USO DE "R" APLICADO A LAS CIENCIAS VETERINARIAS

Roberto Bustillos Huilca, MVZ, MSc.

Universidad Nacional de Loja

Carrera de Medicina Veterinaria y Zootecnia

Loja, Enero 2019

Contenido

Estadística básica.

- Los datos obtenidos cuando realizamos cualquier experimento presentan variabilidad:
  - El peso de un ternero al nacer varía.
  - La cantidad de lluvia recogida en un día en una determinada zona varía.
  - La altura de una planta sometida a dos tipos de abono varía, etc.
- La Estadística es una disciplina que se ha desarrollado en respuesta a los experimentadores cuyos datos exhiben variabilidad.
- Los conceptos y métodos de la estadística nos permiten describir la variabilidad, planificar la investigación teniéndola en cuenta y analizar los datos para extraer el máximo de información de los mismos así como determinar la fiabilidad de las conclusiones que podamos obtener a partir de estos datos.
- R es un lenguaje que permite implementar técnicas estadísticas.

Estadística básica.

- Variable $\longrightarrow$ Característica de interés.
- Muestra observada $\longrightarrow$ Conjunto de valores de la variable observados obtenidos de manera homogénea.
- Tamaño muestral $\longrightarrow$ Número de datos observados.
- La manera de describir la muestra (nuestros datos) depende del tipo de atributo:
  - Cualitativo Intrínsecamente no tiene carácter numérico (categórica).
    - Nominal (sin orden entre los valores): Sexo.
    - Ordinal (con valores ordenados): Nivel de estudios.
  - Cuantitativo Intrínsecamente numérico.
    - Discreto (cantidad finita o numerable de valores): Número de hijos.
    - Continuo (valores en toda la recta real): Altura.

Distribución de Probabilidad.

- R tiene las distribuciones de probabilidad más comunes implementadas en la librería BASE.
- Para cada una de ellas (distrib), disponemos de 4 versiones:

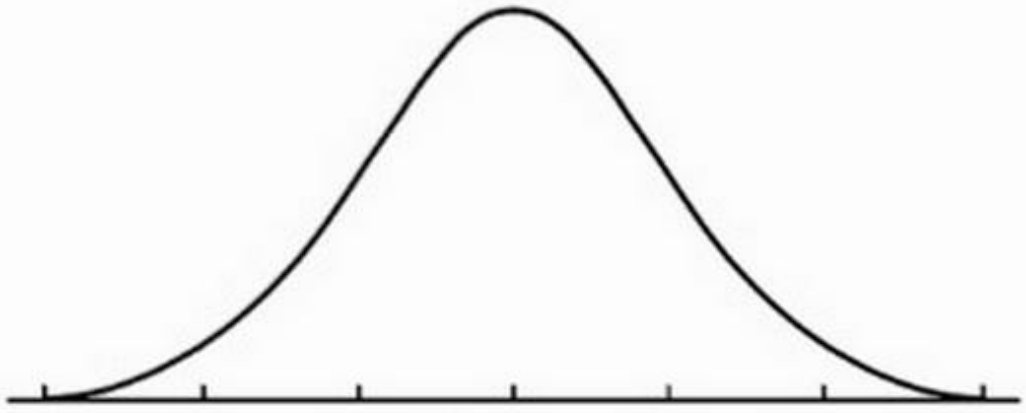
generador de numeros aleatorios	rdistrib
función densidad/probabilidad	ddistrib
función distribución	pdistrib
función inversa distribución (cuantiles)	qdistrib

Examples

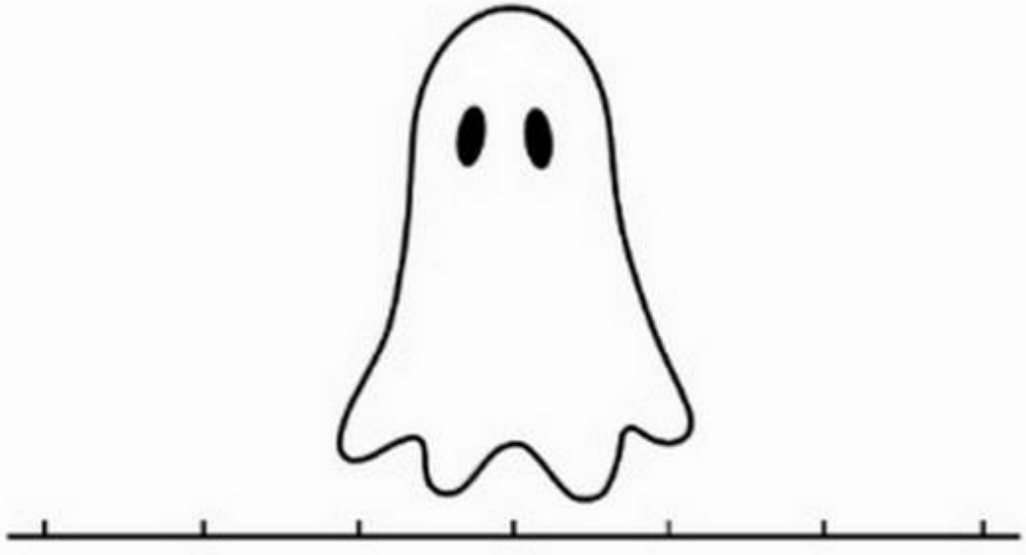
x.norm <- rnorm(500) Simulación de 500 datos normales

Distribuciones de probabilidad en la librería BASE.

Función	Utilidad
Normal	rnorm(n, mean=0, sd=1)
exponencial	rexp(n, rate=1)
gamma	rgamma(n, shape, scale=1)
Poisson	rpois(n, lambda)
Weibull	rweibull(n, shape, scale=1)
Cauchy	rcauchy(n, location=0, scale=1)
beta	rbeta(n, shape1, shape2)
t de Student	rt(n, df)
F (Snedecor)	rf(n, df1, df2)
Pearson χ^2	rchisq(n, df)
binomial	rbinom(n, size, prob)
geométrica	rgeom(n, prob)
hypergeométrica	rhyper(nn, m, n, k)
logística	rlogis(n, location=0, scale=1)
lognormal	rlnorm(n, meanlog=0, sdlog=1)
binomial negativa	rnbinom(n, size, prob)
uniforme	runif(n, min=0, max=1)



Normal Distribution



Paranormal Distribution

Tablas de frecuencia para variables categóricas.

- Hemos visto que un factor es un vector utilizado para especificar una clasificación discreta de los elementos de otro vector de igual longitud,
- y que en R existen dos tipos de factores (variables categóricas).
- Del mismo modo, dos factores definen una tabla de doble entrada, y así sucesivamente.
- La función table( ) calcula tablas de frecuencias a partir de factores de igual longitud.

Examples

```
x <- as.factor(1:5)
table(x)
age <- gl(4,5,15,labels=c("Terneros","Vaconas","Vacas","Toros"))
table(age)
```

Medidas de localización, dispersión y forma para variables cuantitativas continuas.

- La forma más sencilla de empezar a describir unos datos cuantitativos es realizar un resumen estadístico.
- Para analizar la curtosis y la asimetría de unos datos podemos utilizar dos funciones de la librería e1071: skewness( ) y kurtosis( ).

Examples

```
a <- rgamma(50,1,3)
summary(a)
mean(a); median(a); quantile(a)
sd(a); var(a); range(a); IQR(a)
min(a); which.min(a)
max(a); which.max(a)
```

Medidas de localización y dispersión más comunes.

Función	Utilidad
sum(..., na.rm=FALSE)	Suma
max(..., na.rm=FALSE)	Máximo
min(..., na.rm=FALSE)	Mínimo
which.min(x)	Posición del máximo
which.max(x)	Posición del mínimo
pmax(...,na.rm=FALSE)	Máximo en paralelo
pmin(...,na.rm=FALSE)	Mínimo en paralelo
cumsum(x), cumprod(x)	Sumas y prods acumulados
cummax(x), cummin(x)	max's y min's acumulados
mean(x, trim=0, na.rm=FALSE)	Media
weighted.mean(x,w,na.rm=FALSE)	Media ponderada
median(x,na.rm=FALSE)	Mediana
quantile(x,prob=(0,0.25,0.5,0.75,1),na.rm=F)	Cuantiles
fivenum(x, na.rm=FALSE)	5-Tukey: min, lower-hinge mediana, upper-hinge, máximo
summary(x, na.rm=FALSE)	min,1c,mediana,media,3c,max
IQR(x, na.rm=FALSE)	Rango inter-cuartílico
range(...,na.rm=FALSE, finite=FALSE)	Rango
var(x, y=x, na.rm=FALSE, use)	Varianza
sd(x, na.rm=FALSE)	Desviación Típica
mad(x,center,constant=1.4426, na.rm=FALSE)	Desviación mediana absoluta

Examples

library(e1071)

Consideremos dos distribuciones asimétricas (Betas) y las vamos a comparar con la normal que es simétrica:

```
sim <- 5000
s1 <- skewness(rbeta(sim,2,3))
s2 <- skewness(rbeta(sim,3,2))
s3 <- skewness(rnorm(sim,0.5,0.5))
s1; s2; s3
```

Consideremos una distribución normal y una Student, más achatada, y las comparamos.

```
k1 <- kurtosis(rnorm(sim))
k2 <- kurtosis(rt(sim,3))
k1; k2
```

Descripción gráfica de datos en R.

