Assignment 2

Ryan Dill
CS 432
February 10, 2018

1. Write a Python program that extracts 1000 unique links from Twitter. Omit links from the Twitter domain (twitter.com). You might want to take a look at:

https://pythonprogramming.net/twitter-api-streaming-tweets-python-tutorial/
http://adilmoujahid.com/posts/2014/07/twitter-analytics/

see also:

http://docs.tweepy.org/en/v3.5.0/index.html
https://github.com/bear/python-twitter
https://dev.twitter.com/rest/public

But there are many other similar resources available on the web.
Note that only Twitter API 1.1 is currently available; version 1
code will no longer work.

Also note that you need to verify that the final target URI (i.e.,
the one that responds with a 200) is unique. You could have many
different shortened URIs for www.cnn.com (t.co, bit.ly, goo.gl,
etc.). For example:

```
$ curl -IL --silent https://t.co/DpO767Md1v | egrep -i "(HTTP/1.1|^location:)"
HTTP/1.1 301 Moved Permanently
location: https://goo.gl/40yQo2
HTTP/1.1 301 Moved Permanently
Location: https://soundcloud.com/roanoketimes/ep-95-talking-hokies-recruiting-one-week-before-signing-day
HTTP/1.1 200 OK
```

You might want to use the streaming or search feature to find URIs. If
you find something inappropriate for any reason you see fit, just
discard it and get some more links. We just want 1000 links that
were shared via Twitter.

Hold on to this collection and upload it to github -- we'll use it
later throughout the semester.

For question one I made three separate python scripts.

- *assignment2.py*
- *assignment2_2.py*
- *assignment2_3.py*

Assignment2.py used tweepy to collect links from Twitter. I collected a total of 1000 links.

Assignment2_2.py is used to sort the total collection of links into groups that return either a 200, 300, or 400 message.

Assignment2_3.py is used to remove all the duplicate links, remove all the twitter links, and to display the total number of lines in each file.

Out of the 1000 links that were gathered, 753 reached a server code of 200. 195 had a server code in the 300 range. 52 had a server code in the 400 range. After removing the twitter links and duplicates, there were a total of 380 total links remaining.

Here are 10 links from the cleaned URLs:

http://www.espn.com/mens-college-basketball/story/_/id/22396698/top-16-seeds-ncaa-tournament-be
https://www.sports.ru/basketball/1060114712.html
https://newstrandblog.wordpress.com/2018/02/11/northern-kentucky-university-vs-green-bay-basketball-feb-10-7/
http://warrennolan.com/basketball/2018/nitty-live
https://www.instagram.com/p/BfEMWLzn_ER/
https://www.instagram.com/p/BfEJ_wLF3lU/
http://www.tri-cityherald.com/sports/high-school/prep-basketball/article199544294.html
http://watchsoccergames.live/game-thread-syracuse-vs-wake-forest-0100-pm-est-ncaa-basketball-livestream/
http://www.northwestgeorgianews.com/calhoun_times/sports/high_school/prep-basketball-jackets-continue-streak-by-winning-sixth-straight-region/article_1401a2de-0ee9-11e8-b0fd-5f47ad09d9e5.html
https://basketball.realgm.com/wiretap/248945/Rockets-Sign-Then-Waive-Bobby-Brown

2. Download the TimeMaps for each of the target URIs. We'll use the ODU
Memento Aggregator, so for example:

URI-R = http://www.cs.odu.edu/

URI-T = http://memgator.cs.odu.edu/timemap/link/http://www.cs.odu.edu/

or:

URI-T = http://memgator.cs.odu.edu/timemap/json/http://www.cs.odu.edu/

(depending on which format you'd prefer to parse)

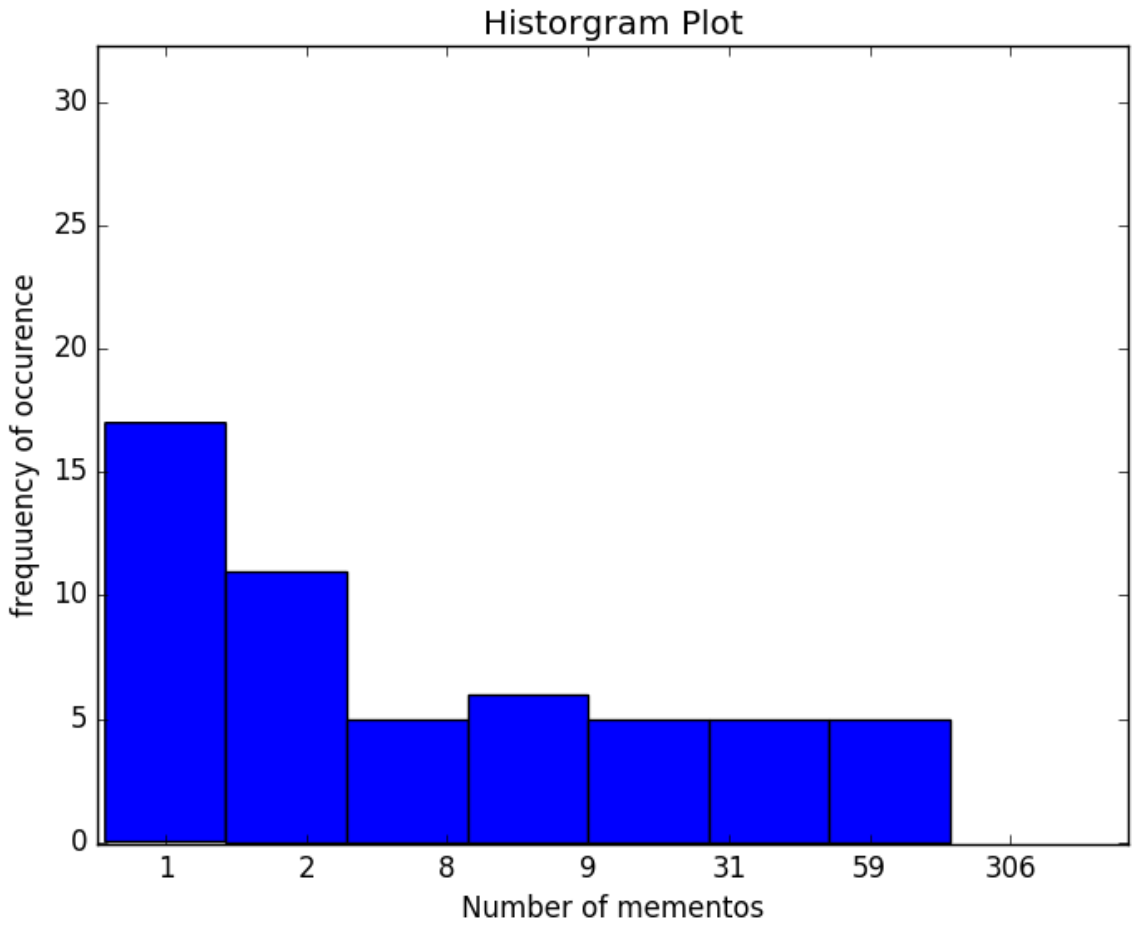
Create a histogram* of URIs vs. number of Mementos (as computed from the TimeMaps). For example, 100 URIs with 0 Mementos, 300 URIs with 1 Memento, 400 URIs with 2 Mementos, etc. The x-axis will have the number of mementos, and the y-axis will have the frequency of occurence.

* = https://en.wikipedia.org/wiki/Histogram

What's a TimeMap?
See: http://www.mementoweb.org/guide/quick-intro/
And the week 4 lecture.

I created a program, assignment2_4.py that attached the memgator suffix to each of the URI's. There were a total of 380 links that were processed.

This is a histogram of the data.



3. Estimate the age of each of the 1000 URIs using the "Carbon Date" tool:

http://ws-dl.blogspot.com/2017/09/2017-09-19-carbon-dating-web-version-40.html

Note: you should use "docker" and install it locally. You can do it like this:

http://cd.cs.odu.edu/cd?url=http://www.cs.odu.edu/

But it will inevitably crash when everyone tries to use it at the last minute.

For URIs that have > 0 Mementos and an estimated creation date, create a graph with age (in days) on the x-axis and number of mementos on the y-axis.

Not all URIs will have Mementos, and not all URIs will have an

estimated creation date. Show how many fall into either categories.
For example,

total URIs: 1000
no mementos: 137
no date estimate: 212

These are the totals for the URIs submitted to the carbon date website.

Total URIs: 380

no mementos: 197

no date estimate: 2