REAL_ESTATE_PROJECT IN BENGLORE CITY

In [ ]:

In [ ]:
```
'''1sqft = 12* 12 = 144'''
```

1. DATA CLEANING

In [1]:
```
import pandas as pd
data = pd.read_csv("C:\\Users\\sriha\\OneDrive\\Desktop\\pb excel\\Bengaluru_Hou
df = pd.DataFrame(data)
df.head()
```

Out[1]:

	area_type	availability	location	size	society	total_sqft	bath	balco
0	Super built-up Area	19-Dec	Electronic City Phase II	2 BHK	Coomee	1056	2.0	
1	Plot Area	Ready To Move	Chikka Tirupathi	4 Bedroom	Theanmp	2600	5.0	
2	Built-up Area	Ready To Move	Uttarahalli	3 BHK	NaN	1440	2.0	
3	Super built-up Area	Ready To Move	Lingadheeranahalli	3 BHK	Soiewre	1521	3.0	
4	Super built-up Area	Ready To Move	Kothanur	2 BHK	NaN	1200	2.0	

In [2]:
```
df.shape
```

Out[2]:
```
(13320, 9)
```

In [3]:
```
df.area_type. unique()
```

Out[3]:
```
array(['Super built-up  Area', 'Plot  Area', 'Built-up  Area',
       'Carpet  Area'], dtype=object)
```

In [4]:
```
len(df.area_type.unique())
```

Out[4]:
```
4
```

In [5]:
```
df.groupby('area_type')['area_type'].agg('count')
```

Out[5]:
```
area_type
Built-up  Area          2418
Carpet  Area              87
Plot  Area              2025
Super built-up  Area    8790
Name: area_type, dtype: int64
```

In [6]:
```
df.isnull().sum()
```

Out[6]:
```
area_type          0
availability       0
location           1
size              16
society         5502
total_sqft         0
bath              73
balcony          609
price              0
dtype: int64
```

In [7]:
```
df1 =df.drop (['availability','society','balcony','area_type'], axis ="columns")
df1.head()
```

Out[7]:

	location	size	total_sqft	bath	price
0	Electronic City Phase II	2 BHK	1056	2.0	39.07
1	Chikka Tirupathi	4 Bedroom	2600	5.0	120.00
2	Uttarahalli	3 BHK	1440	2.0	62.00
3	Lingadheeranahalli	3 BHK	1521	3.0	95.00
4	Kothanur	2 BHK	1200	2.0	51.00

In [8]:
```
# finding null values
df1.isnull().sum()
```

Out[8]:
```
location       1
size          16
total_sqft     0
bath          73
price          0
dtype: int64
```

In [9]:
```
'''' there is null values are 90
    remove that null values
'''

df2= df1.dropna()
df2.isnull().sum()
```

Out[9]:
```
location      0
size          0
total_sqft    0
bath          0
price         0
dtype: int64
```

In [10]:
```
df.shape
```

Out[10]:
```
(13320, 9)
```

In [11]:
```
df1.shape
```

Out[11]:
```
(13320, 5)
```

In [12]:
```
df2.shape       # 13320 - 13246 = 74
```

Out[12]:
```
(13246, 5)
```

In [13]:
```
df2.head()
```

Out[13]:

	location	size	total_sqft	bath	price
0	Electronic City Phase II	2 BHK	1056	2.0	39.07
1	Chikka Tirupathi	4 Bedroom	2600	5.0	120.00
2	Uttarahalli	3 BHK	1440	2.0	62.00
3	Lingadheeranahalli	3 BHK	1521	3.0	95.00
4	Kothanur	2 BHK	1200	2.0	51.00

In [14]:
```
df2.location.unique()
```

Out[14]:
```
array(['Electronic City Phase II', 'Chikka Tirupathi', 'Uttarahalli', ...,
       '12th cross srinivas nagar banshankari 3rd stage',
       'Havanur extension', 'Abshot Layout'], dtype=object)
```

In [15]:
```
len(df2.location.unique())
```

Out[15]:
```
1304
```

In [16]:
```
df2['size'].unique()
```

Out[16]:
```
array(['2 BHK', '4 Bedroom', '3 BHK', '4 BHK', '6 Bedroom', '3 Bedroom',
       '1 BHK', '1 RK', '1 Bedroom', '8 Bedroom', '2 Bedroom',
       '7 Bedroom', '5 BHK', '7 BHK', '6 BHK', '5 Bedroom', '11 BHK',
       '9 BHK', '9 Bedroom', '27 BHK', '10 Bedroom', '11 Bedroom',
       '10 BHK', '19 BHK', '16 BHK', '43 Bedroom', '14 BHK', '8 BHK',
       '12 Bedroom', '13 BHK', '18 Bedroom'], dtype=object)
```

In [17]:
```
''' Now here is problem that
    that, I want to separate the BHK numbers
                    separate the BHK charcters'''

df2['BHK'] = df2['size'].apply (lambda x : int(x.split(' ')[0]))
df2.head()
```

```
c:\users\sriha\appdata\local\programs\python\python37\lib\site-packages\ipykernel
_launcher.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  """
```

Out[17]:

	location	size	total_sqft	bath	price	BHK
0	Electronic City Phase II	2 BHK	1056	2.0	39.07	2
1	Chikka Tirupathi	4 Bedroom	2600	5.0	120.00	4
2	Uttarahalli	3 BHK	1440	2.0	62.00	3
3	Lingadheeranahalli	3 BHK	1521	3.0	95.00	3
4	Kothanur	2 BHK	1200	2.0	51.00	2

In [18]:
```
df2.BHK.unique()   # there are number of BHK houses are there
```

Out[18]:
```
array([ 2,  4,  3,  6,  1,  8,  7,  5, 11,  9, 27, 10, 19, 16, 43, 14, 12,
       13, 18], dtype=int64)
```

In [19]:
```
# now lets check above BHK> 20
df2[df2.BHK> 20]
```

Out[19]:

	location	size	total_sqft	bath	price	BHK
1718	2Electronic City Phase II	27 BHK	8000	27.0	230.0	27
4684	Munnekollal	43 Bedroom	2400	40.0	660.0	43

In [20]:
```
df2.total_sqft.unique()
```

Out[20]:
```
array(['1056', '2600', '1440', ..., '1133 - 1384', '774', '4689'],
      dtype=object)
```

In [21]:
```
''' Now here a problem that all sqft_values are numbers BUT,
                    some are     sqft_values are IN BETWEEN values

    Let find how many there are'''

df3= df2.copy()
df3.head()
```

Out[21]:

	location	size	total_sqft	bath	price	BHK
0	Electronic City Phase II	2 BHK	1056	2.0	39.07	2
1	Chikka Tirupathi	4 Bedroom	2600	5.0	120.00	4
2	Uttarahalli	3 BHK	1440	2.0	62.00	3
3	Lingadheeranahalli	3 BHK	1521	3.0	95.00	3
4	Kothanur	2 BHK	1200	2.0	51.00	2

In [22]:
```
df3.drop('size', axis ="columns", inplace=True)
```

In [23]:
```
df3.head()
```

Out[23]:

	location	total_sqft	bath	price	BHK
0	Electronic City Phase II	1056	2.0	39.07	2
1	Chikka Tirupathi	2600	5.0	120.00	4
2	Uttarahalli	1440	2.0	62.00	3
3	Lingadheeranahalli	1521	3.0	95.00	3
4	Kothanur	1200	2.0	51.00	2

In [24]:
```
# let check how many values are '1133 - 1384' like between values
def btwn_values(x):
    try:
        float(x)
    except:
        return False
    return True

df3[~(df3.total_sqft.apply (btwn_values))]
```

Out[24]:

	location	total_sqft	bath	price	BHK
30	Yelahanka	2100 - 2850	4.0	186.000	4
122	Hebbal	3067 - 8156	4.0	477.000	4
137	8th Phase JP Nagar	1042 - 1105	2.0	54.005	2
165	Sarjapur	1145 - 1340	2.0	43.490	2
188	KR Puram	1015 - 1540	2.0	56.800	2
...	...	...	...	...	...
12975	Whitefield	850 - 1060	2.0	38.190	2
12990	Talaghattapura	1804 - 2273	3.0	122.000	3
13059	Harlur	1200 - 1470	2.0	72.760	2
13265	Hoodi	1133 - 1384	2.0	59.135	2
13299	Whitefield	2830 - 2882	5.0	154.500	4

190 rows × 5 columns

In [25]:
```
# there are 190 values there
'''So, that values I dont want to remove Instead of
   Give Average for that values like
   (2100 + 2850)/ 2 ===> 2950/ 2====> 2475
   '''
   #  Now these values to convert into AVERAGE values
#  Like (x+y)/2
```

```
def convert_btwn_values(x):
    splits = x.split('-')
    if len (splits)==2:
        return (float (splits[0]) + float(splits[1])) / 2
    try:
        return float(x)
    except :
        return None
```

In [26]:
```
convert_btwn_values('1200 - 1470')    # 2670/2  ====> 1335
```

Out[26]:
```
1335.0
```

In [27]:
```
convert_btwn_values('2100 - 2850')
```

Out[27]:
```
2475.0
```

In [28]:
```
convert_btwn_values('1056')
```

Out[28]:
```
1056.0
```

In [29]:
```
# Now I want  these average  values into  main data
df4 = df3.copy()
df4.total_sqft = df4['total_sqft'] .apply(convert_btwn_values)
```

In [30]:
```
df4.total_sqft.unique()
```

Out[30]:
```
array([1056. , 2600. , 1440. , ..., 1258.5,  774. , 4689. ])
```

In [31]:
```
df4.loc[122]
```

Out[31]:
```
location      Hebbal
total_sqft    5611.5
bath               4
price            477
BHK                4
Name: 122, dtype: object
```

In [32]:
```
df4.head()
```

Out[32]:

	location	total_sqft	bath	price	BHK
0	Electronic City Phase II	1056.0	2.0	39.07	2
1	Chikka Tirupathi	2600.0	5.0	120.00	4
2	Uttarahalli	1440.0	2.0	62.00	3
3	Lingadheeranahalli	1521.0	3.0	95.00	3
4	Kothanur	1200.0	2.0	51.00	2

In [ ]:

In [ ]:

2. FUTURE ENGINEERING

In [33]:
```
'''1. To find price_Per_sqft.
   2. Location no:of places '''

df4.head()
```

Out[33]:

	location	total_sqft	bath	price	BHK
0	Electronic City Phase II	1056.0	2.0	39.07	2
1	Chikka Tirupathi	2600.0	5.0	120.00	4
2	Uttarahalli	1440.0	2.0	62.00	3
3	Lingadheeranahalli	1521.0	3.0	95.00	3
4	Kothanur	1200.0	2.0	51.00	2

In [34]:
```
df4.bath.unique()
```

Out[34]:
```
array([ 2.,  5.,  3.,  4.,  6.,  1.,  9.,  8.,  7., 11., 10., 14., 27.,
       12., 16., 40., 15., 13., 18.])
```

In [35]:
```
df4[df4.bath> 10]
```

Out[35]:

	location	total_sqft	bath	price	BHK
938	5th Phase JP Nagar	1260.0	11.0	290.0	9
1078	BTM 1st Stage	3300.0	14.0	500.0	9
1718	2Electronic City Phase II	8000.0	27.0	230.0	27
1768	1 Ramamurthy Nagar	1200.0	11.0	170.0	11
1953	KR Puram	1200.0	12.0	110.0	8
1979	Hongasandra	990.0	12.0	120.0	8
3096	Jp nagar 8th Phase .	12000.0	12.0	525.0	10
3379	1Hanuman Nagar	2000.0	16.0	490.0	19
3609	Koramangala Industrial Layout	10000.0	16.0	550.0	16
4684	Munnekollal	2400.0	40.0	660.0	43
4916	1Channasandra	1250.0	15.0	125.0	14
6937	5th Block Hbr Layout	2600.0	12.0	675.0	9
7979	1 Immadihalli	6000.0	12.0	150.0	11
8106	Wilson Garden	1850.0	12.0	300.0	8
8636	Neeladri Nagar	4000.0	12.0	160.0	10
9935	1Hoysalanagar	5425.0	13.0	275.0	13
10695	Electronic City	1200.0	13.0	150.0	9
11128	Jigani	1200.0	11.0	105.0	10
11559	1Kasavanhalli	1200.0	18.0	200.0	18
13067	Defence Colony	7150.0	13.0	3600.0	10

In [36]:
```
len(df4[df4.bath>=10])
```

Out[36]: 33

In [37]:
```
#  To find the Number of Location

len(df4.location.unique())
```

Out[37]: 1304

In [38]:
```
df4.location = df4.location.apply(lambda x : x.strip())
```

In [39]:
```
location_status = df4.groupby('location')['location'].agg('count').sort_values(a
location_status
```

Out[39]:
```
location
Whitefield           535
Sarjapur  Road       392
Electronic City      304
Kanakpura Road       266
Thanisandra          236
                    ...
LIC Colony             1
Kuvempu Layout         1
Kumbhena Agrahara      1
Kudlu Village,         1
1 Annasandrapalya      1
Name: location, Length: 1293, dtype: int64
```

In [40]:
```
location_status[location_status<=10]
```

Out[40]:
```
location
BTM 1st Stage          10
Basapura               10
Sector 1 HSR Layout    10
Naganathapura          10
Kalkere                10
                       ..
LIC Colony              1
Kuvempu Layout          1
Kumbhena Agrahara       1
Kudlu Village,          1
1 Annasandrapalya       1
Name: location, Length: 1052, dtype: int64
```

In [41]:
```
# Now I gonna to create all which is less 10 places that  named as "others"
location_less_than_10 = location_status[location_status<=10]
location_less_than_10
```

