Text Data Sources

March 22, 2012

These activities are intended to provide starter code for Project 2. Download `DataImport.py` and open it in IDLE.

Project Gutenberg

1. Go to `http://www.gutenberg.org/`. Search for a book of your choice.

   What formats can we download these books in?

   We will want `Plain Text UTF-8` files. What size is the file? `Mb` stands for Megabyte and `Kb` stands for Kilobyte (they might be capitalized differently). 1 Mb is the same as 1024 Kb.

2. If the book you found is larger than 1Mb, find a book that is smaller than 1Mb.

3. Download the text file (by right-clicking and selecting `Save Link/Target As...`). Open it in a text editor (such as Notepad).

4. There are some lines that have to do with Project Gutenberg. The function `removeLicenseFromProjectGutenberg` in `DataImport.py` removes most of these lines.

   Before running the function, look at the program. What are the inputs? What are the ouputs?

   There are three things that this program does to "clean up" the text. What are they?

5. Run `removeLicenseFromProjectGutenberg` with the file you downloaded an specify an output file name of your choice. Open up the resulting output file and verify that it worked.

6. Note that there are some **rules** when using data from Project Gutenberg (which can be found online as well as in the license information). Make sure you follow these instructions if you use this type of data.

English Dictionary

1. Project Gutenberg contains the 1913 US Webster's Unabridged Dictionary, but it has lots of formatting and extra information. Fortunately, Ralph S. Sutherland has a simpler version of the dictionary at `http://www.mso.anu.edu.au/~ralph/OPTED/`.

2. This data is formated so there is one definition per line. What are the three things that each line should have?

3. Click on the letter with the smallest file size (again, these sizes are in Megabytes or Kilobytes).

4. Right-click the page and select `View Page Source...`. We want to turn this text file into a tab-delimited file that JUST has the three items specified above. Very generally, can you explain what patterns you would want to identify if you were designing a regular expression for this? Remember that if you say a phrase, it can almost always be converted into a regular expression.

5. The function `getWebsterDictionary` in `DataImport.py` takes, as input, a letter of the alphabet and the name of an output file to write to. It pulls the webpage from Sutherland's website, removes the extra formatting, and writes an output file that is TAB delimited (remember tabs are represented as `'\t'` in Python).

6. Run `getWebsterDictionary` on the letter you looked at above and an output file name of your choice. When it is done, open the output file and verify that it worked.

7. Again, there are some **rules** when using data (available on the website). Make sure you follow these instructions if you use this type of data.

Twitter Data

We have been collecting tweets since Jan. 23, 2011. See the lecture for more information about how they were collected.

1. Download and save `Tweets_With_Elect_Mar20.txt`. This is a list of all the tweets from Tuesday that contained the regular expression `'[eE][lL][eE][cC][tT]'`. There are 1,043 of them.
2. Open it in a text editor (such as Notepad). Each line has the following columns (separated by commas):
   (a) Unique ID of the tweet
   (b) Time of the tweet (in `unix time`: the number of milliseconds that have passed since Jan. 1, 1970)
   (c) Source of tweet
   (d) User name of tweeter
   (e) ID of tweeter
   (f) Latitude of tweet location
   (g) Longitude of tweet location
   (h) Country of tweet (if known)
   (i) Place of tweet (if known)
   (j) Retweet count
   (k) `true` if the tweet is favorited, `false` otherwise
   (l) `true` if the tweet is a retweet, `false` otherwise
   (m) Tweet
3. Consider the following example line: `1,934,‘‘Twitter for iPhone’’,‘‘Bart Simpson’’,32,98453,-87.45984,‘‘United States’’,‘‘’’,0,false,false,‘‘Don’t have a cow, man!’’`
   Who is the tweeter? What is the tweet?
4. Suppose we want to split the line above into the 13 columns. Can we use the `split` function and split on commas?
5. The function `twitterExample` is `DataImport.py` takes an integer `n` and prints the first `n` tweets in `Tweets_With_Elect_Mar20.txt`. Run the `twitterExample` with a number of different `n` values.
6. What are the tweets in this file about? Are they all about the election?

American Presidency Project

The American Presidency Project `http://www.presidency.ucsb.edu/` is an archive that contains hundreds of thousands of documents related to American politics. This site was used in the Politilines example we looked at in the beginning of the semester (`http://politilines.periscopic.com/`).

1. Go to the Debates site: `http://www.presidency.ucsb.edu/debates.php`. Click on `Republican Candidates Debate in Mesa, Arizona`.
2. First look for patterns in the text. How do we know who speaks when? Very generally, how would you design a regular expression to get the speaker and the phrases for each line/set of lines?
3. Right-click on the page and view the page source. We want to get the transcript from this file. Where is the text that we want to extract? Hint: search for a couple phrases you KNOW appear in the transcript.
4. The function `getTranscript` in `DataImport.py` pulls the source from a url, formats the text, and writes the result to an output file. Run `getTranscript` on the URL for the Arizona Republican Debate. Open the file and verify that it worked.