Confidence Intervals

This tutorial is going to demonstrate how to load data, clean/manipulate a dataset, and construct a confidence interval for the difference between two population proportions and means.

We will use the 2015-2016 wave of the NHANES data for our analysis.

*Note: We have provided a notebook that includes more analysis, with examples of confidence intervals for one population proportions and means, in addition to the analysis I will show you in this tutorial. I highly recommend checking it out!

For our population proportions, we will analyze the difference of proportion between female and male smokers. The column that specifies smoker and non-smoker is "SMQ020" in our dataset.

For our population means, we will analyze the difference of mean of body mass index within our female and male populations. The column that includes the body mass index value is "BMXBMI".

Additionally, the gender is specified in the column "RIAGENDR".

In [ ]:

```
import pandas as pd
import numpy as np
import matplotlib
matplotlib.use('Agg')
import seaborn as sns
%matplotlib inline
import matplotlib.pyplot as plt
```

In [ ]:

```
url = "nhanes_2015_2016.csv"
```

Investigating and Cleaning Data

In [ ]:

```
# Recode SMQ020 from 1/2 to Yes/No into new variable SMQ020x
da["SMQ020x"] = da.SMQ020.replace({1: "Yes", 2: "No", 7: np.nan, 9: np.nan})
```

In [ ]:

```
# Recode RIAGENDR from 1/2 to Male/Female into new variable RIAGENDRx
da["RIAGENDRx"] = da.RIAGENDR.replace({1: "Male", 2: "Female"})
```

In [ ]:

```
dx = da[["SMQ020x", "RIAGENDRx"]].dropna()
```

In [ ]:

```
# Recode SMQ020x from Yes/No to 1/0 into existing variable SMQ020x
```

In [ ]:

```
dz = dx.groupby("RIAGENDRx").agg({"SMQ020x": [np.mean, np.size]})
dz.columns = ["Proportion", "Total n"]
dz
```

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:

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

Best Estimate±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:

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

Population Proportion or Mean ±(t−multiplier∗ Standard Error)

The Standard Error (SE) is calculated differenly for population proportion and mean:

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

Standard Error for Population Proportion=Population Proportion∗(1−Population Proportion)Number Of Observatio

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

Standard Error for Mean=Standard DeviationNumber Of Observations

Lastly, the standard error for difference of population proportions and means is:

$$\text{Standard Error for Difference of Two Population Proportions Or Means} = \sqrt{(\text{SE}_1)^2 + (\text{SE}_2)^2}$$

Standard Error for Difference of Two Population Proportions Or Means=(SE 1)2+(SE 2)2

Difference of Two Population Proportions

In [ ]:

```
p = .304845
n = 2972
se_female = np.sqrt(p * (1 - p)/n)
se_female
```

In [ ]:

```
p = .513258
n = 2753
se_male = np.sqrt(p * (1 - p)/ n)
se_male
```

In [ ]:

```
se_diff = np.sqrt(se_female**2 + se_male**2)
se_diff
```

In [ ]:

```
d = .304845 - .513258
lcb = d - 1.96 * se_diff
ucb = d + 1.96 * se_diff
(lcb, ucb)
```

Difference of Two Population Means

In [ ]:

```
da["BMXBMI"].head()
```

In [ ]:

```
da.groupby("RIAGENDRx").agg({"BMXBMI": [np.mean, np.std, np.size]})
```

In [ ]:

```
sem_female = 7.753319 / np.sqrt(2976)
sem_male = 6.252568 / np.sqrt(2759)
(sem_female, sem_male)
```

In [ ]:

```
sem_diff = np.sqrt(sem_female**2 + sem_male**2)
sem_diff
```

In [ ]:

```
d = 29.939946 - 28.778072
```

In [ ]:

```
lcb = d - 1.96 * sem_diff
ucb = d + 1.96 * sem_diff
(lcb, ucb)
```