05. - Data Collection_CU_45_02_valoracion_sim_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 02. Simular datos de puntuaciones de establecimiento

- Para cada establecimiento de los obtenidos en pois, simular su puntuación mientras se obtienen los datos reales de puntuaciones.

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.

[5]:
```
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

ELIMINAR O AÑADIR LO QUE TOQUE. COPIAR VERSIONES AL FINAL Y QUITAR CÓDIGO DE VERSIONES

- {tcltk} para selección interactiva de archivos locales
- {readr} para leer y escribir archivos csv
- {dplyr} para explorar datos
- {tidyr} para organización de datos

[6]:
```
library(readr)
library(dplyr)
library(tidyr)

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

0.1.3 Paths

[7]:
```
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 de establecimientos turísticos

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_01 <- tcltk::tk_choose.files(multi = FALSE)
```

OPCION B: Especificar el nombre de archivo

[9]:
```
iFile_01 <- "CU_18_05_12_pois_distrito.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_12_pois_distrito.csv
```

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

[10]:
```
data_01 <- read_csv(file_data_01)
```

```
Rows: 24780 Columns: 7
  Column specification

Delimiter: ","
chr (5): grupo, tipo, nombre, 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]:
```
data_01 |> glimpse()
```

```
Rows: 24,780
Columns: 7
$ grupo  <chr> "turismo", "hosteleria", "hosteleria",
"hosteleria", "comercio"…
$ tipo   <chr> "hotel", "restaurant", "pub", "pub",
"supermarket", "fast_food"…
$ nombre <chr> "NH Ciudad de la Imagen", "Café Comercial",
"Sidrería la Camoch…
$ X      <dbl> -3.788176, -3.702002, -3.701686, -3.696329,
-3.706888, -3.60722…
$ Y      <dbl> 40.39844, 40.42873, 40.42703, 40.42760,
40.48035, 40.43337, 40.…
$ CMUN   <chr> "115", "079", "079", "079", "079", "079",
"079", "079", "079", …
$ CDIS   <chr> "01", "01", "01", "01", "08", "20", "01",
"01", "01", "01", "01…
```

Muestra de datos:

[12]:
```
data_01 |> slice_head(n = 5)
```

	grupo <chr>	tipo <chr>	nombre <chr>	X <dbl>	Y <dbl>	CMUN <chr>	CD <ch
A spec_tbl_df: 5 x 7	turismo	hotel	NH Ciudad de la Imagen	-3.788176	40.39844	115	01
	hosteleria	restaurant	Café Comercial	-3.702002	40.42873	079	01
	hosteleria	pub	Sidrería la Camocha	-3.701686	40.42703	079	01
	hosteleria	pub	Gran Cafe Santander	-3.696329	40.42760	079	01
	comercio	supermarket	Alcampo	-3.706888	40.48035	079	08

2. Datos de indicadores INE

OPCION A: Seleccionar fichero en ventana para mayor comodidad

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

[13]:
```
# file_data_02 <- tcltk::tk_choose.files(multi = FALSE)
```

OPCION B: Especificar el nombre de archivo

[14]:
```
iFile_02 <- "CU_18_05_10_indicadores_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_10_indicadores_distritos.csv
```

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

[15]:
```
data_02 <- read_csv(file_data_02)
```

```
Rows: 247 Columns: 22
  Column specification

Delimiter: ","
chr  (2): CMUN, dist
dbl (20): nsec, t3_1, t1_1, t2_1, t2_2, t4_1, t4_2, t4_3, t5_1, t6_1,
t7_1, …

  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:

[16]:
```
data_02 |> glimpse()
```

```
Rows: 247
Columns: 22
$ CMUN            <chr> "001", "002", "003", "004", "005",
"005", "005", "005"…
$ dist            <chr> "01", "01", "01", "01", "01", "02",
"03", "04", "05", …
$ nsec            <dbl> 1, 2, 1, 4, 23, 36, 16, 18, 32, 67,
19, 37, 29, 35, 1,…
$ t3_1            <dbl> NA, 39.40747, 47.45970, 41.46662,
45.99877, 42.75131, …
$ t1_1            <dbl> 55, 4793, 248, 9946, 31006, 54049,
28117, 38778, 43620…
$ t2_1            <dbl> NA, 0.4777851, 0.3952000, 0.5015109,
0.5298303, 0.5115…
$ t2_2            <dbl> NA, 0.5222149, 0.6048000, 0.4984891,
0.4701697, 0.4884…
$ t4_1            <dbl> NA, 0.1696289, 0.1371000, 0.1810684,
0.1141978, 0.1468…
$ t4_2            <dbl> NA, 0.7218958, 0.6573000, 0.6420633,
0.6427127, 0.6644…
$ t4_3            <dbl> NA, 0.10847530, 0.20560000,
0.17684664, 0.24307467, 0.…
$ t5_1            <dbl> NA, 0.15562992, 0.11290000,
0.15474242, 0.19647849, 0.…
$ t6_1            <dbl> NA, 0.1959363, 0.1411000, 0.1912543,
0.2400069, 0.2147…
$ t7_1            <dbl> NA, 0.41318314, 0.15420000,
0.15732451, 0.07438075, 0.…
$ t8_1            <dbl> NA, 0.40995322, 0.14490000,
0.15046840, 0.06545227, 0.…
$ t9_1            <dbl> NA, 0.4407990, 0.5280000, 0.2942034,
0.3850913, 0.2235…
$ t10_1           <dbl> NA, 0.10715222, 0.09320000,
0.16675872, 0.14129129, 0.…
$ t11_1           <dbl> NA, 0.6008132, 0.5000000, 0.4777910,
0.4565679, 0.4588…
$ t12_1           <dbl> NA, 0.6730932, 0.5514000, 0.5729139,
0.5316057, 0.5641…
$ X               <dbl> -3.635710, -3.480171, -3.849961,
-3.989259, -3.364638,…
$ Y               <dbl> 41.09315, 40.52396, 40.92033,
40.23240, 40.48268, 40.4…
$ densidad_hab_km <dbl> 2.514002, 242.602224, 9.647117,
451.618447, 17549.3832…
$ area_km2        <dbl> 21.877471, 19.756620, 25.707163,
22.023015, 1.766786, …
```

Muestra de datos:

[17]:
```
data_02 |> slice_head(n = 5)
```

A spec_tbl_df: 5 x 22

CMUN <chr>	dist <chr>	nsec <dbl>	t3_1 <dbl>	t1_1 <dbl>	t2_1 <dbl>	t2_2 <dbl>	t4_1 <dbl>	t4_2 <db
001	01	1	NA	55	NA	NA	NA	NA
002	01	2	39.40747	4793	0.4777851	0.5222149	0.1696289	0.72
003	01	1	47.45970	248	0.3952000	0.6048000	0.1371000	0.65
004	01	4	41.46662	9946	0.5015109	0.4984891	0.1810684	0.64
005	01	23	45.99877	31006	0.5298303	0.4701697	0.1141978	0.64

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 POIs de hostelería, comercio y turismo

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

- Calcular indicadores del INE por municipios

[18]:
```
tdata_02 <- data_02 |>
  mutate(across(c(t2_1:t2_2, t4_1:t12_1), ~ .x*t1_1)) |>
  group_by(CMUN) |>
  summarise(t3_1 = weighted.mean(x = t3_1, w = t1_1, na.rm = TRUE),
            t1_1 = sum(t1_1),
            across(c(t2_1:t2_2, t4_1:t12_1), ~sum(.x, na.rm = TRUE)/sum(t1_1,␣
 ↪na.rm = TRUE)),
            .groups = "keep")
```

[19]:
```
tdata_02 |> glimpse()
```

```
Rows: 179
Columns: 16
Groups: CMUN [179]
$ CMUN  <chr> "001", "002", "003", "004", "005", "006",
"007", "008", "009", "…
$ t3_1  <dbl> NaN, 39.40747, 47.45970, 41.46662, 42.62488,
40.89541, 43.98716,…
$ t1_1  <dbl> 55, 4793, 248, 9946, 195570, 116895, 171575,
3078, 20704, 15178,…
$ t2_1  <dbl> 0.0000000, 0.4777851, 0.3952000, 0.5015109,
0.5145997, 0.5220080…
$ t2_2  <dbl> 0.0000000, 0.5222149, 0.6048000, 0.4984891,
0.4854003, 0.4779920…
```

```
$ t4_1  <dbl> 0.0000000, 0.1696289, 0.1371000, 0.1810684,
0.1522553, 0.1733979…
$ t4_2  <dbl> 0.0000000, 0.7218958, 0.6573000, 0.6420633,
0.6621785, 0.6579805…
$ t4_3  <dbl> 0.00000000, 0.10847530, 0.20560000, 0.17684664,
0.18556448, 0.16…
$ t5_1  <dbl> 0.00000000, 0.15562992, 0.11290000, 0.15474242,
0.15881821, 0.14…
$ t6_1  <dbl> 0.0000000, 0.1959363, 0.1411000, 0.1912543,
0.1905063, 0.2271027…
$ t7_1  <dbl> 0.00000000, 0.41318314, 0.15420000, 0.15732451,
0.06873490, 0.06…
$ t8_1  <dbl> 0.00000000, 0.40995322, 0.14490000, 0.15046840,
0.05804835, 0.05…
$ t9_1  <dbl> 0.0000000, 0.4407990, 0.5280000, 0.2942034,
0.3385710, 0.4468360…
$ t10_1 <dbl> 0.00000000, 0.10715222, 0.09320000, 0.16675872,
0.15005627, 0.10…
$ t11_1 <dbl> 0.0000000, 0.6008132, 0.5000000, 0.4777910,
0.4955043, 0.5243721…
$ t12_1 <dbl> 0.0000000, 0.6730932, 0.5514000, 0.5729139,
0.5799657, 0.5837953…
```

[20]:
```
tdata_02 |> slice_head(n = 5)
```

A grouped_df: 179 x 16

CMUN	t3_1	t1_1	t2_1	t2_2	t4_1	t4_2	t4_3
<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
001	NaN	55	0.0000000	0.0000000	0.0000000	0.0000000	0.000000
002	39.40747	4793	0.4777851	0.5222149	0.1696289	0.7218958	0.108475
003	47.45970	248	0.3952000	0.6048000	0.1371000	0.6573000	0.205600
004	41.46662	9946	0.5015109	0.4984891	0.1810684	0.6420633	0.176846
005	42.62488	195570	0.5145997	0.4854003	0.1522553	0.6621785	0.185564
006	40.89541	116895	0.5220080	0.4779920	0.1733979	0.6579805	0.168614
007	43.98716	171575	0.5197152	0.4802848	0.1534518	0.6221151	0.224435
008	42.32070	3078	0.4899000	0.5101000	0.1780000	0.6313000	0.190700
009	39.85821	20704	0.5094248	0.4905752	0.1771307	0.6895597	0.133305
010	40.57497	15178	0.5095979	0.4904021	0.1794548	0.6811853	0.139355
011	45.52490	663	0.4992000	0.5008000	0.1357000	0.6561000	0.208100
012	39.29520	1365	0.4842000	0.5158000	0.1978000	0.6696000	0.132600
013	41.10474	60156	0.5124696	0.4875304	0.1778163	0.6541860	0.168001
014	38.91674	56315	0.5001971	0.4998029	0.1917218	0.6824303	0.125844
015	33.82565	33757	0.5017270	0.4982730	0.2666313	0.6700695	0.063268
016	47.65140	109	0.4404000	0.5596000	0.0917000	0.7064000	0.201800
017	41.85640	1818	0.4934000	0.5066000	0.1529000	0.6947000	0.152400
018	41.98571	6096	0.5014386	0.4985614	0.1651801	0.6741785	0.160591
019	42.17040	1749	0.4837000	0.5163000	0.1618000	0.6650000	0.173200
020	49.36620	213	0.4225000	0.5775000	0.1127000	0.6056000	0.281700
021	45.07410	810	0.4877000	0.5123000	0.1704000	0.5951000	0.234600
022	37.44676	59000	0.5114951	0.4885049	0.2280971	0.6606521	0.111236
023	39.36300	8006	0.4969797	0.5030203	0.2083559	0.6696437	0.122057
024	44.58720	218	0.3945000	0.6055000	0.0917000	0.7202000	0.188100
025	48.10590	529	0.4178000	0.5822000	0.0888000	0.6144000	0.296800
026	39.35909	10699	0.5074452	0.4925548	0.1829349	0.6909112	0.126185
027	42.50080	1943	0.5229000	0.4771000	0.1714000	0.6366000	0.192000
028	42.46490	2706	0.4948000	0.5052000	0.1693000	0.6763000	0.154500
029	42.15580	841	0.4839000	0.5161000	0.1653000	0.6825000	0.152200
030	42.66860	2776	0.5112000	0.4888000	0.1520000	0.6726000	0.175400
...	...	...	...	...	...	...	...
157	42.54150	1023	0.4976000	0.5024000	0.1623000	0.6500000	0.187700
158	43.83660	1022	0.4824000	0.5176000	0.1624000	0.6331000	0.204500
159	44.92680	792	0.4836000	0.5164000	0.1616000	0.6174000	0.221000
160	40.63154	13254	0.4943692	0.5056308	0.1799224	0.6784545	0.141598
161	37.54293	77931	0.5078077	0.4921923	0.2101806	0.6897520	0.100077
162	37.51653	4306	0.4854155	0.5145845	0.2199235	0.6667516	0.113324
163	41.91280	619	0.4830000	0.5170000	0.1535000	0.6947000	0.151900
164	41.55260	4705	0.5001097	0.4998903	0.1700330	0.6675695	0.162397
165	40.16640	3065	0.4861000	0.5139000	0.1762000	0.6799000	0.143900
166	43.30490	515	0.4660000	0.5340000	0.1301000	0.6990000	0.170900
167	38.13674	12542	0.4979184	0.5020816	0.1831998	0.7227704	0.094012
168	41.30110	2086	0.4779000	0.5221000	0.1663000	0.6702000	0.163500
169	41.06150	2392	0.4812000	0.5188000	0.1660000	0.7007000	0.133400
170	42.30861	3438	0.4892354	0.5107646	0.1576937	0.6570668	0.185281
171	41.33609	6775	0.4900346	0.5099654	0.1734372	0.6546015	0.171937
172	36.80356	15046	0.4947329	0.5052671	0.2179169	0.6832198	0.098833
173	44.78390	768	0.4818000	0.5182000	0.1328000	0.6549000	0.212200
174	43.23110	2648	0.4943000	0.5057000	0.1605000	0.6420000	0.197500
175	38.82900	1515	0.5083000	0.4917000	0.2158000	0.6627000	0.121500
176	37.91979	22453	0.5120099	0.4879901	0.1941522	0.6916640	0.114192

- Añadir indicadores a la tabla de POIs

[21]:
```
tdata_01 <- data_01 |>
  left_join(tdata_02 |> select(CMUN, t3_1), by = "CMUN") |>
  replace_na(list(t3_1 = 100))
```

[22]:
```
tdata_01 |> glimpse()
```

```
Rows: 24,780
Columns: 8
$ grupo  <chr> "turismo", "hosteleria", "hosteleria",
"hosteleria", "comercio"…
$ tipo   <chr> "hotel", "restaurant", "pub", "pub",
"supermarket", "fast_food"…
$ nombre <chr> "NH Ciudad de la Imagen", "Café Comercial",
"Sidrería la Camoch…
$ X      <dbl> -3.788176, -3.702002, -3.701686, -3.696329,
-3.706888, -3.60722…
$ Y      <dbl> 40.39844, 40.42873, 40.42703, 40.42760,
40.48035, 40.43337, 40.…
$ CMUN   <chr> "115", "079", "079", "079", "079", "079",
"079", "079", "079", …
$ CDIS   <chr> "01", "01", "01", "01", "08", "20", "01",
"01", "01", "01", "01…
$ t3_1   <dbl> 41.46522, 43.76242, 43.76242, 43.76242,
43.76242, 43.76242, 43.…
```

[23]:
```
tdata_01 |> slice_head(n = 5)
```

A spec_tbl_df: 5 x 8

grupo <chr>	tipo <chr>	nombre <chr>	X <dbl>	Y <dbl>	CMUN <chr>	CD <ch
turismo	hotel	NH Ciudad de la Imagen	-3.788176	40.39844	115	01
hosteleria	restaurant	Café Comercial	-3.702002	40.42873	079	01
hosteleria	pub	Sidrería la Camocha	-3.701686	40.42703	079	01
hosteleria	pub	Gran Cafe Santander	-3.696329	40.42760	079	01
comercio	supermarket	Alcampo	-3.706888	40.48035	079	08

0.4 Synthetic Data Generation

No aplica

0.5 Fake Data Generation

Ante la falta de datos, se simulan. Se podría intentar obtenerlos por webscrapping de webs de turismo como tripadvisor

La simulación se basa en la población del INE

Se asume puntuación media del alojamiento

[24]:
```
set.seed(1)
tdata_01$puntos <- sapply(seq_along(tdata_01$nombre),
       function(x){
         qq <- quantile(tdata_01$t3_1, probs = 0.5, na.rm = TRUE)
         if(tdata_01$t3_1[x] < qq){
           p <- c(0.1, 0.3, 0.3, 0.15, 0.15)
         } else{
           p <- c(0.1, 0.15, 0.15, 0.3, 0.3)
         }
         sample(1:5, 1, prob = p)
       })
```

[25]:
```
tdata_01 |> glimpse()
```

```
Rows: 24,780
Columns: 9
$ grupo  <chr> "turismo", "hosteleria", "hosteleria",
"hosteleria", "comercio"…
$ tipo   <chr> "hotel", "restaurant", "pub", "pub",
"supermarket", "fast_food"…
$ nombre <chr> "NH Ciudad de la Imagen", "Café Comercial",
"Sidrería la Camoch…
$ X      <dbl> -3.788176, -3.702002, -3.701686, -3.696329,
-3.706888, -3.60722…
$ Y      <dbl> 40.39844, 40.42873, 40.42703, 40.42760,
40.48035, 40.43337, 40.…
$ CMUN   <chr> "115", "079", "079", "079", "079", "079",
"079", "079", "079", …
$ CDIS   <chr> "01", "01", "01", "01", "08", "20", "01",
"01", "01", "01", "01…
$ t3_1   <dbl> 41.46522, 43.76242, 43.76242, 43.76242,
43.76242, 43.76242, 43.…
$ puntos <int> 2, 4, 4, 1, 5, 2, 1, 3, 3, 5, 5, 5, 3, 4, 2,
4, 3, 1, 4, 2, 1, …
```

[26]:
```
tdata_01 |> slice_head(n = 5)
```

A spec_tbl_df: 5 x 9

grupo <chr>	tipo <chr>	nombre <chr>	X <dbl>	Y <dbl>	CMUN <chr>	CD <ch
turismo	hotel	NH Ciudad de la Imagen	-3.788176	40.39844	115	01
hosteleria	restaurant	Café Comercial	-3.702002	40.42873	079	01
hosteleria	pub	Sidrería la Camocha	-3.701686	40.42703	079	01
hosteleria	pub	Gran Cafe Santander	-3.696329	40.42760	079	01
comercio	supermarket	Alcampo	-3.706888	40.48035	079	08

0.6 Open Data

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

[27]:
```
data_to_save <- tdata_01
```

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

[28]:
```
caso <- "CU_45"
proceso <- '_05'
tarea <- "_02"
archivo <- ""
proper <- "_valoracion_sim"
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

[29]:
```
# 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/.

[30]:
```
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_02_valoracion_sim.csv'

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

[31]:
```
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, tidyr 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 * tidyr 1.3.0

0.8.3 Data structures

Objeto `data`

- Hay 24780 filas con información de las siguientes variables:
  - grupo
  - tipo
  - nombre
  - X
  - Y
  - CMUN
  - CDIS
  - t3_1
  - puntos

Observaciones generales sobre los datos

- Se dispone de una observación por establecimiento, y la puntuación se entiende promedio

0.8.4 Consideraciones para despliegue en piloto

- No aplica

0.8.5 Consideraciones para despliegue en producción

- Se deben obtener datos reales de las valoraciones de los establecimientos
- 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 asignado municipio y distrito a los establecimientos
- Se han simulado datos de valoraciones

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

- Se deben agregar los datos por municipios y por distritos para combinar con los movimientos de turistas y los recuentos de otros establecimientos

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
```