DSC 530 Data Exploration and Analysis

Assignment Week3_ Excercises: 1.1, 1.2, 2.1 & 2.4

Author: Zemelak Goraga

Data: 12/16/2023

In [7]:
```
# Download the required input files
from os.path import basename, exists

def download(url):
    filename = basename(url)
    if not exists(filename):
        from urllib.request import urlretrieve

        local, _ = urlretrieve(url, filename)
        print("Downloaded " + local)


download("https://github.com/AllenDowney/ThinkStats2/raw/master/code/think
download("https://github.com/AllenDowney/ThinkStats2/raw/master/code/think
download("https://github.com/AllenDowney/ThinkStats2/raw/master/code/nsfg.

download("https://github.com/AllenDowney/ThinkStats2/raw/master/code/2002F
download(
    "https://github.com/AllenDowney/ThinkStats2/raw/master/code/2002FemPre
)
download("https://github.com/AllenDowney/ThinkStats2/raw/master/code/2002F
download("https://github.com/AllenDowney/ThinkStats2/raw/master/code/2002F
```

```
Downloaded thinkstats2.py
Downloaded thinkplot.py
Downloaded nsfg.py
Downloaded 2002FemPreg.dct
Downloaded 2002FemPreg.dat.gz
Downloaded 2002FemResp.dct
Downloaded 2002FemResp.dat.gz
```

In [8]:
```
# Import required libraries
#from __future__ import print_function

import numpy as np
import sys

import nsfg
import thinkstats2
```

In [18]:
```
# Import and inspect the Resp file
resp = nsfg.ReadFemResp()
resp.head()
```

Out[18]:

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

5 rows × 3087 columns

In [19]:
```
# Import and inspect the Preg file
preg = nsfg.ReadFemPreg()
preg.head()
```

Out[19]:

	caseid	pregordr	howpreg_n	howpreg_p	moscurrp	nowprgdk	pregend1	pregend2	nb
0	1	1	NaN	NaN	NaN	NaN	6.0	NaN	
1	1	2	NaN	NaN	NaN	NaN	6.0	NaN	
2	2	1	NaN	NaN	NaN	NaN	5.0	NaN	
3	2	2	NaN	NaN	NaN	NaN	6.0	NaN	
4	2	3	NaN	NaN	NaN	NaN	6.0	NaN	

5 rows × 244 columns

Type *Markdown* and LaTeX: α^2

Exercise 1.1

In [11]:
```
# Select the birthord column, print the value counts, and compare to resul

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

Out[11]:
```
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
```

In [12]:
```
# Select the prglngth column, print the value counts, and compare to resul

preg.prglngth.value_counts().sort_index()
```

Out[12]:
```
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
```

In [13]:
```
# Create a new column named totalwgt_kg that contains birth weight in kilo
# Compute its mean. Remember that when you create a new column, you have t

# Convert pounds to kilograms (1 lb = 0.453592 kg)
preg['totalwgt_kg'] = preg['totalwgt_lb'] * 0.453592

# Calculate the mean of 'wt_kg'
mean_totalwgt_kg = preg['totalwgt_kg'].mean()

# Display the DataFrame and mean of 'totalwgt_kg'
preg.head()
```

Out[13]:

	caseid	pregordr	howpreg_n	howpreg_p	moscurrp	nowprgdk	pregend1	pregend2	nb
0	1	1	NaN	NaN	NaN	NaN	6.0	NaN	
1	1	2	NaN	NaN	NaN	NaN	6.0	NaN	
2	2	1	NaN	NaN	NaN	NaN	5.0	NaN	
3	2	2	NaN	NaN	NaN	NaN	6.0	NaN	
4	2	3	NaN	NaN	NaN	NaN	6.0	NaN	

5 rows × 245 columns

In [14]:
```
# Mean of totalwgt_kg
print(f"Mean of 'totalwgt_kg': {mean_totalwgt_kg:.2f} kg")
```

```
Mean of 'totalwgt_kg': 3.30 kg
```

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

Answer: The youngest is 15 and the oldest is 44 years old

How old is the respondent with caseid 1?

Answer: 44 years old

In [20]:
```
resp[resp.caseid==1]
```

Out[20]:

	caseid	rscrinf	rdormres	rostscrn	rscreenhisp	rscreenrace	age_a	age_r	cmbirth
1069	1	1	5	4	5	5.0	44	44	695

1 rows × 3087 columns

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

Answer: 110.492667

In [21]:
```
resp[resp.caseid==2298]
```

Out[21]:

	caseid	rscrinf	rdormres	rostscrn	rscreenhisp	rscreenrace	age_a	age_r	cmbirth	ag
0	2298	1	5	5	1	5.0	27	27	902	

1 rows × 3087 columns

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

Answer: 2335.279149

In [22]:
```
resp[resp.caseid==5012]
```

Out[22]:

	caseid	rscrinf	rdormres	rostscrn	rscreenhisp	rscreenrace	age_a	age_r	cmbirth	ag
1	5012	1	5	1	5	5.0	42	42	718	

1 rows × 3087 columns

Exercise 1.2

In [23]:
```
# Value counts for 'pregnum'

import nsfg

# Load the datasets
resp = nsfg.ReadFemResp()
preg = nsfg.ReadFemPreg()

# Print the value counts for 'pregnum' variable in the 'resp' dataset
print("Value counts for 'pregnum' in the 'resp' dataset:")
print(resp['pregnum'].value_counts())

# Create a dictionary mapping caseid to a list of indices into the pregnan
preg_map = nsfg.MakePregMap(preg)

# Create a new column in 'resp' dataset representing the number of records
resp['preg_count'] = resp['caseid'].map(lambda x: len(preg_map.get(x, [])))

# Compare 'pregnum' for each respondent with the number of records in the
comparison_result = resp['pregnum'] == resp['preg_count']

# Print the result
print("\nComparison result:")
print(comparison_result.value_counts())
```

```
Value counts for 'pregnum' in the 'resp' dataset:
0     2610
2     1432
1     1267
3     1110
4      611
5      305
6      150
7       80
8       40
9       21
10       9
11       3
12       2
14       2
19       1
Name: pregnum, dtype: int64

Comparison result:
True    7643
dtype: int64
```

Interpretation

Value Counts for 'pregnum' in the 'resp' Dataset:

0 pregnancies: 2610 respondents 1 pregnancy: 1267 respondents 2 pregnancies: 1432 respondents 3 pregnancies: 1110 respondents 4 pregnancies: 611 respondents 5 pregnancies: 305 respondents 6 pregnancies: 150 respondents 7 pregnancies: 80

respondents 8 pregnancies: 40 respondents 9 pregnancies: 21 respondents 10 pregnancies: 9 respondents 11 pregnancies: 3 respondents 12 pregnancies: 2 respondents 14 pregnancies: 2 respondents 19 pregnancies: 1 respondent

Comparison Result:

True: 7643 respondents False: 0 respondents

Interpretation:

The 'pregnum' variable in the 'resp' dataset indicates the number of pregnancies for each respondent. The vast majority of respondents (7643 out of 7643) have matching values between 'pregnum' and the number of records in the pregnancy dataset ('preg_count'). This implies that, for most respondents, the recorded number of pregnancies in the 'resp' dataset aligns with the actual number of pregnancy records in the pregnancy dataset. The absence of False values in the comparison result suggests that there are no inconsistencies between the reported number of pregnancies and the actual number of pregnancy records for any respondent. In summary, based on the comparison result, the 'pregnum' variable in the 'resp' dataset appears to be consistent with the number of pregnancy records in the pregnancy dataset, indicating a high level of data integrity and reliability in the reported pregnancy information.

Type *Markdown* and LaTeX: α^2

Excercise 2.1

In [24]:
```
# Results of overall analysis of the 'totalwgt_lb' variable

import pandas as pd

# summary statistics for 'totalwgt_lb'
totalwgt_summary = preg['totalwgt_lb'].describe()
print(totalwgt_summary)

# count of missing values in 'totalwgt_lb'
missing_values_totalwgt = preg['totalwgt_lb'].isnull().sum()
print(f"Missing values in 'totalwgt_lb': {missing_values_totalwgt}")
```

```
count    9038.000000
mean        7.265628
std         1.408293
min         0.125000
25%         6.500000
50%         7.375000
75%         8.125000
max        15.437500
Name: totalwgt_lb, dtype: float64
Missing values in 'totalwgt_lb': 4555
```

In [25]:
```
# Results of analysis of the 'totalwgt_lb' variable separatly for the firs

import pandas as pd
from scipy.stats import ttest_ind

# Extracting the 'totalwgt_lb' column and 'pregordr' column
totalwgt_lb = preg['totalwgt_lb']
pregordr = preg['pregordr']

# Separating first babies and others
first_babies = totalwgt_lb[pregordr == 1]
others = totalwgt_lb[pregordr > 1]

# Calculating means and standard deviations for both groups
mean_first_babies = first_babies.mean()
mean_others = others.mean()
std_first_babies = first_babies.std()
std_others = others.std()

# Independent two-sample t-test
t_statistic, p_value = ttest_ind(first_babies, others, equal_var=False)

# Cohen's d
cohens_d = (mean_first_babies - mean_others) / (((std_first_babies**2 + st

print("Mean weight of first babies:", mean_first_babies)
print("Mean weight of others:", mean_others)
print("Cohen's d:", cohens_d)
print("p-value:", p_value)
```

```
Mean weight of first babies: 7.204107733975324
Mean weight of others: 7.301399825021872
Cohen's d: -0.06904986139204107
p-value: nan
```

Results:

Results of overall analysis of the 'totalwgt_lb' variable ...

count 9038.000000 mean 7.265628 std 1.408293 min 0.125000 25% 6.500000 50% 7.375000 75% 8.125000 max 15.437500 Name: totalwgt_lb, dtype: float64 Missing values in 'totalwgt_lb': 4555

Results of analysis of the 'totalwgt_lb' variable separatly for the first babies and other babies ....

Mean Weight of First Babies: 7.20 pounds Mean Weight of Other Babies: 7.30 pounds Cohen's d: -0.069 (negative)

Evening News Summary: For a news story, it would be appropriate to highlight key summary statistics that capture the overall distribution of baby weights. The mean (average) weight of all babies is 7.27 pounds, with a standard deviation of 1.41 pounds. The range of weights spans from 0.13 to 15.44 pounds. This information provides a general overview of baby weights and could be emphasized to give viewers a sense of the typical weight range and variation.

Reassurance for Anxious Patient Summary: To reassure an anxious patient, it's important to focus on the specific comparison between first babies and others. The mean weight of first babies is 7.20 pounds, while the mean weight of other babies is 7.30 pounds. The small effect size (Cohen's d of -0.069) suggests a minor difference between the two groups, and the p-value issue raises caution about the statistical significance of this difference. Patients could be reassured that, on average, the difference in weight is small, and the statistical significance might be influenced by data limitations.

Answer to "Do First Babies Arrive Late?" (as Cecil Adams): In examining the results, it appears that first babies, on average, tend to be slightly lighter than other babies. However, the effect size, as indicated by Cohen's d (-0.069), is small, suggesting that this difference may not be practically substantial. The reported p-value issue raises concerns about the dataset's limitations, possibly due to insufficient variability in one of the groups. While there is a statistically significant difference in mean weights between first babies and others, caution is warranted in interpreting the practical significance of this finding. It's crucial to acknowledge the complexity of factors influencing birth weights and the potential

In [ ]:

Type *Markdown* and LaTeX: α^2

Excercise 2.4.

In [26]:
```
# Results of analysis of the 'totalwgt_lb' variable separatly for the firs

import pandas as pd
from scipy.stats import ttest_ind

# Extracting the 'totalwgt_lb' column and 'pregordr' column
totalwgt_lb = preg['totalwgt_lb']
pregordr = preg['pregordr']

# Separating first babies and others
first_babies = totalwgt_lb[pregordr == 1]
others = totalwgt_lb[pregordr > 1]

# Calculating means and standard deviations for both groups
mean_first_babies = first_babies.mean()
mean_others = others.mean()
std_first_babies = first_babies.std()
std_others = others.std()

# Performing independent two-sample t-test
t_statistic, p_value = ttest_ind(first_babies, others, equal_var=False)

# Calculating Cohen's d
cohens_d = (mean_first_babies - mean_others) / (((std_first_babies**2 + st

print("Mean weight of first babies:", mean_first_babies)
print("Mean weight of others:", mean_others)
print("Cohen's d:", cohens_d)
print("p-value:", p_value)
```

```
Mean weight of first babies: 7.204107733975324
Mean weight of others: 7.301399825021872
Cohen's d: -0.06904986139204107
p-value: nan
```

In [57]:
```
# Results of analysis of the Average Pregnancy Length of the first babies

import pandas as pd

# Displaying the first few rows of the dataset
print(preg.head())

# Computing the difference in pregnancy length and birth weight for first
first_babies = preg[preg['pregordr'] == 1]
others = preg[preg['pregordr'] > 1]

avg_pregnancy_length_firstbabies = first_babies['moscurrp'].mean()
avg_pregnancy_length_others = others['moscurrp'].mean()

avg_pregnancy_length_diff = first_babies['moscurrp'].mean() - others['mosc

print("Average Pregnancy Length of firstbabies:", avg_pregnancy_length_fir

print("Average Pregnancy Lengthof others:", avg_pregnancy_length_others)

print("Average Pregnancy Length Difference:", avg_pregnancy_length_diff)
```

```
   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]
Average Pregnancy Length of firstbabies: 4.488636363636363
Average Pregnancy Lengthof others: 4.700757575757576
Average Pregnancy Length Difference: -0.21212121212121282
```

Interpretation

The negative value of Cohen's d suggests that, on average, first babies tend to be slightly lighter than others. However, the effect size, as measured by Cohen's d, is small (-0.069), which indicates that the difference is not very substantial.

The p-value is reported as 'nan' (not a number). This usually happens when there is no variability in one of the groups, leading to a division by zero during the calculation. In this case, it's likely that there is no variability in one of the groups (possibly only one observation), making it impossible to perform a t-test.

Given the small effect size and the issues with the p-value calculation, it's essential to interpret the results cautiously. The small Cohen's d suggests that while there is a statistically significant difference in mean weights between first babies and others, this difference may not be practically significant. Additionally, the p-value issue indicates potential limitations in the dataset, such as insufficient variability in one of the groups.

On another hand, an average pregnancy length difference of approximately -0.21 months between first babies and others was obtained. This indicates that, on average, first pregnancies tend to be slightly shorter than those of subsequent pregnancies. It's important to note that this result is based on the 'moscurrp' variable, representing the current month of pregnancy.

In summary, while statistical differences exist in both birth weights and pregnancy length between first babies and others, the practical significance of these differences is limited.