CSC100/CSC200 Lab #11: Quantifying Uncertainty

Dominick Bello

Fall 2022 | Data Science for the World

Please complete this notebook by filling in the cells provided. When you're done:

1. Remember to put your name in the header at the top of this notebook where it says `author`.
2. Select `Knit (Knit to Word)` from the toolbar menu.
3. Read that file! If any of your lines are too long and get cut off, we won't be able to see them, so break them up into multiple lines and knit again.
4. Save that Word document as a PDF file.
5. Submit BOTH this `.Rmd` file and the **PDF** file you generated to Gradescope. Some questions are autograded and you may improve your score on the tests given by resubmitting your work as many times as you like up to the deadline.
6. **Passing the automatic tests given does not guarantee full credit on any question.** The tests are provided to help catch some common mistakes, but it is *your* responsibility to answer the questions correctly.

If you are having trouble submitting, ask your lab instructor for help.

This lab assignment is due **November 16 at 11:59PM**.

Reading:

- Chapter 9 Textbook

Run the cell below to prepare the notebook.

Part I: Rock Band Popularity. The University of Lost World has conducted a staff and faculty survey regarding their most favorite rock bands. The university received 200 votes, which are summarized as follows:

- Pink Floyd (35%)
- Led Zeppelin (22%)
- Allman Brothers Band (20%)
- Yes (12%)
- Uncertain (11%)

In the following, we will use `"P"`, `"L"`, `"A"`, `"Y"`, and `"U"` to refer to the artists. The following tibble `rock_bands` summarizes the information:

```
rock_bands <- tibble(
  band_initial = c("P", "L", "A", "Y", "U"),
  proportion = c(0.35, 0.22, 0.20, 0.12, 0.11),
  votes = proportion * 200
)
rock_bands
```

```
## # A tibble: 5 × 3
##   band_initial proportion votes
##   <chr>             <dbl> <dbl>
## 1 P                  0.35    70
## 2 L                  0.22    44
## 3 A                  0.2     40
## 4 Y                  0.12    24
## 5 U                  0.11    22
```

These proportions represent just a *sample* of the population of University of Lost World. We will attempt to estimate the corresponding population parameters - the *proportion* of listening preference for each rock band in the population of University of Lost World staff and faculty. We will use confidence intervals to compute a range of values that reflects the uncertainty of our estimate.

Question 1. Using `rock_bands`, generate a tibble `votes` containing 200 rows corresponding to the votes. You can group by `band_initial` and repeat each band's row `votes` number of times by using `rep(1, each = votes)` within a `slice()` call (remember computing *within groups*?). Then form a tibble with a single column named `vote`.

Here is what the first few rows of this tibble should look like:

vote
A
A
A
A
A

…

```
votes <- rock_bands %>%
  group_by(band_initial) %>%
  slice(rep(1, each = votes)) %>%
  select(vote = band_initial) %>%
  ungroup()

votes
```

```
## # A tibble: 200 × 1
##    vote
##    <chr>
##  1 A
##  2 A
##  3 A
##  4 A
##   5 A
##   6 A
##   7 A
##   8 A
##   9 A
## 10 A
## # … with 190 more rows
```

```
. = ottr::check("tests/band_p1_q1.R")
```

```
## All tests passed!
```

We will conduct bootstrapping using the tibble votes.

Question 2. Write a function `one_resampled_statistic(num_resamples)` that receives the number of samples to sample *with replacement* (why not without?) from `votes`. The function resamples from the tibble `votes` `num_resamples` number of times and then computes the proportion of votes for each of the 5 rock bands. It returns the result as a *tibble* in the same form as `rock_bands`, but containing the resampled votes and proportions from the bootstrap.

Here is one possible tibble after running `one_resampled_statistic(100)`. The answer will be different each time you run this!

vote	votes	proportion
A	23	0.23
L	19	0.19
P	40	0.40
U	7	0.07
Y	11	0.11

```
one_resampled_statistic <- function(num_resamples) {
sample_of_rock_bands <- slice_sample(votes, n = num_resamples, replace =
TRUE)
tibby<- sample_of_rock_bands %>%
  group_by(vote) %>%
  summarize(votes = n(),
  proportion = votes/num_resamples)
return(tibby)
}

one_resampled_statistic(100) # a sample call
```

```
## # A tibble: 5 × 3
##   vote  votes proportion
##   <chr> <int>      <dbl>
## 1 A        19       0.19
## 2 L        16       0.16
## 3 P        45       0.45
## 4 U          9       0.09
## 5 Y         11       0.11

. = ottr::check("tests/band_p1_q2.R")

## All tests passed!
```

Question 3. Let us set two names, `num_resamples` and `trials`, to use when conducting the bootstrapping. `trials` is the desired number of resampled proportions to simulate for each of the bands. This can be set to some large value; let us say 1,000 for this experiment. But what value should `num_resamples` be set to, which will be the argument passed to `one_resampled_statistic(num_resamples)` in the next step?

```
num_resamples <- 100
trials <- 1000

. = ottr::check("tests/band_p1_q3.R")

## All tests passed!
```

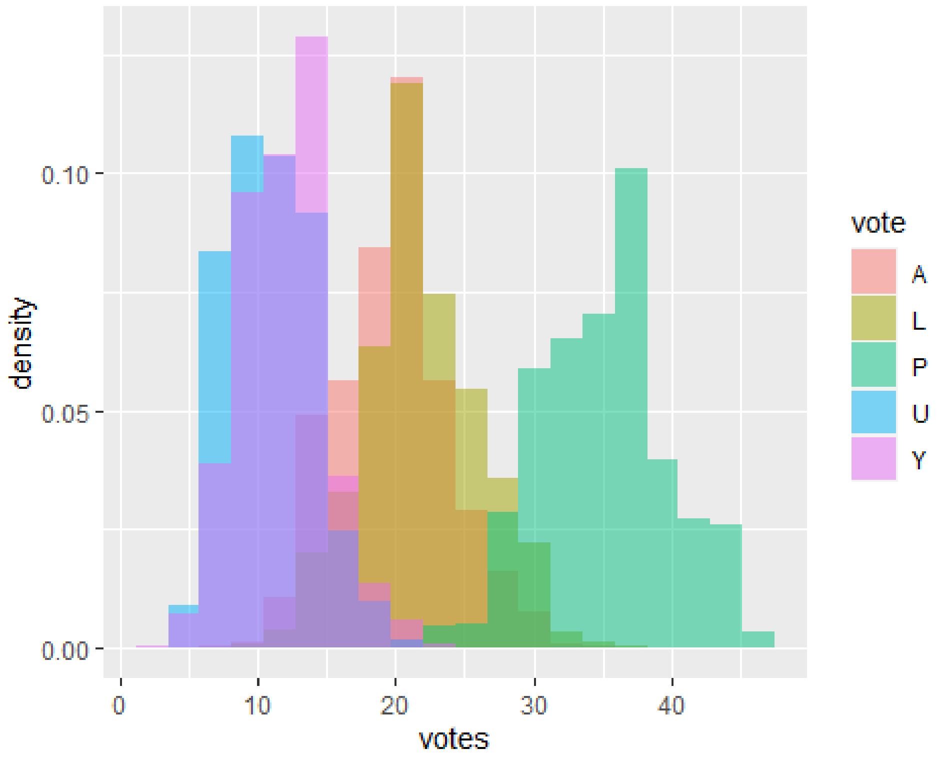
The following code chunk conducts the bootstrapping using your `one_resampled_statistic()` function and the names `trials` and `num_resamples` you created above. It stores the results in a vector `bstrap_props_tibble`.

```
bstrap_props_tibble <- replicate(n = trials,
                                 one_resampled_statistic(num_resamples),
                                 simplify = FALSE) %>%
                       bind_rows()
bstrap_props_tibble

## # A tibble: 5,000 × 3
##    vote  votes proportion
##    <chr> <int>      <dbl>
##  1 A        20       0.2
##  2 L        21       0.21
##  3 P        41       0.41
##  4 U         5       0.05
##  5 Y        13       0.13
##  6 A        18       0.18
##  7 L        26       0.26
##  8 P        35       0.35
##  9 U        14       0.14
## 10 Y         7       0.07
## # … with 4,990 more rows
```

Question 4. Generate an overlaid histogram using `bstrap_props_tibble`, showing the five distributions for each band. Be sure to use a positional adjustment to avoid stacking in the bars. You may also wish to set an alpha to see each distribution better. Use 20 for the number of bins.

```
ggplot(bstrap_props_tibble, aes(x=votes, fill = vote)) +
   geom_histogram(alpha = 0.5, aes(y = ..density..), position = 'identity',
bins = 20)
```



We can see significant difference in the popularity between some bands. For instance, we see that the bootstrapped proportions for P is significantly higher than Y's by virtue of no overlap between their two distributions; conversely, U and Y overlap each other completely showing no significant preference for U over Y and vice versa. Let us formalize this intuition for these three bands using an approximate 95% confidence interval.

Question 5. Define a function `cf95` that receives a *vector* `vec` and returns the approximate "middle 95%" using `quantile`.

```
cf95 <- function(vec) {
 desired_area <- 0.95
middle95 <- quantile(vec, 0.5 + (desired_area / 2) * c(-1, 1), type = 1)
middle95
}

. = ottr::check("tests/band_p1_q5.R")

## All tests passed!
```

Let us examine the 95% confidence intervals of the bands P, Y, and U, respectively.

```
bstrap_props_tibble %>% filter(vote == "P") %>% pull(proportion) %>% cf95()
# 95% CI for P
```

```
##  2.5% 97.5%
##  0.26  0.44

bstrap_props_tibble %>% filter(vote == "Y") %>% pull(proportion) %>% cf95()
# 95% CI for Y

##  2.5% 97.5%
##  0.06  0.19

bstrap_props_tibble %>% filter(vote == "U") %>% pull(proportion) %>% cf95()
# 95% CI for U

##  2.5% 97.5%
##  0.06  0.18
```

Question 6. By looking at the upper and lower endpoints of each interval, and the overlap between intervals (if any), can you say whether P is more popular than Y or U? How about for Y, is Y more popular than U?

P is more popular than Y and U. Y is more popular than U.

Part II: More Confidence Intervals! Your instructors computed the following approximate 95% confidence interval for the proportion of band P votes (your answer might have been different and that is okay!).

$$[.285, .42]$$

Question 1. Can we say that 95% of the population of faculty lies in the range $[.285, .42]$? Explain your answer.

Yes because the data layers over each other. So the population estimate can fall into the given range.

Question 2. Can we say that there is a 95% probability that the interval $[.285, .42]$ contains the *true* proportion of the population who listens to band P? Explain your answer.

HINT: Be careful with this one! Think about how chance enters the picture. Once some interval has been generated, as above, is there *chance* involved in determining whether the parameter was captured? If not, what does it mean to have an "approximate 95% confidence interval"?

Yes because the confidence interval we got for P is similar. Chance plays a part each simulation that you create, but the confidence interval only cares if the valued fall within the interval 95% of the time. This gives leeway for when the population is not within the confidence interval.

Question 3. The instructors also created 80%, 90%, and 99% confidence intervals from one sample for the popularity of band P, but they forgot to label which confidence interval represented which percentages! Match the interval to the percent of confidence the interval represents. Then, explain your thought process.

- [0.265,0.440]
- [0.305,0.395]
- [0.285,0.420]

```
1.    99%

2.    80%

3.    90%
```

Yowza – you finished Lab #11!