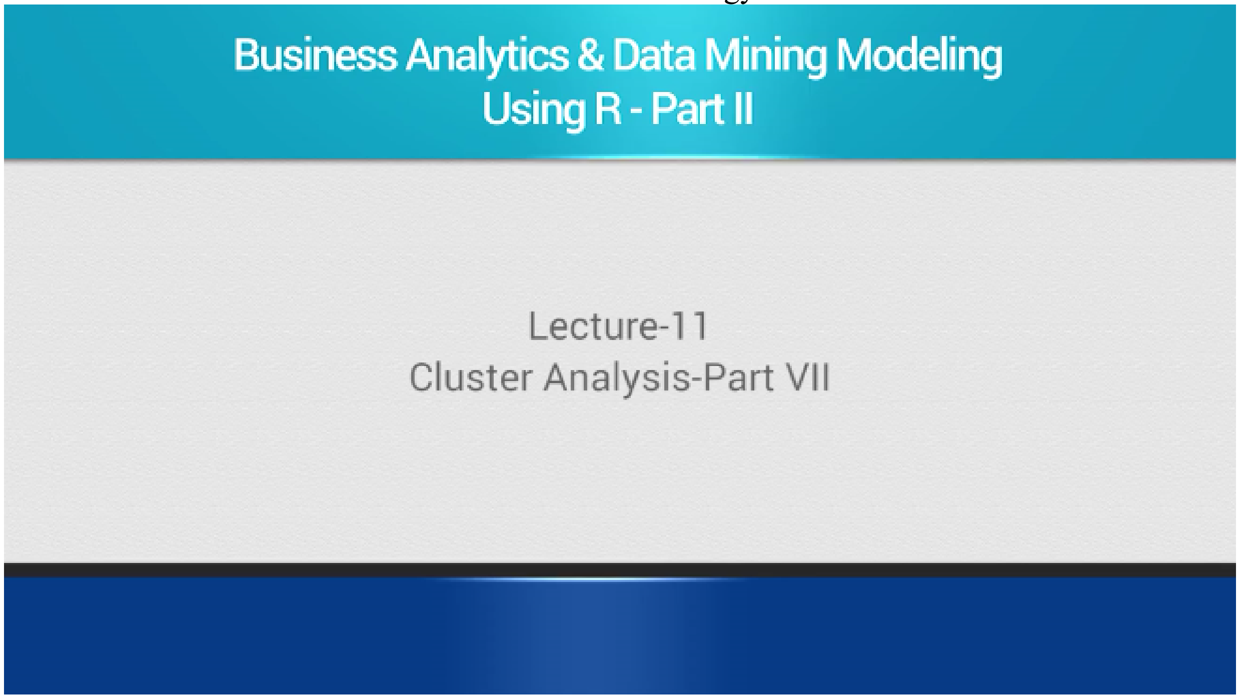
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
NPTEL
NPTEL ONLINE CERTIFICATION COURSE
Business Analytics & Data Mining Modeling
Using R – Part II
Lecture-11
ClusterAnalysis– Part VII
With
Dr. Gaurav Dixit
Department of Management Studies
Indian Institute of Technology Roorkee

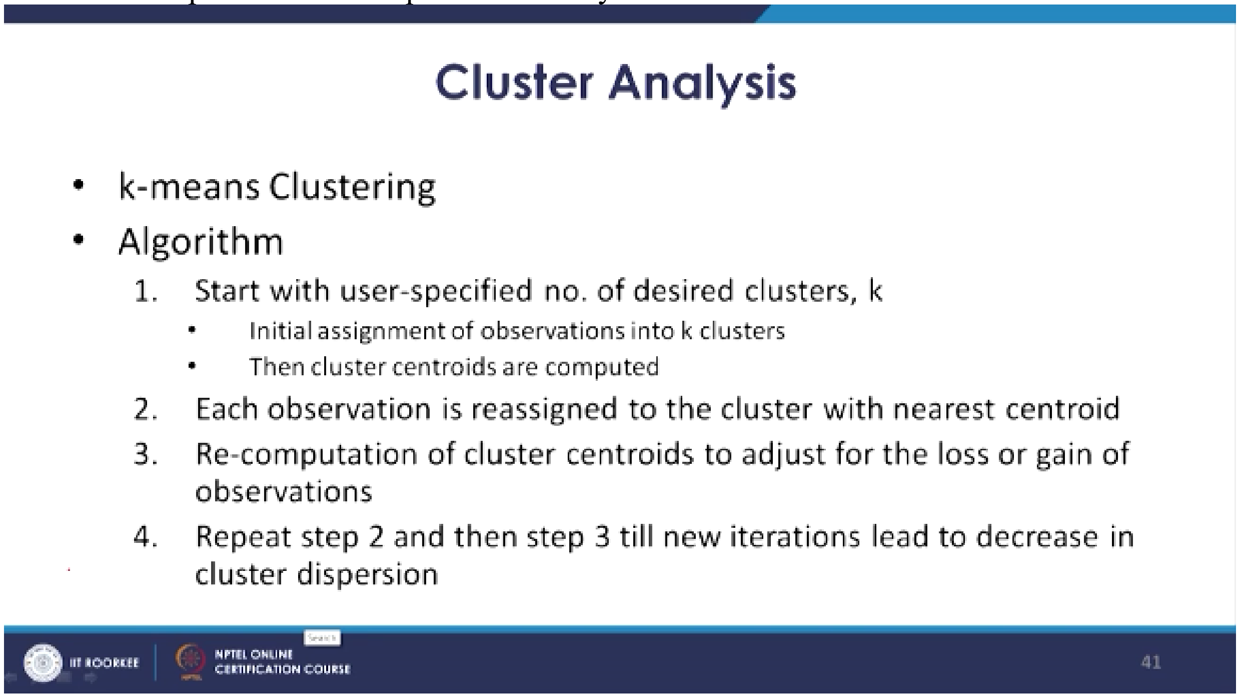


Business Analytics & Data Mining Modeling Using R - Part II

Lecture-11
Cluster Analysis-Part VII

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

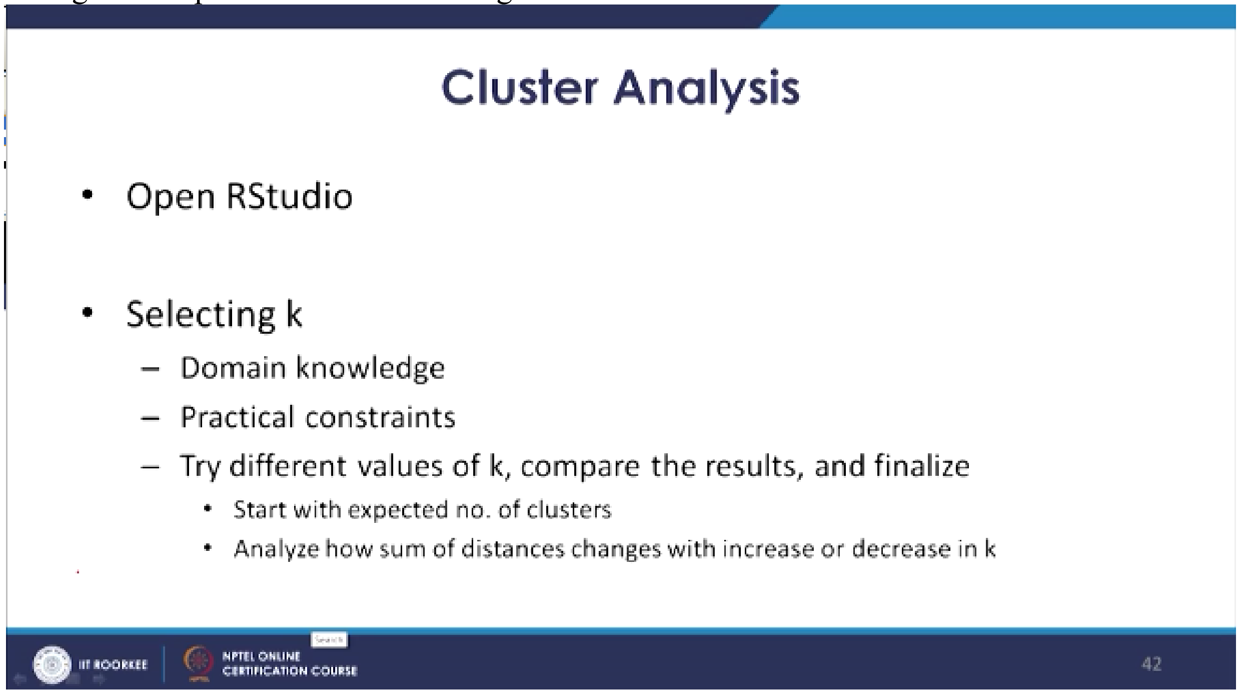
Welcome to the course Business Analytics and Data Mining Modeling Using R – Part 2, so in previous few lectures we have been discussing cluster analysis. So this is going to be the last lecture of this particular technique cluster analysis.



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 in previous lecture we talked about K means clustering, we understood the algorithm of this particular technique, we also did an exercise in R to understand the steps, so what we discussed about K means these, you know let's go through these steps quickly once again, so we typically start with user specified number of desired clusters that is K, then initial assignment of observations into these K clusters is done, then after that clusters centroids are computed and once this is done then in the next step for each observation is reassigned to the cluster that nearest centroid and as a result of this re-computation of cluster centroids to adjust for the loss or gain of observation that is done, and we keep on repeating these step 2 and then followed by you know, and then step 3 till there is decreased in cluster dispersion, so these are the main steps of K means clustering which we discussed in the previous lecture as well. Then in R also we talked about you know we did a small exercise, and discussed how K means clustering can be implemented, so we took first 5 cereals and then we started, we assumed 2 cluster you know first cluster with just, if you know first two cereals and the second cluster with the remaining three cereals, and we saw that when we compute the centroids and then when we start you know reassigning the observation into the closest centroid then what happened, one of the cluster was actually got reassigned and then when we repeated the exercise we saw that, that was the final configuration that all the you know observations they were into their closest you know, they were into the cluster with closest centroids. So we, in that exercise we have gone through the steps of K means clustering.



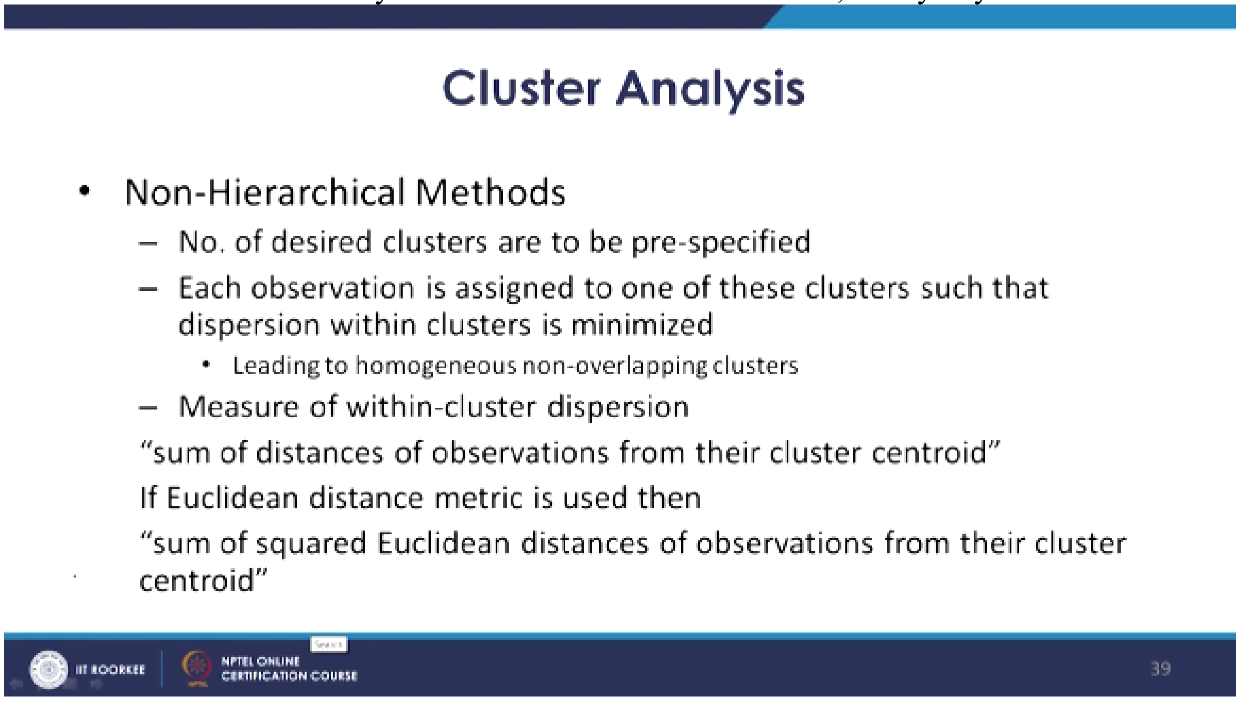
Cluster Analysis

- Open RStudio
- Selecting k
  - Domain knowledge
  - Practical constraints
  - Try different values of k, compare the results, and finalize
    - Start with expected no. of clusters
    - Analyze how sum of distances changes with increase or decrease in k

IIT ROORKEE
NPTEL ONLINE CERTIFICATION COURSE
42

Now let’s discuss few more important points about K means clustering, so from the steps of K means clustering as we can see, selecting suitable K for our clustering task is an important exercise which is part of this method, so let’s discuss a view point, how do we decide which is going to be the you know best K for our clustering analysis, cluster analysis, so selecting K few important points, for example domain knowledge, so sometimes domain knowledge will guide you in terms of the number of expected cluster, so as you know that in K means clustering, we need to pre-specify the number of clusters that we want and domain knowledge about the particular data set about the problem at hand can actually help us in terms of the clusters that we are looking for, for example if we talk about the customer segmentation so we can always you know understand that, in the kind of income groupings that we expect from a target population, because we will have some pre-existing knowledge about the target population like you know there is going to be a segment which is earning less than rupees 5,000 per month then there is going to be a segment which is earning between 5,000 and about 12,000, then you know 12,000 to you now 25,000 and in that fashion e can have our own expectation from the you know target

population, so that kind of domain knowledge can always be used you know when we apply, K means clustering.
Similarly the practical constraint can also sometimes guide or in a way restrict us in terms of selecting the number of clusters, so in many you know business applications sometimes it might be difficult for us to plan our you know if it is about the customer segmentation and that is to be used later on for the marketing and promotional schemes, then sometimes it might be difficult or costly for us to administratively or economically implement those schemes for a large number of clusters, so therefore we might you know like to have fewer number of clusters and then you know come up with the promotional and marketing schemes for them, and implement them, so therefore the practical constraint you know if they could be administrative in nature and financial in nature, they can also put a limit on the number of clusters that we can you know select for our cluster analysis, specifically K means clustering.
So beside these two you know, beside these two points the remaining approach is which could be the default approach you know more often they're not, that we can try different values of K and compare the results and finalize, so we can go about doing our clustering, so we might have still we have to start with you know a given number of clusters that are more likely to be there, and then we can start you know with those number of clusters and experiment, so we can you know try different values of K and then compare the results, and based on that the best possible number of you know will be able to pick the best possible value for K, so start with the expected number of clusters and analyze how sum of distances changes with increase or decrease in K, so as we talked about in the previous lecture also, so let's have a look at it again, so as we talked about that in you know non-hierarchical methods, we try to you know minimize



Cluster Analysis

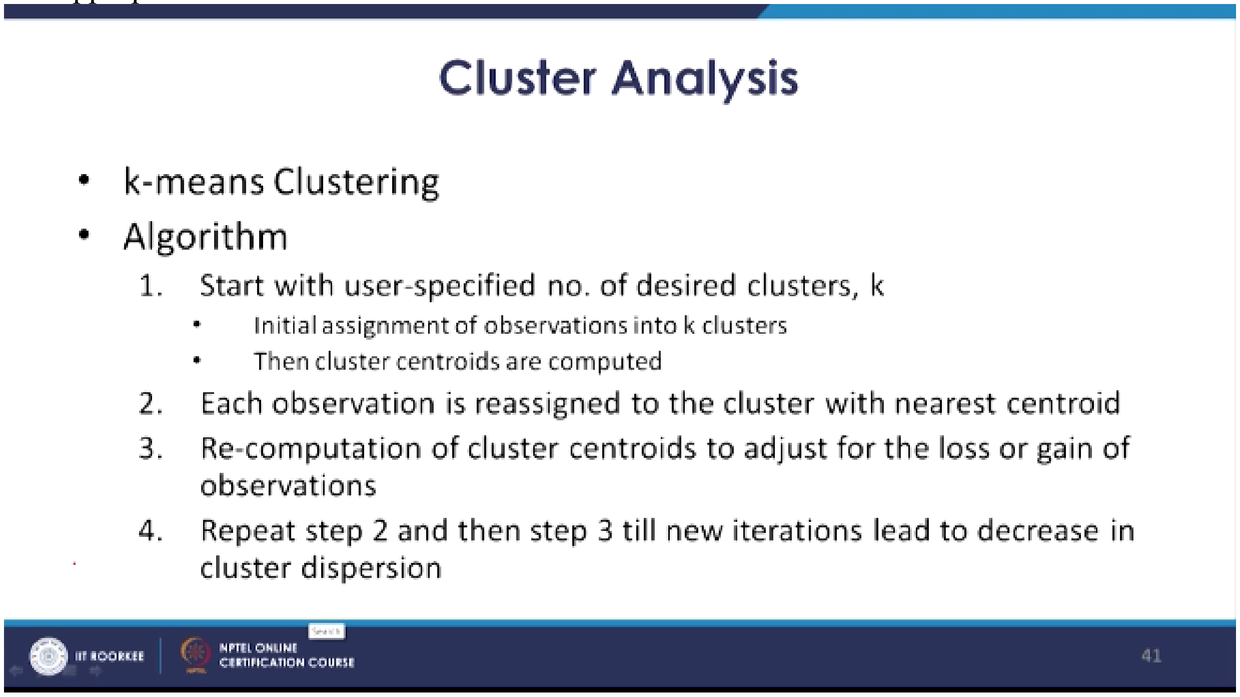
- Non-Hierarchical Methods
  - No. of desired clusters are to be pre-specified
  - Each observation is assigned to one of these clusters such that dispersion within clusters is minimized
    - Leading to homogeneous non-overlapping clusters
  - Measure of within-cluster dispersion

  "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"

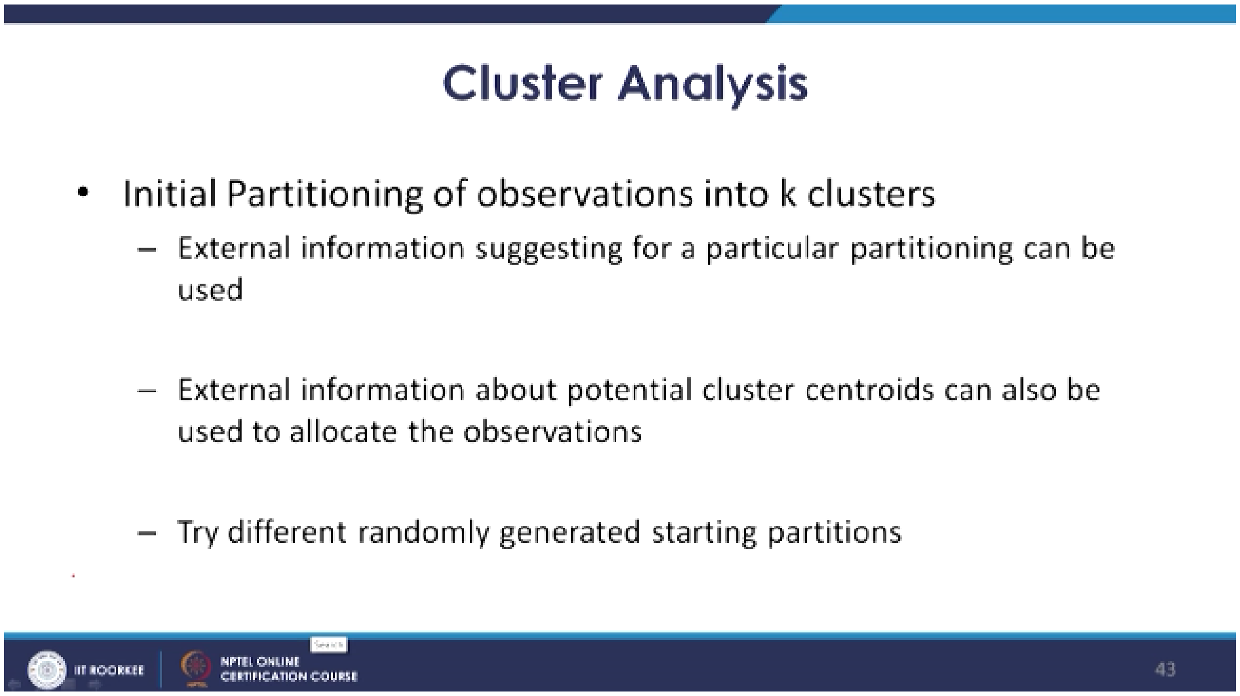
this within cluster dispersion, and this sum of distances, sum of squared distances of observation from their centroid, this is the measure that is strictly used, so keeping this particular measure in mind if we are trying out different values of K, then we can analyze for different values of K how this sum of distances is changing, because that is the measure, that is the you know metric that we use in K means clustering where we look to minimize the dispersion, so depending on, the results depending on this comparison you know the number of the particular K value for which this sum of distances is probably lower, probably we can go with that value of K, so these are you know different phase in which we can you know select and appropriate value for K.



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 few other important points about K means clustering, so let's go back to the algorithm, so there in the first point itself we get that, we need to decide a suitable value for K, and then you'll also see that something that is mentioned in the slide also that initial assignment of observation into clusters, how that should be done, so that can also play an important role in deciding your clustering results, so how initial partitioning of observations into you know a given number of clusters is to be done that also we need to take care off. So few points which could be helpful in this decision that we have mentioned here in the slide, so let's discuss them,



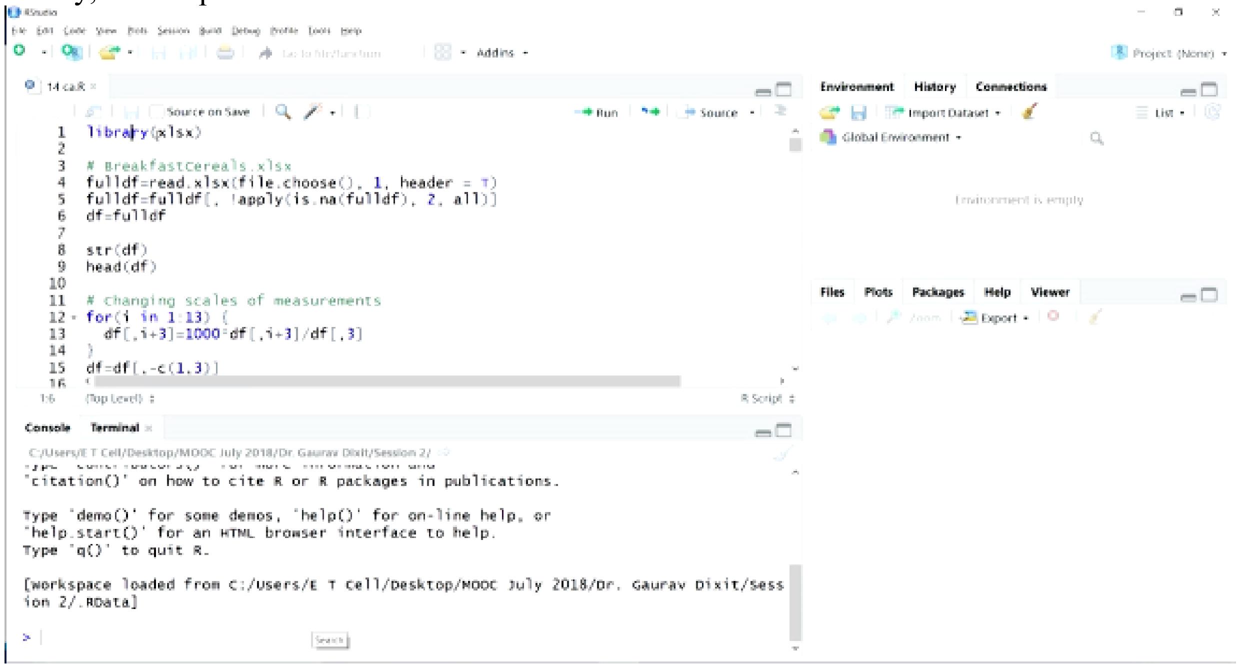
Cluster Analysis

- Initial Partitioning of observations into k clusters
  - External information suggesting for a particular partitioning can be used
  - External information about potential cluster centroids can also be used to allocate the observations
  - Try different randomly generated starting partitions

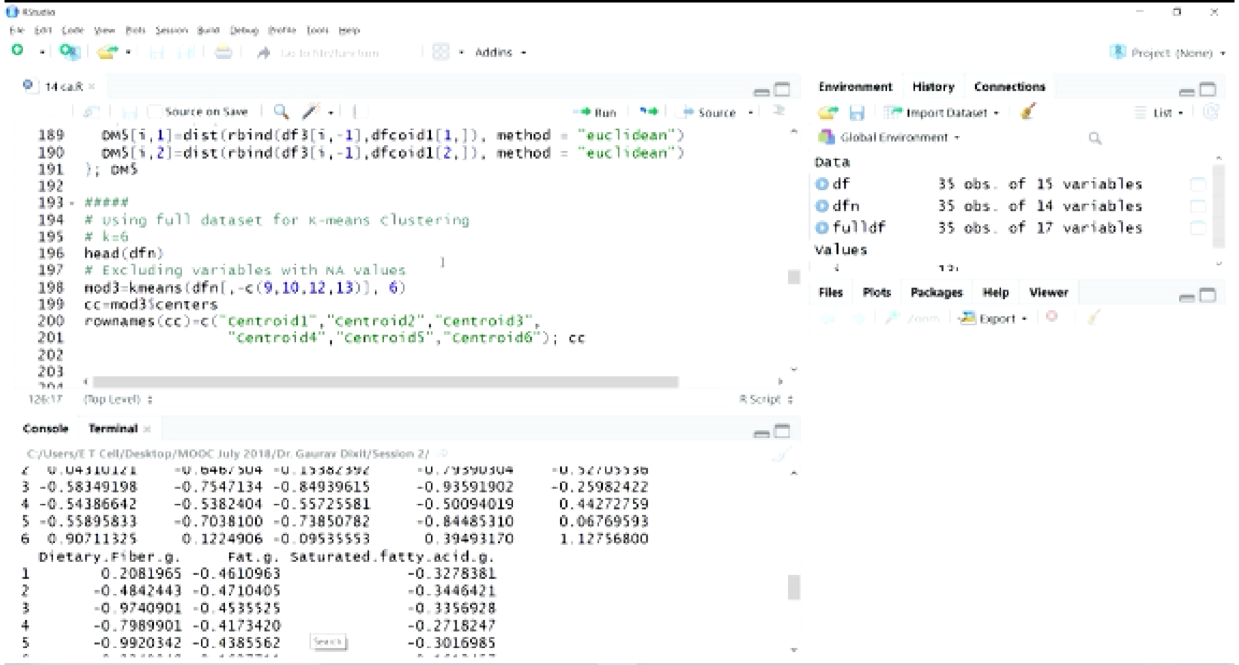
so first one is external information suggesting for a particular partitioning can be used, so if we have you know certain information that is not part of you know, that is not part of the dataset itself and it is suggesting that a particular kind of you know, particular observations you know, group of observation should be in you know in certain clusters should be in one group, so that kind of information can always be used, for example if we are doing customer segmentation and we know that you know a group of customers they are coming from a certain locality they are coming from a metro city or they might be coming from a TL2 or TL3 cities so that kind of geographical you know, geographical you know clustering that this, that information can also be used for to implement this kind of partitioning, so because this step where we are doing initial partitioning of observation to K clusters, can actually reduce our computation time, so if we are thinking about you know 100s and 1000s of records and applying clustering analysis, because as we know that we are into unsupervised learning you know method you know into that domain and there typically we need, we deal with a large number of observations, so therefore anything that can reduce the computation time for us that is always recommended and preferred.

So as I suggested for example customer segmentation you know we can incorporate the geographical testing if we feel that, that information is going to be you know important in our clustering process, in our clustering result then certainly that can be incorporated in the you know partitioning, initial partitioning itself and later on the algorithm will take care as the new centroids are computed and the distances of each observation from this new centroids, centroids are you know, are computed we can you know that reassignment on all those things are going to you know take care of the you know a good enough solution for us.

Now let’s move to the next point that is external information about potential cluster centroids can also be used to allocate the observations, so if we happen to know the dataset that we have the observations that we have, and you know the problem domain that is there, the context of the problem that is there we have some idea, we have some idea about where this centroids you know, the clusters and number of cluster and their hospital centroids are going to be, then we can use that details about those centroids at their you know potential you know coordinates, if

we have some idea about that then those, that information can also be plugged into you know initial partitioning scheme of K means clustering, and which could be a very good starting point, you know to reduce in terms of, you know reducing the computation time of this particular technique.
So external information about you know particular partitioning or centroids itself can always be used in the initial partitioning, if we don't have you know this kind of information to help us in initial partitioning, then of course we can you know that is the default approach where we can try different, randomly generated starting positions and then select which one is more suitable, because there could be various you know candidate models even based on same technique because you know different configurations could be there, specifically in the context of cluster analysis, so therefore it makes sense to try different randomly generated starting positions and then you know use the best one that is that the risk there for later analysis.
So let's move forward, so before we discuss further let's go back to R studio and do an exercise, so in previous lecture we did an exercise using just you know the first five cereals, and then we understood how the K means clustering can be applied and how, what kind of results we got, now what we are going to do is we'll apply the K means clustering on full dataset and we'll analyze the results later on, so let's go through few of the code, let's go through, run through few lines of code that we have to do for importing datasetanother things, so let's load this library, let's import the dataset.

```
library(xlsx)

# BreakfastCereals.xlsx
fulldf=read.xlsx(file.choose(), 1, header = T)
fulldf=fulldf[, !apply(is.na(fulldf), 2, all)]
df=fulldf

str(df)
head(df)

# changing scales of measurements
for(i in 1:13) {
  df[,i+3]=1000*df[,i+3]/df[,3]
}
df=df[,-c(1,3)]
```
```
C:/Users/E T Cell/Desktop/MOOC July 2018/Dr. Gaurav Dixit/Session 2/
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[workspace loaded from C:/Users/E T Cell/Desktop/MOOC July 2018/Dr. Gaurav Dixit/Sess
ion 2/.RData]

>
```
Let's take a new copy of this dataset like we've been doing in previous lectures as well, so some of the transformation steps are going to be common for K means clustering also, so we'll go through these transformations and then use it for applying K means clustering. So we need to find that part of code where we had done the scaling of full dataset, because now we are going to use the full dataset you can see here, normalization of all variables, so let's go ahead with this, so we'll normalize so we can see here, the variables that are part of this and first 6 observations, so this full data set these observations and variables are going to be part of our you know next K means clustering, so let's move ahead, so now we'll go back to the full dataset, part of code where we apply K means clustering, so this part we've discussed before K

means clustering on first 5 cereals, this part we have gone through in previous lecture, now here so we are now going to use full dataset for K means clustering, so here we are again using the



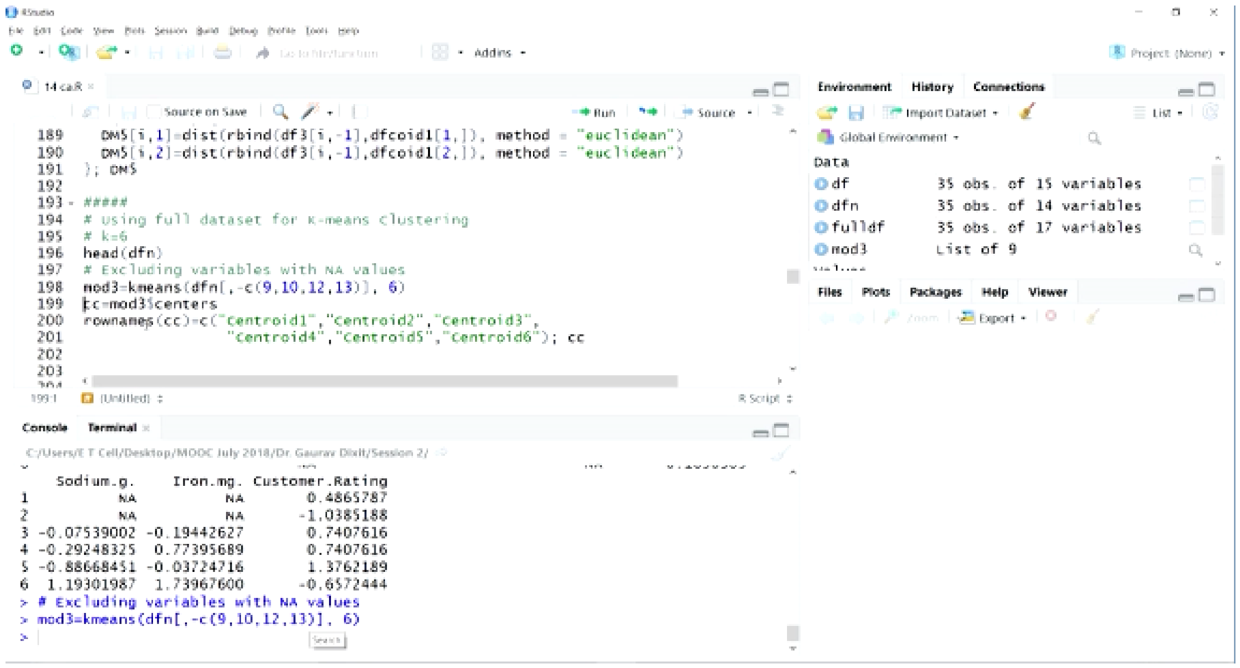
```r
  DMS[i,1]=dist(rbind(df3[i,-1],dfcoid1[1,]), method = "euclidean")
  DMS[i,2]=dist(rbind(df3[i,-1],dfcoid1[2,]), method = "euclidean")
}; DMS

#####
# using full dataset for K-means clustering
# k=6
head(dfn)
# Excluding variables with NA values
mod3=kmeans(dfn[,-c(9,10,12,13)], 6)
cc=mod3$centers
rownames(cc)=c("Centroid1","Centroid2","Centroid3",
               "Centroid4","Centroid5","Centroid6"); cc
```

```
Data
df        35 obs. of 15 variables
dfn       35 obs. of 14 variables
fulldf    35 obs. of 17 variables
```

```
2  0.04310121    -0.6467504 -0.15382392    -0.79590304    -0.52705536
3 -0.58349198    -0.7547134 -0.84939615    -0.93591902    -0.25982422
4 -0.54386642    -0.5382404 -0.55725581    -0.50094019     0.44272759
5 -0.55895833    -0.7038100 -0.73850782    -0.84485310     0.06769593
6  0.90711325     0.1224906 -0.09535553     0.39493170     1.12756800
  Dietary.Fiber.g.      Fat.g. Saturated.fatty.acid.g.
1        0.2081965 -0.4610963              -0.3278381
2       -0.4842443 -0.4710405              -0.3446421
3       -0.9740901 -0.4535525              -0.3356928
4       -0.7989901 -0.4173420              -0.2718247
5       -0.9920342 -0.4385562              -0.3016985
```

K, value of K as 6, so let's have a look at the first 6 observations, so these are the first 6 observation we can see the variables that are part of this.

Now as you can see few of these variables they have NA values so we'll get rid of these variables, so let's identify them so you can see here in the code I have written 9, 10, and 12, 13 so you can see 1, 2, 3, 4, 5, and then 6, 7, 8, 9 you can see some NA values even in the first 6 observation and 10 also, 10 and 11 seems to be fine then in 12 and 13 also, in the first 6 observation itself we are able to see some NA values so that in the code itself also the same is reflected, column numbers 9, 10, 12, 13 have been left out, and K means is applied, so K means is the function that we have used here, so and the second argument is where we specify the number of clusters that we want out of this clustering process, so let's go through this code.

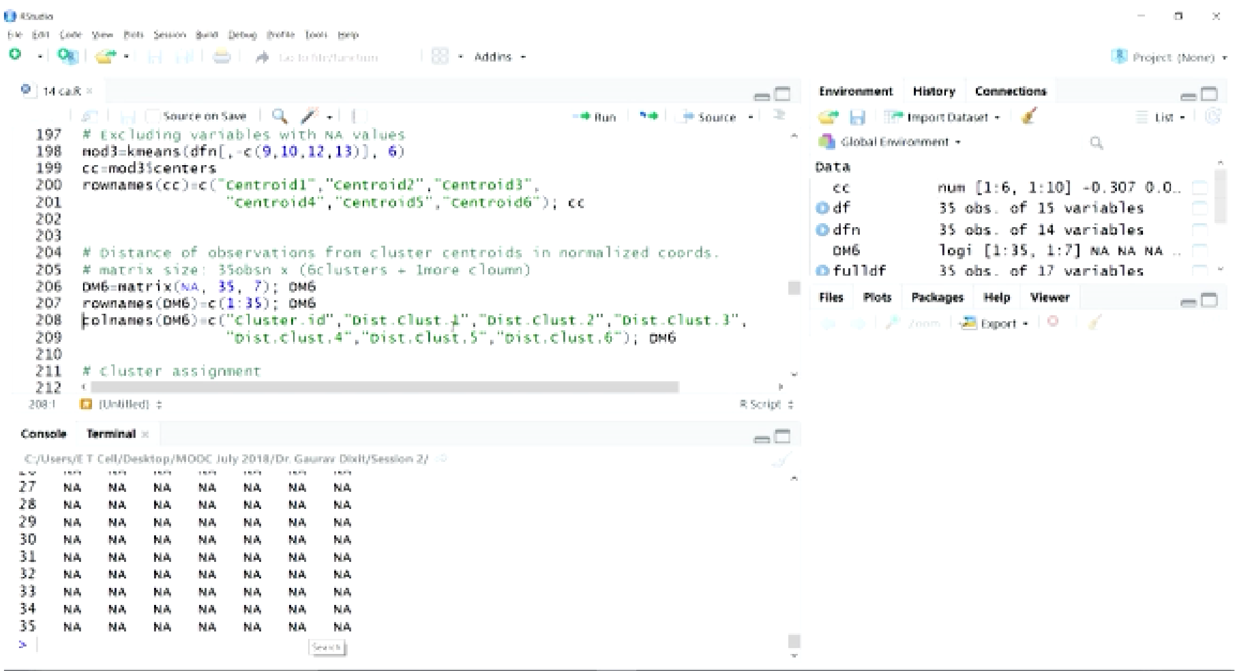


```r
  DM5[i,1]=dist(rbind(df3[i,-1],dfcoid1[1,]), method = "euclidean")
  DM5[i,2]=dist(rbind(df3[i,-1],dfcoid1[2,]), method = "euclidean")
}; DM5

#####
# Using full dataset for K-means clustering
# k=6
head(dfn)
# Excluding variables with NA values
mod3=kmeans(dfn[,-c(9,10,12,13)], 6)
cc=mod3$centers
rownames(cc)=c("Centroid1","Centroid2","Centroid3",
               "Centroid4","Centroid5","Centroid6"); cc
```

```
    Sodium.g.    Iron.mg. Customer.Rating
1          NA          NA       0.4865787
2          NA          NA      -1.0385188
3 -0.07539002 -0.19442627       0.7407616
4 -0.29248325  0.77395689       0.7407616
5 -0.88668451 -0.03724716       1.3762189
6  1.19301987  1.73967600      -0.6572444
> # Excluding variables with NA values
> mod3=kmeans(dfn[,-c(9,10,12,13)], 6)
>
```

Now the model has been K means clustering, model has been built, now let’s look at the clusters centroids so there are many attributes which are returned from this k means function, so we’ll access the important ones, so first one is where we get the cluster centroids, so this is the code and then what we are doing here in the next line row names we are changing the names of row names to the, to reflect the centroids this is done.

Now next thing that we are going to do is the distance of observations from cluster centroids in normalized coordinates, so as we know that there are 35 observations and you know 6 clusters, plus one more column that we are going to have that is for the cluster ID, so we’ll like to have you know one column where we know which observation is assigned to which clust cluster, so that cluster is going to be represented by cluster ID, so and then there are going to be you know 6 centroids, so since this matrix is going to have the distances between all the observation and each of the cluster centroid, so let’s initialize this 35 x 7 matrix, rownames are going to be 1 to 35 for the observations, and then column names as you can see first one is cluster ID, and then we have the you know distance from you know these clusters which is actually distance from those clusters centroids, so let’s change this.

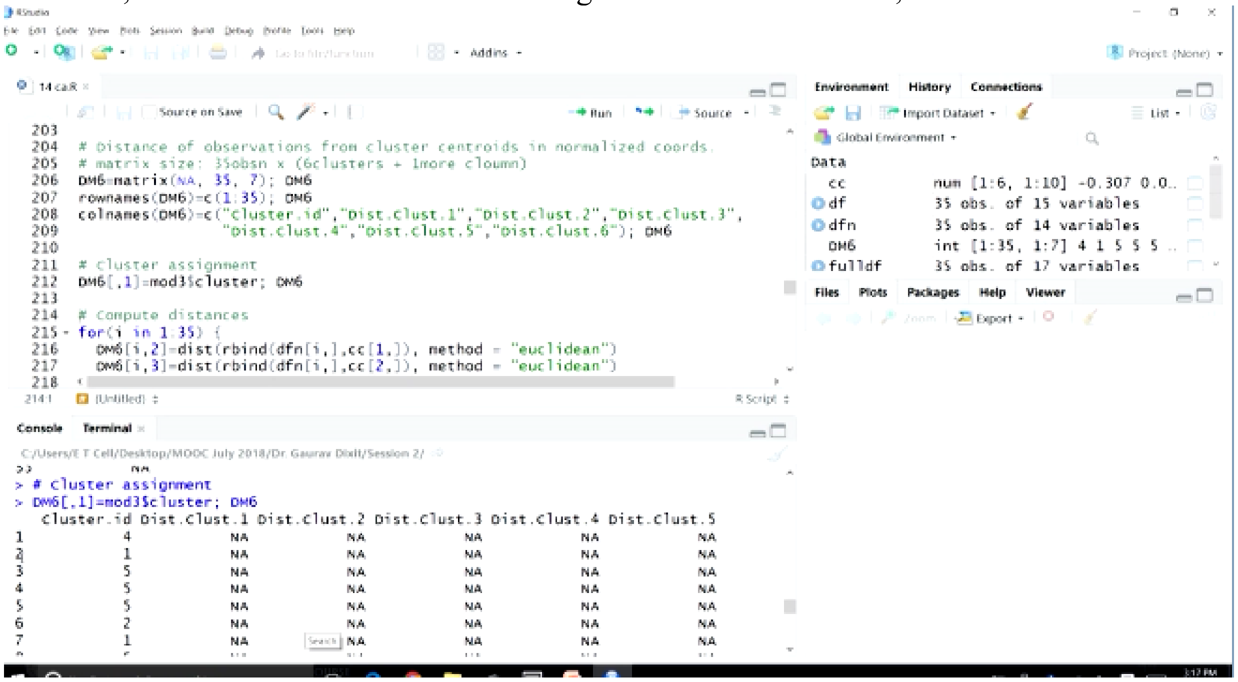


```
# Excluding variables with NA values
mod3=kmeans(dfn[,-c(9,10,12,13)], 6)
cc=mod3$centers
rownames(cc)=c("Centroid1","Centroid2","Centroid3",
               "Centroid4","Centroid5","Centroid6"); cc


# Distance of observations from cluster centroids in normalized coords.
# matrix size: 35obsn x (6clusters + 1more cloumn)
DM6=matrix(NA, 35, 7); DM6
rownames(DM6)=c(1:35); DM6
colnames(DM6)=c("Cluster.id","Dist.Clust.1","Dist.Clust.2","Dist.Clust.3",
               "Dist.Clust.4","Dist.Clust.5","Dist.Clust.6"); DM6

# Cluster assignment
```

```
27   NA   NA   NA   NA   NA   NA   NA
28   NA   NA   NA   NA   NA   NA   NA
29   NA   NA   NA   NA   NA   NA   NA
30   NA   NA   NA   NA   NA   NA   NA
31   NA   NA   NA   NA   NA   NA   NA
32   NA   NA   NA   NA   NA   NA   NA
33   NA   NA   NA   NA   NA   NA   NA
34   NA   NA   NA   NA   NA   NA   NA
35   NA   NA   NA   NA   NA   NA   NA
>
```

```
cc        num [1:6, 1:10] -0.307 0.0...
df        35 obs. of 15 variables
dfn       35 obs. of 14 variables
DM6       logi [1:35, 1:7] NA NA NA ...
fulldf    35 obs. of 17 variables
```

Then cluster assignment, so as part of the model that we have just built in this particular attribute mode 3 dollar cluster, so in the cluster, the cluster assignments have been given so let's have a look at this result, so you can see here in the output that the first column has been filled with data, and first observation has been assigned to cluster number 4, second observation

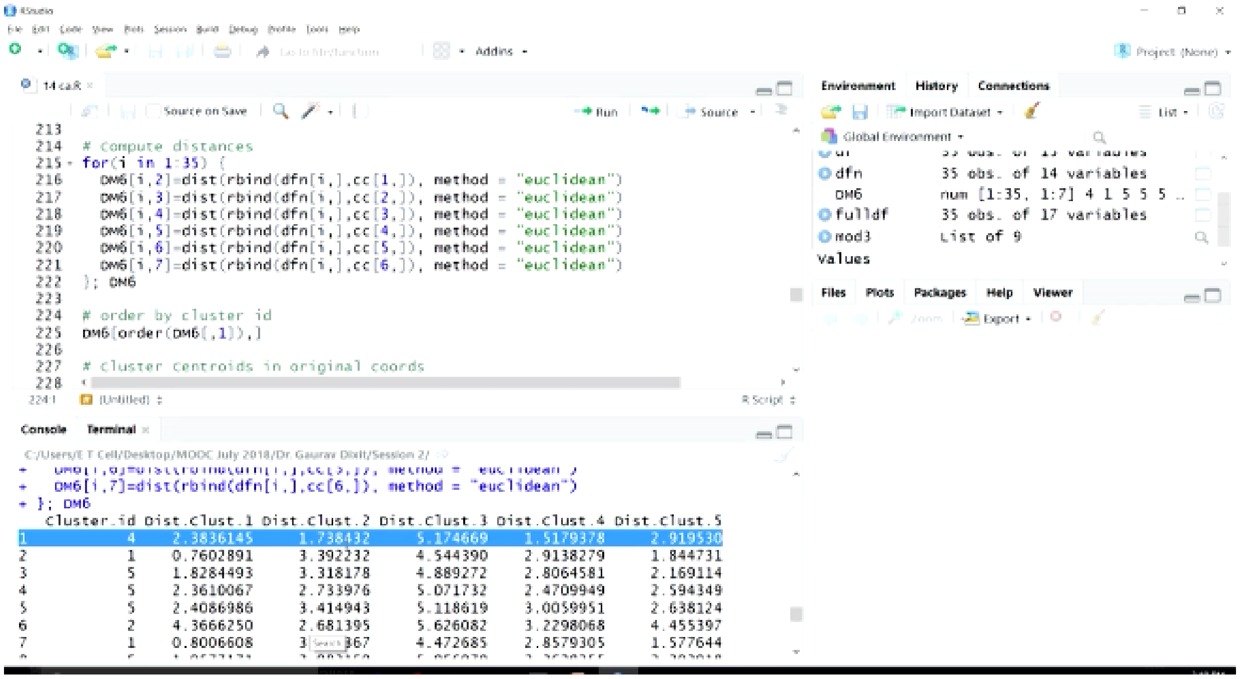


```
# Distance of observations from cluster centroids in normalized coords.
# matrix size: 35obsn x (6clusters + 1more cloumn)
DM6=matrix(NA, 35, 7); DM6
rownames(DM6)=c(1:35); DM6
colnames(DM6)=c("Cluster.id","Dist.Clust.1","Dist.Clust.2","Dist.Clust.3",
               "Dist.Clust.4","Dist.Clust.5","Dist.Clust.6"); DM6

# Cluster assignment
DM6[,1]=mod3$cluster; DM6

# Compute distances
for(i in 1:35) {
  DM6[i,2]=dist(rbind(dfn[i,],cc[1,]), method = "euclidean")
  DM6[i,3]=dist(rbind(dfn[i,],cc[2,]), method = "euclidean")
```

```
> # Cluster assignment
> DM6[,1]=mod3$cluster; DM6
  Cluster.id Dist.Clust.1 Dist.Clust.2 Dist.Clust.3 Dist.Clust.4 Dist.Clust.5
1          4           NA           NA           NA           NA           NA
2          1           NA           NA           NA           NA           NA
3          5           NA           NA           NA           NA           NA
4          5           NA           NA           NA           NA           NA
5          5           NA           NA           NA           NA           NA
6          2           NA           NA           NA           NA           NA
7          1           NA           NA           NA           NA           NA
```

```
cc        num [1:6, 1:10] -0.307 0.0...
df        35 obs. of 15 variables
dfn       35 obs. of 14 variables
DM6       int [1:35, 1:7] 4 1 5 5 5 ...
fulldf    35 obs. of 17 variables
```

has been assigned to cluster number 1, third observation has been assigned to cluster number 5, and fourth observation has been assigned to cluster number 5, so in this fashion we can see the you know assignment of these observation into different clusters we can see here, now we are going to compute the distances between these observation and clusters, cluster centroids so you can see here, I'm running a far loop it is running from 1 to 35 for each of those 35 observations, and then the distance is being computed, distance is being computed here between these observations and you know the cluster centroids, so we can see for each of these observations that are going to be 6 you know such distances because there are 6 centroids, so let's compute this.



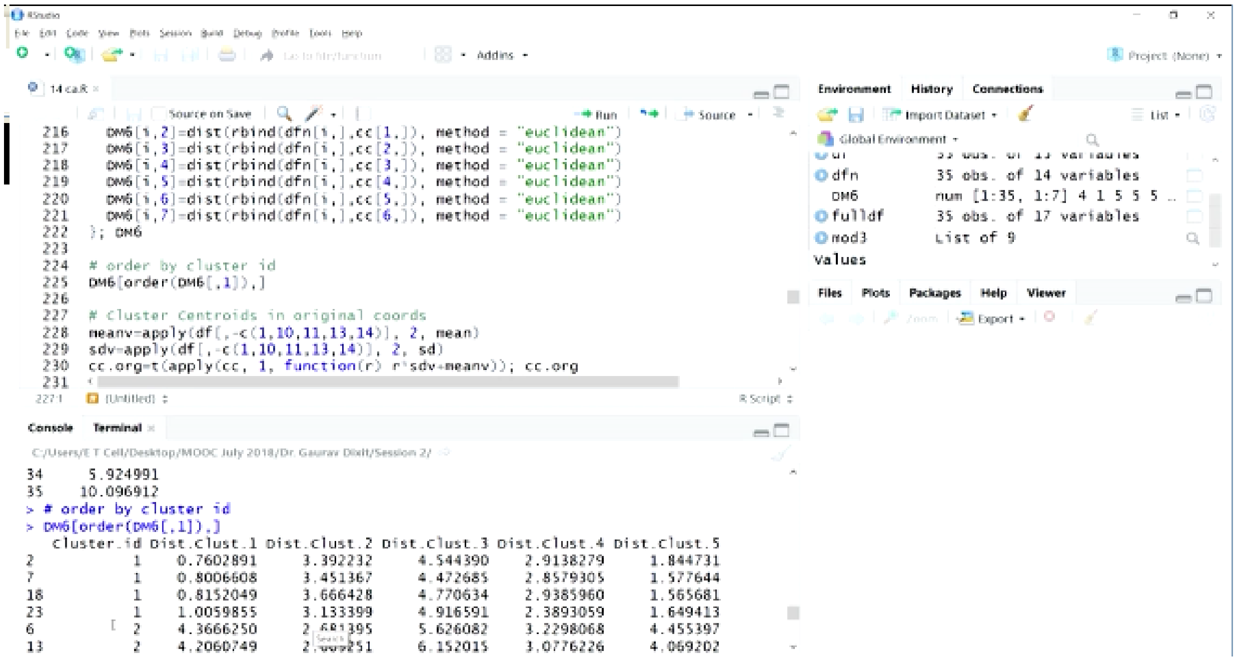
```
# compute distances
for(i in 1:35) {
  DM6[i,2]=dist(rbind(dfn[i,],cc[1,]), method = "euclidean")
  DM6[i,3]=dist(rbind(dfn[i,],cc[2,]), method = "euclidean")
  DM6[i,4]=dist(rbind(dfn[i,],cc[3,]), method = "euclidean")
  DM6[i,5]=dist(rbind(dfn[i,],cc[4,]), method = "euclidean")
  DM6[i,6]=dist(rbind(dfn[i,],cc[5,]), method = "euclidean")
  DM6[i,7]=dist(rbind(dfn[i,],cc[6,]), method = "euclidean")
}; DM6

# order by cluster id
DM6[order(DM6[,1]),]

# cluster centroids in original coords
```

```
dfn      35 obs. of 14 variables
DM6      num [1:35, 1:7] 4 1 5 5 5 ...
fulldf   35 obs. of 17 variables
mod3     List of 9
Values
```

```
+   DM6[i,7]=dist(rbind(dfn[i,],cc[6,]), method = "euclidean")
+ }; DM6
  Cluster.id Dist.Clust.1 Dist.Clust.2 Dist.Clust.3 Dist.Clust.4 Dist.Clust.5
1          4    2.3836145     1.738432     5.174669    1.5179378     2.919530
2          1    0.7602891     3.392232     4.544390    2.9138279     1.844731
3          5    1.8284493     3.318178     4.889272    2.8064581     2.169114
4          5    2.3610067     2.733976     5.071732    2.4709949     2.594349
5          5    2.4086986     3.414943     5.118619    3.0059951     2.638124
6          2    4.3666250     2.681395     5.626082    3.2298068     4.455397
7          1    0.8006608     3[illegible]367     4.472685    2.8579305     1.577644
```

Now we can see here in the output, that all the metric cells have been filled with the you know values, so the first column as we discussed the cluster assignment and then distances of this observation from each of this clusters, so if we look at the first row then we can see it has been assigned to cluster number 4, and its distance from cluster 1 is given 2.38, its distance from cluster 2 is 1.73, its distance from cluster 3 is 5.17, its distance from cluster 4 is 1.51, and its distance from cluster 5 is 2.91, if you observe this first observation itself you can see the distance of this observation from its own cluster is the smallest one, you can see 1.51, so that is why it is, it has been assigned to this cluster, so the distance of this observation from its own clusters centroid is the smallest one and other you know centroids, distance from other cluster centroid is rightly on the higher side.

Similarly we can analyze for other observations also, now what we are going to do is we'll order this particular output using cluster ID, so we'll get all the observations which are part of the same clusters, and also there you know distances from the cluster centroids, so let's run this code which will give us this output, so let's scroll up, so we can see here the first four

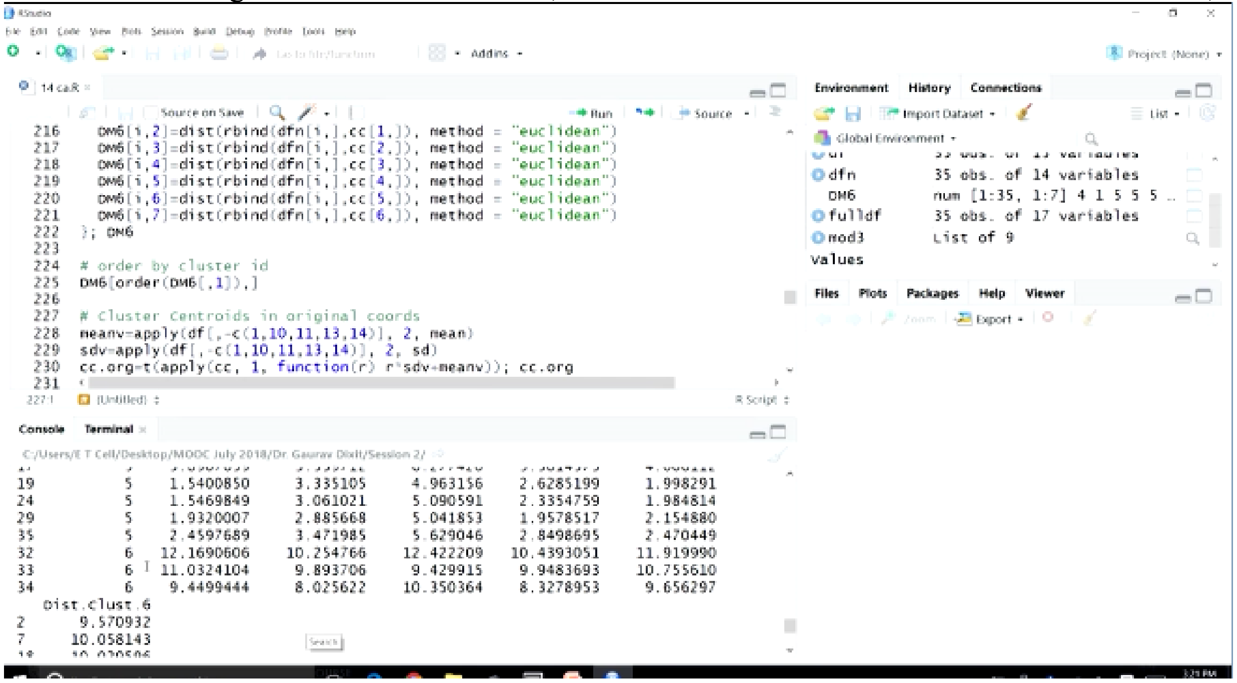


```
    DM6[i,2]=dist(rbind(dfn[i,],cc[1,]), method = "euclidean")
    DM6[i,3]=dist(rbind(dfn[i,],cc[2,]), method = "euclidean")
    DM6[i,4]=dist(rbind(dfn[i,],cc[3,]), method = "euclidean")
    DM6[i,5]=dist(rbind(dfn[i,],cc[4,]), method = "euclidean")
    DM6[i,6]=dist(rbind(dfn[i,],cc[5,]), method = "euclidean")
    DM6[i,7]=dist(rbind(dfn[i,],cc[6,]), method = "euclidean")
}; DM6

# order by cluster id
DM6[order(DM6[,1]),]

# cluster Centroids in original coords
meanv=apply(df[,-c(1,10,11,13,14)], 2, mean)
sdv=apply(df[,-c(1,10,11,13,14)], 2, sd)
cc.org=t(apply(cc, 1, function(r) r*sdv+meanv)); cc.org
```

```
dfn      35 obs. of 14 variables
DM6      num [1:35, 1:7] 4 1 5 5 5 ...
fulldf   35 obs. of 17 variables
mod3     List of 9
Values
```

```
34      5.924991
35     10.096912
> # order by cluster id
> DM6[order(DM6[,1]),]
   cluster.id Dist.Clust.1 Dist.Clust.2 Dist.Clust.3 Dist.Clust.4 Dist.Clust.5
2           1    0.7602891     3.392232     4.544390    2.9138279     1.844731
7           1    0.8006608     3.451367     4.472685    2.8579305     1.577644
18          1    0.8152049     3.666428     4.770634    2.9385960     1.565681
23          1    1.0059855     3.133399     4.916591    2.3893059     1.649413
6           2    4.3666250     2.681395     5.626082    3.2298068     4.455397
13          2    4.2060749     2.[illegible]251     6.152015    3.0776226     4.069202
```

observations which are part of cluster 1 or observation number 2, 7, 18 and 23, so these observations are part of cluster 1, and distances of these observation from 6 cluster centroids also can be seen here, now we have performed this, we can see 4 observations are part of cluster 1, then followed by number of observations which are part of cluster 2, then followed by 3 observations which are part of cluster 3, and then observations which are part of cluster 4 followed by, so in this fashion so on we can see the observations, we can identify these observation through the ID and we can see, so last one we can see that observation number 32,

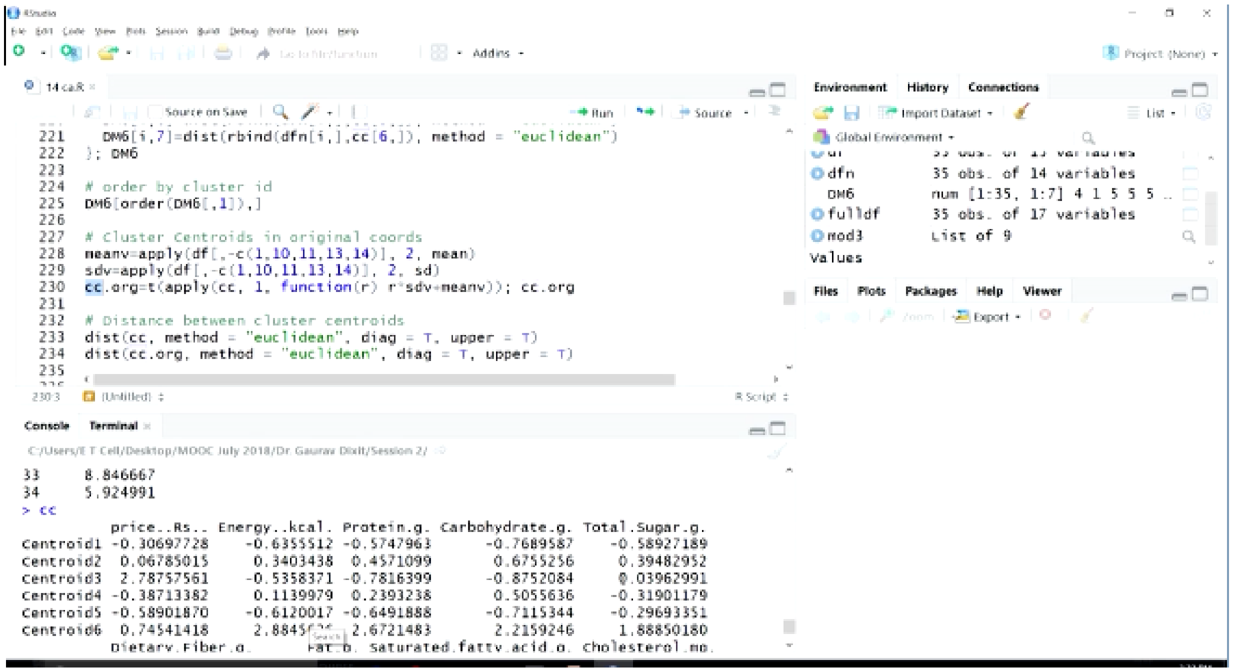


```
    DM6[i,2]=dist(rbind(dfn[i,],cc[1,]), method = "euclidean")
    DM6[i,3]=dist(rbind(dfn[i,],cc[2,]), method = "euclidean")
    DM6[i,4]=dist(rbind(dfn[i,],cc[3,]), method = "euclidean")
    DM6[i,5]=dist(rbind(dfn[i,],cc[4,]), method = "euclidean")
    DM6[i,6]=dist(rbind(dfn[i,],cc[5,]), method = "euclidean")
    DM6[i,7]=dist(rbind(dfn[i,],cc[6,]), method = "euclidean")
}; DM6

# order by cluster id
DM6[order(DM6[,1]),]

# cluster Centroids in original coords
meanv=apply(df[,-c(1,10,11,13,14)], 2, mean)
sdv=apply(df[,-c(1,10,11,13,14)], 2, sd)
cc.org=t(apply(cc, 1, function(r) r*sdv+meanv)); cc.org
```

```
dfn      35 obs. of 14 variables
DM6      num [1:35, 1:7] 4 1 5 5 5 ...
fulldf   35 obs. of 17 variables
mod3     List of 9
Values
```

```
19          5    1.5400850     3.335105     4.963156    2.6285199     1.998291
24          5    1.5469849     3.061021     5.090591    2.3354759     1.984814
29          5    1.9320007     2.885668     5.041853    1.9578517     2.154880
35          5    2.4597689     3.471985     5.629046    2.8498695     2.470449
32          6   12.1690606    10.254766    12.422209   10.4393051    11.919990
33          6   11.0324104     9.893706     9.429915    9.9483693    10.755610
34          6    9.4499444     8.025622    10.350364    8.3278953     9.656297
   Dist.Clust.6
2      9.570932
7     10.058143
```

33, and 34 they are part of cluster number 6, so we can identify which observation has been assigned to which cluster and their distances from these cluster centroids, and we can also see that these observations they are closest to their you know assigned you know cluster centroid.

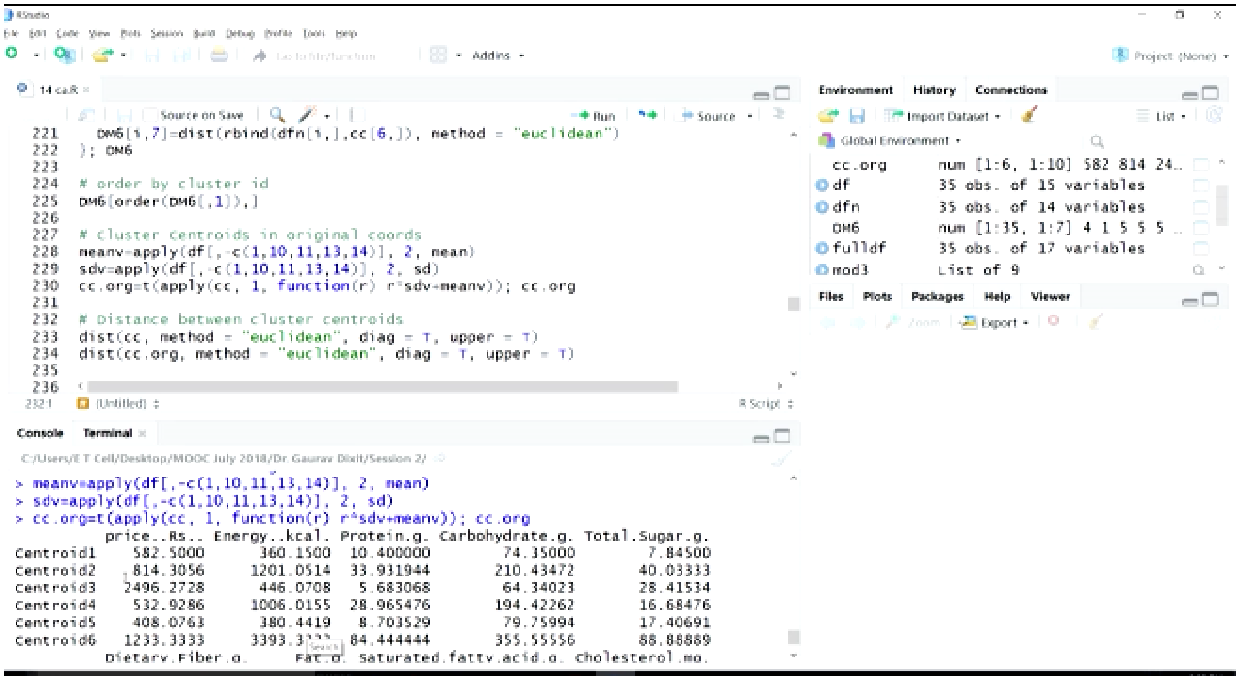


```
  DM6[i,7]=dist(rbind(dfn[i,],cc[6,]), method = "euclidean")
}; DM6

# order by cluster id
DM6[order(DM6[,1]),]

# Cluster Centroids in original coords
meanv=apply(df[,-c(1,10,11,13,14)], 2, mean)
sdv=apply(df[,-c(1,10,11,13,14)], 2, sd)
cc.org=t(apply(cc, 1, function(r) r*sdv+meanv)); cc.org

# Distance between cluster centroids
dist(cc, method = "euclidean", diag = T, upper = T)
dist(cc.org, method = "euclidean", diag = T, upper = T)
```

```
dfn       35 obs. of 14 variables
DM6       num [1:35, 1:7] 4 1 5 5 5 ...
fulldf    35 obs. of 17 variables
mod3      List of 9
Values
```

```
C:/Users/E T Cell/Desktop/MOOC July 2018/Dr. Gaurav Dixit/Session 2/
33      8.846667
34      5.924991
> cc
           price..Rs.. Energy..kcal. Protein.g. Carbohydrate.g. Total.Sugar.g.
Centroid1 -0.30697728    -0.6355512 -0.5747963      -0.7689587    -0.58927189
Centroid2  0.06785015     0.3403438  0.4571099       0.6755256     0.39482952
Centroid3  2.78757561    -0.5358371 -0.7816399      -0.8752084     0.03962991
Centroid4 -0.38713382     0.1139979  0.2393238       0.5055636    -0.31901179
Centroid5 -0.58901870    -0.6120017 -0.6491888      -0.7115344    -0.29693351
Centroid6  0.74541418     2.8845[illegible]  2.6721483       2.2159246     1.88850180
           Dietary.Fiber.g.      Fat.g. Saturated.fatty.acid.g. Cholesterol.mg.
```

So let’s move ahead, now if we are interested in looking at the coordinates of these cluster centroids, so till now what we have seen is that earlier that CC value, so this CC value that this actually contains the coordinates for, coordinates for cluster centroids in normalized scales, so as you can see here these are the 6 cluster centroid, and these coordinates you know different you know dimensions, price, energy, protein, carbohydrates and other variables, other dimensions, so these are you know these are coordinates in the normalized you know scales, if we want these coordinates in the original scales we’ll have to use this particular piece of code where you can see that the mean value for the average value for each of these dimensions is going to be computed, and then standard deviation value for each of these dimension is going to be computed, now these mean and standard deviation value are going to be used to convert the normalized scale value into original scale values, so let’s go through this code, first we compute the mean values then standard deviation value and then you can see here, we are applying these values on you know CC which is cluster centroid coordinates on normalized scales, and we’ll get the centroids on using original scales.

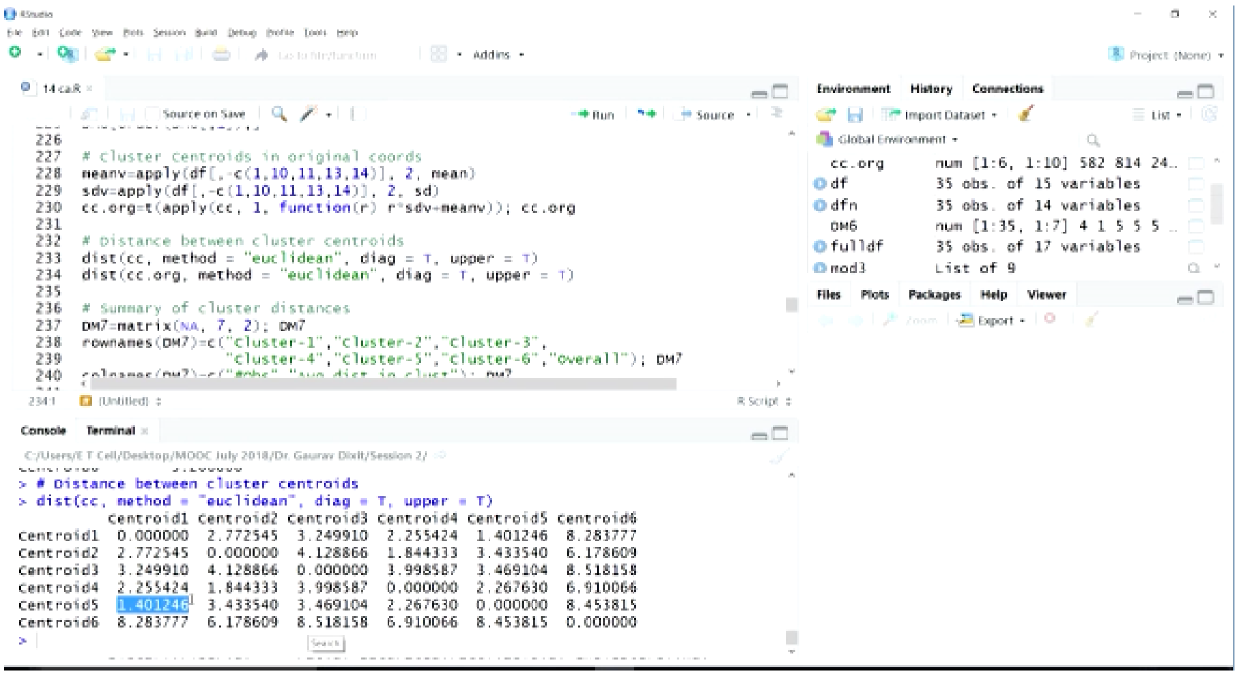


```r
  DM6[i,7]=dist(rbind(dfn[i,],cc[6,]), method = "euclidean")
}; DM6

# order by cluster id
DM6[order(DM6[,1]),]

# cluster centroids in original coords
meanv=apply(df[,-c(1,10,11,13,14)], 2, mean)
sdv=apply(df[,-c(1,10,11,13,14)], 2, sd)
cc.org=t(apply(cc, 1, function(r) r*sdv+meanv)); cc.org

# Distance between cluster centroids
dist(cc, method = "euclidean", diag = T, upper = T)
dist(cc.org, method = "euclidean", diag = T, upper = T)
```

```
cc.org      num [1:6, 1:10] 582 814 24..
df          35 obs. of 15 variables
dfn         35 obs. of 14 variables
DM6         num [1:35, 1:7] 4 1 5 5 5 ..
fulldf      35 obs. of 17 variables
mod3        List of 9
```

```
C:/Users/E T Cell/Desktop/MOOC July 2018/Dr. Gaurav Dixit/Session 2/
> meanv=apply(df[,-c(1,10,11,13,14)], 2, mean)
> sdv=apply(df[,-c(1,10,11,13,14)], 2, sd)
> cc.org=t(apply(cc, 1, function(r) r*sdv+meanv)); cc.org
          price..Rs.. Energy..kcal. Protein.g. Carbohydrate.g. Total.Sugar.g.
Centroid1    582.5000      360.1500  10.400000        74.35000        7.84500
Centroid2    814.3056     1201.0514  33.931944       210.43472       40.03333
Centroid3   2496.2728      446.0708   5.683068        64.34023       28.41534
Centroid4    532.9286     1006.0155  28.965476       194.42262       16.68476
Centroid5    408.0763      380.4419   8.703529        79.75994       17.40691
Centroid6   1233.3333     3393.3333  84.444444       355.55556       88.88889
          Dietary.Fiber.g.      Fat.g. Saturated.fatty.acid.g. Cholesterol.mg.
```

Now let’s have a look at the output, so as you can see now the centroids and their coordinates using the original scales, now you can see the price values they are in their original scales, centroid 1 is having the price value of 582.5, energy value of 360.15 and similarly other values, similarly we can look at the original values for other centroids.
Now all these computations that we have just done can be used for the next important you know, next important computation that we wanted, is the distance between clusters centroids, we are interested in analyzing the distance between cluster centroids because that can give us idea about how clusters are separated from each other which is important because the objective of cluster analysis to get the meaningful and insightful clusters and looking at the distances between these cluster centroids will give us a sense about how you know separated these clusters are, so let’s compute this, so coordinates in the normalized scales and in the original scales both we have, so we can have a look at both of this outputs, so this particular piece of code is going to compute for us the distances between cluster centroids, so let’s run this so we can see here, so in this matrix we can see each of the 6 centroid and their distance with the remaining centroids, so this is symmetrical matrix some of the values are going to be common,



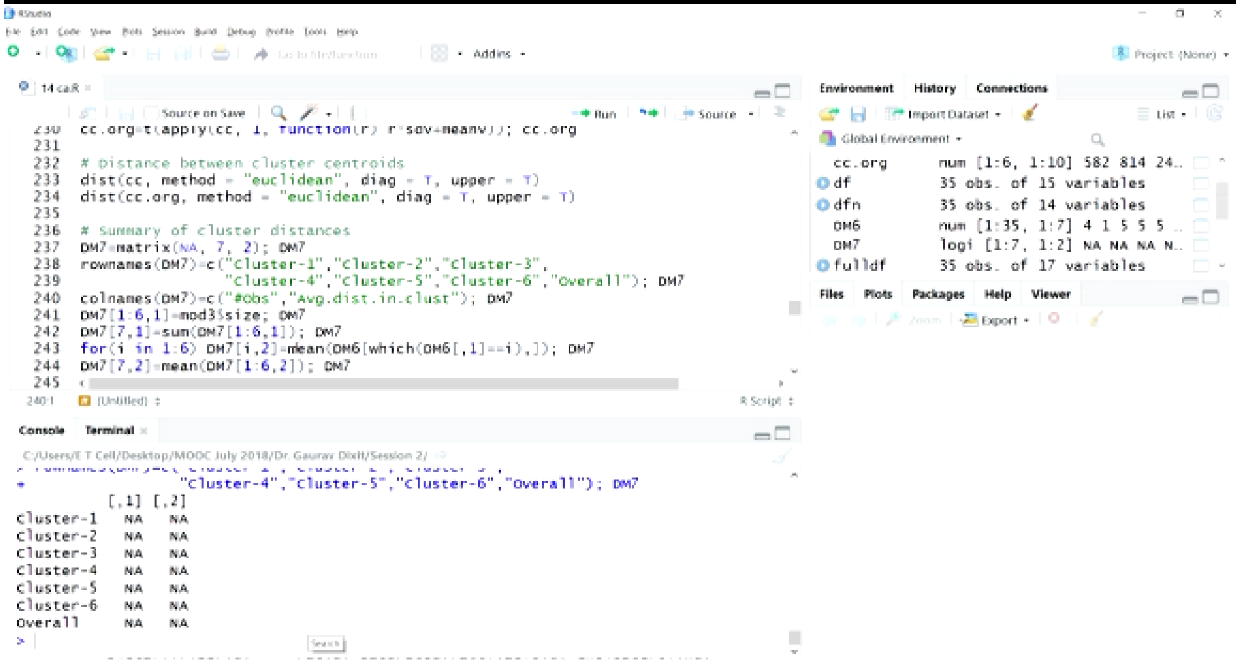
```
# cluster centroids in original coords
meanv=apply(df[,-c(1,10,11,13,14)], 2, mean)
sdv=apply(df[,-c(1,10,11,13,14)], 2, sd)
cc.org=t(apply(cc, 1, function(r) r*sdv+meanv)); cc.org

# Distance between cluster centroids
dist(cc, method = "euclidean", diag = T, upper = T)
dist(cc.org, method = "euclidean", diag = T, upper = T)

# Summary of cluster distances
DM7=matrix(NA, 7, 2); DM7
rownames(DM7)=c("Cluster-1","Cluster-2","Cluster-3",
                "Cluster-4","Cluster-5","Cluster-6","Overall"); DM7
```

```
cc.org    num [1:6, 1:10] 582 814 24..
df        35 obs. of 15 variables
dfn       35 obs. of 14 variables
DM6       num [1:35, 1:7] 4 1 5 5 5 ..
fulldf    35 obs. of 17 variables
mod3      List of 9
```

```
> # Distance between cluster centroids
> dist(cc, method = "euclidean", diag = T, upper = T)
          Centroid1 Centroid2 Centroid3 Centroid4 Centroid5 Centroid6
Centroid1  0.000000  2.772545  3.249910  2.255424  1.401246  8.283777
Centroid2  2.772545  0.000000  4.128866  1.844333  3.433540  6.178609
Centroid3  3.249910  4.128866  0.000000  3.998587  3.469104  8.518158
Centroid4  2.255424  1.844333  3.998587  0.000000  2.267630  6.910066
Centroid5  1.401246  3.433540  3.469104  2.267630  0.000000  8.453815
Centroid6  8.283777  6.178609  8.518158  6.910066  8.453815  0.000000
>
```

so if we look at the centroid 1 here and its distance from other centroids 2, 3, 4, 5 so we can see here, so this centroid 1 seems to be closest to centroid 5, so centroid 1 seem to be closest to centroid 5, and much far away from the centroid 6, that means cluster 6, so centroid 1 and centroid 6 they are farthest, and centroid 1 and centroid 5 they are closest, so this can also give us some idea about, though we have this 6 clusters which clusters can be joined and merged, so this idea also we can get from here.

So let's move forward, so original scales we can also compute the original scales here, so we can see the values in original scales and also you'll you know see the same thing, centroid 1 if we look at the centroid 1 and we can see it is much far away from centroid 6, and much closer with centroid 5, so similar kind of analysis we can do for other centroids also.

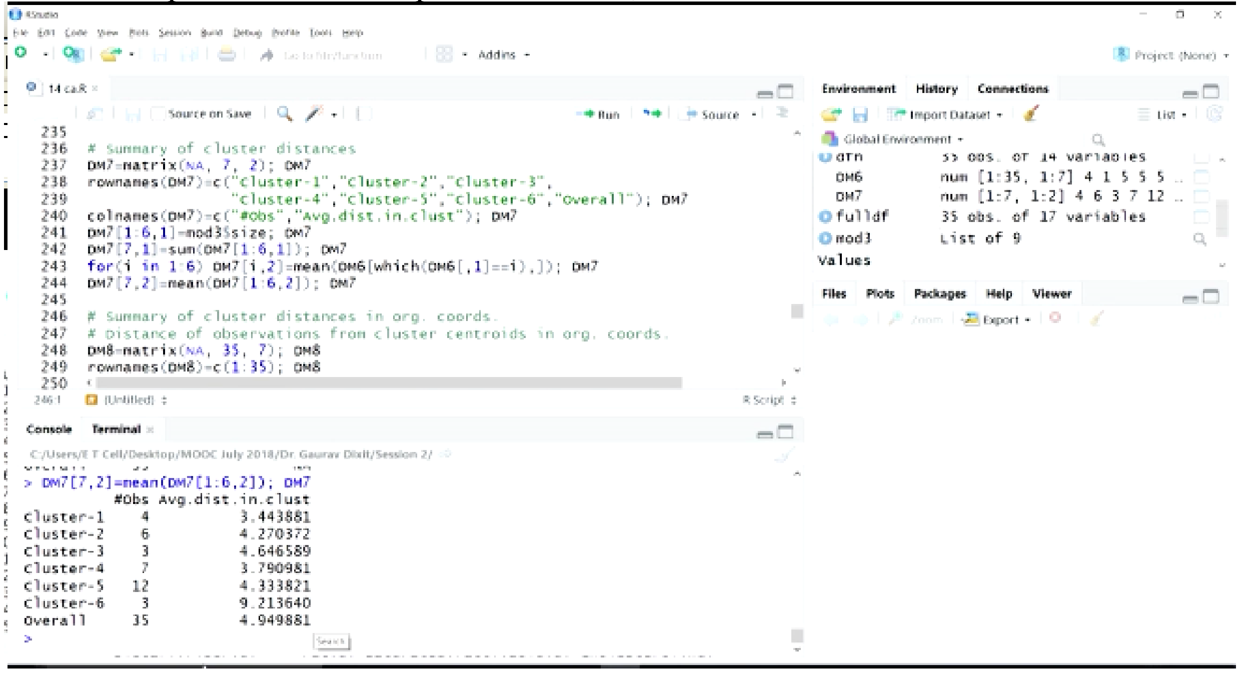


```
cc.org=t(apply(cc, 1, function(r) r*sdv+meanv)); cc.org

# Distance between cluster centroids
dist(cc, method = "euclidean", diag = T, upper = T)
dist(cc.org, method = "euclidean", diag = T, upper = T)

# Summary of cluster distances
DM7=matrix(NA, 7, 2); DM7
rownames(DM7)=c("Cluster-1","Cluster-2","Cluster-3",
                "Cluster-4","Cluster-5","Cluster-6","Overall"); DM7
colnames(DM7)=c("#Obs","Avg.dist.in.clust"); DM7
DM7[1:6,1]=mod3$size; DM7
DM7[7,1]=sum(DM7[1:6,1]); DM7
for(i in 1:6) DM7[i,2]=mean(DM6[which(DM6[,1]==i),]); DM7
DM7[7,2]=mean(DM7[1:6,2]); DM7
```

```
cc.org    num [1:6, 1:10] 582 814 24..
df        35 obs. of 15 variables
dfn       35 obs. of 14 variables
DM6       num [1:35, 1:7] 4 1 5 5 5 ..
DM7       logi [1:7, 1:2] NA NA NA N..
fulldf    35 obs. of 17 variables
```

```
+                 "Cluster-4","Cluster-5","Cluster-6","Overall"); DM7
          [,1] [,2]
Cluster-1   NA   NA
Cluster-2   NA   NA
Cluster-3   NA   NA
Cluster-4   NA   NA
Cluster-5   NA   NA
Cluster-6   NA   NA
Overall     NA   NA
>
```

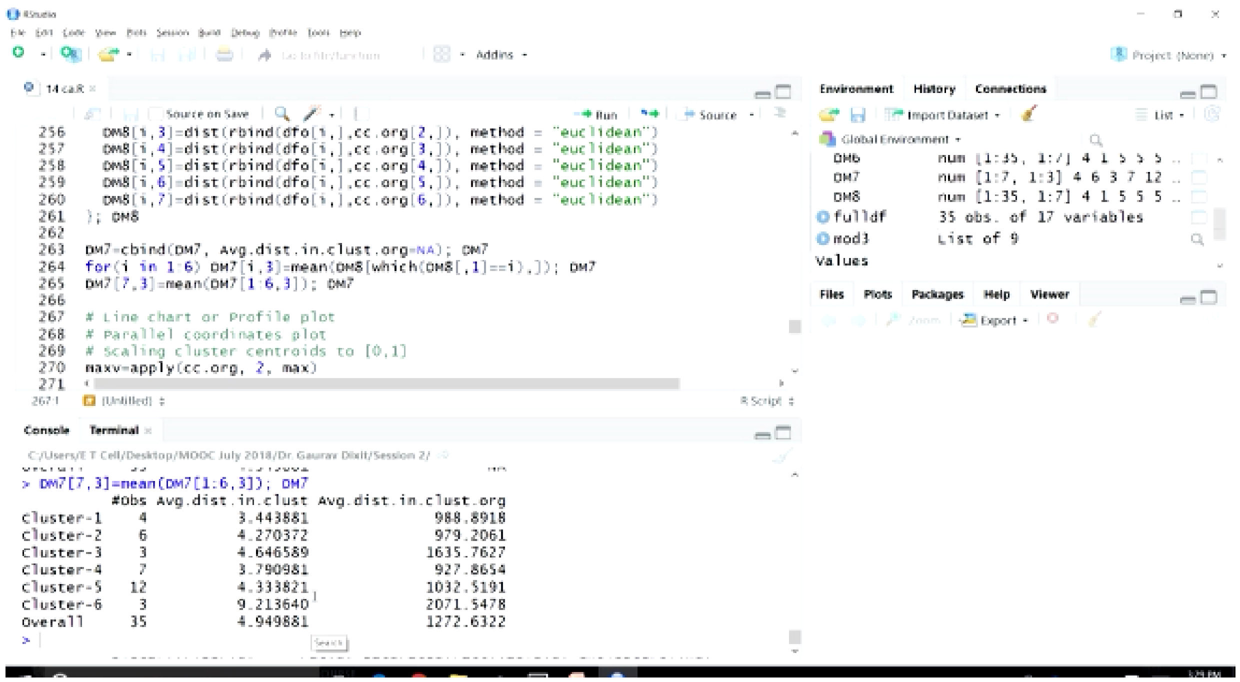
Now let’s look at some other important piece of information, now what we are going to do is we are going to summarize these cluster distances, so let’s create this matrix where we have you know this is 7 by 2, so let’s initialize this matrix, row names you can see we have 6 clusters and then the overall results, so that is why we needed 6, we needed 7 rows here, so let’s run this, 2 columns, so 1 indicating the number of observation so we’ll get to know the number of observation in a summary format, now which cluster is having you know what number of observation, and the average distance that is there in cluster, so that will also give us some idea about the dispersion within that particular cluster, so let’s run this.



```
# Summary of cluster distances
DM7=matrix(NA, 7, 2); DM7
rownames(DM7)=c("cluster-1","Cluster-2","Cluster-3",
                "cluster-4","Cluster-5","Cluster-6","Overall"); DM7
colnames(DM7)=c("#Obs","Avg.dist.in.clust"); DM7
DM7[1:6,1]=mod3$size; DM7
DM7[7,1]=sum(DM7[1:6,1]); DM7
for(i in 1:6) DM7[i,2]=mean(DM6[which(DM6[,1]==i),]); DM7
DM7[7,2]=mean(DM7[1:6,2]); DM7

# Summary of cluster distances in org. coords.
# Distance of observations from cluster centroids in org. coords.
DM8=matrix(NA, 35, 7); DM8
rownames(DM8)=c(1:35); DM8
```

```
> DM7[7,2]=mean(DM7[1:6,2]); DM7
          #Obs Avg.dist.in.clust
Cluster-1    4          3.443881
Cluster-2    6          4.270372
Cluster-3    3          4.646589
Cluster-4    7          3.790981
Cluster-5   12          4.333821
Cluster-6    3          9.213640
Overall     35          4.949881
>
```

So we’ll go through this part of code to find out this summary table, so we can see here cluster 1 having 4 observations, cluster 2 having 6 observations, cluster 3, 3, and then cluster 4 is 7, cluster 5 is the biggest cluster with having 12 observations, so overall 35 as you can see, so cluster 3 and cluster 6 are the smallest cluster having just 3 observations.
Now if we analyze the average distance in each of these clusters, we can find from here that the cluster 1 it is having the smallest average distance among all the clusters and it is much lower than the average, so average is closer to 5 and the overall average is closer to 5, and the average in for this cluster 1 is closer to 3.5, so there is a significant difference, so cluster 1 seems to be you know having lower you know dispersion, so it is seems to be more homogeneous in that sense, if we look at the clusters 6, so average distance is much higher, it is almost you know double of the overall and if we compared with the cluster 1 it is much higher, so cluster 6 seems to be having higher dispersion, so less homogeneous cluster.

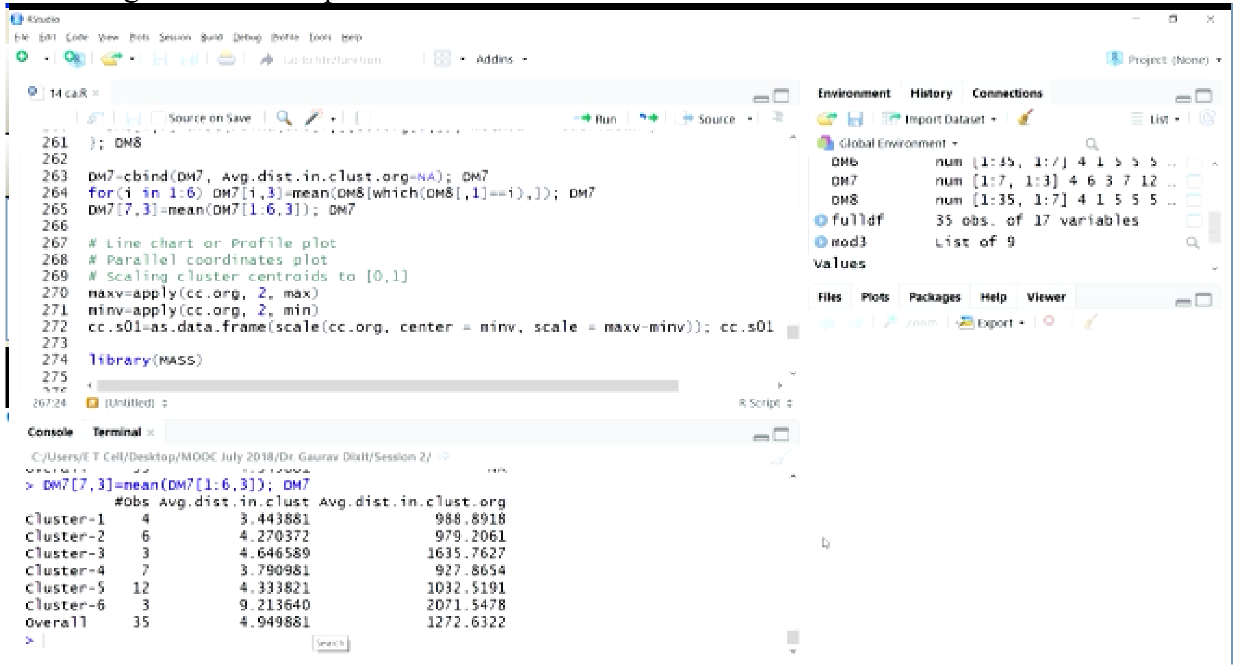


```
    DM8[i,3]=dist(rbind(dfo[i,],cc.org[2,]), method = "euclidean")
    DM8[i,4]=dist(rbind(dfo[i,],cc.org[3,]), method = "euclidean")
    DM8[i,5]=dist(rbind(dfo[i,],cc.org[4,]), method = "euclidean")
    DM8[i,6]=dist(rbind(dfo[i,],cc.org[5,]), method = "euclidean")
    DM8[i,7]=dist(rbind(dfo[i,],cc.org[6,]), method = "euclidean")
}; DM8

DM7=cbind(DM7, Avg.dist.in.clust.org=NA); DM7
for(i in 1:6) DM7[i,3]=mean(DM8[which(DM8[,1]==i),]); DM7
DM7[7,3]=mean(DM7[1:6,3]); DM7

# Line chart or Profile plot
# Parallel coordinates plot
# Scaling cluster centroids to [0,1]
maxv=apply(cc.org, 2, max)
```

```
DM6        num [1:35, 1:7] 4 1 5 5 5 ..
DM7        num [1:7, 1:3] 4 6 3 7 12 ..
DM8        num [1:35, 1:7] 4 1 5 5 5 ..
fulldf     35 obs. of 17 variables
mod3       List of 9
Values
```

```
> DM7[7,3]=mean(DM7[1:6,3]); DM7
          #Obs Avg.dist.in.clust Avg.dist.in.clust.org
Cluster-1    4          3.443881              988.8918
Cluster-2    6          4.270372              979.2061
Cluster-3    3          4.646589             1635.7627
Cluster-4    7          3.790981              927.8654
Cluster-5   12          4.333821             1032.5191
Cluster-6    3          9.213640             2071.5478
Overall     35          4.949881             1272.6322
>
```

Same kind of analysis we can also do using the you know original coordinates, so let's go through this part of code which will give us that result, so the similar kind of table but now we'll have the distances, average distances in the original coordinates that is all, so this is the table as you can see here, first column is same then we have the average distance in cluster that was using the normalized scale and then read the last column is having the original scales, so we can see here that cluster 1, average distances lowest, cluster it is, so we can see here, in terms of original scales, results are more or less similar, but in terms of original scale the cluster 4 is having, seems to be having the lowest, other results you know comparable, if we compare these clusters you can see clusters 6 is having highest to dispersion, and cluster 1 and 4 and 2 are having the lowest dispersion.

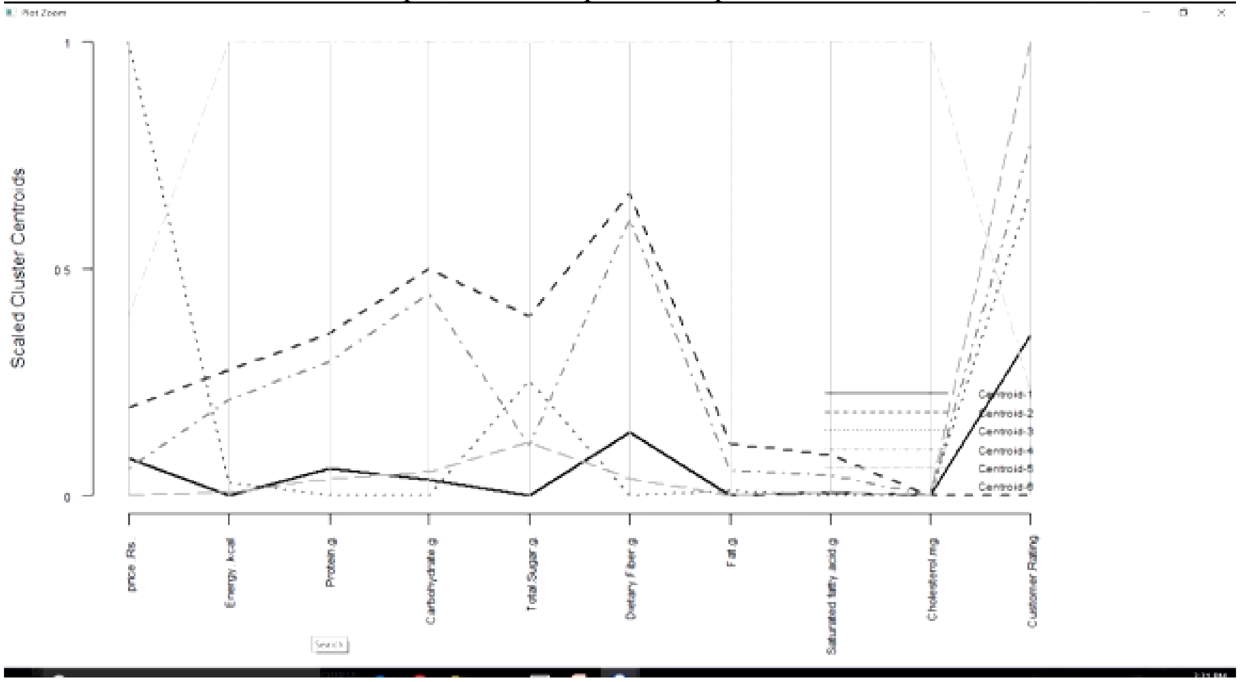


```
}; DM8

DM7=cbind(DM7, Avg.dist.in.clust.org=NA); DM7
for(i in 1:6) DM7[i,3]=mean(DM8[which(DM8[,1]==i),]); DM7
DM7[7,3]=mean(DM7[1:6,3]); DM7

# Line chart or Profile plot
# Parallel coordinates plot
# Scaling cluster centroids to [0,1]
maxv=apply(cc.org, 2, max)
minv=apply(cc.org, 2, min)
cc.s01=as.data.frame(scale(cc.org, center = minv, scale = maxv-minv)); cc.s01

library(MASS)
```

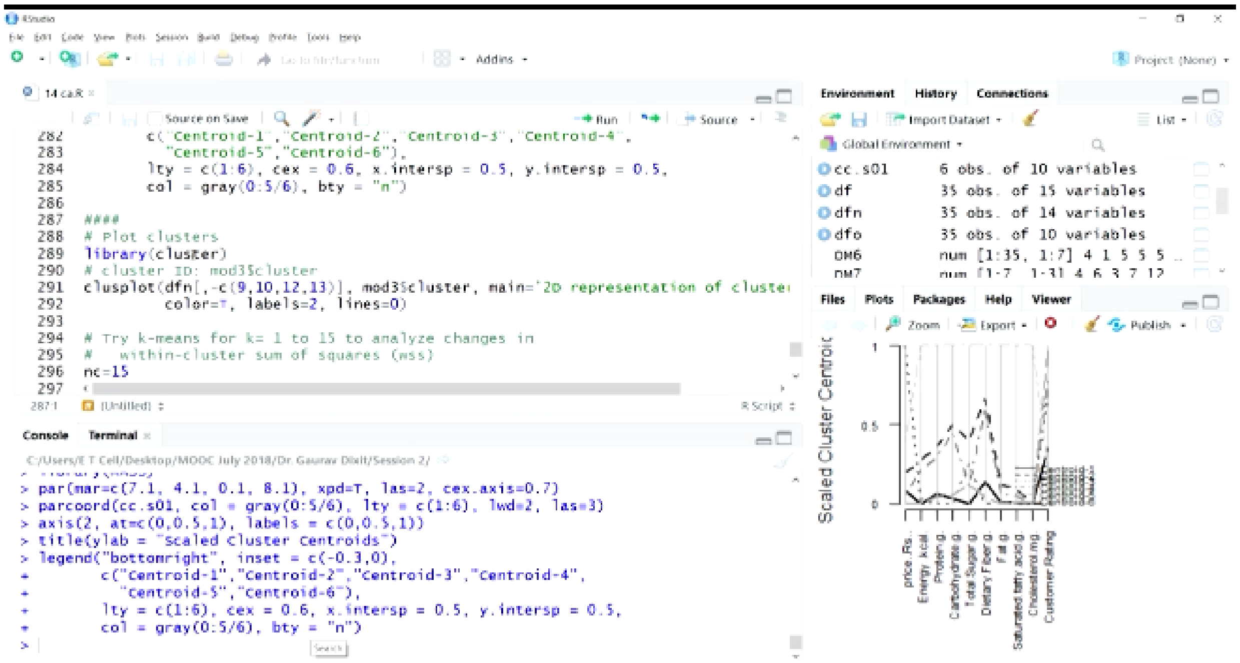
```
DM6        num [1:35, 1:7] 4 1 5 5 5 ..
DM7        num [1:7, 1:3] 4 6 3 7 12 ..
DM8        num [1:35, 1:7] 4 1 5 5 5 ..
fulldf     35 obs. of 17 variables
mod3       List of 9
Values
```

```
> DM7[7,3]=mean(DM7[1:6,3]); DM7
          #Obs Avg.dist.in.clust Avg.dist.in.clust.org
Cluster-1    4          3.443881              988.8918
Cluster-2    6          4.270372              979.2061
Cluster-3    3          4.646589             1635.7627
Cluster-4    7          3.790981              927.8654
Cluster-5   12          4.333821             1032.5191
Cluster-6    3          9.213640             2071.5478
Overall     35          4.949881             1272.6322
>
```

Now let's analyze this results further, so what we are going to do is we are going to create a parallel coordinates plot, this particular plot is going to help us in terms of characterizing these clusters, so let's go through this part of the code, so before we create the parallel coordinates plot as we have discussed in our previous codes, when we were discussing the visualization techniques we have to change the scales to 0, 1, so let's do this, once the data has been changed with the scale we can use this particular library mass and there we can call this function parcoord which will give us the parallel coordinates plot, so let's go through this part of the code we can see here in the output, so in this particular plot we can see a 6 lines here, each



representing you know centroids for you know their clusters, so we can always analyze this particular you know plot in terms of you know characterizing these clusters, so if we look at this particular cluster this one, so this particular cluster is having the you know higher values for energy, protein and many of the you know variables it is having the higher value except the price, price is in the medium range and the last one rating is also in the medium range, so it is in terms of, it is higher in terms of energy and everything else, so it seems to be more you know rich kind of cereal, that breakfast cereal that is there, so in that sense we can characterize this, so rich cereal could be one characterization for this particular you know, in this particular cluster, so rich set of cereals can be one characterization.
Similarly if we look at this you know bold line, so this particular cluster and along with one more cluster, but this one more specifically is having values, lower values along all the dimensions, so in that sense if we look at this particular cluster then it is having almost lower values along all, for all the variables so therefore it can be you know, it can be called you know low richness you know cereal breakfast cereals and we can see the price is on the, in the lower side, however rating is in the you know medium range, so therefore this particular lecture can be, can also be characterize, can be characterize as low you know low rich level, so in this fashion we can characterize different clusters depending on the kind of you know values that they are taking, so this is one way to understand the clusters and the characterize.

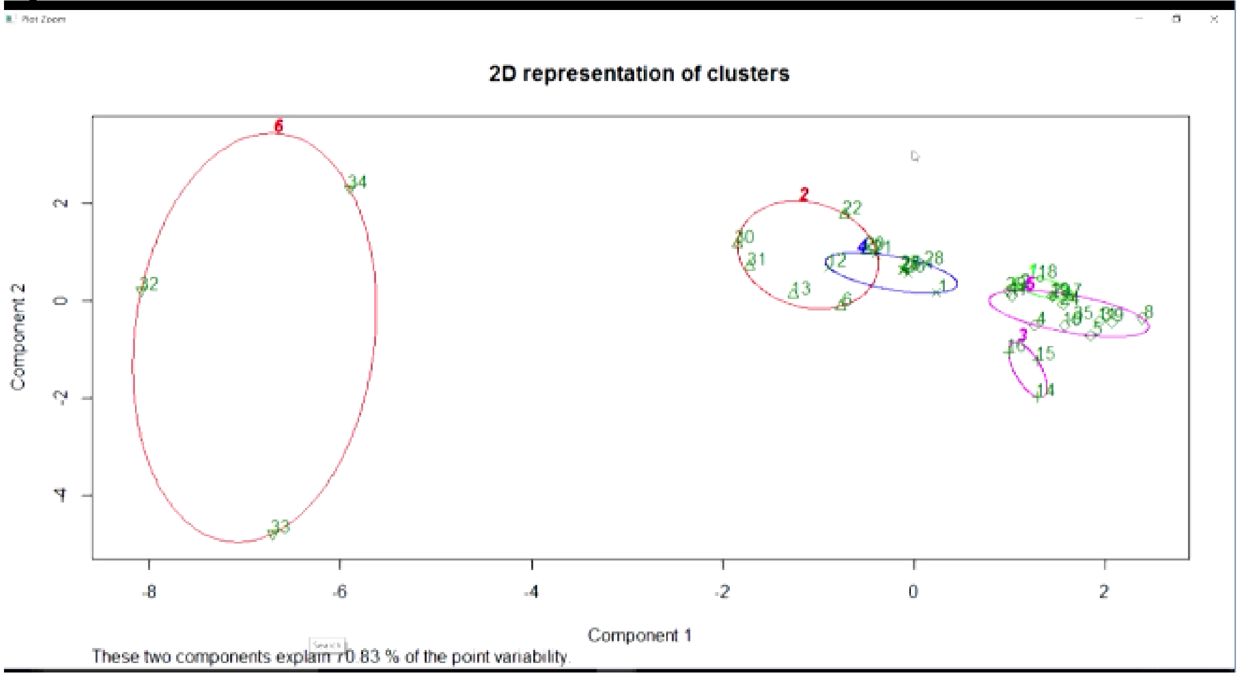


```
        c("Centroid-1","Centroid-2","Centroid-3","Centroid-4",
          "Centroid-5","Centroid-6"),
        lty = c(1:6), cex = 0.6, x.intersp = 0.5, y.intersp = 0.5,
        col = gray(0:5/6), bty = "n")

####
# Plot clusters
library(cluster)
# cluster ID: mod3$cluster
clusplot(dfn[,-c(9,10,12,13)], mod3$cluster, main='2D representation of cluste
         color=T, labels=2, lines=0)

# Try k-means for k= 1 to 15 to analyze changes in
#   within-cluster sum of squares (wss)
nc=15
```

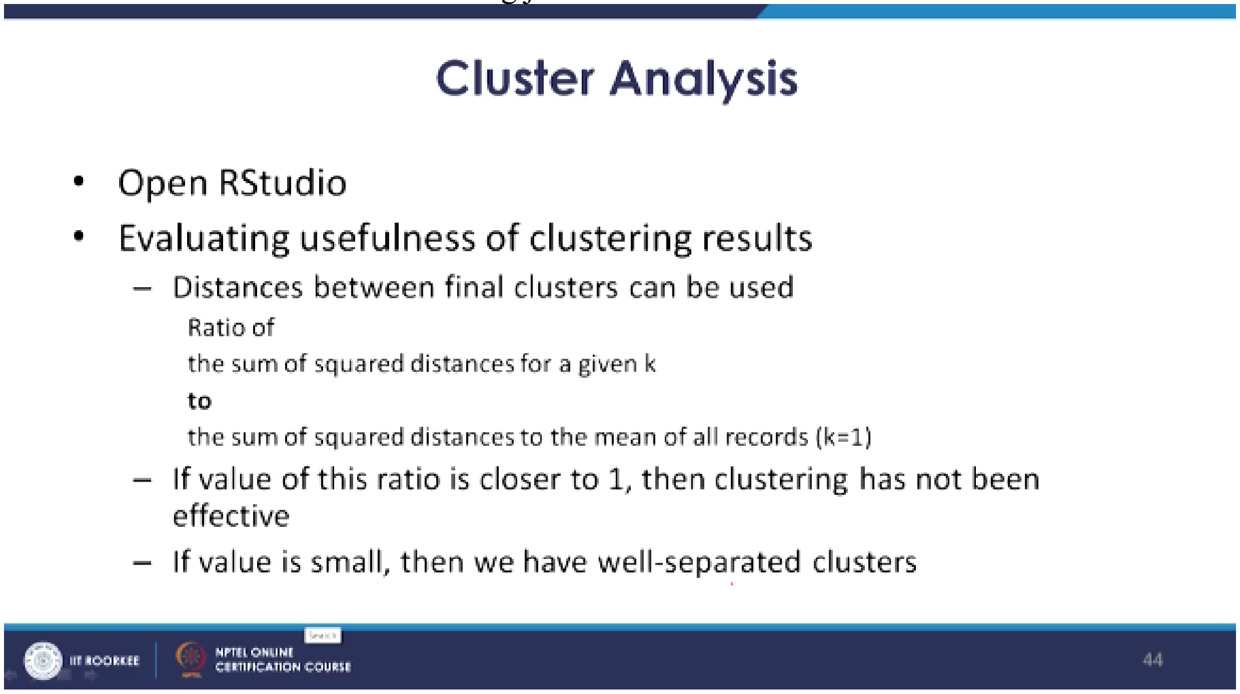
```
C:/Users/E T Cell/Desktop/MOOC July 2018/Dr. Gaurav Dixit/Session 2/
> par(mar=c(7.1, 4.1, 0.1, 8.1), xpd=T, las=2, cex.axis=0.7)
> parcoord(cc.s01, col = gray(0:5/6), lty = c(1:6), lwd=2, las=3)
> axis(2, at=c(0,0.5,1), labels = c(0,0.5,1))
> title(ylab = "Scaled Cluster Centroids")
> legend("bottomright", inset = c(-0.3,0),
+        c("Centroid-1","Centroid-2","Centroid-3","Centroid-4",
+          "Centroid-5","Centroid-6"),
+        lty = c(1:6), cex = 0.6, x.intersp = 0.5, y.intersp = 0.5,
+        col = gray(0:5/6), bty = "n")
>
```

```
cc.s01    6 obs. of 10 variables
df        35 obs. of 15 variables
dfn       35 obs. of 14 variables
dfo       35 obs. of 10 variables
OM6       num [1:35, 1:7] 4 1 5 5 5 ...
```

Now what we are going to do is next we are going to create a plot, so for this we are going to use this particular library cluster and this plot is going to be 2D representation of clusters, so even though there are many dimensions because of the presence of many variables, but those variables are going to be brought down to 2D you know levels, so 2D representation of clusters is something that we are going to create, so this is the function clust plot that is going to be used, so in this we'll get this output, so first we are passing on the data then we are passing on the cluster assignments how this you know different clusters you know the different observation have been assigned to different clusters and this information is going to be used to create a 2D representation, so let's run this.



Before we run this let's clear the previous part so that we get a more clear picture here, so this is the plot we can see here, so in this 2D representation this is 2 dimensional component 1 and component 2, however this is good enough 2D representation because we can see that these 2 components explain 70.83% of the point variability, this is much similar to what we did, this is like multi-dimensional scaling, what we did in PCA, so this analysis is quite similar to that, but what we are seeing right now is the graphical representation of that analysis, so we can see here the cluster number 6 and the points that are observation which are part of this cluster 6, which are clearly visible in this plot and we can also see that these points are, these observation the cereals are also you know, average distance is higher, right, if we look at the other clusters so there are more you know a bit of overlapping clusters, however let me remind you this is 2D representation of the whole clustering process, so you can see cluster number 2 the observation which are part of a cluster number 1, cluster number 4, and observation which are part of it cluster number 5 which is very closely going along with the cluster number you know 1 and then the cluster number 3 is here having just 3 observations.



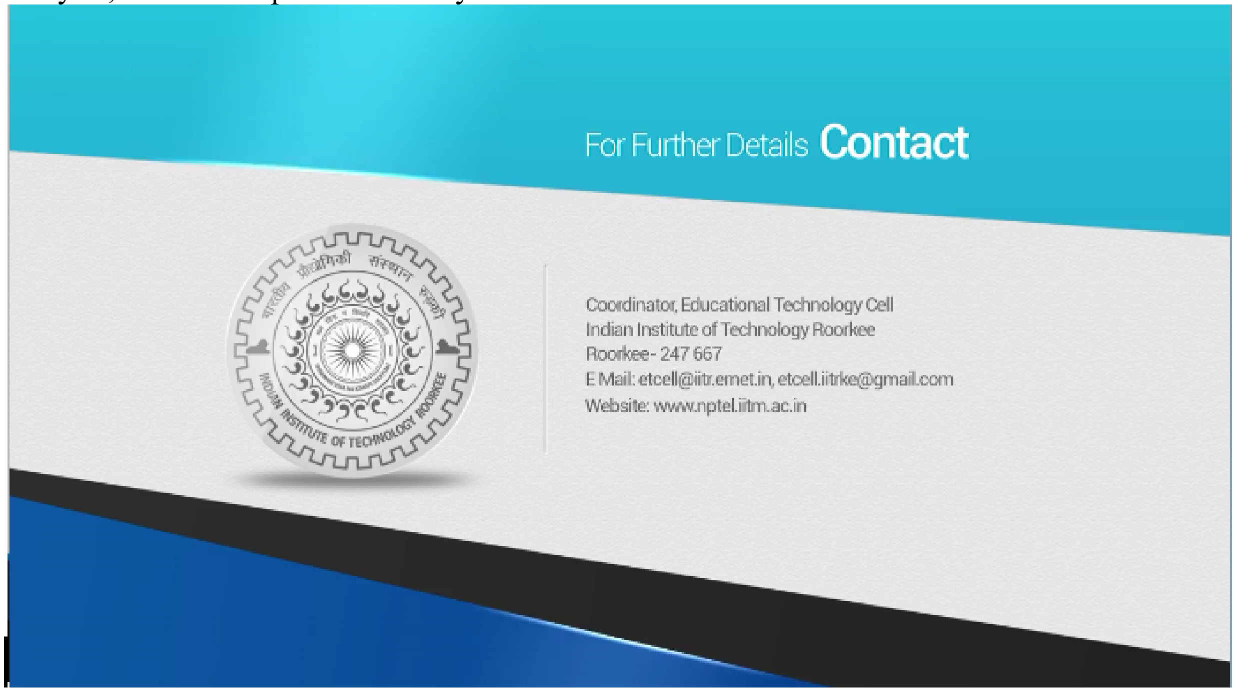
Cluster Analysis

- Open RStudio
- Evaluating usefulness of clustering results
  - Distances between final clusters can be used
    Ratio of
    the sum of squared distances for a given k
    to
    the sum of squared distances to the mean of all records (k=1)
  - If value of this ratio is closer to 1, then clustering has not been effective
  - If value is small, then we have well-separated clusters

IIT ROORKEE NPTEL ONLINE CERTIFICATION COURSE 44

Cluster number 3 and 6 having just 3 observation and we can see other cluster were having few more observations here, so in this fashion also we can understand the clusters and the characterizing, characterization process that we did can also be completed. So with this we'll like to go back to our discussion, so let's go back, so few more points about specifically about the K means clustering, so how do we evaluate usefulness of these clustering results, so one metric that could be used is the distances between final clusters that can actually be used to compute this ratio, the sum of squared distances for a given K to the sum of squared distances to the mean of all records, so this particular ratio value will give us about the idea about whether the clustering results are useful or not, so this is more about understanding whether the clusters are separated or not, so if the clusters are separated then probably the clustering process, clustering results are good, so this particular ratio will give us that indication, so the value of this ratio comes out to be closer to 1, then probably clustering has not been effective and there are too many overlaps, significant overlaps, if the value of this particular ratio is small, then probably we have well separated clusters.

So if we have well separated cluster then it is more likely that the clustering results are going to be useful, and as we have seen in the R studio, in the R exercise that we can always characterize them, analyze them and generate meaningful insights from them, so that is the end of cluster analysis, so we'll stop here. 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

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