Assignment_DEA

Tirumal Achina

2023-11-03

Summary:

In summary, the data centers have varying levels of efficiency. To enhance energy consumption and performance, it is critical to concentrate on individuals with lower efficiency ratings, particularly those in the bottom quartile. Implementing innovative and open energy regulations and corrective measures in these data centers can result in considerable cost savings and improved sustainability. The emphasis should be on maintaining current efficiency levels for data centers with better efficiency ratings.

The values of u_efficiency vary from 0.4754 to 1.0000. The mean value is 0.8535, indicating that the procedure is efficient in general. However, there is substantial variety in the u_efficiency values, indicating that there is still opportunity for improvement in some areas.Coming to the median value(0.9957) which is slightly higher than the mean which shows that more than half of the values are above that value.

Analyzing the outcomes:-

1.Efficiency Scores:

A number of 1 shows perfect efficiency, whereas a score of 0 indicates utter inefficiency.

- Maximum Efficiency Data Centers: - Data Centers 1, 2, 5, 7, 10, 13 and 15 perform extraordinarily well, with efficiency ratings of 1. They are efficiently optimizing their resources and energy use.

2.High Performance Data Centers:

- Data Centers 3, 14, and 16 have efficiency scores close to one (about 0.99), indicating that they, too, are running effectively. There is, however, still potential for development.

3.Data Centers with Room for Growth:

- Data Centers 8, 9, and 11 have lower efficiency scores, indicating operational inefficiencies. Implementing energy-saving measures and process improvements may help them.

4.Significantly Inefficient Data Centers:

- The lowest efficiency scores (approximately 0.47) are seen in Data Centers 4, 6, 12, 17 and 18, suggesting considerable inefficiencies. To minimize energy consumption and increase performance, they should prioritize energy-saving technology and operational enhancements.

5.Peers :

- This gives the relation between the Decision Making Units(DMU) that has similar output and input characteristics and used in terms of efficiency comparisions. These are used to measure performance of specific DMU and provides insights to improve its efficiency.
- Example: here 11th DMU has relation with 2,13,15 DMU’s.

6.Lambda :

- When calculating a DMU's efficiency score, the "lambda" function in DEA is utilized to determine the weights or multipliers that are applied to each of its inputs and outputs. The relative significance or contribution of each input and output to the overall efficiency of the DMU is represented by these weights.
- Example: If the lamda value is 1, then that attribute has more impact in determining the Decision Making Units efficiency score and if the value is less than 1 also conttibutes to the score but not an only factor.

Conclusion:

- The majority of the numbers are close to one, indicating good efficiency. However, the minimum and the occurrence of values below the median and mean imply that there are a few values that are much lower than 1. Use DEA to discover inefficiencies and implement remedial measures. Make energy-saving measures a top priority to make your data centers more sustainable and cost-effective.
- Identifying and eliminating bottlenecks is one technique to increase process efficiency. Investing in new technology or equipment is another strategy to increase efficiency.

#Installing and loading the "Benchmarking" Library to perform the DEA Analysis.

```
library("Benchmarking")
```

```
## Warning: package 'Benchmarking' was built under R version 4.3.2
```

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

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

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

#Loading the sample dataset "energy".

```
tiru <- read.csv("energy.csv")
str(tiru)
```

```
## 'data.frame':    18 obs. of  9 variables:
##  $ X                 : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Energy.Policy     : chr  "Always" "Margin" "Gamma" "Always" ...
##  $ Scheduling.Model  : chr  "Monolithic" "Mesos" "Omega" "Mono." ...
##  $ Work.Load         : chr  "High" "High" "High" "Low" ...
##  $ D.C..Size         : int  1000 1000 1000 1000 1000 1000 5000 5000 5000 5000 ...
##  $ Shut.Downs        : int  37166 13361 14252 36404 19671 32407 6981 9877 33589 8578 ...
##  $ Computing.Time..h.: num  104.4 104.3 104.2 49.2 49.6 ...
##  $ MWh.Consumed      : num  49 49.6 49.6 23.9 24.6 ...
##  $ Queue.Time..ms.   : num  90.1 1093 0.1 78.3 1188.7 ...
```

#Here we are feeding the inputs and outputs from the information that we had.

```
inputs <- tiru[,c("D.C..Size","Shut.Downs")]
outputs <- tiru[,c("Computing.Time..h.","MWh.Consumed","Queue.Time..ms.")]


#we make a model and feed input and output to the model
melody <- dea(inputs, outputs, RTS =   "crs")
u_efficiency <- eff(melody)
peers(melody)
```

```
##       peer1 peer2 peer3
##  [1,]     1    NA    NA
##  [2,]     2    NA    NA
##  [3,]     1     2    NA
##  [4,]     2    NA    NA
##  [5,]     5    NA    NA
##  [6,]     2    NA    NA
##  [7,]     7    NA    NA
##  [8,]     2    10    13
##  [9,]     2    15    NA
## [10,]    10    NA    NA
## [11,]     2    13    15
## [12,]     2    15    NA
## [13,]    13    NA    NA
## [14,]     2    13    15
## [15,]    15    NA    NA
## [16,]     2    15    NA
## [17,]     2    13    NA
## [18,]     2    15    NA
```

```
lambda(melody)
```

```
##                L1         L2 L5 L7       L10       L13        L15
##  [1,] 1.000000000 0.00000000  0  0 0.0000000 0.0000000 0.00000000
##  [2,] 0.000000000 1.00000000  0  0 0.0000000 0.0000000 0.00000000
##  [3,] 0.009970484 0.98915099  0  0 0.0000000 0.0000000 0.00000000
##  [4,] 0.000000000 0.48177241  0  0 0.0000000 0.0000000 0.00000000
##  [5,] 0.000000000 0.00000000  1  0 0.0000000 0.0000000 0.00000000
##  [6,] 0.000000000 0.48721047  0  0 0.0000000 0.0000000 0.00000000
##  [7,] 0.000000000 0.00000000  0  1 0.0000000 0.0000000 0.00000000
##  [8,] 0.000000000 0.22098286  0  0 0.5914729 0.1734861 0.00000000
##  [9,] 0.000000000 2.03346741  0  0 0.0000000 0.0000000 0.27553094
## [10,] 0.000000000 0.00000000  0  0 1.0000000 0.0000000 0.00000000
## [11,] 0.000000000 0.53626578  0  0 0.0000000 0.4082527 0.02840485
## [12,] 0.000000000 0.26256674  0  0 0.0000000 0.0000000 0.21144095
## [13,] 0.000000000 0.00000000  0  0 0.0000000 1.0000000 0.00000000
## [14,] 0.000000000 0.04516562  0  0 0.0000000 0.8554257 0.13443418
## [15,] 0.000000000 0.00000000  0  0 0.0000000 0.0000000 1.00000000
## [16,] 0.000000000 0.02236541  0  0 0.0000000 0.0000000 0.99479451
## [17,] 0.000000000 0.89985422  0  0 0.0000000 0.3814863 0.00000000
## [18,] 0.000000000 0.93720988  0  0 0.0000000 0.0000000 0.38461980
```

```
#getting the efficiency values.
u_efficiency
```

```
##  [1] 1.0000000 1.0000000 0.9991215 0.4817724 1.0000000 0.4872105 1.0000000
##  [8] 0.9826417 0.9577554 1.0000000 0.9805683 0.4753953 1.0000000 0.9943765
## [15] 1.0000000 0.9970310 0.5290248 0.4783408
```

```
summary(u_efficiency)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.4754  0.6362  0.9957  0.8535  1.0000  1.0000
```