

10.- Imbalanced Analysis_CU_18_20_infra_meteo_v_01

June 13, 2023

#

CU18_Infraestructuras_eventos

Citizenlab Data Science Methodology > II - Data Processing Domain *** > # 10.- Imbalanced Analysis

Data Balancing is the process to obtain an adequate data balance if is required, in order to have the adequate amount of data that reflects the intrinsic structure of the problem to be solved.

0.1 Tasks

Imbalanced Analysis

Evaluate Imbalanced Classification Models

Select appropriate metrics

Data Balancing

- Undersampling the Majority Class
- Oversampling the Minority Class
- Mix under-oversampling
- Evaluate a model with random oversampling and undersampling

Cost-Sensitive Algorithms

0.2 File

- Input File: CU_18_09.3_20_diario_infra
- 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=es_ES.UTF-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

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

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

OPCION B: Especificar el nombre de archivo

```
[4]: iFile <- "CU_18_09.3_20_diario_infra.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_18_09.3_20_diario_infra.csv

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

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

Rows: 377727 Columns: 10

Column specification

Delimiter: ","

dbl (9): id_inf, capacidad, demanda, evento_infra, evento_zona, tmed, prec,...

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:

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

Rows: 377,727

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, ...

\$ 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, ...

\$ 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...

Muestra de los primeros datos:

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

	id_inf <dbl>	fecha <date>	capacidad <dbl>	demanda <dbl>	evento_infra <dbl>	evento_zona <dbl>	tmed <dbl>	p <dbl>
A spec_tbl_df: 5 × 10	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

0.5 Imbalanced Analysis

```
[ ]:
```

```
[8]: # Visualizando los datos para comprobar como de balanceados están
```

```
[9]: # summarize class distribution
```

```
[10]: # Generate and plot imbalanced classification dataset
```

0.6 Evaluate Imbalanced Classification Models

```
[ ]:
```

0.7 Undersampling the Majority Class

```
[11]: # example of undersampling the majority class
```

```
[12]: # Generate new data set
```

```
[ ]:
```

```
[13]: # Visualizando los datos para comprobar como de balanceados están
```

0.8 Oversampling the Minority Class

```
[14]: # example of oversampling the minority class
```

```
[15]: # Generate new data set
```

```
[ ]:
```

```
[16]: # Visualizando los datos para comprobar como de balanceados están
```

0.9 Combine Data Undersampling and Oversampling with SMOTEENN

```
[17]: # example of both undersampling and oversampling
```

```
[18]: # Generate new data set
```

```
[ ]:
```

```
[19]: # Visualizando los datos para comprobar como de balanceados están
```

0.10 Evaluating a model with random oversampling and undersampling

```
[ ]:
```

0.11 Cost-Sensitive Algorithms

```
[20]: # example of cost sensitive logistic regression for imbalanced classification
```

0.12 Data Save

- Solo si se han hecho cambios
- No aplica

Identificamos los datos a guardar

```
[21]: 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

0.12.1 Proceso 10

```
[22]: caso <- "CU_18"  
proceso <- '_10'  
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 sufixo2 si es necesario
- Cambiar datos por datos_xx si es necesario

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

```
[24]: # file_save <- paste0(caso, proceso, tarea, archivo, proper, extension)
      # path_out <- paste0(oPath, file_save)
      # write_csv(data_to_save_XXXXX, path_out)

      # cat('File saved as: ')
      # path_out
```

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

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

0.13 REPORT

A continuación se realizará un informe de las acciones realizadas

0.14 Main Actions Carried Out

- No aplica el proceso al caso

0.15 Main Conclusions

- Al no haber clasificación no procede analizar el desbalanceo.

0.16 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

```
[ ]:
```