Estimation and Hypothesis Testing With a Single Variable

Babak Shahbaba and Sam Behseta
UC Irvine and CSU Fullerton

July and August, 2022

The Magic of Normal Distribution

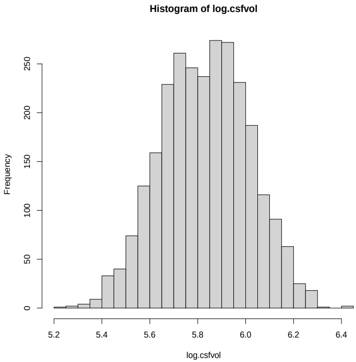
- Let's study some estimation problems with R.

The Magic of Normal Distribution

- Let's study some estimation problems with R.
- Recall from one of our in-class activities that the distribution of the logged values of *csfvol* was fairly symmetric. Let's call that feature *log.csfvol*.

The Magic of Normal Distribution

- Let's study some estimation problems with R.
- Recall from one of our in-class activities that the distribution of the logged values of *csfvol* was fairly symmetric. Let's call that feature *log.csfvol*.
- Below is the histogram of *log.csfvol*:



The Magic of Normal Distribution

- Suppose the underlying distribution that might have generated *log.csfvol* is normal with the mean $\mu = 5.83$, and standard deviation $sd = 0.18$.

The Magic of Normal Distribution

- Suppose the underlying distribution that might have generated *log.csfvol* is normal with the mean $\mu = 5.83$, and standard deviation $sd = 0.18$.
- With your teammates, analyze the code below:

```
simulations=matrix(0,nrow=1000,ncol=1000)

for(i in 1:1000)
{
  simulations[i,]<-rnorm(1000,5.83,0.18)
}

sample.mean<-numeric(1000)
sample.mean<-apply(simulations,1,mean)
hist(sample.mean,breaks=15)

mean(sample.mean)
sd(sample.mean)
0.18/sqrt(1000)
```

The Magic of Normal Distribution

- Suppose the underlying distribution that might have generated *log.csfvol* is normal with the mean $\mu = 5.83$, and standard deviation $sd = 0.18$.
- With your teammates, analyze the code below:

```
simulations=matrix(0,nrow=1000,ncol=1000)

for(i in 1:1000)
{
  simulations[i,]<-rnorm(1000,5.83,0.18)
}

sample.mean<-numeric(1000)
sample.mean<-apply(simulations,1,mean)
hist(sample.mean,breaks=15)

mean(sample.mean)
sd(sample.mean)
0.18/sqrt(1000)
```

- We just verified the Central Limit Theorem (CLT) via R!

The Magic of Normal Distribution

- Here is the code for creating the histogram of the sample mean:

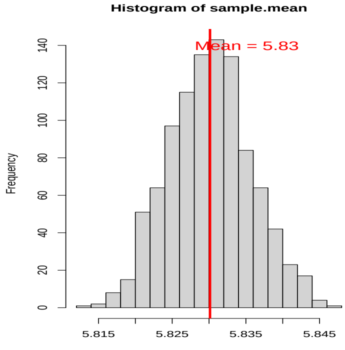
```
hist(sample.mean,breaks=15)
abline(v = mean.xbar,col = "red", lwd = 3)
text(x =mean.xbar+0.005 , y = mean.xbar*24,
     paste("Mean =",5.83),col ="red", cex = 1.4)
```

The Magic of Normal Distribution

- Here is the code for creating the histogram of the sample mean:

```
hist(sample.mean,breaks=15)
abline(v = mean.xbar,col = "red", lwd = 3)
text(x =mean.xbar+0.005 , y = mean.xbar*24,
     paste("Mean =",5.83),col ="red", cex = 1.4)
```

- Here is the histogram of the sample mean, reflecting its distribution:



The Classical Confidence Interval for the Population Mean

- Let's revisit what we learn in the earlier lecture, using the Alzheimer's data.

The Classical Confidence Interval for the Population Mean

- Let's revisit what we learn in the earlier lecture, using the Alzheimer's data.
- Suppose we are interested in building a 95% confidence interval for the mean of *naccicv* or *vol*.

The Classical Confidence Interval for the Population Mean

- Let's revisit what we learn in the earlier lecture, using the Alzheimer's data.
- Suppose we are interested in building a 95% confidence interval for the mean of *naccicv* or *vol*.
- Suppose the 2700 data points in our data set form a population, and thus we know the mean and standard deviation of *naccicv*. The mean and standard deviation of the population happen to be $\mu = 1376.9$ and $\sigma = 134.8$, respectively.

The Classical Confidence Interval for the Population Mean

- Let's revisit what we learn in the earlier lecture, using the Alzheimer's data.
- Suppose we are interested in building a 95% confidence interval for the mean of *naccicv* or *vol*.
- Suppose the 2700 data points in our data set form a population, and thus we know the mean and standard deviation of *naccicv*. The mean and standard deviation of the population happen to be $\mu = 1376.9$ and $\sigma = 134.8$, respectively.
- Our aim is to demonstrate the idea behind the 95% confidence interval for μ.

The Classical Confidence Interval for the Population Mean

- Carefully analyze the code below and the graph resulting from it.

```
z_star_95 <- qnorm(0.975)
z_star_95

library(statsr)
library(dplyr)
library(ggplot2)

alz<-alzheimer_data
attach(alz)

n <- 100
samp.alz <- sample_n(alz, n)
dim(samp.alz)

samp.alz %>%
  summarise(lower = mean(vol) - z_star_95 * (sd(vol) / sqrt(n)),
            upper = mean(vol) + z_star_95 * (sd(vol) / sqrt(n)))


params <- alz %>%
  summarise(mu = mean(vol))
params
```

The Classical Confidence Interval for the Population Mean

- ```
  ci <- alz %>%
    rep_sample_n(size = n, reps = 100, replace = TRUE) %>%
  summarise(lower = mean(naccicv) - z_star_95 * (sd(naccicv) / sqrt(n)),
              upper = mean(naccicv) + z_star_95 * (sd(naccicv) / sqrt(n)))

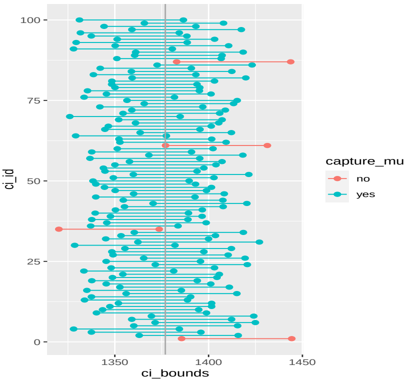
  ci %>%
    slice(1:10)

  ci <- ci %>%
    mutate(capture_mu = ifelse(lower < params$mu & upper > params$mu, "yes", "no"))

  ci_data <- data.frame(ci_id = c(1:100, 1:100),
                        ci_bounds = c(ci$lower, ci$upper),
                        capture_mu = c(ci$capture_mu, ci$capture_mu))


  ggplot(data = ci_data, aes(x = ci_bounds, y = ci_id,
                             group = ci_id, color = capture_mu)) +
    geom_point(size = 2) +  # add points at the ends, size = 2
    geom_line() +           # connect with lines
    geom_vline(xintercept = params$mu, color = "darkgray") # draw vertical line
  ```
  ```

The Classical Confidence Interval for the Population Mean



- Let's begin with a viable hypothesis.

```
> vol.sample=sample(vol,100,replace=FALSE)
> t.test(vol.sample,mu=1376,alternative="two.sided",conf.level=0.95)
One Sample t-test
data:  vol.sample
t = 0.022598, df = 99, p-value = 0.982
alternative hypothesis: true mean is not equal to 1376
95 percent confidence interval:
 1348.383 1404.253
sample estimates:
mean of x
 1376.318
```

- Let's begin with a viable hypothesis.

```
> vol.sample=sample(vol,100,replace=FALSE)
> t.test(vol.sample,mu=1376,alternative="two.sided",conf.level=0.95)
One Sample t-test
data:  vol.sample
t = 0.022598, df = 99, p-value = 0.982
alternative hypothesis: true mean is not equal to 1376
95 percent confidence interval:
 1348.383 1404.253
sample estimates:
mean of x
 1376.318
```

- Let's consider a less-reliable hypothesis.

```
t.test(vol.sample,mu=1340,alternative="two.sided",conf.level=0.95)
```

- Let's begin with a viable hypothesis.

```
> vol.sample=sample(vol,100,replace=FALSE)
> t.test(vol.sample,mu=1376,alternative="two.sided",conf.level=0.95)
One Sample t-test
data:  vol.sample
t = 0.022598, df = 99, p-value = 0.982
alternative hypothesis: true mean is not equal to 1376
95 percent confidence interval:
 1348.383 1404.253
sample estimates:
mean of x
 1376.318
```

- Let's consider a less-reliable hypothesis.

```
t.test(vol.sample,mu=1340,alternative="two.sided",conf.level=0.95)
```

- The concept of reliability is tightly related to *prior knowledge* about the problem. As such, *Bayesian statistical methods* would offer a more meaningful, and often more accurate, solutions to the problem of point estimation.

- Or even better, we can write our own function. In the function below, *dat* is a vector and *conf.level* represents the confidence level, a number between 0 to 1.

```
my.confidence<-function(dat,conf.level)
{
  n<-length(dat)
   z_star_cl<-qnorm(1-(1-conf.level)/2)
  lower<- mean(vol) - z_star_cl * (sd(dat) / sqrt(n))
  upper<-mean(vol) + z_star_cl *(sd(dat) /sqrt(n))
            ci<-c(lower,upper)
            return(ci)
  }

my.confidence(vol.sample,0.95)
```

- Here, μ is the proportion of the characteristic of interest in the population. Therefor, $1 - \mu$ would represent the proportion of the alternative outcome. We can use a readily available command in R or write our own function for hypothesis testing and confidence intervals associated with μ. Let's demonstrate both through the feature *female* from the Alzheimer's data.

- Readily available command in R:

```
female<-as.factor(female)
fem.counts<-summary(female)[2]
prop.test(fem.counts,2700)
```

- Readily available command in R:

```
female<-as.factor(female)
fem.counts<-summary(female)[2]
prop.test(fem.counts,2700)
```

- Or write your own code:

```
my.prop.confidence<-function(counts,tot,conf.level)
{
  z_star_cl<-qnorm(1-(1-conf.level)/2)
  p<-counts/tot
  se.dat<-sqrt((p*(1-p))/tot)
  lower<- p - z_star_cl * se.dat
  upper<- p + z_star_cl * se.dat
  ci<-c(lower,upper)
  return(ci)
}

my.prop.confidence(1549,2700,0.95)
```

- Consider the variable *diagnosis*. Create a new variable from *diagnosis* with only two categories: whether the subject has diagnosed with Alzheimer's (1) or not (0).

- Consider the variable *diagnosis*. Create a new variable from *diagnosis* with only two categories: whether the subject has diagnosed with Alzheimer's (1) or not (0).
- Test the hypothesis that less than 45% of subjects in the population are diagnosed with Alzheimer's.

Activity

- Consider the variable *diagnosis*. Create a new variable from *diagnosis* with only two categories: whether the subject has diagnosed with Alzheimer's (1) or not (0).
- Test the hypothesis that less than 45% of subjects in the population are diagnosed with Alzheimer's.
- Based on our working data, construct a 95% confidence interval for the proportion of subjects in the population, diagnosed with Alzheimer's.