Lab 9

Stats 32: Introduction to R for Undergraduates

Harrison Li

4/30/2024

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

Today's lab will be a bit unusual in that we will not look at any real data. Instead, we will examine various simulated datasets, in an effort to better understand the nuances of hypothesis testing.

An iron bar factory

Suppose Metallic Factory produces iron bars that are supposed to be of length 15cm. A quality control engineer at Metallic Factory would naturally want to know whether her machines are truly producing bars that are, on average, 15 cm. A natural set of hypotheses she could test would be: - H_0: $\mu = 15$ - H_1: $\mu \neq 15$ where μ is the true mean length of all bars produced at Metallic Factory.

1. Using this notation, write down hypotheses for a one-sided testing scenario, where the alternative hypothesis is that the iron bars have length longer than 15cm.

This engineer does not know the value of μ. Her goal is to estimate μ based on a finite sample of bars. However, we will play the role of an omnipotent deity that knows μ, so that we can use simulation to see what happens under various different scenarios for the true underlying distribution.

Let's start by assuming the bar lengths are uniform between 14.9 cm and 15.1 cm. Then in fact $\mu = 15$ (i.e. H_0 is true). We assume the engineer only has time to look at 90 bars. The following code simulates the length of 90 bars drawn from this assumed *data-generating distribution* (again, unknown to the engineer). We use the `runif()` function which conveniently generates uniform random numbers. Before doing that, we set the "seed" so that the randomness can be reproduced.

```
set.seed(2024)
bar_lengths <- runif(n=90, min=14.9, max=15.1)
bar_lengths
```

```
##  [1] 15.06739 14.96417 15.03607 15.03963 14.99140 15.04028 14.98314 14.96064
##  [9] 15.07532 14.92381 15.08038 15.09062 15.04071 15.00316 14.98989 15.06845
## [17] 14.92206 15.07264 14.92621 14.95909 14.90945 14.92240 15.03609 15.07000
## [25] 15.05947 14.95846 15.02305 15.06312 14.90011 14.90423 14.98823 14.95806
## [33] 14.93524 15.09468 15.03503 15.03619 14.91083 15.09322 14.90443 15.02947
## [41] 15.07457 15.08209 15.02478 14.97839 15.07029 14.98964 14.98313 14.95585
## [49] 15.03087 15.03010 14.91227 15.04822 14.99059 14.99790 15.05065 15.04317
## [57] 14.90398 14.93838 14.99129 14.95564 14.91499 14.98532 14.97449 15.02788
## [65] 14.96134 15.06296 15.07715 15.04377 14.96178 14.95745 15.03032 14.93969
## [73] 15.02641 14.98880 14.96247 15.04170 14.95682 14.90718 15.03316 14.94408
## [81] 14.93830 14.94907 14.99963 14.90352 15.03797 14.99658 15.09515 14.96158
## [89] 14.92314 15.07175
```

Let's compute the p-value for the above hypotheses according to a one sample t test:

```
t.test(bar_lengths, alternative="two.sided", mu=15)
```

```
##
##  One Sample t-test
##
## data:  bar_lengths
## t = -0.37829, df = 89, p-value = 0.7061
## alternative hypothesis: true mean is not equal to 15
## 95 percent confidence interval:
##  14.98562 15.00978
## sample estimates:
## mean of x
##   14.9977
```

With a p-value of 0.706, we cannot reject the null H_0 at any reasonable significance level (e.g. 0.05). This should be re-assuring, since H_0 is in fact true.

2. What is the p-value according to the one-sided test in problem 1, using the data in `bar_lengths`? Do you reject the null hypothesis?

However, recall that a significance level of 0.05 means that the probability of rejection, when the null is true, is at most 0.05. In other words, there is up to a 5% chance of falsely rejecting, even when the null is true.

What does this really mean? Since we're omnipotent, we can take many samples of size 90 from the uniform data-generating distribution above. Let's get 1,000 such samples:

```
samples <- matrix(runif(n=1000*90, min=14.9, max=15.1), ncol=1000)
```

Each row in `samples` corresponds to an independent sample of 90 bar lengths drawn uniformly between 14.9 and 15.1.

For each of these 1,000 rows, we can run a t test and extract p values, giving a vector of 1,000 p-values. Note the use of `purrr::map_dbl`, which applies a function to each element of a list, and returns the result as a numeric vector (assuming the function indeed returns a number). To use `purrr::map_dbl`, we convert `samples` to a data frame (recall from the early lectures that a data frame is a list under the hood, with each column a new element of the list):

```
p_values <- purrr::map_dbl(.x=as.data.frame(samples),
                           .f=function(x) t.test(x, alternative="two.sided", mu=15)$p.value)
```

The `.f` argument is an *anonymous function* - a function without a name that takes in a single input "x" (we could've named it anything else) and returns `t.test(x, alternative="two.sided", mu=15)$p.value`. We could've alternatively defined the function ahead of time using the usual function definition syntax:

```
get_p_value_t_test <- function(x){
  return(t.test(x, alternative="two.sided", mu=15)$p.value)
}
```

```
p_values_alt <- purrr::map_dbl(.x=as.data.frame(samples),
                               .f=get_p_value_t_test)
identical(p_values, p_values_alt)
```

```
## [1] TRUE
```

The `identical()` function shows that we get the exact same p values using this approach. The anonymous function syntax is merely a shortcut.

How many of our p values are smaller than 0.05? In other words, how many of the 1,000 tests lead to a rejection at the 0.05 significance level?

We use an interesting feature in R known as logical coercion. Basically, logical coercion allows you to apply functions that ordinarily operate on numeric vectors (e.g. `mean()` or `sum()`) on logical vectors, where `TRUE` gets coded as 1 and `FALSE` gets coded as 0.

This allows us to write code like this:

```
sum(p_values < 0.05)
```

```
## [1] 42
```

3. Given our description of logical coercion, explain why the above line of code returns the number of p-values in the vector `p_values` that are smaller than 0.05.

We see that 42 out of 1000, or 4.2%, of the p-values were less than 0.05. This is close to 5%. Indeed, if we had infinitely many samples, the proportion would be pretty much exactly 5%.

4. What value does the code `mean(p_values < 0.05)` return? Give the numeric value and explain how to interpret it in the context of this problem.
5. Use `ggplot()` to generate an appropriate plot showing the distribution of the 1,000 p-values computed above in `p_values`.

Power

What if H_0 were not true? Let's say the bar lengths were actually uniform between 14.92 and 15.12 (for a true average length of 15.02).

We repeat the above code to estimate via simulation the probability, under this new truth, that the p-value is less than 0.05:

```
new_samples <- matrix(runif(n=1000*90, min=14.92, max=15.12), ncol=1000)
new_p_values <- purrr::map_dbl(.x=as.data.frame(new_samples),
                               .f=get_p_value_t_test)
sum(new_p_values < 0.05)
```

```
## [1] 906
```

Wow! Now 906 out of 1000, or 90.6%, of the p-values are less than 0.05. That is, the power of the t test, under this alternative, is approximately 90.6%. Equivalently, the type II error rate is 9.4%.

Effect sizes

A word of caution: statistical significance does not mean operational significance.

Suppose the true mean bar size was 15.00001 cm. Technically, H_0 is false, and with large enough sample sizes, a t-test (or any other reasonable test, like the Wilcoxon test) would reject the null most of the time.

However, this doesn't necessarily mean anyone cares. Maybe it really doesn't matter that the average bar is 0.00001cm longer. Maybe it does, depending on the application (though here I'd probably be more concerned about the variability across different bars, rather than the tiny bias in average length).

Likewise, suppose the cholesterol drug had a true lowering effect of 0.0001. That clearly doesn't warrant anybody spending the time and money to get the drug. But if you have a large enough sample, you can still reject the null hypothesis that the drug is ineffective!

Thus, many scientists and statisticians advocate for estimating *effect sizes* rather than looking for statistical significance. That is, focus your energy on better estimating the magnitude of the true effect, rather than on whether there is (some) nonzero effect.

Some ways to do this are explored in more advanced statistics courses.