House Price Pridiction

-> build a model using sklearn and linear regression using banglore home prices dataset from kaggle.com. -> model building we will cover almost all data science concepts such as data load and cleaning, outlier detection and removal, feature engineering, dimensionality reduction, gridsearchcv for hyperparameter tunning, k fold cross validation etc. Technology and tools wise this project covers,

1) Python 2) Numpy and Pandas for data cleaning 3) Matplotlib for data visualization 4) Sklearn for model building 5) Jupyter notebook

Importing libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

Reading file---> Bangaluru House Data

In [2]:
```
data=pd.read_csv("Bengaluru_House_Data.csv")
```

In [3]:
```
data.head()
```

	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 [4]:
```
data.tail()
```

	area_type	availability	location	size	society	total_sqft	bath	balcony	price
13315	Built-up Area	Ready To Move	Whitefield	5 Bedroom	ArsiaEx	3453	4.0	0.0	231.0
13316	Super built-up Area	Ready To Move	Richards Town	4 BHK	NaN	3600	5.0	NaN	400.0
13317	Built-up Area	Ready To Move	Raja Rajeshwari Nagar	2 BHK	Mahla T	1141	2.0	1.0	60.0
13318	Super built-up Area	18-Jun	Padmanabhanagar	4 BHK	SollyCl	4689	4.0	1.0	488.0
13319	Super built-up Area	Ready To Move	Doddathoguru	1 BHK	NaN	550	1.0	1.0	17.0

In [5]:
```
data.shape
```

```
(13320, 9)
```

In [6]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13320 entries, 0 to 13319
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   area_type     13320 non-null  object 
 1   availability  13320 non-null  object 
 2   location      13319 non-null  object 
 3   size          13304 non-null  object 
 4   society       7818 non-null   object 
 5   total_sqft    13320 non-null  object 
 6   bath          13247 non-null  float64
 7   balcony       12711 non-null  float64
 8   price         13320 non-null  float64
dtypes: float64(3), object(6)
memory usage: 936.7+ KB
```

Data cleaning

In [7]:
```
data.groupby('area_type')['area_type'].agg('count')
```

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

In [8]:
```
data['size'].unique()
```

```
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', nan, '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 [9]:
```
data['total_sqft'].unique()
```

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

In [10]:
```
data['area_type'].unique()
```

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

In [11]:
```
data.columns
```

```
Index(['area_type', 'availability', 'location', 'size', 'society',
       'total_sqft', 'bath', 'balcony', 'price'],
      dtype='object')
```

In [12]:
```
df1=data.drop(['area_type', 'availability', 'balcony', 'society'],axis='columns')
```

In [13]:
```
df1.head()
```

	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]:
```
df1.isnull().sum()
```

```
location       1
size          16
total_sqft     0
bath          73
price          0
dtype: int64
```

In [15]:
```
df1.isnull().sum().sum()
```

```
90
```

In [16]:
```
df2=df1.dropna()
```

In [17]:
```
df2.isnull().sum()
```

```
location      0
size          0
total_sqft    0
bath          0
price         0
dtype: int64
```

In [18]:
```
df2['bhk']=df2['size'].apply(lambda x:int(x.split(' ')[0]))
```

In [19]:
```
df2.head()
```

	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 [20]:
```
df2['bhk'].unique()
```

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

In [21]:
```
df2[df2.bhk>20]
```

	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 [22]:
```
df2['total_sqft'].unique()
```

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

In [23]:
```
def is_float(x):
    try:
        float(x)
    except:
        return False
    return True
```

In [24]:
```
df2[df2['total_sqft'].apply(is_float)]
```

	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
...	...	...	...	...	...	...
13315	Whitefield	5 Bedroom	3453	4.0	231.00	5
13316	Richards Town	4 BHK	3600	5.0	400.00	4
13317	Raja Rajeshwari Nagar	2 BHK	1141	2.0	60.00	2
13318	Padmanabhanagar	4 BHK	4689	4.0	488.00	4
13319	Doddathoguru	1 BHK	550	1.0	17.00	1

13056 rows × 6 columns

In [25]:
```
df2[~df2['total_sqft'].apply(is_float)].head(10)
```

	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
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 [26]:
```
def change_format(x):
    tokens=x.split('-')
    if(tokens==2):
        return (float(token[0])+float(token[1]))/2
    try:
        return float(x)
    except:
        return None
```

In [27]:
```
df3=df2.copy()
```

In [28]:
```
df3['total_sqft']=df3['total_sqft'].apply(change_format)
```

In [29]:
```
df3['total_sqft'].unique()
```

```
array([1056., 2600., 1440., ..., 2758.,  774., 4689.])
```

In [30]:
```
df3[~df3['total_sqft'].apply(is_float)].head(10)
#empty hence format changed...
```

	location	size	total_sqft	bath	price	bhk

Feature Engineering

In [31]:
```
df3['cost_per_sqft']=df3['price']*100000/df3['total_sqft']
```

In [32]:
```
df3.head()
```

	location	size	total_sqft	bath	price	bhk	cost_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 [33]:
```
len(df3.location.unique())
```

```
1304
```

