Examples and Exercises from Think Stats, 2nd Edition¶

http://thinkstats2.com

Copyright 2016 Allen B. Downey

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

In [1]:

```
from __future__ import print_function, division

import nsfg
```

Examples from Chapter 1¶

Read NSFG data into a Pandas DataFrame.

In [2]:

```
preg = nsfg.ReadFemPreg()
preg.head()
```

Out[2]:

	caseid	pregordr	howpreg_n	howpreg_p	moscurrp	nowprgdk	pregend1	pregend2	nbrnaliv	multbrth
0	1	1	NaN	NaN	NaN	NaN	6.0	NaN	1.0	NaN
1	1	2	NaN	NaN	NaN	NaN	6.0	NaN	1.0	NaN
2	2	1	NaN	NaN	NaN	NaN	5.0	NaN	3.0	5.0
3	2	2	NaN	NaN	NaN	NaN	6.0	NaN	1.0	NaN
4	2	3	NaN	NaN	NaN	NaN	6.0	NaN	1.0	NaN

5 rows Ã— 244 columns

Print the column names.

In [3]:

```
preg.columns
```

Out[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.

In [4]:

```
preg.columns[1]
```

Out[4]:

```
'pregordr'
```

Select a column and check what type it is.

In [5]:

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

Out[5]:

```
pandas.core.series.Series
```

Print a column.

In [6]:

```
pregordr
```

Out[6]:

```
0        1
1        2
2        1
3        2
4        3
```

```
        ..
13588    1
13589    2
13590    3
13591    4
13592    5
Name: pregordr, Length: 13593, dtype: int64
```

Select a single element from a column.

In [7]:

```
pregordr[0]
```

Out[7]:

```
1
```

Select a slice from a column.

In [8]:

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

Out[8]:

```
2    1
3    2
4    3
Name: pregordr, dtype: int64
```

Select a column using dot notation.

In [9]:

```
pregordr = preg.pregordr
```

Count the number of times each value occurs.

In [10]:

```
preg.outcome.value_counts().sort_index()
```

Out[10]:

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

Check the values of another variable.

In [11]:

```
preg.birthwgt_lb.value_counts().sort_index()
```

Out[11]:

```
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.

In [12]:

```
caseid = 10229
```

```
preg_map = nsfg.MakePregMap(preg)
indices = preg_map[caseid]
preg.outcome[indices].values
```

Out[12]:

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

Exercises¶

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

In [13]:

```
# Solution
preg.birthord.value_counts().sort_index()
```

Out[13]:

```
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.

In [14]:

```
preg.birthord.isnull().sum()
```

Out[14]:

```
4445
```

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

In [15]:

```
# Solution
preg.prglngth.value_counts(bins=3).sort_index()
```

Out[15]:

```
(-0.051000000000000004, 16.667]    3634
(16.667, 33.333]                   1149
(33.333, 50.0]                     8810
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:

In [16]:

```
# round off to two decimal digits, used the round (value, 2)
round(preg.totalwgt_lb.mean(),2)
```

Out[16]:

```
7.27
```

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.

In [17]:

```
# Solution
# 1 kg = 2.20462 pounds and 1 pound = 35.274 Ounce
# Rounded off result to 2 decimal digit (value, 2)
preg["totalwgt_kg"] = preg.birthwgt_lb/2.20462 + preg.birthwgt_oz/35.274
round(preg["totalwgt_kg"].mean(),2)
```

Out[17]:

```
3.3
```

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

In [18]:

```
resp = nsfg.ReadFemResp()
```

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

In [19]:

```
resp.head()
```

Out[19]:

	caseid	rscrinf	rdormres	rostscrn	rscreenhisp	rscreenrace	age_a	age_r	cmbirth	agescrn	...	pubassis
0	2298	1	5	5	1	5.0	27	27	902	27	...	0
1	5012	1	5	1	5	5.0	42	42	718	42	...	0
2	11586	1	5	1	5	5.0	43	43	708	43	...	0
3	6794	5	5	4	1	5.0	15	15	1042	15	...	0
4	616	1	5	4	1	5.0	20	20	991	20	...	0

5 rows Ã— 3087 columns

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

In [20]:

```
# Solution
resp.age_r.value_counts().sort_index()
# Youngest respondents are 15 year old
# oldest respondents are 44 years old
```

Out[20]:

```
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:

In [21]:

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

Out[21]:

	caseid	**rscrinf**	**rdormres**	**rostscrn**	**rscreenhisp**	**rscreenrace**	**age_a**	**age_r**	**cmbirth**	**agescrn**	**...**	**pubassis**
0	2298	1	5	5	1	5.0	27	27	902	27	...	0

1 rows Ã— 3087 columns

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

In [22]:

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

Out[22]:

	caseid	**pregordr**	**howpreg_n**	**howpreg_p**	**moscurrp**	**nowprgdk**	**pregend1**	**pregend2**	**nbrnaliv**	**mult**
2610	2298	1	NaN	NaN	NaN	NaN	6.0	NaN	1.0	NaN
2611	2298	2	NaN	NaN	NaN	NaN	6.0	NaN	1.0	NaN
2612	2298	3	NaN	NaN	NaN	NaN	6.0	NaN	1.0	NaN
2613	2298	4	NaN	NaN	NaN	NaN	6.0	NaN	1.0	NaN

4 rows Ã— 245 columns

How old is the respondent with `caseid` 1?

In [23]:

```
# Solution goes here
resp[resp.caseid==1].age_r
```

Out[23]:

```
1069    44
Name: age_r, dtype: int64
```

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

In [24]:

```
# Solution
```

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

Out[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?

In [25]:

```
# Solution
preg[preg.caseid==5012].totalwgt_lb
```

Out[25]:

```
5515    6.0
Name: totalwgt_lb, dtype: float64
```

In [ ]: