Week 4-Visualization

January 16, 2023

[2]:
```
import pandas
import numpy
import seaborn
import matplotlib.pyplot as plt
```

[3]:
```
data = pandas.read_csv('nesarc_pds.csv', low_memory=False)
```

[4]:
```
print (len(data))
```

```
43093
```

[5]:
```
print (len(data.columns))
```

```
3010
```

[6]:
```
# Convert data types from 'Object' to 'Float'
data["S2AQ19"] = data["S2AQ19"].apply(pandas.to_numeric,errors="coerce")
data["S4AQ1"] = data["S4AQ1"].apply(pandas.to_numeric,errors="coerce")
data["S5Q1"] = data["S5Q1"].apply(pandas.to_numeric,errors="coerce")
data["S5Q3"] = data["S5Q3"].apply(pandas.to_numeric,errors="coerce")
```

[7]:
```
# Determine data types for variables of interest post change to 'Float'
data['S2AQ19'].dtype
data['S4AQ1'].dtype
data['S5Q1'].dtype
data['S5Q3'].dtype
```

[7]:
```
dtype('int64')
```

[8]:
```
#Adding more descriptive titles for key variables
print('Counts for S2AQ19: AGE AT START OF PERIOD OF HEAVIEST DRINKING')
c3 = data['S2AQ19'].value_counts(sort = False, normalize=False).sort_index()
print (c3)
#
print('Normalized counts for S2AQ19: AGE AT START OF PERIOD OF HEAVIEST␣
 ↪DRINKING')
p3 = data['S2AQ19'].value_counts(sort = False, normalize=True).sort_index()
print (p3)
```

```
#
print('Counts for S4AQ1: EVER HAD 2-WEEK PERIOD WHEN FELT SAD, BLUE, DEPRESSED,␣
 ↪OR DOWN MOST OF TIME')
c6 = data['S4AQ1'].value_counts(sort = False, normalize=False).sort_index()
print (c6)
#
print('Normalized counts for S4AQ1: EVER HAD 2-WEEK PERIOD WHEN FELT SAD, BLUE,␣
 ↪DEPRESSED, OR DOWN MOST OF TIME')
p6 = data['S4AQ1'].value_counts(sort = False, normalize=True).sort_index()
print (p6)
#
print('Counts for S5Q1: HAD 1+ WEEK PERIOD OF EXCITEMENT/ELATION THAT SEEMED␣
 ↪NOT NORMAL SELF')
c9 = data['S5Q1'].value_counts(sort = False, normalize=False).sort_index()
print (c9)
#
print('Normalized counts for S5Q1: HAD 1+ WEEK PERIOD OF EXCITEMENT/ELATION␣
 ↪THAT SEEMED NOT NORMAL SELF')
p9 = data['S5Q1'].value_counts(sort = False, normalize=True).sort_index()
print (p9)
#
print('Counts for S5Q3 : D 1+ WEEK PERIOD IRRITABLE/EASILY ANNOYED THAT CAUSED␣
 ↪YOU TO SHOUT/BREAK THINGS/START FIGHTS OR ARGUMENTS')
c10 = data['S5Q3'].value_counts(sort = False, normalize=False).sort_index()
print (c10)
#
print('Normalized counts for S5Q3 : D 1+ WEEK PERIOD IRRITABLE/EASILY ANNOYED␣
 ↪THAT CAUSED YOU TO SHOUT/BREAK THINGS/START FIGHTS OR ARGUMENTS')
p10 = data['S5Q3'].value_counts(sort = False, normalize=True).sort_index()
print (p10)
```

```
Counts for S2AQ19: AGE AT START OF PERIOD OF HEAVIEST DRINKING
5.0       70
6.0        3
7.0        2
8.0       10
9.0        6
        …
87.0       3
88.0       2
90.0       3
91.0       1
99.0    1409
Name: S2AQ19, Length: 87, dtype: int64
Normalized counts for S2AQ19: AGE AT START OF PERIOD OF HEAVIEST DRINKING
5.0     0.002010
6.0     0.000086
```

```
7.0     0.000057
8.0     0.000287
9.0     0.000172
          …
87.0    0.000086
88.0    0.000057
90.0    0.000086
91.0    0.000029
99.0    0.040457
Name: S2AQ19, Length: 87, dtype: float64
Counts for S4AQ1: EVER HAD 2-WEEK PERIOD WHEN FELT SAD, BLUE, DEPRESSED, OR DOWN
MOST OF TIME
1    12785
2    29416
9      892
Name: S4AQ1, dtype: int64
Normalized counts for S4AQ1: EVER HAD 2-WEEK PERIOD WHEN FELT SAD, BLUE,
DEPRESSED, OR DOWN MOST OF TIME
1    0.296684
2    0.682617
9    0.020699
Name: S4AQ1, dtype: float64
Counts for S5Q1: HAD 1+ WEEK PERIOD OF EXCITEMENT/ELATION THAT SEEMED NOT NORMAL
SELF
1     2805
2    39164
9     1124
Name: S5Q1, dtype: int64
Normalized counts for S5Q1: HAD 1+ WEEK PERIOD OF EXCITEMENT/ELATION THAT SEEMED
NOT NORMAL SELF
1    0.065092
2    0.908825
9    0.026083
Name: S5Q1, dtype: float64
Counts for S5Q3 : D 1+ WEEK PERIOD IRRITABLE/EASILY ANNOYED THAT CAUSED YOU TO
SHOUT/BREAK THINGS/START FIGHTS OR ARGUMENTS
1     3402
2    38620
9     1071
Name: S5Q3, dtype: int64
Normalized counts for S5Q3 : D 1+ WEEK PERIOD IRRITABLE/EASILY ANNOYED THAT
CAUSED YOU TO SHOUT/BREAK THINGS/START FIGHTS OR ARGUMENTS
1    0.078946
2    0.896201
9    0.024853
Name: S5Q3, dtype: float64
```

