INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
NPTEL
NPTEL ONLINE CERTIFICATION COURSE

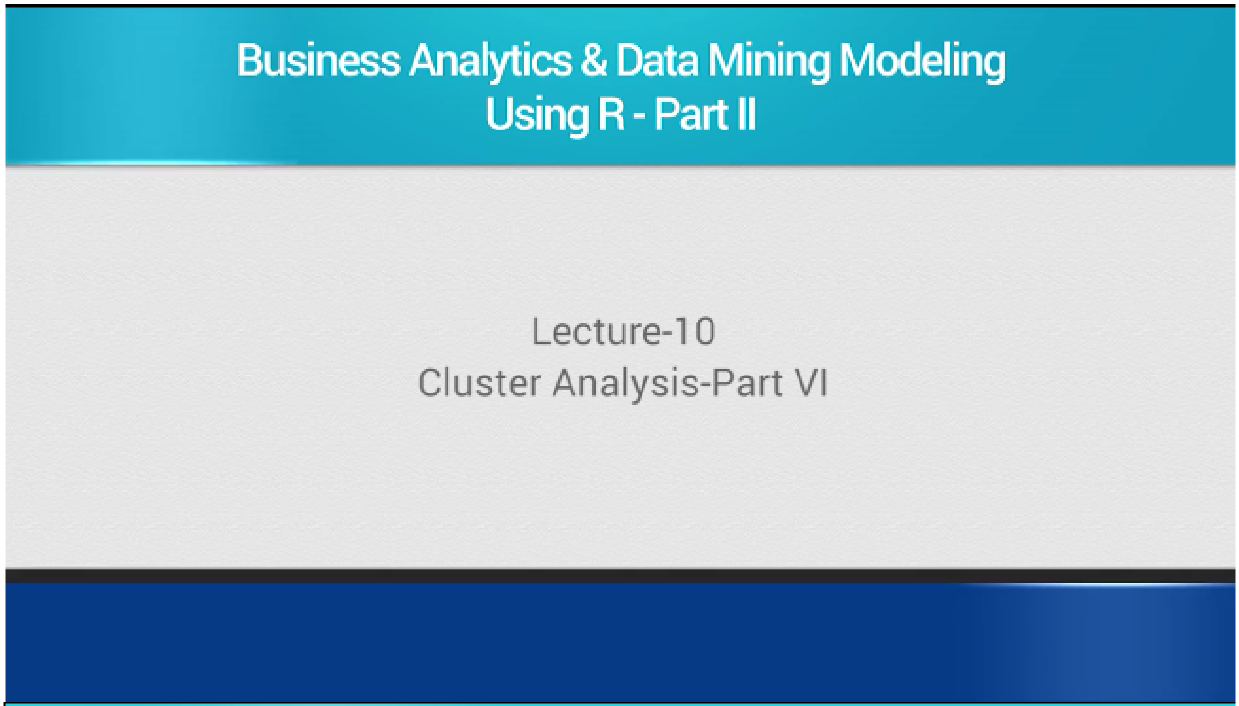
Business Analytics & Data Mining Modeling Using R – Part II

Lecture-10

ClusterAnalysis– Part VI

With

Dr. Gaurav Dixit
Department of Management Studies
Indian Institute of Technology Roorkee

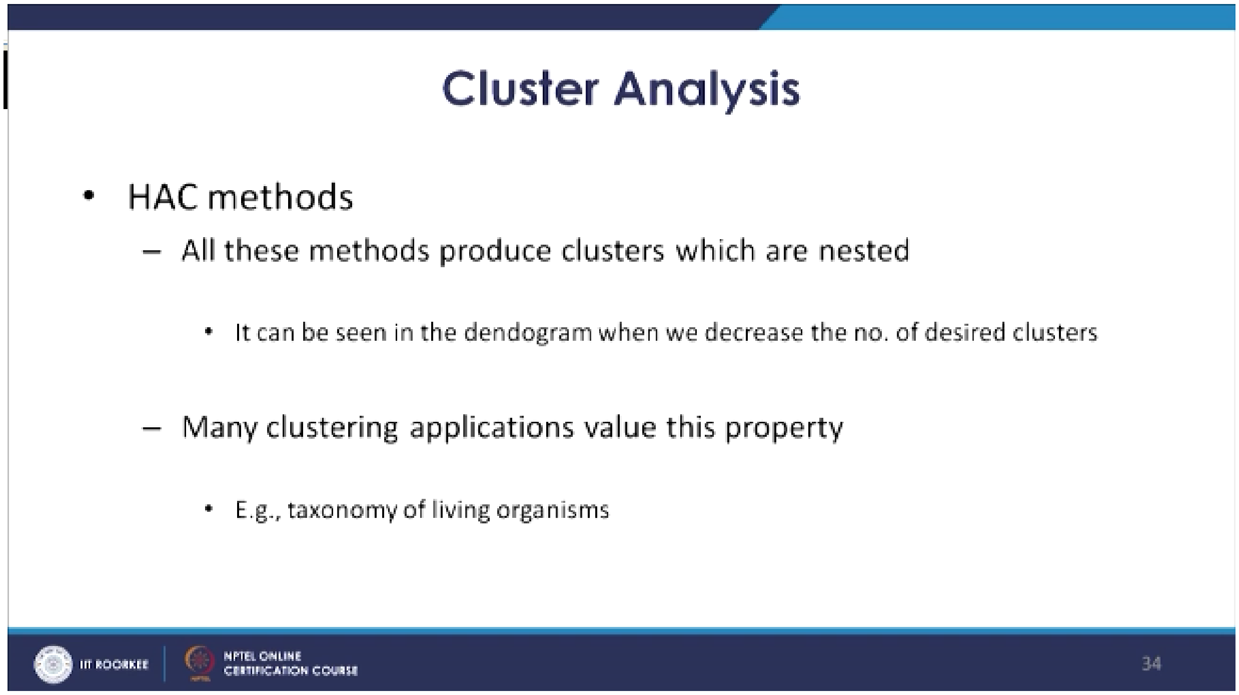


Business Analytics & Data Mining Modeling
Using R - Part II

Lecture-10
Cluster Analysis-Part VI

With
Dr. Gaurav Dixit
Department of Management Studies
Indian Institute of Technology Roorkee

Welcome to the course Business Analytics and Data Mining Modeling Using R – Part 2, so in previous few lectures we have been discussing cluster analysis. So in previous lectures specifically we stopped at the point where we did a small exercise in R, and how an applied cluster analysis on full dataset, we applied Ward's approach, single linkage and average linkage, and we saw how you know the model results can be you know easily understood from the Dendogram, we also saw if we have a number of you know, if we have a number of desired clusters that we want to create, then how that is possible in R, and using Dendogram how that is easily, how that can be done. We also saw in that process each of those clusters and the observations that are going to be part of those clusters can be clearly seen from the Dendogram.



Cluster Analysis

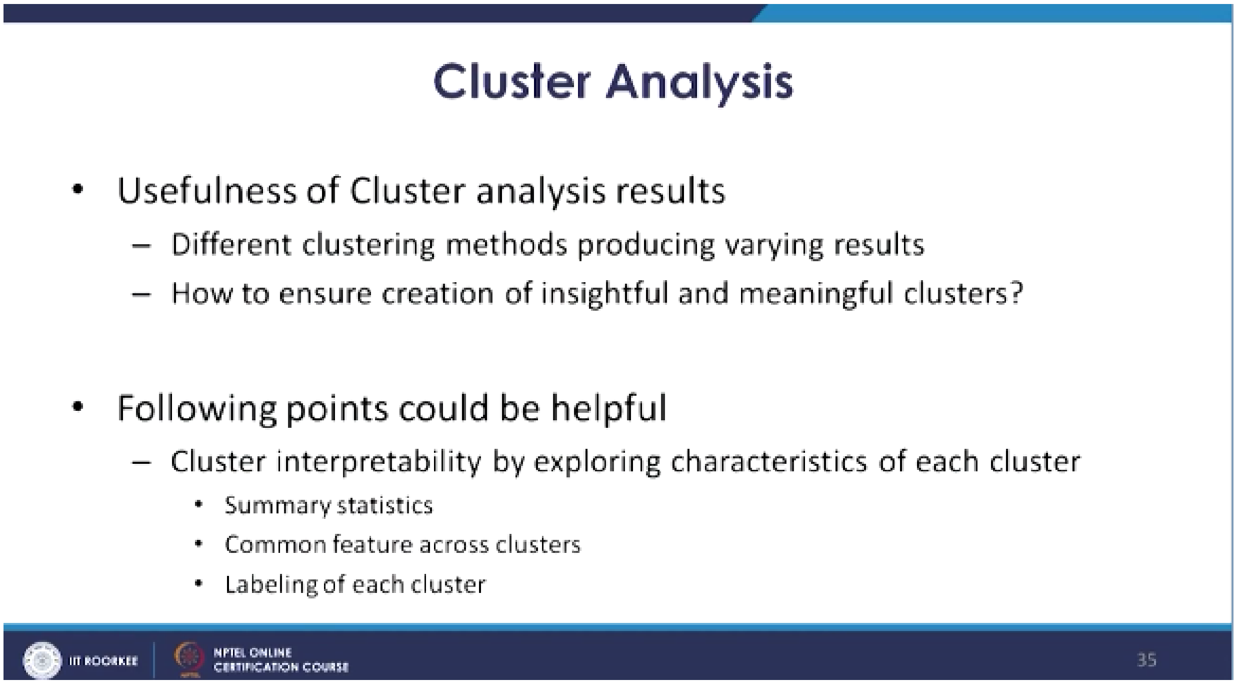
- HAC methods
  - All these methods produce clusters which are nested
    - It can be seen in the dendogram when we decrease the no. of desired clusters
  - Many clustering applications value this property
    - E.g., taxonomy of living organisms

So now let's continue our discussion on cluster analysis, so specifically HAC methods that we have discussed so far, so one thing that you would notice in this that all these methods HAC methods that we have discussed so far, they produce clusters which are nested so that was clearly seen in the Dendogram as well.

Now we can also see that Dendogram, we can see that in the Dendogram when we decrease the number of desires clusters then we can clearly see the nestedness of those clusters was clearly seen in the previous lecture.

