Introducción a R

Sesión 1
Una Visión General

Joaquín Ortega Sánchez

Centro de Investigación en Matemáticas, CIMAT
Guanajuato, Gto., Mexico
Oficina A-1, email: jortega@cimat.mx, http://www.cimat.mx/∼jortega

Verano de Probabilidad y Estadística
Junio-Julio 2009

Outline

Introducción

Probabilidad

Simulación

Gráficas

Modelación

Outline

Introducción

Probabilidad

Simulación

Gráficas

Modelación

Introducción

- R es una implementación gratuita de S, un ambiente de programación para estadística desarrollado en Bell Labs a partir de los años 70.
- Disponible en la red a través del Comprehensive R Archive Network (CRAN). La dirección en internet es http://cran.r-project.org/.

Introducción

Antes de comenzar vamos a cargar desde el servidor Mercurio el material del curso. Para esto vamos al menú Inicio y seleccionamos *Ejecutar*. En la ventana que se abre escribimos

\\ mercurio\temp\Verano08R

Se abre una nueva ventana con cinco documentos en formato pdf, que son las presentaciones correspondientes a las cinco sesiones del curso, y una carpeta llamada guiones. Las presentaciones no las van a necesitar para lo que vamos a hacer en clase, pero sí para repasar o completar lo que no tengamos tiempo de hacer.
Por otro lado, la carpeta de guiones tiene las instrucciones que deben ejecutar a lo largo a cada clase.

Introducción

Hay un guión para cada clase con nombre `Veranok` con $k = 1, \ldots, 5$.

Para poder tener acceso a ellos durante las secciones tenemos que colocar los guiones en la siguiente carpeta:

C:\Archivos de programa\R\R-2.8.0\curso

Una vez colocados allí tenemos fácil acceso a ellos desde `R`

Ahora abrimos el programa haciendo 'click' dos veces sobre el ícono de `R` en el escritorio.

Introducción

- Al abrir el programa se abre una ventana llamada '`R console`' que llamaremos la consola. En esta ventana se escriben las instrucciones que ejecutará el programa.
- El símbolo > en la consola es el apuntador ('prompt') e indica que el programa está listo para recibir instrucciones.
- En lo que sigue escribiremos los comandos de R precedidos por el símbolo del apuntador, y usaremos letras de tipo `typewriter` para destacar el texto que se escribe en la consola.
- La consola es útil para comandos sencillos que van a ser ejecutados de inmediato.

Introducción

Veamos algunos ejemplos:

```
> 25*43
  [1] 1075

> (2596 - 3361)^67
  -1.604413e+193

> sqrt(5)
  2.236068
```

Introducción

- Con frecuencia resulta más conveniente usar otra ventana llamada 'script' para interactuar con R.
- Permite escribir instrucciones sin que se ejecuten de manera automática, y así desarrollar un guión o 'script' constituido por una serie de instrucciones concatenadas.
- Los 'scripts' pueden ser guardados para ser ejecutados o modificados posteriormente.
- Lo que vamos a realizar a continuación se encuentra en un 'script' llamado `Verano1`. Para abrirlo vamos al menú 'File' y seleccionamos 'Open script...'. Luego abrimos la carpeta `curso` y abrimos `Verano.1`.

Introducción

- Para ejecutar las instrucciones hay que seleccionarlas y luego presionar 'Control-r' o bien presionar el botón derecho del ratón y seleccionar 'Run line or selection'. Al hacer esto, las instrucciones seleccionadas se ejecutan en la consola y los resultados correspondientes aparecen allí.
- En esta primera sesión introductoria haremos un recorrido rápido sobre algunas de las facilidades más interesantes de R, tanto de visualización como de análisis, con un mínimo de explicaciones.
- Las dos primeras instrucciones que aparecen cargan librerías de programas y rutinas que vienen con la distribución estándar de `R` pero no se cargan automáticamente. Las ejecutamos de una vez.

Outline

Introducción

Probabilidad

Simulación

Gráficas

Modelación

Probabilidad

- Escogemos al azar 16 estudiantes de una población con 30 % de mujeres. ¿Cuál es la probabilidad de que ninguno, uno, dos, . . . , dieciseis de los estudiantes seleccionados sean mujeres?

```
> round(dbinom(0:16, 16, 0.3),3)
 [1] 0.003 0.023 0.073 0.146 0.204 0.210
 [7] 0.165 0.101 0.049 0.019 0.006 0.001
[13] 0.000 0.000 0.000 0.000 0.000
> plot(dbinom(0:16, 16, 0.3), type='h',
   xlab='Numero de mujeres',
   ylab='Probabilidad', lwd = 2)
```

Probabilidad

- ¿Cuál es la probabilidad de que dos o menos mujeres sean escogidas?

```
> pbinom(2,16,0.3,lower.tail=T)
 [1] 0.09935968
```

- Generar 100 muestras aleatorias para esta situación.

```
> rbinom(100,16,0.3)
  [1]  4  5  4  5  4  3  5  6  1  7  5  0  5  7  5  5  6
 [18]  4  4  7  2  2  5  7  5  5 11  7  7  4  6  4  5  7
 [35]  8  4  4  5  4  5  2  8  5  5  5  6  4  4  4  5  4
 [52]  7  4  7  5  3  4  5  7  8  3  5  6  6  9  5  4  8
 [69]  8  4  3  7  4  5  1  3  6  3  6  4  6  7  6  7  4
 [86]  6  3  2  5  5  8  4  4  1  4  2  8 10  5  4
```

Probabilidad

- Como segundo ejemplo consideramos el problema de los cumpleaños: Si en un salón hay n estudiantes ¿cuál es la probabilidad de que al menos dos de ellos tengan el mismo cumpleaños?
- Supondremos:
  - que hay 365 días en un año (no hay bisiestos)
  - todos los días tienen igual probabilidad
  - los cumpleaños de los estudiantes son independientes
- Sea C el evento que nos interesa y A su complemento, es decir, que no haya dos estudiantes o más estudiantes en el grupo que cumplan años el mismo día.

Probabilidad

- La probabilidad de A es

$$P(A) = \frac{365 \cdot 364 \cdots (365 - n + 1)}{365^n} = \frac{365!}{(365 - n)!365^n}$$

y por lo tanto

$$P(C) = 1 - \frac{365!}{(365 - n)!365^n}$$

- Si tratamos de calcular directamente el factorial de 365, el programa emite un mensaje de error porque el número es demasiado grande para poder calcularlo directamente.

Probabilidad

- Sin embargo, es posible calcularlo usando la función `lfactorial()`, que calcula el logaritmo del factorial, y luego usar la función exponencial:

```
prob <- function(n)
    1 - exp(lfactrial(365)
      - lfactorial(365 -n)*log(365))
> prob(c(20, 22, 25, 30))
> [1] 0.4114384 0.4756953 0.5686997 0.7063162
> plot(1:100, prob(1:100))
> plot(1:100, prob(1:100), pch=15)
```

Probabilidad

- Como último ejemplo de esta sección vamos a calcular la probabilidad de las distintas manos que se pueden dar en un juego de poker.
- La siguiente tabla muestra las posible manos

pareja	dos cartas de igual valor
dos pares	dos cartas de igual valor y dos de otro
trío	tres cartas de igual valor
escalera	cartas en orden ascendente
full house	pareja y trío
color	todas las cartas de igual pinta
escalera de color	cartas en orden ascendente de igual pinta
poker	cuatro cartas de igual valor
escalera real	escalera de color terminando en as

Probabilidad

```
> poker <- data.frame(
  nada = choose(13,5)*4^5 - 10*4^5
      - 4*choose(13,5) + 10*4,
  par = 13* choose(4,2)*choose(12,3)*4^3,
  dos.pares = choose(13,2)*choose(4,2)^2*11*4,
  trio = 13*choose(4,3)*choose(12,2)*4^2,
  escalera = 10*4^5 - 10*4,
  full.house = 13*choose(4,3)*12*choose(4,2),
  color = 4*choose(13,5) - 10*4,
  cuatro = 13*choose(4,4)*12*4,
  esca.color = 10*4 - 4,
  esca.real = 4)
> sum(poker) - choose(52, 5)
> poker / choose(52,5)
```

Outline

Introducción

Probabilidad

Simulación

Gráficas

Modelación

Simulación

- Teóricamente, dos muestras independientes con distribución normal deben tener correlación cero. Sin embargo, en la práctica esto no es cierto en general. Veamos un ejemplo con dos muestras de tamaño 20 de la distribución normal típica y calculamos la correlación ρ entre ellas.

Simulación

```
> nn <- 20
> muestra.1 <- rnorm(nn)
> muestra.2 <- rnorm(nn)
> cor.test(muestra.1,muestra.2)

        Pearson's product-moment correlation

data: muestra.1 and muestra.2
t = 0.0583, df = 18, p-value = 0.9542
alternative hypothesis: true correlation is
     not equal to 0
95 percent confidence interval:
 -0.4314117 0.4534957
sample estimates:
       cor
0.01373032
```

Simulación

```
> plot(muestra.1, muestra.2, pch=16)
> abline(h=0, lty=2)
> abline(v=0, lty=2)
> title(paste('r=',
      round(cor(muestra.1,muestra.2),3)))
```

Simulación

Supongamos ahora que tenemos dos muestras aleatorias de distribuciones normales típicas y queremos decidir si son independientes o no mirando su correlación. Nuestra 'hipótesis nula' es que sí lo son.

Hemos visto que la correlación puede ser distinta de cero aun cuando las muestras sean independientes. Por lo tanto siempre podemos cometer dos tipos de error:

- Podemos decir que no son independientes cuando sí lo son.
  Esto se llama un error de Tipo I.
- Podemos decir que son independientes cuando no lo son.
  Esto es un error de Tipo II.

Simulación

Ahora parece razonable que si la correlación que obtenemos es *pequeña*, nos inclinemos por la opción de independencia, mientras que si es *grande* nos inclinemos más bien por la ausencia de independencia.

Esto quiere decir que escogemos un intervalo $(-\eta, \eta)$ y si la correlación empírica de nuestra muestra cae dentro de este intervalo decimos que las muestras son independientes, mientras que si cae fuera decimos que no lo son (o al menos que la evidencia no soporta la hipótesis de independencia).

¿Cómo escogemos η?

Simulación

Un posible enfoque es tratar de controlar los errores que podemos cometer, es decir, escoger η de modo que la probabilidad de cometer un error de Tipo I esté controlada.

Podemos pedir, por ejemplo, que si tomamos dos muestras de tamaño 20 al azar, la probabilidad de cometer este tipo de error sea menor que 0.9.

¿Cómo determinamos η para que esto sea cierto?

Vamos a hacerlo por simulación.

Simulación

Para hacer esto vamos a simular 1000 veces este problema:

- En cada ocasión simulamos dos muestras de tamaño 20 de distribuciones normales típicas y calculamos su coeficiente de correlación.
- Tenemos ahora 1000 replicas de la correlación para este caso y buscamos el valor $\hat{\eta}$ tal que 10 % de las correlaciones queden fuera del intervalo $(-\hat{\eta}, \hat{\eta})$.

Simulación

```
> cor.2 <- function(nn)
    {
    muestra.1 <- rnorm(nn)
    muestra.2 <- rnorm(nn)
    cor(muestra.1,muestra.2)
    }
> resultados <- numeric(1000)
> tam <- 1000
> for (i in 1:tam)
    {
    resultados[i] <- cor.2(20)
    }
```

Simulación

```
> hist(resultados, xlab='r',
    main='Coeficientes de correlacion')
> text(-0.5,120,paste('5% tiene r <',
    round(sort(resultados)[tam/20],3)))
> arrows(-0.45, 110, round(sort(resultados)
    [tam/20], 3),0, lwd=2, col='red')
> text(0.5, 110. paste('5% tiene r >',
    round(sort(resultados)[tam - tam/20],3)))
> arrows(0.45, 110, round(sort(resultados)
    [tam/20], 3),0, lwd=2, col='red')
```

Simulación

```
> text(0,75,paste('eta =',round(
    sort(abs(resultados))[tam-tam/10], 3)),
    col='blue', ces=1.5)
> arrows(-0.25, 70, -round(sort
    (abs(resultados))[tam-tam/10], 3),0,
    col='blue',lwd=3)
> arrows( 0.25, 70, round(sort
    (abs(resultados))[tam-tam/10], 3),0,
    col='blue',lwd=3)
```

Simulación

- De modo que un intervalo $[-0{,}375, 0{,}369]$ cubre 90 % de los coeficientes de correlación muestrales r para muestras de tamaño 20.

