Assignment 4: Data Wrangling

Soumya Mathew

OVERVIEW

This exercise accompanies the lessons in Environmental Data Analytics on Data Wrangling

Directions

1. Rename this file `<FirstLast>_A03_DataExploration.Rmd` (replacing `<FirstLast>` with your first and last name).
2. Change "Student Name" on line 3 (above) with your name.
3. Work through the steps, **creating code and output** that fulfill each instruction.
4. Be sure to **answer the questions** in this assignment document.
5. When you have completed the assignment, **Knit** the text and code into a single PDF file.

The completed exercise is due on Friday, Oct 7th @ 5:00pm.

Set up your session

1. Check your working directory, load the `tidyverse` and `lubridate` packages, and upload all four raw data files associated with the EPA Air dataset, being sure to set string columns to be read in a factors. See the README file for the EPA air datasets for more information (especially if you have not worked with air quality data previously).

2. Explore the dimensions, column names, and structure of the datasets.

```
# 1
getwd()
```

```
## [1] "C:/Users/user/Desktop/Soumya/Year 2/EDA/EDA-Fall2022/Assignments"
```

```
# install.packages(tidyverse)
library(tidyverse)
# install.packages(lubridate)
library(lubridate)
library(dplyr)

# loadinng data sets
EPAair_O3_18 <- read.csv("C:/Users/user/Desktop/Soumya/Year 2/EDA/EDA-Fall2022/Data/Raw/EPAair_O3_NC2018
    stringsAsFactors = TRUE)

EPAair_O3_19 <- read.csv("C:/Users/user/Desktop/Soumya/Year 2/EDA/EDA-Fall2022/Data/Raw/EPAair_O3_NC2019
    stringsAsFactors = TRUE)
```

```
EPAair_PM25_18 <- read.csv("C:/Users/user/Desktop/Soumya/Year 2/EDA/EDA-Fall2022/Data/Raw/EPAair_PM25_N
    stringsAsFactors = TRUE)

EPAair_PM25_19 <- read.csv("C:/Users/user/Desktop/Soumya/Year 2/EDA/EDA-Fall2022/Data/Raw/EPAair_PM25_N
    stringsAsFactors = TRUE)

# 2 Exploring dataset 1 colnames(EPAair_O3_18) head(EPAair_O3_18)
# summary(EPAair_O3_18) str(EPAair_O3_18) dim(EPAair_O3_18)

# Exploring dataset 2 colnames(EPAair_O3_19) head(EPAair_O3_19)
# summary(EPAair_O3_19) str(EPAair_O3_19) dim(EPAair_O3_19)

# Exploring dataset 3 colnames(EPAair_PM25_18) head(EPAair_PM25_18)
# summary(EPAair_PM25_18) str(EPAair_PM25_18) dim(EPAair_PM25_18)

# Exploring dataset 4 colnames(EPAair_PM25_19) head(EPAair_PM25_19)
# summary(EPAair_PM25_19) str(EPAair_PM25_19) dim(EPAair_PM25_19)
```

Wrangle individual datasets to create processed files.

3. Change date to date
4. Select the following columns: Date, DAILY_AQI_VALUE, Site.Name, AQS_PARAMETER_DESC, COUNTY, SITE_LATITUDE, SITE_LONGITUDE
5. For the PM2.5 datasets, fill all cells in AQS_PARAMETER_DESC with "PM2.5" (all cells in this column should be identical).
6. Save all four processed datasets in the Processed folder. Use the same file names as the raw files but replace "raw" with "processed".

```
# 3 changing date format
head(EPAair_O3_19$Date)
```

```
## [1] 01/01/2019 01/02/2019 01/03/2019 01/04/2019 01/05/2019 01/06/2019
## 365 Levels: 01/01/2019 01/02/2019 01/03/2019 01/04/2019 ... 12/31/2019
```

```
EPAair_O3_18$Date <- c(mdy(EPAair_O3_18$Date))
EPAair_O3_19$Date <- c(mdy(EPAair_O3_19$Date))
EPAair_PM25_18$Date <- c(mdy(EPAair_PM25_18$Date))
EPAair_PM25_19$Date <- c(mdy(EPAair_PM25_19$Date))



# 4 selecting columns
EPAair_O3_18 <- EPAair_O3_18[, c("Date", "DAILY_AQI_VALUE", "Site.Name", "AQS_PARAMETER_DESC",
    "COUNTY", "SITE_LATITUDE", "SITE_LONGITUDE")]

EPAair_O3_19 <- EPAair_O3_19[, c("Date", "DAILY_AQI_VALUE", "Site.Name", "AQS_PARAMETER_DESC",
    "COUNTY", "SITE_LATITUDE", "SITE_LONGITUDE")]

EPAair_PM25_18 <- EPAair_PM25_18[, c("Date", "DAILY_AQI_VALUE", "Site.Name", "AQS_PARAMETER_DESC",
    "COUNTY", "SITE_LATITUDE", "SITE_LONGITUDE")]

EPAair_PM25_19 <- EPAair_PM25_19[, c("Date", "DAILY_AQI_VALUE", "Site.Name", "AQS_PARAMETER_DESC",
    "COUNTY", "SITE_LATITUDE", "SITE_LONGITUDE")]

# 5 changing values for AQS_PARAMETER_DESC in PM datasets
EPAair_PM25_18 <- mutate(EPAair_PM25_18, AQS_PARAMETER_DESC = "PM2.5")

EPAair_PM25_19 <- mutate(EPAair_PM25_19, AQS_PARAMETER_DESC = "PM2.5")

# 6 Saving processed datasets
write.csv(EPAair_O3_18, row.names = FALSE, file = "C:/Users/user/Desktop/Soumya/Year 2/EDA/EDA-Fall2022/

write.csv(EPAair_O3_19, row.names = FALSE, file = "C:/Users/user/Desktop/Soumya/Year 2/EDA/EDA-Fall2022/

write.csv(EPAair_PM25_18, row.names = FALSE, file = "C:/Users/user/Desktop/Soumya/Year 2/EDA/EDA-Fall202

write.csv(EPAair_PM25_19, row.names = FALSE, file = "C:/Users/user/Desktop/Soumya/Year 2/EDA/EDA-Fall202
```

Combine datasets

7. Combine the four datasets with `rbind`. Make sure your column names are identical prior to running this code.

8. Wrangle your new dataset with a pipe function (%>%) so that it fills the following conditions:

- Include all sites that the four data frames have in common: "Linville Falls", "Durham Armory", "Leggett", "Hattie Avenue", "Clemmons Middle", "Mendenhall School", "Frying Pan Mountain", "West Johnston Co.", "Garinger High School", "Castle Hayne", "Pitt Agri. Center", "Bryson City", "Millbrook School" (the function `intersect` can figure out common factor levels)
- Some sites have multiple measurements per day. Use the split-apply-combine strategy to generate daily means: group by date, site, aqs parameter, and county. Take the mean of the AQI value, latitude, and longitude.
- Add columns for "Month" and "Year" by parsing your "Date" column (hint: `lubridate` package)
- Hint: the dimensions of this dataset should be 14,752 x 9.

9. Spread your datasets such that AQI values for ozone and PM2.5 are in separate columns. Each location on a specific date should now occupy only one row.

