Lab Assignment 2: More with Twitter feeds

In this assignment we will continue our exploration of Twitter data. A starting file titled assignment2.py is provided to you and can be found at the course's GitHub page[1], and you will fill in the missing parts. You will need a small JSON file called keys.json at the same level, that contains your Twitter access information, as described in the notes. You should already have this in place from the first assignment.

You should be looking at the assignment2.py file as you read along.

The first 65 lines of the script are very similar to the ones from the first assignment. The only difference is that now you have a function called getTweets that you can use to retrieve all tweets for a given query. Read the function's description and look at the use cases on lines 68-69 to know how to use it. The first couple of questions won't require you to use this function, as you will be working with the two pre-fetched examples of tweets from the two presidential candidates, which are saved to the variables hillary and donald.

1. Write a function called getHashtags which takes as arguments a list of tweets and returns a list of pairs of a hashtag string and the number of occurences of that hashtag in the list of tweets. For instance the result might look like this: [("#theBest", 15), ("#donaldForPrez", 10), ...]. The list should be sorted with most frequent hashtags first.
2. Write a function called getHashtagsDictthat takes the same input as the functiongetHashtags' but it instead returns a dictionary whose keys are the different hashtags and whose values are the corresponding counts.
3. Create a list of those hashtags, if any, that have been used by both candidates. The entries in the list should be dictionaries with three entries, one called "hashtag" with value the tag string, one called "hillary" and one called "donald". The values for the two candidates should themselves be dictionaries, containing two entries: the "count" of tweets containing the tag and the "percent" of tweets from that candidate containing the tag. To help you in it you should create a couple of helper functions:
   - A function that takes a count and a total and returns a dictionary containing an entry with the count and an entry with the percent, computed as the count over the total. This is the kind of dicionary that you would associate with the candidate entries.
   - A function that takes as input the hashtag string, the counts for the two candidates and the totals for the two candidates, and creates the dictionary entry that contains the "hashtag", "hillary", "donald" entries.
4. Count how many times each candidate has mentioned their opponent in a tweet, as well as the percent of tweets where they have done so. You should look for uses of the candidates' first and/or last names in the tweet text both upper and lower case, and not just a "mention" in the Twitter sense.

[1]https://github.com/skiadas/DataWranglingCourse/blob/gh-pages/assignments/assignment2.py

5. Count how many and what percent of each candidates' tweets have been actual tweets and not retweets.
6. Using the timestamp information that is part of each tweet, create a list of triples, where each triple consists of a date, the number of tweets that Hillary Clinton had that day, and the number of tweets that Donald Trump had that day. Then print the information in that list in a column format, with one row for each day, making sure that the numbers align and having titles for each column.
7. The next two problems are somewhat harder than the earlier problems. We will create a function that returns all a users's followers, as long as there are no more than 5000 of them. It will have to steps. In this step, you will implement a function called fetchFollowerIds. It will be given a user's screen name and it will return the ids of the users following that user. It will only return the first 5000 hits, and for now we are going to be OK with that.

   - You should read in this page[2] about the kind of request you would need to perform, and what kind of result you should get back. A sample call to get you started is mentioned right above the function in the assignments file.
   - You will need to use oauth.get directly, with an appropriate query. You should read in this page[3] about the format of the "response" object that this call produces, and in particular how to get at the JSON content in it.
   - The JSON content is going to have some more stuff in it. You should extract the appropriate list before returning it.

8. In this problem we continue from the previous one, and we will implement a function called fetchUsers. It will take as input a list of ids and is supposed to return a list of objects corresponding to those users.

   - You should read in this page[4] how such a request should look like.
   - You should use oauth.post for this, read this page[5] for how to do that.
   - These requests are limited to 100 user ids. You will therefore need to go through the list of ids you have, and work on it 100 at a time, making a request for each group of 100 and accumulating the results. You should NOT make a request for each id individually, you should make batch requests for 100 ids at a time.

9. Combine the two previous methods to write a function getFollowers that takes as input a screen name and returns the list of follower objects, essentially calling fetchUsers on the result of calling fetchFollowerIds. You should test your function with the user with screen name "HanoverCollege" (2804 followers total).

You should submit your completed Python file as an email attachment to me. The name of the file should include your first and last name, in addition to the assignment's number. It should contain no whitespaces.

[2]https://dev.twitter.com/rest/reference/get/followers/ids
[3]http://docs.python-requests.org/en/master/user/quickstart/#response-content
[4]https://dev.twitter.com/rest/reference/get/users/lookup
[5]http://docs.python-requests.org/en/master/user/quickstart/#response-content