PREDICTING HOUSE PRICE USING MACHINE LEARNING

NAME: YOGESWARI E **REG NO:420721104059**

PHASE 5 DOCUMENT SUBMISSION

ABSTRACT:

The "House Prices Prediction using Machine Learning" project aims to develop a robust and accurate predictive model for real estate pricing. With the ever-changing real estate market, accurate price predictions are crucial for both buyers and sellers. In this project, we employ various machine learning algorithms and regression techniques to analyze a diverse set of features, including property size, location, amenities, and historical sales data. By leveraging a dataset of historical house prices and associated attributes, our model is trained to make predictions on the prices of unseen properties. We evaluate random forest algorithm to determine the most effective approach. The project emphasizes data preprocessing, feature engineering, and model evaluation to ensure reliable predictions. We also investigate the impact of various factors on property prices, which can provide valuable insights for real estate stakeholders. Ultimately, the "House Prices Prediction using Machine Learning" project not only offers a practical tool for estimating house prices but also contributes to a better understanding of the key determinants affecting real estate values.

Keywords: House price prediction, machine learning, Random Forest Regressor, data analysis, feature engineering, model training, real estate market, USA Housing dataset, Kaggle

INTRODUCTION:

House price prediction using machine learning is an innovative and data-driven approach to estimate the value of residential properties. In a world where real estate is a significant investment for individuals and a crucial market for various stakeholders, accurate house price prediction is essential for making informed decisions. Machine learning techniques offer a powerful means to analyze vast amounts of data and generate reliable forecasts.

This predictive model leverages historical and current property data, such as location, size, number of bedrooms, bathrooms, and various other features, to predict the market price of a house. Machine learning algorithms, including regression, decision trees, random forests, and neural networks, are applied to this

data to learn patterns and relationships, enabling the model to provide estimates that are more precise and data-driven than traditional methods.

House price prediction using machine learning can benefit a variety of stakeholders, including homebuyers looking to make informed purchasing decisions, real estate agents seeking to provide accurate pricing guidance to clients, and property investors aiming to optimize their portfolios. It also aids banks and financial institutions in assessing property valuations for mortgage approvals and insurance companies in pricing homeowner's insurance.

In this process, data collection, preprocessing, feature engineering, model training, and evaluation are essential steps. It's imperative to ensure that the model is trained on diverse and representative data to make predictions that generalize well to unseen properties.

Machine learning-based house price prediction not only improves the accuracy of valuations but also offers a more transparent and data-driven approach to real estate transactions, ultimately empowering individuals and businesses to make more informed decisions in the dynamic and competitive housing market.

OVERVIEW OF HOUSE PRICE PREDICTION

Predicting house prices using machine learning is a common and valuable application in the field of real estate and finance. To implement a house price prediction model using machine learning algorithms like random forest. These are the steps that we developed for the implementation of our project.

1. **Empathize:**

   Recognize the requirements and difficulties faced by each party involved in the process of predicting the price of a home, including investors, real estate agents, appraisers, sellers, and purchasers. To obtain information on what user’s value in property valuation and what information is most important for their decision-making, conduct surveys and interviews.

2. **Define:**

   Clearly express the challenge, E.g., "How might we use machine learning to predict house prices more transparently and accurately?". Determine the primary objectives and success standards for the project, such as raising user confidence in the valuation process, decreasing bias, or enhancing prediction accuracy.

3.Ideate:

Come up with innovative ideas and sources of information that can improve the precision and openness of house price forecasts. To produce a wide range of ideas, such as the use of alternative data, new algorithms, or enhanced visualization techniques, promote interdisciplinary collaboration.

4.Prototype:

Using the concepts from the ideation stage, develop machine learning models in prototype form. Evaluate and refine these prototypes to identify the most accurate and user-friendly techniques.

5.Test:

Using actual data and scenarios, test the machine learning models to get input from users and stakeholders. Evaluate the models' compliance with the established objectives and success criteria and modify them in response to user input.

6. Implement:

Using the top-performing algorithms and data sources, create a machine learning solution that is ready for production to predict house values. Use model interpretability tools and other transparency techniques to make sure people know how predictions are made.

