SI330 Project Proposal Guidelines (with data resource list)

Due: Friday, March 9, 2018 at 11:59:00pm.

This is a brief description of our expectations for your final project, and how to proceed with the first step of the project: writing a one-page proposal. At the end of this document we also provide a list of interesting data sources that you might use as a starting point for project ideas.

Objectives

Overall, the goal of the SI 330 individual project is to give you the opportunity to bring your unique interests, creativity and ingenuity to our data world, demonstrating your ability to apply the skills and concepts that you learn in the course on real datasets of your choice. In particular, your project should show off your ability to take two datasets that have very different formats and/or access methods and manipulate them to combine them and extract a useful byproduct: something more than you could have gotten from either dataset by itself. The manipulation could involve filtering, format conversion, handling missing or noisy data, matching records from one data source with corresponding records in the other, and so on, and must make significant use of programming tools we've covered in class.

Before starting work on the actual project, you'll write a short project proposal that is meant to be a high-level summary and does not need to contain technical details or code. You can adjust your project later if necessary as your project progresses and you get exposed to new methods. Submitting the proposal now is intended to get you thinking about questions and datasets you're interested in, and how those might be answered with the tools we're exploring in this class.

Per the syllabus, the final project is worth 25% of your final grade, of which this proposal counts for 2% and the final report and presentation count for 23%. (More information on the final report requirements will be given later in the term.)

Proposal Guidelines

Use the following outline for your proposal, which shouldn't need more than about one page.

1. Summarize and motivate your proposed project in a few sentences.

- What is your proposed project and why are you proposing it?
- What are the question(s) you want to answer, or goal to achieve?

2. Describe two different data sources you plan to access, manipulate and bring together. The data sources must require different access mechanisms and/or use different data formats. (For example, you might pick one data source that uses a Web API that returns JSON, and the second might use SQL to query a database, or fetch and parse an HTML page.)

For each data set, you should summarize these properties:

- Name

- Size (in records and/or bytes)
- Location (give the URL or other access method)
- Format
- Access method

3. Describe with 1-2 sentences for each point below what data manipulation is likely to be needed:

- What initial processing will have to be done on each?
- How will you combine the datasets, and what will be produced as output?
- What new information will result from combining them?

4. Describe in 1-2 sentences one interesting visualization you could include in a final presentation and report that would show the value/answer a particular question in your final output dataset (that would not be possible with either of the original datasets alone).

You can propose an alternative project structure with prior approval from me. We will give feedback on some project proposals that aren't sufficiently clear or well-chosen.

To help with project ideas, we've posted a list of interesting potential data sources below.

What to submit

Please submit your proposal as a PDF file with the name SI330W18_project_proposal_***youruniquename***.pdf

Ideas for Data Resources

We've gathered a set of a few Web resources and fun links that you might find fun try while exploring project ideas. Disclaimer: We've tried to make these as up-to-date as possible but some of these may be stale (and we would be grateful if you did notify the instructors of any old or new links for this list).

These resources are generally Web APIs that you can call by building and fetching a special kind of URL, which the server will process and return results in XML, JSON, or some other structured format. They provide everything from social media streams to image metadata to question-answering.

Some of the APIs require that users register and get a special "API key" string that you embed in the URL when you make an API call. In one or two cases, I've registered a key for the whole class to use and noted it in the description.

This is just a small (but high-quality) sample of what's out there. Please do share any cool resources you find with the instructors and we will post them here.

Meta-resources

Kaggle.com (https://www.kaggle.com) is the go-to place for data sets and sample analyses for data scientists.

This is the **Programmable Web** meta-directory I showed you in the first lecture that has thousands of data sources. I recommend starting here: http://www.programmableweb.com/apis/directory

This website **has dozens of public datasets -** some fun, some a bit, well.. quirky. http://rs.io/100-interesting-data-sets-for-statistics/

This website also **has dozens of public datasets in CSV format**, many from scientific studies or survey - also sometimes quirky.. https://vincentarelbundock.github.io/Rdatasets/datasets.html

The **Academic Torrents site** has a growing number of datasets, including a few text collections that might be of interest to some of you (Wikipedia, email, twitter, academic, etc) for current or future projects.
http://academictorrents.com/browse.php?cat=6

Wolfram | Alpha API A computational knowledge engine with question-answering abilities and access to large databases of the world's facts
Experiment with the urllib2 module and a URL like this: http://api.wolframalpha.com/v2/query?input=XXXXX&appid=459LEL-W3KJ88H8Y7
For example: http://api.wolframalpha.com/v2/query?input=canada%20population&appid=459LEL-W3KJ88H8Y7
where you replace the XXXXX with what you want to know. NOTE: the '&appid=459LEL-W3KJ88H8Y7' part is vital; it is a W|A AppID I got for the class. Feel free to use that one.
You can get your own and read more about the API, here: http://products.wolframalpha.com/api/
And you can explore how it works here: http://products.wolframalpha.com/api/explorer.html

Visualization Resources
Simple interactive timelines using TimelineJS. http://timeline.knightlab.com/
Geographic charts using OpenHeatMap. http://www.openheatmap.com/
And plot.ly: https://plot.ly/

Very good example analysis: https://www.kaggle.com/mrisdal/titanic/exploring-survival-on-the-titanic

Text Analytics Resources

If you're thinking about a project that involves text (e.g. twitter, facebook, text from web pages, etc) there's an online API service from datumbox that might be useful for doing text analytics: it supports keyword extraction, about 14 flavors of text classifier, and document similarity.

http://www.datumbox.com/machine-learning-api/

You send it API calls in the form of HTTP requests, and it returns the results in JSON.

I've got an API key for the class:

d3ce53ca1cead4e08490df097c890967

You can use this with the "API Sandbox" to check out the results from various API calls with your sample text.

There’s also monkeylearn.com, which we touched on in the NLP section of the course.

Geographic resources

Bing Maps API http://www.microsoft.com/maps/developers/web.aspx

Bing Spatial Data Services http://msdn.microsoft.com/en-us/library/ff701734.aspx

Google Time Zone API https://developers.google.com/maps/documentation/timezone/

World Bank API: http://data.worldbank.org/developers/api-overview

Google offers a wide variety of APIs: https://developers.google.com/maps/web-services/

Search and Language resources

Bing Search API http://sdrv.ms/17Le3x5

Microsoft Translator API http://www.microsofttranslator.com/dev/

Microsoft Academic search API http://academic.research.microsoft.com/about/Microsoft%20Academic%20Search%20API%20User%20Manual.pdf

Google Books n-gram corpus http://books.google.com/ngrams

ngram Dataset on AWS: http://aws.amazon.com/datasets/8172056142375670

Web collections

Common Crawl: • Currently 6 billion Web documents (81 Tb) • Amazon S3 Public Data Set http://aws.amazon.com/datasets/41740

https://commoncrawl.atlassian.net/wiki/display/CRWL/About+the+Data+Set

Award project using Common Crawl: http://norvigaward.github.io/entries.html Python example: http://www.freelancer.com/projects/Python-Data-Processing/Python-script-for-CommonCrawl.html

URL Search: http://urlsearch.commoncrawl.org/?q=research.microsoft.com Downloadable JSON metadata file with the address and offset of the data for each URL

Business/commercial data

Yelp http://www.yelp.com/developers/documentation/v2/search_api

Lots of options at www.kaggle.com

Social media

Twitter APIs Documentation https://developer.twitter.com/

Random inspirational examples

The programming language popularity visualization I showed in the first lecture
http://www.dataists.com/2010/12/ranking-the-popularity-of-programming-langauges/

HTTP Archive + BigQuery = Web Performance Answers
http://www.igvita.com/2013/06/20/http-archive-bigquery-web-performance-answers/

Which words are associated with geeks vs nerds?
http://slackprop.wordpress.com/2013/06/03/on-geek-versus-nerd/

Digital History Hacks:
http://digitalhistoryhacks.blogspot.com/2007/01/exploratory-bibliography.html
"The first step is to get a complete set of recommendations via the Amazon API. We pass my blog post through a simple scraper to get the ASINs (Amazon Standard ID Numbers) for each book on the list. In a loop, we then submit each ASIN to Amazon and get back an XML file that includes ASINs for recommended books. We create a big list of (ASIN, ASIN) pairs. Each one of these is a recommendation: customers who bought the first book also bought the second one. Since we will want to play with these data later on, we use a high-level Python module called pickle to save them to disk. Python source code for the first step is here.
Now that we've got a big list of recommendations, the second step is to concentrate on the books that come up most frequently. We unpickle our data then create a list of recommended books and filter out any that appear on the original list. We count the number of times each is recommended and sort to create a frequency list with most frequently recommended books at the top of the list. Python source for the second step is here." (see site)
And follow-on work using Amazon and graph visualization here:
http://babbaging.blogspot.com/2007/04/python-amazon-graphs-oh-my.html