In [1]:
```python
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```python
housing = pd.read_csv('C:\\Users\91820\\Downloads\\original.CSV')
```

In [3]:
```python
housing
```

Out[3]:

	price	crime_rate	resid_area	air_qual	room_num	age	dist1	dist2	dist3	dist4	teache
0	24.0	0.00632	32.31	0.538	6.575	65.2	4.35	3.81	4.18	4.01	24
1	21.6	0.02731	37.07	0.469	6.421	78.9	4.99	4.70	5.12	5.06	22
2	34.7	0.02729	37.07	0.469	7.185	61.1	5.03	4.86	5.01	4.97	22
3	33.4	0.03237	32.18	0.458	6.998	45.8	6.21	5.93	6.16	5.96	21
4	36.2	0.06905	32.18	0.458	7.147	54.2	6.16	5.86	6.37	5.86	21
...	...	...	...	...	...	...	...	...	...	...	
501	22.4	0.06263	41.93	0.573	6.593	69.1	2.64	2.45	2.76	2.06	19
502	20.6	0.04527	41.93	0.573	6.120	76.7	2.44	2.11	2.46	2.14	19
503	23.9	0.06076	41.93	0.573	6.976	91.0	2.34	2.06	2.29	1.98	19
504	22.0	0.10959	41.93	0.573	6.794	89.3	2.54	2.31	2.40	2.31	19
505	19.0	0.04741	41.93	0.573	6.030	80.8	2.72	2.24	2.64	2.42	19

506 rows × 19 columns

In [4]:
```python
housing.describe()
```

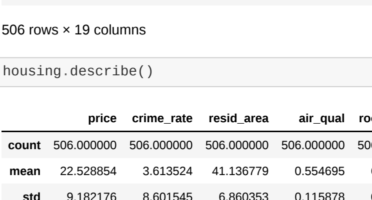
Out[4]:

	price	crime_rate	resid_area	air_qual	room_num	age	dist1
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000
mean	22.528854	3.613524	41.136779	0.554695	6.284634	68.574901	3.971996
std	9.182176	8.601545	6.860353	0.115878	0.702617	28.148861	2.108532
min	5.000000	0.006320	30.460000	0.385000	3.561000	2.900000	1.130000
25%	17.025000	0.082045	35.190000	0.449000	5.885500	45.025000	2.270000
50%	21.200000	0.256510	39.690000	0.538000	6.208500	77.500000	3.385000
75%	25.000000	3.677082	48.100000	0.624000	6.623500	94.075000	5.367500
max	50.000000	88.976200	57.740000	0.871000	8.780000	100.000000	12.320000

In [5]:
```python
# observation: missing values in "n_hos_beds"
```

In [6]:
```python
plt.scatter(x='n_hot_rooms',y='price',data=housing)
```

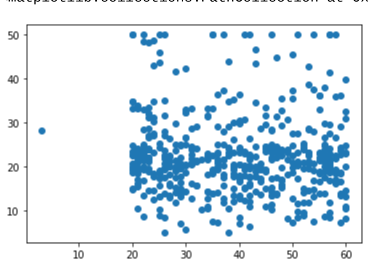
Out[6]: <matplotlib.collections.PathCollection at 0x1535c943048>



In [7]:
```python
# observation: two outliers
```

In [8]:
```python
plt.scatter(x='rainfall',y='price',data=housing)
```

Out[8]: <matplotlib.collections.PathCollection at 0x1535cd33848>



In [9]:
```python
# observation: single outlier
```

In [10]:
```python
housing.head()
```

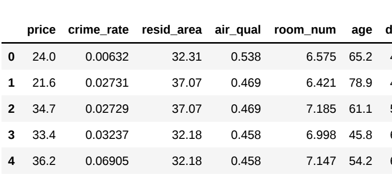
Out[10]:

	price	crime_rate	resid_area	air_qual	room_num	age	dist1	dist2	dist3	dist4	teachers
0	24.0	0.00632	32.31	0.538	6.575	65.2	4.35	3.81	4.18	4.01	24.7
1	21.6	0.02731	37.07	0.469	6.421	78.9	4.99	4.70	5.12	5.06	22.2
2	34.7	0.02729	37.07	0.469	7.185	61.1	5.03	4.86	5.01	4.97	22.2
3	33.4	0.03237	32.18	0.458	6.998	45.8	6.21	5.93	6.16	5.96	21.3
4	36.2	0.06905	32.18	0.458	7.147	54.2	6.16	5.86	6.37	5.86	21.3

In [11]:
```python
import seaborn as sns
```

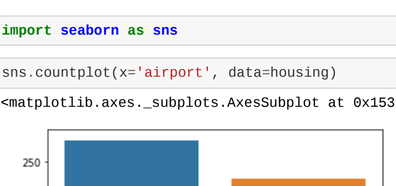
In [12]:
```python
sns.countplot(x='airport', data=housing)
```

Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x1535ed7d048>



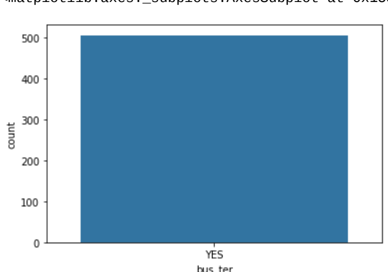
In [13]:
```python
sns.countplot(x='waterbody', data=housing)
```

Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x1535edef488>



In [14]:
```python
sns.countplot(x='bus_ter', data=housing)
```

Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x1535ee53148>



In [15]:
```python
# observation: all values are "yes", the data is not going to effect predictions.
```

In [16]:
```python
housing.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 19 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   price        506 non-null    float64
 1   crime_rate   506 non-null    float64
 2   resid_area   506 non-null    float64
 3   air_qual     506 non-null    float64
 4   room_num     506 non-null    float64
 5   age          506 non-null    float64
 6   dist1        506 non-null    float64
 7   dist2        506 non-null    float64
 8   dist3        506 non-null    float64
 9   dist4        506 non-null    float64
 10  teachers     506 non-null    float64
 11  poor_prop    506 non-null    float64
 12  airport      506 non-null    object
 13  n_hos_beds   498 non-null    float64
 14  n_hot_rooms  506 non-null    float64
 15  waterbody    506 non-null    object
 16  rainfall     506 non-null    int64
 17  bus_ter      506 non-null    object
 18  parks        506 non-null    float64
dtypes: float64(15), int64(1), object(3)
memory usage: 75.2+ KB
```

In [17]:
```python
ul = np.percentile(housing.n_hot_rooms,[99])[0]
```

In [18]:
```python
ul
```

Out[18]: 15.399519999999999

In [19]:
```python
housing[(housing.n_hot_rooms>ul)]
```

Out[19]:

	price	crime_rate	resid_area	air_qual	room_num	age	dist1	dist2	dist3	dist4	teache
2	34.7	0.02729	37.07	0.4690	7.185	61.1	5.03	4.86	5.01	4.97	22
166	50.0	2.01019	49.58	0.6050	7.929	96.2	2.11	1.91	2.31	1.86	25
204	50.0	0.02009	32.68	0.4161	8.034	31.9	5.41	4.80	5.28	4.99	25
267	50.0	0.57834	33.97	0.5750	8.297	67.0	2.60	2.13	2.43	2.52	27
369	50.0	5.66998	48.10	0.6310	6.683	96.8	1.55	1.28	1.65	0.94	19
423	13.4	7.05042	48.10	0.6140	6.103	85.1	2.08	1.80	2.34	1.87	19

In [20]:
```python
housing.n_hot_rooms[(housing.n_hot_rooms>3*ul)] = 3*ul
```

```
C:\Users\91820\anaconda3\envs\tensorflow\lib\site-packages\ipykernel_l
auncher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas
-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """Entry point for launching an IPython kernel.
```

In [21]:
```python
uv = np.percentile(housing.rainfall,[1])[0]
```

In [22]:
```python
uv
```

Out[22]: 20.0

In [23]:
```python
housing[(housing.rainfall<uv)]
```

Out[23]:

	price	crime_rate	resid_area	air_qual	room_num	age	dist1	dist2	dist3	dist4	teache
213	28.1	0.14052	40.59	0.489	6.375	32.3	4.11	3.92	4.18	3.57	21

In [24]:
```python
housing.rainfall[(housing.rainfall<uv)]=0.3*uv
```

```
C:\Users\91820\anaconda3\envs\tensorflow\lib\site-packages\ipykernel_l
auncher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas
-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """Entry point for launching an IPython kernel.
```

In [25]:
```python
housing[(housing.rainfall<uv)]
```

Out[25]:

	price	crime_rate	resid_area	air_qual	room_num	age	dist1	dist2	dist3	dist4	teache
213	28.1	0.14052	40.59	0.489	6.375	32.3	4.11	3.92	4.18	3.57	21

In [26]:
```python
housing.n_hos_beds=housing.n_hos_beds.fillna(housing.n_hos_beds.mean
())
```

In [27]:
```python
housing.info()
```

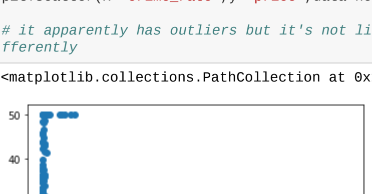
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 19 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   price        506 non-null    float64
 1   crime_rate   506 non-null    float64
 2   resid_area   506 non-null    float64
 3   air_qual     506 non-null    float64
 4   room_num     506 non-null    float64
 5   age          506 non-null    float64
 6   dist1        506 non-null    float64
 7   dist2        506 non-null    float64
 8   dist3        506 non-null    float64
 9   dist4        506 non-null    float64
 10  teachers     506 non-null    float64
 11  poor_prop    506 non-null    float64
 12  airport      506 non-null    object
 13  n_hos_beds   506 non-null    float64
 14  n_hot_rooms  506 non-null    float64
 15  waterbody    506 non-null    object
 16  rainfall     506 non-null    int64
 17  bus_ter      506 non-null    object
 18  parks        506 non-null    float64
dtypes: float64(15), int64(1), object(3)
memory usage: 75.2+ KB
```

In [28]:
```python
# fixed missing values
```

In [29]:
```python
plt.scatter(x='crime_rate',y='price',data=housing)

# it apparently has outliers but it's not linear so will be treated di
fferently
```

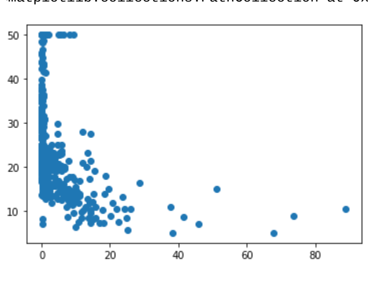
Out[29]: <matplotlib.collections.PathCollection at 0x1535ef06248>



In [30]:
```python
housing.crime_rate = np.log(1+housing.crime_rate)
```

In [31]:
```python
plt.scatter(x='crime_rate',y='price',data=housing)
```

Out[31]: <matplotlib.collections.PathCollection at 0x1535ef5bc88>



In [32]:
```python
housing['avg_dist']= (housing.dist1+housing.dist2+housing.dist3+housin
g.dist4)/4
```

In [33]:
```python
del housing['dist1']
```

In [34]:
```python
del housing['dist2']
```

In [35]:
```python
del housing['dist3']
```

In [36]:
```python
del housing['dist4']
```

In [37]:
```python
housing.describe()
```

Out[37]:

	price	crime_rate	resid_area	air_qual	room_num	age	teachers
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000
mean	22.528854	0.813418	41.136779	0.554695	6.284634	68.574901	21.544466
std	9.182176	1.022731	6.860353	0.115878	0.702617	28.148861	2.164946
min	5.000000	0.006300	30.460000	0.385000	3.561000	2.900000	18.000000
25%	17.025000	0.078853	35.190000	0.449000	5.885500	45.025000	19.800000
50%	21.200000	0.228336	39.690000	0.538000	6.208500	77.500000	20.950000
75%	25.000000	1.542674	48.100000	0.624000	6.623500	94.075000	22.600000
max	50.000000	4.499545	57.740000	0.871000	8.780000	100.000000	27.400000

In [38]:
```python
#all locations have bus terminal no effect on prediction so
```

In [39]:
```python
del housing['bus_ter']
```

In [40]:
```python
housing.head()
```

Out[40]:

	price	crime_rate	resid_area	air_qual	room_num	age	teachers	poor_prop	airport
0	24.0	0.006300	32.31	0.538	6.575	65.2	24.7	4.98	YES
1	21.6	0.026944	37.07	0.469	6.421	78.9	22.2	9.14	NO
2	34.7	0.026924	37.07	0.469	7.185	61.1	22.2	4.03	NO
3	33.4	0.031857	32.18	0.458	6.998	45.8	21.3	2.94	YES
4	36.2	0.066770	32.18	0.458	7.147	54.2	21.3	5.33	NO

In [41]:
```python
housing = pd.get_dummies(housing)
```

In [42]:
```python
housing.head()
```

Out[42]:

	price	crime_rate	resid_area	air_qual	room_num	age	teachers	poor_prop	n_hos_beds
0	24.0	0.006300	32.31	0.538	6.575	65.2	24.7	4.98	5.480
1	21.6	0.026944	37.07	0.469	6.421	78.9	22.2	9.14	7.332
2	34.7	0.026924	37.07	0.469	7.185	61.1	22.2	4.03	7.394
3	33.4	0.031857	32.18	0.458	6.998	45.8	21.3	2.94	9.268
4	36.2	0.066770	32.18	0.458	7.147	54.2	21.3	5.33	8.824

In [43]:
```python
del housing['airport_NO']
```

In [44]:
```python
del housing['waterbody_None']
```

In [45]:
```python
# dealing with categorical data
```

In [46]:
```python
housing.corr()
```

Out[46]:

	price	crime_rate	resid_area	air_qual	room_num	age	teachers
price	1.000000	-0.466527	-0.484754	-0.429300	0.696304	-0.377999	0.505655
crime_rate	-0.466527	1.000000	0.660283	0.707587	-0.288784	0.559591	-0.390052
resid_area	-0.484754	0.660283	1.000000	0.763651	-0.391676	0.644779	-0.383248
air_qual	-0.429300	0.707587	0.763651	1.000000	-0.302188	0.731470	-0.188933
room_num	0.696304	-0.288784	-0.391676	-0.302188	1.000000	-0.240265	0.355501
age	-0.377999	0.559591	0.644779	0.731470	-0.240265	1.000000	-0.261515
teachers	0.505655	-0.390052	-0.383248	-0.188933	0.355501	-0.261515	1.000000
poor_prop	-0.740836	0.608970	0.603800	0.590879	-0.613808	0.602339	-0.374044
n_hos_beds	0.108880	-0.004089	0.005799	-0.049553	0.032009	-0.021012	-0.008056
n_hot_rooms	0.017007	0.056570	-0.003761	0.007238	0.014583	0.013918	-0.037007
rainfall	-0.047200	0.082151	0.055845	0.091956	-0.064718	0.074684	-0.045928
parks	-0.391574	0.638951	0.707635	0.915544	-0.282817	0.673850	-0.187004
avg_dist	0.249289	-0.586371	-0.708022	-0.769247	0.205241	-0.747906	0.232452
airport_YES	0.182867	-0.134486	-0.115401	-0.073903	0.163774	0.005101	0.069437
waterbody_Lake	0.036233	-0.025390	-0.026590	-0.046393	-0.004195	0.003452	0.048717
waterbody_Lake and River	-0.037497	0.009076	0.051649	0.013849	0.010554	-0.004354	-0.046981
waterbody_River	0.071751	-0.060099	-0.098976	-0.037772	0.046251	-0.088609	0.094256

In [47]:
```python
del housing['parks']
```

In [50]:
```python
from sklearn.linear_model import LinearRegression
```

In [51]:
```python
y = housing['price']
```

In [52]:
```python
X = housing[['room_num']]
```

In [53]:
```python
lm = LinearRegression()
```

In [54]:
```python
lm.fit(X,y)
```

Out[54]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normali
ze=False)
```

In [55]:
```python
print(lm.intercept_,lm.coef_)
```

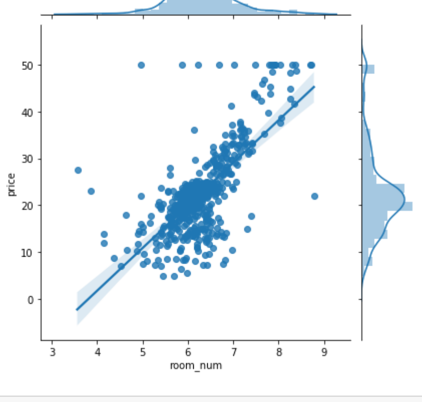
```
-34.65924312309721 [9.09966966]
```

In [56]:
```python
lm.predict(X)
```

Out[56]:
```
array([25.17108491, 23.76973578, 30.72188341, 29.02024518, 30.3760959
6,
       23.85163281, 20.04797089, 21.50391804, 16.58099675, 19.9751735
3,
       23.36935032, 20.02067188, 18.92871152, 19.4746917 , 20.8123431
4,
       18.42822969, 19.34729633, 19.84777816, 14.98855456, 17.4545650
4,
       16.0259169 , 19.62028642, 21.23092795, 18.23713663, 19.2471999
6,
       16.28980732, 18.23713663, 20.36645933, 24.44311134, 26.0719522
1,
       17.32716966, 20.59395107, 19.48379137, 17.21797363, 20.8123431
4,
       19.32909699, 18.49192738, 18.57382441, 19.62938609, 25.3530783
,
       29.25683659, 26.9455205 , 21.47661903, 21.85880515, 20.5666520
6,
       17.0450799 , 17.99144555, 20.21176495, 14.46987339, 16.3171063
3,
       19.60208708, 20.98523687, 24.58870605, 19.92057552, 18.9196118
5,
       31.30426226, 23.42394834, 27.3641053 , 21.25822696, 19.2744989
7,
       17.58196041, 19.62938609, 24.08822422, 26.87272314, 29.9848101
6,
       22.57767906, 18.00054522, 18.82861516, 16.24430897, 18.8923128
4,
       23.7333371 , 19.58388774, 20.53025338, 22.16819392, 22.4229846
7,
       22.54128038, 22.47758269, 21.21272861, 22.04989822, 18.7922164
8,
       26.5542347 , 25.57147038, 22.68687509, 21.45841969, 23.4785463
5,
       25.67156674, 20.0752699 , 21.03983488, 29.10214221, 29.7573184
2,
       23.7333371 , 23.62414107, 23.96082885, 21.85880515, 22.2045926
,
       25.62606839, 21.42202101, 38.76599139, 36.50017364, 32.8239071
,
       26.5542347 , 27.04561686, 23.62414107, 21.1854296 , 21.4584196
9,
       18.58292408, 18.44642903, 21.0944329 , 24.25201828, 22.0225992
1,
       21.71321044, 26.44503866, 19.14710359, 20.77594446, 22.2500909
5,
       19.28359864, 21.54031672, 20.12986792, 18.77401714, 17.4909637
2,
       18.7558178 , 19.97517353, 19.58388774, 18.62842242, 18.8377148
3,
       19.81137948, 16.4172027 , 17.14517627, 23.86073248, 16.6355947
7,
       24.10642356, 22.90526717, 23.32385197, 18.31903366, 17.7275551
3,
       22.98716419, 19.41099401, 24.07002488, 18.63752209, 21.3128249
7,
       21.52211738, 11.01199892, 14.50627207, 15.09775059,  9.9564372
3,
       21.12173191, 16.55369774, 10.16572964, 12.53164375, 16.2716079
8,
       21.04893455, 14.51537174, 10.94830123, 17.29077098, 21.1126322
4,
       21.32192464, 13.31421534, 28.51976335, 20.53935305, 24.5796063
8,
       22.21369227, 33.48818298, 36.33637959, 41.55049031, 18.6102230
8,
       20.85784149, 37.49203764, 18.81951549, 22.84156948, 23.5968420
6,
       18.80131615, 18.8468145 , 16.04411624, 23.72423744, 18.6557214
3,
       24.90719449, 20.12076825, 22.8051708 , 27.76449077, 28.8564511
3,
       35.99969181, 21.24912729, 30.44889332, 25.06188888, 16.3353056
7,
       21.33102431, 36.60027001, 27.05471653, 24.99819119, 30.7218834
1,
       28.5925607 , 26.66343074, 30.65818572, 27.21851059, 25.4349753
3,
       37.00065547, 31.65004971, 30.01210917, 31.53175401, 28.8109527
8,
       30.26689992, 21.41292134, 34.58924301, 36.80046274, 38.4475029
5,
       18.94691086, 22.90526717, 17.96414654, 20.52115371, 13.9693915
6,
       19.57478807, 14.51537174, 18.18253861, 23.35115098, 14.5881690
9,
       21.59491473, 18.91961185, 25.78076278, 19.49289104, 23.3329516
4,
       28.5925607 , 21.43112068, 27.93738449, 25.56237071, 40.5586263
1,
       44.73537469, 38.50210097, 30.52169067, 35.28081791, 24.9617925
1,
       19.76588113, 32.78750842, 41.20470286, 40.38573259, 26.5451350
3,
       20.72134645, 25.68066641, 32.29612626, 24.31571596, 25.4531746
7,
       28.10117854, 20.80324347, 23.19645659, 23.51494503, 16.2352093
,
       16.34440534, 20.92153918, 21.9953002 , 23.87893182, 26.4723376
7,
       24.37031398, 23.92443017, 28.64715872, 40.49492862, 20.9215391
8,
       18.81041582, 33.16969455, 44.54428162, 32.06863452, 27.6006967
1,
       30.88567746, 33.77027274, 41.75978271, 32.0140365 , 30.9129764
7,
       15.9349202 , 29.16583989, 40.84071607, 33.31528926, 19.2108012
8,
       18.62842242, 22.12269557, 24.83439713, 35.32631626, 26.8363244
6,
       27.70989275, 31.46805632, 27.455102  , 24.32481563, 27.3277066
2,
       36.50017364, 28.74725509, 34.90773145, 37.43743962, 29.8392154
5,
       24.06092521, 22.03169888, 21.84060581, 22.8051708 , 25.0800882
1,
       27.77359044, 30.38519563, 25.67156674, 21.0944329 , 20.0206718
8,
       26.10835089, 24.9344935 , 18.02784423, 23.07816089, 29.4115309
7,
       27.86458713, 25.30757996, 24.44311134, 28.87465046, 31.1859665
6,
       25.54417137, 32.86030578, 27.6643944 , 25.71706509, 19.6839841
,
       10.59341411, 21.04893455, 20.14806726, 22.35928699, 25.0982875
5,
       17.2543723 , 19.15620326, 17.95504687, 23.41484867, 20.9670375
3,
       23.81523413, 23.36025065, 20.31186131, 17.28167131, 23.7151377
7,
       23.86073248, 22.77787179, 20.69404744, 18.73761846, 22.9689648
5,
       21.24912729, 17.26347197, 20.22086461, 22.81427047, 22.7596724
5,
       20.27546263, 18.74671813, 18.98330954, 20.47565537, 19.8022798
1,
       19.64758543, 31.23146491, 24.85259647, 26.27214494, 27.8918861
4,
       20.06617023, 19.01060855, 24.6342044 , 25.71706509, 28.4833646
7,
       24.39761299, 25.20748359, 18.88321317, 26.56333437, 16.8721861
8,
       19.356396  , 21.86790482, 23.53314437, 21.0944329 , 20.9579378
6,
       23.56044338, 22.22279194, 14.13318561, 18.14613993, 45.2358565
2,
       -2.25531945, 10.50241741,  0.49278079, 10.5661151 , 26.1538492
4,
       29.18403923, 21.9043035 , 18.80131615,  9.98373624,  2.9951899
4,
       31.88664112, 25.84446047, 27.16391257, 23.39664933, 21.9680011
9,
       28.74725509, 24.89809482, 15.71652813, 15.57093342,  5.0881139
7,
       13.35971369,  7.67242015, 10.83910519,  9.74714483, 14.3879763
6,
       17.32716966, 20.40285801, 11.1666933 , 21.6950111 , 18.9105121
8,
       24.22471927, 23.62414107, 17.63655843, 14.96125555, 18.5920237
5,
       19.82047915, 23.05996155, 23.6150414 , 14.0148899 , 15.6710297
8,
       17.05417957,  2.99518994, 16.37170435, 16.45360137, 27.6916934
1,
       17.72755513, 25.91725782,  7.45402808, 12.24955399,  6.4621640
8,
       23.88803149, 27.05471653, 13.60540477, 19.54748906, 27.4369026
6,
       23.67873909, 19.99337287, 16.73569113, 20.87604083, 15.9804185
5,
       18.99240921, 18.4555287 , 21.77690813, 21.6950111 , 23.3966493
3,
       23.1054599 , 27.51879968, 23.80613446, 23.90623083, 21.8315061
5,
       25.66246707, 24.13372257, 21.32192464, 19.34729633, 16.5445980
7,
       18.28263498, 23.63324074, 21.93160251, 24.35211464, 18.6102230
8,
       24.11552323, 23.04176221, 22.22279194, 21.62221374, 23.7333371
,
       26.75442743, 25.89905848, 22.64137675, 32.6146147 , 26.5633343
7,
       24.71610143, 19.72038278, 19.356396  , 22.67777542, 20.6758481
,
       26.31764329, 23.36025065, 22.82337014, 24.60690539, 21.8406058
1,
       17.74575447, 19.50199071, 19.96607386, 19.2653993 , 17.3271696
6,
       21.45841969, 22.02259921, 23.9153305 , 28.85645113, 14.7246641
4,
       21.41292134, 24.34301497, 13.60540477, 21.62221374, 22.0225992
1,
       22.14089491, 26.7635271 , 29.59352437, 17.77305348, 18.7649174
7,
       22.77787179, 20.9761372 , 19.07430624, 14.97035522, 14.6063684
3,
       11.68537447, 19.78408047, 19.78408047, 17.27257164, 19.2653993
,
       16.93588387, 14.38797636, 18.0642429 , 20.11166858, 16.0168172
3,
       20.18446594, 25.33487897, 21.03073521, 28.82005245, 27.1639125
7,
       20.21176495])
```

In [58]:
```python
sns.jointplot(x= housing['room_num'], y=housing['price'], data = housi
ng, kind='reg')
```

Out[58]: <seaborn.axisgrid.JointGrid at 0x1535eec1a08>



In [ ]: