Assignment 4: Data Wrangling (Fall 2024)

Chunyi Xu

OVERVIEW

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

Directions

1. Rename this file `<FirstLast>_A04_DataWrangling.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.
6. Ensure that code in code chunks does not extend off the page in the PDF.

Set up your session

1a. Load the `tidyverse`, `lubridate`, and `here` packages into your session.

1b. Check your working directory.

1c. Read in 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. Add the appropriate code to reveal the dimensions of the four datasets.

```
#1a
#Noting down codes to install packages
#install.packages("tidyverse");
#install.packages("lubridate");
#install.packages("here")

#Loading necessary packages (tidyverse, lubridate, here)
library (tidyverse)
library (lubridate)
library (here)
```

```
## Warning: package 'here' was built under R version 4.3.3
```

```
#1b
#Checking working directory is the project folder
here()
```

```
## [1] "/Users/xuchunyi/Desktop/EDA_Spring2025/EDA_Spring2025"
```

```r
#1c
#Reading in all four raw data files associated with the EPA Air dataset
EPAair_O3_NC2018 <- read.csv(
  file = here("./Data/Raw/EPAair_O3_NC2018_raw.csv"),
  stringsAsFactors = TRUE)

EPAair_O3_NC2019 <- read.csv(
  file = here("./Data/Raw/EPAair_O3_NC2019_raw.csv"),
  stringsAsFactors = TRUE)

EPAair_PM25_NC2018 <- read.csv(
  file = here("./Data/Raw/EPAair_PM25_NC2018_raw.csv"),
  stringsAsFactors = TRUE)

EPAair_PM25_NC2019 <- read.csv(
  file = here("./Data/Raw/EPAair_PM25_NC2019_raw.csv"),
  stringsAsFactors = TRUE)

#2
#Checking the dimensions of the four datasets

#Checking the dimensions of the dataset EPAair_O3_NC2018
# It has 9737 rows and 20 columns.
dim(EPAair_O3_NC2018)
```

```
## [1] 9737   20
```

```r
#Checking the dimensions of the dataset EPAair_O3_NC2019
# It has 10592 rows and 20 columns.
dim(EPAair_O3_NC2019)
```

```
## [1] 10592    20
```

```r
#Checking the dimensions of the dataset EPAair_PM25_NC2018
# It has 8983 rows and 20 columns.
dim(EPAair_PM25_NC2018)
```

```
## [1] 8983   20
```

```r
#Checking the dimensions of the dataset EPAair_PM25_NC2019
# It has 8581 rows and 20 columns.
dim(EPAair_PM25_NC2019)
```

```
## [1] 8581   20
```

> All four datasets should have the same number of columns but unique record counts (rows). Do your datasets follow this pattern? # Yes, all four datasets have the same number of columns but different number of rows.

Wrangle individual datasets to create processed files.

3. Change the Date columns to be date objects.
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 the Date column to be date objects for EPAair_O3_NC2018
EPAair_O3_NC2018$Date <- as.Date(EPAair_O3_NC2018$Date, format = "%m/%d/%Y")

#Changing the Date column to be date objects for EPAair_O3_NC2019
EPAair_O3_NC2019$Date <- as.Date(EPAair_O3_NC2019$Date, format = "%m/%d/%Y")

#Changing the Date column to be date objects for EPAair_PM25_NC2018
EPAair_PM25_NC2018$Date <- as.Date(EPAair_PM25_NC2018$Date, format = "%m/%d/%Y")

#Changing the Date column to be date objects for EPAair_PM25_NC2019
EPAair_PM25_NC2019$Date <- as.Date(EPAair_PM25_NC2019$Date, format = "%m/%d/%Y")

#4
#Selecting subset from EPAair_O3_NC2018
EPAair_O3_NC2018_sub1 <- select(EPAair_O3_NC2018, Date, DAILY_AQI_VALUE, Site.Name, AQS_PARAMETER_DESC,

#Selecting subset from EPAair_O3_NC2019
EPAair_O3_NC2019_sub1 <- select(EPAair_O3_NC2019, Date, DAILY_AQI_VALUE, Site.Name, AQS_PARAMETER_DESC,

#Selecting subset from EPAair_PM25_NC2018
EPAair_PM25_NC2018_sub1 <- select(EPAair_PM25_NC2018, Date, DAILY_AQI_VALUE, Site.Name, AQS_PARAMETER_D

#Selecting subset from EPAair_PM25_NC2019
EPAair_PM25_NC2019_sub1 <- select(EPAair_PM25_NC2019, Date, DAILY_AQI_VALUE, Site.Name, AQS_PARAMETER_D

#5
#Changing AQS_PARAMETER_DESC to PM2.5 for EPAair_PM25_NC2018_sub1
EPAair_PM25_NC2018_sub1 <- mutate(EPAair_PM25_NC2018_sub1, AQS_PARAMETER_DESC = "PM2.5")

#Changing AQS_PARAMETER_DESC to PM2.5 for EPAair_PM25_NC2019_sub1
EPAair_PM25_NC2019_sub1 <- mutate(EPAair_PM25_NC2019_sub1, AQS_PARAMETER_DESC = "PM2.5")

#6
#Saving processed dataset for EPAair_O3_NC2018_sub1
write.csv(EPAair_O3_NC2018_sub1, row.names = FALSE, file = "./Data/Processed/EPAair_O3_NC2018_processed

#Saving processed dataset for EPAair_O3_NC2019_sub1
write.csv(EPAair_O3_NC2019_sub1, row.names = FALSE, file = "./Data/Processed/EPAair_O3_NC2019_processed

#Saving processed dataset for EPAair_PM25_NC2018_sub1
write.csv(EPAair_PM25_NC2018_sub1, row.names = FALSE, file = "./Data/Processed/EPAair_PM25_NC2018_proces
```

```
#Saving processed dataset for EPAair_PM25_NC2019_sub1
write.csv(EPAair_PM25_NC2019_sub1, row.names = FALSE, file = "./Data/Processed/EPAair_PM25_NC2019_proce
```

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 only 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 - but it will include sites with missing site information, which you don't want...)

- Some sites have multiple measurements per day. Use the split-apply-combine strategy to generate daily means: group by date, site name, 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_NC1819_Processed.csv"

```
#7
#Combining the four datasets
EPAair_Combined <- rbind(EPAair_O3_NC2018_sub1, EPAair_O3_NC2019_sub1, EPAair_PM25_NC2018_sub1, EPAair_

#8
#Wrangling my processed dataset with a pipe function
EPAair_Combined_processed <-
  EPAair_Combined %>%
  filter(Site.Name %in% c("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" )) %>%
  group_by(Date, Site.Name, AQS_PARAMETER_DESC,COUNTY)%>%
  summarise(mean_AQI_value = mean(DAILY_AQI_VALUE),
            mean_latitude = mean(SITE_LATITUDE),
            mean_longitude = mean(SITE_LONGITUDE)) %>%
  mutate(Month = month(Date))%>%
  mutate(Year = year(Date))
```

```
## ‘summarise()‘ has grouped output by ’Date’, ’Site.Name’, ’AQS_PARAMETER_DESC’.
## You can override using the ‘.groups‘ argument.
```

```
#checking the dimension
#It has 14752 rows and 9 columns
dim(EPAair_Combined_processed)
```

```
## [1] 14752     9
```

```
#9
#Spreading the datasets to make AQI values for ozone and PM2.5 in separate columns
EPAair_Combined_Spread <-
  EPAair_Combined_processed %>%
  pivot_wider(names_from = AQS_PARAMETER_DESC,
              values_from = mean_AQI_value
              )

#10
#Checking the dimensions of the new tidy dataset
#It has 8976 rows and 9 columns
dim (EPAair_Combined_Spread)
```

```
## [1] 8976    9
```

```
#11
#Saving processed dataset for EPAair_Combined_Spread
write.csv(EPAair_Combined_Spread, row.names = FALSE, file = "./Data/Processed/EPAair_O3_PM25_NC1819_Pro
```

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 mean **ozone** values are not available (use the function `drop_na` in your pipe). It's ok to have missing mean PM2.5 values in this result.
13. Call up the dimensions of the summary dataset.

```
#12
#Generating a summary data frame by using a pipe function
EPA_Combined_Summary <-
   EPAair_Combined_Spread %>%
  group_by(Site.Name, Month,Year) %>%
  summarise(mean_AQI_Ozone = mean(Ozone),
            mean_AQI_PM2.5 = mean(PM2.5)) %>%
  drop_na (mean_AQI_Ozone)
```

```
## ‘summarise()‘ has grouped output by ’Site.Name’, ’Month’. You can override
## using the ‘.groups‘ argument.
```

```r
#13
#Checking the dimensions of the summary dataset
#It has 182 rows and 5 columns
dim (EPA_Combined_Summary)
```

```
## [1] 182   5
```

14. Why did we use the function `drop_na` rather than `na.omit`? Hint: replace `drop_na` with `na.omit` in part 12 and observe what happens with the dimensions of the summary date frame.

    Answer: By using "na.omit" rather than "drop_na", the dimensions of the summary data frame changed to 101 rows and 5 variables. This is mainly because the observations with "NA" in the mean_AQI_PM2.5 column were dropped too by using the "na.omit" function, even though we specified in the function "na.omit" that we only want it to perform on the "mean_AQI_Ozone" column. By using "na.omit", we cannot keep the observations with "NA" in the mean_AQI_PM2.5. Therefore, we want to use "drop.na" rather than "na.omit" function, as the former is a more precise and targeted command and yields expected results.