p8105_hw2_mc5698.Rmd

2024-09-27

#Question 1

```
#loading necessary packages
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(readxl)
```

```
#clean the dataset
nyc_t =
  read_csv(
    "/Users/nicolechen/Downloads/p8105_hw2_mc5698/dataset/NYC_Transit_Subway_Entrance_And_Exit_Data.csv
    col_types = cols(Route8 = "c", Route9 = "c", Route10 = "c", Route11 = "c")) |>
  janitor::clean_names() |>
  select(
    line, station_name, station_latitude, station_longitude, route1, route2, route3, route4, route5, rou
  mutate(
    entry = ifelse(entry == "YES", TRUE, FALSE))
```

The dataset contains line, station_name, station_latitude, station_longitude, route1, route2, route3, route4, route5, route6, route7, route8, route9, route10, route11, entry, vending, entrance_type, ada. For the data cleaning, I removed unnecessary columns and convert the entry variable from character to a logical variable by using `case_match` function. The dimension of the resulting dataset is 1868, 19. These data are mostly tidy but we could pivot different route columns into one variable.

```
distinct_stations=
  nyc_t|>
  distinct(station_name,line)
nrow(distinct_stations)
```

```
## [1] 465
```

```r
ada_stations=
  nyc_t|>
  filter(ada==TRUE)|>
  distinct(station_name, line)

nrow(ada_stations)
```

```
## [1] 84
```

```r
no_vending=
  nyc_t |>
  filter(vending == "NO") |>
  pull(entry)

proportion_entry= mean(no_vending)
proportion_entry
```

```
## [1] 0.3770492
```

There are 465 distinct stations. 84 stations are ADA compliant.The proportion of station entrances/exits without vending allow entrance is `proportion_entry`.

```r
transfrom_ent=
  nyc_t |>
  pivot_longer(
    route1:route11,
    names_to = "route_num",
    values_to = "route")

A_stations=
    transfrom_ent |>
    filter(route == "A") |>
  select(station_name, line) |>
  distinct()


ada_stations =
  transfrom_ent |>
  filter(route == "A", ada == TRUE) |>
  select(station_name, line) |>
  distinct()
```

There are 56 distinct stations serve the A train and 16stations serve the A train and ADA compliant.

#Question 2

```r
#clean the datasets
mr_trash_wheel =
  readxl::read_excel("/Users/nicolechen/Downloads/p8105_hw2_mc5698/dataset/202409TrashWheelCollectionDa
  filter(!is.na(Dumpster)) |>
  mutate(Sports_Balls = as.integer(round(`Sports Balls`)),
         Year = as.character(Year),
         Trash_Wheel = "Mr. Trash Wheel")
```

```
## New names:
## * `` -> `...15`
## * `` -> `...16`
```

```
professor_trash_wheel =
  readxl::read_excel("/Users/nicolechen/Downloads/p8105_hw2_mc5698/dataset/202409TrashWheelCollectionDa
  filter(!is.na(Dumpster)) |>
  mutate(Year = as.character(Year),
         Trash_Wheel = "Professor Trash Wheel")

gwynnda_trash_wheel =
  readxl::read_excel("/Users/nicolechen/Downloads/p8105_hw2_mc5698/dataset/202409TrashWheelCollectionDa
  filter(!is.na(Dumpster)) |>
  mutate(Year = as.character(Year),
         Trash_Wheel = "Gwynnda Trash Wheel")

combined_data =
  bind_rows(mr_trash_wheel, professor_trash_wheel, gwynnda_trash_wheel)
combined_data
```

```
## # A tibble: 1,033 x 18
##    Dumpster Month Year  Date                `Weight (tons)`
##       <dbl> <chr> <chr> <dttm>                        <dbl>
##  1        1 May   2014  2014-05-16 00:00:00            4.31
##  2        2 May   2014  2014-05-16 00:00:00            2.74
##  3        3 May   2014  2014-05-16 00:00:00            3.45
##  4        4 May   2014  2014-05-17 00:00:00            3.1
##  5        5 May   2014  2014-05-17 00:00:00            4.06
##  6        6 May   2014  2014-05-20 00:00:00            2.71
##  7        7 May   2014  2014-05-21 00:00:00            1.91
##  8        8 May   2014  2014-05-28 00:00:00            3.7
##  9        9 June  2014  2014-06-05 00:00:00            2.52
## 10       10 June  2014  2014-06-11 00:00:00            3.76
## # i 1,023 more rows
## # i 13 more variables: `Volume (cubic yards)` <dbl>, `Plastic Bottles` <dbl>,
## #   Polystyrene <dbl>, `Cigarette Butts` <dbl>, `Glass Bottles` <dbl>,
## #   `Plastic Bags` <dbl>, Wrappers <dbl>, `Sports Balls` <dbl>,
## #   `Homes Powered*` <dbl>, ...15 <lgl>, ...16 <lgl>, Sports_Balls <int>,
## #   Trash_Wheel <chr>
```

By reading and cleaning the datasets, I combined the three datasets from `Mr. Trash Wheel`, `Professor Trash Wheel` and `Gwynnda Trash Wheel`. There are 1033 observations in the combined dataset. This dataset includes key variables such as`Dumpster`, which shows the the number of dumpster filled by trash, and `Cigarette Butts` which means the number of cigarette they collected. It also includes the specific time of the trash such as `Year`, `Date`, `Month` and `Trash_Wheel` indicates different trash types correspond to the different trash wheel. Moreover, it provides the detailed volumn and types for each trash wheel.

```
tw_professor =
  combined_data |>
  filter(Trash_Wheel == "Professor Trash Wheel") |>
  summarise(total_weight = sum(`Weight (tons)`, na.rm = TRUE))
```

```r
cb_gwynnda_june2022 =
  combined_data |>
  filter(Trash_Wheel == "Gwynnda Trash Wheel", Year == "2022", Month == "June") |>
  summarise(total_cig_butts = sum(`Cigarette Butts`, na.rm = TRUE))
```

The total weight of trash collected by Professor Trash Wheel wad 246.74. The total number of cigarette butts collected by Gwynnda in June of 2022 was 1.812×10^4.

#Question 3

```r
#read and clean the datasets
bakers =
  read_csv("/Users/nicolechen/Downloads/p8105_hw2_mc5698/dataset/gbb_datasets/bakers.csv") |>
  rename(Baker = `Baker Name`) |>
  mutate(source = "bakers")
```

```
## Rows: 120 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): Baker Name, Baker Occupation, Hometown
## dbl (2): Series, Baker Age
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
bakes =
  read_csv("/Users/nicolechen/Downloads/p8105_hw2_mc5698/dataset/gbb_datasets/bakes.csv") |>
  mutate(source = "bakes")
```

```
## Rows: 548 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): Baker, Signature Bake, Show Stopper
## dbl (2): Series, Episode
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
results =
  read_csv("/Users/nicolechen/Downloads/p8105_hw2_mc5698/dataset/gbb_datasets/results.csv", skip = 2) |>
  rename(Series = series,
         Episode = episode,
         Baker = baker,
         Technical = technical,
         Result = result) |>
  filter(!is.na(Series)) |>
  mutate(Series = as.numeric(Series),
         Episode = as.numeric(Episode))
```

```
## Rows: 1136 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
```

```
## chr (2): baker, result
## dbl (3): series, episode, technical
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
viewers =
  read_csv("/Users/nicolechen/Downloads/p8105_hw2_mc5698/dataset/gbb_datasets/viewers.csv") |>
  pivot_longer(cols = starts_with("Series"),
               names_to = "Series",
               names_prefix = "Series ",
               values_to = "Viewership") |>
  mutate(Series = as.numeric(Series))
```

```
## Rows: 10 Columns: 11
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (11): Episode, Series 1, Series 2, Series 3, Series 4, Series 5, Series ...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
#check for completeness and correctness across datasets
missing_bakers =
  results %>%
  anti_join(bakers, by = c("Baker", "Series"))
```

```
#rename the column names and merge the datasets
results =
  results |>
  mutate(Baker = tolower(trimws(Baker)),
         Series = as.numeric(Series))

bakers =
  bakers |>
  mutate(Baker = tolower(trimws(Baker)))

bakes =
  bakes |>
  mutate(Baker = tolower(trimws(Baker)),
         Series = as.numeric(Series))

final_datasets=
  results|>
  left_join(bakers, by = c("Baker", "Series")) |>
  left_join(bakes, by = c("Baker", "Series", "Episode"))%>%
  left_join(viewers, by = "Series")%>%
  arrange(Series, Episode.x) %>%
  select(Series, Episode.x, Baker, Baker_Age = `Baker Age`, Baker_Occupation = `Baker Occupation`, Techn
```

```
## Warning in left_join(., viewers, by = "Series"): Detected an unexpected many-to-many relationship be
## i Row 1 of `x` matches multiple rows in `y`.
## i Row 1 of `y` matches multiple rows in `x`.
```

```
## i If a many-to-many relationship is expected, set ‘relationship =
##   "many-to-many"‘ to silence this warning.
```

```
head(final_datasets)
```

```
## # A tibble: 6 x 14
##   Series Episode.x Baker  Baker_Age Baker_Occupation Technical Result Viewership
##    <dbl>     <dbl> <chr>      <dbl> <chr>                <dbl> <chr>       <dbl>
## 1      1         1 annet~        NA <NA>                     2 IN           2.24
## 2      1         1 annet~        NA <NA>                     2 IN           3
## 3      1         1 annet~        NA <NA>                     2 IN           3
## 4      1         1 annet~        NA <NA>                     2 IN           2.6
## 5      1         1 annet~        NA <NA>                     2 IN           3.03
## 6      1         1 annet~        NA <NA>                     2 IN           2.75
## # i 6 more variables: Signature_Bake <chr>, Showstopper <chr>, Hometown <chr>,
## #   source.x <chr>, source.y <chr>, Episode.y <dbl>
```

```
write_csv(final_datasets, "/Users/nicolechen/Downloads/p8105_hw2_mc5698/dataset/gbb_datasets/final_datas
```

For this project, I cleaned and organized data from `bakers.csv`, `bakes.csv`, `results.csv`, and `viewers.csv`. Firstly, I renamed the `Baker Name` column to `Baker` in the `bakers.csv` file to make it easier to match with other datasets. Then, I noticed that the baker names had different formats, so I used a function to convert all the names to lowercase and remove extra spaces.I also transformed the `Series` and `Episode` · columns to numeric for easier merging. For `viewers.csv`, I reshaped the data from wide to long format to align with the other files. After checking for missing bakers between the files, I merged the datasets step by step: first, results with `bakers`, then with `bakes`, and finally with `viewers`. I organized the data by series and episode to make it more readable. The final dataset contains all relevant information, including bakers' details, results, and viewership.

```
#Create a reader-friendly table showing the star baker or winner of each episode in Seasons 5 through 1
star_baker =
  final_datasets %>%
  filter(Series >= 5 & Series <= 10, Result %in% c("STAR BAKER", "WINNER")) %>%
  select(Series, Episode = Episode.x, Baker, Result) %>%
  arrange(Series, Episode)
star_baker
```

```
## # A tibble: 600 x 4
##    Series Episode Baker Result
##     <dbl>   <dbl> <chr> <chr>
##  1      5       1 nancy STAR BAKER
##  2      5       1 nancy STAR BAKER
##  3      5       1 nancy STAR BAKER
##  4      5       1 nancy STAR BAKER
##  5      5       1 nancy STAR BAKER
##  6      5       1 nancy STAR BAKER
##  7      5       1 nancy STAR BAKER
##  8      5       1 nancy STAR BAKER
##  9      5       1 nancy STAR BAKER
## 10      5       1 nancy STAR BAKER
## # i 590 more rows
```

From the table, I found that some people such as Richard become star bakers or winners in multiple episodes, which might make their overall success predictable.

```
#import, clean, tidy, and organize the viewership data
head(viewers, 10)
```

```
## # A tibble: 10 x 3
##    Episode Series Viewership
##      <dbl>  <dbl>      <dbl>
##  1       1      1       2.24
##  2       1      2       3.1
##  3       1      3       3.85
##  4       1      4       6.6
##  5       1      5       8.51
##  6       1      6      11.6
##  7       1      7      13.6
##  8       1      8       9.46
##  9       1      9       9.55
## 10       1     10       9.62
```

```
average_viewership =
  viewers %>%
  group_by(Series) %>%
  summarise(Average_Viewership =
              mean(Viewership, na.rm = TRUE))

season_1 =
  average_viewership %>%
  filter(Series == 1) %>%
  pull(Average_Viewership)

season_5 =
  average_viewership %>%
  filter(Series == 5) %>%
  pull(Average_Viewership)
```

The average viewership in Season 1 is 2.77, and the average viewership in Season 5 is approximately 10.0393, showing the growth in the show's popularity.

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

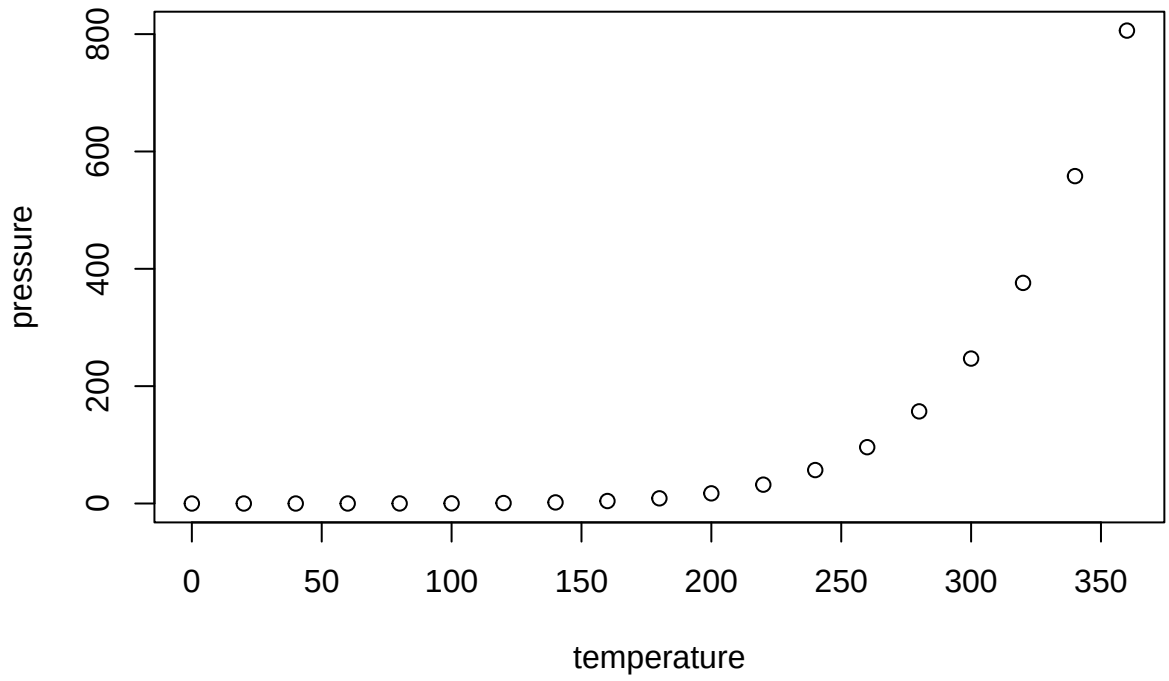
```
summary(cars)
```

```
##      speed           dist
##  Min.   : 4.0   Min.   :  2.00
##  1st Qu.:12.0   1st Qu.: 26.00
```

```
##  Median :15.0   Median : 36.00
##  Mean   :15.4   Mean   : 42.98
##  3rd Qu.:19.0   3rd Qu.: 56.00
##  Max.   :25.0   Max.   :120.00
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.