yli130_Assignment5

Yanxi Li

11/6/2021

Question 1

Formulate input & output

```
# run DEA analysis using benchmarking library.
library(Benchmarking)
```

```
## Warning: package 'Benchmarking' was built under R version 4.0.5
```

```
## Loading required package: lpSolveAPI
```

```
## Loading required package: ucminf
```

```
## Loading required package: quadprog
```

```
# import input and output data as vectors.
# we have 2 input and 2 output columns, make the ncol = 2.
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)

# give input & output columns names
colnames(x) <- c('Staff_Hours_per_Day', 'Supplies_per_Day')
colnames(y) <- c('Reimbursed_Patient-Days','Privately_Paid_Patient-Days')

# show input & output columns
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
```

```r
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
```

Perform DEA analysis under 6 assumptions

we have 6 assumptions which are FDH, CRS, VRS, IRS, DRS, FRH. In the DEA efficiency document, RTS for FRH names 'add'.

```r
# FDH
e_FDH <- dea(x, y, RTS = 'fdh')          # provide input output and assumption
e_FDH                                    # show efficiency
```

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

```r
peers(e_FDH)                             # identify the peers
```

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

```r
lambda(e_FDH)                            # 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
```

For FDH assumption, all of the facilities are efficient.

```r
# CRS
e_CRS <- dea(x, y, RTS = 'crs')
e_CRS
```

```
## [1] 1.0000 1.0000 1.0000 1.0000 0.9775 0.8675
```

```
peers(e_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
```

```
lambda(e_CRS)
```

```
##             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
```

For CRS assumption, DMU 1,2,3,4 are efficient, DMU(5) is 0.9775 efficient and DMU(6) is only 0.8675 efficient.

The peer units for DMU(5) are [1,2,4] with relative weight [0.2, 0.0805, 0.5383]. For DMU(6) are also [1,2,4] but with relative weight [0.3429, 0.3950, 0.1311].

```
# VRS
e_VRS <- dea(x, y, RTS = 'vrs')
e_VRS
```

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

```
peers(e_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
```

```
lambda(e_VRS)
```

```
##             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
```

For VRS assumption, DMU 1,2,3,4,5 are efficient. Only DMU(6) is 0.8963 efficient.

The peer units for DMU(6) are [1,2,5] with relative weight [0.4014, 0.3423, 0.2563].

```r
# IRS
e_IRS <- dea(x, y, RTS = 'irs')
e_IRS
```

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

```r
peers(e_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
lambda(e_IRS)
```

```
##             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
```

IRS assumption is same with the VRS, DMU 1,2,3,4,5 are efficient. Only DMU(6) is 0.8963 efficient.

The peer units for DMU(6) are [1,2,5] with relative weight [0.4014, 0.3423, 0.2563].

```r
# DRS
e_DRS <- dea(x, y, RTS = 'drs')
e_DRS
```

```
## [1] 1.0000 1.0000 1.0000 1.0000 0.9775 0.8675
```

```r
peers(e_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
```

```r
lambda(e_DRS)
```

```
##             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
```

DRS assumption is the same with CRS, DMU 1,2,3,4 are efficient, DMU(5) is 0.9775 efficient and DMU(6) is only 0.8675 efficient.

The peer units for DMU(5) are [1,2,4] with relative weight [0.2, 0.0805, 0.5383]. For DMU(6) are also [1,2,4] but with relative weight [0.3429, 0.3950, 0.1311].

```r
# FRH
e_FRH <- dea(x, y, RTS = 'add')          # assumption name is 'add'
e_FRH
```

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

```r
peers(e_FRH)
```

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

```r
lambda(e_FRH)
```

```
##      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
```

For FRH assumption, all of the facilities are efficient.

The results show that only for FDH and FRH, all facilities are efficient. The other assumptions for 5 or 6 facility is not efficient.

Summarize in Tabular format

Efficiency data-frame

```
# Create columns
Assumption_Name <- c('FDH', 'CRS', 'VRS', 'IRS', 'DRS', 'FRH')
Facility_1_Effi <- c(1, 1, 1, 1, 1, 1)
Facility_2_Effi <- c(1, 1, 1, 1, 1, 1)
Facility_3_Effi <- c(1, 1, 1, 1, 1, 1)
Facility_4_Effi <- c(1, 1, 1, 1, 1, 1)
Facility_5_Effi <- c(1, 0.9775, 1, 1, 0.9775, 1)
Facility_6_Effi <- c(1, 0.8675, 0.8963, 0.8963, 0.8675, 1)
# Create data-frame
Efficiency_Dataframe <- data.frame(Assumption_Name, Facility_1_Effi, Facility_2_Effi, Facility_3_Effi,
                                   Facility_4_Effi, Facility_5_Effi, Facility_6_Effi)
# show efficiency data-frame
Efficiency_Dataframe
```

```
##   Assumption_Name Facility_1_Effi Facility_2_Effi Facility_3_Effi
## 1             FDH               1               1               1
## 2             CRS               1               1               1
## 3             VRS               1               1               1
## 4             IRS               1               1               1
## 5             DRS               1               1               1
## 6             FRH               1               1               1
##   Facility_4_Effi Facility_5_Effi Facility_6_Effi
## 1               1          1.0000          1.0000
## 2               1          0.9775          0.8675
## 3               1          1.0000          0.8963
## 4               1          1.0000          0.8963
## 5               1          0.9775          0.8675
## 6               1          1.0000          1.0000
```

Not efficient peers & weight

```
# create columns
Assumption_Name2 <- c('CRS', 'VRS', 'IRS', 'DRS')
Fac5_Effi <- c(0.9775, 1, 1, 0.9775)
Fac5_Peers <- c(124, 5/NA, 5/NA, 124)
Fac5_Weight <- c('0.2, 0.0805, 0.5383', 1/NA, 1/NA, '0.2, 0.0805, 0.5383')
Fac6_Effi <- c(0.8675, 0.8963, 0.8963, 0.8675)
Fac6_Peers <- c(124, 125, 125, 124)
Fac6_Weight <- c('0.3429, 0.3950, 0.1311', '0.4014, 0.3423, 0.2563',
                 '0.4014, 0.3423, 0.2563', '0.3429, 0.3950, 0.1311')

# create data-frame
Peers_Weight_Dataframe <- data.frame(Assumption_Name2, Fac5_Effi, Fac5_Peers, Fac5_Weight,
                                     Fac6_Effi, Fac6_Peers, Fac6_Weight)
# show results
Peers_Weight_Dataframe
```

```
##   Assumption_Name2 Fac5_Effi Fac5_Peers         Fac5_Weight Fac6_Effi
## 1              CRS    0.9775        124 0.2, 0.0805, 0.5383    0.8675
```

```
## 2              VRS    1.0000         NA                 <NA>    0.8963
## 3              IRS    1.0000         NA                 <NA>    0.8963
## 4              DRS    0.9775        124 0.2, 0.0805, 0.5383    0.8675
##   Fac6_Peers            Fac6_Weight
## 1        124 0.3429, 0.3950, 0.1311
## 2        125 0.4014, 0.3423, 0.2563
## 3        125 0.4014, 0.3423, 0.2563
## 4        124 0.3429, 0.3950, 0.1311
```

Compare and Contrast Results

```
FDH & FRH make every data point efficient which corresponds to what we learned that
                                                             FDH and  FRH can include more data,

CRS & DRS have the same results. VRS & IRS have the same results.

For all assumptions, facility from 1-4 are all efficient, only different on 5 and 6.
```

Question 2

Expressions

```
P = 20 x1 + 15 x2 + 25 x3;

6 x1 + 4 x2 + 5 x3 - (y1p - y1m) = 50;

8 x1 + 7 x2 + 5 x3 - (y2p - y2m) = 75;
```

Objective Function

```
Max: 20 x1 + 15 x2 + 25 x3 - 6 y1p - 6 y1m - 3 y2m;
```

Formulate and Solve the model

```
LP file

// Objective function
max: 20 x1 + 15 x2 + 25 x3 - 6 y1p - 6 y1m - 3 y2m;

// Constraints
6 x1 + 4 x2 + 5 x3 + y1m - y1p = 50;
8 x1 + 7 x2 + 5 x3 + y2m - y2p = 75;
```

```r
library(lpSolveAPI)                            # import library
LP_Question2 <- read.lp('Question_2.lp')       # read lp file
LP_Question2                                   # show lp model
```

```
## 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(LP_Question2)                             # solution
```

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

```
get.objective(LP_Question2)                     # get objective value
```

```
## [1] 225
```

```
get.variables(LP_Question2)                     # get decision variables
```

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

From the lp file results, we can see the objective function value is 225.

I create a data-frame to show the decision variables results clearly.

```
# create columns
Decision_Variables <- c('x1', 'x2', 'x3', 'y1p', 'y1m', 'y2m', 'y2p')
Values <- c(0, 0, 15, 25, 0, 0, 0)

# create data-frame
Decision_Variables_Results <- data.frame(Decision_Variables, Values)

# show data-frame
Decision_Variables_Results
```

```
##   Decision_Variables Values
## 1                 x1      0
## 2                 x2      0
## 3                 x3     15
## 4                y1p     25
## 5                y1m      0
## 6                y2m      0
## 7                y2p      0
```

The decision results show that the y1p is 25 which means the employment level should increase 25: from 50 to 75.

The solution shows it has penalty for employment level, I try to use priority streamline method to improve the penalty.

The improve model I increase the coefficient for y1p, y1m and y2m.

Improve model

lp improve modle file

```
// Objective function
max: 20 x1 + 15 x2 + 25 x3 - 12 y1p - 12 y1m - 6 y2m;

// Constraints
6 x1 + 4 x2 + 5 x3 + y1m - y1p = 50;
8 x1 + 7 x2 + 5 x3 + y2m - y2p = 75;
```

```r
LP_Question2_improve <- read.lp('Question_2_improve.lp')   # read file
LP_Question2_improve                                        # show model
```

```
## Model name:
##              x1    x2    x3   y1p   y1m   y2m   y2p
## Maximize     20    15    25   -12   -12    -6     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
```

```r
solve(LP_Question2_improve)                                 # solution
```

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

```r
get.objective(LP_Question2_improve)                         # get objective values
```

```
## [1] 208.3333
```

```r
get.variables(LP_Question2_improve)                         # get decision variables values
```

```
## [1] 0.000000 8.333333 3.333333 0.000000 0.000000 0.000000 0.000000
```

We can see the improve model solve the penalty problem since for y1p, y1m and y2m are all 0.

But the objective values has decreased from 225 to 208.3.

If the company focus on remove all the penalties, they should choose the improve model although the objective value is not that large.

If the company focus on maximize the z value which is the objective function value, they should choose the original model although it has penalty on employment level.