05. - Data Collection_CU_04_04_secciones_y_zonas_v_01

June 8, 2023

#

CU04_Optimización de vacunas

Citizenlab Data Science Methodology > II - Data Processing Domain *** > # 05.- Data Collection

Data Collection is the process to obtain and generate (if required) necessary data to model the problem.

0.0.1 04. Secciones censales y zonas básicas de salud

Table of Contents

Settings

Data Load

ETL Processes

Import data from: CSV, Excel, Tab, JSON, SQL, and Parquet files

Synthetic Data Generation

Fake Data Generation

Open Data

Data Save

Main Conclusions

Main Actions

Acciones done

Acctions to perform

0.1 Settings

0.1.1 Encoding

Con la siguiente expresión se evitan problemas con el encoding al ejecutar el notebook. Es posible que deba ser eliminada o adaptada a la máquina en la que se ejecute el código.

[ ]:
```
Sys.setlocale(category = "LC_ALL", locale = "es_ES.UTF-8")
```

'es_ES.UTF-8/es_ES.UTF-8/es_ES.UTF-8/C/es_ES.UTF-8/C'

0.1.2 Packages to use

- {readr} from *tidyverse* for reading and writing csv files
- {tcltk} for selecting files and paths (if needed)
- {dplyr} for data exploration
- {readxl} for reading excel files
- {stringr} for manipulating strings

[27]:
```
library(readr)
library(tcltk)
library(dplyr)
library(readxl)
library(stringr)
```

0.1.3 Paths

[28]:
```
iPath <- "Data/Input/"
oPath <- "Data/Output/"
```

0.2 Data Load

If there are more than one input file, make as many sections as files to import.

1. Correspondencia secciones censales y zonas sanitarias OPCION A: Seleccionar fichero en ventana para mayor comodidad

Data load using the {tcltk} package. Ucomment the line if not using this option

[29]:
```
# file_data <- tcltk::tk_choose.files(multi = FALSE)
```

OPCION B: Especificar el nombre de archivo

Instrucciones - Los ficheros de entrada del proceso están siempre en Data/Input/.
- Si hay más de un fichero de entrada, se crean tantos objetos iFile_xx y file_data_xx como ficheros de entrada (xx número correlativo con dos dígitos, rellenar con ceros a la izquierda)

[30]:
```
iFile <- "cosalu09.xls"
file_data <- paste0(iPath, iFile)

if(file.exists(file_data)){
    cat("Se leerán datos del archivo: ", file_data)
} else{
    warning("Cuidado: el archivo no existe.")
}
```

```
Se leerán datos del archivo:  Data/Input/cosalu09.xls
```

Data file to dataframe

[31]:
```
data <- read_excel(file_data)
```

Estructura de los datos:

[32]:
```
glimpse(data)
```

```
Rows: 4,194
Columns: 3
$ `Código zona básica salud`  <chr> "1", "1", "1", "1", "1",
"1", "1", "1", "1…
$ `Literal zona básica salud` <chr> "Abrantes", "Abrantes",
"Abrantes", "Abran…
$ `Código sección censal`     <chr> "079611157", "079611158",
"079611159", "07…
```

Muestra de datos:

[33]:
```
slice_head(data, n = 5)
```

A tibble: 5 × 3

Código zona básica salud <chr>	Literal zona básica salud <chr>	Código sección censal <chr>
1	Abrantes	079611157
1	Abrantes	079611158
1	Abrantes	079611159
1	Abrantes	079611160
1	Abrantes	079611161

0.3 ETL Processes

0.3.1 Import data from: CSV, Excel, Tab, JSON, SQL, and Parquet files

Se han importado en el apartado Data Load anterior:

- Correspondencia de zonas básicas de salud y secciones censales de la Comunidad de Madrid

0.3.2 Extract data

- Filtrar datos de indicadores solo de la Comunidad de Madrid

0.3.3 Transform data

Para asegurar la trazabilidad, guardar las transformaciones en un objeto diferente con el prefijo `t`

Estandarizar tabla de correspondencia secciones

- Nombres de columnas
- Código de zona
- Códigos de municipio - distrito - sección
- Quitar información que no se va a usar

[34]:
```
tdata <- data |>
    transmute(id_zona = sprintf("%03s",
                                `Código zona básica salud`),
        nombre_zona = `Literal zona básica salud`,
```

```
        id_seccion_censal = `Código sección censal`) |>
    mutate(CMUN = str_sub(id_seccion_censal, end = 3),
           dist = str_sub(id_seccion_censal, 5, 6),
           secc = str_sub(id_seccion_censal, 7, 9)) |>
    select(-id_seccion_censal)
```

Datos transformados:

[35]:
```
glimpse(tdata)
```

```
Rows: 4,194
Columns: 5
$ id_zona     <chr> "001", "001", "001", "001", "001", "001",
"001", "001", "0…
$ nombre_zona <chr> "Abrantes", "Abrantes", "Abrantes",
"Abrantes", "Abrantes"…
$ CMUN        <chr> "079", "079", "079", "079", "079", "079",
"079", "079", "0…
$ dist        <chr> "11", "11", "11", "11", "11", "11", "11",
"11", "11", "11"…
$ secc        <chr> "157", "158", "159", "160", "161", "162",
"163", "165", "1…
```

[36]:
```
tdata |> slice_head(n = 5)
```

A tibble: 5 × 5

id_zona <chr>	nombre_zona <chr>	CMUN <chr>	dist <chr>	secc <chr>
001	Abrantes	079	11	157
001	Abrantes	079	11	158
001	Abrantes	079	11	159
001	Abrantes	079	11	160
001	Abrantes	079	11	161

0.4 Synthetic Data Generation

Estos datos no requieren tareas de este tipo.

