



Program Code: J620-002-4:2020

Program Name: FRONT-END SOFTWARE DEVELOPMENT

Title : Case Study - Confidence Intervals (Nap no Nap)

Name: Chuay Xiang Ze

IC Number: 021224070255

Date : 14/07/2023

Introduction : Learning about confidence intervals in real data.

Conclusion : Managed to complete tasks about confidence intervals with a nap dataset.

Case Study - Confidence Intervals (Nap no Nap)

In this assessment, you will look at data from a study on toddler sleep habits. The confidence intervals you create and the questions you answer in this Jupyter notebook will be used to answer questions in the cell below.

In [1]:

```
import numpy as np
import pandas as pd
from scipy.stats import t
pd.set_option('display.max_columns', 30) # set so can see all columns of the DataFrame
```

Your goal is to analyse data which is the result of a study that examined differences in a number of sleep variables between napping and non-napping toddlers. Some of these sleep variables included: Bedtime (lights-off time in decimalized time), Night Sleep Onset Time (in decimalized time), Wake Time (sleep end time in decimalized time), Night Sleep Duration (interval between sleep onset and sleep end in minutes), and Total 24-Hour Sleep Duration (in minutes). Note: [Decimalized time](https://en.wikipedia.org/wiki/Decimal_time) (https://en.wikipedia.org/wiki/Decimal_time) is the representation of the time of day using units which are decimally related.

The 20 study participants were healthy, normally developing toddlers with no sleep or behavioral problems. These children were categorized as napping or non-napping based upon parental report of children's habitual sleep patterns. Researchers then verified napping status with data from actigraphy (a non-invasive method of monitoring human rest/activity cycles by wearing of a sensor on the wrist) and sleep diaries during the 5 days before the study assessments were made.

ref: Akacem LD, Simpkin CT, Carskadon MA, Wright KP Jr, Jenni OG, Achermann P, et al. (2015) The Timing of the Circadian Clock and Sleep Differ between Napping and Non-Napping Toddlers. PLoS ONE 10(4): e0125181. <https://doi.org/10.1371/journal.pone.0125181>
(<https://doi.org/10.1371/journal.pone.0125181>).

In [3]:

```
import pandas as pd
import numpy as np
```

In [15]:

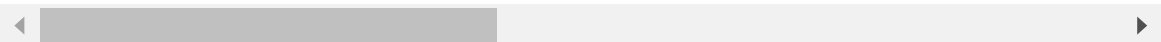
```
# Import the data
df = pd.read_csv("./nap_no_nap.csv")
```

In [16]:

```
# First, Look at the DataFrame to get a sense of the data
df
```

Out[16]:

	id	sex	age (months)	dlmo time	days napped	napping	nap lights outl time	nap sleep onset	nap midsleep	nap sleep offset	nap wake time	du
0	1	female	33.7	19.24	0	0	NaN	NaN	NaN	NaN	NaN	
1	2	female	31.5	18.27	0	0	NaN	NaN	NaN	NaN	NaN	
2	3	male	31.9	19.14	0	0	NaN	NaN	NaN	NaN	NaN	
3	4	female	31.6	19.69	0	0	NaN	NaN	NaN	NaN	NaN	
4	5	female	33.0	19.52	0	0	NaN	NaN	NaN	NaN	NaN	
5	6	female	36.2	18.22	4	1	14.00	14.22	15.00	15.78	16.28	
6	7	male	36.3	19.28	1	1	14.75	15.03	15.92	16.80	16.08	1
7	8	male	30.0	21.06	5	1	13.09	13.43	14.44	15.46	15.82	1
8	9	male	33.2	19.38	2	1	14.41	14.42	15.71	17.01	16.60	1
9	10	female	37.1	19.93	3	1	13.12	13.42	14.31	15.19	15.30	1
10	11	male	32.9	18.79	4	1	13.99	14.03	14.85	15.68	16.10	
11	12	female	35.0	19.65	5	1	13.18	13.45	14.33	15.21	15.35	1
12	13	male	35.1	19.83	3	1	13.94	14.48	15.26	16.03	15.78	
13	14	female	35.6	19.88	4	1	12.68	13.08	13.92	14.76	15.00	1
14	15	female	36.6	19.94	4	1	12.71	12.88	13.80	14.72	14.88	1
15	16	male	36.5	20.25	3	1	13.74	14.68	15.66	16.64	16.45	1
16	17	female	33.7	20.33	5	1	13.15	13.87	14.49	15.11	15.40	
17	18	male	36.4	20.16	5	1	12.47	12.56	13.30	14.05	14.25	
18	19	female	33.6	19.68	3	1	14.71	14.85	15.46	16.07	16.20	
19	20	male	33.8	20.51	3	1	12.68	13.54	14.30	15.07	15.23	



Question: What variable is used in the column 'napping' to indicate a toddler takes a nap?

In [51]:

```
# a number -> 1
df.describe()
```

Out[51]:

	id	age (months)	dlmo time	days napped	napping	nap lights outl time	nap sleep onset	nap midslee
count	20.00000	20.000000	20.000000	20.000000	20.000000	15.000000	15.000000	15.000000
mean	10.50000	34.185000	19.637500	2.700000	0.750000	13.508000	13.862667	14.716667
std	5.91608	2.029331	0.703419	1.894591	0.444262	0.763556	0.745735	0.773150
min	1.00000	30.000000	18.220000	0.000000	0.000000	12.470000	12.560000	13.300000
25%	5.75000	32.975000	19.270000	0.750000	0.750000	12.900000	13.425000	14.305000
50%	10.50000	33.750000	19.685000	3.000000	1.000000	13.180000	13.870000	14.490000
75%	15.25000	36.225000	19.995000	4.000000	1.000000	13.995000	14.450000	15.360000
max	20.00000	37.100000	21.060000	5.000000	1.000000	14.750000	15.030000	15.920000

Question: What is the sample size n ? What is the sample size for toddlers who nap, n_1 , and toddlers who don't nap, n_2 ?

In [11]:

```
n1 = len(df[df['napping'] == 1])
n2 = len(df[df['napping'] == 0])
print(n1)
print(n2)
```

15
5

Average bedtime confidence interval for napping and non napping toddlers

Create two 95% confidence intervals for the average bedtime, one for toddler who nap and one for toddlers who don't.

Before any analysis, we will convert 'night bedtime' into decimalized time.

In [17]:

```
# Convert 'night bedtime' into decimalized time
df.loc[:, 'night bedtime'] = np.floor(df['night bedtime'])*60 + np.round(df['night bedtim
df
```

Out[17]:

	id	sex	age (months)	dlmo time	days napped	napping	nap lights outl time	nap sleep onset	nap midsleep	nap sleep offset	nap wake time	du
0	1	female	33.7	19.24	0	0	NaN	NaN	NaN	NaN	NaN	
1	2	female	31.5	18.27	0	0	NaN	NaN	NaN	NaN	NaN	
2	3	male	31.9	19.14	0	0	NaN	NaN	NaN	NaN	NaN	
3	4	female	31.6	19.69	0	0	NaN	NaN	NaN	NaN	NaN	
4	5	female	33.0	19.52	0	0	NaN	NaN	NaN	NaN	NaN	
5	6	female	36.2	18.22	4	1	14.00	14.22	15.00	15.78	16.28	
6	7	male	36.3	19.28	1	1	14.75	15.03	15.92	16.80	16.08	1
7	8	male	30.0	21.06	5	1	13.09	13.43	14.44	15.46	15.82	1
8	9	male	33.2	19.38	2	1	14.41	14.42	15.71	17.01	16.60	1
9	10	female	37.1	19.93	3	1	13.12	13.42	14.31	15.19	15.30	1
10	11	male	32.9	18.79	4	1	13.99	14.03	14.85	15.68	16.10	
11	12	female	35.0	19.65	5	1	13.18	13.45	14.33	15.21	15.35	1
12	13	male	35.1	19.83	3	1	13.94	14.48	15.26	16.03	15.78	
13	14	female	35.6	19.88	4	1	12.68	13.08	13.92	14.76	15.00	1
14	15	female	36.6	19.94	4	1	12.71	12.88	13.80	14.72	14.88	1
15	16	male	36.5	20.25	3	1	13.74	14.68	15.66	16.64	16.45	1
16	17	female	33.7	20.33	5	1	13.15	13.87	14.49	15.11	15.40	
17	18	male	36.4	20.16	5	1	12.47	12.56	13.30	14.05	14.25	
18	19	female	33.6	19.68	3	1	14.71	14.85	15.46	16.07	16.20	
19	20	male	33.8	20.51	3	1	12.68	13.54	14.30	15.07	15.23	

Now, isolate the column 'night bedtime' for those who nap into a new variable, and those who didn't nap into another new variable.

In [47]:

```
naps = df[df['napping'] == 1]['night bedtime']  
bedtime_nap = naps  
bedtime_nap
```

Out[47]:

```
5      1235.0  
6      1260.0  
7      1321.0  
8      1224.0  
9      1278.0  
10     1185.0  
11     1218.0  
12     1222.0  
13     1226.0  
14     1228.0  
15     1246.0  
16     1243.0  
17     1202.0  
18     1190.0  
19     1218.0  
Name: night bedtime, dtype: float64
```

In [9]:

```
bedtime_nap
```

Out[9]:

```
5      1235.0  
6      1260.0  
7      1321.0  
8      1224.0  
9      1278.0  
10     1185.0  
11     1218.0  
12     1222.0  
13     1226.0  
14     1228.0  
15     1246.0  
16     1243.0  
17     1202.0  
18     1190.0  
19     1218.0  
Name: night bedtime, dtype: float64
```

In [23]:

```
no_naps = df[df['napping'] == 0]
bedtime_no_nap = no_naps['night bedtime']
bedtime_no_nap
```

Out[23]:

```
0    1245.0
1    1163.0
2    1200.0
3    1186.0
4    1161.0
Name: night bedtime, dtype: float64
```

In [11]:

```
bedtime_no_nap
```

Out[11]:

```
0    1245.0
1    1163.0
2    1200.0
3    1186.0
4    1161.0
Name: night bedtime, dtype: float64
```

Now find the sample mean bedtime for nap and no_nap.

In [27]:

```
nap_mean_bedtime = bedtime_nap.sum() / len(bedtime_nap)
nap_mean_bedtime
```

Out[27]:

```
1233.0666666666666
```

In [12]:

```
nap_mean_bedtime =
```

```
1233.0666666666666
```

In [29]:

```
no_nap_mean_bedtime = bedtime_no_nap.sum() / len(bedtime_no_nap)
no_nap_mean_bedtime
```

Out[29]:

```
1191.0
```

In [13]:

```
no_nap_mean_bedtime =
```

```
1191.0
```

Constructing Confidence Intervals

Now that we have the population proportions of male and female smokers, we can begin to calculate confidence intervals. From lecture, we know that the equation is as follows:

$$\text{Best Estimate} \pm \text{Margin of Error}$$

Where the *Best Estimate* is the **observed population proportion or mean** from the sample and the *Margin of Error* is the **t-multiplier**.

The equation to create a 95% confidence interval can also be shown as:

$$\text{Population Proportion or Mean} \pm (t - \text{multiplier} * \text{Standard Error})$$

The Standard Error is calculated differently for population proportion and mean:

$$\text{Standard Error for Population Proportion} = \sqrt{\frac{\text{Population Proportion} * (1 - \text{Population Proportion})}{\text{Number Of Observations}}}$$

$$\text{Standard Error for Mean} = \frac{\text{Standard Deviation}}{\sqrt{\text{Number Of Observations}}}$$

Now find the standard error for \bar{X}_{nap} and $\bar{X}_{no nap}$.

In [56]:

```
import math
import numpy as np

nap_std_err = bedtime_nap.std() / math.sqrt(len(bedtime_nap))
print(nap_std_err)

no_nap_std_err = bedtime_no_nap.std() / math.sqrt(len(bedtime_no_nap))
print(no_nap_std_err)
```

```
8.893800230479801
15.339491516996253
```

In [14]:

Out[14]:

```
8.893800230479801
```

In [15]:

Out[15]:

```
15.339491516996253
```


Question: Given our sample sizes of n_1 and n_2 for napping and non napping toddlers respectively, how many degrees of freedom (df) are there for the associated t distributions?

In [57]:

```
n1 = len(bedtime_nap) - 1
n2 = len(bedtime_no_nap) - 1
print(n1, n2)
```

14 4

To build a 95% confidence interval, what is the value of t^* ? You can find this value using the percent point function:

```
from scipy.stats import t
```

```
t.ppf(probability, df)
```

This will return the quantile value such that to the left of this value, the tail probability is equal to the input probability (for the specified degrees of freedom).

Example: to find the t^* for a 90% confidence interval, we want t^* such that 90% of the density of the t distribution lies between $-t^*$ and t^* . Or in other words if $X \sim t(df)$:

$$P(-t^* < X < t^*) = .90$$

Which, because the t distribution is symmetric, is equivalent to finding t^* such that:

$$P(X < t^*) = .95$$

So the t^* for a 90% confidence interval, and lets say $df=10$, will be:

```
t_star = t.ppf(.95, df=10)
```

Question: What is t^* for nap and no nap?

In [16]:

```
from scipy.stats import t
```

In [59]:

```
nap_t_star = t.ppf(.975, df = 14)
nap_t_star
```

Out[59]:

2.1447866879169273

In [17]:

```
# Find the t_stars for the 95% confidence intervals
nap_t_star =
```

Out[17]:

2.1447866879169273

In [60]:

```
no_nap_t_star = t.ppf(.975, df = 4)
no_nap_t_star
```

Out[60]:

2.7764451051977987

In [18]:

```
no_nap_t_star =
```

Out[18]:

2.7764451051977987

Now to create our confidence intervals. For the average bedtime for nap and no nap, find the upper and lower bounds for the respective confidence intervals.

Question: What are the 95% confidence intervals, rounded to the nearest ten, for the average bedtime (in decimalized time) for toddlers who nap and for toddlers who don't nap?

$$CI = \bar{X} \pm t^* \cdot s.e.(\bar{X})$$

In [127]:

```
import numpy as np
from scipy.stats import norm
CI = nap_mean_bedtime + np.array([-1, 1]) * nap_t_star * nap_std_err
tuple(CI)
```

Out[127]:

(1213.991362327341, 1252.1419710059922)

In [108]:

```
#1214, 1252
```

In [126]:

```
CI = no_nap_mean_bedtime + np.array([-1, 1]) * no_nap_t_star * no_nap_std_err
tuple(CI)
```

Out[126]:

(1148.4107438614126, 1233.5892561385874)

In [20]:

```
#1148, 1234
```

Out[20]:

```
(1148.4107438614126, 1233.5892561385874)
```

Challenge problem 1: Write a function that inputs the column containing the data you want to build your confidence interval from and returns the confidence interval as a list or tuple (i.e. [upper, lower] or (upper, lower)).

In [133]:

```
import numpy as np
from scipy.stats import t
def find_ci(data_column, confidence):
    sample_mean = np.mean(data_column)
    sample_std = np.std(data_column)
    n = len(data_column)
    t_value = t.ppf((1 + confidence) / 2, df = n - 1)
    margin_of_error = t_value * sample_std / np.sqrt(n)
    lower_bound = sample_mean - margin_of_error
    upper_bound = sample_mean + margin_of_error
    return (lower_bound, upper_bound)
```

Out[133]:

```
(1214.6381718847074, 1251.4951614486258)
```

Challenge problem 2: Create the intervals using the statsmodels function stats.weightstats.DescrStatsW:

```
stats.weightstats.DescrStatsW(data).tconfint_mean(alpha=.05)
```

In [138]:

```
from statsmodels.stats.weightstats import DescrStatsW
stats_nap = DescrStatsW(bedtime_nap).tconfint_mean(alpha=0.05)
stats_no_nap = DescrStatsW(bedtime_no_nap).tconfint_mean(alpha=0.05)
print(stats_nap)
print(stats_no_nap)
```

```
(1213.991362327341, 1252.1419710059922)
(1148.4107438614126, 1233.5892561385874)
```

In [28]:

Out[28]:

```
(1213.991362327341, 1252.1419710059922)
```

In [29]:

Out[29]:

(1148.4107438614126, 1233.5892561385874)