Clase 2a - Preprocesamiento de la Encuesta Permantente de Hogares

José Rodríguez de la Fuente

Tabla de contenidos

Importación de datos

Librerías

Para esta clase, necesitaremos tener instaladas las siguientes librerías:

```
install.packages("tidyverse")
install.packages("haven")
```

El formato *.csv*

La Encuesta Permanente de Hogares (**EPH**), desde la página del **INDEC** puede ser descargada, desde 2016, en formato *.txt* o *.xls*. Lo más recomendable es descargarla en *.txt*, ya que es similar al formato *.csv*. Dicho formato, *Comma Separated Values*, es uno de los más utilizados para almacenar datos tabulares: cada fila representa una observación y cada columna una variable. Los valores de las columnas están separados por comas (o punto y coma, dependiendo del idioma) y la primera fila suele contener los nombres de las variables.

Usando *R base*, podemos leer los datos a través de la función `read.csv()`. En este caso probaremos abrir la última base de la EPH disponible, descargándola del sitio web del INDEC. Es recomendable tener una carpeta de *fuentes* o *bases* en el proyecto de R para poder importar directamente la base y asignarla a un objeto.

```
eph_ind_225 <- read.csv("bases/usu_individual_T225.txt", sep = ";")
```

Otra forma de cargar los datos es a través de la librería `readr`[1]. Esta librería es parte del *tidyverse* y es una alternativa más rápida y eficiente para leer datos. Para cargar los datos con `readr`, primero llamamos a la librería *tidyverse* y luego usamos la función `read_delim()`.

```
library(tidyverse)

eph_ind_225 <- read_delim("bases/usu_individual_T225.txt", delim = ";")
```

Fijense que lo que le estamos indicando a `R` es que abra el archivo que se encuentra en la dirección `bases/usu_individual_T225.txt`, que se encuentra en la carpeta del proyecto, y que lo guarde en un objeto llamado **eph_ind_225**.

Para comprobar que la operación se realizó con éxito debemos revisar el ambiente de trabajo (*environment*). En dicha ventana, se nos indica la cantidad de observaciones (46086) y variables (235) que tiene el objeto en cuestión. Si hacemos *click* en el mismo, accederemos a la base de datos completa.

> **Dato importante**
>
> Muchas veces nos encontramos con que las bases de datos con información separada por comas o puntos y comas (;) no están en formato *.csv* sino en formato *.txt*, como es el caso de la Encuesta Permanente de Hogares del INDEC. En estos casos, podemos utilizar la misma función `read.csv()` o `read_csv()`, ya que ambas permiten especificar el separador de columnas a través del argumento `sep`. Por ejemplo, si el separador es punto y coma, podemos usar `read.csv("archivo.txt", sep = ";")` o

[1] *Machete* del paquete `readr`

`read_csv("archivo.txt", delim = ";")`.

El formato nativo: *.rds*

Otra forma de guardar y cargar datos es a través del formato nativo de `R`: **.rds**. Para guardar un objeto en este formato, usamos la función `saveRDS()` y para cargarlo, usamos `readRDS()`.

Ahora probaremos abrir una de las bases de datos de la encuesta del **Latinobarómetro** ya que desde su sitio web brindan sus datos en diversos formatos. Descarguemos el archivo del año 2023 en formato **.rds** y lo colocamos en la carpeta *bases*.

Nuevamente, para abrir este archivo debemos asignarselo a un objeto, en este caso lo llamaremos *latinobarometro*.

```
latinobarometro <- readRDS("bases/Latinobarometro_2023_Esp_Rds_v1_0.rds")
```

Si bien esto dependerá de la memoria RAM de cada computadora, abrir un archivo en formato **.rds** suele ser considerablemente más rápido que abrir un archivo en formato **.csv**, sobre todo cuando tienen una gran cantidad de casos y de variables.

Como podemos observar desde el ambiente de trabajo, la base de datos de personas tiene 19205 observaciones y 274 variables.

Abriendo otros formatos

Desde `R` también podemos abrir otros formatos comúnmente utilizados de forma sencilla. Por ejemplo, si queremos abrir un archivo en formato *.xlsx* o *.xls* podemos hacerlo a través del paquete `readxl`. Por otro lado, si queremos traer datos específicos en formato *.dta* (Stata) o *.sav* (SPSS), podemos hacerlo a través de los paquetes `haven` y `foreign`, respectivamente, sin ningún inconveniente.

Por ejemplo, descarguemos la base de Latinobarómetro nuevamente pero en formato *.sav*. La función que utilizaremos es `read_spss()` del paquete `haven` y asignaremos la base a un objeto llamado *latinobatrometro_spss*.

```
library(haven)
latinobarometro_spss <-
  ↪  read_spss("bases/Latinobarometro_2023_Esp_Spss_v1_0.sav")
```

Trayendo datos desde la web

Otra de las potencialidades que ofrecen estas funciones es que permiten leer datos directamente desde alguna URL de la web. Por ejemplo, hagamos una prueba importando alguna base que se encuentre disponible en el repositorio de datos abiertos de Argentina.

Probemos con la Encuesta Nacional de Consumos Culturales de 2017. Para ello, utilizaremos la función `read_csv()` del paquete `readr` y le pasaremos como argumento la URL del archivo.

```
cc2017 <-
  ↪  read_csv("https://datos.cultura.gob.ar/dataset/251c2ac2-e670-451c-9dbf-a4212af225b5/reso
```

Show 10 entries Search:

id	pondera_dom	fecha	region	sexo	edad	p1	p1otros	p2	p2_1	p2_1otro	p2_2	p3	p4	p5	p6horas	p6minutos	horas_radio	minutos_radio	p7_1	p7_2	p7_3	p7_4	p7_5	p8a
1	8608	23/5/2017	CENTRO	Varón	19	LA FORMA DE PENSAR, IDEOLOGIA, VALORES		SI				SI	ALGUNAS VECES AL MES	FM	1	0	1	0	ALGUNAS VECES AL MES	NUNCA	NUNCA	ALGUNAS VECES AL MES	NUNCA	MUSICALES
2	2869	10/5/2017	CENTRO	Varón	24	EL ARTE, LO CULTO		SI				NO	ALGUNOS DIAS POR SEMANA	FM	1	0	1	0	ALGUNOS DIAS POR SEMANA	ALGUNOS DIAS POR SEMANA	NUNCA	ALGUNOS DIAS POR SEMANA	NUNCA	INFORMATIVOS / NOTICIEROS
3	8765	22/5/2017	CENTRO	Varón	30	LA FORMA DE PENSAR, IDEOLOGIA, VALORES		SI				NO	ALGUNOS DIAS POR SEMANA	FM	1	30	1	30	NUNCA	TODOS O CASI TODOS LOS DIAS	NUNCA	ALGUNOS DIAS POR SEMANA	NUNCA	INFORMATIVOS / NOTICIEROS
4	12838	9/5/2017	CENTRO	Varón	53	MUSICA		SI				SI	TODOS O CASI TODOS LOS DIAS	AMBAS POR IGUAL	3	0	3	0	TODOS O CASI TODOS LOS DIAS	ALGUNOS DIAS POR SEMANA	ALGUNAS VECES AL MES	TODOS O CASI TODOS LOS DIAS	NUNCA	INFORMATIVOS / NOTICIEROS
5	20223	22/5/2017	CENTRO	Varón	70	LA FORMA DE PENSAR, IDEOLOGIA, VALORES		SI				NO	TODOS O CASI TODOS LOS DIAS	AMBAS POR IGUAL	3	0	3	0	TODOS O CASI TODOS LOS DIAS	ALGUNOS DIAS POR SEMANA	NUNCA	NUNCA	NUNCA	INFORMATIVOS / NOTICIEROS
6	5474	9/5/2017	CENTRO	Mujer	15	LA TRADICION, LA HISTORIA		SI				NO	TODOS O CASI TODOS LOS DIAS	FM	2	0	2	0	TODOS O CASI TODOS LOS DIAS	NUNCA	NUNCA	NUNCA	NUNCA	INFORMATIVOS / NOTICIEROS
7	9999	29/5/2017	CENTRO	Mujer	25	LA TRADICION, LA HISTORIA		SI				SI	TODOS O CASI TODOS LOS DIAS	FM	2	30	2	30	NUNCA	NUNCA	TODOS O CASI TODOS LOS DIAS	TODOS O CASI TODOS LOS DIAS	NUNCA	MUSICALES
8	4072	9/5/2017	CENTRO	Mujer	40	LA TRADICION, LA HISTORIA		SI				SI	TODOS O CASI TODOS LOS DIAS	FM	4	0	4	0	TODOS O CASI TODOS LOS DIAS	NUNCA	NUNCA	ALGUNAS VECES AL MES	NUNCA	INFORMATIVOS / NOTICIEROS
9	2977	10/5/2017	CENTRO	Mujer	53	TEATRO		SI				NO	TODOS O CASI TODOS LOS DIAS	FM	4	0	4	0	TODOS O CASI TODOS LOS DIAS	TODOS O CASI TODOS LOS DIAS	NUNCA	NUNCA	NUNCA	MUSICALES
10	35896	22/5/2017	CENTRO	Mujer	66	LOS MODALES, EL BUEN COMPORTAMIENTO		NO	PORQUE NO TIENE EQUIPOS DE RADIO		NO						0	0						

Showing 1 to 10 of 2,802 entries Previous 1 2 3 4 5 … 281 Next

Librerías de descarga de datos

Para algunas bases de datos, principalmente aquellas que se realizan con cierta frecuencia en el tiempo, existen paquetes específicos elaborados por usuarios o instituciones oficiales que permiten descargar los datos en forma, más o menos automática.

Algunos de estos paquetes solicitan acceso a una API (*Application Programming Interface*) para poder descargar los datos. Otros, simplemente, descargan los datos de la web. Si bien no es objeto de este seminario explorar estas alternativas, algunos ejemplos de paquetes de descarga de datos son:

- `ipumsr`: permite la manipulación y descarga de archivos del proyecto **IPUMS**, que contiene, entre otras fuentes, microdatos censales de distintos países.
- `eph`: permite la descarga de datos de la Encuesta Permanente de Hogares (EPH) de Argentina. Al mismo tiempo facilita la construcción de paneles, pools de datos y estimaciones de pobreza.
- `PNADcIBGE`: permite la descarga de datos de la Pesquisa Nacional por Amostra de Domicílios (PNAD) de Brasil.
- `WDI`: permite la descarga de datos del Banco Mundial.
- `Rilostat`: permite la descarga de datos de la Organización Internacional del Trabajo (OIT).

La librería `eph`

La librería `eph`[2] es un paquete de `R` que permite descargar los datos de la Encuesta Permanente de Hogares (EPH) de Argentina. Esta encuesta es una fuente fundamental para el estudio del mercado laboral y las condiciones de vida en el país. El paquete `eph` facilita la descarga de los datos, la construcción de paneles y pools de datos, y la estimación de pobreza.

Vamos a utilizarla para descargar algún trimestre de la EPH como ejemplo. Para ello, primero debemos instalar el paquete y luego cargarlo.

```
install.packages("eph")
```

```
library(eph)
```

Ya instalado y activado el paquete, podemos consultar en la guía de usuario del mismo las distintas funciones que tiene. Utilizaremos la función `get_microdata()` para descargar los microdatos de la EPH. Por ejemplo, si queremos descargar los datos del cuarto cuatrimestre de 2024, debemos específicar los parámetros *year*, *period* y *type*. Ya que necesitamos "guardar" la base en algún lado, vamos a asignarsela a un objeto que se llamará eph_hog_424.

```
eph_hog_424 <- get_microdata(year = 2024, period = 4, type = "hogar")
```

Si todo salió bien, en el *ambiente de trabajo* vamos a visualizar un nuevo objeto llamado *eph_hog_424*.

[2]Carolina Pradier, Guido Weksler, Pablo Tiscornia, Natsumi Shokida, Germán Rosati, & Diego Kozlowski. (2023). ropensci/eph V1.0.0 (1.0.0). Zenodo. https://doi.org/10.5281/zenodo.8352221

El paquete `eph` tiene otras funciones interesantes que permiten reconstruir los paneles (`organize_panels()`), obtener las etiquetas de las ocupaciones (`organize_cno()`), obtener las líneas de pobreza (`get_poverty_lines()`) o calcularla (`calculate_poverty()`), entre otras. Por otro lado, utilizando los parámetros que vimos podemos construir pooles de datos, personalizando los años y los trimestres. A continuación dejamos algunos ejemplos:

```
# Pool base individuos, todos los trimestres de 2020 a 2022
eph_ind_2020_2022 <- get_microdata(year = 2020:2022, period = 1:4, type =
 ↪  "individual")

# Pool base hogares, todos los trimestres de 2020 a 2022, solo segundo y
 ↪  cuarto trimestre
eph_hog_2020_2022 <- get_microdata(year = 2020:2022, period = c(2,4), type =
 ↪  "hogar")
```

Elementos de la base datos

Clases de objetos

Una vez importada la base, el siguiente paso que solemos dar es observar cómo está dispuesta la matriz de datos. En el caso de `RStudio`, a diferencia de **SPSS**, las bases de datos se mantienen ocultas en el ambiente de trabajo. Para visualizarlas, podemos hacer *click* en el objeto específico, o si queremos un resumen, podemos escribir el nombre del objeto en la consola o en el *script*.

```
eph_ind_225
```

```
# A tibble: 46,086 x 235
   CODUSU    ANO4 TRIMESTRE NRO_HOGAR COMPONENTE   H15 REGION MAS_500 AGLOMERADO
   <chr>    <dbl>     <dbl>     <dbl>      <dbl> <dbl>  <dbl> <chr>         <dbl>
 1 TQRMNOQ~  2025         2         1          2     1      1 S                32
 2 TQRMNOQ~  2025         2         1          3     1      1 S                32
 3 TQRMNOQ~  2025         2         1          1     1      1 S                32
 4 TQRMNOQ~  2025         2         1          2     1      1 S                32
 5 TQRMNOQ~  2025         2         1          3     1      1 S                32
 6 TQRMNOQ~  2025         2         2          1     1     42 N                26
 7 TQRMNOQ~  2025         2         1          1     1     42 N                26
 8 TQRMNOQ~  2025         2         1          2     0     42 N                26
 9 TQRMNOQ~  2025         2         1          3     0     42 N                26
10 TQRMNOQ~  2025         2         1          1     1     42 N                26
```

```
# i 46,076 more rows
# i 226 more variables: PONDERA <dbl>, CH03 <dbl>, CH04 <dbl>, CH05 <chr>,
#   CH06 <dbl>, CH07 <dbl>, CH08 <dbl>, CH09 <dbl>, CH10 <dbl>, CH11 <dbl>,
#   CH12 <dbl>, CH13 <dbl>, CH14 <chr>, CH15 <dbl>, CH15_COD <dbl>, CH16 <dbl>,
#   CH16_COD <dbl>, NIVEL_ED <dbl>, ESTADO <dbl>, CAT_OCUP <dbl>,
#   CAT_INAC <dbl>, IMPUTA <dbl>, PP02C1 <dbl>, PP02C2 <dbl>, PP02C3 <dbl>,
#   PP02C4 <dbl>, PP02C5 <dbl>, PP02C6 <dbl>, PP02C7 <dbl>, PP02C8 <dbl>, ...
```

En el objeto *eph_ind_225*, se puede observar la estructura tripartita del dato. En las filas se disponen las personas, en las columnas las variables y en las celdas los valores de las variables para cada persona. Estas tienen una identificación única conformada por las variables `CODUSU`, `NRO_HOGAR` y `COMPONENTE`. Esto refleja que la información se encuentra ordenada, o como se suele decir, en un formato *tidy*. Como veremos a continuación, paquetes estrella como `dplyr`, `ggplot2` y el resto de los del `tidyverse` están diseñados para trabajar con datos ordenados.

Otra cuestión relevante es que las variables pueden ser de distintos tipos. Por ejemplo, algunas variables son numéricas, otras son categóricas y otras son de texto. Para saber qué tipo de variable es cada una, podemos usar la función `class()`. Por ejemplo, la variable `CH06` (edad) es numérica. Mientras que la variable `CODUSU` es cadena, ya que almacena números y letras.

```
class(eph_ind_225$CH06)
```

```
[1] "numeric"
```

```
class(eph_ind_225$CODUSU)
```

```
[1] "character"
```

> **Atención**
>
> En la línea de código escrita recién podemos ver que para señalar a una variable en una base de datos, en `R base`, se utiliza el símbolo *$*. Esto indica que estamos apuntando a una columna determinada del objeto especificado. El autocompletado es una gran ventaja cuando necesitamos identificar alguna columna.

Al observar la clase del objeto `eph_ind_225`, podemos ver que es un *data.frame*, específicamente en formato *tibble*. Este es el tipo de objeto que se utiliza en `R` para almacenar bases de datos de estructura tabular.

```
class(eph_ind_225)
```

```
[1] "spec_tbl_df" "tbl_df"      "tbl"         "data.frame"
```

Los factores

En R, las variables categóricas son conocidas como *factores* y tienen la particularidad de tener un conjunto conocido de valores posibles y sobre los cuales puede establecerse un orden. Su diferenciación, respecto a otros tipos de objetos como los *character* o *numeric*, radica en que algunos análisis estadísticos más avanzados requieren que las variables categóricas sean tratadas como factores.

Por ejemplo, en la EPH contamos con la variable `REGION`, que indica la región en la que vive la persona encuestada. Si bien la variable fue cargada como numérica, sabemos que, en realidad, es una variable categórica. Para convertirla en factor, usamos la función `factor()`, en donde, además, podemos declarar las distintas etiquetas (*labels*) que tienen las categorías. Para no sobreescribir la variable original, la guardamos en una nueva variable llamada `region_f`.

```
eph_ind_225$region_f <- factor(eph_ind_225$REGION,
                               labels = c("Gran Buenos Aires",
                                          "Noroeste",
                                          "Noreste",
                                          "Cuyo",
                                          "Pampeana",
                                          "Patagonia"))
```

Con la función `levels()` podemos ver los niveles que tiene la variable `region_f`, es decir, como están ordenadas las categorías.

```
levels(eph_ind_225$region_f)
```

```
[1] "Gran Buenos Aires" "Noroeste"          "Noreste"
[4] "Cuyo"              "Pampeana"          "Patagonia"
```

> **Atención**
>
> Las variables de tipo *factor* son una de las principales diferencias existentes entre R y otros programas estadísticos como **SPSS** o **Stata**. En estos las variables categóricas suelen tener un valor numérico y una etiqueta asociada a cada valor, en forma de *metadata*.

En R debemos utilizar paquetes específicos como `labelled` para poder trabajar de forma similar.