Stat 243 – takeHome

Sharon Velpula

March 15, 2022

1.

(a) No, because we constructed a 95% confidence interval. The t star value in this case is 2.45 and want to find values that are within 2 standard deviations pf the mean.

(b) qnorm(0.975)

(c)

```
isInside = function(param, ciBounds) {
  ciBounds[1] <= param & param <= ciBounds[2]
}
enn <- 7
#tStar <- qt(0.975, df = enn - 1)
zStar <- qnorm(0.975)
runResults <- do(10000) * {
  sampledVals <- rnorm(enn, 70, 3)
  xbar <- mean(~sampledVals)
  ess <- sd(~sampledVals)
  tStat <- (xbar - 70)/(ess/sqrt(enn))
  myCI <- xbar + c(-1, 1) * zStar * ess/sqrt(enn)
  isInside(70, myCI)
}
prop(runResults == TRUE) # the effective coverage rate of the simulation
```

```
## prop_TRUE
##    0.8996
```

From running several times, I have seen that it does not ascend to 95% because we are using a z*, which in this case no longer is the standard normal percentile.

(d)

```
isInside = function(param, ciBounds) {
ciBounds[1] <= param & param <= ciBounds[2]
}
enn <- 7
tStar <- qt(0.975, df = enn - 1)
runResults <- do(10000) * {
  #sampledVals <- rnorm(enn, 70, 3)
  sampledVals <- rexp(enn, 0.02)
  xbar <- mean(~sampledVals)
  ess <- sd(~sampledVals)
  myCI <- xbar + c(-1, 1) * tStar * ess/sqrt(enn)
  isInside(70, myCI)
}
```

```
prop(runResults == TRUE) # the effective coverage rate of the simulation
```

```
## prop_TRUE
##    0.6873
```

Since we use rexp, it results in large standard deviations and unpredictable sample means, which are used in the computation even with a t* value.

(e)

```
isInside = function(param, ciBounds) {
ciBounds[1] <= param & param <= ciBounds[2]
}
enn <- 35
tStar <- qt(0.975, df = enn - 1)
runResults <- do(10000) * {
  sampledVals <- rexp(enn, 0.02)
  xbar <- mean(~sampledVals)
  ess <- sd(~sampledVals)
  myCI <- xbar + c(-1, 1) * tStar * ess/sqrt(enn)
  isInside(70, myCI)
}
prop(runResults == TRUE) # the effective coverage rate of the simulation
```

```
## prop_TRUE
##    0.3669
```

prop_TRUE is very low in comparison to the other sections. This is because we have more values in the sample, which leads to a shorter CI with values below the given population mean, 70.

(f)

```
isInside = function(param, ciBounds) {
  ciBounds[1] <= param & param <= ciBounds[2]
}
enn <- 7
tStar <- qt(0.95, df = enn - 1)
runResults <- do(10000) * {
  sampledVals <- rnorm(enn, 70, 3)
  xbar <- mean(~sampledVals)
  ess <- sd(~sampledVals)
  tStat <- (xbar - 70)/(ess/sqrt(enn))
  myCI <- xbar + c(-1, 1) * tStar * ess/sqrt(enn)
  isInside(70, myCI)
}
prop(runResults == TRUE) # the effective coverage rate of the simulation
```

```
## prop_TRUE
##    0.9051
```

Yes, it is approx. 90 which is what we expect since we constructed a 90% CI.

2.

(a)

```
#Generating a bootstrap sample
Yes <- filter(NHANES, SleepTrouble == "Yes")
No <- filter(NHANES, SleepTrouble == "No")
```

```
#Bootstrap statistic found by the difference between sample means of Weight
diff <- mean(~Weight, data = Yes, na.rm = TRUE) -
  mean(~Weight, data = No, na.rm = TRUE)

#Repeating many trials of the above
manyDiffs <- do(5000) * (mean(~Weight, data = resample(Yes), na.rm = TRUE) -
                           mean(~Weight, data = resample(No), na.rm = TRUE))

#Finding the standard deviation of the resulting bootstrap distribution
std <- sd(~result, data = manyDiffs)

#Constructing 95% CI using bootstrap percentiles
qdata(~result, data = manyDiffs, c(0.025, 0.975))
```

```
##      2.5%     97.5%
## 0.8741337 3.1453345
```

(b)

1. Let us assume we have one big bag called POPULATION containing slips of paper with details of a person (corresponding to columns in NHANES like weight, gender , etc).
2. We bring two other bags called Sample1 and Sample2. We pick out slips from the POPULATION bag and separate them into the bags SAMPLE1 and SAMPLE2 based on the people's answers to whether or not they have trouble sleeping. Let Sample1 be the bag with slips of people who answered Yes and Sample2 with those slips who answered No.
3. From Sample1, take out each slip and note down the person's weight, as well as counting the number of slips being taken out. Now, add all the weights and divide them by the number of slips. This will give you a mean, which we will name Mean1.
4. Repeat the same for Sample2 and name the mean, Mean2.
5. Subtract both the means, Mean1 - Mean2 and that gives us a single bootstrap statistic.

(c)

```
Yes <- filter(NHANES, SleepTrouble == "Yes")
No <- filter(NHANES, SleepTrouble == "No")

diff <- mean(~Weight, data = Yes, na.rm = TRUE) - mean(~Weight, data = No,
                                                       na.rm = TRUE)

std1 <- sd(~Weight, data = Yes, na.rm = TRUE)
std2 <- sd(~Weight, data = No, na.rm = TRUE)

#Used favstats to find the n values for both data sets
enn1 <- 5751
enn2 <- 1955

stderr <- sqrt((std1^2/enn1) + (std2^2/enn2))
dof <- 3226.58 #Used Welch's formula and manually calculated it

tStar <- qt(0.975, df = dof)

lower <- (diff - (tStar*stderr))
upper <- (diff + (tStar*stderr))
```

```
lower
```

```
## [1] 0.8931643
```

```
upper
```

```
## [1] 3.076975
```

We see that the CI obtained here is similar to that of (a).

3.

(a) H0: population mean (Yes) - population mean (No) = 0 OR muY - muN = 0 Ha: population mean (Yes) - population mean (No) != 0 OR muY - muN != 0

```
#Computing the randomization test statistic
tStat <- diffmean(Pulse ~ SleepTrouble, data = NHANES, na.rm=TRUE)

#Repeating a random process multiple times
manyDiffs <- do(5000) * diffmean(Pulse ~ shuffle(SleepTrouble), data =
                                     NHANES, na.rm = TRUE)

#Finding the pvalue which is basically a portion of the generated distribution
pvalue <- nrow(filter(manyDiffs, abs(diffmean) >= tStat))/5000

pvalue
```

```
## [1] 0.009
```

The pvalue that we got is close to 0 which is almost always less than any significant level that could be defined. This means we have sufficient evidence to reject the null hypothesis, meaning, there is a difference in the mean hours of sleep for people who have trouble sleeping and for those who do not have trouble sleeping. This makes sense because, those who do not have trouble sleeping would probably sleep consistently with a good amount of hours and have a better pulse rate while those who do have trouble sleeping might sleep inconsistently with varied hours, thereby affecting their health, which may be reflected by their pulse rate. The sample results are statistically significant.

(b) We assume that the null hypothesis is true and that there is no difference in the means of pulse of those with no sleep trouble and with sleep trouble. Assuming we have a bag (sample) full of slips of paper with details of a person on each slip, distinguished by their responses to SleepTrouble. We now shuffle the slips of paper so that the responses are mixed up and we divide the sample into two separate samples, bag1 and bag2 with equal number of paper slips in each. Now we find the means of pulse for each bag1 and bag2, and find the difference between them. This is our randomization statistic.

(c) I found df using favstats to find for pulse, the standard deviations and size of the sample, grouped by the SleepTrouble variable. Then I applied the conservative method and calculated it.

I found the SE using this formula, sqrt((s1ˆ2)/n1 + (s2ˆ2)/n2).

```
SE <- 0.323
#dof <- 3201.09
dof <- 1913

meandiff <- diffmean(Pulse ~ SleepTrouble, data = NHANES, na.rm=TRUE)

tStat <- meandiff / SE

pvalue <- 2*(1-pt(tStat, df=dof))
```

```
pvalue
```

```
##    diffmean
## 0.005932752
```

The pvalues for both (a) and (c) are close to 0 and/or less than 0.05, which means that we have enough evidence to reject the null hypothesis and say that there is a difference between the means of pulse rates for those who have trouble sleeping and those who do not. The sample results are statistically significant.