Class Eight Lab

Tom Zwiller

2024-04-10

#1. Installing and Loading in the Ergm Package

```
#Installing the libraries we need
library(ergm)
library(dplyr)
library(magrittr)
library(ggraph)
library(sna)
library(btergm)
```

#2. Loading in the Node and Edge Data

```
#Node Data
nodes <- read.csv('/Users/TomTheIntern/Desktop/Mendoza/Mod 4/Networks/Lab 2/nodelist.csv')
summary(nodes)
```

```
##        ID           Name                 Age           Gender
##  Min.   : 1.00   Length:12          Min.   :21.00   Length:12
##  1st Qu.: 3.75   Class :character   1st Qu.:23.00   Class :character
##  Median : 6.50   Mode  :character   Median :36.50   Mode  :character
##  Mean   : 6.50                      Mean   :38.00
##  3rd Qu.: 9.25                      3rd Qu.:45.75
##  Max.   :12.00                      Max.   :65.00
```

```
#Edge Data
edges <- read.csv('/Users/TomTheIntern/Desktop/Mendoza/Mod 4/Networks/Lab 2/edgelist.csv')
summary(edges)
```

```
##     ego_num         alter_num        ego                alter
##  Min.   : 1.000   Min.   : 1.000   Length:40          Length:40
##  1st Qu.: 2.750   1st Qu.: 2.750   Class :character   Class :character
##  Median : 5.000   Median : 5.000   Mode  :character   Mode  :character
##  Mean   : 5.575   Mean   : 5.575
##  3rd Qu.: 9.000   3rd Qu.: 9.000
##  Max.   :12.000   Max.   :12.000
##      type              strength
##  Length:40          Min.   :1.00
##  Class :character   1st Qu.:2.00
##  Mode  :character   Median :4.00
##                     Mean   :3.45
##                     3rd Qu.:4.25
##                     Max.   :5.00
```

```r
net_sna <- network(edges, matrix.type = "edgelist",
                   directed = T, vertex.attr = nodes)
net_sna
```

```
##  Network attributes:
##   vertices = 12
##   directed = TRUE
##   hyper = FALSE
##   loops = FALSE
##   multiple = FALSE
##   bipartite = FALSE
##   total edges= 40
##     missing edges= 0
##     non-missing edges= 40
##
##  Vertex attribute names:
##     Age Gender ID Name vertex.names
##
##  Edge attribute names:
##     alter ego strength type
```

#3. Adding Two Terms to the Model

```r
#basic ERGM
model1 <- ergm(net_sna ~ edges +  nodematch("Gender") + gwesp(0.05, fixed = TRUE))
```

#4. Interpret the Model

```r
summary(model1)
```

```
## Call:
## ergm(formula = net_sna ~ edges + nodematch("Gender") + gwesp(0.05,
##     fixed = TRUE))
##
## Monte Carlo Maximum Likelihood Results:
##
##                        Estimate Std. Error MCMC % z value Pr(>|z|)
## edges                   -3.6930     0.6108      0  -6.046  < 1e-04 ***
## nodematch.Gender         1.3202     0.3734      0   3.536 0.000406 ***
## gwesp.OTP.fixed.0.05     1.6594     0.5252      0   3.159 0.001581 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##      Null Deviance: 183.0  on 132  degrees of freedom
##  Residual Deviance: 128.6  on 129  degrees of freedom
##
## AIC: 134.6  BIC: 143.3  (Smaller is better. MC Std. Err. = 0.3547)
```

Goodness of Fit

```r
gof.model1_btergm <- btergm::gof(model1, nsim = 100, # checking goodness of fit of model estimates agai
                                 statistics = c(deg, odeg, ideg, triad.directed))
```

```
##
## Starting GOF assessment on a single computing core....

##
## No 'target' network(s) provided. Using networks on the left-hand side of the model formula as observe

## Simulating 100 networks from the following formula:
##  net_sna ~ edges + nodematch("Gender") + gwesp(0.05, fixed = TRUE)

## One network from which simulations are drawn was provided.

## Processing statistic: Degree

## Processing statistic: Outdegree

## Processing statistic: Indegree

## Processing statistic: Triad census
```

```r
gof.model1_btergm
```

```
## Degree

##    obs sim: mean median min max    Pr(>z)
## 0    0      0.61      0   0   4   0.41654
## 1    1      0.57      0   0   3   0.56685
## 2    2      0.69      1   0   3   0.08103 .
## 3    5      1.05      1   0   4 1.434e-07 ***
## 4    2      1.43      1   0   6   0.44776
## 5    1      2.03      2   0   6   0.17012
## 6    0      2.27      2   0   6   0.00250 **
## 7    1      1.85      2   0   6   0.25760
## 8    0      1.02      1   0   4   0.17431
## 9    0      0.36      0   0   5   0.63160
## 10   0      0.12      0   0   1   0.87302
## 11   0      0.00      0   0   0   1.00000

##
## Note: Small p-values indicate a significant difference
##       between simulations and observed network(s).

## Outdegree
```

```
##    obs sim: mean median min max   Pr(>z)
## 0    0      1.42      1   0   5 0.127240
## 1    1      0.91      1   0   4 0.922997
## 2    2      1.56      1   0   5 0.636528
## 3    5      2.26      2   0   6 0.003253 **
## 4    2      2.69      3   0   7 0.458658
## 5    1      1.80      2   0   6 0.390231
## 6    0      0.88      1   0   5 0.344599
## 7    1      0.33      0   0   2 0.471785
## 8    0      0.12      0   0   1 0.897453
## 9    0      0.03      0   0   1 0.974297
## 10   0      0.00      0   0   0 1.000000

##
## Note: Small p-values indicate a significant difference
##       between simulations and observed network(s).

## Indegree

##    obs sim: mean median min max   Pr(>z)
## 0    0      1.45      1   0   4 0.110907
## 1    1      0.90      1   0   4 0.912457
## 2    2      1.68      2   0   6 0.724982
## 3    5      2.04      2   0   6 0.001137 **
## 4    2      2.64      3   0   6 0.481672
## 5    1      1.81      2   0   4 0.373190
## 6    0      1.04      1   0   3 0.252883
## 7    1      0.37      0   0   2 0.488547
## 8    0      0.06      0   0   2 0.947407
## 9    0      0.01      0   0   1 0.991228
## 10   0      0.00      0   0   0 1.000000

##
## Note: Small p-values indicate a significant difference
##       between simulations and observed network(s).

## Triad census

##      obs sim: mean median min max    Pr(>z)
## 003   69     39.55   38.5   7  77 0.0810722 .
## 012    0     63.99   65.0  33  93 0.0001503 ***
## 102  113     28.50   28.0  10  53 5.574e-07 ***
## 021D   0      6.86    6.0   0  21 0.6844792
## 021U   0      7.14    7.0   0  27 0.6723357
## 021C   0     15.16   14.0   2  42 0.3691765
## 111D   0     11.21   11.0   1  22 0.5066680
## 111U   0     11.63   12.0   3  27 0.4908774
## 030T   0      6.72    5.0   0  30 0.6905819
## 030C   0      2.34    2.0   0  15 0.8897567
## 201   27      3.67    3.0   0  11 0.1669795
## 120D   0      3.89    3.0   0  11 0.8177593
## 120U   0      3.65    3.0   1  11 0.8288231
```

```
## 120C   0       6.25    5.0   0  16 0.7112151
## 210    0       7.59    7.0   2  19 0.6529983
## 300    0       0.00    0.0   0   0 1.0000000

##
## Note: Small p-values indicate a significant difference
##       between simulations and observed network(s).
```

I tinkered with the model for a bit, swapping variables until I found a paring that was statistically significant and finally found that nodematch worked well with gwesp.

Nodematch focused on Gender makes the most sense; I tended to have more men in my network and those men generally had relationships with other men; the only exceptions were myself and my girlfriend, her father and her mother and my friend and his girlfriend.

And then Gwesp was strong because most of the time two nodes had a handful of shared partners, hence it being statistically significant.

All things considered, the model had a low AIC (134) and BIC (142.7), and was relatively accurate