

# Statistics.

```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import math
import scipy.stats
```

```
In [2]: import os
os.chdir(r'D:\Sagun Shakya\Python\Data Sets\article-resources-master\From Good to Great Da
```

```
In [3]: hospital = pd.read_csv('hospital_ratings.csv', index_col = 'Unnamed: 0')
hospital.head(2)

# Each row represents a hospital.
```

Out[3]:

	mortality_rate	quality_rating
0	average	1
1	below_average	4

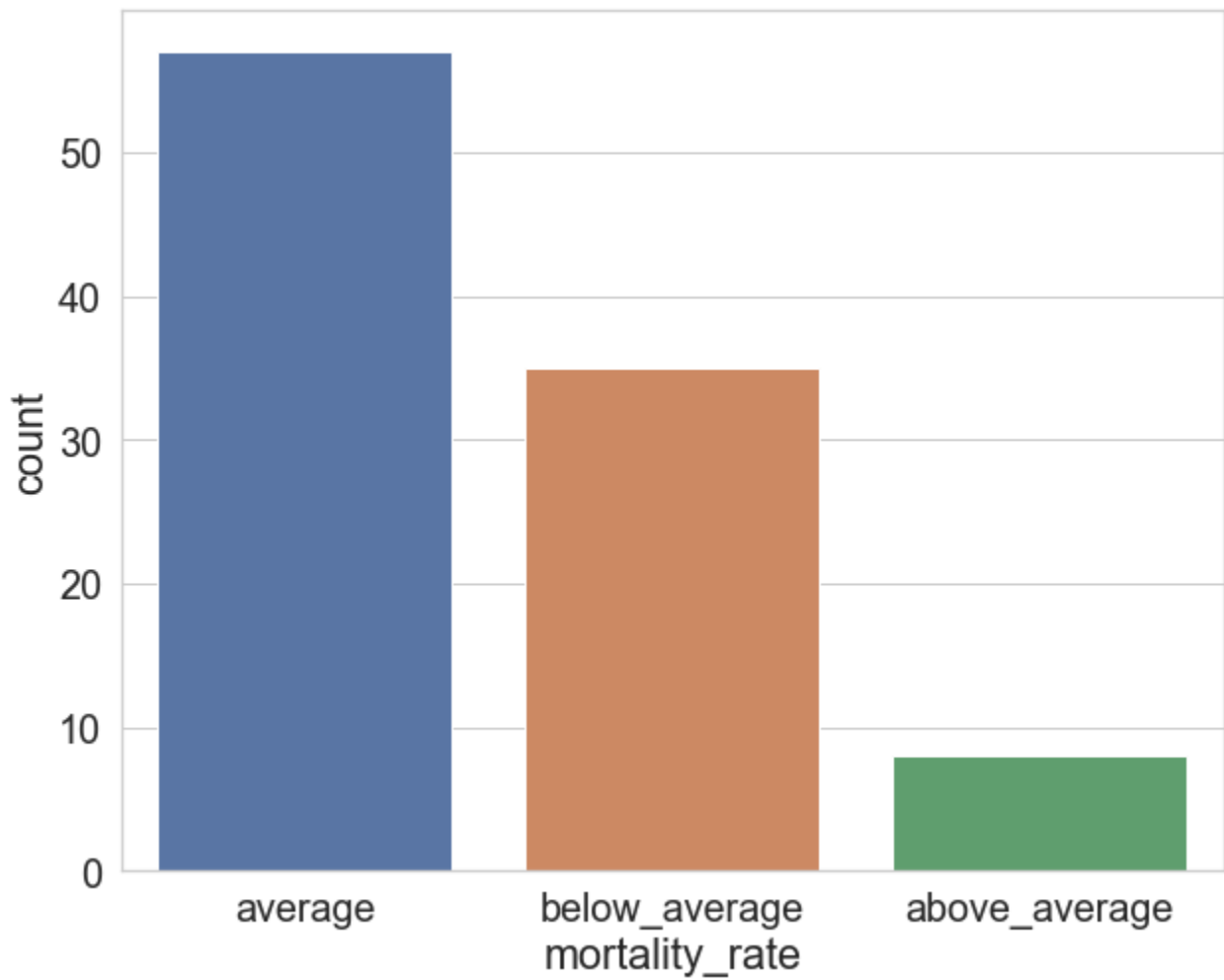
```
In [4]: hospital.isnull().sum()
```

```
Out[4]: mortality_rate    0
quality_rating          0
dtype: int64
```

```
In [5]: hospital['mortality_rate'].value_counts()
```

```
Out[5]: average          57
below_average          35
above_average           8
Name: mortality_rate, dtype: int64
```

```
In [17]: plt.figure( figsize = (10,8) )  
sns.set(style = 'whitegrid', font_scale = 1.8)  
  
sns.countplot(hospital['mortality_rate'])  
plt.show()
```



**Encoding the ordinal data called 'mortality\_rate' as follows:**

- below\_average: +1

- average: 0
- above\_average: -1

```
In [7]: mortality_map = dict()
mortality_map['below_average'] = 1
mortality_map['average']=0
mortality_map['above_average'] = -1
```

```
In [8]: hospital['mortality_rate_coded'] = hospital['mortality_rate'].map(mortality_map)
```

```
In [9]: hospital['mortality_rate_coded'].value_counts()
```

```
Out[9]: 0    57
        1    35
        -1    8
        Name: mortality_rate_coded, dtype: int64
```

```
In [10]: hospital.head()
```

```
Out[10]:
```

	mortality_rate	quality_rating	mortality_rate_coded
0	average	1	0
1	below_average	4	1
2	above_average	1	-1
3	average	4	0
4	average	4	0

## Pearson's correlation coefficient.

```
In [33]: from scipy.stats import pearsonr

x = pearsonr(hospital['quality_rating'], hospital['mortality_rate_coded'])
x
```

```
Out[33]: (0.47371205555642815, 6.426261109600425e-07)
```

```
In [34]: Pearson,_ = pearsonr(hospital['quality_rating'], hospital['mortality_rate_coded'])

Pearson
```

```
Out[34]: 0.47371205555642815
```

```
In [31]: hospital.corr(method = 'pearson')
```

```
Out[31]:
```

	quality_rating	mortality_rate_coded
quality_rating	1.000000	0.473712
mortality_rate_coded	0.473712	1.000000

However, this is not applicable for ordinal data. We need to use non-parametric measure like

## Spearman's Correlation coefficient.

### Spearman's Correlation:

- The calculation of Spearman's correlation coefficient and subsequent significance testing of it requires the following data assumptions to hold:

- interval or ratio level or ordinal;
- monotonically related.

- Note: unlike Pearson's correlation, there is no requirement of normality and hence it is a nonparametric statistic.
- Pearson's Correlation is sensitive to outliers as well as skewness.

```
In [18]: hospital.corr(method = 'spearman')
```

```
In [27]: from scipy.stats import spearmanr
a= spearmanr(hospital['quality_rating'], hospital['mortality_rate_coded'])
```

```
In [28]: a
```

```
Out[28]: SpearmanrResult(correlation=0.37909763025792015, pvalue=0.00010043425124583103)
```

```
In [30]: Spearman,_ = spearmanr(hospital['quality_rating'], hospital['mortality_rate_coded'])
Spearman
```

```
Out[30]: 0.37909763025792015
```

```
In [38]: PercentDiff = ((np.abs(Pearson - Spearman))/ Pearson) * 100
PercentDiff
```

```
Out[38]: 19.972982361061657
```

Had you reported this correlation using pearson as opposed to spearman, you might have seriously mislead a customer or coworker.

## Opiod Prescription Problem.

### Problem Description:

- prescriber\_id: A random code that identifies pharmacists.
- num\_opioid\_prescriptions: The number of opioid prescriptions prescribed on a given day.
- num\_prescriptions: The number of total prescriptions prescribed on a given day.

```
In [42]: prescribe = pd.read_csv(r'prescriptions.csv', index_col = 'Unnamed: 0')
prescribe.head()
```

Out[42]:

	prescriber_id	num_opioid_prescriptions	num_prescriptions
0	BDRSDDDB2&RA	4	7
1	3\$N3WG4&BBO	25	28
2	S2NGGOGBRB4	1	1
3	S2NGGOGBRB4	18	22
4	NBORABOESOI	1	1

```
In [44]: prescribe.shape
```

Out[44]: (150, 3)

```
In [46]: prescribe.dropna().head(2)
```

Out[46]:

	prescriber_id	num_opioid_prescriptions	num_prescriptions
0	BDRSDDDB2&RA	4	7
1	3\$N3WG4&BBO	25	28

**Inspecting the number of people who are over prescribing opioids.**

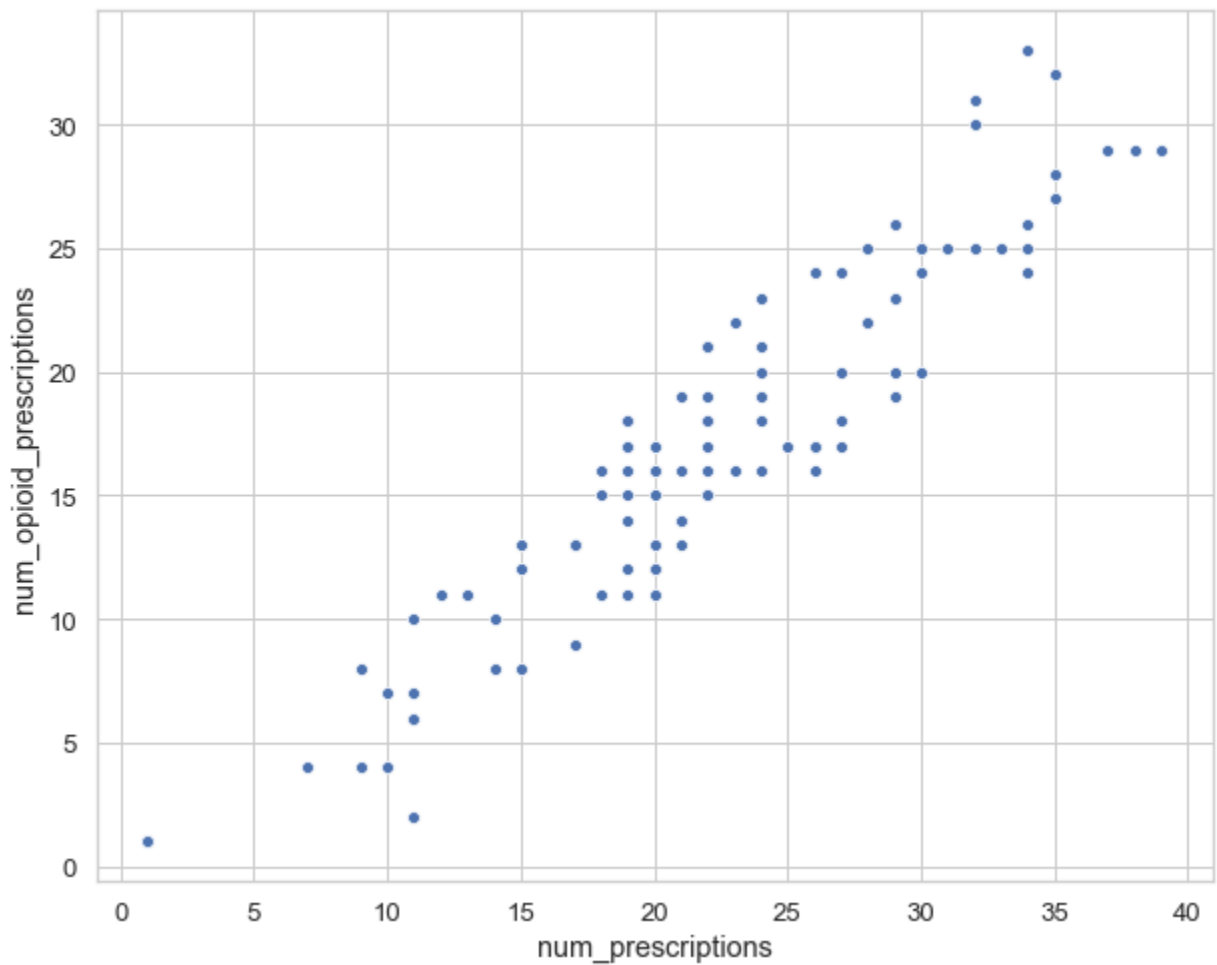
```
In [51]: cscribe['opioid_fraction'] = (prescribe['num_opioid_prescriptions'] / prescribe['num_prescrip
```

```
In [52]: prescribe.head(15)
```

```
Out[52]:
```

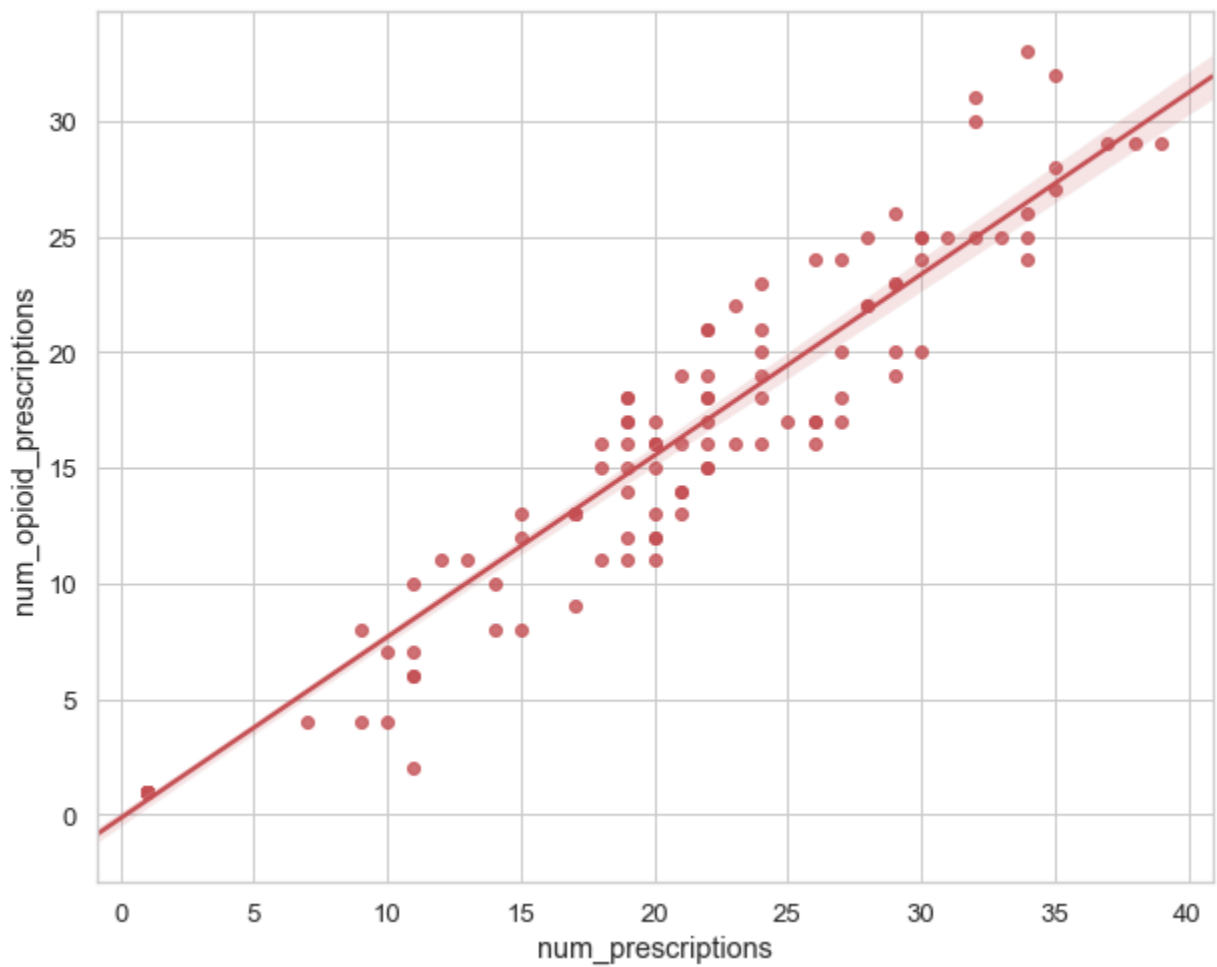
	prescriber_id	num_opioid_prescriptions	num_prescriptions	opioid_fraction
0	BDRSDDDB2&RA	4	7	57.142857
1	3\$N3WG4&BBO	25	28	89.285714
2	S2NGGOGBRB4	1	1	100.000000
3	S2NGGOGBRB4	18	22	81.818182
4	NBORABOESOI	1	1	100.000000
5	4O4EAGWROGB	20	30	66.666667
6	WPOARS4GW9I	11	12	91.666667
7	PDS\$GG9GOGA	29	37	78.378378
8	BBP34DOGADB	1	1	100.000000
9	S\$BO&OGO99O	16	19	84.210526
10	EDB3BB2AGAA	1	1	100.000000
11	ASG\$GDGBBGA	24	27	88.888889
12	OOIOEAIOSSR	16	20	80.000000
13	BSDN9OGBIG	1	1	100.000000
14	WGOBDRDRAN\$	1	1	100.000000

```
In [54]: plt.figure( figsize = (10,8) )  
sns.set(style = 'whitegrid', font_scale = 1.2)  
  
sns.scatterplot(x = prescribe['num_prescriptions'], y = prescribe['num_opioid_prescription'])  
plt.show()
```



```
In [57]: plt.figure( figsize = (10,8) )
sns.set(style = 'whitegrid', font_scale = 1.2)

p = sns.regplot(x = prescribe['num_prescriptions'], y = prescribe['num_opioid_prescription
                color = 'r' )
plt.show()
```





```
In [59]: x_val = p.get_lines()[0].get_xdata()    #Returns a numpy array of x_values.  
        y_val = p.get_lines()[0].get_ydata()    #Returns a numpy array of y_values.
```

```
In [65]: from scipy.stats import linregress  
  
        slope, intercept, r_value, p_value, stdErr = linregress(x_val, y_val)
```

```
In [66]: slope
```

```
Out[66]: 0.7843199719658177
```

```
In [67]: intercept
```

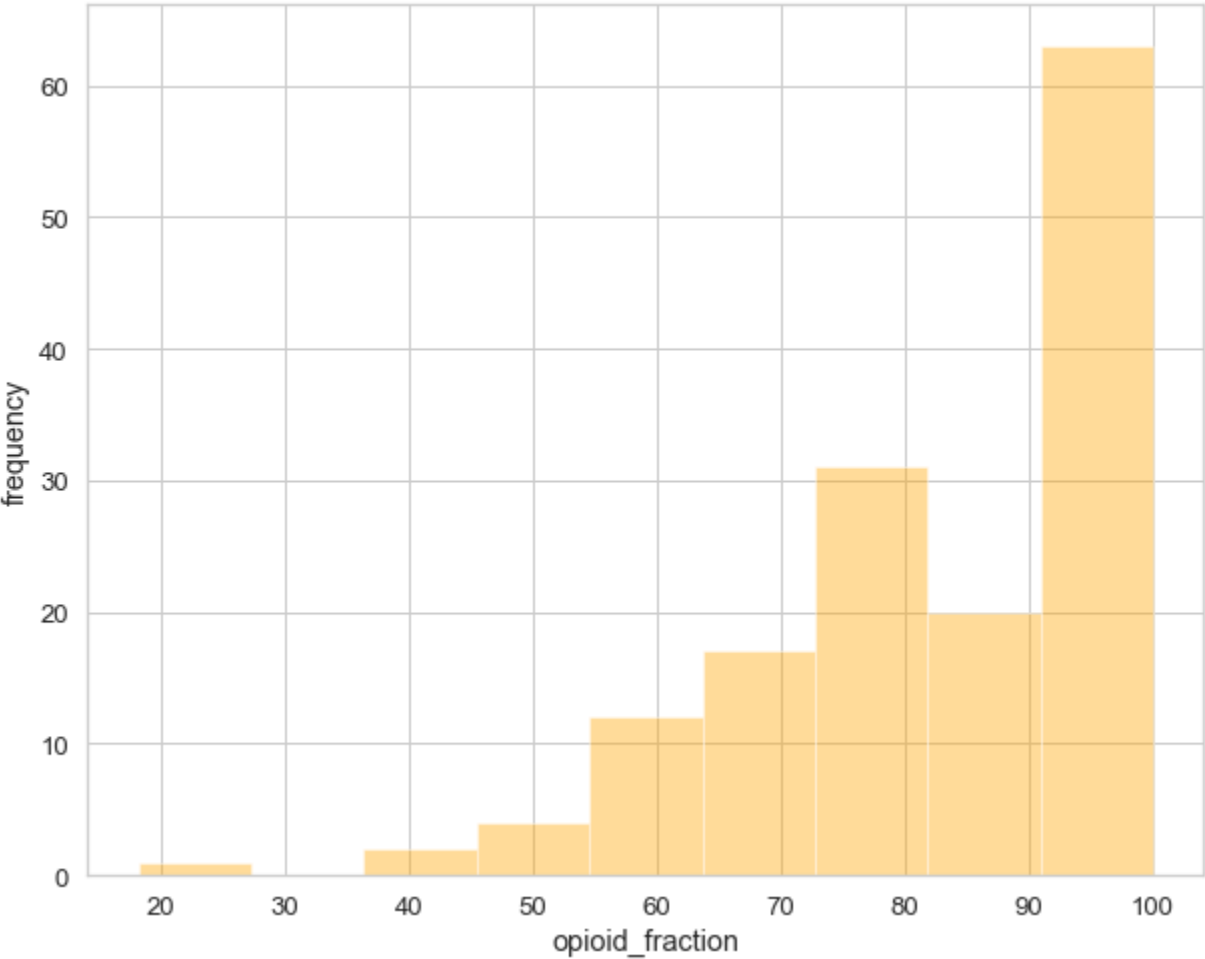
```
Out[67]: -0.17133823400578585
```

```
In [69]: R_squared = r_value ** 2  
        R_squared
```

```
Out[69]: 1.0
```

**Inspecting using opioid fraction.**

```
In [78]: plt.figure( figsize = (10,8) )
sns.set(style = 'whitegrid', font_scale = 1.2)
sns.distplot(prescribe['opioid_fraction'], kde= 0, color = 'orange')
plt.ylabel('frequency')
plt.show()
```



Many doctors prescribe opioids.

Grouping by ID's.

```
In [79]: prescribers = prescribe.groupby('prescriber_id').agg('sum')
```

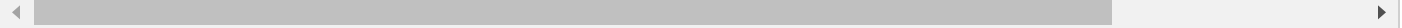
```
In [81]: prescribers.head()
```

Out[81]:

	num_opioid_prescriptions	num_prescriptions	opioid_fraction
prescriber_id			
\$39AGIIOGO2	15	22	68.181818
\$DG2GOGDD9D	14	18	176.470588
\$OBDNSADD96	37	57	280.384615
&4OO2EWOB	16	20	80.000000
&64O2OASRNA	22	23	195.454545

```
In [82]: del prescribers['opioid_fraction']
```

```
In [83]: prescribers['percentage'] = (prescribers['num_opioid_prescriptions'] / prescribers['num_pr
```



```
In [84]: prescribers.head()
```

Out[84]:

	num_opioid_prescriptions	num_prescriptions	percentage
prescriber_id			
\$39AGIIOGO2	15	22	68.181818
\$DG2GOGDD9D	14	18	77.777778
\$OBDNSADD96	37	57	64.912281
&4OO2EW OB	16	20	80.000000
&64O2OASRNA	22	23	95.652174

```
In [85]: prescribers.percentage.max()
```

Out[85]: 100.0

```
In [90]: prescribers.reset_index()
```

```
Out[90]:
```

	prescriber_id	num_opioid_prescriptions	num_prescriptions	percentage
0	\$39AGIIOGO2	15	22	68.181818
1	\$DG2GOGDD9D	14	18	77.777778
2	\$OBDNSADD96	37	57	64.912281
3	&4OO2EW OB	16	20	80.000000
4	&64O2OASRNA	22	23	95.652174
5	&AGO\$DEOPRR	47	58	81.034483
6	&OS2DN4DADO	1	1	100.000000
7	&SOEB4RODOG	39	51	76.470588
8	2BDPWROOBOR	13	17	76.470588
9	3\$N3WG4&BBO	32	39	82.051282
10	326G4D6ORWO	1	1	100.000000
11	3NG4ONN46GO	2	2	100.000000
12	3S36SB&ORN6	18	22	81.818182
13	4BBA2OGP9G6	1	1	100.000000
14	4BOSRODA\$GI	49	52	94.230769
15	4G6EIEBSRPS	27	34	79.411765
16	4O4EAGWROGB	21	31	67.741935
17	4R9RDN2G6D9	1	1	100.000000
18	4R9WBG4SONO	8	22	36.363636
19	4WA6NBOO&DB	1	1	100.000000
20	64AGGG6A9RA	34	39	87.179487
21	6BBSPODOSOD	17	27	62.962963
22	6DSBA2DOD3B	11	19	57.894737
23	6GG9DOSGBP6	13	20	65.000000
24	6GGGA4DPW&D	4	9	44.444444
25	6W349S&GD&N	1	1	100.000000
26	9ASI3O2B3GR	11	12	91.666667
27	9R3BBNGOSDR	1	1	100.000000
28	ASG\$GDGBBGA	35	47	74.468085
29	B\$AREPGD&RG	35	49	71.428571
...	...	...	...	...
63	OOSRDIO2ASW	28	35	80.000000
64	ORDG4S3GSWO	23	31	74.193548
65	ORPB9E2SBPO	6	11	54.545455
66	OWGW4PB4S94	25	31	80.645161
67	PBIOORDIBSG	50	64	78.125000

	prescriber_id	num_opioid_prescriptions	num_prescriptions	percentage
68	PBNDDAEOOPP	50	70	71.428571
69	PDS\$GG9GOGA	48	61	78.688525
70	PGWSD49S6OW	1	1	100.000000
71	PO&SOORGOPE	8	15	53.333333
72	PR\$RRSDOAGW	11	13	84.615385
73	R424SG\$2DEE	1	1	100.000000
74	R4DGWEOR2WG	7	10	70.000000
75	RA&RI23GSOO	22	28	78.571429
76	RADBOGNS4DS	24	26	92.307692
77	RBEOAOWBRAA	2	2	100.000000
78	RGD9&PEEGGO	29	38	76.315789
79	RNA6SBBAG6B	18	19	94.736842
80	RO4GO&6S\$B2	16	23	69.565217
81	S\$BO&OGO99O	33	41	80.487805
82	S2NGGOGBRB4	39	45	86.666667
83	SBO3WOOBIBP	22	23	95.652174
84	SEB6O6DPG3G	43	54	79.629630
85	SGSGRIIBDBA	31	33	93.939394
86	SNGS64GENRP	23	29	79.310345
87	SPBRGEG\$6BP	19	29	65.517241
88	SRD <i>GDS</i> DG\$	26	37	70.270270
89	W3NBOIPD3\$O	1	1	100.000000
90	W3O <i>ROB</i> 3R4	13	21	61.904762
91	WGOBDRDRAN\$	28	35	80.000000
92	WPOARS4GW9I	11	12	91.666667

93 rows × 4 columns

```
In [96]: prescribers = prescribers.sort_values('percentage', ascending = 0)
```

In [97]: `prescribers.head(20)`

Out[97]:

	num_opioid_prescriptions	num_prescriptions	percentage
prescriber_id			
DEW6WBN9W	1	1	100.000000
3NG4ONN46GO	2	2	100.000000
6W349S&GD&N	1	1	100.000000
4WA6NBOO&DB	1	1	100.000000
NBORABOESOI	1	1	100.000000
4R9RDN2G6D9	1	1	100.000000
OAIGB9OGSE\$	1	1	100.000000
PGWSD49S6OW	1	1	100.000000
BO&2BGDNGO&	1	1	100.000000
4BBA2OGP9G6	1	1	100.000000
G\$\$RSN6NDOD	1	1	100.000000
R424SG\$2DEE	1	1	100.000000
326G4D6ORWO	1	1	100.000000
DEWG\$NPBODS	1	1	100.000000
RBEOAOWBRAA	2	2	100.000000
&OS2DN4DADO	1	1	100.000000
9R3BBNGOSDR	1	1	100.000000
W3NBOIPD3\$O	1	1	100.000000
G4\$AGE&RRAG	1	1	100.000000
SBO3WOOBIBP	22	23	95.652174

The End.