ASSIGNMENT 3

BHAWANI PRIYA BODDI (20BCD7008)

In [16]:

```
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
```

2. Load the dataset into the tool.

In [19]:

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

In [20]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 545 entries, 0 to 544
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   price             545 non-null    int64 
 1   area              545 non-null    int64 
 2   bedrooms          545 non-null    int64 
 3   bathrooms         545 non-null    int64 
 4   stories           545 non-null    int64 
 5   mainroad          545 non-null    object
 6   guestroom         545 non-null    object
 7   basement          545 non-null    object
 8   hotwaterheating   545 non-null    object
 9   airconditioning   545 non-null    object
 10  parking           545 non-null    int64 
 11  furnishingstatus  545 non-null    object
dtypes: int64(6), object(6)
memory usage: 51.2+ KB
```

In [21]:

```
df.head()
```

Out[21]:

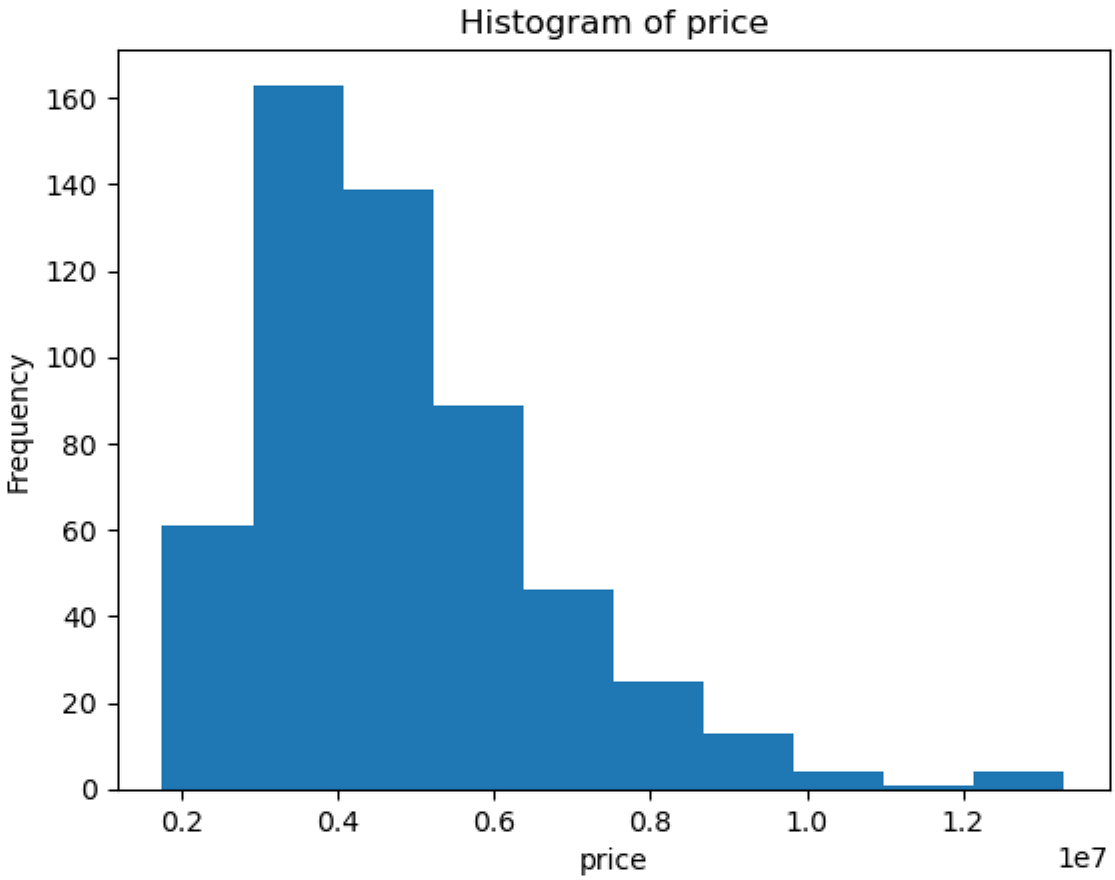
	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditic
0	13300000	7420	4	2	3	yes	no	no	no	
1	12250000	8960	4	4	4	yes	no	no	no	
2	12250000	9960	3	2	2	yes	no	yes	no	
3	12215000	7500	4	2	2	yes	no	yes	no	
4	11410000	7420	4	1	2	yes	yes	yes	no	

3. Perform Below Visualizations.

1. Univariate Analysis

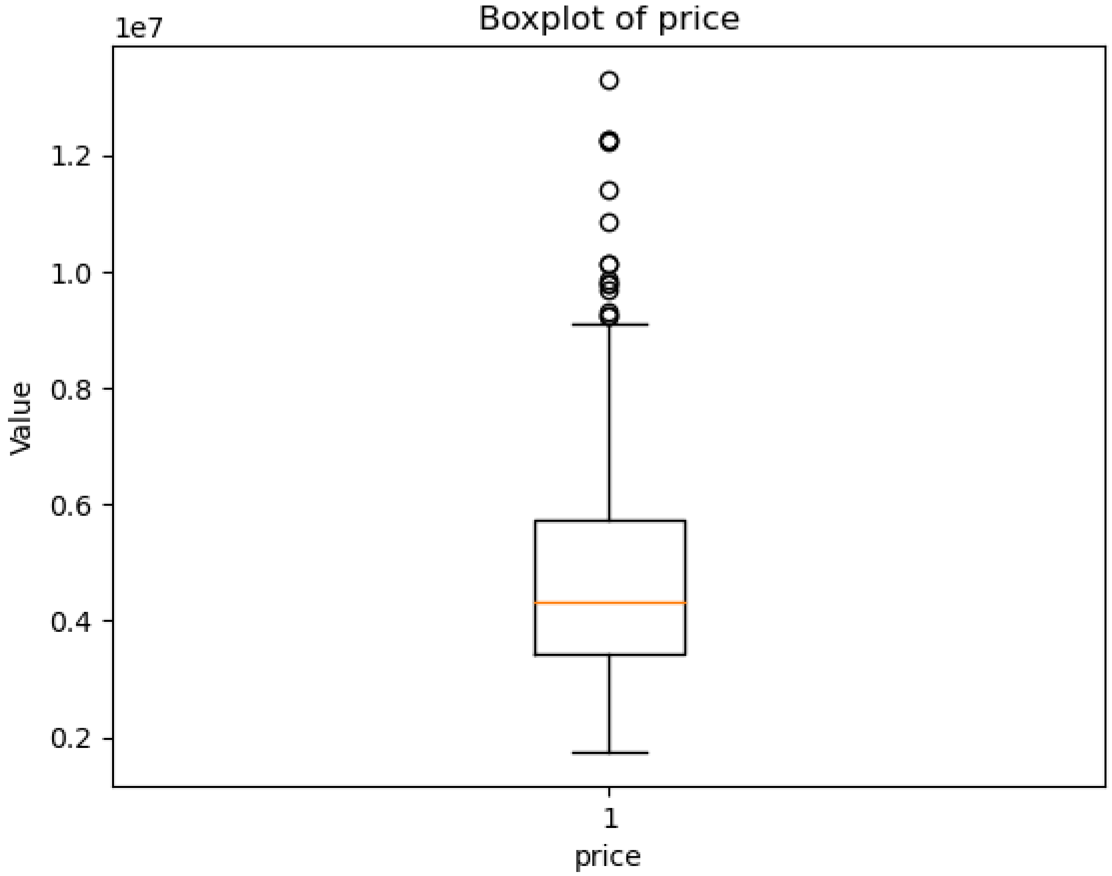
In [22]:

```
#3. Perform Below Visualizations.
#1. Univariate Analysis
# Histogram
plt.hist(df['price'], bins=10)
plt.title('Histogram of price')
plt.xlabel('price')
plt.ylabel('Frequency')
plt.show()
```

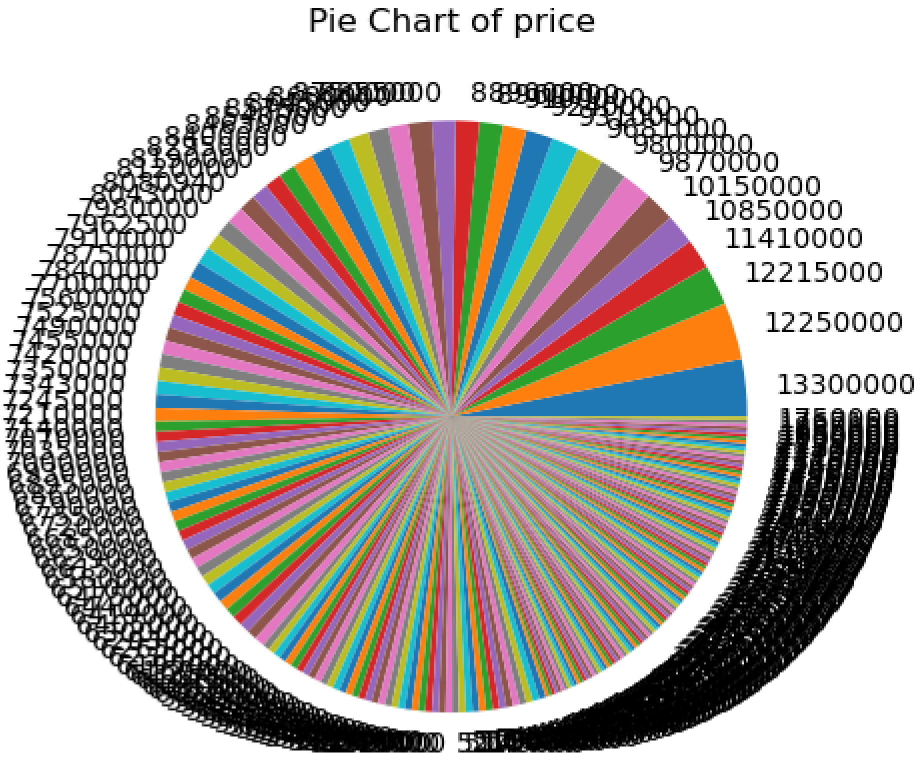


In [23]:

```
# Boxplot
plt.boxplot(df['price'])
plt.title('Boxplot of price')
plt.xlabel('price')
plt.ylabel('Value')
plt.show()
```



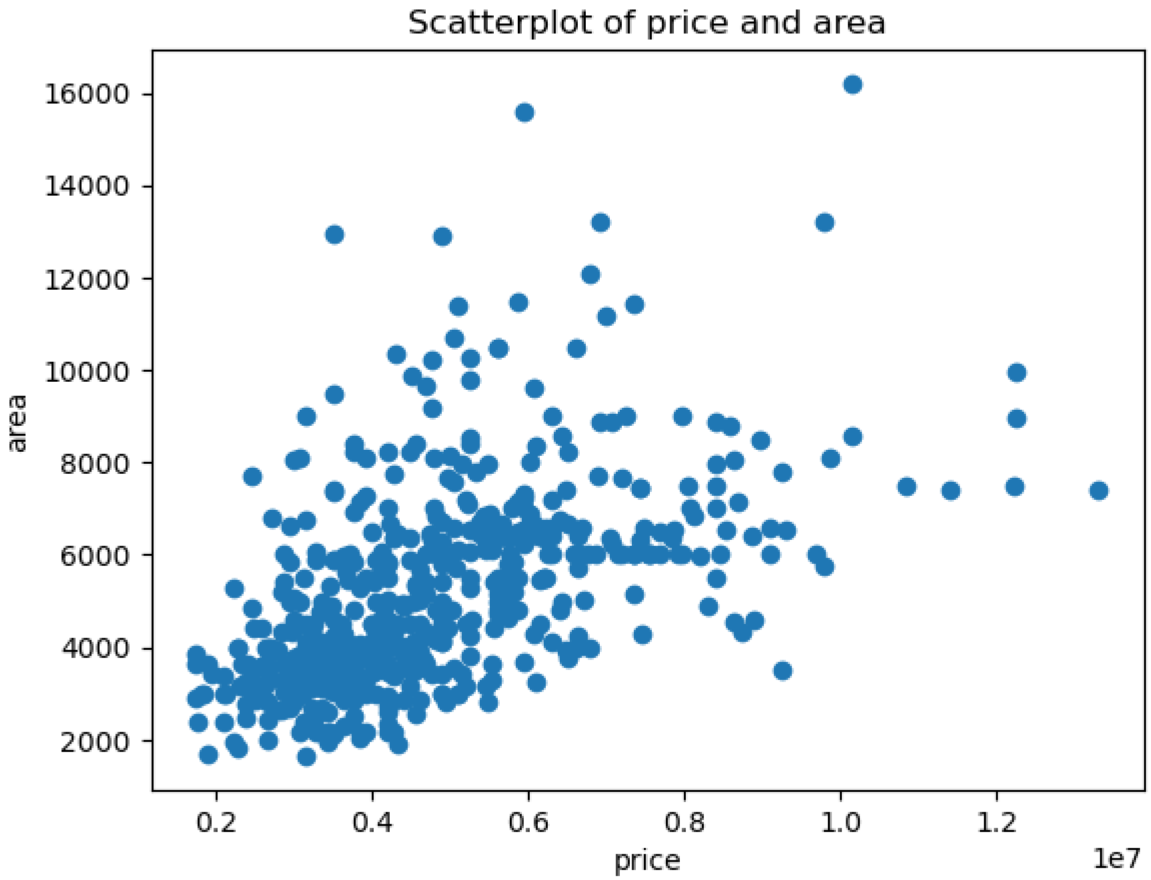
In [24]:
```
#Pie Chart
plt.pie(df['price'].value_counts(), labels=df['price'].unique())
plt.title('Pie Chart of price')
plt.show()
```



2.Bivariate analysis

In [25]:

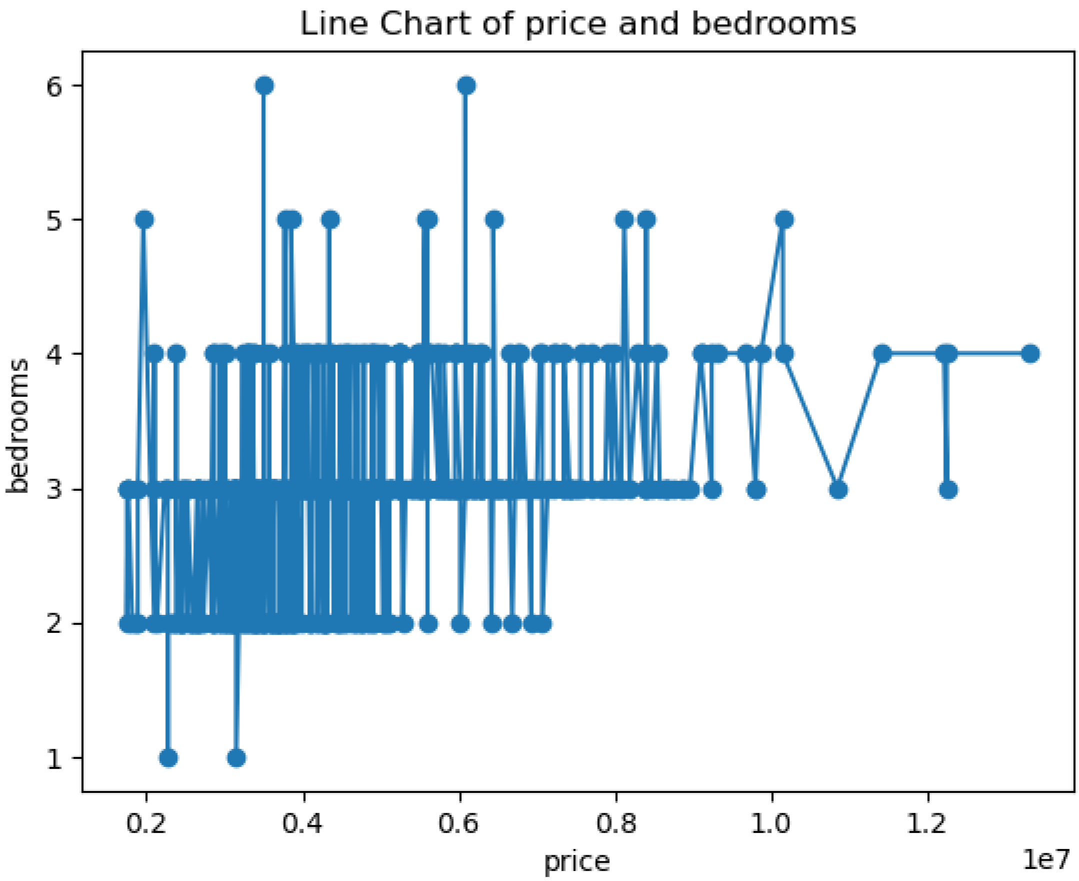
```
# Bivariate analysis
# Scatterplot
plt.scatter(df['price'], df['area'])
plt.title('Scatterplot of price and area')
plt.xlabel('price')
plt.ylabel('area')
plt.show()
```



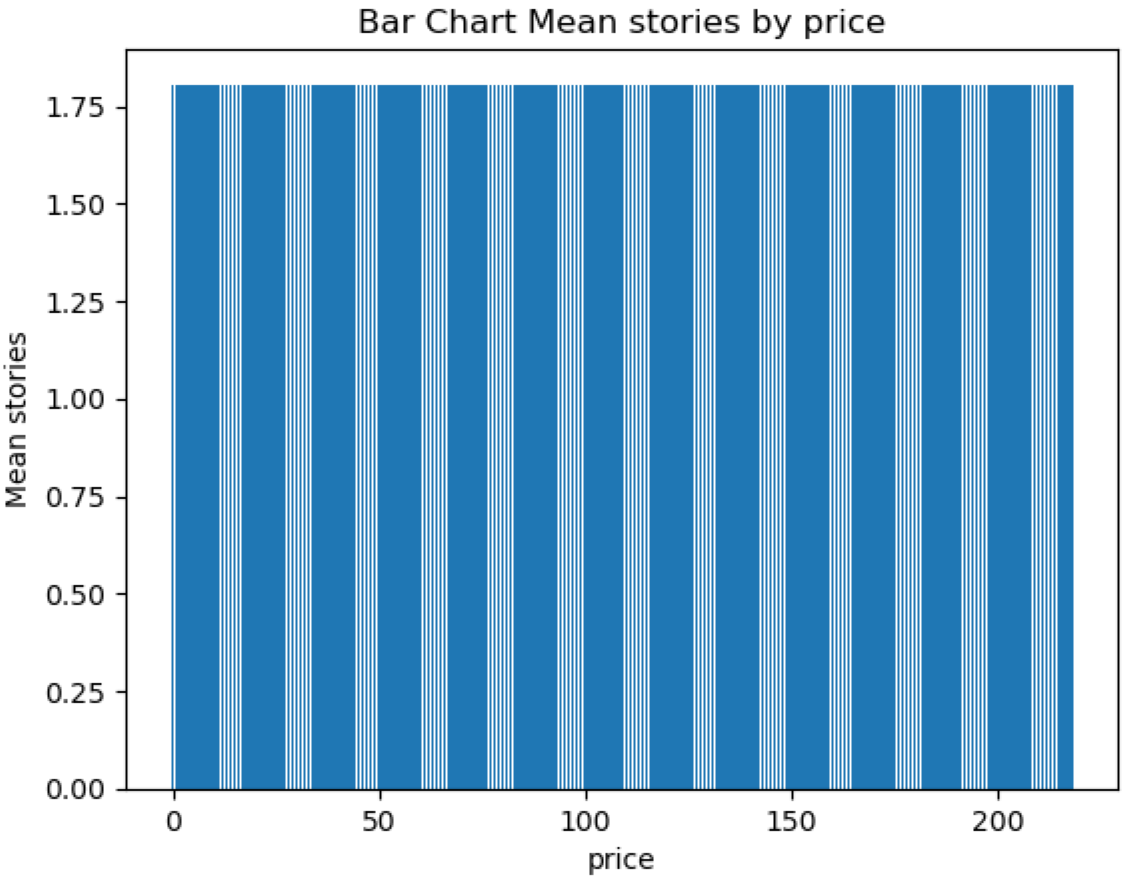
In [27]:

```
# Line chart
plt.plot(df['price'], df['bedrooms'], 'o-')
plt.title('Line Chart of price and bedrooms')
plt.xlabel('price')
plt.ylabel('bedrooms')
plt.show()
```

Loading [MathJax]/extensions/Safe.js



```
In [34]: # Bar chart
         plt.bar(df['price'].unique(), df['stories'].mean(), align='center')
         plt.title('Bar Chart Mean stories by price')
         plt.xlabel('price')
         plt.ylabel('Mean stories')
         plt.show()
```

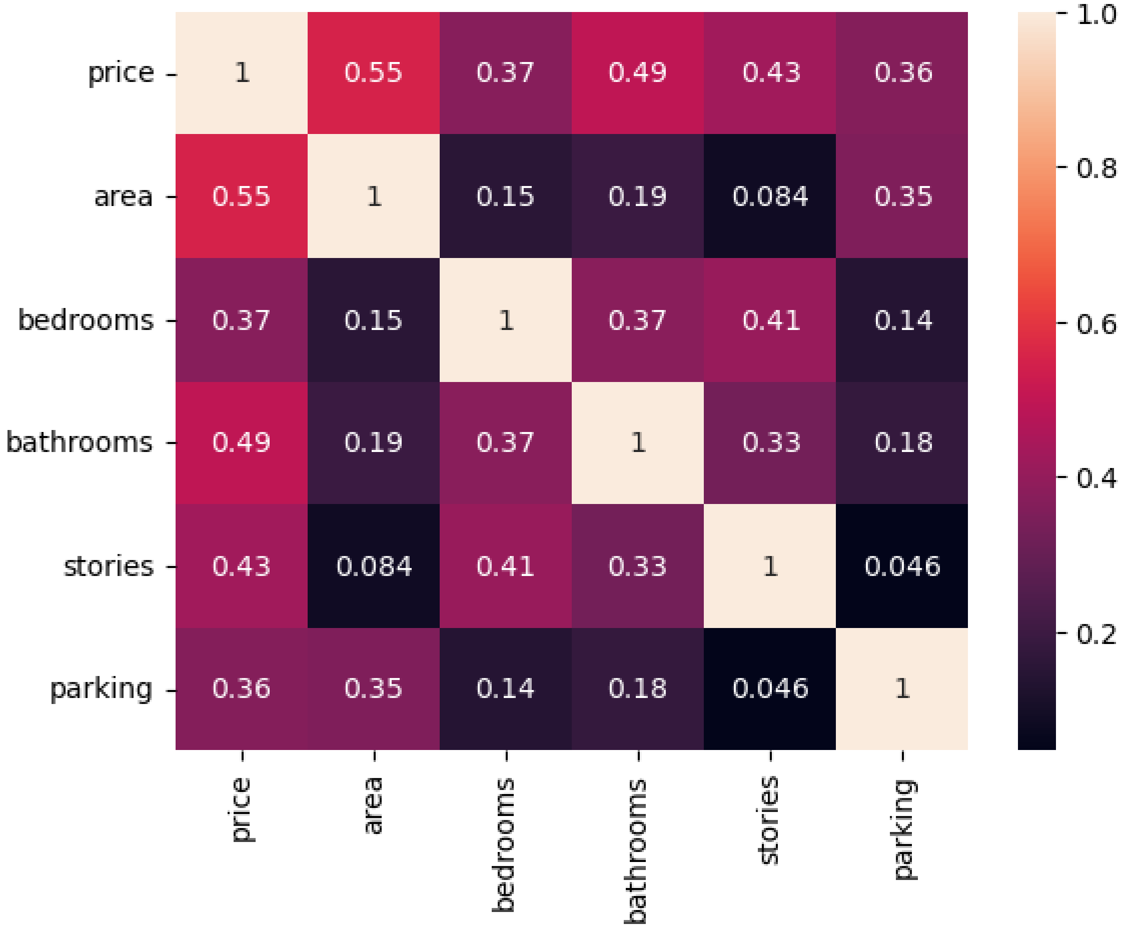


3. Multivariate analysis

In [32]:

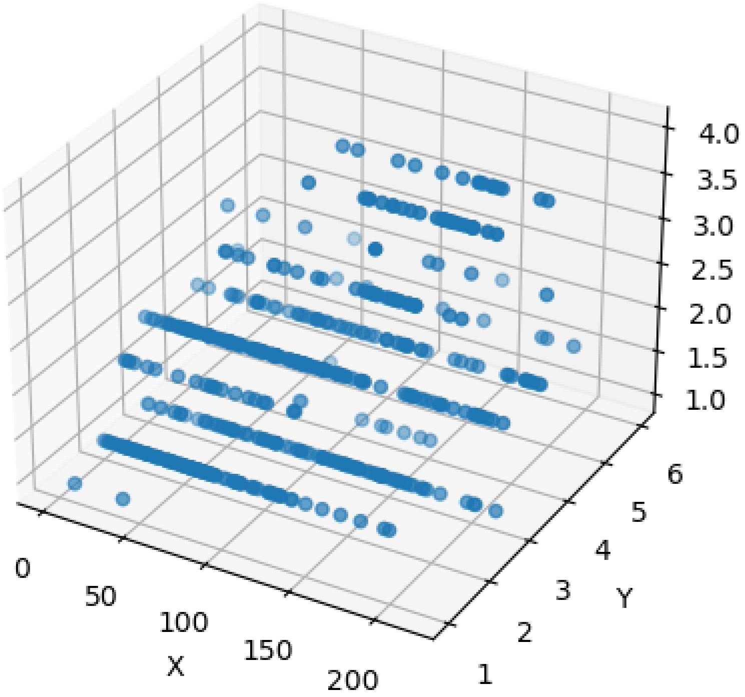
```
# Multivariate analysis
# Heatmap
df['price'] = df['price'].astype('category').cat.codes
sns.heatmap(df.corr(), annot=True)
plt.show()
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_3832\46082707.py:4: FutureWarning: The default
value of numeric_only in DataFrame.corr is deprecated. In a future version, it will defa
ult to False. Select only valid columns or specify the value of numeric_only to silence
this warning.
  sns.heatmap(df.corr(), annot=True)
```



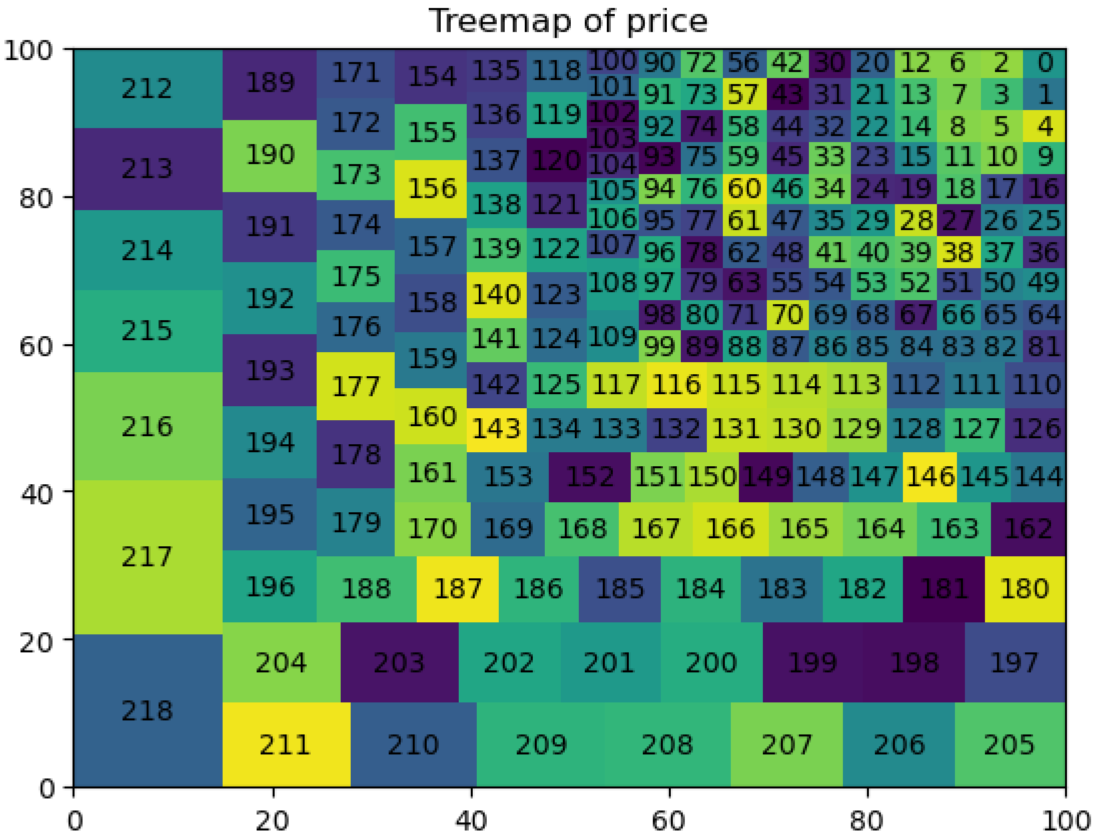
In [36]:

```
# Multivariate analysis
# 3D scatterplot
from mpl_toolkits.mplot3d import Axes3D
x = df['price']
y = df['bedrooms']
z = df['stories']
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(x, y, z)
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
plt.show()
```



In [47]:

```python
# Treemap
import squarify
plt.figure()
squarify.plot(df['price'].value_counts(), label=df['price'].unique())
plt.title('Treemap of price')
plt.show()
```



4. Perform descriptive statistics on the dataset.

Loading [MathJax]/extensions/Safe.js

In [49]:
```
#4. Perform descriptive statistics on the dataset.
df.describe()
```

Out[49]:

	price	area	bedrooms	bathrooms	stories	parking
count	545.000000	545.000000	545.000000	545.000000	545.000000	545.000000
mean	95.728440	5150.541284	2.965138	1.286239	1.805505	0.693578
std	56.256108	2170.141023	0.738064	0.502470	0.867492	0.861586
min	0.000000	1650.000000	1.000000	1.000000	1.000000	0.000000
25%	51.000000	3600.000000	2.000000	1.000000	1.000000	0.000000
50%	87.000000	4600.000000	3.000000	1.000000	2.000000	0.000000
75%	137.000000	6360.000000	3.000000	2.000000	2.000000	1.000000
max	218.000000	16200.000000	6.000000	4.000000	4.000000	3.000000

In [50]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 545 entries, 0 to 544
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   price             545 non-null    int16 
 1   area              545 non-null    int64 
 2   bedrooms          545 non-null    int64 
 3   bathrooms         545 non-null    int64 
 4   stories           545 non-null    int64 
 5   mainroad          545 non-null    object
 6   guestroom         545 non-null    object
 7   basement          545 non-null    object
 8   hotwaterheating   545 non-null    object
 9   airconditioning   545 non-null    object
 10  parking           545 non-null    int64 
 11  furnishingstatus  545 non-null    object
dtypes: int16(1), int64(5), object(6)
memory usage: 48.0+ KB
```

5. Check for Missing values and deal with them

In [51]:
```
#5. Check for Missing values and deal with them
df.isnull().sum()
```

Out[51]:
```
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
```

6. Find the outliers and replace the outliers

In [52]:
```
#6. Find the outliers and replace them outliers
target_column = 'price'
Q1 = df[target_column].quantile(0.25)
Q3 = df[target_column].quantile(0.75)
IQR = Q3 - Q1
```

In [53]:
```
IQR
```

Out[53]:
```
86.0
```

In [54]:
```
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
```

In [55]:
```
lower_bound
```

Out[55]:
```
-78.0
```

In [56]:
```
upper_bound
```

Out[56]:
```
266.0
```

In [57]:
```
outliers = df[(df[target_column] < lower_bound) | (df[target_column] > upper_bound)]
```

In [58]:
```
median_value = df[target_column].median()
df.loc[(df[target_column] < lower_bound) | (df[target_column] > upper_bound), target_col
```

In [59]:
```
median_value
```

Out[59]:
```
87.0
```

In [60]:
```
df
```

Out[60]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	aircondition
0	218	7420	4	2	3	yes	no	no	no	
1	217	8960	4	4	4	yes	no	no	no	
2	217	9960	3	2	2	yes	no	yes	no	
3	216	7500	4	2	2	yes	no	yes	no	
4	215	7420	4	1	2	yes	yes	yes	no	
...	...	...	...	...	...	...	...	...	...	
540	2	3000	2	1	1	yes	no	yes	no	
541	1	2400	3	1	1	no	no	no	no	
542	0	3620	2	1	1	yes	no	no	no	
543	0	2910	3	1	1	no	no	no	no	
544	0	3850	3	1	2	yes	no	no	no	

545 rows × 12 columns

In [61]:
```
print(df)
```

Loading [MathJax]/extensions/Safe.js

```
     price  area  bedrooms  bathrooms  stories mainroad guestroom basement  \
0      218  7420         4          2        3      yes        no       no   
1      217  8960         4          4        4      yes        no       no   
2      217  9960         3          2        2      yes        no      yes   
3      216  7500         4          2        2      yes        no      yes   
4      215  7420         4          1        2      yes       yes      yes   
..     ...   ...       ...        ...      ...      ...       ...      ...   
540      2  3000         2          1        1      yes        no      yes   
541      1  2400         3          1        1       no        no       no   
542      0  3620         2          1        1      yes        no       no   
543      0  2910         3          1        1       no        no       no   
544      0  3850         3          1        2      yes        no       no   

    hotwaterheating airconditioning  parking furnishingstatus  
0                no             yes        2        furnished  
1                no             yes        3        furnished  
2                no              no        2   semi-furnished  
3                no             yes        3        furnished  
4                no             yes        2        furnished  
..              ...             ...      ...              ...  
540              no              no        2      unfurnished  
541              no              no        0   semi-furnished  
542              no              no        0      unfurnished  
543              no              no        0        furnished  
544              no              no        0      unfurnished  

[545 rows x 12 columns]
```

7. Check for Categorical columns and perform encoding.

In [62]:

```
#7. Check for Categorical columns and perform encoding.
from sklearn.preprocessing import LabelEncoder
df.dtypes
```

Out[62]:

```
price                int16
area                 int64
bedrooms             int64
bathrooms            int64
stories              int64
mainroad            object
guestroom           object
basement            object
hotwaterheating     object
airconditioning     object
parking              int64
furnishingstatus    object
dtype: object
```

In [63]:

```
categorical_columns = df.select_dtypes(include=['object']).columns
df_encoded = pd.get_dummies(df, columns=categorical_columns)
```

In [64]:

```
categorical_columns
```

Out[64]:

```
Index(['mainroad', 'guestroom', 'basement', 'hotwaterheating',
       'airconditioning', 'furnishingstatus'],
      dtype='object')
```

In [66]:

```
print(df_encoded)
```

```
     price  area  bedrooms  bathrooms  stories  parking  mainroad_no  \
0      218  7420         4          2        3        2            0   
1      217  8960         4          4        4        3            0   
2      217  9960         3          2        2        2            0   
3      216  7500         4          2        2        3            0   
4      215  7420         4          1        2        2            0   
..     ...   ...       ...        ...      ...      ...          ...   
540      2  3000         2          1        1        2            0   
541      1  2400         3          1        1        0            1   
542      0  3620         2          1        1        0            0   
543      0  2910         3          1        1        0            1   
544      0  3850         3          1        2        0            0   

     mainroad_yes  guestroom_no  guestroom_yes  basement_no  basement_yes  \
0               1             1              0            1             0   
1               1             1              0            1             0   
2               1             1              0            0             1   
3               1             1              0            0             1   
4               1             0              1            0             1   
..            ...           ...            ...          ...           ...   
540             1             1              0            0             1   
541             0             1              0            1             0   
542             1             1              0            1             0   
543             0             1              0            1             0   
544             1             1              0            1             0   

     hotwaterheating_no  hotwaterheating_yes  airconditioning_no  \
0                     1                    0                   0   
1                     1                    0                   0   
2                     1                    0                   1   
3                     1                    0                   0   
4                     1                    0                   0   
..                  ...                  ...                 ...   
540                   1                    0                   1   
541                   1                    0                   1   
542                   1                    0                   1   
543                   1                    0                   1   
544                   1                    0                   1   

     airconditioning_yes  furnishingstatus_furnished  \
0                      1                           1   
1                      1                           1   
2                      0                           0   
3                      1                           1   
4                      1                           1   
..                   ...                         ...   
540                    0                           0   
541                    0                           0   
542                    0                           0   
543                    0                           1   
544                    0                           0   

     furnishingstatus_semi-furnished  furnishingstatus_unfurnished  
0                                  0                             0  
1                                  0                             0  
2                                  1                             0  
3                                  0                             0  
4                                  0                             0  
..                               ...                           ...  
540                                0                             1  
541                                1                             0  
542                                0                             1  
543                                0                             0  
544                                0                             1  
```

```
[545 rows x 19 columns]
```

8. Split the data into dependent and independent variables.

In [65]:

```
#8. Split the data into dependent and independent variables.
dependent_variable = 'price'
independent_variables = df.drop(dependent_variable, axis=1)
dependent_variable = df[dependent_variable]
```

In [67]:

```
print(dependent_variable)
```

```
0      218
1      217
2      217
3      216
4      215
      ...
540      2
541      1
542      0
543      0
544      0
Name: price, Length: 545, dtype: int16
```

In [68]:

```
independent_variables
```

Out[68]:

	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	pa
0	7420	4	2	3	yes	no	no	no	yes	
1	8960	4	4	4	yes	no	no	no	yes	
2	9960	3	2	2	yes	no	yes	no	no	
3	7500	4	2	2	yes	no	yes	no	yes	
4	7420	4	1	2	yes	yes	yes	no	yes	
...	...	...	...	...	...	...	...	...	...	
540	3000	2	1	1	yes	no	yes	no	no	
541	2400	3	1	1	no	no	no	no	no	
542	3620	2	1	1	yes	no	no	no	no	
543	2910	3	1	1	no	no	no	no	no	
544	3850	3	1	2	yes	no	no	no	no	

545 rows × 11 columns

In [69]:

```
print(independent_variables)
```

```
     area  bedrooms  bathrooms  stories mainroad guestroom basement  \
0    7420         4          2        3      yes        no       no   
1    8960         4          4        4      yes        no       no   
2    9960         3          2        2      yes        no      yes   
3    7500         4          2        2      yes        no      yes   
4    7420         4          1        2      yes       yes      yes   
..    ...       ...        ...      ...      ...       ...      ...   
540  3000         2          1        1      yes        no      yes   
541  2400         3          1        1       no        no       no   
542  3620         2          1        1      yes        no       no   
543  2910         3          1        1       no        no       no   
544  3850         3          1        2      yes        no       no   

    hotwaterheating airconditioning  parking furnishingstatus  
0                no             yes        2        furnished  
1                no             yes        3        furnished  
2                no              no        2   semi-furnished  
3                no             yes        3        furnished  
4                no             yes        2        furnished  
..              ...             ...      ...              ...  
540              no              no        2      unfurnished  
541              no              no        0   semi-furnished  
542              no              no        0      unfurnished  
543              no              no        0        furnished  
544              no              no        0      unfurnished  

[545 rows x 11 columns]
```

9. Scale the independent variables

In [70]:

```
#9. Scale the independent variables
from sklearn.preprocessing import StandardScaler
columns_to_scale = ['price', 'bedrooms', 'bathrooms', 'area', 'stories', 'parking']
scaler = StandardScaler()
df[columns_to_scale] = scaler.fit_transform(df[columns_to_scale])
```

In [71]:

```
df
```

Out[71]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating
0	2.175477	1.046726	1.403419	1.421812	1.378217	yes	no	no	no
1	2.157685	1.757010	1.403419	5.405809	2.532024	yes	no	no	no
2	2.157685	2.218232	0.047278	1.421812	0.224410	yes	no	yes	no
3	2.139893	1.083624	1.403419	1.421812	0.224410	yes	no	yes	no
4	2.122101	1.046726	1.403419	-0.570187	0.224410	yes	yes	yes	no
...	...	...	...	...	...	...	...	...	...
540	-1.667633	-0.991879	-1.308863	-0.570187	-0.929397	yes	no	yes	no
541	-1.685425	-1.268613	0.047278	-0.570187	-0.929397	no	no	no	no
542	-1.703217	-0.705921	-1.308863	-0.570187	-0.929397	yes	no	no	no
543	-1.703217	-1.033389	0.047278	-0.570187	-0.929397	no	no	no	no
544	-1.703217	-0.599839	0.047278	-0.570187	0.224410	yes	no	no	no

545 rows × 12 columns

```
        price      area  bedrooms  bathrooms   stories mainroad guestroom  \
0    2.175477  1.046726  1.403419   1.421812  1.378217      yes        no   
1    2.157685  1.757010  1.403419   5.405809  2.532024      yes        no   
2    2.157685  2.218232  0.047278   1.421812  0.224410      yes        no   
3    2.139893  1.083624  1.403419   1.421812  0.224410      yes        no   
4    2.122101  1.046726  1.403419  -0.570187  0.224410      yes       yes   
..        ...       ...       ...        ...       ...      ...       ...   
540 -1.667633 -0.991879 -1.308863  -0.570187 -0.929397      yes        no   
541 -1.685425 -1.268613  0.047278  -0.570187 -0.929397       no        no   
542 -1.703217 -0.705921 -1.308863  -0.570187 -0.929397      yes        no   
543 -1.703217 -1.033389  0.047278  -0.570187 -0.929397       no        no   
544 -1.703217 -0.599839  0.047278  -0.570187  0.224410      yes        no   

    basement hotwaterheating airconditioning   parking furnishingstatus  
0         no              no             yes  1.517692        furnished  
1         no              no             yes  2.679409        furnished  
2        yes              no              no  1.517692   semi-furnished  
3        yes              no             yes  2.679409        furnished  
4        yes              no             yes  1.517692        furnished  
..       ...             ...             ...       ...              ...  
540      yes              no              no  1.517692      unfurnished  
541       no              no              no -0.805741   semi-furnished  
542       no              no              no -0.805741      unfurnished  
543       no              no              no -0.805741        furnished  
544       no              no              no -0.805741      unfurnished  

[545 rows x 12 columns]
```

10.Split the data into training and testing

In [73]:

```
#10.Split the data into training and testing
from sklearn.model_selection import train_test_split
X = df.drop('price', axis=1)
y = df['price']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=4
```

In [74]:

```
X_train
```

Out[74]:

	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	aircondition
167	-0.253922	-1.308863	1.421812	-0.929397	yes	no	no	no	
368	0.225750	-1.308863	-0.570187	-0.929397	no	no	no	no	
301	-0.752043	0.047278	-0.570187	0.224410	yes	no	no	no	
527	-1.528742	-1.308863	-0.570187	-0.929397	no	no	yes	no	
382	-0.922695	0.047278	-0.570187	0.224410	yes	no	yes	no	
...	...	...	...	...	...	...	...	...	
71	0.391790	1.403419	1.421812	2.532024	yes	no	no	no	
106	0.138117	1.403419	1.421812	-0.929397	yes	no	yes	no	
270	-0.300045	0.047278	1.421812	1.378217	yes	no	no	yes	
435	-0.512207	-1.308863	-0.570187	-0.929397	yes	no	no	no	
102	0.161178	0.047278	1.421812	2.532024	yes	yes	no	no	

408 rows × 11 columns

Out[75]:

	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	aircondition
316	0.345668	1.403419	1.421812	0.224410	no	no	yes	no	
77	0.622401	0.047278	1.421812	1.378217	yes	no	no	no	
360	-0.512207	-1.308863	-0.570187	-0.929397	yes	no	no	no	
90	-0.069433	0.047278	-0.570187	0.224410	yes	no	no	no	
493	-0.549105	0.047278	-0.570187	-0.929397	yes	no	no	no	
...	...	...	...	...	...	...	...	...	
172	1.498725	0.047278	-0.570187	0.224410	yes	yes	yes	no	
124	0.633932	0.047278	1.421812	2.532024	yes	no	no	no	
388	-0.692084	0.047278	-0.570187	0.224410	yes	no	no	no	
521	-0.699002	-1.308863	-0.570187	-0.929397	no	no	no	no	
503	-0.530656	0.047278	-0.570187	-0.929397	yes	no	no	no	

137 rows × 11 columns

In [76]:

```
y_train
```

Out[76]:

```
167    0.520805
368   -0.635687
301   -0.262051
527   -1.525296
382   -0.706855
         ...   
71     1.285868
106    0.983401
270   -0.155298
435   -0.920362
102    1.001194
Name: price, Length: 408, dtype: float64
```

In [77]:

```
y_test
```

Out[77]:

```
316   -0.386596
77     1.232492
360   -0.600102
90     1.125739
493   -1.276205
         ...   
172    0.503013
124    0.876648
388   -0.742440
521   -1.454127
503   -1.329582
Name: price, Length: 137, dtype: float64
```

In [78]:

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

In [79]:

```
df['mainroad']=le.fit_transform(df['mainroad'])
df['guestroom']=le.fit_transform(df['guestroom'])
df['basement']=le.fit_transform(df['basement'])
df['hotwaterheating']=le.fit_transform(df['hotwaterheating'])
df['airconditioning']=le.fit_transform(df['airconditioning'])
df['furnishingstatus']=le.fit_transform(df['furnishingstatus'])
```

Loading [MathJax]/extensions/Safe.js

Out[80]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	aircc
0	2.175477	1.046726	1.403419	1.421812	1.378217	1	0	0	0	
1	2.157685	1.757010	1.403419	5.405809	2.532024	1	0	0	0	
2	2.157685	2.218232	0.047278	1.421812	0.224410	1	0	1	0	
3	2.139893	1.083624	1.403419	1.421812	0.224410	1	0	1	0	
4	2.122101	1.046726	1.403419	-0.570187	0.224410	1	1	1	0	

11. Build the Model

In [81]:

```
#11. Build the Model
from sklearn.linear_model import LinearRegression
model=LinearRegression()
X_train, X_test, y_train, y_test = train_test_split(df, df['price'], test_size=0.25)
```

In [83]:

```
model.fit(X_train,y_train)
```

Out[83]:

```
▼ LinearRegression
LinearRegression()
```

12. Train the model

In [84]:

```
#12. Train the model
X_train
```

Out[84]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating
147	0.698727	0.161178	0.047278	1.421812	0.224410	1	0	0	0
284	-0.226467	1.208154	-1.308863	-0.570187	-0.929397	1	0	0	0
396	-0.742440	-0.696696	-1.308863	-0.570187	-0.929397	1	0	0	0
88	1.143531	1.042114	0.047278	-0.570187	-0.929397	1	1	1	0
138	0.734311	-0.069433	0.047278	-0.570187	1.378217	1	0	0	0
...	...	...	...	...	...	...	...	...	...
242	0.022624	-0.696696	0.047278	-0.570187	0.224410	1	0	0	0
343	-0.493349	-0.493758	-1.308863	-0.570187	-0.929397	1	0	0	0
59	1.445998	0.391790	0.047278	1.421812	2.532024	1	1	0	0
317	-0.386596	-0.073123	0.047278	1.421812	0.224410	1	0	0	0
185	0.414052	-0.991879	0.047278	-0.570187	0.224410	1	0	1	0

408 rows × 12 columns

In [85]:

```
y_train
```

Loading [MathJax]/extensions/Safe.js

Out[85]:

```
147    0.698727
284   -0.226467
396   -0.742440
88     1.143531
138    0.734311
         ...   
242    0.022624
343   -0.493349
59     1.445998
317   -0.386596
185    0.414052
Name: price, Length: 408, dtype: float64
```

13. Test the model

In [86]:

```
#13. Test the model
score = model.score(X_test, y_test)
```

In [87]:

```
X_test
```

Out[87]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating
539	-1.649841	-0.996491	-1.308863	-0.570187	-0.929397	0	0	0	0
328	-0.439973	-0.300045	0.047278	1.421812	0.224410	0	0	1	0
226	0.111585	0.008975	0.047278	-0.570187	2.532024	1	0	0	0
342	-0.493349	0.923119	0.047278	-0.570187	0.224410	1	0	0	0
490	-1.258413	-0.369228	0.047278	-0.570187	0.224410	0	0	0	1
...	...	...	...	...	...	...	...	...	...
110	0.930025	0.668524	0.047278	-0.570187	-0.929397	1	1	1	0
155	0.663142	0.437912	0.047278	1.421812	-0.929397	1	0	1	0
209	0.253922	0.723870	0.047278	-0.570187	-0.929397	1	0	0	0
7	2.086516	5.096263	2.759560	3.413810	0.224410	1	0	0	0
500	-1.329582	-1.084123	0.047278	-0.570187	-0.929397	1	0	0	0

137 rows × 12 columns

In [88]:

```
y_test
```

Out[88]:

```
539   -1.649841
328   -0.439973
226    0.111585
342   -0.493349
490   -1.258413
         ...   
110    0.930025
155    0.663142
209    0.253922
7      2.086516
500   -1.329582
Name: price, Length: 137, dtype: float64
```

In [89]:

```
score
```

Out[89]:

```
1.0
```

Loading [MathJax]/extensions/Safe.js

```
In [90]: predictions = model.predict(X_test)
```

```
In [91]: predictions
```

```
Out[91]: array([-1.64984094, -0.43997282,  0.11158471, -0.49334935, -1.25841302,
               -0.97373817, -0.74243985, -0.26205104,  1.12573887,  1.25028412,
                0.25392213,  1.55275115,  1.60612768, -0.51114153,  1.33924501,
               -0.457765  , -0.79581638,  1.81963382, -1.32958173,  0.6097657 ,
               -0.95594599, -0.97373817, -0.92036163, -0.457765  , -0.19088232,
                0.14716906, -0.457765  , -0.38659628, -0.84919292,  0.44963609,
               -0.6178946 , -0.10192143,  0.37846738,  0.85885619,  1.87301035,
               -0.26205104, -0.56451807,  0.37846738, -1.13386777, -0.03075272,
                0.2005456 , -1.34737391,  1.57054332,  0.50301263, -1.40075045,
               -0.26205104, -1.6854253 , -0.31542757,  0.48522045,  1.07236233,
               -1.70321748,  2.10430867,  0.09379253, -0.74243985,  0.07600035,
               -1.66763312, -1.4363348 , -1.2762052 , -1.20503648,  0.18275342,
               -0.54672589,  0.3606752 ,  1.46379025,  0.94781709, -0.26205104,
                1.37482936, -0.92036163, -0.40438846, -0.65347896, -1.29399738,
               -0.83140074, -1.22282866,  0.50301263,  0.44963609, -0.24425886,
               -0.74243985, -0.2086745 ,  0.02262382, -1.20503648, -0.79581638,
               -0.77802421, -1.22282866,  1.94417907, -0.22646668,  0.07600035,
                1.64171204,  1.97976342, -0.74243985,  1.5883355 , -0.56451807,
               -0.74243985, -0.36880411,  0.82327184,  1.76625728, -1.40075045,
               -0.97373817,  2.17547738, -0.81360856,  0.00483164,  1.23249194,
                0.25392213,  1.21469976, -1.06269906, -0.65347896,  1.23249194,
               -1.08049124,  2.06872432,  1.41041372,  0.32509085, -0.457765  ,
               -0.65347896,  0.69872659, -1.06269906,  2.03313996, -0.99153035,
               -0.49334935,  2.05093214, -0.60010242,  1.49937461, -0.72464767,
                1.05457015, -0.13750579, -0.92036163, -0.79581638, -1.57867223,
               -1.70321748,  0.41405174, -0.74243985, -1.09828341, -1.20503648,
                0.2005456 ,  1.23249194,  0.93002491,  0.66314223,  0.25392213,
                2.08651649, -1.32958173])
```

14. Measure the performance using Metrics

```
In [92]: #14. Measure the performance using Metrics
         from sklearn.metrics import mean_squared_error,r2_score, mean_absolute_error
         y_pred = model.predict(X_test)
```

```
In [93]: error=y_test-y_pred
```

```
In [94]: error
```

```
Out[94]: 539   -6.661338e-16
         328    1.165734e-15
         226    9.575674e-16
         342    6.661338e-16
         490   -3.108624e-15
                     ...
         110   -6.661338e-16
         155    5.662137e-15
         209   -7.993606e-15
         7     -1.909584e-14
         500   -3.774758e-15
         Name: price, Length: 137, dtype: float64
```

```
In [95]: se=error*error
```

```
In [96]: se
```

Loading [MathJax]/extensions/Safe.js

```
Out[96]: 539    4.437343e-31
         328    1.358936e-30
         226    9.169352e-31
         342    4.437343e-31
         490    9.663546e-30
                    ...
         110    4.437343e-31
         155    3.205980e-29
         209    6.389773e-29
         7      3.646510e-28
         500    1.424880e-29
         Name: price, Length: 137, dtype: float64
```

```
In [97]: mse=np.mean(se)
```

```
In [98]: mse
```

```
Out[98]: 2.464189690914093e-29
```

```
In [99]: mse2=mean_squared_error(y_test,y_pred)
```

```
In [100… mse2
```

```
Out[100]: 2.464189690914093e-29
```

```
In [101… mae=mean_absolute_error(y_test,y_pred)
```

```
In [102… mae
```

```
Out[102]: 3.9692625706925375e-15
```

```
In [103… rmse=np.sqrt(mse2)
```

```
In [104… rmse
```

```
Out[104]: 4.964060526337378e-15
```

```
In [105… r2=r2_score(y_test,y_pred)
```

```
In [106… r2
```

```
Out[106]: 1.0
```

```
In [ ]:
```