[9]:

```
# Reduce data set to drinkers <21yrs old
sub1=data[(data['S2AQ19']<=21)]
print (len(sub1))
```

```
14162
```

[10]:

```
# Convert data types from 'Object' to 'Float'
sub1["S2AQ19"] = sub1["S2AQ19"].apply(pandas.to_numeric,errors="coerce")
sub1["S4AQ1"] = sub1["S4AQ1"].apply(pandas.to_numeric,errors="coerce")
sub1["S4AQ6A"] = sub1["S4AQ6A"].apply(pandas.to_numeric, errors="coerce")
sub1["S5Q1"] = sub1["S5Q1"].apply(pandas.to_numeric,errors="coerce")
sub1["S5Q3"] = sub1["S5Q3"].apply(pandas.to_numeric,errors="coerce")
sub1["S5Q8B"] = sub1["S5Q8B"].apply(pandas.to_numeric,errors="coerce")
```

[11]:

```
print('Frequency Table for S2AQ19: AGE AT START OF PERIOD OF HEAVIEST DRINKING␣
 ↪[5-21 Age; 99. Unknown; BL. NA, lifetime abstainer]')
c7 = sub1['S2AQ19'].value_counts(sort = False, normalize=False).sort_index()
print (c7)
```

```
Frequency Table for S2AQ19: AGE AT START OF PERIOD OF HEAVIEST DRINKING [5-21
Age; 99. Unknown; BL. NA, lifetime abstainer]
5.0       70
6.0        3
7.0        2
8.0       10
9.0        6
10.0      16
11.0       9
12.0      39
13.0      52
14.0     138
15.0     309
16.0     772
17.0    1214
18.0    3347
19.0    1959
20.0    2380
21.0    3836
Name: S2AQ19, dtype: int64
```

[12]:

```
print('Frequency Table for Percentage of Drinkers <21yrs old by age')
pt2 = sub1.groupby('S2AQ19').size() * 100 / len(sub1)
print(pt2)
#
print()
```

```
Frequency Table for Percentage of Drinkers <21yrs old by age
S2AQ19
```

```
5.0      0.494280
6.0      0.021183
7.0      0.014122
8.0      0.070611
9.0      0.042367
10.0     0.112978
11.0     0.063550
12.0     0.275385
13.0     0.367180
14.0     0.974439
15.0     2.181895
16.0     5.451207
17.0     8.572236
18.0    23.633668
19.0    13.832792
20.0    16.805536
21.0    27.086570
dtype: float64
```

[13]:

```
# Data Management Action 1: Set aside missing data
sub1['S4AQ1']=sub1['S4AQ1'].replace(9,numpy.nan)
sub1['S5Q1']=sub1['S5Q1'].replace(9,numpy.nan)
sub1['S5Q3']=sub1['S5Q3'].replace(9,numpy.nan)
```

[14]:

```
# Data Management Action 1: Frequency tables to confirm '9' missing values have␣
 ↪been coded out
print('Counts for S4AQ1: EVER HAD 2-WEEK PERIOD WHEN FELT SAD, BLUE, DEPRESSED,␣
 ↪OR DOWN MOST OF TIME')
c6 = sub1['S4AQ1'].value_counts(sort = False, normalize=False).sort_index()
print (c6)
#
print('Normalized counts for S4AQ1: EVER HAD 2-WEEK PERIOD WHEN FELT SAD, BLUE,␣
 ↪DEPRESSED, OR DOWN MOST OF TIME')
p6 = sub1['S4AQ1'].value_counts(sort = False, normalize=True).sort_index()
print (p6)
#
print('Counts for S5Q1: HAD 1+ WEEK PERIOD OF EXCITEMENT/ELATION THAT SEEMED␣
 ↪NOT NORMAL SELF')
c9 = sub1['S5Q1'].value_counts(sort = False, normalize=False).sort_index()
print (c9)
#
print('Normalized counts for S5Q1: HAD 1+ WEEK PERIOD OF EXCITEMENT/ELATION␣
 ↪THAT SEEMED NOT NORMAL SELF')
p9 = sub1['S5Q1'].value_counts(sort = False, normalize=True).sort_index()
print (p9)
#
```

```
print('Counts for S5Q3 : D 1+ WEEK PERIOD IRRITABLE/EASILY ANNOYED THAT CAUSED␣
 ↪YOU TO SHOUT/BREAK THINGS/START FIGHTS OR ARGUMENTS')
c10 = sub1['S5Q3'].value_counts(sort = False, normalize=False).sort_index()
print (c10)
#
print('Normalized counts for S5Q3 : D 1+ WEEK PERIOD IRRITABLE/EASILY ANNOYED␣
 ↪THAT CAUSED YOU TO SHOUT/BREAK THINGS/START FIGHTS OR ARGUMENTS')
p10 = sub1['S5Q3'].value_counts(sort = False, normalize=True).sort_index()
print (p10)
```

```
Counts for S4AQ1: EVER HAD 2-WEEK PERIOD WHEN FELT SAD, BLUE, DEPRESSED, OR DOWN
MOST OF TIME
1.0    4506
2.0    9477
Name: S4AQ1, dtype: int64
Normalized counts for S4AQ1: EVER HAD 2-WEEK PERIOD WHEN FELT SAD, BLUE,
DEPRESSED, OR DOWN MOST OF TIME
1.0    0.322248
2.0    0.677752
Name: S4AQ1, dtype: float64
Counts for S5Q1: HAD 1+ WEEK PERIOD OF EXCITEMENT/ELATION THAT SEEMED NOT NORMAL
SELF
1.0     1246
2.0    12682
Name: S5Q1, dtype: int64
Normalized counts for S5Q1: HAD 1+ WEEK PERIOD OF EXCITEMENT/ELATION THAT SEEMED
NOT NORMAL SELF
1.0    0.08946
2.0    0.91054
Name: S5Q1, dtype: float64
Counts for S5Q3 : D 1+ WEEK PERIOD IRRITABLE/EASILY ANNOYED THAT CAUSED YOU TO
SHOUT/BREAK THINGS/START FIGHTS OR ARGUMENTS
1.0     1611
2.0    12337
Name: S5Q3, dtype: int64
Normalized counts for S5Q3 : D 1+ WEEK PERIOD IRRITABLE/EASILY ANNOYED THAT
CAUSED YOU TO SHOUT/BREAK THINGS/START FIGHTS OR ARGUMENTS
1.0    0.1155
2.0    0.8845
Name: S5Q3, dtype: float64
```

[15]:

```
# Data Management Action 2: Create secondary varialbe 'MentalHealthScore'
sub1['MentalHealthScore']=sub1['S4AQ1']+sub1['S5Q1']+sub1['S5Q3']
```

[16]:

```
# Data Management Action 2: Frequency table to confirm seconday variable␣
 ↪'MentalHealthScore'
print('Top 25 Rows Confirming MentalHealthScore Calculation')
```

```
sub2=sub1[['IDNUM', 'S4AQ1','S5Q1', 'S5Q3', 'MentalHealthScore']]
sub2.head(25)
```

```
Top 25 Rows Confirming MentalHealthScore Calculation
```

[16]:

```
    IDNUM  S4AQ1  S5Q1  S5Q3  MentalHealthScore
1       2    2.0   2.0   2.0                6.0
3       4    2.0   1.0   2.0                5.0
4       5    2.0   2.0   2.0                6.0
5       6    2.0   2.0   2.0                6.0
6       7    1.0   1.0   1.0                3.0
8       9    1.0   2.0   1.0                4.0
9      10    2.0   1.0   2.0                5.0
12     13    1.0   2.0   2.0                5.0
16     17    2.0   2.0   2.0                6.0
17     18    1.0   2.0   2.0                5.0
19     20    1.0   2.0   1.0                4.0
21     22    2.0   2.0   2.0                6.0
24     25    2.0   2.0   2.0                6.0
30     31    2.0   2.0   2.0                6.0
31     32    1.0   2.0   1.0                4.0
37     38    1.0   2.0   2.0                5.0
39     40    2.0   2.0   2.0                6.0
40     41    1.0   2.0   2.0                5.0
41     42    2.0   2.0   2.0                6.0
44     45    1.0   2.0   1.0                4.0
45     46    2.0   2.0   2.0                6.0
51     52    2.0   2.0   2.0                6.0
52     53    2.0   2.0   2.0                6.0
53     54    2.0   2.0   2.0                6.0
54     55    2.0   2.0   2.0                6.0
```

[17]:

```
# Data Management Action 3: Grouping values within individual variables to␣
 ↪create MentalHealthCondition based off MentalHealthScore
def MentalHealthCondition (row):
    if row['MentalHealthScore'] == 3:
        return 'Yes'
    if row['MentalHealthScore'] > 3:
        return 'No'

sub1['MentalHealthCondition'] = sub1.apply (lambda row: MentalHealthCondition␣
 ↪(row), axis=1)

print('Top 25 Rows Confirming MentalHealthCondition Calculation')
sub2=sub1[['IDNUM', 'S2AQ19','S4AQ1','S5Q1', 'S5Q3', 'MentalHealthScore',␣
 ↪'MentalHealthCondition']]
sub2.head(25)
```

```
Top 25 Rows Confirming MentalHealthCondition Calculation
```

```
[17]:     IDNUM  S2AQ19  S4AQ1  S5Q1  S5Q3  MentalHealthScore MentalHealthCondition
      1       2    21.0    2.0   2.0   2.0                6.0                    No
      3       4    16.0    2.0   1.0   2.0                5.0                    No
      4       5    18.0    2.0   2.0   2.0                6.0                    No
      5       6    18.0    2.0   2.0   2.0                6.0                    No
      6       7    18.0    1.0   1.0   1.0                3.0                   Yes
      8       9    21.0    1.0   2.0   1.0                4.0                    No
      9      10    17.0    2.0   1.0   2.0                5.0                    No
      12     13    21.0    1.0   2.0   2.0                5.0                    No
      16     17    18.0    2.0   2.0   2.0                6.0                    No
      17     18    20.0    1.0   2.0   2.0                5.0                    No
      19     20    18.0    1.0   2.0   1.0                4.0                    No
      21     22    19.0    2.0   2.0   2.0                6.0                    No
      24     25    21.0    2.0   2.0   2.0                6.0                    No
      30     31    19.0    2.0   2.0   2.0                6.0                    No
      31     32    17.0    1.0   2.0   1.0                4.0                    No
      37     38    20.0    1.0   2.0   2.0                5.0                    No
      39     40    20.0    2.0   2.0   2.0                6.0                    No
      40     41    16.0    1.0   2.0   2.0                5.0                    No
      41     42    18.0    2.0   2.0   2.0                6.0                    No
      44     45    15.0    1.0   2.0   1.0                4.0                    No
      45     46    19.0    2.0   2.0   2.0                6.0                    No
      51     52    20.0    2.0   2.0   2.0                6.0                    No
      52     53    19.0    2.0   2.0   2.0                6.0                    No
      53     54    20.0    2.0   2.0   2.0                6.0                    No
      54     55    19.0    2.0   2.0   2.0                6.0                    No
```

```
[18]: # Data Management Action 3: Frequency table for grouping values within␣
       ↪individual variables to create MentalHealthCondition based off␣
       ↪MentalHealthScore
      print('Counts for MentalHealthCondition; 1 = BiPolar; 2 = Not BiPolar')
      c11 = sub2['MentalHealthCondition'].value_counts(sort = False, normalize=False).
       ↪sort_index()
      print (c11)
      #
      print('Percentages for for MentalHealthCondition; 1 = BiPolar; 2 = Not BiPolar')
      p11 = sub2['MentalHealthCondition'].value_counts(sort = False, normalize=True).
       ↪sort_index()
      print (p11)
      #
```

```
Counts for MentalHealthCondition; 1 = BiPolar; 2 = Not BiPolar
No     13472
Yes      423
Name: MentalHealthCondition, dtype: int64
Percentages for for MentalHealthCondition; 1 = BiPolar; 2 = Not BiPolar
```

```
No     0.969557
Yes    0.030443
Name: MentalHealthCondition, dtype: float64
```

[19]:

```
# After applying the various data managment actions and creating frequency␣
 ↪tables, the number of people who experience BiPolar disorder is 423 of 14162,
# or ~3%. This count excludes those with missing data. This shows that BiPolar␣
 ↪disorder is experienced by a small portion of the population.
# However, the number of observations for responsdents with BiPolar disorder␣
 ↪should be enough to show a correlation with drinking at a young age.
# Further analysis can be done on individual aspects of BiPolar disorder that␣
 ↪occur in isolation, including depression, anger and elevation
```

[20]:

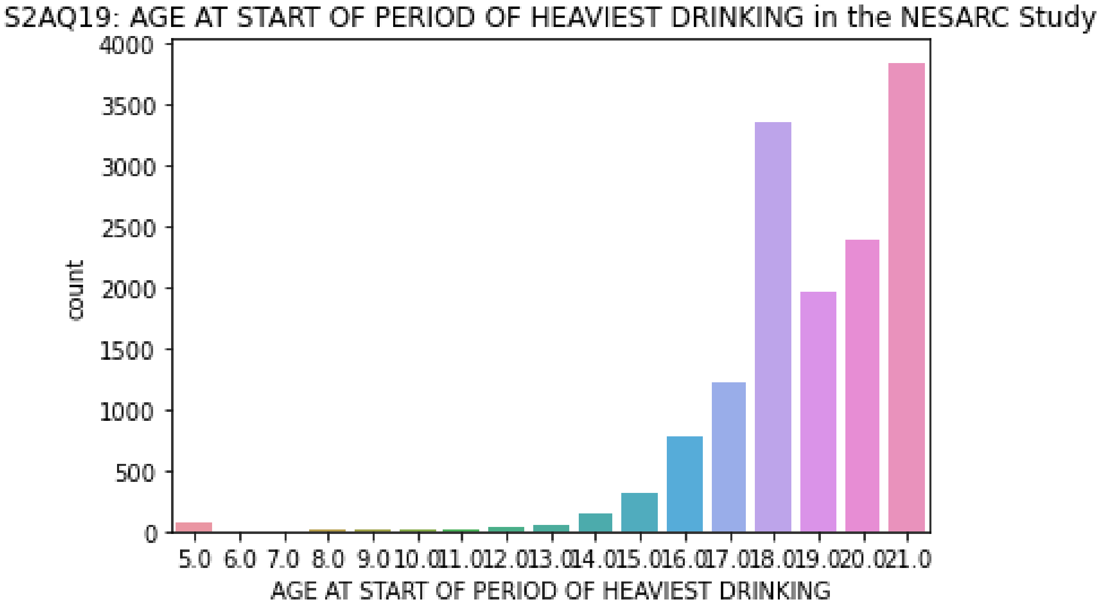
```
# Counts for single conditions and MentalHealthCondition
print('Percentages for depression only: 1=Yes, 2=No')
p11 = sub2['S4AQ1'].value_counts(sort = False, normalize=True).sort_index()
print (p11)
#
print('Percentages for elation only: 1=Yes, 2=No')
p12 = sub2['S5Q1'].value_counts(sort = False, normalize=True).sort_index()
print (p12)
#
print('Percentages for irritable only: 1=Yes, 2=No')
p13 = sub2['S5Q3'].value_counts(sort = False, normalize=True).sort_index()
print (p13)
#
print('Percentages for for MentalHealthCondition; 1 = BiPolar; 2 = Not BiPolar')
p14 = sub2['MentalHealthCondition'].value_counts(sort = False, normalize=True).
 ↪sort_index()
print (p14)
#
```

```
Percentages for depression only: 1=Yes, 2=No
1.0    0.322248
2.0    0.677752
Name: S4AQ1, dtype: float64
Percentages for elation only: 1=Yes, 2=No
1.0    0.08946
2.0    0.91054
Name: S5Q1, dtype: float64
Percentages for irritable only: 1=Yes, 2=No
1.0    0.1155
2.0    0.8845
Name: S5Q3, dtype: float64
Percentages for for MentalHealthCondition; 1 = BiPolar; 2 = Not BiPolar
No     0.969557
Yes    0.030443
```

```
Name: MentalHealthCondition, dtype: float64
```

[21]:
```
# Univariant bar graph for categorical variable S2AQ19
# First change format from numerical to categorical
sub2["S2AQ19"] = sub2["S2AQ19"].astype('category')
seaborn.countplot(x="S2AQ19", data=sub1)
plt.xlabel('AGE AT START OF PERIOD OF HEAVIEST DRINKING')
plt.title('S2AQ19: AGE AT START OF PERIOD OF HEAVIEST DRINKING in the NESARC␣
 ↪Study')
```

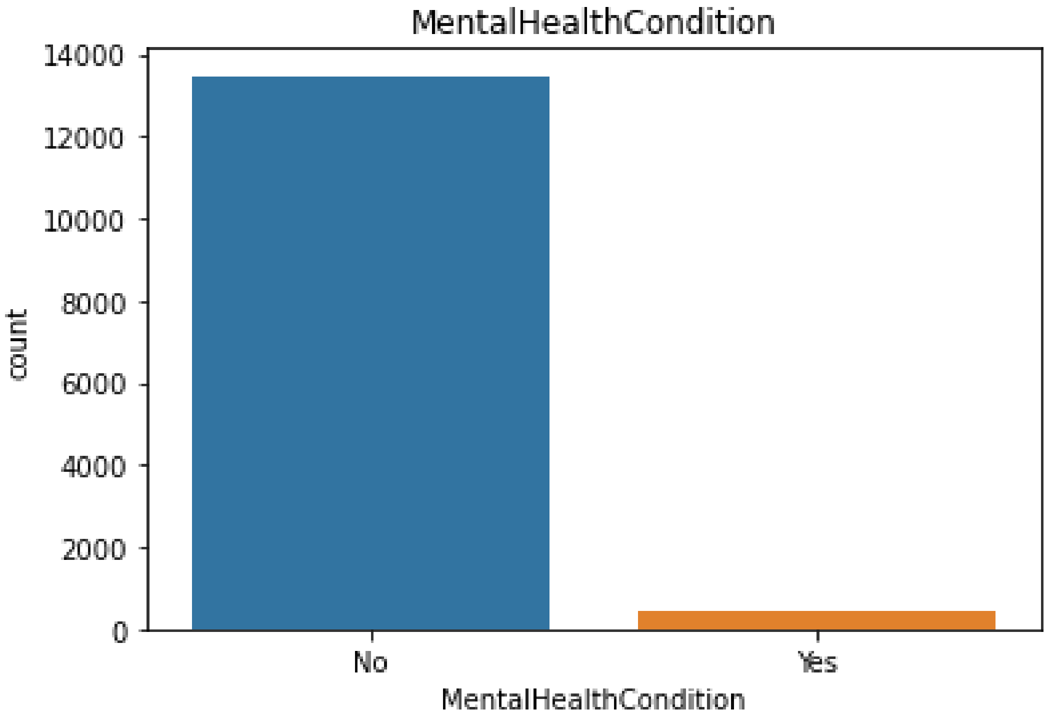
[21]:
```
Text(0.5, 1.0, 'S2AQ19: AGE AT START OF PERIOD OF HEAVIEST DRINKING in the
NESARC Study')
```



[ ]:
```
# The 'S2AQ19: AGE AT START OF PERIOD OF HEAVIEST DRINKING' graph reveals a␣
 ↪bimodal distribution with peaks for young adults
# who startd their heaviest drinking at 18 and 21yrs of age' Since this␣
 ↪analysis is a categorical to categorical, we'll also
# look the relationship of drinking with only depression, elation or␣
 ↪irritability, along with bi-polar disorder
```

[22]:
```
# Univariant bar graph for categorical variable S2AQ19
# First change format from numerical to categorical
sub2["MentalHealthCondition"] = sub2["MentalHealthCondition"].astype('category')
seaborn.countplot(x="MentalHealthCondition", data=sub2)
plt.xlabel('MentalHealthCondition')
plt.title('MentalHealthCondition')
```

```
[22]: Text(0.5, 1.0, 'MentalHealthCondition')
```



```
[41]: # Reduce data set to bi-polar and separate
      sub3=sub1[(sub1['MentalHealthCondition']=='Yes')]
      sub4=sub1[(sub1['MentalHealthCondition']=='No')]
      print (len(sub3))
      print (len(sub4))
```

```
423
13472
```

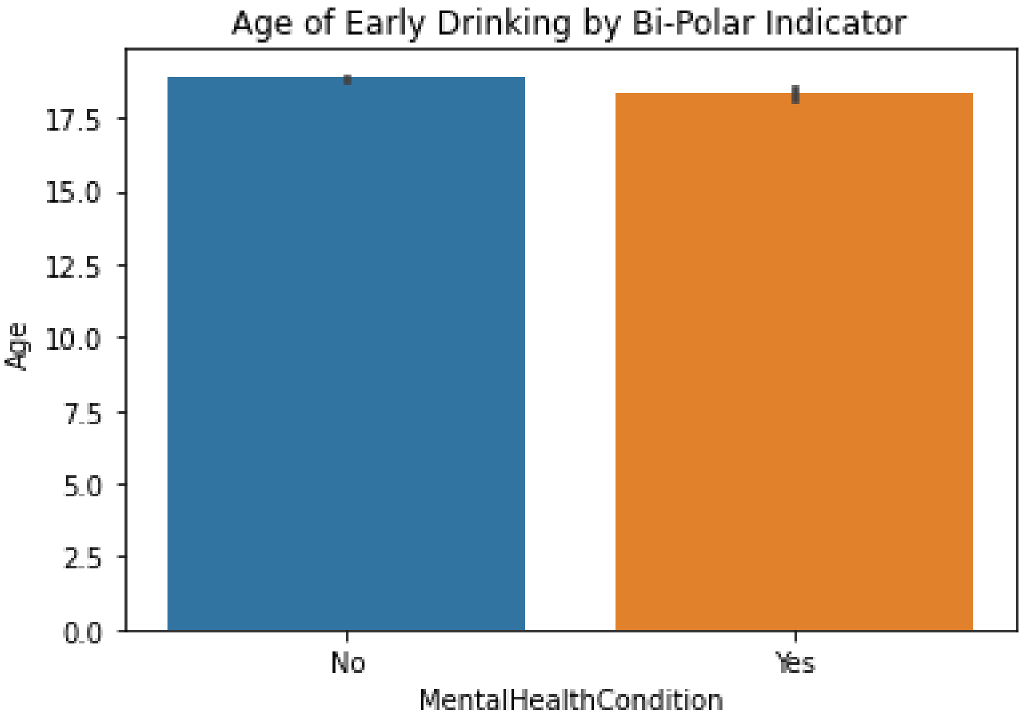
```
[ ]: #The distribution of 'No' is much higher at 423 than 'Yes' at 13472
```

```
[23]: #confirm variables are categorical
      sub2['S2AQ19'] = sub2['S2AQ19'].astype('category')
      sub2['MentalHealthCondition'] = sub2['MentalHealthCondition'].astype('category')
      #seaborn.barplot(x='MentalHealthCondition', y='S2AQ19', data=sub2)
      sub2['S2AQ19'].dtype
      sub2['MentalHealthCondition'].dtype
```

```
[23]: CategoricalDtype(categories=['No', 'Yes'], ordered=False)
```

```
[24]: #Change S2AQ19 back to 'float' to enable comparison of variables
      sub2=sub2.explode('S2AQ19')
      sub2['S2AQ19'] = sub2['S2AQ19'].astype('float')
      seaborn.barplot(x='MentalHealthCondition', y='S2AQ19', data=sub2)
      plt.ylabel('Age')
      plt.title('Age of Early Drinking by Bi-Polar Indicator')
```

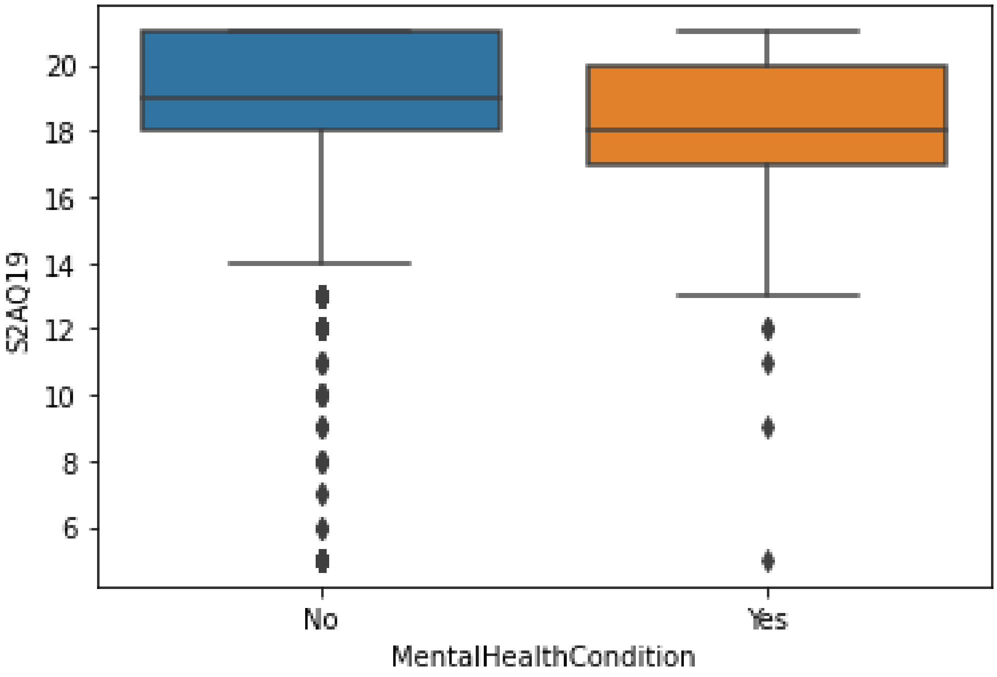
```
[24]: Text(0.5, 1.0, 'Age of Early Drinking by Bi-Polar Indicator')
```



```
[ ]: #The barplot illustraed a slight differences between the age of heavist␣
      ↪drinking and bipolar indicator of 'yes' and 'no';
     #Other graphs will be necessary to illustrate any differences
```

```
[25]: seaborn.boxplot(x='MentalHealthCondition', y='S2AQ19', data=sub2)
```

```
[25]: <matplotlib.axes._subplots.AxesSubplot at 0x7f7e609fd310>
```



```
[ ]: #The boxplot illustrates a difference between the age of heavist drinking and␣
      ↪bipolar indicator of 'yes' and 'no';
     #the average age of heaviest drinking is ~18yrs old for bi-polar while it is␣
      ↪~19yrs old for non bi-polar;
     #the boxplot also shows that the upper and lower quartiles for 'Yes' are below␣
      ↪'No'
```

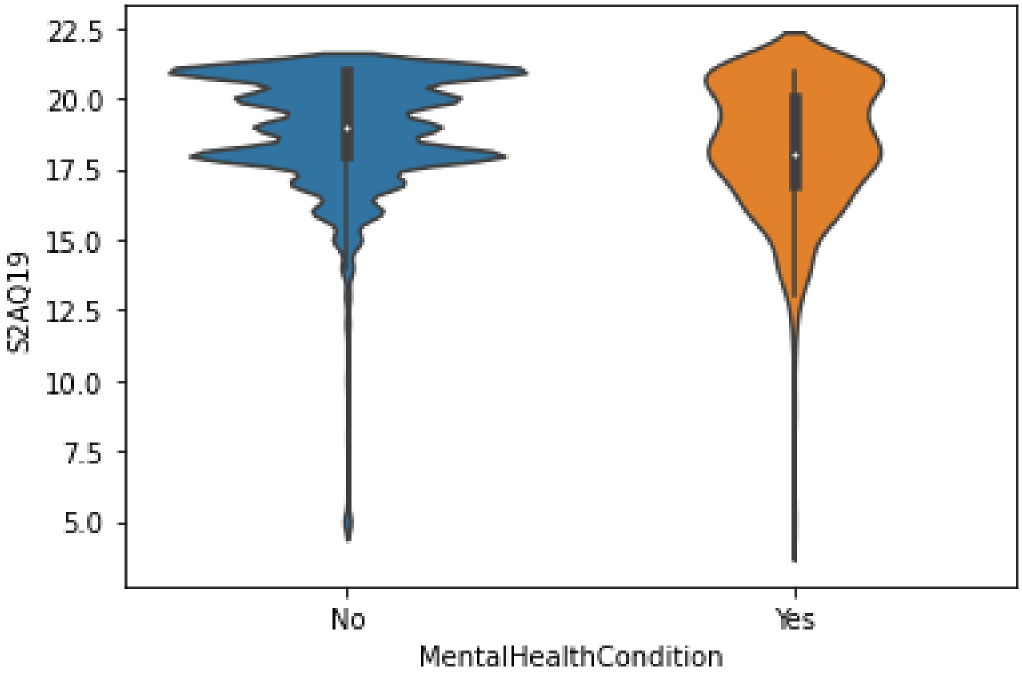
```
[40]: print('mean')
      meanYes = sub3['S2AQ19'].mean()
      meanNo = sub4['S2AQ19'].mean()
      print (meanYes)
      print (meanNo)
```

```
mean
18.35933806146572
18.88078978622328
```

```
[ ]: #A comparison of means confirms the 'No' averge age is approximiatly .5 yrs␣
      ↪older than 'Yes'
```

```
[26]: seaborn.violinplot(x='MentalHealthCondition', y='S2AQ19', data=sub2)
```

```
[26]: <matplotlib.axes._subplots.AxesSubplot at 0x7f7e609697d0>
```



```
[ ]: #The violin chart shows that 'Yes' is denser in general for younger ages,␣
      ↪reducing the average age below that of 'No'
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: