Data Cleaning

## Quarto

Quarto enables you to weave together content and executable code into a finished document. To learn more about Quarto see <https://quarto.org>.

#loading in the packages  
  
library(rio)  
library(fixest)  
library(dplyr)

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':  
  
 filter, lag

The following objects are masked from 'package:base':  
  
 intersect, setdiff, setequal, union

library(vtable)

Loading required package: kableExtra

Attaching package: 'kableExtra'

The following object is masked from 'package:dplyr':  
  
 group\_rows

library(tidyverse)

── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
✔ forcats 1.0.0 ✔ readr 2.1.4  
✔ ggplot2 3.4.4 ✔ stringr 1.5.1  
✔ lubridate 1.9.3 ✔ tibble 3.2.1  
✔ purrr 1.0.2 ✔ tidyr 1.3.0

── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
✖ dplyr::filter() masks stats::filter()  
✖ kableExtra::group\_rows() masks dplyr::group\_rows()  
✖ dplyr::lag() masks stats::lag()  
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

#reading in the google trends data  
  
# Get a vector of filenames using list.files()  
# makes a list of all the file names "trends\_up\_to" and saves it as filenames  
filenames <- list.files(pattern = "trends\_up\_to\_", full.names = TRUE)  
  
# Read in the files using import\_list() and bind them together into a single dataset  
# imports and combines all the files  
data <- import\_list(filenames, rbind = TRUE, fill=TRUE)

#aggregating in the google trends data  
  
# Load the lubridate package  
library(lubridate)  
  
# Use str\_sub to get the first ten characters out of the monthorweek variable  
# takes first 10 characters from the monthorweek columm  
data <- data %>%  
 mutate(date = str\_sub(monthorweek, 1, 10))  
  
# Use ymd() function from lubridate to convert the string into a date variable  
# converts it into actual date object  
data <- data %>%  
 mutate(date = floor\_date(ymd(date), "month"))

# aggregating in the google trends data part 2  
# changes it to a one unit change in sd so that we can compare apples to apples  
# There are a couple missing index observations in the data (usually from rows after the data ends). However, a mean() or sd() function will return a NA if any of the observations are missing. Don't forget to use na.rm = TRUE in these functions  
  
data <- data %>%  
 group\_by(schname, keyword) %>%  
 mutate(standardized\_index = (index - mean(index, na.rm = TRUE)) / sd(index, na.rm = TRUE))  
  
#aggregating to the school-month level  
   
data <- data %>%  
 group\_by(schname, date) %>%  
 summarize(standardized\_index = mean(standardized\_index))

`summarise()` has grouped output by 'schname'. You can override using the  
`.groups` argument.

data <- na.omit(data)

#reading in the scorecard data  
  
scorecard\_data <- import("Most+Recent+Cohorts+(Scorecard+Elements).csv")  
  
# Read in the id\_name\_link file  
id\_name\_link <- import("id\_name\_link.csv")  
  
  
#filters for only universities that are Predominantly bachelor's-degree granting  
scorecard\_data <- filter(scorecard\_data, PREDDEG == 3)

#merge in the scorecard data  
  
# Count how many times each school name appears in id\_name\_link  
school\_name\_counts <- id\_name\_link %>%  
 group\_by(schname) %>%  
 mutate(n = n()) %>%  
 ungroup()  
  
# Filter out school names that show up more than once  
filtered\_id\_name\_link <- school\_name\_counts %>%  
 filter(n == 1)

#joining data together  
  
# Join Google trends data to id\_name\_link using schname  
joined\_data <- inner\_join(filtered\_id\_name\_link, data, by = c("schname" = "schname"))  
  
# Then join Scorecard data using unitid and/or opeid columns  
#final\_data <- inner\_join(joined\_data, scorecard\_data, by = c("UNITID", "OPEID"))  
  
final\_data <- inner\_join(joined\_data, scorecard\_data, by = c("unitid" = "UNITID", "opeid" = "OPEID"))

#exporting data  
  
library(writexl)  
  
write\_xlsx(final\_data, "final\_data.xlsx")