PREDICTING HOUSE PRICE USING MACHINE LEARNING

Introduction:

House Price Prediction

- Whether you're a homeowner looking to estimate the value of your property, a real estate investor seeking profitable opportunities, or a data scientist aiming to build a predictive model, the foundation of this endeavor lies in loading and preprocessing the dataset.

- Building a house price prediction model is a data-driven process that involves harnessing the power of machine learning to analyze historical housing data and make informed price predictions. This journey begins with the fundamental steps of data loading and preprocessing.

- This introduction will guide you through the initial steps of the process. We'll explore how to import essential libraries, load the housing dataset, and perform critical preprocessing steps. Data preprocessing is crucial as it helps clean, format, and prepare the data for further analysis. This includes handling missing values, encoding categorical variables, and ensuring that the data is appropriately scaled.

Given data set:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Address
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06	208 Michael Ferry 674\nLaurabury, N 3701...
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 Johnson View 079\nLake Kathlee
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06	9127 Elizabeth Stravenue\nDaniel WI 06482...
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	USS Barnett\nFPO 44820
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond\n 09386
...	...	...	...	...	...	...	...
4995	60567.944140	7.830362	6.137356	3.46	22837.361035	1.060194e+06	USNS Williams\nF 30153-7653
4996	78491.275435	6.999135	6.576763	4.02	25616.115489	1.482618e+06	PSC 9258, Box 8489\nAPO AA 42 3352
4997	63390.686886	7.250591	4.805081	2.13	33266.145490	1.030730e+06	4215 Tracy Garde 076\nJoshualand, 01...
4998	68001.331235	5.534388	7.130144	5.44	42625.620156	1.198657e+06	USS Wallace\nFPO 73316
4999	65510.581804	5.992305	6.792336	4.07	46501.283803	1.298950e+06	37778 George Rid Apt. 509\nEast Ho

5000 Rows x 7 Columns

Necessary step to follow:

• Import Libraries:

Start by importing the necessary libraries:

Program:

import

pandas

as pd

```
import

numpy

as np

from sklearn.model_selection import

train_test_split from

sklearn.preprocessing import

StandardScaler
```

- **Load the Dataset:**

Load your dataset into a Pandas DataFrame. You can typically find house price datasets in CSV format, but you can adapt this code to other formats as needed.

Program:

```
df = pd.read_csv('

E:\USA_Housing.csv ')

Pd.read()
```

- **. Exploratory Data Analysis (EDA):**

Perform EDA to understand your data better. This includes checking for missing values, exploring

the data's statistics, and visualizing it to identify patterns.

Program:

Check for

missing values

print(df.isnull()

.sum())

#

Explor

e

statistic

s

print(df

.descri

be())

Visualize the data (e.g., histograms, scatter plots, etc.)

- ### **. Feature Engineering:**

    Depending on your dataset, you may need to create new features or transform existing ones. This can involve

one-hot encoding categorical variables, handling date/time data, or scaling numerical features.

Program:

\# Example: One-hot encoding for categorical variables

df = pd.get_dummies(df, columns=[' Avg. Area Income ', ' Avg. Area House Age '])

• . Split the Data:

Split your dataset into training and testing sets. This helps you evaluate your model's performance later.

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)

• . Feature Scaling:

Apply feature scaling to normalize your data, ensuring that all features have similar scales. Standardization (scaling to mean=0 and std=1) is a common choice.

Program:

scaler = StandardScaler()

X_train =

scaler.fit_transform(X_trai

n) X_test =

scaler.transform(X_test)

Importance of loading and processing dataset:

Loading and preprocessing the dataset is an important first step in building any machine learning model. However, it is especially important for house price prediction models, as house price datasets are often complex and noisy.

By loading and preprocessing the dataset, we can ensure that the machine learning algorithm is able to learn from the data effectively and accurately.

Challenges involved in loading and preprocessing a house price dataset;

There are a number of challenges involved in loading and preprocessing a house price dataset, including:

☐ **H**

andling missing values:

House price datasets often contain missing values, which can

be due to a variety of factors, such as human error or incomplete data collection. Common methods for handling missing values include dropping the rows with missing values, imputing the missing values with the mean or median of the

feature, or using a more sophisticated method such as multiple imputation.

- **Encoding categorical variables:**

House price datasets often contain categorical features, such as the type of house, the neighborhood, and the school district. These features need to be encoded before they can be used by machine learning models. One common way to encode categorical variables is to use one-hot encoding.

