Practical 2 - JosiahTeh

December 8, 2021

1 First Name: Josiah

2 Last Name: Teh

[2]:
```
import pandas as pd
import numpy as np
```

[3]:
```
nesarc = pd.read_csv('nesarc.csv', low_memory=False)
pd.set_option('display.float_format', lambda x:'%f'%x)
```

[4]:
```
nesarc['S2AQ5A'] = pd.to_numeric(nesarc['S2AQ5A'], errors='coerce')
nesarc['S2AQ5B'] = pd.to_numeric(nesarc['S2AQ5B'], errors='coerce')
nesarc['S2AQ5D'] = pd.to_numeric(nesarc['S2AQ5D'], errors='coerce')
```

3 For Beer drinking status (S2AQ5A) fill in nan value with 11 & print first 10 rows

[5]:
```
nesarc['S2AQ5A'].head(10) #get count in each category
```

[5]:
```
0        NaN
1   1.000000
2        NaN
3        NaN
4        NaN
5   2.000000
6   2.000000
7   2.000000
8   1.000000
9   2.000000
Name: S2AQ5A, dtype: float64
```

[6]:
```
# hint lecture cell 5
nesarc['S2AQ5A'].fillna(11, inplace=True)
nesarc['S2AQ5A'].head(10)
```

[6]:
```
0   11.000000
1    1.000000
2   11.000000
3   11.000000
4   11.000000
5    2.000000
6    2.000000
7    2.000000
8    1.000000
9    2.000000
Name: S2AQ5A, dtype: float64
```

4 For S2BQ1B1 - Effects of beer drinking (Beer Dependence) in the last 12 months replace 9 (unknown) in S2BQ1B1 (effects of beer consumtion in the last 12 months) to nan

5 & print first 10 rows

[11]:
```
nesarc['S2BQ1B1'] = pd.to_numeric(nesarc['S2BQ1B1'], errors='coerce')
nesarc['S2BQ1B1'].head(10) # get count in each category
```

[11]:
```
0   9.000000
1   0.000000
2   9.000000
3   9.000000
4   9.000000
5   0.000000
6   0.000000
7   0.000000
8   0.000000
9   1.000000
Name: S2BQ1B1, dtype: float64
```

[12]:
```
# hint lecture cell 10
nesarc['S2BQ1B1']= nesarc['S2BQ1B1'].replace(9, np.nan)
nesarc['S2BQ1B1'].head(10)# get count in each category after replacing
```

[12]:
```
0        NaN
1   0.000000
2        NaN
3        NaN
4        NaN
5   0.000000
6   0.000000
7   0.000000
8   0.000000
```

```
9    1.000000
Name: S2BQ1B1, dtype: float64
```

6 Recode S2BQ1B1 so that

7 0 is no

8 1 is yes

9 currently 2 is no

10 & print first 5 rows

[13]:
```
# hint lecture cell 15
recode = {2:0, 1:1}
nesarc['S2BQ1B1']= nesarc['S2BQ1B1'].map(recode)
nesarc['S2BQ1B1'].head(5)
```

[13]:
```
0   NaN
1   NaN
2   NaN
3   NaN
4   NaN
Name: S2BQ1B1, dtype: float64
```

11 Obtain a subset of nesarc data, with the following criteria

12 Age from 26 to 50

13 Beer drinking status - S2AQ5A = Y

[14]:
```
nesarc['AGE'] = pd.to_numeric(nesarc['AGE'])

#subset data to young adults age 26 to 50 who have drink beer in the past 12␣
 ↪months
sub1=nesarc[(nesarc['AGE']>=26) & (nesarc['AGE']<=50) & (nesarc['S2AQ5A']==1)]
```

14 Copy sub 1 to sub 2

[15]:
```
sub2 = sub1.copy()
sub2.head()
len(sub2)
```

[15]:
```
10517
```

15 Use sub2 data

16 Print the count of HOW OFTEN DRANK BEER IN LAST 12 MONTHS (S2AQ5B)

[16]:
```
# hint lecture cell 18
c_beer_feq = sub2['S2AQ5B'].value_counts(sort=False, dropna=False)
print ('counts for original S2AQ5B')
print(c_beer_feq)
```

```
counts for original S2AQ5B
6.000000     1579
5.000000     1485
99.000000      25
4.000000     1310
9.000000     1226
10.000000    1270
8.000000      682
7.000000     1229
1.000000      417
2.000000      369
3.000000      925
Name: S2AQ5B, dtype: int64
```

17 Based on my research, I'm assuming that drinking less than once a month is not going to affect a person. So, we are going replace the following in 'HOW OFTEN DRANK BEER IN LAST 12 MONTHS (S2AQ5B)' to nan

18 8

19 9

20 10

21 99

[17]:
```
# hint lecture cell 9
sub2['S2AQ5B']= sub2['S2AQ5B'].replace(8, np.nan)
sub2['S2AQ5B']= sub2['S2AQ5B'].replace(9, np.nan)
sub2['S2AQ5B']= sub2['S2AQ5B'].replace(10, np.nan)
sub2['S2AQ5B']= sub2['S2AQ5B'].replace(99, np.nan)
```

22 Use sub2 data

23 Print the count of HOW OFTEN DRANK BEER IN LAST 12 MONTHS (S2AQ5B) with 8, 9, 10 and 99 set nan

[18]:
```
# hint lecture cell 18
c_beer_feq_nan = sub2['S2AQ5B'].value_counts(sort=False, dropna=False)
print ('counts for original S2AQ5B with 8, 9, 10 and 99 set to NAN ')
print(c_beer_feq_nan)
```

```
counts for original S2AQ5B with 8, 9, 10 and 99 set to NAN
NaN         3203
6.000000    1579
5.000000    1485
4.000000    1310
7.000000    1229
1.000000     417
2.000000     369
3.000000     925
Name: S2AQ5B, dtype: int64
```

24 Use sub2 data

25 Count the NUMBER OF BEERS USUALLY CONSUMED ON DAYS WHEN DRANK BEER IN LAST 12 MONTHS (S2AQ5D)

[19]:
```
# hint lecture cell 18
c_beer_quan = sub2['S2AQ5D'].value_counts(sort=False, dropna=False)
print ('counts for S2AQ5D')
print(c_beer_quan)
```

```
counts for S2AQ5D
15.000000       7
6.000000      702
18.000000      12
5.000000      278
20.000000       3
17.000000       1
99.000000      31
4.000000      749
9.000000       19
11.000000       1
10.000000      53
8.000000      106
7.000000       57
```

```
14.000000       3
30.000000       1
24.000000      12
1.000000     3625
12.000000     150
13.000000       1
2.000000     3087
3.000000     1619
Name: S2AQ5D, dtype: int64
```

26 Use sub2

27 Replace the 99 in 'NUMBER OF BEERS USUALLY CONSUMED ON DAYS WHEN DRANK BEER IN LAST 12 MONTHS (S2AQ5D)' to nan

[20]:
```
# hint lecture cell 10
sub2['S2AQ5D']= sub2['S2AQ5D'].replace(99, np.nan)
```

28 Print the count of 'NUMBER OF BEERS USUALLY CONSUMED ON DAYS WHEN DRANK BEER IN LAST 12 MONTHS (S2AQ5D)'- with 99 set to NAN

[21]:
```
# hint lecture cell 18
c_beer_quan_nan = sub2['S2AQ5D'].value_counts(sort=False, dropna=False)
print ('counts for S2AQ5D with 99 set to NAN')
print(c_beer_quan_nan)
```

```
counts for S2AQ5D with 99 set to NAN
NaN            31
15.000000       7
6.000000      702
18.000000      12
5.000000      278
20.000000       3
17.000000       1
4.000000      749
9.000000       19
11.000000       1
10.000000      53
8.000000      106
7.000000       57
14.000000       3
30.000000       1
24.000000      12
```

```
1.000000     3625
12.000000     150
13.000000       1
2.000000     3087
3.000000     1619
Name: S2AQ5D, dtype: int64
```

29 Use sub2

30 Recode HOW OFTEN DRANK BEER IN LAST 12 MONTHS (S2AQ5B)

31 as following

32 1 to 7

33 2 to 6

34 3 to 5

35 5 to 3

36 6 to 2

37 7 to 1

38 so that larger categorical numbers indicate more frequently someone drinks beer

39 print the count for BEER-FEQ

[22]:
```
# hint lecture cell 15
recode1 = {1:7, 2:6, 3:5, 5:3, 6:2, 7:1} #recoding so that higher numbers mean␣
 ↪more smoking frequency
sub2['BEER_FEQ'] = sub2['S2AQ5B'].map(recode1)

recode_beer_feq = sub2['BEER_FEQ'].value_counts(sort=False, dropna=False) #get␣
 ↪count in each category
print ('counts for S2AQ5B')
print(recode_beer_feq)
```

```
counts for S2AQ5B
NaN         4513
6.000000     369
5.000000     925
```

```
7.000000     417
1.000000    1229
2.000000    1579
3.000000    1485
Name: BEER_FEQ, dtype: int64
```

40 Use sub 2

41 Recode HOW OFTEN DRANK BEER IN LAST 12 MONTHS (S2AQ5B)

42 as following

43 1 to 30

44 2 to 26

45 3 to 14

46 4 to 8

47 5 to 4

48 6 to 2.5

49 7 to 1

50 so that larger categorical numbers indicate more frequently someone drinks beer

51 print count of BEER_REQMO

[23]:
```
# hint lecture cell 15
#recoding values for S2AQ5B into a new variable, BEER_FEQMO
recode2 = {1:30, 2:26, 3:14, 4:8, 5:4, 6:2.5, 7:1} #recode to quantitative␣
 ↪variable
sub2['BEER_FEQMO'] = sub2['S2AQ5B'].map(recode2)

recode_beer_feq_m = sub2['BEER_FEQMO'].value_counts(sort=False, dropna=False)␣
 ↪#get new count in each category
print ('counts for BEER_FEQMO')
print(recode_beer_feq_m)
```

```
counts for BEER_FEQMO
NaN          3203
```

```
2.500000     1579
26.000000     369
4.000000     1485
8.000000     1310
14.000000     925
30.000000     417
1.000000     1229
Name: BEER_FEQMO, dtype: int64
```

52 Use sub2

53 Create secondary variable NUMBEERMO_EST

54 NUMBEERMO_EST = BEER_FEQMO * S2AQ5D

[24]:
```
# hint lecture cell 17
sub2['NUMBEERMO_EST']= sub2['BEER_FEQMO']*sub2['S2AQ5D']#get the number of␣
 ↪beers consumed  per month
sub2['NUMBEERMO_EST'].head()
```

[24]:
```
1          NaN
8          NaN
12    4.000000
16         NaN
24         NaN
Name: NUMBEERMO_EST, dtype: float64
```

55 print the count for age

[25]:
```
#examining frequency distributions for age
c_age = sub2['AGE'].value_counts(sort=False)
print ('counts for AGE')
print(c_age)
```

```
counts for AGE
32    502
40    497
48    377
33    423
41    445
49    331
26    325
34    462
42    463
50    325
27    397
```

```
35    416
43    398
28    347
36    464
44    381
29    407
37    498
45    434
30    443
38    504
46    396
31    453
39    464
47    365
Name: AGE, dtype: int64
```

56 use sub2

57 print percentag for age

[26]:
```
# hint lecture cell 19
p_age = sub2['AGE'].value_counts(sort=False, dropna=False, normalize=True)
print ('percentages for AGE')
print (p_age)
```

```
percentages for AGE
32   0.047732
40   0.047257
48   0.035847
33   0.040221
41   0.042312
49   0.031473
26   0.030902
34   0.043929
42   0.044024
50   0.030902
27   0.037748
35   0.039555
43   0.037843
28   0.032994
36   0.044119
44   0.036227
29   0.038699
37   0.047352
45   0.041267
30   0.042122
38   0.047922
```

```
46    0.037653
31    0.043073
39    0.044119
47    0.034706
Name: AGE, dtype: float64
```

58 Group age into 3 groups

59 26 - 33

60 34 - 41

61 42 - 50

[28]:
```
# hint lecture cell 20
# splits into 3 groups (26-50) - remember that Python starts counting from 0,␣
 ↪not 1
sub2['AGEGROUP3'] = pd.cut(sub2.AGE, [25, 33, 41, 51])
```

62 print the count of this new group

[30]:
```
# hint lecture cell 20
c_age_group = sub2['AGEGROUP3'].value_counts(sort=False, dropna=False)
print('counts for AGEGROUP3')
print(c_age_group)
```

```
counts for AGEGROUP3
(25, 33]    3297
(33, 41]    3750
(41, 51]    3470
Name: AGEGROUP3, dtype: int64
```

63 print the percentage of this new group

[31]:
```
# hint lecture cell 20
print('percentages for AGEGROUP3')
p_age_group = sub2['AGEGROUP3'].value_counts(sort=False, dropna=False,␣
 ↪normalize=True)
print(p_age_group)
```

```
percentages for AGEGROUP3
(25, 33]    0.313492
(33, 41]    0.356566
(41, 51]    0.329942
Name: AGEGROUP3, dtype: float64
```

64 Print the crosstab between AGEGROUP3 and AGE

[32]:
```
# hint lecture cell 21
#crosstabs evaluating which ages were put into which AGEGROUP3
print (pd.crosstab(sub2['AGEGROUP3'], sub2['AGE']))
```

```
AGE          26   27   28   29   30   31   32   33   34   35  …   41   42  \
AGEGROUP3                                                     …
(25, 33]    325  397  347  407  443  453  502  423    0    0  …    0    0
(33, 41]      0    0    0    0    0    0    0    0  462  416  …  445    0
(41, 51]      0    0    0    0    0    0    0    0    0    0  …    0  463

AGE          43   44   45   46   47   48   49   50
AGEGROUP3
(25, 33]      0    0    0    0    0    0    0    0
(33, 41]      0    0    0    0    0    0    0    0
(41, 51]    398  381  434  396  365  377  331  325

[3 rows x 25 columns]
```