Fundamentos de la Ciencia de Datos

Práctica 1

Grado en Ingeniería Informática
Universidad de Alcalá

Pablo García García
Abel López Martínez
Álvaro Jesús Martínez Parra
Raúl Moratilla Núñez

14 de noviembre de 2023

Índice general

Índice de figuras

Introducción

El lenguaje R

El lenguaje R, es un software de uso gratuito comúnmente usado en tareas relacionadas con la estadística, como el análisis o visualización de datos; o en general la propia Ciencia de Datos. Para ello cuenta con una gran cantidad de paquetes y herramientas que facilitan el trabajo.

Figura 1: Logo del lenguaje R

El CRAN (Comprehensive R Archive Network, `https://cran.r-project.org/`) es un repositorio de recursos en línea que se utiliza para facilitar la distribución, el intercambio y el acceso a una amplia gama de software y paquetes relacionados con el lenguaje de programación R. La página web de CRAN sirve como el portal central para acceder a estos recursos y ofrece una variedad de apartados y enlaces útiles para los usuarios de R. A continuación, proporcionamos una descripción de los distintos enlaces a los que se puede acceder desde la página principal del CRAN:

- Mirrors: Esta sección permite a los usuarios seleccionar un espejo (mirror) cercano para descargar paquetes y recursos. Los espejos son servidores que almacenan copias de los paquetes y datos de CRAN, lo que ayuda a mejorar la velocidad de descarga y la disponibilidad de los recursos.
- What's new?: En esta sección, los usuarios pueden encontrar información sobre las últimas actualizaciones y novedades en el mundo de R y los paquetes disponibles en CRAN. Esto es útil para estar al tanto de las últimas características y mejoras.
- Search: El enlace "Search" permite a los usuarios buscar paquetes y recursos específicos en el repositorio de CRAN. Además, se puede utilizar la función de búsqueda avanzada del motor de búsqueda de Google.

- CRAN Team: Aquí se puede encontrar información sobre las personas y equipos que trabajan en el mantenimiento y desarrollo de CRAN. Es útil para conocer a las personas detrás de esta valiosa fuente de recursos.

- About R: Esta sección proporciona información sobre el lenguaje de programación R en general. Incluye enlaces a la página de inicio de R y a "The R Journal", una publicación académica relacionada con R.

- Software: Esta sección ofrece acceso a diversas fuentes y binarios relacionados con R, lo que permite a los usuarios descargar e instalar R en su sistema. También proporciona acceso a paquetes, Task Views y otros recursos.

- Documentation: Aquí los usuarios pueden encontrar documentación esencial relacionada con R. Esto incluye manuales, preguntas frecuentes (FAQs) y contribuciones de la comunidad para ayudar a los usuarios a comprender y utilizar R de manera efectiva.

En R, los paquetes son extensiones de software que contienen funciones, datos y documentación para realizar tareas específicas. Antes de utilizar un paquete, debes instalarlo y cargarlo en tu sesión de R. Algunas de las funciones más útiles para preparar los paquetes de un proyecto son:

- **Paquetes por defecto**:
  Mediante `getOption("defaultPackages")` se muestra una lista de los paquetes que se cargan automáticamente cuando inicias una sesión de R. Son los paquetes básicos que R carga por defecto. Para cambiar la lista de archivos que R carga por defecto podemos acceder a la siguiente ubicación (instalación de R por defecto):
  `C:/Program Files/R/R-4.3.1/library/base/R/RProfile`, y modificar el archivo como se observa en el Código 1, añadiendo al vector `dp` los paquetes que deseemos.

```
46 local({dp <- Sys.getenv("R_DEFAULT_PACKAGES")
47        if(identical(dp, "")) ## it fact methods is done first
48            dp <- c("datasets", "utils", "grDevices", "graphics",
49                    "stats", "methods", "knitr", "arules")
50        else if(identical(dp, "NULL")) dp <- character(0)
51        else dp <- strsplit(dp, ",")[[1]]
52        dp <- sub("[[:blank:]]*([[:alnum:]]+)", "\\1", dp) # strip
              whitespace
53        options(defaultPackages = dp)
54    })
```

Código 1: Modificación en fichero `Rprofile`

- **Instalación de paquetes**:
  La instalación de paquetes puede ser realizada de tres formas distintas:

  1. `install.packages("nombre_del_paquete")`
     A esta función se le debe pasar por parámetro el nombre del paquete que se desea instalar.

2. `install.packages(übicacion_del_paquete", rep=NULL)`
   A la función también se le puede pasar por parámetros la ubicación del archivo, que recomendablemente debe estar en una carpeta temporal en "c:/", este archivo lo descargamos desde:
   `https://cran.r-project.org/` > `Packages` > `Table of available packages, sorted by name` > Elegimos el paquete y descargamos la versión `r-release` de la sección `Windows binaries`.
3. `utils:::menuInstallPkgs()`
   Tras la ejecución de este comando aparecerá una ventana donde se podrá elegir el mirror desde el que se va a descargar el paquete, y tras elegir el mirror (Spain (Madrid) en nuestro caso), aparece otra ventana donde se puede elegir el paquete que se quiere instalar, tras hacer doble click, este se instalará automáticamente.

- **Información de un paquete**:
  Cuando ejecutas `library(help="nombre_del_paquete")`, R te mostrará información detallada sobre el paquete especificado. Esto incluye una descripción del paquete y una lista de las funciones que contiene, junto con sus descripciones.

- **Carga de paquetes**:
  Si ejecutas `library(nombre_del_paquete)` con el nombre de un paquete, R cargará el paquete en tu sesión para que puedas utilizar sus funciones y objetos.

- **Lista de paquetes instalados**:
  Al ejecutar `library()` sin argumentos, R te mostrará una lista de los paquetes que están actualmente cargados en tu sesión de R. Esto te permite verificar qué paquetes están disponibles para su uso.

- **Lista de paquetes cargados**:
  Mediante `search()` podemos ver un listado completo de los paquetes actualmente cargados en memoria.

A parte, es recomendable descargar y conocer a conciencia el manual y las viñetas de todos los paquetes que usemos en nuestros proyectos (disponible en la página web del CRAN).

El lenguaje LaTeX

Para la realización de esta práctica, se empleará el concepto de **programación literaria**, que consiste en crear un documento en el que se combine texto con código, de manera que este se pueda explicar y entender de una manera mucho más sencilla. Una forma de realizar esto con código R, es el uso del lenguaje LaTeX, que es un sistema de composición de documentos enfocado al ámbito científico. Es algo similar a un lenguaje de marcas con el que poder definir la estructura de un documento, pero cuenta con la particularidad de que es un lenguaje Turing–completo, por lo que cualquier algoritmo puede ser implementado dando una mayor flexibilidad, aunque no sea su objetivo principal. Veremos ahora los pasos seguidos para su instalación. Para poder trabajar, lo mínimo que necesitaremos es un

compilador de LaTeX, en este caso se ha optado por la distribución MiKTeX que lo incluye, ya que estamos trabajando en Windows. Además, para una mayor comodidad trabajando con el código, se ha optado por el IDE TeXstudio, uno de los más conocidos en la comunidad.

Una vez hemos tratado ambos lenguajes, necesitamos entender con qué tipos de extensiones se suelen trabajar para ver el proceso de integración con R (sin entrar en profundidad). Estas dependen de cómo queremos almacenar nuestro documento, o cómo están almacenadas dependencias de estos, como por ejemplo, imágenes. Esta tarea se realiza usando un compilador u otro.

Para ello nos fijaremos en la Figura 2. Por ahora nos quedaremos con las extensiones que trabajaremos más a menudo, que serán `.Rnw`, `.tex`, y `.pdf`. La primera de ellas representan los archivos que tienen código LaTeX y R "mezclado", la segunda aquellos que contienen código LaTeX puro, y la última nuestro documento final.

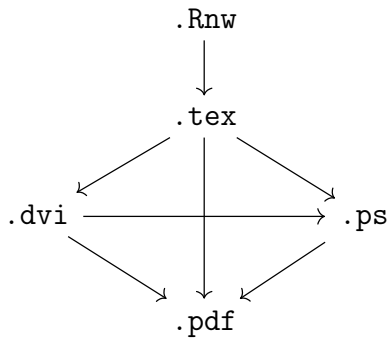

Figura 2: Esquema de extensiones en LaTeX

Existen dos herramientas que nos permiten trabajar con archivos `.Rnw`, estas son Sweave y Knitr. A pesar de que en la asignatura ha sido propuesta la primera de ellas, optaremos por la segunda, pues existieron diversos errores al compilar archivos con esta, y al ser más antigua, los documentos finales tenían menos calidad. Knitr nos ofrece mayor calidad y un mejor formato en el código fuente R mostrado. Para instalarla basta con escribir `install.packages('knitr')` en una consola de R. Sweave viene ya por defecto con `utils`.

Por último, se explicará cómo hemos agilizado el proceso de trabajo con Knitr y TeXStudio. Lo primero será hacer que R cargue por defecto Knitr, para ello modificaremos el archivo `Rprofile` en `libray/base/R` dentro de la carpeta de instalación de R, añadiendo `knitr` al resto de paquetes que carga por defecto. Una vez hecho esto, iremos a la configuración de TeXStudio, y aquí a *Compilar*. En la zona *Órdenes de usuario*, crearemos una nueva tal y como se ve en la Figura 3, de manera que le digamos dónde están los binarios de R, para que pueda crear un fichero `.tex`, y posteriormente invocar a nuestro compilador, para obtener nuestro documento en `.pdf`. TeXStudio se encargará de reemplazar el símbolo % por el nombre del archivo que le ordenamos compilar.

Figura 3: Creación de orden de usuario

Ahora basta modificar el botón verde del IDE para que en vez de invocar al compilador pdfLaTeX, lleve a cabo la instrucción que le hemos dado. Para ello volveremos al menú de compilación en el que nos ubicábamos previamente, y observaremos la sección de *Meta-Órdenes*. Modificaremos el valor del campo *Compilador por defecto*, escribiendo `txs:///knitr` para que se ejecute la orden que previamente hemos creado, o podemos hacerlo de manera gráfica como se observa en la Figura 4. Ahora bastará pulsar el botón verde o F5 para ver a nuestra izquierda el código de nuestro documento, y a la derecha actualizado, el documento PDF final.

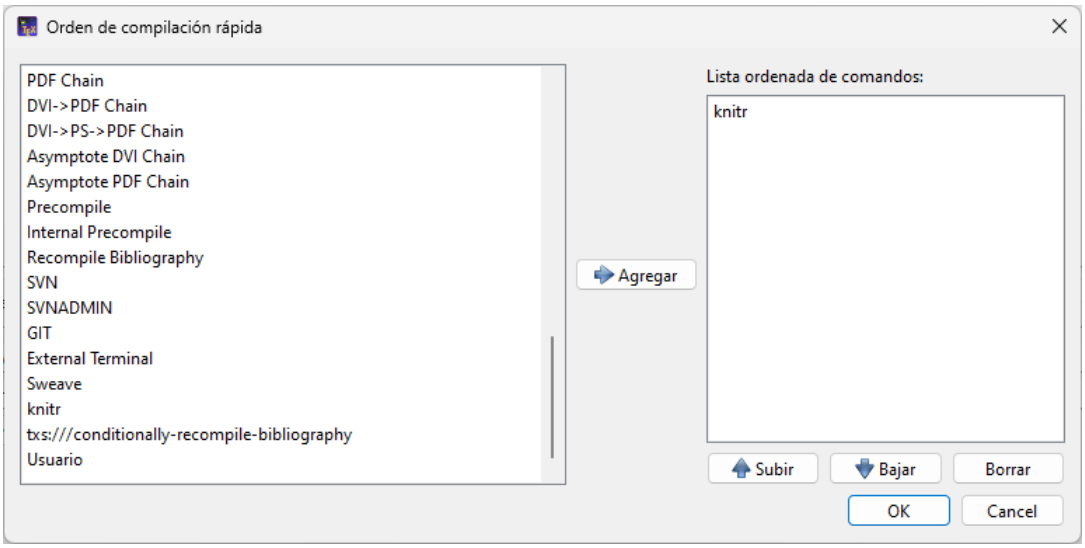

Figura 4: Modificación de compilación

Por último, para llevar un mejor control de versiones del proyecto, y de coordinación entre los miembros del grupo, se usará un repositorio de GitHub. Añadiremos un archivo `.gitignore` para no cargar en el repositorio los archivos temporales generados durante la compilación. Otra alternativa que se podría haber usado, es usar Overleaf (aquí usaríamos la extensión `.Rtex` en vez de `.Rnw`), ya que no sería necesaria la instalación de ningún software, y también trabaja con Knitr. Sin embargo, la integración de GitHub en Overleaf es de pago, por lo que optamos por usar la configuración explicada hasta el momento, para poder tener un mejor control de versiones sin coste alguno.

Por último, mencionar que al igual que R posee su repositorio de paquetes (que ya hemos visto que incluye más cosas) llamado CRAN, LaTeX que en realidad es "un subconjunto" del lenguaje TeX, también tiene su propio portal llamado CTAN o Comprehensive TeX Archive Network (`https://www.ctan.org/`) de donde se descargan los paquetes y otros materiales para el lenguaje.

Figura 5: Mascota de TeX y el CTAN

Parte 1

Ejercicios guiados

En esta primera parte de esta práctica, repetirán los ejercicios explicados y realizados por el profesor en las clases de laboratorio, utilizando los mismos procedimientos vistos plasmándolos en este documento.

1.1. Descripción de los datos

"*El primer conjunto de datos, que se empleará para realizar el análisis de descripción de datos, estará formado por datos de una característica cualitativa, nombre, y otra cuantitativa, radio, de los satélites menores de Urano, es decir, aquellos que tienen un radio menor de 50 Km, dichos datos, los primeros cualitativos nominales, y los segundos cuantitativos continuos, son: (Nombre, radio en Km): Cordelia, 13; Ofelia, 16; Bianca, 22; Crésida, 33; Desdémona, 29; Julieta, 42; Rosalinda, 27; Belinda, 34; Luna-1986U10, 20; Calíbano, 30; Luna-999U1, 20; Luna 1999U2, 15.*"

Para comenzar con la resolución de este ejercicio, deberemos escribir los datos en un fichero `.txt`, cumpliendo las siguientes normas:

- Existirá una tabulación entre dato y dato.
- La primera columna numera las filas, y en la primera fila se introduce un espacio y el nombre de las variables.
- Se introducirá un salto de línea en la última fila
- Para los números decimales se utilizarán puntos.
- Al escribir nombres, no se deberán introducir espacios.

Obedeciendo a estas normas, copiamos los datos en un fichero llamado `satelites.txt`, y lo cargamos en R de la siguiente manera:

```
s <- read.table("data/satelites.txt")
print(s)

##          nombre radio
## 1      Cordelia    13
## 2        Ofelia    16
## 3        Bianca    22
## 4       Crésida    33
## 5     Desdémona    29
## 6       Julieta    42
## 7     Rosalinda    27
## 8       Belinda    34
## 9  Luna-1986U10    20
## 10     Calíbano    30
## 11   Luna-999U1    20
## 12  Luna-1999U2    15
```

Ahora en la variable `s` tenemos un dataframe con los datos de nuestros satélites. En los dataframes se accede por `[fila, columna]`, y también podemos consultar las dimensiones con la función `dim`. Sería de esperar que nos dijera que tiene 12 filas (los 12 datos), y 2 columnas (`nombre` y `radio`).

```
dim(s)

## [1] 12  2
```

También podemos ordenar el dataframe, en función de una de las magnitudes (columnas), usando la función `order` aplicando recursivamente el concepto de acceder por filas y columnas. Veamos un ejemplo, si en `s` teníamos guardado nuestro dataframe, y queremos ordenar por `radio`, la manera de hacerlo sería la siguiente:

```
s_ordered <- s[order(s$radio), ]
print(s_ordered)

##          nombre radio
## 1      Cordelia    13
## 12  Luna-1999U2    15
## 2        Ofelia    16
## 9  Luna-1986U10    20
## 11   Luna-999U1    20
## 3        Bianca    22
## 7     Rosalinda    27
## 5     Desdémona    29
## 10     Calíbano    30
## 4       Crésida    33
```

```
## 8       Belinda    34
## 6       Julieta    42
```

Podemos introducir nuevos criterios a la ordenación, como por ejemplo, hacerlo en orden descendente. Para esto usaremos la función `rev`.

```
s_ordered_rev <- s[rev(order(s$radio)), ]
print(s_ordered_rev)

##          nombre radio
## 6       Julieta    42
## 8       Belinda    34
## 4       Crésida    33
## 10     Calíbano    30
## 5     Desdémona    29
## 7     Rosalinda    27
## 3        Bianca    22
## 11   Luna-999U1    20
## 9  Luna-1986U10    20
## 2        Ofelia    16
## 12  Luna-1999U2    15
## 1      Cordelia    13
```

También suele ser útil conocer cuántos elementos tiene una columna. Podemos averiguarlo con la función `length`, veamos un ejemplo.

```
length(s$radio)

## [1] 12
```

Otro valor que nos podemos plantear calcular es el rango. Para ello podemos usar las funciones `max` y `min`. Debemos tener cuidado con la función `range` y no confundirnos, pues nos dará los valores máximo y mínimo.

```
r <- max(s$radio) - min(s$radio)
print(r)

## [1] 29

range(s$radio)

## [1] 13 42
```

Para una mejor lectura, podemos cambiar la forma de obtener la columna de los radios:

```
radio <- s$radio
```

La idea de calcular la diferencia de el máximo y el mínimo a mano parece funcionar, sin embargo, para futuros casos sería más ágil tener codificada una función como la siguiente.

```
rango <- function(radio){max(radio) - min(radio)}
rango(radio)

## [1] 29
```

Sin embargo, al salir de R, la definición de la función se pierde, por lo que deberemos guardarla en un fichero, y posteriormente cargarlo en futuras ejecuciones. Lo haremos de la siguiente manera:

```
dump("rango", file = "fn/rango.R")
source("fn/rango.R")
```

Volviendo al estudio de nuestros datos, veamos cómo calcular las diferentes frecuencias. Como en R no existe una función para las frecuencias relativas, se definirá y guardará una propia.

```
fabs_radio <- table(radio)
fabsacum_radio <- cumsum(fabs_radio)
frecrel <- function(r){table(r)/length(r)}
dump("frecrel", file = "fn/frecrel.R")

print(fabs_radio)

## radio
## 13 15 16 20 22 27 29 30 33 34 42
##  1  1  1  2  1  1  1  1  1  1  1

print(fabsacum_radio)

## 13 15 16 20 22 27 29 30 33 34 42
##  1  2  3  5  6  7  8  9 10 11 12

print(frecrel(radio))

## r
##         13         15         16         20         22         27         29
## 0.08333333 0.08333333 0.08333333 0.16666667 0.08333333 0.08333333 0.08333333
##         30         33         34         42
## 0.08333333 0.08333333 0.08333333 0.08333333
```

Otro valor que podemos calcular es la media aritmética de los datos, para ello se cuenta con la función `mean`.

```
mr <- mean(radio)
print(mr)

## [1] 25.08333
```

Ahora calcularemos la desviación típica, para ello se cuenta con la función `sd`.

```
sdr <- sd(radio)
print(sdr)

## [1] 8.857029
```

Sin embargo, el resultado obtenido no es el esperado. Esto se debe a que esta función realiza el siguiente cálculo

$$\sigma = \sqrt{\frac{\sum_{i=0}^{n}(x_i - \bar{x})^2}{n-1}}$$

que es más utilizado en inferencia estadística, porque hace que se parezca más a una campana de Gauss (menos sesgo), mientras que la fórmula vista en clase utiliza un factor de n en vez de $n-1$ en el denominador (dentro de la raíz). Para ello, el profesor lo corrigió de la siguiente manera:

```
sdr2 <- sqrt((sdr^2)*(length(radio)-1)/length(radio))
print(sdr2)

## [1] 8.47996
```

En realidad lo que se está realizando es el siguiente "ajuste":

$$\sigma' = \sqrt{\sigma^2 \cdot \frac{n-1}{n}}$$

Una vez hemos visto cómo se calcula la desviación típica, podremos ver cómo calcular la varianza. Como sabemos que es el cuadrado de la desviación típica, bastaría con elevar al cuadrado si no fuera por el "fallo" de $n-1$ visto previamente. En este caso, el profesor lo arregló para el caso particular de la siguiente manera:

```
varr <- var(radio)
varr <- 11/12 * varr
print(varr)

## [1] 71.90972
```

Otro de los valores que se ha enseñado cómo calcular, es la mediana. Para este caso existe la función `median`.

```
medianr <- median(radio)
print(medianr)

## [1] 24.5
```

En último lugar, el profesor enseñó cómo calcular cuantiles, y para ello mostró la función `quantile`, pero se mencionó que se obtienen resultados diferentes a los esperados debido a la forma que tiene de calcularlos, y se deberá programar. Aquí se muestra un ejemplo de cómo se calcularía el primer cuartil.

```
cuar1 <- quantile(radio, 0.25)
print(cuar1)

## 25%
##  19
```

Como añadido, el profesor explicó cómo abrir un ejemplo de Sweave, cómo pasarlo a un fichero que LaTeX pudiese leer, y cómo compilarlo a PDF. Las instrucciones son las que se verán a continuación, aunque como ya se explicó en la introducción, usaremos otra forma de trabajar con estos archivos a lo largo de la práctica.

```
rnwfile<-system.file("Sweave", "example-1.Rnw", package="utils")
Sweave(rnwfile)
tools::texi2pdf("example-1.tex")
```

1.2. Asociación

"*El segundo conjunto de datos, que se empleará para realizar el análisis de asociación, estará formado por las siguientes 6 cestas de la compra:* {*Pan, Agua, Leche, Naranjas*}, {*Pan, Agua, Café, Leche*}, {*Pan, Agua, Leche*}, {*Pan, Café, Leche*}, {*Pan, Agua*}, {*Leche*}."

1.3. Detección de datos anómalos

1.3.1. Primer ejercicio

"*El tercer conjunto de datos, que se empleará para realizar el análisis de detección de datos anómalos utilizando técnicas con base estadística, estará formado por los siguientes 7 valores de resistencia y densidad para diferentes tipos de hormigón* {*Resistencia, Densidad*}*:* {*3, 2; 3.5, 12; 4.7, 4.1; 5.2, 4.9; 7.1, 6.1; 6.2, 5.2; 14, 5.3*}*. Aplicar las medidas de ordenación a la resistencia y las de dispersión a la densidad.*"

Lo primero explicado por el profesor fue la preparación de los datos proporcionados para que `arules` fuese capaz de tratar con ellos. Para ello se introduce una matriz de ceros y unos mediante el paquete `Matrix`, que indique en cada suceso, qué elementos contiene. La matriz es la siguiente:

$$\begin{pmatrix} 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Además, deberemos indicar las dimensiones de esta matriz (6×5), que estamos introduciendo los datos por filas (`byrow=TRUE`), y con `dimnames` ponemos los nombres a las filas y las columnas. El código es el siguiente.

```
muestra <- Matrix(c(1, 1, 0, 1, 1,
1, 1, 1, 1, 0,
1, 1, 0, 1, 0,
1, 0, 1, 1, 0,
1, 1, 0, 0, 0,
0, 0, 0, 1, 0), 6, 5, byrow = TRUE, dimnames = list(
c("suceso1", "suceso2", "suceso3", "suceso4", "suceso5", "suceso6"),
c("Pan", "Agua", "Café", "Leche", "Naranjas")), sparse=TRUE)
muestra

## 6 x 5 sparse Matrix of class "dgCMatrix"
##         Pan Agua Café Leche Naranjas
## suceso1   1    1    .     1        1
## suceso2   1    1    1     1        .
## suceso3   1    1    .     1        .
## suceso4   1    .    1     1        .
## suceso5   1    1    .     .        .
## suceso6   .    .    .     1        .
```

A continuación, se ha enseñado cómo mostrar la matriz con puntos y barras, en vez de con unos y ceros. Se consigue con la función `as` y el parámetro `nsparseMatrix`.

```
muestrangCMatrix <- as(muestra, "nsparseMatrix")
muestrangCMatrix

## 6 x 5 sparse Matrix of class "ngCMatrix"
##         Pan Agua Café Leche Naranjas
## suceso1   |    |    .     |        |
## suceso2   |    |    |     |        .
## suceso3   |    |    .     |        .
## suceso4   |    .    |     |        .
```

```
## suceso5   |    |    .     .        .
## suceso6   .    .    .     |        .
```

Sin embargo, para el algoritmo debemos pasarle justo la transpuesta de la matriz con la que trabajamos, por ello se utiliza la función `t`.

```
transpmuestrangCMatrix <- t(muestrangCMatrix)
transpmuestrangCMatrix

## 5 x 6 sparse Matrix of class "ngCMatrix"
##          suceso1 suceso2 suceso3 suceso4 suceso5 suceso6
## Pan            |       |       |       |       .
## Agua           |       |       |       .       .
## Café           .       |       .       |       .
## Leche          |       |       |       |       .       |
## Naranjas       |       .       .       .       .       .
```

Podemos consultar algunos datos acerca de los datos de nuestra matriz podemos usar la función `as` con el parámetro `transactions`.

1.3.2. Segundo ejercicio

"*El cuarto conjunto de datos, que se empleará para realizar el análisis de detección de datos anómalos utilizando técnicas basadas en la proximidad y en la densidad, estará formado por las siguientes 5 calificaciones de estudiantes: 1.* {*4, 4*}*; 2.* {*4, 3*}*; 3.* {*5, 5*}*; 4.* {*1, 1*}*; 5.* {*5, 4*} *donde las características de las calificaciones son: (Teoría, Laboratorio).*"

Parte 2

Ejercicios autónomos

2.1. Descripción de los datos

"*El primer conjunto de datos, que se empleará para realizar el análisis de descripción de datos, estará formado por datos de una característica cuantitativa, distancia, desde el domicilio de cada estudiantes hasta la Universidad, dichos datos, cuantitativos continuos, son: 16.5, 34.8, 20.7, 6.2, 4.4, 3.4, 24, 24, 32, 30, 33, 27, 15, 9.4, 2.1, 34, 24, 12, 4.4, 28, 31.4, 21.6, 3.1, 4.5, 5.1, 4, 3.2, 25, 4.5, 20, 34, 12, 12, 12, 12, 5, 19, 30, 5.5, 38, 25, 3.7, 9, 30, 13, 30, 30, 26, 30, 30, 1, 26, 22, 10, 9.7, 11, 24.1, 33, 17.2, 27, 24, 27, 21, 28, 30, 4, 46, 29, 3.7, 2.7, 8.1, 19, 16.*"

2.2. Asociación

"*El segundo conjunto de datos, que se empleará para realizar el análisis de asociación, estará formado por las siguientes conjuntos de extras incluidos en 8 ventas de coches: {X, C, N, B}, {X, T, B, C}, {N, C, X}, {N, T, X, B}, {X, C, B}, {N}, {X, B, C}, {T, A}. Donde: {X: Faros de Xenon, A: Alarma, T: Techo Solar, N: Navegador, B: Bluetooth, C: Control de Velocidad}, son los extras que se pueden incluir en cada coche.*"

2.3. Detección de datos anómalos

2.3.1. Primer ejercicio

"*El tercer conjunto de datos, que se empleará para realizar el análisis de detección de datos anómalos utilizando técnicas con base estadística, estará formado por los siguientes 10 valores de velocidades de respuesta y temperaturas normalizadas de un microprocesador {Velocidad, Temperatura}: {10, 7.46; 8, 6.77; 13, 12.74; 9, 7.11; 11, 7.81; 14, 8.84; 6, 6.08; 4, 5.39; 12, 8.15; 7, 6.42; 5, 5.73}. Aplicar las medidas de ordenación a la velocidad y las de dispersión a la temperatura.*"

2.3.2. Segundo ejercicio

"*El cuarto conjunto de datos, que se empleará para realizar el análisis de detección de datos anómalos utilizando técnicas basadas en la proximidad y en la densidad, estará formado por el número de Mujeres y Hombres inscritos en una serie de cinco seminarios que se han impartido sobre biología. Los datos son: {Mujeres, Hombres}: 1. {9, 9}; 2. {9, 7}; 3. {11, 11}; 4. {2, 1}; 5. {11, 9}.*"