Importing `neccessary libraries`

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as pyp
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler
```

Exploratory Data Analysis

Understanding the dataset

```
df = pd.read_csv('/home/srinivas/Downloads/datasets/csv datasets/Bengaluru_House_Data.csv')
df.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

```
df.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
```

We observed that many features have null values .so we have to handle them.
Also there are some features that are not useful for prediction.
For example if we observe there are some columns like `availability` ,`society`, `balcony` aren't useful

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

`area_type` is also not useful .Because in reality the peploe who are checking for the houses aren't payed much interest in the type of are .so we can neglet that feature.

```
df1.head()
```

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

checking for `null` values

2	Uttarahalli	3 BHK	1440	2.0	62.00

```
df1.isnull().sum()
```

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

```
df1.dropna(inplace = True)
```

`size` feature give us the information about the number of bedrooms.

```
df1['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', '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)
```

now we are creating a new feature called `bhk` nothing but bedrooms,because the `size` is categorical.We want numerical vlaues for predicting the house price

```
df1['bhk'] = df1['size'].apply(lambda x : int(x.split(' ')[0]))
```

```
df1.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

```
df1['bhk'].unique()
```

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

```
df1[df1['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

we can observe that `total_sqft` with 2400 contains `bhk` with 43 bedrooms.which is a wrong detail actually

```
df1['total_sqft'].unique()
```

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

if we observe that ,the `total_sqft` is **object** but it was to be **numerical**.Also the data format is not correct in `total_sqft` .First we correct them.

```
def is_float(x):
    try:
```

```
        float(x)
    except:
        return False
    return True
```

```
df1[~df1['total_sqft'].apply(lambda x: is_float(x))].value_counts()
```

```
location        size       total_sqft      bath  price    bhk
Whitefield      4 BHK      2830 - 2882     5.0   154.500  4      2
 Rachenahalli   1 RK       385 - 440       1.0   19.800   1      1
Pragathi Nagar  1 BHK      799 - 803       1.0   33.645   1      1
Rajaji Nagar    4 BHK      2563 - 2733     5.0   251.500  4      1
Rajapura        2 BHK      86.72Sq. Meter  2.0   40.000   2      1
                                                                ..
Hormavu         1 BHK      527 - 639       1.0   28.275   1      1
                2 BHK      943 - 1220      2.0   38.665   2      1
                3 BHK      1469 - 1766     3.0   73.595   3      1
Hosa Road       1 BHK      615 - 985       1.0   39.990   1      1
arudi           3 Bedroom  6Acres          2.0   80.000   3      1
Length: 189, dtype: int64
```

we can observe the data in different format.

```
df1.duplicated().value_counts()
```

```
False    12365
True       881
dtype: int64
```

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

`checking`

```
change('777')
```

```
777.0
```

```
df2 = df1.copy()
df2['total_sqft'] =df1['total_sqft'].apply(lambda x : change(x))
```

```
df2.head()
```

	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

we made an average value of data that is in the format `'799 - 803'`,and updated with the average value.

```
df3 = df2.copy()
df3['price_per_sq'] = df3['price']*100000/df3['total_sqft']
```

we created a new feature called `price_per_sq` for futher analysis.

```
df3.head()
```

	location	size	total_sqft	bath	price	bhk	price_per_sq
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

```
df3.location.value_counts()
```

```
Whitefield                                534
Sarjapur  Road                            392
Electronic City                           302
Kanakpura Road                            266
Thanisandra                               233
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
Hoodi                                      88
KR Puram                                   88
Electronics City Phase 1                   87
Yeshwanthpur                               85
Begur Road                                 84
Sarjapur                                   81
Harlur                                     79
Kasavanhalli                               79
Banashankari                               74
Hormavu                                    74
Kengeri                                    73
Ramamurthy Nagar                           73
Koramangala                                72
Hosa Road                                  72
Varthur                                    70
Old Madras Road                            70
Jakkur                                     68
JP Nagar                                   67
Kothanur                                   66
Kaggadasapura                              64
Nagarbhavi                                 63
Akshaya Nagar                              62
Thigalarapalya                             62
TC Palaya                                  60
8th Phase JP Nagar                         57
Malleshwaram                               57
Rachenahalli                               56
Budigere                                   54
HSR Layout                                 53
Hennur                                     52
Jigani                                     52
Jalahalli                                  52
Hulimavu                                   52
Bisuvanahalli                              51
Panathur                                   51
Ramagondanahalli                           50
Hegde Nagar                                49
Bhoganhalli                                48
Mysore Road                                48
Gottigere                                  48
```

```
df3['location'].nunique()
```

```
1304
```

we have so many locations whose value_count is less than 10.so we are converting those locations to `other`. Because those locatoins are not famous while predicting the house price also they likely provide same prediction.

```
pd.options.display.max_rows = 9999
locations = df3['location'].value_counts()
```

```
filt = locations.values<10
less_ten = locations[filt]
```

```
df4 = df3.copy()
df4['location'] = df4['location'].apply(lambda x : 'other' if x in less_ten else x)
```

```
df4['location'] = df4['location'].apply(lambda x : x.strip())
df4['location'].nunique()
```

```
254
```

we observed that the unique columns reduced from `1304` to `254`

```
df4.location.value_counts()
```

```
other                       2776
Whitefield                   534
Sarjapur  Road               392
Electronic City              302
Kanakpura Road               266
Thanisandra                  233
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
KR Puram                      88
Hoodi                         88
Electronics City Phase 1      87
Yeshwanthpur                  85
Begur Road                    84
Sarjapur                      81
Harlur                        79
Kasavanhalli                  79
Hormavu                       74
Banashankari                  74
Ramamurthy Nagar              73
Kengeri                       73
Koramangala                   72
Hosa Road                     72
Varthur                       70
Old Madras Road               70
Jakkur                        68
JP Nagar                      67
Kothanur                      66
Kaggadasapura                 64
Nagarbhavi                    63
Akshaya Nagar                 62
Thigalarapalya                62
TC Palaya                     60
8th Phase JP Nagar            57
Malleshwaram                  57
Rachenahalli                  56
Budigere                      54
HSR Layout                    53
Hennur                        52
Jalahalli                     52
Hulimavu                      52
Jigani                        52
Panathur                      51
Bisuvanahalli                 51
Ramagondanahalli              50
Hegde Nagar                   49
Gottigere                     48
Bhoganhalli                   48
```

Now we are removing the outliers.

Assume that the total_sqft per bebroom shoul be more than 300 sq_ft.

so remove the records whose total_sqft/bedroom is < 300

```
df4 = df4[~(df4['total_sqft']/df3['bhk']<300)]
```

```
df4.shape[0]
```

```
12502
```

```
data = df4.groupby('location')
```

Now we are finding the outliers in each location and removing those records whose `price_per_sq` is outlier

```
def outlier_removal(dframe):
    out_df = pd.DataFrame()
    for key,subdf in dframe.groupby('location'):
        mean = subdf['price_per_sq'].mean()
        std = subdf['price_per_sq'].std()
        outdf = subdf[(subdf['price_per_sq']>(mean-std)) & (subdf['price_per_sq']<=(mean+std)
        out_df = pd.concat([outdf,out_df],ignore_index = True)
    return out_df
```

```
df5 = outlier_removal(df4)
df5.shape
```

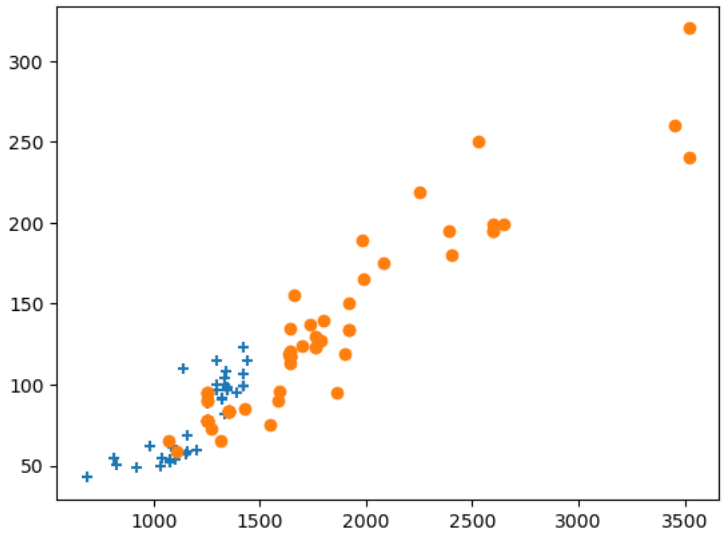
```
(10280, 7)
```

```
df4['location'].nunique()
```

```
254
```

```
def plott(df,location):
    df1 = df[(df['location']==location) & (df['bhk']==2)]
    df2 = df[(df['location']==location)& (df['bhk']==3)]
    pyp.scatter(df1['total_sqft'],df1['price'],marker = '+')
    pyp.scatter(df2['total_sqft'],df2['price'])
    pyp.show()
```

```
plott(df5,'Hebbal')
```



If we observe that,for a same location `Hebbal` and some same `total_sqft`,the price of the house with `2` bedrooms is higher than `3` bedrooms.This is actually not correct,so we consider those too as outliers and remove those records.

```
df5.location
```

```
0                    other
1                    other
2                    other
3                    other
4                    other
                ...
10275    1st Block Jayanagar
10276    1st Block Jayanagar
```

```
10277    1st Block Jayanagar
10278    1st Block Jayanagar
10279    1st Block Jayanagar
Name: location, Length: 10280, dtype: object
```

we are going to remove the rows for the same area ,where 3 bed rooms costs is less than 2 bed rooms cost.

This is done by simply removing the 3 bed room apartments for the same area where price is less than the mean of the 2 bed room apartment.

```
data = df4.groupby('location')
```

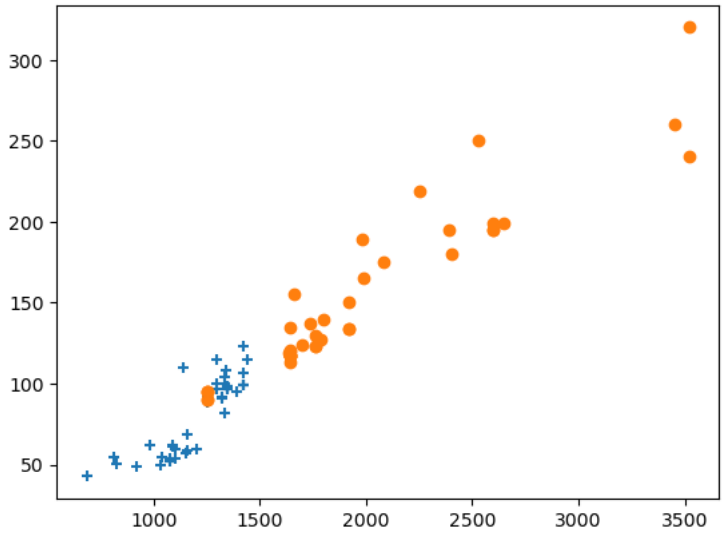
```
def remove_outliers(df):
    remove_index = np.array([])
    for ind,ind_df in df.groupby('location'):
        dic = {}
        for bhk,bhk_df in ind_df.groupby('bhk'):
            dic[bhk]={
                'mean':np.mean(bhk_df.price_per_sq),
                'std':np.std(bhk_df.price_per_sq),
                'count':bhk_df.shape[0]
            }
        for bhk,bhk_df in ind_df.groupby('bhk'):
            stats = dic.get(bhk-1)
            if stats and stats['count']>5:
                remove_index = np.append(remove_index,bhk_df[bhk_df['price_per_sq']<stats['me
    return df.drop(remove_index,axis='rows')
```

```
df6 = remove_outliers(df5)
```

```
df6.shape
```

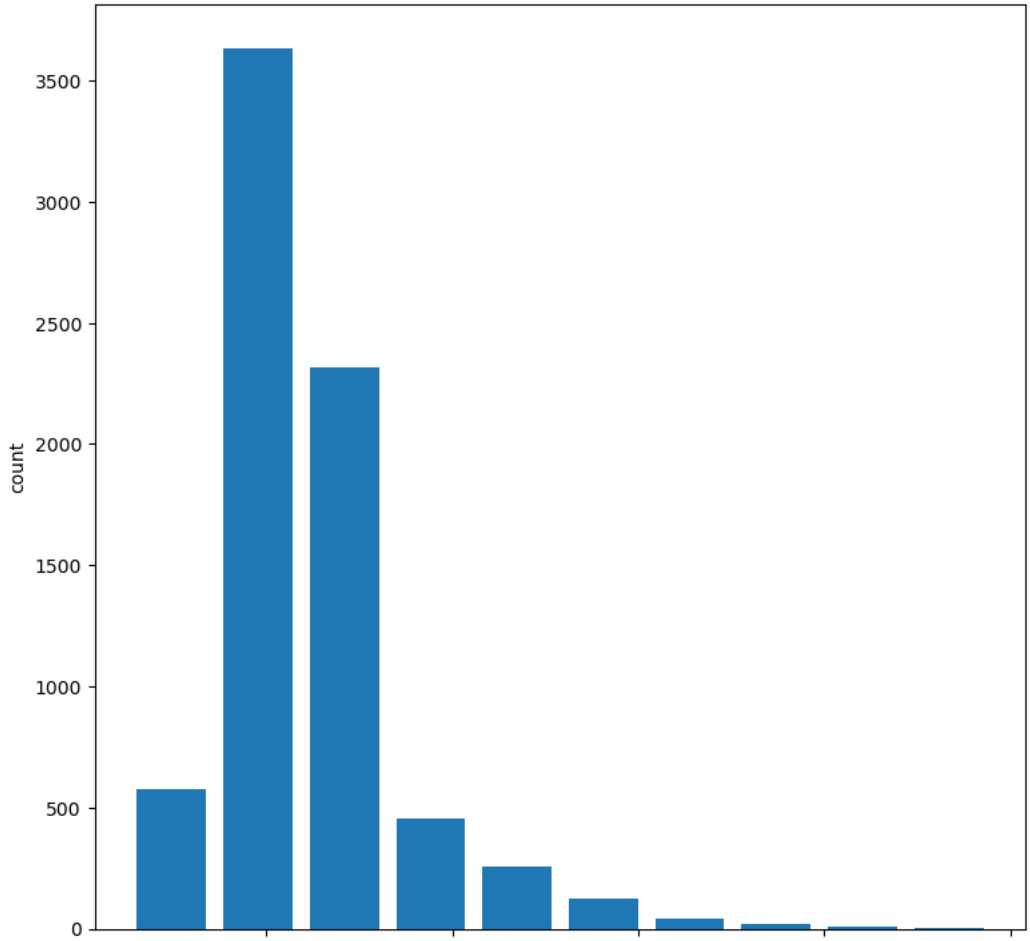
```
(7430, 7)
```

```
plott(df6,'Hebbal')
```



we can observe that most of the outliers are removed .In the plot,we can see that, for the location `Hebbal` and for the square_ft(around `1500`) the houses with 3 bedrooms are removed.Like that ,it is done in all locations.

```
f,ax = pyp.subplots(figsize= (9,9))
ax.hist(df6.price_per_sq,rwidth=0.8)
ax.set_xlabel('price_per_square_feet')
ax.set_ylabel('count')
pyp.show()
```



we observe that , the price_per_square_feet follows a `normal distribution` and the average price_per_square_feet is around `5000`.

```
df6.bath.unique()
```

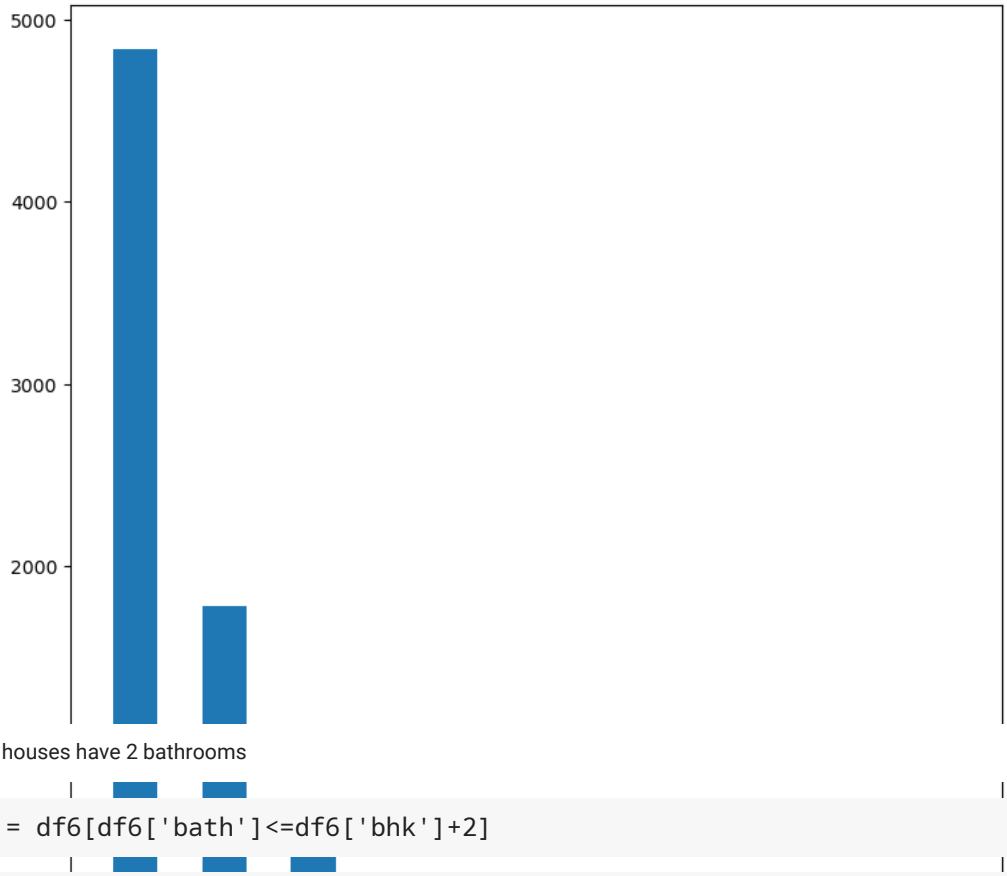
```
array([ 4.,  1.,  2.,  3.,  8.,  9.,  5.,  6., 12., 16.,  7., 13., 14.])
```

```
df6[df6.bath>10]
```

	location	size	total_sqft	bath	price	bhk	price_per_sq
533	other	10 BHK	12000.0	12.0	525.0	10	4375.000000
619	other	16 BHK	10000.0	16.0	550.0	16	5500.000000
1323	other	11 BHK	6000.0	12.0	150.0	11	2500.000000
1639	other	13 BHK	5425.0	13.0	275.0	13	5069.124424
4907	Neeladri Nagar	10 BHK	4000.0	12.0	160.0	10	4000.000000
9690	BTM 1st Stage	9 Bedroom	3300.0	14.0	500.0	9	15151.515152

if you observe the bathrooms.some houses has extra bathrooms than bedrooms.like in `BTM 1st Stage` ,it has `14 bathrooms` and `9 bedrooms`.
it's common of having more bathrooms ,but it should have limit.so we are removing by taking condition of `bathrooms are not bedrooms+2`

```
f,ax = pyp.subplots(figsize=(9,9))
ax.hist(df6['bath'],rwidth=0.5)
pyp.show()
```



most houses have 2 bathrooms

```
df7 = df6[df6['bath']<=df6['bhk']+2]
```

```
df7.shape
```

```
(7425, 7)
```

now the dataset is `preprcessed and cleaned`.

```
df7.head()
```

	location	size	total_sqft	bath	price	bhk	price_per_sq
0	other	3 BHK	2770.0	4.0	290.00	3	10469.314079
2	other	1 RK	510.0	1.0	25.25	1	4950.980392
6	other	2 BHK	1500.0	2.0	185.00	2	12333.333333
7	other	2 BHK	840.0	2.0	45.00	2	5357.142857
8	other	3 Bedroom	4395.0	3.0	240.00	3	5460.750853

We don't want `size` and `price_per_sq` features.

```
df8 = df7.drop(['size','price_per_sq'],axis='columns')
```

```
df8.head()
```

	location	total_sqft	bath	price	bhk
0	other	2770.0	4.0	290.00	3
2	other	510.0	1.0	25.25	1
6	other	1500.0	2.0	185.00	2
7	other	840.0	2.0	45.00	2
8	other	4395.0	3.0	240.00	3

changing categorical values of the location (applying one hot encoding)

```
dummies = pd.get_dummies(df7['location'])
df8 = pd.concat([df8,dummies],axis='columns')
```

```
df8.drop('other',axis='columns',inplace=True)
```

Since I don't want the `other` column while in deployment phase .I can remove that feature

```
df8.head()
```

	location	total_sqft	bath	price	bhk	1st Block Jayanagar	1st Block Koramangala	1st Phase JP Nagar	2 Pha Judici Layo
0	other	2770.0	4.0	290.00	3	0	0	0	
2	other	510.0	1.0	25.25	1	0	0	0	
6	other	1500.0	2.0	185.00	2	0	0	0	
7	other	840.0	2.0	45.00	2	0	0	0	
8	other	4395.0	3.0	240.00	3	0	0	0	

5 rows × 511 columns

Model Building

```
x = df8.drop(['price','location'],axis='columns')
y = df8['price']
```

```
x.head()
```

	total_sqft	bath	bhk	1st Block Jayanagar	1st Block Koramangala	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	l
0	2770.0	4.0	3	0	0	0	0	0	
2	510.0	1.0	1	0	0	0	0	0	
6	1500.0	2.0	2	0	0	0	0	0	
7	840.0	2.0	2	0	0	0	0	0	
8	4395.0	3.0	3	0	0	0	0	0	

5 rows × 509 columns

```
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=4)
```

```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train,y_train)
lr.score(x_test,y_test)
```

```
0.8223541798087829
```

```
from sklearn.model_selection import ShuffleSplit
cv = ShuffleSplit(n_splits=5,test_size=0.2,random_state=4)
cross_val_score(LinearRegression(),x,y,cv=cv)
```

```
array([0.82235418, 0.82601456, 0.84809415, 0.86283934, 0.8740958 ])
```

These are the `score` and `cross_val_scores` I achieved by using `Linear_regression` technique.
But I want to check other models also for obtaining best model to handle the given data.

```
from sklearn.linear_model import Lasso
from sklearn.model_selection import RandomizedSearchCV
def algo(x,y):
    score=[]
    dic ={
        'linearregression':{
            'model':LinearRegression(),
            'params':{
                'fit_intercept':[True,False]
            }
        },
        'lasso':{
            'model':Lasso(),
            'params':{
                'alpha':[1,2],
                'selection':['random','cyclic']
            }
        }
    }
    cv = ShuffleSplit(n_splits=5,test_size=5,random_state=4)
    for name,models in dic.items():
        model = RandomizedSearchCV(models['model'],models['params'],cv=cv,return_train_score=
        model.fit(x,y)
        score.append({
            'model':name,
            'score':model.best_score_,
            'params':model.best_params_
        })
    return pd.DataFrame(score)
model = algo(x,y)
```

I created a function and used the technique `RandomizedSearchCV` to find the best model with best pareameters.

```
model
```

	model	score	params
0	linearregression	0.860003	{'fit_intercept': True}
1	lasso	0.638810	{'selection': 'cyclic', 'alpha': 1}

From the data we can understand that **linear regression** fits better for above dataset
If we want to `deploy` our model,first we have to save the model.
For that purpose I used the `pickle` module to save my model

```
def predict_vals(sqft,bath,bhk,location):
    loc = np.where(x.columns==location)[0][0]
    a=np.zeros(len(x.columns))
    a[0]=sqft
    a[1]=bath
    a[2]=bhk
    if loc>=0:
        a[loc]=1
    return lr.predict([a])
```

```
predict_vals(1000,2,2,'Indira Nagar')[0]
```

```
/usr/lib/python3/dist-packages/sklearn/base.py:420: UserWarning: X does not have valid feature names, but LinearRegressi
  warnings.warn(
187.85636257528432
```

we predicted the price of the house located at 'Indira Nagar' with 1000 square_feet ,2 bathrooms and 2 bedrooms

```
import pickle
```

```
with open('bengaluru_dataset.pickle','wb') as f:
    pickle.dump(lr,f)
```

saving the model using the `pickle`

```
with open('bengaluru_dataset.pickle','rb') as f:
    model = pickle.load(f)
```

Opened the saved model and predicting another price value.

```
def predict_vals(sqft,bath,bhk,location):
    loc = np.where(x.columns==location)[0][0]
    a=np.zeros(len(x.columns))
    a[0]=sqft
    a[1]=bath
    a[2]=bhk
    if loc>=0:
        a[loc]=1
    return model.predict([a])
```

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

```
/usr/lib/python3/dist-packages/sklearn/base.py:420: UserWarning: X does not have valid feature names, but LinearRegressi
  warnings.warn(
array([187.85636258])
```