npillana_QMM-5

11/5/2021

Loading required libraries

```
library(Benchmarking)

## Loading required package: lpSolveAPI

## Loading required package: ucminf

## Loading required package: quadprog

library(lpSolveAPI)

DMU1<- read.lp("C:/Users/nihar/OneDrive/Desktop/Fall
Assignments/QMM/Assignment 5/DMU1.lp")
DMU1

## Model name:
##                u1     u2     v1     v2
## Maximize    14000   3500      0      0
## R1          14000   3500   -150   -0.2  <=  0
## R2          14000  21000   -400   -0.7  <=  0
## R3          42000  10500   -320   -1.2  <=  0
## R4          28000  43000   -520     -2  <=  0
## R5          19000  25000   -350   -1.2  <=  0
## R6          14000  15000   -320   -0.7  <=  0
## R7              0      0    150    0.2   =  1
## Kind          Std    Std    Std    Std
## Type         Real   Real   Real   Real
## Upper         Inf    Inf    Inf    Inf
## Lower           0      0      0      0

solve(DMU1)

## [1] 0

get.objective(DMU1)

## [1] 1

get.variables(DMU1)

## [1] 7.142857e-05 0.000000e+00 5.172414e-03 1.120690e+00
```

The lp is able to acheive maximum efficiency 1 for DMU1. The proposed inputs and outputs when we use the weights 5.17 and 1.12 for the outputs, 7.14 and 0.00 for the input for maximum efficiency.

```r
DMU2<- read.lp("C:/Users/nihar/OneDrive/Desktop/Fall
Assignments/QMM/Assignment 5/DMU2.lp")
DMU2

## Model name:
##               u1     u2     v1     v2
## Maximize   14000  21000      0      0
## R1         14000   3500   -150   -0.2  <=  0
## R2         14000  21000   -400   -0.7  <=  0
## R3         42000  10500   -320   -1.2  <=  0
## R4         28000  43000   -520     -2  <=  0
## R5         19000  25000   -350   -1.2  <=  0
## R6         14000  15000   -320   -0.7  <=  0
## R7             0      0    400    0.7   =  1
## Kind         Std    Std    Std    Std
## Type        Real   Real   Real   Real
## Upper        Inf    Inf    Inf    Inf
## Lower          0      0      0      0

solve(DMU2)

## [1] 0

get.objective(DMU2)

## [1] 1

get.variables(DMU2)

## [1] 0.000000e+00 4.761905e-05 1.299694e-03 6.858890e-01
```

The lp is able to acheive maximum efficiency 1 for DMU2. The proposed inputs and outputs when we use the weights 1.29 and 6.8 for the outputs, 0.00 and 4.7 for the input for maximum efficiency.

```r
DMU3<- read.lp("C:/Users/nihar/OneDrive/Desktop/Fall
Assignments/QMM/Assignment 5/DMU3.lp")
DMU3

## Model name:
##               u1     u2     v1     v2
## Maximize   42000  10500      0      0
## R1         14000   3500   -150   -0.2  <=  0
## R2         14000  21000   -400   -0.7  <=  0
## R3         42000  10500   -320   -1.2  <=  0
## R4         28000  43000   -520     -2  <=  0
## R5         19000  25000   -350   -1.2  <=  0
## R6         14000  15000   -320   -0.7  <=  0
## R7             0      0    320    1.2   =  1
## Kind         Std    Std    Std    Std
## Type        Real   Real   Real   Real
```

```
## Upper        Inf     Inf     Inf     Inf
## Lower          0       0       0       0
```

```
solve(DMU3)
```

```
## [1] 0
```

```
get.objective(DMU3)
```

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

```
get.variables(DMU3)
```

```
## [1] 2.380952e-05 0.000000e+00 1.724138e-03 3.735632e-01
```

The lp is able to acheive maximum efficiency 1 for DMU3. The proposed inputs and outputs when we use the weights 1.7 and 3.7 for the outputs, 2.3 and 0.00 for the input for maximum efficiency.

```
DMU4<- read.lp("C:/Users/nihar/OneDrive/Desktop/Fall Assignments/QMM/Assignment 5/DMU4.lp")
DMU4
```

```
## Model name:
##                 u1     u2     v1     v2
## Maximize     28000  42000      0      0
## R1           14000   3500   -150   -0.2  <=  0
## R2           14000  21000   -400   -0.7  <=  0
## R3           42000  10500   -320   -1.2  <=  0
## R4           28000  43000   -520     -2  <=  0
## R5           19000  25000   -350   -1.2  <=  0
## R6           14000  15000   -320   -0.7  <=  0
## R7               0      0    520      2   =  1
## Kind           Std    Std    Std    Std
## Type          Real   Real   Real   Real
## Upper          Inf    Inf    Inf    Inf
## Lower            0      0      0      0
```

```
solve(DMU4)
```

```
## [1] 0
```

```
get.objective(DMU4)
```

```
## [1] 0.9836182
```

```
get.variables(DMU4)
```

```
## [1] 1.055657e-05 1.638177e-05 1.923077e-03 0.000000e+00
```

The lp is able to acheive efficiency 0.98 with DMU4. The proposed inputs and outputs when we use the weights 1.9 and 0.0 for the outputs, 1.05 and 1.63 for the input for maximum efficiency. Even though we provide the greatest weight to deposits, DMU4 is not efficient.

```r
DMU5<- read.lp("C:/Users/nihar/OneDrive/Desktop/Fall
Assignments/QMM/Assignment 5/DMU5.lp")
DMU5

## Model name:
##                u1     u2     v1     v2
## Maximize  19000  25000      0      0
## R1        14000   3500   -150   -0.2  <=  0
## R2        14000  21000   -400   -0.7  <=  0
## R3        42000  10500   -320   -1.2  <=  0
## R4        28000  43000   -520     -2  <=  0
## R5        19000  25000   -350   -1.2  <=  0
## R6        14000  15000   -320   -0.7  <=  0
## R7            0      0    350    1.2   =  1
## Kind        Std    Std    Std    Std
## Type       Real   Real   Real   Real
## Upper       Inf    Inf    Inf    Inf
## Lower         0      0      0      0

solve(DMU5)

## [1] 0

get.objective(DMU5)

## [1] 0.961371

get.variables(DMU5)

## [1] 1.117916e-05 2.995868e-05 1.033058e-03 5.320248e-01
```

The lp is able to achieve efficiency 0.96 for DMU5. The proposed inputs and outputs when we use the weights 1.03 and 5.3 for the outputs, 1.11 and 2.99 for the input for maximum efficiency. Even though we provide the greatest weight to deposits, DMU5 is not efficient.

```r
DMU6<- read.lp("C:/Users/nihar/OneDrive/Desktop/Fall
Assignments/QMM/Assignment 5/DMU6.lp")
DMU6

## Model name:
##                u1     u2     v1     v2
## Maximize  14000  15000      0      0
## R1        14000   3500   -150   -0.2  <=  0
## R2        14000  21000   -400   -0.7  <=  0
## R3        42000  10500   -320   -1.2  <=  0
## R4        28000  43000   -520     -2  <=  0
## R5        19000  25000   -350   -1.2  <=  0
## R6        14000  15000   -320   -0.7  <=  0
## R7            0      0    320    0.7   =  1
## Kind        Std    Std    Std    Std
## Type       Real   Real   Real   Real
```

```
## Upper        Inf     Inf     Inf     Inf
## Lower          0       0       0       0
```

```
solve(DMU6)
```

```
## [1] 0
```

```
get.objective(DMU6)
```

```
## [1] 0.8618663
```

```
get.variables(DMU6)
```

```
## [1] 1.590217e-05 4.261572e-05 1.469508e-03 7.567965e-01
```

The lp is able to acheive efficiency 0.86 for DMU6. The proposed inputs and outputs when we use the weights 1.46 and 7.56 for the outputs, 1.59 and 4.26 for the input for maximum efficiency. Even though we provide the greatest weight to deposits, DMU6 is not efficient.

First we will define our inputs and outputs as vectors . we have 2 inputs (Staff hours, Supplies) and 2 outputs("Reimbursed Patient_Days","Privately Paid Patient_Day)

```
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(14000,14000,42000,28000,19000,14000,3500,21000,10500,42000,25000,150
00),ncol = 2)
colnames(x) <- c("Staff_Hours", "Supplies")
colnames(y) <- c("Reimbursed Patient_Days","Privately Paid Patient_Days")
print(x)
```

```
##      Staff_Hours Supplies
## [1,]         150      0.2
## [2,]         400      0.7
## [3,]         320      1.2
## [4,]         520      2.0
## [5,]         350      1.2
## [6,]         320      0.7
```

```
print(y)
```

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

```
Matrix<- cbind(x,y)
row.names(Matrix) = c("Faci1", "Faci2", "Faci3", "Faci4", "Faci5", "Faci6")
Matrix
```

```
##       Staff_Hours Supplies Reimbursed Patient_Days Privately Paid
Patient_Days
## Faci1         150      0.2                     14000
3500
## Faci2         400      0.7                     14000
21000
## Faci3         320      1.2                     42000
10500
## Faci4         520      2.0                     28000
42000
## Faci5         350      1.2                     19000
25000
## Faci6         320      0.7                     14000
15000
```

Formulate and perform DEA analysis under all DEA assumptions of FDH, CRS, VRS, IRS, DRS, and FRH.

```
#Free disposability hull
FDH <- dea(x,y, RTS = "fdh")
FDH

## [1] 1 1 1 1 1 1

peers(FDH)

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

FDH_Weights <- lambda(FDH)
```

The peer for each facility is same as the peer.

```
#Constant returns to scale, convexity and free disposability
CRS <- dea(x,y, RTS = "crs")
CRS

## [1] 1.0000 1.0000 1.0000 1.0000 0.9775 0.8675

#Identify Peers
peers(CRS)

##      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
```

```r
#Identify Lambda
CRS_Weights <- lambda(CRS)
```

The results show DMU 1,2,3,4 are efficient and DMU 5 is 0.9775, DMU 6 0.867 The peer for 5 and 6 are 1,2,3

```r
#Variable returns to scale, convexity and free disposability
VRS <- dea(x,y, RTS = "vrs")
VRS
```

```
## [1] 1.0000 1.0000 1.0000 1.0000 1.0000 0.8963
```

```r
peers(VRS)
```

```
##      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
```

```r
VRS_Weights <- lambda(VRS)
```

All facilities are efficient except DMU5 which is 0.8963. The peer for 6 are 1,2,5

```r
#Increasing returns to scale, (up-scaling, but not down-scaling), convexity
and free disposability
IRS <- dea(x,y, RTS = "irs")
IRS
```

```
## [1] 1.0000 1.0000 1.0000 1.0000 1.0000 0.8963
```

```r
peers(IRS)
```

```
##      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
```

```r
IRS_Weights <- lambda(IRS)
```

All facilities are efficient except DMU5 which is 0.8963. The peer for 6 are 1,2,5

```
#Decreasing returns to scale, convexity, down-scaling and free disposability
DRS <- dea(x,y, RTS = "drs")
DRS

## [1] 1.0000 1.0000 1.0000 1.0000 0.9775 0.8675

peers(DRS)

##      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

DRS_Weights <- lambda(DRS)
```

The results show DMU 1,2,3,4 are efficient and DMU 5 is 0.9775, DMU 6 0.867 The peer for 5 and 6 are 1,2,4

```
FRH <- dea(x,y, RTS="add")
FRH

## [1] 1 1 1 1 1 1

peers(FRH)

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

FRH_Weights <- lambda(FRH)
```

All facilities are efficient. The peer for each facility is same as the peer.

```
as.data.frame(Matrix)

##       Staff_Hours Supplies Reimbursed Patient_Days Privately Paid
Patient_Days
## Faci1         150      0.2                   14000
3500
## Faci2         400      0.7                   14000
21000
## Faci3         320      1.2                   42000
10500
## Faci4         520      2.0                   28000
42000
## Faci5         350      1.2                   19000
```

```
25000
## Faci6         320      0.7                   14000
15000
```

```
DataFrame<- data.frame(CRS = c(1.0000, 1.0000, 1.0000, 1.0000, 0.9775,
0.8675), FDH = c(1, 1, 1 ,1 ,1 ,1), VRS =c(1.0000, 1.0000, 1.0000, 1.0000
,1.0000, 0.8963), IRS = c(1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 0.8963),
DRS = c(1.0000, 1.0000, 1.0000 ,1.0000 ,0.9775, 0.8675), FRH = c(1, 1, 1 ,1,
1, 1))
DataFrame
```

```
##      CRS FDH    VRS    IRS    DRS FRH
## 1 1.0000   1 1.0000 1.0000 1.0000   1
## 2 1.0000   1 1.0000 1.0000 1.0000   1
## 3 1.0000   1 1.0000 1.0000 1.0000   1
## 4 1.0000   1 1.0000 1.0000 1.0000   1
## 5 0.9775   1 1.0000 1.0000 0.9775   1
## 6 0.8675   1 0.8963 0.8963 0.8675   1
```

From the above output the Facilities 1,2,3,4 are fully efficient for all the assumptions and Facilities 5,6 are not efficient. Facility 5 is fully efficient for FDH, VRS, IRS and FRH assumptions. It is observed that 97.7% efficient for CRS and DRS assumptions. Facility 6 is fully efficient for FDH and FRS assumptions. For Facility 6 CRS and DRS assumptions 86.7% efficient. For Facility 6 IRS and VRS assumptions 89.6% efficient.

DEA Analysis Summary for Hope Vally Health Care Association: Under FDH and FRH all facilities are efficient

Question 2 : GOAL PRORAMMING

Maximize Z = P - 6C - 3D, where P = total (discounted) profit over the life of the new products, C = change (in either direction) in the current level of employment, D = decrease (if any) in next year's earnings from the current year's level.

Profit P is defined as: P = 20x1 +15x2 + 25x3

Employment level is defined as : 6x1 + 4x2 + 5x3 = 50

Next year Earnings goal is defined as: 8x1 + 7x2 + 5x3 >=75

1) Model_Formulation:

Let us consider y1 - Employment Level minus the target and y2 - Next Year Earnings minus the Target y1+ - Penalty for employment level goal exceeding 50 y1- - Penalty for employment level goal decreasing below 50 y2+ - Exceed the next year earnings y2- - Penalty for not reaching the next year earnings

y1 = 6x1 + 4x2 + 5x3 - 50 y2 = 8x1 + 7x2 + 5x3 -75

For Employment level goal

y1 = y1+ - y1- where y1+, y1- >=0 y1+ - y1-= 6x1 + 4x2 + 5x3 - 50

For Next year earnings goal

y2 = y2+ - y2- where y2+, y2- >=0 y2+ - y2- = 8x1 + 7x2 + 5x3 -75

Final Formulation is expressed as

Max P = 20x1 +15x2 + 25x3 6x1 + 4x2 + 5x3 - (y1+ - y1-) = 50 8x1 + 7x2 + 5x3 - (y2+ - y2-) = 75

Where, xj >=0, where j=1,2,3 yi + >=0, where i= 1,2 yi - >=0, where i= 1,2

2)Managements objective function Objective Function

Maximize Z = P - 6C - 3D

Objective function in terms of x1, x2, x3, y1+, y1- , y2+ and y2- Max Z = 20x1 +15x2 + 25x3 - 6y1+ - 6y1- - 3y2- 6x1 + 4x2 + 5x3 - (y1+ + y1-) = 50 8x1 + 7x2 + 5x3 - (y2+ + y2-) = 75 Where, xj >=0 where j=1,2,3 yi + >=0 where i= 1,2 yi - >=0 where i= 1,2

3) Formulate and solve the linear programming model

```
GoalProgram<- read.lp("C:/Users/nihar/OneDrive/Desktop/Fall Assignments/QMM/Assignment 5/Emax.lp")
GoalProgram

## Model name:
##              x1    x2    x3   y1p   y1m   y2m   y2p
## Maximize     20    15    25    -6    -6    -3     0
## 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(GoalProgram)

## [1] 0

get.objective(GoalProgram)

## [1] 225

get.variables(GoalProgram)

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

```
get.constraints(GoalProgram)

## [1] 50 75
```

The penalty is 225 if you are not satisfying the goals on the objective function. The results show that x1 = 0, x2 = 0, x3 = 15, y1+ = 25, y1− = 0, y2+ = 0, y2− = 0, which explains the Next years Earnings (y2) expectations are fully satisfied. Emax need to produce 15 units of product 3 and none of product 1 and 2 to achieve 225 millions in profit.