CS 532: Assignment 6

Dinesh Kumar Paladhi

Spring 2016

Contents

1 Problem 1

```
 D3 graphing (5 points)

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-vie

Be sure to include the URI(s) for your D3 graph in your report.
```

1.1 Solution

1. The aim of this question is to get list of followers for any account and find the friendship among the followers,i.e to find whether any followers follow each other so that they will have the links to the source account and links to each other.
2. Here I used my account "dineshpaladhi" because I have more than 50 followers.u.
3. So, here my first step is to get all my followers list.
4. I wrote python code for it using tweepy library and the code can be found in listing1.
5. I have taken name, screen name, imageurl, and gave an id to each follower of mine.
6. Now I have to find the friendship among all my followers, the code for this is found in listing2.
7. Here I got a lot of errors like "Rate Limit Error", "Not Authorized Error",etc.
8. It took a lot of time to figure this out and I found that "Rate Limit Error" is because of the excess input I am giving to the program, so I broken down my input list into smaller parts and ran the program repeatedly with smaller inputs.
9. I deleted few users and their combination of friendships because they give "Not Authorized Error" as they have some privacy settings. This list can be found in figure1.
10. Later, I got all the combination of possible friendships among my users. I have 61 followers so total combinations that I got is 61*60/2 = 1830. The sample list of this can be found in figure2.
11. This list is then processed using the reference provided in the questions and it gave me a list of followers who follow each other. The sample list of this can be found in figure3.
12. The final sample json file can be seen in figure4 and this json file is given as input to the html code which can be seen in listing3.
13. The d3 graph can be seen at this link `http://bl.ocks.org/PaladhiDinesh/56e1843c31960ecfe919` and the sample screenshot of it can be seen in figure5.
14. This is a directed graph, so we can view who follows who using the arrow mark.

1.2 Code Listing

```
import tweepy
import json
import time

output_file = open("followers.json", "w")

ACCESS_TOKEN = '118623489-QsSuqItzx8cnReRHI67ylffqpOPNs7z4Qp8hcOiI'  # Variables that contains the user credentials to access Twitter API
ACCESS_SECRET = 'PAPovgDO6QPy9QV8BbllM8p2MGWrcLLD8pesMHjXxTEMl'
CONSUMER_KEY = 'wxSZ8GSC7aRC7dAsM3m7UqgIg'
CONSUMER_SECRET = 'HuauQk780HuKWyQky9e4J6QM1DlwVxHXuvrLbgHGWhkmRXlvE4'


count = 0

auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)    #Authentication is handled by the tweepy.AuthHandler class
auth.set_access_token(ACCESS_TOKEN, ACCESS_SECRET)

elements={}
edge_elements ={}
count =0
output_file.write('{\n "nodes": [\n')
#try:
api = tweepy.API(auth) # Construct the API instance
elements['screen_name']="dineshpaladhi"
elements['name']="dinesh"
elements['image_url']="https://github.com/favicon.ico"
elements['id']=0
output_file.write(json.dumps(elements)+',\n')
for user in tweepy.Cursor(api.followers, screen_name="dineshpaladhi").items():
        count=count+1
        elements['screen_name']=user.screen_name
        elements['name']=user.name
        elements['id']=count
        elements['image_url']=user.profile_image_url_https
        output_file.write(json.dumps(elements)+',\n')
output_file.write('],\n')
output_file.write('"links": [\n')
count=0
for user in tweepy.Cursor(api.followers, screen_name="dineshpaladhi").items():
        count=count+1
        edge_elements['source']= count
        edge_elements['target']=0
        output_file.write(json.dumps(edge_elements)+',\n')
output_file.write(']\n}')
```

Listing 1: Python code for getting twitter followers for my account

1.3 Code Listing

```
import tweepy
import json
ACCESS_TOKEN = '118623489-QsSuqItzx8cnReRHI67ylffqpOPNs7z4Qp8hcOiI'  # Variables that contains the user credentials to access Twitter API
ACCESS_SECRET = 'PAPovgDO6QPy9QV8BbllM8p2MGWrcLLD8pesMHjXxTEMl'
CONSUMER_KEY = 'wxSZ8GSC7aRC7dAsM3m7UqgIg'
CONSUMER_SECRET = 'HuauQk780HuKWyQky9e4J6QM1DlwVxHXuvrLbgHGWhkmRXlvE4'



auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)    #Authentication is handled by the tweepy.AuthHandler class
auth.set_access_token(ACCESS_TOKEN, ACCESS_SECRET)

api = tweepy.API(auth)

dict1={}
#f2 = open('f2.json','a')
#f3= open ('trues.json','w')
f5=open('follower_links','r+')
def function():
        edge_elements ={}
        my_foll = open('followers.json','r')
        f1 = open('f1.json','w')

        count=0
        # for user in tweepy.Cursor(api.followers, screen_name="dineshpaladhi").items():
                # count=count+1
        each_line = json.load(my_foll)
        dict={}

        for i in range(1,61):
                source_name= each_line["nodes"][i]["screen_name"]
                #print source_name
                for j in range(i,61):
                # try:
                        target_name=each_line["nodes"][j+1]["screen_name"]
                        dict['source']=source_name
                        dict['target']=target_name
                        f1.write(json.dumps(dict)+',\n')

                        # print source_name+','+target_name



def func():
        a=open('f1.json','r')
        each_line1 = json.load(a)
        for user in each_line1:
                source_name=user["source"]
                target_name=user["target"]
                frndshp = api.show_friendship(source_screen_name=source_name, target_screen_name=target_name,count=180)
                # #print type(frndshp)
                #if (frndshp[0].following == True):
                dict1['source1']=frndshp[0].screen_name
                dict1['following']=frndshp[0].following
                dict1['target1']=frndshp[1].screen_name
                dict1['followed_by']=frndshp[0].followed_by
                f2.write(json.dumps(dict1)+',\n')
dict3={}
def get_trues():
        f2 = open('f2.json','r')
        each_line2 = json.load(f2)
        for user in each_line2:
                if(user["followed_by"]== True):
                        dict3['source']=user["target1"]
                        dict3['target']=user["source1"]
                        f3.write(json.dumps(dict3)+',\n')
                if(user["following"]== True):
```

```
                        dict3['source']=user["source1"]
                        dict3['target']=user["target1"]
                        f3.write(json.dumps(dict3)+',\n')
dict5={}
set=()
def getids():
        f4 = open('followers.json','r')
        each_line4 = json.load(f4)
        f3 = open('trues.json','r')
        each_line3 = json.load(f3)
        for user in each_line3:
                src_name=user['source']
                tar_name=user['target']
                #print src_name+','+tar_name
                for i in range(1,61):
                        source_n= each_line4["nodes"][i]["screen_name"]
                        # if(src_name==source_n):
                                # dict5['source']=each_line4["nodes"][i]["id"]
                                # f5.write(json.dumps(dict5)+',\n')
                        if(tar_name==source_n):
                                dict5['target']=each_line4["nodes"][i]["id"]
                                f5.write(json.dumps(dict5)+',\n')


getids()
#get_trues()
#function()
#func()
```

Listing 2: Python code to find out friendship between my followers.

1.4 Code Listing

```
<!DOCTYPE html>
<meta charset="utf-8">
<style>

.link {
  stroke: #ccc;
}

.node text {
  pointer-events: none;
  font: 10px sans-serif;
}

</style>
<body>
<script src="//d3js.org/d3.v3.min.js"></script>
<script>

var width =1000,
    height = 1000

var svg = d3.select("body").append("svg")
    .attr("width", width)
    .attr("height", height);

var force = d3.layout.force()
    .gravity(0.05)
    .distance(300)
    .charge(-100)
    .size([width, height]);

d3.json("followers.json", function(error, json) {
  if (error) throw error;

  svg.append("defs").selectAll("marker")
    .data(["suit", "licensing", "resolved"])
  .enter().append("marker")
    .attr("id", function(d) { return d; })
    .attr("viewBox", "0 -5 10 10")
    .attr("refX", 25)
    .attr("refY", 0)
    .attr("markerWidth", 6)
    .attr("markerHeight", 6)
    .attr("orient", "auto")
  .append("path")
    .attr("d", "M0,-5L10,0L0,5 L10,0 L0, -5")
    .style("stroke", "#4679BD")
    .style("opacity", "0.6");

  force
      .nodes(json.nodes)
      .links(json.links)
      .start();

  var link = svg.selectAll(".link")
      .data(json.links)
    .enter().append("line")
      .attr("class", "link")
          .style("marker-end",  "url(#suit)"); // Modified line

  var node = svg.selectAll(".node")
      .data(json.nodes)
    .enter().append("g")
      .attr("class", "node")
      .call(force.drag);

  node.append("image")
        .attr("xlink:href",  function(d) { return d.image_url;})
      .attr("x", -8)
      .attr("y", -8)
```

```
71        .attr("width", 25)
72        .attr("height", 25);
73
74    node.append("text")
75        .attr("dx", 12)
76        .attr("dy", ".35em")
77        .text(function(d) { return d.name });
78
79    force.on("tick", function() {
80      link.attr("x1", function(d) { return d.source.x; })
81          .attr("y1", function(d) { return d.source.y; })
82          .attr("x2", function(d) { return d.target.x; })
83          .attr("y2", function(d) { return d.target.y; });
84
85      node.attr("transform", function(d) { return "translate(" + d.x + "," + d.y + ")"; });
86    });
87  });
88
89  </script>
```

Listing 3: HTML code with d3 to get force directed graph

1.5 Results

Deleted Users


```
{"source": "erikaris", "target": "JPravallika"},
{"source": "erikaris", "target": "joseph_udithraj"},
{"source": "erikaris", "target": "Rajitha1829"},
{"source": "erikaris", "target": "satvikgadam"},
{"source": "kolanuvamshi", "target": "MommyOddenino"},
{"source": "joseph_udithraj", "target": "Rajitha1829"},
{"source": "joseph_udithraj", "target": "satvikgadam"},
```


Figure 1: Sample list of users who give not authorized error

Sample output1


```
{"source": "erikaris", "target": "doddavarunreddy"},
{"source": "erikaris", "target": "9ulovesu"},
{"source": "erikaris", "target": "kolanuvamshi"},
{"source": "erikaris", "target": "raviyyaahhoo"},
{"source": "erikaris", "target": "ShivaniBima"},
{"source": "erikaris", "target": "ManthenaBhavani"},
{"source": "erikaris", "target": "Manoj_Chandra11"},
{"source": "erikaris", "target": "RithikaR9"},
{"source": "erikaris", "target": "majetisiri"},
{"source": "erikaris", "target": "KumarPaladhi"},
{"source": "erikaris", "target": "Skumars317"},
{"source": "erikaris", "target": "papu100030"},
{"source": "erikaris", "target": "CsUdaykumar"},
{"source": "erikaris", "target": "RithvikKranti"},
{"source": "erikaris", "target": "bill_owns24x7"},
{"source": "erikaris", "target": "TCATIndia"},
{"source": "erikaris", "target": "rlnsrlns"},
{"source": "erikaris", "target": "alokraj68"},
{"source": "erikaris", "target": "JPravallika"},
{"source": "erikaris", "target": "vamsikrishna657"},
{"source": "erikaris", "target": "maithri3"},
{"source": "erikaris", "target": "KlaraAdrinson"},
{"source": "erikaris", "target": "DeonneLivingsto"},
{"source": "erikaris", "target": "LatrinaMcRattig"},
{"source": "erikaris", "target": "MommyOddenino"},
{"source": "erikaris", "target": "triskadieka"},
{"source": "erikaris", "target": "urs_dineshj"},
{"source": "erikaris", "target": "joseph_udithraj"},
{"source": "erikaris", "target": "mohanchandranp"},
{"source": "erikaris", "target": "VasaviMIC"},
{"source": "erikaris", "target": "vineethchandr12"},
{"source": "erikaris", "target": "Harishkumar1772"},
{"source": "erikaris", "target": "pranay2012"},
{"source": "erikaris", "target": "Paladhi"},
{"source": "erikaris", "target": "BvriteceA10"},
{"source": "erikaris", "target": "reddy1516"},
{"source": "erikaris", "target": "Harish0123Te"},
{"source": "erikaris", "target": "Bharathmotha"},
{"source": "erikaris", "target": "sunny24051018"},
{"source": "erikaris", "target": "Rajitha1829"},
```


Figure 2: Total combinations of possible friendship among my followers

Sample output2


```
[{"following": false, "source1": "erikaris", "followed_by": false, "target1": "doddavarunreddy"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "9ulovesu"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "kolanuvamshi"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "raviyyaahhoo"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "ShivaniBima"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "ManthenaBhavani"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "Manoj_Chandra11"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "RithikaR9"},
{"following": true, "source1": "erikaris", "followed_by": true, "target1": "majetisiri"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "KumarPaladhi"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "Skumars317"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "papu100030"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "CsUdaykumar"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "RithvikKranti"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "bill_owns24x7"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "TCATIndia"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "rlnsrlns"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "alokraj68"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "vamsikrishna657"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "maithri3"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "KlaraAdrinson"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "DeonneLivingsto"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "LatrinaMcRattig"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "MommyOddenino"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "triskadieka"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "urs_dineshj"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "mohanchandranp"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "VasaviMIC"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "vineethchandr12"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "Harishkumar1772"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "pranay2012"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "Paladhi"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "BvriteceA10"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "reddy1516"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "Harish0123Te"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "Bharathmotha"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "sunny24051018"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "bhanupradeep1"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "amrutpatil3210"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "AkkineniAkshay"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "vaisham92"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "apoorvadasari16"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "freinds4u2002"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "aruna_teluguone"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "telugufirst"},
{"following": false, "source1": "erikaris", "followed_by": false, "target1": "destroyinangel"},
```


Figure 3: Friendship among followers are shown here as source and target

Sample final json file


```
{"id": 40, "image_url": "https://pbs.twimg.com/profile_images/2818970364/bfed1152c1611fd70a647ae314dda6e7_normal.jp
{"id": 41, "image_url": "https://abs.twimg.com/sticky/default_profile_images/default_profile_3_normal.png", "screen
{"id": 42, "image_url": "https://pbs.twimg.com/profile_images/2648759119/84c20aa2aa6323786cfb8ad7e53134ce_normal.jp
{"id": 43, "image_url": "https://pbs.twimg.com/profile_images/2260531874/Vetrimaaran-Siddharth_normal.jpg", "screen
{"id": 44, "image_url": "https://abs.twimg.com/sticky/default_profile_images/default_profile_5_normal.png", "screen
{"id": 45, "image_url": "https://pbs.twimg.com/profile_images/702385100558897156/hT1Ry8mw_normal.jpg", "screen_name
{"id": 46, "image_url": "https://pbs.twimg.com/profile_images/877387796/Appu_jm_normal.JPG", "screen_name": "apoorv
{"id": 47, "image_url": "https://abs.twimg.com/sticky/default_profile_images/default_profile_0_normal.png", "screen
{"id": 48, "image_url": "https://abs.twimg.com/sticky/default_profile_images/default_profile_2_normal.png", "screen
{"id": 49, "image_url": "https://pbs.twimg.com/profile_images/567043481311256577/Di4NYmjG_normal.png", "screen_name
{"id": 50, "image_url": "https://pbs.twimg.com/profile_images/1102796321/PhotoFunia-a2682e_normal.jpg", "screen_nam
{"id": 51, "image_url": "https://abs.twimg.com/sticky/default_profile_images/default_profile_4_normal.png", "screen
{"id": 52, "image_url": "https://abs.twimg.com/sticky/default_profile_images/default_profile_5_normal.png", "screen
{"id": 53, "image_url": "https://abs.twimg.com/sticky/default_profile_images/default_profile_1_normal.png", "screen
{"id": 54, "image_url": "https://abs.twimg.com/sticky/default_profile_images/default_profile_4_normal.png", "screen
{"id": 55, "image_url": "https://abs.twimg.com/sticky/default_profile_images/default_profile_6_normal.png", "screen
{"id": 56, "image_url": "https://pbs.twimg.com/profile_images/552928112426491904/Y74yoiU9_normal.jpeg", "screen_nam
{"id": 57, "image_url": "https://abs.twimg.com/sticky/default_profile_images/default_profile_3_normal.png", "screen
{"id": 58, "image_url": "https://pbs.twimg.com/profile_images/224835851/girl-w-pink-hair-ties-1_normal.jpg", "scree
{"id": 59, "image_url": "https://pbs.twimg.com/profile_images/553265160161218560/T8Z23mXY_normal.jpeg", "screen_nam
{"id": 60, "image_url": "https://pbs.twimg.com/profile_images/685750015994806272/NQIG1DUP_normal.jpg", "screen_name
{"id": 61, "image_url": "https://pbs.twimg.com/profile_images/1876827960/419881_2745270110659_1224709823_32412429_1
],
"links": [
{"source": 1, "target": 0},
{"source": 2, "target": 0},
{"source": 3, "target": 0},
{"source": 4, "target": 0},
{"source": 5, "target": 0},
{"source": 6, "target": 0},
{"source": 7, "target": 0},
{"source": 8, "target": 0},
{"source": 9, "target": 0},
{"source": 10, "target": 0},
{"source": 11, "target": 0},
{"source": 12, "target": 0},
{"source": 13, "target": 0},
{"source": 14, "target": 0},
{"source": 15, "target": 0},
{"source": 16, "target": 0},
{"source": 17, "target": 0},
{"source": 18, "target": 0},
{"source": 19, "target": 0},
{"source": 20, "target": 0},
{"source": 21, "target": 0},
{"source": 22, "target": 0},
```


Figure 4: Final json file with nodes as followers and links as friendship between them

D3 Graph

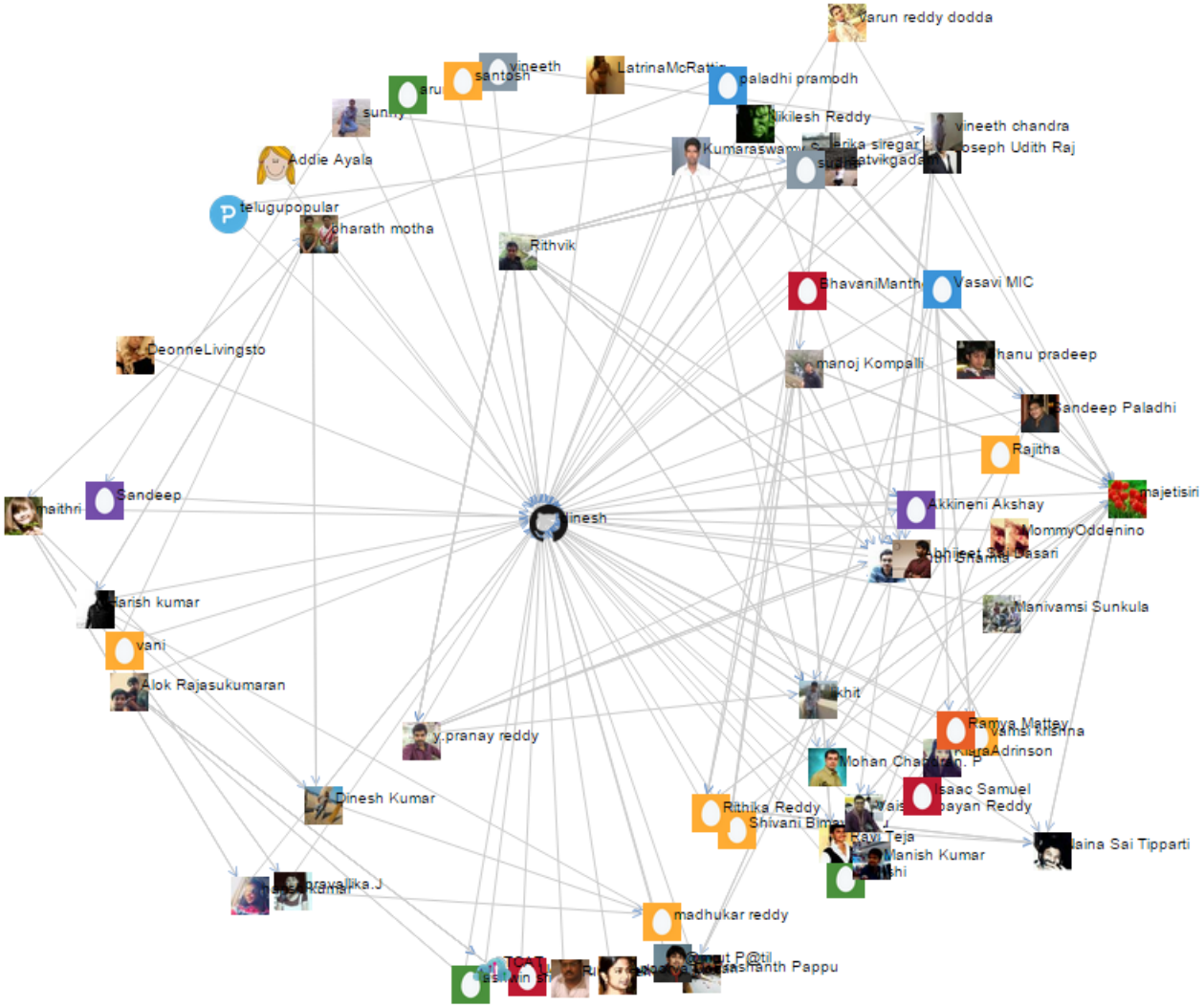

Figure 5: D3 graph showing my twitter followers and friendship among my followers

2 Problem 2

```
Gender homophily in your Twitter graph (5 points)

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?
```

2.1 Solution

1. This question is continuation of the above question with some changes. Here we need to use the output of the previous question and modify the output.
2. I need to find out the genders of all my followers and find out whether there is gender homophily or not.
3. I used follower's first name to find out their genders using the reference provided in the question.
4. So, I took all my followers list in json file and gave it as input to my python code which processes each follower's first name and gives their gender as output.
5. The python code used here can be found in listing4 and the sample output of it can be seen in figure6.
6. Then I created a new json file which includes all the follower's names,genders and I gave sample id for each of them. This can be seen here in figure7.
7. The output tells me whether a follower is "Male","Female" or "Null".
8. I removed all the users who have their gender as "Null" because it was said so in the question and this helps me to find gender homophily easily.
9. This json file is given as input to my html code which gives a force directed graph. The code of the html file can be seen in listing5.
10. The working model of this d3 graph can be seen at this link `http://bl.ocks.org/PaladhiDinesh/c7ad7ffc4fb17f4f9411` and the sample screenshot of it can be seen in figure8.
11. This is an undirected graph and when hovered on a node with the mouse pointer you can see the name of the follower. The nodes in blue are Male and in orange are Female.
12. This twitter graph exhibits gender homophily as we can easily see that more blue are more connected to blue nodes which says that Male followers are more connected to other Male followers.

Table 1: Table with followers and gender

Name — Gender
erika — female
varun — male
Naina — female
vamshi — male
Ravi — male
Shivani — female
BhavaniManthena — null
manoj — male
Rithika — null
majetisiri— null
Manish — male
Kumaraswamy — male
Prashanth — male
Uday — male
Rithvik — null
ashwin — male
TCAT — null
NaveenKumar — null
pravallika — null
maithri — null
KlaraAdrinson — null
DeonneLivingsto — null
LatrinaMcRattig — null
MommyOddenino— null
Nikilesh— null
Dinesh— male
Joseph— male
Mohan— male
Vasavi— female
vineeth— male
harish‘— male
pranay— null
Sandeep— male
vani— female
madhukar— male
Harish— male
bharath— male
sunny— male
Rajitha— female
bhanu — male
mrut — null
Akkineni — male
Vaishampayan— null
Apoorva— female
paladhi — null
aruna— female
telugupopular— null
satvikgadam — null
sudha — female
Sandeep — male
Ramya — female

2.2 Code Listing

```
1  import requests
2  import json
3  input= open('followers.json','r')
4  output=open('gender_data.json','w')
5  each_line=json.load(input)
6  dict={}
7  for i in range(1,61):
8          name= each_line["nodes"][i]["name"]
9
10         name=name.partition(" ")
11         fname=name[0]
12         #print fname
13         url="https://api.genderize.io/?name=" +fname
14         gen=requests.get(url)
15 #        print type(gen)
16         dict['gen_data']=gen.content
17         #print gen_data
18         #print type(gen_data)
19         output.write(json.dumps(dict)+',\n')
```

Listing 4: Python Code for getting finding gender of each follower

2.3 Code Listing

```
<!DOCTYPE html>
<meta charset="utf-8">
<style>

.link {
  stroke: #ccc;
}

.node text {
  pointer-events: none;
  font: 20px sans-serif;
}

</style>
<body>
<script src="//d3js.org/d3.v3.min.js"></script>
<script>

var width = 1000,
    height = 1000

        var color = d3.scale.category10();

var svg = d3.select("body").append("svg")
    .attr("width", width)
    .attr("height", height);

var force = d3.layout.force()
    .gravity(0.05)
    .distance(325)
    .charge(-100)
    .size([width, height]);

d3.json("followers_gen.json", function(error, json) {
  if (error) throw error;

  force
      .nodes(json.nodes)
      .links(json.links)
      .start();

  var link = svg.selectAll(".link")
      .data(json.links)
    .enter().append("line")
      .attr("class", "link")
         // .style("marker-end",  "url(#suit)") // Modified line ;

  var node = svg.selectAll(".node")
      .data(json.nodes)
    .enter().append("circle")
      .attr("class", "node")
          .attr("r", 9)
      .style("fill", function(d) { return color(d.gender); })
      .call(force.drag);

        node.append("title")
      .text(function(d) { return d.name; });

        var setEvents = node
          // Append hero text
          .on( 'click', function (d) {
              d3.select("h1").html(d.name);
              d3.select("h2").html(d.screenName);
              d3.select("h3").html ("Take me to " + "<a href='" + d.link + "' >"  + d.name +
                    " web page    "+ "</a>" );
           })

          .on( 'mouseenter', function() {
            // select element in current context
            d3.select( this )
```

```
                .transition()
                .attr("x", function(d) { return -60;})
                .attr("y", function(d) { return -60;})
                .attr("height", 100)
                .attr("width", 100);
            })
            // set back
            .on( 'mouseleave', function() {
              d3.select( this )
                .transition()
                .attr("x", function(d) { return -25;})
                .attr("y", function(d) { return -25;})
                .attr("height", 50)
                .attr("width", 50);
            });
          svg.append("defs").selectAll("marker")
    .data(["suit", "licensing", "resolved"])
  .enter().append("marker")
    .attr("id", function(d) { return d; })
    .attr("viewBox", "0 -5 10 10")
    .attr("refX", 25)
    .attr("refY", 0)
    .attr("markerWidth", 6)
    .attr("markerHeight", 6)
    .attr("orient", "auto")
  .append("path")
    .attr("d", "M0,-5L10,0L0,5 L10,0 L0, -5")
    .style("stroke", "#4679BD")
    .style("opacity", "0.6");
  node.append("text")
      .attr("dx", 20)
      .attr("dy", ".35em")
      .text(function(d) { return d.name });

  force.on("tick", function() {
    link.attr("x1", function(d) { return d.source.x; })
        .attr("y1", function(d) { return d.source.y; })
        .attr("x2", function(d) { return d.target.x; })
        .attr("y2", function(d) { return d.target.y; });

    node.attr("transform", function(d) { return "translate(" + d.x + "," + d.y + ")"; });
  });
});

</script>
```

Listing 5: HTML code with d3 to get force directed graph showing gender homophily

2.4 Results


```
[{"gen_data": "{\"name\":\"erika\",\"gender\":\"female\",\"probability\":\"0.99\",\"count\":1544}"},
{"gen_data": "{\"name\":\"varun\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":129}"},
{"gen_data": "{\"name\":\"Naina\",\"gender\":\"female\",\"probability\":\"0.96\",\"count\":27}"},
{"gen_data": "{\"name\":\"vamshi\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":5}"},
{"gen_data": "{\"name\":\"Ravi\",\"gender\":\"male\",\"probability\":\"0.98\",\"count\":356}"},
{"gen_data": "{\"name\":\"Shivani\",\"gender\":\"female\",\"probability\":\"1.00\",\"count\":60}"},
{"gen_data": "{\"name\":\"BhavaniManthena\",\"gender\":null}"},
{"gen_data": "{\"name\":\"manoj\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":186}"},
{"gen_data": "{\"name\":\"Rithika\",\"gender\":null}"},
{"gen_data": "{\"name\":\"majetisiri\",\"gender\":null}"},
{"gen_data": "{\"name\":\"Manish\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":310}"},
{"gen_data": "{\"name\":\"Kumaraswamy\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":1}"},
{"gen_data": "{\"name\":\"Prashanth\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":28}"},
{"gen_data": "{\"name\":\"Uday\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":30}"},
{"gen_data": "{\"name\":\"Rithvik\",\"gender\":null}"},
{"gen_data": "{\"name\":\"ashwin\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":142}"},
{"gen_data": "{\"name\":\"TCAT\",\"gender\":null}"},
{"gen_data": "{\"name\":\"R.NaveenKumar\",\"gender\":null}"},
{"gen_data": "{\"name\":\"Alok\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":111}"},
{"gen_data": "{\"name\":\"pravallika.J\",\"gender\":null}"},
{"gen_data": "{\"name\":\"vamsi\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":18}"},
{"gen_data": "{\"name\":\"maithri\",\"gender\":null}"},
{"gen_data": "{\"name\":\"KlaraAdrinson\",\"gender\":null}"},
{"gen_data": "{\"name\":\"DeonneLivingsto\",\"gender\":null}"},
{"gen_data": "{\"name\":\"LatrinaMcRattig\",\"gender\":null}"},
{"gen_data": "{\"name\":\"MommyOddenino\",\"gender\":null}"},
{"gen_data": "{\"name\":\"Nikilesh\",\"gender\":null}"},
{"gen_data": "{\"name\":\"Dinesh\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":154}"},
{"gen_data": "{\"name\":\"Joseph\",\"gender\":\"male\",\"probability\":\"0.99\",\"count\":2213}"},
{"gen_data": "{\"name\":\"Mohan\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":103}"},
{"gen_data": "{\"name\":\"Vasavi\",\"gender\":\"female\",\"probability\":\"1.00\",\"count\":2}"},
{"gen_data": "{\"name\":\"vineeth\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":20}"},
{"gen_data": "{\"name\":\"harish\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":79}"},
{"gen_data": "{\"name\":\"y.pranay\",\"gender\":null}"},
{"gen_data": "{\"name\":\"Sandeep\",\"gender\":\"male\",\"probability\":\"0.89\",\"count\":272}"},
{"gen_data": "{\"name\":\"vani\",\"gender\":\"female\",\"probability\":\"1.00\",\"count\":35}"},
{"gen_data": "{\"name\":\"madhukar\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":6}"},
{"gen_data": "{\"name\":\"Harish\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":79}"},
{"gen_data": "{\"name\":\"bharath\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":34}"},
{"gen_data": "{\"name\":\"sunny\",\"gender\":\"male\",\"probability\":\"0.60\",\"count\":391}"},
{"gen_data": "{\"name\":\"Rajitha\",\"gender\":\"female\",\"probability\":\"0.73\",\"count\":11}"},
{"gen_data": "{\"name\":\"bhanu\",\"gender\":\"male\",\"probability\":\"0.75\",\"count\":32}"},
{"gen_data": "{\"name\":\"@mrut\",\"gender\":null}"},
{"gen_data": "{\"name\":\"Akkineni\",\"gender\":\"male\",\"probability\":\"1.00\",\"count\":1}"},
{"gen_data": "{\"name\":\"Vaishampayan\",\"gender\":null}"},
{"gen_data": "{\"name\":\"Apoorva\",\"gender\":\"female\",\"probability\":\"0.80\",\"count\":10}"},
{"gen_data": "{\"name\":\"paladhi\",\"gender\":null}"},
{"gen_data": "{\"name\":\"aruna\",\"gender\":\"female\",\"probability\":\"0.65\",\"count\":31}"},
{"gen_data": "{\"name\":\"telugupopular\",\"gender\":null}"},
{"gen_data": "{\"name\":\"satvikgadam\",\"gender\":null}"},
```


Figure 6: Sample list of generated genders for each follower

Final json


```
{
 "nodes": [
{"id": 0,"gender":"male","name": "dinesh"},
{"id": 1,"gender":"female","name": "erika siregar"},
{"id": 2,"gender":"male","name": "varun  reddy dodda"},
{"id": 3,"gender":"female","name": "Naina Sai Tipparti"},
{"id": 4,"gender":"male","name": "vamshi"},
{"id": 5,"gender":"male","name": "Ravi Teja"},
{"id": 6,"gender":"female","name": "Shivani Bimavarapu"},
{"id": 7,"gender":"male","name": "manoj Kompalli"},
{"id": 8,"gender":"male","name": "Manish Kumar"},
{"id": 9,"gender":"male","name": "Kumaraswamy S"},
{"id": 10,"gender":"male","name": "Prashanth Pappu"},
{"id": 11,"gender":"male","name": "Uday kumar reddy cs"},
{"id": 12,"gender":"male","name": "ashwin srinivas"},
{"id": 13,"gender":"male","name": "Alok Rajasukumaran"},
{"id": 14,"gender":"male","name": "vamsi krishna"},
{"id": 15,"gender":"male","name": "Dinesh Kumar"},
{"id": 16,"gender":"male","name": "Joseph Udith Raj"},
{"id": 17,"gender":"male","name": "Mohan Chandran. P"},
{"id": 18,"gender":"female","name": "Vasavi MIC"},
{"id": 19,"gender":"male","name": "vineeth chandra"},
{"id": 20,"gender":"male","name": "harish kumar"},
{"id": 21,"gender":"male","name": "Sandeep Paladhi"},
{"id": 22,"gender":"female","name": "vani"},
{"id": 23,"gender":"male","name": "madhukar reddy"},
{"id": 24,"gender":"male","name": "Harish kumar"},
{"id": 25,"gender":"male","name": "bharath motha"},
{"id": 26,"gender":"male","name": "sunny"},
{"id": 27,"gender":"female","name": "Rajitha"},
{"id": 28,"gender":"male","name": "bhanu pradeep"},
{"id": 29,"gender":"male","name": "Akkineni Akshay"},
{"id": 30,"gender":"female","name": "Apoorva Dasari"},
{"id": 31,"gender":"female","name": "aruna"},
{"id": 32,"gender":"female","name": "sudha"},
{"id": 33,"gender":"male","name": "Sandeep"},
{"id": 34,"gender":"female","name": "Ramya Mattey"},
{"id": 35,"gender":"male","name": "vineeth"},
{"id": 36,"gender":"male","name": "Isaac Samuel"},
{"id": 37,"gender":"male","name": "santosh"},
{"id": 38,"gender":"female","name": "Addie Ayala"},
{"id": 39,"gender":"male","name": "Maruthi Sharma"},
{"id": 40,"gender":"male","name": "Abhijeet Sai Dasari"}
],
"links": [
{"source": 1, "target": 0},
{"source": 2, "target": 0},
{"source": 3, "target": 0},
{"source": 4, "target": 0},
{"source": 5, "target": 0},
```


Figure 7: Sample final json file with nodes as followers and links as friendship between them along with their gender

D3 Graph

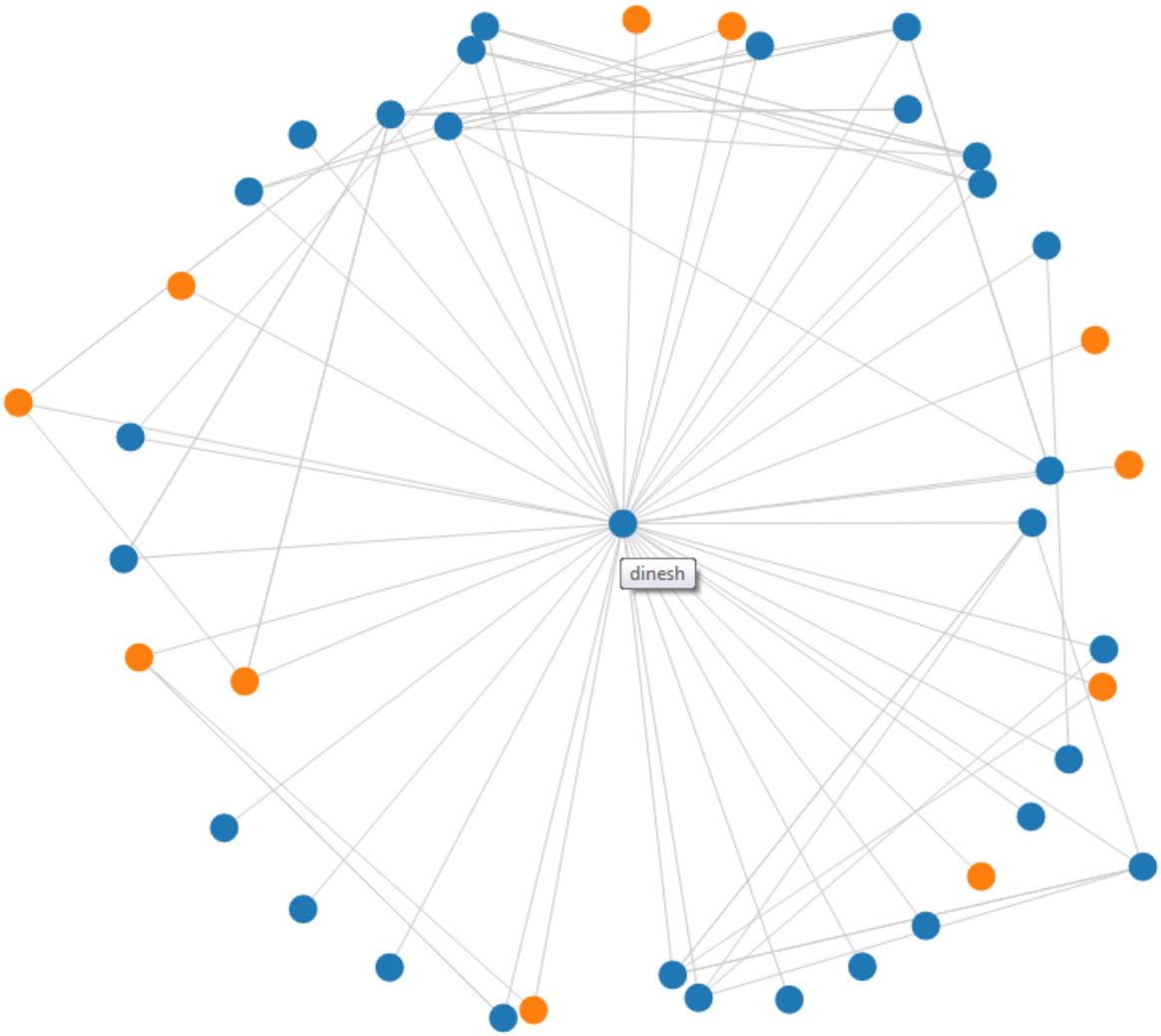

Figure 8: D3 Graph with gender homophily showing blue color nodes as “Men” and orange color nodes as “Female”

3 Problem 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 beween
connected and disconnected.
```

3.1 Solution

1. This question is similar to the previous assignment. In previous assignment I created the graph using python library and now with d3.js.
2. The first step here is to convert the GraphML file into json file so that i can give the json file as input to my Html code.
3. The sample graphMl file can be seen in figure9.
4. I wrote a python code for converting the graphMl to json and that code can be seen here in listing6.
5. I used "BeautifulSoup" in order to get the data from the GraphML and the final sample json file can be viewed here in figure10.
6. This json file is given as input to the Html code which can be seen here in listing7.
7. In order to toggle between before and after split graphs, I inserted buttons and the sample graphs can be seen here in figure11 and figure12.
8. The working model of d3 graph can be viewed in this link `http://bl.ocks.org/PaladhiDinesh/617c3ef60a692d2f972a`.

3.2 Code Listing

```
import sys
import json
from bs4 import BeautifulSoup

input_file = open('karate.GraphML','r')
output_file = open('karate.json','w')

soup_data=BeautifulSoup(input_file)
#print soup_data
elements={}
edge_elements={}
output_file.write('{\n "nodes": [\n')
for node in soup_data.find_all('node'):
        node_data=dict(node.attrs)
        #print node_data
        id=node_data['id']
        id_val=int(id.strip('n'))
        #print id
        #print id_val
        faction,name = node.find_all('data')
        faction_val=faction.contents
        name_val=name.contents
        elements['id']=id_val
        elements['faction']=int(faction_val[0])
        elements['name']=str(name_val[0])
        #print count
        output_file.write(json.dumps(elements)+',\n')
        if(id_val == 33):
                output_file.write('],\n')
                break
output_file.write('"links": [\n')
for edge in soup_data.find_all('edge'):
        edge_data=dict(edge.attrs)
        edge_src=edge_data['source']
        edge_src_Val=int(edge_src.strip('n'))
        edge_target=edge_data['target']
        edge_target_Val=int(edge_target.strip('n'))
        weight = edge.find('data')
        weight_val=int(weight.contents[0])
        edge_elements['source']=edge_src_Val
        edge_elements['target']=edge_target_Val
        edge_elements['weight']=weight_val
        output_file.write(json.dumps(edge_elements)+',\n')
        if(edge_src_Val == 32):
                output_file.write(']\n}')
                break
```

Listing 6: Python Code for converting graphml to json

3.3 Code Listing

```
<!DOCTYPE html>
<meta charset="utf-8">
<style>

.node {
  stroke: #999;
  stroke-width: 2.1px;
}

.link {
  stroke: #999;
  stroke-opacity: 1.6;
}

</style>
<body>
<div id="option1">
    <input name="split1" type="button" value="Before the split" onclick="beforesplit()">

</div>

<script src="//d3js.org/d3.v3.min.js"></script>
<script>

var width = 900,
    height = 600;

var color = d3.scale.category20();

var force = d3.layout.force()
    .charge(-400)
    .linkDistance(10)
    .size([width, height]);

var svg = d3.select("body").append("svg")
    .attr("width", width)
    .attr("height", height);

d3.json("karate.json", function(error, graph) {
  if (error) throw error;

  force
      .nodes(graph.nodes)
      .links(graph.links)
      .start();

  var link = svg.selectAll(".link")
      .data(graph.links)
    .enter().append("line")
      .attr("class", "link")


  var node = svg.selectAll(".node")
      .data(graph.nodes)
    .enter().append("circle")
      .attr("class", "node")
      .attr("r", 5)

      .call(force.drag);


  node.append("title")
      .text(function(d) { return d.name; });

  force.on("tick", function() {
    link.attr("x1", function(d) { return d.source.x; })
        .attr("y1", function(d) { return d.source.y; })
        .attr("x2", function(d) { return d.target.x; })
        .attr("y2", function(d) { return d.target.y; });

```

```
    node.attr("cx", function(d) { return d.x; })
        .attr("cy", function(d) { return d.y; });
  });
});

function beforesplit()
{
d3.selectAll('.node').style("fill", function(d) { return color(); })
}
function aftersplit()
{
d3.selectAll('.node').style("fill", function(d) { return color(d.faction); })
}


</script>
<div id="option">
    <input name="split" type="button" value="After the split" onclick="aftersplit()">

</div>
```

Listing 7: HTML code with d3 to get force directed graph

3.4 Results

```
<?xml version="1.0" encoding="UTF-8"?>
<graphml xmlns="http://graphml.graphdrawing.org/xmlns"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://graphml.graphdrawing.org/xmlns
         http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd">
<!-- Created by igraph -->
  <key id="name" for="graph" attr.name="name" attr.type="string"/>
  <key id="Citation" for="graph" attr.name="Citation" attr.type="string"/>
  <key id="Author" for="graph" attr.name="Author" attr.type="string"/>
  <key id="Faction" for="node" attr.name="Faction" attr.type="double"/>
  <key id="name" for="node" attr.name="name" attr.type="string"/>
  <key id="weight" for="edge" attr.name="weight" attr.type="double"/>
  <graph id="G" edgedefault="undirected">
    <data key="name">Zachary&apos;s karate club network</data>
    <data key="Citation">Wayne W. Zachary. An Information Flow Model for Conflict and Fissior
    <data key="Author">Wayne W. Zachary</data>
    <node id="n0">
      <data key="Faction">1</data>
      <data key="name">Mr Hi</data>
    </node>
    <node id="n1">
      <data key="Faction">1</data>
      <data key="name">Actor 2</data>
    </node>
    <node id="n2">
      <data key="Faction">1</data>
      <data key="name">Actor 3</data>
    </node>
    <node id="n3">
      <data key="Faction">1</data>
      <data key="name">Actor 4</data>
    </node>
    <node id="n4">
      <data key="Faction">1</data>
      <data key="name">Actor 5</data>
    </node>
    <node id="n5">
      <data key="Faction">1</data>
      <data key="name">Actor 6</data>
    </node>
    <node id="n6">
      <data key="Faction">1</data>
      <data key="name">Actor 7</data>
    </node>
    <node id="n7">
      <data key="Faction">1</data>
      <data key="name">Actor 8</data>
    </node>
    <node id="n8">
      <data key="Faction">2</data>
```

Figure 9: Sample GraphML file for karate club

Final json


```
{
 "nodes": [
{"id": 0, "name": "Mr Hi", "faction": 1},
{"id": 1, "name": "Actor 2", "faction": 1},
{"id": 2, "name": "Actor 3", "faction": 1},
{"id": 3, "name": "Actor 4", "faction": 1},
{"id": 4, "name": "Actor 5", "faction": 1},
{"id": 5, "name": "Actor 6", "faction": 1},
{"id": 6, "name": "Actor 7", "faction": 1},
{"id": 7, "name": "Actor 8", "faction": 1},
{"id": 8, "name": "Actor 9", "faction": 2},
{"id": 9, "name": "Actor 10", "faction": 2},
{"id": 10, "name": "Actor 11", "faction": 1},
{"id": 11, "name": "Actor 12", "faction": 1},
{"id": 12, "name": "Actor 13", "faction": 1},
{"id": 13, "name": "Actor 14", "faction": 1},
{"id": 14, "name": "Actor 15", "faction": 2},
{"id": 15, "name": "Actor 16", "faction": 2},
{"id": 16, "name": "Actor 17", "faction": 1},
{"id": 17, "name": "Actor 18", "faction": 1},
{"id": 18, "name": "Actor 19", "faction": 2},
{"id": 19, "name": "Actor 20", "faction": 1},
{"id": 20, "name": "Actor 21", "faction": 2},
{"id": 21, "name": "Actor 22", "faction": 1},
{"id": 22, "name": "Actor 23", "faction": 2},
{"id": 23, "name": "Actor 24", "faction": 2},
{"id": 24, "name": "Actor 25", "faction": 2},
{"id": 25, "name": "Actor 26", "faction": 2},
{"id": 26, "name": "Actor 27", "faction": 2},
{"id": 27, "name": "Actor 28", "faction": 2},
{"id": 28, "name": "Actor 29", "faction": 2},
{"id": 29, "name": "Actor 30", "faction": 2},
{"id": 30, "name": "Actor 31", "faction": 2},
{"id": 31, "name": "Actor 32", "faction": 2},
{"id": 32, "name": "Actor 33", "faction": 2},
{"id": 33, "name": "John A", "faction": 2}
],
"links": [
{"source": 0, "target": 1, "weight": 4},
{"source": 0, "target": 2, "weight": 5},
{"source": 0, "target": 3, "weight": 3},
{"source": 0, "target": 4, "weight": 3},
{"source": 0, "target": 5, "weight": 3},
{"source": 0, "target": 6, "weight": 3},
{"source": 0, "target": 7, "weight": 2},
{"source": 0, "target": 8, "weight": 2},
{"source": 0, "target": 10, "weight": 2},
{"source": 0, "target": 11, "weight": 3},
{"source": 0, "target": 12, "weight": 1},
{"source": 0, "target": 13, "weight": 3},
```


Figure 10: Sample Final json

D3 Graph

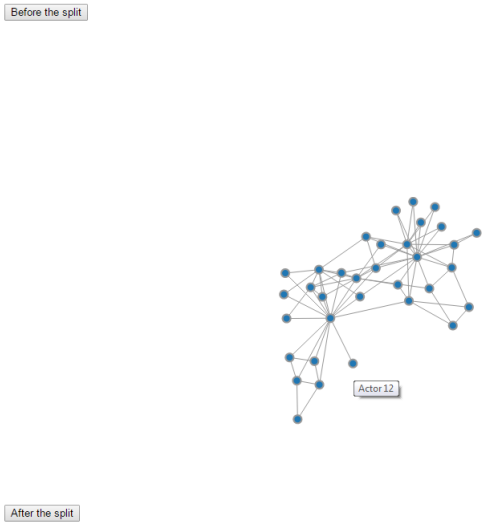

Figure 11: D3 Graph before split

D3 Graph

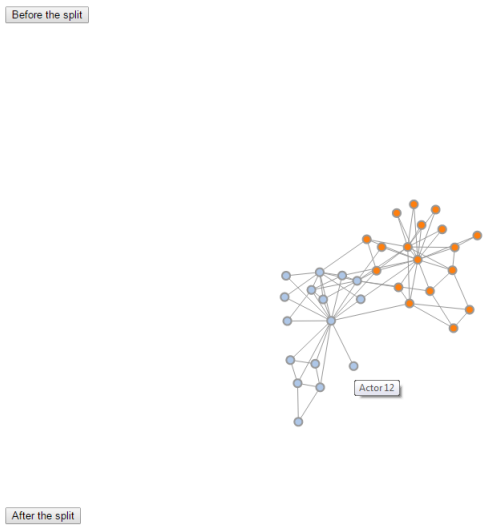

Figure 12: D3 Graph after split

Bibliography

[1] TweepError :https://github.com/tweepy/tweepy/issues/191, AlexeyProskuryakov, 2012

[2] Parsing values from a JSON file in Python :http://stackoverflow.com/questions/2835559/parsing-values-from-a-json-file-in-python, Justin Peel, 2010

[3] d3.js Error :http://stackoverflow.com/questions/11357974/d3-js-cannot-read-property-weight-of-undefined ZachB, 2012

[4] GET friendships/show :https://dev.twitter.com/rest/reference/get/friendships/show, Twitter, 2015

[5] Stack Overflow :http://stackoverflow.com/questions/6551446/can-i-run-html-files-directly-from-github-ir chmanie, 2013

[6] Determine the gender of a first name :https://genderize.io/, Stroemgren, 2013

[7] Force-Directed Graph :https://bl.ocks.org/mbostock/4062045, Mike Bostock, 2016

[8] Labeled Force Layout :https://bl.ocks.org/mbostock/950642, Mike Bostock, 2016