Assignment 5

Lukas van der Watt

11/8/2021

**Problem 1**

R\_markdown file is as follows:

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

## Loading required package: lpSolveAPI

## Loading required package: ucminf

## Loading required package: quadprog

Now, we read our input data. We will read the data as input and output as vectors. Remember our problem had 5 DMUs with expenses as input and loans and deposits as outputs.

x <- matrix(c(150,400,320,520,350,320,0.2,0.7,1.2,2.0,1.2,0.7),ncol = 2)  
#z <- matrix(c(0.2,0.7,1.2,2.0,1.2,0.7))  
y <- matrix(c(14000,14000,42000,28000,19000,14000,3500,21000,10500,42000,25000,15000),ncol = 2)  
  
colnames(x) <- c("Staff Hours per Day","Supplies Per Day")  
colnames(y) <- c("Reimbursement","Privately Paid")  
  
  
x

## Staff Hours per Day Supplies Per Day  
## [1,] 150 0.2  
## [2,] 400 0.7  
## [3,] 320 1.2  
## [4,] 520 2.0  
## [5,] 350 1.2  
## [6,] 320 0.7

y

## Reimbursement Privately Paid  
## [1,] 14000 3500  
## [2,] 14000 21000  
## [3,] 42000 10500  
## [4,] 28000 42000  
## [5,] 19000 25000  
## [6,] 14000 15000

We now run the DEA analysis. We use the option of CRS, Constant Return to Scale. More on this later.

e <- dea(x,y,RTS = "crs") # provide the input and output   
e

## [1] 1.0000 1.0000 1.0000 1.0000 0.9775 0.8675

peers(e) # identify the peers

## 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(e) # identify the relative weights given to the peers

## 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.isoquant(x,y,RTS="crs") # plot the results

The results indicate that DMUs 1, 2, 3 and 4 are efficient. DMU(6) is only 87% efficient, and DMU(5) is 98% efficient. Further, the peer units for DMU(5) are 1,2 and 4, with relative weights 0.2, 0.08 and 0.54. Similarly for DMU(6), the peer units are 1,2 and 4, with weights 0.34,0.39 and 0.13 respectively.

e <- dea(x,y,RTS = "fdh") # provide the input and output   
e

## [1] 1 1 1 1 1 1

peers(e) # identify the peers

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

lambda(e) # identify the relative weights given to the peers

## 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.isoquant(x,y,RTS="fdh") # plot the results

The results indicate that all DMUs are efficient and all DMU’s carry the same weight.

e <- dea(x,y,RTS = "vrs") # provide the input and output   
e

## [1] 1.0000 1.0000 1.0000 1.0000 1.0000 0.8963

peers(e) # identify the peers

## 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(e) # identify the relative weights given to the peers

## 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.isoquant(x,y,RTS="vrs") # plot the results

The results indicate that DMUs 1, 2, 3, 4 and 5 are efficient. DMU(6) is 90% efficient. Further, the peer units for DMU(6) are 1,2 and 5, with relative weights 0.4 and 0.34 and 0.26.

e <- dea(x,y,RTS = "irs") # provide the input and output   
e

## [1] 1.0000 1.0000 1.0000 1.0000 1.0000 0.8963

peers(e) # identify the peers

## 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(e) # identify the relative weights given to the peers

## 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.isoquant(x,y,RTS="irs") # plot the results

The results indicate that DMUs 1, 2, 3 and 4 are efficient. DMU(6) is only 89% efficient. Further, the peer units for DMU(6) are 1,2 and 5, with relative weights 0.4 , 0.34 and 0.26 respectively.

e <- dea(x,y,RTS = "drs") # provide the input and output   
e

## [1] 1.0000 1.0000 1.0000 1.0000 0.9775 0.8675

peers(e) # identify the peers

## 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(e) # identify the relative weights given to the peers

## 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.isoquant(x,y,RTS="drs") # plot the results

The results indicate that DMUs 1, 2, 3 and 4 are efficient. DMU(5) is 98% efficient. Similarly DMU(6) is at an efficiency of 87%. Further, the peer units for DMU(5) are 1,2 and 4, with relative weights 0.2 , 0.08 and 0.54 respectively. The peer units for DMU(6) are 1,2 and 4, with relative weights 0.34 , 0.39 and 0.13 respectively.

e <- dea(x,y,RTS = "add") # provide the input and output   
e

## [1] 1 1 1 1 1 1

peers(e) # identify the peers

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

lambda(e) # identify the relative weights given to the peers

## 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

The results indicate that all DMUs are efficient and all DMU’s carry the same weight.

Tabled Summary of Results:

|  |  |  |  |
| --- | --- | --- | --- |
| DEA Assumption | DMU | Peers | Respective Weights |
| CRS | 1 | 1 | 1 |
| 2 | 2 | 1 |
| 3 | 3 | 1 |
| 4 | 4 | 1 |
| 5 | 1,2,4 | 0.2, 0.08, 0.54 |
| 6 | 1,2,4 | 0.34, 0.39, 0.13 |
| FDH | 1 | 1 | 1 |
| 2 | 2 | 1 |
| 3 | 3 | 1 |
| 4 | 4 | 1 |
| 5 | 5 | 1 |
| 6 | 6 | 1 |
| VRS | 1 | 1 | 1 |
| 2 | 2 | 1 |
| 3 | 3 | 1 |
| 4 | 4 | 1 |
| 5 | 5 | 1 |
| 6 | 1,2,5 | 0.40, 0.34, 0.26 |
| IRS | 1 | 1 | 1 |
| 2 | 2 | 1 |
| 3 | 3 | 1 |
| 4 | 4 | 1 |
| 5 | 5 | 1 |
| 6 | 1,2,5 | 0.40 ,0.34,0.26 |
| DRS | 1 | 1 | 1 |
| 2 | 2 | 1 |
| 3 | 3 | 1 |
| 4 | 4 | 1 |
| 5 | 1,2,4 | 0.20, 0.08,0.54 |
| 6 | 1,2,4 | 0.34, 0.39, 0.13 |
| FRH | 1 | 1 | 1 |
| 2 | 2 | 1 |
| 3 | 3 | 1 |
| 4 | 4 | 1 |
| 5 | 5 | 1 |
| 6 | 6 | 1 |

FDH & FRH is the highest efficiency of all the assumptions and is therefore favorable above the other DEA Assumptions. The FRH is larger than the FDH assumption and smaller that the CRS Assumption. Furthermore, the second best preference is the VRS Assumption as it has a good efficiency with majority DMU’s at 1 and the 6th DMU at 90%. The IRS and VRS assumptions are very similar, but the VRS is better as its DMU(6) has a 1% superiority to the IRS DMU(6). These Assumptions are based on minimum extrapolation.

**Problem 2**

Handwritten formulation:

A piece of paper with writing

Description automatically generated with low confidence

The .lp formulation is as follows:

Text

Description automatically generated

The R\_markdown file is as follows:

gp\_sl <- read.lp("Emax.lp")  
gp\_sl

## Model name:   
## x1 x2 x3 y1m y1p y2m y2p   
## Maximize 20 15 25 -6 -6 -3 -3   
## R1 6 4 5 -1 1 0 0 = 50  
## R2 8 7 5 0 0 -1 1 = 75  
## Kind Std Std Std Std Std Std Std   
## Type Real Real Real Real Real Real Real   
## Upper Inf Inf Inf Inf Inf Inf Inf   
## Lower 0 0 0 0 0 0 0

solve(gp\_sl) #Solving the lp formulation

## [1] 0

get.objective(gp\_sl) # getting the max objective function value This value is in millions of dollars

## [1] 225

get.variables(gp\_sl)# getting the variable in the order they are present in the function

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

The .lp formulation problem was solved successfully. Objective function Z = 225 million dollars

x1 = 0

x2 = 0

x3 = 15

y1m (positive) = 25

y1p (negative) = 0

y2m (positive) = 0

y2p (negative) = 0