Out[41]:
```
location
BTM 1st Stage          10
Basapura               10
Sector 1 HSR Layout    10
Naganathapura          10
Kalkere                10
                       ..
LIC Colony              1
Kuvempu Layout          1
Kumbhena Agrahara       1
Kudlu Village,          1
1 Annasandrapalya       1
Name: location, Length: 1052, dtype: int64
```

In [42]:
```
df4.location = df4.location.apply (lambda x : "OTHER <=10 " if x in location_les
df4.head(20)
```

Out[42]:

	location	total_sqft	bath	price	BHK
0	Electronic City Phase II	1056.0	2.0	39.07	2
1	Chikka Tirupathi	2600.0	5.0	120.00	4
2	Uttarahalli	1440.0	2.0	62.00	3
3	Lingadheeranahalli	1521.0	3.0	95.00	3
4	Kothanur	1200.0	2.0	51.00	2
5	Whitefield	1170.0	2.0	38.00	2
6	Old Airport Road	2732.0	4.0	204.00	4
7	Rajaji Nagar	3300.0	4.0	600.00	4
8	Marathahalli	1310.0	3.0	63.25	3
9	OTHER <=10	1020.0	6.0	370.00	6
10	Whitefield	1800.0	2.0	70.00	3
11	Whitefield	2785.0	5.0	295.00	4
12	7th Phase JP Nagar	1000.0	2.0	38.00	2
13	Gottigere	1100.0	2.0	40.00	2
14	Sarjapur	2250.0	3.0	148.00	3
15	Mysore Road	1175.0	2.0	73.50	2
16	Bisuvanahalli	1180.0	3.0	48.00	3
17	Raja Rajeshwari Nagar	1540.0	3.0	60.00	3
18	OTHER <=10	2770.0	4.0	290.00	3
19	OTHER <=10	1100.0	2.0	48.00	2

In [43]:
```
# Here Comes the houses and apartments are ehich is less than 10 floors apartmen
# are written as '1-10 floors apartment'
```

In [ ]:
```

```

In [44]:
```
'''Lets find out the Price_per_sqft  for total_sqft'''

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

Out[44]:

	location	total_sqft	bath	price	BHK
0	Electronic City Phase II	1056.0	2.0	39.07	2
1	Chikka Tirupathi	2600.0	5.0	120.00	4
2	Uttarahalli	1440.0	2.0	62.00	3
3	Lingadheeranahalli	1521.0	3.0	95.00	3
4	Kothanur	1200.0	2.0	51.00	2

In [45]:
```
df5['price_per_sqft'] = df5['price'] * 100000/ df5['total_sqft']
df5.head()
```

Out[45]:

	location	total_sqft	bath	price	BHK	price_per_sqft
0	Electronic City Phase II	1056.0	2.0	39.07	2	3699.810606
1	Chikka Tirupathi	2600.0	5.0	120.00	4	4615.384615
2	Uttarahalli	1440.0	2.0	62.00	3	4305.555556
3	Lingadheeranahalli	1521.0	3.0	95.00	3	6245.890861
4	Kothanur	1200.0	2.0	51.00	2	4250.000000

In [46]:
```
df5.shape
```

Out[46]:
```
(13246, 6)
```

In [ ]:

In [ ]:

In [ ]:

3. OUTLIERS REMOVAL

In [47]:
```
#   1. FIND total_sqft/ BHKs  per BHK
#   2. REMOVE THE LESS AND HIGH MEAN VALUES OF  MEAN AND STD
```

In [48]:
```
df6 = df5.copy()
df6.head()
```

Out[48]:

	location	total_sqft	bath	price	BHK	price_per_sqft
0	Electronic City Phase II	1056.0	2.0	39.07	2	3699.810606
1	Chikka Tirupathi	2600.0	5.0	120.00	4	4615.384615
2	Uttarahalli	1440.0	2.0	62.00	3	4305.555556
3	Lingadheeranahalli	1521.0	3.0	95.00	3	6245.890861
4	Kothanur	1200.0	2.0	51.00	2	4250.000000

In [49]:
```
df6['total_sqft/BHKs'] = df6['total_sqft'] / df6['BHK']
df6.head()
```

Out[49]:

	location	total_sqft	bath	price	BHK	price_per_sqft	total_sqft/BHKs
0	Electronic City Phase II	1056.0	2.0	39.07	2	3699.810606	528.0
1	Chikka Tirupathi	2600.0	5.0	120.00	4	4615.384615	650.0
2	Uttarahalli	1440.0	2.0	62.00	3	4305.555556	480.0
3	Lingadheeranahalli	1521.0	3.0	95.00	3	6245.890861	507.0
4	Kothanur	1200.0	2.0	51.00	2	4250.000000	600.0

In [50]:
```
'''The term "1BHK" or "one-bedroom house" refers to a home with
just one bedroom, one hall, one kitchen, one bathroom, and one dining/living roo
which together take up between 450 and 600 square feet of carpet space'''

# That way I want to remove which is less than total_sqft/BHK <=300
df6[df6['total_sqft/BHKs'] <=300]
```

Out[50]:

	location	total_sqft	bath	price	BHK	price_per_sqft	total_sqft/BHKs
9	OTHER <=10	1020.0	6.0	370.0	6	36274.509804	170.000000
45	HSR Layout	600.0	9.0	200.0	8	33333.333333	75.000000
58	Murugeshpalya	1407.0	4.0	150.0	6	10660.980810	234.500000
68	Devarachikkanahalli	1350.0	7.0	85.0	8	6296.296296	168.750000
70	OTHER <=10	500.0	3.0	100.0	3	20000.000000	166.666667
...	...	...	...	...	...	...	...
13281	Margondanahalli	1375.0	5.0	125.0	5	9090.909091	275.000000
13300	Hosakerehalli	1500.0	6.0	145.0	5	9666.666667	300.000000
13303	Vidyaranyapura	774.0	5.0	70.0	5	9043.927649	154.800000
13306	OTHER <=10	1200.0	5.0	325.0	4	27083.333333	300.000000
13311	Ramamurthy Nagar	1500.0	9.0	250.0	7	16666.666667	214.285714

926 rows × 7 columns

In [51]:
```
#The lookout type of table and there are nearly 936 values are there,
# so I want to remove that  which is <= 300 from my table..

df7= df6[~(df6['total_sqft/BHKs']<= 300)]
df7.shape
```

Out[51]:
```
(12320, 7)
```

In [52]:
```
df6.shape
```

Out[52]: (13246, 7)

In [53]:
```
13246-12320
```

Out[53]: 926

In [54]:
```
df7.head()
```

Out[54]:

	location	total_sqft	bath	price	BHK	price_per_sqft	total_sqft/BHKs
0	Electronic City Phase II	1056.0	2.0	39.07	2	3699.810606	528.0
1	Chikka Tirupathi	2600.0	5.0	120.00	4	4615.384615	650.0
2	Uttarahalli	1440.0	2.0	62.00	3	4305.555556	480.0
3	Lingadheeranahalli	1521.0	3.0	95.00	3	6245.890861	507.0
4	Kothanur	1200.0	2.0	51.00	2	4250.000000	600.0

In [55]:
```
df8= df7.copy()
df8.drop('total_sqft/BHKs', axis="columns", inplace=True)
df8.head()
```

Out[55]:

	location	total_sqft	bath	price	BHK	price_per_sqft
0	Electronic City Phase II	1056.0	2.0	39.07	2	3699.810606
1	Chikka Tirupathi	2600.0	5.0	120.00	4	4615.384615
2	Uttarahalli	1440.0	2.0	62.00	3	4305.555556
3	Lingadheeranahalli	1521.0	3.0	95.00	3	6245.890861
4	Kothanur	1200.0	2.0	51.00	2	4250.000000

In [56]:
```
df8.price_per_sqft.describe()
```

Out[56]:
```
count     12274.000000
mean       6211.880230
std        4053.214807
min         267.829813
25%        4200.000000
50%        5263.157895
75%        6825.474875
max      176470.588235
Name: price_per_sqft, dtype: float64
```

In [ ]:

In [ ]:

This important in Price_prediction

In [ ]:

In [57]:
```
# According to Middle class they cam't buy 'Low' and 'Very Much High' , so all a
# That way I want to remove all  'min' and 'Max' values.
#  so, I want  to remove the all less  MEAN and SD values.
import pandas as pd
def  outlier_removal(df):
    df_out = pd.DataFrame()
    for keys , sub_df in df.groupby('location'):

        mean = np.mean(sub_df.price_per_sqft)
        std  = np.std (sub_df.price_per_sqft)

        removal_df =sub_df[(sub_df.price_per_sqft > (mean-std )) & (sub_df.price
        df_out = pd.concat([removal_df, df_out] ,ignore_index= True)

    return df_out
```

In [58]:
```
df8.shape  # Before removing
```

Out[58]:
```
(12320, 6)
```

In [59]:
```
df9 = outlier_removal(df8)  # after removing
df9.shape
```

Out[59]:
```
(10016, 6)
```

There are nearly 2000 values are removed.....

now when compared to same BHK House size is same but, the Price is also a different example --> uttarhalli location ,the total_sqft of house is 2BHK pricer is 81,000 same location the total_sqft of house is 2BHK pricer is 1,21,000

```
The price is totaly different


Like that we find like similar cases in datasets


By using Scatter diagrams we find what are simliar BHKs
and Prices are the different
```

In [60]:
```
df9.head()
```

Out[60]:

	location	total_sqft	bath	price	BHK	price_per_sqft
0	Yeshwanthpur	2502.0	3.0	138.00	3	5515.587530
1	Yeshwanthpur	1693.0	3.0	108.00	3	6379.208506
2	Yeshwanthpur	667.0	1.0	36.85	1	5524.737631
3	Yeshwanthpur	1950.0	4.0	130.00	4	6666.666667
4	Yeshwanthpur	1170.0	2.0	57.00	2	4871.794872

In [61]:
```
df9.location.unique()
```

Out[61]:
```
array(['Yeshwanthpur', 'Yelenahalli', 'Yelahanka New Town', 'Yelahanka',
       'Yelachenahalli', 'Whitefield', 'Vittasandra',
       'Vishwapriya Layout', 'Vishveshwarya Layout', 'Vijayanagar',
       'Vidyaranyapura', 'Vasanthapura', 'Varthur Road', 'Varthur',
       'Uttarahalli', 'Ulsoor', 'Tumkur Road', 'Tindlu',
       'Thyagaraja Nagar', 'Thubarahalli', 'Thigalarapalya',
       'Thanisandra', 'Talaghattapura', 'TC Palaya', 'Sultan Palaya',
       'Subramanyapura', 'Sonnenahalli', 'Sompura', 'Somasundara Palya',
       'Singasandra', 'Shivaji Nagar', 'Shampura', 'Seegehalli',
       'Sector 7 HSR Layout', 'Sector 2 HSR Layout',
       'Sarjapura - Attibele Road', 'Sarjapur  Road', 'Sarjapur',
       'Sarakki Nagar', 'Sanjay nagar', 'Sahakara Nagar', 'Rayasandra',
       'Ramamurthy Nagar', 'Ramagondanahalli', 'Rajiv Nagar',
       'Rajaji Nagar', 'Raja Rajeshwari Nagar', 'Rachenahalli',
       'R.T. Nagar', 'Prithvi Layout', 'Poorna Pragna Layout',
       'Pattandur Agrahara', 'Parappana Agrahara', 'Panathur',
       'Pai Layout', 'Padmanabhanagar', 'Old Madras Road',
       'Old Airport Road', 'OTHER <=10 ', 'OMBR Layout', 'Nehru Nagar',
       'Neeladri Nagar', 'Narayanapura', 'Nagavarapalya', 'Nagavara',
       'Nagasandra', 'Nagarbhavi', 'NRI Layout', 'NGR Layout',
       'Mysore Road', 'Murugeshpalya', 'Munnekollal', 'Mico Layout',
       'Marsur', 'Margondanahalli', 'Marathahalli', 'Malleshwaram',
       'Malleshpalya', 'Mallasandra', 'Mahalakshmi Layout', 'Mahadevpura',
       'Magadi Road', 'Lingadheeranahalli', 'Lakshminarayana Pura',
       'Laggere', 'LB Shastri Nagar', 'Kundalahalli',
       'Kumaraswami Layout', 'Kudlu Gate', 'Kudlu', 'Kothanur',
       'Kothannur', 'Koramangala', 'Konanakunte', 'Kogilu', 'Kodihalli',
       'Kodigehalli', 'Kodigehaali', 'Kodichikkanahalli',
       'Kereguddadahalli', 'Kengeri Satellite Town', 'Kengeri',
       'Kenchenahalli', 'Kaval Byrasandra', 'Kathriguppe',
       'Kasturi Nagar', 'Kasavanhalli', 'Karuna Nagar', 'Kannamangala',
       'Kanakpura Road', 'Kanakapura', 'Kammasandra', 'Kammanahalli',
       'Kambipura', 'Kalyan nagar', 'Kalena Agrahara', 'Kaikondrahalli',
       'Kaggalipura', 'Kaggadasapura', 'Kadugodi', 'Kadubeesanahalli',
       'KR Puram', 'Judicial Layout', 'Jigani', 'Jalahalli East',
       'Jalahalli', 'Jakkur', 'JP Nagar', 'Indira Nagar', 'Iblur Village',
       'ITPL', 'ISRO Layout', 'Hulimavu', 'Hosur Road', 'Hoskote',
       'Hosakerehalli', 'Hosa Road', 'Hormavu', 'Horamavu Banaswadi',
       'Horamavu Agara', 'Hoodi', 'Hennur Road', 'Hennur', 'Hegde Nagar',
       'Hebbal Kempapura', 'Hebbal', 'Harlur', 'Haralur Road',
       'HSR Layout', 'HRBR Layout', 'HBR Layout', 'HAL 2nd Stage',
       'Gunjur', 'Gubbalala', 'Green Glen Layout', 'Gottigere',
       'Gollarapalya Hosahalli', 'Giri Nagar', 'Garudachar Palya',
       'GM Palaya', 'Frazer Town', 'Electronics City Phase 1',
       'Electronic City Phase II', 'Electronic City', 'EPIP Zone',
       'Dommasandra', 'Domlur', 'Doddathoguru', 'Doddakallasandra',
       'Doddaballapur', 'Dodda Nekkundi', 'Devarachikkanahalli',
       'Devanahalli', 'Dasarahalli', 'Dasanapura', 'Cunningham Road',
       'Cox Town', 'Cooke Town', 'Choodasandra', 'Chikkalasandra',
       'Chikkabanavar', 'Chikka Tirupathi', 'Channasandra', 'Chandapura',
       'Chamrajpet', 'CV Raman Nagar', 'Budigere', 'Brookefield',
       'Bommenahalli', 'Bommasandra Industrial Area', 'Bommasandra',
       'Bommanahalli', 'Bisuvanahalli', 'Binny Pete', 'Billekahalli',
       'Bhoganhalli', 'Bharathi Nagar', 'Benson Town', 'Bellandur',
       'Begur Road', 'Begur', 'Battarahalli', 'Basaveshwara Nagar',
       'Basavangudi', 'Bannerghatta Road', 'Bannerghatta',
       'Banjara Layout', 'Banaswadi', 'Banashankari Stage VI',
       'Banashankari Stage V', 'Banashankari Stage III',
       'Banashankari Stage II', 'Banashankari', 'Balagere',
       'Badavala Nagar', 'Babusapalaya', 'BTM Layout', 'BTM 2nd Stage',
```

```
'BEML Layout', 'Attibele', 'Arekere', 'Ardendale', 'Anjanapura',
'Anekal', 'Ananth Nagar', 'Anandapura', 'Amruthahalli',
'Ambedkar Nagar', 'Ambalipura', 'Akshaya Nagar', 'Abbigere',
'AECS Layout', '9th Phase JP Nagar', '8th Phase JP Nagar',
'7th Phase JP Nagar', '6th Phase JP Nagar', '5th Phase JP Nagar',
'5th Block Hbr Layout', '2nd Stage Nagarbhavi',
'2nd Phase Judicial Layout', '1st Phase JP Nagar',
'1st Block Jayanagar'], dtype=object)
```

In [62]:
```
len(df9.location.unique())
```

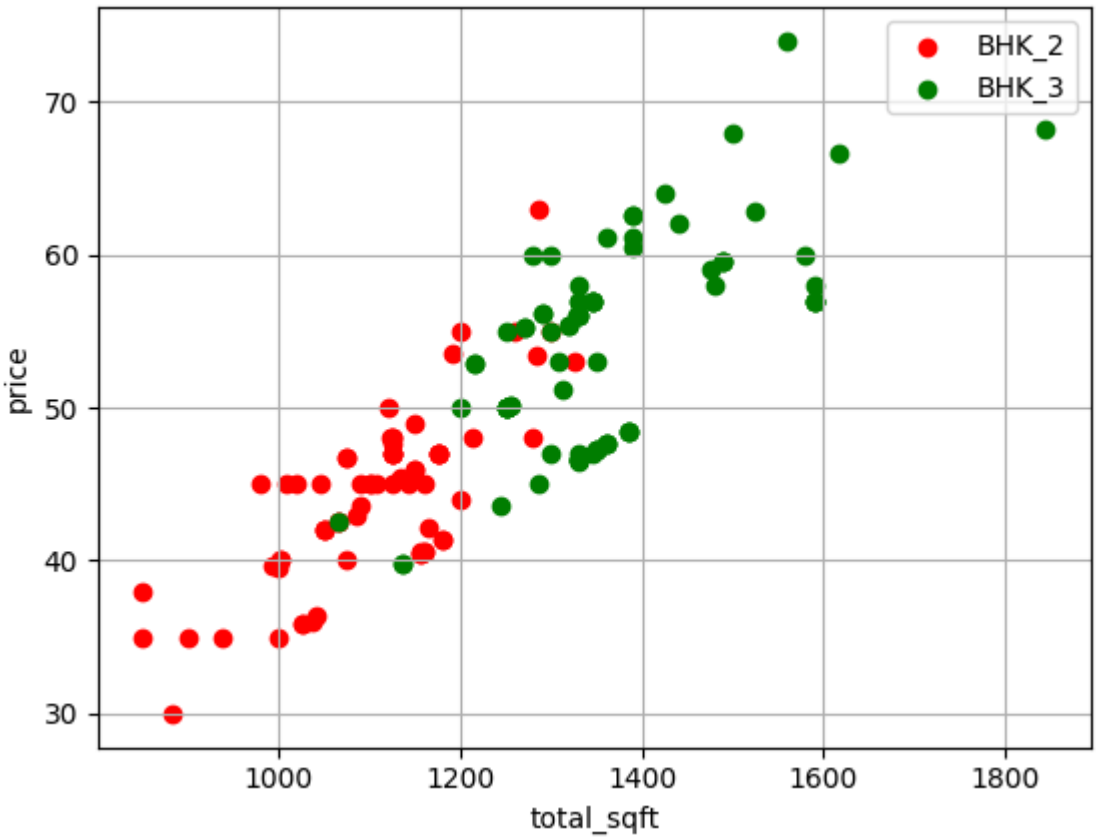
Out[62]: 242

In [ ]:
```
```

In [63]:
```
# By we see the 3BHK and 2BHK houses are using Visual representation

import matplotlib.pyplot as pp
def scatter_plot(df,location):
    bhk_2 = df[(df.location==location) & (df.BHK==2)]
    bhk_3 = df[(df.location==location) & (df.BHK==3)]
    pp.scatter (bhk_2.total_sqft, bhk_2.price , c="red", label= "BHK_2")
    pp.scatter (bhk_3.total_sqft, bhk_3.price , c="green", label = "BHK_3")
    pp.legend()
    pp.xlabel('total_sqft')
    pp.ylabel('price')
    pp.grid()
    pp.show()
```

In [64]:
```
scatter_plot(df9,'Uttarahalli')
```



In [65]:
```
'''So when I'm looking the image so it comes 1300 squared feet area
    that it comes  2_bhk house and 3_BHK House
```

```
    At same price .....
    So, like that that prices I want to remove that from my data.'''
```

Out[65]:
```
"So when I'm looking the image so it comes 1300 squared feet area\n    that it comes  2_bhk house and 3_BHK House\n    At same price .....\n    So, like that that prices I want to remove that from my data."
```

In [66]:
```
def remove_bhk_outliers(df):
    which_combined = np.array([ ])

    for location, location_df in df.groupby('location'):
        bhk_stats ={}

        for BHK, BHK_df  in location_df.groupby('BHK'):
            bhk_stats[BHK]={
                'mean' : np.mean(BHK_df['price_per_sqft']),
                'std'  : np.std (BHK_df['price_per_sqft']),
                'count': BHK_df.shape[0]
            }

        for BHK, BHK_df in location_df.groupby('BHK'):
            stats = bhk_stats.get(BHK -1)
            if stats and stats['count'] >5 :
                which_combined =np.append(which_combined,BHK_df[BHK_df['price_pe

    return df.drop (which_combined, axis= 'index')
```

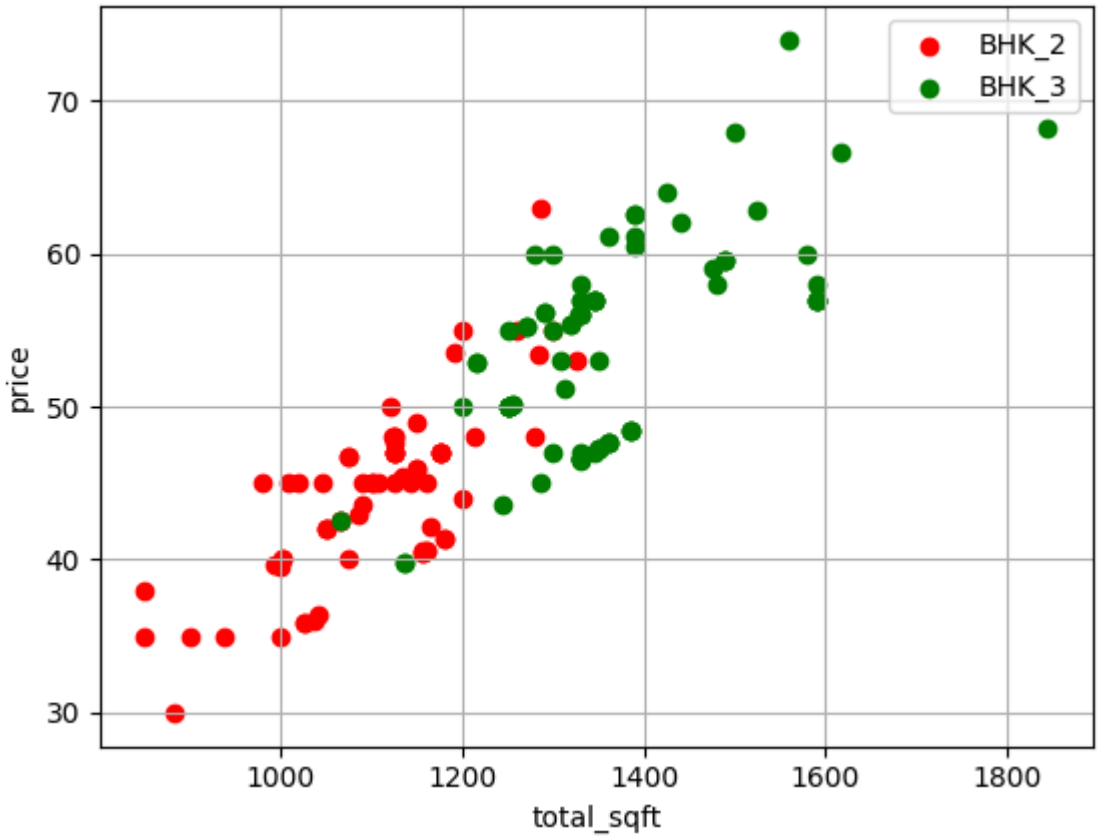
In [67]:
```
df9.shape
```

Out[67]:
```
(10016, 6)
```

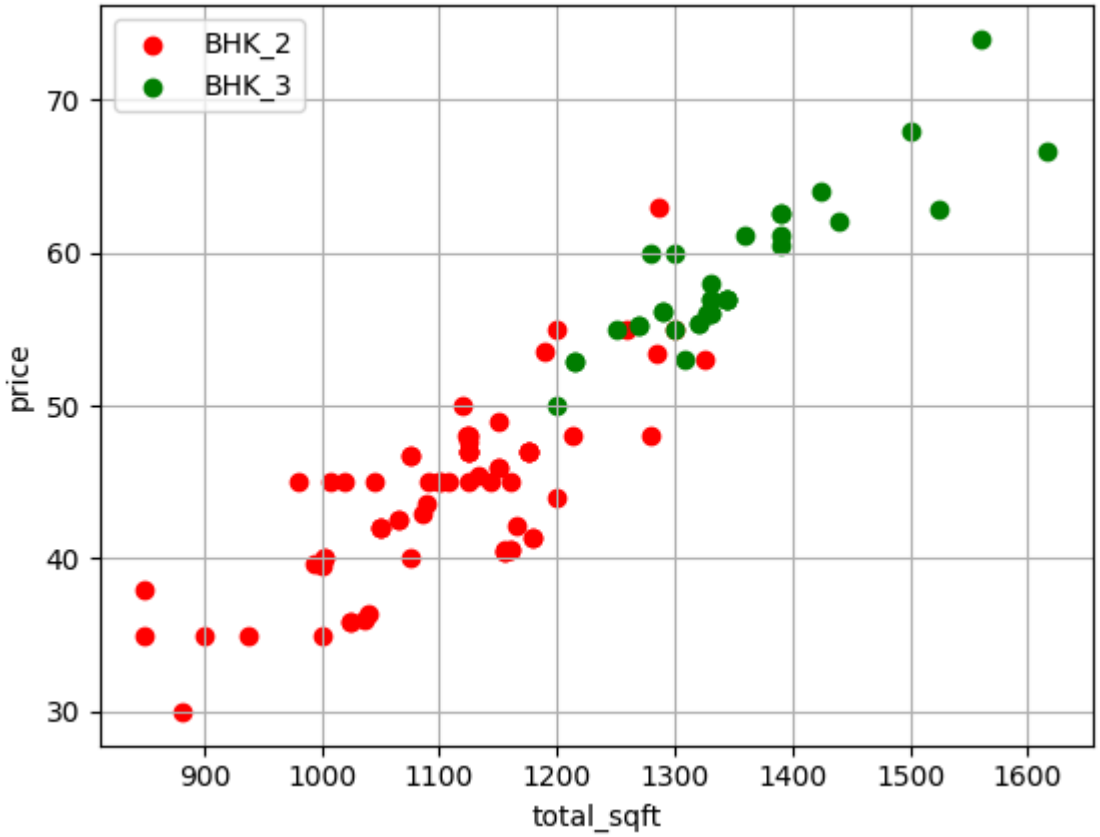
In [68]:
```
df10=remove_bhk_outliers(df9)    # There are Nearly 3000  like are removed
df10.shape
```

Out[68]:
```
(7164, 6)
```

In [69]:
```
scatter_plot(df9,"Uttarahalli")
```



In [70]:
```
scatter_plot(df10,"Uttarahalli")
```

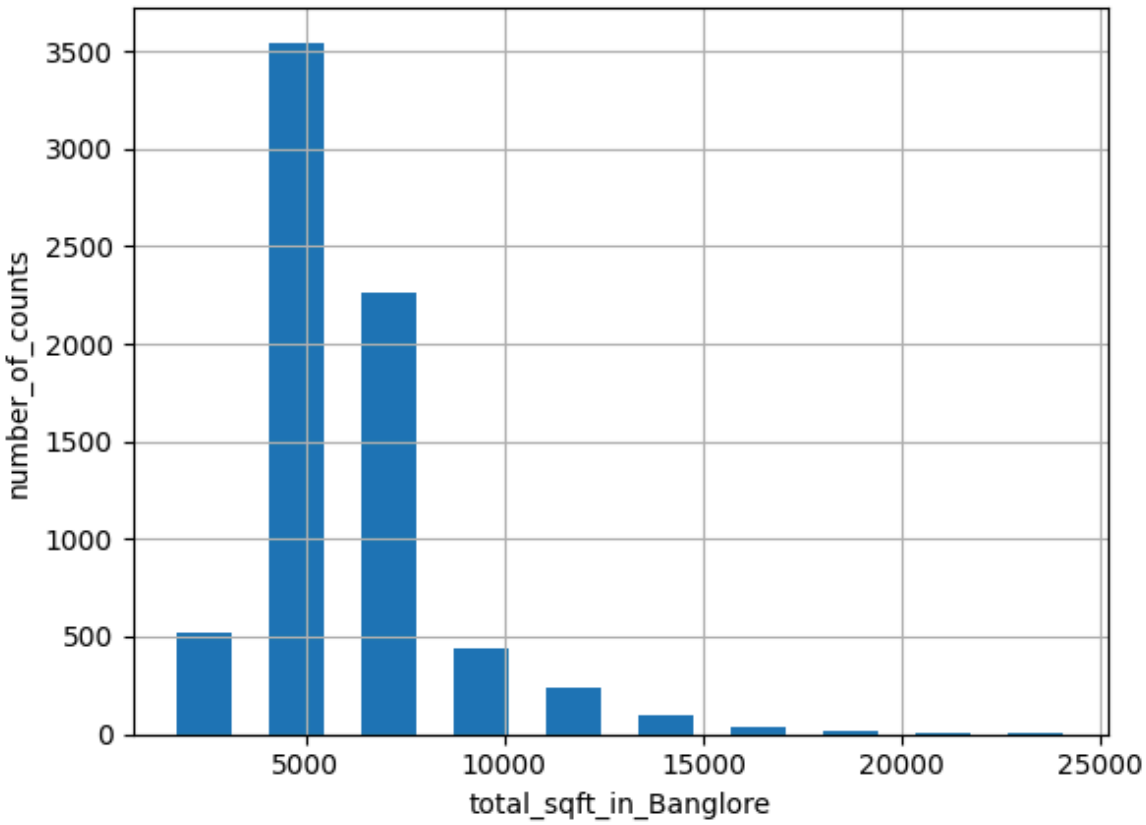


In [71]:
```
df10.shape
```

Out[71]: (7164, 6)

In [72]:
```
'''Now just look at the  how much Squarefeet are high selling  in the all cities

pp.hist(df10.price_per_sqft ,rwidth=0.6)
pp.xlabel("total_sqft_in_Banglore")
pp.ylabel("number_of_counts")
pp.grid()
pp.show()
```



In [73]:
```
df10 .head()
```

Out[73]:

	location	total_sqft	bath	price	BHK	price_per_sqft
1	Yeshwanthpur	1693.0	3.0	108.00	3	6379.208506
2	Yeshwanthpur	667.0	1.0	36.85	1	5524.737631
3	Yeshwanthpur	1950.0	4.0	130.00	4	6666.666667
5	Yeshwanthpur	665.0	1.0	36.85	1	5541.353383
9	Yeshwanthpur	671.0	1.0	36.85	1	5491.803279

In [74]:
```
df10.bath.unique()
```

Out[74]:
```
array([ 3.,  1.,  4.,  2.,  5.,  6.,  7.,  8.,  9., 12., 16., 13.])
```

In [75]:
```
#So now let's look at that bathroom.
# The number of BHK == bathrooms , if it considered
# The number of BHK == bathrooms + 2 , remove that .....
```

In [76]:
```
df10[df10.bath > df10.BHK+2 ]
```

Out[76]:

	location	total_sqft	bath	price	BHK	price_per_sqft
1341	Thanisandra	1806.0	6.0	116.0	3	6423.034330
3068	OTHER <=10	11338.0	9.0	1000.0	6	8819.897689
4916	Nagasandra	7000.0	8.0	450.0	4	6428.571429
8447	Chikkabanavar	2460.0	7.0	80.0	4	3252.032520

In [77]:
```
df10.shape
```

Out[77]: (7164, 6)

In [78]:
```
df11 = df10[df10.bath < df10.BHK+2 ]   # nearly 100 are removed
df11.shape
```

Out[78]: (7088, 6)

In [ ]:

In [79]:
```
# here now data cleaning and everthing over
# next performing using Machine Learning
# Before we removing Unnessary columns
df11.head()
```

Out[79]:

	location	total_sqft	bath	price	BHK	price_per_sqft
1	Yeshwanthpur	1693.0	3.0	108.00	3	6379.208506
2	Yeshwanthpur	667.0	1.0	36.85	1	5524.737631
3	Yeshwanthpur	1950.0	4.0	130.00	4	6666.666667
5	Yeshwanthpur	665.0	1.0	36.85	1	5541.353383
9	Yeshwanthpur	671.0	1.0	36.85	1	5491.803279

In [80]:
```
df12 = df11.drop('price_per_sqft' , axis="columns")
df12.head()
```

Out[80]:

	location	total_sqft	bath	price	BHK
1	Yeshwanthpur	1693.0	3.0	108.00	3
2	Yeshwanthpur	667.0	1.0	36.85	1
3	Yeshwanthpur	1950.0	4.0	130.00	4
5	Yeshwanthpur	665.0	1.0	36.85	1
9	Yeshwanthpur	671.0	1.0	36.85	1

In [ ]:

4. MODEL BUILDING

MACHINE LEARNING

In [ ]:

In [81]:
```
df12.shape
```

Out[81]: (7088, 5)

In [82]:
```
len(df12.location.unique())
```

Out[82]: 242

In [83]:
```
dummy  = pd.get_dummies(df12.location)
dummy.head()
```

Out[83]:

	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	5th Block Hbr Layout	5th Phase JP Nagar	6th Phase JP Nagar	7th Phase JP Nagar	8th Phase JP Nagar	9 Pha Nag
1	0	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	
5	0	0	0	0	0	0	0	0	0	
9	0	0	0	0	0	0	0	0	0	

5 rows × 242 columns

In [84]:
```
dummy.shape
```

Out[84]: (7088, 242)

In [85]:
```
df13 = pd.concat([df12,dummy], axis ="columns")
df13.head()
```

Out[85]:

	location	total_sqft	bath	price	BHK	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi
1	Yeshwanthpur	1693.0	3.0	108.00	3	0	0	0	0
2	Yeshwanthpur	667.0	1.0	36.85	1	0	0	0	0
3	Yeshwanthpur	1950.0	4.0	130.00	4	0	0	0	0
5	Yeshwanthpur	665.0	1.0	36.85	1	0	0	0	0
9	Yeshwanthpur	671.0	1.0	36.85	1	0	0	0	0

5 rows × 247 columns

In [86]:
```
df14 = df13.drop("location", axis ="columns")
df14.head()
```

Out[86]:

	total_sqft	bath	price	BHK	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	5th Block Hbr Layout	5 Pha Nag
1	1693.0	3.0	108.00	3	0	0	0	0	0	
2	667.0	1.0	36.85	1	0	0	0	0	0	
3	1950.0	4.0	130.00	4	0	0	0	0	0	
5	665.0	1.0	36.85	1	0	0	0	0	0	
9	671.0	1.0	36.85	1	0	0	0	0	0	

5 rows × 246 columns

In [87]:
```
df14.shape
```

Out[87]: (7088, 246)

In [ ]:
```

```

In [88]:
```
#         Y = mX + C
```

In [89]:
```
X = df14.drop('price', axis ="columns")  # Dependent variable
y = df14.price            # Independent Variable
```

In [90]:
```
X.head()
```

Out[90]:

	total_sqft	bath	BHK	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	5th Block Hbr Layout	5th Phase JP Nagar	6 Pha Nag
1	1693.0	3.0	3	0	0	0	0	0	0	
2	667.0	1.0	1	0	0	0	0	0	0	
3	1950.0	4.0	4	0	0	0	0	0	0	
5	665.0	1.0	1	0	0	0	0	0	0	
9	671.0	1.0	1	0	0	0	0	0	0	

5 rows × 245 columns

In [91]:
```
y.head()
```

Out[91]:
```
1    108.00
2     36.85
3    130.00
5     36.85
9     36.85
Name: price, dtype: float64
```

In [ ]:

In [ ]:

In [92]:
```
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)
```

In [93]:
```
len(X_test)
```

Out[93]:
```
1418
```

In [94]:
```
len(X_train)
```

Out[94]:
```
5670
```

In [95]:
```
len(X)
```

Out[95]:
```
7088
```

In [ ]:

In [96]:
```
from sklearn. linear_model import LinearRegression
Lr = LinearRegression()
Lr.fit(X_train,y_train)
Lr.score(X_test,y_test)
```

Out[96]:
```
0.8470245463514008
```

In [97]:
```
y_pred= Lr.predict(X_test)
y_pred
```

Out[97]:
```
array([ 99.85717392, 301.27079582, 281.15257645, ...,  63.56637764,
        35.45453453,  94.01435852])
```

In [98]:
```
'''ShuffleSplit = It is used for cross-validation, which is a technique for
                    assessing the performance of a model by dividing the dataset
                    It uses to Random Shuffle , Create Multiple Splits , Adjusta

from sklearn.model_selection import ShuffleSplit
from sklearn.model_selection import cross_val_score

cv =ShuffleSplit(n_splits=10, test_size=0.2)
cross_val_score(LinearRegression(),X,y, cv=cv)
```

Out[98]:
```
array([0.80699942, 0.86897948, 0.88064204, 0.87465336, 0.87901427,
       0.86769892, 0.86554255, 0.87295709, 0.87296915, 0.86903503])
```

In [ ]:

In [99]:
```
from sklearn.tree import DecisionTreeRegressor
DTR = DecisionTreeRegressor()
DTR.fit(X_train,y_train)
DTR.score(X_test,y_test)
```

Out[99]: 0.6654185515069233

In [100…
```
X.head()
```

Out[100]:

	total_sqft	bath	BHK	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	5th Block Hbr Layout	5th Phase JP Nagar	6 Pha Nag
1	1693.0	3.0	3	0	0	0	0	0	0	
2	667.0	1.0	1	0	0	0	0	0	0	
3	1950.0	4.0	4	0	0	0	0	0	0	
5	665.0	1.0	1	0	0	0	0	0	0	
9	671.0	1.0	1	0	0	0	0	0	0	

5 rows × 245 columns

In [101…
```
X.columns
```

Out[101]:
```
Index(['total_sqft', 'bath', 'BHK', '1st Block Jayanagar',
       '1st Phase JP Nagar', '2nd Phase Judicial Layout',
       '2nd Stage Nagarbhavi', '5th Block Hbr Layout', '5th Phase JP Nagar',
       '6th Phase JP Nagar',
       ...
       'Vijayanagar', 'Vishveshwarya Layout', 'Vishwapriya Layout',
       'Vittasandra', 'Whitefield', 'Yelachenahalli', 'Yelahanka',
       'Yelahanka New Town', 'Yelenahalli', 'Yeshwanthpur'],
      dtype='object', length=245)
```

In [ ]:
```
```

predict the Price on location

In [105…
```
def predicted_price(location,total_sqft,bath,BHK):
    location_index = np.where(X.columns ==location)[0][0]

    x  = np.zeros(len(X.columns))
    x[0]= total_sqft
    x[1]= bath
    x[2]= BHK

    if location_index >=0:
        x[location_index] =1


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

In [106…
```
predicted_price("Uttarahalli", 1800 , 2,2)
```

```
c:\users\sriha\appdata\local\programs\python\python37\lib\site-packages\sklearn\base.py:451: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names
  "X does not have valid feature names, but"
```

Out[106]: 99.46133804321289

In [108…
```
predicted_price("Uttarahalli",1600,2,2)
```

```
c:\users\sriha\appdata\local\programs\python\python37\lib\site-packages\sklearn\base.py:451: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names
  "X does not have valid feature names, but"
```

Out[108]: 83.48165893554688

In [109…
```
predicted_price("Uttarahalli",1600,3,2)
```

```
c:\users\sriha\appdata\local\programs\python\python37\lib\site-packages\sklearn\base.py:451: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names
  "X does not have valid feature names, but"
```

Out[109]: 87.82004928588867

In [110…
```
predicted_price("Uttarahalli",1600,3,3)
```

```
c:\users\sriha\appdata\local\programs\python\python37\lib\site-packages\sklearn\base.py:451: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names
  "X does not have valid feature names, but"
```

Out[110]: 84.81619262695312

In [111…
```
predicted_price("Uttarahalli",1600,2,3)
```

```
c:\users\sriha\appdata\local\programs\python\python37\lib\site-packages\sklearn\base.py:451: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names
  "X does not have valid feature names, but"
```

Out[111]: 80.47780227661133

In [ ]:
```
```