CS595 Intro to Web Science, Assignment #5

Valentina Neblitt-Jones

October 21, 2013

The “friendship paradox” (`http://en.wikipedia.org/wiki/Friendship_paradox`) says that your friends have more friends than you do.

Question 1

Determine if the friendship paradox holds for your Facebook account. Create a graph of the number of friends (y-axis) and the friends sorted by number of friends (x-axis). (The friends don't need to be labeled on the x-axis.) Do include yourself in the graph and label yourself accordingly.

Compute the mean, standard deviation, and median of the number of friends that your friends have.

You can download your network in an XML file by using the NameGenWeb Facebook app:

`https://apps.facebook.com/namegenweb`

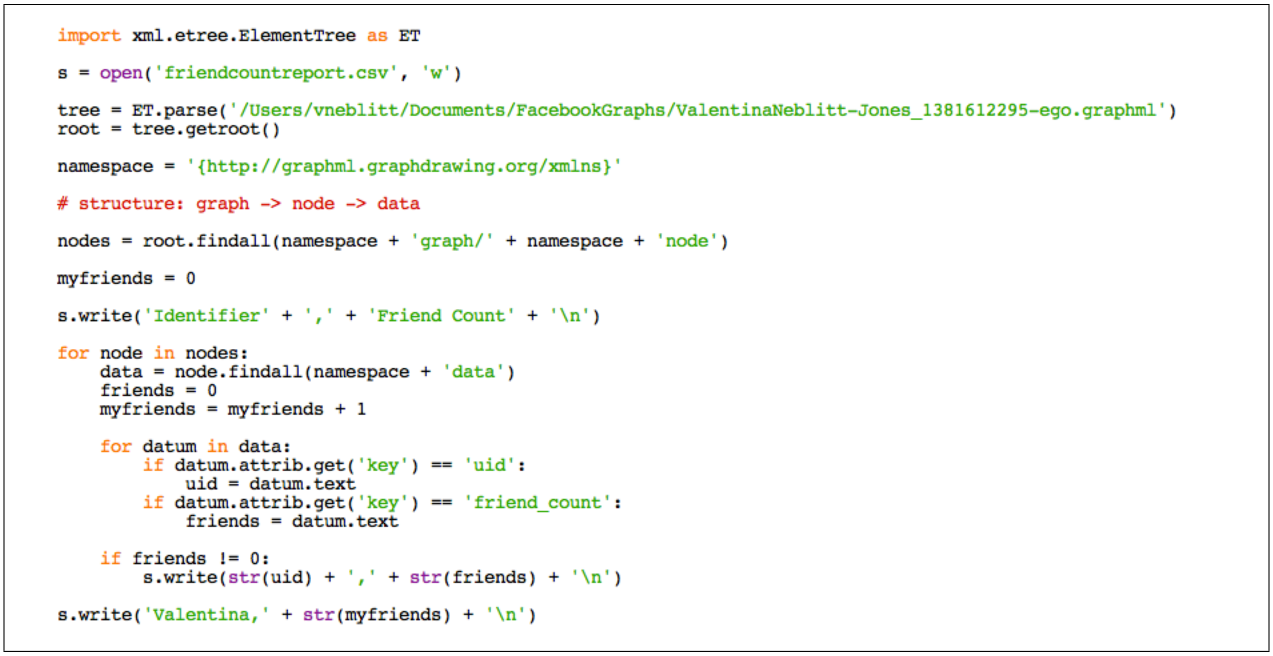
You will need to give this app permission to access your Facebook data. Make sure you select ”Friend Count” as an Extended Attribute. When you download the data, download it in the GraphML format.

If you do not have a Facebook account, email me and I will send you my GraphML file.

Answer to Question 1

My GraphML file is available if necessary, but since it had real names in it I left it on my hard drive instead of loading it to GitHub. When I chose the anonymize option for generating the graph, it also did not provide the friend count so I had to run it without that option. The nodes in the file also contained a unique number for each person and in DoD we would say that it is sufficient for PII violation, but I could find no way to use the number to identify a person in Facebook. I did keep the names on a version of my program because I was curious about who my most popular friend was. Surprise! It was a computer scientist.

I used ElementTree to parse my XML with Python. There was a lot of misunderstanding as I tried to understand how to step through this XML to get at the elements I actually needed.



```
import xml.etree.ElementTree as ET

s = open('friendcountreport.csv', 'w')

tree = ET.parse('/Users/vneblitt/Documents/FacebookGraphs/ValentinaNeblitt-Jones_1381612295-ego.graphml')
root = tree.getroot()

namespace = '{http://graphml.graphdrawing.org/xmlns}'

# structure: graph -> node -> data

nodes = root.findall(namespace + 'graph/' + namespace + 'node')

myfriends = 0

s.write('Identifier' + ',' + 'Friend Count' + '\n')

for node in nodes:
    data = node.findall(namespace + 'data')
    friends = 0
    myfriends = myfriends + 1

    for datum in data:
        if datum.attrib.get('key') == 'uid':
            uid = datum.text
        if datum.attrib.get('key') == 'friend_count':
            friends = datum.text

    if friends != 0:
        s.write(str(uid) + ',' + str(friends) + '\n')

s.write('Valentina,' + str(myfriends) + '\n')
```

Figure 1: Showing Friend Count

Facebook states that I have 240 friends. The GraphML file only produced 230 friends. I was unable to discover what was at the root of this inaccuracy and what could have been special about the 10 that were left out of the file.

However, I was able to figure out a different peculiarity with the data. I discovered when reviewing the GraphML file that some nodes did not contain a Friend Count. I investigated two of those cases and found a difference in summary friends data. Figure 2 shows a friend whose friend count was in the file and Figure 3 shows a friend whose friend count was not in the file. Some of my friends only allow me to see our mutual friends not all of their friends and therefore the application was unable to pull a friend count for those individuals. According to the graph file, I have 230 friends, but my report on friend counts only has 208 of those friends' friend counts. This means that 22 of my friends have the mutual friends only setting. I omitted nodes that did not contain a friend count (Figure 4).

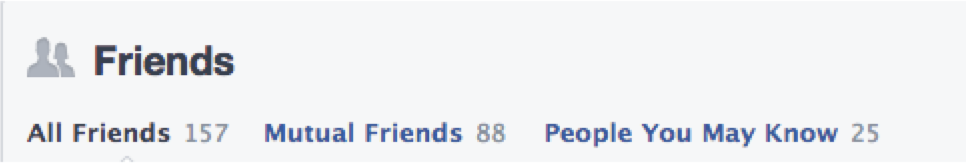


Figure 2: Showing Friend Count

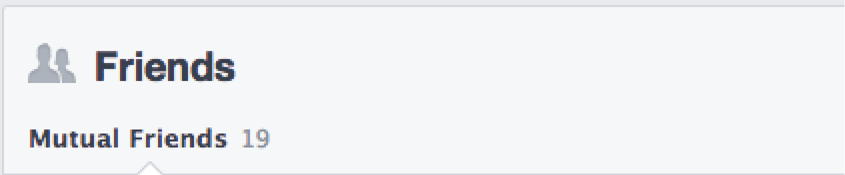
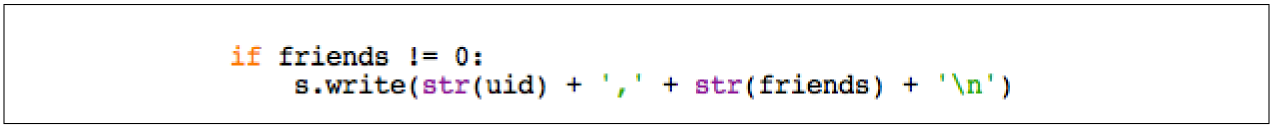


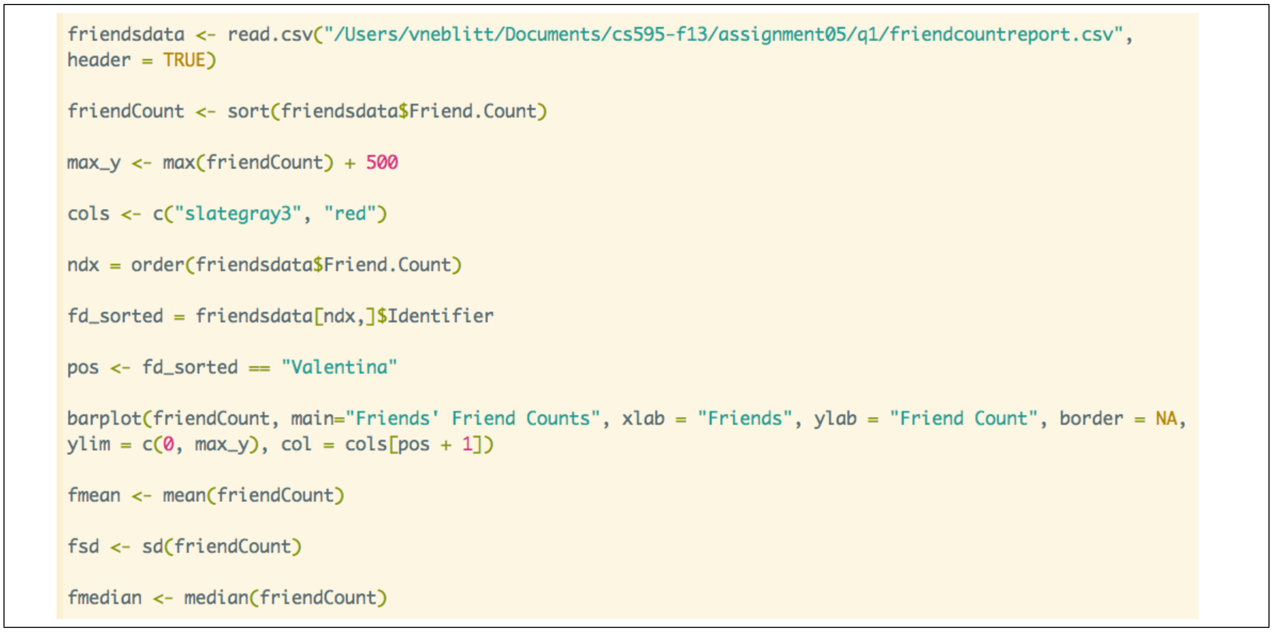
Figure 3: Hiding Friend Count



```
if friends != 0:
    s.write(str(uid) + ',' + str(friends) + '\n')
```

Figure 4: Code that Omits Friends with No Friend Count Available

I used RStudio to produce the graph for these questions. Figure 5 shows my script to produce the graph. The friends data was sorted via friend count. I needed to set the limit for the y-axis since the default graph tends to not stretch up to the highest number needed. The part about coloring the bar that represented me was tricky. The data was sorted on friend count, but I needed to access identifier which means that I needed to sort on data on a second dimension and keep the first one intact.



```
friendsdata <- read.csv("/Users/vneblitt/Documents/cs595-f13/assignment05/q1/friendcountreport.csv",
header = TRUE)

friendCount <- sort(friendsdata$Friend.Count)

max_y <- max(friendCount) + 500

cols <- c("slategray3", "red")

ndx = order(friendsdata$Friend.Count)

fd_sorted = friendsdata[ndx,]$Identifier

pos <- fd_sorted == "Valentina"

barplot(friendCount, main="Friends' Friend Counts", xlab = "Friends", ylab = "Friend Count", border = NA,
ylim = c(0, max_y), col = cols[pos + 1])

fmean <- mean(friendCount)

fsd <- sd(friendCount)

fmedian <- median(friendCount)
```

Figure 5: FriendCountGraph.R

The graph proved that my friends have more friends than I do. I am represented by the red bar in Figure 6. Figure 7 shows the part of the R script that calculated the mean, median and standard deviation below.

- mean = 343.828
- median = 258
- standard deviation = 303.303

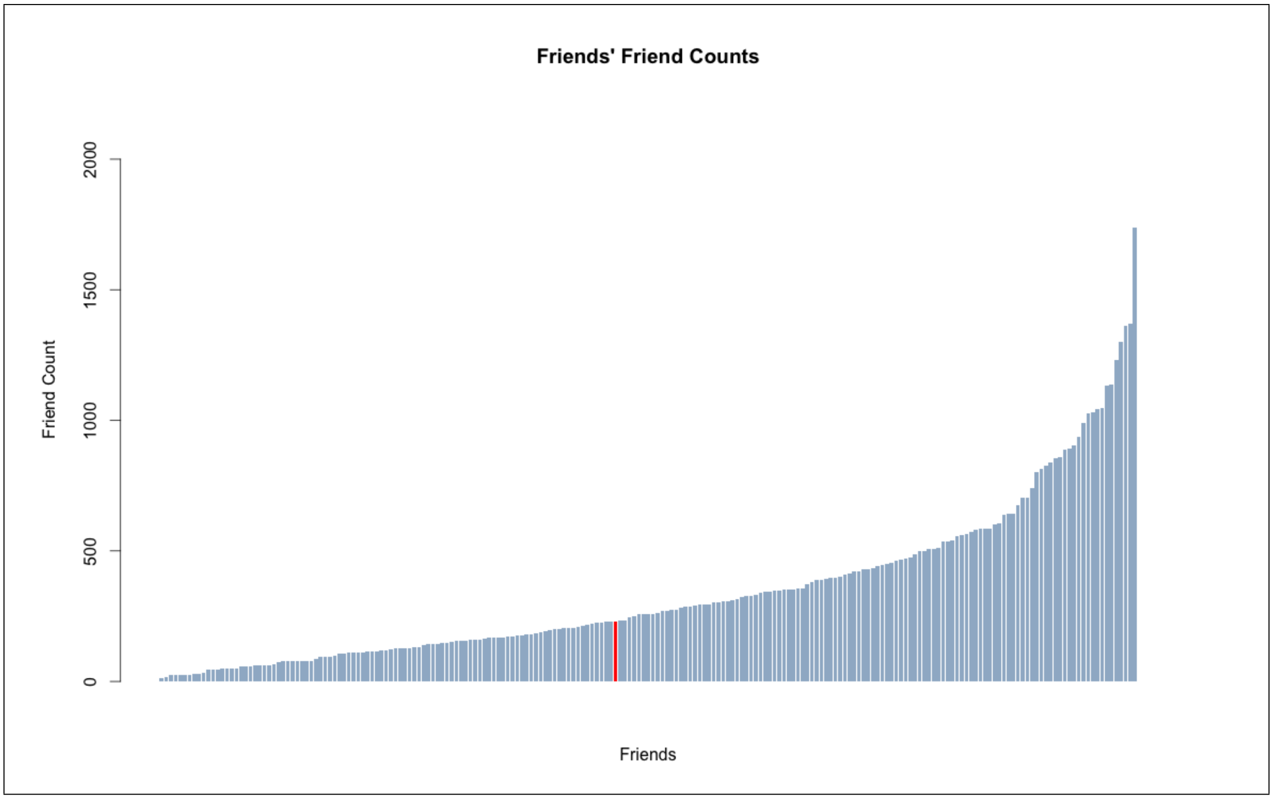
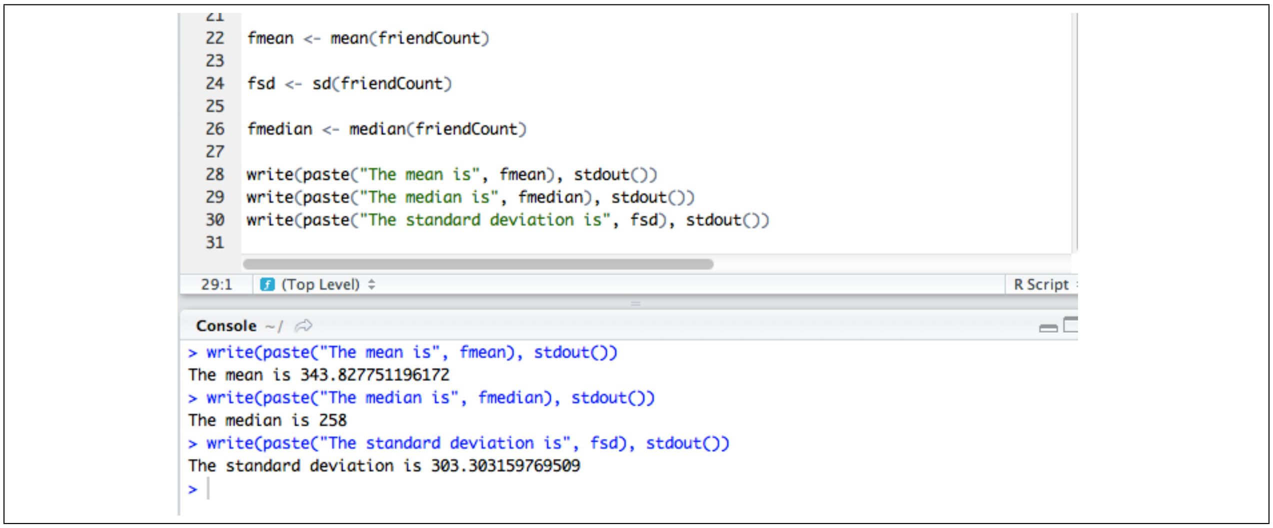


Figure 6: Facebook Friend Counts


```
fmean <- mean(friendCount)

fsd <- sd(friendCount)

fmedian <- median(friendCount)

write(paste("The mean is", fmean), stdout())
write(paste("The median is", fmedian), stdout())
write(paste("The standard deviation is", fsd), stdout())

```

```
> write(paste("The mean is", fmean), stdout())
The mean is 343.827751196172
> write(paste("The median is", fmedian), stdout())
The median is 258
> write(paste("The standard deviation is", fsd), stdout())
The standard deviation is 303.303159769509
> 
```

Figure 7: Mean, Median, and Standard Deviation

Question 2

Determine if the friendship paradox holds for your Twitter account. Since Twitter is a directed graph, use “followers” as value you measure (i.e., “do your followers have more followers than you?”)

Generate the same graph as in question #1, and calculate the same mean, standard deviation, and median values.

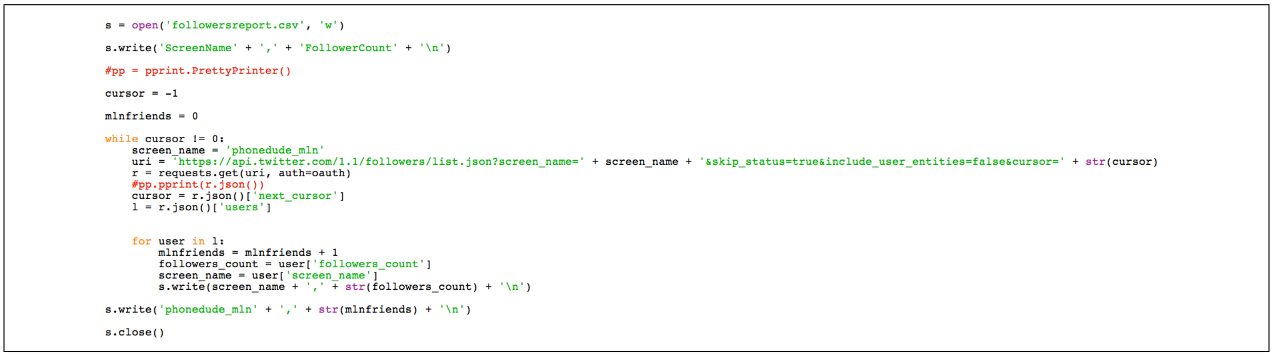
For the Twitter 1.1 API to help gather this data, see:

`https://dev.twitter.com/docs/api/1.1/get/followers/list`

If you do not have followers on Twitter (or don’t have more than 20), then use my Twitter account “phonedude_mln”.

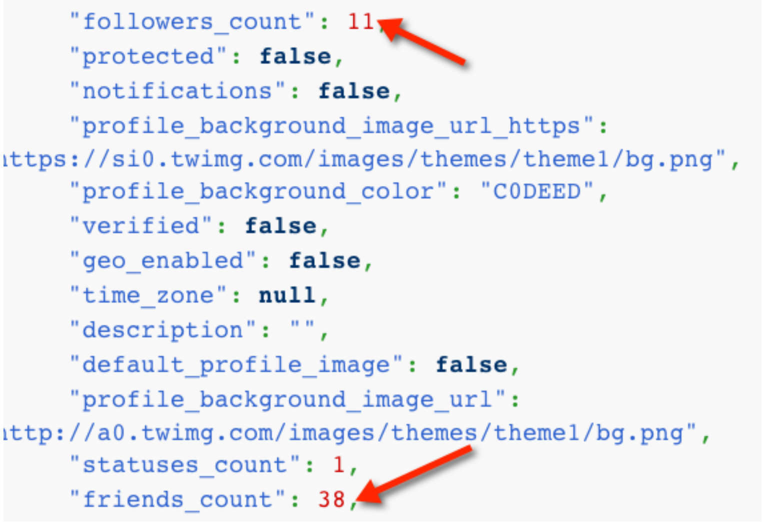
Answer to Question 2

My own Twitter account did not come close to meeting the threshold for this assignment. I am following 23 and only have 6 followers. So I used phonedude_mln. The top part of the code is identical to the work for Assignment #2 Question #1. Therefore Figure 8 shows the part that is different. The JSON format (Figure 9) showed that I needed to get the “followers_count” to gather the data for this question.



```
s = open('followersreport.csv', 'w')

s.write('ScreenName' + ',' + 'FollowerCount' + '\n')

#pp = pprint.PrettyPrinter()

cursor = -1

mlnfriends = 0

while cursor != 0:
    screen_name = 'phonedude_mln'
    uri = 'https://api.twitter.com/1.1/followers/list.json?screen_name=' + screen_name + '&skip_status=true&include_user_entities=false&cursor=' + str(cursor)
    r = requests.get(uri, auth=oauth)
    #pp.pprint(r.json())
    cursor = r.json()['next_cursor']
    l = r.json()['users']


    for user in l:
        mlnfriends = mlnfriends + 1
        followers_count = user['followers_count']
        screen_name = user['screen_name']
        s.write(screen_name + ',' + str(followers_count) + '\n')

s.write('phonedude_mln' + ',' + str(mlnfriends) + '\n')

s.close()
```

Figure 8: GetFollowers.py

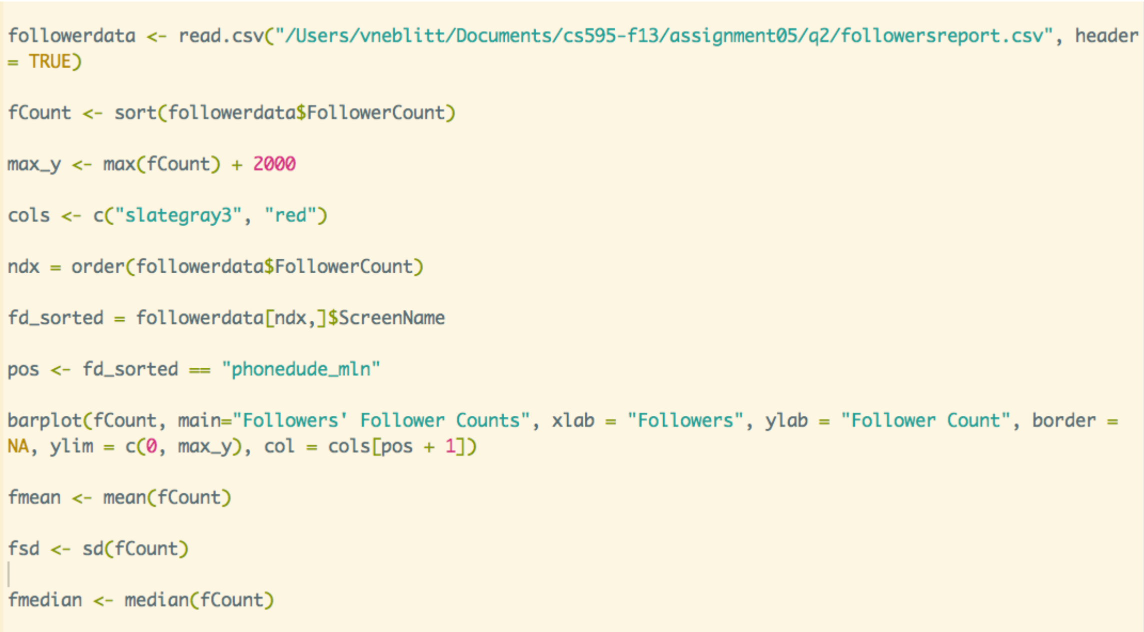


```
     "followers_count": 11,
     "protected": false,
     "notifications": false,
     "profile_background_image_url_https":
ttps://si0.twimg.com/images/themes/theme1/bg.png",
     "profile_background_color": "C0DEED",
     "verified": false,
     "geo_enabled": false,
     "time_zone": null,
     "description": "",
     "default_profile_image": false,
     "profile_background_image_url":
ttp://a0.twimg.com/images/themes/theme1/bg.png",
     "statuses_count": 1,
     "friends_count": 38,
```

Figure 9: JSON Format

I was able to mostly reuse the code from Question #1 for this graph. I had to change input file, graph labels and variable names. Again the red bar on Figure 11 represents phonedude_mln.

- mean = 515.197
- median = 196
- standard deviation = 1251.050



```
followerdata <- read.csv("/Users/vneblitt/Documents/cs595-f13/assignment05/q2/followersreport.csv", header
= TRUE)

fCount <- sort(followerdata$FollowerCount)

max_y <- max(fCount) + 2000

cols <- c("slategray3", "red")

ndx = order(followerdata$FollowerCount)

fd_sorted = followerdata[ndx,]$ScreenName

pos <- fd_sorted == "phonedude_mln"

barplot(fCount, main="Followers' Follower Counts", xlab = "Followers", ylab = "Follower Count", border =
NA, ylim = c(0, max_y), col = cols[pos + 1])

fmean <- mean(fCount)

fsd <- sd(fCount)

fmedian <- median(fCount)
```

Figure 10: FollowerCountGraph.R

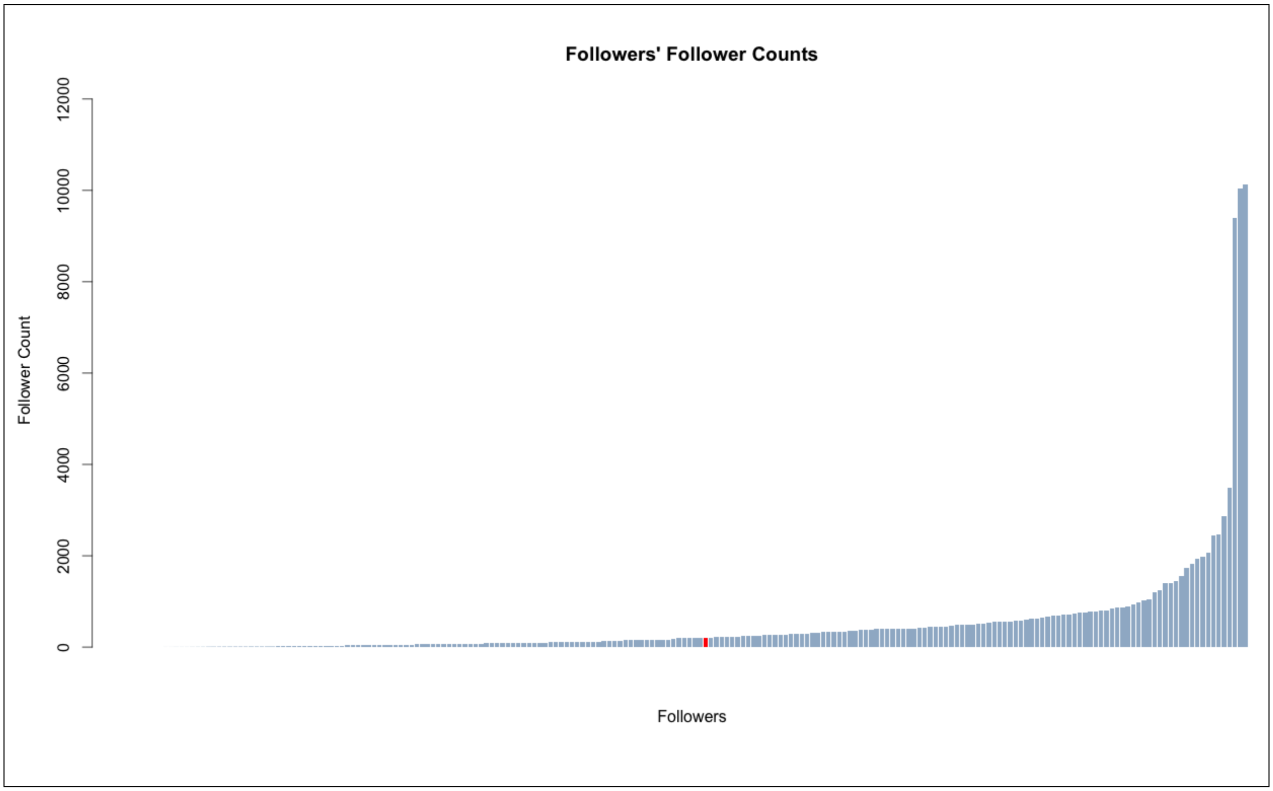
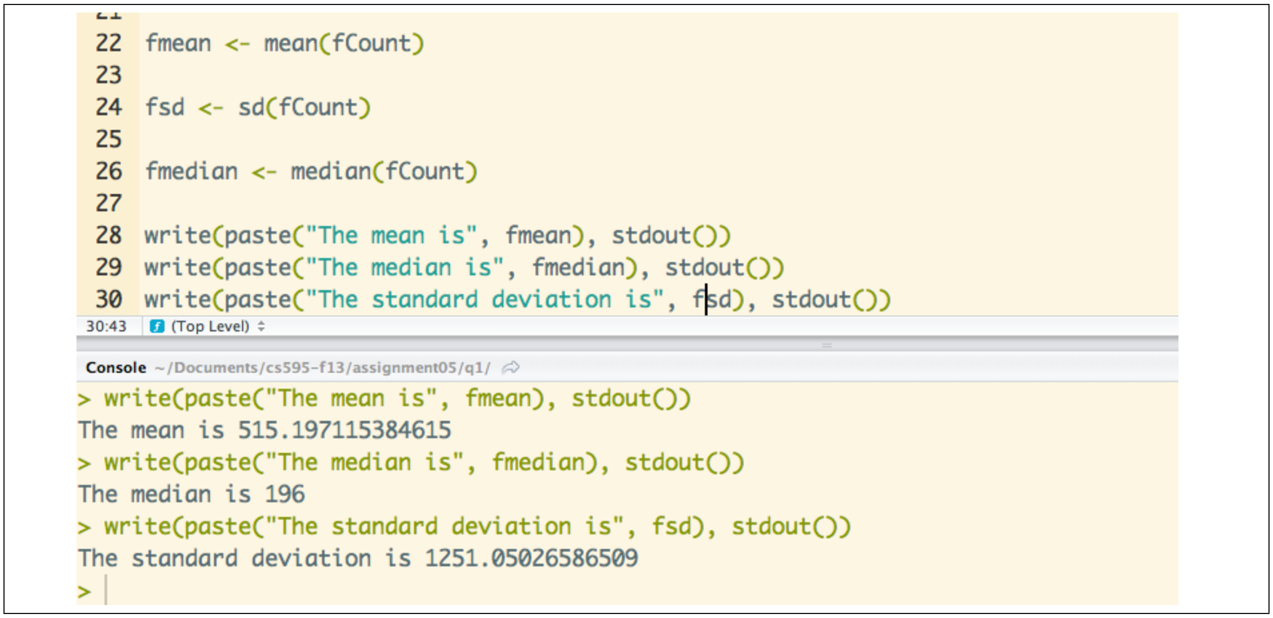


Figure 11: Twitter Followers Count



```
fmean <- mean(fCount)

fsd <- sd(fCount)

fmedian <- median(fCount)

write(paste("The mean is", fmean), stdout())
write(paste("The median is", fmedian), stdout())
write(paste("The standard deviation is", fsd), stdout())
```

```
> write(paste("The mean is", fmean), stdout())
The mean is 515.197115384615
> write(paste("The median is", fmedian), stdout())
The median is 196
> write(paste("The standard deviation is", fsd), stdout())
The standard deviation is 1251.05026586509
>
```

Figure 12: Mean, Median, and Standard Deviation

Extra Credit - LinkedIn (2 points)

Repeat question #1, but with your LinkedIn profile.

Answer to Extra Credit - LinkedIn

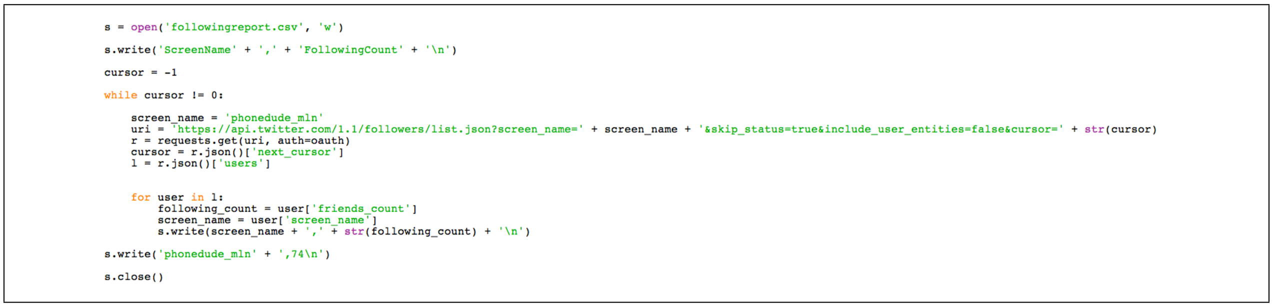
Not attempted.

Extra Credit - Twitter (1 point)

Repeat question #2, but change “followers” to “following”? In other words, are the people I am following following more people?

Answer to Extra Credit - Twitter

Again I had to use phonedude_mln and the top part of the code is identical to the work for Assignment #2 Question #1. Therefore Figure 13 shows the part that is different. This time I needed the “friends_count” data from the output (Figure 9).

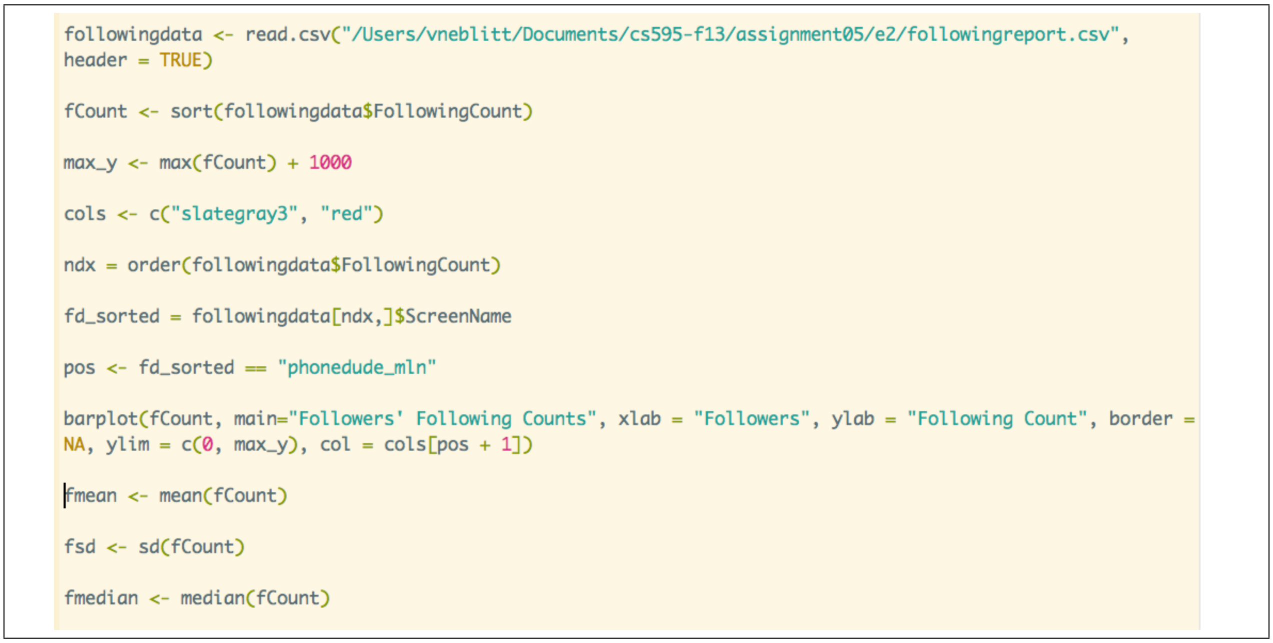


```
s = open('followingreport.csv', 'w')

s.write('ScreenName' + ',' + 'FollowingCount' + '\n')

cursor = -1

while cursor != 0:

    screen_name = 'phonedude_mln'
    uri = 'https://api.twitter.com/1.1/followers/list.json?screen_name=' + screen_name + '&skip_status=true&include_user_entities=false&cursor=' + str(cursor)
    r = requests.get(uri, auth=oauth)
    cursor = r.json()['next_cursor']
    l = r.json()['users']


    for user in l:
        following_count = user['friends_count']
        screen_name = user['screen_name']
        s.write(screen_name + ',' + str(following_count) + '\n')

s.write('phonedude_mln' + ',74\n')

s.close()
```

Figure 13: GetFollowing.py

Again, I was able to mostly reuse the code from Question #1 for this graph. I had to change input file, graph labels and variable names. The red bar on Figure 15 represents phonedude_mln.

- mean = 507
- median = 283
- standard deviation = 576.134



```
followingdata <- read.csv("/Users/vneblitt/Documents/cs595-f13/assignment05/e2/followingreport.csv",
header = TRUE)

fCount <- sort(followingdata$FollowingCount)

max_y <- max(fCount) + 1000

cols <- c("slategray3", "red")

ndx = order(followingdata$FollowingCount)

fd_sorted = followingdata[ndx,]$ScreenName

pos <- fd_sorted == "phonedude_mln"

barplot(fCount, main="Followers' Following Counts", xlab = "Followers", ylab = "Following Count", border =
NA, ylim = c(0, max_y), col = cols[pos + 1])

fmean <- mean(fCount)

fsd <- sd(fCount)

fmedian <- median(fCount)
```

Figure 14: FollowingCountGraph.R

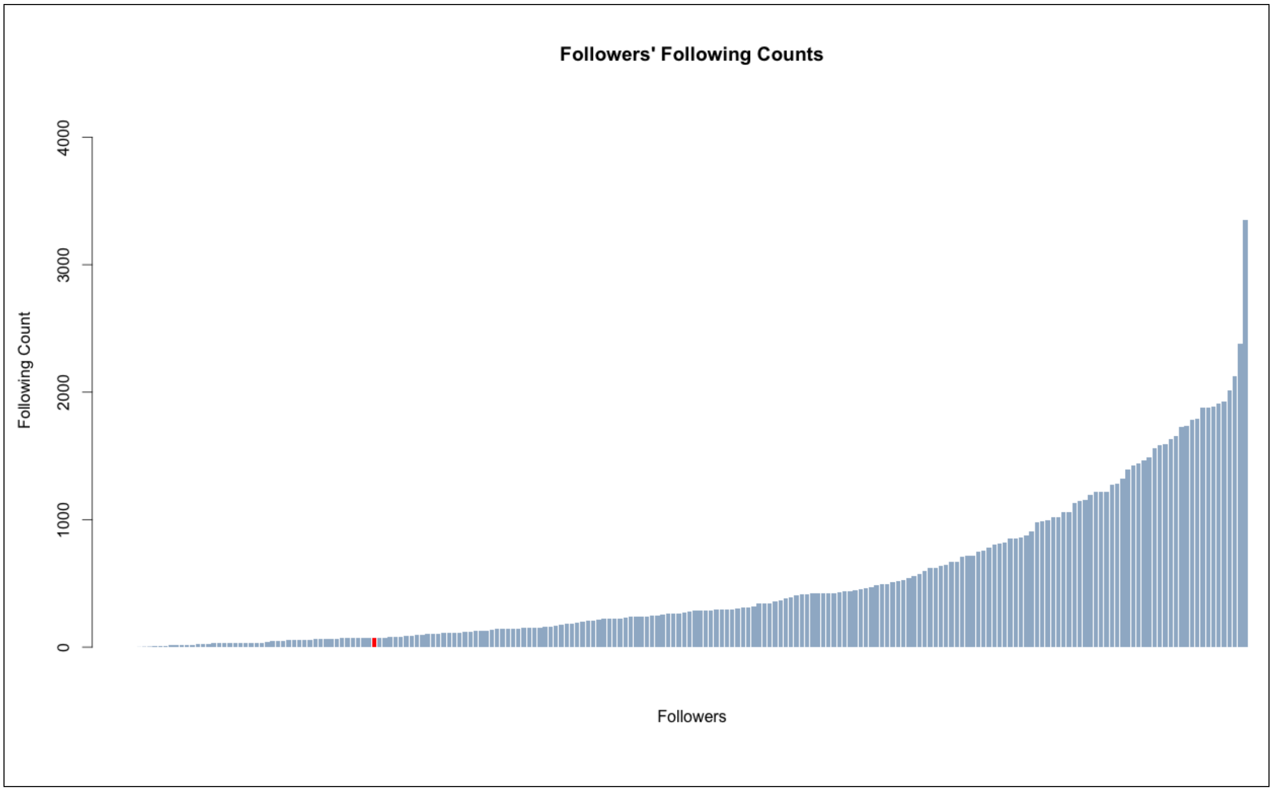
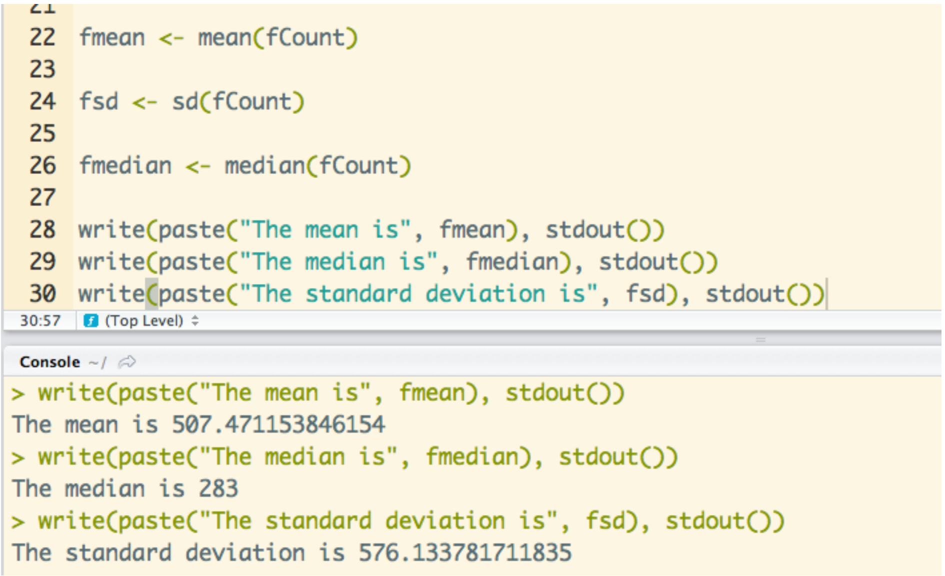


Figure 15: Twitter Following Count



```
fmean <- mean(fCount)

fsd <- sd(fCount)

fmedian <- median(fCount)

write(paste("The mean is", fmean), stdout())
write(paste("The median is", fmedian), stdout())
write(paste("The standard deviation is", fsd), stdout())
```

```
> write(paste("The mean is", fmean), stdout())
The mean is 507.471153846154
> write(paste("The median is", fmedian), stdout())
The median is 283
> write(paste("The standard deviation is", fsd), stdout())
The standard deviation is 576.133781711835
```

Figure 16: Mean, Median, and Standard Deviation

Resources

- Grosfield, Troy. Parsing XML with Python using ElementTree. `http://blog.troygrosfield.com/2010/12/18/parsing-xml-with-python-using-elementtree/`
- McCown, Frank. Producing Simple Graphs with R. `http://www.harding.edu/fmccown/r/`
- Poulson, Barton. R Statistics Essential Training. `http://www.lynda.com/course20/R-tutorials/R-Statistics-Essential-Training/142447-2.html`
- Python.org. The ElementTree XML API. `http://docs.python.org/3.3/library/xml.etree.elementtree.html`
- Seminar for Statistics. R Documentation: Arithmetic Mean. `http://stat.ethz.ch/R-manual/R-devel/library/base/html/mean.html`
- Seminar for Statistics. R Documentation: Concatenate Strings. `http://stat.ethz.ch/R-manual/R-devel/library/base/html/paste.html`
- Seminar for Statistics. R Documentation: Median Value. `http://stat.ethz.ch/R-manual/R-patched/library/stats/html/median.html`
- Seminar for Statistics. R Documentation: Standard Deviation. `http://stat.ethz.ch/R-manual/R-patched/library/stats/html/sd.html`
- Stack Overflow. Change colors or particular bars in a bar chart. `http://stackoverflow.com/questions/13112974/change-colours-of-particular-bars-in-a-bar-chart`
- Stack Overflow. How do you print to stderr in R? `http://stackoverflow.com/questions/1109017/how-do-you-print-to-stderr-in-r`
- Stack Overflow. Understanding the Order() function in R. `http://stackoverflow.com/questions/2315601/understanding-the-order-function-in-r`
- Twitter Developers Documentation. GET followers/list. `https://dev.twitter.com/docs/api/1.1/get/followers/list`
- Twitter Developers Documentation. Using cursors to navigate collections. `https://dev.twitter.com/docs/misc/cursoring`