Affiliation Networks - Chapter 9

Nick Lauerman

Contents

Libraries and data used

Libraries

```
library(UserNetR)
library(igraph)
```

```
##
## Attaching package: 'igraph'
```

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

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

Data

```
data(hwd)
```

defining affiliation netwirks

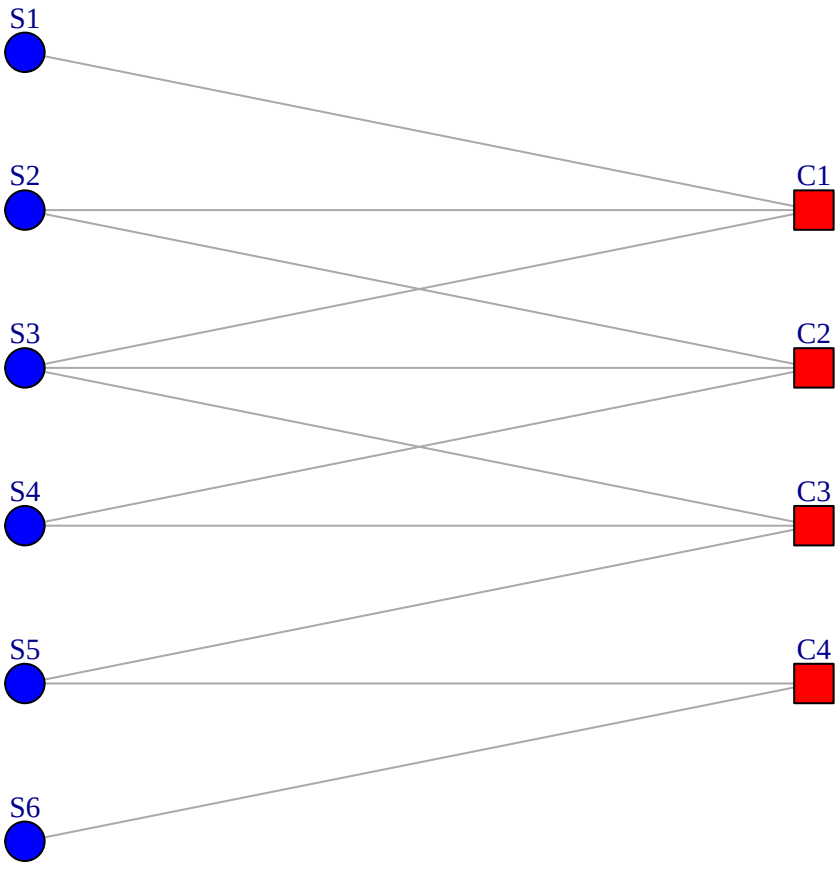
affiliation as 2-mode networks

```
C1 <- c(1,1,1,0,0,0)
C2 <- c(0,1,1,1,0,0)
C3 <- c(0,0,1,1,1,0)
C4 <- c(0,0,0,0,1,1)
aff.df <- data.frame(C1,C2,C3,C4)
rownames(aff.df) <- c("S1","S2","S3","S4","S5","S6")
aff.df
```

```
##    C1 C2 C3 C4
## S1  1  0  0  0
## S2  1  1  0  0
## S3  1  1  1  0
## S4  0  1  1  0
## S5  0  0  1  1
## S6  0  0  0  1
```

bipartite graphs

```
bn <- graph.incidence(aff.df)
plt.x <- c(rep(2,6),
           rep(4,4))
plt.y <- c(7:2,
           6:3)
lay <- as.matrix(cbind(plt.x,
                       plt.y))

shapes <- c("circle",
            "square")
colors <- c("blue",
            "red")
plot(bn,
     vertex.color = colors[V(bn)$type + 1],
     vertex.shape = shapes[V(bn)$type + 1],
     vertex.size = 10,
     vertex.label.degree = -pi/2,
     vertex.label.dist = 1.2,
     vertex.label.cex = 0.9,
     layout = lay)
```



Affiliation Network Basics

creating affiliation networks from incudence matrices

```
bn <- graph.incidence(aff.df)
bn
```

```
## IGRAPH 40297e6 UN-B 10 11 --
## + attr: type (v/l), name (v/c)
## + edges from 40297e6 (vertex names):
##  [1] S1--C1 S2--C1 S2--C2 S3--C1 S3--C2 S3--C3 S4--C2 S4--C3 S5--C3 S5--C4
## [11] S6--C4
```

```
get.incidence(bn)
```

```
##    C1 C2 C3 C4
## S1  1  0  0  0
## S2  1  1  0  0
## S3  1  1  1  0
## S4  0  1  1  0
## S5  0  0  1  1
## S6  0  0  0  1
```

```
V(bn)$type
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE
```

```
V(bn)$name
```

```
##  [1] "S1" "S2" "S3" "S4" "S5" "S6" "C1" "C2" "C3" "C4"
```

##Creating Affiliation network from edge list

```
el.df <- data.frame(rbind(c("S1","C1"),
                          c("S2","C1"),
                          c("S2","C2"),
                          c("S3","C1"),
                          c("S3","C2"),
                          c("S3","C3"),
                          c("S4","C2"),
                          c("S4","C3"),
                          c("S5","C3"),
                          c("S5","C4"),
                          c("S6","C4")))
el.df
```

```
##    X1 X2
## 1  S1 C1
## 2  S2 C1
## 3  S2 C2
## 4  S3 C1
## 5  S3 C2
## 6  S3 C3
## 7  S4 C2
## 8  S4 C3
## 9  S5 C3
## 10 S5 C4
## 11 S6 C4
```

```
bn2 <- graph.data.frame(el.df,
                        directed = FALSE)
bn2
```

```
## IGRAPH 40347ae UN-- 10 11 --
## + attr: name (v/c)
## + edges from 40347ae (vertex names):
##  [1] S1--C1 S2--C1 S2--C2 S3--C1 S3--C2 S3--C3 S4--C2 S4--C3 S5--C3 S5--C4
## [11] S6--C4
```

```
V(bn2)$type <- V(bn2)$name %in% el.df[ ,1]
bn2
```

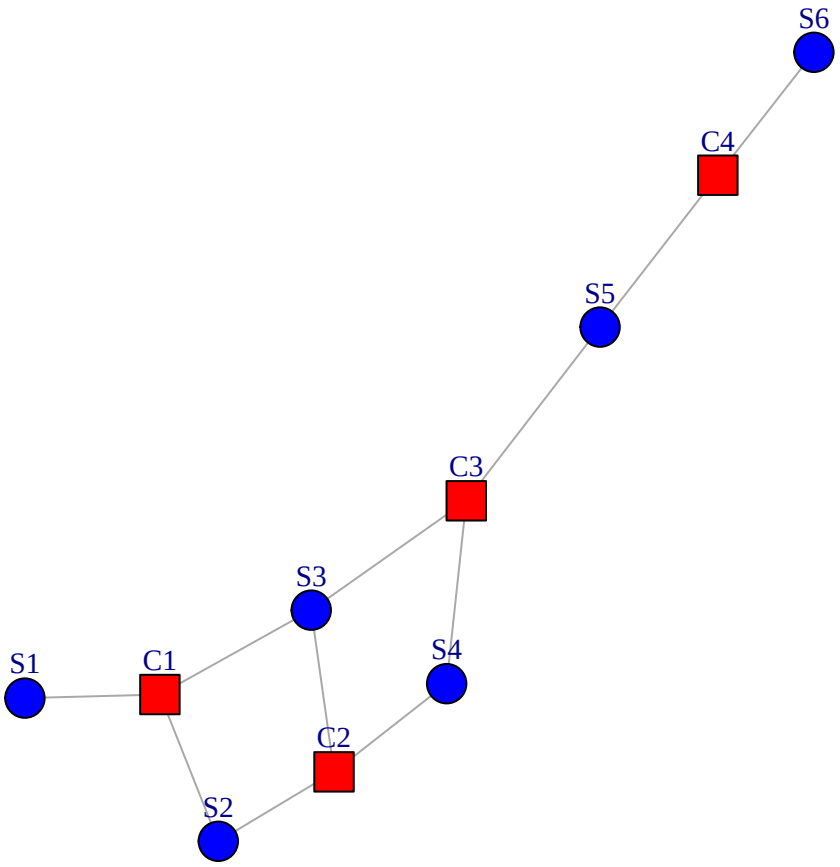
```
## IGRAPH 40347ae UN-B 10 11 --
## + attr: name (v/c), type (v/l)
## + edges from 40347ae (vertex names):
##  [1] S1--C1 S2--C1 S2--C2 S3--C1 S3--C2 S3--C3 S4--C2 S4--C3 S5--C3 S5--C4
## [11] S6--C4
```

```
graph.density(bn) == graph.density(bn2)
```

```
## [1] TRUE
```

plotting Affiliation networks

```
shapes <- c("circle",
            "square")
colors <- c("blue",
            "red")
plot(bn,
     vertex.color = colors[V(bn)$type + 1],
     vertex.shape = shapes[V(bn)$type + 1],
     vertex.size = 10,
     vertex.label.degree = -pi/2,
     vertex.label.dist = 1.2,
     vertex.label.cex = 0.9)
```




```
bn.pr <- bipartite.projection(bn)
bn.pr
```

```
## $proj1
## IGRAPH 403e8d0 UNW- 6 8 --
## + attr: name (v/c), weight (e/n)
## + edges from 403e8d0 (vertex names):
## [1] S1--S2 S1--S3 S2--S3 S2--S4 S3--S4 S3--S5 S4--S5 S5--S6
##
## $proj2
## IGRAPH 403e8d0 UNW- 4 4 --
## + attr: name (v/c), weight (e/n)
## + edges from 403e8d0 (vertex names):
## [1] C1--C2 C1--C3 C2--C3 C3--C4
```

```
graph.density(bn.pr$proj1)
```

```
## [1] 0.5333333
```

```
bn.students <- bn.pr$proj1
bn.class <- bn.pr$proj2
graph.density(bn.students)
```

```
## [1] 0.5333333
```

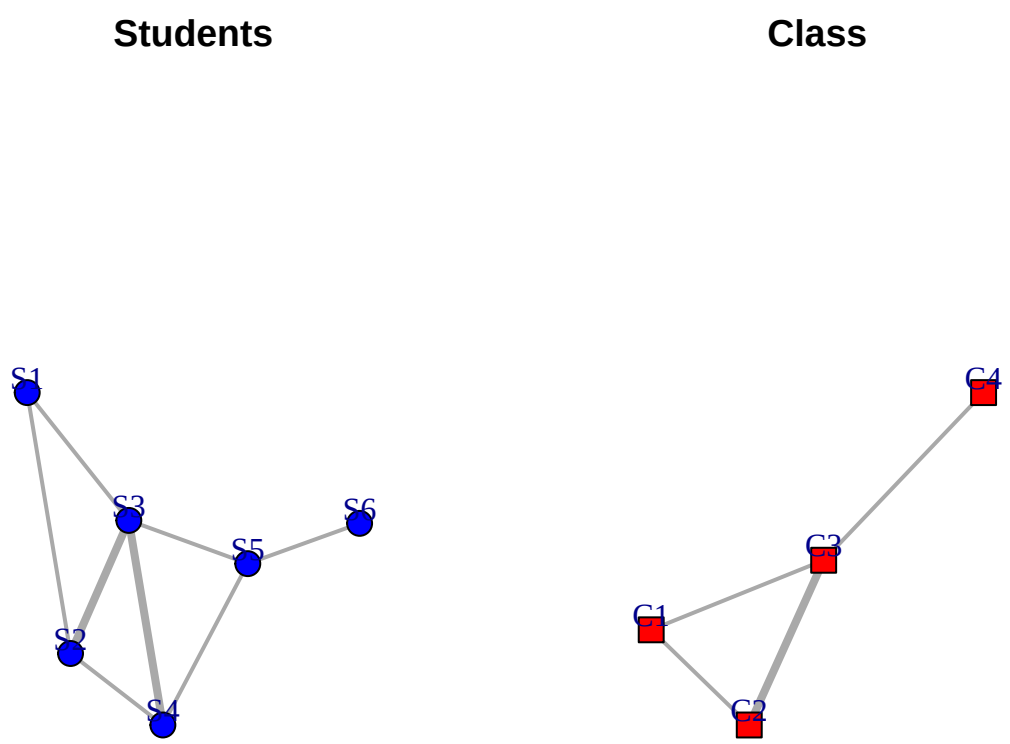
```
get.adjacency(bn.students,
              sparse = FALSE,
              attr = "weight")
```

```
##    S1 S2 S3 S4 S5 S6
## S1  0  1  1  0  0  0
## S2  1  0  2  1  0  0
## S3  1  2  0  2  1  0
## S4  0  1  2  0  1  0
## S5  0  0  1  1  0  1
## S6  0  0  0  0  1  0
```

```
get.adjacency(bn.class,
              sparse = FALSE,
              attr = "weight")
```

```
##    C1 C2 C3 C4
## C1  0  2  1  0
## C2  2  0  2  0
## C3  1  2  0  1
## C4  0  0  1  0
```

```
op <- par(mfrow = c(1,2))
plot(bn.students,
     vertex.color = "blue",
     vertex.shape = "circle",
     main = "Students",
     edge.width = E(bn.students)$weight * 2,
     vertex.size = 15,
     vertex.label.degree = -pi/2,
     vertex.label.dist = 1.2,
     vertex.label.cex = 1)
plot(bn.class,
     vertex.color = "red",
     vertex.shape = "square",
     main = "Class",
     edge.width = E(bn.students)$weight * 2,
     vertex.size = 15,
     vertex.label.degree = -pi/2,
     vertex.label.dist = 1.2,
     vertex.label.cex = 1)
```




```
par(op)
```

examples

Hollywood Actors as an affiliation network

```
data(hwd)
h1 <- hwd
h1
```

```
## IGRAPH 9cdab39 UN-B 1365 1600 --
## + attr: name (v/c), type (v/l), year (v/n), IMDBrating (v/n),
## | MPAArating (v/c)
```

```
## + edges from 9cdab39 (vertex names):
##  [1] Inception          --Leonardo DiCaprio
##  [2] Inception          --Joseph Gordon-Levitt
##  [3] Inception          --Ellen Page
##  [4] Inception          --Tom Hardy
##  [5] Inception          --Ken Watanabe
##  [6] Inception          --Dileep Rao
##  [7] Inception          --Cillian Murphy
## + ... omitted several edges
```

```
V(h1)$name[1:10]
```

```
##  [1] "Inception"
##  [2] "Alice in Wonderland"
##  [3] "Kick-Ass"
##  [4] "Toy Story 3"
##  [5] "How to Train Your Dragon"
##  [6] "Despicable Me"
##  [7] "Scott Pilgrim vs. the World"
##  [8] "Hot Tub Time Machine"
##  [9] "Harry Potter and the Deathly Hallows: Part 1"
## [10] "Tangled"
```

```
V(h1)$type[1:10]
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

```
V(h1)$IMDBrating[1:10]
```

```
##  [1] 8.8 6.5 7.8 8.4 8.2 7.7 7.5 6.5 7.7 7.9
```

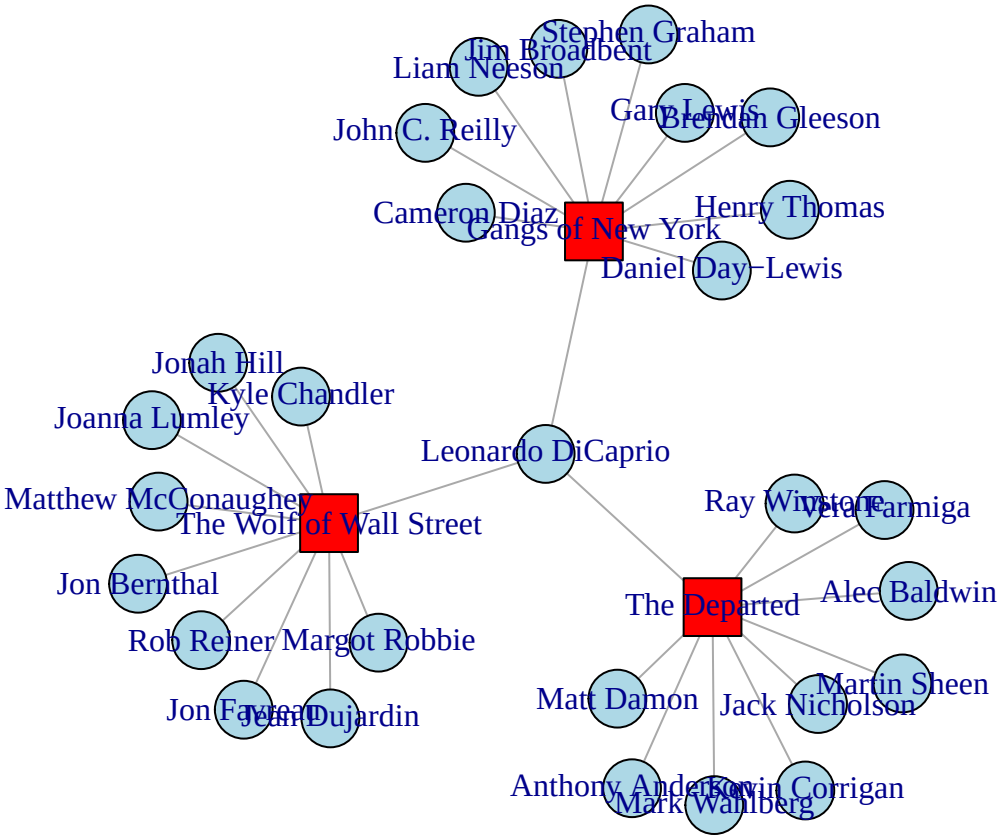
```
V(h1)$name[155:165]
```

```
##  [1] "Notting Hill"               "Eyes Wide Shut"
##  [3] "The Green Mile"             "10 Things I Hate About You"
##  [5] "American Pie"               "Girl, Interrupted"
##  [7] "Leonardo DiCaprio"          "Joseph Gordon-Levitt"
##  [9] "Ellen Page"                 "Tom Hardy"
## [11] "Ken Watanabe"
```

```
V(h1)$shape <- ifelse(V(h1)$type == TRUE,
                      "square",
                      "circle")
V(h1)$shape[1:10]
```

```
##  [1] "square" "square" "square" "square" "square" "square" "square" "square"
##  [9] "square" "square"
```

```
V(h1)$color <- ifelse(V(h1)$type == TRUE,
                      "red",
                      "lightblue")
h2 <- subgraph.edges(h1,
                     E(h1)[inc(V(h1)[name %in%
                                       c("The Wolf of Wall Street",
                                         "Gangs of New York",
                                         "The Departed")])])
plot(h2,
     layout = layout_with_kk)
```




```
graph.density(h1)
```

```
## [1] 0.001718711
```

```
table(degree(h1, v=V(h1)[type==FALSE]))
```

```
##
##   1   2   3   4   5   6   7   8
## 955 165  47  23  11   2   1   1
```

```
mean(degree(h1, v=V(h1)[type==FALSE]))
```

```
## [1] 1.327801
```

```
V(h1)$deg <- degree(h1)
V(h1)[type == FALSE & deg > 4]$name
```

```
##  [1] "Leonardo DiCaprio" "Emma Watson"       "Richard Griffiths"
##  [4] "Harry Melling"     "Daniel Radcliffe"  "Rupert Grint"
##  [7] "James Franco"      "Ian McKellen"      "Martin Freeman"
## [10] "Bradley Cooper"    "Christian Bale"    "Samuel L. Jackson"
## [13] "Natalie Portman"   "Brad Pitt"         "Liam Neeson"
```

```
busy_actor <- data.frame(cbind(
  Actor = V(h1)[type == FALSE & deg > 4]$name,
  Movies = V(h1)[type == FALSE & deg >4]$deg
))
busy_actor[order(busy_actor$Movies,
                 decreasing = TRUE), ]
```

```
##                Actor Movies
## 5   Daniel Radcliffe      8
## 11    Christian Bale      7
## 1  Leonardo DiCaprio      6
## 2        Emma Watson      6
## 3  Richard Griffiths      5
## 4      Harry Melling      5
## 6       Rupert Grint      5
## 7       James Franco      5
## 8       Ian McKellen      5
## 9     Martin Freeman      5
## 10    Bradley Cooper      5
## 12 Samuel L. Jackson      5
## 13   Natalie Portman      5
## 14         Brad Pitt      5
## 15       Liam Neeson      5
```

```
for(i in 161:1365){
  V(h1)[i]$totrating <- sum(V(h1)[nei(i)]$IMDBrating)
}
max(V(h1)$totrating, na.rm = TRUE)
```

```
## [1] 60.9
```

```
pop_actor <- data.frame(cbind(
  Actor = V(h1)[type == FALSE &
                  totrating > 40]$name,
  Popularity = V(h1)[type == FALSE &
                       totrating > 40]$totrating))
pop_actor[order(pop_actor$Popularity,
                decreasing = TRUE), ]
```

```
##               Actor Popularity
## 3  Daniel Radcliffe       60.9
## 4    Christian Bale       55.5
## 1 Leonardo DiCaprio       49.6
## 2       Emma Watson         45
## 5         Brad Pitt       40.5
```

```
for(i in 161:1365){
  V(h1)[i]$avgrating <- mean(V(h1)[nei(i)]$IMDBrating)
}
num <- V(h1)[type == FALSE]$deg
```

```r
avgpop <- V(h1)[type == FALSE]$avgrating
summary(lm(avgpop ~ num))
```

```
##
## Call:
## lm(formula = avgpop ~ num)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.9858 -0.4330  0.1977  0.6170  1.6142
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.33868    0.05440 134.911   <2e-16 ***
## num          0.04714    0.03527   1.337    0.182
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9605 on 1203 degrees of freedom
## Multiple R-squared:  0.001483,   Adjusted R-squared:  0.0006528
## F-statistic: 1.786 on 1 and 1203 DF,  p-value: 0.1816
```

```r
scatter.smooth(num,
               avgpop,
               col = "lightblue",
               ylim = c(2, 10),
               span = 0.8,
               xlab = "Number of Movies",
               ylab = "Avg. Popularity")
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## pseudoinverse used at 0.965
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## neighborhood radius 1.035
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## reciprocal condition number 5.4652e-015
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## There are other near singularities as well. 1
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## pseudoinverse used at 0.965
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## neighborhood radius 1.035
```

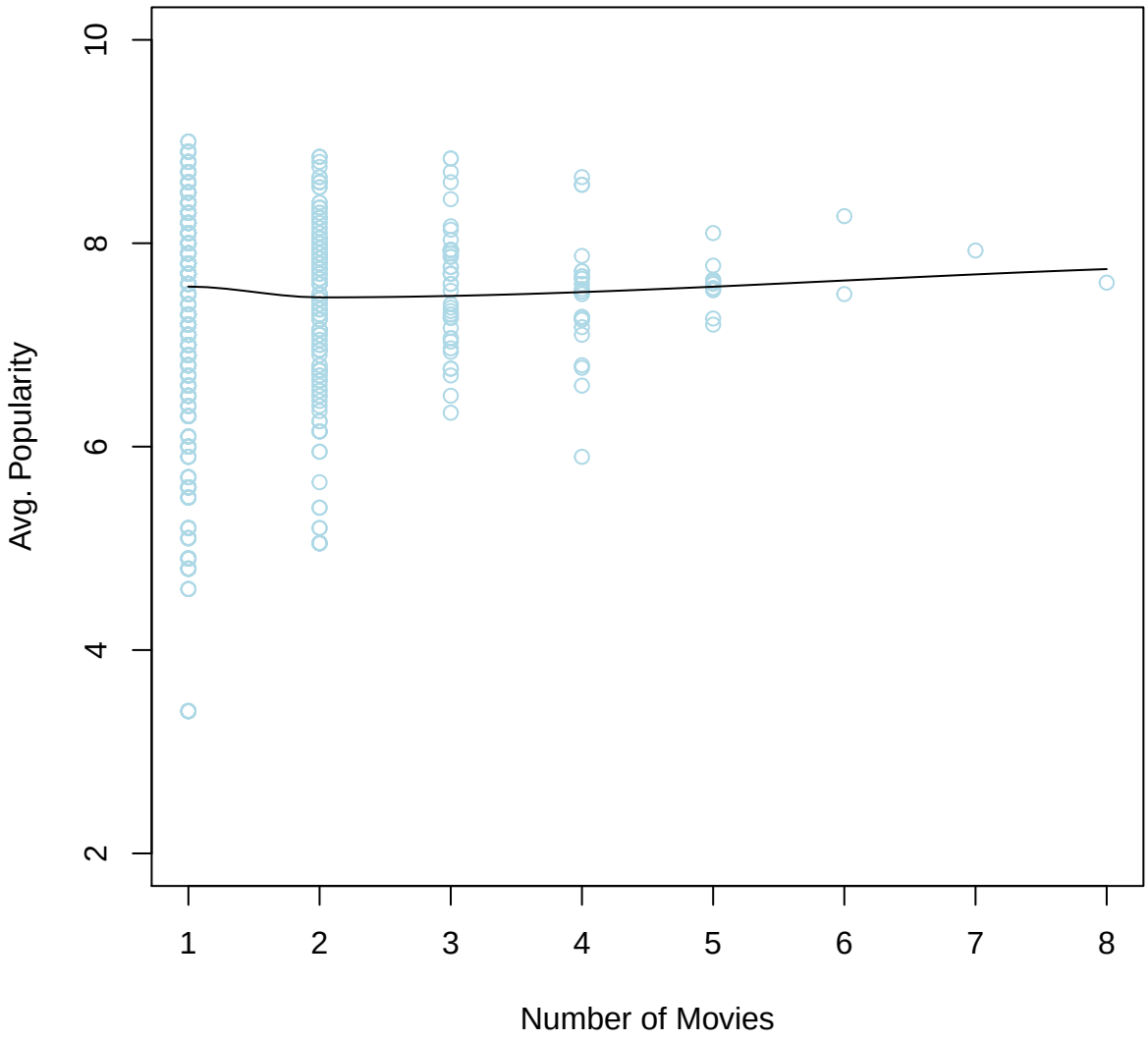
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## reciprocal condition number 1.9145e-015
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## There are other near singularities as well. 1
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## pseudoinverse used at 0.965
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## neighborhood radius 1.035
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## reciprocal condition number 1.2706e-015

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## pseudoinverse used at 0.965

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## neighborhood radius 1.035

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## reciprocal condition number 2.3999e-015

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## pseudoinverse used at 0.965

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## neighborhood radius 1.035

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## reciprocal condition number 5.4652e-015

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = FALSE, :
## There are other near singularities as well. 1
```


Avg. Popularity
10
8
6
4
2
1
2
3
4
5
6
7
8
Number of Movies