  El intervalo simétrico es $(-0{,}372, 0{,}372)$.

  Los resultados de cada quien serán ligeramente distintos.

Simulación

- Como otro ejemplo del uso de R para simulación vamos a considerar un paseo al azar simple.
- Consideremos una partícula que comienza en el cero y en cada instante se mueve hacia arriba o hacia abajo una unidad.
- Si definimos una sucesión de variables $\{X_n, n \geq 1\}$ de variables que toman valor 1 con probabilidad p y valor -1 con probabilidad $q = 1 - p$, la posición de la partícula en el instante n está determinada por

$$S_n = \sum_{i=1}^{n} X_i$$

Simulación

- Si los pasos sólo pueden ser unitarios, se dice que el paseo al azar es simple.
- Si las probabilidades $p = q = 1/2$, decimos que el paseo es simétrico.
- Las siguientes instrucciones simulan un paseo al azar simple simétrico.

```
> n <- 100
> (paseo <- cumsum(c(0,2*rbinom(n,1,0.5)-1)))
```

Simulación

- Las instrucciones a continuación hacen una gráfica de una trayectoria del paseo al azar simple simétrico.

```
> soporte <- 1:n
> grafico <- function(k)
{
plot(1:k, paseo[soporte <= k],
    xlim=c(1,n), ylim = c(-n/10, n/10),
    type = 'l', main = 'Paseo al Azar',
    ylab='Posición')
abline(h=0, col='red')
Sys.sleep(0.05)
}
> sapply(soporte, grafico)
```

Simulación

- Para finalizar esta sección haremos una gráfica para la distribución normal bivariada.

```
> rango <- seq(-3,3,by =0.1)
> muestra.1 <- dnorm(rango, 0, 1)
> muestra.2 <- dnorm(rango, 0, 0.7)
> mesh <- outer(muestra.1, muestra.2,
    function(x,y) x*y)
> persp(rango, rango, mesh, phi=20, theta=30,
    expand =0.8, ylab="N(0,0.7)",
    xlab="N(0,1)", zlab="FD",
    main="Distribucion Normal Bivariada")
```

Simulación

1. Ahora limpiamos el archivo de trabajo:

```
> ls()
[1] 'cor.2'   'i'   'mesh'   'muestra.1'
[5] 'muestra.2'   'nn'   'rand.1'   'rand.2'
[9] 'rango'   'resultados'   'tam'
> rm(cor.2, i, mesh, muestra.1, muestra.2,
    nn, rand.1, rand.2, rango, resultados,
    tam)
> ls()
character(0)
```

Outline

Introducción

Probabilidad

Simulación

Gráficas

Modelación

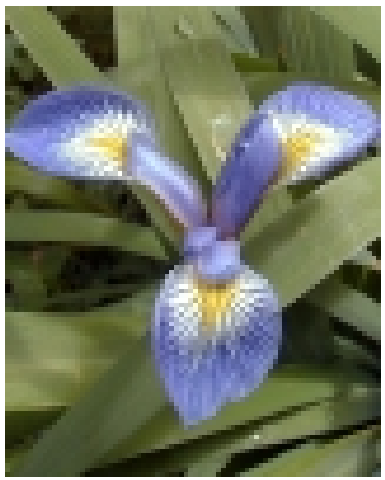
Gráficas

A continuación veremos un ejemplo típico de visualización y análisis a partir de los datos de Anderson sobre tres variedades de Iris, que fueron usados por Fisher en 1936 en la introducción de su método de análisis discriminante. El conjunto tiene datos para 50 muestras de flores de cada una las especies *setosa, versicolor* y *virginica*. Las mediciones corresponden a largo y ancho del pétalo y largo y ancho del sépalo de cada flor, en centímetros.

Gráficas

Versicolor

Virginica



Setosa

Gráficas

```
> data(iris)
> help(iris)
> str(iris)
'data.frame': 150 obs. of 5 variables:
$ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 ...
$ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 ...
$ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 ...
$ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 ...
$ Species : Factor w/ 3 levels "setosa",
    "versicolor",..: 1 1 1 1 1 1 1 ...
```

Gráficas

Para obtener alguna información básica sobre las variables usamos

```
> summary(iris)
Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
Median :5.800   Median :3.000   Median :4.350   Median :1.300
Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
       Species
 setosa     :50
 versicolor:50
 virginica  :50
```

Gráficas

Para la exploración visual de los datos usaremos el ambiente gráfico matricial que está incluido en el paquete `lattice` que cargamos al inicio. También usaremos algunas funciones del paquete `MASS`. Comenzamos por el histograma de la longitud de los pétalos

```
> histogram(~ Petal.Length,data=iris)
```

Obtenemos un histograma multimodal y nos preguntamos si las modas dependen de la especie.

```
> histogram(~ Petal.Length| Species,
    data=iris)
```

Gráficas

Podemos hacer otros gráficos con los mismos datos para hacernos una idea más clara sobre su distribución:

```
> densityplot(~ Petal.Length|Species,
    data=iris)
```

```
> bwplot(~ Petal.Length|Species,data=iris)
```

Estas gráficas sugieren que la longitud de los pétalos es una característica que depende de la especie. Más adelante investigaremos esto con mayor profundidad, pero por ahora continuaremos explorando las capacidades gráficas de `R`.

Gráficas

Comenzamos por hacer una gráfica de la longitud vs. el ancho de los pétalos.

```
> attach(iris)
> plot(Petal.Length,Petal.Width, pch=16)
```

Esta gráfica también la podemos obtener con la siguiente instrucción:

```
> plot(Petal.Width ~ Petal.Length, pch=16)
```

Gráficas

Ahora vamos a usar algunas de las opciones para mejorar el gráfico.

```
> plot(Petal.Length,Petal.Width,pch=20,
    cex=1.5, xlab='Longitud del petalo
    (cm.)', ylab='Ancho del petalo (cm.)',
    main='Datos de Anderson sobre
    plantas Iris',col=c('slateblue',
    'firebrick','darkolivegreen')
    [as.numeric(Species)])
```

Gráficas

Vemos claramente que las especies tienen tamaños distintos: *setosa* es la menor, *versicolor* es intermedia y *virginica* la mayor, pero la razón entre la longitud y el ancho parece ser constante en las tres especies. A continuación añadimos rectas verticales y horizontales en las medias y medianas y una rejilla.

```
> abline(v=mean(Petal.Length), lty=2, col='red',lwd=2)
> abline(h=mean(Petal.Width), lty=2, col='red',lwd=2)
> abline(v=median(Petal.Length),lty=2,col='blue',lwd=2)
> abline(h=median(Petal.Width),lty=2,col='blue',lwd=2)
> grid()
```

Gráficas

Añadimos ahora los centroides de media y mediana con rombos grandes de color:

```
> points(mean(Petal.Length),
    mean(Petal.Width), cex=2, pch=23,
    col='black',bg='red')
> points(median(Petal.Length),
    median(Petal.Width), cex=2, pch=23,
    col='black',bg='blue')
> title(sub='Centroides: media (rojo)
    y mediana (azul)')
```

Gráficas

También podemos añadir texto en la gráfica indicando al comienzo en qué lugar de la gráfica queremos ubicarlo.

```
> text(1,2.4, 'Tres especies de Iris', pos=4,
    col='navyblue')
> legend(1,2.4,levels(Species), pch=20,
    bty='n', col=c('slateblue', 'firebrick',
    'darkolivegreen'), cex=1.2)
```

Gráficas

Ahora incluimos en la gráfica rectas de regresión calculadas a partir de un modelo. En el primer caso se trata de mínimos cuadrados (`lm`) y el segundo un método de regresión robusta (`lqs`) incluido en el paquete `MASS`.

```
> abline(lm(Petal.Width ~ Petal.Length),
    lty = 'longdash', col='red',lwd=2)
> abline(lqs(Petal.Width ~ Petal.Length),
    lty = 2, col='blue',lwd=2)
```

Gráficas

Veamos otras gráficas de la longitud de los pétalos

```
> boxplot(Petal.Length ~ Species, horizontal
    = T, col='lightblue', boxwex=.5,
    xlab='Longitud del petalo (cm)',
    ylab='Especie', main='Box plot agrupado')
> coplot(Petal.Width ~ Petal.Length |
    Species, col=as.numeric(Species),
    pch=as.numeric(Species))
> pairs(iris[,1:4], col=as.numeric(Species),
    main='Graficas a pares')
> stars(iris[,1:4], key.loc=c(2,35),
    mar=c(2,2,10,2), main= 'Star plot de
    plantas individuales', frame=T)
```

Gráficas

También es posible interactuar con los gráficos

```
> plot(Petal.Length, Petal.Width)
> (p <- identify(Petal.Length, Petal.Width))
```

Ahora seleccionamos con el ratón los puntos que deseamos identificar en el gráfico. Al hacer click sobre el punto de interés aparece el número que lo identifica en el conjunto de datos.

Gráficas

Para terminar podemos usar el boton derecho del ratón y al hacerlo aparecen los números de los puntos que seleccionamos en la cónsola.

```
 [1]  44 135 145
 > iris[p,]
     Sepal.Length  Sepal.Width  Petal.Length  Petal.Width
44            5.0          3.5           1.6          0.6
135           6.1          2.6           5.6          1.4
145           6.7          3.3           5.7          2.5
      Species
      setosa
      virginica
      virginica
```

Gráficas

Otras gráficas bivariadas.

```
> xyplot(Petal.Width ~ Petal.Length,
    data=iris, groups=Species,
    auto.key=T)
> xyplot(Petal.Width ~ Petal.Length|Species,
    data=iris, groups=Species, auto.key=T)
```

Gráficas

Graficas Trivariadas.

```
> pl1 <- cloud(Sepal.Length ~ Petal.Length *
    Petal.Width, group=Species, data=iris,
    pch=20, main='Datos sobre plantas Iris',
    screen=list(z=30, x=-60))
> data(volcano)
> pl2 <- wireframe(volcano, shade=TRUE,
    aspect=c(61/87,0.4), ligth.source =
    c(10,0,10), zoom=1.1, box=F,
    scales=list(draw=F), xlab="",
    ylab="", zlab="", main='Volcan
    Maunga Whau, Auckland')
```

Gráficas

```
> pl3 <- levelplot(volcano, col.regions=
    gray(0:16/16), main='Grafica de Nivel')
> pl4 <- contourplot(volcano, at=
    seq(floor(min(volcano)/10)*10,
    ceiling(max(volcano)/10)*10, by=10),
    main='Curvas de Nivel' sub='Intervalo
    entre curvas 10 m.', region=T,
    col.regions=terrain.colors(100))
```

Gráficas

```
> print(pl1, split=c(1,1,2,2), more=T)
> print(pl2, split=c(2,1,2,2), more=T)
> print(pl3, split=c(1,2,2,2), more=T)
> print(pl4, split=c(2,2,2,2), more=F)
> rm(pl1, pl2, pl3, pl4)
```

Outline

Introducción

Probabilidad

Simulación

Gráficas

Modelación

Modelación

Regresemos al análisis de la longitud de los pétalos. Habíamos visto a través de los gráficos que esta variable parece depender de la especie. Veamos esto con mayor profundidad haciendo algunos modelos. Comenzamos por hacer un modelo lineal.

```
> model <- lm(formula = Petal.Length ~
    Species, data = iris)
> summary(model)
```

Modelación

```
Call: lm(formula = Petal.Length ~ Species, data = iris)
Residuals:
Min 1Q Median 3Q Max
-1.260 -0.258 0.038 0.240 1.348
Coefficients:

                  Estimate   Std. Error   t value   Pr(>|t|)
     (Intercept)    1.46200      0.06086     24.02     <2e-16   ***
Speciesversicolor   2.79800      0.08607     32.51     <2e-16   ***
 Speciesvirginica   4.09000      0.08607     47.52     <2e-16   ***
--
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 0.4303 on 147 degrees of freedom
Multiple R-Squared: 0.9414, Adjusted R-squared: 0.9406

F-statistic: 1180 on 2 and 147 DF, p-value: < 2.2e-16
```

Modelación

```
> anova(model)
Analysis of Variance Table
Response: Petal.Length
             Df  Sum Sq Mean Sq F value    Pr(>F)
Species       2  437.10  218.55  1180.2 < 2.2e-16 ***
Residuals   147   27.22    0.19
--
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

De acuerdo a estos resultados las tres especies pueden separarse.

Modelación

Veamos ahora si hay una relación entre la longitud de los pétalos y la de los sépalos, sin importar la especie.

```
> par(mfrow=c(1,1))
> plot(Sepal.Length ~ Petal.Length, data =
    iris, col=as.numeric(Species),pch=20)
```

Probamos primero un modelo que no separa las especies.

```
> model <- lm(Sepal.Length ~ Petal.Length,
    data=iris)
> summary(model)
```

Modelación

```
> model <- lm(Sepal.Length ~ Petal.Length, data=iris)
> summary(model)
Call: lm(formula = Sepal.Length ~ Petal.Length, data = iris)
Residuals:
Min 1Q Median 3Q Max
-1.24675 -0.29657 -0.01515 0.27676 1.00269
Coefficients:

             Estimate   Std. Error   t value   Pr(>|t|)
 (Intercept)   4.30660      0.07839     54.94     <2e-16   ***
Petal.Length   0.40892      0.01889     21.65     <2e-16   ***
--
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 0.4071 on 148 degrees of freedom
Multiple R-Squared: 0.76, Adjusted R-squared: 0.7583

F-statistic: 468.6 on 1 and 148 DF, p-value: < 2.2e-16
```

Modelación

La longitud de los pétalos puede explicar 76 % de la variabilidad de la longitud de los sépalos. Hacemos una gráfica de los residuos para ver si el ajuste es bueno.

```
plot(residuals(model) ~ fitted(model),
    col=as.numeric(iris$Species), pch=20)
    abline(h=0)
```

La primera especie tiene más residuos altos que bajos y las otras dos parecen tener un patrón, que sugiere que las especies podrían tener distintas relaciones entre las dos variables. Para ver esto ajustamos un modelo mixto.

```
> model <- lm(Sepal.Length ~ Petal.Length *
    Species, data=iris)
> summary(model)
```

Modelación

```
Call: lm(formula = Sepal.Length ~ Petal.Length * Species,
    data = iris)
Residuals:
Min 1Q Median 3Q Max
-0.73479 -0.22785 -0.03132 0.24375 0.93608
Coefficients:
```

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	4.2132	0.4074	10.341	< 2e-16	***
Petal.Length	0.5423	0.2768	1.959	0.05200	.
Speciesversicolor	-1.8056	0.5984	-3.017	0.00302	**
Speciesvirginica	-3.1535	0.6341	-4.973	1.85e-06	***
Petal.Length:Speciesversicolor	0.2860	0.2951	0.969	0.33405	
Petal.Length:Speciesvirginica	0.4534	0.2901	1.563	0.12029	

```
--
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 0.3365 on 144 degrees of freedom
Multiple R-Squared: 0.8405, Adjusted R-squared: 0.8349

F-statistic: 151.7 on 5 and 144 DF, p-value: < 2.2e-16
```

Modelación

El ajuste es mejor. Veamos ahora si los términos de interacción son necesarios.

```
> model.2 <- lm(Sepal.Length ~ Petal.Length
    | Species, data=iris)
> anova(model,model.2)
Analysis of Variance Table
Model 1: Sepal.Length ~ Petal.Length * Species
Model 2: Sepal.Length ~ Petal.Length/Species
  Res.Df     RSS  Df  Sum of Sq      F    Pr(>F)
1    144 16.3007
2    146 19.1998  -2    -2.8991 12.805 7.611e-06 ***
--
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Componentes Principales

Ahora hacemos un análisis en componentes principales para ver cuánta información redundante hay en las cuatro variables.

```
> pca <- prcomp(iris[,1:4],scale=T)
> summary(pca)
Importance of components:
```

	PC1	PC2	PC3	PC4
Standard deviation	1.71	0.956	0.3831	0.14393
Proportion of Variance	0.73	0.229	0.0367	0.00518
Cumulative Proportion	0.73	0.958	0.9948	1.00000

```
> screeplot(pca)
```

Componentes Principales

Las dos primeras componentes principales incluyen 96 % de la varianza total. Para ver la relación de las componentes principales con las variables usamos la siguiente instrucción.

```
> pca$rotation
```

	PC1	PC2	PC3	PC4
Sepal.Length	0.5210659	-0.37741762	0.7195664	0.2612863
Sepal.Width	-0.2693474	-0.92329566	-0.2443818	-0.1235096
Petal.Length	0.5804131	-0.02449161	-0.1421264	-0.8014492
Petal.Width	0.5648565	-0.06694199	-0.6342727	0.5235971

```
> biplot(pca)
```

Modelación

A continuación tratamos de modelar la distribución de la longitud de los pétalos para una de las especies. Intentamos con una distribución normal.

```
> petal.v <- iris$Petal.Length[iris$Species
    == "versicolor"]
> par(mfrow=c(1,2))
> hist(petal.v, breaks=12, xlab="Longitud de
    los petalos", main=Ïris versicolor")
> qqnorm(petal.v)
> qqline(petal.v)
> shapiro.test(petal.v)
        Shapiro-Wilk normality test
    data: petal.v
    W = 0.966, p-value = 0.1585
```

Modelación

De acuerdo a los gráficos el ajuste no es muy bueno pero la prueba de Shapiro-Wilks no rechaza la hipótesis nula de normalidad, así que estimamos los parámetros de la población y hacemos un histograma con la densidad normal superpuesta.

```
> par(mfrow=c(1,1))
> mu <- mean(petal.v)
> sigma <- sd(petal.v)
> puntos <- seq(3,5.5,length=100)
> puntos.normal <- dnorm(puntos,
    mean=mu,sd=sigma)
> hist(petal.v, breaks=12, xlab="Longitud de
    los petalos", main=Ïris versicolor",
    probability=T)
> lines(puntos, puntos.normal)
```

Modelación

Ahora simulamos una nueva muestra de esta distribución del mismo tamaño que la original.

```
  > petal.v.2 <-
sort(rnorm(length(petal.v),mu,sigma))
  > summary(petal.v.2)
   Min.  1st Qu.  Median   Mean  3rd Qu.   Max.
 3.068    3.942   4.140  4.220    4.499  5.321
```

Modelación

A continuación hacemos la predicción de la longitud de los sépalos de acuerdo al modelo ajustado previamente. Mostramos también un intervalo de 90 % así como los puntos de la muestra original.

```
> sepals.2 <- predict(model, data.frame(Petal.Length =
    petal.v.2, Species = "versicolor"),
    interval = 'prediction', level=0.9)
> plot(petal.v.2,sepals.2[,'fit'], type= 'l',
    col='blue', lwd=1.5, xlab='Longitud de
    petalos simulada', ylab='Prediccion de longitud
    de sepalos', main='Iris versicolor')
> lines(petal.v.2,sepals.2[,'lwr'], col='red', lty=2)
> lines(petal.v.2,sepals.2[,'upr'], col='red', lty=2)
> points(petal.v, iris$Sepal.Length[iris$Species ==
    'versicolor'],pch=20)
> rug(petal.v.2)
> rug(sepals.2[,1],side=2)
```

Fin

Para concluir la sesión limpiamos el archivo de trabajo:

```
> ls()

[1]  'iris'     'model'           'model.2'  'mu'
[5]  'p'        'pca'             'petal.v'  'petal.v.2'
[9]  'puntos'   'puntos.normal'   'sigma'

> rm(iris, model, model.2, mu, p, pca,
    petal.v, petal.v.2, puntos,
    puntos.normal, sigma)
> ls()
[1] 'iris'
```