Forward School

## Program Code: J620-002-4:2020

## Program Name: FRONT-END SOFTWARE DEVELOPMENT

## Title : Hypothesis Testing Exercise In Python

**Name: Phua Yan Han**

**IC Number: 050824070059**

**Date : 11/7/23**

**Introduction :**

**Conclusion :**

## Creating confidence intervals in python

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 following graded assignment.

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 D
```

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) 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.

You are specifically interested in the results for the Bedtime, Night Sleep Duration, and Total 24-Hour Sleep Duration.

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 [2]:
```
# Import the data
df = pd.read_csv("../Data files/nap_no_nap.csv")
```

In [3]:
```
# First, look at the DataFrame to get a sense of the data
df
```

Out[3]:

| | id | sex | age (months) | dlmo time | days napped | napping | nap lights outl time | nap sleep onset | nap midsleep | nap sleep offset | nap wake time | na duratio |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | female | 33.7 | 19.24 | 0 | 0 | NaN | NaN | NaN | NaN | NaN | Na |
| **1** | 2 | female | 31.5 | 18.27 | 0 | 0 | NaN | NaN | NaN | NaN | NaN | Na |
| **2** | 3 | male | 31.9 | 19.14 | 0 | 0 | NaN | NaN | NaN | NaN | NaN | Na |
| **3** | 4 | female | 31.6 | 19.69 | 0 | 0 | NaN | NaN | NaN | NaN | NaN | Na |
| **4** | 5 | female | 33.0 | 19.52 | 0 | 0 | NaN | NaN | NaN | NaN | NaN | Na |
| **5** | 6 | female | 36.2 | 18.22 | 4 | 1 | 14.00 | 14.22 | 15.00 | 15.78 | 16.28 | 93.7 |
| **6** | 7 | male | 36.3 | 19.28 | 1 | 1 | 14.75 | 15.03 | 15.92 | 16.80 | 16.08 | 106.0 |
| **7** | 8 | male | 30.0 | 21.06 | 5 | 1 | 13.09 | 13.43 | 14.44 | 15.46 | 15.82 | 121.6 |
| **8** | 9 | male | 33.2 | 19.38 | 2 | 1 | 14.41 | 14.42 | 15.71 | 17.01 | 16.60 | 155.5 |
| **9** | 10 | female | 37.1 | 19.93 | 3 | 1 | 13.12 | 13.42 | 14.31 | 15.19 | 15.30 | 106.6 |
| **10** | 11 | male | 32.9 | 18.79 | 4 | 1 | 13.99 | 14.03 | 14.85 | 15.68 | 16.10 | 98.7 |
| **11** | 12 | female | 35.0 | 19.65 | 5 | 1 | 13.18 | 13.45 | 14.33 | 15.21 | 15.35 | 105.8 |
| **12** | 13 | male | 35.1 | 19.83 | 3 | 1 | 13.94 | 14.48 | 15.26 | 16.03 | 15.78 | 93.3 |
| **13** | 14 | female | 35.6 | 19.88 | 4 | 1 | 12.68 | 13.08 | 13.92 | 14.76 | 15.00 | 100.7 |
| **14** | 15 | female | 36.6 | 19.94 | 4 | 1 | 12.71 | 12.88 | 13.80 | 14.72 | 14.88 | 110.7 |
| **15** | 16 | male | 36.5 | 20.25 | 3 | 1 | 13.74 | 14.68 | 15.66 | 16.64 | 16.45 | 117.3 |
| **16** | 17 | female | 33.7 | 20.33 | 5 | 1 | 13.15 | 13.87 | 14.49 | 15.11 | 15.40 | 74.2 |
| **17** | 18 | male | 36.4 | 20.16 | 5 | 1 | 12.47 | 12.56 | 13.30 | 14.05 | 14.25 | 89.8 |
| **18** | 19 | female | 33.6 | 19.68 | 3 | 1 | 14.71 | 14.85 | 15.46 | 16.07 | 16.20 | 73.0 |
| **19** | 20 | male | 33.8 | 20.51 | 3 | 1 | 12.68 | 13.54 | 14.30 | 15.07 | 15.23 | 91.6 |

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

In [4]:
```
df['napping'].describe()
```

Out[4]:
```
count    20.000000
mean      0.750000
std       0.444262
min       0.000000
25%       0.750000
50%       1.000000
75%       1.000000
max       1.000000
Name: napping, dtype: float64
```

**Question**: What is the sample size $n$?

In [5]:
```
n=len(df['napping'])
```

# Hypothesis tesing

We will look at two hypothesis test, each with $\alpha = .025$:

1. Is the average bedtime for toddlers who nap later than the average bedtime for toddlers who don't nap?

   $$H_0 : \mu_{nap} = \mu_{no\ nap},\ H_a : \mu_{nap} > \mu_{no\ nap}$$

   Or equivalently:

   $$H_0 : \mu_{nap} - \mu_{no\ nap} = 0,\ H_a : \mu_{nap} - \mu_{no\ nap} > 0$$

2. The average 24 h sleep duration (in minutes) for napping toddlers is different from toddlers who don't nap.

   $$H_0 : \mu_{nap} = \mu_{no\ nap},\ H_a : \mu_{nap} \neq \mu_{no\ nap}$$

   Or equivalenty:

   $$H_0 : \mu_{nap} - \mu_{no\ nap} = 0,\ H_a : \mu_{nap} - \mu_{no\ nap} \neq 0$$

Aside: This $\alpha$ level is equivalent to $\alpha = .05$ and then applying the [Bonferonni correction (https://en.wikipedia.org/wiki/Bonferroni_correction)](https://en.wikipedia.org/wiki/Bonferroni_correction).

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

In [6]:
```
# Convert 'night bedtime' into decimalized time
df.loc[:,'night bedtime'] = np.floor(df['night bedtime'])*60 + np.round(df['ni
```

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 [7]:
```
nap_bedtime = df[df['napping'] ==1]['night bedtime']
nap_bedtime
```

Out[7]:
```
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 [8]:
```
no_nap_bedtime = df[df['napping'] ==0]['night bedtime']
no_nap_bedtime
```

Out[8]:
```
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 [9]:
```
nap_mean_bedtime = df[df['napping'] ==1]['night bedtime'].mean()
nap_mean_bedtime
```

Out[9]:
```
1233.0666666666666
```

In [10]:
```
no_nap_mean_bedtime = df[df['napping'] ==0]['night bedtime'].mean()
no_nap_mean_bedtime
```

Out[10]:
```
1191.0
```

**Question**: What is the sample difference of mean bedtime for nappers minus no nappers?

Now find the sample standard deviation for $X_{nap}$ and $X_{no\ nap}$.

In [11]:
```
import statistics
nap_s_bedtime = statistics.stdev(nap_bedtime)
nap_s_bedtime
```

Out[11]:
```
34.445540177143954
```

In [12]:
```
no_nap_s_bedtime =  statistics.stdev(no_nap_bedtime)
no_nap_s_bedtime
```

Out[12]:
```
34.30014577228499
```

| | Nap Bedtime | No Nap Bedtime |
|---|---|---|
| Mean | 1233.07 | 1191 |
| Std Dev, s | 34.35 | 34.30 |
| Sample Size (n) | 15 | 5 |

We expect the variance in sleep time for toddlers who nap and toddlers who don't nap to be the same. Calculate the pooled standard error of $\bar{X}_{nap} - \bar{X}_{no\ nap}$.

$$s.e.(\bar{X}_{nap} - \bar{X}_{no\ nap}) = S_P\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}$$

In [13]:
```
##pooled SE
import math
n1 = len(df[df['napping'] == 1])
se_nap = nap_bedtime.std() / math.sqrt(n1)
se_nap
n2 = len(df[df['napping'] == 0])
se_no_nap = no_nap_bedtime.std() / math.sqrt(n2)
se_no_nap
sp = math.sqrt((((nap_s_bedtime**2) * (n1 - 1)) + ((no_nap_s_bedtime**2) * (n2
pooled_se = sp * math.sqrt((1 / n1) + (1 / n2))
pooled_se
```

Out[13]: 17.77094313065816

**Question**: What is the pooled s.e.$(\bar{X}_{nap} - \bar{X}_{no\ nap})$?

In [14]:
```
import math
n1 = len(df[df['napping'] == 1])
se_nap = nap_bedtime.std() / math.sqrt(n1)
se_nap
n2 = len(df[df['napping'] == 0])
se_no_nap = no_nap_bedtime.std() / math.sqrt(n2)
se_no_nap
sp = math.sqrt((((nap_s_bedtime**2) * (n1 - 1)) + ((no_nap_s_bedtime**2) * (n2
pooled_se = sp * math.sqrt((1 / n1) + (1 / n2))
pooled_se
```

Out[14]: 17.77094313065816

**Question**: Given our sample size of $n$, how many degrees of freedom $(df)$ are there for the associated $t$ distribution?

In [24]:
```
n=len(df)-1
n
```

Out[24]: 19

Now find the $t$-test statistic for our hypothesis test using

- pooled s.e.$(\bar{X}_{nap} - \bar{X}_{no\ nap})$
- $\bar{X}_{nap} - \bar{X}_{no\ nap}$
- $\mu_{0,\ nap} - \mu_{0,\ no\ nap}$, the population difference in means under the null hypothesis

**Question**: What is the $t$-test statistic for the hypothesis test?

To find the p-value, we can use the function:

```
t.cdf(y, df)
```

Which for $X \sim t(df)$ returns $P(X \leq y)$.

Because of the symmetry of the $t$ distrubution, we have that

```
1-t.cdf(y, df)
```

returns $P(X > y)$

This function, t.cdf(y, df), will give you the same value as finding the one-tailed probability of y on a t-table with the specified degrees of freedom.

Using the function t.cdf and your $t$-test statistics, find the p-value.

In [25]:
```
from scipy.stats import t
t_statistics = ((nap_mean_bedtime- no_nap_mean_bedtime) / pooled_se)
p_value = 1 - t.cdf(t_statistics, n-1)
print("t-statistics:", t_statistics)
print("p-value:", p_value)
```

```
t-statistics: 2.367160052079275
p-value: 0.014667451430902756
```

**Question**: What is the p-value to the nearest hundredth?

In [26]:
```
p_value*100
```

Out[26]:
```
1.4667451430902756
```

Calculate the $t$ test statistics and corresponding p-value using the scipy function scipy.stats.ttest_ind(a, b, equal_var=True) and check with your answer.

In [27]:
```
from scipy.stats import ttest_ind
t_statistics2, p_value2 = ttest_ind(nap_bedtime, no_nap_bedtime, equal_var=Tru
print("t-statistics:", t_statistics2)
print("p-value:", p_value2)
# This is for the alternative method using ttest_ind()
```

```
t-statistics: 2.367160052079275
p-value: 0.029334902861805394
```

Does scipy.stats.ttest_ind return values for a one-sided or two-sided test? Can you think of a way to recover the results you got using 1-t.cdf from the p-value given by scipy.stats.ttest_ind?

In [28]:
```
##two-sided
p_value*2
```

Out[28]:
```
0.029334902861805512
```

**Question**: Do you reject or fail to reject the null hypothesis that the difference in average bedtimes for napping versus non napping toddlers is 0?

Using the scipy.stats.ttest_ind function, find the $t$-test statistic and p-value for the second hypothesis test:
2. The average total 24 h sleep duration (in minutes) for napping toddlers is different from toddlers who don't nap.

$$H_0 : \mu_{nap} = \mu_{no\ nap},\ H_a : \mu_{nap} \neq \mu_{no\ nap}$$

Or equivalenty:

$$H_0 : \mu_{nap} - \mu_{no\ nap} = 0,\ H_a : \mu_{nap} - \mu_{no\ nap} \neq 0$$

**Question**: What is the $t$-test statistic and p-value?

In [20]:
```
from scipy.stats import ttest_ind
nonap24 = df[df['napping']==0]['24 h sleep duration']
nap24 = df[df['napping']==1]['24 h sleep duration']

ttest_ind(nap24,nonap24)
```

Out[20]:
```
Ttest_indResult(statistic=1.4811248223284985, pvalue=0.1558664953018476)
```

**Question**: For $\alpha = .025$, do you reject or fail to reject the null hypothesis?

In [29]:
```
print("we failed to reject the null hypothesis since our value for p value is (
```

```
we failed to reject the null hypothesis since our value for p value is 0.0293
34902861805394 therefore it is larger then alpha so we do not have sufficient
evidence to reject
```