10508

Estadística - Introducción a R (2008)

R es un sistema para análisis estadísticos y gráficos creado por Ross Ihaka y Robert Gentleman. Éste tiene una naturaleza doble de programa y lenguaje de programación y es considerado como un dialecto del lenguaje S creado por los Laboratorios AT&T Bell.

R se distribuye gratuitamente bajo los términos de la *GNU General Public Licence*, su desarrollo y distribución son llevados a cabo por varios estadísticos conocidos como el *Grupo Nuclear de Desarrollo de R.*

Los archivos necesarios para instalar R, ya sea desde las fuentes o binarios pre-compilados, se distribuyen desde el sitio de internet *Comprehensive R Archive Network* (CRAN) junto con las instrucciones de instalación.

Una de las características más sobresalientes de R es su enorme flexibilidad. Mientras que programas clásicos muestran directamente los resultados de un análisis, R guarda estos resultados como un "objeto", de tal manera que se puede hacer un análisis sin necesidad de mostrar su resultado inmediatamente. El usuario puede extraer solo aquella parte de los resultados que le interesa. Esta característica de ser un lenguaje *Orientado a Objetos*, significa que las variables, datos, funciones, resultados, etc., se guardan en la memoria activa del computador en forma de *objetos* con un *nombre* específico. El usuario puede modificar o manipular estos objetos con *operadores* (aritméticos, lógicos, y comparativos) y *funciones* (que a su vez son objetos).

Su flexibilidad y el hecho de incorporar las técnicas estadísticas más recientes hace que cada vez este más implantado en todos los campos científicos.

Para cargar el paquete, módulo base, en la PC deberás seguir los siguientes pasos:

1) Buscar la dirección http://www.r-project.org/

Aparece la página principal : The R Project for Statistical Computing

2) Del lado izquierdo de esa ventana está la opción de "Download", hacer clik en CRAN, y luego seleccionar alguna dirección de un país cercano, por ejemplo Brasil o Chile para bajar el programa.

3) Luego como R está disponible en diversas formas, si la PC que tienes está bajo Windows, elegir la opción: "Windows (95 and later)" . En la nueva ventana seleccionar el módulo "base" y bajar el archivo ejecutable, que ocupa un espacio de 29MB, y se denomina "R-2.5.1-win32.exe"

4) Luego ejecutar el fichero. Instalará el sistema base y los paquetes recomendados.

1.1.1 Consultar la ayuda en R

R ofrece una ayuda por temas tecleando **help()**. De esta manera:

>help(nombre del comando)

ó también se puede acceder tecleando:

? y el nombre del comando

1.2 MANIPULACIÓN DE DATOS

1.2.1 Operaciones

Los operadores elementales son los usuales, +, -, *, / y. Por ejemplo, para calcular (7x3)+12/2-7^2+√4 escribimos:

```
>(7*3)+12/2-7^2+sqrt(4)
[1]-20
```

El símbolo **<-** se utiliza para asignar a x el valor 7:

```
>x<-7
```

1.2.2 Vectores

La función que R utiliza es **c(...).** Para crear el vector (1.5, 2, 3) se escribe:

```
>x<-c(1.5, 2, 3)
>x
[1] 1.5 2.0 3.0
```

y si se quiere elevar cada coordenada al cuadrado

```
> x^2
[1] 2.25 4.00 9.00
```

Podemos aplicar vectores para diversos usos estadísticos. Por ejemplo calcular los cuantiles muestrales de probabilidades indicas en un vector como (0.20, 0.30, 0.90):

```
> quantile(x,probs=c(0.20,0.30,0.90))
 20% 30% 90%
   2   3   9
```

Algunos comandos básicos relacionados son **max** y **min** que seleccionan el máximo y mínimo valor de los componentes de un vector respectivamente; **length(x)** es el número de elementos del vector x; **sum(x)** es la suma de todos los elementos y **prod(x)** es el producto, también indicado con **"*".** Las funciones **mean(x), var(x)** y **median(x)** proporcionan la media, la varianza y la mediana muestral respectivamente.

Del mismo modo podemos utilizar la función **c(...) con valores cualitativos:**

```
> y<-c("A","tabla","libro")
> y
[1] "A" "tabla" "libro"
```

Si tenemos dos o más vectores por ejemplo x=(2,3,4,1) e y=(1,1,3,7) y que remos organizar por columnas utilizamos el comando **cbind**

```
> x<-c(2,3,4,1)
> y<-c(1,1,3,7)
> cbind(x,y)
   x y
[1,] 2 1
[2,] 3 1
[3,] 4 3
[4,] 1 7
```

y si queremos organizar por filas utilizamos **rbind**

```
> rbind(x,y)
  [,1] [,2] [,3] [,4]
x   2   3   4   1
y   1   1   3   7
```

Para que cada columna de un archivo R se convierta en un vector. Por ejemplo si un archivo se llama "muestra":

```
>attach(muestra)
```

1.2.3 Secuencias

El comando **seq** sirve para crear secuencias de números, muy útiles por ejemplo en la creación de gráficos. Por ejemplo

```
> seq(0,1,0.2)
[1] 0.0 0.2 0.4 0.6 0.8 1.0
```

crea la secuencia de 0 a 1 con intervalos de amplitud 0.2. Si la amplitud es 1 basta con escribir el principio y el final de la secuencia de la forma:

```
> 0:8
[1] 0 1 2 3 4 5 6 7 8
```

El comando **rep(a,n)** se utiliza para repetir un número o carácter a, n veces. Por ejemplo:

```
> rep(1,5)
[1] 1 1 1 1 1
```

También podemos **combinar c(...) y rep(...,n)** del siguiente modo:

```
> rep(c(0,"x"),3)
[1] "0" "x" "0" "x" "0" "x"

> rep(c(1,2,3),length=10)
 [1] 1 2 3 1 2 3 1 2 3 1

> c(rep(1,3),rep(2,3),rep(3,3))
[1] 1 1 1 2 2 2 3 3 3

> rep(c(1,2,3),each=2)
[1] 1 1 2 2 3 3
```

Los **corchetes** que preceden a las secuencias de resultados permiten indicar los componentes de la secuencia obtenida. Sea x la secuencia de 0 a 30 de amplitud 10:

```
> x<-seq(0,30,10)
> x
[1]  0 10 20 30
```

Para seleccionar el valor que ocupa el tercer lugar tecleamos:

```
> x[3]
[1] 20
```

Para seleccionar los valores que ocupan el primero y segundo lugar tecleamos:

```
> x[1:2]
[1]  0 10
```

Si los valores no son continuos por ejemplo el primero y el tercero:

```
> x[c(1,3)]
[1]  0 20
```

Para obtener todos los valores del vector excepto alguno, por ejemplo todos excepto el segundo y el tercero, escribimos:

```
> x[-c(2,3)]
[1]  0 30
```

1.2.4 Operadores lógicos y valores perdidos

Para trabajar con operadores lógicos se utilizan símbolos

Símbolo	Función
<	menor que
>	mayor que
<=	menor o igual que
>=	mayor o igual que
==	igual que
!=	distinto de
|	uno u otro

Por ejemplo podemos obtener valores menores de 20 en una muestra dada, de la siguiente manera:

```
> x<-c(10,15,19,20,25,30,17)
> x[x<20]
[1] 10 15 19 17
```

1.3 DISTRIBUCIONES

R proporciona las funciones de densidad, de distribución y los cuantiles o percentiles de las distribuciones más importantes. También existen funciones que simulan muestras aleatorias de dichas distribuciones. La primera letra de cada comando indica la función. Por ejemplo, si q y x son vectores de cuantiles o percentiles, p es un vector de probabilidad y n el tamaño de la muestra, entonces en una distribución N(0,1):

- **pnorm(x, mean=0, sd=1)** calcula $F_X(x) = P(X \leq x)$
- **qnorm(q, mean=0,sd=1)** calcula x_q tal que $P(X \leq x_q) = q$
- **dnorm(x, mean=0,sd=1)** calcula $f_X(x)$
- **rnorm(n, mean=0,sd=1)** proporciona una muestra aleatoria de tamaño n procedente de una distribución N(0,1).

Las distribuciones disponibles más importantes y a las que se les puede aplicar los prefijos anteriores son:

Distribuciones	Nombre	Parámetros
binomial	binom	n, p
ji-cuadrado	chisq	g.l.=n
exponencial	exp	λ
F	f	g.l.=(m,n)
gamma	gamma	α, λ
normal	norm	μ,σ
Poisson	pois	λ
t de Student	t	g.l.=n
uniforme	unif	a,b
beta	beta	α,β

1.4 CREACIÓN DE FUNCIONES

R es un lenguaje que permite crear nuevas funciones. Una función **se define** por la asignación de la forma:

>nombre<-función(argumento1, argumento2,....){expresión}

La **expresión** es una fórmula o grupo de fórmulas que utilizan los **argumentos1, argumento2**, etc. para calcular su valor. El valor de dicha expresión es el valor que R proporciona en su salida. Por ejemplo una función para calcular valores de una densidad normal estándar:

>dnormalest<-function(x){(1/sqrt(2*pi)*exp(-(x^2)/2)}

también puede servirnos para calcular valores defunciones que utilizan indicadoras, como $f(x)=(x-1)\, I_{(1,2)}(x)+(3-x)\, I_{(2,3)}(x)$:

```
>dfun<-function(x){
if (x>1&x<2){x-1}
else if (x>2&x<3){3-x}
else if {0}
}
```

1.5 Gráficos

El comando **plot** es uno de los más utilizados para realizar gráficos. Si escribimos plot(x,y) donde x e y son vectores con n coordenadas, entonces R los pares de puntos tomando las coordenadas de ambos vectores($P_i=(x_i,y_i)$ para i=1,...,n).

Función	Descripción
• **plot** coordenadas de los vectores.	Representa las n- uplas de puntos tomando las
• **points(x,y,...)**	Añade puntos a un gráfico ya existente.
• **lines(x,y,...)**	Añade líneas a un gráfico ya existente.
• **segments(x_1,y_1,x_2,y_2)** en la figura.	Añade el segmento AB con a=(x_1,y_1) y B=(x_2,y_2)
• **title**(título) de línea se teclea **/n**.	Añade título a la figura ya existente. Para saltar
• **boxplot(x)** x.	Realiza el diagrama de cajas del vector numérico
• **Hist(x,nclass=n, breaks=c(...),probability=T)**	Dibuja el histograma de frecuencias relativas de la variable x con n intervalos y tomando las fronteras en los componentes del vector c(...).
• **persp(x,y,z)**	Crea un gráfico en perspectiva a partir de los datos en forma de x,y,z , donde x e y son vectores y z es una matriz.
• **polygon(x,y)** coordenadas los valores de vectores x e y.	Dibuja un polígono cuyos vértices tienen por

Para poder elegir en cada gráfico la forma, el tamaño, el color, los textos añadidos, etc., existe una amplia colección de opciones dentro del comando **plot**.

Parámetro	Descripción
• **type="p"**	Los datos los representa con puntos.
• **type="l"**	Línea conectando los datos.
• **type="h"**	Los datos los representa con barras verticales.
• **type="b"**	Puntos y líneas uniendo los puntos.
• **xlab="nombre eje x"**	Nombre en el eje OX.
• **ylab="nombre eje y"**	Nombre en el eje OY
• **xlim=c(xmin,xmax)**	Escala en el eje OX.
• **ylim=c(ymin.ymax)**	Escala en el eje OY.
• **pch="*"**	Carácter para los puntos(ejemplo pch="P")
• **lwd=1**	Ancho de línea(1= por defecto, 2= doblemente ancha, etc.
• **Col=1**	Color.