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

```
df1 = pd.read_csv("/content/realestate.csv")
df1.head()
```

	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
1	Plot Area	Ready To Move	Chikka Tirupathi	4 Bedroom	Theanmp	2600	5.0	3
2	Built-up Area	Ready To Move	Uttarahalli	3 BHK	NaN	1440	2.0	3

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

```
(13320, 5)
```

```
df3 = df2.dropna()
df3.isnull().sum()
```

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

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

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py: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_
  """Entry point for launching an IPython kernel.
array([ 2,  4,  3,  6,  1,  8,  7,  5, 11,  9, 27, 10, 19, 16, 43, 14, 12,
       13, 18])
```

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

```
df3[~df3['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

```
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
```

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

	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

```
df4.loc[30]
```

```
location      Yelahanka
size              4 BHK
total_sqft         2475
bath                  4
price               186
bhk                   4
Name: 30, dtype: object
```

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

```
len(df5.location.unique())
```

```
1298
```

```
df5['location']=df5['location'].apply(lambda x : x.strip())
locationstats = df5.groupby('location')['location'].agg('count').sort_values(ascending=False)
locationstats
```

```
location
Whitefield              533
Sarjapur  Road          392
Electronic City         304
Kanakpura Road          264
Thanisandra             235
                       ... 
Kumbhena Agrahara         1
Kudlu Village,            1
Konappana Agrahara        1
Kodanda Reddy Layout      1
1 Annasandrapalya         1
Name: location, Length: 1287, dtype: int64
```

```
len(locationstats[locationstats<=10])
```

```
1047
```

```
locationstatslessthanten=locationstats[locationstats<=10]
```

```
locationstatslessthanten
```

```
location
Dodsworth Layout        10
BTM 1st Stage           10
Sadashiva Nagar         10
Thyagaraja Nagar        10
Kalkere                 10
                        ..
Kumbhena Agrahara        1
Kudlu Village,           1
Konappana Agrahara       1
```

```
Kodanda Reddy Layout     1
1 Annasandrapalya        1
Name: location, Length: 1047, dtype: int64
```

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

```
241
```

```
#Detecting outliers
df5.head()
```

	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

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

	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

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

```
df6.shape
```

```
(12456, 7)
```

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

```
count    12456.000000
mean      6308.502826
std       4168.127339
```

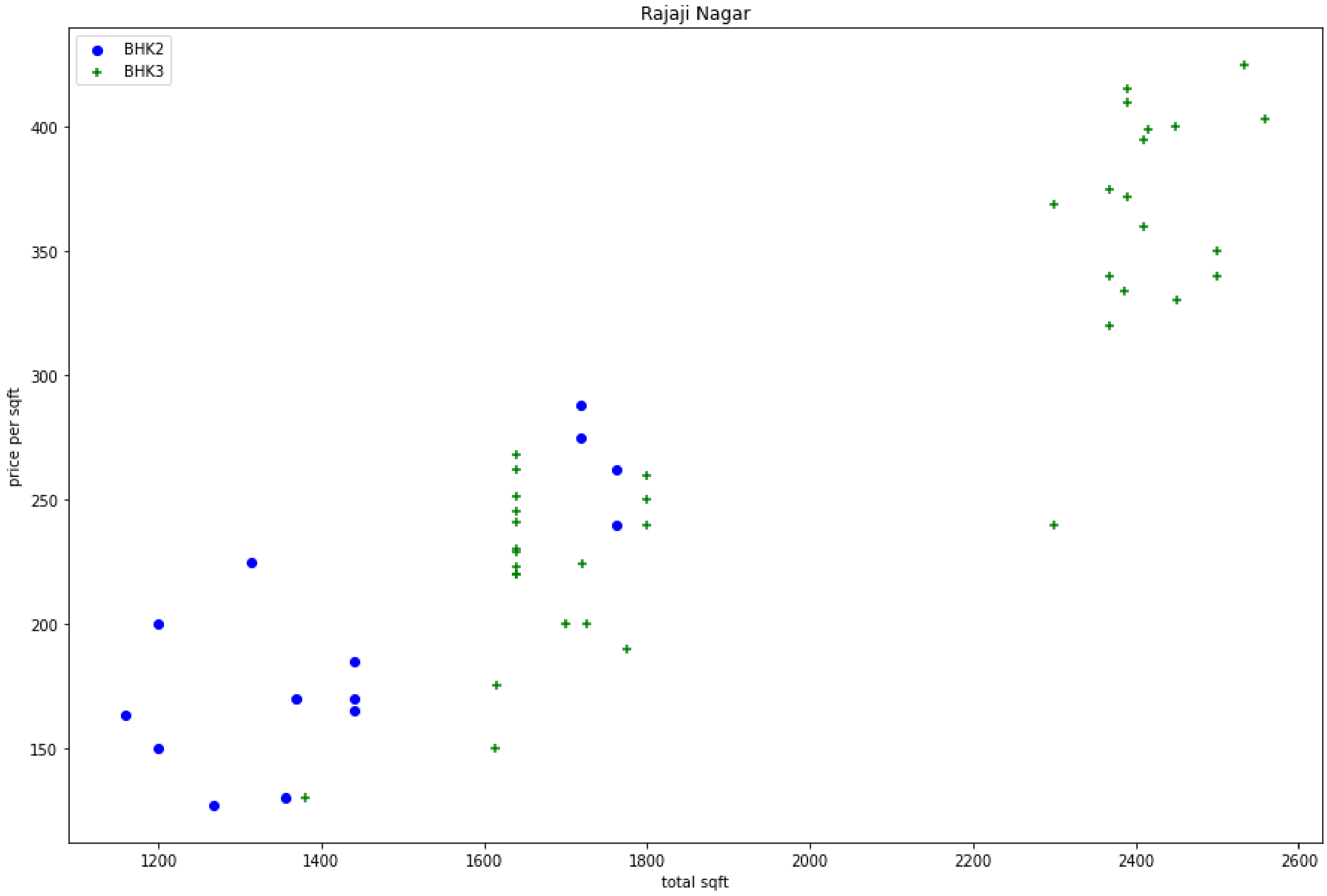
```
min         267.829813
25%        4210.526316
50%        5294.117647
75%        6916.666667
max      176470.588235
Name: price_per_sqft, dtype: float64
```

```
import numpy as np
def removeoutliers(df):
  dfout = pd.DataFrame()
  for key,subdf in df.groupby('location'):
    m = np.mean(subdf.price_per_sqft)
    st = np.std(subdf.price_per_sqft)
    reduceddf = subdf[(subdf.price_per_sqft>(m-st))&(subdf.price_per_sqft<=(m+st))]
    dfout = pd.concat([dfout,reduceddf],ignore_index=True)
  return dfout
```

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

```
(10242, 7)
```

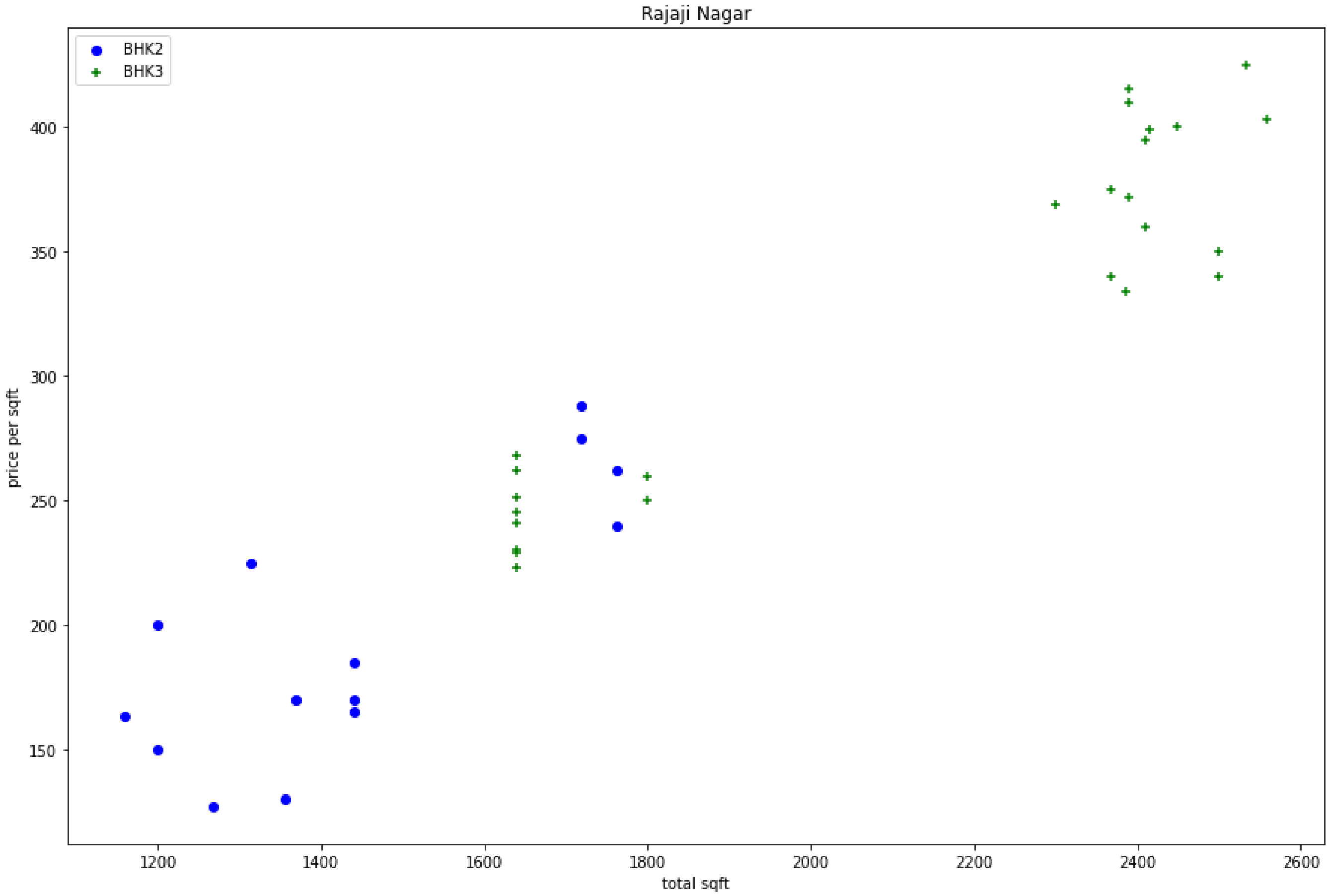
```
import matplotlib.pyplot as plt
def plotscatterplot(df,location):
  bhk2 = df[(df.location==location) & (df.bhk==2)]
  bhk3 = df[(df.location==location) & (df.bhk==3)]
  matplotlib.rcParams['figure.figsize']=(15,10)
  plt.scatter(bhk2.total_sqft,bhk2.price,color='blue',label='BHK2')
  plt.scatter(bhk3.total_sqft,bhk3.price,color='green',label='BHK3',marker='+')
  plt.xlabel("total sqft")
  plt.ylabel("price per sqft")
  plt.legend()
  plt.title(location)

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



```
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<(st
    return df.drop(exclude_indices,axis='index')
df8 = remove_bhk_outliers(df7)
df8.shape
```

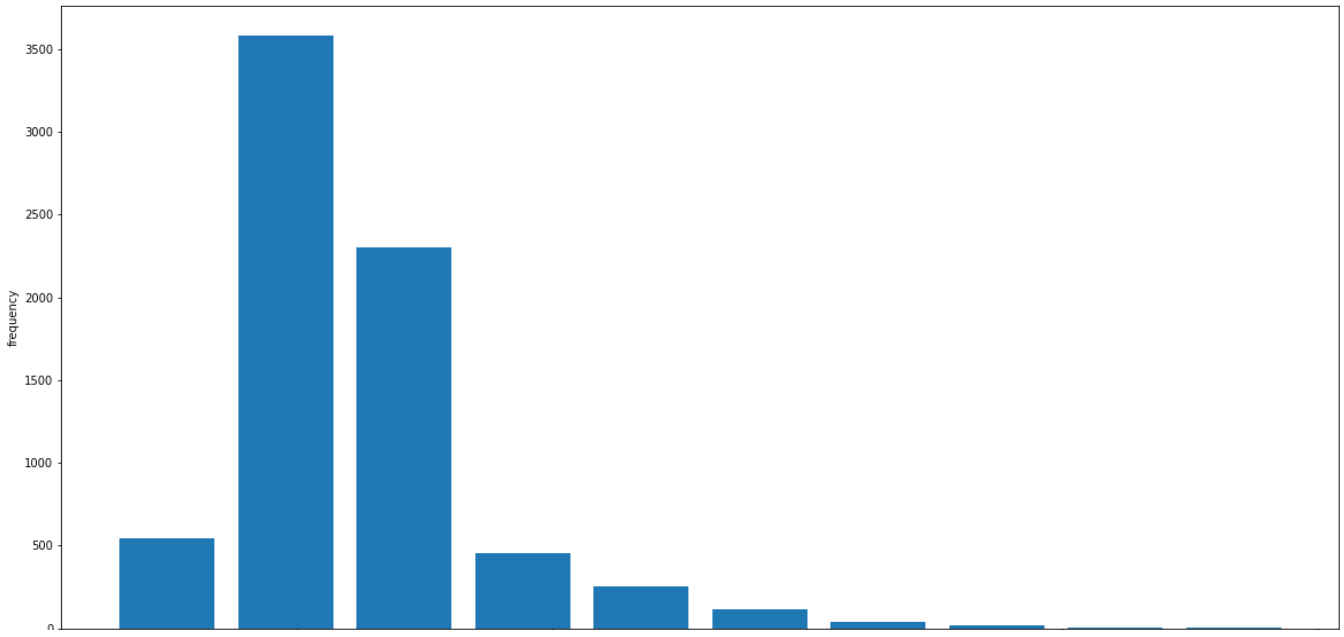
```
     (7317, 7)
```

```
plotscatterplot(df8,"Rajaji Nagar")
```



```
matplotlib.rcParams['figure.figsize'] = (20,10)
plt.hist(df8.price_per_sqft,rwidth=0.8)
plt.xlabel("price")
plt.ylabel("frequency")
```

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



```
df8.bath.unique()
```

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

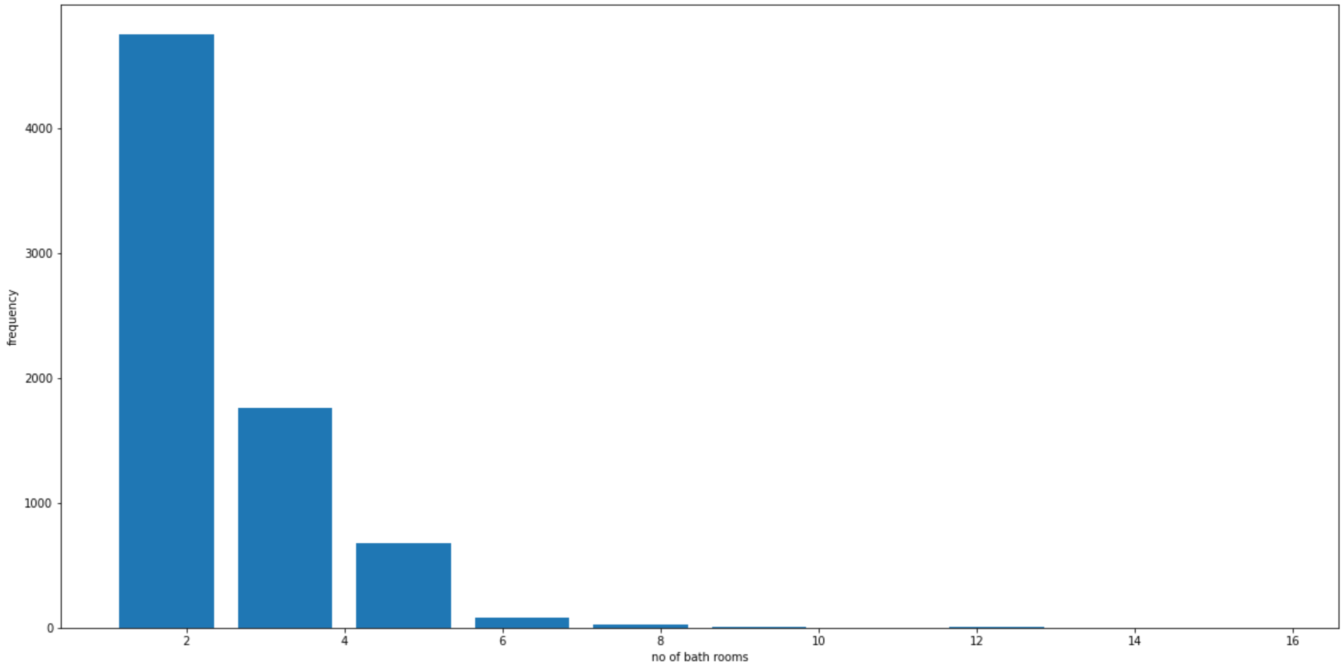
```
df8[df8.bath>4]
```

	location	size	total_sqft	bath	price	bhk	price_per_sqft
9	1st Phase JP Nagar	5 Bedroom	1500.0	5.0	85.0	5	5666.666667
36	2nd Stage Nagarbhavi	6 Bedroom	3000.0	8.0	451.0	6	15033.333333
37	2nd Stage Nagarbhavi	6 Bedroom	2400.0	8.0	450.0	6	18750.000000
45	5th Block Hbr Layout	5 Bedroom	3600.0	5.0	130.0	5	3611.111111
46	5th Block Hbr Layout	6 BHK	5100.0	5.0	300.0	6	5882.352941
...	...	...	...	...	...	...	...
10146	other	4 Bedroom	3100.0	5.0	425.0	4	13709.677419
10203	other	4 BHK	6652.0	6.0	660.0	4	9921.828022
10210	other	4 Bedroom	6688.0	6.0	700.0	4	10466.507177
10222	other	6 Bedroom	1800.0	5.0	140.0	6	7777.777778
10241	other	4 BHK	3600.0	5.0	400.0	4	11111.111111

297 rows × 7 columns

```
import matplotlib.pyplot as plt
plt.hist(df8.bath,rwidth=0.8)
plt.xlabel("no of bath rooms")
plt.ylabel("frequency")
```

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



```
df9 = df8[df8.bath<df8.bhk+2]
df10 = df9.drop(['size','price_per_sqft'],axis = 'columns')
df10
```

	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

```
dummies = pd.get_dummies(df10.location)
```

... ...

```
dummies
```

	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
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	1	0	0	0	0	0	0	0	0	0
4	1	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...
10233	0	0	0	0	0	0	0	0	0	0
10234	0	0	0	0	0	0	0	0	0	0
10237	0	0	0	0	0	0	0	0	0	0
10238	0	0	0	0	0	0	0	0	0	0
10241	0	0	0	0	0	0	0	0	0	0

7239 rows × 241 columns

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

```
df11
```

	location	total_sqft	bath	price	bhk	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi
0	1st Block Jayanagar	2850.0	4.0	428.0	4	1	0	0	0
1	1st Block Jayanagar	1630.0	3.0	194.0	3	1	0	0	0
2	1st Block Jayanagar	1875.0	2.0	235.0	3	1	0	0	0
3	1st Block Jayanagar	1200.0	2.0	130.0	3	1	0	0	0
4	1st Block Jayanagar	1235.0	2.0	148.0	2	1	0	0	0
...	...	...	...	...	...	...	...	...	...
10233	other	1200.0	2.0	70.0	2	0	0	0	0
10234	other	1800.0	1.0	200.0	1	0	0	0	0

```
df12 = df11.drop('location',axis=1)
df12
```

	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
0	2850.0	4.0	428.0	4	1	0	0	0	0	
1	1630.0	3.0	194.0	3	1	0	0	0	0	
2	1875.0	2.0	235.0	3	1	0	0	0	0	
3	1200.0	2.0	130.0	3	1	0	0	0	0	
4	1235.0	2.0	148.0	2	1	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	
10233	1200.0	2.0	70.0	2	0	0	0	0	0	
10234	1800.0	1.0	200.0	1	0	0	0	0	0	
10237	1353.0	2.0	110.0	2	0	0	0	0	0	
10238	812.0	1.0	26.0	1	0	0	0	0	0	
10241	3600.0	5.0	400.0	4	0	0	0	0	0	

7239 rows × 244 columns

```
X = df12.drop('price',axis=1)
```

```
X = df12.drop('price',axis=1)
X
```

	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
0	2850.0	4.0	4	1	0	0	0	0	0	
1	1630.0	3.0	3	1	0	0	0	0	0	
2	1875.0	2.0	3	1	0	0	0	0	0	
3	1200.0	2.0	3	1	0	0	0	0	0	
4	1235.0	2.0	2	1	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	
10233	1200.0	2.0	2	0	0	0	0	0	0	
10234	1800.0	1.0	1	0	0	0	0	0	0	
10237	1353.0	2.0	2	0	0	0	0	0	0	
10238	812.0	1.0	1	0	0	0	0	0	0	
10241	3600.0	5.0	4	0	0	0	0	0	0	

7239 rows × 243 columns

```
y = df12.price
y
```

```
0        428.0
1        194.0
2        235.0
3        130.0
4        148.0
         ...
10233     70.0
10234    200.0
10237    110.0
10238     26.0
10241    400.0
Name: price, Length: 7239, dtype: float64
```

```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest = train_test_split(X,y,test_size=0.2,random_state=10)
```

```
from sklearn.linear_model import LinearRegression
lr_clf = LinearRegression()
lr_clf.fit(xtrain,ytrain)
lr_clf.score(xtest,ytest)
```

```
0.8629132245229443
```

```
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([0.82702546, 0.86027005, 0.85322178, 0.8436466 , 0.85481502])
```

```
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	0.847796	{'normalize': False}
1	lasso	0.726751	{'alpha': 2, 'selection': 'random'}
2	decision_tree	0.715751	{'criterion': 'mse', 'splitter': 'best'}

```
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]
```

```
predict_price('2nd Phase Judicial Layout',1000,2,2)
```

```
28.47722993537434
```

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

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