Clase 2

Erick Cuevas Fernández

25/8/2019

Documentos csv

comma separate values

read.csv

Funciones que leen .csv

- `read.csv(file_path)`
- `read_csv(file_path)`

Argumentos de la función

```
read.csv(file, header = TRUE, sep = ",", quote = "\"", dec = ".", fill = TRUE, comment.char
= "", ...)
```

```
read.csv2(file, header = TRUE, sep = ";", quote = "\"", dec = ",", fill = TRUE, comment.char
= "", ...)
```

Tambien podemos leer archivos separados por espacio y tabuladores:

```
read.table(file, header = FALSE, sep = "", quote = "\"'", dec = ".", numerals = c("allow.loss",
"warn.loss", "no.loss"), row.names, col.names, as.is = !stringsAsFactors, na.strings =
"NA", colClasses = NA, nrows = -1, skip = 0, check.names = TRUE, fill = !blank.lines.skip,
strip.white = FALSE, blank.lines.skip = TRUE, comment.char = "#", allowEscapes = FALSE,
flush = FALSE, stringsAsFactors = default.stringsAsFactors(), fileEncoding = "", encoding
= "unknown", text, skipNul = FALSE)
```

```
read.delim(file, header = TRUE, sep = "\t", quote = "\"", dec = ".", fill = TRUE, comment.char
= "", ...)
```

```
read.delim2(file, header = TRUE, sep = "\t", quote = "\"", dec = ",", fill = TRUE, comment.char
= "", ...)
```

Ejemplo para leer un archivo .csv

```
names_data <- c("age", "sex", "cp", "trestbps", "chol", "fbs", "restecg",
                "thalach", "exang", "oldpeak", "slope", "ca", "thal", "num")
data_cleveland <- read.csv("Datasets/processed.cleveland.data")
colnames(data_cleveland) <- names_data

str(data_cleveland)
```

```
## 'data.frame':    302 obs. of  14 variables:
##  $ age     : num  67 67 37 41 56 62 57 63 53 57 ...
##  $ sex     : num  1 1 1 0 1 0 0 1 1 1 ...
##  $ cp      : num  4 4 3 2 2 4 4 4 4 4 ...
```

```
##  $ trestbps: num  160 120 130 130 120 140 120 130 140 140 ...
##  $ chol    : num  286 229 250 204 236 268 354 254 203 192 ...
##  $ fbs     : num  0 0 0 0 0 0 0 0 1 0 ...
##  $ restecg : num  2 2 0 2 0 2 0 2 2 0 ...
##  $ thalach : num  108 129 187 172 178 160 163 147 155 148 ...
##  $ exang   : num  1 1 0 0 0 0 1 0 1 0 ...
##  $ oldpeak : num  1.5 2.6 3.5 1.4 0.8 3.6 0.6 1.4 3.1 0.4 ...
##  $ slope   : num  2 2 3 1 1 3 1 2 3 2 ...
##  $ ca      : Factor w/ 5 levels "?","0.0","1.0",..: 5 4 2 2 2 4 2 3 2 2 ...
##  $ thal    : Factor w/ 4 levels "?","3.0","6.0",..: 2 4 2 2 2 2 2 4 4 3 ...
##  $ num     : int  2 1 0 0 0 3 0 2 1 0 ...
```

Attribute Information:

- age: age in years
- sex: sex (1 = male; 0 = female)
- cp: chest pain type

```
-- Value 1: typical angina
-- Value 2: atypical angina
-- Value 3: non-anginal pain
-- Value 4: asymptomatic
```

- trestbps: resting blood pressure (in mm Hg on admission to the hospital)
- chol: serum cholestoral in mg/dl
- fbs: (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
- restecg: resting electrocardiographic results

```
-- Value 0: normal
-- Value 1: having ST-T wave abnormality (T wave inversions and/or ST
            elevation or depression of > 0.05 mV)
-- Value 2: showing probable or definite left ventricular hypertrophy
            by Estes' criteria
```

- thalach: maximum heart rate achieved
- exang: exercise induced angina (1 = yes; 0 = no)
- oldpeak = ST depression induced by exercise relative to rest
- slope: the slope of the peak exercise ST segment

```
-- Value 1: upsloping
-- Value 2: flat
-- Value 3: downsloping
```

- ca: number of major vessels (0-3) colored by flourosopy
- thal: 3 = normal; 6 = fixed defect; 7 = reversable defect
- num: diagnosis of heart disease (angiographic disease status)

```
-- Value 0: < 50% diameter narrowing
-- Value 1: > 50% diameter narrowing
```

Source Information:

(a) Creators:
- 1. Hungarian Institute of Cardiology. Budapest: Andras Janosi, M.D.
- 2. University Hospital, Zurich, Switzerland: William Steinbrunn, M.D.
- 3. University Hospital, Basel, Switzerland: Matthias Pfisterer, M.D.
- 4. V.A. Medical Center, Long Beach and Cleveland Clinic Foundation: Robert Detrano, M.D., Ph.D.

(b) Donor: David W. Aha (aha@ics.uci.edu) (714) 856-8779

(c) Date: July, 1988*

Documentos json

.json

JavaScript Object Notation

El tipo de archivo con mayor uso en las bases de datos, ya que consumo muy poco espacio de memoria. La paquetería *jsonlite* nos permite leer este tipo de archivos en R.

```
library(jsonlite)
```

fromJSON()

```
id <- 681
url_chem <- paste0("https://pubchem.ncbi.nlm.nih.gov/rest/pug_view/data/compound/"
                   ,id,
                   "/json")

response <- fromJSON(url_chem)

# str(response)

archivo_sections<-response$Record$Section$Section
```

Desde JSON obtenemos una lista o un data.frame, y ya que sabemos manejar listas, podemos empezar a obtener los datos que deseamos.

Documentos xml

XML significa Extensible Markup Language (Lenguaje de Marcado Extensible), antes era muy utilizado en las bases de datos, pero progresivamente han ido sustituyendolo por archivos tipo **json**.

Hay multiples paqueterias en R para lidiar con este tipo de archivos, cada una tiene sus especificaciones; este tipo de archivos tienen sus detalles al momento de manejarlos, ya que hay que hacer una serie de pasos para obtener la información de grandes bases de datos; en lo personal recomiendo usar este tipo de archivos solo si es la unica forma de obtener la información.

Las paqueterias que sirven para tratar estos archivos en R son:

- **XML**
- **XML2**
- **httr**

Entre algunos, tiene su arte *parsear* estos documentos.

Ejemplo

```
library(XML)
url<-"Datasets/cd_catalog.xml"
#xmlParse agrega ubicacion al fichero
xmldoc<-xmlParse(url)  #XMLInternalDocument
#Esto nos coloca en el nodo raiz
rootnode<-xmlRoot(xmldoc) #Para extraer
rootnode[1]
```

```
## $CD
## <CD>
##   <TITLE>Empire Burlesque</TITLE>
##   <ARTIST>Bob Dylan</ARTIST>
##   <COUNTRY>USA</COUNTRY>
##   <COMPANY>Columbia</COMPANY>
##   <PRICE>10.90</PRICE>
##   <YEAR>1985</YEAR>
## </CD>
##
## attr(,"class")
## [1] "XMLInternalNodeList" "XMLNodeList"
```

```
#Primera diferencia con CSV es que XML hay que extraer unos por uno los datos
#Para hacer data.frame con XML, se extrae cada valor del valor raiz
cds_data<-xmlSApply(rootnode,function(x)xmlSApply(x,xmlValue)) #Recorre todos los nodos y aplica la fun
#Ahora se procesa un poco, se hace una transposición (t), y especificarle que Row no tiene nombres
cds.catalog<-data.frame(t(cds_data),row.names=NULL)
head(cds.catalog,2)
```

```
##              TITLE       ARTIST COUNTRY     COMPANY PRICE YEAR
## 1 Empire Burlesque    Bob Dylan     USA    Columbia 10.90 1985
## 2  Hide your heart Bonnie Tylor      UK CBS Records  9.90 1988
```

```
cds.catalog[1:5,]
```

```
##                 TITLE          ARTIST COUNTRY        COMPANY PRICE YEAR
## 1    Empire Burlesque       Bob Dylan     USA       Columbia 10.90 1985
## 2     Hide your heart    Bonnie Tylor      UK    CBS Records  9.90 1988
## 3       Greatest Hits    Dolly Parton     USA            RCA  9.90 1982
## 4 Still got the blues       Gary More      UK Virgin redords 10.20 1990
## 5               Eros Eros Ramazzotti      EU            BMG  9.90 1997
```

Documentos txt

```
texto <- readLines("Datasets/ejemplo.txt")
```

```
## Warning in readLines("Datasets/ejemplo.txt"): incomplete final line found
## on 'Datasets/ejemplo.txt'
```

```
head(texto)
```

```
## [1] "Glucose is the long-established, obligatory fuel for brain that fulfills many"
## [2] "      critical functions, including ATP production, oxidative stress management, and"
## [3] "      synthesis of neurotransmitters, neuromodulators, and structural components."
## [4] "      Neuronal glucose oxidation exceeds that in astrocytes, but both rates increase in"
## [5] "      direct proportion to excitatory neurotransmission; signaling and metabolism are"
## [6] "      closely coupled at the local level. Exact details of neuron-astrocyte"
```

Y desee aqui hay todo un universo de analisis de texto que se puede lograr, desde contar la frecuencia de palabras, desglosar un texto en forma de vector, hacer un analisis de sentimientos del texto, hacer nube de palabras, ect, etc.

Para más información se pueden acercar a mi o estudiar por su cuenta en el libro:

Text Mining with R

EXTRA

Con R tambien podemos enlazarnos con otras plataformas para manejar bases de datos como **SQL** y **SPSS**, para acceder a estos datos se usan las paqueterias **"sqldf"** y **"haven"**.

La paquetería **"foreign"** nos permite abrir la extension de los siguientes archivos:

- .arff
- .dta
- .mtp
- .xport
- .ssd
- .octave
- .spss
- .dbf
- .epiinfo
- .systat

Y la paquetería **"readxl"** podemos leer archivos que creamos en *Excel*.