



UAX

UNIVERSIDAD ALFONSO X EL SABIO

R Statistics

GRADO EN BIOMEDICINA

Bioinformática

idelhgar@uax.es

2023

Estadística básica

- **R tiene una serie de funciones para hacer una serie de estadísticas**
 - **Sum()**
 - **Mean()**
 - **Median()**
 - **Max()**
 - **Min()**
 - **Sd()**
 - **Var()**
 - **Length()**
 - **Quantile()**
- **A parte, tiene un montón de paquetes que realizan análisis estadístico. Pero nosotros nos vamos a centrar en los que vienen por defecto en R::base.**

Shapiro-Wilk test

- **Shapiro.test()**
- Se emplea para determinar si la distribución de una muestra es normal o no.
- Solamente se puede emplear para una muestra de entre 3 y 5000 observaciones
- ***Para muestras mayores se emplea un quantile-quantile plot

```
shapiro.test(numericVector) # Does myVec follow a normal disbn?
```

```
# Example: Test a normal distribution
```

```
set.seed(100)
```

```
normaly_disb <- rnorm(100, mean=5, sd=1) # generate a normal distribution
```

```
shapiro.test(normaly_disb) # the shapiro test.
```

```
#=> Shapiro-Wilk normality test
```

```
#=>
```

```
#=> data: normaly_disb
```

```
#=> W = 0.98836, p-value = 0.535
```

```
# Example: Test a uniform distribution
```

```
set.seed(100)
```

```
not_normaly_disb <- runif(100) # uniform distribution.
```

```
shapiro.test(not_normaly_disb)
```

```
#=> Shapiro-Wilk normality test
```

```
#=> data: not_normaly_disb
```

```
#=> W = 0.96509, p-value = 0.009436
```

T-test

- **Método de t-Student (t.test)**
- Método clásico para comparar medias de dos muestras que tienen una distribución normal (campana de Gauss).
- Se le describe como test paramétrico.
- También se puede emplear para una sola muestra

U-test (Wilcoxon)

- **U.test o test de Wilcoxon (`wilcox.test()`)**
- Método para comparar dos medianas de dos muestras.
- Se emplean cuando los datos no están distribuidos con una distribución normal.
- Se considera como no paramétrico

```
set.seed(100)
x <- rnorm(50, mean = 10, sd = 0.5)
t.test(x, mu=10) # testing if mean of x could be
#=> One Sample t-test
#=>
#=> data:  x
#=> t = 0.70372, df = 49, p-value = 0.4849
#=> alternative hypothesis: true mean is not equal to 10
#=> 95 percent confidence interval:
#=>  9.924374 10.157135
#=> sample estimates:
#=> mean of x
#=> 10.04075
```

```
numeric_vector <- c(20, 29, 24, 19, 20, 22, 28, 23, 19, 19)
wilcox.test(numeric_vector, mu=20, conf.int = TRUE)
#> Wilcoxon signed rank test with continuity correction
#>
#> data:  numeric_vector
#> V = 30, p-value = 0.1056
#> alternative hypothesis: true location is not equal to 20
#> 90 percent confidence interval:
#> 19.00006 25.99999
#> sample estimates:
#> (pseudo)median
#> 23.00002
```

```
x <- c(0.80, 0.83, 1.89, 1.04, 1.45, 1.38, 1.91, 1.64, 0.73, 1.46)
y <- c(1.15, 0.88, 0.90, 0.74, 1.21)
wilcox.test(x, y, alternative = "g") # g for greater
#=> Wilcoxon rank sum test
#=>
#=> data: x and y
#=> W = 35, p-value = 0.1272
#=> alternative hypothesis: true location shift is greater than 0
```

```
t.test(1:10, y = c(7:20)) # P = .00001855
#=> Welch Two Sample t-test
#=>
#=> data: 1:10 and c(7:20)
#=> t = -5.4349, df = 21.982, p-value = 1.855e-05
#=> alternative hypothesis: true difference in means is not equal to 0
#=> 95 percent confidence interval:
#=> -11.052802 -4.947198
#=> sample estimates:
#=> mean of x mean of y
#=> 5.5 13.5
```

Use paired = TRUE for 1-to-1 comparison of observations.

t.test(x, y, **paired** = **TRUE**) *# when observations are paired, use 'paired' argument.*

wilcox.test(x, y, **paired** = **TRUE**) *# both x and y are assumed to have similar shapes*

Kruskal-Wallis test

- **Kruskal.test()**
- Es una extensión del de Wilcoxon pero para más de dos muestras.
- Si se emplea para dos muestras sería exactamente igual que el de Wilcoxon.

Kolmogorov and Smirnov Test

- **Ks.test()**
- Se emplea para testar si dos muestras tienen la misma distribución
- La hipótesis nula es que tienen la misma distribución. Si el p-value es menor de 0.05 son de diferentes distribuciones.

```
ks.test(x, y) # x and y are two numeric vector
```

```
# From different distributions
```

```
x <- rnorm(50)
```

```
y <- runif(50)
```

```
ks.test(x, y) # perform ks test
```

```
#=> Two-sample Kolmogorov-Smirnov test
```

```
#=>
```

```
#=> data: x and y
```

```
#=> D = 0.58, p-value = 4.048e-08
```

```
#=> alternative hypothesis: two-sided
```

```
# Both from normal distribution
```

```
x <- rnorm(50)
```

```
y <- rnorm(50)
```

```
ks.test(x, y) # perform ks test
```

```
#=> Two-sample Kolmogorov-Smirnov test
```

```
#=>
```

```
#=> data: x and y
```

```
#=> D = 0.18, p-value = .3959
```

Fisher's F-test

- **var.test()** **Fisher.test()**
- Se emplea para ver si dos muestras tienen la misma varianza
- También se emplea para analizar tablas de contingencia cuando los tamaños de las muestras son pequeños.
- También `fligner.test()` y `bartlett.test()` son parecidos

```
var.test(x, y) # Do x and y have the same variance?
```

Chi-squared test

- **Chisq.test()**
- Test para asociación.
- Identifica diferencias significativas entre grupos categóricos. Los datos tienen que estar en una tabla de contingencia.

```
chisq.test(table(categorical_X, categorical_Y), correct = FALSE) # Yates continuity correction not applied
#or
summary(table(categorical_X, categorical_Y)) # performs a chi-squared test.
# Sample results
#=> Pearson's Chi-squared test
#=> data: M
#=> X-squared = 30.0701, df = 2, p-value = 2.954e-07
```

How to tell if x, y are independent?

There are two ways to tell if they are independent:

By looking at the p-Value: If the p-Value is less than 0.05, we fail to reject the null hypothesis that the x and y are independent. So for the example output above, (p-Value=2.954e-07), we reject the null hypothesis and conclude that x and y are not independent.

From Chi.sq value: For 2 x 2 contingency tables with 2 degrees of freedom (d.o.f), if the Chi-Squared calculated is greater than 3.841 (critical value), we reject the null hypothesis that the variables are independent. To find the critical value of larger d.o.f contingency tables, use `qchisq(0.95, n-1)`, where n is the number of variables.

Correlation test

- **Cor.test()**
- Test para ver la correlación entre dos variables. La hipótesis nula es que no hay correlación entre las dos.

```
cor.test(x, y) # where x and y are numeric vectors.
```

```
cor.test(cars$speed, cars$dist)
#=> Pearson's product-moment correlation
#=>
#=> data: cars$speed and cars$dist
#=> t = 9.464, df = 48, p-value = 1.49e-12
#=> alternative hypothesis: true correlation is not equal to 0
#=> 95 percent confidence interval:
#=> 0.6816422 0.8862036
#=> sample estimates:
#=> cor
#=> 0.8068949
```

False Discovery Rate (FDR)

- Las correcciones de p-value cuando hacemos un gran número de test estadísticos son imprescindibles para asegurar que nuestro proyecto sea publicado y tenga una calidad decente.
- En algunos casos se baja el nivel de significancia (ej. 0.01) y en otros se aplica algunas correcciones como la Bonferroni o la de Benjamini_Hochberg
- https://rpubs.com/Joaquin_AR/236898