Air Quality Data Analysis: A comprehensive examination

Nicole Lastra Quiroz

Packages and libraries

We proceed with the installation of packages and libraries necessary for this work, using the following command: `install.packages(c("mice", "factoextra", "gridExtra"))`,considering that `mice` will be used for handling data imputation, `factoextra` for factor analysis, and `gridExtra` for graphical layout.

Then, the following libraries are loaded:

```r
library("mice")
library("factoextra")
library("gridExtra")
library("tidyverse")
library("visdat")
library("dlookr")
library("e1071")
library("flextable")
library("inspectdf")
library("qqplotr")
library("ggpmisc")
library("fdth")
library("PerformanceAnalytics")
library("corrplot")
```

Data loading

We begin the data analysis by loading the CSV file, which is stored in a dataframe named `air`, using the `read.csv()` function. With this, we can observe the basic structure of the dataframe, including the column names and a few sample data rows.

```r
air <- read.csv("/Users/nicolelastraquiroz/Documents/R studio/airq_dt.csv", header = T,
                     sep    = ",",
                     dec    = "." )
head(air)
```

```
##   Ozono RadSol Vient Temp Mes Dia
## 1    21    259  15.5   76   9  12
## 2    32     92  12.0   61   5  24
## 3    20     37   9.2   65   6  18
## 4    NA    139   8.6   82   7  11
## 5    28    273  11.5   82   8  13
## 6    NA    266  14.9   58   5  26
```

Dataframe structure analysis

```r
dim(air)
```

```
## [1] 100   6
```

```r
str(air)
```

```
## 'data.frame':    100 obs. of  6 variables:
##  $ Ozono : int  21 32 20 NA 28 NA NA 14 89 19 ...
##  $ RadSol: int  259 92 37 139 273 266 98 334 229 99 ...
##  $ Vient : num  15.5 12 9.2 8.6 11.5 14.9 11.5 11.5 10.3 13.8 ...
##  $ Temp  : int  76 61 65 82 82 58 80 64 90 59 ...
##  $ Mes   : int  9 5 6 7 8 5 6 5 8 5 ...
##  $ Dia   : int  12 24 18 11 13 26 28 16 8 8 ...
```

In general, the `air` dataframe contains 100 observations (rows) and 6 numeric variables (colums). This variables (5 integers and 1 numeric) are related to air quality measurements:

- `Ozono` variable represents the **concentration of ozone (O_3)** in the atmosphere, measured in parts per billion (ppb), which is crucial for assessing air quality and its impact on health and the environment.
- `RadSol` indicates the **solar radiation received**, typically measured in watts per square meter (W/m^2), which plays a significant role in determining atmospheric conditions.
- `Vient` variable refers to **wind speed**, measured in kilometers per hour (km/h), reflecting how air movement affects pollutant dispersion.
- `Temp` denotes the **ambient temperature in degrees Celsius (°C)**, which can influence chemical reactions in the atmosphere and, consequently, air quality.
- `Mes` and `Dia` represent the **month** and **day**, respectively, providing temporal context for the observations recorded in the dataset.

Missing data study

It is important to note that the presence of NA (missing values) can interfere with subsequent analyses, so they must be considered when performing primary studies and observations of the dataframe. Hence, it is essential to generate appropriate imputations for the dataset being worked on.

In this case, although missing values (NA) were already visible in the earlier review of the dataframe, now we use `summary()` to get a quick count and check if there are null values in the dataset (initially considering that all 6 variables are numeric, as seen in the earlier review).

```r
summary(air)
```

```
##      Ozono            RadSol           Vient            Temp      
##  Min.   :  1.00   Min.   :  7.00   Min.   : 2.30   Min.   :56.00  
##  1st Qu.: 16.00   1st Qu.: 98.75   1st Qu.: 7.40   1st Qu.:72.00  
##  Median : 29.00   Median :209.50   Median :10.30   Median :78.00  
##  Mean   : 41.07   Mean   :185.48   Mean   :10.21   Mean   :77.52  
##  3rd Qu.: 51.50   3rd Qu.:259.00   3rd Qu.:12.15   3rd Qu.:84.25
##  Max.   :168.00   Max.   :334.00   Max.   :20.70   Max.   :96.00
##  NA's   :18       NA's   :4
##       Mes             Dia
##  Min.   :5.00   Min.   : 1.00
##  1st Qu.:6.00   1st Qu.: 9.00
##  Median :7.00   Median :15.00
##  Mean   :7.06   Mean   :15.39
##  3rd Qu.:8.00   3rd Qu.:22.00
##  Max.   :9.00   Max.   :31.00
##
```

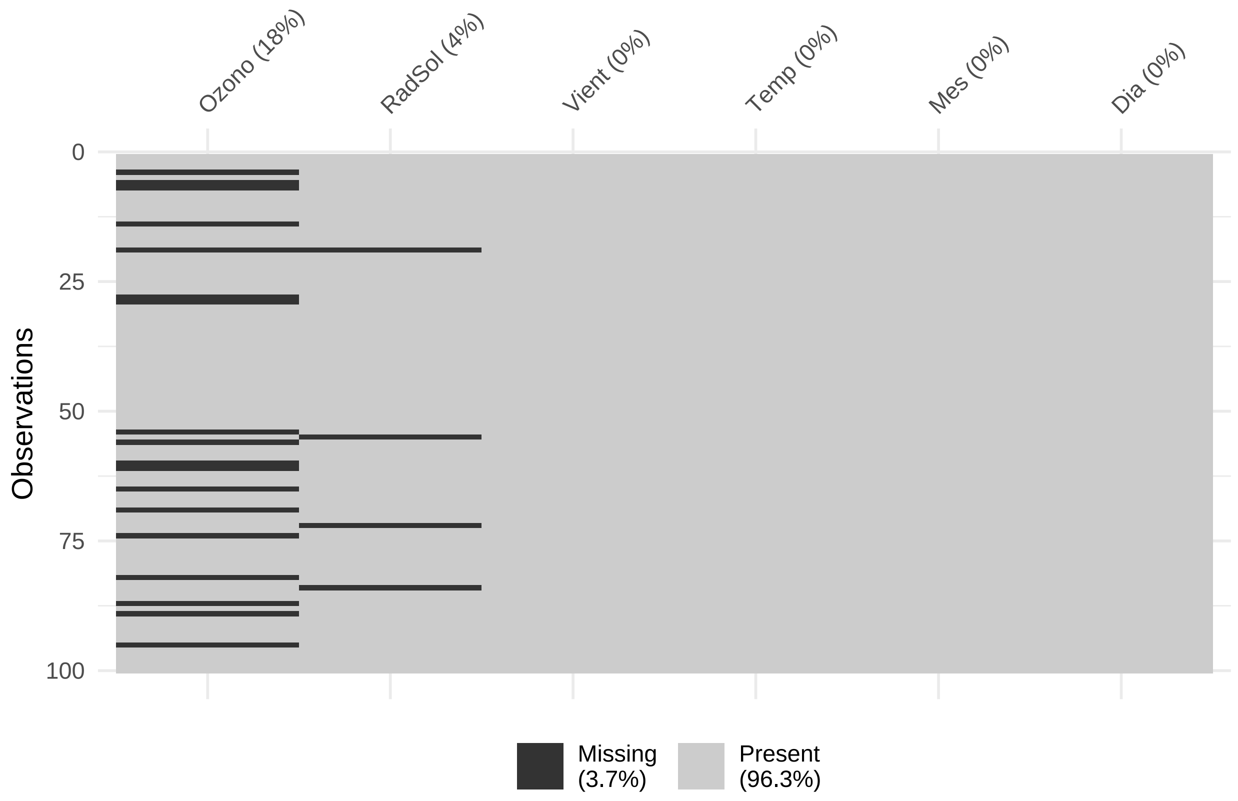
As a complement to this, and considering other methods, we add a detailed review using column-wise NA counts with `colSums(is.na())`.

```r
colSums(is.na(air))
```

```
##  Ozono RadSol  Vient   Temp    Mes    Dia
##     18      4      0      0      0      0
```

Also, a graphical visualization of the missing values using `visdat::vis_miss()`.

```r
visdat::vis_miss(air)
```



From the three methods, it is confirmed that there are columns with NA values, specifically: `Ozono` and `RadSol`, with 18 and 4 missing values, respectively, accounting for a total of 3.7% missing data versus 96.3% valid data.

Imputation of missing values

The missing values could be imputed using the mean of each variable, as this is a simple and effective strategy when the percentage of missing data is low, as in this case. However, we will use the `mice` package (*Multivariable Imputation by Chained Equations*), which offers a reliable and robust method for most cases of data imputation.

```
imp_dt <- mice(air, m = 1, seed = 100)
```

```
##
##  iter imp variable
##   1   1  Ozono  RadSol
##   2   1  Ozono  RadSol
##   3   1  Ozono  RadSol
##   4   1  Ozono  RadSol
##   5   1  Ozono  RadSol
```

```
air_impt <- complete(imp_dt, 1)
```

The imputation result was stored in a new dataframe called `air_impt`.

Verification of missing values

To rule out any potential human error in the imputation process (code errors, incorrectly saved imputed data, etc.), we will check again for NA values.

```
colSums(is.na(air_impt))
```

```
##  Ozono RadSol  Vient   Temp    Mes    Dia
##      0      0      0      0      0      0
```

```
str(air_impt)
```

```
## 'data.frame':    100 obs. of  6 variables:
##  $ Ozono : int  21 32 20 16 28 19 28 14 89 19 ...
##  $ RadSol: int  259 92 37 139 273 266 98 334 229 99 ...
##  $ Vient : num  15.5 12 9.2 8.6 11.5 14.9 11.5 11.5 10.3 13.8 ...
##  $ Temp  : int  76 61 65 82 82 58 80 64 90 59 ...
##  $ Mes   : int  9 5 6 7 8 5 6 5 8 5 ...
##  $ Dia   : int  12 24 18 11 13 26 28 16 8 8 ...
```

In summary, it is confirmed that there are no missing values after the imputation process, so we can proceed with the next steps.

Full diagnosis of categorical variables

Although the dataset appears to contain mainly numeric variables, `Mes`(month) and `Dia` (day) can be treated as categorical variables due to their context. Therefore, in this case, we will first transform them into categories and then analyze them accordingly.

Transformation of month and day variables into categories

```
air_impt$Mes<-factor(air_impt$Mes)
air_impt$Dia<-factor(air_impt$Dia)
```

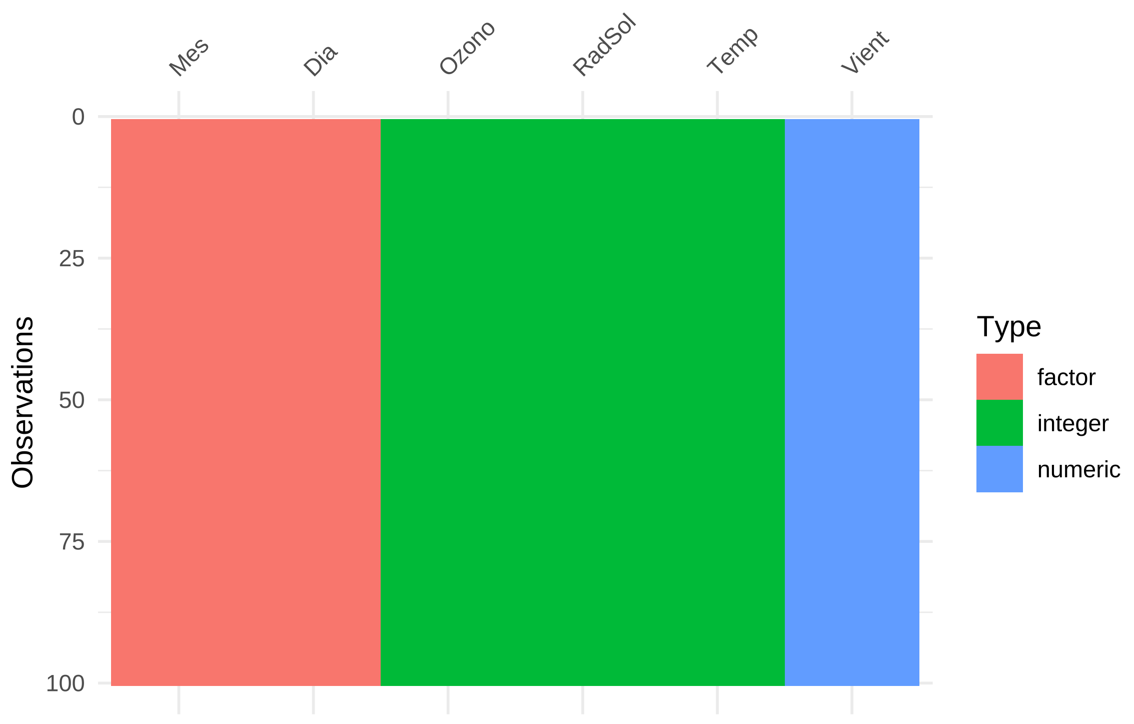
Review of the transformation

```
str(air_impt)
```

```
## 'data.frame':    100 obs. of  6 variables:
##  $ Ozono : int  21 32 20 16 28 19 28 14 89 19 ...
##  $ RadSol: int  259 92 37 139 273 266 98 334 229 99 ...
##  $ Vient : num  15.5 12 9.2 8.6 11.5 14.9 11.5 11.5 10.3 13.8 ...
##  $ Temp  : int  76 61 65 82 82 58 80 64 90 59 ...
##  $ Mes   : Factor w/ 5 levels "5","6","7","8",..: 5 1 2 3 4 1 2 1 4 1 ...
##  $ Dia   : Factor w/ 31 levels "1","2","3","4",..: 12 24 18 11 13 26 28 16 8 8 ...
```

Graphical visualization of data types

```
visdat::vis_dat(air_impt,sort_type = TRUE)
```



Categorical variables analysis

```
ft1 <- flextable(diagnose_category(air_impt))
ft1 <- set_caption(ft1, caption = "Details of categorical variables in the dataset")
ft1
```

Table 1: Details of categorical variables in the dataset

variables	levels	N	freq	ratio	rank
Mes	8	100	24	24	1
Mes	5	100	22	22	2
Mes	9	100	21	21	3
Mes	7	100	17	17	4
Mes	6	100	16	16	5
Dia	2	100	5	5	1
Dia	9	100	5	5	1
Dia	14	100	5	5	1
Dia	29	100	5	5	1
Dia	4	100	4	4	5
Dia	7	100	4	4	5
Dia	10	100	4	4	5
Dia	12	100	4	4	5
Dia	15	100	4	4	5
Dia	18	100	4	4	5

Normality study of quantitative variables

```
numeric_vars <- select_if(air_impt, is.numeric)
summary(numeric_vars)
```

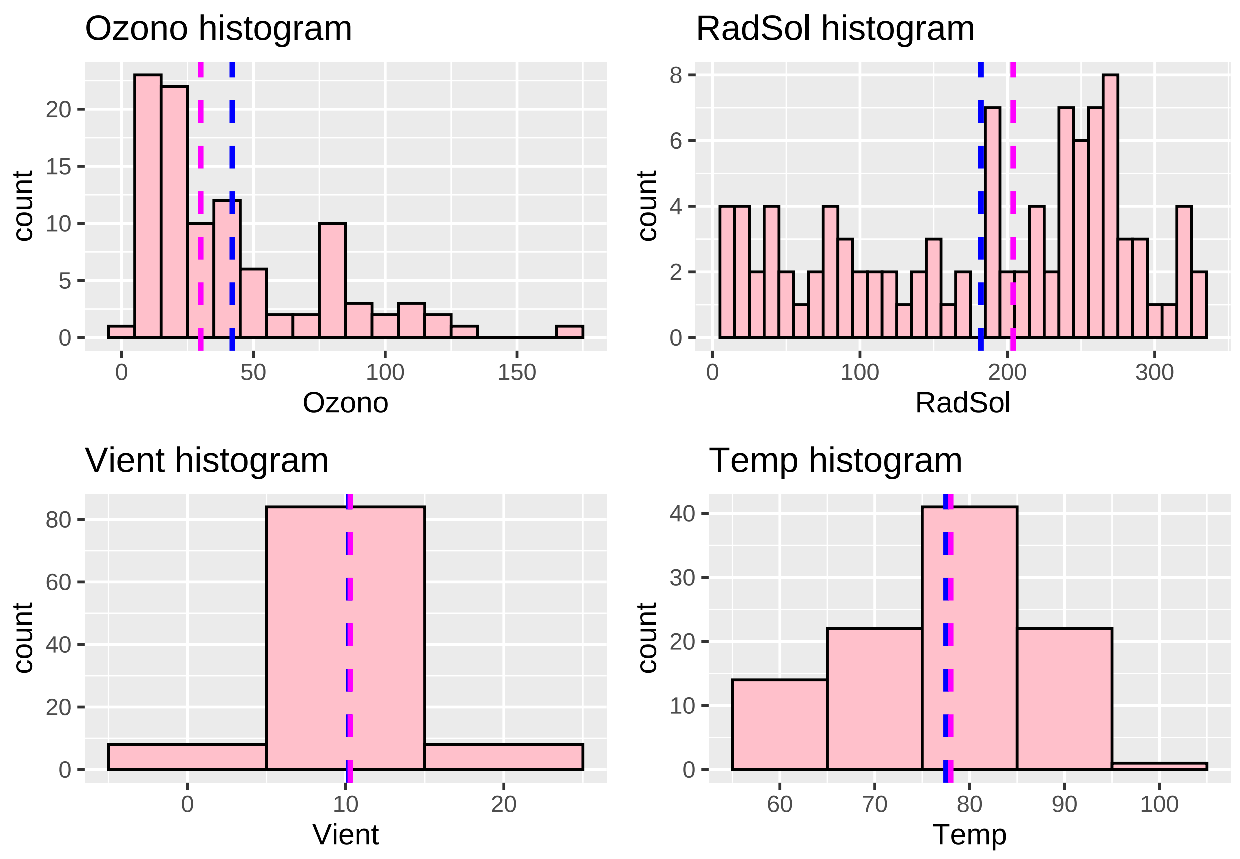
```
##      Ozono            RadSol        Vient            Temp
##  Min.   :  1.00   Min.   :  7   Min.   : 2.30   Min.   :56.00
##  1st Qu.: 16.00   1st Qu.: 92   1st Qu.: 7.40   1st Qu.:72.00
##  Median : 30.00   Median :204   Median :10.30   Median :78.00
##  Mean   : 42.01   Mean   :182   Mean   :10.21   Mean   :77.52
##  3rd Qu.: 64.25   3rd Qu.:259   3rd Qu.:12.15   3rd Qu.:84.25
##  Max.   :168.00   Max.   :334   Max.   :20.70   Max.   :96.00
```

Graphical method analysis

Frequency histograms with mean and median lines

To analyze the numeric variables: `Ozono`, `RadSol`, `Vient`, `Temp`, histograms will be used. These help visualize the distribution of the data and allow us to identify whether the variables follow a symmetrical distribution or are skewed.

```r
plots <- list()

for (col in colnames(numeric_vars)) {
  mean_val <- mean(numeric_vars[[col]], na.rm = TRUE)
  median_val <- median(numeric_vars[[col]], na.rm = TRUE)

  p <- ggplot(numeric_vars, aes_string(x = col)) +
        geom_histogram(binwidth = 10, fill = "pink", color = "black") +
        geom_vline(xintercept = mean_val, color = "blue", linetype = "dashed", size = 1) +
        geom_vline(xintercept = median_val, color = "magenta", linetype = "dashed", size = 1) +
        ggtitle(paste(col, "histogram"))
  plots[[col]] <- p
}

n_col <- 2
do.call("grid.arrange", c(plots, ncol = n_col))
```



Vertical lines indicating the mean (blue) and median (magenta) were added to each variable, making it easier to compare both measures and identify asymmetry. If the mean and median are far apart, it is likely that the distribution is skewed, as seen with the variables `Ozono` and `RadSol`.

Normal probability distribution graph

We first utilized auxiliary resources to generate frequency tables:

```r
tb.freq <- function(x){
  f_i <- as.vector(table(x)) # absolute frequency
  F_i <- cumsum(f_i) # cumulative frequency
  h_i <- f_i / length(x) # relative frequency
  H_i <- F_i / length(x) # cumulative relative frequency
  tf <- cbind(f_i, F_i, h_i, H_i)
  row.names(tf) <- names(table(x))
  tf
}
```

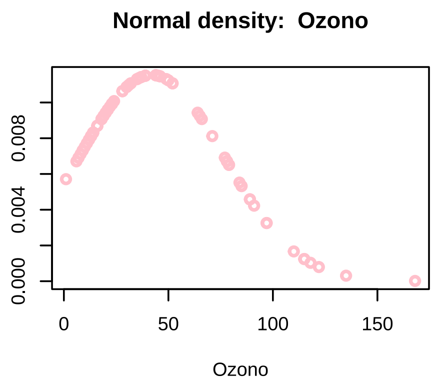
Then, using the created function, we generated the frequency tables based on Sturges' rule. Additionally, intervals were defined, data cuts and groupings were generated, and the frequency tables were visualized.

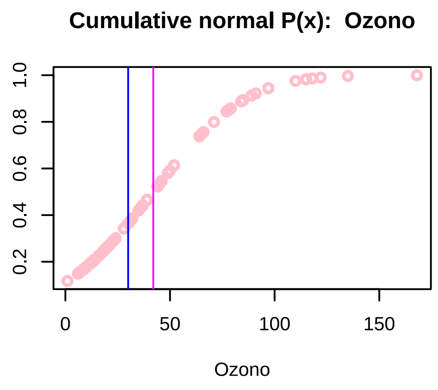
```r
# Frequency distribution tables
for (var_name in colnames(numeric_vars)) {
  print(paste("Frequency distribution table for:", var_name))
  tabla <- fdt(numeric_vars[[var_name]], breaks = "Sturges")
  print(tabla)

  # Interval calculation
  interval <- (max(numeric_vars[[var_name]]) - min(numeric_vars[[var_name]])) / (1 + 3.322 * log10(nrow
  print(paste("Calculated interval for", var_name, ":", interval))

  # Data cuts and grouping
  cut_var <- cut(numeric_vars[[var_name]], breaks = seq(min(numeric_vars[[var_name]]), max(numeric_vars
  numeric_vars[[paste(var_name, "cut", sep = ".")]] <- cut_var

  # Frequency table
  print(tb.freq(cut_var))
}
```

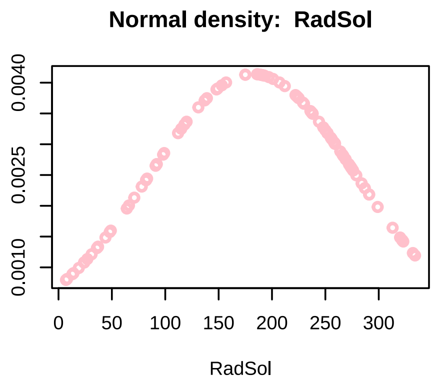
```
## [1] "Frequency distribution table for: Ozono"
##       Class limits  f   rf rf(%)  cf cf(%)
##     [0.99,22.0762) 40 0.40    40  40    40
##  [22.0762,43.1625) 23 0.23    23  63    63
##  [43.1625,64.2488) 12 0.12    12  75    75
##   [64.2488,85.335) 13 0.13    13  88    88
##   [85.335,106.421)  5 0.05     5  93    93
##  [106.421,127.507)  5 0.05     5  98    98
##  [127.507,148.594)  1 0.01     1  99    99
##   [148.594,169.68)  1 0.01     1 100   100
## [1] "Calculated interval for Ozono : 21.847200418629"
##             f_i F_i  h_i  H_i
## (1,22.8]     39  39 0.39 0.39
## (22.8,44.7]  26  65 0.26 0.65
## (44.7,66.5]  11  76 0.11 0.76
## (66.5,88.4]  11  87 0.11 0.87
## (88.4,110]    6  93 0.06 0.93
## (110,132]     4  97 0.04 0.97
## (132,154]     1  98 0.01 0.98
## [1] "Frequency distribution table for: RadSol"
##       Class limits  f   rf rf(%)  cf cf(%)
##     [6.93,48.2312) 15 0.15    15  15    15
##  [48.2312,89.5325)  8 0.08     8  23    23
##  [89.5325,130.834)  9 0.09     9  32    32
##  [130.834,172.135)  7 0.07     7  39    39
##  [172.135,213.436) 13 0.13    13  52    52
##  [213.436,254.737) 18 0.18    18  70    70
##  [254.737,296.039) 22 0.22    22  92    92
##   [296.039,337.34)  8 0.08     8 100   100
## [1] "Calculated interval for RadSol : 42.7786499215071"
##             f_i F_i  h_i  H_i
## (7,49.8]     15  15 0.15 0.15
## (49.8,92.6]  10  25 0.10 0.25
## (92.6,135]    7  32 0.07 0.32
## (135,178]     8  40 0.08 0.40
## (178,221]    11  51 0.11 0.51
## (221,264]    25  76 0.25 0.76
## (264,306]    16  92 0.16 0.92
## [1] "Frequency distribution table for: Vient"
##     Class limits  f   rf rf(%)  cf cf(%)
##   [2.277,4.6057)  8 0.08     8   8     8
##  [4.6057,6.9345) 15 0.15    15  23    23
##  [6.9345,9.2632) 17 0.17    17  40    40
##  [9.2632,11.592) 32 0.32    32  72    72
##  [11.592,13.921) 10 0.10    10  82    82
##   [13.921,16.25) 13 0.13    13  95    95
##   [16.25,18.578)  3 0.03     3  98    98
##  [18.578,20.907)  2 0.02     2 100   100
## [1] "Calculated interval for Vient : 2.40711669283098"
##             f_i F_i  h_i  H_i
## (2.3,4.71]    7   7 0.07 0.07
## (4.71,7.11]  15  22 0.15 0.22
## (7.11,9.52]  17  39 0.17 0.39
## (9.52,11.9]  32  71 0.32 0.71
## (11.9,14.3]  15  86 0.15 0.86
## (14.3,16.7]  10  96 0.10 0.96
## (16.7,19.1]   1  97 0.01 0.97
## [1] "Frequency distribution table for: Temp"
##   Class limits  f   rf rf(%)  cf cf(%)
##  [55.44,60.63)  6 0.06     6   6     6
##  [60.63,65.82)  8 0.08     8  14    14
##  [65.82,71.01) 10 0.10    10  24    24
##   [71.01,76.2) 20 0.20    20  44    44
##   [76.2,81.39) 22 0.22    22  66    66
##  [81.39,86.58) 15 0.15    15  81    81
##  [86.58,91.77) 11 0.11    11  92    92
##  [91.77,96.96)  8 0.08     8 100   100
## [1] "Calculated interval for Temp : 5.23286237571952"
##              f_i F_i  h_i  H_i
## (56,61.2]      7   7 0.07 0.07
## (61.2,66.5]    6  13 0.06 0.13
## (66.5,71.7]   10  23 0.10 0.23
## (71.7,76.9]   20  43 0.20 0.43
## (76.9,82.2]   27  70 0.27 0.70
## (82.2,87.4]   13  83 0.13 0.83
## (87.4,92.6]   12  95 0.12 0.95
```

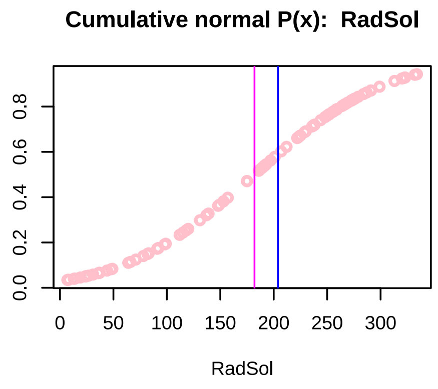
Next, we generated normal density and cumulative probability graphs for each of the numeric variables.

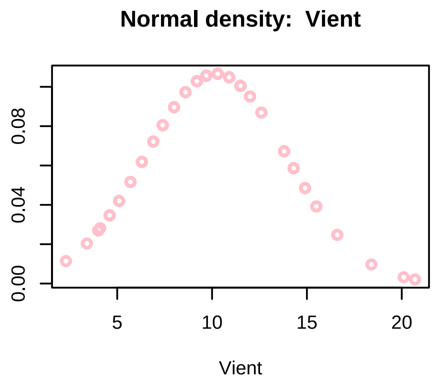
```r
par(mfrow = c(4, 2))

for (var_name in colnames(numeric_vars)[1:4]) {

# Density vs normal distribution graph

  plot(numeric_vars[[var_name]],
       dnorm(numeric_vars[[var_name]],
             mean = mean(numeric_vars[[var_name]],
                         na.rm = TRUE),
             sd = sd(numeric_vars[[var_name]],
                     na.rm = TRUE)),
       ylab = "",xlab = var_name,
       lwd = 2, col = "pink", main = paste("Normal density: ", var_name))

# Cumulative probability vs normal distribution graph

  plot(numeric_vars[[var_name]],
       pnorm(numeric_vars[[var_name]],
             mean = mean(numeric_vars[[var_name]],na.rm = TRUE),
             sd = sd(numeric_vars[[var_name]],na.rm = TRUE)),
       xlab = var_name,
       ylab = "",
       lwd  = 2,
       col  = "pink",
       main = paste("Cumulative normal P(x): ", var_name))

# Mean and median as vertical lines

  abline(v = mean(numeric_vars[[var_name]], na.rm = TRUE),
         col = "magenta")
  abline(v = median(numeric_vars[[var_name]], na.rm = TRUE),
         col = "blue")

}
```

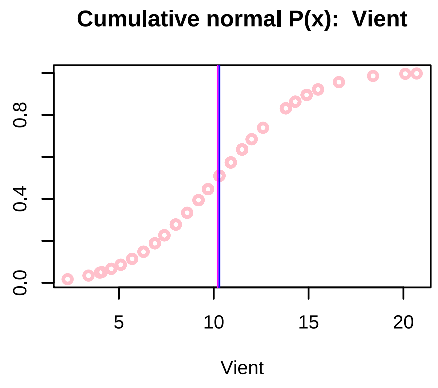

Normal density: Ozono
0.000
0.004
0.008
0
50
100
150
Ozono

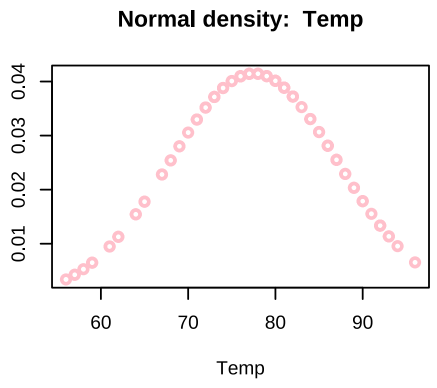

Cumulative normal P(x): Ozono
0.2
0.4
0.6
0.8
1.0
0
50
100
150
Ozono

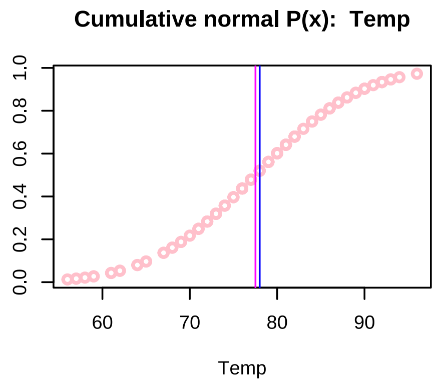

Normal density: RadSol
0.0010
0.0025
0.0040
0
50
100
150
200
250
300
RadSol


Cumulative normal P(x): RadSol
0.0
0.2
0.4
0.6
0.8
0
50
100
150
200
250
300
RadSol


Normal density: Vient
0.00
0.04
0.08
5
10
15
20
Vient


Cumulative normal P(x): Vient
0.0
0.4
0.8
5
10
15
20
Vient


Normal density: Temp
0.01
0.02
0.03
0.04
60
70
80
90
Temp


Cumulative normal P(x): Temp
0.0
0.2
0.4
0.6
0.8
1.0
60
70
80
90
Temp

Mathematical Analysis

In addition to the graphical analysis, a mathematical approach is applied, defining hypotheses, and calculating `skewness` and `kurtosis` for each variable.

Hypothesis Formulation

We define the following hypotheses:

- H_0 : The variable is normally distributed.
- H_1 : The variable is not normally distributed.

A significance level of 0.05 is applied. If the p-value is greater than this threshold, we fail to reject the null hypothesis, implying that the variable may follow a normal distribution.

Shapiro-Wilk Test

The Shapiro-Wilk test, which is robust for small sample sizes, is performed to assess normality.

```
for (col in colnames(air_impt)[1:4]) {
  cat("\nVariable:", col)
  cat("\nPrueba de Shapiro-Wilk:\n")
  print(shapiro.test(air_impt[[col]]))
}
```

```
##
## Variable: Ozono
## Prueba de Shapiro-Wilk:
##
##  Shapiro-Wilk normality test
##
## data:  air_impt[[col]]
## W = 0.85664, p-value = 2.102e-08
##
##
## Variable: RadSol
## Prueba de Shapiro-Wilk:
##
##  Shapiro-Wilk normality test
##
## data:  air_impt[[col]]
## W = 0.92599, p-value = 2.968e-05
##
##
## Variable: Vient
## Prueba de Shapiro-Wilk:
##
##  Shapiro-Wilk normality test
##
## data:  air_impt[[col]]
## W = 0.9834, p-value = 0.242
##
```

```
##
## Variable: Temp
## Prueba de Shapiro-Wilk:
##
##  Shapiro-Wilk normality test
##
## data:  air_impt[[col]]
## W = 0.97598, p-value = 0.06438
```

Normality summary for numerical variables

```
ft2 <- flextable(normality(numeric_vars))
ft2 <- set_caption(ft2, caption = "Normality table by variables")

ft2
```

Table 2: Normality table by variables

vars	statistic	p_value	sample
Ozono	0.8566350	0.00000002101889	100
RadSol	0.9259890	0.00002968172350	100
Vient	0.9833954	0.24199785241031	100
Temp	0.9759791	0.06437900956539	100

Skewness

`Skewness` measures the symmetry of the distribution. Values close to 0 indicate that the data distribution is relatively symmetrical.

```
for (col in colnames(air)[1:4]) {
  cat("\nVariable:", col)
  cat("\nSkewness:", skewness(air_impt[[col]], na.rm = TRUE))
}
```

```
##
## Variable: Ozono
## Skewness: 1.257036
## Variable: RadSol
## Skewness: -0.3720618
## Variable: Vient
## Skewness: 0.3007779
## Variable: Temp
## Skewness: -0.3287795
```

The variable `Ozone` shows the highest skewness (around 1.26), while the other variables display low skewness values (approximately 0.3). `Vient` has positive skewness, whereas `RadSol` and `Temp` exhibit negative skewness.

