Today's big idea: Sampling distributions

Concepts (found in Lock5 Section 3.1)

population vs. sample

Sample size n

Sample statistic: a quantity computed from the sample

examples:

$\bar{x}$, s, median, any percentile — from a quantitative variable

count, proportion ($\hat{p}$) — from a categorical variable

population parameters

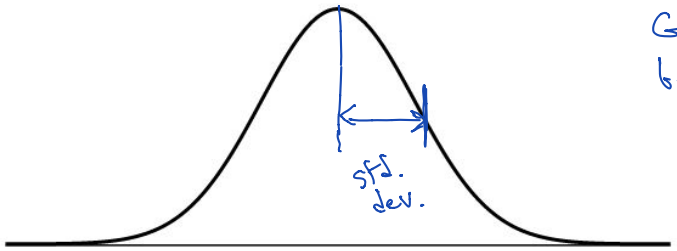
- usually unknowable
- counterparts to sample statistics that
  - we denote by (usually) Greek letters

    μ = population mean, σ = population std. dev — quantitative vars

    p = population proportion — categorical vars
  - are considered "fixed"

Normal Distributions

gendarme's hat
normal dists.
Gaussian dists
bell curves



If we want to refer to the std. dev. of a sampling dist., let's agree to call it the <u>standard error</u>.

Notes on sampling

Class Feb. 22

February 22, 2021

Population and parameters

The data of players from Major League Baseball in the year 2018 who had a count of at least 100 at-bats will be considered the **population**, not as a **sample** from some larger population. First let's load it and display the names of variables in the data set.

```
mlb18 <- read.csv("http://scofield.site/teaching/data/csv/mlb18abEligible.csv")
names(mlb18)
```

```
##  [1] "X"                    "name"                 "team"
##  [4] "position"             "games"                "AB"
##  [7] "R"                    "H"                    "doubles"
## [10] "triples"              "HR"                   "RBI"
## [13] "walks"                "strike_outs"          "stolen_bases"
## [16] "caught_stealing_base" "AVG"                  "OBP"
## [19] "SLG"                  "OPS"
```

It's average number of runs scored can be calculated from this full population:

```
mean(~R, data=mlb18)
```

```
## [1] 47.18807
```

As long as this is our population, this number stays fixed. It is called a **parameter**. We denote the population mean by the Greek letter μ, so for runs scored, the population mean $\mu = 47.19$. There are other numbers one might measure from the entire population of `runs scored`, such as the standard deviation (σ), the first quartile, and so on. Again, so long as we use this population, these numbers are fixed *parameters*.

Sampling, and sample statistics

For many statistical studies, The population is not of manageable size, unlike the case for MLB 2018 data. We seek to make an *inference* about the value of μ based on sample data.

To draw a sample of size $n = 3$ from the population above, R provides the command

```
sample(mlb18, size=3)
```

```
##       X         name team position games  AB  R  H doubles triples HR RBI walks
## 359 359    Blanco, G   SF       CF    68 189 19 41       7       3  2  12    12
## 411 411  Phillips, B   KC       RF    51 134 15 25       4       3  2  11    11
## 431 431     Perez, R  CLE        C    62 179 16 30       9       1  2  19    21
##     strike_outs stolen_bases caught_stealing_base   AVG   OBP   SLG   OPS
## 359          58            6                    2 0.217 0.262 0.317 0.580
## 411          61            1                    1 0.187 0.252 0.306 0.558
## 431          70            1                    0 0.168 0.256 0.263 0.519
```

```
##     orig.id
## 359     359
## 411     411
## 431     431
```

which displays 3 randomly-selected player records from the full data set.

To estimate μ, the population mean for `runs scored`, we might select a random sample of some manageable size, and calculate the **sample mean**, denoted by $\overline{x}$. You can do this using several commands/steps

```
mySample <- sample(mlb18, size=10)
mean(~R, data=mySample)
```

```
## [1] 53.8
```

or combine the above into a single compound command

```
mean(~R, data=sample(mlb18, size=10))
```

```
## [1] 51.4
```

Here I have used the sample size $n = 10$.

I calculated the mean of my sample here, but one could do other sorts of calculations using the sampled data, including the *median*, *3rd quartile*, *standard deviation* (denoted by s), etc. Any calculation performed from sample data in this manner is referred to as a **sample statistic**, or just a **statistic**.

To illustrate, the next command draws a sample of size $n = 10$ and produces the sample standard deviation s:

```
sd(~R, data=sample(mlb18, 10))
```

```
## [1] 35.31509
```

Unlike population parameters, sample statistics vary depending on the sample.

Sampling distributions

We want to estimate the population parameter μ, the average number of `runs scored` by players in the population, using the mean $\overline{x}$ of a sample. We actually took a sample of size $n = 10$ above, and calculated from that sample $\overline{x} = 51.4$, somewhat different from the actual value $\mu = 47$, which we had the luxury of being able to calculate. Next, I draw another sample of size $n = 10$ to repeat the calculation of $\overline{x}$, finding it again to be different from the correct value.

```
mean(~R, data=sample(mlb18, size=10))
```

```
## [1] 46.5
```

One begins to wonder things like this:

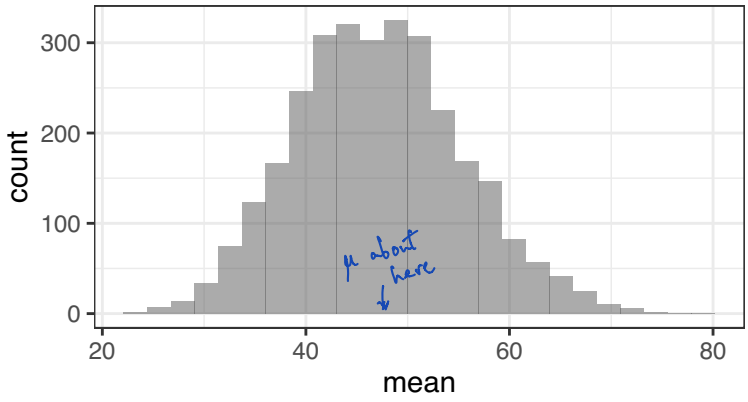
- While I don't expect $\overline{x}$ to hit μ exactly, is it a number I expect to be *accurate*—that is, have no more of a tendency to miss low in greater amounts than it misses high. In other words, is $\overline{x}$ an **unbiased estimator** of μ?
- Just how variable is $\overline{x}$? What values can it take, and how frequently do they arise? What is the **distribution** of $\overline{x}$? The thing we want is usually called the **sampling distribution of the sample mean**.

To get a picture of the sampling distribution, I'll draw not just one sample of size $n = 10$, but many. That is what the first line in the code block below does, storing the result in a data frame called `manyMeans`. I follow this up by displaying the first few lines of `manyMeans`, then making a histogram of the full list of 3000 $\overline{x}$ computed from samples taken.

```
manyMeans <- do(3000) * mean(~R, data=sample(mlb18, 10))
head(manyMeans)
```

```
##   mean
## 1 65.6
## 2 52.8
## 3 44.1
## 4 53.0
## 5 48.6
## 6 48.4
```

```
gf_histogram(~mean, data=manyMeans)
```



The middle of this distribution, its *mean* can be computed as well,

```
mean(~mean, data=manyMeans)
```

```
## [1] 47.1418
```

which shows that μ, the population parameter, is in the center, even if it is often missed by $\overline{x}$. In other words, $\overline{x}$ is an unbiased estimator of μ.

What we did above may be described in these terms: We repeated, 3000, times the process

- draw a sample of size 10
- calculate the mean from that sample

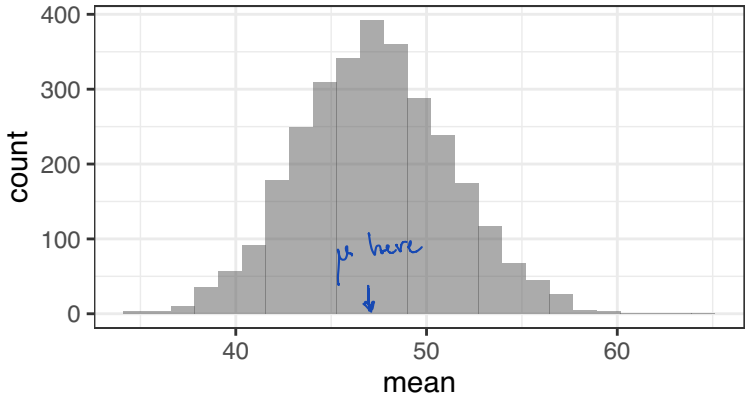
If we did this again—that is, if we repeated the process another 3000 times—we would surely be using different samples, and witness slight changes in the overall outcome. But, without a careful eye, the changes might be so small as to be hardly noticeable. What will make a *very* noticeable change is to alter the sample size. In what follows, I nearly duplicate the commands from above, but make the size of samples $n = 40$.

```
manyMeans <- do(3000) * mean(~R, data=sample(mlb18, 40))
head(manyMeans)
```

New sample size

```
##     mean
## 1 54.500
## 2 47.425
## 3 51.050
## 4 43.625
## 5 41.325
## 6 48.650
```

```
gf_histogram(~mean, data=manyMeans)
```




```
mean(~mean, data=manyMeans)
```

```
## [1] 47.29178
```

Observations and definitions

In both cases, the sampling distributions appear roughly bell-shaped, and both are centered at μ (or quite close to it). There is a difference in *spread*, with the sampling distribution of $\overline{x}$ being less spread out when $n = 40$ than when $n = 10$. We use the standard deviation as a measure of that spread.

```
sd(~mean, data=manyMeans)
```

```
## [1] 3.975387
```

Since there are various standard deviations one could be referring to at this stage, this standard deviation, specific to the sampling distribution of $\overline{x}$, is called the **standard error of the mean**, or $\text{SE}_{\overline{x}}$.