anova_example

Zongyan Wang

Sunday, December 06, 2015

```
#Linear modelling on data
# Load package

library(ggplot2)
library(data.table)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
##
## The following objects are masked from 'package:data.table':
##
##     between, last
##
## The following objects are masked from 'package:stats':
##
##     filter, lag
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(agricolae)
library(igraph)
```

```
##
## Attaching package: 'igraph'
##
## The following object is masked from 'package:agricolae':
##
##     similarity
##
## The following objects are masked from 'package:dplyr':
##
##     %>%, as_data_frame, groups, union
##
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
##
## The following object is masked from 'package:base':
##
##     union
```

```
library('Matrix')
WISC = TRUE
NY = FALSE
setwd("~/projects/data")


if (NY) {
b= c(rep("character", 6),rep("factor",4), "numeric", rep("factor",6), "character", "character", "charac
wi = fread("ny_DT.txt",colClasses = b)
setkey(wi, NPI)
Ewi = fread("ny_Et.txt",sep = ",",  colClasses = c("character", "character","numeric", "numeric", "nume
setkey(Ewi, V1)
#Import data
phy_drugs = fread("ny_card.txt", sep=",")
}
if (WISC) {
  b= c(rep("character", 6),rep("factor",4), "numeric", rep("factor",6), "character", "character", "chara
  wi = fread("wi_DT.txt",colClasses = b)
  setkey(wi, NPI)
  Ewi = fread("wi_Et.txt",sep = ",",  colClasses = c("character", "character","numeric", "numeric", "nu
  setkey(Ewi, V1)
  #Import data
  phy_drugs = fread("wi_card_all.txt", sep=",")
}

hospitals = unique(phy_drugs[, 20])
setkey(phy_drugs,NPI)
#phy_drugs = phy_drugs[GENERIC_NAME=="METOPROLOL SUCCINATE"]
phy_drugs = phy_drugs[like(SPECIALTY_DESC,"Cardiology")]

#First aggregate ratios over physicians
grouped_by_physician = phy_drugs %>%
  group_by(NPI) %>%
  select(drug_name = DRUG_NAME, generic_name = GENERIC_NAME, total_claim_cnt = TOTAL_CLAIM_COUNT)

#Calculates ratios based on number of brand name vs total claims
phy_bg_ratios = summarise(grouped_by_physician,
                          bg_ratio = (sum(as.vector(total_claim_cnt[as.vector(drug_name) != as.vector(g
)


#Consider graph and its corresponding adjacency matrix
wi_card = wi[unique(as.character(phy_drugs$NPI))]
setkey(Ewi,V1)
Ewi = Ewi[V1 %in% unique(as.character(phy_drugs$NPI))]  # so cool! and fast!
setkey(Ewi,V2)
Ewi = Ewi[V2 %in% unique(as.character(phy_drugs$NPI))]
Ewi = Ewi[complete.cases (Ewi)]  #lots of NA's.  Have not inspected why.
el=as.matrix(Ewi)[,1:2] #igraph needs the edgelist to be in matrix format
g=graph.edgelist(el,directed = F) # this creates a graph.
E(g)$weight=as.numeric(Ewi$V3)

# Calculate models over npis and peers
```

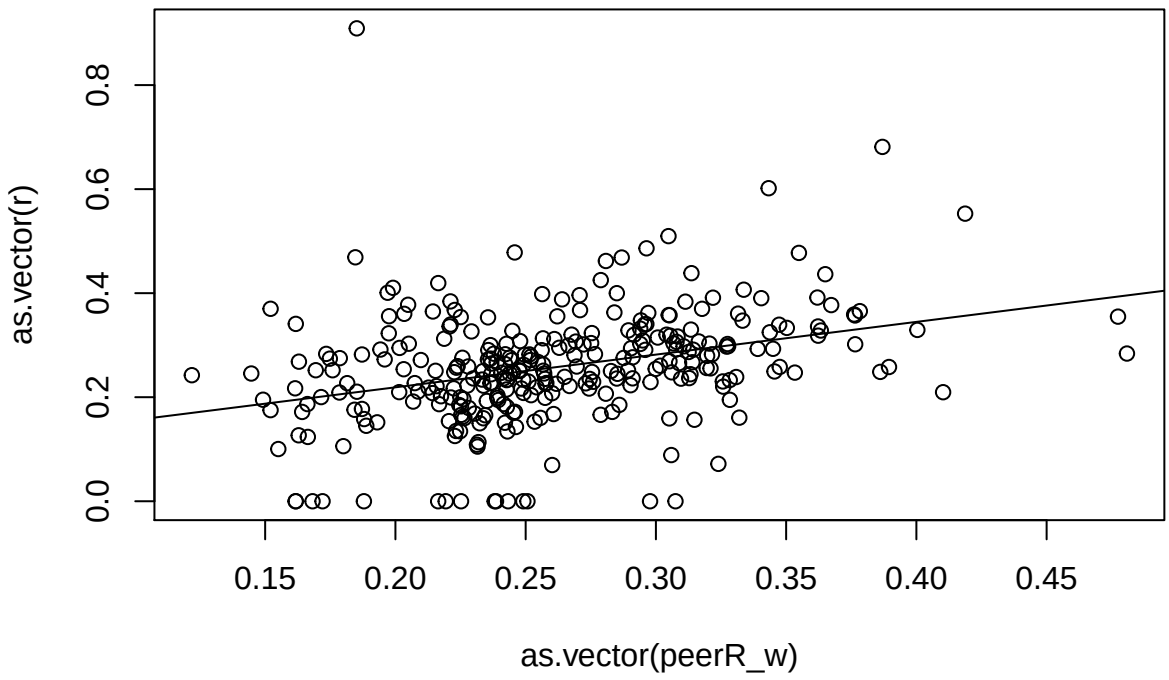
```
# Graph matrices (weighted)
numNPI = length(V(g))
A_w = Matrix(get.adjacency(graph = g, attr="weight"), sparse = TRUE) #
#L_w = as.matrix(graph.laplacian(g))
D_w = matrix(data=0,nrow = numNPI,ncol = numNPI)
D_w = Matrix(D_w, sparse = TRUE)
for (i in 1:numNPI){
  D_w[i,i] = 1/sum(A_w[i,])
  if (D_w[i,i]==0) {
    D_w[i,i]=1
  }
}
#D_w = solve(D_w)
```

Weighted analysis

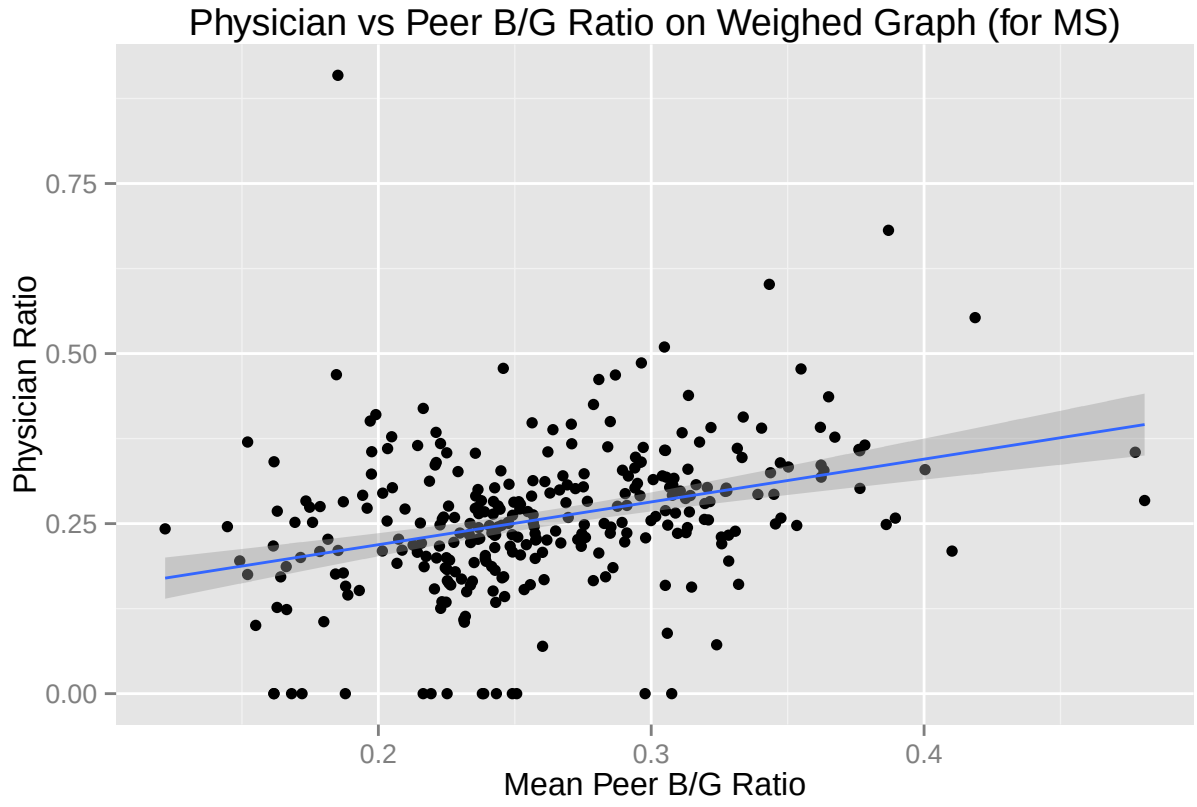
```
#Weighted sum of peer ratios matrix
r = as.vector(phy_bg_ratios[.(as.integer(V(g)$name))]$bg_ratio)
peerR_w = D_w%*%A_w%*%r
model_peerR_w = lm(r ~ as.vector(peerR_w))

## An example of how to visualize lm model

## plot with basic package
plot(as.vector(r) ~ as.vector(peerR_w))
abline(model_peerR_w)
```




```
ggplot(model_peerR_w, aes(x = as.vector(peerR_w), y = r)) +
  geom_point() +
  stat_smooth(method = "lm") + ggtitle("Physician vs Peer B/G Ratio on Weighed Graph (for MS)") +
  labs(x="Mean Peer B/G Ratio",y="Physician Ratio")
```




```
summary(model_peerR_w)
```

```
##
## Call:
## lm(formula = r ~ as.vector(peerR_w))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.28663 -0.04927 -0.00142  0.04800  0.69942
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         0.09321    0.02726   3.420 0.000713 ***
## as.vector(peerR_w)  0.62897    0.10186   6.175 2.13e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1035 on 303 degrees of freedom
## Multiple R-squared:  0.1118, Adjusted R-squared:  0.1088
## F-statistic: 38.13 on 1 and 303 DF,  p-value: 2.128e-09
```

```
cor.test(as.vector(peerR_w), r)
```

```
##
```

```
##  Pearson's product-moment correlation
##
## data:  as.vector(peerR_w) and r
## t = 6.1746, df = 303, p-value = 2.128e-09
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.2306655 0.4304580
## sample estimates:
##       cor
## 0.3343124
```

```
cor.test(as.vector(peerR_w), r, method = "spearman")
```

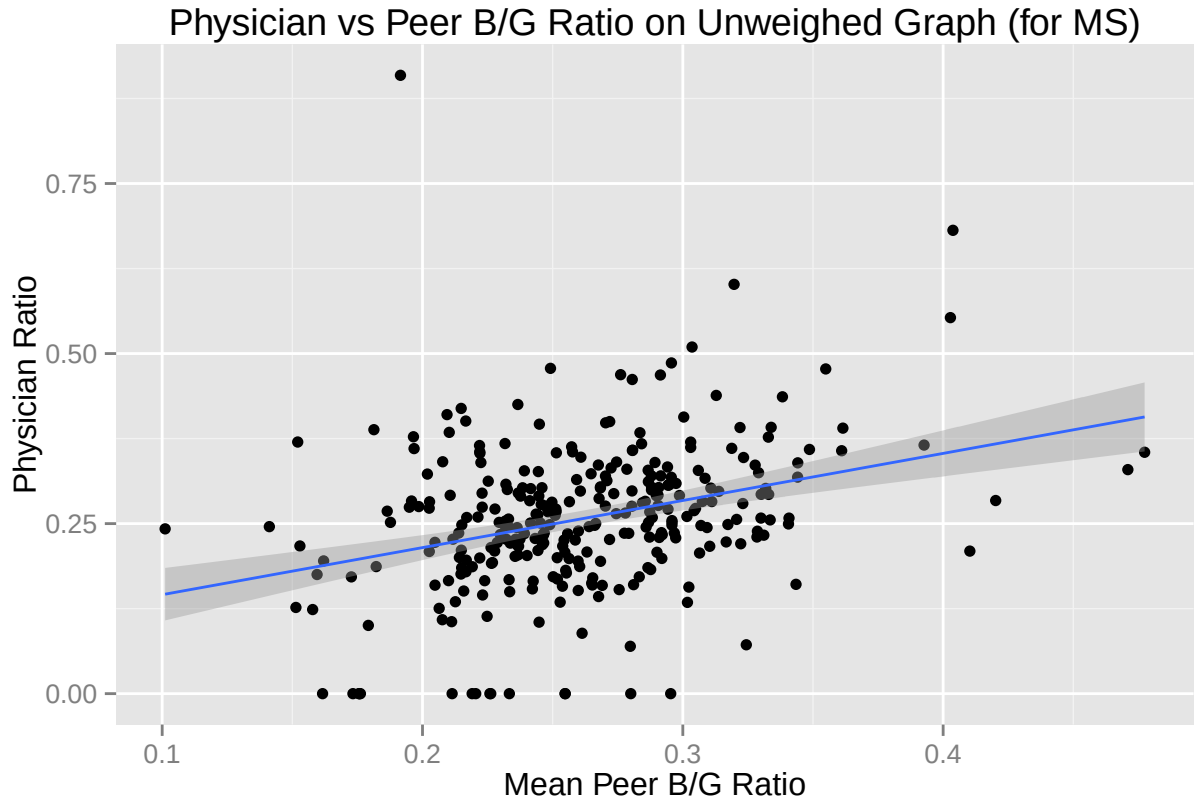
```
## Warning in cor.test.default(as.vector(peerR_w), r, method = "spearman"):
## Cannot compute exact p-value with ties
```

```
##
##  Spearman's rank correlation rho
##
## data:  as.vector(peerR_w) and r
## S = 2938800, p-value = 7.946e-12
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.3785164
```

Unweighted case

```
#Calculate matrices (unweighted)
A_u = as.matrix(get.adjacency(graph = g)) #
#L_u = as.matrix(graph.laplacian(g), weight = NA)
D_u = matrix(data=0,nrow = numNPI,ncol = numNPI)
for (i in 1:numNPI){
  D_u[i,i] = 1/sum(A_u[i,])
  if (D_u[i,i]==0) {
    D_u[i,i]=1
  }
}

#Weighted sum of peer ratios matrix
r = as.vector(phy_bg_ratios[.(as.integer(V(g)$name))]$bg_ratio)
peerR_u = D_u%*%A_u%*%r
model_peerR_u = lm(r ~ peerR_u)
# ## plot withm ggplot
ggplot(model_peerR_u, aes(x = as.vector(peerR_u), y = r)) +
   geom_point() +
   stat_smooth(method = "lm") + ggtitle("Physician vs Peer B/G Ratio on Unweighed Graph (for MS)") +
   labs(x="Mean Peer B/G Ratio",y="Physician Ratio")
```




```
summary(model_peerR_u)
```

```
##
## Call:
## lm(formula = r ~ peerR_u)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.28075 -0.05180 -0.00318  0.05335  0.70022
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.07623    0.03103   2.456   0.0146 *
## peerR_u      0.69250    0.11636   5.951 7.34e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.104 on 303 degrees of freedom
## Multiple R-squared:  0.1047, Adjusted R-squared:  0.1017
## F-statistic: 35.42 on 1 and 303 DF,  p-value: 7.345e-09
```

```
cor.test(as.vector(peerR_u), r)
```

```
##
```

```
##  Pearson's product-moment correlation
##
## data:  as.vector(peerR_u) and r
## t = 5.9513, df = 303, p-value = 7.345e-09
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.2191618 0.4205351
## sample estimates:
##       cor
## 0.3235066
```

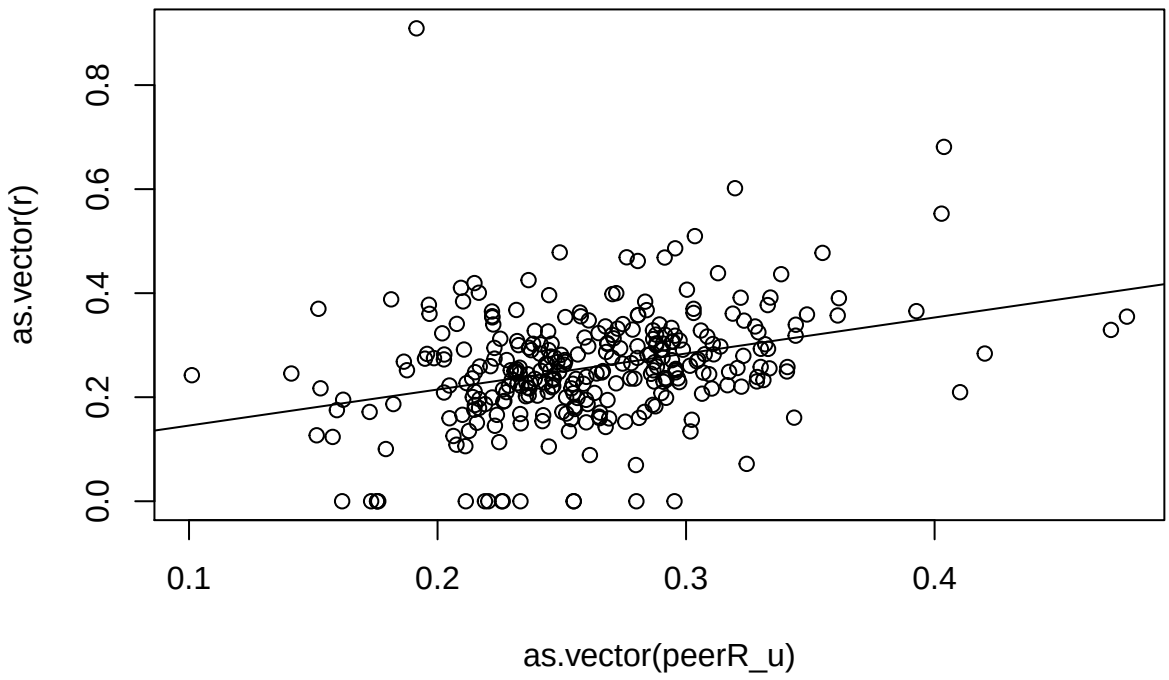
```
cor.test(as.vector(peerR_u), r, method = "spearman")
```

```
## Warning in cor.test.default(as.vector(peerR_u), r, method = "spearman"):
## Cannot compute exact p-value with ties
```

```
##
##  Spearman's rank correlation rho
##
## data:  as.vector(peerR_u) and r
## S = 3163600, p-value = 3.131e-09
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.3309898
```

```
## An example of how to visualize lm model

## plot with basic package
plot(as.vector(r) ~ as.vector(peerR_u))
abline(model_peerR_u)
```




```
#rr = as.vector(outer(r,r))
rr = matrix(nrow = length(r),ncol = length(r))
for (i in 1:length(r)){
  for (j in 1:length(r)){
    #rr[i,j] = (r[i]-r[j])^2
    rr[i,j] = (r[i]-r[j])^2
  }
}
```