Master Tutorial

TITLE

Web Scraping with R

SHORTENED TITLE

Web Scraping with R

ABSTRACT

A lot of data exists on the web! Accessing that data requires understanding HTTP requests, security tokens, data transfer file formats, and data cleaning. This tutorial session will walk you through how to access and process web-based data using the powerful statistical language R. Bring your laptop (optional) for this interactive session (download session materials here: ).

PRESS PARAGRAPH

Gathering data no longer requires handing out questionnaires or setting up experiments. Many interesting questions can be answered by accessing data on the web. The statistical computer language R allows scientists and practitioners to scrape the web and obtain a world of data without leaving one’s seat, including the distribution of the population, the sentiment of consumers and employees, and the change in stock prices. This session will give a hands-on and step-by-step tutorial on how to access and process web-based data for the purposes of analyses.

WORD COUNT: 1898

**Web Scraping with R**

R (R Core Team, 2018) is an open-source programming language that is designed for statistical computing (Hornik, 2013). R can perform anything from standard data analysis (e.g. Multiple Regression, Hierarchical Linear Modeling, or Structural Equation Modeling) to highly specialized computations that may be unique to a scientific field. R is a programming *language* and not just a statistical analysis package. Many data scientists and practitioners can contribute to R by writing new and unique software, called “packages” in R. At the time of this writing, the Comprehensive R Archive Network (CRAN) contains 12,959 available packages, including packages to read data in varying formats (e.g., readr, open.xlsx, haven, rjson, officer), access databases (e.g., DBI, odbc, RSQLite), clean data (e.g., dplyr, tidyr, stringr, reshape2), perform data analyses and machine learning (e.g., caret, xgboost, randomForest, caret, survival), visualize results (e.g., shiny, ggplot2), and interface with other programming languages (e.g., Rcpp, reticulate, RJava). These packages, just like R itself, are free of charge.

This Master Tutorial will teach attendees how to leverage R and several packages to access data from various websites and put that data into a form useful for data analysis. We will teach users how to request information from servers, access web APIs (including those that require authentication keys), extract useful information from web requests, and process online data for the purposes of further analyses. Attendees should be familiar with R and have both R and RStudio installed prior to the workshop. We will walk through and explain each line of code in detail, but we will have little time to review the basics of R itself.

*Proposed Session*

The proposed tutorial is a continuation of previous tutorials attempting to simplify R for the I/O community. By some measures, R has become one of the ten most popular programming languages (Cass, 2018) and is growing in popularity nearly as quickly as the general-purpose programming language Python (Robinson, 2017). R’s popularity may be partly due to the large ecosystem of support pages, books, blogs, tutorials, and R specific conferences. Several R packages, including dplyr (Wickham, François, Henry, and Müller), ggplot2 (Wickham, 2009), and data.table (Dowle and Srinivasan, 2018) have over 12,000 mentions on Stack Overflow (Robinson, 2017).

Considerable information is contained on the internet. This data is often behind APIs, on unstructured HTML pages, or in particular formats such as XML or JSON. Putting this data into usable format for analyses can be difficult and require specific knowledge. R now has a wide variety of tools to access, process, and analyze web-based data (Munzert, Rubba, Meißner, and Nyhuis, 2015). Unfortunately, I-O psychologists often lack the training to be able to access this data for their own research purposes without tedious, manual processes or help from computer programmers. Building on several R-based master tutorials over the past few years (e.g., Schwall, Lustenberger, Beatty, and Jones, 2014; Schwall, Beatty, & Jones, 2015; Goebl, Jones, & Semmell, 2016; Goebl, Jones, & Semmell, 2018), this tutorial aims to bridge the gap between the typical I/O psychologist’s experience and R’s ability to solve research problems. We will explain web-scraping as implemented by R in two parts.

The first part of the tutorial will introduce packages that pull data from the web and put that data into a usable format within the R working session. We will introduce the terminology of web requests (such as PUT, POST, GET) and show how to construct appropriate requests to pull data from websites and translate that data for appropriate analysis. Our discussion will emphasize several packages that make pulling data from the web (such as httr; Wickham, 2017), and translating that data for use in further analyses (such as xml2; Wickham, Hester, and Ooms, 2018; and rjson; Couture-Beil, 2014) easier and consistent.

The second part of the tutorial will guide participants in setting up a developer platform on Twitter and using API requests to pull data referencing a particular company within a window. At the same time, we will pull financial data on that company using different APIs during the same window and compare the sentiment of Tweets (e.g., Silge and Robinson, 2018) with the financial data of the company. Although our example relates Twitter to financial performance, the principles behind our demonstration apply to any web scraping activity.

Audience members are encouraged to bring laptops and to have downloaded the materials ahead of time. For those who wish to follow along, we will make available all materials and R scripts at . We request 80 minutes for the tutorial, with the approximate time for each topic as well as additional information provided below. Note that none of the authors are affiliated with the producers of any of the packages described and that there are no material gains (financial or otherwise) for them. All packages are free-of-charge and complete.

**Topic #1: Web Requests and Data Formats (30 minutes)**

In principle, data on a website can be stored in the same format as data on one’s own hard drive. A website is hosted on a server, and that server can contain any types of files. Unfortunately, standard file formats are not amenable for efficiently representing data with complex structure. For instance, a flat file would not easily be able to depict hierarchical relationships between a company, its teams, the individuals on those teams, the traits of those individuals, and connections with other teams. One could have separate columns for “company”, “team”, “individuals”, etc., but the elements of those columns would need to be repeated more times than necessary to adequately capture all relationships in a flat format.

Two formats that typically represent web-based data are XML (Extensible Markup Language) and JSON (JavaScript Object Notation). XML was developed by the World Wide Web Consortium (the same group that has set the standard for many other web-based formatting languages, including CSS and HTML). XML is written in a similar style to HTML, with tags, attributes, and elements. However, HTML tags are fixed and not mutable (without changes to the standards), whereas XML tags are customizable. This capacity allows XML to capture a similar structure to HTML-based documents (and easy insertion into HTML-type webpages and applications) with the flexibility of self-defined groupings. One can also set a schema so that XML must conform to a pre-specified structure. This capacity provides predictability, so that users know the format of any data they encounter, and code written to process that data can also process new data that adheres to the same schema.

Similarly, JSON is a standard-based data transfer format that allows for pre-defined schemas so that data structures are easily predictable and processable. However, JSON format was designed to work in JavaScript and is, thus, object-based rather than tag-based. JSON files are simply a nested array of either arrays or values that are either named or ordered. For instance, JSON formats can easily represent an individual person (see <https://en.wikipedia.org/wiki/JSON>) with a named value “firstName”, a named value “lastName”, an array of named values for “address” (where the array includes “streetAddress”, “city”, “state”, and “postal code”), and an array of arrays for “phoneNumbers” (where the each sub-array contains individual phone numbers marked by “type”, and “number”). Note that this format is easily extensible. If we want to create a list of people, we simply need to add the JSON representing this person as a sub-array of one representing all people in a particular group. Both XML and JSON formats are common methods of online data transfer, but each has a particular structure and, thus, required methods to efficiently extract parts of that structure.

To extract data from webpages, we need to enable R to communicate with websites. When accessing websites from a browser, we simply type the webpage into the search bar, and the browser immediately extracts and renders the relevant information. The browser communicates with the web using a language called Hypertext Transfer Protocol (HTTP). After typing a domain name into a search bar, the browser asks a Domain Name System (DNS) server the relevant IP address. Then the search bar sends a relevant request for information to the server assigned IP address, and the server sends a response, which the browser displays as a functioning website. Often, one needs to login to a website or communicate appropriate credentials before the server will release data. Browsers do all of this automatically. To access the same data in R, one needs a way of communicating with servers using the appropriate protocols and then consuming and processing that information. The C library “libcurl” was written to transfer data using various protocols, including HTTP, FTP, IMAP. R contains the library “curl” (Ooms, 2018) that provides R users an access point to the libcurl library and the library “httr” (Wickham, 2017) with verbs (such as GET, POST, PUT, etc.) that simplify accessing HTTP data by formatting appropriate requests with “curl”.

**Topic #2: API Setup (10-15 minutes)**

Many websites do not simply allow people to make a data request and receive unlimited data hosted on the appropriate servers. Often, users must setup an application on the desired website and then provide authentication credentials in R for use with the particular application. Examples of websites requiring users to setup applications include Twitter (<https://developer.twitter.com/>) and Facebook (https://developers.facebook.com/).

This section will simply walk participants through creating the appropriate (free) application on Twitter and extracting the appropriate credentials from the application to access the developer interface from R. Information from this section would be useful in setting up R to extract data from websites with developer credentials and authentication keys. For participants wanting to explore the web scraping example but do not want to setup an application from scratch, we will also provide already-created authentication keys. We will also explain functionality of the “twitteR” package (Gentry, 2015) in formulating and sending requests for Tweets that fall within a particular date range or contain particular words.

**Topic #3: Example Web Scraping (30 minutes)**

To integrate the first several sections (on data format, APIs, and web requests), we will provide a complete example that includes scraping data from Twitter that mentions particular companies and financial data of those companies. We will explain how to process and clean the data using tools in R, and we will use sentiment analysis to compare the type of Tweet with financial information from the companies. The latter topic is not intended for in-depth exploration but, rather, provides an complete example of a question one might ask as well as how to go about trying to answer that question. If time permits, we will allow groups to explore the data (and the Twitter API) on their own and walk around to help answer questions they might have about approaching certain problems.

**Topic #4: Wrapup (5-10 minutes)**

Finally, the presenters will answer audience questions and help with technical problems encountered during previous sections. The presenters will provide additional materials for participants to read for self-study and include links to useful materials for approaching different web-scraping problems.

**Learning Objectives**

This workshop is designed to help you:

1. Explain online data formats and data transfer protocols to colleagues and understand why data on the web differs from commonly used data formats.
2. Setup applications to access web APIs from R, including knowing how to find and use authentication keys.
3. Formulate questions that require online data sources and know how to approach answering those questions without trepidation or fear.

**Presenter Information**

Jeff Jones

Director, Talent Analytics and Data Sys

Korn Ferry

33 South Sixth Street

Suite 4900

Minneapolis, MN 55402

Phone: 612-373-3418

Email: jeff.jones@kornferry.com

Membership Status: Member

Steven Nydick

Data Scientist Programmer

Korn Ferry

33 South Sixth Street

Suite 4900

Minneapolis, MN 55402

Phone: 612-373-3548

Email: steven.nydick@kornferry.com

Benjamin Wiseman

Data Scientist Programmer

Korn Ferry

33 South Sixth Street

Suite 4900

Minneapolis, MN 55402

Phone: 612-373-3445

Email: benjamin.wiseman@kornferry.com

**Presenter Bios**

Steven Nydick is a Data Scientist Developer at the Korn Ferry Institute, where he designs R-based tools and scoring algorithms. He is the lead author and maintainer of the catIrt R package as well as several internal R packages helping with everything from plotting to powerpoint generation to interfacing with servers. He has contributed to developing psychometric models and corresponding estimation algorithms that have been published in *Applied Psychological Methods* and the *Journal of Educational and Behavioral Statistics*. Steven received his Ph.D at the University of Minnesota in Psychometrics and Quantitative Psychology, where he primarily studied IRT-based adaptive test for selection and classification. He also has an M.S. in Statistics from the University of Minnesota.

**References**

Cass, S. (2018). The 2018 top programming languages. Retrieved August 29, 2018, from <https://spectrum.ieee.org/at-work/innovation/the-2018-top-programming-languages>

Couture-Beil, A. (2014). Rjson: JSON for R. R package version 0.2.15.

Dowle, M., & Srinivasan, A. (2018). data.table: Extension of `data.frame`. R package version 1.11.4.

Gentry, J. (2015). twitter: R based Twitter client. R package version 1.1.9.

Hornik, K. (2017). The R FAQ. Retrieved from http://CRAN.R-project.org/doc/FAQ/R-FAQ.html

Jones, J. A., Goebl, A. P., & Semmel, S. G. (2016, April). *Handling big(gish) data in R:*

*An introductory and interactive tutorial.* Master Tutorial at the annual meeting of the Society of Industrial and Organizational Psychology, Anaheim, CA.

Jones, J. A., Goebl, A. P., & Semmel, S. G. (2018, April). *Machine Learning in R: A Tutorial and Jam Session.* Master Tutorial at the annual meeting of the Society of Industrial and Organizational Psychology, Chicago, IL.

Munzert, S., Rubba, C., Meißner, P., & Nyhuis, D. (2015). *Automated Data Collection with R*. New York, NY: Wiley.

Ooms, J. (2018). Curl: A modern and flexible web client for R. R package version 3.2.

R Core Team. (2018). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. URL https://www.R-project.org/.

Robinson, D. (2017). The impressive growth of R. Retrieved August 29, 2018, from

https://stackoverflow.blog/2017/10/10/impressive-growth-r/

RStudio - RStudio IDE. (n.d.). Retrieved September 2, 2015, from <http://www.rstudio.com/ide/>

Schwall, A., Lustenberger, D., Beatty, A., & Jones, J. A. (2014, May). *Getting started*

*with R: Examples and Lessons Learned.* Panel discussion at the annual meeting

of the Society of Industrial and Organizational Psychology, Honolulu, HI.

Schwall, A., Beatty, A., & Jones, J. A. (2015, May). *Getting started with R: An*

*Interactive and Hands on Tutorial.* Master Tutorial at the annual meeting of the

Society of Industrial and Organizational Psychology, Philadelphia, PA.

Silge, J., & Robinson, D. (2017). *Text Mining with R: A Tidy Approach*. Sebastopol, CA: O’Reilly Media,

Inc. Retrieved August 31, 2018, from https://www.tidytextmining.com/.

Wickham, H. (2009). *ggplot2: Elegant Graphics for Data Analysis*. New York, NY: Springer.

Wickham, H. (2017). httr: Tools for working with URLs and HTTP. R package version 1.3.1.

Wickham, H., François, R., Henry, L., & Müller, K. (2018). dplyr: A grammar of data manipulation. R

package version 0.7.6.

Wickham, H., Hester, J., & Ooms, J. (2018). Xml2; Parse XML, R package version 1.2.0

**Appendix**

CV Jeff Jones

CV Steven Nydick

CV Benjamin Wiseman

Steven Nydick

Email: Steven.Nydick@KornFerry.com

**Education:**

PhD, University of Minnesota, Psychometrics/Quantitative Psychology, 2013.

Advisor: Niels Waller

MA, University of Minnesota, Psychometrics/Quantitative Psychology, 2012.

Advisor: Niels Waller

MS, University of Minnesota, Statistics, 2011.

Advisor: Sanford Weisberg

BS, Syracuse University, Mathematics and Psychology, 2006.

**Professional Experience:**

Data Scientist Developer, Korn Ferry, 2018 – Present.

Senior Psychometrician, Pearson VUE, 2016 – 2018.

Psychometrician, Pearson VUE, 2013 – 2016.

Research Assistant, University of Minnesota, 2013 – Present.

Intern in Psychometrics, ARRT, 2012 – 2013.

Intern in Psychometrics, ACT, 2011.

Graduate Instructor/Section Leader, University of Minnesota, 2007 – 2013.

**Awards:**

Doctoral Dissertation Fellowship, 2013

Graduate Research Partnership Program, 2010

Archimedes Prize in Mathematics, 2006

**Manuscripts Published and In Press:**

Wang, C. & Nydick, S. W. (2015). Comparing two algorithms for calibrating the restricted non-

compensatory multidimensional IRT model. *Applied Psychological Measurement*, *39*, 119-134.

Nydick, S. W. (2014). The sequential probability ratio test and binary item response models. *Journal of*

*Educational and Behavioral Statistics*, *39*, 203-230.

**Software:**

Nydick, S. W. (2014). catIrt: An R package for simulating computerized adaptive tests. R package version

0.5-0).

**Presentations and Workshops:**

Nydick, S. W. (2016, April). The expected likelihood in computerized classification testing. Paper

presented at the annual meeting of the National Council on Measurement in Education, Washington, DC.

Nydick, S. W. (2014, April). Multidimensional mastery testing with CAT. Paper presented at the annual

meeting of the National Council on Measurement in Education, Philadelphia, PA.

Nydick, S. W., Wang, C., & Xiong, X. (2014, April). Measuring multidimensional growth—a higher-order

IRT perspective. Paper presented at the annual meeting of the American Educational Research Association, Philadelphia, PA.

Nydick, S. W., Nozawa, Y., & Zhu, R. (2012, April). Accuracy and efficiency in classifying examinees using

computerized adaptive tests: An application to a large scale test. Paper presented at the Annual Meeting of the National Council on Measurement in Education, Vancouver, BC.

Nydick, S. W., & Weiss, D. J. (2010, June). Accepting the null: No change in change CAT. Paper presented

at the IACAT conference on CAT, Arnhem, NL.

Nydick, S. W., & Weiss, D. J. (2009). A hybrid simulation procedure, evaluated for the development of

CATs. In D. J. Weiss (Ed.) *Proceedings of the 2009 GMAC Conference on Computerized Adaptive Testing.*

**Unpublished Manuscripts:**

Nydick, S. W. (2013). *Intro to R for Psychologists.* Minneapolis, MN: Author.

**Courses Taught:**

Introduction to Data Analysis/Statistics for Undergraduates

Honors Introduction to Data Analysis/Statistics for Undergraduates

Analysis of Psychological Data for Graduate Students