0.5 Fake Data Generation

Estos datos no requieren tareas de este tipo.

0.6 Open Data

Los datos se han obtenido de fuentes públicas, ver apartado conclusiones.

0.7 Data Save

Este proceso, puede copiarse y repetirse en aquellas partes del notebbok que necesiten guardar datos. Recuerde cambiar la extensión añadida del fichero para diferenciarlas

Identificamos los datos a guardar

[37]:
```
data_to_save <- tdata
```

Estructura de nombre de archivos:

- Código del caso de uso, por ejemplo “CU_04”
- Número del proceso que lo genera, por ejemplo ”_05”.
- Número de la tarea que lo genera, por ejemplo ”_01”
- En caso de generarse varios ficheros en la misma tarea, llevarán _01 _02 … después
- Nombre: identificativo de “properData”, por ejemplo ”_zonasgeo”
- Extensión del archivo

Ejemplo: ”CU_04_05_01_01_zonasgeo.json, primer fichero que se genera en la tarea 01 del proceso 05 (Data Collection) para el caso de uso 04 (vacunas)

Importante mantener los guiones bajos antes de proceso, tarea, archivo y nombre

0.7.1 Proceso 05

[38]:
```
caso <- "CU_04"
proceso <- '_05'
tarea <- "_04"
archivo <- ""
proper <- "_zonas_secciones"
extension <- ".csv"
```

OPCION A: Uso del paquete “tcltk” para mayor comodidad

- Buscar carpeta, escribir nombre de archivo SIN extensión (se especifica en el código)
- Especificar sufijo2 si es necesario
- Cambiar data por data_xx si es necesario
- Descomentar si se usa esta opción

[39]:
```
# file_save <- paste0(caso, proceso, tarea, tcltk::tkgetSaveFile(), proper,␣
 ↪extension)
# path_out <- paste0(oPath, file_save)
# write_csv(data, path_out)

# cat('File saved as: ')
# path_out
```

OPCION B: Especificar el nombre de archivo

- Los ficheros de salida del proceso van siempre a Data/Output/.

[40]:
```
file_save <- paste0(caso, proceso, tarea, archivo, proper, extension)
path_out <- paste0(oPath, file_save)
write_csv(data_to_save, path_out)

cat('File saved as: ')
```

```
path_out
```

```
File saved as:
```

'Data/Output/CU_04_05_04_zonas_secciones.csv'

Copia del fichero a Input Será usado en otros notebooks

[44]:
```
path_in <- paste0(iPath, file_save)
file.copy(path_out, path_in, overwrite = TRUE)
```

TRUE

0.8 Main Conclusions

List and describe the general conclusions of the analysis carried out.

0.8.1 Prerequisites

This working code needs the following conditions:

- For using the interactive selection of file, the {tcltk} package must be installed. It is not needed in production.
- The {readr}, {dplyr}, {readxl} and {stringr} packages must be installed. They are part of the *tidyverse*.
- The data paths `Data/Input` and `Data/Output` must exist (relative to the notebook path)

0.8.2 Configuration Management

This notebook has been tested with the following versions of R and packages. It cannot be assured that later versions work in the same way: * R 4.2.2 * tcltk 4.2.2 * readr 2.1.3 * dplyr 1.0.10 * readxl 1.4.1 * stringr 1.5.0

0.8.3 Data structures

Objeto `data` (correspondencia secciones censales y zonas de salud)

- Hay 4194 secciones censales asignadas a zonas sanitarias
- La sección censal se compone de: MMMMDDSSS (código de municipio de 4 dígitos, código de distrito y código de sección censal). Los datos del INE están por código de municipio de 3 dígitos, que son los tres primeros del código de 4 dígitos.
- El código de zona básica viene en numérico, mientras que el archivo de zonas viene en cadena de tres caracteres con ceros a la izquierda.

0.8.4 Consideraciones para despliegue en piloto

- Datos obtenidos de https://www.madrid.org/iestadis/fijas/clasificaciones/cozousalu.htm. Archivo `cosalu09.xls`: Zonificación de salud de la Comunidad de Madrid. Se ha descargado a la carpeta `Data/Input`
- Si la información geográfica cambia, se debería actualizar el fichero y volver a ejecutar todos los procesos.

0.8.5 Consideraciones para despliegue en producción

- Se deben crear los procesos ETL en producción necesarios para que los datos de entrada estén actualizados

0.9 Main Actions

Acciones done Indicate the actions that have been carried out in this process

- Se han transformado los códigos de zonas sanitarias para poder unir las tablas posteriormente
- Se han estandarizado los códigos de municipio, distrito y sección censal

Acctions to perform Indicate the actions that must be carried out in subsequent processes

- Se debe asignar a cada sección censal su zona de salud.
- Se deben calcular los indicadores del INE por zona de salud

0.10 CODE TO DEPLOY (PILOT)

A continuación se incluirá el código que deba ser llevado a despliegue para producción, dado que se entiende efectúa operaciones necesarias sobre los datos en la ejecución del prototipo

Description

- No hay nada que desplegar en el piloto, ya que estos datos son estáticos o en todo caso cambian con muy poca frecuencia, altamente improbable durante el proyecto.

CODE

[42]:
```
# incluir código
```