Lab assignment 2: Social network analysis and modeling

By Jiamin Ou

In these exercises, you will analyze various social networks and model the diffusion process in a network using the packages in RStudio, such as the *igraph*, *network* and *intergraph*. If you prefer Python, you can use Python to finish all or some of the assignments. You can find these packages in Python with same or similar names. Noted that all the referenced codes provided here are based in R, and tutors for the social network part are not necessarily familiar with Python.

Apart from this, you follow the principles of group work as during the first four labs.

In the following text, explanation is plain text, instructions are underlined, questions to answer are labelled, and R code snippets are written in `different font`. The number of points available for each questions gives an idea of the expected depth of your answers.

Note on submission file:

You will submit the answer in a PDF file via Blackboard before the deadline of 9 April 2023, 23:59. Each group only need to submit a file, with clear indications of your names and student numbers. For most questions, you don't need to submit the codes. Exceptions are Question 11, 14, 18, 19 and 20, which you should submit the codes along with your answer.

Installation:

If you do not already have Rstudio installed, first download and install R: https://cran.rstudio.com/; Then download and install RStudio: https://posit.co/download/rstudio-desktop/

Exercise one: Analyzing an offline and online social networks

We will start with two social networks from the real world: 1) an offline social network for friendship relations between students in a high school 2) an online social network for Facebook friends.

Download the "Highschool_network_att.csv", "Highschool_network_att.csv", and "Facebook_network_att.csv", "Facebook_network_edge.csv" and `Facebook_edge.csv` from BlackBoard. Import the `.csv` data into `R` and build the networks.

Build network objects

```
#call library
library(igraph)
library(RColorBrewer)
library(visNetwork)

#datainput
highschool_edge<-read.csv("Highschool_network_edge.csv",header=FALSE)
```

```
highschool_att<-read.csv("Highschool_network_att.csv",header = TRUE)
facebook_edge<-read.csv("Facebook_network_edge.csv",header=FALSE)
facebook_att<-read.csv("Facebook_network_att.csv",header = TRUE)

#build high school network
highschool_nodes<-data.frame(name=as.character(highschool_att$NodeID),
                  gender=as.character(highschool_att$Gender),
                  hall=as.character(highschool_att$Hall))
highschool_edges<-data.frame(from=c(as.character(highschool_edge[,1])),
                  to=c(as.character(highschool_edge[,2])))
Highschool<-graph_from_data_frame(highschool_edges,directed =
FALSE,vertices = highschool_nodes)
co <- components(Highschool)
Highschool <- induced.subgraph(Highschool, which(co$membership ==
which.max(co$csize))) #use only the largest component for analysis
summary(Highschool)

#build facebook network
facebook_nodes<-data.frame(name=as.character(facebook_att$NodeID))
facebook_edges<-data.frame(from=c(as.character(facebook_edge[,1])),
                           to=c(as.character(facebook_edge[,2])))
Facebook<-graph_from_data_frame(facebook_edges,directed = FALSE,vertices =
facebook_nodes)
summary(Facebook)
```

Discuss within your group, what are the meanings of nodes and links in these two networks.

Node-level centrality measures

In the lecture, we introduced a few metrics to measure node-level centrality: degree, betweenness, closeness, eigenvector.

Question 1 (3 points):

- Find out the node ID of a) highest degree b) highest betweenness c) highest closeness and d) highest eigenvector in the Highschool network;
- Highlight the above nodes in the Highschool network;
- Explain why these metrics identify the same node or different nodes as the most central one.

```
#function to calculate centrality metrics
degree(Highschool, mode = "all")
closeness(Highschool, normalized = TRUE)
betweenness(Highschool, directed = FALSE, normalized = TRUE)
eigen_centrality(Highschool)

#function to visualize the network (with interaction)
set.seed(100)
Highschool_interactive_layout<-
visNetwork(data.frame(id=V(Highschool)$name), highschool_edges, main =
"Highschool",submain="Can zoom in/out to check the IDs and ties") %>%
  visIgraphLayout(layout = "layout_nicely",smooth =  FALSE) %>%
```

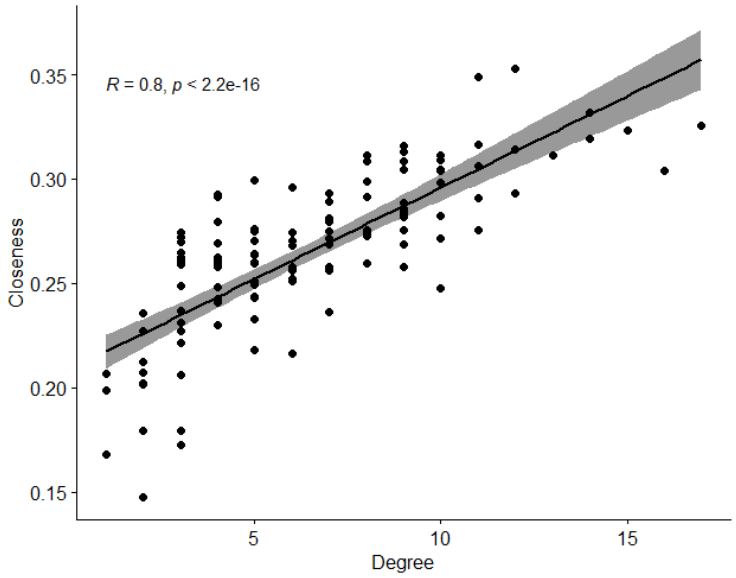
```
  visNodes(shape="circle",label = TRUE) %>%
  visOptions(highlightNearest = list(enabled = T, hover = T),
nodesIdSelection = T)

Highschool_interactive_layout
```

Question 2 (5 points):

- Study the correlations between a) degree and betweenness, b) degree and closeness, c) degree and eigenvector *for all the nodes* in the Highschool network;
- Study the correlations between a) degree and betweenness, b) degree and closeness, c) degree and eigenvector *for all the nodes* in the Facebook network;
- From the above results, how well do different metrics correlate with each other? Which centrality metric will you use and why?

Please provide *more than just a correlation coefficient* to answer Question 2. You are suggested to study the correlations by developing a scatter plot as below, in which every dot represents a node in the network, with xlab as degree and ylab as closeness.



Using network-level measures to explain degrees of separation

In the lecture, we mentioned the small-world experiment comprised several experiments conducted by Stanley Milgram and other researchers examining the average path length for social networks of people in the United States. The research was groundbreaking in that it suggested that human society is a small-world-type network characterized by short path-lengths. The notion of six degrees of separation later on grew from Malcolm Gladwell's *The Tipping Point* that states that everyone in the world is at most 6 connections away from everyone else. On top of that, he claimed that

“Six degrees of separation doesn’t [just] mean that everyone is linked to everyone else in just six steps. It means that a very small number of people are linked to everyone else in a few steps, and the rest of us are linked to the world through those special few.”

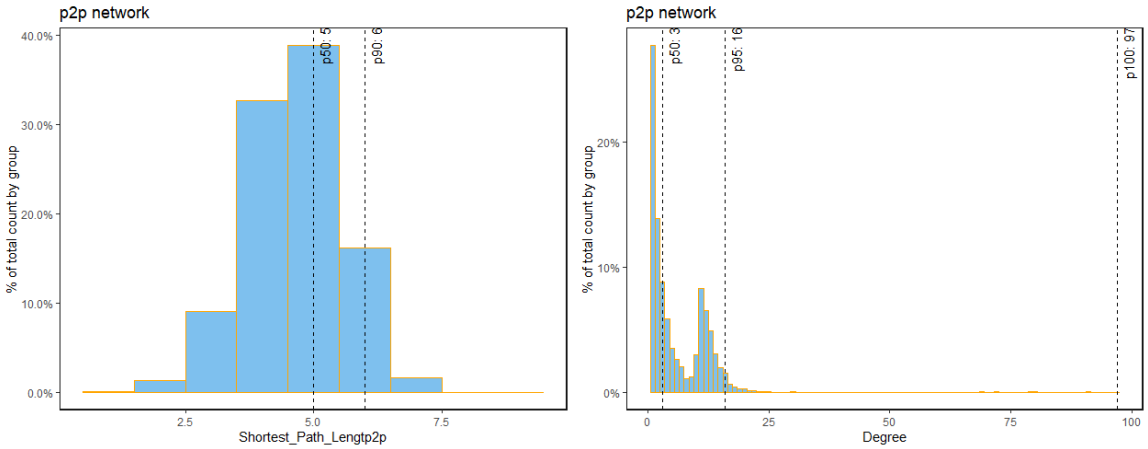
Question 3 (5 points):

- For both the Highschool and Facebook networks, calculate the shortest path lengths between every pair of two nodes. How many percentage of nodes can be reached within 6 path lengths? Does “six degree of separation” apply to each network?
- Study the degree distribution of these two networks, are they similar? Then use degree distribution to explain the degree of separation you answered above.

You can calculate the shortest path lengths between every pair of two nodes using *igraph* package:

```
distances(
  Highschool,
  v = V(Highschool),
  to = V(Highschool),
  mode = c("all", "out", "in"),
  weights = NULL) #Shortest path lengths between every pair of two nodes in the network
```

You can consider to use histogram with percentile to study the shortest path lengths and the degree distribution:



(Note: the vertical reference line shows the values of shortest path length or the degree at certain percentiles.)

Meso-scale structure

The above exercises use the distribution of shortest path lengths, and the distribution of degree, to study how well nodes in the network are connected to each other. Another important feature of a network is its meso-scale structure—grouping of nodes based on their distinctive interaction patterns, or, nodes with similar properties are grouped together instead

of being treated individually. Discuss within your group, why meso-scale structure is important in our understanding of a network?

Now you will analyze the meso-scale structure of the Highschool network. First, check out the node attributes of Highschool network. You can find 1) the gender and 2) the residential hall of each student. A hypothesis can be formed as: If two students share some common characteristics, such as living in the same residential hall or of the same gender, their chance of being friends are higher.

Test the above hypothesis by the following steps (Question 4, 4 points):

1) Visualize the network and color the nodes by gender and residential hall, respectively.
2) Build 8 subgraphs of the original network according to gender and residential hall: 1 subgraph for female student, 1 subgraph for male student, 1 subgraph for students with unknown gender, and 5 subgraphs for students living in residential hall from 1501 to 1505, respectively.
   For example, to build a subgraph of all female students, you should keep all the nodes of female students and the edges between them. Other nodes and edges are removed.
3) Study the edge density of all the subgraph and compare them to the edge density of the original network. What is your conclusion for the hypothesis?

Codes to visualize the network and calculate subgraph density:

```
## visualize the network by gender###
library(RColorBrewer)
coul  <- brewer.pal(length(unique( V(Highschool)$gender)), "Set2")
my_color <- coul[as.numeric(as.factor(V(Highschool)$gender))]
set.seed(10)
plot(Highschool, vertex.color = my_color,
     vertex.size=5,
     layout=layout.fruchterman.reingold(Highschool),vertex.label=NA,
     main="Highschool network by gender")
legend("bottomleft", legend=levels(as.factor(V(Highschool)$gender)) ,col = coul , bty = "n", pch=20 , pt.cex = 1.5, cex = 1.5, horiz = FALSE, inset = c(0.1, 0.1))


#introduce subgraph by gender, calculate their edge densities
group <- as.factor(unique(V(Highschool)$gender))
sapply(levels(group), function(x) {
  y <- induced_subgraph(Highschool, which(V(Highschool)$gender==x))
  paste0("Density for ", x, " friends is ", edge_density(y))
})
```

To better understand the meso-scale structure, we will study community detection algorithm. One important stand of community detection algorithm is based on ***modularity***, which tries to maximize the difference between the actual number of edges in a community and the expected number of edges in the community. However optimizing modularity in a network is NP-hard, therefore have to use heuristics.

Question 5 (4 points):

1) Calculate the modularity of the Highschool network if community is merely identified by a) gender and b) residential hall, respectively.
2) Search the Louvain Community Detection and explain the algorithm in your own words.
3) Use the Louvain Community Detection to identify communities in the Highschool network. Compare the modularity value produced by the Louvain algorithm to those in 1), and explain the reasons for the differences.

Codes to customize community and calculate modularity:

```
### customize community by gender ###
genderCommunity<-V(Highschool)$gender
genderCommunity<-replace(genderCommunity,genderCommunity=="female",1)
genderCommunity<-replace(genderCommunity,genderCommunity=="male",2)
genderCommunity<-replace(genderCommunity,genderCommunity=="unknown",3)
genderCommunity<-as.numeric(genderCommunity)

gender.clustering <- make_clusters(Highschool, membership=genderCommunity)
modularity(gender.clustering)

### Louvain algorithm ###
Louv<-cluster_louvain(graph.object)
modularity(Louv)
```

Self-exercise: we did not cover all the network metrics in the above exercise, such as local and global clustering coefficient, diameter and component. But some of them might appear in the exam. Within your own group, explain to each other the meaning and possible application of other network metrics that are not covered in the above exercise. Check if your understanding are aligned. Reach your tutor if you have unsolved issues.

Exercise two: Network formation models

Network relationships come in many shapes and sizes, and so there is no single model which encompasses them all. But over time, people do summarize some common paradigm that can be used to build a synthetic network. In the lecture, we mentioned three major architectures to build a synthetic network, which is Erdos-Renyi Random Graph Model, Small-world Random Graph Model, and Barabasi-Albert (BA) model.

Erdos-Renyi random network

If you know how many nodes are in your network, as well as the probability that any two of them are connected (i.i.d and random), you can generate an E-R random graph using

`sample_gnp()`. In `sample_gnp(n, p, directed = FALSE, loops = FALSE)`, the graph has 'n' vertices and for each edge the probability that it is present in the graph is 'p'.

If you know the number of nodes and the number of edges, you can use `sample_gnm()`. In `sample_gnm(n, m, directed = FALSE, loops = FALSE)`, the graph has 'n' vertices and 'm' edges, and the 'm' edges are chosen uniformly randomly from the set of all possible edges. This set includes loop edges as well if the loops parameter is TRUE.

Question 6 (3 points):

1) Develop three networks with the same number of vertices (n), but different probability (p); Name them as ER1, ER2, and ER3. Develop the plots of ER1, ER2 and ER3, describe how these three graphs look differently as *p* increase and explain why.
2) For a large n (e.g., $n=1000$), study the relation between clustering coefficient of the network and p, and explain the reason for such a relation. (You can use the function of `transitivity (graph.object)` to calculate clustering coefficient).

Small-world network

Next, let's move on to the small world model (Watts and Strotgatz model). It assumes that you know a certain number of persons (k) and that you are more likely to know your closest neighbors. The algorithm though more complicated than the Erdős-Rényi model's. We have 3 parameters. The number of the population (N), the number of close neighbors (k) and a rewiring probability (p).

Because this model generates some conglomerates of people knowing each other, it is really easy to be linked indirectly (and with a very few number of steps) with anyone in the map. This is why we call this kind of model a small world model.

The small-world model is built by introducing a rewiring probability to a regular lattice. Run the following code and discuss with your group mate how the rewiring probability changes the network structure:

```
Regular<-watts.strogatz.game(dim=1,size=300,nei=6, p=0)

plot(Regular, layout=layout.circle, vertex.label=NA, vertex.size=5, main=
"Network with zero rewiring probability ")

SW1<-watts.strogatz.game(dim=1,size=300,nei=6, p=0.001)

plot(SW1, layout=layout.circle, vertex.label=NA, vertex.size=5, main=
"Network with 0.001 rewiring probability ")

SW2<-watts.strogatz.game(dim=1,size=300,nei=6, p=0.01)

plot(SW2, layout=layout.circle, vertex.label=NA, vertex.size=5, main=
"Network with 0.01 rewiring probability ")

SW3<-watts.strogatz.game(dim=1,size=300,nei=6, p=0.1)

plot(SW3, layout=layout.circle, vertex.label=NA, vertex.size=5, main=
"Network with 0.1 rewiring probability ")
```

Question 7 (2 points):

Check the clustering coefficient and average path length of the Regular, SW1, SW2 and SW3. Describe the trend of clustering coefficient and average path length as *p* increase. Which graph does mimic the desirable attributes of a small world network?

You might realize not every value of p can return you a small-world network that you are looking for. Then a question arises as how can one find the range of p. In the Figure 2 of Watts and Strogatz (1998), it explains how can one decide the range of *p* by looking at the dynamics between path length and clustering coefficient.

Question 8 (5 points):

1) Start with a regular network of `size=300, nei=6`, first reproduce the Figure 2 of Watts and Strogatz (1998). Then provide the range of p which can turn this regular network (`size=300, nei=6`) into a small-world network.
2) Do you need to rewire significant amount of connections to make the network small-world-like?
3) In the paper of Watts and Strogatz (1998), they pointed out that the value of *p* has two important implications:

"The idealized construction above reveals the key role of short cuts. It suggests that the small-world phenomenon might be common in sparse networks with many vertices, as even a tiny fraction of short cuts would suffice."
"Thus, infectious diseases are predicted to spread much more easily and quickly in a small world; the alarming and less obvious point is how few short cuts are needed to make the world small."

Use your own words to explain these two implications. For the second implication, connect it with the spread of COVID.

Barabasi-Albert (BA) scale-free network

The third architecture is to generate scale-free graphs according to the Barabasi-Albert model. This model is developed by a recursive algorithm. Two parameters are needed, the initial number of nodes (n_0) and the total number of node (N). At the beginning, every initial node (the n_0 first nodes) knows the other ones, then, we create, one by one the other node. At the creation of a new node, this node is linked to an already existing node. The probability that the new node is linked to a certain node is proportional to the number of edges this node already has. In other word, the more links you have, the more likely new nodes will be linked to you.

This model is for any network respecting the idea of "rich get richer". The more friends one node has, the more likely the new nodes will be friend with him. This kind of model is relevant for internet network. For example, the more famous is the website, the more likely this website will be known by other websites.

You can easily generate a scale-free network for a given size:

```
g0 <- barabasi.game(100, power = 1, m = NULL, out.dist = NULL, out.seq =
NULL, out.pref = FALSE, zero.appeal = 1, directed = FALSE,algorithm
="psumtree", start.graph = NULL)

plot(g0, vertex.label= NA, edge.arrow.size=0.02,vertex.size =5, main =
"Scale-free network model, power=1")
```

Question 9 (3 points):

1) What does the `power` in the above function mean? How can it govern the structure of the network? (Hint: Change the value of power from 0.05, 0.5, 1, 1.5; See how the plot evolves; if you still fail to see the difference, visualize the vertex size according to the edge number, you can consider the code below.)
2) For two networks with a power of 0.5 and 1.5, respectively, what will be their resilience for 1) random attack, and 2) targeted attack? (the meanings of 'random attack' and 'targeted attack' are the same as what is mentioned in Lecture 6, scale-free network)

Exercise three: Simulation of simple and complex contagion

In this exercise, you will simulate the spread of simple and complex contagion in the Highschool network.

The strength of weak ties in simple contagion

For the Highschool network, identify five edges which after deletion, there will be significant gain of the average path lengths of the network. In other words, if such five edges did not exist, the average path length of the network would increase significant. Provide your answer in the format of A-B, in which A and B are the node ID. Are they weak ties or strong ties? (Question 10, 2 points)

To find out such edges, you can call the `Highschool_interactive_layout` function in Exercise 1 to visualize the network or use other nodal attributes that you think it is useful.

Simulate the spread of simple contagion in the Highschool network (Question 11, 6 points):

1) Build a simple independent cascade (IC) model with the following characteristics:

- Each node in the network has two statuses: infected (value=*1*) or healthy (value =*0*);
- At Day 0, all the nodes in the network are healthy;
- At Day 1, an infected node (N_0, node ID= S5) is introduced to the network;
- At the following days, all the nodes connecting to an infected node will have a chance of 0.15 (p=0.15) being infected.

- Once infected, the node will remain contagious and infected until the end of simulation.
- Model the contagion process for 4 weeks.

2) Apply the IC model to the Highschool network, record the number of newly infected people by day (i.e., newly confirmed cases by day) for further analysis.

(Please note that, due to the inherent randomness of the IC model, even though the contagion process starts from the same person, the final contagion outcome can be varied. Therefore, run your IC model 100 times and take the average results. If you still have questions, ask the teacher. Submit the codes of this question along with your answer*).*

```
#You can develop your own code according to the model description in Q12; Here is one example to build an IC model function in R. If you decide to use the code below, you need to understand the function in order to get the output required by this question and the following questions.


stopifnot(require(data.table))
stopifnot(require(Matrix))

calculate_value <- function(node, each_neighbors,Pprob){
  return(each_neighbors[[node]][ which(runif(length(each_neighbors[[node]]), 0, 1)<=Pprob)])
  #'runif' is a function to generate random number in R
}
#This function:
#1) searches the neighbors of contagious node;
#2) To those who are connected to a contagious node, generates a random number and compare to the
#probability of p, if random number<p, this node will be infected and return the value of 1

IC<-function(node_seed,network,Pprob){

  #prepare input for the 'calculate_value' function#
  adj_matrix <- igraph::as_adjacency_matrix(network, type = 'both')
  each_neighbors <- which(adj_matrix > 0, arr.ind = TRUE)
  each_neighbors <- split(each_neighbors[, 2], each_neighbors[, 1]) #get the neigbhour list of each node

  nNode<-vcount(network)
  node_status <- rep.int(0, nNode) #start from a healthy population
  day_infected<-vector()#Total number of infected population
  new_infected <- list()  # Record the ID of person getting infected at each time step

  day<-1
  node_status[as.numeric(node_seed)] <- 1 # infected(value=1) health(value=0)
  day_infected[day] <- sum(node_status )
  new_infected[[day]]<-node_seed #The ID of the person infected in Day 1 (Patient Zero)

  #simulate the spread of virus within 4 weeks##
  for (day in c(2:28)){
```

```
    ContagiousID<-which(node_status == 1)
    infectedID<-unlist(lapply(ContagiousID,calculate_value,each_neighbors,Pprob))
    newinfectedID<- setdiff(infectedID, which(node_status == 1))

    #Update the node status and other variables
    node_status[newinfectedID] <- 1
    day_infected[day] <- length(newinfectedID)
    new_infected[[day]]<-newinfectedID

    day=day+1
  }
  return(day_infected)  #return the number of newly infected people by day
  #return(list(day_infected,new_infected)) #if you want to see the ID of infected
ppl in each day, use this command instead
}
```

Now you are going to test the "strength of weak ties" in the simple contagion (Question 12, 6 points):

1) Delete the 5 edges that you have identified in Q11 from the Highschool network and form a new network (Highschool 2);
2) Delete 5 strong ties from the Highschool network and form a new network (Highschool 3);
3) Apply the IC models you developed in Q12 on the original Highschool network, Highschool2 and Highschool3. Record the number of newly infected people by day.
4) Generate a plot (with x-axis as Day, y-axis as the number of newly infected people by day) to compare the results from Step 3.
5) Recall the "strength of weak ties" from the lecture, do the results in Step 3&4 support such a claim and why?

Question 13 (8 points): In the above exercises, the "strength of weak ties" are tested in a simplified IC model with a specific probability *p*. Do you think your observation in Q13 holds regardless of the contagiousness of the virus? To find out,

1) Play around the probability *p* in the IC model. Change the value of *p* to high and low ends, run the IC model again on Highschool, Highschool 2 and Highschool 3, and see if you will observe different things (2 points).
2) The above IC model is a simplified version of the SIR model. In the SIR model, node have three status: **S**, **I**, or **R**, (**S**usceptible, **I**nfectious, or **R**ecovered). Modify the IC model to a SIR model with the following characteristics:

- Each node in the network has three statuses: **S**usceptible, **I**nfectious, or **R**ecovered.
- At Day 0, all the nodes in the network are **S**usceptible;
- At Day 1, an infectious node (N_0, node ID= S5) is introduced to the network;
- At the following days, all the nodes connecting to the infectious node will have a chance of 0.15 (p=0.15) being infected.

- Every infected node will remain infectious for 3 days, i.e., only the infected nodes activated from the past 3 days can transmit the virus to their neighbours. After that, their status becomes Recovered, which cannot be either Infectious or Susceptible again.
- Model the contagion process for 4 weeks.
  After you build the SIR model, repeat the exercise of Q13 3)-5) on the SIR model, and check if the "strength of weak ties" still holds. (6 points)

Now you will build a threshold model to simulate the spread of "once-a-week-beef" campaign in Highschool network. Below is the model description:

- Each node in the network has two status: adopt the behaviour (value=*1*) or refuse to adopt (value =*0*);
- At Day 0, *no one* in the network adopts the behaviour;
- At Day 1, a group of enthusiastic people (seed nodes, N_s, Node ID={59,63,91,92,99}) in the network decided to take action and adopt their own behaviours (change the status value to 1);
- At the following days, for nodes who haven't adopted ((value=*0*), they will check the status of their neighbours to decide to adopt or not:
  - For example, for node i, it has a predefined threshold of Θi, as defined in the *r* file. And node i has Ni number of neighbours in the whole network.
  - Among all the neighbours of Node i, if more than $Ni * \Theta i$ of them have adopted, node i will also adopt and change the status value to 1.
- Once adopted, the status of this node will remain as 1 till the end.

An arbitrary assumption about the thresholds of each node in the Highschool network has been made, which can be found in the "Highschool_network_att.csv". Build a threshold model according to the above model description and the predefined thresholds of each node, answer the following questions (Question 14, 5 points):

1) By seeding 5 nodes (ID=59,63,91,92,99), how many people in the network can be activated?
2) Use the "width of a bridge" from the lecture to explain why the contagion fails to reach the following two communities: a) the one consisted of Node 55, 107, 93, 109, 80, 28; b) the one consisted of Node 110, 39, 10, 1, 50, 106.

(Note: please submit the codes of this question along with your answer)

#You can develop your own code according to the model description in Q15; Here is one example to build a threshold model function in R. If you decide to use the code below, you need to understand the function in order to get the output required by this question and answer the following questions.

```
stopifnot(require(data.table))
stopifnot(require(Matrix))


calculate_adoptedNei <- function(node, node_status, each_neighbors){
  return(mean(node_status[each_neighbors[[node]]] == 1)) ### to calculate the percentage of adopted neigbhours
}


ThModel<-function(node_seed,network,threshold){
  #prepare input for the 'calculate_value' function#
  adj_matrix <- igraph::as_adjacency_matrix(network, type = 'both')
  each_neighbors <- which(adj_matrix > 0, arr.ind = TRUE)
  each_neighbors <- split(each_neighbors[, 2], each_neighbors[, 1]) #get the neigbhour list of each node


  nNode<-vcount(network)
  node_status <- rep.int(0, nNode)
  neighbour_status<-rep.int(0, nNode)  ##percentage of adopted neighbours
  new_infected <- list()
  day_total_infected <- rep(0,28) ### Total number of active people by end of each day



  ### Day 1 ####
  day <- 1
  node_status[as.numeric((node_seed))] <- 1
  new_infected[[day]] <-node_seed
  day_total_infected[day]=sum(node_status == 1)
  ####

  for (day in c(2:28)){
    NotAdopted <- which(node_status == 0)
    Adopted <- which(node_status == 1)

    neighbour_status[NotAdopted] <- unlist(lapply(NotAdopted, calculate_adoptedNei, node_status, each_neighbors))

    new_infected[[day]] <- setdiff(which(neighbour_status > threshold), Adopted)
    node_status[new_infected[[day]]] <- 1  #update the staus to 1 for those newly adopted
    day_total_infected[day] <- sum(node_status)

    day <- day + 1
  }
  #return(day_total_infected)
  return(list(day_total_infected,new_infected))
}
```

Threshold model relies heavily on the notion of "threshold" in human decision-making. However, approximating the real thresholds of a population is difficult, if not impossible. Therefore, assumption about the threshold has to be made. In Question 15, the threshold distribution is decided arbitrarily, which follows a uniform distribution from 0 to 1. In the lecture, we did a small survey to ask for your threshold to join the "once-a-week-beef" campaign. It resulted in an empirical distribution of the threshold of students in this class. Question 15 (6 points) : Apply the empirical distribution of the threshold of students in this class to the Highschool network, and answer the following questions:

1) How are you going to do it? Explain your method into steps.
2) What are the limitations of your method? What procedures are you going to take to address such limitations?
3) After you apply the empirical threshold distribution to the Highschool network, by using Node ID=59,63,91,92,99 as seeds, how many people in the network can be activated?

Question 16 (3 points) : Search the application cases of threshold model from literatures or other online source, chose one case and explain how they can get the threshold "right" for their model. (please provide the details of the literature or other online source that you are citing.)

Exercise four: Influence maximization

In the lecture, we discussed a few heuristics for the influence maximization problem in social network. Apply degree heuristics and betweenness heuristics to the IC model you have developed in Question 11 (! Please change the initially infected node to S107!). Answer the following questions (Question 17, 5 points):

1) You can immunize 3 nodes in the network, which after immunization, will never spread the virus to other connected nodes. According to degree heuristics and betweenness heuristics, which 3 nodes should be immunized in order to contain the virus?
2) Immunize the 3 nodes suggested by degree heuristics and betweenness heuristics, respectively, which heuristic provides the better outcome regarding a) the final activated number of people and b) flattening the daily infection curve (please provide figure in your answer)?
3) Do you think the observation in 2) (i.e., degree heuristic preforms better than betweenness heuristics, or the opposite) is sensitive to a) the network structure and b) parameter in the IC model? And Why?

(Important note: In Question 12, the initially infected node is S5. Please change it to S107 to answer Question 18. In other words, at Day 1, an infected node (N_0, node ID= S107) is introduced to the network.)

In addition to heuristics, we also introduced the greedy algorithm in the lecture. Develop a greedy algorithm to the IC model you have developed in Question 11 (! Please change the initially infected node to S107!). Answer the following questions (Question 18, 5 points):

1) You can immunize 3 nodes in the network, which after immunization, will never spread the virus to other connected nodes. According greedy algorithm, which 3 nodes should be immunized in order to contain the virus?
2) Compared to the result from greedy algorithm to those from degree heuristic and betweenness heuristic. Regarding a) the final activated number of people and b) flattening the daily infection curve (please provide figure in your answer), does greedy algorithm provide the best result? And explain the reason.

(Important note: In Question 12, the initially infected node is S5. Please change it to S107 to answer Question 18. In other words, at Day 1, an infected node (N_0, node ID= S107) is introduced to the network. Please submit the codes of this question along with your answer.)

Next you will study the influence maximization problem in the threshold model. Following below steps to answer Question 19 (8 points):

1) Use the threshold model for the "once-a-beef" campaign you build in Question 14, reset the seed nodes to a null set;
2) According to degree heuristics, which nodes should be included in the seed set in order to maximize the spread of the campaign? The size of seed set is 7, i.e., you can choose 7 nodes to activate to kick off the contagion process.
3) According to betweenness heuristics, which nodes should be included in the seed set in order to maximize the spread of the campaign? The size of seed set is 7, i.e., you can choose 7 nodes to activate to kick off the contagion process.
4) According to greedy algorithm, which nodes should be included in the seed set in order to maximize the spread of the campaign? The size of seed set is 7, i.e., you can choose 7 nodes to activate to kick off the contagion process.
5) Compared the results from degree heuristics, betweenness heuristics and greedy algorithm, which method provides the best outcome? Please show the results of three methods in a figure.

(Please submit the codes of this question along with your answer.)

Question 20 (7 points): You have compared the performance of degree heuristics, betweenness heuristics and greedy algorithm. Can you propose an even more efficient algorithm (i.e., achieve even higher diffusion rate with even less percentage of nodes using as seeds)? Answer this question by the following steps:

1) Description of your algorithm and the reason why you think it will be more effective;
2) Test your algorithm in the threshold model for the "once-a-beef" campaign used in Question 19; Comments on its effectiveness;
3) Test your algorithm in a large real-world network (e.g., n≥1000) using threshold model. For the larger network, you can choose one from the database, and make a

subgraph from it (e.g., choose only 1000 nodes). Describe the network you choose, your setting about threshold, and comment on the effectiveness of your algorithm on this network.

(Note: The proposed algorithm does not need to excel in all the settings, but should outperform at least some of the existing heuristics or greedy algorithm in some scenarios. And please submit the codes of this question along with your answer.)