Control 2, respuestas

Web Scraping y acceso a datos desde la web con R

Cristián Ayala

Ponderación 20% de la nota final del curso
Formato Desarrollar esta tarea en un Rmarkdown generando un `.pdf`, agregando comentarios cuando sea necesario.

1 Objetivo:

Interesa indagar sobre el cine chileno. Queremos saber la evolución del número de películas chilenas estrenadas por año y su calificación según la nota dada por IMDb.

Para ello usaremos el sitio web IMDb para filtrar películas chilenas realizadas en Chile. En total son **295**[1] según se muestra en esta búsqueda:

https://www.imdb.com/search/title/?title_type=feature&countries=cl&locations=chile

2 Tareas:

2.1 Captura de datos

1) Desde esa página web capturar los siguientes datos de esas 295 películas:

Los objetos están dentro de `<div>` de nombre `#main` con clase `lister-item-content`.

```
* Título: `.lister-item-header a`
* Año de estreno: `.lister-item-header .lister-item-year`
* Puntaje IMDb: `.ratings-imdb-rating strong`
* Géneros: `.genre`
```

Cada página muestra 50 películas y utiliza el parámetro `start=NUMERO` para mostrar desde la película número `NUMERO` las 50 siguientes.

[1] Número de películas al momento de diseñar este control.

```
url_1 <- 'https://www.imdb.com/search/title/?title_type=feature&countries=cl&locations=chile'
url_parse_1 <- parse_url(url_1)

l_pelicula_1 <- read_html(build_url(url_parse_1))
```

¿Cuántas son las películas totales que están presente en la búsqueda?

```
n_peliculas <- l_pelicula_1 |>
  html_elements('.nav div.desc') |>
  html_text2()

n_peliculas <- n_peliculas |>
  str_extract('(?<= of )\\d+') |>
  as.integer()

n_peliculas
```

```
[1] 296
```

Creamos ahora los intervalos de búsqueda

```
n_pel_por_pagina <- 50

intervalos <- seq(1,
                  ceiling(n_peliculas/n_pel_por_pagina) * n_pel_por_pagina,
                  n_pel_por_pagina)

intervalos
```

```
[1]   1  51 101 151 201 251
```

Saco la página 1 porque ya la tengo capturada

```
intervalos <- intervalos[-1]
```

Construyo los links para cada una de las páginas de búsqueda.

```
querys <- map(intervalos, ~c(url_parse_1$query, 'start' = .))

f_urls <- function(.query){
  url_parse_1['query'] <- list(.query)

  url_parse_1 |>
    build_url()
}
```

```
l_urls <- map(querys, f_urls)

l_urls
```

```
[[1]]
[1] "https://www.imdb.com/search/title/?title_type=feature&countries=cl&locations=chile&start=51"

[[2]]
[1] "https://www.imdb.com/search/title/?title_type=feature&countries=cl&locations=chile&start=101"

[[3]]
[1] "https://www.imdb.com/search/title/?title_type=feature&countries=cl&locations=chile&start=151"

[[4]]
[1] "https://www.imdb.com/search/title/?title_type=feature&countries=cl&locations=chile&start=201"

[[5]]
[1] "https://www.imdb.com/search/title/?title_type=feature&countries=cl&locations=chile&start=251"
```

Lectura de cada hoja

```
l_peliculas <- map(l_urls, read_html)
```

Agrego la primera hoja ya capturada. Tengo un total de 6 hojas

```
l_peliculas <- append(list(l_pelicula_1), l_peliculas)

length(l_peliculas)
```

```
[1] 6
```

Selección de datos de interés

```
# Obtener lista de nodo de películas
selectores <- c(index = '.lister-item-index',
                titulo = '.lister-item-header a',
                anio = '.lister-item-header .lister-item-year',
                ranking = '.ratings-imdb-rating strong',
                genero = '.genre')

f_capturar_elementos <- function(.html, .selector, .names_sel){

  links <- NULL # Objeto solo para links en el caso de estar capturando el título
```

```
  html <- .html |>
    html_elements('#main .lister-item-content')

  # Captura general del elemento de interés.
  data <- html |>
    html_element(.selector) |>
    html_text() |>
    str_squish()

  # Captura de link a la película solo si estoy viendo elemento nominado título
  if (.names_sel == 'titulo'){
    links <- html |>
      html_element(.selector) |>
      html_attr('href')
  }

  # Devuelvo los datos capturados: un vector con texto y links.
  setNames(list(data, links),
           nm = c(.names_sel, 'link'))
}

# Itero todos los selectores en todas las páginas de películas que capturamos

df_paliculas <- map(l_peliculas,
                    function(l_pel){
                      map2(selectores, names(selectores),
                          function(selector, names_sel){
                            f_capturar_elementos(l_pel, selector, names_sel)
                          }
                      )
                    }
)

df_paliculas_fin <- map_df(df_paliculas, flatten_dfc)

df_paliculas_fin |> dim()
```

```
[1] 296   6
```

2) Guardar esa información en un data.frame

```
head(df_paliculas_fin)
```

```
# A tibble: 6 x 6
  index titulo                      link                  anio  ranking genero
  <chr> <chr>                       <chr>                 <chr> <chr>   <chr>
```

```
1 1.    Knock Knock: Seducción Fatal /title/tt3605418/?ref~ (I) ~ 4.9     Drama~
2 2.    Diarios de motocicleta       /title/tt0318462/?ref~ (200~ 7.8     Adven~
3 3.    Los 33                       /title/tt2006295/?ref~ (201~ 6.8     Biogr~
4 4.    El Príncipe                  /title/tt7945236/?ref~ (201~ 6.4     Drama
5 5.    Una Mujer Fantástica         /title/tt5639354/?ref~ (201~ 7.2     Drama
6 6.    Ema                          /title/tt8800266/?ref~ (201~ 6.8     Drama~
```

Corregiremos alguna de las variables extraídas para el análisis siguiente.

```r
df_paliculas_fin <- df_paliculas_fin |>
  mutate(
    # Remover punto final en index
    index = str_remove(index, '\\.'),
    # Extraer los números de la variable anio
    anio = str_extract(anio, '\\d+'),
    # Separar un solo string de género en distintas palabras
    genero = str_split(genero, ', ?')
    )

df_paliculas_fin <- df_paliculas_fin |>
  mutate(across(c(index), as.integer),
         across(c(ranking), as.double),
         anio = as.Date(paste0(anio, '-01-01', '%Y-%M-$d')))

head(df_paliculas_fin)
```

```
# A tibble: 6 x 6
  index titulo                       link              anio       ranking genero
  <int> <chr>                        <chr>             <date>       <dbl> <list>
1     1 Knock Knock: Seducción Fatal /title/tt3605418~ 2015-01-01     4.9 <chr>
2     2 Diarios de motocicleta       /title/tt0318462~ 2004-01-01     7.8 <chr>
3     3 Los 33                       /title/tt2006295~ 2015-01-01     6.8 <chr>
4     4 El Príncipe                  /title/tt7945236~ 2019-01-01     6.4 <chr>
5     5 Una Mujer Fantástica         /title/tt5639354~ 2017-01-01     7.2 <chr>
6     6 Ema                          /title/tt8800266~ 2019-01-01     6.8 <chr>
```

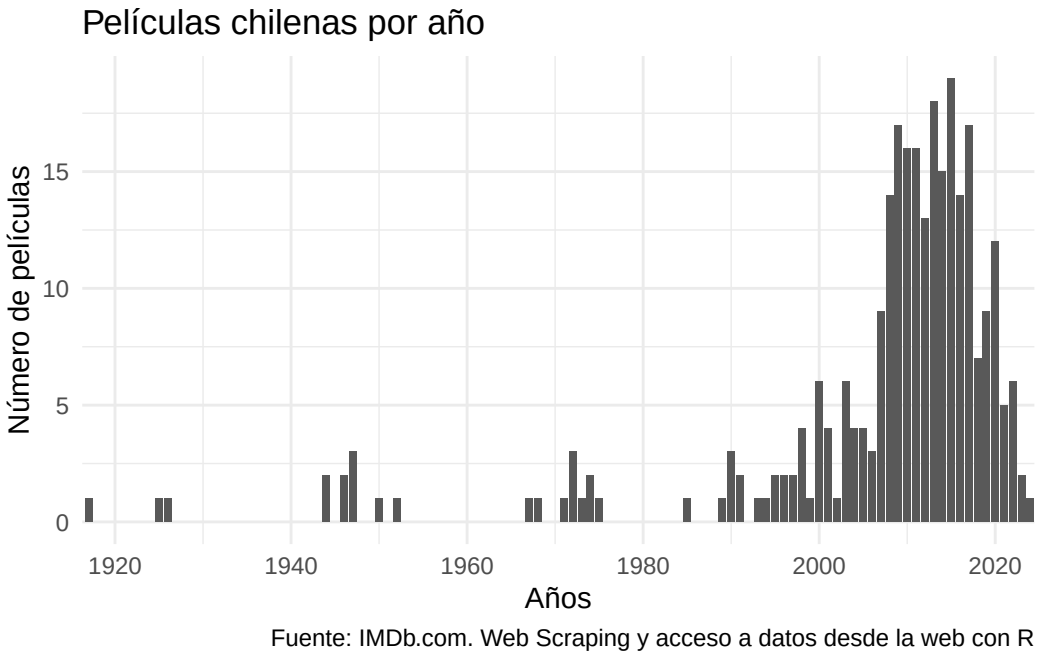
2.2 Análisis

3) Graficar la evolución del el *número de películas* (eje y) estrenadas por *año* (eje x).

```r
df_peliculas_anio <- df_paliculas_fin |>
  count(anio, name = 'n_peliculas')

df_peliculas_anio |>
  ggplot(aes(x = anio, y = n_peliculas)) +
  geom_col() +
```

```
  scale_x_date('Años', expand = expansion(add = c(100, 0))) +
  labs(title = 'Películas chilenas por año',
       caption = 'Fuente: IMDb.com. Web Scraping y acceso a datos desde la web con R',
       y = 'Número de películas') +
  theme_minimal()
```

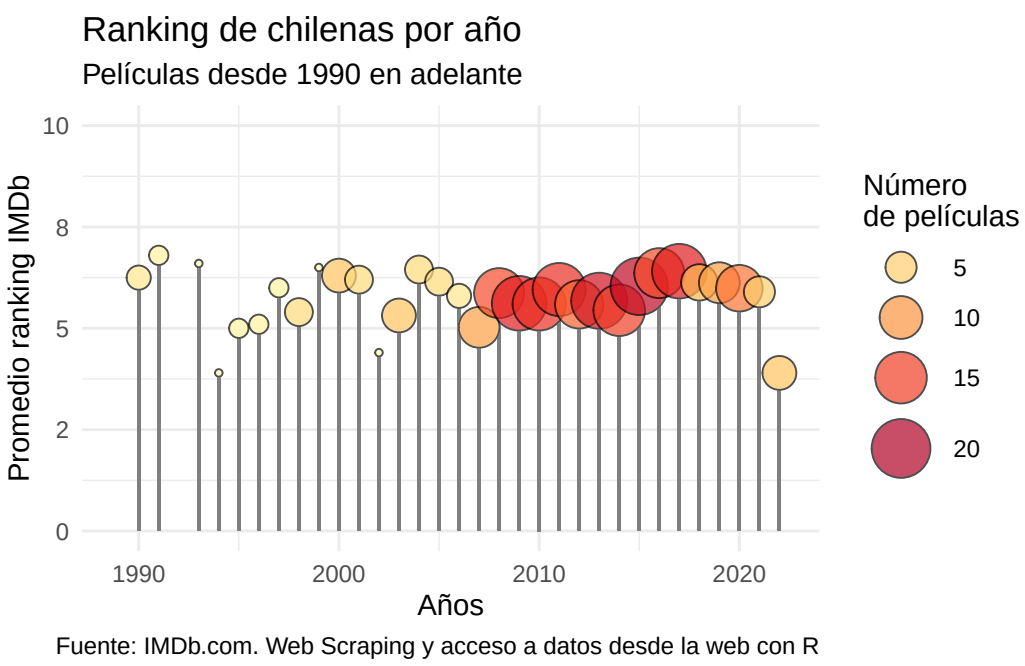


5) Graficar la evolución del el *ranking IMDb* promedio (eje y) *estrenadas desde 1990* a la fecha (eje x).

```
df_peliculas_rank <- df_paliculas_fin |>
  filter(anio >= as.Date("1990-01-01")) |>
  group_by(anio) |>
  summarise(n_peliculas = n(),
            ranking = mean(ranking, na.rm = TRUE))


df_peliculas_rank |>
  ggplot(aes(x = anio, y = ranking,
             fill = n_peliculas,
             size = n_peliculas)) +
  geom_col(width = 60, fill = 'gray50',
           show.legend = F) +
  geom_point(colour = 'white') +
  geom_point(shape = 21,
```

```
            alpha = .7) +
scale_x_date('Años', expand = expansion(add = c(1000, 0))) +
scale_y_continuous(limits = c(0, 10),
                   labels = round) +
scale_fill_distiller('Número\nde películas',
                     palette = 'YlOrRd',
                     direction = 1,
                     limits = c(1, 20),
                     breaks = scales::pretty_breaks(4)) +
scale_size_continuous('Número\nde películas',
                      range = c(1, 10),
                      limits = c(1, 20),
                      breaks = scales::pretty_breaks(4)) +
guides(fill = guide_legend(),
       size = guide_legend()) +
labs(title = 'Ranking de chilenas por año',
     subtitle = 'Películas desde 1990 en adelante',
     caption = 'Fuente: IMDb.com. Web Scraping y acceso a datos desde la web con R',
     y = 'Promedio ranking IMDb') +
theme_minimal()
```



6) ¿Cuál es el *género* que tienen el *mejor puntaje promedio* considerando películas estrenadas desde 1990 a la fecha?

Modificar base para que la unidad de análisis sea `genero`.

```r
df_genero <- df_paliculas_fin |>
  filter(anio >= as.Date("1990-01-01")) |>
  select(index, ranking, genero) |>
  unnest_longer(col = genero)

df_genero <- df_genero |>
  group_by(genero) |>
  summarise(n_peliculas = n(),
            ranking = mean(ranking, na.rm = TRUE)) |>
  arrange(-ranking)

head(df_genero)
```

```
# A tibble: 6 x 3
  genero    n_peliculas ranking
  <chr>           <int>   <dbl>
1 Music               5    7.32
2 Biography          11    7.19
3 Animation           3    7.1
4 History             7    6.97
5 Drama             164    6.16
6 Romance            27    6.02
```

El género con mejor puntaje promedio desde 1990 es **musical**.