Fundamentos de la Ciencia de Datos

Prueba de Laboratorio 1 (PL1)

Jorge Revenga Martín de Vidales
Ángel Salgado Aldao
Adrián García

Grado en Ingeniería Informática
Universidad de Alcalá

12 de noviembre de 2023

Índice

1. Introducción - Consideraciones previas

1.1. Funciones básicas

Para utilizar una función en R se escribe el nombre de la función, seguido de los parámetros de entrada entre paréntesis e.g.: *función(parámetros)*

- Función *contributors()*: Muestra los creadores del programa (R)
- Función *help()*: Abre un HTML con información sobre la función *help()* o de la función entre paréntesis de haberla. Para todas las funciones que programemos (para todas las que existan) en R debe poder usarse la función *help()*.

  En el archivo HTML se distinguen varios elementos:

  - *"función {paquete}"*: la función de la que se obtiene información seguida del paquete al que pertenece.
  - Description: descripción de la función.
  - Usage: aparece la función y todos los argumentos que se le pueden introducir.
  - Arguments: Explicación de los argumentos o parámetros.
  - Details: Detalles adicionales de la función.
  - Offline help: Ayuda sin conexión.
  - Note: Nota del autor.
  - References: Referencias.
  - Examples: Ejemplos de uso de la función.
- Función *getwd()* se utiliza para obtener el directorio de trabajo actual (working directory)
- Función *Setwd("C:/...")* permite cambiar el nuevo directorio de trabajo en el que queramos trabajar
- *help.start()*: Manda a un compendio de todas las ayudas disponibles para trabajar con R
- Función *list.files()*: Muestra todos los archivos en el directorio. *dir()* hace lo mismo

1.2. Introducción de un archivo .txt en R

Para introducir una tabla desde un archivo *.txt* en R con seguridad de que vaya a cargar correctamente se deben seguir las siguientes reglas (pueden no ser todas necesarias en todos los casos)

- Debe haber una tabulación entre dato y dato, más tabulaciones no lo rompen del todo
- Debe haber una primera columna que numere las filas. Excepto en la primera fila, que sólo habrá una tabulación, seguida de los nombres de las variables separados por tabulaciones
- Hay que introducir un enter al final del documento
- Los números decimales deben ser introducidos con puntos, no con comas
- Los datos tienen que ir juntos, los espacios deben ser separados con guiones u otros símbolos

1.3. Uso de Sweave

Vamos a generar un archivo pdf con código R embedido en el mismo de forma automática, para esto haremos uso de Sweave

- *rnwfile¡-system.file("Sweave", "example-1.Rnw", package="utils")*: obtiene la ubicación del archivo "example-1.Rnw"que está incluido en el paquete ütilsz la almacena en la variable "rnwfile'
- *Sweave(rnwfile)*: Genera un conjunto de documentos nuevos, entre ellos un .txt, un .pdf y demás, a partir de la variable definida
- *tools::texi2pdf("example-1.tex")*: Genera el .tex de sweave en un pdf. No funciona sin un compilador de latex (como miktex) en la máquina. Al descargar latex se puede seleccionar una opción para que se realice la carga automáticamente

1.4. Paquetes por defecto al abrir R

- *getOption("defaultPackages")*: muestra los (6) paquetes por defecto que cargan al abrir R
- *library(help="base")*: Muestra las funciones del paquete "base"junto con una descripción básica, *library(help="utils")* muestra funciones útiles para trabajar con R. Estos son parte de los 23 paquetes instalados por defecto
- *library()*: Muestra los paquetes en la biblioteca (los instalados manualmente + los de la biblioteca estándar)
- *library(paquete)*: Carga el paquete, también podemos hacerlo si nos vamos a paquetes/cargar paquete

1.5. Instalar paquetes nuevos

- Nos vamos a paquetes/seleccionar el espejo CRAN
- Seleccionamos el repositorio que queremos utilizar y elegimos el paquete (También se puede instalar con install.packages("paquete"))
- También se puede instalar desde el dispositivo local, para ello hay que irse a la página oficial de CRAN.
  - Para aprender se usan las viñetas y luego el manual
  - Se descarga la versión r-release
  - Se crea un directorio temporal (temp) y se mete ahí el zip descargado
  - *install.packages("c:direccion_al_zip/tmp/nombre.zip",repos=NULL)*: lo instala
- Para instalar el paquete Arules vamos a descargarlo, también hay que descargar el manual y las viñetas, luego usar install.packages para las dependencias

2. Ejercicios con ayuda del profesor

Realización de cuatro ejercicios con ayuda del profesor en los que se van a realizar, utilizando el entorno R, un análisis de descripción de datos, un análisis de asociación y dos análisis de detección de datos anómalos, aplicando todos los conceptos teóricos vistos en cada lección.

2.1. Análisis de descripción de 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.

Solución:

- *s¡-read.table("satelites.txt ")*: Se asigna los valores de la tabla satelites (almacenada en el directorio de trabajo) a la variable "s". Al teclear introducir el nombre de la variable como comando se deberían mostrar los datos
- *dim(s)*: Devuelve las dimensiones de una matriz. En este caso *[1] 12 2* quiere decir 12 filas, 2 columnas
- *so=s[order(s$radio),]*: ordena las filas de un DataFrame 's' según los valores de la variable "Radio". Para hacerlo, se usa la función *order()* para obtener el orden de las filas en función de los valores de "Radio", y luego se aplica ese orden al DataFrame 's' utilizando los corchetes *[ ]*. Como resultado, so contendrá las filas de s ordenadas según los valores de "Radio".
- *so=s[rev(order(s$radio)),]*: Añadiendo *rev()* obtenemos las filas en orden decreciente

- *length(s$radio)*: Devuelve la longitud de la columna
- Para realizar el cálculo del rango:
  - *rangor=max(s$radio)-min(s$radio)* : siguiendo lo visto en teoría, el resultado es 29
  - *range(s$radio)*: devuelve el número menor y el mayor, no la diferencia. Queremos una función que calcule el rango, por lo que primero la creamos utilizando el comando *function()*.
  - Podemos definir la variable Radio=s$radio para ahorrar tiempo
  - *rango=function(Radio)max(Radio)-min(Radio)*: En los paréntesis, especificamos los argumentos que la función recibirá, y entre llaves, definimos las instrucciones que llevará a cabo.
  - *dump(rango", file="rango.R")*: empleamos la función dump para guardar nuestra función en el archivo "rango.R". Para utilizarla posteriormente, la cargaremos utilizando el comando source (*source(rango.R")*).
- *frecabsradio¡-table(s$radio)*: calcula la frecuencia absoluta para todos los valores
- *frecabscumradio¡-cumsum(s$radio)*: calcula la frecuencia absoluta acumulada para todos los valores
- *frecrel=function(radio)table(radio)/length(radio)*: no tenemos función para la frecuencia relativa, por lo que la definimos. La función se puede escribir de forma diferente, ya que en este caso 'radio' no es una referencia a una variable sino un parámetro de entrada de la función, por lo que *Frecrel¡-function(x)table(x)/length(x)* serviría de igual manera. *X=s$Radio* y *Frecrelradio=frecrel(x)* calcula usando la función definida
- *Frecrelacumradio¡-cumsum(frecrelradio)*: no tenemos función para la frecuencia relativa acumulada tampoco, se puede calcular usando *cumsum()* de la frecuencia relativa
- *mr¡-mean(radio)*: Calcula la media
- *sdr¡-sd(radio)*: Para la desviación estándar se usa la función sd, el problema yace en que esta función realiza la desviación típica muestral (la cual divide entre N-1 y no entre N) para realizar inferencias probabilísticas. Hay casos como es el de este ejercicio que tenemos todos los datos (la población entera) y nos interesa usar la desviación estándar poblacional
- *sdr=sqrt((sdr$\hat{2}$)*11/12))*: creamos la función, quitándole la raíz cuadrada con el $\hat{2}$ para multiplicar después por N-1/N, para después añadirle la raiz cuadrada de nuevo. Este código puede ser fácilmente optimizado y definido en una función para su reutilización
- *varr=var(radio)*: Calcula la varianza (en este caso se trata también de la varianza muestral)
- *varr=varr*11/12*: Calculamos la varianza poblacional de la misma forma que la desviación típica, sin tener que preocuparnos por la raíz cuadrada
- *Medianr¡-median(s$Radio)*: Para la mediana usamos la función *median()*, pero esta función usa distribuciones de probabilidad en vez de usar ecuaciones y muchas veces no dará el resultado correcto

- *cuart1=quantile(radio, 0.25)*: Calcula el primer cuartil, puede usarse para calcular cualquier centil (0. %). Le ocurre el mismo problema que a la mediana

2.2. Análisis de 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.

Solución

- *library(arules)*: Carga el paquete arules
- *search()*: Muestra los paquetes cargados
- *library(Matrix)*: Carga el paquete Matrix
- *muestra¡-Matrix(c(1,1,0,1,1,1,1,1,1,1,0,1,1,0,1,0,1,0,1,0,1,1,0,1,1,0,0,0,0,0,0,1,0),6,5,byrow=TRUE,dimn c("Pan","Agua","Café","Leche","Naranjas")),sparse=TRUE)*: Carga los datos del problema
- *muestrangCMatrix¡-as(muestra,"nsparseMatrix")*: utilizamos la función as para convertir el objeto muestra a una representación de matriz dispersa (sparse matrix)
- *traspmuestrangCMatrix¡-t(muestrangCMatrix)*: Trasponemos y tenemos la matriz como arules nos la pide
- *transacciones¡-as(traspmuestrangCMatrix,"transactions")*
- *summary(transacciones)*: Muestra más info
- *asociaciones¡-apriori(transacciones, parameter=list(support=0.5,confidence=0.8)*: Aplica el algoritmo apriori con umbral de soporte de 0.5 y de confianza de 0.8
- *inspect(asociaciones)*: Muestra las asociaciones que pasan el algoritmo

2.3. Análisis de detección de datos anómalos

Esto es un poco más de texto en el párrafo.

2.3.1. Técnicas con base estadística

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.

Caja y Bigotes

- *(muestra=t(matrix(c(3,2,3.5,12,4.7,4.1,5.2,4.9,7.1,6.1,6.2,5.2,14,5.3),2,7,dimnames=list(c("r","d")*
- *(muestra=data.frame(muestra))*:
- *(boxplot(muestra$r,range=1.5,plot=FALSE))*: Boxplot de forma predeterminada muestra gráficamente la resolución con caja y bigotes, por lo que se añade *plot=FALSE* para que no lo haga
- *(cuar1r¡-quantile(muestra$r, 0.25))*: Cálculo del primer cuartil
- *(cuar3r¡-quantile(muestra$r, 0.75))* Cálculo del tercer cuartil
- *(int=c(cuar1r-1.5*(cuar3r-cuar1r),cuar3r+1.5*(cuar3r-cuar1r)))*: calcula el rango
- *for(i in 1:length(muestra$r)) if(muestra$r[i]¡int[1] —— muestra$r[i]¿int[2])print("el suceso"); print(i); print("es un suceso anómalo o outlier")*

Desviación típica

- *sdd=sqrt(var(muestra$d)*length(muestra$d)-1)/length(muestra$d)*
- *(intdes=c(mean(muestra$d)-2*sdd,mean(muestra$d)+2*sdd))*:
- *for(i in 1:length(muestra$d)) if (muestra$d[i]¡intdes[1] —— muestra$d[i]¿intdes[2])print("el suceso");print(i);print(muestra$d[i]);print("es un suceso anómalo o outlier")*

2.3.2. Técnicas basadas en la proximidad y en la densidad

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).

Vecinos próximos

- *(muestra=matrix(c(4,4,4,3,5,5,1,1,5,4,),2,5))*: obtenemos la matriz
- *(muestra=t(muestra))*
- Calculamos las distancias euclídeas: *(distancias=as.matrix(dist(muestra)))*
- Hay que ordenar los valores *(distancias=matrix(distancias,5,5))*
  for (i in 1:5)distancias[,i]=sort(distancias[,i]); (distanciasordenadas=distancias)
- Como el primer vecino es él mismo, la distancia es cero, por lo que vamos a usar k=4 *for(i in 1:5)if(distanciasordenadas[4,i]¿2.5)print(i);print("es un suceso anómalo o outlier")*
- *(distanciasM=as.matrix(dist(muestra,method="manhattan")))*

Local Outlier Factor Existen varios paquetes para hacer este cálculo que utilizan métodos distintos al visto en teoría: RLof, DDoutlier, DMwR son algunos de ellos.

3. Ejercicios de forma autónoma

Realización de cuatro ejercicios de forma autónoma por cada grupo de estudiantes en los que se van a realizar, utilizando el entorno R, un análisis de descripción de datos, un análisis de asociación y dos análisis de detección de datos anómalos, aplicando todos los conceptos teóricos vistos en cada lección:

3.1. Análisis de descripción de 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.

3.2. Análisis de 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

3.3. Análisis de detección de datos anómalos

Esto es un poco más de texto en el párrafo.

3.3.1. Técnicas con base estadística

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.

3.3.2. Técnicas basadas en la proximidad y en la densidad

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.