Exercise 5

107.330 - Statistical Simulation and Computerintensive Methods, WS24

12433688 - Yash Lucas

27.11.2024

Contents

Task 1

Consider a two sample problem and the hypothesis H0: 1= 2 vs H1: 1 2 , where 1 and 2 are the corresponding sample locations. The two samples are:

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, -0.472, 1.638, -0.578, 0.947, -0.329, -0.188, 0.794, 0.894, -1.227, 1.059) 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, 1.401, 0.007, 0.568, -0.005, 0.696)

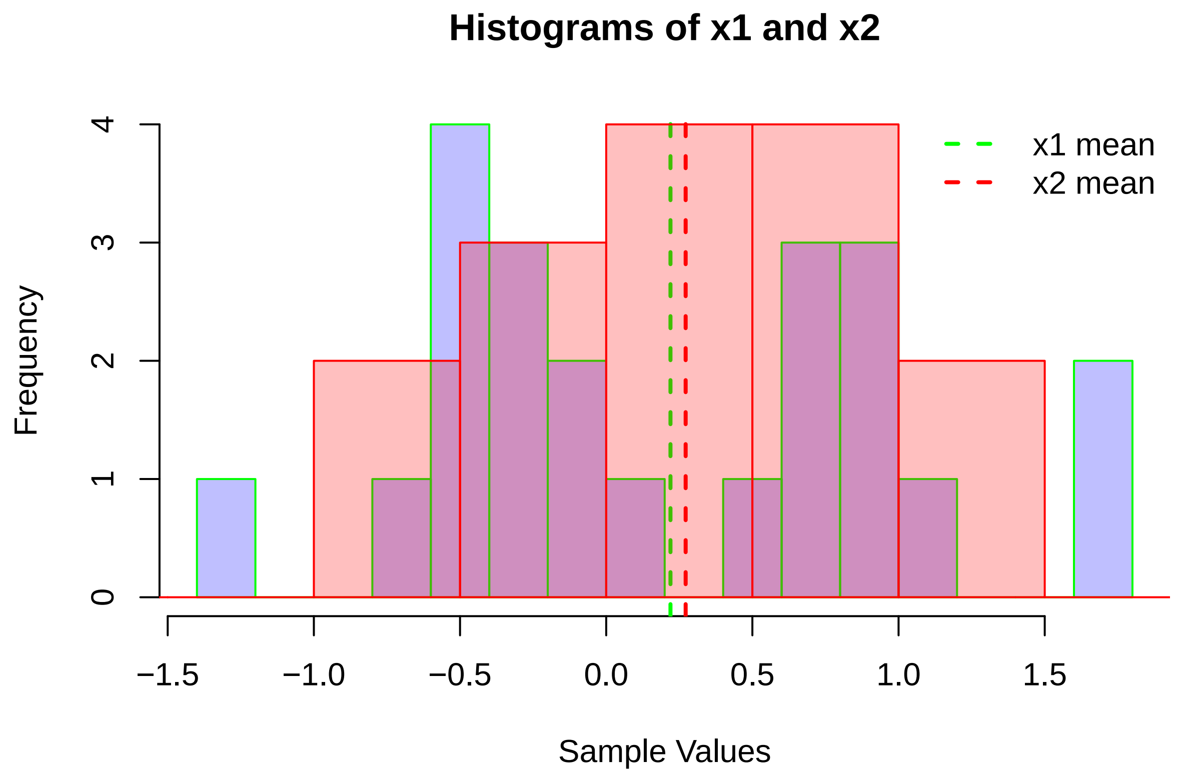
```r
library("boot")
set.seed(12433688)
x1<- c(-0.673, -0.584, 0.572, -0.341, -0.218, 0.603, -0.415, -0.013, 0.763, 0.804, 0.054, 1.746, -0.472
x2<- c(0.913, -0.639, 2.99, -5.004, 3.118, 0.1, 1.128, 0.579, 0.32, -0.488, -0.994, -0.212, 0.413, 1.40
```

Task 1.1

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

```r
c_x1 <- rgb(0, 0, 1, 0.25)
c_x2 <- rgb(1, 0, 0, 0.25)
hist(x1, col = c_x1, breaks = 15, xlab = "Sample Values", main = "Histograms of x1 and x2", border = "g
abline(v = mean(x1), col = "green", lwd = 2, lty = 2)

hist(x2, col = c_x2, breaks = 15, add = TRUE, border = "red")
abline(v = mean(x2), col = "red", lwd = 2, lty = 2)
legend("topright", legend = c("x1 mean", "x2 mean"), col = c("green", "red"), lty = 2, lwd = 2, bty = "
```



Task 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.

```
resample_means <- function(x1, x2) {
  n1 <- length(x1)
  n2 <- length(x2)

  resample_x1 <- sample(x1, n1, replace = TRUE)
  resample_x2 <- sample(x2, n2, replace = TRUE)

  mean_resample_x1 <- mean(resample_x1)
  mean_resample_x2 <- mean(resample_x2)

  x1_centered <- x1 - mean(x1)
  x2_centered <- x2 - mean(x2)

  combined_centered <- c(x1_centered, x2_centered)

  resample_combined_x1 <- sample(combined_centered, n1, replace = TRUE)
  resample_combined_x2 <- sample(combined_centered, n2, replace = TRUE)
  mean_combined_x1 <- mean(resample_combined_x1)
  mean_combined_x2 <- mean(resample_combined_x2)

  return(list(
    mean_resample_x1 = mean_resample_x1,
    mean_resample_x2 = mean_resample_x2,
    mean_combined_x1 = mean_combined_x1,
    mean_combined_x2 = mean_combined_x2
  ))
}
```

```
results <- resample_means(x1, x2)
results
```

```
## $mean_resample_x1
## [1] 0.1545455
##
## $mean_resample_x2
## [1] -0.007166667
##
## $mean_combined_x1
## [1] -0.1935386
##
## $mean_combined_x2
## [1] 0.3340578
```

First Method (Bootstrapping)::

-This is a simple default method for calculating test statistics and creating resamples. - By maintaining each group's unique structure, it draws attention to the distinctions between them. - However, because it presumes that the groups are fundamentally different, it might not represent the null hypothesis (H0) conditions.

Second Method (Centering and Resampling from Combined Data):

-This approach more accurately captures the circumstances under H0, in which it is assumed that the groups have the same underlying distribution. - Since it produces a more balanced test environment, it is especially helpful when sample sizes are very varied. - Individual group differences, which could be crucial in some analyses, are lost, though.

Task 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
test <- function(x1, x2, n) {
  sample_r <- function(x, n) {
    matrix(sample(x, length(x) * n, replace = TRUE), ncol = n)
  }

  sample_c <- function(x1, x2, n) {
    x1_c <- x1 - mean(x1)
    x2_c <- x2 - mean(x2)
    combined <- c(x1_c, x2_c)
    matrix(sample(combined, length(combined) * n, replace = TRUE), ncol = n)
  }

  repl.x1 = sample_r(x1, n)
  repl.x2 = sample_r(x2, n)
  centr = sample_c(x1, x2, n)

  t_val = t.test(x1, x2)$statistic

  t_1 = numeric(n)
  t_2 = numeric(n)
  count1 = count2 = 0

  for (i in 1:n) {
    t_1[i] = t.test(repl.x1[, i], repl.x2[, i])$statistic
    if (abs(t_1[i]) > abs(t_val)) count1 = count1 + 1
    t_2[i] = t.test(centr[1:length(x1), i], centr[(length(x1) + 1):(length(x1) + length(x2)), i])$stati
    if (abs(t_2[i]) > abs(t_val)) count2 = count2 + 1
  }

  p1 = (count1 + 1) / (n + 1)
  p2 = (count2 + 1) / (n + 1)

  ci95_1 = quantile(t_1, c(0.025, 0.975))
  ci99_1 = quantile(t_1, c(0.005, 0.995))

  ci95_2 = quantile(t_2, c(0.025, 0.975))
  ci99_2 = quantile(t_2, c(0.005, 0.995))
  dec_95_1 = ifelse(p1 < 0.05, "Reject H0", "Fail to reject H0")
  dec_95_2 = ifelse(p2 < 0.05, "Reject H0", "Fail to reject H0")

  dec_99_1 = ifelse(p1 < 0.01, "Reject H0", "Fail to reject H0")
  dec_99_2 = ifelse(p2 < 0.01, "Reject H0", "Fail to reject H0")

  result = data.frame(
    method = c("Separate", "Combined"),
    counts = c(count1, count2),
    p_value = c(p1, p2),
    CIUp95 = c(ci95_1[2], ci95_2[2]),
    CILw95 = c(ci95_1[1], ci95_2[1]),
    CIUp99 = c(ci99_1[2], ci99_2[2]),
    CILw99 = c(ci99_1[1], ci99_2[1]),
    decision_95 = c(dec_95_1, dec_95_2),
    decision_99 = c(dec_99_1, dec_99_2)
  )

  return(result)
}
```

```
n=10000
result = test(x1, x2,n)
result
```

```
##     method counts  p_value   CIUp95    CILw95   CIUp99    CILw99
## 1 Separate   9096 0.909609 1.695823 -2.556718 2.148150 -3.230176
## 2 Combined   9103 0.910309 1.969453 -1.976427 2.542347 -2.514589
##         decision_95       decision_99
## 1 Fail to reject H0 Fail to reject H0
## 2 Fail to reject H0 Fail to reject H0
```

Both p-values are significantly higher than the significance thresholds (0.05 and 0.01), indicating that the null hypothesis cannot be ruled out. Additionally, the confidence intervals imply that there is no substantial difference between the means of x1 and x2.

Task 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.

```
perm_test <- function(x1, x2, n) {
  observed_t = t.test(x1, x2)$statistic

  combined = c(x1, x2)
  perm_t = numeric(n)

  for (i in 1:n) {
    shuffled = sample(combined)
    perm_t1 = shuffled[1:length(x1)]
    perm_t2 = shuffled[(length(x1) + 1):length(shuffled)]
    perm_t[i] = t.test(perm_t1, perm_t2)$statistic
  }

  p_value = (sum(abs(perm_t) >= abs(observed_t)) + 1) / (n + 1)

  ci95 = quantile(perm_t, c(0.025, 0.975))
  ci99 = quantile(perm_t, c(0.005, 0.995))

  decision_95 = ifelse(p_value < 0.05, "Reject H0", "Fail to reject H0")
  decision_99 = ifelse(p_value < 0.01, "Reject H0", "Fail to reject H0")

  result = data.frame(
    p_value = p_value,
    CIUp95 = ci95[2],
    CILw95 = ci95[1],
    CIUp99 = ci99[2],
    CILw99 = ci99[1],
    decision_95 = decision_95,
    decision_99 = decision_99
  )

  return(result)
}

result = perm_test(x1, x2,n)
result
```

```
##         p_value   CIUp95    CILw95   CIUp99    CILw99       decision_95
## 97.5% 0.9129087 1.896543 -1.973934 2.442875 -2.526817 Fail to reject H0
##             decision_99
## 97.5% Fail to reject H0
```

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

Task 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_bootstrap <- function(x1, x2, n) {
  w_obs <- wilcox.test(x1, x2)$statistic
  c1 <- c2 <- c3 <- 0
  w1 <- w2 <- w3 <- numeric(n)

  combined <- c(x1, x2)
  for (i in 1:n) {
    r1 <- sample(x1, replace = TRUE)
    r2 <- sample(x2, replace = TRUE)
    w1[i] <- wilcox.test(r1, r2, exact = FALSE)$statistic
    if (abs(w1[i]) > abs(w_obs)) c1 <- c1 + 1
    samp <- sample(combined, length(combined), replace = TRUE)
    w2[i] <- wilcox.test(samp[1:length(x1)], samp[-(1:length(x1))], exact = FALSE)$statistic
    if (abs(w2[i]) > abs(w_obs)) c2 <- c2 + 1
    perm <- sample(combined, length(combined), replace = FALSE)
    w3[i] <- wilcox.test(perm[1:length(x1)], perm[-(1:length(x1))], exact = FALSE)$statistic
    if (abs(w3[i]) > abs(w_obs)) c3 <- c3 + 1
  }
  result <- data.frame(
    method = c("Separate", "Combined", "Permuted"),
    counts = c(c1, c2, c3),
    p_value = c((c1 + 1) / (n + 1), (c2 + 1) / (n + 1), (c3 + 1) / (n + 1)),
    CI_Low_95 = c(quantile(w1, 0.025), quantile(w2, 0.025), quantile(w3, 0.025)),
    CI_Up_95 = c(quantile(w1, 0.975), quantile(w2, 0.975), quantile(w3, 0.975)),
    CI_Low_99 = c(quantile(w1, 0.005), quantile(w2, 0.005), quantile(w3, 0.005)),
    CI_Up_99 = c(quantile(w1, 0.995), quantile(w2, 0.995), quantile(w3, 0.995))
  )

  return(result)
}
```

```
result <- w_bootstrap(x1,x2,n)
result
```

```
##     method counts   p_value CI_Low_95 CI_Up_95 CI_Low_99 CI_Up_99
## 1 Separate   4782 0.4782522       109      255    88.000  275.005
## 2 Combined   6690 0.6690331       126      269   104.500  290.500
## 3 Permuted   6613 0.6613339       127      270   104.995  291.000
```

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)
x3 <- runif(200, -2, 2)
eps <- rt(200, 5)
y <- 3 + 2 * x1 + x2 + eps
d1 <- data.frame(y,x1,x2,x3)
```

Task 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
run_bootstrap <- function(data, n) {
  model <- lm(y ~ ., data)
  resids <- resid(model)
  yhat <- fitted(model)
  add_data <- data.frame(resids, yhat)

  get_coefs <- function(d) {
    coef(lm(y ~ x1 + x2 + x3, data = d))
  }

  generate_sample <- function(d, res) {
    d$y <- res$yhat + sample(res$resids, replace = TRUE)
    return(d)
  }

  boot_result <- boot(data, get_coefs, R = n, sim = "parametric", ran.gen = generate_sample, mle = add_
  return(boot_result)
}
```

```r
result <- run_bootstrap(d1, 1000)
result
```

```
##
## PARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = data, statistic = get_coefs, R = n, sim = "parametric",
##     ran.gen = generate_sample, mle = add_data)
##
##
## Bootstrap Statistics :
##       original        bias    std. error
## t1*  2.9160629  0.0124410122  0.48175767
## t2*  1.9940063 -0.0004855966  0.05519226
## t3*  1.0335027 -0.0039598119  0.15013597
## t4* -0.0288487 -0.0018776831  0.07625802
```

```r
get_coef_quantiles <- function(boot_result) {
  coef_quantiles <- data.frame(
    x1 = quantile(boot_result$t[, 2], probs = c(0.005, 0.025, 0.975, 0.995)),
    x2 = quantile(boot_result$t[, 3], probs = c(0.005, 0.025, 0.975, 0.995)),
    x3 = quantile(boot_result$t[, 4], probs = c(0.005, 0.025, 0.975, 0.995))
  )
}

get_coef_quantiles(result)
```

From the above table we can conclude qverall, x1 and x2 appear to be robust predictors of the dependent variable, with stable and significant confidence intervals. x3, however, seems uncertain in its effect and might not contribute significantly to the model, especially considering its confidence interval includes zero.

Task 2.2

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 <- boot(d1, reg_fun, R = 1000)
model_boot
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = d1, statistic = reg_fun, R = 1000)
##
##
## Bootstrap Statistics :
##       original        bias    std. error
## t1*  2.9160629 -0.0153436486  0.40890444
## t2*  1.9940063  0.0008995381  0.04727468
## t3*  1.0335027  0.0043457843  0.14050603
## t4* -0.0288487 -0.0037001624  0.07523192
```

```
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%  1.880703 0.6946338 -0.2221864
## 2.5%  1.906792 0.7719823 -0.1831794
## 97.5% 2.086308 1.3132451  0.1180443
## 99.5% 2.117395 1.3694726  0.1574718
```

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

Task 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.

Residual bootstrap generates fresh samples while maintaining the structure of the original model's error terms by resampling based on the residuals from the fitted model. This method makes the assumption that the residuals accurately reflect the variability in the data and that the model fits the data adequately.

The associations between the predictors and the response variable can be preserved using this method by resampling the residuals and adding them to the fitted values. The model's stability during resampling is revealed by the p-values and confidence intervals, which show the variability of the model's coefficients.

However, ordinary bootstrap does not assume a particular model structure; instead, it resamples the data points with replacement. Instead of depending on the fitted model, it uses the observed data to create fresh samples. More versatility is provided by this method, which also leaves the relationship between the variables unrestricted. More variation in the observed data can be captured by the conventional bootstrap, which may lead to more variable coefficient estimates, possibly broader confidence ranges, and larger variability in p-values.

The reliance on the fitted model and the method of resampling the data are the main distinctions between these two strategies. Standard bootstrap resamples the data points themselves, allowing for more flexibility and maybe more variety in the findings, whereas residual bootstrap maintains the error structure and is best when the model's fit is presumed to be accurate. Both approaches have advantages and disadvantages, and the decision between them is based on the model fit and data assumptions.

Task 3

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

Methodology of Bootstrapping - Bootstrapping is a resampling approach that involves repeatedly sampling with replacement from the observed data in order to estimate the distribution of a statistic. It can be used in both parametric (assuming a model) and non-parametric (without assuming a model) situations. With this approach, p-values and confidence intervals can be estimated without depending on conventional parametric assumptions.

Advantages of Bootstrapping - Because of its adaptability, bootstrapping can be used with non-normal data and sophisticated models. It is resistant to non-standard data since it doesn't require any particular distributional assumptions. The approach can be used to a variety of analytical domains, including machine learning and fundamental statistics, and it is also reasonably simple to apply.

Disadvantages of Bootstrapping - Despite its benefits, bootstrapping requires a lot of computing power, particularly when dealing with complicated models or big datasets. Its accuracy depends on how representative the sample is, and it may also be susceptible to outliers. The efficacy of parametric bootstrapping depends on the underlying model being properly specified.