Assignment 4: Data Wrangling

Chengxi Li

OVERVIEW

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

Directions

1. Change "Student Name" on line 3 (above) with your name.
2. Work through the steps, **creating code and output** that fulfill each instruction.
3. Be sure to **answer the questions** in this assignment document.
4. When you have completed the assignment, **Knit** the text and code into a single PDF file.
5. After Knitting, submit the completed exercise (PDF file) to the dropbox in Sakai. Add your last name into the file name (e.g., "Fay_A04_DataWrangling.Rmd") prior to submission.

The completed exercise is due on Tuesday, Feb 16 @ 11:59pm.

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. See the README file for the EPA air datasets for more information (especially if you have not worked with air quality data previously).

```
getwd()
```

```
## [1] "C:/Users/li_ch/Desktop/DKU/Year 2/Term 2/Environmental Data Analytics/GIT Hub/Environmental_Data
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.5     v dplyr   1.0.4
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
O3_2018 <- read.csv("./Data/Raw/EPAair_O3_NC2018_raw.csv", stringsAsFactors = TRUE)
O3_2019 <- read.csv("./Data/Raw/EPAair_O3_NC2019_raw.csv", stringsAsFactors = TRUE)
```

```
PM25_2018 <- read.csv("./Data/Raw/EPAair_PM25_NC2018_raw.csv", stringsAsFactors = TRUE)
PM25_2019 <- read.csv("./Data/Raw/EPAair_PM25_NC2019_raw.csv", stringsAsFactors = TRUE)
```

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

```
#1 dimensions of the datasets
dim(PM25_2019)
```

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

```
dim(PM25_2018)
```

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

```
dim(O3_2019)
```

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

```
dim(O3_2018)
```

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

```
#2 column names of the datasets
colnames(PM25_2019)
```

```
##  [1] "Date"                           "Source"
##  [3] "Site.ID"                        "POC"
##  [5] "Daily.Mean.PM2.5.Concentration" "UNITS"
##  [7] "DAILY_AQI_VALUE"                "Site.Name"
##  [9] "DAILY_OBS_COUNT"                "PERCENT_COMPLETE"
## [11] "AQS_PARAMETER_CODE"             "AQS_PARAMETER_DESC"
## [13] "CBSA_CODE"                      "CBSA_NAME"
## [15] "STATE_CODE"                     "STATE"
## [17] "COUNTY_CODE"                    "COUNTY"
## [19] "SITE_LATITUDE"                  "SITE_LONGITUDE"
```

```
colnames(PM25_2018)
```

```
##  [1] "Date"                           "Source"
##  [3] "Site.ID"                        "POC"
##  [5] "Daily.Mean.PM2.5.Concentration" "UNITS"
##  [7] "DAILY_AQI_VALUE"                "Site.Name"
##  [9] "DAILY_OBS_COUNT"                "PERCENT_COMPLETE"
## [11] "AQS_PARAMETER_CODE"             "AQS_PARAMETER_DESC"
## [13] "CBSA_CODE"                      "CBSA_NAME"
## [15] "STATE_CODE"                     "STATE"
## [17] "COUNTY_CODE"                    "COUNTY"
## [19] "SITE_LATITUDE"                  "SITE_LONGITUDE"
```

```
colnames(O3_2019)
```

```
##  [1] "Date"
##  [2] "Source"
##  [3] "Site.ID"
##  [4] "POC"
##  [5] "Daily.Max.8.hour.Ozone.Concentration"
##  [6] "UNITS"
##  [7] "DAILY_AQI_VALUE"
##  [8] "Site.Name"
```

```
##  [9] "DAILY_OBS_COUNT"
## [10] "PERCENT_COMPLETE"
## [11] "AQS_PARAMETER_CODE"
## [12] "AQS_PARAMETER_DESC"
## [13] "CBSA_CODE"
## [14] "CBSA_NAME"
## [15] "STATE_CODE"
## [16] "STATE"
## [17] "COUNTY_CODE"
## [18] "COUNTY"
## [19] "SITE_LATITUDE"
## [20] "SITE_LONGITUDE"
```

```
colnames(O3_2018)
```

```
##  [1] "Date"
##  [2] "Source"
##  [3] "Site.ID"
##  [4] "POC"
##  [5] "Daily.Max.8.hour.Ozone.Concentration"
##  [6] "UNITS"
##  [7] "DAILY_AQI_VALUE"
##  [8] "Site.Name"
##  [9] "DAILY_OBS_COUNT"
## [10] "PERCENT_COMPLETE"
## [11] "AQS_PARAMETER_CODE"
## [12] "AQS_PARAMETER_DESC"
## [13] "CBSA_CODE"
## [14] "CBSA_NAME"
## [15] "STATE_CODE"
## [16] "STATE"
## [17] "COUNTY_CODE"
## [18] "COUNTY"
## [19] "SITE_LATITUDE"
## [20] "SITE_LONGITUDE"
```

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
O3_2018$Date <- as.Date(O3_2018$Date, format = "%m/%d/%Y")
O3_2019$Date <- as.Date(O3_2019$Date, format = "%m/%d/%Y")
PM25_2018$Date <- as.Date(PM25_2018$Date, format = "%m/%d/%Y")
PM25_2019$Date <- as.Date(PM25_2019$Date, format = "%m/%d/%Y")

#4
O3_2018 <- select(O3_2018,Date, DAILY_AQI_VALUE, Site.Name, AQS_PARAMETER_DESC, COUNTY, SITE_LATITUDE, S
O3_2019 <- select(O3_2019,Date, DAILY_AQI_VALUE, Site.Name, AQS_PARAMETER_DESC, COUNTY, SITE_LATITUDE, S
```

```
PM25_2018 <- select(PM25_2018,Date, DAILY_AQI_VALUE, Site.Name, AQS_PARAMETER_DESC, COUNTY, SITE_LATITU
PM25_2019 <- select(PM25_2019,Date, DAILY_AQI_VALUE, Site.Name, AQS_PARAMETER_DESC, COUNTY, SITE_LATITU

#5
PM25_2018$AQS_PARAMETER_DESC <-("PM2.5")
PM25_2019$AQS_PARAMETER_DESC <-("PM2.5")

#6
write.csv(O3_2018, row.names = FALSE, file = "./Data/Processed/EPAair_O3_NC2018_processed.csv")
write.csv(O3_2019, row.names = FALSE, file = "./Data/Processed/EPAair_O3_NC2019_processed.csv")
write.csv(PM25_2018, row.names = FALSE, file = "./Data/Processed/EPAair_PM25_NC2018_processed.csv")
write.csv(PM25_2019, row.names = FALSE, file = "./Data/Processed/EPAair_PM25_NC201_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 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
O3_PM25_NC1819 <- rbind(PM25_2019,PM25_2018,O3_2019,O3_2018)

#8
O3_PM25_NC1819 <-
  O3_PM25_NC1819 %>%
  filter(Site.Name == "Linville Falls" | Site.Name == "Durham Armory" | Site.Name == "Leggett" |Site.Na
  group_by(Date, Site.Name, AQS_PARAMETER_DESC, COUNTY) %>%
  summarise(DAILY_AQI_VALUE = mean(DAILY_AQI_VALUE),
            SITE_Longitude = mean(SITE_LONGITUDE),
            SITE_Latitude = mean(SITE_LATITUDE)) %>%
  mutate(Month = month (Date), Year = year (Date))
```

```
## `summarise()` has grouped output by 'Date', 'Site.Name', 'AQS_PARAMETER_DESC'. You can override using
```

```
#9
O3_PM25_NC1819 <- pivot_wider(O3_PM25_NC1819, names_from = AQS_PARAMETER_DESC, values_from = DAILY_AQI_

#10
dim(O3_PM25_NC1819)
```

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

```
#11
write.csv(O3_PM25_NC1819, row.names = FALSE, file = "./Data/Processed/EPAair_O3_PM25_NC1718_Processed.cs
```

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.

```
#12a
Summary <-
  O3_PM25_NC1819 %>%
  group_by(Site.Name, Month, Year) %>%
  summarise(Mean_O3 = mean(Ozone),
            Mean_PM2.5 = mean(PM2.5))
```

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

```
#12b
Summary <-
  Summary %>%
  drop_na(Month) %>%
  drop_na(Year)

#Summary <-
  #Summary %>%
  #na.omit(Month) %>%
  #na.omit(Year)


#13
dim(Summary)
```

```
## [1] 308   5
```

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

    Answer: when applying "drop_na" to month and year, only the rows with missing month and year will be droped. However, if we use "na.omit", any row with missing value (na) will be removed. We use "drop_na" becasue we want to remove the rows without timestamp, but not those missing one AQI measurement.