Guided Project: Analyzing Movie Ratings

Eda AYDIN

29 05 2021

Contents

Introduction

- Title: Movies' ratings versus user votes
- Usually, we can find a lot of information online about the ranking of movies, universities, supermarkets, etc. We can use these data to supplement information from another database or facilitate trend analysis. However, it's not easy to choose the right criterion because several might be interesting (e.g., movies' ratings and user votes). In this project, we want to extract information on the most popular movies from early 2020 and check if the ratings are in alignment with the votes. If yes, then we can consider either one or the other without loss of information.

Loading the Web Page

```r
# Loading the `rvest`, `dplyr`, and `ggplot2` packages
library(rvest)
```

```
## Warning: package 'rvest' was built under R version 4.0.4
```

```r
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.0.5

##
## 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
```

```r
library(ggplot2)
# Specifying the URL where we will extract video data
url <- "http://dataquestio.github.io/web-scraping-pages/IMDb-DQgp.html"
# Loading the web page content using the `read_html()` function
wp_content <- read_html(url)
```

String Manipulation Reminder

```r
# Converting "10.50" into numeric
as.numeric("10.50")
```

```
## [1] 10.5
```

```r
# Converting the vector `c("14.59", "3.14", "55")` into numeric
as.numeric(c("14.59", "3.14", "55"))
```

```
## [1] 14.59  3.14 55.00
```

```r
# Parsing the vector `c("14 min", "17,35", "(2012)", "1,2,3,4")` into numeric
readr::parse_number(c("14 min", "17,35", "(2012)", "1,2,3,4"))
```

```
## [1]   14 1735 2012 1234
```

```r
# Removing whitespaces at the begining and end of `" Space before and after should disappear     "`
stringr::str_trim(" Space before and after should disappear     ")
```

```
## [1] "Space before and after should disappear"
```

Extracting Elements from the Header

```r
# Extracting the movie's titles
## Finding the title CSS selector
title_selector <- ".lister-item-header a"
## Identifying the number of elements this selector will select from Selector Gadget
n_title <- 30
## Extracting the movie titles combining the `html_nodes()` and `html_text()` function
titles <- wp_content %>%
  html_nodes(title_selector) %>%
  html_text()
## Printing titles vector
titles
```

```
##  [1] "Mulan"
##  [2] "The Call"
##  [3] "Greenland"
##  [4] "Don't Listen"
##  [5] "Unhinged"
##  [6] "Ava"
##  [7] "The Hunt"
##  [8] "Ghosts of War"
##  [9] "Hamilton"
## [10] "The Old Guard"
## [11] "The Secret: Dare to Dream"
## [12] "The Outpost"
## [13] "Extraction"
## [14] "Train to Busan Presents: Peninsula"
## [15] "Greyhound"
## [16] "The King of Staten Island"
## [17] "A Quiet Place Part II"
## [18] "Bloodshot"
## [19] "The Dark and the Wicked"
## [20] "Arkansas"
## [21] "The Rental"
## [22] "Trolls World Tour"
## [23] "Sputnik"
## [24] "Eurovision Song Contest: The Story of Fire Saga"
## [25] "Inheritance"
## [26] "Spenser Confidential"
## [27] "The Tax Collector"
## [28] "The Way Back"
## [29] "The Silencing"
## [30] "Archive"
```

```r
# Extracting the movie's years
## Using a process similar to the one we used to extract the titles
year_selector <- ".lister-item-year"
n_year <- 30
years <- wp_content %>%
  html_nodes(year_selector) %>%
  html_text()
## Converting the years from character to numeric data type
years <- readr::parse_number(years)
```

```r
## Printing years vector
years
```

```
##  [1] 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020
## [16] 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020
```

Extracting Movie's Features

```r
# Extracting the movie's runtimes
## Finding the title CSS selector
runtime_selector <- ".runtime"
## Identifying the number of elements this selector will select from Selector Gadget
n_runtime <- 30
## Extracting the movie runtimes combining the `html_nodes()` and `html_text()` function
runtimes <- wp_content %>%
  html_nodes(runtime_selector) %>%
  html_text()
## Converting the runtimes from character to numeric data type
runtimes <- readr::parse_number(runtimes)
## Printing runtimes vector
runtimes
```

```
##  [1] 115 112 119  97  90  96  90  94 160 125 107 123 116 116  91 136  97 109  95
## [20] 117  88  90 113 123 111 111  95 108  93 109
```

```r
# Extracting the movie's genres
## Extracting the movie genres using a similar process as previously
genre_selector <- ".genre"
n_genre <- 30
genres <- wp_content %>%
  html_nodes(genre_selector) %>%
  html_text()
## Removing whitespaces at the end of genre characters
genres <- stringr::str_trim(genres)
## Printing genres vector
genres
```

```
##  [1] "Action, Adventure, Drama"     "Horror, Mystery, Thriller"
##  [3] "Action, Drama, Thriller"      "Drama, Horror, Thriller"
##  [5] "Action, Thriller"             "Action, Crime, Drama"
##  [7] "Action, Horror, Thriller"     "Horror, Thriller, War"
##  [9] "Biography, Drama, History"    "Action, Adventure, Fantasy"
## [11] "Drama, Romance"               "Action, Drama, History"
## [13] "Action, Thriller"             "Action, Horror, Thriller"
## [15] "Action, Drama, History"       "Comedy, Drama"
## [17] "Drama, Horror, Sci-Fi"        "Action, Drama, Sci-Fi"
## [19] "Horror"                       "Crime, Drama, Thriller"
## [21] "Horror, Thriller"             "Animation, Adventure, Comedy"
## [23] "Drama, Horror, Sci-Fi"        "Comedy, Music"
## [25] "Drama, Mystery, Thriller"     "Action, Comedy, Crime"
## [27] "Action, Crime, Drama"         "Drama, Sport"
## [29] "Action, Crime, Thriller"      "Drama, Sci-Fi, Thriller"
```

Extracting Movie's Ratings

```
# Extracting the movie's user ratings
## Finding the user rating CSS selector
user_rating_selector <- ".ratings-imdb-rating"
## Identifying the number of elements this selector will select from Selector Gadget
n_user_rating <- 29
## Extracting the user rating combining the `html_nodes()` and `html_attr()` function
user_ratings <- wp_content %>%
  html_nodes(user_rating_selector) %>%
  html_attr("data-value")
## Converting the user rating from character to numeric data type
user_ratings <- as.numeric(user_ratings)
## Printing user ratings vector
user_ratings
```

```
##  [1] 5.5 7.2 6.3 6.1 6.0 5.3 6.5 5.5 8.6 6.6 6.4 6.8 6.7 5.4 7.0 7.1 5.7 6.2 5.9
## [20] 5.7 6.1 6.4 6.5 5.5 6.2 4.7 6.7 6.1 6.3
```

```
# Extracting the movie's metascores
## Extracting the movie metascore using a similar process as previously
metascore_selector <- ".metascore"
n_metascore <- 25
metascores <- wp_content %>%
  html_nodes(metascore_selector) %>%
  html_text()
## Removing whitespaces at the end of metascores and converting them into numeric
metascores <- stringr::str_trim(metascores)
metascores <- as.numeric(metascores)
## Printing metascores vector
metascores
```

```
##  [1] 66 40 39 50 38 90 70 32 71 56 51 64 67 44 72 55 63 51 61 50 31 49 22 66 67
```

Extracting Movie's Votes

```
# Extracting the movie's votes
## Finding the vote CSS selector
vote_selector <- ".sort-num_votes-visible :nth-child(2)"
## Identifying the number of elements this selector will select from Selector Gadget
n_vote <- 29
## Extracting the votes combining the `html_nodes()` and `html_text()` function
votes <- wp_content %>%
  html_nodes(vote_selector) %>%
  html_text()
## Converting the vote from character to numeric data type
votes <- readr::parse_number(votes)
## Printing votes vector
votes
```

```
##  [1]  80231   6559  27482   4879  25316  20882  62277   4886  48663 117211
## [11]   5420  17001 151319  16815  60885  29278  57763   3701   7951  13965
## [21]  14792  10516  69186   7837  65768   7416  28300   9355   9271
```

Dealing with missing data

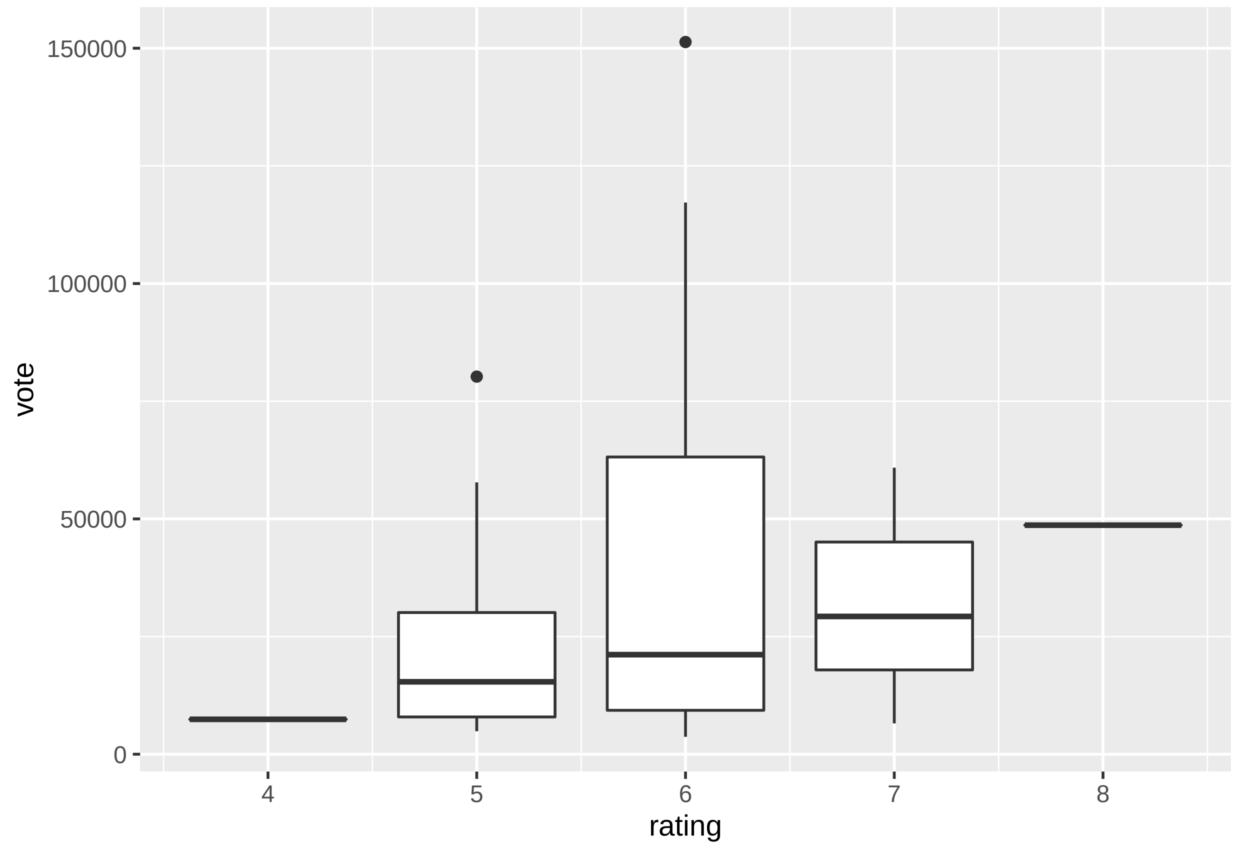
```
# Copy-pasting the `append_vector()` in our Markdown file
append_vector <- function(vector, inserted_indices, values){
  ## Creating the current indices of the vector
  vector_current_indices <- 1:length(vector)
  ## Adding `0.5` to the `inserted_indices`
  new_inserted_indices <- inserted_indices + seq(0, 0.9, length.out = length(inserted_indices))
  ## Appending the `new_inserted_indices` to the current vector indices
  indices <- c(vector_current_indices, new_inserted_indices)
  ## Ordering the indices
  ordered_indices <- order(indices)
  ## Appending the new value to the existing vector
  new_vector <- c(vector, values)
  ## Ordering the new vector wrt the ordered indices
  new_vector[ordered_indices]
}
# Using the `append_vector()` function to insert `NA` into the metascores vector after the positions 1,
metascores <- append_vector(metascores, c(1, 1, 1, 13, 24), NA)
metascores
```

```
##  [1] 66 NA NA NA 40 39 50 38 90 70 32 71 56 51 64 67 NA 44 72 55 63 51 61 50 31
## [26] 49 22 66 NA 67
```

```
# Removing the 17th element from the vectors: titles, years, runtimes, genres, and metascores
## Saving the result back to these vectors.
titles <- titles[-17]
years <- years[-17]
runtimes <- runtimes[-17]
genres <- genres[-17]
metascores <- metascores[-17]
```

Putting all together and Visualize

```
# Creating a dataframe with the data we previously extracted: titles, years, runtimes, genres, user rat
## Keeping only the integer part of the user ratings using the `floor()` function. For example, `3.4` b
movie_df <- tibble::tibble("title" = titles,
                           "year" = years,
                           "runtime" = runtimes,
                           "genre" = genres,
                           "rating" = floor(user_ratings),
                           "metascore" = metascores,
                           "vote" = votes)
# Creating a boxplot that show the number of vote again the user rating
ggplot(data = movie_df,
       aes(x = rating, y = vote, group = rating)) +
  geom_boxplot()
```


vote
rating