- Los datos son la mejor forma de simplificar lo complejo. Un buen gráfico suele ser más accesible que una tabla. Sin embargo es muy importante tener claro que gráfico queremos hacer.
- Las facilidades gráficas de R constituyen una de las componentes más importantes de este lenguaje.
- R incluye muchas y muy variadas funciones para hacer gráficas estadísticas estándar.
- Permite además construir otras nuevas a la medida del usuario (aunque a veces hacer cosas simples no es fácil).
- Permite exportar gráficas en distintos formatos: PDF, JPEG, GIF, etc.
- Otra alternativa es ggplot.

Gráficas en R.

- R tiene dos sistemas de producir gráficos:
  - El tradicional, que es el que veremos principalmente.
  - Gráficos Trellis (paquete Lattice) del que veremos algunos ejemplos.
- Podemos dividir los comandos para efectuar las gráficas en tres grupos:
  - Funciones para crear gráficas de alto nivel, es decir ya programadas y que admiten diferentes posibilidades.
  - Funciones de bajo nivel, que permiten un control más fino del dibujo y permiten crear gráficas a medida.
  - Funciones para el uso interactivo, para extraer información de una gráfica o una modificación mediante el ratón.

Función plot( )

- La función plot( ) es el procedimiento gráfico de alto nivel más habitual para dibujar datos.

Examples

```
x <- (0:65)/10
y <- sin(x)
plot(x)
plot(x,y)
plot(x,y,main="Función Seno")
z <- cos(x)
windows( ) Crea una ventana nueva
plot(x,z, main="Función Coseno")
```

Opciones de la función plot( ).

Algunas de las más útiles

- `main`: Cambia el título del gráfico
- `sub`: Cambia el subtítulo del gráfico
- `type`: Tipo de gráfico (puntos, líneas, etc.)
- `xlab, ylab`: Cambia las etiquetas de los ejes
- `xlim, ylim`: Cambia el rango de valores de los ejes
- `lty`: Cambia el tipo de línea; `lwd`: Cambia el grosor de línea
- `col`: Color con el que dibuja

Examples

```
plot(x,y,main="Seno",type="l")
plot(x,z,main="Coseno",lty=2,col="red",type="l")
plot(x,z,main="Coseno",lty=3,col="blue",type="l",xlim=c(0,2),
ylab="cos(x)")
```

Procedimientos de bajo nivel.

Se puede dibujar sobre la gráfica ya creada:

Los más habituales

- `points(x, y, ...)`: Dibuja una nube de puntos
- `lines(x, y, ...)`: Dibuja una línea que une todos los puntos
- `ablines()`: Dibuja una línea recta dada la interc. y pendiente
- `polygons(x, y, ...)`: Dibuja un polígono cerrado
- `text(x, y, labels, ...)`: Escribe texto en unas coordenadas

Examples

```
plot(x,y,main="Funciones seno y coseno",type="l")
lines(x,z,col="blue",lty=2)
text(x=c(0.5,0.5),y=c(0,1),labels=c("sin(x)","cos(x)"),
col=c("black","blue"))
```

Leyendas.

Descripción

La función `legend(x, y, legend, ...)` permite añadir leyendas a un gráfico:

- `x,y` : Esquina sup. izda. de la leyenda
- `legend`: Texto de la leyenda
- `bty`: Tipo de borde ("n" para omitir)

Examples

```
plot(x,y,main="Funciones seno y coseno",type="l")
lines(x,z,col="blue",lty=2)
legend(x=3,y=1,legend=c("sin(x)","cos(x)"),lty=c(1,2),
col=c("black","blue"))
```

Funciones gráficas interactivas.

Existen una serie de funciones que permiten completar los gráficos de manera interactiva por parte del usuario.

Descripción

- `identify(x, y, etiquetas)` identifica los puntos con el ratón y escribe la correspondiente etiqueta.
- `locator()` devuelve las coordenadas de los puntos.

Examples

```
plot(x,y,main="Funciones seno y coseno",type="l")
lines(x,z,col=2,lty=2)
legend(locator(1),legend=c("sin(x)","cos(x)"),lty=c(1,2),col=c(1,2))
s <- 1:10 ; t <- sample(1:10)
nombres <- paste("punto",s,".",t,sep="")
plot(s,t); identify(s,t,labels=nombres)
```

Gráficos para datos discretos.

Los más habituales

Para representar variables categóricas o cuantitativas discretas (con pocas clases):

- Diagramas de puntos: `dotplot()`
- Diagramas de barras: `barplot()`
- Diagramas de quesos: `pie()`

Examples

```
library(lattice)
a <- rbinom(100,5,0.3)
dotplot(table(a),horizontal=F)
par(mfrow=c(2,2)); plot(a,type="h")
barplot(table(a),col=rainbow(length(table(a))))
pie(table(a))
```

Gráficos para datos continuos.

Los más habituales

- Diagramas de cajas: `boxplot()`
- Diagramas de tallo y hojas: `stem()`
- Diagramas de puntos: `stripchart()`

Examples

```
library(lattice)
b <- rnorm(100); b.f <- rbinom(100,5,0.3)
stem(b)
par(mfrow=c(2,2))
boxplot(b); boxplot(b~b.f); boxplot(split(b,b.f),col="cyan")
stripchart(b)
stripchart(b,method="jitter",add=T,at=1.2)
stripchart(round(b,1),method="stack",add=T,at=0.7)
```

Representación de datos multivariantes.

Diversos tipos de gráficos para representar varias variables conjuntamente (relaciones entre ellas).

Los más habituales

- Gráficos de tendencias para tablas de contingencia: `dotchart()`
- Gráficos de dispersión: `plot()` y `pairs()`
- Gráficos condicionados: `coplot()`.

Examples

Gráficos de tendencia para tablas de contingencia

```
data(VADeaths)
dotchart(VADeaths,main="Death Rates in Virginia - 1940")
```

Gráficos condicionados

```
data(quakes)
coplot(lat~long|depth,data=quakes)
```

Examples

Gráfico de dispersión para revisar relaciones entre variables.

```
W <- matrix(rnorm(1000),ncol=2);colnames(W) <- c("a","b")
plot(W)
W <- matrix(rnorm(1000),ncol=5);colnames(W) <-
c("a","id","edad","loc","peso")
pairs(W)
data(iris)
razas <- unclass(iris$Species)
plot(iris[1:2],pch=21,bg=c("red","green3","blue")[razas])
pairs(iris[1:4],main="Anderson Iris Data-3 species",
pch=21,bg=c("red","green3","blue")[razas])
data(swiss)
pairs(swiss,panel=panel.smooth,lwd=2,cex=1.5,col="blue")
```

Gráficos para estudiar la distribución de unos datos.

Para estudiar la posible distribución de unos datos tenemos diferentes funciones.

Los más habituales

- Histogramas: `hist()`
- Gráficos *qq*: `qqplot()`, `qqnorm()` y `qqline()`. Dos posibles usos:
  - Comparación de cuantiles empíricos versus cuantiles teóricos: para comprobar si los datos se parecen a una determinada distribución
  - Comparación de dos distribuciones empíricas entre sí
- Estimación de la función de distribución empírica: `ecdf()`
- Estimación kernel de la función de densidad: `density()`

Examples

Histogramas

z <- rnorm(500); hist(z); hist(z,5)

Gráficos qq.

Examples

y <- rnorm(500)
Comparación de los cuantiles muestrales con los de una Normal
qqnorm(y); qqline(y)
Comparación de los cuantiles muestrales de dos muestras
y.t <- rt(500,3)
qqplot(y,y.t,xlab="Dist. Normal",ylab="Dist. St(3)"); qqline(y)
Comparar cuantiles muestrales con los de una distribución dada
library(lattice)
qqmath(y,distribution=function(p)qt(p,df=5))
qqmath(y,distribution=function(p)qgamma(p,shape=3,rate=5))

Representación en 3D.

Los más habituales

- Gráficos en tres dimensiones: `image()`
- Gráficos de contorno: `contour()`. Permite añadir líneas de nivel.
- Las librerías `MASS` y `ks` tienen funciones para estimar kernels bivariantes.

Examples

```
y <- seq(-1,1,0.05); z <- seq(-1,1,0.05)
f <- function(y,z) cos(z)/(1+y^2)
w <- outer(y,z,f)
image(y,z,w); contour(y,z,w,add=T)
```

Función par.

Colocar varias gráficas en una ventana

Los siguientes parámetros permiten diseñar el número de gráficas en cada dispositivo gráfico

- `mfrow`: Nº de filas y columnas en la ventana. Los huecos se rellenan por filas.
- `mfcol`: Ídem pero se rellena por columnas.

Examples

```
x <- (0:65)/10
y <- sin(x)
par(mfrow=c(1,2))
plot(x,y,main="Seno",type="l",ylab="sin(x)")
plot(x,y,main="Seno",type="l",ylab="sin(x)")
```

Exportando gráficos.

- Para guardar una gráfica, podemos copiar y pegar desde la ventana gráfica a un tratamiento de textos que los permita.
- Lo mejor es enviar directamente la gráfica a un dispositivo (pdf, postscript, etc.) utilizando funciones como pdf( ) o postscript ( ).

Examples

```
pdf("prueba.pdf",paper="special",width=13,height=7)
hist(x <- rnorm(100),prob=T)
dev.off( )
```