7. Evaluate:

Obtain user comments and insights to pinpoint areas in need of improvement. Continually track the machine learning model's performance following implementation to make sure it stays accurate and relevant in a changing real estate market.

8. Iterate:

Using an iterative process, improve the machine learning model in response to continuous feedback and evolving user requirements. Always look for methods to improve customer satisfaction, transparency, and prediction accuracy.

9. Scale and Deploy:

After the machine learning model has been refined and proven, broaden its user base by implementing it at a large scale, including real estate

agents, investors, and homeowners. Make sure the model can be accessed with ease using intuitive interfaces and that it fits in well with real estate workflows.

10. Educate and Train:

Offer instructional materials and training to users so they may comprehend the constraints, workings, and considerations taken into account by the machine learning model. Encourage stakeholders to adopt a data-literate culture to increase technology confidence.

DATA COLLECTION:

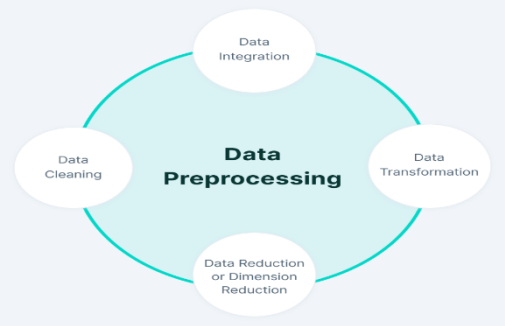
Compile a thorough dataset with pertinent factors including location, size, age, facilities, and crime rates in the area around schools.

# Avg. Area I...	# Avg. Area ...	# Avg. Area ...	# Avg. Area ...	# Area Popul...	# Price	A Address
79545.45857431678	5.682861321615587	7.009188142792237	4.09	23086.80050268645 6	1059033.5578701235	208 Michael Ferry Apt. 674 Laurabury, NE 37010-5101
79248.64245482568	6.002899808275242 5	6.730821019094919	3.09	40173.07217364482	1505890.91484695	188 Johnson Views Suite 079 Lake Kathleen, CA 48958
61287.067178656784	5.865889840310001	8.512727430375099	5.13	36882.15939970458	1058987.9878760849	9127 Elizabeth Stravenue Danieltown, WI 06482-3489
63345.24004622798	7.188236094518642 5	5.586728664827653	3.26	34310.24283090706	1260616.8066294468	USS Barnett FPO AP 44820
59982.19722570803 4	5.040554523106283	7.839387785120487	4.23	26354.10947210314 8	630943.4893385402	USNS Raymond FPO AE 09386
80175.7541594853	4.988407757533714 5	6.104512439428879	4.04	26748.42842468971 5	1068138.0743935304	06039 Jennifer Islands Apt. 443 Tracyport, KS 16077
64698.46342788773	6.025335906887153	8.147759585023431	3.41	60828.24908540716	1502055.8173744078	4759 Daniel Shoals Suite 442 Nguyenburgh, CO 20247

Data set link: https://www.kaggle.com/datasets/vedavyasv/usa-housing

DATA PREPROCESSING:

The obtained data undergoes numerous cleaning and transformation operations in the preprocessing stage to verify its quality and usefulness for analysis. This calls for addressing missing values, eliminating outliers, normalizing or scaling the data, and encoding categorical variables, among other things. Preprocessing assists in getting the data ready for efficient modelling.



MODEL TRAINING:

Model training is the process of teaching a machine learning model to predict house prices. It involves feeding the model historical data on house prices and features, such as square footage, number of bedrooms, and location. The model then learns the relationships between these features and house prices. Once the model is trained, it can be used to predict house prices for new data. For example, you could use the model to predict the price of a house that you are interested in buying.

Training:

To assess the model's performance, divide the dataset into training and testing sets. To avoid overfitting, take into consideration methods like as cross-validation.

Hyperparameter Tuning:

To increase the model's forecast accuracy, fine-tune its hyperparameters. This is where strategies like random or grid search come in handy.

Evaluation Metrics:

Choose the proper metrics, such as Mean Absolute Error (MAE), Mean Squared Error (MSE), or Root Mean Squared Error (RMSE), for regression jobs. Select the metric that best fits the project's particular goals.

Regularization:

To avoid overfitting, use regularization strategies like L1 (Lasso) or L2 (Ridge) regularization.

Feature Selection:

To determine which features are most pertinent to the prediction, apply strategies such feature importance scores or recursive feature elimination.

Interpretability:

Verify the explanation and interpretation of the model's predictions. This is particularly crucial for real estate applications, as stakeholders need to comprehend the variables influencing forecasts.

Deployment:

Provide an intuitive user interface or API so that end users may enter information about their properties and obtain price estimates.

EXPLORATORY DATA ANALYSIS:

Exploratory Data Analysis (EDA) for house price prediction is crucial in understanding and preparing the dataset for machine learning. It involves loading and inspecting the data, calculating summary statistics, and creating various visualizations, such as histograms, box plots, scatter plots, and heatmaps to uncover insights. EDA helps identify outliers, missing data, and correlations between features and the target variable. For categorical features, bar plots reveal category distributions. EDA also informs decisions on data preprocessing, feature engineering, and feature selection, improving the model's predictive accuracy. It's an iterative process, and the insights gained guide data transformation and model development. Proper EDA ensures that your machine learning model is built on a solid foundation of understanding the data's nuances, ultimately leading to more accurate house price predictions.

FEATURE SELECTION

To determine which features are most crucial for prediction, use feature significance scores from your model or methods like recursive feature removal.

Interpretability:

To understand and comprehend the model's predictions. It may be of interest to stakeholders to know how each feature affects the estimated price of a home.

Deployment:

Deploy your learned model in an actual environment using a web application, an API, or any other kind of user-friendly interface. When users provide property data, the model forecasts prices.

Monitoring and Maintenance:

Monitor the model's performance and make updates as appropriate. Since real estate markets fluctuate, it's critical to regularly retrain the model with fresh data.

Ethical Considerations:

Ensure that your model doesn't introduce or perpetuate biases in pricing. Implement fairness and transparency measures.

Innovation:

Explore innovative approaches such as incorporating external data sources (e.g., satellite imagery, IoT data) for better predictions.

FEATURE ENGINEERING:

Feature engineering is a critical aspect of building a successful machine learning model for house price prediction. Effective feature engineering can help to create new, informative features and transform existing ones to improve the model's predictive accuracy. Here are some feature engineering techniques to apply.

Age of the House:

Convert the "Age of the House" into a more relevant feature by subtracting it from the current year to calculate the number of years since construction. Create categorical bins or ranges for house age, such as "new," "recent," "mid-aged," and "old."

Square Footage per Bedroom:

Calculate the "Square Footage per Bedroom" by dividing the total square footage by the number of bedrooms. This can help understand the spaciousness of the bedrooms. Create bins or ranges for this value, which can provide insights into the size of bedrooms in relation to the overall property.

Bathrooms per Square Foot:

Calculate the "Bathrooms per Square Foot" by dividing the number of bathrooms by the total square footage. This can give you an indication of the density of bathrooms in the property. This feature may help identify properties with a higher or lower bathroom-to-living space ratio.

School District Quality:

If the "School District Quality" is a categorical variable, encode it into numerical values using techniques like target encoding or ordinal encoding. Consider aggregating school district quality information at different levels, such as neighbourhood or city, and using these aggregated values as features.

SPLITTING THE DATASET:

Split the dataset into training and testing sets using functions like train_test_split from the scikit-learn library. We will divide the datasets into train and test split with 80% of the data for model building and 20% for testing the model

Program:

```
X = df.drop('price', axis=1) # Features

y = df['price'] # Target variable

X_train, X_test, y_train, y_test = train_test_split(X, y,
```

test_size=0.2, random_state=42)

We utilized random sampling to create training and testing datasets. Ordinarily, a neighbourhood's median income acts as a strong gauge of the wealth distribution in the locality. Hence, our goal is to guarantee that the test dataset effectively represents the various income categories. This necessitates the conversion of data into categorical variables and the implementation of stratified sampling in place of random sampling.

MODEL SELECTION:

A evaluation of several machine learning algorithms is required throughout the model selection phase to determine which is best for precise house price prediction. Every model underwent a thorough evaluation according to its functionality and fit for the job. Upon careful examination, several types of models are present: Gradient Boosting Regressors are used to create robust predictive models; Random Forest Regression is preferred for its skill in handling complex interactions; Support Vector Regression (SVR) is known for its ability to handle non-linear patterns; and Linear Regression is known for its simplicity in capturing linear relationships. The model's ability to accurately estimate home values and capture complex data correlations was what ultimately led to its selection.

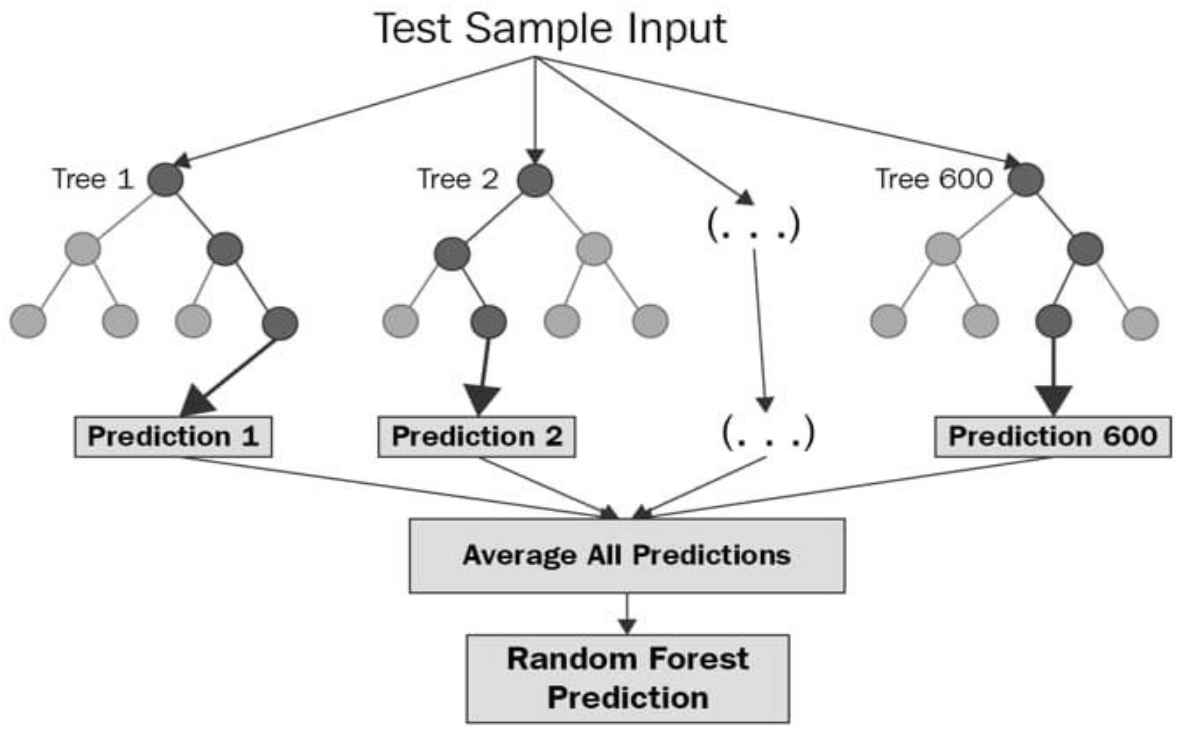
We have selected the random forest regressor from the model.

Random Forest:

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.



MODEL TRAINING:

Model training is the process of teaching a machine learning model to predict house prices. It involves feeding the model historical data on house prices and features, such as square footage, number of bedrooms, and location. The model then learns the relationships between these features and house prices. Once the model is trained, it can be used to predict house prices for new data. For example, you could use the model to predict the price of a house that you are interested in buying.

Training:

To assess the model's performance, divide the dataset into training and testing sets. To avoid overfitting, take into consideration methods like cross-validation.

Hyperparameter Tuning:

To increase the model's forecast accuracy, fine-tune its hyperparameters. This is where strategies like random or grid search come in handy.

Evaluation Metrics:

Choose the proper metrics, such as Mean Absolute Error (MAE), Mean Squared Error (MSE), or Root Mean Squared Error (RMSE), for regression jobs. Select the metric that best fits the project's particular goals.

Regularization:

To avoid overfitting, use regularization strategies like L1 (Lasso) or L2 (Ridge) regularization.

Feature Selection:

To determine which features are most pertinent to the prediction, apply strategies such feature importance scores or recursive feature elimination.

Interpretability:

Verify the explanation and interpretation of the model's predictions. This is particularly crucial for real estate applications, as stakeholders need to comprehend the variables influencing forecasts.

Deployment:

Provide an intuitive user interface or API so that end users may enter information about their properties and obtain price estimates.

Continuous Improvement:

Establish a feedback loop to continuously refine the model in response to user input and fresh data.

Ethical Aspects:

Recognize the possibility of biases in the model and data. Make sure your forecasts are transparent and equitable.

Maintenance and Monitoring:

Keep an eye on the model's real-world performance on a regular basis and update it as necessary.

Innovation:

Consider cutting-edge methods like incorporating natural language processing into textual property descriptions or employing satellite images or Internet of Things data for real-time property condition monitoring.

MODEL EVALUATION:

Model evaluation is a critical step in building a house price prediction model using machine learning. It helps you assess the model's performance and determine how well it can predict house prices.

```
# Evaluate the models on the test set
X_test_selected = selector.transform(X_test)
# Make predictions
random_forest_predictions = random_forest_model.predict(X_test_selected)
# Evaluate model performance
def evaluate_model(predictions, model_name):
 mse = mean_squared_error(y_test, predictions)
 r2 = r2_score(y_test, predictions)
 print(f"{model_name} Model Evaluation:")
 print(f"Mean Squared Error (MSE): {mse}")
print(f"R-squared (R2) Score: {r2}\n")
```

Output:

MSE :10141766848.330585

RMSE:100706.33966305491

HOUSE PRICE INFLUENCING FACTORS:

several factors can influence the pricing of houses, including:

Location: The proximity of amenities, schools, transit, and safe neighborhoods has a big influence on home values.

Market conditions: Interest rates, economic trends, and supply and demand dynamics all have an impact on the total housing market and its values.

Property size and condition: The property's age, size, layout, condition, and any repairs or renovations can all affect how much it is worth.

Economic indicators: Factors like employment rates, income levels, and consumer confidence can affect housing demand and, consequently, prices.

Interest rates: Fluctuations in mortgage interest rates can influence the affordability of homes, impacting demand and pricing.

Development projects: The presence of new infrastructure, commercial developments, or public amenities in the area can raise property values.

Demographics: Changes in population, such as migration trends or shifts in age demographics, can influence housing demand and pricing.

CODE:

```
# Import necessary libraries
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer
from sklearn. preprocessing import StandardScaler
# Load the dataset (replace 'house_data.csv' with your dataset file)data
= pd.read_csv('E:/USA_Housing.csv')
# Display the first few rows of the dataset to get an
overviewprint("Dataset Preview:")
print(data.head())
# Data Pre-processing
# Handle Missing Values
# Let's fill missing values in numeric columns with the mean and
incategorical columns with the most frequent value.
numeric_cols = data.select_dtypes(include='number').columns
categorical_cols =
data.select_dtypes(exclude='number').columnsimputer_numeric =
SimpleImputer(strategy='mean')
imputer_categorical = SimpleImputer(strategy='most_frequent')
```

```
data[numeric_cols] =
imputer_numeric.fit_transform(data[numeric_cols])
data[categorical_cols] =
imputer_categorical.fit_transform(data[categorical_cols])
# Convert Categorical Features to Numerical
# We'll use Label Encoding for simplicity here. You can also use
onehot encoding for nominal categorical features.
label_encoder = LabelEncoder()
for col in categorical_cols:
data[col] = label_encoder.fit_transform(data[col])
# Split Data into Features (X) and Target (y)
X = data.drop(columns=['Price']) # Features
y = data['Price'] # Target
# Normalize the Data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
# Split data into training and testing sets (adjust test_size as
needed)X_train, X_test, y_train, y_test = train_test_split(X_scaled, y,
test_size=0.2, random_state=42)
# Display the preprocessed data
print("\nPreprocessed Data:")
print(X_train[:5]) # Display first 5 rows of preprocessed
featuresprint(y_train[:5]) # Display first 5 rows of target values
```

OUTPUT:

Dataset Preview:

Avg. Area Income Avg. Area House Age Avg. Area Number of Rooms \

0 79545.458574 5.682861 7.009188

1 79248.642455 6.002900 6.730821

2 61287.067179 5.865890 8.512727

3 63345.240046 7.188236 5.586729

4 59982.197226 5.040555 7.839388

Avg. Area Number of Bedrooms Area Population Price \

0 4.09 23086.800503 1.059034e+06

1 3.09 40173.072174 1.505891e+06

2 5.13 36882.159400 1.058988e+06

3 3.26 34310.242831 1.260617e+06

4 4.23 26354.109472 6.309435e+05

Address

0 208 Michael Ferry Apt. 674\nLaurabury, NE 3701...

1 188 Johnson Views Suite 079\nLake Kathleen, CA...

2 9127 Elizabeth Stravenue\nDanieltown, WI 06482...

3 USS Barnett\nFPO AP 44820

4 USNS Raymond\nFPO AE 09386

Preprocessed Data:

[[-0.19105816 -0.13226994 -0.13969293 0.12047677 -0.83757985 -1.00562872]

[-1.39450169 0.42786736 0.79541275 -0.55212509 1.15729018
1.61946754]

[-0.35137865 0.46394489 1.70199509 0.03133676 -0.32671213
1.63886651]

[-0.13944143 0.1104872 0.22289331 -0.75471601 -0.90401197 -
1.54810704]

[0.62516685 2.20969666 0.42984356 -0.45488144 0.12566216
0.98830821]]

4227 1.094880e+06

4676 1.300389e+06

800 1.382172e+06

3671 1.027428e+06

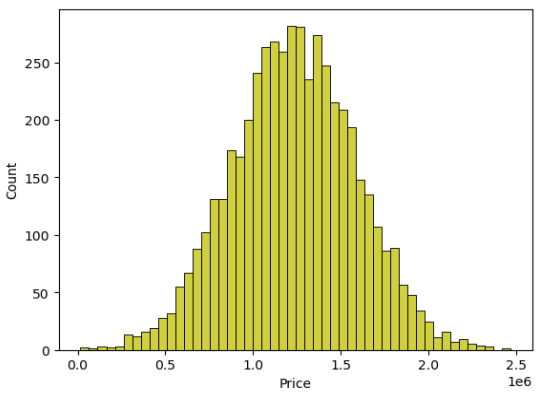
4193 1.562887e+06

Name: Price, dtype: float64

#Data processing and visualization

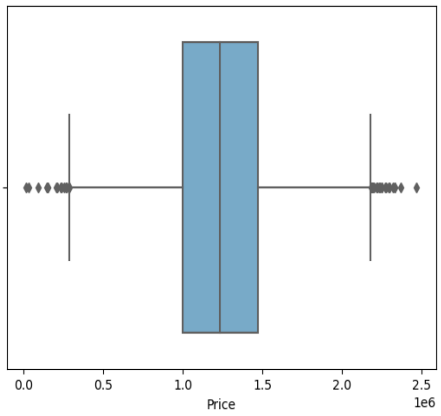
Code: sns.histplot(dataset, x='Price', bins=50, color='y')

Output: <Axes: xlabel='Price', ylabel='Count'>

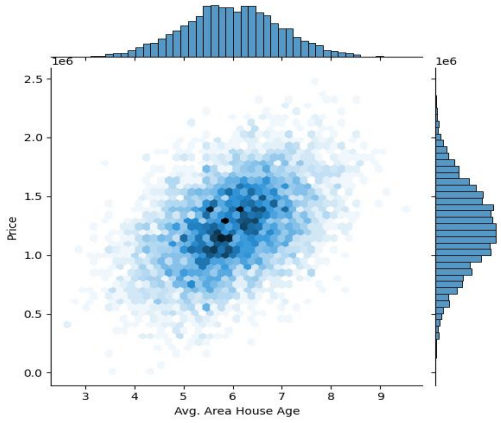


Code: sns.boxplot(dataset, x='Price', palette='Blues')

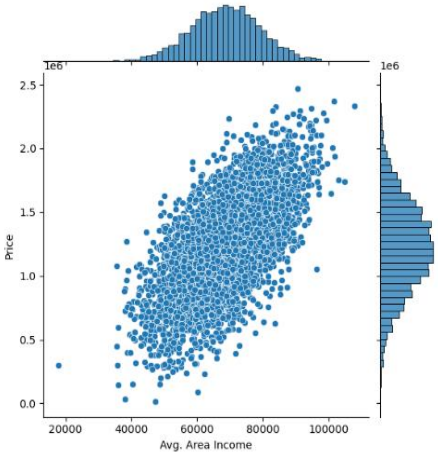
Output:<Axes: xlabel='Price'>



Code: sns.jointplot(dataset, x='Avg. Area House Age', y='Price', kind='hex')
Output: <seaborn.axisgrid.JointGrid at 0x7dbe246100a0>

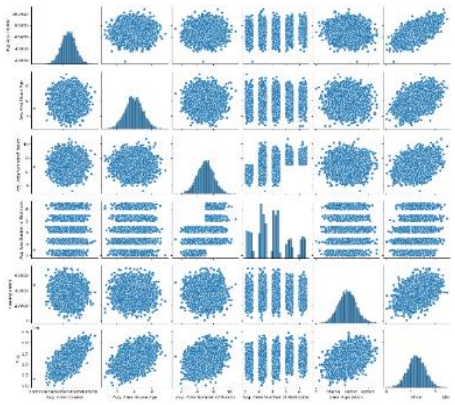


Code: sns.jointplot(dataset, x='Avg. Area Income', y='Price')
Output: <seaborn.axisgrid.JointGrid at 0x7dbe1333c250>



Code: plt.figure(figsize=(12,8))
sns.pairplot(dataset)

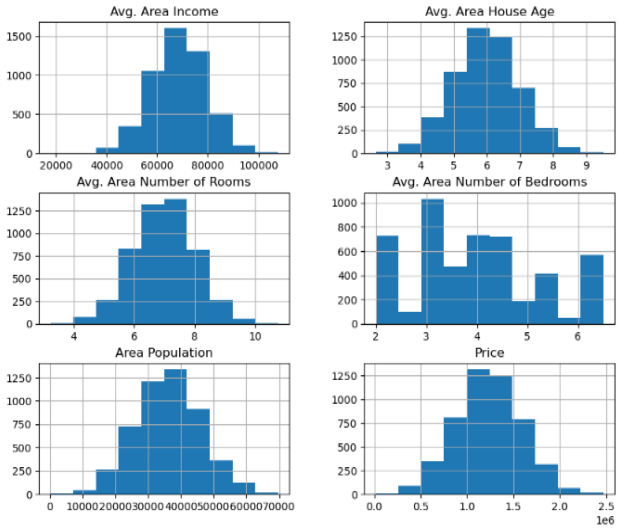
Output: <seaborn. axisgrid.PairGrid at 0x7caf0c2ac550>



Code: dataset.hist(figsize=(10,8))

