

LINEAR REGRESSION MODEL

Description: This document demonstrates the process of visualizing a linear regression model using the training data and testing data from the House_price_prediction dataset. The purpose is to understand the relationship between the Area and price by fitting a linear model and visualizing the regression line.

1. Importing Libraries

```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import seaborn as sns
import matplotlib.pyplot as plt
```

Explanation:

This section imports the necessary libraries:

- numpy for numerical operations.
- train_test_split from sklearn for splitting the dataset.
- LinearRegression from sklearn for building the linear regression model.
- seaborn for loading and visualizing datasets.
- matplotlib.pyplot for plotting graphs.

2. Loading the Dataset

```
data_set=pd.read_csv('house_predict.csv')
data_set
```

Explanation:

To load a dataset using pandas, you can use the pd.read_csv() function. This function reads a CSV (Comma-Separated Values) file and creates a DataFrame, which is a 2-dimensional labeled data structure with columns of potentially different types.

Output:



	area	price
0	8450	208500
1	9600	181500
2	11250	223500
3	9550	140000
4	14260	250000
...
1455	7917	175000
1456	13175	210000
1457	9042	266500
1458	9717	142125
1459	9937	147500

1460 rows x 2 columns

4. Selecting Relevant Columns

```
data_set=data_set[['area','price']]
```

✓ 0.0s

Explanation:

This filters the dataset to include only the columns "area" and "price" which are the features used for this analysis.

5. Defining Features and Target Variables

```
x=data_set[['area']]
y=data_set[['price']]
✓ 0.0s
```

Explanation:

Here, x represents the feature variable (area), and y represents the target variable (price).

Output:

area	
0	8450
1	9600
2	11250
3	9550
4	14260
...	...
1455	7917
1456	13175
1457	9042
1458	9717
1459	9937
1460 rows × 1 columns	

price	
0	208500
1	181500
2	223500
3	140000
4	250000
...	...
1455	175000
1456	210000
1457	266500
1458	142125
1459	147500
1460 rows × 1 columns	

6. Plotting the Data Points

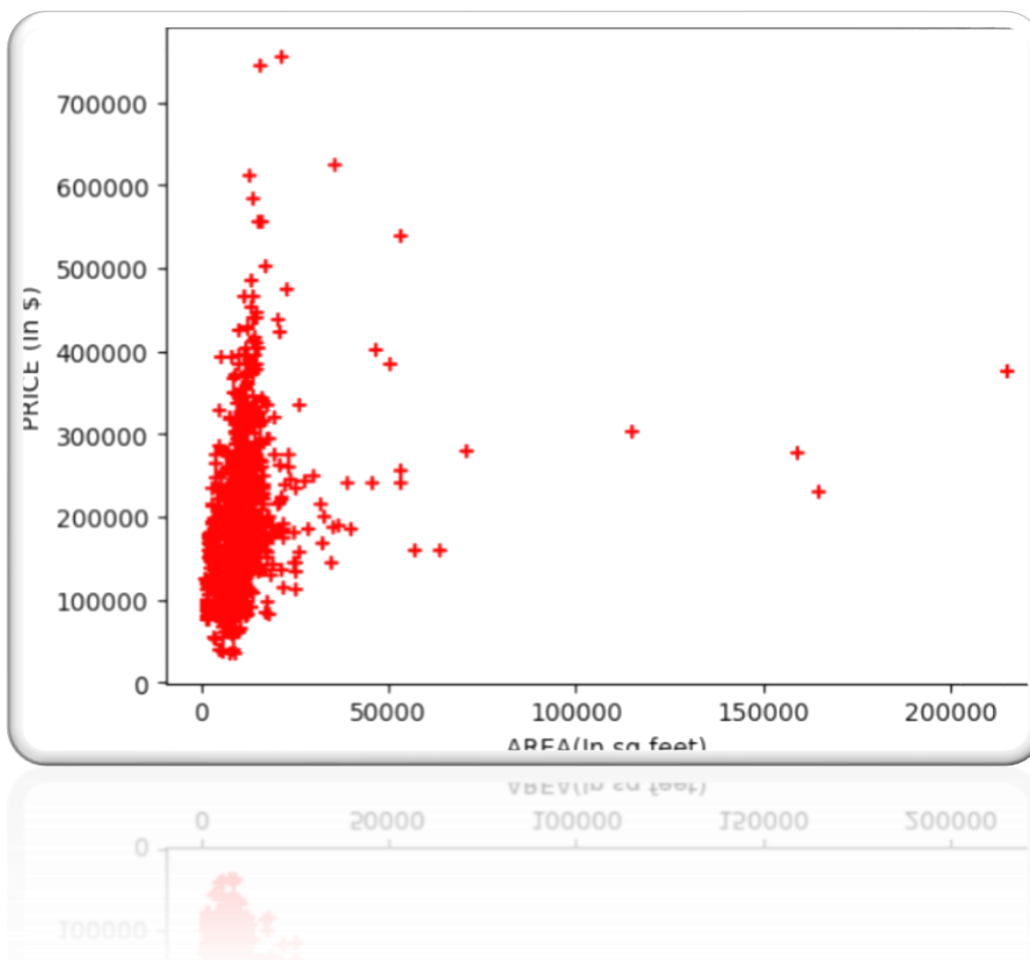
```
plt.scatter(x,y,color='red',marker='+')  
plt.xlabel('AREA(In sq feet)')  
plt.ylabel('PRICE (In $)')
```

✓ 0.2s

Explanation:

This section creates a scatter plot to visualize the relationship between area and price. `plt.scatter()` generates the plot, while `plt.xlabel()` and `plt.ylabel()` label the axes.

Output:



7. Splitting the Dataset

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

✓ 0.0s

Explanation:

The dataset is split into training and test sets. 80% of the data is used for training, and 20% is used for testing



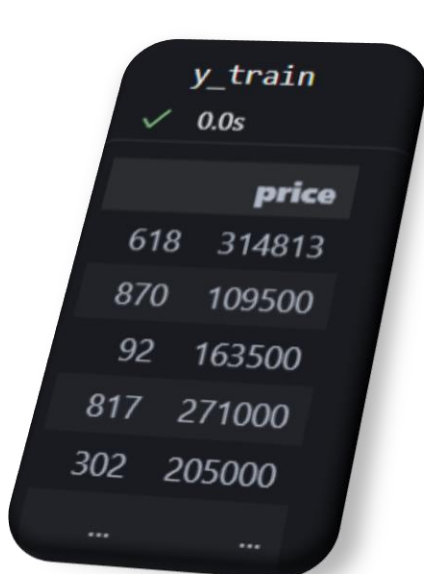
A dark-themed rectangular display showing the 'x_train' dataset. At the top, it says 'x_train' in yellow, followed by a green checkmark and '0.0s'. Below this is a header 'area' in white. The data is presented as a table with two columns. The first column contains the values 618, 870, 92, 817, and 302. The second column contains the values 11694, 6600, 13360, 13265, and 13704. At the bottom, there are three dots in each column.

area	
618	11694
870	6600
92	13360
817	13265
302	13704
...	...



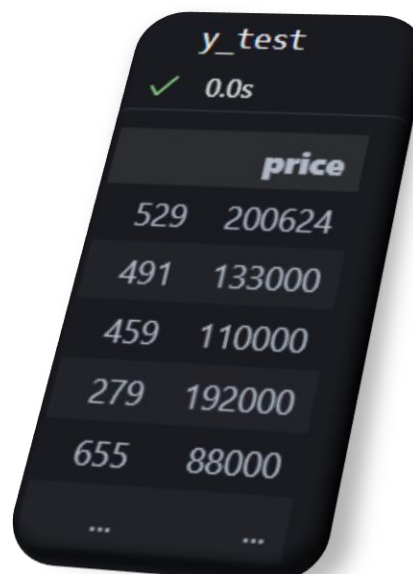
A dark-themed rectangular display showing the 'x_test' dataset. At the top, it says 'x_test' in yellow, followed by a green checkmark and '0.0s'. Below this is a header 'area' in white. The data is presented as a table with two columns. The first column contains the values 529, 491, 459, 279, and 655. The second column contains the values 32668, 9490, 7015, 10005, and 1680. At the bottom, there are three dots in each column.

area	
529	32668
491	9490
459	7015
279	10005
655	1680
...	...



A dark-themed rectangular display showing the 'y_train' dataset. At the top, it says 'y_train' in yellow, followed by a green checkmark and '0.0s'. Below this is a header 'price' in white. The data is presented as a table with two columns. The first column contains the values 618, 870, 92, 817, and 302. The second column contains the values 314813, 109500, 163500, 271000, and 205000. At the bottom, there are three dots in each column.

price	
618	314813
870	109500
92	163500
817	271000
302	205000
...	...



A dark-themed rectangular display showing the 'y_test' dataset. At the top, it says 'y_test' in yellow, followed by a green checkmark and '0.0s'. Below this is a header 'price' in white. The data is presented as a table with two columns. The first column contains the values 529, 491, 459, 279, and 655. The second column contains the values 200624, 133000, 110000, 192000, and 88000. At the bottom, there are three dots in each column.

price	
529	200624
491	133000
459	110000
279	192000
655	88000
...	...

8. Reshaping Data

```
x_train=np.array(x_train).reshape(len(x_train),1)
x_test=np.array(x_test).reshape(len(x_test),1)
y_train=np.array(y_train).reshape(len(y_train),1)
y_test=np.array(y_test).reshape(len(y_test),1)
```

Explanation:

The data is reshaped to ensure it meets the input requirements for the linear regression model.

9. Creating and Training the Linear Regression Model

```
model=LinearRegression()
model.fit(x_train,y_train)
```

✓ 0.0s

LinearRegression ⓘ ?

LinearRegression()

10. Extracting Model Parameters

```
m=model.coef_
c=model.intercept_
```

✓ 0.0s

Explanation:

The model coefficients (slope) and intercept are extracted from the trained model.

11. Making Predictions

```
y_pred=m*x_train+c  
y_pred
```

✓ 0.0s

```
y_pred=m*x_test+c  
y_pred
```

✓ 0.0s

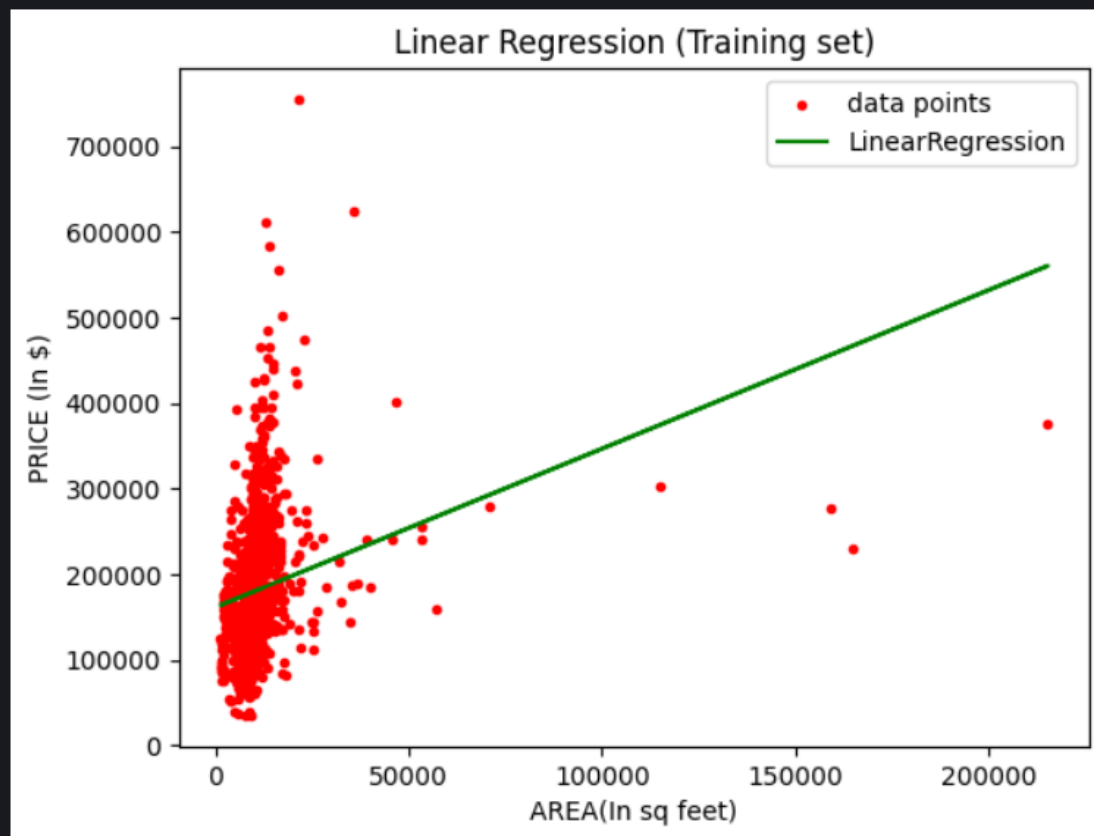
Explanation:

Predicted values for the training and testing are calculated using the linear model.

