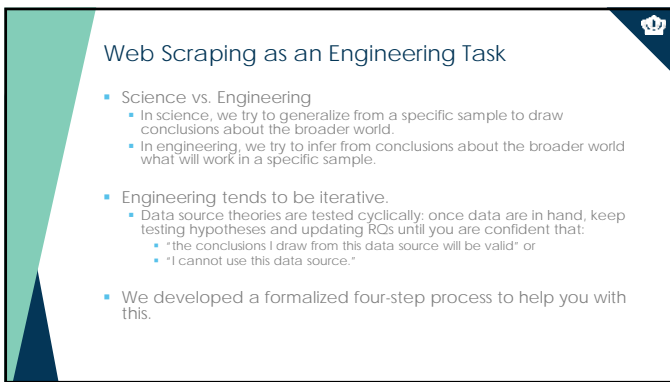




Technical Overview

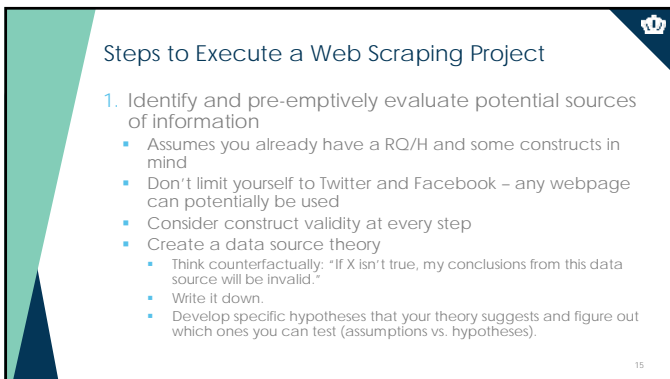
What are the steps involved in scraping social media?
What are APIs and how do I access them?
What are web pages?

1



Web Scraping as an Engineering Task

- Science vs. Engineering
 - In science, we try to generalize from a specific sample to draw conclusions about the broader world.
 - In engineering, we try to infer from conclusions about the broader world what will work in a specific sample.
- Engineering tends to be iterative.
 - Data source theories are tested cyclically; once data are in hand, keep testing hypotheses and updating RCs until you are confident that:
 - "the conclusions I draw from this data source will be valid" or
 - "I cannot use this data source."
- We developed a formalized four-step process to help you with this.



Steps to Execute a Web Scraping Project

1. Identify and pre-emptively evaluate potential sources of information
 - Assumes you already have a RQ/H and some constructs in mind
 - Don't limit yourself to Twitter and Facebook – any webpage can potentially be used
 - Consider construct validity at every step
 - Create a data source theory
 - Think counterfactually: "If X isn't true, my conclusions from this data source will be invalid."
 - Write it down.
 - Develop specific hypotheses that your theory suggests and figure out which ones you can test (assumptions vs. hypotheses).

15

Steps to Execute a Web Scraping Project

2. Develop a coding system
 - a) Identify the specific constructs you want to assess
 - b) Identify the specific pieces of information you want to grab from each website
 - Remember to include info to test your data source theory
 - c) Determine where each piece of information appears on each webpage
 - d) Determine how cases are replicated in terms of the webpages
 - Is there one case on each webpage?
 - If multiple cases are represented on each webpage, how are they represented?

16

Steps to Execute a Web Scraping Project

3. Code a scraper and potentially a crawler
 - When scraping, data will come from one of two sources depending upon which website's data you're trying to access
 - If an API is available, you want to use the API
 - Returns **structured** data with variables pre-defined
 - Legally unambiguous
 - If an API is not available, you'll need to scrape manually
 - Returns **unstructured** data
 - Requires a lot more work
 - Legally ambiguous in some cases

17

Overview of API Calls

- **API: Application Programming Interface**
 - A data gateway into someone else's system
 - Created by the provider of the service
 - Almost universally intended and designed for real-time access by other websites, but you can use them too
 - Requires learning API documentation - they're all different
- You generally access APIs using one of these HTTP protocols:
 - **GET requests:** request is embedded in a URL
 - **POST requests:** request is embedded in a larger system of document requests sent by your web browser
- We will focus on a GET requests, because they're more common and much easier

18

How Hypertext Transfer Protocol (HTTP) Works

- It's very difficult to describe how the Internet works in aggregate because there are many moving parts, even for the seemingly simplest tasks
- We'll focus on HTTP requests, the kind sent by your web browser
 - Can be conceptualized as a sequential set of exchanges of information



- Once one of the ways that a server can customize its content to you is with GET requests: a single webpage on a server can deliver different content depending upon parameters sent by a client

19

A Starter GET API Request

- Let's start easy. I've created an API at <http://scraping.tntlab.org/add.php>
- It adds two numbers, x and y.
- Try:
 - <http://scraping.tntlab.org/add.php>
 - <http://scraping.tntlab.org/add.php?x=1>
 - <http://scraping.tntlab.org/add.php?x=1&y=muffin>
 - <http://scraping.tntlab.org/add.php?x=1&y=8>

20


API Request Structure

- <http://scraping.tntlab.org/add.php?x=1&y=8>
- This GET request has two main parts:
 - URL (*uniform resource locator*): <http://scraping.tntlab.org/add.php>
 - Query string:
 - Begins with ?
 - Fields/methods come before =
 - Values/parameters come after =
- Try different field/value pairs and see what happens
- All of this must be coded manually by the API developer
 - Try to add a field called *format* with value *csv* and try again
 - Change the value to *tab* and try again
 - Change the value to *matrix* and try again

21

What if there isn't an API?

- Then we need to grab data by hand and create an algorithm to provide the computer with a template for how to interpret it



- The first time you visit a webpage, your web browser sends a GET request without any fields or values
- The file that is initially returned is (usually) an HTML document: *hypertext markup language* (a specific kind of XML)
- This file is the one we will need to interpret, but without the aid of a web browser to view it
 - You've seen raw HTML yourself if you've ever clicked "View Source"

22

The Basic Structure of HTML: Tags

```

<html>
<head>
<title>My First Webpage</title>
</head>
<body>
<h1>My First Heading</h1>
<p>This is my first paragraph of info!<br />And a line break!</p>
<p>This is my <b>second paragraph</b> of info!</p>
</body>
</html>
    
```

- Opening tags are just words
- Self-closing tags *may* have a trailing /
- Closing tags are the same words, preceded by /

- Some tags are structural, like *html*, *head*, *title*, *body*, *h1*, *p*
- Some tags are inline, like *b*
- If the creator created valid HTML, nesting is always complete

23

The Basic Structure of HTML: Tags

```

<html>
  <head>
    <title>My First Webpage</title>
  </head>
  <body>
    <h1>My First Heading</h1>
    <p>This is my first paragraph of info!<br />And a line break!</p>
    <p>This is my <b>second paragraph</b> of info!</p>
  </body>
</html>
    
```

- Reorganizing HTML by its structural tags improves *readability* but your browser doesn't care. See <http://scraping.intlilab.org/first.html>

My First Heading

This is my first paragraph of info!
And a line break!

This is my second paragraph of info!

24

Regular Expressions

- Regular expressions are enormously powerful and can be very confusing, even if you know what you're doing
 - Can be used to identify or replace text
- Examples of simple regex replacement with "x": I have 9 dogs.
 - `\d` Match any digit I have x dogs.
 - `[ade]` Match letters a, d, or e I hxxx 9 xogs.
 - `\w` Match any alphanumeric x xxxx x xxxx.
 - `\W` Match any non-alphanumeric lxxhavex9xdogsx
 - `\s` Match any whitespace lxxhavex9xdogs.
- Can get really, really complicated
 - `^\([0-9]{3}\) | [0-9]{3}-[0-9]{3}-[0-9]{4}$`
- Learn with <https://regexone.com/>

28

Identifying Specific Tags in the DOM

- Useful things to know about HTML when DOM snooping
 - Correctly written HTML only allows one *id* attribute per document
 - class* attributes are used to group "similar kinds of information" that appears multiple times
- Match your XPath to the level of information being extracted from each page individually
- So where's Fred's name in the DOM?
 - `span[@id="fb-timeline-cover-name"]`
 - `#fb-timeline-cover-name`

29

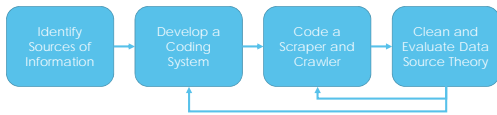
When Scraping Pages, You'll Need a Crawler

- Crawling involves algorithmically, iteratively reading links on a webpage and following them
 - Similar process conceptually: look at the webpages you're trying to grab and figure out where the links are
 - Identify the commonalities between all links you want to follow
- <http://reddit.com/r/IOPsychology>

30

Steps to Execute a Web Scraping Project

4. Clean the data and revise the data source theory
 - Once you have your data in hand, run all hypothesis tests possible from your data source theory
 - You will almost certainly identify problems with your coding system at this stage; time to revise



31
