05. - Data Collection_CU_18_15_distritos_meteo_v_01

June 13, 2023

#

CU18_Infraestructuras_eventos

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 15. Agregar datos meteorológicos por distritos y año

- Estimar variables en distritos
- Agrupar promedios anuales de los distritos en las variables meteorológicas
- Eliminando variables que no se usarán en modelos

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 Packages to use

- {tcltk} para selección interactiva de archivos locales
- {sf} para trabajar con georeferenciación

- {gstat} para cálculos geoestadísticos
- {readr} para leer y escribir archivos csv
- {dplyr} para explorar datos
- {stringr} para manipulación de cadenas de caracteres
- {tidyr} para quitar perdidos

[1]:
```
library(readr)
library(dplyr)
library(sf)
library(gstat)
library(stringr)
library(tidyr)
```

```
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


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

0.1.2 Paths

[2]:
```
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 meteorológicos de estaciones

OPCION A: Seleccionar fichero en ventana para mayor comodidad

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

[3]:
```
# file_data_01 <- tcltk::tk_choose.files(multi = FALSE)
```

OPCION B: Especificar el nombre de archivo

[4]:
```
iFile_01 <- "CU_18_05_14_aemet.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 leer<U+00E1>n datos del archivo:  Data/Input/CU_18_05_14_aemet.csv
```

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

[5]:
```
data_01 <- read_csv(file_data_01)
```

```
Rows: 4547 Columns: 22
-- Column specification --------------------------------------------------------
Delimiter: ","
chr   (9): indicativo, nombre, provincia, horatmin, horatmax, dir,
horaracha…
dbl  (12): altitud, tmed, prec, tmin, tmax, velmedia, racha, sol,
presMax, p…
date  (1): fecha

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

Estructura de los datos:

[6]:
```
glimpse(data_01)
```

```
Rows: 4,547
Columns: 22
$ fecha       <date> 2019-01-01, 2019-01-02, 2019-01-03,
2019-01-04, 2019-01-0~
$ indicativo  <chr> "2462", "2462", "2462", "2462", "2462",
"2462", "2462", "2~
$ nombre      <chr> "PUERTO DE NAVACERRADA", "PUERTO DE
NAVACERRADA", "PUERTO ~
$ provincia   <chr> "MADRID", "MADRID", "MADRID", "MADRID",
"MADRID", "MADRID"~
$ altitud     <dbl> 1894, 1894, 1894, 1894, 1894, 1894, 1894,
1894, 1894, 1894~
$ tmed        <dbl> 6.4, 5.2, 7.6, 6.0, 7.2, 4.6, 4.1, 6.2,
```

```
-0.8, -4.1, -5.2, ~
$ prec        <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0~
$ tmin        <dbl> 1.6, -0.8, 2.7, 1.7, 1.8, 0.9, 1.2, 1.4,
-3.5, -7.5, -10.3~
$ horatmin    <chr> "23:40", "08:00", "23:59", "01:30",
"18:00", "07:10", "Var~
$ tmax        <dbl> 11.3, 11.3, 12.6, 10.2, 12.5, 8.2, 7.0,
10.9, 2.0, -0.7, -~
$ horatmax    <chr> "12:40", "12:40", "12:00", "12:10",
"12:50", "11:40", "13:~
$ dir         <chr> "36", "04", "15", "15", "03", "35", "36",
"02", "36", "36"~
$ velmedia    <dbl> 3.3, 2.8, 1.9, 1.4, 3.1, 3.3, 2.8, 2.8,
5.3, 3.1, 2.8, 1.9~
$ racha       <dbl> 8.9, 11.4, 8.9, 8.3, 8.3, 11.9, 7.2,
11.1, 12.8, 8.6, 8.9,~
$ horaracha   <chr> "23:59", "17:10", "04:00", "00:40",
"06:50", "08:10", "06:~
$ sol         <dbl> 7.9, 7.9, 7.3, 8.0, 8.2, 8.0, 8.1, 8.1,
3.6, 1.2, 8.2, 7.8~
$ presMax     <dbl> 823.1, 822.0, 819.5, 822.7, 824.3, 823.3,
821.8, 821.3, 81~
$ horaPresMax <chr> "10", "00", "10", "23", "10", "00", "11",
"Varias", "00", ~
$ presMin     <dbl> 821.7, 818.8, 818.0, 818.5, 822.4, 821.0,
820.0, 818.4, 81~
$ horaPresMin <chr> "05", "24", "05", "05", "Varias", "16",
"14", "24", "24", ~
$ X           <dbl> -4.010556, -4.010556, -4.010556,
-4.010556, -4.010556, -4.~
$ Y           <dbl> 40.79306, 40.79306, 40.79306, 40.79306,
40.79306, 40.79306~
```

Muestra de datos:

[7]:
```
slice_head(data_01, n = 5)
```

A spec_tbl_df: 5 x 22

fecha <date>	indicativo <chr>	nombre <chr>	provincia <chr>	altitud <dbl>	tmed <dbl>
2019-01-01	2462	PUERTO DE NAVACERRADA	MADRID	1894	6.4
2019-01-02	2462	PUERTO DE NAVACERRADA	MADRID	1894	5.2
2019-01-03	2462	PUERTO DE NAVACERRADA	MADRID	1894	7.6
2019-01-04	2462	PUERTO DE NAVACERRADA	MADRID	1894	6.0
2019-01-05	2462	PUERTO DE NAVACERRADA	MADRID	1894	7.2

2. Datos de contornos distritos

OPCION A: Seleccionar fichero en ventana para mayor comodidad

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

[8]:
```
# file_data_02 <- tcltk::tk_choose.files(multi = FALSE)
```

OPCION B: Especificar el nombre de archivo

[9]:
```
iFile_02 <- "CU_18_05_03_distritos_geo.json"
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 leer<U+00E1>n datos del archivo:  Data/Input/CU_18_05_03_distritos_geo.json
```

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

[10]:
```
data_02 <- st_read(file_data_02)
```

```
Reading layer `CU_18_05_03_distritos_geo' from data source
  `/Users/emilio.lcano/academico/gh_repos/__transferencia/citizenlab/CitizenLab-
Research-and-Development/casos_urjc/notebooks/II_data_processing/18_infraestruct
uras/Data/Input/CU_18_05_03_distritos_geo.json'
  using driver `GeoJSON'
Simple feature collection with 247 features and 2 fields
Geometry type: GEOMETRY
Dimension:     XY
Bounding box:  xmin: -4.579006 ymin: 39.8848 xmax: -3.052983 ymax: 41.16584
Geodetic CRS:  WGS 84
```

Estructura de los datos:

[11]:
```
glimpse(data_02)
```

```
Rows: 247
Columns: 3
$ CMUN     <chr> "001", "002", "003", "004", "005", "005",
"005", "005", "005"~
$ CDIS     <chr> "01", "01", "01", "01", "01", "02", "03",
"04", "05", "01", "~
$ geometry <POLYGON [arc_degree]> POLYGON ((-3.64502
41.12129…, POLYGON ((-3~
```

Muestra de datos:

[12]:
```
data_02 |> tibble() |> slice_head(n = 5)
```

	CMUN	CDIS	geometry
	<chr>	<chr>	<POLYGON [arc_degree]>
A tibble: 5 x 3	001	01	POLYGON ((-3.64502 41.12129...
	002	01	POLYGON ((-3.503032 40.526,...
	003	01	POLYGON ((-3.808664 40.8921...
	004	01	POLYGON ((-4.00197 40.25642...
	005	01	POLYGON ((-3.361691 40.4762...

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:

- Datos diarios meteorológicos por estación
- Contornos de distritos censales

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

- Corregir geometrías distritos y eliminar vacíos

[13]:
```
tdata_02 <- data_02 |> st_make_valid()
tdata_02 <- tdata_02 |> filter(!st_is_empty(tdata_02))
```

Data extract

- Seleccionar variables temperatura, precipitaciones y presión
- Agrupar por año

[14]:
```
tdata_01 <- data_01 |>
    select(nombre, tmed, prec, velmedia, presMax, X, Y) |>
    group_by(nombre, X, Y) |>
    summarise(across(everything(), ~mean(.x, na.rm = TRUE)))
```

```
`summarise()` has grouped output by 'nombre', 'X'. You can override
using the
`.groups` argument.
```

[16]:
```
glimpse(tdata_01)
```

```
Rows: 13
Columns: 7
Groups: nombre, X [13]
$ nombre   <chr> "ARANJUEZ", "BUITRAGO DEL LOZOYA", "COLMENAR
VIEJO", "GETAFE"~
$ X        <dbl> -3.546111, -3.613611, -3.765000, -3.722222,
-3.555556, -3.724~
```

```
$ Y        <dbl> 40.06722, 41.00694, 40.69611, 40.29944,
40.46667, 40.45167, 4~
$ tmed     <dbl> 15.584874, 12.800000, 14.219178, 16.398356,
15.673973, 14.484~
$ prec     <dbl> 0.7657224, 1.3893023, 1.1323288, 0.9484932,
0.9210959, 1.0198~
$ velmedia <dbl> 1.822830, 2.017447, 2.549589, 3.575890,
3.466301, 2.097973, 3~
$ presMax  <dbl> NaN, 905.8165, 905.7819, 948.3597, 952.3433,
NaN, 940.5129, 9~
```

[17]:
```
tdata_01
```

A grouped_df: 13 x 7

nombre <chr>	X <dbl>	Y <dbl>	tmed <dbl>	prec <dbl>
ARANJUEZ	-3.546111	40.06722	15.584874	0.7657224
BUITRAGO DEL LOZOYA	-3.613611	41.00694	12.800000	1.3893023
COLMENAR VIEJO	-3.765000	40.69611	14.219178	1.1323288
GETAFE	-3.722222	40.29944	16.398356	0.9484932
MADRID AEROPUERTO	-3.555556	40.46667	15.673973	0.9210959
MADRID, CIUDAD UNIVERSITARIA	-3.724167	40.45167	14.484516	1.0198347
MADRID, CUATRO VIENTOS	-3.786111	40.37556	16.144658	1.0112329
MADRID, RETIRO	-3.678056	40.41194	16.143288	1.0726027
PUERTO ALTO DEL LE<U+00D3>N	-4.141944	40.70639	9.677966	1.3355932
PUERTO DE NAVACERRADA	-4.010556	40.79306	8.427123	3.1600000
ROBLEDO DE CHAVELA	-4.250000	40.42778	14.799711	0.6211155
SOMOSIERRA	-3.580278	41.13556	10.631250	5.0754325
TORREJ<U+00D3>N DE ARDOZ	-3.443611	40.48861	15.345753	0.7750685

Data transform

- Convertir a objeto `sf` para poder hacer las estimaciones

[19]:
```
tdata_01_grouped_sf <- st_as_sf(tdata_01, coords = c("X", "Y"), crs = 4326)
```

Data transform

- Estimar valores medios de cada distrito con los datos puntuales
- Unir a la tabla

[20]:
```
tmed <- idw(tmed ~ 1,
            locations = tdata_01_grouped_sf |> drop_na(tmed),
            newdata = tdata_02)

prec <- idw(prec ~ 1,
            locations = tdata_01_grouped_sf |> drop_na(prec),
            newdata = tdata_02)

velmedia <- idw(velmedia ~ 1,
```

```
          locations = tdata_01_grouped_sf |> drop_na(velmedia),
          newdata = tdata_02)
presMax <- idw(presMax ~ 1,
          locations = tdata_01_grouped_sf |> drop_na(presMax),
          newdata = tdata_02)
```

```
[inverse distance weighted interpolation]
[inverse distance weighted interpolation]
[inverse distance weighted interpolation]
[inverse distance weighted interpolation]
```

[21]:
```
tdata_02_meteo <- tdata_02 |>
    mutate(tmed = tmed$var1.pred,
           prec = prec$var1.pred,
           velmedia = velmedia$var1.pred,
           presMax = presMax$var1.pred) |>
    st_drop_geometry()
```

[22]:
```
glimpse(tdata_02_meteo)
```

```
Rows: 246
Columns: 6
$ CMUN     <chr> "001", "002", "003", "004", "005", "005",
"005", "005", "005"~
$ CDIS     <chr> "01", "01", "01", "01", "01", "02", "03",
"04", "05", "01", "~
$ tmed     <dbl> 11.47600, 15.31806, 12.00935, 15.13839,
15.27439, 15.31156, 1~
$ prec     <dbl> 3.6992270, 0.8713863, 2.0387942, 1.0548851,
0.8788330, 0.8511~
$ velmedia <dbl> 2.090249, 3.352072, 2.986153, 2.919731,
3.331738, 3.372923, 3~
$ presMax  <dbl> 904.2719, 946.6977, 880.7764, 931.9199,
945.8412, 946.8914, 9~
```

[23]:
```
tdata_02_meteo |> slice_head(n = 5)
```

A data.frame: 5 x 6

CMUN <chr>	CDIS <chr>	tmed <dbl>	prec <dbl>	velmedia <dbl>	presMax <dbl>
001	01	11.47600	3.6992270	2.090249	904.2719
002	01	15.31806	0.8713863	3.352072	946.6977
003	01	12.00935	2.0387942	2.986153	880.7764
004	01	15.13839	1.0548851	2.919731	931.9199
005	01	15.27439	0.8788330	3.331738	945.8412

0.4 Synthetic Data Generation

No aplica

0.5 Fake Data Generation

No aplica

0.6 Open Data

Los datos originales fueron descargados de fuentes abiertas:

- Meteorológicas de AEMET
- Geográficas (distritos) del INE

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

[24]:
```
data_to_save <- tdata_02_meteo
```

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

[25]:
```
caso <- "CU_18"
proceso <- '_05'
tarea <- "_15"
archivo <- ""
proper <- "_distritos_meteo"
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/.

[26]:
```
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_18_05_15_distritos_meteo.csv'

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

[27]:
```
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 {sf}, {gstat}, {readr}, {dplyr}, {stringr} and {tidyr} packages must be installed.
- 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 * sf 1.0.9 * gstat 2.1.0 * readr 2.1.3 * dplyr 1.0.10 * stringr 1.5.0 * tidyr 1.3.0

0.8.3 Data structures

Objeto `data`

- Hay 246 filas con las variables:
  - CMUN

  - CDIS
  - tmed
  - prec
  - velmedia
  - presMax

[ ]:
```
colnames(tdata_02_meteo)
```

Observaciones generales sobre los datos

- Se han agregado los datos totales porque se tiene un año. Si se usan más años en los modelos, el agrupamiento debe ser de la granularidad de la que haya datos.

0.8.4 Consideraciones para despliegue en piloto

- Si llegan datos reales se deberían descargar los datos de las fechas que coincidan

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 interpolado los datos meteorológicos a los ditritos a través de los centroides
- Se han agrupado los datos

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

- Se deben unir a los datos agregados de distritos del resto de variables

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
```