Srividya Majeti

Assignment 6

CS 532: Introduction to Web Science
Dr. Michael Nelson
Spring 2016

March 18, 2016

Contents

1

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.

To see if two users follow each other, see: `https://dev.twitter.com/rest/reference/get/friendships/show`
Attractiveness of the graph counts! Nodes should be labeled (avatar images are even better), and edge types (follows, following) should be marked.
Note: for getting GitHub to serve HTML (and other media types), see: `http://stackoverflow.com/questions/6551446/can-i-run-html-files-directly-from-github-instead-of-just-viewing-their-source`
Be sure to include the URI(s) for your D3 graph in your report.

Following are the steps I have taken to the solve the problem:

- Using the Twitter API I extracted all my followers and stored 'screen_name', 'name', 'profile_image_url', and 'index number' in a JSON file named as 'nodesData'.
- In function 'links()' I stored the source and target nodes in a dictionary which gives the data for directed links. This code is listed in Listing 1.1.
- I took the above generated 'nodesData' file as input and obtained all possible pairs for my followers. This is stored in the file 'sourceTarget'.
- For all these possible pairs I checked the existence of friendship between each other using 'show_friendship' API.
- This code is listed in Listing 1.2 .
- The final output Json is stored in 'finalJsonData'. This data is taken as input for D3 code to generate a graph.
- The Output of the graph is located in URI `http://bl.ocks.org/majetisiri/b0fe8280da899311640c`. The screen shot of the graph is illustrated in Figure 2.2.

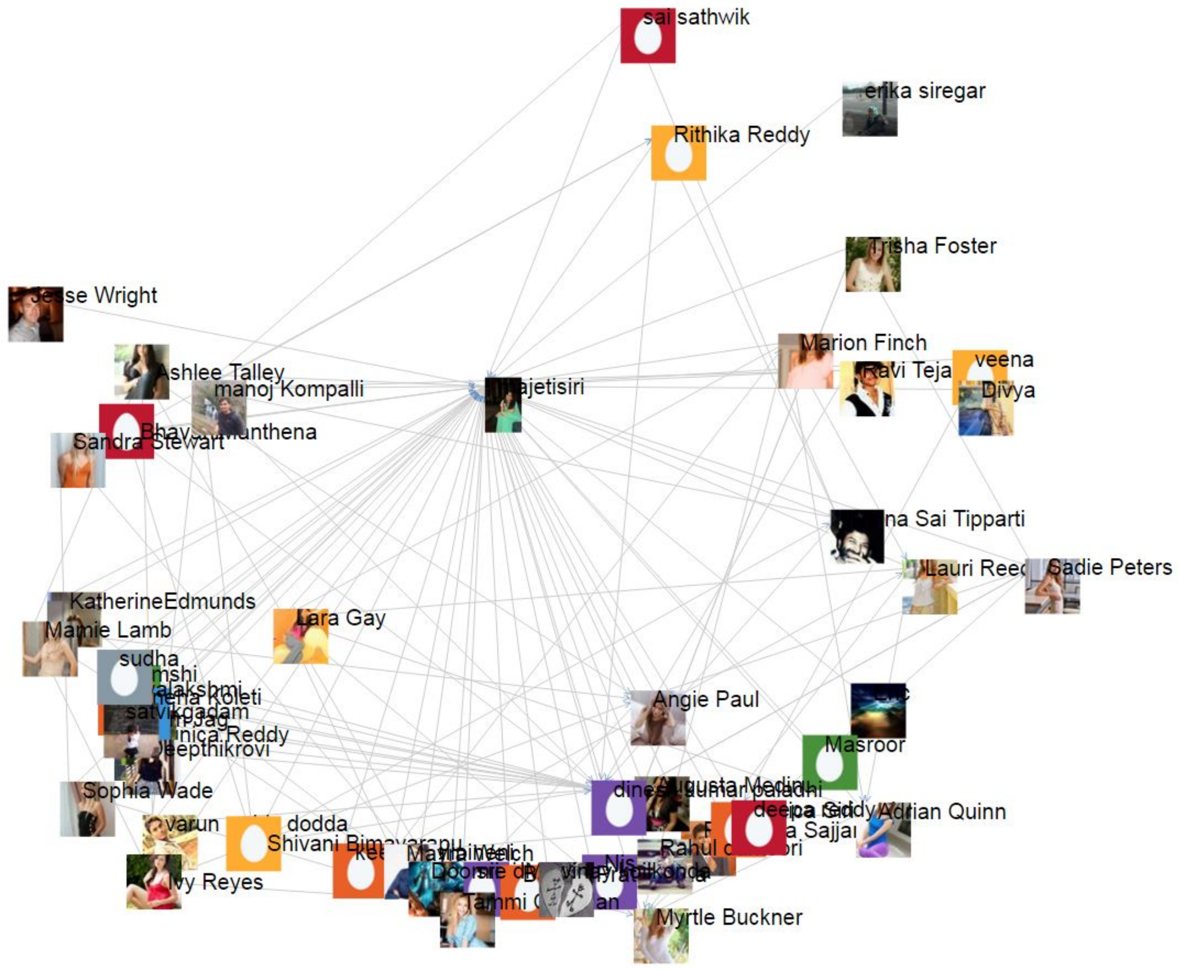


Fig. 1.1. Directed graph between me and my twitter followers including links between my followers if they have any

Code Listing

```
import tweepy
import sys
import json
import time

CONSUMER_KEY = 'wTSsHE3PTA3ZZPiaKHEiQnLtf'
CONSUMER_SECRET = '
    UblYYCmNYIEffAY4T4QHGHXwAWMFqiueXdxf35xZFhoK3AECP1'
```

```
ACCESS_KEY = '157985123-
    WFvzlfDa8KStBZzevMfQBTM7fi8zKHYl2LQpTfGr'
ACCESS_SECRET = '
    lSax0XLwIimJ4VVbuU5OY9BpBic4vsSFi0riAq3DPvTxU'

auth = tweepy.auth.OAuthHandler(CONSUMER_KEY,
    CONSUMER_SECRET)
auth.set_access_token(ACCESS_KEY, ACCESS_SECRET)
api = tweepy.API(auth)

f = open('nodesData','w')
def nodes():
  count = 1
  page_count= 0
  userData = []
  for user in tweepy.Cursor(api.followers, screen_name='
      majetisiri').items():
    usr = {}
    usr['screenName'] = user.screen_name
    usr['name'] =  user.name
    usr['id'] =  count
    usr['img'] =  user.profile_image_url
    usr['link'] = "https://twitter.com/"+user.screen_name
    usr['size'] =  40000
    page_count += 1
    userData.append(usr)
    count+= 1
  f.write(json.dumps(userData)+"\n")
  f.close()

f1 = open('linksData','w')
read = open('nodesData','r')
def links():
  userData =[]
  for line in read:
    data= json.loads(line)
    for user in data:
      dict = {}
      dict['source'] = user['id']
      dict['target'] =  0
      userData.append(dict)
  f1.write(json.dumps(userData)+"\n")
  f1.close()

nodes()
links()
```

