Assignment 5

CS 595: Introduction to Web Science

Fall 2013

Shawn M. Jones

Finished on October 20, 2013

1

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.

Mean	302.555555555556
Median	225.5
Std Dev	236.389147508571

Table 1: Statistics on the count of my Facebook Friends' Friends, values straight from R

Answer

Downloading the graphl file from the NameGenWeb gave me nothing when I anonymized it, so I had to work with the non-anonymized data. The Python script `processFBGraph.py` used to process it is shown in Listing 1. It turns the data into a comma-separated stream that can be output to a file as shown below.

```
./processFBGraph.py not-anonymized-fb-data.graphml > fb-
    frienddata.csv
```

Processing the data was yet another adventure, the script shown in Listing 2, its statistics shown in Table 1, and the graph shown in Figure 1.

The R script runs like so:

```
bash $ --> ./processFBGraphOutput.R fb-frienddata.csv q1-barplot
    .png 154 'ME!!!'
Mean:  302.555555555556
Median:  225.5
Std Dev:  236.389147508571
null device
          1
```

Seeing as I am the only person with 154 friends on my circle, I was able to color the single bar red using the code on lines 34 and 35 in Listing 2. Also of note, some friends do not share their friend counts, so they are left out of the data collected.

Figure 1 shows that I am more popular than about 25% of my friends, but my friends do have more friends than I do. Referencing Table 1, I have fewer friends than the median.

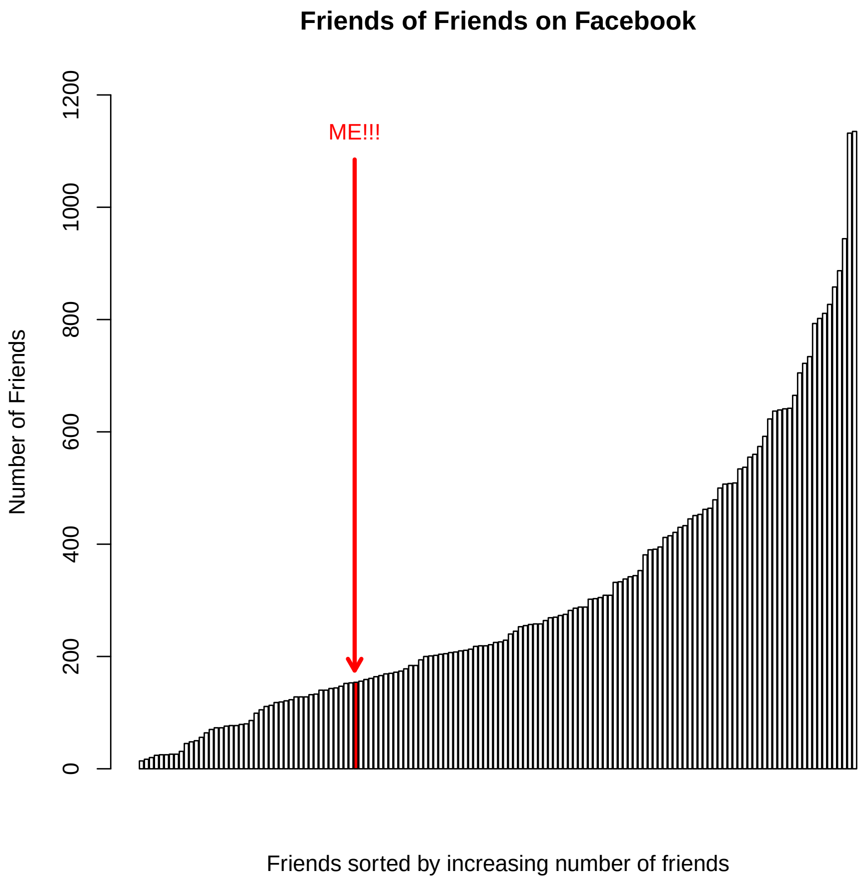

Figure 1: Bar plot showing the count of my Facebook Friends' Friends

```
#!/usr/local/bin/python3

import sys
from xml.dom.minidom import parseString

def getFriendInfo(xml):

    dom = parseString(xml)
    countDict = {}

    for element in dom.getElementsByTagName("data"):
        if (element.attributes['key'].value == 'name'):
            name = element.childNodes[0].data

        if (element.attributes['key'].value == 'friend_count'):
            count = element.childNodes[0].data

            countDict[name] = count
            name = ''
            count = ''

    return countDict

def getFriendCount(xml):

    dom = parseString(xml)
    return len(dom.getElementsByTagName("node"))


if __name__ == "__main__":

    graphmlFile = sys.argv[1]

    f = open(graphmlFile)
    xml = f.read()
    f.close()

    myFriendCount = getFriendCount(xml)
    friendInfo = getFriendInfo(xml)

    print("Name,Friend Count")
    print('ME,' + str(myFriendCount))

    for friend in friendInfo:
        print(friend + ',' + friendInfo[friend])
```

Listing 1: Python program for processing GraphML file from NameGenWeb Facebook App

```r
#!/usr/bin/Rscript

args <- commandArgs(trailingOnly = TRUE)

inputfile <- args[1]
outputfile <- args[2]
mylocation <- args[3]
mytext <- args[4]

data <- read.csv(inputfile)

incdata <- sort(data$Friend.Count)

meanOut <- paste("Mean: ", mean(incdata), collapse = "")

medianOut <- paste("Median: ", median(incdata), collapse = "")

sdOut <- paste("Std Dev: ", sd(incdata), collapse = "")

write(meanOut, stdout())
write(medianOut, stdout())
write(sdOut, stdout())

pdf(outputfile)

# these are used to acquire names for labels later
#ndx = order(data$Friend.Count)
#xlabels <- data[ndx,]$Name

# for the coloring of specific bars in the barplot:
# http://stackoverflow.com/questions/13112974/change-colours-of-
    particular-bars-in-a-bar-chart
# create a vector containing the items equal to my number of
    friends
#mylocation = mylocation + 1
pos <- (incdata == mylocation)
cols <- c("white", "red") # colors to use (first is everyone but
     me)

# draw the barplot
barplot(incdata, main="Friends of Friends on Facebook", xlab="
    Friends sorted by increasing number of friends", ylab="Number
     of Friends", col=cols[pos + 1], ylim=c(0, max(incdata) +
    100))
#barplot(incdata, main="Friends of Friends on Facebook", xlab="
    Friends sorted by increasing number of friends", ylab="Number
     of Friends", col=cols[pos + 1], ylim=c(0, max(incdata) +
    100), names.arg=xlabels, las=3, cex.names=0.4)
```

```
# annotation and arrow
# http://blog.earlh.com/index.php/2009/07/labeling-plots-annotations-legends-etc-part-6-in-a-series/
text(x=match(c(mylocation), incdata) + 8, y=max(incdata), labels=mytext, col='red')
arrows(x0=match(c(mylocation), incdata) + 8, y0=(max(incdata) - 50), x1=match(c(mylocation), incdata) + 8, y1=175, length=0.1, lwd=3, col='red')

dev.off()
```

Listing 2: R program for bar plot shown in Figure 1

2

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 calcuate 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".

Mean	520.846534653465
Median	199
Std Dev	1264.79341369106

Table 2: Statistics on the count of phonedude_mln's Twitter followers' followers, values straight from R

Answer

Because I use Twitter as more of a *content consumption* service, I have very few followers, so few that I lack sufficient sample size to actually answer "do your followers have more followers than you?". Fortunately, I have `phonedude_mln` that I can test with.

The first script, shown in Listing 3, queries the Twitter API for information on the followers of `phonedude_mln` using the function on lines 84-98 and then prints it using the function on lines 100 - 124.

This script is run like below, to produce a CSV file.

```
./countTwitterFollowers.py 2 phonedude_mln > phonedude_mln.csv
```

From the output, I can see that `phonedude_mln` has 201 followers. The R script shown in Listing 4 creates a similar bar plot to that shown in answer one and produces the statistics shown in 2.

```
bash $ --> ./processTwitterFollowerOutput.R phonedude_mln.csv q2
    -barplot.pdf 201 "phonedude_mln"
Mean:   520.846534653465
Median:   199
Std Dev:   1264.79341369106
null device
          1
```

It turns out that `phonedude_mln` is doing better on Twitter than I am doing on Facebook. He has more followers than the median of 199, but his followers still have more followers than he does.

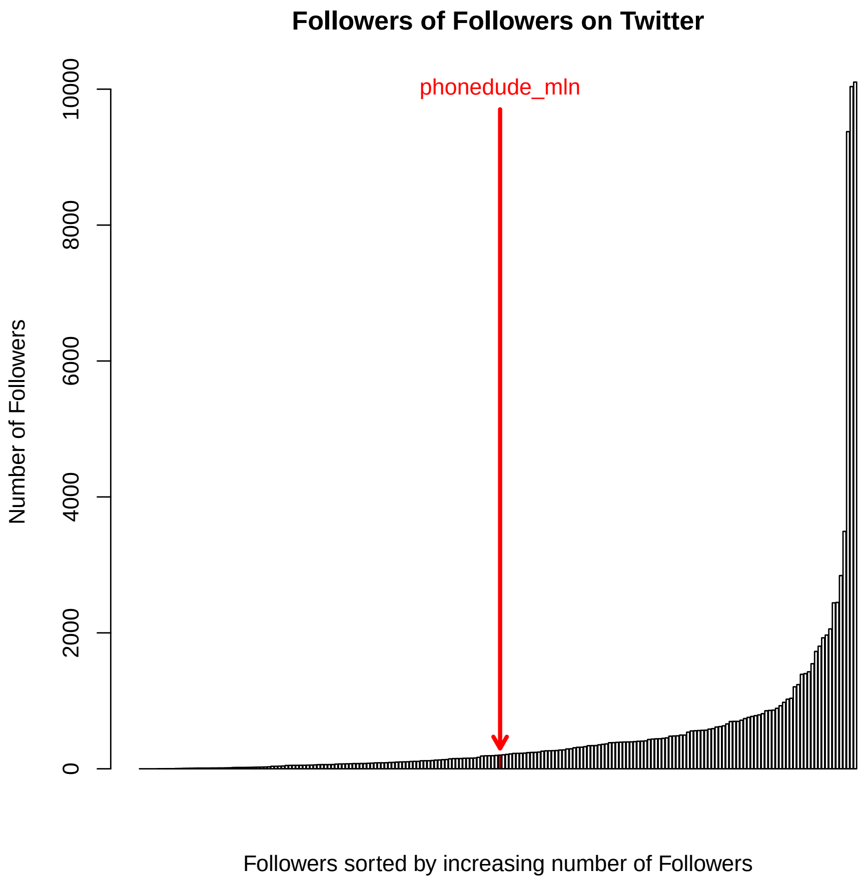

Figure 2: Bar plot showing the count of phonedude_mln's Twitter followers' followers

```
#!/usr/local/bin/python3

# -*- encoding: utf-8 -*-
from __future__ import unicode_literals
import requests
from requests_oauthlib import OAuth1
from urllib.parse import parse_qs
import json
import time
import sys

# ugly, but necessary, globals; saw no need to change this
# strategy from the example
REQUEST_TOKEN_URL = "https://api.twitter.com/oauth/request_token
    "
AUTHORIZE_URL = "https://api.twitter.com/oauth/authorize?
    oauth_token="
ACCESS_TOKEN_URL = "https://api.twitter.com/oauth/access_token"

CONSUMER_KEY = "n7jt1uMTwGCcIzDvey8g0A"
CONSUMER_SECRET = "0r6HUrVD36W4MULgWETKMxrQsCICNy1OFFNc2iW4o"

OAUTH_TOKEN = "528649269-
    SffJ0Rei5PzLYd2NSJPnnm28dP5nlAnt7E1gRGwo"
OAUTH_TOKEN_SECRET = "htrwXF09pS8tP8cMzFrxmMryavdPXd0zPiJHRnLs"

class APIError(Exception):
    """
        If something goes wrong with the API, throw one of these
            .
        (avoids sys.exit in the middle of the program)
    """

    def __init__(self, value):
        self.value = value

    def __str__(self):
        return repr(self.value)

def setup_oauth():
    """
        Authorize your app via identifier.
        Code inspired by:
        http://thomassileo.com/blog/2013/01/25/using-twitter-
            rest-api-v1-dot-1-with-python/
    """

    # Request token
    oauth = OAuth1(CONSUMER_KEY, client_secret=CONSUMER_SECRET)
    r = requests.post(url=REQUEST_TOKEN_URL, auth=oauth)

    credentials = parse_qs(r.content)

    resource_owner_key = credentials[b'oauth_token'][0].decode(
        encoding='UTF-8')
    resource_owner_secret = credentials[b'oauth_token_secret'
        ][0].decode(encoding='UTF-8')

    # Authorize
    authorize_url = AUTHORIZE_URL + resource_owner_key
    print('Please go here and authorize: ' + authorize_url)

    verifier = input('Please input the verifier: ')
    oauth = OAuth1(CONSUMER_KEY,
                   client_secret=CONSUMER_SECRET,
                   resource_owner_key=resource_owner_key,
                   resource_owner_secret=resource_owner_secret,
                   verifier=verifier)

    # Finally, Obtain the Access Token
    r = requests.post(url=ACCESS_TOKEN_URL, auth=oauth)
    credentials = parse_qs(r.content)
    token = credentials[b'oauth_token'][0].decode(encoding='UTF
        -8')
    secret = credentials[b'oauth_token_secret'][0].decode(
        encoding='UTF-8')

    return token, secret


def get_oauth():
    """
        Code inspired by:
        http://thomassileo.com/blog/2013/01/25/using-twitter-
            rest-api-v1-dot-1-with-python/
    """
    oauth = OAuth1(CONSUMER_KEY,
                client_secret=CONSUMER_SECRET,
                resource_owner_key=OAUTH_TOKEN,
                resource_owner_secret=OAUTH_TOKEN_SECRET)
    return oauth


def call_followers_list_api(oauth, count, screenName, cursor):
    url = \
        "https://api.twitter.com/1.1/followers/list.json?
            screen_name=" + \
        screenName + "&count=" + str(count) + "&cursor=" +
            cursor

    response = requests.get( url, auth=oauth )

    if 'errors' in response:
        raise APIError(
            json.dumps(
                response.json(), sort_keys=True,
                indent=4, separators=(',', ': '))
                )

    return response

def print_friend_counts(oauth, numberOfCalls, count, screenName)
    :

    cursor="-1"

    print("Name,Friend Count,Followees")

    followers_count = 0

    for i in range(0, numberOfCalls):

        response = call_followers_list_api(oauth, count,
            screenName, cursor)

        # as per:
        # https://dev.twitter.com/discussions/1053
        # friends_count - number of users the user follows
        # followers_count - number of users that follow the user
        for entry in response.json()['users']:
            ident = str(entry['screen_name'])
            followers = str(entry['followers_count'])
            print(ident + ',' + followers)
            followers_count += 1

        cursor = str(response.json()['next_cursor'])

    print(screenName + ',' + str(followers_count))

def usage():

    print("Usage: " + sys.argv[0] + " <apiCalls> <screenName>")


if __name__ == "__main__":

    #startingid = "400000000000000000"
    try:
        apiCalls = int(sys.argv[1])
        screenName = sys.argv[2]
    except IndexError as e:
        usage()
        sys.exit(1)

    if not OAUTH_TOKEN:
        token, secret = setup_oauth()
        print( "OAUTH_TOKEN: " + token )
        print( "OAUTH_TOKEN_SECRET: " + secret )
        print( )
    else:
        oauth = get_oauth()
        count = 200

        try:
            print_friend_counts(oauth, apiCalls, count,
                screenName)
        except APIError as e:
            sys.stderr.write(e.value)
            sys.exit(254)
```

Listing 3: Python program for acquiring Twitter followers for phonedude_mln

```
#!/usr/bin/Rscript

args <- commandArgs(trailingOnly = TRUE)

inputfile <- args[1]
outputfile <- args[2]
mylocation <- as.integer(args[3])
mytext <- args[4]

data <- read.csv(inputfile)

incdata <- sort(data$Friend.Count)

meanOut <- paste("Mean: ", mean(incdata), collapse = "")

medianOut <- paste("Median: ", median(incdata), collapse = "")

sdOut <- paste("Std Dev: ", sd(incdata), collapse = "")

write(meanOut, stdout())
write(medianOut, stdout())
write(sdOut, stdout())

pdf(outputfile)

#ndx = order(data$Friend.Count)
#xlabels <- data[ndx,]$Name

# for the coloring of specific bars in the barplot:
# http://stackoverflow.com/questions/13112974/change-colours-of-
    particular-bars-in-a-bar-chart
# create a vector containing the items equal to my number of
    friends
#mylocation = mylocation + 1
pos <- (incdata == mylocation)
cols <- c("white", "red") # colors to use (first is everyone but
    me)

# draw the barplot
barplot(incdata, main="Followers of Followers on Twitter", xlab=
    "Followers sorted by increasing number of Followers", ylab="
    Number of Followers", col=cols[pos + 1], ylim=c(0, max(
    incdata) + 100))
#barplot(incdata, main="Followers of Followers on Twitter", xlab
    ="Friends sorted by increasing number of friends", ylab="
    Number of Friends", col=cols[pos + 1], ylim=c(0, max(incdata)
    + 100), names.arg=xlabels, las=3, cex.names=0.4)

```

```
# annotation and arrow
# http://blog.earlh.com/index.php/2009/07/labeling-plots-
    annotations-legends-etc-part-6-in-a-series/
text(x=match(c(mylocation), incdata) + 20, y=max(incdata) - 100,
     labels=mytext, col='red')
arrows(x0=match(c(mylocation), incdata) + 20, y0=(max(incdata))
   - 400, x1=match(c(mylocation), incdata) + 20, y1=300, length
   =0.1, lwd=3, col='red')

dev.off()
```

Listing 4: R program for bar plot shown in Figure 2

3

Question

```
Extra credit, 2 points:

3.  Repeat question #1, but with your LinkedIn profile.
```

Answer

Not attempted.

4

Question

```
Extra credit, 1 point:

4.  Repeat question #2, but change "followers" to "following"?  In
other words, are the people I am following following more people?
```

Mean	508.120192307692
Median	283
Std Dev	575.712396307217

Table 3: Statistics on the count of phonedude_mln's Twitter followers' followees, values straight from R

Answer

The result appears to hold in the case of "following", as shown in Figure 3. The Python code for this case is in Listing 5 and the R code for this Figure is in Listing 6. The statistics are shown in Table 3.

Both scripts are executed as shown in the answer to Question 2.

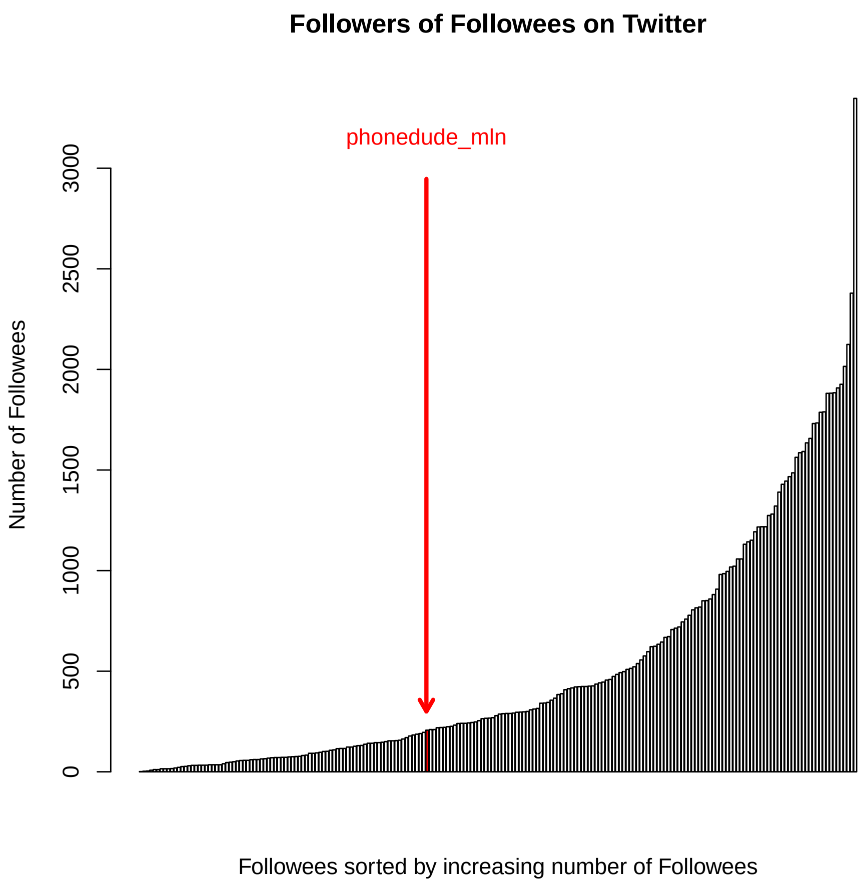

Figure 3: Bar plot showing the count of phonedude_mln's Twitter followers' friends

```
#!/usr/local/bin/python3

# -*- encoding: utf-8 -*-
from __future__ import unicode_literals
import requests
from requests_oauthlib import OAuth1
from urllib.parse import parse_qs
import json
import time
import sys

# ugly, but necessary, globals; saw no need to change this
# strategy from the example
REQUEST_TOKEN_URL = "https://api.twitter.com/oauth/request_token
    "
AUTHORIZE_URL = "https://api.twitter.com/oauth/authorize?
    oauth_token="
ACCESS_TOKEN_URL = "https://api.twitter.com/oauth/access_token"

CONSUMER_KEY = "n7jt1uMTwGCcIzDvey8g0A"
CONSUMER_SECRET = "0r6HUrVD36W4MULgWETKMxrQsCICNy1OFFNc2iW4o"

OAUTH_TOKEN = "528649269-
    SffJ0Rei5PzLYd2NSJPnnm28dP5nlAnt7E1gRGwo"
OAUTH_TOKEN_SECRET = "htrwXF09pS8tP8cMzFrxmMryavdPXd0zPiJHRnLs"

class APIError(Exception):
    """
        If something goes wrong with the API, throw one of these
            .
        (avoids sys.exit in the middle of the program)
    """

    def __init__(self, value):
        self.value = value

    def __str__(self):
        return repr(self.value)

def setup_oauth():
    """
        Authorize your app via identifier.
        Code inspired by:
        http://thomassileo.com/blog/2013/01/25/using-twitter-
            rest-api-v1-dot-1-with-python/
    """

    # Request token
    oauth = OAuth1(CONSUMER_KEY, client_secret=CONSUMER_SECRET)
    r = requests.post(url=REQUEST_TOKEN_URL, auth=oauth)

    credentials = parse_qs(r.content)

    resource_owner_key = credentials[b'oauth_token'][0].decode(
        encoding='UTF-8')
    resource_owner_secret = credentials[b'oauth_token_secret'
        ][0].decode(encoding='UTF-8')

    # Authorize
    authorize_url = AUTHORIZE_URL + resource_owner_key
    print('Please go here and authorize: ' + authorize_url)

    verifier = input('Please input the verifier: ')
    oauth = OAuth1(CONSUMER_KEY,
                   client_secret=CONSUMER_SECRET,
                   resource_owner_key=resource_owner_key,
                   resource_owner_secret=resource_owner_secret,
                   verifier=verifier)

    # Finally, Obtain the Access Token
    r = requests.post(url=ACCESS_TOKEN_URL, auth=oauth)
    credentials = parse_qs(r.content)
    token = credentials[b'oauth_token'][0].decode(encoding='UTF
        -8')
    secret = credentials[b'oauth_token_secret'][0].decode(
        encoding='UTF-8')

    return token, secret


def get_oauth():
    """
        Code inspired by:
        http://thomassileo.com/blog/2013/01/25/using-twitter-
            rest-api-v1-dot-1-with-python/
    """
    oauth = OAuth1(CONSUMER_KEY,
                client_secret=CONSUMER_SECRET,
                resource_owner_key=OAUTH_TOKEN,
                resource_owner_secret=OAUTH_TOKEN_SECRET)
    return oauth


def call_followers_list_api(oauth, count, screenName, cursor):
    url = \
        "https://api.twitter.com/1.1/followers/list.json?
            screen_name=" + \
        screenName + "&count=" + str(count) + "&cursor=" +
            cursor

    response = requests.get( url, auth=oauth )

    if 'errors' in response:
        raise APIError(
            json.dumps(
                response.json(), sort_keys=True,
                indent=4, separators=(',', ': '))
                )

    return response

def print_friend_counts(oauth, numberOfCalls, count, screenName)
    :

    cursor="-1"

    print("Name,Friend Count")

    followers_count = 0

    for i in range(0, numberOfCalls):

        response = call_followers_list_api(oauth, count,
            screenName, cursor)

        # as per:
        # https://dev.twitter.com/discussions/1053
        # friends_count - number of users the user follows
        # followers_count - number of users that follow the user
        for entry in response.json()['users']:
            ident = str(entry['screen_name'])
            followers = str(entry['friends_count'])
            print(ident + ',' + followers)
            followers_count += 1

        cursor = str(response.json()['next_cursor'])

    print(screenName + ',' + str(followers_count))

def usage():

    print("Usage: " + sys.argv[0] + " <apiCalls> <screenName>")


if __name__ == "__main__":

    #startingid = "400000000000000000"
    try:
        apiCalls = int(sys.argv[1])
        screenName = sys.argv[2]
    except IndexError as e:
        usage()
        sys.exit(1)

    if not OAUTH_TOKEN:
        token, secret = setup_oauth()
        print( "OAUTH_TOKEN: " + token )
        print( "OAUTH_TOKEN_SECRET: " + secret )
        print( )
    else:
        oauth = get_oauth()
        count = 200

        try:
            print_friend_counts(oauth, apiCalls, count,
                screenName)
        except APIError as e:
            sys.stderr.write(e.value)
            sys.exit(254)
```

Listing 5: Python program for acquiring Twitter phonedude_mln's Twitter following followers_mln

```
#!/usr/bin/Rscript

args <- commandArgs(trailingOnly = TRUE)

inputfile <- args[1]
outputfile <- args[2]
mylocation <- as.integer(args[3])
mytext <- args[4]

data <- read.csv(inputfile)

incdata <- sort(data$Friend.Count)

meanOut <- paste("Mean: ", mean(incdata), collapse = "")

medianOut <- paste("Median: ", median(incdata), collapse = "")

sdOut <- paste("Std Dev: ", sd(incdata), collapse = "")

write(meanOut, stdout())
write(medianOut, stdout())
write(sdOut, stdout())

pdf(outputfile)

#ndx = order(data$Friend.Count)
#xlabels <- data[ndx,]$Name

# for the coloring of specific bars in the barplot:
# http://stackoverflow.com/questions/13112974/change-colours-of-
    particular-bars-in-a-bar-chart
# create a vector containing the items equal to my number of
    friends
#mylocation = mylocation + 1
pos <- (incdata == mylocation)
cols <- c("white", "red") # colors to use (first is everyone but
     me)

# draw the barplot
barplot(incdata, main="Followers of Followers on Twitter", xlab=
    "Followers sorted by increasing number of Followers", ylab="
    Number of Followers", col=cols[pos + 1], ylim=c(0, max(
    incdata) + 100))
#barplot(incdata, main="Followers of Followers on Twitter", xlab
    ="Friends sorted by increasing number of friends", ylab="
    Number of Friends", col=cols[pos + 1], ylim=c(0, max(incdata)
     + 100), names.arg=xlabels, las=3, cex.names=0.4)

```

```
# annotation and arrow
# http://blog.earlh.com/index.php/2009/07/labeling-plots-
    annotations-legends-etc-part-6-in-a-series/
text(x=match(c(mylocation), incdata) + 20, y=max(incdata) - 100,
     labels=mytext, col='red')
arrows(x0=match(c(mylocation), incdata) + 20, y0=(max(incdata))
   - 400, x1=match(c(mylocation), incdata) + 20, y1=300, length
   =0.1, lwd=3, col='red')

dev.off()
```

Listing 6: R program for bar plot shown in Figure 3