12. Plotting Predictions(Training set)

```
plt.scatter(x_train,y_train,color='r',marker='.',label='data points')  
plt.plot(x_train,y_pred,color='g',label='LinearRegression')  
plt.xlabel('AREA(In sq feet)')  
plt.ylabel('PRICE (In $)')  
plt.title('Linear Regression (Training set)')  
plt.legend()  
plt.show()
```

✓ 0.1s



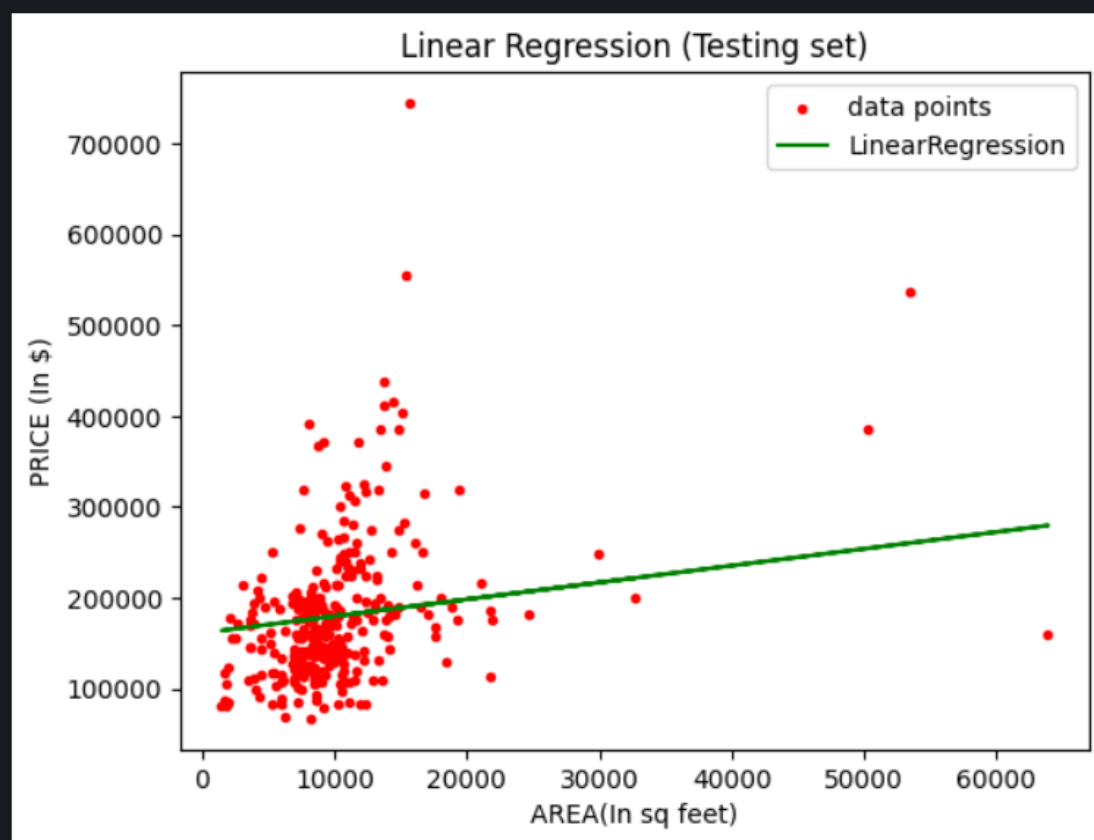
Explanation:

Finally, a scatter plot of the training data is created along with a regression line to visualize the linear relationship.

13. Plotting Predictions(Testing set)

```
plt.scatter(x_test,y_test,color='r',marker='.',label='data points')
plt.plot(x_test,y_pred,color='g',label='LinearRegression')
plt.xlabel('AREA(In sq feet)')
plt.ylabel('PRICE (In $)')
plt.title('Linear Regression (Testing set)')
plt.legend()
plt.show()
```

✓ 0.1s



Explanation:

Finally, a scatter plot of the testing data is created along with a regression line to visualize the linear relationship.
