Statistical Thinking: Week 3 Lab

Due 12noon Monday 24 August 2020

Introduction

Our focus this week is build the randomisation test for the Gender discrimination case study data. In addition, we will undertake a traditional CLT-based test for equal population proportions, and a further test regarding a single, given proportion.

In terms of **R** skills, we will look at some new geoms for plots, review the *pivot_wider()* command and its companion *pivot_longer()*. These two commands are handy for reshaping a tibble to suit the analyst's need to use particular available **R** functions. We will also iterate using a for-loop.

A. Preliminary Instructions

B. Lab Exercises

C. Lab Submission Instructions

A. Instructions

Before you begin **B. Lab Exercises**, review each of these detailed instructions.

- Create, name and save a new .Rmd file as **Lab03_nnnnnnnn.Rmd**, with your Monash student ID number replacing the segment **nnnnnnnn** in the file name. Including in the **YAML** section
  - a suitable title (e.g. Statistical Thinking Lab 3")
  - put your name and student-id as the author information
  - set the date "Week 3, 2020".
  - ensure all code chunks will show using the `knitr::opts_chunk$set(echo = TRUE)` command in the initial code chunk.[1]
- Read through all of the exercises before attempting the lab.
- Review any relevant help files or other sources of information.
- Format your **R Markdown** files using (sub-)section heading titles corresponding to the lab exercise section headings.
- Consider using a ggplot **theme** to achieve a pleasing and consistent style for all plots in your lab report. (This is optional, but fun to do!) For information on ggplot2 themes, see:
  - **R** help for *ggplot2::theme* (note the link there to a section on *complete themes*)
  - this R graphics tutorial

[1] It is OK to keep `include=FALSE` for this initial code chunk, as seen in the initial code chunk named "setup" of the R Markdown template file.

  - the *ggthemes** package noted in the link above.

- Add a code chunk that loads the required packages, which are at a minimum[2]:
  - *tidyverse*, *broom* and *kableExtra*
  - install any of the above packages not previously installed on your machine.

- Review the Lab submission instructions in Section **C** for additional remarks, particularly before you submit your files.

B. Lab Exercises

Randomization case study: gender discrimination

Consider the case study example discussed in Section 2.1 of the prescribed ISRS textbook. The research question we hope to answer is, *"Are females discriminated against in promotion decisions made by male managers?"*

The participants in this study were 48 male bank supervisors attending a management institute at the University of North Carolina in 1972. They were asked to assume the role of the personnel director of a bank and were given a job applicant resume to judge whether the applicant should be promoted to a branch manager position. The files given to the participants were identical, except that half of them indicated the job candidate was male and the other half indicated the job candidate was female. These files were randomly assigned to the supervisors to make their judgement.

The *gender* (male or female) associated with the assigned file and the resulting promotion **decision** to (promote or not promote) was recorded. Using the results of the study, we would like to evaluate if females are unfairly discriminated against in promotion decisions. In this study, a smaller proportion of females are promoted than males (0.583 versus 0.875), but it is unclear whether the difference provides convincing evidence that females are unfairly discriminated against, or if the outcomes could likely have occurred by chance with the actual true promotion opportunity was the same for all, regardless of gender.

The textbook provides the following summary table about the study outcomes:

		decision		
		promoted	not promoted	Total
gender	male	21	3	24
	female	14	10	24
	Total	35	13	48

Table 2.1: Summary results for the gender discrimination study.

We are interesting in testing the upper one-sided hypothesis test:

$$H_0 : p_M - p_F = 0 \quad \text{vs.} \quad H_1 : p_M - p_F > 0,$$

where p_M and p_F denote the population proportions of male and female job applicants promoted, respectively.

In this lab we will do two approximate versions of the test, using a CLT-based test and a Randomisation test based on permutations of the original data. Along the way, we will review and discover many useful **R** functions and packages too.

Good luck and have fun!

[2]For example, students may also need to include the *ggthemes* package.

Exercise 1: Construct the case study tibble

1. Create a tibble named *GDS*, as shown in the code chunk below, that is related to the table labelled Table 2.1 above. Explain what is contained in *GDS*.

```
GDS <- tibble(gender = c(rep("male", 24), rep("female", 24)),
    decision = c(rep("promoted", 21), rep("not promoted", 3),
        rep("promoted", 14), rep("not promoted", 10)))
```

Exercise 2: Produce a summary tibble, by group

2a. Use the **R** code chunk below to produce and print the tibble named *GDSS4* that effectively replicates the tallies as displayed in Table 2.1 and also includes the relevant sample proportions.

Notice the use of the back-ticks around the value `not promoted` to ensure the entire phrase is treated as a single value when used as an argument in the *mutate()* function.[3]

```
GDSS1 <- GDS %>% group_by(gender, decision) %>% tally() %>% ungroup()
GDSS2 <- GDSS1 %>% pivot_wider(names_from = decision, values_from = n)
GDSS3 <- GDSS2 %>% mutate(total = promoted + `not promoted`) %>%
    mutate(prop = round(promoted/total, digits = 3)) %>% select(gender,
    promoted, `not promoted`, total, prop)
GDSS4 <- GDSS3 %>% arrange(desc(row_number()))
GDSS4 %>% kable() %>% kable_styling()
```

2b. Describe the contents of the progressively produced tibbles, *GDSS1* through *GDSS4*, as produced in the code chunk considered in part **2a.** Each of the four stages is identified by a comment in the code chunk.

Exercise 3: pivot_wider vs. pivot_longer()

In **Exercise 2** we used the *pivot_wider()* command from the *tidyr* package to reshape the "longer" tibble named *GDSS1* to the "wider" form named *GDSS2*. This is shown in the code chunk portion indicated as "stage 2".

The *GDSS2* tibble contains three variables"
+ gender, stored in rows,
+ decision, spread across the column names `not promoted`, and `promoted`, and
+ `count` stored in the cell values of `not promoted` and `promoted`.

We use *pivot_longer()* to revert back to the form in *GDSS1*, as shown in the **R** code chunk below.

```
GDSS2 %>% pivot_longer(-gender, names_to = "decision", values_to = "n")
```

3. Find and report the names of two (now "retired") *tidyr** functions that *pivot_wider* and *pivot_longer* have effectively replaced.

Exercise 4: Visualise the data

4a. Replicate the **R** commands in the code chunk below in your Lab report and produce a single plot comprised of two barplots, each showing the counts of the *decision* variable for a gender, and positioned side-by-side.

[3]Rendered Back-tick marks in a .pdf appear as a single quotation mark. Be aware of this issue if using copy-paste as the pasted code will not run properly in **R**. Copy-paste from the rendered .html file does not appear to have this problem.

```
ggplot(GDS, aes(x = decision, color = gender, fill = gender)) +
    geom_bar() + facet_wrap(~factor(gender), nrow = 1)
```

Notice the following:

- The *geom_barplot()* plot provides a summary of the "tidy data" *GDS* tibble (i.e. it used the individual level data, and is not produced by a summarised version of it).
- By using[4] *facet_wrap(~factor(gender))*, the two barplots share the same vertical (y-) axis, and adds a legend with distinct colours for the different values of the *gender* variable.

4b. Explain what happens if the *facet_wrap()* function is removed from the code chunk shown in part **4a**.

```
ggplot(GDS, aes(x = decision, color = gender, fill = gender)) +
    geom_bar()
```

Exercise 5: The sample difference in proportions

Let $\hat{p}_M$ and $\hat{p}_F$ denote the sample corresponding sample quantities.

5a. The **R** code chunk below provides three different **R** expressions for computing *xobs*, by extracting the relevant elements of *GDSS4*. While all provide the same numerical value, which expression do you think perhaps should be preferred, and why?

5b. Calculate, save and report the difference in the sample proportions, $xobs = \hat{p}_M - \hat{p}_F$, using the method preferred in part **5a.**

```
GDSS4$prop[1] - GDSS4$prop[2]
GDSS4$prop[GDSS4$gender == "male"] - GDSS4$prop[GDSS4$gender ==
    "female"]
(GDSS4 %>% pull(prop))[1] - (GDSS4 %>% pull(prop))[2]
```

Exercise 6: Do a traditional CLT-based hypothesis test

In this exercise we will use the available **R** function *prop.test()* to do the relevant traditional CLT-based test. Instructions are given in the **R** code chunk below.

6a. Comment on the other function arguments[5] needed, if any, to obtain the desired hypothesis results from the *prop.test()* function, as shown in the **code chunk**.

```
test_x <- as.matrix(GDSS4 %>% select(promoted, `not promoted`))
CLTtest <- prop.test(test_x)
tidy(CLTtest)
```

6b. Report the conclusion of the hypothesis test on the basis of this approximate test, and if relevant, provide a statement regarding the "strength of evidence" against H_0 suggested by the test.

6c. The *tidy()* function used in the previous code chunk comes from the *broom* package. What does this function do?

[4] See also *facet_grid()*.

[5] You are not required to discuss the use of the Yates' continuity correction, but retain the default setting "correct=TRUE".

Exercise 7: The Randomisation test

The calculations to implement the randomisation test in this setting are shown in the **R** code chunk below.

```
n <- nrow(GDS)
R <- 1000
Rxobs <- array(dim = R)
set.seed(2020.3)
RGDS <- GDS
for (r in 1:R) {
    RGDS <- RGDS %>% mutate(gender = sample(RGDS$gender, n, replace = FALSE))
    RGDSS3 <- RGDS %>% group_by(gender, decision) %>% tally() %>%
        ungroup() %>% pivot_wider(names_from = decision, values_from = n) %>%
        mutate(total = `not promoted` + promoted) %>% mutate(prop = round(promoted/total,
        digits = 3))
    Rxobs[r] <- RGDSS3$prop[GDSS3$gender == "male"] - RGDSS3$prop[GDSS3$gender ==
        "female"]
}
```

7a. Replicate the code above in your Lab report. In addition, add the required **R** code to calculate an appropriate (and approximate) p-value of the test. Name the object containing the p-value as "pval".

7b. What does the *set.seed(2020.3)* command do, and why is it included in the code for the Randomisation test?

Exercise 8: Visualise the sampling distribution and the p-value

Consider the image below of an **R** code chunk containing both **R** commands and the **code chunk delimiters** that have been customised to contain certain *knitr* options.[6]

```
Rxobs_tbl <- as_tibble(Rxobs) %>% mutate(r = 1:R)
Rxobs_tbl %>% ggplot(aes(value)) + geom_histogram(colour = "blue",
    fill = "blue", alpha = 0.5, bins = 5) + xlab(expression(xobs^"[r]")) +
    ggtitle(expression(paste("Approximate sampling distribution of ",
        hat(p)[M] - hat(p)[F], " under ", H[0])))
```

8a. Replicate the **R** code above in your Lab report, with the following adjustments:
i. Modify the value used in the "bins" option in *geom_histogram()* so that the histogram bars distinguish the unique *Rxobs* values, and
ii. Include a vertical red line with x-intercept at the observed *xobs* value.

8b. Report the conclusion of the hypothesis test on the basis of this approximate test, and if relevant, provide a statement regarding the "strength of evidence" against H_0 suggested by the test.

C. Lab submission instructions**

Lab 3 submissions must be completed before 12noon on Monday 24 August 2020. To submit the Week 3 Lab, you must complete the Week 3 Lab Submission "quiz" in Moodle (with correct answers), following all instructions provided on Moodle.

Your complete submission will be assessed with regard to

[6] For more information about **R** code chunk delimiters and available *knitr* options, see Chapter 3: Code Chunks in **RStudio**'s **R Markdown** guide.

i. Completion of the required exercises

ii. Inclusion of the appropriately named **Lab03_IDnnnnnnnn.Rmd** and a print of the `Lab03_IDnnnnnnnn.html` document as a .pdf document, named **Lab03_IDnnnnnnnn.pdf**.

iii. You can attempt the submission quiz as many times as you wish, however ONLY the final **submitted attempt** will be marked.

iv. You must **formally submit** at least one Lab 3 submission quiz on Moodle to gain credit for this assessment task (i.e. you must press the 'submit' button for the Lab submission quiz to be accepted for marking).[7]

Final remarks

- Remember that answers containing an **R** code chunk are incomplete unless they are accompanied by a clear statement of what the relevant code chunk produces, with all graphs and tables described.
- Ensure all plots produced have appropriate titles, axis labels, legends, etc, as appropriate. Similarly, ensure all tables are nicely formatted (e.g. using options from the *kableExtra* package), are clearly positioned and labelled to correspond to the text.
- Proofread your submissions, checking for correct spelling, grammar and punctuation as well as to ensure the text expresses what you intend.
- Be sure to double check that you have uploaded both files, and that these are the correct versions of your files, *before you do your final submission*!

[7] This is a change from previous Lab submissions. The change has been made because correcting for empty final quiz attempts submitted at expiry takes far too much time to reconcile.