

Programación y Uso de Librerías en R: Herramientas de Análisis y Visualización de Datos

Juan Luis Peñaloza Figueroa
Universidad Complutense de Madrid

Milagros Dones Tacero
Universidad Autónoma de Madrid

Carmen Gladys Vargas Pérez
Universidad Complutense de Madrid

AÑO: 2025

SCRIPT_9: CAPÍTULO XI: PROBABILIDADES EN R

TABLAS DE FRECUENCIAS

Cargamos una variable cuantitativa discreta

```
> edad<-c(11,12,12,15,12,41)
```

```
> edad
```

Cargamos una variable cuantitativa continua

```
> altura=c(50,65,120,156,60,182)
```

```
> altura
```

Cargamos una variable cualitativa nominal

```
> sexo=as.factor(c("Hombre","Mujer","Mujer","Hombre","Mujer",  
"Mujer"))
```

```
> sexo
```

Cargamos una variable cualitativa ordinal

```
> niveleducativo=as.factor(c("Sin instrucción","Educ.  
Básica","Educ.Básica", "Universitaria", "Unviversitaria","Educ.  
Básica"))
```

```
> niveleducativo
```

Datos relacionados

```
> datos=data.frame(edad,altura,sexo,niveleducativo)
```

```
> datos
```

TABLAS DE FRECUENCIA SIMPLE

Tablas de frecuencias: sexo

```
> tab_sexo <- table(sexo)
```

```
> tab_sexo
```

Tablas de frecuencia: edad

```
> tab_edad <- table(edad)
```

```
> tab_edad
```

Tablas de frecuencias de doble entrada

```
> tab_sexo_edad <- table(sexo,edad)
```

```
> tab_sexo_edad
```

```
> prop.table(tab_sexo_edad, margin = 1)
```

Aplicando la función table.freq()

```
> tab_Edad <- hist(edad, plot=FALSE)
```

```
> tab_Edad2=table.freq(tab_Edad); tab_Edad2
```

VARIABLES ALEATORIAS

```
# Simulación de Montecarlo para variables discretas
> estudiantes <- rep(c("mujer", "hombre"), times = c(4, 6))
> estudiantes
# Simular un determinado número de veces el experimento
> estudiantes <- rep(c("mujer", "hombre"), times = c(4, 6))
> num_veces <- 100
> resultados <- replicate(num_veces, {
+   sample(estudiantes, 1)
+ })
> head(resultados)
# Proporción de cada categoría
> frec<- table(resultados)
> frec<-table(resultados) > prop.table(frec)
# Repetimos el experimento
> estudiantes <- rep(c("mujer", "hombre"), times = c(4, 6))
> num_veces<-10000
> resultados1<-replicate(num_veces,{
+ sample(estudiantes,1)
+ })
> frec1<-table(resultados1)
> prop.table(frec1)
```

COMBINATORIAS Y PERMUTACIONES

```
> install.packages("gtools")
> library(gtools)
> destud<- c("Jenny", "Freddy", "Yasan", "Iker", "Pamela",
"Alexandra", "Bladimir", "Enrique", "Karen", "Christian")
> result1<-permutations(10,3,v=destud)
> head(result1)
# Calcular la probabilidad de que Fredy gane la competencia y que
# Pamela quede en segundo lugar
> result1<-permutations(10,3,v=destud)
> totres<-nrow(result1)
> mean(result1[,1]=="Freddy" & result1[,2]=="Pamela")
# Combinaciones
> destud<- c("Jenny", "Freddy", "Yasan", "Iker", "Pamela",
"Alexandra", "Bladimir", "Enrique", "Karen", "Christian")
> combinations(10, 2, v = destud)
> result2 <- combinations(10, 2, v = destud)
> head(result2)
> nrow(resultados)
# Probabilidad
> mean((result2[, 1]=="Pamela" & result2[, 2]=="Enrique") |
(result2[, 1]=="Enrique" & result2[, 2]=="Pamela"))
# Generamos una muestra
> muestra <-sample(destud,2)
> head(muestra)
# Simulación de Monte-Carlo
> n<-10000
> result3<-replicate(n,{
+   equipo<-sample(destud,2)
+ cumple_condicion<-(equipo[1]=="Pamela" & equipo[2]=="Enrique"
|equipo[2]=="Pamela" & equipo[1]=="Enrique")
+   cumple_condicion
+ })
# Probabilidad pedida
```

```

> mean(result3)
# Replicar el experimento para confiar en los resultados de la simulación?
> n-veces <-10*2^(1:17)
> n_veces
> destud<-c("Jenny","Francisco","Yana","Iker","Pamela","Carla",
"Manuel", "Enrique","Ainoa","Cristobal")
# Construimos una función relacionada con n
> Prob_por_muestral<-function(n) {
+ resultado<-replicate(n,{
+ equipo<-sample(destud,2)
+ cumple_condicion<-(equipo[1]=="Pamela" &
equipo[2]=="Enrique"|equipo[2]=="Pamela" &equipo[1]=="Enrique")
+ cumple_condicion
+ })
+ mean(resultado)
+ }

## PROBABILIDAD UTILIZANDO FUNCIONES
> prob_por_muestra(10000)
> Prob_por_muestral(10000)
> Prob_por_muestral(20000)
> Prob_por_muestral(100000)
# Función "prob_por_muestra"
> prob1<-sapply(n_veces,prob_por_muestra)
> prob1
# Diagrama de dispersión
> install.packages("ggplot2")
> library(ggplot2)
> install.packages("dplyr")
> library(dplyr)
> prob2 <- data.frame(n = num_veces, probabilidad = prob)
> prob2 %>% ggplot() + aes(n, probabilidad) + geom_line() +
geom_point() + xlab("# de veces del experimento")
# Cumpleaños en clases
> días<- 1:365
> colegas<-sample(días, 50. Replace =T)
> head(colegas)
> duplicated(colegas)
> any(duplicated(colegas))
# Simulación de Montecarlo con 10 mil repeticiones
> nveces<-10000
> resultado<-replicate(nveces, {
+ colegas<-sample(días, 50, replace=T)
+ #retorna un valor lógico si hay duplicados
+ any(duplicated(colegas))
+ })
# Probabilidad
> mean(resultado)
# Estimamos la probabilidad dependiendo del número de estudiantes por clase
> clases1<-1:80
> prob<-sapply(clases1,calcula_prob)
> prob
> Probabilidad0<-data.frame(n=clases1,probabilidad=prob)
> head(Probabilidad0)
> Probabilidad0 %>%ggplot()+aes(n,probabilidad)+geom_point()+
xlab("Número de alumnos por clase")
# Experimento: vector de palos y vector de números → Crear la combinatoria y tener la
baraja completa
> numeros <- c("As", "Dos", "Tres", "Cuatro", "Cinco", "Seis",
"Siete","Ocho", "Nueve", "Diez", "Jack", "Reina", "Rey")

```

```

> palos <- c("de Corazones","de Diamantes","de Picas", "de Tréboles")
> combinatoria <- expand.grid(numero = numeros, palo = palos)
> paste(combinatoria$numero, combinatoria$palo)
> baraja<-paste(combinatoria$numero, combinatoria$palo)
> mean(baraja=="Rey de Diamantes")
# Creamos primero el vector de "Reina de..."
> reinas <- paste("Reina", palos)
> head(reinas)
# Cálculo de la probabilidad
> mean(baraja %in% reinas)

