Predicting House Prices using Machine Learning

Abstract

- Developing an accurate prediction model for housing prices is always needed for socioeconomic development and well-being of citizens
- A diverse set of machine learning algorithms such as XGBoost, CatBoost, Random Forest, Lasso, Voting Regressor, and others, are being employed to predict the housing prices using public available datasets
- a house seller/buyer or a real estate broker can get insight in making better-informed decisions considering the housing price prediction.
- The empirical results illustrate that based on prediction model performance, Coefficient of Determination (R 2), Mean Square Error (MSE), Mean Absolute Error (MAE), and computational time, the XGBoost algorithm performs superior than the other models to predict the housing price.

What is machine learning?

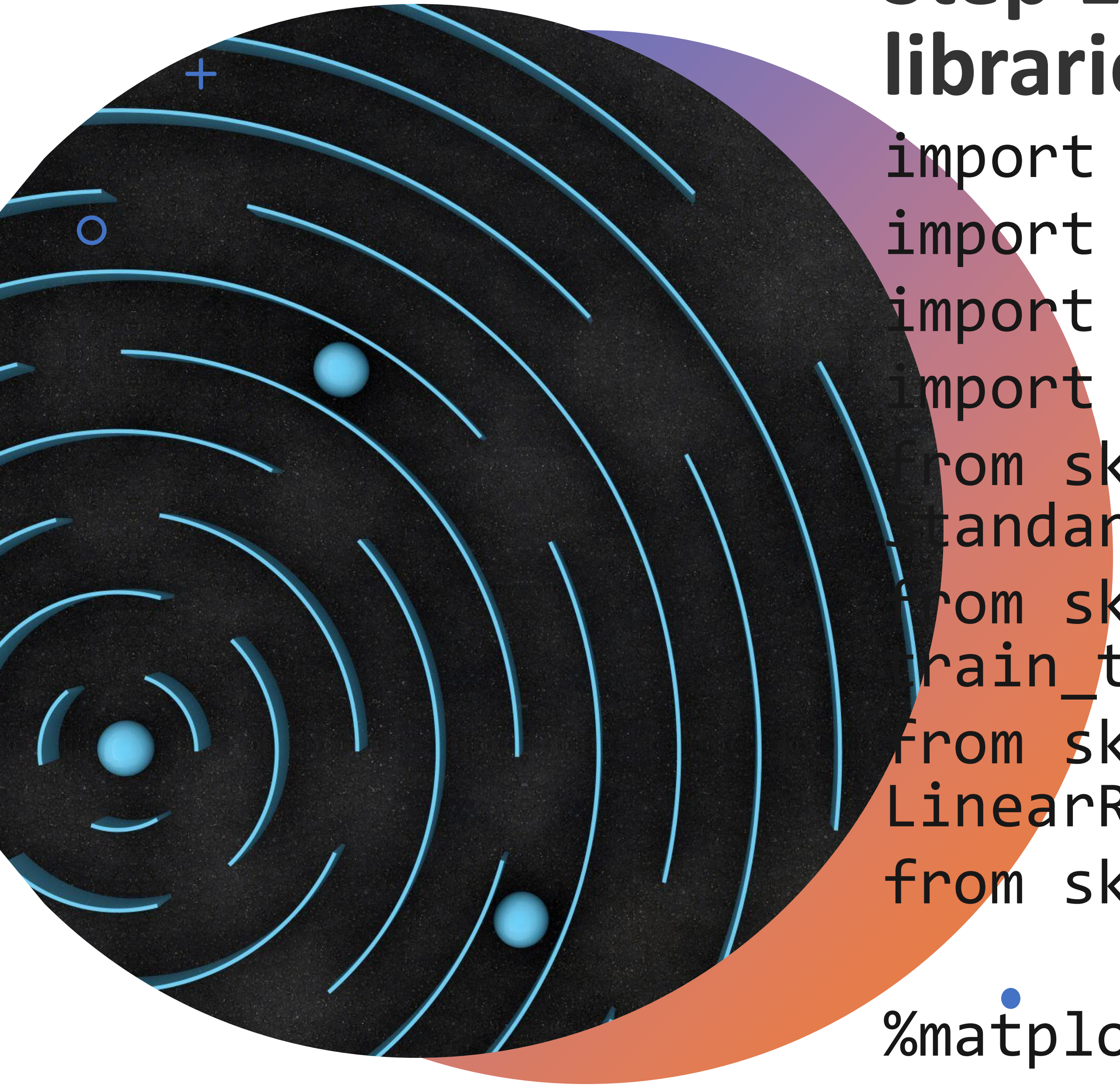
- Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.

Problem definition

- Prices of real estate properties are sophisticatedly linked with our economy.
- Despite this, we do not have accurate measure of house prices based on the vast amount of data available.
- Proper and justify prices of properties can bring in a lot transparency and trust back to real estate industry, which is very important for most consumers especially in India

Data source

- Step 1 – Importing required libraries.
- Step 2 – Reading our input data for House Price Prediction.
- Step 3 – Describing our data.
- Step 4 – Analyzing information from our data.
- Step 5 – Plots to visualize data of House Price Prediction.
- Step 6 – Scaling our data.
- Step 7 – Splitting our data for training and test purposes.
- Step 8 – Training our Linear Regression model for House Price Prediction.
- Step 9 – Let’s visualize our predictions of House Price Prediction.
- Step 10 – Plotting the residuals of our House Price Prediction model.
- Step 11 – Observe the coefficients.

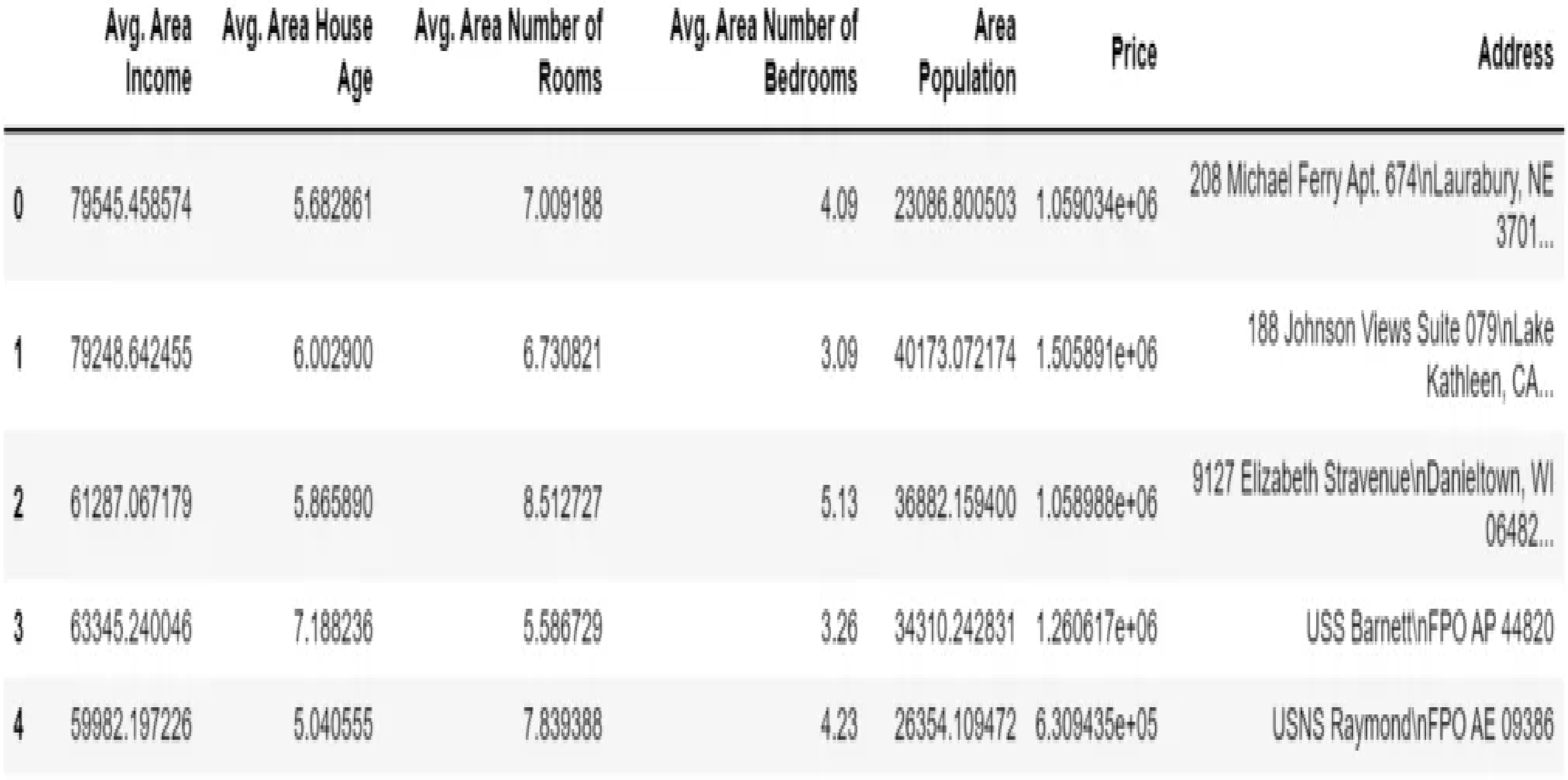


Step 1 – Importing required libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score

%matplotlib inline
```

Step 2 – Reading our input data for House Price Prediction.

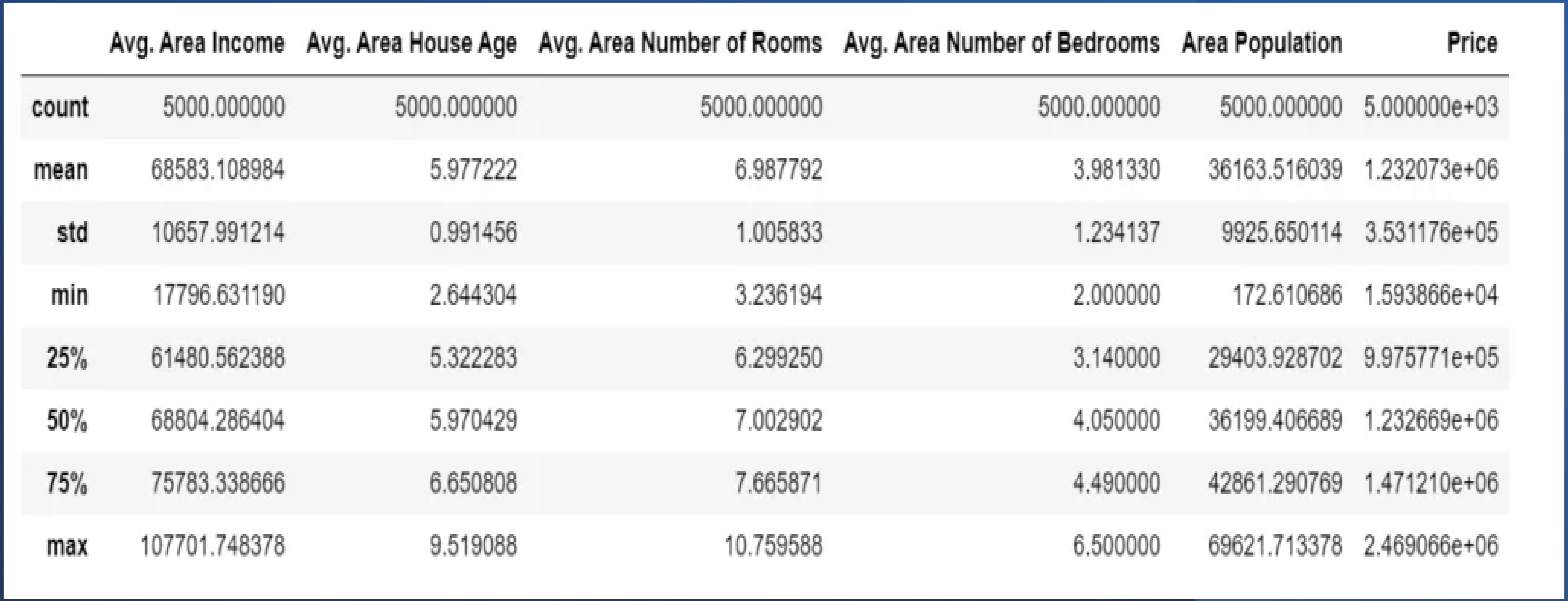
```
customers =
pd.read_csv('USA_Housing.csv')
customers.head()
```



	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

•Step 3 – Describing our data.

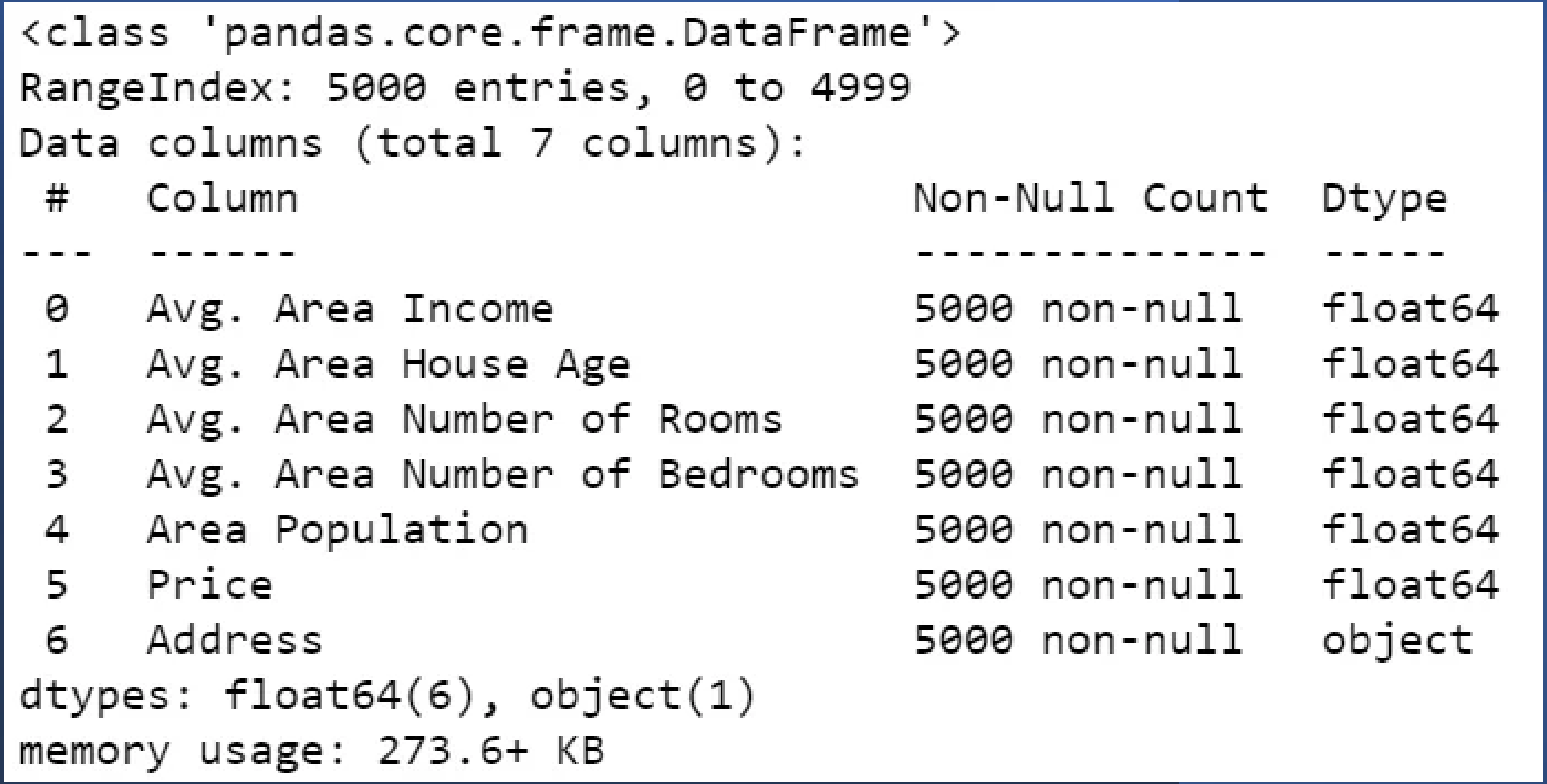
```
customers.describe()
```



	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

Step 4 – Analyzing information from our data.

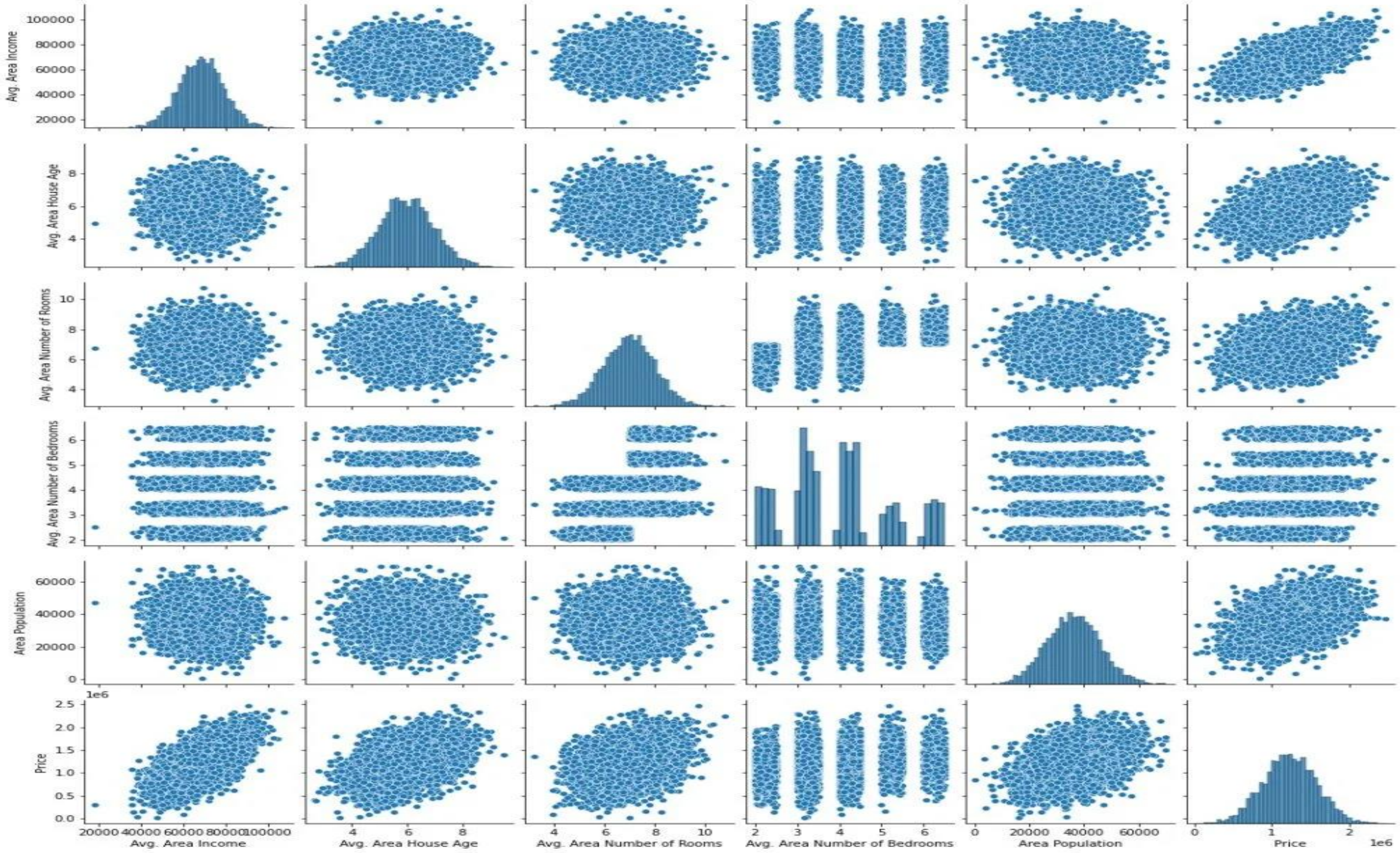
```
customers.info()
```



```
<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
```

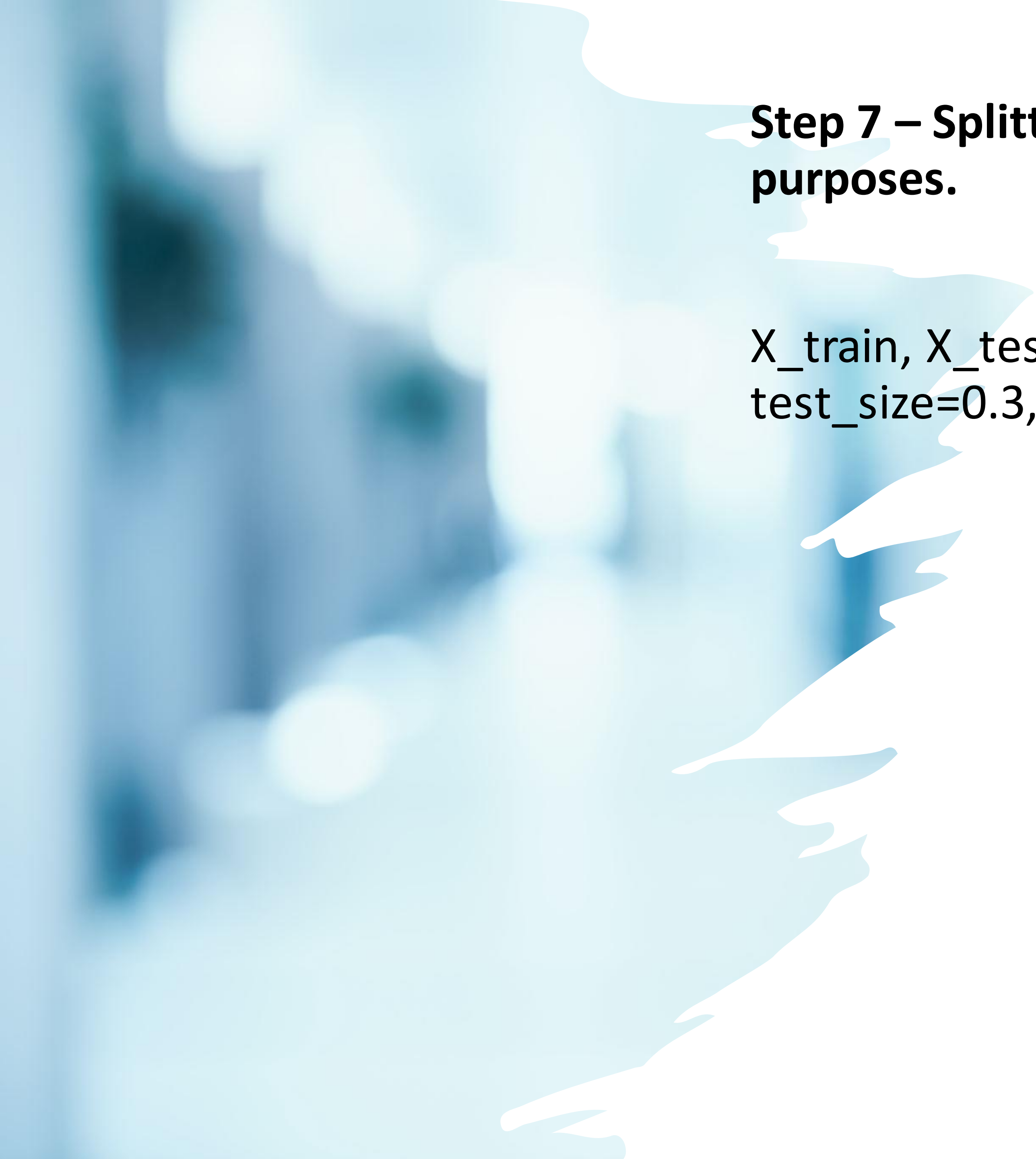
Step 5 – Plots to visualize data of House Price Prediction.

```
sns.pairplot(customers)
```


Avg. Area Income
Avg. Area House Age
Avg. Area Number of Rooms
Avg. Area Number of Bedrooms
Area Population
Price
1e6

Step 6 – Scaling our data.

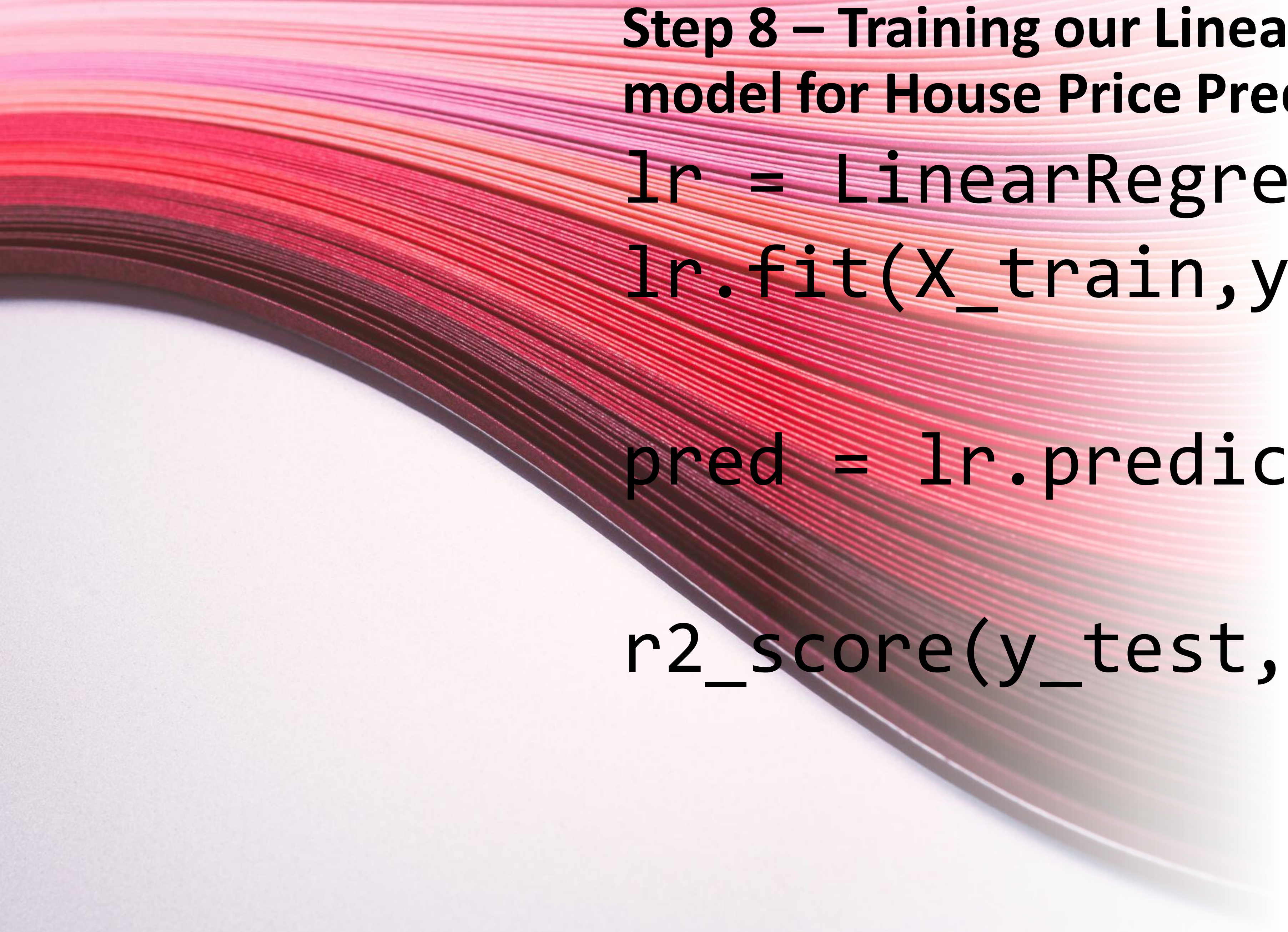
```
scaler = StandardScaler()

X=customers.drop(['Price','Address'],axis=1)
y=customers['Price']

cols = X.columns

X = scaler.fit_transform(X)
```



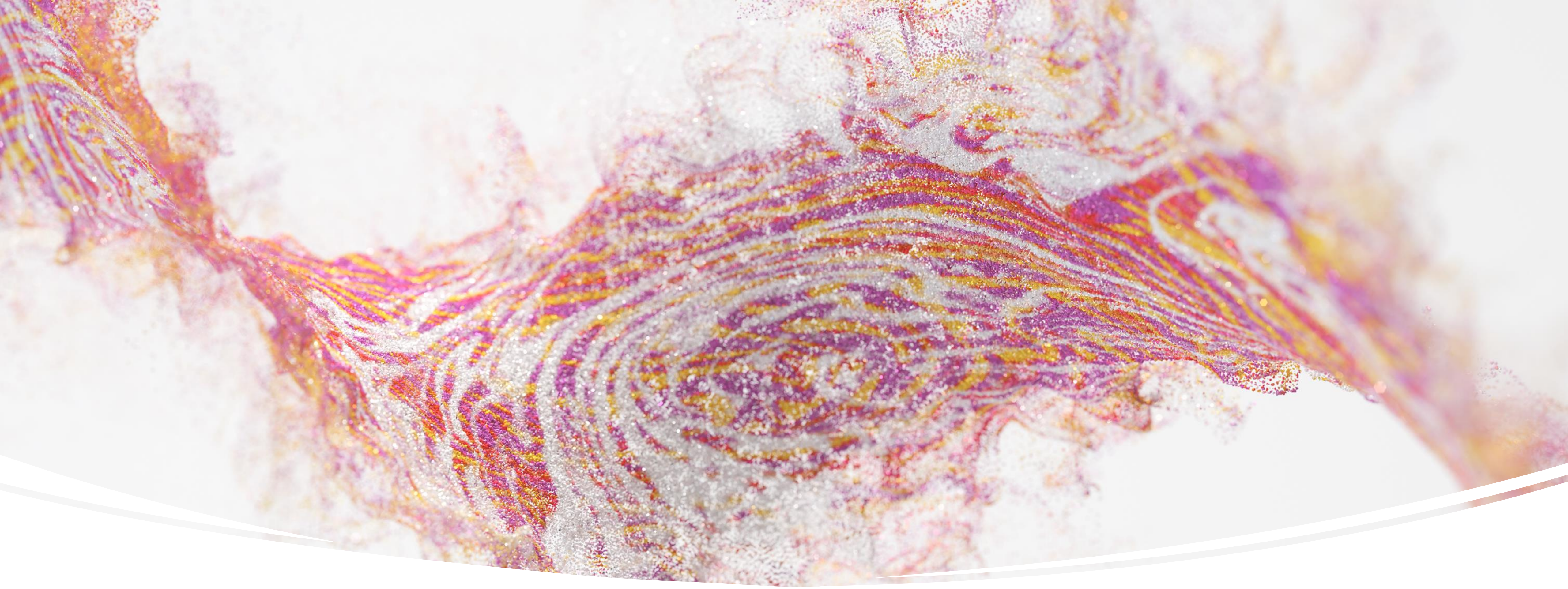
Step 7 – Splitting our data for training and test purposes.

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=101)
```

Step 8 – Training our Linear Regression model for House Price Prediction.

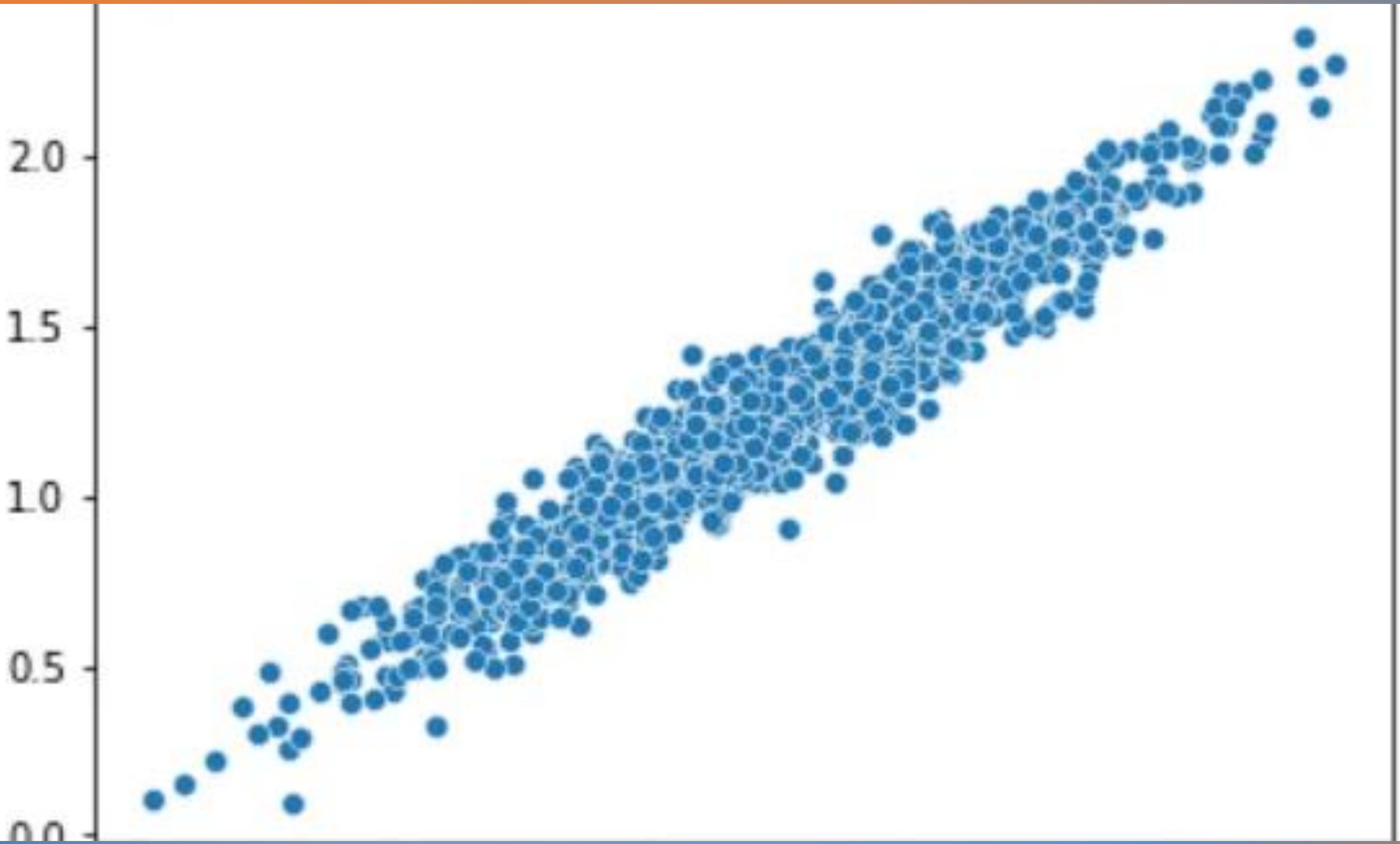


```
lr = LinearRegression()
lr.fit(X_train,y_train)

pred = lr.predict(X_test)

r2_score(y_test,pred)
```



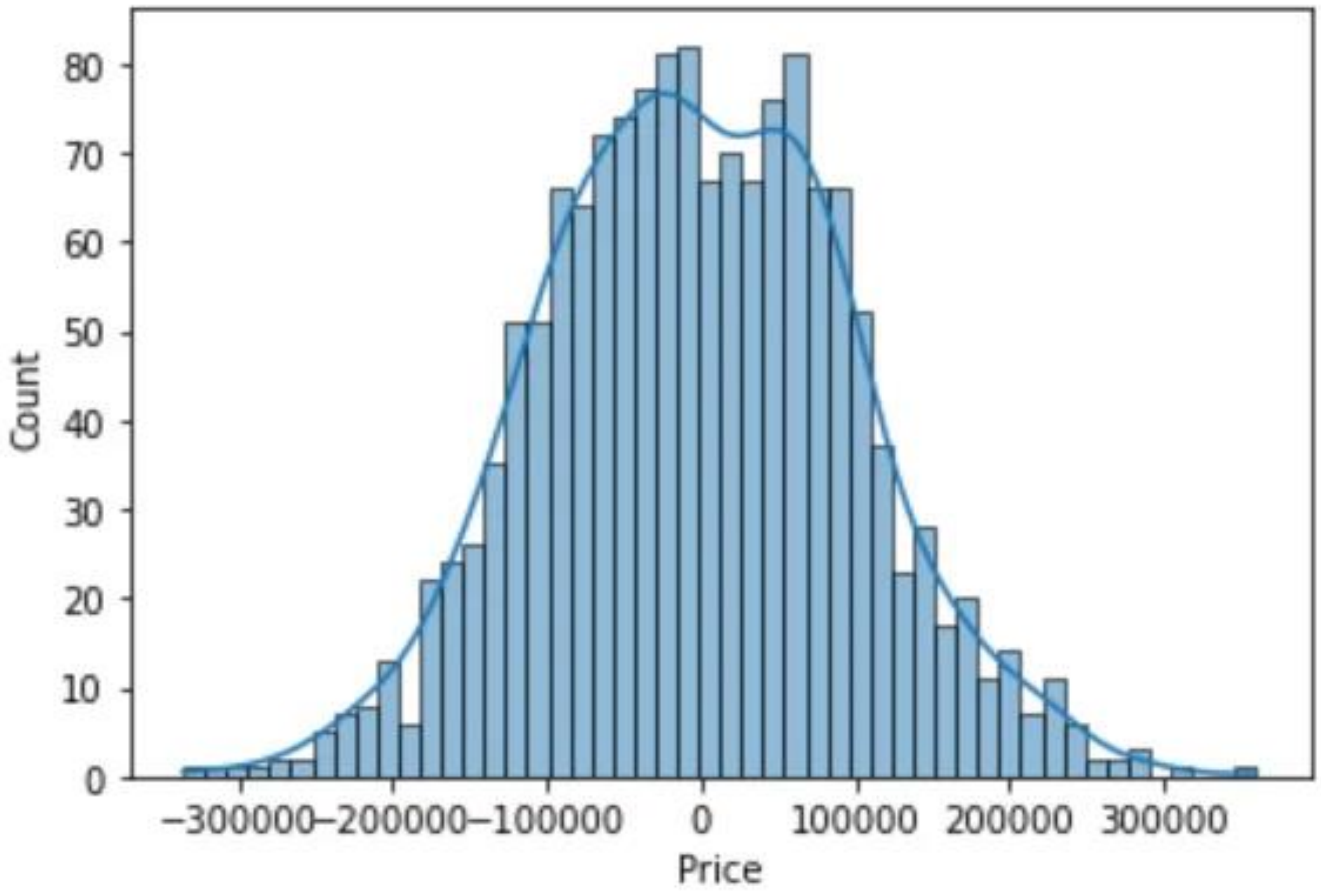
Step 9 – Let’s visualize our predictions of House Price Prediction.

```
sns.scatterplot(x=y_test, y=pred)
```


2.0
1.5
1.0
0.5
0.0

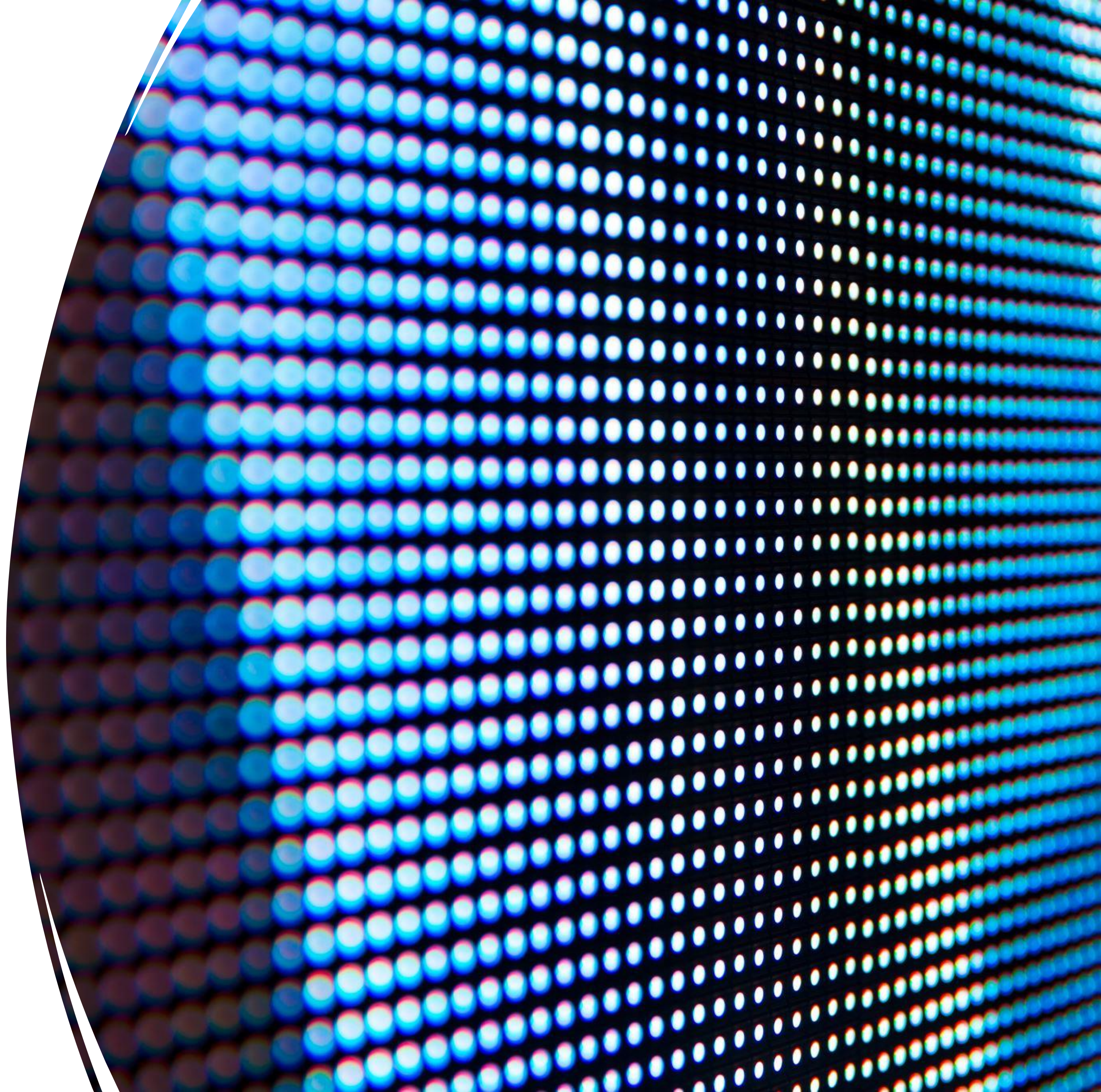
Step 10 – Plotting the residuals of our House Price Prediction model.

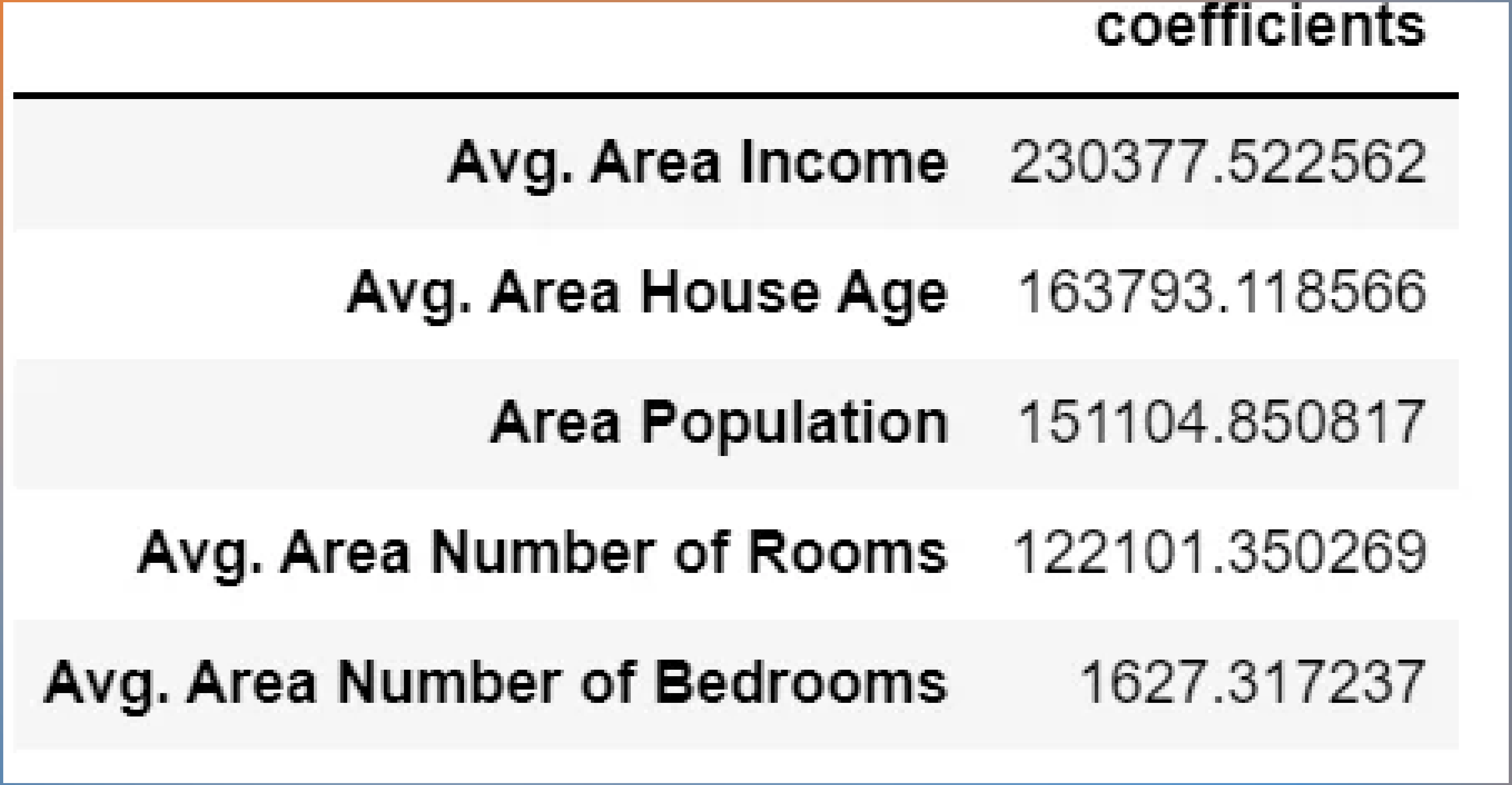
```
sns.histplot((y_test-pred),bins=50,kde=True)
```


Count
Price
0
10
20
30
40
50
60
70
80
−300000
−200000
−100000
0
100000
200000
300000

Step 11 – Observe the coefficients.

```
cdf=pd.DataFrame(lr.coef_, cols, ['coefficients']).sort_values('coefficients',ascending=False)
cdf
```





	coefficients
Avg. Area Income	230377.522562
Avg. Area House Age	163793.118566
Area Population	151104.850817
Avg. Area Number of Rooms	122101.350269
Avg. Area Number of Bedrooms	1627.317237

Data preprocessing for House Price Prediction:

Data Preprocessing:

Data preprocessing is a predominant step in machine learning to yield highly accurate and insightful results. Greater the quality of data, greater is the reliance on the produced results. **Incomplete, noisy, and inconsistent data** are the properties of large real-world datasets. Data preprocessing helps in increasing the quality of data by filling in missing incomplete data, smoothing noise and resolving inconsistencies.

- **Incomplete data** can occur for a number of reasons. Attributes of interest may not always be available, such as customer information for sales transaction data. Relevant data may not be recorded due to a misunderstanding, or because of equipment malfunctions.
- There are many possible reasons for **noisy data** (having incorrect attribute values). The data collection instruments used may be faulty. There may have been human or computer errors occurring at data entry. Errors in data transmission can also occur. Incorrect data may also result from inconsistencies in naming conventions or data codes used, or inconsistent formats for input fields, such as date.

There are a number of data preprocessing techniques available such as,

- **Data Cleaning**
- **Data Integration**
- **Data Transformation**
- **Data Reduction**

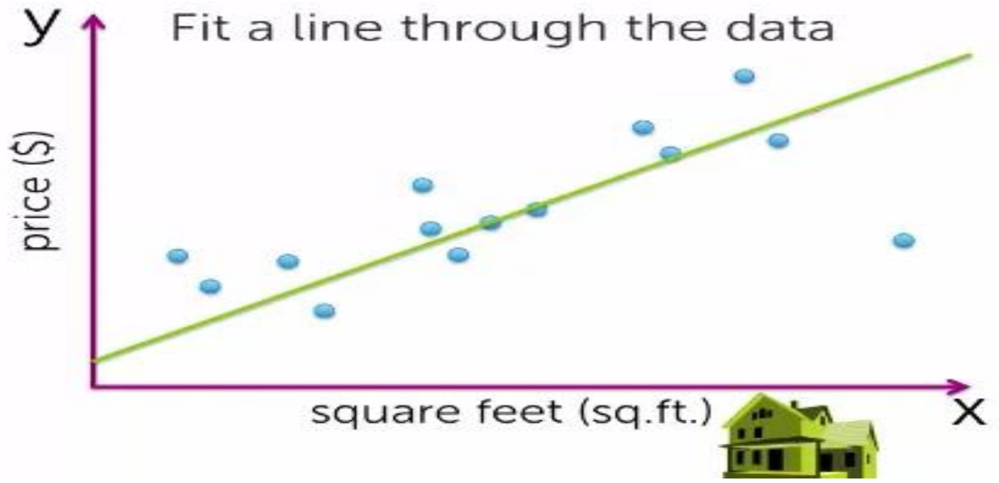
Feature Selection:

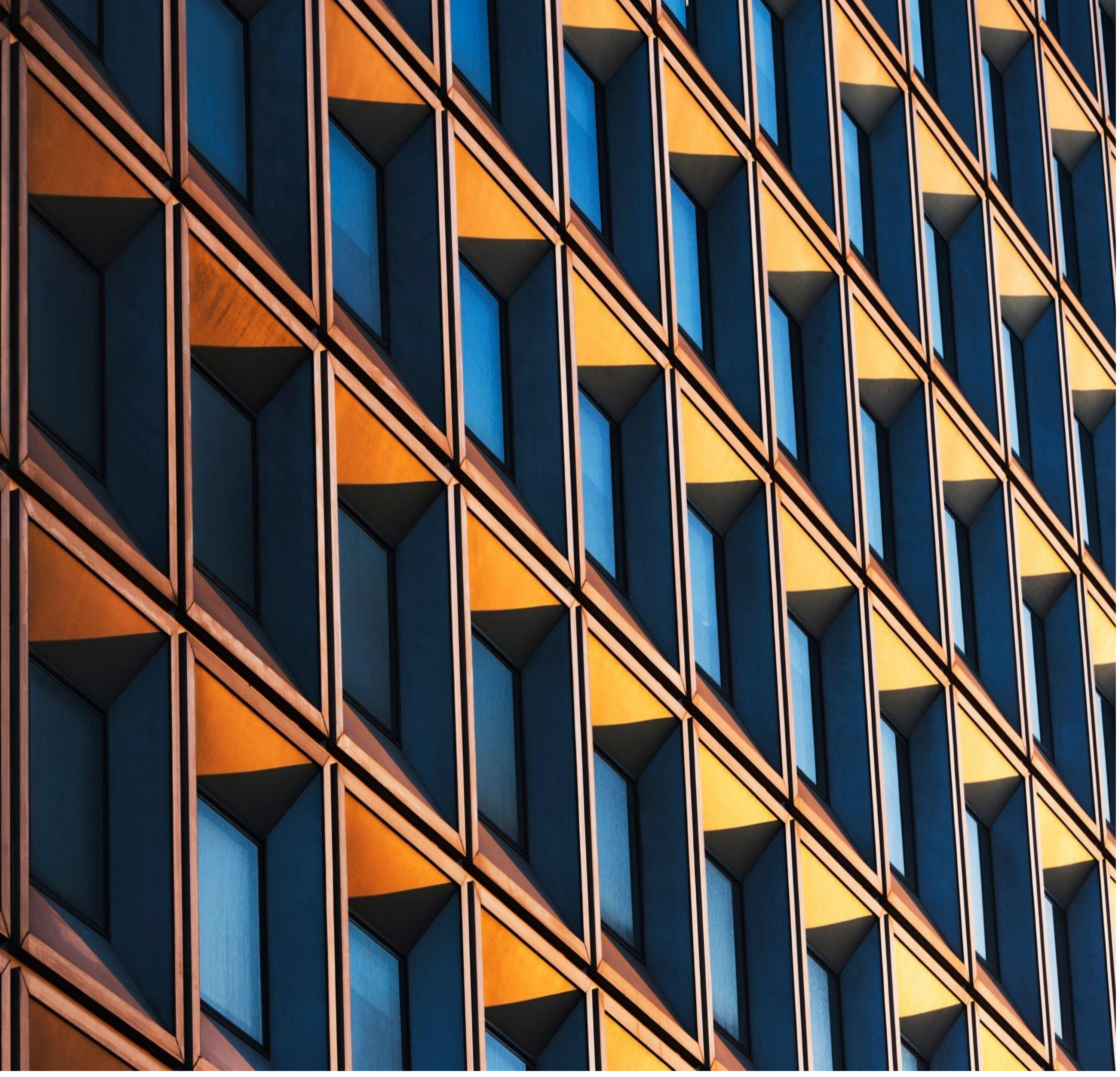
- This notebook tries various feature selection techniques.
- Based in these techniques, select features that contribute most to the output variable.
- Techniques used are
  - Correlation and Mutual Information for Numerical Features
  - SelectFromModel
  - SelectKBest

Linear Regression:

- Linear regression is a widely used learning algorithm that involves fitting a straight line to a dataset.
- Imagine having a dataset of house sizes and prices from a city like Portland, USA.
- By plotting the data points on a graph, with house size on the horizontal axis and price on the vertical axis, you can visualize the relationship between the two variables.
- To predict the price of a house based on its size, a linear regression model is built. The model fits a straight line to the data points, aiming to capture the underlying trend. By extending this line, you can estimate the price for a given house size.
- Linear regression falls under the category of regression models, as it predicts numerical values, such as house prices in dollars.

Use a **linear** regression model





Training a Linear Regression Model

To train the linear regression model, a training set is used. The training set comprises pairs of **input features** *(eg: square feet)* and corresponding **output targets** *(eg: house price)*.

Let:

- *“m” be the number of training examples.*
- *“x” be the input feature representing a specific property of the house, such as its size.*
- *“y” be the output target, representing the actual price of the house.*
- *“i” be the index of a specific training example.*

A specific training example is denoted as (x^i, y^i).

Therefore, the equation can be written as:

Training set: $\{(x^1, y^1), (x^2, y^2), \ldots, (x^m, y^m)\}$

where each pair ***(x^i, y^i)*** *represents a specific house in the dataset, and “m” denotes the number of training examples.*

The Function f(x) and Model Representation:

When training the model, a learning algorithm produces a function (f) that takes **an input feature (x)** and **outputs an estimate or prediction (ŷ)**. In linear regression, the function f(x) is defined as:

where:

•*"w" and "b" are numeric values that determine the line's* ***slope*** *and* ***intercept****, respectively*

A crucial aspect of linear regression is the cost function. The cost function measures the model's performance by quantifying the difference between predicted values and actual targets. This concept is widely applicable in machine learning and plays a vital role in training advanced AI models.

In the next article, we'll delve deeper into the cost function and explore how it contributes to training linear regression models.

In conclusion, linear regression is a valuable tool for predicting house prices based on specific features. Its simplicity and effectiveness make it widely used in the field of machine learning. By fitting a straight line to the data, the linear regression model can estimate the price of a house based on its size. This example demonstrates the concept of supervised learning, where the model is trained using labeled data to predict numeric outputs.

$$f_{w,b}(x^{(i)}) = wx^{(i)} + b$$

. absolute error (MAE), Mean squared error (MSE), Median absolute error (MedAE) and Coefficient of determination (R2). They are all defined below.

Mean absolute error (MAE)

Mean absolute error measures the prediction error by taking the mean of all absolute values of all errors, that is:

MAE = Pn i=0 |yi − yˆi | n

Where n is the number of samples, y are the target values and yˆ are the predicted values. A MAE closer to 0 means that the model predicts with lower error and that the prediction is better the closer the MAE is to 0.

Mean squared error (MSE)

Mean squared error is similar to MAE, but the impact of a term is quadratically proportional to its size. It measures the prediction error by taking the mean of all squared absolute values of all errors, that is:

MSE= Pn i=0(yi − yˆi)

2 n METHODS As for MAE, values close to 0 are better.

Median absolute error (MedAE)

The median absolute error (MedAE) is the median of all absolute differences between the predicted value and the target value. In difference to MAE and MSE, the median absolute error is more robust to outliers by virtue of using the median instead of the mean.

MedAE = median(|y1 − yˆ1|...|yn − yˆn|)

A low MedAE means little error and a good prediction.

Coefficient of determination (R2)

The coefficient of determination, R2 , is the ratio between the explained variance and the total variance. Another, perhaps more intuitive way of understanding it is that it is the fraction of the variance that is predictable by (or explained by) by the model. It is frequently used to evaluate how well a regression model fits the data. The coefficient value ranges from 0 to 1 where 1 is better, meaning perfect determination and 0 meaning no determination. In this study, it is calculated using the scikit-learn r2_score function as follows .

R 2 (y, yˆ) = 1 − Pnsamples−1 i=0 (yi − yˆi) 2 Pnsamples−1 i=0 (yi − y¯) 2

Where yˆi is the predicted value of the sample, y is the corresponding actual value over n samples and y¯ is the arithmetic mean as such: y¯ = 1 n samples n samples X−1 i=0 yi

Conclusion

In conclusion, using machine learning in Python is a powerful tool for predicting house prices. By gathering and cleaning data, visualizing patterns, and training and evaluating our models, we can make informed decisions in the dynamic world of real estate.
By leveraging advanced algorithms and data analysis, we can make accurate predictions and inform decision-making processes. This approach empowers buyers, sellers, and investors to make informed choices in a dynamic and competitive market, ultimately maximizing their opportunities and outcomes.