Análisis de datos y estadística descriptiva con R y R-Commander

Daniel Liviano Solís
Maria Pujol Jover

PID_00208273

UOC
Universitat Oberta de Catalunya

Índice

Introducción 5

Objetivos 6

1. Estadística descriptiva con R-Commander 7
1.1. Iniciar sesión con R-Commander 7
1.2. Introducción de datos 7
1.3. Importación de datos 9
1.4. Análisis descriptivo 10
1.5. Análisis gráfico 14
1.5.1. Histograma 14
1.5.2. Diagrama de barras 15
1.5.3. Diagrama de caja 16
1.5.4. Diagrama de dispersión 17
1.6. Transformación de variables 19

2. Análisis del mercado de trabajo en España 21
2.1. Importación y manejo de datos 21
2.2. Estadística descriptiva 24
2.3. Representación gráfica 25
2.3.1. Diagrama de dispersión 25
2.3.2. Histograma y función de densidad 27
2.3.3. Diagrama de caja 29
2.3.4. Gráficos compuestos 30

3. Análisis demográfico en Cataluña 32
3.1. Manejo de bases de datos 32
3.2. Creación y análisis de variables 34
3.3. Creación y análisis de factores 35
3.4. Representación gráfica 38
3.4.1. Gráficos con componente factorial 38
3.4.2. La librería *Lattice* 39

Bibliografía 44

Introducción

Este módulo tiene dos grandes objetivos. Por una parte, el primer capítulo pretende introducir el análisis estadístico descriptivo usando R-Commander, lo cual incluye la introducción e importación de datos, la creación y transformación de variables, el cálculo de estadísticos básicos (a nivel univariante y multivariante) y la realización de gráficos. Estos contenidos se corresponden, aproximadamente, a las asignaturas introductorias de estadística cursadas en la UOC.

Por otra parte, los capítulos segundo y tercero profundizan en las posibilidades que ofrece R para realizar análisis estadísticos descriptivos sin usar R-Commander, es decir, solo mediante código. Evidentemente, es un análisis más complejo pero ofrece muchas más posibilidades, tanto en cuanto al cálculo de estadísticos y la extracción de información estadística en general, como en el terreno del análisis gráfico.

Objetivos

1. Ser capaz de introducir datos y variables directamente.

2. Conocer los principales cálculos de estadística descriptiva, y saber implementarlos con R-Commander.

3. Saber elegir en cada caso el tipo de gráfico necesario, según el análisis que se lleve a cabo.

4. Poder exportar los resultados obtenidos con R y con R-Commander en diferentes formatos.

1. Estadística descriptiva con R-Commander

1.1. Iniciar sesión con R-Commander

Empecemos recordando que para iniciar R-Commander hemos iniciar una sesión de R, y a continuación introducir en la consola la siguiente instrucción:

```
> library(Rcmdr)
```

Es importante que el programa R-Commander ya esté instalado en el ordenador[1]. Si todo es correcto, obtendremos una ventana con la interfaz R-Commander, tal y como se muestra en la figura 1:

Figura 1: Interfaz de R-Commander

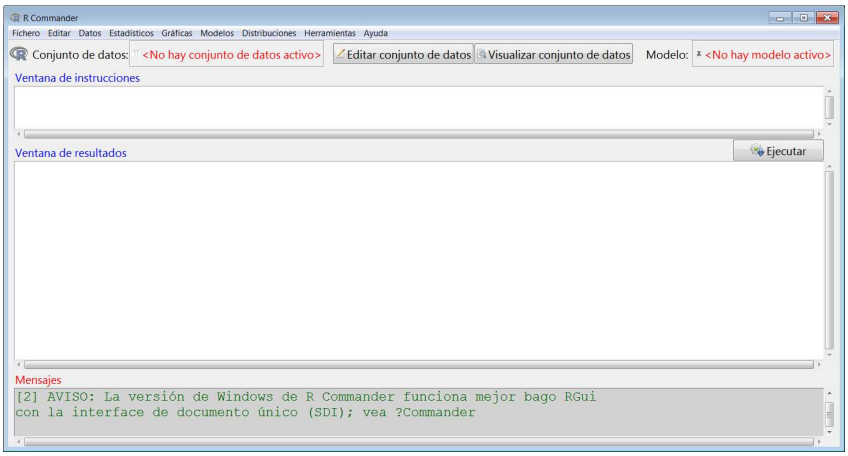


Figura 1

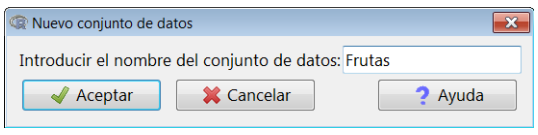
Cuando aparece la ventana de R-Commander, la sesión de R continúa abierta, de manera que ambas ventanas se mantienen abiertas y operativas simultáneamente.

1.2. Introducción de datos

Una de las opciones que tiene el usuario es la de introducir manualmente los datos del análisis. En este caso, la ruta que deberemos seguir será la siguiente:

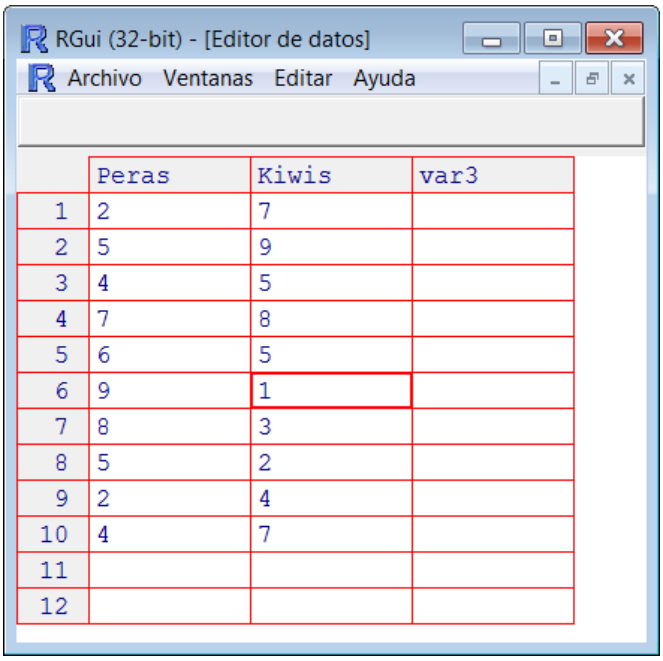
Datos / Nuevo conjunto de datos

Entonces aparecerá la siguiente ventana, en la cual tendremos que introducir el nombre que le queramos dar a nuestro conjunto de datos (por ejemplo, *Frutas*):



[1] El primer módulo explica detalladamente cómo se realiza la instalación de R-Commander.

Una vez introducido el nombre del conjunto de datos, pulsamos en *Aceptar*. Seguidamente, aparecerá una hoja de cálculo con celdas vacías donde deberemos introducir nuestras variables en columnas, especificando también el nombre de cada variable en la primera fila. En nuestro ejemplo, crearemos dos variables ficticias: *Peras* y *Kiwis*. Para introducir los nombres en cada columna, pulsaremos sobre los encabezados *var1, var2,...* e introduciremos el nombre deseado en cada caso, pulsando después en la opción *numeric* o *character*, en función de si se trata de una variable numérica o de texto. Los valores se introducen con el teclado:

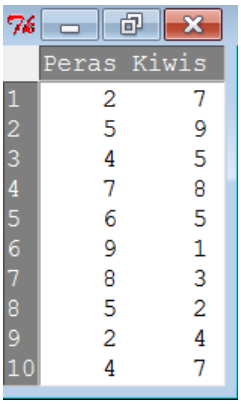


	Peras	Kiwis	var3
1	2	7	
2	5	9	
3	4	5	
4	7	8	
5	6	5	
6	9	1	
7	8	3	
8	5	2	
9	2	4	
10	4	7	
11			
12			

Introducir y guardar datos

Una vez hemos introducido las variables en columnas, y le hemos dado un nombre a la primera celda de cada columna, simplemente cerrando esta ventana se guardan los datos. Si queremos volver a acceder a este editor de datos, simplemente hemos de acceder a la opción *Editar conjunto de datos*.

Cuando tengamos creado nuestro conjunto de datos, simplemente cerramos la ventana y las variables con sus valores quedarán guardadas. En el momento en el que queramos ver los datos en memoria, lo haremos mediante la opción *Visualizar conjunto de datos*. En nuestro caso obtendríamos la siguiente ventana:



	Peras	Kiwis
1	2	7
2	5	9
3	4	5
4	7	8
5	6	5
6	9	1
7	8	3
8	5	2
9	2	4
10	4	7

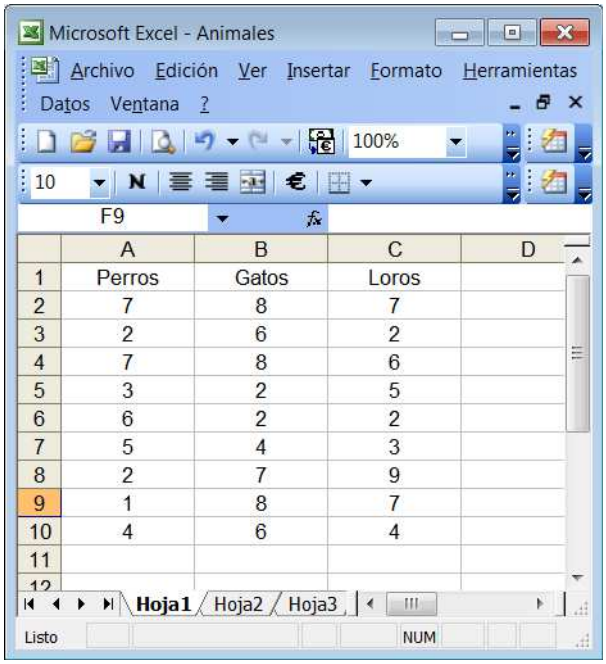
Visualizar datos

Esta ventana no se puede editar ni modificar, ya que simplemente muestra las variables que componen el conjunto de datos.

De igual manera, podemos introducir y/o modificar datos mediante la opción *Editar conjunto de datos*.

1.3. Importación de datos

Otra opción que nos ofrece R-Commander es la de importar datos de un archivo externo, es decir, creado previamente con cualquier otro programa. Hay varias maneras de hacer esta operación. En este manual, nos limitamos a explicar la manera más fácil y directa. Supongamos que los datos originales están en formato Excel (con extensión *xls*), y que las variables están dispuestas en columnas, con la primera fila reservada para el nombre de cada variable. En este caso, tenemos 9 observaciones y tres variables: *Perros*, *Gatos* y *Loros*:



	A	B	C	D
1	Perros	Gatos	Loros	
2	7	8	7	
3	2	6	2	
4	7	8	6	
5	3	2	5	
6	6	2	2	
7	5	4	3	
8	2	7	9	
9	1	8	7	
10	4	6	4	
11				

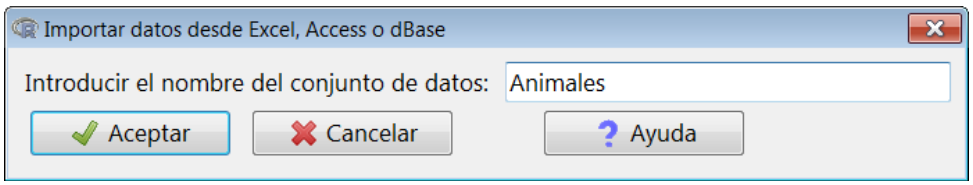
Importar datos con decimales

Seguramente tendremos configurado el programa Excel para que use las comas como separador de decimales. Esto no supone ningún problema: R-Commander convertirá automáticamente estas comas en puntos, ya que, como se ha comentado, R usa el punto como separador decimal.

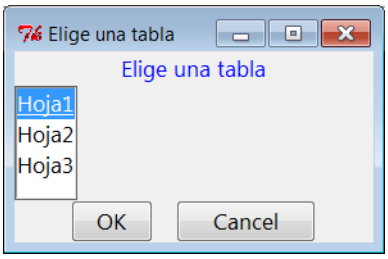
Para cargar desde R-Commander estos datos, deberemos seguir la siguiente ruta:

Datos / Importar datos / desde conjunto de datos Excel, Access o dBase...

Igual que antes, aparecerá una ventana que nos preguntará el nombre que le queremos dar a nuestro conjunto de datos. En este ejemplo, lo llamaremos (*Animales*):



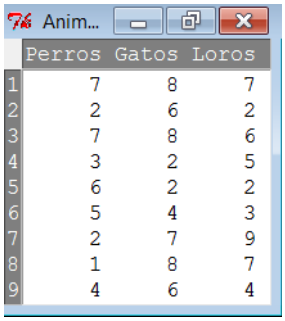
Al pulsar en *Aceptar*, aparecerá la siguiente pantalla, donde hemos de especificar la ruta de acceso al archivo Excel con los datos originales, especificando además en qué hoja están los datos que queremos utilizar:



¡Cuidado con las hojas en documentos de Excel!

Muchas veces nos encontraremos con archivos de Excel con más de una hoja con datos. Es fundamental tenerlas identificadas para hacer este paso correctamente, y no importar datos que no corresponden.

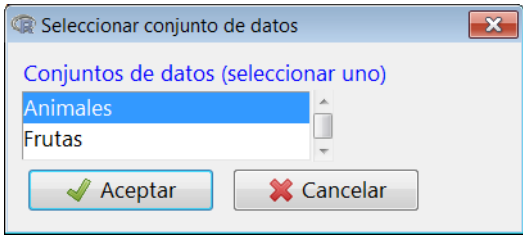
Podemos visualizar ahora, como hemos hecho antes, el conjunto de datos que acabamos de importar para comprobar que se han cargado correctamente:



	Perros	Gatos	Loros
1	7	8	7
2	2	6	2
3	7	8	6
4	3	2	5
5	6	2	2
6	5	4	3
7	2	7	9
8	1	8	7
9	4	6	4

1.4. Análisis descriptivo

A menudo sucederá que tengamos diferentes conjuntos de datos cargados pero R únicamente nos permite mantener uno activo, que es con el que trabajaremos. Por tanto, el primer paso a la hora de hacer un análisis descriptivo es el de activar un conjunto de datos de entre los que hayamos importado o introducido. Esto lo haremos pulsando sobre *Conjunto de datos* y eligiendo uno de ellos. En nuestro ejemplo, trabajaremos con el conjunto de datos *Animales*:



Un primer grupo de estadísticos básicos de las variables incluidas en un conjunto de datos es el que nos ofrece el siguiente menú desplegable:

Estadísticos / Resúmenes / Conjunto de datos activo

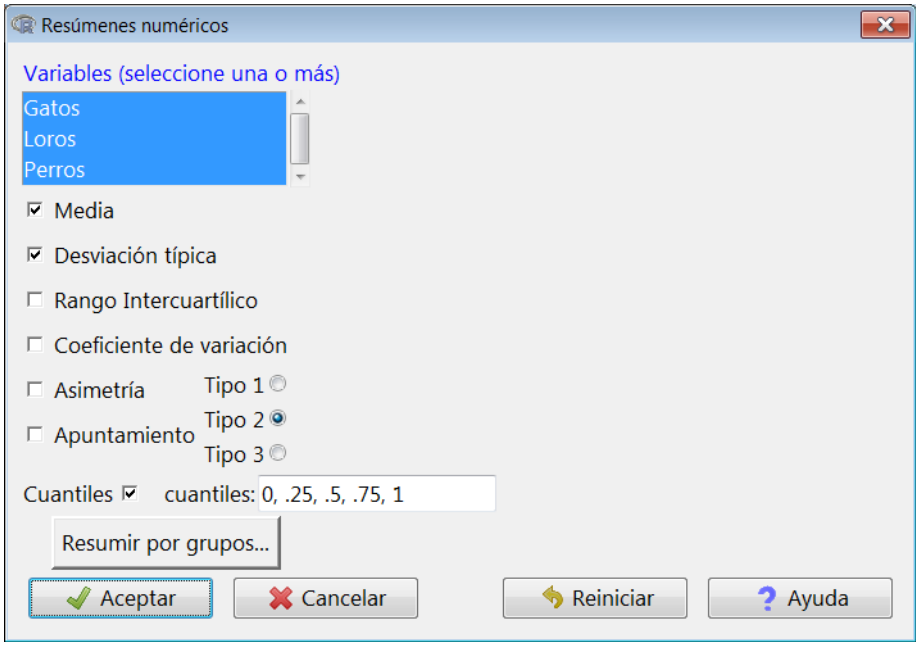
El resultado obtenido en la consola se muestra a continuación. Vemos que incluye los valores mínimo y máximo de cada variable, la media aritmética y los tres cuartiles (el segundo es la mediana):

```
> summary(Animales)
     Perros           Gatos           Loros
 Min.   :1.000   Min.   :2.000   Min.   :2
 1st Qu.:2.000   1st Qu.:4.000   1st Qu.:3
 Median :4.000   Median :6.000   Median :5
 Mean   :4.111   Mean   :5.667   Mean   :5
 3rd Qu.:6.000   3rd Qu.:8.000   3rd Qu.:7
 Max.   :7.000   Max.   :8.000   Max.   :9
```

Muchas veces no tendremos suficiente con los estadísticos básicos y desearemos obtener medidas adicionales, como la asimetría, la curtosis, el coeficiente de variación, la desviación típica o algunos cuantiles. Para ello existe una opción en la que se puede elegir entre un conjunto de estadísticos. Para acceder a esta opción, la ruta que deberemos seguir será la siguiente:

Estadísticos / Resúmenes / Resúmenes numéricos

Obtendremos el siguiente menú, en el cual seleccionaremos, de las variables que nos interesen, los estadísticos que queramos obtener.



Cálculo de cuantiles

Para calcular diferentes cuantiles, hemos de introducir específicamente cuáles deseamos calcular, teniendo en cuenta que el separador de decimales es el punto y que los diferentes cuantiles se separan con comas. En este ejemplo, estamos calculando el mínimo (0), los tres cuartiles (0, 25, 0, 5 y 0, 75) y el máximo (1).

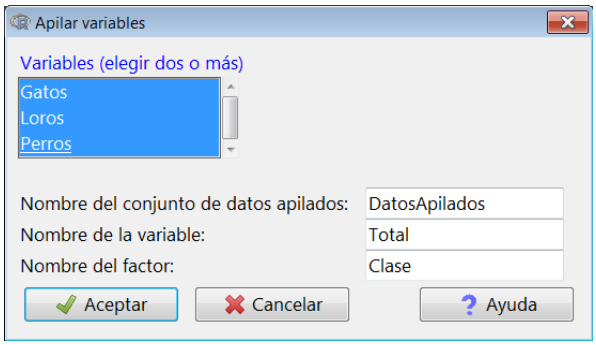
El resultado obtenido en la ventana de resultados se muestra a continuación:

```
> numSummary(Animales[,c("Gatos", "Loros", "Perros")],
+ statistics=c("mean", "sd", "quantiles"),
+ quantiles=c(0,.25,.5,.75,1))
           mean       sd 0% 25% 50% 75% 100% n
Gatos  5.666667 2.449490  2   4   6   8    8 9
Loros  5.000000 2.449490  2   3   5   7    9 9
Perros 4.111111 2.260777  1   2   4   6    7 9
```

Cabe mencionar que en ocasiones no tenemos dispuestos los datos de la manera adecuada para obtener los análisis deseados. Supongamos que deseamos agrupar las tres variables en una sola, haciendo que tenga $9 \times 3 = 27$ observaciones, y que queremos crear una variable cualitativa asociada (o factor) que indique, para cada una de las 27 observaciones, qué clase de animal es. En R-Commander las instrucciones más sencillas que hemos de seguir para realizar esto son las siguientes:

Datos / Conjunto de datos activo / Apilar variables del conjunto de datos activo

A continuación aparecerá el siguiente cuadro de diálogo, que nos permitirá dar nombre al nuevo conjunto de datos, a la variable agrupada y al factor. Al nuevo conjunto de datos lo llamaremos *DatosApliados*, la nueva variable con 27 observaciones se llamará *Total*, y el factor, que es una variable cualitativa con caracteres, se llamará *Clase*, y especificará para cada observación qué animal es.



Si visualizamos el nuevo conjunto de datos mediante la opción *Visualizar conjunto de datos*, vemos cómo se compone de dos variables: una numérica (*Total*) y el factor asociado (*Clase*):

	Total	Clase
1	8	Gatos
2	6	Gatos
3	8	Gatos
4	2	Gatos
5	2	Gatos
6	4	Gatos
7	7	Gatos
8	8	Gatos
9	6	Gatos
10	7	Loros
11	2	Loros
12	6	Loros
13	5	Loros
14	2	Loros
15	3	Loros
16	9	Loros
17	7	Loros
18	4	Loros
19	7	Perros
20	2	Perros
21	7	Perros
22	3	Perros
23	6	Perros
24	5	Perros
25	2	Perros
26	1	Perros
27	4	Perros

Crear nuevos conjuntos de datos

Hay que tener en cuenta que, al hacer esta operación, estamos creando un nuevo conjunto de datos diferente del anterior. En este caso, el conjunto *DatosApilados* tendrá solo dos variables: una con las observaciones (*Total*) y otra con el tipo de animal que es (*Clase*). **Hay que tener siempre presente cuál de los dos conjuntos de datos está activo** ya que los cálculos que efectuemos se aplicarán solo sobre el conjunto de datos que esté activo.

Veamos las estadísticas básicas de estas nuevas variables, y cómo hay 9 animales de cada clase. Recordemos la ruta que debemos seguir para obtener este resultado:

Estadísticos / Resúmenes / Conjunto de datos activo

```
> summary(DatosApilados)
     Total          Clase
 Min.   :1.000   Gatos :9
 1st Qu.:2.500   Loros :9
 Median :5.000   Perros:9
 Mean   :4.926
 3rd Qu.:7.000
 Max.   :9.000
```

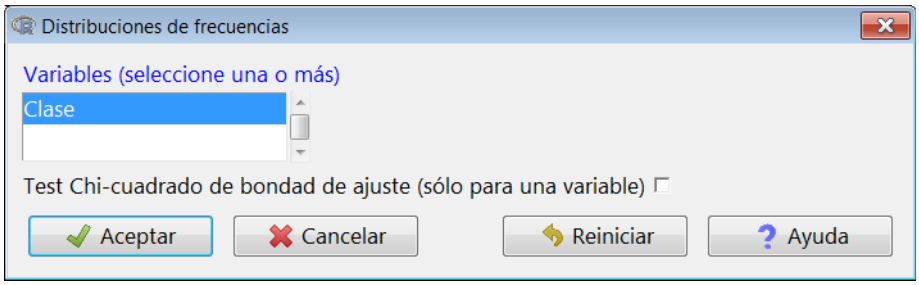
Resumen de factores

Fijémonos en que para la variable *Clase*, que es un factor con caracteres, esta función nos calcula el número de observaciones, en este caso 9 observaciones de cada tipo de animal.

Si lo que deseamos es una tabla de frecuencias del factor (variable cualitativa) que acabamos de crear, utilizaremos la siguiente ruta:

Estadísticos / Resúmenes / Distribución de frecuencias

Como podemos observar, R-Commander nos ofrece la opción de marcar la posibilidad de la realización del Test Chi-cuadrado de bondad del ajuste, que se estudiará más adelante:



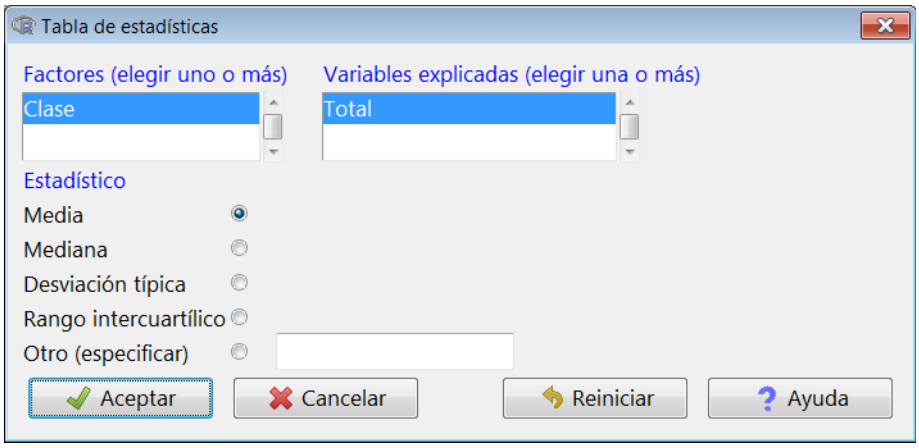
Esto es lo que obtenemos en la ventana de resultados. Observamos que contiene la frecuencia absoluta (9 animales de cada clase) y relativa (un 33 % de cada clase):

```
> .Table <- table(DatosApilados$Clase)

> .Table  # counts for Clase

 Gatos  Loros Perros
     9      9      9

> round(100*.Table/sum(.Table), 2)  # percentages for Clase

 Gatos  Loros Perros
 33.33  33.33  33.33

> remove(.Table)
```

También es posible obtener estadísticos específicos de la variable apilada según el factor. Esto se realiza siguiendo la siguiente ruta:

Estadísticos / Resúmenes / Tablas de estadísticas

Con ello obtenemos un menú en el que hay que elegir la variable de interés, el factor (en este caso solo está la clase de animal), y el estadístico que nos interesa. Por ejemplo, calcularemos la media aritmética:



Diferentes factores para una variable

En este menú hemos de elegir la variable que analizar y los factores. Es posible que tengamos más de un factor asociado a una variable, con lo que es importante no confundirse y elegir el que nos interesa.

Vemos que el resultado nos muestra cómo la media aritmética es superior para las observaciones que corresponden a los gatos:

```
> tapply(DatosApilados$Total, list(Clase=DatosApilados$Clase), mean,
+ na.rm=TRUE)
Clase
   Gatos    Loros   Perros
5.666667 5.000000 4.111111
```

1.5. Análisis gráfico

Antes de nada, hay que dejar claro que el menú desplegable de R-Commander solo nos ofrece una minúscula parte de todo el potencial de R en lo que respecta a análisis gráfico. Aun así, R-Commander es más que suficiente para cubrir el análisis que se realiza en los cursos de estadística en los grados de la UOC.

1.5.1. Histograma

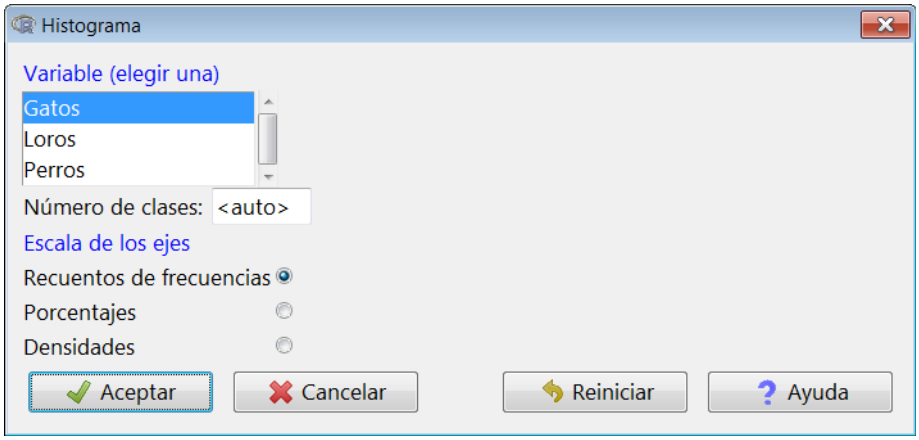
Empezaremos el análisis con el histograma, que es una representación gráfica de una variable continua en forma de barras verticales, donde la superficie de cada barra es proporcional a la frecuencia de los valores representados. En nuestro ejemplo, calcularemos el histograma de la variable *Gatos*. El primer paso será activar el conjunto de datos *Animales*, y después accederemos a la siguiente ruta del menú desplegable:

Gráficas / Histograma

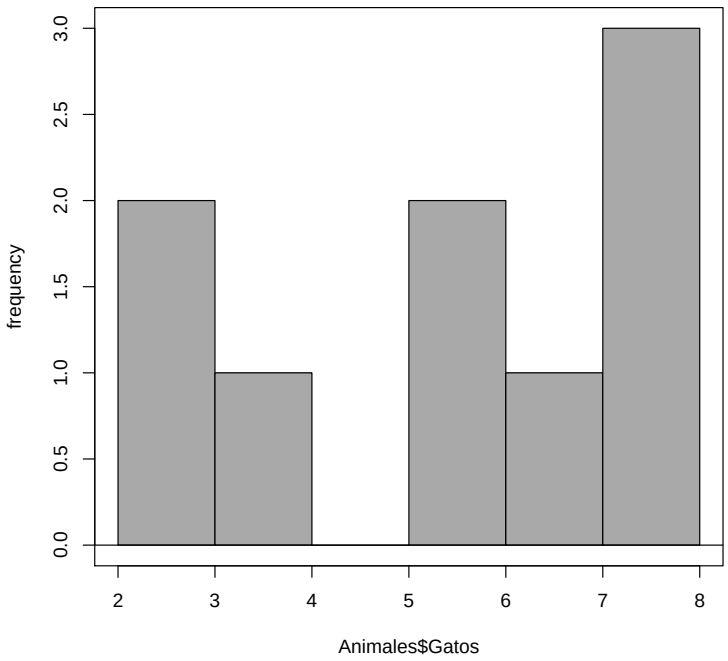
Histograma y diagrama de barras

Aunque se parecen, estos dos tipos de gráfico no son idénticos. El histograma se emplea para representar **datos cuantitativos continuos**, mientras que el **diagrama de barras** se usa para representar gráficamente datos cuantitativos discretos o datos cualitativos.

