Quinto Laboratorio - Introducción al R

Bioestadistica 2022

Intervalos de confianza

El más usual es, dada una muestra aleatoria simple $X_1, \ldots, X_n$, construir un intervalo de confianza para la media teórica μ a nivel $1 - \alpha$. Creémos una función que sirva para el caso datos normales, varianza conocida (este caso también servira cuando tengamos muestras grandes para una distribución cualquiera). Veamos cómo se podria hacer.

```
norm.interval = function(datos, varianza=var(datos),
                         nivel.conf = 0.95){
  z = qnorm((1 - nivel.conf)/2, lower.tail = FALSE)
  m = mean(datos)
  dt = sqrt(varianza/length(datos))
  c(m - z * dt, m + z * dt)
}
```

Ahora generamos una muestra gaussiana de tamaño n=50, media 0 y varianza 1. Luego calculamos un intervalo de confianza para μ a nivel 0.95.

```
X = rnorm(50,0,1)
norm.interval(X,1)
```

```
## [1] -0.1503198  0.4040417
```

Ahora nos gustaría estudiar qué sucede cuando realizamos esto muchas veces. En teoría, como es un intervalo de confianza a nivel 0.95, debería suceder que 1 de cada 20 veces la media teórica no se encuentre en el intervalo de confianza calculado para la muestra simulada (siempre y cuando dicha muestra cumpla la hipótesis nula).

Simulamos 100 muestras de tamaño n, y veamos cómo varían los intervalos de confianza.

```
nMC = 100 ; n = 30
mu = 0 ; sigma = 1
muestras = matrix(rnorm(nMC * n,mu,sigma),n)
int.conf = apply(muestras,2,function(x) norm.interval(x,1))
```

Veamos qué intervalos contienen a mu y cuáles no.

```
adentro=-(int.conf[1,] <= mu & int.conf[2,] >= mu) +2
table(adentro)
```

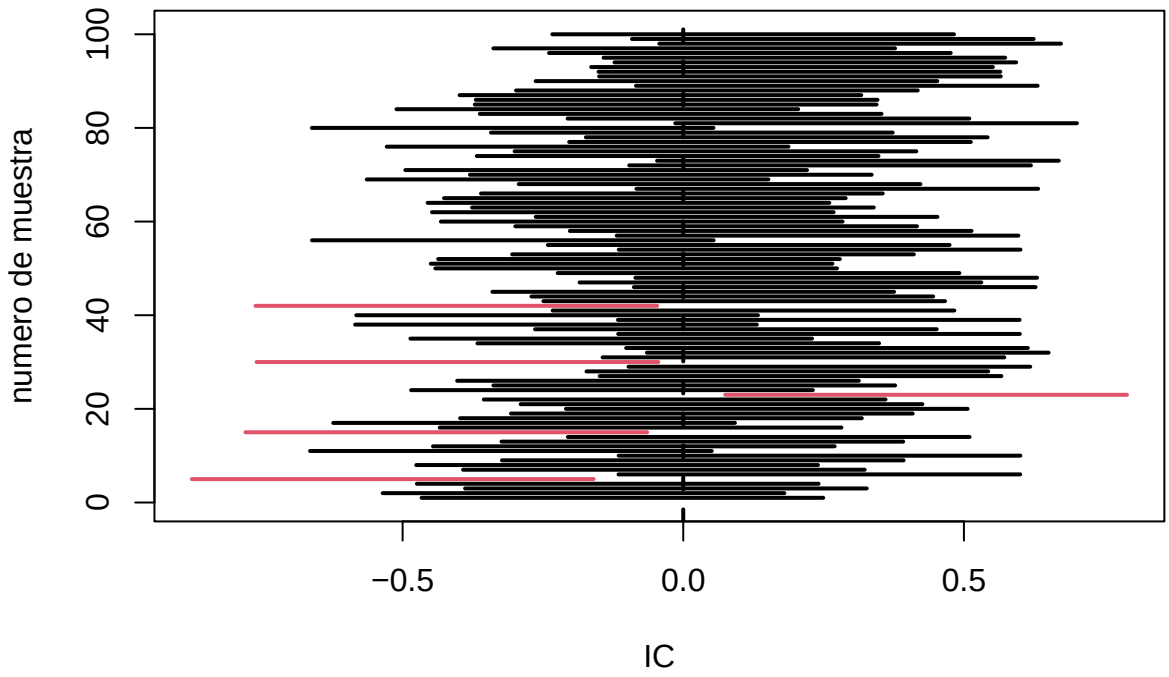
```
## adentro
##  1  2
## 95  5
```

Grafiquemos ahora los intervalos de confianza para cada muestra, para poder visualizar los resultados de una manera conveniente.

```
plot(range(int.conf), c(0, 1+nMC),
       type = "n", xlab = "IC",
       ylab = "numero de muestra")
```

```
for (i in 1:nMC) {
  lines(int.conf[, i], rep(i,2),
        lwd=2, col=adentro[i])
}

abline(v = 0, lwd = 2, lty = 2)
```



Tests de hipótesis

Test de Kolmogorov-Smirnov

Los test de hipótesis son un procedimiento para juzgar si una propiedad que se supone en una población estadística es compatible con lo observado en una muestra de dicha población. Son herramientas fundamentales para el análisis estadístico de datos, que se usan todo el tiempo a nivel académico e industrial. Nuestro objetivo es aprender a implementar los test de hipótesis estudiados en el teórico. Comencemos con el tests de Kolmogorov-Smirnov, tanto para una muestra como para dos muestras.

```
?ks.test
```

Veamos un ejemplo para un test de una muestra.

```
sim = rnorm(10000, 2, 2)
tks = ks.test(sim, "pnorm", 2, 2)
tks$statistic
```

```
##           D
## 0.005938409
```

```
tks$p.value
```

```
## [1] 0.8723315
```

Observemos que podemos indicarle a la función *ks.test* qué familia de distribuciones queremos testear (a través de "pnorm", por ejemplo), y también los parámetros correspondientes.

También podemos realizar tests de dos muestras.

```
n_sims = 10
muestra1 = rnorm(n_sims, 0, 3)
muestra2 = runif(n_sims, -6, 6)
tks = ks.test(muestra1, muestra2)
tks$p.value
```

```
## [1] 0.1678213
```

```
n_sims = 1000
muestra1 = rnorm(n_sims, 0, 3)
muestra2 = runif(n_sims, -6, 6)
tks = ks.test(muestra1, muestra2)
tks$p.value
```

```
## [1] 3.609819e-06
```

Tests de Lilliefors

A través del paquete *KScorrect*, podemos calcular tests de Lilliefors. La manera de calcular la distribución del estadístico (para todos los casos) es a través de simulaciones. Es importante ver la ayuda de la función para saber sobre qué familias de distribuciones podemos implementar este test.

```
# install.packages("KScorrect")
library(KScorrect)
?LcKS
```

Veamos como se implementan los test.

```
sim = rnorm(10000, 3, 2)
tl = LcKS(sim, "pnorm", nreps = 10000)
tl$p.value
```

```
## [1] 0.7887211
```

Test chi cuadrado

Otro test estudiado en el teórico es el test chi cuadrado. Si bien hemos visto que el test chi cuadrado tiene muchas aplicaciones e interpretaciones, el caso básico es el discreto. La función que nos permitirá realizar tests chi cuadrado es *chisq.test*.

```
?chisq.test
```

De vuelta, hay muchos parámetros disponibles para ajustar el test. En particular, el parámetro *correct* aplica una corrección de continuidad para el caso de tablas de 2×2 (se recomienda mantener en *FALSE*), y tiene la opción de simular la distribución de la hipótesis nula para el caso en que no esperemos que las hipótesis del test se cumplan (por ejemplo, pocas observaciones).

Comencemos realizando un test de ajuste. Simulemos binomiales $Bin(10, 0.3)$, y verifiquemos que las frecuencias observadas se ajustan a las esperadas.

```
sim = rbinom(10000, 10, 0.5)
p = dbinom(0:10, 10, 0.5)
ct = chisq.test(table(sim), p = p, correct = FALSE)
ct$statistic
```

```
## X-squared
##  5.617087
```

```
ct$p.value
```

```
## [1] 0.8463431
```

Es posible que, si la frecuencia esperada para una categoría es baja y disponemos de pocas observaciones, no haya ninguna observación para dicha categoría. En esos casos, hay que tener cuidado al comparar el resultado de la función *table* con el vector p (pueden tener distinto largo).

Veamos ahora un test de dependencia para dos muestras. Volveremos a usar la base de datos *mtcars*, y estudiaremos la dependencia entre la cantidad de cilindros y la cantidad de carburadores.

```
head(mtcars)
```

```
##                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```
table(mtcars$cyl, mtcars$carb)
```

```
##
##    1 2 3 4 6 8
##   4 5 6 0 0 0 0
##   6 2 0 0 4 1 0
##   8 0 4 3 6 0 1
```

Compararemoms los resultados del test para la distribución del estadístico asintótica y para la distribución del estadístico simulada.

```
ct_asin = chisq.test(mtcars$cyl, mtcars$carb, correct = FALSE)
```

```
## Warning in chisq.test(mtcars$cyl, mtcars$carb, correct = FALSE): Chi-squared
## approximation may be incorrect
```

```
ct_asin$p.value
```

```
## [1] 0.006632478
```

```
ct_sim = chisq.test(mtcars$cyl, mtcars$carb, simulate.p.value = TRUE, B = 100000)
ct_sim$p.value
```

```
## [1] 0.001469985
```

Podemos observar varias cosas:

- Si bien ambos test rechazan H_0 (las variables son dependientes), hay diferencias no despreciables entre los p-valores obtenidos.
- Al simular la distribución de la hipótesis nula, obtenemos distintos valores al correr el código varias veces, e incluso puede que sean valores no tan cercanos para distintas corridas. Esto quiere decir que, de ser posible, debemos tomar valores grandes de B, y así reducir la varianza del p-valor obtenido.
- El resultado más confiable debería ser el obtenido a partir de simulaciones, siempre y cuando podamos asegurar que usamos una cantidad suficientemente grande de simulaciones (B grande para este caso). Esto se debe a que podemos controlar la cantidad de simulaciones realizadas (los resultado mejoran al aumentar las simulaciones) y no la cantidad de observaciones en la muestra (los resultados asintóticos mejoran al tener más observaciones). Si la distribución del estadístico no dependiera de un resultado asintótico y fuese un resultado exacto, entonces es mejor usar ese resultado y no las simulaciones.