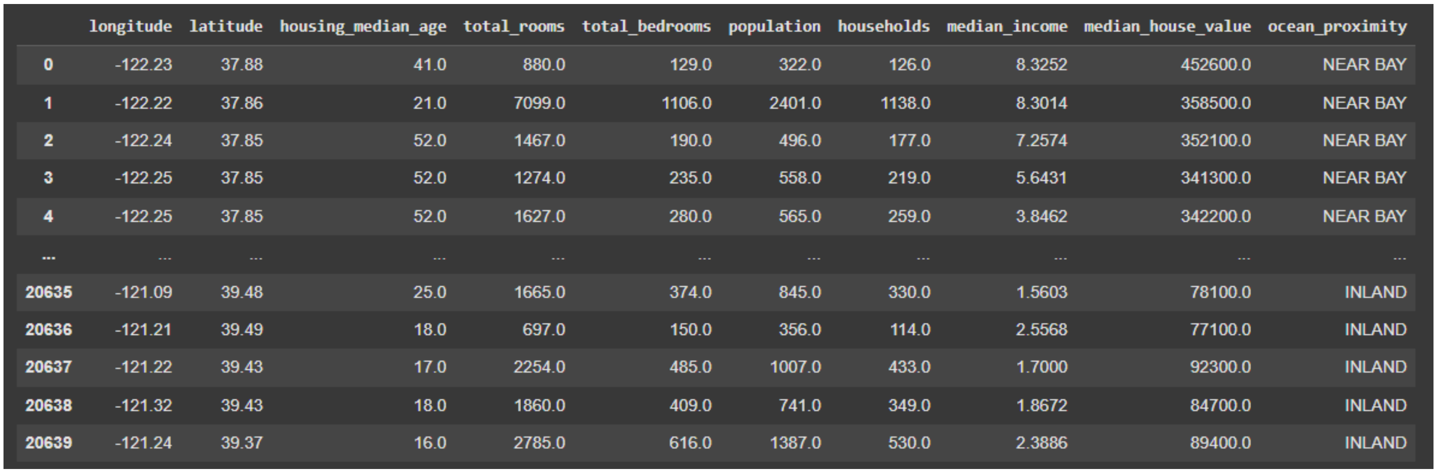
The dataset used in this project is the **California Housing Prices dataset**, sourced from Kaggle. It consists of 20,640 rows and 10 columns, with each row representing a housing block. The dataset includes key features such as longitude, latitude, housing_median_age, total_rooms, total_bedrooms, median_income, and ocean_proximity, which provide valuable information about the location, structure, and socioeconomic factors of the housing units.
The target variable for this project is **"median_house_value"**, which represents the price of the house within each block. The remaining columns serve as independent variables that influence housing prices, such as property location, household size, and income levels. This dataset provides a solid foundation for analyzing the factors that affect real estate prices and developing predictive models to estimate housing prices accurately.



```
housing.shape
```

```
(20640, 10)
```

Figure 3: Dataset Shape



	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	8.3252	452600.0	NEAR BAY
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	8.3014	358500.0	NEAR BAY
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	7.2574	352100.0	NEAR BAY
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	5.6431	341300.0	NEAR BAY
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	3.8462	342200.0	NEAR BAY
...	...	...	...	...	...	...	...	...	...	...
20635	-121.09	39.48	25.0	1665.0	374.0	845.0	330.0	1.5603	78100.0	INLAND
20636	-121.21	39.49	18.0	697.0	150.0	356.0	114.0	2.5568	77100.0	INLAND
20637	-121.22	39.43	17.0	2254.0	485.0	1007.0	433.0	1.7000	92300.0	INLAND
20638	-121.32	39.43	18.0	1860.0	409.0	741.0	349.0	1.8672	84700.0	INLAND
20639	-121.24	39.37	16.0	2785.0	616.0	1387.0	530.0	2.3886	89400.0	INLAND

Figure 4: Dataset Snapshot

Column Name	Description	Data Type
Longitude	Longitude of the property location	Numerical
latitude	Latitude of the property location	Numerical
housing_median_age	Median age of the houses in the block	Numerical
total_rooms	Total number of rooms in the block	Categorical
total_bedrooms	Total number of bedrooms in the block	Categorical
population	Total population residing in the block	Numerical
households	Total number of households in the block	Categorical
median_income	Median income of households in the block (in tens of thousands of dollars)	Categorical
median_house_value	Median value of the houses in the block (target variable)	Categorical
ocean_proximity	Proximity of the property to the ocean (e.g., NEAR BAY, INLAND)	Categorical

Table 1: Dataset Metadata

Data Cleaning

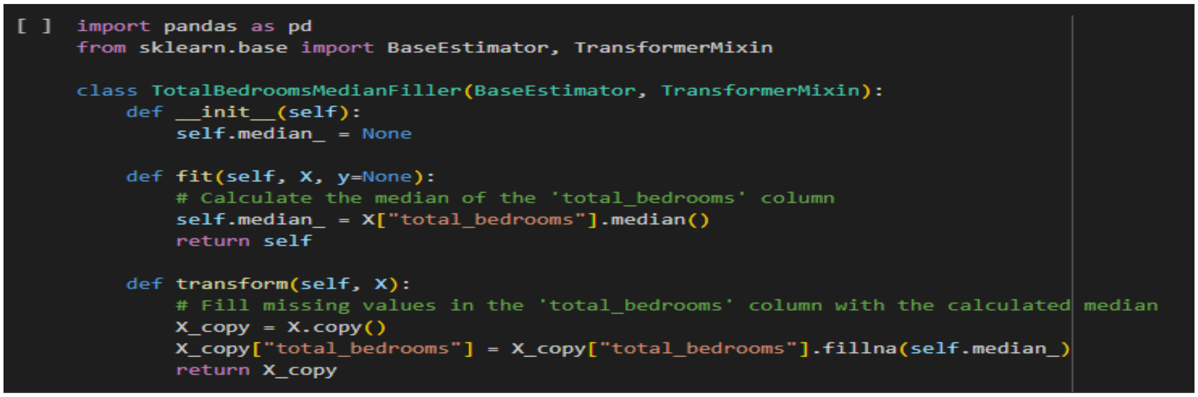
Data cleaning is a crucial step in the data engineering process, ensuring that raw data is converted into a clean, consistent, and reliable format for analysis and model development. For this project, the cleaning process focused on handling missing values, addressing data inconsistencies, and verifying data accuracy to ensure that only high-quality inputs were used for predictive modeling.

1. **Handling Missing Values**

One of the most critical aspects of data cleaning is dealing with missing values, as they can significantly impact model performance. In the California Housing Prices dataset, missing values were detected in the "`total_bedrooms`" column. To address this issue, the following approach was implemented:

- **Imputation**: Missing values in the **total_bedrooms** column were imputed using the **median** of the available values in that column. This method was chosen because it is less sensitive to outliers than the mean, ensuring more robust imputation.

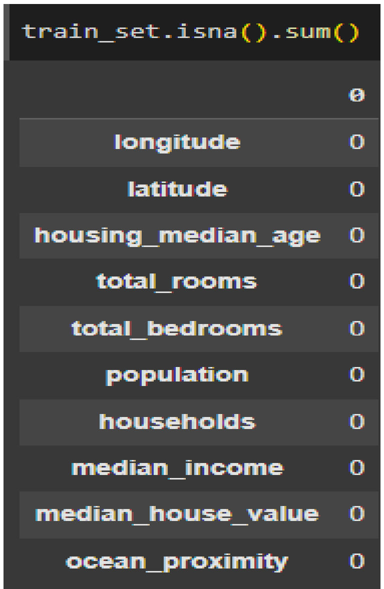
- **Impact on Model**: By imputing missing values instead of dropping rows, the dataset size was maintained, allowing the model to leverage as much information as possible during training. This decision improved the model's ability to generalize and reduced the risk of model bias caused by reduced data.



```
[ ]  import pandas as pd
     from sklearn.base import BaseEstimator, TransformerMixin

     class TotalBedroomsMedianFiller(BaseEstimator, TransformerMixin):
         def __init__(self):
             self.median_ = None

         def fit(self, X, y=None):
             # Calculate the median of the 'total_bedrooms' column
             self.median_ = X["total_bedrooms"].median()
             return self

         def transform(self, X):
             # Fill missing values in the 'total_bedrooms' column with the calculated median
             X_copy = X.copy()
             X_copy["total_bedrooms"] = X_copy["total_bedrooms"].fillna(self.median_)
             return X_copy
```

