

## 09.3.- Data Cleansing-Outliers\_05\_servicios\_completo\_v\_01

June 16, 2023

#

CUxx\_Nombre del caso de uso

Citizenlab Data Science Methodology > II - Data Processing Domain \*\*\* > # 09.3.- Data Cleansing  
- Outliers

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

Basic operations	Text data analysis
	Delete Needless/Irrelevant/Private Columns Inconsistent Data. Expected values Zeroes Columns with a Single Value Columns with Very Few Values Columns with Low Variance Duplicates (rows/samples) & (columns/features) Data
Missing Values	Missing Values Identification
	Missing Values Per Sample Missing Values Per Feature Zero Missing Values Other Missing Values Null/NaN Missing Values Delete Missing Values Deleting Rows with Missing Values in Target Column Deleting Rows with Missing Values Deleting Features with some Missing Values Deleting Features using Rate Missing Values
	Basic Imputation
	Imputation by Previous Row Value Imputation by Next Row Value
	Statistical Imputation
	Selection of Imputation Strategy Constant Imputation Mean Imputation Median Imputation Most Frequent Imputation Interpolation Imputation
	Prediction Imputation (KNN Imputation )
	Evaluating k-hyperparameter in KNN Imputation Applying KNN Imputation
	Iterative Imputation
	Evaluating Different Imputation Order Applying Iterative Imputation
Outliers	Outliers - Univariate
	Visualizing Outliers Distribution Box Plots Isolation Forest Outliers Identification Grubbs' Test Z-Score Standard Deviation Method Interquartile Range Method Tukey's method Internally studentized residuals AKA z-score method Median Absolute Deviation method
	Outliers - MultiVariate
	Visualizing Outliers ScatterPlots Outliers Identification Mahalanobis Distance Robust Mahalanobis Distance DBSCAN Clustering PyOD Library
	Automatic Detection and Removal of Outliers
	Compare Algorithms LocalOutlierFactor IsolationForest Minimum Covariance Determinant

## 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: xxxxxxxxxxxx
- Output File: No aplica

### 0.3.1 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'
```

## 0.4 Settings

### 0.4.1 Libraries to use

```
[2]: library(readr)
library(dplyr)
library(sf)
library(tidyr)
library(ggplot2)
# 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

### 0.4.2 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. Uncomment 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_092_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\_092\_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)
```

	Fecha <date>	Servicio <chr>	CMUN <chr>	CDIS <chr>	CSEC <chr>	Futbol <dbl>	nservicios <dbl>	capacidad <dbl>	tmed <d
A spec_tbl_df: 5 × 19	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

## 0.6 Outliers - Univariate

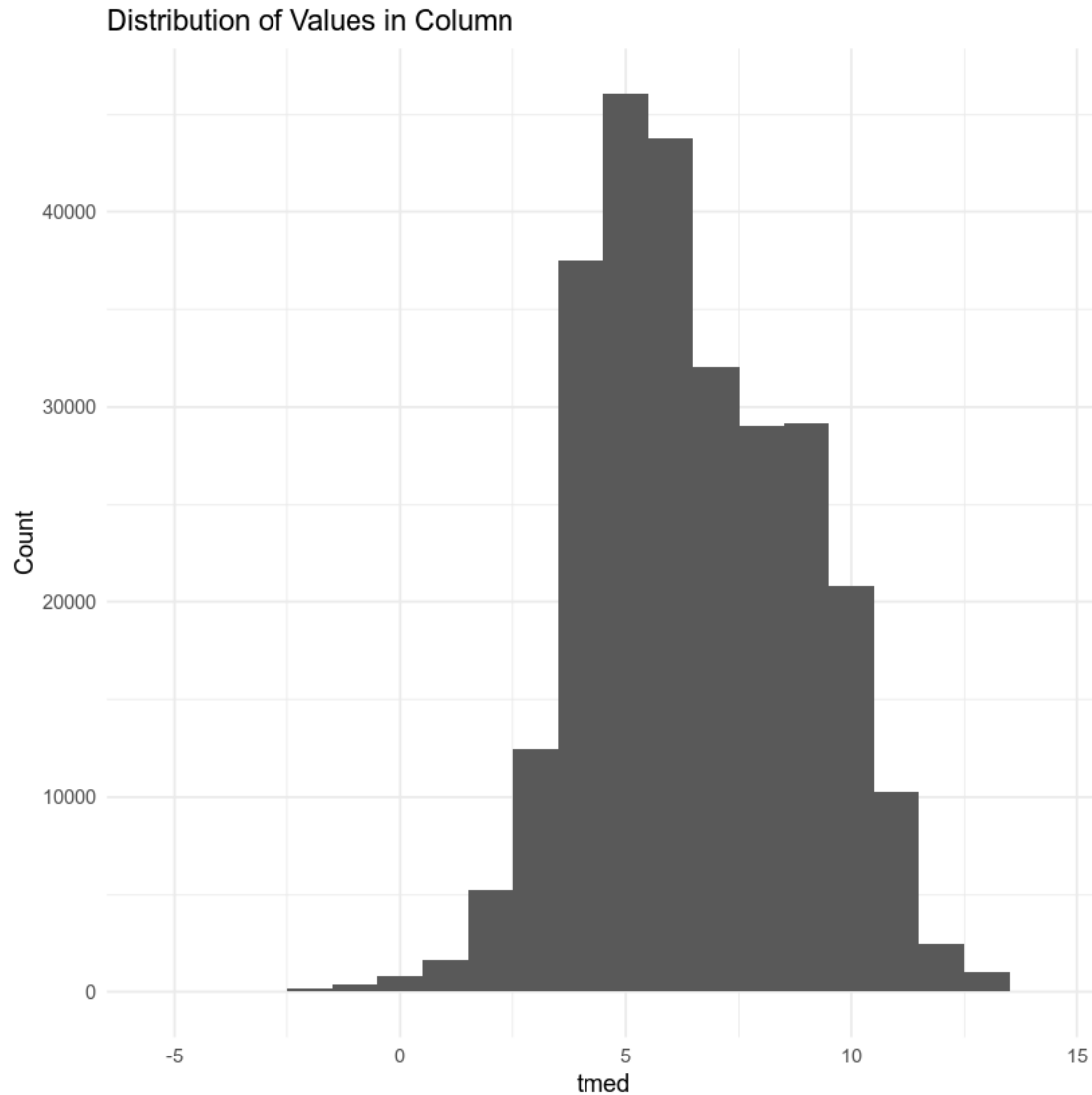
### 0.6.1 Visualizing Outliers

**Distribution** Selecting feature to analyze

```
[9]: # Selecting feature to analyze
column_name <- "tmed"
```

Operation

```
[10]: ggplot(data, aes(x = data[[column_name]])) +
  geom_histogram(binwidth = 1) +
  labs(x = column_name, y = "Count") +
  ggtitle("Distribution of Values in Column") +
  theme_minimal()
```



**Box Plots** Are great to summarize and visualize the distribution of variables easily and quickly.

[ ]:

Selecting feature to analyze

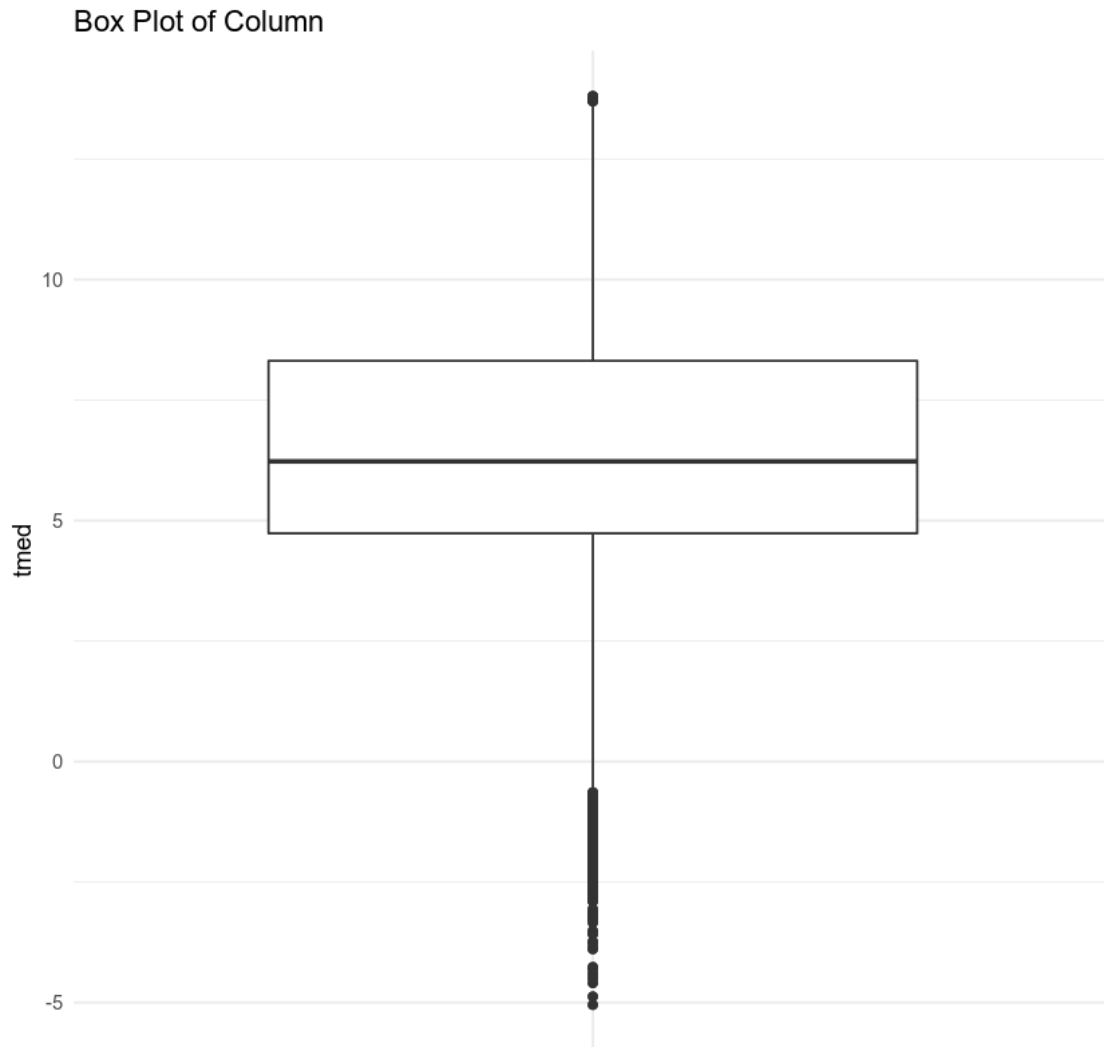
```
[11]: # Selecting feature to analyze  
      column_name <- "tmed"
```

Operation

```
[12]: # Analyze specifics features  
      # Create a box plot
```



```
ggplot(data, aes(x = "", y = data[[column_name]])) +
  geom_boxplot() +
  labs(x = "", y = column_name) +
  ggtitle("Box Plot of Column") +
  theme_minimal()
```



**Isolation Forest**    Selecting feature to analyze

```
[13]: # Selecting feature to analyze
```

Operation

```
[ ]:
```

## 0.6.2 Outliers Identification

**Grubbs' Test** Selecting feature to analyze

```
[14]: # Selecting feature to analyze
      column_name <- "tmed"
```

Operation

```
[15]: library(outliers)
      outliers <- grubbs.test(data[[column_name]], opposite = FALSE)
      print(outliers)
```

Grubbs test for one outlier

```
data: data[[column_name]]
G = 4.86396, U = 0.99991, p-value = 0.1569
alternative hypothesis: lowest value -5.04694059394377 is an outlier
```

**Z-Score** Selecting feature to analyze

```
[16]: # Selecting feature to analyze
      column_name <- "tmed"
```

Operation

```
[17]: # Define a threshold to identify an outlier.
      # List of row numbers with outlier
      # Choose the numeric column from your data

      # Calculate the z-score
      z_scores <- data %>% select(all_of(column_name)) %>% scale()

      # Define a threshold for identifying outliers (e.g., z-score > 5 or z-score < -5)
      threshold <- 5

      # Find the row numbers with z-scores exceeding the threshold
      outlier_rows <- which(abs(z_scores) > threshold)

      # Print the row numbers with outliers
      print(outlier_rows)

      data_cleaned <- subset(data, !(row.names(data) %in% outlier_rows))
```

integer(0)

Selecting feature to analyze

```
[18]: # Selecting all features to analyze
```

Operation

```
[ ]:
```

```
[19]: # Define a threshold to identify an outlier.  
# The first row contains the list of row numbers  
# and second array respective column numbers
```

**Standard Deviation Method**    Selecting num of std

```
[20]: # Selecting num of std
```

Operation

```
[21]: # identify outliers with standard deviation
```

**Interquartile Range Method**    Selecting factor k

```
[22]: # Selecting factor k
```

Operation

```
[23]: # identify outliers with standard deviation
```

**Tukey's method**

```
[24]: #Tukey's method
```

**Internally studentized residuals AKA z-score method**

```
[25]: #Internally studentized method (z-score)
```

**Median Absolute Deviation method**

```
[26]: #MAD method
```

## 0.7 Outliers - MultiVariate

### 0.7.1 Visualizing Outliers

ScatterPlots: a common way to plot multivariate outliers is the scatter plot.

**ScatterPlots**    A common way to plot multivariate outliers is the scatter plot.

```
[ ]:
```

Selecting feature to analyze

```
[27]: # Selecting feature to analyze
```

Operation

```
[ ]:
```

### 0.7.2 Outliers Identification

#### Mahalanobis Distance

```
[ ]:
```

Selecting feature to analyze

```
[28]: # Selecting features to analyze
```

Operation

```
[29]: # Analyze selected features
```

```
[30]: # Analyze all dataset
```

#### Robust Mahalanobis Distance

```
[31]: #Robust Mahalonibis Distance
```

Selecting feature to analyze

```
[32]: # Selecting features to analyze
```

Operation

```
[33]: # Analyze selected features
```

```
[34]: # Analyze all dataset
```

#### DBSCAN Clustering Selecting feature to analyze

```
[35]: # Selecting feature to analyze
```

Operation

```
[ ]:
```

```
[36]: # specify & fit model
```

```
[37]: # visualize outputs
```

```
[38]: # outliers dataframe
```

```
[39]: # Index of rows with outliers
```

```
[40]: # Outliers Dataframe
```

```
[ ]:
```

## 0.8 Data Save

- Solo si se han hecho cambios
- No aplica

Identificamos los datos a guardar

```
[41]: 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

### 0.8.1 Proceso 09.2

```
[42]: caso <- "CU_34"  
proceso <- '_093'  
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

```
[43]: # 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/.

```
[44]: 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\_093\_05\_servicios\_completo.csv'

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

```
[45]: 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

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

### 0.11 Main Conclusions

- Los datos están limpios para el despliegue

### 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

```
[ ]:
```