09.2.- Data Cleansing-Missing-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 *** > # 09.2.- Data Cleansing - Missing

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_09.1_01_lista_espera_completo_clean_v_01.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.

[27]:
```
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

[28]:
```
library(readr)
library(dplyr)
library(sf)
library(tidyr)
library(stringr)
library(zoo)
library(imputeTS)
```

0.4.3 Paths

[29]:
```
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

[30]:
```
# file_data <- tcltk::tk_choose.files(multi = FALSE)
```

OPCION B: Especificar el nombre de archivo

[31]:
```
iFile <- "CU_25_09.1_01_lista_espera_completo_clean_v_01.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_09.1_01_lista_espera_completo_clean_v_01.csv
```

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

[32]:
```
data <- read.csv(file_data)
```

Visualizo los datos.

Estructura de los datos:

[33]:
```
data |> glimpse()
```

```
Rows: 55,680
Columns: 46
$ 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…
$ 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…
$ 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:

[34]:
```
data |> slice_head(n = 5)
```

	Hospital <chr>	Especialidad <chr>
A data.frame: 5 × 46	HOSPITAL REY JUAN CARLOS	UROLOGÍA
	HOSPITAL CENTRAL DE LA DEFENSA GOMEZ ULLA	ODONTOESTOMATOLOG
	HOSPITAL UNIVERSITARIO DEL TAJO	GINECOLOGÍA
	HOSPITAL UNIVERSITARIO DEL TAJO	DERMATOLOGÍA
	HOSPITAL UNIVERSITARIO INFANTA ELENA	ODONTOESTOMATOLOG

0.6 Missing Values

0.6.1 Missing Values Identification

Missing Values Per Sample

[35]:
```
# Create a dataframe with ID and number of missing values per sample
missing_per_sample_df <- data.frame(ID = 1:nrow(data), NAs = rowSums(is.
  ↪na(data)))

# Filter rows with more than one missing value
filtered_missing_per_sample_df <-␣
  ↪missing_per_sample_df[missing_per_sample_df$NAs > 1, ]

# Print the filtered dataframe
print(filtered_missing_per_sample_df)
```

```
        ID NAs
138    138   4
241    241   4
577    577   4
579    579   4
614    614   4
759    759   4
786    786   4
951    951   4
1087  1087   4
1183  1183   4
1348  1348   4
1487  1487   4
1920  1920   4
2049  2049   4
2081  2081   4
2102  2102   4
2351  2351   4
2414  2414   4
2535  2535   4
2981  2981   4
3115  3115   4
3362  3362   4
3367  3367   4
3445  3445   4
```

3475	3475	4
3495	3495	4
3542	3542	4
3587	3587	4
3672	3672	4
3745	3745	4
3905	3905	4
4030	4030	4
4166	4166	4
4287	4287	4
4352	4352	4
4611	4611	4
4660	4660	4
4799	4799	4
5021	5021	4
5023	5023	4
5051	5051	4
5073	5073	4
5193	5193	4
5196	5196	4
5199	5199	4
5401	5401	4
5484	5484	4
5709	5709	4
5803	5803	4
6016	6016	4
6092	6092	4
6144	6144	4
6292	6292	4
6391	6391	4
6491	6491	4
6556	6556	4
6661	6661	4
6680	6680	4
6690	6690	4
6757	6757	4
7052	7052	4
7055	7055	4
7260	7260	4
7330	7330	4
7619	7619	4
7668	7668	4
7706	7706	4
7851	7851	4
8077	8077	4
8263	8263	4
8329	8329	4
8341	8341	4

```
8426   8426   4
8560   8560   4
8599   8599   4
8610   8610   4
8648   8648   4
8756   8756   4
8782   8782   4
9209   9209   4
9213   9213   4
9553   9553   4
9635   9635   4
9934   9934   4
10404 10404   4
10468 10468   4
10512 10512   4
10548 10548   4
10616 10616   4
10729 10729   4
10991 10991   4
11214 11214   4
11218 11218   4
11269 11269   4
11310 11310   4
11768 11768   4
11846 11846   4
11907 11907   4
11943 11943   4
12014 12014   4
12035 12035   4
12065 12065   4
12091 12091   4
12212 12212   4
12247 12247   4
12282 12282   4
12362 12362   4
12558 12558   4
12791 12791   4
12917 12917   4
12926 12926   4
12950 12950   4
13394 13394   4
13579 13579   4
13708 13708   4
13723 13723   4
13739 13739   4
13897 13897   4
13953 13953   4
13979 13979   4
```

```
14131 14131   4
14161 14161   4
14419 14419   4
14454 14454   4
14462 14462   4
14589 14589   4
14845 14845   4
14868 14868   4
14950 14950   4
15031 15031   4
15251 15251   4
15365 15365   4
15462 15462   4
15496 15496   4
15562 15562   4
15779 15779   4
15802 15802   4
16098 16098   4
16147 16147   4
16200 16200   4
16310 16310   4
16335 16335   4
16591 16591   4
16720 16720   4
16775 16775   4
17161 17161   4
17233 17233   4
17341 17341   4
17521 17521   4
17709 17709   4
17723 17723   4
17738 17738   4
17832 17832   4
17876 17876   4
18144 18144   4
18204 18204   4
18264 18264   4
18301 18301   4
18351 18351   4
18430 18430   4
18465 18465   4
18580 18580   4
18656 18656   4
18669 18669   4
18835 18835   4
19195 19195   4
19406 19406   4
19808 19808   4
```

```
19829 19829   4
19913 19913   4
19968 19968   4
20033 20033   4
20086 20086   4
20170 20170   4
20199 20199   4
20292 20292   4
20373 20373   4
20464 20464   4
20639 20639   4
20704 20704   4
20770 20770   4
20771 20771   4
20958 20958   4
21050 21050   4
21094 21094   4
21130 21130   4
21224 21224   4
21403 21403   4
21551 21551   4
21580 21580   4
21623 21623   4
21632 21632   4
21658 21658   4
21716 21716   4
21754 21754   4
21774 21774   4
21809 21809   4
22119 22119   4
22222 22222   4
22233 22233   4
22396 22396   4
22515 22515   4
22525 22525   4
22572 22572   4
22648 22648   4
22849 22849   4
23045 23045   4
23235 23235   4
23465 23465   4
23468 23468   4
23778 23778   4
23978 23978   4
24003 24003   4
24101 24101   4
24208 24208   4
24217 24217   4
```

```
24265 24265   4
24269 24269   4
24405 24405   4
24425 24425   4
24460 24460   4
24491 24491   4
24514 24514   4
24721 24721   4
24730 24730   4
24832 24832   4
25027 25027   4
25190 25190   4
25232 25232   4
25623 25623   4
25684 25684   4
25715 25715   4
25754 25754   4
26083 26083   4
26154 26154   4
26421 26421   4
26579 26579   4
27427 27427   4
27498 27498   4
27644 27644   4
27673 27673   4
27722 27722   4
28097 28097   4
28285 28285   4
28492 28492   4
28648 28648   4
28813 28813   4
28978 28978   4
29002 29002   4
29005 29005   4
29107 29107   4
29125 29125   4
29176 29176   4
29195 29195   4
29218 29218   4
29241 29241   4
29382 29382   4
29400 29400   4
29521 29521   4
29561 29561   4
29815 29815   4
29867 29867   4
29927 29927   4
30083 30083   4
```

```
30394 30394   4
30499 30499   4
30920 30920   4
30924 30924   4
31182 31182   4
31350 31350   4
31451 31451   4
31723 31723   4
31746 31746   4
31963 31963   4
32224 32224   4
32286 32286   4
32334 32334   4
32437 32437   4
32767 32767   4
32866 32866   4
32996 32996   4
33228 33228   4
33317 33317   4
33412 33412   4
33502 33502   4
33557 33557   4
33795 33795   4
33828 33828   4
34531 34531   4
34653 34653   4
35002 35002   4
35067 35067   4
35148 35148   4
35223 35223   4
35339 35339   4
35385 35385   4
35939 35939   4
35983 35983   4
36145 36145   4
36226 36226   4
36231 36231   4
36265 36265   4
36349 36349   4
36417 36417   4
36632 36632   4
36708 36708   4
36838 36838   4
36959 36959   4
37047 37047   4
37089 37089   4
37278 37278   4
37431 37431   4
```

```
37672 37672   4
37806 37806   4
37950 37950   4
38274 38274   4
38428 38428   4
38590 38590   4
38596 38596   4
38673 38673   4
38683 38683   4
38815 38815   4
39206 39206   4
39390 39390   4
39502 39502   4
39591 39591   4
39661 39661   4
39674 39674   4
39800 39800   4
40114 40114   4
40135 40135   4
40226 40226   4
40366 40366   4
40371 40371   4
40384 40384   4
40430 40430   4
40574 40574   4
40750 40750   4
40787 40787   4
40968 40968   4
41104 41104   4
41282 41282   4
41283 41283   4
41483 41483   4
41793 41793   4
41999 41999   4
42042 42042   4
42267 42267   4
42379 42379   4
42438 42438   4
42560 42560   4
42563 42563   4
42653 42653   4
42665 42665   4
42710 42710   4
42718 42718   4
42861 42861   4
42931 42931   4
42939 42939   4
42963 42963   4
```

```
43002 43002   4
43056 43056   4
43069 43069   4
43236 43236   4
43237 43237   4
43285 43285   4
43359 43359   4
43471 43471   4
43857 43857   4
43914 43914   4
44150 44150   4
44218 44218   4
44225 44225   4
44251 44251   4
44725 44725   4
44835 44835   4
44881 44881   4
44919 44919   4
45095 45095   4
45249 45249   4
45271 45271   4
45289 45289   4
45319 45319   4
45402 45402   4
45460 45460   4
45570 45570   4
45614 45614   4
45787 45787   4
46161 46161   4
46194 46194   4
46297 46297   4
46349 46349   4
46394 46394   4
46665 46665   4
46690 46690   4
46863 46863   4
46888 46888   4
47034 47034   4
47131 47131   4
47207 47207   4
47381 47381   4
47504 47504   4
47583 47583   4
47608 47608   4
47658 47658   4
47681 47681   4
47884 47884   4
48110 48110   4
```

```
48194 48194   4
48217 48217   4
48262 48262   4
48414 48414   4
48681 48681   4
48706 48706   4
48830 48830   4
49060 49060   4
49085 49085   4
49154 49154   4
49258 49258   4
49287 49287   4
49305 49305   4
49583 49583   4
50036 50036   4
50114 50114   4
50247 50247   4
50275 50275   4
50454 50454   4
50738 50738   4
50833 50833   4
51025 51025   4
51254 51254   4
51418 51418   4
51836 51836   4
51999 51999   4
52050 52050   4
52306 52306   4
52440 52440   4
52582 52582   4
52627 52627   4
52923 52923   4
53175 53175   4
53341 53341   4
53541 53541   4
53565 53565   4
53798 53798   4
53827 53827   4
53997 53997   4
54016 54016   4
54123 54123   4
54150 54150   4
54304 54304   4
54343 54343   4
54621 54621   4
54694 54694   4
54778 54778   4
54967 54967   4
```

```
55018 55018   4
55114 55114   4
55248 55248   4
55369 55369   4
55396 55396   4
55432 55432   4
55478 55478   4
55581 55581   4
```

Missing Values Per Feature

[36]:
```r
# Calculate the number of missing values per feature
missing_per_feature <- colSums(is.na(data))

# Print the number of missing values per feature
print(missing_per_feature)
```

```
         Hospital      Especialidad   total_pacientes               ano
                0                 0               464                 0
           semana           CODCNH         id_area       nombre_area
                0                 0                 0                 0
       cmunicipio         Municipio           CAMAS             Clase
                0                 0                 0                 0
      Dependencia               TAC              RM               GAM
                0                 0                 0                 0
              HEM               ASD             ALI             SPECT
                0                 0                 0                 0
           MAMOS                DO            DIAL                 X
                0                 0                 0                 0
               Y              t3_1            t1_1              t2_1
                0                 0                 0                 0
            t2_2              t4_1            t4_2              t4_3
                0                 0                 0                 0
            t5_1              t6_1            t7_1              t8_1
                0                 0                 0                 0
            t9_1             t10_1           t11_1             t12_1
                0                 0                 0                 0
      capacidad         pacientes         consultas hospitalizaciones
                0                 0               464               464
          Target          is_train
             464                 0
```

Zero Missing Values

[37]:
```r
# Detecting columns with minimum value of zero (0).

# Calculate the frequency of zeros in each column
zero_counts <- sapply(data, function(x) sum(x == 0, na.rm = TRUE))
```

```r
# Calculate the proportion of zeros in each column
zero_proportions <- zero_counts / nrow(data)

# Set a threshold for the proportion of zeros
zero_threshold <- 0.9  # Adjust as needed

# Identify columns with a high proportion of zeros
columns_with_high_zeros <- names(zero_proportions[zero_proportions >=␣
 ↪zero_threshold])

# Print the columns with a high proportion of zeros
print(columns_with_high_zeros)
```

```
character(0)
```

Select column to replace

[38]:
```r
# Select column to replace
#BCO
```

Operation

[39]:
```r
# Replace zero missing values by nan
# Replace columns with a high proportion of zeros with NaN
data[, columns_with_high_zeros] <- NA
```

Other Missing Values Select column to replace

[40]:
```r
# Select column to replace and missing value
```

Operation

[41]:
```r
# Replace other missing values by nan
```

Null/NaN Missing Values

[42]:
```r
# Intuitivamente: miramos nº datos en todas las columnas
# los null no los cuenta --> debe hacer el mismo nº por columna
```

[43]:
```r
# Podemos mirar directamente info donde viene
```

[44]:
```r
# Contamos los nulos de forma explícita
```

[45]:
```r
# summarize the number of rows with missing values for each column
```

0.6.2 Delete Missing Values

Deleting Rows with Missing Values in Target Column

[46]:
```r
data <- data[!is.na(data$Target), ]
```

Deleting Rows with Missing Values Only in case of high data size

[47]:
```r
# Eliminamos las filas con valores nulos
```

Deleting Features with some Missing Values Only with many features and for non-relevant features

[48]:
```r
# Selecciono las columnas con algún valor missing:
# Delete features with missing values
clean_data <- data %>%
  select(where(~ !any(is.na(.))))
```

Deleting Features using Rate Missing Values

[49]:
```r
# Number of data
# Define the minimum number of non-missing values
n <- 10  # For example, at least 10 non-missing values

# Count the number of non-missing values for each feature
non_missing_counts <- colSums(!is.na(data))

# Get the names of features that have fewer than n non-missing values
features_to_delete <- names(non_missing_counts[non_missing_counts < n])

# Remove the features with fewer than n non-missing values
data <- data[, !(names(data) %in% features_to_delete)]
```

[50]:
```r
# Number of missing data
n<- 10  # For example, at least 10 missing values
# Count the number of missing values for each feature
missing_counts <- colSums(is.na(data))

# Get the names of features that have missing values
features_to_delete <- names(missing_counts[missing_counts > n])

# Remove the features with missing values
data <- data[, !(names(data) %in% features_to_delete)]
```

[51]:
```r
# Rate (%) of missing data
# Define the rate threshold for missing values
rate_threshold <- 0.6  # For example, 60% missing values

# Calculate the percentage of missing values for each feature
missing_rate <- colMeans(is.na(data))

# Get the names of features that exceed the rate threshold
features_to_delete <- names(missing_rate[missing_rate > rate_threshold])
```

```r
# Remove the features with high missing value rates
data <- data[, !(names(data) %in% features_to_delete)]
```

0.6.3 Basic Imputation

Imputation by Previous Row Value

[52]:
```r
imputed_data
```

```
Error in eval(expr, envir, enclos): object 'imputed_data' not found
Traceback:
```

[ ]:
```r
# Sustituimos valores null por otro valor: VALOR FILA ANTERIOR

# Replace null values with the previous row's value
imputed_data <- data %>%
  mutate(across(.cols = everything(), .fns = ~ if_else(is.na(.), lag(.), .)))
```

Imputation by Next Row Value

[ ]:
```r
# Sustituimos valores null por otro valor: VALOR FILA SIGUIENTE
imputed_data <- data
for (col in names(imputed_data)) {
  missing_indices <- which(is.na(imputed_data[[col]]))
  for (i in missing_indices) {
    next_value <- imputed_data[[col]][i+1]
    imputed_data[[col]][i] <- next_value
  }
}
```

0.6.4 Statistical Imputation

A popular approach for data imputation is to calculate a statistical value for each column (such as a mean) and replace all missing values for that column with the statistic.

Selection of Imputation Strategy

[ ]:
```r
# The mean accuracy of each approach can then be compared.
#
# Specific results may vary given the stochastic nature of
# the learning algorithm, the evaluation procedure, or
# differences in numerical precision. Consider running the
# example a few times and compare the average performance.
#
```

[ ]:
```r
# Plot model performance for comparison
# box and whisker plot is created for each set of results,
# allowing the distribution of results to be compared.
```

Constant Imputation Select constant value

[ ]:
```r
# Constant imputation
# Replace missing values with constant values
str_impute<- "impute"
numeric_impute <- 0
imputed_data <- data %>%
  mutate_if(is.character, ~ifelse(is.na(.), str_impute, .)) %>%
  mutate_if(is.numeric, ~ifelse(is.na(.), numeric_impute, .))
```

Mean Imputation

[ ]:
```r
# Sustituimos valores null por otro valor: MEDIA
```

[ ]:
```r
# Sustituyo
# Impute missing values with the column means

imputed_data <- data %>%
  mutate(across(where(is.numeric), ~ ifelse(is.na(.), mean(., na.rm = TRUE), .
  ↪)))
```

Median Imputation

[ ]:
```r
# Sustituimos valores null por otro valor: MEDIA
# Miro la mediana
```

[ ]:
```r
# Sustituyo

imputed_data <- data %>%
  mutate(across(where(is.numeric), ~ ifelse(is.na(.), median(., na.rm = TRUE), .
  ↪)))
```

Most Frequent Imputation

[ ]:
```r
# Sustituyo
get_mode <- function(x) {
  freq_table <- table(x)
  mode <- as.numeric(names(freq_table)[freq_table == max(freq_table)])
  return(mode)
}

imputed_data <- data %>%
  mutate(across(everything(), ~ ifelse(is.na(.),get_mode(.), .)))
```

Interpolation Imputation

[ ]:
```r
# Sustituimos valores null por otro valor: INTERPOLANDO
# Métodos de interpolación
# 'linear', 'time', 'index', 'values', 'nearest', 'zero', 'slinear',
# 'quadratic', 'cubic', 'barycentric', 'krogh', 'polynomial', 'spline'
# 'piecewise_polynomial', 'pchip'
```

0.6.5 Prediction Imputation (KNN Imputation)

An approach to missing data imputation is to use a model to predict the missing values.

Evaluating k-hyperparmeter in KNN Imputation Select numbers of neighbors to evaluate

[ ]:
```
# Numbers of neighbors to evaluate
k = 5
```

Operation

[ ]:
```

```

Applying KNN Imputation

[ ]:
```
imputed_data <- data %>%
  mutate(across(where(is.numeric), ~ ifelse(is.na(.), impute_knn(., k = k), .)))
```

Select numbers of neighbors to evaluate

[ ]:
```
# Generating de new Data dataframe
```

0.6.6 Iterative Imputation

Evaluating Different Imputation Order We can experiment with different imputation order strategies, such as descending, right-to-left (Arabic), left-to-right (Roman), and random.

[ ]:
```
# compare iterative imputation strategies for the horse colic dataset
```

Applying Iterative Imputation Select strategie

[ ]:
```
# Selecting strategie
# strategies = ['ascending', 'descending', 'roman', 'arabic', 'random']
```

```
Error in impute_knn(): could not find function "impute_knn"
Traceback:
```

Operation

[ ]:
```
# Generating the new Data dataframe
```

0.7 Data Save

- Solo si se han hecho cambios
- No aplica

Identificamos los datos a guardar

[53]:
```
data_to_save <- data
```

Hospital	Especialidad	total_pacientes	ano
0	0	0	0
semana	CODCNH	id_area	nombre_area
0	0	0	0
cmunicipio	Municipio	CAMAS	Clase
0	0	0	0
Dependencia	TAC	RM	GAM
0	0	0	0
HEM	ASD	ALI	SPECT
0	0	0	0
MAMOS	DO	DIAL	X
0	0	0	0
Y	t3_1	t1_1	t2_1
0	0	0	0
t2_2	t4_1	t4_2	t4_3
0	0	0	0
t5_1	t6_1	t7_1	t8_1
0	0	0	0
t9_1	t10_1	t11_1	t12_1
0	0	0	0
capacidad	pacientes	consultas	hospitalizaciones
0	0	0	0
Target	is_train		
0	0		

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

0.7.1 Proceso 09.2

[54]:
```
caso <- "CU_25"
proceso <- '_09.2'
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/.

[55]:
```
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_09.2_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

[56]:
```
path_in <- paste0(iPath, file_save)
file.copy(path_out, path_in, overwrite = TRUE)
```

TRUE

0.8 REPORT

A continuación se realizará un informe de las acciones realizadas

0.9 Main Actions Carried Out

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

0.10 Main Conclusions

- Los datos están limpios para el despliegue

0.11 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

[ ]: