Stat 343, Thu 12-Nov-2020 -- Thu 12-Nov-2020

Thursday, November 12th 2020

Wk 11, Th

Topic:: CIs for a proportion

Inference for Proportions

We have a rule of thumb which says if

$$np \geq 10 \qquad \text{and} \qquad n(1-p) \geq 10,$$

then it is reasonable to view a proportion $\hat{\pi} = \frac{1}{n}\sum X_i$ from an i.i.d. random sample $\mathbf{X} = \langle X_1, \ldots, X_n \rangle \overset{\text{i.i.d.}}{\sim} \mathsf{Binom}(1, \pi)$ as having a sampling distribution that is well-approximated by $\mathsf{Norm}(\pi, \sqrt{\frac{\pi(1-\pi)}{n}})$ or, equivalently,

$$\frac{\hat{\pi} - \pi}{\sqrt{\frac{\pi(1-\pi)}{n}}} \sim \mathsf{Norm}(0, 1).$$

Confidence interval construction

We wish to estimate the unknown proportion π in a population via a confidence interval, using the sample proportion $\hat{\pi}$ from an i.i.d.random sample as estimator. We discuss multiple ways for constructing this confidence interval and compare coverage rates from simulations against the advertised rate.

1. **Wald method**.
   - Used once already in a class example
   - Takes as its estimate of standard error $\text{SE}_{\hat{\pi}} = \sqrt{\dfrac{\hat{\pi}(1-\hat{\pi})}{n}}$
   - For a Level (100C)% confidence interval, takes $\alpha = 1 - C$ and $z = z_{\alpha/2} = \Phi^{-1}(1 - \alpha/2)$.
   - Bounds come from
   $$\hat{\pi} \pm z_{\alpha/2}\sqrt{\frac{\hat{\pi}(1-\hat{\pi})}{n}}.$$

   The coverage rate does not match well the advertised rate.

   The `binom.test()` function, as augmented by the Mosaic package, can be induced to give Wald confidence intervals for π using the switch `ci.method="Wald"`.

2. **Score method**. If we conducted an hypothesis test for hypotheses

   $$\mathbf{H}_0:\ \pi = \pi_0 \qquad \text{vs.} \qquad \mathbf{H}_a:\ \pi \neq \pi_0,$$

   our test statistic under the normal approximation would be

   $$z \ = \ \frac{\hat{\pi} - \pi_0}{\sqrt{\frac{\pi_0(1-\pi_0)}{n}}}.$$

   For $\alpha = 1 - C$, the equation

   $$z_{\alpha/2} \ = \ \frac{|\hat{\pi} - \pi_0|}{\sqrt{\frac{\pi_0(1-\pi_0)}{n}}}$$

is quadratic in π_0, and produces two solutions, ones that place $\hat{\pi}$ right at the boundary between the rejection and non-rejection regions:

$$\pi_0 \;=\; \frac{\hat{\pi} + \frac{z_*}{2n} \pm z_* \sqrt{\frac{\hat{\pi}(1-\hat{\pi})}{n} + \frac{z_*^2}{4n^2}}}{1 + \frac{z_*^2}{n}}.$$

The score method uses these as the endpoints of a level $(100 \times C)\%$ confidence interval. The resulting interval is seldom symmetric about the point estimate $\hat{\pi}$.

The `prop.test()` function constructs intervals using the Score method. The coverage rate is pretty close to that advertised.

3. **Plus 4 method**. In this approach, we replace the estimator $\hat{\pi}$ above with

$$\hat{\pi} \;=\; \frac{X+2}{n+4},$$

effectively acting like the sample is 4 larger than it truly is, with two of those four being "successes". This method arises naturally from the Score method in the case of $C = 0.95$ (95% confidence) by letting $n \to \infty$.

Using this $\hat{\pi}$ and z_* obtained as in the other methods, we take as our confidence interval bounds

$$\hat{\pi} \pm z_* \sqrt{\frac{\hat{\pi}(1-\hat{\pi})}{n+4}}.$$

4. **Clopper-Pearson method**. Not a lot of details here, but it is the default method used by `binom.test()`.

The `mosaic` package in R provides a way of empirically testing coverage rates through the `CIsim()` command. A lot of examples of its use are in the textbook. Some points of interest:

- The command requires
  - `n`: a sample size
  - `rdist`: function used to draw samples
  - `args`: to provide a list of parameters to the function specified in `rdist`. Since all examples of such functions require parameters, such as

    `rnorm`: mean=____, sd=____
    `rexp()`: rate=____
    `rbinom()`: size=____, p=____

    `args` provides a way to send those parameters.
  - `samples`: number of iterations a sample of size `n` is to be drawn
  - `estimand`: the target mean which each confidence interval from a sample is assessed to see if contained therein

  - `method`: the function to use in the building of confidence intervals from samples. The default here is `t.test()`. The pertinent value in the return list from this function is named `conf.int`. In section 4.5, Pruim gave us a different function, `zci()`, to stand in as a confidence interval constructor under the pretense that σ was known.

- If the number `samples` is low enough (perhaps 100(?) or less), the command produces a plot visually depicting the confidence intervals, coloring them by whether they contain the target mean or not.