10. Call up the dimensions of your new tidy dataset.

11. Save your processed dataset with the following file name: "EPAair_O3_PM25_NC1718_Processed.csv"

```
# 7 loading processed datasets
EPAair_O3_18 <- read.csv("C:/Users/user/Desktop/Soumya/Year 2/EDA/EDA-Fall2022/Data/Processed/EPAair_O3_
    stringsAsFactors = TRUE)

EPAair_O3_19 <- read.csv("C:/Users/user/Desktop/Soumya/Year 2/EDA/EDA-Fall2022/Data/Processed/EPAair_O3_
    stringsAsFactors = TRUE)

EPAair_PM25_18 <- read.csv("C:/Users/user/Desktop/Soumya/Year 2/EDA/EDA-Fall2022/Data/Processed/EPAair_
    stringsAsFactors = TRUE)

EPAair_PM25_19 <- read.csv("C:/Users/user/Desktop/Soumya/Year 2/EDA/EDA-Fall2022/Data/Processed/EPAair_
```

```r
    stringsAsFactors = TRUE)

# Combining datasets
EPAair_Combined <- rbind(EPAair_O3_18, EPAair_O3_19, EPAair_PM25_18, EPAair_PM25_19)

# 8
EPAair_Combined$Date <- c(mdy(EPAair_Combined$Date))
```

```
## Warning: All formats failed to parse. No formats found.
```

```r
# Flitering datasets
EPAair_Combined_Filter <- EPAair_Combined %>%
    filter(Site.Name == "Linville Falls" | Site.Name == "Durham Armory" | Site.Name ==
        "Leggett" | Site.Name == "Hattie Avenue" | Site.Name == "Clemmons Middle" |
        Site.Name == "Mendenhall School" | Site.Name == "Frying Pan Mountain" | Site.Name ==
        "West Johnston Co." | Site.Name == "Garinger High School" | Site.Name ==
        "Castle Hayne" | Site.Name == "Pitt Agri. Center" | Site.Name == "Bryson City" |
        Site.Name == "Millbrook School") %>%
    group_by(Date, Site.Name, AQS_PARAMETER_DESC, COUNTY) %>%
    summarise_at(vars(DAILY_AQI_VALUE, SITE_LATITUDE, SITE_LONGITUDE), list(mean = mean)) %>%
    mutate(month = month(Date), year = year(Date))

# 9 spreading dataset
EPAair_Combined_Filter <- pivot_wider(EPAair_Combined_Filter, names_from = AQS_PARAMETER_DESC,
    values_from = DAILY_AQI_VALUE_mean)

# 10 checking dimension
dim(EPAair_Combined_Filter)
```

```
## [1] 13  9
```

```r
# 11 saving processed dataset
write.csv(EPAair_Combined_Filter, row.names = FALSE, file = "C:/Users/user/Desktop/Soumya/Year 2/EDA/EDA
```

Generate summary tables

12. Use the split-apply-combine strategy to generate a summary data frame. Data should be grouped by site, month, and year. Generate the mean AQI values for ozone and PM2.5 for each group. Then, add a pipe to remove instances where a month and year are not available (use the function `drop_na` in your pipe).

13. Call up the dimensions of the summary dataset.

```r
# 12a Creating summary table
EPAair_Combined_summary <- EPAair_Combined_Filter %>%
    group_by(Site.Name, month, year) %>%
    summarise_at(vars(PM2.5, Ozone), list(mean = mean))

# 12b Dropped NA Ozone and PM2.5 rows
EPAair_Combined_summary <- EPAair_Combined_Filter %>%
    drop_na(c(PM2.5, Ozone)) %>%
    group_by(Site.Name, month, year) %>%
    summarise_at(vars(PM2.5, Ozone), list(mean = mean))

# 13 checking dimension
dim(EPAair_Combined_summary)
```

```
## [1] 13  5
```

14. Why did we use the function `drop_na` rather than `na.omit`?

```
# checking 12b with na.omit function for comparison
EPAair_Combined_summary2 <- EPAair_Combined_Filter %>%
    na.omit(c(PM2.5, Ozone)) %>%
    group_by(Site.Name, month, year) %>%
    summarise_at(vars(PM2.5, Ozone), list(mean = mean))
```

> Answer: drop_na() only keeps the "complete" rows (where no rows contain missing values). na.omit() handle vectors, matrices and data frames comprising vectors and matrices (only). Using any of the functions gave same output.