05. - Data Collection_CU_55_01_gasto_comunidad_v_01

June 15, 2023

#
CU55_Modelo agregado de estimación del gasto medio por turista
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 01. Obtener datos de gasto turístico

- Gasto medio diario por turista según país de origen en la Comunidad de Madrid, trimestral.

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.

```
In [1]: Sys.setlocale(category = "LC_ALL", locale = "es_ES.UTF-8")
```

'LC_COLLATE=es_ES.UTF-8;LC_CTYPE=es_ES.UTF-8;LC_MONETARY=es_ES.UTF-8;LC_NUMERIC=C;LC_TIME=es_ES.UTF-8'

0.1.2 Packages to use

ELIMINAR O AÑADIR LO QUE TOQUE. COPIAR VERSIONES AL FINAL Y QUITAR CÓDIGO DE VERSIONES

- {tcltk} para selección interactiva de archivos locales
- {sf} para trabajar con georeferenciación
- {readr} para leer y escribir archivos csv
- {dplyr} para explorar datos
- {stringr} para manipulación de cadenas de caracteres
- {tidyr} para organización de datos

```
In [3]: library(INEbaseR)
        library(dplyr)
        library(stringr)
        library(readr)
        library(tidyr)

        p <- c("tcltk", "INEbaseR", "readr", "dplyr", "stringr", "tidyr")
```

0.1.3 Paths

```
In [4]: iPath <- "Data/Input/"
        oPath <- "Data/Output/"
```

0.2 Data Load

No aplica: los datos se descargan a través de la API del INE

0.3 ETL Processes

No aplica

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

0.4 Synthetic Data Generation

No aplica

0.5 Fake Data Generation

No aplica

0.6 Open Data

Este proceso debe tener como resultado un fichero .csv con los datos a analizar.

Se importarán datos "open" si es necesario que serán total o parcialmente parte de los datos finales.

La tabla del INE donde se encuentran estos datos es la 37672

```
In [5]: t <- 37672
```

Se obtienen las series de datos para después hacer la búsqueda y obtener sus metadatos

```
In [6]: data_01 <- get_tables(t, resource = "data") |>
          filter(str_detect(Nombre, "Total", negate = TRUE))
```

Se crea el data.frame donde se guardarán los metadatos para el caso

```
In [7]: data_02 <- data.frame(cod_ccaa = character(),
                              cod_pais = character())
```

Se descargan los metadatos de diterés de las series para el caso.

```
In [8]: series <- data_01$COD
        for(i in seq_along(series)){
          m <- get_series(series[i], resource = "metadata", tip = "M")$MetaData
          data_02[i, 1] <- m |> filter(Variable[1] == 70) |>
                   pull(Codigo)
          data_02[i, 2] <- m |> filter(Variable[1] == 96) |>
                   pull(Codigo)
          #cat(i, ": ", series[i], "\n")
        }
```

```
Warning message:
"Using one column matrices in `filter()` was deprecated in dplyr 1.1.0.
 Please use one dimensional logical vectors instead."
```

Se unen las series a los metadatos, se expanden se quitan las columnas que no hacen falta

```
In [9]: data_03 <- data_01 |>
          bind_cols(data_02) |>
          select(-c(COD:FK_Escala)) |>
          filter(cod_ccaa == 13) |>
          unnest(Data) |>
          select(-c(cod_ccaa, Secreto, Fecha, FK_TipoDato))
```

Se imputan valores perdidos

```
In [10]: tdata <- data_03 |>
           group_by(FK_Periodo) |>
           mutate(Valor = if_else(is.na(Valor), median(Valor, na.rm = TRUE), Valor)) |>
           ungroup()
```

Estructura de los datos

```
In [13]: tdata |> glimpse()
```

```
Rows: 680
Columns: 4
$ FK_Periodo <int> 22, 21, 20, 19, 22, 21, 20, 19, 22, 21, 20, 19, 22, 21, 20,
$ Anyo       <int> 2022, 2022, 2022, 2022, 2021, 2021, 2021, 2021, 2020, 2020,
$ Valor      <dbl> 101.480, 109.300, 122.010, 97.010, 98.500, 101.140, 112.930
$ cod_pais   <chr> "126", "126", "126", "126", "126", "126", "126", "126", "12
```

Muestra de los primeros datos

```
In [14]: tdata |> slice_head(n = 10)
```

A tibble: 10 Œ 4

FK_Periodo <int>	Anyo <int>	Valor <dbl>	cod_pais <chr>
22	2022	101.48	126
21	2022	109.30	126
20	2022	122.01	126
19	2022	97.01	126
22	2021	98.50	126
21	2021	101.14	126
20	2021	112.93	126
19	2021	94.18	126
22	2020	94.73	126
21	2020	114.79	126

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

```
In [15]: 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

```
In [16]: caso <- "CU_55"
         proceso <- '_05'
         tarea <- "_01"
         archivo <- ""
         proper <- "_gasto_comunidad"
         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

```
In [17]: # file_save <- paste0(caso, proceso, tarea, tcltk::tkgetSaveFile(), proper, extension)
         # path_out <- paste0(oPath, file_save)
         # write_csv(data_to_save_xxxxx, 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/.

```
In [18]: 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_55_05_01_gasto_comunidad.csv'

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

```
In [19]: 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 INEbaseR, readr, dplyr, stringr, tidyr 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 4.2.3 * INEbaseR 0.1.0 * readr 2.1.3 * dplyr 1.1.0 * stringr 1.5.0 * tidyr 1.3.0

0.8.3 Data structures

Objeto `tdata`

- Los datos de origen se han descargado directamente del INE y el código funcionaría para tenerlo actualizado a medida que llegaran datos nuevos
- Hay 680 filas con información de las siguientes variables:
  - FK_Periodo
  - Anyo
  - Valor
  - cod_pais

  Tenemos los siguientes trimestres:

```
In [20]: tdata |> count(Anyo, FK_Periodo)
```

A tibble: 20 Œ 3

Anyo <int>	FK_Periodo <int>	n <int>
2018	19	34
2018	20	34
2018	21	34
2018	22	34
2019	19	34
2019	20	34
2019	21	34
2019	22	34
2020	19	34
2020	20	34
2020	21	34
2020	22	34
2021	19	34
2021	20	34
2021	21	34
2021	22	34
2022	19	34
2022	20	34
2022	21	34
2022	22	34

Observaciones generales sobre los datos

- Solo hay datos a nivel de comundiad autónoma por trimestre

0.8.4 Consideraciones para despliegue en piloto

- No aplica, se desplegará con los datos generados en esta tarea

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 obtenido los datos de gasto medio en la CM por país y trimestre

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

- Se deben extrapolar estos datos a los datos por municipio receptor

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

```
In [21]: # incluir código
```