```

TABLA DE CONTINGENCIA

```

# Tabla de contingencia del futbol
> tabfutbol<-cbind(tabfutbol, Total = apply(tabfutbol, 1, sum))
> tabfutbol
> tabfutbol<-rbind(tabfutbol, Total = apply(tabfutbol, 2, sum))
> tabfutbol
# Tabla de contingencia del baloncesto
> x1 <-c(8,16)
> x2 <-c(7,3)
> tabbaloncesto<-rbind(x1,x2)
> tabbaloncesto
> colnames(tabbaloncesto)<-c("F","M")
> rownames(tabbaloncesto)<-c("FALSE","TRUE")
> tabbaloncesto
> tabbaloncesto<-table(datos$basquetbol,datos$sexo)
> tabbaloncesto
>tabbaloncesto<-cbind(tabbaloncesto,Total=apply(tabbaloncesto, 1,sum))
> tabbaloncesto<-rbind(tabbaloncesto,Total=apply(tabbaloncesto,
2,sum))
> tabbaloncesto
# Tabla de contingencia del Voleybol
> tabvoleybol<-table(datos$voleybol, datos$sexo)
> tabvoleybol
> tabvoleybol<-cbind(tabvoleybol,Total=apply(tabvoleybol,1,sum))
> tabvoleybol<-rbind(tabvoleybol,Total=apply(tabvoleybol,2,sum))
> tabvoleybol
# Tabla contingencia del Ajedrez
> tabajedrez<-table(datos$ajedrez, datos$sexo)
> tabajedrez
> tabajedrez<-cbind(tabajedrez,Total=apply(tabajedrez,1,sum))
> tabajedrez<-rbind(tabajedrez,Total=apply(tabajedrez,2,sum))
> tabajedrez

```

PROBABILIDADES Y FRECUENCIA RELATIVA

```

# Proporciones en tablas de contingencia
> f1<-tabfutbol
> f1
> n<-max(f1)
> n
> n<-length(datos$nombrres)
> n
# Determinar las proporciones de cada celda de la tabla
> tabfutbolprop <- round(f1/n, 4)
> tabfutbolprop
> f2<-tabbaloncesto
> tabbaloncestoprop<-round(f2/n,4)
> tabbaloncestoprop
# Consideremos una variable aleatoria X con distribución normal, media igual a 50 y varianza igual a 25
# Calcular la probabilidad de que X sea menor o igual a 48:  $P(X \leq 48)$ .

```

```

> pnorm(48, mean = 50, sd = sqrt(25))
# Calcular la probabilidad de que X sea mayor a 48:  $P(X > 48)$ .
> pnorm(48, mean = 50, sd = sqrt(25), lower.tail = FALSE)
# ¿Cuál es el valor de X que deja un 90% de los datos por bajo él?  $P(X \leq x_0) = 0,90$ 
> qnorm(0.90, mean = 50, sd = sqrt(25))
# Generar un conjunto de 12 datos que sigan una distribución normal de media 50 y varianza 25:
> rnorm(12, mean=50, sd=sqrt(25))
> curve(dnorm(x, mean = 50, sd = sqrt(25)), xlim = c(35,65), xlab =
"Valores de X", ylab = "Densidad de X")
# La función d, en este caso dnorm()
> y = 3 + 2x #Modelo determinista
> y = 3 + 2x + e #Modelo probabilístico
> x<-c(1:50)
> y1<-3+2*x
> y1
> y2<-3+2*x+rnorm(50, mean=3, sd=3) #e=rnorm(50,3,3)
> y2
> par(mfrow=c(1,2))
> plot(x,y2)
> plot(x,y1)

## VALOR ESPERADO
# Probabilidad de que un jugador gane cada vez
> jugadas<- 1
> probG <- 2/37
> probP <- 1 - probG
> muestra1<-sample(c(10,-1), jugadas, replace=T, prob=c(probG,probP))
> muestra1
# Cálculo del valor esperado:  $E(X) = a \cdot p + b \cdot (1-p)$ 
> a<-10
> b<- -1
> probG<-2/37
> probP<-1 - probG
> VE<-a*probG + b*ProbP
> jugadas<-40
> muestra2<-sample(c(10,-1), jugadas, replace=T, prob=
c(probG,probP))
> muestra2
> sum(muestra2)
# Calcular el valor esperado de la suma
> jugadas<-40
> probG<- 2/37
> probP<- 1 - probG
> a<-10
> b<- -1
# Valor esperado de la suma
> ESuma<-jugadas*(a*probG + b*probP)
> ESuma
# Error estándar de la suma
> SE_sum<-sqrt(jugadas)*abs(10- -1)*sqrt(probG*probP)
> SE_sum

## PROBABILIDAD CONDICIONADA
> install.packages("knitr")

```

```

> library(knitr)
# Calcular  $P(A/B) = ??$ 
# Inicializamos las probabilidades de las variables
> prob.A<-0.6
> prob.B<-0.4
> prob.A.Inter.B<-0.18
> prob.B.Inter.A<-0.18
> Prob.A.dado.B<-prob.A.Inter.B/prob.B
> Prob.A.dado.B
> paste("La prob de A dado B es:",Prob.A.dado.B*100,"%")
# Nos piden calcular  $P(B/A) = ??$ 
> Prob.B.dado.A<-prob.B.Inter.A/prob.A
> Prob.B.dado.A
# Ilustración 3. Exámenes
# inicializamos las probabilidades correspondientes
> p.Ex1<-0.45
> p.Ex1.Inter.Ex2<-0.30
> p.Ex2.dado.Ex1<-p.Ex1.Inter.Ex2/p.Ex1
> p.Ex2.dado.Ex1

## DIAGRAMA DE UN ÁRBOL DE DECISIÓN
# Datos del problema
> PA1<-0.6
> PA2<-0.4
> PG.PA1<-0.97
> PB.PA1<-0.03
> PG.PA2<-0.96
> PB.PA2<-0.04
# Árbol de decisión del proveedor 1
> PA1.I.G <-PA1*PG.PA1
> cat("P(G/A1) de que una pieza sea buena y sea suministrada sea el proveedor 1:",PA1.I.G)
> P(G/A1)
> PA1.I.B<-PA1*PB.PA1
> cat("P(B/A1) de que una pieza sea defectuosa y sea suministrada sea el proveedor 1:",PA1.I.B)
> P(B/A1)
## Árbol de decisión del proveedor 2
> PA2.I.G<-PA2*PG.PA2
> cat("P(G/A2) de que una pieza sea buena y sea suministrada por el proveedor 2:",PA2.I.G)
> P(G/A2)
> PA2.I.B<-PA2*PB.PA2
> cat("P(B/A2) de que una pieza sea defectuosa y sea suministrada por el proveedor 2:",PA2.I.B)
> P(B/A2)

## TEOREMA DE BAYES
> PA1<-0.6
> PA2<-0.4
> PG.PA1<-0.97
> PB.PA1<-0.03
> PG.PA2<-0.96
> PB.PA2<-0.04

# Probabilidad de que una pieza defectuosa (Bad) sea suministrada por el proveedor1.
> TB.PA1.B<- (PA1*PB.PA1) / (PA1*PB.PA1+PA2*PB.PA2)

```

```

> TB.PA1.B
[1] 0.5294118
> cat("TBayes: La prob de que una pieza defectuosa sea
suministrada por el proveedor1 es",TB.PA1.B)

## COEFICIENTE DE VARIACIÓN (CV)
> coef_var <- function(x, na.rm = FALSE) {
sd(x, na.rm=na.rm) / mean(x, na.rm=na.rm) }
# Calcular el CV para el vector w definido a continuación.
> w<-c(5,-3,NA,8,8,7)
# Creamos una función del coeficiente de variación
> coef_var<-function(x,na.rm=F) {
+ sd(x,na.rm=na.rm)/mean(x,na.rm=na.rm)
+ }
# Calculo del coeficiente de variación del vector 'w'
> coef_var(x=w,na.rm=T)

## TIPIFICACIÓN DE UNA VARIABLE
> z <- scale(vector, center= T, scale= T)
> ingresos<- c(2000,2500,3100,2700,2800,3000,2650,2200,
1800,2600,4100,2700,2900, 3900,5000)
> ingresos
> CV<-scale(ingresos,center=T, scale=T)
> ingresos_z<-scale(ingresos, center=T,scale=T)
> head(ingresos_z)

## TEOREMA DE MARKOV Y LAS CADENAS DE MARKOV
# Creamos la matriz de transición P:
> P = matrix(c(0,0.5,0.5,.5,0,.5,.5,.5,0),nrow = 3,byrow = TRUE)
> P
# creamos el objeto "markovchain"
> mc = new("markovchain",transitionMatrix=P,states=c("a","b",
"c"),name="Cadena 1")
> head(mc)
# Resumen de la cadena de markov 1
> summary(mc)
Cadena 1 Markov chain that is composed by:
Closed classes: a b c
Recurrent classes:
{a,b,c} Transient classes: NONE
The Markov chain is irreducible
The absorbing states are: NONE
# Para visualizar la transición de la cadena
> plot(mc)
# Probabilidad de transición
> transitionProbability(object = mc , t0="a", t1="c")
> mc[2,3]
# Computar la matriz de transición en n pasos
> n<-5 # número de pasos al futuro
> mc^5
#Caragamos los datos en R
>tprob<-matrix(c(0.7,0.21,0.09,0.12,0.7,0.18,0.01,0.5,0.49),nrow=3,
byrow=T)
> tprob
>cadena<-new("markovchain",states=c("Alta","Media","Baja"),
transitionMatrix = tprob, name = "cadena")
> cadena

```

```

# Distribución de (5%, 60%, 35%) para las 3 clases
> Inicial<-c(0.05,0.6,0.35)
> round(Inicial*cadena^3,3)
# Pregunta: Una persona de clase baja donde podría estar dentro de 3 generaciones
> Inicial<-c(0,0,1)
> round(Inicial*cadena^3,3)
# Lo mismo podemos hacer para la clase media
> Inicial<-c(0,1,0)
> round(Inicial*cadena^3,3)
# Concepto “steady” y autovalores
> steadyStates(cadena)
> steadyStates(cadena)*cadena^3
> plot(cadena)

```

SUCESIÓN DE VARIABLES ALEATORIAS

```

# Convergencia en Probabilidad
# Generamos una muestra
> runif(10)
> m<-100
> n<-10
> X<-matrix(runif(n*m), nrow=m,ncol=n)
> head(X,3)
> mean(X[1,])
> apply(X,1,mean)
> mean_samples <- function(n = 10){X = matrix(runif(n*m), nrow = m,
ncol = n) return(apply(X, 1, mean))
}
# Convergencia en Distribución
> m<-100
> mean_samples<-function(n=10){
+ X=matrix(rnorm(n*m),nrow=m,ncol=n)
+ return(apply(X,1,mean))
+ }
> B<-matrix(NA,100,20)
> for(i in 1:20){
+ B[,i]=mean_samples(i*10)
+ }
> colnames(B)=as.character(seq(10,200,by=10))
> boxplot(B)

```

INTEGRALES Y DERIVADAS AVANZADAS EN R

```

# Calcular la integral de la función  $f(x) = x$  entre 0 y 1
> integrate(function(x) x, lower=0,upper=1)
# Calcular la integral de la función  $f(x) = e^{-x}$  entre 0 e Inf
> integrate(function(x) exp(-x), lower=0,upper=Inf)
# Calcular la integral de la función  $f(x) = 2/(3-x)^{(1/2)}$  entre 0 y 3
> integrate(function(x) 1/sqrt(3-x), lower=0,upper=3)

```

CÁLCULO DE DERIVADAS EN R

```

> install.packages("Deriv")
> library(Deriv)
# Derivar la expresión:  $2x^2 + 2$ 
> f1<-function(x) 2*x^2+2
> f1
# Derivar la expresión:  $y^3 + 3y^2$ 
> g1<-expression(y^3+3*y^2)
> g1
# Derivar la expresión:  $(2 - x^3)^3$ 
> f3<-function(x) (2-x^3)^3

```



```
> f3prima<-Deriv(f3,"x")
> f3prima function (x)
# Gráfica de la derivada
> gx<-function(x) x^2
> gprima1<-Deriv(gx,"x")
> gprima1 function (x) 2 * x
> gprimax<-gprima1(x)
> plot(x,gprimax)
```