Contenido

[Curso 3 2](#_Toc32941809)

[Week 1 2](#_Toc32941810)

[Chequear y crear directorios desde R 2](#_Toc32941811)

[Descargar archivos desde internet 2](#_Toc32941812)

[Abrir Archivos 2](#_Toc32941813)

[Xlsx 2](#_Toc32941814)

[Paginas Web 3](#_Toc32941815)

[Abrir Json 4](#_Toc32941816)

[Data table 5](#_Toc32941817)

[SQL 7](#_Toc32941818)

[Read from web 18](#_Toc32941819)

[Subsetting 18](#_Toc32941820)

[Summarazing data 20](#_Toc32941821)

[Creating New Variables 28](#_Toc32941822)

[Reshaping Data 31](#_Toc32941823)

[Dplyr 32](#_Toc32941824)

[Editing text variables 33](#_Toc32941825)

[Video 2 35](#_Toc32941826)

[Metacaracteres 35](#_Toc32941827)

[Video 3 36](#_Toc32941828)

Curso 3

Week 1

tafuenza

17-11-2019

### Chequear y crear directorios desde R

Si quiero crear o chequear si existe una carpeta dentro de mi “work directory (wd)” debo usar:

file.exists("DirectoryName") # Chequea si el directorio existe  
dir.create("DirectoryName") # Crea el directorio si no existe

### Descargar archivos desde internet

Para descargar un archivo directamente desde internet, debemos utilizar el siguiente codigo:

download.file(fileURL, destfile, method) # URL, carpeta de destino y nombre de archivo (data/mydata.csv), method (Curl)

# Abrir Archivos

El clasico archivo que se va a leer es un archivo .csv

data <- read.csv(file,header,sep,row.names,row) # Archivo, si tiene o no titulo las variables, como estan separadas las columnas, si las filas tienen nombres, cuantas filas quiero leer  
data <- read.csv(quotes, na.strings, skip) # Si existen valores entre comillas, como se representan los NA, saltar algunas lineas desde el comienzo

# Xlsx

library("xlsx"")  
data <- read.xlsx("data.xls", sheetIndex = 1, HEADER = TRUE, colIndex = 2:3, row.Index = 1:4)

# Paginas Web

R permite leer directamente archivos de paginas web como XML

library('XML')  
url <- "http://www.w3schools.com/xml/simple.xml"  
download.file(url, destfile = "foo.xml")  
doc <- xmlTreeParse("foo.xml", useInternal=TRUE)  
rootNode <- xmlRoot(doc)  
xmlName(rootNode)

## [1] "breakfast\_menu"

rootNode[[1]]

## <food>  
## <name>Belgian Waffles</name>  
## <price>$5.95</price>  
## <description>Two of our famous Belgian Waffles with plenty of real maple syrup</description>  
## <calories>650</calories>  
## </food>

rootNode[[1]][[1]]

## <name>Belgian Waffles</name>

xmlSApply(rootNode,xmlValue)

## food   
## "Belgian Waffles$5.95Two of our famous Belgian Waffles with plenty of real maple syrup650"   
## food   
## "Strawberry Belgian Waffles$7.95Light Belgian waffles covered with strawberries and whipped cream900"   
## food   
## "Berry-Berry Belgian Waffles$8.95Light Belgian waffles covered with an assortment of fresh berries and whipped cream900"   
## food   
## "French Toast$4.50Thick slices made from our homemade sourdough bread600"   
## food   
## "Homestyle Breakfast$6.95Two eggs, bacon or sausage, toast, and our ever-popular hash browns950"

#Para encontrar otros datos especificos se puede usar  
#/node  
#//node  
#node[@attr-name]  
#node[@attr-name = 'bob']  
xpathSApply(rootNode,'//name',xmlValue)

## [1] "Belgian Waffles" "Strawberry Belgian Waffles"   
## [3] "Berry-Berry Belgian Waffles" "French Toast"   
## [5] "Homestyle Breakfast"

xpathSApply(rootNode,'//price',xmlValue)

## [1] "$5.95" "$7.95" "$8.95" "$4.50" "$6.95"

# Abrir Json

library('jsonlite')

## Warning: package 'jsonlite' was built under R version 3.6.1

URL <- "https://api.github.com/users/jtleek/repos"  
data <- fromJSON(URL)  
names(data)

## [1] "id" "node\_id" "name"   
## [4] "full\_name" "private" "owner"   
## [7] "html\_url" "description" "fork"   
## [10] "url" "forks\_url" "keys\_url"   
## [13] "collaborators\_url" "teams\_url" "hooks\_url"   
## [16] "issue\_events\_url" "events\_url" "assignees\_url"   
## [19] "branches\_url" "tags\_url" "blobs\_url"   
## [22] "git\_tags\_url" "git\_refs\_url" "trees\_url"   
## [25] "statuses\_url" "languages\_url" "stargazers\_url"   
## [28] "contributors\_url" "subscribers\_url" "subscription\_url"   
## [31] "commits\_url" "git\_commits\_url" "comments\_url"   
## [34] "issue\_comment\_url" "contents\_url" "compare\_url"   
## [37] "merges\_url" "archive\_url" "downloads\_url"   
## [40] "issues\_url" "pulls\_url" "milestones\_url"   
## [43] "notifications\_url" "labels\_url" "releases\_url"   
## [46] "deployments\_url" "created\_at" "updated\_at"   
## [49] "pushed\_at" "git\_url" "ssh\_url"   
## [52] "clone\_url" "svn\_url" "homepage"   
## [55] "size" "stargazers\_count" "watchers\_count"   
## [58] "language" "has\_issues" "has\_projects"   
## [61] "has\_downloads" "has\_wiki" "has\_pages"   
## [64] "forks\_count" "mirror\_url" "archived"   
## [67] "disabled" "open\_issues\_count" "license"   
## [70] "forks" "open\_issues" "watchers"   
## [73] "default\_branch"

names(data$owner) # Los data frame te permite tener dataframes unos dentro de otros

## [1] "login" "id" "node\_id"   
## [4] "avatar\_url" "gravatar\_id" "url"   
## [7] "html\_url" "followers\_url" "following\_url"   
## [10] "gists\_url" "starred\_url" "subscriptions\_url"   
## [13] "organizations\_url" "repos\_url" "events\_url"   
## [16] "received\_events\_url" "type" "site\_admin"

# Tambien puedo transformar un DataFrame a JSON con toJSON()

# Data table

Data.table() es capaz de hacer lo mismo que data.frame() pero tiene algunas opciones interesantes que pueden ser utiles

library('data.table')

## Warning: package 'data.table' was built under R version 3.6.1

set.seed(1)  
DT <- data.table( x = rnorm(10), y = rbinom(10,1,0.5), z = rpois(10,1))  
tables() ## Permite conocer algunas caracteristicas de la tabla de datos

## NAME NROW NCOL MB COLS KEY  
## 1: DT 10 3 0 x,y,z   
## Total: 0MB

DT[2,]

## x y z  
## 1: 0.1836433 0 1

DT[c(2,3)] # Cuando le entrego un indice, siempre lo comprende como las filas

## x y z  
## 1: 0.1836433 0 1  
## 2: -0.8356286 1 1

DT[,c(2,3)] # Si lo tratas de hacer por columnas no funciona

## y z  
## 1: 1 1  
## 2: 0 1  
## 3: 1 1  
## 4: 0 0  
## 5: 0 2  
## 6: 0 1  
## 7: 0 2  
## 8: 0 0  
## 9: 1 1  
## 10: 0 1

DT[,list(sum(x),mean(z))] # Le puedes entregar una lista de funciones que hacer por variable

## V1 V2  
## 1: 1.322028 1

DT[,w := x^2] # Es mas eficiente al agregar nuevas variables  
DT[,m:= {tmp <-(x+z); log2(tmp+5)}] # Puedo crear funciones para agregar nuevas variables  
DT[,a:= x>0] # Comprobar logicas y agregar nueva variable  
DT[,b:= mean(x+w), by = a] # Calcula el promedio de todos los a = TRUE y lo de A = FALSE separados  
DT

## x y z w m a b  
## 1: -0.6264538 1 1 0.39244438 2.425874 FALSE -0.1826973  
## 2: 0.1836433 0 1 0.03372487 2.628457 TRUE 1.2852371  
## 3: -0.8356286 1 1 0.69827518 2.368593 FALSE -0.1826973  
## 4: 1.5952808 0 0 2.54492084 2.721434 TRUE 1.2852371  
## 5: 0.3295078 0 2 0.10857537 2.873716 TRUE 1.2852371  
## 6: -0.8204684 0 1 0.67316837 2.372822 FALSE -0.1826973  
## 7: 0.4874291 0 2 0.23758708 2.904470 TRUE 1.2852371  
## 8: 0.7383247 0 0 0.54512337 2.520630 TRUE 1.2852371  
## 9: 0.5757814 1 1 0.33152416 2.717162 TRUE 1.2852371  
## 10: -0.3053884 0 1 0.09326207 2.509597 FALSE -0.1826973

set.seed(2)  
DATA <- data.table(x=sample(letters[1:3], 1E5,TRUE))  
DATA[,.N, by = x] # Cuenta la cantidad de veces que se repite cada uno de los elementos de x

## x N  
## 1: a 33098  
## 2: c 33566  
## 3: b 33336

DT <- data.table(x=rep(c("a","b","c"), each = 100), y= rnorm(300))  
setkey(DT,x)  
A <- DT['a'] ## Al asignar a x como la llave, al entregarle directamente algun cambio simplemente usa la llave  
A[1:10]

## x y  
## 1: a 1.66474613  
## 2: a -0.82298104  
## 3: a -0.31633398  
## 4: a -1.24322526  
## 5: a -1.18849475  
## 6: a -2.09007244  
## 7: a -1.03697558  
## 8: a 0.09620607  
## 9: a -1.19671910  
## 10: a -1.51553259

Week 2

Tomás

18 de noviembre de 2019

# SQL

library(DBI)

## Warning: package 'DBI' was built under R version 3.6.1

library(RMySQL)  
ucscDB <- dbConnect(MySQL(),user = 'genome', host = 'genome-mysql.cse.ucsc.edu')  
result <- dbGetQuery(ucscDB,'show databases'); dbDisconnect(ucscDB)

## [1] TRUE

result ## Entrega una lista de todas las bases de datos disponibles en el host

## Database  
## 1 acaChl1  
## 2 ailMel1  
## 3 allMis1  
## 4 allSin1  
## 5 amaVit1  
## 6 anaPla1  
## 7 ancCey1  
## 8 angJap1  
## 9 anoCar1  
## 10 anoCar2  
## 11 anoGam1  
## 12 anoGam3  
## 13 apaSpi1  
## 14 apaVit1  
## 15 apiMel1  
## 16 apiMel2  
## 17 aplCal1  
## 18 aptFor1  
## 19 aptMan1  
## 20 aquChr2  
## 21 araMac1  
## 22 ascSuu1  
## 23 balAcu1  
## 24 balPav1  
## 25 bisBis1  
## 26 bosTau2  
## 27 bosTau3  
## 28 bosTau4  
## 29 bosTau5  
## 30 bosTau6  
## 31 bosTau7  
## 32 bosTau8  
## 33 bosTau9  
## 34 bosTauMd3  
## 35 braFlo1  
## 36 bruMal2  
## 37 bucRhi1  
## 38 burXyl1  
## 39 caeAng2  
## 40 caeJap1  
## 41 caeJap4  
## 42 caePb1  
## 43 caePb2  
## 44 caePb3  
## 45 caeRem2  
## 46 caeRem3  
## 47 caeRem4  
## 48 caeSp111  
## 49 caeSp51  
## 50 calAnn1  
## 51 calJac1  
## 52 calJac3  
## 53 calMil1  
## 54 canFam1  
## 55 canFam2  
## 56 canFam3  
## 57 capCar1  
## 58 carCri1  
## 59 cavPor3  
## 60 cb1  
## 61 cb3  
## 62 cb4  
## 63 ce10  
## 64 ce11  
## 65 ce2  
## 66 ce4  
## 67 ce6  
## 68 cerSim1  
## 69 chaVoc2  
## 70 cheMyd1  
## 71 chlSab2  
## 72 chlUnd1  
## 73 choHof1  
## 74 chrPic1  
## 75 chrPic2  
## 76 ci1  
## 77 ci2  
## 78 ci3  
## 79 colLiv1  
## 80 colStr1  
## 81 corBra1  
## 82 corCor1  
## 83 cotJap2  
## 84 criGri1  
## 85 criGriChoV1  
## 86 criGriChoV2  
## 87 cucCan1  
## 88 danRer1  
## 89 danRer10  
## 90 danRer11  
## 91 danRer2  
## 92 danRer3  
## 93 danRer4  
## 94 danRer5  
## 95 danRer6  
## 96 danRer7  
## 97 dasNov3  
## 98 dipOrd1  
## 99 dirImm1  
## 100 dm1  
## 101 dm2  
## 102 dm3  
## 103 dm6  
## 104 dp2  
## 105 dp3  
## 106 droAna1  
## 107 droAna2  
## 108 droEre1  
## 109 droGri1  
## 110 droMoj1  
## 111 droMoj2  
## 112 droPer1  
## 113 droSec1  
## 114 droSim1  
## 115 droSim2  
## 116 droVir1  
## 117 droVir2  
## 118 droYak1  
## 119 droYak2  
## 120 eboVir3  
## 121 echTel1  
## 122 echTel2  
## 123 egrGar1  
## 124 equCab1  
## 125 equCab2  
## 126 equCab3  
## 127 eriEur1  
## 128 eriEur2  
## 129 eurHel1  
## 130 falChe1  
## 131 falPer1  
## 132 felCat3  
## 133 felCat4  
## 134 felCat5  
## 135 felCat8  
## 136 felCat9  
## 137 ficAlb2  
## 138 fr1  
## 139 fr2  
## 140 fr3  
## 141 fulGla1  
## 142 gadMor1  
## 143 galGal2  
## 144 galGal3  
## 145 galGal4  
## 146 galGal5  
## 147 galGal6  
## 148 galVar1  
## 149 gasAcu1  
## 150 gavSte1  
## 151 gbMeta  
## 152 geoFor1  
## 153 go  
## 154 go080130  
## 155 go140213  
## 156 go150121  
## 157 go180426  
## 158 gorGor3  
## 159 gorGor4  
## 160 gorGor5  
## 161 haeCon2  
## 162 halAlb1  
## 163 halLeu1  
## 164 hetBac1  
## 165 hetGla1  
## 166 hetGla2  
## 167 hg16  
## 168 hg17  
## 169 hg18  
## 170 hg19  
## 171 hg19Patch10  
## 172 hg19Patch13  
## 173 hg38  
## 174 hg38Patch11  
## 175 hgFixed  
## 176 hgcentral  
## 177 information\_schema  
## 178 latCha1  
## 179 lepDis1  
## 180 letCam1  
## 181 loaLoa1  
## 182 loxAfr3  
## 183 macEug1  
## 184 macEug2  
## 185 macFas5  
## 186 manPen1  
## 187 melGal1  
## 188 melGal5  
## 189 melHap1  
## 190 melInc2  
## 191 melUnd1  
## 192 merNub1  
## 193 mesUni1  
## 194 micMur1  
## 195 micMur2  
## 196 mm10  
## 197 mm10Patch4  
## 198 mm5  
## 199 mm6  
## 200 mm7  
## 201 mm8  
## 202 mm9  
## 203 monDom1  
## 204 monDom4  
## 205 monDom5  
## 206 musFur1  
## 207 myoLuc2  
## 208 nanPar1  
## 209 nasLar1  
## 210 necAme1  
## 211 nipNip1  
## 212 nomLeu1  
## 213 nomLeu2  
## 214 nomLeu3  
## 215 ochPri2  
## 216 ochPri3  
## 217 oncVol1  
## 218 opiHoa1  
## 219 oreNil1  
## 220 oreNil2  
## 221 oreNil3  
## 222 ornAna1  
## 223 ornAna2  
## 224 oryCun2  
## 225 oryLat2  
## 226 otoGar3  
## 227 oviAri1  
## 228 oviAri3  
## 229 oviAri4  
## 230 panPan1  
## 231 panPan2  
## 232 panRed1  
## 233 panTro1  
## 234 panTro2  
## 235 panTro3  
## 236 panTro4  
## 237 panTro5  
## 238 panTro6  
## 239 papAnu2  
## 240 papAnu4  
## 241 papHam1  
## 242 pelCri1  
## 243 pelSin1  
## 244 performance\_schema  
## 245 petMar1  
## 246 petMar2  
## 247 petMar3  
## 248 phaCar1  
## 249 phaLep1  
## 250 phoRub1  
## 251 picPub1  
## 252 ponAbe2  
## 253 ponAbe3  
## 254 priExs1  
## 255 priPac1  
## 256 priPac3  
## 257 proCap1  
## 258 proteins120806  
## 259 proteins121210  
## 260 proteins140122  
## 261 proteins150225  
## 262 proteins160229  
## 263 proteins180404  
## 264 proteome  
## 265 pteGut1  
## 266 pteVam1  
## 267 pygAde1  
## 268 pytBiv1  
## 269 rheMac1  
## 270 rheMac10  
## 271 rheMac2  
## 272 rheMac3  
## 273 rheMac8  
## 274 rhiRox1  
## 275 rn3  
## 276 rn4  
## 277 rn5  
## 278 rn6  
## 279 sacCer1  
## 280 sacCer2  
## 281 sacCer3  
## 282 saiBol1  
## 283 sarHar1  
## 284 serCan1  
## 285 sorAra1  
## 286 sorAra2  
## 287 sp120323  
## 288 sp121210  
## 289 sp140122  
## 290 sp150225  
## 291 sp160229  
## 292 sp180404  
## 293 speTri2  
## 294 strCam1  
## 295 strPur1  
## 296 strPur2  
## 297 strRat2  
## 298 susScr11  
## 299 susScr2  
## 300 susScr3  
## 301 taeGut1  
## 302 taeGut2  
## 303 tarSyr1  
## 304 tarSyr2  
## 305 tauEry1  
## 306 tetNig1  
## 307 tetNig2  
## 308 thaSir1  
## 309 tinGut2  
## 310 triMan1  
## 311 triSpi1  
## 312 triSui1  
## 313 tupBel1  
## 314 turTru2  
## 315 tytAlb1  
## 316 uniProt  
## 317 vicPac1  
## 318 vicPac2  
## 319 visiGene  
## 320 xenLae2  
## 321 xenTro1  
## 322 xenTro2  
## 323 xenTro3  
## 324 xenTro7  
## 325 xenTro9  
## 326 zonAlb1

hg19 <- dbConnect(MySQL(),user = 'genome', host = 'genome-mysql.cse.ucsc.edu', db = 'hg19')  
allTables <- dbListTables(hg19)  
length(allTables)

## [1] 12425

allTables[1:5] # Ejemplo especifico de una database que tiene mas de 10.000 data frames dentro

## [1] "HInv" "HInvGeneMrna" "acembly" "acemblyClass"  
## [5] "acemblyPep"

dbListFields(hg19,'affyU133Plus2')

## [1] "bin" "matches" "misMatches" "repMatches" "nCount"   
## [6] "qNumInsert" "qBaseInsert" "tNumInsert" "tBaseInsert" "strand"   
## [11] "qName" "qSize" "qStart" "qEnd" "tName"   
## [16] "tSize" "tStart" "tEnd" "blockCount" "blockSizes"   
## [21] "qStarts" "tStarts"

dbGetQuery(hg19,'select count(\*) from affyU133Plus2') ## Cuenta todas las filas que existen en la tabla affy....

## count(\*)  
## 1 58463

affydata <- dbReadTable(hg19, 'affyU133Plus2') # Retorna la dataframe affy... con esto puedo trabajarla

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 0 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 1 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 2 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 3 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 4 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 5 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 6 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 7 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 8 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 11  
## imported as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 12  
## imported as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 13  
## imported as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 15  
## imported as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 16  
## imported as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 17  
## imported as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 18  
## imported as numeric

class(affydata)

## [1] "data.frame"

head(affydata)

## bin matches misMatches repMatches nCount qNumInsert qBaseInsert  
## 1 585 530 4 0 23 3 41  
## 2 585 3355 17 0 109 9 67  
## 3 585 4156 14 0 83 16 18  
## 4 585 4667 9 0 68 21 42  
## 5 585 5180 14 0 167 10 38  
## 6 585 468 5 0 14 0 0  
## tNumInsert tBaseInsert strand qName qSize qStart qEnd tName  
## 1 3 898 - 225995\_x\_at 637 5 603 chr1  
## 2 9 11621 - 225035\_x\_at 3635 0 3548 chr1  
## 3 2 93 - 226340\_x\_at 4318 3 4274 chr1  
## 4 3 5743 - 1557034\_s\_at 4834 48 4834 chr1  
## 5 1 29 - 231811\_at 5399 0 5399 chr1  
## 6 0 0 - 236841\_at 487 0 487 chr1  
## tSize tStart tEnd blockCount  
## 1 249250621 14361 15816 5  
## 2 249250621 14381 29483 17  
## 3 249250621 14399 18745 18  
## 4 249250621 14406 24893 23  
## 5 249250621 19688 25078 11  
## 6 249250621 27542 28029 1  
## blockSizes  
## 1 93,144,229,70,21,  
## 2 73,375,71,165,303,360,198,661,201,1,260,250,74,73,98,155,163,  
## 3 690,10,32,33,376,4,5,15,5,11,7,41,277,859,141,51,443,1253,  
## 4 99,352,286,24,49,14,6,5,8,149,14,44,98,12,10,355,837,59,8,1500,133,624,58,  
## 5 131,26,1300,6,4,11,4,7,358,3359,155,  
## 6 487,  
## qStarts  
## 1 34,132,278,541,611,  
## 2 87,165,540,647,818,1123,1484,1682,2343,2545,2546,2808,3058,3133,3206,3317,3472,  
## 3 44,735,746,779,813,1190,1195,1201,1217,1223,1235,1243,1285,1564,2423,2565,2617,3062,  
## 4 0,99,452,739,764,814,829,836,842,851,1001,1016,1061,1160,1173,1184,1540,2381,2441,2450,3951,4103,4728,  
## 5 0,132,159,1460,1467,1472,1484,1489,1497,1856,5244,  
## 6 0,  
## tStarts  
## 1 14361,14454,14599,14968,15795,  
## 2 14381,14454,14969,15075,15240,15543,15903,16104,16853,17054,17232,17492,17914,17988,18267,24736,29320,  
## 3 14399,15089,15099,15131,15164,15540,15544,15549,15564,15569,15580,15587,15628,15906,16857,16998,17049,17492,  
## 4 14406,20227,20579,20865,20889,20938,20952,20958,20963,20971,21120,21134,21178,21276,21288,21298,21653,22492,22551,22559,24059,24211,24835,  
## 5 19688,19819,19845,21145,21151,21155,21166,21170,21177,21535,24923,  
## 6 27542,

query <- dbSendQuery(hg19, 'select \* from affyU133Plus2 where misMatches between 1 and 3')

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 0 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 1 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 2 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 3 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 4 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 5 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 6 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 7 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 8 imported  
## as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 11  
## imported as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 12  
## imported as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 13  
## imported as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 15  
## imported as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 16  
## imported as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 17  
## imported as numeric

## Warning in .local(conn, statement, ...): Unsigned INTEGER in col 18  
## imported as numeric

affyMis <- fetch(query); quantile(affyMis$misMatches)

## 0% 25% 50% 75% 100%   
## 1 1 2 2 3

affyMissmall <- fetch(query, n=10); dbClearResult(query);

## [1] TRUE

dbDisconnect(hg19)

## [1] TRUE

# Read from web

Week 3

Tomás

19 de noviembre de 2019

# Subsetting

library(plyr)  
set.seed(123)  
x <- data.frame('var1' = sample(1:5),"var2" = sample(6:10), "var3" = sample(11:15))  
x <- x[sample(1:5),] ; x$var2[c(1,3)] = NA  
x[,1] # Primera columna del dataframe

## [1] 3 4 1 5 2

x[,"var1"] # Columna var1 del dataframe

## [1] 3 4 1 5 2

x[1:2,"var2"]#Columna var 2 y filas 1:2

## [1] NA 10

x[(x$var1 <= 3 & x$var3 > 11),] # Busco filas que tengan var 1 mayor/igual que 3 y var 3 menor que 11

## var1 var2 var3  
## 1 3 NA 12  
## 5 1 NA 15  
## 2 2 6 13

x[(x$var1 <= 3 | x$var3 > 15),] # Busco filas que var 1 sea mayor o igual a 3 o que var3 sea mayor que 15

## var1 var2 var3  
## 1 3 NA 12  
## 5 1 NA 15  
## 2 2 6 13

x[which(x$var2 >8),] #La funcion which entrega los indices donde se comple esa condicion sin contar los NA

## var1 var2 var3  
## 4 4 10 14

x[x$var2 >8,] # Sin la funcion which considera los NA

## var1 var2 var3  
## NA NA NA NA  
## 4 4 10 14  
## NA.1 NA NA NA

sort(x$var1) # Ordena la columna de menor a mayor

## [1] 1 2 3 4 5

sort(x$var1, decreasing = TRUE)

## [1] 5 4 3 2 1

sort(x$var2, na.last = TRUE)

## [1] 6 7 10 NA NA

x[order(x$var1),] #Me ordena el dataframe en funcion de la variable 1

## var1 var2 var3  
## 5 1 NA 15  
## 2 2 6 13  
## 1 3 NA 12  
## 4 4 10 14  
## 3 5 7 11

x[order(x$var1,x$var3),] # Si existe empate en la variable 1, se orden por la variable 3

## var1 var2 var3  
## 5 1 NA 15  
## 2 2 6 13  
## 1 3 NA 12  
## 4 4 10 14  
## 3 5 7 11

arrange(x,var1) # Mismo caso que order, pero mas compracto

## var1 var2 var3  
## 1 1 NA 15  
## 2 2 6 13  
## 3 3 NA 12  
## 4 4 10 14  
## 5 5 7 11

arrange(x,desc(var1)) # Decreciente

## var1 var2 var3  
## 1 5 7 11  
## 2 4 10 14  
## 3 3 NA 12  
## 4 2 6 13  
## 5 1 NA 15

x$var4 <- rnorm(5) # Añadir columna nueva  
y <- cbind(x, "var5" = rnorm(5)) # Lo mismo pero debo crear otro dataframe, rbind agrega filas

# Summarazing data

data <- read.csv("Restaurants.csv")  
head(data,3) # Visualizar las 3 primeras lineas

## name zipCode neighborhood councilDistrict policeDistrict  
## 1 410 21206 Frankford 2 NORTHEASTERN  
## 2 1919 21231 Fells Point 1 SOUTHEASTERN  
## 3 SAUTE 21224 Canton 1 SOUTHEASTERN  
## Location.1  
## 1 4509 BELAIR ROAD\nBaltimore, MD  
## 2 1919 FLEET ST\nBaltimore, MD  
## 3 2844 HUDSON ST\nBaltimore, MD

tail(data,3) # Visualizar las 3 ultimas lineas

## name zipCode neighborhood councilDistrict policeDistrict  
## 1325 ZINK'S CAFÂ\220 21213 Belair-Edison 13 NORTHEASTERN  
## 1326 ZISSIMOS BAR 21211 Hampden 7 NORTHERN  
## 1327 ZORBAS 21224 Greektown 2 SOUTHEASTERN  
## Location.1  
## 1325 3300 LAWNVIEW AVE\nBaltimore, MD  
## 1326 1023 36TH ST\nBaltimore, MD  
## 1327 4710 EASTERN Ave\nBaltimore, MD

summary(data) # Resumen de la data

## name zipCode neighborhood  
## MCDONALD'S : 8 Min. :-21226 Downtown :128   
## POPEYES FAMOUS FRIED CHICKEN: 7 1st Qu.: 21202 Fells Point : 91   
## SUBWAY : 6 Median : 21218 Inner Harbor: 89   
## KENTUCKY FRIED CHICKEN : 5 Mean : 21185 Canton : 81   
## BURGER KING : 4 3rd Qu.: 21226 Federal Hill: 42   
## DUNKIN DONUTS : 4 Max. : 21287 Mount Vernon: 33   
## (Other) :1293 (Other) :863   
## councilDistrict policeDistrict  
## Min. : 1.000 SOUTHEASTERN:385   
## 1st Qu.: 2.000 CENTRAL :288   
## Median : 9.000 SOUTHERN :213   
## Mean : 7.191 NORTHERN :157   
## 3rd Qu.:11.000 NORTHEASTERN: 72   
## Max. :14.000 EASTERN : 67   
## (Other) :145   
## Location.1   
## 1101 RUSSELL ST\nBaltimore, MD: 9   
## 201 PRATT ST\nBaltimore, MD : 8   
## 2400 BOSTON ST\nBaltimore, MD : 8   
## 300 LIGHT ST\nBaltimore, MD : 5   
## 300 CHARLES ST\nBaltimore, MD : 4   
## 301 LIGHT ST\nBaltimore, MD : 4   
## (Other) :1289

str(data) #Entrega la clase de las variables y algunas caracteristicas del df

## 'data.frame': 1327 obs. of 6 variables:  
## $ name : Factor w/ 1277 levels "#1 CHINESE KITCHEN",..: 9 3 992 1 2 4 5 6 7 8 ...  
## $ zipCode : int 21206 21231 21224 21211 21223 21218 21205 21211 21205 21231 ...  
## $ neighborhood : Factor w/ 173 levels "Abell","Arlington",..: 53 52 18 66 104 33 98 133 98 157 ...  
## $ councilDistrict: int 2 1 1 14 9 14 13 7 13 1 ...  
## $ policeDistrict : Factor w/ 9 levels "CENTRAL","EASTERN",..: 3 6 6 4 8 3 6 4 6 6 ...  
## $ Location.1 : Factor w/ 1210 levels "1 BIDDLE ST\nBaltimore, MD",..: 835 334 554 755 492 537 505 530 507 569 ...

quantile(data$councilDistrict, na.rm = TRUE) # Me entrega los quantiles

## 0% 25% 50% 75% 100%   
## 1 2 9 11 14

table(data$zipCode, useNA = "ifany") # Entrega una tabla con las frecuencias de las observaciones (useNA = "ifany", entrega una columna extra si encuentra observaciones NA)

##   
## -21226 21201 21202 21205 21206 21207 21208 21209 21210 21211   
## 1 136 201 27 30 4 1 8 23 41   
## 21212 21213 21214 21215 21216 21217 21218 21220 21222 21223   
## 28 31 17 54 10 32 69 1 7 56   
## 21224 21225 21226 21227 21229 21230 21231 21234 21237 21239   
## 199 19 18 4 13 156 127 7 1 3   
## 21251 21287   
## 2 1

table(data$councilDistrict,data$zipCode) # Entrega una tabla de dos dimensiones con dos variables

##   
## -21226 21201 21202 21205 21206 21207 21208 21209 21210 21211 21212  
## 1 0 0 37 0 0 0 0 0 0 0 0  
## 2 0 0 0 3 27 0 0 0 0 0 0  
## 3 0 0 0 0 0 0 0 0 0 0 0  
## 4 0 0 0 0 0 0 0 0 0 0 27  
## 5 0 0 0 0 0 3 0 6 0 0 0  
## 6 0 0 0 0 0 0 0 1 19 0 0  
## 7 0 0 0 0 0 0 0 1 0 27 0  
## 8 0 0 0 0 0 1 0 0 0 0 0  
## 9 0 1 0 0 0 0 0 0 0 0 0  
## 10 1 0 1 0 0 0 0 0 0 0 0  
## 11 0 115 139 0 0 0 1 0 0 0 1  
## 12 0 20 24 4 0 0 0 0 0 0 0  
## 13 0 0 0 20 3 0 0 0 0 0 0  
## 14 0 0 0 0 0 0 0 0 4 14 0  
##   
## 21213 21214 21215 21216 21217 21218 21220 21222 21223 21224 21225  
## 1 2 0 0 0 0 0 0 7 0 140 1  
## 2 0 0 0 0 0 0 0 0 0 54 0  
## 3 2 17 0 0 0 3 0 0 0 0 0  
## 4 0 0 0 0 0 0 0 0 0 0 0  
## 5 0 0 31 0 0 0 0 0 0 0 0  
## 6 0 0 15 1 0 0 0 0 0 0 0  
## 7 0 0 6 7 15 6 0 0 0 0 0  
## 8 0 0 0 0 0 0 0 0 2 0 0  
## 9 0 0 0 2 8 0 0 0 53 0 0  
## 10 0 0 0 0 0 0 1 0 0 0 18  
## 11 0 0 0 0 9 0 0 0 1 0 0  
## 12 13 0 0 0 0 26 0 0 0 0 0  
## 13 13 0 1 0 0 0 0 0 0 5 0  
## 14 1 0 1 0 0 34 0 0 0 0 0  
##   
## 21226 21227 21229 21230 21231 21234 21237 21239 21251 21287  
## 1 0 0 0 1 124 0 0 0 0 0  
## 2 0 0 0 0 0 0 1 0 0 0  
## 3 0 1 0 0 0 7 0 0 2 0  
## 4 0 0 0 0 0 0 0 3 0 0  
## 5 0 0 0 0 0 0 0 0 0 0  
## 6 0 0 0 0 0 0 0 0 0 0  
## 7 0 0 0 0 0 0 0 0 0 0  
## 8 0 2 13 0 0 0 0 0 0 0  
## 9 0 0 0 11 0 0 0 0 0 0  
## 10 18 0 0 133 0 0 0 0 0 0  
## 11 0 0 0 11 0 0 0 0 0 0  
## 12 0 0 0 0 2 0 0 0 0 0  
## 13 0 1 0 0 1 0 0 0 0 1  
## 14 0 0 0 0 0 0 0 0 0 0

sum(is.na(data$councilDistrict)) # Verifica si existen NA y los suma, en este caso no existe (Si hay NA is.na entrega un TRUE, TRUE se interpreta como 1)

## [1] 0

any(is.na(data$councilDistrict)) # Pregunta si existe algun dato NA, any verifica si existe algun TRUE

## [1] FALSE

all(data$zipCode > 0) #Verifica que todas las observaciones de esa variables son mayores que 0, si alguna es menor que 0 entrega FALSE

## [1] FALSE

colSums(is.na(data)) # Suma por cada columna la cantidad de NA (se puede hacer por filas tambien "rowSums)

## name zipCode neighborhood councilDistrict   
## 0 0 0 0   
## policeDistrict Location.1   
## 0 0

table(data$zipCode %in% c("21212","21213")) #Permite encontrar cuantas observaciones son parte de esos dos strings

##   
## FALSE TRUE   
## 1268 59

data[data$zipCode %in% c("21212","21213"),] #Con esto puedo crear un subset que contiene solo las filas que pertenencen al character

## name zipCode  
## 29 BAY ATLANTIC CLUB 21212  
## 39 BERMUDA BAR 21213  
## 92 ATWATER'S 21212  
## 111 BALTIMORE ESTONIAN SOCIETY 21213  
## 187 CAFE ZEN 21212  
## 220 CERIELLO FINE FOODS 21212  
## 266 CLIFTON PARK GOLF COURSE SNACK BAR 21213  
## 276 CLUB HOUSE BAR & GRILL 21213  
## 289 CLUBHOUSE BAR & GRILL 21213  
## 291 COCKY LOU'S 21213  
## 362 DREAM TAVERN, CARRIBEAN U.S.A. 21213  
## 373 DUNKIN DONUTS 21212  
## 383 EASTSIDE SPORTS SOCIAL CLUB 21213  
## 417 FIELDS OLD TRAIL 21212  
## 475 GRAND CRU 21212  
## 545 RANDY'S BAR 21213  
## 604 MURPHY'S NEIGHBORHOOD BAR & GRILL 21212  
## 616 NEOPOL 21212  
## 620 NEW CLUB THUNDERBIRD INC. 21213  
## 626 NEW MAYFIELD, INC. 21213  
## 678 IKAN SEAFOOD 21212  
## 711 KAY-CEE CLUB 21212  
## 763 LA'RAE 21213  
## 777 LEMONGRASS BALTIMORE 21213  
## 779 LEN'S SANDWICH SHOP 21213  
## 845 MCDONALD'S 21213  
## 852 MCDONALD'S 21212  
## 873 NEW REX LIQUORS,INC. 21212  
## 895 OK TAVERN 21213  
## 919 PANERA BREAD 21212  
## 940 PEIWEI ASIAN DINER 21212  
## 949 PERGUSA ENTERPRISES 21212  
## 957 PHANTOM'S BAR AND GRILL 21213  
## 976 POPEYES FAMOUS FRIED CHICKEN 21212  
## 994 ROBBIE'S NEST 21213  
## 1017 RUTLAND BAR 21213  
## 1018 RYAN'S DAUGHTER 21212  
## 1022 saigon remembered restaurant 21212  
## 1053 SHIRLEY'S HONEY HOLE 21213  
## 1120 STEEPLE CHASE II 21213  
## 1122 SUBWAY 21213  
## 1153 TAM-TAM 21212  
## 1155 TASTE 21212  
## 1159 TAYLORS EAST 21213  
## 1186 THE EDGE BAR & LOUNGE 21213  
## 1187 THE EDGE BAR & LOUNGE - KITCHEN AREA 21213  
## 1198 THE HOLLOW BAR & GRILL 21212  
## 1209 THE NEW BUCKETT'S LOUNGE 21213  
## 1232 THREE ACE'S 21213  
## 1246 TORAIN'S HIDE-A-WAY 21213  
## 1259 TSUNAMI BALTIMORE 21213  
## 1287 VITO'S PIZZA 21212  
## 1298 WENDY'S OLD FASHIONED HAMBURGERS #96 21212  
## 1304 WHITTEN'S (4502-04) 21213  
## 1312 wozi lounge 21212  
## 1319 YETI RESTAURANT & CARRYOUT 21212  
## 1320 YORK CLUB TAVERN 21212  
## 1323 ZEN WEST ROADSIDE CANTINA 21212  
## 1325 ZINK'S CAFÂ\220 21213  
## neighborhood councilDistrict policeDistrict  
## 29 Downtown 11 CENTRAL  
## 39 Broadway East 12 EASTERN  
## 92 Chinquapin Park-Belvedere 4 NORTHERN  
## 111 South Clifton Park 12 EASTERN  
## 187 Rosebank 4 NORTHERN  
## 220 Chinquapin Park-Belvedere 4 NORTHERN  
## 266 Darley Park 14 NORTHEASTERN  
## 276 Orangeville Industrial Area 13 EASTERN  
## 289 Orangeville Industrial Area 13 EASTERN  
## 291 Broadway East 12 EASTERN  
## 362 Broadway East 13 EASTERN  
## 373 Homeland 4 NORTHERN  
## 383 Broadway East 13 EASTERN  
## 417 Mid-Govans 4 NORTHERN  
## 475 Chinquapin Park-Belvedere 4 NORTHERN  
## 545 Broadway East 12 EASTERN  
## 604 Mid-Govans 4 NORTHERN  
## 616 Chinquapin Park-Belvedere 4 NORTHERN  
## 620 Middle East 13 EASTERN  
## 626 Belair-Edison 13 NORTHEASTERN  
## 678 Chinquapin Park-Belvedere 4 NORTHERN  
## 711 Homeland 4 NORTHERN  
## 763 Oliver 12 EASTERN  
## 777 Little Italy 1 SOUTHEASTERN  
## 779 Broadway East 12 EASTERN  
## 845 South Clifton Park 12 EASTERN  
## 852 Radnor-Winston 4 NORTHERN  
## 873 Wilson Park 4 NORTHERN  
## 895 Biddle Street 13 EASTERN  
## 919 Lake Walker 4 NORTHERN  
## 940 Cedarcroft 4 NORTHERN  
## 949 Rosebank 4 NORTHERN  
## 957 Belair-Edison 3 NORTHEASTERN  
## 976 Winston-Govans 4 NORTHERN  
## 994 Broadway East 12 EASTERN  
## 1017 Broadway East 12 EASTERN  
## 1018 Chinquapin Park-Belvedere 4 NORTHERN  
## 1022 Mid-Govans 4 NORTHERN  
## 1053 Broadway East 13 EASTERN  
## 1120 Biddle Street 13 EASTERN  
## 1122 Oliver 12 EASTERN  
## 1153 Mid-Govans 4 NORTHERN  
## 1155 Mid-Govans 4 NORTHERN  
## 1159 Berea 13 EASTERN  
## 1186 Broadway East 12 EASTERN  
## 1187 Broadway East 12 EASTERN  
## 1198 Rosebank 4 NORTHERN  
## 1209 Broadway East 13 EASTERN  
## 1232 Belair-Edison 3 NORTHEASTERN  
## 1246 Broadway East 12 EASTERN  
## 1259 Little Italy 1 SOUTHEASTERN  
## 1287 Cedarcroft 4 NORTHERN  
## 1298 Homeland 4 NORTHERN  
## 1304 Claremont-Freedom 13 NORTHEASTERN  
## 1312 Guilford 4 NORTHERN  
## 1319 Rosebank 4 NORTHERN  
## 1320 Homeland 4 NORTHERN  
## 1323 Rosebank 4 NORTHERN  
## 1325 Belair-Edison 13 NORTHEASTERN  
## Location.1  
## 29 206 REDWOOD ST\nBaltimore, MD  
## 39 1801 NORTH AVE\nBaltimore, MD  
## 92 529 BELVEDERE AVE\nBaltimore, MD  
## 111 1932 BELAIR RD\nBaltimore, MD  
## 187 438 BELVEDERE AVE\nBaltimore, MD  
## 220 529 BELVEDERE AVE\nBaltimore, MD  
## 266 2701 ST LO DR\nBaltimore, MD  
## 276 4217 ERDMAN AVE\nBaltimore, MD  
## 289 4217 ERDMAN AVE\nBaltimore, MD  
## 291 2101 NORTH AVE\nBaltimore, MD  
## 362 2300 LAFAYETTE AVE\nBaltimore, MD  
## 373 5422 YORK RD\nBaltimore, MD  
## 383 1203 COLLINGTON AVE\nBaltimore, MD  
## 417 5723 YORK RD\nBaltimore, MD  
## 475 527 BELVEDERE AVE\nBaltimore, MD  
## 545 2135 NORTH AVE\nBaltimore, MD  
## 604 5847 YORK RD\nBaltimore, MD  
## 616 529 BELVEDERE AVE\nBaltimore, MD  
## 620 2201 CHASE ST\nBaltimore, MD  
## 626 3349 BELAIR RD\nBaltimore, MD  
## 678 529 BELVEDERE AVE\nBaltimore, MD  
## 711 201 HOMELAND AVE\nBaltimore, MD  
## 763 1000 HOFFMAN ST\nBaltimore, MD  
## 777 1300 BANK STREET\nBaltimore, MD  
## 779 1500 WASHINGTON ST\nBaltimore, MD  
## 845 2001 BROADWAY\nBaltimore, MD  
## 852 5100 YORK RD\nBaltimore, MD  
## 873 4637 YORK RD\nBaltimore, MD  
## 895 2301 BIDDLE ST\nBaltimore, MD  
## 919 6307 1 2 YORK RD\nBaltimore, MD  
## 940 6302 YORK RD\nBaltimore, MD  
## 949 5928 YORK RD\nBaltimore, MD  
## 957 3539 BELAIR RD\nBaltimore, MD  
## 976 5002 YORK RD\nBaltimore, MD  
## 994 2250 NORTH AVE\nBaltimore, MD  
## 1017 1508 RUTLAND AVE\nBaltimore, MD  
## 1018 600 BELVEDERE AVE\nBaltimore, MD  
## 1022 5857 york rd\nBaltimore, MD  
## 1053 2300 OLIVER ST\nBaltimore, MD  
## 1120 2401 CHASE ST\nBaltimore, MD  
## 1122 1400 NORTH AVE\nBaltimore, MD  
## 1153 5722 YORK RD\nBaltimore, MD  
## 1155 510 BELVEDERE AVE\nBaltimore, MD  
## 1159 1201 POTOMAC ST\nBaltimore, MD  
## 1186 2015 FEDERAL ST\nBaltimore, MD  
## 1187 2015 FEDERAL ST\nBaltimore, MD  
## 1198 5921 YORK RD\nBaltimore, MD  
## 1209 1432 CHESTER ST\nBaltimore, MD  
## 1232 3534 belair RD\nBaltimore, MD  
## 1246 1701 ELLSWORTH ST\nBaltimore, MD  
## 1259 1300 BANK ST\nBaltimore, MD  
## 1287 6304 YORK RD\nBaltimore, MD  
## 1298 5615 YORK RD\nBaltimore, MD  
## 1304 4502 ERDMAN AVE\nBaltimore, MD  
## 1312 4515 YORK RD\nBaltimore, MD  
## 1319 5926 YORK RD\nBaltimore, MD  
## 1320 5407 YORK RD\nBaltimore, MD  
## 1323 5916 YORK RD\nBaltimore, MD  
## 1325 3300 LAWNVIEW AVE\nBaltimore, MD

data("UCBAdmissions")  
df <- as.data.frame(UCBAdmissions)  
summary(df)

## Admit Gender Dept Freq   
## Admitted:12 Male :12 A:4 Min. : 8.0   
## Rejected:12 Female:12 B:4 1st Qu.: 80.0   
## C:4 Median :170.0   
## D:4 Mean :188.6   
## E:4 3rd Qu.:302.5   
## F:4 Max. :512.0

xtabs(Freq ~ Gender + Admit , data = df) # Crea una tabla cruzada con el centro "Freq" y de variables Gender y Admit

## Admit  
## Gender Admitted Rejected  
## Male 1198 1493  
## Female 557 1278

data("warpbreaks")  
df1 <- as.data.frame(warpbreaks)  
warpbreaks$replicate <- rep(1:9, len = 54)  
xt <- xtabs(breaks ~ . , data = warpbreaks) # Le entrego todas las variables me hace todas las tablas posibles  
ftable(xt) #Encuentro un resumen de lo anterior

## replicate 1 2 3 4 5 6 7 8 9  
## wool tension   
## A L 26 30 54 25 70 52 51 26 67  
## M 18 21 29 17 12 18 35 30 36  
## H 36 21 24 18 10 43 28 15 26  
## B L 27 14 29 19 29 31 41 20 44  
## M 42 26 19 16 39 28 21 39 29  
## H 20 21 24 17 13 15 15 16 28

# Creating New Variables

data <- read.csv("Restaurants.csv")  
s1 <- seq(1,10,by=2) ; s1 #Crea una secuencia de numeros del 1 al 10 cada dos

## [1] 1 3 5 7 9

s2 <- seq(1,10, lenght = 3); s2 #Crea una secuencia de exactamente 3 numeros

## Warning: In seq.default(1, 10, lenght = 3) :  
## extra argument 'lenght' will be disregarded

## [1] 1 2 3 4 5 6 7 8 9 10

x <- c(1,3,8,25,100); seq(along = x) #Crea una secuencia empezando por 1 del mismo largo que el vector

## [1] 1 2 3 4 5

data$nearme <- data$neighborhood %in% c("Roland Park", "Homeland") #Con esto he creadro una variable que indica cuales restaurantes esta cerca mio segun los vecindarios que he espicificado  
table(data$nearme)

##   
## FALSE TRUE   
## 1314 13

data$zipwrong <- ifelse(data$zipCode < 0,TRUE,FALSE) # Crea una nueva variable si el zip code es menor que 0  
table(data$zipwrong,data$zipCode<0)

##   
## FALSE TRUE  
## FALSE 1326 0  
## TRUE 0 1

data$zipgroups <- cut(data$zipCode,breaks = quantile(data$zipCode)) #Divide en los cuartiles los zipcode  
table(data$zipgroups)

##   
## (-2.123e+04,2.12e+04] (2.12e+04,2.122e+04] (2.122e+04,2.123e+04]   
## 337 375 282   
## (2.123e+04,2.129e+04]   
## 332

table(data$zipgroups,data$zipCode)

##   
## -21226 21201 21202 21205 21206 21207 21208 21209  
## (-2.123e+04,2.12e+04] 0 136 201 0 0 0 0 0  
## (2.12e+04,2.122e+04] 0 0 0 27 30 4 1 8  
## (2.122e+04,2.123e+04] 0 0 0 0 0 0 0 0  
## (2.123e+04,2.129e+04] 0 0 0 0 0 0 0 0  
##   
## 21210 21211 21212 21213 21214 21215 21216 21217  
## (-2.123e+04,2.12e+04] 0 0 0 0 0 0 0 0  
## (2.12e+04,2.122e+04] 23 41 28 31 17 54 10 32  
## (2.122e+04,2.123e+04] 0 0 0 0 0 0 0 0  
## (2.123e+04,2.129e+04] 0 0 0 0 0 0 0 0  
##   
## 21218 21220 21222 21223 21224 21225 21226 21227  
## (-2.123e+04,2.12e+04] 0 0 0 0 0 0 0 0  
## (2.12e+04,2.122e+04] 69 0 0 0 0 0 0 0  
## (2.122e+04,2.123e+04] 0 1 7 56 199 19 0 0  
## (2.123e+04,2.129e+04] 0 0 0 0 0 0 18 4  
##   
## 21229 21230 21231 21234 21237 21239 21251 21287  
## (-2.123e+04,2.12e+04] 0 0 0 0 0 0 0 0  
## (2.12e+04,2.122e+04] 0 0 0 0 0 0 0 0  
## (2.122e+04,2.123e+04] 0 0 0 0 0 0 0 0  
## (2.123e+04,2.129e+04] 13 156 127 7 1 3 2 1

library(Hmisc)

## Warning: package 'Hmisc' was built under R version 3.6.1

## Loading required package: lattice

## Loading required package: survival

## Loading required package: Formula

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.6.1

##   
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:plyr':  
##   
## is.discrete, summarize

## The following objects are masked from 'package:base':  
##   
## format.pval, units

data$zipgroups <- cut2(data$zipCode,g=4) # Divide los zip code en 4 grupos  
table(data$zipgroups)

##   
## [-21226,21205) [ 21205,21220) [ 21220,21227) [ 21227,21287]   
## 338 375 300 314

data$zcf <- factor(data$zipCode) ; data$zcf[1:10] # Convierte una variable que es integer en un factor, asi se pueden realizar regresiones lineales por ejemplo donde no necesariamente "mas zipcode" implica algo mejor

## [1] 21206 21231 21224 21211 21223 21218 21205 21211 21205 21231  
## 32 Levels: -21226 21201 21202 21205 21206 21207 21208 21209 ... 21287

class(data$zcf)

## [1] "factor"

library(plyr)  
data2 <- mutate(data, zipGroups=cut2(zipCode,g=4))

# Reshaping Data

library(reshape2)  
library(datasets)  
data("mtcars")  
head(mtcars)

## mpg cyl disp hp drat wt qsec vs am gear carb  
## Mazda RX4 21.0 6 160 110 3.90 2.620 16.46 0 1 4 4  
## Mazda RX4 Wag 21.0 6 160 110 3.90 2.875 17.02 0 1 4 4  
## Datsun 710 22.8 4 108 93 3.85 2.320 18.61 1 1 4 1  
## Hornet 4 Drive 21.4 6 258 110 3.08 3.215 19.44 1 0 3 1  
## Hornet Sportabout 18.7 8 360 175 3.15 3.440 17.02 0 0 3 2  
## Valiant 18.1 6 225 105 2.76 3.460 20.22 1 0 3 1

mtcars$carname <- rownames(mtcars)  
carMelt <- melt(mtcars, id=c("carname","gear","cyl"), measure.cars = c("mpg","hp"))  
cylData <- dcast(carMelt, cyl ~ variable) ; cylData

## Aggregation function missing: defaulting to length

## cyl mpg disp hp drat wt qsec vs am carb  
## 1 4 11 11 11 11 11 11 11 11 11  
## 2 6 7 7 7 7 7 7 7 7 7  
## 3 8 14 14 14 14 14 14 14 14 14

#ddply() # Se puede splitear un dataframe, mandarle una funcion y despues lo deja como un dataframe.

# Dplyr

library(dplyr)

## Warning: package 'dplyr' was built under R version 3.6.1

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:Hmisc':  
##   
## src, summarize

## The following objects are masked from 'package:plyr':  
##   
## arrange, count, desc, failwith, id, mutate, rename, summarise,  
## summarize

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

chicago <- readRDS("chicago.rds")  
head(select(chicago, city:dptp)) # Select me permite utilizar los nombres de las variables y no sus indices

## city tmpd dptp  
## 1 chic 31.5 31.500  
## 2 chic 33.0 29.875  
## 3 chic 33.0 27.375  
## 4 chic 29.0 28.625  
## 5 chic 32.0 28.875  
## 6 chic 40.0 35.125

head(select(chicago, -(city:dptp))) # Puedo seleccionar todas menos la que le indico -()

## date pm25tmean2 pm10tmean2 o3tmean2 no2tmean2  
## 1 1987-01-01 NA 34.00000 4.250000 19.98810  
## 2 1987-01-02 NA NA 3.304348 23.19099  
## 3 1987-01-03 NA 34.16667 3.333333 23.81548  
## 4 1987-01-04 NA 47.00000 4.375000 30.43452  
## 5 1987-01-05 NA NA 4.750000 30.33333  
## 6 1987-01-06 NA 48.00000 5.833333 25.77233

chic.f <- filter(chicago, chicago$pm25tmean2 > 30) # Filtrar con una condicion  
chic.f <- filter(chicago, chicago$pm25tmean2 > 30 & chicago$tmpd > 80) # Mas de una condicion  
chicago <- arrange(chicago,date) # Puedo organizar por alguna variable  
chicago <- rename(chicago, pm25 = pm25tmean2, dewpoint = dptp) #Renombra una variable  
chicago <- mutate(chicago, pm25detrend = pm25-mean(pm25, na.rm = TRUE)) #Permite crear otra variable con informacion de alguna existente  
chicago <- mutate(chicago, tempcat = factor(1\*(tmpd >80), labels = c("cold","hot")))  
hot\_cold <- group\_by(chicago,tempcat)

## Warning: Factor `tempcat` contains implicit NA, consider using  
## `forcats::fct\_explicit\_na`

summarize(hot\_cold, pm25 = mean(pm25, na.rm = TRUE), o3 = max(o3tmean2), n02 = mean(no2tmean2))

## # A tibble: 3 x 4  
## tempcat pm25 o3 n02  
## <fct> <dbl> <dbl> <dbl>  
## 1 cold 16.0 66.6 25.2  
## 2 hot 26.5 63.0 25.2  
## 3 <NA> 47.7 9.42 37.4

Week 4

Tomás

22 de noviembre de 2019

# Editing text variables

#Video 1  
  
data <- read.csv("Baltimore\_Fixed\_Speed\_Cameras.csv")  
names(data)

## [1] "address" "direction" "street" "crossStreet"   
## [5] "intersection" "Location.1"

tolower(names(data)) # Deja todo en minuscula (se puede en mayuscula)

## [1] "address" "direction" "street" "crossstreet"   
## [5] "intersection" "location.1"

splitnames <- strsplit(names(data), "\\.") #splitea un string  
splitnames[[6]][1]

## [1] "Location"

review <- data.frame( hola\_chao = "hola", chao\_hola = "chao")  
names(review)# Quiero quitar los "\_"

## [1] "hola\_chao" "chao\_hola"

sub("\_","",names(review)) #Quita los "\_" y los cambia por nada "" (gsub() reemplaza todos y no solo el primero)

## [1] "holachao" "chaohola"

grep("Alameda", data$intersection) # Busca las coincidencias y me entrega los indices

## [1] 65 69 79

grep("Alameda", data$intersection, value = TRUE) # Busca las coincidencias y me entrega los valores

## [1] "E 33rd & The Alameda" "The Alameda & 33rd St"   
## [3] "Harford \n & The Alameda"

table(grepl("Alameda", data$intersection)) # Entre el numero de veces que aparece "Alameda"

##   
## FALSE TRUE   
## 77 3

library (stringr)  
nchar("Jeffrey Leek") #Me da la cantidad de caracteres

## [1] 12

substr("Jefrey Leek",1,7) #Me da los caracteresd del 1 al 7

## [1] "Jefrey "

paste("hola, Chao") #Pega strings

## [1] "hola, Chao"

paste0("hola","chao") #Pega sin espacios

## [1] "holachao"

str\_trim("Tomas ") #Quita los espacios innecesarios

## [1] "Tomas"

# Video 2

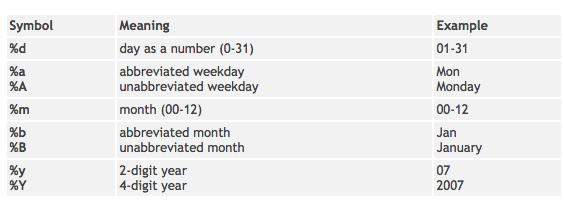
## Metacaracteres

Al buscar caracteres dentro de un texto puedo querer buscarlos de forma mas especifica que simplemente por la palabra (Funcion grep()):

* ^Hola: Me busca todas las lineas que comienzen con Hola (Si esta en medio no lo identifica)
* Hola$: Busca todas las frases que terminen con Hola
* [Bb][Uu][Ss][Hh]: Busca la palabra “Bush” en todos sus formatos, independiente si es mayuscula o minuscula
* [[1]](#footnote-1)[a-zA-Z]: Busca linea que comienzen con un numero del 0-9 y sigan con letras de la “a” a la “z”
* [^?.]$: Busca linea que no terminen en “?” ni “.”
* 9.11: Busca cualquier frase que tenga un 9 seguido de cualquier caracter y despues un 11
* flood|fire: Busca cualquiera de las dos palabras (pueden ser infinitos |)
* ^([Gg]ood|[Bb]ad): Busca la palabra good o bad en un principio (El parentesis hace que “^” sea para ambos)
* ^Hola (.)?: Me busca la palabra Hola en el inicio y como alternativa el punto (Se pone el slash debido a que el punto simboliza cualquier metacaracter pero quiero que me busque el punto
* (.\*): Me busca cualquier cosas dentro de un parentesis repetido las veces que quiera
* [Bb](+(%5E%20)+%20+){1,5} debate: Busca la palabra bush, despues un espacio, seguido de algo que no es un espacio, otro espacio y la palabra debate que este separado por 1 a 5 veces esa combinacion

# 

# Video 3



Formato de Fechas

date <- date()  
class(date)

## [1] "character"

date1 <- Sys.Date()  
class(date1)

## [1] "Date"

format(date1, "%A %B %y") # Convierte la fecha a un formato que el programa entiende

## [1] "viernes noviembre 19"

x <- c("1junio1960","2junio1960") ; x

## [1] "1junio1960" "2junio1960"

y <- as.Date(x,"%d%b%Y") ; y

## [1] "1960-06-01" "1960-06-02"

y[1] - y[2]

## Time difference of -1 days

weekdays(date1) # Dia

## [1] "viernes"

months(date1) #Mes

## [1] "noviembre"

julian(date1) #Me entrega los dias desde el "origen"

## [1] 18222  
## attr(,"origin")  
## [1] "1970-01-01"

library(lubridate)

##   
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':  
##   
## date

#Solo sirve para pasar numeros a fechas  
ymd("20140108") #Convierte a fecha en el orden que le entrego (year,month,day)

## [1] "2014-01-08"

myd("11/2014.11") #(month,year,day), Le puedo entregar casi cualquier cosa entre medio

## [1] "2014-11-11"

ymd\_hms("20140108 10:15:03") #Puedo agregarle la hora

## [1] "2014-01-08 10:15:03 UTC"

ymd\_hms("20140108 10:15:03", tz="Pacific/Auckland") #Puedo agregarle la hora y la zona horaria

## [1] "2014-01-08 10:15:03 NZDT"

1. 0-9 [↑](#footnote-ref-1)