

```
In [62]: import pandas as pd  
import numpy as np
```

```
In [63]: data=pd.read_csv(r"D:\ML\real_estate_price_size_year_view 3rd march new.csv")  
data
```

```
Out[63]:
```

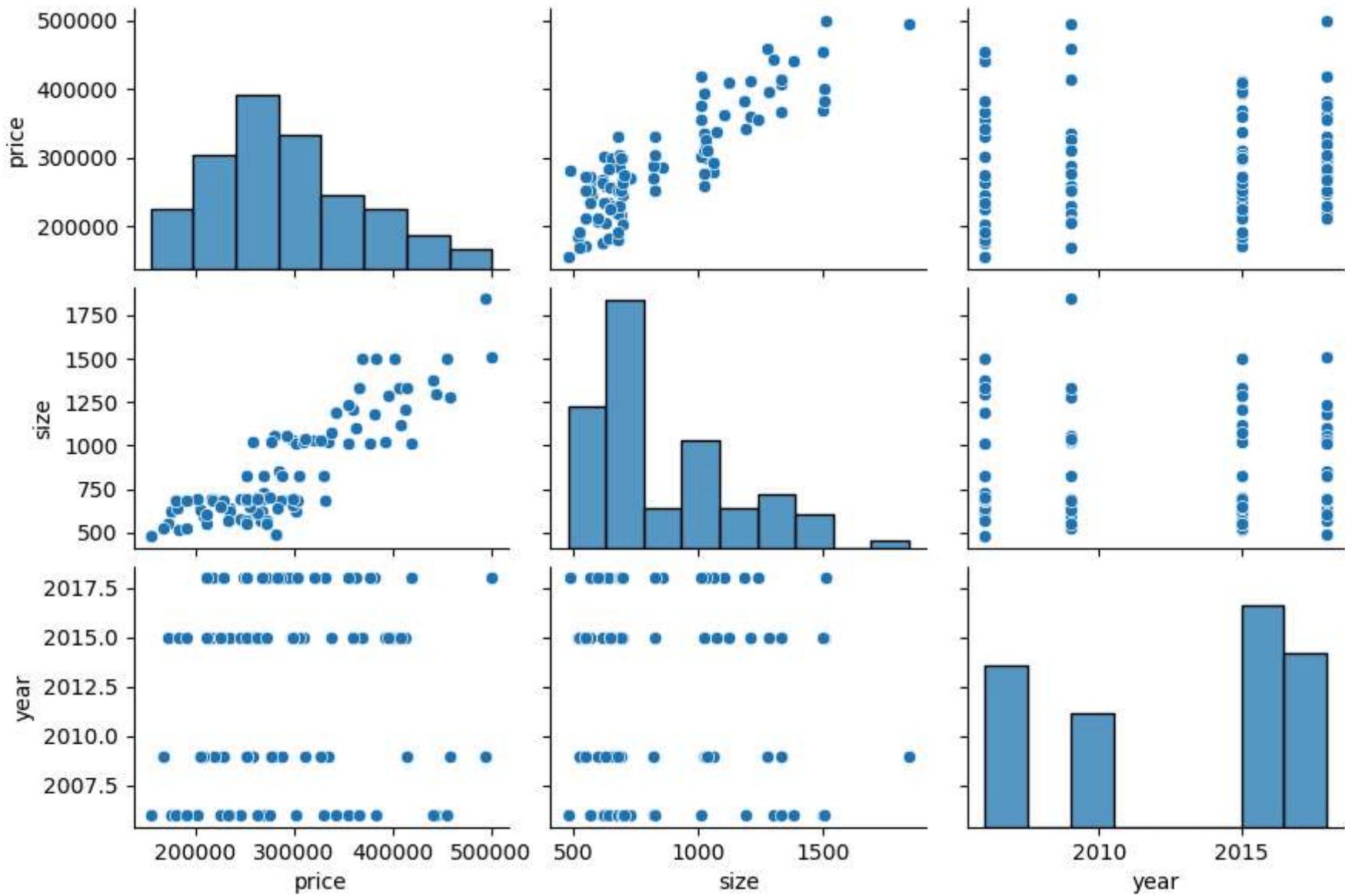
	price	size	year	view
0	234314.144	643.09	2015	No sea view
1	228581.528	656.22	2009	No sea view
2	281626.336	487.29	2018	Sea view
3	401255.608	1504.75	2015	No sea view
4	458674.256	1275.46	2009	Sea view
...	...	...	...	...
95	252460.400	549.80	2009	Sea view
96	310522.592	1037.44	2009	No sea view
97	383635.568	1504.75	2006	No sea view
98	225145.248	648.29	2015	No sea view
99	274922.856	705.29	2006	Sea view

100 rows × 4 columns

```
In [64]: import seaborn as sns  
sns.pairplot(data,height=2,aspect=1.5)
```

```
D:\Anaconda\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight  
self._figure.tight_layout(*args, **kwargs)  
<seaborn.axisgrid.PairGrid at 0x25de6465950>
```

```
Out[64]:
```



In [65]: `data.describe()`

Out[65]:

	price	size	year
<b>count</b>	100.000000	100.000000	100.000000
<b>mean</b>	292289.470160	853.024200	2012.600000
<b>std</b>	77051.727525	297.941951	4.729021
<b>min</b>	154282.128000	479.750000	2006.000000
<b>25%</b>	234280.148000	643.330000	2009.000000
<b>50%</b>	280590.716000	696.405000	2015.000000
<b>75%</b>	335723.696000	1029.322500	2018.000000
<b>max</b>	500681.128000	1842.510000	2018.000000

In [66]: `data.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 4 columns):
 #   Column   Non-Null Count   Dtype  
 ---  --  
 0   price    100 non-null    float64 
 1   size     100 non-null    float64 
 2   year     100 non-null    int64  
 3   view     100 non-null    object  
dtypes: float64(2), int64(1), object(1)
memory usage: 3.3+ KB
```

In [67]: `data1=pd.get_dummies(data)`  
`data1`

Out[67]:

	price	size	year	view_No sea view	view_Sea view
0	234314.144	643.09	2015	True	False
1	228581.528	656.22	2009	True	False
2	281626.336	487.29	2018	False	True
3	401255.608	1504.75	2015	True	False
4	458674.256	1275.46	2009	False	True
...	...	...	...	...	...
95	252460.400	549.80	2009	False	True
96	310522.592	1037.44	2009	True	False
97	383635.568	1504.75	2006	True	False
98	225145.248	648.29	2015	True	False
99	274922.856	705.29	2006	False	True

100 rows × 5 columns

In [68]:

```
x=data1.drop('price',axis='columns')  
x
```

Out[68]:

	size	year	view_No sea view	view_Sea view
0	643.09	2015	True	False
1	656.22	2009	True	False
2	487.29	2018	False	True
3	1504.75	2015	True	False
4	1275.46	2009	False	True
...	...	...	...	...
95	549.80	2009	False	True
96	1037.44	2009	True	False
97	1504.75	2006	True	False
98	648.29	2015	True	False
99	705.29	2006	False	True

100 rows × 4 columns

In [69]:

```
y=data1.price  
print(x)  
print(y)
```

```
      size  year  view_No sea view  view_Sea view
0    643.09  2015        True  False
1    656.22  2009        True  False
2    487.29  2018       False  True
3   1504.75  2015        True  False
4   1275.46  2009       False  True
..     ...
95   549.80  2009       False  True
96  1037.44  2009        True  False
97  1504.75  2006        True  False
98   648.29  2015        True  False
99   705.29  2006       False  True
```

[100 rows x 4 columns]

```
0    234314.144
1    228581.528
2    281626.336
3    401255.608
4    458674.256
..
95   252460.400
96   310522.592
97   383635.568
98   225145.248
99   274922.856
Name: price, Length: 100, dtype: float64
```

```
In [70]: from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.4, random_state=2)
```

```
In [71]: print(x_train)
print(x_test)
print(y_train)
print(y_test)
```

	size	year	view_No	sea	view	view_Sea	view
12	694.52	2015		True		False	
53	727.88	2006		False		True	
87	1028.41	2009		False		True	
54	647.50	2015		False		True	
95	549.80	2009		False		True	
32	597.90	2009		True		False	
19	1027.76	2018		True		False	
26	570.89	2018		False		True	
60	828.16	2018		True		False	
55	1508.84	2018		False		True	
9	694.52	2009		True		False	
96	1037.44	2009		True		False	
17	623.94	2006		False		True	
59	569.17	2015		False		True	
57	1283.85	2015		False		True	
41	682.26	2018		True		False	
64	685.48	2015		False		True	
45	698.29	2015		False		True	
97	1504.75	2006		True		False	
8	682.26	2018		False		True	
71	643.41	2006		True		False	
94	698.29	2006		False		True	
90	694.52	2018		True		False	
98	648.29	2015		True		False	
86	479.75	2006		True		False	
80	681.07	2006		True		False	
50	647.50	2015		True		False	
52	1021.95	2009		True		False	
66	1009.25	2006		False		True	
88	601.66	2018		True		False	
70	1021.95	2009		True		False	
46	633.19	2009		True		False	
68	685.48	2018		False		True	
69	1496.36	2006		False		True	
81	1122.34	2015		False		True	
58	827.84	2006		False		True	
33	525.81	2015		True		False	
38	685.48	2018		False		True	
51	1021.95	2015		False		True	
42	823.21	2009		False		True	
4	1275.46	2009		False		True	
39	698.29	2015		True		True	
							False

37	570.25	2006	False	True
20	620.71	2015	False	True
31	681.07	2006	True	False
63	1021.95	2009	False	True
47	698.29	2006	True	False
85	1009.25	2018	False	True
93	698.29	2018	True	False
49	617.05	2015	False	True
34	857.54	2018	True	False
7	620.82	2006	True	False
75	685.48	2018	False	True
82	681.07	2006	True	False
43	1334.10	2009	True	False
22	1207.45	2015	False	True
72	656.22	2015	False	True
15	1379.72	2006	False	True
40	1021.95	2015	True	False
	size	year	view_No sea view	view_Sea view
83	643.09	2018	False	True
30	1010.33	2006	True	False
56	1032.06	2018	True	False
24	525.81	2009	True	False
16	690.54	2018	True	False
23	518.38	2015	True	False
2	487.29	2018	False	True
27	1334.10	2015	False	True
28	681.07	2015	False	True
13	1009.25	2018	True	False
99	705.29	2006	False	True
92	694.52	2015	False	True
76	1183.46	2018	False	True
14	1300.96	2006	False	True
0	643.09	2015	True	False
21	549.69	2015	True	False
3	1504.75	2015	True	False
29	1496.36	2015	True	False
61	698.50	2015	True	False
79	1188.62	2006	False	True
35	622.97	2018	False	True
11	1842.51	2009	False	True
84	685.48	2018	False	True
44	1060.36	2018	True	False
	5	575.19	2006	False
				True

25	1103.30	2018	False	True
77	1334.10	2006	True	False
74	685.48	2018	True	False
62	1205.62	2015	True	False
65	827.09	2015	True	False
1	656.22	2009	True	False
18	681.07	2006	True	False
48	633.19	2015	False	True
36	823.21	2006	True	False
78	682.26	2009	False	True
6	570.89	2015	False	True
89	1236.93	2018	True	False
91	1071.55	2015	True	False
10	1060.36	2009	True	False
12	215472.104			
53	269523.056			
87	327252.112			
54	255629.160			
95	252460.400			
32	207742.248			
19	299416.976			
26	271793.312			
60	251188.824			
55	500681.128			
9	218630.608			
96	310522.592			
17	234178.160			
59	251332.592			
57	395242.096			
41	217468.224			
64	302393.384			
45	300061.480			
97	383635.568			
8	331101.344			
71	181587.576			
94	262477.856			
90	251140.656			
98	225145.248			
86	154282.128			
80	180307.216			
50	225656.120			
52	258637.008			

```
70    276875.632
46    204302.976
68    294582.944
69    454512.760
81    408637.816
58    330677.128
33    191486.896
38    292965.216
51    393069.760
42    287350.000
4     458674.256
67    271726.752
39    245747.200
37    233493.208
20    268125.080
31    225452.320
63    334938.872
47    201778.048
85    376253.808
93    266684.248
49    262423.504
34    285223.176
7     175716.480
75    286161.600
82    190909.056
43    414682.648
22    412569.472
72    298926.496
15    440201.616
40    310045.712
Name: price, dtype: float64
83    282683.544
30    301635.728
56    320345.520
24    168047.264
16    248337.600
23    183459.488
2     281626.336
27    406852.304
28    297760.440
13    418753.008
99    274922.856
```

```
14    444192.008
0     234314.144
21    171795.240
3     401255.608
29    368988.432
61    263311.696
79    342988.456
35    302000.920
11    494778.992
84    303597.216
44    293044.496
73    211724.096
5     245050.280
25    362519.720
77    365863.936
74    228313.024
62    359674.440
65    304587.272
1     228581.528
18    225451.984
48    257828.416
36    269225.920
78    251560.040
6     265129.064
89    354512.112
91    338078.168
10    279555.096
Name: price, dtype: float64
```

```
In [72]: from sklearn import metrics
MAE=metrics.mean_absolute_error(y_test,y_test_predicted)
print(MAE)
MSE=metrics.mean_squared_error(y_test,y_test_predicted)
print(MSE)
RMSE=np.sqrt(metrics.mean_squared_error(y_test,y_test_predicted))
print(RMSE)
R2=metrics.r2_score(y_test,y_test_predicted)
print(R2)
```

23161.93802506976  
918151864.5451638  
30301.02084988497

```
In [73]: from sklearn.linear_model import LinearRegression
```

```
In [74]: equation=LinearRegression()
```

```
In [82]: equation.fit(x_train, y_train)
```

```
Out[82]: ▾ LinearRegression  
LinearRegression()
```

```
In [83]: equation.intercept_
```

```
Out[83]: -5778726.325474448
```

```
In [84]: equation.coef_
```

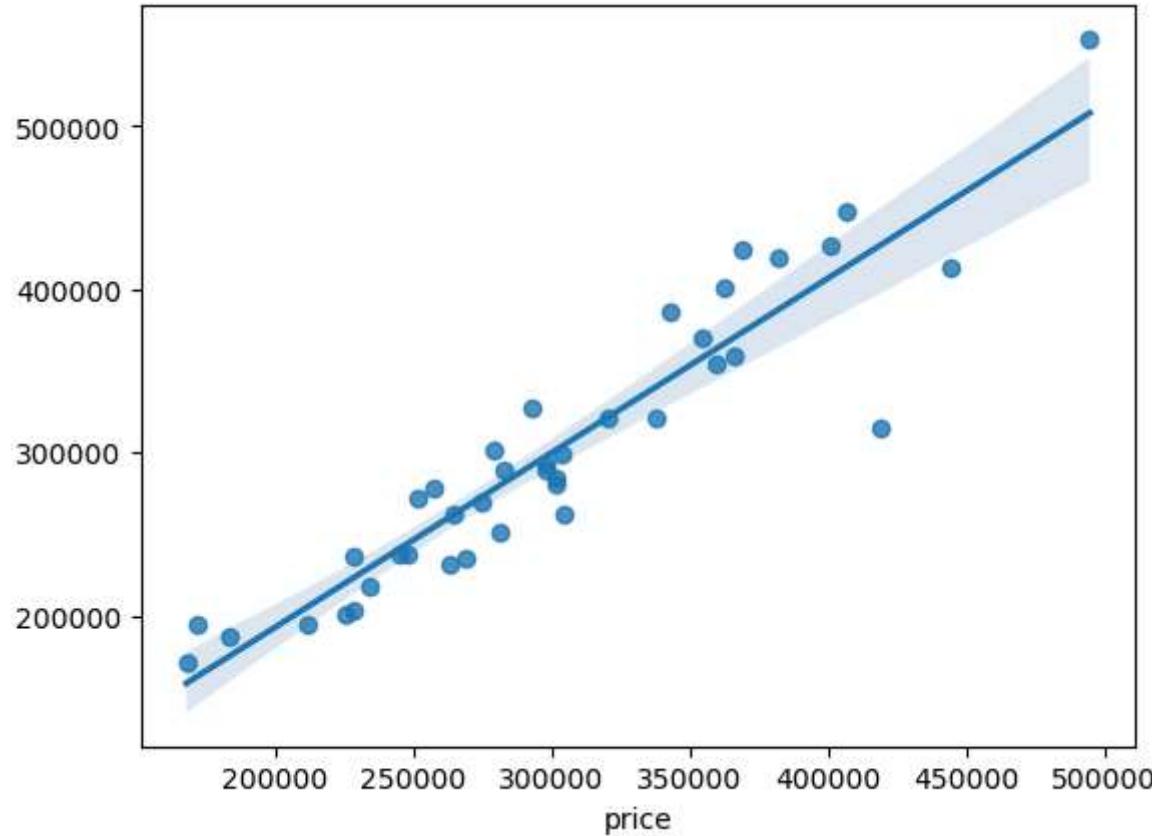
```
Out[84]: array([ 242.12526906, 2914.12242819, -31226.44185297, 31226.44185297])
```

```
In [85]: y_test_predicted=equation.predict(x_test)  
y_test_predicted
```

```
Out[85]: array([288907.51573793, 280403.2467037, 320634.09793863, 171831.07862285,  
237943.47604893, 187516.82244284, 251184.39881823, 447476.13062718,  
289361.06617231, 315111.22055134, 268998.23833529, 292617.65104118,  
419744.74738039, 413224.99735682, 217712.26474743, 195097.76461714,  
426341.92408649, 424310.49307906, 231128.4259061, 386024.64463052,  
284035.95532442, 553090.30410134, 299171.20589342, 327486.24305305,  
195124.39839674, 237497.74083046, 400335.98581247, 358796.14506756,  
236718.32218748, 353914.99235229, 262263.31425465, 203406.63496109,  
200681.08061269, 277768.10828967, 235096.76635702, 272164.46067337,  
262683.70402717, 370238.30181114, 321453.25752928, 301259.14119938])
```

```
In [86]: sns.regplot(x=y_test, y=y_test_predicted)
```

```
Out[86]: <Axes: xlabel='price'>
```



In [ ]:

In [ ]: