Homework 03 Instructions STAT/CS 287

We've learned a little about working with text data from previous assignments. Now let's study some data from a structured format: Twitter's JSON API.

Please read these instructions completely before you begin.

To submit:

1. Compress your HW03 working directory as per the "preparing and submitting your homework" slides. Make sure the zipped file includes your `.py` script file(s), your writeup, and any other files you may have generated while completing the assignment. Please do not include the original Twitter data in your submission.
2. Upload your zipped file to Blackboard. No other files should be submitted.

Please come to office hours with questions!

Twitter analysis

I have used Twitter's "streaming" API to gather a corpus of approximately 75k public tweets that occurred in a two-week period immediately before the **2012 presidential election**. All tweets are related to either Barack Obama or Mitt Romney.

We want to find out if there are differences in the texts of tweets related to Obama compared with the texts of tweets related to Romney. Do people choose **different words** in these different contexts? Are people more **negative** when discussing one candidate compared with the other?

Processing natural language programmatically is extremely challenging, so we'll focus here on a **word frequency analysis**.

Twitter data

I have saved the tweets in a single compressed text file for you to use. Download this file and save it to your HW03 working directory.

- Unfortunately, some number of Twitter records have been corrupted due to problems with their API servers. You will need to **repair** these records.

Each line of this file is a JSON string encoding a single tweet as a dict of all of the data given from Twitter's API. Using the `gzip` module, you can loop over a gzipped file just like a regular text file **without unzipping it**. Here's a snippet of code that loops over the data and makes a dict, called `tweet`, for each tweet:

```
import json
import gzip

# In my case, I have the json file in a subfolder called HW03_data
for line in gzip.open("HW03_data/HW03_twitterData.json.txt.gz", 'rt', encoding='utf-8'):
    tweet = json.loads(line.strip())
```

(We will learn more about encodings soon.)

Notice how `gzip.open` makes it easy to loop over the compressed file line-by-line!

Each `tweet` dict provides a lot of data. For our purposes here, you will only need to use two keys: `"created_at"` and `"text"`:

- `tweet["created_at"]` is a string storing the timestamp for when the tweet was written. You will need to parse this using `time`, `datetime`, etc. Be sure to consult the lecture notes, as needed.
- `tweet["text"]` stores the text of the tweet itself.

It may take several hundred MBs of RAM to load all tweets into memory at the same time. You may have more luck loading each tweet one at a time and then keeping just the `"created_at"` and `"text"` data.

```
for line in gzip.open("HW03_data/HW03_twitterData.json.txt.gz", 'rt', encoding='utf-8'):
    tweet = json.loads(line.strip())
    print(tweet["created_at"], type(tweet["created_at"]))
    break # just print the first tweet
```

P1. Gather and process the data

First, download and store the data. As mentioned, some lines will be corrupted. Please fix these corrupted lines programmatically and keep records for how many and what kind of "repairs" were necessary.

Next, figure out a way to determine if a tweet was written about Obama or Romney. To do this think about **all the ways Obama's name can appear in a string of text, and likewise for Romney**. "Obama," "Barack," "Mitt," "Barry"? How will you deal with UpPeR and lower case letters? Punctuation? Will "Barry" give false positives?

- Recall the function I wrote for loading A Tale of Two Cities. Some of its techniques may prove useful for you to **sanitize** the tweet texts. For **reproducibility** keep a proper description of what you are doing to the tweet text and why!
- Python makes it very easy to determine if a substring is present in a string, using the `in` operator:

```
if "alpha" in "alphabet":
    print("substring 'alpha' found!")
```

Build a single text "corpus" of all tweets about Obama, and another of all tweets about Romney. One tweet may appear in both corpuses if the tweet mentions both candidates.

- You can store a corpus as a list of strings, or one giant string, or however else you prefer.

In your writeup, please answer the following:

- P1.1 - How did you retrieve the data? Where do you store the data so that you only download it once?
- P1.2 - When trying to load the data into Python, how many records needed to be repaired for each type of error you encountered? After repairs, how many Twitter records are present in the data?
- P1.3 - What Python data structure did you use to work with the data? Describe how and where in your code you loaded the data (please reference specific Python file names and line numbers).
- P1.4 - Describe the criteria you used to split the tweets into the "O" and "R" corpora (please reference specific Python file names and line numbers).

Be sure to *show all your work*. Specifically, any processing or changes you do to the data that are not documented in your submitted code and writeup risk losing points. It is critical to keep an exact, correct record of all data manipulations you do as data scientists.

P2. Count the tweets

Next, make a **time series plot** of the number of tweets about Obama and Romney as a function of time, one curve for each candidate (Blue for Obama, Red for Romney?):

- Consult lecture notes for information on plotting timestamped data with matplotlib.
- To "smooth" the time series, **truncate each timestamp to the hour**. So a tweet from 12:17 PM would be at 12, for example. The simplest way to do this is to chop off the minutes and seconds characters from the `"created_at"` string before you parse it into a date.
- Don't forget to label your figure axes.

Save this plot to a file using the matplotlib's `plt.savefig` command and include it in your writeup. Please ensure the text within your figure is clear and legible. Add a descriptive caption so a reader can understand the contents of the figure.

P3. Analyze tweet text

Now, let's analyze the "O" corpus and the "R" corpus to see if we can learn anything interesting about their respective word choices.

Compute the number of times a word appears in either corpus. Let's call this $f(w)$, so that $f_{\mathrm{R}}(w) = 18$ means that word w appeared 18 times in the Romney corpus. This does not mean that w appeared in 18 tweets, because the same word can be used multiple times in a tweet.

Now, we need a good way to quantify that a word is very **common** in one corpus and simultaneously **rare** in the other. Yule (1900, 1912, 1944) introduced a nice coefficient for exactly this purpose. To compare word w in the O and the R corpus, compute:

$$c(w) = \frac{f_{\mathrm{O}}(w) - f_{\mathrm{R}}(w)}{f_{\mathrm{O}}(w) + f_{\mathrm{R}}(w)}$$

If a word appears almost entirely in the O corpus, and almost never in the R corpus, it will have $c \approx 1$. And $c \approx -1$ means the word is heavily used in Romney tweets and rarely used in Obama tweets.

- P3.1 - Can you think of any problems with this choice of coefficient $c(w)$? Do you know of any better alternatives?

Please compute c for all unique words that appear in **both** the O corpus and the R corpus. As output **save to a `txt` file** the 100 most "O-slanted" words (100 largest c values) and the 100 most "R-slanted" words (100 smallest c values). The output should be 100 lines and four space-separated columns:

```
wordI 0.99833 wordJ -0.99812
wordX 0.92423 wordY -0.93433
.     .       .     .
.     .       .     .
.     .       .     .
```

and so forth.

- P3.2 - Include these 100 formatted lines in your writeup.
- P3.3 - Take a look at the words you find as you answered 3.2. Do they make sense? Write a few sentences in your writeup describing what you've found. Does any information about the 2012 election pop out at you?

Disclaimer You may see naughty words in the tweet text, just as you would using the Twitter website (or, honestly, almost anything online). If this will be a problem, please let me know! But keep in mind that it's going to be awfully hard to be a data scientist without seeing the occasional naughty word in your "experiments".

References:

1. Yule, G. U. (1900). On the association of attributes in statistics: with illustrations from the material of the childhood society, &c. Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character, 194, 257-319.
2. Yule, G. U. (1912). On the methods of measuring association between two attributes. Journal of the Royal Statistical Society, 75(6), 579-652.
3. Yule, C. U. (1944). The statistical study of literary vocabulary. Cambridge University Press.