MACHINE LEARNING PROJECT

REPORT AND ANALYSIS

GROUP:

A.CHARITHA REDDY : SE22UCSE016

ABHIGNA KAPPAGANTULA : SE22UCSE003

K.RITHWIKA : SE22UCSE121

SRUTHI LIGAMPALLI : SE22UCSE150

We start by importing all the necessary libraries:

```
# importing libraries
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plot
import seaborn as sns
from sklearn.preprocessing import OneHotEncoder
from scipy.stats import norm
from scipy.stats.mstats import winsorize
from sklearn.impute import SimpleImputer, KNNImputer
from scipy.stats import zscore
from scipy import stats
```

TASK 1:

```
#loading dataset into python
abc=pd.read_csv("C:\\Users\\bhara\\Downloads\\Gurgaon_RealEstate.csv")
#to identify all features and datatypes
print(abc.info())
```

```
TASK 1
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3803 entries, 0 to 3802
Data columns (total 23 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   property_type         3803 non-null   object
 1   society               3802 non-null   object
 2   sector                3803 non-null   object
 3   price                 3785 non-null   float64
 4   price_per_sqft        3785 non-null   float64
 5   area                  3785 non-null   float64
 6   areaWithType          3803 non-null   object
 7   bedRoom               3803 non-null   int64
 8   bathroom              3803 non-null   int64
 9   balcony               3803 non-null   object
 10  floorNum              3784 non-null   float64
 11  facing                2698 non-null   object
 12  agePossession         3803 non-null   object
 13  super_built_up_area   1915 non-null   float64
 14  built_up_area         1733 non-null   float64
 15  carpet_area           1944 non-null   float64
 16  studyroom             3803 non-null   int64
 17  servantroom           3803 non-null   int64
 18  storeroom             3803 non-null   int64
 19  poojaroom             3803 non-null   int64
 20  others                3803 non-null   int64
 21  furnishing_type       3803 non-null   int64
 22  luxury_score          3803 non-null   int64
dtypes: float64(7), int64(9), object(7)

memory usage: 683.5+ KB
None
```

After loading all the necessary libraries, we check for duplicate rows:

```
#to check if there is any duplicate row
DUPLICATEE_ROWSS=abc[abc.duplicated()]
#to print duplicate rows.
print("Duplicate Rows:",DUPLICATEE_ROWSS)
```

```
Duplicate Rows:      property_type                    society      sector
price  \
72           flat            smart world gems   sector 89   0.95
357          flat         umang monsoon breeze   sector 78   0.75
502          flat   central park flower valley   sector 33   1.40
528          flat                essel towers   sector 14   2.20
580          flat    signature global city 63a  sector 63a   1.65
…             …                          …          …    …
3740         flat         siddhartha apartment   sector 95   1.10
3754         flat                sare petioles   sector 92   1.30
3755         flat        vatika sovereign park   sector 99   3.25
3767         flat                  ss the leaf   sector 85   1.93
3769         flat                 maruti vihar   sector 28   1.00

      price_per_sqft    area  \
72            8600.0  1105.0
357           6053.0  1239.0
502          21538.0   650.0
528          12500.0  1760.0
580          15263.0  1081.0
…              …     …
3740          4327.0  2542.0
3754          6373.0  2040.0
3755          9672.0  3360.0
3767          8075.0  2390.0
3769          9090.0  1100.0

                                      areaWithType  bedRoom  bathroom  \
72                     Carpet area: 1103 (102.47 sq.m.)        2         2
357   Built Up area: 1239 (115.11 sq.m.)Carpet area:…        2         2
502                Super Built up area 650(60.39 sq.m.)        3         3
528                    Carpet area: 1760 (163.51 sq.m.)        3         4
580              Super Built up area 1081(100.43 sq.m.)        2         2
…                                                  …      …       …
3740                   Carpet area: 2542 (236.16 sq.m.)        4         3
3754                Built Up area: 2040 (189.52 sq.m.)        4         4
3755             Super Built up area 3360(312.15 sq.m.)        4         4
3767  Super Built up area 2408(223.71 sq.m.)Built Up…        3         4
3769             Super Built up area 1100(102.19 sq.m.)        1         1

      balcony  …  super_built_up_area built_up_area carpet_area  studyroom  \
72          2  …                 NaN           NaN      1103.0          1
357         1  …                 NaN        1239.0      1100.0          0
502         3  …               650.0           NaN         NaN          0
528        3+  …                 NaN           NaN      1760.0          0
580         2  …              1081.0           NaN         NaN          0
```

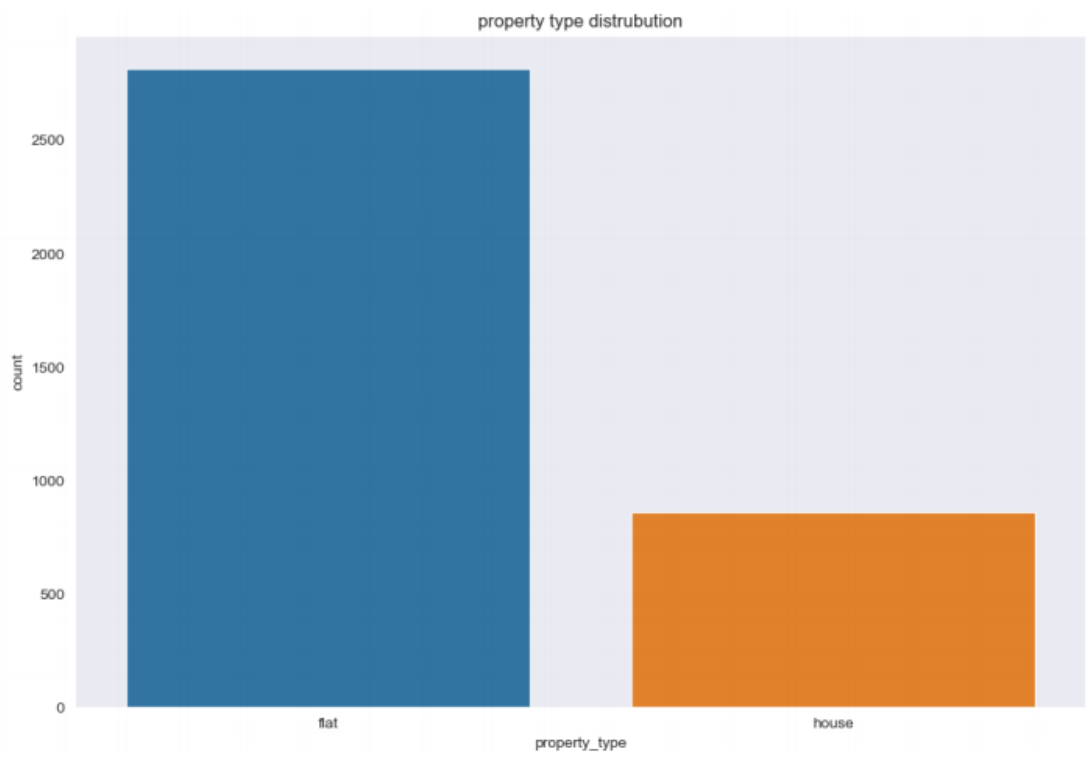
```
...    ...  ...                  ...          ...         ...         ...
3740     3  ...                  NaN          NaN      2542.0           0
3754     0  ...                  NaN       2040.0         NaN           0
3755    3+  ...               3360.0          NaN         NaN           0
3767     3  ...               2408.0       2400.0      2390.0           0
3769     1  ...               1100.0          NaN         NaN           1

      servantroom  storeroom  poojaroom  others  furnishing_type  luxury_score
72              1          0          0       0                0            38
357             0          0          0       0                0             0
502             0          0          0       0                0            54
528             0          0          0       0                0             0
580             0          0          0       0                0             0
...           ...        ...        ...     ...              ...           ...
3740            1          0          0       1                0            22
3754            0          0          0       0                0             0
3755            1          0          0       0                0           123
3767            1          0          0       0                0           174
3769            0          0          1       0                1            80

[126 rows x 23 columns]
```

```
#to remove duplicate rows
abc=abc.drop_duplicates()
```

we now start exploring each column of our dataset:

EXPLORING property_type COLUMN:

```
#to explore property_type column(flat or house)
plot.figure(figsize=(12,8))
sns.countplot(x='property_type',data=abc)
plot.title('property type distrubution')
plot.xlabel('property_type')
plot.ylabel('count')
plot.show()
```

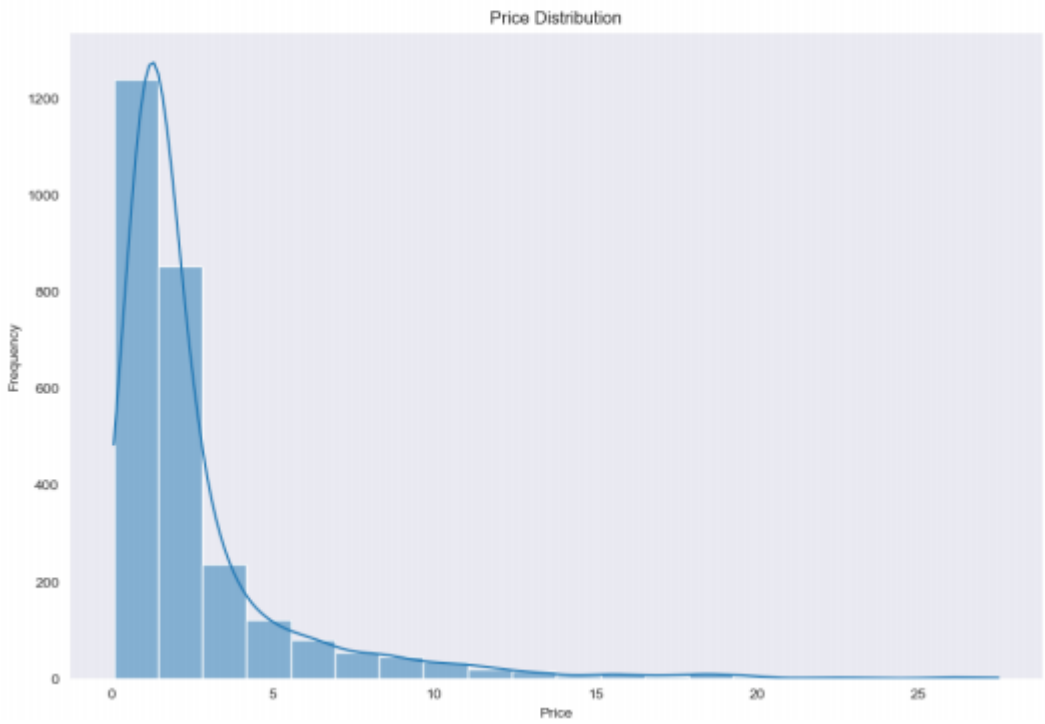


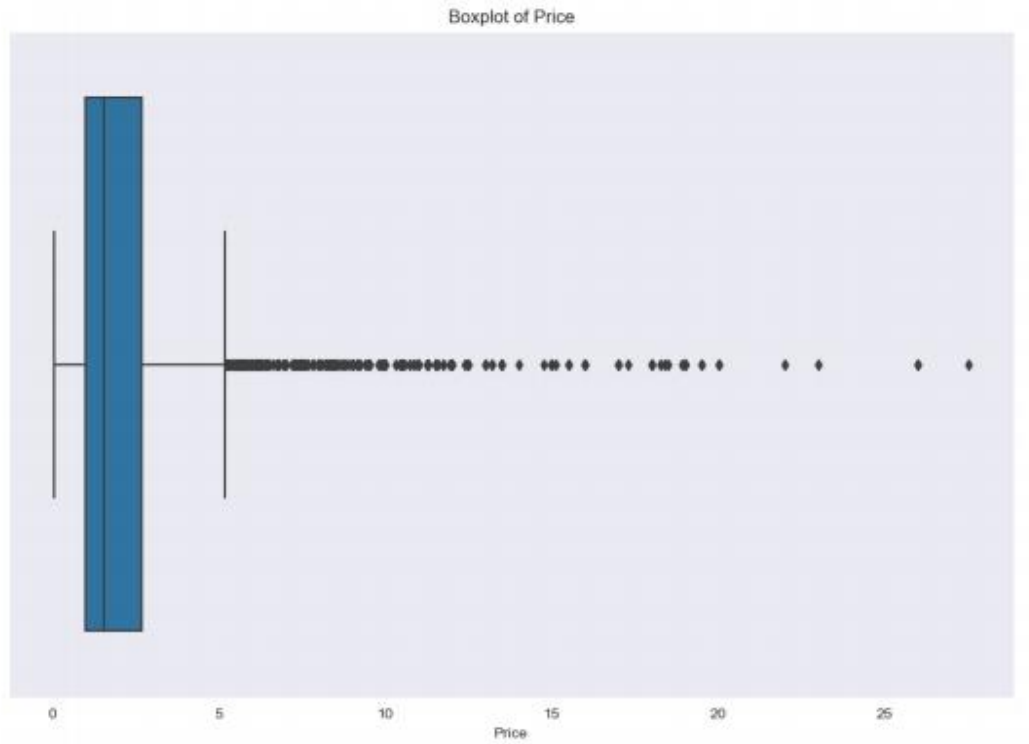

```
# to explore society column
society_counts = abc['society'].value_counts()
# Define a threshold for the minimum number of flats or houses per society
threshold = 6
valid_societies = society_counts[society_counts >= threshold].index
abc = abc[abc['society'].isin(valid_societies)]
```

EXPLORING PRICE COLUMN :

```
# to explore price column
print("Missing values in price column:", abc['price'].isnull().sum())
print("Descriptive statistics for price column:\n", abc['price'].describe())
plot.figure(figsize=(12,8))
sns.histplot(abc['price'], bins=20, kde=True)
plot.title('Price Distribution')
plot.xlabel('Price')
plot.ylabel('Frequency')
plot.show()
plot.figure(figsize=(12,8))
sns.boxplot(x=abc['price'])
plot.title('Boxplot of Price')
plot.xlabel('Price')
plot.show()
print("Skewness of price column:", abc['price'].skew())
print("Kurtosis of price column:", abc['price'].kurt())
```

```
Missing values in price column: 14


Descriptive statistics for price column:
 count    2729.000000
mean        2.542774
std         2.951175
min         0.070000
25%         1.000000
50%         1.550000
75%         2.680000
max        27.500000
Name: price, dtype: float64
```





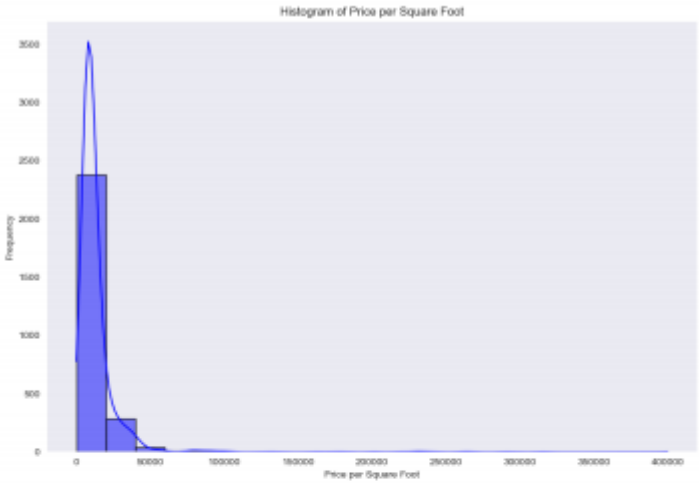

```
Skewness of price column: 3.2596843372474167
Kurtosis of price column: 13.932438148143515
Number of missing values: 14
```

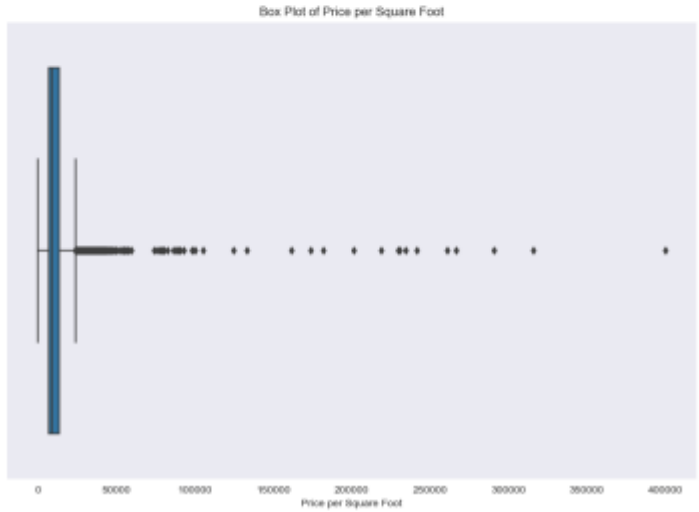
EXPLORING PRICE_PER_SQFT COLUMN :

```
# to explore price_per_sqft

missing_values = abc['price_per_sqft'].isnull().sum()
print("Number of missing values:", missing_values)

print("Descriptive statistics of price_per_sqft column:")
print(abc['price_per_sqft'].describe())
#hostogram for price_per_sqft
plot.figure(figsize=(12,8))
sns.histplot(abc['price_per_sqft'], bins=20, kde=True, color='blue',
edgecolor='black')
plot.xlabel('Price per Square Foot')
plot.ylabel('Frequency')
plot.title('Histogram of Price per Square Foot')
plot.show()
#boxplot for outliers of price_per_sqft
plot.figure(figsize=(12,8))
sns.boxplot(x=abc['price_per_sqft'])
plot.xlabel('Price per Square Foot')
plot.title('Box Plot of Price per Square Foot')
```

```
plot.show()
#to check skewness and kurtosis
skewness = abc['price_per_sqft'].skew()
kurtosis = abc['price_per_sqft'].kurtosis()
print("Skewness:",skewness)
print("Kurtosis:",kurtosis)
```

```
Name: price_per_sqft, dtype: float64
Descriptive statistics of price_per_sqft column:
count      2729.000000
mean      13550.251008
std       20032.737581
min           4.000000
25%        6953.000000
50%        9025.000000
75%       13778.000000
max      400000.000000
Name: price_per_sqft, dtype: float64
```





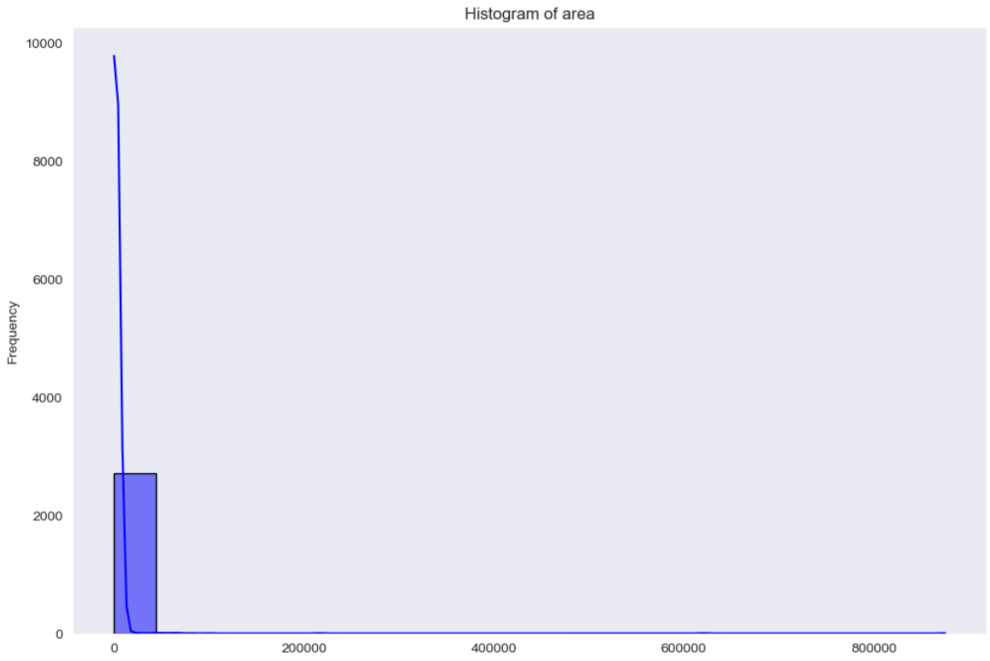

```
Skewness: 10.0750733210979
Kurtosis: 133.5677710633802
Number of missing values: 14
```

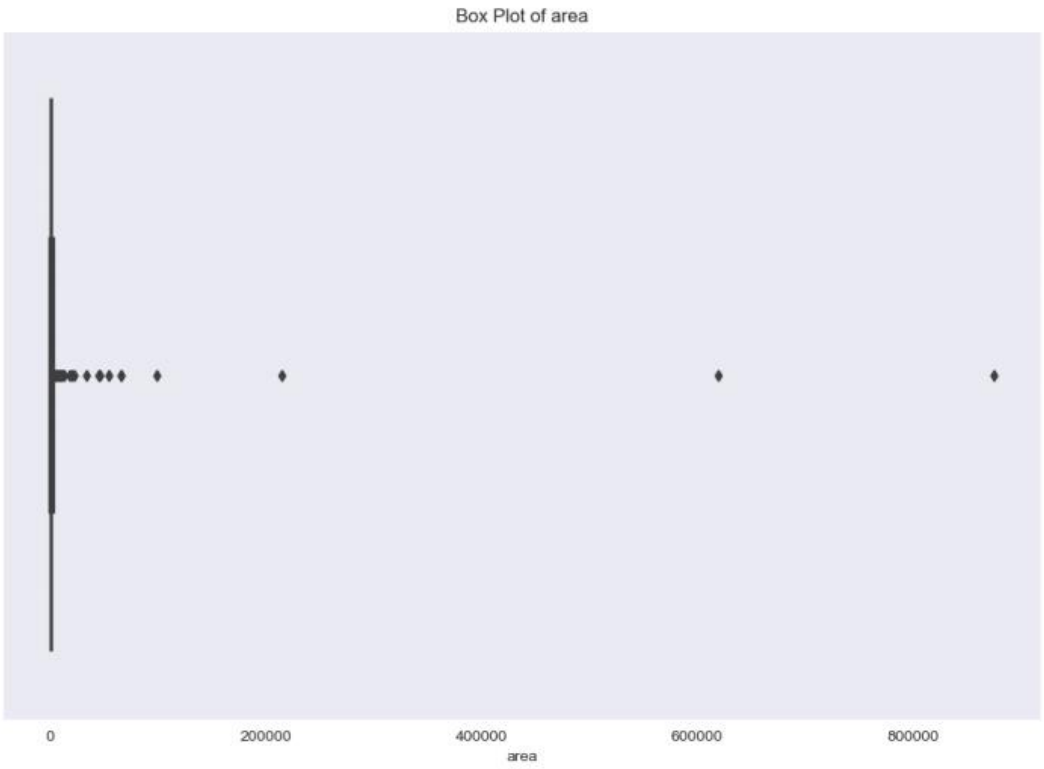
EXPLORING AREA COLUMN:

```
# to explore area column
missing_values = abc['area'].isnull().sum()
print("Number of missing values:", missing_values)

print("Descriptive statistics of area column:")
print(abc['area'].describe())
#hostogram for area
plot.figure(figsize=(12,8))
```

```
sns.histplot(abc['area'], bins=20, kde=True, color='blue', edgecolor='black')
plot.xlabel('area')
plot.ylabel('Frequency')
plot.title('Histogram of area')
plot.show()
#boxplot for outliers of area
plot.figure(figsize=(12,8))
sns.boxplot(x=abc['area'])
plot.xlabel('area')
plot.title('Box Plot of area')
plot.show()
#to check skewness and kurtosis
skewness = abc['area'].skew()
kurtosis = abc['area'].kurtosis()
print("Skewness:",skewness)
print("Kurtosis:",kurtosis)
```

```
Descriptive statistics of area column:
count      2729.000000
mean       2701.393551
std       21131.306236
min          50.000000
25%        1280.000000
50%        1752.000000
75%        2298.000000
max      875000.000000
Name: area, dtype: float64
```





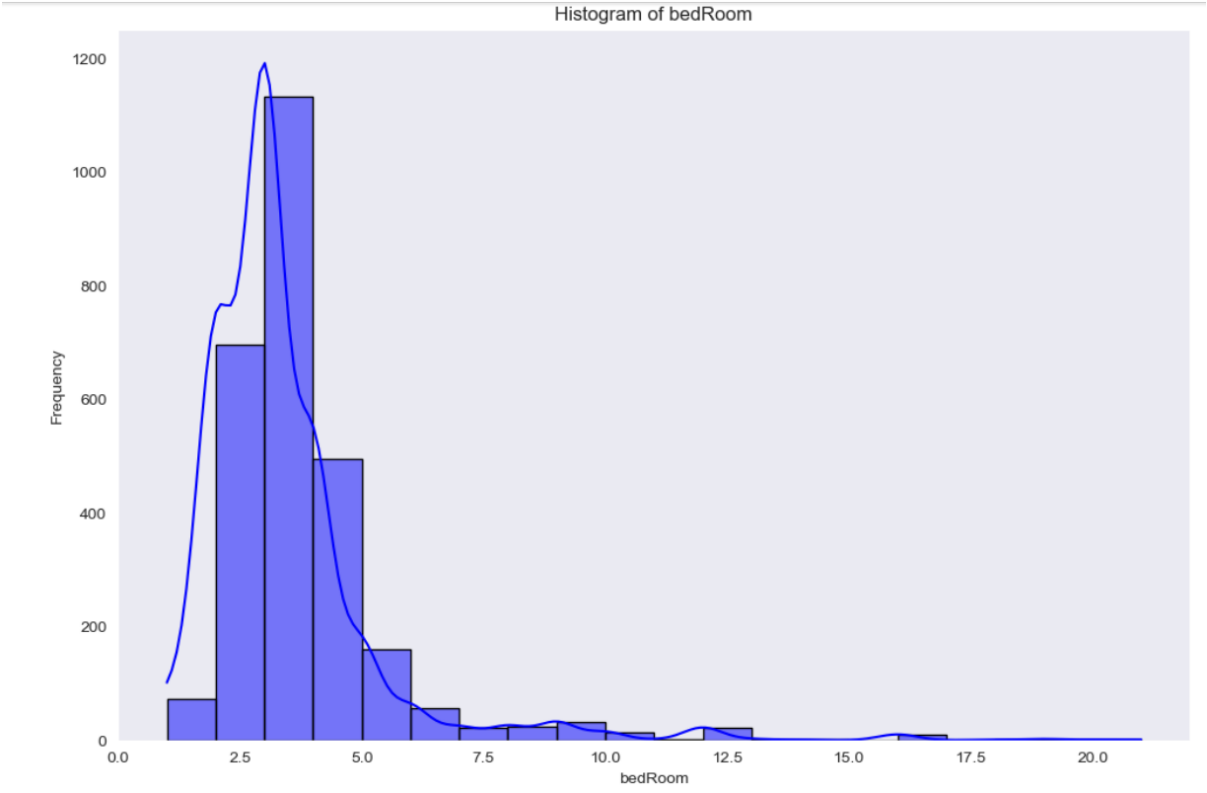

```
Skewness: 35.39166905081257
Kurtosis: 1335.3184397805203
Number of missing values: 0
```

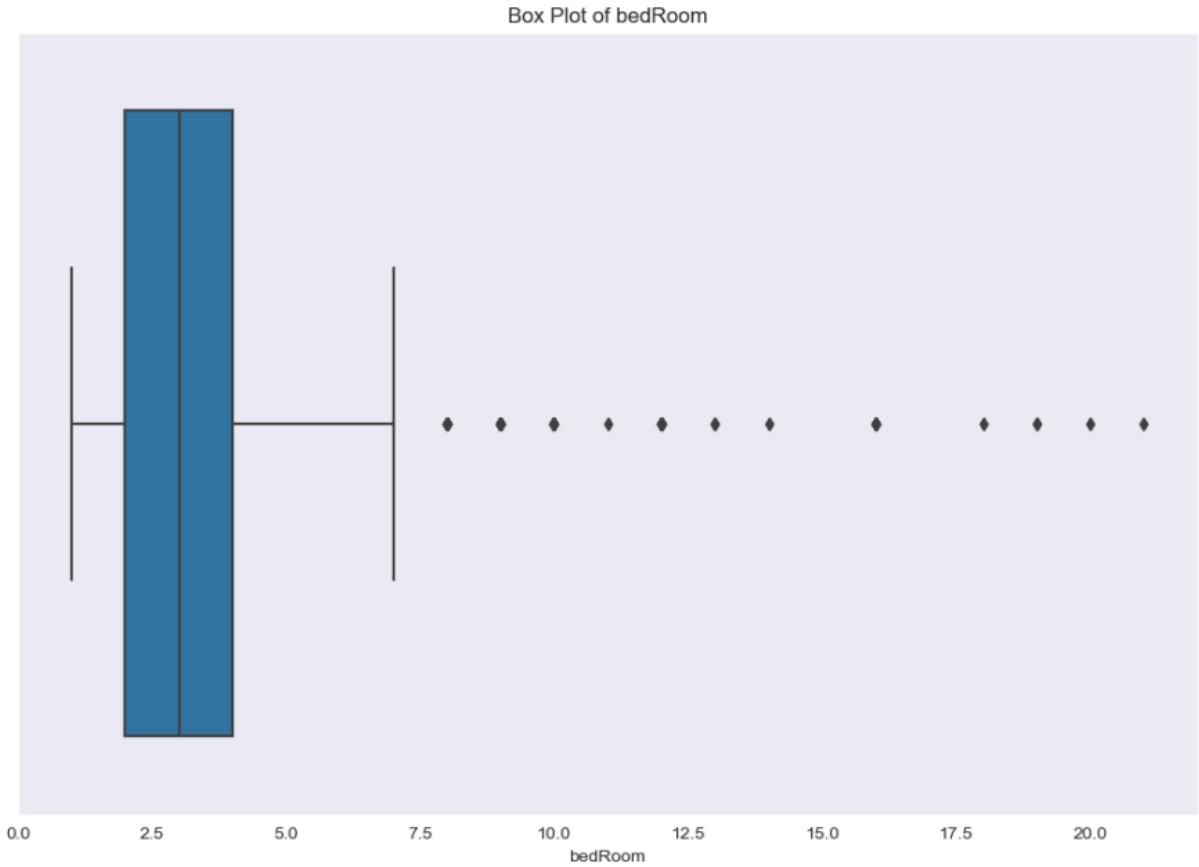
EXPLORING BEDROOM COLUMN:

```
# to explore bedroom
missing_values = abc['bedRoom'].isnull().sum()
print("Number of missing values:", missing_values)

print("Descriptive statistics of bedRoom column:")
print(abc['bedRoom'].describe())
#hostogram for bedroom
plot.figure(figsize=(12,8))
sns.histplot(abc['bedRoom'], bins=20, kde=True, color='blue',
edgecolor='black')
plot.xlabel('bedRoom')
plot.ylabel('Frequency')
plot.title('Histogram of bedRoom')
plot.show()
#boxplot for outliers of bedroom
plot.figure(figsize=(12,8))
sns.boxplot(x=abc['bedRoom'])
plot.xlabel('bedRoom')
plot.title('Box Plot of bedRoom')
plot.show()
```

```python
#to check skewness and kurtosis
skewness = abc['bedRoom'].skew()
kurtosis = abc['bedRoom'].kurtosis()
print("Skewness:",skewness)
print("Kurtosis:",kurtosis)
```

```
Descriptive statistics of bedRoom column:
count    2743.000000
mean        3.396646
std         1.924965
min         1.000000
25%         2.000000
50%         3.000000
75%         4.000000
max        21.000000
Name: bedRoom, dtype: float64
```





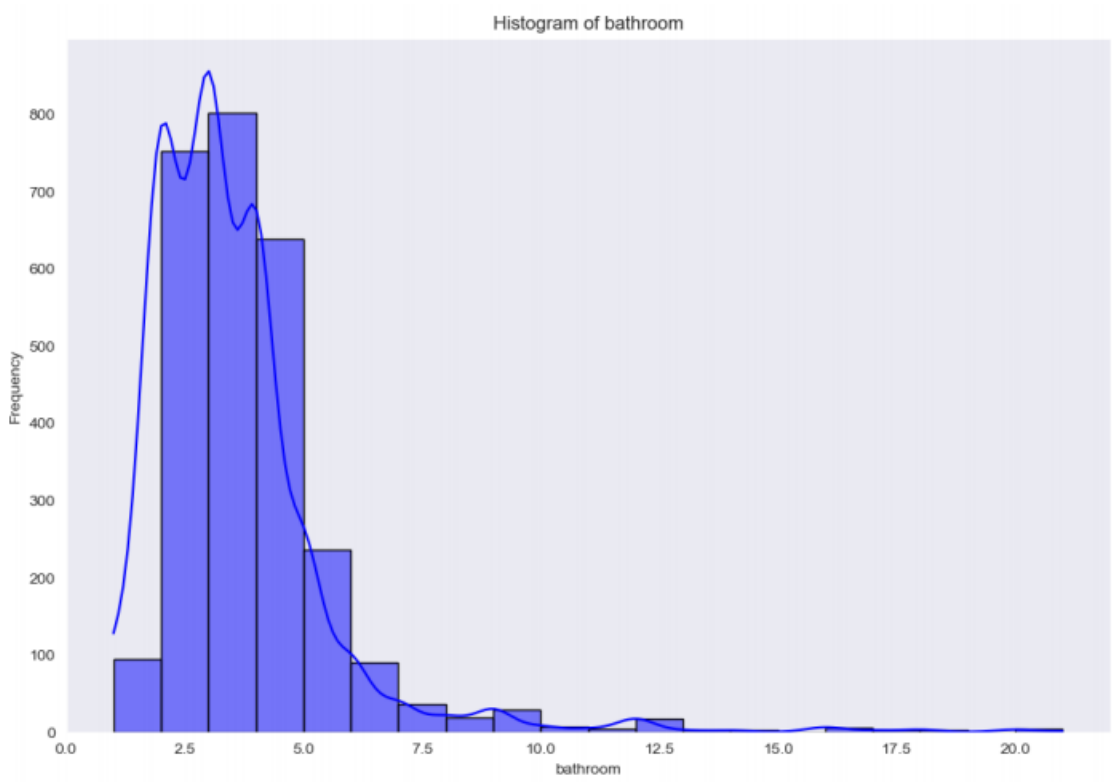

```
Skewness: 3.573969118226737
Kurtosis: 18.886627353995557
Number of missing values: 0
```

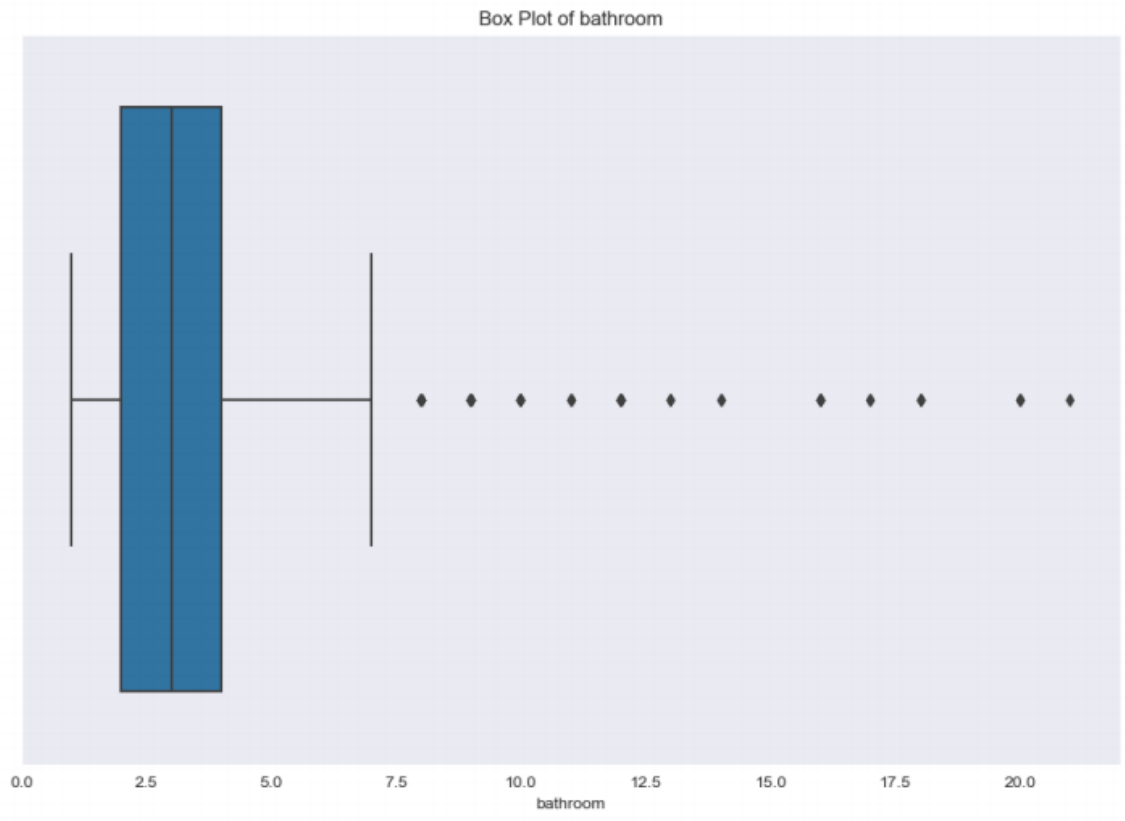
EXPLORING BATHROOM COLUMN:

```
# to explore bathroom
missing_values = abc['bathroom'].isnull().sum()
print("Number of missing values:", missing_values)

print("Descriptive statistics of bathroom column:")
print(abc['bathroom'].describe())
#hostogram for bathroom
plot.figure(figsize=(12,8))
sns.histplot(abc['bathroom'], bins=20, kde=True, color='blue',
edgecolor='black')
plot.xlabel('bathroom')
plot.ylabel('Frequency')
plot.title('Histogram of bathroom')
plot.show()
#boxplot for outliers of bathroom
plot.figure(figsize=(12,8))
sns.boxplot(x=abc['bathroom'])
plot.xlabel('bathroom')
plot.title('Box Plot of bathroom')
```

```
plot.show()
#to check skewness and kurtosis
skewness = abc['bathroom'].skew()
kurtosis = abc['bathroom'].kurtosis()
print("Skewness:",skewness)
print("Kurtosis:",kurtosis)
```

```
Descriptive statistics of bathroom column:
count    2743.000000
mean        3.491797
std         1.970013
min         1.000000
25%         2.000000
50%         3.000000
75%         4.000000
max        21.000000
Name: bathroom, dtype: float64
```



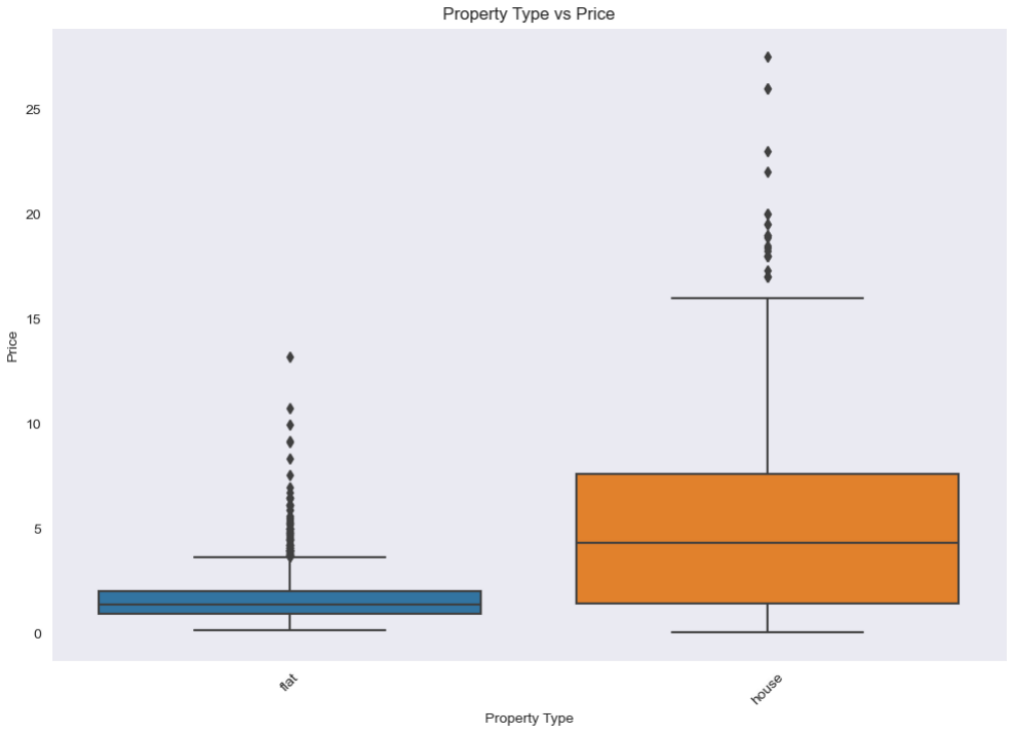



```
Skewness: 3.305127893275285
Kurtosis: 18.11944304418882
```

MULTIVARIATE ANALYSIS OF ALL COLUMNS VS TARGET COLUMN :

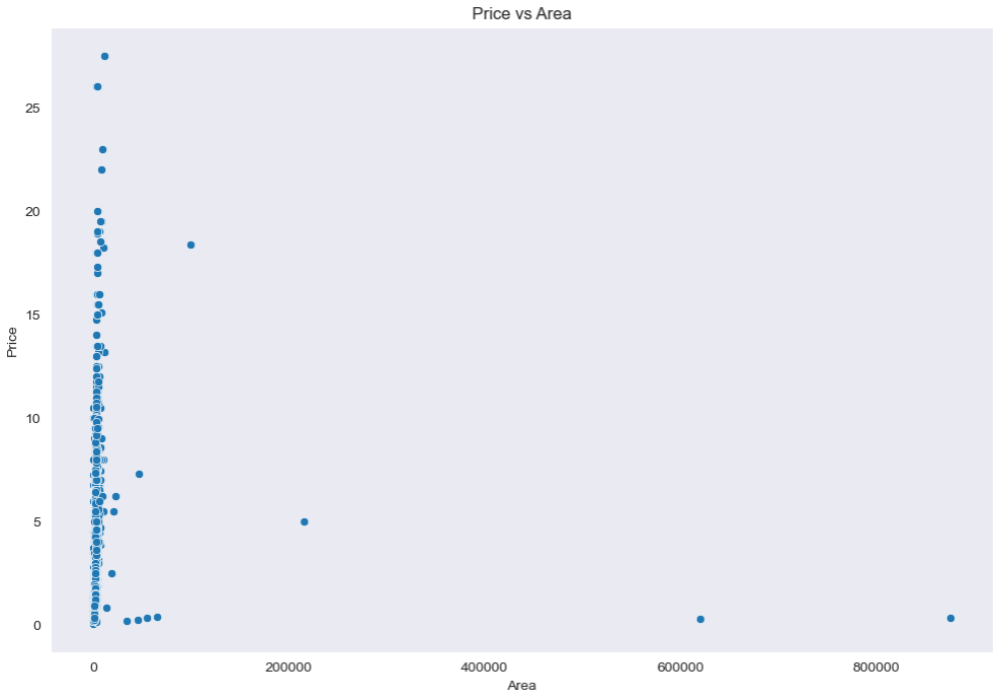
PROPERTY TYPE VS PRICE :

```
# Property type vs price
plot.figure(figsize=(12,8))
sns.boxplot(x='property_type', y='price', data=abc)
plot.title('Property Type vs Price')
plot.xlabel('Property Type')
plot.ylabel('Price')
plot.xticks(rotation=45)
plot.show()
```



PRICE vs AREA:

```
# Scatter plot between price and area
plot.figure(figsize=(12,8))
sns.scatterplot(x='area', y='price', data=abc)
plot.title('Price vs Area')
plot.xlabel('Area')
plot.ylabel('Price')
plot.show()
```



TASK 2:

MISSING VALUES HANDLING:

```
#MISSING VALUE HANDLING
missval = abc.isnull().sum()
missdist = (missval / len(abc)) * 100
print("Missing Values Distribution:")
print(missval)
print(missdist)
```

```
TASK 2
Missing Values Distribution:
property_type             0
society                   0
sector                    0
price                    14
price_per_sqft           14
area                     14
areaWithType              0
bedRoom                   0
bathroom                  0
balcony                   0
floorNum                 14
facing                  659
agePossession             0
super_built_up_area    1224
built_up_area          1469
carpet_area            1388
studyroom                 0
servantroom               0
storeroom                 0
poojaroom                 0
others                    0
furnishing_type           0
luxury_score              0
dtype: int64
property_type          0.000000
society                0.000000
sector                 0.000000
price                  0.510390
price_per_sqft         0.510390
area                   0.510390
areaWithType           0.000000
bedRoom                0.000000
bathroom               0.000000
balcony                0.000000
floorNum               0.510390
facing                24.024790
agePossession          0.000000
super_built_up_area   44.622676
built_up_area         53.554502
carpet_area           50.601531
studyroom              0.000000
servantroom            0.000000
storeroom              0.000000
poojaroom              0.000000
others                 0.000000
furnishing_type        0.000000
luxury_score           0.000000
dtype: float64
```

```
# FILLING NULL VALUES WITH MEAN OF A COLUMN
abcd=abc.fillna(value=abc['carpet_area'].mean())
abcd=abc.fillna(value=abc['built_up_area'].mean())
```

```
abcd=abc.fillna(value=abc['super_built_up_area'].mean())
abcd=abc.fillna(value=abc['price'].mean())
abcd=abc.fillna(value=abc['price_per_sqft'].mean())
```

```
#HANDLING MISSING VALUES OF CATEGORICAL COLUMNS USING MODE:

abcd["society"].replace(np.NaN, abcd["society"].mode()[0], inplace=True)
abcd["facing"].replace(np.NaN, abcd["facing"].mode()[0], inplace=True)
print("\n MISSING VALUES AFTER HANDLING:")
print(abcd.isnull().sum())
```

```
 MISSING VALUES AFTER HANDLING
property_type          0
society                0
sector                 0
price                  0
price_per_sqft         0
area                   0
areaWithType           0
bedRoom                0
bathroom               0
balcony                0
floorNum               0
facing                 0
agePossession          0
super_built_up_area    0
built_up_area          0
carpet_area            0
studyroom              0
servantroom            0
storeroom              0
poojaroom              0
others                 0
furnishing_type        0
luxury_score           0
dtype: int64
```

```
# Display initial data with missing values
print("Initial Data with Missing Values:")
print(abc.head(10))
```

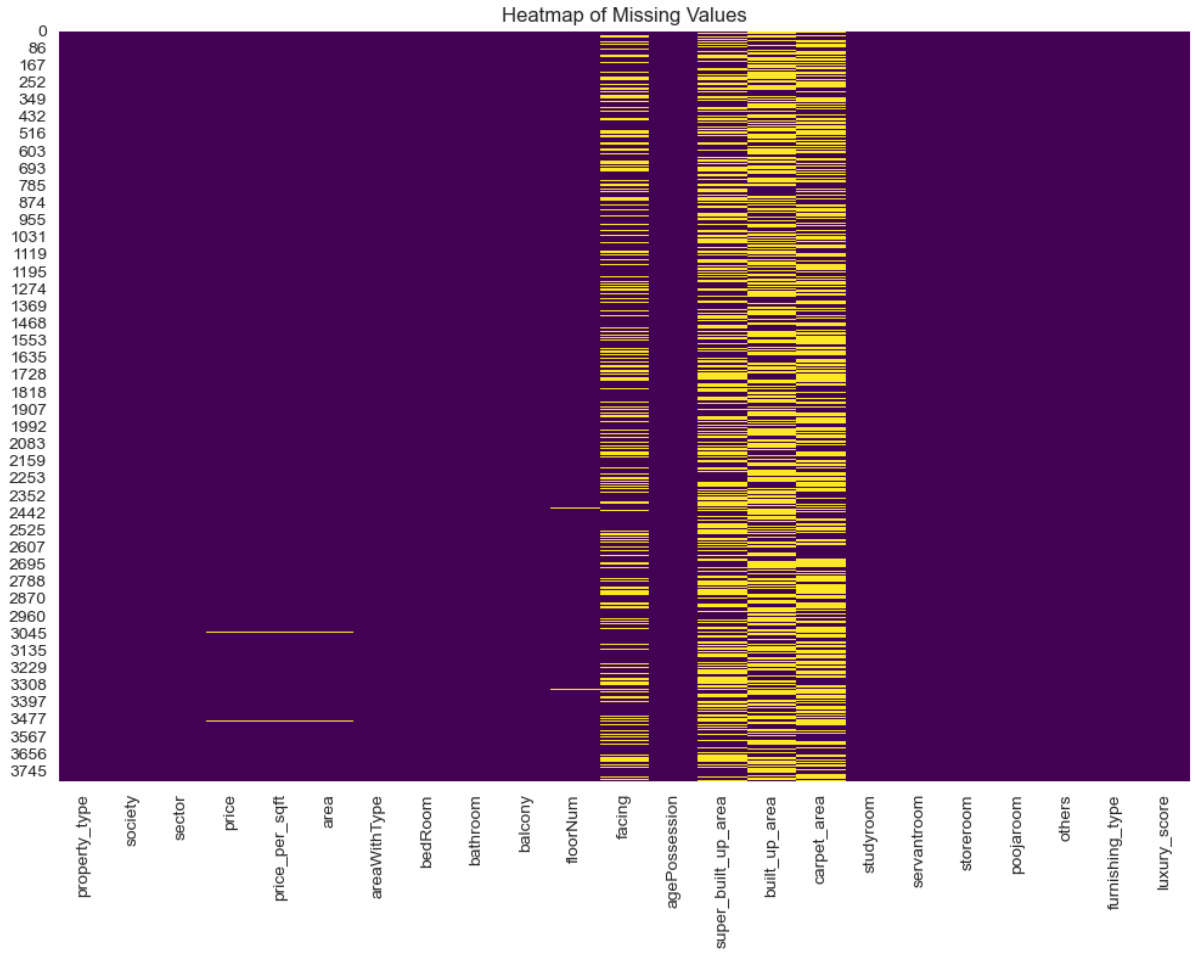
Initial Data with Missing Values:

	property_type	society	sector	price	\
0	flat	signature global park 4	sector 36	0.82	
1	flat	smart world gems	sector 89	0.95	
3	flat	breez global hill view	sohna road	0.32	
4	flat	bestech park view sanskruti	sector 92	1.60	
7	flat	experion the heartsong	sector 108	2.00	
8	flat	adani m2k oyster grande	sector 102	1.90	
9	house	independent	sector 105	1.20	
11	house	independent	sector 109	1.55	
12	flat	dlf regency park	sector 28	1.60	
13	flat	ats tourmaline	sector 109	2.25	

	price_per_sqft	area	areaWithType	\
0	7585.0	1081.0	Super Built up area 1081(100.43 sq.m.)Carpet a…	
1	8600.0	1105.0	Carpet area: 1103 (102.47 sq.m.)	
3	5470.0	585.0	Built Up area: 1000 (92.9 sq.m.)Carpet area: 5…	

```
4          8020.0  1995.0  Super Built up area 1995(185.34 sq.m.)Built Up…
7          8554.0  2338.0            Super Built up area 2338(217.21 sq.m.)
8          9105.0  2087.0            Super Built up area 1889(175.49 sq.m.)
9         10122.0  1186.0                  Plot area 1185.51(110.14 sq.m.)
11         6567.0  2360.0               Built Up area: 2360 (219.25 sq.m.)
12        14545.0  1100.0                 Carpet area: 1100 (102.19 sq.m.)
13         8704.0  2585.0            Super Built up area 2585(240.15 sq.m.)

    bedRoom  bathroom balcony  …  super_built_up_area built_up_area  \
0         3         2       2  …                1081.0           NaN
1         2         2       2  …                   NaN           NaN
3         2         2       1  …                   NaN       1000.00
4         3         4      3+  …                1995.0       1615.00
7         3         3      3+  …                2338.0           NaN
8         3         4       3  …                1889.0           NaN
9         6         2       1  …                   NaN       1185.51
11        3         2       0  …                   NaN       2360.00
12        2         2       2  …                   NaN           NaN
13        3         4       3  …                2585.0           NaN

   carpet_area  studyroom  servantroom  storeroom  poojaroom  others  \
0        650.0          0            0          0          0       0
1       1103.0          1            1          0          0       0
3        585.0          0            0          0          0       0
4       1476.0          0            1          0          0       1
7          NaN          0            1          0          0       0
8          NaN          0            1          0          0       0
9          NaN          0            0          0          0       0
11         NaN          0            0          0          0       0
12      1100.0          0            0          0          0       0
13         NaN          0            1          0          0       0

    furnishing_type  luxury_score
0                 0             8
1                 0            38
3                 0            49
4                 1           174
7                 0            95
8                 0           165
9                 0             9
11                0             0
12                0            52
13                0           101

[10 rows x 23 columns]
```

```
# Visualize missing data
sns.heatmap(abc.isnull(), cbar=False, cmap="viridis")
plot.title('Heatmap of Missing Values')
plot.show()
```



The strategies that we used for handling missing values are – deletion, mean, mode and we tried one hot encoding (to convert categorical data to numerical).

```
#Deletion
abc_deletion = abc.dropna()
print("\nData after Deletion:")
print(abc_deletion.head(10))
```

```
Data after Deletion:
   property_type                    society      sector  price  \
4           flat  bestech park view sanskruti   sector 92   1.60
17          flat                 ss the leaf   sector 85   1.09
18          flat       capital residences 360  sector 70a   1.15
34          flat     emaar mgf the palm drive   sector 66   3.50
36          flat             dlf the skycourt   sector 86   1.54
47          flat      la vida by tata housing  sector 113   3.00
57          flat                 m3m skywalk   sector 74   1.50
59          flat       experion the heartsong  sector 108   1.65
63          flat                 ss the leaf   sector 85   1.12
73          flat               dlf the ultima   sector 81   2.30

    price_per_sqft    area                                    areaWithType  \
4           8020.0  1995.0  Super Built up area 1995(185.34 sq.m.)Built Up…
17          6666.0  1635.0  Super Built up area 1640(152.36 sq.m.)Built Up…
18         11500.0  1000.0  Super Built up area 1450(134.71 sq.m.)Built Up…
34         15909.0  2200.0  Super Built up area 2200(204.39 sq.m.)Built Up…
```



```
36          9221.0  1670.0  Super Built up area 1929(179.21 sq.m.)Built Up…
47         14285.0  2100.0  Super Built up area 2691(250 sq.m.)Built Up ar…
57         12500.0  1200.0  Super Built up area 1400(130.06 sq.m.)Built Up…
59          8237.0  2003.0  Super Built up area 2003(186.08 sq.m.)Built Up…
63          6453.0  1736.0  Super Built up area 1741(161.74 sq.m.)Built Up…
73         10936.0  2103.0  Super Built up area 2103(195.38 sq.m.)Built Up…

    bedRoom  bathroom balcony  …  super_built_up_area built_up_area  \
4         3         4      3+  …                1995.0       1615.00
17        2         2       3  …                1640.0       1638.00
18        2         2       3  …                1450.0       1400.00
34        3         3       3  …                2200.0       2125.00
36        3         3      3+  …                1929.0       1780.00
47        3         3      3+  …                2691.0       2460.00
57        2         2       3  …                1400.0       1300.00
59        3         4       3  …                2003.0       1771.32
63        2         2       3  …                1741.0       1485.00
73        3         3      3+  …                2103.0       1617.00

   carpet_area  studyroom  servantroom  storeroom  poojaroom  others  \
4      1476.00          0            1          0          0       1
17     1635.00          0            0          1          0       0
18     1000.00          0            0          0          1       0
34     2000.00          0            1          1          0       0
36     1670.00          0            1          0          0       0
47     2100.00          0            1          0          0       0
57     1200.00          0            0          1          0       0
59     1302.01          1            0          0          0       0
63     1335.00          0            0          0          0       1
73     1257.00          1            1          0          1       1

    furnishing_type  luxury_score
4                 1           174
17                0           174
18                0           132
34                1           149
36                0           174
47                0           167
57                0           174
59                1            75
63                0            49
73                2            49

[10 rows x 23 columns]
```

We initially tried OneHotEncoding to convert categorical data to numerical data :

```
#ONEHOT ENCODING
# from sklearn.preprocessing import OneHotEncoder
```

```
# abcd.tail()
# abcd.dtypes
# abcd["society"].unique()
# abcd["facing"].unique()
# ohe = OneHotEncoder()
# ohe.fit_transform(abcd[["society","facing"]]).toarray()
# featurearr = ohe.fit_transform(abcd[["society","facing"]]).toarray()
# featlabels = ohe.get_feature_names_out(["society","facing"])
# np.array(featlabels).ravel()
# featlabels = np.array(featlabels).ravel()
# print(featlabels)
# pd.dataframe(featurearr, columns = featlabels)
# features = pd.dataframe(featurearr, columns = featlabels)
# print(features)
# pd.concat([abc, features], axis=1)
# abcnew = pd.concat([abc, features], axis=1)
```

TASK 3:

OUTLIERS DETECTION:

We used IQR method and Z-Score method based on the distribution of data of that particular column, z- score if the distribution is normal and IQR if otherwise.

```
print("OUTLIERS DETECTION")
# Select only numerical columns
numerical_abc = abc.select_dtypes(include=[np.number])
# Display descriptive statistics
print(numerical_abc.describe())
```

```
TASK 3
OUTLIERS DETECTION
              price  price_per_sqft           area      bedRoom     bathroom  \
count  2729.000000     2729.000000    2729.000000  2743.000000  2743.000000
```



```
mean      2.542774    13550.251008    2701.393551     3.396646     3.491797
std       2.951175    20032.737581   21131.306236     1.924965     1.970013
min       0.070000        4.000000      50.000000     1.000000     1.000000
25%       1.000000     6953.000000    1280.000000     2.000000     2.000000
50%       1.550000     9025.000000    1752.000000     3.000000     3.000000
75%       2.680000    13778.000000    2298.000000     4.000000     4.000000
max      27.500000   400000.000000  875000.000000    21.000000    21.000000

          floorNum  super_built_up_area  built_up_area    carpet_area  \
count  2729.000000          1519.000000    1274.000000    1355.000000
mean      7.075852          1927.782113    2518.458422    2055.091798
std       6.111480           685.562046   20647.942198   16810.256926
min       0.000000            89.000000       2.000000      33.000000
25%       3.000000          1534.000000    1126.250000     900.000000
50%       5.000000          1852.000000    1650.000000    1300.000000
75%      10.000000          2215.000000    2400.000000    1754.000000
max      51.000000          6926.000000  737147.000000  607936.000000

         studyroom  servantroom    storeroom    poojaroom       others  \
count  2743.000000  2743.000000  2743.000000  2743.000000  2743.000000
mean      0.207437     0.387532     0.097703     0.192490     0.110828
std       0.405545     0.487276     0.296967     0.394328     0.313976
min       0.000000     0.000000     0.000000     0.000000     0.000000
25%       0.000000     0.000000     0.000000     0.000000     0.000000
50%       0.000000     0.000000     0.000000     0.000000     0.000000
75%       0.000000     1.000000     0.000000     0.000000     0.000000
max       1.000000     1.000000     1.000000     1.000000     1.000000

       furnishing_type  luxury_score
count      2743.000000   2743.000000
mean          0.401750     77.372220
std           0.589004     53.219729
min           0.000000      0.000000
25%           0.000000     38.000000
50%           0.000000     66.000000
75%           1.000000    118.000000
max           2.000000    174.000000
```

Z-Score Method:

```
# Z-score method
z_score = np.abs(stats.zscore(numerical_abc))
outlier_z = (z_score > 3).any(axis=1)
print(f"Outliers detected by Z-score:\n{numerical_abc[outlier_z]}")
```

```
Outliers detected by Z-score:
      price  price_per_sqft    area  bedRoom  bathroom  floorNum  \
17     1.09          6666.0  1635.0        2         2       9.0
19     5.50         38194.0  1440.0       18        18       4.0
20     1.77          7350.0  2408.0        3         4       7.0
25     9.00         27778.0  3240.0        8         5       3.0
26     7.00         28283.0  2475.0        7         7       3.0
…       …              …       …        …         …
3752   1.78          7392.0  2408.0        3         4       6.0
3791   6.44         26502.0  2430.0        4         5       3.0
3794   8.00         26298.0  3042.0        9         9       4.0
```



```
3799   6.00          9634.0  6228.0        5         5       2.0
3801  15.50         28233.0  5490.0        5         6       3.0

      super_built_up_area  built_up_area  carpet_area  studyroom  servantroom  \
17                 1640.0         1638.0       1635.0          0            0
19                    NaN         1440.0          NaN          0            1
20                 2408.0            NaN          NaN          0            0
25                    NaN         3240.0          NaN          0            1
26                    NaN         2475.0          NaN          1            1
…                      …              …            …          …            …
3752               2408.0            NaN          NaN          0            0
3791                  NaN         2430.0          NaN          1            1
3794                  NaN         3042.0          NaN          1            1
3799                  NaN         6228.0          NaN          1            1
3801                  NaN         5490.0          NaN          1            1

      storeroom  poojaroom  others  furnishing_type  luxury_score
17            1          0       0                0           174
19            0          0       0                2            70
20            1          0       0                0           107
25            1          1       0                1            26
26            1          1       0                2           151
…           …          …       …                …           …
3752          1          0       0                0           113
3791          1          1       0                0           138
3794          1          1       0                2           110
3799          1          1       0                0           160
3801          1          1       0                0            76

[313 rows x 16 columns]
```

IQR Method:

```
# IQR method
Q1 = numerical_abc.quantile(0.25)
Q3 = numerical_abc.quantile(0.75)
IQR = Q3 - Q1
outlier_iqr = ((numerical_abc < (Q1 - 1.5 * IQR)) | (numerical_abc > (Q3 + 1.5
* IQR))).any(axis=1)
print(f"Outliers detected by IQR:\n{numerical_abc[outlier_iqr]}")
```

```
# Visualization of distributions and outliers
fig, ax = plot.subplots(len(numerical_abc.columns), 2, figsize=(16, 4 *
len(numerical_abc.columns)))
for i, feature in enumerate(numerical_abc.columns):
 sns.histplot(numerical_abc[feature], kde=True, ax=ax[i, 0])
 ax[i, 0].set_title(f'Histogram of {feature}')
 sns.boxplot(x=numerical_abc[feature], ax=ax[i, 1])
 ax[i, 1].set_title(f'Box plot of {feature}')
plot.tight_layout()
plot.show()
```

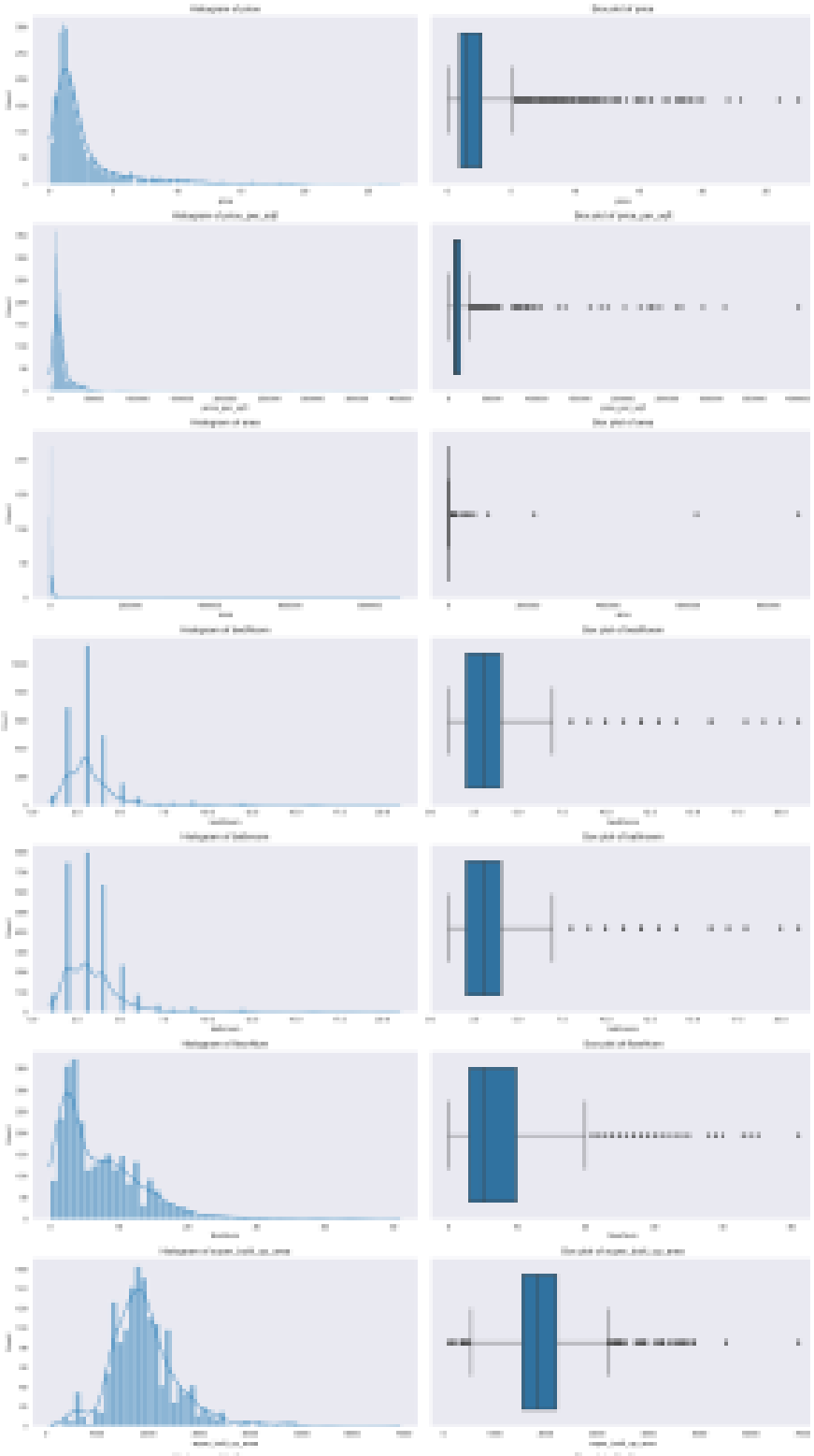
```
Outliers detected by IQR:
      price  price_per_sqft    area  bedRoom  bathroom  floorNum  \
1      0.95          8600.0  1105.0        2         2       4.0
4      1.60          8020.0  1995.0        3         4      10.0
14     2.77         14025.0  1975.0        4         4       3.0
15     1.20         14184.0   846.0        4         4       2.0
17     1.09          6666.0  1635.0        2         2       9.0
…       …               …       …        …       …         …
3791   6.44         26502.0  2430.0        4         5       3.0
3794   8.00         26298.0  3042.0        9         9       4.0
3795   0.87          5965.0  1459.0        2         2      10.0
3799   6.00          9634.0  6228.0        5         5       2.0
3801  15.50         28233.0  5490.0        5         6       3.0

      super_built_up_area  built_up_area  carpet_area  studyroom  servantroom  \
1                     NaN            NaN       1103.0          1            1
4                  1995.0         1615.0       1476.0          0            1
14                    NaN            NaN       1975.0          1            1
```

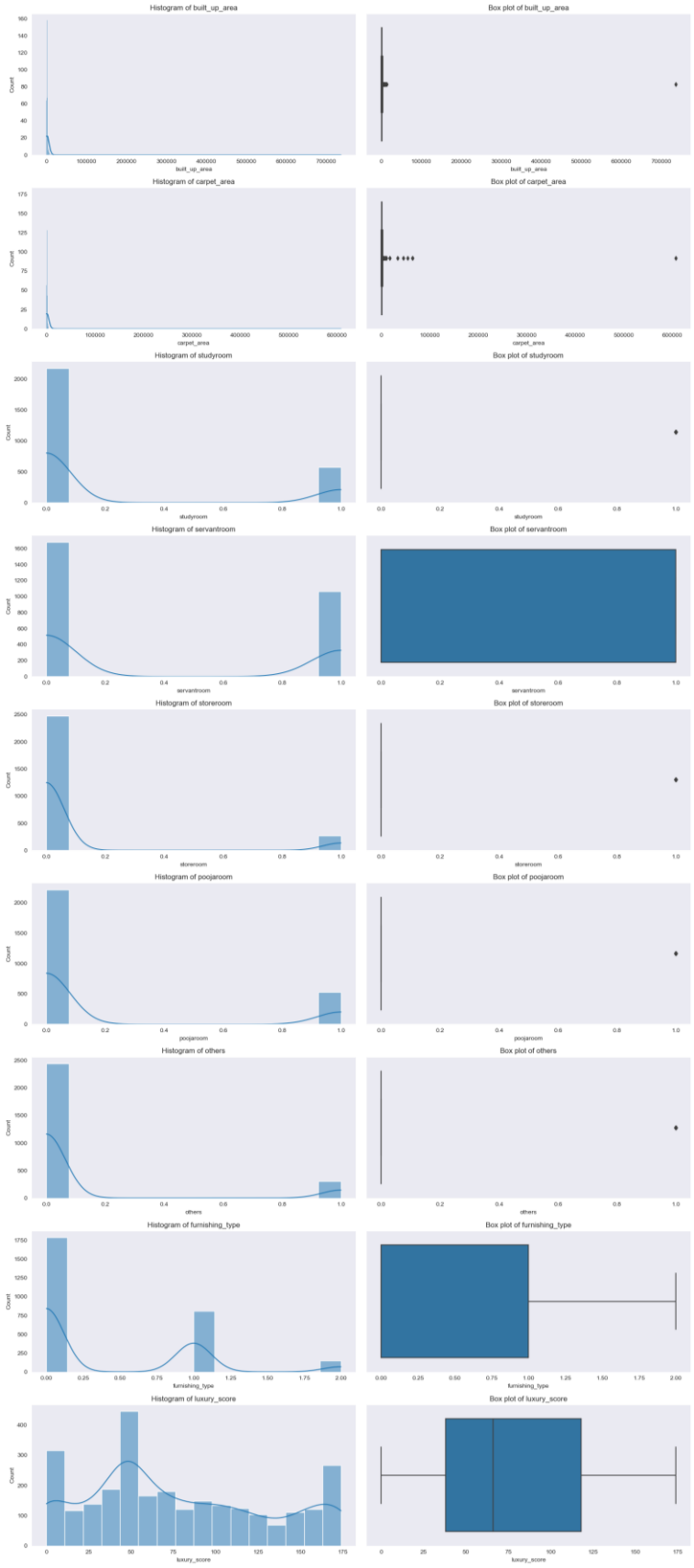


```
15                    NaN          846.0          NaN          0            0
17                 1640.0         1638.0       1635.0          0            0
…                     …              …            …          …            …
3791                  NaN         2430.0          NaN          1            1
3794                  NaN         3042.0          NaN          1            1
3795               1457.0            NaN        849.0          1            0
3799                  NaN         6228.0          NaN          1            1
3801                  NaN         5490.0          NaN          1            1

      storeroom  poojaroom  others  furnishing_type  luxury_score
1             0          0       0                0            38
4             0          0       1                1           174
14            0          0       0                2           174
15            0          1       0                1             7
17            1          0       0                0           174
…         …        …       …             …          …
3791          1          1       0                0           138
3794          1          1       0                2           110
3795          0          0       0                0            72
3799          1          1       0                0           160
3801          1          1       0                0            76

[1430 rows x 16 columns]
```

```
# Visualization of distributions and outliers
fig, ax = plot.subplots(len(numerical_abc.columns), 2, figsize=(16, 4 *
len(numerical_abc.columns)))
for i, feature in enumerate(numerical_abc.columns):
 sns.histplot(numerical_abc[feature], kde=True, ax=ax[i, 0])
 ax[i, 0].set_title(f'Histogram of {feature}')
 sns.boxplot(x=numerical_abc[feature], ax=ax[i, 1])
 ax[i, 1].set_title(f'Box plot of {feature}')
```

```
plot.tight_layout()
plot.show()
```





Our initial code for z-score:

```
#OUTLIERS DETECTION FOR NORMALLY DISTRUBUTED COLUMN THROUGH ZSCORE
#%matplotlib inline
#matplotlib.rcParams['figure.figsize'] = (12,8)
#bedroom

#plot.hist(abcd.bedRoom, bins=20, rwidth=0.8)
#plot.xlabel('bedRoom')
#plot.ylabel('count')
#plot.show()

#rngg = np.arange(abcd.bedRoom.min(), abcd.bedRoom.max(), 0.1)
#plot.plot(rngg, norm.pdf(rngg, abcd.bedRoom.mean(), abcd.bedRoom.std()))
#upperlim = abcd.bedRoom.mean() + 3*abcd.bedRoom.std()
#lowerlim = abcd.bedRoom.mean() - 3*abcd.bedRoom.std()
#abcd[(abcd.bedRoom>upperlim)|(abcd.bedRoom<lowerlim)]
#abcde=abcd[(abcd.bedRoom>upperlim)|(abcd.bedRoom<lowerlim)]
#abcd['zscore'] = (abcd.bedRoom - abcd.bedRoom.mean())/abcd.bedRoom.std()
#bathroom
#plot.hist(abcd.bathroom, bins=20, rwidth=0.8)
#plot.xlabel('bathroom')
#plot.ylabel('count')
#plot.show()

#rngg = np.arange(abcd.bathroom.min(), abcd.bathroom.max(), 0.1)
#plot.plot(rngg, norm.pdf(rngg, abcd.bathroom.mean(), abcd.bathroom.std()))
#upperlim = abcd.bathroom.mean() + 3*abcd.bathroom.std()
#lowerlim = abcd.bathroom.mean() - 3*abcd.bathroom.std()
#abcd[(abcd.bathroom>upperlim)|(abcd.bathroom<lowerlim)]
#abcde=abcd[(abcd.bathroom>upperlim)|(abcd.bathroom<lowerlim)]
#abcd['zscore'] = (abcd.bathroom - abcd.bathroom.mean())/abcd.bathroom.std()

# #age possession
# plot.hist(abcd.agePossession, bins=20, rwidth=0.8)
# plot.xlabel('agePossession')
# plot.ylabel('count')
# plot.show()

# rngg = np.arange(abcd.agePossession.min(), abcd.agePossession.max(), 0.1)
# plot.plot(rngg, norm.pdf(rngg, abcd.agePossession.mean(),
abcd.agePossession.std()))
# upperlim = abcd.agePossession.mean() + 3*abcd.agePossession.std()
# lowerlim = abcd.agePossession.mean() - 3*abcd.bedRoom.std()
# abcd[(abcd.agePossession>upperlim)|(abcd.agePossession<lowerlim)]
# abcde=abcd[(abcd.agePossession>upperlim)|(abcd.agePossession<lowerlim)]
# abcd['zscore'] = (abcd.agePossession -
abcd.agePossession.mean())/abcd.agePossession.std()
```

TASK 4:

OUTLIERS HANDLING:

```
print("OUTLIERS HANDLING")
# Select only numerical columns
numerical_abc = abc.select_dtypes(include=[np.number])
# Display initial descriptive statistics
print("Initial Descriptive Statistics:")
print(numerical_abc.describe())
```

```
TASK 4
OUTLIERS HANDLING
Initial Descriptive Statistics:
              price  price_per_sqft           area      bedRoom     bathroom  \
count  2729.000000     2729.000000    2729.000000  2743.000000  2743.000000
mean      2.542774    13550.251008    2701.393551     3.396646     3.491797
std       2.951175    20032.737581   21131.306236     1.924965     1.970013
min       0.070000        4.000000      50.000000     1.000000     1.000000
25%       1.000000     6953.000000    1280.000000     2.000000     2.000000
50%       1.550000     9025.000000    1752.000000     3.000000     3.000000
75%       2.680000    13778.000000    2298.000000     4.000000     4.000000
max      27.500000   400000.000000  875000.000000    21.000000    21.000000

          floorNum  super_built_up_area  built_up_area    carpet_area  \
count  2729.000000          1519.000000    1274.000000    1355.000000
mean      7.075852          1927.782113    2518.458422    2055.091798
std       6.111480           685.562046   20647.942198   16810.256926
min       0.000000            89.000000       2.000000      33.000000
25%       3.000000          1534.000000    1126.250000     900.000000
50%       5.000000          1852.000000    1650.000000    1300.000000
75%      10.000000          2215.000000    2400.000000    1754.000000
max      51.000000          6926.000000  737147.000000  607936.000000

         studyroom  servantroom    storeroom    poojaroom       others  \
count  2743.000000  2743.000000  2743.000000  2743.000000  2743.000000
mean      0.207437     0.387532     0.097703     0.192490     0.110828
std       0.405545     0.487276     0.296967     0.394328     0.313976
min       0.000000     0.000000     0.000000     0.000000     0.000000
25%       0.000000     0.000000     0.000000     0.000000     0.000000
50%       0.000000     0.000000     0.000000     0.000000     0.000000
75%       0.000000     1.000000     0.000000     0.000000     0.000000
max       1.000000     1.000000     1.000000     1.000000     1.000000

       furnishing_type  luxury_score
count      2743.000000   2743.000000
mean          0.401750     77.372220
std           0.589004     53.219729
min           0.000000      0.000000
25%           0.000000     38.000000
50%           0.000000     66.000000
75%           1.000000    118.000000
max           2.000000    174.000000
```

```
# Z-score method to identify outliers
z_score = np.abs(stats.zscore(numerical_abc))
outlier_z = (z_score > 3).any(axis=1)
print(f"\nOutliers detected by Z-score:\n{numerical_abc[outlier_z]}")
```

```
Outliers detected by Z-score:
      price  price_per_sqft    area  bedRoom  bathroom  floorNum  \
```



```
17     1.09          6666.0  1635.0        2         2       9.0
19     5.50         38194.0  1440.0       18        18       4.0
20     1.77          7350.0  2408.0        3         4       7.0
25     9.00         27778.0  3240.0        8         5       3.0
26     7.00         28283.0  2475.0        7         7       3.0
…       …              …       …         …         …
3752   1.78          7392.0  2408.0        3         4       6.0
3791   6.44         26502.0  2430.0        4         5       3.0
3794   8.00         26298.0  3042.0        9         9       4.0
3799   6.00          9634.0  6228.0        5         5       2.0
3801  15.50         28233.0  5490.0        5         6       3.0

      super_built_up_area  built_up_area  carpet_area  studyroom  servantroom  \
17                 1640.0         1638.0       1635.0          0            0
19                    NaN         1440.0          NaN          0            1
20                 2408.0            NaN          NaN          0            0
25                    NaN         3240.0          NaN          0            1
26                    NaN         2475.0          NaN          1            1
…                      …              …            …          …            …
3752               2408.0            NaN          NaN          0            0
3791                  NaN         2430.0          NaN          1            1
3794                  NaN         3042.0          NaN          1            1
3799                  NaN         6228.0          NaN          1            1
3801                  NaN         5490.0          NaN          1            1

      storeroom  poojaroom  others  furnishing_type  luxury_score
17            1          0       0                0           174
19            0          0       0                2            70
20            1          0       0                0           107
25            1          1       0                1            26
26            1          1       0                2           151
…           …          …       …                …             …
3752          1          0       0                0           113
3791          1          1       0                0           138
3794          1          1       0                2           110
3799          1          1       0                0           160
3801          1          1       0                0            76

[313 rows x 16 columns]
```

```
# IQR method to identify outliers
Q1 = numerical_abc.quantile(0.25)
```

```
Q3 = numerical_abc.quantile(0.75)
IQR = Q3 - Q1
outlier_iqr = ((numerical_abc < (Q1 - 1.5 * IQR)) | (numerical_abc > (Q3 + 1.5
* IQR))).any(axis=1)
print(f"\nOutliers detected by IQR:\n{numerical_abc[outlier_iqr]}")
```

```
Outliers detected by IQR:
      price  price_per_sqft    area  bedRoom  bathroom  floorNum  \
1      0.95          8600.0  1105.0        2         2       4.0
4      1.60          8020.0  1995.0        3         4      10.0
14     2.77         14025.0  1975.0        4         4       3.0
15     1.20         14184.0   846.0        4         4       2.0
17     1.09          6666.0  1635.0        2         2       9.0
…       …              …       …        …         …         …
```

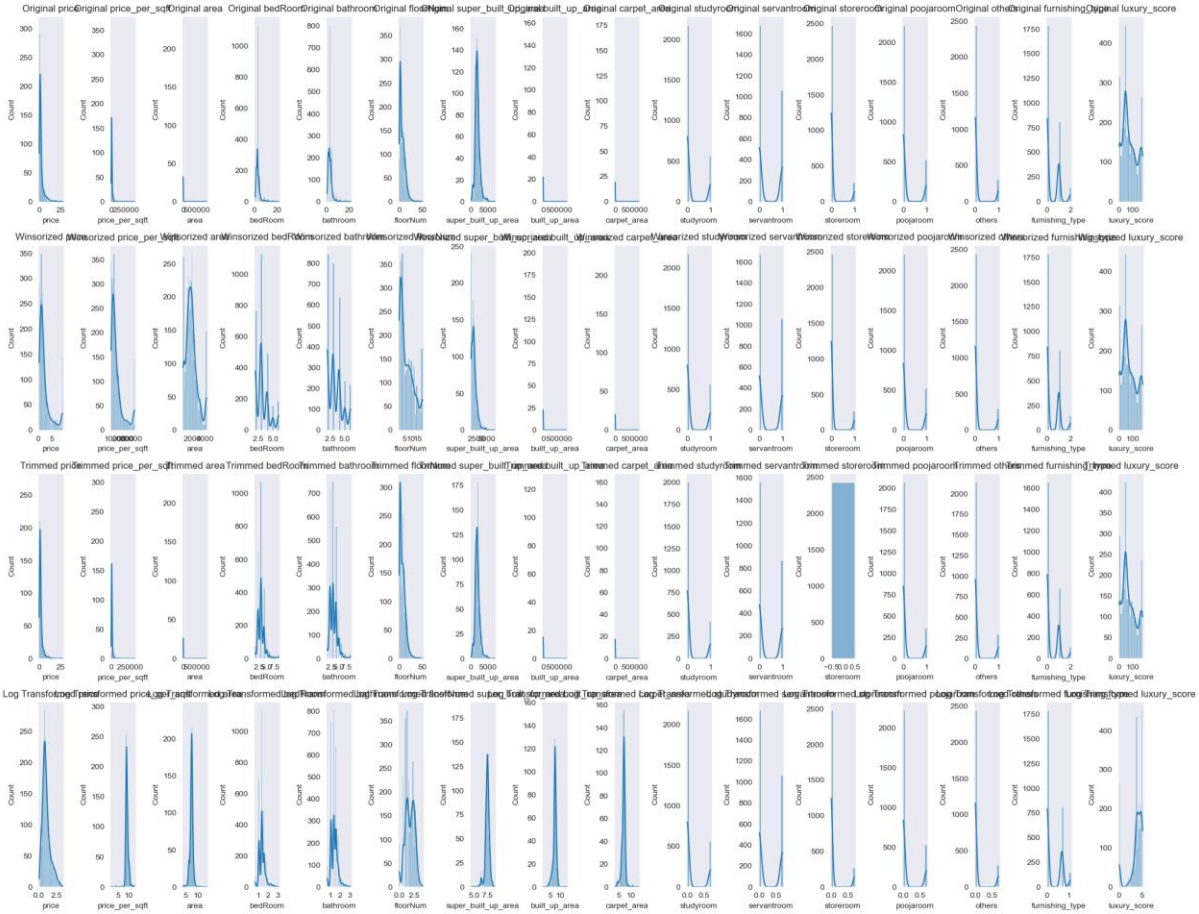


```
3791   6.44         26502.0  2430.0        4         5       3.0
3794   8.00         26298.0  3042.0        9         9       4.0
3795   0.87          5965.0  1459.0        2         2      10.0
3799   6.00          9634.0  6228.0        5         5       2.0
3801  15.50         28233.0  5490.0        5         6       3.0

      super_built_up_area  built_up_area  carpet_area  studyroom  servantroom  \
1                     NaN            NaN       1103.0          1            1
4                  1995.0         1615.0       1476.0          0            1
14                    NaN            NaN       1975.0          1            1
15                    NaN          846.0          NaN          0            0
17                 1640.0         1638.0       1635.0          0            0
…                     …              …            …          …            …
3791                  NaN         2430.0          NaN          1            1
3794                  NaN         3042.0          NaN          1            1
3795               1457.0            NaN        849.0          1            0
3799                  NaN         6228.0          NaN          1            1
3801                  NaN         5490.0          NaN          1            1

      storeroom  poojaroom  others  furnishing_type  luxury_score
1             0          0       0                0            38
4             0          0       1                1           174
14            0          0       0                2           174
15            0          1       0                1             7
17            1          0       0                0           174
…         …          …       …              …          …
3791          1          1       0                0           138
3794          1          1       0                2           110
3795          0          0       0                0            72
3799          1          1       0                0           160
3801          1          1       0                0            76

[1430 rows x 16 columns]
```

The three strategies that we used to handle outliers are – Winsorization, trimming, transformation.

```
#STRATERGY 1
winsorized_abc = numerical_abc.apply(lambda x: winsorize(x, limits=[0.05,
0.05]))
# Strategy 2: Trimming
trimmed_abc = numerical_abc[~outlier_z]
# Strategy 3: Log Transformation (example for one feature, can be applied to
all if needed)
transformed_abc = numerical_abc.apply(lambda x: np.log1p(x))

# Visualize the distributions and outliers before and after handling
fig, axs = plot.subplots(4, len(numerical_abc.columns), figsize=(20, 16))
for i, feature in enumerate(numerical_abc.columns):
 sns.histplot(numerical_abc[feature], kde=True, ax=axs[0, i])
 axs[0, i].set_title(f'Original {feature}')

 sns.histplot(winsorized_abc[feature], kde=True, ax=axs[1, i])
 axs[1, i].set_title(f'Winsorized {feature}')

 sns.histplot(trimmed_abc[feature], kde=True, ax=axs[2, i])
 axs[2, i].set_title(f'Trimmed {feature}')

 sns.histplot(transformed_abc[feature], kde=True, ax=axs[3, i])
 axs[3, i].set_title(f'Log Transformed {feature}')
plot.tight_layout()
plot.show()
```

DISTRIBUTIONS AND OUTLIERS BEFORE AND AFTER HANDLING:



```
# Function to calculate metrics for evaluating impact
def calculatemetrics(data, title):
 metrics = pd.DataFrame({
 'Mean': data.mean(),
 'Std Dev': data.std(),
 'Skewness': data.skew(),
 'Kurtosis': data.apply(lambda x: stats.kurtosis(x))
 })
 print(f'\n{title}')
 print(metrics)
# Evaluate the impact of outlier handling
print("\nMetrics Before Handling Outliers:")
calculatemetrics(numerical_abc, 'Original Data Metrics')
print("\nMetrics After Winsorization:")
calculatemetrics(winsorized_abc, 'Winsorized Data Metrics')
print("\nMetrics After Trimming:")
calculatemetrics(trimmed_abc, 'Trimmed Data Metrics')
print("\nMetrics After Log Transformation:")
calculatemetrics(transformed_abc, 'Log Transformed Data Metrics')
```

Metrics Before Handling Outliers:

Original Data Metrics

	Mean	Std Dev	Skewness	Kurtosis
price	2.542774	2.951175	3.259684	NaN
price_per_sqft	13550.251008	20032.737581	10.075073	NaN
area	2701.393551	21131.306236	35.391669	NaN
bedRoom	3.396646	1.924965	3.573969	18.850031
bathroom	3.491797	1.970013	3.305128	18.084245
floorNum	7.075852	6.111480	1.664099	NaN
super_built_up_area	1927.782113	685.562046	1.203216	NaN
built_up_area	2518.458422	20647.942198	35.434849	NaN
carpet_area	2055.091798	16810.256926	34.748794	NaN
studyroom	0.207437	0.405545	1.443865	0.082468
servantroom	0.387532	0.487276	0.461957	-1.786829
storeroom	0.097703	0.296967	2.711346	5.343357
poojaroom	0.192490	0.394328	1.560806	0.433450
others	0.110828	0.313976	2.480806	4.147668
furnishing_type	0.401750	0.589004	1.167388	0.343298
luxury_score	77.372220	53.219729	0.363170	-0.986924

Metrics After Winsorization:

Winsorized Data Metrics

	Mean	Std Dev	Skewness	Kurtosis
price	2.395680	2.236050	1.806248	2.417591
price_per_sqft	12036.445133	7770.454684	1.723186	2.155577
area	1882.041560	917.621581	0.866093	0.533601
bedRoom	3.224207	1.124152	0.963238	0.361961
bathroom	3.337951	1.223362	0.673864	-0.435727
floorNum	6.920525	5.313603	0.804126	-0.438282
super_built_up_area	1962.782930	631.477308	1.812154	NaN
built_up_area	2544.620689	20645.164189	35.445390	NaN
carpet_area	2075.014964	16808.233511	34.757820	NaN
studyroom	0.207437	0.405545	1.443865	0.082468
servantroom	0.387532	0.487276	0.461957	-1.786829
storeroom	0.097703	0.296967	2.711346	5.343357
poojaroom	0.192490	0.394328	1.560806	0.433450
others	0.110828	0.313976	2.480806	4.147668
furnishing_type	0.401750	0.589004	1.167388	0.343298
luxury_score	77.372220	53.219729	0.363170	-0.986924

Metrics After Trimming:

Trimmed Data Metrics

	Mean	Std Dev	Skewness	Kurtosis
price	2.141850	2.312878	3.986094	NaN
price_per_sqft	11913.573317	15656.447531	11.036487	NaN
area	2666.931433	22325.286962	33.802623	NaN
bedRoom	3.111523	1.231997	1.784819	5.526982
bathroom	3.209053	1.313984	1.182991	2.655949
floorNum	7.349298	6.258988	1.604544	NaN
super_built_up_area	1925.340575	687.500057	1.193075	NaN
built_up_area	2513.330006	22774.379171	32.180282	NaN
carpet_area	2096.264416	17516.876384	33.369894	NaN
studyroom	0.175309	0.380309	1.708919	0.916800
servantroom	0.357202	0.479274	0.596386	-1.644763
storeroom	0.000000	0.000000	0.000000	NaN
poojaroom	0.148971	0.356133	1.972957	1.887756
others	0.119753	0.324739	2.343788	3.486560
furnishing_type	0.363374	0.568176	1.297497	0.690644
luxury_score	75.094650	52.933424	0.432061	-0.899865

Metrics After Log Transformation:

Log Transformed Data Metrics

	Mean	Std Dev	Skewness	Kurtosis
price	1.068912	0.563501	1.126075	NaN

price_per_sqft	9.231233	0.684857	-0.855867	NaN
area	7.409788	0.672240	0.222323	NaN
bedRoom	1.418422	0.328361	1.144526	3.258734
bathroom	1.434988	0.347730	0.772225	2.012752
floorNum	1.812529	0.770171	-0.184259	NaN
super_built_up_area	7.497767	0.390533	-1.421860	NaN
built_up_area	7.283939	0.902704	-1.116387	NaN
carpet_area	7.079669	0.740400	-0.097632	NaN
studyroom	0.143784	0.281103	1.443865	0.082468
servantroom	0.268617	0.337754	0.461957	-1.786829
storeroom	0.067723	0.205842	2.711346	5.343357
poojaroom	0.133424	0.273327	1.560806	0.433450
others	0.076820	0.217631	2.480806	4.147668
furnishing_type	0.263160	0.370020	0.842614	-0.910015
luxury_score	3.849022	1.421740	-1.798791	2.407955