Inferencia Causal - lista 2

2024-03-21

QUESTÃO 3

```r
library(Matching)
```

```
## Loading required package: MASS

## ##
## ##  Matching (Version 4.10-14, Build Date: 2023-09-13)
## ##  See https://www.jsekhon.com for additional documentation.
## ##  Please cite software as:
## ##   Jasjeet S. Sekhon. 2011. ``Multivariate and Propensity Score Matching
## ##   Software with Automated Balance Optimization: The Matching package for R.''
## ##   Journal of Statistical Software, 42(7): 1-52.
## ##
```

```r
data(lalonde)

# t studentizada ----------------------------------------------------------
z <- lalonde$treat
y <- lalonde$re78

n <- nrow(lalonde)
n1 <- sum(z) # quantos receberam tratamento
n0 <- n - n1 # quantos não receberam tratamento
tau <- mean(y[z == 1]) - mean(y[z == 0]) # efeito causal médio
s2 = var(y)

media_y_1 <- mean(y[z == 1])
s2_1 <- sum((y[z == 1] - media_y_1)^2)/(n1 - 1)

media_y_0 <- mean(y[z == 0])
s2_0 <- sum((y[z == 0] - media_y_0)^2)/(n0 - 1)

est_teste_frt <- tau / sqrt(s2_1/n1 + s2_0/n0)

# FRT Monte Carlo
mc <- 10^5
est_teste_frt_mc <- rep(0, mc)
for (i in 1:mc) {
  zpermut <- sample(z) # permutando z
  media_y_1 <- mean(y[zpermut == 1])
  s2_1 <- sum((y[zpermut == 1] - media_y_1)^2)/(n1 - 1)

  media_y_0 <- mean(y[zpermut == 0])
  s2_0 <- sum((y[zpermut == 0] - media_y_0)^2)/(n0 - 1)
```

```r
  tau_permut <- mean(y[zpermut == 1]) - mean(y[zpermut == 0]) # efeito causal medio da permutação
  est_teste_frt_mc[i] <- tau_permut/sqrt(s2_1/n1 + s2_0/n0) # estatistica do teste da permutação
}

pvalor_frt_mc <- mean(abs(est_teste_frt_mc) >= abs(est_teste_frt)); pvalor_frt_mc
```

```
## [1] 0.00714
```

```r
# H0: não existe efeito causal

est_test_frt_classico_t <- t.test(y[z == 1], y[z == 0], var.equal = FALSE)$statistic
pvalor_frt_classico_t <- t.test(y[z == 1], y[z == 0], var.equal = FALSE)$p.value
```

```r
c(est_teste_frt, pvalor_frt_mc)
```

```
## [1] 2.674146 0.007140
```

```r
c(est_test_frt_classico_t, pvalor_frt_classico_t)
```

```
##           t
## 2.674145798 0.007892971
```

```r
# wilcoxon ------------------------------------------------------------
z <- lalonde$treat
y <- lalonde$re78

W_obs <- sum(rank(y)[z == 1])

# FRT Monte Carlo
mc <- 10^5
W_mc <- rep(0, mc)
for (i in 1:mc) {
  zpermut <- sample(z) # permutando z
  W_mc[i] <- sum(rank(y)[zpermut == 1]) # estatistica do teste da permutação
}

pvalor_W <- mean(W_mc >= W_obs)

stat <- wilcox.test(y[z == 1], y[z==0], exact = FALSE)$statistic
p <- wilcox.test(y[z == 1], y[z==0], exact = FALSE)$p.value
```

```r
c(W_obs, pvalor_W)
```

```
## [1] 44607.50000     0.00522
```

```r
c(stat, p)
```

```
##            W
## 2.740250e+04 1.094664e-02
```

QUESTÃO 04

```r
set.seed(123)
n = 20
y0 = rnorm(n)
tau = rnorm(n, -0.5)
y1 = y0 + tau
```

```r
z = rbinom(n, 1, 0.5)
y = z*y1 + (1-z)*y0
n1 = sum(z)
n0 = n - n1

## efeito causal medio
# mean(y[z == 1]) - mean(y[z == 0])

s2 =  sum((y-mean(y))^2)/(n-1)
estat = (mean(y[z==1]) - mean(y[z==0]))/(n*s2/(n1*n0))

permutation = function(n, n1){
  M = choose(n, n1)
  treat.index = combn(n, n1)
  Z_matriz = matrix(0, n, M)
  for(m in 1:M){
    treat = treat.index[, m]
    Z_matriz[treat, m] = 1
  }
  return(Z_matriz)
}

Z_matriz = permutation(n, n1)

## p_FRT
estat_comb = c()
for(i in 1:ncol(Z_matriz)){
  zpermut = Z_matriz[, i]
  estat_comb[i] = (mean(y[zpermut==1]) - mean(y[zpermut==0]))/(n*s2/(n1*n0))
}
mean(abs(estat_comb) >= abs(estat))
```

```
## [1] 0.06135389
```

```r
## p_FRT chapeu (mc)
R = 10^5
estat_mc = c()
for(i in 1:R){
  zpermut = sample(z)
  estat_mc[i] = (mean(y[zpermut==1]) - mean(y[zpermut==0]))/(n*s2/(n1*n0))
}

mean(abs(estat_mc) >= abs(estat))
```

```
## [1] 0.06223
```

```r
## p_FRT tio (finite-sample valid Monte Carlo approximation)
(1+sum(abs(estat_mc) >= abs(estat)))/(R+1)
```

```
## [1] 0.06223938
```

QUESTÃO 06

```r
set.seed(123)
```

```r
## Simulando o experimento
n = 100
n1 = 60
n0 = 40
y0 = sort(rexp(n), decreasing = TRUE)
tau = 1
y1 = y0 + tau
z = sample(c(rep(1, n1), rep(0, n0)))

## Efeito causal medio estimado
tau_hat = mean(y1[z == 1]) - mean(y0[z == 0])
tau_hat
```

```
## [1] 1.092852
```

```r
## Variancia estimada (conservadora) (chapeu)
V_hat = var(y1[z == 1])/n1 + var(y0[z == 0])/n0
V_hat
```

```
## [1] 0.03718348
```

```r
## Variancia estimada (conservadora) (til)
V_til = (1/n)*(sqrt(n0/n1)*sd(y1[z == 1])+sqrt(n1/n0)*sd(y1[z == 0]))^2
V_til
```

```
## [1] 0.03466293
```

```r
## IC - Variancia chapeu
intervalo_confianca = c(tau_hat - 1.96*sqrt(V_hat), tau_hat + 1.96*sqrt(V_hat))
intervalo_confianca
```

```
## [1] 0.7149044 1.4707987
```

```r
## IC - Variancia til
intervalo_confianca = c(tau_hat - 1.96*sqrt(V_til), tau_hat + 1.96*sqrt(V_til))
intervalo_confianca
```

```
## [1] 0.7279391 1.4577640
```

```r
## Variancia estimada
var_hat_tau = var(y1)/n1 + var(y0)/n0 - var(y1 - y0)/n
var_hat_tau
```

```
## [1] 0.044947
```

```r
## Permutando o tratamento - variancia chapeu
tau_hat_p = c()
V_hat_p = c()
lim_sup = c()
lim_inf = c()
est_hat = c()

for (i in 1:10^4) {
  z_permut = sample(z)
  tau_hat_p[i] = mean(y1[z_permut == 1]) - mean(y0[z_permut == 0])
  V_hat_p[i] = var(y1[z_permut == 1])/n1 + var(y0[z_permut == 0])/n0
  lim_sup[i] = tau_hat_p[i] + 1.96*sqrt(V_hat_p[i])
  lim_inf[i] = tau_hat_p[i] - 1.96*sqrt(V_hat_p[i])
```

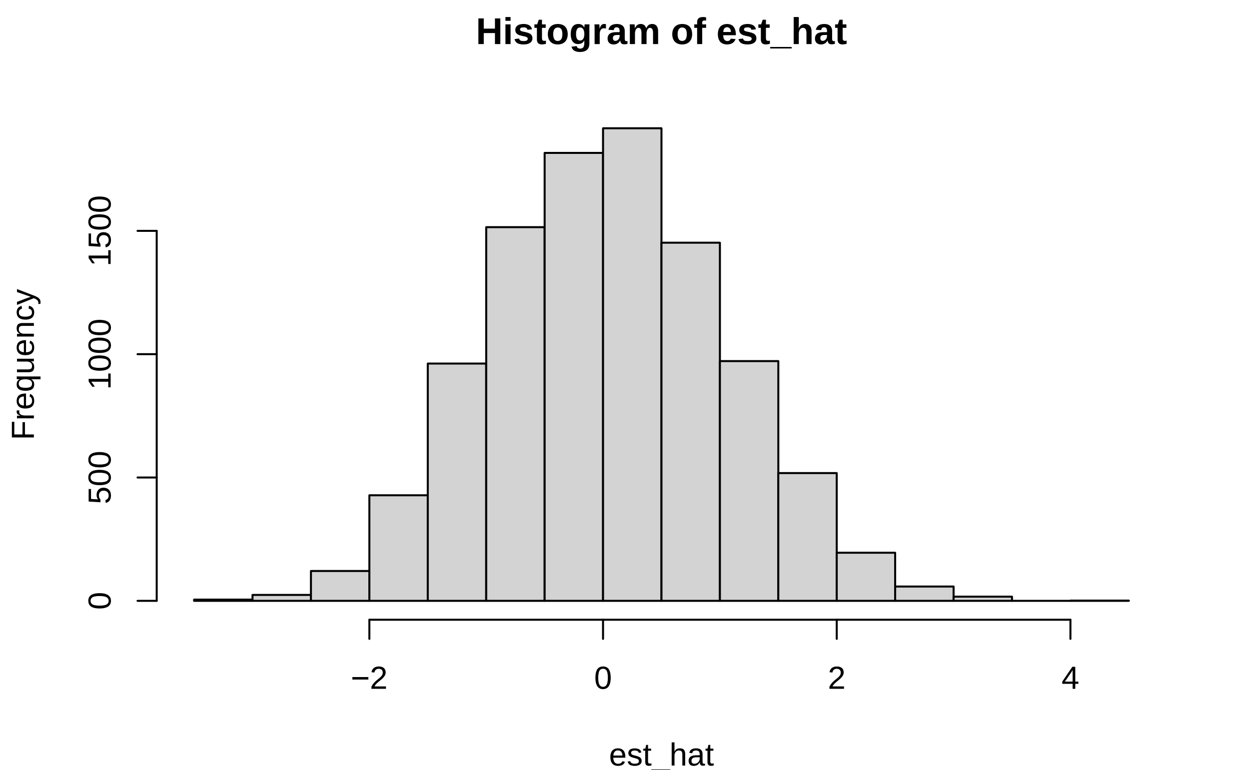
```
  est_hat[i] = (tau_hat_p[i] - tau) / sqrt(V_hat_p[i])
}
mean(V_hat_p)
```

```
## [1] 0.04501226
```

```
cobertura_hat = c()
for (i in 1:10^4) {
  cobertura_hat[i] = ifelse(lim_inf[i] < tau && tau < lim_sup[i], 1, 0)
}
mean(cobertura_hat)
```

```
## [1] 0.9547
```

```
hist(est_hat)
```




```
## Permutando o tratamento - variancia til
tau_hat_p = c()
V_til_p = c()
lim_sup = c()
lim_inf = c()
est_til = c()

for(i in 1:10^4) {
  z_permut = sample(z)
  tau_hat_p[i] = mean(y1[z_permut == 1]) - mean(y0[z_permut == 0])
  V_til_p[i] = (1/n)*(sqrt(n0/n1)*sd(y1[z_permut == 1])+sqrt(n1/n0)*sd(y1[z_permut == 0]))^2
  lim_sup[i] = tau_hat_p[i] + 1.96*sqrt(V_hat_p[i])
  lim_inf[i] = tau_hat_p[i] - 1.96*sqrt(V_hat_p[i])
  est_til[i] = (tau_hat_p[i] - tau) / sqrt(V_hat_p[i])
}
```

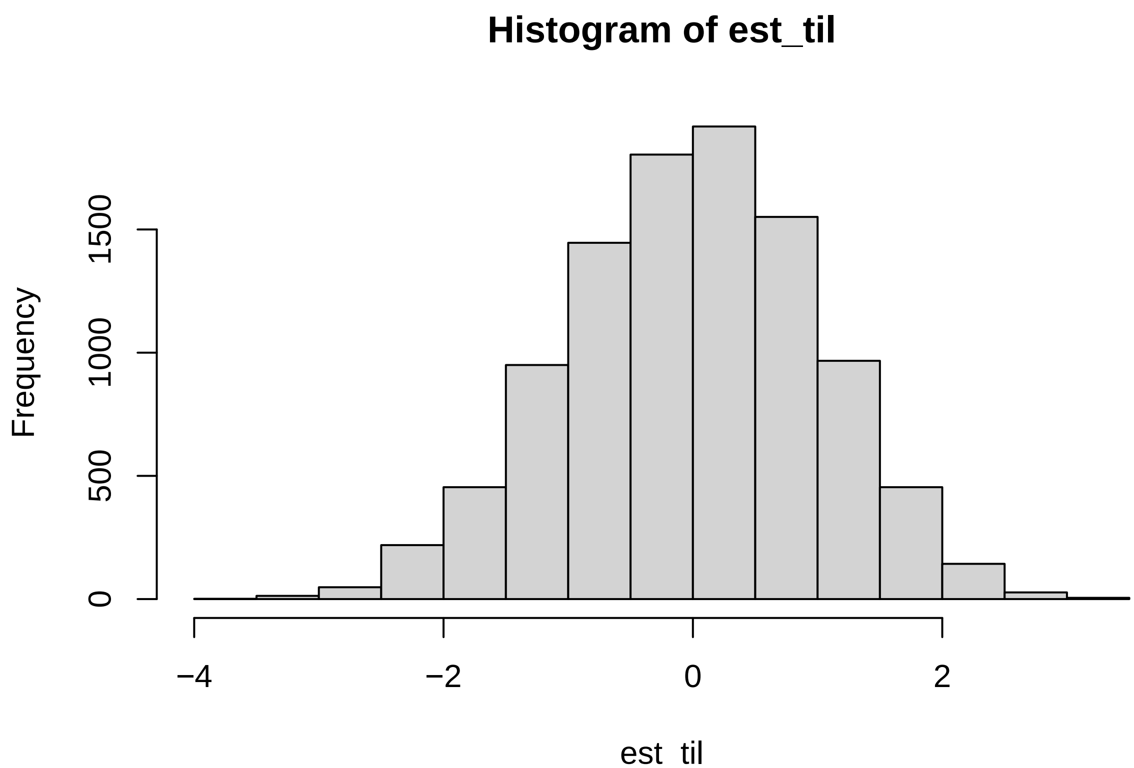
```r
mean(V_til_p)
```

```
## [1] 0.04332417
```

```r
cobertura_til = c()
for (i in 1:10^4) {
  cobertura_til[i] = ifelse(lim_inf[i] < tau && tau < lim_sup[i], 1, 0)
}
mean(cobertura_til)
```

```
## [1] 0.9497
```

```r
hist(est_til)
```




```r
mean(V_til_p)
```

```
## [1] 0.04332417
```

```r
mean(V_hat_p)
```

```
## [1] 0.04501226
```

```r
mean(cobertura_hat)
```

```
## [1] 0.9547
```

```r
mean(cobertura_til)
```

```
## [1] 0.9497
```