05. - Data Collection_CU_18_20_infra_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 20. Interpolar variables clima a infraestructuras

- A partir de los datos de las estaciones de medición AEMET, se interpolan por kriging los valores estimados en cada infraestructura

Table of Contents

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.

[ ]:
```
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
- {sf} para trabajar con georeferenciación
- {readr} para leer y escribir archivos csv
- {dplyr} para explorar datos
- {tidyr} para organización de datos
- {gstat} para operaciones geoestadísticas

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

p <- c("tcltk", "sf", "readr", "dplyr", "tidyr", "gstat")
```

```
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
```

0.1.3 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 estaciones AEMET

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)
```

	fecha <date>	indicativo <chr>	nombre <chr>	provincia <chr>	altitud <dbl>	tmed <dbl>
A spec_tbl_df: 5 x 22	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 infraestructuras (con identificador)

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_19_01_infraestructuras.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 leer<U+00E1>n datos del archivo:
Data/Input/CU_18_05_19_01_infraestructuras.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: 1138 Columns: 19
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (8): grupo, tipo, nombre, CODMUN, DIRECCION, info, CMUN, CDIS
dbl (11): id_inf, dist_aeropuerto, dist_intercambiador,
dist_cercanias, dist…

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:

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

```
Rows: 1,138
Columns: 19
$ id_inf              <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
11, 12, 13, 14, 15,~
$ grupo               <chr> "Transporte", "Transporte",
"Transporte", "Transpo~
$ tipo                <chr> "Intercambiadores",
"Intercambiadores", "Intercamb~
$ nombre              <chr> "Grandes Intercambiadores Plaza
El<U+00ED>ptica", "Grande~
$ CODMUN              <chr> "079", "079", "079", "079",
"079", "079", "079", "~
$ DIRECCION           <chr> "Plaza El<U+00ED>ptica 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> 0.0000000, 0.0000000, 0.0000000,
0.0000000, 0.0000~
$ 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, ~
```

Muestra de datos:

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

	id_inf <dbl>	grupo <chr>	tipo <chr>	nombre <chr>
A spec_tbl_df: 5 x 19	1	Transporte	Intercambiadores	Grandes Intercambiadores Plaza El<U+00ED>
	2	Transporte	Intercambiadores	Grandes Intercambiadores Moncloa
	3	Transporte	Intercambiadores	Grandes Intercambiadores Pr<U+00ED>ncipe
	4	Transporte	Intercambiadores	Grandes Intercambiadores Plaza de Castilla
	5	Transporte	Intercambiadores	Grandes Intercambiadores Avenida de Am<U+

3. Datos del diario de infraestructuras

OPCION A: Seleccionar fichero en ventana para mayor comodidad

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

[ ]:
```
# file_data_03 <- tcltk::tk_choose.files(multi = FALSE)
```

OPCION B: Especificar el nombre de archivo

[23]:
```
iFile_03 <- "CU_18_05_19_02_diario_infra.csv"
file_data_03 <- paste0(iPath, iFile_03)

if(file.exists(file_data_03)){
    cat("Se leerán datos del archivo: ", file_data_03)
} else{
    warning("Cuidado: el archivo no existe.")
}
```

```
Se leer<U+00E1>n datos del archivo:  Data/Input/CU_18_05_19_02_diario_infra.csv
```

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

[24]:
```
data_03 <- read_csv(file_data_03)
```

```
Rows: 415370 Columns: 6
-- Column specification --------------------------------------------------------
Delimiter: ","
dbl  (5): id_inf, capacidad, demanda, evento_infra, evento_zona
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:

[25]:
```
glimpse(data_03)
```

```
Rows: 415,370
Columns: 6
$ id_inf       <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
13, 14, 15, 16, 17~
$ fecha        <date> 2019-01-01, 2019-01-01, 2019-01-01,
2019-01-01, 2019-01-~
$ capacidad    <dbl> 993, 996, 1036, 1020, 992, 1026, 1007,
976, 1037, 972, 94~
$ demanda      <dbl> 883, 888, 922, 1134, 1103, 1139, 897,
1086, 1150, 861, 83~
$ evento_infra <dbl> 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0,
0, 0, 1, 0, 1, 1, ~
$ evento_zona  <dbl> 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0,
1, 1, 1, 1, 0, 1, ~
```

Muestra de datos:

[26]:
```
slice_head(data_03, n = 5)
```

A spec_tbl_df: 5 x 6

id_inf <dbl>	fecha <date>	capacidad <dbl>	demanda <dbl>	evento_infra <dbl>	evento_zona <dbl>
1	2019-01-01	993	883	1	1
2	2019-01-01	996	888	0	0
3	2019-01-01	1036	922	0	0
4	2019-01-01	1020	1134	1	0
5	2019-01-01	992	1103	1	1

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 de infraestructuras tratados con anterioridad
- Datos meteorológicos

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

- Convertir tablas de datos en objetos espaciales para poder interpolar

[13]:
```
tdata_01 <- data_01 |>
  st_as_sf(coords = c("X", "Y"), crs = 4326)
tdata_02 <- data_02 |>
  st_as_sf(coords = c("X", "Y"), crs = 4326)
```

- Interpolar temperatura media, precipitación, velocidad media del viento y presión máxima en las coordenadas de cada infraestructura.

NOTA: esta operación puede tardar varios minutos.

