Final Project STAT206

Anshara Fatima

2025-07-01

```r
# Clear environment
rm(list = ls())
```

```r
# Step1: Load Dataset

my_data <- read.csv("air_quality_health_dataset.csv", stringsAsFactors = FALSE)

# Preview the data
head(my_data)
```

```
##          city     date aqi pm2_5 pm10  no2   o3 temperature humidity
## 1 Los Angeles 1/1/2020  65  34.0 52.7  2.2 38.5        33.5       33
## 2     Beijing 1/2/2020 137  33.7 31.5 36.7 27.5        -1.6       32
## 3      London 1/3/2020 266  43.0 59.6 30.4 57.3        36.4       25
## 4 Mexico City 1/4/2020 293  33.7 37.9 12.3 42.7        -1.0       67
## 5       Delhi 1/5/2020 493  50.3 34.8 31.2 35.6        33.5       72
## 6       Cairo 1/6/2020  28  67.2 44.9 41.9 47.8         7.9       89
##   hospital_admissions population_density hospital_capacity
## 1                   5              Rural              1337
## 2                   4              Urban              1545
## 3                  10           Suburban              1539
## 4                  10              Urban               552
## 5                   9           Suburban              1631
## 6                  11              Urban              1291
```

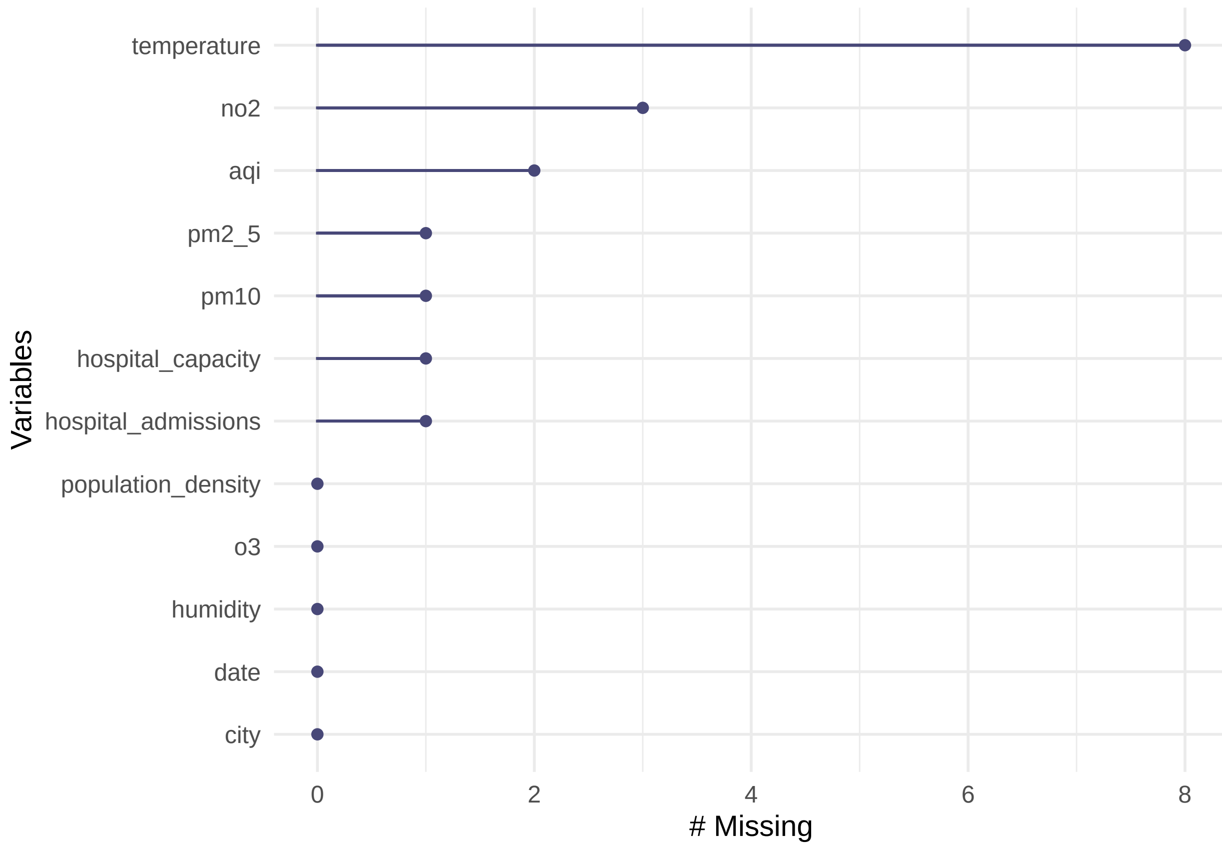
```r
#Step 2: Handling Missing Data

# Load library
library(naniar) #naniar package helps find, explore and fix missing data eg. N/A values
```

```
## Warning: package 'naniar' was built under R version 4.4.3
```

```r
# Summary of missing values
colSums(is.na(my_data)) #tells us how many missing values are there in each column after turning all N/
```

```
##                city                date                 aqi               pm2_5
##                   0                   0                   2                   1
##                pm10                 no2                  o3         temperature
##                   1                   3                   0                   8
##            humidity hospital_admissions  population_density   hospital_capacity
##                   0                   1                   0                   1
```

```
# Visualize missing data
gg_miss_var(my_data) #naniar package function. shows a bar graph that displays how much data is missing
```




```
# Impute missing numeric values with mean
# Looks at every column one by one. If it is a number column, it finds average of that column while ign
my_data[] <- lapply(my_data, function(x) {
  if (is.numeric(x)) {
    x[is.na(x)] <- mean(x, na.rm = TRUE)
  }
  return(x)
})
```

```
# This code will find outliers in my data and will replace them with a safe limit
# Step 3: Function to cap outliers using IQR

#  The following function creates a custom function called cap_outliers. It will work on one column at
cap_outliers <- function(x) {
  if (is.numeric(x)) {                    #if the column is in numbers, do:
    Q1 <- quantile(x, 0.25, na.rm = TRUE) # no. below with 25% data lies (low
    Q3 <- quantile(x, 0.75, na.rm = TRUE) # no. below which 75% data lies (upper
    IQR_val <- Q3 - Q1   # difference between Q1 & Q3 telling us how far spread out
    lower <- Q1 - 1.5 * IQR_val # Lower limit: Anything smaller than this is an
    upper <- Q3 + 1.5 * IQR_val # Upper limit: Anything bigger than this is an
    x[x < lower] <- lower       # if a number is too small, change to lower limit
    x[x > upper] <- upper       # if a number is too big, change to upper limit
```

```r
  }
  return(x)
}

# Apply to numeric columns, each column one by one
my_data[] <- lapply(my_data, cap_outliers)
```

```r
# Step 4: Create new column; percentage of hospital capacity used by respiratory admissions

#This new column tells me what % of the hospital beds are being used by patients

my_data$resp_admission_pct <- (my_data$hospital_admissions / my_data$hospital_capacity) * 100

# View the new column (First few results)
head(my_data$resp_admission_pct)
```

```
## [1] 0.3739716 0.2588997 0.6497726 1.8115942 0.5518087 0.8520527
```

```r
# Add to the data preview; lets me see my data in a table
head(my_data[, c("city", "hospital_admissions", "hospital_capacity", "resp_admission_pct")])
```

```
##          city hospital_admissions hospital_capacity resp_admission_pct
## 1 Los Angeles                   5              1337          0.3739716
## 2     Beijing                   4              1545          0.2588997
## 3      London                  10              1539          0.6497726
## 4 Mexico City                  10               552          1.8115942
## 5       Delhi                   9              1631          0.5518087
## 6       Cairo                  11              1291          0.8520527
```

```r
# Step 5: Subset Data for Each City
cities <- unique(my_data$city) # looks at city column in my data and finds
                               # all city names & removes duplicates
for (c in cities) {       # this starts a loop. for every city, do the same:
  city_data <- subset(my_data, city == c)  #pick all data for this one city
  write.csv(city_data, paste0(c, ".csv"), row.names = FALSE) #saves .csv file for
}
```

```r
# Step 6: List city files

# List all CSV files in the directory
city_files <- list.files(pattern = "\\.csv$", ignore.case = TRUE)

# Keep only safe file names with regular letters, numbers, space, underscore, dot, or dash
clean_city_files <- city_files[grepl("^[A-Za-z0-9 _.-]+\\.csv$", city_files)]

# Check which files are valid
print(clean_city_files)
```

```
##  [1] "air_quality_health_dataset.csv" "Beijing.csv"
##  [3] "Cairo.csv"                      "combined_data.csv"
##  [5] "Delhi.csv"                      "London.csv"
##  [7] "Los Angeles.csv"                "Mexico City.csv"
##  [9] "Sao Paulo.csv"                  "Tokyo.csv"
```

```r
# Open each .csv file one by one and it shows me a small part of it so I can check the data
for (file in clean_city_files) {
  city_data <- read.csv(file)
  cat("Preview of:", file, "\n")
  print(head(city_data[, 1:6]))
}
```

```
## Preview of: air_quality_health_dataset.csv
##          city     date aqi pm2_5 pm10  no2
## 1 Los Angeles 1/1/2020  65  34.0 52.7  2.2
## 2     Beijing 1/2/2020 137  33.7 31.5 36.7
## 3      London 1/3/2020 266  43.0 59.6 30.4
## 4 Mexico City 1/4/2020 293  33.7 37.9 12.3
## 5       Delhi 1/5/2020 493  50.3 34.8 31.2
## 6       Cairo 1/6/2020  28  67.2 44.9 41.9
## Preview of: Beijing.csv
##      city      date aqi pm2_5  pm10  no2
## 1 Beijing  1/2/2020 137  33.7  31.5 36.7
## 2 Beijing 1/10/2020 279  27.1 101.0 47.8
## 3 Beijing 1/11/2020 484  56.6  46.3 33.2
## 4 Beijing 1/15/2020 475  45.0  47.6 38.5
## 5 Beijing 1/22/2020 164  31.1  66.1 20.2
## 6 Beijing 1/27/2020 382  19.5  33.3 24.8
## Preview of: Cairo.csv
##    city      date aqi pm2_5 pm10  no2
## 1 Cairo  1/6/2020  28  67.2 44.9 41.9
## 2 Cairo 1/23/2020 429  61.3 43.2 30.1
## 3 Cairo  2/1/2020 263  47.4 63.0 32.0
## 4 Cairo  4/9/2020 285  46.7 50.7 46.5
## 5 Cairo 4/13/2020 320  33.5 57.1 23.8
## 6 Cairo 5/16/2020 190  56.0 49.7 46.3
## Preview of: combined_data.csv
##          city aqi pm2_5 pm10  no2   o3
## 1 Los Angeles  65  34.0 52.7  2.2 38.5
## 2     Beijing 137  33.7 31.5 36.7 27.5
## 3      London 266  43.0 59.6 30.4 57.3
## 4 Mexico City 293  33.7 37.9 12.3 42.7
## 5       Delhi 493  50.3 34.8 31.2 35.6
## 6       Cairo  28  67.2 44.9 41.9 47.8
## Preview of: Delhi.csv
##    city      date aqi pm2_5 pm10  no2
## 1 Delhi  1/5/2020 493  50.3 34.8 31.2
## 2 Delhi  1/9/2020 342  44.9 63.4 31.0
## 3 Delhi 1/16/2020 423  34.2 31.4 36.1
## 4 Delhi 1/18/2020 475  44.4 34.8 41.4
## 5 Delhi 1/26/2020 184  40.3 57.1 15.6
## 6 Delhi 1/28/2020 132  47.1 51.1 23.4
## Preview of: London.csv
##     city      date aqi pm2_5    pm10  no2
## 1 London  1/3/2020 266  43.0 59.6000 30.4
## 2 London 1/14/2020 290   0.4 43.0000 25.3
## 3 London 1/19/2020  34  41.5 17.3000 40.3
## 4 London 2/11/2020 257  59.1 49.7000 37.6
```

```
## 5 London 2/14/2020 347  36.3 50.1183 34.1
## 6 London 2/15/2020  52  31.3 78.5000 31.6
## Preview of: Los Angeles.csv
##          city      date aqi pm2_5 pm10  no2
## 1 Los Angeles  1/1/2020  65  34.0 52.7  3.2
## 2 Los Angeles  1/7/2020 217  29.0 63.7 22.3
## 3 Los Angeles  1/8/2020 449  60.8 56.2 40.0
## 4 Los Angeles 1/21/2020 402  47.0 13.8 29.2
## 5 Los Angeles 1/25/2020 385  24.0 31.5 28.1
## 6 Los Angeles 1/31/2020 388  30.6 53.8 10.7
## Preview of: Mexico City.csv
##          city      date aqi pm2_5 pm10  no2
## 1 Mexico City  1/4/2020 293  33.7 37.9 12.3
## 2 Mexico City 1/13/2020 276  46.3 50.5 32.8
## 3 Mexico City 1/17/2020 269  41.3 46.0 22.9
## 4 Mexico City 1/20/2020 234  42.3 47.3 17.2
## 5 Mexico City 1/30/2020 470  32.6 37.0 39.7
## 6 Mexico City  2/9/2020  40  39.4 24.4 10.0
## Preview of: Sao Paulo.csv
##        city      date aqi pm2_5 pm10  no2
## 1 São Paulo 6/26/2020 392  57.6 24.4 43.5
## 2 São Paulo 1/20/2021 301  22.5 20.2 46.9
## 3 São Paulo  2/8/2021 396  41.8 21.6 47.9
## 4 São Paulo 2/13/2021 385  40.1 40.0 21.1
## 5 São Paulo 3/22/2021  97  23.6 45.7 23.6
## 6 São Paulo  4/1/2021  26  27.3 84.7 13.3
## Preview of: Tokyo.csv
##    city      date aqi pm2_5 pm10  no2
## 1 Tokyo 1/12/2020 472  44.7 56.6 35.1
## 2 Tokyo 1/24/2020 161  44.4 51.6 25.3
## 3 Tokyo 4/10/2020 359  35.7 35.4 35.6
## 4 Tokyo 4/19/2020 469  24.0 73.3 34.7
## 5 Tokyo 5/11/2020 349  57.2 14.9 34.8
## 6 Tokyo 5/22/2020  97  61.5 47.8 26.7
```

```r
# Step 7: Combine all the data into one file and write down a function that will help calculate five va

# Load necessary packages
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.4.3

##
## 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
```

```r
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 4.4.3
```

```r
# List all .csv files
city_files <- list.files(pattern = "\\.csv$", ignore.case = TRUE)

# Keep only file names with safe characters (A-Z, a-z, 0-9, underscore, space, dot, dash)
city_files <- city_files[grepl("^[A-Za-z0-9 _.-]+\\.csv$", city_files)]

# Now safely read the cleaned files
library(dplyr)

city_data_list <- lapply(city_files, function(file) {
  df <- read.csv(file)
  # Select only consistent columns from each file
  select(df, city, aqi, pm2_5, pm10, no2, o3, temperature, humidity,
         hospital_admissions, hospital_capacity)
})

# Combine all cleaned city data into one dataset
combined_data <- bind_rows(city_data_list)


# Combine the list into one data frame
combined_data <- bind_rows(city_data_list)

# Function for five-number summary + SD: This function creates a simple summary table of important numb

five_summary <- function(df) {
  numeric_df <- df %>% select(where(is.numeric))

  summary_df <- data.frame(
    Variable = names(numeric_df),
    Min = sapply(numeric_df, min, na.rm = TRUE),
    Max = sapply(numeric_df, max, na.rm = TRUE),
    Mean = sapply(numeric_df, mean, na.rm = TRUE),
    Median = sapply(numeric_df, median, na.rm = TRUE),
    SD = sapply(numeric_df, sd, na.rm = TRUE)
  )

  return(summary_df)
}

# Run the summary function
summary_table <- five_summary(combined_data)

# Display the results
kable(summary_table, caption = "Five-Number Summary (plus SD) of Continuous Variables")
```

Table 1: Five-Number Summary (plus SD) of Continuous Variables

	Variable	Min	Max	Mean	Median	SD
aqi	aqi	0	499.0	249.370518	249.0	144.477616
pm2_5	pm2_5	0	109.9	35.136815	35.1	14.741615
pm10	pm10	0	143.5	50.106042	50.0	19.759136
no2	no2	0	71.4	30.003453	30.0	9.941203
o3	o3	0	93.5	39.977137	40.0	11.972940
temperature	temperature	-5	40.0	17.523445	17.5	12.960691
humidity	humidity	20	94.0	56.950966	57.0	21.629559
hospital_admissions	hospital_admissions	0	25.0	8.033134	8.0	3.673724
hospital_capacity	hospital_capacity	50	1999.0	1024.454050	1026.0	561.970119

```r
# This is to save combined_data (after combining all cities)
write.csv(combined_data, "combined_data.csv", row.names = FALSE)
```

```r
# Step 8: Which city has the greatest number of cases reported?
library(dplyr)

# Group by city and sum hospital admissions
city_cases <- combined_data %>%
  group_by(city) %>%
  summarise(total_admissions = sum(hospital_admissions, na.rm = TRUE)) %>%
  arrange(desc(total_admissions))

# Display the full table
print(city_cases)
```

```
## # A tibble: 8 x 2
##   city        total_admissions
##   <chr>                  <dbl>
## 1 Delhi                3823974
## 2 Beijing              3209040
## 3 Mexico City          1924200.
## 4 Los Angeles          1297341
## 5 London               1010880
## 6 Tokyo                 886635
## 7 Cairo                 393318
## 8 São Paulo             249732
```

```r
# Show the city with the greatest number of cases
top_city <- city_cases[1, ]
cat("City with the greatest number of hospital admissions:",
    top_city$city, "with", top_city$total_admissions, "cases.")
```

```
## City with the greatest number of hospital admissions: Delhi with 3823974 cases.
```

```r
# Step 9: Compare the temperatures of rural, suburban and urban areas

# Add area_type column manually
combined_data$area_type <- case_when(
```

