Socioeconomic Factors and Disease

by
Chetan Munugala

Background

When looking for the causes of disease, we often focus on the individual and their personal health markers. However, the environment and circumstances that an individual finds themselves in may also contribute to disease outcomes. In this project, I explore various socioeconomic factors and how they correlate with disease.

Below you will find Python code that I used to compile, clean, visualize, and analyze relevant data, along with explanations of what I am doing. Additional information is included in the README file in the same folder as this project.

Data Import and Cleaning

I begin by importing necessary packages, reading csv files with the necessary data into data frames (links in README), and cleaning the data. This cleaning process involves dropping irrelevant data, standardizing naming conventions within my data frame, splitting strings on appropriate characters, and more.

In [64]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import re as re
from scipy import stats
import warnings
```

In [65]:
```
#ignoring certain warnings to make jupyter notebook presentation more elegant

warnings.simplefilter('ignore')
```

In [66]:
```
#importing all csv files containing data from links in README

income = pd.read_csv('Income Level.csv')
wilderness = pd.read_csv('Area of Wilderness.csv')
depression = pd.read_csv('Depression Rates.csv')
education = pd.read_csv('Education by State.csv')
heartattack = pd.read_csv('Heart Attack by State.csv')
natlparks = pd.read_csv('National Parks.csv')
unemployment = pd.read_csv('Unemployment Rate by State.csv')
cancer = pd.read_csv('Cancer Mortality by State.csv')
heartdisease = pd.read_csv('Heart Disease by State.csv')
suicide = pd.read_csv('Suicide by State.csv')
crime = pd.read_csv('Violent Crime by State.csv')
```

In [67]:
```
states = {'AK': 'Alaska','AL': 'Alabama', 'AR': 'Arkansas','AS': 'American Samoa','AZ': 'Ari
zona','CA': 'California','CO': 'Colorado',
          'CT': 'Connecticut','DC': 'District of Columbia','DE': 'Delaware','FL': 'Florida',
'GA': 'Georgia','GU': 'Guam','HI': 'Hawaii',
          'IA': 'Iowa','ID': 'Idaho','IL': 'Illinois','IN': 'Indiana','KS': 'Kansas','KY':
'Kentucky', 'LA': 'Louisiana',
          'MA': 'Massachusetts','MD': 'Maryland','ME': 'Maine','MI': 'Michigan','MN': 'Minne
sota','MO': 'Missouri','MP': 'Northern Mariana Islands',
          'MS': 'Mississippi','MT': 'Montana','NA': 'National','NC': 'North Carolina','ND':
'North Dakota','NE': 'Nebraska',
          'NH': 'New Hampshire','NJ': 'New Jersey','NM': 'New Mexico','NV': 'Nevada','NY':
'New York','OH': 'Ohio',
          'OK': 'Oklahoma','OR': 'Oregon','PA': 'Pennsylvania','PR': 'Puerto Rico','RI': 'Rh
ode Island','SC': 'South Carolina',
          'SD': 'South Dakota','TN': 'Tennessee','TX': 'Texas','UT': 'Utah','VA': 'Virginia'
,'VI': 'Virgin Islands',
          'VT': 'Vermont','WA': 'Washington','WI': 'Wisconsin','WV': 'West Virginia','WY':
'Wyoming'}
```

In [68]:
```
income.head()
```

Out[68]:

	Rank	State or territory	Per capita income	Median household income	Median family income	Population	Number of households	Number of families
0	1.0	District of Columbia	$45,877	$71,648	$84,094	658,893	277,378	117,864
1	2.0	Connecticut	$39,373	$70,048	$88,819	3,596,677	1,355,817	887,263
2	3.0	New Jersey	$37,288	$69,160	$87,951	8,938,175	2,549,336	1,610,581
3	4.0	Massachusetts	$36,593	$71,919	$88,419	6,938,608	3,194,844	2,203,675
4	5.0	Maryland	$36,338	$73,971	$89,678	5,976,407	2,165,438	1,445,972

Here is the head of a data frame of imported income data. I decided to keep only the State, Per capita income, and Median household income column. I dropped the rest of the columns and changed the names of the State column.

In [69]:
```
income.drop(['Rank','Number of households','Number of families','Median family income',],axi
s=1,inplace = True)
```

In [70]:
```
income.rename(columns={'State or territory':'State'},inplace=True)
income.set_index('State',inplace=True)
```

In [71]:
```
state_data = pd.DataFrame()
state_data = state_data.append(income)
```

Here, I created a data frame called state_data, which will eventually hold all of my relevant information. I added the cleaned income data frame to state_data and will proceed to do so for the rest of my categories.

Below you will find code for cleaning up the rest of the data frames containing my imported data.

In [72]:
```
wilderness.drop(['Wilderness','Located in','Agency','Size (ha)','Designated'],axis=1,inplace
=True)
wilderness = wilderness[~wilderness.State.str.contains(',')]
wilderness = wilderness.replace({'State':states})
wilderness['Size (acres)'] = wilderness['Size (acres)'].str.replace(',','')
wilderness['Size (acres)'] = pd.to_numeric(wilderness['Size (acres)'],downcast = 'integer')
wilderness = wilderness.groupby(['State']).sum()
wilderness.rename(columns={'Size (acres)':'Acres of Wilderness'},inplace=True)
```

In [73]:
```
depression.drop(['Rank','% Change in Depression Rate','Access Rank'],axis=1,inplace=True)
depression.set_index('State',inplace=True)
depression.rename(columns={'Depression Rate':'Depression Rate (%)'},inplace=True)
```

In [74]:
```
education.drop(['% High school graduate or higher','High School rank',"Bachelor's rank",'% A
dvanced degree',
               'Advanced rank'],axis=1,inplace=True)
education.rename(columns={'State, federal district, or territory':'State'},inplace=True)
```

In [75]:
```
heartattack.drop(['RANK'],axis=1,inplace=True)
heartattack.rename(columns={'VALUE':'% Heart Attack in Adults','STATE':'State'},inplace=True
)
heartattack.head()
heartattack.set_index('State',inplace=True)
```

In [76]:
```
natlparks['Location'] = natlparks['Location'].apply(lambda x:re.split(r'(^[^\d]+)', x)[1])
natlparks.drop(['Name','Image','Date established as park[5][10]','Recreation visitors (2018)
[8]','Description'],axis=1,inplace=True)
natlparks.rename(columns={'Location':'State','Area (2019)[11]':'Acres of National Park'},inp
lace=True)
natlparks['Acres of National Park'] = natlparks['Acres of National Park'].apply(lambda x: x.
split(' ')[0])
natlparks['Acres of National Park'] = natlparks['Acres of National Park'].str.replace(',',''
)
natlparks['Acres of National Park'] = pd.to_numeric(natlparks['Acres of National Park'],down
cast = 'integer')
natlparks = natlparks[~natlparks.State.str.contains(',')]
natlparks.set_index('State',inplace=True)
natlparks = natlparks.groupby(['State']).sum()
```

In [77]:
```
unemployment.drop(['Rank'],axis=1,inplace=True)
unemployment.rename(columns={'Unemployment Rate':'Unemployment Rate (%)'})
unemployment.set_index('State',inplace=True)
```

In [78]:
```
cancer = cancer.loc[cancer['YEAR'] == 2018]
cancer.drop(['YEAR','DEATHS','URL'],axis=1,inplace=True)
cancer.rename(columns={'STATE':'State','RATE':'Cancer Deaths per 100,000 Persons'},inplace=T
rue)
cancer = cancer.replace({'State':states})
cancer.set_index('State',inplace=True)
```

In [79]:
```
heartdisease['VALUE'] = heartdisease['VALUE']*100
heartdisease.rename(columns={'STATE':'State','VALUE':'%Heart Disease in Adults'},inplace=Tru
e)
heartdisease.drop('RANK',axis=1,inplace=True)
heartdisease.set_index('State',inplace=True)
```

In [80]:
```
suicide.rename(columns={'STATE':'State','VALUE':'Suicide Deaths per 100,000 Persons'},inplac
e=True)
suicide.drop('RANK',axis=1,inplace=True)
suicide.set_index('State',inplace=True)
```

In [81]:
```
crime.rename(columns={'STATE':'State','VALUE':'Violent Crimes per 100,000 Persons'},inplace=
True)
crime.drop('RANK',axis=1,inplace=True)
crime.set_index('State',inplace=True)
```

In [85]:
```
state_data = pd.merge(state_data,wilderness,how='outer',on='State')
state_data = pd.merge(state_data,depression,how='outer',on='State')
state_data = pd.merge(state_data,education,how='outer',on='State')
state_data = pd.merge(state_data,heartattack,how='outer',on='State')
state_data = pd.merge(state_data,natlparks,how='outer',on='State')
state_data = pd.merge(state_data,unemployment,how='outer',on='State')
state_data = pd.merge(state_data,cancer,how='outer',on='State')
state_data = pd.merge(state_data,heartdisease,how='outer',on='State')
state_data = pd.merge(state_data,suicide,how='outer',on='State')
state_data = pd.merge(state_data,crime,how='outer',on='State')
state_data.set_index('State',inplace=True)
state_data.drop(['United States[21]','District of Columbia','Guam','Puerto Rico','Northern M
ariana Islands','American Samoa','U.S. Virgin Islands','United States'],inplace=True)
```

Now I have successfully cleaned the data and put all of the information I am investigating into one data frame - state_data. In the process, I dropped some territories of the United States because there was a lot of missing data. Next, we must figure out how to deal with the remaining missing pieces of data in the data frame above.

In this instance, we fill the missing values for Acres of Wilderness and Acres of National Park with zero because as far as I can tell, they were zero because there is no designation for them within those states.

In [86]:
```
state_data['Acres of Wilderness'].fillna(value=0,inplace=True)
state_data['Acres of National Park'].fillna(value=0,inplace=True)
```

In [87]:
```
state_data.head()
```

Out[87]:

State	Per capita income	Median household income	Population	Acres of Wilderness	Depression Rate (%)	% Bachelor's degree or higher	% Heart Attack in Adults	Acres of National Park	Unemployment Rate	Cancer Deaths per 100,000 Persons
Connecticut	$39,373	$70,048	3,596,677	0.0	16.98%	38.40%	3.70%	0.0	3.7	134.1
New Jersey	$37,288	$69,160	8,938,175	10341.0	12.98%	38.10%	4.10%	0.0	3.8	141.3
Massachusetts	$36,593	$71,919	6,938,608	3244.0	19.56%	42.10%	4.10%	0.0	2.8	142.8
Maryland	$36,338	$73,971	5,976,407	0.0	16.12%	39.00%	3.90%	0.0	3.3	149.9
New Hampshire	$34,691	$66,532	1,326,813	175989.0	21.54%	36.00%	4.40%	0.0	2.6	143.7

At this point, we want to do some exploratory data analysis and visualization to see if there are any interesting relationships between the data. However, we must first convert all values into a format appropriate for graphing with the seaborn library.

- First, we must get rid of any extra punctuation such as dollar signs, commas, and percent signs.
- Second, we must convert all strings to numeric values. The code to achieve this is shown below.

In [88]:
```
state_data['Per capita income'] = state_data['Per capita income'].str.replace('$','')
state_data['Per capita income'] = state_data['Per capita income'].str.replace(',','')
state_data['Per capita income'] = pd.to_numeric(state_data['Per capita income'],downcast =
'integer')

state_data['Median household income'] = state_data['Median household income'].str.replace(
'$','')
state_data['Median household income'] = state_data['Median household income'].str.replace(
',','')
state_data['Median household income'] = pd.to_numeric(state_data['Median household income'],
downcast = 'integer')

state_data['Depression Rate (%)'] = state_data['Depression Rate (%)'].str.replace('%','')
state_data['Depression Rate (%)'] = pd.to_numeric(state_data['Depression Rate (%)'],downcast
= 'integer')

state_data["% Bachelor's degree or higher"] = state_data["% Bachelor's degree or higher"].st
r.replace('%','')
state_data["% Bachelor's degree or higher"] = pd.to_numeric(state_data["% Bachelor's degree
 or higher"],downcast = 'integer')

state_data["% Heart Attack in Adults"] = state_data["% Heart Attack in Adults"].str.replace(
'%','')
state_data["% Heart Attack in Adults"] = pd.to_numeric(state_data["% Heart Attack in Adults"
],downcast = 'integer')

state_data.head()
```

Out[88]:

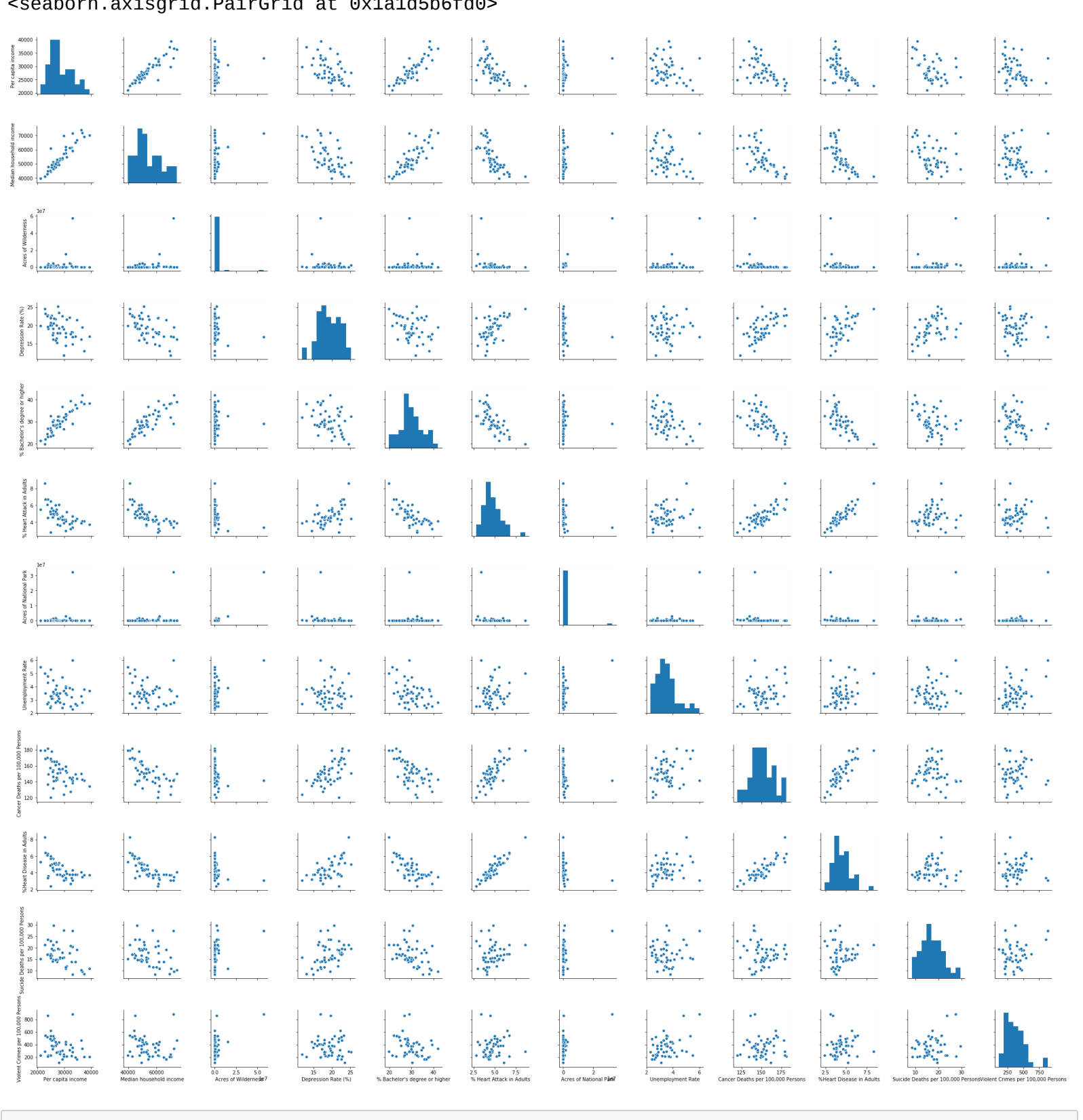
State	Per capita income	Median household income	Population	Acres of Wilderness	Depression Rate (%)	% Bachelor's degree or higher	% Heart Attack in Adults	Acres of National Park	Unemployment Rate	Cancer Deaths per 100,000 Persons
Connecticut	39373	70048	3,596,677	0.0	16.98	38.4	3.7	0.0	3.7	134.1
New Jersey	37288	69160	8,938,175	10341.0	12.98	38.1	4.1	0.0	3.8	141.3
Massachusetts	36593	71919	6,938,608	3244.0	19.56	42.1	4.1	0.0	2.8	142.8
Maryland	36338	73971	5,976,407	0.0	16.12	39.0	3.9	0.0	3.3	149.9
New Hampshire	34691	66532	1,326,813	175989.0	21.54	36.0	4.4	0.0	2.6	143.7

Exploratory Data Visualization and Analysis

Now we are ready to visualize our data. I begin by using two general plot types, pairplot and clustermap, to quickly look over relationships between data and see which relationships I want to further explore.

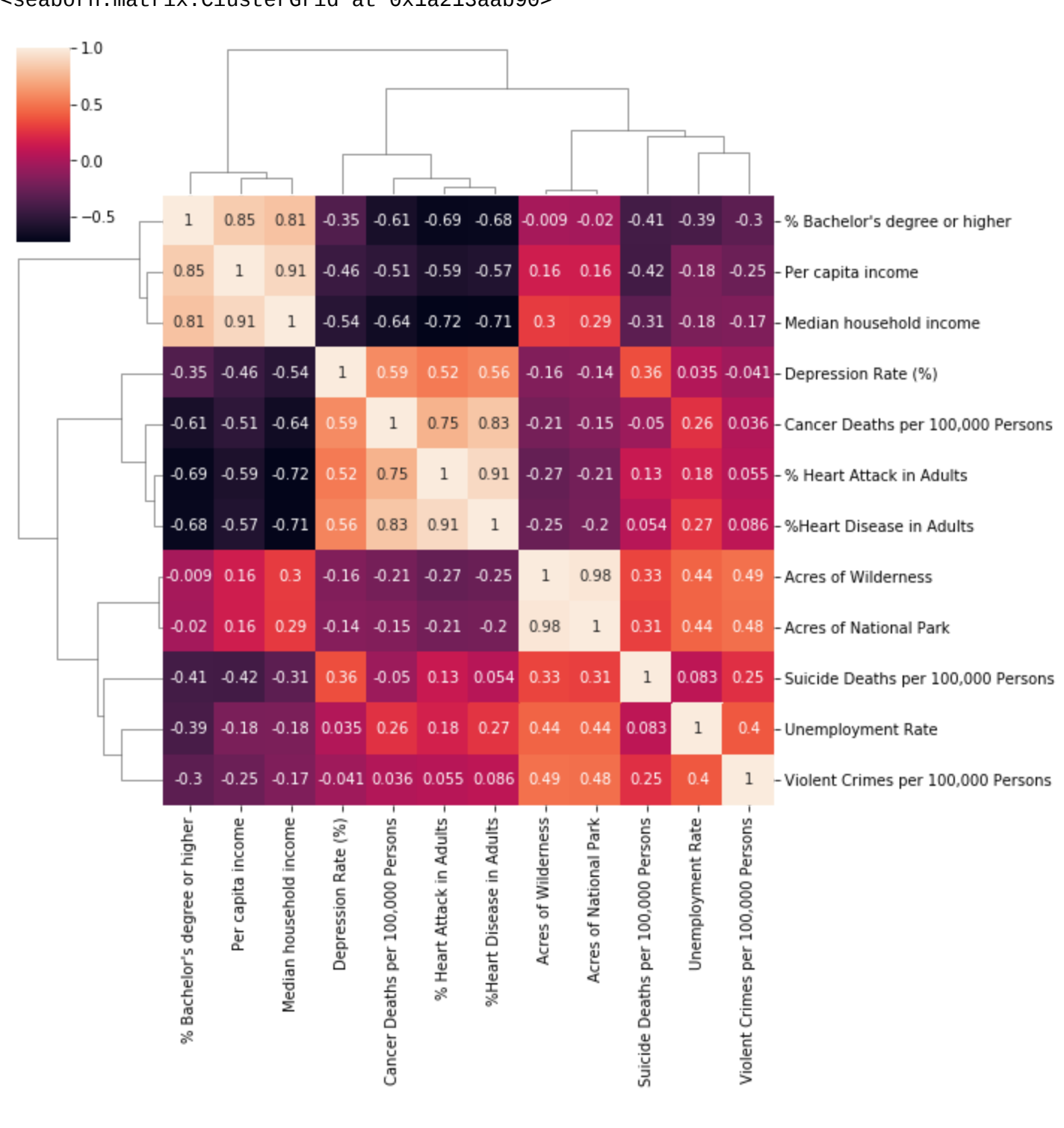
In [25]:
```
sns.pairplot(state_data)
```

Out[25]: `<seaborn.axisgrid.PairGrid at 0x1a1d5b6fd0>`



In [26]:
```
sns.clustermap(state_data.corr(),annot=True)
```

Out[26]: `<seaborn.matrix.ClusterGrid at 0x1a213aab90>`



Although there are many relationships we can explore, to avoid making this notebook too long, I will explore the three that were most interesting to me. Those are:

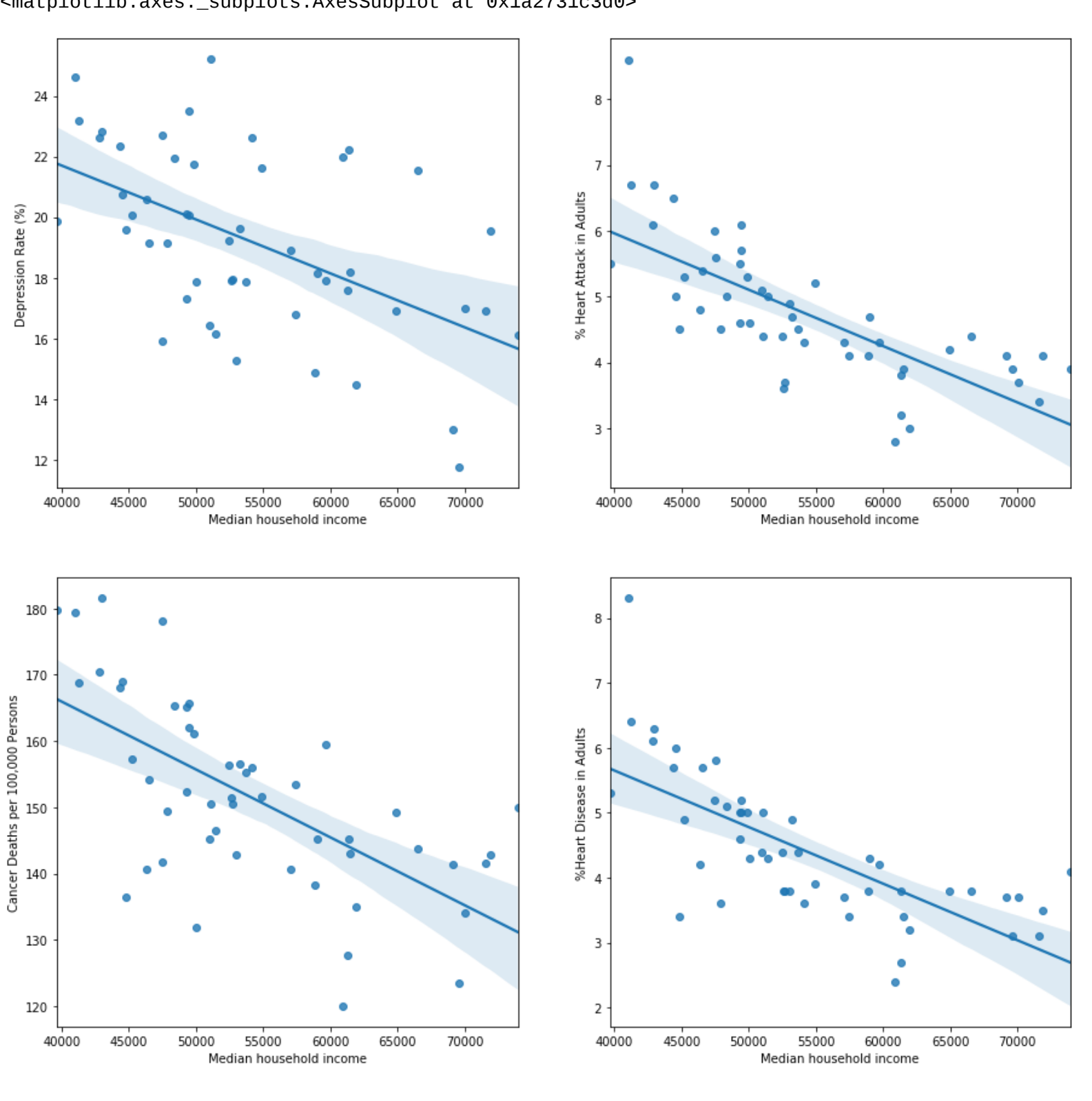
1. Effects of Income on Disease
2. Effects of Nature Accesibility on Disease
3. Effects of Education on Disease

Effects of Income on Disease

In [89]:
```
#code to make a table with pearson r values for correlations between variables with per capi
ta income

Depression = state_data['Median household income'].corr(state_data['Depression Rate (%)'])
Heart_Attack = state_data['Median household income'].corr(state_data['% Heart Attack in Adul
ts'])
Cancer_Deaths = state_data['Median household income'].corr(state_data['Cancer Deaths per 10
0,000 Persons'])
Heart_Disease = state_data['Median household income'].corr(state_data['%Heart Disease in Adu
lts'])

DepressionP = state_data['Per capita income'].corr(state_data['Depression Rate (%)'])
Heart_AttackP = state_data['Per capita income'].corr(state_data['% Heart Attack in Adults'])
Cancer_DeathsP = state_data['Per capita income'].corr(state_data['Cancer Deaths per 100,000
 Persons'])
Heart_DiseaseP = state_data['Per capita income'].corr(state_data['%Heart Disease in Adults'
])

corrs = pd.DataFrame((Depression,Heart_Attack,Cancer_Deaths,Heart_Disease),columns=['Median
 Household Income (pearsonr)'])
corrs['Per Capita Income (pearsonr)'] = [DepressionP,Heart_AttackP,Cancer_DeathsP,Heart_Dise
aseP]
corrs['']= ['Depression','Heart Attack','Cancer Deaths','Heart Disease']
corrs.set_index('',inplace=True)
```

In [90]:
```
#plot four subplots showing relationships between variables

fig, axs = plt.subplots(nrows=2,ncols=2,figsize=(15,15))
sns.regplot(x='Median household income',y='Depression Rate (%)',data=state_data,ax=axs[0][0
])
sns.regplot(x='Median household income',y='% Heart Attack in Adults',data=state_data,ax=axs[
0][1])
sns.regplot(x='Median household income',y='Cancer Deaths per 100,000 Persons',data=state_dat
a,ax=axs[1][0])
sns.regplot(x='Median household income',y='%Heart Disease in Adults',data=state_data,ax=axs[
1][1])
```

Out[90]: `<matplotlib.axes._subplots.AxesSubplot at 0x1a2731c3d0>`



As we can see from the correlation data and the visualizations, as median household income decreases we see an increase in depression, heart attacks, cancer deaths, and heart disease. This data suggests that having financial resources can help to increase health with regards to these conditions. I could speculate many reasons why this may be the case, but a deeper dive is required to find out with certainty.

In [91]:
```
corrs
```

Out[91]:

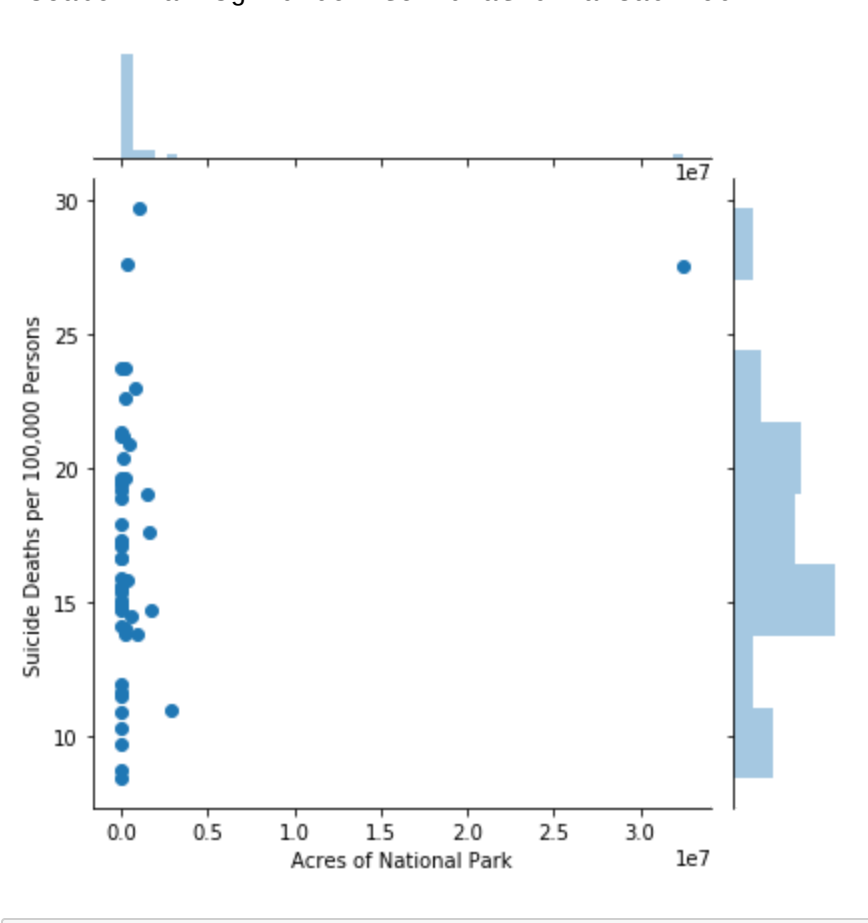
	Median Household Income (pearsonr)	Per Capita Income (pearsonr)
Depression	-0.536394	-0.455101
Heart Attack	-0.720379	-0.589971
Cancer Deaths	-0.641760	-0.509000
Heart Disease	-0.711137	-0.571779

Additionally, we can see in the table above that Median Household Income is more strongly correlated with each of these diseases than Per Capita Income. This suggests that what a group of individuals living together is making is more important than what an individual is making.

Effects of Nature Accesibility on Disease

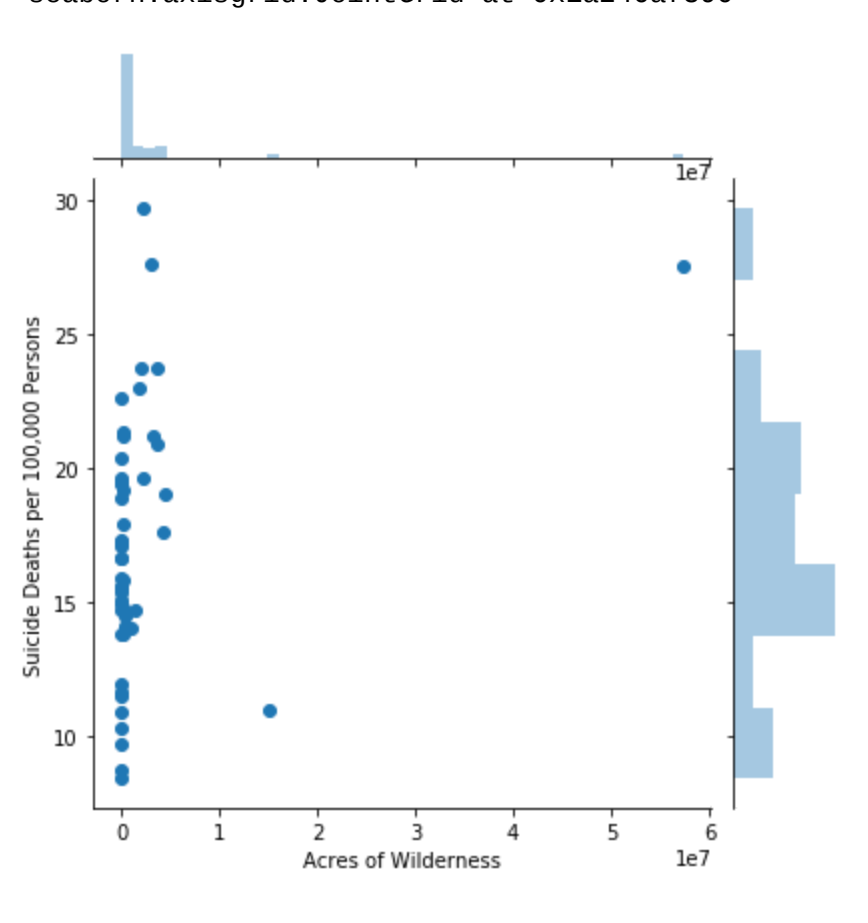
In [29]:
```
sns.jointplot(x=state_data['Acres of National Park'],y=state_data['Suicide Deaths per 100,00
0 Persons'])
```

Out[29]: `<seaborn.axisgrid.JointGrid at 0x1a23a01790>`



In [30]:
```
sns.jointplot(x=state_data['Acres of Wilderness'],y=state_data['Suicide Deaths per 100,000 P
ersons'])
```

Out[30]: `<seaborn.axisgrid.JointGrid at 0x1a246af890>`

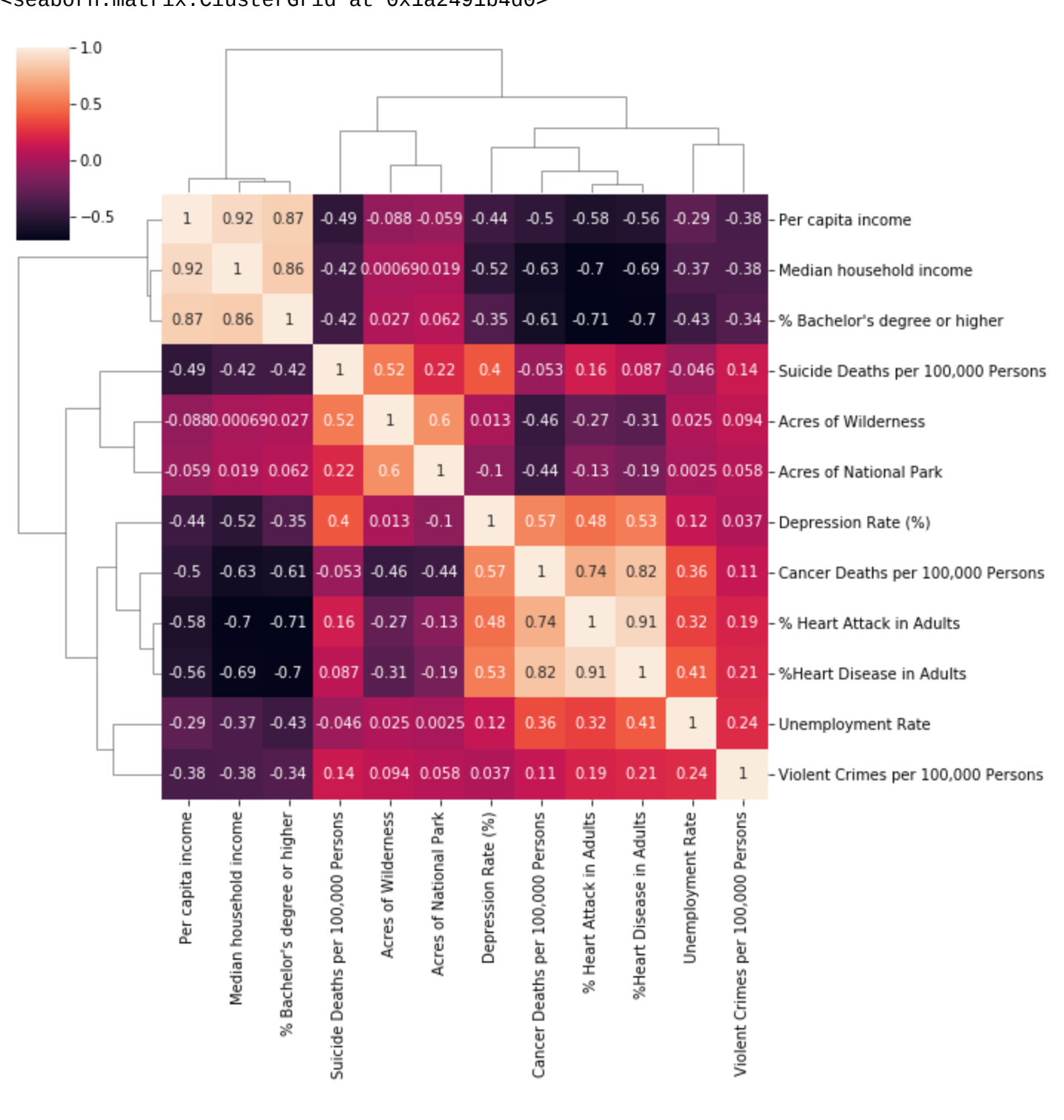


I start by looking at the distribution of data in "Acres of Wilderness" and "Acres of National Park". As we can see from the plots above, there are outliers that are making the visualizations hard to interpret and likely skewing correlation coefficients. For that reason, I chose to remove the three outliers in each of these data sets and take another look at their correlation with disease. The results are shown in a clustermap below.

In [31]:
```
no_outliers = state_data[(state_data['Acres of Wilderness']< 1.4e7) & (state_data['Acres of
 National Park'] < 2e6)]
```

In [32]:
```
sns.clustermap(no_outliers.corr(),annot=True)
```

Out[32]: `<seaborn.matrix.ClusterGrid at 0x1a2491b4d0>`



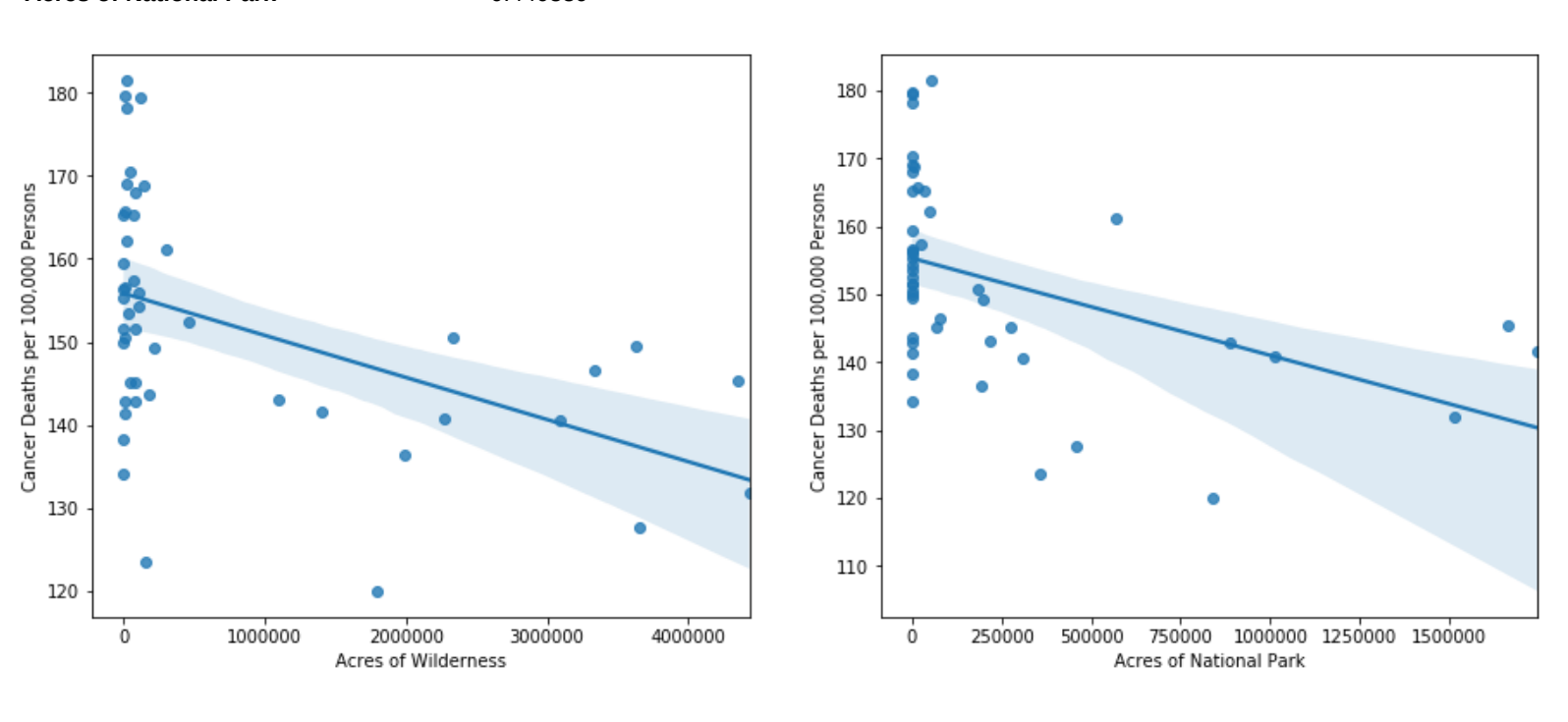
As you can see, the clustering of these variables changed along with the correlation coefficients for the "Acres of Wilderness" and "Acres of National Park" columns. Specifically, there seem to be interesting relationships between these variables and "Cancer Deaths per 100,000 Persons" and "Suicide Deaths per 100,000 Persons".

In [33]:
```
#code to make a table with pearson r values for correlations between variables with cancer d
eaths

Wilderness = no_outliers["Acres of Wilderness"].corr(no_outliers['Cancer Deaths per 100,000
 Persons'])
National_Parks = no_outliers["Acres of National Park"].corr(no_outliers['Cancer Deaths per 1
00,000 Persons'])

corrs = pd.DataFrame((Wilderness, National_Parks),columns=['Cancer Deaths (pearsonr)'])
corrs['']= ['Acres of Wilderness', 'Acres of National Park']
corrs.set_index('',inplace=True)
```

In [34]:
```
#plot two subplots showing relationships between variables


fig, axs = plt.subplots(nrows=1,ncols=2,figsize=(15,6))


sns.regplot(x='Acres of Wilderness',y='Cancer Deaths per 100,000 Persons',data=no_outliers,a
x=axs[0])
sns.regplot(x='Acres of National Park',y='Cancer Deaths per 100,000 Persons',data=no_outlier
s,ax=axs[1])

corrs
```

Out[34]:

	Cancer Deaths (pearsonr)
Acres of Wilderness	-0.461495
Acres of National Park	-0.440889

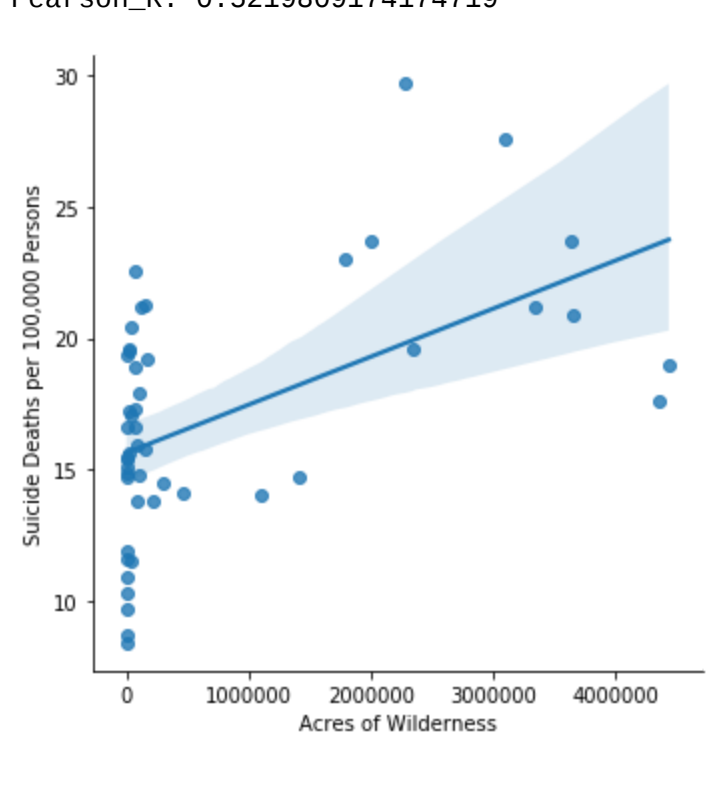


As you can see from the plots above, Cancer Deaths seem to go down as the Acres of Wilderness and National Parks go up in a given state. However, we must take this with a grain of salt because of the minimum number of data points. Still, it is an interesting observation and is an area I would like to explore in more detail in the future.

In [50]:
```
Wilderness = no_outliers["Acres of Wilderness"].corr(no_outliers['Suicide Deaths per 100,000
Persons'])
```

In [51]:
```
sns.lmplot(x='Acres of Wilderness',y='Suicide Deaths per 100,000 Persons',data=no_outliers)
print('Pearson_R:',Wilderness)
```

```
Pearson_R: 0.5219809174174719
```



Interestingly, suicide deaths seem to go up as Acres of Wilderness in a state goes up. When starting this project, one of hypotheses was that people who spent more time in nature were more healthy, especially when it came to mental health. For that reason, this result is surprising to me and definitely worth looking at again.

Effects of Having a Bachelors Degree on Disease

In [52]:
```
#code to make a table with pearson r values for correlations between variables with educatio
n

Depression = state_data["% Bachelor's degree or higher"].corr(state_data['Depression Rate
 (%)'])
Heart_Attack = state_data["% Bachelor's degree or higher"].corr(state_data['% Heart Attack i
n Adults'])
Cancer_Deaths = state_data["% Bachelor's degree or higher"].corr(state_data['Cancer Deaths p
er 100,000 Persons'])
Heart_Disease = state_data["% Bachelor's degree or higher"].corr(state_data['%Heart Disease
 in Adults'])
Suicide_Deaths = state_data["% Bachelor's degree or higher"].corr(state_data['Suicide Deaths
per 100,000 Persons'])


corrs = pd.DataFrame((Depression,Heart_Attack,Cancer_Deaths,Heart_Disease,Suicide_Deaths),co
lumns=['Bachelors degree or higher (pearsonr)'])
corrs['']= ['Depression','Heart Attack','Cancer Deaths','Heart Disease','Suicide Deaths']
corrs.set_index('',inplace=True)
```

In [53]:
```
#plot five subplots showing relationships between variables

fig, axs = plt.subplots(nrows=3,ncols=2,figsize=(15,15))
sns.regplot(x="% Bachelor's degree or higher",y='Depression Rate (%)',data=state_data,ax=axs
[0][0])
sns.regplot(x="% Bachelor's degree or higher",y='% Heart Attack in Adults',data=state_data,a
x=axs[0][1])
sns.regplot(x="% Bachelor's degree or higher",y='Cancer Deaths per 100,000 Persons',data=sta
te_data,ax=axs[1][0])
sns.regplot(x="% Bachelor's degree or higher",y='%Heart Disease in Adults',data=state_data,a
x=axs[1][1])
sns.regplot(x="% Bachelor's degree or higher",y='Suicide Deaths per 100,000 Persons',data=st
ate_data,ax=axs[2][0])

corrs
```

Out[53]:

	Bachelors degree or higher (pearsonr)
Depression	-0.351254
Heart Attack	-0.687355
Cancer Deaths	-0.608123
Heart Disease	-0.684748
Suicide Deaths	-0.413785