MCDM Techniques using R

Prof. Gaurav Dixit
Department of Management Studies
Indian Institute of Technology - Roorkee

Lecture – 10
ELECTRE – Part III

Welcome to the course MCDM Techniques using R. So in previous few lectures, we have been discussing this particular specific technique called ELECTRE. So let us do a quick recap of what we have discussed till now and then we will move forward. So this part, we have discussed the ELECTRE, it means elimination and choice expressing reality method. We talked about that here also you know pairwise comparisons are involved, but the methodology and algorithm is quite different.

Slightly complex techniques, several technical parameters and the steps that are involved are slightly complex. For different kinds of problems, there are different kind of ELECTRE methods. So all this part was discussed in the previous lectures. We talked about the outranking relation where you know we denoted using a S b where a and b are both alternatives. Then this particular notation means a S b means that a is at least as good as b, this is an example of an outranking relation.

So we measured this outranking relation using the outranking degree which we denote using S(a, b) where a and b you know being two alternatives. So all this parts were discussed. Then we talked about the concordance degree and the discordance degree. So this part was also discussed. Then we talked about difference scenarios and how the partial concordance degree, how this is going to be computed. We also discussed the same thing through this particular graph where we talked about how partial concordance degree denoted by ci(a,b) is actually computed.

Then we talked about the global concordance degree which is nothing but the weighted sum of all the partial concordance degrees. Then we talked about the partial discordance degree, so this was also discussed. We also discussed the scenarios how this is going to be computed based on the given parameters. So in the concordance, the preference and indifference threshold are taken into count, and in discordance, the veto threshold as well as the preference threshold are taken into account.

So this particular graphic also, through this graphic we talked about the partial discordance degree in previous few lectures. Then we talked about the global outranking degree which is based on the summarization of these two degrees which are concordance and discordance degrees. Then we also briefly discussed about the next phase of ELECTRE that is distillation procedure. So in this we talked about that there are two procedures within this, ascending and descending distillation procedures, and a final ranking is generated based on the two pre-orders which are generated using these two procedures.

So this part we were able to cover in previous few lectures. So now in this particular lecture, we will start through an exercise. So we will go through a particular decision problem and do our modeling in R environment using RStudio. Then, we will also try to understand the underlying mathematics behind all those computation, so that part also we will try to cover in this lecture and in the coming lecture as well, but let us first start with this particular exercise.

So goal of this particular decision problem is ranking of 10 French cars and alternatives and criteria. Those details we will discuss in the RStudio. So let us open RStudio. **(Video Starts: 04:43).** So here the package that we require is called OutrankingTtools package. So just like in previous lectures where we covered an R exercise for AHP method, so just like there, we first install a particular package MCDA and AHP package. So similarly here also you can complete your installation step before moving ahead.

So you will have to type install packages and then within the parenthesis, within the double quotes you will have to type the name of this particular package as given here, so this name as it is because this is case sensitive. So therefore, in the install.packages function within the double quotes, you have to type this and your installation will install and will have the required functionality for us to go ahead and do our ELECTRE modeling.

So let us come back to the problem here. So the example that we are taking is ranking of French cars, so that is the goal for this decision problem. Then as you can see the goal is mentioned here in the comment section. So as we saw in the previous lecture when we did an exercise using R, the comments are actually using the hash symbol. We comment out all the points that we have to write apart from the code. So you can see here goal is ranking of French cars and criteria, we have 7 criteria that we are going to consider to achieve this goal.

So these are the 7 criteria. So because this is related to car and we are to perform a ranking exercise of French cars, so you can see here these are some of the criteria that we are going to consider here. So these are related to car performance and acceleration and other things so they are mentioned here, top, speed all those things. So different aspects of cars they had been considered under the 7 criteria. Then we have 10 alternatives. So these are the code names for 10 French cars that we are going to consider in this particular exercise.

So these are the names of alternatives. So we have 7 criteria to consider and 10 alternatives. So first thing that we are going to do is, first we will name the criteria. So you can always create this kind of character vector in the RStudio in the R environment, so criteria and c is the function to create this particular vector or variable. So you can see all the criteria names are part of this line of code. So c is the function which is going to concatenate all this. So c is the function which is going to combine all these characteristics.

More information on this particular function, you can always refer the help panel. So if you just type c here and you will get more details, see combine values into vector or list. So you can look at more details here. The function and different arguments which are part of this function, you can see here. So actually the values which are to be combined are to be passed as arguments. If you are interested looking few examples, you can always look here. few examples are always given in this help page, so you can look.

For example this 1, 7 to 9, so these particular values can be combined using this particular function, so combine values into a vector or list. So what we have done, the names of criteria that we are trying to combine using this particular line of code. So let us run this and immediately you would see that in the environment panel here you would see a criteria variable has been created which is actually a character vector and having 7 values as you can see here and the values are also mentioned here.

So, first we need to get the names and create this, then we need to specify the criteria weights. So you can see here again we are using the same function c and to combine these values. Now this time these are quantitative values you can see. So corresponding to each of the criterion, one particular weight has been initialized. So you can see 0.3, 0.1, 0.3, 0.2, then

0.1, then 0.2 and 0.1. So correspondingly we will have 7 values, so corresponding to each of the criteria, we have to give one value everywhere.

So let us execute this line and you would see the criteria weights. This is a numerical vector consisting of all the numeric values here, as you can see the environment section, we have this. So now we have created these two vectors, character vector for criteria names and then the numeric vector for the criteria weights. Now as we talked about in the previous lecture that we need to set the preference direction for these criteria, whether we would like to minimize or maximize the criteria. So that particular aspect also is to be required here as input.

So we are going to create another variable called minmax criteria. So here we have indicated for each criteria whether it is to be minimized or maximized. So you can see first criteria that is Prix it is to be minimized, then second criteria that is Vmax it is to be maximized, the C120 is minimized and Coff is to be maximized then acceleration is to be minimized and then the two other variables they are to be minimized as well. So for corresponding to each criterion that is there in the given set of criteria, we have to indicate our preference direction.

So let us execute this line of code and we will create another variable. So you can see another character vector has been created if you look at the environment section here, so another character vector has been created and having 7 values for each of the criteria. So you can see these values have been initialized. Now once this part is covered, then we will come back to the alternatives. So all the information, the names and the initial inputs, initialization for criteria has been done.

Now, we will start with creating a vector for the names of alternatives. So here again, same function c is being used. Now this time, character values are being combined to create a character vector. So let us compute this, execute this line, and we will get this. You can see in the environment section another character vector has been created, which is alternatives and there are 10 values which are part of this character vector. So you can see the names of alternatives have been allocated.

So once this is done, then we can move ahead and then we can create another important aspect of ELECTRE where we have to give the performance numbers for all the alternatives with respect to each criterion. So here, the matrix that we are going to compute, so each row is going to correspond to an alternative, so in each of the alternatives in each of the row, we will have 1 alternative. So since in this decision problem we have 10 alternatives, therefore we are going to create 10 rows.

Each column is going to correspond to criterion so since we have 7 criteria, therefore we are going to create 7 columns, so 10 x 7 matrix we are going to create. Each cell of this particular 10 x 7 matrix is going to have a value related to performance. So let us create this matrix. So now there are different mechanisms in R environment to create this particular matrix. So just like in the AHP exercise that we did where we had used the matrix function and the c function that is combined function to actually create the matrix, so here again we are going to use the same thing.

Now at this point you need to understand the matrix function in more detail. So in the last exercise for AHP that we did in R, we did not talk much about the matrix function. So let us understand this function as well because this is one function that we might be using quite often. So you can see the matrix, this is part of the base package here, you can see the name of the package here, so this is part of the base package, and the matrix creates a matrix from a given set of values.

So of course if you look at the usage here, then you can see the first argument that is passed in this particular matrix function is data. So first we need to pass on this data that we are going to use to generate this matrix, then number of rows and number of columns, and then another important argument here is byrow that initializes as false that is default value. So number of row for example in this case we are going to have 10 rows and columns we are going to have 7.

So you can see here in the code that we have here number of row is initialized as 10 and number of column is initialized as 7. Then we can also have another argument called dimnames, which is nothing but dimension names. So in this case because this is a matrix that we are going to generate, it is between alternatives in the row side row dimension and the criteria on the column side. So therefore, we can use this dimnames argument. So you can read here in the help section that we need to assign.

We need to create a list data structure so that can be done using the list function just like what we have done here. So we are using list function and the first argument of list function here is to passed corresponding to the rows which are alternatives, so names of alternatives we have already created, so the same argument, see this variable we are passing as the first argument and then the criteria. So appropriately dimension names would be picked up from these two variables, alternatives and criteria, which have nothing but the names of alternatives and names of criteria.

So we will get the appropriate dimension names. We will also get dimensions of the matrix which is 10 x 7. Now before we go back to the first argument, let us talk about this particular argument byrow. So if you look at the description that is given here in the help section, so this is a logical data type and if false, which is the default setting for this particular argument, the matrix is filled by columns.

So by default if you don't change this particular argument byrow or don't make it any change into it, so a default value of false would be taken and the matrix is going to be filled by columns. So whatever values that we give in the first argument that is the data part, so they are going to be filled by columns. So all values would be taken sequentially and the first column is going to be generated, and once the first column is generated, then second column takes the values in the sequential fashion.

They will keep on taking and it will generate the second column and third column and so on so forth. So you have to have this clear understanding depending on the kind of matrix that you want, the kind of data that you have, how you would like to fill the matrix. So this particular argument byrow is going to be important in this aspect. Now let us look at the first argument, the example for our exercise that we are taking here.

So the numbers you can see in here that we are using c function combined function to combine a number of numeric values. Now here, you will see that first we have a number of values which are in similar kind of range like 103000, then 101300 and you can see almost 10 values are there. So these 10 values are in similar range. So actually what we want is because the columns are going to have the criteria name, so for each criterion we are having the performance numbers for each corresponding alternatives.

So first column you are going to have criteria 1, and for the criteria 1, we are going to have performance numbers for all the alternatives. So therefore, these values are indicated here, first 10 values for the same. So therefore what we want is, we would like to have in the final matrix that is created these values in the first column right, then next 10 values here you can look at this particular matrix in the second column.

So if we look at the way we have given the input in our data argument, the first argument, and the way the matrix is being created specifically as indicated by the byrow argument, so we will get the desired output. Had it been some other way, for example if the values that we are indicating here that we are passing here as the first argument, data argument, if the values are to be displayed like row wise, so whatever values we are indicating they are to be displayed.

First value is to be given first element in the first row, then the second value will go into the second element of the first row, and so on so forth. If that was the filling order for the values in the matrix, then we will have to change this argument byrow, it has to be made true so that the values are filled row wise. So, however, for our case, the default setting is working well. So we will just run this code and we will get the matrix. So let us run this. So in the output, you can see here we have created 10 x 7 matrix.

You can see these criteria in AHP also, this particular ELECTRE method also and other techniques also we can consider typically criteria as we can see the variables in our statistical analysis right. So these criteria is very **(()) (22:17)** tool the variables that we have and for each variables and the alternatives we can take analogy of records that we have. When we do a statistical analysis, typically the data that we have is on the column side we have variables and on the row side we have records.

So each record and corresponding to variable, they are going to have values. Similarly here, computer disk generated this matrix and on the column side we have criteria and for corresponding to each criteria, each of our alternative is having one particular number, one particular value, which is indicating its performance with respect to particular criteria. So this matrix, now we have generated. Now let us move forward. So before we move ahead, there is another way through which we can generate matrix.

So just now we have used two function matrix and combine, so combine was used to combine the values that we wanted to pass in the matrix function as first argument. Now there is another way to create this matrix. So here we can use the cbind function. So if you are interested in more details about cbind function, again you can use this help panel and you can type here cbind and then we will get the help page here, the manual page of cbind function. So again, this is also part of the base package.

So you can look at the title combine R objects by rows or columns right. So this is how this function is going to be used. So here we are using cbind. So there is another function called rbind. So cbind is if you want to combine by columns and rbind is to be used if you want to combine by rows. So here since we are using cbind, so therefore we are going to combine by column wise, so therefore in this first argument that we have passed here as part of cbind function.

You can see here that first argument is the values which we would like to display in the first column and the second argument is consisting of the values which we would like to display in the second column. So for each of these columns, we are using the c function, the combined function, and all of these arguments are then represented in this fashion and cbind function is going to create the matrix structure for us. So let us run this code and you will see a similar kind of output there, a similar kind of output can be computed here.

So let run this much. So in this script section that we have here we write the file so you can select any selective part of the code, and if it is a complete code where it can be executed, you can always select that part of the code and you can again press run and you will get the output here. If we look at this execution of cbind function, you can see we have got the similar matrix here. Now let us go back to the previous code and let us run this because we wanted to have the dimension names as well. So now we have the performance table also.

So this particular matrix that we have just created, we are referring this one as the performance table for alternatives with respect to each criterion. Now there are a certain other parameters that we need to initialize just like we talked about in previous two lectures on ELECTRE that there are number of technical parameters that we need to specify that we need to initialize. So now we are going to do exactly the same in this R exercise. So we talked about 3 particular parameters in previous two lectures.

One is indifference threshold, the another one is the preference threshold and the third one is the veto threshold. So you can see in the commented lines here, we are going to define thresholds with respect to each criterion. So again, these thresholds are also with respect to the criterion. So indifference typically denoted using q, the preference threshold typically denoted using p and the veto threshold typically denoted using v.

So now for the initialization, we are going to create these variables; alpha_q so this is one which is related indifference threshold, the beta_q which is also related to indifference threshold. So why we have these two variables here. So these threshold that we create they can be either absolute or relative. So if these thresholds are absolute, then we are actually considering fixed values for these thresholds with respect to each criterion.

Then in that sense, only one variable is enough, but if we are defining these parameters defining these thresholds as relative, so then in that case, we will have to create another variable. So that is why we have alpha and beta, alpha_q and beta_q, alpha_p and beta_p, and alpha_v and beta_v. So for each type of threshold, we have these two variables, one is indicating at least having the values so that the relative value for each of these threshold can be computed.

Now typically the values that we have in alpha_q, they are used to compute, they are there to indicate how the relative computation is to be done. So typically this is based on the linear dependence. For example, percentage of the best performance or percentage of the worst performance. So if we have something like 0.1 here, so for example here you can see 0.1 in alpha_q, so it could be something like the performance, the threshold value is actually 10% of the best performance or worst performance.

Now how do we determine that whether the best performance is to be taken among alternatives or the worst performance is to be taken. For that, we have another parameter which is called mode of definition. So you can see here in the commented line a mode of definition is indicated. So this is to be defined for each threshold and with respect to each criterion. So for each threshold, we have to define this. So you can see here a mode of definition there could be of two types direct or inverse.

So you know direct is one we are taking with respect to worst performance and inverse when we are looking for the best performance. So based on that the alpha_q and alpha_p and alpha_v are going to be used. The values of these variables are going to be used and the appropriate modification would be done in beta_q, beta_p and beta_v. So typically, it is the value because generally we take the linear dependence, so because this is like percentage kind of things, typically these values are multiples.

So you can see in the alpha_q, alpha_p and alpha_v, the values are in a smaller range between kind of 0 and 1 and if you look at the beta_q and beta_p and beta_v, the values look on the higher range right, so these two are multiplied and that is how we typically compute the thresholds **(Video Ends: 31:13).** So at this point, we would like to stop here and we will continue our discussion and we will continue doing this particular exercise in R in the next lecture. Thank you.