Documento de cambios

Ángela Alarcón Ballester, Lucía Ponce Salmerón y Carlos Sánchez Polo

2023-11-10

Introducción

En este proyecto analizaremos y trataremos los datos obtenidos de los registros de la calidad del aire por horas en zonas de Valencia. Vamos a realizar una exploración inicial de los datos y responderemos las preguntas que se deriven de ellos.

Importación y acondicionamiento de los datos

Cargamos la librerías que necesitaremos a lo largo del proyecto:

```
library(readr) # Entrada de datos
library(dplyr) # Manipulación de datos
```

```
##
## 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
```

```
library(ggplot2) # Para las gráficas
```

Cargamos y visualizamos los datos:

```
rvvcca_d_horarios_2016_2020 <- read_delim("ProyectoAED2023_files/data/rvvcca_d_horarios_2016-2020.csv",
  delim = ";", escape_double = FALSE,
  #Las columnas Fecha y Fecha creación las definimos como fechas en formato "YYYY-MM-DD".
  col_types = cols(Fecha = col_date(format = "%Y-%m-%d"),
  `Fecha creacion` = col_date(format = "%Y-%m-%d"), ), trim_ws = TRUE)

head(rvvcca_d_horarios_2016_2020)
```

```
## # A tibble: 6 x 30
##      Id Fecha      `Dia de la semana` `Dia del mes` Hora                Estacion
##   <dbl> <date>     <chr>                      <dbl> <dttm>              <chr>
## 1 50008 2016-10-24 Lunes                         24 0001-01-01 15:00:00 Politec~
## 2 50009 2016-10-24 Lunes                         24 0001-01-01 16:00:00 Politec~
## 3 50011 2016-10-24 Lunes                         24 0001-01-01 18:00:00 Politec~
## 4 50013 2016-10-24 Lunes                         24 0001-01-01 20:00:00 Politec~
## 5 50014 2016-10-24 Lunes                         24 0001-01-01 21:00:00 Politec~
## 6 50015 2016-10-24 Lunes                         24 0001-01-01 22:00:00 Politec~
## # i 24 more variables: PM1 <dbl>, PM2.5 <dbl>, PM10 <dbl>, NO <dbl>, NO2 <dbl>,
## #   NOx <dbl>, O3 <dbl>, SO2 <dbl>, CO <dbl>, `Velocidad del viento` <dbl>,
## #   `Direccion del viento` <dbl>, NH3 <dbl>, C7H8 <dbl>, C6H6 <dbl>,
## #   Ruido <dbl>, C8H10 <dbl>, Temperatura <dbl>, `Humedad relativa` <dbl>,
## #   Presion <dbl>, Radiacion <dbl>, Precipitacion <dbl>,
## #   `Velocidad maxima del viento` <dbl>, `Fecha creacion` <date>,
## #   `Fecha baja` <lgl>
```

Nos damos cuenta de que la columna `Hora` está en un formato complejo de manejar:

```
unique(rvvcca_d_horarios_2016_2020$Hora)
```

```
##  [1] "0001-01-01 15:00:00 UTC" "0001-01-01 16:00:00 UTC"
##  [3] "0001-01-01 18:00:00 UTC" "0001-01-01 20:00:00 UTC"
##  [5] "0001-01-01 21:00:00 UTC" "0001-01-01 22:00:00 UTC"
##  [7] "0001-01-01 01:00:00 UTC" "0001-01-01 02:00:00 UTC"
##  [9] "0001-01-01 04:00:00 UTC" "0001-01-01 12:00:00 UTC"
## [11] "0001-01-01 14:00:00 UTC" "0001-01-01 19:00:00 UTC"
## [13] "0001-01-01 00:00:00 UTC" "0001-01-01 05:00:00 UTC"
## [15] "0001-01-01 06:00:00 UTC" "0001-01-01 07:00:00 UTC"
## [17] "0001-01-01 10:00:00 UTC" "0001-01-01 13:00:00 UTC"
## [19] "0001-01-01 11:00:00 UTC" "0001-01-01 17:00:00 UTC"
## [21] "0001-01-01 23:00:00 UTC" "0001-01-01 03:00:00 UTC"
## [23] "0001-01-01 08:00:00 UTC" "0001-01-01 09:00:00 UTC"
```

Cambiamos el formato de la columna `Hora` a uno más manejable:

```
rvvcca_d_horarios_2016_2020$Hora <- format(rvvcca_d_horarios_2016_2020$Hora, format="%H:%M:%S")
unique(rvvcca_d_horarios_2016_2020$Hora)
```

```
##  [1] "15:00:00" "16:00:00" "18:00:00" "20:00:00" "21:00:00" "22:00:00"
##  [7] "01:00:00" "02:00:00" "04:00:00" "12:00:00" "14:00:00" "19:00:00"
## [13] "00:00:00" "05:00:00" "06:00:00" "07:00:00" "10:00:00" "13:00:00"
## [19] "11:00:00" "17:00:00" "23:00:00" "03:00:00" "08:00:00" "09:00:00"
```

Ordenamos por fecha:

```
indices_orden <- order(rvvcca_d_horarios_2016_2020$Fecha)
rvvcca_d_horarios_2016_2020 <- rvvcca_d_horarios_2016_2020[indices_orden, ]
```

Comprobamos que la columna `Fecha baja` está vacía, por tanto, no nos aporta ninguna información y la eliminamos:

```r
unique(rvvcca_d_horarios_2016_2020$`Fecha baja`)
```

```
## [1] NA
```

```r
rvvcca_d_horarios_2016_2020$`Fecha baja` <- NULL
```

Por último, visualizamos el estado actual de nuestros datos:

```r
head(rvvcca_d_horarios_2016_2020)
```

```
## # A tibble: 6 x 29
##      Id Fecha      `Dia de la semana` `Dia del mes` Hora    Estacion   PM1 PM2.5
##   <dbl> <date>     <chr>                     <dbl> <chr>   <chr>    <dbl> <dbl>
## 1     3 2016-01-01 Viernes                       1 02:00:~ Avda. F~    NA    NA
## 2     4 2016-01-01 Viernes                       1 03:00:~ Avda. F~    NA    NA
## 3     6 2016-01-01 Viernes                       1 05:00:~ Avda. F~    NA    NA
## 4     7 2016-01-01 Viernes                       1 06:00:~ Avda. F~    NA    NA
## 5    11 2016-01-01 Viernes                       1 10:00:~ Avda. F~    NA    NA
## 6    13 2016-01-01 Viernes                       1 12:00:~ Avda. F~    NA    NA
## # i 21 more variables: PM10 <dbl>, NO <dbl>, NO2 <dbl>, NOx <dbl>, O3 <dbl>,
## #   SO2 <dbl>, CO <dbl>, `Velocidad del viento` <dbl>,
## #   `Direccion del viento` <dbl>, NH3 <dbl>, C7H8 <dbl>, C6H6 <dbl>,
## #   Ruido <dbl>, C8H10 <dbl>, Temperatura <dbl>, `Humedad relativa` <dbl>,
## #   Presion <dbl>, Radiacion <dbl>, Precipitacion <dbl>,
## #   `Velocidad maxima del viento` <dbl>, `Fecha creacion` <date>
```

```r
tail(rvvcca_d_horarios_2016_2020)
```

```
## # A tibble: 6 x 29
##       Id Fecha      `Dia de la semana` `Dia del mes` Hora   Estacion   PM1 PM2.5
##    <dbl> <date>     <chr>                     <dbl> <chr>  <chr>    <dbl> <dbl>
## 1 352653 2020-12-31 Jueves                       31 10:00~ Consell~    NA    NA
## 2 352656 2020-12-31 Jueves                       31 13:00~ Consell~    NA    NA
## 3 352658 2020-12-31 Jueves                       31 15:00~ Consell~    NA    NA
## 4 352659 2020-12-31 Jueves                       31 16:00~ Consell~    NA    NA
## 5 352662 2020-12-31 Jueves                       31 19:00~ Consell~    NA    NA
## 6 352663 2020-12-31 Jueves                       31 20:00~ Consell~    NA    NA
## # i 21 more variables: PM10 <dbl>, NO <dbl>, NO2 <dbl>, NOx <dbl>, O3 <dbl>,
## #   SO2 <dbl>, CO <dbl>, `Velocidad del viento` <dbl>,
## #   `Direccion del viento` <dbl>, NH3 <dbl>, C7H8 <dbl>, C6H6 <dbl>,
## #   Ruido <dbl>, C8H10 <dbl>, Temperatura <dbl>, `Humedad relativa` <dbl>,
## #   Presion <dbl>, Radiacion <dbl>, Precipitacion <dbl>,
## #   `Velocidad maxima del viento` <dbl>, `Fecha creacion` <date>
```

Valores faltantes

El estudio de los valores faltantes es esencial para garantizar que los análisis de datos sean sólidos y confiables. Como en la tabla previa hemos identificado valores faltantes en los datos, en este apartado procederemos a analizarlos. Primero realizamos un resumen de la cantidad de valores faltantes en cada columna:

```r
valores_faltantes <- sapply(rvvcca_d_horarios_2016_2020, function(x) sum(is.na(x)))
valores_faltantes
```

```
##                          Id                       Fecha
##                           0                           0
##            Dia de la semana                 Dia del mes
##                           0                           0
##                        Hora                    Estacion
##                           0                           0
##                         PM1                       PM2.5
##                      286441                      183359
##                        PM10                          NO
##                      183354                       88740
##                         NO2                         NOx
##                       71383                       88738
##                          O3                         SO2
##                       87741                       91245
##                          CO       Velocidad del viento
##                      236277                        7874
##       Direccion del viento                         NH3
##                        6020                      313088
##                        C7H8                        C6H6
##                      313156                      313635
##                       Ruido                       C8H10
##                      309508                      313053
##                 Temperatura           Humedad relativa
##                        9033                       16357
##                     Presion                   Radiacion
##                        6276                        6814
##               Precipitacion Velocidad maxima del viento
##                        7414                        8045
##             Fecha creacion
##                           0
```

Creamos un gráfico de barras para visualizar mejor la proporción:

```r
valores_faltantes_df <- data.frame(Columna = names(valores_faltantes), Valores_Faltantes = valores_falta

ggplot(valores_faltantes_df, aes(x = Columna, y = valores_faltantes)) +
  geom_bar(stat = "identity", fill = "lightblue") +
  labs(title = "Proporción de Valores Faltantes por Variable", x = "Variable", y = "Valores Faltantes")
  theme(axis.text.x = element_text(angle = 45, hjust = 1))  # Rota las etiquetas del eje x
```

Podemos ver como hay variables con un gran número de valores faltantes, en especial, los contaminantes atmosféricos (`NO`, `NO2`, `SO2`,...) y la información meteorológica (`Precipitacion`, `Radiacion`,...). Esto es debido a que cada estación dispone de unos sensores distintos de medición de estos fenómenos. Además, el ruido también dispone de muchos valores faltantes, ya que depende de lo tranquila que sea cada estación.

Por otra parte, vemos en la gráfica 1 que los contaminantes atmosféricos y el ruido son los que proporcionalmente tienen más valores faltantes.

Por tanto, para poder realizar un mejor análisis vamos a agrupar nuestro dataframe por `Estacion`:

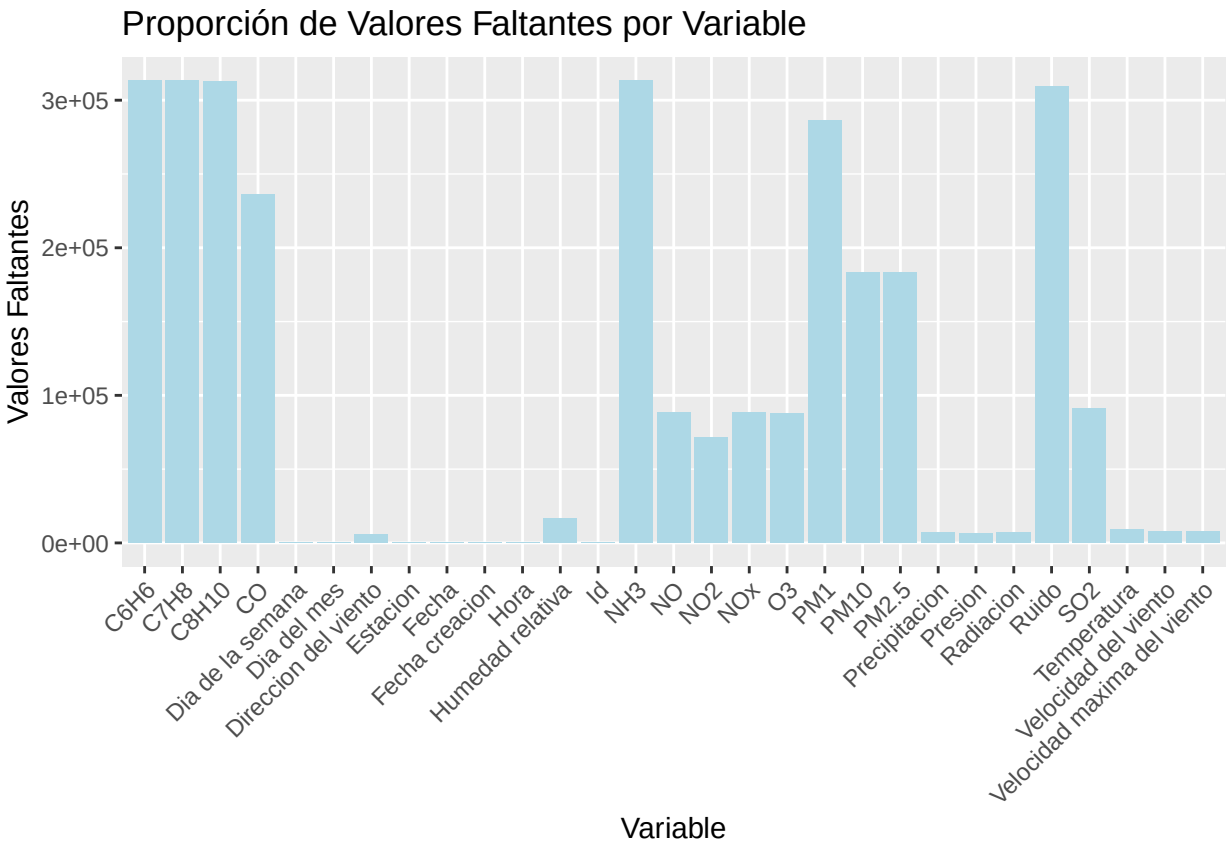

Figure 1: Proporciones de valores faltantes.

```r
grupos <- rvvcca_d_horarios_2016_2020 %>% group_by(Estacion)

lista_dataframes <- split(grupos, f = grupos$Estacion)

nombres_dataframes <- unique(rvvcca_d_horarios_2016_2020$Estacion)
lista_dataframes <- setNames(lista_dataframes, nombres_dataframes)
```

Limpiamos las columnas vacías de cada dataframe:

```r
for (estacion in nombres_dataframes) {
  lista_dataframes[[estacion]] <- lista_dataframes[[estacion]] %>%
    select_if(~!all(is.na(.)))
}
```

Convertimos la lista de dataframes en dataframes independientes:

```r
list2env(lista_dataframes, envir = .GlobalEnv)
```

```
## <environment: R_GlobalEnv>
```

Ahora, veamos que parámetros se han medido en cada estación:

```r
# Crear una lista con los nombres de los parámetros a observar
parametros <- names(`Puerto Valencia`)

# Crear un dataframe vacío con las zonas como filas y los parámetros como columnas
tabla_zonas <- data.frame(Zona = character(0), stringsAsFactors = FALSE)

# Iterar sobre los dataframes y extraer los nombres de las columnas
for (zona in 1:length(lista_dataframes)) {
  # Crear una fila con el nombre de la zona
  fila_zona <- data.frame(Zona = names(lista_dataframes)[zona], stringsAsFactors = FALSE)

  # Iterar sobre los parámetros y agregar TRUE o FALSE según si existen en la zona
  for (parametro in parametros) {
    fila_zona[[parametro]] <- parametro %in% colnames(lista_dataframes[[zona]])
  }

  # Unir la fila de la zona al dataframe
  tabla_zonas <- rbind(tabla_zonas, fila_zona)
}
```

Visualización

Conclusiones