In [247…
```
import pandas as pd
import numpy as np
```

In [248…
```
house=pd.read_csv('Banglore_Housing_Prices.csv')
house
```

Out[248]:

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

13320 rows × 5 columns

In [249…
```
null_val=house.isnull().sum()
null_val
```

Out[249]:
```
location       1
size          16
total_sqft     0
bath          73
price          0
dtype: int64
```

In [250…
```
house['location'].fillna(method='ffill',inplace=True)
```

In [251…
```
house.dropna(subset='size',inplace=True)
house.dropna(subset='bath',inplace=True)
```

In [252…
```
null_val=house.isnull().sum()
null_val
```

Out[252]:
```
location      0
size          0
total_sqft    0
bath          0
price         0
dtype: int64
```

In [253…
```
house['size'].unique()
```

Out[253]:
```
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 [254…
```
house['size'].replace(['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'],[2,4,3,4,6,3,1,1,1,8,2,7,5,7,6,5,11,9,9,27,10,11
```

In [255…
```
house['total_sqft'].unique()
```

Out[255]:
```
array(['1056', '2600', '1440', ..., '1133 - 1384', '774', '4689'],
      dtype=object)
```

In [256…
```
def convert_sqft_into_number(x):
    token = x.split('-')
    if len(token) == 2:
        return (float(token[0]) + float(token[1])) / 2
    try:
        return float(x)
    except:
        return None
```

In [257…
```
house1 = house.copy()

house1['total_sqft'] = house1['total_sqft'].apply(convert_sqft_into_number)
```

In [258…
```
house=house1
```

In [259…
```
house['price_per_sqft'] = house['price']*100000 / house['total_sqft']
house.head()
```

Out[259]:

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

In [260…
```
house.price_per_sqft.describe()
```

Out[260]:
```
count    1.320100e+04
mean     7.920566e+03
std      1.067231e+05
min      2.678298e+02
25%      4.267782e+03
50%      5.438066e+03
75%      7.317073e+03
max      1.200000e+07
Name: price_per_sqft, dtype: float64
```

In [261…
```
house.describe()
```

Out[261]:

| | size | total_sqft | bath | price | price_per_sqft |
|---|---|---|---|---|---|
| **count** | 13247.000000 | 13201.000000 | 13247.000000 | 13247.000000 | 1.320100e+04 |
| **mean** | 2.801917 | 1555.306169 | 2.692610 | 112.387400 | 7.920566e+03 |
| **std** | 1.295710 | 1237.276637 | 1.341458 | 149.071136 | 1.067231e+05 |
| **min** | 1.000000 | 1.000000 | 1.000000 | 8.000000 | 2.678298e+02 |
| **25%** | 2.000000 | 1100.000000 | 2.000000 | 50.000000 | 4.267782e+03 |
| **50%** | 3.000000 | 1275.000000 | 2.000000 | 72.000000 | 5.438066e+03 |
| **75%** | 3.000000 | 1672.000000 | 3.000000 | 120.000000 | 7.317073e+03 |
| **max** | 43.000000 | 52272.000000 | 40.000000 | 3600.000000 | 1.200000e+07 |

In [262…

```
house2=house[~(house['total_sqft']/house['size']<300)]
```

In [263…

```
def remove_outliers(df):
    df_output=pd.DataFrame()
    for key,subdf in df.groupby('location'):
        m=np.mean(subdf.price_per_sqft)
        st=np.std(subdf.price_per_sqft)
        gen_df=subdf[(subdf.price_per_sqft > (m-st))&(subdf.price_per_sqft<=(m+st))]
        df_output=pd.concat([df_output,gen_df],ignore_index=True)
    return df_output
house3=remove_outliers(house2)
house3.describe()
```

Out[263]:

| | size | total_sqft | bath | price | price_per_sqft |
|---|---|---|---|---|---|
| **count** | 9260.000000 | 9260.000000 | 9260.000000 | 9260.000000 | 9260.000000 |
| **mean** | 2.557883 | 1504.737585 | 2.464255 | 94.158715 | 5724.681625 |
| **std** | 0.846329 | 893.377483 | 0.951529 | 110.655686 | 2536.074109 |
| **min** | 1.000000 | 300.000000 | 1.000000 | 10.000000 | 1250.000000 |
| **25%** | 2.000000 | 1110.000000 | 2.000000 | 49.000000 | 4258.695469 |
| **50%** | 2.000000 | 1283.000000 | 2.000000 | 67.000000 | 5185.251646 |
| **75%** | 3.000000 | 1650.000000 | 3.000000 | 100.000000 | 6404.402624 |
| **max** | 10.000000 | 30400.000000 | 14.000000 | 2912.000000 | 35000.000000 |

In [264…

```
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('size'):
            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('size'):
            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')
house4 = remove_bhk_outliers(house3)
house4.shape
```

Out[264]: (7509, 6)

In [265…]
```
house4
```

Out[265]:

| | location | size | total_sqft | bath | price | price_per_sqft |
|---|---|---|---|---|---|---|
| **0** | Devarabeesana Halli | 3 | 1672.0 | 3.0 | 150.00 | 8971.291866 |
| **1** | Devarabeesana Halli | 3 | 1750.0 | 3.0 | 149.00 | 8514.285714 |
| **2** | Devarabeesana Halli | 3 | 1750.0 | 3.0 | 150.00 | 8571.428571 |
| **4** | Devarachikkanahalli | 2 | 1250.0 | 2.0 | 40.00 | 3200.000000 |
| **5** | Devarachikkanahalli | 2 | 1200.0 | 2.0 | 83.00 | 6916.666667 |
| **...** | ... | ... | ... | ... | ... | ... |
| **9255** | frazertown | 3 | 2900.0 | 3.0 | 325.00 | 11206.896552 |
| **9256** | manyata park | 3 | 1780.0 | 3.0 | 84.83 | 4765.730337 |
| **9257** | tc.palya | 2 | 880.0 | 2.0 | 48.00 | 5454.545455 |
| **9258** | tc.palya | 2 | 1000.0 | 2.0 | 55.00 | 5500.000000 |
| **9259** | tc.palya | 3 | 1400.0 | 2.0 | 78.00 | 5571.428571 |

7509 rows × 6 columns

In [266…]
```
house5=house4
```

In [267…]
```
from sklearn.preprocessing import LabelEncoder
loc=['location']
le = LabelEncoder()
house5[loc] = house5[loc].apply(le.fit_transform)
house5
```

Out[267]:

| | location | size | total_sqft | bath | price | price_per_sqft |
|---|---|---|---|---|---|---|
| **0** | 0 | 3 | 1672.0 | 3.0 | 150.00 | 8971.291866 |
| **1** | 0 | 3 | 1750.0 | 3.0 | 149.00 | 8514.285714 |
| **2** | 0 | 3 | 1750.0 | 3.0 | 150.00 | 8571.428571 |
| **4** | 1 | 2 | 1250.0 | 2.0 | 40.00 | 3200.000000 |
| **5** | 1 | 2 | 1200.0 | 2.0 | 83.00 | 6916.666667 |
| **...** | ... | ... | ... | ... | ... | ... |
| **9255** | 762 | 3 | 2900.0 | 3.0 | 325.00 | 11206.896552 |
| **9256** | 763 | 3 | 1780.0 | 3.0 | 84.83 | 4765.730337 |
| **9257** | 764 | 2 | 880.0 | 2.0 | 48.00 | 5454.545455 |
| **9258** | 764 | 2 | 1000.0 | 2.0 | 55.00 | 5500.000000 |
| **9259** | 764 | 3 | 1400.0 | 2.0 | 78.00 | 5571.428571 |

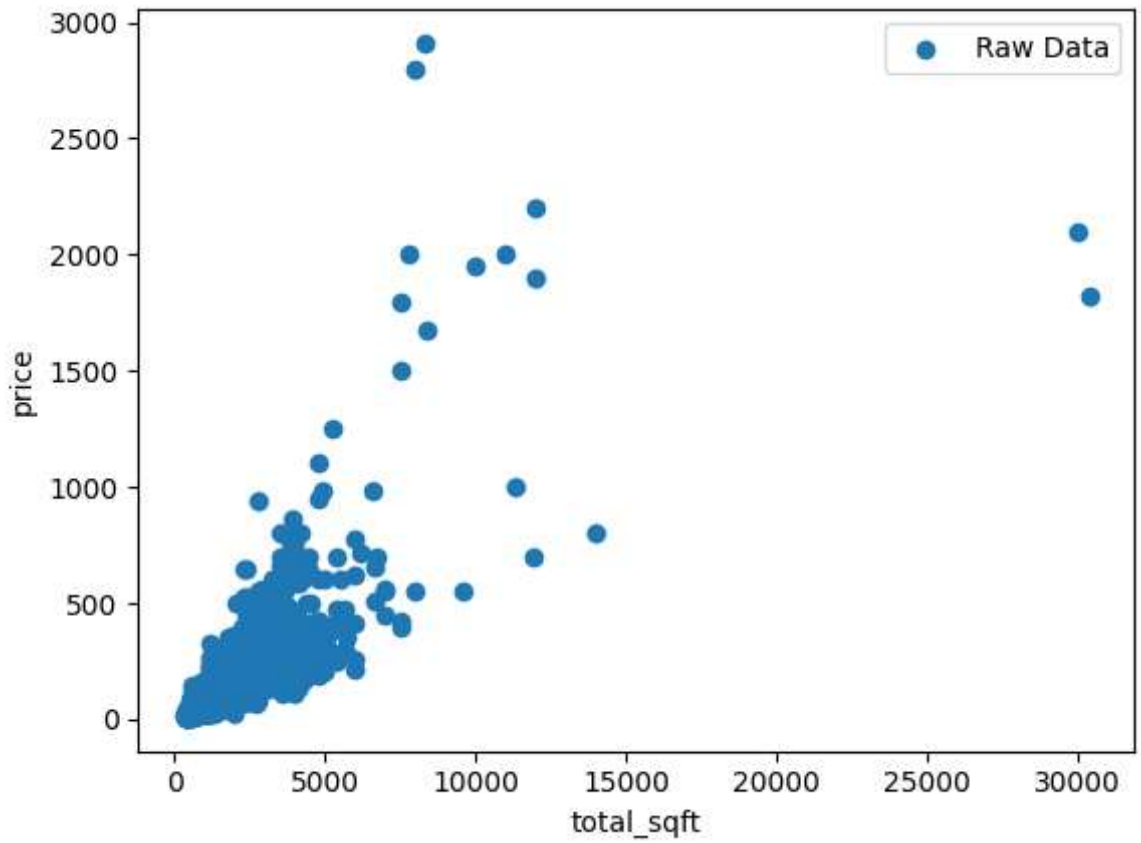
7509 rows × 6 columns

In [268…]
```
cleaned_data=house5
```

In [269…]
```
import matplotlib.pyplot as plt
total_sqft = cleaned_data['total_sqft']
price= cleaned_data['price']
plt.scatter(total_sqft,price, label='Raw Data')
plt.xlabel('total_sqft')
```

```
plt.ylabel('price')
plt.legend()
```

Out[269]: <matplotlib.legend.Legend at 0x1e9f66e6c10>




In [270…
```
house5.head()
```

Out[270]:

| | location | size | total_sqft | bath | price | price_per_sqft |
|---|---|---|---|---|---|---|
| **0** | 0 | 3 | 1672.0 | 3.0 | 150.0 | 8971.291866 |
| **1** | 0 | 3 | 1750.0 | 3.0 | 149.0 | 8514.285714 |
| **2** | 0 | 3 | 1750.0 | 3.0 | 150.0 | 8571.428571 |
| **4** | 1 | 2 | 1250.0 | 2.0 | 40.0 | 3200.000000 |
| **5** | 1 | 2 | 1200.0 | 2.0 | 83.0 | 6916.666667 |

In [271…
```
x=house5.drop(['price','bath'],axis=1)
y=house5['price']
```

In [272…
```
x.shape
```

Out[272]: (7509, 4)

In [273…
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x,y,test_size=0.3,random_state=101)
```

In [274…
```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

Out[274]: ((5256, 4), (2253, 4), (5256,), (2253,))

In [284…
```
y_test.describe()
```

Out[284]:
```
count    2253.000000
mean       96.376460
std       116.104378
min        10.000000
25%        50.000000
50%        69.000000
75%       104.000000
max      2912.000000
Name: price, dtype: float64
```

In [275…
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(X_train,y_train)
lr.score(X_test,y_test)
```

Out[275]:
```
0.8332026111275647
```

In [276…
```
pred = lr.predict(X_test)
pred
```

Out[276]:
```
array([181.2688959 , 122.46264329,  73.87822944, ...,  31.3580742 ,
        72.23505505, 322.91396007])
```

In [279…
```
lr.predict([[200,5,3000,9000]])
```

```
C:\Users\Chaitanya\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\base.py:
420: UserWarning: X does not have valid feature names, but LinearRegression was fitted with f
eature names
  warnings.warn(
```

Out[279]:
```
array([264.18370071])
```

In [285…
```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

In [286…
```
print("Mean Absolute Error:", mean_absolute_error(y_test, pred))
print("Mean Squared Error:", mean_squared_error(y_test, pred))
print("R-squared:", r2_score(y_test, pred))
```

```
Mean Absolute Error: 16.849862382246624
Mean Squared Error: 2247.468619112665
R-squared: 0.8332026111275647
```

In [ ]:
```
```