Listing 1.1. Python code for getting my followers data

Code Listing

```
import tweepy
import commands
import json
import time

CONSUMER_KEY = 'wTSsHE3PTA3ZZPiaKHEiQnLtf'
CONSUMER_SECRET = '
    UblYYCmNYIEffAY4T4QHGHXwAWMFqiueXdxf35xZFhoK3AECP1'
ACCESS_KEY = '157985123-
    WFvzlfDa8KStBZzevMfQBTM7fi8zKHYl2LQpTfGr'
ACCESS_SECRET = '
    lSax0XLwIimJ4VVbuU5OY9BpBic4vsSFi0riAq3DPvTxU'

auth = tweepy.auth.OAuthHandler(CONSUMER_KEY,
    CONSUMER_SECRET)
auth.set_access_token(ACCESS_KEY, ACCESS_SECRET)
api = tweepy.API(auth)

f1=open('sourceTarget','w')
def getSource():
  read=open('nodesData','r')
  data= json.load(read)
  list= []
  for user in range(0,len(data)):
    sourceScreenName= data[user]["screenName"]
    for user1 in range(user,len(data)-1):
      targetScreenName = data[user1+1]["screenName"]
      checkSourceFollowersAndFollowing(sourceScreenName,
          targetScreenName)

def checkSourceFollowersAndFollowing(sourceScreenName,
    targetScreenName):
  dict= {}
  count = 0
  dict['source']= sourceScreenName
  dict['target']= targetScreenName
  f1.write(json.dumps(dict)+",\n")



def getAllLinks():
  f2 = open('linksData','w')
  read = open('sourceTarget','r')
  data= json.load(read)
  for user in data:
    dict= {}
    sourceScreenName= user["source"]
    targetScreenName= user["target"]
    result = api.show_friendship(source_screen_name=
        sourceScreenName, target_screen_name=
        targetScreenName)
    dict['followed_by'] =result[0].followed_by
    dict['following'] =result[0].following
    dict['screen_name1']= result[0].screen_name
    dict['screen_name2']= result[1].screen_name
    f2.write(json.dumps(dict)+",\n")

def getTrueLinks():
  read = open('linksData','r')
  f2 = open('trueLinksData','w')
  data= json.load(read)
  for user in data:
    dict = {}
    if user["following"] == True:
      dict['source']= user["screen_name1"]
      dict['target']= user["screen_name2"]
      f2.write(json.dumps(dict)+",\n")
    elif user["followed_by"]== True:
      dict['source']= user["screen_name2"]
      dict['target']= user["screen_name1"]
      f2.write(json.dumps(dict)+",\n")

def passTrueLinksSourceAndTarget():
  read = open('trueLinksData','r')
  data= json.load(read)
  for user in data:
    getIds(user["source"],user["target"])

def getIds(name1,name2):
  read = open('nodesData','r')
  f2 = open('linkIds','a')
  data= json.load(read)
  for user in data:
    dict ={}
    if name1 == user["screenName"]:
      id = user["id"]
      # print name1
      dict['source'] = id
      f2.write(json.dumps(dict)+",\n")
    elif name2 == user["screenName"]:
      id = user["id"]
      # print name2
      dict['target'] = id
      f2.write(json.dumps(dict)+",\n")
  f2.close()
getSource()
getAllLinks()
```

```
getTrueLinks()
passTrueLinksSourceAndTarget()
```

Listing 1.2. Python code for checking if friendship exists between my twitter followers

2

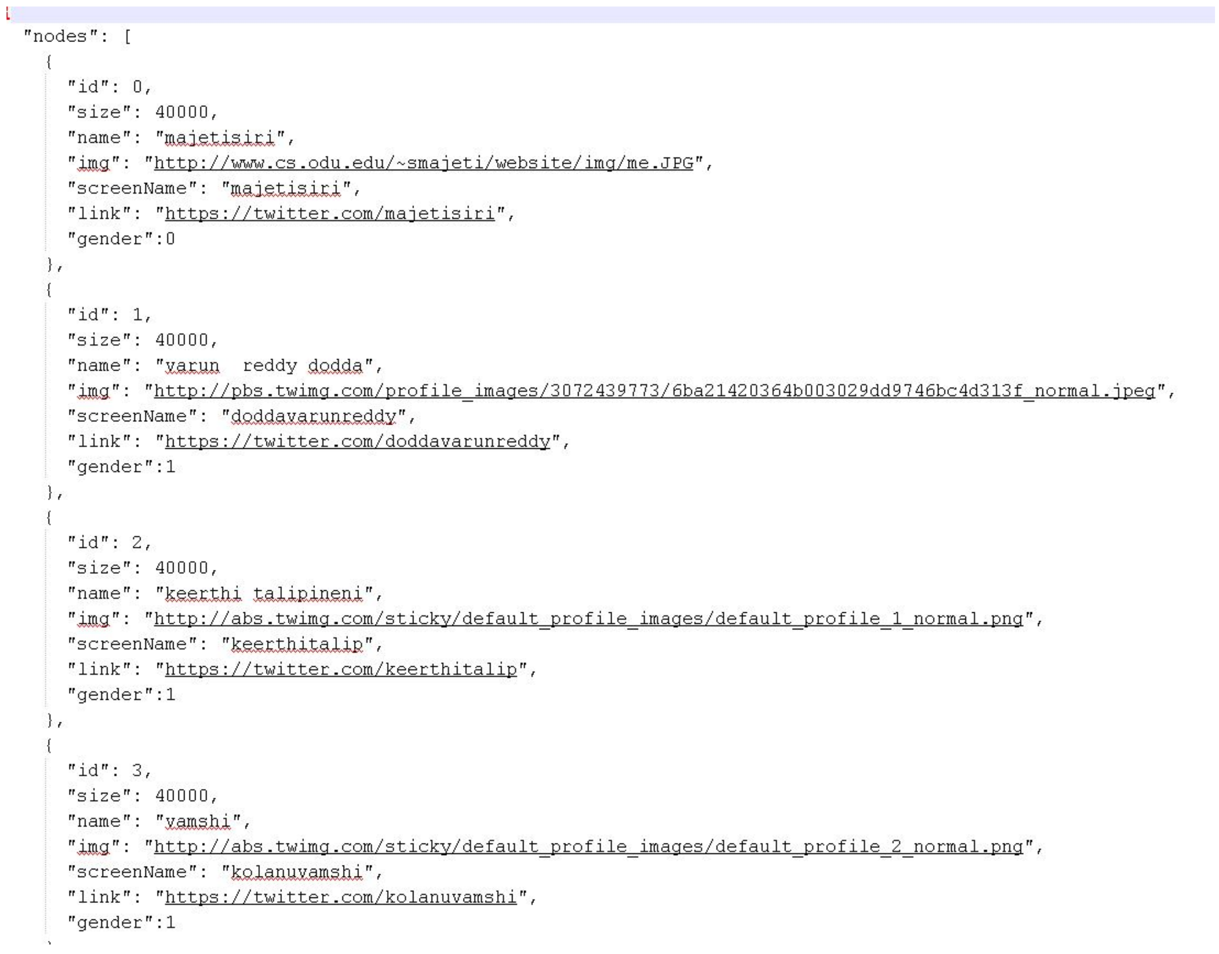
Question 2

Take the Twitter graph you generated in question 1 and test for male-female homophily. For the purposes of this question you can consider the graph as undirected (i.e., no distinction between "follows" and "following"). Use the twitter name (not "screen name"; for example "Michael L. Nelson" and not "@phonedude_mln") and programatically determine if the user is male or female. Some sites that might be useful:
`https://genderize.io/`
`https://pypi.python.org/pypi/gender-detector/0.0.4`
Create a table of Twitter users and their likely gender. List any accounts that can't be determined and remove them from the graph. Perform the homophily test as described in slides 11-15, Week 7. Does your Twitter graph exhibit gender homophily?

Following are the steps I have taken to solve the problem:

- I split the names of each follower into last name, first name and initial.
- By considering the first name of each user I found the gender of each follower using genderize API.
- I updated the JSON data in question 1 by adding gender to each follower.

- The screen shot of updated JSON data is in Figure 2.1



```
{
  "nodes": [
    {
      "id": 0,
      "size": 40000,
      "name": "majetisiri",
      "img": "http://www.cs.odu.edu/~smajeti/website/img/me.JPG",
      "screenName": "majetisiri",
      "link": "https://twitter.com/majetisiri",
      "gender":0
    },
    {
      "id": 1,
      "size": 40000,
      "name": "varun  reddy dodda",
      "img": "http://pbs.twimg.com/profile_images/3072439773/6ba21420364b003029dd9746bc4d313f_normal.jpeg",
      "screenName": "doddavarunreddy",
      "link": "https://twitter.com/doddavarunreddy",
      "gender":1
    },
    {
      "id": 2,
      "size": 40000,
      "name": "keerthi talipineni",
      "img": "http://abs.twimg.com/sticky/default_profile_images/default_profile_1_normal.png",
      "screenName": "keerthitalip",
      "link": "https://twitter.com/keerthitalip",
      "gender":1
    },
    {
      "id": 3,
      "size": 40000,
      "name": "vamshi",
      "img": "http://abs.twimg.com/sticky/default_profile_images/default_profile_2_normal.png",
      "screenName": "kolanuvamshi",
      "link": "https://twitter.com/kolanuvamshi",
      "gender":1
```

Fig. 2.1. Output of updated nodes data

- The code is listed in Listing 2.1.

- This data is taken as input for D3 to generate a graph which differentiates the followers based on gender. All male followers are represented in color 'orange', female followers in color 'green' and those whose gender is not determined are represented in color 'blue'. This graph is illustrated in Fgure 2.2

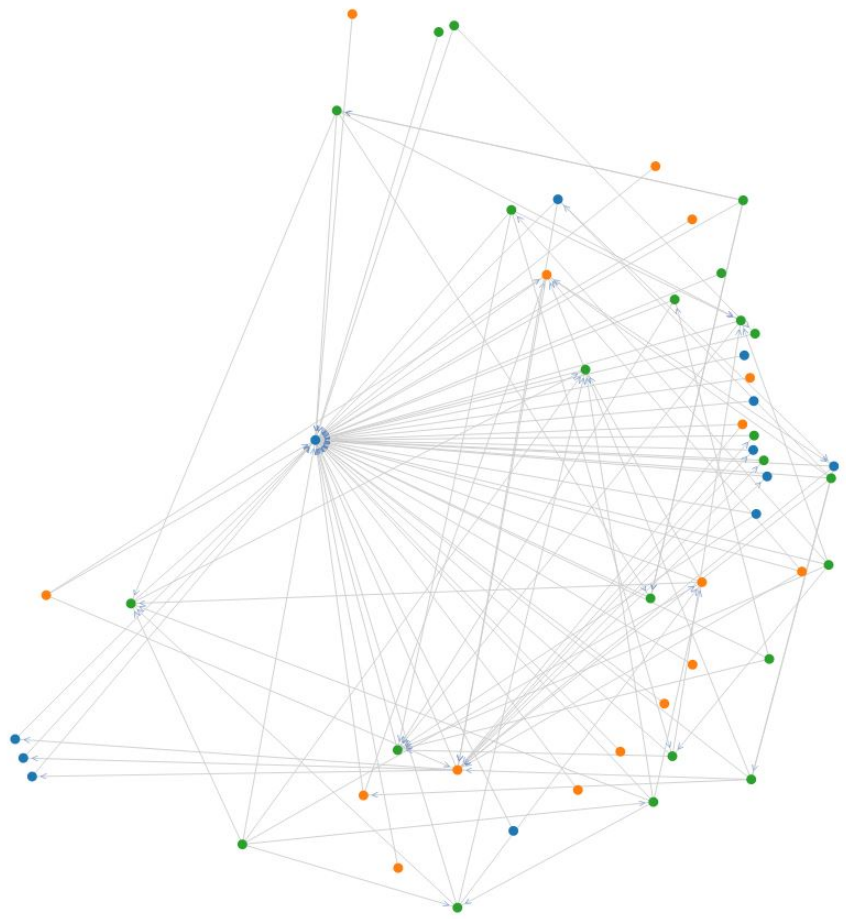


Fig. 2.2. Output Graph with followers distinguished based on gender

- This graph is located at URI `http://bl.ocks.org/majetisiri/fd87d5725027a5441f78`

Code Listing

```
import requests
import json

def function(input):
  LastName = name.split()

read=open('nodesData','r')
f1= open('gender','w')
list = []
for line in read:
  data= json.loads(line)
  for user in data:
    name = (str(user['name']))
    print "\n"+name
    name = name.partition(" ")
    firstName = name[0]
    print "FirstName:"+firstName
    url = "https://api.genderize.io/?name=" +firstName
    r = requests.get(url)
    genderData= r.content
    print genderData
    print type(genderData)
    list.append(r.content)
  f1.write(str(list))
```

Listing 2.1. Python code for getting gender for followers using their first name

3

Extra-Credit Question-3

Using D3, create a graph of the Karate club before and after the split.
- Weight the edges with the data from:
`http://vlado.fmf.uni-lj.si/pub/networks/data/ucinet/zachary.dat`
- Have the transition from before/after the split occur on a mouse click. This is a toggle, so the graph will go back and forth between connected and disconnected.

Following are the steps I have taken to the solve the problem:

- I converted the KarateClub graphML data into JSON structure.
- The code is listed in Listing 3.1
- I took the converted karateClub JSON data as input for my D3 code and generated a force directed graph.
- The output of the graph is located at URI `http://bl.ocks.org/majetisiri/316e3a1537b469154779`. The graph appears more accurately in 'Google chrome' than in any other browsers.
- Above the graph their are 2 buttons with captions 'Before split' and 'After split'. If we click on the button 'Before split' it generates a graph with all the nodes in same color. When we hover on the nodes of the graph, it displays the name of the node. This is illustrated in Figure 3.1.

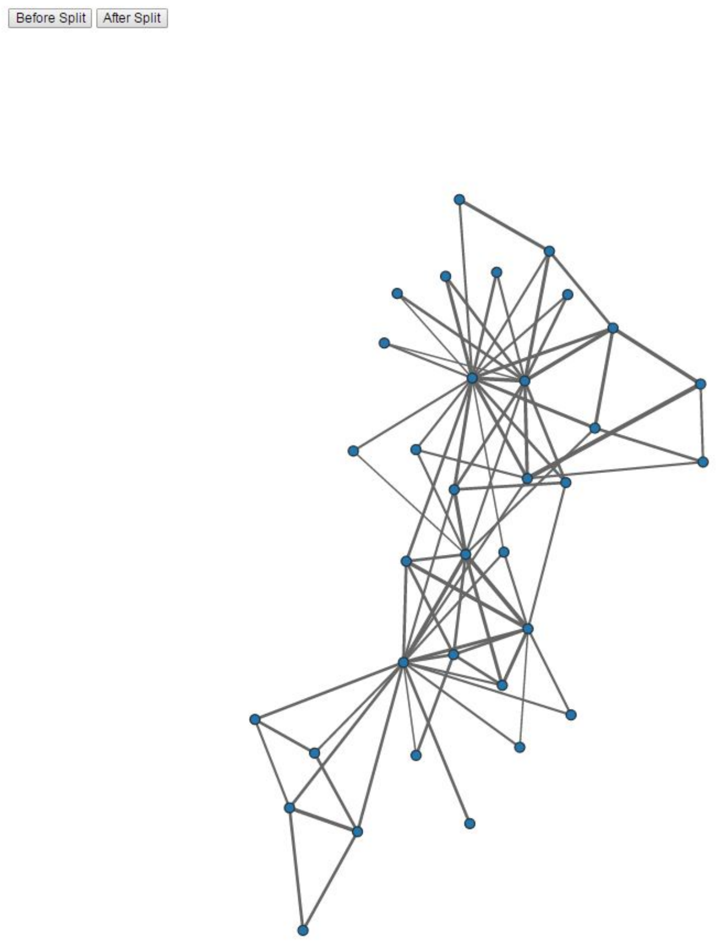


Fig. 3.1. Graph before split

- If we click on the button ‘After split’ it generates a graph which differentiates 2 groups based on faction. This is illustrated in Figure 3.1.

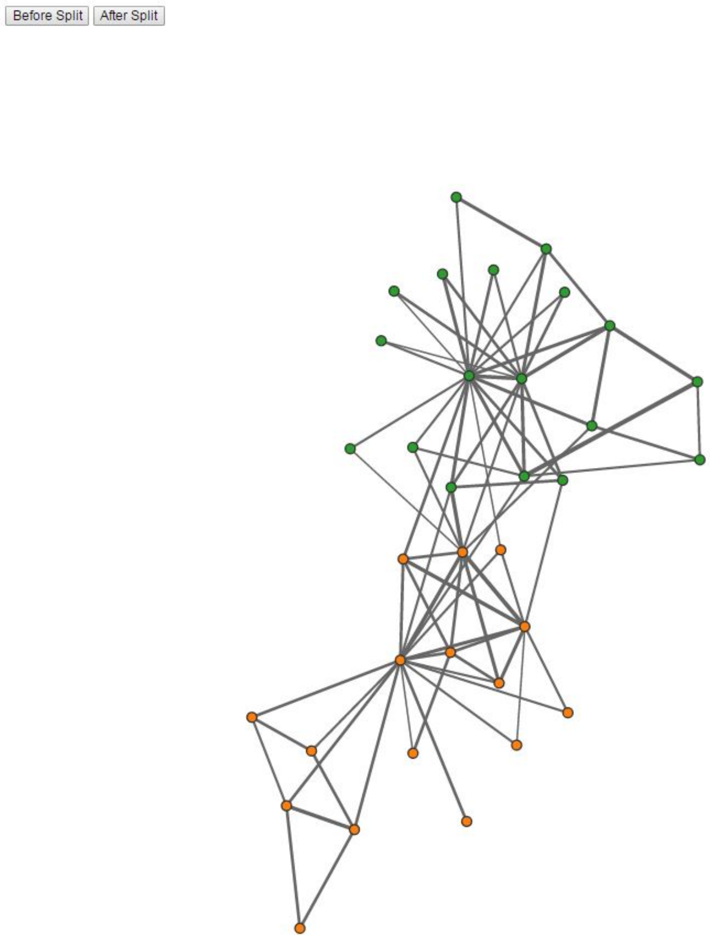


Fig. 3.2. Graph after split which distinguishes two groups based on Faction

Code Listing

```
from xml.etree import ElementTree
import json

with open('karate.GraphML', 'rt') as f:
  f1= open('karateClub.json','w')
  tree = ElementTree.parse(f)
  root = tree.getroot()
  print root
  data = {}
  data1={}
  nodes = []
```

```
links= []
count = 0
f1.write('{' + '\n' + '"nodes":\n' + '[' +"\n" )
for parent in root:
  for child in parent:
    if child.tag == ('{http://graphml.graphdrawing.org/
        xmlns}node'):
      id = child.get('id')
      print "id:",id
      for children in child:
        if children.attrib.get('key') == 'name':
          name= children.text
          print "name:",name
          data["name"] = name
          f1.write(json.dumps(data)+',\n')
        if children.attrib.get('key') == 'Faction':
          Faction= children.text
          print "Faction:",Faction
          data["faction"] = int(Faction)
          data["color"] = 1
f1.write('],' + '\n' + '"links":\n' + '[' +"\n" )
for parent in root:
  for child in parent:
    if child.tag == ('{http://graphml.graphdrawing.org/
        xmlns}edge'):
      source = child.get('source')
      count +=1
      data1["source"] = int(source)
      target = child.get('target')
      data1["target"] = int(target)
      for children in child:
        if children.attrib.get('key') == 'weight':
          weight= children.text
          data1["weight"] = int(weight)
          data1["id"] = "e"+str(count)
          f1.write(json.dumps(data1)+",\n")
f1.write(']' + '\n' + "}")
```

Listing 3.1. Python code for converting KarateClub XML graph to JSON data

References

1. D3 graph for question 1: http://bl.ocks.org/mbostock/1153292, Mike Bostock, 2016
2. D3 graph for question 3: https://bl.ocks.org/mbostock/4062045, Mike Bostock, 2016
3. How to change data on click button: http://bl.ocks.org/d3noob/7030f35b72de721622b8, d3noob Block, 2015
4. Force directed layout with images: http://bl.ocks.org/eesur/be2abfb3155a38be4de4, Sundars Block , 2015
5. How to split a name: http://stackoverflow.com/questions/1720503/parsing-peoples-first-and-last-name-in-python , Joel Spolsky, 2008