Machine Learning Workshop - iGraph Example

Brian Clark

Winter 2019

Contents

Overview

This vignette provides a brief introduction to networks using R's `igraph` package which is explained **here**. It also shows an example of fitting a stochastic block model using the `blockmodels` package. For more informaiton on stochastic block models, see **this document** by Emmanuel Abbe.

iGraph Package

Network analysis or graph theory involves describing a network of individuals and how they interact. Common finance applications include models to assess systemic risk (Billio et al. (2012)) who develop several measures of systemic risk based on networks. Another good example is a series of papers by Gerard Hoberg (USC) and Gordon Phillips (Dartmouth). They developed a new measure of industries based on text mining of corporate reports. Essentially, what they do is look for similarities in the product market space based on SEC reports. They then convert the text mining to similarity scores and use those scores to cluster firms into groups. The clusters represent industries. The data is freely available **here**. We will be using one of their annual datasets in this vignette. However, this document only very briefly describes network theory; the main intent is to provide an example of how to run some basic network models in R.

The fist step is to import the data. We will search over the current directory for a pattern matching our file. In practice, this is a useful way if yo have batched data (e.g., a series of years of data with structured names - data1997.txt, data1998.txt, etc.). The `list.files` function is goin gto search over a given diretory for files matching a patter - in this example `tnicall*txt`.

```
rm(list=ls())

setwd("C:/Users/CLARKB2/Documents/Classes/ML Course")

dir()
```

```
##   [1] "10Algorithms-08.pdf"
##   [2] "13_datafest_cart_talk.pdf"
##   [3] "2019_Shaft_MPF_Woods.xlsx"
##   [4] "APM.pdf"
##   [5] "Best paper award PPBRC 2019 Brian Clark.pdf"
##   [6] "bib.bib"
##   [7] "cart.tree.png"
##   [8] "Class01_001_Intro_to_R.Rmd"
##  [9] "Class01_001_Intro_to_R[rmd2r].R"
## [10] "Class05_001_optimization_Ndim.Rmd"
## [11] "Class05_001_optimization_Ndim[rmd2r].R"
## [12] "Class06_002_portfolioOptimization_nl.Rmd"
## [13] "Class06_002_portfolioOptimization_nl[rmd2r].R"
## [14] "Comparative_Study_Id3_Cart_And_C4.5_Deci.pdf"
## [15] "Compress_Mort_Data_Annual.R"
## [16] "ESLII.pdf"
## [17] "ISLR Seventh Printing.pdf"
## [18] "kMeannsCluster.R"
## [19] "KNN.png"
## [20] "M_2017-12-05.pdf"
## [21] "Master Machine Learning Algorithms.pdf"
## [22] "ML_002_KNN.nb.html"
## [23] "ML_002_KNN.Rmd"
## [24] "ML_Course_Agenda.pdf"
## [25] "ML_Course_Agenda.Rmd"
## [26] "ML_Course_C45.R"
## [27] "ML_Course_Ensemble.nb.html"
## [28] "ML_Course_Ensemble.pdf"
## [29] "ML_Course_Ensemble.R"
## [30] "ML_Course_Ensemble.Rmd"
## [31] "ML_Course_Ensemble[rmd2r].R"
## [32] "ML_Course_iGraph.pdf"
## [33] "ML_Course_iGraph.Rmd"
## [34] "ML_Course_Ind.Rda"
## [35] "ML_Course_Keras.R"
## [36] "ML_Course_Kmeans.nb.html"
## [37] "ML_Course_Kmeans.pdf"
## [38] "ML_Course_Kmeans.Rmd"
## [39] "ML_Course_Kmeans[rmd2r].R"
## [40] "ML_Course_KNN_Function.nb.html"
## [41] "ML_Course_KNN_Function.pdf"
## [42] "ML_Course_KNN_Function.Rmd"
## [43] "ML_Course_KNN_Function[rmd2r].R"
## [44] "ML_Course_KNN_Script.R"
## [45] "ML_Course_Mortgage_Data.pdf"
## [46] "ML_Course_Mortgage_Data.R"
## [47] "ML_Course_Mortgage_Data.Rmd"
## [48] "ML_Course_Mortgage_Data[rmd2r].R"
## [49] "ML_Course_OCC_001.nb.html"
## [50] "ML_Course_OCC_001.pdf"
## [51] "ML_Course_OCC_001.Rmd"
## [52] "ML_Course_Optimization_Basics.pdf"
## [53] "ML_Course_Optimization_constrOptim_nl.pdf"
## [54] "ML_Course_Ridge_LASSO.nb.html"
## [55] "ML_Course_Ridge_LASSO.pdf"
## [56] "ML_Course_Ridge_LASSO.Rmd"
## [57] "ML_Course_Ridge_LASSO[rmd2r].R"
## [58] "ML_Course_SVM.nb.html"
## [59] "ML_Course_SVM.pdf"
## [60] "ML_Course_SVM.Rmd"
## [61] "ML_Course_SVM[rmd2r].R"
## [62] "ML_Course_Trees.nb.html"
## [63] "ML_Course_Trees.pdf"
## [64] "ML_Course_Trees.R"
## [65] "ML_Course_Trees.Rmd"
## [66] "ML_Course_Trees[rmd2r].R"
## [67] "ML_Seminar_NonQuant_OCC_001.nb.html"
## [68] "ML_Seminar_NonQuant_OCC_001.pdf"
## [69] "ML_Seminar_NonQuant_OCC_001.Rmd"
## [70] "Mortgage_Annual.Rda"
## [71] "Motrgage_Annual.Rda"
## [72] "New folder"
## [73] "Non-Quant-OCC_FINAL_FINAL-forECON.pptx"
## [74] "Regression_Trees_and_Rule_Based_Models_Install.R"
## [75] "rmarkdown-reference.pdf"
## [76] "RMD2R.R"
## [77] "Rpart_details.pdf"
## [78] "SBM_ML_Course"
## [79] "The caret Package.pdf"
## [80] "tnicall1997.txt"
## [81] "trees.txt"
## [82] "Value_of_Location_in_Derivative_Markets.pdf"
```

```
files <- list.files(pattern = glob2rx("tnicall*txt"),
                    full.names=TRUE)

print(files)
```

```
## [1] "./tnicall1997.txt"
```

Once we have the list of files, we can read it in. Note that this step takes a bit of time since the raw file is large. We will reduce the file size before running any network analyses.

```
# Read the first file in (for testing use 1997):
df <- read.table(files[1],sep="\t",header=T)

# Take a look at the first 10 observations
head(df,10)
```

```
##     score gvkey1 gvkey2 ball year
## 1      NA   1004   1004   NA 1997
## 2  0.0000   1004   1013    1 1997
## 3  0.0363   1004   1021    1 1997
## 4  0.0000   1004   1034    1 1997
## 5  0.0000   1004   1038    1 1997
## 6  0.0000   1004   1043    1 1997
## 7  0.0567   1004   1045    1 1997
## 8  0.0000   1004   1050    1 1997
## 9  0.0699   1004   1056    1 1997
## 10 0.0000   1004   1072    1 1997
```

```
# drop ball and year vars:
df <- df[,c(-4,-5)]
```

The next thing that we want to know is how many unique firms we have. Our graph (network) is going to be defined by what's called an adjacency matrix which is a square matrix of dimension n, where n is the number of nodes (in our case firms). The links between the nodes are called edges and those are given by the pairwise connections between our nodes. In the Gordon and Phillips data, the edges are the pairwise similarity scores.

The next step is then to get the number of unique firms so we know how large our graph is going to be.

```
# Get the unique firms:
unq.1 <- unique(df$gvkey1)
unq.1.n <- length(unq.1)
print(unq.1.n)
```

```
## [1] 7518
```

```
unq.1.n*unq.1.n
```

```
## [1] 56520324
```

```
unq.2 <- unique(df$gvkey2)
unq.2.n <- length(unq.2)
print(unq.2.n)
```

```
## [1] 7518
```

```
unq.2.n*unq.2.n
```

```
## [1] 56520324
```

```
m <- dim(df)[1]
print(m)
```

```
## [1] 56111398
```

Note that we have 7,518 firms s our adjacency matrix wil have over 56 million edges! If we don't reduce the size of the problem, the rest of the code will not run (on my machine anyway).

But first, let's get some basic information on the scores. We want to count the number of scores that are approximately equal to zero. This gives us an idea of how sparse our graph is going to be.

```
# Count no. of scores = 0:
score.0 <- (df$score[df$score <= 0.00001])
# print(score.0)
length(score.0)
```

```
## [1] 29601126
```

```
score.0plus <- (df$score[df$score > 0.00001])
length(score.0plus)
```

```
## [1] 26517790
```

```
cutoff <- 0.2132
n.score.below <- length((df$score[df$score <= cutoff]))
n.score.above <- length((df$score[df$score > cutoff]))
n.score.above/m
```

```
## [1] 0.004549307
```

The next step is to sample the data by selecting 50 random firms. We will have an adjacency matrix of dimesion 50 × 50.

```
# Make the adjency matrix:
library("igraph")
```

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

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

```
##
##     decompose, spectrum
```

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

```r
library("blockmodels")
```

```
## Loading required package: Rcpp
```

```
## Loading required package: parallel
```

```
## Loading required package: digest
```

```r
# make the network:

# remove missing scores (when gvkey1=gvkey2):
df <- df[-which(is.na(df$score)),]
m <- dim(df)[1]

# Reorder the columns:
df <- df[,c(2,3,1)]

# make a sample network by retaining a x% of the firms:
keep <- sample(unq.1,50)
df.50 <- df[df$gvkey1 %in% keep,]
df.50 <- df.50[df.50$gvkey2 %in% keep,]
```

At this point, it makes sense to save the data so next time we run this we don't need to wait so long.

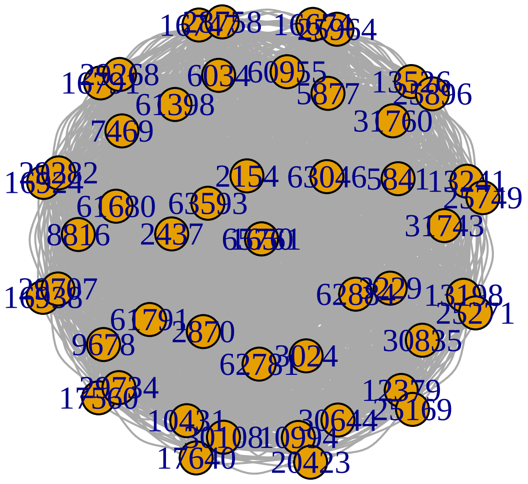
```r
save(df.50,file="ML_Course_Ind.Rda")
```

Finally, we can make the adjacency matrix and we have our graph. Since the similarity scores are symmetric, we set `directed=FALSE`. A directed graph is one in which the edges are not symmetric in the adjacency matrix.

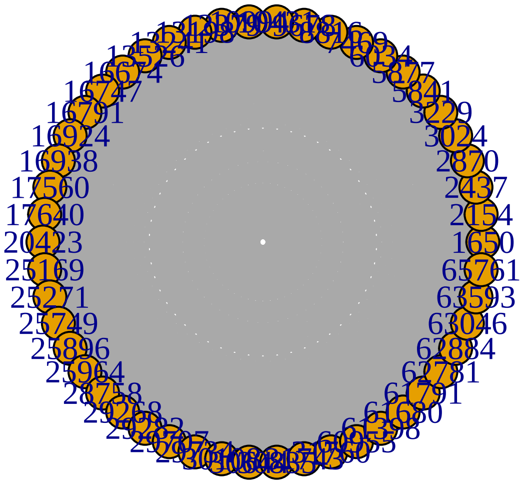
```r
# make the network and adjency matrix:
net <- graph_from_data_frame(d=df.50,directed=FALSE)
adj <- as.matrix(as_adjacency_matrix(net,attr="score"))
```

Next, let's plot the graph. Network plots can be represented in many ways. Here are a few examples (check the help file - component_wise, layout_as_bipartite, layout_as_star, layout_as_tree, layout_in_circle, layout_nicely, layout_on_grid, layout_randomly, layout_with_dh, layout_with_fr, layout_with_gem, layout_with_graphopt, layout_with_kk, layout_with_lgl, layout_with_mds, layout_with_sugiyama, layout_, merge_coords, norm_coords, normalize).

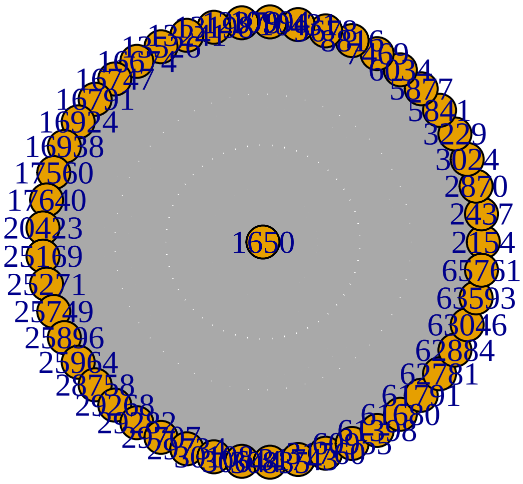
```r
l <- layout_on_sphere(net)
plot(net, layout=l)
```



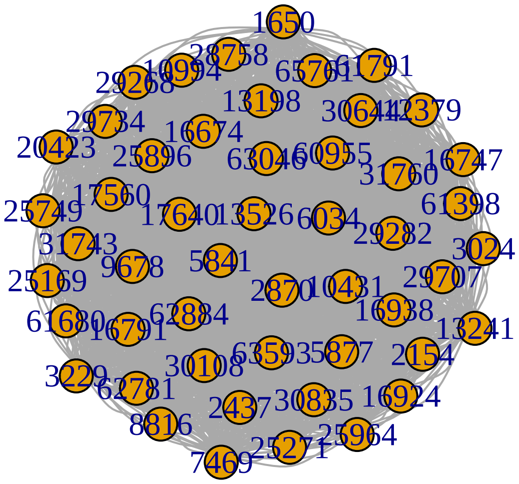
```r
l2 <- layout_in_circle(net)
plot(net, layout=l2)
```



```
l3 <- layout_as_star(net)
plot(net, layout=l3)
```



```
l4 <- layout_nicely(net)
plot(net,layout=l4)
```



Finally, we can fit a stochastic block modeling using the `blockmodels` package. The main inputs are the type of block model and the distribution. Since our edges are close to normal, we will use a Gaussion block model. Many times you have a network where the connections are 0/1 - in that case you would use a Bernoulli block model. Note that the block models in this packages can also handle covariates.

```r
# Fit the stochastic block model:
model.gaussian <- BM_gaussian(membership_type="SBM",adj=adj,
                              verbosity=1,explore_min=20,
                              explore_max=20,autosave="SBM_ML_Course",
                              plotting="")
model.gaussian$estimate()
```

```
## -> Estimation for 1 groups
##
-> Computation of eigen decomposition used for initalizations
##
## -> Pass 1
##
```

```
-> Pass 2
##
```

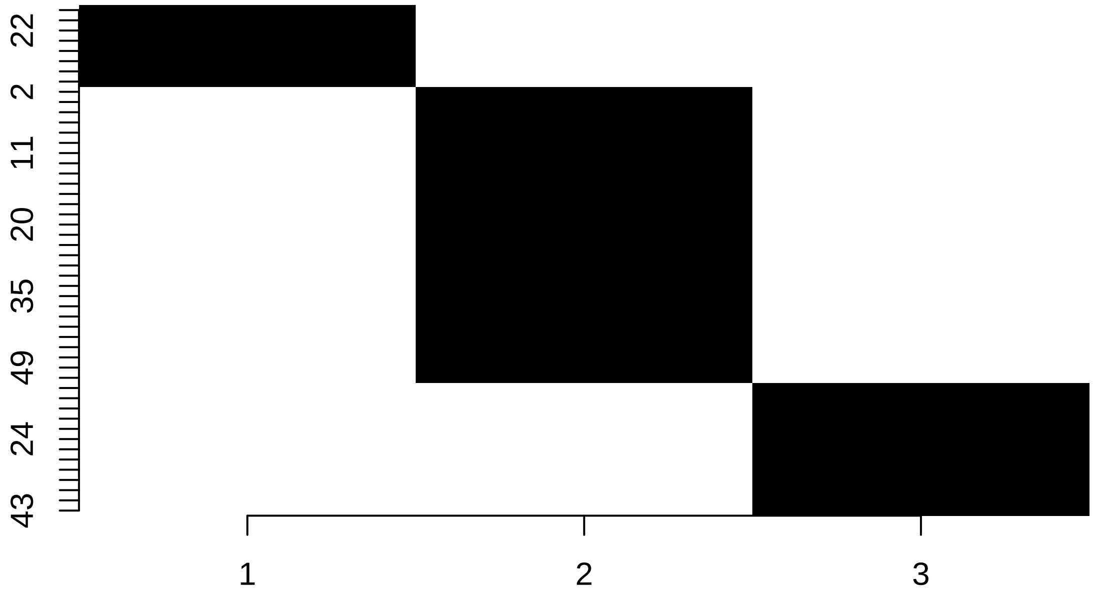
```
-> Pass 3
##
```

```
-> Pass 4
```

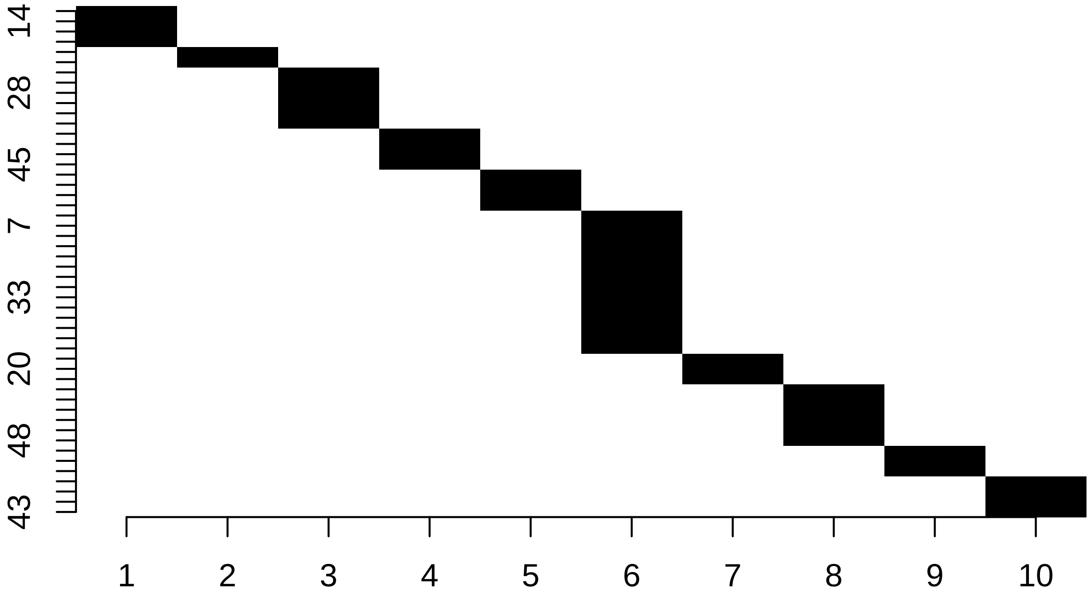
```
Z <- model.gaussian$memberships[[10]]$Z
Z.table <- apply(Z,2,table)
for (i in 1:length(Z.table)){
  message(sprintf('-------%d---------',i))
  print(Z.table[[i]])
}
```

```
## -------1---------
## [1] 46

## -------2---------

## [1] 4

## -------3---------

## [1] 48

## -------4---------

## [1] 2

## -------5---------

## [1] 44

## -------6---------

## [1] 6

## -------7---------

## [1] 46

## -------8---------

## [1] 4

## -------9---------

## [1] 46

## -------10---------

## [1] 4

## -------11---------

## [1] 36

## -------12---------

## [1] 14

## -------13---------

## [1] 47

## -------14---------

## [1] 3

## -------15---------

## [1] 44

## -------16---------

## [1] 6

## -------17---------

## [1] 47

## -------18---------

## [1] 3

## -------19---------
## [1] 46

## -------20---------

## [1] 4
```

```
model.gaussian$memberships[[3]]$plot()
```



```
model.gaussian$memberships[[10]]$plot()
```

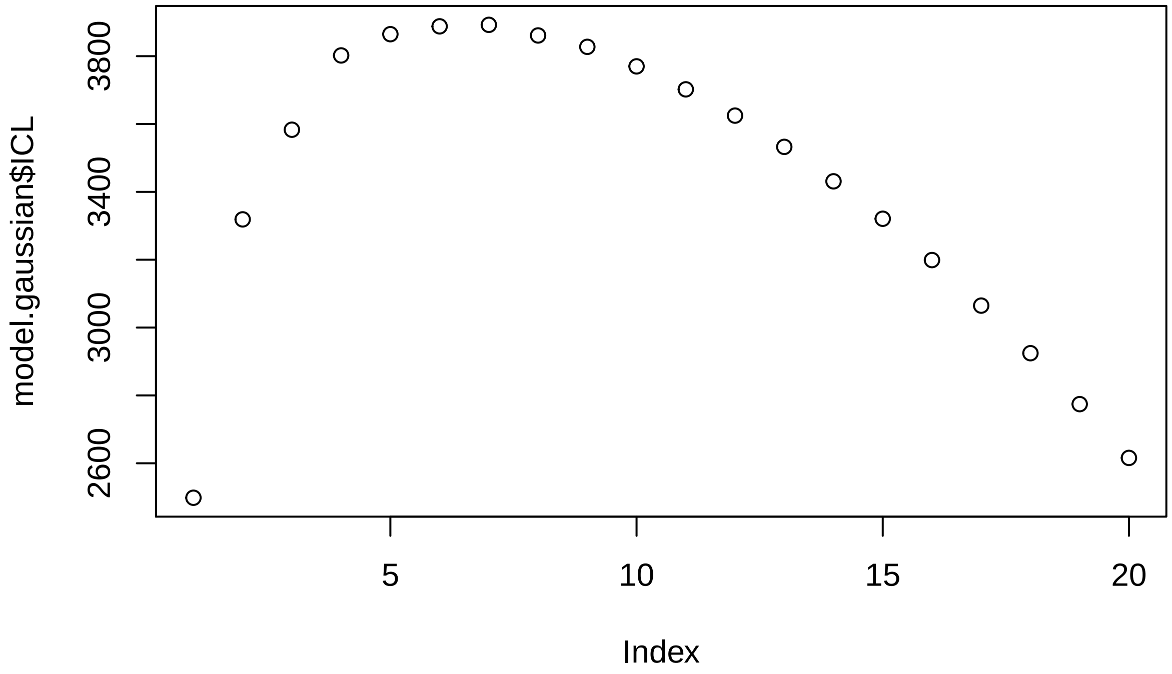


How to decide the number of communities? The SBM models output a statistic, ICL which is essentially the tuning parameter.

```
summary(model.gaussian)
```

```
##      Length       Class        Mode
##           1 BM_gaussian          S4
```

```
plot(model.gaussian$ICL)
```



```
print(which.max(model.gaussian$ICL))
```

```
## [1] 7
```

```
netB <- centr_betw(net, directed = TRUE, nobigint = TRUE, normalized = TRUE)$res
netC <- centr_clo(net, mode = "total", normalized = TRUE)$res
netD <- centr_degree(net, mode = "total", loops = TRUE,
                             normalized = TRUE)$res
netE <- centr_eigen(net, directed = TRUE, scale = TRUE,
                            options = arpack_defaults, normalized = TRUE)$`vector`
```

We can plot the ICL values or just find the maximum which in this case was 7 meaning seven communities gives the best fit.

References

Billio, Monica, Mila Getmansky, Andrew W Lo, and Loriana Pelizzon. 2012. “Econometric Measures of Connectedness and Systemic Risk in the Finance and Insurance Sectors.” *Journal of Financial Economics* 104 (3). Elsevier: 535–59.