MATH 118: Notes O

Webscraping Tables with `rvest`

Emily Malcolm-White

```r
#LOAD PACKAGES
library(tidyverse)
```

Data doesn't just magically appear on your computer you need to get it from somewhere.

Often times, we download data (.csv files or other) and save it locally on our computer.

Other times, we download it from R packages (like we did with the gapminder dataset).

Obtaining Data From The Web

For example, maybe we are interested in renting an apartment or house in Vermont (or studying the rental market in Vermont). You might navigate to Craigslist to get some information: https://vermont.craigslist.org/search/apa

We could spend many hours writing down and creating a spreadsheet with the information about each available apartment… or…

When you enter a URL into your browser, your browser connects to the web server at that URL and asks for the *source code* for that website. We can view the source code in a web brower by clicking on *view source.*

Web scraping is a process by which we can use R (or other software) to systematically go through the source code to extract content and data.

STOP: Are you allowed to scrape that website?

Before scraping data from the web, you should always check whether or not your are *allowed* to scrape it. There are two places you can look: the `robots.txt` file and the Terms of Service Document.

In the Craigslist terms of service document, we find the following text *"You agree not to copy/collect CL content via robots, spiders, scripts, scrapers, crawlers, or any automated or manual equivalent (e.g. by hand).

Wikipedia on the other hand, doesn't explicit state that web scraping is disallowed.

Web Scraping

We need the package `rvest` to help us with this.

```
library(rvest)
```

Webscraping Tables from Wikipedia

```
youtube_videos <- read_html("https://en.wikipedia.org/wiki/List_of_most-viewed_YouTube_vid
  html_element(".wikitable") %>%
  html_table()

youtube_videos
```

```
# A tibble: 31 x 7
   No.   `Video name` Uploader `Views (billions)` `Publication date` Note  ``
   <chr> <chr>        <chr>    <chr>              <chr>              <chr> <chr>
 1 1.    "\"Baby Sha~ Pinkfon~ 13.18              June 17, 2016      "[A]" <NA>
 2 2.    "\"Despacit~ Luis Fo~ 8.23               January 12, 2017   "[B]" <NA>
 3 3.    "\"Johny Jo~ LooLoo ~ 6.76               October 8, 2016    ""    <NA>
 4 4.    "\"Bath Son~ Cocomel~ 6.33               May 2, 2018        ""    <NA>
 5 5.    "\"Shape of~ Ed Shee~ 6.05               January 30, 2017   "[C]" <NA>
 6 6.    "\"See You ~ Wiz Kha~ 5.98               April 6, 2015      "[D]" <NA>
 7 8.    "\"Wheels o~ Cocomel~ 5.46               May 24, 2018       ""    <NA>
 8 7.    "\"Phonics ~ ChuChu ~ 5.42               March 6, 2014      ""    <NA>
 9 9.    "\"Uptown F~ Mark Ro~ 4.99               November 19, 2014  ""    <NA>
10 10.   "\"Learning~ Miroshk~ 4.94               February 27, 2018  ""    <NA>
# i 21 more rows
```

Web scraping doesn't always format perfectly. Let's clean it up!

```r
library(janitor)
```

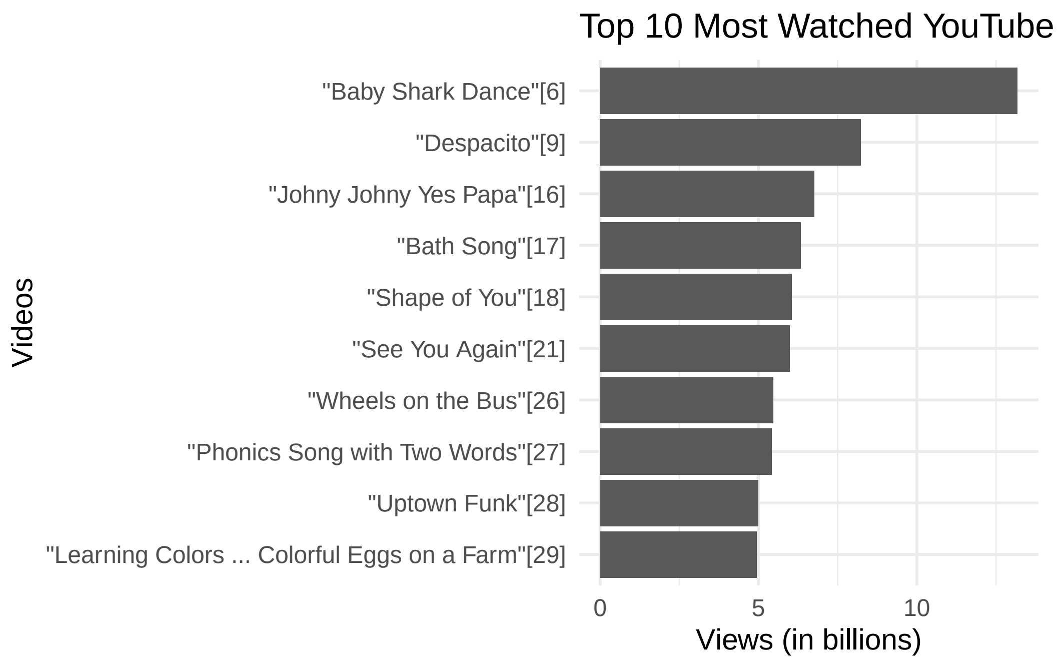
Figure 1: Artwork by @allisonhorst

```r
youtube_videos <- clean_names(youtube_videos)

youtube_videos$views_billions <- as.numeric(youtube_videos$views_billions)

top10 <- youtube_videos %>%
  arrange(desc(views_billions)) %>%
  slice(1:10)

top10
```

```
# A tibble: 10 x 7
   no    video_name          uploader views_billions publication_date note  x
   <chr> <chr>               <chr>             <dbl> <chr>            <chr> <chr>
 1 1.    "\"Baby Shark Dan~ Pinkfon~          13.2  June 17, 2016    "[A]" <NA>
 2 2.    "\"Despacito\"[9]" Luis Fo~           8.23 January 12, 2017 "[B]" <NA>
 3 3.    "\"Johny Johny Ye~ LooLoo ~           6.76 October 8, 2016  ""    <NA>
 4 4.    "\"Bath Song\"[17~ Cocomel~           6.33 May 2, 2018      ""    <NA>
 5 5.    "\"Shape of You\"~ Ed Shee~           6.05 January 30, 2017 "[C]" <NA>
 6 6.    "\"See You Again\~ Wiz Kha~           5.98 April 6, 2015    "[D]" <NA>
 7 8.    "\"Wheels on the ~ Cocomel~           5.46 May 24, 2018     ""    <NA>
 8 7.    "\"Phonics Song w~ ChuChu ~           5.42 March 6, 2014    ""    <NA>
 9 9.    "\"Uptown Funk\"[~ Mark Ro~           4.99 November 19, 20~ ""    <NA>
10 10.   "\"Learning Color~ Miroshk~           4.94 February 27, 20~ ""    <NA>
```

Once we have this data, we can make cool plots!

```r
ggplot(top10, aes(x=views_billions, y=reorder(video_name, views_billions))) +
  geom_bar(stat="identity") +
  xlab("Views (in billions)") +
  ylab("Videos") +
  ggtitle("Top 10 Most Watched YouTube Videos of All Time") +
  theme_minimal()
```


Top 10 Most Watched YouTube
"Baby Shark Dance"[6]
"Despacito"[9]
"Johny Johny Yes Papa"[16]
"Bath Song"[17]
"Shape of You"[18]
"See You Again"[21]
"Wheels on the Bus"[26]
"Phonics Song with Two Words"[27]
"Uptown Funk"[28]
"Learning Colors ... Colorful Eggs on a Farm"[29]
Videos
Views (in billions)