Naruto Shippuden IMDb Web Scraping Codes

Rohan Shinde

8/01/2022

Introduction

We need the episode air dates data, the Director and Writer of the episode as well as the IMDb rating and IMDb votes of each episode of **Naruto:Shippuden**. To do this we refer to the data on these 2 resources:

- Naruto Shippuden List of episodes Wikipedia webpage
- Naruto Shippuden IMDb webpage

Web-scraping the Wikipedia webpage

We first import the first relevant table on the wikipedia webpage whose URL is *(https://en.wikipedia.org/wiki/List_of_Naruto:_Shippuden_episodes)* using the **read_html**, **html_nodes**, and **html_table** functions belonging to the **rvest** library as follows:

```
options(warn = -1)
###### Loading the libraries

library(tidyverse, quietly = T)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.5     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(rvest, quietly = T)
```

```
##
## Attaching package: 'rvest'
```

```
## The following object is masked from 'package:readr':
##
##     guess_encoding
```

```
### Link to the tables
ship_wiki <- "https://en.wikipedia.org/wiki/List_of_Naruto:_Shippuden_episodes"

### Number of episodes in each season of
season_wise <- (read_html(ship_wiki) %>%
                  html_nodes(xpath = '//*[@id="mw-content-text"]/div[1]/table[2]') %>%
```

```
                  html_table())[[1]][-1,3] %>%
  pull() %>%
  rep(1:21, .)


season1 <- (read_html(ship_wiki) %>%
  html_nodes(xpath = '//*[@id="mw-content-text"]/div[1]/table[3]') %>%
  html_table())[[1]]
```

Note that the *season_wise* tibble in the above code pulls out the number of episodes in each season of Naruto:Shippuden. We have used the *xpath* argument in *html_nodes* function to import the concerned table. The *season1* tibble consists of the data on the first season of Naruto Shippuden available on Wikipedia.

This *season1* tibble needs a bit of renaming of columns to be done. Also to form the final dataset of all seasons we first form a copy of *season1* data and then merge that copy with the tibbles obtained from other seasons. This is done as follows:

```
## Renaming columns in season1 data

season1 <- season1 %>%
  mutate(`No. inseason` = No.) %>%
  rename("No.overall" = "No.") %>%
  select("No.overall", `No. inseason`, everything())

##Forming a copy of season1 data

final <- season1
```

We now form a for loop to web-scrape the data in other tables too (the tables for Season 2 to Season 21).

```
for(i in 4:23){
  season_temp <- (read_html(ship_wiki) %>%
    html_nodes(xpath = paste('//*[@id="mw-content-text"]/div[1]/table[',i,']',
                             sep = "")) %>%
    html_table())[[1]]

  final <- bind_rows(final,season_temp) ## To merge all season data into
  Sys.sleep(1)    ## To avoid unncessary strain on the server
}
```

The range of *for* loop has been chosen so that only the required tables on the webpage are scraped.

We now need to clean the obtained data into a more usable form. This is done as follows:

```
final2_wiki <- final %>%

  ## First rename the columns appropriately

  rename("English_air_date" = `English air date<U+200A>[17]`,
         "original_air_date" = `Original air date`,
         "episode_number_overall" = "No.overall",
         "episode_number_in_season" = `No. inseason`) %>%

  ## Form a new column english_air_date that contains the english
  ## episdoe air date of each episode
  mutate(
    english_air_date = ifelse(!is.na(`English air date`), `English air date`,
```

```
                              ifelse(
                                !is.na(English_air_date) & English_air_date != "TBA",
                                English_air_date,NA)) %>%

      ## Text cleaning in the formed column

      str_replace_all(pattern = "\\[.+",replacement = "") %>%

      ## Changing the column data into date format

      lubridate::mdy(),

    ## Creating the original_air_date column, cleaning the text data,
    ## and then converting it into date format

    original_air_date = original_air_date %>%
      str_replace_all("\\(.*","") %>%
      lubridate::mdy(),

    ## Adding another column which represents the season wise episode
    ## number
    season = season_wise) %>%

  ## Removing unnecessary columns

  select(-`English air date`, -English_air_date) %>%

  ## Rearranging the columns

  select(episode_number_overall, season, everything())


## Clearing up the R environment space by removing unrequired tibbles

rm(list=c("season_temp", "season1", "i", "ship_wiki", "final", "season_wise"))
```

This completes the web scraping of Wikipedia webpage. Now we move towards web scraping of the IMDb webpages

Web-scraping of IMDb webpages

The data of ratings on IMDb pages is available on different webpages based on the year the episode details were added on IMDb. The required years are 2009 to 2017 and one episode was recently added to IMDb database in 2021. We first web scrape the ratings, votes, description and episode number of each episode in the year 2009.

```
years <- c(2010:2017,2021)

## Extract the ratimgs of each episode added in 2009

ratings_s1 <- read_html("https://www.imdb.com/title/tt0988824/episodes?year=2009") %>%
  html_nodes(".ipl-rating-star.small .ipl-rating-star__rating" ) %>%
  html_text() %>%
  parse_number()  ## Changing the format to number
```

```
## Extract the votes of each episode added in 2009

votes_s1 <- read_html("https://www.imdb.com/title/tt0988824/episodes?year=2009") %>%
  html_nodes(".ipl-rating-star__total-votes" ) %>%
  html_text() %>%
  parse_number()    ## Changing the format to number

## Extract the description of each episode added in 2009

description <- read_html("https://www.imdb.com/title/tt0988824/episodes?year=2009") %>%
  html_nodes(".item_description" ) %>%
  html_text() %>%
  str_squish()    ## Remove extra spaces

## Extract the episode number of each episode added in 2009

episode <- read_html("https://www.imdb.com/title/tt0988824/episodes?year=2009") %>%
  html_nodes(".zero-z-index") %>%
  html_text() %>%
  str_squish()    ## Remove extra spaces

### Cleaning the above obtained vector

episode <- episode[str_length(episode) > 1] %>%
  str_replace_all("^S1, Ep","") %>%
  parse_number()   ## Changing the format to number
```

We now form a dataset from these above obtained vectors and form a copy of that dataset to merge the data from other years (which we will obtain using a *for* loop later on) to this copy.

```
initial_data <- as_tibble(data.frame(epsode_num = episode,
                                     rating = ratings_s1,
                                     votes = votes_s1,
                                     desc = description))
df <- initial_data
```

We now write the *for* loop:

```
for(i in years){

  ## Noting the link for the data based on the for loop argument

  link <- paste("https://www.imdb.com/title/tt0988824/episodes?year=",i,sep="")

  ## Extracting the ratings of each episode of the year in the for loop

  ratings_temp <- read_html(link) %>%
    html_nodes(".ipl-rating-star.small .ipl-rating-star__rating" ) %>%
    html_text() %>%
    parse_number()  ## Changing the format to number
  Sys.sleep(1)

  ## Extracting the votes of each episode of the year in the for loop
```

```
  votes_temp <- read_html(link) %>%
    html_nodes(".ipl-rating-star__total-votes" ) %>%
    html_text() %>%
    parse_number()  ## Changing the format to number
  Sys.sleep(1)

  ## Extracting the description of each episode of the year in the for
  ## loop

  description_temp <- read_html(link) %>%
    html_nodes(".item_description" ) %>%
    html_text() %>%
    str_squish()    ## Remove extra spaces
  Sys.sleep(1)

  ## Extracting the episode number of each episode of the year in the
  ## for loop

  episode_temp <- read_html(link) %>%
    html_nodes(".zero-z-index") %>%
    html_text() %>%
    str_squish()  ## Remove extra spaces

  ## Cleaning the above vector

  episode_temp <- episode_temp[str_length(episode_temp) > 1] %>%
    str_replace_all("^S1, Ep","") %>%
    parse_number()   ## Changing the format to number


  ## Merging the obtained data from the year in the for loop to the
  ## earlier copy of the data

  df <- bind_rows(df,as_tibble(data.frame(epsode_num = episode_temp,
                                          rating = ratings_temp,
                                          votes = votes_temp,
                                          desc = description_temp)))
  Sys.sleep(3)
}

df <- df %>%
  rename("episode_num" = "epsode_num")
```

Now, the dataset formed above has an anomaly since it has 502 rows but the anime had only 500 episodes. We first check if there are NA's in the *episode_num* column because none of the episode has same episode number nor are the episode numbers less than 1 or greater than 500.

```
df %>%
  filter(!(episode_num %in% 1:500))
```

```
## # A tibble: 2 x 4
##   episode_num rating votes desc
##         <dbl>  <dbl> <dbl> <chr>
## 1          NA    7.9   100 The townspeople are going about their daily business~
```

```
## 2          NA    6.9   144 The townspeople are going about their daily business~
```

Indeed there are two such rows which upon further inspection are the same episodes (which were recorded in the IMDb database as two different episodes) and infact a copy of another episode which is already numbered in the numbered 500 episodes. So we remove these two rows from data.

```
## Removing the rows with NA in episode_num column and renaming the
## desc column

df <- df %>%
  filter(!(is.na(episode_num))) %>%
  arrange(episode_num) %>%
  rename("description" = "desc")

## Making a final copy of the data

final_imdb <- df

## Removing unnecessary objects from R environment

rm(list = c("df", "initial_data", "description",
            "description_temp", "episode", "episode_temp",
            "i", "link", "ratings_s1",
            "ratings_temp", "votes_s1", "votes_temp", "years"))
```

Merging the data obtained from Wikipedia webpage and IMDb webpages

We now merge the data from these two websites to form the final dataset.

```
naruto_ratings_data <- left_join(final2_wiki,final_imdb,
                                 by = c("episode_number_overall" = "episode_num"))
```