08.- Data Split_CU_53_02_spi_v_01

June 13, 2023

#

CU53_impacto de las políticas de inversión en sanidad, infraestructuras y promoción turística en el SPI

Citizenlab Data Science Methodology > II - Data Processing Domain *** > # 08.- Data Split

Data Split is the process of selecting the appropriate division of the data set into train, test and validation set.

0.1 Tasks

Train and Test Rate Evaluation

Split Train and Test Datasets

0.2 Consideraciones casos CitizenLab programados en R

- Puede que algunas de las tareas de este proceso se realicen en los notebooks de los procesos de deploy al estar relacionadas con otras de esos procesos. En esos casos, en este notebook se referencia al notebook del proceso correspondiente
- Puede que el proceso no aplique a los ficheros del caso de uso, y se indique "No aplica" de forma generalizada.
- Si en el nombre de archivo del notebook no aparece ningún sufijo, el notebook se refiere al caso globalmente

0.3 File

- Input File: CU_53_06_02_spi
- Output File: CU_53_08_02_spi

0.4 Settings

0.4.1 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'

0.4.2 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
```

0.4.3 Paths

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

0.5 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_53_06_02_spi.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_53_06_02_spi.csv
```

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

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

```
Rows: 2364 Columns: 17
  Column specification


Delimiter: ","
dbl (17): rank_score_spi, score_spi, score_bhn, score_fow, score_opp,
score_…

 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: 2,364
Columns: 17
$ rank_score_spi <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA,
NA, NA, NA, 164, 16…
$ score_spi      <dbl> 65.24, 64.87, 64.55, 64.31, 63.62,
63.29, 62.65, 62.32,…
$ score_bhn      <dbl> 75.80, 75.35, 74.91, 74.48, 73.98,
73.43, 72.88, 72.35,…
$ score_fow      <dbl> 63.62, 63.18, 62.91, 62.46, 61.03,
60.52, 59.26, 58.70,…
$ score_opp      <dbl> 56.28, 56.07, 55.83, 55.98, 55.84,
55.92, 55.80, 55.92,…
$ score_nbmc     <dbl> 82.09, 82.01, 81.79, 81.53, 81.23,
80.90, 80.53, 80.09,…
$ score_ws       <dbl> 80.74, 79.84, 78.69, 77.99, 77.20,
76.59, 76.17, 75.89,…
$ score_sh       <dbl> 79.36, 78.86, 78.50, 77.63, 76.92,
75.92, 74.97, 74.06,…
$ score_ps       <dbl> 61.02, 60.68, 60.66, 60.78, 60.56,
60.31, 59.86, 59.37,…
$ score_abk      <dbl> 72.42, 72.50, 72.36, 72.27, 71.90,
71.67, 71.16, 70.61,…
$ score_aic      <dbl> 75.96, 74.44, 73.71, 72.87, 68.90,
67.40, 63.91, 62.46,…
```

```
$ score_hw       <dbl> 58.21, 57.91, 57.84, 57.25, 56.72,
56.19, 55.78, 55.39,…
$ score_eq       <dbl> 47.90, 47.87, 47.73, 47.45, 46.58,
46.82, 46.19, 46.33,…
$ score_pr       <dbl> 60.40, 60.38, 61.01, 61.49, 61.97,
62.86, 63.30, 64.26,…
$ score_pfc      <dbl> 62.27, 62.36, 62.39, 62.58, 62.46,
62.39, 62.05, 61.84,…
$ score_incl     <dbl> 42.97, 42.14, 41.39, 41.86, 41.48,
41.62, 41.70, 42.00,…
$ score_aae      <dbl> 59.50, 59.39, 58.52, 58.00, 57.46,
56.81, 56.17, 55.59,…
```

Muestra de los primeros datos:

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

A spec_tbl_df: 5 × 17

rank_score_spi <dbl>	score_spi <dbl>	score_bhn <dbl>	score_fow <dbl>	score_opp <dbl>	score_nbmc <dbl>	score_ <dbl
NA	65.24	75.80	63.62	56.28	82.09	80.74
NA	64.87	75.35	63.18	56.07	82.01	79.84
NA	64.55	74.91	62.91	55.83	81.79	78.69
NA	64.31	74.48	62.46	55.98	81.53	77.99
NA	63.62	73.98	61.03	55.84	81.23	77.20

0.6 Split Rate Evaluation

An soft evaluation is performed in order to estimate Split Rate.

0.6.1 Classification Rate Evaluation

[9]:
```
# Implementar código solo si aplica
```

Métricas para la Soft-Evaluation

[10]:
```
# Implementar código solo si aplica
```

Modelos para la Soft-Evaluation

[11]:
```
# Implementar código solo si aplica
```

Operation

[12]:
```
# Implementar código solo si aplica
```

0.6.2 Regression Rate Evaluation

Parámetros para la Soft-Evaluation

[13]:
```
# Implementar código solo si aplica
```

Métricas para la Soft-Evaluation

[14]:
```
# Implementar código solo si aplica
```

Modelos para la Soft-Evaluation

[15]:
```
# Implementar código solo si aplica
```

Operation

[16]:
```
# Implementar código solo si aplica
```

Clustering Rate Evaluation You do not use training and valing in unsupervised learning. There is no objective function in unsupervised learning to val the performance of the algorithm.

0.7 Split Train and Test datasets

Estimada el porcentaje de división (Rate of Split) procedemos a realiar la división correspondiente del fichero entre Train y Test.

Parámetros para el Split

[17]:
```
# Establecer la semilla aleatoria para reproducibilidad
set.seed(123)

# Proporción de división (porcentaje)
split_rate <- 0.8  # 80% para entrenamiento, 20% para prueba
```

Operation

[18]:
```
# Generate the train dataset
train_data <- data %>%
  slice_sample(prop = split_rate, replace = FALSE) %>%
  mutate(is_train = TRUE)

# Generate the test dataset
test_data <- data %>%
  anti_join(train_data)%>%
  mutate(is_train = FALSE)

data_to_save <- rbind(train_data, test_data)
```

```
Joining with `by = join_by(rank_score_spi, score_spi, score_bhn,
score_fow, score_opp, score_nbmc, score_ws, score_sh,
score_ps, score_abk, score_aic, score_hw, score_eq, score_pr, score_pfc,
score_incl, score_aae)`
```

[19]:
```
data_to_save
```

rank_score_spi <dbl>	score_spi <dbl>	score_bhn <dbl>	score_fow <dbl>	score_opp <dbl>	score_nbmc <dbl>	score_ws
NA	NA	NA	NA	NA	75.01	83.09
80	67.59	79.16	65.40	58.22	86.67	86.44
97	60.10	74.55	51.25	54.49	72.88	83.35
46	73.96	81.88	70.69	69.32	86.33	88.07
84	62.86	79.45	61.22	47.92	83.91	77.71
99	61.43	77.84	57.63	48.83	87.72	78.15
150	45.57	47.15	45.21	44.34	54.66	47.82
74	66.56	80.41	62.82	56.46	92.38	78.47
105	59.45	66.16	54.62	57.56	72.21	66.32
36	79.93	88.01	77.07	74.71	91.17	94.88
143	44.25	54.20	46.20	32.34	66.70	55.90
154	46.58	44.86	48.63	46.24	55.91	30.89
69	67.71	80.42	69.13	53.59	91.56	82.05
168	32.39	28.96	33.29	34.91	36.38	21.97
141	48.89	63.31	48.66	34.69	74.23	63.23
164	37.97	56.86	35.38	21.67	60.12	54.29
1	90.53	91.18	90.79	89.61	93.87	98.79
NA	NA	82.28	66.73	NA	85.51	88.71
64	70.85	79.02	71.16	62.36	85.50	83.44
75	66.15	68.20	62.45	67.78	73.34	71.66
125	47.13	44.10	46.95	50.33	57.05	41.28
86	65.13	75.15	61.19	59.06	77.30	83.87
27	82.26	92.60	83.01	71.16	95.39	98.89
81	63.98	76.05	61.00	54.90	84.32	79.66
144	48.89	52.80	54.34	39.54	74.38	56.89
102	57.05	78.17	50.37	42.62	91.17	80.63
30	79.96	89.24	75.67	74.96	95.15	89.88
126	54.08	54.01	57.65	50.59	66.19	55.23
25	83.02	87.77	81.22	80.07	96.34	90.46
78	66.45	83.38	61.98	54.01	94.83	83.59
NA	NA	NA	NA	NA	73.55	62.47
NA	NA	NA	NA	NA	73.11	61.69
NA	NA	NA	NA	NA	72.85	60.80
NA	NA	NA	NA	NA	72.52	60.08
NA	NA	NA	NA	NA	NA	NA
NA	NA	NA	NA	NA	74.09	80.55
NA	NA	NA	NA	NA	78.49	88.46
NA	NA	NA	NA	NA	NA	92.19
NA	NA	NA	NA	NA	NA	90.10
NA	NA	NA	NA	NA	NA	90.02
NA	NA	NA	NA	NA	NA	87.61
NA	NA	NA	NA	NA	NA	90.73
NA	NA	NA	NA	NA	NA	90.70
NA	NA	NA	NA	NA	NA	90.57
NA	NA	NA	NA	NA	85.79	89.11
NA	NA	NA	NA	NA	85.64	88.89
NA	NA	NA	NA	NA	NA	92.14
NA	NA	NA	NA	NA	NA	90.85
NA	NA	NA	NA	NA	NA	90.52
NA	NA	NA	NA	NA	NA	90.42

A tibble: 2362 × 18

0.8 Data Save

- Solo si se han hecho cambios
- No aplica

Identificamos los datos a guardar

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

0.8.1 Proceso 08

[20]:
```
caso <- "CU_53"
proceso <- '_08'
tarea <- "_02"
archivo <- ""
proper <- "_spi"
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

[21]:
```
# 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/.

[22]:
```
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_53_08_02_spi.csv'

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

[23]:
```
path_in <- paste0(iPath, file_save)
file.copy(path_out, path_in, overwrite = TRUE)
```

TRUE

0.9 REPORT

A continuación se realizará un informe de las acciones realizadas

0.10 Main Actions Carried Out

- No aplica el proceso al caso
- En caso de ser necesaria la división en train y test, el código queda preparado

0.11 Main Conclusions

- En este caso no se hace división de datos

0.12 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

[24]:
```
# incluir código
```