Cargando datos

Carlos IvÃ¡n Espinosa

10 de octubre de 2018

Contents

Pueden descargar este documento en pdf haciendo clic aquí

Leyendo Datos

Vamos a leer unos datos almacenados en formato csv. Existen varias formas para leer datos desde un archivo txt, csv o xls. Empezaremos con los primeros formatos. Antes de nada necesitamos los datos con los que vamos a trabajar los podemos encontrar aquí en formato csv y aquí en formato xlsx. Descargamos los archivos y los ponemos en la carpeta del proyecto. Para descargar los archivos csv hacer clic en **RAW**

1. El primer paso es saber si nuestro archivo está en la ubicación del directorio de mi proyecto, para esto utilizaremos la función **dir()**

Teclea en tu consola esta función y mira si tus datos se encuentran ahí?

En la consola debería aparecer AMEBIASIS_LOJA, si no aparece significa que los datos no estan en la carpeta del proyecto generada. Confirma esto y vuelvelo a intentar.

2. Ahora a cargar los datos usando las funciones read.table() y read_excel()

Usando read.table

```
ameLoja<-read.table("AMEBIASIS_LOJA.csv", header=TRUE, sep=';')
```

Si en la consola no ha salido ningún error eso quiere decir que los datos han sido cargados correctamente. Antes de continuar supongo que hay algunas dudas con el código que acabamos de subir.

¿Que significa header=TRUE?

Bueno lo que le estamos diciendo es que la primera fila se encuentra los nombres de las variables.

*Y **sep**?

En este caso, estamos diciendo que la separación entre columnas es una coma. Al ser formato csv esto es evidente, pero usted podría tener un txt separado por tabulaciones por ejemplo, o por cualquier otro caracter.

Usando read_excel

Antes de nada necesitamos instalar el paquete `readxl` desde internet, esto se hace una única vez con la función `install.packages()`. Teclea en la consola `install.packages("readxl")`, para que corra esta función necesitas estar conectado a internet.

```
require(readxl)
```

```
## Loading required package: readxl
```

```
read_excel("AMEBIASIS_LOJA.xlsx",
                      sheet = 1, na = "NA")
```

```
## # A tibble: 3,019 x 9
##    Cantón Distrito `Dis Distribucio~ Sexo   `Edad en años` `N X`   `N Y`
##    <chr>  <chr>    <chr>             <chr>           <dbl> <chr>   <chr>
##  1 LOJA   11D01    LOJA              Hombre              1 683.88~ 957.48~
##  2 LOJA   11D01    LOJA              Hombre             13 683.88~ 957.48~
##  3 LOJA   11D01    LOJA              Hombre             14 683.88~ 957.48~
##  4 LOJA   11D01    LOJA              Hombre              2 683.88~ 957.48~
##  5 LOJA   11D01    LOJA              Hombre              2 68.989~ 956.98~
##  6 LOJA   11D01    LOJA              Hombre             22 683.88~ 957.48~
##  7 LOJA   11D01    LOJA              Hombre              3 683.88~ 957.48~
##  8 LOJA   11D01    LOJA              Hombre             30 683.88~ 957.48~
##  9 LOJA   11D01    LOJA              Hombre             36 68.989~ 956.98~
## 10 LOJA   11D01    LOJA              Hombre              4 683.88~ 957.48~
## # ... with 3,009 more rows, and 2 more variables: Consultas <dbl>,
## #   Parroquia <chr>
```

¿Qué paso?

La función se ejecutó pero solo se escribió en la consola, claro olvidamos *asignar* estos datos a un objeto, para *asignar* usamos la flecha (<-) , esta flecha debe estar precedido por el nombre del objeto.

```
library(readxl)

ameLojaE<-read_excel("AMEBIASIS_LOJA.xlsx",
                      sheet = 1, na = "NA")
```

Le he puesto a este objeto al final *E* para saber que es la tabla que he abierto desde excel, si no ponemos esto el objeto que creamos antes será sobrescrito, y R no nos avisará que se sobrescribe así que hay que tener cuidado.

Cuando leemos desde un archivo excel lo primero que debemos hacer es llamar al paquete que nos permite leer archivos excel `readxl` esto lo hacemos con la función *library* podríamos utilizar también la función *require*.

Lo siguiente es escribir el nombre del archivo, recuerde que R es sensible a las mayúsculas así que debe poner exactamente como lo muestra el nombre de su archivo. Posteriormente, le decimos en que hoja se encuentran los datos *sheet* y que debe poner en las celdas vacías *na*, en este este caso le decimos que ponga "NA"

Verificando los datos

Siempre cuando cargo unos datos es necesario asegurarnos que los datos están bien subidos, para esto utilizaremos dos funciones **head** y **str**

```
head(ameLoja)
```

```
##   Cantón Distrito Dis.Distribucion   Sexo Edad.en.años                 N.X
## 1   LOJA    11D01             LOJA Hombre            1 683.887.999.999.509
## 2   LOJA    11D01             LOJA Hombre           13 683.887.999.999.509
## 3   LOJA    11D01             LOJA Hombre           14 683.887.999.999.509
## 4   LOJA    11D01             LOJA Hombre            2 683.887.999.999.509
## 5   LOJA    11D01             LOJA Hombre            2  68.989.299.999.942
## 6   LOJA    11D01             LOJA Hombre           22 683.887.999.999.509
##                   N.Y Consultas              Parroquia
## 1 957.489.600.000.001         1           CHUQUIRIBAMBA
## 2 957.489.600.000.001         2           CHUQUIRIBAMBA
## 3 957.489.600.000.001         1           CHUQUIRIBAMBA
## 4 957.489.600.000.001         1           CHUQUIRIBAMBA
## 5 956.987.200.000.001         1 TAQUIL (MIGUEL RIOFRÍO)
## 6 957.489.600.000.001         1           CHUQUIRIBAMBA
```

Como ven esta función lo que hace es mostrarnos los seis primeros datos de cada columna. Esto es muy importante para chequear que no se haya cargado los datos de forma errónea. Ahora veamos lo que hace **str**

```
str(ameLoja)
```

```
## 'data.frame':    3019 obs. of  9 variables:
##  $ Cantón          : Factor w/ 16 levels "CALVAS","CATAMAYO",..: 7 7 7 7 7 7 7 7 7 7 ...
##  $ Distrito        : Factor w/ 9 levels "11D01","11D02",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Dis.Distribucion: Factor w/ 9 levels "CALVAS,GONZANAMA,QUILANGA",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ Sexo            : Factor w/ 2 levels "Hombre","Mujer": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Edad.en.años    : int  1 13 14 2 2 22 3 30 36 4 ...
##  $ N.X             : Factor w/ 94 levels "560.125.999.999.995",..: 56 56 56 56 50 56 56 56 50 56 ...
##  $ N.Y             : Factor w/ 94 levels "948.835.300.000.001",..: 73 73 73 73 62 73 73 73 62 73 ...
##  $ Consultas       : int  1 2 1 1 1 1 2 1 1 1 ...
##  $ Parroquia       : Factor w/ 62 levels "12 DE DICIEMBRE (CAB.EN ACHIOTES)",..: 15 15 15 15 56 15 15
```

Esta función nos permite saber las características del objeto y luego, de cada una de las variables. Como vemos tenemos diferentes tipos de variables algunas son categóricas y otras numéricas.

Seguramente se estarán preguntando que son estos datos, bueno estos datos corresponden a estadísticas de amebiasis en la provincia de Loja. Vamos a realizar un primer gráfico para conocer como la incidencia de amebiasis se distribuye en los diferentes cantones.

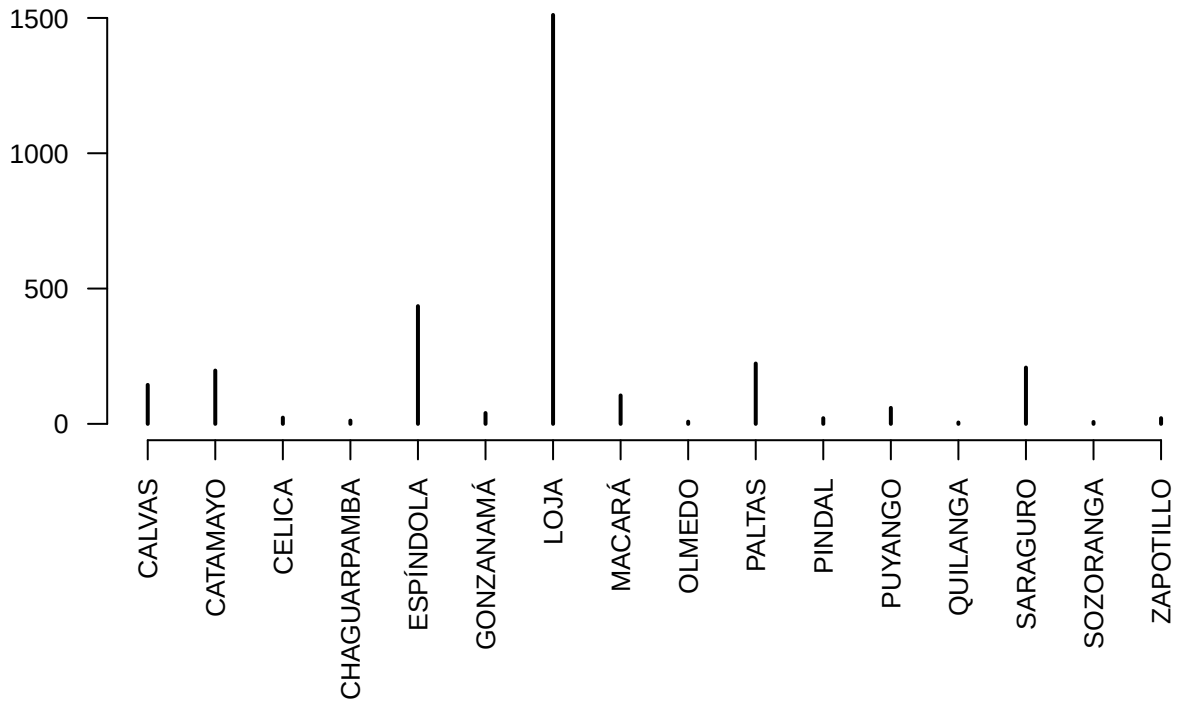
```
table(ameLoja$Cantón)
```

```
##
##       CALVAS     CATAMAYO       CELICA CHAGUARPAMBA    ESPÍNDOLA
##          144          197           23           12          435
##    GONZANAMÁ         LOJA       MACARÁ       OLMEDO       PALTAS
##           40         1511          105            8          223
```

```
##       PINDAL      PUYANGO     QUILANGA     SARAGURO    SOZORANGA
##           21           59            5          208            7
##    ZAPOTILLO
##           21
```

Creo que sería mejor verlo en un gráfico, lo que hacemos es poner plot al principio de la función table.

```
par(mar=c(9,3,2,1))
plot(table(ameLoja$Cantón), las=2, cex.axis=0.8)
```



Como podemos ver en Loja la cantidad de Amebiasis es mucho mayor que en los otros cantones.

Pero, les parece que esta conclusión de que en el Cantón Loja hay más amebiasis es correcta.

Bueno les dejo con la duda, o mejor con el trabajo.

Ustedes deben trabajar con estos datos y decir si esta conclusión es cierta.

Ejercicio 1.

Seguramente seguirán preguntándose si el gráfico de amebiasis es correcto o no (*eso espero*).

`¿Qué realmente me está diciendo el gráfico?`

Lo que realmente me dice, es la cantidad de amebiasis en cada cantón, no la incidencia de amebiasis. Vamos a trabajar. Contesten las siguientes preguntas y desarrollen los ejercicios.

1. Busque en internet la definición de incidencia y corrija los datos que hemos obtenido. Para esto debemos *asignar* la tabla en un objeto, espero recuerde como hacerlo.
2. Queremos graficar estos resultados, pero que se grafique de una forma ordenada del cantón con menos incidencia al cantón con más incidencia. Para esto utilizaremos la función **order**. Use *?* o *help()* para saber cómo puede utilizar la función order.

Con esto hemos terminado el primer ejercicio. Espero que estén satisfechos ya están trabajando en `r` han aprendido como cargar los datos usando read.table y read_xls, han visto cómo podemos utilizar funciones como; order, head y str, y hemos iniciado a trabajar con gráficas. En las siguientes lecciones nos adentraremos en funciones que nos servirán para conocer nuestros datos.