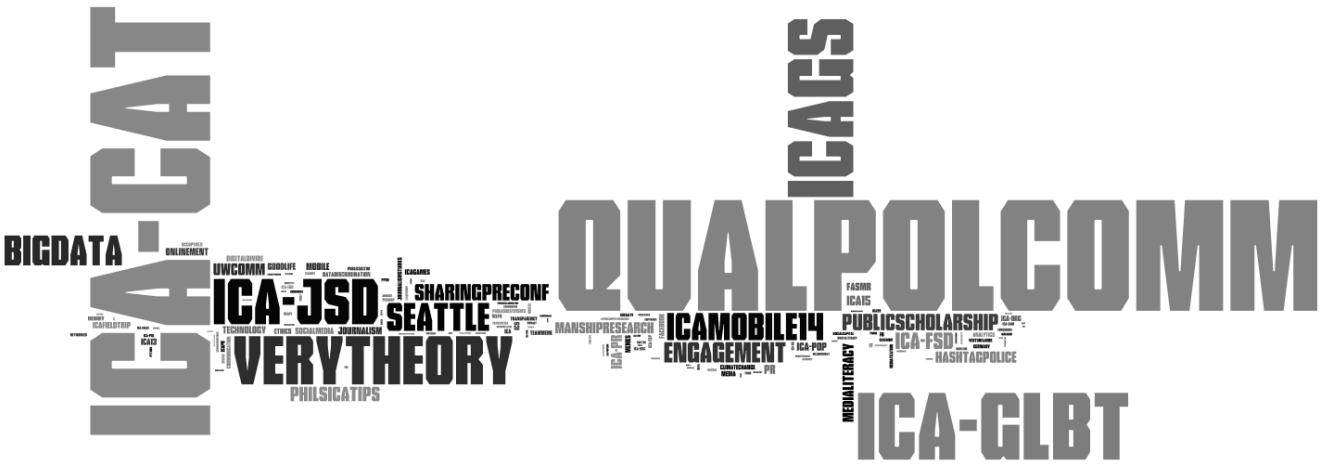
#ICA14 – Tag cloud of ICA 2014 Tweets

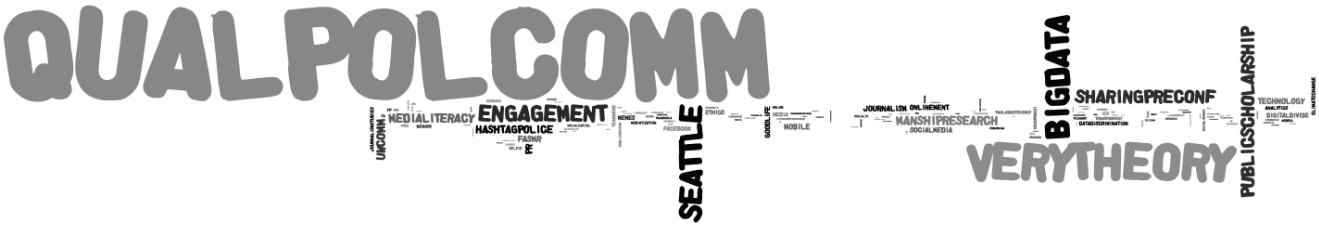
Word clouds are generally not all that helpful given how the words are taken out of their context (sentences). In certain settings, however, they do provide meaningful information. Hashtags are one of those contexts – they are *meant* to be single words. With that, here is a word cloud generated using all of the hashtags included in all 9,010 tweets sent with the *#ica14* hashtag as of Monday, May 26th @ 1pm EST.

Here is a first cloud, with the *#ica14* tag excluded.



The larger the tag, the more frequently it was used. You can see that ICA section-related tags predominated, with *#ica_cat* and *#ica_glbt* leading the pack. (Note that, after downloading and processing the Twitter data in Python, I used Wordle to generate the word cloud. A quirk of Wordle is that it will split words with an underscore in them, so I've replaced underscores with hyphens. So, read "*ica-cat*" as "*#ica_cat*," etc.)

To help highlight non-section tags, here is a version omitting any tag with "ica" in it.

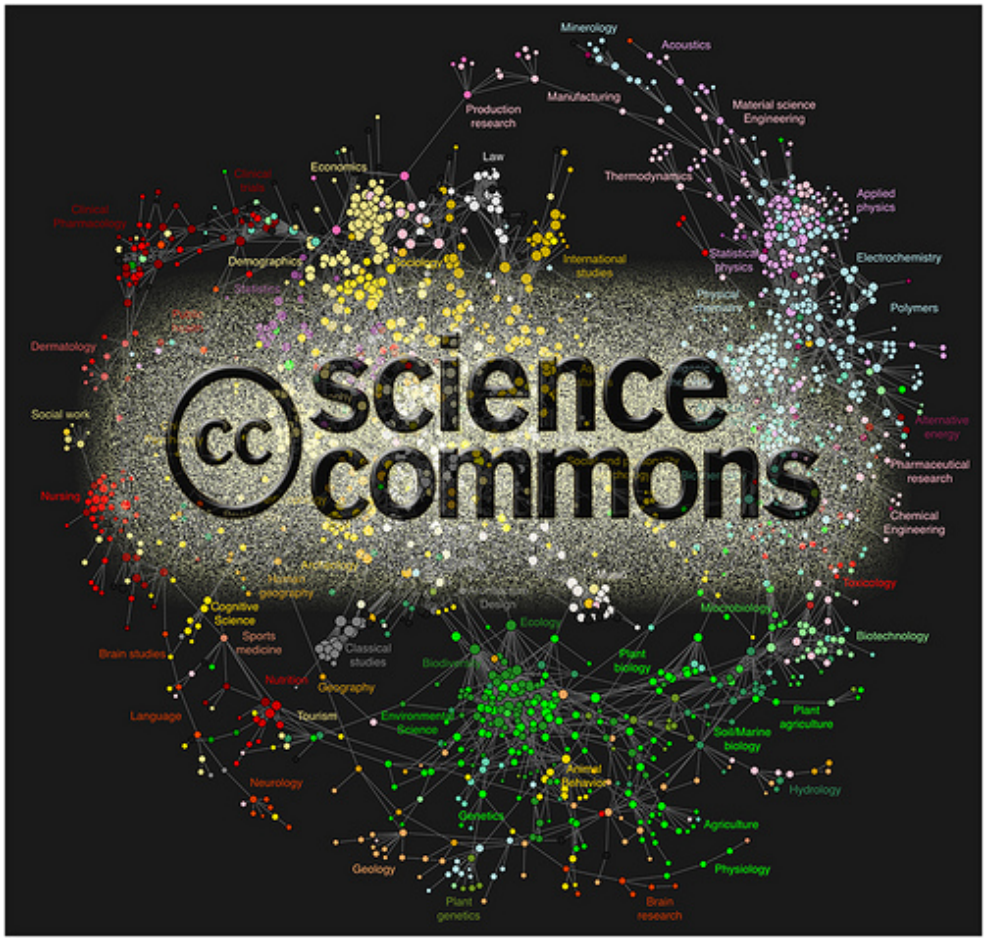


The *#qualpolcomm* taggers were highly active. To help see other topics, here's a final tag cloud without *#qualpolcomm*.

I'll leave the analysis up to you. Some interesting patterns here!

If anyone is interested in the Python used to generate this let me know.

Replication Data



By: Duncan Hull

One of the core tenets of scientific research is *replication.* This gets at the *reproducibility* standard of scientific research. Despite calls for more replication in the social sciences, replication studies are still rather rare. In part, this is the product of journal editors' and reviewers' strong preferences for original research. It is also due to scholars not making their data publicly available. Many of my colleagues in academia, especially those who conduct experimental (lab experiments) research, do not typically make their data publicly available, though even here anonymized data should be available.

By: Seattle Municipal Archives

Replication datasets are not valuable solely for replication studies. In any dataset there are unused variables. A budding scholar or a research-oriented practitioner might be interested in your "leftover" variables or data points. You can't foresee what others will find interesting.

What You Can Do

If you have data, share it. Not only is this being generous, but there is some evidence it may even be good for your career (citations, etc.). If you don't have the capacity to warehouse it yourself, there are archives available for you. A good choice is Gary King's Dataverse Network Project.

My Data

In the spirit of replication and extension, I would like to let people know which data sources I have available. If you'd like any of it, shoot me a message and we'll figure out a way to get it to you.

Spanish Nationalist Event Data, 1977-1996

First, there is a replication archive of data I used in my dissertation. If you're interested in Spanish nationalist contentious politics – specifically, data on violent and non-violent nationalist protests – check out http://contentiouspolitics.gregorysaxton.net/. This site was set up to make publicly available the data used in my dissertation and subsequent publications. There you will find background information on the project, codebooks, data, and copies of articles published using the data. You can browse and search the data and view various interactive graphs. The entire dataset is also available for downloading.

Twitter Data

Twitter data are generally publicly available. However, if you have a pre-defined set of users you can only grab their latest 3,200 tweets, which in some cases is only one year's worth of data. And in other cases, especially if you want to follow a specific hashtag or collect user mentions or retweets, you can only go back one week in time. For this reason, sometimes it can very helpful if someone else has the historical data you may need. Here are some of the historical data I have, showing the sample of organizations, date range for which data are available, and citations for articles that used the data. If you are interested in it for your own research purposes let me know.

[bibshow file=saxton.bib, format=apa template=av-bibtex-modified]

- *Nonprofit Times 100* organizations – 2009
- *145 advocacy nonprofit organizations* – April 2012
- 38 US community foundations (tweets as well as mentions) – July-August 2011

Facebook Data

Facebook data for organizations is typically public and can be downloaded via the Facebook Graph API. That said, I have some data available on a sample of large nonprofit organizations.

- *Nonprofit Times 100* organizations – December 2009
- *Nonprofit Times 100* organizations – April-May 2013

Website Data

Website data. Historical data can often be gathered from the Internet Archive Wayback Machine, but "robots exclusions" and other errors can prevent this. The following datasets are available:

- 117 US community foundations (transparency and accountability data) – fall 2005 (Saxton, Guo, & Brown, 2007; Saxton & Guo, 2011)[bibcite key=Saxton2007][bibcite key=Saxton2011]
- 400 random US nonprofit organizations – fall 2007 (Saxton, Guo, & Neely, 2014)[bibcite key=Saxton2014]

This is only a partial list of the data I have available. I'll add to this as more data become cleaned and available.

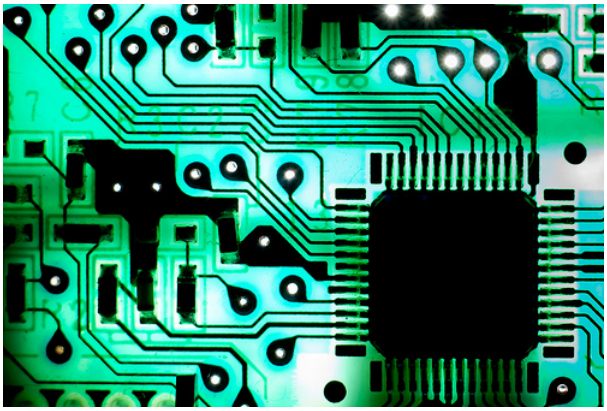
References
[/bibshow]

Python: Where to Start?

For the complete beginner, getting up and running in a new programming language can be a daunting task. I'll try to simplify it here by walking you through the key steps.

Choose a version of Python

There is more than one version of the Python language. Most people choose between version 2.7 and 3.3. Go for 2.7, because some of the utilities you'll be using won't be readily available in the newer 3.3.



Now you have to choose where to get the software. There are lots of choices. If you own a Mac, it comes with Python pre-installed. Forget about that one. For Unix, Windows, and Mac users alike I'd recommend you install Anaconda Python 2.7. This distribution of Python is free and easy to install. Moreover, it includes most of the add-on packages necessary for scientific computing, including Numpy, Pandas, iPython, Statsmodels, Sqlalchemy, and Matplotlib.

Familiarize Yourself with the Basics

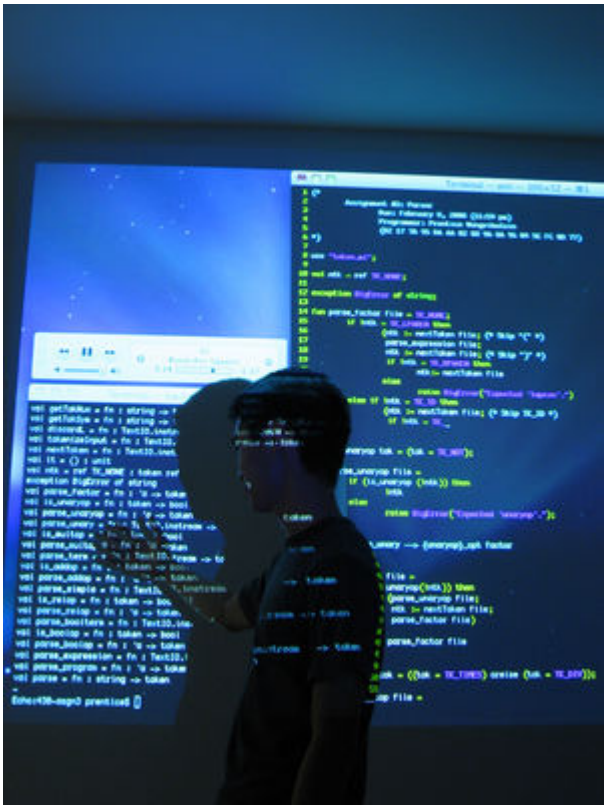
I recommend a two-pronged learning approach: 1) familiarizing yourself with the basics of the language and 2) playing around with other people's "recipes." For the former, I highly recommend doing some of the many excellent tutorials now available online for learning how to use and run Python. A

great place to start is Codeacademy. It takes an estimated 13 hours to complete this self-paced, 12-part interactive tutorial. Do one each day and you'll have a basic understanding in less than two weeks.

Play around with code – Step 1

Python code is written in plain text files, typically given a ".py" extension. Once a script (e.g., *twitter.py*) has been written, it can then be "run" by Python and perform whichever tasks are laid out in the script. So, you'll need to find a good program for editing these text files. On my Mac I use TextWrangler. Vim is popular on other machines. Your choice here is not terribly important. These programs are all generally free and easy to install. The one benefit you'll get from these programs over say, the basic TextEdit app on the Mac is that they will highlight various elements of the Python code syntax, as in the following example, which helps in code formatting:

I would also recommend that you familiarize yourself with the iPython Notebook, which comes included with Anaconda Python. The link provides an overview of the Notebook. Simply put, it has become a boon for *interactive* code development, that is, for "playing around" with the code. I now use the iPython Notebook for developing all of my code. In the same window it allows you to write blocks of code, run them, check whether they worked as intended and, if not, modify them. Highly recommended for learning as it allows for quick error checking.



Play around with code – Step 2

This brings us to the second part of your learning strategy: playing around with other people's code. There are lots of places to find pre-existing code. Just Google it. Just start with something simple, copy it into the iPython Notebook or Textwrangler, etc., and run it. Modify. Play around with. Try to understand it. Annotate your code to help facilitate the learning process.

If you're interested in the same types of questions as I am you may find some of my code useful. A good place to start, once you've made it through some of the CodeAcademy tutorials, is my Python Code Tutorial: Part I.

Ask Questions

You will run into errors. You will run into problems. You will run into roadblocks. The Web will be your friend. If you've checked your code, Googled error messages, and can't find an answer, StackOverflow will be your friend. It is likely

someone has already asked the same question there. If not, ask away and you will typically get an answer the same day. Just respect the community norms and you will find StackOverflow to be a tremendously helpful resource.

Contribute

Python and other open-source communities only thrive through the volunteer efforts of countless individuals across the globe. They will have contributed to your learning, so once you've developed sufficient knowledge give back. Answer others' questions on StackOverflow. Write a tutorial on your blog. Share your code on Github.

Step-by-Step Tutorial

I've also just begun a series of step-by-step tutorials to walk you through installing Python and running you first code. Take a look if you're interested, and please share this on social media if you know others who may be interested.