VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**LÊ ĐÀO DUY TÂN - 52100104**

**VÕ ĐÌNH MINH TRÍ - 51900641**

**TRẦN QUANG LUÂN - 52100254**

**DATA ANALYSIS AND HOUSE PRICE FORECASTING**

**FINAL REPORT**

**BUSINESS INTELLIGENCE SYSTEMS**

**Ho Chi Minh City , 2023**

VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**LÊ ĐÀO DUY TÂN - 52100104**

**VÕ ĐÌNH MINH TRÍ - 51900641**

**TRẦN QUANG LUÂN - 52100254**

**DATA ANALYSIS AND HOUSE PRICE FORECASTING**

**FINAL REPORT**

**BUSINESS INTELLIGENCE SYSTEMS**

Instructor

**Ph.D Duong Huu Phuc**

**Ho Chi Minh City , 2023**

**EXPRESSING GRATITUDE**

First of all, we would like to sincerely thank Ph.D Duong Huu Phuc. During the process of studying business intelligence systems, the teacher dedicatedly guided and supported us to master the necessary issues in this subject. Above all, you have equipped us with enough knowledge to be able to complete this final report.

Next, we would like to send my sincere thanks to the Department of Information Technology at Ton Duc Thang University. The Faculty has created all conditions for us to study and research this subject. And especially the teachers in the department are always ready to share useful knowledge to help us complete our final report in the best possible way.

Finally, due to limited knowledge, we know that our midterm report has many shortcomings and limitations. We hope for your guidance and contributions to improve our final report. we are more perfect. Wishing all teachers good health.

*November 7, 2023, Ho Chi Minh City*

*Author*

*Le Dao Duy Tan*

*Vo Dinh Minh Tri*

*Tran Quang Luan*

**THE REPORT IS COMPLETED**

**AT TON DUC THANG UNIVERSITY**

I hereby declare that this is my own research project and is under the scientific guidance of Ph.D Duong Huu Phuc. The research content and results in this topic are honest and have not been published in any form before. The data in the tables for analysis, comments, and evaluation were collected by the author from different sources and clearly stated in the reference section.

In addition, the Project also uses a number of comments, assessments as well as data from other authors and other organizations, all with citations and source notes.

**If any fraud is detected, I will take full responsibility for the content of my Project**. Ton Duc Thang University is not involved in copyright violations caused by me during the implementation process (if any).

*November 7, 2023, Ho Chi Minh City*

*Author*

*Le Dao Duy Tan*

*Vo Dinh Minh Tri*

*Tran Quang Luan*

**DATA ANALYSIS AND HOUSE PRICE FORECASTINGABSTRACT**

Final project on the topic of data analysis and housing price prediction using machine learning models.

In the first chapter, the process of data collection is described, along with an exploration of outlier values to highlight any anomalies in the dataset. Subsequently, data normalization and handling of outlier values are performed. Following this, relationships between variables are identified to analyze their correlations, utilizing Exploratory Data Analysis (EDA) techniques.

Moving on to the second chapter, after establishing relationships between variables and gaining an understanding of the dataset, machine learning models are constructed to predict housing prices. Finally, an evaluation of the predictive models is conducted.

The third chapter encompasses the visualization of the models by constructing a website for inputting features and selecting a model to predict housing prices for new data.

**TABLE OF CONTENTS**

[ILLUSTRATION INVENTORY vi](#_Toc150511568)

[TABLES DIRECTORY vii](#_Toc150511569)

[ABBREVIATIONS CATALOG viii](#_Toc150511570)

[CHƯƠNG 1. INTRODUCTION AND TOPIC OVERVIEW 1](#_Toc150511571)

[1.1 Prọect specification 1](#_Toc150511572)

[1.2 Project Objectives 1](#_Toc150511573)

[1.3 Data specification 1](#_Toc150511574)

[1.4 Outlier values 2](#_Toc150511575)

[1.5 Data cleaning 4](#_Toc150511576)

[1.6 Relationship 5](#_Toc150511577)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 8](#_Toc150511578)

[2.1 Mạng neural hồi quy 8](#_Toc150511579)

[2.1.1 Recurrent Neural Network (RNN) 8](#_Toc150511580)

[2.1.2 Long Short-term Memory (LSTM) 8](#_Toc150511581)

[2.2 Mô hình Transformer 9](#_Toc150511582)

[2.2.1 Encoder và Decoder 9](#_Toc150511583)

[2.2.2 Attention 9](#_Toc150511584)

[CHƯƠNG 3. WEB-BASED VISUALIZATIO 10](#_Toc150511585)

[3.1 Overview 10](#_Toc150511586)

[3.2 Source Code Structure: 10](#_Toc150511587)

[3.2.1 App.py 10](#_Toc150511588)

[3.2.2 Index.html 11](#_Toc150511589)

[3.2.3 Running the Interface 12](#_Toc150511590)

[TÀI LIỆU THAM KHẢO 14](#_Toc150511591)

# ILLUSTRATION INVENTORY

[Figure 1.3.1: Data 2](#_Toc150511451)

[Figure 1.4.1: The chart illustrates data for the variable ‘price’ 3](#_Toc150511452)

[Figure 1.4.2: The chart illustrates data for the variable ‘room’ 3](#_Toc150511453)

[Figure 1.5.1: Read data 4](#_Toc150511454)

[Figure 1.5.2: Code to clean data 5](#_Toc150511455)

[Figure 1.6.1: Relationship between price and level 5](#_Toc150511456)

[Figure 1.6.2: Relationship between price and levels 6](#_Toc150511457)

[Figure 1.6.3: Relationship between price and area 6](#_Toc150511458)

[Figure 1.6.4: Relationship between price and kitchen area 7](#_Toc150511459)

[Figure 1.6.5: Relationship between price and geo lat 7](#_Toc150511460)

# TABLES DIRECTORY

# ABBREVIATIONS CATALOG

|  |  |
| --- | --- |
| EDA | Exploratory Data Analysis |
|  |  |
|  |  |
|  |  |
|  |  |

# INTRODUCTION AND TOPIC OVERVIEW

## Project specification

Nowadays, the real estate market is increasingly growing. The demand for stable living spaces for personal development, the need to purchase properties for business use, to open companies, offices, and to construct various projects are on the rise. Therefore, the requirement for real estate companies is to assess and determine the price at which houses should be purchased to align with the general market prices. Subsequently, they can strategize their business direction or property acquisitions to increase profitability for their company.

## Project Objectives

Collecting data about houses in order to analyze based on that data. Then, to develop a tool that can predict house prices based on input factors to determine the appropriate price for that house for purchase.

## Data specification

* Data source: Data was collected from the following source: https://www.kaggle.com/datasets/mrdaniilak/russia-real-estate-20182021/data
* The dataset used is the Russia Real Estate 2018-2021 dataset (Price Prediction) with over 5,477,007 real estate listings from Russia.
* The dataset consists of 13 features:
* date - date of publication of the announcement;
* time - the time when the ad was published;
* geo\_lat - Latitude
* geo\_lon - Longitude
* region - Region of Russia. There are 85 subjects in the country in total.
* building\_type - Facade type. 0 - Other. 1 - Panel. 2 - Monolithic. 3 - Brick. 4 - Blocky. 5 - Wooden
* object\_type - Apartment type. 1 - Secondary real estate market; 2 - New building;
* level - Apartment floor
* levels - Number of storeys
* rooms - the number of living rooms. If the value is "-1", then it means "studio apartment"
* area - the total area of the apartment
* kitchen\_area - Kitchen area
* price - Price. in rubles
* Data format: CSV file.
* This dataset may also contain errors and outliers that require further investigation.

A screen shot of a computer

Description automatically generated

Figure 1.3.1: Data

## Outlier values

A white background with black text

Description automatically generated

Figure 1.4.1: The chart illustrates data for the variable ‘price’

From the chart above, the negative values in the chart indicate that the house prices are unreasonable. This may be due to some data entry errors or legal issues in real estate transactions. The solution is to remove these negative values.

A cross with a line

Description automatically generated with medium confidence

Figure 1.4.2: The chart illustrates data for the variable ‘room’

From the chart above, there are certain times when real estate is sold. This indicates the need to exploit the selling times.

## Data cleaning

* First, read the dataset using the pandas library and convert it into a DataFrame

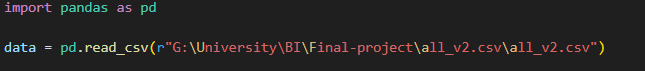


Figure 1.5.1: Read data

* Data.head() to view the top 5 rows of the data to understand the structure and data.isna().sum() to check for null values in the dataset. But after checking, there is no null data in the dataset, so there is no need to use a function to remove null data.
* From the available data, the "date" and "time" fields are currently in object format, so if they are passed into a sklearn model, it will not be processed correctly, causing an error. It converts the 'date' and 'time' columns in the DataFrame to the datetime data type. The 'date' column is converted using pd.to\_datetime, and the 'time' column is converted with a specific time format.
* Checking if there are any hidden missing values in the dataset. It does this by searching for values such as 'N/A,' 'NA,' 'NaN,' 'None,' 'Missing,' or an empty string ('') in the DataFrame and counts how many times these values appear in each column. After, to check for records where the 'price' column has values less than or equal to 0 and where the 'rooms' column has values less than -1. Then, it identifies the rows with negative prices and rooms and removes them from the DataFrame to retain only records with positive values.

A screenshot of a computer

Description automatically generated

Figure 1.5.2: Code to clean data

## Relationship

A blue and orange squares

Description automatically generated

Figure 1.6.1: Relationship between price and level

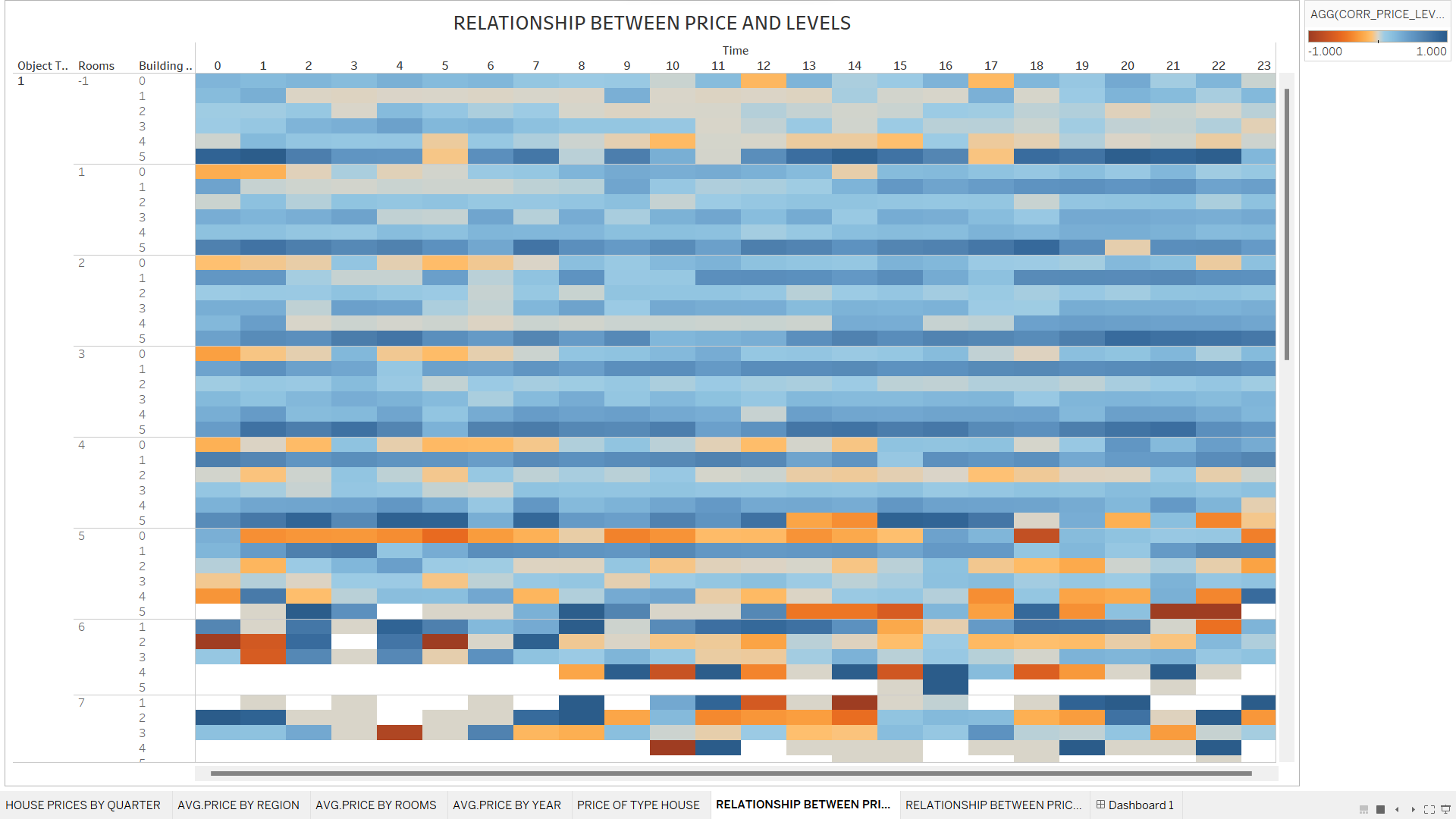


Figure 1.6.2: Relationship between price and levels

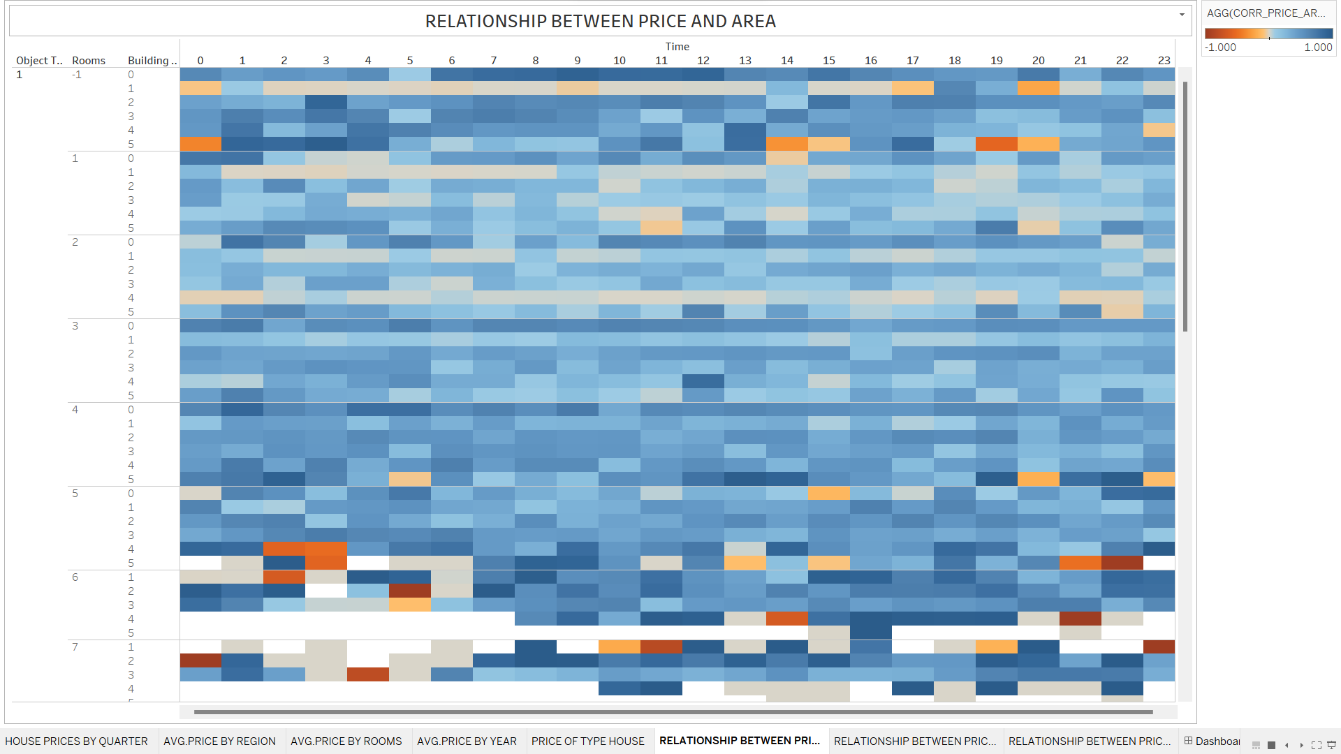


Figure 1.6.3: Relationship between price and area

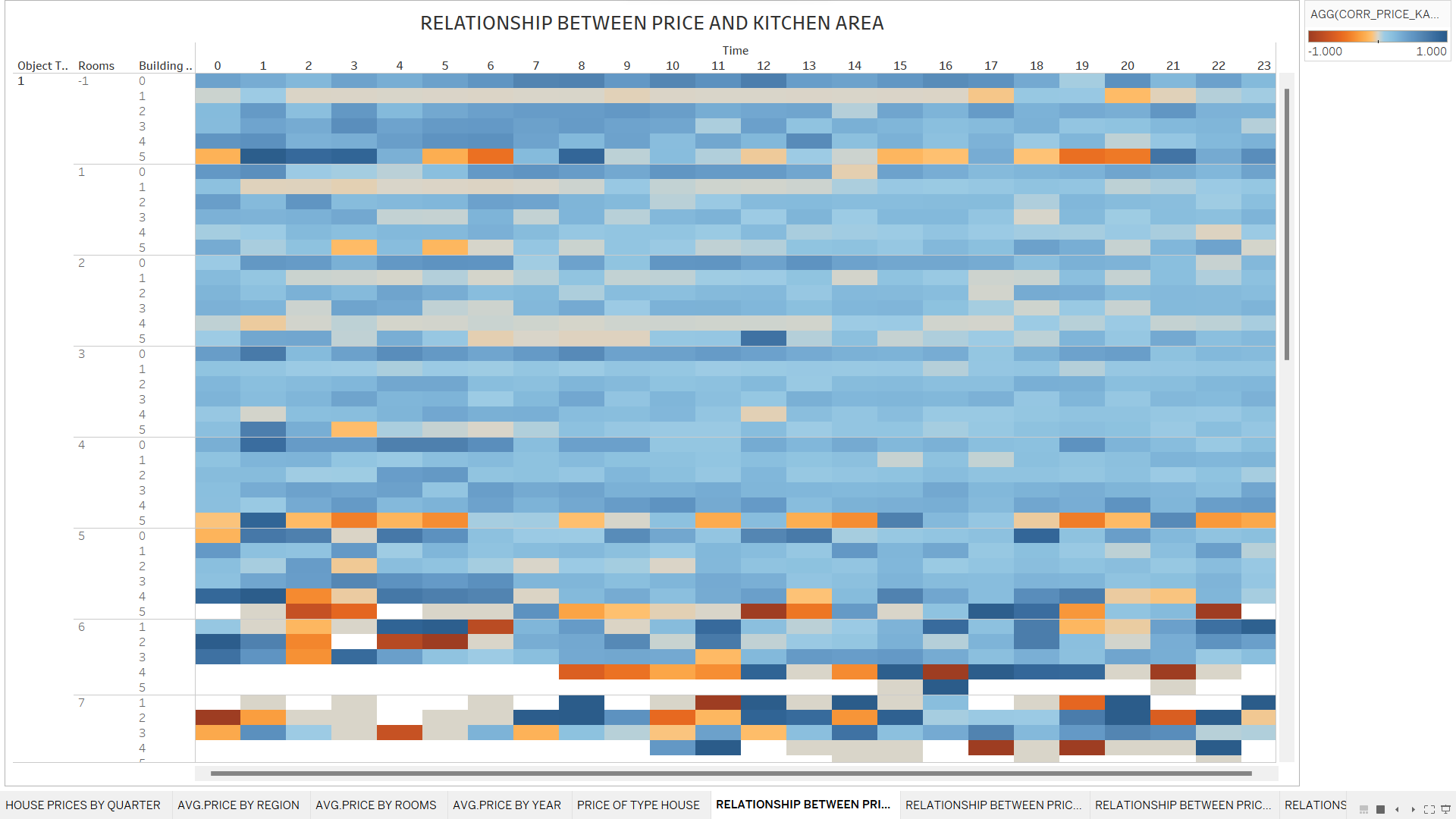


Figure 1.6.4: Relationship between price and kitchen area

A blue and orange squares

Description automatically generated

Figure 1.6.5: Relationship between price and geo lat

# MACHINE LEARNING MODELS

## Overview

### Linear Regression

Linear Regression is a model that assumes a linear relationship between features and house prices. It tries to find the best straight line (in the case of simple linear regression) or hyperplane (in the case of multiple linear regression) to fit the training data.

