PROJECT NAME: A Linear Regression Model to Predict House Prices in the USA Using Machine Learning

BY: Jacktone Etemesi

EMAIL: jacktoneetemesi1@gmail.com

Introduction

This notebook file contains the code for a linear regression model for predicting the price of houses in the USA based on the following attributes:

- *Average income earned by people in the area*
- *Average age of houses in the area*
- *Average number of rooms in houses within the area*
- *Average number of bedrooms in houses within the area*
- *Population of the area.*

Data Source

The data used in this analysis was collected and compiled by GANTALA SWETHA and can be found on this Link:

1. Importing Data & Relevant Libraries

In [ ]:
```
# importing relevant libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [ ]:
```
# importing data

df=pd.read_csv('USA_Housing.csv')
```

2. Data Cleaning

Under this section, we will take a peek into our dataset to understand it better. We will do this by exploring the following:

1. *Variable names*
2. *Variables data types*
3. *Some descriptive statistics on numerical columns*
4. *Correlation between variables*

In [ ]:
```
df.head()
```

Out[ ]:

	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 Suite 079\nLake Kathleen, CA...
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 AE 09386

In [ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 7 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   Avg. Area Income              5000 non-null   float64
 1   Avg. Area House Age           5000 non-null   float64
 2   Avg. Area Number of Rooms     5000 non-null   float64
 3   Avg. Area Number of Bedrooms  5000 non-null   float64
 4   Area Population               5000 non-null   float64
 5   Price                         5000 non-null   float64
 6   Address                       5000 non-null   object 
dtypes: float64(6), object(1)
memory usage: 273.6+ KB
```

In [ ]:
```
df.describe()
```

Out[ ]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5.000000e+03
mean	68583.108984	5.977222	6.987792	3.981330	36163.516039	1.232073e+06
std	10657.991214	0.991456	1.005833	1.234137	9925.650114	3.531176e+05
min	17796.631190	2.644304	3.236194	2.000000	172.610686	1.593866e+04
25%	61480.562388	5.322283	6.299250	3.140000	29403.928702	9.975771e+05
50%	68804.286404	5.970429	7.002902	4.050000	36199.406689	1.232669e+06
75%	75783.338666	6.650808	7.665871	4.490000	42861.290769	1.471210e+06
max	107701.748378	9.519088	10.759588	6.500000	69621.713378	2.469066e+06

In [ ]:
```
# checking for nulls
df.isnull().sum()
```

Out[ ]:
```
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
dtype: int64
```

In [ ]:
```
#checking for duplicate values
df.duplicated().sum()
```

Out[ ]:
```
0
```

In [ ]:
```
# checking for Multicollinearity between our variables
df.corr(numeric_only=True)
```

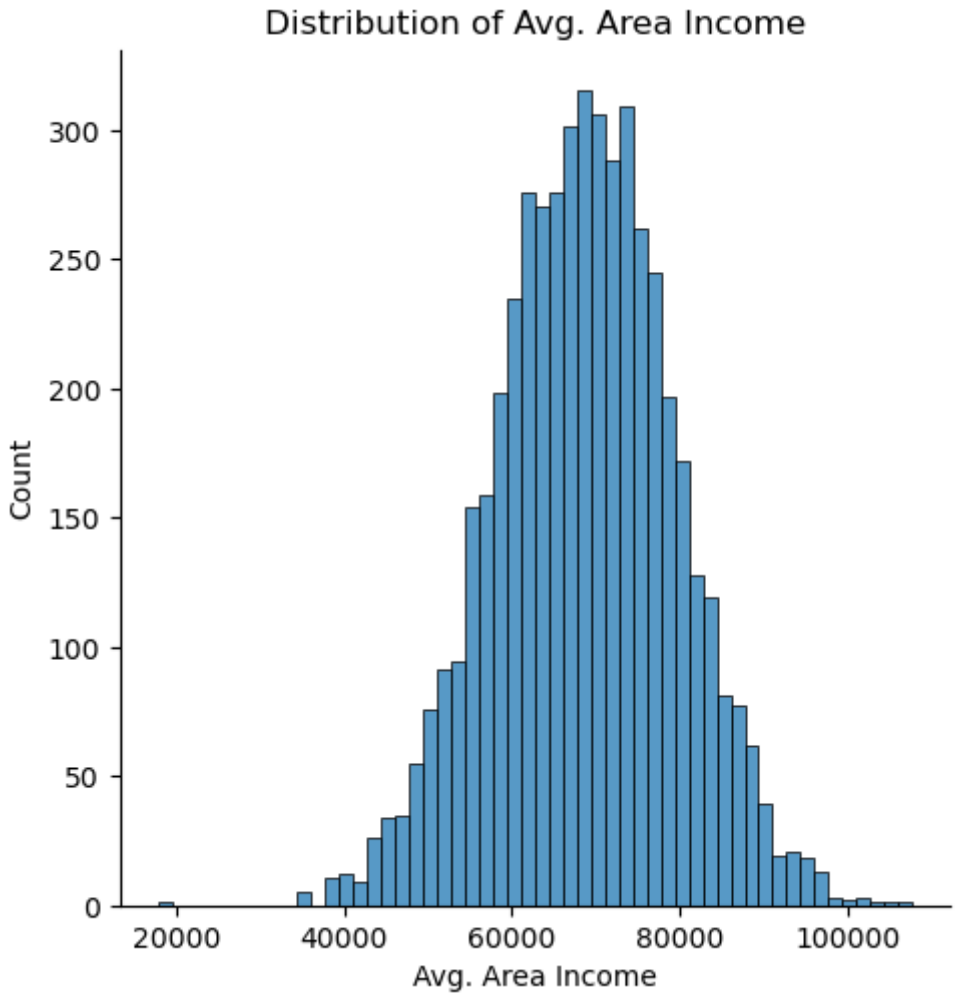
Out[ ]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price
Avg. Area Income	1.000000	-0.002007	-0.011032	0.019788	-0.016234	0.639734
Avg. Area House Age	-0.002007	1.000000	-0.009428	0.006149	-0.018743	0.452543
Avg. Area Number of Rooms	-0.011032	-0.009428	1.000000	0.462695	0.002040	0.335664
Avg. Area Number of Bedrooms	0.019788	0.006149	0.462695	1.000000	-0.022168	0.171071
Area Population	-0.016234	-0.018743	0.002040	-0.022168	1.000000	0.408556
Price	0.639734	0.452543	0.335664	0.171071	0.408556	1.000000

3. Exploratory Data Analysis

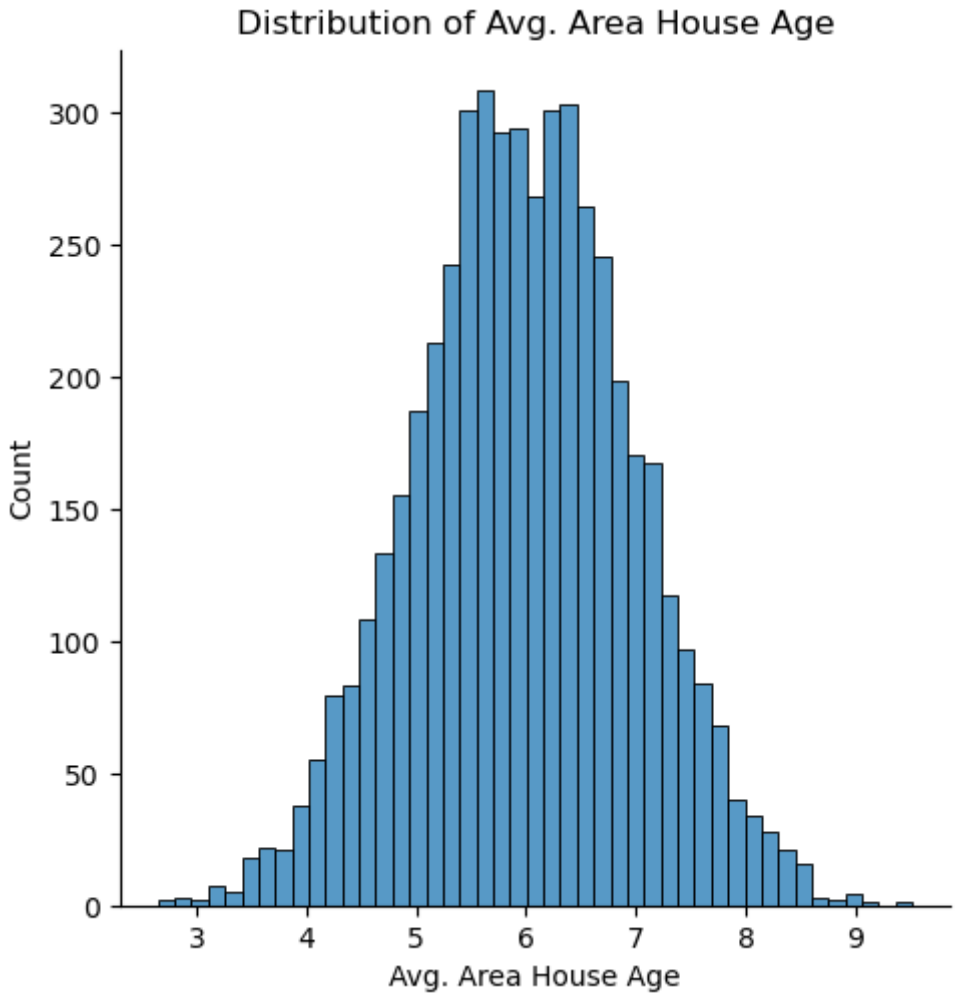
In [ ]:
```
sns.displot(data=df,x='Avg. Area Income',label=True).set(title='Distribution of Avg. Area Income')
```

Out[ ]:
```
<seaborn.axisgrid.FacetGrid at 0x14fdc2daa00>
```



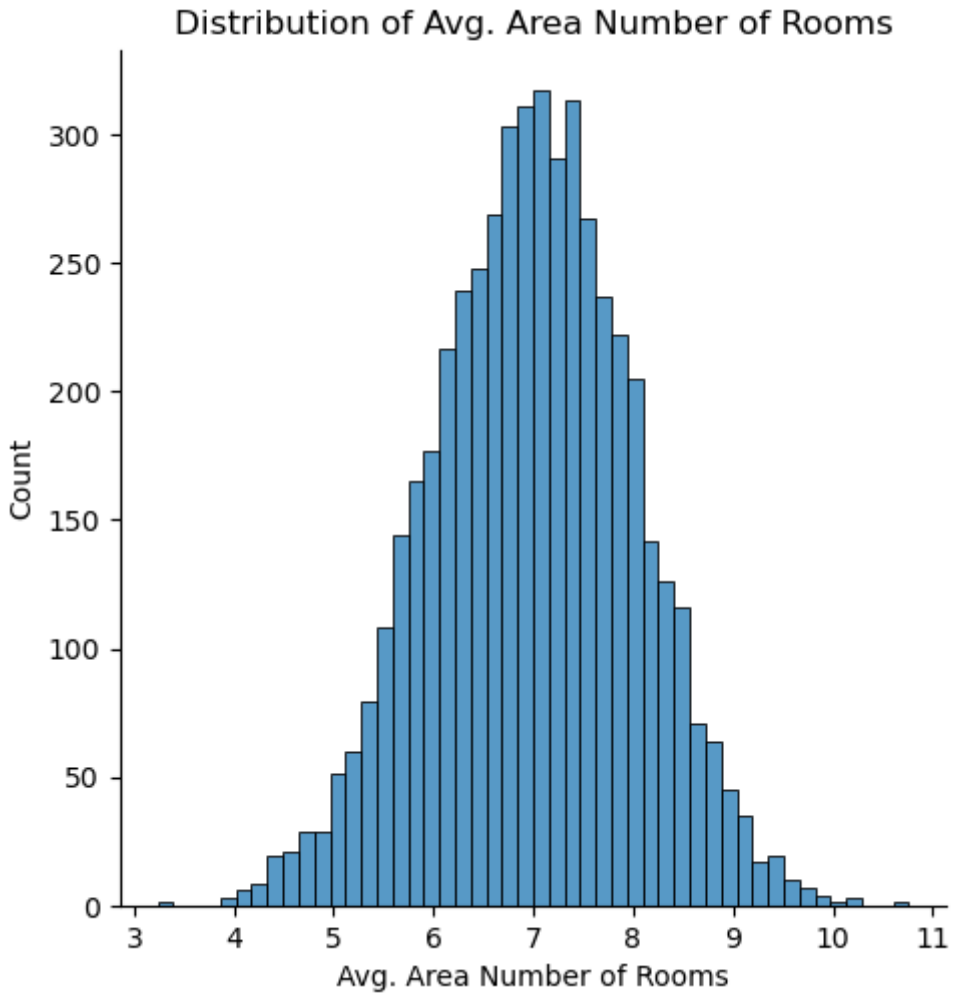
In [ ]:
```
sns.displot(data=df,x='Avg. Area House Age').set(title='Distribution of Avg. Area House Age')
```

Out[ ]:
```
<seaborn.axisgrid.FacetGrid at 0x14fe197a9a0>
```



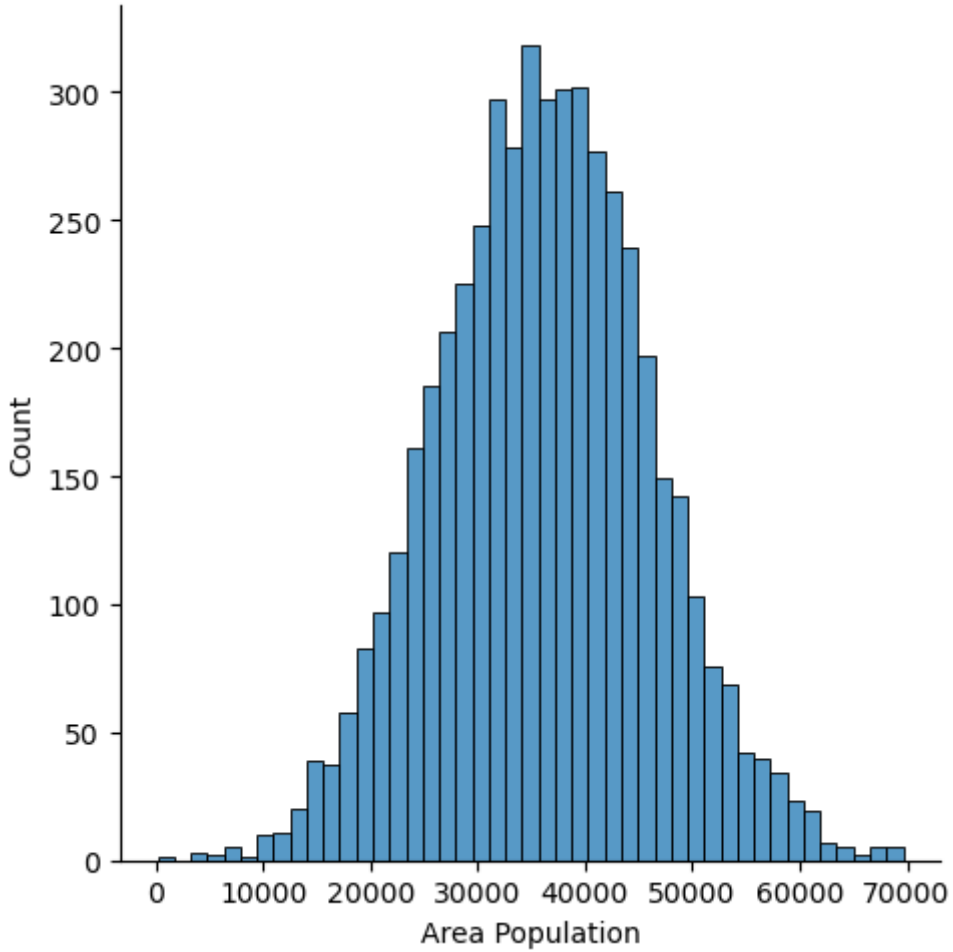
In [ ]:
```
sns.displot(data=df,x='Avg. Area Number of Rooms').set(title='Distribution of Avg. Area Number of Rooms')
```

Out[ ]:
```
<seaborn.axisgrid.FacetGrid at 0x14fdc1beb80>
```



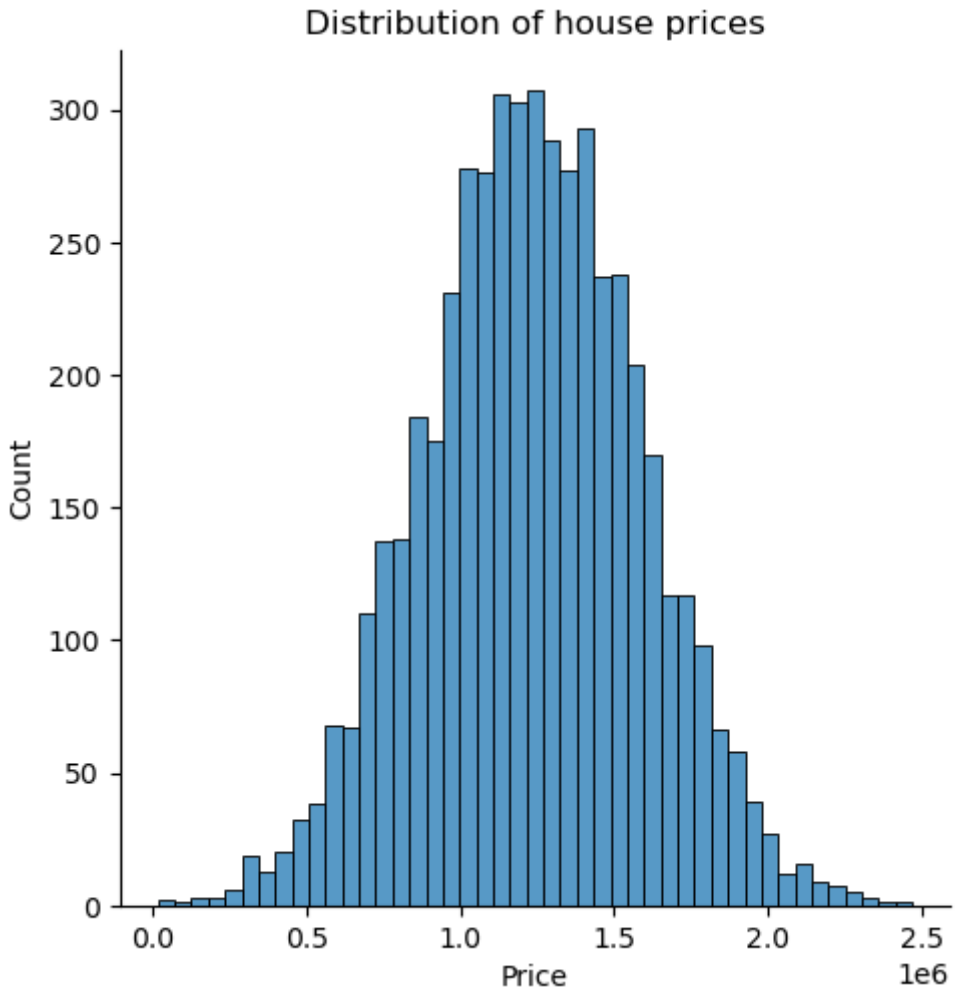
In [ ]:
```
sns.displot(data=df,x='Area Population')
```

Out[ ]:
```
<seaborn.axisgrid.FacetGrid at 0x14fe19d4ac0>
```



In [ ]:
```
sns.displot(data=df,x='Price').set(title='Distribution of house prices')
```

Out[ ]:
```
<seaborn.axisgrid.FacetGrid at 0x14fe19011f0>
```



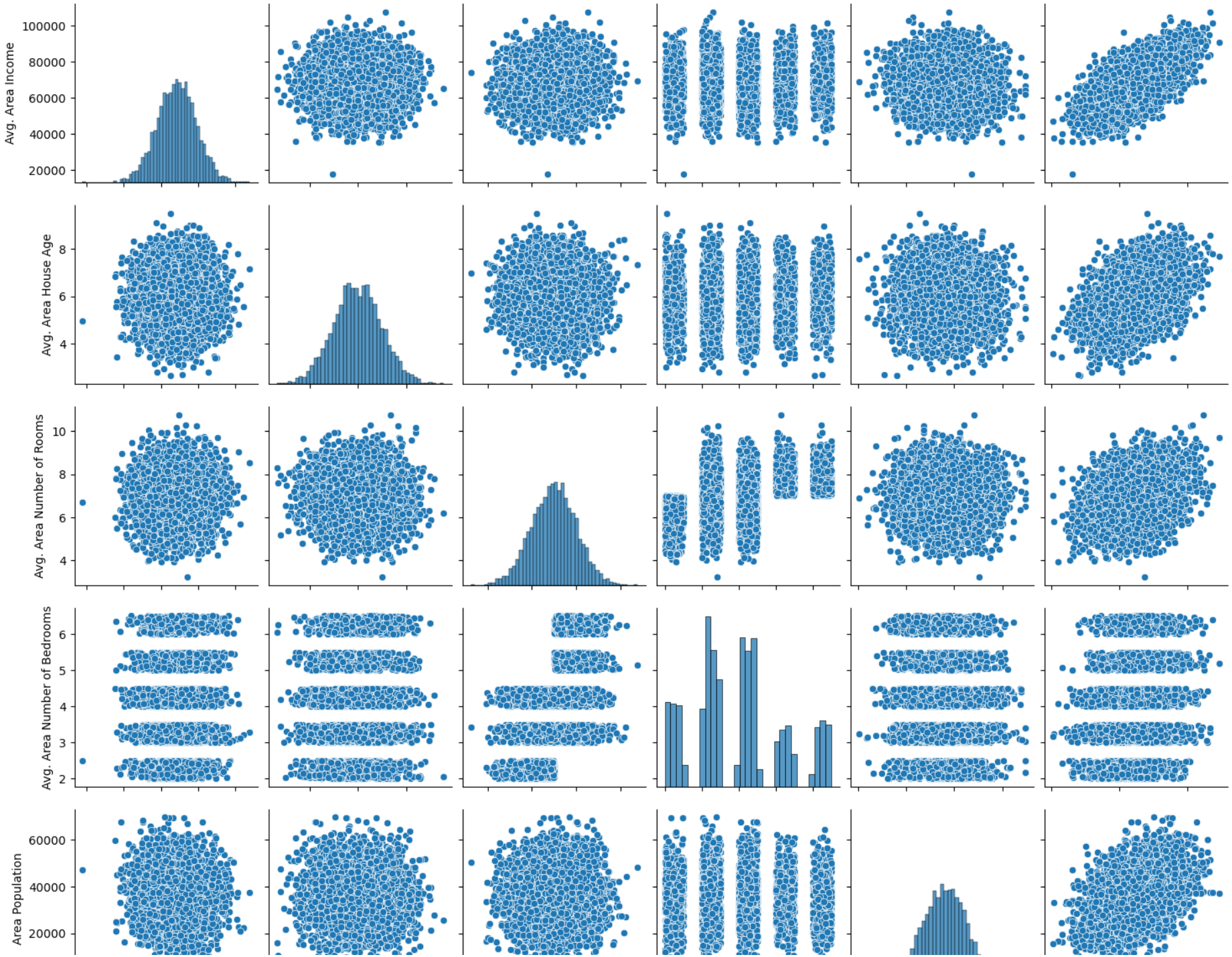
In [ ]:
```
# checking for Multicollinearity between our variables
df.corr(numeric_only=True)
```

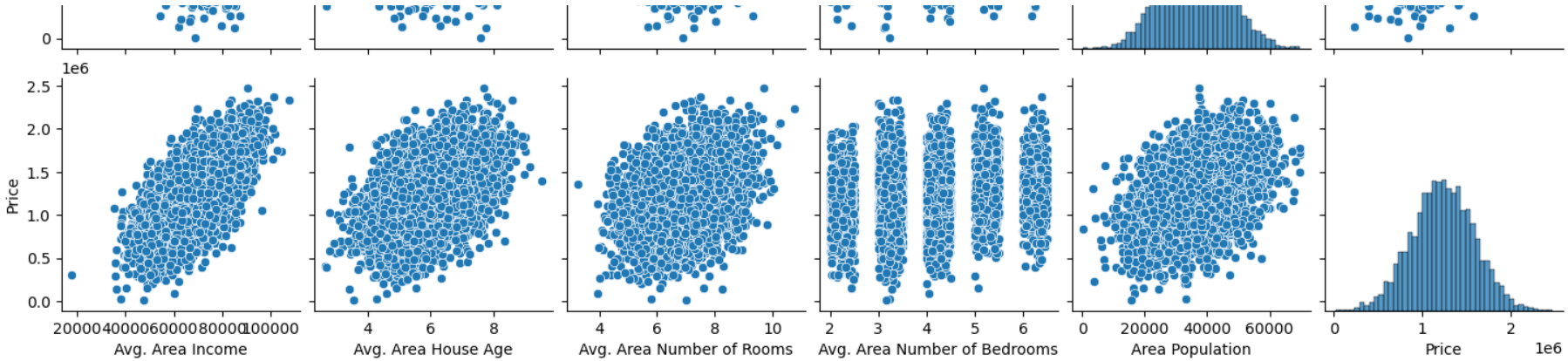
Out[ ]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price
Avg. Area Income	1.000000	-0.002007	-0.011032	0.019788	-0.016234	0.639734
Avg. Area House Age	-0.002007	1.000000	-0.009428	0.006149	-0.018743	0.452543
Avg. Area Number of Rooms	-0.011032	-0.009428	1.000000	0.462695	0.002040	0.335664
Avg. Area Number of Bedrooms	0.019788	0.006149	0.462695	1.000000	-0.022168	0.171071
Area Population	-0.016234	-0.018743	0.002040	-0.022168	1.000000	0.408556
Price	0.639734	0.452543	0.335664	0.171071	0.408556	1.000000

In [ ]:
```
# visualizing the correlation
sns.pairplot(df)
```

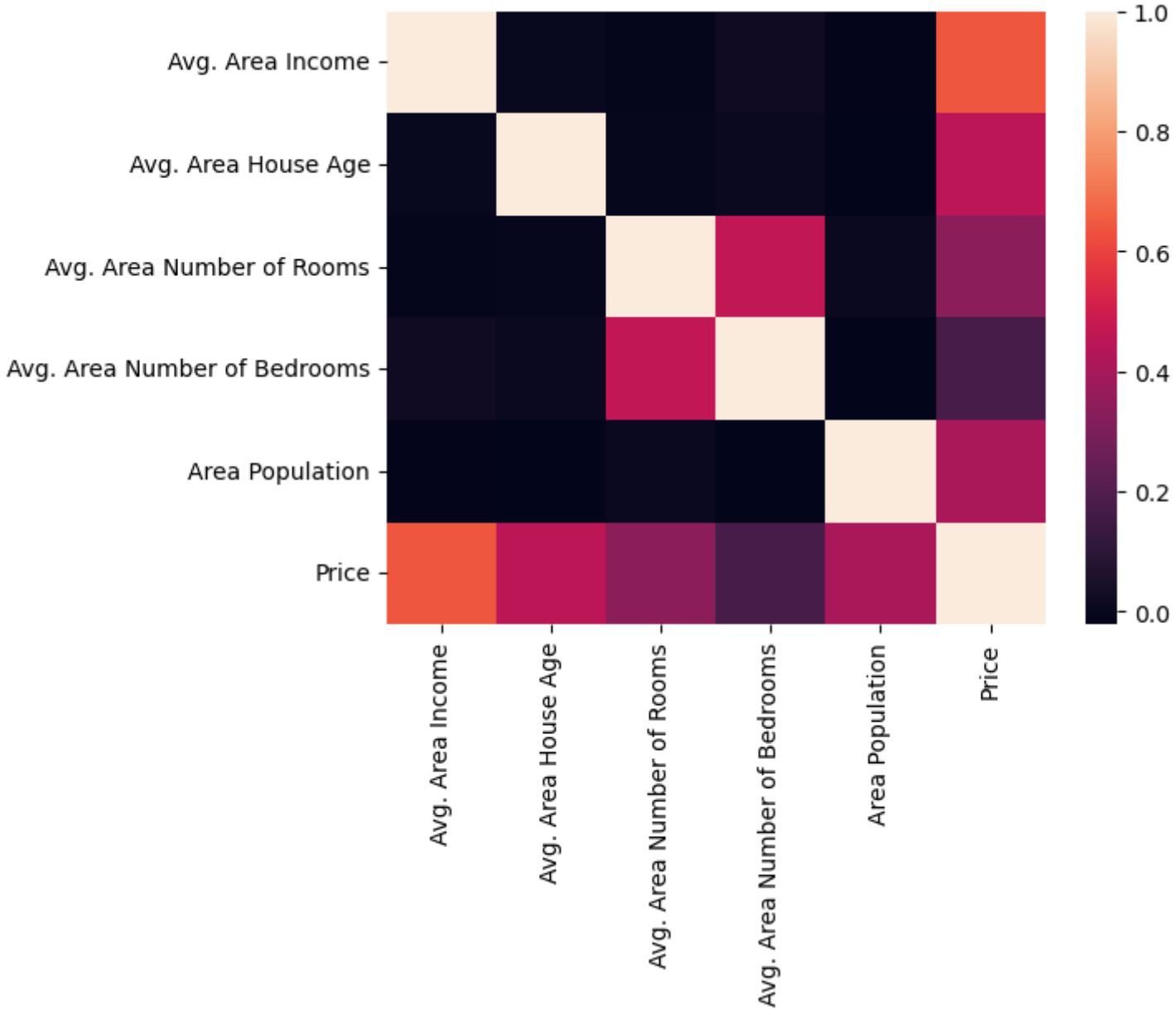
Out[ ]:
```
<seaborn.axisgrid.PairGrid at 0x14fe1e01760>
```


Avg. Area Income
Avg. Area House Age
Avg. Area Number of Rooms
Avg. Area Number of Bedrooms
Area Population
100000
80000
60000
40000
20000
8
6
4
10
8
6
4
6
5
4
3
2
60000
40000
20000



In [ ]:
```
sns.heatmap(df.corr(numeric_only=True))
```

Out[ ]:
```
<Axes: >
```



4. Fitting and Training our model

In [ ]:
```
# defining dependent and independent variables

x=df[df.columns[:-2]]
y=df['Price']
```

In [ ]:
```
# splitting data into train an test data
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70,random_state=101)
```

In [ ]:
```
# fitting data

from sklearn.linear_model import LinearRegression

lmodel=LinearRegression()

lmodel.fit(X=x_train,y=y_train)
```

Out[ ]:
```
▾ LinearRegression
LinearRegression()
```

In [ ]:
```
# training our model

predictions=lmodel.predict(x_test)
```

In [ ]:
```
# Evaluating Model performance

from sklearn import metrics

mae=round(metrics.mean_absolute_error(y_test,predictions),3)
mse=round(metrics.mean_squared_error(y_test,predictions),3)
mrte=round(np.sqrt(mse),3)
r_value=round(metrics.explained_variance_score(y_test,predictions),3)
#print(f"Mean Absolute Error:Mean Squared Error: {}\nMean Square Root Error: {}\nr statistic: {}".format(mae,mse,mrte,r_value))

print(f'Mean Absolute Error: {mae}\nMean Squared Error: {mse}\nMean Square Root Error: {mrte}\nr statistic: {r_value}')
```

```
Mean Absolute Error: 81257.558
Mean Squared Error: 10169125565.897
Mean Square Root Error: 100842.082
r statistic: 0.919
```

Loss functions are used to evaluate the performance of our predictive model by measuring the discrepancy between our predicted values and the actual values. Let's explain the following loss functions of our model:

1. *Mean Absolute Error (MAE):*
    - *The Mean Absolute Error measures the average absolute difference between our predicted values and the actual values.*
    - *In our model, the MAE is 81257.558. On average, our predicted house prices deviate from the actual prices by approximately $81,257.558.*
2. *Mean Squared Error (MSE):*
    - *The Mean Squared Error measures the average squared difference between our predicted values and the actual values.*
    - *In our model, the MSE is 10169125565.897. On average, our predicted house prices deviate from the actual prices by approximately $10,169,125,565.897 (squared).*
3. *Mean Square Root Error (MRTE):*
    - *The Mean Square Root Error is the square root of the Mean Squared Error. It provides a more interpretable measure of error compared to MSE.*
    - *In our model, the MRTE is 100842.082. On average, our predicted house prices deviate from the actual prices by approximately $100,842.082.*
4. *r statistic:*
    - *The r statistic represents the correlation coefficient or the coefficient of determination (R-squared).*
    - *In our model, the r statistic is 0.919. This indicates a strong positive correlation between our predicted house prices and the actual prices. Approximately 91.9% of the variability in the house prices can be explained by our model.*

In [ ]:
```
# formulating coefficient matrix

coeff=lmodel.coef_

intercept=lmodel.intercept_

coeff_matrix=pd.DataFrame(data=coeff,index=df.columns[0:-2],columns=['Coefficients'])
coeff_matrix
```

Out[ ]:

	Coefficients
Avg. Area Income	21.617635
Avg. Area House Age	165221.119872
Avg. Area Number of Rooms	121405.376596
Avg. Area Number of Bedrooms	1318.718783
Area Population	15.225196

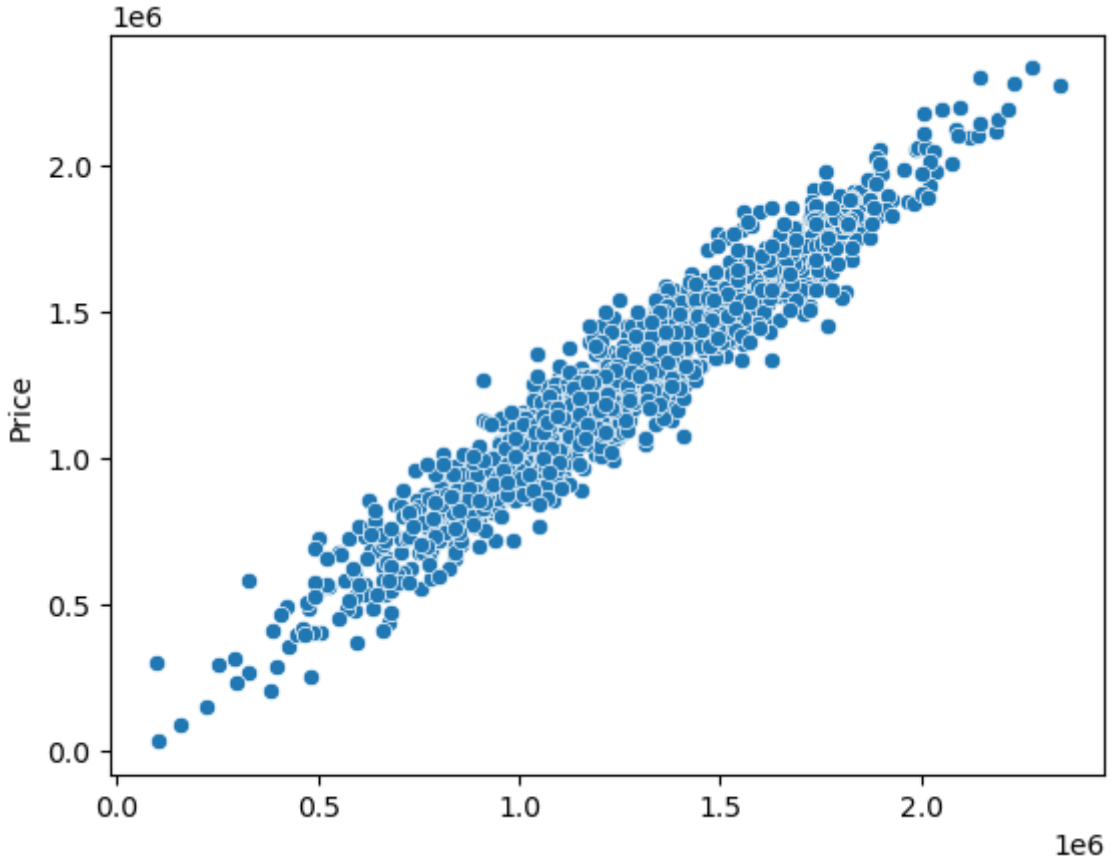
Our Model's coefficients can be explained as follows:

1. *Avg. Area Income:*
    - *The coefficient for Avg. Area Income is 21.617635. This means that, on average, a one-unit increase in the average area income is associated with an increase of $21.617635 in the predicted house price, holding other variables constant.*
2. *Avg. Area House Age:*
    - *The coefficient for Avg. Area House Age is 165221.119872. This indicates that, on average, a one-unit increase in the average area house age is associated with an increase of $165,221.119872 in the predicted house price, holding other variables constant.*
3. *Avg. Area Number of Rooms:*
    - *The coefficient for Avg. Area Number of Rooms is 121405.376596. This suggests that, on average, a one-unit increase in the average number of rooms in houses within the area is associated with an increase of $121,405.376596 in the predicted house price, holding other variables constant.*
4. *Avg. Area Number of Bedrooms:*
    - *The coefficient for Avg. Area Number of Bedrooms is 1318.718783. This indicates that, on average, a one-unit increase in the average number of bedrooms in houses within the area is associated with an increase of $1,318.718783 in the predicted house price, holding other variables constant.*
5. *Area Population:*
    - *The coefficient for Area Population is 15.225196. This suggests that, on average, a one-unit increase in the area population is associated with an increase of $15.225196 in the predicted house price, holding other variables constant.*

These coefficients provide insights into the relationships between the predictor variables and the predicted house prices in our model.

In [ ]:
```
sns.scatterplot(x=predictions,y=y_test)
```

Out[ ]:
```
<Axes: ylabel='Price'>
```

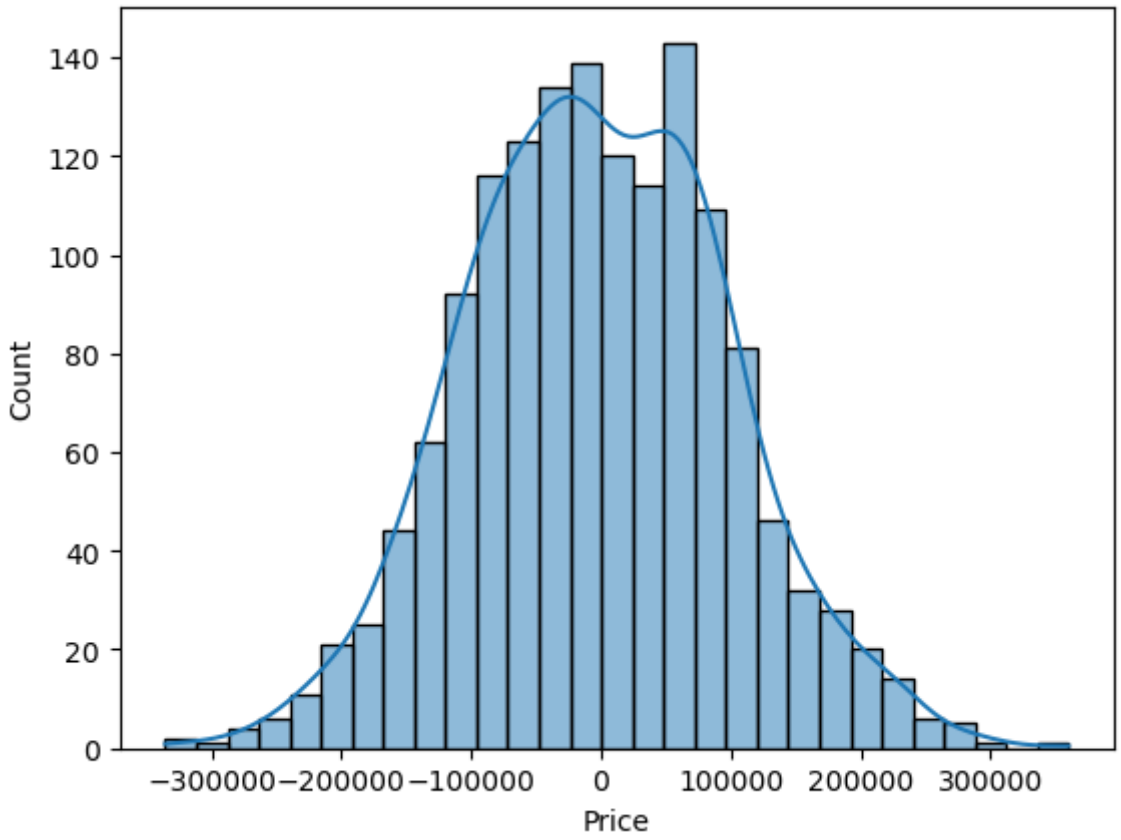


We observe a strong correlation between our predicted and observed Y values, indicating that our model has a high confidence interval. This suggests that our model is effectively capturing the relationship between the predictor variables and the target variable, leading to accurate predictions.

In [ ]:
```
# plotting for distribution of residuals

sns.histplot(y_test-predictions,kde=True)
```

Out[ ]: <Axes: xlabel='Price', ylabel='Count'>



The residuals, which represent the difference between the observed and predicted y values, exhibit a normal distribution. This suggests that our model is well balanced and adequately captures the underlying patterns in the data. The coefficients of our model are as follows:

In [ ]:
```
coeff_matrix
```

Out[ ]:

	Coefficients
Avg. Area Income	21.617635
Avg. Area House Age	165221.119872
Avg. Area Number of Rooms	121405.376596
Avg. Area Number of Bedrooms	1318.718783
Area Population	15.225196

From the coefficient matrix above, it is observed that Avg.Area house age greatly contributes to the price of a house followed by the number of rooms. By assigning these coefficients alphabet characters a, b, c, d, e, we get the following coefficient matrix :

In [ ]:
```
symbols='a b c d e'.split(' ')
coeff_matrix['symbol']=symbols
coeff_matrix
```

Out[ ]:

	Coefficients	symbol
Avg. Area Income	21.617635	a
Avg. Area House Age	165221.119872	b
Avg. Area Number of Rooms	121405.376596	c
Avg. Area Number of Bedrooms	1318.718783	d
Area Population	15.225196	e

Therefore, a linear equation to determine the price of a house in the USA is as follows:

House price = ***21.384444a + 162,975.483408b + 121,802.121132c + 1,934.163056d + 15.189607e***