Assignment 6 - Social Network Analysis

Charles Lang

Oct 13, 2016

In Assignment 2 we will be looking at some disciplinary data from a middle school. The file "discipline-data.csv" shows which teacher sent which student to the principal during the semester.

We will be using the "igraph" package to visualize the disciplinary relationships between teachers and students as a network. You can read more about igraph here:

<http://igraph.org/r/>

Start by installing the "igraph" package. Once you have installed igraph, load the package.

library(igraph)

##   
## Attaching package: 'igraph'

## The following objects are masked from 'package:stats':  
##   
## decompose, spectrum

## The following object is masked from 'package:base':  
##   
## union

Now upload the data file "discipline-data.csv" as a data frame called "D1". Each row is a disciplinary action from a teacher to a student so the first line shows that teacher "E" sent student "21" to the principal. It also shows the gender of both the teacher and student and the student's main elective field of study ("major"") and the field that the teacher instructs in ("t.expertise").

D1<-read.csv("discipline-data.csv")

Before you proceed, you will need to change the data type of the student id variable. Since it is a number R will automatically think it is an integer and code it as such (look at the list of variables by clicking on the data frame arrow in the Data pane. Here you will see the letters "int"" next to the stid variable, that stands for integer). However, in this case we are treating the variable as a category, there is no numeric meaning in the variable. So we need to change the format to be a category, what R calls a "factor". We can do this with the following code:

D1$stid <- as.factor(D1$stid)

igraph requires data to be in a particular structure. There are several structures that it can use but we will be using a combination of an "edge list" and a "vertex list". As you might imagine the edge list contains a list of all the relationships between students and teachers and any characteristics of those edges that we might be interested in. There are two essential variables in the edge list a "from" variable and a "to" variable that descibe the relationships between vertices (a disciplinary action is given "from" and teacher "to" a student). While the vertix list contains all the characteristics of those vertices, in our case gender and major.

So let's convert our data into an edge list!

First we will isolate the variables that are of interest: tid and stid

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:igraph':  
##   
## %>%, as\_data\_frame, groups, union

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

D2 <- dplyr::select(D1, tid, stid)

Since our data represnts every time a teacher sends a student to the principal there are multiple rows when the same teacher sends the same student. We want to collapse these into a single row, with a variable that shows how many times a teacher-student pair appears.

EDGE <- dplyr::count(D2, tid, stid)  
  
names(EDGE) <- c("from", "to", "count")

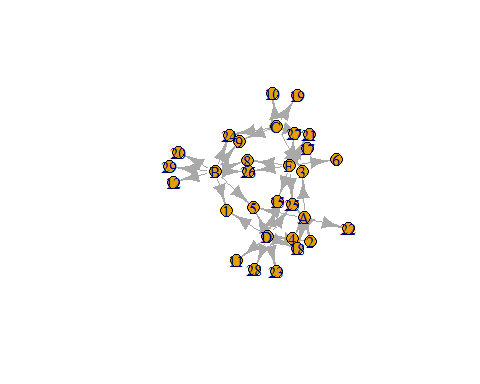
EDGE is your edge list. Now we need to make the vertex list, a list of all the teachers and students and their characteristics in our network.

#First we will separate the teachers from our original data frame  
V.TCH <- dplyr::select(D1, tid, t.gender, t.expertise)  
#Remove all the repeats so that we just have a list of each teacher and their characteristics  
V.TCH <- unique(V.TCH)  
#Add a variable that describes that they are teachers  
V.TCH$group <- "teacher"  
  
#Now repeat this process for the students  
V.STD <- dplyr::select(D1, stid, s.gender, s.major)  
V.STD <- unique(V.STD)  
V.STD$group <- "student"  
  
#Make sure that the student and teacher data frames have the same variables names  
names(V.TCH) <- c("id", "gender", "topic", "group")  
names(V.STD) <- c("id", "gender", "topic", "group")  
  
#Bind the two data frames together (you will get a warning because the teacher data frame has 5 types of id (A,B,C,D,E) and the student has 25 (1-30), this isn't a problem)  
VERTEX <- dplyr::bind\_rows(V.TCH, V.STD)

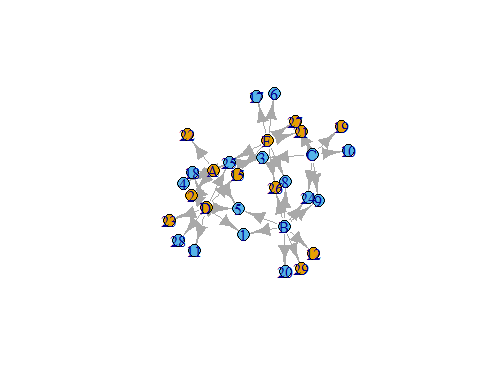
## Warning in bind\_rows\_(x, .id): Unequal factor levels: coercing to character  
  
## Warning in bind\_rows\_(x, .id): Unequal factor levels: coercing to character

Now we have both a Vertex and Edge list it is time to plot our graph!

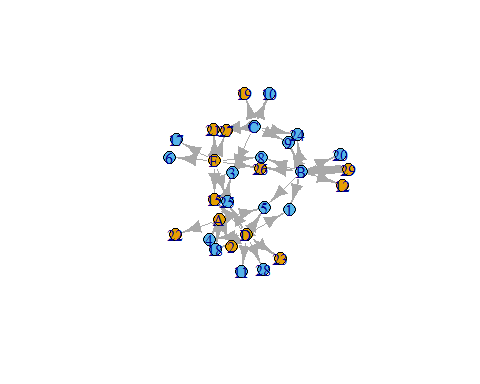
#Load the igraph package  
  
 library(igraph)  
  
#First we will make an object that contains the graph information using our two dataframes EDGE and VERTEX. Notice that we have made "directed = TRUE" - our graph is directed since discipline is being given from a teacher to a student.  
  
g <- graph.data.frame(EDGE, directed=TRUE, vertices=VERTEX)  
  
#Now we can plot our graph using the force directed graphing technique - our old friend Fruchertman-Reingold!  
  
plot(g,layout=layout.fruchterman.reingold)



#There are many ways to change the attributes of the graph to represent different characteristics of the newtork. For example, we can color the nodes according to gender.  
  
plot(g,layout=layout.fruchterman.reingold, vertex.color=VERTEX$gender)



#We can change the thickness of the edge according to the number of times a particular teacher has sent a particular student to the principal.  
  
plot(g,layout=layout.fruchterman.reingold, vertex.color=VERTEX$gender, edge.width=EDGE$count)



#Other attributes you may want to change are:  
  
#Arrow size  
#edge.arrow.size=  
  
#Remove the labels  
#vertex.label=NA  
  
#Vertex size  
#vertex.size=  
   
#And many others!

Play around with different graphs and then save your final version to the repo. Create a Zotero item for this R program and then commit, push and pull.

Once you have done this you need to practice with data from our class. Please create a **person-network** with the data set hudk4050-classes.csv. Once you have done this, also [look up](http://igraph.org/r/) how to generate the following network metrics: betweeness centrality and dregree. **Who is the most central person in the network?**

#Load data  
E1<-read.csv("hudk4050-classes.csv")  
#Load Dplyr  
library("dplyr")  
  
#combine first name and last name  
E1$name<-do.call(paste, c(E1[c("First.name", "Last.name")], sep=""))  
  
#gather and subest the data  
p<-E1 %>% tidyr::gather(course, classes, 3:7)

## Warning: attributes are not identical across measure variables; they will  
## be dropped

p2<- p %>% dplyr::select(name, classes)  
  
 # Assign NA to the blank classes so we can later omit them  
p2[p2==""] <- NA  
p2NA<-na.omit(p2)  
  
#Convert to matrix  
p2c<-data.matrix(p2NA$classes)  
p2n<-data.matrix(p2NA$name)  
  
# Now we loop through the data, if two classes are the same, we make vectors with the names of each in the class.  
  
i=2  
j=1  
idx=1  
x<-0  
y<-0  
  
while(j<140){  
 i=1  
while(i<140){  
if(identical(p2c[i], p2c[j])){  
 x[idx]<-p2n[i]   
 y[idx]<-p2n[j]  
 idx=idx+1  
   
}  
  
i=i+1  
 }  
j=j+1  
}  
  
#Reformat the data: remove NA, convert to matrix, bind the two matrices, give the columns names  
x1<-na.omit(x)  
y1<-na.omit(y)  
x2<-data.matrix(x1)  
y2<-data.matrix(y1)  
newdata<-cbind2(x2,y2)  
names(newdata) <- c("from", "to")  
  
  
# Manipulate data  
  
counts<-matrix( rep( 1, len=918), nrow = 918)  
newdata2<-cbind(newdata, counts)  
newdata2<-data.frame(newdata2)  
names(newdata2) <- c("from", "to")  
  
  
from<-as.character(newdata2$from)  
to<-as.character(newdata2$to)  
  
i=1  
idx=1  
x<-0  
y<-0  
  
while(i<918){  
if(identical(to[i],from[i])){  
   
 i=i+1 }  
 else{  
 x[idx]<-to[i]   
 y[idx]<-from[i]  
 idx=idx+1  
   
}  
  
i=i+1  
}  
  
counts<-matrix( rep( 1, len=773), nrow = 773)  
counts2<-as.numeric(counts)  
newdata3<-cbind(x,y,counts2)  
newdata3<-as.data.frame(newdata3)  
names(newdata3) <- c("to", "from", "counts")  
  
  
library(plyr)

## -------------------------------------------------------------------------

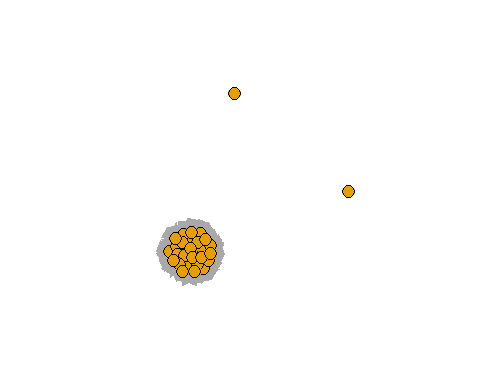
## You have loaded plyr after dplyr - this is likely to cause problems.  
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:  
## library(plyr); library(dplyr)

## -------------------------------------------------------------------------

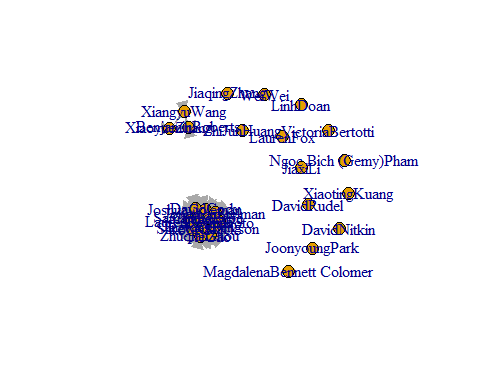
##   
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':  
##   
## arrange, count, desc, failwith, id, mutate, rename, summarise,  
## summarize

newdata3$counts<-as.numeric(newdata3$counts)  
  
newdata4<-ddply(newdata3, .(from,to), summarize, counts = sum(counts))  
  
  
#Now lets graph!  
library(igraph)  
  
#t1 are the edges  
  
t1<-unique(p$name)  
  
#now we can graph  
  
g <- graph.data.frame(newdata4, directed=TRUE, vertices=t1)  
  
#plot graph  
  
  
plot(g,layout=layout.fruchterman.reingold, vertex.label=NA)



#It's ugly, but its something.  
  
# Lets take away this class and see who is still connected!   
newdata5<-subset(newdata4, counts!=1)  
g2 <- graph.data.frame(newdata5, directed=TRUE, vertices=t1)  
plot(g2,layout=layout.fruchterman.reingold)



# Im not connected to anyone outside of class.  
  
betw<-estimate\_betweenness(g, vids = V(g), directed = TRUE, cutoff=10, weights = NULL, nobigint = TRUE)  
  
betw

## MagdalenaBennett Colomer VictoriaBertotti DavidCody   
## 0.0000000 0.7500000 1.0958429   
## JoshuaColeman LinhDoan JingtongFeng   
## 1.1791762 0.0000000 0.8804348   
## LaurenFox JieGao ShreyaGoel   
## 1.0413997 0.8387681 1.1375095   
## DevanGoto ChuhengHu ZhiJunHuang   
## 1.1375095 0.8804348 1.0958429   
## RobertJackson XiaotingKuang JiaxiLi   
## 1.0958429 1.0958429 1.0958429   
## DavidNitkin JoonyoungPark SamanthaPepe   
## 1.0958429 1.0958429 1.0958429   
## Ngoc Bich (Gemy)Pham BenjaminRoberts LaurenRomine   
## 1.0958429 0.8369565 0.8387681   
## DavidRudel JonathanStelman XiangyuWang   
## 1.0413997 1.0958429 1.0958429   
## WeiWei JiaqingZhang XiaoyueZhang   
## 1.0958429 1.0958429 1.0958429   
## ZhuqianZhou   
## 1.0958429

max(betw)

## [1] 1.179176

# =Joshua Coleman 1.179176