QMM ASSIGNMENT- DEA ANALYSIS

ASSIGNMENT MODULE – 8

DIVYA CHANDRASEKARAN_811284790

2023-10-26

Assignment Instructions: Module 8 - DEA

Purpose
The purpose of this assignment is to explore the use of DEA. Students will formulate and solve DEA problems under different assumptions. They will then compare and contrast these results. In addition, this will help you master the following module outcomes:

- Formulate models for DEA.
- Solve models for DEA.
- Interpret DEA results.
- Apply DEA results.

Directions
Hope Valley Health Care Association

The Hope Valley Health Care Association owns and operates six nursing homes in adjoining states. An evaluation of their efficiency has been undertaken using two inputs and two outputs. The inputs are staffing labor (measured in average hours per day) and the cost of supplies (in thousands of dollars per day). The outputs are the number of patient-days reimbursed by third-party sources and the number of patient-days reimbursed privately. A summary of performance data is shown in the table below:

DMU	Staff hours per day	Supplies per day	Reimbursed patient-days	Privately paid patient-days
Facility 1	100	0.3	15000	3500
Facility 2	300	0.6	15000	20000
Facility 3	320	1.2	40000	11000
Facility 4	500	2	28000	42000
Facility 5	350	1.4	20000	25000
Facility 6	340	0.7	14000	15000

Questions
1. Formulate and perform DEA analysis under all DEA assumptions of FDH, CRS, VRS, IRS, DRS, and FRH.
2. Determine the Peers and Lambdas under each of the above assumptions
3. Summarize your results in a tabular format
4. Compare and contrast the above results

Objective:

The objective is to evaluate the relative efficiency of 6 nursing home facilities (DMUs) operated by the Hope Valley Health Care Association. Efficiency will be evaluated using Data Envelopment Analysis (DEA) under different assumptions.

Problem Statement:

The Association wants to determine how efficiently each facility is utilizing its inputs (staff hours and supply costs) to produce outputs (reimbursed and private patient days).

DEA will be used to calculate an efficiency score for each DMU. The score represents how much each DMU's inputs can be proportionally reduced without reducing outputs. A score of 1 represents an efficient DMU.

DEA makes different assumptions about returns to scale. Different models will be tested:

- FDH: Free Disposal Hull (no assumption on returns to scale)

- CRS: Constant Returns to Scale

- VRS: Variable Returns to Scale

- IRS: Increasing Returns to Scale

- DRS: Decreasing Returns to Scale

- FRH: Free Replicability Hull

The results under each model will be analyzed to determine:

- Efficiency scores for each DMU

- Peer DMUs and lambdas (peer weights) for each DMU

- How scores and peers change across models

- Best and worst performing DMUs

This will help the Association evaluate performance, identify benchmarks, and make decisions about resource allocation across the 6 facilities. The choice of optimal DEA model will depend on assumptions about returns to scale in this industry.

The key differences between the models are:

- FDH makes the least assumptions, so scores are most lenient.

- CRS assumes constant returns to scale, VRS allows variable returns.

- IRS and DRS test for increasing or decreasing returns specifically.

- FRH tests for free replicability of best practice.

So, the choice of model depends on the assumptions we want to make about the production process. VRS is commonly used as the default.

In **summary**, the key objective is to leverage DEA to evaluate nursing home efficiency under different assumptions and provide actionable insights to the Association.

```r
#LOAD THE NECESSARY LIBRARIES
library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✓ dplyr     1.1.3     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.3     ✓ tibble    3.2.1
## ✓ lubridate 1.9.2     ✓ tidyr     1.3.0
## ✓ purrr     1.0.2
## ── Conflicts ──────────────────────────────────────────
tidyverse_conflicts() ──
## ✕ dplyr::filter() masks stats::filter()
## ✕ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all
conflicts to become errors

library(Benchmarking)

## Loading required package: lpSolveAPI
## Loading required package: ucminf
## Loading required package: quadprog
```

HERE, WE ARE GOING TO COMPUTE THE FORMULATION

```r
#CREATING THE VECTORS WITH THE VALUES GIVEN
input <- matrix(c(100, 300, 320, 500, 350, 340, 300, 600, 1200, 2000, 1400,
700),ncol = 2)
output <- matrix(c(15000, 15000, 40000, 28000, 20000, 14000, 3500, 20000,
11000, 42000, 25000, 15000),ncol = 2)

#ASSIGN THE COLUMN NAMES
colnames(output) <- c("staff_hours_perday","supplies_perday")
colnames(input) <- c("reimbursed_patient_days", "privately_paid_patient-
days")

# To see the values of Input
input

##      reimbursed_patient_days privately_paid_patient-days
## [1,]                     100                         300
## [2,]                     300                         600
## [3,]                     320                        1200
## [4,]                     500                        2000
## [5,]                     350                         1400
## [6,]                     340                          700

# To see the values of Output
output

##      staff_hours_perday supplies_perday
## [1,]              15000            3500
## [2,]              15000           20000
## [3,]              40000           11000
## [4,]              28000           42000
## [5,]              20000           25000
## [6,]              14000           15000
```

As evident, the values we have obtained align with the performance data table for the six nursing homes under the ownership of Hope Valley Health Care Association. In the upcoming section, we will conduct a Data Envelopment Analysis (DEA). DEA is a valuable analytical technique that aids organizations in pinpointing and optimizing their resource allocation to improve efficiency and adopt superior practices.

DEA ANALYSIS USING FDH

Next, we will construct and execute the DEA analysis using the Free Disposability Hull (FDH) approach. In FDH, the underlying assumption is the ability to eliminate any surplus inputs or outputs, essentially allowing us to produce fewer outputs with additional inputs as needed. This concept is often referred to as "free disposability," as it enables flexibility in adjusting input-output combinations.

```
#DEA ANALYSIS USING FREE DISPOSAL HULL

# Provide the input and output
analysis_fdh<- dea(input, output, RTS = "fdh")

# Create a data frame with efficiency values
eff_fdh <- as.data.frame(analysis_fdh$eff)

# Assigning an appropriate name
```

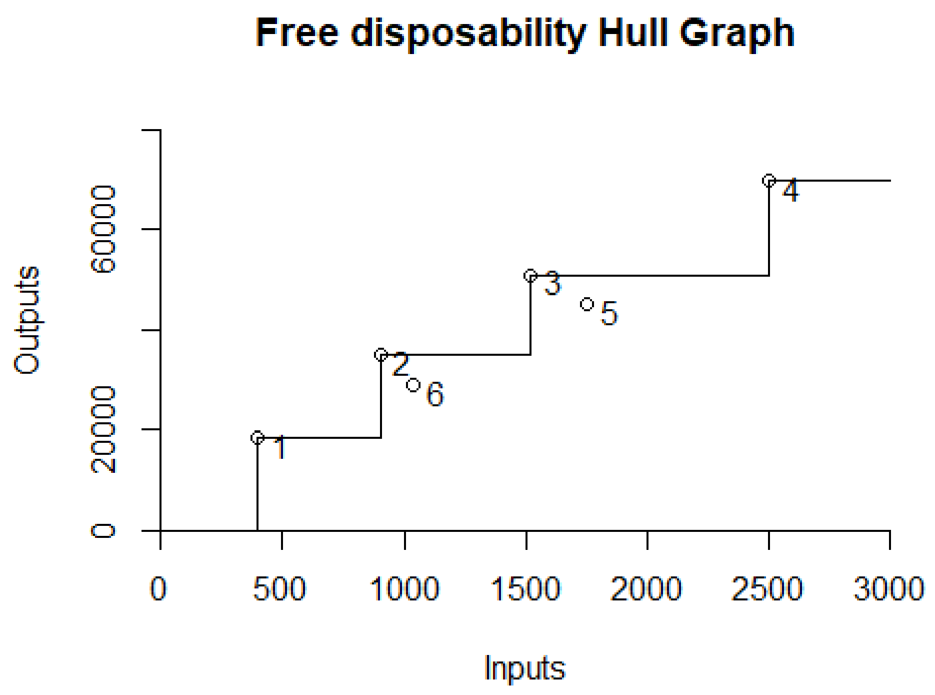
```r
colnames(eff_fdh) <- c("efficiency_fdh")

# Determining the peers
peer_fdh <- peers(analysis_fdh)

# Assigning a name for peer
colnames(peer_fdh) <- c("peer1_fdh")

# Identifying the relative weights given to the peers using lambda function
lambda_fdh <- lambda(analysis_fdh)

# Assigning an appropriate column name for Lambda function
names(lambda_fdh) <- c("Lambda1_fdh", "Lambda2_fdh", "Lambda3_fdh",
"Lambda4_fdh", "Lambda5_fdh", "Lambda6_fdh")

# Creating a tabular data with peer, lambda, and efficiency
peer_lamb_eff_fdh <- cbind(peer_fdh, lambda_fdh, eff_fdh)

# Presenting the summary chart
peer_lamb_eff_fdh
```

```
##   peer1_fdh L1 L2 L3 L4 L5 efficiency_fdh
## 1         1  1  0  0  0  0      1.0000000
## 2         2  0  1  0  0  0      1.0000000
## 3         3  0  0  1  0  0      1.0000000
## 4         4  0  0  0  1  0      1.0000000
## 5         5  0  0  0  0  1      1.0000000
## 6         2  0  1  0  0  0      0.8823529
```

As we've comprehended from this module, peers play a crucial role in identifying inefficient Decision Making Units (DMUs) or units. The Lambda values represent the initial weights assigned from the peer units when solving the DEA model.

The summary chart presented above reaffirms that each DMU or facility is fully utilizing its capacity and operating efficiently. In this scenario, every peer unit has been allocated a weight of one, resulting in Lambda values of 1, which consequently leads to an efficiency score of 1 for all units, except the last one having an efficiency of 88.23%. This means that this particular facility is not making the most efficient use of its inputs to generate outputs when compared to its peers in the dataset. The lower efficiency score could be due to various reasons, such as suboptimal resource allocation, underutilization of inputs, or other operational inefficiencies. It suggests that there is room for improvement in the performance of the facility analyzed using FDH. Further investigation and adjustments in resource allocation or operational processes may be necessary to bring its efficiency in line with that of other facilities.

```r
# Plotting the results for DEA ANALYSIS USING FDH
dea.plot(input, output, RTS = "fdh", ORIENTATION = "in-out", txt = TRUE, xlab
= 'Inputs', ylab = 'Outputs', main = "Free disposability Hull Graph")
```



DEA ANAYLSIS USING CRS

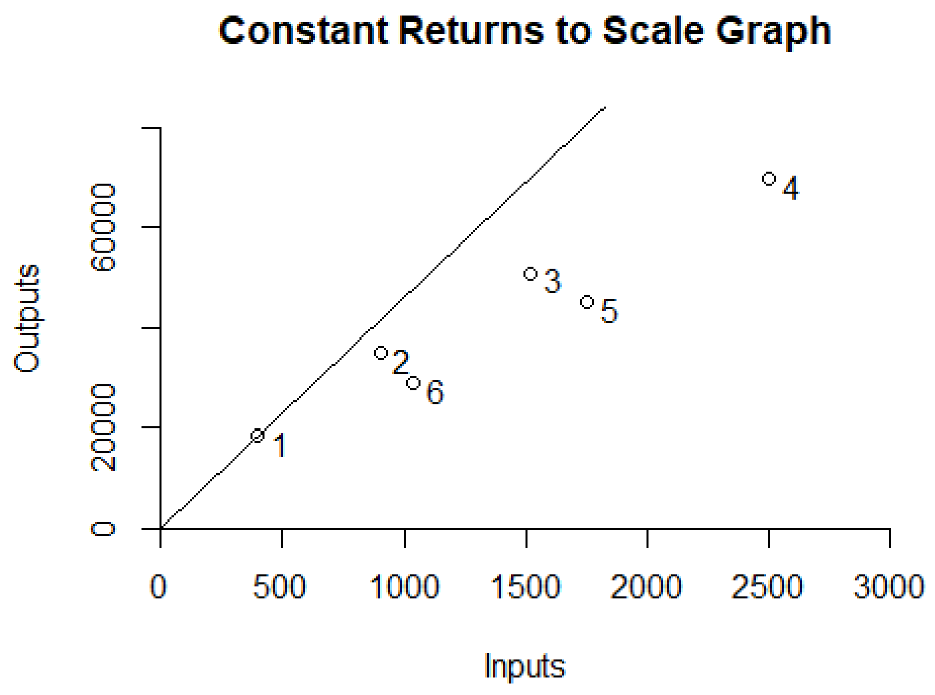
Our next step involves formulating and conducting the DEA analysis using the Constant Returns to Scale (CRS) approach. CRS is an integral part of the scaling assumption, enabling us to assess whether there exist feasible ways to either expand or reduce the scale of operations.

```r
#DEA ANALYSIS USING CONSTANT RETURN TO SCALE

# Provide the input and output
analysis_crs <- dea(input,output,RTS = "crs")
```

```r
# To see the efficiency values
eff_crs <- as.data.frame(analysis_crs$eff)

# Assigning an appropriate name
colnames(eff_crs) <- c("efficiency_crs")

# Determining the peers
peer_crs <- peers(analysis_crs)
str(peer_crs)

##  int [1:6, 1:2] 1 2 1 4 1 1 NA NA 4 NA ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr [1:2] "peer1" "peer2"

# Assigning an appropriate name for peer
names(peer_crs) <- c("peer1_crs", "peer2_crs", "peer3_crs")

# Identifying the relative weights given to the peers using lambda function
lambda_crs <- lambda(analysis_crs)

# Assigning a column name for Lambda function
names(lambda_crs) <- c("Lambda1_crs", "Lambda2_crs", "Lambda3_crs",
"Lambda4_crs")

# Creating a tabular data with peer, lambda, and efficiency
peer_lamb_eff_crs <- cbind(peer_crs, lambda_crs, eff_crs)

# Presenting the summary chart
peer_lamb_eff_crs

##   peer1 peer2        L1        L2         L4 efficiency_crs
## 1     1    NA 1.0000000 0.0000000 0.00000000      1.0000000
## 2     2    NA 0.0000000 1.0000000 0.00000000      1.0000000
## 3     1     4 2.5789474 0.0000000 0.04699248      0.8793468
## 4     4    NA 0.0000000 0.0000000 1.00000000      1.0000000
## 5     1     4 0.2631579 0.0000000 0.57330827      0.8941998
## 6     1     2 0.2222222 0.7111111 0.00000000      0.7047619
```

In the context of Constant Returns to Scale (CRS), facilities 1, 2, and 4 are operating at full efficiency, as corroborated by the Lambda values and peer assessments. Conversely, facilities 3, 5, and 6 require input from facilities 1, 2, and 4, as evident from the peer evaluations and Lambda values displayed earlier. This indicates that these three facilities (3, 5, and 6) have the potential for enhancement since they are currently operating at efficiencies of 87.93%, 89.41%, and 70.47%, leaving room for improvement.

```r
# Plotting the results for DEA ANALYSIS USING CRS
dea.plot(input,output,RTS="crs",ORIENTATION="in-out",
txt=TRUE, xlab = 'Inputs', ylab= 'Outputs', main= "Constant Returns to Scale
Graph")
```



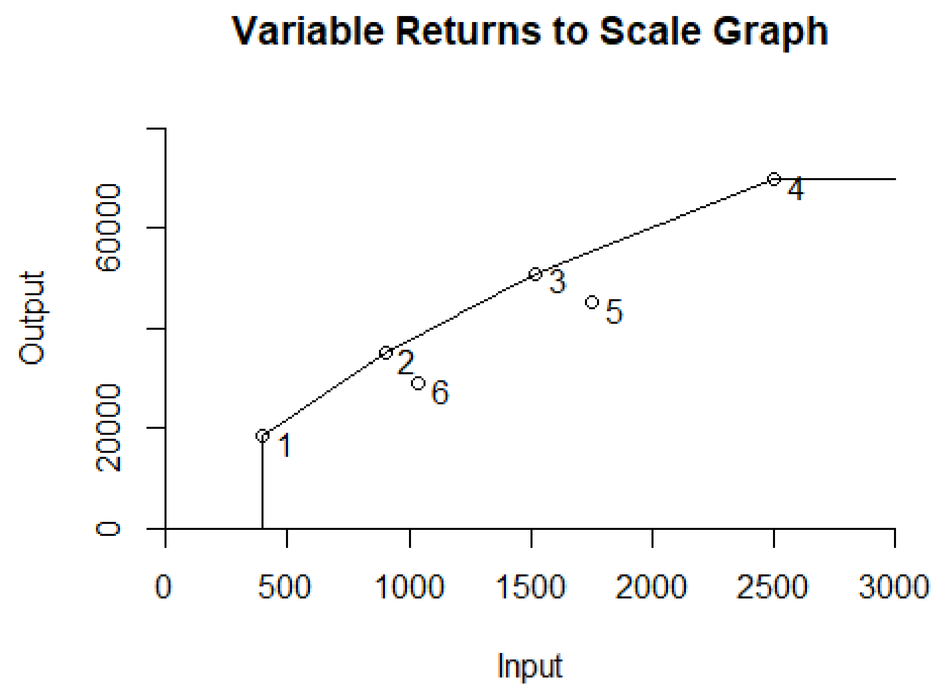
DEA ANALYSIS USING VRS

Our next step involves crafting and executing the DEA analysis using Variable Returns to Scale (VRS). VRS, like CRS, is an integral part of the scaling assumption, but it serves to assess the efficiency of variables when changes in scale are not necessarily proportional, whether it's an increase or decrease in operations.

```r
#DEA ANALYSIS USING VARIABLE RETURN TO SCALE

# Provide the input and output
analysis_vrs <- dea(input,output,RTS = "vrs")

# To see the efficiency values
eff_vrs <- as.data.frame(analysis_vrs$eff)
```

```r
# Assigning an appropriate name
colnames(eff_vrs) <- c("efficiency_vrs")

# Determining the peers
peer_vrs <- peers(analysis_vrs)

# Assigning an appropriate name for peer
names(peer_vrs) <- c("peer1_vrs", "peer2_vrs", "peer3_vrs")

# Identifying the relative weights given to the peers using lambda function
lambda_vrs <- lambda(analysis_vrs)

# To assign a column name for Lambda function
names(lambda_vrs) <- c("Lamda1_vrs", "Lambda2_vrs", "Lambda3_vrs",
"Lambda4_vrs", "Lambda5_vrs")

# Creating a tabular data with peer, lambda, and efficiency
peer_lamb_eff_vrs <- cbind(peer_vrs, lambda_vrs, eff_vrs)

# Presenting the summary chart
peer_lamb_eff_vrs

##   peer1 peer2        L1        L2 L3        L4 efficiency_vrs
## 1     1    NA 1.0000000 0.0000000  0 0.0000000      1.0000000
## 2     2    NA 0.0000000 1.0000000  0 0.0000000      1.0000000
## 3     3    NA 0.0000000 0.0000000  1 0.0000000      1.0000000
## 4     4    NA 0.0000000 0.0000000  0 1.0000000      1.0000000
## 5     1     4 0.4415584 0.0000000  0 0.5584416      0.9239332
## 6     1     2 0.3030303 0.6969697  0 0.0000000      0.7272727

# Plotting the results for DEA ANALYSIS USING VRS
dea.plot(input,output,RTS="vrs",ORIENTATION="in-out",
txt=TRUE, xlab = 'Input', ylab= 'Output', main="Variable Returns to Scale
Graph")
```

When employing the Variable Returns to Scale (VRS) analysis, we can determine that facilities 1, 2, 3, 4, and 5 are effectively utilizing their capacity and operating at peak efficiency. In contrast, facility 6 and facility 5 are not performing at their full potential, with an efficiency rating of 72.72% and 92.39%.



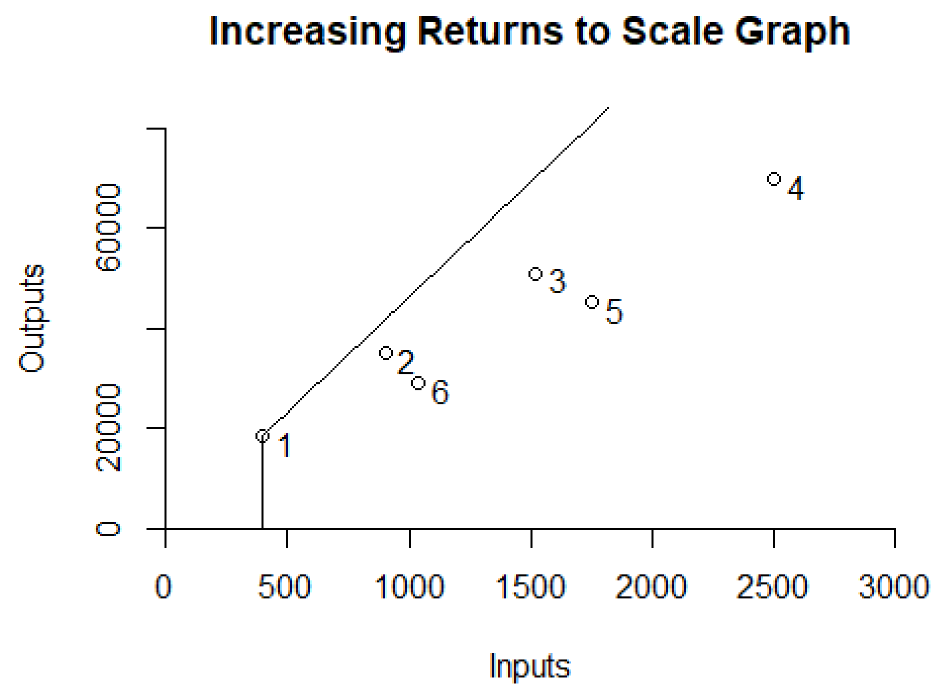
DEA ANALYSIS USING IRS

```r
#DEA USIING INCREASING RETURN TO SCALE

# Provide the input and output
analysis_irs <- dea(input,output,RTS = "irs")

# To see the efficiency values
eff_irs <- as.data.frame(analysis_irs$eff)

# Assigning a name
colnames(eff_irs) <- c("efficiency_irs")

# Determining the peers
peer_irs <- peers(analysis_irs)

# Assigning an appropriate name for peer
names(peer_irs) <- c("peer1_irs", "peer2_irs", "peer3_irs")

# Identifying the relative weights given to the peers using lambda function
lambda_irs <- lambda(analysis_irs)
```

```
# Assigning a column name for Lambda function
names(lambda_irs) <- c("Lambda1_irs", "Lambda2_irs", "Lambda3_irs",
"Lambda4_irs", "Lambda5_irs")

# Creating a tabular data with peer, lambda, and efficiency
peer_lamb_eff_irs <- cbind(peer_irs, lambda_irs, eff_irs)

# Presenting the summary chart
peer_lamb_eff_irs

##   peer1 peer2        L1        L2         L4 efficiency_irs
## 1     1    NA 1.0000000 0.0000000 0.00000000      1.0000000
## 2     2    NA 0.0000000 1.0000000 0.00000000      1.0000000
## 3     1     4 2.5789474 0.0000000 0.04699248      0.8793468
## 4     4    NA 0.0000000 0.0000000 1.00000000      1.0000000
## 5     1     4 0.4415584 0.0000000 0.55844156      0.9239332
## 6     1     2 0.3030303 0.6969697 0.00000000      0.7272727
```

Increasing Returns to Scale (IRS) exhibits a similar pattern to Variable Returns to Scale (VRS), wherein facilities 1, 2, and 4 are functioning at peak efficiency. However, facilities 3,5, and 6 lag behind with an efficiency of 87.93%, 92.39%, and 72.72% and require enhancements through support from units 1, 2, and 4 to reach optimal efficiency.

```
# Plotting the results for DEA ANALYSIS USING IRS
dea.plot(input,output,RTS="irs",ORIENTATION="in-out",
txt=TRUE, xlab = 'Inputs', ylab= 'Outputs', main= "Increasing Returns to
Scale Graph")
```



DEA ANALYSIS USING DRS

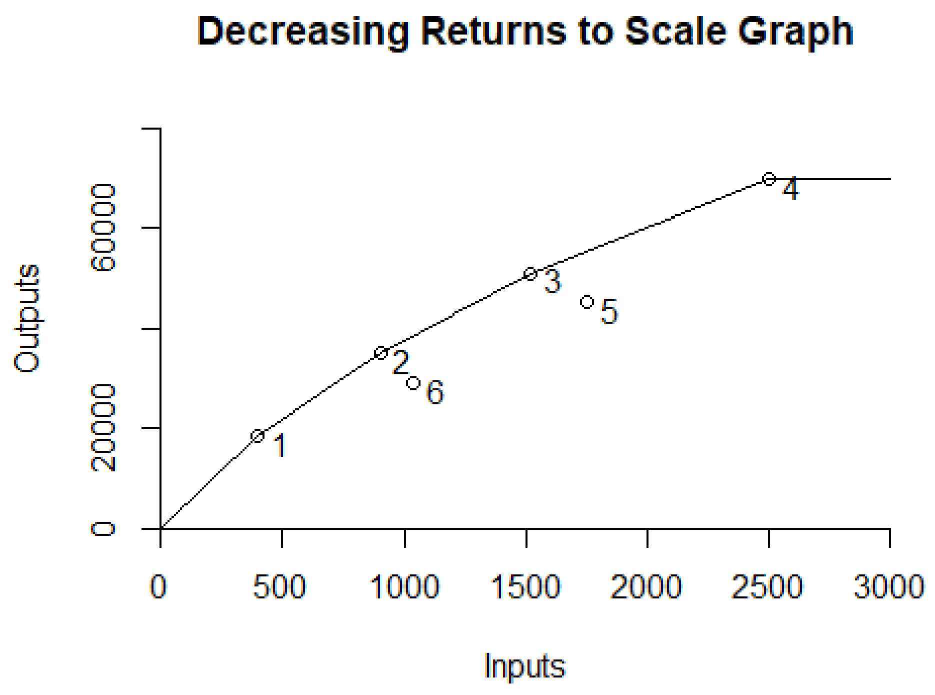
DRS (Decreasing Returns to Scale) stands in contrast to IRS (Increasing Returns to Scale) as it aims to reduce the scale of operation in any feasible production process.

```
#DEA ANALYSIS USING DECREASING RETURNS TO SCALE

# Provide the input and output
analysis_drs <- dea(input,output,RTS = "drs")

# To see the efficiency values
eff_drs <- as.data.frame(analysis_drs$eff)

# Assigning an appropriate name
colnames(eff_drs) <- c("efficiency_drs")

# Determining the peers
peer_drs <- peers(analysis_drs)
```

```
#Assigning an appropriate name for peer
names(peer_drs) <- c("peer1_drs", "peer2_drs", "peer3_drs")

# Identifying the relative weights given to the peers using lambda function
lambda_drs <- lambda(analysis_drs)

#Assigning a column name for Lambda
names(lambda_drs) <- c("Lambda1_drs", "Lambda2_drs", "Lambda3_drs",
"Lambda4_drs")

# Creating a tabular data with peer, lambda, and efficiency
peer_lamb_eff_drs <- cbind(peer_drs, lambda_drs, eff_drs)

# Presenting the summary chart
peer_lamb_eff_drs

##   peer1 peer2        L1        L2 L3        L4 efficiency_drs
## 1     1    NA 1.0000000 0.0000000  0 0.0000000      1.0000000
## 2     2    NA 0.0000000 1.0000000  0 0.0000000      1.0000000
## 3     3    NA 0.0000000 0.0000000  1 0.0000000      1.0000000
## 4     4    NA 0.0000000 0.0000000  0 1.0000000      1.0000000
## 5     1     4 0.2631579 0.0000000  0 0.5733083      0.8941998
## 6     1     2 0.2222222 0.7111111  0 0.0000000      0.7047619
```

In the context of Decreasing Returns to Scale (DRS), facilities 1, 2, 3, and 4 demonstrate high efficiency. However, when it comes to facilities 5 and 6, there exists potential for improvement. These two facilities can enhance their efficiency by tapping into the resources of facilities 1, 2, 3, and 4, as demonstrated in the earlier table, enabling them to reach a maximum efficiency score of 1.

```
# Plotting the results for DEA ANALYSIS USING DRS
dea.plot(input,output,RTS="drs",ORIENTATION="in-out",
txt=TRUE, xlab = 'Inputs', ylab= 'Outputs', main= "Decreasing Returns to
Scale Graph")
```



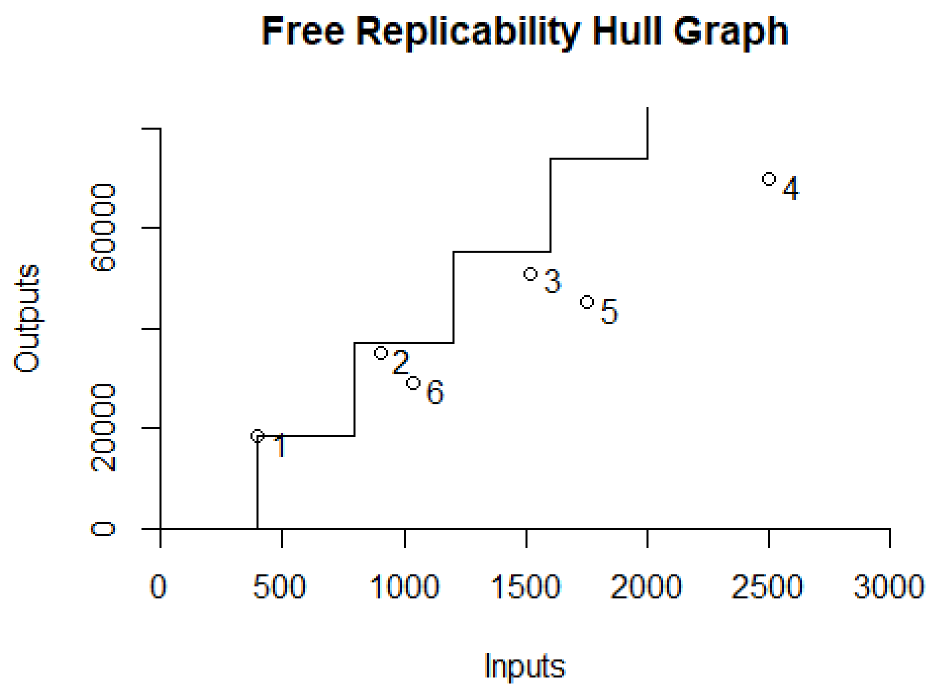
DEA ANALYSIS USING FRH

```
#DEA ANALYSIS USING FREE REPLICABLILTY HULL

# Provide the input and output
analysis_frh <- dea(input,output,RTS = "add")

# To see the efficiency values
eff_frh <- as.data.frame(analysis_frh$eff)

# To assign an appropriate name
colnames(eff_frh) <- c("efficiency_frh")

# Determining the peers
peer_frh <- peers(analysis_frh)

# Assigning a name for the peer
names(peer_frh) <- c("peer1_frh")

# Identifying the relative weights given to the peers using lambda function
lambda_frh <- lambda(analysis_frh)
```

```r
# Assigning a column name for Lambda function
names(lambda_frh) <- c ("Lambda1_frh", "Lambda2_frh", "Lambda3_frh",
"Lambda4_frh", "Lambda5_frh", "Lambda6_frh")

# Creating a tabular data with peer, lambda, and efficiency
peer_lamb_eff_frh <- cbind(peer_frh, lambda_frh, eff_frh)

# Presenting the summary chart
peer_lamb_eff_frh

##   peer1 L1 L2 L3 L4 L5 efficiency_frh
## 1     1  1  0  0  0  0      1.0000000
## 2     2  0  1  0  0  0      1.0000000
## 3     3  0  0  1  0  0      1.0000000
## 4     4  0  0  0  1  0      1.0000000
## 5     5  0  0  0  0  1      1.0000000
## 6     2  0  1  0  0  0      0.8823529
```

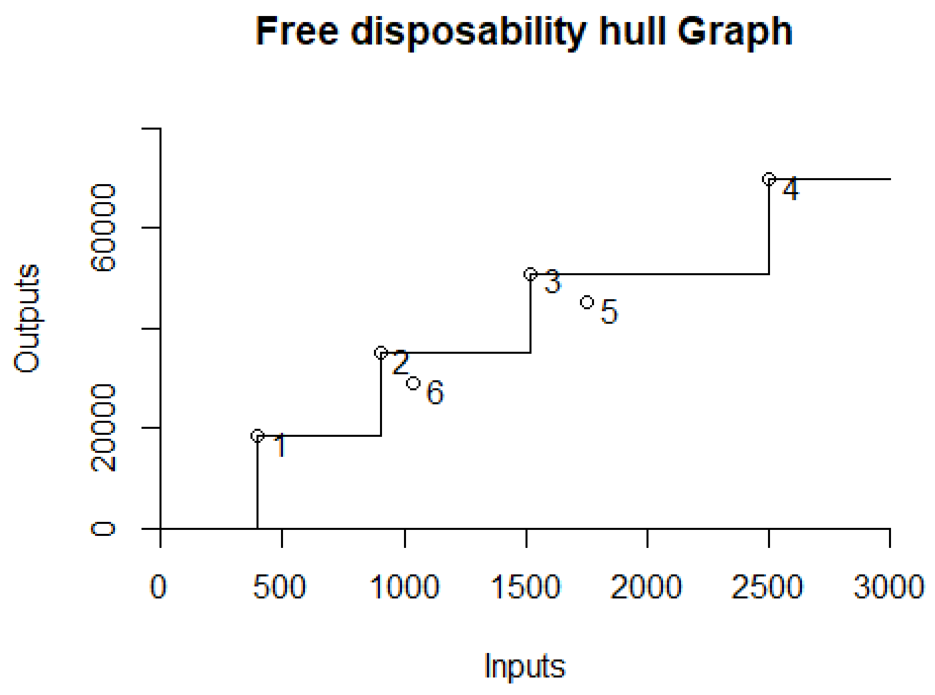
Free Replicability Hull (FRH) demonstrates exceptional efficiency across all its Decision-Making Units (DMUs). Its performance aligns with that of Free Disposability Hull (FDH), where all values have corresponding peers, Lambdas, and efficiencies of 1 except facility 6 which has an efficiency of 88.23%.

```r
# Plotting the results for DEA ANALYSIS USING FRH
dea.plot(input,output,RTS="add",ORIENTATION="in-out",
txt=TRUE, xlab = 'Inputs', ylab= 'Outputs', main = "Free Replicability Hull
Graph")
```

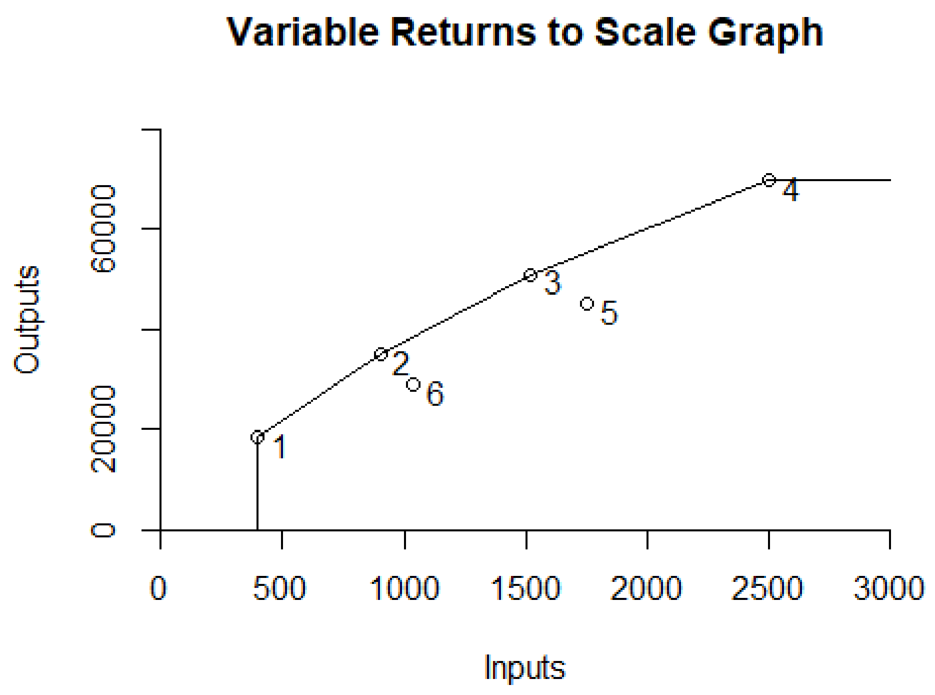


COMPARISON BETWEEN DIFFERENT ASSUMPTIONS

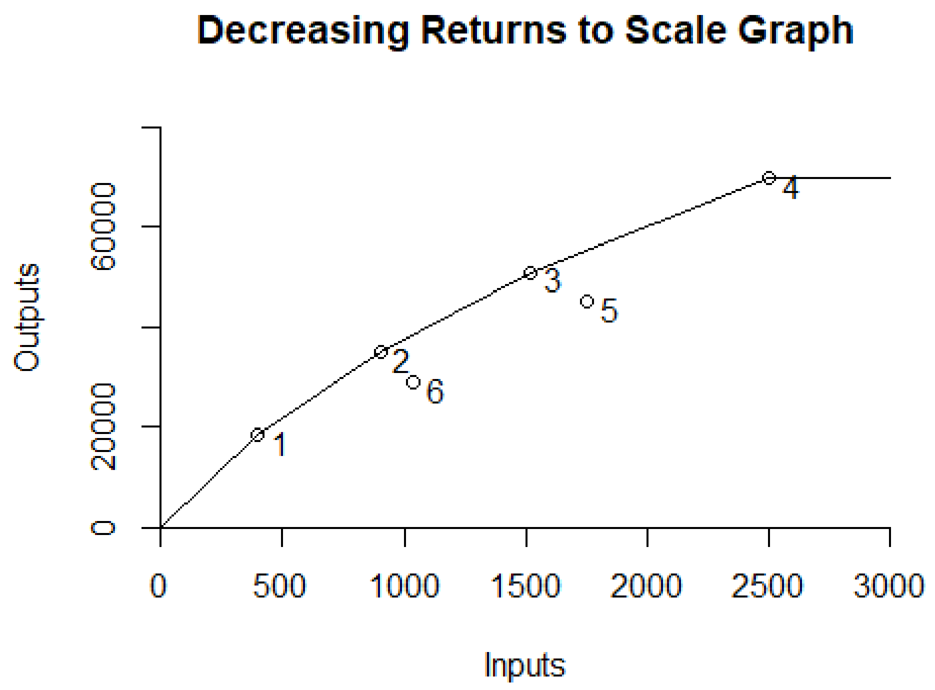
```
#COMPARISON BETWEEN DIFFERENT ASSUMPTIONS FROM OBTAINED RESULTS ABOVE

dea.plot(input,output,RTS="fdh",ORIENTATION="in-out",
txt=TRUE, xlab = 'Inputs', ylab= 'Outputs', main= "Free disposability hull
Graph")
```



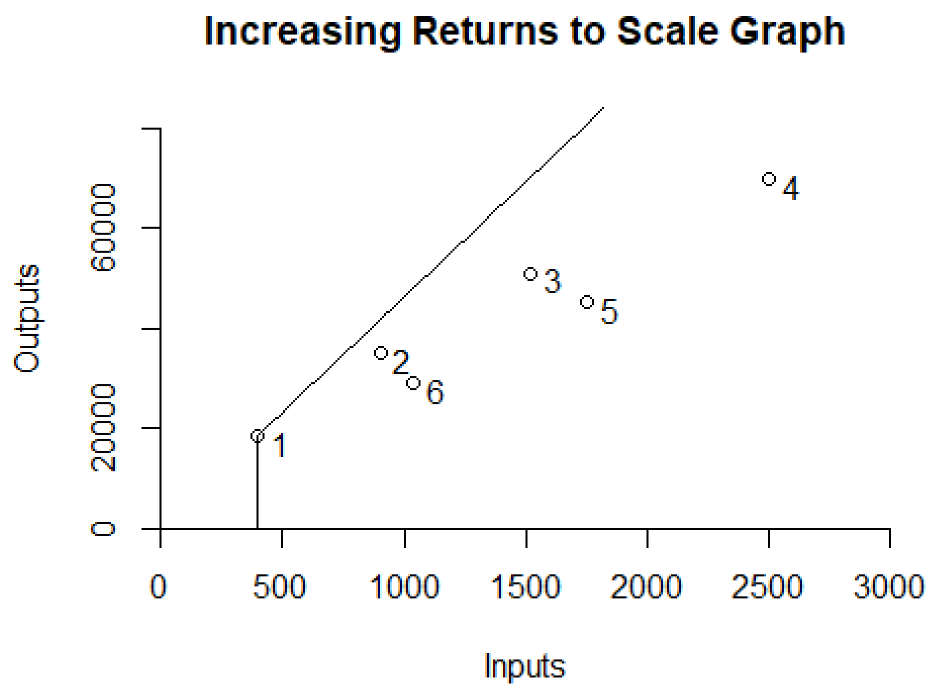
```
dea.plot(input,output,RTS="vrs",ORIENTATION="in-out",
txt=TRUE, xlab = 'Inputs', ylab= 'Outputs', main= "Variable Returns to Scale
Graph")
```



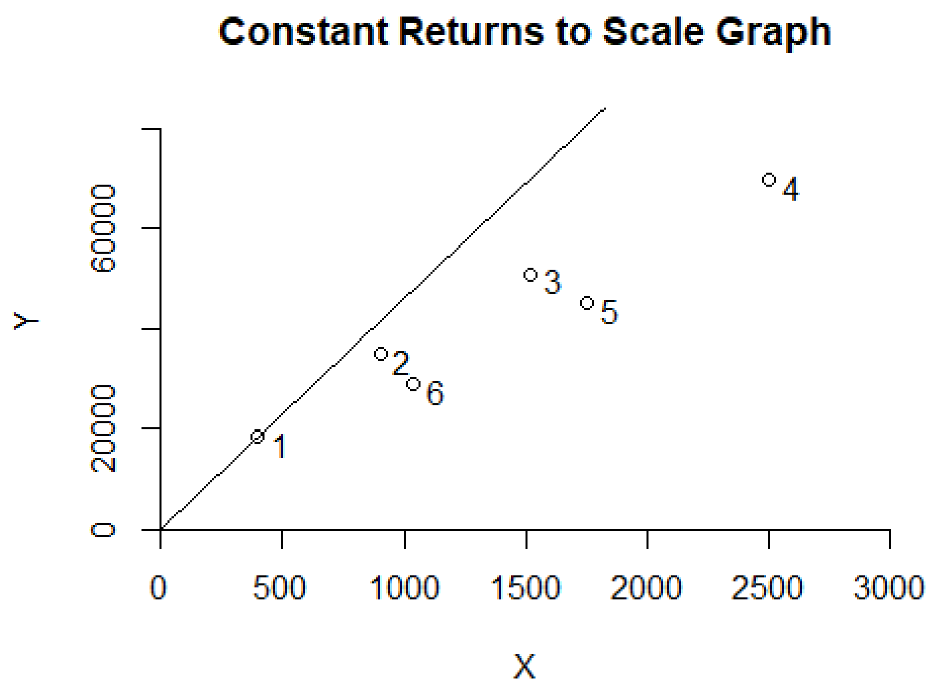
```
dea.plot(input,output,RTS="drs",ORIENTATION="in-out",
txt=TRUE, xlab = 'Inputs', ylab= 'Outputs', main= "Decreasing Returns to
Scale Graph")
```



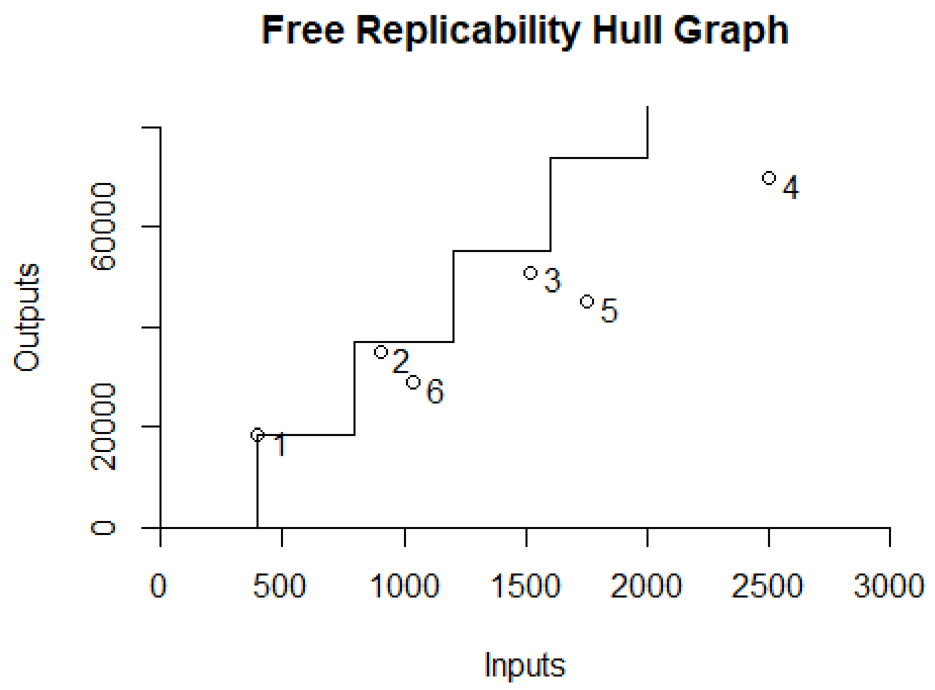
```
dea.plot(input,output,RTS="irs",ORIENTATION="in-out",
txt=TRUE, xlab = 'Inputs', ylab= 'Outputs', main= "Increasing Returns to
Scale Graph")
```



```
dea.plot(input,output,RTS="crs",ORIENTATION="in-out",
txt=TRUE, main="Constant Returns to Scale Graph")
```



```r
dea.plot(input,output,RTS="add",ORIENTATION="in-out",
txt=TRUE, xlab = 'Inputs', ylab= 'Outputs', main= "Free Replicability Hull
Graph")
```



In this module, we've learned that all DEA models follow the principle of estimating technology using a minimal extrapolation approach. When we examine the **FDH model**, it represents the most constrained technology set, striving to achieve minimal outputs (such as the number of patient days reimbursed by third-party sources and the number of patient days reimbursed privately) with a more extensive set of inputs (including staffing labor and the cost of supplies).

FDH is often preferred by firms, but it has limitations due to its underlying assumptions. While all efficiencies in this model are rated at 1, it's not as efficient as it may appear when compared to other models because there are still areas/units with room for improvement.

VRS, on the other hand, is larger than FDH, filling in the spaces that FDH leaves unutilized. In VRS, it's evident that facility 6 can enhance its efficiency. DRS and IRS models expand the technology sets even further. **DRS** aims to increase the set for a smaller input value, indicating that unit 5 and 6 could improve their efficiency. **IRS**, in contrast, seeks to extend the technology for larger input values, highlighting that facility 6 has room for improvement as well.

CRS represents the most extensive technology set, offering insights into possible scaling up or down. Based on efficiency values, units 5 and 6 are identified as needing improvement.

As for **FRH**, it falls between FDH and CRS in terms of size and, as discussed in class through an arrow network, seeks to replace deterministic data with random variables.

COMPILING THE RESULTS FOR EFFICIENCY

```
# COMPILE THE RESULTS FOR EFFICIENCY FOR ALL THE MODELS
results <- data.frame(FDH = eff_fdh, CRS = eff_crs, VRS = eff_vrs, IRS =
eff_irs, DRS = eff_drs, FRH = eff_frh)

# NOW, LET'S DISPLAY THE RESULTS
results

##   efficiency_fdh efficiency_crs efficiency_vrs efficiency_irs
efficiency_drs
## 1      1.0000000      1.0000000      1.0000000      1.0000000
1.0000000
## 2      1.0000000      1.0000000      1.0000000      1.0000000
1.0000000
## 3      1.0000000      0.8793468      1.0000000      0.8793468
1.0000000
## 4      1.0000000      1.0000000      1.0000000      1.0000000
1.0000000
## 5      1.0000000      0.8941998      0.9239332      0.9239332
0.8941998
## 6      0.8823529      0.7047619      0.7272727      0.7272727
0.7047619
##   efficiency_frh
## 1      1.0000000
## 2      1.0000000
## 3      1.0000000
## 4      1.0000000
## 5      1.0000000
## 6      0.8823529
```

Here, we are compiling the efficiency values from different DEA models since it offers a holistic view of how various Decision-Making Units (DMUs) perform. It enables comparative analysis, robustness testing, and the identification of areas for improvement. When DMUs consistently perform well across models, it increases confidence in their efficiency. Conversely, variations suggest sensitivity to specific model assumptions. This compilation helps in decision-making to specific goals and provides strategic insights for resource allocation and management. Ultimately, it gives a comprehensive understanding of efficiency, facilitating more informed decisions to enhance overall performance.