ML_Project

November 29, 2020

[121]:
```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
%matplotlib inline
import matplotlib
matplotlib.rcParams["figure.figsize"] = (20,10)
```

[124]:
```
df1 = pd.read_csv("/home/archit/jupyter_folder/Bengaluru_House_Data.csv")
df1.head()
```

[124]:

	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

[125]:
```
df1.shape
```

[125]:
```
(13320, 9)
```

[126]:
```
df1.columns
```

[126]:
```
Index(['area_type', 'availability', 'location', 'size', 'society',
       'total_sqft', 'bath', 'balcony', 'price'],
      dtype='object')
```

[127]:
```
df1['area_type'].unique()
```

[127]:
```
array(['Super built-up  Area', 'Plot  Area', 'Built-up  Area',
       'Carpet  Area'], dtype=object)
```

[128]:
```
df1['area_type'].value_counts()
```

[128]:
```
Super built-up  Area    8790
Built-up  Area          2418
Plot  Area              2025
Carpet  Area              87
Name: area_type, dtype: int64
```

[129]:
```
df2 = df1.drop(['area_type','society','balcony','availability'],axis='columns')
df2.shape
```

[129]:
```
(13320, 5)
```

[130]:
```
df2.isnull().sum()
```

[130]:
```
location       1
size          16
total_sqft     0
bath          73
price          0
dtype: int64
```

[131]:
```
df3 = df2.dropna()
df3.isnull().sum()
```

[131]:
```
location      0
size          0
total_sqft    0
bath          0
price         0
dtype: int64
```

[132]:
```
df3.shape
```

[132]:
```
(13246, 5)
```

[133]:
```
df3['bhk'] = df3['size'].apply(lambda x: int(x.split(' ')[0]))
df3.bhk.unique()
```

```
<ipython-input-133-681cf3aca53d>:1: 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/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df3['bhk'] = df3['size'].apply(lambda x: int(x.split(' ')[0]))
```

[133]:
```
array([ 2,  4,  3,  6,  1,  8,  7,  5, 11,  9, 27, 10, 19, 16, 43, 14, 12,
       13, 18])
```

[134]:
```
def is_float(x):
    try:
        float(x)
    except:
        return False
    return True
```

[135]:
```
df3[~df3['total_sqft'].apply(is_float)].head(10)
```

[135]:

	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

[136]:
```
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
```

[137]:
```
df4 = df3.copy()
df4.total_sqft = df4.total_sqft.apply(convert_sqft_to_num)
df4 = df4[df4.total_sqft.notnull()]
df4.head(2)
```

[137]:

	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

[138]:
```
df4.loc[30]
```

[138]:
```
location      Yelahanka
size              4 BHK
total_sqft         2475
```

```
bath                  4
price               186
bhk                   4
Name: 30, dtype: object
```

[139]:
```
(2100+2850)/2
```

[139]:
```
2475.0
```

[140]:
```
df5 = df4.copy()
df5['price_per_sqft'] = df5['price']*100000/df5['total_sqft']
df5.head()
```

[140]:

	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

[141]:
```
df5_stats = df5['price_per_sqft'].describe()
df5_stats
```

[141]:
```
count    1.320000e+04
mean     7.920759e+03
std      1.067272e+05
min      2.678298e+02
25%      4.267701e+03
50%      5.438331e+03
75%      7.317073e+03
max      1.200000e+07
Name: price_per_sqft, dtype: float64
```

[142]:
```
df5.to_csv("bhp.csv",index=False)
```

[143]:
```
df5.location = df5.location.apply(lambda x: x.strip())
location_stats = df5['location'].value_counts(ascending=False)
location_stats
```

[143]:
```
Whitefield         533
Sarjapur  Road     392
Electronic City    304
Kanakpura Road     264
Thanisandra        235
                   …
JakkurYelahanka      1
Double Road          1
1Kasavanhalli        1
Hoskote near         1
Allalasandra         1
Name: location, Length: 1287, dtype: int64
```

[144]:
```
len(location_stats[location_stats<=10])
```

[144]:
```
1047
```

[145]:
```
location_stats_less_than_10 = location_stats[location_stats<=10]
location_stats_less_than_10
```

[145]:
```
Kalkere                 10
Dodsworth Layout        10
Sadashiva Nagar         10
1st Block Koramangala   10
Basapura                10
                        ..
JakkurYelahanka          1
Double Road              1
1Kasavanhalli            1
Hoskote near             1
Allalasandra             1
Name: location, Length: 1047, dtype: int64
```

[146]:
```
len(df5.location.unique())
```

[146]:
```
1287
```

[147]:
```
df5.location = df5.location.apply(lambda x: 'other' if x in␣
 ↪location_stats_less_than_10 else x)
len(df5.location.unique())
```

[147]:
```
241
```

[148]:
```
df5.head(10)
```

[148]:

	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
8              Marathahalli      3 BHK      1310.0   3.0   63.25    3
9                     other  6 Bedroom      1020.0   6.0  370.00    6

   price_per_sqft
0     3699.810606
1     4615.384615
2     4305.555556
3     6245.890861
4     4250.000000
5     3247.863248
6     7467.057101
7    18181.818182
8     4828.244275
9    36274.509804
```

[149]:
```
df5[df5.total_sqft/df5.bhk<300].head()
```

[149]:
```
               location       size  total_sqft  bath  price  bhk  \
9                 other  6 Bedroom      1020.0   6.0  370.0    6
45          HSR Layout  8 Bedroom       600.0   9.0  200.0    8
58        Murugeshpalya  6 Bedroom      1407.0   4.0  150.0    6
68  Devarachikkanahalli  8 Bedroom      1350.0   7.0   85.0    8
70                other  3 Bedroom       500.0   3.0  100.0    3

    price_per_sqft
9     36274.509804
45    33333.333333
58    10660.980810
68     6296.296296
70    20000.000000
```

[150]:
```
df6 = df5[~(df5.total_sqft/df5.bhk<300)]
df6.shape
```

[150]:
```
(12456, 7)
```

[151]:
```
df6.price_per_sqft.describe()
```

[151]:
```
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
```

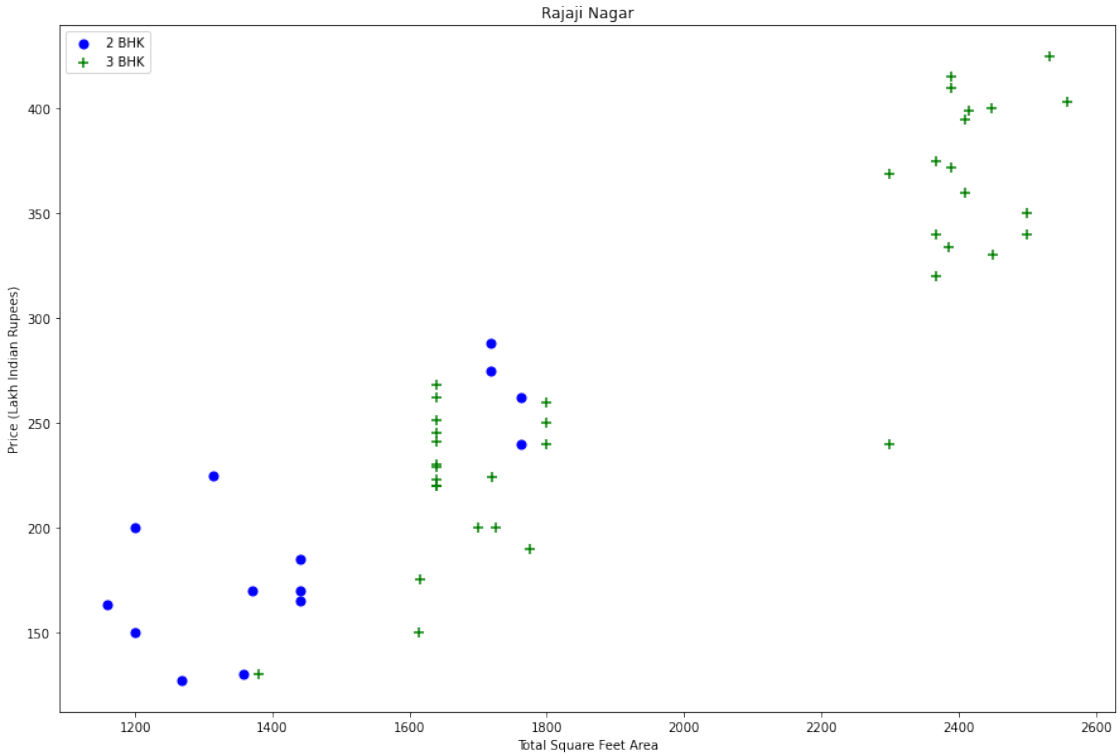
[152]:
```
def remove_pps_outliers(df):
    df_out = pd.DataFrame()
    for key, subdf in df.groupby('location'):
        m = np.mean(subdf.price_per_sqft)
        st = np.std(subdf.price_per_sqft)
        reduced_df = subdf[(subdf.price_per_sqft>(m-st)) & (subdf.
 ↪price_per_sqft<=(m+st))]
        df_out = pd.concat([df_out,reduced_df],ignore_index=True)
    return df_out
df7 = remove_pps_outliers(df6)
df7.shape
```

[152]:
```
(10242, 7)
```

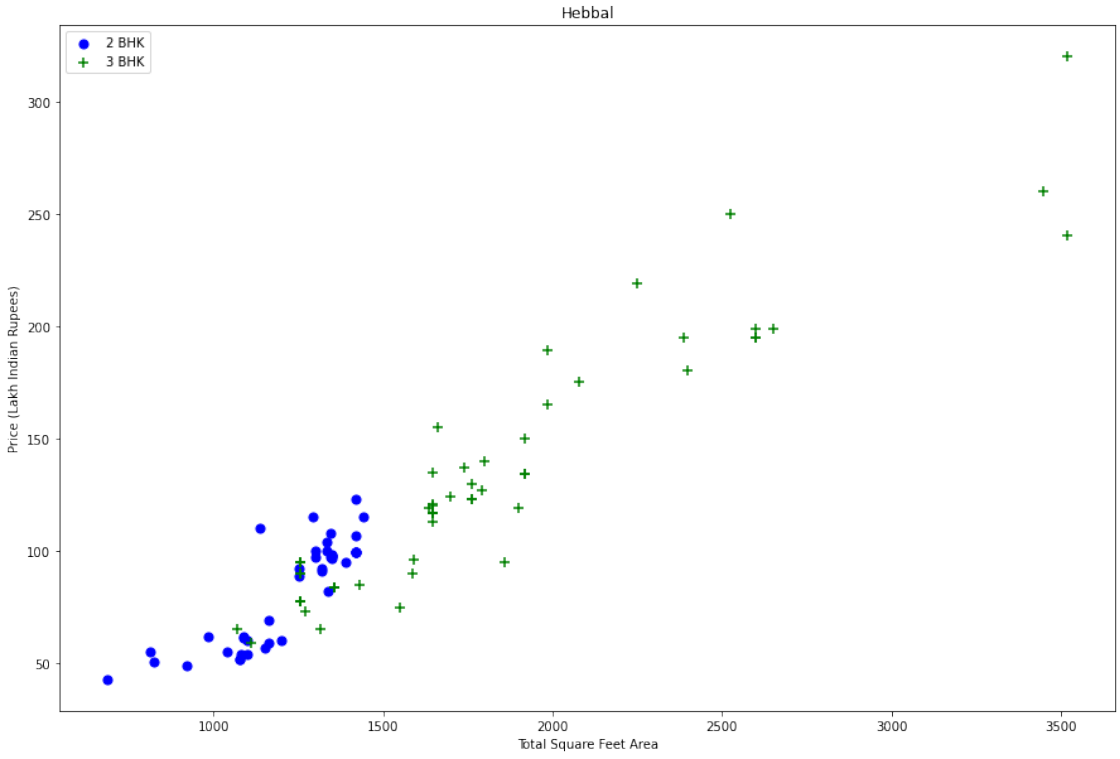
[154]:
```
print("hellp")
```

```
hellp
```

[153]:
```
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(df7,"Rajaji Nagar")
```



[155]:
```
plot_scatter_chart(df7,"Hebbal")
```



[157]:
```
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_per_sqft<(stats['mean'])].index.values)
    return df.drop(exclude_indices,axis='index')
df8 = reove_bhk_outliers(df7)
# df8 = df7.copy()
df8.shape
```

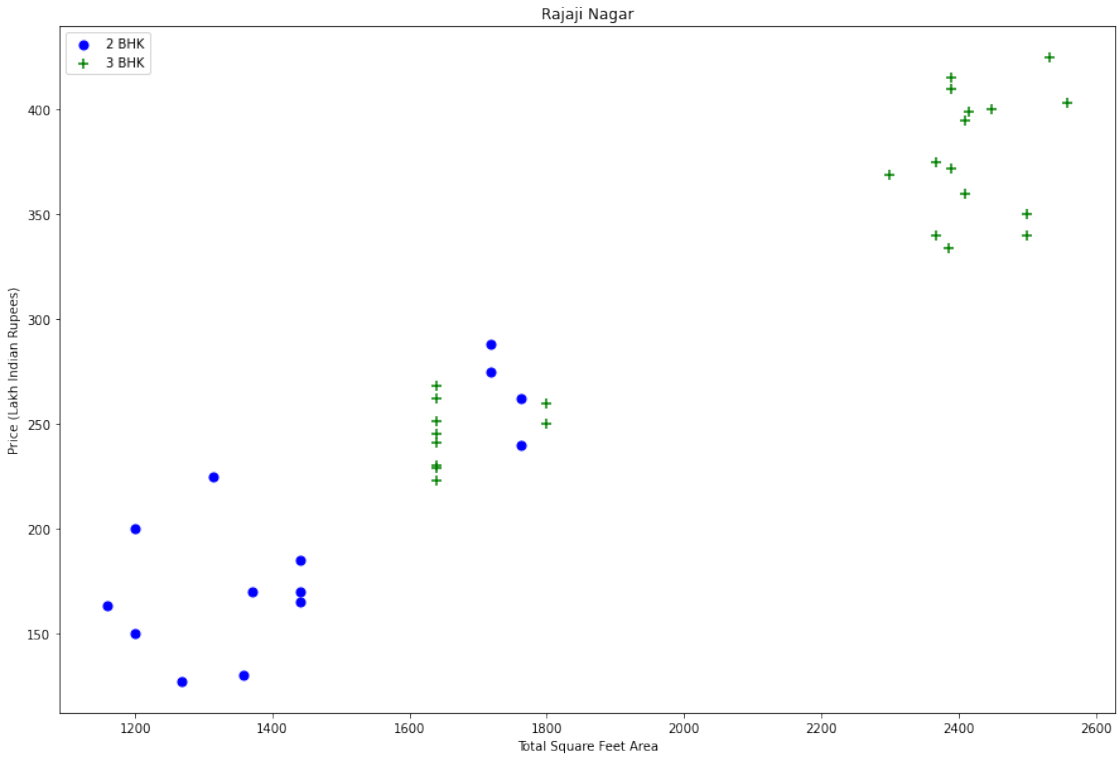
```
      ␣
↪---------------------------------------------------------------------------

       NameError                                 Traceback (most recent call␣
↪last)

       <ipython-input-157-39d7ed3d2f65> in <module>
        14                 exclude_indices = np.append(exclude_indices,␣
↪bhk_df[bhk_df.price_per_sqft<(stats['mean'])].index.values)
        15     return df.drop(exclude_indices,axis='index')
   ---> 16 df8 = reove_bhk_outliers(df7)
        17 # df8 = df7.copy()
        18 df8.shape


       NameError: name 'reove_bhk_outliers' is not defined
```

[158]:
```
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'):
```

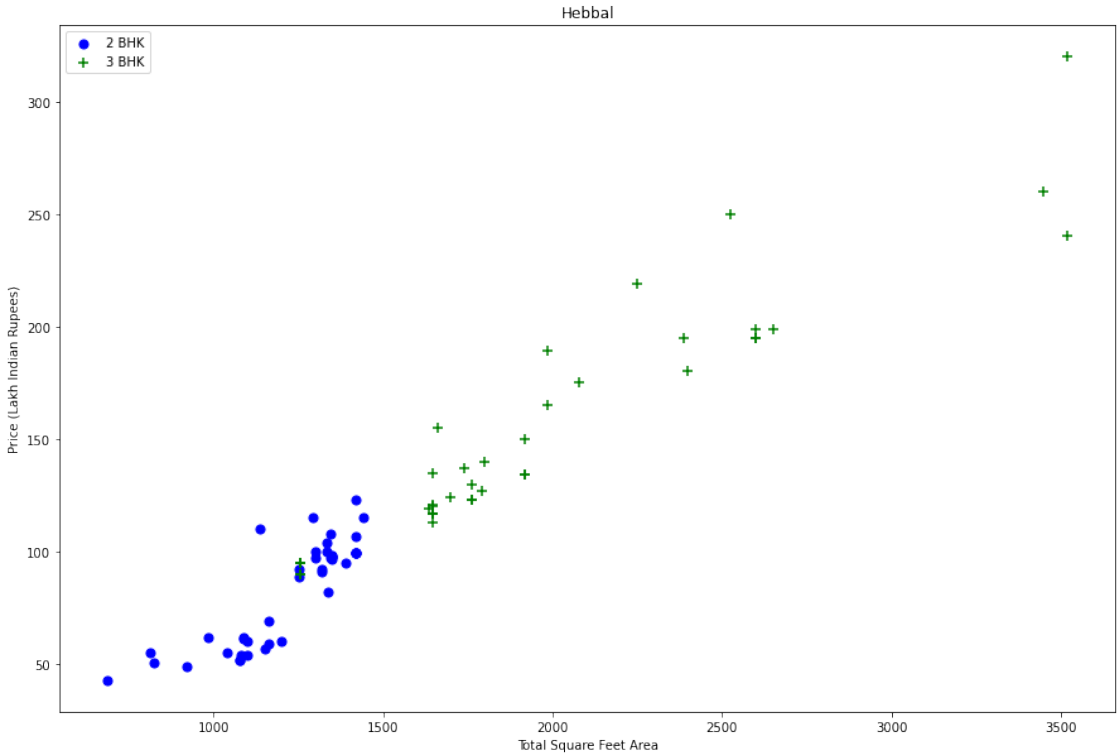
```
            bhk_stats[bhk] = {
                'mean': np.mean(bhk_df.price_per_sqft),
                'std': np.std(bhk_df.price_per_sqft),
                'count': bhk_df.shape[0]
            }
        for bhk, bhk_df in location_df.groupby('bhk'):
            stats = bhk_stats.get(bhk-1)
            if stats and stats['count']>5:
                exclude_indices = np.append(exclude_indices, bhk_df[bhk_df.
↪price_per_sqft<(stats['mean'])].index.values)
    return df.drop(exclude_indices,axis='index')
df8 = remove_bhk_outliers(df7)
# df8 = df7.copy()
df8.shape
```

[158]: (7317, 7)

[159]:
```
plot_scatter_chart(df8,"Rajaji Nagar")
```

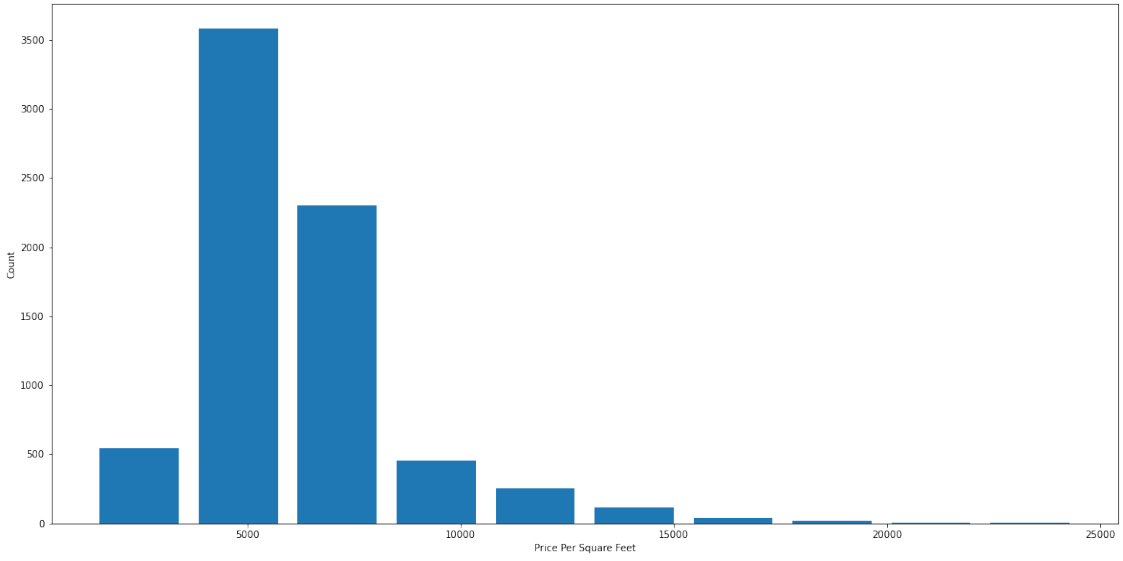


[160]:
```
plot_scatter_chart(df8,"Hebbal")
```



[161]:
```
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")
```

[161]: Text(0, 0.5, 'Count')



[162]:
```
d
```

```
␣
↪---------------------------------------------------------------------------

NameError                                 Traceback (most recent call␣
↪last)

<ipython-input-162-e983f374794d> in <module>
----> 1 d

NameError: name 'd' is not defined
```

[163]:
```
s
```

```
␣
↪---------------------------------------------------------------------------

NameError                                 Traceback (most recent call␣
↪last)

<ipython-input-163-ded5ba42480f> in <module>
----> 1 s

NameError: name 's' is not defined
```

[164]:
```
s
```

```
␣
↪---------------------------------------------------------------------------

NameError                                 Traceback (most recent call␣
↪last)

<ipython-input-164-ded5ba42480f> in <module>
----> 1 s
```

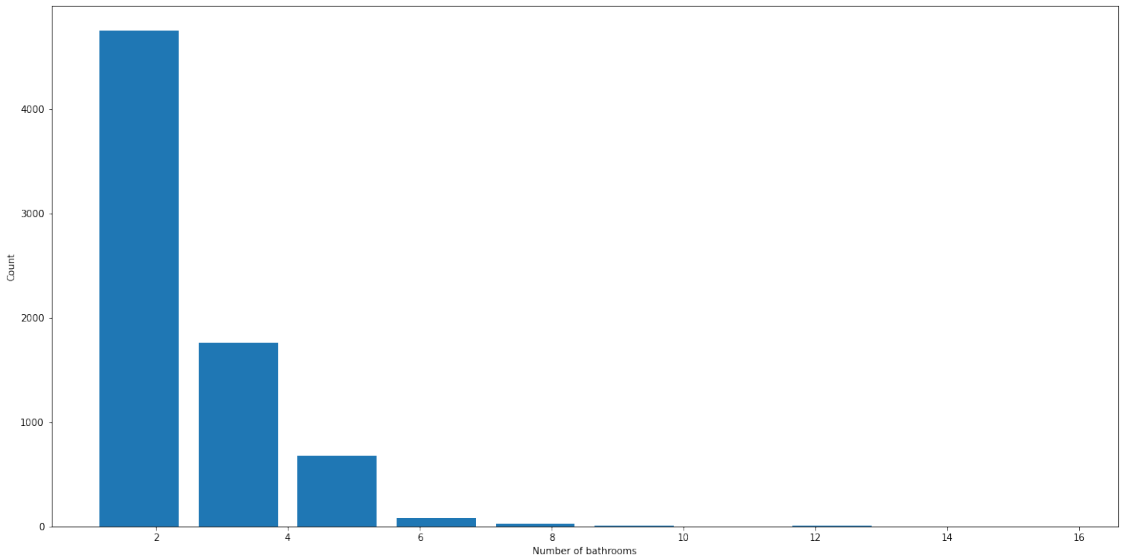
```
NameError: name 's' is not defined
```

[165]:
```
df8.bath.unique()
```

[165]:
```
array([ 4.,  3.,  2.,  5.,  8.,  1.,  6.,  7.,  9., 12., 16., 13.])
```

[166]:
```
plt.hist(df8.bath,rwidth=0.8)
plt.xlabel("Number of bathrooms")
plt.ylabel("Count")
```

[166]:
```
Text(0, 0.5, 'Count')
```



[167]:
```
df8[df8.bath>10]
```

[167]:

	location	size	total_sqft	bath	price	bhk	price_per_sqft
5277	Neeladri Nagar	10 BHK	4000.0	12.0	160.0	10	4000.000000
8483	other	10 BHK	12000.0	12.0	525.0	10	4375.000000
8572	other	16 BHK	10000.0	16.0	550.0	16	5500.000000
9306	other	11 BHK	6000.0	12.0	150.0	11	2500.000000
9637	other	13 BHK	5425.0	13.0	275.0	13	5069.124424

[168]:
```
df8[df8.bath>df8.bhk+2]
```

[168]:

	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
8408	other	6 BHK	11338.0	9.0	1000.0	6	8819.897689

[169]:
```
df9 = df8[df8.bath<df8.bhk+2]
df9.shape
```

[169]:
```
(7239, 7)
```

[170]:
```
df9.head(2)
```

[170]:

	location	size	total_sqft	bath	price	bhk	price_per_sqft
0	1st Block Jayanagar	4 BHK	2850.0	4.0	428.0	4	15017.543860
1	1st Block Jayanagar	3 BHK	1630.0	3.0	194.0	3	11901.840491

[171]:
```
df10 = df9.drop(['size','price_per_sqft'],axis='columns')
df10.head(3)
```

[171]:

	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

[172]:
```
dummies = pd.get_dummies(df10.location)
dummies.head(3)
```

[172]:

	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	…	Vishveshwarya Layout	Vishwapriya Layout	Vittasandra	Whitefield	Yelachenahalli	Yelahanka	Yelahanka New Town
0	1	0	0	0	0	0	0	0	0	0	…	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	…	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0	0	…	0	0	0	0	0	0	0

```
   Yelenahalli  Yeshwanthpur  other
0            0             0      0
1            0             0      0
2            0             0      0

[3 rows x 241 columns]
```

[173]:
```
df11 = pd.concat([df10,dummies.drop('other',axis='columns')],axis='columns')
df11.head(
```

```
  File "<ipython-input-173-2567da96c7fa>", line 2
    df11.head(
             ^
SyntaxError: unexpected EOF while parsing
```

[174]:
```
df11 = pd.concat([df10,dummies.drop('other',axis='columns')],axis='columns')
df11.head()
```

[174]:
```
              location  total_sqft  bath  price  bhk  1st Block Jayanagar  \
0  1st Block Jayanagar      2850.0   4.0  428.0    4                    1
1  1st Block Jayanagar      1630.0   3.0  194.0    3                    1
2  1st Block Jayanagar      1875.0   2.0  235.0    3                    1
3  1st Block Jayanagar      1200.0   2.0  130.0    3                    1
4  1st Block Jayanagar      1235.0   2.0  148.0    2                    1

   1st Phase JP Nagar  2nd Phase Judicial Layout  2nd Stage Nagarbhavi  \
0                   0                          0                     0
1                   0                          0                     0
2                   0                          0                     0
3                   0                          0                     0
4                   0                          0                     0

   5th Block Hbr Layout  …  Vijayanagar  Vishveshwarya Layout  \
0                     0  …           0                     0
1                     0  …           0                     0
2                     0  …           0                     0
3                     0  …           0                     0
4                     0  …           0                     0

   Vishwapriya Layout  Vittasandra  Whitefield  Yelachenahalli  Yelahanka  \
0                   0            0           0               0          0
1                   0            0           0               0          0
2                   0            0           0               0          0
3                   0            0           0               0          0
```

```
4                    0            0           0                0          0

   Yelahanka New Town  Yelenahalli  Yeshwanthpur
0                   0            0             0
1                   0            0             0
2                   0            0             0
3                   0            0             0
4                   0            0             0

[5 rows x 245 columns]
```

[175]:
```
df12 = df11.drop('location',axis='columns')
df12.head(2)
```

[175]:
```
   total_sqft  bath  price  bhk  1st Block Jayanagar  1st Phase JP Nagar  \
0      2850.0   4.0  428.0    4                    1                   0
1      1630.0   3.0  194.0    3                    1                   0

   2nd Phase Judicial Layout  2nd Stage Nagarbhavi  5th Block Hbr Layout  \
0                          0                     0                     0
1                          0                     0                     0

   5th Phase JP Nagar  …  Vijayanagar  Vishveshwarya Layout  \
0                   0  …            0                     0
1                   0  …            0                     0

   Vishwapriya Layout  Vittasandra  Whitefield  Yelachenahalli  Yelahanka  \
0                   0            0           0               0          0
1                   0            0           0               0          0

   Yelahanka New Town  Yelenahalli  Yeshwanthpur
0                   0            0             0
1                   0            0             0

[2 rows x 244 columns]
```

[176]:
```
l
```

```
      ␣
↪---------------------------------------------------------------------------

       NameError                                 Traceback (most recent call␣
↪last)

       <ipython-input-176-cde25b5e10ad> in <module>
   ----> 1 l
```

```
NameError: name 'l' is not defined
```

[177]:
```
s
```

```
␣
↪---------------------------------------------------------------------------

NameError                                 Traceback (most recent call␣
↪last)

<ipython-input-177-ded5ba42480f> in <module>
----> 1 s

NameError: name 's' is not defined
```

[178]:
```
X = df12.drop(['price'],axis='columns')
X.head(3)
```

[178]:

	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
0	2850.0	4.0	4	1	0	0	0	0	0	0	…	0	0	0	0	0	0	0	0	0	0
1	1630.0	3.0	3	1	0	0	0	0	0	0	…	0	0	0	0	0	0	0	0	0	0
2	1875.0	2.0	3	1	0	0	0	0	0	0	…	0	0	0	0	0	0	0	0	0	0

```
2              0          0                    0            0             0

[3 rows x 243 columns]
```

[179]:
```
X.shape
```

[179]:
```
(7239, 243)
```

[180]:
```
y = df12.price
y.head(3)
```

[180]:
```
0    428.0
1    194.0
2    235.0
Name: price, dtype: float64
```

[181]:
```
len(y)
```

[181]:
```
7239
```

[182]:
```
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)
```

[189]:
```
from sklearn.linear_model import LinearRegression
lr_clf = LinearRegression()
lr_clf.fit(X_train,y_train)
lr_clf.score(X_test,y_test)
```

[189]:
```
0.8629132245229443
```

[199]:
```
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)
```

[199]:
```
array([0.82702546, 0.86027005, 0.85322178, 0.8436466 , 0.85481502])
```

[205]:
```
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)
```

[205]:

	model	best_score	best_params
0	linear_regression	0.847796	{'normalize': False}
1	lasso	0.726774	{'alpha': 2, 'selection': 'random'}
2	decision_tree	0.716591	{'criterion': 'friedman_mse', 'splitter': 'best'}

[206]:
```
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]
```

[207]:
```
predict_price('1st Phase JP Nagar',1000, 2, 2)
```

[207]:
```
83.86570258312189
```

[208]:
```
predict_price('1st Phase JP Nagar',1000, 3, 3)
```

[208]:
```
86.08062284986961
```

[209]:
```
predict_price('Indira Nagar',1000, 3, 3)
```

[209]:
```
195.5268975985465
```

[210]:
```
import pickle
with open('banglore_home_prices_model.pickle','wb') as f:
    pickle.dump(lr_clf,f)
```

[211]:
```
import json
columns = {
    'data_columns' : [col.lower() for col in X.columns]
}
with open("columns.json","w") as f:
    f.write(json.dumps(columns))
```

[ ]:
```

```