Assignment 3

SANSKAR DAGAR

sanskar.dagar2020@vitstudent.ac.in

VIT Applied Data Science 2023

QUESTION:

Problem Statement: House Price Prediction

Description:- House price prediction is a common problem in the real estate industry and involves predicting the selling price of a house based on various features and attributes. The problem is typically approached as a regression problem, where the target variable is the price

of the house, and the features are various attributes of the house

The features used in house price prediction can include both quantitative and categorical variables, such as the number of bedrooms, house area, bedrooms, furnished, nearness to main road, and various amenities such as a garage and other factors that may influence the value of the property.

Accurate predictions can help agents and appraisers price homes correctly, while homeowners can use the predictions to set a reasonable asking price for their properties. Accurate house price prediction can also be useful for buyers who are looking to make informed decisions about purchasing a property and obtaining a fair price for their investment.

Attribute Information:

Name - Description

1- Price-Prices of the houses

2- Area- Area of the houses

3- Bedrooms- No of house bedrooms

4- Bathrooms- No of bathrooms

5- Stories- No of house stories

6- Main Road- Weather connected to Main road

7- Guestroom-Weather has a guest room

8- Basement-Weather has a basement

9- Hot water heating- Weather has a hot water heater

10-Airconditioning-Weather has a air conditioner

11-Parking- No of house parking

12-Furnishing Status-Furnishing status of house

Building a Regression Model

1. Download the dataset: Dataset

2. Load the dataset into the tool.

3. Perform Below Visualizations.

 Univariate Analysis

 Bi-Variate Analysis

 Multi-Variate Analysis

4. Perform descriptive statistics on the dataset.

5. Check for Missing values and deal with them.

6. Find the outliers and replace them outliers

7. Check for Categorical columns and perform encoding.

8. Split the data

into dependent and independent variables.

9. Scale the independent

variables

10. Split the data into training and testing

11. Build the Model

12. Train the Model

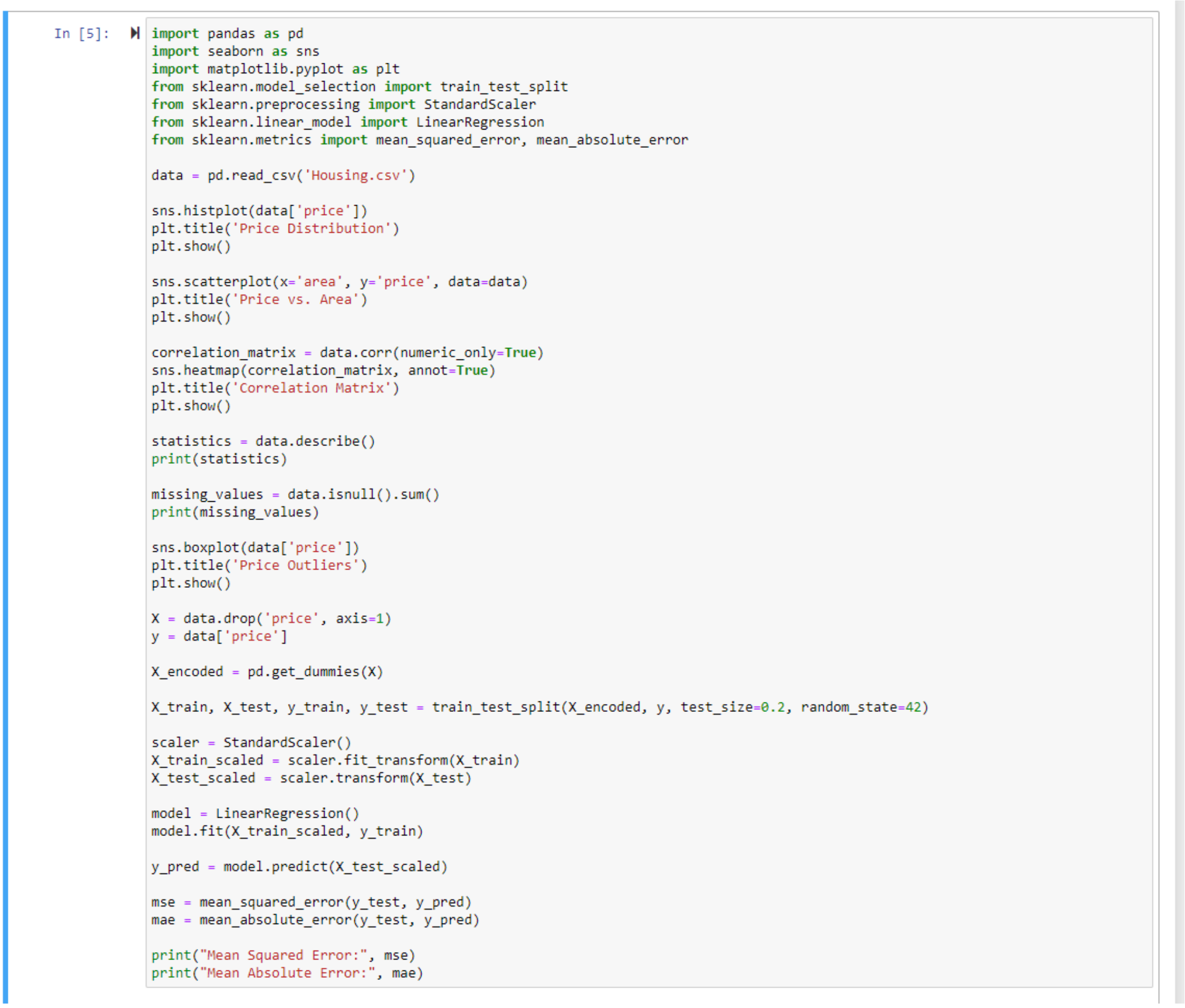
13. Test the Model

14. Measure the performance using Metrics.

CODE:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error

data = pd.read_csv('Housing.csv')

sns.histplot(data['price'])
plt.title('Price Distribution')
plt.show()

sns.scatterplot(x='area', y='price', data=data)
plt.title('Price vs. Area')
plt.show()

correlation_matrix = data.corr(numeric_only=True)
sns.heatmap(correlation_matrix, annot=True)
plt.title('Correlation Matrix')
plt.show()

statistics = data.describe()
print(statistics)

missing_values = data.isnull().sum()
print(missing_values)
```

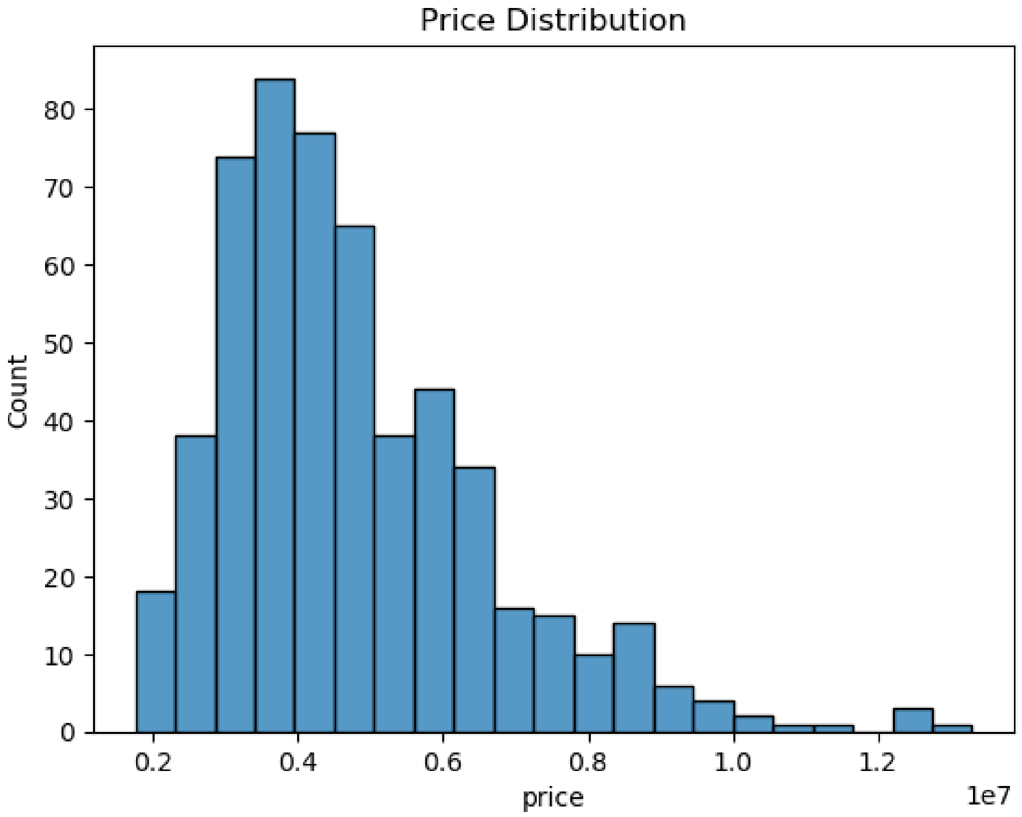
```
sns.boxplot(data['price'])
plt.title('Price Outliers')
plt.show()

X = data.drop('price', axis=1)
y = data['price']

X_encoded = pd.get_dummies(X)

X_train, X_test, y_train, y_test = train_test_split(X_encoded, y, test_size=0.2,
random_state=42)

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

model = LinearRegression()
model.fit(X_train_scaled, y_train)

y_pred = model.predict(X_test_scaled)

mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)

print("Mean Squared Error:", mse)
print("Mean Absolute Error:", mae)
```

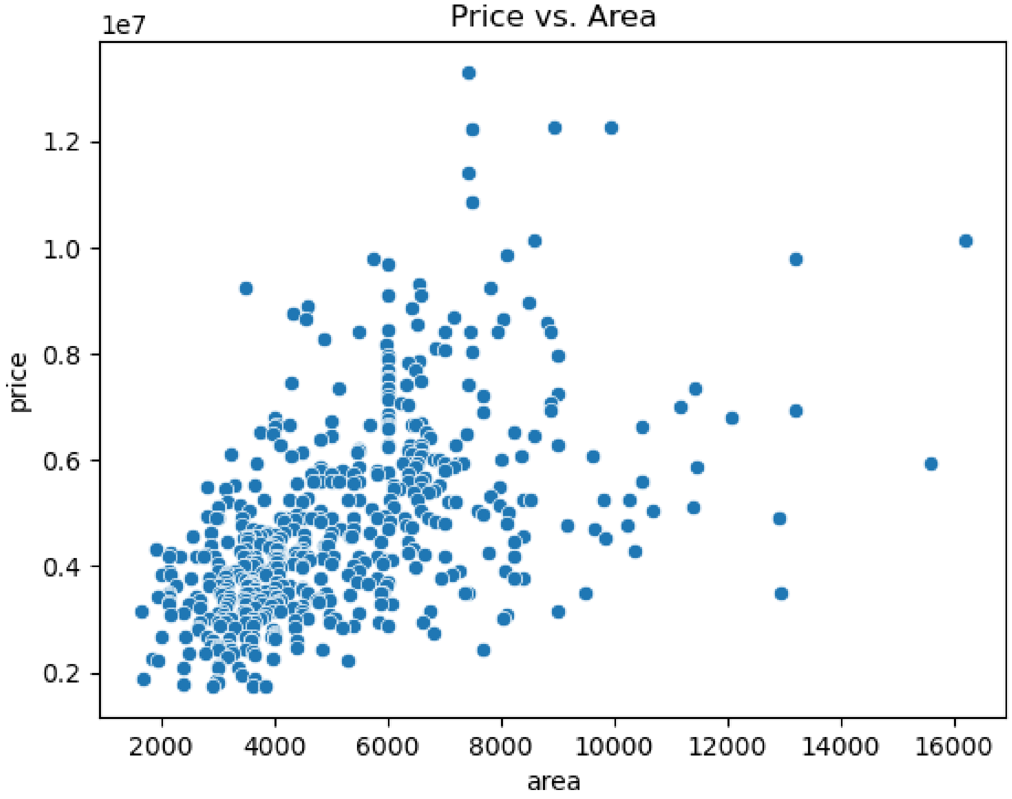


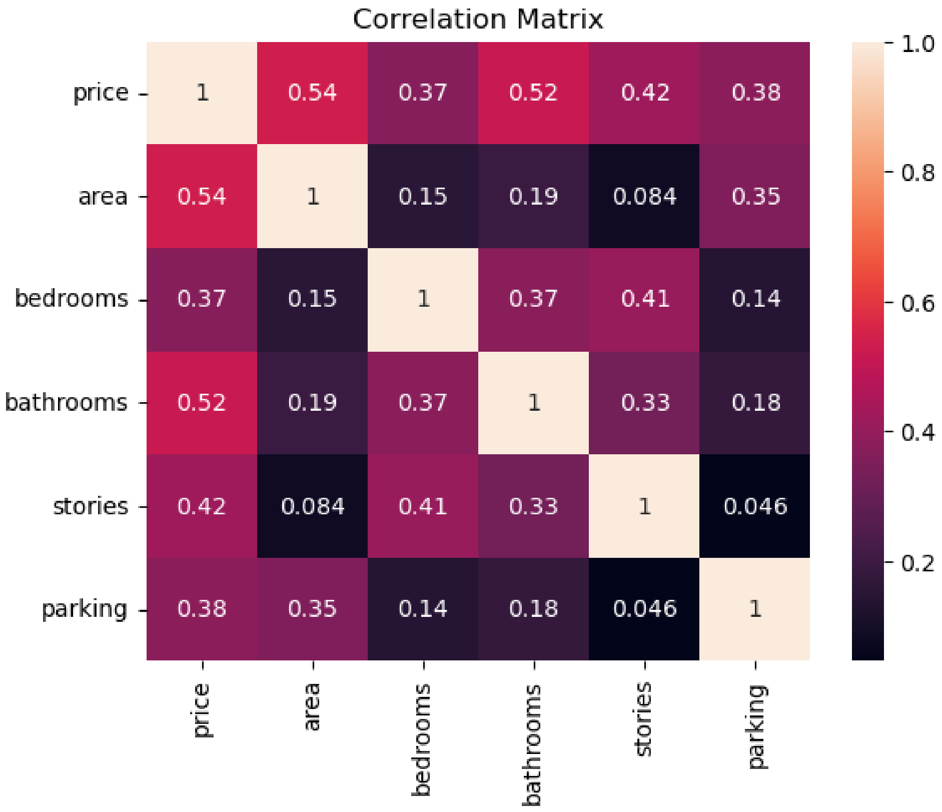
In [5]:

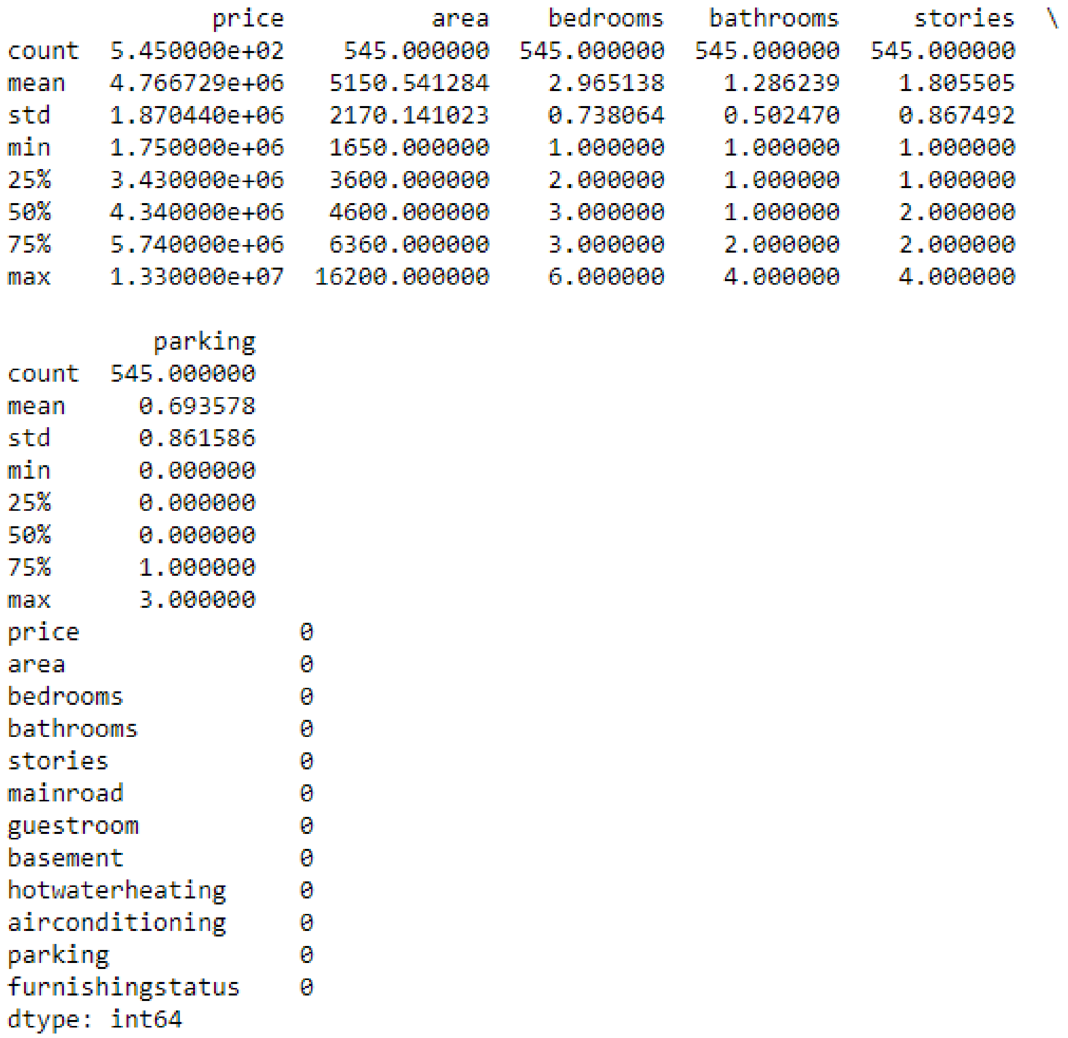
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error

data = pd.read_csv('Housing.csv')

sns.histplot(data['price'])
plt.title('Price Distribution')
plt.show()

sns.scatterplot(x='area', y='price', data=data)
plt.title('Price vs. Area')
plt.show()

correlation_matrix = data.corr(numeric_only=True)
sns.heatmap(correlation_matrix, annot=True)
plt.title('Correlation Matrix')
plt.show()

statistics = data.describe()
print(statistics)

missing_values = data.isnull().sum()
print(missing_values)

sns.boxplot(data['price'])
plt.title('Price Outliers')
plt.show()

X = data.drop('price', axis=1)
y = data['price']

X_encoded = pd.get_dummies(X)

X_train, X_test, y_train, y_test = train_test_split(X_encoded, y, test_size=0.2, random_state=42)

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

model = LinearRegression()
model.fit(X_train_scaled, y_train)

y_pred = model.predict(X_test_scaled)

mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)

print("Mean Squared Error:", mse)
print("Mean Absolute Error:", mae)
```

OUTPUT:

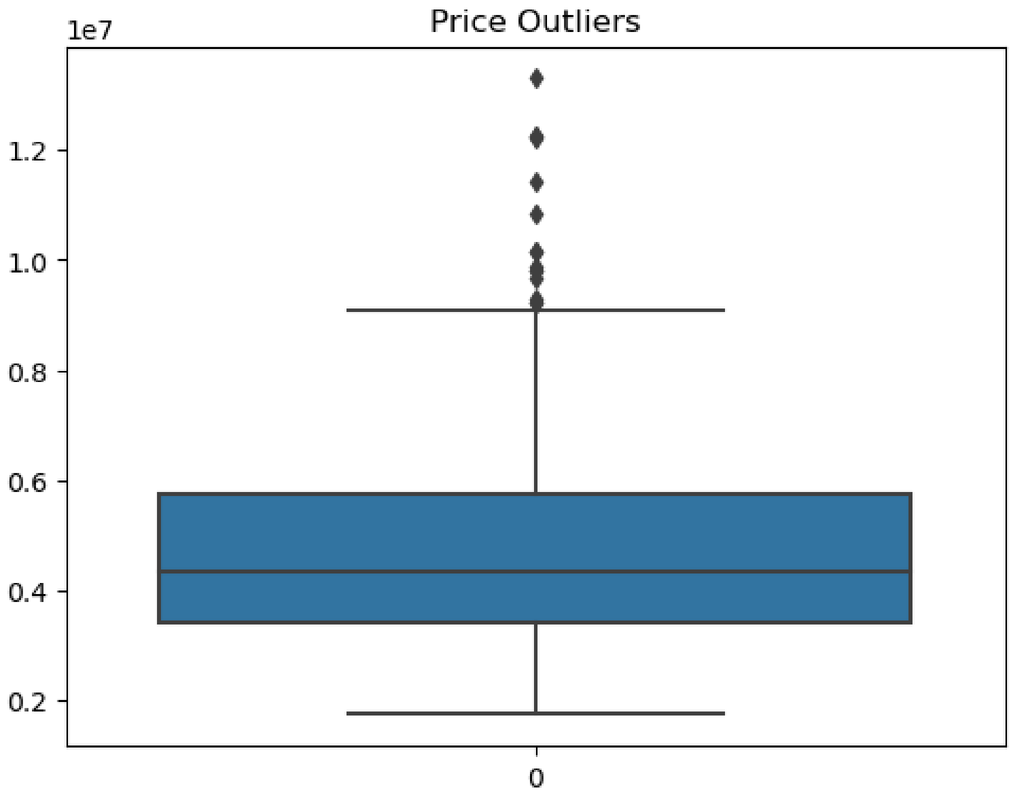

Price Distribution
Count
0
10
20
30
40
50
60
70
80
0.2
0.4
0.6
0.8
1.0
1.2
price
1e7


Price vs. Area
1e7
price
0.2
0.4
0.6
0.8
1.0
1.2
2000
4000
6000
8000
10000
12000
14000
16000
area


Correlation Matrix
price 1 0.54 0.37 0.52 0.42 0.38
area 0.54 1 0.15 0.19 0.084 0.35
bedrooms 0.37 0.15 1 0.37 0.41 0.14
bathrooms 0.52 0.19 0.37 1 0.33 0.18
stories 0.42 0.084 0.41 0.33 1 0.046
parking 0.38 0.35 0.14 0.18 0.046 1
price
area
bedrooms
bathrooms
stories
parking
1.0
0.8
0.6
0.4
0.2



```
              price          area    bedrooms   bathrooms     stories  \
count  5.450000e+02    545.000000  545.000000  545.000000  545.000000
mean   4.766729e+06   5150.541284    2.965138    1.286239    1.805505
std    1.870440e+06   2170.141023    0.738064    0.502470    0.867492
min    1.750000e+06   1650.000000    1.000000    1.000000    1.000000
25%    3.430000e+06   3600.000000    2.000000    1.000000    1.000000
50%    4.340000e+06   4600.000000    3.000000    1.000000    2.000000
75%    5.740000e+06   6360.000000    3.000000    2.000000    2.000000
max    1.330000e+07  16200.000000    6.000000    4.000000    4.000000

          parking
count  545.000000
mean     0.693578
std      0.861586
min      0.000000
25%      0.000000
50%      0.000000
75%      1.000000
max      3.000000
price               0
area                0
bedrooms            0
bathrooms           0
stories             0
mainroad            0
guestroom           0
basement            0
hotwaterheating     0
airconditioning     0
parking             0
furnishingstatus    0
dtype: int64
```



```
Mean Squared Error: 1837637189871.7092
Mean Absolute Error: 988116.1632405716
```