05. - Data Collection_CU_45_04_interno_prov_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 04. Transformar datos de movimiento de turistas nacionales por provincia

- Generar csv consolidado de datos de número de turistas por provincia 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: 53001.csv

Son datos de turistas nacionales por provincia de origen y municipio 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

[35]:
```
## 53001: Número de turistas mensuales por CCAA y provincia de origen␣
 ↪desagregados por municipio de destino
# t <- 53001
# 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.

[36]:
```
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
- {mapSpain} para obtener datos de municipios
- {readr} para leer y escribir archivos csv
- {dplyr} para explorar datos

[37]:
```
library(readr)
library(mapSpain)
library(dplyr)
library(stringr)
```

0.1.3 Paths

[38]:
```
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

[39]:
```
# file_data <- tcltk::tk_choose.files(multi = FALSE)
```

OPCION B: Especificar el nombre de archivo

[40]:
```
iFile <- "53001.csv"
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/53001.csv
```

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

[41]:
```
data_01 <- read_delim(file_data, delim = ";",
                      skip = 1,
                      escape_double = FALSE, trim_ws = TRUE,
                      col_types = "cccc-cc",
                      col_names = c("total_nacional", "total_ccaa", "provincia",␣
 ↪"municipio_destino",
                         "periodo", "turistas"))
```

Estructura de los datos:

[42]:
```
data_01 |> glimpse()
```

```
Rows: 23,420,160
Columns: 6
$ total_nacional    <chr> "Total Nacional", "Total Nacional",
"Total Nacional"…
$ total_ccaa        <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA,
NA, NA, NA, NA, …
$ provincia         <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA,
NA, NA, NA, NA, …
$ municipio_destino <chr> "Ababuj", "Ababuj", "Ababuj",
"Ababuj", "Ababuj", "A…
$ periodo           <chr> "2022M10", "2022M09", "2022M08",
"2022M07", "2022M06…
$ turistas          <chr> "46", "74", "237", "118", "148",
"86", "192", "48", …
```

Muestra de datos:

[43]:
```
data_01 |> slice_head(n = 5)
```

	total_nacional <chr>	total_ccaa <chr>	provincia <chr>	municipio_destino <chr>	periodo <chr>	turistas <chr>
A spec_tbl_df: 5 x 6	Total Nacional	NA	NA	Ababuj	2022M10	46
	Total Nacional	NA	NA	Ababuj	2022M09	74
	Total Nacional	NA	NA	Ababuj	2022M08	237
	Total Nacional	NA	NA	Ababuj	2022M07	118
	Total Nacional	NA	NA	Ababuj	2022M06	148

0.3 Open Data

Datos de municipios de {mapSpain} para obtener los códigos

[44]:
```
data_02 <- esp_get_munic()
```

[45]:
```
data_02 |> glimpse()
```

```
Rows: 8,131
Columns: 8
$ codauto       <chr> "01", "01", "01", "01", "01", "01",
"01", "01", "01", "0…
$ ine.ccaa.name <chr> "Andalucía", "Andalucía", "Andalucía",
"Andalucía", "And…
$ cpro          <chr> "04", "04", "04", "04", "04", "04",
"04", "04", "04", "0…
$ ine.prov.name <chr> "Almería", "Almería", "Almería",
"Almería", "Almería", "…
$ cmun          <chr> "001", "002", "003", "004", "005",
"006", "007", "008", …
$ name          <chr> "Abla", "Abrucena", "Adra",
"Albánchez", "Alboloduy", "A…
$ LAU_CODE      <chr> "04001", "04002", "04003", "04004",
"04005", "04006", "0…
$ geometry      <GEOMETRY [arc_degree]> POLYGON ((-2.77744
37.23836…, POLYGO…
```

[46]:
```
data_02 |> tibble() |> filter(cpro == "01")
```

codauto	ine.ccaa.name	cpro	ine.prov.name	cmun	name
<chr>	<chr>	<chr>	<chr>	<chr>	<chr>
16	País Vasco	01	Araba/Álava	001	Alegría-Dulantzi
16	País Vasco	01	Araba/Álava	002	Amurrio
16	País Vasco	01	Araba/Álava	003	Aramaio
16	País Vasco	01	Araba/Álava	004	Artziniega
16	País Vasco	01	Araba/Álava	006	Armiñón
16	País Vasco	01	Araba/Álava	008	Arratzua-Ubarrundia
16	País Vasco	01	Araba/Álava	009	Asparrena
16	País Vasco	01	Araba/Álava	010	Ayala / Aiara
16	País Vasco	01	Araba/Álava	011	Baños de Ebro / Mañueta
16	País Vasco	01	Araba/Álava	013	Barrundia
16	País Vasco	01	Araba/Álava	014	Berantevilla
16	País Vasco	01	Araba/Álava	016	Bernedo
16	País Vasco	01	Araba/Álava	017	Campezo / Kanpezu
16	País Vasco	01	Araba/Álava	018	Zigoitia
16	País Vasco	01	Araba/Álava	019	Kripan
16	País Vasco	01	Araba/Álava	020	Kuartango
16	País Vasco	01	Araba/Álava	021	Elburgo / Burgelu
16	País Vasco	01	Araba/Álava	022	Elciego
16	País Vasco	01	Araba/Álava	023	Elvillar / Bilar
16	País Vasco	01	Araba/Álava	027	Iruraiz-Gauna
16	País Vasco	01	Araba/Álava	028	Labastida / Bastida
16	País Vasco	01	Araba/Álava	030	Lagrán
16	País Vasco	01	Araba/Álava	031	Laguardia
16	País Vasco	01	Araba/Álava	032	Lanciego / Lantziego
16	País Vasco	01	Araba/Álava	033	Lapuebla de Labarca
16	País Vasco	01	Araba/Álava	034	Leza
16	País Vasco	01	Araba/Álava	036	Laudio / Llodio
16	País Vasco	01	Araba/Álava	037	Arraia-Maeztu
16	País Vasco	01	Araba/Álava	039	Moreda de Álava / Moreda Araba
16	País Vasco	01	Araba/Álava	041	Navaridas
16	País Vasco	01	Araba/Álava	042	Okondo
16	País Vasco	01	Araba/Álava	043	Oyón-Oion
16	País Vasco	01	Araba/Álava	044	Peñacerrada-Urizaharra
16	País Vasco	01	Araba/Álava	046	Erriberagoitia / Ribera Alta
16	País Vasco	01	Araba/Álava	047	Ribera Baja / Erribera Beitia
16	País Vasco	01	Araba/Álava	049	Añana
16	País Vasco	01	Araba/Álava	051	Agurain / Salvatierra
16	País Vasco	01	Araba/Álava	052	Samaniego
16	País Vasco	01	Araba/Álava	053	San Millán / Donemiliaga
16	País Vasco	01	Araba/Álava	054	Urkabustaiz
16	País Vasco	01	Araba/Álava	055	Valdegovía / Gaubea
16	País Vasco	01	Araba/Álava	056	Harana / Valle de Arana
16	País Vasco	01	Araba/Álava	057	Villabuena de Álava / Eskuernaga
16	País Vasco	01	Araba/Álava	058	Legutio
16	País Vasco	01	Araba/Álava	059	Vitoria-Gasteiz
16	País Vasco	01	Araba/Álava	060	Yécora / Iekora
16	País Vasco	01	5 Araba/Álava	061	Zalduondo
16	País Vasco	01	Araba/Álava	062	Zambrana
16	País Vasco	01	Araba/Álava	063	Zuia
16	País Vasco	01	Araba/Álava	901	Iruña Oka / Iruña de Oca

A tibble: 51 x 8

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

- Obtener municipios de la Comunidad de Madrid

[47]:
```
edata_02 <- data_02 |>
    tibble() |>
    filter(cpro == "28") |>
    select(cmun, name)
```

- Unir con municipios coincidentes

[48]:
```
edata_01 <- data_01 |>
  inner_join(edata_02,
             by = c("municipio_destino" = "name"))
```

[49]:
```
edata_01 |> glimpse()
```

```
Rows: 521,280
Columns: 7
$ total_nacional    <chr> "Total Nacional", "Total Nacional",
"Total Nacional"…
$ total_ccaa        <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA,
NA, NA, NA, NA, …
$ provincia         <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA,
NA, NA, NA, NA, …
$ municipio_destino <chr> "Acebeda, La", "Acebeda, La",
"Acebeda, La", "Acebed…
$ periodo           <chr> "2022M10", "2022M09", "2022M08",
"2022M07", "2022M06…
$ turistas          <chr> NA, NA, NA, "39", "125", "31", NA,
NA, NA, ".", NA, …
$ cmun              <chr> "001", "001", "001", "001", "001",
"001", "001", "00…
```

[50]:
```
edata_01 |> slice_head(n = 5)
```

	total_nacional <chr>	total_ccaa <chr>	provincia <chr>	municipio_destino <chr>	periodo <chr>	turistas <chr>	cmun <chr>
A spec_tbl_df: 5 x 7	Total Nacional	NA	NA	Acebeda, La	2022M10	NA	001
	Total Nacional	NA	NA	Acebeda, La	2022M09	NA	001
	Total Nacional	NA	NA	Acebeda, La	2022M08	NA	001
	Total Nacional	NA	NA	Acebeda, La	2022M07	39	001
	Total Nacional	NA	NA	Acebeda, La	2022M06	125	001

- Transformar periodo a mes para poder unir después, identificar NAs por secreto estadístico y convertir a numérica el número de turistas

[52]:
```
tdata <- edata_01 |>
    mutate(mes = str_replace(periodo, "M", "-")) |>
    select(-periodo) |>
    mutate(secreto = if_else(turistas == ".", 1, 0)) |>
    mutate(turistas = as.numeric(turistas))
```

```
Warning message:
"There was 1 warning in `mutate()`.
  In argument: `turistas = as.numeric(turistas)`.
Caused by warning:
! NAs introducidos por coerción"
```

[53]:
```
tdata |> glimpse()
```

```
Rows: 521,280
Columns: 8
$ total_nacional    <chr> "Total Nacional", "Total Nacional",
"Total Nacional"…
$ total_ccaa        <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA,
NA, NA, NA, NA, …
$ provincia         <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA,
NA, NA, NA, NA, …
$ municipio_destino <chr> "Acebeda, La", "Acebeda, La",
"Acebeda, La", "Acebed…
$ turistas          <dbl> NA, NA, NA, 39, 125, 31, NA, NA,
NA, NA, NA, NA, NA,…
$ cmun              <chr> "001", "001", "001", "001", "001",
"001", "001", "00…
$ mes               <chr> "2022-10", "2022-09", "2022-08",
"2022-07", "2022-06…
$ secreto           <dbl> NA, NA, NA, 0, 0, 0, NA, NA, NA, 1,
NA, NA, NA, NA, …
```

[54]:
```
tdata |> slice_head(n = 5)
```

	total_nacional <chr>	total_ccaa <chr>	provincia <chr>	municipio_destino <chr>	turistas <dbl>	cmun <chr>	mes <chr>	secre <dbl
A tibble: 5 x 8	Total Nacional	NA	NA	Acebeda, La	NA	001	2022-10	NA
	Total Nacional	NA	NA	Acebeda, La	NA	001	2022-09	NA
	Total Nacional	NA	NA	Acebeda, La	NA	001	2022-08	NA
	Total Nacional	NA	NA	Acebeda, La	39	001	2022-07	0
	Total Nacional	NA	NA	Acebeda, La	125	001	2022-06	0

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

[55]:
```
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

[56]:
```
caso <- "CU_45"
proceso <- '_05'
tarea <- "_04"
archivo <- ""
proper <- "_interno_prov"
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

[57]:
```
# 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/.

[58]:
```
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_04_interno_prov.csv'

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

[59]:
```
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 * stringr 1.5.0

0.8.3 Data structures

Objeto `data`

- Hay 521280 filas con información de las siguientes variables:
  - total_nacional
  - total_ccaa
  - provincia
  - municipio_destino
  - periodo
  - turistas
  - cmun

Observaciones generales sobre los datos

- Además de los valores de provincias, vienen datos totales por comunidad autónoma y total país
- En teoría serían la suma de las provincias que componen las regiones, pero podría no coincidir por el secreto estadístico (pocos datos en ununa provincia determinada, que sí se suman al total pero no se publican)
- Vienen dos tipos de valores perdidos: NA y un punto, que es valor no publicado
- Los NA es posible que sean cero, pero al no estar seguros se dejan como NA
- Los datos disponibles en el archivo utilizado son en este rango de meses

[60]:
```
data_to_save |> pull(mes) |> range()
```

1. '2019-07' 2. '2022-10'

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 provincia de origen
- Se ha creado la variable "secreto" para distinguir si es necesario los datos faltantes

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

[61]:
```
# incluir código
```