23S-EC ENGR-232E-LEC-1 Project 2: Social Network Mining

SARAH WILEN, ROBERT OZTURK, YAMAN YUCEL

TOTAL POINTS

96 / 100

QUESTION 1

Facebook network 30 pts

1.1 Data exploration **5 / 5**

✓ **- 0 pts** *Correct*

1.2 Personalized network **5 / 5**

✓ **- 0 pts** *Correct*

1.3 Core node’s personalized network **10 / 10**

✓ **- 0 pts** *Correct*

1.4 Friend recommendation in personalized networks **10 / 10**

✓ **- 0 pts** *Correct*

QUESTION 2

Google+ network 20 pts

2.1 Data exploration **5 / 5**

✓ **- 0 pts** *Correct*

2.2 Community structure of personal networks **5 / 5**

✓ **- 0 pts** *Correct*

2.3 Circles and communities **10 / 10**

✓ **- 0 pts** *Correct*

QUESTION 3

Cora dataset 50 pts

3.1 Idea 1 **15 / 15**

✓ **- 0 pts** *Correct*

3.2 Idea 2 **15 / 15**

✓ **- 0 pts** *Correct*

3.3 Idea 3 **16 / 20**

✓ **- 4 pts** *part b with p = 0 is wrong*

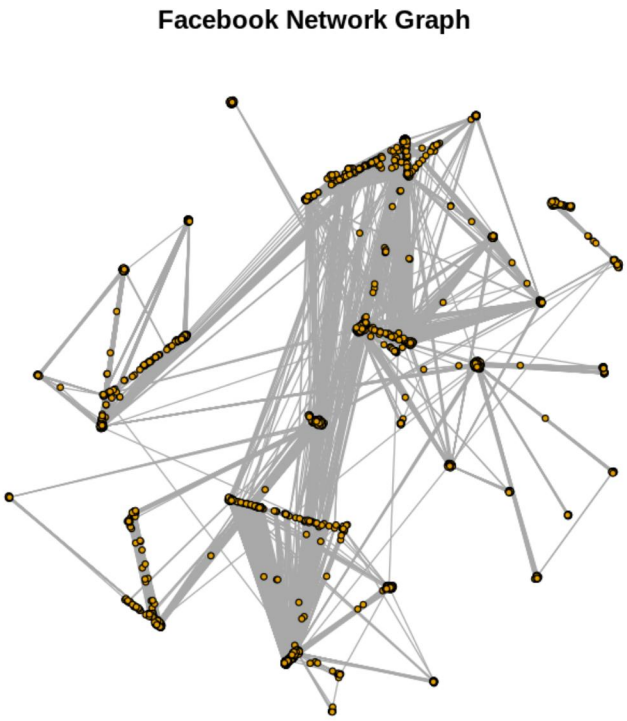
ECE 232E Project 2: Social Network Mining

Sarah Wilen, Yaman Yucel, Robert Ozturk

Facebook Network

Structural Properties of the Facebook Network

Question 1



Question 1.1

Number of nodes: 4039
Number of edges: 88,234

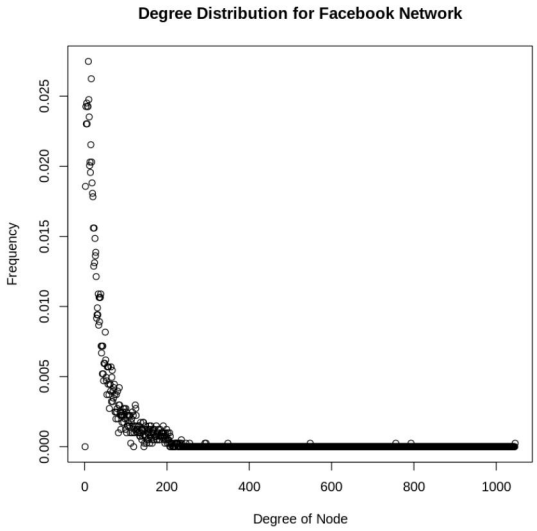
Question 1.2

The graph is connected.

Question 2

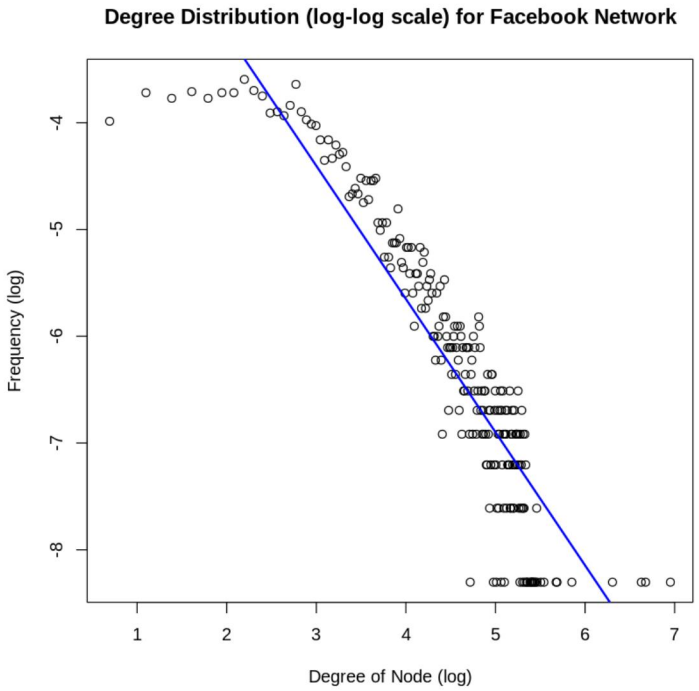
The diameter of the connected graph is 8. Therefore, the length of the shortest path between two furthest nodes is 8.

Question 3



Average Degree: 43.691013

Question 4



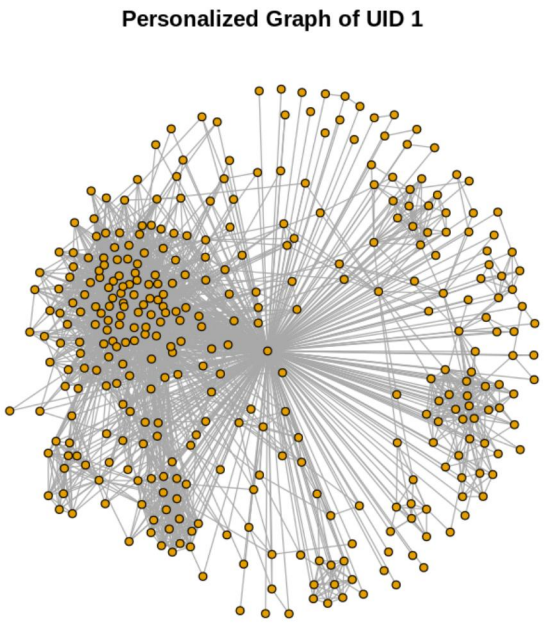
The slope of the line fit to the log-log degree distribution is -1.2475.

1.1 Data exploration **5 / 5**

✓ **- 0 pts** *Correct*

Personalized network

Question 5



Number of nodes: 348
Number of edges: 2866

Question 6

Diameter of personalized graph: 2

Upper bound: 2
Lower bound: 1

Question 7

The upper bound for the personalzied graph has to be 2 because we create the personalzied graph by taking the target node's (UID 1) one-hop neighbors as the subgraph. Therefore, the furthest distance between any two nodes in the graph will only be from source node through the target node to the destination node. This gives a path lengt of 2.

The lower bound is 1 in the case where the source and destination nodes are already connected. All users already know each other directly. There exists direct edges between all nodes in the newtork in the case where the diameter is 1. Specifically, in the simplified example case of a network with only three nodes, if

all three nodes are connected, then the shortest path between the source and destinattion node is just 1.
Core node's personalized network

Question 8

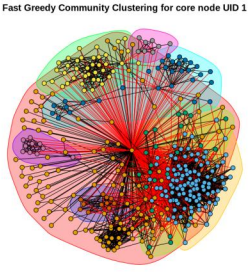
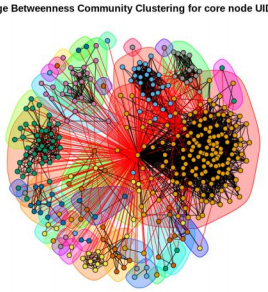
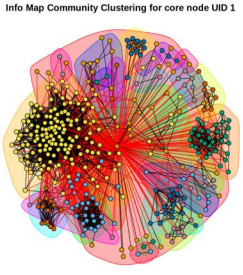
Number of core nodes: 40
Average Degree of Code Nodes: 279.375000

Question 9

Modularity	Fast-Greedy	Edge-Betweenness	Infomap
User 1	0.413101	0.353302	0.389118
User 108	0.435929	0.506755	0.508223
User 349	0.25175	0.133528	0.095464
User 484	0.507002	0.489095	0.515279
User 1087	0.145531	0.027624	0.026907

User 484 has the largest modularity and user 1087 has the least. We can see this realized in the figures as well since user 484 has dense connections between nodes within clusters, but sparser in between clusters. We notice user 1087's clusters are not as dense. For example, looking at the fast-greedy community structure, the blue cluster spans a large area even covering other clusters, but in user 484, the cluster colors are well defined, dense, and contained within themselves.

In general, edge-betweenness resulted in the lowest modularity scores. Fast-greedy performs, on average, the highest modularity scores. Since the fast-greedy algorithm yields the local optimum, this fast-greedy algorithm's cost function is to increase the modularity. With this said, everything took a really long time to run on User 108, and when investigated, we notice that user 108 has the largest network (by node size).

Community Structure	Fast-Greedy	Edge-Betweenness	Infomap
User 1			

1.2 Personalized network **5 / 5**

✓ **- 0 pts** *Correct*

all three nodes are connected, then the shortest path between the source and destinattion node is just 1.
Core node's personalized network

Question 8

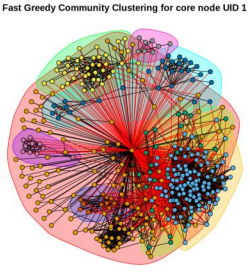
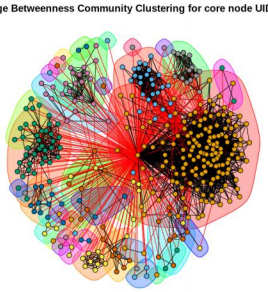
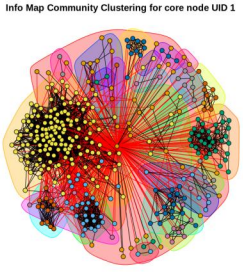
Number of core nodes: 40
Average Degree of Code Nodes: 279.375000

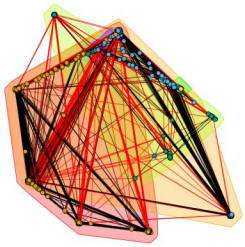
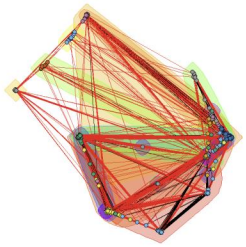
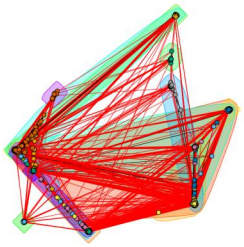
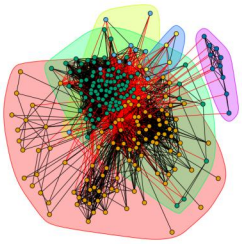
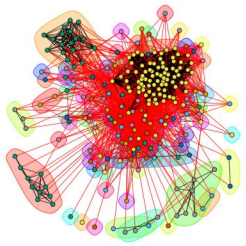
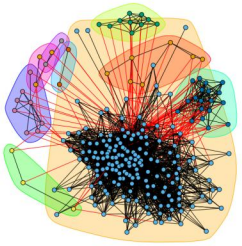
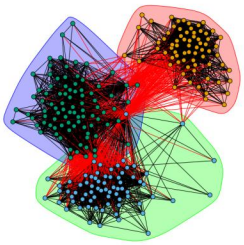
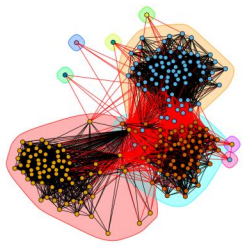
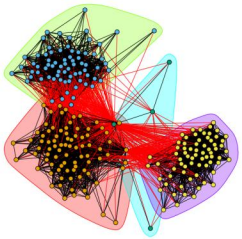
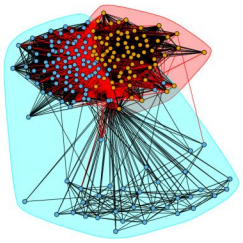
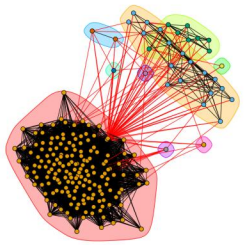
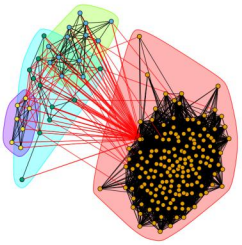
Question 9

Modularity	Fast-Greedy	Edge-Betweenness	Infomap
User 1	0.413101	0.353302	0.389118
User 108	0.435929	0.506755	0.508223
User 349	0.25175	0.133528	0.095464
User 484	0.507002	0.489095	0.515279
User 1087	0.145531	0.027624	0.026907

User 484 has the largest modularity and user 1087 has the least. We can see this realized in the figures as well since user 484 has dense connections between nodes within clusters, but sparser in between clusters. We notice user 1087's clusters are not as dense. For example, looking at the fast-greedy community structure, the blue cluster spans a large area even covering other clusters, but in user 484, the cluster colors are well defined, dense, and contained within themselves.

In general, edge-betweenness resulted in the lowest modularity scores. Fast-greedy performs, on average, the highest modularity scores. Since the fast-greedy algorithm yields the local optimum, this fast-greedy algorithm's cost function is to increase the modularity. With this said, everything took a really long time to run on User 108, and when investigated, we notice that user 108 has the largest network (by node size).

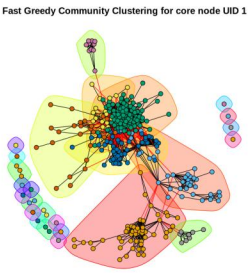
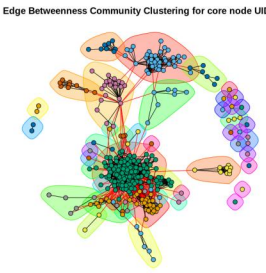
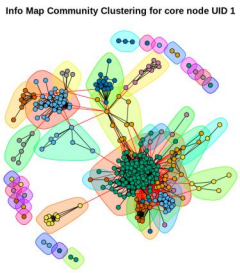
Community Structure	Fast-Greedy	Edge-Betweenness	Infomap
User 1			

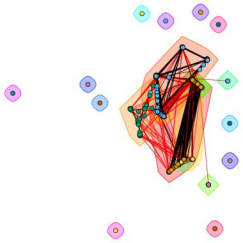
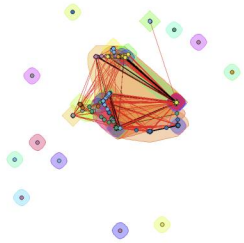
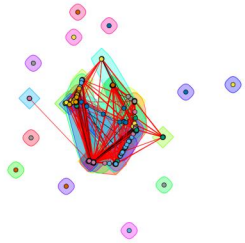
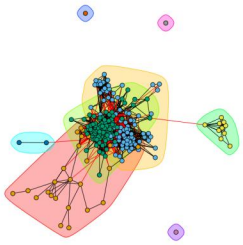
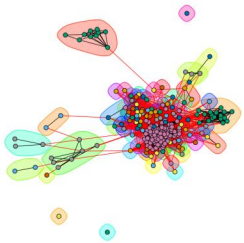
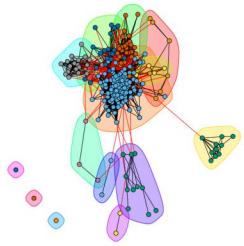
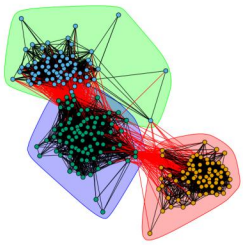
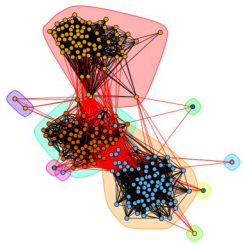
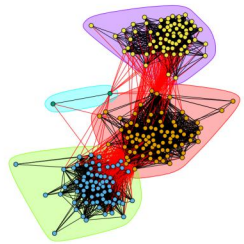
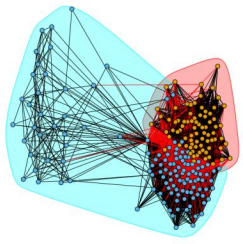
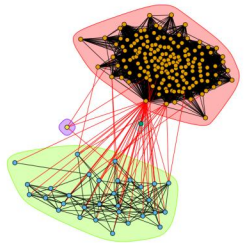
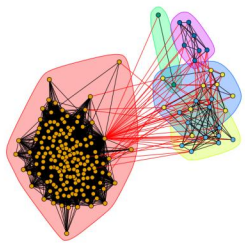
User 108	Fast Greedy Community Clustering for core node UID 108 	Edge Betweenness Community Clustering for core node UID 108 	Info Map Community Clustering for core node UID 108 
User 349	Fast Greedy Community Clustering for core node UID 349 	Edge Betweenness Community Clustering for core node UID 349 	Info Map Community Clustering for core node UID 349 
User 484	Fast Greedy Community Clustering for core node UID 484 	Edge Betweenness Community Clustering for core node UID 484 	Info Map Community Clustering for core node UID 484 
User 1087	Fast Greedy Community Clustering for core node UID 1087 	Edge Betweenness Community Clustering for core node UID 1087 	Info Map Community Clustering for core node UID 1087 

Question 10

We notice quite clearly after removing the core node, the moduarlity scores become much higher than with the core nodes. The core nodes are those with many neighbors and, therefore, have connections between a lot of different communities or friend groups in the personalized networks. Therefore, when we remove the core nodes, the clusters (the modules) become very clearly separated. There is no longer a link node that connects multiple communities together, hence, reesulting in really distinctly separate communities. We notice from the community graphs, there is very little overlap in the colored clusters meaning the communities are well-separated, hence, resulting in the increased modularity score.

Modularity	Fast-Greedy	Edge-Betweenness	Infomap
User 1	0.441853	0.416146	0.418008
User 108	0.435929	0.521322	0.520181
User 349	0.245692	0.150566	0.246578
User 484	0.534214	0.515441	0.543444
User 1087	0.148196	0.03295	0.027372

Community Structure	Fast-Greedy	Edge-Betweenness	Infomap
User 1			

User 108	Fast Greedy Community Clustering for core node UID 108 	Edge Betweenness Community Clustering for core node UID 108 	Info Map Community Clustering for core node UID 108 
User 349	Fast Greedy Community Clustering for core node UID 349 	Edge Betweenness Community Clustering for core node UID 349 	Info Map Community Clustering for core node UID 349 
User 484	Fast Greedy Community Clustering for core node UID 484 	Edge Betweenness Community Clustering for core node UID 484 	Info Map Community Clustering for core node UID 484 
User 1087	Fast Greedy Community Clustering for core node UID 1087 	Edge Betweenness Community Clustering for core node UID 1087 	Info Map Community Clustering for core node UID 1087 

Question 11

The degree of a node is defined to be the number of edges connected to that node. Core node is connected to every node in the personalized network of the core node, therefore a non-core node should be connected to some neighbors of the core node. In other terms, every neighbor of a non-core node should be a neighbor of the core node in the personalized network of the core node. The number of nodes shared among core-node and non-core nodes can be easily computed with the following formula.

$$Embeddedness(core\ node,\ non\ -\ core\ node)\ = d(non\ -\ core\ \ node)_{c}\ -\ 1$$

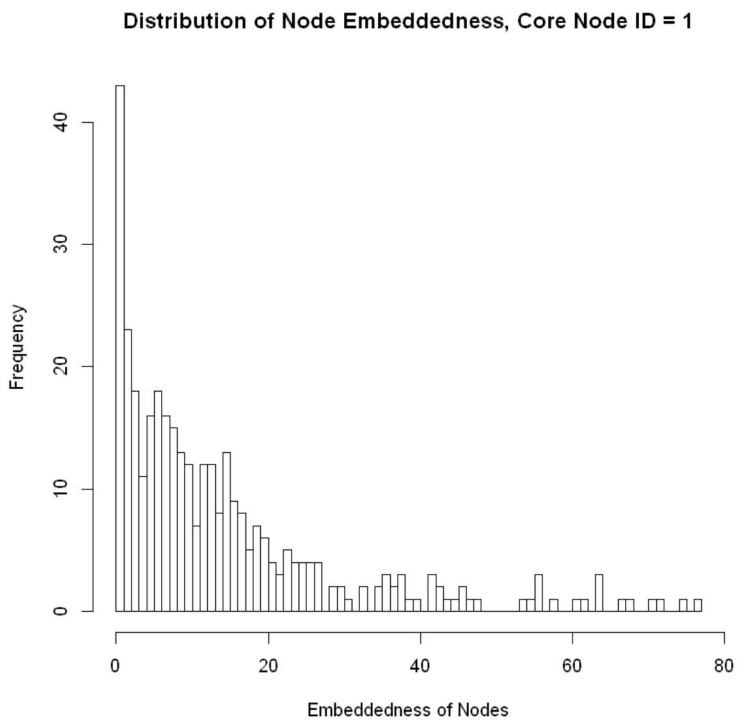
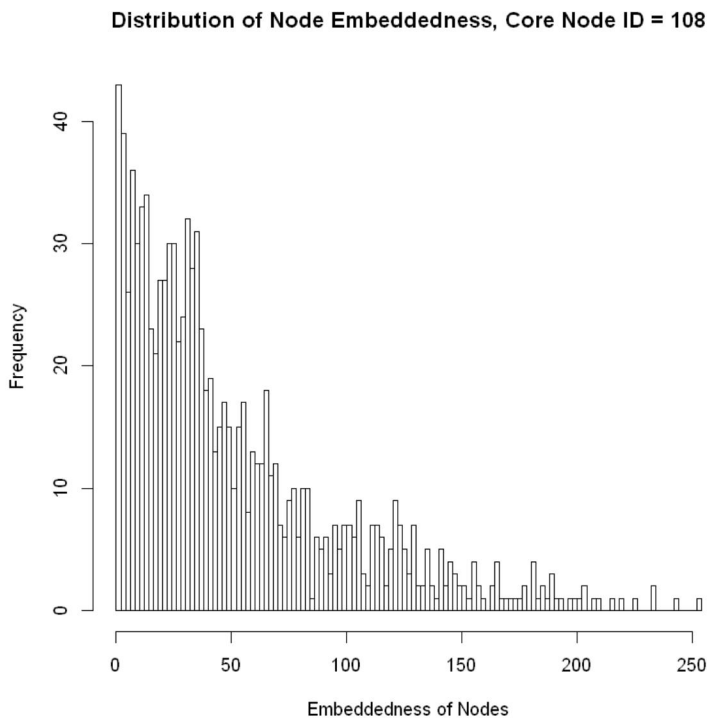
One needs to discard 1 since the degree of a non-core node also consists of the core node. Embeddedness is defined as the number of friends a node shares with the core node which states that core node should not be counted as a friend node.

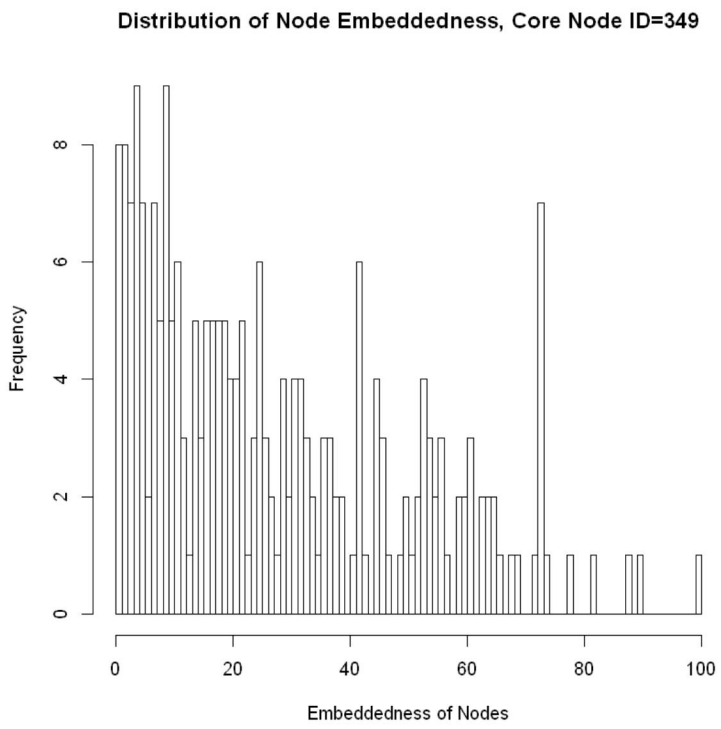
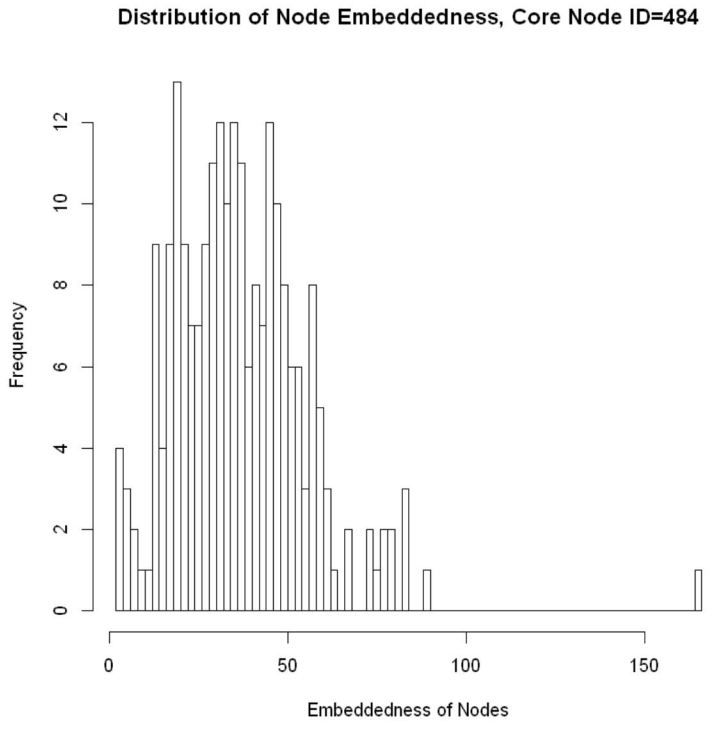
Question 12

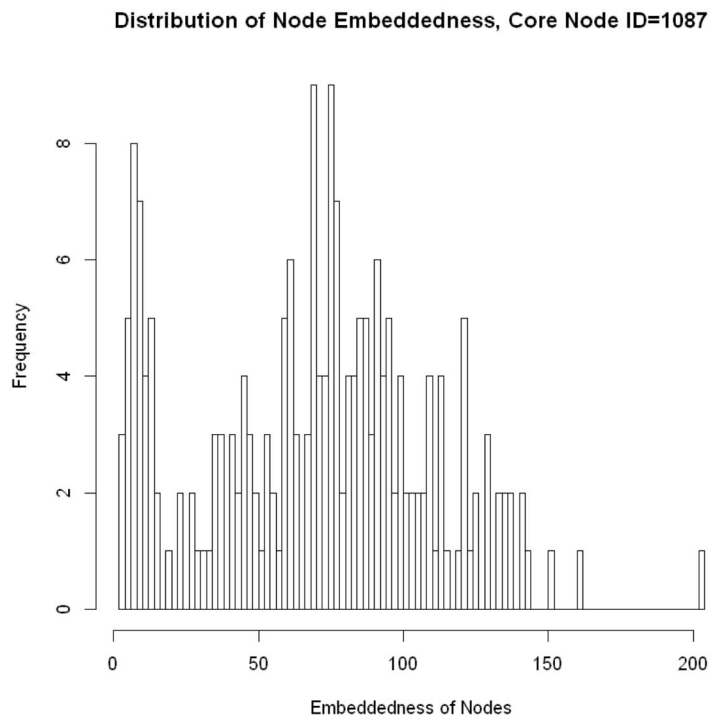
- Embeddedness of a node is defined as the number of mutual friends a node shares with the core node which can be easily computed with the expression in Question 11.

- Dispersion of a node is defined as the sum of distances between every pair of the mutual friends the node shares with the core node. The distances should be calculated in a modified graph where the node (whose dispersion is being computed) and the core node are removed. In order to find dispersion of a non-core node, one needs to first find the mutual friends between non-core node and core node which are basically the neighbors of non-core nodes excluding the core node. Then a graph excluding the core node and non-core node should be constructed. The next step is to calculate the shortest distance from a mutual friend to another mutual friend, since the graph is undirected we have not included repeating pairs. For example, the shortest path from n1 to n2 is calculated but n2 to n1 is not used for finding the dispersion. Lastly, summing all shortest paths yields the dispersion of that non core node.

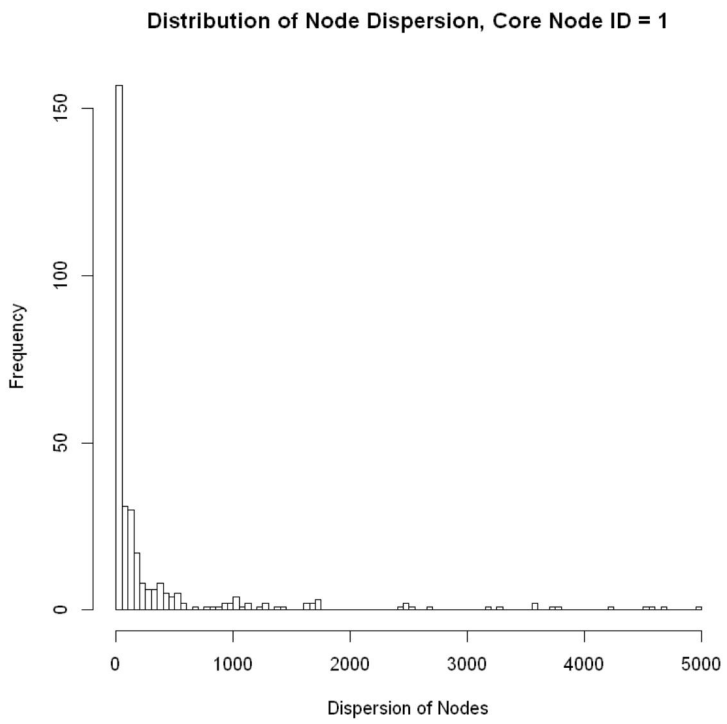
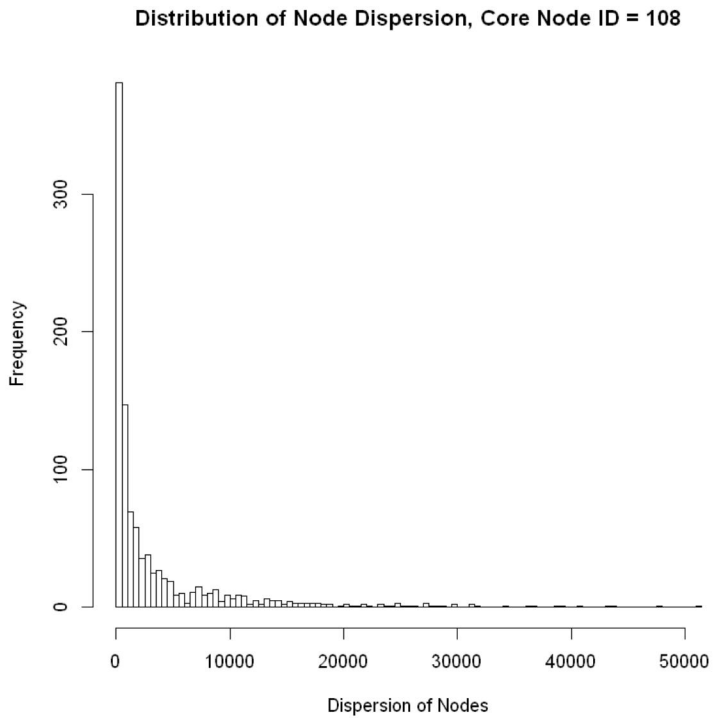
  For dispersion, if there is no path between mutual friends, that path(Inf distance) is not included in the summation.

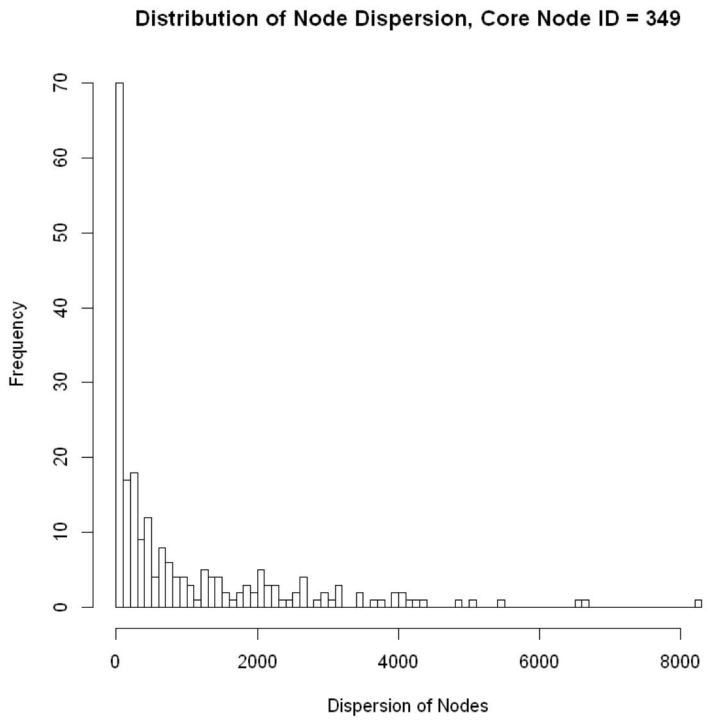
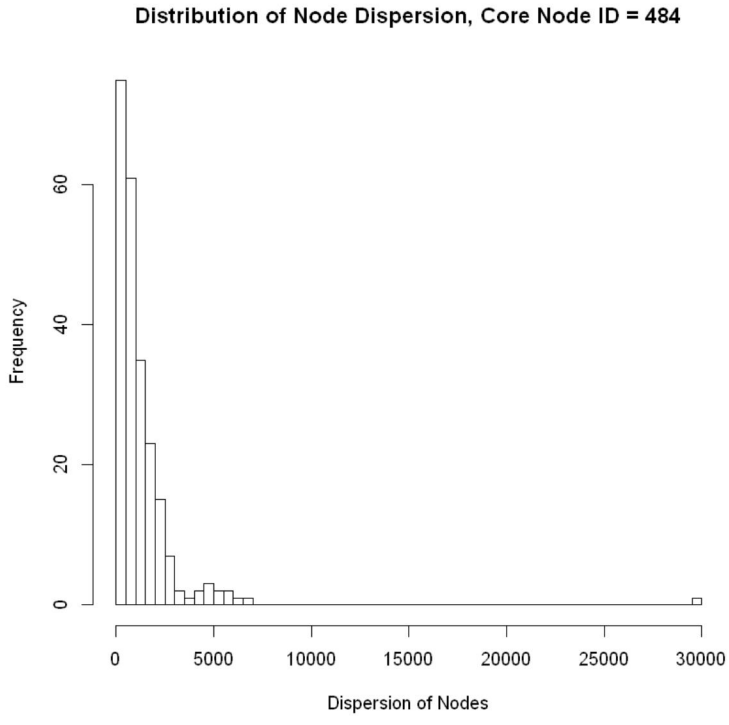
	Embeddedness Histogram
User 1	
User 108	

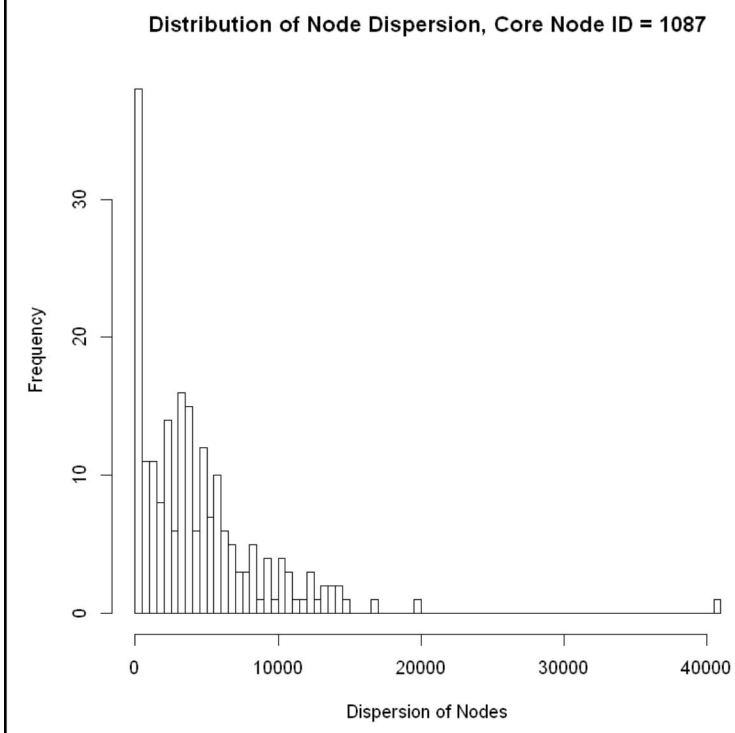
User 349	
User 484	

User 1087	

Distribution of Node Embeddedness, Core Node ID=1087
Frequency
0
2
4
6
8
0
50
100
150
200
Embeddedness of Nodes

	Dispersion Histogram
User 1	
User 108	

User 349	
User 484	

User 1087	

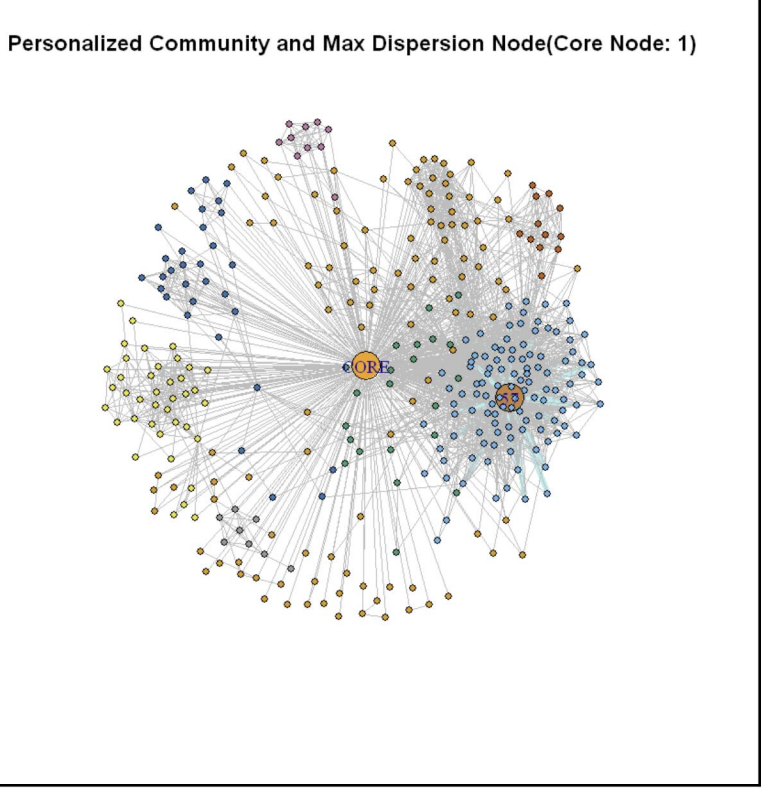
For embeddedness, we can conclude that non-core nodes that have high degree, also have high embeddedness. Therefore, distribution of the histogram is very similar to the degree distribution. In particular, the embeddedness histogram of core node 1. 108 and 349 nearly follow a power-law distribution and the embeddedness histograms of core nodes 484 and 1087 are similar to binomial distributions. High embeddedness is only observed when the number of mutual friends with the core node is high. The highest embeddedness is observed at the core node 108 network.

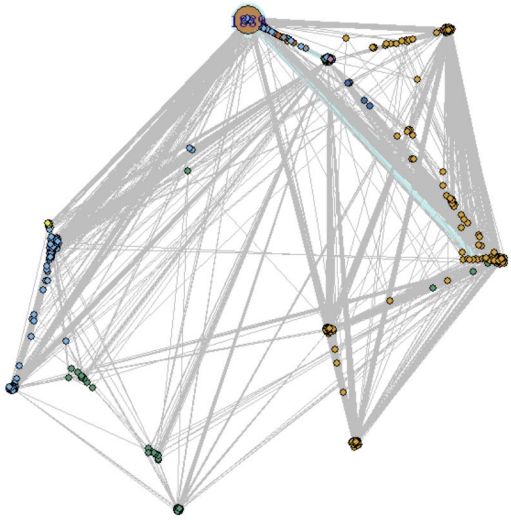
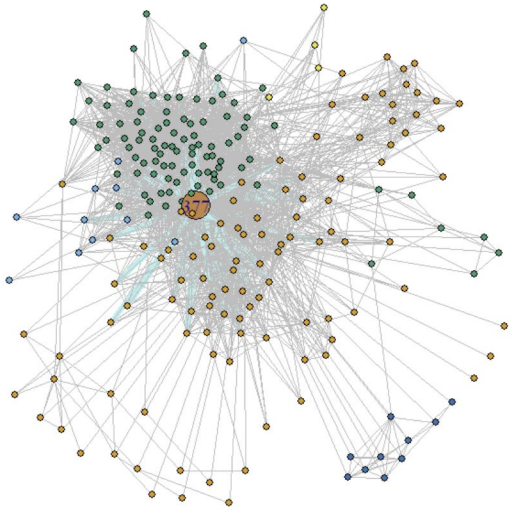
When dispersion increases, the number of nodes that have that dispersion decreases. Also note that dispersion of a node increases when there are more mutual friends present or in other words have high embeddedness. In addition to the number of mutual friends, those friends should have multiple nodes between them to get high dispersion. The dispersion histogram can easily show the dispersion of the personalized network. Most of the mutual friends have small distances in each personalized network, so we have observed a decreasing trend in the histogram. In result, the distribution histograms of dispersion are a bit similar to power-law distributions.

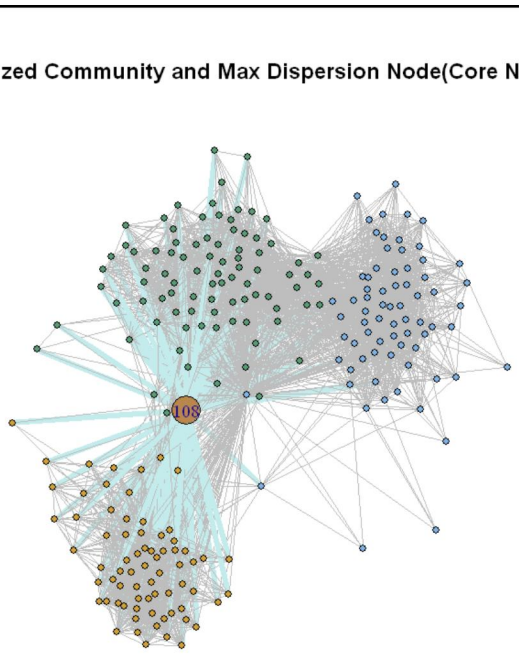
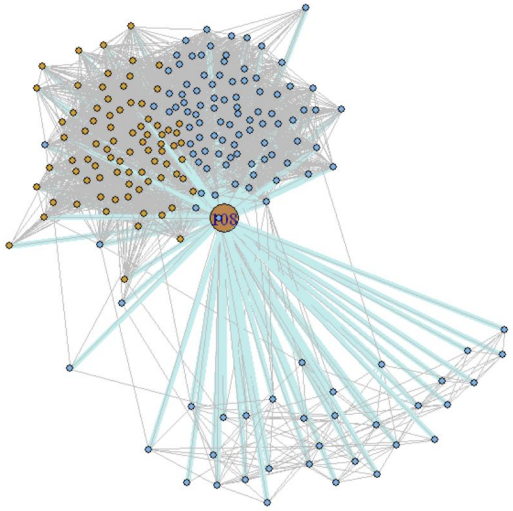
Question 13

Boundaries for communities are not drawn, they are labeled with different colors. Core and max dispersion nodes are marked with a big circle. Edges of max nodes are colored with turquoise.

	Max Dispersion ID
User 1	57
User 108	1889
User 349	377
User 484	108
User 1087	108

	Max Dispersion Graph
User 1	

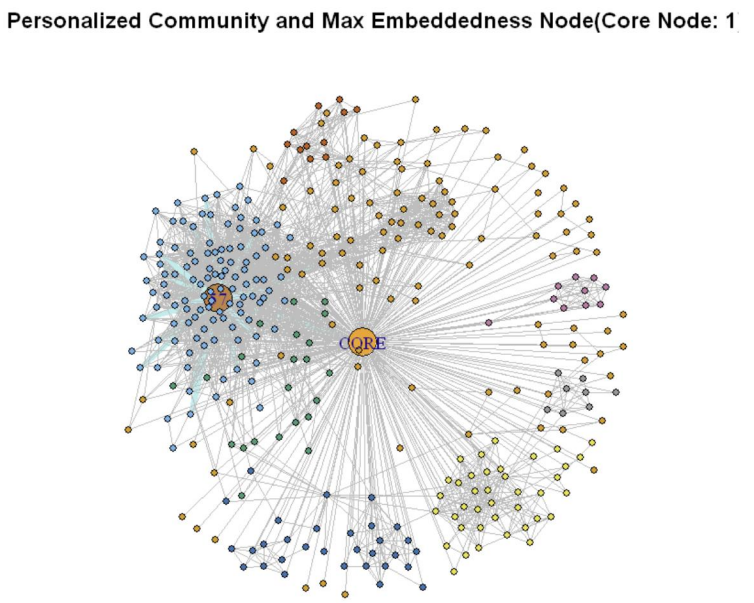
User 108	**Personalized Community and Max Dispersion Node(Core Node: 108)** 
User 349	**Personalized Community and Max Dispersion Node(Core Node: 349)** 

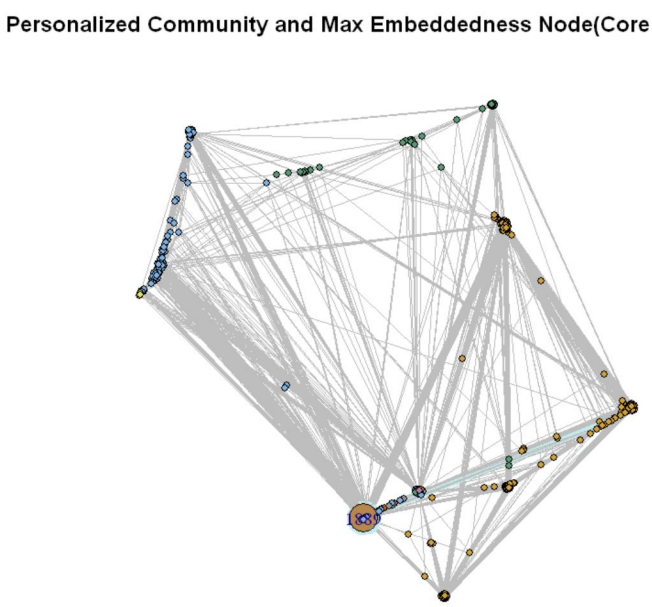
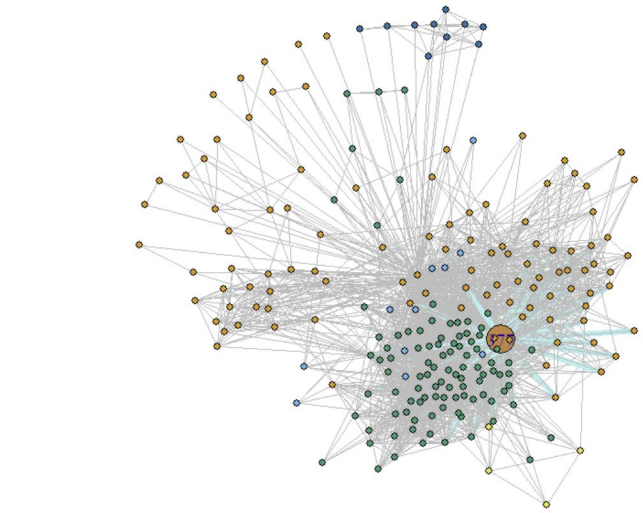
User 484	
User 1087	

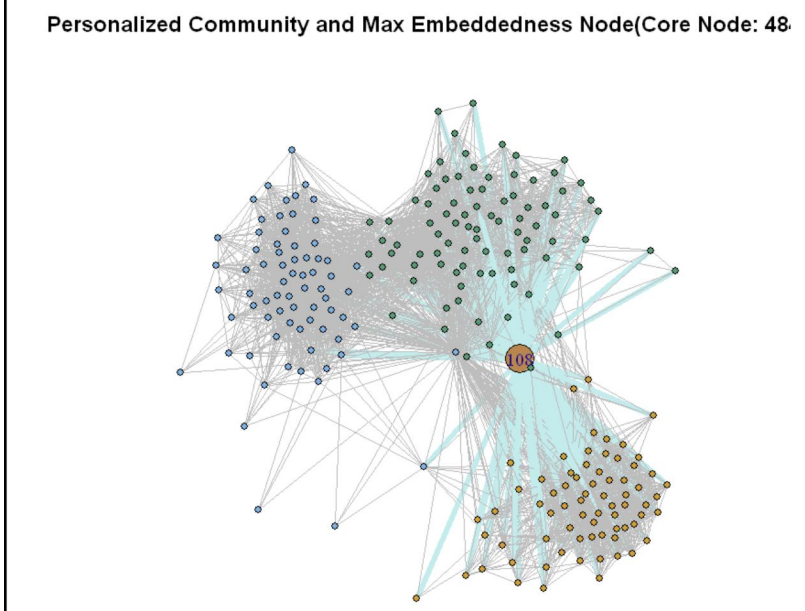
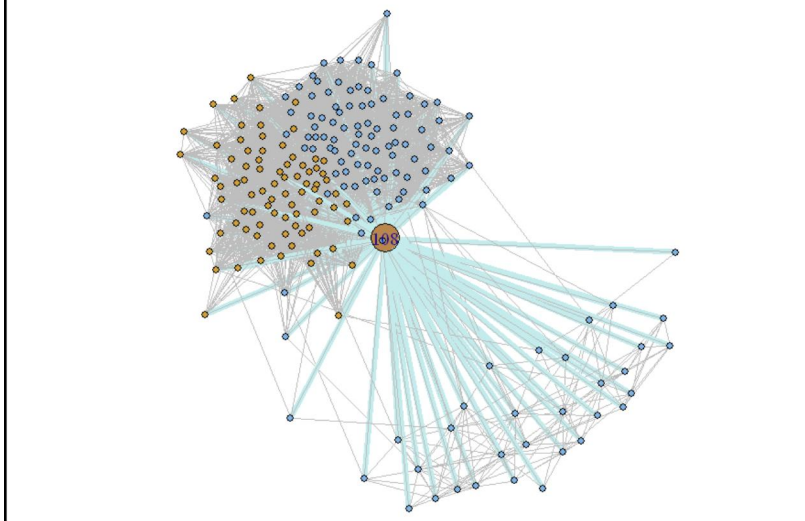
Question 14

Boundaries for communities are not drawn, they are labeled with different colors. Core and max nodes are marked with a big circle. Edges of max nodes are colored with turquoise.

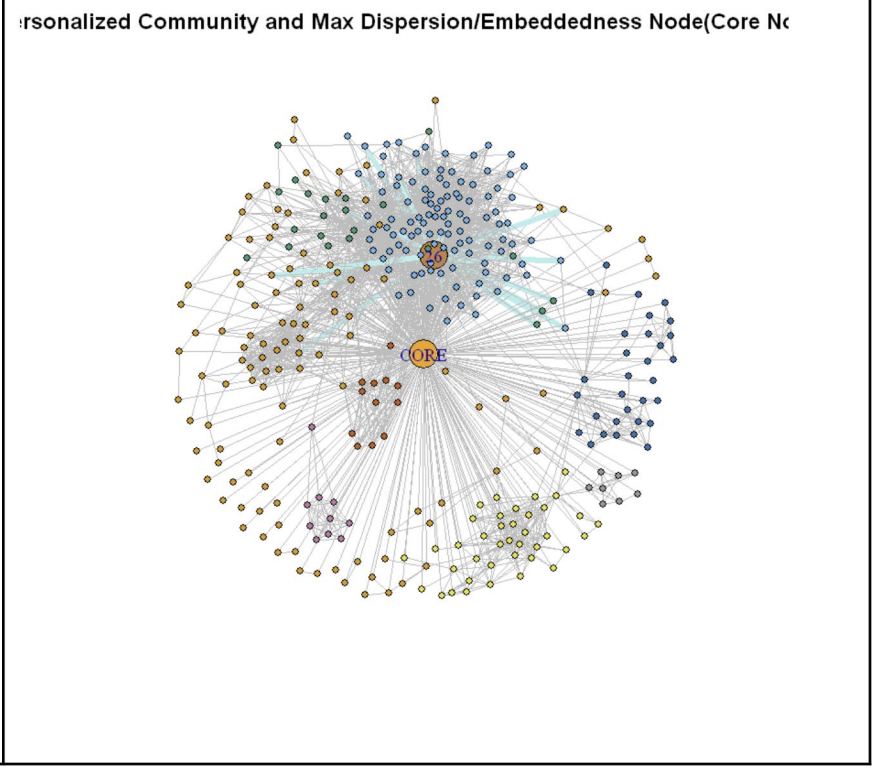
	Max Embeddedness ID
User 1	57
User 108	1889
User 349	377
User 484	108
User 1087	108

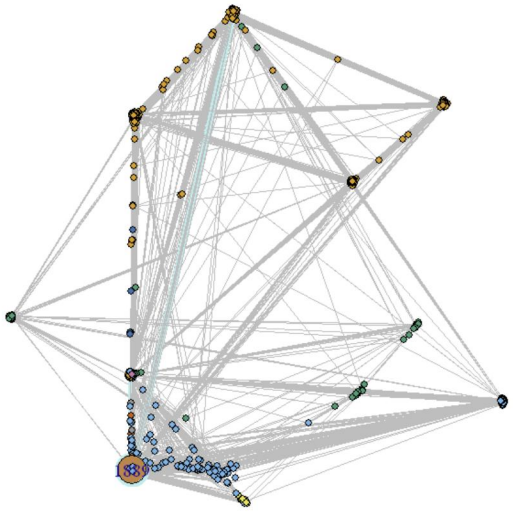
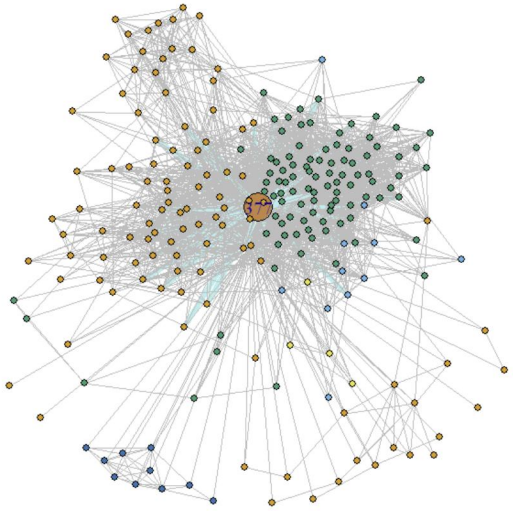
	Max Embeddedness Graph
User 1	

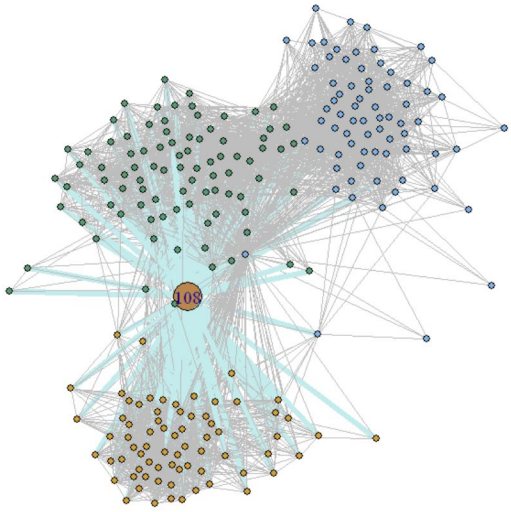
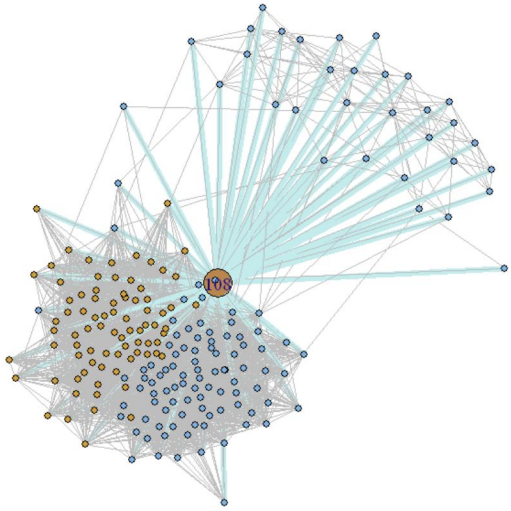
User 108	
User 349	

User 484	
User 1087	

	Max $\frac{Dispersion}{Embeddedness}$ ID
User 1	26
User 108	1889
User 349	377
User 484	108
User 1087	108

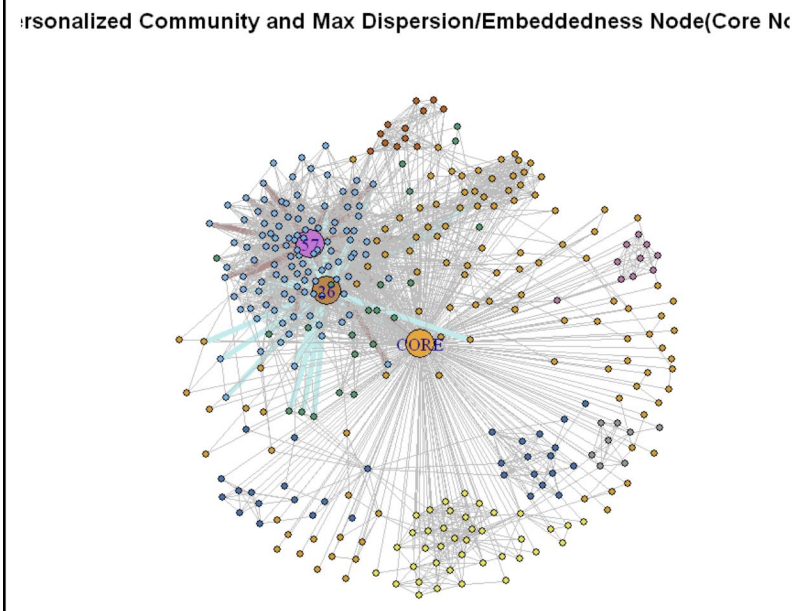
	Max $\frac{Dispersion}{Embeddedness}$ Graph
User 1	

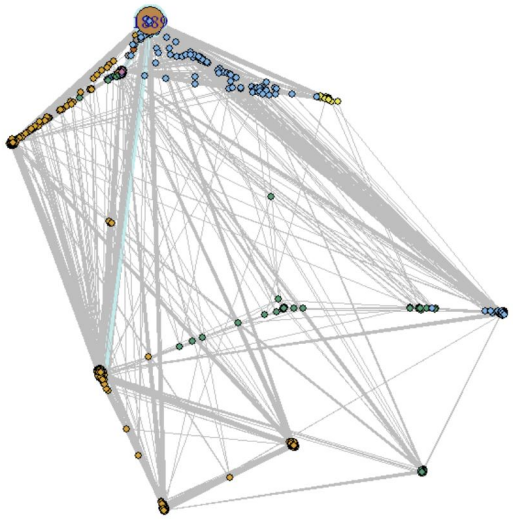
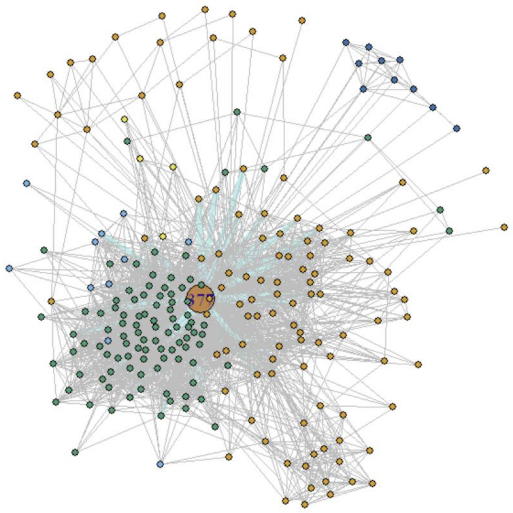
User 108	sonalized Community and Max Dispersion/Embeddedness Node(Core No 
User 349	sonalized Community and Max Dispersion/Embeddedness Node(Core No 

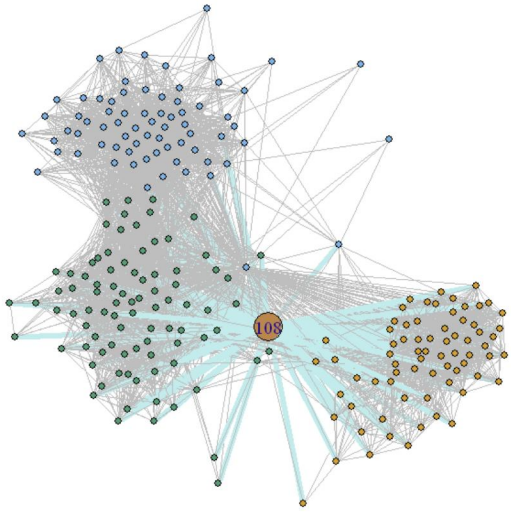
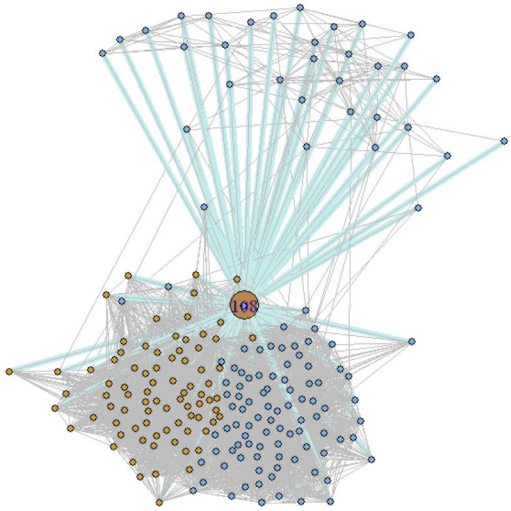
User 484	sonalized Community and Max Dispersion/Embeddedness Node(Core Noc 
User 1087	sonalized Community and Max Dispersion/Embeddedness Node(Core Nod 

	Max $\frac{Dispersion}{Embeddedness}$ ID	Max Embeddedness ID
User 1	26	57
User 108	**1889**	**1889**
User 349	**377**	**377**
User 484	**108**	**108**
User 1087	**108**	**108**

Note that, 4 max nodes have the same ratio and embeddedness. Max Embeddedness Node's edges are colored with red and node is colored with purple.

	Max $\frac{Dispersion}{Embeddedness}$ and Max Embeddedness Combined Graph
User 1	

User 108	**sonalized Community and Max Dispersion/Embeddedness Node(Core No** 
User 349	**sonalized Community and Max Dispersion/Embeddedness Node(Core No** 

User 484	sonalized Community and Max Dispersion/Embeddedness Node(Core No 
User 1087	›mmunity and Max Dispersion/Embeddedness Node + Embeddedness No 

1.3 Core node’s personalized network **10 / 10**

✓ **- 0 pts** *Correct*

Question 15

	Max $\frac{Dispersion}{Embeddedness}$ ID	Max Embeddedness ID	Max Dispersion ID
User 1	*26*	*57*	*57*
User 108	***1889***	***1889***	***1889***
User 349	***377***	***377***	***377***
User 484	***108***	***108***	***108***
User 1087	***108***	***108***	***108***

In a personalized network, **high embeddedness** refers to the degree to which an individual is connected to others within their network. More specifically, it refers to the number of connections or ties an individual has with others in the network, as well as the strength and frequency of those connections.When an individual has high embeddedness in their personalized network, it means that they are well-connected to many other individuals in the network, and they likely have frequent and strong interactions with those individuals. This can have a number of potential benefits, such as increased access to information, resources, and social support. However, high embeddedness can also have some potential drawbacks, such as being more susceptible to the spread of rumors or misinformation within the network, or feeling pressure to conform to the expectations or norms of the group. Overall, the effects of high embeddedness in a personalized network will depend on a variety of factors, such as the specific network in question and the goals and needs of the individual.

Embeddedness of a node can be easily computed with expression in Question 11. If a non-core node has a high degree, that node also shares a high number of mutual friends with the core node. In facebook terms, non-core nodes that have the maximum embeddedness and core nodes are best friends forever(BFF). When the edges of nodes inside the personalized network increase, those nodes have higher embeddedness value. Also, note that probability of having connections increases when the number of nodes inside the network increases. This can be summarized by social people having larger graphs, whereas asocial people have very small graphs. From the plots in Question 14, we can see that nodes that have the highest embeddedness have many edges incident to that node and share many mutual friends with the core node. But, embeddedness does not indicate a strong relationship between nodes. Nodes can share a high number of mutual friends but it does not give us information about how strong these relationships are.

In a personalized network, **high dispersion** refers to the degree to which an individual's connections or ties are not connected to one another. More specifically, it refers to the lack of connections or ties between the different individuals or groups that an individual is connected to within their network. When an individual has high dispersion in their personalized network, it means that their connections or ties are spread out among different individuals or groups that are not necessarily connected to one another. This can have both potential benefits and drawbacks. On the one hand, high dispersion can provide an

individual with access to a diverse range of information, resources, and perspectives from different parts of their network. On the other hand, high dispersion can also result in a lack of cohesion or unity within the network, as individuals and groups may not be connected or working together towards common goals. This can make it more difficult for individuals to build strong relationships or establish trust within the network. Overall, the effects of high dispersion in a personalized network will depend on a variety of factors, such as the specific network in question and the goals and needs of the individual.

If a node shares a high number of mutual friends with the core node and distances between each pair of mutual friends are high, the dispersion will be also high. Note that the number of mutual friends is the embeddedness of the node, therefore dispersion and embeddedness are highly correlated measures. However, when networks are dispersed meaning there is no direct connection between mutual friends, dispersion gets bigger. In Question 13, we can easily observe that the nodes with maximum dispersion have many connections within the network. Also, those nodes are connected to different communities, since the distances between max nodes and different communities nodes are high. This phenomenon can be easily seen at the core node 484 and 1087.

Dispersion divided by embeddedness is a ratio used to describe the structure of a personalized network. It is calculated by dividing an individual's level of dispersion by their level of embeddedness in the network. This ratio provides insight into the extent to which an individual is connected to diverse sources of information and perspectives within their network, while also maintaining a certain degree of embeddedness or connectedness to others. A higher ratio indicates that an individual is more dispersed in their network relative to their embeddedness, while a lower ratio indicates the opposite. A high dispersion divided by embeddedness ratio suggests that an individual is more connected to diverse sources of information and perspectives within their network, but may have a lower level of connectedness to others. Conversely, a low dispersion divided by embeddedness ratio suggests that an individual may be more embedded or connected to others within their network. As with the embeddedness divided by dispersion ratio, the interpretation of the dispersion divided by embeddedness ratio will depend on the specific context and goals of the individual in question.

High ratio is obtained for 484 and 1087 networks since the maximum nodes are dispersed over the network and have many connections with other communities. Other networks 1,108 and 349 have smaller ratio, thus they make few connections with other communities while having connections within the community. Ratio can be used to distinguish community connections.

Friend recommendation in personalized networks

Question 16

N_r contains nodes 31, 53 , 75, 90, 93, 102, 118, 133, 134, 136, 137.

Therefore, $|N_r| = 11$. Note that, these ids are new ids obtained when constructing the graph of a personalized network of core node 415. These nodes have degree 24 and will be used in the next part for friend recommendation.

Question 17

For each node in $|N_r|$, we have calculated the accuracy of the friend recommendation algorithm 10 times for each measure.

The algorithm breakdown is as follows, suppose Node i corresponds to User i:

1) Find the neighbors of Node i in the personalized network of core node 415
2) Delete a neighbor from the network with probability 0.25 and add that node to the removed list
3) Find the remaining neighbors and add the current node id to that set.
4) Find possible recommendations by taking the set difference between modified network and remaining neighbors.
5) Using the measure of your choice, calculate the score of the possible new neighbor.
6) Take the best removed list's size of nodes and calculate the accuracy for that step and user.
7) Accuracy is calculated as the size of the intersection between the set obtained at step 6 and removed list divided by the size of the removed list.

These 7 steps are calculated for each user 10 times using a single measure function. Every result is averaged across steps and then across users to obtain a single result. We have skipped iterations that have resulted in the removed list's size = 0.

Measure: Common Neighbors	**Accuracy**
User 31	0.29905
User 53	1.00000
User 75	0.87143
User 90	0.82476
User 93	0.41988
User 102	1.00000
User 118	0.84167
User 133	1.00000
User 134	1.00000
User 136	0.91893
User 137	0.98571
Average accuracy across users	**0.83286**

Measure: Jaccard	**Accuracy**
User 31	0.09592
User 53	0.95893
User 75	0.90071
User 90	0.80881
User 93	0.47036
User 102	0.98571
User 118	0.89389
User 133	0.96333
User 134	0.98333
User 136	0.89655
User 137	0.92500
Average accuracy across users	**0.80750**

Measure: Adamic Adar	**Accuracy**
User 31	0.30345
User 53	1.00000
User 75	0.88893
User 90	0.82960
User 93	0.43811
User 102	1.00000
User 118	0.83671
User 133	1.00000
User 134	1.00000
User 136	0.87405
User 137	0.96905

Average accuracy across users	**0.83090**

Measure	**Accuracy Score**
Common Neighbor	0.83286
Jaccard	0.80750
Adamic Adar	0.83090

In result, the best recommendation algorithm is using the common neighbor measure, the worst recommendation algorithm is using the Jaccard measure. However, accuracies are very close to each other, therefore one algorithm might be better than the other when used a different seed or number of iterations for each user.

1.4 Friend recommendation in personalized networks **10 / 10**

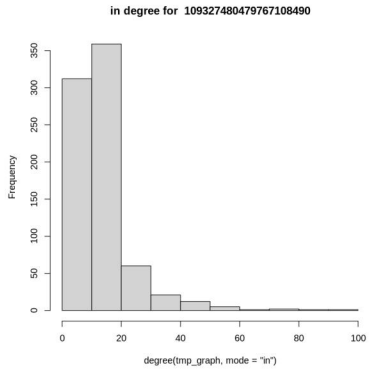
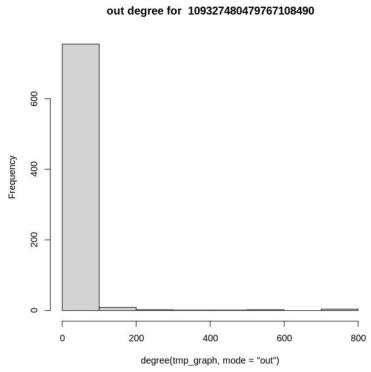
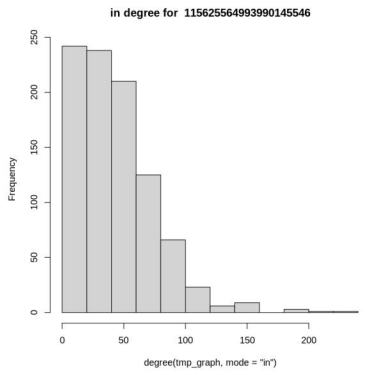
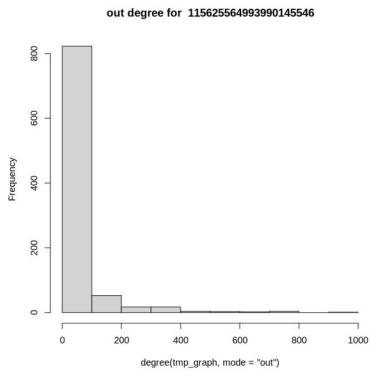
✓ **- 0 pts** *Correct*

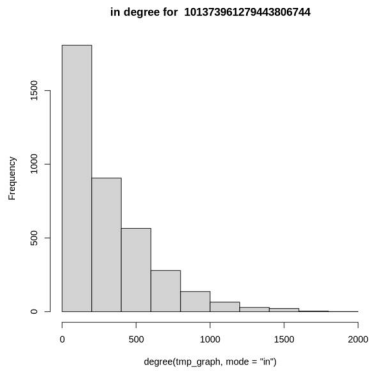
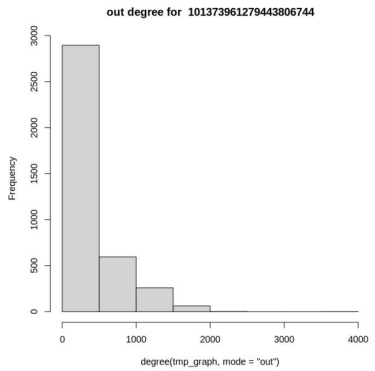
Google+ Network

Question 18

There are 57 personal networks with users who have more than 2 circles.

Question 19

Node ID	In-degree Degree Distribution	Out-degree degree Distribution
109—-490		
114—546		

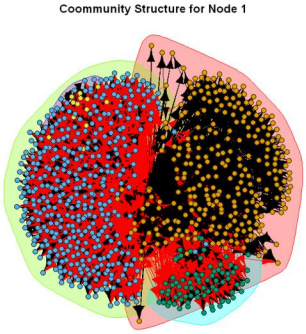
101—744		

The in-degree distributions differ in that UID ending in 744 and 490 decay a lot quicker as degree increases, whereas, UID ending in 546 gradually decreases in frequency as degree increase meaning this node has a lot of users with high in-degrees, which means there are generally a larger amount of people that are followed by others. In other words, the members of this network have similar followings to each other. For the two UIDs of 744 and 490, there is a smaller group of people in which others are connected to. However, comparing 744 and 490, we see that 744 has overall higher degrees than 490, so amongst the small group of people with large followings, 744's people have larger following than the 490 group.

The out degree distributions for all of the specific nodes are similar. However, the UID 490 has the heaviest right-skew, whereas, the UID 744 has the least heavy skew to the right. The slow decay of out-degree distribution for UID 744 could suggest that the community clusters are not as distinct (low modularity)

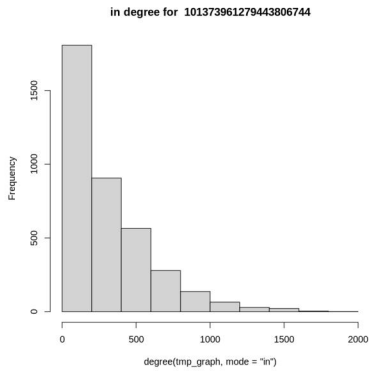
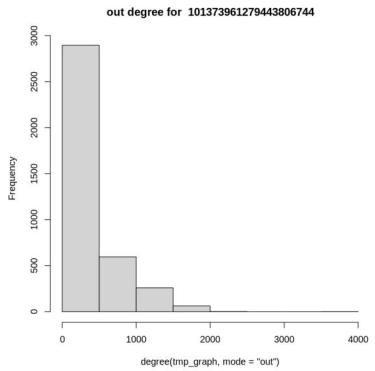
Community structure of personal networks

Question 20

Node ID	Modularity	Community Structure
109—-490	0.252765	

2.1 Data exploration **5 / 5**

✓ **- 0 pts** *Correct*

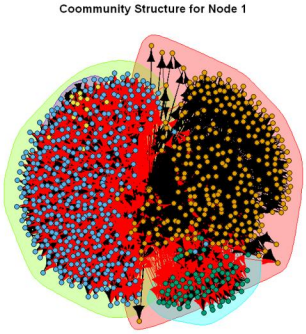
101—744		

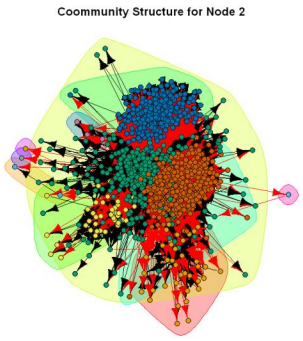
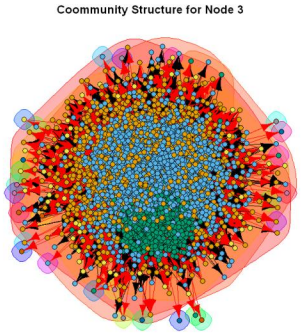
The in-degree distributions differ in that UID ending in 744 and 490 decay a lot quicker as degree increases, whereas, UID ending in 546 gradually decreases in frequency as degree increase meaning this node has a lot of users with high in-degrees, which means there are generally a larger amount of people that are followed by others. In other words, the members of this network have similar followings to each other. For the two UIDs of 744 and 490, there is a smaller group of people in which others are connected to. However, comparing 744 and 490, we see that 744 has overall higher degrees than 490, so amongst the small group of people with large followings, 744's people have larger following than the 490 group.

The out degree distributions for all of the specific nodes are similar. However, the UID 490 has the heaviest right-skew, whereas, the UID 744 has the least heavy skew to the right. The slow decay of out-degree distribution for UID 744 could suggest that the community clusters are not as distinct (low modularity)

Community structure of personal networks

Question 20

Node ID	Modularity	Community Structure
109—-490	0.252765	

114—546	0.319473	
101—744	0.191090	

The modularity score for the second personal network is the highest and the third personal network being the lowest. As we can see from the community structure, the second personal network has more distinct clusters than the third personal network.

We also made this same conclusion from looking at the degree distribution graph that the third personal network would have the lowest modularity score because of the slow decay of out-degree distribution.

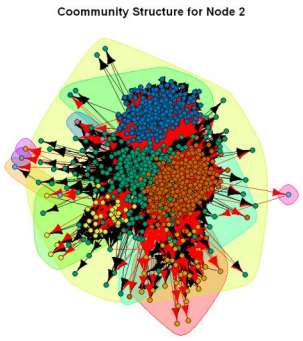
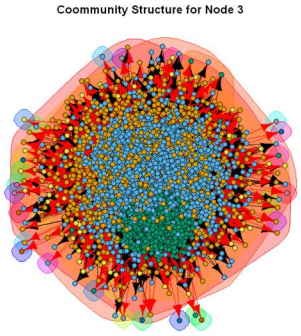
Question 21

The higher the homogeneity is, this means that all of the clusters within a network contains members of a single circle. In other words, if all the members of a cluster, K, share the same circle, C, then the homogeneity score is closer to 1. Perfect homogeneity: all the nodes in a particular cluster have the same circle (class).

The higher the completeness is, this means that all the members of a given cluster are members of the same community. In other words, if all the members in a circle, C, are all in the same cluster, K, then the completeness score is closer to 1. Perfect completeness: if all the nodes belonging to a particular class are clustered together.

2.2 Community structure of personal networks **5 / 5**

✓ **- 0 pts** *Correct*

114—546	0.319473	
101—744	0.191090	

The modularity score for the second personal network is the highest and the third personal network being the lowest. As we can see from the community structure, the second personal network has more distinct clusters than the third personal network.

We also made this same conclusion from looking at the degree distribution graph that the third personal network would have the lowest modularity score because of the slow decay of out-degree distribution.

Question 21

The higher the homogeneity is, this means that all of the clusters within a network contains members of a single circle. In other words, if all the members of a cluster, K, share the same circle, C, then the homogeneity score is closer to 1. Perfect homogeneity: all the nodes in a particular cluster have the same circle (class).

The higher the completeness is, this means that all the members of a given cluster are members of the same community. In other words, if all the members in a circle, C, are all in the same cluster, K, then the completeness score is closer to 1. Perfect completeness: if all the nodes belonging to a particular class are clustered together.

Question 22

Node ID	Homogeneity	Completeness
109—-490	0.8518851	0.3298739
114—546	0.4518903	-3.423962
101—744	0.003866707	-1.504248

Node 1:
The higher the homogeneity score is to 1, the more members of a cluster, K, share the same circle, C. The lower completeness score means that not all members in a circle belong to the same community. However, this score is positive, so it indicates that some users that belong to the same circle have not been classified into the same community.

Node 2:
This node has lower homogeneity than node 1 implying that around half of members of a given cluster share the same circle. Some users in the same cluster are in different circles. The negative completeness score indicates that there are members in circles not in any clusters (i.e. $H(K) < H(K|C)$

Node 3:
The homogeneity score is almost 0 meaning that users from completely different circles have been assigned the same community indicating there is high mismatch. The completeness score is also negative meaning that some users in circles have not been assigned a cluster.

Cora Dataset

Question 23

In this section we use Graph Convolutional Networks to classify the papers from the dataset using the text features and edge connection in the graph.

2.3 Circles and communities **10 / 10**

✓ **- 0 pts** *Correct*

Question 22

Node ID	Homogeneity	Completeness
109—-490	0.8518851	0.3298739
114—546	0.4518903	-3.423962
101—744	0.003866707	-1.504248

Node 1:
The higher the homogeneity score is to 1, the more members of a cluster, K, share the same circle, C. The lower completeness score means that not all members in a circle belong to the same community. However, this score is positive, so it indicates that some users that belong to the same circle have not been classified into the same community.

Node 2:
This node has lower homogeneity than node 1 implying that around half of members of a given cluster share the same circle. Some users in the same cluster are in different circles. The negative completeness score indicates that there are members in circles not in any clusters (i.e. $H(K) < H(K|C)$

Node 3:
The homogeneity score is almost 0 meaning that users from completely different circles have been assigned the same community indicating there is high mismatch. The completeness score is also negative meaning that some users in circles have not been assigned a cluster.

Cora Dataset

Question 23

In this section we use Graph Convolutional Networks to classify the papers from the dataset using the text features and edge connection in the graph.

We use the following structure and hyperparameters, with the processing layers following the same specs. Training was run with a batch size of 120 and for 1000 epochs or when accuracy stagnated for 500 epochs.

```
Model: "gnn_model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 preprocess (Sequential)     (2708, 64)                97508

 graph_conv1 (GraphConvLayer  multiple                 13184
 )

 graph_conv2 (GraphConvLayer  multiple                 13184
 )

 postprocess (Sequential)    (2708, 64)                4416

 logits (Dense)              multiple                  455

=================================================================
Total params: 128,747
Trainable params: 124,985
Non-trainable params: 3,762
_________________________________________________________________
```

Hidden units	20
Learning rate	.008
Dropout rate	.81
Graph Conv Layer Activation	relu

We achieve an accuracy of 66.2% , suggesting that the text features can be utilized fairly well to predict the class of the papers.

Question 24

For this section we implemented the Node2Vec algorithm in order to extract structural features of the graph and its nodes. Node2Vec learns representations of the nodes through by taking biased random walks and using a model to extract embeddings for each node in the graph. The added dimensionality reduction can make the structural representation more efficient for use in training. When classifying using Node2Vec with a support vector, classifiers were able to achieve a test accuracy of **70.8%**.

When trained purely on the 1433-dimensional text features the classifier produced a test accuracy of **55.4%**. We see that classifying with just Node2Vec embeddings outperformed the text features by a

3.1 Idea 1 **15 / 15**

✓ **- 0 pts** *Correct*

We use the following structure and hyperparameters, with the processing layers following the same specs. Training was run with a batch size of 120 and for 1000 epochs or when accuracy stagnated for 500 epochs.

```
Model: "gnn_model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 preprocess (Sequential)     (2708, 64)                97508

 graph_conv1 (GraphConvLayer  multiple                 13184
 )

 graph_conv2 (GraphConvLayer  multiple                 13184
 )

 postprocess (Sequential)    (2708, 64)                4416

 logits (Dense)              multiple                  455

=================================================================
Total params: 128,747
Trainable params: 124,985
Non-trainable params: 3,762
_________________________________________________________________
```

Hidden units	20
Learning rate	.008
Dropout rate	.81
Graph Conv Layer Activation	relu

We achieve an accuracy of 66.2% , suggesting that the text features can be utilized fairly well to predict the class of the papers.

Question 24

For this section we implemented the Node2Vec algorithm in order to extract structural features of the graph and its nodes. Node2Vec learns representations of the nodes through by taking biased random walks and using a model to extract embeddings for each node in the graph. The added dimensionality reduction can make the structural representation more efficient for use in training. When classifying using Node2Vec with a support vector, classifiers were able to achieve a test accuracy of **70.8%**.

When trained purely on the 1433-dimensional text features the classifier produced a test accuracy of **55.4%**. We see that classifying with just Node2Vec embeddings outperformed the text features by a

considerable amount. This suggests that the structure of the graph contains more information about the class of a node than its word features.Two papers researching a topic from two different angles may share a lot of words in common but won't align in class. If one cites another, however, often to build upon their research, it may be much more likely for them to be in the same class.

We then combined the features, appending along the column space. When train on this combined set of features, we obtained a test accuracy of **74.6%**. Here we see this best accuracy from this set. This may be as each feature set captures different niches of the dataset.

The best accuracy was obtained by combining the two feature sets, which was **74.6%**.

Question 25

a)

For this third section we utilize a PageRank approach. We took 20 seed seed documents for each class and performed random walks with varying teleportation probabilities that would teleport the walker to a node of the same class. The probabilities chosen were 0, .1, and .2.

To obtain the results we ran the PageRank only on the GCC for each node and did 1000 random walks. We recorded the class-wise visitation frequency for the unlabeled nodes and the predicted class was the class that led to the maximum visits to that node.

For teleport probability at 0.

	Precision	Recall	F1-score	Support
Case Based	.78	.72	.75	285
Genetic Algorithms	.85	.95	.90	406
Neural Networks	.85	.59	.69	726
Probabilistic Methods	.77	.77	.77	379
Reinforcement Learning	.66	.85	.75	214
Rule Learning	.42	.90	.57	131
Theory	.62	.60	.61	344

Accuracy: 72.9%

The accuracy with the teleport probability at 0 was quite good, although we see some relatively low F1 scores for the Rule learning and Theory classes.

3.2 Idea 2 **15 / 15**

✓ **- 0 pts** *Correct*

considerable amount. This suggests that the structure of the graph contains more information about the class of a node than its word features.Two papers researching a topic from two different angles may share a lot of words in common but won't align in class. If one cites another, however, often to build upon their research, it may be much more likely for them to be in the same class.

We then combined the features, appending along the column space. When train on this combined set of features, we obtained a test accuracy of **74.6%**. Here we see this best accuracy from this set. This may be as each feature set captures different niches of the dataset.

The best accuracy was obtained by combining the two feature sets, which was **74.6%**.

Question 25

a)

For this third section we utilize a PageRank approach. We took 20 seed seed documents for each class and performed random walks with varying teleportation probabilities that would teleport the walker to a node of the same class. The probabilities chosen were 0, .1, and .2.

To obtain the results we ran the PageRank only on the GCC for each node and did 1000 random walks. We recorded the class-wise visitation frequency for the unlabeled nodes and the predicted class was the class that led to the maximum visits to that node.

For teleport probability at 0.

	Precision	Recall	F1-score	Support
Case Based	.78	.72	.75	285
Genetic Algorithms	.85	.95	.90	406
Neural Networks	.85	.59	.69	726
Probabilistic Methods	.77	.77	.77	379
Reinforcement Learning	.66	.85	.75	214
Rule Learning	.42	.90	.57	131
Theory	.62	.60	.61	344

Accuracy: 72.9%

The accuracy with the teleport probability at 0 was quite good, although we see some relatively low F1 scores for the Rule learning and Theory classes.

For teleport probability at .1

	Precision	Recall	F1-score	Support
Case Based	.74	.72	.73	285
Genetic Algorithms	.86	.90	.88	406
Neural Networks	.83	.60	.69	726
Probabilistic Methods	.75	.77	.76	379
Reinforcement Learning	.65	.86	.74	214
Rule Learning	.50	.89	.64	131
Theory	.61	.64	.63	344

Accuracy: 72.9%

Although the overall accuracy remained the same, we see improvement in the F1 score for the classes that no teleportation struggled with. This may suggest that these classes are more scattered and not necessarily clustered, thus being slightly aided through teleportation.

For teleport probability at .2

	Precision	Recall	F1-score	Support
Case Based	.67	.74	.71	285
Genetic Algorithms	.85	.87	.86	406
Neural Networks	.82	.59	.69	726
Probabilistic Methods	.72	.75	.74	379
Reinforcement Learning	.65	.84	.73	214
Rule Learning	.53	.84	.65	131
Theory	.60	.62	.61	344

Accuracy: 71.6%

We see a dip in accuracy as we increase the teleportation further, suggesting a ideal teleportation probability for this network. Perhaps it cannot explore the network as much.

b)
For this section we modify the baseline transition probability of the random walks such that they are proportional to the cosine similarity between each others text features.

For teleport probability at 0

	Precision	Recall	F1-score	Support
Case Based	.79	.72	.75	285
Genetic Algorithms	.84	.93	.88	406
Neural Networks	.86	.60	.71	726
Probabilistic Methods	.77	.76	.77	379
Reinforcement Learning	.66	.87	.75	214
Rule Learning	.44	.90	.59	131
Theory	.59	.60	.59	344

Accuracy = 72.9%

The accuracy does not seem to show much improvement except perhaps a smoother spread of F1 scores, suggesting that the original algorithm works well already.

For teleport probability at 0.1

	Precision	Recall	F1-score	Support
Case Based	.76	.72	.71	285
Genetic Algorithms	.84	.89	.86	406
Neural Networks	.84	.60	.70	726
Probabilistic Methods	.74	.78	.76	379
Reinforcement Learning	.65	.87	.74	214
Rule Learning	.49	.85	.62	131
Theory	.62	.64	.63	344

Accuracy = 73.1%

Here we see the highest accuracy so far, suggesting that the cosine similarity made an improvement alongside the teleportation.

For teleport probability at 0.2

	Precision	Recall	F1-score	Support
Case Based	.63	.69	.66	285
Genetic Algorithms	.85	.88	.86	406
Neural Networks	.83	.58	.68	726
Probabilistic Methods	.72	.75	.74	379
Reinforcement Learning	.64	.83	.72	214
Rule Learning	.49	.82	.61	131
Theory	.59	.60	.59	344

Accuracy: 70.5%

We see an expected dip in the accuracy.

3.3 Idea 3 **16 / 20**

✓ **- 4 pts** *part b with p = 0 is wrong*