### Decision Tree

The Decision Tree model creates a tree of decisions based on if-else rules using features. Each node in the tree represents a decision or a feature attribute. Decision trees can be used for classification and predicting house prices based on different decisions made on various features.

### Random Forest

Random Forest is an ensemble learning model based on decision trees. It builds multiple independent decision trees and combines their predictions to make the final prediction. Random Forest is often good at handling nonlinear features and avoiding overfitting.

### Gradient Boosting Regression

Gradient Boosting Regression is a machine learning model belonging to the Gradient Boosting family. It is an ensemble learning model where weak decision trees are built sequentially and try to improve the prediction error step by step. Gradient Boosting Regression is commonly used for numeric value prediction (regression) in house price prediction tasks.

### Huber Regression

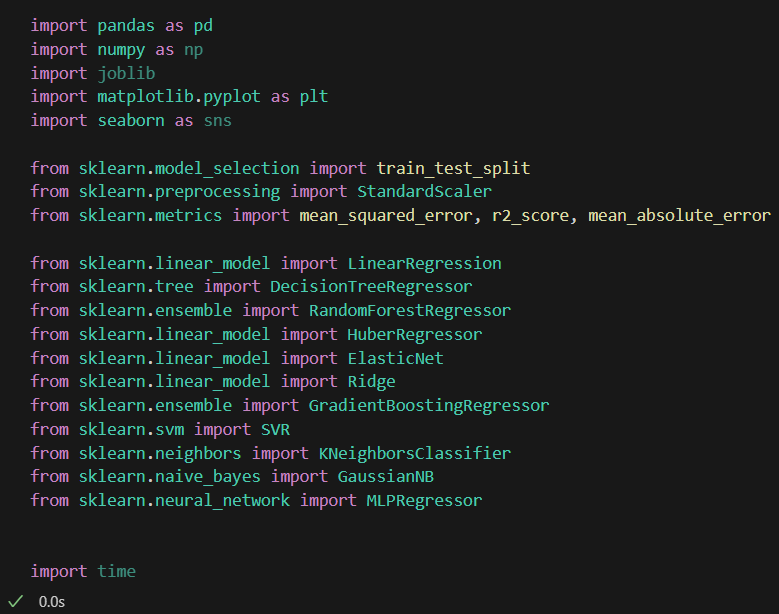
Huber Regression is a machine learning model belonging to the Ridge Regression family. It is used to predict numeric values, but it is more flexible than Ridge Regression as it can handle noise and outliers well. Huber Regression uses a Huber loss function to minimize the prediction error and reduce the impact of outlier data points.

### Elastic Net

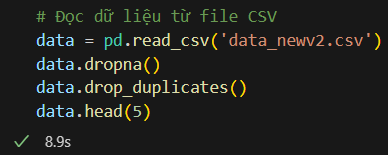
Elastic Net is a machine learning model that combines both Ridge Regression and Lasso Regression. It uses both regularization penalties to predict numeric values and performs feature selection. Elastic Net is often used in problems with a large number of features and some unimportant features.

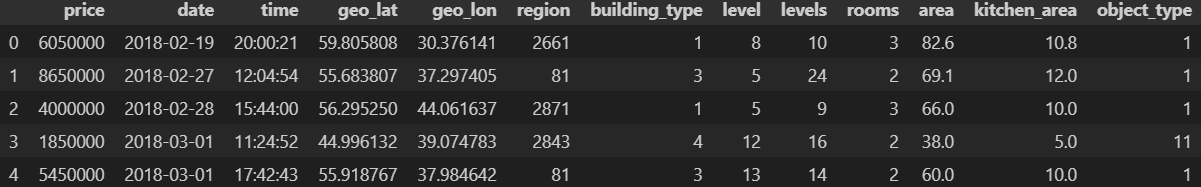
## Code Implementation

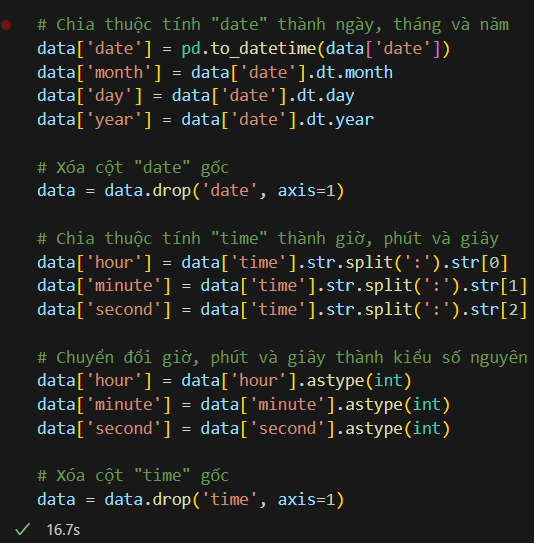
The code starts by importing the necessary libraries, including pandas, numpy, joblib, matplotlib, and seaborn. These libraries will be used for reading data, data processing, training and evaluating models.

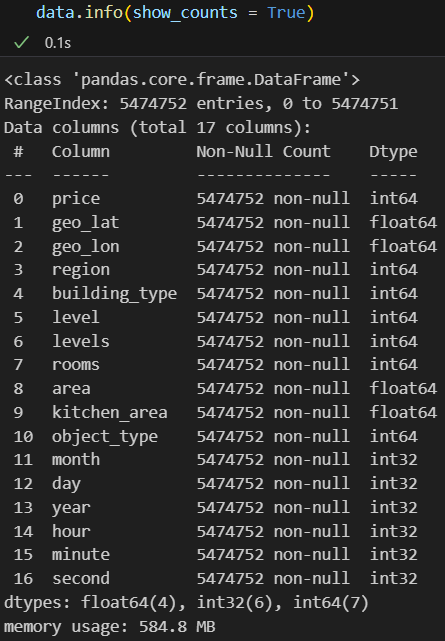


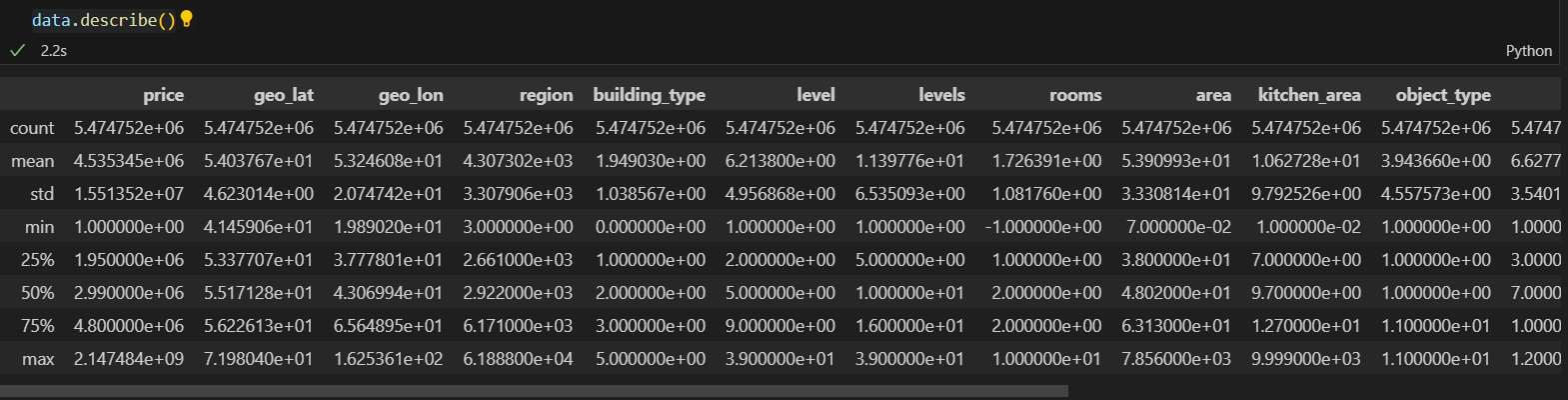
Next, the data is read from a CSV file using the pandas' read\_csv method and stored in the data variable. Some data processing is performed such as dropping rows with missing values (dropna), dropping duplicate rows (drop\_duplicates), splitting the "date" column into day, month, and year (pd.to\_datetime), splitting the "time" column into hour, minute, and second (str.split), converting data types, and dropping unnecessary columns.

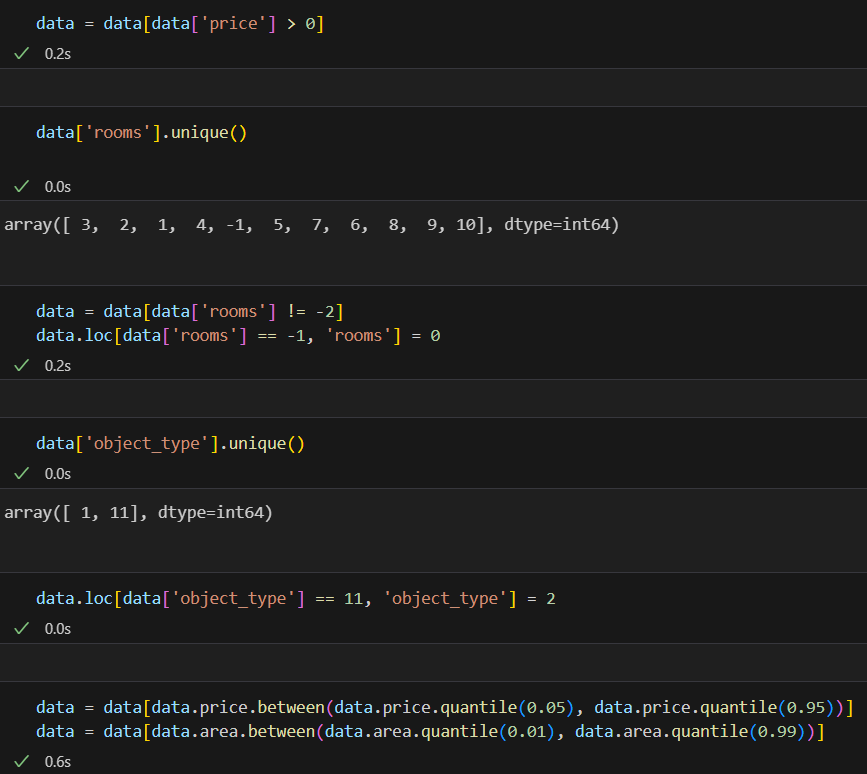




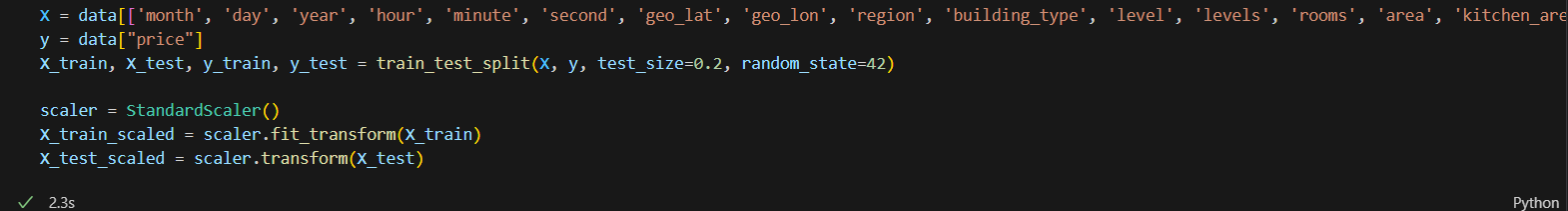




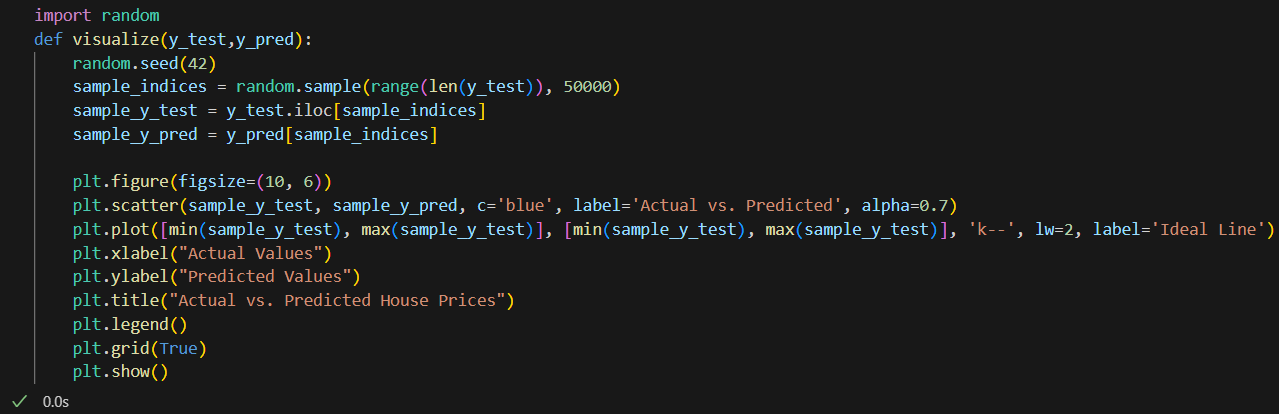




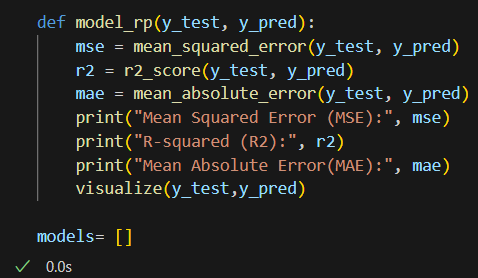
Then, the data is split into training and testing sets using train\_test\_split from sklearn. The feature attributes are selected and stored in the X variable, and the target attribute is stored in the y variable. The data is normalized using StandardScaler and split into normalized training and testing sets (fit\_transform and transform).



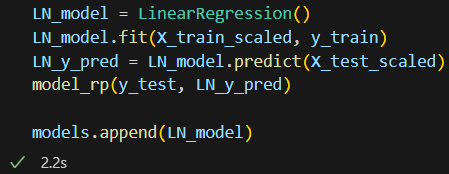
Next, a visualize function is defined to visualize the prediction results. This function uses the matplotlib library to plot a scatter plot between the actual values and the predicted values.

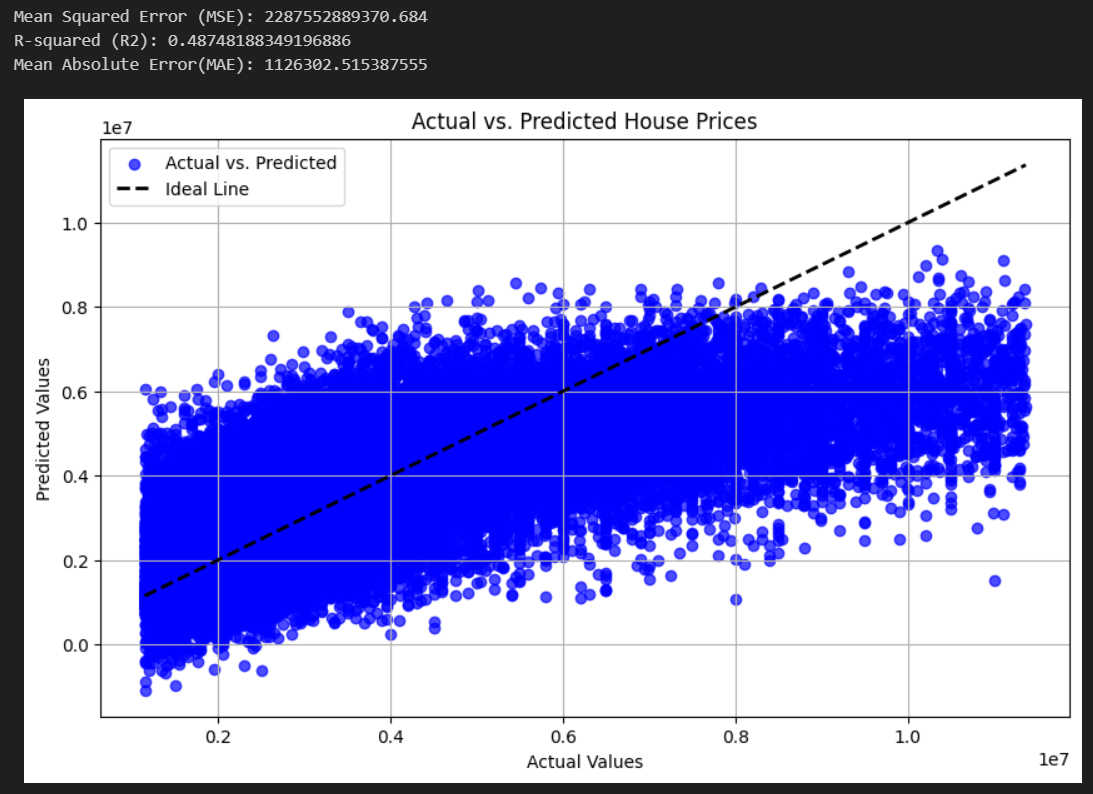


Next is the model\_rp function to evaluate the performance of the models. This function calculates evaluation metrics such as Mean Squared Error (MSE), R-squared (R2), and Mean Absolute Error (MAE), and then calls the visualize function to visualize the results.

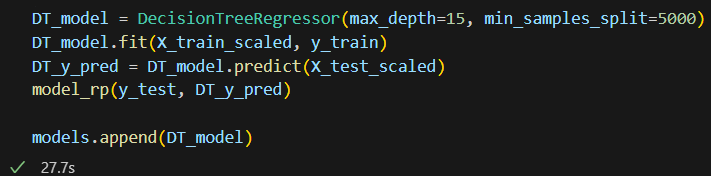


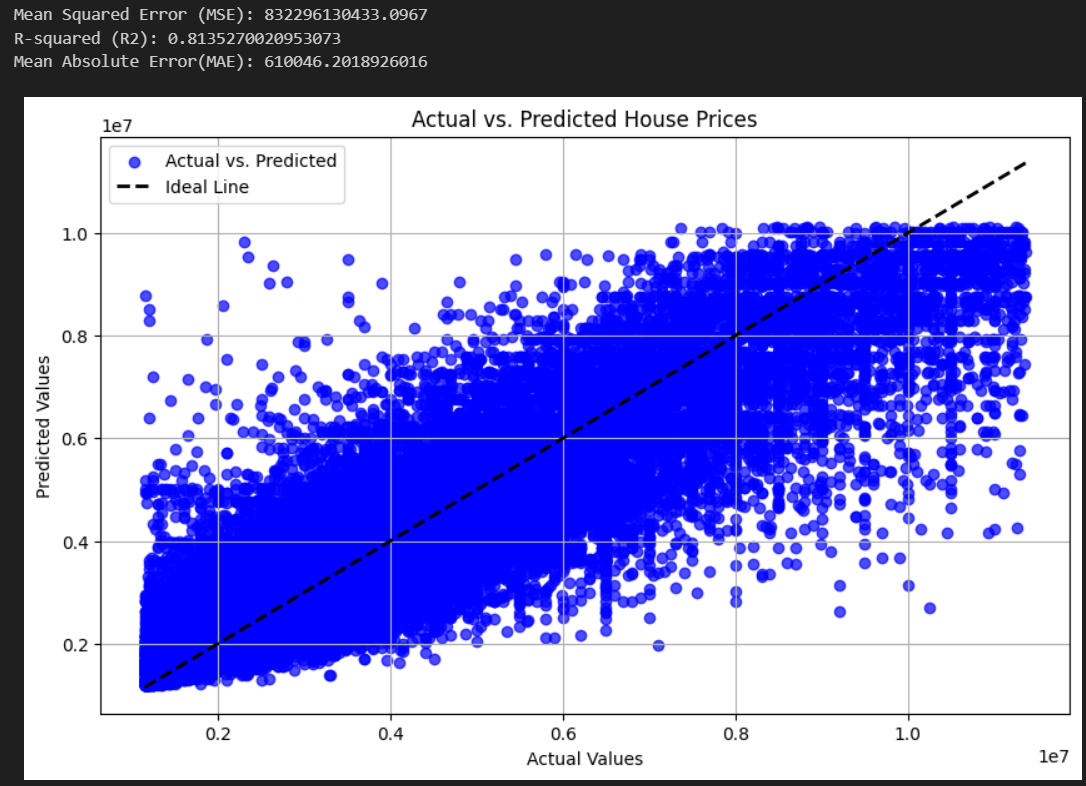
The Linear Regression model (LN\_model) is created by initializing an instance of LinearRegression(). The model is trained on the training data (X\_train\_scaled, y\_train) by calling the fit() method. Then, the model is used to predict house prices on the test set (X\_test\_scaled) by calling the predict() method. The predicted results are stored in the LN\_y\_pred variable. The function model\_rp(y\_test, LN\_y\_pred) is called to evaluate the performance of the model and visualize the results. The LN\_model is added to the list of models.



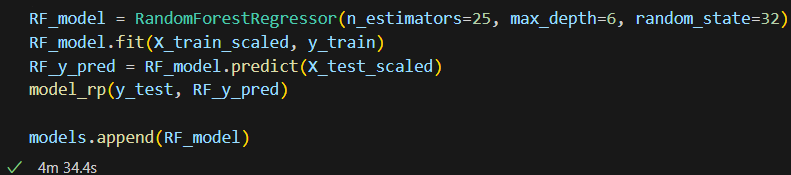


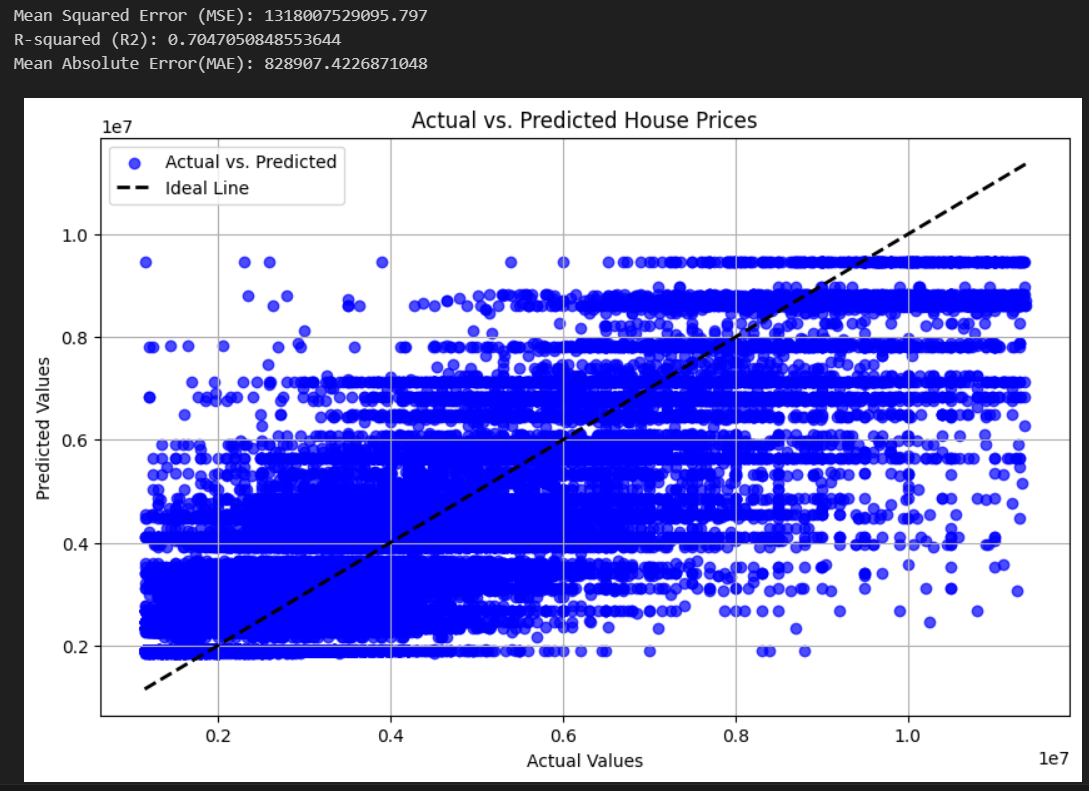
The Decision Tree model (DT\_model) is created by initializing an instance of DecisionTreeRegressor() with parameters max\_depth=15 and min\_samples\_split=5000. The model is trained and evaluated in a similar way to the Linear Regression model. The DT\_model and the prediction results (DT\_y\_pred) are added to the list of models.



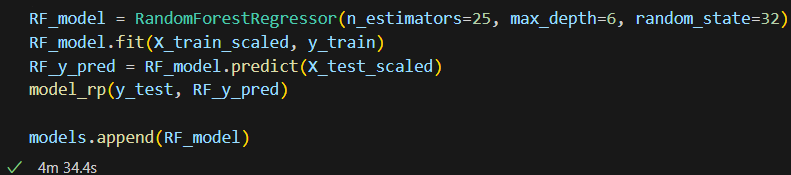


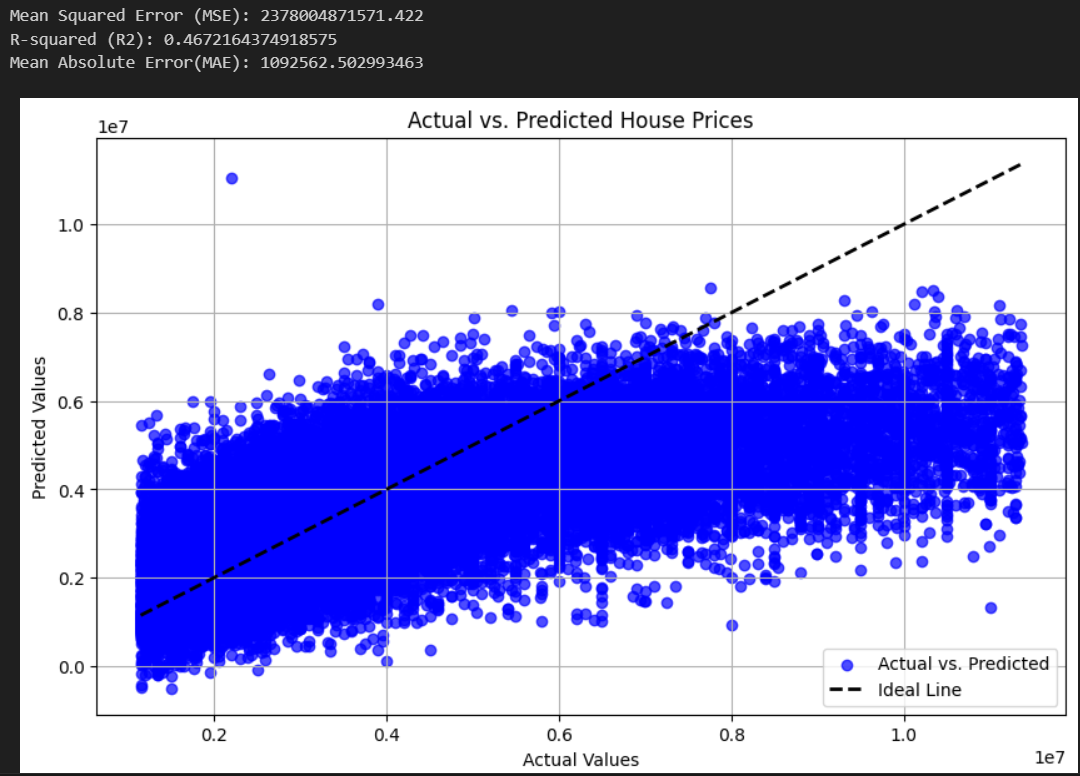
The Random Forest model (RF\_model) is created by initializing an instance of RandomForestRegressor() with parameters n\_estimators=25, max\_depth=6, and random\_state=32. The model is trained and evaluated in a similar way to the Linear Regression model. The RF\_model and the prediction results (RF\_y\_pred) are added to the list of models.



