# Homework 08: Hypothesis Testing, P-values, Bootstrapping

---

**Name**: CJ Kennedy

---

This assignment is due on Canvas by **6:00PM on Friday November 4**. Your solutions to theoretical questions should be done in Markdown directly below the associated question. Your solutions to computational questions should include any specified Python code and results as well as written commentary on your conclusions. Remember that you are encouraged to discuss the problems with your classmates, but **you must write all code and solutions on your own**.

**NOTES**:

- Any relevant data sets should be available in the Homework 01 assignment write-up on Canvas. To make life easier on the grader if they need to run your code, do not change the relative path names here. Instead, move the files around on your computer.
- If you're not familiar with typesetting math directly into Markdown then by all means, do your work on paper first and then typeset it later. Remember that there is a reference guide linked on Canvas on writing math in Markdown. **All** of your written commentary, justifications and mathematical work should be in Markdown.
- Because you can technically evaluate notebook cells is a non-linear order, it's a good idea to do $\text{Kernel} \rightarrow \text{Restart \& Run All}$ as a check before submitting your solutions. That way if we need to run your code you will know that it will work as expected.
- It is **bad form** to make your reader interpret numerical output from your code. If a question asks you to compute some value from the data you should show your code output **AND** $\text{write a summary of the results}$ in Markdown directly below your code.
- This probably goes without saying, but... For any question that asks you to calculate something, you **must show all work and justify your answers to receive credit**. Sparse or nonexistent work will receive sparse or nonexistent credit.

---

The standard imports for this HW:

In [1]:
```
# Per the standard import pandas as 'pd' and numpy as 'np'
import pandas as pd
import numpy as np
import matplotlib.pylab as plt
%matplotlib inline
import scipy.stats as stats
from scipy.stats import norm
```

# Problem 1

In this HW you will need to use `.std()` when you are finding the test statistic. However, there are two kinds of standard deviations: those for a sample and those for a population.

Consider the python list below:

In [2]:
```
py_list = [4,2,3,4,2,3]
```

***(3 points)*** Find both the sample standard deviation and the popluation standard deviation by hand.

TeX your work below:

Solution:

First, the mean is 3 and the sum of each element minus the mean squared is: 1+1+0+1+1+0=4

**Sample Standard Deviation**:

$$\sigma = \sqrt{\frac{\sum_i^n (x_i - \bar{x})^2}{n-1}}$$

$$= \sqrt{\frac{\sum_i^n (x_i - \bar{x})^2}{5}}$$

$$= \sqrt{\frac{4}{5}}$$

$$= .894$$

**Population Standard Deviation**:

$$\sigma = \sqrt{\frac{\sum_i^n (x_i - \bar{x})^2}{n}}$$

$$= \sqrt{\frac{4}{6}}$$

$$= .8165$$

***(2 points)*** Make an array from `py_list` and call it `np_array`.

In [3]:
```
# Code your answer here:
np_array = np.array(py_list)
print(np_array)
```

```
[4 2 3 4 2 3]
```

***(2 points)*** Make a Pandas Series from the list and call it `dfSeries`.

In [4]:
```
# Code your answer here:
dfSeries = pd.Series(py_list)
print(dfSeries)
```

```
0    4
1    2
2    3
3    4
4    2
5    3
dtype: int64
```

***(2 points)*** Find `np_array.std()` and `dfSeries.std()`.

What type of standard deviation does `np_array.std()` return?

What type of standard deviation does `dfSeries.std()` return?

In [5]:
```
# Code your solution here:
np_std = np_array.std()
df_std = dfSeries.std()
print("Numpy STD:",np_std)
print("DF STD:",df_std)
print("The panda series returns the sample standard deviation.")
print("The numpy list returns the population standard deviation.")
```

```
Numpy STD: 0.816496580927726
DF STD: 0.8944271909999159
The panda series returns the sample standard deviation.
The numpy list returns the population standard deviation.
```

***(2 points)*** Now find `np_array.std(ddof=0)` and `dfSeries.std(ddof=0)`, and `np_array.std(ddof=1)` and `dfSeries.std(ddof=1)`.

What do these return?

