CS-432/532 Introduction to Web Science:
Assignment #5:
The "Karate Club (Zachary, 1977)"

Due on Thursday, March 3, 2016

Dr. Michael L. Nelson

Plinio Vargas
pvargas@cs.odu.edu

Contents

List of Figures

Listings

List of Tables

Problem 1

We know the result of the Karate Club (Zachary, 1977) split. Prove or disprove that the result of split could have been predicted by the weighted graph of social interactions. How well does the mathematical model represent reality?

Generously document your answer with all supporting equations, code, graphs, arguments, etc.

1.1 Approach

Social interaction among individuals can be represented as a graph, where nodes signify a particular individual of the social network, and edges between nodes represent some type of relationship between them. An individual may enter the social network influenced by a node belonging to the network or by other means. If the inclusion into the network was influenced by a node within the network, the bond between those two nodes is much stronger than if they just associated by mere interaction in the network.

Similarly, disputes and differences may cause rupture in a relationship, this could be represented as removing an edge between two nodes in the graph. This conflict may be so common among other members within a social network that a collection of edges removal could translate in splitting a group (network) into two or more separate entities. In *graph* studies, particularly undirected *graphs* (which is our case), if a node can reach other nodes within the *graph*, it is denoted as a *connected component*. We are going to use this definition to check if at any point after removing an edge the *graph*, fulfills the characteristic of a *connected component*.

Then, if the number of connected components in an original graph G is equal to one (1), we will be able to check if G has split after the removal of an edge (u, v) between nodes u and v that are part of G by testing the condition:

$$\forall n \ \in G \ where \ n \ is \ a \ node \ of \ G. \ n \ can \ traverse \ \rightarrow |G| - 1 \ nodes \tag{1}$$

We are going to be emulating the breaking of the well known and studied "Karate Club"[1]. An important concept identified in the original study[1] is how the information flowed within the network. As friction began to arise meeting were called by a faction within the club, which was not interested that rival faction received the information. However, since the network is still in one $connected - component$, all the members were able somehow get the information. The leader of a faction who usually passed the information was labeled as the *source*, and the leader of a rival faction was labeled as *sink*.

In a single split of $G = (V, E)$ we will create an identical graph $G' = (V', E')$ where:

$$sink = max(|v_1|, |v_2|, \cdots, |v_n|) \tag{2}$$

Where v is a vertex in G' and n is the number of vertices in the graph. To find the source, we make $G' = G' - sink - all \ edges(u', v') \ adjecent \ to \ sink$. Then *source* is:

$$n = G.size$$

$$source = max(|v_1|, |v_2|, \cdots, |v_n|) \tag{3}$$

Where v is a vertex in G', and n is the number of vertices in the new graph.

Equations (2) and (3) are relevant in determining edges removal. Since splitting the network consists of making invalid the condition of equation (1), we are trying to break the edges not strongly related to *source* that are an information flow bridge to *sink*.

Problem 1 continued on next page...

Although the solution on how the Karate Club [1] splits into two different groups has been very well documented, and many resources are available to simulate and predict the outcome, we need to "Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful"[2] and "Essentially, all models are wrong, but some are useful"[2]. Also the compared simulation model was not able to have 100% accuracy. Node #9 was expected to split to be color coded white, but other factors not input in the graphs weighted more than the expected result.

Borrowing from week 6's ODU Spring-16 Web Science class slide number 66, it is easy to visualize actual result. If we were to separate the network below into two different ones, we would need to cut the edges joining two nodes with different color.

Figure 1: Karate Club Split-Color-Coded Nodes

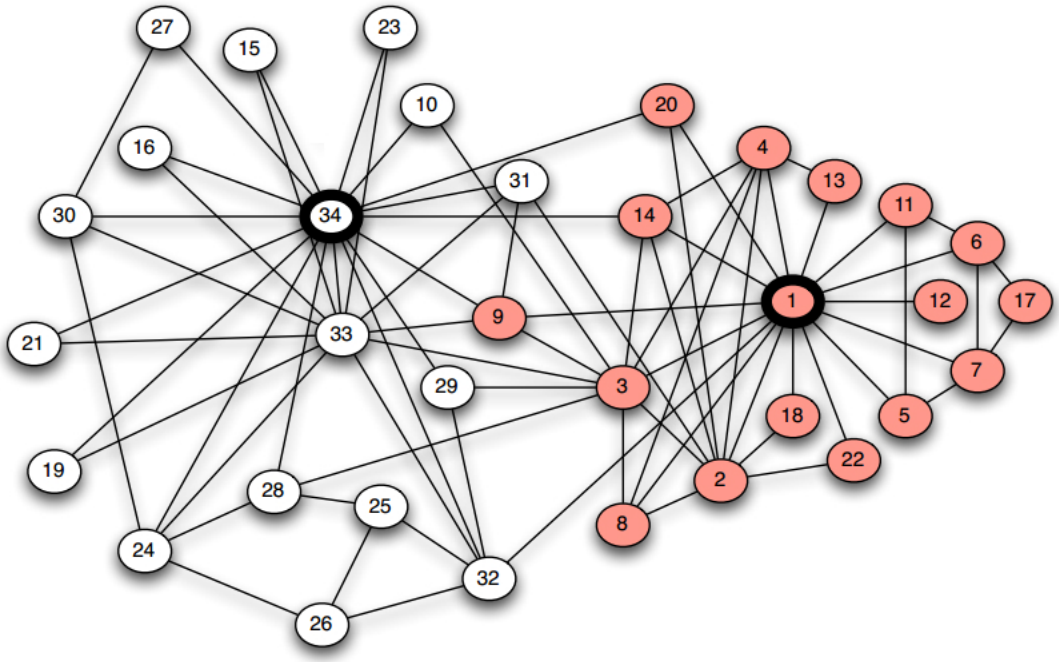

Figure 1 shows social interactions among members of the club outside the facility. The color coded nodes represent how the grouping of nodes after the club split.

Problem 1 continued on next page...

Making the split when we don't have color code node is not as simple.

Figure 2: Karate Club Homogeneous Color Nodes

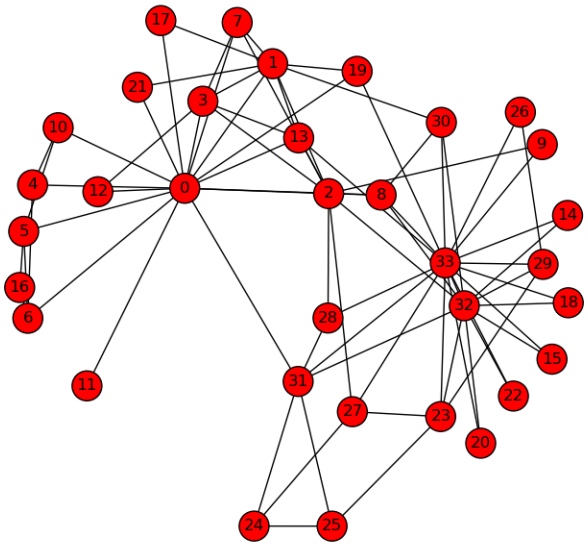

Figure 2 shows social interactions among members of the club outside the facility. Although this graph is not that complex, it is not trivial to figure out where the faction split after node colors become homogeneous. The graph was generated using *matplotlib* Python library. The code for graphic generation is included in $< DrawOriginalClub.py >$ which is attached to this document.

The Python library ***networkx*** was very useful to a create our model. Many graph's functionality, such as creating a node, adding adjacent vertices, or removing edges is easy to handle with this library. Although the data for [1] is available in ***networkx***. The data to create the graph was upload as a matrix from [3] with the purpose of manipulating $source - sink$ rows in the graph.

1.2 Solution

The Python application < *KarateClub.py* > attached to this document was implemented to create our social network splitting model.

Listing 1: Finding Source-Sink in Graph

```
72 # find sink
73 vertices = G.degree()
74 sink = [vertex for vertex, degree in vertices.items() if degree == max(vertices.
       values())][0]
75
76 # find source
77 del vertices[sink]
78 source = [vertex for vertex, degree in vertices.items() if degree == max(vertices.
       values())][0]
```

To find *sink* and *source* we can apply equations (2) and (3) respectively. First, line 73 gets all vertices in our graph G. We get the vertex with highest degree for *sink*. Finally, we delete vertex containing the sink value (line 77). The next vertex with highest degree becomes our *source*.

Figure 3: Karate Club Showing Sink-Source Nodes

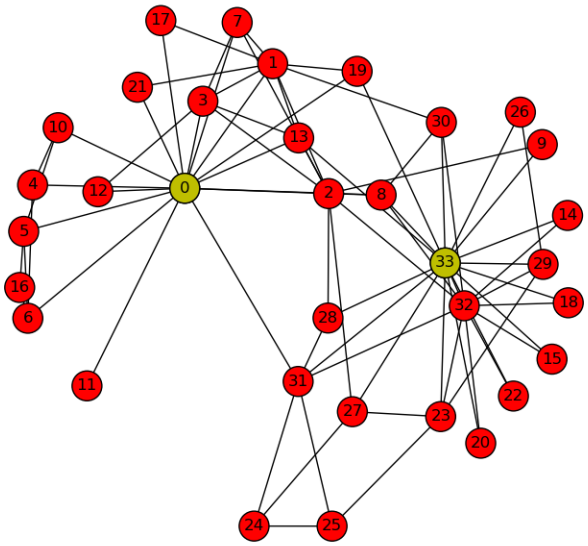

Figure 3 shows social interactions among members of the club outside the facility. It is still not trivial to figure out where the faction will split. The graph provides a much richer view than Figure 2. We can have a general ideal of the outcome by focusing on the nodes color-coded yellow for *source* (0) and *sink* (33). The graph was generated using *matplotlib* Python library. The code for graphic generation is included in < *DrawOriginalClub.py* > which is attached to this document.

A python implementation Girvan-Newman Algorithm was extracted from [4]. Minor modifications were made to adapt our python3 environment. The heart of our solution resides in the listing below:

Listing 2: Calculating Edge Betweenness for Graph G

```
80 # using Girvan-Newman"
81 init_ncomp = nx.number_connected_components(G)    #no of components
82
83 # print('init', init_ncomp)
84 ncomp = init_ncomp
85 while ncomp <= init_ncomp + 2:
86     bw = nx.edge_betweenness_centrality(G, weight='weight')    #edge betweenness
       for G
87     #find the edge with max centrality
88     max_ = max(bw.values())
89
90     #find the edge with the highest centrality and remove all of them if there is
       more than one!
91     for k, v in bw:
92         if float(bw[(k, v)]) == max_:
93             G.remove_edge(k, v)    # remove the central edge
94             print('(%d, %d)' % (k, v))
95     ncomp = nx.number_connected_components(G)    # recalculate the no of
       components
```

Line 80 initializes the number of components in G which originally is one (1). Lines 85-95 is a while-loop that verifies if equation (1) is satisfied. The number of connected components in G is modified after the edge with the highest flow has been found (line 92). Lines 91-94 iterates through edges in G and removes edges (u, v) which has the highest betweeness value. The time complexity for this algorithm is $O(|V||E|)$.

Running $< KarateClub.py >$ shows that 11 iterations to remove edges in G were required to have 2 connected-components in the graph. The specific edges removed from G are shown below:

```
Starting Time: Wed, Mar 02, 2016 at 19:39:14
(0, 31)
(0, 2)
(0, 8)
(13, 33)
(19, 33)
(2, 32)
(1, 30)
(1, 2)
(2, 3)
(2, 13)
(2, 7)

End Time: Wed, Mar 02, 2016 at 19:39:14
Execution Time: 0.26 seconds
```

Drawing the Graph after creating more than one connected component is represented below:

Problem 1 continued on next page...

Figure 4: Karate Club Split Graph Projection

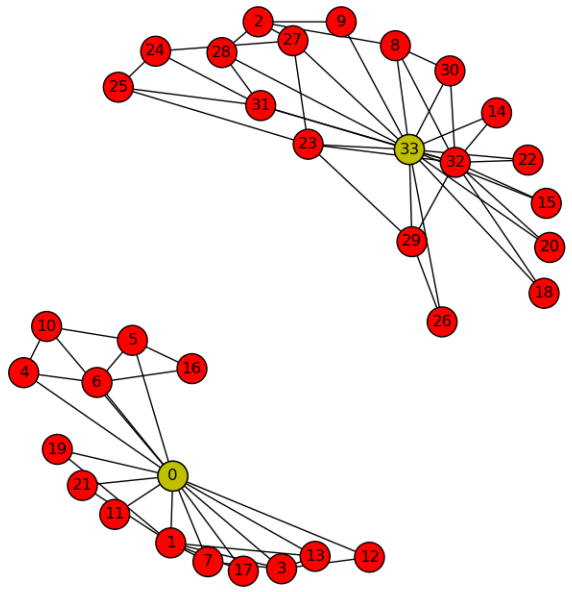

Figure 4 shows the projected Karate Club separation. Since we already know the results from Figure 1, to calculate how accurate our projection is, we can count the number of accurate vertices for both groups and divide that number into the total number of vertices in the graph.

Node comparison has to be defined since Figure 1 starts with node 1, while we start with node 0. Thus, when comparing with Figure 1, we will be adding 1 to the compared node, but the listing of node discrepancy will be based on nodes from Figure 4.

1.3 Comparing Results

Comparing Figure 1 and Figure 4 yields interesting results. The *source* faction properly projected the following nodes:

{0,1,3,4,5,6,7,10,11,12,13,16,18,19,21}

Only two (2) nodes were mis-projected resulting in a confidence level of 32/34 or 94%. Using weighted adjacent matrix data in Table 2 with the same algorithm, yielded the same result.

Problem 2 - Extra Credit

We know the group split in two different groups. Suppose the disagreements in the group were more nuanced -- what would the clubs look like if they split into groups of 3, 4, and 5?

2.1 Approach

We are going to use the same approach as in 1.1, however in line 85 we will increase the number of connected components by one, depending on the desire output. For example, if we desire 3 connected component result, we may modify line 85 as follows:

```
85 while ncomp <= init_ncomp + 1:
```

2.2 Karate Club 3-Split Solution

The following edges were removed to obtain 3 separate connected components from the original graph:

```
Starting Time: Thu, Mar 03, 2016 at 19:35:20
(0, 31)
(0, 2)
(0, 8)
(13, 33)
(19, 33)
(2, 32)
(1, 30)
(1, 2)
(2, 3)
(2, 13)
(2, 7)
(9, 33)
(27, 33)
(2, 9)

End Time: Thu, Mar 03, 2016 at 19:35:21
Execution Time: 0.29 seconds
```

Figure 5: Karate Club 3-Split Graph Projection

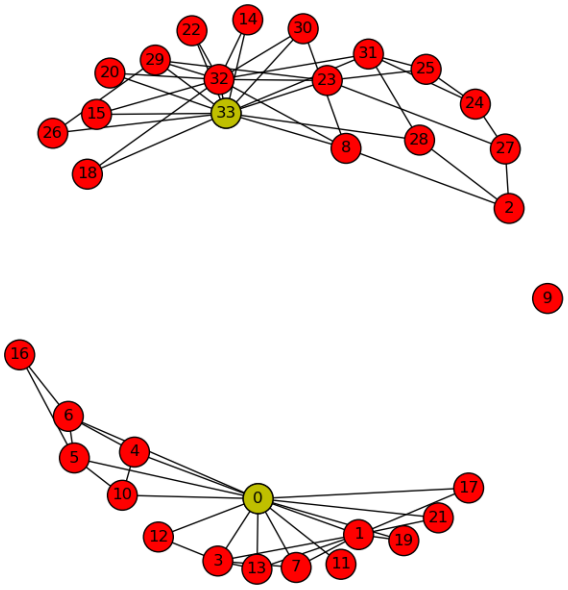

Figure 5 shows the projected Karate Club separation in three (3) different groups. Only vertex 9 was separated from the first split, and it took three extra edges removal to make node 9 and independent component.

2.3 Karate Club 4-Split Solution

The following edges were removed to obtain 4 separate connected components from the original graph:

```
Starting Time: Thu, Mar 03, 2016 at 19:40:20
(0, 31)
(0, 2)
(0, 8)
(13, 33)
(19, 33)
(2, 32)
(1, 30)
(1, 2)
(2, 3)
(2, 13)
(2, 7)
(9, 33)
(27, 33)
(2, 9)
(0, 6)
(0, 5)
(0, 4)
(0, 10)
```

```
End Time: Thu, Mar 03, 2016 at 19:40:20
Execution Time: 0.32 secondss
```

Figure 6: Karate Club 4-Split Graph Projection

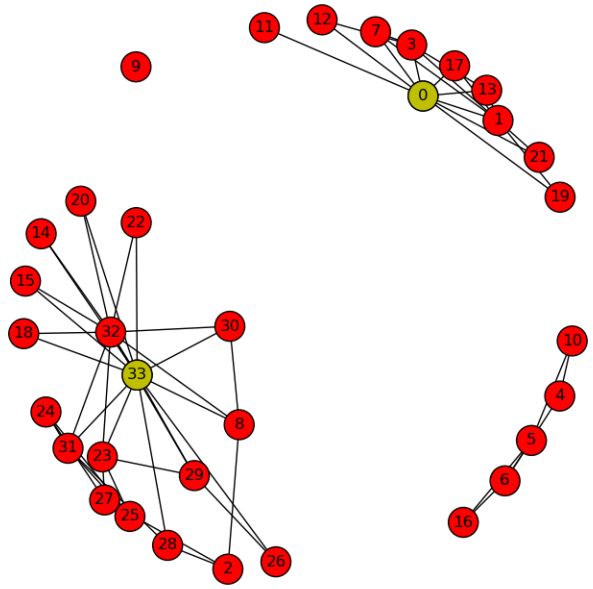

Figure 6 shows the projected Karate Club separation in four (4) different groups. Overall, 18 edges were removed to create a four-way split.

2.4 Karate Club 5-Split Solution

The following edges were removed to obtain 5 separate connected components from the original graph:

```
Starting Time: Thu, Mar 03, 2016 at 19:42:30
(0, 31)
(0, 2)
(0, 8)
(13, 33)
(19, 33)
(2, 32)
(1, 30)
(1, 2)
(2, 3)
(2, 13)
(2, 7)
(9, 33)
(27, 33)
(2, 9)
(0, 6)
```

```
(0, 5)
(0, 4)
(0, 10)
(31, 33)
(31, 32)
(28, 33)
(23, 25)
(23, 27)
(2, 8)

End Time: Thu, Mar 03, 2016 at 19:42:31
Execution Time: 0.37 seconds
```

Figure 7: Karate Club 5-Split Graph Projection

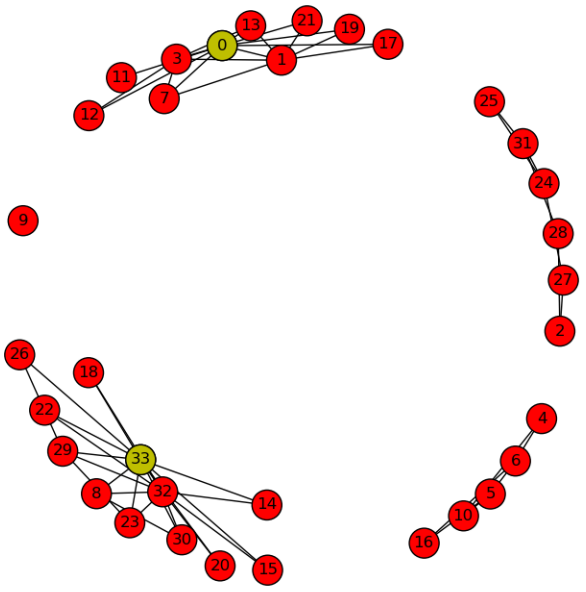

Figure 7 shows the projected Karate Club separation in five (5) different groups. Overall, 24 edges were removed to create a five-way split.

Table 1: Unweighted Karate Club Adjecent Matrix

row/col	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
0	0	1	1	1	1	1	1	1	1	0	1	1	1	1	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0
1	1	0	1	1	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0
2	1	1	0	1	0	0	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0
3	1	1	1	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1
9	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
10	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
16	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
19	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
21	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	1	1
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
27	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1
28	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	1
30	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
31	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	1	1	1
32	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	1	0	1	0	1	1	0	0	0	0	0	1	1	1	0	1
33	0	0	0	0	0	0	0	0	1	1	0	0	0	1	1	1	0	0	1	1	1	0	1	1	0	0	1	1	1	1	1	1	1	0

Adjacent matrix was used to generate graphs in Figure 2 and Figure 3. The data was obtained from `http://vlado.fmf.uni-lj.si/pub/networks/data/ucinet/zachary.dat` [3].

Table 2: Weighted Karate Club Adjecent Matrix

row/col	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
0	0	4	5	3	3	3	3	2	2	0	2	3	1	3	0	0	0	2	0	2	0	2	0	0	0	0	0	0	0	0	2	0	0
1	4	0	6	3	0	0	0	4	0	0	0	0	0	5	0	0	0	1	0	2	0	2	0	0	0	0	0	0	0	0	2	0	0	0
2	5	6	0	3	0	0	0	4	5	1	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	0	0	0	2	0
3	3	3	3	0	0	0	0	3	0	0	0	0	3	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	3	0	0	0	0	0	2	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	0	0	0	0	0	5	0	0	0	3	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	3	0	0	0	2	5	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	2	4	4	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	2	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	3	4
9	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
10	2	0	0	0	3	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	1	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	3	5	3	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	2
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	4
16	0	0	0	0	0	3	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2
19	2	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	1
21	2	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	3
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	4	0	3	0	0	5	4
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	3	0	0	0	2	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	2	0	0	0	0	0	0	7	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	2
27	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	3	0	0	0	0	0	0	0	0	4
28	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	2
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	4	0	0	0	0	0	4	2
30	0	2	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	3
31	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	7	0	0	2	0	0	0	4	4
32	0	0	2	0	0	0	0	0	3	0	0	0	0	0	3	3	0	0	1	0	3	0	2	5	0	0	0	0	0	4	3	4	0	5
33	0	0	0	0	0	0	0	0	4	2	0	0	0	3	2	4	0	0	2	1	1	0	3	4	0	0	2	4	2	2	3	4	5	0

Adjacent matrix was used to generate graphs in Figure 2 and Figure 3. The data was obtained from `http://vlado.fmf.uni-lj.si/pub/networks/data/ucinet/zachary.dat` [3].

References

[1] Karate Club (Zachary, 1977). (n.d.) Retrieved February 27, 2016, from http://aris.ss.uci.edu/~lin/76.pdft

[2] Box, G. E. P., and Draper, N. R., (1987), *Empirical Model Building and Response Surfaces*p.74, p.424, John Wiley & Sons, New York, NY.

[3] Karate Club Data Set. (n.d.) Retrieved March 2, 2016, from http://vlado.fmf.uni-lj.si/pub/networks/data/ucinet/zachary.dat/

[4] Python Girvan-Newman Algorithm. (n.d.) Retrieved March 2, 2016, from https://github.com/kjahan/community/blob/master/cmty.pyl/