RWorksheet#5_group(3)

Songaling, Porras, Sumintao, Lumahan

2023-12-22

1. Each group needs to extract the top 50 tv shows in Imdb.com. It will include the rank, the title of the tv show, tv rating, the number of people who voted, the number of episodes, the year it was released.

```
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(rvest)
library(polite)
library(httr)

url <- "https://www.imdb.com/chart/tvmeter/?ref_=nv_tvv_mptv"

session <- bow(url, user_agent = "Educational")
session
```

```
## <polite session> https://www.imdb.com/chart/tvmeter/?ref_=nv_tvv_mptv
##     User-agent: Educational
##     robots.txt: 34 rules are defined for 2 bots
##    Crawl delay: 5 sec
##   The path is scrapable for this user-agent
```

```
#RANK
ranking <- scrape(session) %>%
    html_nodes("div.sc-94da5b1b-0.soBIM.meter-const-ranking.sc-479faa3c-6.glWBvR.cli-meter-title-header
    html_text
ranks_top50 <- ranking[1:50]
ranks_top50 <- substr(ranks_top50, 1, 2)
ranks_top50
```

```
##  [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## [26] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
```

```
#TITLE
title_list <- scrape(session) %>%
    html_nodes("h3.ipc-title__text") %>%
    html_text
```

```r
titles_top50 <- title_list[2:51]
titles_top50
```

```
##  [1] "Reacher"                          "Doctor Who"
##  [3] "Fargo"                            "Obliterated"
##  [5] "My Life with the Walter Boys"     "The Crown"
##  [7] "Monarch: Legacy of Monsters"      "Slow Horses"
##  [9] "Rick and Morty"                   "Game of Thrones"
## [11] "A Murder at the End of the World" "Archie"
## [13] "The Gilded Age"                   "Fallout"
## [15] "Jujutsu Kaisen"                   "The Boys"
## [17] "Mr. & Mrs. Smith"                 "Vigil"
## [19] "Invincible"                       "Yu Yu Hakusho"
## [21] "True Detective"                   "Blue Eye Samurai"
## [23] "Yellowstone"                      "Brooklyn Nine-Nine"
## [25] "For All Mankind"                  "Suits"
## [27] "The Fall of the House of Usher"   "Grey's Anatomy"
## [29] "Young Sheldon"                    "Friends"
## [31] "Breaking Bad"                     "The Santa Clauses"
## [33] "The Walking Dead"                 "Shameless"
## [35] "Attack on Titan"                  "The Bear"
## [37] "The Office"                       "The Sopranos"
## [39] "Supernatural"                     "Lawmen: Bass Reeves"
## [41] "Sweet Home"                       "Succession"
## [43] "Black Mirror"                     "A Nearly Normal Family"
## [45] "Stranger Things"                  "The Big Bang Theory"
## [47] "House of the Dragon"              "All the Light We Cannot See"
## [49] "American Horror Story"            "Ted Lasso"
```

```r
#TV RATING
tvratings <- scrape(session) %>%
    html_nodes("span.ipc-rating-star.ipc-rating-star--base.ipc-rating-star--imdb.ratingGroup--imdb-ratin
    html_text
tv_ratingtop50 <- tvratings[1:50]
tv_rating50only <- substr(tv_ratingtop50, 1, 3)
tv_rating50only
```

```
##  [1] "8.1" "8.6" "8.9" "6.7" "7.1" "8.6" "7.1" "8.0" "9.1" "9.2" "7.2" "7.1"
## [13] "8.0" "8.6" "8.7" "7.4" "8.7" "7.4" "8.9" "8.8" "8.7" "8.4" "8.1" "8.4"
## [25] "8.0" "7.6" "7.6" "8.9" "9.5" "6.4" "8.1" "8.5" "9.1" "8.6" "9.0" "9.2"
## [37] "8.4" "7.6" "7.3" "8.9" "8.7" "6.9" "8.7" "8.2" "8.5" "7.5" "8.0" "8.8"
## [49] "7.8" "8.0"
```

```r
#VOTES
votes_num <- scrape(session) %>%
    html_nodes("span.ipc-rating-star--voteCount") %>%
    html_text
num_votestop50 <- votes_num[1:50]
num_votestop50
```

```
##  [1] " (163K)" " (241K)" " (399K)" " (9.7K)" " (4.3K)" " (244K)" " (13K)"
##  [8] " (53K)"  " (583K)" " (2.2M)" " (10K)"  " (1.2K)" " (33K)"  " (98K)"
## [15] " (613K)" " (25K)"  " (183K)" " (3.7K)" " (618K)" " (35K)"  " (200K)"
## [22] " (349K)" " (59K)"  " (462K)" " (88K)"  " (334K)" " (88K)"  " (1.1M)"
## [29] " (2.1M)" " (10K)"  " (1.1M)" " (272K)" " (481K)" " (172K)" " (683K)"
## [36] " (451K)" " (473K)" " (5.7K)" " (29K)"  " (248K)" " (623K)" " (7.8K)"
## [43] " (1.3M)" " (851K)" " (351K)" " (33K)"  " (338K)" " (324K)" " (2.6K)"
## [50] " (5.1K)"
```

```
#NUMBER OF EPISODES
eps_num <- scrape(session) %>%
    html_nodes("span.sc-479faa3c-8.bNrEFi.cli-title-metadata-item:contains('eps')") %>%
    html_text %>%
  gsub("\\D", "", .)
num_eps_top50 <- eps_num[1:50]
num_eps_top50
```

```
##  [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## [26] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
```

```
#YEAR RELEASED
year_released <- scrape(session) %>%
    html_nodes("span.sc-479faa3c-8.bNrEFi.cli-title-metadata-item:contains('20')") %>%
    html_text
year_releasedtop50 <- year_released[1:50]


# CREATE DATAFRAME
tv_shows_data <- data.frame(
  Rank = ranks_top50,
  Title = titles_top50,
  TV_Rating = tv_rating50only,
  Num_Votes = num_votestop50,
  Num_Episodes = num_eps_top50,
  Year_Released = year_releasedtop50
)

View(tv_shows_data)
```

```
## Warning in View(tv_shows_data): unable to open display
```

```
## Error in .External2(C_dataviewer, x, title): unable to start data viewer
```

2. From the 50 tv shows, select at least 5 tv shows to scrape the user reviews that will include the reviewer's name, date of reviewed, user rating, title of the review, and text reviews.

```
list_link <- scrape(session) %>%
  html_nodes('a.ipc-title-link-wrapper') %>%
  html_attr('href')

link_of_list5 <- list_link[1:5]

for (i in 1:5) {
 link_of_list5[i] <- paste0("https://imdb.com", link_of_list5[i], sep = "")
}

link_of_list5
```

```
## [1] "https://imdb.com/title/tt9288030/?ref_=chttvm_t_1"
## [2] "https://imdb.com/title/tt0436992/?ref_=chttvm_t_2"
## [3] "https://imdb.com/title/tt2802850/?ref_=chttvm_t_3"
## [4] "https://imdb.com/title/tt11097240/?ref_=chttvm_t_4"
## [5] "https://imdb.com/title/tt8323628/?ref_=chttvm_t_5"
```

```
library(stringr)

split_links <- strsplit(link_of_list5, "/")
split_links
```

```
## [[1]]
## [1] "https:"             ""                   "imdb.com"           "title"
## [5] "tt9288030"          "?ref_=chttvm_t_1"
##
## [[2]]
## [1] "https:"             ""                   "imdb.com"           "title"
## [5] "tt0436992"          "?ref_=chttvm_t_2"
##
## [[3]]
## [1] "https:"             ""                   "imdb.com"           "title"
## [5] "tt2802850"          "?ref_=chttvm_t_3"
##
## [[4]]
## [1] "https:"             ""                   "imdb.com"           "title"
## [5] "tt11097240"         "?ref_=chttvm_t_4"
##
## [[5]]
## [1] "https:"             ""                   "imdb.com"           "title"
## [5] "tt8323628"          "?ref_=chttvm_t_5"
```

```
unique_ids <- sapply(split_links, function(link) {

  id_part <- grep("tt[0-9]+", link, value = TRUE)

  if (length(id_part) > 0) {
    return(id_part)
  } else {
    return(NA)
  }
})

# Pasting the Unique ID to imdb site to access the review section
for (i in 1:5) {
 unique_ids[i] <- paste0("https://imdb.com/title/",unique_ids[i],"/reviews?ref_=tt_urv", sep = "") }

unique_ids
```

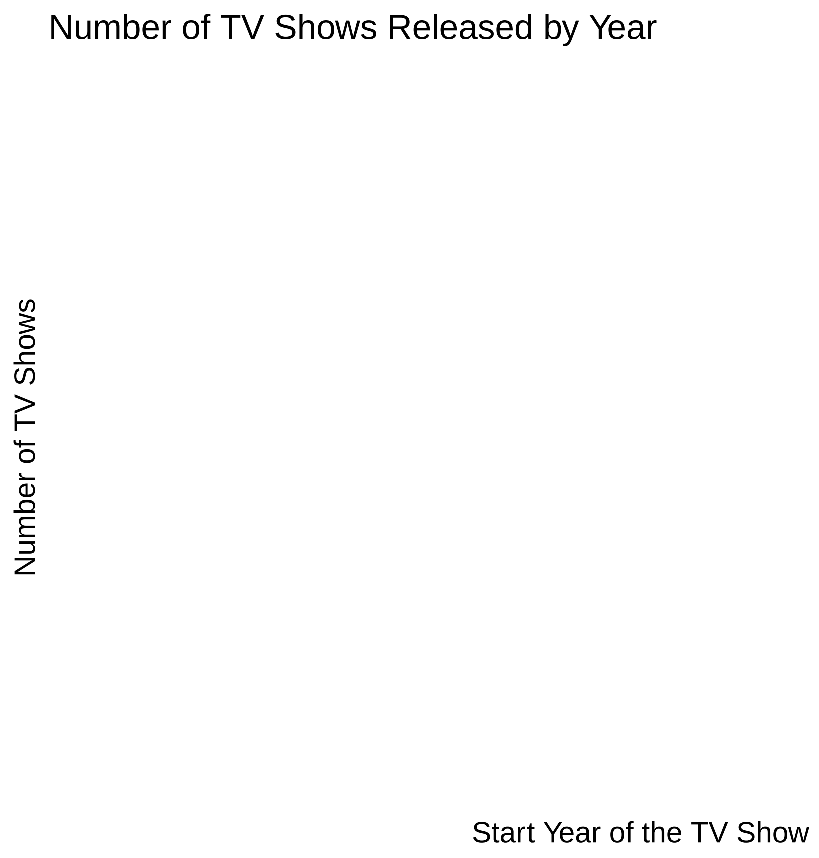
```
## [1] "https://imdb.com/title/tt9288030/reviews?ref_=tt_urv"
## [2] "https://imdb.com/title/tt0436992/reviews?ref_=tt_urv"
## [3] "https://imdb.com/title/tt2802850/reviews?ref_=tt_urv"
## [4] "https://imdb.com/title/tt11097240/reviews?ref_=tt_urv"
## [5] "https://imdb.com/title/tt8323628/reviews?ref_=tt_urv"
```

```
rowrow <- 1
put <- 1

review_list <- list()
rank_one_review <- data.frame()
rank_two_review <- data.frame()
```

```r
rank_three_review <- data.frame()
rank_four_review <- data.frame()
rank_five_review <- data.frame()

for (rowrow in 1:5){

  url  <- unique_ids[rowrow]
  session2 <- bow(url, user_agent = "Education")
  webpage <- scrape(session2)

  nameofreviewer <- html_text(html_nodes(webpage,"span.display-name-link"))
  nameofreviewer5 <- nameofreviewer[1:5]
  nameofreviewer5

  date_of_review <- html_text(html_nodes(webpage,"span.review-date"))
  date_of_review5 <- date_of_review[1:5]
  date_of_review5

  user_rating <- html_text(html_nodes(webpage,"span.rating-other-user-rating"))
  user_rating5 <-  user_rating[1:5]
  user_rating5


  title_of_review <- html_text(html_nodes(webpage,"a.title"))
  title_of_review5 <- title_of_review[1:5]
  title_of_review5

  text_review <- html_text(html_nodes(webpage,"div.text.show-more__control"))
  text_review5 <- text_review[1:5]
  text_review5


  for(put in 1:5){

    if(rowrow == 1){

      rank_one_review[put, 1] <- nameofreviewer5[put]
      rank_one_review[put, 2] <- date_of_review5[put]
      rank_one_review[put, 3] <- user_rating5[put]
      rank_one_review[put, 4] <- title_of_review5[put]
      rank_one_review[put, 5] <- text_review5[put]


    } else if(rowrow == 2){

      rank_two_review[put, 1] <- nameofreviewer5[put]
      rank_two_review[put, 2] <- date_of_review5[put]
      rank_two_review[put, 3] <- user_rating5[put]
      rank_two_review[put, 4] <- title_of_review5[put]
      rank_two_review[put, 5] <- text_review5[put]


    }else if(rowrow == 3){
      rank_three_review[put, 1] <- nameofreviewer5[put]
      rank_three_review[put, 2] <- date_of_review5[put]
      rank_three_review[put, 3] <- user_rating5[put]
      rank_three_review[put, 4] <- title_of_review5[put]
      rank_three_review[put, 5] <- text_review5[put]

    }else if(rowrow == 4){

      rank_four_review[put, 1] <- nameofreviewer5[put]
      rank_four_review[put, 2] <- date_of_review5[put]
      rank_four_review[put, 3] <- user_rating5[put]
      rank_four_review[put, 4] <- title_of_review5[put]
      rank_four_review[put, 5] <- text_review5[put]

    }else if(rowrow == 5){

      rank_five_review[put, 1] <- nameofreviewer5[put]
      rank_five_review[put, 2] <- date_of_review5[put]
      rank_five_review[put, 3] <- user_rating5[put]
      rank_five_review[put, 4] <- title_of_review5[put]
      rank_five_review[put, 5] <- text_review5[put]

     }
  }

  rowrow <- rowrow + 1
}
```

3. Create a time series graph for the tv shows released by year. Which year has the most number of tv shows released?

```
library(ggplot2)

tv_shows_data <- tv_shows_data %>%
  mutate(Start_Year = as.numeric(str_extract(Year_Released, "\\b\\d{4}\\b"))) %>%
  filter(!is.na(Start_Year))


ggplot(tv_shows_data, aes(x = Start_Year)) +
  geom_bar(stat = "count", fill = "blue") +
  labs(title = "Number of TV Shows Released by Year",
       x = "Start Year of the TV Show",
       y = "Number of TV Shows") +
  theme_minimal()
```



4. Select 3 products from Amazon of the same category. Extract the price, description, ratings and reviews of each product.

```r
url <- "https://www.amazon.com/SteelSeries-Apex-Hybrid-Mechanical-Gaming-Keyboard/dp/B07ZGDD6B1/ref=sr_

session <- bow(url,
               user_agent = "Educational Purpose")
session
```

```
## <polite session> https://www.amazon.com/SteelSeries-Apex-Hybrid-Mechanical-Gaming-Keyboard/dp/B07ZGD
##     User-agent: Educational Purpose
##     robots.txt: 154 rules are defined for 4 bots
##    Crawl delay: 5 sec
##   The path is scrapable for this user-agent
```

```r
product_name <- scrape(session) %>%
                html_nodes("span.a-size-large.product-title-word-break") %>%
                html_text
```

```
## Error in UseMethod("xml_find_all"): no applicable method for 'xml_find_all' applied to an object of
```

```r
product_name
```

```
## Error in eval(expr, envir, enclos): object 'product_name' not found
```

```r
product_price <- scrape(session) %>%
    html_nodes("div.a-section.a-spacing-none.aok-align-center") %>%
    html_text
```

```
## Error in UseMethod("xml_find_all"): no applicable method for 'xml_find_all' applied to an object of
```

```
product_price
```

```
## Error in eval(expr, envir, enclos): object 'product_price' not found
```

```
product_description <- scrape(session) %>%
    html_nodes("div.a-expander-collapsed-height.a-row.a-expander-container.a-spacing-none.a-expander-pa
    html_text
```

```
## Error in UseMethod("xml_find_all"): no applicable method for 'xml_find_all' applied to an object of
```

```
product_description
```

```
## Error in eval(expr, envir, enclos): object 'product_description' not found
```

```
product_ratings <- scrape(session) %>%
    html_nodes("span.a-size-base.a-color-base") %>%
    html_text
```

```
## Error in UseMethod("xml_find_all"): no applicable method for 'xml_find_all' applied to an object of
```

```
product_ratings
```

```
## Error in eval(expr, envir, enclos): object 'product_ratings' not found
```