chap01ex

September 15, 2021

1 Examples and Exercises from Think Stats, 2nd Edition

http://thinkstats2.com

Copyright 2016 Allen B. Downey

MIT License: https://opensource.org/licenses/MIT

[1]:
```
from __future__ import print_function, division

import nsfg
```

1.1 Examples from Chapter 1

Read NSFG data into a Pandas DataFrame.

[2]:
```
preg = nsfg.ReadFemPreg()
preg.head()
```

[2]:
```
   caseid  pregordr  howpreg_n  howpreg_p  moscurrp  nowprgdk  pregend1  \
0       1         1        NaN        NaN       NaN       NaN       6.0
1       1         2        NaN        NaN       NaN       NaN       6.0
2       2         1        NaN        NaN       NaN       NaN       5.0
3       2         2        NaN        NaN       NaN       NaN       6.0
4       2         3        NaN        NaN       NaN       NaN       6.0

   pregend2  nbrnaliv  multbrth  ...  laborfor_i  religion_i  metro_i  \
0       NaN       1.0       NaN  ...           0           0        0
1       NaN       1.0       NaN  ...           0           0        0
2       NaN       3.0       5.0  ...           0           0        0
3       NaN       1.0       NaN  ...           0           0        0
4       NaN       1.0       NaN  ...           0           0        0

       basewgt  adj_mod_basewgt      finalwgt  secu_p  sest  cmintvw  \
0  3410.389399      3869.349602   6448.271112       2     9      NaN
1  3410.389399      3869.349602   6448.271112       2     9      NaN
2  7226.301740      8567.549110  12999.542264       2    12      NaN
3  7226.301740      8567.549110  12999.542264       2    12      NaN
4  7226.301740      8567.549110  12999.542264       2    12      NaN
```

```
   totalwgt_lb
0       8.8125
1       7.8750
2       9.1250
3       7.0000
4       6.1875

[5 rows x 244 columns]
```

Print the column names.

```
[3]: preg.columns
```

```
[3]: Index(['caseid', 'pregordr', 'howpreg_n', 'howpreg_p', 'moscurrp', 'nowprgdk',
            'pregend1', 'pregend2', 'nbrnaliv', 'multbrth',
            ...
            'laborfor_i', 'religion_i', 'metro_i', 'basewgt', 'adj_mod_basewgt',
            'finalwgt', 'secu_p', 'sest', 'cmintvw', 'totalwgt_lb'],
           dtype='object', length=244)
```

Select a single column name.

```
[4]: preg.columns[1]
```

```
[4]: 'pregordr'
```

Select a column and check what type it is.

```
[5]: pregordr = preg['pregordr']
     type(pregordr)
```

```
[5]: pandas.core.series.Series
```

Print a column.

```
[7]: pregordr
```

```
[7]: 0        1
     1        2
     2        1
     3        2
     4        3
     5        1
     6        2
     7        3
     8        1
     9        2
     10       1
     11       1
     12       2
     13       3
     14       1
     15       2
     16       3
```

```
17        1
18        2
19        1
20        2
21        1
22        2
23        1
24        2
25        3
26        1
27        1
28        2
29        3
         ..
13563    2
13564    3
13565    1
13566    1
13567    1
13568    2
13569    1
13570    2
13571    3
13572    4
13573    1
13574    2
13575    1
13576    1
13577    2
13578    1
13579    2
13580    1
13581    2
13582    3
13583    1
13584    2
13585    1
13586    2
13587    3
13588    1
13589    2
13590    3
13591    4
13592    5
Name: pregordr, Length: 13593, dtype: int64
```

Select a single element from a column.

[6]:
```
pregordr[0]
```

[6]:
```
1
```

Select a slice from a column.

[7]:
```
pregordr[2:5]
```

[7]:
```
2    1
3    2
4    3
Name: pregordr, dtype: int64
```

Select a column using dot notation.

[8]:
```
pregordr = preg.pregordr
```

Count the number of times each value occurs.

[9]:
```
preg.outcome.value_counts().sort_index()
```

[9]:
```
1    9148
2    1862
3     120
4    1921
5     190
6     352
Name: outcome, dtype: int64
```

Check the values of another variable.

[10]:
```
preg.birthwgt_lb.value_counts().sort_index()
```

[10]:
```
0.0        8
1.0       40
2.0       53
3.0       98
4.0      229
5.0      697
6.0     2223
7.0     3049
8.0     1889
9.0      623
10.0     132
11.0      26
12.0      10
13.0       3
14.0       3
15.0       1
Name: birthwgt_lb, dtype: int64
```

Make a dictionary that maps from each respondent's `caseid` to a list of indices into the pregnancy `DataFrame`. Use it to select the pregnancy outcomes for a single respondent.

[11]:
```
caseid = 10229
preg_map = nsfg.MakePregMap(preg)
```

```
indices = preg_map[caseid]
preg.outcome[indices].values
```

[11]:
```
array([4, 4, 4, 4, 4, 4, 1], dtype=int64)
```

1.2 Exercises

Select the `birthord` column, print the value counts, and compare to results published in the codebook

[12]:
```
preg.birthord.value_counts().sort_index()
```

[12]:
```
1.0     4413
2.0     2874
3.0     1234
4.0      421
5.0      126
6.0       50
7.0       20
8.0        7
9.0        2
10.0       1
Name: birthord, dtype: int64
```

We can also use `isnull` to count the number of nans.

[13]:
```
preg.birthord.isnull().sum()
```

[13]:
```
4445
```

Select the `prglngth` column, print the value counts, and compare to results published in the codebook

[14]:
```
preg.prglngth.value_counts().sort_index()
```

[14]:
```
0       15
1        9
2       78
3      151
4      412
5      181
6      543
7      175
8      409
9      594
10     137
11     202
12     170
13     446
14      29
15      39
16      44
17     253
```

```
18       17
19       34
20       18
21       37
22      147
23       12
24       31
25       15
26      117
27        8
28       38
29       23
30      198
31       29
32      122
33       50
34       60
35      357
36      329
37      457
38      609
39     4744
40     1120
41      591
42      328
43      148
44       46
45       10
46        1
47        1
48        7
50        2
Name: prglngth, dtype: int64
```

To compute the mean of a column, you can invoke the `mean` method on a Series. For example, here is the mean birthweight in pounds:

[15]:
```
preg.totalwgt_lb.mean()
```

[15]:
```
7.265628457623368
```

Create a new column named totalwgt_kg that contains birth weight in kilograms. Compute its mean. Remember that when you create a new column, you have to use dictionary syntax, not dot notation.

[16]:
```
totalwgt_kg = preg['totalwgt_lb']/2.2046
```

`nsfg.py` also provides `ReadFemResp`, which reads the female respondents file and returns a `DataFrame`:

[17]:
```
resp = nsfg.ReadFemResp()
```

`DataFrame` provides a method `head` that displays the first five rows:

[18]:
```
resp.head()
```

[18]:
```
   caseid  rscrinf  rdormres  rostscrn  rscreenhisp  rscreenrace  age_a  \
0    2298        1         5         5            1          5.0     27
1    5012        1         5         1            5          5.0     42
2   11586        1         5         1            5          5.0     43
3    6794        5         5         4            1          5.0     15
4     616        1         5         4            1          5.0     20

   age_r  cmbirth  agescrn  ...  pubassis_i      basewgt  adj_mod_basewgt  \
0     27      902       27  ...           0  3247.916977      5123.759559
1     42      718       42  ...           0  2335.279149      2846.799490
2     43      708       43  ...           0  2335.279149      2846.799490
3     15     1042       15  ...           0  3783.152221      5071.464231
4     20      991       20  ...           0  5341.329968      6437.335772

      finalwgt  secu_r  sest  cmintvw  cmlstyr  screentime   intvlngth
0  5556.717241       2    18     1234     1222    18:26:36  110.492667
1  4744.191350       2    18     1233     1221    16:30:59   64.294000
2  4744.191350       2    18     1234     1222    18:19:09   75.149167
3  5923.977368       2    18     1234     1222    15:54:43   28.642833
4  7229.128072       2    18     1233     1221    14:19:44   69.502667

[5 rows x 3087 columns]
```

Select the age_r column from resp and print the value counts. How old are the youngest and oldest respondents?

[23]:
```
resp.age_r.value_counts().sort_index()
```

[23]:
```
15     217
16     223
17     234
18     235
19     241
20     258
21     267
22     287
23     282
24     269
25     267
26     260
27     255
28     252
29     262
30     292
31     278
32     273
33     257
```

```
34    255
35    262
36    266
37    271
38    256
39    215
40    256
41    250
42    215
43    253
44    235
Name: age_r, dtype: int64
```

We can use the `caseid` to match up rows from `resp` and `preg`. For example, we can select the row from `resp` for `caseid` 2298 like this:

[19]:
```
resp[resp.caseid==2298]
```

[19]:
```
   caseid  rscrinf  rdormres  rostscrn  rscreenhisp  rscreenrace  age_a  \
0    2298        1         5         5            1          5.0     27

   age_r  cmbirth  agescrn  ...  pubassis_i      basewgt  adj_mod_basewgt  \
0     27      902       27  ...           0  3247.916977      5123.759559

      finalwgt  secu_r  sest  cmintvw  cmlstyr  screentime   intvlngth
0  5556.717241       2    18     1234     1222    18:26:36  110.492667

[1 rows x 3087 columns]
```

And we can get the corresponding rows from `preg` like this:

[20]:
```
preg[preg.caseid==2298]
```

[20]:
```
      caseid  pregordr  howpreg_n  howpreg_p  moscurrp  nowprgdk  pregend1  \
2610    2298         1        NaN        NaN       NaN       NaN       6.0
2611    2298         2        NaN        NaN       NaN       NaN       6.0
2612    2298         3        NaN        NaN       NaN       NaN       6.0
2613    2298         4        NaN        NaN       NaN       NaN       6.0

      pregend2  nbrnaliv  multbrth  ...  laborfor_i  religion_i  metro_i  \
2610       NaN       1.0       NaN  ...           0           0        0
2611       NaN       1.0       NaN  ...           0           0        0
2612       NaN       1.0       NaN  ...           0           0        0
2613       NaN       1.0       NaN  ...           0           0        0

          basewgt  adj_mod_basewgt     finalwgt  secu_p  sest  cmintvw  \
2610  3247.916977      5123.759559  5556.717241       2    18      NaN
2611  3247.916977      5123.759559  5556.717241       2    18      NaN
2612  3247.916977      5123.759559  5556.717241       2    18      NaN
2613  3247.916977      5123.759559  5556.717241       2    18      NaN
```

```
      totalwgt_lb
2610       6.8750
2611       5.5000
2612       4.1875
2613       6.8750

[4 rows x 244 columns]
```

How old is the respondent with `caseid` 1?

```
[21]: resp[resp.caseid==1].age_r
```

```
[21]: 1069    44
      Name: age_r, dtype: int64
```

What are the pregnancy lengths for the respondent with `caseid` 2298?

```
[24]: preg[preg.caseid==2298].prglngth
```

```
[24]: 2610    40
      2611    36
      2612    30
      2613    40
      Name: prglngth, dtype: int64
```

What was the birthweight of the first baby born to the respondent with `caseid` 5012?

```
[25]: preg[preg.caseid==5012].birthwgt_lb
```

```
[25]: 5515    6.0
      Name: birthwgt_lb, dtype: float64
```

```
[ ]:
```