In [34]:
```
df3.location=df3.location.apply(lambda x:x.strip(""))
location_stats=df3.groupby('location')['location'].agg('count').sort_values(ascending=False)
```

In [35]:
```
location_stats
```

```
location
Whitefield              534
Sarjapur  Road          392
Electronic City         302
Kanakpura Road          266
Thanisandra             233
                       ...
 Banaswadi                1
Kanakadasa Layout         1
Kanakapur main road       1
Kanakapura  Rod           1
whitefiled                1
Name: location, Length: 1304, dtype: int64
```

In [36]:
```
len(location_stats[location_stats<=10])
```

```
1063
```

In [37]:
```
location_stats_lessthan_10=location_stats[location_stats<=10]
location_stats_lessthan_10
```

```
location
Dodsworth Layout         10
1st Block Koramangala    10
Nagappa Reddy Layout     10
Ganga Nagar              10
Dairy Circle             10
                         ..
 Banaswadi                1
Kanakadasa Layout         1
Kanakapur main road       1
Kanakapura  Rod           1
whitefiled                1
Name: location, Length: 1063, dtype: int64
```

In [38]:
```
len(df3.location.unique())
```

```
1304
```

In [39]:
```
df3.location=df3.location.apply(lambda x:'other' if x in location_stats_lessthan_10 else x)
```

In [40]:
```
df3.location
```

```
0        Electronic City Phase II
1                Chikka Tirupathi
2                     Uttarahalli
3              Lingadheeranahalli
4                        Kothanur
                    ...
13315                   Whitefield
13316                        other
13317       Raja Rajeshwari Nagar
13318              Padmanabhanagar
13319                 Doddathoguru
Name: location, Length: 13246, dtype: object
```

In [41]:
```
len(df3.location.unique())
```

```
242
```

In [42]:
```
df3.head(10)
```

	location	size	total_sqft	bath	price	bhk	cost_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

Outliers

normally square ft per bedroom is 300

In [43]:
```
df3[df3.total_sqft/df3.bhk<300].head()
```

	location	size	total_sqft	bath	price	bhk	cost_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	other	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 [44]:
```
df3.shape
```

```
(13246, 7)
```

In [45]:
```
df4=df3[~(df3.total_sqft/df3.bhk<300)]
```

In [46]:
```
df4.shape
```

```
(12502, 7)
```

using standard deviation

In [47]:
```
df4.cost_per_sqft.describe()
```

```
count     12312.000000
mean       6323.403514
std        4187.211055
min         267.829813
25%        4208.545855
50%        5300.000000
75%        6938.987948
max      176470.588235
Name: cost_per_sqft, dtype: float64
```

In [48]:
```
def remove_cps_outliers(df):
    df_out = pd.DataFrame()
    for key, subdf in df.groupby('location'):
        m = np.mean(subdf.cost_per_sqft)
        st = np.std(subdf.cost_per_sqft)
        reduced_df = subdf[(subdf.cost_per_sqft>(m-st)) & (subdf.cost_per_sqft<=(m+st))]
        df_out = pd.concat([df_out,reduced_df],ignore_index=True)
    return df_out
df5 = remove_cps_outliers(df4)
df5.shape
```

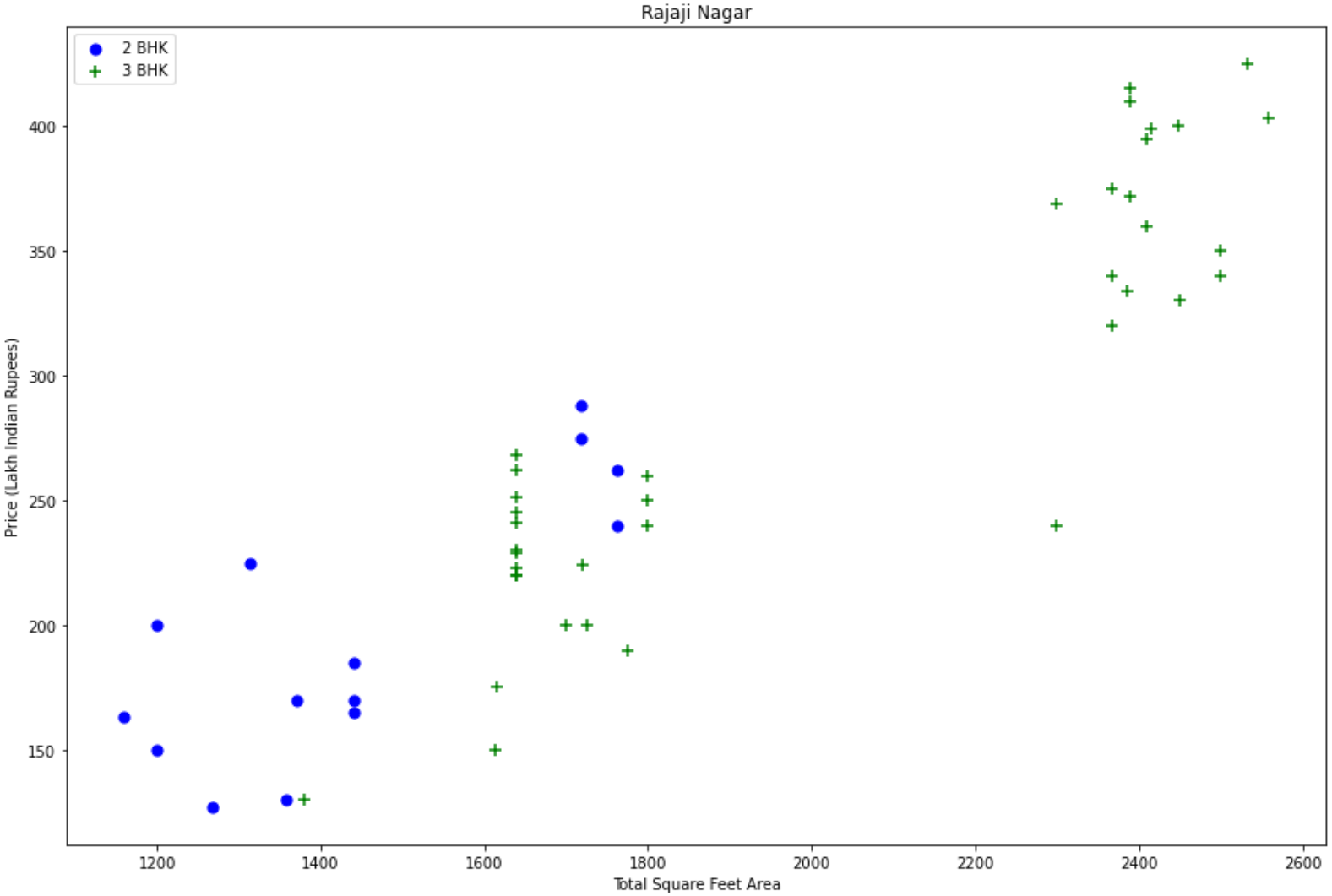
```
(10143, 7)
```

In [49]:
```
import matplotlib
```

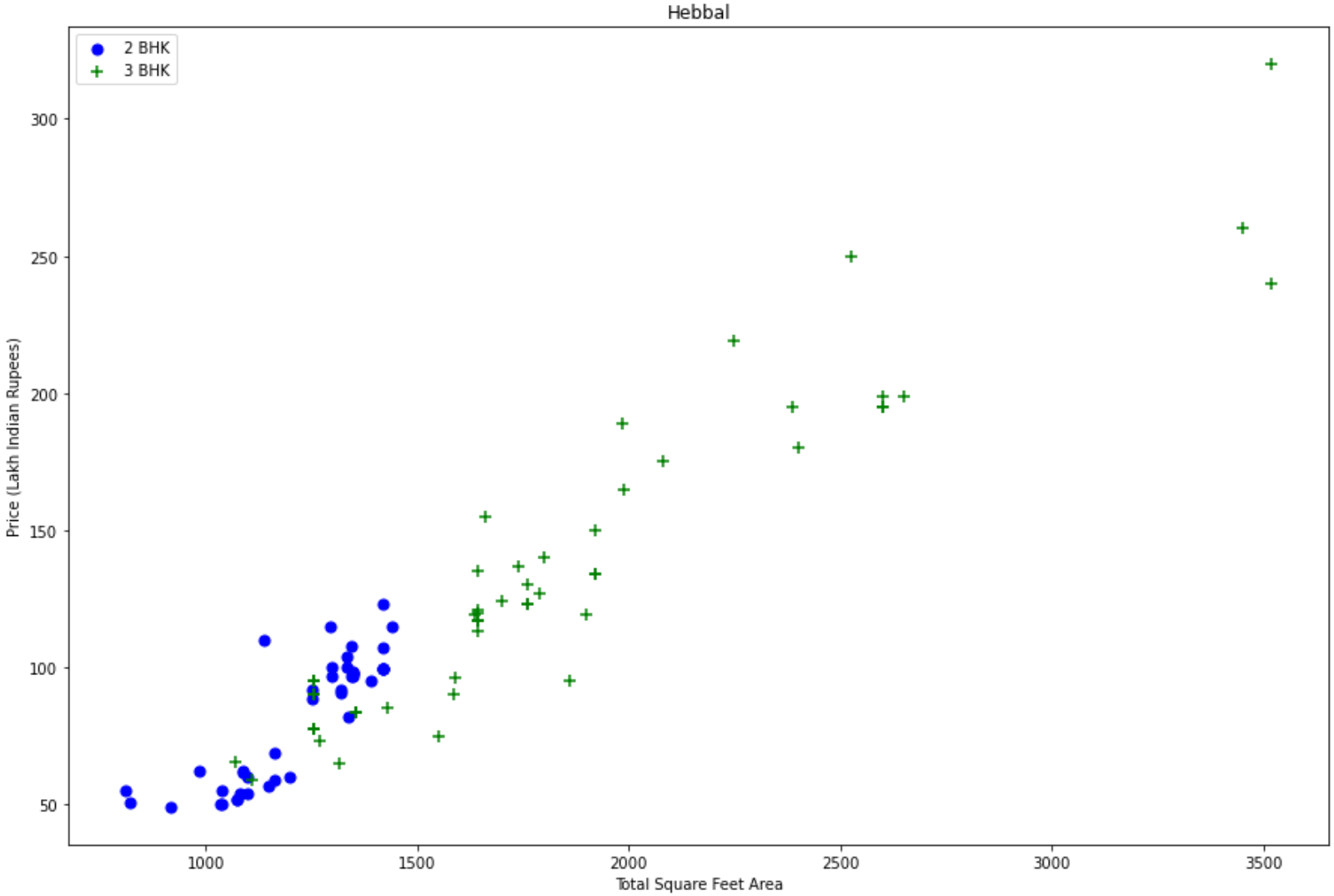
In [50]:

```
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=50)
    plt.scatter(bhk3.total_sqft,bhk3.price,marker='+', color='green',label='3 BHK', s=50)
    plt.xlabel("Total Square Feet Area")
    plt.ylabel("Price (Lakh Indian Rupees)")
    plt.title(location)
    plt.legend()

plot_scatter_chart(df5,"Rajaji Nagar")
```


Rajaji Nagar
2 BHK
3 BHK
Price (Lakh Indian Rupees)
Total Square Feet Area
150
200
250
300
350
400
1200
1400
1600
1800
2000
2200
2400
2600

In [51]:
```
plot_scatter_chart(df5,"Hebbal")
```

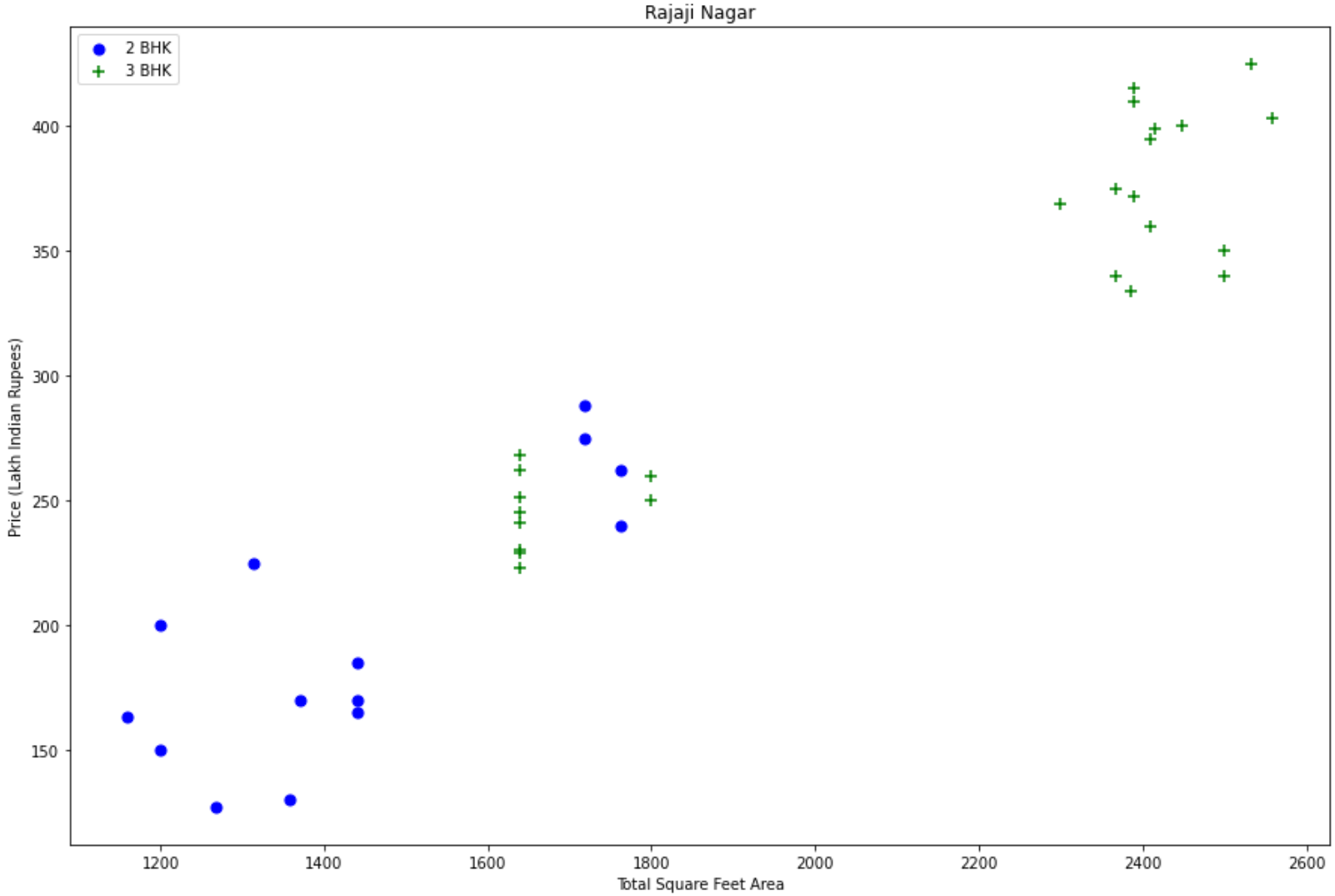


In [52]:

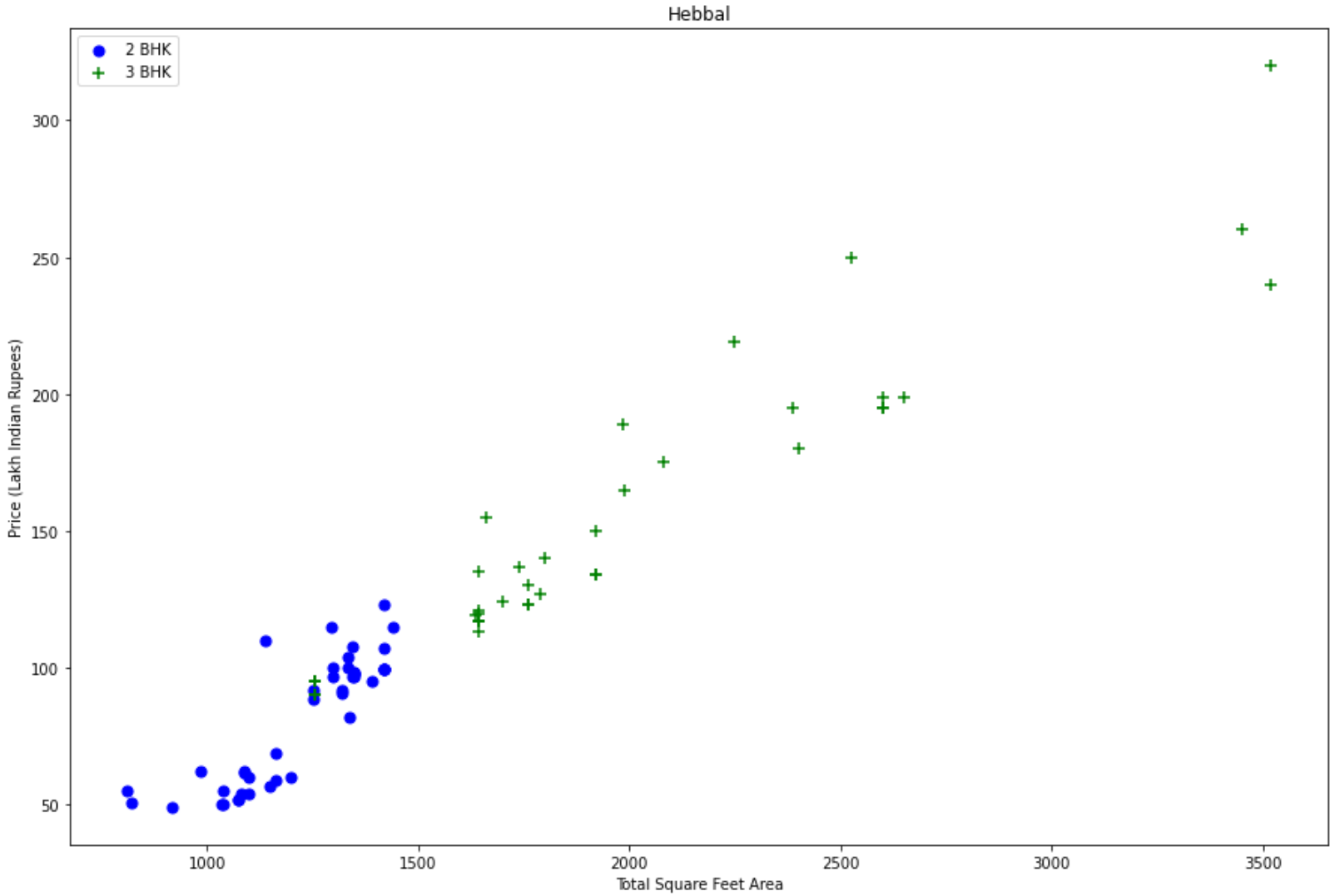
```
def remove_bhk_outliers(df):
    exclude_indices = np.array([])
    for location, location_df in df.groupby('location'):
        bhk_stats = {}
        for bhk, bhk_df in location_df.groupby('bhk'):
            bhk_stats[bhk] = {
                'mean': np.mean(bhk_df.cost_per_sqft),
                'std': np.std(bhk_df.cost_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.cost_per_sqft<(stats['mean'])].i
    return df.drop(exclude_indices,axis='index')
df6 = df5.copy()
df6 = remove_bhk_outliers(df5)
df6.shape
```

```
(7227, 7)
```

In [53]:
```
plot_scatter_chart(df6,"Rajaji Nagar")
```



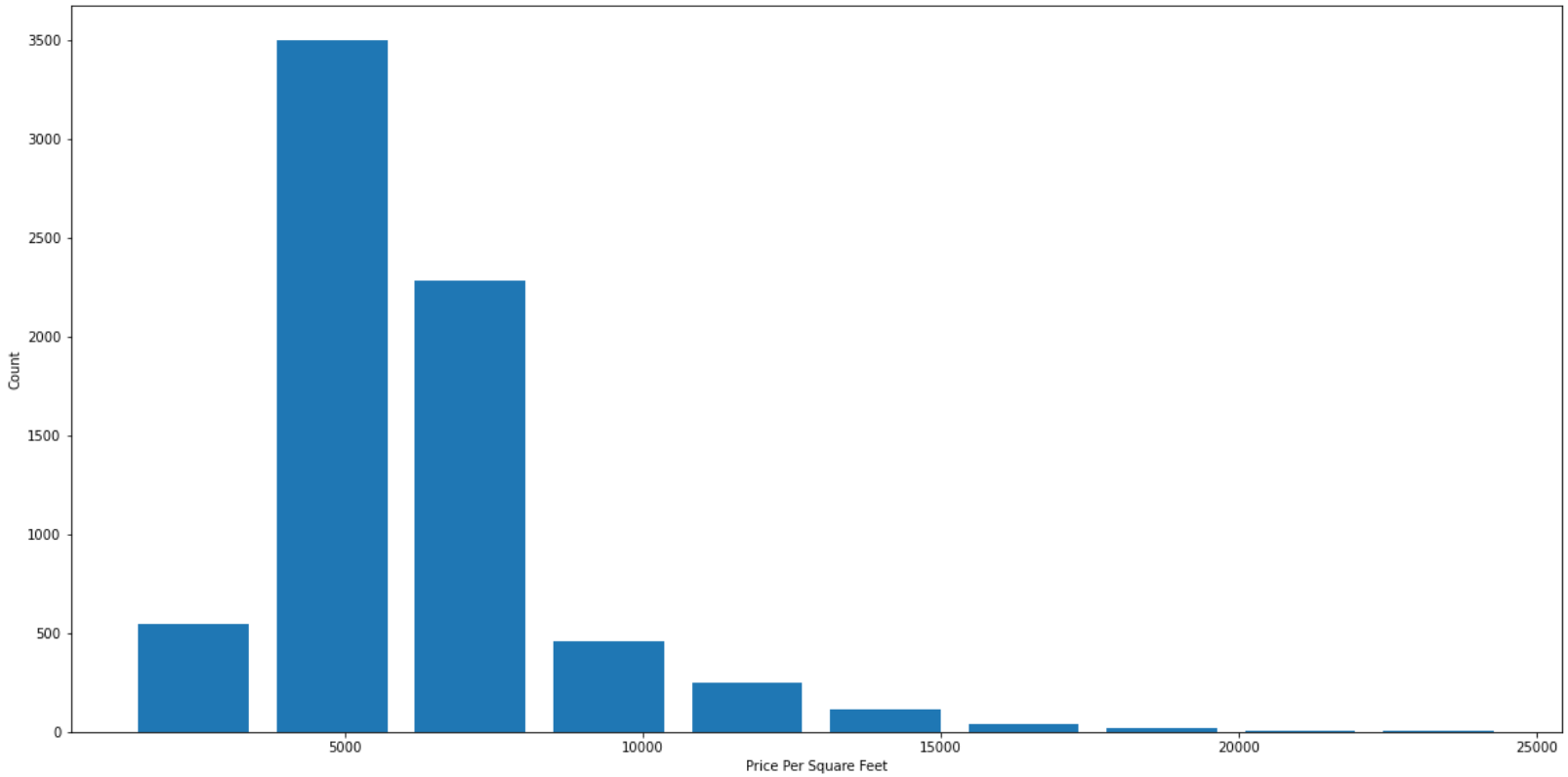
In [54]:
```
plot_scatter_chart(df6,"Hebbal")
```



In [55]:

```
import matplotlib
matplotlib.rcParams["figure.figsize"] = (20,10)
plt.hist(df6.cost_per_sqft,rwidth=0.8)
plt.xlabel("Price Per Square Feet")
plt.ylabel("Count")
```

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



Outlier Removal Using Bathrooms Feature

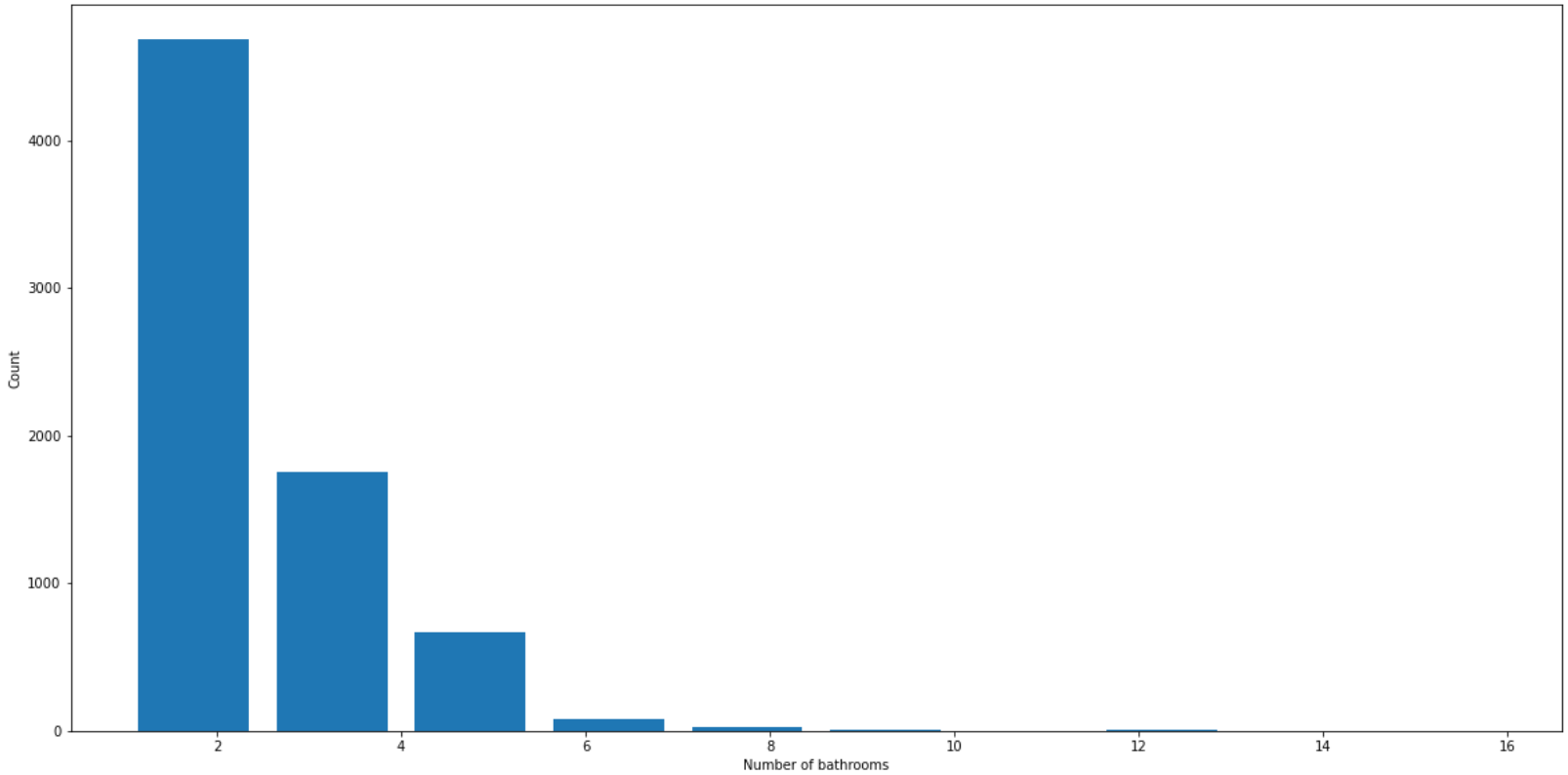
In [56]:
```
df6.bath.unique()
```

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

In [57]:

```
plt.hist(df6.bath,rwidth=0.8)
plt.xlabel("Number of bathrooms")
plt.ylabel("Count")
```

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



In [58]:
```
df6[df6.bath>10]
```

	location	size	total_sqft	bath	price	bhk	cost_per_sqft
5215	Neeladri Nagar	10 BHK	4000.0	12.0	160.0	10	4000.000000
8385	other	10 BHK	12000.0	12.0	525.0	10	4375.000000
8474	other	16 BHK	10000.0	16.0	550.0	16	5500.000000
9211	other	11 BHK	6000.0	12.0	150.0	11	2500.000000
9541	other	13 BHK	5425.0	13.0	275.0	13	5069.124424

In [59]:
```
df6[df6.bath>df6.bhk+2]
```

	location	size	total_sqft	bath	price	bhk	cost_per_sqft
1622	Chikkabanavar	4 Bedroom	2460.0	7.0	80.0	4	3252.032520
5176	Nagasandra	4 Bedroom	7000.0	8.0	450.0	4	6428.571429
6632	Thanisandra	3 BHK	1806.0	6.0	116.0	3	6423.034330
8309	other	6 BHK	11338.0	9.0	1000.0	6	8819.897689

In [60]:
```
df6.shape
```

```
(7227, 7)
```

In [61]:
```
df6.head(2)
```

	location	size	total_sqft	bath	price	bhk	cost_per_sqft
1	Devarachikkanahalli	2 BHK	1250.0	2.0	40.0	2	3200.000000
2	Devarachikkanahalli	2 Bedroom	1200.0	2.0	83.0	2	6916.666667

In [62]:
```
df7 = df6.drop(['size','cost_per_sqft'],axis='columns')
df7.head(3)
```

	location	total_sqft	bath	price	bhk
1	Devarachikkanahalli	1250.0	2.0	40.0	2
2	Devarachikkanahalli	1200.0	2.0	83.0	2
3	Devarachikkanahalli	1170.0	2.0	40.0	2

Use One Hot Encoding For Location

In [63]:
```
dummies = pd.get_dummies(df7.location)
dummies.head(3)
```

	Devarachikkanahalli	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	...	Vishveshwarya Layout	Vi
1	1	0	0	0	0	0	0	0	0	0	...	0	0
2	1	0	0	0	0	0	0	0	0	0	...	0	0
3	1	0	0	0	0	0	0	0	0	0	...	0	0

3 rows × 242 columns

In [64]:
```
df8 = pd.concat([df7,dummies.drop('other',axis='columns')],axis='columns')
df8.head()
```

	location	total_sqft	bath	price	bhk	Devarachikkanahalli	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	...	Vijayanaga
1	Devarachikkanahalli	1250.0	2.0	40.0	2	1	0	0	0	0	...	0
2	Devarachikkanahalli	1200.0	2.0	83.0	2	1	0	0	0	0	...	0
3	Devarachikkanahalli	1170.0	2.0	40.0	2	1	0	0	0	0	...	0
4	Devarachikkanahalli	1425.0	2.0	65.0	3	1	0	0	0	0	...	0
5	Devarachikkanahalli	947.0	2.0	43.0	2	1	0	0	0	0	...	0

5 rows × 246 columns

In [65]:
```
df8.tail()
```

	location	total_sqft	bath	price	bhk	Devarachikkanahalli	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	...	Vijayanagar	Vi
10134	other	1200.0	2.0	70.0	2	0	0	0	0	0	...	0	0
10135	other	1800.0	1.0	200.0	1	0	0	0	0	0	...	0	0
10138	other	1353.0	2.0	110.0	2	0	0	0	0	0	...	0	0
10139	other	812.0	1.0	26.0	1	0	0	0	0	0	...	0	0
10142	other	3600.0	5.0	400.0	4	0	0	0	0	0	...	0	0

5 rows × 246 columns

In [66]:
```
df9 = df8.drop('location',axis='columns')
df9.head(2)
```

	total_sqft	bath	price	bhk	Devarachikkanahalli	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	5th Block Hbr Layout	...	Vijayanagar	Vishves
1	1250.0	2.0	40.0	2	1	0	0	0	0	0	...	0	0
2	1200.0	2.0	83.0	2	1	0	0	0	0	0	...	0	0

2 rows × 245 columns

Model building

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

```
(7227, 245)
```

In [68]:
```
X = df9.drop(['price'],axis='columns')
X.head(3)
```

	total_sqft	bath	bhk	Devarachikkanahalli	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	5th Block Hbr Layout	5th Phase JP Nagar	...	Vijayanagar	Vishve
1	1250.0	2.0	2	1	0	0	0	0	0	0	...	0	0
2	1200.0	2.0	2	1	0	0	0	0	0	0	...	0	0
3	1170.0	2.0	2	1	0	0	0	0	0	0	...	0	0

3 rows × 244 columns

In [69]:
```
X.shape
```

```
(7227, 244)
```

In [70]:
```
y=df9.price
y.head()
```

```
1    40.0
2    83.0
3    40.0
4    65.0
5    43.0
Name: price, dtype: float64
```

In [71]:
```
len(y)
```

```
7227
```

In [72]:
```
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=10)
```

In [73]:
```
from sklearn.linear_model import LinearRegression
lr_clf = LinearRegression()
lr_clf.fit(X_train,y_train)
lr_clf.score(X_test,y_test)
```

```
0.8756701257688209
```

Use K Fold cross validation to measure accuracy of our LinearRegression model

In [74]:
```
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)
```

```
array([0.79826963, 0.8119522 , 0.89140998, 0.82271717, 0.87656831])
```

Find best model using GridSearchCV

In [75]:

```
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_score=False)
        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)
```

	model	best_score	best_params
0	linear_regression	0.840183	{'normalize': False}
1	lasso	0.697018	{'alpha': 2, 'selection': 'random'}
2	decision_tree	0.704098	{'criterion': 'mse', 'splitter': 'best'}

Test the model for few properties

In [76]:
```
X.columns
```

```
Index(['total_sqft', 'bath', 'bhk', ' Devarachikkanahalli',
       '1st Block Jayanagar', '1st Phase JP Nagar',
       '2nd Phase Judicial Layout', '2nd Stage Nagarbhavi',
       '5th Block Hbr Layout', '5th Phase JP Nagar',
       ...
       'Vijayanagar', 'Vishveshwarya Layout', 'Vishwapriya Layout',
       'Vittasandra', 'Whitefield', 'Yelachenahalli', 'Yelahanka',
       'Yelahanka New Town', 'Yelenahalli', 'Yeshwanthpur'],
      dtype='object', length=244)
```

In [77]:
```
np.where(X.columns=='2nd Stage Nagarbhavi')
```

```
(array([7], dtype=int64),)
```

In [78]:
```
np.where(X.columns=='2nd Stage Nagarbhavi')[0][0]
```

```
7
```

In [79]:
```
def predict_price(location,sqft,bath,bhk):
    loc_index = np.where(X.columns==location)[0][0]

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

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

In [80]:
```
predict_price('1st Phase JP Nagar',1000, 2, 2)
```

```
90.32236170239628
```

In [81]:
```
predict_price('Indira Nagar',1000, 2, 2)
```

```
184.7755966586537
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: