

# Computational Social Science

## Scraping the web II

Dr. Thomas Davidson

Rutgers University

February 16, 2022

# Plan

1. How to scrape a website in R, part II
2. Crawling websites using R
3. selenium and browser automation
4. Next week

# How to scrape a web page

## Using rvest to scrape HTML

```
library(rvest)
library(tidyverse)
library(stringr)
library(lubridate)
```

# How to scrape a web page

The screenshot shows a web browser displaying the 'thecatsite' forum. The URL in the address bar is <https://thecatsite.com/threads/advice-on-cat-introductions-feeling-a-bit-lost.422848/>. The forum header includes navigation links for 'CAT ARTICLES', 'FORUMS', 'MEDIA', and 'MEMBERS', along with buttons for 'BROWSE BY CAT TOPICS', 'CONNECT WITH US', 'LOG IN', 'REGISTER', and 'SEARCH'. A yellow banner promotes joining the community to reduce ads by 90%.

The thread title is 'Advice on Cat Introductions - Feeling a Bit Lost' by user 'Fumama22', dated Dec 22, 2020. The thread starter's profile shows they joined on Dec 22, 2020, have 19 messages, and a reaction score of 12.

The first post, dated Dec 22, 2020, at 11:01 AM, contains the following text:

Hi all,

I'm new to the forum and have been reading all of your excellent thoughts and posts on cat introductions. Our new cat is Florence, a 4 year old spayed female from the local humane society. Our resident cat is Howthorne, a 10 1/2 year old neutered male. Howthorne has always been an anxious and not particularly cuddly cat (he has moments, but is NOT a lap cat - he is still scared of my stepkids after four years) and his friend and my beloved cat Timmyson passed away in August. They were not bonded, and fought occasionally, but generally speaking got along and did friendly nose touches.

We have had Florence for about 6 weeks. We lost one week (week #2) of introduction time because she was very sick with a URI (she is healthy now). She is declawed (something we found out after we knew we wanted her) and was surrendered by her family for "not getting along with their kids." They did not provide any additional information. From my perspective, she's one of the sweetest cats I've ever known. She has spirit but she is quite confident now and good-natured.

The right sidebar features a 'LATEST POSTS' section with the following entries:

- Happy New Year!**  
Latest: lozlie - 2 minutes ago  
The Cat Lounge
- Adult Cats Fighting - Play, Territory, What to Do?**  
Latest: cat nap - 9 minutes ago  
Cat Behavior
- Thread Devoted To Sharing Your Favorite Albums And Songs - 2020**  
Latest: rubysnana - 13 minutes ago  
The Cat Lounge
- post funny picture and memes here**  
Latest: WhenaHellbreaksfree - 14 minutes ago  
Fur Pictures and Video Only!
- Growing a Human - it's a gift!**  
Latest: PhotoBee - 17 minutes ago

# How to scrape a web page

## Using rvest to scrape HTML

We used rvest to read in this URL.

```
url <- "https://thecatsite.com/threads/advice-on-cat-introductions-feel  
thread <- read_html(url)
```

# How to scrape a web page

## Creating a function to collect and store data

```
get.posts <- function(thread) {  
  messages <- thread %>% html_nodes(".message-body") %>%  
    html_text() %>% str_trim()  
  users <- thread %>% html_nodes(".message-userDetails") %>%  
    html_text() %>% str_trim() %>% str_split('\n') %>% map(1)  
  timestamps <- thread %>% html_nodes(".u-concealed .u-dt") %>%  
    html_attr("datetime") %>% ymd_hms(tz="EST")  
  timestamps <- timestamps[-1] # remove first timestamp  
  data <- as_tibble(cbind(messages, unlist(users), timestamps))  
  colnames(data) <- c("message", "user", "timestamp")  
  return(data)  
}
```

# How to scrape a web page

## Using the function

We then used this function to extract information from the forum.

```
results <- get.posts(thread)
tail(results)
```

```
## # A tibble: 6 x 3
```

```
##   message
```

```
##   <chr>
```

```
## 1 "That's great, thank you \n pearl99\n! Our five minute visual~ Fur
```

```
## 2 "Furmama22 said:\n\n\n\nThanks so much for commenting \nA\n A~ Art
```

```
## 3 "Thank you for your perspective! I'll keep that in mind."      Fur
```

```
## 4 "Furmama22 said:\n\n\n\n\nThat's great, thank you \n pearl99\n!~ pea
```

```
## 5 "Furmama22 said:\n\n\n\n\nThank you \nC\n calicosrspecial\nand ~ cal
```

```
## 6 "Furmama22 said:\n\n\n\n\nWhen he does need to go in the room, ~ Mam
```

# How to scrape a web page

## Pagination

The next step is to figure out how we can navigate the different pages of the thread. Inspection of the HTML shows the CSS element `pageNav-jump` contains the relevant information.

```
thread %>% html_nodes(".pageNav-jump")
```

```
## {xml_nodeset (2)}
```

```
## [1] <a href="/threads/advice-on-cat-introductions-feeling-a-bit-lost
```

```
## [2] <a href="/threads/advice-on-cat-introductions-feeling-a-bit-lost
```



# How to scrape a web page

## Pagination

In this case I want both the links *and* the descriptions.

```
links <- thread %>% html_nodes(".pageNav-jump") %>%  
  html_attr("href")  
desc <- thread %>% html_nodes(".pageNav-jump") %>%  
  html_text()  
pagination.info <- data.frame(links, desc) %>%  
  filter(str_detect(desc, "Next")) %>% distinct()  
head(pagination.info)
```

```
##
```

```
## 1 /threads/advice-on-cat-introductions-feeling-a-bit-lost.422848/pag
```

# How to scrape a web page

## Pagination

We can then use the base URL to get the link to the next page.

```
base <- "https://thecatsite.com"
next.page <- paste(base, pagination.info$links, sep = '')
print(next.page)

## [1] "https://thecatsite.com/threads/advice-on-cat-introductions-feel"
```

# How to scrape a web page

## Pagination

Let's verify this works by using the `get.posts` function.

```
results <- get.posts(read_html(next.page))
results[1:5,]
```

```
## # A tibble: 5 x 3
```

```
##   message
```

```
##   <chr>
```

```
## 1 "Thank you all for responding! Merry Christmas to all of you!~ Fur
```

```
## 2 "Sounds like a reason to be merry to me!" Mam
```

```
## 3 "Well I suppose it's always one step forward two steps back. ~ Fur
```

```
## 4 "AWWWWWWWW! She is adorable! And that really wasn't even a w~ Mam
```

```
## 5 "Thank you!" Fur
```

# How to scrape a web page

## Pagination function

```
get.next.page <- function(thread){  
  links <- thread %>% html_nodes(".pageNav-jump") %>%  
    html_attr("href")  
  desc <- thread %>% html_nodes(".pageNav-jump") %>%  
    html_text()  
  pagination.info <- data.frame(links, desc) %>%  
    filter(str_detect(desc, "Next")) %>% distinct()  
  base <- "https://thecatsite.com"  
  next.page <- paste(base, pagination.info$links, sep = '')  
  return(next.page)  
}  
get.next.page(thread)  
  
## [1] "https://thecatsite.com/threads/advice-on-cat-introductions-feel"
```

# How to scrape a web page

## Testing the pagination function

We can easily use this function to paginate. In this case we use `get.next.page` to get the link to page 2, read the HTML for page 2, then use `get.next.page` to extract the link to page 3.

```
thread.2 <- read_html(get.next.page(thread))  
page.3 <- get.next.page(thread.2)  
page.3
```

```
## [1] "https://thecatsite.com/threads/advice-on-cat-introductions-feel"
```

# How to scrape a web page

## Testing the pagination function

What happens when we run out of pages? In this case there is no link to the next page. The `get.next.page` function does not produce an error, but only returns the base URL.

```
get.next.page(read_html("https://thecatsite.com/threads/advice-on-cat-i  
## [1] "https://thecatsite.com/threads/advice-on-cat-introductions-feel
```

# How to scrape a web page

## Improving the function

```
get.next.page <- function(thread){
  links <- thread %>% html_nodes(".pageNav-jump") %>%
    html_attr("href")
  desc <- thread %>% html_nodes(".pageNav-jump") %>%
    html_text()
  pagination.info <- data.frame(links, desc) %>%
    filter(str_detect(desc, "Next")) %>% distinct()
  if (dim(pagination.info)[1] == 1) {
    base <- "https://thecatsite.com"
    next.page <- paste(base, pagination.info$links, sep = '')
    return(next.page)
  } else {
    return("Final page")
  }
}
```

# How to scrape a web page

## Testing the pagination function

We now get this message when we try to paginate on the final page.

```
get.next.page(read_html("https://thecatsite.com/threads/advice-on-cat-i  
## [1] "https://thecatsite.com/threads/advice-on-cat-introductions-feel
```



# How to scrape a web page

## Paginate and scrape

```
paginate.and.scrape <- function(url){  
  thread <- read_html(url)  
  posts <- get.posts(thread)  
  next.page <- get.next.page(thread)  
  while (!str_detect(next.page, "Final page"))  
  {  
    thread <- read_html(next.page)  
    posts <- rbind(posts, get.posts(thread))  
    next.page <- get.next.page(thread)  
    Sys.sleep(1) # wait 1 second  
  }  
  return(posts)  
}
```

# How to scrape a web page

## Paginate and scrape

```
full.thread <- paginate.and.scrape(url)
dim(full.thread)
```

```
## [1] 545    3
```

```
print(head(full.thread))
```

```
## # A tibble: 6 x 3
```

```
##   message
```

```
##   <chr>
```

```
## 1 "Hi all,\nI'm new to the forum and have been reading all of y~ Fur
```

```
## 2 "Furmama22 said:\n\n\nHi all,\nI'm new to the forum and hav~ cal
```

```
## 3 "Thank you SO much for taking the time to reply. I really rea~ Fur
```

```
## 4 "I don't think I can add a thing to \nC\n calicosrspecial\n's~ Mam
```

```
## 5 "Thanks so much for your thoughts and comments! And thank you~ Fur
```

```
## 6 "You certainly came to the right place! And, in my experience~ Mam
```

# How to scrape a web page

## Crawling a website

- ▶ Now we have a function we can use to paginate and scrape the data from threads on the website
- ▶ The next goal is to write a crawler to traverse the website and retrieve information from all of the threads we are interested in.
- ▶ Fortunately, these threads are organized in a similar way
  - ▶ Each page contains 20 threads and links to the next page

# How to scrape a web page

## Crawling a website

```
get.threads <- function(url) {  
  f <- read_html(url)  
  title <- f %>% html_nodes(".structItem-title") %>%  
    html_text() %>% str_trim()  
  link <- f %>% html_nodes(".structItem-title a") %>%  
    html_attr("href") %>% str_trim()  
  link <- data.frame(link)  
  link <- link %>% filter(str_detect(link, "/threads/"))  
  threads <- data.frame(title, link)  
  return(threads)  
}
```

# How to scrape a web page

## Crawling a website

```
forum.url <- "https://thecatsite.com/forums/cat-behavior.5/"  
  
threads <- get.threads(forum.url)
```

# How to scrape a web page

## Crawling a website

```
print(threads$title)
```

```
## [1] "And so it begins... new kitty comes home and her highness is n
## [2] "Litter box help"
## [3] "Onset of aggression with neutering"
## [4] "Time for a behaviorist to help with our feral cats?"
## [5] "Leaving for 3 nights will be back midday on the 4th day, I hav
## [6] "Angry Brother Cats"
## [7] "New Kitty & some mystery bathroom issues"
## [8] "How Do I Get My Cat Used To Being Indoors?"
## [9] "Help/Advice for introductions"
## [10] "Need guidance on cat introductions"
## [11] "Eating behavior."
## [12] "Passive Aggressive Jerk Cat"
## [13] "My cat randomly peed on the carpet??"
## [14] "Struggling With Play Time"
## [15] "Kitten Introduction"
## [16] "Bonded Pair Bullying"
```

# How to scrape a web page

## Crawling a website

```
print(threads$link)
```

```
## [1] "/threads/and-so-it-begins-new-kitty-comes-home-and-her-highnes
## [2] "/threads/litter-box-help.440602/"
## [3] "/threads/onset-of-aggression-with-neutering.440564/"
## [4] "/threads/time-for-a-behaviorist-to-help-with-our-feral-cats.44
## [5] "/threads/leaving-for-3-nights-will-be-back-midday-on-the-4th-d
## [6] "/threads/angry-brother-cats.440576/"
## [7] "/threads/new-kitty-some-mystery-bathroom-issues.440370/"
## [8] "/threads/how-do-i-get-my-cat-used-to-being-indoors.440571/"
## [9] "/threads/help-advice-for-introductions.428629/"
## [10] "/threads/need-guidance-on-cat-introductions.437781/"
## [11] "/threads/eating-behavior.440511/"
## [12] "/threads/passive-aggressive-jerk-cat.440091/"
## [13] "/threads/my-cat-randomly-peed-on-the-carpet.440548/"
## [14] "/threads/struggling-with-play-time.339851/"
## [15] "/threads/kitten-introduction.440319/"
## [16] "/threads/bonded-pair-bullying.440523/"
```

# How to scrape a web page

## Crawling a website

**Exercise:** Write code to iterate over the first 5 pages of threads. You will need to use `get.threads`, `paginate.and.scrape`, and `get.next.page`. Store the results as a tibble in an object called `results`. Make sure to also retain the name of each thread. *Note that this may take a while to run. You should test it on a small subset to verify it works.*

```
# Complete code here
P <- 2
url <- forum.url
results <- tibble()

for (p in 1:P) {
  threads <- get.threads(url)
  for (t in 1:2) {
    page.url <- paste(base, threads$link[t], sep = '')
    new.results <- paginate.and.scrape(page.url)
    new.results$threads <- threads$title[t]
    results <- bind_rows(results, new.results)
  }
}
```



# How to scrape a web page

## Storing the results

The results should consist of a few thousand messages and associated metadata. Save the results of this crawl to as a CSV file.

```
library(readr)
write_csv(results, "cat_crawl.csv")
```

# How to scrape a web page

## Data storage

- ▶ If you try to collect all the data you need before saving it, you run the risk of data loss if your script crashes
  - ▶ This risk increases as you collect more data
    - ▶ More memory on your computer is being used
    - ▶ Increased likelihood of encountering anomalies that cause errors
- ▶ Reasonable solutions
  - ▶ Continuously save results to disk (e.g. concatenate each thread to a CSV)
  - ▶ Store results in chunks (e.g. each thread in a new CSV)

# How to scrape a web page

## Data storage

- ▶ A more robust solution
  - ▶ Write output to a relational database
    - ▶ This helps to organize the data and makes it easier to query and manage, particularly with large datasets
    - ▶ I recommend PostgreSQL, a free open-source, SQL-compatible relational database

# How to scrape a web page

## Data storage

- ▶ If collecting a lot of data, I recommend use a server to run the code and to store scraped data
- ▶ Requirements
  - ▶ Access to a server (\$)
    - ▶ But most universities have free computing clusters
  - ▶ Command line knowledge
  - ▶ Database knowledge
- ▶ It is beyond the scope of this class to cover this material, but I highly recommend you develop this infrastructure if you continue to work in this area

# How to scrape a web page

## Logging

- ▶ Log the progress of your webscraper
  - ▶ Simple option:
    - ▶ Print statements in code
  - ▶ Better option:
    - ▶ Use a log file
  - ▶ To keep yourself updated:
    - ▶ Use a Slack App to send yourself messages

# How to scrape a web page

## Javascript and browser automation

- ▶ Many websites use Javascript, which cause problems for web-scrapers as it cannot directly be parsed to HTML
- ▶ We can get around this by doing the following
  - ▶ Automatically to open a browser (e.g. Google Chrome)
  - ▶ Load a website in the browser
  - ▶ Read the HTML from the browser into R
- ▶ We can also use browser automation to click buttons, fill in forms, or enter login info

# How to scrape a web page

## Selenium

- ▶ Selenium WebDriver and the package RSelenium (<https://github.com/ropensci/RSelenium>) is the most popular approach
- ▶ **However**, RSelenium requires a complicated set up using a Docker container
  - ▶ This is a little technical and I've had trouble getting it to work
  - ▶ It may be easier to use selenium in Python then read the data into R
    - ▶ <https://python-bloggers.com/2020/07/rvspython-3-setting-up-selenium-limitations-with-the-rselenium-package-getting-past-them/>

# How to scrape a web page

## Using reticulate to run selenium in Python

This Python code uses selenium to open up a Chrome browser, visit a website, and collect the HTML. It then closes the browser.

```
from selenium import webdriver
driver = webdriver.Chrome()
driver.get('https://www.sociology.rutgers.edu')
html = driver.page_source
driver.close()
```

This will only work if the Chrome driver has been downloaded and is in your PATH. See <https://chromedriver.chromium.org/getting-started>



# How to scrape a web page

## Using reticulate to run selenium in Python

I saved the code in the previous chunk as a file called `get_html.py`. We can use `reticulate` to run the Python code then pass objects from Python to R. In this case we use Python to run selenium and get the HTML, then read it into R using `rvest`.

```
library(reticulate)

#py_install("selenium") # uncomment to install selenium.py
source_python('../code/get_html.py') # run python script

html.text <- read_html(py$html) %>% html_text()
```

# How to scrape a web page

## Inspecting the results in R

Reticulate allowed us to run a Python script then pass the results to R. We can then use the same commands as above to process it.

```
print(substr(html.text, 1, 35))
```

## Next week

- ▶ Online surveys and experiments
- ▶ Intro to RShiny
- ▶ Homework 2 released