Assignment 4: Data Wrangling

Queenie Wei

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
#install.packages("here")
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.3.1

## Warning: package 'lubridate' was built under R version 4.3.1
```

```
library(lubridate)
library(dplyr)
library(here)
```

```
## Warning: package 'here' was built under R version 4.3.1
```

```r
library(tidyr)

#1b
here()
```

```
## [1] "C:/Users/ziyaw/Downloads/EDE_Fall2023"
```

```r
getwd()
```

```
## [1] "C:/Users/ziyaw/Downloads/EDE_Fall2023"
```

```r
#1c
##reading in the documents
epa.2018o3 <- read.csv(here("Data/Raw/EPAair_O3_NC2018_raw.csv"),
                       stringsAsFactors = TRUE)
epa.2019o3 <- read.csv(here("Data/Raw/EPAair_O3_NC2019_raw.csv"),
                       stringsAsFactors = TRUE)
epa.2018pm2.5 <- read.csv(here("Data/Raw/EPAair_PM25_NC2018_raw.csv"),
                          stringsAsFactors = TRUE)
epa.2019pm2.5 <- read.csv(here("Data/Raw/EPAair_PM25_NC2019_raw.csv"),
                          stringsAsFactors = TRUE)

#2
##revealing the dimensions, column names, and structure of each dataset
glimpse(epa.2018o3)
```

```
## Rows: 9,737
## Columns: 20
## $ Date                                 <fct> 03/01/2018, 03/02/2018, 03/03/201~
## $ Source                               <fct> AQS, AQS, AQS, AQS, AQS, AQS, AQS~
## $ Site.ID                              <int> 370030005, 370030005, 370030005, ~
## $ POC                                  <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ Daily.Max.8.hour.Ozone.Concentration <dbl> 0.043, 0.046, 0.047, 0.049, 0.047~
## $ UNITS                                <fct> ppm, ppm, ppm, ppm, ppm, ppm, ppm~
## $ DAILY_AQI_VALUE                      <int> 40, 43, 44, 45, 44, 28, 33, 41, 4~
## $ Site.Name                            <fct> Taylorsville Liledoun, Taylorsvil~
## $ DAILY_OBS_COUNT                      <int> 17, 17, 17, 17, 17, 17, 17, 17, 1~
## $ PERCENT_COMPLETE                     <dbl> 100, 100, 100, 100, 100, 100, 100~
## $ AQS_PARAMETER_CODE                   <int> 44201, 44201, 44201, 44201, 44201~
## $ AQS_PARAMETER_DESC                   <fct> Ozone, Ozone, Ozone, Ozone, Ozone~
## $ CBSA_CODE                            <int> 25860, 25860, 25860, 25860, 25860~
## $ CBSA_NAME                            <fct> "Hickory-Lenoir-Morganton, NC", "~
## $ STATE_CODE                           <int> 37, 37, 37, 37, 37, 37, 37, 37, 3~
## $ STATE                                <fct> North Carolina, North Carolina, N~
## $ COUNTY_CODE                          <int> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, ~
## $ COUNTY                               <fct> Alexander, Alexander, Alexander, ~
## $ SITE_LATITUDE                        <dbl> 35.9138, 35.9138, 35.9138, 35.913~
## $ SITE_LONGITUDE                       <dbl> -81.191, -81.191, -81.191, -81.19~
```

```r
glimpse(epa.2019o3)
```

```
## Rows: 10,592
## Columns: 20
## $ Date                                 <fct> 01/01/2019, 01/02/2019, 01/03/201~
## $ Source                               <fct> AirNow, AirNow, AirNow, AirNow, A~
## $ Site.ID                              <int> 370030005, 370030005, 370030005, ~
## $ POC                                  <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ Daily.Max.8.hour.Ozone.Concentration <dbl> 0.029, 0.018, 0.016, 0.022, 0.037~
## $ UNITS                                <fct> ppm, ppm, ppm, ppm, ppm, ppm, ppm~
## $ DAILY_AQI_VALUE                      <int> 27, 17, 15, 20, 34, 34, 27, 35, 3~
## $ Site.Name                            <fct> Taylorsville Liledoun, Taylorsvil~
## $ DAILY_OBS_COUNT                      <int> 24, 24, 24, 24, 24, 24, 24, 24, 2~
## $ PERCENT_COMPLETE                     <dbl> 100, 100, 100, 100, 100, 100, 100~
## $ AQS_PARAMETER_CODE                   <int> 44201, 44201, 44201, 44201, 44201~
## $ AQS_PARAMETER_DESC                   <fct> Ozone, Ozone, Ozone, Ozone, Ozone~
## $ CBSA_CODE                            <int> 25860, 25860, 25860, 25860, 25860~
## $ CBSA_NAME                            <fct> "Hickory-Lenoir-Morganton, NC", "~
## $ STATE_CODE                           <int> 37, 37, 37, 37, 37, 37, 37, 37, 3~
## $ STATE                                <fct> North Carolina, North Carolina, N~
## $ COUNTY_CODE                          <int> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, ~
## $ COUNTY                               <fct> Alexander, Alexander, Alexander, ~
## $ SITE_LATITUDE                        <dbl> 35.9138, 35.9138, 35.9138, 35.913~
## $ SITE_LONGITUDE                       <dbl> -81.191, -81.191, -81.191, -81.19~
```

```
glimpse(epa.2018pm2.5)
```

```
## Rows: 8,983
## Columns: 20
## $ Date                           <fct> 01/02/2018, 01/05/2018, 01/08/2018, 01/~
## $ Source                         <fct> AQS, AQS, AQS, AQS, AQS, AQS, AQS, AQS,~
## $ Site.ID                        <int> 370110002, 370110002, 370110002, 370110~
## $ POC                            <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ Daily.Mean.PM2.5.Concentration <dbl> 2.9, 3.7, 5.3, 0.8, 2.5, 4.5, 1.8, 2.5,~
## $ UNITS                          <fct> ug/m3 LC, ug/m3 LC, ug/m3 LC, ug/m3 LC,~
## $ DAILY_AQI_VALUE                <int> 12, 15, 22, 3, 10, 19, 8, 10, 18, 7, 24~
## $ Site.Name                      <fct> Linville Falls, Linville Falls, Linvill~
## $ DAILY_OBS_COUNT                <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ PERCENT_COMPLETE               <dbl> 100, 100, 100, 100, 100, 100, 100, 100,~
## $ AQS_PARAMETER_CODE             <int> 88502, 88502, 88502, 88502, 88502, 8850~
## $ AQS_PARAMETER_DESC             <fct> Acceptable PM2.5 AQI & Speciation Mass,~
## $ CBSA_CODE                      <int> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ CBSA_NAME                      <fct> "", "", "", "", "", "", "", "", "", "",~
## $ STATE_CODE                     <int> 37, 37, 37, 37, 37, 37, 37, 37, 37, 37,~
## $ STATE                          <fct> North Carolina, North Carolina, North C~
## $ COUNTY_CODE                    <int> 11, 11, 11, 11, 11, 11, 11, 11, 11, 11,~
## $ COUNTY                         <fct> Avery, Avery, Avery, Avery, Avery, Aver~
## $ SITE_LATITUDE                  <dbl> 35.97235, 35.97235, 35.97235, 35.97235,~
## $ SITE_LONGITUDE                 <dbl> -81.93307, -81.93307, -81.93307, -81.93~
```

```
glimpse(epa.2019pm2.5)
```

```
## Rows: 8,581
## Columns: 20
## $ Date                           <fct> 01/03/2019, 01/06/2019, 01/09/2019, 01/~
## $ Source                       <fct> AQS, AQS, AQS, AQS, AQS, AQS, AQS, AQS,~
## $ Site.ID                      <int> 370110002, 370110002, 370110002, 370110~
## $ POC                          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ Daily.Mean.PM2.5.Concentration <dbl> 1.6, 1.0, 1.3, 6.3, 2.6, 1.2, 1.5, 1.5,~
## $ UNITS                        <fct> ug/m3 LC, ug/m3 LC, ug/m3 LC, ug/m3 LC,~
## $ DAILY_AQI_VALUE              <int> 7, 4, 5, 26, 11, 5, 6, 6, 15, 7, 14, 20~
## $ Site.Name                    <fct> Linville Falls, Linville Falls, Linvill~
## $ DAILY_OBS_COUNT              <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ PERCENT_COMPLETE             <dbl> 100, 100, 100, 100, 100, 100, 100, 100,~
## $ AQS_PARAMETER_CODE           <int> 88502, 88502, 88502, 88502, 88502, 8850~
## $ AQS_PARAMETER_DESC           <fct> Acceptable PM2.5 AQI & Speciation Mass,~
## $ CBSA_CODE                    <int> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ CBSA_NAME                    <fct> "", "", "", "", "", "", "", "", "", "",~
## $ STATE_CODE                   <int> 37, 37, 37, 37, 37, 37, 37, 37, 37, 37,~
## $ STATE                        <fct> North Carolina, North Carolina, North C~
## $ COUNTY_CODE                  <int> 11, 11, 11, 11, 11, 11, 11, 11, 11, 11,~
## $ COUNTY                       <fct> Avery, Avery, Avery, Avery, Avery, Aver~
## $ SITE_LATITUDE                <dbl> 35.97235, 35.97235, 35.97235, 35.97235,~
## $ SITE_LONGITUDE               <dbl> -81.93307, -81.93307, -81.93307, -81.93~
```

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
##changing format of the Date column to date objects
epa.2018o3$Date <- as.Date(epa.2018o3$Date, format = "%m/%d/%Y")
epa.2019o3$Date <- as.Date(epa.2019o3$Date, format = "%m/%d/%Y")
epa.2019pm2.5$Date <- as.Date(epa.2019pm2.5$Date, format = "%m/%d/%Y")
epa.2018pm2.5$Date <- as.Date(epa.2018pm2.5$Date, format = "%m/%d/%Y")

#4
#filtering so that the new dataframe only has 7 columns that we need
selected.epa.2018o3 <- select(epa.2018o3,Date, DAILY_AQI_VALUE, Site.Name,
                              AQS_PARAMETER_DESC, COUNTY, SITE_LATITUDE,
                              SITE_LONGITUDE)
selected.epa.2019o3 <- select(epa.2019o3,Date, DAILY_AQI_VALUE, Site.Name,
                              AQS_PARAMETER_DESC, COUNTY, SITE_LATITUDE,
                              SITE_LONGITUDE)
selected.epa.2018pm2.5 <- select(epa.2018pm2.5,Date, DAILY_AQI_VALUE, Site.Name,
                              AQS_PARAMETER_DESC, COUNTY, SITE_LATITUDE,
                              SITE_LONGITUDE)
selected.epa.2019pm2.5 <- select(epa.2019pm2.5,Date, DAILY_AQI_VALUE, Site.Name,
                              AQS_PARAMETER_DESC, COUNTY, SITE_LATITUDE,
                              SITE_LONGITUDE)
```

```
#5
selected.epa.2018pm2.5 <- mutate(selected.epa.2018pm2.5,
                                 AQS_PARAMETER_DESC = "PM2.5")
selected.epa.2019pm2.5 <- mutate(selected.epa.2019pm2.5,
                                 AQS_PARAMETER_DESC = "PM2.5")

#6
##saving the different documents to the processed folder
write.csv(selected.epa.2018o3, row.names = FALSE, file =
            here("Data/Processed/EPAair_O3_NC2018_processed.csv"))
write.csv(selected.epa.2019o3, row.names = FALSE, file =
            here("Data/Processed/EPAair_O3_NC2019_processed.csv"))
write.csv(selected.epa.2018pm2.5, row.names = FALSE, file =
            here("Data/Processed/EPAair_PM25_NC2018_processed.csv"))
write.csv(selected.epa.2019pm2.5, row.names = FALSE, file =
            here("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.

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
modified.epa.2018o3 <- read.csv(here("Data/Processed/EPAair_O3_NC2018_processed.csv"), stringsAsFactors
modified.epa.2019o3 <- read.csv(here("Data/Processed/EPAair_O3_NC2019_processed.csv"), stringsAsFactors
modified.epa.2018pm2.5 <- read.csv(here("Data/Processed/EPAair_PM25_NC2018_processed.csv"), stringsAsFa
modified.epa.2019pm2.5 <- read.csv(here("Data/Processed/EPAair_PM25_NC2019_processed.csv"), stringsAsFa
##checking if the colnames are the same
colnames(modified.epa.2018o3) == colnames(modified.epa.2019o3)
```

```
## [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

```
colnames(modified.epa.2018pm2.5)== colnames(modified.epa.2019pm2.5)
```

```
## [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

```
colnames(modified.epa.2018pm2.5)== colnames(modified.epa.2019o3)
```

```
## [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

```
#?rbind
combined_201819_o3pm2.5 <- rbind(modified.epa.2018o3, modified.epa.2019o3,
                                 modified.epa.2018pm2.5, modified.epa.2019pm2.5)

#8
###filter out any sites that are not "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"
###Using 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



new_combined_201819_o3pm2.5 <-
  combined_201819_o3pm2.5 %>%
  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(meanAQI = mean(DAILY_AQI_VALUE),
             meanlat = mean(SITE_LATITUDE),
            meanlong = mean(SITE_LONGITUDE),
            .groups = 'drop') %>%
  mutate(Month = month(Date), Year = year(Date))
```

```
## Warning: There were 2 warnings in ‘mutate()‘.
## The first warning was:
## i In argument: ‘Month = month(Date)‘.
## Caused by warning:
## ! tz(): Don’t know how to compute timezone for object of class factor; returning "UTC".
## i Run ‘dplyr::last_dplyr_warnings()‘ to see the 1 remaining warning.
```

```
dim(new_combined_201819_o3pm2.5)
```

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

```
#9
# Spreading the datasets such that AQI values for ozone and
# PM2.5 are in separate columns using pivot_wider
new_combined_201819_o3pm2.5.spread2 <- pivot_wider(new_combined_201819_o3pm2.5,
                        names_from = AQS_PARAMETER_DESC, values_from = meanAQI)

duplicate_rows <- new_combined_201819_o3pm2.5.spread2[duplicated
                (new_combined_201819_o3pm2.5.spread2[c('Date', 'Site.Name')]) |
        duplicated(new_combined_201819_o3pm2.5.spread2[c('Date', 'Site.Name')],
         fromLast = TRUE), ]
dim(duplicate_rows)
```

```
## [1] 0 9
```

```
######works cited: chatGPT
######prompt: use the unique function to see if there are rows with the same
######"Date and Site.Name" values in r
###### the prompt is wrong but chatgpt was able to recognize that and use
####### another method
#10
#### checking the dimension of the new dataset
dim(new_combined_201819_o3pm2.5.spread2)
```

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

```
#11 ###savign the new dataset
write.csv(new_combined_201819_o3pm2.5.spread2, row.names = FALSE, file =
            here("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
##this code groups teh data by site, month, and year,
##generates the mean aqi values for ozone and pm2.5 respectively,
## and removes where the mean ozone values are not available
new_combined_201819_o3pm2.5.spread2_summary <-
  new_combined_201819_o3pm2.5.spread2 %>%
  group_by(Site.Name, Month, Year) %>%
  summarise(meanAQI_Ozone = mean(Ozone),
            meanAQI_pm2.5 = mean (PM2.5),
            .groups = 'drop') %>%
  drop_na(meanAQI_Ozone)
```

```
#colnames(new_combined_201819_o3pm2.5.spread2)
# colnames(new_combined_201819_o3pm2.5.spread2_summary)



#13
##dimensions of the new dataframe
dim(new_combined_201819_o3pm2.5.spread2_summary)
```

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

```
##saves the new dataframe
write.csv(new_combined_201819_o3pm2.5.spread2_summary, row.names = FALSE, file =
            here("Data/Processed/EPAair_O3_PM25_NC1819_Summary_Processed.csv"))
```

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

    Answer: The two are very similar with different syntaxes. The syntax for na.omit is a lot more complicated because it automatically deletes all instances with nas in a dataframe. As a result, there needs to be a lot of workaround in order for it to work. na.omit also doesnt take in an argument for a specific row. Additionally, drop_na is also in the dplyr package, which would be easier to follow.