Kurtosis

`Kurtosis` assesses the "height" and "width" of the tails of the distribution. Positive kurtosis indicates heavier tails than a normal distribution, while negative kurtosis indicates lighter tails.

```r
for (col in colnames(air_impt)[1:4]) {
  cat("\nVariable:", col)
  cat("\nKurtosis:", kurtosis(air_impt[[col]], na.rm = TRUE))
}
```

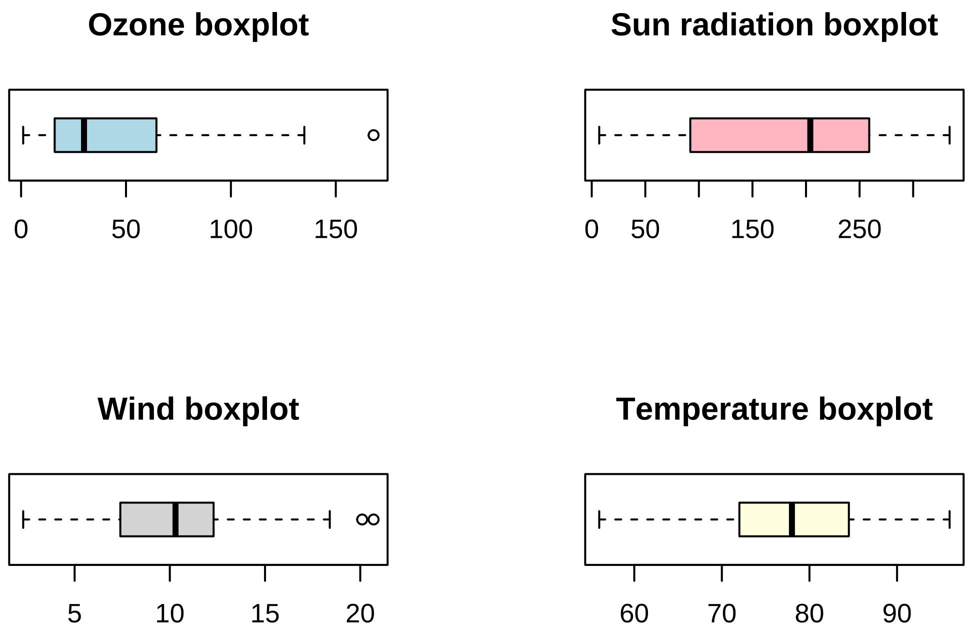
```
##
## Variable: Ozono
## Kurtosis: 1.053549
## Variable: RadSol
## Kurtosis: -1.161496
## Variable: Vient
## Kurtosis: -0.1396004
## Variable: Temp
## Kurtosis: -0.4922435
```

The variables `RadSol`, `Vient`, and `Temp` display platykurtic tendencies, with `RadSol` showing the most pronounced trend. On the other hand, `Ozone` exhibits leptokurtic behavior.

Outlier analysis

Boxplots allow for the identification of outliers, represented as points outside the "whiskers" of the graph.

```r
par(mfrow = c(2, 2))

boxplot(air_impt$Ozono,  main       = "Ozone boxplot",
                         col        = "lightblue",
                         horizontal = TRUE
                         )

boxplot(air_impt$RadSol, main       = "Sun radiation boxplot",
                         col        = "lightpink",
                         horizontal = TRUE
                         )

boxplot(air_impt$Vient,  main       = "Wind boxplot",
                         col        = "lightgrey",
                         horizontal = TRUE
                         )

boxplot(air_impt$Temp,   main       = "Temperature boxplot",
                         col        = "lightyellow",
                         horizontal = TRUE
                         )
```



Through these boxplots, we visually identify outliers in `Ozone` and `Wind`. To analyze these outliers in detail, we use the `flextable(diagnose_outlier())` function:

```
ft3 <- flextable::flextable(diagnose_outlier(air_impt))
ft3 <- set_table_properties(ft3, width = 0.75, layout = "autofit")
ft3 <- align(ft3, align = "center", part = "all")
ft3 <- set_caption(ft3, caption = "Outlier details by variables")

ft3
```

Table 3: Outlier details by variables

variables	outliers_cnt	outliers_ratio	outliers_mean	with_mean	without_mean
Ozono	1	1	168.0	42.010	40.737374
RadSol	0	0		182.010	182.010000
Vient	2	2	20.4	10.206	9.997959
Temp	0	0		77.520	77.520000

The outlier analysis provides critical information, including the number and mean values of the outliers:

- Ozone: 1 outlier with a mean of 168.
- Wind : 2 outliers with a mean of 20.4.

This data is essential for making informed decisions regarding the treatment of these outliers, such as whether to remove them or implement other adjustments.

Conclusions

The dataset had a low percentage of missing values (3.7%), primarily in the `Ozono` and `RadSol` variables, which were effectively imputed using the mice package. The transformation of the month and day variables into categorical types confirmed their suitability for analysis, with no unexpected distribution patterns. The Shapiro-Wilk test and histogram analysis showed significant deviations from normality in variables like `Ozono` and `RadSol`, with Ozone having the highest skewness (1.26), suggesting the need for transformations or non-parametric modeling. Visualizations supported these findings, highlighting the potential impact of extreme events on air quality metrics.

Future Work

Given the skewed distribution of certain variables, applying transformations (e.g., logarithmic or square root) to these variables may improve the results of further statistical or machine learning models. Additional analyses, such as multivariate relationships between variables, could provide deeper insights into the factors driving variations in air quality metrics.