Documento de cambios

Ángela Alarcón Ballester, Lucía Ponce Salmerón y Carlos Sánchez Polo

2023-11-12

Introducción

En este proyecto analizaremos y trataremos los datos por horas de calidad del aire de las estaciones de la red de vigilancia de la ciudad 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
library(ggplot2) # Para las gráficas
library(pheatmap)
```

```
## Warning: package 'pheatmap' was built under R version 4.3.2
```

```
library(purrr)
```

```
## Warning: package 'purrr' was built under R version 4.3.2
```

```
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 4.3.2
```

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>
```

Convertimos la columna `Estacion` en factor:

```
rvvcca_d_horarios_2016_2020$Estacion <- as.factor(rvvcca_d_horarios_2016_2020$Estacion)
```

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:

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

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

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

Utilizamos la función `glimpse` para obtener una visión general de los datos:

```
glimpse(rvvcca_d_horarios_2016_2020)
```

```
## Rows: 352,666
## Columns: 29
## $ Id                            <dbl> 3, 4, 6, 7, 11, 13, 14, 17, 19, 25, 31, ~
## $ Fecha                         <date> 2016-01-01, 2016-01-01, 2016-01-01, 201~
## $ ‘Dia de la semana‘            <chr> "Viernes", "Viernes", "Viernes", "Vierne~
## $ ‘Dia del mes‘                 <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
## $ Hora                          <chr> "02:00:00", "03:00:00", "05:00:00", "06:~
## $ Estacion                      <fct> Avda. Francia, Avda. Francia, Avda. Fran~
## $ PM1                           <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ PM2.5                         <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ PM10                          <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ NO                            <dbl> 2, 5, 40, 25, 3, 5, NA, 3, 5, 4, 9, 21, ~
## $ NO2                           <dbl> 48, 58, 53, 50, 24, 31, NA, 17, 51, 26, ~
## $ NOx                           <dbl> 52, 66, 114, 89, 27, 38, NA, 21, 58, 31,~
## $ O3                            <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, 24, ~
## $ SO2                           <dbl> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3~
## $ CO                            <dbl> 0.1, 0.1, 0.3, 0.3, 0.1, 0.1, 0.1, 0.1, ~
## $ ‘Velocidad del viento‘        <dbl> 0.5, 0.2, 0.6, 0.3, 0.2, 0.4, 1.1, 3.1, ~
## $ ‘Direccion del viento‘        <dbl> 328, 295, 300, 321, 297, 264, 187, 141, ~
## $ NH3                           <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, 9, 9~
## $ C7H8                          <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ C6H6                          <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ Ruido                         <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ C8H10                         <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ Temperatura                   <dbl> 12.9, 12.6, 12.4, 12.8, 13.4, 16.0, 16.1~
## $ ‘Humedad relativa‘            <dbl> 79, 81, 83, 79, 74, 65, 67, 76, 84, 79, ~
## $ Presion                       <dbl> 1012, 1012, 1012, 1011, 1013, 1011, 1010~
## $ Radiacion                     <dbl> 0, 0, 0, 0, 122, 346, 365, 152, 1, 0, 0,~
## $ Precipitacion                 <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ ‘Velocidad maxima del viento‘ <dbl> 1.2, 0.8, 1.6, 0.7, 0.5, 0.7, 2.6, 4.5, ~
## $ ‘Fecha creacion‘              <date> 2023-02-15, 2023-02-15, 2023-02-15, 202~
```

Observamos que contamos con las siguientes variables:

- Información general (Id, Fecha, Día de la semana, Día del mes, Hora, Fecha de creación). Son de tipo double, factor y fecha.

- Contaminantes atmosféricos (PM1, PM2.5, PM10, NO, NO2, NOx, O3, SO2, CO, NH3, C7H8, C6H6, C8H10) y Ruido. Todos son de tipo double.
- Información meteorológica (Velocidad del viento, Dirección del viento, Temperatura, Humedad relativa, Presión, Radiación, Precipitación, Velocidad máxima del viento). Todos son de tipo double.
- Estacion (Avda. Francia, Bulevard Sur, Molino del Sol, Pista Silla, Politécnico, Viveros, Centro, Consellería Meteo, Nazaret Meteo, Puerto València). Es de tipo factor.

Valores faltantes

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

```
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_faltantes)

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
```

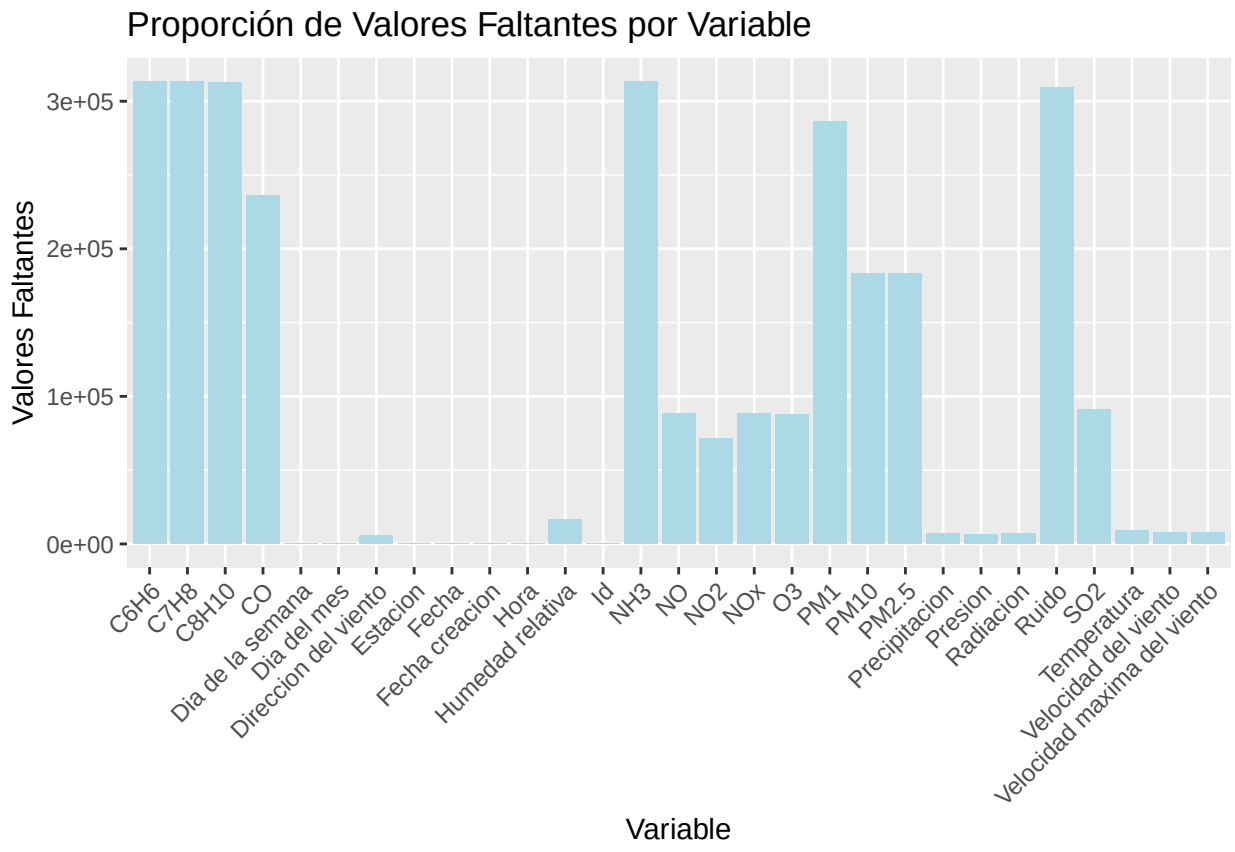


Figure 1: Proporciones de valores faltantes.

Podemos ver como hay variables con un gran número de valores faltantes, en especial, en los contaminantes atmosféricos y la información meteorológica. 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`:

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

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

```r
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)
```

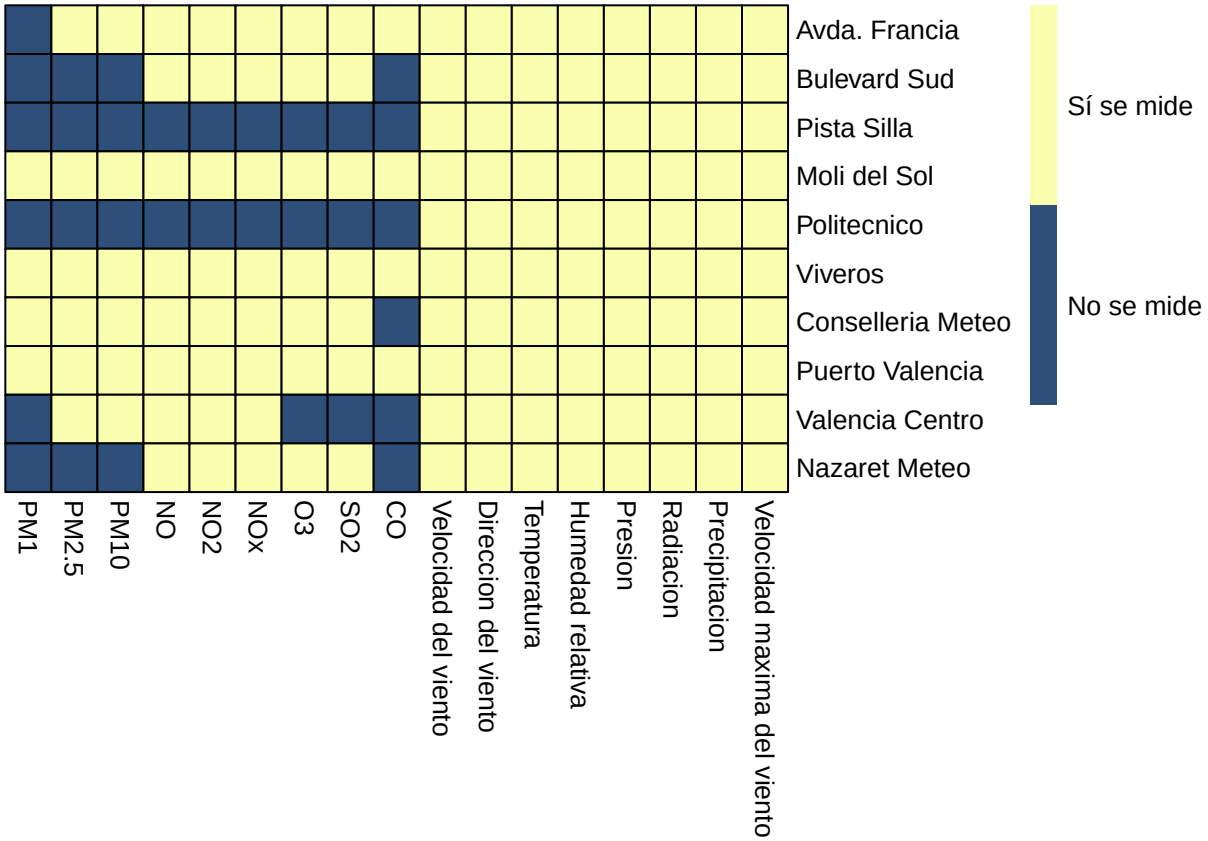
```
## <environment: R_GlobalEnv>
```

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)
}

rownames(tabla_zonas) <-tabla_zonas$Zona
tabla_zonas <- subset(tabla_zonas, select = -c(Zona,Id,Fecha,
        `Dia de la semana`,`Dia del mes`,Hora,Estacion,`Fecha creacion`))
tabla_numeric <- as.data.frame(sapply(tabla_zonas, as.numeric))
```