- **Scaling the features:**

It is often helpful to scale the features before training a machine learning model. This can help to improve the performance of the model and make it more robust to outliers. There are a variety of ways to scale the features, such as min-max scaling and standard scaling.

- **Splitting the dataset into training and testing sets:**

Once the data has been pre-processed, we need to split the dataset into training and testing sets. The training set will be used to train the model, and the testing set will be used to evaluate the performance of the model on unseen data. It is important to split the dataset in a way that is representative of the real world distribution of the data.

How to overcome the challenges of loading and preprocessing a house price dataset:

There are a number of things that can be done to overcome the challenges of loading and preprocessing a house price dataset, including:

- **Use a data preprocessing library:**

  There are a number of libraries available that can help with data preprocessing tasks, such as handling missing values, encoding categorical variables, and scaling the features.

- **Carefully consider the specific needs of your model:**

  The best way to preprocess the data will depend on the specific machine learning algorithm that you are using. It is important to carefully consider the requirements of the algorithm and to preprocess the data in a way that is compatible with the algorithm.

- **Validate the preprocessed data:**

  It is important to validate the preprocessed data to ensure that it is in a format that can be used by the machine learning algorithm and that it is of high quality. This can be done by inspecting the data visually or by using statistical methods.

• Loading the dataset:

- Loading the dataset using machine learning is the process of bringing the data into the machine learning environment so that it can be used to train and evaluate a model.

- The specific steps involved in loading the dataset will vary depending on the machine learning library or framework that is being used.

However, there are some general steps that are common to most machine learning frameworks:

- **Identify the dataset:**

  The first step is to identify the dataset that you want to load. This dataset may be stored in a local file, in a database, or in a cloud storage service.

- **Load the dataset:**

  Once you have identified the dataset, you need to load it into the machine learning environment. This may involve using a built-in function in the machine learning library, or it may involve writing your own code.

- **Preprocess the dataset:**

  Once the dataset is loaded into the machine learning environment, you may need to preprocess it before you can start training and evaluating your model. This may involve cleaning the data, transforming

the data into a suitable format, and splitting the data into training and test sets.

Here, how to load a dataset using machine learning in Python

Program:

```python
import
pandas as
pd import
numpy as
np import
seaborn
as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import
train_test_split from
sklearn.preprocessing import
StandardScaler


from sklearn.metrics import
r2_score,
mean_absolute_error,mean_sq
uared_error
from sklearn.linear_model import
LinearRegression from
sklearn.linear_model import Lasso
```

```python
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
import xgboost as xg
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
```

/opt/conda/lib/python3.10/site-packages/scipy/__init__.py:146: UserWarning: A NumPy version >=1.16.5 and <1.23.0 is required for this version of SciPy (detected version 1.23.5

  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"

Loading Dataset:

```python
dataset = pd.read_csv('E:/USA_Housing.csv')
```

Data Exploration:

Dataset:

Output:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Address
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06	208 Michael Ferry Apt 674\nLaurabury, NE 3701...
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 Johnson Views Su 079\nLake Kathleen, C
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06	9127 Elizabeth Stravenue\nDanieltown WI 06482...
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	USS Barnett\nFPO AP 44820
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond\nFPO 09386
...	...	...	...	...	...	...	...
4995	60567.944140	7.830362	6.137356	3.46	22837.361035	1.060194e+06	USNS Williams\nFPO A 30153-7653
4996	78491.275435	6.999135	6.576763	4.02	25616.115489	1.482618e+06	PSC 9258, Box 8489\nAPO AA 42991 3352
4997	63390.686886	7.250591	4.805081	2.13	33266.145490	1.030730e+06	4215 Tracy Garden Su 076\nJoshualand, VA 01...
4998	68001.331235	5.534388	7.130144	5.44	42625.620156	1.198657e+06	USS Wallace\nFPO AE 73316
4999	65510.581804	5.992305	6.792336	4.07	46501.283803	1.298950e+06	37778 George Ridges Apt. 509\nEast Holly,

- ## Preprocessing the dataset:

- Data preprocessing is the process of cleaning, transforming, and integrating data in order to make it ready for analysis.

