05. - Data Collection_CU_45_05_interno_muni_v_01

June 11, 2023

#

CU45_Planificación y promoción del destino en base a los patrones en origen de los turistas

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 05. Transformar datos de movimiento de turistas nacionales por municipio

- Generar csv consolidado de datos de número de turistas por municipio de origen

Los datos se descargan de aquí, copiados a Input:

- Descarga directa de https://ine.es/dynt3/inebase/es/index.htm?padre=8578&capsel=8579
- Archivo: exp_tmov_interno_mun_2022.xlsx

Son datos de turistas nacionales por municipio de origen y provincia de destino a partir de los datos de antenas móviles publicados en INE

Se encuentran en INEBASE pero la descarga disrecta completa no es posible por la restricción de volumen. Se puede a futuro automatizar

[3]:
```
## 53002: Número de turistas por municipio de origen y destino
# t <- 53002
# groups_id <- get_tables(t, resource = "group")
# groups_id
# s <- get_tables(t, resource = "data")
# > s
# $status
# [1] "No puede mostrarse por restricciones de volumen"
```

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.

[4]:
```
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
- {readxl} para leer archivos de Excel
- {dplyr} para explorar datos

[26]:
```
library(readr)
library(readxl)
library(dplyr)

p <- c("tcltk", "readr", "dplyr", "readxl")
```

0.1.3 Paths

[6]:
```
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)

OPCION A: Seleccionar fichero en ventana para mayor comodidad

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

[7]:
```
# file_data <- tcltk::tk_choose.files(multi = FALSE)
```

OPCION B: Especificar el nombre de archivo

[13]:
```
iFile <- "exp_tmov_interno_mun_2022.xlsx"
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/exp_tmov_interno_mun_2022.xlsx
```

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

[17]:
```
hi <- excel_sheets(file_data)[-1]

data <- lapply(hi, function(x) read_excel(file_data, sheet = x) |>␣
 ↪filter(prov_dest_cod == "28")) |>
  bind_rows()
```

Estructura de los datos:

[18]:
```
data |> glimpse()
```

```
Rows: 79,246
Columns: 10
$ mes           <chr> "2022-01", "2022-01", "2022-01",
"2022-01", "2022-01", "…
$ mun_orig_cod  <chr> "01001", "01002", "01009", "01013",
"01036", "01051", "0…
$ mun_orig      <chr> "Alegría-Dulantzi", "Amurrio",
"Asparrena", "Barrundia",…
$ dest_cod      <chr> "28079", "28079", "28079", "28079",
"28079", "28079", "2…
$ dest          <chr> "Madrid", "Madrid", "Madrid", "Madrid",
"Madrid", "Madri…
$ turistas      <dbl> 96, 67, 69, 30, 149, 49, 56, 80, 38,
38, 60, 95, 36, 69,…
$ prov_orig_cod <chr> "01", "01", "01", "01", "01", "01",
"01", "01", "01", "0…
$ prov_orig     <chr> "Araba/Álava", "Araba/Álava",
"Araba/Álava", "Araba/Álav…
$ prov_dest_cod <chr> "28", "28", "28", "28", "28", "28",
```

```
"28", "28", "28", "2…
$ prov_dest     <chr> "Madrid", "Madrid", "Madrid", "Madrid",
"Madrid", "Madri…
```

Muestra de los primeros datos:

[20]:
```
data |> slice_head(n = 5)
```

A tibble: 5 x 10

mes <chr>	mun_orig_cod <chr>	mun_orig <chr>	dest_cod <chr>	dest <chr>	turistas <dbl>	prov_orig_cod <chr>
2022-01	01001	Alegría-Dulantzi	28079	Madrid	96	01
2022-01	01002	Amurrio	28079	Madrid	67	01
2022-01	01009	Asparrena	28079	Madrid	69	01
2022-01	01013	Barrundia	28079	Madrid	30	01
2022-01	01036	Laudio/Llodio	28079	Madrid	149	01

0.3 Open Data

No aplica

0.4 ETL Processes

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

Se han importado en el apartado Data Load anterior:

- Turismo interno por municipio en la comunidad de Madrid

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 transformation

- No aplica

0.5 Synthetic Data Generation

No aplica

0.6 Fake Data Generation

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

[23]:
```
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

[21]:
```
caso <- "CU_45"
proceso <- '_05'
tarea <- "_05"
archivo <- ""
proper <- "_interno_mun"
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

[ ]:
```
# 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/.

[24]:
```
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_45_05_05_interno_mun.csv'

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

[25]:
```
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, readxl, stringr 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 1.1.0 * readxl 1.4.1

0.8.3 Data structures

Objeto `data`

- Hay 79246 filas con información de las siguientes variables:
  - mes
  - mun_orig_cod
  - mun_orig
  - dest_cod
  - dest
  - turistas
  - prov_orig_cod
  - prov_orig
  - prov_dest_cod
  - prov_dest

Observaciones generales sobre los datos

- Los datos vienen completos, no hay valores perdidos
- Los datos disponibles en el archivo utilizado son en este rango de meses

[29]:
```
data_to_save |> pull(mes) |> range()
```

1. '2022-01' 2. '2022-12'

0.8.4 Consideraciones para despliegue en piloto

- No aplica

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 guardado los datos de turismo interior por municipio de origen

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

- Se deben unir a los datos de establecimientos por municipio para los modelos

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

[ ]:
```
# incluir código
```