05. - Data Collection_CU_04_18_escucha_gripe_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 18. Unir datos globales por semana (sin zona)

- Búsquedas gripe y tuits

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.

[1]:
```
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

- {tcltk} para selección interactiva de archivos locales
- {readr} para leer y escribir archivos csv
- {dplyr} para explorar datos

[2]:
```
library(sf)
library(readr)
library(dplyr)
library(stringr)
library(tidyr)
```

```
Linking to GEOS 3.10.2, GDAL 3.4.2, PROJ 8.2.1; sf_use_s2() is TRUE


Attaching package: 'dplyr'


The following objects are masked from 'package:stats':

    filter, lag


The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

0.1.3 Paths

[3]:
```
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.

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)

1. Datos de twitter

OPCION A: Seleccionar fichero en ventana para mayor comodidad

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

[4]:
```
# file_data_01 <- tcltk::tk_choose.files(multi = FALSE)
```

OPCION B: Especificar el nombre de archivo

[5]:
```
iFile_01 <- "CU_04_05_12_tuits.csv"
file_data_01 <- paste0(iPath, iFile_01)

if(file.exists(file_data_01)){
    cat("Se leerán datos del archivo: ", file_data_01)
} else{
    warning("Cuidado: el archivo no existe.")
}
```

```
Se leerán datos del archivo:  Data/Input/CU_04_05_12_tuits.csv
```

Data file to dataframe Usar la función adecuada según el formato de entrada (xlsx, csv, json, ...)

[6]:
```
data_01 <- read_csv(file_data_01)
```

```
Rows: 74 Columns: 3
  Column specification

Delimiter: ","
dbl (3): ano, semana, tuits_gripe

 Use `spec()` to retrieve the full column specification for this
data.
 Specify the column types or set `show_col_types = FALSE` to quiet
this message.
```

Estructura de los datos:

[7]:
```
data_01 |> glimpse()
```

```
Rows: 74
Columns: 3
$ ano         <dbl> 2021, 2021, 2021, 2021, 2021, 2021, 2021,
2021, 2021, 2021…
$ semana      <dbl> 36, 37, 38, 39, 40, 41, 42, 43, 44, 45,
46, 47, 48, 49, 50…
$ tuits_gripe <dbl> 97, 79, 112, 143, 112, 130, 254, 190,
198, 160, 206, 280, …
```

Muestra de datos:

[8]:
```
data_01 |> slice_head(n = 5)
```

	ano <dbl>	semana <dbl>	tuits_gripe <dbl>
A spec_tbl_df: 5 x 3	2021	36	97
	2021	37	79
	2021	38	112
	2021	39	143
	2021	40	112

2. Datos de Google

OPCION A: Seleccionar fichero en ventana para mayor comodidad

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

[9]:
```
# file_data_02 <- tcltk::tk_choose.files(multi = FALSE)
```

OPCION B: Especificar el nombre de archivo

[10]:
```
iFile_02 <- "CU_04_05_13_interes_gripe.csv"
file_data_02 <- paste0(iPath, iFile_02)

if(file.exists(file_data_02)){
    cat("Se leerán datos del archivo: ", file_data_02)
} else{
    warning("Cuidado: el archivo no existe.")
}
```

```
Se leerán datos del archivo:  Data/Input/CU_04_05_13_interes_gripe.csv
```

Data file to dataframe Usar la función adecuada según el formato de entrada (xlsx, csv, json, ...)

[11]:
```
data_02 <- read_csv(file_data_02)
```

```
Rows: 74 Columns: 3
  Column specification

Delimiter: ","
dbl (3): ano, semana, interes_gripe

 Use `spec()` to retrieve the full column specification for this
data.
 Specify the column types or set `show_col_types = FALSE` to quiet
this message.
```

Estructura de los datos:

[12]:
```
data_02 |> glimpse()
```

```
Rows: 74
Columns: 3
```

```
$ ano           <dbl> 2021, 2021, 2021, 2021, 2021, 2021,
2021, 2021, 2021, 20…
$ semana        <dbl> 36, 37, 38, 39, 40, 41, 42, 43, 44, 45,
46, 47, 48, 49, …
$ interes_gripe <dbl> 13, 15, 19, 29, 38, 65, 100, 94, 63,
70, 64, 64, 57, 67,…
```

Muestra de datos:

[13]:
```
data_02 |> slice_head(n = 5)
```

A spec_tbl_df: 5 x 3

ano <dbl>	semana <dbl>	interes_gripe <dbl>
2021	36	13
2021	37	15
2021	38	19
2021	39	29
2021	40	38

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:

- Número de tuits de gripe por semana
- Interés del término gripe por semana

Incluir apartados si procede para: Extracción de datos (select, filter), Transformación de datos, (mutate, joins, …). Si es necesario tratar datos perdidos, indicarlo también en NB 09.2

Si no aplica: Estos datos no requieren tareas de este tipo.

Data transform

- Unión de los dos conjuntos de datos

[14]:
```
data <- data_01 |>
    inner_join(data_02,
        by = c("ano", "semana"))
```

[15]:
```
data |> glimpse()
```

```
Rows: 74
Columns: 4
$ ano           <dbl> 2021, 2021, 2021, 2021, 2021, 2021,
2021, 2021, 2021, 20…
$ semana        <dbl> 36, 37, 38, 39, 40, 41, 42, 43, 44, 45,
46, 47, 48, 49, …
$ tuits_gripe   <dbl> 97, 79, 112, 143, 112, 130, 254, 190,
198, 160, 206, 280…
```

```
$ interes_gripe <dbl> 13, 15, 19, 29, 38, 65, 100, 94, 63,
70, 64, 64, 57, 67,…
```

[16]:
```
data |> slice_head(n = 5)
```

A spec_tbl_df: 5 x 4

ano <dbl>	semana <dbl>	tuits_gripe <dbl>	interes_gripe <dbl>
2021	36	97	13
2021	37	79	15
2021	38	112	19
2021	39	143	29
2021	40	112	38

0.4 Synthetic Data Generation

No aplica

0.5 Fake Data Generation

No aplica

0.6 Open Data

No aplica

0.7 Data Save

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

Identificamos los datos a guardar

[17]:
```
data_to_save <- data
```

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

[18]:
```
caso <- "CU_04"
proceso <- '_05'
tarea <- "_18"
archivo <- ""
proper <- "_escucha_gripe"
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 datos por datos_xx si es necesario

[19]:
```
# file_save <- paste0(caso, proceso, tarea, tcltk::tkgetSaveFile(), proper,␣
 ↪extension)
# path_out <- paste0(oPath, file_save)
# write_csv(data_to_save_, 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/.

[20]:
```
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_18_escucha_gripe.csv'

Copia del fichero a Input Si el archivo se va a usar en otros notebooks, copiar a la carpeta Input

[21]:
```
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

Para que funcione este código se necesita:

- Las rutas de archivos `Data/Input` y `Data/Output` deben existir (relativas a la ruta del *notebook*)
- El paquete tcltk instalado para seleccionar archivos interactivamente. No se necesita en producción.
- Los paquetes readr, dplyr deben estar instalados.

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

0.8.3 Data structures

Objeto `data`

- Los datos de origen
- Hay xxx filas con las variables:
  - ano
  - semana
  - tuits_gripe
  - interes_gripe

Observaciones generales sobre los datos

- No se dispone de detalle por zonas, solo a nivel de Comunidad de Madrid

0.8.4 Consideraciones para despliegue en piloto

- Se desplegarán los datos estáticos sin actualizar en tiempo real

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
- Para los datos de Twitter es necesario tener una cuenta con los suficientes permisos

0.9 Main Actions

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

- Se han unido las tablas de tendencias de google y tuits

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

- Se deben unir los datos al fichero global con todos los datos (se repetirán para cada zona)

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

[22]:
```
# incluir código
```