Assignment 5

Jiahao Chen

11/7/2021

# DEA Analysis

#install.packages("Benchmarking")  
library(lpSolveAPI)  
library(Benchmarking)

## Loading required package: ucminf

## Loading required package: quadprog

x<-matrix(c(150,400,320,520,350,320,0.2,0.7,1.2,2.0,1.2,0.7),ncol = 2)  
y<-matrix(c(14000,14000,42000,28000,19000,14000,3500,21000,10500,42000,25000,15000),ncol = 2)  
colnames(y)<-c("REMPD","PRIPD")  
colnames(x)<-c("SHPD","SUPD")

# FDH  
d1<-dea(x,y,RTS = 0)  
d1

## [1] 1 1 1 1 1 1

peers(d1)

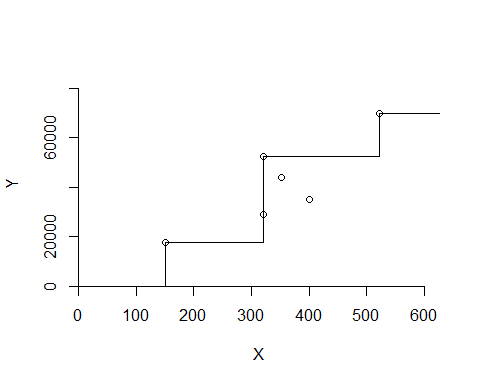
## peer1  
## [1,] 1  
## [2,] 2  
## [3,] 3  
## [4,] 4  
## [5,] 5  
## [6,] 6

lambda(d1)

## L1 L2 L3 L4 L5 L6  
## [1,] 1 0 0 0 0 0  
## [2,] 0 1 0 0 0 0  
## [3,] 0 0 1 0 0 0  
## [4,] 0 0 0 1 0 0  
## [5,] 0 0 0 0 1 0  
## [6,] 0 0 0 0 0 1

dea.plot.frontier(x,y,RTS = 0)

## Number '0' is 'fdh'



The results indicate all DMUS are efficient in FDH assumption.

#VRS  
d2<-dea(x,y,RTS = 1)  
d2

## [1] 1.0000 1.0000 1.0000 1.0000 1.0000 0.8963

peers(d2)

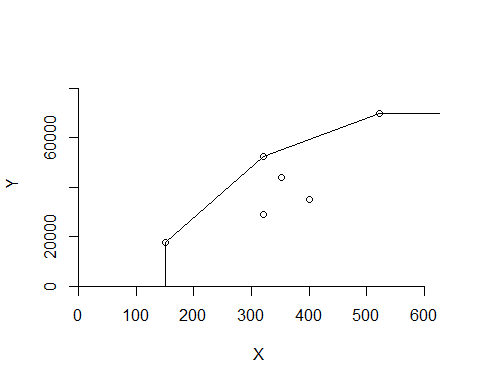
## peer1 peer2 peer3  
## [1,] 1 NA NA  
## [2,] 2 NA NA  
## [3,] 3 NA NA  
## [4,] 4 NA NA  
## [5,] 5 NA NA  
## [6,] 1 2 5

lambda(d2)

## L1 L2 L3 L4 L5  
## [1,] 1.0000000 0.0000000 0 0 0.0000000  
## [2,] 0.0000000 1.0000000 0 0 0.0000000  
## [3,] 0.0000000 0.0000000 1 0 0.0000000  
## [4,] 0.0000000 0.0000000 0 1 0.0000000  
## [5,] 0.0000000 0.0000000 0 0 1.0000000  
## [6,] 0.4014399 0.3422606 0 0 0.2562995

dea.plot.frontier(x,y,RTS = 1)

## Number '1' is 'vrs'



The results indicate that the DMU(6) is not efficent. The peers unit for DMU(6) are 1,2 and 5, with the relatives weight 0.40, 0.34 and 0.26.

#DRS  
d3<-dea(x,y,RTS = 2)  
d3

## [1] 1.0000 1.0000 1.0000 1.0000 0.9775 0.8675

peers(d3)

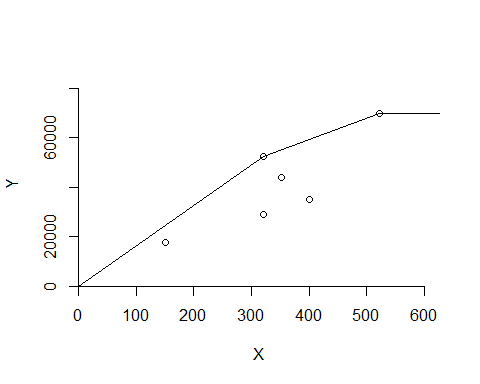
## peer1 peer2 peer3  
## [1,] 1 NA NA  
## [2,] 2 NA NA  
## [3,] 3 NA NA  
## [4,] 4 NA NA  
## [5,] 1 2 4  
## [6,] 1 2 4

lambda(d3)

## L1 L2 L3 L4  
## [1,] 1.0000000 0.00000000 0 0.0000000  
## [2,] 0.0000000 1.00000000 0 0.0000000  
## [3,] 0.0000000 0.00000000 1 0.0000000  
## [4,] 0.0000000 0.00000000 0 1.0000000  
## [5,] 0.2000000 0.08048142 0 0.5383307  
## [6,] 0.3428571 0.39499264 0 0.1310751

dea.plot.frontier(x,y,RTS = 2)

## Number '2' is 'drs'



The results indicate that the DMU(5) and DMU(6) are not efficent. The peers unit for DMU(5) are 1,2 and 4, with the relatives weight 0.20, 0.08 and 0.54. The peers unit for DMU(6) are 1,2 and 4, with the relatives weight 0.34, 0.39 and 0.13.

#CRS  
d4<-dea(x,y,RTS = 3)  
d4

## [1] 1.0000 1.0000 1.0000 1.0000 0.9775 0.8675

peers(d4)

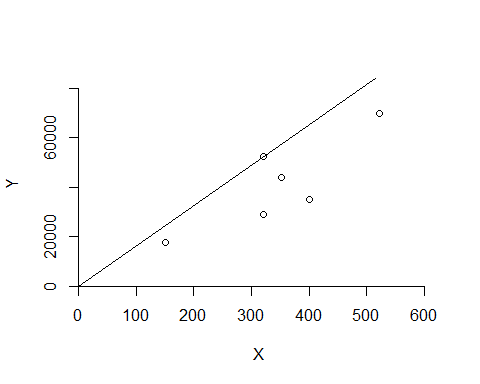
## peer1 peer2 peer3  
## [1,] 1 NA NA  
## [2,] 2 NA NA  
## [3,] 3 NA NA  
## [4,] 4 NA NA  
## [5,] 1 2 4  
## [6,] 1 2 4

lambda(d4)

## L1 L2 L3 L4  
## [1,] 1.0000000 0.00000000 0 0.0000000  
## [2,] 0.0000000 1.00000000 0 0.0000000  
## [3,] 0.0000000 0.00000000 1 0.0000000  
## [4,] 0.0000000 0.00000000 0 1.0000000  
## [5,] 0.2000000 0.08048142 0 0.5383307  
## [6,] 0.3428571 0.39499264 0 0.1310751

dea.plot.frontier(x,y,RTS = 3)

## Number '3' is 'crs'



The results indicate that the DMU(5) and DMU(6) are not efficent. The peers unit for DMU(5) are 1,2 and 4, with the relatives weight 0.20, 0.08 and 0.54. The peers unit for DMU(6) are 1,2 and 4, with the relatives weight 0.34, 0.39 and 0.13.

#IRS  
d5<-dea(x,y,RTS = 4)  
d5

## [1] 1.0000 1.0000 1.0000 1.0000 1.0000 0.8963

peers(d5)

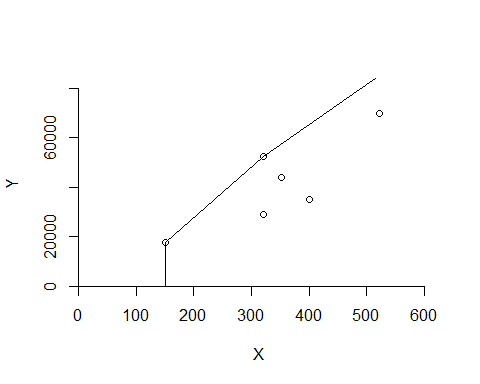
## peer1 peer2 peer3  
## [1,] 1 NA NA  
## [2,] 2 NA NA  
## [3,] 3 NA NA  
## [4,] 4 NA NA  
## [5,] 5 NA NA  
## [6,] 1 2 5

lambda(d5)

## L1 L2 L3 L4 L5  
## [1,] 1.0000000 0.0000000 0 0 0.0000000  
## [2,] 0.0000000 1.0000000 0 0 0.0000000  
## [3,] 0.0000000 0.0000000 1 0 0.0000000  
## [4,] 0.0000000 0.0000000 0 1 0.0000000  
## [5,] 0.0000000 0.0000000 0 0 1.0000000  
## [6,] 0.4014399 0.3422606 0 0 0.2562995

dea.plot.frontier(x,y,RTS = 4)

## Number '4' is 'irs'



The results indicate that the DMU(6) is not efficient. The peers unit for DMU(6) are 1,2 and 5, with the relatives weight 0.40, 0.34 and 0.26.

#FRH  
d6<-dea(x,y,RTS = 6)  
d6

## [1] 1 1 1 1 1 1

peers(d6)

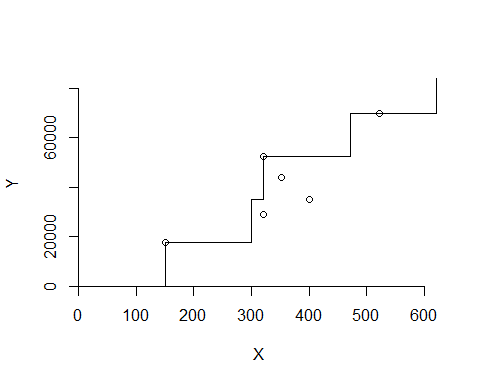
## peer1  
## [1,] 1  
## [2,] 2  
## [3,] 3  
## [4,] 4  
## [5,] 5  
## [6,] 6

lambda(d6)

## L1 L2 L3 L4 L5 L6  
## [1,] 1 0 0 0 0 0  
## [2,] 0 1 0 0 0 0  
## [3,] 0 0 1 0 0 0  
## [4,] 0 0 0 1 0 0  
## [5,] 0 0 0 0 1 0  
## [6,] 0 0 0 0 0 1

dea.plot.frontier(x,y,RTS = 6)

## Number '6' is 'add'



The results indicate all DMUS are efficient in FRH assumption.

sum<-matrix(c("FDH",0,0,0,"VRS","DMU6","1,2,5","0.4,0.34 and 0.26","DRS","DMU5 DMU6","1,2,4","0.20, 0.08 and 0.54, 0.34, 0.39 and 0.13","CRS","DMU5 DMU6","1,2,4","0.20, 0.08 and 0.54, 0.34, 0.39 and 0.13","IRS","DMU6","1,2,5","0.4,0.34 and 0.26","FRH",0,0,0),nrow = 4)  
colnames(sum)<-c("t1","t2","t3","t4","t5","t6")  
rownames(sum)<-c("DEA TYPE","Non-efficient","relative peers","relative weights")  
sum<-as.data.frame(sum)  
sum

## t1 t2 t3  
## DEA TYPE FDH VRS DRS  
## Non-efficient 0 DMU6 DMU5 DMU6  
## relative peers 0 1,2,5 1,2,4  
## relative weights 0 0.4,0.34 and 0.26 0.20, 0.08 and 0.54, 0.34, 0.39 and 0.13  
## t4 t5 t6  
## DEA TYPE CRS IRS FRH  
## Non-efficient DMU5 DMU6 DMU6 0  
## relative peers 1,2,4 1,2,5 0  
## relative weights 0.20, 0.08 and 0.54, 0.34, 0.39 and 0.13 0.4,0.34 and 0.26 0

Compared to these six methods, The all DMU in FDH and FRH are efficient. The DMU in IRS and VRS have the same peers and relative weights for unefficient DMUS. The DMU in DRS and CRS have the same condition too. Im my opinion, I will choose FDH and FRH methods. They could be efficient and fexible.

# Goal Programming

##1

lpec1<-make.lp(0,7)  
set.objfn(lpec1,c(20,15,25,-6,-6,0,-3))  
lp.control(lpec1,sense='max')

## $anti.degen  
## [1] "fixedvars" "stalling"   
##   
## $basis.crash  
## [1] "none"  
##   
## $bb.depthlimit  
## [1] -50  
##   
## $bb.floorfirst  
## [1] "automatic"  
##   
## $bb.rule  
## [1] "pseudononint" "greedy" "dynamic" "rcostfixing"   
##   
## $break.at.first  
## [1] FALSE  
##   
## $break.at.value  
## [1] 1e+30  
##   
## $epsilon  
## epsb epsd epsel epsint epsperturb epspivot   
## 1e-10 1e-09 1e-12 1e-07 1e-05 2e-07   
##   
## $improve  
## [1] "dualfeas" "thetagap"  
##   
## $infinite  
## [1] 1e+30  
##   
## $maxpivot  
## [1] 250  
##   
## $mip.gap  
## absolute relative   
## 1e-11 1e-11   
##   
## $negrange  
## [1] -1e+06  
##   
## $obj.in.basis  
## [1] TRUE  
##   
## $pivoting  
## [1] "devex" "adaptive"  
##   
## $presolve  
## [1] "none"  
##   
## $scalelimit  
## [1] 5  
##   
## $scaling  
## [1] "geometric" "equilibrate" "integers"   
##   
## $sense  
## [1] "maximize"  
##   
## $simplextype  
## [1] "dual" "primal"  
##   
## $timeout  
## [1] 0  
##   
## $verbose  
## [1] "neutral"

add.constraint(lpec1,c(6,4,5,-1,1,0,0),"=",50)  
add.constraint(lpec1,c(8,7,5,0,0,-1,1),"=",75)

write.lp(lpec1,filename = "Assignment5.lp",type = "lp")

solve(lpec1)

## [1] 0

get.objective(lpec1)

## [1] 225

get.variables(lpec1)

## [1] 0 0 15 25 0 0 0

The best way to reach the goal is that increase the number of employees. It might have the bad influence in the future.