Introduction to Statistical Inference (QTM 100 Lab)

Lecture 5: Sampling Distributions

Justin Eloriaga — Emory University

Fall 2024

Gameplan

Data Preliminaries

Sampling Distribution of Proportions

The `for` loop

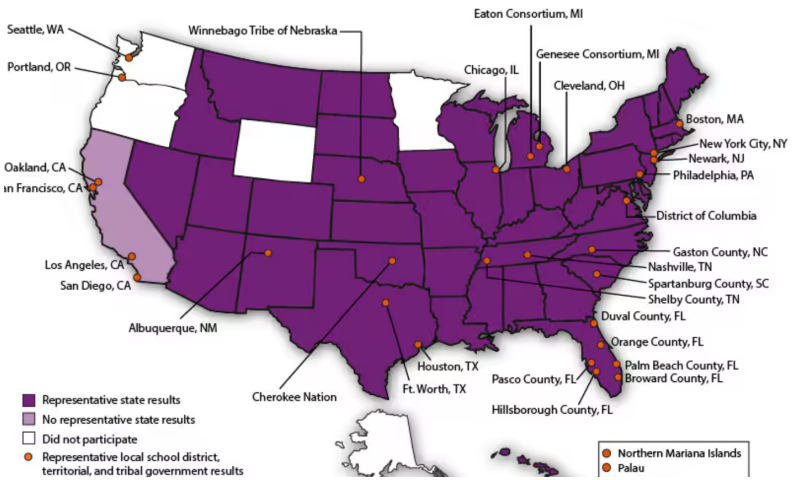
Summarizing Sampling Distribution of Proportions

Data Preliminaries

On Sampling Distributions

- Oftentimes, we are interested in formulating a *sampling distribution* of our estimate in order to learn about the properties of the estimate, such as its distribution.

Youth Risk Behavior and Surveillance System



- CDC will conduct a survey in schools to monitor the largest contributors to youth morbidity and mortality
- They measure things like health risks, alcohol and tobacco use, drunk driving, and the use of seat belts.

Importing the Dataset (again)

- Like before, we can use point-and-click or the working directory

```
setwd("YourFilePath")
```

```
yrbss <- read.csv("yrbss2013.csv", header = TRUE)
```

- Let's also examine the structure and give an overview of the dataset

```
str(yrbss)
```

```
summary(yrbss)
```

Sampling Distribution of Proportions

Let's look at `bullied`

Let's look at the distribution of students that have been bullied. We can use `table()`, `prob.table()`, and `barplot()`

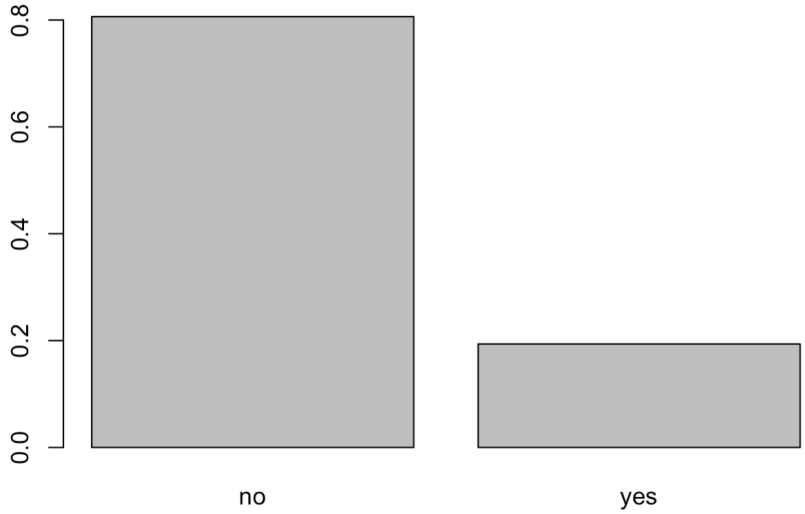
```
Console   Terminal ×   Background Jobs ×
R 4.4.1 · ~/
> table(yrbss$bullied)

  no  yes
6839 1643
> prop.table(table(yrbss$bullied))

       no       yes
0.8062957 0.1937043
```

`table(yrbss$bullied)`

`prop.table(table(yrbss$bullied))`



`barplot(prop.table(table(yrbss$bullied)))`

Taking random samples

We can take random samples (assuming or dataset represents the population) using the `sample()` command.

```
bullied_sample1 <- sample(x = yrbss$bullied, size =
10)
```

In this case, our random sample contains 10 observations. Examining further as before using the following commands.

```
length(bullied_sample1)
prop.table(table(bullied_sample1))
```

Depending on which 10 observations were randomly selected, your estimated proportion could be a bit above or below the true population proportion.

The `for` loop

Loop = iteration

- The idea behind a `for` loop is *iteration*, allowing you to execute the same code for as many times as you want without having to type out every iteration
- Loops are great, but they are a pain in the ass! Probably the most frustrating part about coding haha imho

Without a `for` loop

Suppose we want to take another two other samples of 10 observations and calculate its sample proportion. Code would look like

```
bullied_sample2 <- sample(x = yrbss$bullied, size =
10)
mean(bullied_sample2=="yes")
```

and another.....

```
bullied_sample3 <- sample(x = yrbss$bullied, size =
10)
mean(bullied_sample3=="yes")
```

Superiority of the `for` loop

However, with a `for` loop, we can "simplify" the following as the code below. We can even do this for 1000 times or even more

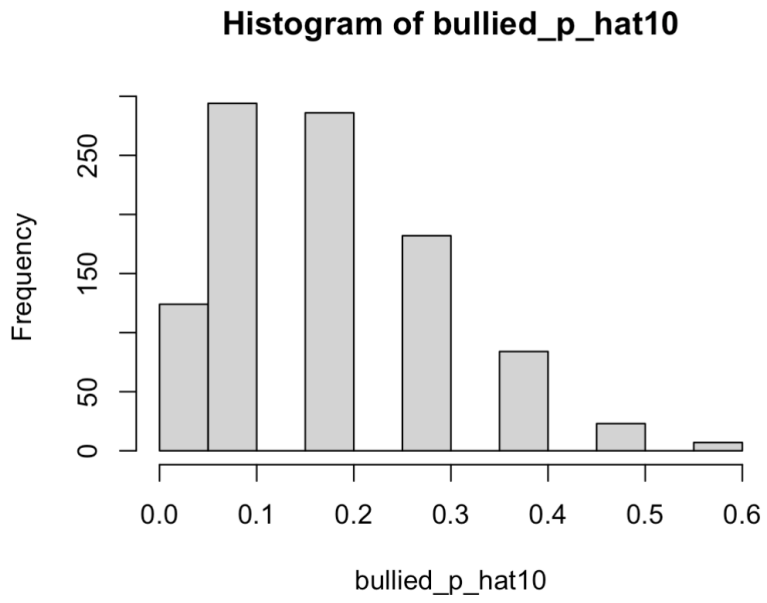
```
bullied_p_hat10 <- rep(NA, 1000)
for(i in 1:1000){
samp <- sample(yrbss$bullied, 10)
bullied_p_hat10[i] <- mean(samp=="yes")
}
```

Let's look at the code line-by-line. Our first "difficult" code lol.

Summarizing Sampling Distribution of Proportions

Visualizing our 1000 samples

The object we created, `bullied_p_hat10` represents a sampling distribution of 1000 sample proportions. We can summarize this distribution using the `hist()` function as before!




```
hist(bullied_p_hat10)
```

Try running `mean()` and `sd()`. What do you notice?

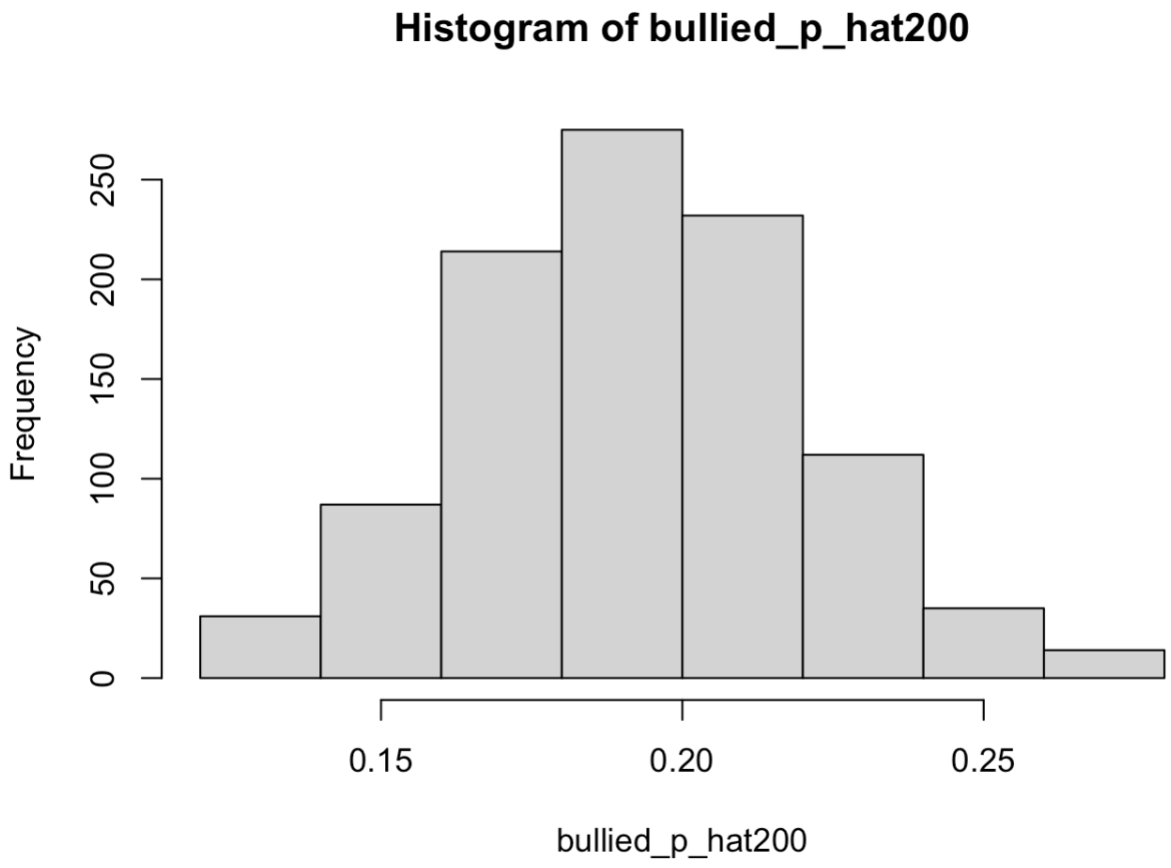
Notice the 1000 samples

Notice that the sampling distribution is *not* normally distributed (because of its small individual sample size $\mathtt{n} = 10$

```
bullied_p_hat200 <- rep(NA, 1000)
for(i in 1:1000){
samp <- sample(yrbss$bullied, 200)
bullied_p_hat10[i] <- mean(samp=="yes")
}
```

What do you notice when you rerun the histogram?

With a larger sample...



`hist(bullied_p_hat200)`

Mean closer to p and much lower spread!