# Web Scrapping Project R Markdown

Salman Virani

3/3/2022

This is my first project in the domain of data science. As I am developing skills of using R, I am always open to constructive feedback.

A bit of necessary steps and then lets dive right into web scrapping!

```
sessionInfo()
```

```
## R version 4.1.2 (2021-11-01)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19042)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_Pakistan.1252  LC_CTYPE=English_Pakistan.1252
## [3] LC_MONETARY=English_Pakistan.1252 LC_NUMERIC=C
## [5] LC_TIME=English_Pakistan.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## loaded via a namespace (and not attached):
##  [1] compiler_4.1.2  magrittr_2.0.2  fastmap_1.1.0   cli_3.1.1
##  [5] tools_4.1.2     htmltools_0.5.2 rstudioapi_0.13 yaml_2.2.2
##  [9] stringi_1.7.6   rmarkdown_2.11  knitr_1.37      stringr_1.4.0
## [13] xfun_0.29       digest_0.6.29   rlang_1.0.1     evaluate_0.14
```

This project inovlves scrapping some data for imdb top 250 movies webpage. Our main package to be used here is 'rvest'. We will scrape the data from imdb and create a R data frame from it, so that we are capable of doing some analysis on it.

```
library(rvest)
library(stringr)
library(tibble)
```

For more information on rvest package, try running '?rvest'. For vignettes in this package, try 'vignette(package = "rvest)'.

```
url <- "https://www.imdb.com/chart/top/"

top_movies <- read_html(url)
```

We need R to read our webpage, and that's where 'read_html' from rvest comes in the picture.

### Extract Titles

'html_nodes' allows us to access the tags in the html. '#' refers to the id within the tag, and '.' refers to the class within the tag. 'a' refers to the links within the tag. 'html_node' instead of 'html_nodes' will return only the first instance. Stringr package allows us to do some string manipulations in our data. Lapply function lets us loop through every vector in our 'titles' list and retreive the second element, which is actually the name of the movie. The code extracts the name of the top 250 movies on imdb as of 03/03/2022.

```
titles <- top_movies %>%
  html_nodes("tbody tr td.titleColumn") %>%
  html_text() %>%
  str_trim() %>%
  str_split("\n") %>%
  lapply(function(movie){
  movie[2]
}) %>%
  unlist() %>%
  str_trim()

head(titles)
```

```
## [1] "The Shawshank Redemption" "The Godfather"
## [3] "The Dark Knight"          "The Godfather: Part II"
## [5] "12 Angry Men"             "Schindler's List"
```

### Extract Years

Almost the same code gives us the years of release of the top 250 movies. Instead of retreiving the second element, this time we retrieve the third element. However, the data is a bit messy, we dont want the parentheses and want the years as integers. 'str_replace' and 'as.integer' comes to rescue for this task.

```
years <- top_movies %>%
  html_nodes("tbody tr td.titleColumn") %>%
  html_text() %>%
  str_trim() %>%
  str_split("\n") %>%
  lapply(function(movie){
  movie[3]
}) %>%
  unlist() %>%
  str_trim() %>%
  str_replace("\\(","") %>%
  str_replace("\\)","") %>%
  as.integer()

head(years)
```

```
## [1] 1994 1972 2008 1974 1957 1993
```

### Create Ranks

```
ranks <- 1:250
```

### Extract Ratings

```
ratings <- top_movies %>%
  html_nodes(".imdbRating strong") %>%
  html_text() %>%
  as.numeric()

head(ratings)
```

```
## [1] 9.2 9.2 9.0 9.0 9.0 8.9
```

### Create a dataframe

We have a data frame ready, and now we can run our quantitative analysis the way we want!

```
top_movies_tibble <- tibble(
  Rank = ranks,
  Title = titles,
  Year = years,
  Rating = ratings
)

head(top_movies_tibble)
```

```
## # A tibble: 6 x 4
##    Rank Title                      Year Rating
##   <int> <chr>                     <int>  <dbl>
## 1     1 The Shawshank Redemption  1994    9.2
## 2     2 The Godfather             1972    9.2
## 3     3 The Dark Knight           2008    9
## 4     4 The Godfather: Part II    1974    9
## 5     5 12 Angry Men              1957    9
## 6     6 Schindler's List          1993    8.9
```

### Just a little bit more: Extracting Links

We have already used the 'html_text' function to extract texts. This time we a link from one of the attributes, so we use the 'html_attr' function. The '.' in the 'paste0' function tells R to put whatever has been piped before the function as its argument. The very first links has been outputted, so you can check that it belongs to 'The Shawshank Redemption'.

```
links <- top_movies %>%
  html_node("tbody tr td.titleColumn a") %>%
  html_attr("href") %>%
  paste0("https://www.imdb.com/", .)

links[1]
```

```
## [1] "https://www.imdb.com//title/tt0111161/?pf_rd_m=A2FGELUUNOQJNL&pf_rd_p=1a264172-ae11-42e4-8ef7-7
```