Assignment 4: Data Wrangling

Zhenghao Lin

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.

The completed exercise is due on Thursday, Sept 28th @ 5:00pm.

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. Apply the `glimpse()` function to reveal the dimensions, column names, and structure of each dataset.

```
#1a
#read library
library(`tidyverse`)
library(`lubridate`)
library(`here`)

#1b
#work directory
getwd()
```

```
## [1] "/Users/lzh/Desktop/EDE_Fall2023"
```

```r
#1c
#read dataset
EPAair_O3_NC2018 <-
  read.csv("/Users/lzh/Desktop/EDE_Fall2023/Data/Raw/EPAair_O3_NC2018_raw.csv",
           as.is = FALSE)

EPAair_PM25_NC2018 <-
  read.csv("/Users/lzh/Desktop/EDE_Fall2023/Data/Raw/EPAair_PM25_NC2018_raw.csv",
           as.is = FALSE)

EPAair_O3_NC2019 <-
  read.csv("/Users/lzh/Desktop/EDE_Fall2023/Data/Raw/EPAair_O3_NC2019_raw.csv",
           as.is = FALSE)

EPAair_PM25_NC2019 <-
  read.csv("/Users/lzh/Desktop/EDE_Fall2023/Data/Raw/EPAair_PM25_NC2019_raw.csv",
           as.is = FALSE)

#2
glimpse(EPAair_O3_NC2018$Date)
```

```
##  Factor w/ 364 levels "01/01/2018","01/02/2018",..: 60 61 62 63 64 65 66 67 68 69 ...
```

```r
glimpse(EPAair_PM25_NC2018$Date)
```

```
##  Factor w/ 365 levels "01/01/2018","01/02/2018",..: 2 5 8 11 14 17 20 23 26 29 ...
```

```r
glimpse(EPAair_O3_NC2019$Date)
```

```
##  Factor w/ 365 levels "01/01/2019","01/02/2019",..: 1 2 3 4 5 6 7 8 9 10 ...
```

```r
glimpse(EPAair_PM25_NC2019$Date)
```

```
##  Factor w/ 365 levels "01/01/2019","01/02/2019",..: 3 6 9 12 15 18 21 24 27 30 ...
```

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".

```r
#3
#date conversion for each dataset
EPAair_O3_NC2018$Date <- mdy(EPAair_O3_NC2018$Date)

EPAair_PM25_NC2018$Date <- mdy(EPAair_PM25_NC2018$Date)

EPAair_O3_NC2019$Date <- mdy(EPAair_O3_NC2019$Date)

EPAair_PM25_NC2019$Date <- mdy(EPAair_PM25_NC2019$Date)

#4
#select required columns
EPAair_O3_NC2018_Processed <- EPAair_O3_NC2018 %>%
  select(Date, DAILY_AQI_VALUE, Site.Name, AQS_PARAMETER_DESC, COUNTY,
         SITE_LATITUDE, SITE_LONGITUDE)

EPAair_PM25_NC2018_Processed <- EPAair_PM25_NC2018 %>%
  select(Date, DAILY_AQI_VALUE, Site.Name, AQS_PARAMETER_DESC, COUNTY,
         SITE_LATITUDE, SITE_LONGITUDE)

EPAair_O3_NC2019_Processed <- EPAair_O3_NC2019 %>%
  select(Date, DAILY_AQI_VALUE, Site.Name, AQS_PARAMETER_DESC, COUNTY,
         SITE_LATITUDE, SITE_LONGITUDE)

EPAair_PM25_NC2019_Processed <- EPAair_PM25_NC2019 %>%
  select(Date, DAILY_AQI_VALUE, Site.Name, AQS_PARAMETER_DESC, COUNTY,
         SITE_LATITUDE, SITE_LONGITUDE)

#5
#change all value under AQS_PARAMETER_DESC column as PM2.5
#for EPAair_PM25_NC2018_Processed & EPAair_PM25_NC2019_Processed
EPAair_PM25_NC2018_Processed <-  EPAair_PM25_NC2018_Processed %>%
  mutate(AQS_PARAMETER_DESC = "PM2.5")

EPAair_PM25_NC2019_Processed <-  EPAair_PM25_NC2019_Processed %>%
  mutate(AQS_PARAMETER_DESC = "PM2.5")

#6
#save datasets into the processed file
write.csv(EPAair_O3_NC2018_Processed, row.names = FALSE,
          file = "/Users/lzh/Desktop/EDE_Fall2023/Data/Processed/EPAair_O3_NC2018_Processed.csv")

write.csv(EPAair_PM25_NC2018_Processed, row.names = FALSE,
          file = "/Users/lzh/Desktop/EDE_Fall2023/Data/Processed/EPAair_PM25_NC2018_Processed.csv")

write.csv(EPAair_O3_NC2019_Processed, row.names = FALSE,
          file = "/Users/lzh/Desktop/EDE_Fall2023/Data/Processed/EPAair_O3_NC2019_Processed.csv")

write.csv(EPAair_PM25_NC2019_Processed, row.names = FALSE,
          file = "/Users/lzh/Desktop/EDE_Fall2023/Data/Processed/EPAair_PM25_NC2019_Processed.csv")
```

Combine datasets

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

```
#7
combined_df <- rbind(EPAair_O3_NC2018_Processed,EPAair_PM25_NC2018_Processed
                     ,EPAair_O3_NC2019_Processed, EPAair_PM25_NC2019_Processed)
```

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"

```
#8
#sites that the four data frames have in common
common_sites <- 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"
)

combined_df_sites <- combined_df %>%
  filter(Site.Name %in% common_sites)

#group by date, site name, AQS parameter, and county.
#Take the mean of the AQI value, latitude, and longitude
combined_df_mean <- combined_df_sites %>%
  group_by(Date, Site.Name, AQS_PARAMETER_DESC, COUNTY) %>%
  summarize(
    Mean_AQI = mean(DAILY_AQI_VALUE, na.rm = FALSE),
    Mean_Latitude = mean(SITE_LATITUDE, na.rm = FALSE),
    Mean_Longitude = mean(SITE_LONGITUDE, na.rm = FALSE)
    )
```

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

```
#Add columns for "Month" and "Year"
combined_df_date <- combined_df_mean %>%
  mutate(
    Month = month(Date),
    Year = year(Date)
  )

#dimension of the dataset combined_df_date
dim(combined_df_date)
```

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

```
#9
#Spread the datasets so that AQI values for ozone and PM2.5 are in separate columns
combined_df_Spread <- pivot_wider(combined_df_date,
    names_from = AQS_PARAMETER_DESC,
    values_from = Mean_AQI
  )

#rename the two new columns
combined_df_Spread <- combined_df_Spread %>%
  rename(Mean_AQI_PM2.5 = PM2.5, Mean_AQI_O3 = Ozone )
#10
#dimension of the dataset combined_df_Spread
dim(combined_df_Spread)
```

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

```
#11
#save the dataset into the processed file
write.csv(combined_df_Spread, row.names = FALSE,
          file = "/Users/lzh/Desktop/EDE_Fall2023/Data/Processed/EPAair_O3_PM25_NC1819_Processed.csv")
```

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
#Summary Data Frame
summary <- combined_df_Spread %>%
  group_by(Site.Name, Month, Year) %>%
  summarize(Mean_Ozone = mean(Mean_AQI_O3, na.rm = TRUE),
            Mean_PM25 = mean(Mean_AQI_PM2.5, na.rm = FALSE)) %>%
  drop_na(Mean_Ozone)
```

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

```
#13
#dimention of summary data frame
dim(summary)
```

```
## [1] 239   5
```

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

    Answer: drop_na(data, …) drops rows where any column specified by … contains a missing value. na.omit returns the object with incomplete cases removed. drop_na usually works with dataframe and na.omit works with objects. In this case, since we are working with data frame, we should use drop_na instead of na.omit.