- This may involve removing errors and inconsistencies, handling missing values, transforming the data into a consistent format, and scaling the data to a suitable range.
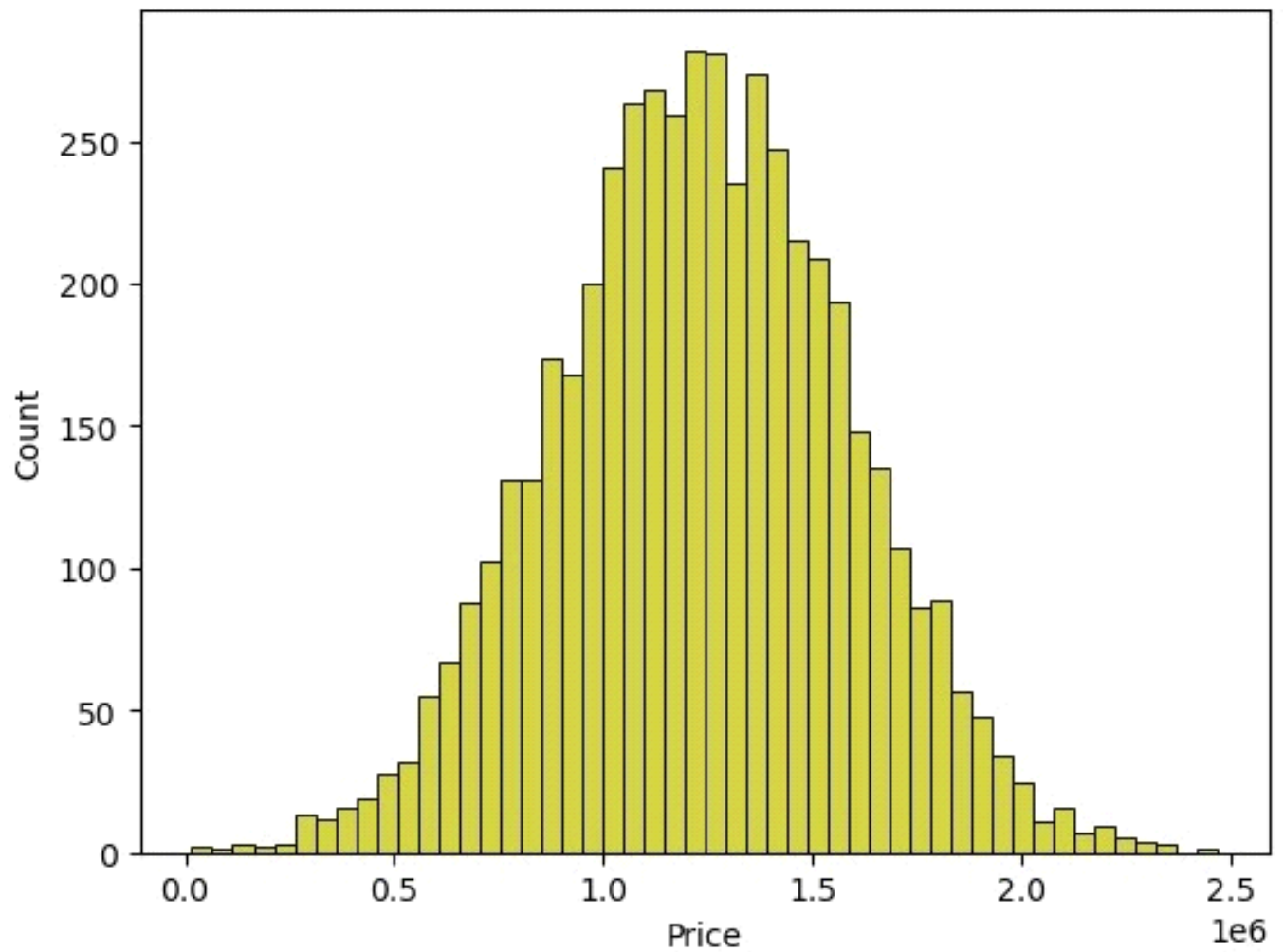
Visualisation and Pre-Processing of Data:

In [1]:

```python
sns.histplot(dataset, x='Price', bins=50, color='y')
```

Out[1]:

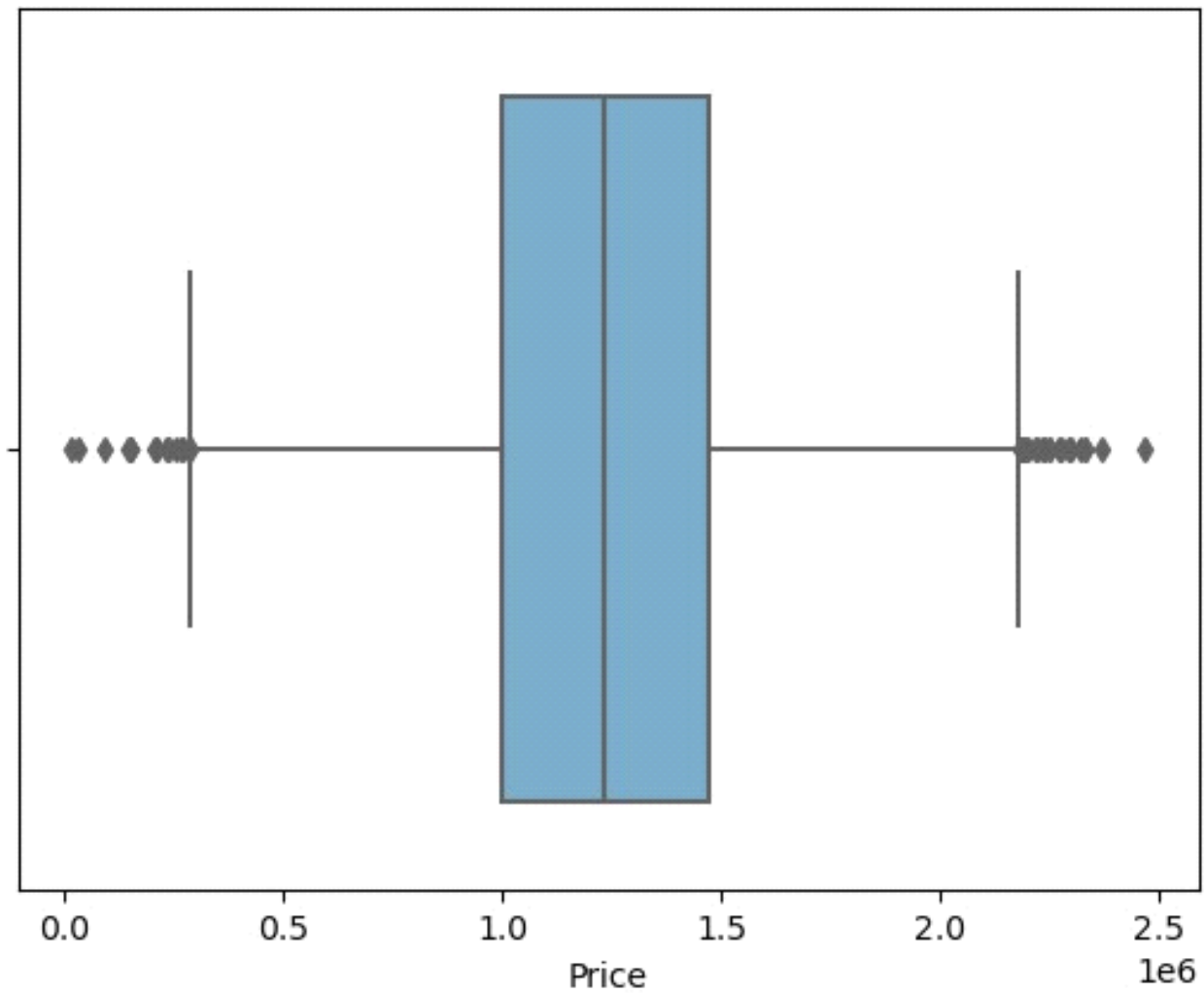<Axes: xlabel='Price', ylabel='Count'>

In [2]:

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

Out[2]:
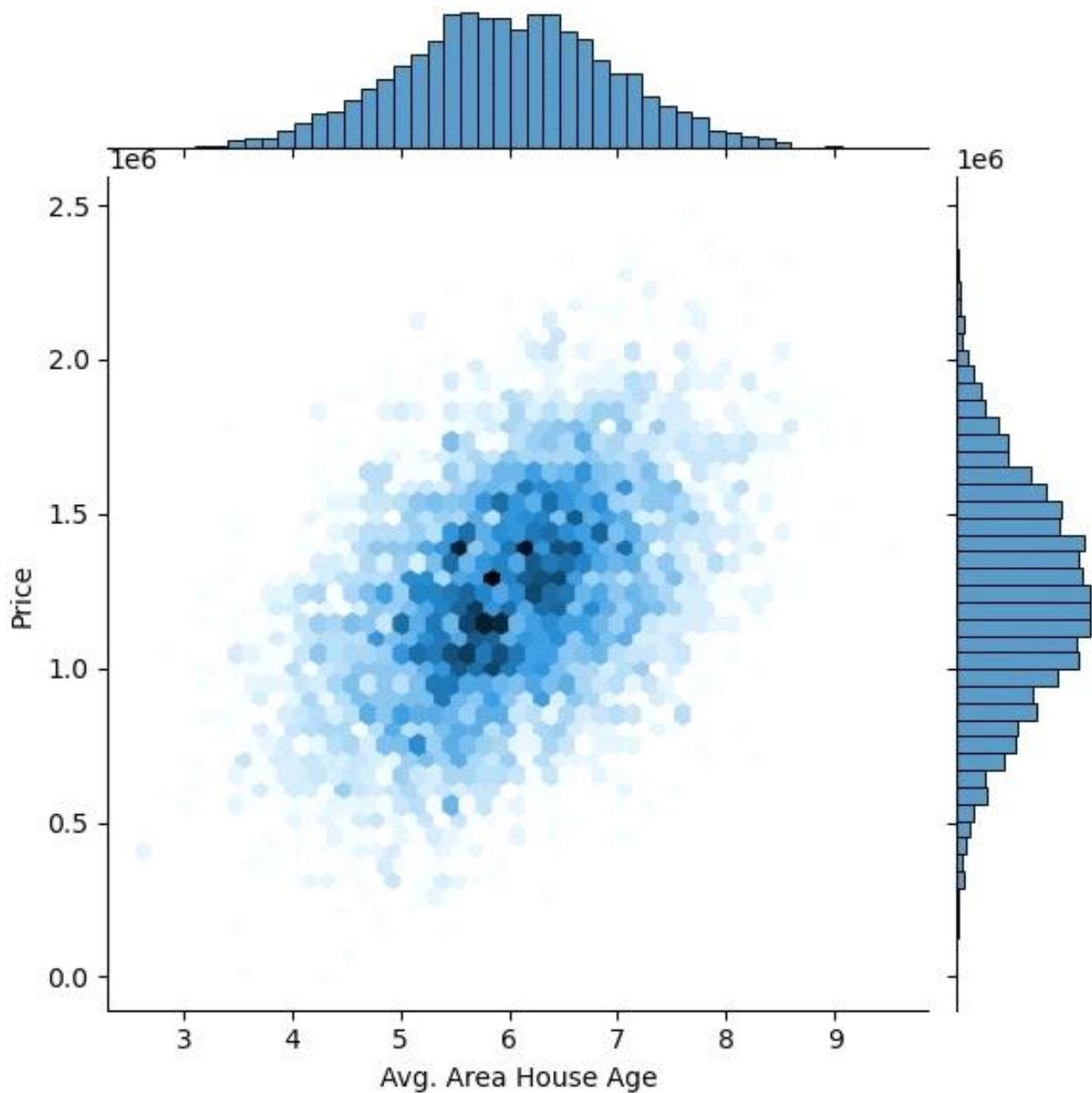
<Axes: xlabel='Price'>

In [3]:

```
sns.jointplot(dataset, x='Avg. Area House Age', y='Price', kind='hex')
```

Out[3]:

<seaborn.axisgrid.JointGrid at 0x7caf1d571810>
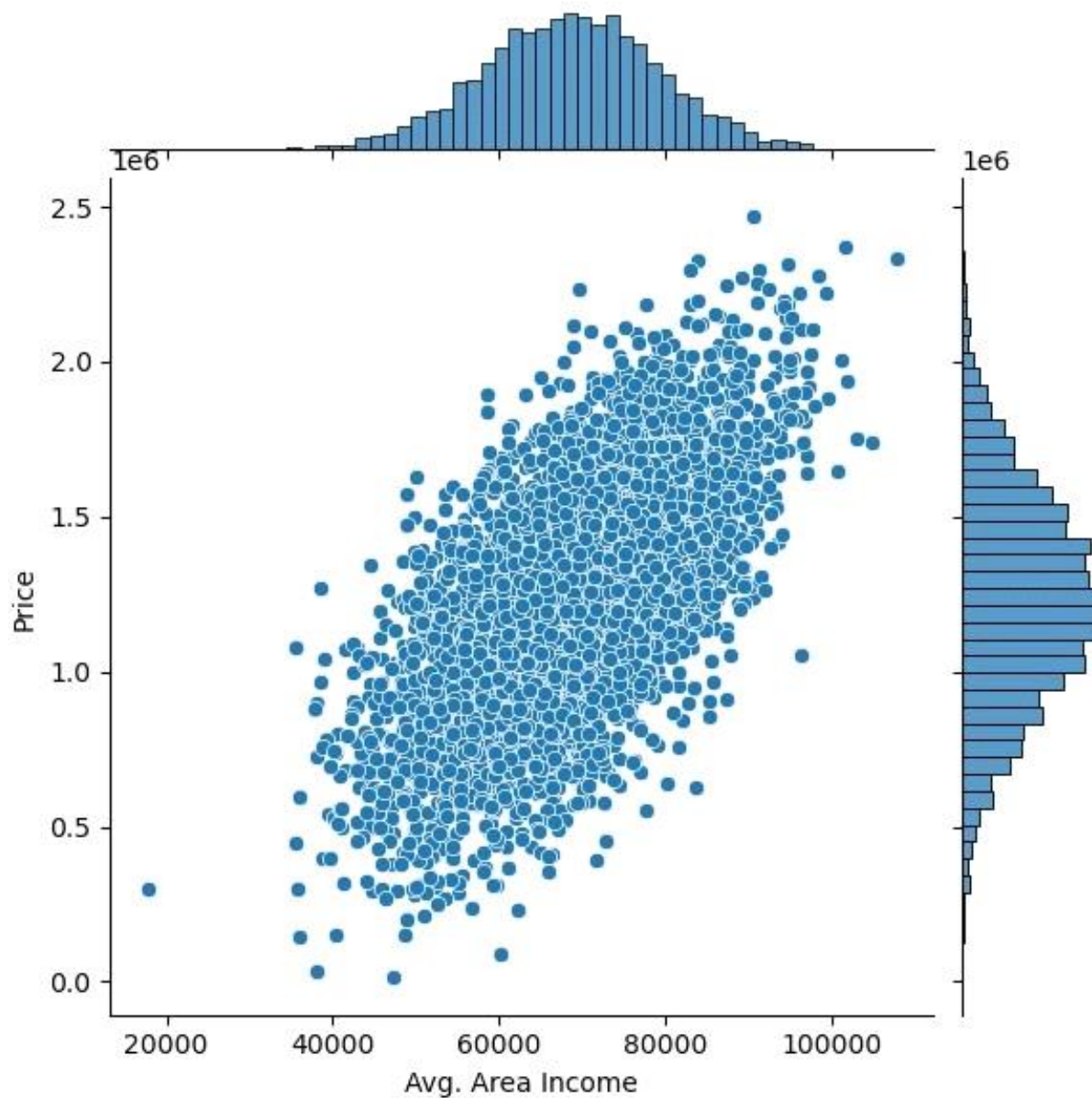
In [4]:

```
sns.jointplot(dataset, x='Avg. Area Income', y='Price')
```

Out[4]:

<seaborn.axisgrid.JointGrid at 0x7caf1d8bf7f0>

In [5]:

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

Out[5]:

<seaborn.axisgrid.PairGrid at 0x7caf0c2ac550>

<Figure size 1200x800 with 0 Axes>

In [6]:

```
dataset.hist(figsize=(10,8))
```

Out[6]:

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'}>,

Visualising Correlation:

In [7]:

	Avg. Area	Avg. Area Hous	Avg. Area Number	Avg. Area Number	Area Populati	P

	Incom e	e Age	of Rooms	of Bedroo ms	on	r i c e
Avg. Area Income	1.0000 00	-0.00200 7	-0.011032	0.019788	-0.01623 4	0.6 3 9 7 3 4
Avg. Area House Age	-0.00200 7	1.00000 0	-0.009428	0.006149	-0.01874 3	0.4 4 5

					2543
Avg. Area Number of Rooms	-0.011032	-0.009428	1.000000	0.462695	0.002040	0.3335664
Avg. Area Number of	0.019788	0.006149	0.4	1.000000	-0.022168	0.1

Bedrooms			62695			71071
Area Population	-0.016234	-0.018743	0.002040	-0.022168	1.000000	0.408556
Price	0.639734	0.452543	0.0333	0.171071	0.408556	1.000

e			5			0
			6			0
			6			0
			4			0

```
dataset.corr(numeri

c_only=True)
```
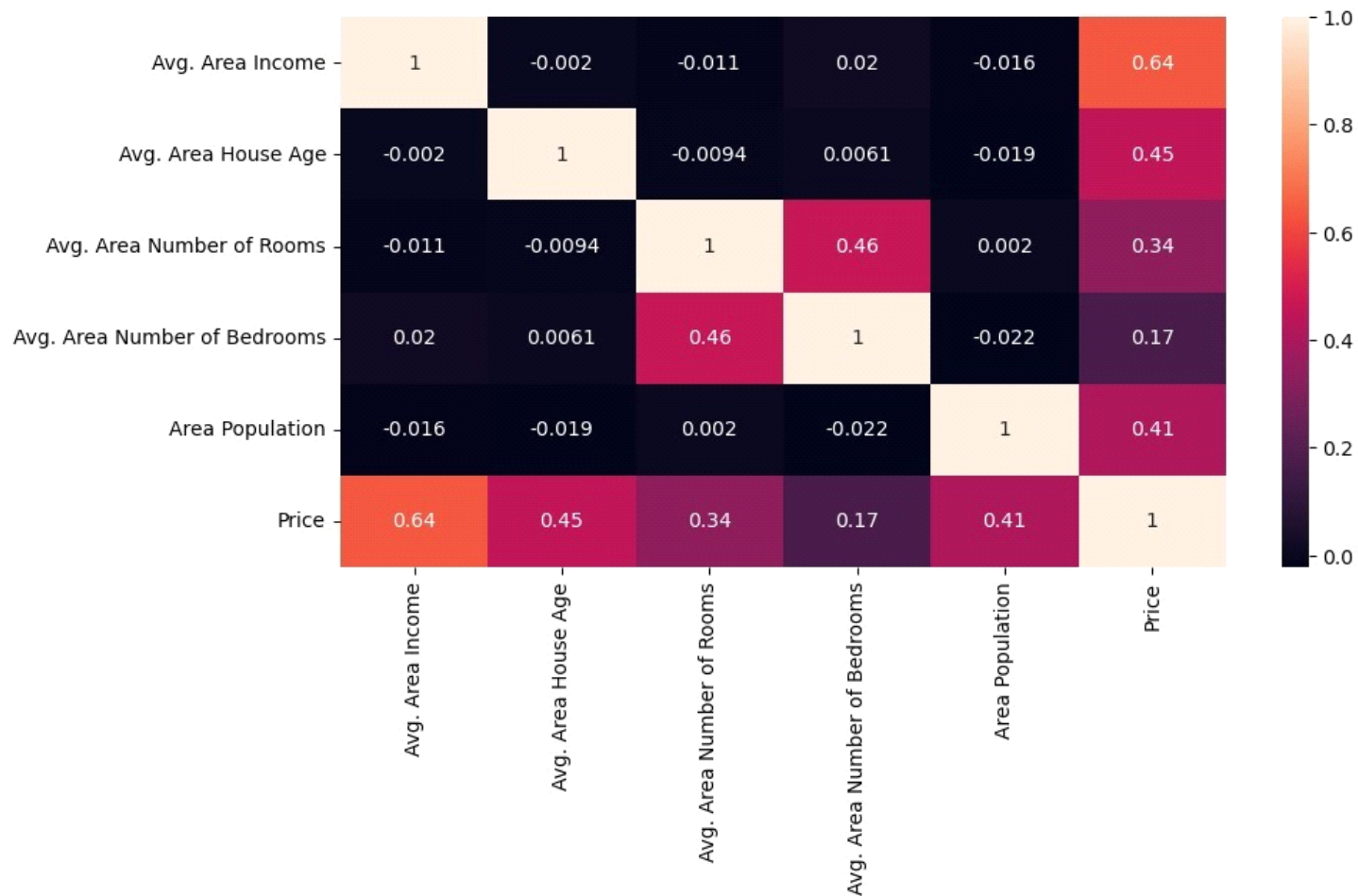
Out[7]:

In [8]:

```
plt.figure(figsize=(10,5))sns.heatmap(dataset.corr(numeric_only =
True), annot= True)
```

Out[8]:

<Axes: >

Some common data preprocessing tasks include:

- **Data cleaning:** This involves identifying and correcting errors and inconsistencies in the data. For example, this may involve removing duplicate records, correcting typos, and filling in missing values.

- **Data transformation:** This involves converting the data into a format that is suitable for the analysis task. For example, this may involve converting categorical data to numerical data, or scaling the data to a suitable range.

- **Feature engineering:** This involves creating new features from the existing data. For example, this may involve creating features that represent interactions between variables, or features that represent summary statistics of the data.

- **Data integration:** This involves combining data from multiple sources into a single dataset. This may involve resolving inconsistencies in the data, such as different data formats or different variable names.

Data preprocessing is an essential step in many data science projects. By carefully preprocessing the data, data scientists can improve the accuracy and reliability of their results.

