11.- Causal Anaysis_05_servicios_completo_v_01

June 16, 2023

#

CUxx_Nombre del caso de uso

Citizenlab Data Science Methodology > II - Data Processing Domain *** > # 11.- ECA - Exploratory Causal Analysis

Exploratory causal analysis (ECA) is the process of discovering the root causes of problems in order to identify appropriate solutions.

0.1 Tasks

Define the key challenge or setback

Determine the causes and effects of the key challenge

Use a diagram or graph to organize information

Formulate a response to the primary causes of your challenge

Review your process and address new causes and effects

0.2 File

- Input File: xxxxxxxxxx
- Output File: No aplica

0.2.1 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.

[1]:
```
Sys.setlocale(category = "LC_ALL", locale = "es_ES.UTF-8")
```

'LC_CTYPE=es_ES.UTF-8;LC_NUMERIC=C;LC_TIME=es_ES.UTF-8;LC_COLLATE=es_ES.UTF-8;LC_MONETARY=es_ES.UTF-8;LC_MESSAGES=en_US.UTF-8;LC_PAPER=es_ES.UTF-8;LC_NAME=C;LC_ADDRESS=C;LC_TELEPHONE=C;LC_MEASUREMENT
8;LC_IDENTIFICATION=C'

0.3 Settings

0.3.1 Libraries to use

[2]:
```
library(readr)
library(dplyr)
library(sf)
library(tidyr)
#library(stringr)
```

```
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 3.11.1, GDAL 3.6.2, PROJ 6.2.1; sf_use_s2() is TRUE

WARNING: different compile-time and runtime versions for GEOS found:

Linked against: 3.11.1-CAPI-1.17.1 compiled against: 3.8.0-CAPI-1.13.1

It is probably a good idea to reinstall sf, and maybe rgeos and rgdal too
```

0.3.2 Paths

[3]:
```
iPath <- "Data/Input/"
oPath <- "Data/Output/"
```

0.4 Data Load

OPCION A: Seleccionar fichero en ventana para mayor comodidad

Data load using the {tcltk} package. Ucomment the line if using this option

[4]:
```
# file_data <- tcltk::tk_choose.files(multi = FALSE)
```

OPCION B: Especificar el nombre de archivo

[5]:
```
iFile <- "CU_34_10_05_servicios_completo.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/CU_34_10_05_servicios_completo.csv
```

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

[6]:
```
data <- read_csv(file_data)
```

```
Rows: 272862 Columns: 19
  Column specification
```

```
Delimiter: ","
chr   (5): Servicio, CMUN, CDIS, CSEC, NSEC
dbl  (12): Futbol, nservicios, capacidad, tmed, prec, velmedia,
presMax, t1_…
lgl   (1): is_train
date  (1): Fecha

 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.
```

Visualizo los datos.

Estructura de los datos:

[7]:
```
data |> glimpse()
```

```
Rows: 272,862
Columns: 19
$ Fecha            <date> 2022-01-12, 2022-01-31, 2022-01-28,
2022-01-06, 2022…
$ Servicio         <chr> "Delivery", "Taxi", "Taxi",
"Delivery", "Delivery", "…
$ CMUN             <chr> "079", "079", "903", "079", "007",
"022", "079", "079…
```

```
$ CDIS             <chr> "14", "01", "01", "04", "04", "01",
"16", "01", "16",…
$ CSEC             <chr> "050", "048", "006", "080", "012",
"004", "041", "033…
$ Futbol           <dbl> 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0,…
$ nservicios       <dbl> 58, 5, 0, 14, 60, 50, 13, 4, 3, 9,
68, 12, 0, 1, 14, …
$ capacidad        <dbl> 80, 69, 56, 80, 70, 56, 80, 69, 69,
69, 80, 80, 69, 6…
$ tmed             <dbl> 7.366319, 8.823406, 7.854915,
4.226603, 4.982656, 7.2…
$ prec             <dbl> -0.009468616, 0.000000000,
0.000000000, 0.010181896, …
$ velmedia         <dbl> 1.5999961, 1.5114967, 2.2536168,
1.0279945, 1.0387037…
$ presMax          <dbl> 954.7939, 948.9795, 940.2553,
945.1884, 948.9570, 943…
$ t1_1             <dbl> 1094, 1251, 2232, 746, 1080, 2256,
692, 1270, 2229, 8…
$ t3_1             <dbl> 45.4360, 41.6091, 44.2016, 47.1729,
48.5361, 43.2877,…
$ NSEC             <chr> "Madrid - 14.050", "Madrid -
01.048", "Tres Cantos - …
$ area             <dbl> 38753.96, 15289.89, 124539.78,
89206.78, 24473.30, 34…
$ elevation        <dbl> 658, 635, 719, 710, 693, 710, 702,
635, 690, 710, 690…
$ densidad_hab_km2 <dbl> 28229.3737, 81818.7738, 17921.9842,
8362.5928, 44129.…
$ is_train         <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,
TRUE, TRUE, TRUE,…
```

Muestra de los primeros datos:

[8]:
```
data |> slice_head(n = 5)
```

A spec_tbl_df: 5 × 19

Fecha <date>	Servicio <chr>	CMUN <chr>	CDIS <chr>	CSEC <chr>	Futbol <dbl>	nservicios <dbl>	capacidad <dbl>	tme <d
2022-01-12	Delivery	079	14	050	1	58	80	7.3
2022-01-31	Taxi	079	01	048	0	5	69	8.8
2022-01-28	Taxi	903	01	006	0	0	56	7.8
2022-01-06	Delivery	079	04	080	0	14	80	4.2
2022-01-21	Delivery	007	04	012	1	60	70	4.9

0.5 Exploratory causal analysis

REFERENCE https://bookdown.org/paul/applied-causal-analysis/

Select columns

[9]:
```
# Seleccionamos las variables a analizar.
```

[ ]:
```

```

0.6 Data Save

- Solo si se han hecho cambios
- No aplica

Identificamos los datos a guardar

[10]:
```
data_to_save <- data
```

Estructura de nombre de archivos:

- Código del caso de uso, por ejemplo “CU_04”
- Número del proceso que lo genera, por ejemplo ”_06”.
- Resto del nombre del archivo de entrada
- Extensión del archivo

Ejemplo: ”CU_04_06_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) y que se ha transformado en el proceso 06

Importante mantener los guiones bajos antes de proceso, tarea, archivo y nombre

0.6.1 Proceso 11

[11]:
```
caso <- "CU_34"
proceso <- '_11'
tarea <- "_05"
archivo <- ""
proper <- "_servicios_completo"
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

[12]:
```
# file_save <- paste0(caso, proceso, tarea, tcltk::tkgetSaveFile(), proper,␣
↪extension)
# path_out <- paste0(oPath, file_save)
# write_csv(data_to_save_xxxx, 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/.

[13]:
```
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_34_11_05_servicios_completo.csv'

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

[14]:
```
path_in <- paste0(iPath, file_save)
file.copy(path_out, path_in, overwrite = TRUE)
```

TRUE

0.7 REPORT

A continuación se realizará un informe de las acciones realizadas

0.8 Main Actions Carried Out

- Se ha realizado un análisis causal básico

0.9 Main Conclusions

- Los datos son adecuados para los modelos que se preveen

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

[ ]:
```

```