Al hacerlo aparecerá un menú en el que deberemos elegir la variable que representar y el tipo de histograma, esto es, si queremos frecuencias absolutas (las cuales sumarían 9), porcentajes o densidades.



El resultado, con las frecuencias absolutas, se muestra a continuación. Es importante tener en cuenta que el gráfico aparece en la consola de R y no en la ventana de R-Commander, con lo que deberemos acudir a la sesión inicial de R para visualizarlo.



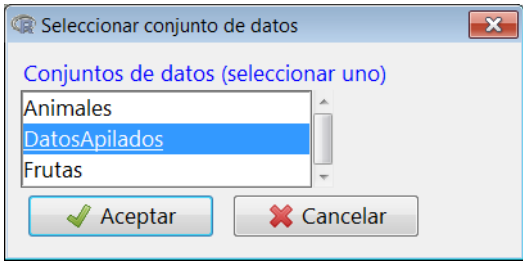
Guardar y exportar gráficos

En la consola original de R, una vez seleccionada la ventana del gráfico, si vamos al menú *Archivo*, se desplegará un menú con opciones para guardar el gráfico en un documento. Veremos que podemos elegir entre varios formatos (EPS, JPG, PDF, etc.).

1.5.2. Diagrama de barras

Como se ha comentado anteriormente, el diagrama de barras se usa con variables discretas o cualitativas, como son los factores. En nuestro ejemplo, este gráfico nos mostrará cuántas observaciones están incluidas en cada una de las categorías del factor.

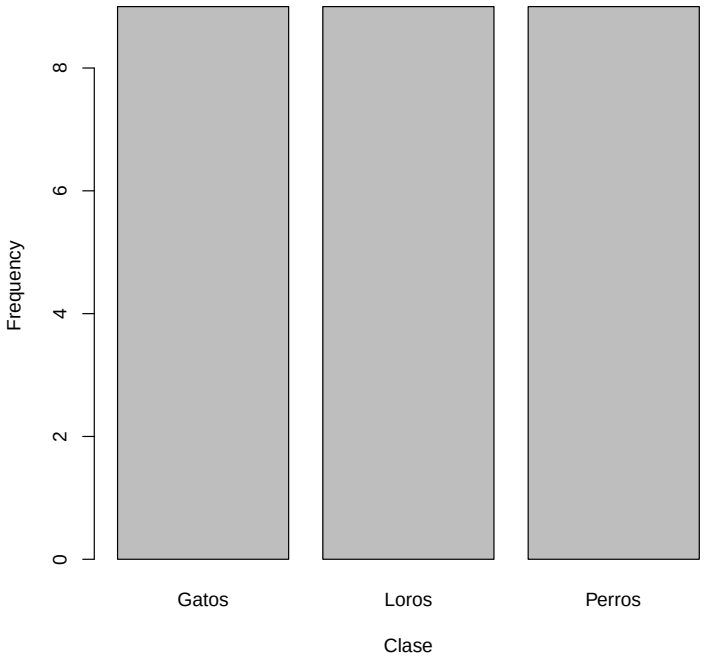
Antes de proceder, es importante cambiar el conjunto de datos activo, esto es, activar el conjunto *DatosApilados*:



Ahora se puede proceder accediendo a la siguiente ruta del menú desplegable:

Gráficas / Gráfica de barras

Como podemos ver, existen 9 observaciones para cada una de las categorías del factor *Clase*.



Opciones del menú no disponibles

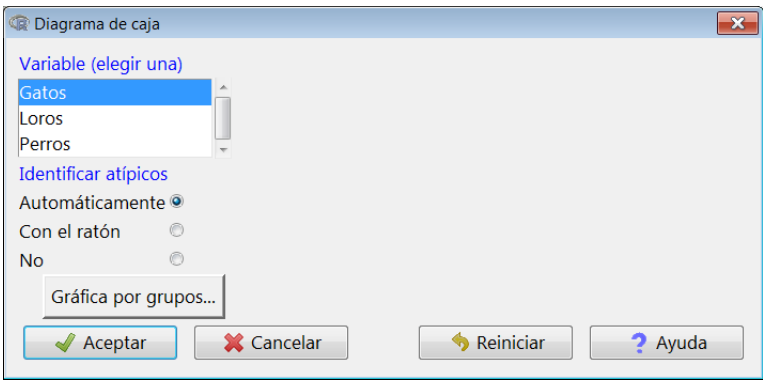
Si intentáramos calcular un diagrama de barras estando activo el conjunto de datos *Animales*, la opción del menú *Gráficas / Gráfica de barras* estaría desactivada, ya que no hay en este conjunto ningún factor para ser representado.

1.5.3. Diagrama de caja

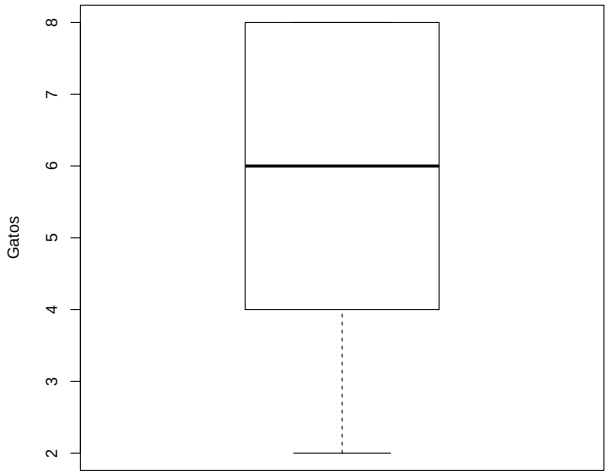
El diagrama de caja proporciona información sobre los valores mínimo y máximo, los cuartiles y posibles valores atípicos, además de la simetría de la distribución de los datos. Calcularemos este gráfico de la variable *Gatos*, con lo que hemos de volver a cambiar el conjunto de datos activo a *Animales*. Una vez hecho esto, la ruta que debemos seguir es la siguiente:

Gráficas / Diagrama de caja

Nos aparecerá un cuadro de diálogo en el que hemos de elegir la variable para la que queremos calcular el diagrama de caja:



En el gráfico resultante podemos observar los tres cuartiles, más el máximo y el mínimo, y cómo la distribución muestra una asimetría negativa.

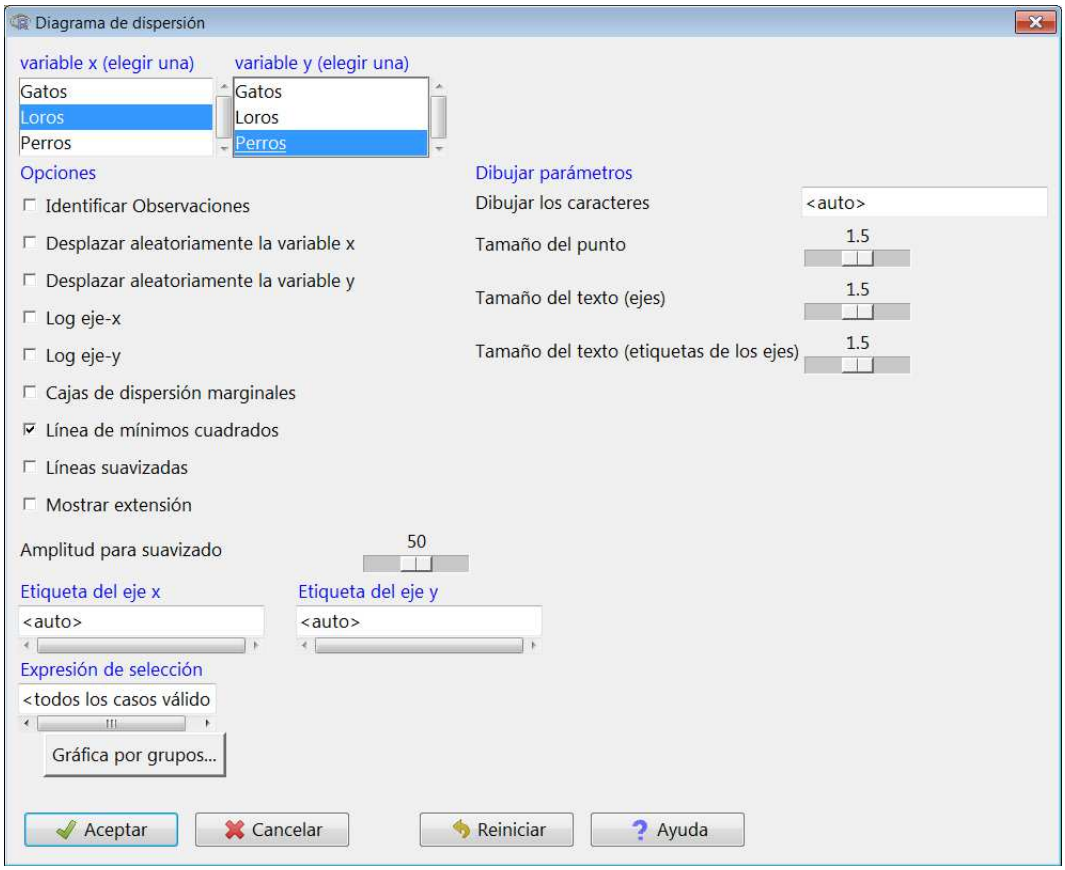


1.5.4. Diagrama de dispersión

Por último, veremos cómo se calcula un diagrama de dispersión para dos variables, consistente en coordenadas cartesianas donde cada observación es un punto diferente, y cuya posición la determinan dos valores: uno en el eje horizontal y otro en el eje vertical, es decir, uno para cada variable. La disposición de los puntos revelará si existe algún tipo de correlación o no entre ambas variables. Por ejemplo, calcularemos dicho diagrama para las variables *Perros* y *Loros*. Accediendo a la siguiente ruta:

Gráficas / Diagrama de dispersión

Obtendremos un completo cuadro de diálogo con múltiples opciones gráficas, como vemos a continuación. En nuestro caso, solo activaremos la opción *Línea de mínimos cuadrados*, para visualizar la recta que mejor se ajusta a los puntos.



Como podemos observar a continuación, el resultado muestra que no existe una clara correlación entre ambas variables, ya que la pendiente es bastante plana y la mayoría de los puntos están muy alejados de la recta estimada.

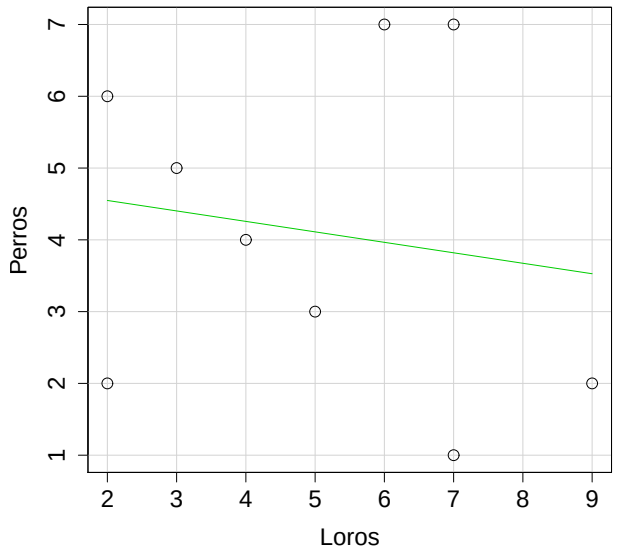


Diagrama de dispersión

Este tipo de gráfico es muy útil a la hora de realizar una exploración previa de los datos, es decir, un análisis descriptivo, ya que proporciona información visual sobre cómo se comportan las variables y cómo están relacionadas entre ellas.

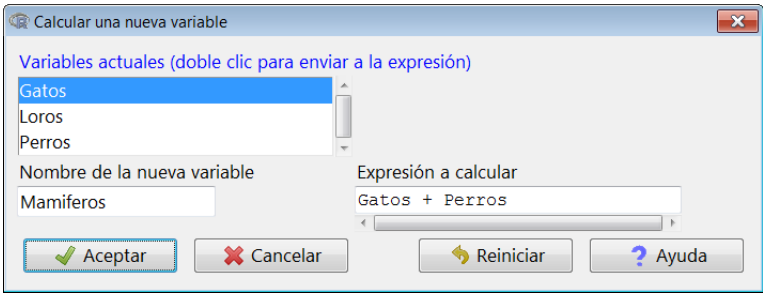
1.6. Transformación de variables

A la hora de efectuar un análisis cuantitativo, es fundamental tener la opción de poder realizar transformaciones de las variables que estamos estudiando. Esto incluye sumar, restar, elevar a una potencia, raíces cuadradas, logaritmos y un largo etcétera. En los dos primeros módulos hemos repasado profundamente este tipo de operaciones, que también se pueden implementar mediante R-Commander. En la opción *Datos* de la barra de menús se pueden efectuar todas estas transformaciones.

Una primera transformación es el cálculo de una variable a partir de otras variables. En este ejemplo, calcularemos la variable *Mamíferos*, que será la suma de las variables *Gatos* y *Perros*. Para hacer este cálculo, recurriremos a esta ruta:

Datos / Modificar variables del conjunto de datos activo / Crear una nueva variable

Aparecerá el siguiente cuadro de diálogo, en el que hemos de introducir, a la izquierda, el nombre de la nueva variable, y a la derecha su expresión. En esta segunda ventana podemos introducir cualquier operación algebraica o función de R.



Creando nuevas variables

En este ejemplo, hemos creado una nueva variable simplemente sumando dos variables originales.

Si visualizamos de nuevo el conjunto de datos, veremos cómo se ha incorporado en la columna derecha la variable que hemos creado recientemente.

	Perros	Gatos	Loros	Mamiferos
1	7	8	7	15
2	2	6	2	8
3	7	8	6	15
4	3	2	5	5
5	6	2	2	8
6	5	4	3	9
7	2	7	9	9
8	1	8	7	9
9	4	6	4	10

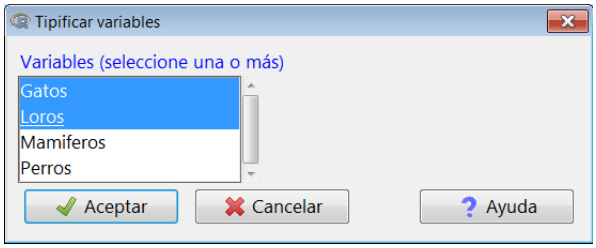
Una transformación muy usada a la hora de realizar inferencia estadística es la estandarización o tipificación. Esta consiste en que, partiendo de una variable aleatoria X que sigue una distribución normal, tal que $X \sim N(\mu, \sigma)$, se transforma en una variable Z tal que

$$Z = \frac{X - \mu}{\sigma}.$$

Los valores de la variable resultante permitirán el cálculo de áreas de probabilidad en la distribución normal estándar, ya que $Z \sim N(0, 1)$. En R-Commander, para estandarizar las variables *Gatos* y *Loros*, la ruta que seguiremos es la siguiente:

Datos / Modificar variables del conjunto de datos activo / Tipificar variables

Nos aparecerá un menú en el que hemos de elegir las variables que deseamos estandarizar.



Estandarizar con R

La función de R `scale` se utiliza para estandarizar vectores y matrices.

Si volvemos a visualizar el conjunto de datos activo, comprobamos cómo se han añadido en las columnas de la derecha las nuevas variables creadas, con una Z delante, que indica que dichas variables han sido estandarizadas.

Animales

	Perros	Gatos	Loros	Mamiferos	Z.Gatos	Z.Loros
1	7	8	7	15	0.9525793	0.8164966
2	2	6	2	8	0.1360828	-1.2247449
3	7	8	6	15	0.9525793	0.4082483
4	3	2	5	5	-1.4969104	0.0000000
5	6	2	2	8	-1.4969104	-1.2247449
6	5	4	3	9	-0.6804138	-0.8164966
7	2	7	9	9	0.5443311	1.6329932
8	1	8	7	9	0.9525793	0.8164966
9	4	6	4	10	0.1360828	-0.4082483

2. Análisis del mercado de trabajo en España

2.1. Importación y manejo de datos

En esta sección hacemos un análisis estadístico del mercado de trabajo en España. Para ello, trabajamos con datos de la EPA provenientes del Instituto Nacional de Estadística (INE). Específicamente, para el 4.º trimestre del 2012 y para cada provincia consideramos las siguientes variables:

Encuesta de población activa (EPA)

La EPA está elaborada por el Instituto Nacional de Estadística (INE) con periodicidad trimestral y tiene como objetivo ofrecer indicadores sobre la situación del mercado laboral en España. Se considera como el mejor indicador de la evolución del empleo y desempleo en España.

- **act:** población activa.
- **inact:** población inactiva.
- **empl:** población activa empleada.
- **par:** población aciva en paro.

El primer paso en el análisis es especificar la ruta al directorio de trabajo. Esto es fundamental, ya que los datos se extraerán desde ese ahí, y todo lo que se guarde también se almacenará en esa carpeta. La función que utilizaremos será `setwd`, (*set working directory*). Si nuestro directorio de trabajo se encuentra en *C:/directorio*, simplemente hemos de introducir la siguiente instrucción, **sin olvidar introducir el directorio entre comillas**.

¡Organización!

Es fundamental disponer de la información y los archivos bien organizados en una carpeta. Esto nos facilitará el trabajo a la hora de importar y exportar datos provenientes del análisis estadístico.

```
> setwd("C:/directorio")
```

El siguiente paso es importar los datos de un archivo externo o crearlos nosotros mismos. Como hemos visto en el primer capítulo de este módulo, existen varias opciones a la hora de importar datos de distintas clases de archivos. Trabajando con R directamente, una opción recomendable es que el archivo original de datos sea un documento de texto (formato *txt*) u hoja de cálculo (formato *csv*, esto es, valores separados por comas). En este ejemplo consideramos la primera opción, es decir, un documento de texto. Este documento ha de tener las variables en columnas y suele contener el nombre de cada variable en la primera línea. Es importante que la última línea de este archivo no esté en blanco para que la importación de los datos se lleve a cabo correctamente.

La función más adecuada para importar datos en este formato es `read.delim2`. El primer elemento que introducir es el nombre del archivo entre comillas. Si la primera línea está ocupada por el nombre de la variables (es decir, es un texto y no un número), especificaremos `header=TRUE`. Además, si en los valores numéricos los decima-

Si el documento de texto que contiene los datos que queremos importar no contiene el nombre de las variables en la primera fila, deberemos introducir la opción `header=FALSE` en la función de importación de datos `read.delim2`.

les fueran puntos en lugar de comas, deberíamos especificarlo mediante el comando `dec="."`. La función `read.delim2` guarda los datos en un objeto tipo base de datos, al cual, en un alarde de originalidad, llamaremos `base.datos`:

```
> base.datos <- read.delim2("datos.txt",header=TRUE)
```

Una manera de comprobar que los datos se han importado correctamente es mediante la función `head`, la cual muestra en consola las seis primeras observaciones en filas, con las variables en columnas.

Visualizar bases de datos

Las funciones `head` y `tail` (cabeza y cola) nos permiten visualizar el principio y el final de un conjunto de datos respectivamente, y se suelen utilizar para comprobar que los datos han sido importados o introducidos correctamente.

```
> head(base.datos)
  Provincia   act  empl   par inact
1  Albacete 182.9 120.7  62.2 145.1
2  Alicante 894.8 639.2 255.6 707.7
3   Almería 371.7 235.7 136.0 190.5
4     Alava 158.4 129.4  29.0  98.0
5  Asturias 480.4 366.2 114.1 437.0
6     Ávila  76.1  59.2  16.9  64.4
```

De igual modo, la función `tail` muestra las últimas seis observaciones.

```
> tail(base.datos)
    Provincia    act  empl   par inact
47   Valencia 1304.6 941.2 363.4 774.3
48 Valladolid  269.0 215.7  53.2 178.4
49     Zamora   74.0  55.3  18.7  90.5
50   Zaragoza  488.7 391.6  97.0 310.7
51      Ceuta   33.4  20.8  12.6  27.1
52    Melilla   32.2  23.1   9.1  26.6
```

La función `dim` se aplica tanto a matrices como a bases de datos y nos indica la dimensión del objeto. En este caso, comprobamos que tenemos 5 variables, cada una con 52 observaciones.

La función `dim`

Esta función, aplicada a una base de datos, nos muestra en primer lugar el número de filas (observaciones) y en segundo lugar el número de columnas (variable).

```
> dim(base.datos)
[1] 52  5
```

Siempre es recomendable empezar un análisis con una visión general de las variables. Para ello, como vimos al realizar el análisis descriptivo con R-Commander, existen diferentes funciones entre las cuales se encuentra `summary`. Recordemos que esta función muestra para cada variable el valor mínimo, máximo, la media y los tres cuartiles.

```
> summary(base.datos)
    Provincia       act              empl
 Alava   : 1   Min.   :  32.2   Min.   :  20.8
 Albacete: 1   1st Qu.: 151.9   1st Qu.: 121.3
 Alicante: 1   Median : 298.0   Median : 215.0
 Almería : 1   Mean   : 440.8   Mean   : 326.1
 Asturias: 1   3rd Qu.: 499.3   3rd Qu.: 362.8
 Ávila   : 1   Max.   :3347.4   Max.   :2682.0
 (Other) :46
      par             inact
 Min.   :  7.00   Min.   :  26.6
 1st Qu.: 29.98   1st Qu.: 117.0
 Median : 75.35   Median : 203.7
 Mean   :114.71   Mean   : 296.4
 3rd Qu.:132.32   3rd Qu.: 326.9
 Max.   :665.30   Max.   :1900.3
```

Para hacer referencia a una variable de una base de datos, la sintaxis adecuada será `base.datos$variable`. Así pues, la variable `par` se extrae de la siguiente manera.

El símbolo $

Recordemos que este símbolo se utiliza para acceder a los distintos componentes de un objeto. En este caso, se utiliza para acceder a las diferentes variables que componen una base de datos.

```
> base.datos$par
 [1]  62.2 255.6 136.0  29.0 114.1  16.9 117.4 144.1 651.5
[10]  94.3  32.7  56.2 233.6  53.5  81.1  69.6 131.1 114.0
[19]  21.9  39.5  91.5 162.9  35.0  89.5  14.8 111.6  48.1
[28]  37.3  25.9 665.3 275.4 216.6  52.6  30.3  14.1 198.8
[37] 107.1  28.0  34.1 169.6  15.9 302.5   7.0 104.8  10.1
[46] 107.9 363.4  53.2  18.7  97.0  12.6   9.1
```

No obstante, esto puede resultar engorroso a la hora de programar un análisis con las variables. Por ello, es recomendable usar la función `attach`. De esta manera, se podrá acceder a las variables simplemente dando sus nombres, sin hacer referencia a la base de datos.

La función `attach`

Esta función, aplicada a una base de datos, incrusta todas las variables de esta en la memoria en la forma de objetos (vectores) independientes, de manera que para referirnos a estas variables no haga falta usar repetidamente el símbolo $.

```
> attach(base.datos)
```

Seguiremos el análisis creando nuevas variables a partir de la base de datos importada. Los principales indicadores del mercado laboral son los siguientes:

$$\text{Tasa de actividad} = 100 \cdot \frac{\text{Población activa}}{\text{Población >16 años}}$$

$$\text{Tasa de empleo} = 100 \cdot \frac{\text{Población empleada}}{\text{Población >16 años}}$$

$$\text{Tasa de paro} = 100 \cdot \frac{\text{Población en paro}}{\text{Población activa}}$$

En R, la creación de estas variables es inmediata.

```
> pob <- act+inact
> t.act  <- 100*act/pob
> t.empl <- 100*empl/pob
> t.paro <- 100*par/act
```

Cada una de estas variables tiene 52 observaciones (tantas como provincias). Para calcular las tasas antes descritas a nivel estatal, tendremos que sumar por provincias.

La función `sum`

Recordemos que la función `sum` aplica la operación suma a todos los componentes de un objeto, en este caso las variables de nuestro análisis.

```
> # Tasa de actividad:
> 100*sum(act)/sum(pob)
[1] 59.79814

> # Tasa de empleo:
> 100*sum(empl)/sum(pob)
[1] 44.2359

> # Tasa de paro:
> 100*sum(par)/sum(act)
[1] 26.02201
```

Si quisiéramos estos indicadores en tanto por uno, bastaría con no multiplicar por 100.

2.2. Estadística descriptiva

R ofrece muchas posibilidades para hacer un análisis descriptivo de datos. Además de disponer de muchas librerías con estadísticos y procedimientos, su flexibilidad permite que podamos programar nuestro propio análisis descriptivo a medida. Para empezar, veamos cómo podemos calcular la media aritmética de la tasa de actividad.

```
> mean(t.act)
[1] 57.74514
```

Antes de realizar un análisis descriptivo de las tres tasas calculadas anteriormente, es recomendable unirlas por columnas para formar una matriz 52×3. Además, con la función `row.names` le asignamos un nombre a cada observación (fila), con la variable `Provincia`, la cual contiene los nombres de las provincias.

Las funciones `cbind` y `rbind`

Estas funciones nos permiten unir objetos por columnas y por filas, respectivamente.

```
> indic <- cbind(t.act,t.empl,t.paro)
> row.names(indic) <- Provincia
```

Como vimos anteriormente, la función `summary` muestra el valor mínimo, máximo, la media y los tres cuartiles de cada variable.

```
> summary(indic)
     t.act           t.empl          t.paro
 Min.   :44.98   Min.   :33.62   Min.   :12.59
 1st Qu.:54.93   1st Qu.:37.88   1st Qu.:19.64
 Median :58.13   Median :42.96   Median :24.26
 Mean   :57.75   Mean   :42.93   Mean   :25.56
 3rd Qu.:60.92   3rd Qu.:47.75   3rd Qu.:31.65
 Max.   :66.63   Max.   :51.11   Max.   :40.63
```

Una función muy útil a la hora de realizar un cálculo para un conjunto de variables es `apply`. En este caso, aplicaremos a la matriz de variables `indic`, y solo para la dimensión 2 (columnas), la función `mean` (media aritmética). El resultado será un vector con las medias aritméticas de las tres variables de interés.

```
> apply(indic,2,mean)
   t.act   t.empl   t.paro
57.74514 42.93311 25.56296
```

Un cálculo un poco más complejo pero potencialmente útil es el de crear una función propia para obtener un conjunto de estadísticos para un grupo de variables. En nuestro caso, llamaremos a esta función `estad.basic`, y consistirá en la media, la varianza, la desviación estándar, el mínimo, los tres cuartiles y el máximo (estos últimos calculados con la función `quantile`). El resultado lo redondearemos a 2 decimales mediante la función `round`.

```
> estad.basic <- function(x){
+   est <- cbind(mean(x),var(x),sd(x),t(quantile(x)))
+   colnames(est) <- c("media","var","desv.est","min",
+   "Q1","Q2","Q3","max")
+   return(round(est,2))
+ }
```

La sintaxis de la creación de funciones está ampliamente descrita en el primer módulo.

Si aplicamos esta función a la variable *tasa de actividad* (`t.act`), obtenemos el siguiente resultado:

```
> estad.basic(t.act)
     media  var desv.est   min    Q1    Q2    Q3   max
[1,] 57.75 22.2     4.71 44.98 54.93 58.13 60.92 66.63
```

Si combinamos las funciones `apply` y `estad.basic`, podremos obtener con un solo cálculo los estadísticos seleccionados aplicados a todas las variables simultáneamente, y quedarán los resultados registrados en una matriz.

```
> est.total <- apply(indic,2,estad.basic)
> rownames(est.total)<- c("media","var","desv.est","min",
+ "Q1","Q2","Q3","max")
> print(est.total)
         t.act t.empl t.paro
media    57.75  42.93  25.56
var      22.20  25.99  53.43
desv.est  4.71   5.10   7.31
min      44.98  33.62  12.59
Q1       54.93  37.88  19.64
Q2       58.13  42.96  24.26
Q3       60.92  47.75  31.65
max      66.63  51.11  40.63
```

2.3. Representación gráfica

Las posibilidades gráficas que ofrece R son enormes, mucho más amplias que las que ofrece R-Commander. En esta sección veremos solo una pequeña parte de todo ese potencial. El objetivo es obtener, con los datos analizados hasta ahora, los tipos de gráficos más usados en estadística.

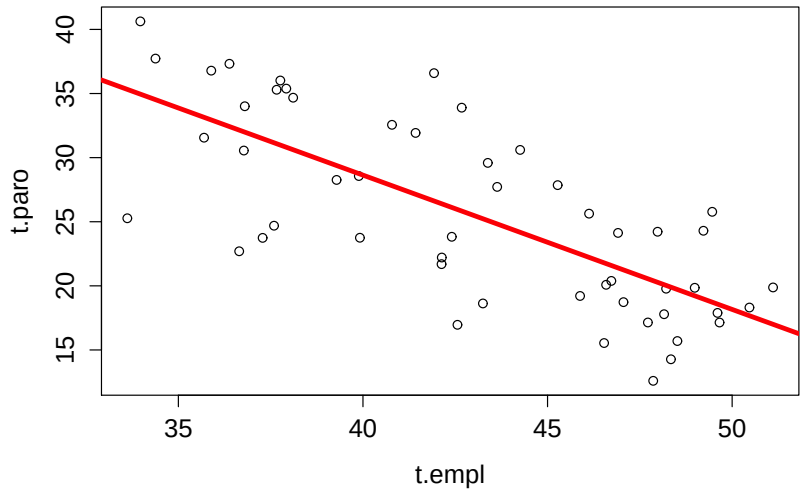
2.3.1. Diagrama de dispersión

Un diagrama de dispersión es un gráfico bidimensional que utiliza las coordenadas cartesianas para mostrar los valores de dos variables, una en cada eje. Si deseamos este diagrama para las variables *tasa de empleo* y *tasa de paro*, bastará con introducirlas como argumentos de la función `plot`. Además, opcionalmente podemos superponer a este gráfico más elementos, por ejemplo rectas mediante la función `abline`. En nuestro ejemplo, añadiremos una recta ajustada a los puntos, para lo cual usaremos la función `lm` (*linear model*). Dibujaremos esta recta en rojo (`red`) y con un grosor `lwd=4`.

Las opciones `col` y `lwd`

Estas opciones son muy útiles a la hora de personalizar los gráficos, ya que permiten definir el color y el grosor de los puntos y líneas representadas.

```
> plot(t.empl,t.paro)
> abline(lm(t.paro~t.empl),col="red",lwd=4)
```

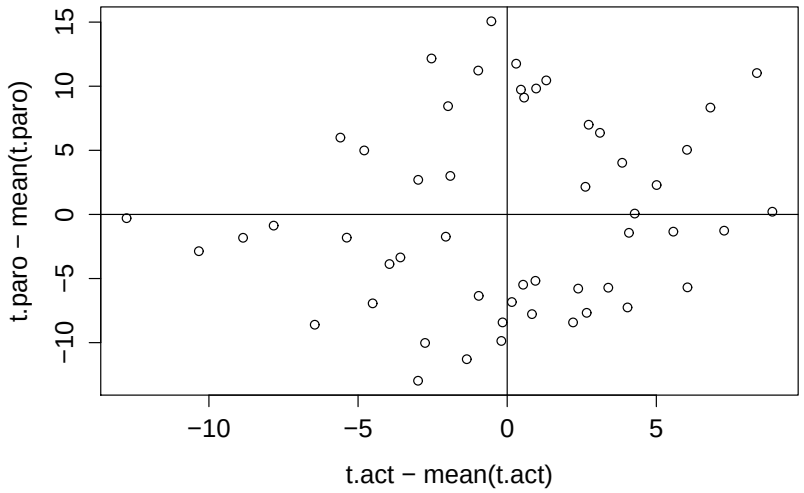


la opción `abline`

Esta opción nos permite añadir todo tipo de líneas a gráficos ya creados, a modo de capas superpuestas.

Otro ejemplo de diagrama de dispersión es el de las variables centradas en el valor cero, para lo cual hay que aplicar la transformación $x_i - \bar{x}$, es decir, restar la media aritmética. Adicionalmente se puede añadir una línea horizontal `h=0` y vertical `v=0` en el valor cero.

```
> plot(t.act-mean(t.act),t.paro-mean(t.paro))
> abline(h=0)
> abline(v=0)
```

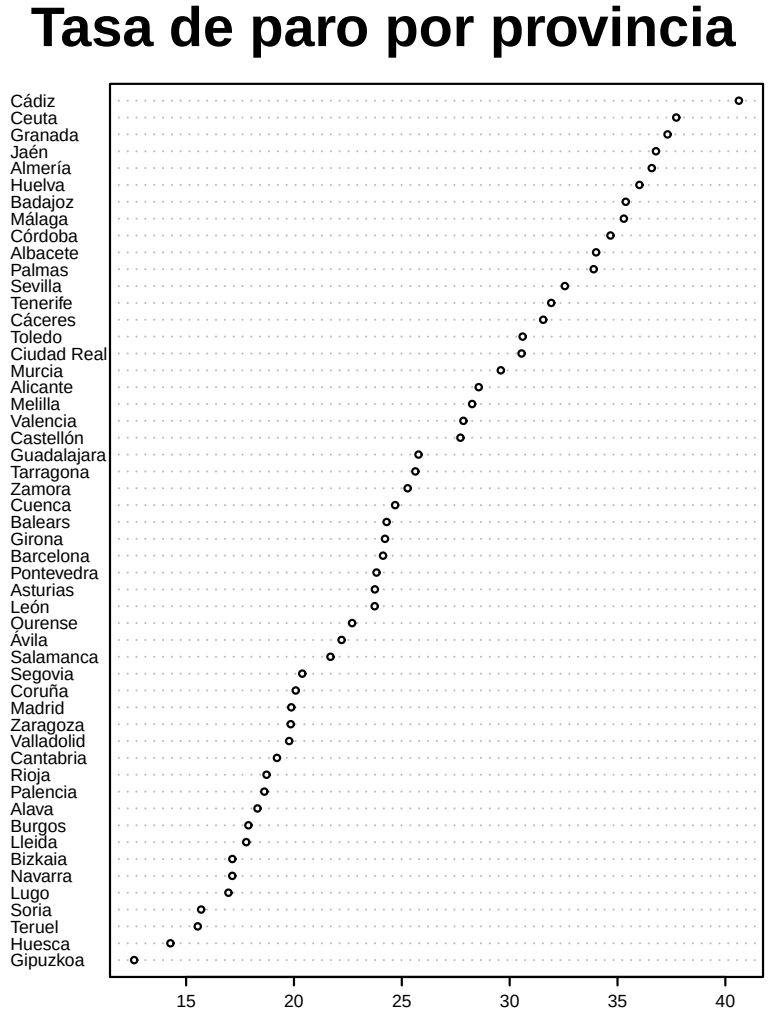


Las opciones `labels` y `cex`

Estas opciones permiten introducir etiquetas y regular la escala del gráfico.

La función `dotchart` dibuja un diagrama de puntos unidimensional, identificando cada punto con su nombre. Para hacerlo más completo, utilizaremos la función `sort` con el fin de que los datos aparezcan ordenados de menor a mayor, y consideraremos los nombres de la matriz `indic`. La instrucción `cex` hace referencia a la escala del gráfico con respecto a 1. A este gráfico le añadiremos un título mediante la función `title`.

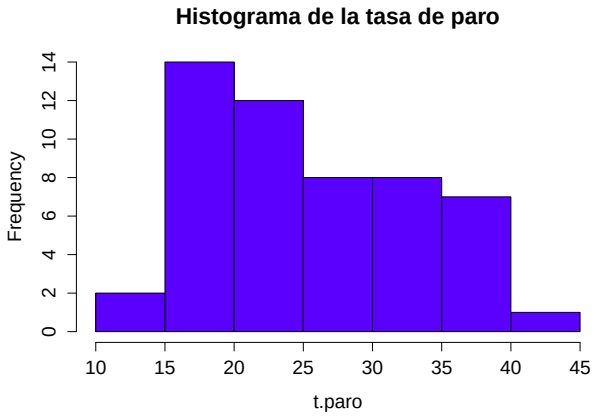
```
> dotchart(sort(indic[,3]),labels=names(indic),cex=.4)
> title("Tasa de paro por provincia")
```



2.3.2. Histograma y función de densidad

Este tipo de gráfico se emplea para visualizar la distribución de una variable. El histograma se representa mediante la función `hist` de la siguiente manera:

```
> hist(t.paro,main="Histograma de la tasa de paro",col="blue")
```



Histograma y diagrama de barras

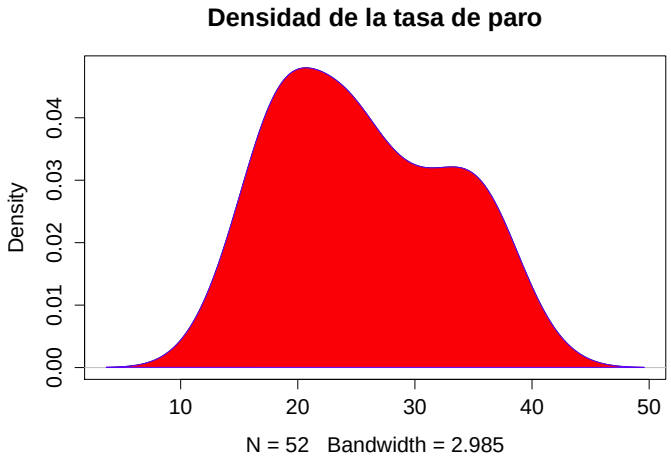
Aunque se parecen, estos dos tipos de gráfico no son idénticos. El histograma se emplea para representar **datos cuantitativos continuos**, mientras que el **diagrama de barras** se usa para representar gráficamente datos cuantitativos discretos o datos cualitativos.

La opción `main`

Esta opción se utiliza para introducir un título general a un gráfico.

La función de densidad, calculada con técnicas no paramétricas, es la generalización del histograma asumiendo que el grosor de las barras tiende a cero. Visto de otra manera, el histograma es discreto y la función de densidad continua. El primer paso será estimar la función de densidad mediante la función `density`, cuyo resultado será un conjunto de valores. Para representarlos gráficamente, usaremos la función `plot`. Opcionalmente, además, podremos colorear esta función (tanto el borde como el interior) mediante la función `polygon`. Es importante ver cómo R elabora gráficos complejos añadiendo capas a partir de una instrucción inicial.

```
> den.paro <- density(t.paro)
> plot(den.paro,main="Densidad de la tasa de paro")
> polygon(den.paro, col="red", border="blue")
```

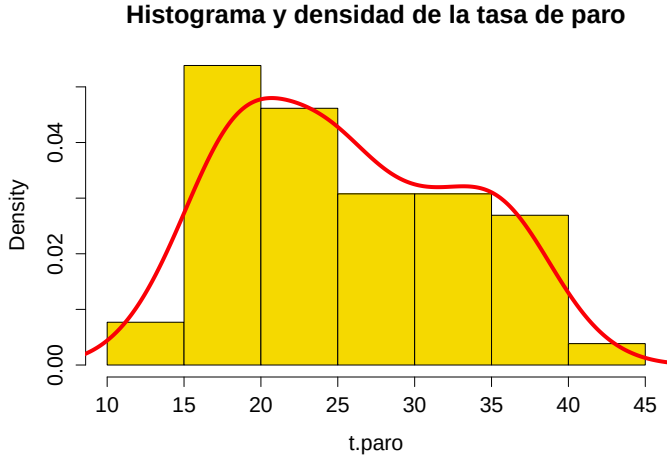


Otra opción es la de combinar un histograma con la estimación de la función de densidad. Mediante la instrucción `freq=FALSE` establecemos que en el eje de abscisas no apareza la frecuencia, sino la densidad de probabilidad. Además, mediante la función `lines` añadimos al gráfico una capa adicional con la estimación de la función de densidad.

```
> hist(t.paro,main="Histograma y densidad de la tasa de paro",
+ col="gold",freq=FALSE)
> lines(den.paro,col="red",lwd=4)
```

Combinando un histograma y la función de densidad

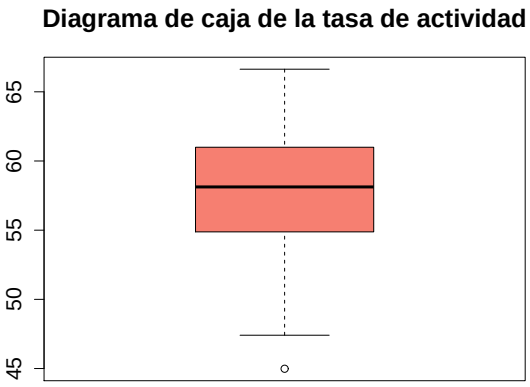
Estas líneas de código muestran cómo se pueden combinar ambos tipos de gráficos, y también cómo en R la creación de gráficos es secuencial, esto es, línea a línea de una manera superpuesta.



2.3.3. Diagrama de caja

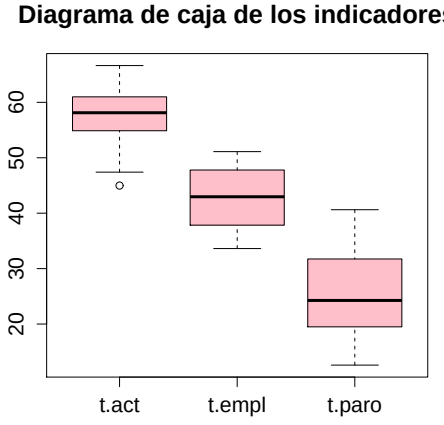
El diagrama de caja consiste en un gráfico basado en cuartiles, mediante el cual se puede visualizar la simetría de la distribución de los datos. La función de R que produce este tipo de gráfico es `boxplot`. Veamos el diagrama de caja de la tasa de actividad:

```
> boxplot(t.act,col="salmon",main="Diagrama de caja de la tasa de
+ actividad")
```



Si lo que nos interesa es la comparación de distintos diagramas de caja de varias variables en un mismo gráfico, introduciremos como primer argumento una matriz (o base de datos) con las variables dispuestas en columnas. En nuestro caso, la matriz `indic` incluye las tres variables del mercado laboral:

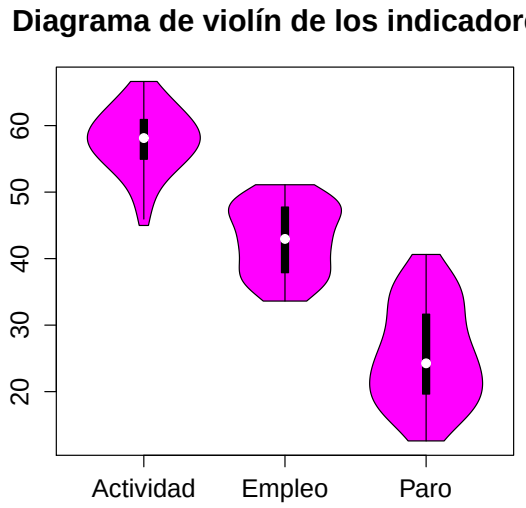
```
> boxplot(indic,col="pink",main="Diagrama de caja de los indicadores")
```



Otra opción interesante es el diagrama de violín, que combina el diagrama de caja con la función de densidad en un solo gráfico. Antes deberemos instalar el paquete `vioplot` y cargar la librería. El diagrama de violín de las tres variables de interés toma la siguiente forma:

Para poder cargar un paquete (*library*), antes hay que tenerlo instalado. Recordemos también que solo es necesario instalar los paquetes una sola vez.

```
> library(vioplot)
> vioplot(t.act,t.empl,t.paro,names=c("Actividad","Empleo",
+ "Paro"))
> title("Diagrama de violín de los indicadores")
```



2.3.4. Gráficos compuestos

A veces, por cuestión de espacio o de síntesis, necesitaremos combinar varios gráficos en un solo archivo o imagen. Esto se hace mediante la instrucción `par`. Así, en el siguiente ejemplo dispondremos cuatro gráficos en dos filas y dos columnas mediante la instrucción `mfrow=c(2,2)`. Los gráficos que introduzcamos a continuación se irán colocando por filas. Una vez obtenido el gráfico, para restaurar los gráficos a sus valores iniciales, concluiremos con la instrucción `dev.off()`, con lo que se borrará el gráfico compuesto.

Elaborando gráficos compuestos

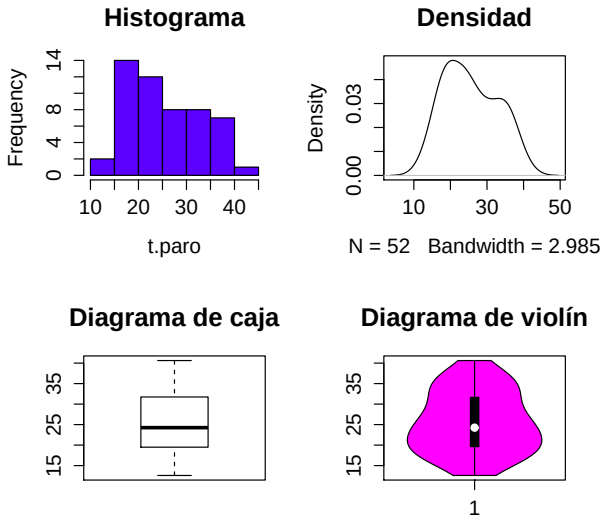
Mediante la función `par` podemos combinar varios gráficos en un solo marco. Mediante la opción `mfrow` establecemos cómo se disponen los diferentes gráficos. En nuestro caso, en dos filas y dos columnas respectivamente.

La instrucción `dev.off()`

Esta instrucción borra el gráfico elaborado y restablece los valores gráficos por defecto. Si queremos guardar el gráfico, deberemos hacerlo **antes** de introducir esta última instrucción.

```
> par(mfrow=c(2,2))
> hist(t.paro,col="blue",main="Histograma")
> plot(density(t.paro),main="Densidad")
> boxplot(t.paro,main="Diagrama de caja")
> vioplot(t.paro)
> title("Diagrama de violín")
> dev.off()
null device
          1
```

Múltiples indicadores de la tasa de paro



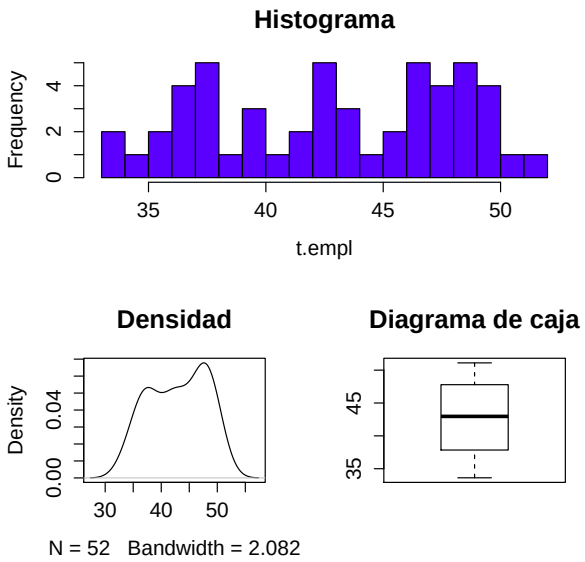
Para hacer composiciones más complejas disponemos de la función `layout`. En el siguiente ejemplo realizaremos tres gráficos en una composición 2×2, es decir, dos filas y dos columnas. Establecemos que el primer gráfico ocupe toda la primera fila (es decir, ambas columnas), mientras que los dos gráficos restantes ocupen cada uno una columna de la segunda fila. En el resultado se puede apreciar cómo el primer gráfico aparece alargado horizontalmente debido a que ocupa el doble de espacio que los otros dos gráficos.

```
> layout(matrix(c(1,1,2,3), 2, 2, byrow = TRUE))
> hist(t.empl,col="blue",main="Histograma",breaks=20)
> plot(density(t.empl),main="Densidad")
> boxplot(t.empl,main="Diagrama de caja")
> dev.off()
null device
          1
```

La función `layout`

Esta función permite disponer gráficos de una manera muy flexible, como se muestra en este ejemplo.

Múltiples indicadores de la tasa de empleo



3. Análisis demográfico en Cataluña

Esta sección está dedicada al estudio de datos demográficos de los municipios de Cataluña. Con este estudio explicaremos cómo hacer un estudio estadístico descriptivo introduciendo variables discretas, que se analizan mediante factores. La base de datos que vamos a analizar incorpora las siguientes variables:

- *municipio:* nombre del municipio.
- *sup:* superficie municipal en km^2.
- *edad:* media de edad de los habitantes.
- *pobl:* población total del municipio.
- *inmig:* porcentaje de población inmigrante.
- *capital:* variable dicotómica que toma el valor 1 si el municipio es capital de comarca, y 0 en caso contrario.
- *costa:* variable dicotómica que toma el valor 1 si el municipio está situado en la costa, y 0 en caso contrario.
- *bcn:* variable dicotómica que toma el valor 1 si el municipio está situado en el área metropolitana de Barcelona, y 0 en caso contrario.

3.1. Manejo de bases de datos

Para empezar el análisis, cargaremos al principio las librerías que vamos a utilizar durante el análisis. La primera es una librería que permite la realización de gráficos de datos con factores, y la segunda permite el cálculo de momentos estadísticos.

```
> library(lattice)
> library(moments)
```

Es recomendable tener instalada una versión reciente de R para garantizar que las librerías más recientes se carguen correctamente.

El siguiente paso, análogamente al caso anterior, será importar los datos desde un archivo externo, en este caso con extensión *R*.

```
> base.datos <- read.delim2("datos_demografia.R",header=TRUE)
```

Veamos las seis primeras observaciones de la base de datos para comprobar que se ha realizado correctamente.

```
> head(base.datos)
            municipio   sup  edad  pobl inmig capital costa bcn
1              Abrera  1.53 40.67 11278  9.47       0     0   1
2  Aguilar de Segarra  0.11 48.90   249  5.22       0     0   0
3              Alella  2.67 44.47  9260  9.01       0     0   1
4              Alpens  0.25 45.95   312  6.41       0     0   0
5  Ametlla del Vallès  4.44 40.92  7796  7.21       0     0   1
6       Arenys de Mar  2.04 46.61 14449 10.97       0     1   1
```

El sumario de la base datos nos proporciona estadísticos básicos de la variable: mínimo y máximo, media aritmética y cuartiles.

```
> summary(base.datos)
                     municipio         sup              edad
   Abrera                  :  1   Min.   : 0.010   Min.   :36.81
   Aguilar de Segarra      :  1   1st Qu.: 0.130   1st Qu.:43.90
   Alella                  :  1   Median : 0.360   Median :47.47
   Alpens                  :  1   Mean   : 1.183   Mean   :47.95
   Ametlla del Vallès (L'):  1   3rd Qu.: 1.120   3rd Qu.:51.37
   Arenys de Mar           :  1   Max.   :75.290   Max.   :68.43
 (Other)                   :935
      pobl             inmig           capital
 Min.   :     29   Min.   : 0.000   Min.   :0.00000
 1st Qu.:    329   1st Qu.: 4.510   1st Qu.:0.00000
 Median :    957   Median : 8.400   Median :0.00000
 Mean   :   7826   Mean   : 9.811   Mean   :0.04357
 3rd Qu.:   3659   3rd Qu.:13.190   3rd Qu.:0.00000
 Max.   :1615908   Max.   :49.930   Max.   :1.00000

    costa              bcn
Min.   :0.00000   Min.   :0.0000
1st Qu.:0.00000   1st Qu.:0.0000
Median :0.00000   Median :0.0000
Mean   :0.07439   Mean   :0.2306
3rd Qu.:0.00000   3rd Qu.:0.0000
Max.   :1.00000   Max.   :1.0000
```

Interpretación del resumen de las variables

En este caso, es interesante ver cómo se interpreta la media aritmética de las variables dicotómicas. En este caso, vemos cómo el $4,3\,\%$ de los municipios son capital, el $7,4\,\%$ están situados en la costa y el $23\,\%$ están situados en el área metropolitana de Barcelona.

Es recomendable usar la función `attach`. De esta manera, se podrá acceder a las variables simplemente dando sus nombres, sin hacer referencia a la base de datos.

```
> attach(base.datos)
```

En muchas ocasiones, para agilizar la explotación de los datos que queremos analizar, nos encontramos con la necesidad de crear bases de datos más reducidas a partir de la base de datos original. En R esto se puede realizar mediante la función `subset`. Empecemos con un primer ejemplo muy sencillo: extraeremos de la base de datos inicial solo dos variables (el municipio y la edad), y solo aquellas observaciones que cumplan la condición `edad>65`. Así pues, la base de datos resultante contiene solo dos variables y tres observaciones que cumplen esta condición.

La función subset

Esta es una función fundamental en el análisis de datos, ya que permite crear bases de datos menores a partir de una base de datos original mediante la introducción de condiciones.

```
> subset(base.datos,edad>65,select=c(municipio,edad))
                municipio  edad
571                Bausen 67.75
819                 Forès 68.43
916  Vallfogona de Riucorb 66.99
```

Seguiremos con otro ejemplo un poco más complejo. En este caso extraemos tres variables (municipio, edad y tasa de inmigración), y solo nos interesan las observaciones que cumplan dos condiciones: media de edad menor a 40 años y tasa de inmigración superior al 30 %. Vemos que la base de datos resultante solo contiene dos municipios.

```
> subset(base.datos,edad<40 & inmig>30,
+        select=c(municipio,edad,inmig))
    municipio  edad inmig
452      Salt 39.54 39.20
624  Guissona 38.50 43.46
```

El siguiente paso es ver cómo podemos reordenar las observaciones de una base de datos según el orden creciente o decreciente de una variable. En este caso, el vector `order(edad)` será un vector que contiene los valores $1 : 941$ según el orden creciente de la variable edad. Si ordenamos la base de datos según este vector, y vemos sus primeros seis valores (`head`), obtenemos el siguiente resultado:

```
> head(base.datos[order(edad),])
               municipio  sup  edad pobl inmig capital costa bcn
857      Pallaresos (Els) 1.00 36.81 3828  2.59       0     0   0
167              Polinyà 2.44 37.09 7403  6.85       0     0   1
866 Pobla de Mafumet (La) 2.77 38.29 2108  7.78       0     0   0
355                Celrà 0.93 38.46 4329 15.64       0     0   0
624             Guissona 1.02 38.50 5683 43.46       0     0   0
504         Vall-llobrega 0.11 38.77  825  9.58       0     0   0
```

La función order

Esta función sirve para ordenar vectores en orden ascendente. Si se le coloca al vector el signo negativo delante, el resultado es el orden descendente.

Hagamos ahora un cálculo análogo al caso anterior pero esta vez ordenando las observaciones *en order decreciente* (mediante el signo negativo) de la variable inmigración `order(-inmig)`:

```
> head(base.datos[order(-inmig),])
              municipio  sup  edad  pobl inmig capital costa bcn
353  Castelló d'Empúries 5.85 43.83 11653 49.93       0     1   0
624             Guissona 1.02 38.50  5683 43.46       0     0   0
940                Salou 4.11 40.02 25754 40.26       0     1   0
398        Lloret de Mar 8.75 40.58 37734 39.58       0     1   0
499                 Ullà 0.18 41.25  1067 39.27       0     0   0
452                 Salt 1.62 39.54 28763 39.20       0     0   0
```

3.2. Creación y análisis de variables

Para el análisis que realizaremos a continuación es necesario crear la variable `densidad`, definida como la población dividida por la superficie y por 1,000.

```
> densidad <- pobl/(1000*sup)
```

Igual que en la sección anterior, crearemos la función `estad.basic`, consistente en la media, la varianza, la desviación estándar, el mínimo, los tres cuartiles y el máximo (estos últimos calculados con la función `quantile`), redondeando a 2 decimales mediante la función `round`.

```
> estad.basic <- function(x){
+   est <- cbind(mean(x),var(x),sd(x),t(quantile(x)))
+   colnames(est) <- c("media","var","desv.est","min",
+   "Q1","Q2","Q3","max")
+   return(round(est,2))
+ }
```

Seguidamente creamos la matriz `demo`, uniendo por columnas las variables densidad, media de edad y tasa de inmigración de cada municipio. A esta matriz le aplicamos la función `estad.basic` por columnas, obteniendo las estadísticas básicas de cada una de estas tres variables para el total de municipios.

```
> demo <- cbind(densidad,edad,inmig)

> est.total <- apply(demo,2,estad.basic)
> rownames(est.total) <- c("media","var","desv.est","min",
+ "Q1","Q2","Q3","max")

> print(est.total)
         densidad  edad inmig
media        5.43 47.95  9.81
var         95.68 25.69 51.31
desv.est     9.78  5.07  7.16
min          0.13 36.81  0.00
Q1           1.98 43.90  4.51
Q2           3.22 47.47  8.40
Q3           5.38 51.37 13.19
max        159.90 68.43 49.93
```

La matriz de correlación lineal de estas variables se calcula mediante la función `cor`:

```
> cor(demo)
            densidad        edad       inmig
densidad  1.00000000 -0.02328298  0.04810744
edad     -0.02328298  1.00000000 -0.28050864
inmig     0.04810744 -0.28050864  1.00000000
```

La función cor

Esta función es vital para calcular cuál es la correlación lineal entre dos o más variables. Introduciendo en la consola `help(cor)` veremos las diferentes maneras de calcular esta matriz.

3.3. Creación y análisis de factores

R ofrece muchas posibilidad de análisis de variables cuantitativas y cualitativas. Especialmente interesante es el de las variables discretas, es decir, que toman un número limitado de números enteros. Las variables cualitativas y discretas suelen contener información sobre las características de las variables. En nuestro ejemplo disponemos de tres variables dicotómicas o binarias: *capital*, *costa* y *bcn*. Es recomendable, mediante la función `factor`, crear factores a partir de estas variables, añadiendo también una etiqueta como se muestra a continuación.

Variables discretas, dicotómicas y binarias

Una variable **discreta** puede tomar un número limitado de valores enteros, y una variable **dicotómica** o **binaria** solo toma dos valores. Es decir, una variable dicotómica es una variable discreta, pero lo contrario no tiene por qué ser cierto.

```
> f_cap <- factor(capital,labels=c("no_cap","cap"))
> f_cos <- factor(costa,labels=c("no_costa","costa"))
> f_bcn <- factor(bcn,labels=c("no_bcn","bcn"))
```

Es fundamental saber crear y manejar factores cuando utilizamos variables cualitativas en nuestro análisis.

La función `table` utiliza los factores para la construcción de una tabla de contingencia para cada combinación de niveles de los factores. Por ejemplo, las 941 observaciones se dividen en cuatro grupos según sean o no capital y estén o no en la costa:

```
> table(f_cap,f_cos)
        f_cos
f_cap    no_costa costa
  no_cap      836    64
  cap          35     6
```

Una función que permite aplicar a una variable una operación diferenciando por el valor de un factor es `tapply`. Por ejemplo, la instrucción `tapply(edad,f_cos,mean)` produce la media de la variable *edad* para los dos grupos (niveles) del factor *costa*. En el siguiente ejemplo comprobamos cómo el número de municipios situados en la costa es de 70, y que tanto la media de edad como la desviación estándar son menores en la costa que en el interior.

```
> muestra <- tapply(edad,f_cos,length)
> media <- tapply(edad,f_cos,mean)
> desv.est <- tapply(edad,f_cos,sd)
> Rtado <- rbind(muestra,media,desv.est)
> row.names(Rtado) <- c("N","Mean","Desv. Est.")
> print(Rtado,digits=3)
           no_costa costa
N            871.00 70.00
Mean          48.21 44.75
Desv. Est.     5.12  2.79
```

La opción `digits`

Esta opción nos permite ajustar el número de decimales del resultado obtenido. Específicamente, nos permite establecer el número mínimo de dígitos significativos para todos los valores mostrados.

Si combinamos la función `tapply` con la función que hemos creado de estadísticos básicos (`estad.basic`), obtendremos una serie de estadísticos para la variable edad, dependiendo de si el municipio está o no en el área de Barcelona. Comprobamos cómo la media de edad en el área metropolitana de Barcelona es menor que en el resto de Cataluña.

```
> tapply(edad,f_bcn,estad.basic)
$no_bcn
     media   var desv.est   min    Q1    Q2    Q3   max
[1,] 48.95 22.57     4.75 36.81 45.44 48.66 52.05 68.43

$bcn
     media   var desv.est   min    Q1    Q2    Q3   max
[1,] 44.62 21.82     4.67 37.09 41.62 43.42 45.49 66.99
```

La función `tapply`

Esta función nos permite obtener información estadística de una variable segmentándola según los valores de un factor.

Siendo un poco más ambiciosos, calcularemos para la variable edad una tabla con los estadísticos de la función `estad.basic` para los tres factores disponibles.

```
> estad.edad <- rbind(
+   estad.basic(edad),
+   tapply(edad,f_cap,estad.basic)$no_cap,
+   tapply(edad,f_cap,estad.basic)$cap,
+   tapply(edad,f_cos,estad.basic)$no_costa,
+   tapply(edad,f_cos,estad.basic)$costa,
+   tapply(edad,f_bcn,estad.basic)$no_bcn,
+   tapply(edad,f_bcn,estad.basic)$bcn
+ )
> rownames(estad.edad) <- c("Total","No Capital","Capital",
+ "Interior","Costa","No BCN","BCN")
```

```
> print(estad.edad)
           media   var desv.est   min    Q1    Q2    Q3   max
Total      47.95 25.69     5.07 36.81 43.90 47.47 51.37 68.43
No Capital 48.10 26.13     5.11 36.81 44.09 47.59 51.52 68.43
Capital    44.55  4.10     2.03 40.42 43.38 44.08 45.70 48.99
Interior   48.21 26.25     5.12 36.81 44.16 47.68 51.60 68.43
Costa      44.75  7.77     2.79 40.02 43.40 44.22 45.51 56.44
No BCN     48.95 22.57     4.75 36.81 45.44 48.66 52.05 68.43
BCN        44.62 21.82     4.67 37.09 41.62 43.42 45.49 66.99
```

Otra funcionalidad de R es el de la creación de variables discretas mediante la partición de variables en intervalos. Por ejemplo, veamos cuál es el rango de la tasa de inmigración:

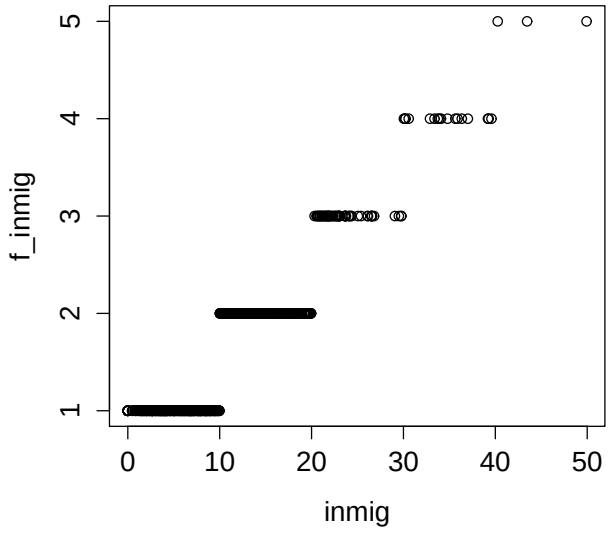
```
> range(inmig)
[1]  0.00 49.93
```

Vemos cómo está acotada, aproximadamente, entre el 0 y el 50 %. Lo que haremos será crear un factor que tome cinco posibles valores discretos, y que se corresponda con los intervalos $[0, 10)$, $[10, 20)$, ...,$[40, 50]$ de la variable `inmig`. Después de crear los puntos de corte, mediante la función `cut` trocearemos la variable `inmig` de manera que obtengamos cinco niveles diferentes en la variable `f_inmig`. Un diagrama de dispersión entre las dos variables permite ver gráficamente la relación entre ambas.

La función cut

Esta función permite obtener una variable discreta a partir de una variable continua. Lo que hace es segmentar esta variable según unos intervalos definidos, creando varias categorías. Mediante la opción `right=FALSE` especificamos que los intervalos sean abiertos por la derecha (y cerrados por la izquierda).

```
> int <- seq(0,50,by=10)
> f_inmig <- cut(inmig,breaks=int,right=FALSE)

> plot(inmig,f_inmig)
```



Segmentación de variables

En este gráfico se puede ver claramente el uso de la función `cut`. El eje horizontal representa la variable continua original, y el eje vertical se corresponde al factor creado, que solo toma cinco valores, correspondientes a los cinco intervalos establecidos.

Una vez creado el factor `f_inmig`, lo podemos cruzar con el factor `f_bcn` para ver cómo se reparten los municipios según su tasa de inmigrantes y su pertenencia al área metropolitana de Barcelona. Para esto, usaremos la función `table`.

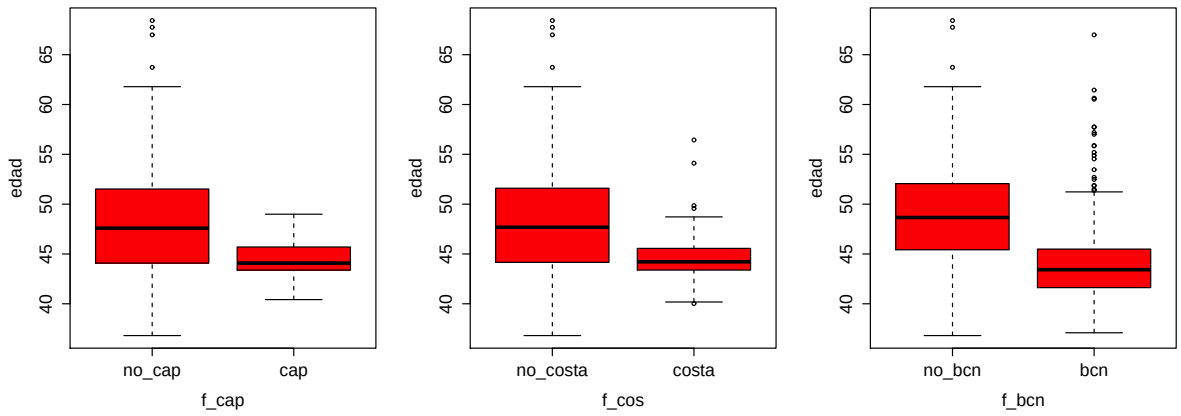
```
> table(f_inmig,f_bcn)
          f_bcn
f_inmig    no_bcn bcn
  [0,10)      417 149
  [10,20)     237  61
  [20,30)      50   7
  [30,40)      17   0
  [40,50)       3   0
```

3.4. Representación gráfica

3.4.1. Gráficos con componente factorial

La presencia de variables cualitativas, discretas o dicotómicas en los datos nos da mucho juego a la hora de componer gráficos donde se muestran posibles relaciones entre estas. Empezaremos con la representación de tres diagramas de caja para la variable `edad`, uno para cada factor (capitalidad, costa y Barcelona).

```
> par(mfrow=c(1,3))
> plot(edad~f_cap,col="red")
> plot(edad~f_cos,col="red")
> plot(edad~f_bcn,col="red")
```



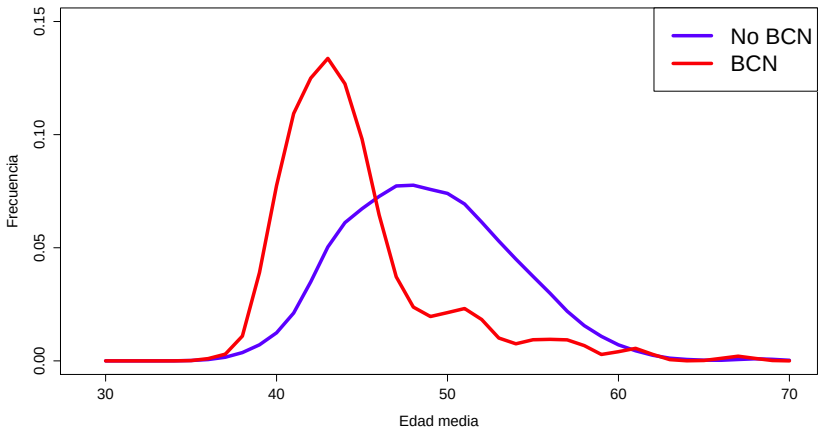
En el siguiente ejemplo haremos algo un poco más complejo: representaremos gráficamente la densidad empírica de la media de edad diferenciando entre municipios dentro y fuera del área metropolitana de Barcelona. El primer paso consistirá en crear dos variables para la media de edad con el fin de calcular posteriormente sus respectivas densidades. Después crearemos un marco para el gráfico, especificando los valores del rango y el dominio, además del nombre de ambos ejes. Este marco, mediante la instrucción `type="n"`, lo dejamos vacío de contenido, ya que seguidamente le añadimos las dos capas mediante la función `lines`, es decir, lo llenamos con las dos funciones de densidad. Por último, incorporamos los valores del eje de abscisas y la leyenda.

```
> edad_no_bcn <- edad[f_bcn=="no_bcn"]
> edad_bcn <- edad[f_bcn=="bcn"]
> d_no_bcn <- density(edad_no_bcn,from=30,to=70,n=41)
> d_bcn <- density(edad_bcn,from=30,to=70,n=41)
```

La función `density`

Esta función se puede limitar a un rango y a un número de puntos determinado mediante la opción `n`.

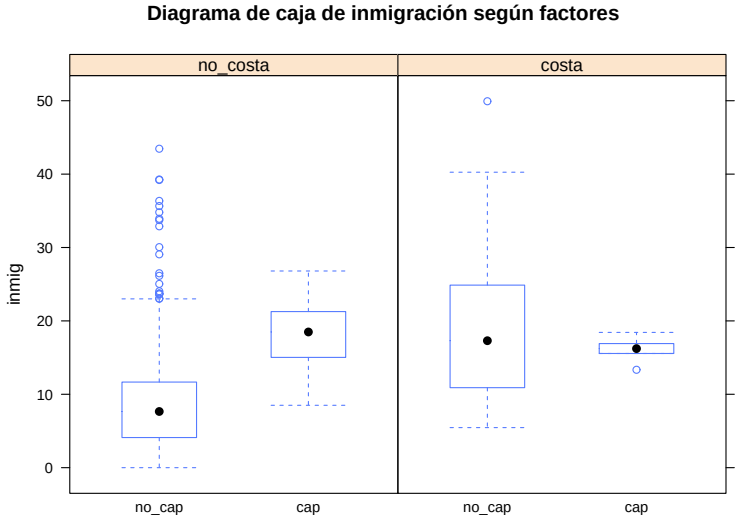
```
> plot(c(0,41),c(0,0.15),type="n",xaxt="n",xlab="Edad media",
+ ylab="Frecuencia")
> lines(d_no_bcn$y,type="l",col="blue",lwd=5)
> lines(d_bcn$y,type="l",col="red",lwd=5)
> lab <- 1+10*0:4
> axis(1,at=lab,labels=10*3:7)
> legend("topright",c("No BCN","BCN"),col=c("blue","red"),
+ lty=1,lwd=5,cex=1.5)
```



3.4.2. La librería *Lattice*

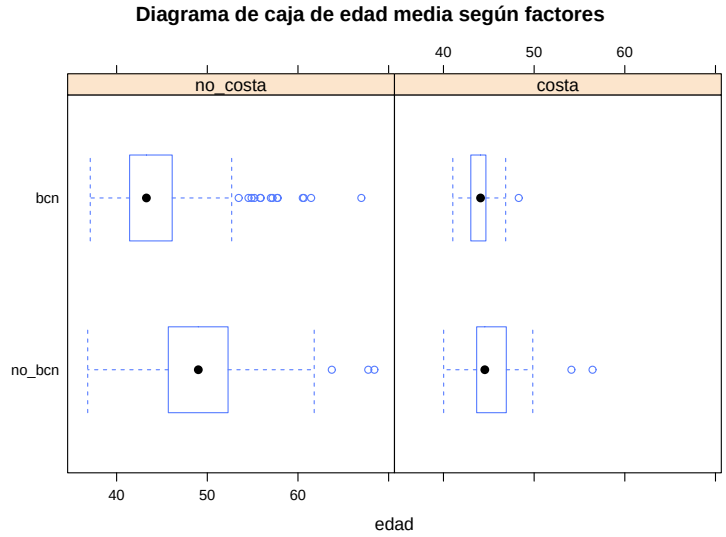
La librería `Lattice` es un sistema de visualización de datos de alto nivel. Es potente y elegante, con un énfasis en los datos multivariantes, como es el caso de este estudio. La primera función de esta librería es `bwplot` que permite representar un diagrama de caja para diferentes niveles de factores. Esto es, especificando `inmig ~ f_cap|f_cos` obtenemos el diagrama para la variable `inmig` según los factores capital y costa.

```
> bwplot(inmig ~ f_cap | f_cos, main="Diagrama de caja
+ de inmigración según factores")
```



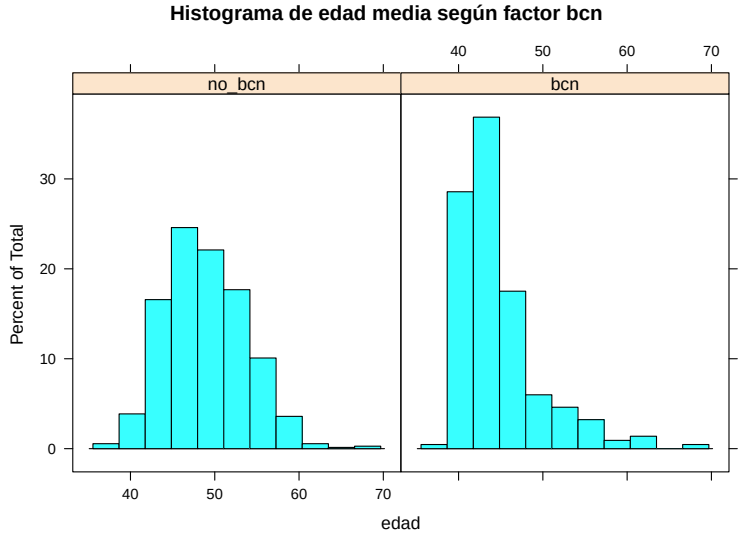
Si cambiamos el orden de los factores (`f_bcn ~ edad|f_cos`), el orden del gráfico también cambia.

```
> bwplot(f_bcn ~ edad | f_cos, main="Diagrama de caja
+ de edad media según factores")
```



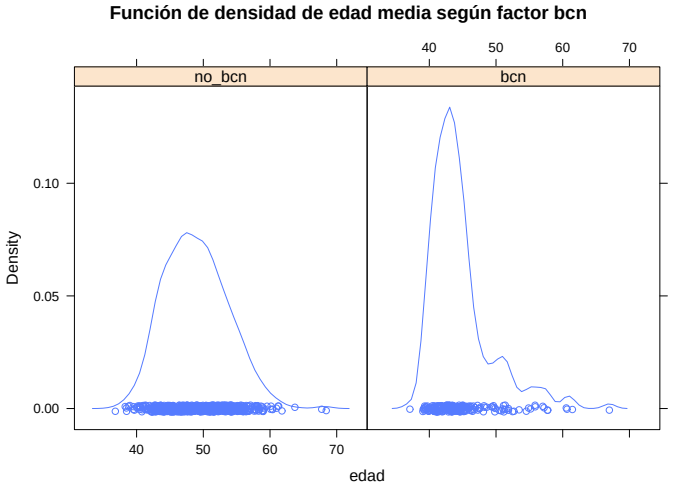
En el siguiente ejemplo, dibujaremos la función de densidad de la variable `edad` dividiéndola en dos subgrupos a partir de la variable `f_bcn`. Esto es, para los municipios que pertenecen al área metropolitana de Barcelona (`f_bcn=1`) y para los que no (`f_bcn=0`).

```
> histogram(~ edad | f_bcn, main="Histograma de
+ edad media según factor bcn")
```



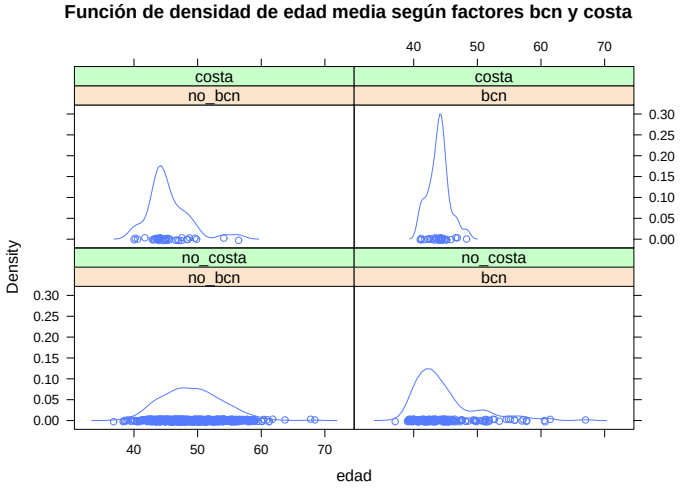
Un gráfico similar al anterior es el siguiente, pero en lugar de calcular el histograma de la variable calcula la función de densidad empírica, además de situar las observaciones en el eje de abscisas.

```
> densityplot(~ edad | f_bcn, main="Función de densidad
+ de edad media según factor bcn")
```



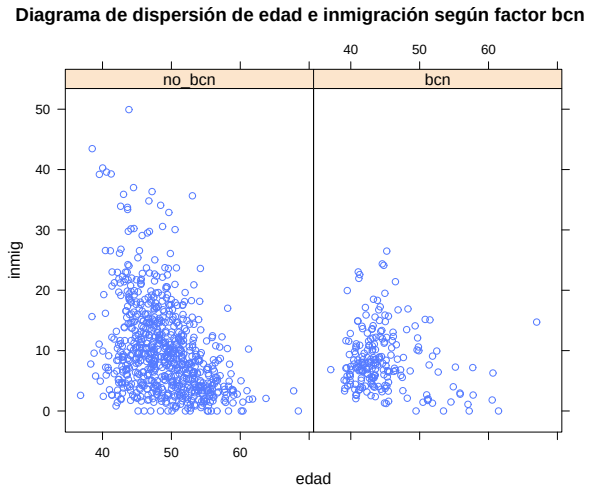
El gráfico anterior lo podemos complementar añadiendo otro factor. Si introducimos `f_bcn*f_cos`, estamos de hecho cruzando dos factores: la pertenencia al área metropolitana de Barcelona y la localización costera del municipio, de manera que el resultado serán cuatro gráficos de la función de densidad empírica de `edad` con las cuatro posibles combinaciones de `f_bcn` y `f_cos`.

```
> densityplot(~ edad | f_bcn*f_cos, main="Función de
+ densidad de edad media según factores bcn y costa")
```



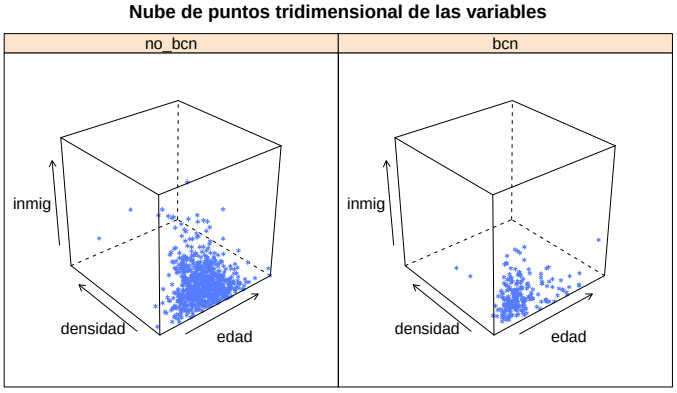
Similarmente a los gráficos anteriores, el diagrama de dispersión de una variable según los valores discretos de un factor se realiza mediante la función `xyplot`. Por ejemplo, si queremos obtener dos diagramas de dispersión de las variables *inmigración* y edad, uno para valores `f_bcn=1` y otro para valores `f_bcn=0`, bastará con introducir las siguientes instrucciones:

```
> xyplot(inmig ~ edad | f_bcn, main="Diagrama de dispersión
+ de edad e inmigración según factor bcn")
```



Un equivalente tridimensional del diagrama de dispersión es la nube de puntos, consistente en un conjunto de vértices en un sistema de coordenadas tridimensional. En estos se representa el valor de cada observación como un punto referenciado mediante tres ejes (X,Y y Z), correspondiente a tres variables. Veamos cómo calcular la nube de puntos de las variables *inmigración*, *edad* y *densidad* según la pertenencia o no al área metropolitana de Barcelona:

```
> cloud(inmig ~ edad*densidad | f_bcn, main="Nube de puntos
+ tridimensional de las variables")
```



Por último, veamos una estructura que resulta muy útil a la hora de analizar las relaciones existentes entre diversas variables. Esta estructura y su interpretación se asemejan a la de una matriz de correlaciones, pero en lugar de valores numéricos, cada elemento es un diagrama de dispersión.

Veámoslo en un ejemplo práctico: para las variables *inmigración*, *edad* y *densidad* tendremos una estructura 3×3, donde los elementos de la diagonal están vacíos. Fijémonos en el hecho de que los diagramas de la diagonal superior son iguales a los de la diagonal inferior invertidos.

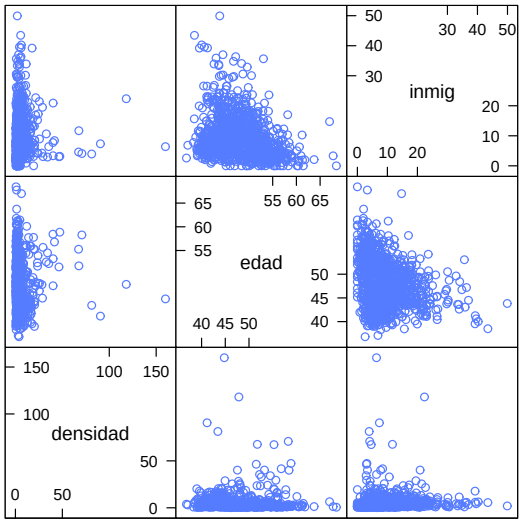
Recordemos que el diagrama de dispersión de una variable consigo misma daría como resultado una serie de puntos alineados perfectamente sobre la diagonal $x = y$.

```
> splom(demo, main="Diagrama de dispersión entre diversas
+ variables")
```

La matriz demo

Recordemos que, en este módulo, hemos creado esta matriz, que incluye tres variables relativas al porcentaje de inmigración, media de edad y densidad de los municipios de la base de datos.

Diagrama de dispersión entre diversas variables



Scatter Plot Matrix

Bibliografía

Gibernans Bàguena, J.; Gil Estallo, À. J.; Rovira Escofet, C. (2009). *Estadística*. Barcelona: Material didáctico UOC.