09.1.- Data Cleansing-Basic_25_01_listas_espera_v_01

June 10, 2023

#

CU25_Modelo de gestión de Lista de Espera Quirúrgica

Citizenlab Data Science Methodology > II - Data Processing Domain *** > # 09.1.- Data Cleansing - Basic

Data Cleaning refers to identifying and correcting (or removing) errors in the dataset that may negatively impact a predictive model, replacing, modifying, or deleting the dirty or coarse data.

0.1 Tasks

0.2 Consideraciones casos CitizenLab programados en R

- La mayoría de las tareas de este proceso se han realizado en los notebooks del proceso 05 Data Collection porque eran necesarias para las tareas ETL. En esos casos, en este notebook se referencia al notebook del proceso 05 correspondiente
- Por tanto en los notebooks de este proceso de manera general se incluyen las comprobaciones necesarias, y comentarios si procede
- Las tareas del proceso se van a aplicar solo a los archivos que forman parte del despliegue, ya que hay muchos archivos intermedios que no procede pasar por este proceso
- El nombre de archivo del notebook hace referencia al nombre de archivo del proceso 05 al que se aplica este proceso, por eso pueden no ser correlativa la numeración
- Las comprobaciones se van a realizar teniendo en cuenta que el lenguaje utilizado en el despliegue de este caso es R

0.3 File

- Input File: CU_25_05_07_03_lista_espera_completo_split.csv
- Output File: No aplica

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.

[140]:
```
Sys.setlocale(category = "LC_ALL", locale = "es_ES.UTF-8")
```

'LC_COLLATE=es_ES.UTF-8;LC_CTYPE=es_ES.UTF-8;LC_MONETARY=es_ES.UTF-8;LC_NUMERIC=C;LC_TIME=es_ES.UTF-8'

0.4.2 Libraries to use

[141]:
```
library(readr)
library(dplyr)
library(sf)
library(tidyr)
library(stringr)
```

0.4.3 Paths

[142]:
```
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

[143]:
```
# file_data <- tcltk::tk_choose.files(multi = FALSE)
```

OPCION B: Especificar el nombre de archivo

[144]:
```
iFile <- "CU_25_08_01_lista_espera_completo_split.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_25_08_01_lista_espera_completo_split.csv
```

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

[145]:
```
data <- read.csv(file_data)
colnames(data)
```

1. 'Hospital' 2. 'Especialidad' 3. 'total_pacientes' 4. 'ano' 5. 'semana' 6. 'CODCNH' 7. 'id_area' 8. 'nombre_area' 9. 'cmunicipio' 10. 'Municipio' 11. 'CAMAS' 12. 'Clase' 13. 'Dependencia' 14. 'TAC' 15. 'RM' 16. 'GAM' 17. 'HEM' 18. 'ASD' 19. 'LIT' 20. 'BCO' 21. 'ALI' 22. 'SPECT' 23. 'PET' 24. 'MAMOS' 25. 'DO' 26. 'DIAL' 27. 'X' 28. 'Y' 29. 't3_1' 30. 't1_1' 31. 't2_1' 32. 't2_2' 33. 't4_1' 34. 't4_2' 35. 't4_3' 36. 't5_1' 37. 't6_1' 38. 't7_1' 39. 't8_1' 40. 't9_1' 41. 't10_1' 42. 't11_1' 43. 't12_1' 44. 'capacidad' 45. 'pacientes' 46. 'consultas' 47. 'hospitalizaciones' 48. 'Target' 49. 'is_train'

Visualizo los datos.

Estructura de los datos:

[146]:
```
data |> glimpse()
```

```
Rows: 55,680
Columns: 49
$ Hospital          <chr> "HOSPITAL REY JUAN CARLOS",
"HOSPITAL CENTRAL DE LA …
$ Especialidad      <chr> "Urología", "Odontoestomatología",
"Ginecología", "D…
$ total_pacientes   <int> 344, 0, 52, 37, 0, 4, 0, 718, 0,
271, 108, 0, 34, 86…
$ ano               <int> 2021, 2020, 2021, 2021, 2021, 2020,
2021, 2020, 2021…
$ semana            <int> 30, 36, 49, 23, 3, 5, 50, 7, 35, 1,
42, 10, 21, 33, …
$ CODCNH            <int> 281348, 280724, 281292, 281292,
281236, 280724, 2807…
$ id_area           <int> 8, 7, 11, 11, 11, 7, 3, 6, 1, 2, 2,
8, 11, 11, 1, 3,…
$ nombre_area       <chr> "Sur-Oeste I", "Centro-Oeste", "Sur
Ii", "Sur Ii", "…
$ cmunicipio        <int> 280920, 280796, 280133, 280133,
281610, 280796, 2800…
$ Municipio         <chr> "Móstoles", "Madrid", "Aranjuez",
"Aranjuez", "Valde…
$ CAMAS             <int> 382, 475, 98, 98, 182, 475, 507,
613, 269, 1143, 156…
$ Clase             <chr> "Hospitales Generales", "Hospitales
Generales", "Hos…
$ Dependencia       <chr> "SERVICIOS E INSTITUTOS DE SALUD DE
LAS COMUNIDADES …
$ TAC               <int> 2, 2, 1, 1, 1, 2, 3, 3, 0, 0, 1, 2,
6, 6, 1, 3, 4, 1…
$ RM                <int> 3, 2, 1, 1, 2, 2, 2, 3, 0, 0, 0, 2,
5, 5, 1, 2, 4, 1…
$ GAM               <int> 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1,
2, 2, 0, 0, 2, 0…
$ HEM               <int> 1, 2, 0, 0, 1, 2, 1, 2, 0, 0, 0, 1,
3, 3, 0, 1, 1, 0…
$ ASD               <int> 2, 1, 1, 1, 1, 1, 1, 3, 0, 0, 0, 1,
2, 2, 0, 1, 2, 1…
$ LIT               <int> 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1,
1, 1, 0, 0, 1, 0…
$ BCO               <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0…
$ ALI               <int> 1, 2, 0, 0, 0, 2, 0, 4, 0, 0, 0, 0,
3, 3, 0, 2, 2, 0…
```

```
$ SPECT             <int> 1, 1, 0, 0, 0, 1, 0, 4, 0, 0, 0, 0,
3, 3, 0, 0, 0, 0…
$ PET               <int> 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
1, 1, 0, 0, 0, 0…
$ MAMOS             <int> 2, 1, 1, 1, 1, 1, 2, 2, 0, 0, 1, 2,
3, 3, 1, 1, 3, 1…
$ DO                <int> 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1,
2, 2, 0, 1, 2, 0…
$ DIAL              <int> 20, 24, 13, 13, 17, 24, 28, 31, 0,
0, 0, 28, 43, 43,…
$ X                 <dbl> -3.870412, -3.745529, -3.610795,
-3.610795, -3.69744…
$ Y                 <dbl> 40.33920, 40.38791, 40.05726,
40.05726, 40.19884, 40…
$ t3_1              <dbl> 42.34715, 45.37878, 42.06149,
42.06149, 42.06149, 45…
$ t1_1              <int> 532487, 511605, 899702, 899702,
899702, 511605, 3830…
$ t2_1              <dbl> 0.5122493, 0.5296804, 0.5240445,
0.5240445, 0.524044…
$ t2_2              <dbl> 0.4877507, 0.4703198, 0.4759555,
0.4759555, 0.475955…
$ t4_1              <dbl> 0.1659665, 0.1054260, 0.1540793,
0.1540793, 0.154079…
$ t4_2              <dbl> 0.6371549, 0.6742432, 0.6753787,
0.6753787, 0.675378…
$ t4_3              <dbl> 0.1968769, 0.2203341, 0.1705449,
0.1705449, 0.170544…
$ t5_1              <dbl> 0.1137647, 0.1744493, 0.1747059,
0.1747059, 0.174705…
$ t6_1              <dbl> 0.1604646, 0.2629599, 0.2641879,
0.2641879, 0.264187…
$ t7_1              <dbl> 0.05422176, 0.05481008, 0.04898547,
0.04898547, 0.04…
$ t8_1              <dbl> 0.04120012, 0.04653221, 0.03679912,
0.03679912, 0.03…
$ t9_1              <dbl> 0.3348780, 0.4914365, 0.3346063,
0.3346063, 0.334606…
$ t10_1             <dbl> 0.13692541, 0.12170996, 0.15173209,
0.15173209, 0.15…
$ t11_1             <dbl> 0.5072726, 0.4915713, 0.5024130,
0.5024130, 0.502413…
$ t12_1             <dbl> 0.5849309, 0.5597213, 0.5900028,
0.5900028, 0.590002…
$ capacidad         <int> 17, 0, 8, 5, 0, 5, 1, 24, 6, 6, 30,
4, 2, 15, 20, 6,…
$ pacientes         <int> 1447, 1211, 1293, 1501, 1240, 1504,
1502, 1533, 1463…
```

```
$ consultas          <int> 573, 45, 108, 103, 44, 42, 36,
1119, 34, 466, 220, 6…
$ hospitalizaciones <int> 12, 0, 2, 2, 0, 1, 0, 4, 0, 12, 3,
0, 2, 4, 1, 2, 15…
$ Target             <dbl> 54.45, 0.00, 37.96, 23.14, 0.00,
6.25, 0.00, 78.20, …
$ is_train           <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,
TRUE, TRUE, TRUE…
```

Muestra de los primeros datos:

[147]:
```
data |> slice_head(n = 5)
```

A data.frame: 5 × 49

Hospital <chr>	Especialidad <chr>	total <int
HOSPITAL REY JUAN CARLOS	Urología	344
HOSPITAL CENTRAL DE LA DEFENSA GOMEZ ULLA	Odontoestomatología	0
HOSPITAL UNIVERSITARIO DEL TAJO	Ginecología	52
HOSPITAL UNIVERSITARIO DEL TAJO	Dermatología	37
HOSPITAL UNIVERSITARIO INFANTA ELENA	Odontoestomatología	0

0.6 Text data analysis

Select columns

[148]:
```
# Select column
text_columns <- sapply(data, is.character)
```

Operation

[149]:
```
# Analizar datos de texto y verificar su corrección
# e.g. faltas ortografía, etc
# pasar a mayúsculas todas las columnas de texto

data[, text_columns] <- lapply(data[, text_columns], function(x) toupper(x))
```

0.7 Delete Columns Needless/Irrelevant/Private

Select columns

No aplica

Operation

[150]:
```
# Eliminamos columnas que consideramos irrelevantes o innecesarias
column_names <- colnames(data)
column_names
```

1. 'Hospital' 2. 'Especialidad' 3. 'total_pacientes' 4. 'ano' 5. 'semana' 6. 'CODCNH' 7. 'id_area' 8. 'nombre_area' 9. 'cmunicipio' 10. 'Municipio' 11. 'CAMAS' 12. 'Clase' 13. 'Dependencia' 14. 'TAC' 15. 'RM' 16. 'GAM' 17. 'HEM' 18. 'ASD' 19. 'LIT' 20. 'BCO' 21. 'ALI' 22. 'SPECT'

23. 'PET' 24. 'MAMOS' 25. 'DO' 26. 'DIAL' 27. 'X' 28. 'Y' 29. 't3_1' 30. 't1_1' 31. 't2_1' 32. 't2_2' 33. 't4_1' 34. 't4_2' 35. 't4_3' 36. 't5_1' 37. 't6_1' 38. 't7_1' 39. 't8_1' 40. 't9_1' 41. 't10_1' 42. 't11_1' 43. 't12_1' 44. 'capacidad' 45. 'pacientes' 46. 'consultas' 47. 'hospitalizaciones' 48. 'Target' 49. 'is_train'

Todas las columnas son relevantes, por lo que no se elimina ninguna.

0.8 Inconsistent Data

No aplica

Select columns and value

[ ]:

Operation

[ ]:

0.9 Expected values

[ ]:

0.10 Zeros

No aplica. ETL satisface los requisitos de calidad de los datos para valores cero.

0.11 Single Value

[151]:
```
# We obtain the number of different values of each column
distinct_counts <- sapply(data, function(x) n_distinct(x, na.rm = TRUE))
distinct_counts
```

Hospital 29 **Especialidad** 16 **total_pacientes** 3528 **ano** 3 **semana** 51 **CODCNH** 29 **id_area** 10 **nombre_area** 10 **cmunicipio** 16 **Municipio** 16 **CAMAS** 28 **Clase** 3 **Dependencia** 3 **TAC** 6 **RM** 6 **GAM** 5 **HEM** 4 **ASD** 4 **LIT** 2 **BCO** 1 **ALI** 5 **SPECT** 5 **PET** 2 **MAMOS** 5 **DO** 3 **DIAL** 15 **X** 29 **Y** 29 **t3_1** 10 **t1_1** 10 **t2_1** 10 **t2_2** 10 **t4_1** 10 **t4_2** 10 **t4_3** 10 **t5_1** 10 **t6_1** 10 **t7_1** 10 **t8_1** 10 **t9_1** 10 **t10_1** 10 **t11_1** 10 **t12_1** 10 **capacidad** 48 **pacientes** 632 **consultas** 4475 **hospitalizaciones** 322 **Target** 12635 **is_train** 2

[152]:
```
# Identify columns with a single unique value
single_value_columns <- which(distinct_counts == 1)
print(single_value_columns)
# Remove columns with a single unique value
data <- data[, -single_value_columns]
colnames(data)
```

```
BCO
 20
```

1. 'Hospital' 2. 'Especialidad' 3. 'total_pacientes' 4. 'ano' 5. 'semana' 6. 'CODCNH' 7. 'id_area' 8. 'nombre_area' 9. 'cmunicipio' 10. 'Municipio' 11. 'CAMAS' 12. 'Clase' 13. 'Dependencia' 14. 'TAC' 15. 'RM' 16. 'GAM' 17. 'HEM' 18. 'ASD' 19. 'LIT' 20. 'ALI' 21. 'SPECT' 22. 'PET' 23. 'MAMOS' 24. 'DO' 25. 'DIAL' 26. 'X' 27. 'Y' 28. 't3_1' 29. 't1_1' 30. 't2_1' 31. 't2_2' 32. 't4_1' 33. 't4_2' 34. 't4_3' 35. 't5_1' 36. 't6_1' 37. 't7_1' 38. 't8_1' 39. 't9_1' 40. 't10_1' 41. 't11_1' 42. 't12_1' 43. 'capacidad' 44. 'pacientes' 45. 'consultas' 46. 'hospitalizaciones' 47. 'Target' 48. 'is_train'

0.12 Very Few Values

Select rate

[153]:
```
rate <- 0.00005
```

Operation

[154]:
```
# Summarize columns that have unique values that are less than
# "porcentage_threshold" percent of the number of rows.


# Show features with over rate rows being the same value
distinct_counts <- sapply(data, function(x) n_distinct(x, na.rm = TRUE))

# Calculate the rate of different values
distinct_rate <- distinct_counts / nrow(data)

# Identify columns with a rate lower than the threshold, excluding 'is_train'␣
 ↪and 'Target' columns
columns_to_remove <- names(distinct_rate[distinct_rate < rate & !
 ↪(names(distinct_rate) %in% c("is_train", "Target"))])

# Print the columns to be removed
print(columns_to_remove)

# Remove columns with a low rate of different values
data <- data[, !(names(data) %in% columns_to_remove)]
```

```
[1] "LIT" "PET"
```

[155]:
```
# Summarize the number of unique values in each column
# followed by the percentage of unique values for each
# variable as a percentage of the total number of rows
# in the dataset.

# Calculate the number of unique values per column
unique_counts <- sapply(data, function(x) n_distinct(x, na.rm = TRUE))

# Calculate the percentage of unique values for each variable
percentage_unique <- (unique_counts / nrow(data)) * 100
```

```r
# Create a summary data frame
summary_df <- data.frame(
  Column = names(unique_counts),
  Unique_Count = unique_counts,
  Percentage_Unique = percentage_unique
)

# Print the summary data frame
print(summary_df)
```

	Column	Unique_Count	Percentage_Unique
Hospital	Hospital	29	0.052083333
Especialidad	Especialidad	16	0.028735632
total_pacientes	total_pacientes	3528	6.336206897
ano	ano	3	0.005387931
semana	semana	51	0.091594828
CODCNH	CODCNH	29	0.052083333
id_area	id_area	10	0.017959770
nombre_area	nombre_area	10	0.017959770
cmunicipio	cmunicipio	16	0.028735632
Municipio	Municipio	16	0.028735632
CAMAS	CAMAS	28	0.050287356
Clase	Clase	3	0.005387931
Dependencia	Dependencia	3	0.005387931
TAC	TAC	6	0.010775862
RM	RM	6	0.010775862
GAM	GAM	5	0.008979885
HEM	HEM	4	0.007183908
ASD	ASD	4	0.007183908
ALI	ALI	5	0.008979885
SPECT	SPECT	5	0.008979885
MAMOS	MAMOS	5	0.008979885
DO	DO	3	0.005387931
DIAL	DIAL	15	0.026939655
X	X	29	0.052083333
Y	Y	29	0.052083333
t3_1	t3_1	10	0.017959770
t1_1	t1_1	10	0.017959770
t2_1	t2_1	10	0.017959770
t2_2	t2_2	10	0.017959770
t4_1	t4_1	10	0.017959770
t4_2	t4_2	10	0.017959770
t4_3	t4_3	10	0.017959770
t5_1	t5_1	10	0.017959770
t6_1	t6_1	10	0.017959770
t7_1	t7_1	10	0.017959770
t8_1	t8_1	10	0.017959770

t9_1	t9_1	10	0.017959770
t10_1	t10_1	10	0.017959770
t11_1	t11_1	10	0.017959770
t12_1	t12_1	10	0.017959770
capacidad	capacidad	48	0.086206897
pacientes	pacientes	632	1.135057471
consultas	consultas	4475	8.036997126
hospitalizaciones	hospitalizaciones	322	0.578304598
Target	Target	12635	22.692169540
is_train	is_train	2	0.003591954

Select percent of the number of rows

[156]:
```
# Select percent of the number of rows
percentage_threshold_value <- 0.0005
```

Operation

[157]:
```
# Get the column names to remove
columns_to_remove <- summary_df$Column[summary_df$Percentage_Unique <␣
 ↪percentage_threshold_value]
print(columns_to_remove)
# Remove the columns from the data
data <- data[, !(names(data) %in% columns_to_remove)]
```

```
character(0)
```

Check final columns

[159]:
```
# Delete columns that have unique values that are less than
# "porcentage_threshold" percent of the number of rows.
#
# NOTE: WE MUST EXCLUDE TARGET COLUMN TO DELETE
colnames(data)
```

1. 'Hospital' 2. 'Especialidad' 3. 'total_pacientes' 4. 'ano' 5. 'semana' 6. 'CODCNH' 7. 'id_area' 8. 'nombre_area' 9. 'cmunicipio' 10. 'Municipio' 11. 'CAMAS' 12. 'Clase' 13. 'Dependencia' 14. 'TAC' 15. 'RM' 16. 'GAM' 17. 'HEM' 18. 'ASD' 19. 'ALI' 20. 'SPECT' 21. 'MAMOS' 22. 'DO' 23. 'DIAL' 24. 'X' 25. 'Y' 26. 't3_1' 27. 't1_1' 28. 't2_1' 29. 't2_2' 30. 't4_1' 31. 't4_2' 32. 't4_3' 33. 't5_1' 34. 't6_1' 35. 't7_1' 36. 't8_1' 37. 't9_1' 38. 't10_1' 39. 't11_1' 40. 't12_1' 41. 'capacidad' 42. 'pacientes' 43. 'consultas' 44. 'hospitalizaciones' 45. 'Target' 46. 'is_train'

0.13 Low Variance

A) Calculating variances

[160]:
```
# Filter numeric columns
numeric_columns <- sapply(data, is.numeric)

# Calculate variance for numeric columns
```

```
variance <- sapply(data[, numeric_columns], var, na.rm = TRUE)

# Print the variance values
print(variance)
```

total_pacientes	ano	semana	CODCNH
7.916352e+05	5.799410e-01	2.232348e+02	2.609947e+05
id_area	cmunicipio	CAMAS	TAC
1.036880e+01	1.348231e+05	9.577061e+04	1.819295e+00
RM	GAM	HEM	ASD
1.424520e+00	1.008342e+00	9.465093e-01	8.513827e-01
ALI	SPECT	MAMOS	DO
1.286587e+00	1.005963e+00	9.369966e-01	4.899018e-01
DIAL	X	Y	t3_1
1.994328e+02	2.441187e-02	1.432827e-02	2.670148e+00
t1_1	t2_1	t2_2	t4_1
4.643055e+10	1.081162e-04	1.079681e-04	5.199907e-04
t4_2	t4_3	t5_1	t6_1
1.532418e-04	6.024165e-04	5.255911e-04	1.290188e-03
t7_1	t8_1	t9_1	t10_1
8.101956e-05	9.758045e-05	1.106921e-02	4.901374e-04
t11_1	t12_1	capacidad	pacientes
2.525667e-04	3.700791e-04	1.501380e+03	8.276700e+03
consultas	hospitalizaciones	Target	
1.781602e+06	6.888084e+02	1.775604e+03	

B) Automatic calculation and representation of variances

Define thresholds to check

[161]:
```
# define thresholds to check
variance_threshold = 0.00001
```

C) Delete variables with low variance

Select column

Operation

[163]:
```
# drop columns with low variance

# Get the column names with variance lower than the threshold
low_variance_columns <- names(variance[variance < variance_threshold])

# Remove the low variance columns from the dataset
data <- data[, !names(data) %in% low_variance_columns]
colnames(data)
```

1. 'Hospital' 2. 'Especialidad' 3. 'total_pacientes' 4. 'ano' 5. 'semana' 6. 'CODCNH' 7. 'id_area' 8. 'nombre_area' 9. 'cmunicipio' 10. 'Municipio' 11. 'CAMAS' 12. 'Clase' 13. 'Dependencia' 14. 'TAC' 15. 'RM' 16. 'GAM' 17. 'HEM' 18. 'ASD' 19. 'ALI' 20. 'SPECT' 21. 'MAMOS' 22. 'DO'

23. 'DIAL' 24. 'X' 25. 'Y' 26. 't3_1' 27. 't1_1' 28. 't2_1' 29. 't2_2' 30. 't4_1' 31. 't4_2' 32. 't4_3' 33. 't5_1' 34. 't6_1' 35. 't7_1' 36. 't8_1' 37. 't9_1' 38. 't10_1' 39. 't11_1' 40. 't12_1' 41. 'capacidad' 42. 'pacientes' 43. 'consultas' 44. 'hospitalizaciones' 45. 'Target' 46. 'is_train'

0.14 Duplicates

Entendido como ERROR –> Eliminar duplicados

0.14.1 Identificación de Duplicates

[166]:
```
# Remove duplicated rows
data <- data[!duplicated(data), ]
```

0.14.2 Eliminación de primera fila duplicada

No aplica

0.14.3 Eliminación de todas las filas duplicadas

No aplica

0.14.4 Eliminación de filas duplicadas en columnas concretas

Select columns

No aplica

Operation

No aplica

0.15 Data Save

- Solo si se han hecho cambios
- No aplica

Identificamos los datos a guardar

[168]:
```
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

0.15.1 Proceso 09.1

[33]:
```
caso <- "CU_25"
proceso <- '_09.1'
tarea <- "_01"
archivo <- "_lista_espera_completo_clean"
proper <- "_v_01"
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_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/.

[170]:
```
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_25_09.1_01_lista_espera_completo_clean_v_01.csv'

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

[171]:
```
path_in <- paste0(iPath, file_save)
file.copy(path_out, path_in, overwrite = TRUE)
```

TRUE

0.16 REPORT

A continuación se realizará un informe de las acciones realizadas

0.17 Main Actions Carried Out

- Si eran necesarias se han realizado en el proceso 05 por cuestiones de eficiencia

0.18 Main Conclusions

- Los datos están limpios para el despliegue

0.19 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

[ ]: