In [1]:

```
#importing libraries
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from scipy import linalg
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
#reading csv
df = pd.read_csv("Mumbai1.csv")
```

In [3]:

```
df
```

Out[3]:

	0	Price	Area	Location	No. of Bedrooms	New/Resale	Gymnasium	Lift Available	C Parkir
0	0	4850000	720	Kharghar	1	0	0	1	
1	1	4500000	600	Kharghar	1	0	1	1	
2	2	6700000	650	Kharghar	1	0	1	1	
3	3	4500000	650	Kharghar	1	0	0	1	
4	4	5000000	665	Kharghar	1	0	0	1	
...	...	...	...	...	...	...	...	...	
6342	6342	2485000	700	Shirgaon	1	1	0	0	
6343	6343	14500000	900	Thane West	2	0	0	0	
6344	6344	14500000	900	Thane West	2	0	0	1	
6345	6345	4100000	1380	Boisar	3	0	0	0	
6346	6346	2750000	700	Badlapur East	1	1	1	1	

6347 rows × 19 columns

In [4]:

```
#dropping unnecessary columns
df2 = df.drop(["Children's Play Area",'Intercom','Landscaped Gardens','Indoor Games','M
aintenance Staff','24x7 Security'],axis = 'columns')
df2
```

Out[4]:

	0	Price	Area	Location	No. of Bedrooms	New/Resale	Gymnasium	Lift Available	C Parkir
0	0	4850000	720	Kharghar	1	0	0	1	
1	1	4500000	600	Kharghar	1	0	1	1	
2	2	6700000	650	Kharghar	1	0	1	1	
3	3	4500000	650	Kharghar	1	0	0	1	
4	4	5000000	665	Kharghar	1	0	0	1	
...	...	...	...	...	...	...	...	...	
6342	6342	2485000	700	Shirgaon	1	1	0	0	
6343	6343	14500000	900	Thane West	2	0	0	0	
6344	6344	14500000	900	Thane West	2	0	0	1	
6345	6345	4100000	1380	Boisar	3	0	0	0	
6346	6346	2750000	700	Badlapur East	1	1	1	1	

6347 rows × 13 columns

In [5]:

```
#cleaning
df2.isnull().sum()
```

Out[5]:

```
0                  0
Price              0
Area               0
Location           0
No. of Bedrooms    0
New/Resale         0
Gymnasium          0
Lift Available     0
Car Parking        0
Clubhouse          0
Gas Connection     0
Jogging Track      0
Swimming Pool      0
dtype: int64
```

In [6]:

```
#checking "Area" column
df2['Area'].unique()
```

Out[6]:

```
array([ 720,  600,  650, ...,  250, 1162,  435], dtype=int64)
```

In [7]:

```
df2['Price'].unique()
```

Out[7]:

```
array([  4850000,   4500000,   6700000,   5000000,  17000000,  12500000,
        10500000,  15000000,   8700000,   9000000,  11000000,   9700000,
         8000000,   8500000,   9300000,   9900000,   4200000,  28000000,
        40000000,  16000000,   9500000,   7500000,  13000000,  13700000,
         7700000,   6000000,   6200000,   6500000,   5200000,   5500000,
        16500000,  11399999,  13500000,  22500000,   5700000,   4800000,
         4300000,   5300000,   4803000,  15500000,   9400000,   5600000,
        17500000,  12000000,  16600000,   4900000,  22000000,  17800000,
        14000000,   7200000,   7000000,  10000000,   6900000,   8300000,
         5100000,   5400000,  17100000,  11800000,   4100000,  19000000,
        13600000,   8600000,   4700000,  16200000,  13200000,  11600000,
         2018000,  65000000,  45000000,   2375000,  19500000,   2000000,
         2500000,   2250000,   2700000,  43500000,   4760000,  18100000,
        23800000,  21200000,  36500000,  23700000,  21400000,  18000000,
        35300000,  15300000,  25000000,  13899999,  33500000,  14500000,
        21600000,  27799999,  27900000,  81600000,  38600000,  60100000,
        31500000,  37500000,  31000000,  21100000,  26000000,  15200000,
         2300000,  33000000,  29000000,  11200000,  11500000,  12100000,
        27000000,  30200000,  41200000,  30500000,  36700000,  20500000,
        32500000,  27500000,  35900000,  42000000,  33100000, 106000000,
       101400000,  18500000,   9200000,  50100000,  34100000,  21800000,
         3000000,  17900000,   6400000,  90000000,   2100000,   3600000,
        10100000,  10800000,  12700000,   8800000,  16800000,  16299999,
         3400000,  33700000,  10700000,  17600000,   4000000,   6250000,
         3800000,   4400000,   2600000,  21000000,  19600000,  18600000,
        21500000,  13799999,   9600000,   2010000,   2005000,   2548000,
         2492000,   2418000,   2400000,  21700000,  16700000,   2900000,
         5425000,   7100000,   7800000,   5900000,   4050000,   5050000,
         7400000,   6800000,   7950000,   5460000,   4950000,   8575000,
         7900000,  36000000,  35000000,  30000000,  14900000,   3700000,
         3850000,   2850000,   6850000,  10200000,   3100000,  32799999,
        36100000,   3682000,   2865000,   2374000,  62500000,  26500000,
        10600000,   3985000,   2200000,   2650000,   4600000,  72000000,
         6100000,   3950000,  11100000,   3500000,   5800000,   8200000,
        29500000,   3588000,   3650000,   3001000,   3300000,   6300000,
         2580000,   3594000,   5760000,   6952000,   2451000,   6137000,
         8100000,   3350000,  85000000, 125000000,  80000000,  28800000,
        45100000,  39200000,  25400000,  20000000,  46500000,  33400000,
         2035000,   6600000,  11700000,  23000000,  20700000,   3874000,
        22100000,  28500000,  18800000,  24000000,   9800000,  28200000,
        26400000,  10900000,   7300000,  19100000,   2170000,  34800000,
        11299999,   8725000,  58500000,  40199999,   2350000,   2800000,
        31100000,  30400000,  12300000,  13300000, 180000000,  29400000,
        24500000,  47000000, 100000000, 360000000,   7600000,   3200000,
         9350000,  48000000,  72500000,  70000000, 175000000,  86500000,
       120000000,  75000000,  32000000, 150000000,  55000000, 400000000,
        25500000, 112500000,   6990000,  14200000,  23300000, 110000000,
        56000000,  52500000,  12600000,  38000000,  92500000,  12200000,
        12400000,  28400000,   6656000,   9750000,  50000000,   9250000,
        20099999,  41500000,  24800000,   5545000,   5556000,   7311000,
         2399000,  60000000,   3467000,   6370000,   3310000,   2770000,
         2670000,   2110000, 200000000,  65300000,  38700000,  51500000,
        14800000,   2050000,   8009999,   8400000,   7430000,  25700000,
        62000000,   2011000,   2014999,   2021000,   2116000,  37000000,
        19900000,  32100000,  18200000,  39600000,  32599999,  18900000,
        21900000,  23500000, 220000000,  42500000,   3720000,  14700000,
        43700000,  63000000,  12900000,  14100000,  31200000,  39900000,
        22200000,  20200000,   2625000, 145000000,   6350000,   3250000,
         2450000,   2750000,   2077000,  12800000,   8900000,   8750000,
```

```
  2476000,  34000000,  19400000,  24100000,   6193000,  10300000,
  8655000,   3900000,   3624000,   7031000,  39000000,   2357000,
 15400000,  17200000,   6050000,  15800000,   7050000,  53000000,
  4665000,  28600000,   2164000,   5994000,   2534000,  19700000,
  2105000, 160000000, 250000000,   4350000, 153600000,   3750000,
260000000,  32300000,  35500000,  11900000,   9100000,  46000000,
140000000, 170000000,  57000000, 420000000,   8250000,   7053000,
  4713000,  95100000, 119900000,   4199000,   7575000,  39400000,
 66600000,  59500000,  43000000,  44000000,   9485000,   8184999,
  9377000,   9958000,   9997000,  10400000,  92400000, 113400000,
139500000,   4925000,   5898000,   6498999,   9497000,   6497000,
  7998000,   9496000,   5894000,   7494000,   6498000,   7495999,
  9499000,   4250000,  15900000,   5550000,  29200000,  59400000,
  7959999,   3979999,   4380000,   5060000,   4580000,   4450000,
  2840000,   2560000,   4650000, 105500000,  88000000, 122500000,
 71000000,   2816000,   2596000,   6559999,   7240000,   7940000,
 25800000,   2394000,   2436000,   2225000,   2565000,  25600000,
 77500000,  82500000,   3853000,   4907000,   8323999,  33800000,
 23900000,   3797999,   3890000,   2425000,   2815000,   3615000,
  3999000,   3954999,   3415000,   3410000,  15100000,   5399000,
 29900000,   8338000,   6426000,   8480000,   6048000,   8330000,
  6120000,  57500000,  41900000,   7837000,   5250000, 230000000,
  7150000,   8320000,   5160000,   5690000,   8413000,   5330000,
 27300000,  67500000,  14600000,   9150000,  15700000,  19300000,
 16399999,  23600000,  16900000,   7450000,  19800000,  38100000,
  9709000,  36400000,  27599999,  78700000,   6217000,   8020000,
 28700000,  51000000,   6419000,  41000000,  58000000,  38500000,
 74500000,  71500000,   3951000,   4151000,   2899000,   5999000,
  5699000,   4099000,   5499000,   5299000,   4399000,   5599000,
  5990000,   4550000,   5325000,   3049000,   2099000,   4499000,
  3372000,   3446000,   4656000,   4621000,   8298999,   7598999,
  7998999,   7798999,   7898999,   6351000,  44500000,  27400000,
  3990000,   4225000,   4140999,  16100000,   7209999,   5637000,
  9450000,   5150000,  31800000,  46200000,  31300000,   8920000,
  7320000,  13100000,  52100000,  14300000,  24600000,   2150000,
  2190000,   5518000,   5650000,   7650000,  64300000,  39100000,
 60500000,  34700000,  25200000, 105000000,   4150000,  82000000,
 54200000,  55400000,  55700000,  60700000,  79200000,  62100000,
 85600000,  54500000,  65500000,  68100000,   2999000,  67000000,
  8016000,   4043000,   4498000,   8294000,   6242000,  17700000,
 99000000,  17400000,  69900000,  57200000,  45199999,  37400000,
 56500000,   9041000,   9167000,   7881000,   5686000,  52600000,
 84500000,  50800000, 101800000,  47800000,  51300000,  20900000,
 17300000,  32700000,  59000000,   8628000,   9745000,   9013000,
  4750000,  47500000,  27100000,  35400000,  39800000,  20299999,
  8844000,   7290000,   6787000,   7198999,   7884000,  35700000,
 31700000,  22300000,  20800000,  19200000,  45500000,  90399999,
 95000000,   2064000,   2888000,   2898000,   2886000,   6998999,
 34500000,   8499000,  36600000,   3747999,   4351000,   4140000,
 26900000,   3425000,   9620000,   8420000,  88699999,   4517000,
  2210000,   2684000,   9520000,   2669000,   2779000,   6299000,
 75100000,  26800000,   5850000,   2378000,   2283000,   2435000,
  5577000,   3329999,  49500000,   2585000,   2574000,   2551000,
  4095000,   2599000,   2582000,   6320000,   3120000,   8359999,
  9230000,   6450000,   2174000,   2156000,   2460000,   7548000,
  3599000,  22400000,  30900000,   6022000,   9570000,  14400000,
  9199000,   6448999,  55599999,   2249000,  94900000,  52400000,
  3424000,   2360000,   2320000,   2640000,  63900000,  51900000,
 47600000,   5099000,   7298999,   8999000,  23200000,   3299000,
  3799000,   4602000,   5675000,   6650000,   7250000,  28100000,
 23400000,  59600000,  45700000,  26200000,   2075000,   8495000,
```

```
 4823000,   7618000,   6454000,   8287000,   7930000,   2990000,
 3185000,   3445000,   3565999,   4267000,   4427000,   3872000,
 3125000,   2725000,   3811999,   4052000,   2950000,   2575000,
 6694000,   6663000,  13400000,   9774000,   9226000,  54000000,
46300000,   8990000,   7731000,   8198999,   9599000,   7498999,
 6575000,   7815000,   8053000,   7652000,   8030000,   7745999,
 6738000,   5229000,   2515000,   6666000,   3977000,   2795000,
 7350000,   4390000,   2882000,   2485000,   3150000,   8381999,
 6186000,   6164000,   8527000,   2331000,   3317000,   8588000,
 2178000,   3877000,   3499000,   3536000,   4361000,   4459000,
 3258000,   4067000,   3087000,   8399000,   8975000,   8599000,
 9618000,   3920000,  27200000,   2139000,   4280000,   2780000,
 2875000,   6220000,   7020000,   9370000,   9550000,   9990000,
 9650000,   8994000,   8822000,   2940000,   4785000,   3663000,
 5555000,   2799000,   5199000,   2550000,   2781000,   5097000,
26700000,   7890000,   7909999,  49000000,  47900000, 107600000,
36800000,  79900000,   3550000,   3872999,   4794000,   4554000,
72300000,   2130000,  38800000,  35200000,  28300000,  29800000,
73700000,  42600000,  44300000,  59700000,  62800000,  78800000,
 3711999,   3728000,  68000000,   8712000,   8668000,   8580000,
 3450000,   4875000,   5350000,  39500000, 115000000,   4425000,
 4489000,   3151000,   8733000,   6898999,   2910000,   3925000,
 4207000,   3622000,   3779999,   3735000,   4072000,   3757000,
 3892000,   3420000,   2677000,   2835000,   3690000,   2970000,
 4275000,   2520000,   4004999,   3397000,   2767000,   4045000,
 3645000,   4117000,   4410000,   3899000,   3960000,   3829999,
 2288000,   3110000,   3243000,  54100000,   5040000,   2390000,
 5583000,  37600000,  41100000,  22799999,  32900000,  37800000,
35100000,  34600000,  36900000,   8950000,   4799000,  31900000,
15600000,  18400000,  26100000,   9601000,   7929000,   4837000,
 6187000,   8212000,   6637000,   7481000,   6862000,   4556000,
18700000,   3325000,   2836000,  21300000,  23100000,   3050000,
 3675000,   6542000,   4475000,   4396000,   4278000,   2948000,
 3847000,   4163000,   2804000,   3133000,   3365999,   5750000,
 5742000,   8355000,   8064000, 138000000,  40400000,   2330000,
 2265000,   2925000,   2445000,   2345000,   2255000,   2310000,
 2440000,   2305000,   4179999,   4170000,   4299000,   4349000,
 6190000,  40300000,   3188000,  91800000,   3061000,   2063000,
 2929000,   4699000,   4999000,   3699000,   3488000,   8192000,
 5401000,   6798999,   4801000,   8995000,   8978000,   5950000,
 2057000,   3722999,   3988000,   3308000,   3661000,   6152000,
 3459000,   4371000,   3574000,   4584000,   3326000,  29700000,
 4260000,   2576000, 240000000,   3064000,  37700000,  79000000,
 2465000], dtype=int64)
```

In [8]:

```
#checking "New/Resale" column
df2['New/Resale'].unique()
```

Out[8]:

```
array([0, 1], dtype=int64)
```

In [9]:

```
#checking "No. of Bedrooms" column
df2['No. of Bedrooms'].unique()
```

Out[9]:

```
array([1, 4, 3, 2, 5, 6, 7], dtype=int64)
```

In [10]:

```
#replacing 'no. of bedrooms' by 'bhk'
df2['bhk'] = df2['No. of Bedrooms']
```

In [11]:

```
df3 = df2.drop(['No. of Bedrooms'],axis = 'columns')
```

In [12]:

```
df3
```

Out[12]:

	0	Price	Area	Location	New/Resale	Gymnasium	Lift Available	Car Parking	Clubhou
0	0	4850000	720	Kharghar	0	0	1	1	
1	1	4500000	600	Kharghar	0	1	1	1	
2	2	6700000	650	Kharghar	0	1	1	1	
3	3	4500000	650	Kharghar	0	0	1	1	
4	4	5000000	665	Kharghar	0	0	1	1	
...	...	...	...	...	...	...	...	...	
6342	6342	2485000	700	Shirgaon	1	0	0	0	
6343	6343	14500000	900	Thane West	0	0	0	0	
6344	6344	14500000	900	Thane West	0	0	1	0	
6345	6345	4100000	1380	Boisar	0	0	0	0	
6346	6346	2750000	700	Badlapur East	1	1	1	1	

6347 rows × 13 columns

In [13]:

```
#defining a function that will help in finding float values in a column
def find_float(x):
    try:
        float(x)
    except:
        return False
    return True
```

In [14]:

```
df3[~df3['Area'].apply(find_float)]
```

Out[14]:

0	Price	Area	Location	New/Resale	Gymnasium	Lift Available	Car Parking	Clubhouse	Connect

In [15]:

```
df3[~df3['Price'].apply(find_float)]
```

Out[15]:

0	Price	Area	Location	New/Resale	Gymnasium	Lift Available	Car Parking	Clubhouse	Connect

In [16]:

```
#feature eng
#calculating price per area
df4 = df3.copy()
df4['price_per_sqft'] = df4['Price']/df4['Area']
df4
```

Out[16]:

	0	Price	Area	Location	New/Resale	Gymnasium	Lift Available	Car Parking	Clubhou
0	0	4850000	720	Kharghar	0	0	1	1	
1	1	4500000	600	Kharghar	0	1	1	1	
2	2	6700000	650	Kharghar	0	1	1	1	
3	3	4500000	650	Kharghar	0	0	1	1	
4	4	5000000	665	Kharghar	0	0	1	1	
...	...	...	...	...	...	...	...	...	
6342	6342	2485000	700	Shirgaon	1	0	0	0	
6343	6343	14500000	900	Thane West	0	0	0	0	
6344	6344	14500000	900	Thane West	0	0	1	0	
6345	6345	4100000	1380	Boisar	0	0	0	0	
6346	6346	2750000	700	Badlapur East	1	1	1	1	

6347 rows × 14 columns

In [17]:

```
#checking uniques in 'location'
df4.Location.unique()
```

Out[17]:

```
array(['Kharghar', 'Sector-13 Kharghar', 'Sector 18 Kharghar',
       'Sector 20 Kharghar', 'Sector 15 Kharghar', 'Dombivali',
       'Churchgate', 'Prabhadevi', 'Jogeshwari West', 'Kalyan East',
       'Malad East', 'Virar East', 'Virar', 'Malad West', 'Borivali East',
       'Mira Road East', 'Goregaon West', 'Kandivali West',
       'Borivali West', 'Kandivali East', 'Andheri East', 'Goregaon East',
       'Wadala', 'Ulwe', 'Dahisar', 'kandivali', 'Goregaon',
       'Bhandup West', 'thakur village kandivali east', 'Santacruz West',
       'Kanjurmarg', 'I C Colony', 'Dahisar W', 'Marol', 'Parel',
       'Lower Parel', 'Worli', 'Jogeshwari East', 'Chembur Shell Colony',
       'Central Avenue', 'Chembur East', 'Diamond Market Road', 'Mulund',
       'Nalasopara West', 'raheja vihar', 'Powai Lake', 'MHADA Colony 20',
       'Tolaram Colony', 'Taloja', 'Thane West', 'Vangani',
       'Sector 5 Ulwe', 'Sector12 New Panvel', 'Sector 17 Ulwe',
       'Sector9 Kamothe', 'Sector 19 Kharghar', 'Navi Basti',
       'Sector12 Kamothe', 'Sector 21 Kamothe', 'Rutu Enclave',
       'taloja panchanand', 'Virar West', 'Chembur', 'Sector 20 Kamothe',
       'Sector 22 Kamothe', 'Sector 18 Kamothe', 'Sector-5 Kamothe',
       'Sector-6A Kamothe', 'Sector 11 Kamothe', 'Sector-18 Ulwe',
       'Sector-12 Kamothe', 'azad nagar', 'Sindhi Society Chembur',
       'Kurla', 'Sahkar Nagar', 'Deonar', 'Thane', 'Jankalyan Nagar',
       'Badlapur', 'Ambarnath', 'Ambernath West', 'Vakola', 'Kamothe',
       'Kamothe Sector 16', 'Almeida Park', 'Khar', 'Bandra West',
       'Pali Hill', '15th Road', 'Palghar', 'Sector13 Kharghar',
       'Sector 21 Kharghar', 'Sector 12 Kharghar', 'Vivek Vidyalaya Marg',
       'Vasai east', 'Nahur', 'Badlapur West', 'Panvel', 'Kalyan',
       'Badlapur East', 'Mira Bhayandar', 'Juhu', 'Naigaon East',
       'Sector 21 Ulwe', 'Bandra East', 'Dronagiri', 'Nerul', 'Karanjade',
       'Sanpada', 'Sector-8 Ulwe', 'Sector-3 Ulwe', 'Sector 23 Ulwe',
       'ULWE SECTOR 19', 'Ghodbunder Road', 'Bhiwandi', 'Vasai',
       'Nala Sopara', 'Dadar East', 'Ghatkopar', 'Breach Candy',
       'Worli South Mumbai', 'Asangaon', 'Koparkhairane Station Road',
       'Kopar Khairane Sector 19A', 'Koper Khairane',
       'Eastern Express Highway Vikhroli', 'Magathane', 'Rawal Pada',
       'Ambernath East', 'Dokali Pada', 'Dattapada', 'Rajendra Nagar',
       'Kulupwadi', 'Samata Nagar Thakur Village', 'Mira Road and Beyond',
       'West Amardeep Colony', 'Pant Nagar', 'mumbai', 'Four Bungalows',
       'no 9', 'kolshet', 'Hiranandani Meadows', 'Kalpataru', 'Petali',
       'Kharghar Sector 34C', 'Ghatkopar East',
       'Mumbai Agra National Highway', 'vasant vihar thane west',
       'Kalyan West', 'Shirgaon', 'Pokhran 2', 'juhu tara', 'Peddar Road',
       'Palm Beach', 'Sector 10', 'Sector 19 Kamothe', 'Tilak Nagar',
       'Ghatkopar West', 'Tardeo', 'Napeansea Road', 'Mahalaxmi',
       'Dahisar West', 'Mulund West', 'Natakwala Lane', 'Link Road',
       'Devidas Cross Lane', 'Soniwadi Road', 'Haridas Nagar', 'Shimpoli',
       'TPS Road', 'Off Shimpoli road', 'Rustomjee Global City',
       'Sunil Nagar', 'Sector 30 Kharghar', 'Sector 12 A', 'Sector 18',
       'Sector13 Khanda Colony', 'Sector16 Airoli', 'Ranjanpada',
       'Sector 15', 'Sector 35G', 'Sector 5', 'Sector 35I Kharghar',
       'Sector35D Kharghar', 'Sector34 A Kharghar', 'Sector 30',
       'Sector 36 Kharghar', 'Sector 11 Belapur', 'Sector-34B Kharghar',
       'Dombivali East', 'Roadpali', 'Sector-50 Seawoods',
       'Mumbai Highway', 'Sector 7 Kharghar', 'Lokhandwala Township',
       'Andheri', 'Andheri West', 'Shastri Nagar', 'Wadala East Wadala',
       'Kalwa', 'PARSIK NAGAR', 'Maharashtra Nagar', 'Patlipada',
       'Belapur', 'Seawoods', 'Majiwada', '4 Bunglows', 'Airoli',
       'Kolshet Road', 'Sector 10 Khanda Colony', 'Pokharan Road',
       'Kharegaon', 'Panch Pakhadi', 'Sector 36 Kamothe',
       'Dombivli (West)', 'DN Nagar Road', 'Godrej Hill', 'Ganesh Nagar',
```

```
'Haware City', 'Mahatma Gandhi Road', 'Akurli Nagar',
'Kasar vadavali', 'Vasai West', 'Mumbai Nashik Expressway',
'Katrap', 'Mira Road', 'Kasheli',
'Western Express Highway Kandivali East', 'Vasind', 'KASHELI',
'Thakurli', 'Shakti Nagar', 'Bhayandar East', 'Dahisar East',
'ulhasnagar 4', 'Sector-26 Taloja', 'Koproli', 'Mumbai Central',
'Greater Khanda', 'link road borivali west', 'Manpada',
'Sector 2 Ulwe', 'Govind nagar', 'Krishanlal Marwah Marg',
'Sector-9 Ulwe', 'Vikhroli', 'Kalamboli', 'Lokhandwala',
'Patel Nagar', 'Yari Road', 'Thakur complex', 'Khar West',
'Sector 11 Kharghar', 'Ghansoli', 'Sector8 Sanpada',
'Jeejamata Nagar', 'Pandurangwadi', 'Shreyas Colony',
'Kannamwar Nagar II', 'gokuldham', 'Bangur Nagar',
'Shivaji Colony', 'Jawahar Nagar', 'Vedant Complex', 'Titwala',
'Manvel pada Road', 'Govandi', 'Shilphata Road Thane',
'Vasant Vihar', 'Thakur Village', 'Samata nagar', 'Wadi Bandar',
'Kapurbawadi', 'Thane Belapur Road Kalwa', 'Suburbs Mumbai',
'Ramdev Park', 'Sector-35 Kamothe',
'Sector 58A Seawoods Navi Mumbai', 'Diva', 'Borivali',
'Gundavali Gaothan', 'Kondivita Road', 'Koldongri', 'Saki Naka',
'Bhayandar West', 'Versova', 'Hanuman Nagar', 'worli sea face ',
'Sea Face', 'Shiv Sagar Estate', 'Grant Road West', 'Sriprastha',
'Off Nepean Sea Road', 'Morya Nagar', 'Upper Worli',
'worli sea Fase', 'Powai', 'Agripada', 'Sion', 'Girgaon',
'IT Colony', 'Bhayandarpada', 'Sector-24 Kamothe', 'Phase 2',
'Sector 6', 'Malad', 'Haji Ali', 'Majiwada thane', 'Yogi Hills',
'Sector 19 Nerul', 'Owale', 'MG Road', 'roadpali navimumbai',
'Taloja Bypass Nitalas Link Road', 'Rasayani', 'Sector 10 Kamothe',
'kavesar', 'Kopara', 'royal palms goregaon east', 'Dadar West',
'vrindavan society', 'CBD Belapur East', 'Parel Village',
'Karave Nagar', 'Sector 19A Nerul', 'Antop Hill', 'Suman Nagar',
'Bhakti Park', 'Hanuman Chowk', 'Willingdon', 'VishnuNagar',
'Vichumbe', 'Vashi', 'Poonam Sagar Complex', 'Santacruz East',
'Mulund East', 'Sector-8 Sanpada', 'Sector-9 Ghansoli', 'Balkum',
'Hiranandani Estates', 'Vartak Nagar', 'Charkop', 'dhanukarwadi',
'Vazira', 'Four Bunglows', 'Palava',
'Oshiwara Police Station Road', 'Seven Bunglow', 'Poddar Road',
'Gulmohar Road', 'Vile Parle E', 'Pokhran Road No 2',
'Sainath Nagar', 'Kapur Bawdi', 'Manpada near Tiku ji ni wadi',
'Kolshet Industrial Area', 'Sector16 Ulwe',
'Anand Nagar Thane West', 'Charkop Sector 8', 'Vile Parle',
'Marine Lines', 'Bandra Kurla Complex', 'sec 50 new',
'Vakola Pipeline Road', 'Neral', 'Shil Phata', 'Ville Parle East',
'matunga east', 'Dharavi', 'vile parle west', 'Anjurdive', 'Uran',
'Boisar', 'Kanjurmarg East', 'Kurla West', 'Sewri', 'Matunga',
'Goregaon (East)', 'Malabar Hill', 'Ambivali', 'Nalasopara East',
'Vijay Nagar', 'Chedda Nagar', 'Kurla East', 'Kharodi',
'Ville Parle West', 'Vikhroli West', 'Wadala East', 'Palidevad',
'Vikroli East', 'Mahim', 'Khalapur', 'Karjat', 'Vitthalwadi',
'Bhoiwada Kalyan', 'Beturkar Pada', 'Ambivli', 'Nilje Gaon',
'Khopoli', 'Taloje', 'Gulal Wadi', 'Tilak Nagar Mumbai',
'Chandivali', 'Nere', 'Kewale', 'Sector 9 Airoli', 'Diva Gaon',
'Sen Nagar', 'Adaigaon', 'Syndicate', 'Gandhar Nagar', 'Byculla',
'Padle Gaon'], dtype=object)
```

In [18]:

```
len(df4.Location.unique())
```

Out[18]:

```
413
```

In [19]:

```
#checking entries per location
df4.Location = df4.Location.apply(lambda x: x.strip())
location_stats = df4.groupby('Location')['Location'].agg('count').sort_values(ascending
=False)
location_stats
```

Out[19]:

```
Location
Kharghar                533
Thane West              418
Mira Road East          390
Ulwe                    319
Kalyan West             176
                       ...
Poonam Sagar Complex      1
Pokhran Road No 2         1
Pokharan Road             1
Poddar Road               1
15th Road                 1
Name: Location, Length: 413, dtype: int64
```

In [20]:

```
location_stats_other = location_stats[location_stats<=10]
location_stats_other
```

Out[20]:

```
Location
Sion                    10
Majiwada                10
Sector 19 Kharghar      10
kandivali                9
Hiranandani Estates      9
                        ..
Poonam Sagar Complex     1
Pokhran Road No 2        1
Pokharan Road            1
Poddar Road              1
15th Road                1
Name: Location, Length: 324, dtype: int64
```

In [21]:

```
len(location_stats_other)
```

Out[21]:

```
324
```

In [22]:

```
df5 = df4.copy()
```

In [23]:

```
df5.Location = df5.Location.apply(lambda x: 'other' if x in location_stats_other else x
)
```

In [24]:

```
df5
```

Out[24]:

	0	Price	Area	Location	New/Resale	Gymnasium	Lift Available	Car Parking	Clubhou
0	0	4850000	720	Kharghar	0	0	1	1	
1	1	4500000	600	Kharghar	0	1	1	1	
2	2	6700000	650	Kharghar	0	1	1	1	
3	3	4500000	650	Kharghar	0	0	1	1	
4	4	5000000	665	Kharghar	0	0	1	1	
...	...	...	...	...	...	...	...	...	
6342	6342	2485000	700	other	1	0	0	0	
6343	6343	14500000	900	Thane West	0	0	0	0	
6344	6344	14500000	900	Thane West	0	0	1	0	
6345	6345	4100000	1380	Boisar	0	0	0	0	
6346	6346	2750000	700	Badlapur East	1	1	1	1	

6347 rows × 14 columns

In [25]:

```
len(df5.Location.unique())
```

Out[25]:

```
90
```

In [26]:

```
#remove outliers
#assume 1bhk to occupy atleast 350sqft
df5[df5['Area']/(df5['bhk'])<350]
```

Out[26]:

	0	Price	Area	Location	New/Resale	Gymnasium	Lift Available	Car Parking	Clubhou
236	236	2300000	310	Kharghar	0	0	0	0	
239	239	11200000	573	Ulwe	1	1	1	1	
259	259	2500000	300	Kandivali East	1	0	0	0	
269	269	32500000	997	other	0	0	1	1	
340	340	21000000	620	other	0	0	0	1	
...	...	...	...	...	...	...	...	...	
6319	6319	5500000	580	Kharghar	0	0	0	0	
6331	6331	39500000	940	other	0	0	1	0	
6332	6332	39000000	940	other	0	0	1	0	
6333	6333	38500000	985	other	0	0	1	0	
6334	6334	38500000	985	other	0	0	1	0	

451 rows × 14 columns

In [27]:

```
#removing rows with area/bhk less than 350
df6 = df5[~(df5['Area']/(df5['bhk'])<350)]
df6
```

Out[27]:

	0	Price	Area	Location	New/Resale	Gymnasium	Lift Available	Car Parking	Clubhou
0	0	4850000	720	Kharghar	0	0	1	1	
1	1	4500000	600	Kharghar	0	1	1	1	
2	2	6700000	650	Kharghar	0	1	1	1	
3	3	4500000	650	Kharghar	0	0	1	1	
4	4	5000000	665	Kharghar	0	0	1	1	
...	...	...	...	...	...	...	...	...	...
6342	6342	2485000	700	other	1	0	0	0	
6343	6343	14500000	900	Thane West	0	0	0	0	
6344	6344	14500000	900	Thane West	0	0	1	0	
6345	6345	4100000	1380	Boisar	0	0	0	0	
6346	6346	2750000	700	Badlapur East	1	1	1	1	

5896 rows × 14 columns

In [28]:

```
df6.price_per_sqft.describe()
```

Out[28]:

```
count      5896.000000
mean      13199.743834
std        9471.416643
min        1597.444089
25%        7052.631579
50%       10000.000000
75%       16752.136752
max      109950.522265
Name: price_per_sqft, dtype: float64
```

In [29]:

```
#checking price per sqft for a location
df5[(df5['Location']=='Kharghar')]
```

Out[29]:

	0	Price	Area	Location	New/Resale	Gymnasium	Lift Available	Car Parking	Clubhou
0	0	4850000	720	Kharghar	0	0	1	1	
1	1	4500000	600	Kharghar	0	1	1	1	
2	2	6700000	650	Kharghar	0	1	1	1	
3	3	4500000	650	Kharghar	0	0	1	1	
4	4	5000000	665	Kharghar	0	0	1	1	
...	...	...	...	...	...	...	...	...	
6190	6190	13000000	1050	Kharghar	1	1	1	1	
6270	6270	6500000	650	Kharghar	0	1	1	1	
6274	6274	7700000	1162	Kharghar	0	0	0	0	
6319	6319	5500000	580	Kharghar	0	0	0	0	
6329	6329	13000000	1220	Kharghar	0	1	1	1	

533 rows × 14 columns

In [30]:

```
#defining a function to remove outliers
def remove_outliers_pps(df):
    df_out = pd.DataFrame()
    for key, subdf in df.groupby('Location'):
        m = np.mean(subdf.price_per_sqft)
        st = np.std(subdf.price_per_sqft)
        reduced_df = subdf[(subdf.price_per_sqft>(m-st)) & (subdf.price_per_sqft<(m+st
))]
        df_out = pd.concat([df_out,reduced_df],ignore_index=True)
    return df_out
```

In [31]:

```
df7 = remove_outliers_pps(df6)
df7
```

Out[31]:

	0	Price	Area	Location	New/Resale	Gymnasium	Lift Available	Car Parking	Clubhou
0	1145	14900000	1245	Airoli	1	0	1	1	
1	1146	14000000	1183	Airoli	1	1	1	1	
2	1884	14800000	1245	Airoli	0	0	1	1	
3	4105	11100000	1050	Airoli	0	0	1	0	
4	4148	7500000	600	Airoli	0	0	1	0	
...	...	...	...	...	...	...	...	...	
4620	6012	13899999	1200	other	0	0	1	0	
4621	6123	4260000	695	other	1	0	1	0	
4622	6238	3500000	650	other	0	0	1	1	
4623	6339	2465000	700	other	1	0	0	0	
4624	6342	2485000	700	other	1	0	0	0	

4625 rows × 14 columns

In [32]:

```
df8 = df7.drop(['price_per_sqft'],axis='columns')
df8
```

Out[32]:

	0	Price	Area	Location	New/Resale	Gymnasium	Lift Available	Car Parking	Clubhou
0	1145	14900000	1245	Airoli	1	0	1	1	
1	1146	14000000	1183	Airoli	1	1	1	1	
2	1884	14800000	1245	Airoli	0	0	1	1	
3	4105	11100000	1050	Airoli	0	0	1	0	
4	4148	7500000	600	Airoli	0	0	1	0	
...	...	...	...	...	...	...	...	...	
4620	6012	13899999	1200	other	0	0	1	0	
4621	6123	4260000	695	other	1	0	1	0	
4622	6238	3500000	650	other	0	0	1	1	
4623	6339	2465000	700	other	1	0	0	0	
4624	6342	2485000	700	other	1	0	0	0	

4625 rows × 13 columns

In [33]:

```
#one hot encoding/dummy
#creating a dummy df with locations as columns
dummies = pd.get_dummies(df8.Location)
dummies
```

Out[33]:

	Airoli	Ambernath East	Ambernath West	Andheri	Andheri East	Andheri West	Badlapur East	Bandra East	Bandra West
0	1	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0
3	1	0	0	0	0	0	0	0	0
4	1	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...
4620	0	0	0	0	0	0	0	0	0
4621	0	0	0	0	0	0	0	0	0
4622	0	0	0	0	0	0	0	0	0
4623	0	0	0	0	0	0	0	0	0
4624	0	0	0	0	0	0	0	0	0

4625 rows × 90 columns

In [34]:

```
#concatinating dummies with our df
df9 = pd.concat([df8,dummies.drop('other',axis='columns')],axis='columns')
df9
```

Out[34]:

	0	Price	Area	Location	New/Resale	Gymnasium	Lift Available	Car Parking	Clubhou
0	1145	14900000	1245	Airoli	1	0	1	1	
1	1146	14000000	1183	Airoli	1	1	1	1	
2	1884	14800000	1245	Airoli	0	0	1	1	
3	4105	11100000	1050	Airoli	0	0	1	0	
4	4148	7500000	600	Airoli	0	0	1	0	
...	...	...	...	...	...	...	...	...	
4620	6012	13899999	1200	other	0	0	1	0	
4621	6123	4260000	695	other	1	0	1	0	
4622	6238	3500000	650	other	0	0	1	1	
4623	6339	2465000	700	other	1	0	0	0	
4624	6342	2485000	700	other	1	0	0	0	

4625 rows × 102 columns

In [35]:

```
df9 = pd.concat([df8,dummies.drop('other',axis='columns')],axis='columns')
df9
```

Out[35]:

	0	Price	Area	Location	New/Resale	Gymnasium	Lift Available	Car Parking	Clubhou
0	1145	14900000	1245	Airoli	1	0	1	1	
1	1146	14000000	1183	Airoli	1	1	1	1	
2	1884	14800000	1245	Airoli	0	0	1	1	
3	4105	11100000	1050	Airoli	0	0	1	0	
4	4148	7500000	600	Airoli	0	0	1	0	
...	...	...	...	...	...	...	...	...	
4620	6012	13899999	1200	other	0	0	1	0	
4621	6123	4260000	695	other	1	0	1	0	
4622	6238	3500000	650	other	0	0	1	1	
4623	6339	2465000	700	other	1	0	0	0	
4624	6342	2485000	700	other	1	0	0	0	

4625 rows × 102 columns

In [36]:

```
df10 = df9.drop('Location',axis='columns')
df10
```

Out[36]:

	0	Price	Area	New/Resale	Gymnasium	Lift Available	Car Parking	Clubhouse	Conne
0	1145	14900000	1245	1	0	1	1	0	
1	1146	14000000	1183	1	1	1	1	1	
2	1884	14800000	1245	0	0	1	1	0	
3	4105	11100000	1050	0	0	1	0	0	
4	4148	7500000	600	0	0	1	0	1	
...	...	...	...	...	...	...	...	...	
4620	6012	13899999	1200	0	0	1	0	1	
4621	6123	4260000	695	1	0	1	0	0	
4622	6238	3500000	650	0	0	1	1	0	
4623	6339	2465000	700	1	0	0	0	0	
4624	6342	2485000	700	1	0	0	0	0	

4625 rows × 101 columns

In [37]:

```
#renaming column names for convenience
df10 = df10.rename(columns={'New/Resale': 'New_or_Resale',
                            'Lift Available': 'Lift_Available',
                            'Car Parking': 'Car_Parking',
                            'Gas Connection': 'Gas_Connection',
                            'Jogging Track': 'Jogging_Track',
                            'Swimming Pool': 'Swimming_Pool'})
df10
```

Out[37]:

	0	Price	Area	New_or_Resale	Gymnasium	Lift_Available	Car_Parking	Clubhc
0	1145	14900000	1245	1	0	1	1	
1	1146	14000000	1183	1	1	1	1	
2	1884	14800000	1245	0	0	1	1	
3	4105	11100000	1050	0	0	1	0	
4	4148	7500000	600	0	0	1	0	
...	...	...	...	...	...	...	...	
4620	6012	13899999	1200	0	0	1	0	
4621	6123	4260000	695	1	0	1	0	
4622	6238	3500000	650	0	0	1	1	
4623	6339	2465000	700	1	0	0	0	
4624	6342	2485000	700	1	0	0	0	

4625 rows × 101 columns

In [38]:

```
df10 = df10.drop('0',axis='columns')
```

In [39]:

```
#final dataset
df10
```

Out[39]:

	Price	Area	New_or_Resale	Gymnasium	Lift_Available	Car_Parking	Clubhouse	
0	14900000	1245	1	0	1	1	0	
1	14000000	1183	1	1	1	1	1	
2	14800000	1245	0	0	1	1	0	
3	11100000	1050	0	0	1	0	0	
4	7500000	600	0	0	1	0	1	
...	...	...	...	...	...	...	...	
4620	13899999	1200	0	0	1	0	1	
4621	4260000	695	1	0	1	0	0	
4622	3500000	650	0	0	1	1	0	
4623	2465000	700	1	0	0	0	0	
4624	2485000	700	1	0	0	0	0	

4625 rows × 100 columns

In [40]:

```
#training model
X = df10.drop('Price',axis='columns')
X
```

Out[40]:

	Area	New_or_Resale	Gymnasium	Lift_Available	Car_Parking	Clubhouse	Gas_Conne
0	1245	1	0	1	1	0	
1	1183	1	1	1	1	1	
2	1245	0	0	1	1	0	
3	1050	0	0	1	0	0	
4	600	0	0	1	0	1	
...	...	...	...	...	...	...	
4620	1200	0	0	1	0	1	
4621	695	1	0	1	0	0	
4622	650	0	0	1	1	0	
4623	700	1	0	0	0	0	
4624	700	1	0	0	0	0	

4625 rows × 99 columns

In [41]:

```
y = df10.Price
y
```

Out[41]:

```
0       14900000
1       14000000
2       14800000
3       11100000
4        7500000
          ...
4620    13899999
4621     4260000
4622     3500000
4623     2465000
4624     2485000
Name: Price, Length: 4625, dtype: int64
```

In [42]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2,random_state=2)
```

In [43]:

```
#checking score
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train,y_train)
regressor.score(X_test,y_test)
```

Out[43]:

```
0.8730166240133534
```

In [44]:

```
X_test[0:1]
```

Out[44]:

	Area	New_or_Resale	Gymnasium	Lift_Available	Car_Parking	Clubhouse	Gas_Connec
208	675	1	0	1	1	0	

1 rows × 99 columns

In [45]:

```
regressor.predict(X_test[0:1])
```

Out[45]:

```
array([3528954.46971792])
```

In [46]:

```
def predict_price(Location,Area,New_or_Resale,Gymnasium,Lift_Available,Car_Parking,Club
house,Gas_Connection,Jogging_Track,Swimming_Pool,bhk):
    loc_index = np.where(X.columns==Location)[0][0]
    x = np.zeros(len(X.columns))
    x[0] = Area
    x[1] = New_or_Resale
    x[2] = Gymnasium
    x[3] = Lift_Available
    x[4] = Car_Parking
    x[5] = Clubhouse
    x[6] = Gas_Connection
    x[7] = Jogging_Track
    x[8] = Swimming_Pool
    x[9] = bhk
    if loc_index >= 0:
        x[loc_index] = 1

    return regressor.predict([x])[0]
```

In [47]:

```
#defining a function to predict the price based on input from user
def predict_prices(Location,Area,New_or_Resale,Gymnasium,Lift_Available,Car_Parking,Clu
bhouse,Gas_Connection,Jogging_Track,Swimming_Pool,bhk):
    index=0
    for i in X.columns:
        if(X.columns[index]==Location):
            loc_index=index
            break
        index=index+1
    x = np.zeros(len(X.columns))

    x[0] = Area
    x[1] = New_or_Resale
    x[2] = Gymnasium
    x[3] = Lift_Available
    x[4] = Car_Parking
    x[5] = Clubhouse
    x[6] = Gas_Connection
    x[7] = Jogging_Track
    x[8] = Swimming_Pool
    x[9] = bhk
    if loc_index >= 0:
        x[loc_index] = 1
    return regressor.predict([x])[0]
```

In [48]:

```
predict_price('Vashi',10000,0,1,0,1,1,0,1,1,2)
```

Out[48]:

```
196493296.48172125
```

In [49]:

```
predict_price('Kharghar',720,0,0,1,1,0,0,0,0,1)
```

Out[49]:

```
4637016.530318784
```

In [50]:

```
predict_prices('Kharghar',720,0,0,1,1,0,0,0,0,1)
```

Out[50]:

```
4637016.530318784
```

In [51]:

```
y_pred = regressor.predict(X_test)
```

In [52]:

```
y_pred
```

Out[52]:

```
array([ 3.52895447e+06,  1.86615029e+07,  2.26810451e+06,  2.32354080e+07,
        1.16977324e+07,  2.34262020e+07,  7.81683900e+06,  6.62551466e+07,
        9.27027264e+06,  4.14888025e+05,  7.29398750e+06,  2.56020026e+06,
        7.86807433e+06,  1.78748842e+07,  1.69850911e+07,  1.21117721e+07,
        7.24674509e+07,  3.79741992e+06,  3.24829110e+07,  7.95025103e+07,
        3.37859826e+07,  5.31249876e+05,  1.86329039e+07,  3.43131178e+07,
        7.03225395e+06,  1.86422483e+07,  2.53239583e+07,  1.65714992e+05,
        1.42235777e+07,  1.36391242e+07,  2.36787494e+07,  1.26893460e+07,
        2.07220452e+07,  6.99140531e+06,  1.51367686e+07,  3.14427669e+06,
        9.09134259e+06,  4.48432462e+06,  2.62736447e+06,  2.30952159e+06,
        5.75944769e+07,  9.91265239e+06,  7.29221457e+06,  1.40895775e+07,
        1.58523413e+07,  1.28387944e+07,  1.32952750e+07,  4.41182354e+06,
        1.27192096e+07,  8.56285258e+07,  1.67049713e+07,  2.08952517e+06,
        7.20981511e+06,  2.78259749e+06, -1.56471125e+06,  3.30788501e+06,
        1.96248042e+07,  1.25576640e+07,  8.61525226e+06,  8.62937703e+05,
        2.52731113e+07,  1.64300353e+07,  3.66957152e+06,  2.10064484e+07,
        2.09200810e+07,  1.09426899e+07,  2.81718164e+07,  2.11493696e+07,
       -3.29873780e+06,  9.52761131e+06,  2.30735088e+07,  1.03893891e+07,
        2.03436911e+06,  2.39741985e+07,  1.47656294e+07,  4.28509302e+07,
        1.77566759e+07,  3.55301764e+06,  1.47972912e+06,  5.37243395e+04,
        1.77886479e+07,  8.70068734e+06,  7.12876762e+06,  1.16551520e+07,
        1.35070673e+07,  1.62951393e+06,  1.43489600e+07,  2.05596997e+07,
        8.55199877e+06,  6.88561986e+06,  1.04804786e+07,  8.14334480e+06,
        6.51030214e+06,  9.33716414e+06,  2.77542056e+06,  1.70026475e+07,
        5.35601117e+06,  3.05135105e+06,  1.10104891e+07,  1.07673414e+07,
        1.79720456e+07,  5.53760796e+06,  9.49147773e+06,  8.98820960e+06,
        6.62114902e+06,  4.21179644e+06,  6.16577609e+06,  7.96822328e+06,
        8.70817940e+06,  2.29779060e+07,  2.09007220e+07,  2.45501056e+06,
        1.20871498e+07,  7.20085583e+06,  7.81180475e+06,  1.64532014e+07,
        3.07192910e+07,  2.14087777e+07,  6.66395566e+07,  2.06321575e+07,
        4.81838801e+07,  9.68427831e+06,  4.93365704e+07,  5.67961458e+06,
        1.18116972e+07,  5.50020798e+07,  8.00365554e+06,  1.42324644e+07,
        2.37796500e+06,  5.89358599e+06,  1.47208002e+06,  1.79332391e+07,
        1.11849084e+06,  1.46308963e+07,  1.20305468e+07,  3.76538783e+06,
        7.16305508e+05,  6.28807347e+06,  2.28986234e+07,  5.36276408e+06,
        2.90236673e+07,  1.66722375e+06,  2.49217207e+07,  9.15015710e+06,
        6.10420726e+06,  9.66449860e+06,  6.18654975e+06,  4.37979027e+06,
        3.45570118e+06,  3.11563602e+06,  2.83331689e+07,  6.14149508e+06,
        2.05035180e+07,  7.54517428e+06,  3.30788501e+06,  3.38757809e+07,
        5.02939240e+06,  1.62629387e+07,  2.35110382e+07,  9.52426046e+06,
        1.64864037e+07,  3.31868346e+06,  4.56727018e+06,  1.18650721e+07,
        2.63232674e+07, -2.18339611e+06,  8.89048844e+06,  2.03551482e+06,
        4.15732694e+06,  1.29546853e+06,  3.89211370e+06,  4.02502607e+06,
        1.38930846e+07,  2.58791157e+07,  6.34521854e+06,  1.15177206e+07,
        2.36947483e+07,  4.35626730e+07,  2.41699677e+07,  6.67071462e+06,
        5.99775577e+06,  2.64932038e+07,  6.95436085e+06,  3.79328805e+07,
        3.03118473e+06,  2.14521600e+06,  4.96826512e+06,  3.29246886e+07,
        2.72105054e+07,  1.06325056e+07,  4.10681213e+06,  7.84138926e+06,
        1.18083714e+07,  8.87208855e+06,  2.00199175e+07, -8.53532119e+05,
        1.55941805e+07,  7.81106252e+06,  1.92071266e+07,  1.78295772e+07,
        6.28937048e+06,  4.40843939e+06,  2.05399570e+07,  1.18521008e+06,
        1.39057315e+07,  2.28543962e+07,  8.06042383e+06,  1.41662413e+07,
        2.29191888e+06,  2.06503212e+07,  1.25858819e+07,  4.92886538e+07,
        1.15332317e+07,  2.80129819e+07,  5.40220438e+06,  9.15015710e+06,
        1.47427286e+07, -2.73631457e+06,  7.41440387e+06,  3.76070332e+07,
        6.62161920e+06,  4.62668558e+06,  8.27016156e+06,  1.81790509e+07,
        3.20227623e+07,  5.01235589e+06,  4.81838801e+07,  2.05581206e+07,
        4.14330735e+06,  1.48646341e+07,  4.47939685e+06,  2.73212237e+07,
        3.84827773e+07,  3.12494576e+07,  1.20553542e+07,  3.89674067e+07,
```

```
 1.58485845e+07,  1.19053925e+07,  1.51307679e+07,  8.17244664e+06,
 1.15245395e+07,  3.02013535e+06,  2.43173094e+07,  1.36595417e+07,
 8.99313807e+06,  4.63416838e+06,  3.03554529e+06,  9.84866564e+06,
 1.01866987e+07,  3.26243845e+07,  7.36045573e+06,  8.34760788e+06,
 9.80757628e+06,  1.34736678e+07,  9.43626422e+06,  1.80842000e+07,
 9.97266964e+06,  1.22097681e+07,  1.21284290e+07,  9.23118492e+05,
 5.74855163e+06,  1.17886666e+06,  2.10910394e+07,  2.34340673e+07,
 1.48125406e+07,  1.18970129e+07,  2.62500926e+07,  2.99539921e+07,
 7.87525987e+06,  2.47084443e+07,  9.40573047e+06,  2.86370126e+07,
 2.50494321e+06,  1.00000724e+07,  1.70077948e+07,  7.93937920e+07,
 1.42136625e+07,  1.74615198e+07, -1.11522265e+06,  5.63455250e+06,
 1.06531986e+07,  9.90617841e+06,  7.86592808e+06,  3.75733899e+06,
 8.26484604e+06,  3.26863643e+07,  1.50368256e+07,  2.29257260e+06,
 1.84024568e+07,  1.95162769e+07,  3.14427669e+06,  6.49388030e+06,
 2.04753284e+07,  1.29369390e+07,  2.84452704e+07,  2.03469271e+06,
 1.77648707e+07,  1.11305431e+07,  4.23808757e+07,  8.74295502e+05,
 1.03095449e+07,  1.62629387e+07,  1.29218115e+07,  2.47218827e+06,
 2.17784110e+07,  2.67155660e+07,  1.10034897e+07,  3.16694389e+06,
 1.82043386e+06,  5.61624100e+06,  2.36355996e+07,  2.15995184e+07,
 4.28305489e+07,  1.38467508e+07,  2.76326156e+07,  1.58726164e+06,
 1.55326590e+06,  2.01512211e+07,  3.16646145e+06,  2.26629153e+07,
 8.46906361e+06,  1.74529872e+07,  1.12826137e+07,  3.39280199e+07,
 2.78405988e+06, -3.95491452e+06,  1.26238870e+07,  2.93971388e+07,
 3.16308054e+07,  4.48610233e+07,  2.12237451e+07,  3.39009123e+06,
 1.36391242e+07,  1.53413968e+07,  1.75061498e+07,  7.93763065e+06,
 1.63719287e+07,  9.76999989e+06,  6.31698233e+06,  8.96778111e+06,
-6.46637987e+05,  5.93275493e+06,  3.58264513e+06,  3.16001941e+06,
 9.46882321e+06,  8.75057573e+06,  3.43058123e+06,  7.63402421e+06,
 3.32438511e+06,  1.69065553e+06,  3.30017525e+07,  3.05776080e+06,
 1.94834616e+07,  3.64098665e+06,  8.16142252e+06,  7.63423997e+06,
 6.35396227e+06,  1.01852893e+07,  3.28156886e+06,  5.24328366e+06,
 4.11484896e+06,  4.16658515e+07,  1.36272781e+06,  5.16166491e+06,
 1.22836012e+07,  3.88752062e+07,  1.32877867e+07,  6.21290791e+06,
 8.21373655e+06,  1.58798712e+07,  6.53521057e+06,  7.12209801e+06,
 1.46922902e+07,  6.44917720e+06,  3.59147785e+06,  2.80793200e+07,
 1.45940326e+07,  8.35401342e+06,  2.81106487e+06,  1.91180791e+07,
 1.92424273e+07,  2.52002283e+07,  9.05961737e+06,  1.07684981e+07,
 1.71366013e+07,  3.02129437e+07,  8.17275826e+06,  1.48946884e+07,
 3.92279289e+07,  2.74535874e+06,  5.13195833e+07,  6.90166417e+06,
 7.13304872e+06,  5.64488167e+06,  8.81633412e+06,  4.81380532e+06,
 1.69781840e+07,  1.56430485e+07,  1.12097924e+06,  1.43057805e+07,
 6.30837822e+06,  1.20669903e+07,  1.22392696e+07,  1.12102040e+07,
 9.01211101e+06,  3.14427669e+06,  6.70839461e+06,  4.29158435e+06,
 5.78045488e+06,  4.51029229e+06,  9.22374474e+06,  4.88592263e+06,
 1.14428802e+07,  9.86053584e+05,  9.58655347e+06,  4.10386823e+06,
 3.93467911e+06,  3.87025454e+06,  1.68118243e+07,  1.06713057e+07,
 2.79846988e+06,  4.91944213e+06,  7.74993736e+06,  1.89071346e+07,
-3.22865172e+06,  6.33676662e+06,  2.25496909e+06,  1.22566605e+07,
 9.04078846e+06,  4.22834191e+06, -1.64912545e+06,  5.00595036e+06,
 1.03321259e+07,  9.12359745e+06,  8.16449246e+06,  2.11036312e+06,
 1.09528346e+07,  8.82321118e+07,  1.27460493e+07,  1.57474917e+07,
 7.10451268e+06,  3.54051609e+06,  2.54227731e+06,  3.81848336e+06,
 1.49380687e+07,  8.24552853e+06,  1.15977425e+07,  3.88752062e+07,
 4.52544070e+06,  1.76499759e+07,  6.36602309e+07,  7.02523916e+05,
 3.26243845e+07,  2.10999142e+06,  1.09292701e+07,  1.56577525e+06,
 1.38135310e+07,  1.49638339e+07,  1.42764701e+07,  1.14917161e+07,
 1.12962502e+07,  2.29212539e+06,  2.78405988e+06,  3.15733633e+07,
-2.42465409e+05,  1.84257407e+07,  1.53362188e+07,  7.65741119e+07,
 1.89509273e+07,  9.87941495e+06,  1.12091884e+07,  2.70578481e+06,
 8.31699918e+06,  1.20850506e+07,  9.87491809e+06,  2.20988811e+07,
 4.38081192e+06,  1.95324012e+07,  8.79309660e+06,  1.44910330e+07,
```

```
8.19306300e+06,  7.68744588e+07,  1.97845600e+06,  1.05140393e+07,
6.36225038e+06,  1.32445156e+07,  2.37066455e+06, -6.09434136e+05,
1.56331697e+07,  1.03935174e+07,  1.19053925e+07,  1.98168701e+07,
2.12116064e+07,  1.55928379e+07,  1.49380687e+07,  2.39741985e+07,
1.02854659e+06,  1.28204212e+07,  1.16487768e+07,  1.99255707e+07,
2.65733255e+07,  1.06474253e+06,  2.67272362e+06,  4.66502900e+06,
4.09077290e+06,  1.22880980e+07,  2.47589278e+06,  9.30256882e+06,
5.19072439e+07,  1.37385608e+07,  9.98093867e+06,  9.94108324e+06,
6.62114902e+06,  2.25287511e+07,  2.01580611e+06,  3.08141188e+07,
2.18317738e+06,  9.06673866e+06,  3.57611488e+07,  2.27576324e+07,
2.03864196e+07,  1.06926542e+07,  1.51986362e+07,  2.51367063e+06,
3.21568446e+06,  1.08489051e+07,  1.58691188e+07,  7.12876762e+06,
1.22162756e+07,  8.32865985e+06,  5.94677277e+06,  7.77753280e+06,
1.13857852e+06,  9.90910000e+06,  7.23522617e+06,  2.41771266e+06,
2.40070036e+07,  2.70511359e+07,  1.76874747e+06,  1.20986594e+07,
2.82641957e+07,  2.58437594e+07,  5.54858917e+05,  3.50331541e+06,
7.86897793e+07,  4.58523438e+06,  2.57625404e+07,  6.82520203e+06,
3.00323864e+06, -5.23041371e+05,  6.63065594e+06,  6.92332150e+06,
3.29401017e+07,  4.28300343e+07,  1.13921038e+07,  3.40757069e+06,
3.21568446e+06,  1.22019900e+07,  2.73035857e+07,  1.66173089e+07,
2.58691214e+07,  1.02459498e+07,  2.01248223e+07,  1.46417631e+06,
3.56132310e+06,  1.93820723e+07,  1.69449641e+07,  3.66356908e+06,
1.46417631e+06,  3.70517255e+07,  1.47021927e+07,  1.84946574e+07,
1.17958951e+07,  3.01713389e+06,  2.25249633e+06,  9.13368580e+06,
1.15849105e+07,  1.20854815e+07,  2.17026317e+06,  1.26002564e+07,
8.22561697e+06,  3.94711324e+06,  1.29333595e+07,  1.94023456e+07,
2.21899497e+06,  2.31618894e+06,  4.28544347e+06,  5.74941488e+06,
3.10695996e+07,  2.20313347e+06,  2.19332822e+07,  3.41926776e+06,
9.48751550e+06,  6.99732742e+06,  2.78634305e+07,  2.72038698e+07,
8.63906615e+06,  1.63516239e+07,  2.19003305e+07,  2.50871505e+07,
1.29272155e+07,  7.41014527e+06,  1.15470517e+07,  7.58645356e+06,
9.90620614e+06,  4.41182354e+06,  1.02810130e+07,  3.16690144e+07,
2.04621093e+07,  1.02937363e+07,  1.49967176e+07,  1.14833117e+08,
1.10382950e+07,  1.40656366e+07,  3.52153820e+07,  2.62780723e+07,
1.01855146e+07,  6.03582108e+06,  4.54382142e+06,  1.03216754e+07,
8.91370736e+06,  1.71243315e+07,  9.02211885e+06,  1.70330182e+07,
1.11878387e+07,  1.67295721e+07,  6.29221987e+07,  3.81887287e+06,
2.37163976e+07,  1.75895533e+07,  1.08606340e+07,  8.07449613e+06,
4.29836620e+07,  2.91883802e+07,  7.23793316e+06,  1.52096964e+07,
1.30935418e+07,  7.76655061e+06,  5.27956295e+07, -2.14727116e+05,
1.15960515e+06,  2.47218827e+06,  5.07366878e+06,  1.11952280e+07,
8.70056944e+06,  5.63455250e+06,  1.31028649e+07,  4.36062558e+07,
8.23674874e+06,  1.33495547e+07,  3.14427669e+06,  8.25702957e+06,
2.19098723e+07,  1.42797764e+07,  3.28332714e+06,  2.17770362e+07,
6.73735578e+06,  3.12817130e+07,  3.89030382e+07,  1.67016435e+07,
2.52123555e+07,  5.39354123e+06,  1.65508042e+07,  2.19094709e+07,
1.10427919e+07,  1.33059124e+07,  2.17037852e+06,  1.33394478e+07,
1.13989984e+07,  1.83063704e+07,  2.51661700e+06,  4.23323619e+06,
1.03593258e+07,  1.21941615e+07,  6.31657787e+06,  5.57175481e+06,
9.23941799e+06,  6.82561058e+06,  1.87034641e+06,  7.57530403e+06,
9.04114211e+07,  6.22498396e+06,  9.23118492e+05,  1.87329031e+06,
9.47077433e+06,  5.78311969e+06,  3.21694920e+06,  1.05234995e+07,
9.83589335e+06,  3.89559498e+06,  3.79958185e+07,  1.18647481e+06,
6.59109595e+06,  2.88574276e+07,  1.23791047e+07,  2.64351143e+06,
9.91241661e+06,  2.22673872e+07,  1.42309046e+07,  7.51673689e+06,
2.13011196e+07,  1.58633677e+07,  8.39380301e+06,  7.93206218e+06,
2.03171093e+07,  1.73823746e+07,  1.17730397e+07,  7.05986679e+06,
2.84198105e+04,  2.23302005e+07,  1.07754268e+08,  2.19003305e+07,
1.42064968e+07,  1.50555496e+07,  1.81439529e+07,  7.15689364e+06,
4.71255317e+06,  2.26471035e+07,  1.17128196e+07,  2.52511591e+06,
7.31313180e+06,  1.87840818e+07,  5.65347674e+06,  2.57311362e+06,
```

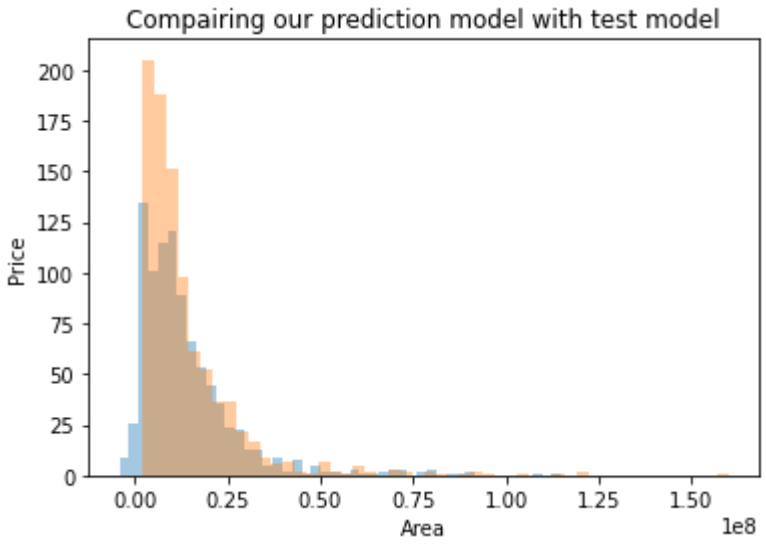
```
 1.10932027e+07,  2.18829244e+07,  1.49602776e+07,  4.92388647e+06,
 2.52296115e+07,  2.69866748e+06,  2.00269461e+07,  2.89985413e+06,
 9.68156749e+06,  2.94572612e+06,  1.09177451e+07,  3.07496128e+07,
 8.99063368e+07,  3.33779696e+07,  1.32089678e+07,  1.06746737e+07,
 3.12560032e+06,  6.95040525e+07,  6.88318803e+07,  6.15834620e+06,
 6.91869803e+07,  6.45849835e+06,  1.08063837e+07,  2.51442815e+06,
 2.44412519e+07,  1.28052549e+07,  2.39152244e+07,  4.82264566e+07,
 6.08002287e+06,  1.98084404e+07,  1.06085387e+07, -1.06525196e+06,
 1.12450328e+07,  4.40843939e+06,  1.13931683e+07,  1.73099670e+07,
 4.03237108e+06,  9.36729949e+06,  1.03915379e+07,  4.16212878e+06,
 5.20626942e+06,  1.77805374e+07,  8.23125040e+06,  1.51588687e+06,
 2.62012725e+06,  1.18056797e+07,  1.28597172e+07,  2.08850933e+07,
 1.88154534e+07,  3.20760617e+06,  1.31716485e+07,  1.05934126e+07,
 1.30445608e+07,  1.20620567e+07,  1.34406921e+07,  1.03490746e+07,
 2.43854822e+07,  1.12247335e+07,  5.50797590e+06,  2.59427620e+06,
-7.60571442e+05,  8.94954802e+06,  1.64732114e+07,  8.53744980e+06,
 4.89266865e+06,  7.64334738e+06,  1.54540856e+07,  3.14427669e+06,
 1.63716362e+06,  1.54869686e+07,  8.67462595e+06,  1.01242580e+07,
 2.91541755e+07,  7.20447235e+06,  2.64907080e+07,  6.20080321e+06,
 1.23400193e+07,  3.48241928e+07,  1.83306910e+07,  9.87218874e+06,
 1.36272781e+06,  2.75084410e+07,  2.44851511e+07,  3.40882217e+07,
-4.13765722e+06,  2.95196544e+07,  1.77082523e+07,  8.96800614e+06,
 1.06035425e+06,  7.20840580e+07,  1.52477016e+07,  5.00746253e+06,
 1.65546714e+07,  6.95221461e+06,  1.68919988e+07,  9.17665971e+06,
 1.28315136e+07,  5.94422787e+06,  7.14853595e+06,  1.56690839e+07,
 1.24123161e+07,  2.39000990e+07,  5.88127615e+07,  2.68971308e+06,
 5.50797590e+06, -2.12810337e+06,  1.97586202e+07,  9.91859005e+06,
 1.79435239e+06,  5.79446370e+06,  4.25920289e+06,  1.77779660e+07,
 1.62629387e+07,  3.31942973e+06,  3.60945109e+06,  4.41182354e+06,
 6.43741909e+06,  1.01995729e+07,  9.07302524e+06,  4.20539195e+06,
 9.95323094e+06,  1.66683821e+07,  1.42764701e+07,  1.00902909e+07,
 1.01137227e+07,  2.84452704e+07,  3.55037157e+06,  7.48971485e+04,
 1.05182423e+07,  2.93298746e+07,  6.02880857e+07,  7.24856515e+06,
 2.17656693e+07,  8.81380283e+06,  1.82356202e+07,  1.11381436e+07,
 1.04231991e+07,  1.91432969e+07,  1.17562277e+07,  1.18910102e+06,
 4.88185371e+06,  2.81641784e+06,  2.14327865e+07,  8.04749114e+06,
 2.72367143e+07,  1.73020884e+07,  1.15695489e+07,  1.05588518e+07,
 5.22513090e+06,  6.06128595e+06,  7.18719394e+07,  2.65163197e+07,
 5.97860551e+07,  9.87218874e+06, -1.15310426e+06,  1.47665919e+06,
 1.61628845e+07,  1.19187234e+07,  1.77566759e+07,  3.84225541e+06,
 2.37796500e+06,  7.78054638e+06,  3.08503510e+06,  1.59478098e+07,
 1.08073358e+07,  1.57953310e+07,  3.74863051e+06,  4.54882748e+07,
 4.60068311e+06,  9.37512798e+06,  6.06128595e+06,  1.56944825e+07,
 3.02961762e+06,  6.96611695e+06,  1.03371381e+07,  4.75937580e+06,
 2.14942357e+07,  1.33193245e+06,  1.02323798e+07,  7.67402768e+06,
 1.12374328e+07,  2.40618090e+06,  1.45623182e+07,  1.19494464e+07,
 1.02306022e+07,  1.55675602e+07,  1.02419882e+07,  4.29192341e+07,
 2.42667501e+07,  1.35037366e+07,  2.09460518e+07,  3.71191186e+07,
 7.40158553e+06,  9.86053584e+05,  7.87417199e+06,  2.16751550e+07,
 1.74965109e+07,  2.78141381e+06,  1.22254412e+07,  7.02724390e+06,
 2.03233729e+07])
```

In [53]:

```
#using a histogram to check accuracy of our model
sns.distplot(y_pred,  kde=False, label='Prediction')
sns.distplot(y_test,  kde=False,label='Test')
plt.legend
plt.title('Compairing our prediction model with test model')
plt.xlabel('Area')
plt.ylabel('Price')
```

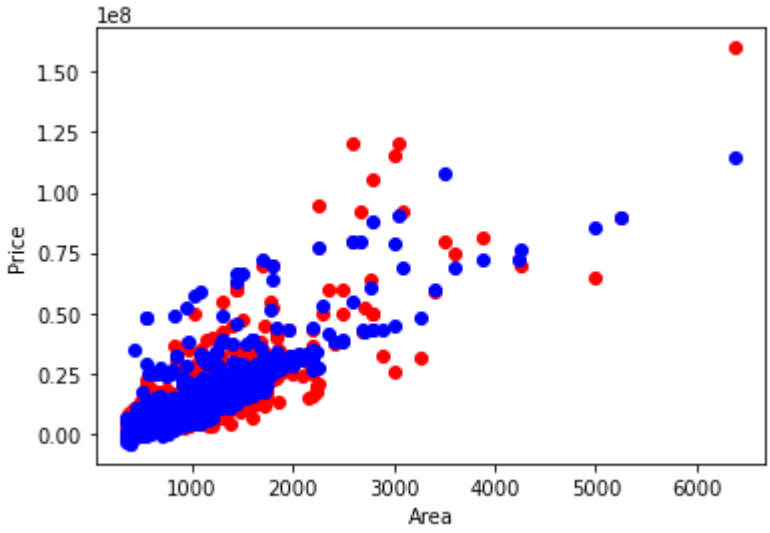
Out[53]:

```
Text(0, 0.5, 'Price')
```



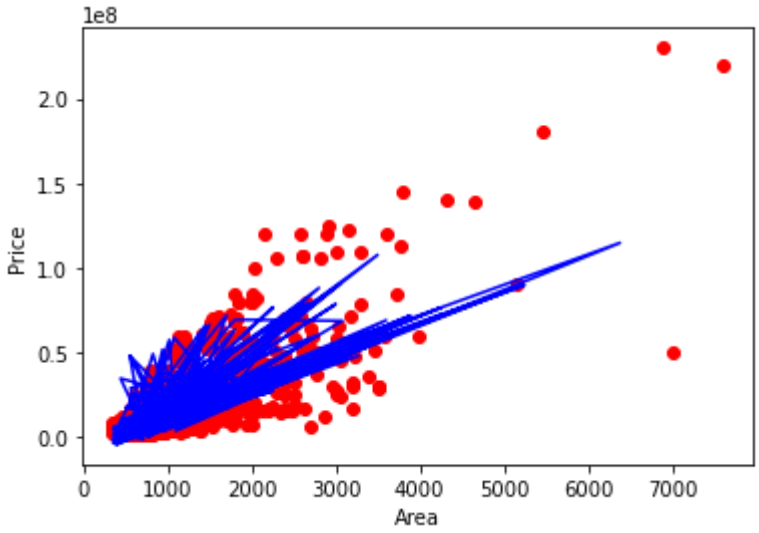
In [54]:

```
#comparing test values and predicted values
plt.scatter(X_test['Area'],y_test, color = 'red')
plt.scatter(X_test['Area'],y_pred, color = 'blue')
plt.xlabel('Area')
plt.ylabel('Price')
plt.show()
```



In [55]:

```
plt.scatter(X_train['Area'],y_train, color = 'red')
plt.plot(X_test['Area'],y_pred, color = 'blue')
plt.xlabel('Area')
plt.ylabel('Price')
plt.show()
```



In [56]:

```
#saving our model using pickle
import pickle
with open('realestate_price_model_mumbai.pickle','wb') as f:
    pickle.dump(regressor,f)
```

In [57]:

```
#saving name of columns in json
import json
columns = {
    'data_columns': [col.lower() for col in X.columns]
}
with open('locations.json','w') as f:
    f.write(json.dumps(columns))
```

In [ ]:

```

```