Now many clustering applications where actually value this property, the nesting of you know, this nesting of clusters, so one example is taxonomy of living organizations, so you know any particular, any particular you know application where we are creating categories and then subcategories and then subset categories, any you know kind of problem where we are looking to you know group things or cluster things into these kind of you know nested forms, right, we are something is going to be part of something and then you know any, if we are looking for certain things in a particular domain, if we are looking for you know classify certain things in a particular domain, trying to create taxonomy, so this particular property, this nested cluster you know property that we get from HAC method, this is going to be quite useful.

Now let's talk a bit about the usefulness of cluster analysis results, so in the previous lectures we applied few approaches under HAC and we saw that the resulting Dendogram were slightly different the way it happened, and there was some variation in the results, right, few things were common, for example singleton clusters were quite the same and almost all the approaches, however some other observations you know, there was a bit change there, depending on the approach, so how do we ensure the insightful, creation of insightful, and meaningful clusters, so we are going to discuss few points which are going to be useful, which are going to be helpful in terms of creating meaningful and insightful clusters.



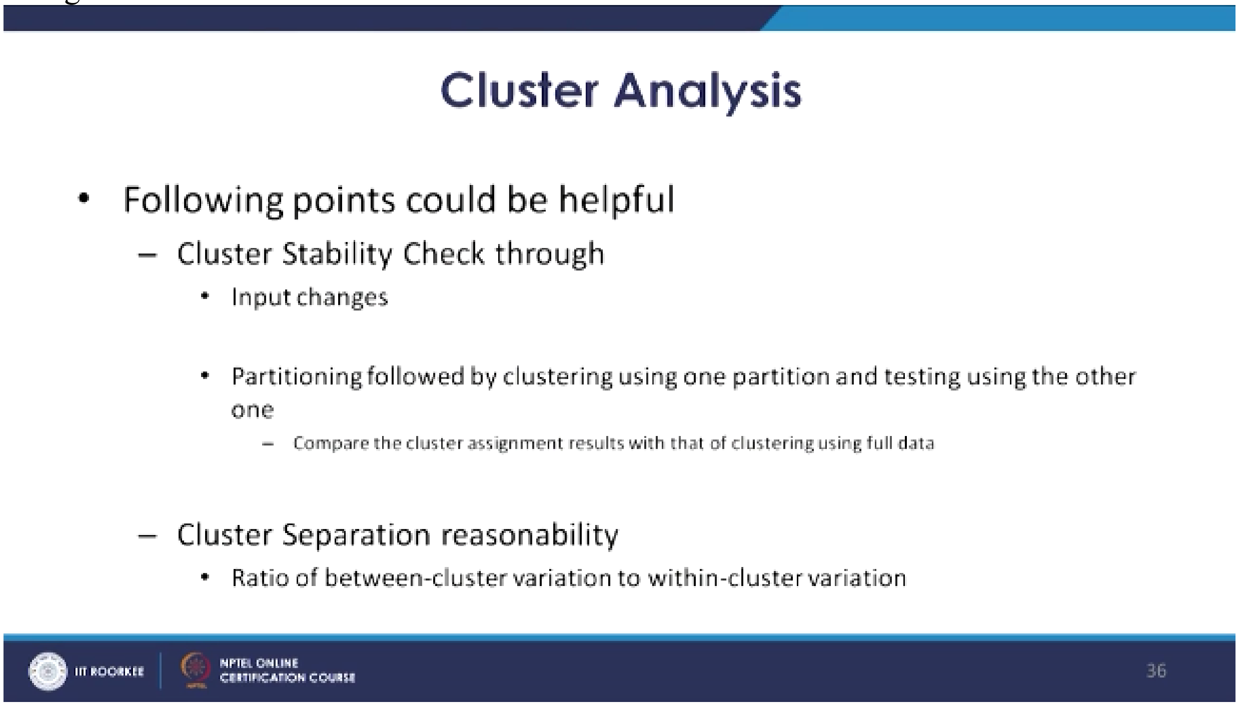
Cluster Analysis

- Usefulness of Cluster analysis results
  - Different clustering methods producing varying results
  - How to ensure creation of insightful and meaningful clusters?
- Following points could be helpful
  - Cluster interpretability by exploring characteristics of each cluster
    - Summary statistics
    - Common feature across clusters
    - Labeling of each cluster

So cluster interpretability this is one important you know thing for creating insightful and meaningful cluster, whether the clusters that you have produced, whether they are you know, whether they can be interpreted, whether they can be understood the way they are, so we can explore some of the characteristics of you know each of these produced clusters, and try to understand you know, try to interpret those clusters what the signify, right, so all those things can be explored, so for example summary statistics is one, so we can look at the summary statistics of each of those you know clusters and try to identify how those clusters are different, whether there is a you know defining characteristic which is you know different, which are you know creating those differences, so those things we can identify and that will give us a sense in terms of interpreting those clusters.

We can also look to identify a common feature across clusters, sometimes this is also you know possible, that for example if we are applying cluster analysis on you know customer segmentation, we are trying to create you know customer segments which could be you know later on used for our marketing and promotion exercises, then in that case sometimes it might so happen that the clustering results might tell you that these customers you know the common feature that you can see is the, might be the geography or the reason, right, so some customers, even though we might have cluster them using a number of demographic and other variables, but eventually when the clusters are formed we might be able to see that you know there is a cluster of you know customers belonging to metro cities then there could be clusters of customers belonging to tier 1, tier 2, tier 3 cities, so geographic you know the common feature might be clearly visible, so we can always look for this kind of common features, summary statistics is always going to be useful, and based on what we learned from some of these things, we can start leveling these clusters, so once we start leveling these clusters our cluster interpretation would be much more meaningful and insightful.

Few other things that could be done, for example we can check the cluster stability, so if you know input changes, if some of the observations change in the dataset, so whether the clusters that were formed they changed significantly or they are stable or robust enough to you know addition or deletion of points, so those kind of checks can also be done, we can always look whether the produced clusters they are robust enough or stable enough for input or data changes.



Cluster Analysis

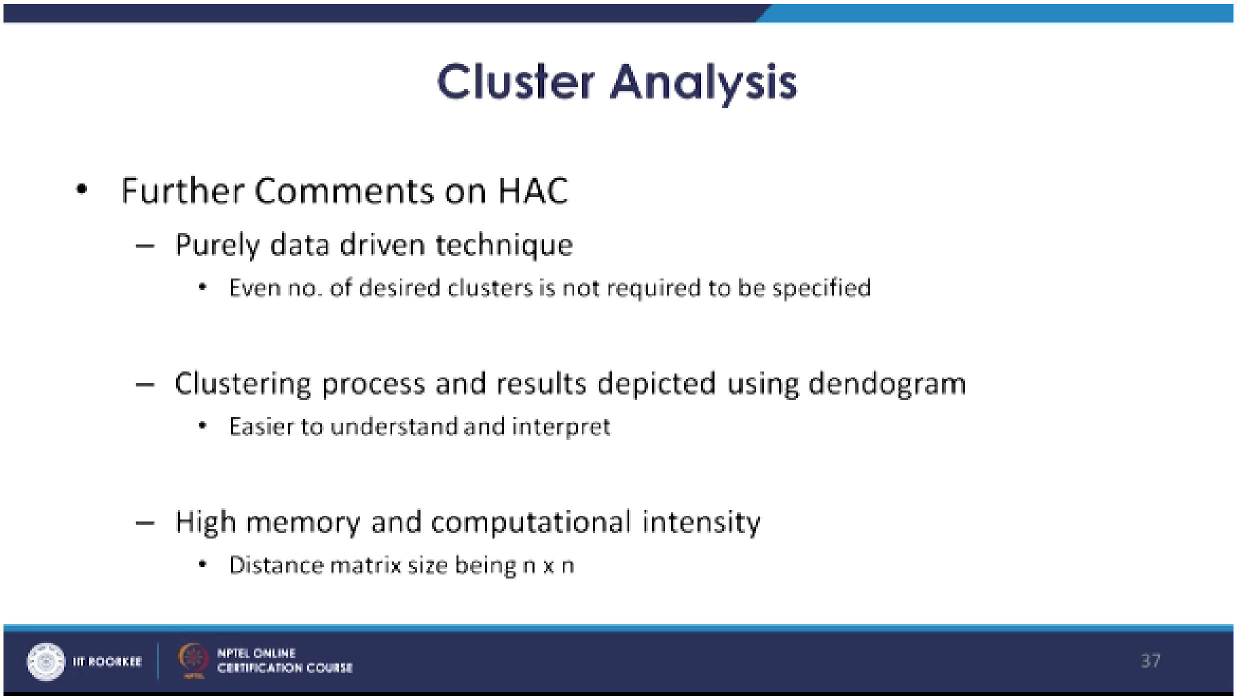
- Following points could be helpful
  - Cluster Stability Check through
    - Input changes
    - Partitioning followed by clustering using one partition and testing using the other one
      - Compare the cluster assignment results with that of clustering using full data
  - Cluster Separation reasonability
    - Ratio of between-cluster variation to within-cluster variation

IIT ROORKEE | NPTEL ONLINE CERTIFICATION COURSE 36

Now another way to check cluster stability could be partitioning followed by clustering, so we can partition the cluster you know and then it can be followed up with the clustering, so we'll get the clusters then the clusters would be you know created using one particular partition and the other partition could be used to test you know these clusters, right, so we can compare how the clusters assignment happened in the partition on which clustering was done, and in the remaining partition where it was applied, so we can compare that and see whether the cluster formation that we did, whether that was stable enough or robust enough, so this approach can also be used.

Another thing that could be done is the cluster separation, whether the clusters they are separated with each other in a reasonable fashion, so that can also be checked, so for this we can use this particular you know measure this particular metric, ratio of between cluster variation to within cluster variation, so this particular value will also give us a sense, idea about how the clusters are separated from each other, so if this value is on the higher side then probably the clusters are separated reasonably, if the value is on the lower side then there might be the separation, the cluster separation might be a problem.

So these are some of the points, cluster interpretability, cluster stability or robustness, and the cluster separation, so these are some of the things that can be analyzed and then we can arrive at whether the produced clusters they provide us, they are going to give us some meaningful or insightful findings.

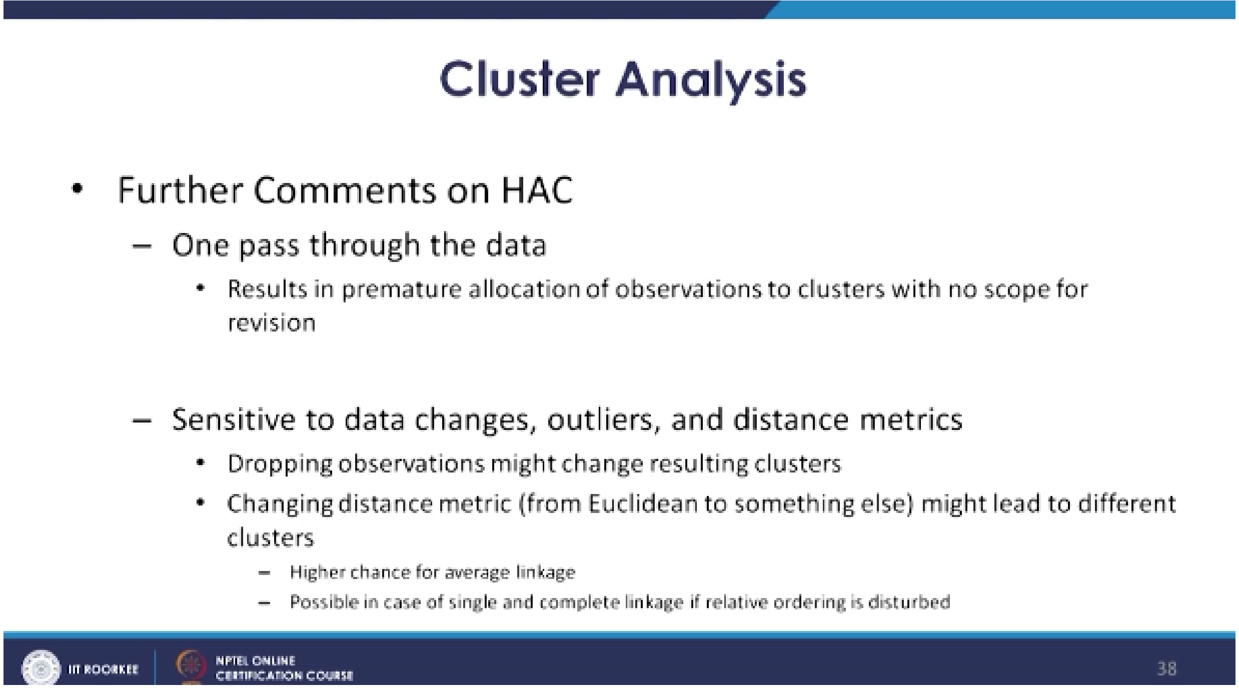


Cluster Analysis

- Further Comments on HAC
  - Purely data driven technique
    - Even no. of desired clusters is not required to be specified
  - Clustering process and results depicted using dendogram
    - Easier to understand and interpret
  - High memory and computational intensity
    - Distance matrix size being n x n

IIT ROORKEE | NPTEL ONLINE CERTIFICATION COURSE 37

To talk about HAC, to summarize few more comments on HAC, now we can see that purely HAC being, purely data driven technique in the sense that we don't even have to specify the number of desired clusters, the process is applied and you know depending on the number of cluster that we want, we can later on you know decide using the Dendogram and we have seen this process, so the whole thing is quite, whole approach is quite data driven.



Cluster Analysis

- Further Comments on HAC
  - One pass through the data
    - Results in premature allocation of observations to clusters with no scope for revision
  - Sensitive to data changes, outliers, and distance metrics
    - Dropping observations might change resulting clusters
    - Changing distance metric (from Euclidean to something else) might lead to different clusters
      - Higher chance for average linkage
      - Possible in case of single and complete linkage if relative ordering is disturbed

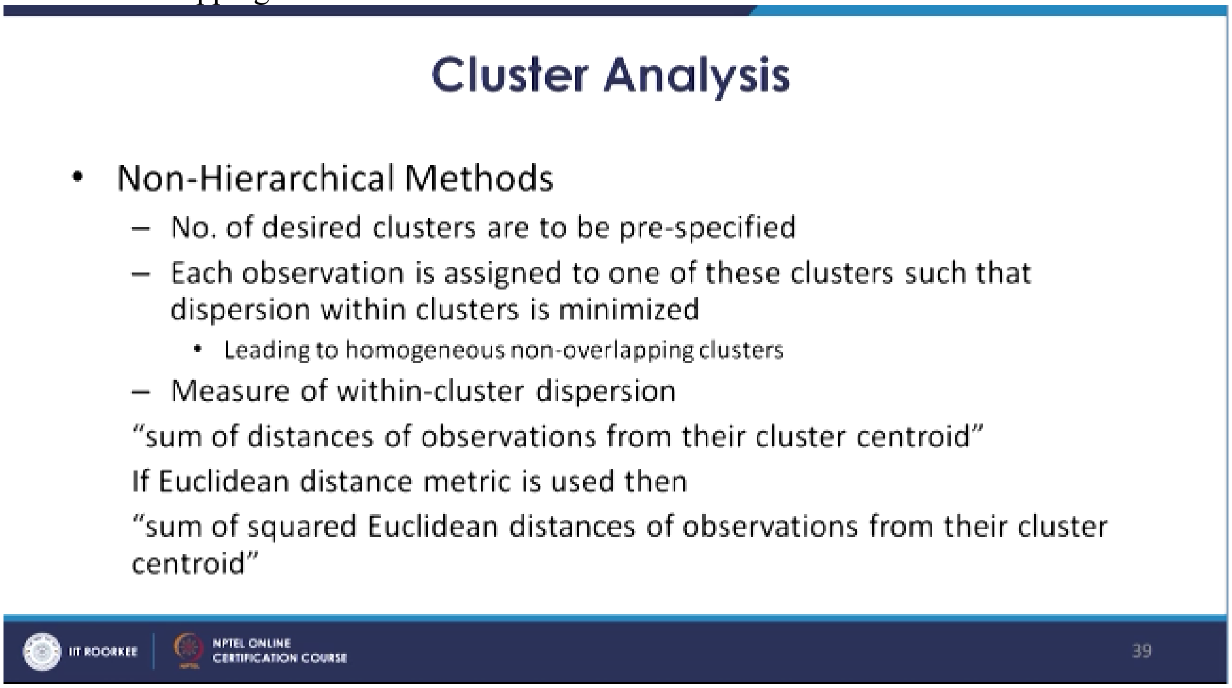
IIT ROORKEE | NPTEL ONLINE CERTIFICATION COURSE 38

So clustering process and results depicted using Dendogram so that we have seen, so we can easily, so using Dendogram it is easier to understand and interpret, so these are some of the plus points of this approach, high memory and computational intensities, so this is one drawback of this particular approach, HAC approaches, because if you remember in the R itself we were creating all the time we were creating this distance metrics you know and these distance metric were of the size N x N, if N is the number of observations, so because of this the lot of storage requirement for implementing this approach, and that could also slow up the processing, so computational intensity would also be on the higher side because of all this you know storage that we required for distance metric computation and the later you know steps also would require, because of this you know they would also require more computational time, so this is one drawback, so the technique overall might be slow.

Then the next limitation of this particular technique is the major drawback where you know we just do one pass through the data, the way whole clustering process spends out in HAC, we just go through the data once, right, so what happens is if you know some premature allocation of records, premature allocation of observations to clusters if that thing happens then we are not left with any scope for revision, so it is just in one go, one pass through the data and we get the you know clusters, and depending on the number of clusters we can always pick, but that result is out and there is no scope for you know revision, so if any premature allocation happens so we cannot do anything with it, so this is one limitation of HAC. Few other limitations, for example sensitive to data changes, if input data a few observations are dropped or added then of course the way clusters are produced that can change, so sensitive to data changes, outliers, sensitive to outliers, so few outliers can also you know determine in a way the clustering, clusters are produced, distance metrics so if we change the distance metric as we saw that Dendogram that we produced you know they were slightly different, but that is just for the you know the distance computation between clusters. If we look at the distance computation between two points, between two observation or any other points, for example you know typically we use Euclidean, so if we change this distance metric from Euclidean to something else then also you know the results might change, so for example average linkage that is something where the high chance of results changing is going to be there because the way average linkage, the way distances are finally computed you know if we change the metric, so those values are also going to be changed, so it is more likely that the results are also going to change.

So in single and complete linkage, they're slightly robust to changes and distance metric, in the sense that they rely in the relative ordering, so as we talked about in the previous lecture, so if relative ordering by the change, if we change the distance metric from Euclidean to something else, if this relative ordering remain same then the results of single and complete linkage are you know typically going to be you know might remain same, however if this relative ordering is disturbed because of this change in the distance metric, then of course the results for even single and complete linkage might also change, so there is going to be a lot of variation in HAC methods, the kind of results, we change the approach, results might change you know add delete few you know observation, the results might change, of course it is you know sensitive to outliers and as we discussed that if we change the distance metric then also results might change, so these are some of the limitations that are there for the HAC approaches, so this brings us to our next part that is non-hierarchical methods, so as we have discussed in previous lectures that you know non-hierarchical methods are slightly different from you know hierarchical methods, but in very first step that we require is the number of desired clusters they had to be pre-specified, so we should have some kind of idea about the number of clusters that we require, and that this particular number is to be specified.

Now what happens in non-hierarchical methods is that after this, once we know that this is the number of cluster, number of cluster that we want, each observation is assigned to one of these clusters such that dispersion within clusters is minimized, so this is one major difference from what we have discussed in HAC approaches, so each observation so there in HAC we were looking to identify the closest observations and we were merging there, but if we look at the non you know non-hierarchical methods the observation is assigned based on the minimization of dispersion, you know, within cluster dispersion. So what happens is it leads to homogeneous and non-overlapping clusters.



Cluster Analysis

- Non-Hierarchical Methods
  - No. of desired clusters are to be pre-specified
  - Each observation is assigned to one of these clusters such that dispersion within clusters is minimized
    - Leading to homogeneous non-overlapping clusters
  - Measure of within-cluster dispersion

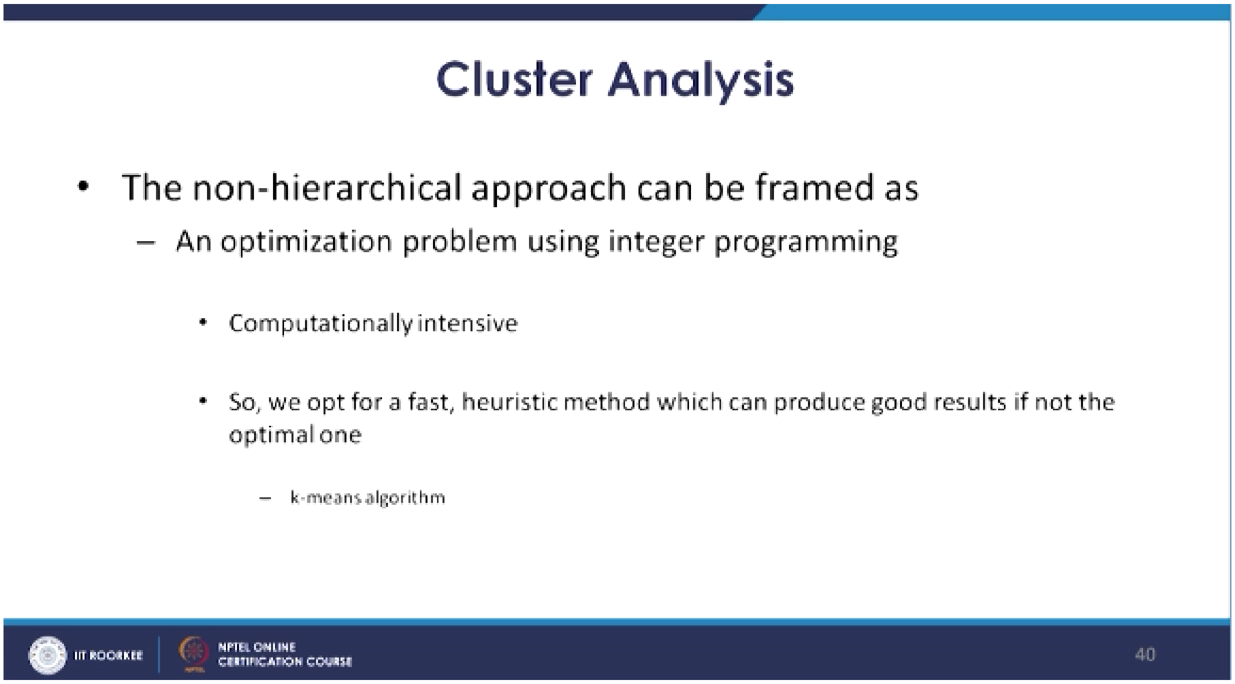
  "sum of distances of observations from their cluster centroid"

  If Euclidean distance metric is used then

  "sum of squared Euclidean distances of observations from their cluster centroid"

IIT ROORKEE | NPTEL ONLINE CERTIFICATION COURSE 39

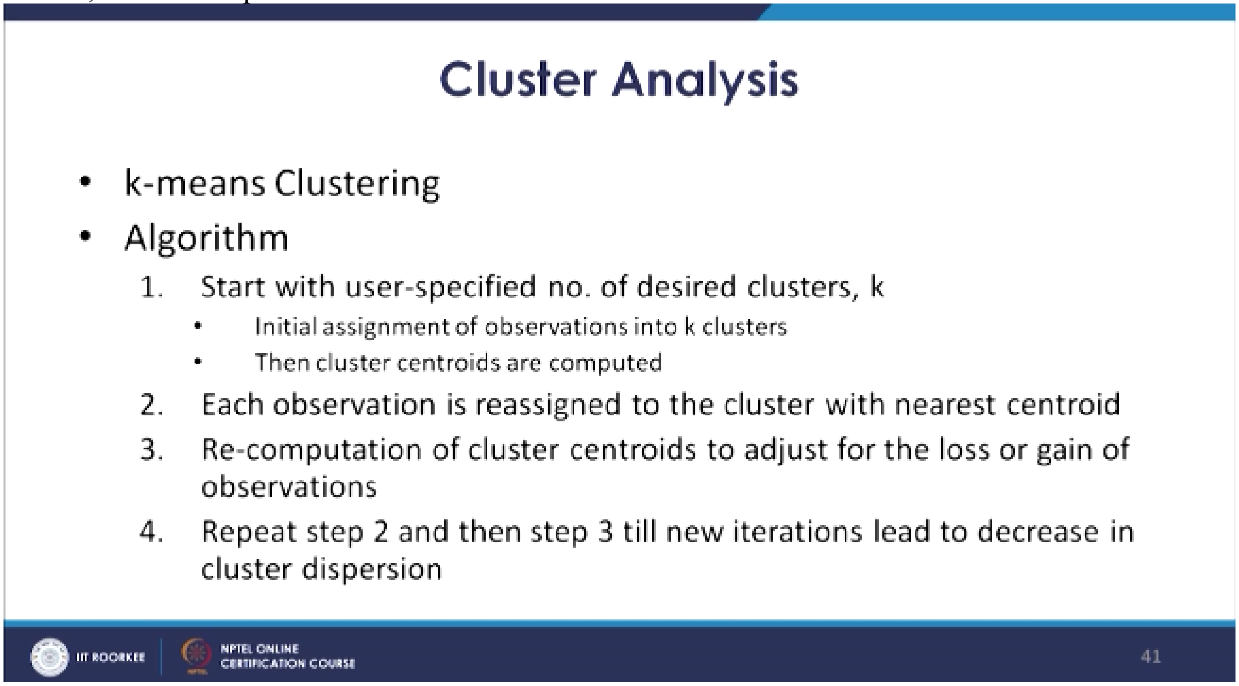
Now typically the more popular and common measure of within cluster dispersion is this, sum of distances of observations from their clusters centroid, so if Euclidean distance metric is used then this, it can become sum of squared Euclidean distances of observations from their clusters centroid, so you can see from the measure itself that all the observations you know they are being you know measured from their centroid, the clusters centroid and the dispersion is being computed, and now the way observation is assigned you know in such a fashion that this dispersion this particular value is minimized, so if you look at the whole, the overall



Cluster Analysis

- The non-hierarchical approach can be framed as
  - An optimization problem using integer programming
    - Computationally intensive
    - So, we opt for a fast, heuristic method which can produce good results if not the optimal one
      - k-means algorithm

formulation of this approach non-hierarchical approach so this approach can be framed as an optimization problem using integer programming, however as we know that integer programming with so many you know with a number of variables, and with a large you know dataset where we might be having, large number of observations, so it might be computationally intensive, because the integer programming will actually look for compute for the optimal you know solution and that might take a you know more time, so that is not desirable so what we can do is we can opt for a fast heuristic method which can produce good results, if not the optimal one.



Cluster Analysis

- k-means Clustering
- Algorithm
  1. Start with user-specified no. of desired clusters, k
     - Initial assignment of observations into k clusters
     - Then cluster centroids are computed
  2. Each observation is reassigned to the cluster with nearest centroid
  3. Re-computation of cluster centroids to adjust for the loss or gain of observations
  4. Repeat step 2 and then step 3 till new iterations lead to decrease in cluster dispersion

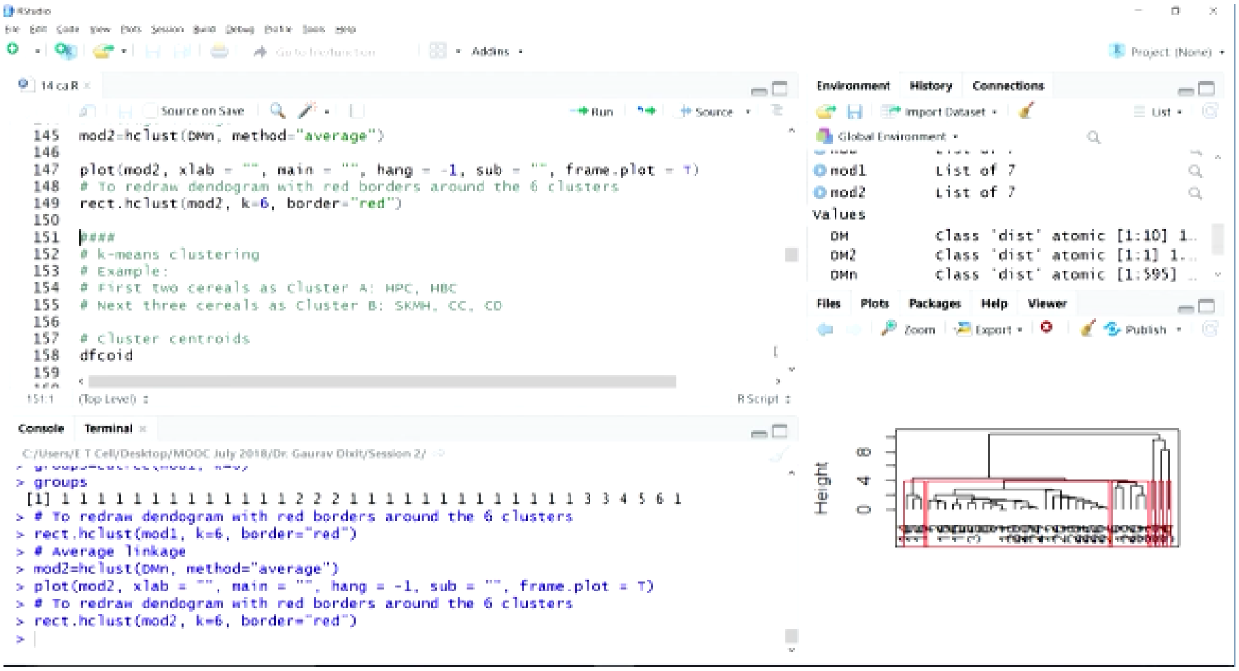
So we are not guaranteed with the, you know, with producing the optimal results but the method can actually produce, give us a good result in a short time, right. So once such method is, K means algorithm, so which is something that we are going to discuss next.

So in non-hierarchical methods this is the algorithm that we are going to cover, K means clustering, so let's start with discussion on the algorithm and we'll also see how it is different from what we discuss in HAC, so the very first step we start with user specified number of desired clusters as I talked about, one difference between hierarchical clustering method and non-hierarchical clustering method is that, in the non-hierarchical you have to specify the number of desired clusters but in hierarchical cluster method everything is part of the process, so the first step is we start with user specified number of desired clusters that is K, and that is how the name comes there, K means clustering.

Now once we know the number of desired clusters then we have to do some initial assignment of observation into these K clusters, right, so this initial assignment will become the basis for implementing next steps of this particular algorithm. Now once this initial assignment has been done we can compute the clusters centroids because now each observation is going to be part of you know some clusters, some clusters as per the you know initial assignment, now we can go ahead and compute the clusters centroid, once this is done we reach to our step number 2, so in this step each observation is now reassigned to the cluster with nearest centroid.

So now all the observations they have been initially initialized to one of the K clusters and then we compute the cluster centroid, and then you know each observation can again be the assigned so we can compute for each observation we can compute, compute its distance from each of those K clusters, and find out which particular centroid is closest to the observation and then reassign this observation to that particular cluster, in this fashion some of the observations you know some clusters might lose some observation and might gain some observation, and this might happen to all the K clusters, so this process will happen.

So this reassignment will result in you know lot of gains and loss of observation from each of those clusters, so what we need to do is we need to re-compute in our third steps, in our third step we can see re-computation of clusters centroids to adjust for the loss or gain of observation, so we will have to re-compute these clusters centroid and once this is done we can again go back to step number 2 and then step number 3, so we can keep on doing this till new iterations lead to decrease in cluster dispersion, so as we talked about the metric that is used is, we look to minimize within cluster dispersion in this process, in this whole process, K means process and in general you know non-hierarchical approach as well, so step 2 and 3 they are going to be repeated till the time the, within cluster dispersion it keeps on decreasing, once you now saturation is decreased, once we can look further decrease this dispersion then we'll stop.



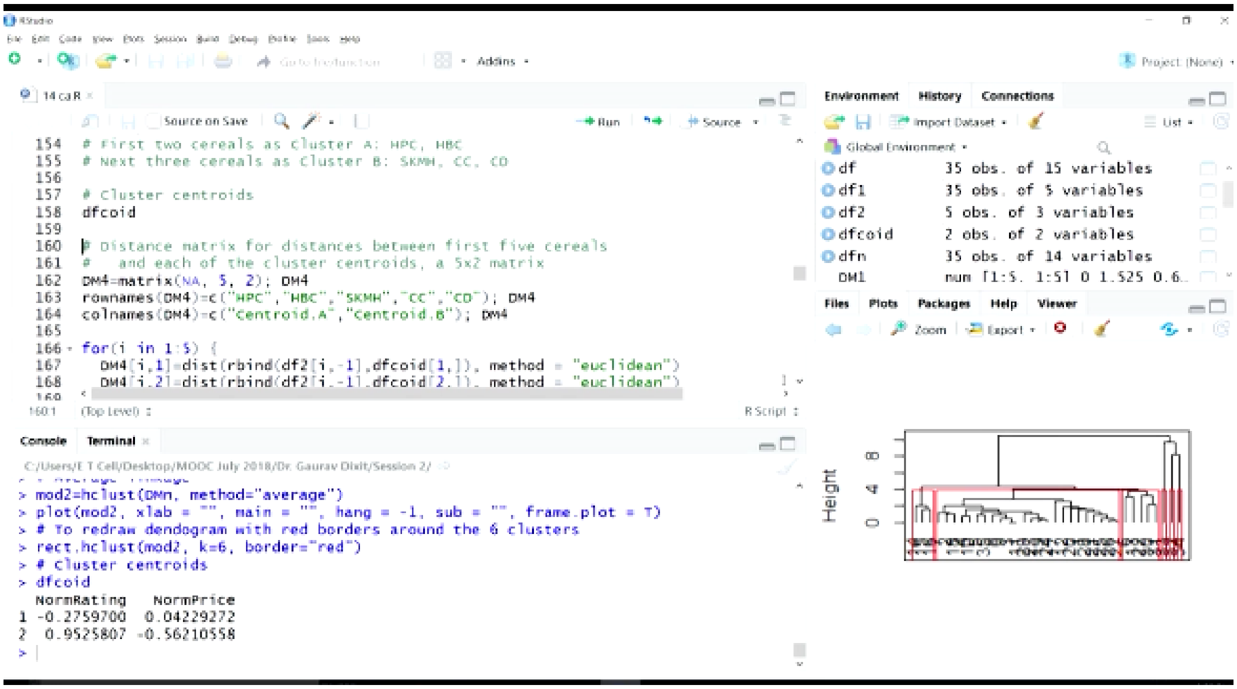
```
mod2=hclust(DMn, method="average")

plot(mod2, xlab = "", main = "", hang = -1, sub = "", frame.plot = T)
# To redraw dendogram with red borders around the 6 clusters
rect.hclust(mod2, k=6, border="red")

####
# k-means clustering
# Example:
# First two cereals as Cluster A: HPC, HBC
# Next three cereals as Cluster B: SKMH, CC, CD

# cluster centroids
dfcoid
```

```
mod1        List of 7
mod2        List of 7
Values
  DM        class 'dist' atomic [1:10] 1..
  DM2       class 'dist' atomic [1:1] 1..
  DMn       class 'dist' atomic [1:595] ...
```

C:/Users/E T Cell/Desktop/MOOC July 2018/Dr. Gaurav Dixit/Session 2/

```
> groups
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 4 5 6 1
> # To redraw dendogram with red borders around the 6 clusters
> rect.hclust(mod1, k=6, border="red")
> # Average linkage
> mod2=hclust(DMn, method="average")
> plot(mod2, xlab = "", main = "", hang = -1, sub = "", frame.plot = T)
> # To redraw dendogram with red borders around the 6 clusters
> rect.hclust(mod2, k=6, border="red")
> 
```

So let's go back to R studio, and we'll go through an exercise in R to understand, this K means clustering process in more detail. So as you can see here we are going to start our this part of cluster analysis, K means clustering, so again let's reconsider the dataset, so this dataset of you know breakfast cereals was about 35 observations that we had, in total we can recheck this from the environment section, so we had 35 observations and 14 variables, so we can see this DFN 35 observation and 14 variables, so within this if we just consider the first five cereals like we did in a HAC, so in this example we are going to consider just you know first 5 of these 35 observations, and we'll also pick just you know, we'll also consider just two variables rating and price, so let's consider that first two cereals are part of one cluster, that is cluster A, HPC and HBC, a similar exercise like we did in HAC, so next three cereals, let's assume their part of cluster B, so these cereals are SKMH-CC and CD, so centroids for these two clusters we have computed before as well, so let's get that, let's see that data again, so if we run this, so these are the you know centroids, these are the cluster centroid, one for cluster A, so row 1 is for cluster A, centroid for cluster A, row 2 is centroid for cluster B, we can see just two variables are being considered, rating and price and that too normalized scale.

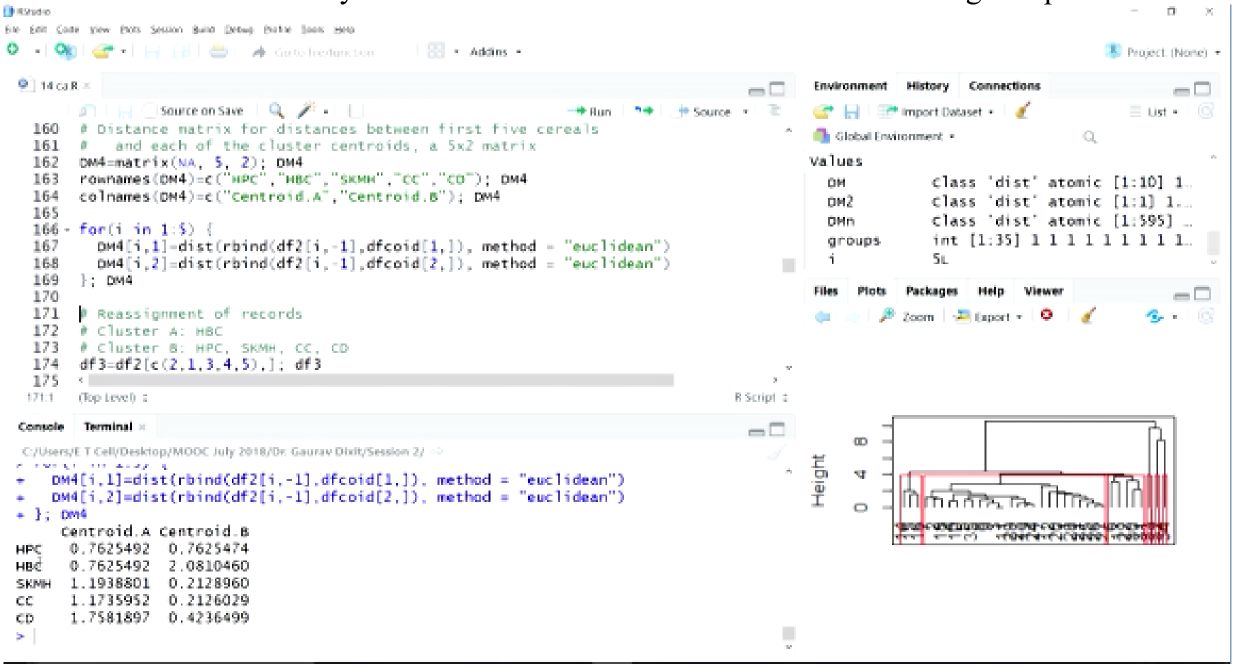


```
# First two cereals as Cluster A: HPC, HBC
# Next three cereals as Cluster B: SKMH, CC, CD

# Cluster centroids
dfcoid

# Distance matrix for distances between first five cereals
#   and each of the cluster centroids, a 5x2 matrix
DM4=matrix(NA, 5, 2); DM4
rownames(DM4)=c("HPC","HBC","SKMH","CC","CD"); DM4
colnames(DM4)=c("Centroid.A","Centroid.B"); DM4

for(i in 1:5) {
  DM4[i,1]=dist(rbind(df2[i,-1],dfcoid[1,]), method = "euclidean")
  DM4[i,2]=dist(rbind(df2[i,-1],dfcoid[2,]), method = "euclidean")
```

```
df       35 obs. of 15 variables
df1      35 obs. of 5 variables
df2      5 obs. of 3 variables
dfcoid   2 obs. of 2 variables
dfn      35 obs. of 14 variables
DM1      num [1:5, 1:5] 0 1.525 0.6...
```

```
> mod2=hclust(DMn, method="average")
> plot(mod2, xlab = "", main = "", hang = -1, sub = "", frame.plot = T)
> # To redraw dendogram with red borders around the 6 clusters
> rect.hclust(mod2, k=6, border="red")
> # Cluster centroids
> dfcoid
  NormRating   NormPrice
1 -0.2759700  0.04229272
2  0.9525807 -0.56210558
> 
```

Now as we talked about, we know the clusters centroid and we have these 5 observation, now we'll compute a distance metric you know which will have distances between these 5 observations and each of the cluster centroid, because we want to identify that each of these observation you know, and you know and their closeness with 2 clusters centroids that we have, so this will actually give us a 5 x 2 metrics, so we will have to initialize this, so this is the 5 x 2 metrics that we are going to initialize, the names are this, so names are this abbreviation of the cereals, so let's run this code. And then in the column side we have two centroids, centroid A and centroid B, so these values let's compute these distance values, so you can see we have a for loop here where we are using the dist function and in this we are using this DF2 data frame which has the normalized you know scale for our two variables that is rating and price.



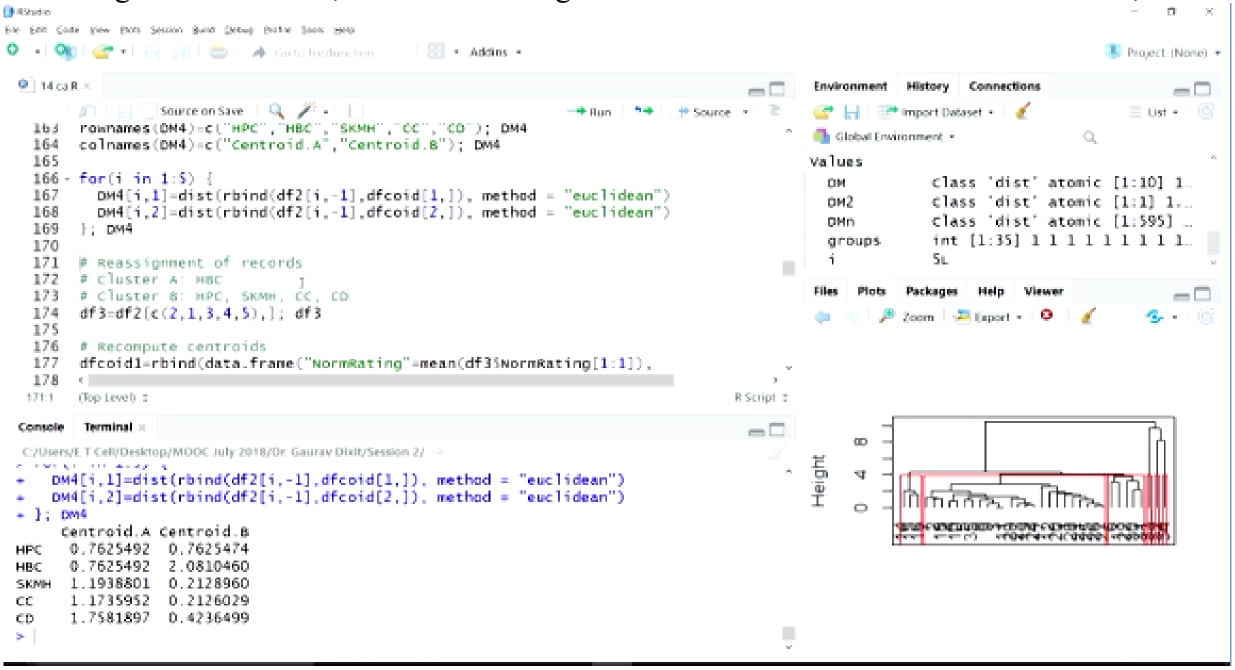
```
# Distance matrix for distances between first five cereals
#   and each of the cluster centroids, a 5x2 matrix
DM4=matrix(NA, 5, 2); DM4
rownames(DM4)=c("HPC","HBC","SKMH","CC","CD"); DM4
colnames(DM4)=c("Centroid.A","Centroid.B"); DM4

for(i in 1:5) {
  DM4[i,1]=dist(rbind(df2[i,-1],dfcoid[1,]), method = "euclidean")
  DM4[i,2]=dist(rbind(df2[i,-1],dfcoid[2,]), method = "euclidean")
}; DM4

# Reassignment of records
# Cluster A: HBC
# Cluster B: HPC, SKMH, CC, CD
df3=df2[c(2,1,3,4,5),]; df3
```

```
Values
  DM       class 'dist' atomic [1:10] 1...
  DM2      class 'dist' atomic [1:1] 1...
  DMn      class 'dist' atomic [1:595] ...
  groups   int [1:35] 1 1 1 1 1 1 1 1 1...
  i        5L
```

```
+   DM4[i,1]=dist(rbind(df2[i,-1],dfcoid[1,]), method = "euclidean")
+   DM4[i,2]=dist(rbind(df2[i,-1],dfcoid[2,]), method = "euclidean")
+ }; DM4
     Centroid.A Centroid.B
HPC   0.7625492  0.7625474
HBC   0.7625492  2.0810460
SKMH  1.1938801  0.2128960
CC    1.1735952  0.2126029
CD    1.7581897  0.4236499
> 
```

Now using the observations using the coordinates of these 5 observations and then the coordinates of centroids, we are trying to compute the distances between these points, so this for loop will give us these distances, so you can see method has been specified as Euclidean, so let's run this loop, so what we get in the output is, in the row side we have this 5 cereals, and for each of these cereals you know we can see the distances from centroid A and centroid B, so this can be clearly seen, so for example HPC its distance from centroid A is 0.7625492 and its distance from centroid B is 0.7625474 this is quite close, so this particular observation is you know, in a way equidistance, equidistant from both the clusters centroids A and B.

If we look at the second observation HBC, so this is closer to cluster A and also part of cluster A, so HPC let me clear this, HPC initially as it you know it was part of cluster A, however we can see from this data that it is in a way equidistance from both the centroids, but HBC it is quite clearly, it was part of cluster A and you know closer to cluster A as well, you know, it is quite far from you know centroid B, then third observation SKMH so clearly we can see this is closer to its own clusters centroid, CC is also closer to its own cluster centroid, CD is also closer to its own clusters centroid, so from this output we can see that last three observations, last three cereals are closer to their own cluster centroids, and HBC is, the second one is closer to its own cluster centroid, but if we look at the first observation though it seems to be equi you know distant from both the cluster centroids, but if we closely look at the you know these values up to 7 decimal points, we can see that HPC is much closer to centroid B, so therefore it can be reassigned to cluster B, so this is what is going to happen as per the algorithm that we'll find out for each observation, and you know its closeness with each of the cluster centroids, so we can see HPC has to be reassigned as per this algorithm, K means clustering, algorithm has to be reassigned to cluster B, so the same thing we have noted here in the comment section,

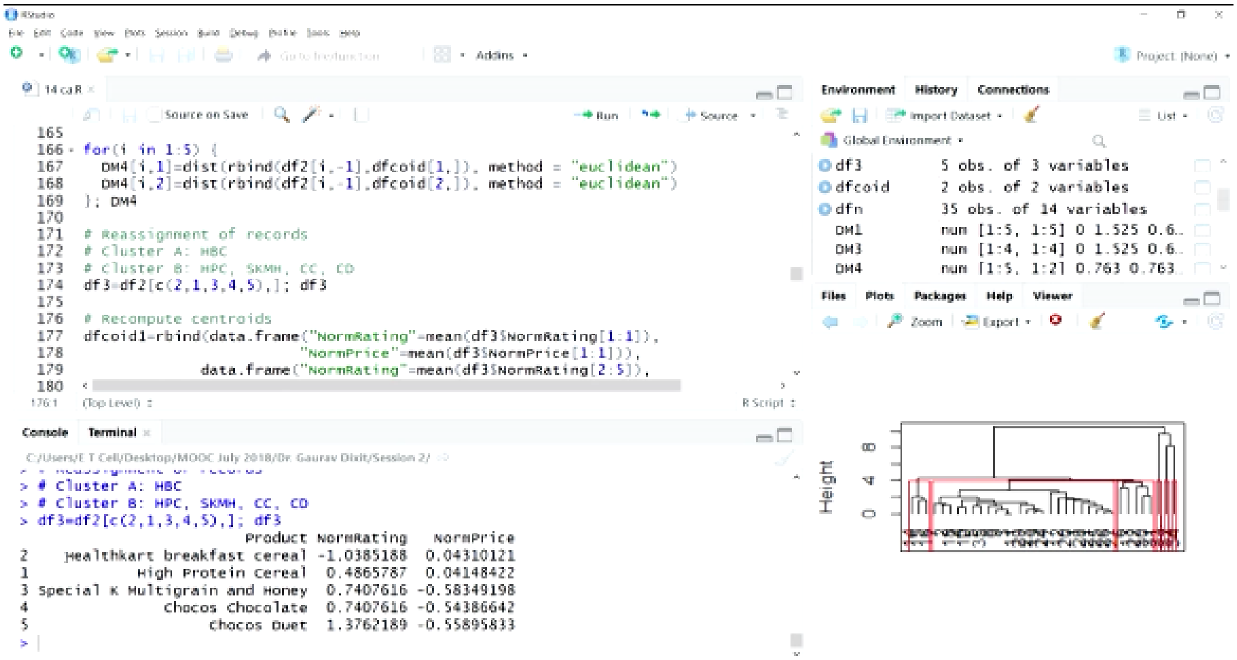


```
rownames(DM4)=c("HPC","HBC","SKMH","CC","CD"); DM4
colnames(DM4)=c("Centroid.A","Centroid.B"); DM4

for(i in 1:5) {
  DM4[i,1]=dist(rbind(df2[i,-1],dfcoid[1,]), method = "euclidean")
  DM4[i,2]=dist(rbind(df2[i,-1],dfcoid[2,]), method = "euclidean")
}; DM4

# Reassignment of records
# Cluster A: HBC
# Cluster B: HPC, SKMH, CC, CD
df3=df2[c(2,1,3,4,5),]; df3

# Recompute centroids
dfcoid1=rbind(data.frame("NormRating"=mean(df3$NormRating[1:1]),
```

```
+   DM4[i,1]=dist(rbind(df2[i,-1],dfcoid[1,]), method = "euclidean")
+   DM4[i,2]=dist(rbind(df2[i,-1],dfcoid[2,]), method = "euclidean")
+ }; DM4
     Centroid.A Centroid.B
HPC   0.7625492  0.7625474
HBC   0.7625492  2.0810460
SKMH  1.1938801  0.2128960
CC    1.1735952  0.2126029
CD    1.7581897  0.4236499
> 
```

```
Values
  DM        Class 'dist'  atomic [1:10] 1…
  DM2       Class 'dist'  atomic [1:1] 1…
  DMn       Class 'dist'  atomic [1:595] …
  groups    int [1:35] 1 1 1 1 1 1 1 1 1…
  i         5L
```

reassignment of records, now cluster A would be left with just one record that is HBC, and cluster B will gain a record that is HPC, so earlier it had 3 records now it will have one more, so in total it will have you know 4 observations in total, and the cluster A would be left with just one observations, so cluster A would be a singleton, cluster, and cluster B would have 4 observations now.



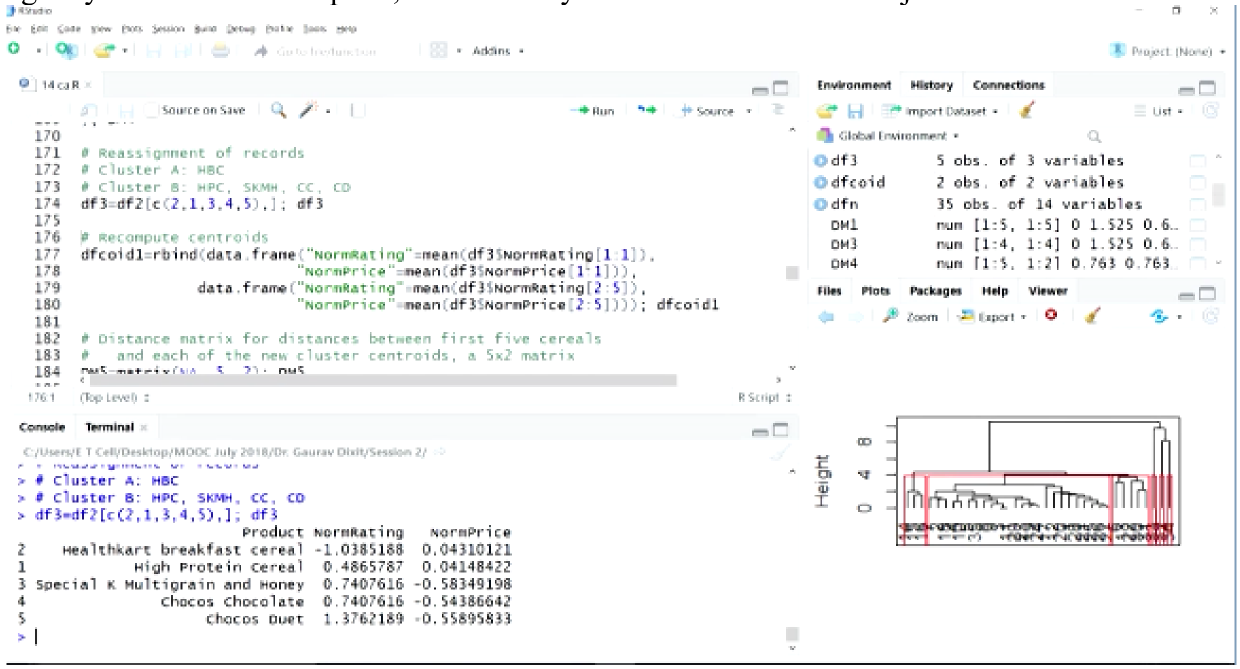
```
for(i in 1:5) {
  DM4[i,1]=dist(rbind(df2[i,-1],dfcoid[1,]), method = "euclidean")
  DM4[i,2]=dist(rbind(df2[i,-1],dfcoid[2,]), method = "euclidean")
}; DM4

# Reassignment of records
# Cluster A: HBC
# Cluster B: HPC, SKMH, CC, CD
df3=df2[c(2,1,3,4,5),]; df3

# Recompute centroids
dfcoid1=rbind(data.frame("NormRating"=mean(df3$NormRating[1:1]),
                         "NormPrice"=mean(df3$NormPrice[1:1])),
              data.frame("NormRating"=mean(df3$NormRating[2:5]),
```

```
> # Cluster A: HBC
> # Cluster B: HPC, SKMH, CC, CD
> df3=df2[c(2,1,3,4,5),]; df3
                         Product NormRating   NormPrice
2    Healthkart breakfast cereal -1.0385188  0.04310121
1           High Protein Cereal  0.4865787  0.04148422
3 Special K Multigrain and Honey  0.7407616 -0.58349198
4               Chocos Chocolate  0.7407616 -0.54386642
5                    Chocos Duet  1.3762189 -0.55895833
> 
```

```
df3      5 obs. of 3 variables
dfcoid   2 obs. of 2 variables
dfn      35 obs. of 14 variables
DM1      num [1:5, 1:5] 0 1.525 0.6...
DM3      num [1:4, 1:4] 0 1.525 0.6...
DM4      num [1:5, 1:2] 0.763 0.763...
```

So now let's reconfigure our you know dataset for these 5 observations, so now here in this data frame we can see, we have done some reordering of observation, so now the observation belonging to, see the only observation belonging to cluster A that is HBC comes first, and the remaining 4 observation belonging to cluster B, so just one observation belonging to cluster A and the remaining 4 observation belonging to cluster B.

Now what we can do is we can again re-compute the cluster centroid, so this is as per the algorithm that we have discussed, we are going to re-compute the new cluster centroid, so here again you can see in the quote, for the first you know cluster we have just one observation so

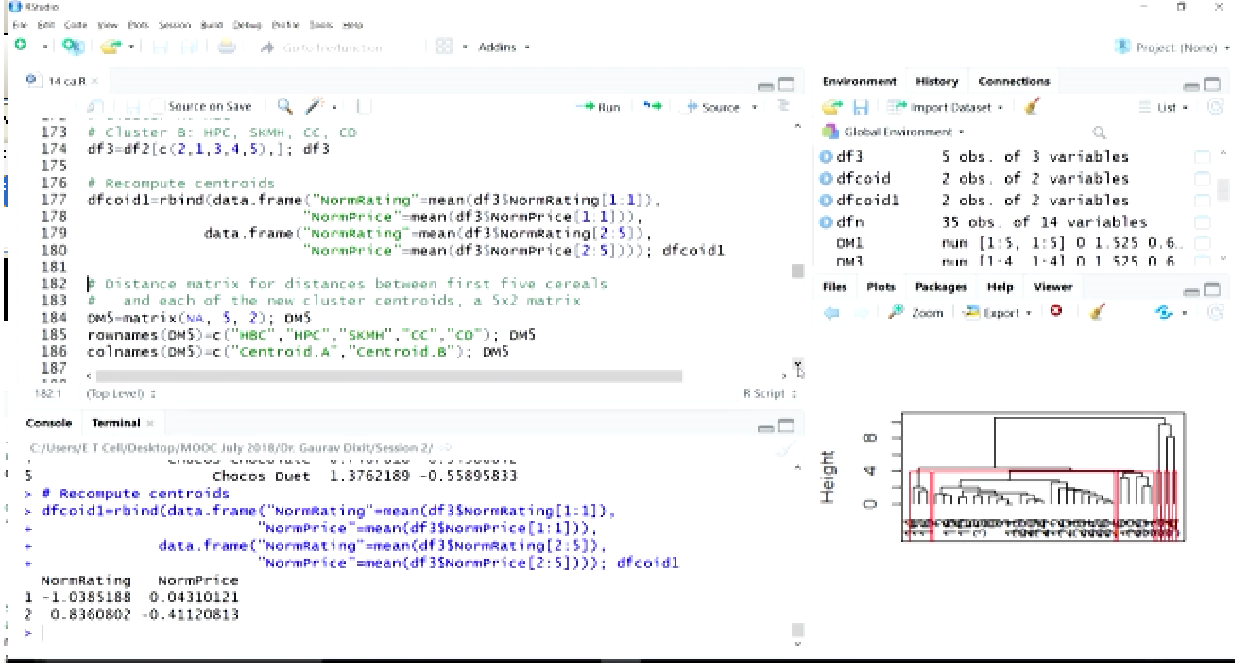


```
# Reassignment of records
# Cluster A: HBC
# Cluster B: HPC, SKMH, CC, CD
df3=df2[c(2,1,3,4,5),]; df3

# Recompute centroids
dfcoid1=rbind(data.frame("NormRating"=mean(df3$NormRating[1:1]),
                         "NormPrice"=mean(df3$NormPrice[1:1])),
              data.frame("NormRating"=mean(df3$NormRating[2:5]),
                         "NormPrice"=mean(df3$NormPrice[2:5]))); dfcoid1

# Distance matrix for distances between first five cereals
#   and each of the new cluster centroids, a 5x2 matrix
DM5=matrix(NA, 5, 2); DM5
```

```
> # Cluster A: HBC
> # Cluster B: HPC, SKMH, CC, CD
> df3=df2[c(2,1,3,4,5),]; df3
                         Product NormRating   NormPrice
2    Healthkart breakfast cereal -1.0385188  0.04310121
1           High Protein Cereal  0.4865787  0.04148422
3 Special K Multigrain and Honey  0.7407616 -0.58349198
4               Chocos Chocolate  0.7407616 -0.54386642
5                    Chocos Duet  1.3762189 -0.55895833
> |
```

```
df3      5 obs. of 3 variables
dfcoid   2 obs. of 2 variables
dfn      35 obs. of 14 variables
DM1      num [1:5, 1:5] 0 1.525 0.6...
DM3      num [1:4, 1:4] 0 1.525 0.6...
DM4      num [1:5, 1:2] 0.763 0.763...
```

the mean values that we have to compute for each of the variables, just one you know, one observation so therefore the mean value is also going to be the value of that observation, so the centroid for cluster A is going to be that single observation itself, but centroid for cluster B is now going to be slightly different, so in this we can see the second one 2 to 5 here, the centroid for second cluster, cluster B four observations are involved, and the mean values are being computed for each of these variables, so let’s run this code. Now we can see the centroids for each of these clusters, cluster A and cluster B, now with this you know, now we have the new cluster centroid, now what we can do is again we can go back to step 2 that we talked about in the algorithm, now again we can compute the distance metric which will have distances between these 5 observation and each of the new cluster centroids, so now again we are going to compute a 5 x 2 metrics which will have all the distances that we need, so let’s initialize this.

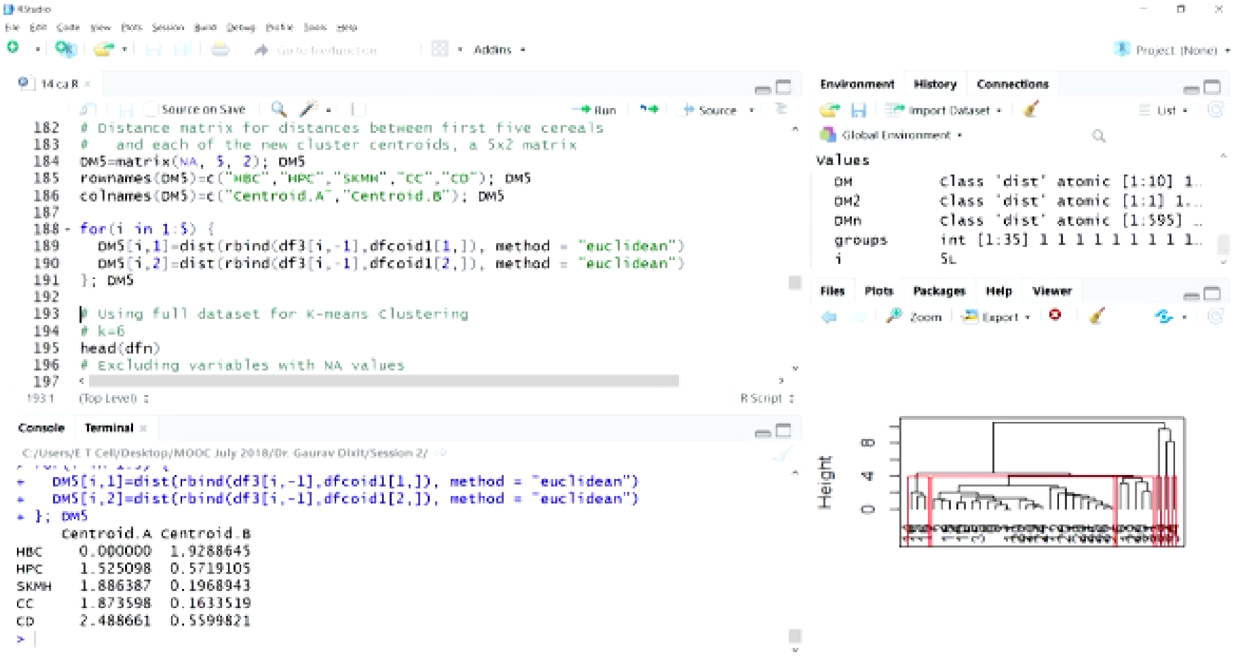


```
# Cluster B: HPC, SKMH, CC, CD
df3=df2[c(2,1,3,4,5),]; df3

# Recompute centroids
dfcoid1=rbind(data.frame("NormRating"=mean(df3$NormRating[1:1]),
                         "NormPrice"=mean(df3$NormPrice[1:1])),
              data.frame("NormRating"=mean(df3$NormRating[2:5]),
                         "NormPrice"=mean(df3$NormPrice[2:5]))); dfcoid1

# Distance matrix for distances between first five cereals
#   and each of the new cluster centroids, a 5x2 matrix
DM5=matrix(NA, 5, 2); DM5
rownames(DM5)=c("HBC","HPC","SKMH","CC","CD"); DM5
colnames(DM5)=c("Centroid.A","Centroid.B"); DM5
```

```
5                  Chocos Duet  1.3762189 -0.55895833
> # Recompute centroids
> dfcoid1=rbind(data.frame("NormRating"=mean(df3$NormRating[1:1]),
+                          "NormPrice"=mean(df3$NormPrice[1:1])),
+               data.frame("NormRating"=mean(df3$NormRating[2:5]),
+                          "NormPrice"=mean(df3$NormPrice[2:5]))); dfcoid1
  NormRating   NormPrice
1 -1.0385188  0.04310121
2  0.8360802 -0.41120813
> 
```

```
df3        5 obs. of 3 variables
dfcoid     2 obs. of 2 variables
dfcoid1    2 obs. of 2 variables
dfn        35 obs. of 14 variables
DM1        num [1:5, 1:5] 0 1.525 0.6...
```

Row names you can see, there is some change, right, first observation belonging to cluster A then followed by 4 observations belonging to cluster B, so let’s change the row names, and the column names, centroid A and B, now we have this far loop which is going to compute for us the Euclidean distances between these 5 observation and these new centroids, so let’s run this code.

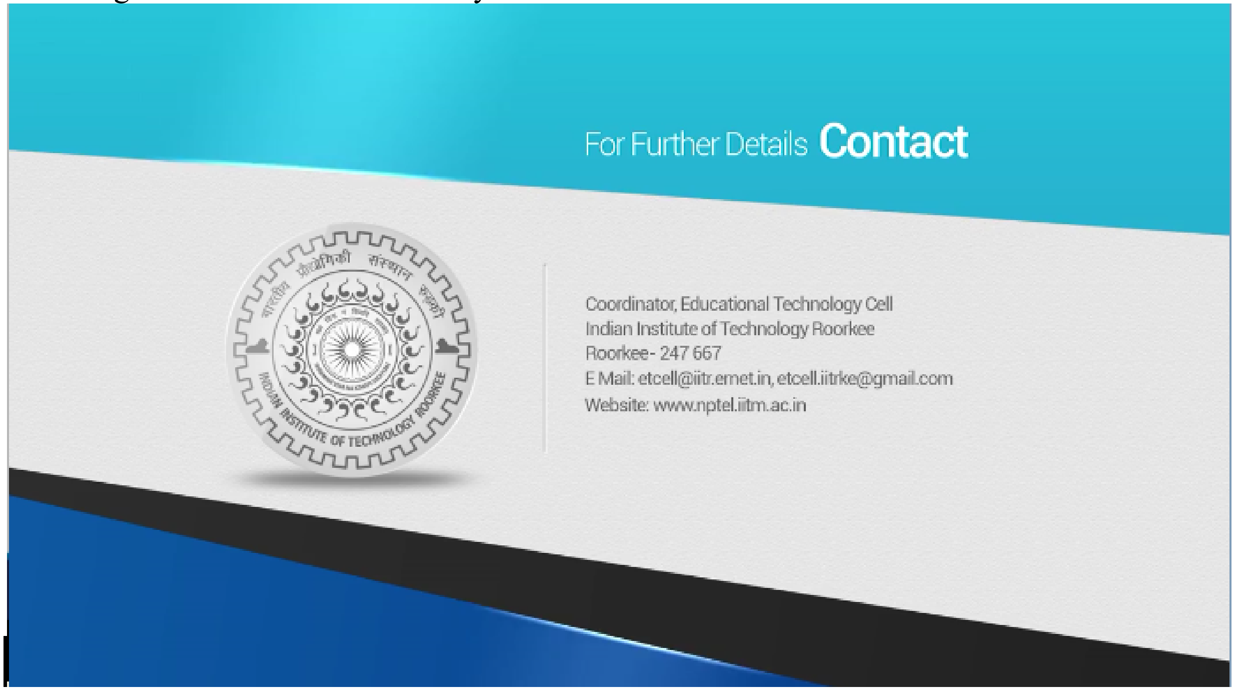


```
# Distance matrix for distances between first five cereals
#   and each of the new cluster centroids, a 5x2 matrix
DM5=matrix(NA, 5, 2); DM5
rownames(DM5)=c("HBC","HPC","SKMH","CC","CD"); DM5
colnames(DM5)=c("Centroid.A","Centroid.B"); DM5

for(i in 1:5) {
  DM5[i,1]=dist(rbind(df3[i,-1],dfcoid1[1,]), method = "euclidean")
  DM5[i,2]=dist(rbind(df3[i,-1],dfcoid1[2,]), method = "euclidean")
}; DM5

# Using full dataset for K-means Clustering
# k=6
head(dfn)
# Excluding variables with NA values
```

```
+   DM5[i,1]=dist(rbind(df3[i,-1],dfcoid1[1,]), method = "euclidean")
+   DM5[i,2]=dist(rbind(df3[i,-1],dfcoid1[2,]), method = "euclidean")
+ }; DM5
     Centroid.A Centroid.B
HBC    0.000000  1.9288645
HPC    1.525098  0.5719105
SKMH   1.886387  0.1968943
CC     1.873598  0.1633519
CD     2.488661  0.5599821
>
```

```
Values
  DM        class 'dist' atomic [1:10] 1.
  DM2       class 'dist' atomic [1:1] 1...
  DMn       class 'dist' atomic [1:595] ...
  groups    int [1:35] 1 1 1 1 1 1 1 1 1...
  i         5L
```

So we can see from here that now HBC, the first observation this is the centroid you know its distance from centroid A is 0, because this observation itself has become the centroid for cluster A being just you know one observation in that cluster, now we can see HPC clearly its distance from its own cluster that is B which is smaller, for SKMH its distance from its own cluster is smaller, CC its distance from its own cluster is smaller, and similarly for the last one CD, so no further reassignment is required so this is the final cluster formation, so first observation HBC goes to cluster A, and the remaining 4 observation go to the cluster B, so this is the final formations, so we have gone through the, K means clustering using this particular small you know, just 5 observation, so we'll stop here, and we'll continue our discussion on K means clustering in the next lecture. Thank you.



For Further Details **Contact**

Coordinator, Educational Technology Cell
Indian Institute of Technology Roorkee
Roorkee- 247 667
E Mail: etcell@iitr.ernet.in, etcell.iitrke@gmail.com
Website: www.nptel.iitm.ac.in

INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
भारतीय प्रौद्योगिकी संस्थान रुड़की

For Further Details Contact
Coordinator Educational Technology Cell
Indian Institute of Technology Roorkee
Roorkee – 247 667
E Mail:-etcell@iitr.ernet.in, iitrke@gmail.com
Website: www.nptel.iitm.ac.in

Acknowledgement
Prof. Ajit Kumar Chaturvedi
Director, IIT Roorkee

NPTEL Coordinator
IIT Roorkee
Prof. B. K Gandhi

Subject Expert
Dr. Gaurav Dixit
Department of Management Studies
IIT Roorkee

Produced by
Mohan Raj.S

Graphics
Binoy V.P

Web Team
Dr. NibeditaBisoyi
Neetesh Kumar
Jitender Kumar
Vivek Kumar
Dharamveer Singh
Gaurav Kumar

An educational Technology cell
IIT Roorkee Production