In [6]:
```
# Code your answer here:
np_std_ddof0 = np_array.std(ddof=0)
np_std_ddof1 = np_array.std(ddof=1)
df_std_ddof0 = dfSeries.std(ddof=0)
df_std_ddof1 = dfSeries.std(ddof=1)
print("Numpy STD with ddof = 0:",np_std_ddof0)
print("DF STD with ddof = 0:",df_std_ddof0)
print("Numpy STD with ddof = 1:",np_std_ddof1)
print("DF STD with ddof = 1:",df_std_ddof1)
print("Both the series and list (ddof=0) returns the sample standard deviation.")
print("Both the series and list (ddof=1) returns the population standard deviation.")
```

```
Numpy STD with ddof = 0: 0.816496580927726
DF STD with ddof = 0: 0.816496580927726
Numpy STD with ddof = 1: 0.8944271909999159
DF STD with ddof = 1: 0.8944271909999159
Both the series and list (ddof=0) returns the sample standard deviation.
Both the series and list (ddof=1) returns the population standard deviation.
```

# Problem 2

A nematologist is interested in determining whether a new worm food (wood bark treated with peanut butter) results in shorter worm length than the standard length of 15.7 cm.

Shorter worms are more desirable as they tend to be stronger and live longer.

The nematologist feeds a random sample of worms with the new food and subsequently obtained the worm lengths found in the `csv` file `worm.csv`.

If the nematologist has in fact discovered a healthy new worm food then this food formula can be patented and sold world wide!

Therefore, the nematologist has hired you to explain whether or not this new food outperforms (with respect to worm length) the old worm food.

***(read-in)*** Read in the csv file here:

In [7]:
```
# read in worm.csv
dfWorm = pd.read_csv("worm.csv")
```

***(1 point)*** Take a look at the first 5 rows of data.

In [8]:
```
# code here for looking at data:
dfWorm.head()
```

Out[8]:

| | length |
|---|---|
| **0** | 11.5 |
| **1** | 15.2 |
| **2** | 16.5 |
| **3** | 15.1 |
| **4** | 11.8 |

In order to determine whether or not this new worm food outperforms the standard food, you and the nematologist decide on a hypthesis test run at the 5% significance level.

***(2 points)*** What does a 5% significance level mean?

Solution:

The significance level determines the largest probability for the test statistic that would have us reject the null hypothesis.

A 5% significance level implies we can say with certainty whether we should reject the null hypothesis with a confidence interal of 95%. The probability of making a Type I error is 5%

***(2 points)*** What is the null hypothesis and alternate hypothesis for this test?

Solution:

The null hypothesis for the test is $H_0 : \mu = 15.7cm$ remains after the food change.

The alternative hypothesis is then $H_1 : \mu < 15.7cm$ reduction after the food change. Significant weight loss occurs ($\alpha = .05$)

***(1 point)*** How many worms were in this sample?

In [9]:
```
# code your answer here:
n = dfWorm.count()
n = n.length
print("There are",np.round(n,2),"worms in this sample")
```

```
There are 33 worms in this sample
```

***(1 point)*** What is the mean of the sample?

In [10]:
```
# Code your solution here:
X = dfWorm.mean()
X = X.length
print("The mean of the sample is",np.round(X,2))

#np_array = dfWorm.to_numpy()
#print(np_array.mean())
```

```
The mean of the sample is 13.66
```

***(1 point)*** What is the standard deviation of the sample?

In [11]:
```
# Code your answer here:
s = dfWorm.std(ddof=0)
s = s.length
print("The standard deviation of the sample is:",np.round(s,3))
```

```
The standard deviation of the sample is: 2.505
```

***(2 points)*** What is the critical value?

In [79]:
```
# Code your answer here:
# we are at a 5% significance critical value
alpha = .05
#p = norm.cdf(15.7,loc=X,scale=s)
#print("P( X \u2264 15.7 | H_0 is true) = {:.3f}".format(p))
z = norm.ppf(alpha) # ppf with custom mu,sigma
print("The critical value z = {:.2f}".format(z))
print("The value to be tested is -z")
```

```
The critical value z = -1.64
The value to be tested is -z
```

***(2 points)*** What is the test statistic?

In [55]:
```
# Code your answer here:
mu_0 = 15.7
Zc = (X-mu_0)/(s/np.sqrt(n))
print("The test statistic for the sample mean is:",np.round(Zc,2))
```

```
The test statistic for the sample mean is: -4.67
```

***(3 points)*** What is the conclusion?

Solution:

If P( X ≤ Zc ) < 5%, then the p-value argument would reject the null hypothesis.

If the Z-value for the sample mean is > the Z-value for the critical value, the critical value argument would reject the null hypothesis.

In other terms if our alt. hypothesis is $H_1 : \theta < \theta_0$ then the rejection region for level $\alpha$ test is the following:

$z \leq -z_{alpha} \Rightarrow -4.67 \leq -1.65$

The above statement on the right is TRUE.

Our test statistic, -4.67, is less than the negative critical value, -1.64. Thus, we do reject the null hypothesis.

Specifically, we conclude there is sufficient evidence to believe, at a 5% significance level, the mean length of worms fed with new food are shorter.

Instead of using a critical value to determine the above answer about the worm food, suppose you decide to base your decision on the p-value for this same data.

***(2 points)*** In general, what is it that a p-value measures?

Solution:

Assuming the null hypothesis is correct, the p-value measures the probability of obtaining results at least as extreme as the observed results of a statistical hypothesis test.

***(2 points)*** What is the p-value for this experiment?

In [80]:
```
# Code your answer here:
mu_0 = 15.7
Zc = (X-mu_0)/(s/np.sqrt(n))
p = norm.cdf(Zc,loc=X,scale=s)
print("p = P( X \u2264 -4.67) = {:.5f}".format(p*100),"%")
```

```
p = P( X ≤ -4.67) = 0.00000 %
```

***(2 points)*** According to the p-value, should we reject the null or fail to reject the null?

Solution:

The p-value is less than 5%, so we would reject the null and accept the alternative.

***(2 points)*** Will the decisions concerning rejecting the null ever be different with respect to using a critical number versus a p-value?

Solution:

Both methods for rejecting the null result in the same answer. So, the decisions will never be different, only the process.

***(2 points)*** According to the acquired p-value what is the largest confidence interval we could have used to reject the null hypothesis?

In [84]:
```
# Code you solution here:
CI = 100-p*100
print("The largest CI we could use to reject the null hypothesis is:",np.round(CI,2),'
```

```
The largest CI we could use to reject the null hypothesis is: 100.0 %
```

# Problem 3

---

Widg's are fairly rare and difficult to come by.

In order to determine the density of a widg, one must destroy them with a crushing mechanism. A sample of of $n = 200$ widg densities has been determined.

From this sample, we would like to determine the probable density of other widgs' in the population. Of course we don't want to crush anymore widgs and they are hard to come by, so we will have to make due with this one sample.

The csv file `strap.csv` is the sample ($n = 200$) obtained from a distribution of widg densities.

***(read-in)*** load the csv into a dataframe called dfWidg.

In [16]:
```
# Code your work here:
dfWidg = pd.read_csv("strap.csv")
dfWidg
```

Out[16]:

| | widg |
|---|---|
| **0** | 2 |
| **1** | 3 |
| **2** | 3 |
| **3** | 1 |
| **4** | 1 |
| **...** | ... |
| **195** | 4 |
| **196** | 2 |
| **197** | 3 |
| **198** | 2 |
| **199** | 2 |

200 rows × 1 columns

***(4 points)*** Write a function to draw $10000$ bootstrapped resamples (with replacement) from this sample of 200 widg densities and compute a bootstrapped confidence interval for the mean at the 90% confidence level.

In [17]:
```
# Code your answer here:
# adapted from lecture 15 on bootsrapping
def bootstrap():
    n = 200
    # create an empty array
    medians = []
    # resample
    for i in range(0,10000):
        resample = np.random.choice(dfWidg["widg"],replace=True)
        med_resample = np.median(resample)
        medians.append(med_resample)
    CI = np.percentile(medians, [5, 95]) # 90% confidence level
    return CI,medians;

#bootstrap();
```

***(2 points)*** What is the meaning of this 90% CI?

Solution:

We can be 90% confident that the population mean of the density of widges falls within the range of 1 to 5.

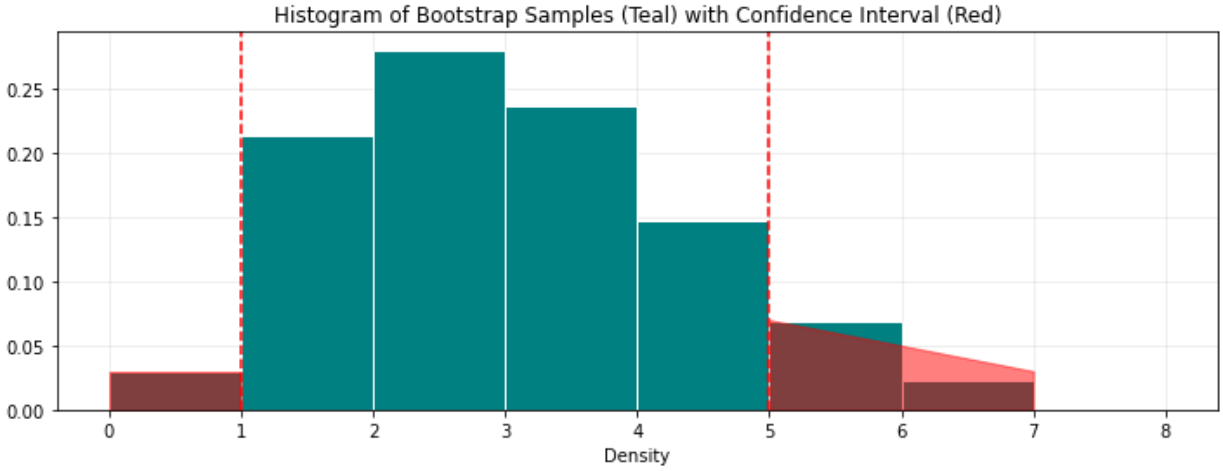
***(4 points)*** write a function that will graph a histogram of our 10000 bootstrap samples with

the confidence interval superimposed on the histogram.

BTW, choose your own colors: https://datascientyst.com/full-list-named-colors-pandas-python-matplotlib/

In [41]:
```
# Code your answer here:
# adaped slightly from nb17
def graph():
    CI,x = bootstrap();
    # Initialize the plot
    fig, ax = plt.subplots(nrows=1, ncols=1, figsize=(12,4))
    # create your bins of "width alpha"
    bins = np.linspace(0, 8, 9)
    # Create a histogram of 'x'.
    pd.Series(x).hist(ax=ax, color="teal", edgecolor="white", bins=bins, density=True)
    ax.grid(alpha=0.25)
    ax.set_axisbelow(True)
    # CI superimpose
    ax.axvline(CI[0], color='r', linestyle='--');
    ax.axvline(CI[1], color='r', linestyle='--');
    ax.fill_between([0, 1],[.03, .03],color='red',alpha=.5)
    ax.fill_between([5, 7],[.07, .03],color='red',alpha=.5)
    ax.set_title("Histogram of Bootstrap Samples (Teal) with Confidence Interval (Red)")
    ax.set_xlabel("Density")
    return fig,ax
graph()
```

Out[41]:
```
(<Figure size 864x288 with 1 Axes>,
 <AxesSubplot:title={'center':'Histogram of Bootstrap Samples (Teal) with Confidence
Interval (Red)'}, xlabel='Density'>)
```




***(2 points)*** What is the mean of the $n = 200$ data, and is this mean found inside the bootstrap CI?

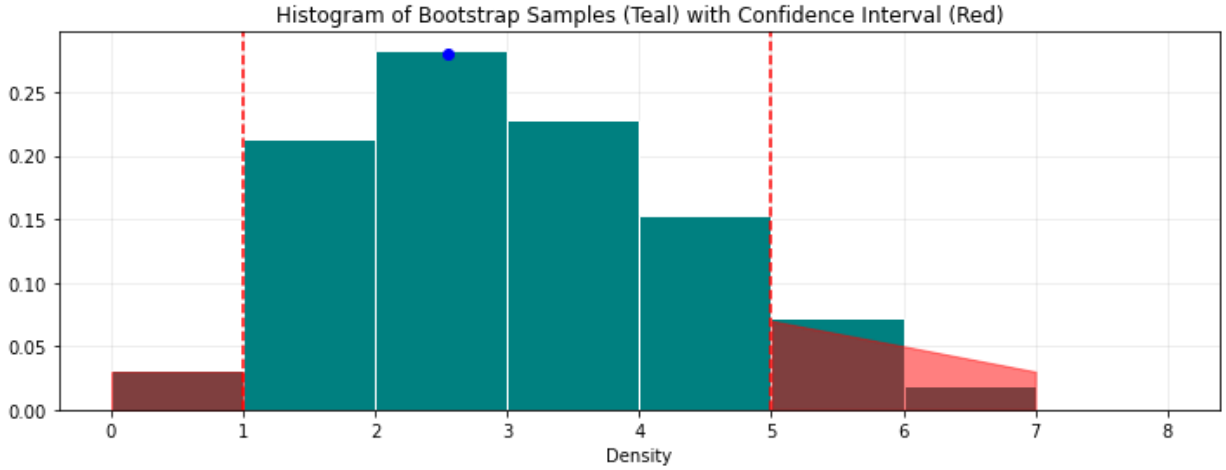
In [19]:
```
# Code your solution here:
mu_sample = dfWidg["widg"].mean()
print("The sample mean is:",mu_sample)
print("This sample mean is in the boostrap CI")
```

```
The sample mean is: 2.545
This sample mean is in the boostrap CI
```

***(2 points)*** Graph a histogram of our 10000 bootstrap samples with the confidence interval

superimposed on the histogram, AND the sample mean as a dot on the CI. BTW, the actual population mean is 2.5. You likely arrived at a mean that is very close to this.

In [69]:
```
# Code your answer here:
fig,ax = graph()
ax.scatter(mu_sample,.28,color='b');
```




In [ ]: