A Before/After Analysis of Speed Data

Gishar!

1/11/2023

Objective:

The objective of this project is to see if ***narrowing the lanes*** has any effects on driver speed.

---

1. Data

The data includes 24 hours of individual speed data at three different locations in both directions of travel for: - before condition - after condition (1 month after) - after condition (4 months after). Here are some details on the locations:

- Residential street
- Speed limit 25 mph
- *Before* condition road width 24 ft, between edges of pavement (not curb to curb), no edge lines
- *After* condition road width 20 ft, edge lines 2 ft from edge of pavement, no center line
- AADT (Average Annual Daily Traffic) of 2000 vehicles per day

1.1. Importing data

The data I have are in csv forms. There are 3 locations and each have 2 directions of travel, and each of those are collected on 3 different times (before condition, after condition right after lane road narrowing, and another one 4 months after the treatment). That is 18 files and I don't like to clutter my code importing files one by one and then work on each one individually. So, I tried to find a way to automate this importing into 18 different dataframes in one run. Each of the dataframes can take the name of the file and the values inside will be imported from the csv file.

- I used `list.files()` to collect the name of the data files into a temporary dataframe
- used `sub()` to remove the .csv from the file names and create individual dataframe names
- used `strsplit()` to extract location and direction from the names
- used `read.csv()` to read from the csv files
- used `assign()` to stored the data into individual dataframes - p.s. modified to not need this.
- add new columns for each dataframe containing the location and direction
- to automate this process, all were done in a *For loop*
- used `rbind()` to bind all dataframes into a single dataframe as csv files were being read
- removed all the interim variables and dataframes

Come to think of it, the easiest way for importing the data would have been to have 18 lines of read.csv() to read the files and then 5 lines for each to form the dataframes at each location. in total it would have been more than 100 lines of code to do what I did in 7 lines within a for loop. I am happy now!

1.2. Feature Engineering

The date and time is imported into the dataframe as character data and for this I will use the `lubridate` library to do some *feature engineering*. For this step, I extracted Year, month, day of month, Time, hour, and minute in separate columns. Although, for this type of work, most likely just the month and hours will be used in the analysis and comparisons. We'll see.

In addition, class of the variables Location, Direction, and Advice Code were assigned as factor

1.3. Cleaning up the date from outliers

The data is collected from a roadway with a speed limit of 25 mph inside a neighborhood. A quick boxplot of the speed data shows that the data includes many outliers that need to be shaved off the data. It can be observed that some of the speed data is shown as above 150 mph which is absurd. Another quick glance at boxplots of speed data for each levels of the mysterious factor variable AdviceCode shows that many of these outliers are tied to the level 128 of this variable.

- Used the `table()` to see how many records are under this level - turns our 304 out of total of 9243
- used `summary()` to see the distribution of speed data under this level - turns out Q1 is already higher than speed limit, and median is 127 mph

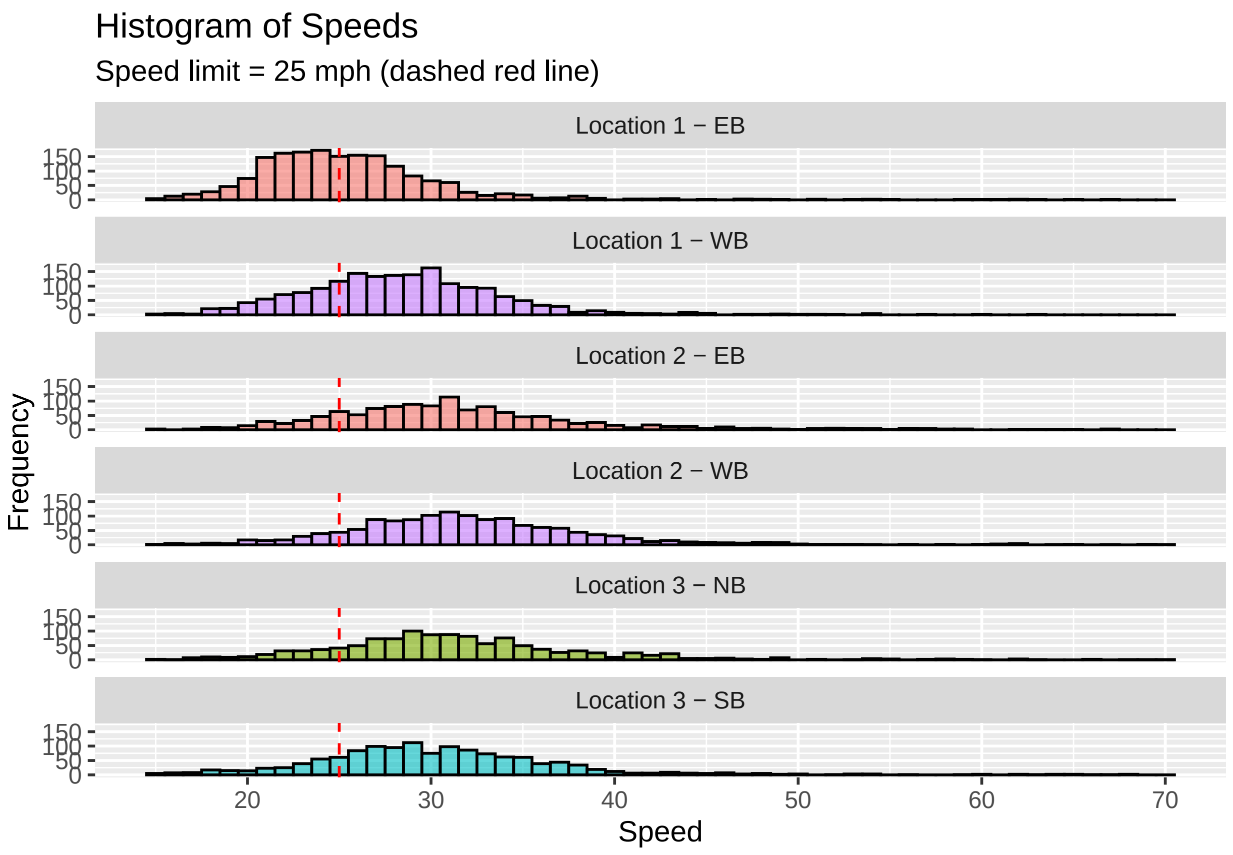
This tells me something strange is happening with this level of AdviceCode and I need to remove it from the data completely. Next, with my knowledge on the speed data, if not rare, it would be suicidal for anyone driving with a speed of higher than 70 mph. I will remove any speed higher than 70 mph. At the same time I will also remove any speed lower than 15 mph, as that would not be a likely case to be witnessed on such a road. First I extracted all the above records and there is no apparent pattern with time of day for these records. I noticed there is no pattern between these outlier speeds and time of day either. There are a total of 548 out of 9243 records that will be thrown away. Speed Data is redefined to exclude the outliers and the cryptic name of the locations were also changed to make them easier to follow by locations 1 to 3.

This will wrap up Part 1 of this project.

2. Analysis Process

The analysis of this data will include the following steps: Data Aggregation, EDA, and Statistical Analysis. First, I called the script for Part 1 to start with the cleaned and engineered dataframe created in the first steps. I also attached the Speed dataframe to make it easier to work with the variables for EDA, possibly. Let's make some initial observation on the distribution of data and see how speed data compares between locations, or maybe between two directions at the same location. For the time being, I am not going to worry about the column "Length" in the dataset which is supposedly about the length of each vehicle in each record. Let's do the following overall EDAs:

- Bar chart of volume for each location by direction in each data collection
- Overall histogram of speeds and by Month
- Histogram of speeds for each location and direction



Based on the histograms, it seems that:

- *Considering all locations along the test road, the 85th percentile speed is approximately 10 mph higher than the speed limit and about 5 mph higher than average*
- *Location 1 has a higher volume as compared to the other locations, also it shows a higher compliance with the speed limit especially in the EB direction out of the three locations/directions, since the distribution is centered closer to the speed limit compared to the rest.*
- *No specific pattern can be observed between the months of data collection*

2.1. Aggregation

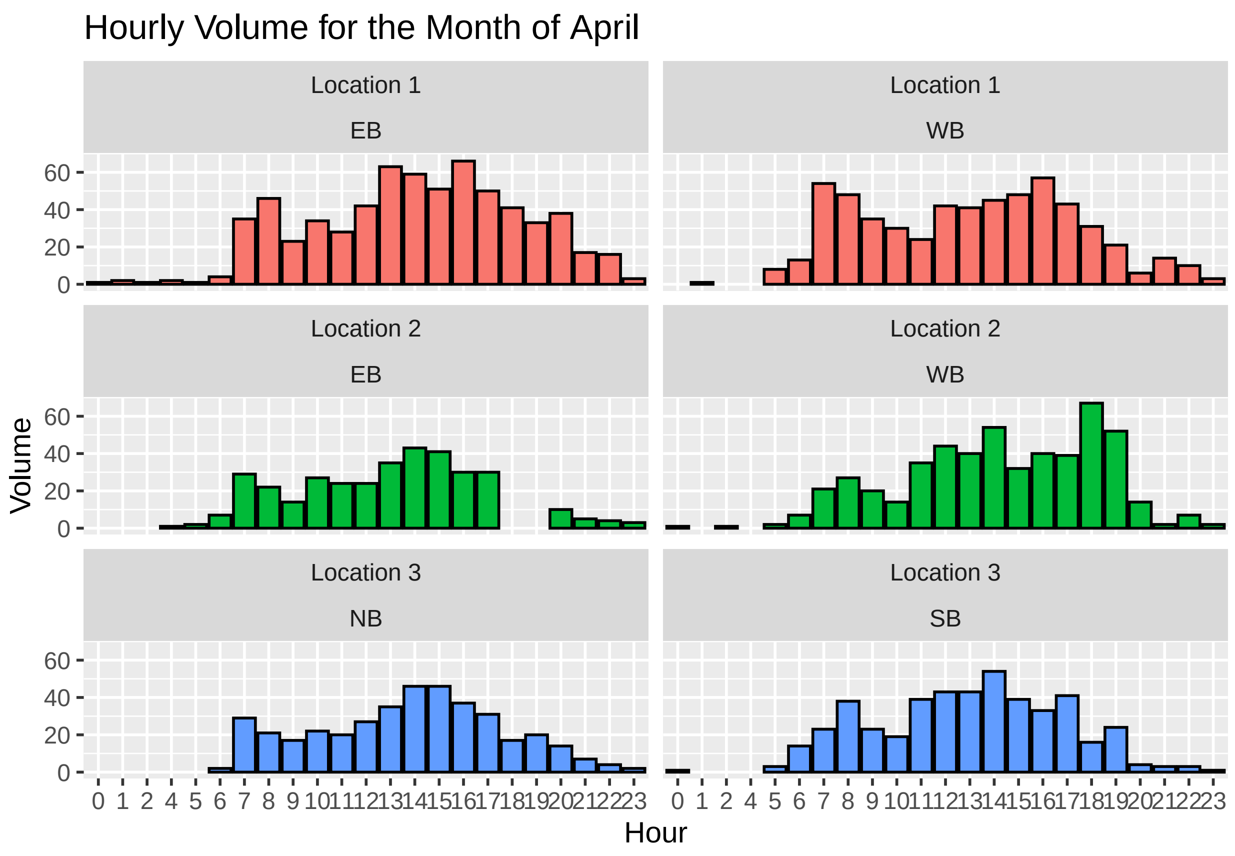
Generally, traffic volumes and speeds are grouped into bins of 15 minutes and 1-hour periods. due to the low volume of traffic in residential streets, there is no point in aggregating over 15-min intervals and so, I am going to make a new dataset to present aggregated statistics for hourly intervals which will be used for EDA and statistical analysis purposes later on. Several functions from the `dplyr` library are handy for this purpose such as `group by`, `summarise` and `arrange`.

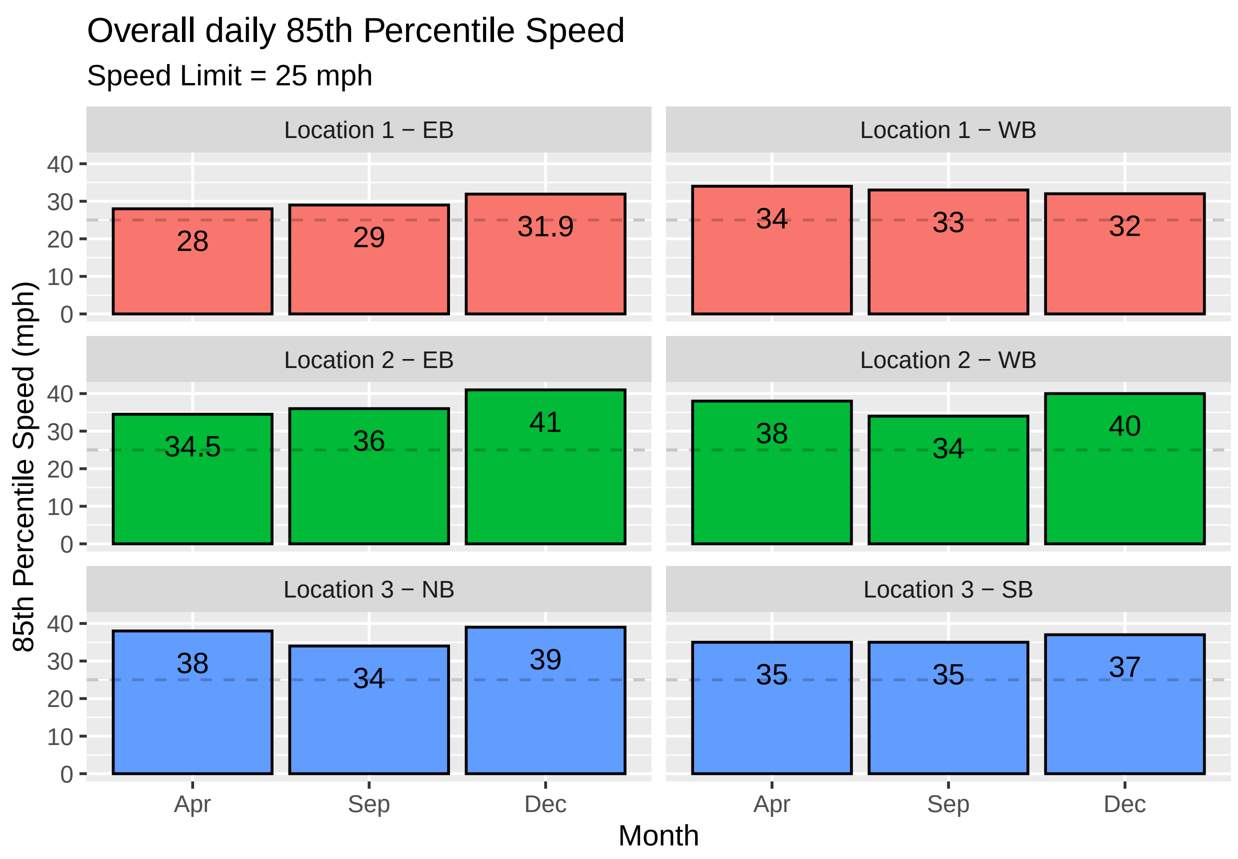
2.2. Visual before/after comparisons

At this I am interested to see how the hourly volumes and maybe 85th percentile speeds are distributed for each location/direction during each of the data collection efforts for the before and after conditions. The EDA shows the followings:

- Hourly volumes in none of the cases exceed 60 vehicles per hour, which is quite low

- Volumes peak around 7 AM for the morning rush hour and no specific pattern for the PM rush hour
- Comparing the 85th percentile speed for a location 1 over the three data collection periods
  - for the EB direction, it seems the 85th percentile increases after the lanes are narrowed
  - for the WB direction, it seems the 85th percentile does change much after the lanes are narrowed



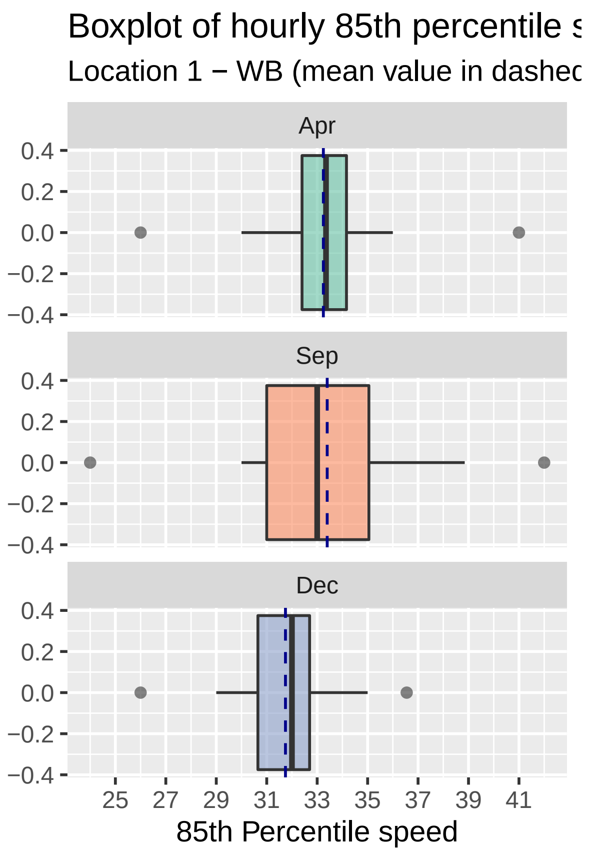


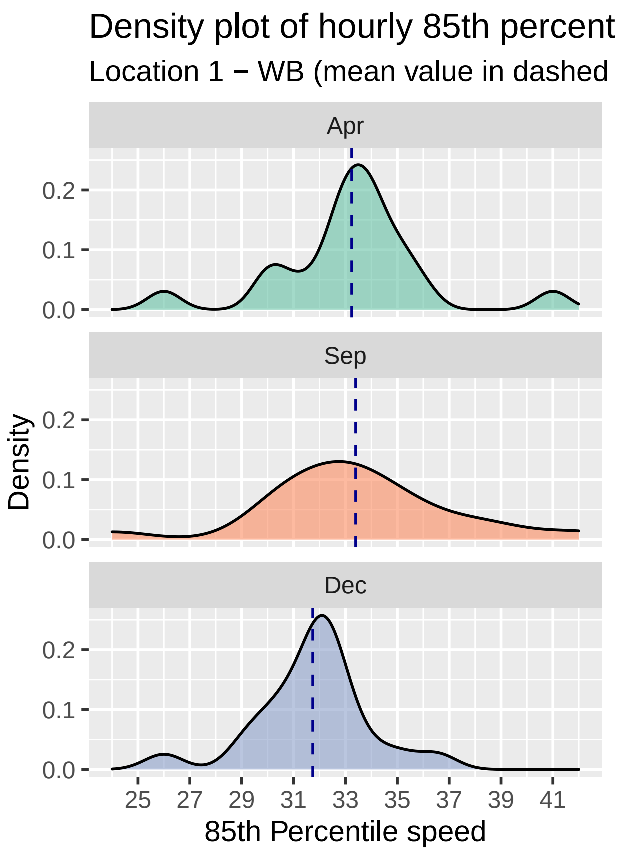
No specific patterns for the 85th percentile speed can be observed over the different data collection periods which makes the results inconclusive as if the lane narrowing is really effective in reducing the speeds of the vehicles. The same is true for the mean and median speeds as well.

2.3. Statistical before/after comparisons

Now that I did some EDA to have an idea of how the treatment has affected the speeds of vehicles, I will conduct some statistical tests to see if the differences observed are statistically significant as well. In my personal opinion, when I saw no specific pattern in the changes of the speed statistics before and after the lane narrowing was implemented, I don't see much need for giving a statistical taste to it as well but since this is a procedure and part of the steps I will do it anyway. I will conduct an ANOVA to see if there is a statistical difference between the average and 85th percentile speed of the vehicles for the three periods at any of the locations (main study question)

```
##                   Sep - Apr Dec - Sep Dec - Apr    ANOVA P.value
## Location 1 - EB  1.3462451  2.647062  3.993307 0.00000079932406
## Location 1 - WB -0.7928022 -1.149505 -1.942307 0.15104952904029
## Location 2 - EB  0.8673654  4.390198  5.257563 0.00010010042761
## Location 2 - WB -4.1989932  5.621305  1.422312 0.00000006747191
## Location 3 - NB -3.5773190  4.621210  1.043891 0.00001377592267
## Location 3 - SB -0.3756559  2.050108  1.674452 0.38318045516780
```





```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = Speed.85thP ~ Month, data = .)
##
## $Month
##               diff       lwr       upr     p adj
## Sep-Apr  0.152619 -2.093990 2.3992284 0.9854048
## Dec-Apr -1.502143 -3.748752 0.7444665 0.2504926
## Dec-Sep -1.654762 -3.873804 0.5642806 0.1808000
```

3. Conclusions

A quick conclusion of this analysis is that the lane narrowing treatment, did not have a conclusive outcome. I looked at 6 different data sets, each over a three data collection period. Before the treatment, a week after and a couple month after the treatment. The results observed were a mix of various observations, with higher or lower speeds observed between locations and dates.

In addition, although the ANOVA tests performed to see if the differences were statistically significant for some, but the higher speed in the after period shows that the objective of installing narrower lanes in the hope of reducing people's speed is not working.

This should end this project. Ciao

```
## - Session info ---------------------------------------------------------------
##  setting  value
##  version  R version 4.2.2 (2022-10-31 ucrt)
##  os       Windows 10 x64 (build 22000)
##  system   x86_64, mingw32
##  ui       RTerm
##  language (EN)
##  collate  English_United States.utf8
##  ctype    English_United States.utf8
##  tz       America/New_York
##  date     2023-02-01
##  pandoc   2.19.2 @ C:/Program Files/RStudio/resources/app/bin/quarto/bin/tools/ (via rmarkdown)
##
## - Packages -------------------------------------------------------------------
##  package       * version date (UTC) lib source
##  assertthat      0.2.1   2019-03-21 [1] CRAN (R 4.2.0)
##  backports       1.4.1   2021-12-13 [1] CRAN (R 4.2.0)
##  broom           0.8.0   2022-04-13 [1] CRAN (R 4.2.0)
##  bslib           0.3.1   2021-10-06 [1] CRAN (R 4.2.0)
##  cellranger      1.1.0   2016-07-27 [1] CRAN (R 4.2.0)
##  cli             3.3.0   2022-04-25 [1] CRAN (R 4.2.0)
##  colorspace      2.0-3   2022-02-21 [1] CRAN (R 4.2.0)
##  crayon          1.5.1   2022-03-26 [1] CRAN (R 4.2.0)
##  crosstalk       1.2.0   2021-11-04 [1] CRAN (R 4.2.0)
##  DBI             1.1.3   2022-06-18 [1] CRAN (R 4.2.2)
##  dbplyr          2.2.1   2022-06-27 [1] CRAN (R 4.2.2)
##  digest          0.6.29  2021-12-01 [1] CRAN (R 4.2.0)
##  dplyr         * 1.0.9   2022-04-28 [1] CRAN (R 4.2.0)
##  DT            * 0.26    2022-10-19 [1] CRAN (R 4.2.2)
##  ellipsis        0.3.2   2021-04-29 [1] CRAN (R 4.2.0)
##  evaluate        0.15    2022-02-18 [1] CRAN (R 4.2.0)
##  fansi           1.0.3   2022-03-24 [1] CRAN (R 4.2.0)
##  farver          2.1.0   2021-02-28 [1] CRAN (R 4.2.0)
##  fastmap         1.1.0   2021-01-25 [1] CRAN (R 4.2.0)
##  forcats       * 0.5.1   2021-01-27 [1] CRAN (R 4.2.0)
##  fs              1.5.2   2021-12-08 [1] CRAN (R 4.2.0)
##  gargle          1.2.1   2022-09-08 [1] CRAN (R 4.2.2)
##  generics        0.1.2   2022-01-31 [1] CRAN (R 4.2.0)
##  ggplot2       * 3.3.6   2022-05-03 [1] CRAN (R 4.2.0)
##  glue            1.6.2   2022-02-24 [1] CRAN (R 4.2.0)
##  googledrive     2.0.0   2021-07-08 [1] CRAN (R 4.2.2)
##  googlesheets4   1.0.1   2022-08-13 [1] CRAN (R 4.2.2)
##  gridExtra     * 2.3     2017-09-09 [1] CRAN (R 4.2.0)
##  gtable          0.3.0   2019-03-25 [1] CRAN (R 4.2.0)
##  haven           2.5.0   2022-04-15 [1] CRAN (R 4.2.0)
##  highr           0.9     2021-04-16 [1] CRAN (R 4.2.0)
##  hms             1.1.1   2021-09-26 [1] CRAN (R 4.2.0)
##  htmltools       0.5.2   2021-08-25 [1] CRAN (R 4.2.0)
##  htmlwidgets     1.5.4   2021-09-08 [1] CRAN (R 4.2.0)
##  httr            1.4.3   2022-05-04 [1] CRAN (R 4.2.0)
##  jquerylib       0.1.4   2021-04-26 [1] CRAN (R 4.2.0)
##  jsonlite        1.8.0   2022-02-22 [1] CRAN (R 4.2.0)
##  knitr           1.39    2022-04-26 [1] CRAN (R 4.2.0)
##  labeling        0.4.2   2020-10-20 [1] CRAN (R 4.2.0)
##  lifecycle       1.0.1   2021-09-24 [1] CRAN (R 4.2.0)
##  lubridate     * 1.8.0   2021-10-07 [1] CRAN (R 4.2.0)
##  magrittr        2.0.3   2022-03-30 [1] CRAN (R 4.2.0)
##  modelr          0.1.10  2022-11-11 [1] CRAN (R 4.2.2)
##  munsell         0.5.0   2018-06-12 [1] CRAN (R 4.2.0)
##  pillar          1.7.0   2022-02-01 [1] CRAN (R 4.2.0)
##  pkgconfig       2.0.3   2019-09-22 [1] CRAN (R 4.2.0)
##  purrr         * 0.3.4   2020-04-17 [1] CRAN (R 4.2.0)
##  R6              2.5.1   2021-08-19 [1] CRAN (R 4.2.0)
##  RColorBrewer  * 1.1-3   2022-04-03 [1] CRAN (R 4.2.0)
##  readr         * 2.1.2   2022-01-30 [1] CRAN (R 4.2.0)
##  readxl          1.4.0   2022-03-28 [1] CRAN (R 4.2.0)
##  reprex          2.0.2   2022-08-17 [1] CRAN (R 4.2.2)
##  rlang           1.0.6   2022-09-24 [1] CRAN (R 4.2.2)
##  rmarkdown       2.14    2022-04-25 [1] CRAN (R 4.2.0)
##  rstudioapi      0.13    2020-11-12 [1] CRAN (R 4.2.0)
##  rvest           1.0.3   2022-08-19 [1] CRAN (R 4.2.2)
##  sass            0.4.1   2022-03-23 [1] CRAN (R 4.2.0)
##  scales          1.2.0   2022-04-13 [1] CRAN (R 4.2.0)
##  sessioninfo     1.2.2   2021-12-06 [1] CRAN (R 4.2.0)
##  stringi         1.7.6   2021-11-29 [1] CRAN (R 4.2.0)
##  stringr       * 1.4.0   2019-02-10 [1] CRAN (R 4.2.0)
##  tibble        * 3.1.7   2022-05-03 [1] CRAN (R 4.2.0)
##  tidyr         * 1.2.0   2022-02-01 [1] CRAN (R 4.2.0)
##  tidyselect      1.1.2   2022-02-21 [1] CRAN (R 4.2.0)
##  tidyverse     * 1.3.2   2022-07-18 [1] CRAN (R 4.2.2)
##  tzdb            0.3.0   2022-03-28 [1] CRAN (R 4.2.0)
##  utf8            1.2.2   2021-07-24 [1] CRAN (R 4.2.0)
##  vctrs           0.4.1   2022-04-13 [1] CRAN (R 4.2.0)
##  withr           2.5.0   2022-03-03 [1] CRAN (R 4.2.0)
##  xfun            0.31    2022-05-10 [1] CRAN (R 4.2.0)
##  xml2            1.3.3   2021-11-30 [1] CRAN (R 4.2.0)
##  yaml            2.3.5   2022-02-21 [1] CRAN (R 4.2.0)
##
##  [1] C:/Users/ma998/AppData/Local/R/win-library/4.2
##  [2] C:/Program Files/R/R-4.2.2/library
##
## ------------------------------------------------------------------------------
```