7: Data Wrangling

Environmental Data Analytics | Kateri Salk

Spring 2019

LESSON OBJECTIVES

1. Describe the usefulness of data wrangling and its place in the data pipeline
2. Wrangle datasets with dplyr functions
3. Apply data wrangling skills to a real-world example dataset

OPENING DISCUSSION

After we've completed basic data exploration on a dataset, what step comes next? How does this help us to ask and answer questions about datasets?

SET UP YOUR DATA ANALYSIS SESSION

In assignment 3, you explored the North Temperate Lakes Long-Term Ecological Research Station data for physical and chemical data. What did you learn about this dataset in your assignment?

We will continue working with this dataset today.

```
getwd()
```

```
## [1] "C:/Users/Felipe/OneDrive - Duke University/1. DUKE/1. Ramos 2 Semestre/EOS-872 Env. Data Analyti
```

```
library(tidyverse)
```

```
## -- Attaching packages ----------------------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

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

```
NTL.phys.data <- read.csv("./Data/Raw/NTL-LTER_Lake_ChemistryPhysics_Raw.csv")

head(NTL.phys.data)
```

```
##   lakeid  lakename year4 daynum sampledate depth temperature_C
## 1      L Paul Lake  1984    148    5/27/84  0.00          14.5
## 2      L Paul Lake  1984    148    5/27/84  0.25            NA
## 3      L Paul Lake  1984    148    5/27/84  0.50            NA
## 4      L Paul Lake  1984    148    5/27/84  0.75            NA
## 5      L Paul Lake  1984    148    5/27/84  1.00          14.5
## 6      L Paul Lake  1984    148    5/27/84  1.50            NA
##   dissolvedOxygen irradianceWater irradianceDeck comments
## 1             9.5            1750           1620     <NA>
## 2              NA            1550           1620     <NA>
```

```
## 3              NA            1150           1620     <NA>
## 4              NA             975           1620     <NA>
## 5             8.8             870           1620     <NA>
## 6              NA             610           1620     <NA>
```

```
colnames(NTL.phys.data)
```

```
##  [1] "lakeid"          "lakename"        "year4"
##  [4] "daynum"          "sampledate"      "depth"
##  [7] "temperature_C"   "dissolvedOxygen" "irradianceWater"
## [10] "irradianceDeck"  "comments"
```

```
summary(NTL.phys.data)
```

```
##      lakeid                lakename         year4          daynum
##  R      :11288   Peter Lake    :11288   Min.   :1984   Min.   : 55.0
##  L      :10325   Paul Lake     :10325   1st Qu.:1991   1st Qu.:166.0
##  T      : 6107   Tuesday Lake  : 6107   Median :1997   Median :194.0
##  W      : 4188   West Long Lake: 4188   Mean   :1999   Mean   :194.3
##  E      : 3905   East Long Lake: 3905   3rd Qu.:2006   3rd Qu.:222.0
##  M      : 1234   Crampton Lake : 1234   Max.   :2016   Max.   :307.0
##  (Other): 1567   (Other)       : 1567
##    sampledate        depth       temperature_C   dissolvedOxygen
##  5/17/94:   84   Min.   : 0.00   Min.   : 0.30   Min.   :  0.00
##  9/5/90 :   64   1st Qu.: 1.50   1st Qu.: 5.30   1st Qu.:  0.30
##  10/1/07:   61   Median : 4.00   Median : 9.30   Median :  5.60
##  9/10/90:   61   Mean   : 4.39   Mean   :11.81   Mean   :  4.97
##  5/10/87:   60   3rd Qu.: 6.50   3rd Qu.:18.70   3rd Qu.:  8.40
##  5/9/88 :   60   Max.   :20.00   Max.   :34.10   Max.   :802.00
##  (Other):38224                   NA's   :3858    NA's   :4039
##  irradianceWater     irradianceDeck
##  Min.   :   -0.337   Min.   :   1.5
##  1st Qu.:   14.000   1st Qu.: 353.0
##  Median :   65.000   Median : 747.0
##  Mean   :  210.242   Mean   : 720.5
##  3rd Qu.:  265.000   3rd Qu.:1042.0
##  Max.   :24108.000   Max.   :8532.0
##  NA's   :14287       NA's   :15419
##                              comments
##  DO Probe bad - Doesn't go to zero:  206
##  DO taken with Jones Lab Meter    :  162
##  NA's                             :38246
##
##
##
##
```

```
dim(NTL.phys.data)
```

```
## [1] 38614    11
```

DATA WRANGLING

Data wrangling takes data exploration one step further: it allows you to process data in ways that are useful for you. An important part of data wrangling is creating tidy datasets, with the following rules:

1. Each variable has its own column
2. Each observation has its own row
3. Each value has its own cell

What is the best way to wrangle data? There are multiple ways to arrive at a specific outcome in R, and we will illustrate some of those approaches. Your goal should be to write the simplest and most elegant code that will get you to your desired outcome. However, there is sometimes a trade-off of the opportunity cost to learn a new formulation of code and the time it takes to write complex code that you already know. Remember that the best code is one that is easy to understand for yourself and your collaborators. Remember to comment your code, use informative names for variables and functions, and use reproducible methods to arrive at your output.

WRANGLING IN R: DPLYR

`dplyr` is a package in R that includes functions for data manipulation (i.e., data wrangling or data munging). `dplyr` is included in the tidyverse package, so you should already have it installed on your machine. The functions act as verbs for data wrangling processes. For more information, run this line of code:

```
vignette("dplyr")
```

```
## starting httpd help server ... done
```

Filter

Filtering allows us to choose certain rows (observations) in our dataset.

A few relevant commands: `==` `!=` `<` `<=` `>` `>=` `&` `|`

```
class(NTL.phys.data$lakeid)
```

```
## [1] "factor"
```

```
class(NTL.phys.data$depth)
```

```
## [1] "numeric"
```

```
# matrix filtering
NTL.phys.data.surface1 <- NTL.phys.data[NTL.phys.data$depth == 0,]

# dplyr filtering
NTL.phys.data.surface2 <- filter(NTL.phys.data, depth == 0)
NTL.phys.data.surface3 <- filter(NTL.phys.data, depth < 0.25)

# Did the methods arrive at the same result?
head(NTL.phys.data.surface1)
```

```
##    lakeid     lakename year4 daynum sampledate depth temperature_C
## 1       L    Paul Lake  1984    148    5/27/84     0          14.5
## 18      R   Peter Lake  1984    149    5/28/84     0          14.8
## 40      T Tuesday Lake  1984    150    5/29/84     0          15.0
## 56      L    Paul Lake  1984    155     6/3/84     0          18.8
## 72      R   Peter Lake  1984    156     6/4/84     0          18.8
## 90      T Tuesday Lake  1984    157     6/5/84     0          21.0
##    dissolvedOxygen irradianceWater irradianceDeck comments
## 1              9.5            1750           1620     <NA>
## 18             9.2            1630           1540     <NA>
```

```
## 40             9.5            1850           1960     <NA>
## 56             8.0            1100           1050     <NA>
## 72             9.0             275            275     <NA>
## 90             8.4            1200           1200     <NA>
```

```
dim(NTL.phys.data.surface1)
```

```
## [1] 1902   11
```

```
head(NTL.phys.data.surface2)
```

```
##   lakeid     lakename year4 daynum sampledate depth temperature_C
## 1      L    Paul Lake  1984    148    5/27/84     0          14.5
## 2      R   Peter Lake  1984    149    5/28/84     0          14.8
## 3      T Tuesday Lake  1984    150    5/29/84     0          15.0
## 4      L    Paul Lake  1984    155     6/3/84     0          18.8
## 5      R   Peter Lake  1984    156     6/4/84     0          18.8
## 6      T Tuesday Lake  1984    157     6/5/84     0          21.0
##   dissolvedOxygen irradianceWater irradianceDeck comments
## 1             9.5            1750           1620     <NA>
## 2             9.2            1630           1540     <NA>
## 3             9.5            1850           1960     <NA>
## 4             8.0            1100           1050     <NA>
## 5             9.0             275            275     <NA>
## 6             8.4            1200           1200     <NA>
```

```
dim(NTL.phys.data.surface2)
```

```
## [1] 1902   11
```

```
head(NTL.phys.data.surface3)
```

```
##   lakeid     lakename year4 daynum sampledate depth temperature_C
## 1      L    Paul Lake  1984    148    5/27/84     0          14.5
## 2      R   Peter Lake  1984    149    5/28/84     0          14.8
## 3      T Tuesday Lake  1984    150    5/29/84     0          15.0
## 4      L    Paul Lake  1984    155     6/3/84     0          18.8
## 5      R   Peter Lake  1984    156     6/4/84     0          18.8
## 6      T Tuesday Lake  1984    157     6/5/84     0          21.0
##   dissolvedOxygen irradianceWater irradianceDeck comments
## 1             9.5            1750           1620     <NA>
## 2             9.2            1630           1540     <NA>
## 3             9.5            1850           1960     <NA>
## 4             8.0            1100           1050     <NA>
## 5             9.0             275            275     <NA>
## 6             8.4            1200           1200     <NA>
```

```
dim(NTL.phys.data.surface3)
```

```
## [1] 1902   11
```

```
# MAtrix keep the row number. dplyr changes the row numbers


# Choose multiple conditions to filter
summary(NTL.phys.data$lakename)
```

```
## Central Long Lake     Crampton Lake    East Long Lake  Hummingbird Lake
```

```
##               539              1234              3905               430
##         Paul Lake        Peter Lake      Tuesday Lake         Ward Lake
##             10325             11288              6107               598
##    West Long Lake
##              4188
```

```
NTL.phys.data.PeterPaul1 <- filter(NTL.phys.data, lakename == "Paul Lake" | lakename == "Peter Lake")
NTL.phys.data.PeterPaul2 <- filter(NTL.phys.data, lakename != "Central Long Lake" &
                                     lakename != "Crampton Lake" & lakename != "East Long Lake" &
                                     lakename != "Hummingbird Lake" & lakename != "Tuesday Lake" &
                                     lakename != "Ward Lake" & lakename != "West Long Lake")
NTL.phys.data.PeterPaul3 <- filter(NTL.phys.data, lakename %in% c("Paul Lake", "Peter Lake"))
# %in% means include.


# Choose a range of conditions of a numeric or integer variable
summary(NTL.phys.data$daynum)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    55.0   166.0   194.0   194.3   222.0   307.0
```

```
NTL.phys.data.JunethruOctober1 <- filter(NTL.phys.data, daynum > 151 & daynum < 305)
NTL.phys.data.JunethruOctober2 <- filter(NTL.phys.data, daynum > 151, daynum < 305) # , is equal to and
NTL.phys.data.JunethruOctober3 <- filter(NTL.phys.data, daynum >= 152 & daynum <= 304)
NTL.phys.data.JunethruOctober4 <- filter(NTL.phys.data, daynum %in% c(152:304)) # 152 and 304 are inclu

# Exercise:
# filter NTL.phys.data for the year 1999
NTL.phys.data.1999 <- filter(NTL.phys.data, year4 == 1999)
# what code do you need to use, based on the class of the variable? Factor "", numbers alone
class(NTL.phys.data$year4)
```

```
## [1] "integer"
```

```
# Exercise:
# filter NTL.phys.data for Tuesday Lake from 1990 through 1999.
NTL.phys.data.19901999 <- filter(NTL.phys.data, lakename == "Tuesday Lake", year4 %in% c(1990:1999))
```

Question: Why don't we filter using row numbers?

> ANSWER: Not reprodusable. Not very efficient. You have to look what you want to do. MAybe in an actualization of the raw data the rows change.

Arrange

Arranging allows us to change the order of rows in our dataset. By default, the arrange function will arrange rows in ascending order.

```
NTL.phys.data.depth.ascending <- arrange(NTL.phys.data, depth)
NTL.phys.data.depth.descending <- arrange(NTL.phys.data, desc(depth))

# Exercise:
# Arrange NTL.phys.data by temperature, in descending order.
NTL.phys.data.temperature.descending <- arrange(NTL.phys.data, desc(temperature_C))
# Which dates, lakes, and depths have the highest temperatures?
head(NTL.phys.data.temperature.descending)
```

```
##   lakeid         lakename year4 daynum sampledate depth temperature_C
## 1      E   East Long Lake  1998    197    7/16/98   0.5          34.1
## 2      H Hummingbird Lake  2002    182     7/1/02   0.0          31.5
## 3      H Hummingbird Lake  2002    200    7/19/02   0.0          29.0
## 4      E   East Long Lake  1995    170    6/19/95   0.0          28.5
## 5      H Hummingbird Lake  2002    182     7/1/02   0.5          28.5
## 6      E   East Long Lake  1995    170    6/19/95   0.5          28.3
##   dissolvedOxygen irradianceWater irradianceDeck comments
## 1             7.5              69            395     <NA>
## 2             6.6              NA             NA     <NA>
## 3             7.4              NA             NA     <NA>
## 4             7.7             996           1095     <NA>
## 5             6.1              NA             NA     <NA>
## 6             7.7             153           1032     <NA>
```

```
#Summer months, East Long Lake, Hummingbird Lake; 0.5 and 1
```

Select

for columns. filter was for rows.

Selecting allows us to choose certain columns (variables) in our dataset.

```
NTL.phys.data.temps <- select(NTL.phys.data, lakename, sampledate:temperature_C) # use comas (and) and
```

Mutate

Mutating allows us to add new columns that are functions of existing columns. Operations include addition, subtraction, multiplication, division, log, and other functions.

```
NTL.phys.data.temps <- mutate(NTL.phys.data.temps, temperature_F = (temperature_C*9/5) + 32)
# the column goes always at the very end. NAs are kept.
```

Pipes

Sometimes we will want to perform multiple commands on a single dataset on our way to creating a processed dataset. We could do this in a series of subsequent commands or create a function. However, there is another method to do this that looks cleaner and is easier to read. This method is called a pipe. We designate a pipe with %>%. A good way to think about the function of a pipe is with the word "then."

Let's say we want to take our raw dataset (NTL.phys.data), *then* filter the data for Peter and Paul lakes, *then* select temperature and observation information, and *then* add a column for temperature in Fahrenheit:

```
NTL.phys.data.processed <-
  NTL.phys.data %>% #then   #you declare the data frame one time
  filter(lakename == "Paul Lake" | lakename == "Peter Lake") %>% #then
  select(lakename, sampledate:temperature_C) %>% #then
  mutate(temperature_F = (temperature_C*9/5) + 32)
# might replace a for loop
```

Notice that we did not place the dataset name inside the wrangling function but rather at the beginning.

Saving processed datasets

```
write.csv(NTL.phys.data.PeterPaul1, row.names = FALSE, file =
"./Data/Processed/NTL-LTER_Lake_ChemistryPhysics_PeterPaul_Processed.csv")
#row.names TRUE creates a row number column
```

CLOSING DISCUSSION

How did data wrangling help us to generate a processed dataset? How does this impact our ability to analyze and answer questions about our data?