CS595 Assignment 5

Jon Robison

October 16, 2013

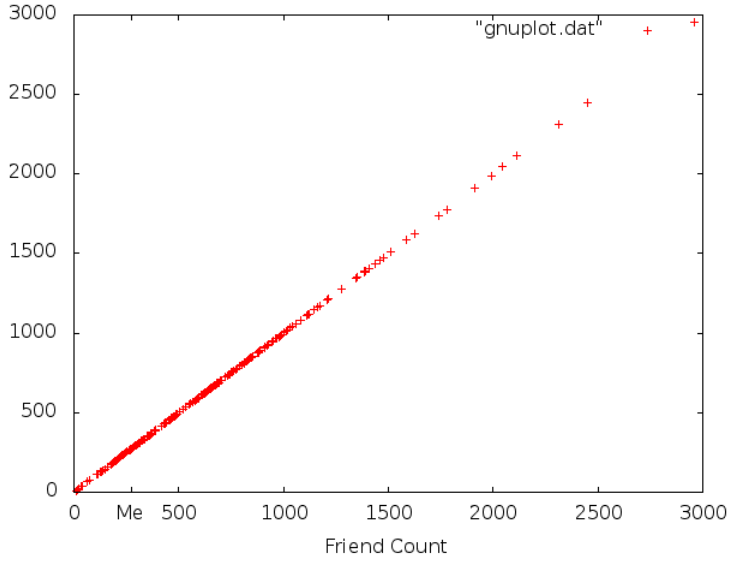
Q1. Determine if the friendship paradox holds for your Facebook account. Create a graph of the number of friends (y-axis) and the friends sorted by number of friends (x-axis). (The friends don't need to be labeled on the x-axis.) Do include yourself in the graph and label yourself accordingly.

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

From a sample size of 272, 9 do not share friend count data publicly and 20 have other privacy restrictions precluding calculation.

See Appendix A for translator
Average: 639.065843621
Std deviation: 465.913488086
Median: 535.0

Figure 1: Facebook Friend Count

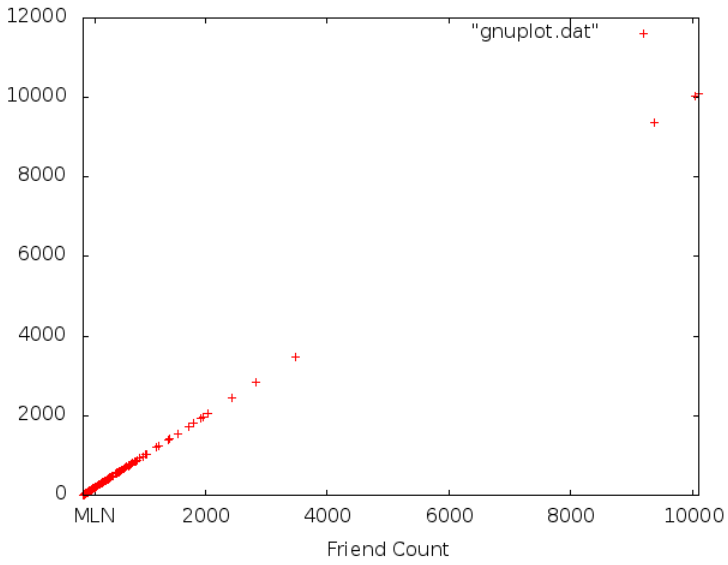


Q2. Determine if the friendship paradox holds for your Twitter account. Since Twitter is a directed graph, use “followers” as value you measure (i.e., “do your followers have more followers than you?”).

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

See Appendix B for program
Average: 521.293532338
Std deviation: 1264.46308524
Median: 197.0

Figure 2: Twitter Friend Count



Q3 EC. Repeat question 1, but with your LinkedIn profile

Q4 EC. Repeat question 2, but change “followers” to “following”? In other words, are the people I am following following more people?

Appendix A

```
#!/usr/bin/python3
#Read graphml file representing output from NameGenWeb on facebook
#and produce gnuplot scatter plot for friend count
import xml.etree.ElementTree as ET
import numpy
from numpy import array
import sys

DEFAULT_FILE='output.graphml'
if len(sys.argv) != 2:
        print('Please pass the path to your graphml file representing ' +
                'output from NameGenWeb, defaulting to ' + DEFAULT_FILE)
        path=DEFAULT_FILE
else:
        path=sys.argv[1]

root=ET.parse(path).getroot()
d={}
missing=0
for node in root.findall('node'):
    friend_count=-1
    name=''
    for data in node.iter('data'):
        if data.attrib['key']=='friend_count':
            friend_count=int(data.text)
        if data.attrib['key']=='Label':
            name=data.text
    if friend_count == -1:
        print(name + ' does not share friend count publicly!')
        missing+=1
    else:
        d[name]=friend_count

print('Missing: ' + str(missing))
with open('output.gnuplot', mode='w') as f:
    for entry in sorted(d.items(), key=lambda x: x[1]):
        value=str(entry[1])
        f.write(value + ' ' + value + '\n')
        print(entry)

friendCounts=array(list(d.values()))
print('Average: ' + str(numpy.mean(friendCounts)))
print('Std deviation: ' + str(numpy.std(friendCounts)))
print('Median: ' + str(numpy.median(friendCounts)))
```

Appendix B

```
#!/bin/python3
from __future__ import unicode_literals
import numpy
from numpy import array
import requests
from requests_oauthlib import OAuth1
import urllib
from urllib.parse import urlparse
import sys
import time

CONSUMER_KEY = "dbr6Ce0ahKsr4QMyIxElQ"
CONSUMER_SECRET = "Bxd17G6TSr74lDzpGkl9ThQRqE5HwDtrPofdJninKLA"
OAUTH_TOKEN = "589075411-qVWno21brc3Kjw24Wg6UeDJVHBE6DsrRbd7r0gGU"
OAUTH_TOKEN_SECRET = "fLEC8DJvPqginylDhhCm0F9xNS2R6Xc8djk6DbVlVI"
LIST_URI = 'https://api.twitter.com/1.1/followers/list.json'
TARGET='phonedude_mln'
ERROR_TIME=10
RATE_TIME=15*60
RATE_EXCEEDED_MESSAGE='Rate limit exceeded'

def parse_arguments():
    if len(sys.argv) != 2:
        print('Please pass the screen_name of the user from which ' +
            ' you wish to measure, defaulting to ' + TARGET)
        return TARGET
    else:
        return sys.argv[1]

def get_oauth():
    oauth = OAuth1(CONSUMER_KEY,
                client_secret=CONSUMER_SECRET,
                resource_owner_key=OAUTH_TOKEN,
                resource_owner_secret=OAUTH_TOKEN_SECRET)
    return oauth

def get_followers(oauth, screen_name):
    s=[]
    cursor = -1
    while cursor != 0:
        while True:
            r = requests.get(url=LIST_URI + '?screen_name=' + screen_name +
                '&count=200&cursor=' + str(cursor), auth=oauth)
            json=r.json()
```

```
            if 'errors' in json:
                if RATE_EXCEEDED_MESSAGE in json['errors'][0]['message']:
                    print('Sleeping from rate error')
                    time.sleep(RATE_TIME)
                else:
                    print('Sleeping from non-rate error')
                    time.sleep(10)
            else:
                break;
        if 'next_cursor' in json:
            cursor = json['next_cursor']
        else:
            cursor = 0
        if 'users' in json:
            for user in json['users']:
                s.append(user['screen_name'])
    return s

def twitter_search(oauth, target):
    followers=get_followers(oauth, target)
    requests=1
    print('Found ' + str(len(followers)) + ' followers, names: ' +
        str(followers))
    counts=[]
    for follower in followers:
        count=len(get_followers(oauth, follower))
        counts.append(count)
        print('Found ' + str(count) + ' followers for user ' + follower)
    write_counts(counts)
    return followers

def write_counts(counts):
    with open('gnuplot.dat', 'w') as f:
        for count in sorted(counts):
            f.write(str(count) + ' ' + str(count) + '\n')

if __name__ == "__main__":
    oauth = get_oauth()
    target=parse_arguments()
    l=twitter_search(oauth, target)
    friendCounts=array(l)
    print('Average: ' + str(numpy.mean(friendCounts)))
    print('Std deviation: ' + str(numpy.std(friendCounts)))
    print('Median: ' + str(numpy.median(friendCounts)))
```