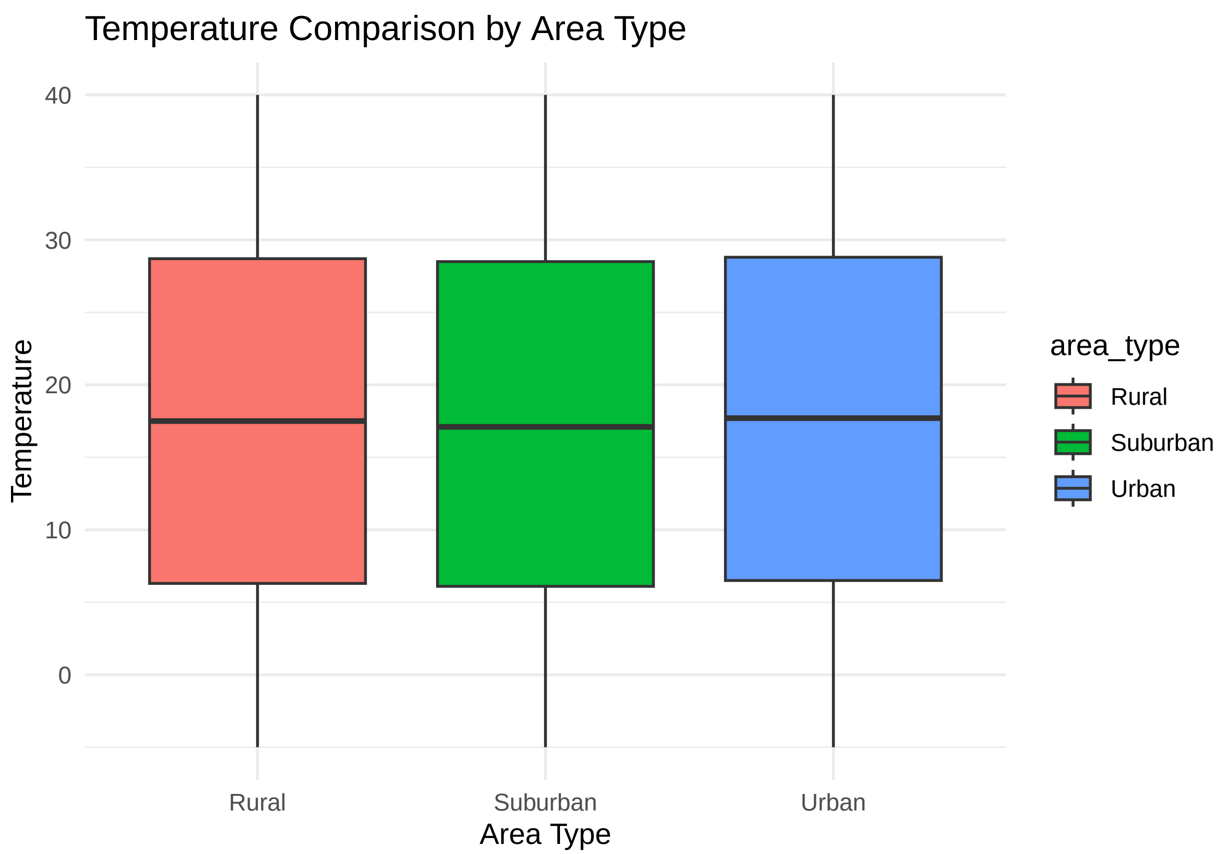
```r
  combined_data$city %in% c("Los Angeles", "Beijing", "Tokyo", "Cairo") ~ "Urban",
  combined_data$city %in% c("London", "São Paulo") ~ "Suburban",
  combined_data$city %in% c("Mexico City", "Delhi") ~ "Rural",
  TRUE ~ "Urban"  # default
)

library(dplyr)
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.4.3
```

```r
# Check if 'area_type' column exists
if("area_type" %in% colnames(combined_data)) {

  # Summary statistics of temperature by area type
  temp_summary <- combined_data %>%
    group_by(area_type) %>%
    summarise(
      Mean_Temperature = mean(temperature, na.rm = TRUE),
      Median_Temperature = median(temperature, na.rm = TRUE),
      SD_Temperature = sd(temperature, na.rm = TRUE),
      .groups = "drop"
    )

  print(temp_summary)

  # Visualize the comparison
  ggplot(combined_data, aes(x = area_type, y = temperature, fill = area_type)) +
    geom_boxplot() +
    labs(title = "Temperature Comparison by Area Type",
         x = "Area Type",
         y = "Temperature") +
    theme_minimal()

} else {
  cat("The dataset does not have an 'area_type' column. Please add it before proceeding.")
}
```

```
## # A tibble: 3 x 4
##   area_type Mean_Temperature Median_Temperature SD_Temperature
##   <chr>                <dbl>              <dbl>          <dbl>
## 1 Rural                 17.5               17.5           13.0
## 2 Suburban              17.3               17.1           13.0
## 3 Urban                 17.6               17.7           13.0
```

```
## Warning: Removed 72 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```

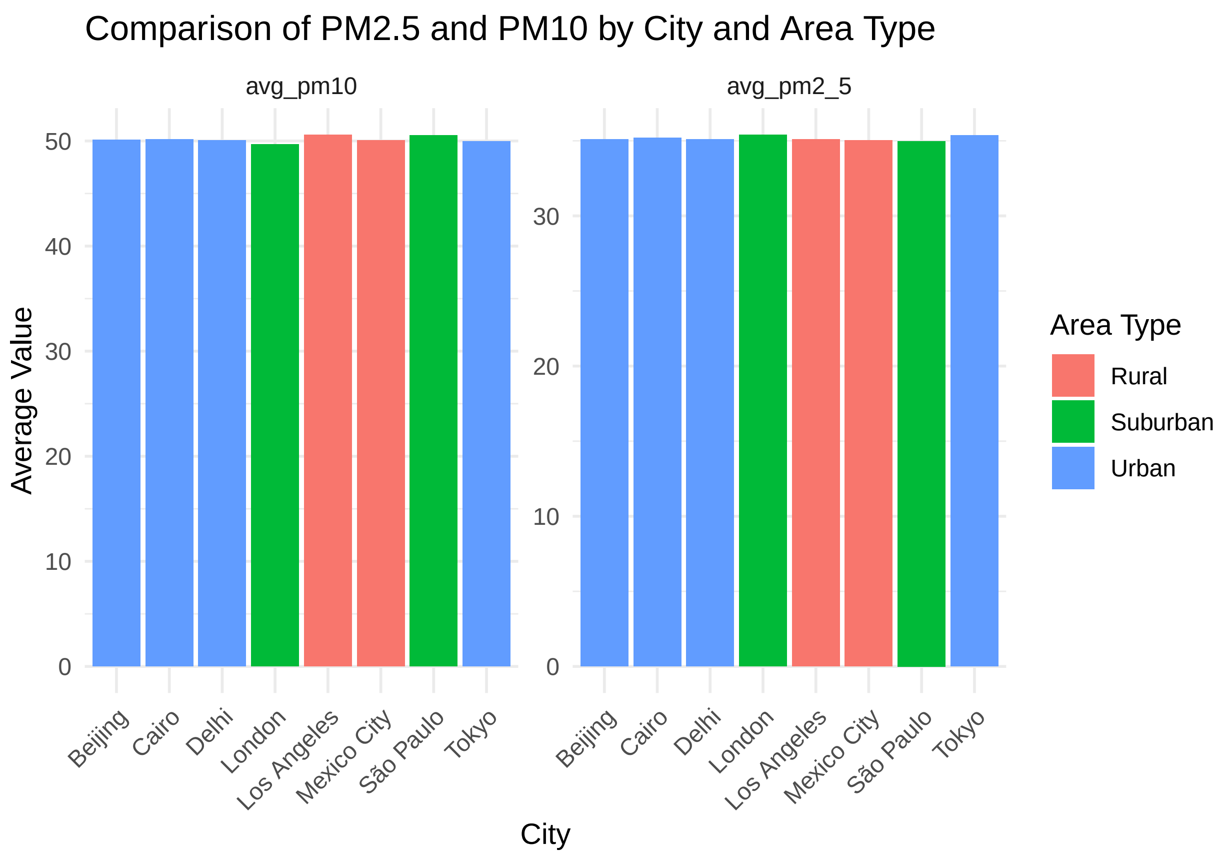



```r
# Do urban areas have higher PM2.5 and PM10 compared to suburban and rural areas?
# show this comparison using a bar chart for each city.

# Load necessary packages
library(dplyr)      # for data manipulation
library(ggplot2)    # for data visualization
library(tidyr)      # for reshaping data
```

```
## Warning: package 'tidyr' was built under R version 4.4.3
```

```r
# ----------------------------
# Add area type to each city
# ----------------------------
# We will classify cities manually into Urban, Suburban, and Rural
combined_data$area_type <- case_when(
  combined_data$city %in% c("Delhi", "Beijing", "Tokyo", "Cairo") ~ "Urban",
  combined_data$city %in% c("London", "São Paulo") ~ "Suburban",
  combined_data$city %in% c("Mexico City", "Los Angeles") ~ "Rural",
  TRUE ~ "Urban"  # any remaining cities will default to Urban
)

# ----------------------------
# Calculate average PM2.5 and PM10 for each city and area type
# ----------------------------
pm_summary <- combined_data %>%
```

```r
  group_by(city, area_type) %>%
  summarise(
    avg_pm2_5 = mean(pm2_5, na.rm = TRUE),   # average PM2.5
    avg_pm10 = mean(pm10, na.rm = TRUE),     # average PM10
    .groups = "drop"
  )

# ----------------------------
# Reshape the data so both PM2.5 and PM10 can be plotted together
# ----------------------------
# This turns two columns (avg_pm2_5 and avg_pm10) into one column of values
pm_long <- pm_summary %>%
  pivot_longer(
    cols = c(avg_pm2_5, avg_pm10),
    names_to = "pollutant",      # this new column will say "avg_pm2_5" or "avg_pm10"
    values_to = "value"          # this will contain the numeric values
  )

# ----------------------------
# Create bar plot to compare pollution levels by area type for each city
# ----------------------------
ggplot(pm_long, aes(x = city, y = value, fill = area_type)) +
  geom_bar(stat = "identity", position = "dodge") +  # make side-by-side bars
  facet_wrap(~ pollutant, scales = "free_y") +       # separate plot for PM2.5 and PM10
  labs(
    title = "Comparison of PM2.5 and PM10 by City and Area Type",
    x = "City",
    y = "Average Value",
    fill = "Area Type"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))  # rotate city names for readability
```




```r
# Step 11: Is there any correlation between the aqi,PM 2.5, PM 10, no2, O3, temperature and humidity in
# Load required packages
library(ggplot2)
library(reshape2)    # for melting the correlation matrix
```

```
## Warning: package 'reshape2' was built under R version 4.4.3
```

```
##
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

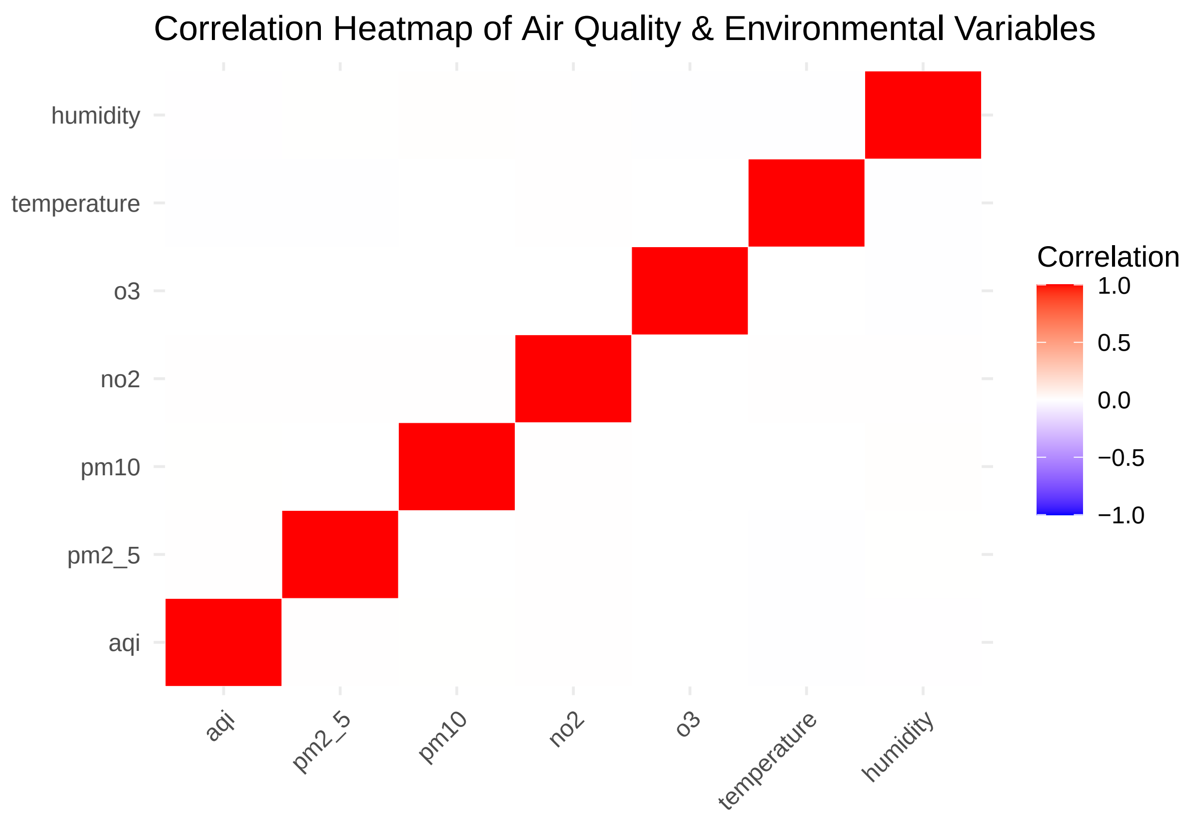
```r
library(corrplot)    # for heatmap
```

```
## Warning: package 'corrplot' was built under R version 4.4.3
```

```
## corrplot 0.95 loaded
```

```r
# ----------------------------
# Select only relevant numeric columns
# ----------------------------
```

```r
# We will only include the variables mentioned in the question
corr_data <- combined_data %>%
  select(aqi, pm2_5, pm10, no2, o3, temperature, humidity)

# ----------------------------
# Calculate correlation matrix
# ----------------------------
# This shows how strongly each variable is related to the others
cor_matrix <- cor(corr_data, use = "complete.obs")

# ----------------------------
# Melt the matrix into long format for plotting with ggplot
# ----------------------------
# Melting turns my big table of correlations into a simple list of variable pairs
#and their values. This makes it easier to plot a heatmap to show correlation.
cor_long <- melt(cor_matrix)

# ----------------------------
# Plot heatmap using ggplot2
# ----------------------------
ggplot(cor_long, aes(Var1, Var2, fill = value)) +
  geom_tile(color = "white") +  # add white grid lines
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                       midpoint = 0, limit = c(-1,1), space = "Lab",
                       name = "Correlation") +
  labs(title = "Correlation Heatmap of Air Quality & Environmental Variables",
       x = "", y = "") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1))
```


Correlation Heatmap of Air Quality & Environmental Variables
humidity
temperature
o3
no2
pm10
pm2_5
aqi
aqi
pm2_5
pm10
no2
o3
temperature
humidity
Correlation
1.0
0.5
0.0
−0.5
−1.0