Ex5

Yelyzaveta Klysa

2023-10-15

Contents

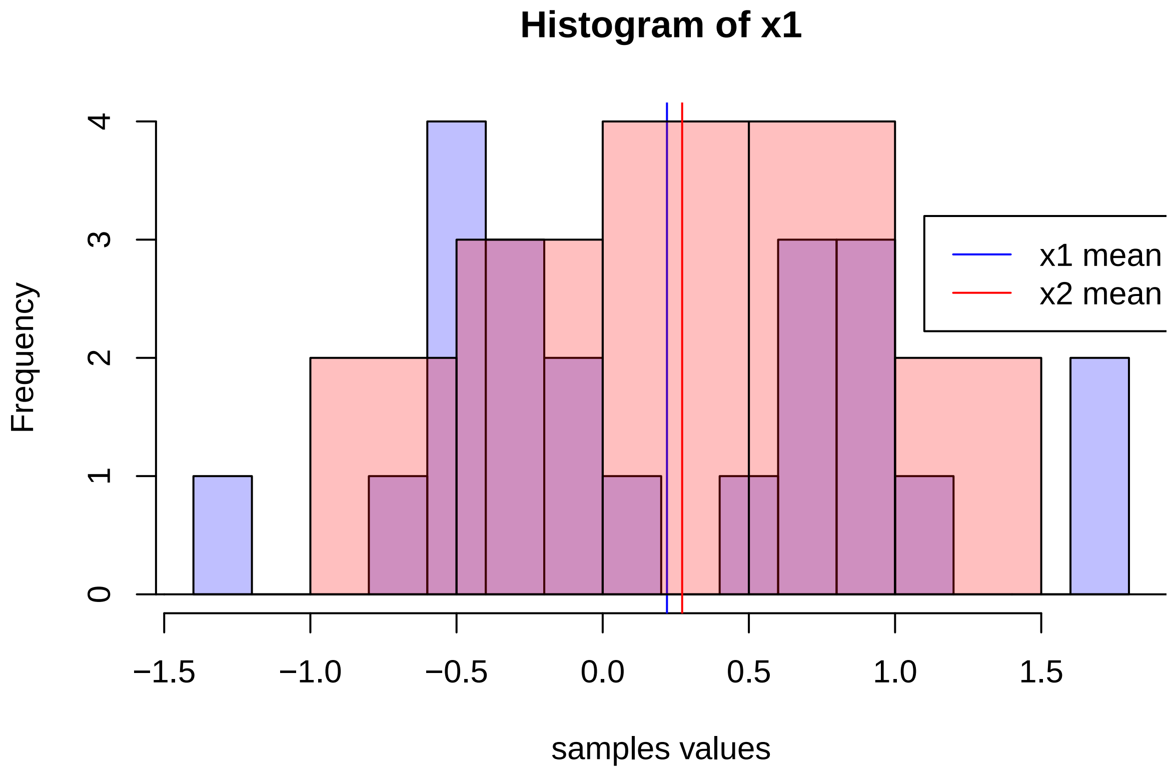
Task 1

Consider a two sample problem and the hypothesis H0:mean1=mean2 vs H1:mean1 is not equal to mean2 , where mean1 and mean2 are the corresponding sample locations. The two samples are:

```
library("boot")
set.seed(12208877)
x1 <- c(-0.673, -0.584, 0.572, -0.341, -0.218, 0.603, -0.415, -0.013, 0.763, 0.804, 0.054, 1.746, -0.47
x2 <- c(0.913, -0.639, 2.99, -5.004, 3.118, 0.1, 1.128, 0.579, 0.32, -0.488, -0.994, -0.212, 0.413, 1.4
```

1.1 Plot the data in a way which visualises the comparison of means appropriately.

```
hist(x1, col=rgb(0,0,1,1/4), breaks=15, xlab="samples values")
abline(v=mean(x1), col="blue")
hist(x2, col=rgb(1,0,0,1/4), breaks=15, add=T)
abline(v=mean(x2), col="red")
legend(1.1, 3.2, legend=c("x1 mean", "x2 mean"), col=c("blue", "red"), lty=1)
```



1.2 Consider different sampling schemes, such as

```
Sampling with replacement from each group
Centering both samples and then resample from the combined samples x1 and x2 for n1 and n2 times.
```

Argue for choice what is more natural and which advantages or disadvantages may apply.

```
sample_replacement <- function(x, n) {
  return (replicate(n, sample(x, length(x), TRUE)))
}

sample_centered <- function(x1, x2, n){
  combined <- c(x1 - mean(x1), x2 - mean(x2))
  sample1 <- replicate(n, sample(combined, length(x1), TRUE))
  sample2 <- replicate(n, sample(combined, length(x2), TRUE))
  return (list("sample_x1"=sample1, "sample_x2"=sample2))
}
```

The first option is a simple bootstrapping, which can be considered as a default choice. We can create samples and create the test statistics. It will provide more differences between the samples. In the second case, the obtained data will depict the conditions under H0 better, since the samples will consist of combined data. The second method has its advantage, when the size of samples are extremely different.

1.3 Bootstrap using both strategies mentioned above using the t-test statistic. Calculate the bootstrap p-value based on 10000 bootstrap samples and 0.95 as well as 0.99 confidence intervals. Make your decision at the significance level 0.05 or 0.01, respectively.

```r
n=10000
repl.x1 = sample_replacement(x1, n=n)
repl.x2 = sample_replacement(x1, n=n)

centr <-  sample_centered(x1, x2, n=n)

t_value <- t.test(x1,x2)$statistic

tests_1 <- numeric(n)
tests_2 <- numeric(n)

count1 <- 0
count2 <- 0

for (i in 1:n){
  tests_1[i] <- t.test(repl.x1[,i], repl.x2[,i])$statistic
  if(abs(tests_1[i]) > abs(t_value)) {
    count1 = count1 + 1
  }

  tests_2[i] <- t.test(centr$sample_x1[,i], centr$sample_x2[,i])$statistic
  if(abs(tests_2[i]) > abs(t_value)) {
    count2 = count2 + 1
  }
}

result = data.frame(counts = (c(count1, count2)),
                    method = c("Separate","Combined"),
                    CIUp95 = c(quantile(tests_1, 0.975), quantile(tests_2, 0.975)),
                    CILw95 = c(quantile(tests_1, 0.025), quantile(tests_2, 0.025)),
                    CIUp99 = c(quantile(tests_1, 0.995), quantile(tests_2, 0.995)),
                    CILw99 = c(quantile(tests_1, 0.005), quantile(tests_2, 0.005)),
                    p_value=c((count1+1)/(n+1), (count2+1)/(n+1)))
result
```

```
##   counts   method   CIUp95    CILw95   CIUp99    CILw99   p_value
## 1   9078 Separate 2.035584 -2.004819 2.679532 -2.741871 0.9078092
## 2   9094 Combined 1.960982 -1.981315 2.566784 -2.579863 0.9094091
```

We cannot reject the null hypothesis due to the p-value being 0.904 and 0.912 respectively. The boundaries of confidence intervals for 2 methods are very similar, though a second one is a bit shifted lower, most probably due to the combination of the values in creating samples.

1.4 What would be a permutation version of the test? Implement the corresponding permutation test and obtain p-value and confidence intervals as in 3. to get a corresponding test decision at the same significance levels.

```
sample_permuted <- function(x1, x2, n) {
  combined <- c(x1,x2)
  perm_x <- replicate(n, sample(combined, length(x1)+length(x2), F))
  sample_1 <- perm_x[1:length(x1),]
  sample_2 <- perm_x[(length(x1)+1):(length(x1)+length(x2)),]
  return (list("sample_x1"=sample_1, "sample_x2"=sample_2))
}

perm <-  sample_permuted(x1, x2, n=n)
tests_p <- numeric(n)
countp <- 0

for (i in 1:n){
  tests_p[i] <- t.test(perm$sample_x1[,i], perm$sample_x2[,i])$statistic
  if(abs(tests_p[i]) > abs(t_value)) {
    countp = countp + 1
  }
}

result = data.frame(counts = (c(countp)),
                    method = c("Permuted"),
                    CIUp95 = c(quantile(tests_p, 0.975)),
                    CILw95 = c(quantile(tests_p, 0.025)),
                    CIUp99 = c(quantile(tests_p, 0.995)),
                    CILw99 = c(quantile(tests_p, 0.005)),
                    p_value=c((countp+1)/(n+1)))
result
```

```
##       counts   method   CIUp95    CILw95   CIUp99    CILw99   p_value
## 97.5%   9122 Permuted 1.916963 -1.948664 2.450887 -2.464605 0.9122088
```

Based on the obtained high p-value, we once again cannot reject H0.

1.5 The Wilxocon rank sum test statistic is the sum of ranks of the observations of sample 1 computed in the combined sample. Use bootstrapping with both strategies mentioned above and obtain p-value and confidence intervals as in 3. to get a corresponding test decision at the same significance levels.

```
w_value <- wilcox.test(x1,x2)$statistic
count1 <- 0
count2 <- 0
count3 <- 0
w_values_1 <- numeric(n)
w_values_2 <- numeric(n)
w_values_3 <- numeric(n)

for (i in 1:n) {
  w_values_1[i] <- wilcox.test(repl.x1[,i], repl.x2[,i], exact=F)$statistic
  if(abs(w_values_1[i]) > abs(w_value)) {
    count1 = count1 + 1
  }
  w_values_2[i] <- wilcox.test(centr$sample_x1[,i], centr$sample_x2[,i], exact=F)$statistic
  if(abs(w_values_2[i]) > abs(w_value)) {
    count2 = count2 + 1
  }
  w_values_3[i] <- wilcox.test(perm$sample_x1[,i], perm$sample_x2[,i], exact=F)$statistic
  if(abs(w_values_3[i]) > abs(w_value)) {
    count3 = count3 + 1
  }
}

result = data.frame(counts = (c(count1, count2, count3)),
                    method = c("Separate","Combined", "Permuted"),
                    CIUp95 = c(quantile(w_values_1, 0.975),
                               quantile(w_values_2, 0.975),
                               quantile(w_values_3, 0.975)),
                    CILw95 = c(quantile(w_values_1, 0.025),
                               quantile(w_values_2, 0.025),
                               quantile(w_values_3, 0.025)),
                    CIUp99 = c(quantile(w_values_1, 0.995),
                               quantile(w_values_2, 0.995),
                               quantile(w_values_3, 0.995)),
                    CILw99 = c(quantile(w_values_1, 0.005),
                               quantile(w_values_2, 0.005),
                               quantile(w_values_3, 0.005)),
                    p_value=c((count1+1)/(n+1), (count2+1)/(n+1), (count3+1)/(n+1)))
result
```

```
##   counts   method   CIUp95   CILw95 CIUp99 CILw99   p_value
## 1   9225 Separate 325.5125 158.4875  347.5    132 0.9225077
## 2   6681 Combined 269.5125 125.5000  292.5    106 0.6681332
## 3   6660 Permuted 269.0000 127.0000  289.0    105 0.6660334
```

The p-values from combined and permuted strategies are quite similar, while the separate one gives quite high p-value that is surprisingly very different from the other two.

1.6 Compare your results to the results using t.test and wilcox.test.

Based on the results above, we can conclude that p-values for combined and permuted strategies are much lower with wilcox test than compared to t-test. However, they are still relatively high, that we can not reject the null hypothesis.

Task 2

Consider the model y=3+2*x1+x2+eps where x1 comes from a normal distribution with mean 2 and variance 3, x2 comes from a uniform distribution between 2 and 4 and eps from a student's t distribution with 5 degrees of freedom . In addition, there is a predictor x3 coming from a uniform distribution between -2 and 2.

2.1 Create a sample of size 200 from the model above and for the independent predictor x3 .

```
x1 <- rnorm(200, 2, sqrt(3))
x2 <- runif(200, 2, 4)
```

```r
x3 <- runif(200, -2, 2)
eps <- rt(200, 5)
y <- 3 + 2 * x1 + x2 + eps
new_data <- data.frame(y,x1,x2,x3)
```

2.2 Do residual bootstrap for linear regression and fit the model y:x1+x2+x3 . Get the percentile CI for the coefficients. Can you exclude x3 ?

```r
model <- lm(y~., new_data)
residuals <- resid(model)
yhat <- fitted(model)
add_data <- data.frame(residuals, yhat)

get_coefs <- function(data){
  coef(lm(y ~ x1 + x2 + x3, data = data))
}
get_sample <- function(d, res) {
  d$y <- (res$yhat + sample(res$residuals, replace=T))
  return(d)
}

model_boot <- boot(new_data, get_coefs, R=1000, sim="parametric", ran.gen=get_sample, mle=add_data)
model_boot
```

```
##
## PARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = new_data, statistic = get_coefs, R = 1000, sim = "parametric",
##     ran.gen = get_sample, mle = add_data)
##
##
## Bootstrap Statistics :
##      original        bias    std. error
## t1* 3.6189299  0.0093524867  0.70431553
## t2* 1.9206219  0.0007636617  0.07596924
## t3* 0.8322374 -0.0035872725  0.22268821
## t4* 0.1878527 -0.0043489246  0.11199788
```

```r
coef_res <- data.frame(
  x1 = quantile(model_boot$t[,2], c(0.005,0.025,0.975, 0.995)),
  x2 = quantile(model_boot$t[,3], c(0.005,0.025,0.975, 0.995)),
  x3 = quantile(model_boot$t[,4], c(0.005,0.025,0.975, 0.995))
)
coef_res
```

```
##             x1        x2          x3
## 0.5%  1.727976 0.2675922 -0.12413268
## 2.5%  1.776424 0.3573477 -0.03569504
## 97.5% 2.068021 1.2388669  0.38083189
## 99.5% 2.106799 1.3763247  0.46604707
```

Based on the results, we can conclude that x3 is insignificant and we can remove it.

2.3 Do pairs bootstrap for linear regression and fit the model y:x1+x2+x3 . Get the percentile CI for the coefficients. Can you exclude x3 ?

```
reg_fun <- function(data, idx){
  coef(lm(y ~ x1 + x2 + x3, data = data[idx,]))
}

model_boot_pairs <- boot(new_data, reg_fun, R=1000)
model_boot_pairs
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = new_data, statistic = reg_fun, R = 1000)
##
##
## Bootstrap Statistics :
##      original        bias    std. error
## t1* 3.6189299  0.0016119962  0.69643861
## t2* 1.9206219  0.0004688326  0.06232198
## t3* 0.8322374 -0.0022580580  0.22386903
## t4* 0.1878527 -0.0026557348  0.11337912
```

```
coef_res_pairs <- data.frame(
  x1 = quantile(model_boot_pairs$t[,2], c(0.005,0.025,0.975, 0.995)),
  x2 = quantile(model_boot_pairs$t[,3], c(0.005,0.025,0.975, 0.995)),
  x3 = quantile(model_boot_pairs$t[,4], c(0.005,0.025,0.975, 0.995))
)
coef_res_pairs
```

```
##             x1        x2          x3
## 0.5%  1.765054 0.2812544 -0.11086595
## 2.5%  1.795320 0.3844804 -0.03048465
## 97.5% 2.037905 1.2692220  0.40701614
## 99.5% 2.069344 1.4311846  0.48188288
```

Here as well, we can conclude that x3 is insignificant based on the intervals, and bootstrap statistics. We can remove x3 from the model.

2.4 Compare the two approaches in 2. and 3. and explain the differences in the sampling approach and how this (might) affect(s) the results.

The main difference of the 2 approaches is the fact that using residual model, we make assumption that the residuals follow the normal distribution. However, in the pairs case, there is no assumption about it. In the case above both methods perform quite similarly, which indicates that the formula specified in the model is not wrong. Since, if it was misspecified, residuals approach would show different results, while pairs strategy would be considered more robust.

Task 3. Summarise the bootstrapping methodology, its advantages and disadvantages based on your exercises for constructing parametric and non-paramteric confidence bounds for estimators, test statistics or model estimates.

Overall, bootstrapping methodology is the same for model estimates as well as for constructing confidence bounds for estimators and hypothesis testing. The gist is to sample the random variables from the original distribution with replacement allowed. Advantage of the non-parametric bootstrapping is that it does not make any assumptions of the distribution. The calculation of the various estimates is very easy. Though, the disadvantage of bootstrapping can be its computation: we need a large number of samples for the estimates to be considered correct (in this exercise we used 10000). Otherwise, the calculated estimates can be misleading, especially if there were outliers in the original distribution.