Project 1: SAT & ACT Analysis

The first markdown cell in a notebook is a great place to provide an overview of your entire project. You will likely want to at least state your

Problem Statement

as well as an

Executive Summary

If you want to, it's great to use relative links to direct your audience to various sections of a notebook. **HERE'S A DEMONSTRATION WITH THE CURRENT SECTION HEADERS**:

Contents:

If you combine your problem statement, executive summary, data dictionary, and conclusions/recommendations, you have an amazing README.md file that quickly aligns your audience to the contents of your project. Don't forget to cite your data sources!

All libraries used should be added here

In [402]:
```
#Imports:

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

2017 Data Import and Cleaning

1. Read In SAT & ACT Data

Read in the `sat_2017.csv` and `act_2017.csv` files and assign them to appropriately named pandas dataframes.

In [403]:
```
#Code:

act17 = pd.read_csv('/Users/rdk/Desktop/GA/GA_DSI_10/Projects/project_1/dat
sat17 = pd.read_csv('/Users/rdk/Desktop/GA/GA_DSI_10/Projects/project_1/dat
```

2. Display Data

Print the first 10 rows of each dataframe to your jupyter notebook

In [404]:
```
#Code:

act17.head(10)
```

Out[404]:

	State	Participation	English	Math	Reading	Science	Composite
0	National	60%	20.3	20.7	21.4	21.0	21.0
1	Alabama	100%	18.9	18.4	19.7	19.4	19.2
2	Alaska	65%	18.7	19.8	20.4	19.9	19.8
3	Arizona	62%	18.6	19.8	20.1	19.8	19.7
4	Arkansas	100%	18.9	19.0	19.7	19.5	19.4
5	California	31%	22.5	22.7	23.1	22.2	22.8
6	Colorado	100%	20.1	20.3	21.2	20.9	20.8
7	Connecticut	31%	25.5	24.6	25.6	24.6	25.2
8	Delaware	18%	24.1	23.4	24.8	23.6	24.1
9	District of Columbia	32%	24.4	23.5	24.9	23.5	24.2

In [405]:
```
sat17.head(10)
```

Out[405]:

	State	Participation	Evidence-Based Reading and Writing	Math	Total
0	Alabama	5%	593	572	1165
1	Alaska	38%	547	533	1080
2	Arizona	30%	563	553	1116
3	Arkansas	3%	614	594	1208
4	California	53%	531	524	1055
5	Colorado	11%	606	595	1201
6	Connecticut	100%	530	512	1041
7	Delaware	100%	503	492	996
8	District of Columbia	100%	482	468	950
9	Florida	83%	520	497	1017

In [406]:
```
sat17.describe()
```

Out[406]:

	Evidence-Based Reading and Writing	Math	Total
count	51.000000	51.000000	51.000000
mean	569.117647	547.627451	1126.098039
std	45.666901	84.909119	92.494812
min	482.000000	52.000000	950.000000
25%	533.500000	522.000000	1055.500000
50%	559.000000	548.000000	1107.000000
75%	613.000000	599.000000	1212.000000
max	644.000000	651.000000	1295.000000

In [407]:
```
act17.describe()
```

Out[407]:

	English	Math	Reading	Science
count	52.000000	52.000000	52.000000	52.000000
mean	20.919231	21.173077	22.001923	21.040385
std	2.332132	1.963602	2.048672	3.151113
min	16.300000	18.000000	18.100000	2.300000
25%	19.000000	19.400000	20.475000	19.900000
50%	20.550000	20.900000	21.700000	21.150000
75%	23.300000	23.100000	24.125000	22.525000
max	25.500000	25.300000	26.000000	24.900000

In [ ]:

3. Verbally Describe Data

Take your time looking through the data and thoroughly describe the data in the markdown cell below.

Answer: The data sets show the participation rates and score breakdowns for each state for both the ACT and SAT for the years 2017 and 2018. Three of the 4 data sets seem to have 'complete' datasets meaning they provide individual subject breakdowns for score where as act_2018 does not, it simply has participation rates and overall score. All datasets show all states, however the act_2017 adds an additional row for national averages making it's row count 1 greater than the 2017 SAT dataset.

The SAT dataset shows large discrepencies between participation rates of states. They range from 2% to 100% which makes me wonder how 100% participation rate is possible. Some descriptive statistics for the dataset are that the mean score is 1126.09 for the overall test with Evidence-Based Reading and Writing having a higher mean (569.11) than Math (547.62).

For the ACT I noticed that the average Math score is higher than the English score but lower than the Reading score, which might tell us something that the SAT couldn’t where reading and writing are lumped together.

I also noticed that SAT participation rates seem to have a larger deviation in terms of participation rates. I noticed many '100%' participation rates and many <5% participation rates. The ACT seems to have more uniform participation rates with not as much variation.

4a. Does the data look complete?

Answer: All the data appears to have 'complete' data in that all 52 states are present with participation rates and scores for both the individual section of the test and the overall test. There is however information between sets that are not shared such as national averages. The data for average composite scores for the ACT is missing because it is an object and not a float therefore Python cannot apply the .describe() function on that.

4b. Are there any obvious issues with the observations?

What is the minimum *possible* value for each test/subtest? What is the maximum *possible* value?

Consider comparing any questionable values to the sources of your data:

- SAT (https://blog.collegevine.com/here-are-the-average-sat-scores-by-state/)
- ACT (https://blog.prepscholar.com/act-scores-by-state-averages-highs-and-lows)

Answer: There appears to be 100% participation rates which is pretty unusual and unlikely. There are also extremely low participation rates like 2%. Also for the 2017 SAT Maryland has a Math score of 52 which is imposible since the minimum is 200. We can find this by running the following to see the lowest values and the reverse to see if values exceed the expected limit.

```
sat17.sort_values(by = 'math', ascending = True).head()
```

SAT Overall Score: Max: 1600 Min: 400 SAT Subtest Scores: Max: 800 Min: 400

ACT Overall Score: Max: 36 Min: 1 ACT Subtest Scores: Max: 36 Min: 1

4c. Fix any errors you identified

The data is available so there's no need to guess or calculate anything. If you didn't find any errors, continue to the next step.

The Maryland Math score needs to be changed to 524 from 52. The composite score for ACT series should be converted to float to run analysis on it. We should also include a national average series for the 2017 SAT to be able to compare it to the ACT

In [408]:
```
sat17.loc[20, 'Math']=524
```

5. What are your data types?

Display the data types of each feature.

In [409]:
```
type(act17)
act17.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 52 entries, 0 to 51
Data columns (total 7 columns):
State            52 non-null object
Participation    52 non-null object
English          52 non-null float64
Math             52 non-null float64
Reading          52 non-null float64
Science          52 non-null float64
Composite        52 non-null object
dtypes: float64(4), object(3)
memory usage: 3.0+ KB
```

In [410]:
```
type(sat17)
sat17.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51 entries, 0 to 50
Data columns (total 5 columns):
State                                 51 non-null object
Participation                         51 non-null object
Evidence-Based Reading and Writing    51 non-null int64
Math                                  51 non-null int64
Total                                 51 non-null int64
dtypes: int64(3), object(2)
memory usage: 2.1+ KB
```

What did you learn?

- Do any of them seem odd?
- Which ones are not as they should be?

Answer: Both datasets are dataframes. Most of the individual categories are the expected data type for each. The SAT scores are integers since their scores are whole numbers and the ACT contains floats as their scores can be decimals. The one that is not expected was the composite score for the ACT and the participiation for the ACT, I expected those to be floats.

6. Fix Incorrect Data Types

Based on what you discovered above, use appropriate methods to re-type incorrectly typed data.

- Define a function that will allow you to convert participation rates to an appropriate numeric type. Use `map` or `apply` to change these columns in each dataframe.

In [411]:
```python
def remove(dataframe, column):
    dataframe[column] = dataframe[column].map(lambda x: x.rstrip('%abcdefgh
```

Note: I added the alphabet above because when I tried to convert one of the columns with scores I was told that there was a letter that was preventing it from being converted so I decided to include this string of letters in the function.

In [412]:
```python
remove(act17, 'Participation')
```

In [413]:
```python
remove(act17, 'Composite')
```

In [414]:
```python
remove(sat17, 'Participation')
```

In [415]:
```python
def convert(dataframe, column):
    dataframe[column] = dataframe[column].astype(float)
```

In [416]:
```python
convert(act17, 'Participation')
```

In [417]:
```python
convert(act17, 'Composite')
```

In [418]:
```python
convert(sat17, 'Participation')
```

- Fix any individual values preventing other columns from being the appropriate type.

- Finish your data modifications by making sure the columns are now typed appropriately.

In [419]:
```python
sat17.head()
```

Out[419]:

	State	Participation	Evidence-Based Reading and Writing	Math	Total
0	Alabama	5.0	593	572	1165
1	Alaska	38.0	547	533	1080
2	Arizona	30.0	563	553	1116
3	Arkansas	3.0	614	594	1208
4	California	53.0	531	524	1055

In [420]:
```
act17.head()
```

Out[420]:

	State	Participation	English	Math	Reading	Science	Composite
0	National	60.0	20.3	20.7	21.4	21.0	21.0
1	Alabama	100.0	18.9	18.4	19.7	19.4	19.2
2	Alaska	65.0	18.7	19.8	20.4	19.9	19.8
3	Arizona	62.0	18.6	19.8	20.1	19.8	19.7
4	Arkansas	100.0	18.9	19.0	19.7	19.5	19.4

- Display the data types again to confirm they are correct.

In [421]:
```
sat17.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51 entries, 0 to 50
Data columns (total 5 columns):
State                                 51 non-null object
Participation                         51 non-null float64
Evidence-Based Reading and Writing    51 non-null int64
Math                                  51 non-null int64
Total                                 51 non-null int64
dtypes: float64(1), int64(3), object(1)
memory usage: 2.1+ KB
```

In [422]:
```
act17.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 52 entries, 0 to 51
Data columns (total 7 columns):
State            52 non-null object
Participation    52 non-null float64
English          52 non-null float64
Math             52 non-null float64
Reading          52 non-null float64
Science          52 non-null float64
Composite        52 non-null float64
dtypes: float64(6), object(1)
memory usage: 3.0+ KB
```

7. Rename Columns

Change the names of the columns to more expressive names so that you can tell the difference the SAT columns and the ACT columns. Your solution should map all column names being changed at once (no repeated singular name-changes). **We will be combining these data with some of the data from 2018, and so you should name columns in an appropriate way**.

Guidelines:

- Column names should be all lowercase (you will thank yourself when you start pushing data to SQL later in the course)
- Column names should not contain spaces (underscores will suffice--this allows for using the `df.column_name` method to access columns in addition to `df['column_name']`.
- Column names should be unique and informative (the only feature that we actually share between dataframes is the state).

In [423]:
```
sat17.rename(columns={'State': 'state', 'Participation': 'part_sat17',
                      'Evidence-Based Reading and Writing': 'readwrite_sat1
                         , inplace=True)
```

In [424]:
```
act17.rename(columns={'State': 'state', 'Participation': 'part_act17',
                      'English': 'eng_act17', 'Math': 'math_act17', 'Readin
                      'Composite': 'total_act17'}
                         , inplace=True)
```

In [425]:
```
sat17.head()
```

Out[425]:

	state	part_sat17	readwrite_sat17	math_sat17	total_sat17
0	Alabama	5.0	593	572	1165
1	Alaska	38.0	547	533	1080
2	Arizona	30.0	563	553	1116
3	Arkansas	3.0	614	594	1208
4	California	53.0	531	524	1055

In [426]:
```
act17.head()
```

Out[426]:

	state	part_act17	eng_act17	math_act17	read_act17	sci_act17	total_act17
0	National	60.0	20.3	20.7	21.4	21.0	21.0
1	Alabama	100.0	18.9	18.4	19.7	19.4	19.2
2	Alaska	65.0	18.7	19.8	20.4	19.9	19.8
3	Arizona	62.0	18.6	19.8	20.1	19.8	19.7
4	Arkansas	100.0	18.9	19.0	19.7	19.5	19.4

8. Create a data dictionary

Now that we've fixed our data, and given it appropriate names, let's create a data dictionary (http://library.ucmerced.edu/node/10249).

A data dictionary provides a quick overview of features/variables/columns, alongside data types and descriptions. The more descriptive you can be, the more useful this document is.

Example of a Fictional Data Dictionary Entry:

Feature	Type	Dataset	Description
county_pop	*integer*	2010 census	The population of the county (units in thousands, where 2.5 represents 2500 people).
per_poverty	*float*	2010 census	The percent of the county over the age of 18 living below the 200% of official US poverty rate (units percent to two decimal places 98.10 means 98.1%)

Here's a quick link to a short guide for formatting markdown in Jupyter notebooks (https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Working%20With%20Markdown%20Cells.h

Provided is the skeleton for formatting a markdown table, with columns headers that will help you create a data dictionary to quickly summarize your data, as well as some examples. **This would be a great thing to copy and paste into your custom README for this project.**

Feature	Type	Dataset	Description
column name	int/float/object	ACT/SAT	This is an example

Feature	Type	Dataset	Description
state	object	ACT/SAT	Lists the state for which the following statistics are corresponding to.
part_sat/act17	float	ACT/SAT	Lists the participation percentage for eligible students in that state.
math_act17	float	ACT	Lists the corresponding average ACT math score for that state. Scores range from 1.0-36.0
math_sat17	int	SAT	Lists the corresponding average SAT math score for that state. Scores range from 200-800
readwrite_sat17	int	SAT	Lists the corresponding average SAT Evidenced Based Reading and Writing score for that state. Scores range from 200-800.
read_act17	float	ACT	Lists the corresponding average ACT Reading score for that state. Scores range from 1.0-36.0
eng_act17	float	ACT	Lists the corresponding average ACT English score for that state. Scores range from 1.0-36.0
sci_act17	float	ACT	Lists the corresponding average ACT Science score for that state. Scores range from 1.0-36.0
total_sat/act17	int/float	ACT/SAT	Lists the corresponding average total scores for both SAT and ACT. SAT scores are integers that range from 400-1600 and ACT scores are floats that range from 1-36.

9. Drop unnecessary rows

One of our dataframes contains an extra row. Identify and remove this from the dataframe.

```
In [427]: act17.drop(act17.index[[0]], inplace=True)
```

In [428]:
```
act17.head()
```

Out[428]:

	state	part_act17	eng_act17	math_act17	read_act17	sci_act17	total_act17
1	Alabama	100.0	18.9	18.4	19.7	19.4	19.2
2	Alaska	65.0	18.7	19.8	20.4	19.9	19.8
3	Arizona	62.0	18.6	19.8	20.1	19.8	19.7
4	Arkansas	100.0	18.9	19.0	19.7	19.5	19.4
5	California	31.0	22.5	22.7	23.1	22.2	22.8

In [429]:
```
act17.reset_index(drop=True)
```

Out[429]:

	state	part_act17	eng_act17	math_act17	read_act17	sci_act17	total_act17
0	Alabama	100.0	18.9	18.4	19.7	19.4	19.2
1	Alaska	65.0	18.7	19.8	20.4	19.9	19.8
2	Arizona	62.0	18.6	19.8	20.1	19.8	19.7
3	Arkansas	100.0	18.9	19.0	19.7	19.5	19.4
4	California	31.0	22.5	22.7	23.1	22.2	22.8
5	Colorado	100.0	20.1	20.3	21.2	20.9	20.8
6	Connecticut	31.0	25.5	24.6	25.6	24.6	25.2
7	Delaware	18.0	24.1	23.4	24.8	23.6	24.1
8	District of Columbia	32.0	24.4	23.5	24.9	23.5	24.2
9	Florida	73.0	19.0	19.4	21.0	19.4	19.8
10	Georgia	55.0	21.0	20.9	22.0	21.3	21.4
11	Hawaii	90.0	17.8	19.2	19.2	19.3	19.0
12	Idaho	38.0	21.9	21.8	23.0	22.1	22.3
13	Illinois	93.0	21.0	21.2	21.6	21.3	21.4
14	Indiana	35.0	22.0	22.4	23.2	22.3	22.6
15	Iowa	67.0	21.2	21.3	22.6	22.1	21.9
16	Kansas	73.0	21.1	21.3	22.3	21.7	21.7
17	Kentucky	100.0	19.6	19.4	20.5	20.1	20.0
18	Louisiana	100.0	19.4	18.8	19.8	19.6	19.5
19	Maine	8.0	24.2	24.0	24.8	23.7	24.3
20	Maryland	28.0	23.3	23.1	24.2	2.3	23.6
21	Massachusetts	29.0	25.4	25.3	25.9	24.7	25.4
22	Michigan	29.0	24.1	23.7	24.5	23.8	24.1
23	Minnesota	100.0	20.4	21.5	21.8	21.6	21.5
24	Mississippi	100.0	18.2	18.1	18.8	18.8	18.6
25	Missouri	100.0	19.8	19.9	20.8	20.5	20.4
26	Montana	100.0	19.0	20.2	21.0	20.5	20.3
27	Nebraska	84.0	20.9	20.9	21.9	21.5	21.4
28	Nevada	100.0	16.3	18.0	18.1	18.2	17.8
29	New Hampshire	18.0	25.4	25.1	26.0	24.9	25.5
30	New Jersey	34.0	23.8	23.8	24.1	23.2	23.9
31	New Mexico	66.0	18.6	19.4	20.4	20.0	19.7
32	New York	31.0	23.8	24.0	24.6	23.9	24.2

	state	part_act17	eng_act17	math_act17	read_act17	sci_act17	total_act17
33	North Carolina	100.0	17.8	19.3	19.6	19.3	19.1
34	North Dakota	98.0	19.0	20.4	20.5	20.6	20.3
35	Ohio	75.0	21.2	21.6	22.5	22.0	22.0
36	Oklahoma	100.0	18.5	18.8	20.1	19.6	19.4
37	Oregon	40.0	21.2	21.5	22.4	21.7	21.8
38	Pennsylvania	23.0	23.4	23.4	24.2	23.3	23.7
39	Rhode Island	21.0	24.0	23.3	24.7	23.4	24.0
40	South Carolina	100.0	17.5	18.6	19.1	18.9	18.7
41	South Dakota	80.0	20.7	21.5	22.3	22.0	21.8
42	Tennessee	100.0	19.5	19.2	20.1	19.9	19.8
43	Texas	45.0	19.5	20.7	21.1	20.9	20.7
44	Utah	100.0	19.5	19.9	20.8	20.6	20.3
45	Vermont	29.0	23.3	23.1	24.4	23.2	23.6
46	Virginia	29.0	23.5	23.3	24.6	23.5	23.8
47	Washington	29.0	20.9	21.9	22.1	22.0	21.9
48	West Virginia	69.0	20.0	19.4	21.2	20.5	20.4
49	Wisconsin	100.0	19.7	20.4	20.6	20.9	20.5
50	Wyoming	100.0	19.4	19.8	20.8	20.6	20.2

10. Merge Dataframes

Join the 2017 ACT and SAT dataframes using the state in each dataframe as the key. Assign this to a new variable.

In [430]:
```
combined_2017 = pd.merge(left = act17,
          right = sat17,
          on = "state")
```

In [431]:
```
combined_2017.head()
```

Out[431]:

	state	part_act17	eng_act17	math_act17	read_act17	sci_act17	total_act17	part_sat17	rea
0	Alabama	100.0	18.9	18.4	19.7	19.4	19.2	5.0	
1	Alaska	65.0	18.7	19.8	20.4	19.9	19.8	38.0	
2	Arizona	62.0	18.6	19.8	20.1	19.8	19.7	30.0	
3	Arkansas	100.0	18.9	19.0	19.7	19.5	19.4	3.0	
4	California	31.0	22.5	22.7	23.1	22.2	22.8	53.0	

11. Save your cleaned, merged dataframe

Use a relative path to save out your data as `combined_2017.csv` .

In [432]:
```
combined_2017.to_csv(r'/Users/rdk/Desktop/GA/GA_DSI_10/Projects/project_1/c
```

2018 Data Import and Cleaning

Links to the 2018 ACT and SAT data are provided in the README. These data live in PDFs, and so you'll get to enjoy practicing some ***manual*** data collection. Save these data as a CSV in your `data` directory, and import, explore, and clean these data in the same way you did above. **Make sure you comment on your steps so it is clear *why* you are doing each process.**

In [433]:
```
act18 = pd.read_csv('/Users/rdk/Desktop/GA/GA_DSI_10/Projects/project_1/dat
sat18 = pd.read_csv('/Users/rdk/Desktop/GA/GA_DSI_10/Projects/project_1/dat
```

In [434]:
```
sat18.head()
```

Out[434]:

	State	Participation	Evidence-Based Reading and Writing	Math	Total
0	Alabama	6%	595	571	1166
1	Alaska	43%	562	544	1106
2	Arizona	29%	577	572	1149
3	Arkansas	5%	592	576	1169
4	California	60%	540	536	1076

Note: To save space on the notebook I printed the head, however for the analysis below I did look at the whole dataset.

In [435]:
```
sat18.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51 entries, 0 to 50
Data columns (total 5 columns):
State                                 51 non-null object
Participation                         51 non-null object
Evidence-Based Reading and Writing    51 non-null int64
Math                                  51 non-null int64
Total                                 51 non-null int64
dtypes: int64(3), object(2)
memory usage: 2.1+ KB
```

From taking a look at the SAT dataset it is clear that I will have to again remove the '%' symbol from the participation column in order to convert it to a float. Everything else seems normal, no out of range numbers, no null values, and all the other data types are as expected.

In [436]:
```
act18.head()
```

Out[436]:

	State	Participation	Composite
0	Alabama	100%	19.1
1	Alaska	33%	20.8
2	Arizona	66%	19.2
3	Arkansas	100%	19.4
4	California	27%	22.7

In [437]:
```
act18.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 52 entries, 0 to 51
Data columns (total 3 columns):
State            52 non-null object
Participation    52 non-null object
Composite        52 non-null float64
dtypes: float64(1), object(2)
memory usage: 1.3+ KB
```

Same as before, need to remove the '%' symbol from the participation column to convert to float. There doesn't appear to be a 'National Average' column as in the ACT 2017 dataset, however there is an extra row for the ACT dataset again. Maine's stats have been printed twice so one of them will need to be dropped. I also noticed that the dataset is not as detailed as the 2017 dataset as individual subtest scores are not listed, only composites.

Step 1: Convert columns to appropriate datatype as before. The only column here that needs to be converted for both tests is participation. We can use the same function as we did for the 2017 tests and remove the % symbol and then convert to float. That could was:

```
def remove(dataframe, column):
    dataframe[column] = dataframe[column].map(lambda x: x.rstrip('%
abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ')
```

In [438]:
```
remove(sat18, 'Participation')
```

In [439]:
```
remove(act18, 'Participation')
```

In [440]:
```
sat18.head()
```

Out[440]:

	State	Participation	Evidence-Based Reading and Writing	Math	Total
0	Alabama	6	595	571	1166
1	Alaska	43	562	544	1106
2	Arizona	29	577	572	1149
3	Arkansas	5	592	576	1169
4	California	60	540	536	1076

In [441]:
```
act18.head()
```

Out[441]:

	State	Participation	Composite
0	Alabama	100	19.1
1	Alaska	33	20.8
2	Arizona	66	19.2
3	Arkansas	100	19.4
4	California	27	22.7

We ran the function and are checking to make sure the column does not have percentage symbols by running df.head( ).

Step 2: Now to be able to manipulate and run numerical functions on the participation rates we want to convert the column into float datatypes. Although the percentages listed are whole numbers, percentages are floats and it will be useful for any sort of manipulation in the future. We will use the same convert function as before, shown below.

```
def convert(dataframe, column):
    dataframe[column] = dataframe[column].astype(float)
```

In [442]:
```
convert(sat18, 'Participation')
```

In [443]:
```
convert(act18, 'Participation')
```

In [444]:
```
sat18.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51 entries, 0 to 50
Data columns (total 5 columns):
State                                 51 non-null object
Participation                         51 non-null float64
Evidence-Based Reading and Writing    51 non-null int64
Math                                  51 non-null int64
Total                                 51 non-null int64
dtypes: float64(1), int64(3), object(1)
memory usage: 2.1+ KB
```

In [445]:
```
act18.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 52 entries, 0 to 51
Data columns (total 3 columns):
State            52 non-null object
Participation    52 non-null float64
Composite        52 non-null float64
dtypes: float64(2), object(1)
memory usage: 1.3+ KB
```

Once again we are just checking to make sure the function did as expected by using df.info( ).

Step 3: As before we want to rename columns for each dataset so that we can differentiate when we merge them. We can pass a similar function as before as shown below.

```
sat17.rename(columns={'State': 'state', 'Participation': 'part_sat'
,
                      'Evidence-Based Reading and Writing': 'readwr
ite_sat', 'Math': 'math_sat', 'Total':'total_sat'}
                         , inplace=True)
```

In [446]:
```
sat18.rename(columns={'State': 'state', 'Participation': 'part_sat18',
                      'Evidence-Based Reading and Writing': 'readwrite_sat1
                         , inplace=True)
```

In [447]:
```
act18.rename(columns={'State': 'state', 'Participation': 'part_act18', 'Com
                         , inplace=True)
```

In [448]:
```
sat18.head()
```

Out[448]:

	state	part_sat18	readwrite_sat18	math_sat18	total_sat18
0	Alabama	6.0	595	571	1166
1	Alaska	43.0	562	544	1106
2	Arizona	29.0	577	572	1149
3	Arkansas	5.0	592	576	1169
4	California	60.0	540	536	1076

In [449]:
```
act18.head()
```

Out[449]:

	state	part_act18	total_act18
0	Alabama	100.0	19.1
1	Alaska	33.0	20.8
2	Arizona	66.0	19.2
3	Arkansas	100.0	19.4
4	California	27.0	22.7

Once again checking to make sure the columns were renamed as expected

Step 4: Create a data dictionary. This is important even if the previous year has a data dictionary. As we can see some columns have been taken out, and some years columns made be added so it's good to always update the data dictionary.

Feature	Type	Dataset	Description
state	object	ACT/SAT	Lists the state for which the following statistics are corresponding to.
part_sat/act18	float	ACT/SAT	Lists the participation percentage for eligible students in that state.
math_sat18	int	SAT	Lists the corresponding average SAT math score for that state. Scores range from 200-800
readwrite_sat18	int	SAT	Lists the corresponding average SAT Evidenced Based Reading and Writing score for that state. Scores range from 200-800.
total_sat/act18	int/float	ACT/SAT	Lists the corresponding average total scores for both SAT and ACT. SAT scores are integers that range from 400-1600 and ACT scores are floats that range from 1-36.

Step 5: Before joining our dataframes on 'state I need to remove the second 'Maine' row from the ACT 2018 dataset mentioned before. This was done before using .drop and inplace = True for the national average row of the ACT 2017 data set.

```
act17.drop(act17.index[[0]], inplace=True)
```

In [450]:
```
act18.drop(act18.index[[20]], inplace=True)
```

Need to reset the index after removing the duplicate Maine row.

In [451]:
```
act18.reset_index(drop=True)
```

Out[451]:

	state	part_act18	total_act18
0	Alabama	100.0	19.1
1	Alaska	33.0	20.8
2	Arizona	66.0	19.2
3	Arkansas	100.0	19.4
4	California	27.0	22.7
5	Colorado	30.0	23.9
6	Connecticut	26.0	25.6
7	Delaware	17.0	23.8
8	District of columbia	32.0	23.6
9	Florida	66.0	19.9
10	Georgia	53.0	21.4
11	Hawaii	89.0	18.9
12	Idaho	36.0	22.3
13	Illinois	43.0	23.9
14	Indiana	32.0	22.5
15	Iowa	68.0	21.8
16	Kansas	71.0	21.6
17	Kentucky	100.0	20.2
18	Louisiana	100.0	19.2
19	Maine	7.0	24.0
20	Maryland	31.0	22.5
21	Massachusetts	25.0	25.5
22	Michigan	22.0	24.2
23	Minnesota	99.0	21.3
24	Mississippi	100.0	18.6
25	Missouri	100.0	20.0
26	Montana	100.0	20.0
27	Nebraska	100.0	20.1
28	Nevada	100.0	17.7
29	New Hampshire	16.0	25.1
30	New Jersey	31.0	23.7
31	New Mexico	67.0	19.4
32	New York	27.0	24.5

	state	part_act18	total_act18
33	North Carolina	100.0	19.1
34	North Dakota	98.0	20.3
35	Ohio	100.0	20.3
36	Oklahoma	100.0	19.3
37	Oregon	42.0	21.3
38	Pennsylvania	20.0	23.5
39	Rhode Island	15.0	24.2
40	South Carolina	100.0	18.3
41	South Dakota	77.0	21.9
42	Tennessee	100.0	19.6
43	Texas	45.0	20.7
44	Utah	100.0	20.4
45	Vermont	24.0	24.1
46	Virginia	24.0	23.9
47	Washington	24.0	22.2
48	West Virginia	65.0	20.3
49	Wisconsin	100.0	20.5
50	Wyoming	100.0	20.0

In [452]:
```
act18.loc[8,'state'] = 'District of Columbia'
```

In [453]:
```
act18
```

Out[453]:

	state	part_act18	total_act18
0	Alabama	100.0	19.1
1	Alaska	33.0	20.8
2	Arizona	66.0	19.2
3	Arkansas	100.0	19.4
4	California	27.0	22.7
5	Colorado	30.0	23.9
6	Connecticut	26.0	25.6
7	Delaware	17.0	23.8
8	District of Columbia	32.0	23.6
9	Florida	66.0	19.9
10	Georgia	53.0	21.4
11	Hawaii	89.0	18.9
12	Idaho	36.0	22.3
13	Illinois	43.0	23.9
14	Indiana	32.0	22.5
15	Iowa	68.0	21.8
16	Kansas	71.0	21.6
17	Kentucky	100.0	20.2
18	Louisiana	100.0	19.2
19	Maine	7.0	24.0
21	Maryland	31.0	22.5
22	Massachusetts	25.0	25.5
23	Michigan	22.0	24.2
24	Minnesota	99.0	21.3
25	Mississippi	100.0	18.6
26	Missouri	100.0	20.0
27	Montana	100.0	20.0
28	Nebraska	100.0	20.1
29	Nevada	100.0	17.7
30	New Hampshire	16.0	25.1
31	New Jersey	31.0	23.7
32	New Mexico	67.0	19.4
33	New York	27.0	24.5

	state	part_act18	total_act18
34	North Carolina	100.0	19.1
35	North Dakota	98.0	20.3
36	Ohio	100.0	20.3
37	Oklahoma	100.0	19.3
38	Oregon	42.0	21.3
39	Pennsylvania	20.0	23.5
40	Rhode Island	15.0	24.2
41	South Carolina	100.0	18.3
42	South Dakota	77.0	21.9
43	Tennessee	100.0	19.6
44	Texas	45.0	20.7
45	Utah	100.0	20.4
46	Vermont	24.0	24.1
47	Virginia	24.0	23.9
48	Washington	24.0	22.2
49	West Virginia	65.0	20.3
50	Wisconsin	100.0	20.5
51	Wyoming	100.0	20.0

Combine your 2017 and 2018 data into a single dataframe

Joining on state names should work, assuming you formatted all your state names identically. Make sure none of your columns (other than state) have identical names. Do yourself a favor and decide if you're encoding participation rates as floats or integers and standardize this across your datasets.

Save the contents of this merged dataframe as `final.csv`.

Use this combined dataframe for the remainder of the project.

In [454]:
```
combined_2018 = pd.merge(left = act18,
         right = sat18,
         on = "state")
```

In [455]:
```
final = pd.merge(left = combined_2017,
                 right = combined_2018,
                 on = 'state')
final
```

Out[455]:

	state	part_act17	eng_act17	math_act17	read_act17	sci_act17	total_act17	part_sat1
0	Alabama	100.0	18.9	18.4	19.7	19.4	19.2	5.(
1	Alaska	65.0	18.7	19.8	20.4	19.9	19.8	38.(
2	Arizona	62.0	18.6	19.8	20.1	19.8	19.7	30.(
3	Arkansas	100.0	18.9	19.0	19.7	19.5	19.4	3.(
4	California	31.0	22.5	22.7	23.1	22.2	22.8	53.(
5	Colorado	100.0	20.1	20.3	21.2	20.9	20.8	11.(
6	Connecticut	31.0	25.5	24.6	25.6	24.6	25.2	100.(
7	Delaware	18.0	24.1	23.4	24.8	23.6	24.1	100.(
8	District of Columbia	32.0	24.4	23.5	24.9	23.5	24.2	100.(
9	Florida	73.0	19.0	19.4	21.0	19.4	19.8	83.(
10	Georgia	55.0	21.0	20.9	22.0	21.3	21.4	61.(
11	Hawaii	90.0	17.8	19.2	19.2	19.3	19.0	55.(
12	Idaho	38.0	21.9	21.8	23.0	22.1	22.3	93.(
13	Illinois	93.0	21.0	21.2	21.6	21.3	21.4	9.(
14	Indiana	35.0	22.0	22.4	23.2	22.3	22.6	63.(
15	Iowa	67.0	21.2	21.3	22.6	22.1	21.9	2.(
16	Kansas	73.0	21.1	21.3	22.3	21.7	21.7	4.(
17	Kentucky	100.0	19.6	19.4	20.5	20.1	20.0	4.(
18	Louisiana	100.0	19.4	18.8	19.8	19.6	19.5	4.(
19	Maine	8.0	24.2	24.0	24.8	23.7	24.3	95.(
20	Maryland	28.0	23.3	23.1	24.2	2.3	23.6	69.(
21	Massachusetts	29.0	25.4	25.3	25.9	24.7	25.4	76.(
22	Michigan	29.0	24.1	23.7	24.5	23.8	24.1	100.(
23	Minnesota	100.0	20.4	21.5	21.8	21.6	21.5	3.(
24	Mississippi	100.0	18.2	18.1	18.8	18.8	18.6	2.(
25	Missouri	100.0	19.8	19.9	20.8	20.5	20.4	3.(
26	Montana	100.0	19.0	20.2	21.0	20.5	20.3	10.(
27	Nebraska	84.0	20.9	20.9	21.9	21.5	21.4	3.(
28	Nevada	100.0	16.3	18.0	18.1	18.2	17.8	26.(
29	New Hampshire	18.0	25.4	25.1	26.0	24.9	25.5	96.(

	state	part_act17	eng_act17	math_act17	read_act17	sci_act17	total_act17	part_sat1
30	New Jersey	34.0	23.8	23.8	24.1	23.2	23.9	70.
31	New Mexico	66.0	18.6	19.4	20.4	20.0	19.7	11.
32	New York	31.0	23.8	24.0	24.6	23.9	24.2	67.
33	North Carolina	100.0	17.8	19.3	19.6	19.3	19.1	49.
34	North Dakota	98.0	19.0	20.4	20.5	20.6	20.3	2.
35	Ohio	75.0	21.2	21.6	22.5	22.0	22.0	12.
36	Oklahoma	100.0	18.5	18.8	20.1	19.6	19.4	7.
37	Oregon	40.0	21.2	21.5	22.4	21.7	21.8	43.
38	Pennsylvania	23.0	23.4	23.4	24.2	23.3	23.7	65.
39	Rhode Island	21.0	24.0	23.3	24.7	23.4	24.0	71.
40	South Carolina	100.0	17.5	18.6	19.1	18.9	18.7	50.
41	South Dakota	80.0	20.7	21.5	22.3	22.0	21.8	3.
42	Tennessee	100.0	19.5	19.2	20.1	19.9	19.8	5.
43	Texas	45.0	19.5	20.7	21.1	20.9	20.7	62.
44	Utah	100.0	19.5	19.9	20.8	20.6	20.3	3.
45	Vermont	29.0	23.3	23.1	24.4	23.2	23.6	60.
46	Virginia	29.0	23.5	23.3	24.6	23.5	23.8	65.
47	Washington	29.0	20.9	21.9	22.1	22.0	21.9	64.
48	West Virginia	69.0	20.0	19.4	21.2	20.5	20.4	14.
49	Wisconsin	100.0	19.7	20.4	20.6	20.9	20.5	3.
50	Wyoming	100.0	19.4	19.8	20.8	20.6	20.2	3.

```
In [456]: final.to_csv(r'/Users/rdk/Desktop/GA/GA_DSI_10/Projects/project_1/data/fina
```

Exploratory Data Analysis

Summary Statistics

Transpose the output of pandas `describe` method to create a quick overview of each numeric feature.

In [457]:
```
final.describe().T
```

Out[457]:
```
                   count         mean        std    min      25%     50%      75%     max
     part_act17     51.0    65.254902  32.140842    8.0    31.00    69.0   100.00   100.0
      eng_act17     51.0    20.931373   2.353677   16.3    19.00    20.7    23.30    25.5
     math_act17     51.0    21.182353   1.981989   18.0    19.40    20.9    23.10    25.3
     read_act17     51.0    22.013725   2.067271   18.1    20.45    21.8    24.15    26.0
      sci_act17     51.0    21.041176   3.182463    2.3    19.90    21.3    22.75    24.9
    total_act17     51.0    21.519608   2.020695   17.8    19.80    21.4    23.60    25.5
     part_sat17     51.0    39.803922  35.276632    2.0     4.00    38.0    66.00   100.0
readwrite_sat17     51.0   569.117647  45.666901  482.0   533.50   559.0   613.00   644.0
     math_sat17     51.0   556.882353  47.121395  468.0   523.50   548.0   599.00   651.0
    total_sat17     51.0  1126.098039  92.494812  950.0  1055.50  1107.0  1212.00  1295.0
     part_act18     51.0    61.725490  34.037085    7.0    28.50    66.0   100.00   100.0
    total_act18     51.0    21.496078   2.111583   17.7    19.95    21.3    23.65    25.6
     part_sat18     51.0    45.745098  37.314256    2.0     4.50    52.0    77.50   100.0
readwrite_sat18     51.0   563.686275  47.502627  480.0   534.50   552.0   610.50   643.0
     math_sat18     51.0   556.235294  47.772623  480.0   522.50   544.0   593.50   655.0
    total_sat18     51.0  1120.019608  94.155083  977.0  1057.50  1098.0  1204.00  1298.0
```

Manually calculate standard deviation

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2}$$

- Write a function to calculate standard deviation using the formula above

In [ ]:
```

```

- Use a **dictionary comprehension** to apply your standard deviation function to each numeric column in the dataframe. **No loops**
- Assign the output to variable `sd` as a dictionary where:
  - Each column name is now a key
  - That standard deviation of the column is the value

Example Output : `{'ACT_Math': 120, 'ACT_Reading': 120, ...}`

In [458]:
```
def get_stddev(column):
    for x in
```

```
  File "<ipython-input-458-6d06193f1128>", line 2
    for x in
            ^
SyntaxError: invalid syntax
```

Do your manually calculated standard deviations match up with the output from pandas `describe` ? What about numpy's `std` method?

Answer

Investigate trends in the data

Using sorting and/or masking (along with the `.head` method to not print our entire dataframe), consider the following questions:

- Which states have the highest and lowest participation rates for the:
  - 2017 SAT?
  - 2018 SAT?
  - 2017 ACT?
  - 2018 ACT?
- Which states have the highest and lowest mean total/composite scores for the:
  - 2017 SAT?
  - 2018 SAT?
  - 2017 ACT?
  - 2018 ACT?
- Do any states with 100% participation on a given test have a rate change year-to-year?
- Do any states show have >50% participation on ***both*** tests either year?

Based on what you've just observed, have you identified any states that you're especially interested in? **Make a note of these and state *why* you think they're interesting.**

You should comment on your findings at each step in a markdown cell below your code block. Make sure you include at least one example of sorting your dataframe by a column, and one example of using boolean filtering (i.e., masking) to select a subset of the dataframe.

SAT 2017 Participation Rates

5 Lowest Participation Rates: North Dakota, Mississippi, Iowa, Missouri, Utah

5 Highest Participation Rates: District of Columbia, Michigan, Conneticut, Delaware, New Hampshire

Notes: I think it is interesting how many of the lowest particpating states are in the Midwest and South and most of the highest participation states are in the North East.

In [ ]:
```
final.sort_values(by = 'part_sat17', ascending = True).head()
```

In [ ]:
```
final.sort_values(by = 'part_sat17', ascending = False).head()
```

SAT 2018 Participation Rates

5 Highest Participation Rates: Colorado, Conneticut, Delaware, Idaho, Michigan

5 Lowest Participation Rates: North Dakota, Tie: Iowa, Mississippi, Nebraska, South Dakota, Wisconsin, Wyoming.

Notes: Again the lowest participating states are in the Midwest and South, would be interesting to see as to why this is. Maybe the ACT is more common?

In [ ]:
```
mask1 = final['part_sat18'] > 96
sat18[mask1]
```

In [ ]:
```
mask2= final['part_sat18'] <5
sat18[mask2]
```

ACT 2017 Participation Rates

5 Highest Participation Rates: Alabama, Kentucky, Wisconsin, Utah, Tennessee

5 Lowest Participation Rates: Maine, New Hampshire, Delaware, Rhode Island, Pennsylvania

Notes: Right away I notice that the southern states have the highest participation rates while the northeastern states have the lowest participation rates for ACT, the inverse of the pattern observed for the SAT.

In [ ]:
```
final.sort_values(by = 'part_act17', ascending = False).head()
```

In [ ]:
```
final.sort_values(by = 'part_act17', ascending = False).head()
```

ACT 2018 Participation Rates

5 Highest Participation Rates: Alabama, Kentucky, Wisconsin, Utah, Tennessee.

5 Lowest Participation Rates: Maine, Rhode Island, New Hampshire, Delaware Pennsylvania

Notes: Both lists for each remained the same from 2017. Wondering if there is a policy set in place for these states to promote the SAT over the ACT or vice versa?

In [ ]:
```
final.sort_values(by = 'part_act18', ascending = False).head()
```

In [ ]:
```
final.sort_values(by = 'part_act18', ascending = True).head()
```

2017 and 2018 Composite Test Score Averages

2017 Scores

- ***2017 Highest Scoring States***: Minnesota, Wisconsin, Iowa, Missouri, Kansas
- ***2017 Lowest Scoring States***: District of Columbia, Delaware, Idaho, Michigan, Maine

---

2018

- ***2018 Highest Scoring States***: Minnesota, Wisconsin, North Dakota, Iowa, Kansas
- ***2018 Lowest Scoring states***: District of Columbia, Delaware, West Virginia, Idaho Utah

Notes: I noticed the highest scoring states had the lowest participation. This could be because that those few students who did the exam studied specifically for the SAT and therefore produced a higher average than a general population who did not care so much about which test they took.

```
In [ ]: final.sort_values(by = 'total_sat17', ascending = False).head()
```

```
In [ ]: final.sort_values(by = 'total_sat17', ascending = True).head()
```

```
In [ ]: final.sort_values(by = 'total_sat18', ascending = False).head()
```

```
In [ ]: final.sort_values(by = 'total_sat18', ascending = True).head()
```

2017 and 2018 ACT Composite Test Scores

2017

- ***2017 Highest Scoring States***: New Hampshire, Massachusetts, Conneticut, Maine, District of Columbia
- ***2017 Lowest Scoring States***: Nevada, Mississippi, South Carolina, Hawaii, North Carolina

2018

- ***2018 Highest Scoring States***: Conneticut, Massachusetts, New Hampshire, New York, Michigan
- ***2018 Lowest Scoring States***: Nevada, South Carolina, Mississippi, Hawaii, Alabama

Notes: The Northeastern states seem to do better on the ACT but like the highest scoring SAT states, had the lowest participation. Again, this might be because the students who took the SAT made an active decision to study for that test instead of following the general population of the state and taking the SAT. If they took an interest in which test they took, they might have also studied harder, intentionality.

```
In [ ]: final.sort_values(by = 'total_act17', ascending = False).head()
```

```
In [ ]: final.sort_values(by = 'total_act17', ascending = True).head()
In [ ]: final.sort_values(by = 'total_act18', ascending = False).head()
```

```
In [ ]: final.sort_values(by = 'total_act18', ascending = True).head()
```

SAT 100% Participation Rates

As shown from the two tables, the only state that had a 100% participation rates for the 2017 SAT that changed in 2018 was District of Columbia which had a 100% participation in 2017 and 92% in 2018.

States with 100% SAT Participation Rates for 2017 and 2018

- Conneticut
- Delaware
- Michigan

```
In [ ]: final[(final["part_sat17"] == 100)]
```

ACT 100% Paticipation Rates

Colardo had a significant drop off in participation rates from 100% in 2017 to 30% in 2018, and Minnesota had a slight drop of 1% in 2018 from 100%.

States that retained 100% participation rates from 2017 to 2018.

- Alabama
- Arkansas
- Colorado
- Kentucky
- Lousiana
- Minnesota
- Mississippi
- Missouri
- Montana
- Nevada
- North Carolina
- Oklahoma
- South Carolina
- Tennessee
- Utah
- Wisconsin
- Wyoming

Notes: Notes: It should be noted that the amount of states with 100% participation rates for the ACT test is significantly more than the states with 100% participation rates for the SAT. This leads me to believe certain states might have a mandate for ACT test taking over SAT of some sort.

```
In [ ]: final[(final["part_act17"] == 100)]
```

States with Over 50% Participation for Each Test

For 2017 Florida, George, and Hawaii reported over 50% participation rates for each test which is impossible if the conditions for detemrining participation were the same. In 2018 there is an increase in this phenomenon with again Florida, George and Hawaii reporting over 100% participation between 2 tests but also North Carolina and South Carolina reporting participation rates over 100% between the two tests. A topic of investigation might be how does each testing agency or each state count 'participation,' but also making note that this is possible because the two tests are not mutually exclusive. Half a state may actually take both the SAT and the ACT.

```
In [ ]: final[(final['part_act17'] > 50) & (final['part_sat17'] > 50)]
```

```
In [ ]: final[(final['part_act18'] > 50) & (final['part_sat18'] > 50)]
```

Visualize the data

There's not a magic bullet recommendation for the right number of plots to understand a given dataset, but visualizing your data is ***always*** a good idea. Not only does it allow you to quickly convey your findings (even if you have a non-technical audience), it will often reveal trends in your data that escaped you when you were looking only at numbers.

Some recommendations on plotting:

- Plots have titles
- Plots have axis labels
- Plots have appropriate tick labels
- All text is legible in a plot
- Plots demonstrate meaningful and valid relationships
- Plots are interpreted to aid understanding

There is such a thing as too many plots, and there are a ***lot*** of bad plots. You might make some! (But hopefully not with the guided prompts below).

Use Seaborn's heatmap with pandas `.corr()` to visualize correlations between all numeric features

Heatmaps are generally not appropriate for presentations, and should often be excluded from reports as they can be visually overwhelming. **However**, they can be extremely useful in identify relationships of potential interest (as well as identifying potential collinearity before modeling).

example:

```
sns.heatmap(df.corr())
```

Please take time to format your output, adding a title. Look through some of the additional arguments and options. (Axis labels aren't really necessary, as long as the title is informative).

```
In [ ]: import numpy as np
        import seaborn as sns
        import matplotlib.pyplot as plt
        import matplotlib.cm as cm
        import pandas as pd
        import bokeh as bk
        %matplotlib inline
```

```
In [ ]: plt.figure(figsize = (14,14))
        plt.title('Correlation Between 2017 and 2018 SAT/ACT Participation and Scor
        plt.xlabel('Test Categories for 2017 and 2018', fontsize = 12)
        plt.ylabel('Test Categories for 2017 and 2018', fontsize = 12)
        sns.heatmap(final.corr(),cmap = 'hot', annot = True, annot_kws={"size": 13}
```

Define a custom function to subplot histograms

We have data for two tests for two years. We only have composite (and not subtest scores) for the 2018 ACT. We should write a function that will take the names of 2+ columns and subplot histograms. While you can use pandas plotting or Seaborn here, matplotlib gives you greater control over all aspects of your plots.

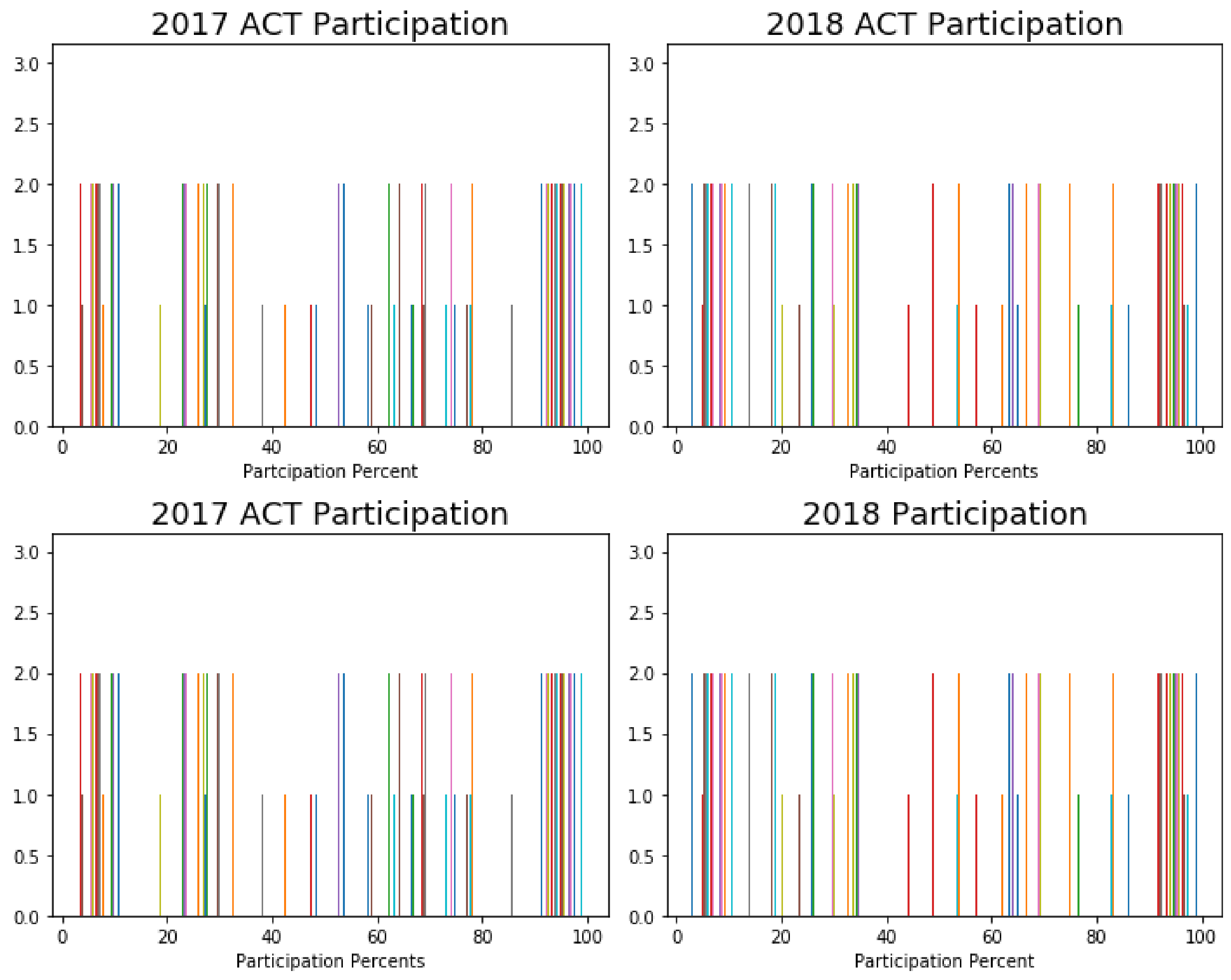
[Helpful Link for Plotting Multiple Figures (https://matplotlib.org/users/pyplot_tutorial.html#working-with-multiple-figures-and-axes)](https://matplotlib.org/users/pyplot_tutorial.html#working-with-multiple-figures-and-axes)

Here's some starter code:

```
In [459]: def subplot_histograms(dataframe, list_of_columns, list_of_titles, list_of_
              nrows = int(np.ceil(len(list_of_columns)/2)) # Makes sure you have enou
              fig, ax = plt.subplots(nrows=nrows, ncols=2) # You'll want to specify y
              ax = ax.ravel() # Ravel turns a matrix into a vector, which is easier t
              for i, column in enumerate(list_of_columns): # Gives us an index value
                  ax[i].hist(dataframe[column]) # feel free to add more settings
                  # Set titles, labels, etc here for each subplot
```

```
In [509]: def subplot_histograms(dataframe, list_of_columns, list_of_titles, list_of_
              nrows = int(np.ceil(len(list_of_columns)/2)) # Makes sure you have enou
              fig, ax = plt.subplots(nrows=nrows, ncols=2, figsize = (10,8)) # You'll
              ax = ax.ravel() # Ravel turns a matrix into a vector, which is easier t
              for i, column in enumerate(list_of_columns): # Gives us an index value
                  ax[i].hist(dataframe[column]) # feel free to add more settings
                  ax[i].set_title(list_of_titles[i], fontsize = 18)
                  ax[i].set_xlabel(list_of_xlabels[i])
              plt.tight_layout()
```

In [508]:
```
subplot_histograms(final, columns = ['part_act17','part_act18','part_sat17'
                         titles = ['2017 ACT Participation', '2018 ACT Particip
                                   '2018 Participation'],
                         xlabels = ['Partcipation Percent', 'Participation Perc
                                    'Participation Percent'])
```



Plot and interpret histograms

For each of the following:

- Participation rates for SAT & ACT
- Math scores for SAT & ACT
- Reading/verbal scores for SAT & ACT

In [ ]:
```
# Code
```

Plot and interpret scatter plots

For each of the following:

- SAT vs. ACT math scores for 2017
- SAT vs. ACT verbal/reading scores for 2017
- SAT vs. ACT total/composite scores for 2017
- Total scores for SAT 2017 vs. 2018
- Composite scores for ACT 2017 vs. 2018

Plot the two variables against each other using matplotlib or Seaborn

Your plots should show:

- Two clearly labeled axes
- A proper title
- Using colors and symbols that are clear and unmistakable

Feel free to write a custom function, and subplot if you'd like. Functions save both time and space.

In [ ]:
```
plt.subplots(figsize = (12,8))
plt.grid(color='grey', linestyle='-', linewidth=0.25, alpha=0.5)
plt.scatter(x = final["math_act17"],y = final["math_sat17"]);
plt.title("2017 Math Score Comparison: SAT vs ACT", fontsize = 24)
plt.ylabel("Average 2017 Math SAT Scores by State", fontsize = 20)
plt.xlabel('Average 2017 Math ACT Scores by State', fontsize = 20)
plt.show()
```

In [ ]:
```
```

In [ ]:
```
plt.subplots(figsize = (12,8))
plt.grid(color='grey', linestyle='-', linewidth=0.25, alpha=0.5)
plt.scatter(x = final["read_act17"],y = final["readwrite_sat17"], color = '
plt.title("2017 Reading/Verbal Score Comparison: SAT vs ACT", fontsize = 24
plt.ylabel("Average 2017 Reading/Writing SAT Scores by State", fontsize = 2
plt.xlabel('Average 2017 Reading ACT Scores by State', fontsize = 20)
plt.show()
```

In [ ]:
```
plt.subplots(figsize = (12,8))
plt.grid(color='grey', linestyle='-', linewidth=0.25, alpha=0.5)
colors = ['#2300A8', '#00A658']
plt.scatter(x = final["total_act17"],y = final["total_sat17"], color = 'm')
plt.title("2017 Total Composite SAT vs ACT", fontsize = 24)
plt.ylabel("Average 2017 SAT Scores by State", fontsize = 20)
plt.xlabel('Average 2017 ACT Scores by State', fontsize = 20)
plt.show()
```

In [ ]:
```
plt.subplots(figsize = (12,8))
plt.grid(color='grey', linestyle='-', linewidth=0.25, alpha=0.5)
colors = ['#2300A8', '#00A658']
plt.scatter(x = final["total_sat17"],y = final["total_sat18"], color = 'y')
plt.title("2017 vs 2018 SAT Score Comparison", fontsize = 24)
plt.ylabel("Average 2018 SAT Scores by State", fontsize = 20)
plt.xlabel('Average 2017 SAT Scores by State', fontsize = 20)
plt.show()
```

In [ ]:
```
plt.subplots(figsize = (12,8))
plt.grid(color='grey', linestyle='-', linewidth=0.25, alpha=0.5)
colors = ['#2300A8', '#00A658']
plt.scatter(x = final["total_act17"],y = final["total_act18"], color='r');
plt.title("2017 vs 2018 ACT Score Comparison", fontsize = 24)
plt.ylabel("Average 2018 ACT Scores by State", fontsize = 20)
plt.xlabel('Average 2017 ACT Scores by State', fontsize = 20)
plt.show()
```

Plot and interpret boxplots

For each numeric variable in the dataframe create a boxplot using Seaborn. Boxplots demonstrate central tendency and spread in variables. In a certain sense, these are somewhat redundant with histograms, but you may be better able to identify clear outliers or differences in IQR, etc.

Multiple values can be plotted to a single boxplot as long as they are of the same relative scale (meaning they have similar min/max values).

Each boxplot should:

- Only include variables of a similar scale
- Have clear labels for each variable
- Have appropriate titles and labels

```
In [ ]: boxplot1 = sns.boxplot(data = final[['total_sat17', 'total_sat18']],
                               saturation = 20, palette="Set2").set(xlabel = 'SAT b
                                                    ylabel = 'Score',
                                                    title = '2017 vs 2018 State Ave
```

```
In [ ]: boxplot2 = sns.boxplot(data = final[['total_act17', 'total_act18']],
                               saturation = 20, palette="Set3").set(xlabel = 'Year'
                                                    ylabel = 'Composite Scores',
                                                    title = '2017 vs 2018 State Ave
```

```
In [ ]: boxplot3 = sns.boxplot(data = final[['part_sat17',  'part_act17', 'part_sat
                                            'part_act18']],
                               saturation = 20, palette ="dark").set(xlabel = 'Year
                                                    ylabel = 'Participation %',
                                                    title = '2017 vs 2018 State Ave
```

```
In [ ]: boxplot4 = sns.boxplot(data = final[['math_sat17', 'math_sat18']],
                               saturation = 10, palette = 'RdBu').set(xlabel = 'Yea
                                                    ylabel = 'Score',
                                                    title = '2017 vs 2018 State Ave
```

```
In [ ]: boxplot5 = sns.boxplot(data = final[['readwrite_sat17', 'readwrite_sat18']]
                               saturation = 20, palette = 'husl').set(xlabel = 'Yea
                                                    ylabel = 'Score',
                                                    title = '2017 vs 2018 State Ave
```

```
In [ ]: boxplot6 = sns.boxplot(data = final[['math_act17', 'sci_act17',
                                              'eng_act17','read_act17']],
                               saturation = 20, palette = 'Set1').set(xlabel = 'Sub
                                                    ylabel = 'Score',
                                                    title = '2017 State Average Ran
```

Feel free to do additional plots below

(do research and choose your own chart types & variables)

Are there any additional trends or relationships you haven't explored? Was there something interesting you saw that you'd like to dive further into? It's likely that there are a few more plots you might want to generate to support your narrative and recommendations that you are building toward. **As always, make sure you're interpreting your plots as you go.**

```
In [ ]:
```

```
In [ ]:
```

Outside Research

(Optional): Using Tableau, create a choropleth map for each variable using a map of the US.

Save this plot as an image file in an images directory, provide a relative path, and insert the image into notebook in markdown.

Based upon your observations, choose **three** states that demonstrate interesting trends in their SAT and/or ACT participation rates. Spend some time doing outside research on state policies that might influence these rates, and summarize your findings below. **Feel free to go back and create new plots that highlight these states of interest.** If you bring in any outside tables or charts, make sure you are explicit about having borrowed them. If you quote any text, make sure that it renders as being quoted. (Make sure that you cite your sources -- check with you local instructor for citation preferences).

```
In [ ]: final[(final['part_act17'] == 100) & (final['part_sat17'] < 5)]
```

Three states that I decided to take closer looks at where Mississippi, Louisiana, and Kentucky. I was curious as to why these states had 100% ACT participation, how that was possible, and why their SAT participation was so low (2-4%). I found that according to Priinceton Review those states have ACT mandates that require all students to take the ACT. For Louisiana I found that the test is paid for and lower income students get multiple tests paid for. The point of geearing the students towards the ACT is more so about preparing them for college and less so about the actual test. Goals outlined from the initiative seemed to be about recognizing and preparing for college early on and making sure graduates move on to college but more importantly trying to help lower income students move onto college, something that is often hindered by the price of standardized testing.

Kentucky's mandateis for 2018-2018 and can be renewed for another 4 years. The individual decision to choose the ACT as a vendor made be unknown to the public however it seems logical that the ACT is chosen as an exit exam over the SAT because it covers a broader range of subjects including science, which is not available on the SAT. Again however the aim seems to be to make sure students are college ready and that lower income students have the opportunity to go onto college as the test is administered for free.

Mississippi has these mandates as well for making students take the ACT. The logic as to why seems to make sense regardless of whatever monetary incentives may exist in terms of lobbying. If students are expected, prepared, and given a free opportunity to take one of these college entrance exams then it's only logical to assume they have a better chance of attending college, which is the primary goal for most education systems

Princeton Review Source: [https://www.princetonreview.com/college-advice/act-sat-state-requirements (https://www.princetonreview.com/college-advice/act-sat-state-requirements)](https://www.princetonreview.com/college-advice/act-sat-state-requirements)
Kentucky ACT Mandate: [https://education.ky.gov/AA/Assessments/Pages/ACT.aspx (https://education.ky.gov/AA/Assessments/Pages/ACT.aspx)](https://education.ky.gov/AA/Assessments/Pages/ACT.aspx)

Conclusions and Recommendations

Based on your exploration of the data, what are you key takeaways and recommendations? Choose one state with a lower participation rate and provide a suggestion for how the College Board might increase participation amongst graduating seniors in this state. Are there additional data you desire that would better inform your investigations?

In [393]:
```
final[(final['part_sat17'] < 50) & (final['part_act17'] < 50)]
```

Out[393]:

	state	part_act17	eng_act17	math_act17	read_act17	sci_act17	total_act17	part_sat17	read
37	Oregon	40.0	21.2	21.5	22.4	21.7	21.8	43.0	

State of Analysis: Oregon

I decided to do my research instead on Oregon, a state with low participation rates for both tests. In my research (README) I found that there are large discrepancies in education between low and high income students in terms of recieving funding per student, resulting in gaps in achievement. Oregon also has one of the lowest graduation rates in the country. These factors obviously then affect in state enrollment which has been on a decline since 2012.

My recommendation was that since this area has a lot of room for growth, would be to start a campaign so that, like Louisianna, Mississippi, and Kentucky, Oregon legislators can be convinced that making the SAT mandatory will help improve college enrollment. Collegeboard could fund a small underperforming school to provide tutoring, SAT prep, and free SAT tests, and then see how much more students eventually enroll in college compared to their peers in other schools. This could then be used to convince Oregon's education department to use the SAT as the official test vendor for their state.

Bonus: Descriptive and Inferential Statistics

Summarizing Distributions

Above, we used pandas `describe` to provide quick summary statistics of our numeric columns. We also demonstrated many visual relationships.

As data scientists, having a complete understanding of data is imperative prior to modeling.

While we will continue to build our analytic tools, we know that measures of ***central tendency***, ***spread***, and ***shape/skewness*** provide a quick summary of distributions.

For each variable in your data, summarize the underlying distributions (in words & statistics)

- Be thorough in your verbal description of these distributions.
- Be sure to back up these summaries with statistics.

Answers:

We generally assuming that data we sample from a population will be normally distributed. Do we observe this trend?

Answer:

Does This Assumption Hold for:

```
- Math
- Reading
- Rates
```

Explain your answers for each distribution and how you think this will affect estimates made from these data.

Answer:

Estimate Limits of Data

Suppose we only seek to understand the relationship between SAT and ACT participation rates in 2017.

Does it make sense to conduct statistical inference given these data specifically?

Why or why not?

(think about granularity, aggregation, the relationships between populations size & rates...consider the actually populations these data describe in answering this question)

Answer:

Is it appropriate to compare these specific SAT and ACT math scores?

Why or why not?

Answer:

Statistical Evaluation of Distributions

If you feel it's appropriate, using methods we discussed in class, run hypothesis tests to compare variables of interest in our dataset.

```
In [ ]: # Code:
```