

MCDM Techniques Using R
Prof. Gaurav Dixit
Department of Management Studies
Indian Institute of Technology – Roorkee

Lecture - 14
TOPSIS - Part II

Welcome to the course MCDM Techniques using R. So in previous lecture we started our discussion on TOPSIS. So we talked about you know different details about TOPSIS. So let us do a quick recap of what we discussed in the previous lecture.

(Refer Slide Time: 00:40)

TOPSIS

- Stands for
 - ‘Technique of Order Preference Similarity to the Ideal Solution’

 - Requires a minimal number of inputs
 - Subjective parameters
 - Criteria weights

 - Output is easy to understand

So we talked about TOPSIS what it stands for Technique of Order Preference Similarity to the Ideal Solution. So we talked about that typically you know only the criteria weights are to be specified by decision maker. Simple technique, easy to understand.

(Refer Slide Time: 00:56)

TOPSIS

- Main idea
 - Best solution is the one which has the shortest distance to the ideal solution and
 - The farthest distance from the anti-ideal solution
 - Example:
 - Two criteria to be maximized having equivalent weights
 - Two alternatives A and B
 - We have to check their closeness with the ideal and anti-ideal solution

Main idea is that the best solution is actually compared with the ideal and anti-ideal solution and you know based on that idea you know we talked about 5 computation steps that are part of this method. So one is performance scoring then we do normalization then weighting of those normalized score then distance computation from ideal and non-ideal points. And then finally computing the closeness ratio.

So these are 5 basic steps that we talked about. We also discussed each of these steps in more detail in the previous lecture.

(Refer Slide Time: 01:33)

TOPSIS

- Open RStudio
 - Example: Ranking of cars
 - Three alternatives: "Corsa", "Clio", "Fiesta"
 - Four criteria: "Purchase Price", "Economy", "Aesthetics", "Boot Capacity"

Then at the last part of the lecture we stopped at this point where we were talking about this particular example exercise that we would like to do in RStudio environment. So this example is again on ranking of car just like we did in ELECTRE. So in this particular case

we have 3 alternatives you know 3 cars you see the names here and then we have the 4 criteria. So you can see the criteria as well purchase price, economy, aesthetics and boot capacity so these are the 4 criteria which are to be considered.

(Refer Slide Time: 02:08)

```

1 # using mcda package
2 #
3 #
4 # goal: ranking of cars
5 # criteria (4): "Purchase Price", "Economy", "Aesthetics", "Boot Capacity"
6 # Alternatives (3): "Corsa", "Clio", "Presta"
7
8 # Matrix with Performance scores
9 performanceTable = matrix(c(5490, 51.4, 8.5, 285,
10 6500, 70.8, 7, 288,
11 6489, 54.3, 7.5, 290),
12 nrow=3, ncol=4,
13 byrow=TRUE); performanceTable
14
15 # names(performanceTable) = c("Corsa", "Clio", "Presta")
16
17 colnames(performanceTable) = c("Purchase Price", "Economy",
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

```

> # Matrix with Performance scores
> performanceTable = matrix(c(5490, 51.4, 8.5, 285,
+ 6500, 70.8, 7, 288,
+ 6489, 54.3, 7.5, 290),
+ nrow=3, ncol=4,
+ byrow=TRUE); performanceTable
+
+      [,1] [,2] [,3] [,4]
+ [1,] 5490 51.4 8.5 285
+ [2,] 6500 70.8 7.0 288
+ [3,] 6489 54.3 7.5 290
+

```

So what we will do is let us open RStudio. So now for implementing TOPSIS model in R environment we have 3 packages and so we will go through each of this packages you know and understand what are the implementations, what are the functions that they have implemented and what is the kind of input that these function (()) (02:38) and the kind of output that is being generated.

However, in terms of functionality there is not much difference, but still one of course each of these interfaces are slightly from each other so we will go through you know one by one. So first package that we are going to cover MCDA package so this package we had also used previously. So you can see goal ranking of cars we have 4 criteria and 3 alternatives. So first we need to generate this performance matrix.

So this performance table just like in the previous techniques we had used the matrix function to generate this performance table. So here you can see since we have 3 alternative and 4 criteria. So you can see we have 12 values in this first argument in the matrix function and these 12 values have been combined using this C function combined function and then the second argument is the number of rows that is 3.

And third argument is number of column that is 4 and by row whether values are to be

arrange by row or by columns. So this particular aspect of matrix function we have talked about in previous lectures as well. Now let us execute this. So let us run this code and you can see here that matrix has been generated. You can see for the column 1 the values are in the similar range and this column 2 the values are in the same range for column 3 and similarly for column 4.

So because these columns are corresponding to the criteria and the rows are corresponding to alternatives right

(Refer Slide Time: 04:40)

```

15 row.names(performanceTable) = c("Corsa", "Clia", "Fiesta")
16
17 colnames(performanceTable) = c("Purchase Price", "Economy",
18 "Aesthetics", "Boot Capacity"); performanceTable
19
20 # Criteria weights
21 weights = c(0.35, 0.25, 0.25, 0.15)
22 names(weights) = colnames(performanceTable); weights
23
24 # Preference direction for each criterion
25 criterionmax = c("min", "max", "max", "max")
26 names(criterionmax) = colnames(performanceTable); criterionmax
27
28 # Virtual ideal alternative
29 positiveIdealSolutions = c(0.17952378, 0.172636015, 0.210499658, 0.087192767)
30 names(positiveIdealSolutions) = colnames(performanceTable); positiveIdealSolutions
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

```

> colnames(performanceTable) = c("Purchase Price", "Economy",
+ "Aesthetics", "Boot Capacity"); performanceTable
+
+ Purchase Price Economy Aesthetics Boot Capacity
+
+ Corsa 5490 51.4 8.5 285
+ Clia 6500 70.6 7.0 288
+ Fiesta 6489 54.3 7.5 290
+
+ # Criteria weights
+ weights = c(0.35, 0.25, 0.25, 0.15)
+ names(weights) = colnames(performanceTable); weights
+ Purchase Price Economy Aesthetics Boot Capacity
+ 0.35 0.25 0.25 0.15
+

```

So let us also define the names of these rows and columns. So name of rows are going to be based on the alternatives that is name of cars in this case. So let us run this and then number of columns are going to be based on the criteria. So we can see the purchase price, economy, aesthetics and boot capacity. These are the 4 criteria that we are considering so the name of column are going to be based on this.

Now we have the named matrix where each of the rows and each of the columns they are named here and the values are already there. Now the second part is the criteria weights. So this is something that is to be specified by the decision maker. So here in this case you can see here we are going to create weights variables for criteria weights and we have 4 criteria. So we need to specify 4 values.

So for the criterion 1 that is purchase price we have 0.35 then for criteria 2 that is economy we have 0.25 then aesthetics we have 0.25 the boot capacity 0.15. So given these values you

can clearly see here that the purchase price as a criteria it is being given more weightage and the boot capacity as a criterion it is being given as the least weight right. So the decision maker they would like to produce a ranking you know based on giving higher weightage to purchase price.

And then economy and aesthetics being given equivalent, but second highest weightage and the last one is boot capacity. So based on this the decision maker would like to you know introduce the rankings. So let us run this code and create this variables and names of these value you can see here. So in the environment section also you can see that these 2 variables performance table having the performances scores.

And the weights having the criteria weights you can see have been you know defined. Now another thing you know is the preference direction for each criteria just like other techniques here also we need to specify this. So criteria min max variables we are going to create and since we have 4 criteria so therefore you know for each of them we need to specify the directions.

So the first criteria as you can see here is the purchase price so purchase price is something that we would like to minimize and therefore criteria minimize variable the first value is min and then the second criterion is economy. So this is something that we would like to maximize and therefore this (()) (07:39) aesthetics we would like to maximize third value is a max. Boot capacity we like to maximize and therefore fourth value is max.

So preference direction is this is how we can actually go about defining the preference direction. So let us run this code and then names of the so you can see we have recorded the preference direction with respect to each criteria.

(Refer Slide Time: 08:13)

```

22 names(weights) = colnames(performanceTable); weights
23
24 # Preference direction for each criterion
25 criteriaMinMax = c("min", "max", "max", "max")
26 names(criteriaMinMax) = colnames(performanceTable); criteriaMinMax
27
28 # Virtual ideal alternative
29 positiveIdealSolutions = c(0.179573776, 0.171636015, 0.159499658, 0.087302767)
30 names(positiveIdealSolutions) = colnames(performanceTable); positiveIdealSoluti
31
32 # Virtual anti-ideal alternative
33 negativeIdealSolutions = c(0.212610118, 0.124958799, 0.131352659, 0.085797547)
34 names(negativeIdealSolutions) = colnames(performanceTable); negativeIdealSoluti
35
36 ## TOPSIS modeling
37 library(MCDA)
38 mod1 = TOPSIS(performanceTable, weights, criteriaMinMax); mod1
39
281 (Untitled) : R Script :

```

```

C:/Users/T Celi/Desktop/MOOC January 2019/Dr. Garav Dhill/Lecture 13 & 14/MCDM/
****
Fiesta      6489      54.3      7.5      290
> # Criteria weights
> weights = c(0.35, 0.25, 0.25, 0.15)
> names(weights) = colnames(performanceTable); weights
Purchase Price      Economy      Aesthetics      Boot Capacity

```

So once this is done then in this package MCDA and other thing that can be done is you know though we talked about in the previous lecture that the ideal alternative and anti-ideal alternative are to be defined. We talked about 3 ways and the third way was where even the ideal and anti-ideal point can be taken as input from decision maker. So this particular packages provides this functionality.

So here you can actually define your ideal and anti-ideal alternatives. So this is one distinguishing feature of this particular package right. So here you can see some values have been specified here so virtual ideal alternatives. So against you know so you can see here 4 values are there so given the 4 criterion that we have. So with respect to each of the criteria we need to find out the alternative having the best value for the ideal alternative.

And having the worst value for the ant-ideal alternative. So you can see a first value in this variable positive ideal solution variable you can see this is lower and the first value in negative ideal solution that is higher. So this is because the first criterion was purchased price which we are supposed to minimize as per the given preference direction. So therefore in the ideal the lowest value we will take and the anti-ideal the highest value we will take.

Similarly, if we take the if we look at the other values in the positive ideal solutions here other values they are on the lower side in comparison to what we have here in the negative ideal solution right. So let us generate these variables as well. So you know we can always using this particular package we can always use decision makers input even for ideal and anti-ideal alternatives.

(Refer Slide Time: 10:31)

```
names(criteriaMinMax) = colnames(performanceTable); criteriaMinMax
# virtual ideal alternative
positiveIdealSolutions = c(0.179573776, 0.171636015, 0.159499658, 0.087302767)
names(positiveIdealSolutions) = colnames(performanceTable); positiveIdealSoluti
# virtual anti-ideal alternative
negativeIdealSolutions = c(0.212610118, 0.124958799, 0.131352659, 0.085797547)
names(negativeIdealSolutions) = colnames(performanceTable); negativeIdealSoluti
# Topsis modeling
library(MCDA)
mod1 = Topsis(performanceTable, weights, criteriaMinMax); mod1
# Considering decision maker's input for ideal and anti-ideal points
mod2 = Topsis(performanceTable, weights, criteriaMinMax,
              positiveIdealSolutions,
              negativeIdealSolutions)
# Output in console:
# virtual anti-ideal alternative
# negativeIdealSolutions = c(0.212610118, 0.124958799, 0.131352659, 0.085797547)
# names(negativeIdealSolutions) = colnames(performanceTable); negativeIdealSolutions
Purchase Price      Economy      Aesthetics      Boot Capacity
```

Now what we are going to do is we will do the TOPSIS modeling so before we can use the functions of MCDA package we need to load this particular library in to R environment so let us run this so MCDA is there. Now we can call this TOPSIS function so more detail on TOPSIS function you can always go into the help section and find out. So in the help section you can always you can see here.

So you can see in the help section the TOPSIS function part of the MCDA package technique for order preference by similarity to ideal solution method and you can look at the arguments you can see here. So we have performance table then criteria weights and criteria min max. So these 3 we have already defined then positive ideal solution negative solution also something that we have defined and the alternatives IDs and the criteria IDs.

So if you are interested in understanding what they mean by each of these argument you can read this page and find out. So at this point 2 specific arguments I would like to talk about alternative IDs and alternative IDs. So here this particular function gives us the flexibility to actually select the alternative that we would like to consider. So let us say if there are 10 IDs and for a particular model that we want to specify.

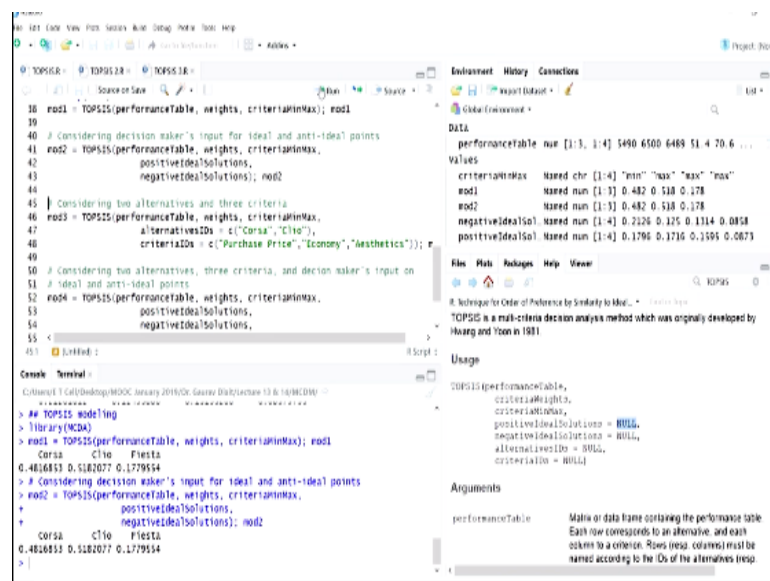
We would like to consider just the 5 alternatives so this can be done using this particular arguments. Similarly, criteria IDs so here also we can do some filtering so if we have 10 criteria to be consider if we want to consider just 5 so we can use this particular argument to actually do this. So in our exercise we will do something similar. So first model that we are

going to build is based on just 3 arguments that is performance table, weights and criterion max.

So if you look at the help section only 3 arguments are mandatory in the sense performance table, criteria weights and criteria min max are to be specified and other could be null as you can see here in the help section. So therefore first model we are going to run is just using 3 arguments where we have the performance scores, the criteria weights and the preference direction for these criteria so let us run this.

So you can see the scores so from these scores you can see that second alternative is having the highest value 0.518 followed by the first alternative having value 0.481 and the third alternative is having value 0.177. So we have got the scores and accordingly we can find out that second alternatives seem to be the best one and you can produce a ranking as well. So this is one model we can build, but because given the functionality that is provided in this package and in this function we can specify some other models as well.

(Refer Slide Time: 14:00)



```
R Console Output:  
38 mod1 = TOPSIS(performanceTable, weights, criteriaMinMax); mod1  
39  
40 # Considering decision maker's input for ideal and anti-ideal points  
41 mod2 = TOPSIS(performanceTable, weights, criteriaMinMax,  
42   positiveIdealSolutions,  
43   negativeIdealSolutions); mod2  
44  
45 # Considering two alternatives and three criteria  
46 mod3 = TOPSIS(performanceTable, weights, criteriaMinMax,  
47   alternativesIDs = c("Corsa", "Clio"),  
48   criteriaIDs = c("Purchase Price", "Economy", "Aesthetics")); r  
49  
50 # Considering two alternatives, three criteria, and decision maker's input on  
51 # ideal and anti-ideal points  
52 mod4 = TOPSIS(performanceTable, weights, criteriaMinMax,  
53   positiveIdealSolutions,  
54   negativeIdealSolutions,  
55   <--->  
56  
Console Terminal:  
C:\Users\1\Call\Desktop\MOOC January 2019\OK_Gaury\04\2\script.R:14:NCEDM >  
> ## TOPSIS modeling  
> library(MCDA)  
> mod1 = TOPSIS(performanceTable, weights, criteriaMinMax); mod1  
Corsa Clio Fiesta  
0.4816853 0.5182077 0.1779554  
> # Considering decision maker's input for ideal and anti-ideal points  
> mod2 = TOPSIS(performanceTable, weights, criteriaMinMax,  
+   positiveIdealSolutions,  
+   negativeIdealSolutions); mod2  
Corsa Clio Fiesta  
0.4816853 0.5182077 0.1779554  
> |
```

Data:
performanceTable: num [1:3, 1:4] 5490 6500 6489 51.4 70.6 ...
criteriaMinMax: named chr [1:4] "min" "max" "max" "max"
mod1: named num [1:3] 0.482 0.518 0.178
mod2: named num [1:3] 0.482 0.518 0.178
negativeIdealSolutions: named num [1:4] 0.2128 0.185 0.114 0.0854
positiveIdealSolutions: named num [1:4] 0.1796 0.1216 0.1395 0.0873

Usage:
TOPSIS(performanceTable,
criteriaWeights,
criteriaIDs,
positiveIdealSolutions = NULL,
negativeIdealSolutions = NULL,
alternativesIDs = NULL,
criteriaMin = NULL)

Arguments:
performanceTable: Matrix or data frame containing the performance table. Each row corresponds to an alternative, and each column to a criterion. Rows (resp. columns) must be named according to the IDs of the alternatives (resp. ...)

So for example in the second model we are going to in the second model we are going to consider the decision maker input for ideal and anti-ideal points. So you can see we have also specified the positive ideal solution and negative ideal solutions the 2 variables that we had you know that we had created. So we are passing these 2 variables as well. Now these ideal and anti-ideal points which are being specified through this function through this called function they are going to be used to perform the TOPSIS modeling.

So let us see whether the results will change or not so let us run this and you see the same results because actually the values that were actually the way we had defined these 2 points ideal and anti-ideal points we are actually using the same values which are actually generated within the function if we do not provide these 2 variables. So the same values were used therefore we are getting the same results here.

However, you can always specify some other values based on the decision makers you know subjective preferences and therefore you will see that results will also change. So this is the second way of specifying model and running it.

(Refer Slide Time: 15:29)

```

43 negativeIdealSolutions); mod2
44
45 # Considering two alternatives and three criteria
46 mod1 = TOPSIS(performanceTable, weights, criteriaMinMax,
47             alternativesIDs = c("Corsa", "Clio"),
48             criteriaIDs = c("Purchase Price", "Economy", "Aesthetics")); #
49
50 # Considering two alternatives, three criteria, and decision maker's input on
51 # ideal and anti-ideal points
52 mod4 = TOPSIS(performanceTable, weights, criteriaMinMax,
53             positiveIdealSolutions,
54             negativeIdealSolutions,
55             alternativesIDs = c("Corsa", "Clio"),
56             criteriaIDs = c("Purchase Price", "Economy", "Aesthetics")); #
57
58 # Print
59
60 # R output:
61 Corsa Clio
62 0.4942822 0.5057179
63
64 # Considering two alternatives, three criteria, and decision maker's input on
65 # ideal and anti-ideal points
66 mod4 = TOPSIS(performanceTable, weights, criteriaMinMax,
67             positiveIdealSolutions,
68             negativeIdealSolutions,
69             alternativesIDs = c("Corsa", "Clio"),
70             criteriaIDs = c("Purchase Price", "Economy", "Aesthetics")); mod4
71
72 Corsa Clio
73 0.5181588 0.4818412

```

Environment

Variable	Class	Dimensions	Values
performanceTable	matrix	[1:3, 1:4]	5490 6500 6469 51.4 70.6 ...
weights	matrix	[1:4]	"min" "max" "max" "max"
criteriaMinMax	named chr	[1-4]	
mod1	named num	[1-3]	0.482 0.518 0.478
mod2	named num	[1-3]	0.482 0.518 0.478
mod3	named num	[1-2]	0.494 0.506
mod4	named num	[1-2]	0.518 0.482

Usage

```

TOPSIS(performanceTable,
       criteriaWeights,
       criteriaMinMax,
       positiveIdealSolutions = NULL,
       negativeIdealSolutions = NULL,
       alternativesIDs = NULL,
       criteriaIDs = NULL)

```

Arguments

- performanceTable: Matrix or data frame containing the performance table. Each row corresponds to an alternative, and each column to a criterion. Rows (resp. columns) must be named according to the IDs of the alternatives (resp. criteria).

Now in the third way now in this third specification the results will change we will see this. So here instead of you know instead of 4 criteria we are going to consider 3 and instead of 3 alternatives we are going to consider you know just alternatives. So you know the argument that we are going to pass here is the first 3 arguments, performance table, weights and criteria min max right. So model 3 is we will consider 2 alternatives and 3 criteria.

So these 2 alternatives you know have been considered and these 3 criteria have been considered. So you can see (()) (16:17) has not been considered and third alternative has not been consider. So we run this model so this is what we get. So see once we remove one alternative so third alternative has been removed one the Fiesta car so this was the value in the previous model.

And now you see since we have just 2 alternatives the numbers have slightly changed. Now

the first alternative score of first alternative is quite close to what we have for the second alternative, but still second alternative remains best even after removing you know third alternative. So for this particular specification we can say in terms of ordering we are getting the same ordering or same rank right.

Now in the fourth model we will consider just 2 alternatives and 3 criteria now but this time using the decision makers input. Now the earlier you know model that we had build there these ideal and anti-ideal points were based on the you know the same values were taken which are actually generated as part of the function. If we do not specify anti-ideal and ideal points now because one alternative and one criterion has been removed.

And on the same input that we had used previously if we use the same input again now the results will change right. So let us run this and you can see here the previous model the values were 0.494 and 0.505. However, in this model the values have changed 0.5181 and 0/5146. If you look at this first alternative has become the best alternative and the second alternative has become the second right.

So because this time we have used the decision makers input so the results are changing in the previous model we had not used the decision makers input for ideal and anti-ideal points. So those ideal and anti-ideal points were generated internally within the function as per the underlying coding of that function. And therefore the results were different. So this particular package and function has lot of functionality to actually specify different types of TOPSIS model.

(Refer Slide Time: 18:53)

```

1 # Joke!
2 # using mcdm package
3 #
4 # Goal: ranking of cars
5 # Criteria (4): "Purchase Price", "Economy", "Aesthetics", "Boot Capacity"
6 # Alternatives (3): "Corsa", "Clio", "Prieta"
7
8 # Matrix with Performance scores
9 performanceTable = matrix(c(5400, 51.4, 8.5, 285,
10 6500, 70.4, 7, 280,
11 6489, 54.1, 7.5, 290),
12 nrow=3, ncol=4,
13 byrow=TRUE); performanceTable
14
15 row.names(performanceTable) = c("Corsa", "Clio", "Prieta")
16
17 col.names(performanceTable) = c("Purchase Price", "Economy",
18 "Aesthetics", "Boot Capacity")
19
20 # Run TOPSIS
21
22 # Output
23
24 Corsa Clio
25 0.4942822 0.5057179
26
27 > # Considering two alternatives, three criteria, and decision maker's input on
28 > # ideal and anti-ideal points
29 > mcdm = TOPSIS(performanceTable, weights, criteriaMmax,
30 + positiveIdealSolutions,
31 + negativeIdealSolutions,
32 + alternativesIDs = c("Corsa", "Clio"),
33 + criteriaIDs = c("Purchase Price", "Economy", "Aesthetics")); mcdm
34
35 Corsa Clio
36 0.5181588 0.5140270
37
38 >

```

Environment: Global environment

DATA	performanceTable	num	[1:3, 1:4]	5400	6500	6489	51.4	70.4	54.1	8.5	7	7.5	285	280	290	
Values	criteriaMmax	named chr	[1:4]	"min"	"max"	"max"	"max"									
	mod1	named num	[1:3]	0.482	0.518	0.178										
	mod2	named num	[1:3]	0.482	0.518	0.178										
	mod3	named num	[1:2]	0.494	0.506											
	mod4	named num	[1:2]	0.518	0.515											

Usage

```

TOPSIS(performanceTable,
       criteriaMmax,
       criteriaMmin,
       positiveIdealSolutions = NULL,
       negativeIdealSolutions = NULL,
       alternativesIDs = NULL,
       criteriaIDs = NULL)

```

Arguments

- performanceTable: Matrix or data frame containing the performance table. Each row corresponds to an alternative, and each column to a criterion. Rows (resp. columns) must be named according to the IDs of the alternatives (resp.

Now what we are going to do is we will look at another package which is offering us some other functionality so we will see about it. So name of this package is MCDM package right. So the example remains same so we have the same goal ranking of cars. We have the same number of criteria and same criteria actually 4 criteria are there purchase price, economy, aesthetic, boot capacity. We have 3 alternative as you can see here.

So we are tackling the same problem same decision problem, the performance matrix is also same, performance scores are same all these details remain same, weights are same, preference direction is also same. Now TOPSIS modeling here now we will have to load this library this package MCDM so let us run this. So it seems this package is not installed so let me first install this particular package.

So once this is done we will use the functions of it. So just like in previous lecture this is the command that we can use and installed out packages and the name of the package is MCDM. So within double codes we can enter this and this would be installed. So in this particular package we actually have 2 functions to perform TOPSIS modeling. So we will specify models using each of those functions.

So let us do that so let us first you know this installation is complete so let us first load this library this is done. Now first function that we are going to use is TOPSIS linear. And the second function that is available is TOPSIS factor. So these 2 TOPSIS function are different in terms of the normalization procedure that is in built in the coding of these functions. So TOPSIS linear is having a normalization procedure which is similar to what we talked the

ideal you know normalization in the previous lecture and the TOPSIS vector.

You know this particular function has a normalization procedure which is similar to what we talked to distributive normalization. However, the actual details are slightly different, but in terms of understanding you know they are similar. So TOPSIS linear the normalization procedure is similar to ideal and TOPSIS factor normalization procedure is similar to what we discussed in distributive.

So let us we are going to specify 2 models for each of these functions. So first one is TOPSIS linear and we are going to specify you know these 3 arguments are available with us. If you compare with the previous package that is you know MCDA package so we have more number of arguments with more functionality to specify different kinds of models. However, in this case we can just specify the basic variable that is performance scores, the weights of the criteria.

And the preference direction and everything else is computed as for the functionality coded into these 2 functions so let us run this. So you can see here this is the score that we get if you look at this number and right and you look at the numbers that we have got you know previous package see so this is what we had got using the previous package right and if you look at the numbers that we have got now right.

So previous package numbers and this one right. So in the second car was found to be the best one. And then now what we have is it is actually the first car which has been found. So we changed the normalization procedure and the results have changed. Now the first car is having the highest value and this is considered to be now this is coming out to be the best car here. However, in the previous package it was different.

This is mainly because so in the previous package the normalization procedure was actually different. It was similar to the distributive normalization that we had.

(Refer Slide Time: 23:57)

```

22 names(weights) = colnames(performanceTable); weights
23
24 # Preference direction for each criterion
25 criteriaMinMax = c("min", "max", "max", "max")
26 names(criteriaMinMax) = colnames(performanceTable); criteriaMinMax
27
28 # Topsis modeling
29 library(MCDA)
30
31 # Normalization procedure: similar to Ideal
32 mod1 = TopsisLinear(performanceTable, weights, criteriaMinMax); mod1$R
33
34 # Normalization procedure: similar to Distributive
35 mod2 = TopsisVector(performanceTable, weights, criteriaMinMax); mod2$R
36
37 # PREFER
38
39 # End of script

```

```

The downloaded binary packages are in
  C:\Users\E T Cell\AppData\Local\Temp\kitespakuP0r\downloaded_packages
> # Topsis modeling
> library(MCDA)
> # Normalization procedure: similar to Ideal
> mod1 = TopsisLinear(performanceTable, weights, criteriaMinMax); mod1$R
[1] 0.5072407 0.4926404 0.2779594
> # Normalization procedure: similar to Distributive
> mod2 = TopsisVector(performanceTable, weights, criteriaMinMax); mod2$R
[1] 0.4816118 0.5182077 0.2779594

```

Environment: R [64-bit] x86_64-w64-mingw32/x64
 Data: 3 obs. of 3 variables
 3 obs. of 3 variables
 performanceTable num [1:3, 1:4] 1490 6500 6480 51.4 70.6 ...
 Values
 criteriaMinMax Named chr [1:4] "min" "max" "max" "max"
 mod1 Named num [1:2] 0.494 0.100
 mod4 Named num [1:2] 0.518 0.115

Usage
 Topsis(performanceTable,
 criteriaMinMax,
 criteriaWeights,
 positiveIdealSolutions = NULL,
 negativeIdealSolutions = NULL,
 alternativeIDs = NULL,
 criteriaInfo = NULL)

Arguments
 performanceTable Matrix or data frame containing the performance table
 Each row corresponds to an alternative, and each column to a criterion. Rows (resp. columns) must be named according to the IDs of the alternatives (resp.

So let us now see we use the second function which is actually using the distributive kind of normalization whether the ranking comes out to be same like the previous package. So this the name of the function Topsis vector as you can see here. So what we will do we will run this and you can see the results are different. So here you know second car comes out to be the best car and the first car comes out to be the second car.

Now if you compare these number to what we had in the previous package so these numbers are same right. So in the previous package that is MCDA package probably they were also using the distribution normalization procedure. Here Topsis factor is using a similar kind of normalization procedure. However, Topsis linear is different and you can see the results also comes out to be different.

So depending on you know the changes depending on the changes the way we specify our model few things that we change here and there the ranking might actually change. So that has been one of the limitation or critic of you know MCDM techniques in general. So now this is the second package that is available in RStudio in R environment now we will talk about the third package that is available.

So name of this package is Topsis and again we are going to use the same example ranking of car. We have 4 criteria same 4 criteria purchase price, economy, aesthetics, boot capacity and same 3 alternatives 3 cars that are to be ranked. You have the same performance table again and then the other details are same, criteria weights they are same. Now preference direction in this package are specified differently.

So in previous 2 package you know we specify using these character values max min however in this case we use -+ these character – and+. So if it is to be criterion is to be minimized then we use – if it is to be maximize then we use +. So therefore you would see that our variable criterion min max now this has changed slightly so we will run this again. Now to be able to run this model using this particular package.

We need to have this TOPSIS package installed. So what we will do I think probably you know this package is not installed in this particular system. So we again will use the same command, same function installed out packages and we will install this with double quotes. We have to specify TOPSIS and let us install this package. This is done now we will load this package so that we are able to use the functions which are part of this one.

So let us run this now the function that is available in this package is named TOPSIS you know in small letters. So more details on this particular package you can always go into the help section and confine to more details also. Similarly, for the other package MCDM package the TOPSIS you know linear and TOPSIS vector function for them also. For more details, you can always refer to the help section. Now here we are going to specify just one model and this function has just 3 basic parameter.

So performance table, weights and criteria min max we have already specify. So we run this line of code then you can see this codes. and if you know if we want to compare in this codes that we have just got you can see the second alternative second car comes out to be the best cars and the scores are similar to you know what we got in the previous 2 packages as well in some of the models.

So here also distributive mode of normalization is used and so we you know just look at the 3 packages that we have covered you know the MCDM, MCDA and this TOPSIS package. So you know the MCDM packages gives us 2 was to normalize so that is different. However, the MCDA package gives us flexibility in terms of specifying the ideal and anti-ideal points and also in terms of you know filtering the alternatives that we would like to consider for a specific model or the criteria that we would like to you know consider for a specific model.

However, this TOPSIS package this particular package provides the least functionality out of

3 packages that we have. So probably you would like to prefer MCDA and MCDM packages for your TOPSIS modeling. So let us go back to our discussion. So now through R we have understood how you know different steps of TOPSIS can actually be implemented and the modeling TOPSIS modeling can actually be done. Now let us move forward and will discuss few more points on TOPSIS.

(Refer Slide Time: 29:38)

TOPSIS

- Further Comments on TOPSIS
 - Sometimes gives illogical results
 - Extreme alternative might get preferred over a superior compromise

 - Euclidean distance metric
 - Magnifies large distances
 - Might lead to different results than what we can get from using Manhattan distance metric based methods

So you know it has been observed that sometimes you know TOPSIS provides illogical results in the sense because the main idea as we talked about is comparison of the alternative from the ideal and anti-ideal points and it might so happen that given the data that you might have that sometimes some extreme points might be picked up as the best solution by the TOPSIS modeling.

And there might be a superior compromise you know alternative that might be available there so that might not get picked. So sometimes you would feel that it is giving illogical results and extreme alternatives they are getting preferred over a superior compromise alternative. Then the second you know second point about TOPSIS is the use of Euclidean distance matrix.

So the way Euclidean distance matrix is defined Mathematically the way it is done, the way computations are done it typically magnifies large distances. So you know so the in a way the values which are in a way you know particular scores which are having values they will (()) (30:53) dominate the overall computation and therefore overall results. So this has been one limitation of Euclidean distance matrix.

That is why you know if you compare the results that you get using the Euclidean distance matrix and compare the results that you might get otherwise using the you know Manhattan distance matrix so the result might be different. So this is another limitation.

(Refer Slide Time: 31:20)

TOPSIS

- Further Comments on TOPSIS
 - Also classified as a 'distance based method'
 - Also classified as a 'reference based approach'

Now as I have talked about that TOPSIS this particular method is referred to as a distance based method as well because of the distance based matrix that we use and distance based computation being the main core of the way alternatives are ranked. Similarly, it is also referred as reference based approach because these alternatives are actually being compared to with respect to ideal or anti-ideal scenario.

So in that sense this method is also called as reference based approach. So with this we would like to stop our discussion on TOPSIS and in the next lecture we will start with another technique. Thank you.