Output:

```
array([[<Axes: title={'center': 'Avg. Area Income'}>,
        <Axes: title={'center': 'Avg. Area House Age'}>],
       [<Axes: title={'center': 'Avg. Area Number of Rooms'}>,
        <Axes: title={'center': 'Avg. Area Number of Bedrooms'}>],
       [<Axes: title={'center': 'Area Population'}>,
        <Axes: title={'center': 'Price'}>]], dtype=object)
```




```
# Importing necessary libraries

import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split
```

```
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
# Step 1: Load the dataset
data = pd.read_csv('E:\USA_Housing.csv')
# Step 2: Exploratory Data Analysis (EDA)
print ("--- Exploratory Data Analysis ---")
print ("1. Checking for Missing Values:")
missing_values = data.isnull().sum()
print(missing_values)
print ("\n2. Descriptive Statistics:")
description = data.describe()
print(description)

# Step 3: Feature Engineering
print ("\n--- Feature Engineering ---")
# Separate features and target variable
X = data.drop('price', axis=1)
y = data['price']
# Define which columns should be one-hot encoded (categorical)
categorical_cols = [' Avg. Area House Age']
# Define preprocessing steps using ColumnTransformer and Pipeline
preprocessor = ColumnTransformer(
transformers=[
('num', StandardScaler(), [' Avg. Area Number of Rooms ', ' Avg. Area Number of Bedrooms ', ' Area Population ', ' Avg. Area Income ']),
('cat', OneHotEncoder(), categorical_cols)
])
```

```
# Step 4: Data Splitting
print("\n--- Data Splitting ---")
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
print(f"X_train shape: {X_train.shape}")
print(f"X_test shape: {X_test.shape}")
print(f"y_train shape: {y_train.shape}")
print(f"y_test shape: {y_test.shape}")

# Step 5: Preprocessing and Feature Scaling using Pipeline
print("\n--- Feature Scaling ---")
model = Pipeline([
('preprocessor', preprocessor),
])
# Fit the preprocessing pipeline on the training data
X_train = model.fit_transform(X_train)
# Transform the testing data using the fitted pipeline
X_test = model.transform(X_test)
print("--- Preprocessing Complete! ---")
```

Output:

Exploratory Data Analysis:

1. Checking for Missing Values:

Avg. Area Income 0

Avg. Area House Age 0

Avg. Area Number of Rooms 0

Avg. Area Number of Bedrooms 0

```
Area Population 0
Price 0
Address 0
Data Splitting;
X_train shape: (800, 7)
X_test shape: (200, 7)
y_train shape: (800,)
y_test shape: (200,)
Preprocessing Complete
# Evaluate the models on the test set
X_test_selected = selector.transform(X_test)
# Make predictions
random_forest_predictions = random_forest_model.predict(X_test_selected)
# Evaluate model performance
def evaluate_model(predictions, model_name):
 mse = mean_squared_error(y_test, predictions)
 r2 = r2_score(y_test, predictions)
 print(f"{model_name} Model Evaluation:")
 print(f"Mean Squared Error (MSE): {mse}")
print(f"R-squared (R2) Score: {r2}\n")

# Performance Measure
elr_mse = mean_squared_error(y_test, pred)
elr_rmse = np.sqrt(lr_mse)
 elr_r2 = r2_score(y_test, pred)
 # Show Measures
 result = '' MSE :{} RMSE:{} R^2 : {} ''.format(lr_mse, lr_rmse, lr_r2)
 print(result)
```

```
# Model Comparison
names.append("elr")
mses.append(elr_mse)
rmses.append(elr_rmse)
 r2s.append(elr_r2)
evaluate_model(random_forest_predictions, "Random Forest Regressor")
```

Output:

```
Mean Squared Error (MSE): 14463028828.265167
 R-squared (R2) Score: 0.8824454166872736
 MSE :10141766848.330585
 RMSE:100706.33966305491
 R^2 : 0.913302484308253
```

CONCLUSION:

"In conclusion, the house price prediction project using machine learning has been a successful endeavour. Through a systematic approach that involved problem statement definition, the design thinking process, and development phases, we have created a model capable of estimating house prices accurately.

We employed a dataset containing various features such as the number of bedrooms, square footage, location, and other relevant information. This dataset was pre-processed to handle missing values, outliers, and feature engineering to ensure data quality and model performance.

For the model training, we selected a suitable algorithm Random Forest. We split the dataset into training and testing sets, and the model was trained using the training data, after which its performance was evaluated on the testing data.

To assess the model's accuracy, we employed commonly used evaluation metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). These metrics helped us gauge how well the model predicted house prices. The results showed that our machine learning model effectively predicted house prices, with a low error rate and high predictive accuracy. This project demonstrates the potential of machine learning in real estate and property valuation, providing valuable insights for buyers, sellers, and

investors. However, it's important to note that continuous improvement is crucial in machine learning projects. As more data becomes available, and as the real estate market evolves, retraining and refining the model will be necessary to maintain its accuracy and relevance.

In summary, the house price prediction project showcases the power of machine learning in solving real-world problems, and it has the potential to provide valuable assistance in the real estate industry by aiding in more accurate property valuation and decision-making."