Web Science: Assignment #4

Alexander Nwala

Puneeth Bikkasandra

Friday, March 2, 2018

Contents

Problem 1

Determine if the friendship paradox holds for my Facebook account.* Compute the mean, median and standard deviation of the number of friends that my friends have. Create a graph of the number of friends (y-axis) and the friends themselves, sorted by number of friends (y-axis).

SOLUTION :

1. To Compute Mean

```
Mean = Sum of all the Friend's friend Count / Total Friend Count
```

2. To compute Median

```
Median =  Middle Item of the Sorted Friends count
```

3. To compute Standard Deviation

```
Standard Deviation = SquareRoot {Sum of all([Square(friend count - Mean)])
/Total Friend Count}
```

Listing 1: faceBookStats.py

```

import csv
import math
from astropy.table import Table, Column
import numpy as np

friendsDictionary ={}
totalCount = 0
friendCountList = []
finalFriendCount = 0
finalMean = 0
finalMedian = 0
finalStandardDeviation = 0

def computeMean():
     global finalMean
     finalMean = round((totalCount / finalFriendCount),2)
     return finalMean

def computeMedian():
    global finalMedian

    mid = 0
    friendCountList.sort(key=int)

    avg = len(friendCountList) % 2
    if(avg == 0):
        mid = len(friendCountList) / 2
        finalMedian = friendCountList[mid]
    else:
        mid = len(friendCountList) / 2
        mid = mid +1
        finalMedian = friendCountList[mid]
    return finalMedian


def computeStandardDeviation():
     global finalStandardDeviation
     global finalMean
     stdDeviationSum = 0
     for num in friendCountList:
          stdDeviationSum = stdDeviationSum + ((int(num) - int(finalMean)) *  (int(num) - int(finalMe
     finalStandardDeviation = round(math.sqrt(stdDeviationSum / finalFriendCount),2)
     return finalStandardDeviation

with open(’acnwala-friendscount.csv’) as csvfile:
     totalCount = 0
     readCSV = csv.reader(csvfile, delimiter=’,’)
     for row in readCSV:
          if  "FRIENDCOUNT" in row[1]:
               continue
          else:
               friendCountList.append(row[1])
               totalCount = totalCount + int(row[1])
               finalFriendCount = finalFriendCount + 1
     print (totalCount)

friendCountList.sort(key=int)
for num in friendCountList:
     print num


mean = [str(computeMean())]
median = [str(computeMedian())]
stdDeviation = [str(computeStandardDeviation())]
t = Table([mean, median, stdDeviation], names=(’Mean’, ’Median’, ’Standard Deviation’))
print t
```

Problem 1 continued on next page. . .

Listing 2: faceBookFriendsGraph.py

```
import matplotlib.pyplot as plt
import csv

x = []
y = []

with open('plotText.txt','r') as csvfile:
    plots = csv.reader(csvfile, delimiter=',')
    for row in plots:
        x.append(int(row[0]))
        y.append(int(row[1]))

plt.plot(x,y)
plt.plot(11,98,marker='*', color='red',markersize=12)
plt.plot(52,431,marker='*', color='red',markersize=12)
plt.plot(63,536.67,marker='*', color='red',markersize=12)
plt.plot(65,542,marker='*', color='red',markersize=12)
plt.annotate('Dr. Nwala\'s Friends : 98' , xy=(17, 98))
plt.annotate('Median : 431' , xy=(40, 500))
plt.annotate('Standard Deviation : 536.67' , xy=(63, 350),)
plt.annotate('Mean : 542' , xy=(65, 700))
plt.xlim(0, 105)
plt.xlabel('All Friends')
plt.ylabel('No. of Friends for Each Friend')
plt.title('Facebook Friends Vs Each Friend\'s count')
plt.legend()
plt.show()
```

The below plot will show the friendship paradox for **Dr.Alexander Nwala's** facebook account

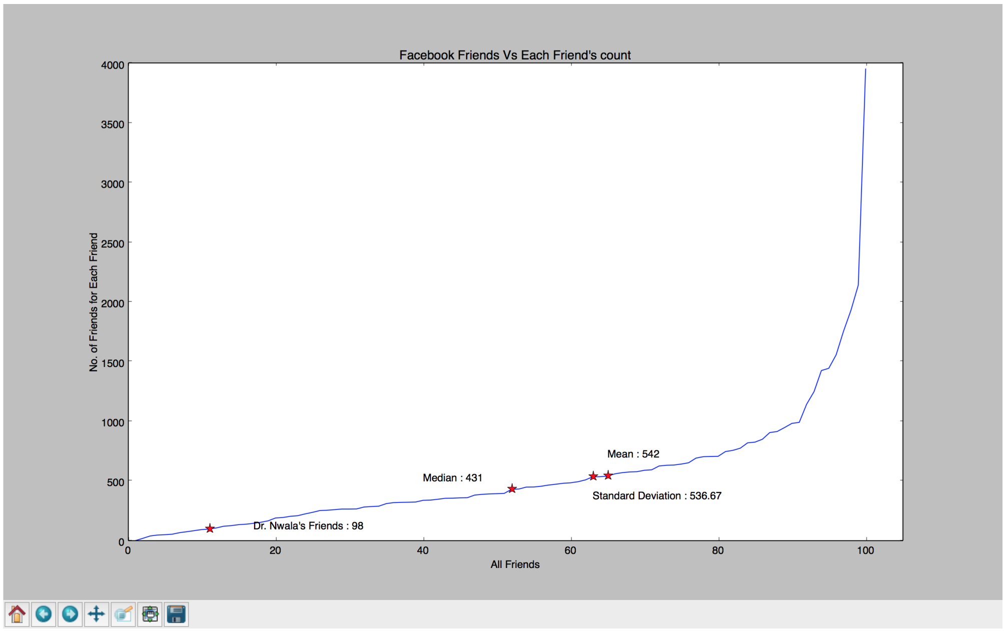

Figure 1: Sample 'curl' with POST

Problem 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 number 1, and calculate the same mean, standard deviation, and median values..

SOLUTION

The program requires to use the access tokens generated while creating the twitter developer account .

1. Download and install the twitter API i.e. **tweepy** :

   ```
   pip install tweepy
   ```

2. Currently running the tweepy on the **acnwala** twitter handle:

   ```
   user = tweepy.Cursor(api.followers, screen_name=twitter_handle,
   count=200).items()
   ```

   The above statement will fetch all the followers details in JSON for the twitter handle.

Listing 3: twitterFollowersCount.py

```
import tweepy
import time
import csv
import math
from astropy.table import Table, Column
import numpy as np
#insert your Twitter keys here

ckey = 'L9QTLPWp2CswcJWRRaNtrsxWO'
csecret = 'nKm7PFmtFAWQYupqffdLz6YWD23VzFlNV8myAei7BaFYDNIoZN'
atoken = '962415725104324608-gJ39MDzaSIlbj44ZBSIuhezb3QcgOAx'
asecret = 'SD9uFFWZH5zfrg4taMdjXkH3vefgmqpPne10EmPiXLijg'
totalCount = 0
listDict = {}
twitter_handle='acnwala'
finalMean = 0
finalMedian = 0
finalStandardDeviation = 0
friendCountList = []
count = 1
finalFriendCount = 0


def computeMean():
    global finalMean
    finalMean = round((finalFriendCount / totalCount),2)
    return finalMean
```

Problem 2 continued on next page. . .

```
def computeMedian():
    global finalMedian

    mid = 0
    friendCountList.sort(key=int)

    avg = len(friendCountList) % 2
    if(avg == 0):
        mid = len(friendCountList) / 2
        finalMedian = friendCountList[mid]
    else:
        mid = len(friendCountList) / 2
        mid = mid +1
        finalMedian = friendCountList[mid]
    return finalMedian


def computeStandardDeviation():
    global finalStandardDeviation
    global finalMean
    stdDeviationSum = 0
    for num in friendCountList:
        stdDeviationSum = stdDeviationSum + ((int(num) - int(finalMean)) *  (int(num) - int(finalMean
    # print('stdDeviationSum',stdDeviationSum)
    finalStandardDeviation = round(math.sqrt(stdDeviationSum / totalCount),2)
    return finalStandardDeviation


auth = tweepy.auth.OAuthHandler(ckey, csecret)
auth.set_access_token(atoken, asecret)
api = tweepy.API(auth,wait_on_rate_limit=True)

if(api.verify_credentials):
    print 'Logged in successfully'

for follower in tweepy.Cursor(api.followers, screen_name=twitter_handle).items():
    totalCount = totalCount + 1
    listDict[follower.screen_name] = follower.friends_count
    friendCountList.append(follower.friends_count)

friendCountList.sort(key=int)
f = open('twitterFollwers-Friends.txt','w')
for friendCount in friendCountList:
    f.write(str(count)+","+str(friendCount))
    f.write('\n')
    count = count + 1
    finalFriendCount = finalFriendCount + friendCount
f.close()

mean = [str(computeMean())]
median = [str(computeMedian())]
```

```
stdDeviation = [str(computeStandardDeviation())]
t = Table([mean, median, stdDeviation], names=('Mean', 'Median', 'Standard Deviation'))
print t
```

Listing 4: twitterFollowersGraph.py

```
import matplotlib.pyplot as plt
import csv

x = []
y = []

with open('twitterFollowers-Friends.txt','r') as csvfile:
    plots = csv.reader(csvfile, delimiter=',')
    for row in plots:
        x.append(int(row[0]))
        y.append(int(row[1]))

plt.plot(x,y)
plt.plot(55,194,marker='*', color='red',markersize=12)
plt.plot(99,551,marker='*', color='red',markersize=12)
plt.plot(163,2321,marker='*', color='red',markersize=12)
plt.plot(191,8601.97,marker='*', color='red',markersize=12)
plt.annotate('Dr. Nwala\'s Followers : 194' , xy=(10, 380))
plt.annotate('Median : 551' , xy=(80, 900))
plt.annotate('Mean : 2321' , xy=(168, 2321),)
plt.annotate('Standard\nDeviation : 8601.97' , xy=(140, 8300))
plt.xlim(0, 200)
plt.ylim(0, 9000)
plt.xlabel('All Followers')
plt.ylabel('No. of Friends for Each Follower')
plt.title('Twitter Followers Vs Each Follower\'s Friend count')
plt.legend()
plt.show()
```

The below plot will show the friendship paradox for **Dr.Alexander Nwala's** twitter handle.

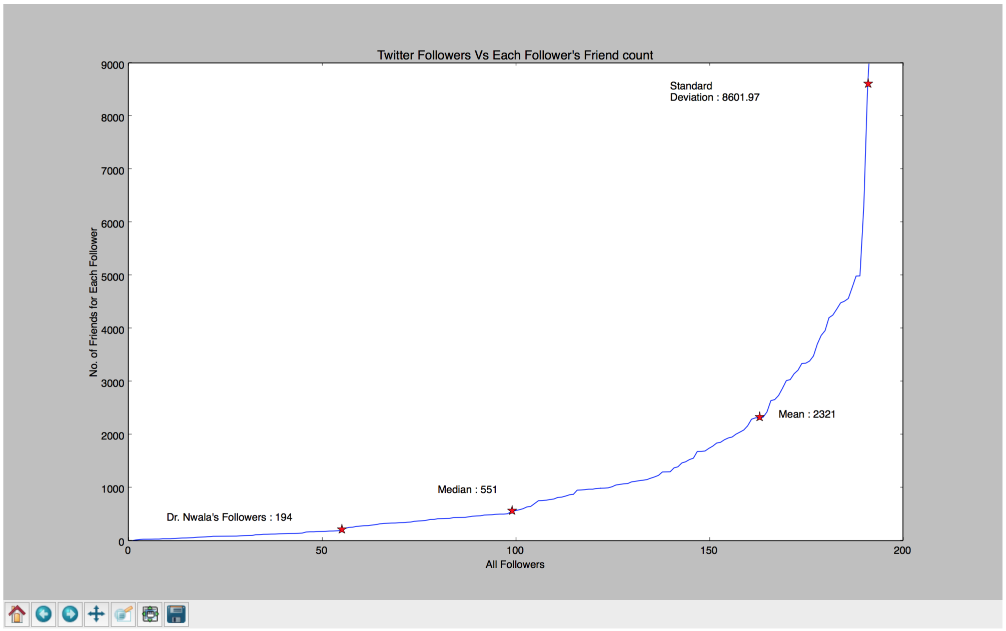

Figure 2: Sample 'curl' with POST

The Sorted friend's list can be found in the **twitterFollowers-Friends.txt** text file.

Problem 3

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

SOLUTION

The program requires to use the access tokens generated while creating the twitter developer account .

1. Running the tweepy on the **acnwala** twitter handle:

   ```
   user = tweepy.Cursor(api.friends, screen_name=twitter_handle,
   count=200).items()
   ```

   The above statement will fetch all the friends list in JSON for the twitter handle.

Listing 5: twitterFriendsCount.py

```
import tweepy
import time
import csv
import math
from astropy.table import Table, Column
import numpy as np
#insert your Twitter keys here

ckey = ’L9QTLPWp2CswcJWRRaNtrsxWO’
csecret = ’nKm7PFmtFAWQYupqffdLz6YWD23VzFlNV8myAei7BaFYDNIoZN’
atoken = ’962415725104324608-gJ39MDzaSIlbj44ZBSIuhezb3QcgOAx’
asecret = ’SD9uFFWZH5zfrg4taMdjXkH3vefgmqpPne10EmPiXLijg’
totalCount = 0
listDict = {}
twitter_handle=’acnwala’
finalMean = 0
finalMedian = 0
finalStandardDeviation = 0
friendCountList = []
count = 1
finalFriendCount = 0


def computeMean():
     global finalMean
     finalMean = round((finalFriendCount / totalCount),2)
     return finalMean


def computeMedian():
     global finalMedian
     mid = 0
     friendCountList.sort(key=int)
     avg = len(friendCountList) % 2
     if(avg == 0):
          mid = len(friendCountList) / 2
```

Problem 3 continued on next page. . .

```
        finalMedian = friendCountList[mid]
    else:
        mid = len(friendCountList) / 2
        mid = mid +1
        finalMedian = friendCountList[mid]
    return finalMedian


def computeStandardDeviation():
    global finalStandardDeviation
    global finalMean
    stdDeviationSum = 0
    for num in friendCountList:
        stdDeviationSum = stdDeviationSum + ((int(num) - int(finalMean)) *  (int(num) - int(finalMe
    finalStandardDeviation = round(math.sqrt(stdDeviationSum / totalCount),2)
    return finalStandardDeviation


auth = tweepy.auth.OAuthHandler(ckey, csecret)
auth.set_access_token(atoken, asecret)
api = tweepy.API(auth,wait_on_rate_limit=True)

if(api.verify_credentials):
    print 'Logged in Successfully'

for following in tweepy.Cursor(api.friends, screen_name=twitter_handle).items():
    totalCount = totalCount + 1
    listDict[following.screen_name] = following.friends_count
    friendCountList.append(following.friends_count)

friendCountList.sort(key=int)
f = open('twitterFriends-Friends.txt','w')
for friendCount in friendCountList:
    f.write(str(count)+","+str(friendCount))
    f.write('\n')
    count = count + 1
    finalFriendCount = finalFriendCount + friendCount
f.close()

mean = [str(computeMean())]
median = [str(computeMedian())]
stdDeviation = [str(computeStandardDeviation())]
t = Table([mean, median, stdDeviation], names=('Mean', 'Median', 'Standard Deviation'))
print t
```

Listing 6: twitterFriendsGraph.py

```
import matplotlib.pyplot as plt
import csv

x = []
y = []

with open(’twitterFriends-Friends.txt’,’r’) as csvfile:
    plots = csv.reader(csvfile, delimiter=’,’)
    for row in plots:
        x.append(int(row[0]))
        y.append(int(row[1]))

plt.plot(x,y)
plt.plot(10,74,marker=’*’, color=’red’,markersize=12)
plt.plot(40,503,marker=’*’, color=’red’,markersize=12)
plt.plot(56,1057,marker=’*’, color=’red’,markersize=12)
plt.plot(61,1555,marker=’*’, color=’red’,markersize=12)
plt.annotate(’Dr. Nwala\’s Friends : 74’ , xy=(12, 50))
plt.annotate(’Median : 503’ , xy=(42, 497))
plt.annotate(’Standard \n Deviation : 1555’ , xy=(63, 1510),)
plt.annotate(’Mean : 1057’ , xy=(61, 1050))
plt.ylim(0, 2000)
plt.xlim(0, 80)
plt.xlabel(’All Friends’)
plt.ylabel(’No. of Friends for Each Friend’)
plt.title(’Twitter Friends Vs Each Friend count’)
plt.legend()
plt.show()
```

The below plot will show the friendship paradox for **Dr.Alexander Nwala’s** twitter handle.

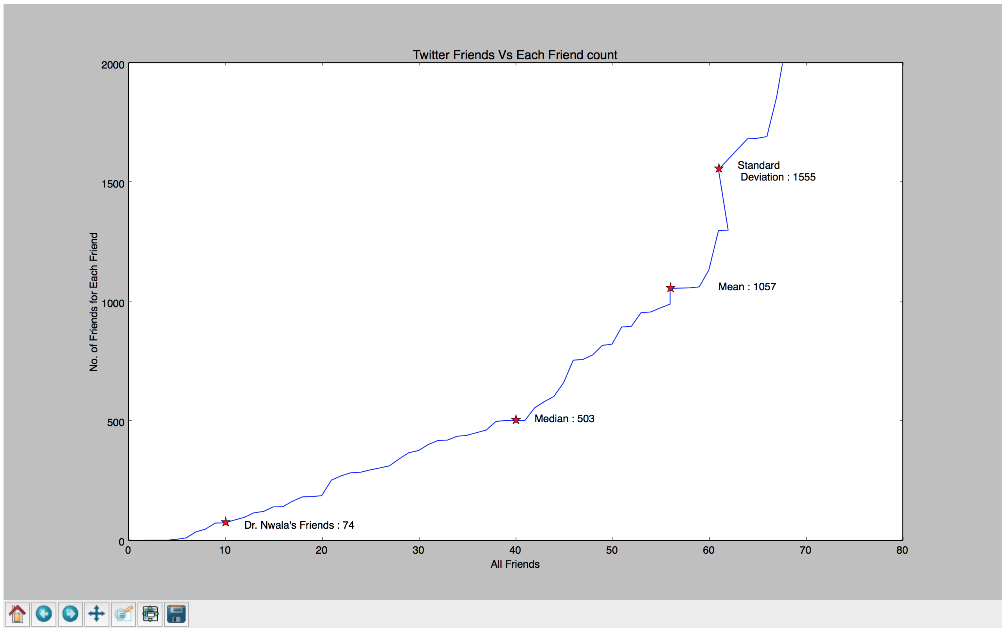

Figure 3: Sample ’curl’ with POST

The Sorted friend's list can be found in the **twitterFriends-Friends.txt** text file.