```
In [1]: import pandas
```

```
In [2]: import numpy
```

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

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

        43093
```

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

        3010
```

```
In [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")
```

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

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

```
In [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 DRINKI
        NG')
        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, DEPRESSE
        D, 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, BLU
        E, 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 N
        OT 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 TH
        AT 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 D
OWN 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, DEPR
ESSED, 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 NOR
MAL SELF
1     2805
2    39164
9     1124
Name: S5Q1, dtype: int64
Normalized counts for S5Q1: HAD 1+ WEEK PERIOD OF EXCITEMENT/ELATION THAT SEE
MED 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 T
O 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 C
```

```
AUSED YOU TO SHOUT/BREAK THINGS/START FIGHTS OR ARGUMENTS
1    0.078946
2    0.896201
9    0.024853
Name: S5Q3, dtype: float64
```

In [29]:

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

```
14162
```

In [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["S5Q1"] = sub1["S5Q1"].apply(pandas.to_numeric,errors="coerce")
sub1["S5Q3"] = sub1["S5Q3"].apply(pandas.to_numeric,errors="coerce")
```

In [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
```

In [28]:

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

```
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
```

In [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)
```

In [33]:

```
# 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 D
OWN MOST OF TIME
1    4506
2    9477
9     179
Name: S4AQ1, dtype: int64
Normalized counts for S4AQ1: EVER HAD 2-WEEK PERIOD WHEN FELT SAD, BLUE, DEPR
ESSED, OR DOWN MOST OF TIME
1    0.318175
2    0.669185
9    0.012639
Name: S4AQ1, dtype: float64
Counts for S5Q1: HAD 1+ WEEK PERIOD OF EXCITEMENT/ELATION THAT SEEMED NOT NOR
MAL SELF
1     1246
2    12682
9      234
Name: S5Q1, dtype: int64
Normalized counts for S5Q1: HAD 1+ WEEK PERIOD OF EXCITEMENT/ELATION THAT SEE
MED NOT NORMAL SELF
1    0.087982
2    0.895495
9    0.016523
Name: S5Q1, dtype: float64
Counts for S5Q3 : D 1+ WEEK PERIOD IRRITABLE/EASILY ANNOYED THAT CAUSED YOU T
O SHOUT/BREAK THINGS/START FIGHTS OR ARGUMENTS
1     1611
2    12337
9      214
Name: S5Q3, dtype: int64
Normalized counts for S5Q3 : D 1+ WEEK PERIOD IRRITABLE/EASILY ANNOYED THAT C
AUSED YOU TO SHOUT/BREAK THINGS/START FIGHTS OR ARGUMENTS
1    0.113755
2    0.871134
9    0.015111
Name: S5Q3, dtype: float64
```

In [34]:

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

In [35]:

```
# 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
```

Out[35]:

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

In [36]:

```
# Data Management Action 3: Grouping values within individual variables to create MentalHealthCondition based off MentalHealthScore
def MentalHealthCondition (row):
    if row['MentalHealthScore'] == 3:
        return 1
    if row['MentalHealthScore'] > 3:
        return 2

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

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

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

Out[36]:

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

In [37]:

```
# 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 = sub1['MentalHealthCondition'].value_counts(sort = False, normalize=True).sort_index()
print (p11)
```

```
Counts for MentalHealthCondition; 1 = BiPolar; 2 = Not BiPolar
1      423
2    13739
Name: MentalHealthCondition, dtype: int64
Percentages for for MentalHealthCondition; 1 = BiPolar; 2 = Not BiPolar
1    0.029869
2    0.970131
Name: MentalHealthCondition, dtype: float64
```

In [38]:

```
# 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
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: