

HOUSE PRICE ANALYSIS VISUALIZATION

NAME: VARSHINI A S

```
from google.colab import files
uploaded = files.upload()
```

Choose Files House Price India.csv
• House Price India.csv(text/csv) • 1524561 bytes, last modified: 10/3/2023 • 100% done
Saving House Price India.csv to House Price India.csv

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import io
df = pd.read_csv(io.BytesIO(uploaded['House Price India.csv']))
df.head()
```

	id	Date	number of bedrooms	number of bathrooms	living area	lot area	number of floors	waterfront present	number of views	condition of the house	...	Be
0	6762810145	42491	5	2.50	3650	9050	2.0	0	4	5	...	1
1	6762810635	42491	4	2.50	2920	4000	1.5	0	0	5	...	1
2	6762810998	42491	5	2.75	2910	9480	1.5	0	0	3	...	1
3	6762812605	42491	4	2.50	3310	42998	2.0	0	0	3	...	2
4	6762812919	42491	3	2.00	2710	4500	1.5	0	0	4	...	1

5 rows × 23 columns

```
df.tail()
```

	id	Date	number of bedrooms	number of bathrooms	living area	lot area	number of floors	waterfront present	number of views	condition of the house	..
14615	6762830250	42734	2	1.5	1556	20000	1.0	0	0	4	..
14616	6762830339	42734	3	2.0	1680	7000	1.5	0	0	4	..
14617	6762830618	42734	2	1.0	1070	6120	1.0	0	0	3	..
14618	6762830709	42734	4	1.0	1030	6621	1.0	0	0	4	..
14619	6762831463	42734	3	1.0	900	4770	1.0	0	0	3	..

5 rows × 23 columns

```
df
```

	id	Date	number of bedrooms	number of bathrooms	living area	lot area	number of floors	waterfront present	number of views	condition of the house	..
0	6762810145	42491	5	2.50	3650	9050	2.0	0	4	5	..
1	6762810635	42491	4	2.50	2920	4000	1.5	0	0	5	..
2	6762810998	42491	5	2.75	2910	9480	1.5	0	0	3	..
3	6762812605	42491	4	2.50	3310	42998	2.0	0	0	3	..
4	6762812919	42491	3	2.00	2710	4500	1.5	0	0	4	..
...
14615	6762830250	42734	2	1.50	1556	20000	1.0	0	0	4	..
14616	6762830339	42734	3	2.00	1680	7000	1.5	0	0	4	..
14617	6762830618	42734	2	1.00	1070	6120	1.0	0	0	3	..
14618	6762830709	42734	4	1.00	1030	6621	1.0	0	0	4	..
14619	6762831463	42734	3	1.00	900	4770	1.0	0	0	3	..

14620 rows × 23 columns

```
df.columns

Index(['id', 'Date', 'number of bedrooms', 'number of bathrooms',
      'living area', 'lot area', 'number of floors', 'waterfront present',
      'number of views', 'condition of the house', 'grade of the house',
      'Area of the house(excluding basement)', 'Area of the basement',
      'Built Year', 'Renovation Year', 'Postal Code', 'Latitude',
      'Longitude', 'living_area_renov', 'lot_area_renov',
      'Number of schools nearby', 'Distance from the airport', 'Price'],
      dtype='object')
```

```
df.dtypes
```

```
id                int64
Date              int64
number of bedrooms    int64
number of bathrooms  float64
living area         int64
lot area           int64
number of floors     float64
waterfront present  int64
number of views      int64
condition of the house    int64
grade of the house    int64
Area of the house(excluding basement)  int64
Area of the basement  int64
Built Year          int64
Renovation Year      int64
Postal Code         int64
Latitude            float64
Longitude           float64
living_area_renov   int64
lot_area_renov      int64
Number of schools nearby    int64
Distance from the airport  int64
Price              int64
dtype: object
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14620 entries, 0 to 14619
Data columns (total 23 columns):
#   Column                                     Non-Null Count  Dtype
---  -
0   id                                         14620 non-null  int64
1   Date                                       14620 non-null  int64
2   number of bedrooms                       14620 non-null  int64
3   number of bathrooms                     14620 non-null  float64
4   living area                             14620 non-null  int64
5   lot area                                14620 non-null  int64
6   number of floors                         14620 non-null  float64
7   waterfront present                      14620 non-null  int64
8   number of views                         14620 non-null  int64
9   condition of the house                  14620 non-null  int64
10  grade of the house                      14620 non-null  int64
11  Area of the house(excluding basement)    14620 non-null  int64
12  Area of the basement                    14620 non-null  int64
13  Built Year                              14620 non-null  int64
14  Renovation Year                          14620 non-null  int64
15  Postal Code                             14620 non-null  int64
16  Latitude                                14620 non-null  float64
17  Longitude                               14620 non-null  float64
18  living_area_renov                       14620 non-null  int64
19  lot_area_renov                          14620 non-null  int64
20  Number of schools nearby                 14620 non-null  int64
21  Distance from the airport               14620 non-null  int64
22  Price                                   14620 non-null  int64
dtypes: float64(4), int64(19)
memory usage: 2.6 MB
```

```
df.shape
```

```
(14620, 23)
```

Univariate Analysis

```
print(df.describe())
```

std	6.237575e+03	67.347991	0.938719	0.769934
min	6.762810e+09	42491.000000	1.000000	0.500000
25%	6.762815e+09	42546.000000	3.000000	1.750000
50%	6.762821e+09	42600.000000	3.000000	2.250000
75%	6.762826e+09	42662.000000	4.000000	2.500000
max	6.762832e+09	42734.000000	33.000000	8.000000

	living area	lot area	number of floors	waterfront present \
count	14620.000000	1.462000e+04	14620.000000	14620.000000
mean	2098.262996	1.509328e+04	1.502360	0.007661
std	928.275721	3.791962e+04	0.540239	0.087193
min	370.000000	5.200000e+02	1.000000	0.000000
25%	1440.000000	5.010750e+03	1.000000	0.000000
50%	1930.000000	7.620000e+03	1.500000	0.000000
75%	2570.000000	1.080000e+04	2.000000	0.000000
max	13540.000000	1.074218e+06	3.500000	1.000000

	number of views	condition of the house	...	Built Year \
count	14620.000000	14620.000000	...	14620.000000
mean	0.233105	3.430506	...	1970.926402
std	0.766259	0.664151	...	29.493625
min	0.000000	1.000000	...	1900.000000
25%	0.000000	3.000000	...	1951.000000
50%	0.000000	3.000000	...	1975.000000
75%	0.000000	4.000000	...	1997.000000
max	4.000000	5.000000	...	2015.000000

	Renovation Year	Postal Code	Lattitude	Longitude \
count	14620.000000	14620.000000	14620.000000	14620.000000
mean	90.924008	122033.062244	52.792848	-114.404007
std	416.216661	19.082418	0.137522	0.141326
min	0.000000	122003.000000	52.385900	-114.709000
25%	0.000000	122017.000000	52.707600	-114.519000
50%	0.000000	122032.000000	52.806400	-114.421000
75%	0.000000	122048.000000	52.908900	-114.315000
max	2015.000000	122072.000000	53.007600	-113.505000

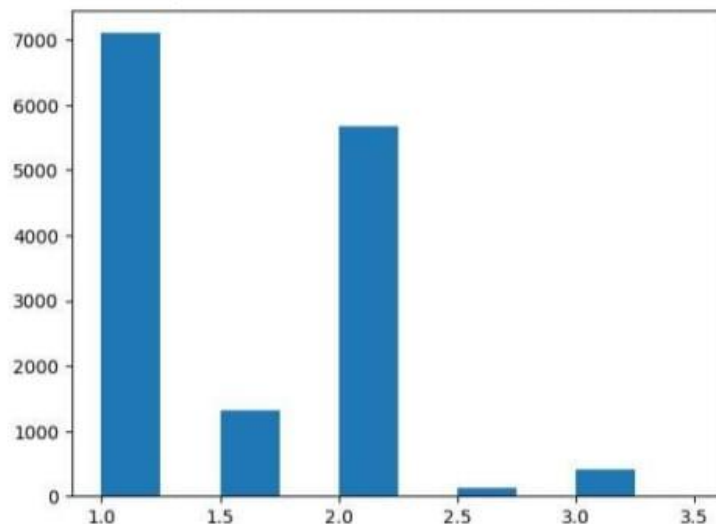
	living_area_renov	lot_area_renov	Number of schools nearby \
count	14620.000000	14620.000000	14620.000000
mean	1996.702257	12753.500068	2.012244
std	691.093366	26058.414467	0.817284
min	460.000000	651.000000	1.000000
25%	1490.000000	5097.750000	1.000000
50%	1850.000000	7620.000000	2.000000
75%	2380.000000	10125.000000	3.000000
max	6110.000000	560617.000000	3.000000

	Distance from the airport	Price
count	14620.000000	1.462000e+04
mean	64.950958	5.389322e+05
std	8.936008	3.675324e+05
min	50.000000	7.800000e+04
25%	57.000000	3.200000e+05
50%	65.000000	4.500000e+05
75%	73.000000	6.450000e+05
max	80.000000	7.700000e+06

[8 rows x 23 columns]

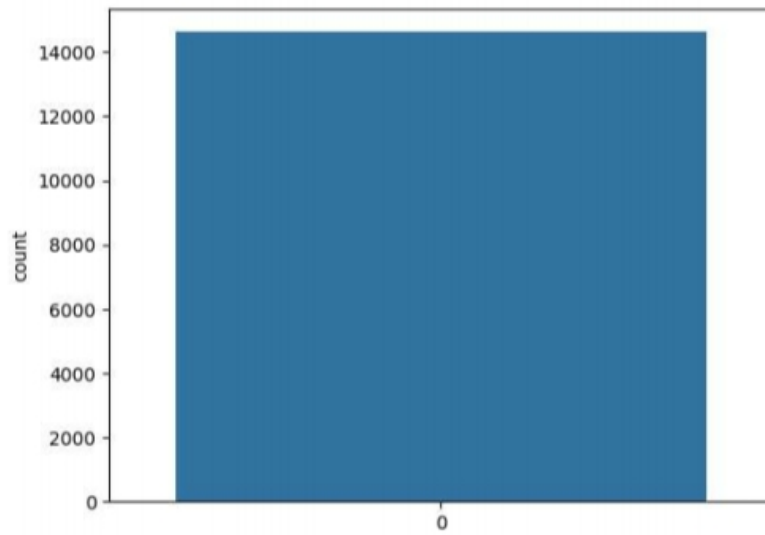
```
plt.hist(df['number of floors'])
```

```
(array([7.103e+03, 0.000e+00, 1.311e+03, 0.000e+00, 5.666e+03, 0.000e+00,
        1.180e+02, 0.000e+00, 4.180e+02, 4.000e+00]),
 array([1. , 1.25, 1.5 , 1.75, 2. , 2.25, 2.5 , 2.75, 3. , 3.25, 3.5 ]),
 <BarContainer object of 10 artists>)
```



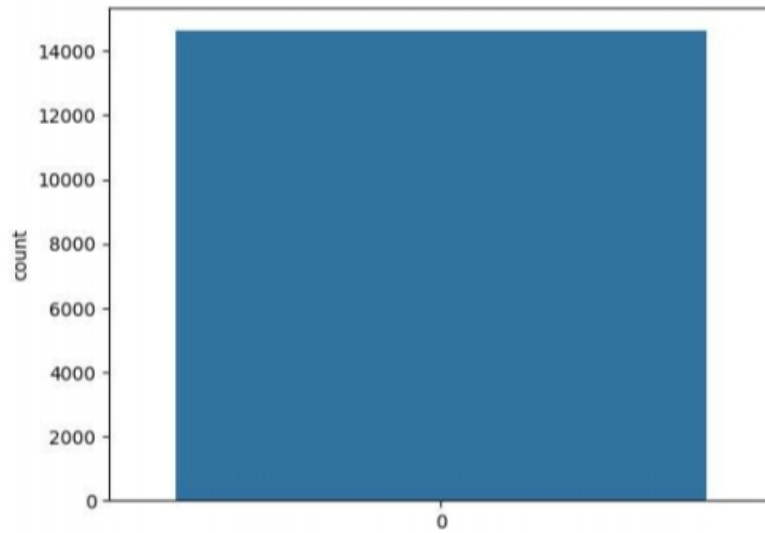
```
sns.countplot(df['number of bedrooms'])
```

<Axes: ylabel='count'>



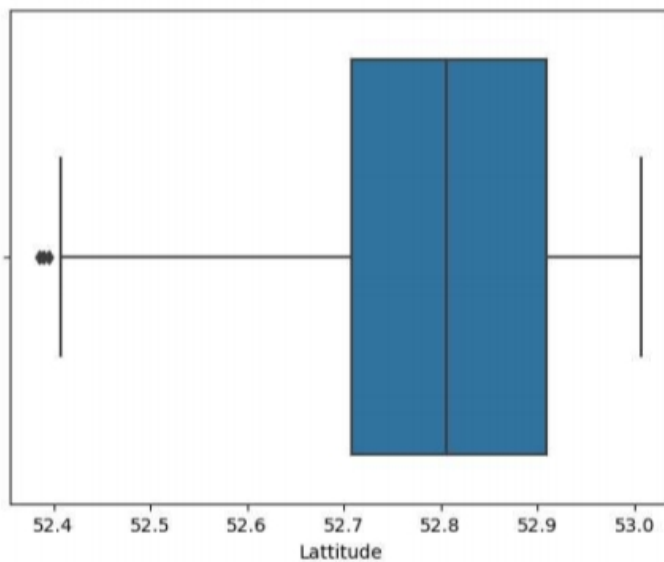
```
sns.countplot(df['Area of the basement'])
```

<Axes: ylabel='count'>



```
sns.boxplot(x=df['Latitude'])
```

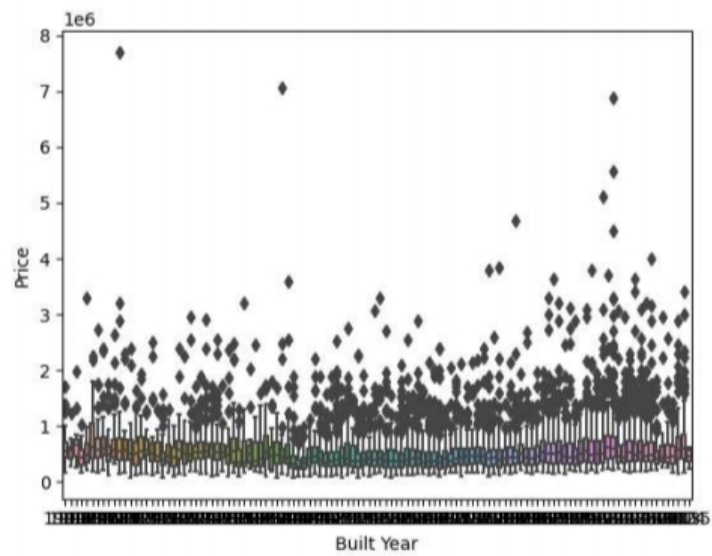
<Axes: xlabel='Latitude'>



Bivariate Analysis

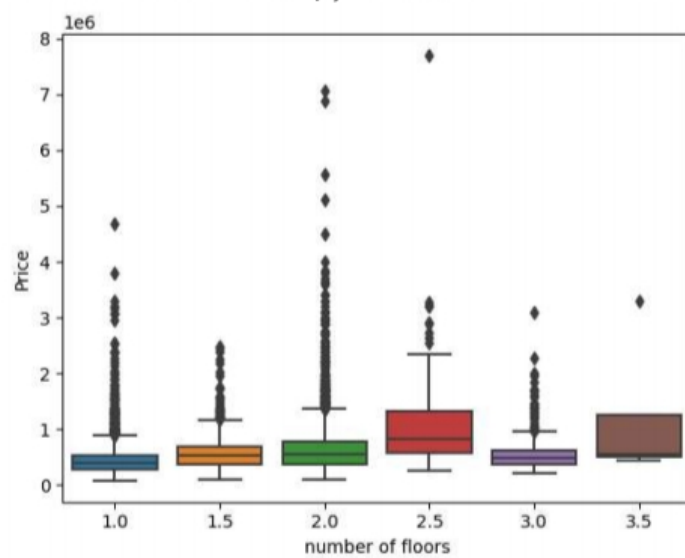
```
sns.boxplot(x=df['Built Year'],y=df['Price'])
```

```
<Axes: xlabel='Built Year', ylabel='Price'>
```



```
sns.boxplot(x=df['number of floors'],y=df['Price'])
```

```
<Axes: xlabel='number of floors', ylabel='Price'>
```



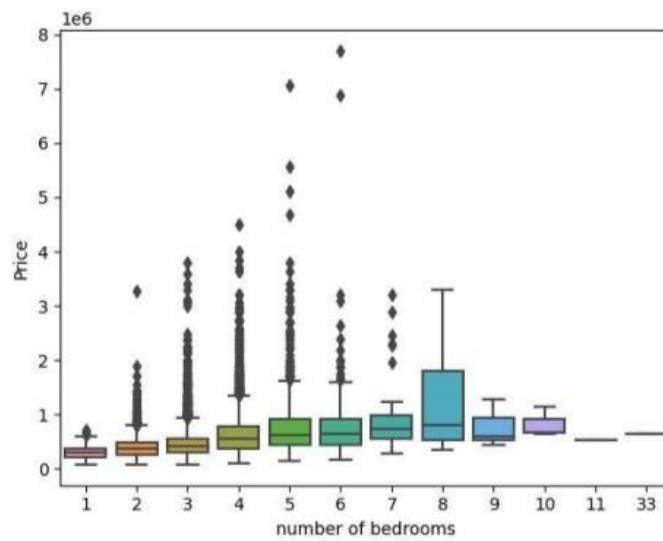
```
sns.lineplot(x=df['Longitude'],y=df['Price'])
```

```
<Axes: xlabel='Longitude', ylabel='Price'>
```

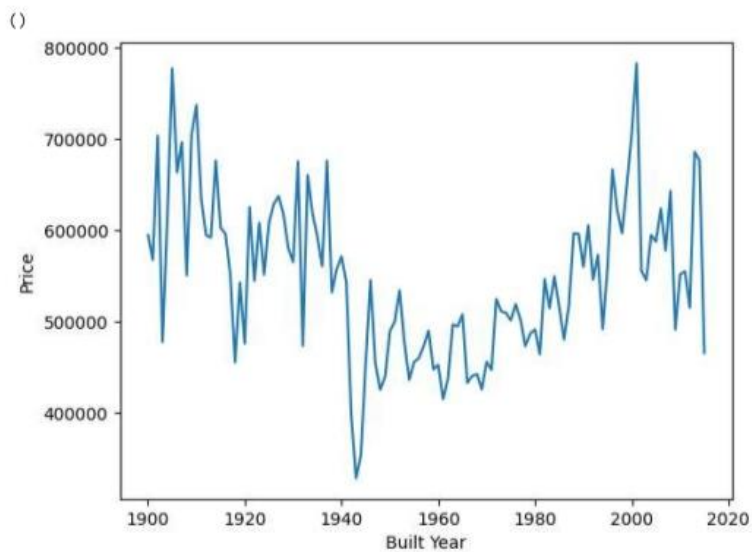


```
sns.boxplot(x=df['number of bedrooms'],y=df['Price'])
```

```
<Axes: xlabel='number of bedrooms', ylabel='Price'>
```



```
sns.lineplot(x=df.groupby('Built Year').mean().index,y=df.groupby('Built Year').mean()['Price'])  
plt.show  
( )
```



```
sns.heatmap(df[['Price','number of bedrooms','number of bathrooms']].corr(),annot=True)
```



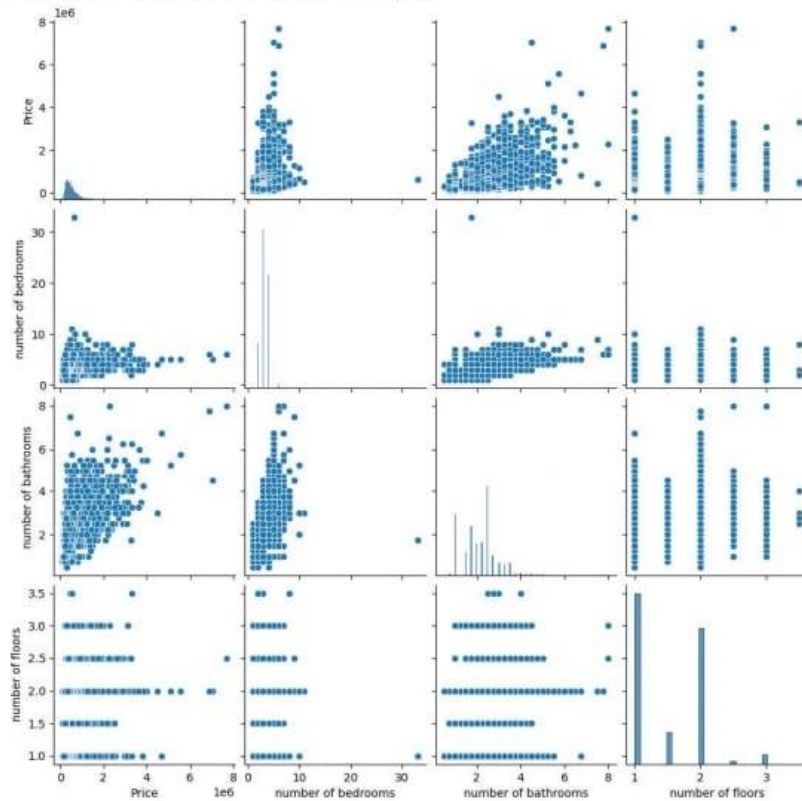
<Axes: >



Multivariate Analysis

```
sns.pairplot(df[['Price', 'number of bedrooms', 'number of bathrooms', 'number of floors']])
```

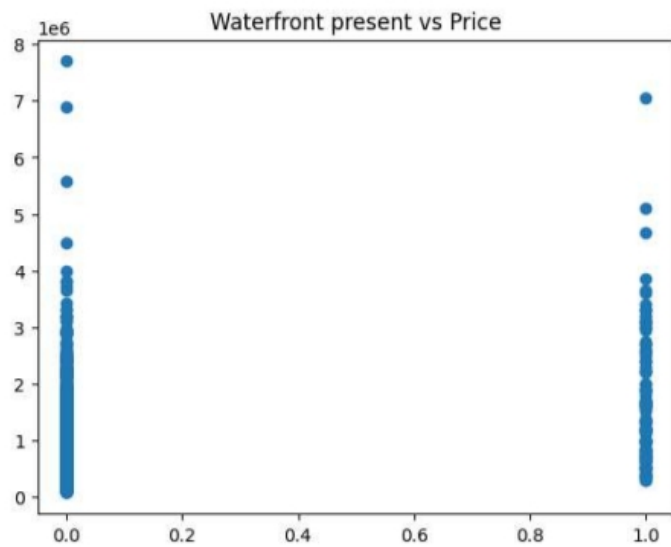
<seaborn.axisgrid.PairGrid at 0x7bffd0d33ee0>



```
df.duplicated().sum()
```

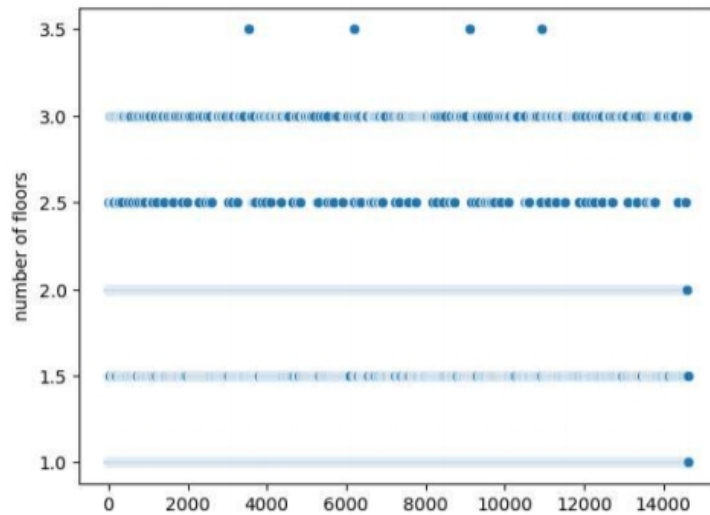
0

```
plt.scatter(df['waterfront present'], df['Price'])  
plt.title("Waterfront present vs Price")
```

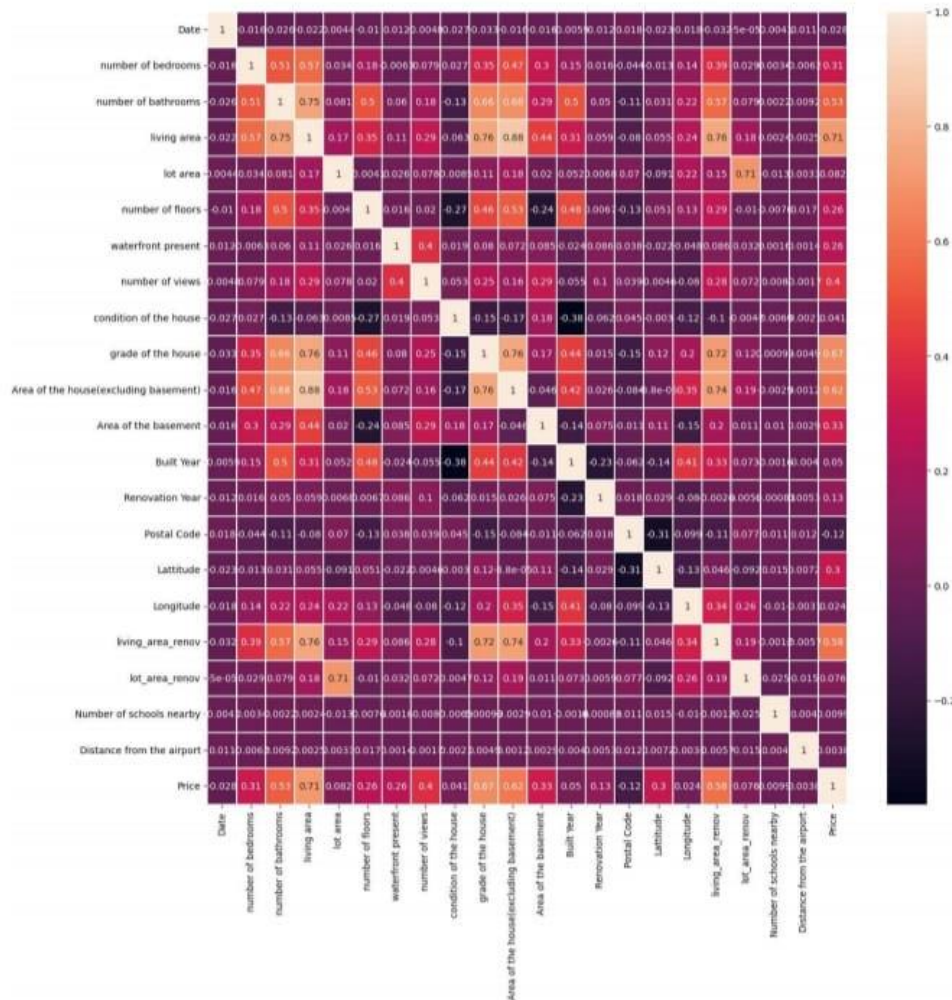


```
sns.scatterplot(df['number of floors'])
```

<Axes: ylabel='number of floors'>



```
plt.subplots(figsize=(15,15))
sns.heatmap(df.drop(['id'],axis=1).corr(),linewidth=0.3,annot=True)
plt.show()
```

```
print(df.describe())
```

std	6.237575e+03	67.347991	0.938719	0.769934
min	6.762810e+09	42491.000000	1.000000	0.500000
25%	6.762815e+09	42546.000000	3.000000	1.750000
50%	6.762821e+09	42600.000000	3.000000	2.250000
75%	6.762826e+09	42662.000000	4.000000	2.500000
max	6.762832e+09	42734.000000	33.000000	8.000000

	living area	lot area	number of floors	waterfront present \
count	14620.000000	1.462000e+04	14620.000000	14620.000000
mean	2098.262996	1.509328e+04	1.502360	0.007661
std	928.275721	3.791962e+04	0.540239	0.087193
min	370.000000	5.200000e+02	1.000000	0.000000
25%	1440.000000	5.010750e+03	1.000000	0.000000
50%	1930.000000	7.620000e+03	1.500000	0.000000
75%	2570.000000	1.080000e+04	2.000000	0.000000
max	13540.000000	1.074218e+06	3.500000	1.000000

	number of views	condition of the house	...	Built Year \
count	14620.000000	14620.000000	...	14620.000000
mean	0.233105	3.430506	...	1970.926402
std	0.766259	0.664151	...	29.493625
min	0.000000	1.000000	...	1900.000000
25%	0.000000	3.000000	...	1951.000000
50%	0.000000	3.000000	...	1975.000000
75%	0.000000	4.000000	...	1997.000000
max	4.000000	5.000000	...	2015.000000

mean	90.924008	122033.002244	52.792848	-114.409007
std	416.216661	19.082418	0.137522	0.141326
min	0.000000	122003.000000	52.385900	-114.709000
25%	0.000000	122017.000000	52.707600	-114.519000
50%	0.000000	122032.000000	52.806400	-114.421000
75%	0.000000	122048.000000	52.908900	-114.315000
max	2015.000000	122072.000000	53.007600	-113.505000

	living_area_renov	lot_area_renov	Number of schools nearby \
count	14620.000000	14620.000000	14620.000000
mean	1996.702257	12753.500068	2.012244
std	691.093366	26058.414467	0.817284
min	460.000000	651.000000	1.000000
25%	1490.000000	5097.750000	1.000000
50%	1850.000000	7620.000000	2.000000
75%	2380.000000	10125.000000	3.000000
max	6110.000000	560617.000000	3.000000

	Distance from the airport	Price
count	14620.000000	1.462000e+04
mean	64.950958	5.389322e+05
std	8.936008	3.675324e+05
min	50.000000	7.800000e+04
25%	57.000000	3.200000e+05
50%	65.000000	4.500000e+05
75%	73.000000	6.450000e+05
max	80.000000	7.700000e+06

[8 rows x 23 columns]

```
print(df.count())
```

id	14620
Date	14620
number of bedrooms	14620
number of bathrooms	14620
living area	14620
lot area	14620
number of floors	14620
waterfront present	14620
number of views	14620
condition of the house	14620
grade of the house	14620
Area of the house(excluding basement)	14620
Area of the basement	14620
Built Year	14620
Renovation Year	14620
Postal Code	14620
Latitude	14620
Longitude	14620
living_area_renov	14620
lot_area_renov	14620
Number of schools nearby	14620
Distance from the airport	14620
Price	14620
dtype: int64	

```
print(df.corr())
```

```

distance from the airport      0.004035
Price                          0.009890

Distance from the airport      Price
id                             -0.004542 -0.773114
Date                           0.011457 -0.027919
number of bedrooms             -0.006157  0.308460
number of bathrooms            0.009206  0.531735
living area                    0.002511  0.712169
lot area                       0.003291  0.081992
number of floors               0.016567  0.262732
waterfront present             0.001448  0.263687
number of views                -0.001657  0.395973
condition of the house         -0.002136  0.041376
grade of the house             0.004940  0.671814
Area of the house(excluding basement) 0.001222  0.615220
Area of the basement           0.002926  0.330202
Built Year                     -0.003968  0.050307
Renovation Year                0.005342  0.133173
Postal Code                    0.011528 -0.115908
Latitude                       0.007193  0.297490
Longitude                      -0.003100  0.024414
living_area_renov              -0.005673  0.584924
lot_area_renov                 -0.014587  0.075535
Number of schools nearby        0.004035  0.009890
Distance from the airport      1.000000  0.003804
Price                          0.003804  1.000000

```

```
[23 rows x 23 columns]
```

```
print(df['Number of schools nearby'].value_counts())
```

```

3    4973
2    4853
1    4794
Name: Number of schools nearby, dtype: int64

```

```

print('Mean:',df['Distance from the airport'].mean())
print('Median:',df['Area of the basement'].median())
print('Mode:',df['grade of the house'].mode())

```

```

Mean: 64.95095759233926
Median: 0.0
Mode: 0    7
Name: grade of the house, dtype: int64

```

Handle the Missing values

```
print(df.isnull().sum())
```

```

id                0
Date              0
number of bedrooms 0
number of bathrooms 0
living area       0
lot area         0
number of floors  0
waterfront present 0
number of views   0
condition of the house 0
grade of the house 0
Area of the house(excluding basement) 0
Area of the basement 0
Built Year        0
Renovation Year   0
Postal Code       0
Latitude          0
Longitude         0
living_area_renov 0
lot_area_renov    0
Number of schools nearby 0
Distance from the airport 0
Price            0
dtype: int64

```

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

```
df.fillna(0,inplace=True)
```

```
df.interpolate(inplace=True)
```

```
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
```

```
x=df.drop(['Price','Date'],axis=1)
x.set_index(['id'],inplace=True)
y=df[['id','Price']]
```

```
x.head()
```

	number of bedrooms	number of bathrooms	living area	lot area	number of floors	waterfront present	number of views	condition of the house	grade of the house	Area of the house(excluding basement)	Area of the basement	Built Year	Reno
id													
6762810145	5	2.50	3650	9050	2.0	0	4	5	10	3370	280	1921	
6762810635	4	2.50	2920	4000	1.5	0	0	5	8	1910	1010	1909	
6762810998	5	2.75	2910	9480	1.5	0	0	3	8	2910	0	1939	
6762812605	4	2.50	3310	42998	2.0	0	0	3	9	3310	0	2001	
6762812919	3	2.00	2710	4500	1.5	0	0	4	8	1880	830	1929	

```
y.head()
```

	id	Price
0	6762810145	2380000
1	6762810635	1400000
2	6762810998	1200000
3	6762812605	838000
4	6762812919	805000

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import r2_score
```

```
x_train,x_test,y_train,y_test = train_test_split(x,y['Price'],test_size =0.1,random_state=2)
model = GradientBoostingRegressor(n_estimators=400,max_depth=5,min_samples_split=2,learning_rate=0.1)
model.fit(x_train,y_train)
```

```

* GradientBoostingRegressor
GradientBoostingRegressor(max_depth=5, n_estimators=400)
```

```
y_pred = model.predict(x_test)
model.score(x_test,y_test)
```

```
0.9119464499193015
```

```
r2_score(y_pred,y_test)
```

```
0.9013689036658794
```



```
y_pred
```

```
array([497766.12740438, 244495.3776842 , 293819.40063242, ...,
        698495.60350629, 297006.00386358, 245881.76921871])
```

```
y_pred_list = y['id'][-len(y_pred):].tolist()
```

```
y_pred_df=pd.DataFrame(y_pred_list,columns=['ID'])
y_pred_df['Predicted Price']= y_pred.round(2)
```

```
y_pred_df
```

ID Predicted Price			
0	6762811233	497766.13	
1	6762811403	244495.38	
2	6762811775	293819.40	
3	6762811861	397555.35	
4	6762812009	474843.29	
...	
1457	6762830250	1041014.57	
1458	6762830339	317512.59	
1459	6762830618	698495.60	
1460	6762830709	297006.00	
1461	6762831463	245881.77	

1462 rows × 2 columns