Program:

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

```python
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
```

```python
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

```python
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

```python
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

2. Descriptive Statistics:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms
count	5000.000000	5000.000000	5000.000000	5000.000000
mean	62748.865	6.028323445	6.997892	4.25
std	2500.025031	3.934212	3.979123	1.462725
min	17796.63	2.644304186	3.236194	2
max	107701.7	9.519088066	10.75959	6.5

Area Population	Price
5000.000000	5000.000000

34897.16035	20314.66
1.469203	50.504174
172.6107	15938.66
69621.71	2469066
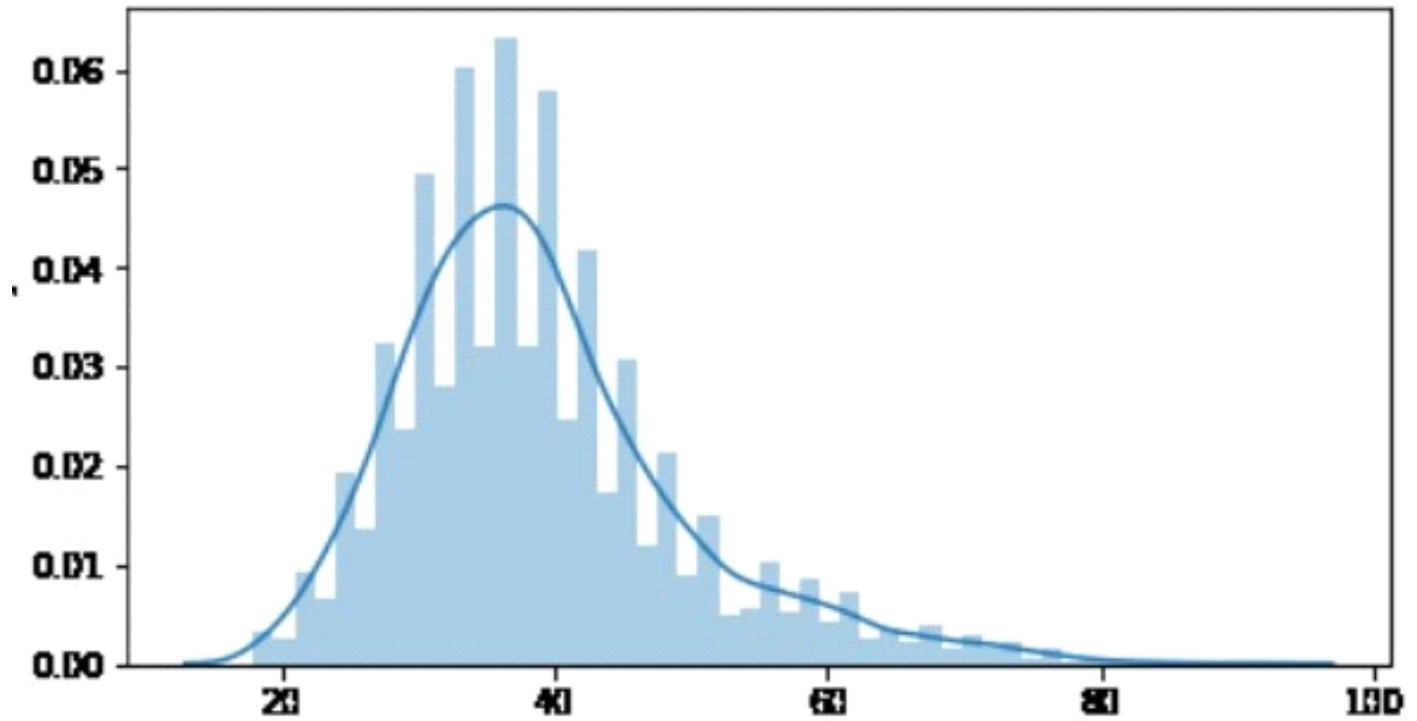
Avg.Area House Age

Data Splitting;

X_train shape: (800, 7)

X_test shape: (200, 7)

y_train shape: (800,)

y_test shape: (200,)

Preprocessing Complete

Conclusion:

- In the quest to build a house price prediction model, we have embarked on a critical journey that begins with loading and preprocessing the dataset.We have traversed through essential steps, starting with importing the necessary libraries to facilitate data manipulation and analysis.

- Understanding the data's structure, characteristics, and any potential issues through exploratory data analysis (EDA) is essential for informed decision-making.

- Data preprocessing emerged as a pivotal aspect of this process. It involves cleaning, transforming, and refining the dataset to ensure that it aligns with the requirements of machine learning algorithms.

- With these foundational steps completed, our dataset is now primed for the subsequent stages of building and training a house price prediction model.