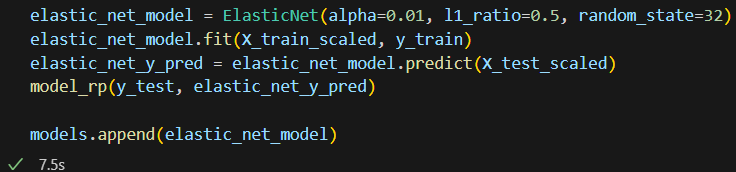


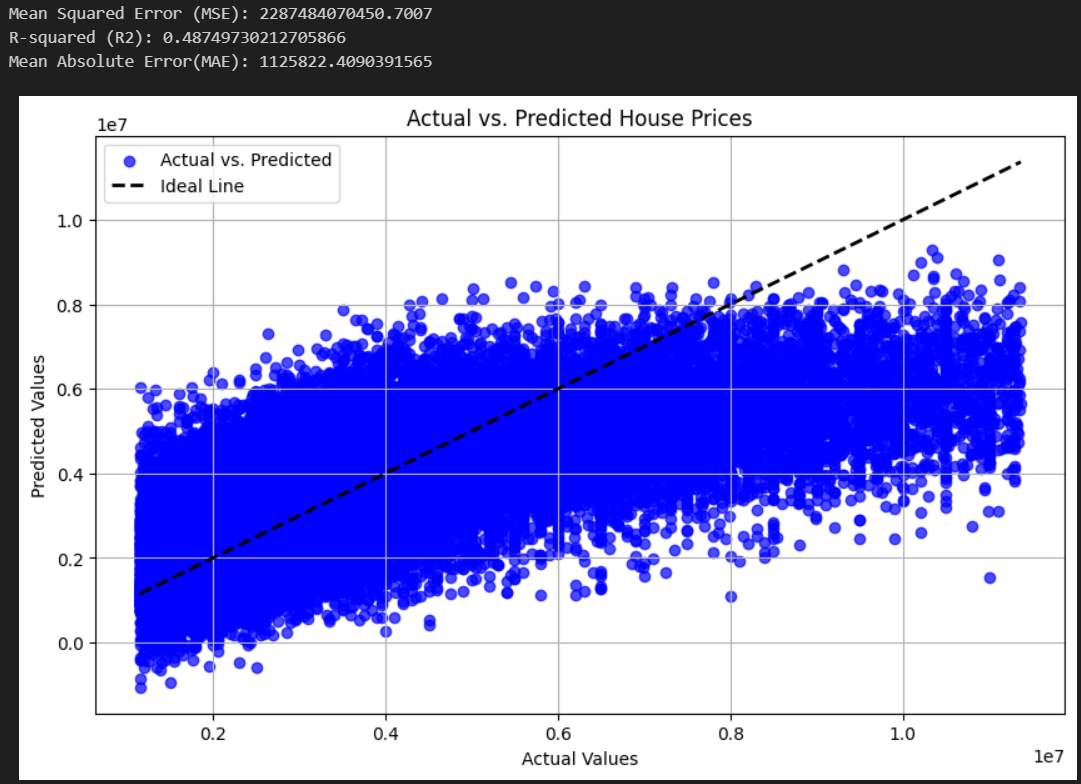
The HuberRegressor model (huber\_model) is created by initializing an instance of HuberRegressor() with parameters epsilon=1.35 and alpha=0.001. The model is trained and evaluated in a similar way to the Linear Regression model. The huber\_model and the prediction results (huber\_y\_pred) are added to the list of models.



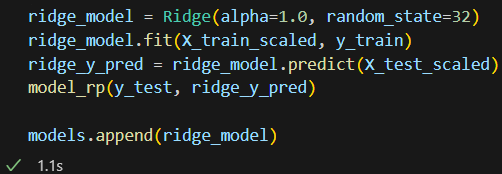


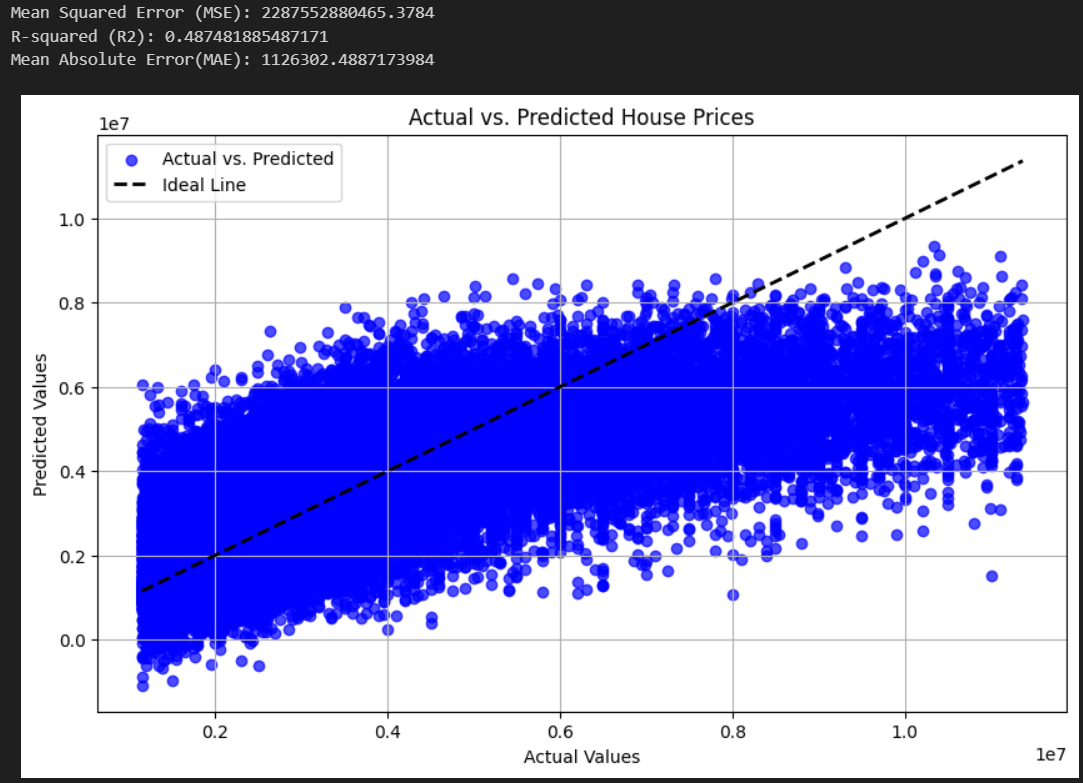
The ElasticNet model (elastic\_net\_model) is created by initializing an instance of ElasticNet() with parameters alpha=0.01 and l1\_ratio=0.5. The model is trained and evaluated in a similar way to the Linear Regression model. The elastic\_net\_model and the prediction results (elastic\_net\_y\_pred) are added to the list of models.



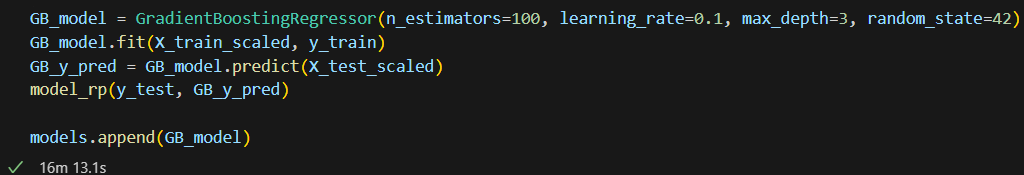


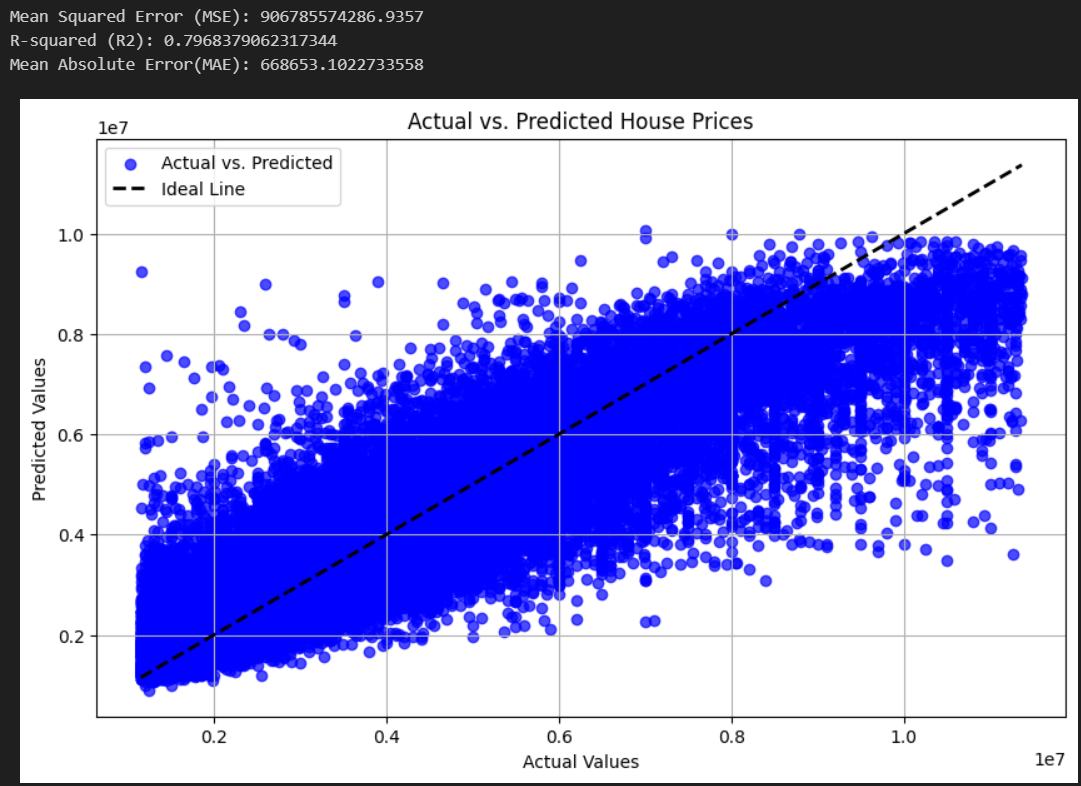
The Ridge model (ridge\_model) is created by initializing an instance of Ridge() with parameter alpha=1.0. The model is trained and evaluated in a similar way to the Linear Regression model. The ridge\_model and the prediction results (ridge\_y\_pred) are added to the list of models.



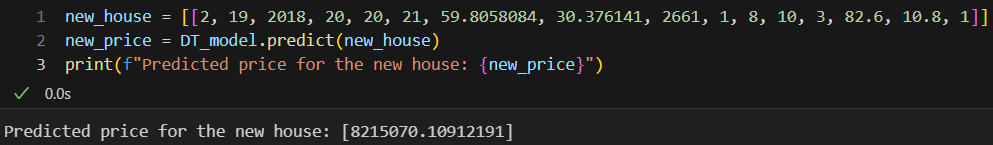


The Gradient Boosting model (GB\_model) is created by initializing an instance of GradientBoostingRegressor() with parameters n\_estimators=100, learning\_rate=0.1, max\_depth=3, and random\_state=42. The model is trained and evaluated in a similar way to the Linear Regression model. The GB\_model and the prediction results (GB\_y\_pred) are added to the list of models.





Finally, a new example of a new house is created, and the predicted house price for that new house is also printed to the screen.



## Analysis and Evaluation

Based on the provided evaluation metrics, we can analyze and evaluate the models for the house price prediction problem:

- Linear Regression Model:

+ Mean Squared Error (MSE): 2,287,552,889,370.684

+ R-squared (R2): 0.48748188349196886

+ Mean Absolute Error (MAE): 1,126,302.515387555

The linear regression model has a high MSE and MAE, indicating that it has a significant amount of prediction error. The R-squared value of 0.487 implies that the model explains only 48.7% of the variance in the target variable, which suggests limited predictive power.

- Decision Tree Regression Model:

+ Mean Squared Error (MSE): 832,296,130,433.0967

+ R-squared (R2): 0.8135270020953073

+ Mean Absolute Error (MAE): 610,046.2018926016

The decision tree regression model performs better than linear regression with a significantly lower MSE and MAE. The R-squared value of 0.814 indicates that the model explains approximately 81.4% of the variance in the target variable, suggesting a good predictive performance.

Random Forest Regression Model:

Mean Squared Error (MSE): 1,318,007,529,095.797

R-squared (R2): 0.7047050848553644

Mean Absolute Error (MAE): 828,907.4226871048

The random forest regression model also performs well with a lower MSE and MAE compared to linear regression. The R-squared value of 0.705 indicates that the model explains approximately 70.5% of the variance in the target variable, suggesting a decent predictive performance.

Huber Regression Model:

Mean Squared Error (MSE): 2,378,004,871,571.422

R-squared (R2): 0.4672164374918575

Mean Absolute Error (MAE): 1,092,562.502993463

The Huber regression model has a higher MSE and MAE compared to the linear regression model. The R-squared value of 0.467 indicates that the model explains approximately 46.7% of the variance in the target variable, which is lower than the linear regression model's performance.

Elastic Net Model:

Mean Squared Error (MSE): 2,287,484,070,450.7007

R-squared (R2): 0.48749730212705866

Mean Absolute Error (MAE): 1,125,822.4090391565

The elastic net model shows similar performance to the linear regression model, with comparable MSE, MAE, and R-squared values.

Ridge Model:

Mean Squared Error (MSE): 2,287,552,880,465.3784

R-squared (R2): 0.487481885487171

Mean Absolute Error (MAE): 1,126,302.4887173984

The ridge model performs similarly to the linear regression model, with similar MSE, MAE, and R-squared values.

Gradient Boosting Regression Model:

Mean Squared Error (MSE): 906,785,574,286.9357

R-squared (R2): 0.7968379062317344

Mean Absolute Error (MAE): 668,653.1022733558

The gradient boosting regression model performs well with a lower MSE and MAE compared to linear regression. The R-squared value of 0.797 indicates that the model explains approximately 79.7% of the variance in the target variable, suggesting a good predictive performance.

Overall, the decision tree regression, random forest regression, and gradient boosting regression models show better performance compared to linear regression and other models. These models have lower MSE and MAE values and higher R-squared values, indicating better predictive power and ability to explain the variance in the target variable.

# WEB-BASED VISUALIZATION

## Overview

We have built a simple web interface integrated with pre-trained machine learning models. The goal of the web is to provide users with a straightforward tool to predict house prices based on the data they upload, as well as input parameters entered by the user and predicted using the model selected by the user.

The web application is constructed using HTML, Python, and utilizes the Flask framework. Flask is a lightweight and powerful Python framework widely used for developing web applications.

## Source Code Structure:

### App.py

First, we need to import ‘**Flask’** and necessary libraries like ‘**joblib’** and ‘**numpy’** to build the web application.

Then, define a dictionary **'model\_paths'** to store the paths to the pkl files of 6 different models (e.g., decision\_tree\_model.pkl, ...). Its main purpose is to manage and organize the models for easy access.

A screen shot of a computer

Description automatically generated

Figure 3.2.1.1 Necessary Libraries and Dictionary

Continuing to define the application's homepage (index) with **GET** and **POST** methods. When a user makes a **POST** request (submits data), a machine learning model will be used based on the user's choice; the user can choose one of the available models. Then, it retrieves input data from the user interface and validates the data. If the data is valid, it uses the model selected by the user to predict house prices based on the input data. The prediction result is displayed on the web page and returned to the user.

A screenshot of a computer screen

Description automatically generated

Figure 3.2.1.2 Necessary Libraries and Dictionary

### Index.html

The website is a simple form that allows users to select a decision tree model and input data. When the user clicks the 'Predict' button, the application processes the data and displays the prediction results on the website.

On the web interface, users will first select a machine learning model (one of six available models) and upload a file. Then, users proceed to enter input data and click the 'Predict' button. After clicking the button, the system will predict the house price based on the user's initially selected machine learning model. If the user leaves any input fields empty, the system will prompt them to enter complete information. If all the required conditions are met, the predicted house price will be displayed on the screen.

### Running the Interface

In order to run the interface, first open the command prompt or PowerShell, navigate to the folder containing the code, and run the command **'python app.py'** After successful execution, the command prompt will display the following content:

A screenshot of a computer

Description automatically generated

Figure 3.2.3.1 The cmd screen when running app.py

At this step, you will see the content **'Running on http://127.0.0.1:5000**', which means the website is running on port 5000. Copy **'http://127.0.0.1:5000'** into any web browser, paste the copied URL, and press Enter. If successful, the website will appear as shown in the image below:

A screenshot of a computer

Description automatically generated

Figure 3.2.3.2 The web after a successful run" "The web after a successful run

Then, to predict house prices, proceed to select a model, upload a file, enter input data, and click the 'Predict' button. If the model is error-free and the input data is valid, the house price will be predicted and displayed on the screen.

A screenshot of a computer

Description automatically generated

Figure 3.2.3.2 The predicted house price is displayed on the screen.

So, we now have a simple web interface that allows users to predict house prices based on input data using a selected model.

# TÀI LIỆU THAM KHẢO

Tiếng Việt

…

Tiếng Anh

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-term Memory. *Neural Computation*, *9*, 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). *Attention Is All You Need* (arXiv:1706.03762). arXiv. https://doi.org/10.48550/arXiv.1706.03762