05. - Data Collection_CU_18_17_infra_distancias_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 17. Cálculo distancias

- Calcular distancias de las infraestructuras a puntos relevantes más cercanos:
  - Aeropuerto
  - Gran Intercambiador
  - Estación de cercanías
  - Hospital
  - Campus universitario
  - Ayuntamiento, ministerio, …
  - Embajadas
  - Centros Comerciales

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
- {readr} para leer y escribir archivos csv
- {dplyr} para explorar datos

[1]:
```
library(sf)
library(readr)
library(dplyr)
```

```
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
```

0.1.2 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. Infraestructuras

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_04_infraestructuras.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_18_05_04_infraestructuras.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: 1633 Columns: 10
  Column specification

Delimiter: ","
chr (8): grupo, tipo, nombre, CODMUN, DIRECCION, info, CMUN, CDIS
dbl (2): X, Y

 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:

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

```
Rows: 1,633
Columns: 10
$ grupo     <chr> "Transporte", "Transporte", "Transporte",
"Transporte", "Tra…
$ tipo      <chr> "Intercambiadores", "Intercambiadores",
"Intercambiadores", …
$ nombre    <chr> "Grandes Intercambiadores Plaza Elíptica",
"Grandes Intercam…
$ CODMUN    <chr> "079", "079", "079", "079", "079", "079",
"079", "079", "079…
$ DIRECCION <chr> "Plaza Elíptica s/n", "Calle Princesa, 89",
"Estación Prínci…
$ X         <dbl> -3.716577, -3.719508, -3.719560, -3.689044,
-3.676731, -3.68…
$ Y         <dbl> 40.38540, 40.43474, 40.42080, 40.46719,
40.43797, 40.47198, …
$ info      <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
```

```
NA, NA, NA, NA, …
$ CMUN      <chr> "079", "079", "079", "079", "079", "079",
"079", "079", "079…
$ CDIS      <chr> "12", "09", "09", "05", "05", "05", "06",
"02", "01", "10", …
```

Muestra de datos:

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

A spec_tbl_df: 5 × 10

grupo <chr>	tipo <chr>	nombre <chr>	CODM <chr>
Transporte	Intercambiadores	Grandes Intercambiadores Plaza Elíptica	079
Transporte	Intercambiadores	Grandes Intercambiadores Moncloa	079
Transporte	Intercambiadores	Grandes Intercambiadores Príncipe Pío	079
Transporte	Intercambiadores	Grandes Intercambiadores Plaza de Castilla	079
Transporte	Intercambiadores	Grandes Intercambiadores Avenida de América	079

2. Administración y servicios

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_08_adm_distritos.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_18_05_08_adm_distritos.csv
```

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

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

```
Rows: 120280 Columns: 7
  Column specification

Delimiter: ","
chr (5): Grupo, Tipo, ETIQUETA, CMUN, CDIS
dbl (2): X, Y

  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:

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

```
Rows: 120,280
Columns: 7
$ Grupo    <chr> "Administración pública", "Administración
pública", "Administ…
$ Tipo     <chr> "Agencia Tributaria", "Agencia Tributaria",
"Agencia Tributar…
$ ETIQUETA <chr> "Administración de la Agencia Tributaria
Oficina Central", "A…
$ X        <dbl> -3.698678, -3.712513, -3.745598, -3.690993,
-3.672477, -3.710…
$ Y        <dbl> 40.45554, 40.44553, 40.37049, 40.41795,
40.45483, 40.48316, 4…
$ CMUN     <chr> "079", "079", "079", "079", "079", "079",
"079", "079", "079"…
$ CDIS     <chr> "06", "07", "11", "03", "05", "08", "04",
"14", "20", "12", "…
```

Muestra de datos:

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

A spec_tbl_df: 5 × 7

Grupo <chr>	Tipo <chr>	ETIQUETA <chr>
Administración pública	Agencia Tributaria	Administración de la Agencia Tributaria O
Administración pública	Agencia Tributaria	Administración de la Agencia Tributaria G
Administración pública	Agencia Tributaria	Administración de la Agencia Tributaria Su
Administración pública	Agencia Tributaria	Administración de la Agencia Tributaria M
Administración pública	Agencia Tributaria	Administración de la Agencia Tributaria C

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:

- Infraestructuras
- Localizaciones de administración y servicios

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

Data transformation

- Convertir coordenadas a objeto `sf`

[13]:
```
tdata_01 <- data_01 |>
    st_as_sf(coords = c("X", "Y")) |>
    st_set_crs(4326)
```

[14]:
```
tdata_02 <- data_02 |>
    st_as_sf(coords = c("X", "Y")) |>
    st_set_crs(4326)
```

- Calcular distancias

[15]:
```
tdata_01$dist_aeropuerto <-
    sapply(1:nrow(tdata_01),
       function(x){
         min(st_distance(tdata_01 |> slice(x),
                         tdata_01 |> filter(tipo == "Aeropuertos")))/1000
       })
```

[16]:
```
tdata_01$dist_intercambiador <-
    sapply(1:nrow(tdata_01),
       function(x){
         min(st_distance(tdata_01 |> slice(x),
                         tdata_01 |> filter(tipo == "Intercambiadores")))/1000
       })
```

[17]:
```
tdata_01$dist_cercanias <-
    sapply(1:nrow(tdata_01),
       function(x){
         min(st_distance(tdata_01 |> slice(x),
                         tdata_01 |> filter(tipo == "Estaciones de␣
↪Cercanías")))/1000
       })
```

[18]:
```
tdata_01$dist_hospital <-
    sapply(1:nrow(tdata_01),
       function(x){
         min(st_distance(tdata_01 |> slice(x),
                         tdata_01 |> filter(tipo == "Hospitales")))/1000
       })
```

[19]:
```
tdata_01$dist_universidad <-
    sapply(1:nrow(tdata_01),
       function(x){
         min(st_distance(tdata_02 |> slice(x),
                         tdata_02 |> filter(Tipo == "Campus universitarios")))/
↪1000
       })
```

[20]:
```
tdata_01$dist_gob <-
    sapply(1:nrow(tdata_01),
       function(x){
         min(st_distance(tdata_02 |> slice(x),
                         tdata_02 |> filter(Tipo == "Ayuntamientos,␣
↪Consejerías, Ministerios, etc.")))/1000
       })
```

[21]:
```
tdata_01$dist_embajada <-
    sapply(1:nrow(tdata_01),
       function(x){
         min(st_distance(tdata_02 |> slice(x),
                         tdata_02 |> filter(Tipo == "Embajadas y consulados")))/
↪1000
       })
```

[22]:
```
tdata_01$dist_ccomercial <-
    sapply(1:nrow(tdata_01),
       function(x){
         min(st_distance(tdata_02 |> slice(x),
                         tdata_02 |> filter(Tipo == "Centros comerciales")))/
↪1000
       })
```

- Pasar geometría a columnas numéricas y eliminar objeto sf

[23]:
```
tdata_01_out <- tdata_01 |>
    bind_cols(st_coordinates(tdata_01)) |>
    st_drop_geometry()
```

[24]:
```
tdata_01_out |> tibble() |> slice_head(n = 5)
```

A tibble: 5 × 18

grupo <chr>	tipo <chr>	nombre <chr>	CODMUN <chr>
Transporte	Intercambiadores	Grandes Intercambiadores Plaza Elíptica	079
Transporte	Intercambiadores	Grandes Intercambiadores Moncloa	079
Transporte	Intercambiadores	Grandes Intercambiadores Príncipe Pío	079
Transporte	Intercambiadores	Grandes Intercambiadores Plaza de Castilla	079
Transporte	Intercambiadores	Grandes Intercambiadores Avenida de América	079

[25]:
```
tdata_01_out |> glimpse()
```

```
Rows: 1,633
Columns: 18
$ grupo               <chr> "Transporte", "Transporte",
"Transporte", "Transpo…
$ tipo                <chr> "Intercambiadores",
"Intercambiadores", "Intercamb…
$ nombre              <chr> "Grandes Intercambiadores Plaza
```

```
Elíptica", "Grande…
$ CODMUN              <chr> "079", "079", "079", "079",
"079", "079", "079", "…
$ DIRECCION           <chr> "Plaza Elíptica s/n", "Calle
Princesa, 89", "Estac…
$ info                <chr> NA, NA, NA, NA, NA, NA, NA, NA,
NA, NA, NA, NA, NA…
$ CMUN                <chr> "079", "079", "079", "079",
"079", "079", "079", "…
$ CDIS                <chr> "12", "09", "09", "05", "05",
"05", "06", "02", "0…
$ dist_aeropuerto     <dbl> 5.2031430, 8.5507475, 7.3002463,
9.9339069, 9.1988…
$ dist_intercambiador <dbl> 0.00000000, 0.00000000,
0.00000000, 0.00000000, 0.…
$ dist_cercanias      <dbl> 1.648973148, 1.468638183,
0.088338692, 0.736877403…
$ dist_hospital       <dbl> 1.8154156, 0.4221115, 1.3855767,
1.0733484, 0.3488…
$ dist_universidad    <dbl> 1.29960755, 0.30460781,
0.92076145, 0.08835165, 0.…
$ dist_gob            <dbl> 0.61772353, 0.62638474,
1.17005116, 0.14904962, 1.…
$ dist_embajada       <dbl> 3.921324e-01, 1.457397e+00,
5.624560e+00, 8.316773…
$ dist_ccomercial     <dbl> 0.73115706, 1.61056736,
0.90608573, 0.77583440, 0.…
$ X                   <dbl> -3.716577, -3.719508, -3.719560,
-3.689044, -3.676…
$ Y                   <dbl> 40.38540, 40.43474, 40.42080,
40.46719, 40.43797, …
```

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

0.4 Synthetic Data Generation

No aplica

0.5 Fake Data Generation

No aplica

0.6 Open Data

Los datos originales fueron obtenidos de fuentes abiertas (CM, OSM)

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

[26]:
```
data_to_save <- tdata_01_out
```

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

[27]:
```
caso <- "CU_18"
proceso <- '_05'
tarea <- "_17"
archivo <- ""
proper <- "_infraestructuras"
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

[28]:
```
# 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/.

[29]:
```
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_17_infraestructuras.csv'

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

[32]:
```
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:

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

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
- readr 2.1.3
- dplyr 1.0.10

0.8.3 Data structures

Objeto `tdata_01_out`

- En los datos de origen tenemos coordenadas de un buen número de infraestructuras y otros puntos de interés
- Hay 1633 filas con las variables:
  - grupo
  - tipo
  - nombre
  - CODMUN
  - DIRECCION
  - info
  - CMUN
  - CDIS

  - dist_aeropuerto
  - dist_intercambiador
  - dist_cercanias
  - dist_hospital
  - dist_universidad
  - dist_gob
  - dist_embajada
  - dist_ccomercial
  - X
  - Y

Observaciones generales sobre los datos

- Las distancias se han calculado en kilómetros, en distancia ortodrómica *(great circle distance)*
- Los datos de metro están por bocas de metro

0.8.4 Consideraciones para despliegue en piloto

- Los datos se importarán directamente

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 calculado distancias entre cada infraestructura y puntos relevantes
- Se han convertido coordenadas numéricas a objetos espaciales y viceversa para poder calcular las distancias ortodrómicas

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

- Se deben agrupar los datos de boca de metro por estaciones
- Se deben unir los datos de infraestructuras a los de eventos diarios para aplicar 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

[31]:
```
# incluir código
```