Assignment 6

CS 432

Breon Day

Question

1.Use D3 to visualize your Twitter followers. Use my twitter account "@phonedude_mln") if you do not have >= 50 followers. For example @hvdsomp follows me, as does @mart1nkle1n. They also follow each other, so they would both have links to me and links to each other.

Solution

Noticing that this had a similar requirements as assignment two i started by using what i had available to set up authentication.Originally i had planned to use tweepy but i found that since the twitter api already returned what i needed in the json format it would be an unnecessary step to take the raw friendship objects returned and sift through them.

note*the references comment will be brought up in the third screenshot

```
from twitter import *
import json
import time
# references https://github.com/ideoforms/python-twitter-examples#
access_token = "829769241068883968-WyE7fyJDMTqToDI8WwwnOfWsdyDvJI5"
access_token_secret = "qCGddiCfrdEYysOJPggkeV5I2AoV2tTYM1225CTOwLGWR"
consumer_key = "enM0tpExhEfj5OWqp8I6KoBh1"
consumer_secret = "mZUL7OVq8NjOWTecsmyUXFjWACYPPQoqU4mNeBz3r9EExLUIYH"
twitter = Twitter(auth=OAuth(access_token, access_token_secret, consumer_key, consumer_secret))
```

Due time constraints i lowered the required follower count and searched through popular twitter users followers until i found two users with low enough follower accounts that satisfied the requirements but still allowed for timely data processing (54+main user)*(54/2)/(180*15)=123.75 minutes, CarmeTaika was found on the first page of https://twitter.com/pewdiepie/followers

```
#username = "phonedude_mln"
#arbitrarily chosen due to the 70 followers count they both follow the pdp
#username = "famato96"
#arbitrarily chosen due to the 54 followers count
username="CarmeTaika"
```

The main user id along with the followers ids were taken and combined using https://dev.twitter.com/rest/reference/get/followers/ids and https://dev.twitter.com/rest/reference/get/users/show respectivly
Examples taken from the referenced link above
The boolean sourceNotAppended exists to ensure its only added once
The lists nodes,links and uris were used to build the output

```
query = twitter.followers.ids(screen_name=username)
mainUserId=twitter.users.show(screen_name=username)
ids=query["ids"]
#add main user id to the id list at the beginning
ids.insert(0,mainUserId["id"])
sourceNotAppended=True

nodes= []
links=[]
subquery=twitter.users.lookup(user_id=ids)
print (subquery)
uris=[]

n=len(ids)
```

The structure of this double for loop ensured i didn't waste time processing the same relationships twice and with a wait time of 5.08 seconds 180*5.08=914.4 i stayed under the 180 requests per 900 seconds. On the initial run the main user was appended,then results of the relationships were appended to the list to be used in building the json

```
for i in range(0,n-1):
    for j in range(i+1,n):
        try:
            count=count+1
            print(count)
            result = twitter.friendships.show(source_id=ids[i], target_id=ids[j])
            source = result["relationship"]["source"]["screen_name"]
            print(source)
            follows = result["relationship"]["source"]["following"]
            target =result["relationship"]["target"]["screen_name"]
            followed_by =result["relationship"]["target"]["following"]
            if i<=0:
                if(sourceNotAppended):
                    nodes.append({"id": source})
                    sourceNotAppended=False
                nodes.append({"id": target})
            print("*-------------------------------------------------------------------*")
            print(source,"follows",target,follows)
            print(source,"followed by", target, followed_by)
            print("*-------------------------------------------------------------------*")
            if(follows):
                links.append({"source": source, "target": target})
            if(followed_by):
                links.append({"source": target, "target": source})
            time.sleep(5.08)
        except:
            continue
```

the jsons were then printed for uri's,names and connections

```
# create a json of screen names
print(json.dumps({"nodes": nodes, "links": links},indent=1, separators=(',', ': ')))
# create a json of uris
print(json.dumps({"URIS": uris},indent=1, separators=(',', ': ')))
```

For the html i chose to use the framework provided by https://bl.ocks.org/mbostock/4062045
A minor addition allowed for names to be made visible

```
var nodelabels = svg.selectAll(".nodelabel")
            .data(graph.nodes)
            .enter().append("text")
            .attr("class", "label")
            .attr("fill", "black")
            .style("font-size", "8px")
            .text(function(d) {  return d.id;  });
```

Sample output and raw html

http://htmlpreview.github.io/?https://github.com/BreonDay/cs532-s17/blob/master/Submissions/A6/Src/D3.html