In [1]:
```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
%matplotlib inline
import matplotlib
matplotlib.rcParams["figure.figsize"] =(20,10)
```

In [3]:
```
df1 = pd.read_csv("C:\\Users\\AMIT\\Downloads\\Bengaluru_House_Data.csv")
df1.head()
```

Out[3]:

	area_type	availability	location	size	society	total_sqft	bath	balcony	price
0	Super built-up Area	19-Dec	Electronic City Phase II	2 BHK	Coomee	1056	2.0	1.0	39.07
1	Plot Area	Ready To Move	Chikka Tirupathi	4 Bedroom	Theanmp	2600	5.0	3.0	120.00
2	Built-up Area	Ready To Move	Uttarahalli	3 BHK	NaN	1440	2.0	3.0	62.00
3	Super built-up Area	Ready To Move	Lingadheeranahalli	3 BHK	Soiewre	1521	3.0	1.0	95.00
4	Super built-up Area	Ready To Move	Kothanur	2 BHK	NaN	1200	2.0	1.0	51.00

In [7]:
```
df1.shape # shows number of rows and columns in a dataset
```

Out[7]:
```
(13320, 9)
```

In [8]:
```
#To get the count of area_type by grouping them and take the count
df1.groupby('area_type')['area_type'].agg('count')
```

Out[8]:
```
area_type
Built-up  Area          2418
Carpet  Area              87
Plot  Area              2025
Super built-up  Area    8790
Name: area_type, dtype: int64
```

In [9]:
```
#droing the columns as which are not necessary in predicting the price
df2 = df1.drop(['area_type','availability','society','balcony'] , axis = 'columns')
df2.shape
```

Out[9]:
```
(13320, 5)
```

In [10]:
```
df2.head()
```

Out[10]:

	location	size	total_sqft	bath	price
0	Electronic City Phase II	2 BHK	1056	2.0	39.07

	location	size	total_sqft	bath	price
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 [11]:
```
#Data Cleaning process starts with handling NA values.
df2.isnull().sum()
```

Out[11]:
```
location       1
size          16
total_sqft     0
bath          73
price          0
dtype: int64
```

In [12]:
```
#Now if we don't want to drop the NA values than what we can do is we can take the m
#then fill the NA values with the median values.
#But the data set is pretty large along with that the na values are less so i can sa
df3 = df2.dropna()
```

In [13]:
```
df3.isnull().sum()
```

Out[13]:
```
location      0
size          0
total_sqft    0
bath          0
price         0
dtype: int64
```

In [14]:
```
df3.shape
```

Out[14]:
```
(13246, 5)
```

In [15]:
```
df3['size'].unique()
```

Out[15]:
```
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 [16]:
```
#This code is performing a transformation on a dataframe df3 in python .
#The apply function applies a lambda function to each element in the 'size' column .
#A lambda function is an anonymous function defined using the keyword lambda.
#x.split(' '): This splits the string x by spaces. For example, if x is '2 BHK', x.s
#list ['2', 'BHK'].
#x.split(' ')[0]: This takes the first element of the list resulting from the split
#this would be '2'.
#int(...): This converts the first element (which is a string) to an integer.
df3['bhk'] = df3['size'].apply(lambda x: int((x.split(' ')[0])))
```

```
C:\Users\AMIT\AppData\Local\Temp/ipykernel_22016/426311518.py:1: SettingWithCopyWarni
ng:
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/stable/us
er_guide/indexing.html#returning-a-view-versus-a-copy
  df3['bhk'] = df3['size'].apply(lambda x: int((x.split(' ')[0])))
```

In [17]:
```
df3.head()
```

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]:
```
df3['bhk'].unique()
```

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]:
```
df3[df3.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]:
```
#There is something error in it like we cannot have 2400 total_sqrt for 43 bedroom
df3.total_sqft.unique()
```

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

In [21]:
```
#To check the values we are having in total_sqft is having float value or not
def is_float(x):
    try:
        float(x)
    except:
        return False
    return True
```