```r
require(pheatmap)
```

```r
# Crea el heatmap con colores azul y amarillo
pheatmap(tabla_numeric, color=hcl.colors(2,palette = "BluYl"),
         labels_row =rownames(tabla_zonas),cluster_rows = F,
         cluster_cols = F, legend_breaks = c(0,0.25,0.75, 1),
         legend_labels = c("","No se mide", "Sí se mide",""),
         border_color = "black")
```



Análisis de las variables (univariante / bivariante)

Análisis de outliers

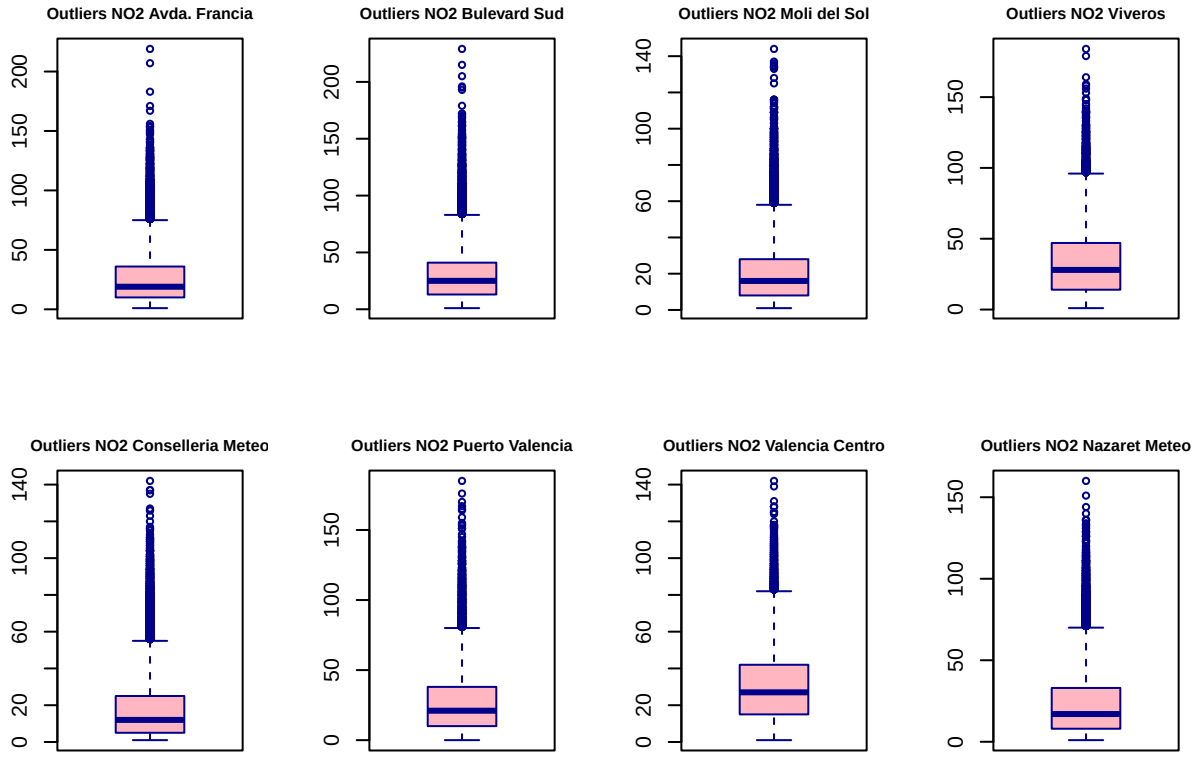
A continuación, exploraremos la calidad del aire mediante boxplots que representan las concentraciones de dos sustancias fundamentales: dióxido de nitrógeno (NO2) y ozono (O3). Cada gráfico proporciona una visión detallada de la distribución de estas sustancias en las distintas estaciones de la red de vigilancia atmosférica de la ciudad de València.

```
generate_boxplot <- function(data, title) {
  boxplot(data, main = title,col = "lightpink", border = "darkblue")
}

par(mfrow = c(2, 4), mar = c(4, 4, 2, 1),cex.main = 0.8)

for (nombre_df in nombres_dataframes) {
  df <- get(nombre_df)

  if ("NO2" %in% names(df) && length(df$NO2) > 0) {
    generate_boxplot(df$NO2, paste("Outliers NO2", nombre_df, sep = " "))
  } else {
    cat("No se encontró la columna 'NO2' en", nombre_df, "\n")
  }
```

```
}
```

```
## No se encontró la columna 'NO2' en Pista Silla
```

```
## No se encontró la columna 'NO2' en Politecnico
```



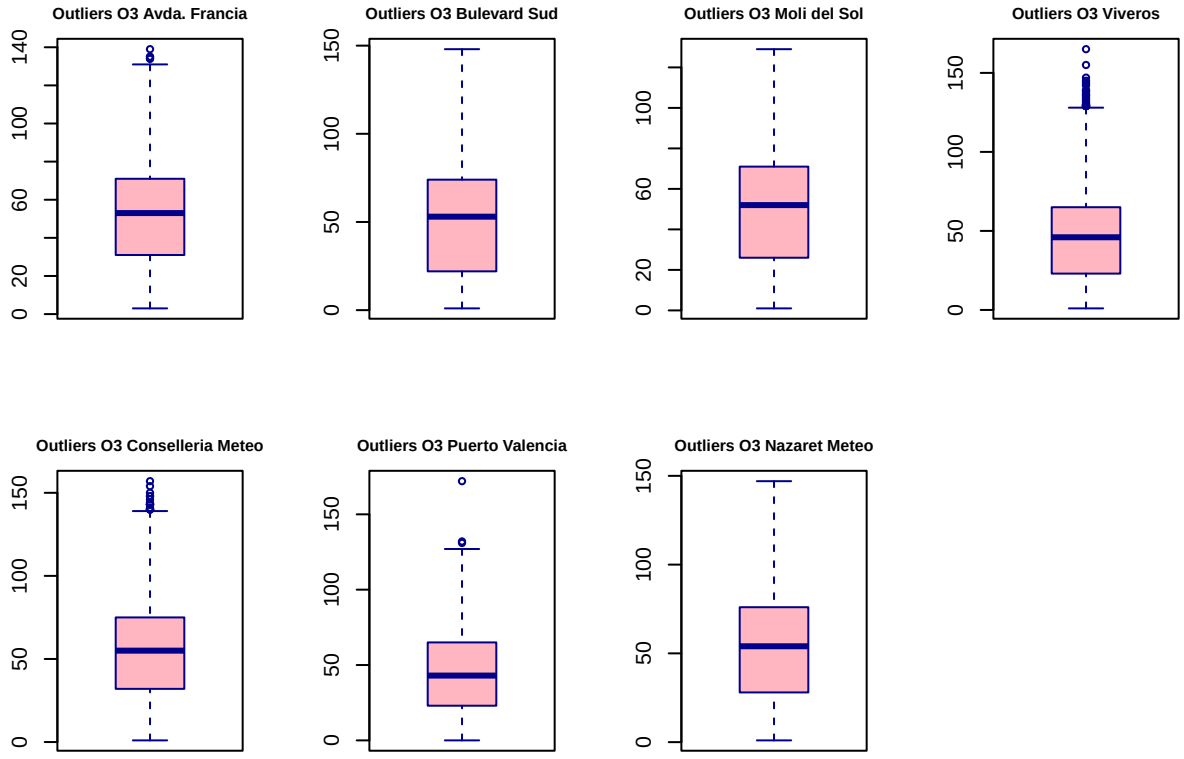
Observamos una notable presencia de outliers en la variable NO2 en todas las estaciones, destacando en la parte superior de los boxplots. Estos boxplots son estrechos, lo que sugiere que la mayor parte de los datos se concentra en un rango reducido de concentraciones de dióxido de nitrógeno. Sin embargo, la presencia de numerosos outliers en la parte superior indica que a menudo se generan concentraciones inusualmente altas de esta sustancia en el aire.

```
par(mfrow = c(2, 4), mar = c(4, 4, 2, 1),cex.main = 0.8)

for (nombre_df in nombres_dataframes) {
  df <- get(nombre_df)

  if ("O3" %in% names(df) && length(df$O3) > 0) {
    generate_boxplot(df$O3, paste("Outliers O3", nombre_df, sep = " "))
  } else {
    cat("No se encontró la columna 'O3' en", nombre_df, "\n")
  }
}
```

```
## No se encontró la columna 'O3' en Pista Silla
```

```
## No se encontró la columna 'O3' en Politecnico
```

```
## No se encontró la columna 'O3' en Valencia Centro
```



En contraste, la presencia de outliers en la variable de ozono es menos pronunciada. Aunque en las estaciones de Viveros y Consellería Meteo se observa una mayor proporción de outliers, en general, los valores presentan una distribución más uniforme. Los boxplots son más anchos, indicando una dispersión más amplia de las concentraciones de ozono.

También hemos considerado oportuno implementar funciones específicas para la identificación de outliers utilizando distintos métodos estadísticos, como la regla de 3 sigma, el identificador Hampel y la regla de percentiles. Este es otro modo de resaltar valores que podrían considerarse atípicos y podemos comparar con los resultados obtenidos mediante el método boxplot.

Función para la regla de 3 sigma:

```
reglasigma <- function(x) {
  media <- mean(x, na.rm = TRUE)
  desviacion <- sd(x, na.rm = TRUE)
  umbral_superior <- media + 3 * desviacion
  umbral_inferior <- media - 3 * desviacion
  outliers <- x[x > umbral_superior | x < umbral_inferior]
  return(outliers)
}
```

Función para el identificador Hampel:

```r
reglahampel <- function(x) {
  mediana <- median(x, na.rm = TRUE)
  mad <- mad(x, na.rm = TRUE)
  umbral_superior <- mediana + 3 * mad
  umbral_inferior <- mediana - 3 * mad
  outliers <- x[x > umbral_superior | x < umbral_inferior]
  return(outliers)
}
```

Función para la regla del boxplot:

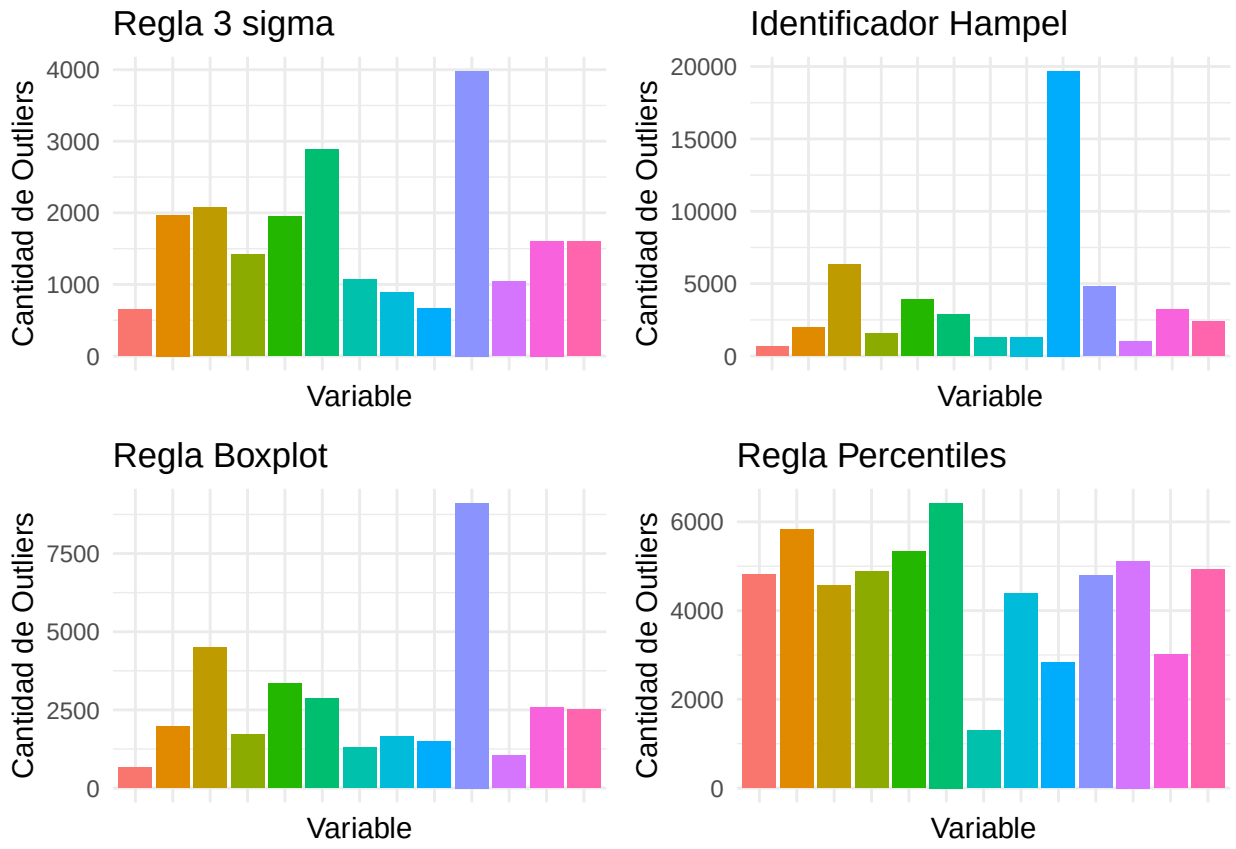
```r
reglaboxplot <- function(x) {
  cuartil_75 <- quantile(x, 0.75, na.rm = TRUE)
  cuartil_25 <- quantile(x, 0.25, na.rm = TRUE)
  iqr <- cuartil_75 - cuartil_25
  umbral_superior <- cuartil_75 + 1.5 * iqr
  umbral_inferior <- cuartil_25 - 1.5 * iqr
  outliers <- x[x > umbral_superior | x < umbral_inferior]
  return(outliers)
}
```

Función para la regla de percentiles:

```r
reglapercentil <- function(x) {
  percentil_5 <- quantile(x, 0.05, na.rm = TRUE)
  percentil_95 <- quantile(x, 0.95, na.rm = TRUE)
  outliers <- x[x < percentil_5 | x > percentil_95]
  return(outliers)
}
```

```r
detectar_outliers <- function(x) {
  sol <- list(
    r_sigma = data.frame(Variable = colnames(x), Metodo = "Regla 3 sigma",
              Outliers = sapply(x, function(col) length(reglasigma(col)))),
    r_hampel = data.frame(Variable = colnames(x), Metodo = "Identificador Hampel",
               Outliers = sapply(x, function(col) length(reglahampel(col)))),
    r_boxplot = data.frame(Variable = colnames(x), Metodo = "Regla Boxplot",
                Outliers = sapply(x, function(col) length(reglaboxplot(col)))),
    r_percentiles = data.frame(Variable = colnames(x), Metodo = "Regla Percentiles",
                    Outliers = sapply(x, function(col) length(reglapercentil(col))))
  )

  return(sol)
}


columnas_a_excluir <- c(1:6, ncol(df))
col <- setdiff(names(df), names(df)[columnas_a_excluir])
outliers_resultados <- detectar_outliers(Viveros[col])

plots <- lapply(outliers_resultados, function(result) {
  ggplot(result, aes(x = Variable, y = Outliers, fill = Variable)) +
    geom_bar(stat = "identity") +
```

```
    labs(title = unique(result$Metodo),
         x = "Variable",
         y = "Cantidad de Outliers") +
    theme_minimal() + theme(axis.text.x = element_blank(), legend.position = "none")
})

grid.arrange(grobs = plots, ncol = 2)
```



La información resultante se presenta de manera clara y comparativa a través de gráficos de barras. Cada barra en estos gráficos representa la cantidad de outliers detectados para una variable específica, empleando diferentes métodos de detección. En este caso, lo hemos aplicado sobre los datos de la estación Viveros.

Conclusiones