```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
%matplotlib inline
```

```
df=pd.read_csv("/content/data (1).csv")
df
```

	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	sqft_above	sqft_baseme
0	2014-05-02 00:00:00	3.130000e+05	3.0	1.50	1340	7912	1.5	0	0	3	1340	
1	2014-05-02 00:00:00	2.384000e+06	5.0	2.50	3650	9050	2.0	0	4	5	3370	2
2	2014-05-02 00:00:00	3.420000e+05	3.0	2.00	1930	11947	1.0	0	0	4	1930	
3	2014-05-02 00:00:00	4.200000e+05	3.0	2.25	2000	8030	1.0	0	0	4	1000	10
4	2014-05-02 00:00:00	5.500000e+05	4.0	2.50	1940	10500	1.0	0	0	4	1140	8
...	...	...	...	...	...	...	...	...	...	...	...	
4595	2014-07-09 00:00:00	3.081667e+05	3.0	1.75	1510	6360	1.0	0	0	4	1510	
4596	2014-07-09 00:00:00	5.343333e+05	3.0	2.50	1460	7573	2.0	0	0	3	1460	
4597	2014-07-09 00:00:00	4.169042e+05	3.0	2.50	3010	7014	2.0	0	0	3	3010	
4598	2014-07-10 00:00:00	2.034000e+05	4.0	2.00	2090	6630	1.0	0	0	3	1070	10
4599	2014-07-10 00:00:00	2.206000e+05	3.0	2.50	1490	8102	2.0	0	0	4	1490	

4600 rows × 18 columns

Далее: Создать код с переменной df | Посмотреть рекомендованные графики | New interactive sheet

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4600 entries, 0 to 4599
Data columns (total 18 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   date           4600 non-null   object
 1   price          4600 non-null   float64
 2   bedrooms       4600 non-null   float64
 3   bathrooms      4600 non-null   float64
 4   sqft_living    4600 non-null   int64
 5   sqft_lot       4600 non-null   int64
 6   floors         4600 non-null   float64
 7   waterfront     4600 non-null   int64
 8   view           4600 non-null   int64
 9   condition      4600 non-null   int64
 10  sqft_above     4600 non-null   int64
 11  sqft_basement  4600 non-null   int64
 12  yr_built       4600 non-null   int64
 13  yr_renovated   4600 non-null   int64
 14  street         4600 non-null   object
 15  city           4600 non-null   object
 16  statezip       4600 non-null   object
 17  country        4600 non-null   object
dtypes: float64(4), int64(9), object(5)
memory usage: 647.0+ KB
```

```
df["floors"].value_counts()
```

```
        count
floors
  1.0    2174
  2.0    1811
  1.5     444
  3.0     128
  2.5      41
  3.5       2
```

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

```
               0
date           0
price          0
bedrooms       0
bathrooms      0
sqft_living    0
sqft_lot       0
floors         0
waterfront     0
view           0
condition      0
sqft_above     0
sqft_basement  0
yr_built       0
yr_renovated   0
street         0
city           0
statezip       0
country        0
```

```
encoder = LabelEncoder()
df['bathrooms']=encoder.fit_transform(df['bedrooms'].values)
df['sqft_living']=encoder.fit_transform(df['sqft_living'].values)
df['floors']=encoder.fit_transform(df['floors'].values)
df['city']=encoder.fit_transform(df['city'].values)
df['country']=encoder.fit_transform(df['country'].values)
df['statezip']=encoder.fit_transform(df['statezip'].values)
df
```

	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	sqft_above	sqft_baseme
0	2014-05-02 00:00:00	3.130000e+05	3.0	3	93	7912	1	0	0	3	1340	
1	2014-05-02 00:00:00	2.384000e+06	5.0	5	406	9050	2	0	4	5	3370	2
2	2014-05-02 00:00:00	3.420000e+05	3.0	3	180	11947	0	0	0	4	1930	
3	2014-05-02 00:00:00	4.200000e+05	3.0	3	191	8030	0	0	0	4	1000	1C
4	2014-05-02 00:00:00	5.500000e+05	4.0	4	181	10500	0	0	0	4	1140	8
...	...	...	...	...	...	...	...	...	...	...	...	
4595	2014-07-09 00:00:00	3.081667e+05	3.0	3	124	6360	0	0	0	4	1510	
4596	2014-07-09 00:00:00	5.343333e+05	3.0	3	115	7573	2	0	0	3	1460	
4597	2014-07-09 00:00:00	4.169042e+05	3.0	3	333	7014	2	0	0	3	3010	
4598	2014-07-10 00:00:00	2.034000e+05	4.0	4	207	6630	0	0	0	3	1070	1C
4599	2014-07-10 00:00:00	2.206000e+05	3.0	3	120	8102	2	0	0	4	1490	

4600 rows × 18 columns

Далее: Создать код с переменной df | Посмотреть рекомендованные графики | New interactive sheet

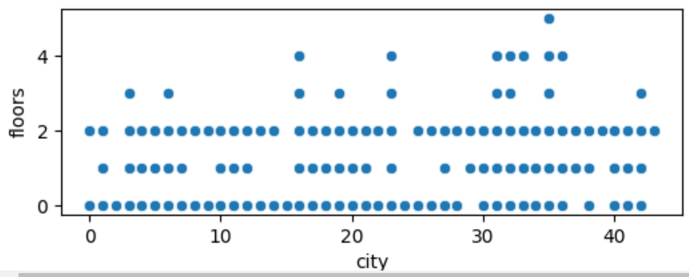
```
df2=df.drop(columns=["date","country","street","statezip"])
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4600 entries, 0 to 4599
Data columns (total 14 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   price          4600 non-null   float64
 1   bedrooms       4600 non-null   float64
 2   bathrooms      4600 non-null   int64  
 3   sqft_living    4600 non-null   int64  
 4   sqft_lot       4600 non-null   int64  
 5   floors         4600 non-null   int64  
 6   waterfront     4600 non-null   int64  
 7   view           4600 non-null   int64  
 8   condition      4600 non-null   int64  
 9   sqft_above     4600 non-null   int64  
 10  sqft_basement  4600 non-null   int64  
 11  yr_built       4600 non-null   int64  
 12  yr_renovated   4600 non-null   int64  
 13  city           4600 non-null   int64  
dtypes: float64(2), int64(12)
memory usage: 503.2 KB
```

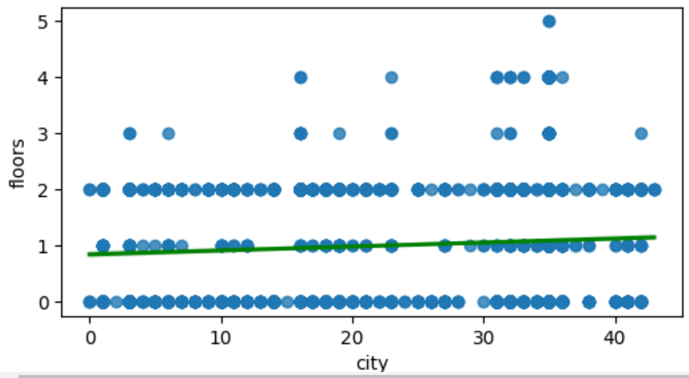
```
df2.corrwith(df2["floors"])
```

	0
price	0.151461
bedrooms	0.177895
bathrooms	0.177895
sqft_living	0.358055
sqft_lot	0.003750
floors	1.000000
waterfront	0.022024
view	0.031211
condition	-0.275013
sqft_above	0.522814
sqft_basement	-0.255510
yr_built	0.467481
yr_renovated	-0.233996
city	0.078481

```
plt.figure(figsize=(6,2))
sns.scatterplot(data=df, x='city', y='floors')
plt.show()
```



```
plt.figure(figsize=(6,3))
sns.regplot(data=df, x='city', y='floors',line_kws={"color":"green"})
plt.show()
```



```
X=df2["floors"].to_numpy()
X
```

```
array([1, 2, 0, ..., 2, 0, 2])
```

```
Y=df2["city"].to_numpy()
Y
```

```
array([36, 35, 18, ..., 32, 35,  9])
```

```
from sklearn.model_selection import train_test_split
trai_et, test_set = train_test_split(df2,test_size=0.20, random_state=42)
```

```
test_set.shape
```

```
(920, 14)
```

```
from sklearn import linear_model
LR_model = linear_model.LinearRegression()
```

```
Xmean=X.mean()
Xmean
```

```
1.0241304347826088
```

```
Ymean=Y.mean()
Ymean
```

```
25.674347826086958
```

```
theta1=sum((X-Xmean)*(Y-Ymean))/sum((X-Xmean)**2)
print(f"{theta1=}")
```

```
theta1=0.8735209259544456
```

```
theta0=Ymean-theta1*Xmean
print(f"{theta0=}")
```

```
theta0=24.779748460397524
```

```
x_test=df2.sample(52,random_state=22)["floors"].to_numpy()
print(x_test)
y_test=df2.sample(52,random_state=22)["city"].to_numpy()
print(y_test)
```

```
[2 0 0 0 0 2 0 0 0 2 0 2 2 0 2 2 2 2 1 2 2 2 0 0 2 1 2 0 1 0 0 2 2 2 2 2 1
 2 0 2 2 2 2 2 0 0 0 0 0 2 0 2]
[31 18 36 19 35 21 31 32 19 32 35 33 35 32 31 35 25 19 17 18 25 38 35  3
 14 35 19 36 27 35 42 35 19 32 27 32 35 35  7 32 38 16 35 21 35  3 19 14
 14 38 35 35]
```

```
y_pridect=theta0 +theta1*x_test

arr1 = np.round(y_pridect,decimals = 2)
print(arr1)
```

```
[26.53 24.78 24.78 24.78 24.78 26.53 24.78 24.78 24.78 26.53 24.78 26.53
 26.53 24.78 26.53 26.53 26.53 26.53 25.65 26.53 26.53 26.53 24.78 24.78
 26.53 25.65 26.53 24.78 25.65 24.78 24.78 26.53 26.53 26.53 26.53 26.53
 25.65 26.53 24.78 26.53 26.53 26.53 26.53 26.53 24.78 24.78 24.78 24.78
 24.78 26.53 24.78 26.53]
```

```
date={
    "Asil qiymatlar": y_test,
    "Bashorat qiymatlar":arr1
}
df1=pd.DataFrame(date)
df1
```

	Asil qiymatlar	Bashorat qiymatlar
0	31	26.53
1	18	24.78
2	36	24.78
3	19	24.78
4	35	24.78
5	21	26.53
6	31	24.78
7	32	24.78
8	19	24.78
9	32	26.53
10	35	24.78
11	33	26.53
12	35	26.53
13	32	24.78
14	31	26.53
15	35	26.53
16	25	26.53
17	19	26.53
18	17	25.65
19	18	26.53
20	25	26.53
21	38	26.53
22	35	24.78
23	3	24.78
24	14	26.53
25	35	25.65
26	19	26.53
27	36	24.78
28	27	25.65
29	35	24.78
30	42	24.78
31	35	26.53
32	19	26.53
33	32	26.53
34	27	26.53
35	32	26.53
36	35	25.65
37	35	26.53
38	7	24.78
39	32	26.53
40	38	26.53
41	16	26.53

Дал Создать код с переменной df1 Посмотреть рекомендованные графики New interactive sheet

```
MAE = np.sum(np.absolute(arr1-y_test))/len(y_test)
print(f"{MAE=}")
MAE = np.sum(np.absolute(arr1-y_test))/len(y_test)
MAE
```

```
MAE=8.627884615384618
8.627884615384618
```