In [1]:

```
import pandas as pd
import numpy as np
```

In [2]:

```
data = pd.read_csv("Bengaluru_House_Data.csv")
```

In [3]:

```
data.head()
```

Out[3]:

	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.shape
```

Out[4]:

```
(13320, 9)
```

In [5]:

```
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
```

In [6]:

```
for column in data.columns:
    print(data[column].value_counts())
    print("*"*20)
```

```
Super built-up  Area    8790
Built-up  Area          2418
Plot  Area              2025
Carpet  Area              87
Name: area_type, dtype: int64
********************
Ready To Move    10581
18-Dec             307
18-May             295
18-Apr             271
18-Aug             200
                 ...  
15-Aug               1
17-Jan               1
16-Nov               1
16-Jan               1
14-Jul               1
Name: availability, Length: 81, dtype: int64
********************
Whitefield                        540
Sarjapur  Road                    399
Electronic City                   302
Kanakpura Road                    273
Thanisandra                       234
                                 ... 
Bapuji Layout                       1
1st Stage Radha Krishna Layout      1
BEML Layout 5th stage               1
singapura paradise                  1
Abshot Layout                       1
Name: location, Length: 1305, dtype: int64
********************
2 BHK         5199
3 BHK         4310
4 Bedroom      826
4 BHK          591
3 Bedroom      547
1 BHK          538
2 Bedroom      329
5 Bedroom      297
6 Bedroom      191
```

```
1 Bedroom      105
8 Bedroom       84
7 Bedroom       83
5 BHK           59
9 Bedroom       46
6 BHK           30
7 BHK           17
1 RK            13
10 Bedroom      12
9 BHK            8
8 BHK            5
11 BHK           2
11 Bedroom       2
10 BHK           2
14 BHK           1
13 BHK           1
12 Bedroom       1
27 BHK           1
43 Bedroom       1
16 BHK           1
19 BHK           1
18 Bedroom       1
Name: size, dtype: int64
********************
GrrvaGr    80
PrarePa    76
Sryalan    59
Prtates    59
GMown E    56
           ..
Amionce     1
JaghtDe     1
Jauraht     1
Brity U     1
RSntsAp     1
Name: society, Length: 2688, dtype: int64
********************
1200    843
1100    221
1500    205
2400    196
600     180
       ...
3580      1
2461      1
1437      1
2155      1
4689      1
Name: total_sqft, Length: 2117, dtype: int64
********************
2.0     6908
3.0     3286
4.0     1226
1.0      788
5.0      524
6.0      273
7.0      102
8.0       64
9.0       43
10.0      13
12.0       7
13.0       3
11.0       3
16.0       2
27.0       1
40.0       1
15.0       1
14.0       1
18.0       1
Name: bath, dtype: int64
********************
2.0    5113
1.0    4897
3.0    1672
0.0    1029
Name: balcony, dtype: int64
********************
75.00     310
65.00     302
55.00     275
60.00     270
45.00     240
         ...
351.00      1
54.10       1
80.64       1
32.73       1
488.00      1
Name: price, Length: 1994, dtype: int64
```

```
********************
```

In [7]:

```
data.isna().sum()
```

Out[7]:

```
area_type          0
availability       0
location           1
size              16
society         5502
total_sqft         0
bath              73
balcony          609
price              0
dtype: int64
```

In [8]:

```
data.drop(columns = ['area_type','availability','society','balcony'],inplace=True)
```

In [9]:

```
data.describe()
```

Out[9]:

	bath	price
count	13247.000000	13320.000000
mean	2.692610	112.565627
std	1.341458	148.971674
min	1.000000	8.000000
25%	2.000000	50.000000
50%	2.000000	72.000000
75%	3.000000	120.000000
max	40.000000	3600.000000

In [10]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13320 entries, 0 to 13319
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   location    13319 non-null  object 
 1   size        13304 non-null  object 
 2   total_sqft  13320 non-null  object 
 3   bath        13247 non-null  float64
 4   price       13320 non-null  float64
dtypes: float64(2), object(3)
memory usage: 520.4+ KB
```

In [11]:

```
data["location"].value_counts()
```

Out[11]:

```
Whitefield                          540
Sarjapur  Road                      399
Electronic City                     302
Kanakpura Road                      273
Thanisandra                         234
                                   ... 
Bapuji Layout                         1
1st Stage Radha Krishna Layout        1
BEML Layout 5th stage                 1
singapura paradise                    1
Abshot Layout                         1
Name: location, Length: 1305, dtype: int64
```

In [12]:

```
data['location'] = data['location'].fillna("Whitefield")
```

In [13]:

```
data['size'].value_counts()  #bedrooms, a hall, and a kitchen(BHK)
```

Out[13]:

```
2 BHK         5199
3 BHK         4310
4 Bedroom      826
4 BHK          591
3 Bedroom      547
1 BHK          538
2 Bedroom      329
5 Bedroom      297
6 Bedroom      191
1 Bedroom      105
8 Bedroom       84
7 Bedroom       83
5 BHK           59
9 Bedroom       46
6 BHK           30
7 BHK           17
1 RK            13
10 Bedroom      12
9 BHK            8
8 BHK            5
11 BHK           2
11 Bedroom       2
10 BHK           2
14 BHK           1
13 BHK           1
12 Bedroom       1
27 BHK           1
43 Bedroom       1
16 BHK           1
19 BHK           1
18 Bedroom       1
Name: size, dtype: int64
```

In [14]:

```
data['size'] = data['size'].fillna('2 BHK')
```

In [15]:

```
data['bath'] = data['bath'].fillna(data['bath'].median())
```

In [16]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13320 entries, 0 to 13319
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   location    13320 non-null  object 
 1   size        13320 non-null  object 
 2   total_sqft  13320 non-null  object 
 3   bath        13320 non-null  float64
 4   price       13320 non-null  float64
dtypes: float64(2), object(3)
memory usage: 520.4+ KB
```

In [17]:

```
data['bhk'] = data['size'].str.split().str.get(0).astype(int)
```

In [18]:

```
data[data.bhk > 20]
```

Out[18]:

	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 [19]:

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

Out[19]:

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

In [20]:

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

In [21]:

```
data['total_sqft'] = data['total_sqft'].apply(convertRange)
```

In [22]:

```
data.head()
```

Out[22]:

	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

Price Per Square Feet

In [23]:

```
data['price_per_sqft'] = data['price'] * 100000 / data['total_sqft']
```

In [24]:

```
data['price_per_sqft']
```

Out[24]:

```
0         3699.810606
1         4615.384615
2         4305.555556
3         6245.890861
4         4250.000000
             ...
13315     6689.834926
13316    11111.111111
13317     5258.545136
13318    10407.336319
13319     3090.909091
Name: price_per_sqft, Length: 13320, dtype: float64
```

In [25]:

```
data.describe()
```

Out[25]:

	total_sqft	bath	price	bhk	price_per_sqft
count	13274.000000	13320.000000	13320.000000	13320.000000	1.327400e+04
mean	1559.626694	2.688814	112.565627	2.802778	7.907501e+03
std	1238.405258	1.338754	148.971674	1.294496	1.064296e+05
min	1.000000	1.000000	8.000000	1.000000	2.678298e+02
25%	1100.000000	2.000000	50.000000	2.000000	4.266865e+03
50%	1276.000000	2.000000	72.000000	3.000000	5.434306e+03
75%	1680.000000	3.000000	120.000000	3.000000	7.311746e+03
max	52272.000000	40.000000	3600.000000	43.000000	1.200000e+07

In [26]:

```
data['location'].value_counts()
```

Out[26]:

```
Whitefield                          541
Sarjapur  Road                      399
Electronic City                     302
Kanakpura Road                      273
Thanisandra                         234
                                   ...
Bapuji Layout                         1
1st Stage Radha Krishna Layout        1
BEML Layout 5th stage                 1
singapura paradise                    1
Abshot Layout                         1
Name: location, Length: 1305, dtype: int64
```

In [27]:

```
data['location'] = data['location'].apply(lambda x: x.strip())
location_count = data['location'].value_counts()
```

In [28]:

```
location_count
```

Out[28]:

```
Whitefield                          542
Sarjapur  Road                      399
Electronic City                     304
Kanakpura Road                      273
Thanisandra                         237
                                   ...
Bapuji Layout                         1
1st Stage Radha Krishna Layout        1
BEML Layout 5th stage                 1
singapura paradise                    1
Abshot Layout                         1
Name: location, Length: 1294, dtype: int64
```

In [29]:

```
location_count_less_10 = location_count[location_count <= 10]
location_count_less_10
```

Out[29]:

```
Dairy Circle                      10
Nagappa Reddy Layout              10
Basapura                          10
1st Block Koramangala             10
Sector 1 HSR Layout               10
                                  ..
Bapuji Layout                      1
1st Stage Radha Krishna Layout     1
BEML Layout 5th stage              1
singapura paradise                 1
Abshot Layout                      1
Name: location, Length: 1053, dtype: int64
```

In [30]:

```
data['location'] = data['location'].apply(lambda x: "other" if x in location_count_less_10 else x)
```

In [31]:

```
data['location'].value_counts()
```

Out[31]:

```
other                  2885
Whitefield              542
Sarjapur  Road          399
Electronic City         304
Kanakpura Road          273
                       ...
Nehru Nagar              11
Banjara Layout           11
LB Shastri Nagar         11
Pattandur Agrahara       11
Narayanapura             11
Name: location, Length: 242, dtype: int64
```

Outlier detection and removal

In [32]:

```
data.describe()
```

Out[32]:

	total_sqft	bath	price	bhk	price_per_sqft
count	13274.000000	13320.000000	13320.000000	13320.000000	1.327400e+04
mean	1559.626694	2.688814	112.565627	2.802778	7.907501e+03
std	1238.405258	1.338754	148.971674	1.294496	1.064296e+05
min	1.000000	1.000000	8.000000	1.000000	2.678298e+02
25%	1100.000000	2.000000	50.000000	2.000000	4.266865e+03
50%	1276.000000	2.000000	72.000000	3.000000	5.434306e+03
75%	1680.000000	3.000000	120.000000	3.000000	7.311746e+03
max	52272.000000	40.000000	3600.000000	43.000000	1.200000e+07

In [33]:

```
(data['total_sqft']/data['bhk']).describe()
```

Out[33]:

```
count    13274.000000
mean       575.074878
std        388.205175
min          0.250000
25%        473.333333
50%        552.500000
75%        625.000000
max      26136.000000
dtype: float64
```

In [34]:

```
data = data[((data['total_sqft']/data['bhk']) >= 300)]
data.describe()
```

Out[34]:

	total_sqft	bath	price	bhk	price_per_sqft
count	12530.000000	12530.000000	12530.000000	12530.000000	12530.000000
mean	1594.564544	2.559537	111.382401	2.650838	6303.979357
std	1261.271296	1.077938	152.077329	0.976678	4162.237981
min	300.000000	1.000000	8.440000	1.000000	267.829813
25%	1116.000000	2.000000	49.000000	2.000000	4210.526316
50%	1300.000000	2.000000	70.000000	3.000000	5294.117647
75%	1700.000000	3.000000	115.000000	3.000000	6916.666667
max	52272.000000	16.000000	3600.000000	16.000000	176470.588235

In [35]:

```
data.shape
```

Out[35]:

```
(12530, 7)
```

In [36]:

```
data.price_per_sqft.describe()
```

Out[36]:

```
count     12530.000000
mean       6303.979357
std        4162.237981
min         267.829813
25%        4210.526316
50%        5294.117647
75%        6916.666667
max      176470.588235
Name: price_per_sqft, dtype: float64
```

In [37]:

```
def remove_outliers_sqft(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
data = remove_outliers_sqft(data)
data.describe()
```

Out[37]:

	total_sqft	bath	price	bhk	price_per_sqft
count	10301.000000	10301.000000	10301.000000	10301.000000	10301.000000
mean	1508.440608	2.471702	91.286372	2.574896	5659.062876
std	880.694214	0.979449	86.342786	0.897649	2265.774749
min	300.000000	1.000000	10.000000	1.000000	1250.000000
25%	1110.000000	2.000000	49.000000	2.000000	4244.897959
50%	1286.000000	2.000000	67.000000	2.000000	5175.600739
75%	1650.000000	3.000000	100.000000	3.000000	6428.571429
max	30400.000000	16.000000	2200.000000	16.000000	24509.803922

In [38]:

```
def bhk_outlier_remover(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'])].in
        return df.drop(exclude_indices, axis = 'index')
```

In [39]:

```
data = bhk_outlier_remover(data)
```

In [40]:

```
data.shape
```

Out[40]:

```
(10301, 7)
```

In [41]:

```
data
```

Out[41]:

	location	size	total_sqft	bath	price	bhk	price_per_sqft
0	1st Block Jayanagar	4 BHK	2850.0	4.0	428.00	4	15017.543860
1	1st Block Jayanagar	3 BHK	1630.0	3.0	194.00	3	11901.840491
2	1st Block Jayanagar	3 BHK	1875.0	2.0	235.00	3	12533.333333
3	1st Block Jayanagar	3 BHK	1200.0	2.0	130.00	3	10833.333333
4	1st Block Jayanagar	2 BHK	1235.0	2.0	148.00	2	11983.805668
...	...	...	...	...	...	...	...
10296	other	2 BHK	1353.0	2.0	110.00	2	8130.081301
10297	other	1 Bedroom	812.0	1.0	26.00	1	3201.970443
10298	other	3 BHK	1440.0	2.0	63.93	3	4439.583333
10299	other	2 BHK	1075.0	2.0	48.00	2	4465.116279
10300	other	4 BHK	3600.0	5.0	400.00	4	11111.111111

10301 rows × 7 columns

In [42]:

```
data.drop(columns=['size','price_per_sqft'], inplace = True)
```

Cleaned data

In [43]:

```
data.head()
```

Out[43]:

	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
4	1st Block Jayanagar	1235.0	2.0	148.0	2

In [44]:

```
data.to_csv("Cleaned_data.csv")
```

In [45]:

```
X = data.drop(columns = ['price'])
y = data['price']
```

In [46]:

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, Lasso, Ridge
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import make_column_transformer
from sklearn.pipeline import make_pipeline
from sklearn.metrics import r2_score
```

In [47]:

```
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2, random_state=0)
```

In [48]:

```
print(X_train.shape)
```

```
(8240, 4)
```

In [49]:

```
print(X_test.shape)
```

```
(2061, 4)
```

Applying Linear Regression

In [50]:

```
column_trans = make_column_transformer((OneHotEncoder(sparse = False),['location']), remainder = 'passthrough')
```

In [51]:

```
scaler = StandardScaler()
```

In [52]:

```
lr = LinearRegression()
```

In [53]:

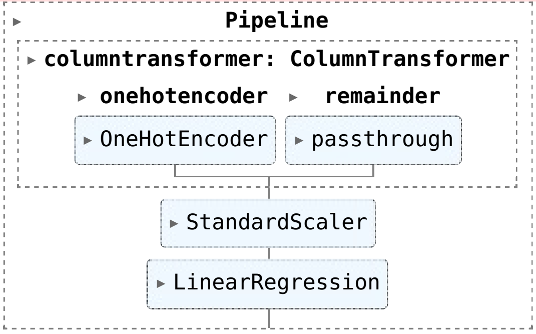
```
pipe = make_pipeline(column_trans, scaler,lr)
```

In [54]:

```
pipe.fit(X_train, y_train)
```

```
C:\Users\absol\anaconda3\lib\site-packages\sklearn\preprocessing\_encoders.py:828: FutureWarning: `sparse` was
renamed to `sparse_output` in version 1.2 and will be removed in 1.4. `sparse_output` is ignored unless you lea
ve `sparse` to its default value.
  warnings.warn(
```

Out[54]:



In [55]:

```
y_pred_lr = pipe.predict(X_test)
```

In [56]:

```
r2_score(y_test, y_pred_lr)
```

Out[56]:

```
0.8294478549591062
```

Applying Lasso

In [57]:

```
lasso = Lasso()
```

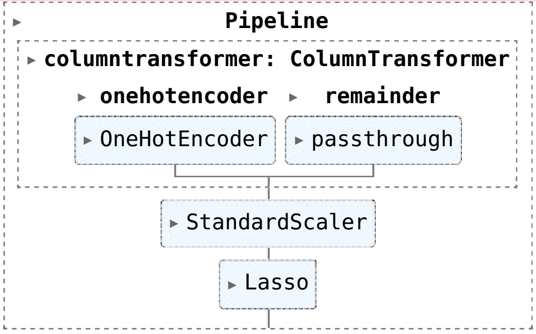
In [57]:
```
lasso = Lasso()
```

In [58]:
```
pipe = make_pipeline(column_trans, scaler, lasso)
```

In [59]:
```
pipe.fit(X_train, y_train)
```

```
C:\Users\absol\anaconda3\lib\site-packages\sklearn\preprocessing\_encoders.py:828: FutureWarning: `sparse` was
renamed to `sparse_output` in version 1.2 and will be removed in 1.4. `sparse_output` is ignored unless you lea
ve `sparse` to its default value.
  warnings.warn(
```

Out[59]:



In [60]:
```
y_pred_lasso = pipe.predict(X_test)
r2_score(y_test, y_pred_lasso)
```

Out[60]:
```
0.8222119691869108
```

Applying Ridge

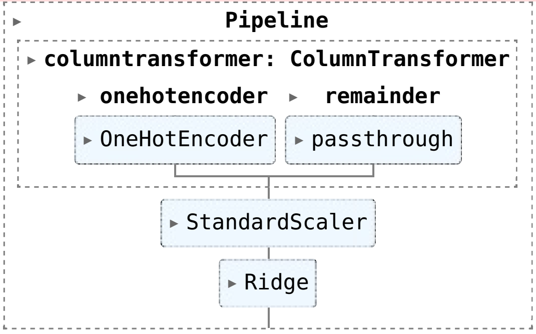
In [61]:
```
ridge = Ridge()
```

In [62]:
```
pipe = make_pipeline(column_trans, scaler, ridge)
```

In [63]:
```
pipe.fit(X_train, y_train)
```

```
C:\Users\absol\anaconda3\lib\site-packages\sklearn\preprocessing\_encoders.py:828: FutureWarning: `sparse` was
renamed to `sparse_output` in version 1.2 and will be removed in 1.4. `sparse_output` is ignored unless you lea
ve `sparse` to its default value.
  warnings.warn(
```

Out[63]:



In [64]:
```
y_pred_ridge = pipe.predict(X_test)
r2_score(y_test, y_pred_ridge)
```

Out[64]:
```
0.8294558115108042
```

In [65]:
```
print("No Regularization: ", r2_score(y_test,  y_pred_lr))
print("Lasso: ", r2_score(y_test, y_pred_lasso))
print("Ridge: ",r2_score(y_test, y_pred_ridge))
```

```
No Regularization:  0.8294478549591062
Lasso:  0.8222119691869108
Ridge:  0.8294558115108042
```

In [66]:
```
import pickle
```

In [67]:
```
pickle.dump(pipe, open('RidgeModel.pkl','wb'))
```

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js