# 3uvc9jr7q

September 6, 2023

[1]:

```
#NYSA SINGH
#21BIT0376
```

[2]:

```
#Importing Libraries

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

[8]:

```
#Loading the Dataset

df = pd.read_csv("D:/Vit/SmartBridge/Assignment 2/House Price India.csv")
```

[9]:

```
df.head()
```

[9]:

```
           id   Date  number_of_bedrooms  number of bathrooms  living area  \
0  6762810145  42491                   5                 2.50         3650
1  6762810635  42491                   4                 2.50         2920
2  6762810998  42491                   5                 2.75         2910
3  6762812605  42491                   4                 2.50         3310
4  6762812919  42491                   3                 2.00         2710

   lot_area  number of floors  waterfront present  number of views  \
0      9050               2.0                   0                4
1      4000               1.5                   0                0
2      9480               1.5                   0                0
3     42998               2.0                   0                0
4      4500               1.5                   0                0

   condition of the house  …  Built Year  Renovation Year  Postal Code  \
0                       5  …        1921                0       122003
1                       5  …        1909                0       122004
2                       3  …        1939                0       122004
3                       3  …        2001                0       122005
4                       4  …        1929                0       122006
```

```
   Lattitude  Longitude  living_area_renov  lot_area_renov  \
0    52.8645   -114.557               2880            5400
1    52.8878   -114.470               2470            4000
2    52.8852   -114.468               2940            6600
3    52.9532   -114.321               3350           42847
4    52.9047   -114.485               2060            4500

   Number of schools nearby  Distance from the airport    Price
0                         2                         58  2380000
1                         2                         51  1400000
2                         1                         53  1200000
3                         3                         76   838000
4                         1                         51   805000

[5 rows x 23 columns]
```

**Performing Visualizations

1) Univariate analysis

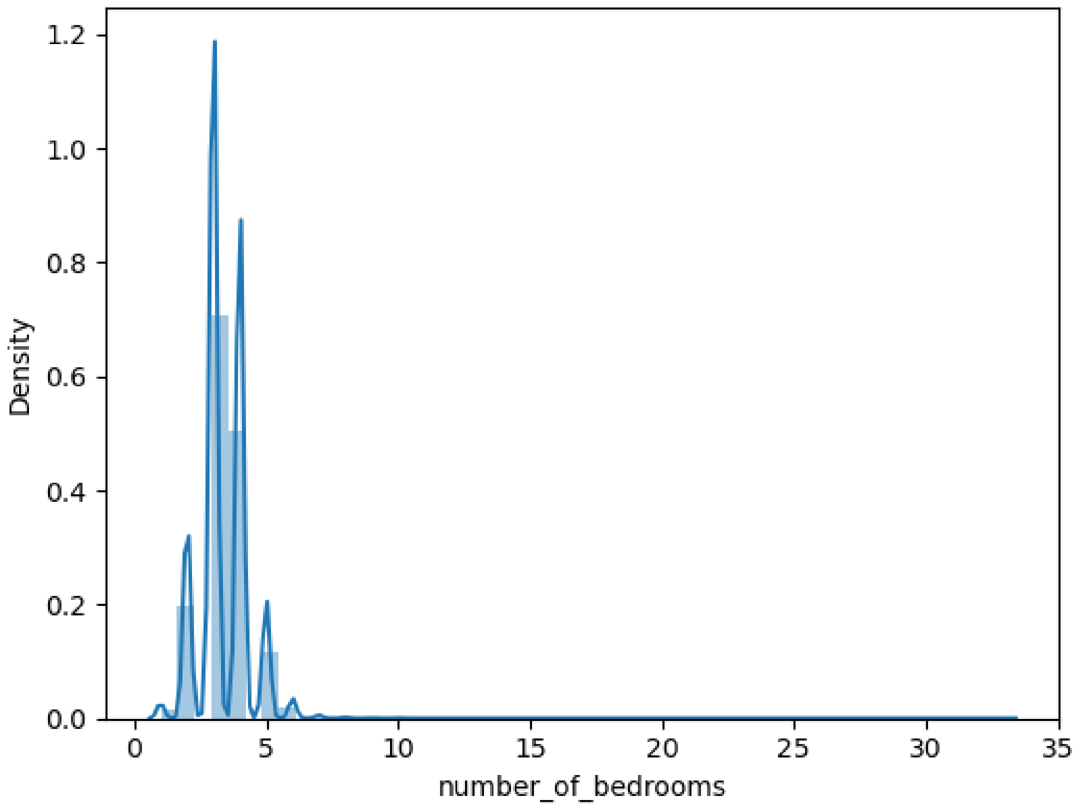
[10]:

```
#1) displot (on number of bedrooms)

sns.distplot(df.number_of_bedrooms)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for
histograms).
  warnings.warn(msg, FutureWarning)
```

[10]:

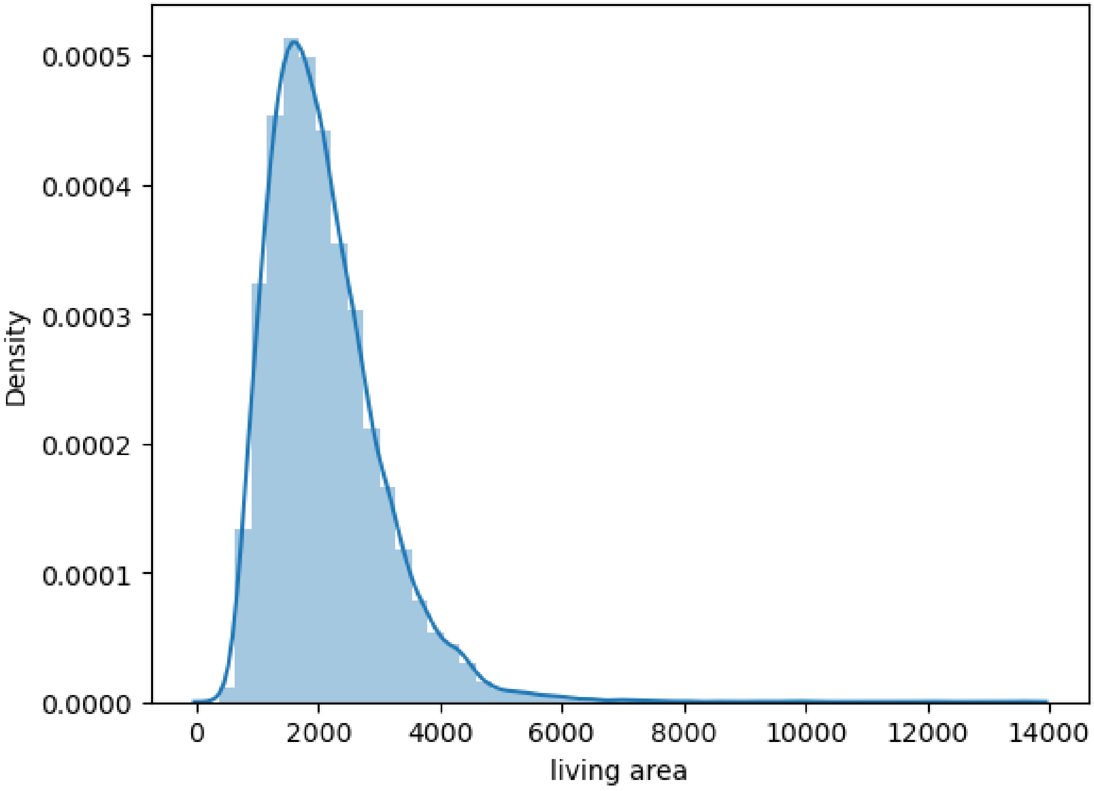
```
<AxesSubplot:xlabel='number_of_bedrooms', ylabel='Density'>
```



```
[11]: #displot on living area

      sns.distplot(df['living area'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for
histograms).
  warnings.warn(msg, FutureWarning)
```

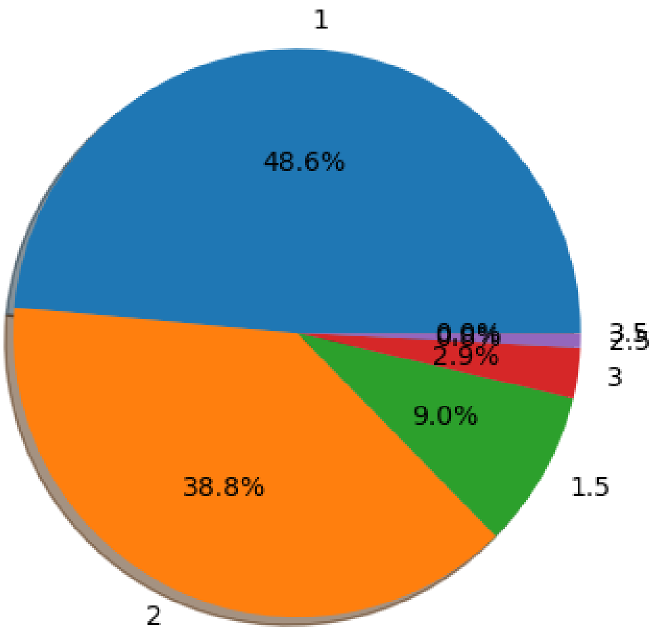
```
[11]: <AxesSubplot:xlabel='living area', ylabel='Density'>
```



[13]:
```
#2) Pie chart (on number of floors)

#checking values count

floor_count = df['number of floors'].value_counts()
floor_count
```

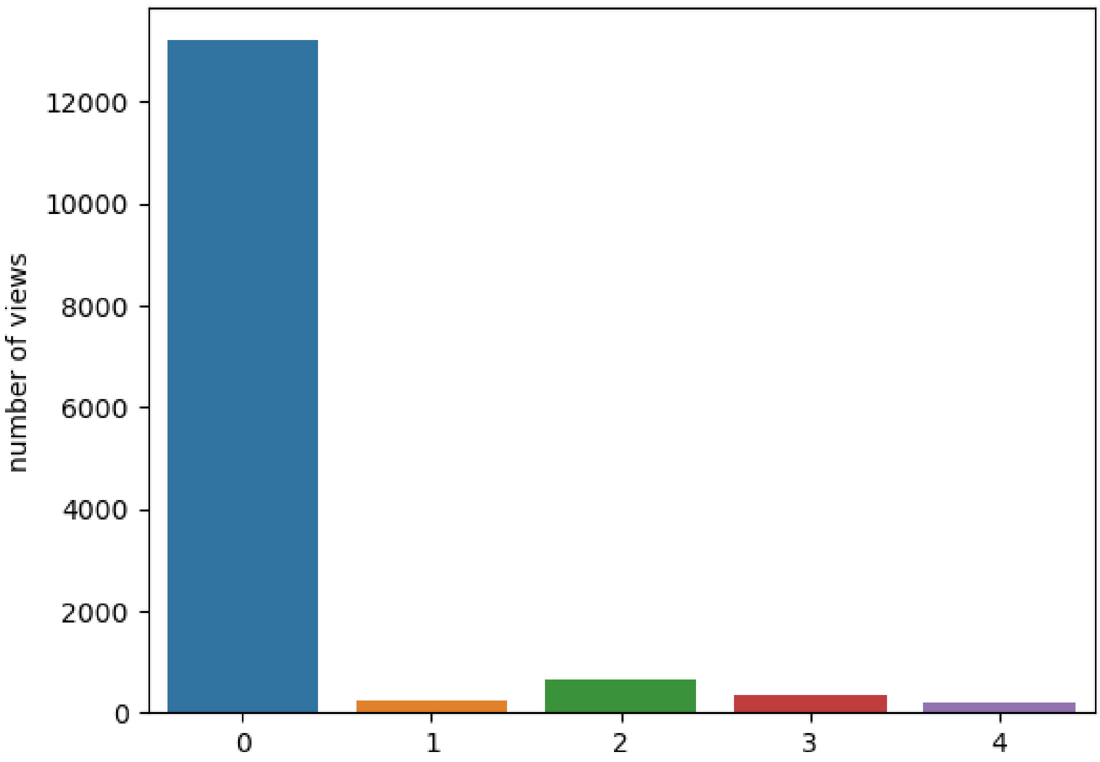
[13]:
```
1.0    7103
2.0    5666
1.5    1311
3.0     418
2.5     118
3.5       4
Name: number of floors, dtype: int64
```

[14]:
```
#since 6 values are present, hence all factors are according to 6 values

plt.pie(df['number of floors'].value_counts(), labels=[1,2,1.5,3,2.5,3.5],␣
 ↪autopct= '%1.1f%%', shadow=True)
plt.show()
```



```
[16]: #3) Bar plot on number of views

      sns.barplot(x=df['number of views'].value_counts().index, y=df['number of␣
       ↪views'].value_counts())
```

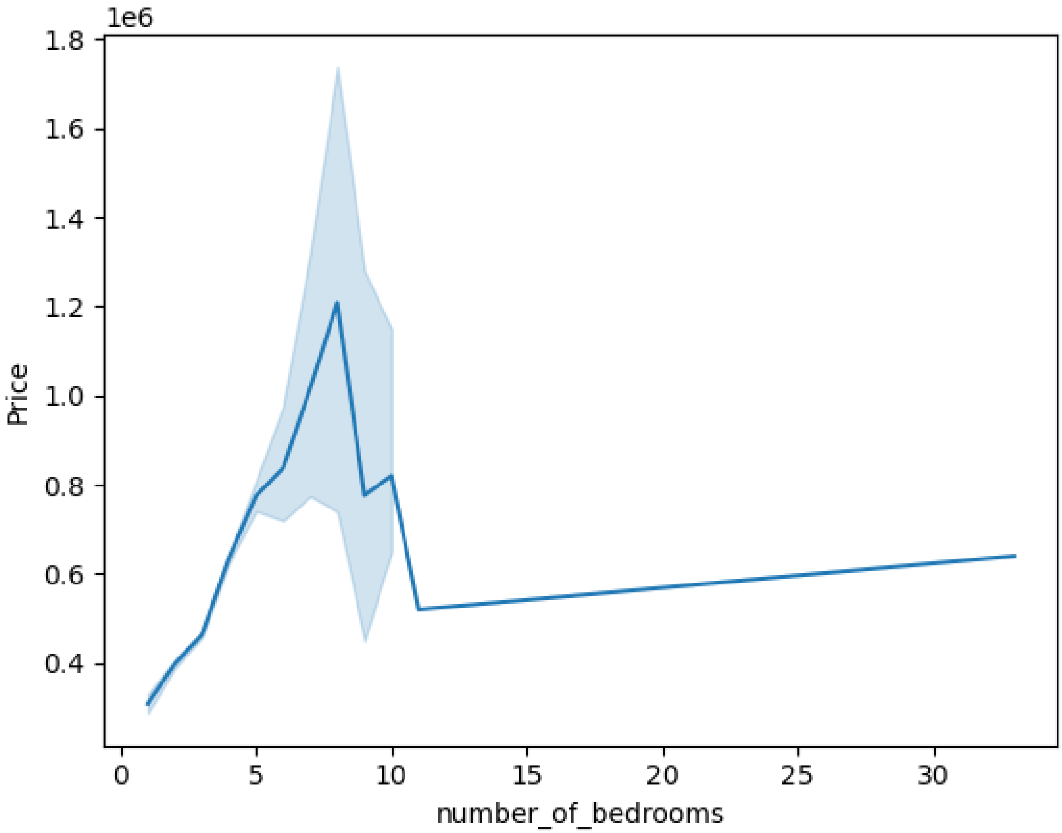
```
[16]: <AxesSubplot:ylabel='number of views'>
```



**2. Bivariate Graphs

[20]:
```
#Line Plot (between price and number of bedrooms)

sns.lineplot(x=df['number_of_bedrooms'] , y=df['Price'])
```

[20]:
```
<AxesSubplot:xlabel='number_of_bedrooms', ylabel='Price'>
```

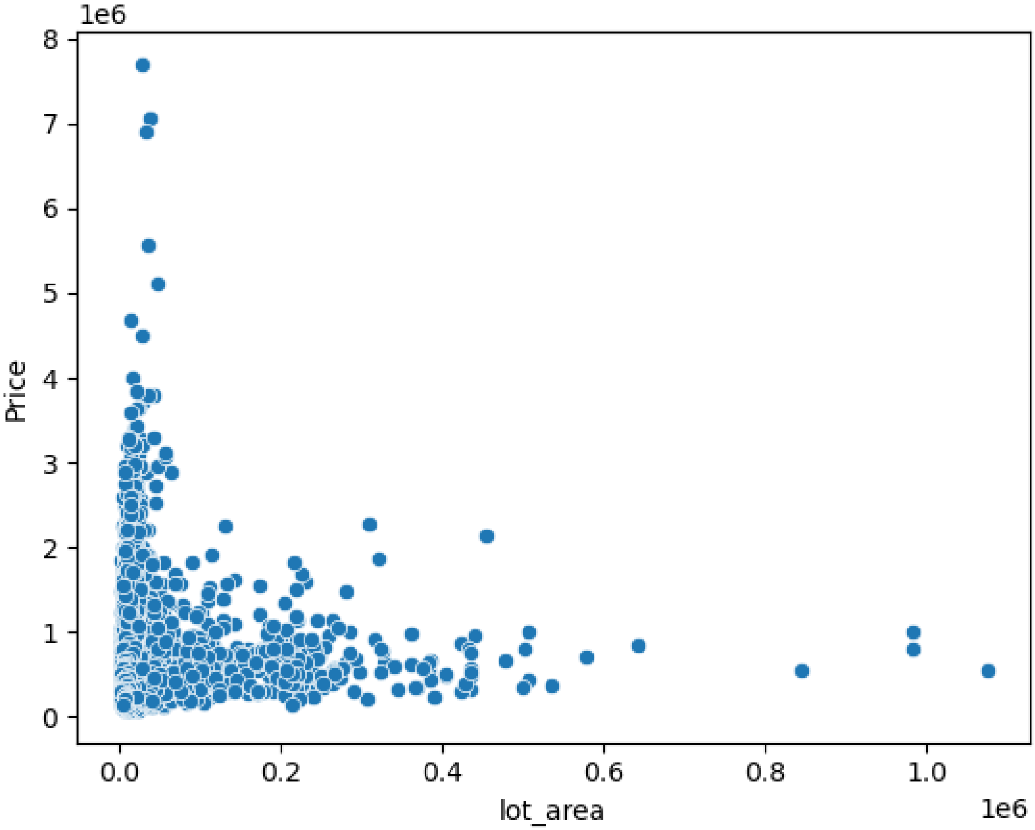


[21]:

```
#Scatter Plot (Between lot_area and Price)

sns.scatterplot(x= df.lot_area, y=df.Price)
```

[21]:

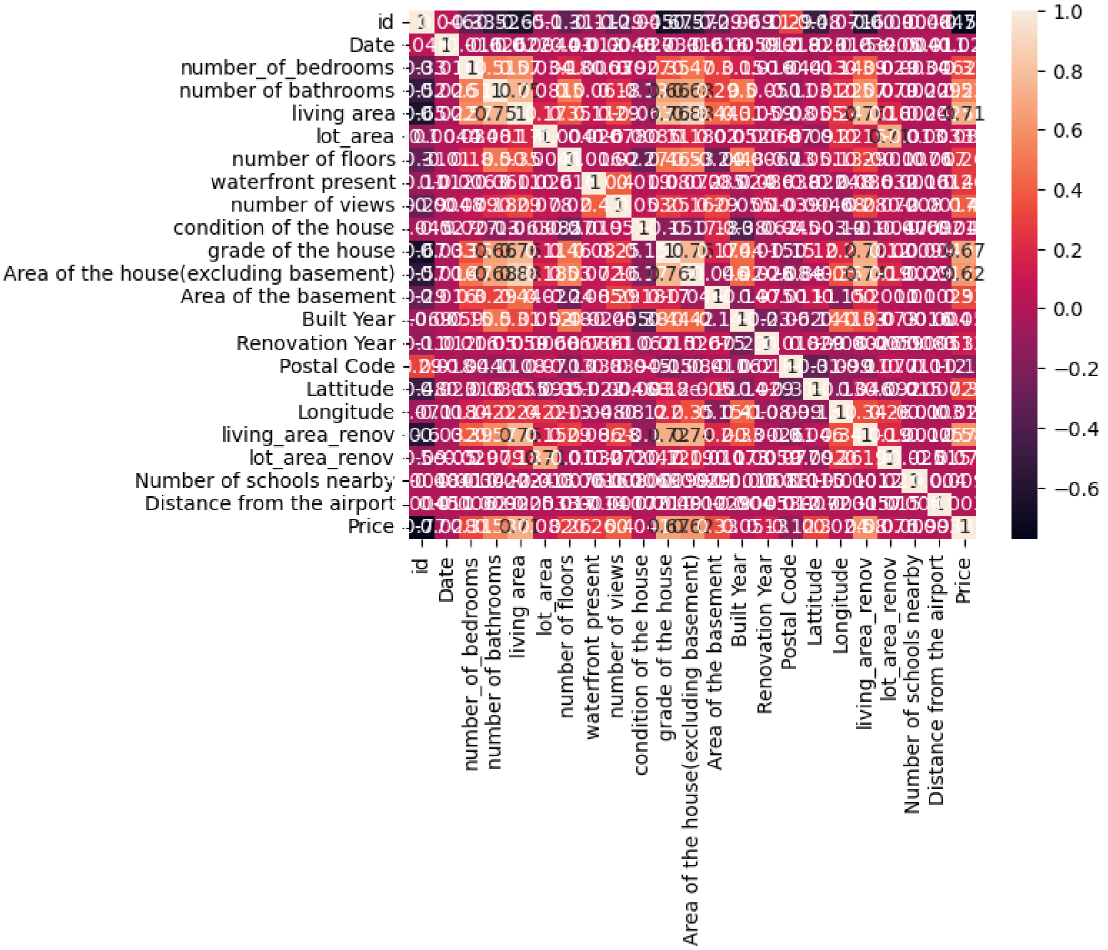
```
<AxesSubplot:xlabel='lot_area', ylabel='Price'>
```



**3 Multivariate Analysis

[26]:
```
#HeatMap

sns.heatmap(df.corr(),annot=True)
```

[26]:
```
<AxesSubplot:>
```



**Descriptive statistics

[28]:
```
df.describe()
```

[28]:
```
                 id          Date  number_of_bedrooms  number of bathrooms  \
count  1.462000e+04  14620.000000        14620.000000         14620.000000
mean   6.762821e+09  42604.538646            3.379343             2.129583
std    6.237575e+03     67.347991            0.938719             0.769934
min    6.762810e+09  42491.000000            1.000000             0.500000
25%    6.762815e+09  42546.000000            3.000000             1.750000
50%    6.762821e+09  42600.000000            3.000000             2.250000
75%    6.762826e+09  42662.000000            4.000000             2.500000
max    6.762832e+09  42734.000000           33.000000             8.000000

        living area      lot_area  number of floors  waterfront present  \
count  14620.000000  1.462000e+04      14620.000000        14620.000000
mean    2098.262996  1.509328e+04          1.502360            0.007661
```

```
std      928.275721  3.791962e+04          0.540239            0.087193
min      370.000000  5.200000e+02          1.000000            0.000000
25%     1440.000000  5.010750e+03          1.000000            0.000000
50%     1930.000000  7.620000e+03          1.500000            0.000000
75%     2570.000000  1.080000e+04          2.000000            0.000000
max    13540.000000  1.074218e+06          3.500000            1.000000

       number of views  condition of the house  …    Built Year  \
count     14620.000000            14620.000000  …  14620.000000
mean          0.233105                3.430506  …   1970.926402
std           0.766259                0.664151  …     29.493625
min           0.000000                1.000000  …   1900.000000
25%           0.000000                3.000000  …   1951.000000
50%           0.000000                3.000000  …   1975.000000
75%           0.000000                4.000000  …   1997.000000
max           4.000000                5.000000  …   2015.000000

       Renovation Year    Postal Code     Lattitude     Longitude  \
count     14620.000000   14620.000000  14620.000000  14620.000000
mean         90.924008  122033.062244     52.792848   -114.404007
std         416.216661      19.082418      0.137522      0.141326
min           0.000000  122003.000000     52.385900   -114.709000
25%           0.000000  122017.000000     52.707600   -114.519000
50%           0.000000  122032.000000     52.806400   -114.421000
75%           0.000000  122048.000000     52.908900   -114.315000
max        2015.000000  122072.000000     53.007600   -113.505000

       living_area_renov  lot_area_renov  Number of schools nearby  \
count       14620.000000    14620.000000              14620.000000
mean         1996.702257    12753.500068                  2.012244
std           691.093366    26058.414467                  0.817284
min           460.000000      651.000000                  1.000000
25%          1490.000000     5097.750000                  1.000000
50%          1850.000000     7620.000000                  2.000000
75%          2380.000000    10125.000000                  3.000000
max          6110.000000   560617.000000                  3.000000

       Distance from the airport         Price
count               14620.000000  1.462000e+04
mean                   64.950958  5.389322e+05
std                     8.936008  3.675324e+05
min                    50.000000  7.800000e+04
25%                    57.000000  3.200000e+05
50%                    65.000000  4.500000e+05
75%                    73.000000  6.450000e+05
max                    80.000000  7.700000e+06
```

```
[8 rows x 23 columns]
```

**Finding Null Values

[29]:

```
#From the above describe we can see that count for all columns is same, hence␣
 ↪we can say that no null values is present. But, we will still check with␣
 ↪other method
```

[30]:

```
#Checking null with isnull().any()

df.isnull().any()
```

[30]:

```
id                                       False
Date                                     False
number_of_bedrooms                       False
number of bathrooms                      False
living area                              False
lot_area                                 False
number of floors                         False
waterfront present                       False
number of views                          False
condition of the house                   False
grade of the house                       False
Area of the house(excluding basement)    False
Area of the basement                     False
Built Year                               False
Renovation Year                          False
Postal Code                              False
Lattitude                                False
Longitude                                False
living_area_renov                        False
lot_area_renov                           False
Number of schools nearby                 False
Distance from the airport                False
Price                                    False
dtype: bool
```

[31]:

```
#We observe that no null values are present in the data
```