Yajie Wang

Miles Abel

7/29/2019

Project 2

1. Facebook Network

For the Facebook data, we create our network model from the edgelist file *facebook_combined.txt*.

1. Structural properties of the Facebook network

Question 1.1

To import the data from the edgelist file, we use read.table() to generate a data frame and then use graph.data.frame() to produce an igraph from this data frame. Figure 1 displays the graph. With the igraph generated, we use vcount() and ecount() to calculate the number of nodes and edges of the network.

$\text{Nodes} = 4{,}039$

$\text{Edges} = 88{,}234$

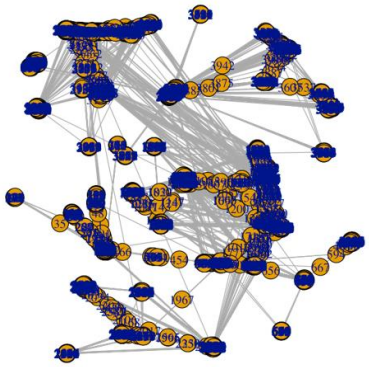

Figure 1: Igraph of Facebook network

Question 1.2

The network is connected; we verify this by using is.connected() on the network.

Question 2

Because the network is connected, we find the diameter with diameter().

$\text{diameter}(g) = 8$

Question 3

We plot the degree distribution using plot(degree_distribution(g)) and find that the frequency of nodes with low degree occurs more often than nodes with high degree. The frequency of node degree decays exponentially as seen in Figure 2. The average degree is found by mean(degree(g)), and is

$$\text{Mean Degree} = 43.69101.$$

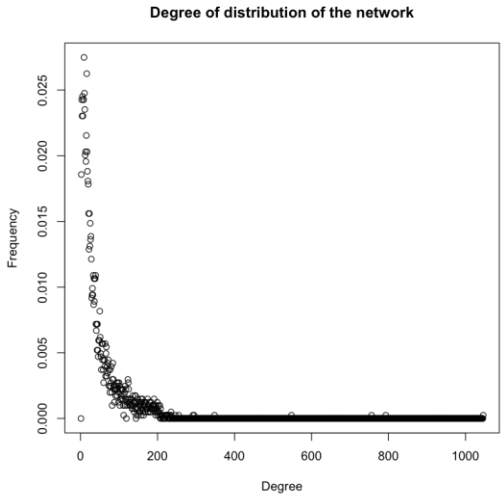

Figure 2: Degree distribution of the Facebook network

Question 4

Comparatively, we plot the degree distribution of Question 3 in log-log scale and fit a linear regression line to this plot, calculating the slope. Aside from the endpoints, the log-log degree distribution exhibits a strong linear relationship. The slope is as follows:

$$\text{Slope} = -1.2475.$$

The slope is lower in magnitude than what is found with power law trending Barabasi-Albert preferential (BA) attachment networks. For BA networks the slope was around -3.

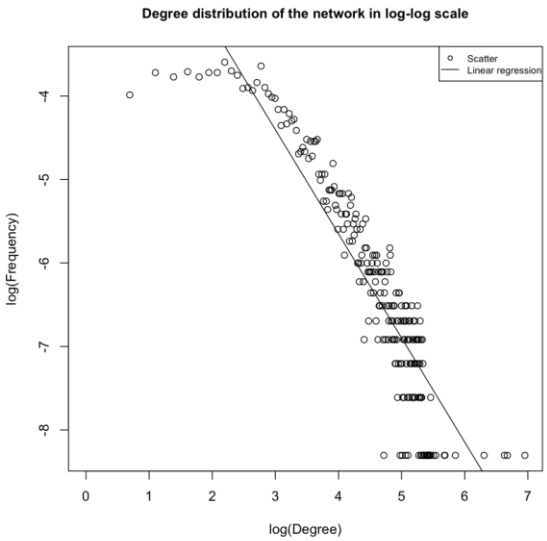

Figure 3: Facebook network degree distribution in log-log scale

2. Personalized network

Question 5

To create a personalized network of the user with ID = 1, we use make_ego_graph() on the Facebook network, g, and set the parameter 'nodes = V(g)[name==0]' inside of make_ego_graph(). We set 'name==0' because node ID = 1 corresponds to node ID = 0 in the edgelist. We find the number of nodes and edges in this personalized network as

Nodes = 348

Edges = 2,866.

From the plot in Figure 4, we see that a definitive relationship between the nodes seems to exist. That is, we see that node 0 is in the center and that all the other nodes seem to connect back to this center node. In fact, they do as node 0 is the ego node, and all the remaining nodes are friends of this ego node. Thus, for a personalized Facebook network, we have a user, u, and all of his friends, which are all at least connected back to the ego node, u.

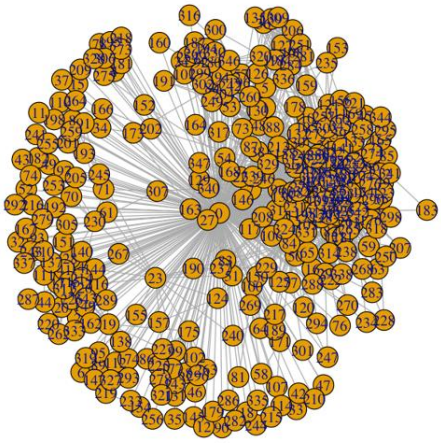

Figure 4: Personalized network with user ID = 1

Question 6

For the personalized network of user ID = 1, the diameter is

diameter (g1[[1]]) = 2.

The trivial upper bound on the diameter is 2, and the trivial lower bound is 1.

Question 7

When the diameter is equal to the upper bound of 2, that means that within the personalized network not all of the friends of the ego node know each other. Put another way, all of the friends in the personalized network know the ego node but all of these friends are not connected.

Contrastingly, when the diameter is equal to the lower bound of 1, that means that all of the friends within the personalized network know each other. That is all of the nodes are connected to each other in the entire network.

3. Core node's personalized network

The definition of a core node will be nodes that have more than 200 neighbors.

Question 8

To determine the number of core nodes that exist in the Facebook network, we use ego_size() on the original Facebook network. We find

Number of core nodes = 41

Average degree of the core nodes = 277.43902.

3.1 Community structure of core node's personalized network

There are five core node ID's for which we generate personalized networks and perform several analytic analyses. The node ID's are: 1, 108, 349, 484, and 1087.

Question 9

For all five node ID's mentioned in the paragraph above, we have displayed the modularity scores using each of the three algorithms: cluster_fast_greedy, cluster_edge_betweeness, and cluster_infomap.

Node ID	Clustering Method	Modularity
1	cluster_fast_greedy	0.41310
	cluster_edge_betweeness	0.35330
	cluster_infomap	0.38912
108	cluster_fast_greedy	0.43593
	cluster_edge_betweeness	0.50676
	cluster_infomap	0.50822
349	cluster_fast_greedy	0.25021
	cluster_edge_betweeness	0.13353
	cluster_infomap	0.09603
484	cluster_fast_greedy	0.50700
	cluster_edge_betweeness	0.48910
	cluster_infomap	0.51564
1087	cluster_fast_greedy	0.14553
	cluster_edge_betweeness	0.02762
	cluster_infomap	0.02691

Table 1: Modularity scores for five node ID's personalized networks, using three different clustering algorithms

Cluster_fast_greedy() tries to find dense subgraphs, called communities by directly optimizing a modularity score

Cluster_edge_betweeness() uses an edge betweenness score for each edge, based on the number of shortest paths through each edge. This model uses the concept that edges connecting separate modules have high edge betweenness. Thus, if we gradually remove the edge with the highest edge betweenness score we will get a hierarchical map, a rooted tree, called a dendrogram of the graph. The leaves represent the individual vertices and the root of the tree represents the whole graph

Cluster_infomap() defines community structure based on minimizing the expected description length of a random walk trajectory

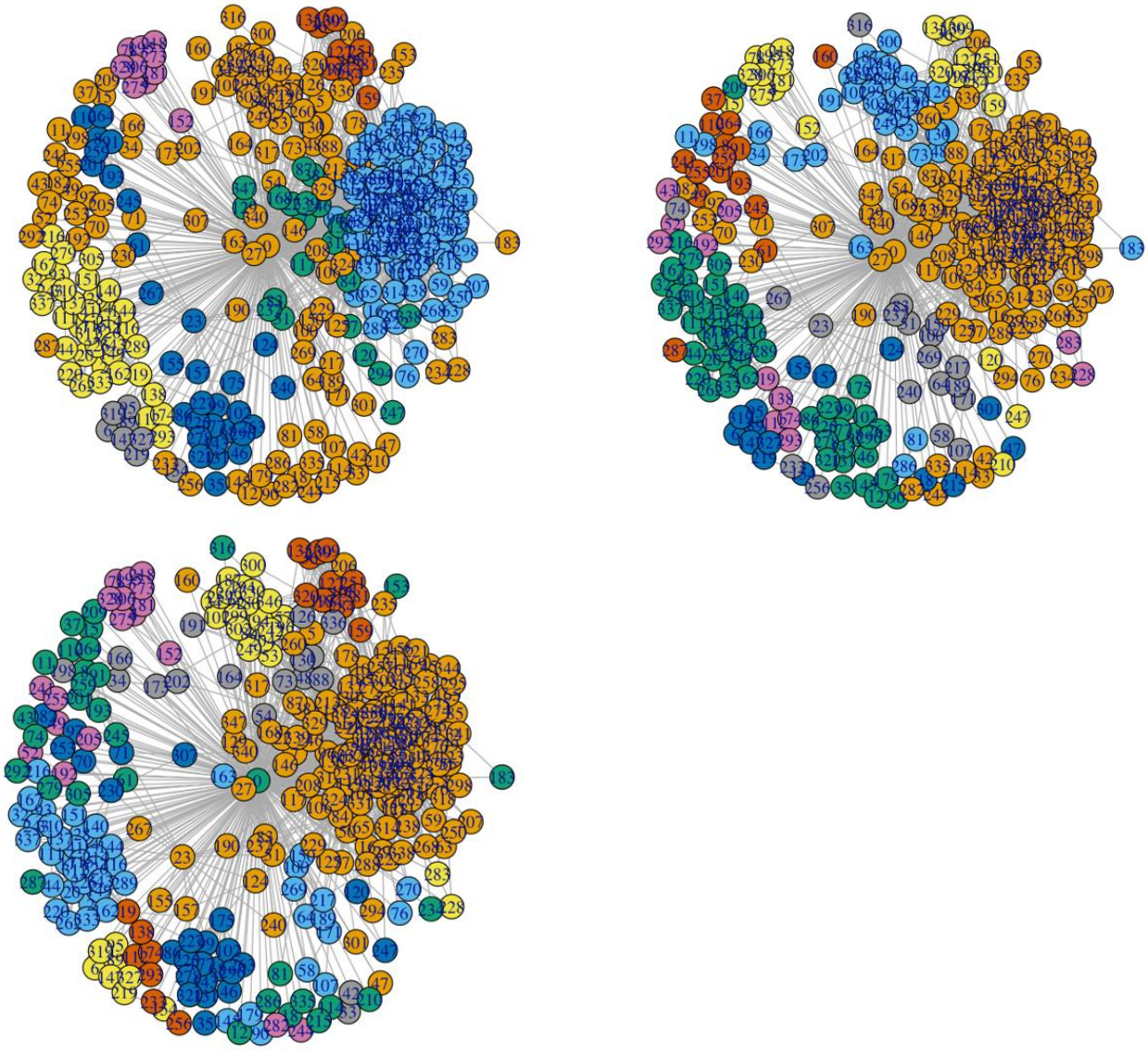

Figure 5: Node ID = 1 with cluster_fast_greedy, cluster_edge_betweeness, and cluster_infomap (starting in the upper left and moving from left to right)

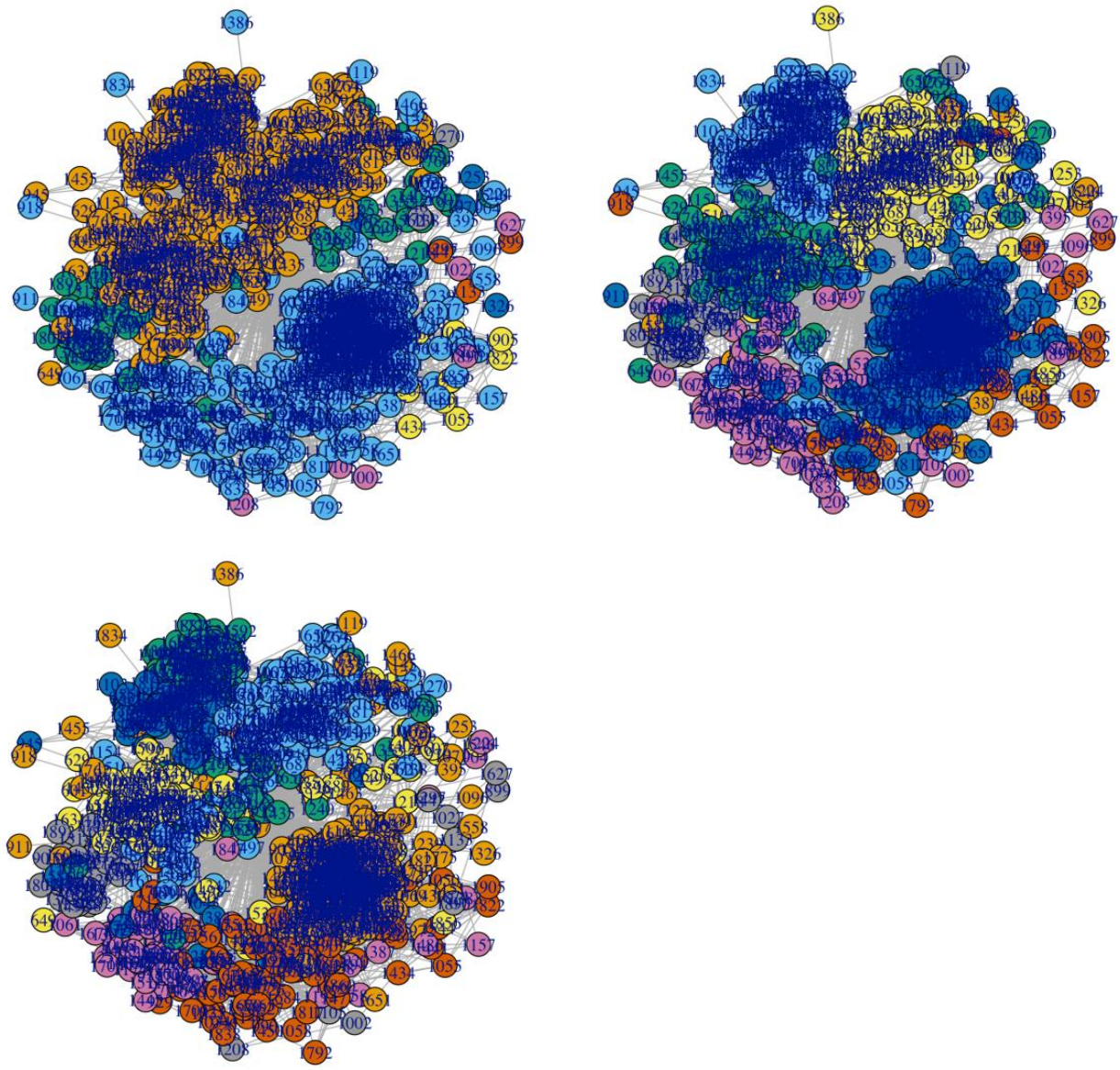

Figure 6: Node ID = 108 with cluster_fast_greedy, cluster_edge_betweeness, and cluster_infomap (starting in the upper left and moving from left to right)

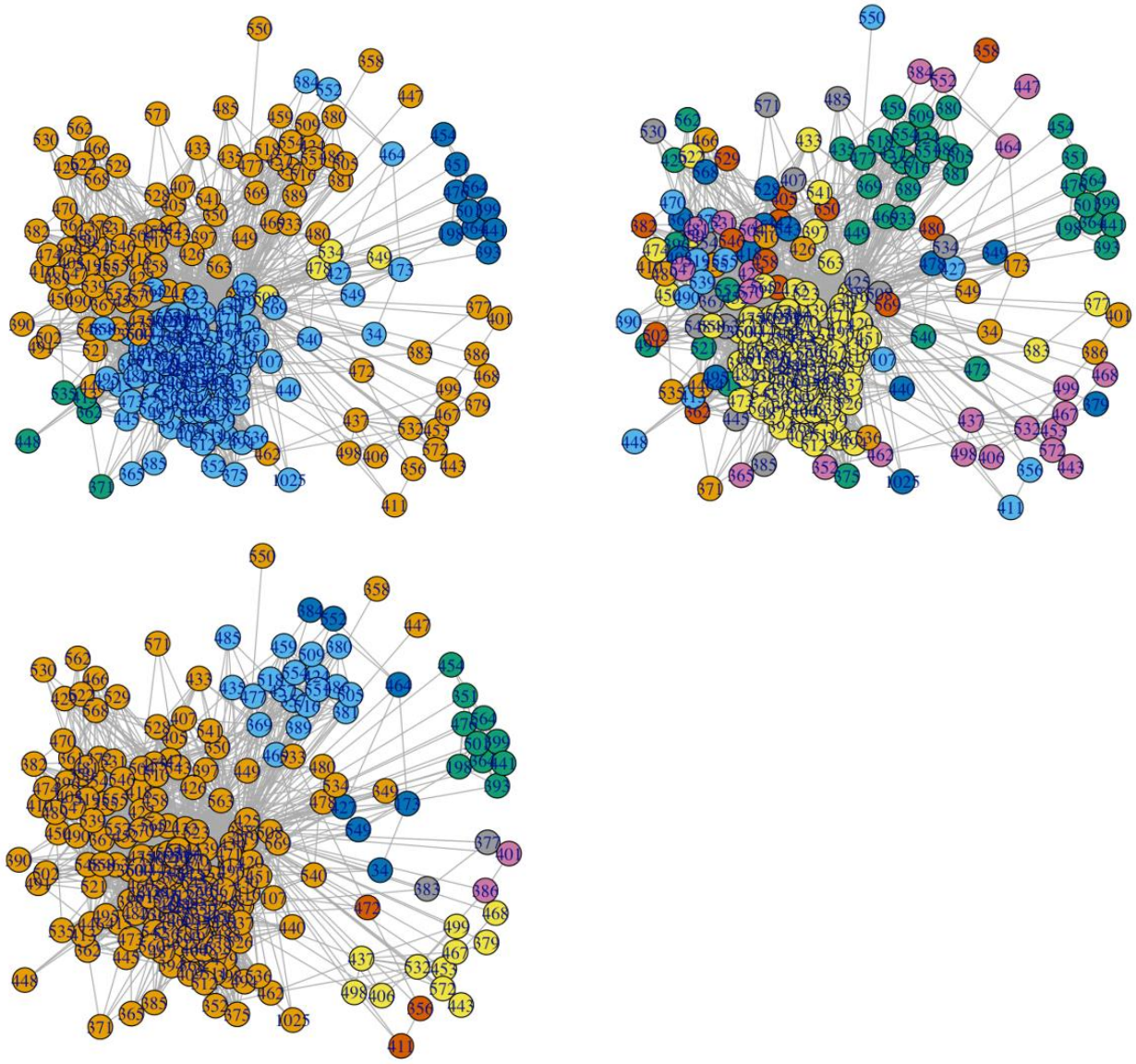

Figure 7: Node ID = 349 with cluster_fast_greedy, cluster_edge_betweeness, and cluster_infomap (starting in the upper left and moving from left to right)

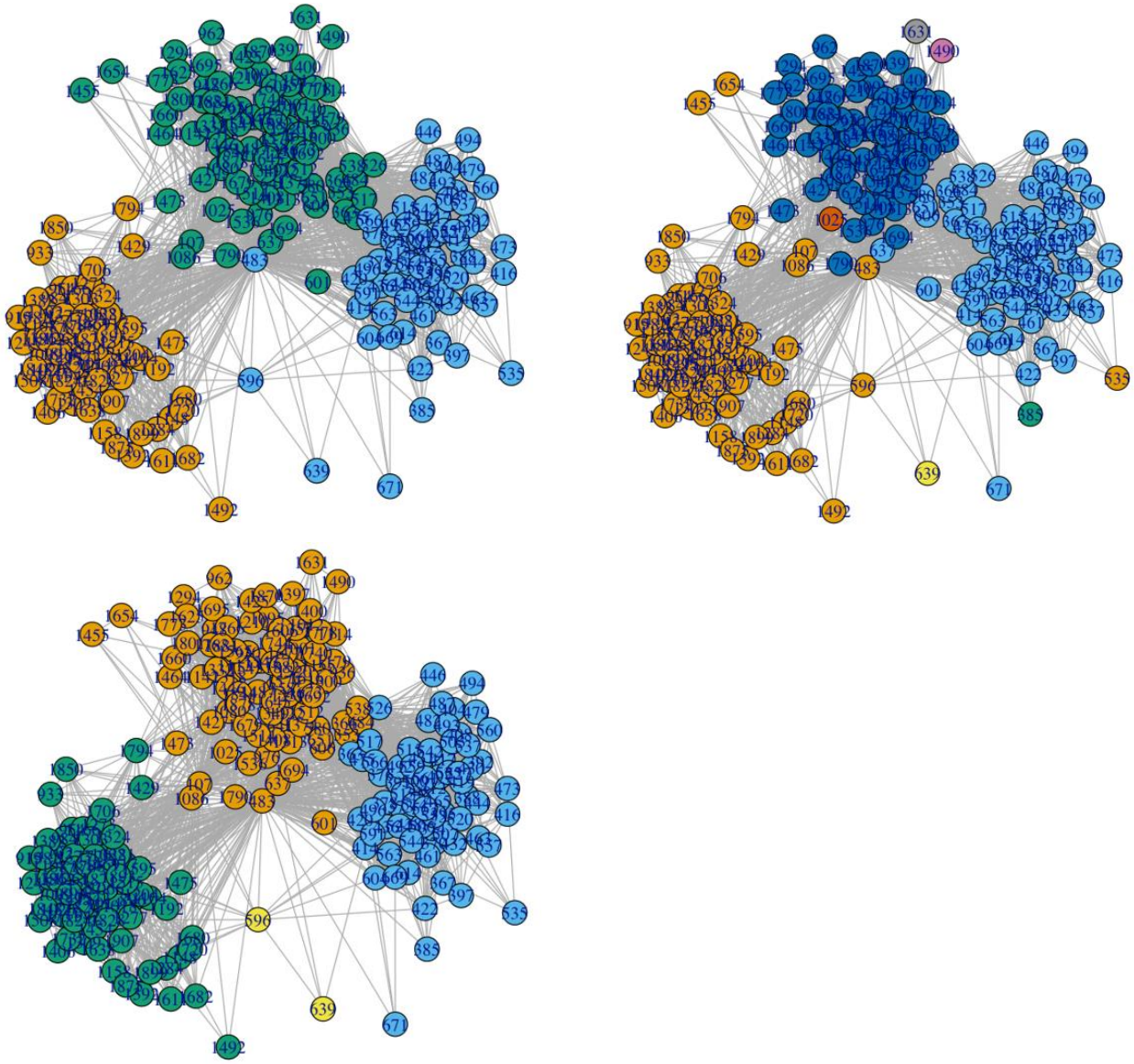

Figure 8: Node ID = 484 with cluster_fast_greedy, cluster_edge_betweeness, and cluster_infomap (starting in the upper left and moving from left to right)

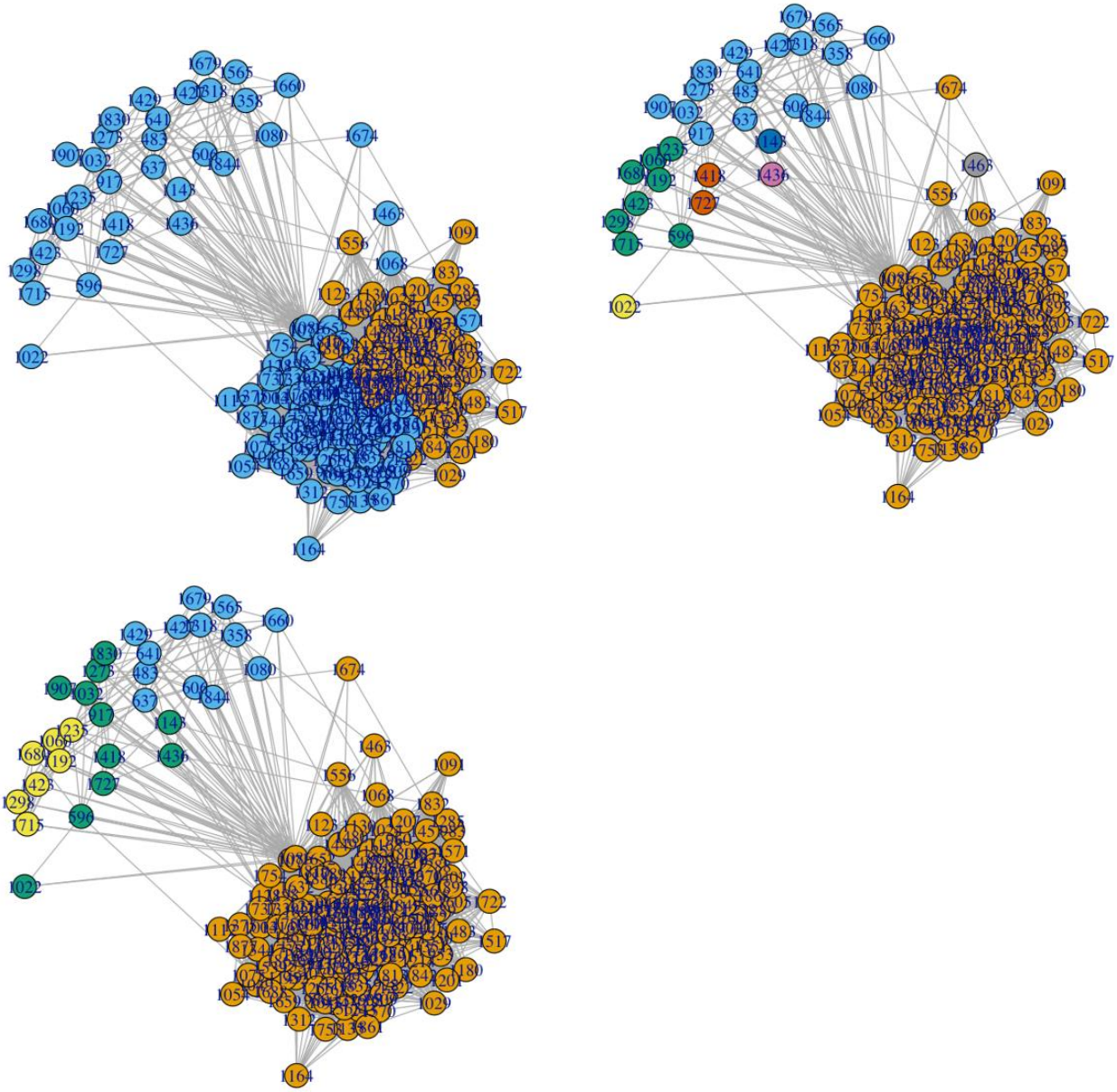

Figure 9: Node ID = 1087 with cluster_fast_greedy, cluster_edge_betweeness, and cluster_infomap (starting in the upper left and moving from left to right)

3.2 Community structure with the core node removed

Question 10

We remove the core node from the personalized networks of Question 9 and perform the same analyses as in Question 9. The modularity scores of the personalized networks for the five node ID's are listed in Table 2. We find that the modularity scores are higher with the core nodes removed. This is because the sub-group is more isolated when there is no bridge to connect the core node with each group.

Node ID	Clustering Method	Modularity
1	cluster_fast_greedy	0.44185
	cluster_edge_betweeness	0.41615
	cluster_infomap	0.41801
108	cluster_fast_greedy	0.45806
	cluster_edge_betweeness	0.52132
	cluster_infomap	0.52052
349	cluster_fast_greedy	0.24569
	cluster_edge_betweeness	0.15057
	cluster_infomap	0.24482
484	cluster_fast_greedy	0.53421
	cluster_edge_betweeness	0.51544
	cluster_infomap	0.54344
1087	cluster_fast_greedy	0.14820
	cluster_edge_betweeness	0.03250
	cluster_infomap	0.02737

Table 2: : Modularity scores for five node ID's personalized networks with the core node removed, using three different clustering algorithms

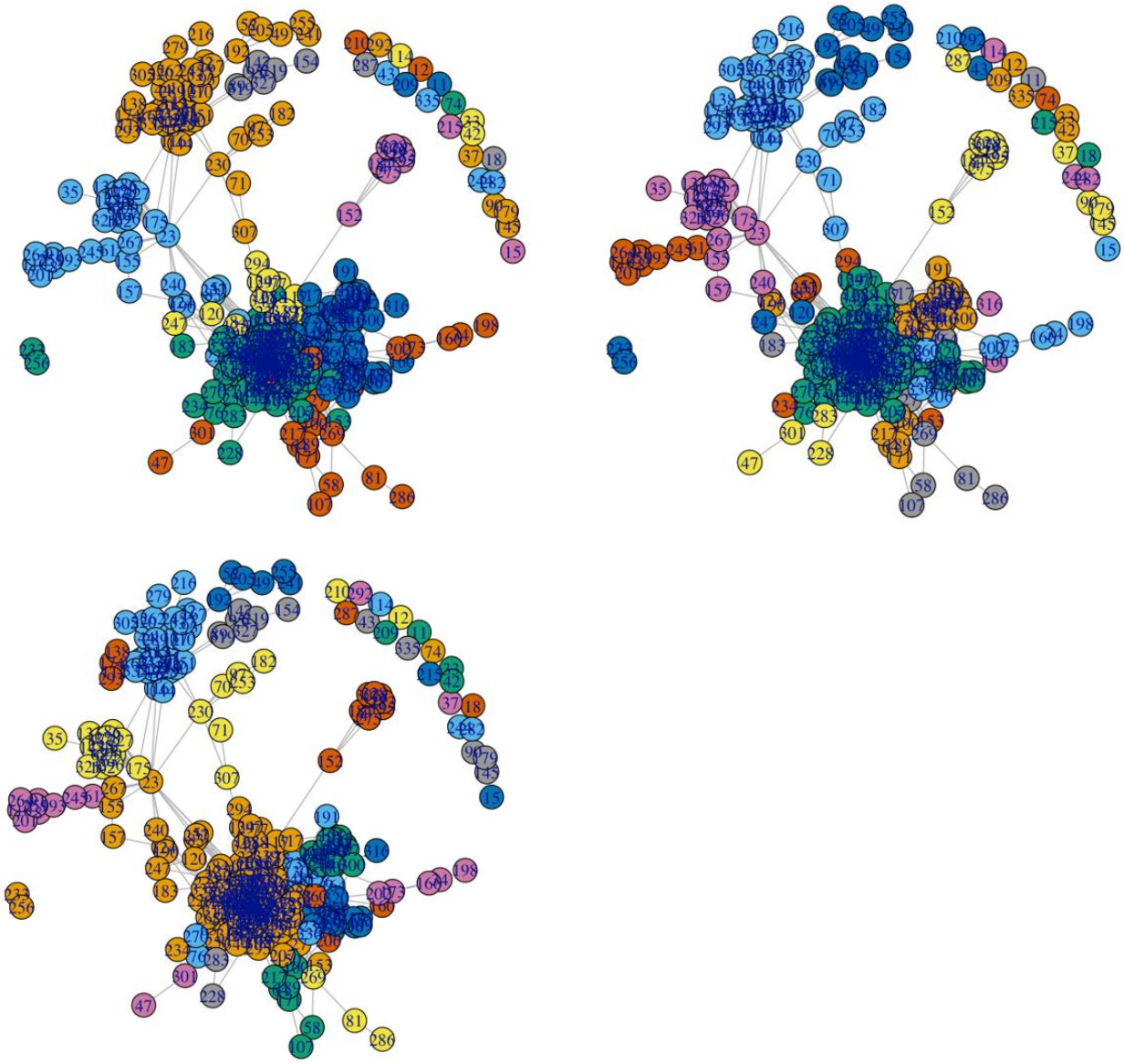

Figure 10: Node ID = 1 personalized with Node 1 removed cluster_fast_greedy, cluster_edge_betweeness, and cluster_infomap (starting in the upper left and moving from left to right)

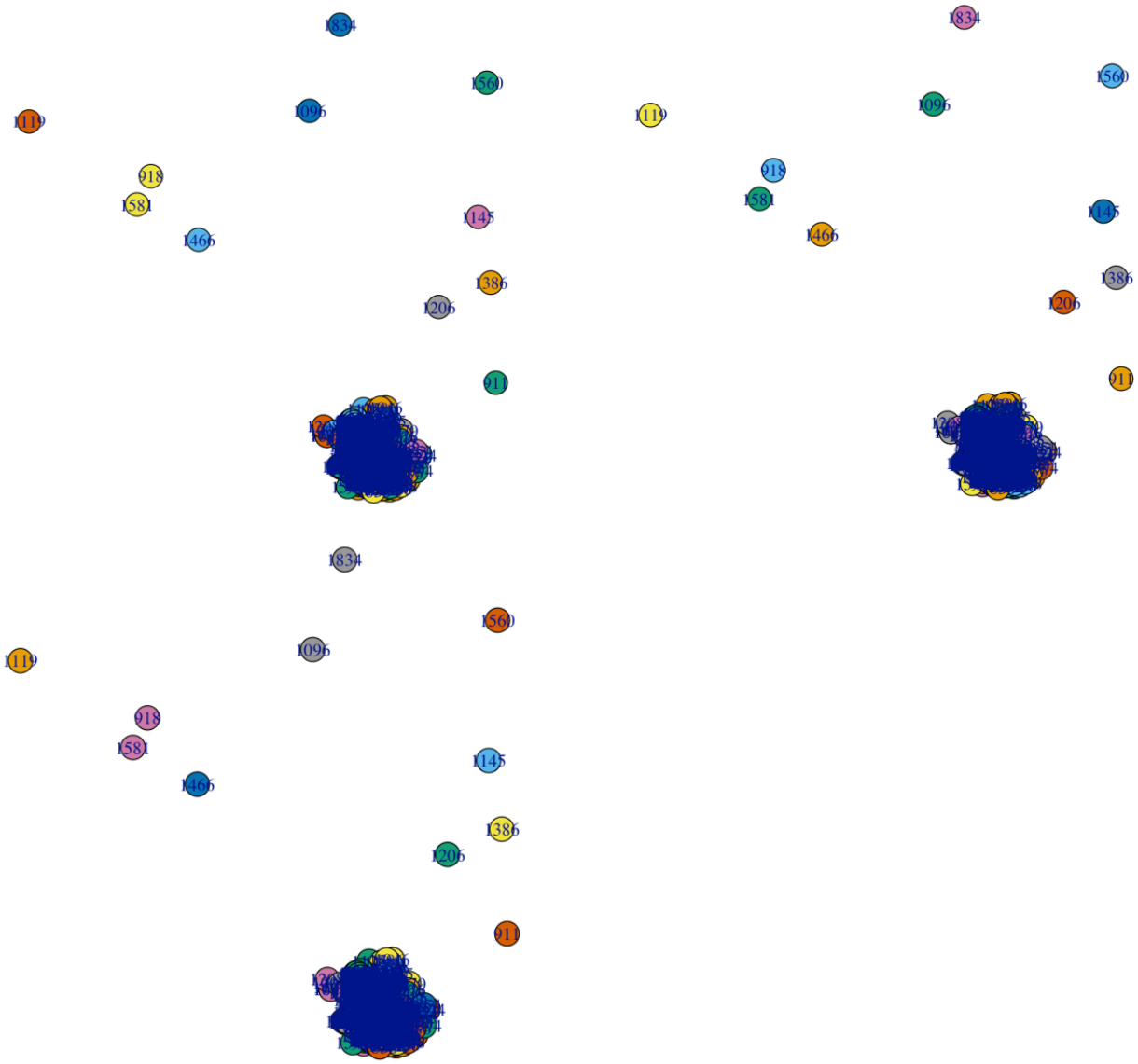

Figure 11: Node ID = 108 personalized with Node 108 removed cluster_fast_greedy, cluster_edge_betweeness, and cluster_infomap (starting in the upper left and moving from left to right)

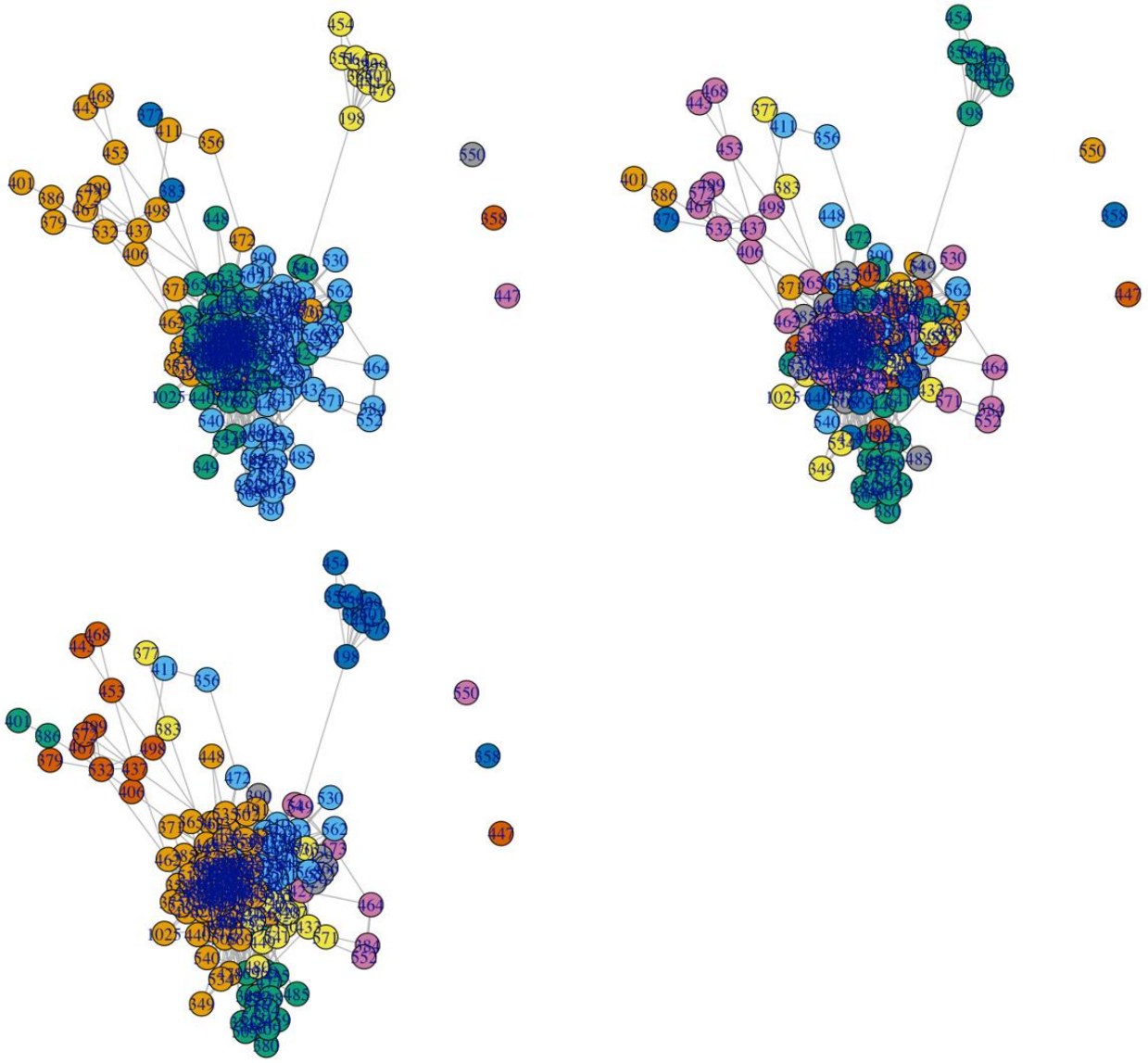

Figure 12: Node ID = 349 personalized with Node 349 removed cluster_fast_greedy, cluster_edge_betweeness, and cluster_infomap (starting in the upper left and moving from left to right)

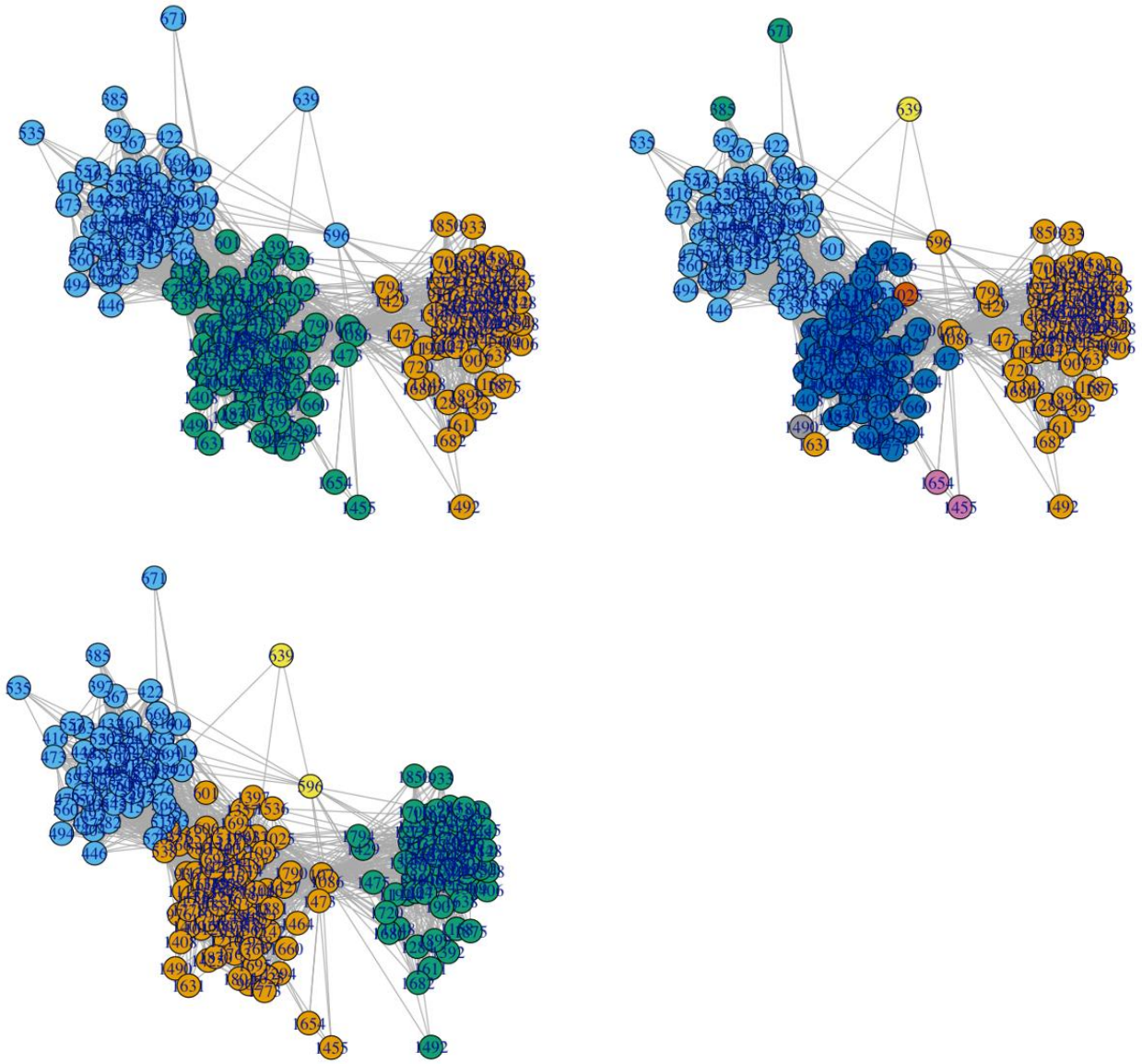

Figure 13: Node ID = 484 personalized with Node 484 removed cluster_fast_greedy, cluster_edge_betweeness, and cluster_infomap (starting in the upper left and moving from left to right)

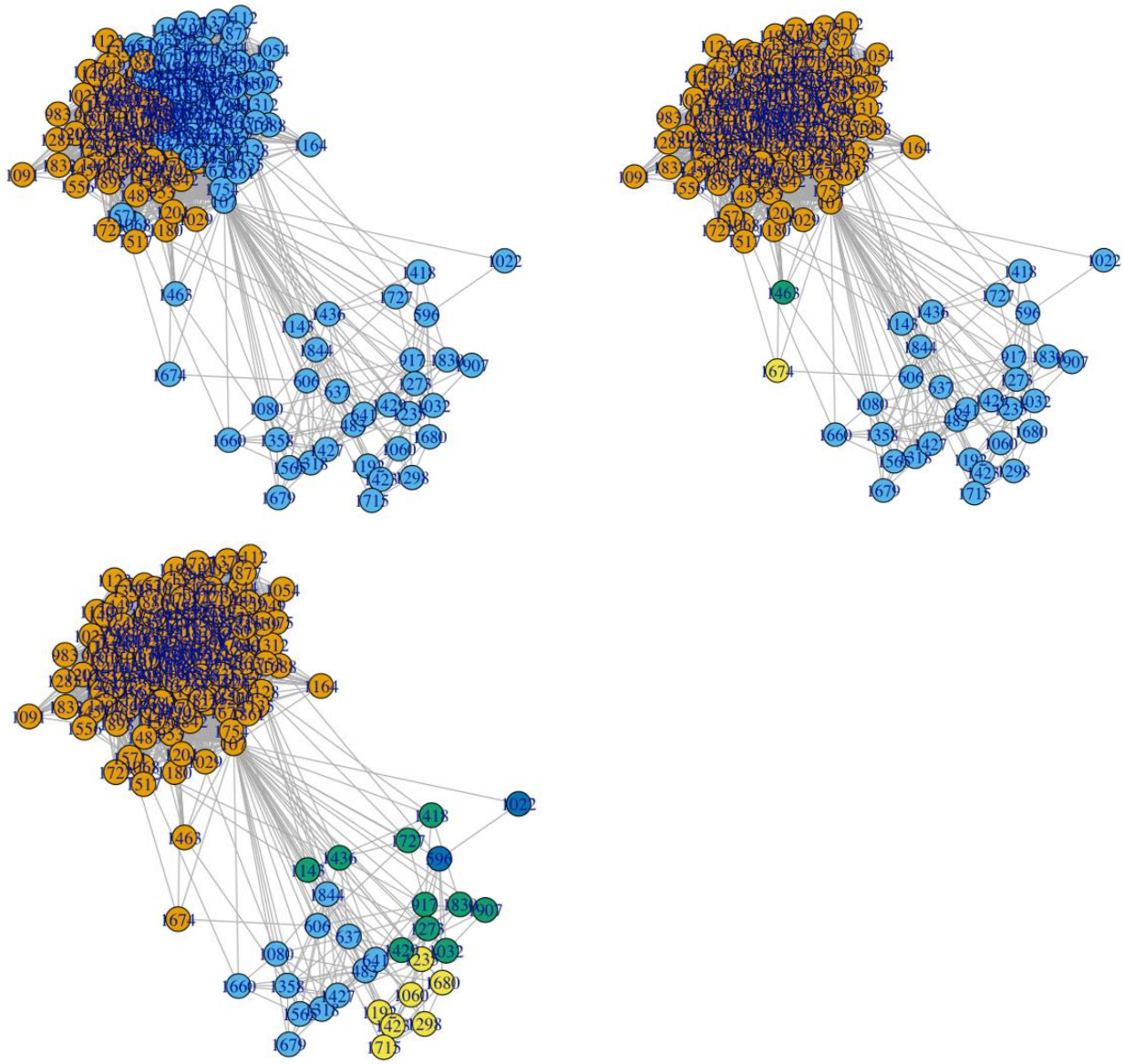

Figure 14: Node ID = 1087 personalized with Node 1087 removed cluster_fast_greedy, cluster_edge_betweeness, and cluster_infomap (starting in the upper left and moving from left to right)

3.3 Characteristic of nodes in the personalized network

Question 11

Embeddedness is a measure of the number mutual friends two people share in a network. We write an expression for embeddedness between the core node and a non-core node to the degree of the non-core node in the personalized network of the core node. With

$g = \text{graph of the personalized network of core node } u,$

then we define embeddedness as

$$\text{embeddedness}(u, v) = \text{degree}(g, v = v) - 1.$$

That is, g is the graph of the personalized network representing core node, u, and v is a non-core node in this personalized network of core node, u. Because core node, u, is already connected to every non-code in the personalized network, simply taking the degree of a non-node, v, will find all the connections of both u and v. However, we subtract one to remove the connection between nodes u and v because we are interested in the number of mutual connections u and v, which excludes the self-connection.

Question 12

For each of the core node's personalized networks, we plot histograms of embeddedness and dispersion. To calculate dispersion, we use the following formula

$$\text{disp}(u, v) = \sum_{s,t \in C_{uv}} d_v(s, t),$$

where d_v is a distance function on the nodes of C_{uv}. C_{uv} is the set of common neighbors for nodes u and v. $d_v(s,t)$ is a piecewise function that is equal to 1 when nodes s and t are not directly connected and have no common neighbors except u and v. Otherwise, $d_v(s,t) = 0$. A high dispersion number indicates a high likelihood that node v is the significant other to u. Looking at Figures 15 – 19, we see a common trend in the dispersion histograms. That is, the frequency of low dispersion numbers is large and there usually is a very small number of occurrences of the highest dispersion numbered bin. This is precisely the type of result we hope to find with dispersion as long as the node v associated with this highest dispersion number is in fact the significant other to user u.

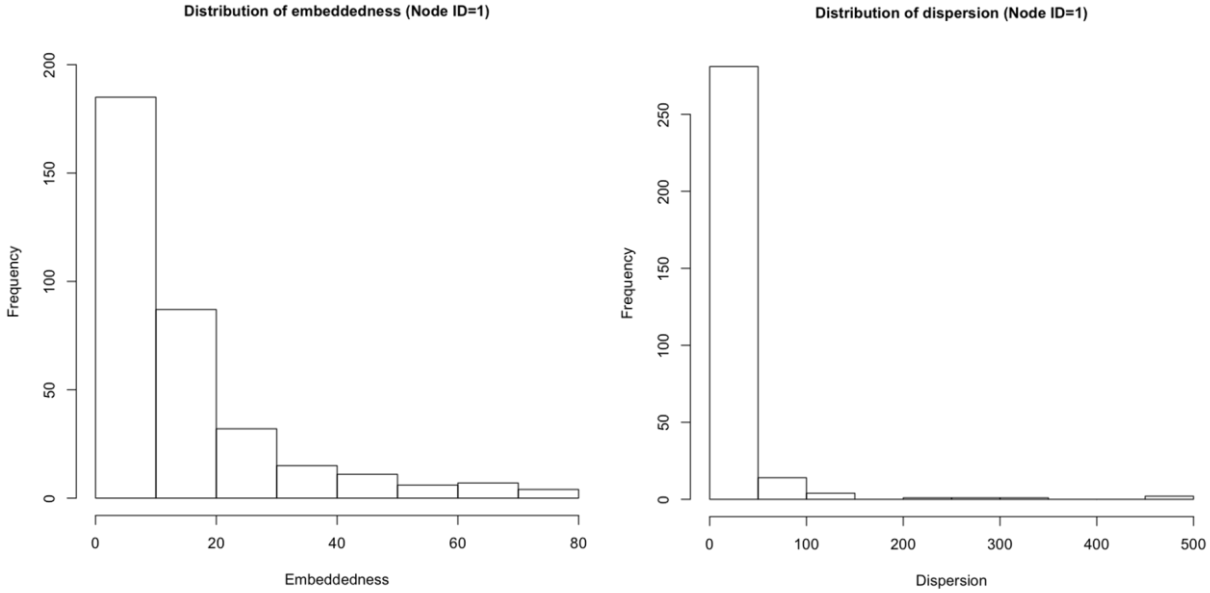

Figure 15: Histogram distribution of embeddedness and dispersion for Node ID = 1

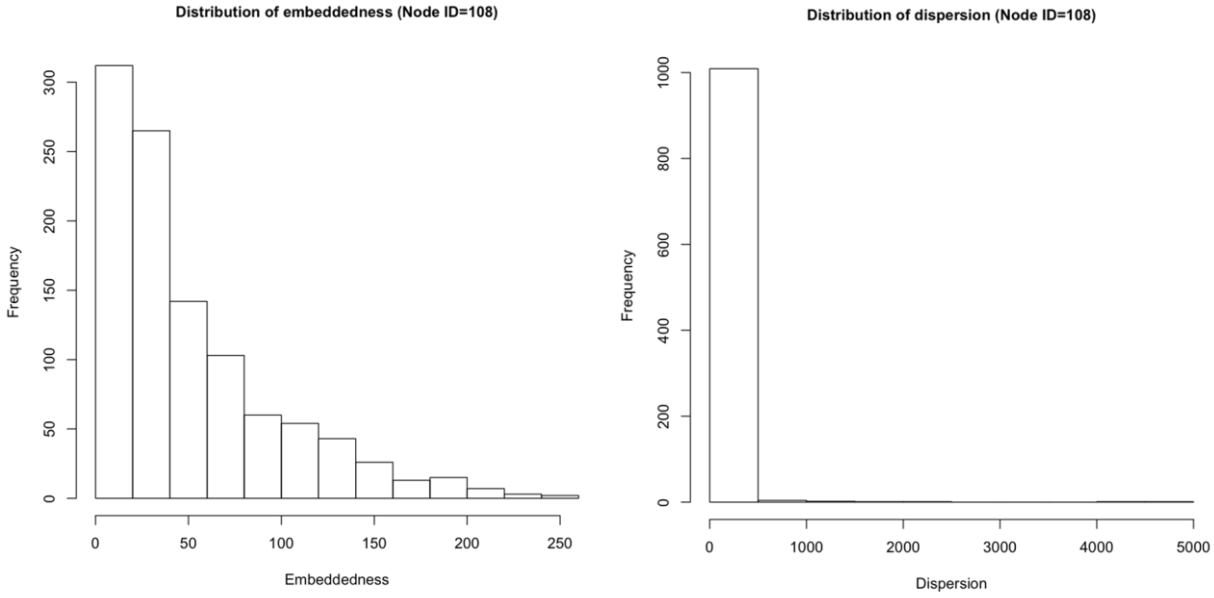

Figure 16: Histogram distribution of embeddedness and dispersion for Node ID = 108

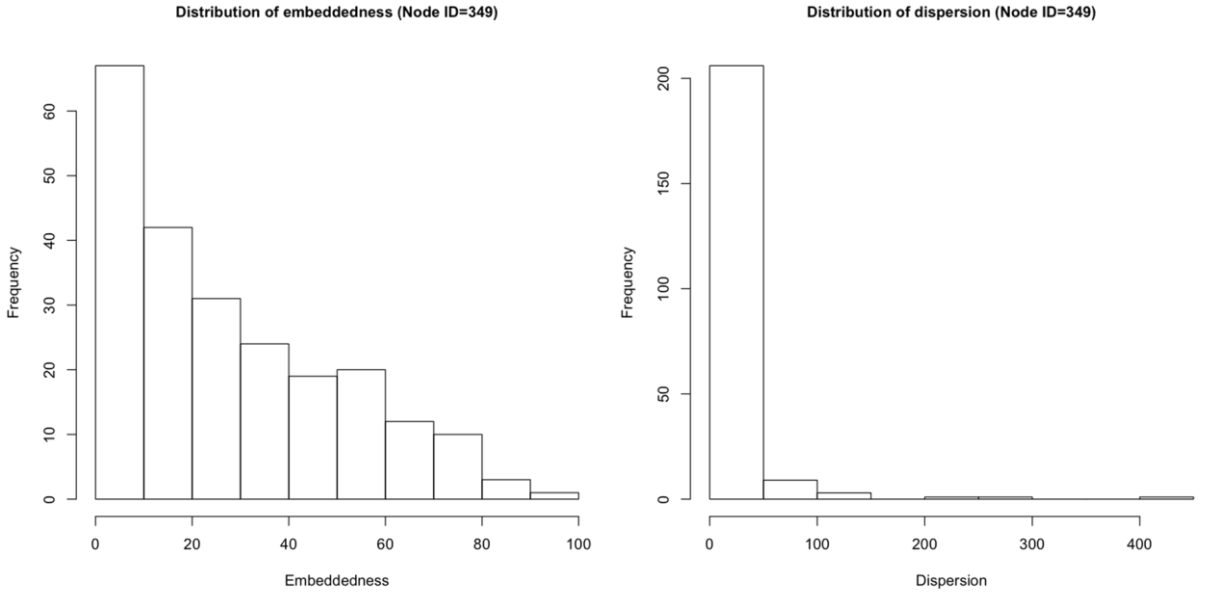

Figure 17: Histogram distribution of embeddedness and dispersion for Node ID = 349

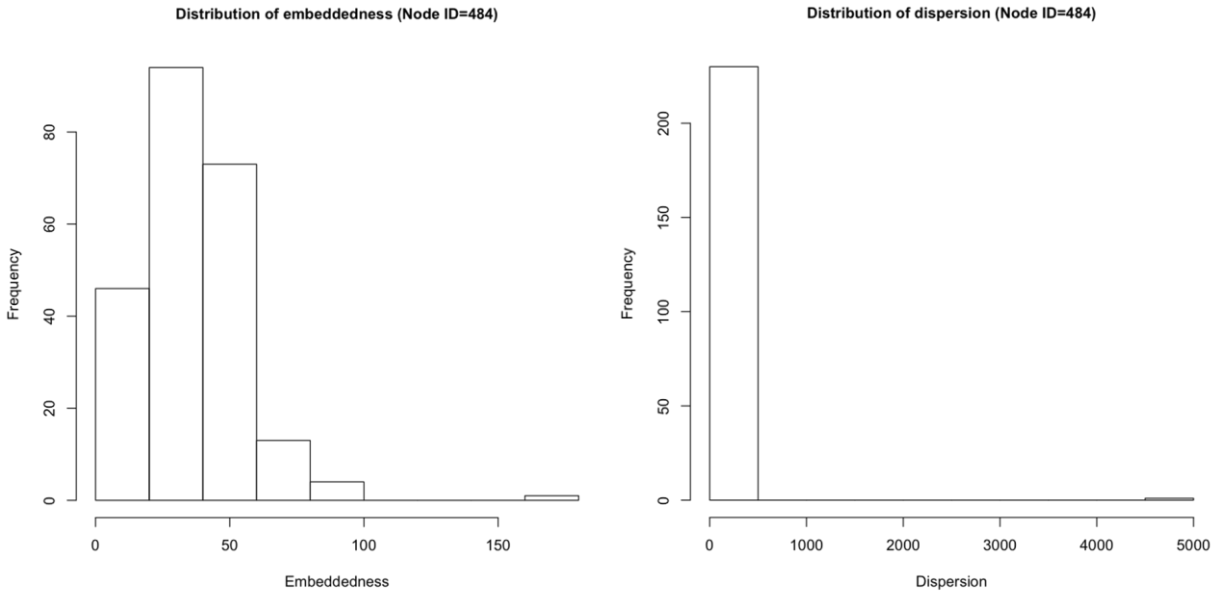

Figure 18: Histogram distribution of embeddedness and dispersion for Node ID = 484

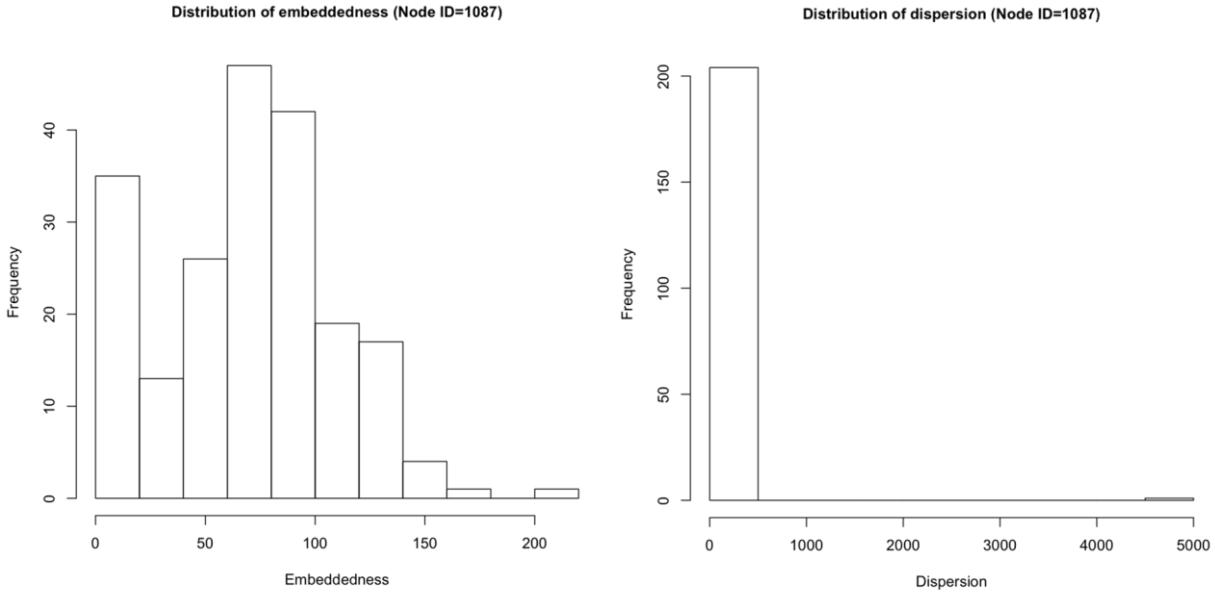

Figure 19: Histogram distribution of embeddedness and dispersion for Node ID = 1087

Question 13

We use the Fast_Greedy algorithm to detect community structure and use colors to highlight this community structure as well as highlight the node with maximum dispersion for all five of the core nodes previously identified. The node with maximum dispersion is identified on each plot, Figure 20-24, in that the node has been enlarged, is inside a square (instead of a circle), and the node number is in red. Additionally, the edges incident to this node have been thickened and colored black to indicate where their location.

Note: in Figure 21, due to the high node density, the node with maximum dispersion is obscured.

Core Node ID	Max Dispersion Node
1	24
108	475
349	561
484	107
1087	107

Table 3: Node with maximum dispersion for each personalized network of the five core nodes identified

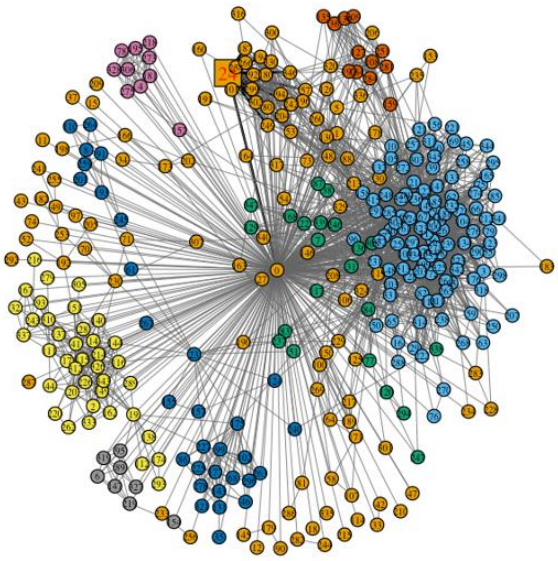

Figure 20: Highlighted community structure and node with maximum dispersion identified for node ID = 1

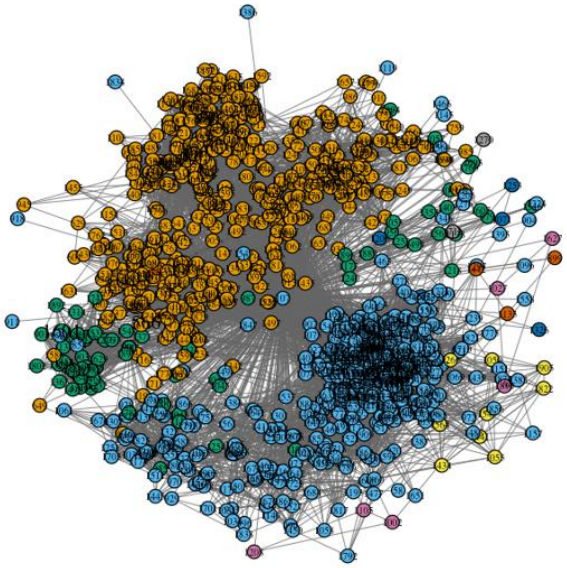

Figure 21: Highlighted community structure and node with maximum dispersion identified for node ID = 108

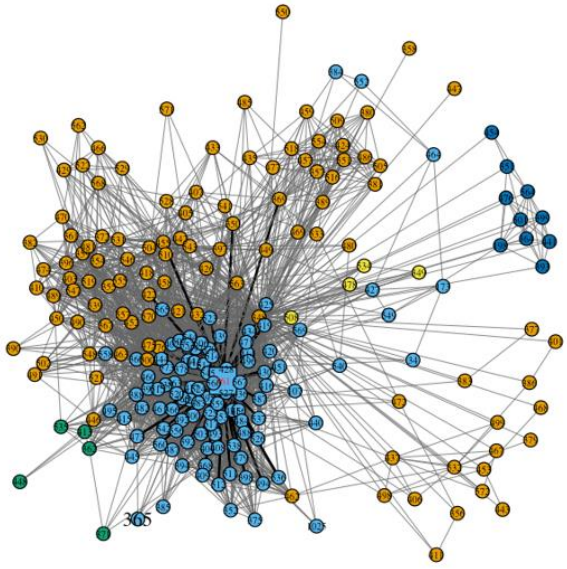

Figure 22: Highlighted community structure and node with maximum dispersion identified for node ID = 349

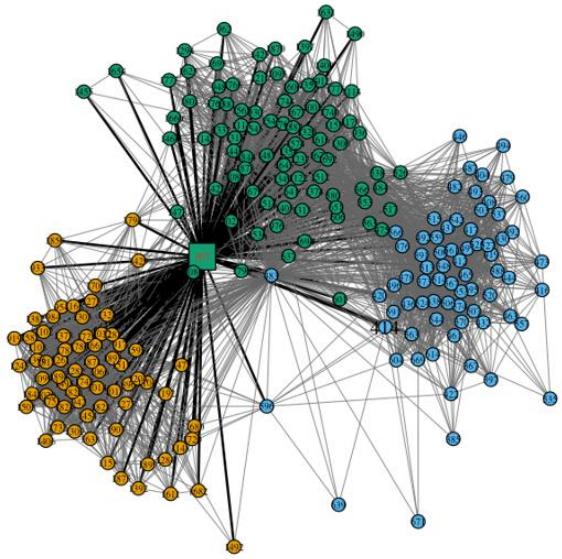

Figure 23: Highlighted community structure and node with maximum dispersion identified for node ID = 484

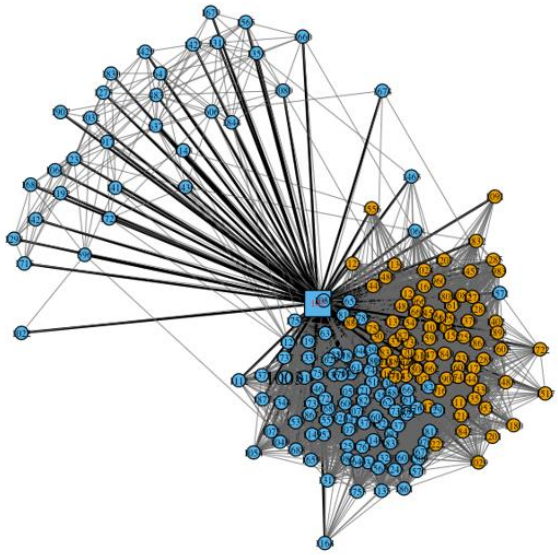

Figure 24: Highlighted community structure and node with maximum dispersion identified for node ID = 1087

Question 14

We find the community structure again using the Fast_Greedy method, but now we highlight the node with maximum embeddedness as well as the node with maximum $\frac{\text{dispersion}}{\text{embeddedness}}$ for the five core nodes used in the previous questions. Refer to Table 4 for more details. For the first three core nodes, the node with maximum embeddedness is not the same as the node with the maximum $\frac{\text{dispersion}}{\text{embeddedness}}$ relationship. However, for the last two cores nodes, the node with maximum embeddedness and maximum $\frac{\text{dispersion}}{\text{embeddedness}}$ are the same.

Note: in Figure 26, the node with maximum $\frac{\text{dispersion}}{\text{embeddedness}}$ is obscured because of the high density of nodes in this graph.

Core Node ID	Max. Dispersion Node	Max. Dispersion / Embeddedness Node
1	55	118
108	1887	475
349	375	561
484	107	107
1087	107	107

Table 4: Node with maximum embeddedness and dispersion / embeddedness for each personalized network of the five core nodes identified

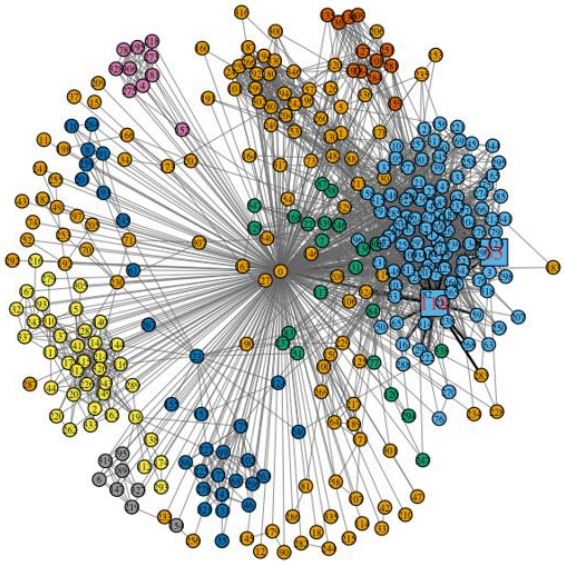

Figure 25: Highlighted community structure and node with maximum embeddedness and dispersion/embeddedness identified for node ID = 1

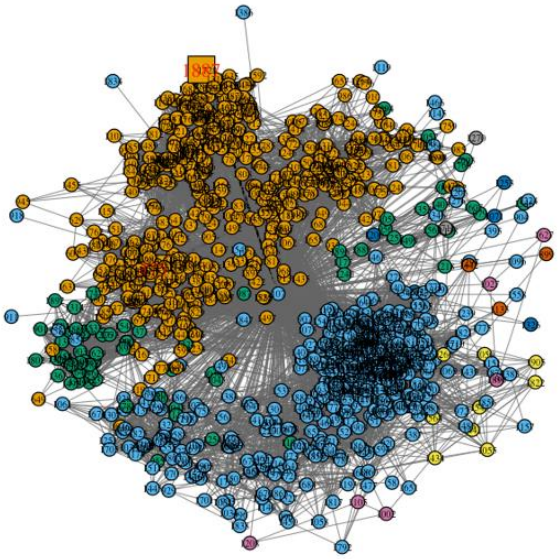

Figure 26: Highlighted community structure and node with maximum embeddedness and dispersion/embeddedness identified for node ID = 108

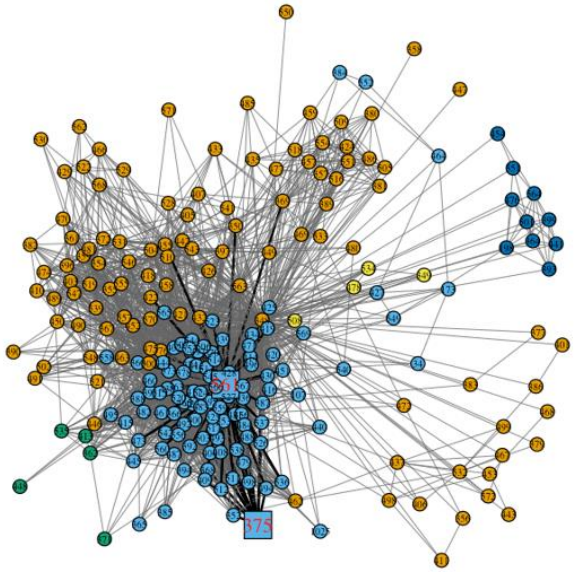

Figure 27: Highlighted community structure and node with maximum embeddedness and dispersion/embeddedness identified for node ID = 349

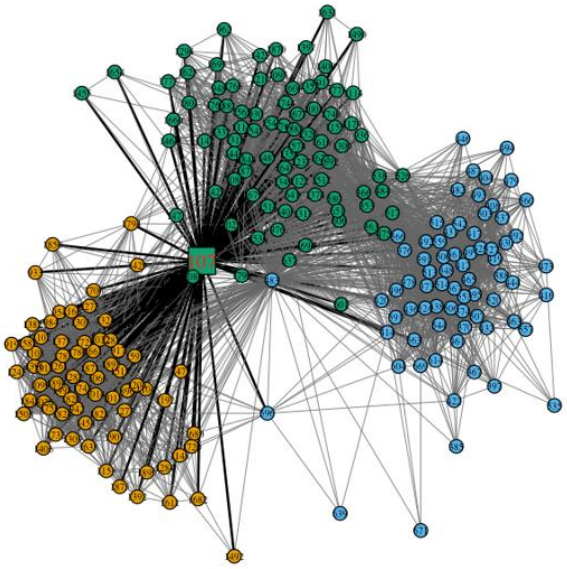

Figure 28: Highlighted community structure and node with maximum embeddedness and dispersion/embeddedness identified for node ID = 484

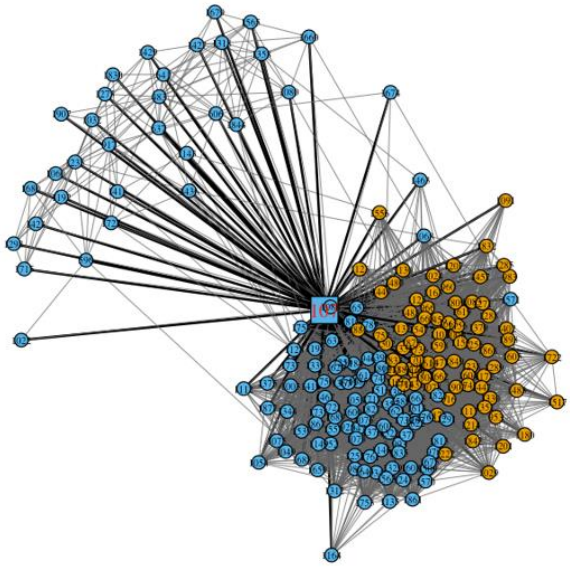

Figure 29: Highlighted community structure and node with maximum embeddedness and dispersion/embeddedness identified for node ID = 1087

Question 15

The nodes with the highest embeddedness tend to be at the center of the highest density cluster (sub-cluster). However, the nodes with maximum dispersion and maximum $\frac{\text{dispersion}}{\text{embeddedness}}$ tend to be around the node bridging multiple clusters. When the bridging node happens to be the one with the most connections, the maximum embeddedness, maximum dispersion, and maximum $\frac{\text{dispersion}}{\text{embeddedness}}$ all tend to select the same node.

4. Friend recommendation in personalized networks

Question 16

We calculate the list of users in the personalized network of node ID = 415 with degree = 24 and call it N_r. There are 11 users the make up the list N_r, i.e., there are 11 people with degree 24 in node ID = 415's personal network.

	User's N_r										
Node ID	496	578	600	615	618	627	643	658	659	661	662

Table 5: List of users N_r

Question 17

We investigate three different friend recommendation algorithms: 1) Common Neighbors measure 2) Jaccard measure 3) Adamic Adar measure. To select which users to recommend friends to, the users from node ID 415's personalized network with degree 24 are chosen, collected in list N_r. We compute the average accuracy of each of the three friend recommendation algorithms. To compute the average accuracy for a user i, we follow a process that randomly removes friends from node i with 0.25 probability (removed friends list is R_i); uses one of the three friend recommendation algorithms to recommend $|R_i|$ new friends to i, denoted as list P_i; and then computes the accuracy as $\frac{|P_i \cap R_i|}{|R_i|}$. This three-step process is repeated 10 times and the average of the accuracy is calculated. This averaging provides the average accuracy for each user in the list N_r. Finally, we take the mean of the average accuracies for each user to calculate the average accuracy for each of the three recommendation measures.

To provide some more insight and detail into how friends are recommended, we describe how we implement the three step process for calculating the average accuracy for each user i in the list N_r. We use a for loop nested inside a for loop as follows in pseudo code.

```
for (uname in unames){
    for (s in 1:10){
        randomly select length(edge_sequence) (which is
            num_friend_recommended = |Ri|) to remove, i.e. removing friends
            from node i
        find the removed friends
        find nodes that are not neighbors of i in edge removed graph (node j)
        use Jaccard neighbor measure to find the Jaccard score for each element
            node j, e.g., Jaccard measure = length(intersect(Si,
            Sj))/length(union(Si,Sj))
        pick the top |Ri| highest Jaccard scores as the friends to be recommended
        calculate the accuracy with |Pi∩Ri|/|Ri|
    }
}
```

We use the same logic as above for the Common Neighbor and Adamic Adar measures
Common Neighbor measure = length(intersect(Si,Sj))
Adamic_Adar Neighbors measure = sum(a+1/log(length(Sk))) if number of neighbor Sk >= 1

The results are:

Common Neighbors measure average accuracy: 0.821700894564672

Jaccard measure average accuracy: 0.799662700365276

Adamic Adar measure average accuracy: 0.829085121964379.

Based on the results, the Adamic Adar measure has the best performance. That is, Adamic Adar beats the next best performer, Common Neighbors measure, by 0.89865%, less than 1%.

Question 18

For the Google+ network data, we find that 57 out of the 132 total networks have more than 2 circles.

Question 19

On the Google+ network, we plot the in-degree and out-degree distributions for personalized networks for the following three node ID's: 109327480479767108490, 115625564993990145546, and 101373961279443806744. The degree distributions are plotted in Figure 30 through Figure 32. The in and out degree distributions do all exhibit exponential decay, but the in degree networks have much more scattering degree frequency, particularly at for the lower degrees, than the out degree distributions.

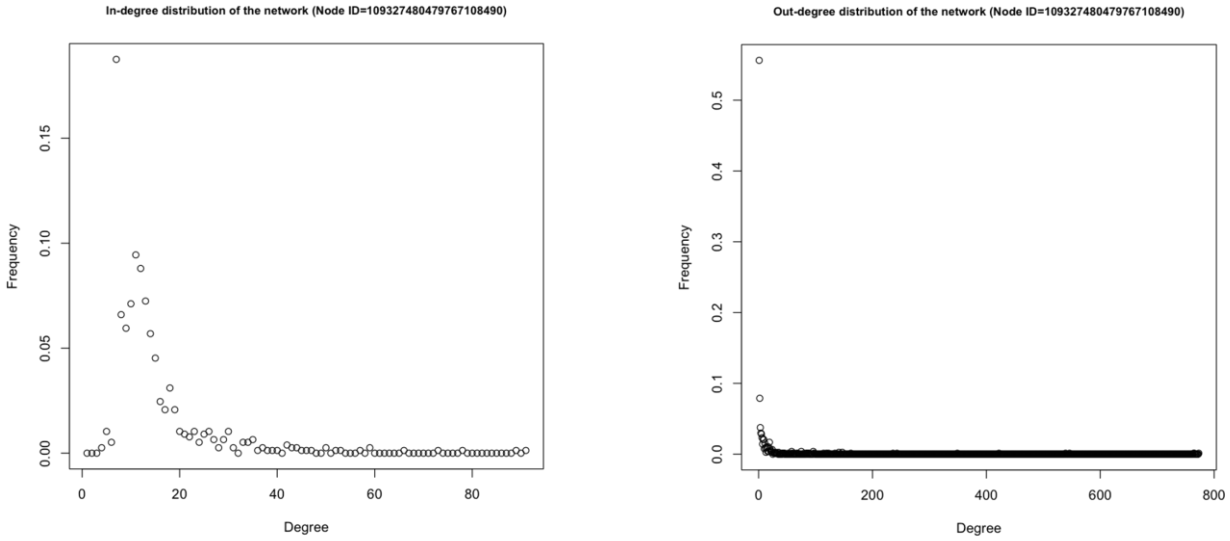

Figure 30: In and out degree distributions for Node ID 109327480479767108490

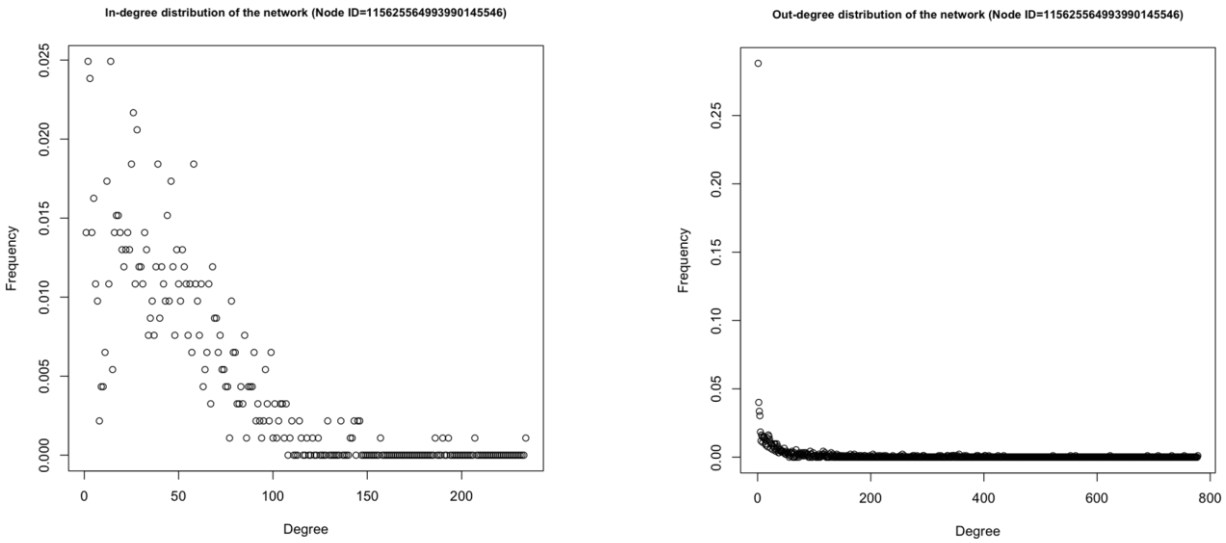

Figure 31: In and out degree distributions for Node ID 11562556493990145546

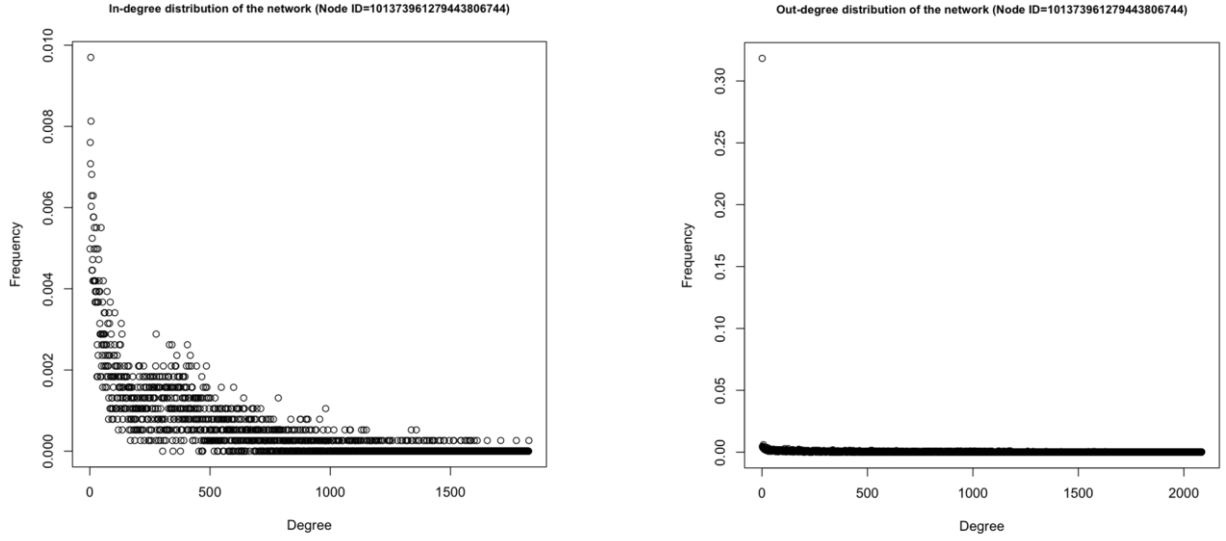

Figure 32: In and out degree distributions for Node ID 101373961279443806744

Question 20

We now extract the community structure, using Walktrap, for each of the three Node ID's personalized networks from Question 19. Figure 33 through Figure 35 display the community structures for these Node ID's. The modularity scores for the first two Node ID's, 109327480479767108490 and 11562556493990145546, are similar but the third is significantly smaller than the other two. That is, the modularity score of

10932748047976710849 0 is 30.3% greater than the modularity score of 10137396127944380674 4.

Node ID	Modularity
10932748047976710849 0	0.2798194
11562556499399014554 6	0.3230868
10137396127944380674 4	0.1950912

Table 6: Modularity scores for each of the three personalized networks based on Walktrap detected community structure

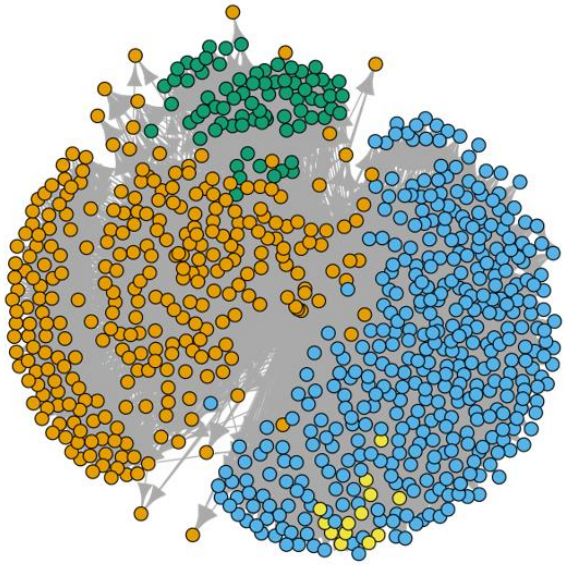

Figure 33: Walktrap community structure for Node ID 10932748047976710849 0

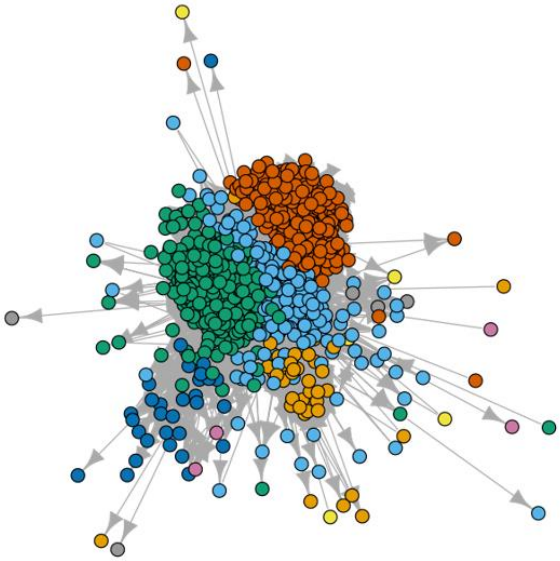

Figure 34: Walktrap community structure for Node ID 115625564993990145546

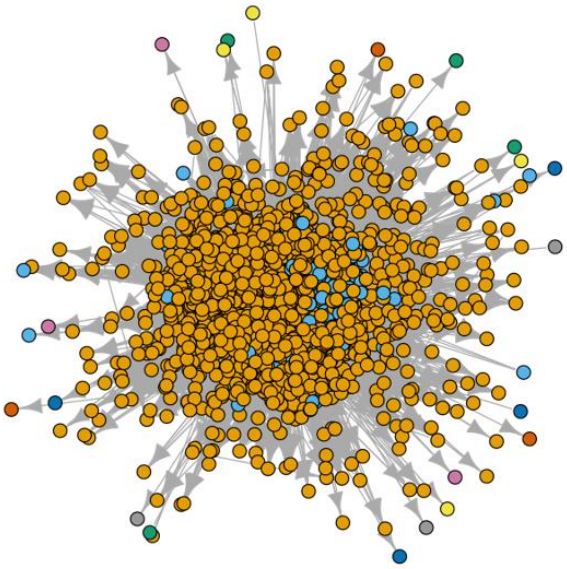

Figure 35: Walktrap community structure for Node ID 101373961279443806744

Question 21

The homogeneity measures to what degree communities contain only members from a single circle, and the completeness measures to what degree members from a given circle are from the same community. A homogeneity score of 1 means that all the communities only contain members from a single circle. Similarly, a completeness score of 1 means that each circle only contains members from the same community.

Taking a closer look at the equations, we discover more insight. First, the equations for homogeneity and completeness.

Homogeneity

$$h = 1 - \frac{H(C|K)}{H(C)}$$

Completeness

$$c = 1 - \frac{H(K|C)}{H(K)}$$

If H(C) or H(K) is equal to 0, then either homogeneity or completeness is undefined, respectively depending on whether H(C) or H(K) is equal to 0. If H(C|K) and H(K|C) are greater than H(C) and H(K) respectively and H(C|K) and H(C) and H(K|C) and H(K) have the same sign, then the values of homogeneity and completeness will be between 0 and 1. However, if $H(C) > H(C|K)$ or $H(K) > H(K|C)$, then the homogeneity or completeness score, respectively, will be less than 0.The equations for H(C), H(K), H(C|K), and H(K|C).

$$H(C) = -\sum_{i=1}^{|C|} \frac{a_i}{N} \log \frac{a_i}{N}$$

$$H(K) = -\sum_{i=1}^{|K|} \frac{b_i}{N} \log \frac{b_i}{N}$$

$$H(C|K) = -\sum_{j=1}^{|K|} \sum_{i=1}^{|C|} \frac{A_{ji}}{N} \log \frac{A_{ji}}{b_j}$$

$$H(K|C) = -\sum_{i=1}^{|C|} \sum_{j=1}^{|K|} \frac{A_{ji}}{N} \log \frac{A_{ji}}{a_j}$$

Question 22

For Node ID=109327480479767108490, $h = 0.894619492175052$ and $c = 0.520001423460879$. If the total number of people with circle information N is roughly the same as the total number of people with community information, such as when the number of true values is roughly the same as the number of predicted values, $H(C) > H(C|K) > 0$ and $H(K) > H(K|C) > 0$. And, thus, $0 < h < 1$ and $0 < c < 1$. This is the case for node ID = 109327480479767108490.

For Node ID=115625564993990145546, $h = 0.792853923841806$ and $c = -0.0649661647569133$. At node ID = 115625564993990145546, there are 31 circles and 25 communities. There are 6467 people who have circle information (N=6467) but only 923 people have community information. Therefore, b is lot smaller than N, which could make H(K) very small. If A_{ji} is just a bit smaller than b_i but a_i is a lot smaller than N, H(K|C) could be bigger than H(K), which makes $\frac{H(K|C)}{H(K)} > 1$ and cause $c < 0$.

For Node ID=10137396127944380674, $\mathrm{h} = -0.941947588891135$ and $\mathrm{c} = 1.43928892513663$. When H(C|K) is larger than H(C), which means conditional entropy of circles is larger than normal entropy at circles, $\mathrm{h} < 0$. There are 29 communities but only 3 circles. Conditional entropy considers all the communities, which introduce more noise than normal circle entropy.

When there are a lot of people in the community that don't have circle information, N could be smaller than b, and $\frac{\mathrm{b_i}}{\mathrm{N}} \log \frac{\mathrm{b_i}}{\mathrm{N}} > 0$. Thus, $\mathrm{H(K)} < 0$ occurs. This is the case for node ID= 10137396127944380674 because $\mathrm{H(K)} < 0, \mathrm{H(K|C)} > 0$, and $\frac{\mathrm{H}(K|C)}{\mathrm{H(K)}} < 0$. Therefore, we find $\mathrm{c} > 1$.