ICO 187 ANÁLISIS DE DATOS
Clase 27: Análisis de datos con R

Año: 2021
Profesor: Sebastián Egaña

1. Utilización de data frames en R y Análisis de datos

Lo primero, corresponde a importar los datos en Rstudio. Considerando que ya se tiene el set de datos arriba, existen dos manera de hacer esto.

1. Insertar el código relacionado

```
data <- read_excel("clase_27.xlsx")
```

En caso de tener el archivo en una carpeta distinta se debe especificar la ruta:

```
data <-
  read_excel("G:/Mi unidad/Docencia 2020 - 2021/UST - 2021 01 Análisis de Datos/document/clase_27.xlsx")
```

Se debe tener en cuenta que la ruta en windows viene con defecto con backslash ("\"), y R necesita que las rutas utilicen slash ("/")

2. Importarlo desde el apartado de "Files" en RStudio.

Esto también produce un código que podemos recuperar.

2. Tidy

Veamos algunos procedimientos para revisar el set de datos.

1. Revisión de las variables.

En general, no se ven errores. El único caso particular es la fecha que fue almacenada como character y no como fecha. Veremos más adelante como solucionar esto.

Analicemos por ejemplo la variable parentesco. Para poder seleccionar una variable dentro de un set de datos debemo utilizar el operador "$", que incluso se informa cuando vemos en detalle el set de datos. Por ejemplo:

```
data$parentesco
```

Deberíamos ahora, revisar que tipo de parentesco se encuentra en el set de datos. Para esto podemos utilizar la función unique, que es similar a la función unicos en Excel.

```
unique(data$parentesco)
```

```
## [1] "Jefe(a) de Hogar"
```

En este caso solo existen Jefe o Jefa de hogar en el set de datos.

Uno de los problemas más grandes que teníamos al utilizar este set de datos, estaba relacionado con la relación entre hogar y persona. Recordar que por defecto, una persona solo puede estar relacionada con un hogar, y una o varias personas constituyen un hogar. Esto es algo casí obvio, pero cuando utilizamos datos parece que se nos olvida.

COnfirmemos esto:

```
unique(data$fibe)

unique(data$id_persona)
```

El problema en este caso, es que el largo del elemento que se genera. Para solucionar esto, podemos utilizar la función length, que mide el largo de un objeto.

```
length(unique(data$fibe))
```

```
## [1] 6000
```

```
length(unique(data$id_persona))
```

```
## [1] 6000
```

En este caso, tenemos una persona por hogar, y considerando que solo existe el parentesco de jefe o jefa de hogar, podemos definir que son solo los jefes o jefas de hogar presente en el set de datos.

2.1. Cambiar formato a la fecha

En este caso, utilizaremos una función de un package de R denominado dplyr. Como vimos anteriormente, debemos si es primera vez para utilizar la librería debemos instalarla y después cargarla.

```
install.packages(dplyr)
library(dplyr)
```

Una manera simple de corregir esto, es modificando la variable que ya tenemos. Para esto, se utiliza la función mutate, que me permite modificar y crear nuevas variables en el set de datos (columnas.)

Veamos un ejemplo:

```
data <- mutate(data, uno = 1)
```

A la vez, podemos incorporar múltiples variables en la misma función:

```
data <- mutate(data, uno = 1, dos = 2, tres = 3, cero = uno - 1)
```

Como carácteristica particular, se debe notar que estamos renombrando el elemnto data, por un elemento que lo incluye pero con mayor cantidad de variables. Si queremos modificar alguna variable que ya existe, solo debemos indicar el nombre de la misma y recalcular:

```r
data <- mutate(data, uno = 5)
```

Para el cambio en el formato de fecha, debemos utilizar la siguiente función:

```r
data <- mutate(data, fecha_nacimiento_2 = as.POSIXct(fecha_nacimiento, format = "%d-%m-%Y"))
```

Lo que puede ser escrito también de la siguiente manera:

```r
data <- data %>%
  mutate(fecha_nacimiento_2 = as.POSIXct(fecha_nacimiento, format = "%d-%m-%Y"))
```

Es aquí, en donde debemos introducir un operador particular de R, el pipe operator " %>% "

2.2. Pipe operator " %>% "

Nos permite indicar aplicar múltiples funciones a un mismo objeto de R sin tener que indicar el objeto en cada función.

En la clase pasada, vimos la siguiente asignación de elementos a una matriz:

```r
g <- c(0,5,0, 10,8,1,-5,-4,-1)

z <- matrix(g, byrow = FALSE)
```

Lo que usando el pipe operator se escribe de la siguiente manera:

```r
g <- c(0,5,0, 10,8,1,-5,-4,-1)

z <- g %>%
  matrix(byrow = FALSE)
```

E incluso de manera más simplificada:

```r
z <- c(0,5,0, 10,8,1,-5,-4,-1) %>%
  matrix(byrow = FALSE)
```

3. Transform

Veamos como generar una variable relacionada con la edad:

```r
data <- data %>%
  mutate(hoy = as.POSIXct("2021-04-25", format = "%Y-%m-%d"),
         edad = (hoy - fecha_nacimiento_2)/365)
```

Pregunta, ¿cómo podría usted definir si la persona es Jefe o Jefa de hogar? 0,5 décimas a quién conteste esto.

4. Evaluación sumativa

4.1. Asignación de data por grupo

Veamos si el siguiente código soluciona el problema:

```r
a <- as_tibble(c(1,2,3,4,5,6,7,8,9,10,11,12)) %>%
  rename(grupos = value) %>%
  mutate(set = sample(1:4, n(), replace = TRUE))

detalle <- a %>%
  group_by(set) %>%
  summarise(n())

detalle
```

5. Fechas Relevantes

Unidad	Evaluación	Ponderación	Fecha
Unidad I	~~Evaluación diagnóstica~~		25/03/2021
	~~Evaluación Individual Participación~~	(5 %)	05/04/2021
	~~Evaluación Grupal~~	(15 %)	27/04/2021 - 04/05/2021
	~~Evaluación Individual - Sumativa I~~	(30 %)	11/05/2021
Unidad II	~~Evaluación Grupal~~	(15 %)	20/06/2021
	~~Evaluación Grupal - Sumativa II~~	(15 %)	27/06/2021
Unidad III	Evaluación Formativa		27/06/2021
	Evaluación Grupal Sesión I- Sumativa III	(20 %)	08/07/2021
	Evaluación grupal Sesión II- Sumativa III	(20 %)	13/07/2021