Visualizing Errors in Hypothesis Testing

November 19, 2021

1 Visualizing Errors in Hypothesis Testing

In Lesson 6 of Module 1, we derived a hypothesis test for

$$H_0 : \mu = \mu_0$$

$$H_1 : \mu = \mu_1$$

for a random sample of size n from the $N(\mu, \sigma^2)$ distribution with σ^2 known.

We assumed that $\mu_0 < \mu_1$ and our test was to

$$\text{Reject } H_0, \text{ in favor of } H_1 \text{ if } \overline{X} > \mu_0 + z_\alpha \frac{\sigma}{\sqrt{n}}$$

.

Suppose that $n = 10$, $\mu_0 = 2$, $\mu_1 = 3$, $\sigma^2 = 5$, and $\alpha = 0.05$. Enter these values in the next cell. Call them mu0, mu1, sigmasq, and alpha.

For example, type `mu0<-2`. Hit Enter to get to the next line and contine until they are all entered. Hit Shift+Enter to run the cell.

[4]:
```
n = 10
mu0 = 2
mu1 = 3
sigmasq = 5
alpha = 0.05
```

Recall that the critical value z_α is the number that cuts off area 0.05 to the right under the standard normal curve. In the next cell, type

`cv<-qnorm(1-alpha)`

Before running the cell, we will compute the cutoff value $\mu_0 + z_\alpha \frac{\sigma}{\sqrt{n}}$. In the lecture videos, we called this value "c". In R, "c",followed by "()" is a function that is used to create vectors. We can also define a variable "c", but to avoid any problems with protected symbols, let us call the cutoff "cee".

Add the following to the cell below and run the cell.

`cee<-mu0+cv*sqrt(sigmasq)/sqrt(n)`

[6]:
```
cv<-qnorm(1-alpha)
cee<-mu0+cv*sqrt(sigmasq)/sqrt(n)
cee
```

3.16308715367667

In the cell above, type `cee` in the third line and run the cell. This will show you the cutoff you have computed.

If the null hypothesis is true, the mean of the normal distribution we are working with is 2. The $N(2, 5)$ distribution will certainly produce values above 3. This cutoff "cee" tells us what value above 2 is large enough for us to start believing that the true mean is no longer 2 but something larger.

Let's simulate 10 values from the normal distribution with mean 2 and variance 5 and putting the results in a vector called "my sample". We can do this by simulating ten N(0,1) random variables, multiplying by the desired standard deviation and adding the desired mean.

In the cell below, type

`mysample<-sqrt(5)*rnorm(10)+2`

Alternatively you could simulate values directly from the desired distribution by typing

`mysample<-rnorm(10,2,sqrt(5))`

After either command, run the cell by holding down the Shift key and hitting Enter.

[7]:
```
mysample<-sqrt(5)*rnorm(10)+2
mysample<-rnorm(10,2,sqrt(5))
```

What is the sample mean for your sample? In the next cell, type

`mean(mysample)`

and run the cell.

[8]:
```
mean(mysample)
```

2.1991965664012

If your result is below your value of cee. You will fail to reject the null hypothesis. You are concluding that the sample did indeed come from the normal distribution with mean 2, which it did.

On the other hand, if your result is above your value of cee, you will reject the null hypothesis. Your sample did, in fact, come from the normal distribution with mean 2 but you are concluding that the mean is actually higher than 2. You have made a Type I Error.

What happened in your case?

If we repeat our little experiment many times, we should see that will are making the Type I Error roughly 5% of the time. Let's try it!

We will repeat our experiment 1,000 times. In the next cell, type

results<-rep(0,1000)

for(i in 1:1000){

mysample<-sqrt(5)*rnorm(10)+2

}

Indentation is not necessary in R but we include it for clarity. "results" is a vector of 1,000 zeros that will indicate the result for each sample.

Underneath the line where you produce a sample, but before the closing brackets on the "for statement", we will check whether or not we will reject H_0. In there, type

```
if(mean(mysample)>cee){

    results[i]<-1

}
```

and then run the entire code by running the cell.

[15]:
```
results<-rep(0,1000)

for( i in 1:1000){

      mysample<-sqrt(5)*rnorm(10)+2
      if(mean(mysample)>cee){

      results[i]<-1

    }

}
```

The "results" vector is filled with 0's and 1's. The 1's occur every time we made a Type I error. The proportion of 1's can be gotten by summing everything in the vector and dividing by 1,000.

```
sum(results)/1000
```

[16]:
```
sum(results)/1000
```

0.055

DId you see something close to 0.05? Just for fun, run both of the last cells again. When you are done, try changing all instances of 1,000 to 10,000. You will, in fact, make a Type I error 5% of the time, but you'll need a lot of trials to make that number come in to focus! This is no different than having a coin with a 50-50 chance of coming up "Heads" when flipped. If you flip it 10 times, you might see 6 "Heads" and 4 "Tails". That doesn't mean that the probability of getting "Heads" is 0.60. You just need more data to see it!

[ ]: