Introduction to Statistical Inference (QTM 100 Lab)

Lecture 9: Inference for Paired Data and Errors in Inference

Justin Eloriaga — Emory University

Fall 2024

Gameplan

Preliminaries

Paired t-test

Errors in Inference

Preliminaries

Overview

Two broad things to consider for today

- Using paired t-tests and when to use them. (We did χ^2 and proportion tests before, what about now?)
- Assessing the performance of the confidence intervals and hypothesis tests that we do

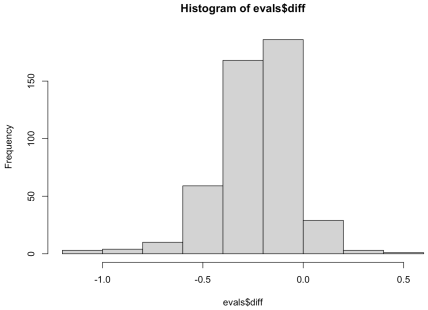
Paired t-test

- Consider the `CourseEvals.csv` dataset we used last week.
- Suppose we have this research question. **Are professor evaluations significantly different from course evaluations?**
- We are looking at two scores (numeric variables) which are likely related.
- We need to use a paired t-test!

Creating diff

We first need to calculate the difference between these two scores for each professor, and look at some stats

```
evals$diff <- evals$course_eval - evals$prof_eval
mean(evals$diff) # Average is -0.18
sd(evals$diff)
hist(evals$diff)
```



Doing a paired t-test

We would like to test the null hypothesis below

$$H_0 : \mu_{diff} = 0$$

$$H_a : \mu_{diff} \neq 0$$

You can run either of these commands

```
t.test(evals$diff)
t.test(evals$course_eval, evals$prof_eval, paired =
T)
```

```
        Paired t-test

data:  evals$course_eval and evals$prof_eval
t = -19.155, df = 462, p-value < 2.2e-16
alternative hypothesis: true mean difference is not equal to 0
95 percent confidence interval:
 -0.1945611 -0.1583547
sample estimates:
mean difference
     -0.1764579
```

Interpreting the paired t-test

- The test stat is $t = -19.16$ with 462 degrees of freedom.
- At the $\alpha = 0.05$ level of significance, we reject H_0
- We are 95% confident that the true average difference is in the interval -0.19 to -0.16 (zero is not in it)
- Average course evaluation is significantly lower than the average professor score
- Hence, professors get higher personal evaluations than course evaluations

$$\mu_{course} - \mu_{prof} < 0 \implies \mu_{course} < \mu_{prof}$$

Errors in Inference

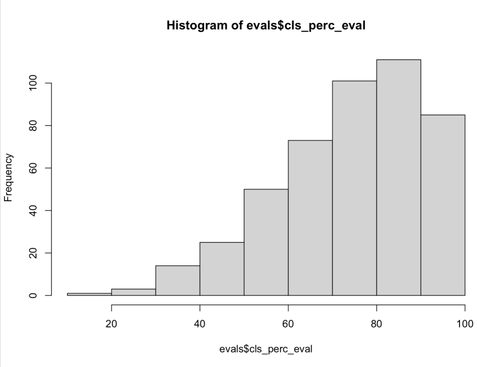
MUST DO BEFORE YOU START

We need to open and run `TestingFunctions.R` in your `R`Studio.

- You should see two functions if you ran it correctly.
  - `inference.means` which randomly selects samples from a given numerical variable and performs inference on that numerical variable
  - `plot.ci` which plots confidence intervals from an object created by `inference.means`

Recalling `cls_perc_eval`

What is the true population distribution of `cls_perc_eval`? Let' run the typical `hist()`, `mean()`, `sd()` commands on `cls_perc_eval`



Among 463 courses, the percentage completion is *left skewed* with a mean of 74.4 and a standard deviation of 16.8

Multiple Samples and Inference

- Let us see if the confidence interval actually captures the true mean value of 74.4. If it does, great! If it does not, there is an *error in estimation*
- Suppose we test the following hypothesis:

$$H_0 : \mu = 74.4$$

$$H_a : \mu \neq 74.4$$

- We risk making a Type I error (rejecting H_0 when H_0 is true). For each sample, we can determine if a Type I error was committed.

Using `inference.means`

`inference.means` has four arguments

- `variable` - numerical variable of interest
- `sample.size` - the sample size n
- `alpha` - the level of significance
- `num.reps` - the number of random samples to generate

Using `inference.means`

Suppose you want 100 samples of size $n = 50$ and you want to perform inference at the $\alpha = 0.05$ level of significance

```
sim1 <- inference.means(variable =
evals$cls_perc_eval, sample.size = 50, alpha = 0.05,
num.reps = 100)
View(sim1)
```

Note, everyone will have different results because we got random samples!

Results

	samp.est	test.stat	p.val	decision	lcl	ucl	capture
10	76.8572	1.0297	0.3082	fail to reject Ho	72.1159	81.5986	yes
11	75.2799	0.3621	0.7189	fail to reject Ho	70.5504	80.0095	yes
12	78.0435	1.6871	0.0979	fail to reject Ho	73.7367	82.3504	yes
13	76.1960	0.8550	0.3967	fail to reject Ho	72.0402	80.3518	yes
14	74.7803	0.1414	0.8881	fail to reject Ho	69.7720	79.7886	yes
15	73.9370	-0.1959	0.8455	fail to reject Ho	68.9014	78.9725	yes
16	71.6558	-1.0690	0.2903	fail to reject Ho	66.4452	76.8665	yes
17	79.9494	2.5508	0.0139	reject Ho	75.5994	84.2993	no
18	70.5537	-1.7776	0.0817	fail to reject Ho	66.1742	74.9333	yes
19	74.5750	0.0705	0.9441	fail to reject Ho	70.3780	78.7720	yes
20	70.2617	-1.4916	0.1422	fail to reject Ho	64.6492	75.8743	yes
21	77.4055	1.3883	0.1713	fail to reject Ho	73.0953	81.7156	yes

- `samp.est` is the point estimate
- `test.stat` is the t stat
- `p.val` is the p-value
- `decision` is the answer
- `lcl` is the lower CB
- `ucl` is the upper CB
- `capture` indicaes if the CI captured the true parameter $\mu = 74.4$.

Assessing Assumptions for Inference

When performing inference about a mean, we have 3 assumptions to assess

1. The data represents a random sample of the population
2. All observations are independent
3. The sampling distribution of the sample mean is approximately normally distributed

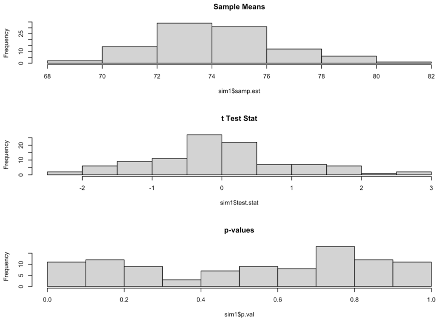
Our `inference.means()` function makes sure (1) and (2) are good, but we need to check if (3) is satisfied!!

Examining the Performance of Hypothesis Testing

Let's look more closely at the distribution of the simulation exercise we did

Run codes that generate a histogram using the `hist()` command on the `samp.est`, `test.stat` and the `p.val`.

Histogram of sample means and t-stats should be approximately **normal** while the p-value should be approximately **uniform**!



Understanding α

If you run `table(sim1$decision)`, you generate a frequency table to determine how many instances you commit a Type I error.

```
> table(sim1$decision)

fail to reject Ho         reject Ho
               95                 5
```

As you can see, the Reject H_0 should be approx 5% since $\alpha = 0.05$. In my case, it is perfectly 5% but slight variations should be expected.

Examining the Performance of the Confidence Intervals

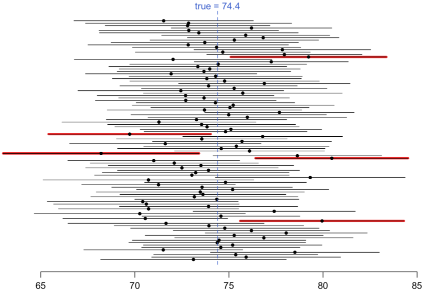
Now examine the inference results related to confidence interval estimation by visualizing the confidence intervals with the `plot.ci` function. This function takes two arguments

1. `results` — the name of the object that contains the simulation results from `inference.means`
2. `true.val` — the true value of the parameter being tested.

Generating a plot of the confidence intervals for the simulation we did using the $\mu = 74.4$ true value needs the command

```
plot.ci(results = sim1, true.val = 74.4)
```

Results from the Confidence Interval Performance Test



- Clearly, we see 5 erroneous bands, similar to what we predicted before.
- Running `table(sim1$capture, sim1$decision)` reinforces this further.

Long Run Performance

- Increasing the sample size (from 100 to say a bigger number) can give us a better idea of the long run performance of the test.
- You expect that the probability of a type 1 error remains the same or improve somewhat.