[21]:
```
meteovars <- c("tmed", "prec", "velmedia", "presMax")
fechas <- tdata_01 |> pull(fecha) |> unique()
resl <- list()

for (i in seq_along(fechas)){
  f <- fechas[i]
#   cat("Calculando", as.character(f), "\n")
  meteod <- tdata_01 |>
    filter(fecha == f) |>
    select(all_of(meteovars))

  res <- tdata_02 |> select(id_inf) |> st_drop_geometry()
  for (j in seq_along(meteovars)){
    thisvar <- meteovars[j]
    meteodc <- meteod |>
      drop_na(all_of(thisvar))

    if(all(meteodc |> pull(thisvar) == 0)){
      res <- res |>
        mutate(!!thisvar := 0)
    } else{

      fo <- as.formula(paste0(thisvar, " ~ 1"))
      v0 <- variogram(fo, meteodc)
      v.m <- suppressWarnings(fit.variogram(v0,
        vgm(c("Exp", "Mat", "Sph", "Log", "Wav", "Bes", "Lin", "Leg"))))
      invisible(capture.output(suppressWarnings(krg <- krige(fo,
                   locations = meteodc,
                   newdata = tdata_02,
                   model = v.m))))
      res <- res |>
        mutate(!!thisvar := krg$var1.pred)
    }
  }
  res <- res |>
    mutate(fecha = f)
  resl[[i]] <- res
}

data <- bind_rows(resl)
```

```
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
```

```
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
[1] "a possible solution MIGHT be to scale semivariances and/or distances"
```

[28]:
```
glimpse(data)
```

```
Rows: 415,370
Columns: 6
$ id_inf   <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,
14, 15, 16, 17, 18~
$ tmed     <dbl> 6.953211, 6.196420, 6.483569, 5.875797,
6.212680, 5.878542, 6~
$ prec     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0~
$ velmedia <dbl> 0.6433886, 0.3417523, 0.4132169, 0.1820178,
0.2118110, 0.1742~
$ presMax  <dbl> 952.9357, 950.2191, 950.8051, 951.6768,
953.5118, 952.1681, 9~
$ fecha    <date> 2019-01-01, 2019-01-01, 2019-01-01,
2019-01-01, 2019-01-01, ~
```

- Unir datos meteorológicos al diario de infraestructuras

[29]:
```
tdata <- data_03 |>
    inner_join(data, by = c("id_inf", "fecha"))
```

[30]:
```
glimpse(tdata)
```

```
Rows: 415,370
Columns: 10
$ id_inf       <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
13, 14, 15, 16, 17~
```

```
$ fecha        <date> 2019-01-01, 2019-01-01, 2019-01-01,
2019-01-01, 2019-01-~
$ capacidad    <dbl> 993, 996, 1036, 1020, 992, 1026, 1007,
976, 1037, 972, 94~
$ demanda      <dbl> 883, 888, 922, 1134, 1103, 1139, 897,
1086, 1150, 861, 83~
$ evento_infra <dbl> 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0,
0, 0, 1, 0, 1, 1, ~
$ evento_zona  <dbl> 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0,
1, 1, 1, 1, 0, 1, ~
$ tmed         <dbl> 6.953211, 6.196420, 6.483569, 5.875797,
6.212680, 5.87854~
$ prec         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, ~
$ velmedia     <dbl> 0.6433886, 0.3417523, 0.4132169,
0.1820178, 0.2118110, 0.~
$ presMax      <dbl> 952.9357, 950.2191, 950.8051, 951.6768,
953.5118, 952.168~
```

[31]:
```
tdata |> slice_head(n = 5)
```

A spec_tbl_df: 5 x 10

id_inf <dbl>	fecha <date>	capacidad <dbl>	demanda <dbl>	evento_infra <dbl>	evento_zona <dbl>	tmed <dbl>	pr <
1	2019-01-01	993	883	1	1	6.953211	0
2	2019-01-01	996	888	0	0	6.196420	0
3	2019-01-01	1036	922	0	0	6.483569	0
4	2019-01-01	1020	1134	1	0	5.875797	0
5	2019-01-01	992	1103	1	1	6.212680	0

0.4 Synthetic Data Generation

No aplica

0.5 Fake Data Generation

No aplica

0.6 Open Data

Los datos originales procedían de fuentes abiertas

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

[32]:
```
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

[33]:
```
caso <- "CU_18"
proceso <- '_05'
tarea <- "_20"
archivo <- ""
proper <- "_diario_infra"
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/.

[34]:
```
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_20_diario_infra.csv'

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

[35]:
```
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 tcltk, sf, readr, dplyr, tidyr, gstat deben estar instalados.

[36]:
```
cat("Los paquetes", paste(p, collapse = ", "), "deben estar instalados.")
```

```
Los paquetes tcltk, sf, readr, dplyr, tidyr, gstat 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
- sf 1.0.9
- readr 2.1.3
- dplyr 1.0.10
- tidyr 1.3.0
- gstat 2.1.0

0.8.3 Data structures

Objeto `tdata`

- Los datos de origen son las ubicaciones de infraestructuras y los datos de las estaciones meteorológicas
- Hay 415370 filas con las variables:
  - id_inf
  - fecha
  - capacidad
  - demanda
  - evento_infra

  - evento_zona
  - tmed
  - prec
  - velmedia
  - presMax

Observaciones generales sobre los datos

- Las estimaciones por interpolación pueden resultar en valores no válidos

0.8.4 Consideraciones para despliegue en piloto

- Los datos de capacidad y demanda son simulados mientras no se tengan datos completos en las tareas anteriores

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 estimado las variables meteorológicas en las coordenadas de cada infraestructura para cada día de 2019
- Se han unido las estimaciones al diario de infraestructuras con datos simulados

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

- Se debe comprobar y en su caso corregir la inconsistencia en las estimaciones (por ejemplo, precipitación o velocidad menor de cero)

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 código que desplegar en el piloto, ya que estos datos son estáticos para el piloto (2019).
- El fichero de salida se debe incluir en la ruta de datos de la implementación ya que será utilizado en los modelos

CODE

```
[ ]: # incluir código
```