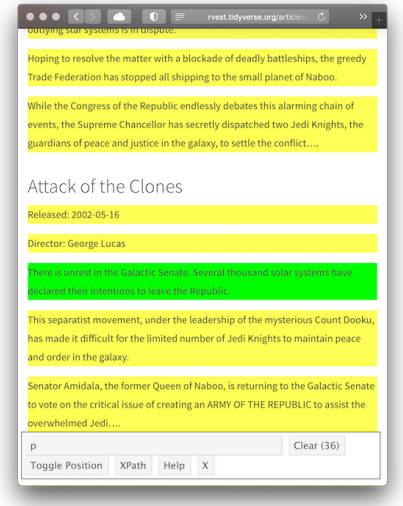
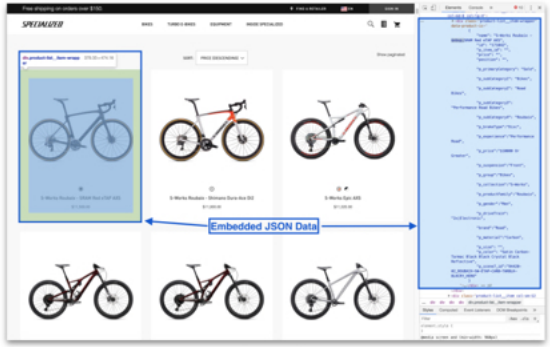
Scraping HTML

Carolina A. de Lima Salge
Assistant Professor
Terry College of Business
University of Georgia

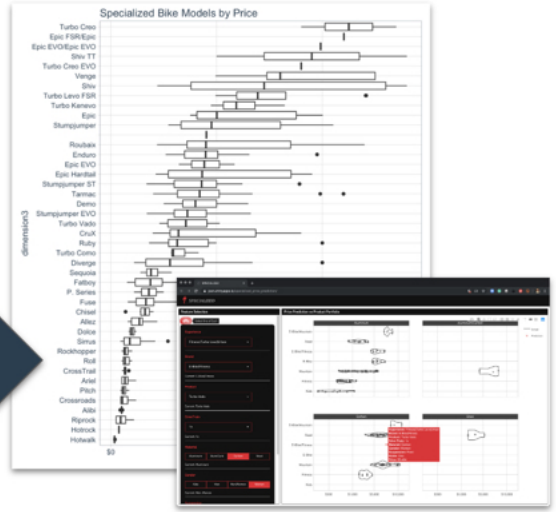
Business Intelligence
Spring 2021





id	brand	price	currencyCode
171042	Specialized	11500.00	USD
170241	Specialized	11000.00	USD
171229	Specialized	11020.00	USD
173494	Specialized	4520.00	USD
173495	Specialized	4520.00	USD
171068	Specialized	2150.00	USD
171069	Specialized	1675.00	USD
171070	Specialized	1250.00	USD
173818	Specialized	1000.00	USD
173814	Specialized	1000.00	USD

1–10 of 399 rows | 2–5 of 15 columns



Terry College of Business
UNIVERSITY OF GEORGIA

Web Scraping

Let's say, **data** is vital for your e-commerce company – you can see the data that you want but the question is: *How will you download it in a usable format?*

- Most would copy and paste manually
- Not feasible for websites with lots of pages
- Solution? 👇🏿👇🏾👇🏽👇🏼👇🏻

> Web scraping is a process of automating the extraction of data in an efficient and fast way. With the help of web scraping, you can extract data from any website, no matter how large is the data, on your computer.

https://towardsdatascience.com/https-medium-com-hiren787-patel-web-scraping-applications-a6f370d31614

Web Scraping

You copied and pasted some data, but how to convert or save it in a format of your choice?

- Ways to save as CSV, Excel, etc
- Able to import, analyze, and use the data

Web scraping simplifies the process of extracting data, speeds it up by automating it and creates easy access through familiar formats

https://towardsdatascience.com/https-medium-com-hiren787-patel-web-scraping-applications-a6f370d31614

Web Scraping

Three common ways to scrape data

- Basic (HTML)
- APIs (Twitter, Yelp)
- Advanced (Javascript)
  - Rselenium: https://rpubs.com/johndharrison/RSelenium-Basics

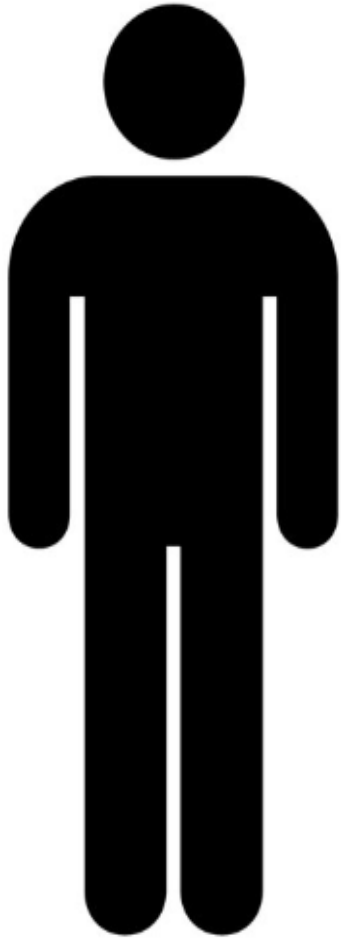
Web Scraping

Three common ways to scrape data

- Basic (HTML)
- APIs (Twitter, Yelp)
- Advanced (Javascript)
  - Rselenium: https://rpubs.com/johndharrison/RSelenium-Basics

Basic HTML

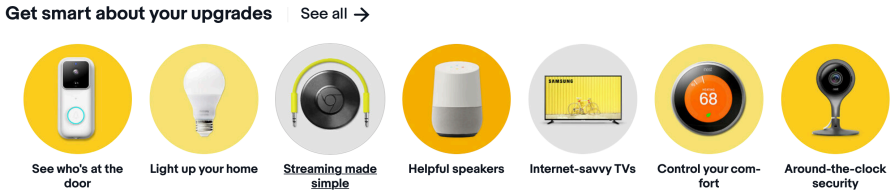
<head>

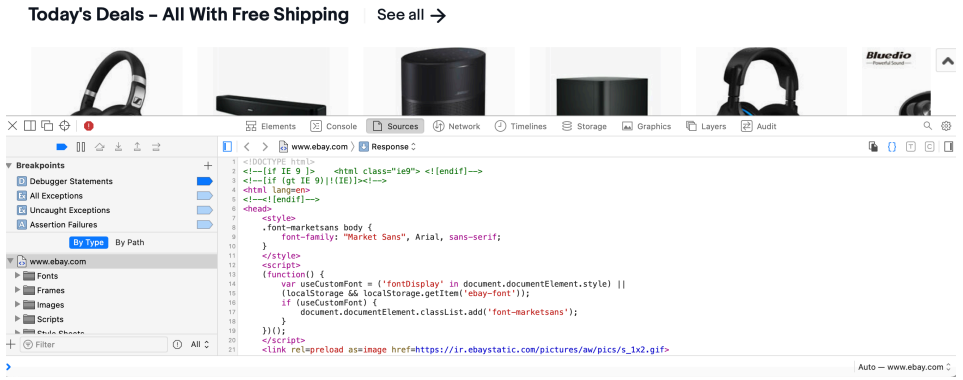


<body>

</head>

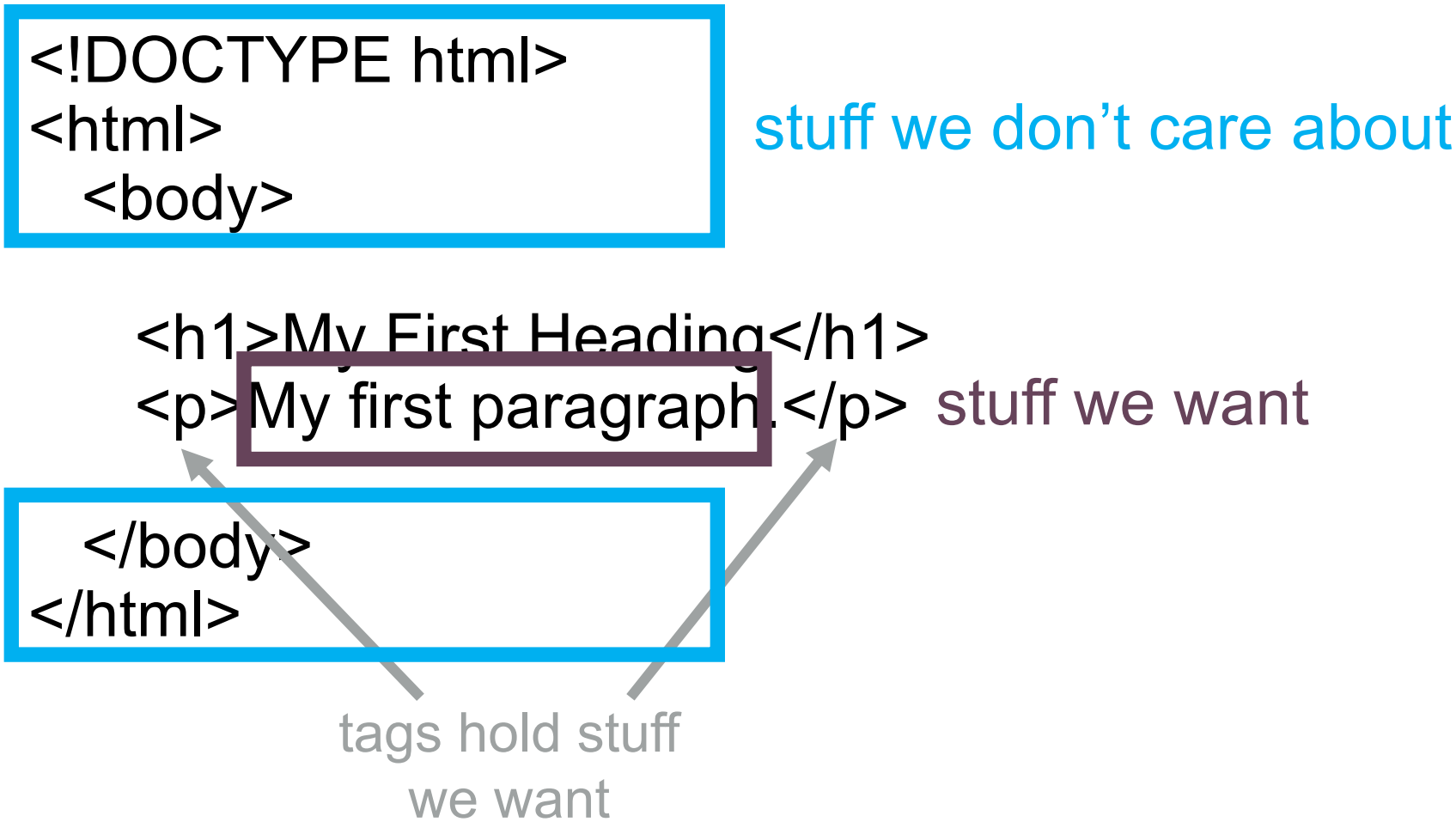
</body>





All HTML documents must start with a document type declaration: `<!DOCTYPE html>`
The HTML document itself begins with `<html>` and ends with `</html>`
The visible part of the HTML document is between `<body>` and `</body>`

HTML for Web Scraping



Finding Elements Inside HTML Code

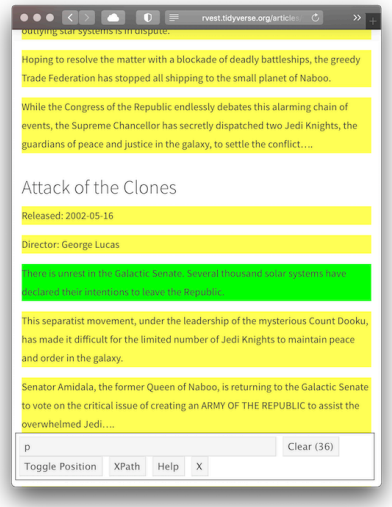
```
<!DOCTYPE html>
<html>
  <body>

    <h1>My First Heading</h1>
    <p class="alpha">My first paragraph.</p>
    <p class="beta">My second paragraph.</p>

  </body>
</html>
```

p .alpha →

Use *selectors to* find elements based primarily on either tag type (*e.g.* p, h1) or attributes (*e.g.* class)

Web Scraping HTML in R

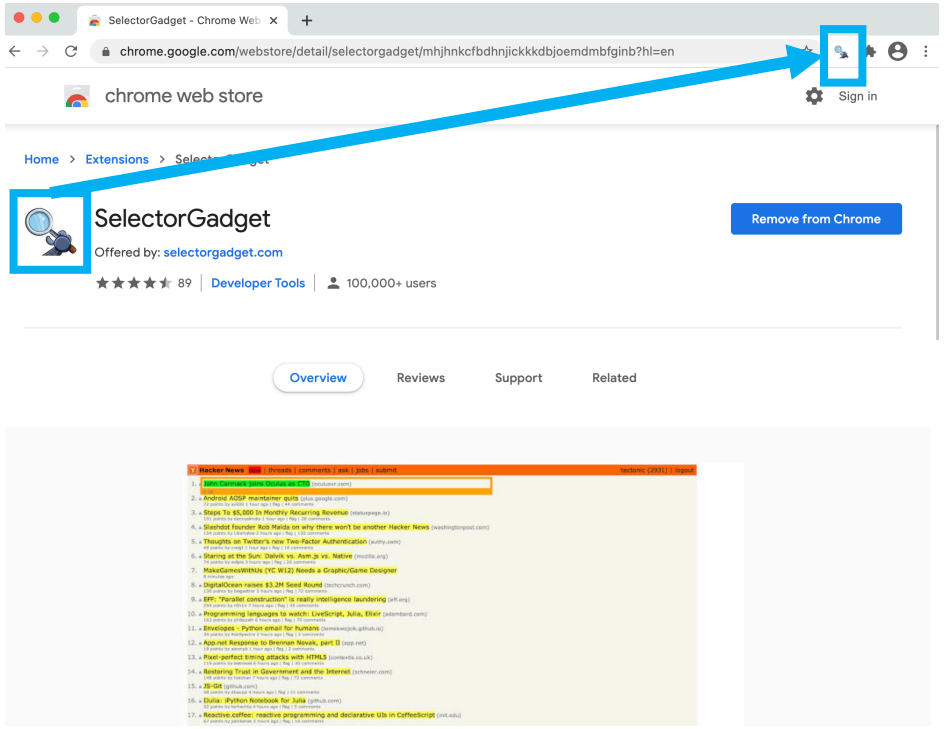
- Get HTML document
- Find and extract text from HTML document
- Create table based on vectors



https://www.business-science.io/business/2017/12/27/six-reasons-to-use-R-for-business.html

Quick Exercise

Open the Google Chrome web browser and install **SelectorGadget**



https://chrome.google.com/webstore/detail/selectorgadget/mhjhnkcfbdhnjickkkdbjoemdmbfginb?hl=en

SelectorGadget

A tool that makes it easy to find elements inside HTML code

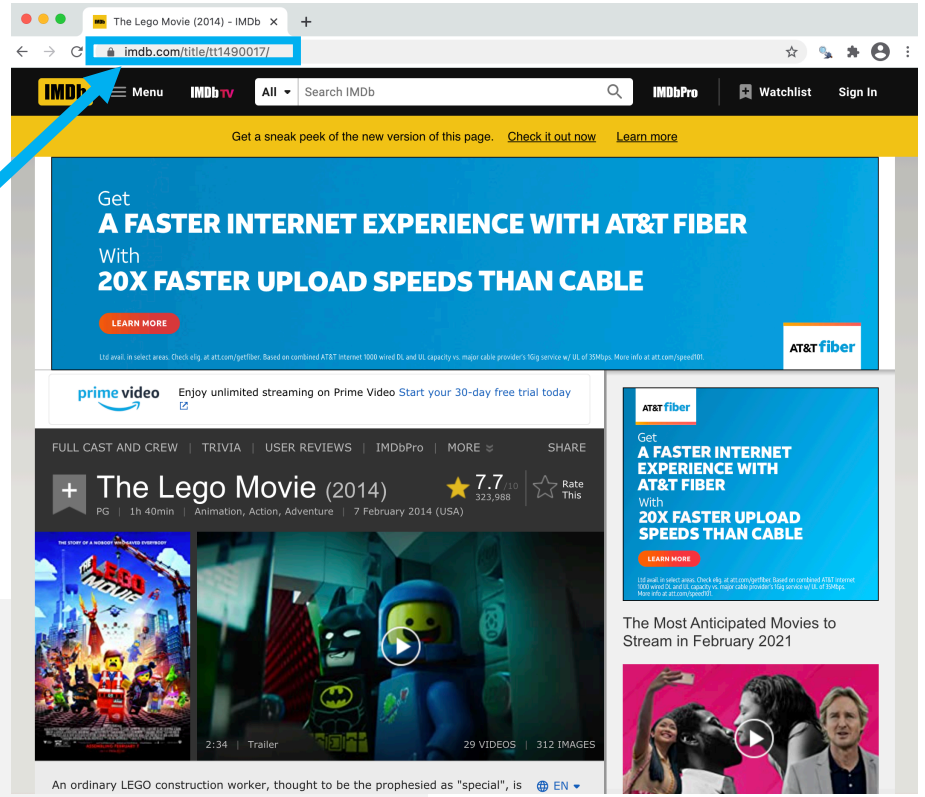
```
<!DOCTYPE html>
<html>
  <body>

    <h1>My First Heading</h1>
    <p class="alpha">My first paragraph.</p>
    <p class="beta">My second paragraph.</p>

  </body>
</html>
```

p .alpha →

Use *selectors to* find elements based primarily on either tag type (*e.g.* p, h1) or attributes (*e.g.* class)

Scraping Movie Data (IMDB)

Website we want to scrape data from




```
library(rvest)
library(tidyverse)

lego_movie <- read_html("http://www.imdb.com/title/tt1490017/")
```

Scraping Movie Data (IMDB)

rvest & tidyverse code

Base R code

```
rating <- lego_movie %>%
  html_nodes("strong span") %>%
  html_text() %>%
  as.numeric()
rating

cast <- lego_movie %>%
  html_nodes(".primary_photo+ td a") %>%
  html_text() %>%
  trimws()
cast
```

Tags holding stuff we want

Pipe function

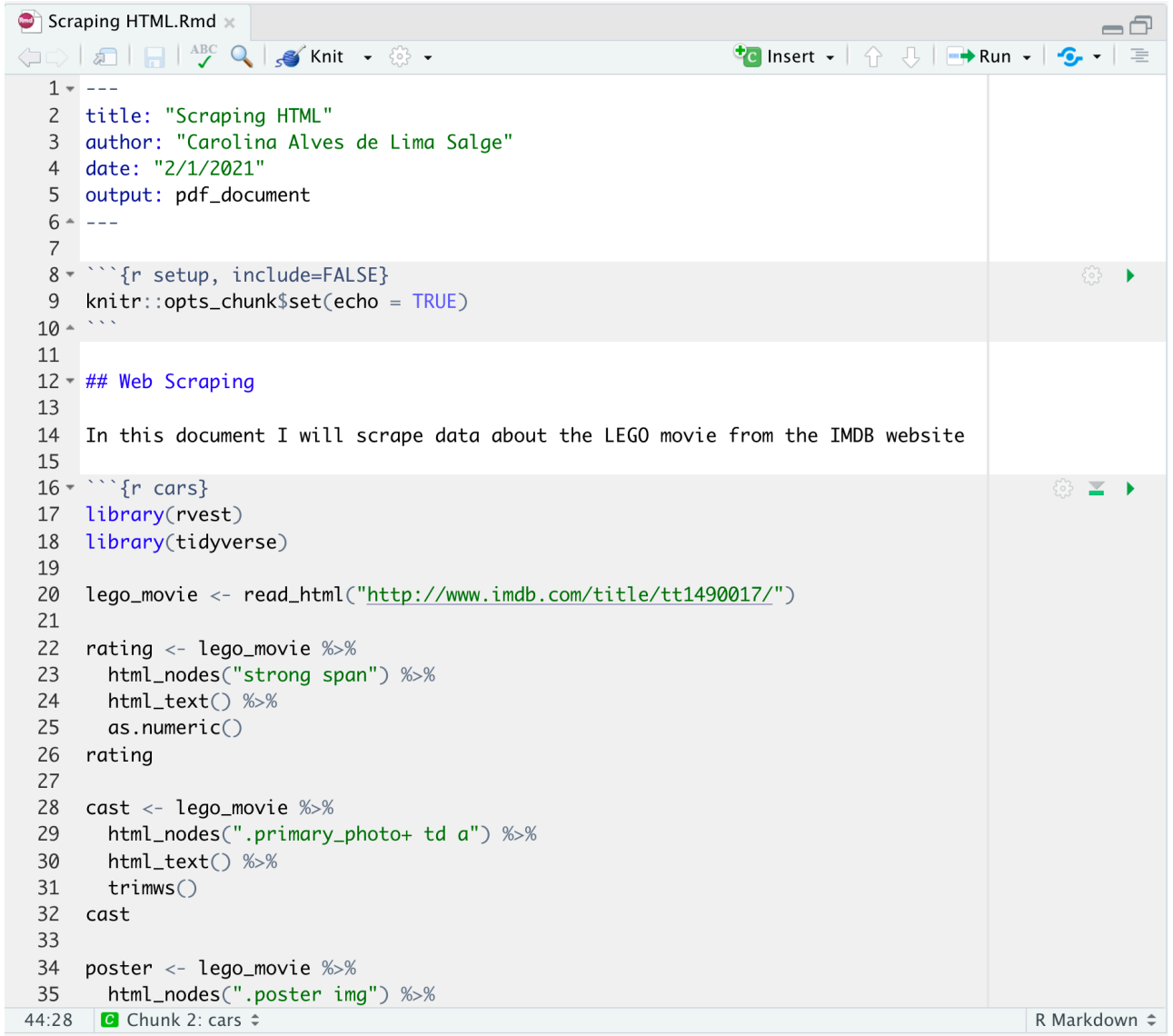
Scraping Movie Data (IMDB)

```
poster <- lego_movie %>%
  html_nodes(".poster img") %>%
  html_attr("src")
poster

lego <- tibble(rating = rating,
       cast = cast,
       poster = poster)
lego

write_csv(lego, "lego.csv")
```

Ready to scrape some data?

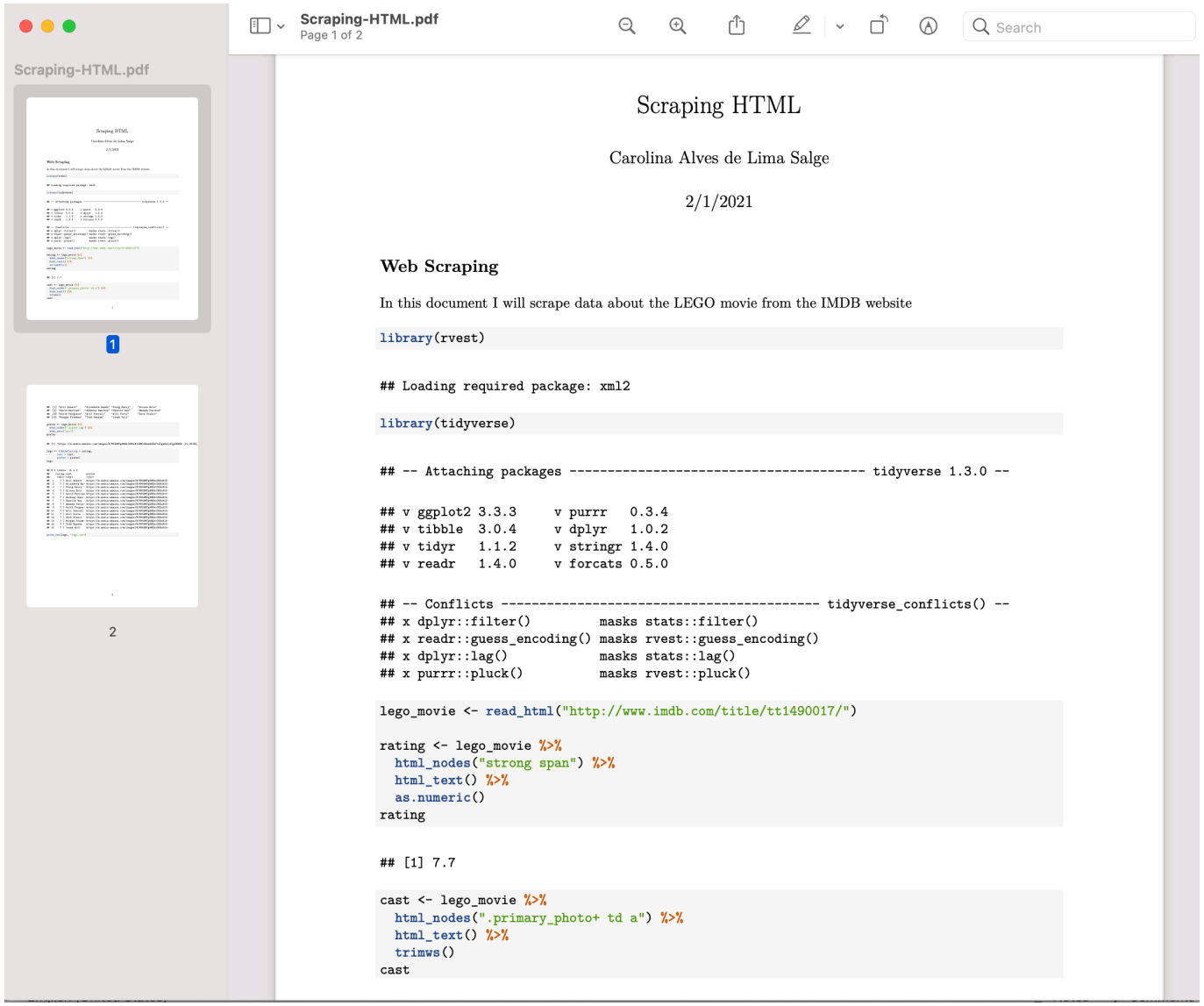
In-Class Exercise

- Open a Rmd file
- Copy and paste code from slides 13 to 15
  - Change the code to add the path to the folder you want to save the data scraped [`write_csv(lego, "lego.csv")`]
- Knit to Word/PDF



In-Class Exercise

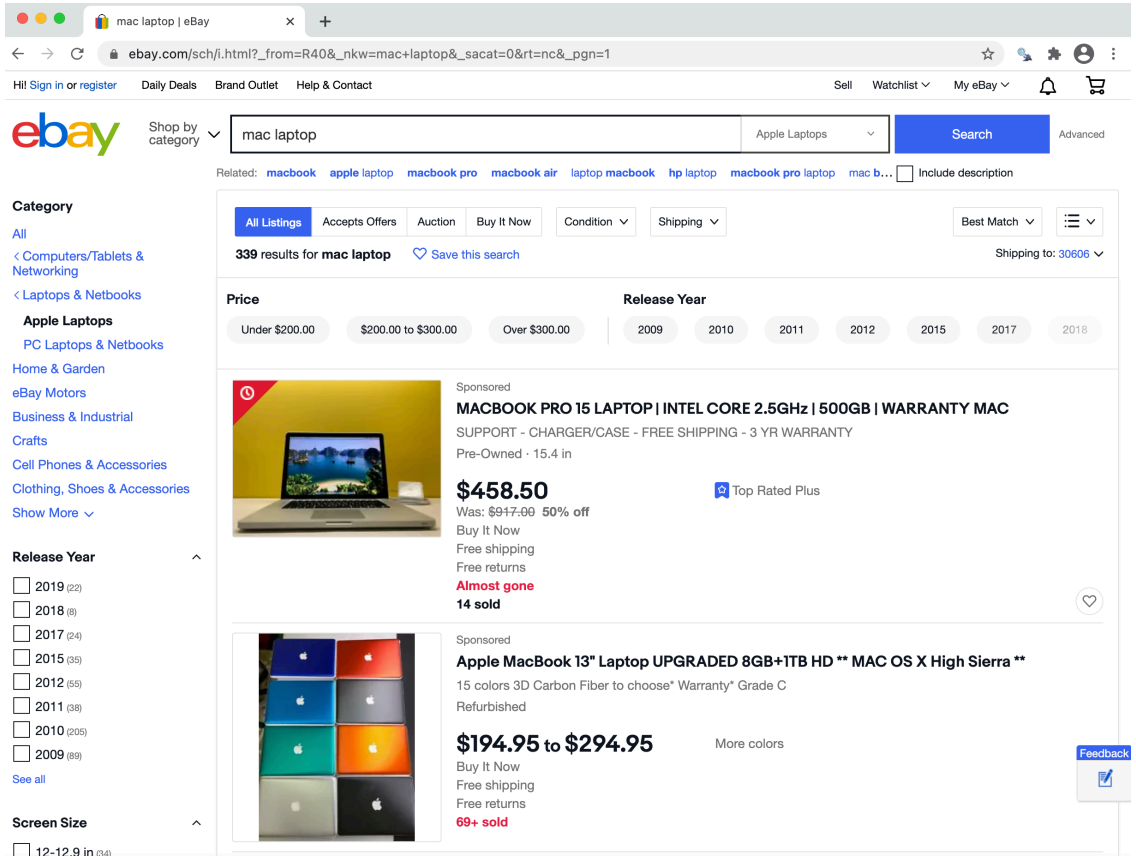
- Open a Rmd file
- Copy and paste code from slides 13 to 15
  - Change the code to add the path to the folder you want to save the data scraped [`write_csv(lego, "lego.csv")`]
- Knit to Word/PDF



At-Home Exercise 1

Scrape the title, price, condition, and photo link for all items in page 1, put the data together in a tibble, and save the tibble as CSV

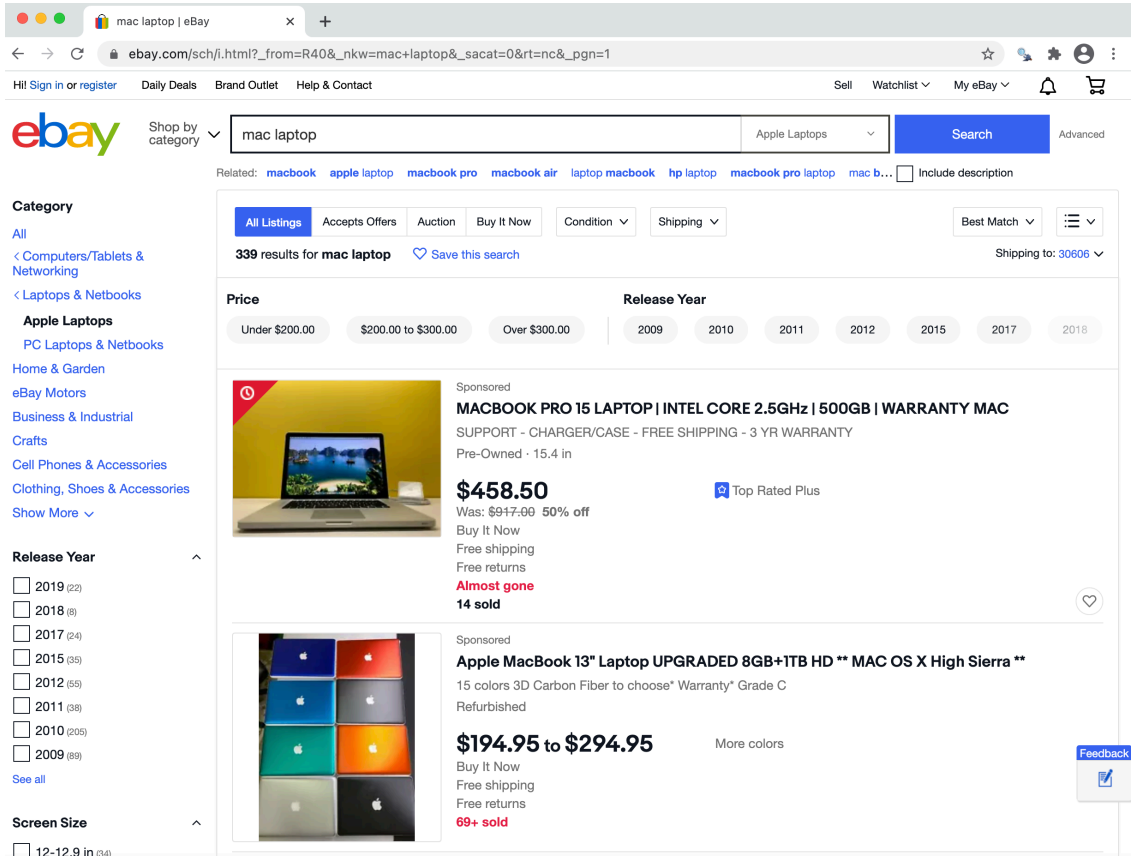
https://www.ebay.com/sch/i.html?_from=R40&_nkw=mac+laptop&_sacat=0&rt=nc&_pgn=1



At-Home Exercise 2 (More Challenging)

Scrape the title, price, condition, and photo link for all items in **ALL pages** (1 through 7), put the data together in a tibble, and save the tibble as CSV

https://www.ebay.com/sch/i.html?_from=R40&_nkw=mac+laptop&_sacat=0&rt=nc&_pgn=1



Resources for Exercise 2

https://rpubs.com/DrRobbieBaldock/690979

https://www.r-bloggers.com/2020/07/tutorial-web-scraping-of-multiple-pages-using-r/

https://towardsdatascience.com/coupling-web-scraping-with-functional-programming-in-r-for-scale-1bc4509eef29

https://www.business-science.io/code-tools/2019/10/07/rvest-web-scraping.html

Thank You!