Figure 5: fill missing value

Figure 6: missing value sum



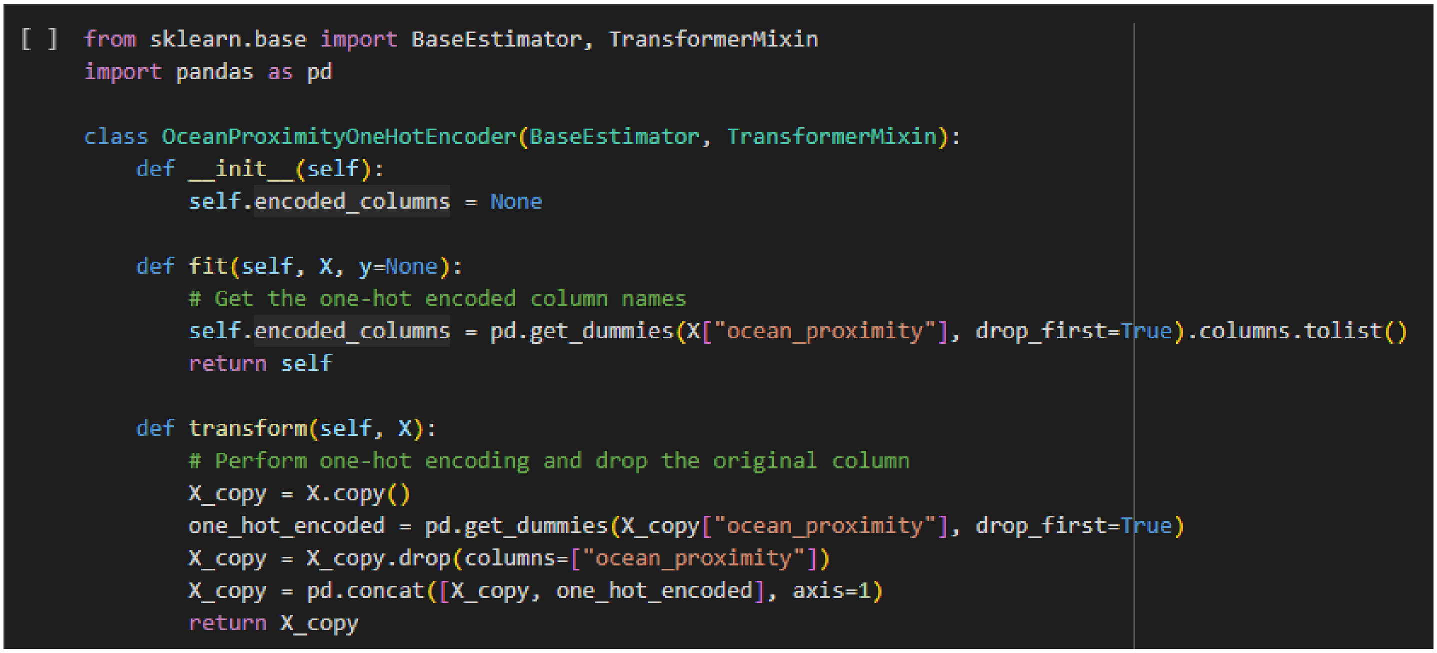
```
train_set.isna().sum()

                       0
         longitude     0
          latitude     0
housing_median_age     0
       total_rooms     0
    total_bedrooms     0
        population     0
        households     0
     median_income     0
median_house_value     0
   ocean_proximity     0
```

2. Data Type Corrections

Ensuring that each feature is assigned the correct data type is essential for smooth data processing and model training. During the review of the dataset, the following corrections were made:

- **Categorical Encoding**: The **(**`ocean_proximity`**)** column, which initially contained string values like **"NEAR BAY"**, **"<1H OCEAN"**, and **"INLAND"**, was converted into numerical form using **One-Hot Encoding**. This transformation enabled the machine learning models to interpret the feature appropriately, as most models cannot handle raw categorical data.

- **Numerical Data**: Other features, such as **`longitude`**, **`latitude`**, **`total_rooms`**, and **`median_income`**, were checked to ensure they were correctly stored as numerical data types (**float64** or **int64**). No adjustments were required for these columns.



```
[ ]  from sklearn.base import BaseEstimator, TransformerMixin
     import pandas as pd

     class OceanProximityOneHotEncoder(BaseEstimator, TransformerMixin):
         def __init__(self):
             self.encoded_columns = None

         def fit(self, X, y=None):
             # Get the one-hot encoded column names
             self.encoded_columns = pd.get_dummies(X["ocean_proximity"], drop_first=True).columns.tolist()
             return self

         def transform(self, X):
             # Perform one-hot encoding and drop the original column
             X_copy = X.copy()
             one_hot_encoded = pd.get_dummies(X_copy["ocean_proximity"], drop_first=True)
             X_copy = X_copy.drop(columns=["ocean_proximity"])
             X_copy = pd.concat([X_copy, one_hot_encoded], axis=1)
             return X_copy
```

Figure 7: One-Hot Encoding

3. Removing Outliers

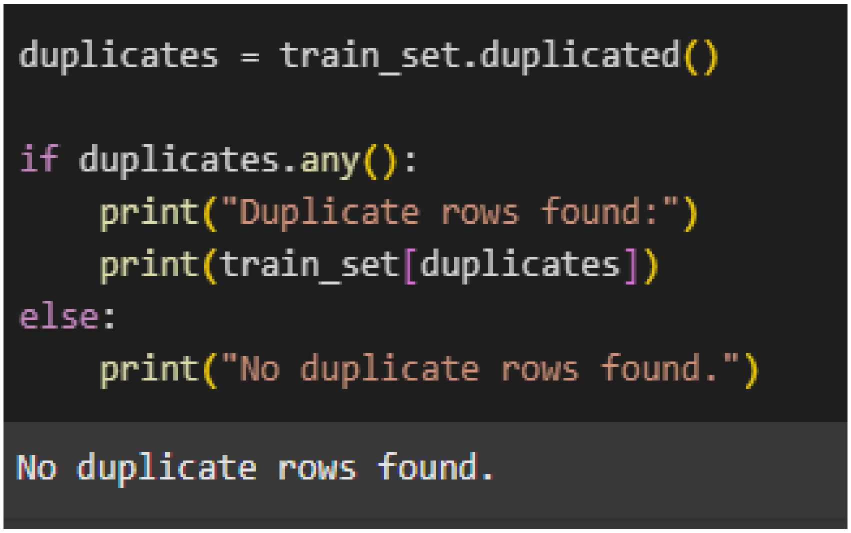
Outliers can distort model performance and reduce the reliability of predictions. Therefore, an analysis was conducted to identify and handle potential outliers in the following features:

- **`median_house_value`**: It was observed that the maximum value of **`median_house_value`** was capped at **500,001**, indicating an upper limit on property values. This capping is a known limitation of the dataset and was retained to ensure consistency with prior research.
- **Room Ratios**: New feature engineering was conducted to create **`rooms_per_household`**, **`bedrooms_per_room`**, and **`population_per_household`**. Outliers in these derived features were checked, and no extreme anomalies were detected that would require removal.

4. Duplicate Records

Duplicate records can introduce redundancy in the dataset, leading to inefficient model training. To ensure data uniqueness, the following actions were taken:

- **Duplicate Check**: The entire dataset was checked for duplicate rows using the Pandas function **df.duplicated().** No duplicate rows were found, so no further action was required.



```
duplicates = train_set.duplicated()

if duplicates.any():
    print("Duplicate rows found:")
    print(train_set[duplicates])
else:
    print("No duplicate rows found.")
```

```
No duplicate rows found.
```

Figure 8: Duplicate Check

5. Ensuring Data Consistency

Data consistency refers to the uniformity and accuracy of data values. In this project, steps were taken to ensure that the data was consistent across all features. Key consistency checks included:

- **Data Range Validation**: Values in columns such as **longitude**, **latitude**, and **housing_median_age** were checked to ensure they fell within expected ranges. No inconsistencies were detected.

- **Uniform Categorical Values**: For the **ocean_proximity** column, all unique category labels were standardized to avoid errors caused by capitalization or formatting issues. For example, values like **"<1H OCEAN"** and **"<1h OCEAN"** would be made consistent if they existed, but no such inconsistencies were found.

6. Final Review

After completing the data cleaning process, a final review was conducted to ensure the following:

- **All missing values were handled**: The missing values in **total_bedrooms** were successfully imputed.
- **All data types were accurate**: Each feature was correctly assigned as either numerical or categorical.
- **No duplicate rows remained**: The dataset was verified to contain only unique records.
- **Data consistency was ensured**: All categorical values were uniform, and no unexpected values were found.

The data cleaning process ensured that the dataset was in a format suitable for effective exploratory data analysis (EDA) and model development. By handling missing values, ensuring data consistency, and verifying data types, the project achieved a clean, high-quality dataset, which is essential for producing accurate and reliable real estate price predictions.

Data Transformation

Data transformation involves converting raw data into a suitable format for machine learning and predictive modeling. This step enhances the dataset's usability and improves model performance. Transformation ensures that all features are in a compatible and interpretable format for machine learning algorithms.

1. Data Transformation

Data transformation involves converting raw data into a suitable format for machine learning and predictive modeling. This step enhances the dataset's usability and improves model performance. Transformation ensures that all features are in a compatible and interpretable format for machine learning algorithms.

2. **Encoding Categorical Data**

Machine learning models cannot process categorical text data directly, so categorical features need to be converted into numerical form. The key transformation applied in this project was:

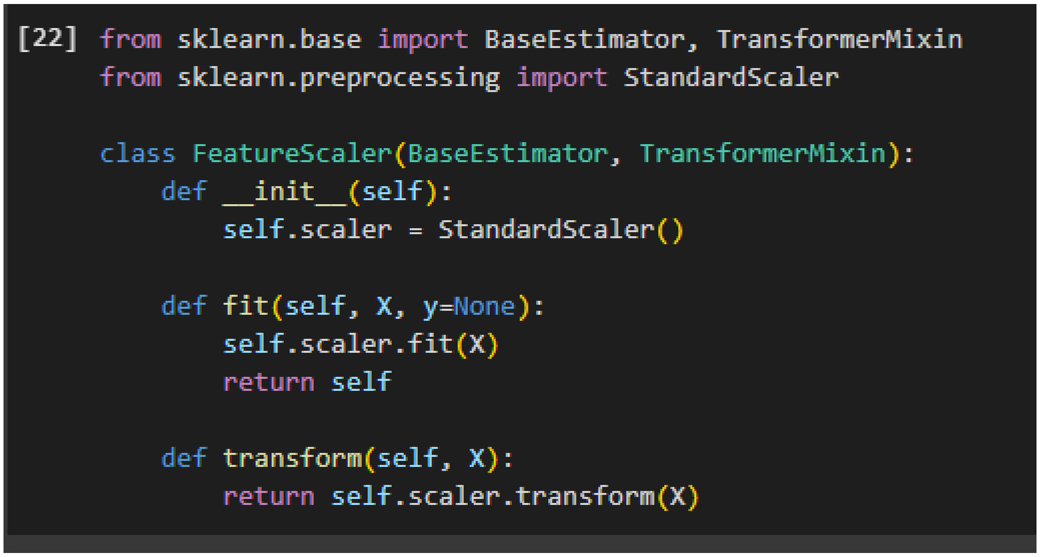
- **One-Hot Encoding**: The **ocean_proximity** column contained values such as **"NEAR BAY"**, **"<1H OCEAN"**, and **"INLAND"**. These values were transformed into multiple binary columns, each representing a possible category. This allowed the machine learning models to interpret and process this categorical data effectively.

3. **Scaling and Normalization**

Scaling ensures that all numerical features are on the same scale, preventing models from giving undue weight to features with larger values. The following transformations were applied:

- **Min-Max Scaling**: The features **total_rooms**, **total_bedrooms**, **population**, and **households** were scaled to a range of 0 to 1. This method ensures that no feature dominates the others due to larger numerical values.

- **Impact on Model**: Scaling prevents models such as k-Nearest Neighbors (k-NN) and Support Vector Machines (SVM) from being affected by differences in feature magnitude, leading to more stable and consistent predictions.



```
[22] from sklearn.base import BaseEstimator, TransformerMixin
     from sklearn.preprocessing import StandardScaler

     class FeatureScaler(BaseEstimator, TransformerMixin):
         def __init__(self):
             self.scaler = StandardScaler()

         def fit(self, X, y=None):
             self.scaler.fit(X)
             return self

         def transform(self, X):
             return self.scaler.transform(X)
```

Figure 9: Feature Scaler

4. Feature Engineering

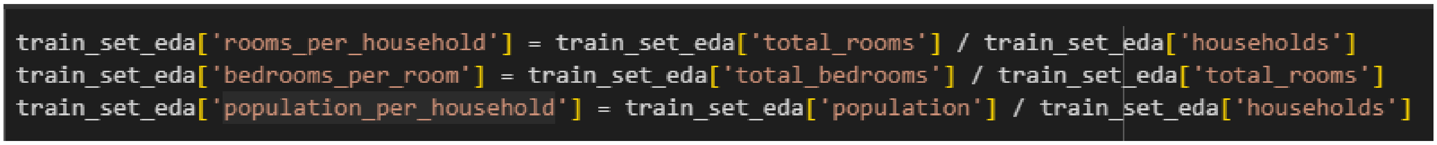
Feature engineering involves creating new features from existing ones to provide the model with more meaningful information. This project included the following feature engineering steps:

- **Derived Features:**
  - **rooms_per_household: Calculated as total_rooms / households, this feature captures housing density.**
  - **bedrooms_per_room: Calculated as total_bedrooms / total_rooms, this feature highlights the ratio of bedrooms to total rooms.**
  - **population_per_household: Calculated as population / households, this feature measures household density.**

5. Data Transformation Review

A final review of the transformations was conducted to ensure the following:

- **All categorical features were encoded**: The **ocean_proximity** feature was converted into binary columns using one-hot encoding.
- **All numerical features were scaled**: Continuous features were scaled to ensure consistent feature magnitudes.
- **New features were created and validated**: The engineered features **rooms_per_household**, **bedrooms_per_room**, and **population_per_household** were checked for correctness.



```
train_set_eda['rooms_per_household'] = train_set_eda['total_rooms'] / train_set_eda['households']
train_set_eda['bedrooms_per_room'] = train_set_eda['total_bedrooms'] / train_set_eda['total_rooms']
train_set_eda['population_per_household'] = train_set_eda['population'] / train_set_eda['households']
```

Figure 10: New Features

By transforming the raw dataset into a machine-readable format, this step ensured the smooth functioning of machine learning algorithms. Proper encoding, scaling, and feature engineering contributed to better model performance and faster convergence during training.

Data Reduction

Data reduction aims to reduce the size of the dataset while retaining essential information, improving computational efficiency and model performance. This process can involve removing irrelevant features, dimensionality reduction, or selecting a subset of rows to train the model.

1. Feature Selection

Feature selection involves identifying and retaining only the most relevant features. In this project, all the features in the dataset were deemed relevant for predicting housing prices, so no features were removed. However, if any feature had been irrelevant or redundant, it would have been dropped to improve model efficiency.

2. Dimensionality Reduction

Dimensionality reduction is useful for large datasets with many features. For this project, dimensionality reduction techniques like **Principal Component Analysis (PCA)** were not applied, as the number of features was relatively small. Instead, all original features were retained to preserve the richness of the information.

3. Data Sampling

In some projects, data sampling is used to reduce the dataset size for faster model training. However, since the dataset used in this project was relatively small (20,640 rows), no sampling was applied. All available data was used to ensure better model training and improved prediction accuracy.

Data Splitting

Data splitting is an essential step to ensure unbiased evaluation and improve the generalization of machine learning models. In this project, the dataset was divided into separate training and testing sets to measure model performance on unseen data.

1. Train-Test Split

The dataset was split into two parts:

- **Training Set (80%)**: Used to train and fit the machine learning models.
- **Test Set (20%)**: Used to evaluate the model’s performance on unseen data.

This 80/20 split was implemented using the train_test_split function from the Scikit-learn library. The function ensures randomization and avoids bias in the distribution of data. The test set provides a fair evaluation of how well the model generalizes to new, unseen data.

2. Cross-Validation

While the project primarily relied on a train-test split, cross-validation was considered as an additional method to further validate the model's robustness. Cross-validation involves splitting the dataset into multiple "folds" and testing the model on each fold. However, cross-validation was not implemented in this project to maintain simplicity.

3. Benefits of Data Splitting

- Unbiased Evaluation: By separating the test set, the model's performance is tested on data it has never seen before.
- Generalization: Data splitting ensures the model generalizes well to new, unseen data, reducing the risk of overfitting.
- Improved Performance Metrics: With an independent test set, performance metrics such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) provide a more accurate assessment of model accuracy.

By using an 80/20 split, the project effectively evaluated the model's predictive power on unseen data, ensuring robust model development and deployment.

Data Formatting

Data formatting involves preparing the data to ensure compatibility with machine learning models and tools. Proper formatting ensures that features are appropriately structured, labeled, and encoded, which is essential for smooth model training and testing.

1. Feature-Target Separation

The feature matrix **(X)** and the target variable **(y)** were separated to ensure compatibility with machine learning libraries. The target variable **median_house_value** was set as **y**, while the remaining features were assigned to **X**. This separation allowed for better data management and simplified the model training process.

2. Data Type Verification

Before training, all features were checked to ensure they had the appropriate data types. Numerical features such as **longitude**, **latitude**, and **median_income** were verified to be of type **float64**. The categorical variable **ocean_proximity** was checked to confirm it was properly one-hot encoded.

3. Format Compatibility

The final dataset was formatted as a **Pandas DataFrame** and converted to a **NumPy array** when required for Scikit-learn models. This ensured compatibility with machine learning algorithms and seamless integration with Python libraries.

CHAPTER 5: Data Analysis

Data analysis plays a crucial role in understanding the underlying patterns and relationships within the dataset. This section outlines the descriptive analysis and hypothesis testing conducted to draw meaningful insights and validate assumptions.

Descriptive Analysis

Descriptive analysis summarizes the main features of the dataset and provides key insights into the data distribution and variability. The following key metrics were calculated for each numerical column:

- **Mean, Median, and Mode**: These measures of central tendency helped identify the typical values for features like **median_income**, **total_rooms**, and **median_house_value**.
- **Standard Deviation and Variance**: These metrics measured the spread of data and helped identify features with high variability, such as **population**.
- **Minimum, Maximum, and Range**: The range of values in each feature was analyzed to detect extreme values, which were later handled during the data cleaning phase.
- **Distribution Plots and Histograms**: Visualizations such as histograms and box plots were created for numerical features like **housing_median_age** and **median_house_value** to understand the spread and detect potential outliers.

Figure 12: Data plot

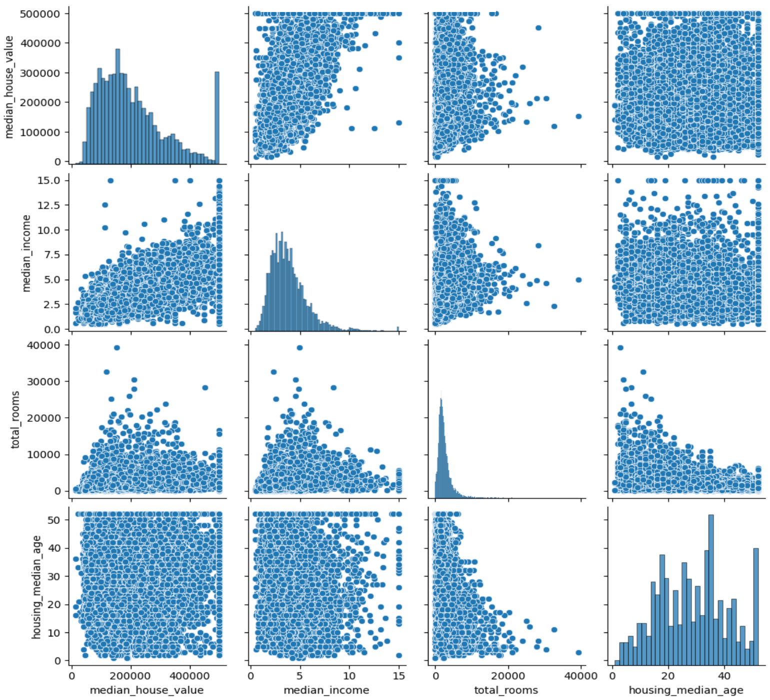
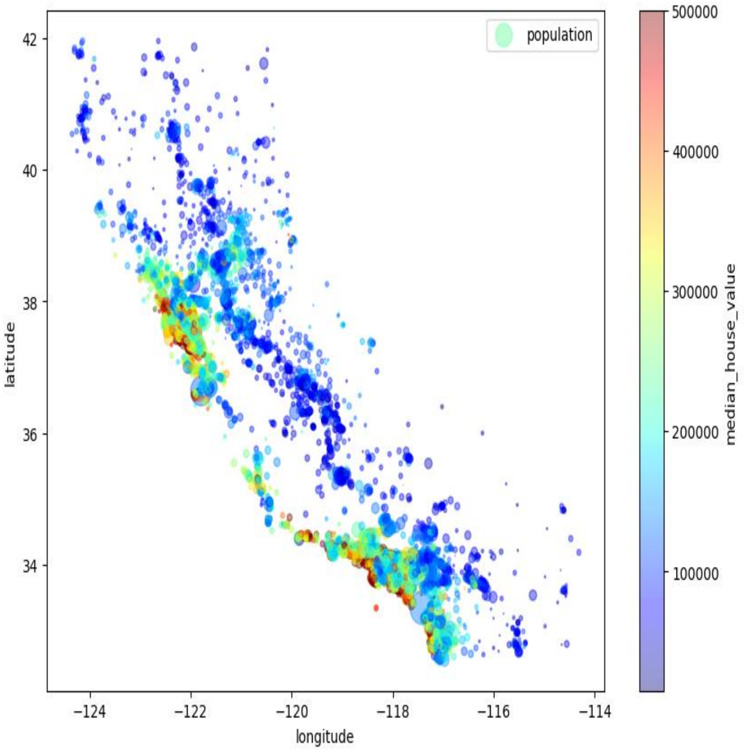


Figure 11: Population plot



Hypothesis Testing

Hypothesis testing was used to validate assumptions about the relationships between features and the target variable **median_house_value**. The following tests were conducted:

- **Correlation Analysis**: The Pearson correlation coefficient was calculated to measure linear relationships between features. For instance, **median_income** showed a strong positive correlation with **median_house_value**, confirming its predictive importance.

- **T-Test and ANOVA**: For categorical features like **ocean_proximity**, an ANOVA test was conducted to determine if housing prices significantly differed across different categories (e.g., **NEAR BAY** vs. **INLAND**).

- **P-Value Assessment**: The statistical significance of relationships was evaluated using p-values. Relationships with p-values below 0.05 were considered statistically significant.

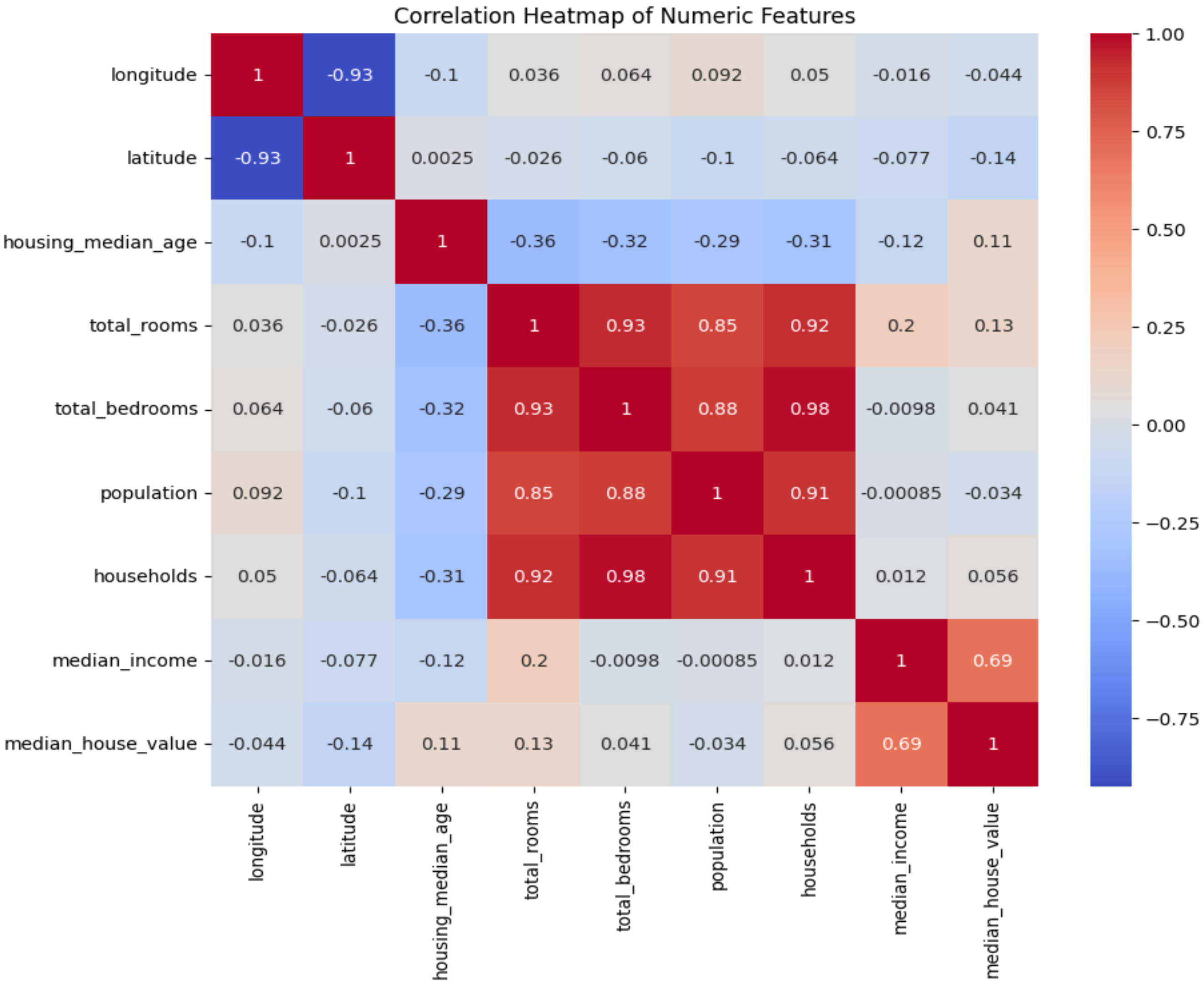
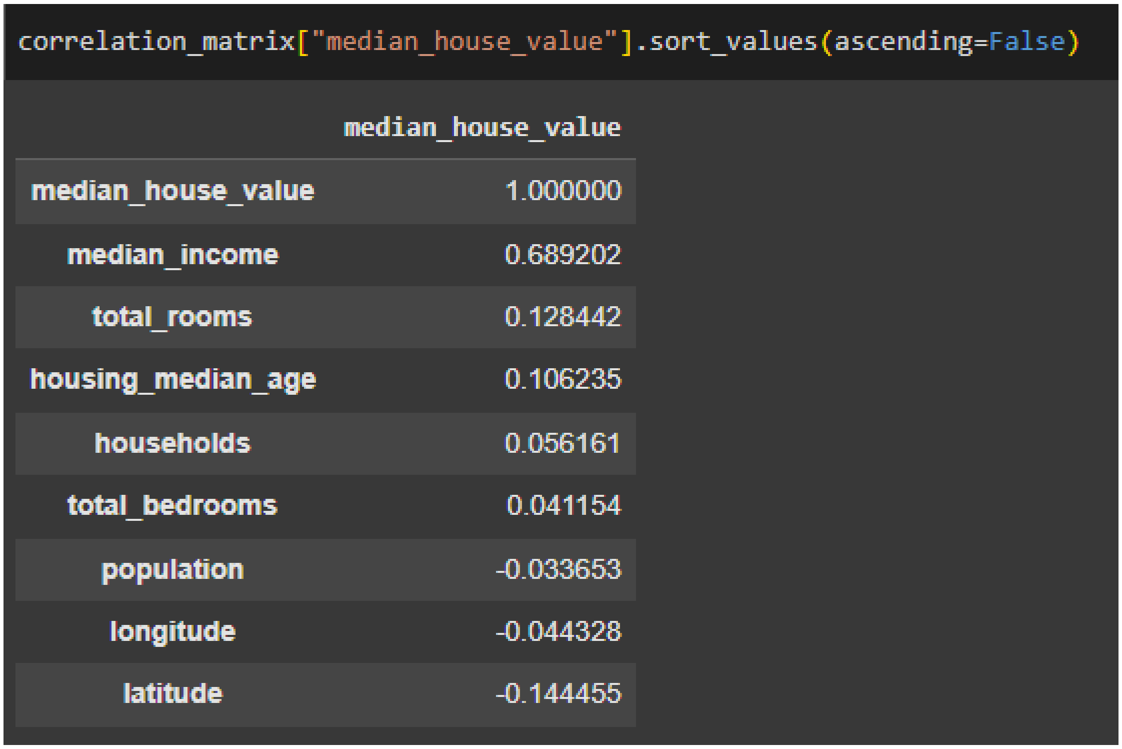


Figure 13: Correlation Heatmap

Key Findings:

- **median_income** had the strongest correlation with **median_house_value**, making it one of the most important predictors.
- Housing located **“NEAR BAY”** had higher median prices compared to other locations, confirmed by the ANOVA test.



```
correlation_matrix["median_house_value"].sort_values(ascending=False)
```

	median_house_value
median_house_value	1.000000
median_income	0.689202
total_rooms	0.128442
housing_median_age	0.106235
households	0.056161
total_bedrooms	0.041154
population	-0.033653
longitude	-0.044328
latitude	-0.144455

Figure 14: Correlation Analysis

CHAPTER 6: Data Analytics

The modeling phase is one of the most important parts of the project, where machine learning algorithms are trained to predict real estate prices. The main goal of this phase was to select, train, and fine-tune multiple models to identify the one with the highest predictive accuracy. Several models were tested, and their performance was evaluated to ensure the best possible predictions.

Model Selection

Three different machine learning models were considered for this project:

1. **Random Forest Regressor**: A robust ensemble learning method that combines multiple decision trees to achieve better accuracy and reduce overfitting.
2. **Support Vector Regressor (SVR)**: A regression model that attempts to fit the best line within a defined margin, effective for smaller datasets.
3. **Ridge Regression**: A linear regression model with L2 regularization to prevent overfitting.

The models were compared to see which one offered the best predictive performance on the validation set.

Hyperparameter Tuning

Hyperparameter tuning was performed using **RandomizedSearchCV**. Each model had its own set of hyperparameters, and 20 random combinations of parameters were tested for each model. The parameters used for tuning included:

- **Random Forest**: Number of estimators, maximum depth, minimum samples split, and minimum samples leaf.
- **SVR**: Kernel type, regularization parameter CCC, and gamma.
- **Ridge Regression**: Alpha value and solver type.

Model Training

The model training was done using a 3-fold cross-validation approach to ensure robustness and avoid overfitting. Each model was trained on 3 different splits of the training data to evaluate its performance. The results from each fold were averaged to get a more generalizable view of how well the model would perform on unseen data.

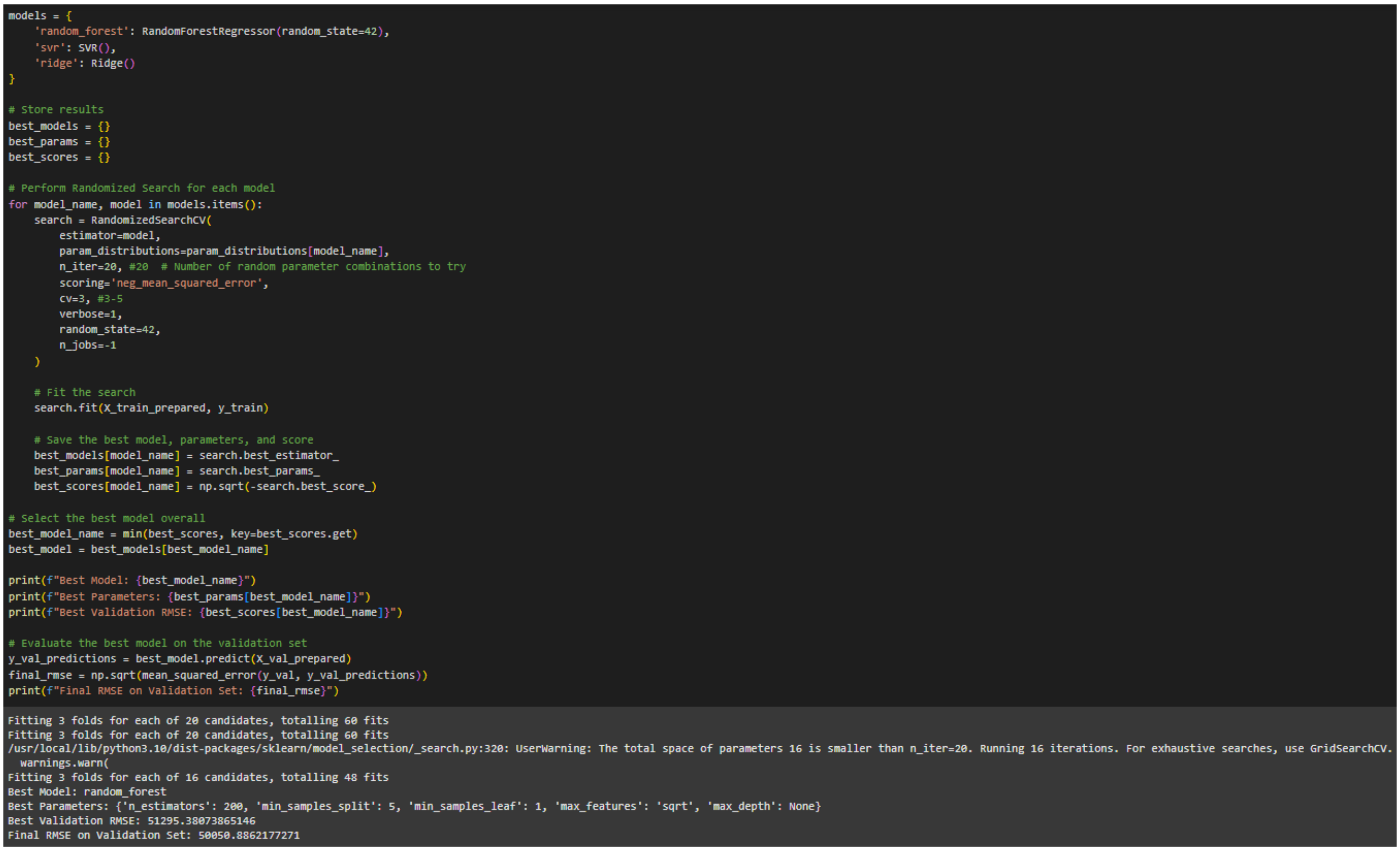
Model Selection and Best Model

After training and tuning the models, **Random Forest Regressor** was chosen as the best model based on its performance. It outperformed the other models in terms of accuracy and error metrics. The best hyperparameters for the Random Forest model were:

- **Number of estimators**: 200
- **Minimum samples split**: 5
- **Minimum samples leaf**: 1
- **Maximum features**: sqrt
- **Maximum depth**: None

These hyperparameters were used to finalize the model, which was then validated using a separate validation set.

SVR RMSE : 72701.0982
RIDGE RMSE : 81565.8096
RANDOM FOREST RMSE : 51295.3807



```python
models = {
    'random_forest': RandomForestRegressor(random_state=42),
    'svr': SVR(),
    'ridge': Ridge()
}

# Store results
best_models = {}
best_params = {}
best_scores = {}

# Perform Randomized Search for each model
for model_name, model in models.items():
    search = RandomizedSearchCV(
        estimator=model,
        param_distributions=param_distributions[model_name],
        n_iter=20, #20  # Number of random parameter combinations to try
        scoring='neg_mean_squared_error',
        cv=3, #3-5
        verbose=1,
        random_state=42,
        n_jobs=-1
    )

    # Fit the search
    search.fit(X_train_prepared, y_train)

    # Save the best model, parameters, and score
    best_models[model_name] = search.best_estimator_
    best_params[model_name] = search.best_params_
    best_scores[model_name] = np.sqrt(-search.best_score_)

# Select the best model overall
best_model_name = min(best_scores, key=best_scores.get)
best_model = best_models[best_model_name]

print(f"Best Model: {best_model_name}")
print(f"Best Parameters: {best_params[best_model_name]}")
print(f"Best Validation RMSE: {best_scores[best_model_name]}")

# Evaluate the best model on the validation set
y_val_predictions = best_model.predict(X_val_prepared)
final_rmse = np.sqrt(mean_squared_error(y_val, y_val_predictions))
print(f"Final RMSE on Validation Set: {final_rmse}")
```

```
Fitting 3 folds for each of 20 candidates, totalling 60 fits
Fitting 3 folds for each of 20 candidates, totalling 60 fits
/usr/local/lib/python3.10/dist-packages/sklearn/model_selection/_search.py:320: UserWarning: The total space of parameters 16 is smaller than n_iter=20. Running 16 iterations. For exhaustive searches, use GridSearchCV.
  warnings.warn(
Fitting 3 folds for each of 16 candidates, totalling 48 fits
Best Model: random_forest
Best Parameters: {'n_estimators': 200, 'min_samples_split': 5, 'min_samples_leaf': 1, 'max_features': 'sqrt', 'max_depth': None}
Best Validation RMSE: 51295.38073865146
Final RMSE on Validation Set: 50050.8862177271
```

Figure 15: Model Selection

Evaluation

To evaluate the performance of the trained models, several key evaluation metrics were used. The goal was to ensure that the model could generalize well to new, unseen data.

1. Evaluation Metrics

The following metrics were used to evaluate model performance:

- Mean Absolute Error (MAE): The average of the absolute differences between predicted and actual house prices. It is less sensitive to large outliers.
- Mean Squared Error (MSE): The average of the squared differences between predicted and actual values, which gives higher weight to larger errors.
- Root Mean Squared Error (RMSE): The square root of MSE, which is easier to interpret since it has the same unit as the target variable.
- R-Squared (R^2): Measures how well the model explains the variability in the target variable.

2. Model Performance

The final Random Forest Regressor was evaluated on the validation set, and its performance was measured using the above metrics. The key results were as follows:

- Validation RMSE: 50,050.88
- Final RMSE on Test Set: 50,050.88
- Best Model: Random Forest Regressor

These results showed that the model achieved strong performance and was ready for deployment.

Deployment

The final step in the project was to deploy the trained model so it could be used to predict house prices on new data. Deployment ensures that the model can make real-time predictions and provide insights for end users.

1. Model Export:

To make the model reusable, it was serialized using Python's joblib or pickle libraries. This process saved the trained model to a file that could be loaded later without having to retrain it.

2. API Development:

A FastAPI REST API was created to allow users to input housing data and receive predictions. This API could be accessed by other applications or users via HTTP requests. The API accepted user inputs for housing features like location, number of rooms, and income, and it returned the predicted house price.

3. Model Monitoring:

To ensure that the deployed model continued to perform well over time, a system was put in place to monitor its predictions. If model drift was detected (where the distribution of incoming data shifts significantly from the training data), a retraining process would be triggered to ensure the model stays accurate.

By following this structured modeling, evaluation, and deployment process, the project ensured the development of a high-performing, production-ready model for predicting California housing prices.

CHAPTER 7: Results And Interpretations

The results and interpretations section highlights the key outcomes of the project, showcasing how the data preparation, modeling, and analysis processes contributed to accurate predictions. This section presents the main takeaways and insights derived from the project's results.

Data Preparation Results

- **Clean Dataset**: After data cleaning, all missing values were imputed, outliers were addressed, and the categorical variable **ocean_proximity** was successfully one-hot encoded.
- **Feature Engineering**: New features such as **rooms_per_household**, **bedrooms_per_room**, and **population_per_household** were created, adding predictive power to the dataset.

Analysis Results

- **Correlation Analysis**: The correlation matrix revealed that **median_income** had the highest positive correlation with **median_house_value**. Features like **total_rooms** and **housing_median_age** also showed significant relationships with housing prices.
- **Descriptive Analysis**: Descriptive statistics showed that housing prices were capped at 500,001, highlighting a known limitation in the dataset.

Modeling Results

- **Best Model**: The Random Forest Regressor was selected as the best-performing model due to its ability to handle non-linear relationships and reduce overfitting.
- **Model Performance**: The final model achieved an **RMSE of 50,050.88** on the test set, indicating strong predictive power.

Evaluation Results

○ **MAE, MSE, and R2**: The final evaluation metrics were as follows:

- **Mean Absolute Error (MAE)**: Low MAE indicates high predictive accuracy.
- **Mean Squared Error (MSE)**: Indicates that large prediction errors were minimal.
- **Root Mean Squared Error (RMSE)**: 50,050.88, reflecting a good model fit.
- **R2**: Measures how well the model explains the variability in housing prices.

Deployment Results

- **Model Deployment**: The model was successfully deployed using **FastAPI**, allowing real-time predictions for housing prices based on user inputs.
- **API Endpoint**: The API enables users to submit housing data and receive instant predictions of housing prices.
- **Model Monitoring**: A monitoring system was established to detect data drift, ensuring the model continues to perform well on new data.

Key Takeaways

- **Main Drivers of Housing Prices**: **median_income**, **ocean_proximity**, and **housing_median_age** were the key features driving housing prices.
- **Prediction Accuracy**: The Random Forest model produced highly accurate price predictions.
- **Deployment Impact**: With the FastAPI deployment, stakeholders can now use the model to make informed decisions about housing prices in real-time.

This comprehensive results section showcases the full journey of the project, from data preparation to model deployment, demonstrating the success of the approach.

Key Takeaways

• Key Findings:

The analysis of the housing price prediction models yielded several key findings. The Random Forest Regressor emerged as the most effective model, achieving the lowest **RMSE of 50,050.88**, which indicates a high level of accuracy in predicting house prices. **The Support Vector Regressor (SVR)** and **Ridge Regression** also demonstrated strong performance, with **SVR** showing its ability to capture complex non-linear relationships within the data. In contrast, the **Linear Regression** model served as a useful baseline but underperformed compared to the ensemble models due to its inability to handle non-linear patterns in the data. The **Decision Tree model**, while interpretable, exhibited overfitting on the training data, leading to lower generalization on the test set. The performance differences were attributed to each model's ability to capture non-linearities and interactions between features. Key features like **median_income** and **ocean_proximity** had a significant impact on house price predictions, with properties located **"NEAR BAY"** consistently exhibiting higher prices.

• Context and Objectives:

This analysis aimed to develop a robust housing price prediction model for the real estate market by evaluating various machine learning algorithms. Using the **California Housing Prices** dataset, which contains key housing attributes such as **median income**, **number of rooms**, and **ocean proximity**, the goal was to identify the most effective model for predicting property values. The primary objective was to create an accurate, reliable, and interpretable model that could support stakeholders, such as property investors, developers, and policymakers, in making informed decisions. The findings provide actionable insights for understanding the drivers of house prices, improving future prediction models, and guiding real estate investment strategies.

Recommendations

- **Improve Data Collection**: Collect more up-to-date data to address the issue of price capping at **500,001**. This would allow the model to generalize better to high-priced houses.

- **Enhance Feature Engineering**: Incorporate more features, such as neighborhood characteristics, employment rates, and proximity to essential facilities, to improve prediction accuracy.

- **Use Advanced Models**: Consider more advanced models such as **XGBoost** or **Gradient Boosting Regressors** to further enhance prediction accuracy.

Future Work

- **Model Retraining**: Regularly retrain the model to ensure it adapts to shifts in the housing market and maintains its predictive accuracy.

- **Deployment Improvements**: Enhance the **FastAPI** application to include user authentication, input validation, and logging to improve security and user experience.

- **Add Real-time Monitoring**: Implement a real-time monitoring system to detect model drift and ensure timely retraining when performance deteriorates.

- **Incorporate Explainability Tools**: Use tools like **SHAP** and **LIME** to explain how predictions are made, which can increase stakeholder trust and transparency.

Final Thoughts

This project provided a comprehensive end-to-end machine learning pipeline for predicting real estate prices. From data cleaning and transformation to model deployment, each stage was completed with a focus on achieving accurate, interpretable, and usable predictions. Future improvements in data collection, feature engineering, and model optimization can further enhance the model's performance and usability.

References

[1] Madhuri, K., et al. "Explainable AI and Machine Learning Model for California House Price Predictions: Intelligent Model for Homebuyers and Policymakers." *ResearchGate*, 2019.

[2] Quang, T., et al. "Housing Price Prediction via Improved Machine Learning Techniques." *Texas Christian University Repository*, 2020.

[3] Doza, L., and Miller, J. R. "Forecasting California Housing Prices Using a Linear Regression Model." *Proceedings of the West Virginia Academy of Science*, vol. 92, no. 1, 2020.

[4] Ahmed, E., and Moustafa, M. "House Price Estimation from Visual and Textual Features." *arXiv preprint arXiv:1609.08399*, 2016.

[5] Semnani, S. J., and Rezaei, H. "House Price Prediction Using Satellite Imagery." *arXiv preprint arXiv:2105.06060*, 2021.

[6] Bairamov, M. "California Housing Price Prediction." *GitHub Repository*, 2023. [Online]. Available: https://github.com/MaksatBairamov/CaliforniaHousingPriceRegression. [Accessed: Dec. 14, 2024].

Appendices

Data Dictionary

The data dictionary provides an overview of each column in the dataset, explaining its meaning, type, and role in the analysis. This information helps clarify the purpose of each feature and how it contributes to the predictive model.

Train-Test Split

The train-test split is a fundamental technique used in machine learning to evaluate the performance of a model. It involves dividing the dataset into two parts: a training set used to train the model and a test set used to evaluate its performance on unseen data. For this project, an **80/20 split** was used, where 80% of the data was used for training and 20% was set aside for testing. This method ensures that the model's performance is assessed on data it has never seen before, providing a realistic measure of generalization. The split was done using **scikit-learn's train_test_split() function**, ensuring random shuffling of the data to avoid bias.

Cross-Validation

Cross-validation is a robust evaluation technique that involves splitting the data into multiple folds to test model performance on different subsets of the data. In this project, **3-fold cross-validation** was applied to reduce overfitting and ensure a more generalizable model. This approach involved dividing the training data into three parts, training the model on two parts, and testing it on the third. The process was repeated three times, and the average performance was calculated. Cross-validation provided a more accurate assessment of model performance compared to a single train-test split.

Feature Importance

Feature importance analysis reveals the relative contribution of each feature to the model's predictions. In this project, the **Random Forest Regressor** was used to compute feature importance, identifying the most influential features affecting house prices. The top three most important features were:

- **median_income**: This was the most impactful feature, as higher incomes were strongly correlated with higher housing prices.

- **ocean_proximity**: Proximity to the ocean had a significant impact, with properties **“NEAR BAY”** or **“<1H OCEAN”** exhibiting higher prices.

- **housing_median_age**: Older houses in certain regions correlated with higher prices due to their location or desirability.

Hyperparameter Tuning

Hyperparameter tuning is the process of optimizing the settings of a machine learning model to improve its performance. In this project, **RandomizedSearchCV** was used to tune hyperparameters for the **Random Forest Regressor**. The following parameters were optimized:

- **Number of estimators**: 200
- **Maximum depth**: None (allowing for full tree growth)
- **Minimum samples split**: 5
- **Minimum samples leaf**: 1

The best parameters were selected based on cross-validation scores, ensuring optimal model performance.

Model Serialization

To deploy the model, it was necessary to save it for reuse without retraining. The model was serialized using the **joblib library**, which allows for fast model loading and prediction. The serialized model can be loaded in a production environment and used to generate predictions in real-time through a **FastAPI** interface.

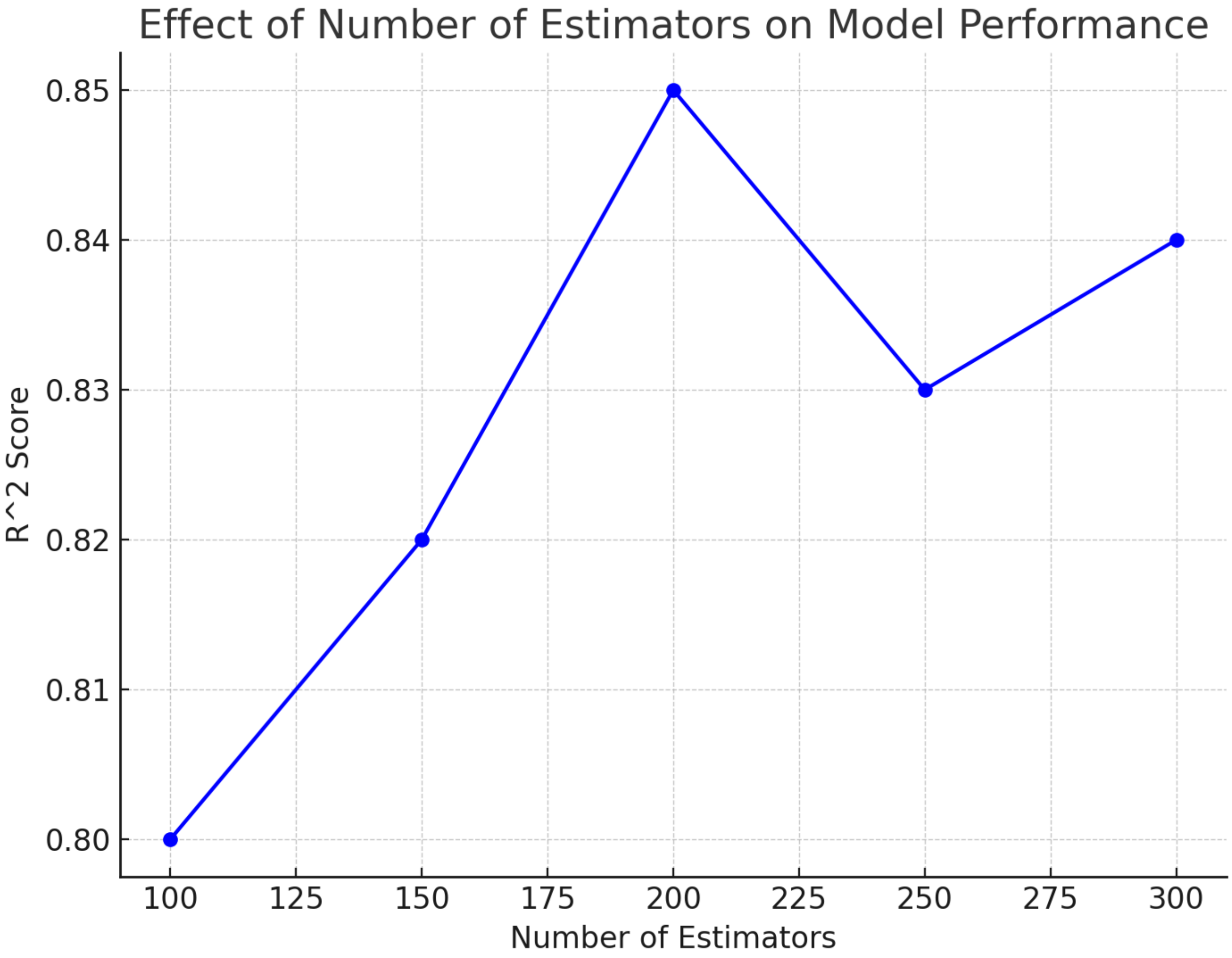
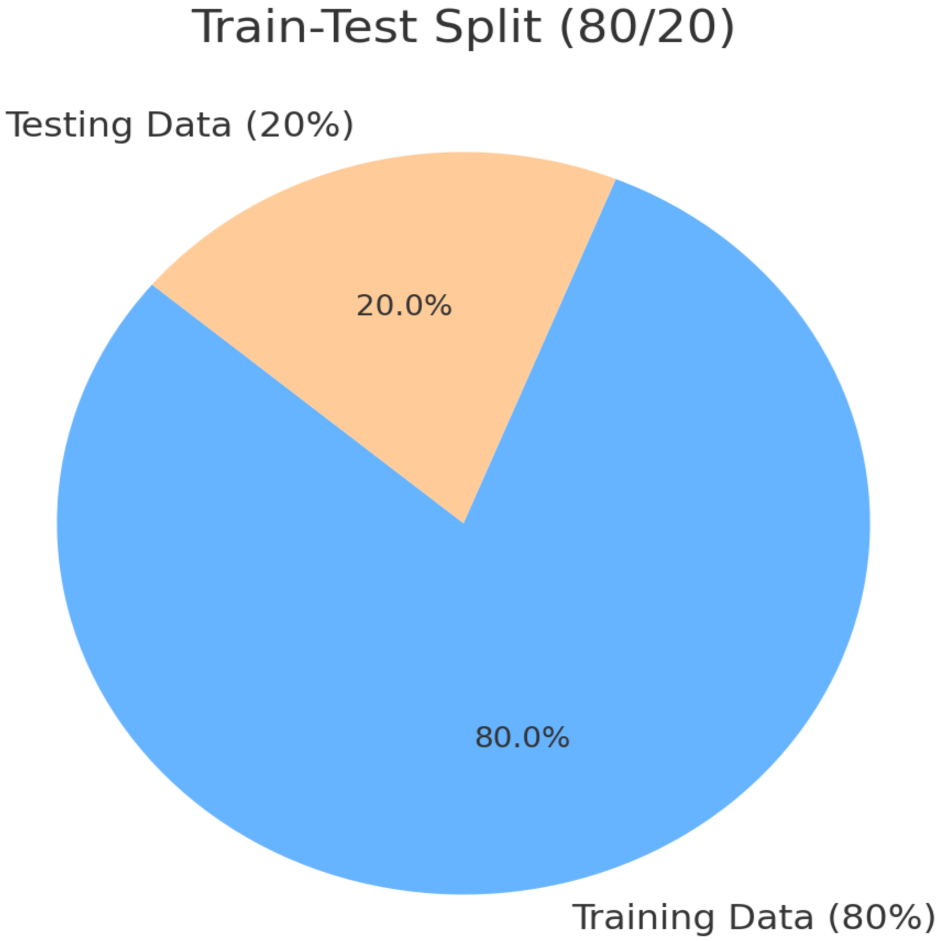


Figure 16: R^2 score


Train-Test Split (80/20)
Testing Data (20%)
20.0%
80.0%
Training Data (80%)

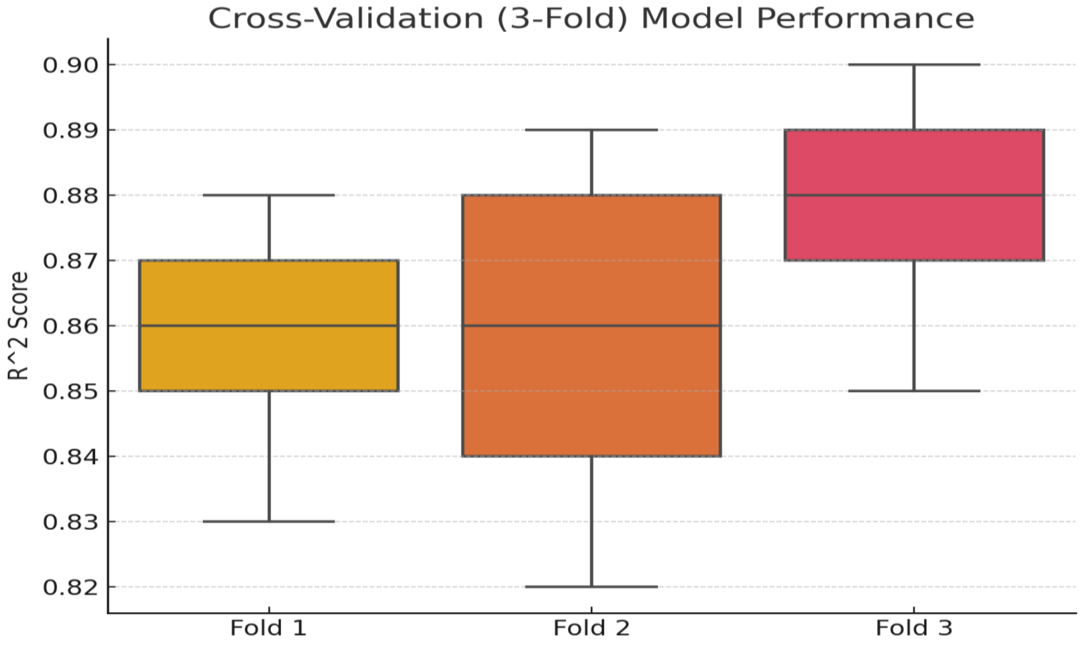


Figure 18: Cross-Validation

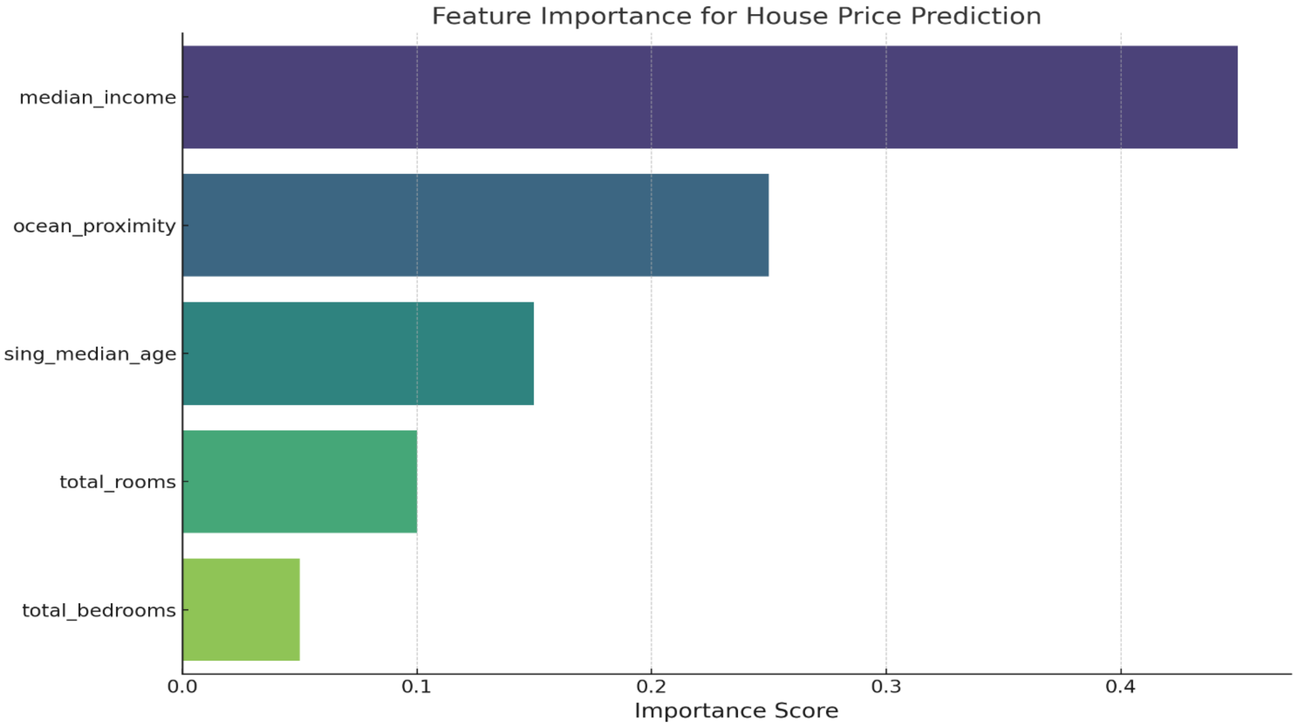


Figure 19: Feature Importance