In [25]:
```
df3[~df3['total_sqft'].apply(is_float)].head(10)
```

Out[25]:

	location	size	total_sqft	bath	price	bhk
30	Yelahanka	4 BHK	2100 - 2850	4.0	186.000	4
122	Hebbal	4 BHK	3067 - 8156	4.0	477.000	4

	location	size	total_sqft	bath	price	bhk
137	8th Phase JP Nagar	2 BHK	1042 - 1105	2.0	54.005	2
165	Sarjapur	2 BHK	1145 - 1340	2.0	43.490	2
188	KR Puram	2 BHK	1015 - 1540	2.0	56.800	2
410	Kengeri	1 BHK	34.46Sq. Meter	1.0	18.500	1
549	Hennur Road	2 BHK	1195 - 1440	2.0	63.770	2
648	Arekere	9 Bedroom	4125Perch	9.0	265.000	9
661	Yelahanka	2 BHK	1120 - 1145	2.0	48.130	2
672	Bettahalsoor	4 Bedroom	3090 - 5002	4.0	445.000	4

In [35]:
```
def convert_sqft_to_num(x):
    tokens = x.split('-')
    if len(tokens) == 2:
        return (float(tokens[0]) + float(tokens[1]))/2
    try:
        return float(x)
    except:
        return None
```

In [27]:
```
convert_sqft_to_num('2144')
```

Out[27]:
```
2144.0
```

In [36]:
```
convert_sqft_to_num('2100 - 2850')
```

Out[36]:
```
2475.0
```

In [37]:
```
convert_sqft_to_num('34.46Sq. Meter')
```

In [38]:
```
df4 = df3.copy()
df4['total_sqft'] = df4['total_sqft'].apply(convert_sqft_to_num)
```

In [40]:
```
df4.head(10)
```

Out[40]:

	location	size	total_sqft	bath	price	bhk
0	Electronic City Phase II	2 BHK	1056.0	2.0	39.07	2
1	Chikka Tirupathi	4 Bedroom	2600.0	5.0	120.00	4
2	Uttarahalli	3 BHK	1440.0	2.0	62.00	3
3	Lingadheeranahalli	3 BHK	1521.0	3.0	95.00	3
4	Kothanur	2 BHK	1200.0	2.0	51.00	2
5	Whitefield	2 BHK	1170.0	2.0	38.00	2
6	Old Airport Road	4 BHK	2732.0	4.0	204.00	4
7	Rajaji Nagar	4 BHK	3300.0	4.0	600.00	4

	location	size	total_sqft	bath	price	bhk
8	Marathahalli	3 BHK	1310.0	3.0	63.25	3
9	Gandhi Bazar	6 Bedroom	1020.0	6.0	370.00	6

In [42]:
```
df4.loc[30]
```

Out[42]:
```
location      Yelahanka
size              4 BHK
total_sqft       2475.0
bath                4.0
price             186.0
bhk                   4
Name: 30, dtype: object
```

In [43]:
```
df4.head(3)
```

Out[43]:

	location	size	total_sqft	bath	price	bhk
0	Electronic City Phase II	2 BHK	1056.0	2.0	39.07	2
1	Chikka Tirupathi	4 Bedroom	2600.0	5.0	120.00	4
2	Uttarahalli	3 BHK	1440.0	2.0	62.00	3

In [ ]:
```
#-----------------------------------FEATURE ENGINEERING-------------------------
```

In [44]:
```
df5 = df4.copy()
df5['price_per_sqft'] = df5['price']*100000/df5['total_sqft']
df5.head()
```

Out[44]:

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

In [45]:
```
df5.location.unique()
```

Out[45]:
```
array(['Electronic City Phase II', 'Chikka Tirupathi', 'Uttarahalli', ...,
       '12th cross srinivas nagar banshankari 3rd stage',
       'Havanur extension', 'Abshot Layout'], dtype=object)
```

In [46]:
```
len(df5.location.unique())
```

Out[46]:
```
1304
```

In [54]:
```
df5.location = df5.location.apply(lambda x : x.strip())
location_stats = df5.groupby('location')['location'].agg('count').sort_values(ascend
location_stats.head(18)
```

Out[54]:
```
location
Whitefield                  535
Sarjapur  Road              392
Electronic City             304
Kanakpura Road              266
Thanisandra                 236
Yelahanka                   210
Uttarahalli                 186
Hebbal                      176
Marathahalli                175
Raja Rajeshwari Nagar       171
Bannerghatta Road           152
Hennur Road                 150
7th Phase JP Nagar          149
Haralur Road                141
Electronic City Phase II    131
Rajaji Nagar                106
Chandapura                   98
Bellandur                    96
Name: location, dtype: int64
```

In [53]:
```
len(location_stats[location_stats <= 10])
```

Out[53]:
```
1052
```

In [57]:
```
location_stats_less_than_10 = location_stats[location_stats <= 10]
location_stats_less_than_10.head(50)
```

Out[57]:
```
location
Basapura                 10
1st Block Koramangala    10
Gunjur Palya             10
Kalkere                  10
Sector 1 HSR Layout      10
Dairy Circle             10
Naganathapura            10
Sadashiva Nagar          10
Nagadevanahalli          10
BTM 1st Stage            10
Nagappa Reddy Layout     10
Dodsworth Layout         10
Ganga Nagar              10
2nd Phase JP Nagar        9
Volagerekallahalli        9
Yemlur                    9
4th Block Koramangala     9
Lingarajapuram            9
Medahalli                 9
Kaverappa Layout          9
Gollahalli                9
Richmond Town             9
Vishwanatha Nagenahalli   9
Chennammana Kere          9
KUDLU MAIN ROAD           9
Banagiri Nagar            9
Peenya                    9
Ejipura                   9
Vignana Nagar             9
Mathikere                 9
Chandra Layout            9
Jakkur Plantation         9
```

```
Kamakshipalya               9
B Narayanapura              9
Huskur                      8
Kanaka Nagar                8
Thirumenahalli              8
Vajarahalli                 8
Vasanth nagar               8
Kattigenahalli              8
Ittamadu                    8
1st Block HRBR Layout       8
Sidedahalli                 8
Sarjapur Road,              8
Cambridge Layout            8
Outer Ring Road East        8
Nelamangala                 8
Jalahalli West              8
MS Pallya                   8
Seetharampalya              8
Name: location, dtype: int64
```

In [58]:
```
len(df5.location.unique())
```

Out[58]:
```
1293
```

In [60]:
```
df5.location = df5.location.apply(lambda x : 'other' if x in location_stats_less_tha
```

In [61]:
```
len(df5.location.unique())
```

Out[61]:
```
242
```

In [63]:
```
df5.head(10)
```

Out[63]:

	location	size	total_sqft	bath	price	bhk	price_per_sqft
0	Electronic City Phase II	2 BHK	1056.0	2.0	39.07	2	3699.810606
1	Chikka Tirupathi	4 Bedroom	2600.0	5.0	120.00	4	4615.384615
2	Uttarahalli	3 BHK	1440.0	2.0	62.00	3	4305.555556
3	Lingadheeranahalli	3 BHK	1521.0	3.0	95.00	3	6245.890861
4	Kothanur	2 BHK	1200.0	2.0	51.00	2	4250.000000
5	Whitefield	2 BHK	1170.0	2.0	38.00	2	3247.863248
6	Old Airport Road	4 BHK	2732.0	4.0	204.00	4	7467.057101
7	Rajaji Nagar	4 BHK	3300.0	4.0	600.00	4	18181.818182
8	Marathahalli	3 BHK	1310.0	3.0	63.25	3	4828.244275
9	other	6 Bedroom	1020.0	6.0	370.00	6	36274.509804

In [ ]:
```
#----------------------------------------Outlier Removal---------------------------
# What is outlier?
# Outlier is a datapoint that significantly deviates from the other observations in
# Outliers can be caused by variability in the data or indicates experimental errors
# They can have significant impact on the results of data analysis and machine learn
# misleading inaccurate conclusions.
```

```
#-------------------------Characteristics of outliers----------------------------
# Extreme values - They often have extreme values that lie far outside the range of
# Isolation -They are isolated from the main body of the data , either in the high o
```

In [64]:
```
df5[df5.total_sqft/df5.bhk < 300].head()
```

Out[64]:

	location	size	total_sqft	bath	price	bhk	price_per_sqft
9	other	6 Bedroom	1020.0	6.0	370.0	6	36274.509804
45	HSR Layout	8 Bedroom	600.0	9.0	200.0	8	33333.333333
58	Murugeshpalya	6 Bedroom	1407.0	4.0	150.0	6	10660.980810
68	Devarachikkanahalli	8 Bedroom	1350.0	7.0	85.0	8	6296.296296
70	other	3 Bedroom	500.0	3.0	100.0	3	20000.000000

In [65]:
```
df5.shape
```

Out[65]:
```
(13246, 7)
```

In [66]:
```
df6 = df5[~(df5.total_sqft/df5.bhk < 300)]
```

In [67]:
```
df6.shape
```

Out[67]:
```
(12502, 7)
```

In [68]:
```
df6.head()
```

Out[68]:

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

In [69]:
```
df6.price_per_sqft.describe()
```

Out[69]:
```
count     12456.000000
mean       6308.502826
std        4168.127339
min         267.829813
25%        4210.526316
50%        5294.117647
75%        6916.666667
max      176470.588235
Name: price_per_sqft, dtype: float64
```

In [70]:
```
def remove_pps_outliers(df):
    df_out = pd.DataFrame()#An empty dataframe is created to store the filtered resu
    #For each location the mean and standard deviation is calculated.
```

```
    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 <
        df_out = pd.concat([df_out, reduced_df], ignore_index = True)
    return df_out
```

In [71]:

```
df7 = remove_pps_outliers(df6)
df7.shape
```

Out[71]:

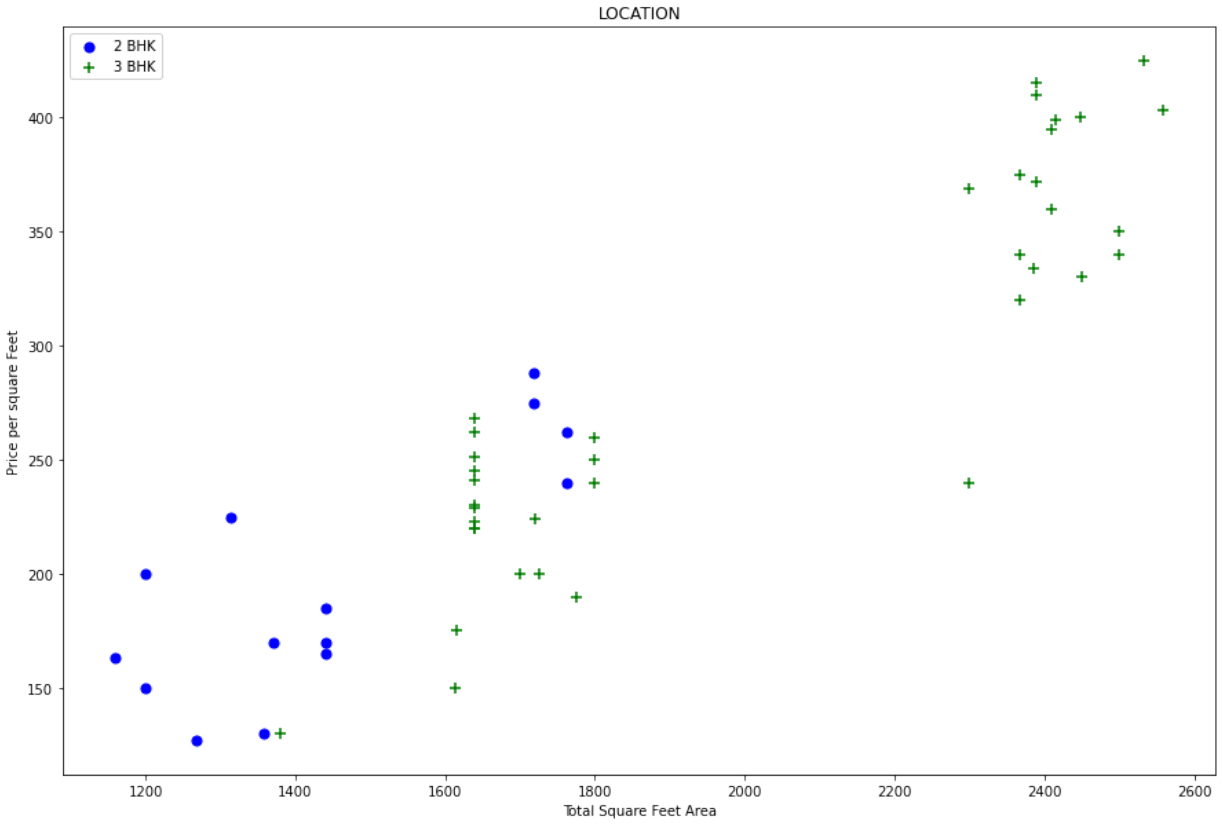
```
(10241, 7)
```

In [74]:

```
def plot_scatter_chart(df,location):
    bhk2 = df[(df.location == location) & (df.bhk == 2)]
    bhk3 = df[(df.location == location) & (df.bhk == 3)]
    matplotlib.rcParams['figure.figsize'] = (15,10)
    plt.scatter(bhk2.total_sqft , bhk2.price , color = 'blue', label = '2 BHK' , s =
    plt.scatter(bhk3.total_sqft , bhk3.price , marker ='+', color = 'green', label =
    plt.xlabel("Total Square Feet Area")
    plt.ylabel("Price per square Feet")
    plt.title("LOCATION")
    plt.legend()
```

In [75]:

```
plot_scatter_chart(df7,"Rajaji Nagar")
```



In [76]:

```
#For cleaning of outliers
# We should also remove properties where for same locations , the price of 3 bedroom
def remove_bhk_outliers(df):
    exclude_indices = np.array([])
    for location , location_df in df.groupby('location'):
        bhk_stats = {}
        for bhk , bhk_df in location_df.groupby('bhk'):
```

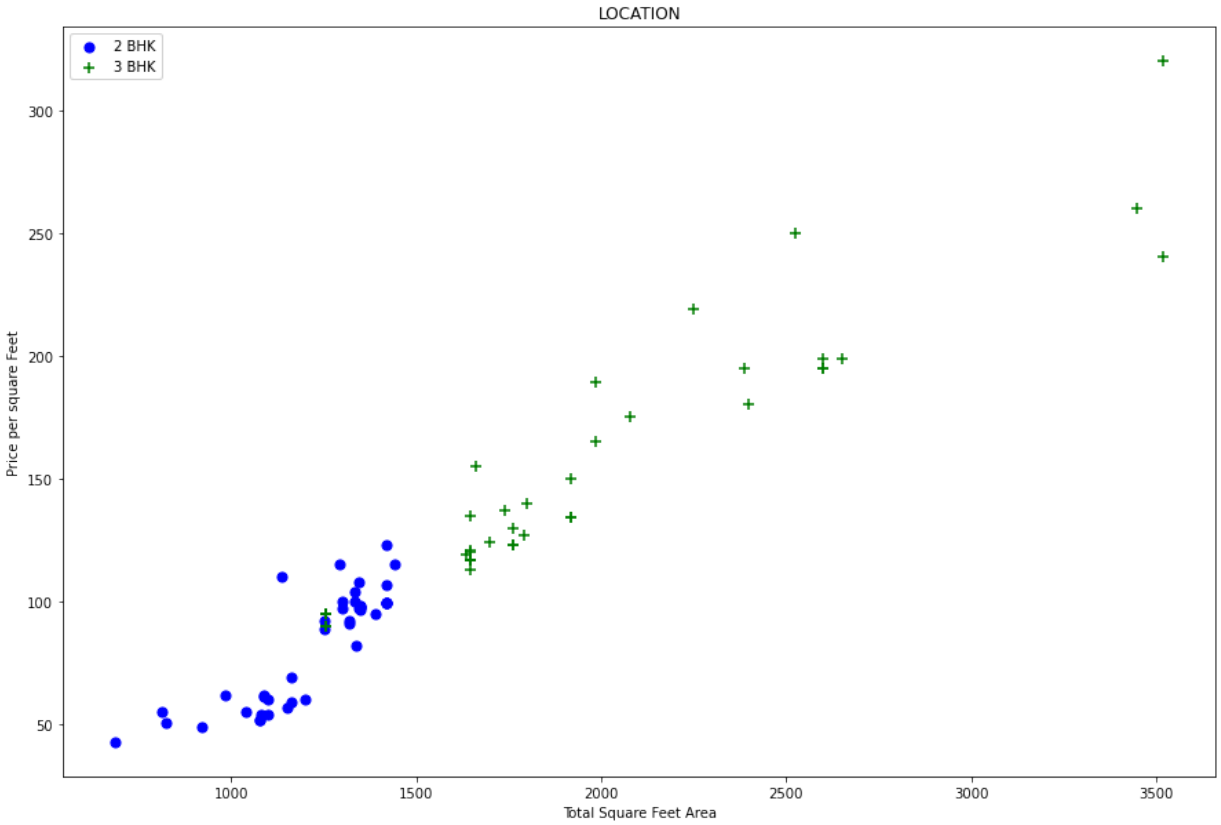
```
            bhk_stats[bhk]={
                'mean':np.mean(bhk_df.price_per_sqft),
                'std':np.std(bhk_df.price_per_sqft),
                'count':bhk_df.shape[0]
            }
        for bhk , bhk_df in location_df.groupby('bhk'):
            stats = bhk_stats.get(bhk-1)
            if stats and stats['count']>5:
                exclude_indices = np.append(exclude_indices , bhk_df[bhk_df.price_pe

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

In [77]:
```
df8 = remove_bhk_outliers(df7)
df8.shape
```

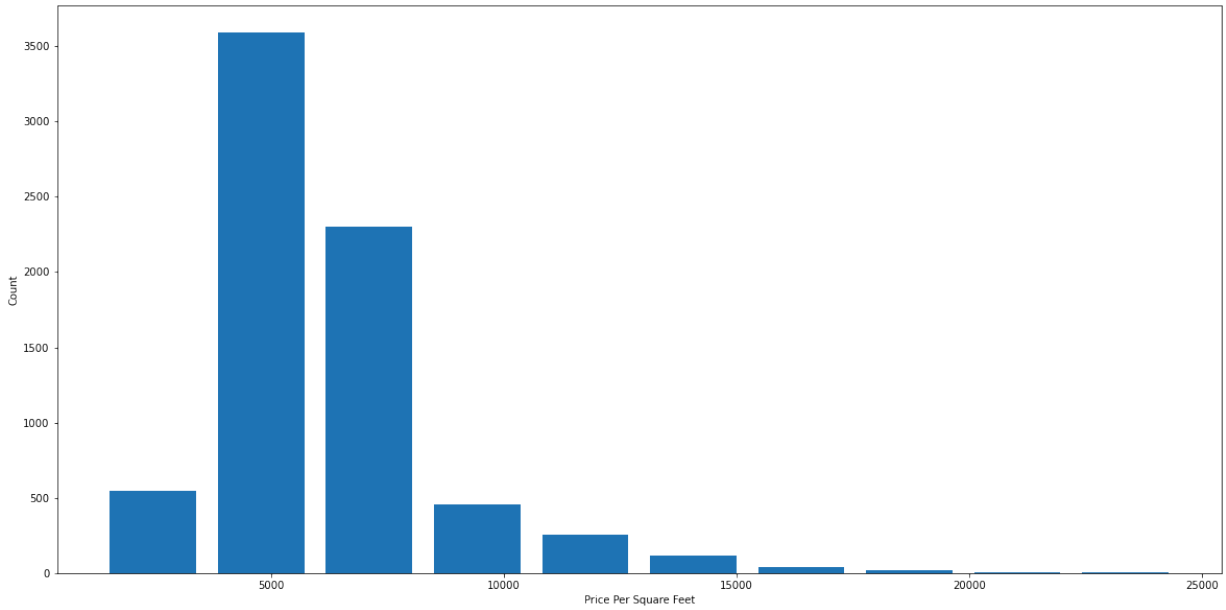
Out[77]:
```
(7329, 7)
```

In [78]:
```
plot_scatter_chart(df8 , "Hebbal")
```



In [79]:
```
import matplotlib
matplotlib.rcParams["figure.figsize"]=(20,10)
plt.hist(df8.price_per_sqft,rwidth=0.8)
plt.xlabel("Price Per Square Feet")
plt.ylabel("Count")
```

Out[79]:
```
Text(0, 0.5, 'Count')
```



In [80]:
```
df8.bath.unique()
```

Out[80]:
```
array([ 4.,  3.,  2.,  5.,  8.,  1.,  6.,  7.,  9., 12., 16., 13.])
```

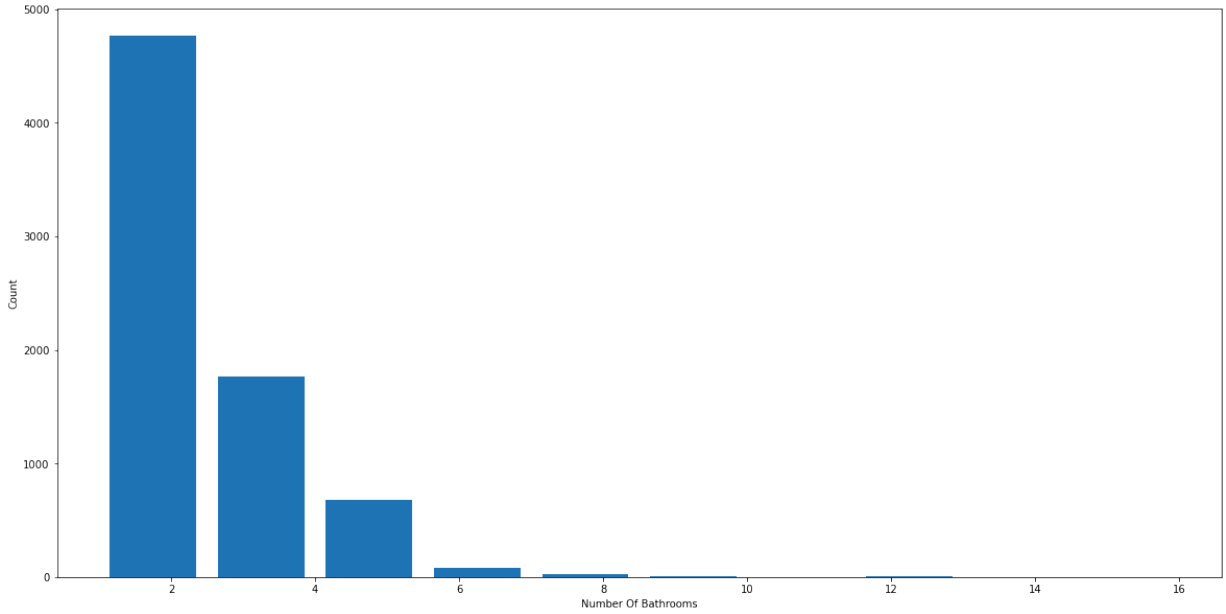
In [81]:
```
df8[df8.bath>10]
```

Out[81]:

	location	size	total_sqft	bath	price	bhk	price_per_sqft
5277	Neeladri Nagar	10 BHK	4000.0	12.0	160.0	10	4000.000000
8486	other	10 BHK	12000.0	12.0	525.0	10	4375.000000
8575	other	16 BHK	10000.0	16.0	550.0	16	5500.000000
9308	other	11 BHK	6000.0	12.0	150.0	11	2500.000000
9639	other	13 BHK	5425.0	13.0	275.0	13	5069.124424

In [82]:
```
plt.hist(df8.bath , rwidth=0.8)
plt.xlabel("Number Of Bathrooms")
plt.ylabel("Count")
```

Out[82]:
```
Text(0, 0.5, 'Count')
```



In [83]:
```
df8[df8.bath > df8.bhk+2]
```

Out[83]:

	location	size	total_sqft	bath	price	bhk	price_per_sqft
1626	Chikkabanavar	4 Bedroom	2460.0	7.0	80.0	4	3252.032520
5238	Nagasandra	4 Bedroom	7000.0	8.0	450.0	4	6428.571429
6711	Thanisandra	3 BHK	1806.0	6.0	116.0	3	6423.034330
8411	other	6 BHK	11338.0	9.0	1000.0	6	8819.897689

In [84]:
```
df9 = df8[df8.bath < df8.bhk+2]
df9.shape
```

Out[84]:
```
(7251, 7)
```

In [86]:
```
df10 = df9.drop(['size','price_per_sqft'] , axis="columns")
df10.head()
```

Out[86]:

	location	total_sqft	bath	price	bhk
0	1st Block Jayanagar	2850.0	4.0	428.0	4
1	1st Block Jayanagar	1630.0	3.0	194.0	3
2	1st Block Jayanagar	1875.0	2.0	235.0	3
3	1st Block Jayanagar	1200.0	2.0	130.0	3
4	1st Block Jayanagar	1235.0	2.0	148.0	2

In [89]:
```
#----------------------------------------Model Building---------------------------
# In location we are having categorical data so in machine learning model cannot int
# this into a numeric column and one of the way to convert categorical data into num
# dummies.
dummies=pd.get_dummies(df10.location)
dummies.head(3)
```

Out[89]:

	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	9th Phase JP Nagar	...	Vishve
0	1	0	0	0	0	0	0	0	0	0	...	
1	1	0	0	0	0	0	0	0	0	0	...	
2	1	0	0	0	0	0	0	0	0	0	...	

3 rows × 242 columns

In [90]:

```
df11 = pd.concat([df10 , dummies.drop('other' , axis = 'columns')] , axis = 'columns
df11.head(3)
```

Out[90]:

	location	total_sqft	bath	price	bhk	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	5th Block Hbr Layout	...	Vijay
0	1st Block Jayanagar	2850.0	4.0	428.0	4	1	0	0	0	0	...	
1	1st Block Jayanagar	1630.0	3.0	194.0	3	1	0	0	0	0	...	
2	1st Block Jayanagar	1875.0	2.0	235.0	3	1	0	0	0	0	...	

3 rows × 246 columns

In [91]:

```
df12 = df11.drop('location', axis = 'columns')
df12.head(2)
```

Out[91]:

	total_sqft	bath	price	bhk	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	5th Block Hbr Layout	5th Phase JP Nagar	...	Vijayana
0	2850.0	4.0	428.0	4	1	0	0	0	0	0	...	
1	1630.0	3.0	194.0	3	1	0	0	0	0	0	...	

2 rows × 245 columns

In [92]:

```
df12.shape
```

Out[92]:

```
(7251, 245)
```

In [93]:

```
x = df12.drop('price' , axis='columns')
x.head(2)
```

Out[93]:

	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	...	Vijayar
0	2850.0	4.0	4	1	0	0	0	0	0	0	...	
1	1630.0	3.0	3	1	0	0	0	0	0	0	...	

2 rows × 244 columns

In [94]:
```
y=df12.price
y.head()
```

Out[94]:
```
0    428.0
1    194.0
2    235.0
3    130.0
4    148.0
Name: price, dtype: float64
```

In [95]:
```
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
```

In [97]:
```
from sklearn.linear_model import LinearRegression
lr_clf = LinearRegression()
lr_clf.fit(x_train,y_train)
lr_clf.score(x_test,y_test)#tell how good our model is.
```

Out[97]:
```
0.8452277697874312
```

In [99]:
```
# Using k fold cross validation to measure accuracy of our Linear Regression model
from sklearn.model_selection import ShuffleSplit
from sklearn.model_selection import cross_val_score

cv = ShuffleSplit(n_splits=5 , test_size=0.2 , random_state=0)
cross_val_score(LinearRegression() , x,y, cv=cv)
```

Out[99]:
```
array([0.82430186, 0.77166234, 0.85089567, 0.80837764, 0.83653286])
```

In [104…
```
# We can see that in 5 iterations we get a score above 80% all the time .This is pre
# algorithms for regression to see if we can get even a better score , ll use GridSe
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import Lasso
from sklearn.tree import DecisionTreeRegressor

def find_best_model_using_gridsearchcv(x,y):
    algos={
        'linear regression':{
            'model':LinearRegression(),
            'params':{
                'normalize':[True,False]
            }
        },
        'lasso':{
            'model':Lasso(),
```

```
            'params':{
                'alpha':[1,2],
                'selection':['random','cyclic']
            }
        },
        'decision_tree':{
            'model':DecisionTreeRegressor(),
            'params':{
                'criterion':['mse','friedman_mse'],
                'splitter':['best','random']
            }
        }
    }
    scores = []
    cv = ShuffleSplit(n_splits=5, test_size=0.2, random_state=0)
    for algo_name, config in algos.items():
        gs =  GridSearchCV(config['model'], config['params'], cv=cv, return_train_sc
        gs.fit(x,y)
        scores.append({
            'model': algo_name,
            'best_score': gs.best_score_,
            'best_params': gs.best_params_
        })

    return pd.DataFrame(scores,columns=['model','best_score','best_params'])

find_best_model_using_gridsearchcv(x,y)
```

Out[104...

	model	best_score	best_params
0	linear regression	0.818354	{'normalize': True}
1	lasso	0.687475	{'alpha': 2, 'selection': 'random'}
2	decision_tree	0.721234	{'criterion': 'friedman_mse', 'splitter': 'best'}

In [109...

```
def predict_price(location,sqft,bath,bhk):
    loc_index = np.where(x.columns==location)[0][0]

    x1 = np.zeros(len(x.columns))
    x1[0] = sqft
    x1[1] = bath
    x1[2] = bhk
    if loc_index >= 0:
        x1[loc_index] = 1

    return lr_clf.predict([x1])[0]
```

In [110...

```
predict_price('1st Phase JP Nagar',1000, 2, 2)
```

Out[110...

```
83.49904677179237
```

In [111...

```
predict_price('1st Phase JP Nagar',1000, 3, 3)
```

Out[111...

```
86.80519395205847
```

In [112...

```
predict_price('Indira Nagar',1000, 2, 2)
```

Out[112…
```
181.2781548400685
```

In [113…
```
predict_price('Indira Nagar',1000, 3, 3)
```

Out[113…
```
184.58430202033463
```

In [ ]:
```
import pickle
with open('banglore_home_prices_model.pickle','wb') as f:
    pickle.dump(lr_clf,f)
```

In [119…
```
import json
columns = {
    'data_columns' : [col.lower() for col in x.columns]
}
with open("columns.json","w") as f:
    f.write(json.dumps(columns))
```

In [120…
```
import os

# Print the current working directory
print(os.getcwd())
```

```
C:\Users\AMIT
```

In [ ]: