FinalTutorial

August 8, 2023

1 Analysis of Suicide by State in Relation to Multiple Risk Factors

By: Matthewos Gashaw, Joseph Jubilee, Jacob Lane & Trevor Lorin

1.0.1 Introduction

According to the World Health Organization, more than 700,000 people die each year from suicide. Suicide is a tragic occurrence that happens in all sorts of areas around the world and is a global phenomenon. Many different things factor into what may cause a person to attempt suicide or be at risk for attempting suicide. The World Health Organization cites a prior suicide attempt as the most important risk factor for suicide. Acknowledging that someone is at risk and getting them proper help is key to suicide prevention. In order to help aid in suicide prevention, it is important to be able to identify all kinds of different risk factors. The World Health Organization webpage for suicide facts contains more in-depth information on known suicide risk factors and prevention.

Previous studies have revealed known risk factors such as a family history of mental health problems, being part of a group that experiences societal discrimination, a diagnosis of a mental disorder, and the largest known risk: a previous suicide attempt. The use of data science can help us determine how other lesser-known lifestyle factors have an effect on suicide rates. Examining other factors such as average household income in the area of residence, average sunlight hours, and the prevalence of firearms can help give insight into what weighs more into an increase in suicides. Comparing these less obvious factors' effects to known factors like depression and alcoholism will show if these factors truly affect suicide. Following through the data science lifecycle with suicide data can help provide statistical data on what factors into being at higher risk for suicide.

Why is This Important?

Studying what has a larger effect on suicide rates can help prevent further deaths by suicide. Suicide is a preventable cause of death, and gaining a better understanding of what leads people to be at risk can help organizations better utilize their preventative resources. Data proving that certain areas may be at a higher risk for suicide can help focus mental health services in those regions. A general knowledge of what puts a person at higher risk also helps individuals seek better resources for themselves and their peers. The data from this tutorial may help determine using risk factors the likelihood a person is at risk for suicide and can help with better suicide prevention.

1.0.2 Purpose

The purpose of this tutorial is to provide a relevant example to guide readers through the data science lifecycle. The data science lifecycle is comprised of 5 stages: Data Collection, Data Processing, Exploratory Analysis and Data Visualization, Model Analysis, and Interpretation. In this tutorial, we will navigate through all 5 stages to demonstrate how data scientists draw conclusions from messy and seemingly unrelated data sets. Understanding what factors into suicide is extremely important for saving lives. Data science can allow for a better understanding of the risk of suicide and what can be done moving forward for more effective prevention. This tutorial demonstrates how data science can be applied to help provide insight into serious matters with results that can be used to make effective changes to make a positive change.

1.0.3 Data Collection

For this first step in the data science lifecycle, the data itself has to be acquired along with the tools that will be used for or data science process. We will be using the language Python to process the data since Python contains many useful libraries for the different steps of the data science lifecycle. Here we begin our code by importing all the necessary libraries.

[1]:
```
!pip install statsmodels
```

```
Requirement already satisfied: statsmodels in /root/venv/lib/python3.7/site-
packages (0.13.5)
Requirement already satisfied: scipy>=1.3 in /shared-
libs/python3.7/py/lib/python3.7/site-packages (from statsmodels) (1.7.3)
Requirement already satisfied: patsy>=0.5.2 in /root/venv/lib/python3.7/site-
packages (from statsmodels) (0.5.3)
Requirement already satisfied: numpy>=1.17 in /shared-
libs/python3.7/py/lib/python3.7/site-packages (from statsmodels) (1.21.6)
Requirement already satisfied: pandas>=0.25 in /shared-
libs/python3.7/py/lib/python3.7/site-packages (from statsmodels) (1.2.5)
Requirement already satisfied: packaging>=21.3 in /shared-libs/python3.7/py-
core/lib/python3.7/site-packages (from statsmodels) (21.3)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /shared-
libs/python3.7/py-core/lib/python3.7/site-packages (from
packaging>=21.3->statsmodels) (3.0.9)
Requirement already satisfied: pytz>=2017.3 in /shared-
libs/python3.7/py/lib/python3.7/site-packages (from pandas>=0.25->statsmodels)
(2022.5)
Requirement already satisfied: python-dateutil>=2.7.3 in /shared-
libs/python3.7/py-core/lib/python3.7/site-packages (from
pandas>=0.25->statsmodels) (2.8.2)
Requirement already satisfied: six in /shared-libs/python3.7/py-
core/lib/python3.7/site-packages (from patsy>=0.5.2->statsmodels) (1.16.0)
WARNING: You are using pip version 22.0.4; however, version 23.1.2 is

available.

You should consider upgrading via the '/root/venv/bin/python -m pip install

--upgrade pip' command.
```

[2]:
```python
# data collection
import numpy as np
import pandas as pd
import re
import random

# plotting graphs and the map of every country
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sb
!pip install statsmodels
from statsmodels.formula.api import ols

# machine learning and filling in missing data
from sklearn import linear_model
from sklearn.impute import SimpleImputer
from sklearn.impute import KNNImputer
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import LinearRegression
from sklearn import preprocessing
from sklearn.cluster import KMeans
```

```
Requirement already satisfied: statsmodels in /root/venv/lib/python3.7/site-
packages (0.13.5)
Requirement already satisfied: patsy>=0.5.2 in /root/venv/lib/python3.7/site-
packages (from statsmodels) (0.5.3)
Requirement already satisfied: scipy>=1.3 in /shared-
libs/python3.7/py/lib/python3.7/site-packages (from statsmodels) (1.7.3)
Requirement already satisfied: packaging>=21.3 in /shared-libs/python3.7/py-
core/lib/python3.7/site-packages (from statsmodels) (21.3)
Requirement already satisfied: numpy>=1.17 in /shared-
libs/python3.7/py/lib/python3.7/site-packages (from statsmodels) (1.21.6)
Requirement already satisfied: pandas>=0.25 in /shared-
libs/python3.7/py/lib/python3.7/site-packages (from statsmodels) (1.2.5)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /shared-
libs/python3.7/py-core/lib/python3.7/site-packages (from
packaging>=21.3->statsmodels) (3.0.9)
Requirement already satisfied: pytz>=2017.3 in /shared-
libs/python3.7/py/lib/python3.7/site-packages (from pandas>=0.25->statsmodels)
(2022.5)
Requirement already satisfied: python-dateutil>=2.7.3 in /shared-
libs/python3.7/py-core/lib/python3.7/site-packages (from
pandas>=0.25->statsmodels) (2.8.2)
Requirement already satisfied: six in /shared-libs/python3.7/py-
core/lib/python3.7/site-packages (from patsy>=0.5.2->statsmodels) (1.16.0)
```

```
WARNING: You are using pip version 22.0.4; however, version 23.1.2 is
available.
You should consider upgrading via the '/root/venv/bin/python -m pip install
--upgrade pip' command.
```

For each new CSV file being read into a DataFrame, the new data frame is then left merged by state into the first data frame containing suicide rates per state. If column names need to be renamed, this is also done while merging the DataFrames.

In this tutorial, we are using CSV files containing suicide rates by state along with other variables that may be risk factors by state. The CSV files used for this tutorial can be downloaded from the sites below.

https://worldpopulationreview.com/state-rankings/suicide-rates-by-state

https://worldpopulationreview.com/state-rankings/average-temperatures-by-state

https://worldpopulationreview.com/state-rankings/guns-per-capita

https://worldpopulationreview.com/state-rankings/average-family-income

https://worldpopulationreview.com/state-rankings/income-inequality-by-state

https://worldpopulationreview.com/state-rankings/state-densities

https://worldpopulationreview.com/state-rankings/sunniest-states

https://www.kff.org/other/state-indicator/distribution-by-age

https://worldpopulationreview.com/state-rankings/marriage-rate-by-state

https://worldpopulationreview.com/state-rankings/alcohol-consumption-by-state

https://worldpopulationreview.com/state-rankings/median-age-by-state

https://www.cdc.gov/nchs/covid19/pulse/mental-health.htm

In this next code block, we use the Pandas library to read the CSV files containing the different data we will be processing. We take each CSV file and read it into a Pandas DataFrame to cleanly store the data in a usable format. Pandas documentation can be found here for further reading.

[3]:
```
suicideDf = pd.read_csv("suicide-rate-unadjusted-by-state.csv")

familyIncomeDf = pd.read_csv("median-income-by-state.
 ↪csv")[['state','FamiliesMedianIncome']]

sunlightDf = pd.read_csv("hours-of-sunlight-by-state.csv")[['state',␣
 ↪'averageAnnualSunlight']]

giniIncomeDf = pd.read_csv("income-gini-coefficient-by-state.
 ↪csv")[['state','giniCoefficient']]
```

```
tempDf = pd.read_csv("average-temp-by-state.csv")[['state',␣
↪'AverageTemperature']]

ageDf = pd.read_csv("median-age-by-state.csv")[['state', 'MedianAge']]

alcDf = pd.read_csv("alcohol-consumption-by-state.
↪csv")[['state','alcoholConsumptionGallons']]

gunsDf = pd.read_csv("guns-per-capita-by-state.csv")[['state',␣
↪'gunsRegistered']]

marriageDf = pd.read_csv("marriage_by_state.csv")[['state', 'Married']]

anxdepDf = pd.read_csv("anxiety-depression-by-state.csv")
```

1.0.4 Data Processing

For this second step of the data science lifecycle, the data must be processed and cleaned so it can be used for analysis and modeling. Raw data files are often not in a clean format and must be processed to be properly used.

Formatting the DataFrame

To prepare the data for analysis and modeling, the data should all be contained in one DataFrame. The suicide-by-state CSV has extra columns containing data unnecessary for our analysis purposes. Hence, we begin our data processing by only grabbing the three columns from that DataFrame that we need: state, population, and suicide rate. Then to put all the data in one DataFrame, we left merge all of the other DataFrames on the state. This results in the one suicide DataFrame now containing all the observations from the other DataFrames.

For the age DataFrame, before we merged the contents we had to rename the location column to "state" to merge the two DataFrames on that column.

[4]:
```
suicideDf = suicideDf[['state', 'pop2023', 'suicideRate']]

suicideDf = suicideDf.merge(ageDf, how='left', on='state')

suicideDf = suicideDf.merge(familyIncomeDf, how='left', on='state')

suicideDf = suicideDf.merge(sunlightDf, how='left', on='state')

suicideDf = suicideDf.merge(giniIncomeDf, how='left', on='state')

suicideDf = suicideDf.merge(tempDf, how='left', on='state')

suicideDf = suicideDf.merge(gunsDf, how='left', on='state')

suicideDf = suicideDf.merge(alcDf, how='left', on='state')
```

```
suicideDf = suicideDf.merge(marriageDf, how='left', on='state')
```

To process the data from the CDC, we start by grabbing the needed columns from the DataFrame. The data originally had rows for each time period recorded, so we grouped by the state and indicator columns to take a mean over all the time periods for each state and type of data. We then use the pivot method to take the data types in the "Indicator" column and make them each their own column, with the average percentage of people reporting those symptoms over time as the values under that column. We then chose to just select the percentage of people reporting symptoms of depression as that is what we chose to focus on.

[5]:
```
# Processing CDC Suicide/Anxiety Data
anxdepDf = anxdepDf[anxdepDf['Group'] == 'By State'][['Indicator', 'State',␣
 ↪'Value', 'Phase']]
# Group by state and average the percentages over all time periods reported
anxdepDf = anxdepDf.groupby(by=['State', 'Indicator']).agg(func='mean').
 ↪reset_index()
anxdepDf = anxdepDf.pivot(index='State', columns='Indicator', values='Value')
anxdepDf.reset_index(inplace=True)
anxdepDf.rename(columns={'State':'state', 'Symptoms of Depressive Disorder':␣
 ↪'PercentDepression'}, inplace=True)
anxdepDf = anxdepDf[['state','PercentDepression']]

suicideDf = suicideDf.merge(anxdepDf, how='left', on='state')
```

[6]:
```
suicideDf.head(10)
```

[6]:
```
          state   pop2023  suicideRate  MedianAge  FamiliesMedianIncome  \
0       Wyoming    583279         32.3       38.0                 81290
1       Montana   1139507         32.0       40.1                 72773
2        Alaska    732984         30.8       34.6                 92648
3    New Mexico   2110011         25.0       38.1                 62611
4  South Dakota    923484         23.2       37.2                 77042
5      Colorado   5868555         22.8       36.9                 92752
6      Oklahoma   4048375         22.1       36.7                 67511
7        Nevada   3209142         21.5       38.2                 74077
8  North Dakota    780588         20.8       35.2                 86798
9      Arkansas   3063152         20.6       38.3                 62067

   averageAnnualSunlight  giniCoefficient  AverageTemperature  gunsRegistered  \
0                 4471.0            43.00                42.3          132806
1                 3847.0            45.87                42.6           22133
2                    NaN            41.74                28.1           15824
3                 5642.0            48.00                54.5           97580
4                 4332.0            44.00                45.8           21130
5                 4960.0            45.90                46.3           92435
6                 4912.0            46.52                60.4           71269
7                 5296.0            45.00                51.1           76888
```

```
8                3925.0           46.00                41.1          13272
9                4725.0           47.00                61.1          79841

   alcoholConsumptionGallons  Married  PercentDepression
0                       2.78     54.3          21.976786
1                       3.10     52.0          21.767857
2                       2.85     49.2          24.432143
3                       2.26     43.9          26.603571
4                       2.87     51.4          19.532143
5                       2.88     49.8          23.250000
6                       1.85     49.0          27.250000
7                       3.42     45.7          27.748214
8                       3.16     52.4          20.489286
9                       1.78     49.7          27.108929
```

Dealing with Missing Values

In the average annual sunlight column for Alaska and Hawaii, there are missing values. These values are considered to be missing at random and will require imputation to continue data analysis. To impute these values we are going to use a function of average temperature since sunlight and temperature are interrelated values.

For our imputation, we are utilizing the K nearest neighbors imputer from the sklearn library. We are setting k equal to two and then are fitting the data between average annual sunlight and average temperature. We then are going to use the imputed values to replace the missing data for average annual sunlight.

[7]:
```
imp = KNNImputer(n_neighbors=2)
imp.fit(suicideDf[['averageAnnualSunlight', 'AverageTemperature']])
imputed = imp.transform(suicideDf[['averageAnnualSunlight',␣
 ↪'AverageTemperature']])
suicideDf['averageAnnualSunlight'] = imputed[:,0]
```

Displayed below is the final processed DataFrame containing all the cleaned data. This DataFrame will be used for analysis and to create a machine-learning model in the later steps of the data science lifecycle.

[8]:
```
suicideDf.head(10)
```

[8]:
```
           state   pop2023  suicideRate  MedianAge  FamiliesMedianIncome  \
0        Wyoming    583279         32.3       38.0                 81290
1        Montana   1139507         32.0       40.1                 72773
2         Alaska    732984         30.8       34.6                 92648
3     New Mexico   2110011         25.0       38.1                 62611
4   South Dakota    923484         23.2       37.2                 77042
5       Colorado   5868555         22.8       36.9                 92752
6       Oklahoma   4048375         22.1       36.7                 67511
7         Nevada   3209142         21.5       38.2                 74077
8   North Dakota    780588         20.8       35.2                 86798
```

9	Arkansas	3063152	20.6	38.3	62067
10	West Virginia	1764786	20.6	42.7	61707
11	Idaho	1973752	20.5	36.6	70885
12	Vermont	647156	20.3	42.8	83023
13	Utah	3422487	20.1	31.1	84590
14	Arizona	7453517	19.5	37.9	73456
15	Maine	1393442	19.5	44.8	76192
16	Oregon	4223973	19.5	39.5	80630
17	Kansas	2936378	19.4	36.9	77620
18	Missouri	6186091	18.7	38.7	72834
19	Kentucky	4518031	17.9	39.0	65893
20	Iowa	3203345	17.5	38.3	79186
21	Tennessee	7134327	17.0	38.8	68793
22	Indiana	6852542	16.4	37.8	73265
23	Mississippi	2930528	16.2	37.7	58923
24	Alabama	5098746	15.8	39.2	66772
25	Georgia	11037723	15.3	36.9	74127
26	Washington	7830827	15.3	37.8	92422
27	South Carolina	5372002	15.2	39.7	68813
28	New Hampshire	1402957	15.1	43.0	97001
29	Wisconsin	5904977	15.1	39.6	80844
30	Nebraska	1972292	15.0	36.6	80125
31	Louisiana	4553384	14.8	37.2	65427
32	Ohio	11747774	14.6	39.5	74391
33	Michigan	10030722	14.3	39.8	75470
34	Texas	30500280	14.2	34.8	76073
35	Florida	22661577	14.0	42.2	69670
36	Minnesota	5722897	13.9	38.1	92692
37	Pennsylvania	12931957	13.9	40.9	80996
38	Hawaii	1433238	13.7	39.4	97813
39	Delaware	1031985	13.6	41.0	84825
40	North Carolina	10832061	13.2	38.9	70978
41	Virginia	8709873	13.2	38.4	93284
42	Illinois	12477595	11.1	38.3	86251
43	Rhode Island	1090483	10.3	40.0	89330
44	California	38915693	10.1	36.7	89798
45	Connecticut	3629055	10.0	41.1	102061
46	Maryland	6154710	9.7	38.8	105790
47	Massachusetts	6974258	8.0	39.6	106526
48	New York	19496810	7.9	39.0	87270
49	New Jersey	9255437	7.1	40.0	104804

	averageAnnualSunlight	giniCoefficient	AverageTemperature	\
0	4471.0	43.00	42.3	
1	3847.0	45.87	42.6	
2	3946.5	41.74	28.1	
3	5642.0	48.00	54.5	

4	4332.0	44.00	45.8
5	4960.0	45.90	46.3
6	4912.0	46.52	60.4
7	5296.0	45.00	51.1
8	3925.0	46.00	41.1
9	4725.0	47.00	61.1
10	4146.0	46.21	52.7
11	4251.0	44.57	44.0
12	3826.0	44.00	43.2
13	4887.0	43.00	49.3
14	5755.0	46.82	61.1
15	3815.0	45.00	41.9
16	3830.0	46.00	48.0
17	4890.0	45.55	55.1
18	4545.0	46.32	55.3
19	4383.0	47.41	56.4
20	4331.0	44.00	48.4
21	4486.0	47.86	58.5
22	4318.0	44.94	52.5
23	4693.0	48.00	64.3
24	4660.0	47.69	63.7
25	4661.0	48.16	64.3
26	3467.0	45.60	47.4
27	4624.0	46.90	63.4
28	3891.0	43.44	44.2
29	4023.0	44.00	44.0
30	4685.0	44.20	49.5
31	4725.0	49.03	67.2
32	4139.0	46.41	51.8
33	4018.0	46.00	45.3
34	5137.0	48.03	65.8
35	4859.0	49.00	71.5
36	3968.0	44.90	41.8
37	3939.0	46.80	49.6
38	4792.0	43.69	70.2
39	4232.0	45.00	56.3
40	4466.0	47.48	59.6
41	4354.0	46.73	56.1
42	4380.0	48.00	52.7
43	3989.0	47.38	50.8
44	5050.0	49.00	59.1
45	3988.0	49.00	50.0
46	4267.0	45.13	55.5
47	3944.0	48.26	48.9
48	3904.0	51.02	46.1
49	4056.0	47.82	53.6

	gunsRegistered	alcoholConsumptionGallons	Married	PercentDepression
0	132806	2.78	54.3	21.976786
1	22133	3.10	52.0	21.767857
2	15824	2.85	49.2	24.432143
3	97580	2.26	43.9	26.603571
4	21130	2.87	51.4	19.532143
5	92435	2.88	49.8	23.250000
6	71269	1.85	49.0	27.250000
7	76888	3.42	45.7	27.748214
8	13272	3.16	52.4	20.489286
9	79841	1.78	49.7	27.108929
10	35264	1.74	49.9	27.292857
11	49566	2.94	54.9	22.500000
12	5872	3.06	48.8	21.150000
13	72856	1.35	55.8	23.898214
14	179738	2.25	47.2	25.339286
15	15371	2.85	50.7	20.823214
16	61383	2.74	49.5	25.821429
17	52634	1.92	52.2	22.723214
18	72996	2.52	49.4	24.260714
19	81068	1.95	49.4	26.850000
20	28494	2.40	51.9	21.596429
21	99159	2.14	49.2	25.405357
22	114019	2.15	48.9	24.000000
23	35494	2.17	45.0	28.591071
24	161641	1.99	48.0	26.732143
25	190050	1.90	46.8	25.173214
26	91835	2.22	51.0	24.075000
27	105601	2.16	47.3	23.532143
28	64135	4.67	51.8	20.925000
29	64878	2.93	50.4	20.317857
30	22234	2.16	52.7	21.219643
31	116831	2.55	43.7	29.216071
32	173405	2.03	47.5	24.192857
33	65742	2.36	48.1	23.187500
34	588696	2.26	48.9	26.487500
35	343288	2.61	46.5	24.987500
36	79307	2.79	51.7	19.267857
37	236377	2.34	48.3	23.667857
38	7859	2.66	49.8	22.535714
39	4852	3.52	47.8	21.498214
40	152238	2.13	48.6	22.873214
41	307822	2.13	49.7	22.580357
42	146487	2.32	47.7	23.289286
43	37152	2.62	44.6	22.091071
44	344622	2.49	46.8	25.705357
45	82400	2.40	47.8	21.671429

46	103109	2.08	47.2	21.894643
47	37152	2.55	46.6	21.782143
48	76207	2.21	45.2	23.201786
49	57507	2.36	49.8	22.789286

1.0.5 Exploratory Analysis / Data Visualization

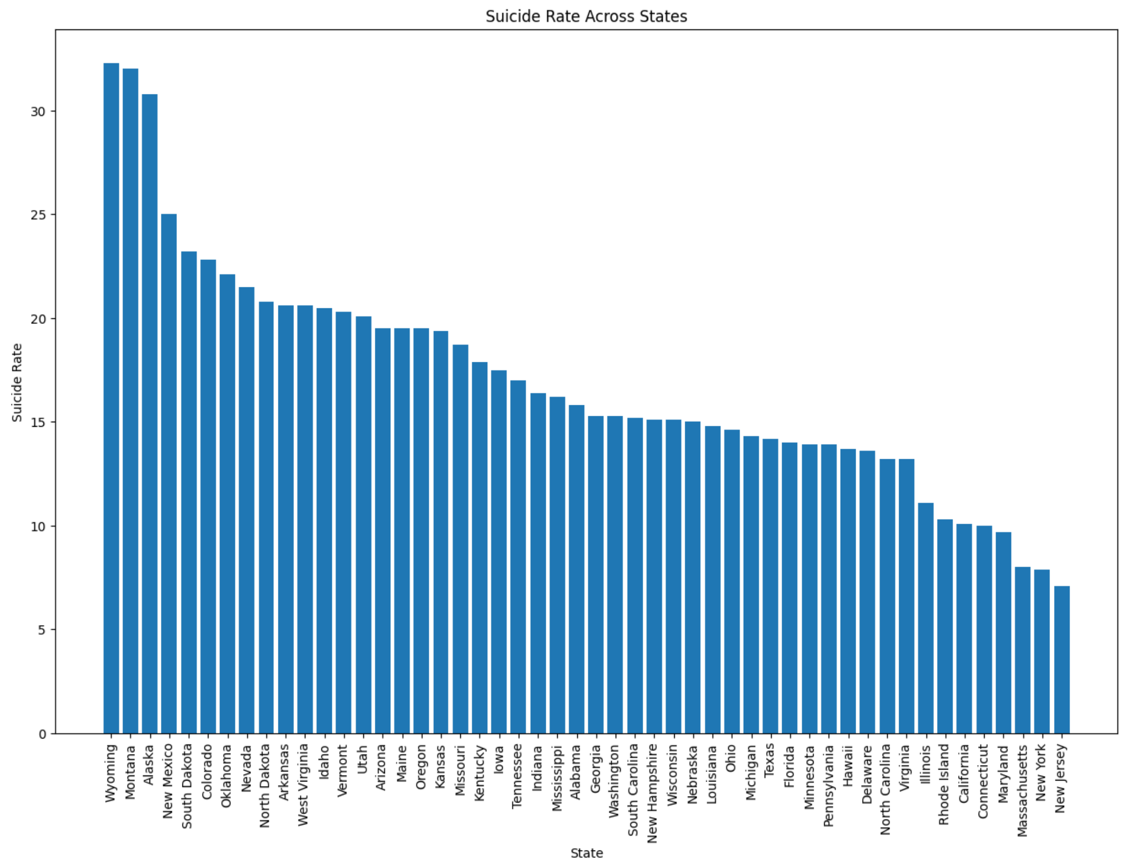
For the next step in the data science lifecycle, we want to create visualizations of our datasets so that any trends between different factors relating to suicide can stand out and be observed.

Exploratory Analysis of Suicide Rates Across States

To better understand the distribution of suicide rates across states, we can plot each of the suicide rates and rank them. As shown below, Wyoming has the highest suicide rate in the entire country, which is over triple the suicide rate of New Jersey, which has the lowest suicide rate in the country.

[9]:
```
plt.figure(figsize=(15,10))
plt.bar(suicideDf['state'], suicideDf['suicideRate'])
#Label the graph
plt.xticks(suicideDf['state'], suicideDf['state'], rotation='vertical')
plt.title('Suicide Rate Across States')
plt.xlabel('State')
plt.ylabel('Suicide Rate')
```

[9]:
```
Text(0, 0.5, 'Suicide Rate')
```



Data Visualization of Suicide Rates vs Other Factors Across States

Using the matplot library, (documentation found here) we can create scatter plots to plot all 50 states' suicide rates compared to other state variables. We also want to add lines of regression on each of the scatter plots to aid us in indicating any correlations between state suicide rates and other state variables.

In the plots displayed below, we can already begin to see some (loose) patterns between the suicide rate and some of the state variables. According to the regression lines, population size, family median income, guns registered, and Gini coefficient are negatively correlated with the suicide rate across states and average annual sunlight, marriage rate, alcohol consumption, and percent depression are positively correlated with the suicide rate across states. This information will aid us in narrowing down the causes of a high suicide rate.

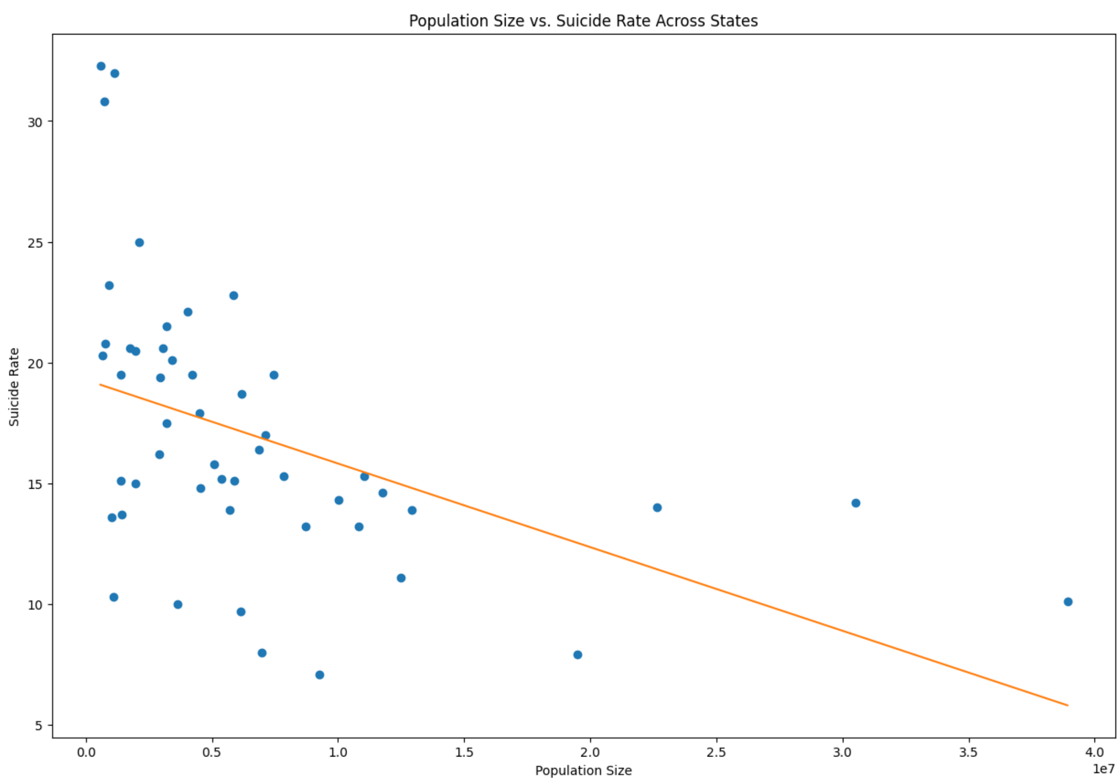
[10]:
```
# Plot for Population Size vs Suicide Rate
y = suicideDf['suicideRate'].values
x = suicideDf['pop2023'].values
# Make the plot with line of regression
z = np.polyfit(x, y, deg=1)
f = np.poly1d(z)
x2 = np.linspace(x.min(), x.max(), 100)
```

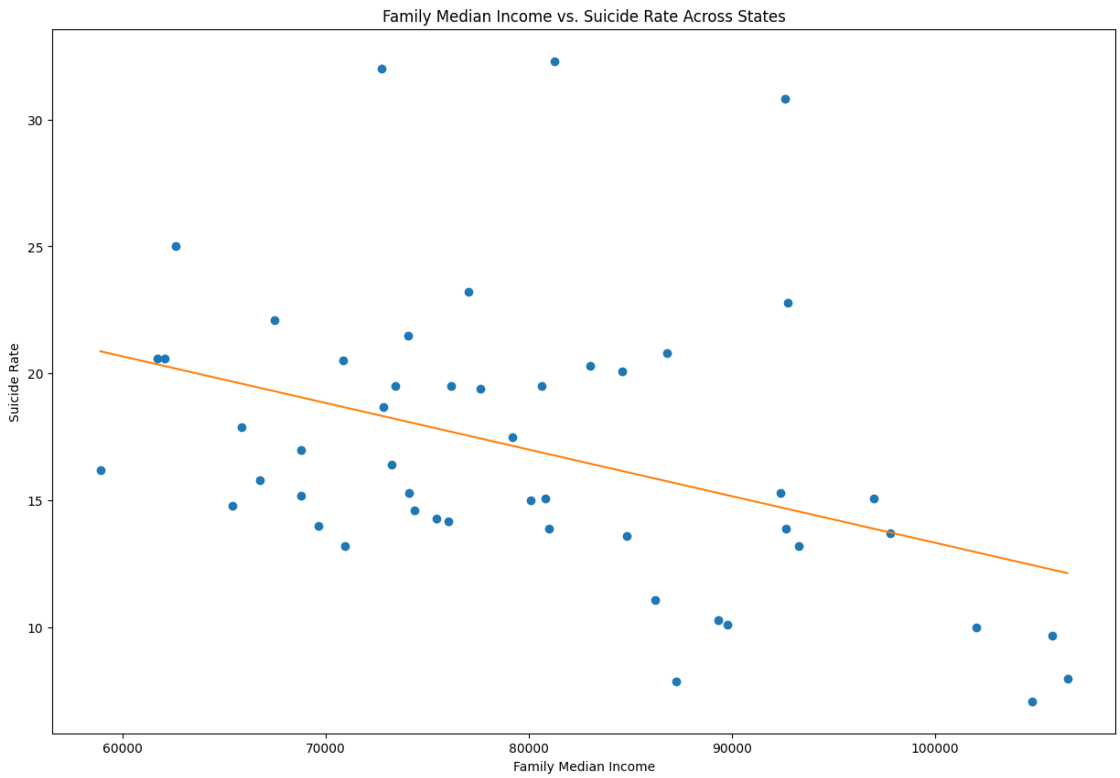
```python
y2 = f(x2)
plt.figure(figsize=(15,10))
# Display chart with points and regression line
plt.plot(x, y,'o', x2, y2)
plt.title("Population Size vs. Suicide Rate Across States")
plt.ylabel("Suicide Rate")
plt.xlabel("Population Size")

# Plot for Family Median Income vs Suicide Rate
y = suicideDf['suicideRate'].values
x = suicideDf['FamiliesMedianIncome'].values
# Make the plot with line of regression
z = np.polyfit(x, y, deg=1)
f = np.poly1d(z)
x2 = np.linspace(x.min(), x.max(), 100)
y2 = f(x2)
plt.figure(figsize=(15,10))
# Display chart with points and regression line
plt.plot(x, y,'o', x2, y2)
plt.title("Family Median Income vs. Suicide Rate Across States")
plt.ylabel("Suicide Rate")
plt.xlabel("Family Median Income")

# Plot for Average Annual Sunlight vs Suicide Rate
y = suicideDf['suicideRate'].values
x = suicideDf['averageAnnualSunlight'].values
# Make the plot with line of regression
z = np.polyfit(x, y, deg=1)
f = np.poly1d(z)
x2 = np.linspace(x.min(), x.max(), 100)
y2 = f(x2)
plt.figure(figsize=(15,10))
# Display chart with points and regression line
plt.plot(x, y,'o', x2, y2)
plt.title("Average Annual Sunlight vs. Suicide Rate Across States")
plt.ylabel("Suicide Rate")
plt.xlabel("Average Annual Sunlight")

# Plot for Guns Registered vs Suicide Rate
y = suicideDf['suicideRate'].values
x = suicideDf['gunsRegistered'].values
# Make the plot with line of regression
z = np.polyfit(x, y, deg=1)
f = np.poly1d(z)
x2 = np.linspace(x.min(), x.max(), 100)
y2 = f(x2)
plt.figure(figsize=(15,10))
```

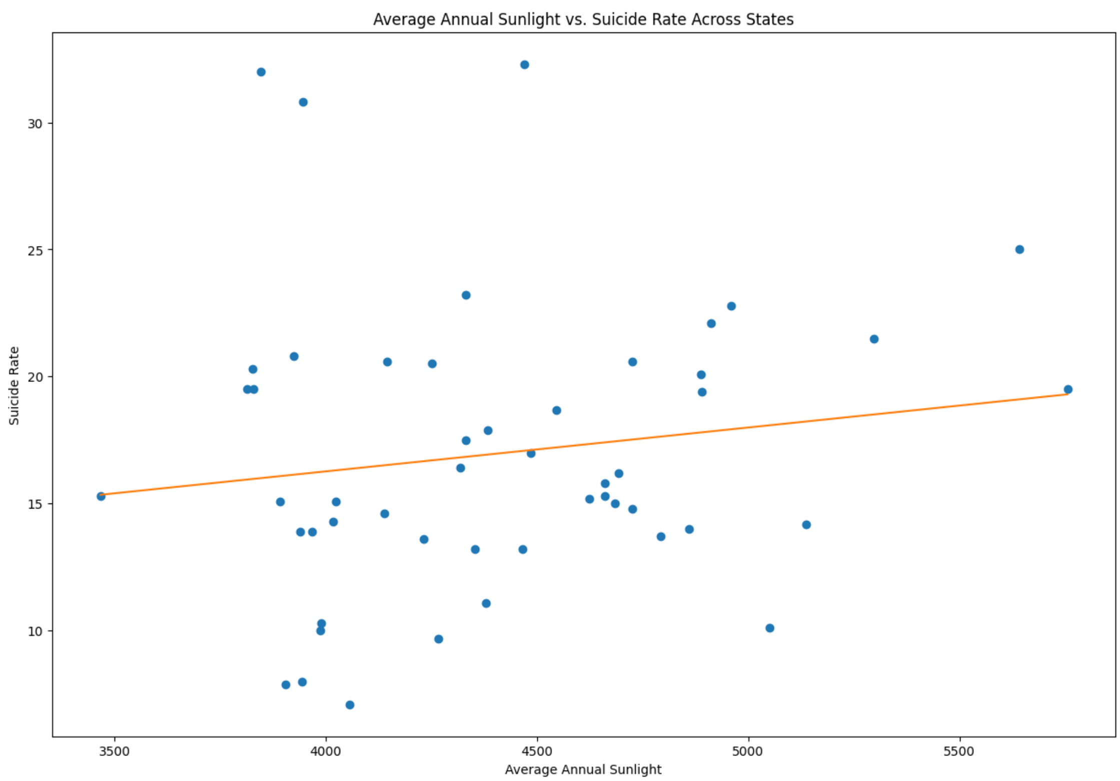
```python
# Display chart with points and regression line
plt.plot(x, y,'o', x2, y2)
plt.title("Guns Registered vs. Suicide Rate Across States")
plt.ylabel("Suicide Rate")
plt.xlabel("Guns Registered")

# Plot for Marriage Rate vs Suicide Rate
y = suicideDf['suicideRate'].values
x = suicideDf['Married'].values
# Make the plot with line of regression
z = np.polyfit(x, y, deg=1)
f = np.poly1d(z)
x2 = np.linspace(x.min(), x.max(), 100)
y2 = f(x2)
plt.figure(figsize=(15,10))
# Display chart with points and regression line
plt.plot(x, y,'o', x2, y2)
plt.title("Marriage Rate vs. Suicide Rate Across States")
plt.ylabel("Suicide Rate")
plt.xlabel("Marriage Rate")

# Plot for Gini Coefficient vs Suicide Rate
y = suicideDf['suicideRate'].values
x = suicideDf['giniCoefficient'].values
# Make the plot with line of regression
z = np.polyfit(x, y, deg=1)
f = np.poly1d(z)
x2 = np.linspace(x.min(), x.max(), 100)
y2 = f(x2)
plt.figure(figsize=(15,10))
# Display chart with points and regression line
plt.plot(x, y,'o', x2, y2)
plt.title("Gini Coefficient vs. Suicide Rate Across States")
plt.ylabel("Suicide Rate")
plt.xlabel("Gini Coefficient")

# Plot for Alcohol Consumption vs Suicide Rate
y = suicideDf['suicideRate'].values
x = suicideDf['alcoholConsumptionGallons'].values
# Make the plot with line of regression
z = np.polyfit(x, y, deg=1)
f = np.poly1d(z)
x2 = np.linspace(x.min(), x.max(), 100)
y2 = f(x2)
plt.figure(figsize=(15,10))
# Display chart with points and regression line
plt.plot(x, y,'o', x2, y2)
```

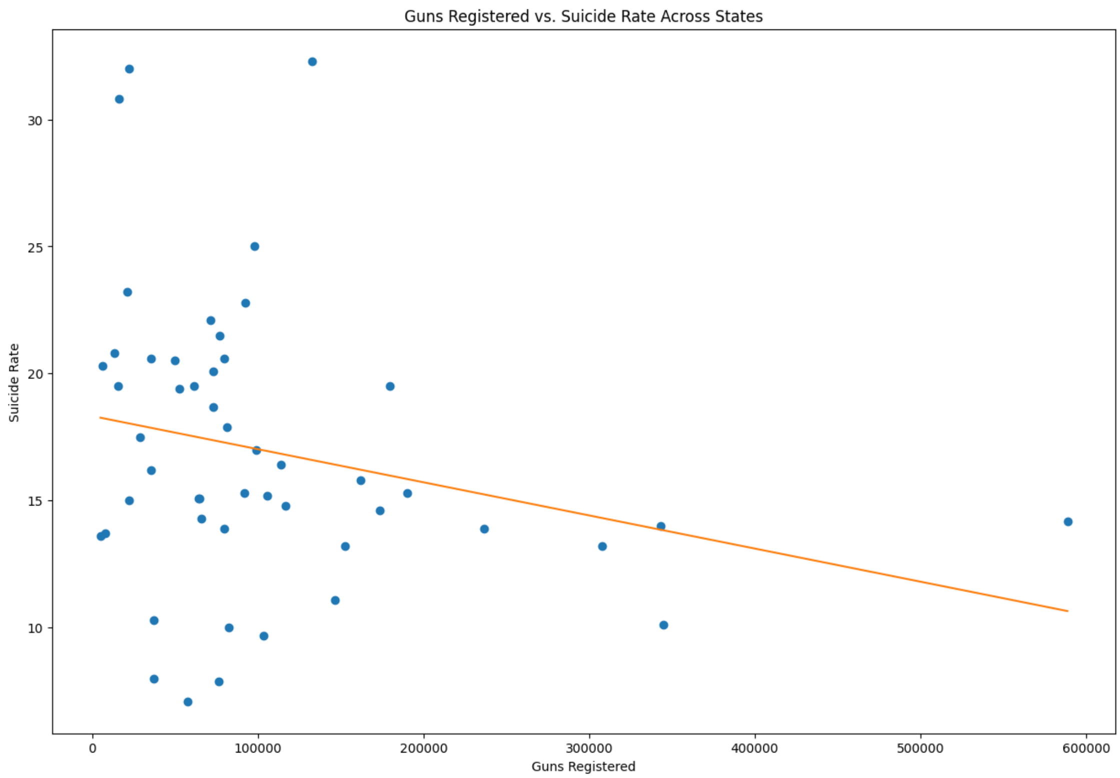
```
plt.title("Alcohol Consumption vs. Suicide Rate Across States")
plt.ylabel("Suicide Rate")
plt.xlabel("Alcohol Consumption")

# Plot for Percent Depression vs Suicide Rate
y = suicideDf['suicideRate'].values
x = suicideDf['PercentDepression'].values
# Make the plot with line of regression
z = np.polyfit(x, y, deg=1)
f = np.poly1d(z)
x2 = np.linspace(x.min(), x.max(), 100)
y2 = f(x2)
plt.figure(figsize=(15,10))
# Display chart with points and regression line
plt.plot(x, y,'o', x2, y2)
plt.title("Percent Depression vs. Suicide Rate Across States")
plt.ylabel("Suicide Rate")
plt.xlabel("Percent Depression")
```

[10]: Text(0.5, 0, 'Percent Depression')

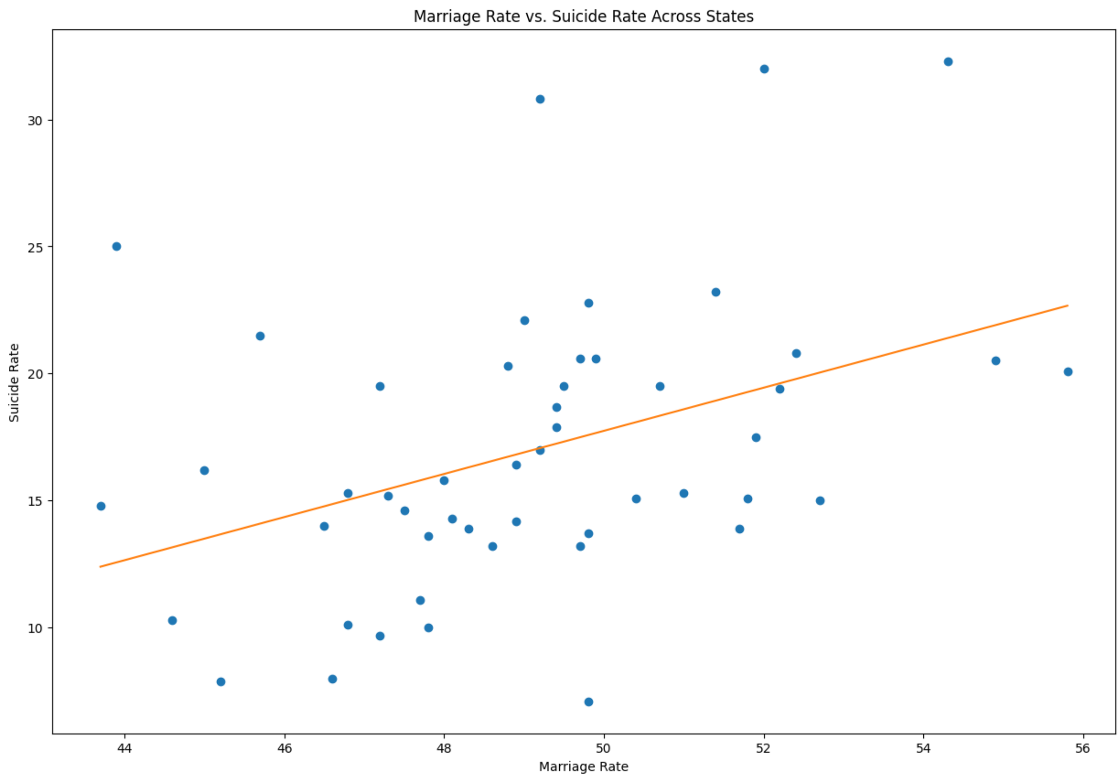


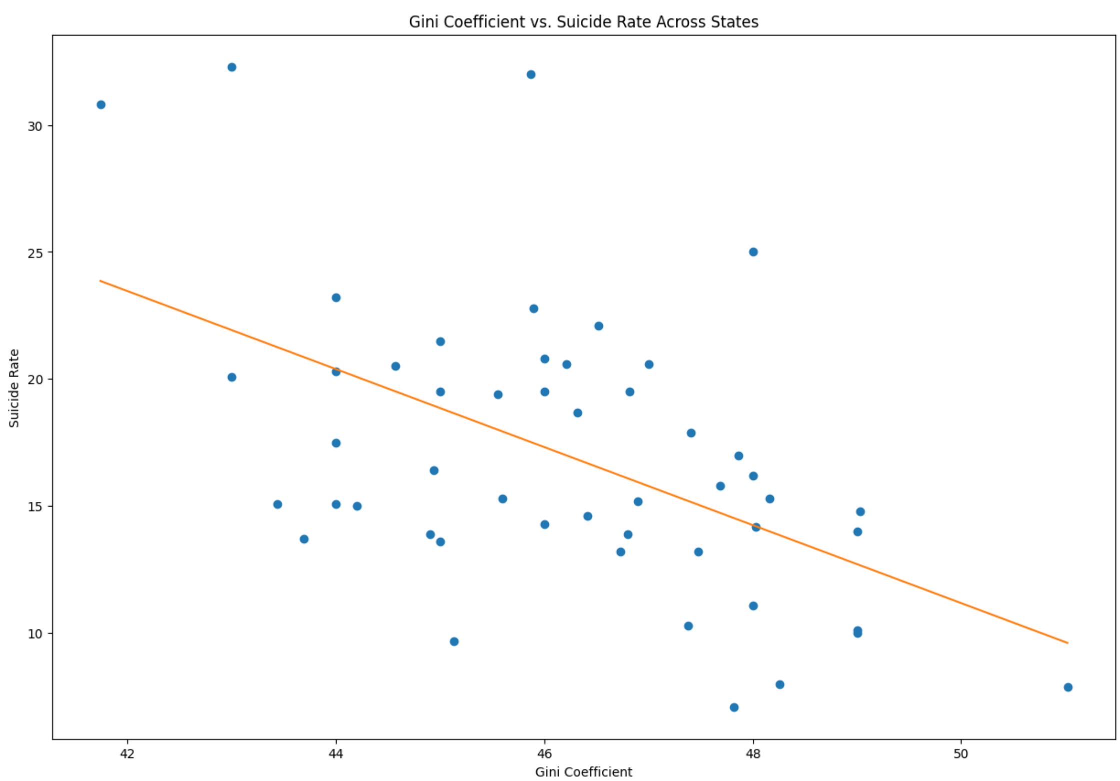

Family Median Income vs. Suicide Rate Across States
Suicide Rate
30
25
20
15
10
60000
70000
80000
90000
100000
Family Median Income

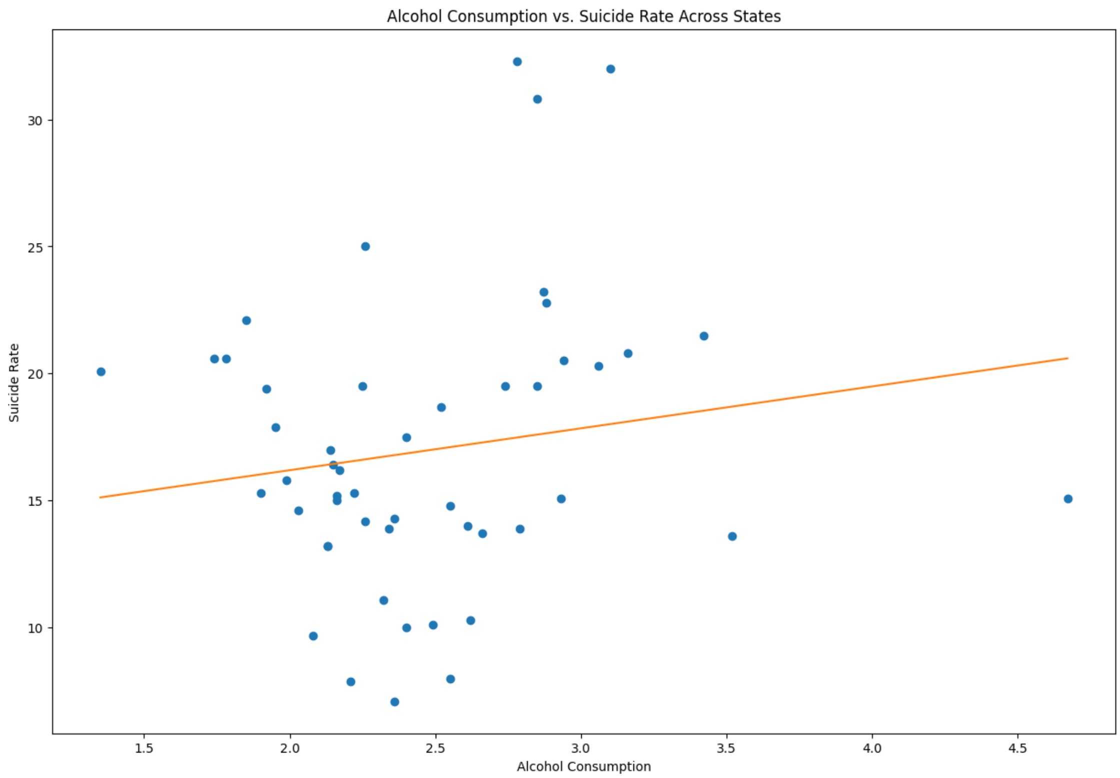

Average Annual Sunlight vs. Suicide Rate Across States
Suicide Rate
30
25
20
15
10
3500
4000
4500
5000
5500
Average Annual Sunlight

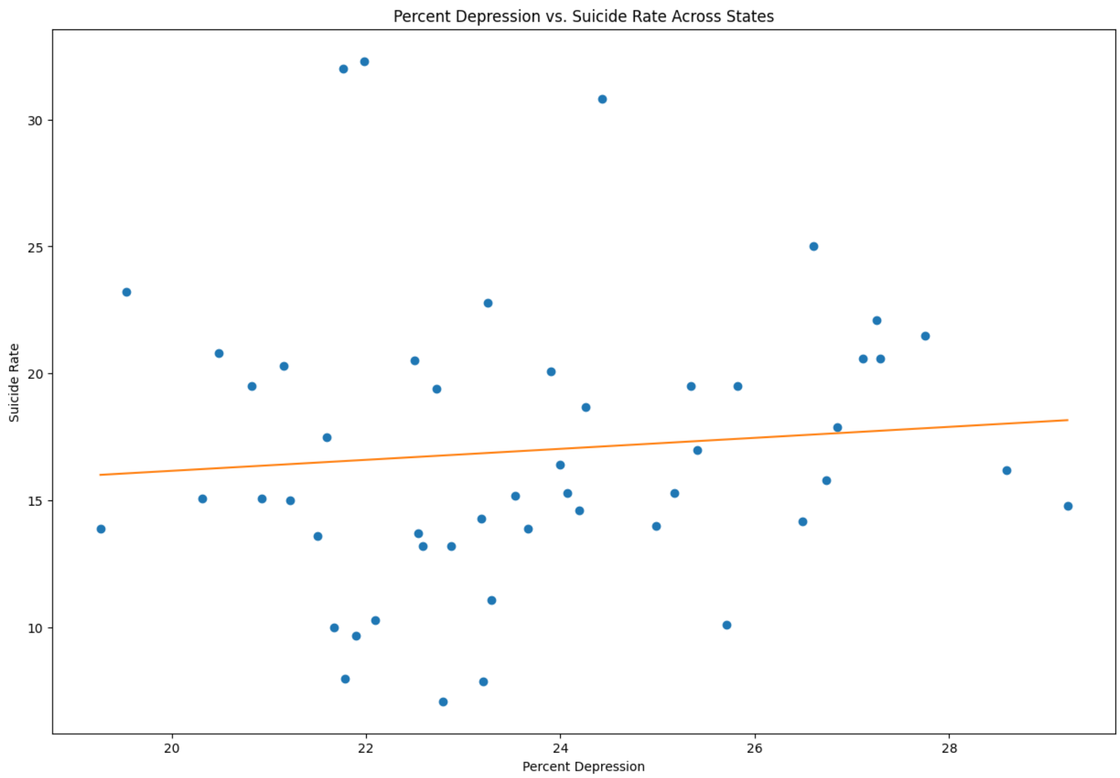


Guns Registered vs. Suicide Rate Across States
Suicide Rate
30
25
20
15
10
0
100000
200000
300000
400000
500000
600000
Guns Registered


Marriage Rate vs. Suicide Rate Across States
Suicide Rate
Marriage Rate
10
15
20
25
30
44
46
48
50
52
54
56


Gini Coefficient vs. Suicide Rate Across States
Suicide Rate
Gini Coefficient
10
15
20
25
30
42
44
46
48
50


Alcohol Consumption vs. Suicide Rate Across States
Suicide Rate
Alcohol Consumption
10
15
20
25
30
1.5
2.0
2.5
3.0
3.5
4.0
4.5



We can expand more upon the data visualization by using statsmodel to calculate OLS regression results between the suicide rate and each of the other state variables. Using this information, we can determine the variables that have the highest correlations with the suicide rate. As shown below, the Gini coefficient has the largest slope intercept to the suicide rate compared to the other variables with a coefficient of 1.648573.

[11]:
```
# Print regression summaries for each of the other state variables
print(ols(formula = 'suicideRate ~ pop2023', data = suicideDf).fit().params)
print(ols(formula = 'suicideRate ~ FamiliesMedianIncome', data = suicideDf).
 ↪fit().params)
print(ols(formula = 'suicideRate ~ averageAnnualSunlight', data = suicideDf).
 ↪fit().params)
print(ols(formula = 'suicideRate ~ gunsRegistered', data = suicideDf).fit().
 ↪params)
print(ols(formula = 'suicideRate ~ Married', data = suicideDf).fit().params)
print(ols(formula = 'suicideRate ~ giniCoefficient', data = suicideDf).fit().
 ↪params)
print(ols(formula = 'suicideRate ~ PercentDepression', data = suicideDf).fit().
 ↪params)
print(ols(formula = 'suicideRate ~ alcoholConsumptionGallons', data =␣
 ↪suicideDf).fit().params)
```

```
Intercept     1.928395e+01
```

```
pop2023     -3.462366e-07
dtype: float64
Intercept                31.684036
FamiliesMedianIncome    -0.000183
dtype: float64
Intercept                 9.356124
averageAnnualSunlight    0.001728
dtype: float64
Intercept          18.324715
gunsRegistered    -0.000013
dtype: float64
Intercept   -24.706903
Married       0.849030
dtype: float64
Intercept          87.929813
giniCoefficient    -1.534966
dtype: float64
Intercept            11.843199
PercentDepression     0.216349
dtype: float64
Intercept                    12.897716
alcoholConsumptionGallons     1.648573
dtype: float64
```

1.0.6 Model Analysis: Hypothesis Testing

When determining whether or not our state variables affect the suicide rate, our null hypothesis is that the given state variable does not affect the state's suicide rate. Our goal is to determine if we can reject this null hypothesis for any of the state variables.

After using the statsmodel to generate the OLS Regression results between suicide rate and each of the state variables shown below, we can see how related the two variables are through the p-values. Family median income and marriage rate both have p-values of 0.004, population size has a p-value of 0.001, and the Gini coefficient has a p-value of 0, which is less than 0.05, indicating to us that we can reject the null hypothesis and conclude that these variables potentially affect the suicide rate. Average annual sunlight, guns registered, depression percentage, and alcohol consumption have p-values of 0.122, 0.085, 0.521, and 0.266 respectively, which are all greater than 0.05, indicating to us that we cannot reject the null hypothesis that these variables do not affect the suicide rate.

[12]:
```
print(ols(formula = 'suicideRate ~ pop2023', data = suicideDf).fit().summary())
print(ols(formula = 'suicideRate ~ FamiliesMedianIncome', data = suicideDf).
 ↪fit().summary())
print(ols(formula = 'suicideRate ~ averageAnnualSunlight', data = suicideDf).
 ↪fit().summary())
print(ols(formula = 'suicideRate ~ gunsRegistered', data = suicideDf).fit().
 ↪summary())
print(ols(formula = 'suicideRate ~ Married', data = suicideDf).fit().summary())
```

```python
print(ols(formula = 'suicideRate ~ giniCoefficient', data = suicideDf).fit().
 ↪summary())
print(ols(formula = 'suicideRate ~ PercentDepression', data = suicideDf).fit().
 ↪summary())
print(ols(formula = 'suicideRate ~ alcoholConsumptionGallons', data =␣
 ↪suicideDf).fit().summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:            suicideRate   R-squared:                       0.212
Model:                            OLS   Adj. R-squared:                  0.195
Method:                 Least Squares   F-statistic:                     12.89
Date:                Thu, 11 May 2023   Prob (F-statistic):           0.000773
Time:                        23:14:18   Log-Likelihood:                -150.89
No. Observations:                  50   AIC:                             305.8
Df Residuals:                      48   BIC:                             309.6
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     19.2840      0.962     20.056      0.000      17.351      21.217
pop2023    -3.462e-07   9.64e-08     -3.591      0.001     -5.4e-07   -1.52e-07
==============================================================================
Omnibus:                        5.075   Durbin-Watson:                   0.474
Prob(Omnibus):                  0.079   Jarque-Bera (JB):                3.936
Skew:                           0.575   Prob(JB):                        0.140
Kurtosis:                       3.754   Cond. No.                     1.34e+07
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 1.34e+07. This might indicate that there are
strong multicollinearity or other numerical problems.
                            OLS Regression Results
==============================================================================
Dep. Variable:            suicideRate   R-squared:                       0.158
Model:                            OLS   Adj. R-squared:                  0.141
Method:                 Least Squares   F-statistic:                     9.010
Date:                Thu, 11 May 2023   Prob (F-statistic):            0.00425
Time:                        23:14:18   Log-Likelihood:                -152.54
No. Observations:                  50   AIC:                             309.1
Df Residuals:                      48   BIC:                             312.9
Df Model:                           1
Covariance Type:            nonrobust
================================================================================
```

```
========
                          coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
--------
Intercept              31.6840      4.956      6.392      0.000      21.718
41.650
FamiliesMedianIncome   -0.0002   6.11e-05     -3.002      0.004      -0.000
-6.06e-05
==============================================================================
Omnibus:                       20.489   Durbin-Watson:                   0.319
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               27.750
Skew:                           1.453   Prob(JB):                     9.42e-07
Kurtosis:                       5.207   Cond. No.                     5.45e+05
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 5.45e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
                            OLS Regression Results
==============================================================================
Dep. Variable:            suicideRate   R-squared:                       0.023
Model:                            OLS   Adj. R-squared:                  0.002
Method:                 Least Squares   F-statistic:                     1.118
Date:                Thu, 11 May 2023   Prob (F-statistic):              0.296
Time:                        23:14:18   Log-Likelihood:                -156.27
No. Observations:                  50   AIC:                             316.5
Df Residuals:                      48   BIC:                             320.4
Df Model:                           1
Covariance Type:            nonrobust
================================================================================
=========
                           coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
---------
Intercept                9.3561      7.246      1.291      0.203      -5.214
23.926
averageAnnualSunlight    0.0017      0.002      1.057      0.296      -0.002
0.005
==============================================================================
Omnibus:                       11.260   Durbin-Watson:                   0.101
Prob(Omnibus):                  0.004   Jarque-Bera (JB):               11.349
Skew:                           0.964   Prob(JB):                      0.00343
Kurtosis:                       4.314   Cond. No.                     4.04e+04
==============================================================================
```

```
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 4.04e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
                            OLS Regression Results
==============================================================================
Dep. Variable:            suicideRate   R-squared:                       0.061
Model:                            OLS   Adj. R-squared:                  0.041
Method:                 Least Squares   F-statistic:                     3.096
Date:                Thu, 11 May 2023   Prob (F-statistic):             0.0848
Time:                        23:14:18   Log-Likelihood:                -155.28
No. Observations:                  50   AIC:                             314.6
Df Residuals:                      48   BIC:                             318.4
Df Model:                           1
Covariance Type:            nonrobust
================================================================================
==
                     coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
--
Intercept         18.3247      1.095     16.737      0.000      16.123
20.526
gunsRegistered -1.304e-05   7.41e-06     -1.760      0.085   -2.79e-05
1.86e-06
==============================================================================
Omnibus:                        6.784   Durbin-Watson:                   0.141
Prob(Omnibus):                  0.034   Jarque-Bera (JB):                5.873
Skew:                           0.653   Prob(JB):                       0.0530
Kurtosis:                       4.054   Cond. No.                     2.08e+05
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.08e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
                            OLS Regression Results
==============================================================================
Dep. Variable:            suicideRate   R-squared:                       0.163
Model:                            OLS   Adj. R-squared:                  0.145
Method:                 Least Squares   F-statistic:                     9.342
Date:                Thu, 11 May 2023   Prob (F-statistic):            0.00365
Time:                        23:14:18   Log-Likelihood:                -152.40
No. Observations:                  50   AIC:                             308.8
Df Residuals:                      48   BIC:                             312.6
```

```
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept    -24.7069     13.656     -1.809      0.077     -52.165       2.751
Married        0.8490      0.278      3.056      0.004       0.291       1.408
==============================================================================
Omnibus:                        8.370   Durbin-Watson:                   0.370
Prob(Omnibus):                  0.015   Jarque-Bera (JB):                7.499
Skew:                           0.878   Prob(JB):                       0.0235
Kurtosis:                       3.716   Cond. No.                         913.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
                            OLS Regression Results
==============================================================================
Dep. Variable:            suicideRate   R-squared:                       0.274
Model:                            OLS   Adj. R-squared:                  0.259
Method:                 Least Squares   F-statistic:                     18.10
Date:                Thu, 11 May 2023   Prob (F-statistic):           9.66e-05
Time:                        23:14:19   Log-Likelihood:                -148.84
No. Observations:                  50   AIC:                             301.7
Df Residuals:                      48   BIC:                             305.5
Df Model:                           1
Covariance Type:            nonrobust
================================================================================
===
                      coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
---
Intercept          87.9298     16.694      5.267      0.000      54.364
121.496
giniCoefficient    -1.5350      0.361     -4.254      0.000      -2.260
-0.809
==============================================================================
Omnibus:                        5.783   Durbin-Watson:                   0.582
Prob(Omnibus):                  0.055   Jarque-Bera (JB):                4.645
Skew:                           0.670   Prob(JB):                       0.0980
Kurtosis:                       3.659   Cond. No.                     1.13e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

```
[2] The condition number is large, 1.13e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
                            OLS Regression Results
==============================================================================
Dep. Variable:            suicideRate   R-squared:                       0.009
Model:                            OLS   Adj. R-squared:                 -0.012
Method:                 Least Squares   F-statistic:                    0.4188
Date:                Thu, 11 May 2023   Prob (F-statistic):              0.521
Time:                        23:14:19   Log-Likelihood:                -156.62
No. Observations:                  50   AIC:                             317.2
Df Residuals:                      48   BIC:                             321.1
Df Model:                           1
Covariance Type:            nonrobust
================================================================================
=====
                        coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
-----
Intercept            11.8432      7.966      1.487      0.144      -4.173
27.859
PercentDepression     0.2163      0.334      0.647      0.521      -0.456
0.889
==============================================================================
Omnibus:                        9.444   Durbin-Watson:                   0.052
Prob(Omnibus):                  0.009   Jarque-Bera (JB):                8.825
Skew:                           0.881   Prob(JB):                       0.0121
Kurtosis:                       4.064   Cond. No.                         237.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
                            OLS Regression Results
==============================================================================
Dep. Variable:            suicideRate   R-squared:                       0.026
Model:                            OLS   Adj. R-squared:                  0.005
Method:                 Least Squares   F-statistic:                     1.265
Date:                Thu, 11 May 2023   Prob (F-statistic):              0.266
Time:                        23:14:19   Log-Likelihood:                -156.19
No. Observations:                  50   AIC:                             316.4
Df Residuals:                      48   BIC:                             320.2
Df Model:                           1
Covariance Type:            nonrobust
================================================================================
=============
                                coef    std err          t      P>|t|
[0.025      0.975]
```

```
--------------------------------------------------------------------------------
-------------
Intercept                    12.8977      3.709      3.478      0.001
5.441      20.355
alcoholConsumptionGallons     1.6486      1.466      1.125      0.266
-1.299       4.596
==============================================================================
Omnibus:                        5.382   Durbin-Watson:                   0.082
Prob(Omnibus):                  0.068   Jarque-Bera (JB):                4.299
Skew:                           0.672   Prob(JB):                        0.117
Kurtosis:                       3.507   Cond. No.                         13.6
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

1.0.7 Model Analysis: Machine Learning

In this next section of the tutorial, we want to utilize a machine learning algorithm to develop a model that could be applied to measure suicide rates in other population groups.

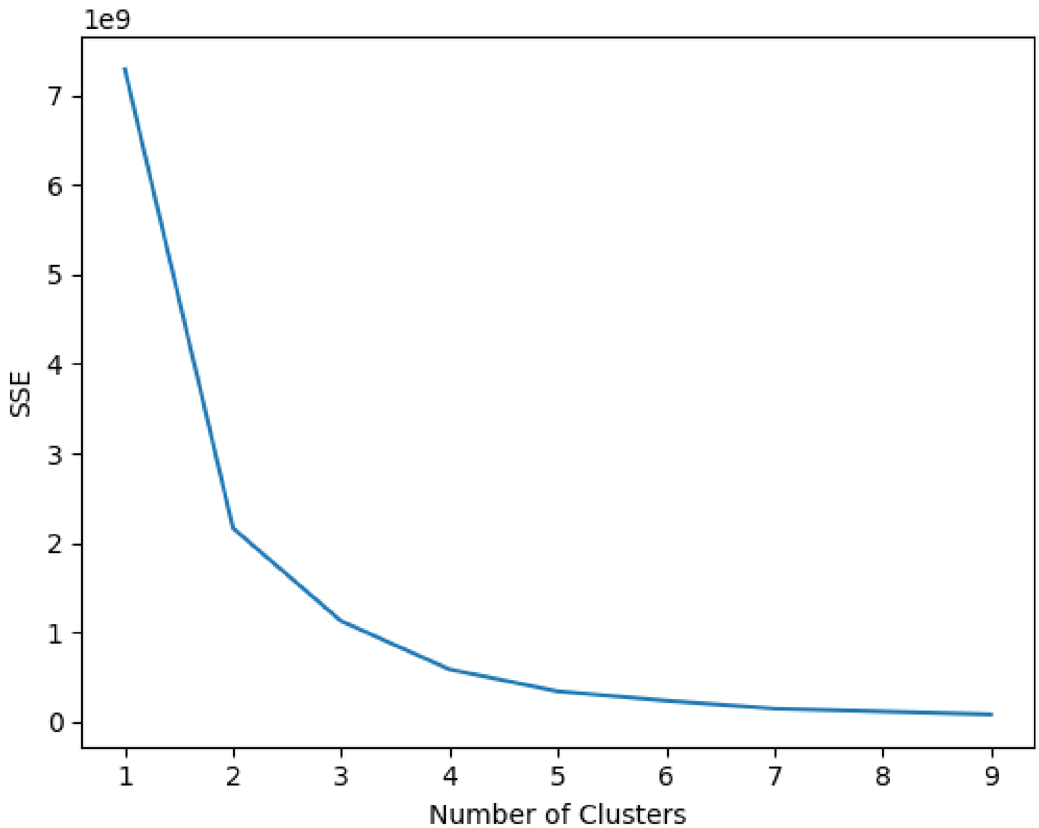
Our first step is to further process the data by standardizing numerical columns. To do this we subtract each value from the mean and divide by the standard deviation. This should help reduce the impact of differences in units and variances between each feature we are including. More can be read about when it is important to standardize data here: https://builtin.com/data-science/when-and-why-standardize-your-data

[13]:
```python
suicideDf['pop2023'] =( suicideDf['pop2023'] - suicideDf['pop2023'].mean() ) /␣
 ↪suicideDf['pop2023'].std()

suicideDf['suicideRate'] =( suicideDf['suicideRate'] - suicideDf['suicideRate'].
 ↪mean() ) / suicideDf['suicideRate'].std()

suicideDf['MedianAge'] =( suicideDf['MedianAge'] - suicideDf['MedianAge'].
 ↪mean() ) / suicideDf['MedianAge'].std()

suicideDf['averageAnnualSunlight'] =( suicideDf['averageAnnualSunlight'] -␣
 ↪suicideDf['averageAnnualSunlight'].mean() ) /␣
 ↪suicideDf['averageAnnualSunlight'].std()

suicideDf['giniCoefficient'] =( suicideDf['giniCoefficient'] -␣
 ↪suicideDf['giniCoefficient'].mean() ) / suicideDf['giniCoefficient'].std()

suicideDf['AverageTemperature'] =( suicideDf['AverageTemperature'] -␣
 ↪suicideDf['AverageTemperature'].mean() ) / suicideDf['AverageTemperature'].
 ↪std()
```

```
suicideDf['gunsRegistered'] =( suicideDf['gunsRegistered'] -␣
 ↪suicideDf['gunsRegistered'].mean() ) / suicideDf['gunsRegistered'].std()

suicideDf['alcoholConsumptionGallons'] =(␣
 ↪suicideDf['alcoholConsumptionGallons'] -␣
 ↪suicideDf['alcoholConsumptionGallons'].mean() ) /␣
 ↪suicideDf['gunsRegistered'].std()

suicideDf['Married'] =( suicideDf['Married'] - suicideDf['Married'].mean() ) /␣
 ↪suicideDf['Married'].std()

suicideDf['PercentDepression'] =( suicideDf['PercentDepression'] -␣
 ↪suicideDf['PercentDepression'].mean() ) / suicideDf['PercentDepression'].
 ↪std()
```

Shown below is an elbow plot of various numbers of k-means clusters, ranging from 1 to 9 clusters compared against their associated sum of squares error (SSE). We want to utilize the sum of squares error because it provides us with insight into how far elements are from their cluster centers. The lower sum of squares error is, the more accurate our clustering model will be. Our goal is to select a good number of clusters to work with so the number of clusters is minimized and the sum of squares error is minimized. In our case, we would want to use 3 or 4 clusters.

[14]:
```
suicideDfClustering = suicideDf.drop(columns=['state'])
sse = [] #Sum of squares error
# Fit KMeans for # of clusters 1-10 and store the SSE
for k in range(1, 10):
    kmeans = KMeans(n_clusters=k).fit(suicideDfClustering)
    suicideDfClustering["labels"] = kmeans.labels_
    sse.append(kmeans.inertia_)
plt.figure()
# Plot and elbow plot of SSE vs. Number of Clusters
plt.plot(range(1, 10), sse)
plt.xlabel("Number of Clusters")
plt.ylabel("SSE")
plt.show()
```

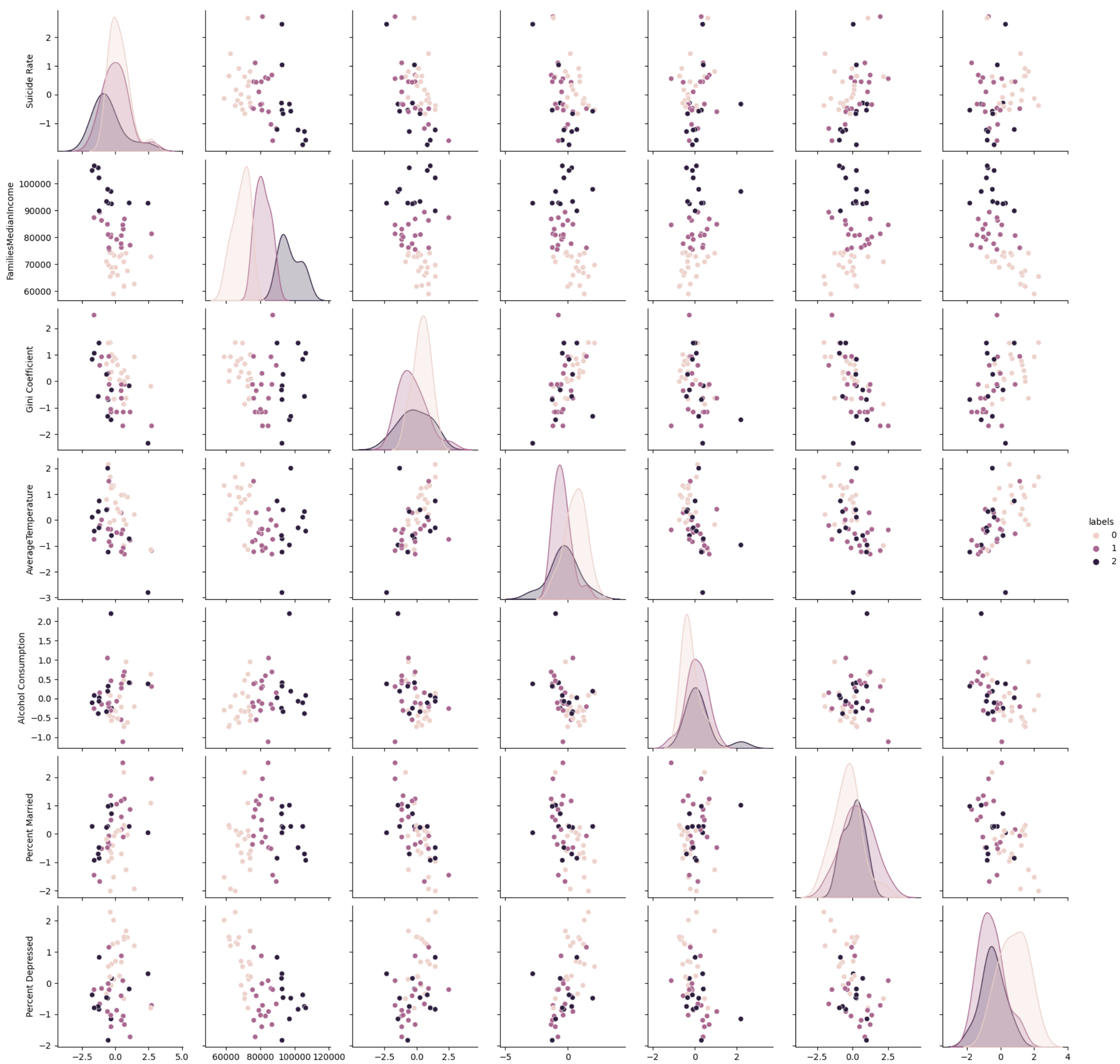


Next, we plotted the clustered data for each pair of data we encountered to see how those pairs relate to our clustering. Here we can compare each feature pairwise to see how much the clustering was impacted by that feature. Most importantly, we can look at how the other features cluster in relation to the suicide rate. Visually, it looks like median income and Gini Coefficient (income inequality) show the clearest separation in relation to the suicide rate.

[15]:
```
kmeans = KMeans(n_clusters=3).fit(suicideDfClustering)
suicideDfClustering['labels'] = kmeans.labels_

suicideDfClustering.rename(columns={
    'pop2023': '2023 Population',
    'suicideRate': 'Suicide Rate',
    'MedianAge': 'Median Age',
    'averageAnnualSunlight': 'Avg Annual Sunlight',
    'giniCoefficient': 'Gini Coefficient',
    'gunsRegistered': 'Guns Registered',
    'alcoholConsumptionGallons': 'Alcohol Consumption',
    'Married': 'Percent Married',
    'PercentDepression': 'Percent Depressed'
}, inplace=True)
```

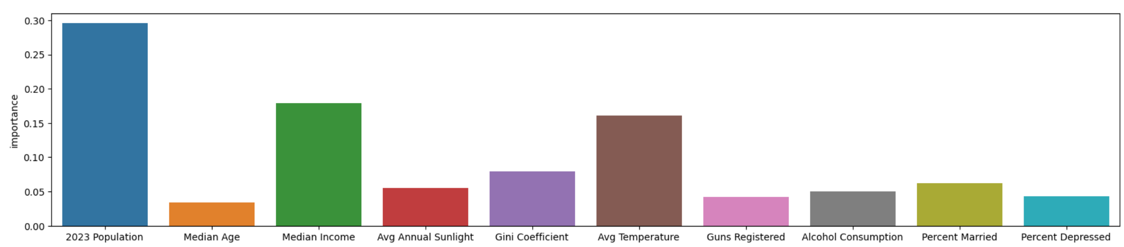
```
suicideDfClustering.drop(columns=['2023 Population', 'Avg Annual Sunlight',␣
 ↪'Guns Registered', 'Median Age'], inplace=True)

plt.figure(figsize=(14,14))
sb.pairplot(suicideDfClustering, hue='labels')
plt.show()
```

```
<Figure size 1400x1400 with 0 Axes>
```



We then trained a Random Forest Regression model on our data, which could then serve as a predictive model for suicide rate based on the other factors in our data set. Random Forest is a method of machine learning which generates many decision trees by randomly sampling the dataset with replacement. While a single decision tree is prone to overfitting the data, this random forest of trees is more likely to average out to an accurate prediction. We were interested in seeing the most important features as determined by this model, which we plotted below.

[16]:
```
X = suicideDf.drop(columns=['state', 'suicideRate'])
y = suicideDf['suicideRate']

REG = RandomForestRegressor(n_estimators=100).fit(X = X, y = y)
featureImportance = pd.DataFrame()
featureImportance['name'] = REG.feature_names_in_
featureImportance['importance'] = REG.feature_importances_

featureImportance['name'] = featureImportance['name'].replace({
    'pop2023': '2023 Population',
    'suicideRate': 'Suicide Rate',
    'MedianAge': 'Median Age',
    'averageAnnualSunlight': 'Avg Annual Sunlight',
    'giniCoefficient': 'Gini Coefficient',
    'gunsRegistered': 'Guns Registered',
    'alcoholConsumptionGallons': 'Alcohol Consumption',
    'Married': 'Percent Married',
    'PercentDepression': 'Percent Depressed',
    'FamiliesMedianIncome': 'Median Income',
    'AverageTemperature': 'Avg Temperature'
})

plt.figure(figsize=(20,4))
sb.barplot(data = featureImportance, x = 'name', y = 'importance')
plt.show()
```



1.0.8 Insight / Conclusion

This is the last part of the data science lifecycle. At this point, the data has been analyzed and now we can make some conclusions about the results.

From our data analysis, we found that the Gini coefficient was the factor that had the strongest correlation with suicide rates per state as the OLS regression results indicated a p-value of 0. The Gini coefficient has a negative relationship with the suicide rate meaning that the more income inequality there is, the lower the suicide rate. Another factor that had a strong correlation with the suicide rate per state was Family Median Income which had a p-value of 0.004. Family Median income has a negative relationship with the suicide rate meaning that the more money a family has, the lower their risk for suicide becomes. This could be because wealthier people have more access to mental health services and will therefore be less inclined to commit suicide.

The results of our machine-learning model indicated that population, median income, and average temperature were the strongest predictors of suicide rates. Most of the factors we analyzed were not strong predictors in the random forests model. Since so many things factor into what causes a person to choose to take their own life, this model may not have much use in predicting suicide rates on a state-wide scale.

We hope this tutorial helps better your understanding of the data science lifecycle and how it can be applied to better understand risk factors for suicide. For more information on some parts of the data science lifecycle check the following links:

Data Processing: https://www.tutorialspoint.com/basics_of_computer_science/basics_of_computer_science_da

Hypothesis Testing: https://www.statisticshowto.com/probability-and-statistics/hypothesis-testing/

Random Forests: https://towardsdatascience.com/understanding-random-forest-58381e0602d2

Clustering in Machine Learning: https://www.geeksforgeeks.org/clustering-in-machine-learning/

If you are someone you know has experienced suicidal thoughts the 24/7 suicide and crisis hotline number is 988 or you can access the hotline's official website here. To learn more about suicide and its risk factors the CDC and the National Institute of Health have webpages providing facts on signs of suicide risk and tips for suicide prevention.

Created in Deepnote