MSCS 264: Homework #14

Due Fri, April 21 at 11pm

Al Ashir Intisar

As always, submit your knitted pdf file to Moodle, but be sure your RMarkdown file is saved and accessible in your Submit folder on the RStudio server.

Part 1: Movie info

The code below shows our in class example, in which we are interested in the top 50 comedy movies.

```
url <- "https://www.imdb.com/search/title/?genres=comedy&explore=title_type,genres&pf_rd_m=A2FGELUUNOQJ

robotstxt::paths_allowed(url) #Don't forget!!!!
```

```
##  www.imdb.com
```

```
## [1] TRUE
```

```
title <- read_html(url) %>%
  html_nodes(".lister-item-header a") %>%
  html_text()

year <- read_html(url) %>%
  html_nodes(".text-muted.unbold") %>%
  html_text()

info <- read_html(url) %>%
  html_nodes(".text-muted:nth-child(2)") %>%
  html_text() %>%
  str_trim()

info
```

```
##  [1] "PG\n                  | \n               92 min\n                 | \n           \nAnimation,
##  [2] "TV-MA\n                 | \n                  30 min\n                 | \n           \nComedy,
##  [3] "TV-MA\n                 | \n                  60 min\n                 | \n           \nComedy,
##  [4] "TV-MA\n                 | \n                  30 min\n                 | \n           \nComedy,
##  [5] "PG-13\n                 | \n                  134 min\n                  | \n           \nAction
##  [6] "Adventure, Comedy, Fantasy           \n                    | \n               Post-productio
##  [7] "16+\n                  | \n           \nComedy"
##  [8] "PG-13\n                 | \n                  130 min\n                  | \n           \nAction
##  [9] "R\n                  | \n                93 min\n                | \n           \nComedy, Fan
## [10] "R\n                  | \n                114 min\n                | \n           \nAction, Co
## [11] "PG-13\n                 | \n                  90 min\n                 | \n           \nAction,
## [12] "TV-MA\n                 | \n                  57 min\n                 | \n           \nComedy,
## [13] "R\n                  | \n                179 min\n                | \n           \nComedy, Dra
```

```
## [14] "TV-MA\n                | \n              33 min\n                | \n           \nComedy,
## [15] "TV-MA\n                | \n              30 min\n                | \n           \nComedy,
## [16] "PG\n                 | \n               104 min\n                | \n           \nAdventure
## [17] "R\n                | \n              139 min\n                | \n           \nAction, Ad
## [18] "TV-MA\n                | \n              60 min\n                | \n           \nComedy,
## [19] "R\n                | \n              95 min\n                | \n           \nComedy, Thri
## [20] "TV-14\n                | \n              25 min\n                | \n           \nComedy"
## [21] "R\n                | \n              189 min\n                | \n           \nComedy, Dra
## [22] "TV-MA\n                | \n              60 min\n                | \n           \nAction,
## [23] "TV-14\n                | \n              22 min\n                | \n           \nAnimati
## [24] "TV-MA\n                | \n              30 min\n                | \n           \nAction,
## [25] "TV-MA\n                | \n              45 min\n                | \n           \nComedy,
## [26] "TV-14\n                | \n              22 min\n                | \n           \nComedy"
## [27] "TV-PG\n                | \n              30 min\n                | \n           \nComedy"
## [28] "TV-14\n                | \n           \nComedy, Musical, Romance"
## [29] "R\n                | \n              107 min\n                | \n           \nComedy, Ho
## [30] "PG-13\n                | \n              124 min\n                | \n           \nAction
## [31] "TV-14\n                | \n              30 min\n                | \n           \nComedy,
## [32] "R\n                | \n              127 min\n                | \n           \nAction, Co
## [33] "PG-13\n                | \n              97 min\n                | \n           \nAction,
## [34] "TV-PG\n                | \n              22 min\n                | \n           \nComedy,
## [35] "PG-13\n                | \n              126 min\n                | \n           \nComedy
## [36] "TV-MA\n                | \n              46 min\n                | \n           \nComedy,
## [37] "PG-13\n                | \n              150 min\n                | \n           \nAction
## [38] "TV-14\n                | \n              45 min\n                | \n           \nComedy,
## [39] "PG\n                 | \n               102 min\n                | \n           \nAnimation
## [40] "TV-14\n                | \n              22 min\n                | \n           \nComedy,
## [41] "R\n                | \n              114 min\n                | \n           \nComedy, Dra
## [42] "TV-MA\n                | \n              22 min\n                | \n           \nComedy"
## [43] "R\n                | \n              147 min\n                | \n           \nComedy, Dra
## [44] "TV-14\n                | \n              44 min\n                | \n           \nComedy,
## [45] "TV-PG\n                | \n              22 min\n                | \n           \nComedy,
## [46] "PG-13\n                | \n              104 min\n                | \n           \nComedy
## [47] "18+\n                | \n           \nComedy, Drama, Thriller"
## [48] "R\n                | \n              107 min\n                | \n           \nAction, Co
## [49] "TV-14\n                | \n              42 min\n                | \n           \nComedy,
## [50] "R\n                | \n              101 min\n                | \n           \nAction, Co
```

```
movies <- tibble(movie = title,
       year = year,
       info = info)

movies
```

```
## # A tibble: 50 x 3
##    movie                                year        info
##    <chr>                                <chr>       <chr>
##  1 The Super Mario Bros. Movie          (2023)      "PG\n                | ~
##  2 Beef                                 (2023- )    "TV-MA\n               ~
##  3 Succession                           (2018-2023) "TV-MA\n               ~
##  4 Ted Lasso                            (2020- )    "TV-MA\n               ~
##  5 Dungeons & Dragons: Honor Among Thieves (2023)   "PG-13\n               ~
##  6 Barbie                               (2023)      "Adventure, Comedy, Fant~
##  7 Jury Duty                            (2023- )    "16+\n                 |~
##  8 Shazam! Fury of the Gods             (2023)      "PG-13\n               ~
```

```
##  9 Renfield                                  (2023)      "R\n                 | \~
## 10 Operation Fortune: Ruse de guerre         (2023)      "R\n                 | \~
## # ... with 40 more rows
## # i Use 'print(n = ...)' to see more rows
```

1. Modify the code above to also create a column that includes the text of the description of the movie.

```
description <- read_html(url) %>%
  html_nodes("#main p:nth-child(3) , p.text-muted:nth-child(4)") %>%
    html_text() %>%
    str_trim()

movies <- tibble(movie = title,
                 year = year,
                 info = info,
                 description = description)

movies
```

```
## # A tibble: 50 x 4
##    movie                                    year        info              descr~1
##    <chr>                                    <chr>       <chr>             <chr>
##  1 The Super Mario Bros. Movie              (2023)      "PG\n           ~ "The s~
##  2 Beef                                     (2023- )    "TV-MA\n        ~ "Two p~
##  3 Succession                               (2018-2023) "TV-MA\n        ~ "The R~
##  4 Ted Lasso                                (2020- )    "TV-MA\n        ~ "Ameri~
##  5 Dungeons & Dragons: Honor Among Thieves  (2023)      "PG-13\n        ~ "A cha~
##  6 Barbie                                   (2023)      "Adventure, Come~ "To li~
##  7 Jury Duty                                (2023- )    "16+\n          ~ "It fo~
##  8 Shazam! Fury of the Gods                 (2023)      "PG-13\n        ~ "The f~
##  9 Renfield                                 (2023)      "R\n            ~ "Renfi~
## 10 Operation Fortune: Ruse de guerre        (2023)      "R\n            ~ "Speci~
## # ... with 40 more rows, and abbreviated variable name 1: description
## # i Use 'print(n = ...)' to see more rows
```

2. Explain why you might have trouble adding a column that indicates the number of votes.

```
votes <- read_html(url) %>%
  html_nodes("p.sort-num_votes-visible") %>%
  html_text()
```

Ans: Even though in the website we can see every movie have number of votes but when I try to scrape it it looks like there is only 48 rows which is inconsistent with the dimension of the movies dataset we ctreated.

3. We will add a column called "content_rating" to the movies tibble, by completing the following steps:

- create a column that is the first 5 characters of the "info" variable.

```
content_rating <- str_sub(info, 1, 5)
```

- use str_trim() to remove ending spaces and \n.

```
content_rating <- str_trim(content_rating)
```

- What to do about the things that are not ratings? Use the following list of "valid_ratings" and an ifelse statement to change these values to NA.

```
valid_content_ratings <- c("G", "PG", "PG-13", "R", "TV-PG", "TV-14", "TV-MA", "18+")

movies <- movies %>%
  mutate(content_rating = ifelse(!(content_rating %in% valid_content_ratings), NA, content_rating))
```

4. Notice that the year column has the format (xxxx) if it is a movie, and (xxxx- xxxx) if it is a show. Use to create a variable that indicates the type (movie or show). Hint: detect "-"

```
movies <- movies %>%
  mutate(category = ifelse(str_detect(year, "-"), "show", "movie"))
```

5. Count how many are movies and how many are shows.

```
movies|>
  summarise(num_movie = sum((category == "movie")), num_show = sum((category == "show")))
```

```
## # A tibble: 1 x 2
##   num_movie num_show
##       <int>    <int>
## 1        25       25
```

6. Create a variable that is the year as a numeric variable. For a show, the year column should show the first year.

```
movies <- movies|>
  mutate(year = parse_number(year))
```

Part 2: TV show characters

Go to this website: transcripts.foreverdreaming.org

Find a TV show that you like. Your goal is to create a graph similar to those in images/hw14_tv_lines.png. (You can choose to make either the bar graph, or the line plot, your choice!)

As you can see, I've chosen to use the 9 season finale episodes for the TV show "The Office". Then, I determine the 5 characters with the most lines (across all the episodes), and then create the graphs. I'll walk you through the steps!

- Pick any show of your choice! I chose to use all the season finales, but you could just pick all the shows in one season, or whatever you want. Just be sure to have at least 5 episodes.

7. Start with one episode. Navigate to the page with the transcript. Use selector gadget to pull:

```
episode_url <- "https://transcripts.foreverdreaming.org/viewtopic.php?t=6375"

robotstxt::paths_allowed(episode_url) #Don't forget!!!!
```

```
## transcripts.foreverdreaming.org

## Warning in request_handler_handler(request = request, handler = on_not_found, :
## Event: on_not_found

## Warning in request_handler_handler(request = request, handler =
## on_file_type_mismatch, : Event: on_file_type_mismatch

## Warning in request_handler_handler(request = request, handler =
## on_suspect_content, : Event: on_suspect_content

##
```

```
## [1] TRUE
```

- a vector of all the speakers of each line.

```
content <- read_html(episode_url) %>%
  html_nodes(".content") %>%
  html_text()


speakers <- regmatches(content, gregexpr("\\w[A-Z]+:", content))|>
  unlist()
speakers <- gsub(":", "", speakers)
```

- the name/title of the episode.

```
ep_title <- read_html(episode_url) %>%
    html_nodes(".first a") %>%
    html_text()
```

8. Put these two things together in a tibble, for example:

```
tibble(speaker, episode_title = episode)
```

```
gilmore_girls <- tibble(speakers, episode_tittle = ep_title)
```

9. Write a function that will do steps 7 and 8, that has the url of the episode as in the input, and a tibble as the output. Then use bind_rows() to put the resulting tibbles together. (See hw 12 key).

```
scrape_ep <- function(ep_url){
  robotstxt::paths_allowed(ep_url)

  content <- read_html(ep_url) %>%
    html_nodes(".content") %>%
    html_text()


  speakers <- regmatches(content, gregexpr("\\w[A-Z]+:", content))|>
    unlist()
  speakers <- gsub(":", "", speakers)


  ep_title <- read_html(ep_url) %>%
    html_nodes(".first a") %>%
    html_text()

  tibble(speakers, episode_tittle = ep_title)

}
```

```
s4_ep15 <- "https://transcripts.foreverdreaming.org/viewtopic.php?t=6375"
s4_ep16 <- "https://transcripts.foreverdreaming.org/viewtopic.php?t=6376"
s4_ep17 <- "https://transcripts.foreverdreaming.org/viewtopic.php?t=6377"
s4_ep18 <- "https://transcripts.foreverdreaming.org/viewtopic.php?t=6378"
s4_ep19 <- "https://transcripts.foreverdreaming.org/viewtopic.php?t=6379"
s4_ep20 <- "https://transcripts.foreverdreaming.org/viewtopic.php?t=6380"

ep_15 <- scrape_ep(s4_ep15)
```

```
## transcripts.foreverdreaming.org
```

```
## Warning in request_handler_handler(request = request, handler = on_not_found, :
## Event: on_not_found
```

```
## Warning in request_handler_handler(request = request, handler =
## on_file_type_mismatch, : Event: on_file_type_mismatch
```

```
## Warning in request_handler_handler(request = request, handler =
## on_suspect_content, : Event: on_suspect_content
```

```
##
```

```
ep_16 <- scrape_ep(s4_ep16)
```

```
## transcripts.foreverdreaming.org
```

```
## Warning in request_handler_handler(request = request, handler = on_not_found, :
## Event: on_not_found
```

```
## Warning in request_handler_handler(request = request, handler =
## on_file_type_mismatch, : Event: on_file_type_mismatch
```

```
## Warning in request_handler_handler(request = request, handler =
## on_suspect_content, : Event: on_suspect_content
```

```
##
```

```
ep_17 <- scrape_ep(s4_ep17)
```

```
## transcripts.foreverdreaming.org
```

```
## Warning in request_handler_handler(request = request, handler = on_not_found, :
## Event: on_not_found
```

```
## Warning in request_handler_handler(request = request, handler =
## on_file_type_mismatch, : Event: on_file_type_mismatch
```

```
## Warning in request_handler_handler(request = request, handler =
## on_suspect_content, : Event: on_suspect_content
```

```
##
```

```
ep_18 <- scrape_ep(s4_ep18)
```

```
## transcripts.foreverdreaming.org
```

```
## Warning in request_handler_handler(request = request, handler = on_not_found, :
## Event: on_not_found
```

```
## Warning in request_handler_handler(request = request, handler =
## on_file_type_mismatch, : Event: on_file_type_mismatch
```

```
## Warning in request_handler_handler(request = request, handler =
## on_suspect_content, : Event: on_suspect_content
```

```
##
```

```
ep_19 <- scrape_ep(s4_ep19)
```

```
## transcripts.foreverdreaming.org
```

```
## Warning in request_handler_handler(request = request, handler = on_not_found, :
## Event: on_not_found
```

```
## Warning in request_handler_handler(request = request, handler =
## on_file_type_mismatch, : Event: on_file_type_mismatch
```

```
## Warning in request_handler_handler(request = request, handler =
## on_suspect_content, : Event: on_suspect_content
```

```
##
```

```
ep_20 <- scrape_ep(s4_ep20)
```

```
## transcripts.foreverdreaming.org
```

```
## Warning in request_handler_handler(request = request, handler = on_not_found, :
## Event: on_not_found
```

```
## Warning in request_handler_handler(request = request, handler =
## on_file_type_mismatch, : Event: on_file_type_mismatch
```

```
## Warning in request_handler_handler(request = request, handler =
## on_suspect_content, : Event: on_suspect_content
```

```
##
```

```
gilmore_girls <- ep_15 |>
bind_rows(ep_16) |>
bind_rows(ep_17) |>
bind_rows(ep_18) |>
bind_rows(ep_19) |>
bind_rows(ep_20)
```

10. Add columns to your resulting tibble that includes the season number and the episode number. Be sure these are numeric! Write this csv file to your student submit folder. (As an example, see office_finales.csv in the Class/Data folder). (Note: If you weren't successful in your scraping, you can use office_finales.csv for problems 11-14. If you want credit for this problem, also show me how you'd create the variables season and episode from "episode_title")

```
gilmore_girls <- gilmore_girls|>
  mutate(season = parse_number(episode_tittle))|>
  mutate(episode = parse_number(str_sub(episode_tittle, 3)))
```

```
write_csv(gilmore_girls, "~/Mscs 264 S23/Submit Section A/Homeworks /gilmore_girls.csv")
```

11. From your tibble, find the 5 characters with the most lines.

```
gilmore_girls|>
  group_by(speakers)|>
  summarise(line_ct = n())|>
  slice_max(line_ct, n = 5)
```

```
## # A tibble: 5 x 2
##   speakers line_ct
##   <chr>      <int>
## 1 LORELAI     1184
## 2 RORY         836
## 3 LUKE         412
## 4 EMILY        338
## 5 PARIS        249
```

12. Create a vector that lists these 5 characters names. Use it to filter your tibble to include only these 5 characters.

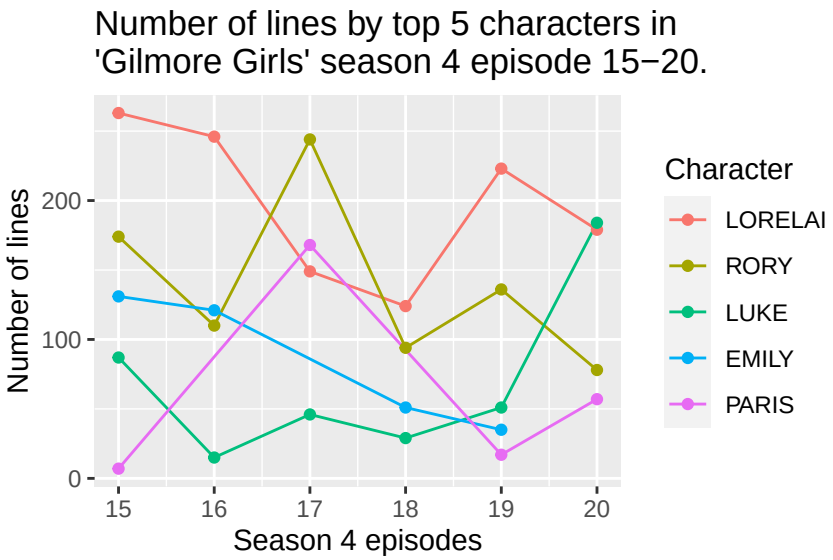
```
main_chr <- gilmore_girls|>
  group_by(speakers)|>
  summarise(line_ct = n())|>
  slice_max(line_ct, n = 5)|>
  select(speakers)
```

```
gilmore_girls_main <- gilmore_girls|>
  filter(speakers %in% main_chr$speakers)
```

13. From this filtered tibble, you should be able to create either a bar graph or a line graph! (It may take more data summarization!). Use appropriate titles, axis and legend labels, and a theme/color palette of your choice. Be sure to order the legend according to most lines spoken over all. (For line graph, could also order by the order in the last episode).

```
gilmore_girls_main|>
  group_by(speakers, episode)|>
  summarise(count = n())|>
  ungroup()|>
  ggplot(aes(x = episode, y = count, color = fct_reorder2(speakers, episode, count, .fun = sum)))+
  geom_line()+
  geom_point()+
  labs(title = "Number of lines by top 5 characters in\n'Gilmore Girls' season 4 episode 15-20.",
       x = "Season 4 episodes", y = "Number of lines", color = "Character")
```

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



14. Tell me something interesting that you see when you look at your graph! Hopefully you've picked a show you like, so you may have some background knowledge/insights!

Ans: We can see that Lorelai is high up in the graph for all 6 episodes because she is the main character of the season. Another fact is that Paris and Rory had a steep rise on episode 17. Looks like their rise and fall is simultaneous because they go to the same school and hate each other. They don't have any related character in common an they hate each other. Therefore, when Paris has a scene it's almost always with Rory.