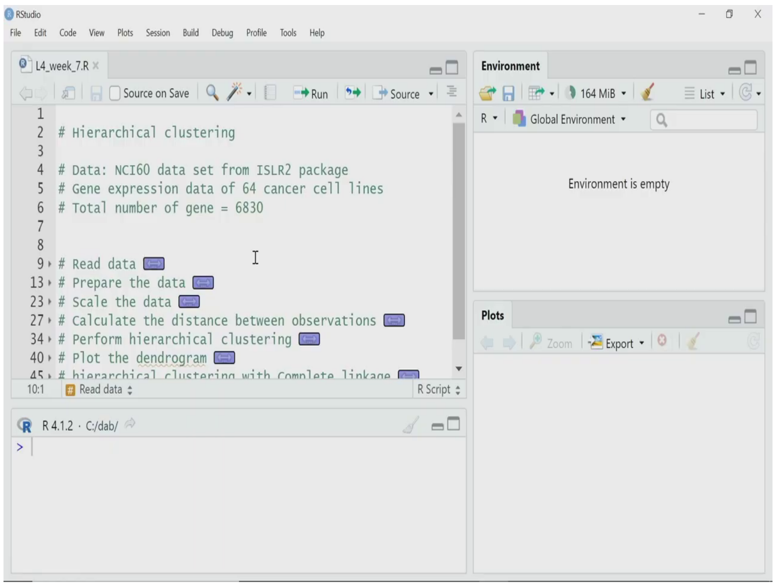
Data Analysis for Biologists

Professor Biplab Bose
Department of Biosciences and Bioengineering
Indian Institute of Technology, Guwahati

Hierarchical Clustering using R

Hello everyone. In this lecture, we will perform hierarchical clustering of a data using R. So, for this particular demonstration, I have chosen a data set, where we have gene expression data of 64 different cancer cell lines, there are lots of gene more than 6000 genes are there. So, what do you want to do, we want to cluster the cell line based upon their gene expression features. And I want to do it like a hierarchical clustering, I want to create a tree I want to create a dendrogram. So, that will show the relation between different cell types. So, let us go to R studio and perform that.

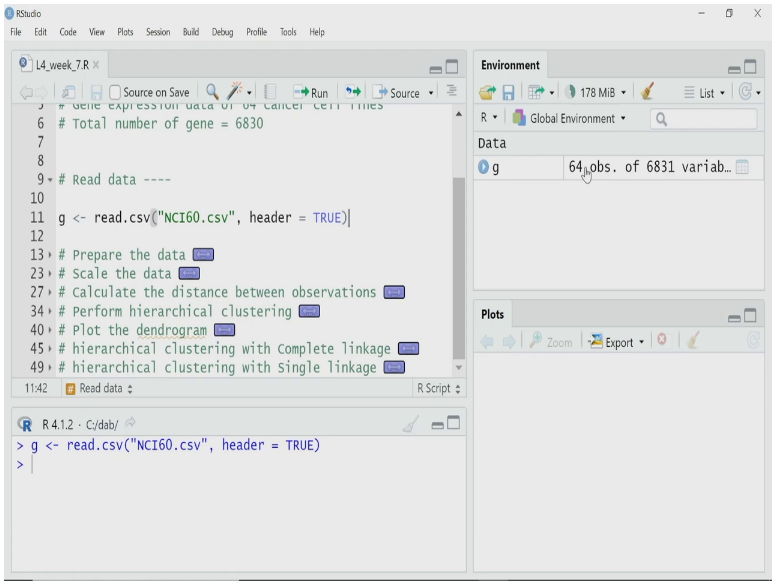
(Refer Slide Time: 1:14)



```
# Hierarchical clustering

# Data: NCI60 data set from ISLR2 package
# Gene expression data of 64 cancer cell lines
# Total number of gene = 6830


# Read data
# Prepare the data
# Scale the data
# Calculate the distance between observations
# Perform hierarchical clustering
# Plot the dendrogram
# hierarchical clustering with Complete linkage
```

So, as I said, the data set is called NCI 60 data set and it is taken from ISLR2 package. So, it has gene expression data for gene of 6830 genes from 64 cancer cell lines. So, I have stored it as a csv file in my machine. So, I will first read it, and then I will look into the data. And then I will process the data before I go into for hierarchical clustering.

(Refer Slide Time: 1:45)



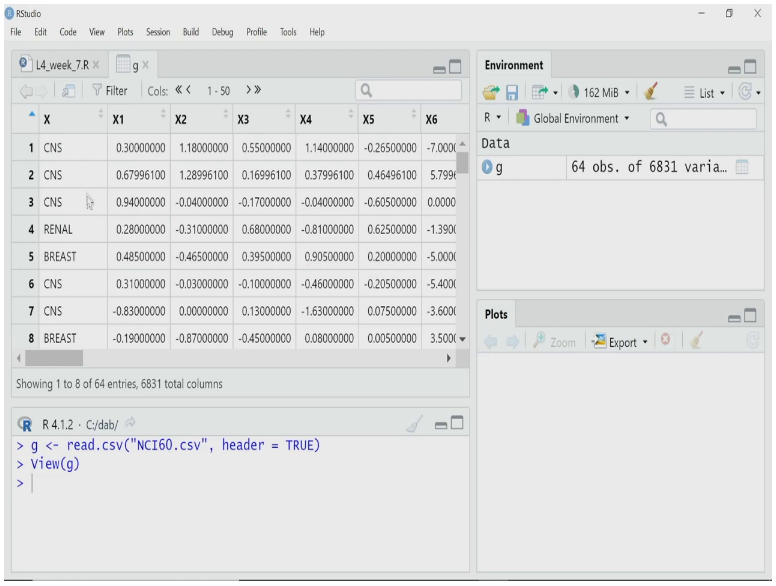
```
# Gene expression data of 64 cancer cell lines
# Total number of gene = 6830


# Read data ----

g <- read.csv("NCI60.csv", header = TRUE)

# Prepare the data
# Scale the data
# Calculate the distance between observations
# Perform hierarchical clustering
# Plot the dendrogram
# hierarchical clustering with Complete linkage
# hierarchical clustering with Single linkage
```

```
> g <- read.csv("NCI60.csv", header = TRUE)
>
```

g ← read.csv("NCI60.csv", header = TRUE)

So, let us first read the data using read dot csv, because that is how I have stored the data. It may take a few seconds, because it is a large data file. Let me open it, I have stored this as g.

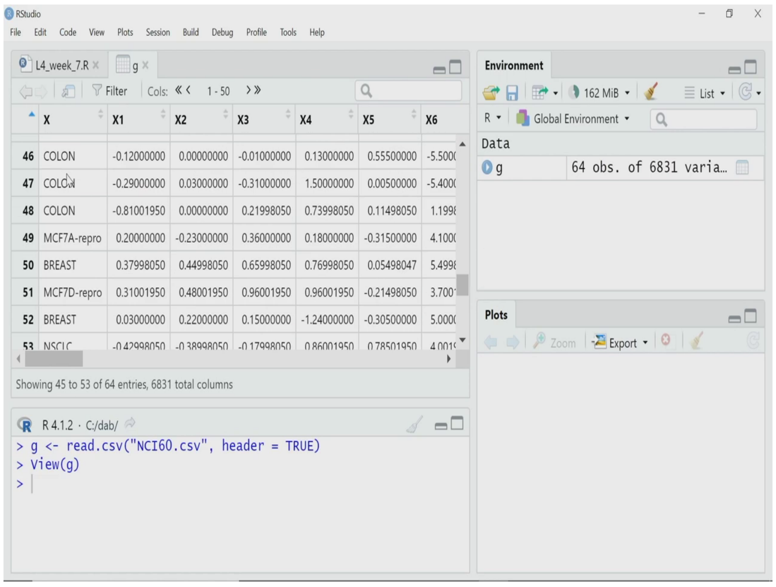
(Refer Slide Time: 1:55)



	X	X1	X2	X3	X4	X5	X6
1	CNS	0.30000000	1.18000000	0.55000000	1.14000000	-0.26500000	-7.000(
2	CNS	0.67996100	1.28996100	0.16996100	0.37996100	0.46496100	5.799(
3	CNS	0.94000000	-0.04000000	-0.17000000	-0.04000000	-0.60500000	0.0000
4	RENAL	0.28000000	-0.31000000	0.68000000	-0.81000000	0.62500000	-1.390(
5	BREAST	0.48500000	-0.46500000	0.39500000	0.90500000	0.20000000	-5.000(
6	CNS	0.31000000	-0.03000000	-0.10000000	-0.46000000	-0.20500000	-5.400(
7	CNS	-0.83000000	0.00000000	0.13000000	-1.63000000	0.07500000	-3.600(
8	BREAST	-0.19000000	-0.87000000	-0.45000000	0.08000000	0.00500000	3.500(

Showing 1 to 8 of 64 entries, 6831 total columns

```
> g <- read.csv("NCI60.csv", header = TRUE)
> View(g)
>
```



	X	X1	X2	X3	X4	X5	X6
46	COLON	-0.12000000	0.00000000	-0.01000000	0.13000000	0.55500000	-5.500
47	COLON	-0.29000000	0.03000000	-0.31000000	1.50000000	0.00500000	-5.400
48	COLON	-0.81001950	0.00000000	0.21998050	0.73998050	0.11498050	1.199
49	MCF7A-repro	0.20000000	-0.23000000	0.36000000	0.18000000	-0.31500000	4.100
50	BREAST	0.37998050	0.44998050	0.65998050	0.76998050	0.05498047	5.499
51	MCF7D-repro	0.31001950	0.48001950	0.96001950	0.96001950	-0.21498050	3.700
52	BREAST	0.03000000	0.22000000	0.15000000	-1.24000000	-0.30500000	5.000
53	NSCLC	-0.42998050	-0.38998050	-0.17998050	0.86001950	0.78501950	4.001

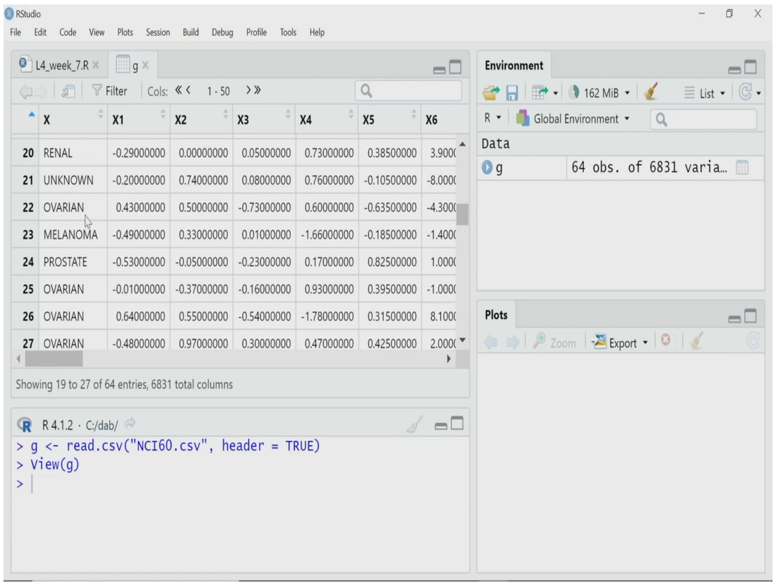
Showing 45 to 53 of 64 entries, 6831 total columns

```
> g <- read.csv("NCI60.csv", header = TRUE)
> View(g)
>
```

Data	
g	64 obs. of 6831 varia…

So, g is the variable where I am storing the data, as I said, is a bit a large file, so it may take a slight time delayed for loading. So, what do we have? I have 64 rows here, each of those are labels for, this first column is the label for each of these cell lines a CNS, CNS, Renal, Breast, these are the type of cell line, cancer cell and so.

(Refer Slide Time: 2:22)

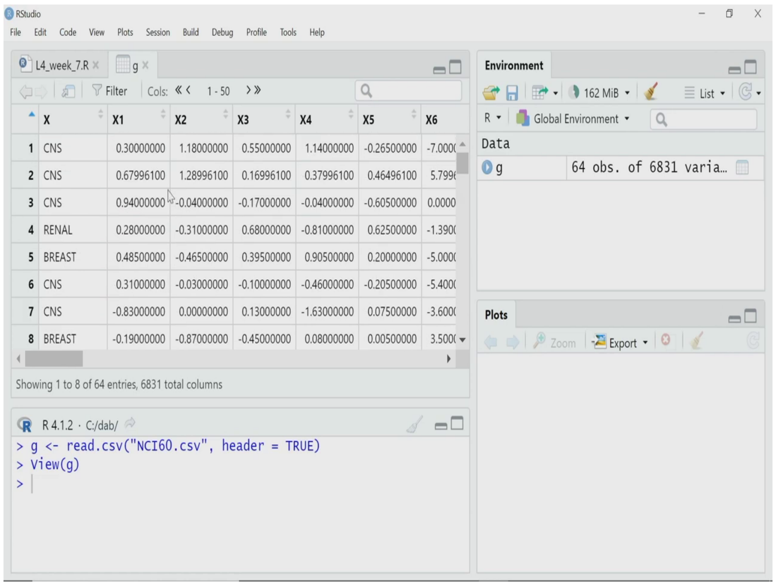


	X	X1	X2	X3	X4	X5	X6
20	RENAL	-0.29000000	0.00000000	0.05000000	0.73000000	0.38500000	3.900
21	UNKNOWN	-0.20000000	0.74000000	0.08000000	0.76000000	-0.10500000	-8.000
22	OVARIAN	0.43000000	0.50000000	-0.73000000	0.60000000	-0.63500000	-4.300
23	MELANOMA	-0.49000000	0.33000000	0.01000000	-1.66000000	-0.18500000	-1.400
24	PROSTATE	-0.53000000	-0.05000000	-0.23000000	0.17000000	0.82500000	1.000
25	OVARIAN	-0.01000000	-0.37000000	-0.16000000	0.93000000	0.39500000	-1.000
26	OVARIAN	0.64000000	0.55000000	-0.54000000	-1.78000000	0.31500000	8.100
27	OVARIAN	-0.48000000	0.97000000	0.30000000	0.47000000	0.42500000	2.000

Showing 19 to 27 of 64 entries, 6831 total columns

```
> g <- read.csv("NCI60.csv", header = TRUE)
> View(g)
>
```

Data	
g	64 obs. of 6831 varia…



	X	X1	X2	X3	X4	X5	X6
1	CNS	0.30000000	1.18000000	0.55000000	1.14000000	-0.26500000	-7.000
2	CNS	0.67996100	1.28996100	0.16996100	0.37996100	0.46496100	5.799
3	CNS	0.94000000	-0.04000000	-0.17000000	-0.04000000	-0.60500000	0.0000
4	RENAL	0.28000000	-0.31000000	0.68000000	-0.81000000	0.62500000	-1.390
5	BREAST	0.48500000	-0.46500000	0.39500000	0.90500000	0.20000000	-5.000
6	CNS	0.31000000	-0.03000000	-0.10000000	-0.46000000	-0.20500000	-5.400
7	CNS	-0.83000000	0.00000000	0.13000000	-1.63000000	0.07500000	-3.600
8	BREAST	-0.19000000	-0.87000000	-0.45000000	0.08000000	0.00500000	3.500

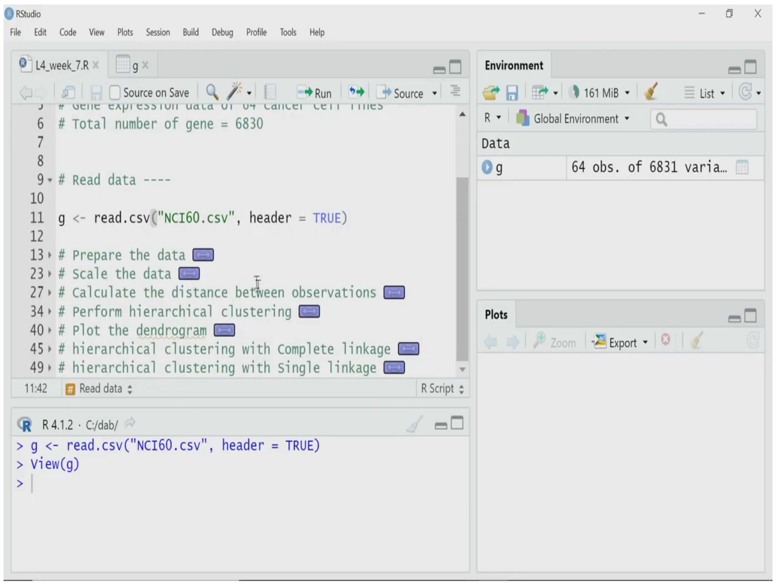
Showing 1 to 8 of 64 entries, 6831 total columns

```
> g <- read.csv("NCI60.csv", header = TRUE)
> View(g)
>
```

g 64 obs. of 6831 varia…

The fifth one is a breast cancer cell line. Whereas your the 22nd one is the ovarian cancer cell line like that. Whereas the other column, from the second one to up to the rest of the 6830 columns, those are for each of the genes for which the gene expression data has been measured. Now, the first thing that I have to do for this whole data, what I will do, I will separate this label column, because this has a label for the each of the cell type, label data I will separate out. And whereas I will keep this numeric data for gene expression as a separate part, so I will create two different variables and store them separately.

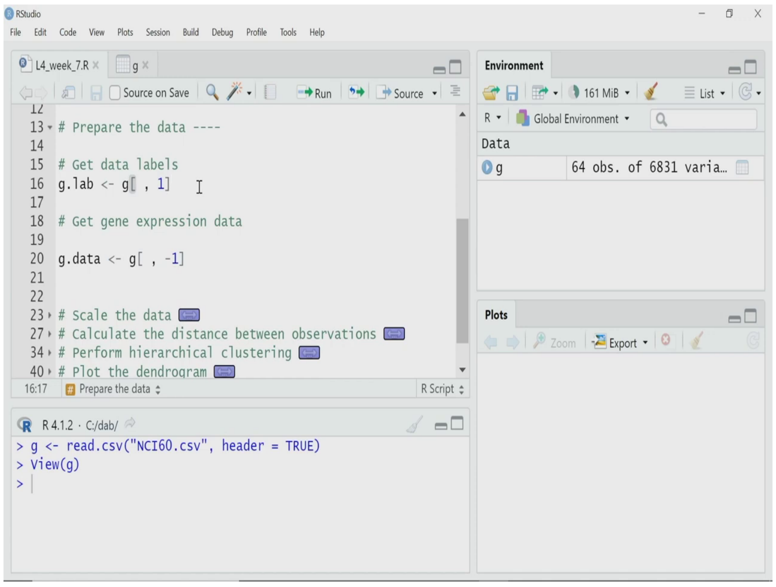
(Refer Slide Time: 3:00)



```
# Gene expression data of 64 cancer cell lines
# Total number of gene = 6830


# Read data ----

g <- read.csv("NCI60.csv", header = TRUE)

# Prepare the data
# Scale the data
# Calculate the distance between observations
# Perform hierarchical clustering
# Plot the dendrogram
# hierarchical clustering with Complete linkage
# hierarchical clustering with Single linkage
```

```
> g <- read.csv("NCI60.csv", header = TRUE)
> View(g)
>
```

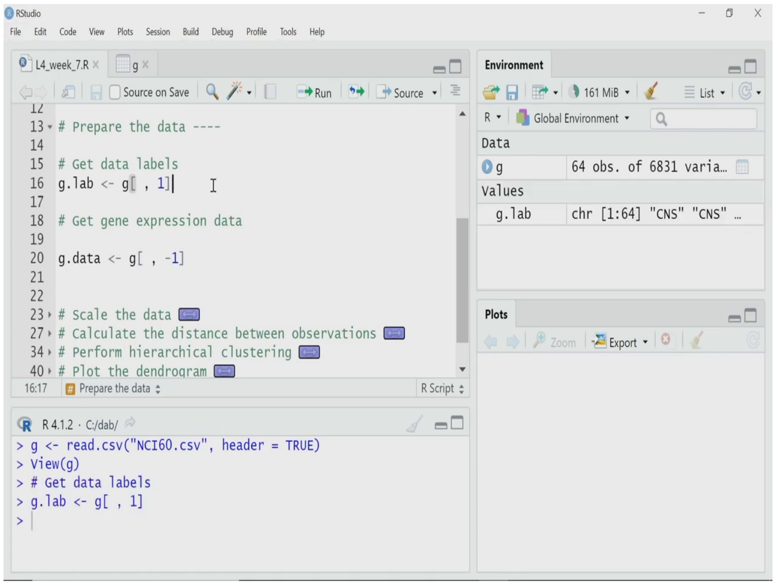
g 64 obs. of 6831 varia…



```
# Prepare the data ----

# Get data labels
g.lab <- g[ , 1]

# Get gene expression data

g.data <- g[ , -1]


# Scale the data
# Calculate the distance between observations
# Perform hierarchical clustering
# Plot the dendrogram
```

```
> g <- read.csv("NCI60.csv", header = TRUE)
> View(g)
>
```

```
Data
g        64 obs. of 6831 varia…
```



```
# Prepare the data ----

# Get data labels
g.lab <- g[ , 1]

# Get gene expression data

g.data <- g[ , -1]


# Scale the data
# Calculate the distance between observations
# Perform hierarchical clustering
# Plot the dendrogram
```

```
> g <- read.csv("NCI60.csv", header = TRUE)
> View(g)
> # Get data labels
> g.lab <- g[ , 1]
>
```

```
Data
g        64 obs. of 6831 varia…
Values
g.lab    chr [1:64] "CNS" "CNS" …
```

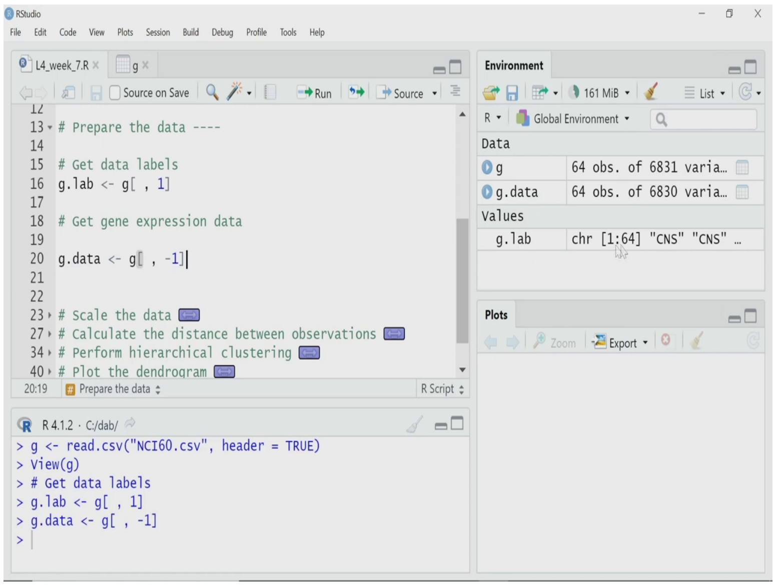
g,lab ← g[, 1]

g,data ← g[, -1]

So, to do that, what I am doing, I will first get the label data. So, g if you consider is a table or a data frame, or a matrix, whatever you imagine it so it has indexes, the row and column. So, I want the first column data. And I want all rows, so the index of row is kept blank. And so, g in the square bracket blank comma 1 that will fetch me all the row data for column 1, and those are the labels. And I am assigning that to g dot lab. So, that will store where that variable is told me the label.

Now, I will fetch the gene expression data. So, that means I want to start from the column 2 and rest of that. So, I can always write 2 colon and the final column number, but I will not do that, I have used it with a trick to discard the first column and fetch rest of the columns.

(Refer Slide Time: 3:58)



```
# Prepare the data ----

# Get data labels
g.lab <- g[ , 1]

# Get gene expression data

g.data <- g[ , -1]


# Scale the data
# Calculate the distance between observations
# Perform hierarchical clustering
# Plot the dendrogram
```

```
> g <- read.csv("NCI60.csv", header = TRUE)
> View(g)
> # Get data labels
> g.lab <- g[ , 1]
> g.data <- g[ , -1]
>
```

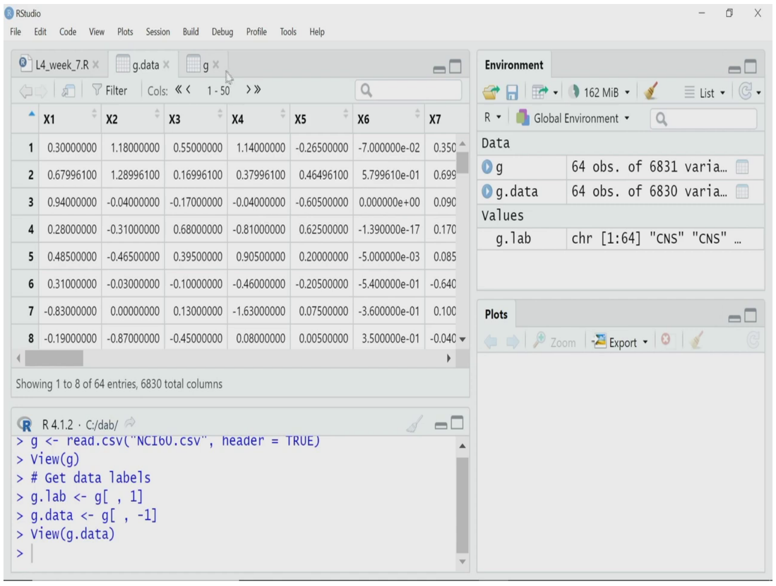
```
Data
g       64 obs. of 6831 varia…
g.data  64 obs. of 6830 varia…
Values
g.lab   chr [1:64] "CNS" "CNS" …
```

g.lab ← g[, 1]

g.data ← g[, -1]

So, what I have done within the square bracket obvious is that all the rows I want so I have leave the index for row empty and the index of column I have written minus 1 that will tell R that okay, you discard the column 1 and fetch rest of the columns. So, and then store that to g dot data. So, that will give you the numerical data for gene expression for all these 6830 genes in these 64 cell types. So, I have done that you can see you can check the g dot data, it will take some time to open.

(Refer Slide Time: 4:30)

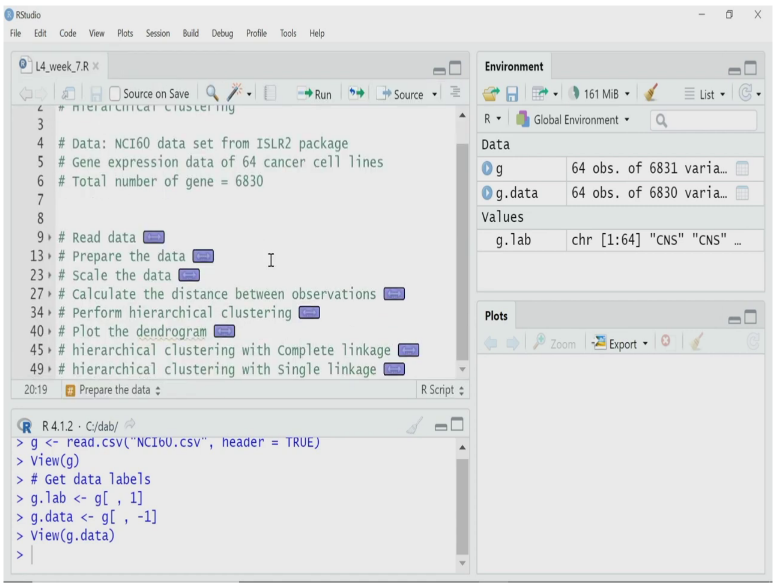


	X1	X2	X3	X4	X5	X6	X7
1	0.30000000	1.18000000	0.55000000	1.14000000	-0.26500000	-7.000000e-02	0.350
2	0.67996100	1.28996100	0.16996100	0.37996100	0.46496100	5.799610e-01	0.699
3	0.94000000	-0.04000000	-0.17000000	-0.04000000	-0.60500000	0.000000e+00	0.090
4	0.28000000	-0.31000000	0.68000000	-0.81000000	0.62500000	-1.390000e-17	0.170
5	0.48500000	-0.46500000	0.39500000	0.90500000	0.20000000	-5.000000e-03	0.085
6	0.31000000	-0.03000000	-0.10000000	-0.46000000	-0.20500000	-5.400000e-01	-0.640
7	-0.83000000	0.00000000	0.13000000	-1.63000000	0.07500000	-3.600000e-01	0.100
8	-0.19000000	-0.87000000	-0.45000000	0.08000000	0.00500000	3.500000e-01	-0.040

Showing 1 to 8 of 64 entries, 6830 total columns

```
> g <- read.csv("NCI60.csv", header = TRUE)
> View(g)
> # Get data labels
> g.lab <- g[ , 1]
> g.data <- g[ , -1]
> View(g.data)
>
```

```
Data
g       64 obs. of 6831 varia…
g.data  64 obs. of 6830 varia…
Values
g.lab   chr [1:64] "CNS" "CNS" …
```



```
# Hierarchical Clustering

# Data: NCI60 data set from ISLR2 package
# Gene expression data of 64 cancer cell lines
# Total number of gene = 6830


# Read data
# Prepare the data
# Scale the data
# Calculate the distance between observations
# Perform hierarchical clustering
# Plot the dendrogram
# hierarchical clustering with Complete linkage
# hierarchical clustering with Single linkage
```

```
Data
g        64 obs. of 6831 varia…
g.data   64 obs. of 6830 varia…
Values
g.lab    chr [1:64] "CNS" "CNS" …
```

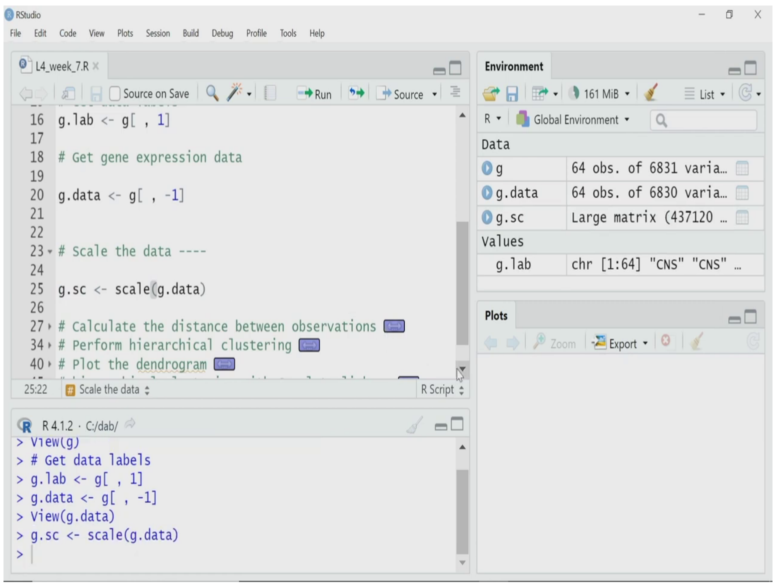
```
> g <- read.csv("NCI60.csv", header = TRUE)
> View(g)
> # Get data labels
> g.lab <- g[ , 1]
> g.data <- g[ , -1]
> View(g.data)
>
```

So, now you do not have label, you can imagine its a data matrix. So, I have fetched that data which I have to analyze. So, before you move for hierarchical clustering, it is wiser to scale the data. What do I mean by scaling data? By scaling data, see I have all these genes as columns, 6830 columns for each of these gene. Within the column, the data have variations. So, what do we do, we calculate the mean and the variance and the standard deviation for each column.

And then we transform the data for each column, the raw data of each column in such a way that the mean of each column will become 0. So, that is called centering of the data. And the standard deviation for each column will also become 1. So, I will get a centered scaled data. So, it is a good practice to do that, it may not be required always. But if your data dispersion gene has different dispersion level, and the mean may be quite large, the variation between means of different gene expression is very large, then your clustering algorithm may give some aberration.

So, that is why it is always better to scale the data, center and scale. So, what I will do, I will use the scale function. So, I will use that g dot data, which stored the numerical data for gene expression for all these genes to scale and store that at g dot sc.

(Refer Slide Time: 6:06)

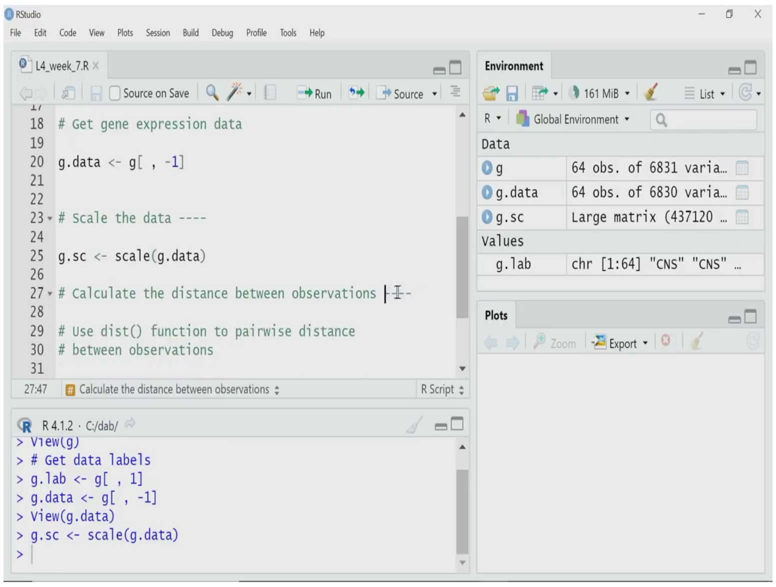


```
g.lab <- g[ , 1]

# Get gene expression data

g.data <- g[ , -1]


# Scale the data ----

g.sc <- scale(g.data)

# Calculate the distance between observations
# Perform hierarchical clustering
# Plot the dendrogram
```

25:22 Scale the data — R Script

```
> View(g)
> # Get data labels
> g.lab <- g[ , 1]
> g.data <- g[ , -1]
> View(g.data)
> g.sc <- scale(g.data)
>
```

```
Data
g        64 obs. of 6831 varia…
g.data   64 obs. of 6830 varia…
g.sc     Large matrix (437120 …
Values
g.lab    chr [1:64] "CNS" "CNS" …
```



```
# Get gene expression data

g.data <- g[ , -1]


# Scale the data ----

g.sc <- scale(g.data)

# Calculate the distance between observations ----

# Use dist() function to pairwise distance
# between observations

```

27:47 Calculate the distance between observations — R Script

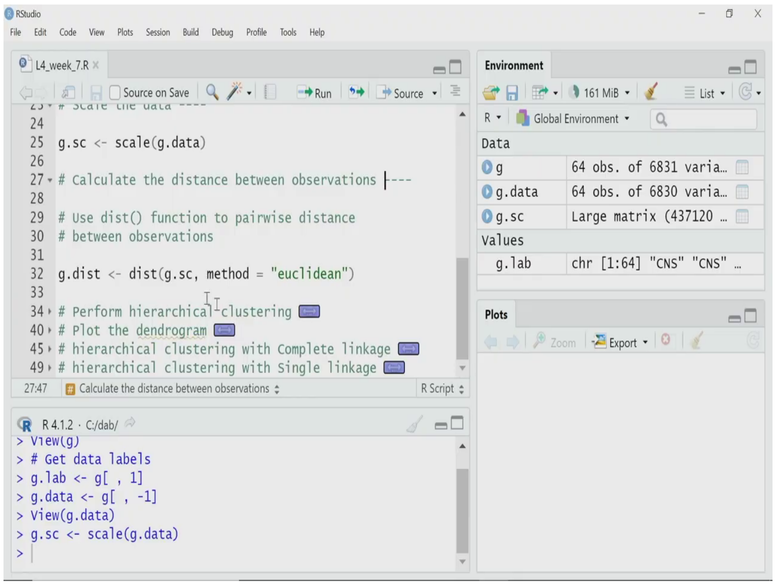
```
> View(g)
> # Get data labels
> g.lab <- g[ , 1]
> g.data <- g[ , -1]
> View(g.data)
> g.sc <- scale(g.data)
>
```

```
Data
g        64 obs. of 6831 varia…
g.data   64 obs. of 6830 varia…
g.sc     Large matrix (437120 …
Values
g.lab    chr [1:64] "CNS" "CNS" …
```

g.sc ← scale(g.data)

So, now I have the scaled data. So, I will perform clustering not on the row data, but on the scaled data. So, before I move forward move for clustering, I have to calculate the distance pairwise distance between each of these cells, each of these cells is observation. So, for each pair of the cell type, we have to calculate the pairwise distance.

(Refer Slide Time: 6:27)



```
# Scale the data ----

g.sc <- scale(g.data)

# Calculate the distance between observations ----

# Use dist() function to pairwise distance
# between observations

g.dist <- dist(g.sc, method = "euclidean")

# Perform hierarchical clustering
# Plot the dendrogram
# hierarchical clustering with Complete linkage
# hierarchical clustering with Single linkage
```

```
27:47   Calculate the distance between observations     R Script
```

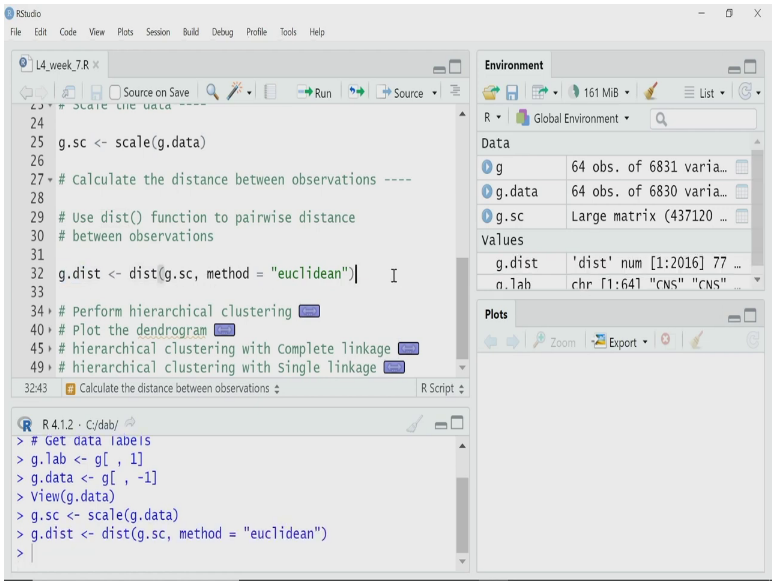
```
Data
g          64 obs. of 6831 varia...
g.data     64 obs. of 6830 varia...
g.sc       Large matrix (437120 ...
Values
g.lab      chr [1:64] "CNS" "CNS" ...
```

```
> View(g)
> # Get data labels
> g.lab <- g[ , 1]
> g.data <- g[ , -1]
> View(g.data)
> g.sc <- scale(g.data)
>
```

g.dist ← dist(g.sc, method = “euclidean”)

So, how do I do I use the distance function to calculate the pairwise distance between these 64 cell types. So, I use the scale data g.sc. And the method of distance if you remember, there are many distance metrics, we have learned few of them. So, here I am using the Euclidean. So, I am telling you calculate the distance between the cell types using the Euclidean distance as a metric. So, I am writing method equal to Euclidean and that distance data is stored in g dot dist.

(Refer Slide Time: 6:58)

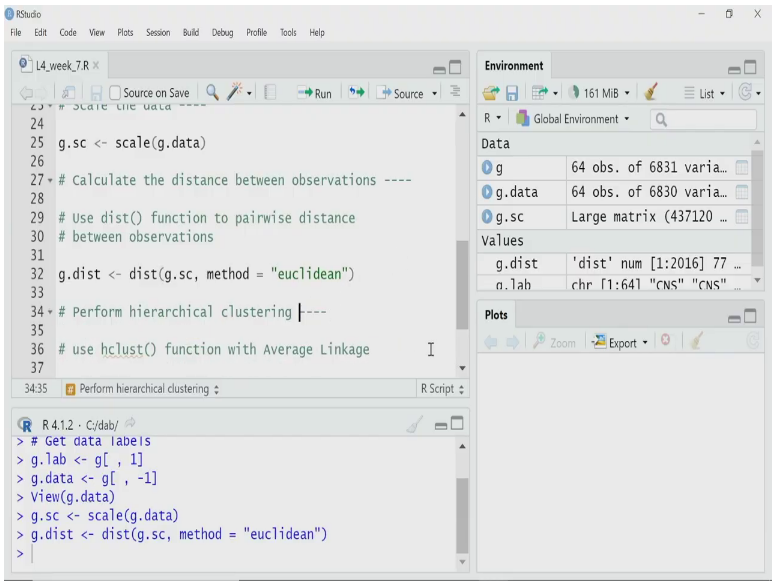


```
# Scale the data ----

g.sc <- scale(g.data)

# Calculate the distance between observations ----

# Use dist() function to pairwise distance
# between observations

g.dist <- dist(g.sc, method = "euclidean")

# Perform hierarchical clustering
# Plot the dendrogram
# hierarchical clustering with Complete linkage
# hierarchical clustering with Single linkage
```

```
32:43   Calculate the distance between observations     R Script
```

```
Data
g          64 obs. of 6831 varia...
g.data     64 obs. of 6830 varia...
g.sc       Large matrix (437120 ...
Values
g.dist     'dist' num [1:2016] 77 ...
g.lab      chr [1:64] "CNS" "CNS" ...
```

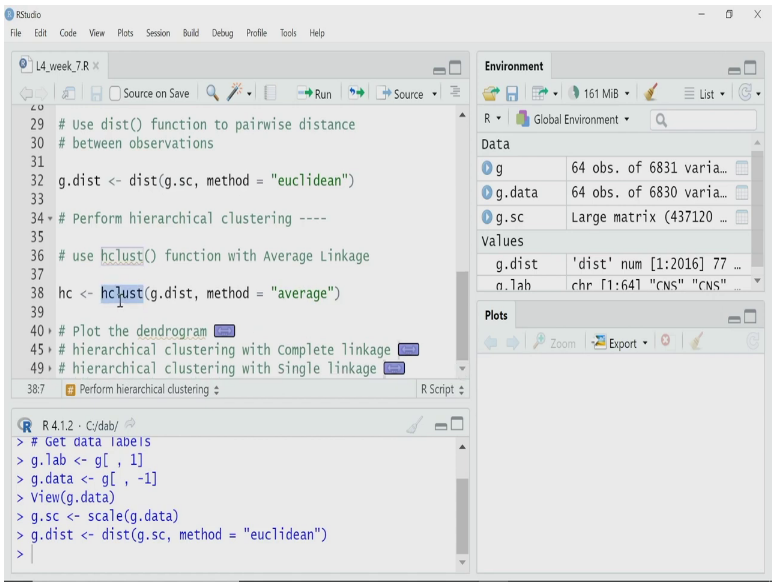
```
> # Get data labels
> g.lab <- g[ , 1]
> g.data <- g[ , -1]
> View(g.data)
> g.sc <- scale(g.data)
> g.dist <- dist(g.sc, method = "euclidean")
>
```



```
# scale the data ----

g.sc <- scale(g.data)

# Calculate the distance between observations ----

# Use dist() function to pairwise distance
# between observations

g.dist <- dist(g.sc, method = "euclidean")

# Perform hierarchical clustering ----

# use hclust() function with Average Linkage
```

```
> # Get data labels
> g.lab <- g[ , 1]
> g.data <- g[ , -1]
> View(g.data)
> g.sc <- scale(g.data)
> g.dist <- dist(g.sc, method = "euclidean")
>
```

```
Data
g        64 obs. of 6831 varia…
g.data   64 obs. of 6830 varia…
g.sc     Large matrix (437120 …
Values
g.dist   'dist' num [1:2016] 77 …
g.lab    chr [1:64] "CNS" "CNS" …
```



```
# Use dist() function to pairwise distance
# between observations

g.dist <- dist(g.sc, method = "euclidean")

# Perform hierarchical clustering ----

# use hclust() function with Average Linkage

hc <- hclust(g.dist, method = "average")

# Plot the dendrogram
# hierarchical clustering with Complete linkage
# hierarchical clustering with Single linkage
```

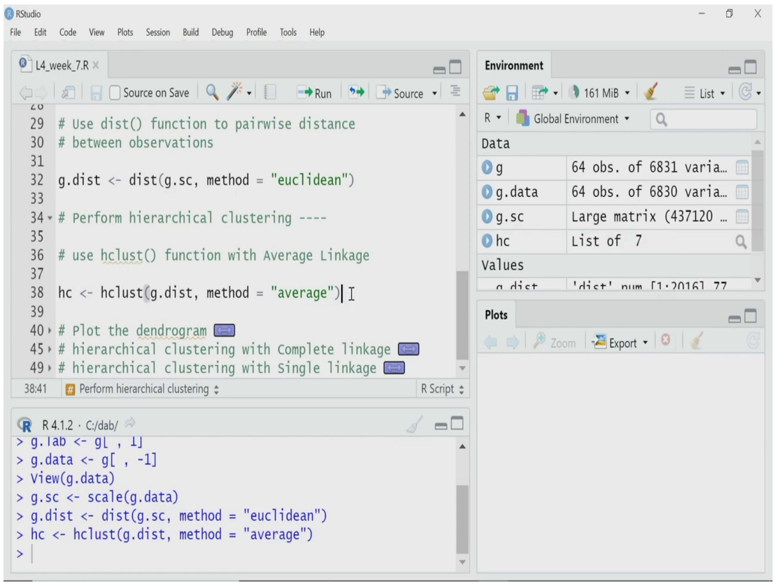
```
> # Get data labels
> g.lab <- g[ , 1]
> g.data <- g[ , -1]
> View(g.data)
> g.sc <- scale(g.data)
> g.dist <- dist(g.sc, method = "euclidean")
>
```

g.dist ← dist(g.sc, method = “euclidean”)

So, now I am ready. I have scaled data from that data, I have calculated the distance pairwise distance. So, now I will perform hierarchical clustering. So, to perform hierarchical clustering, I will use h clust the default h clust function, way of R. Now obviously, the distance measure here will be the arguments for this function, not the original scale data. So, g dot dist will be used as a input as an argument for this function. And if you remember, there are broadly 3 methods of calculating distance between clusters.

One is called complete linkage, one is called single linkage, and other one is the average one. So, what I am doing here in method, I am specifying that use average linkage as a method of calculating distance between clusters. So, I am using h clust with average linkage, and I will store the output of this clustering to the variable hc.

(Refer Slide Time: 8:02)



```
# Use dist() function to pairwise distance
# between observations

g.dist <- dist(g.sc, method = "euclidean")

# Perform hierarchical clustering ----

# use hclust() function with Average Linkage

hc <- hclust(g.dist, method = "average")

# Plot the dendrogram
# hierarchical clustering with Complete linkage
# hierarchical clustering with Single linkage
```

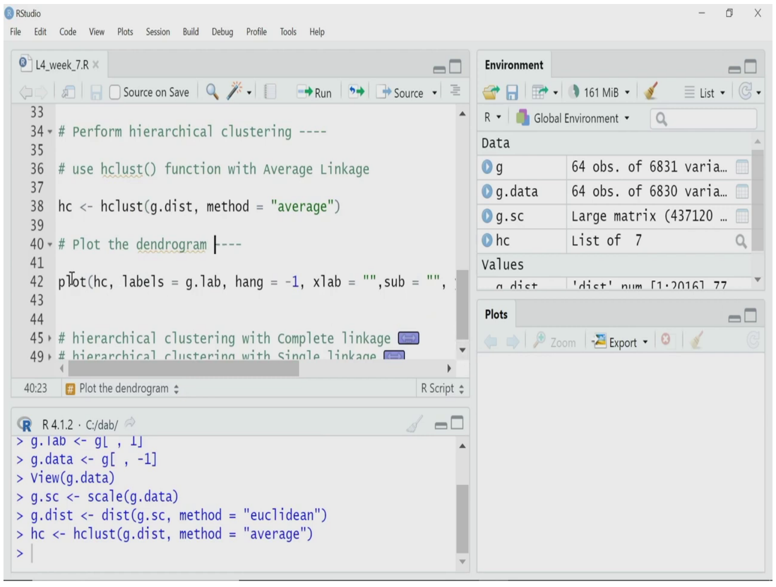
```
> g.lab <- g[ , 1]
> g.data <- g[ , -1]
> View(g.data)
> g.sc <- scale(g.data)
> g.dist <- dist(g.sc, method = "euclidean")
> hc <- hclust(g.dist, method = "average")
>
```

```
Data
g        64 obs. of 6831 varia…
g.data   64 obs. of 6830 varia…
g.sc     Large matrix (437120 …
hc       List of  7
Values
g.dist   'dist' num [1:2016] 77
```

hc ← hclust(g.dist, method = “average”)

I perform, hc has all the information, but I will not dig into that I will go for plotting the dendrogram. If you remember, the hierarchical clustering will eventually give me a tree showing the relation between different clusters.

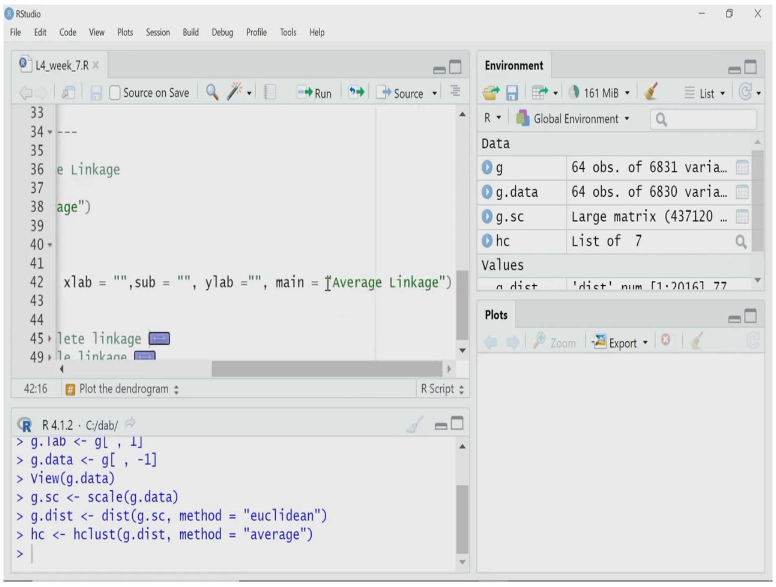
(Refer Slide Time: 8:19)



```
# Perform hierarchical clustering ----

# use hclust() function with Average Linkage

hc <- hclust(g.dist, method = "average")

# Plot the dendrogram ----

plot(hc, labels = g.lab, hang = -1, xlab = "",sub = "",


# hierarchical clustering with Complete linkage
# hierarchical clustering with Single linkage
```

```
> g.lab <- g[ , 1]
> g.data <- g[ , -1]
> View(g.data)
> g.sc <- scale(g.data)
> g.dist <- dist(g.sc, method = "euclidean")
> hc <- hclust(g.dist, method = "average")
>
```

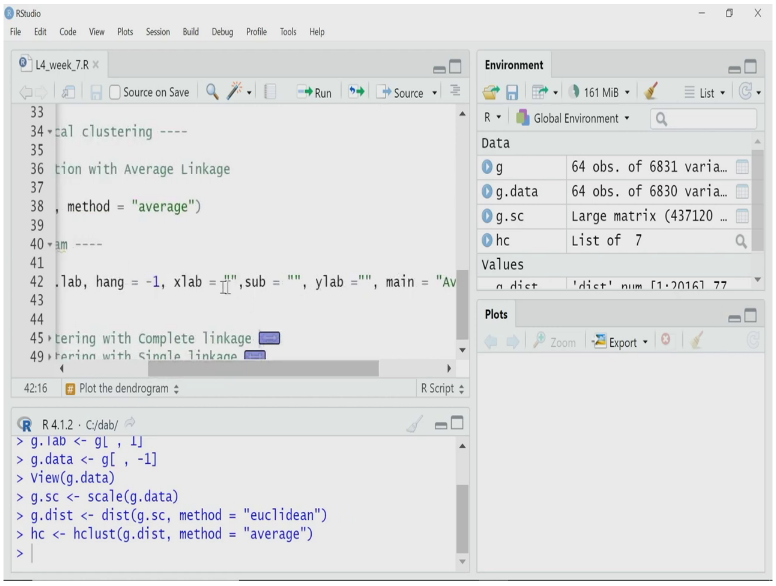
```
Data
g        64 obs. of 6831 varia…
g.data   64 obs. of 6830 varia…
g.sc     Large matrix (437120 …
hc       List of  7
Values
g.dist   'dist' num [1:2016] 77
```



```
---
e Linkage
age")
 xlab = "",sub = "", ylab ="", main = "Average Linkage")
lete linkage
le linkage
```

```
> g.lab <- g[ , 1]
> g.data <- g[ , -1]
> View(g.data)
> g.sc <- scale(g.data)
> g.dist <- dist(g.sc, method = "euclidean")
> hc <- hclust(g.dist, method = "average")
>
```

```
Data
g        64 obs. of 6831 varia…
g.data   64 obs. of 6830 varia…
g.sc     Large matrix (437120 …
hc       List of  7
Values
g.dist   'dist' num [1:2016] 77…
```



```
cal clustering ----
tion with Average Linkage
, method = "average")
am ----
.lab, hang = -1, xlab = "",sub = "", ylab ="", main = "Av
tering with Complete linkage
tering with Single linkage
```

```
> g.lab <- g[ , 1]
> g.data <- g[ , -1]
> View(g.data)
> g.sc <- scale(g.data)
> g.dist <- dist(g.sc, method = "euclidean")
> hc <- hclust(g.dist, method = "average")
>
```

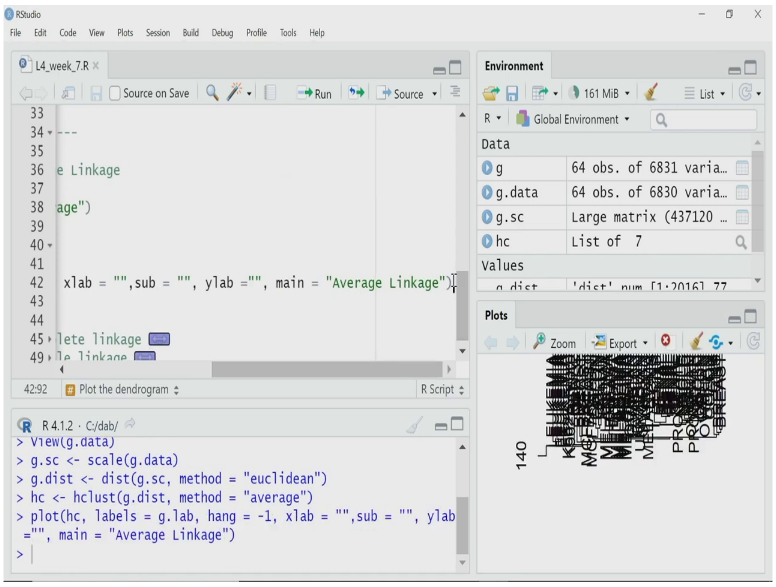
```
Data
g        64 obs. of 6831 varia…
g.data   64 obs. of 6830 varia…
g.sc     Large matrix (437120 …
hc       List of  7
Values
g.dist   'dist' num [1:2016] 77…
```

plot(hc, labels = g.lab, hang = -1, xlab = "", sub = "", ylab = "", main = "Average Linkage")

So, I will plot the dendrogram. To plot the dendrogram. I will use the plot function, it is quite versatile and powerful. So, the input will have all the argument the first argument will be hc that is the output coming from my h cluster function. And what I will do I have to label the leaves, if you remember, a tree will have leaves. So, each of these leaves should be labelled. What will be the label? Label will be the name of the cell line. And I have stored that information in g.lab, I created that few lines back.

So, I will use that information that label information present in my original data set as labels for these leaves. I will put on the top I will name of the dendrogram is average linkage. I do not want any horizontal axis vertical axis label. I do not want the subject as also I do not want to print the subject. So, I keep those empty, this is interesting, I put hang equal to minus 1. So, in that case, what will happen in dendrogram, all the leaves will be on the same line same horizontal line.

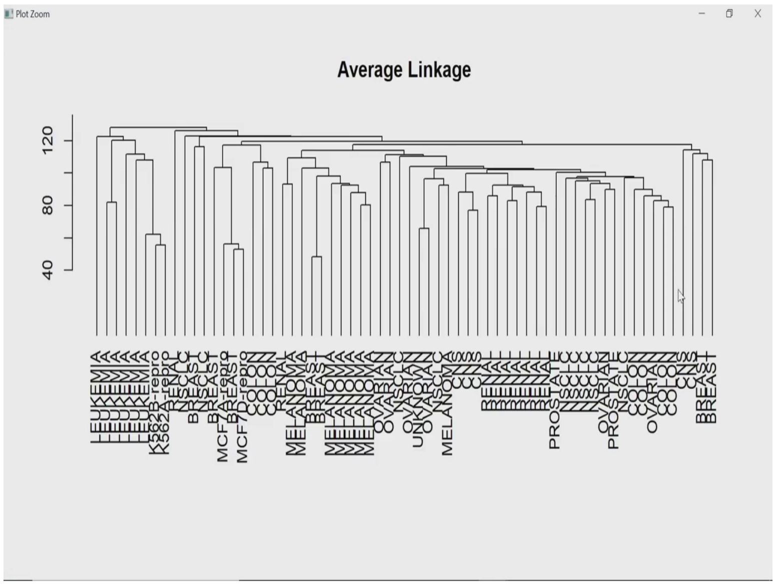
(Refer Slide Time: 9:33)



```
---

e Linkage

age")

 xlab = "",sub = "", ylab ="", main = "Average Linkage")


lete linkage
le linkage
```

```
Data
g        64 obs. of 6831 varia…
g.data   64 obs. of 6830 varia…
g.sc     Large matrix (437120 …
hc       List of  7
Values
```

```
> View(g.data)
> g.sc <- scale(g.data)
> g.dist <- dist(g.sc, method = "euclidean")
> hc <- hclust(g.dist, method = "average")
> plot(hc, labels = g.lab, hang = -1, xlab = "",sub = "", ylab
 ="", main = "Average Linkage")
>
```



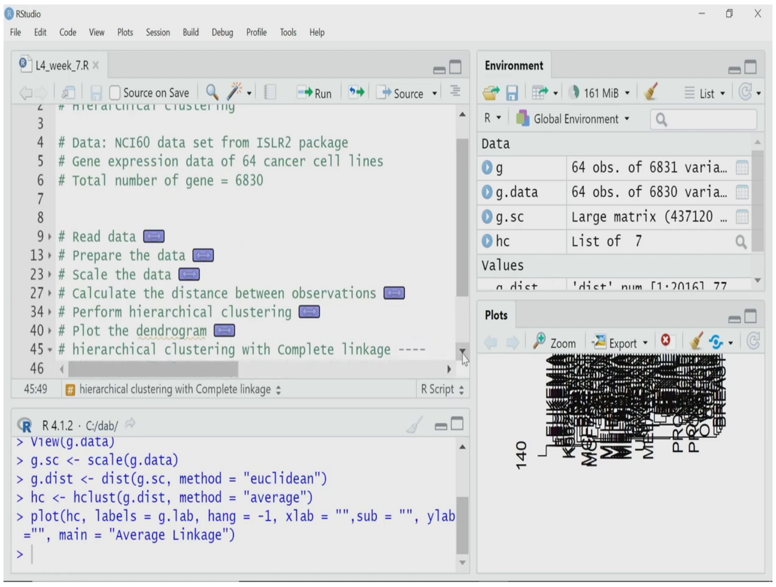
plot(hc, labels = g.lab, hang = -1, xlab = “”, sub = “”, ylab = “”, main = “Average Linkage”)

So, I will execute this plot function. Let me zoom it, make it full screen. So, this is my dendrogram or tree of hierarchical clustering for this gene expression data set. And we have done average linkage to calculate the distance between 2 clusters. Now let us look into it. On the left-hand side, I have K562B, K562A these 2 are leukemia cell line, they form one clusters. And then that cluster fuse with another leukemia cell line. And you can see all these are on the left-hand side are all leukemia cell line, and they are all clustered fused together.

And that is what we want we expect that is not it. Similarly, here, if you see you have the 2-breast cancer cell line, they have formed a cluster whereas, here, we have a stretch of long stretch of renal cancer cell line, which are forming cluster and those clusters again fusing to get bigger clusters. So, that seems that there is a hierarchical clustering using average linkage has worked pretty well for this particular data set. And now, within this I can see that the gene expression data, I can conclude the gene expression data can actually segregate different cell types, based upon the gene expression data, I could segregate these different cell types in different clusters.

So, that was also the purpose of performing the gene expression analysis. Now, I have performed this using average linkage. What about complete linkage and single linkage? So, I will perform the same thing using complete linkage and in also single linkage.

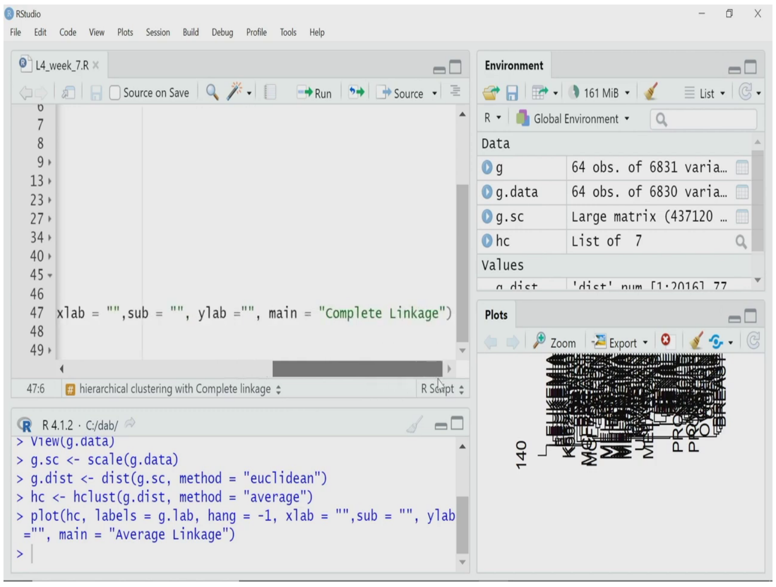
(Refer Slide Time: 11:16)



```
# Hierarchical Clustering

# Data: NCI60 data set from ISLR2 package
# Gene expression data of 64 cancer cell lines
# Total number of gene = 6830


# Read data
# Prepare the data
# Scale the data
# Calculate the distance between observations
# Perform hierarchical clustering
# Plot the dendrogram
# hierarchical clustering with Complete linkage ----
```

```
> View(g.data)
> g.sc <- scale(g.data)
> g.dist <- dist(g.sc, method = "euclidean")
> hc <- hclust(g.dist, method = "average")
> plot(hc, labels = g.lab, hang = -1, xlab = "",sub = "", ylab
 ="", main = "Average Linkage")
>
```

Data	
g	64 obs. of 6831 varia…
g.data	64 obs. of 6830 varia…
g.sc	Large matrix (437120 …
hc	List of 7



```
xlab = "",sub = "", ylab ="", main = "Complete Linkage")
```

```
> View(g.data)
> g.sc <- scale(g.data)
> g.dist <- dist(g.sc, method = "euclidean")
> hc <- hclust(g.dist, method = "average")
> plot(hc, labels = g.lab, hang = -1, xlab = "",sub = "", ylab
 ="", main = "Average Linkage")
>
```

```
Data
g        64 obs. of 6831 varia…
g.data   64 obs. of 6830 varia…
g.sc     Large matrix (437120 …
hc       List of  7
Values
```

plot(hclust(g.dist, method = “complete”),

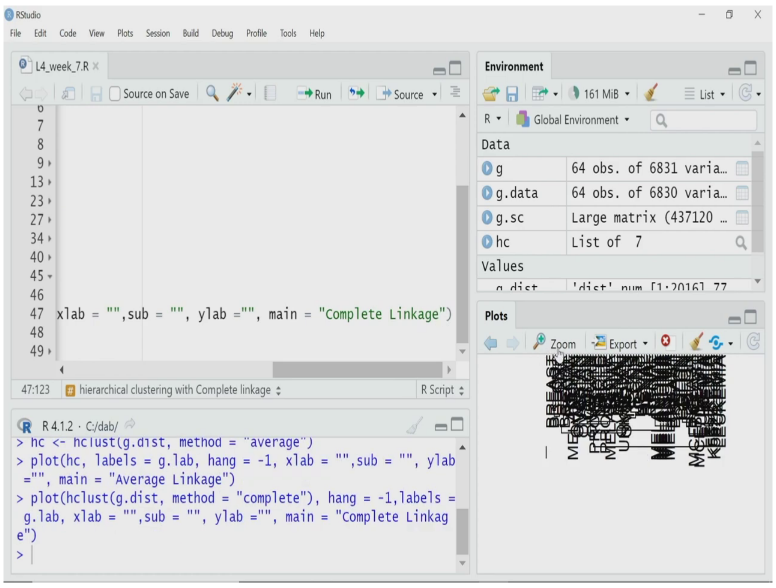
labels = g.lab, hang = -1, xlab = “”, sub = “”, ylab = “”,

main = “Complete Linkage”)

If you remember in complete linkage what we are doing in contrast to average linkage, we will not be taking the distance between the average distance between 2 cluster as a distance between clusters. What we will do? We will calculate the pairwise distance between 2 cluster and I will take the maximum one whereas in the single linkage method, we will take the minimum pairwise distance.

So, I will perform first the complete linkage algorithm, so, I will not separately perform it while plotting I am performing it. So, I am calling the plot function and the first input is I will perform the h clust hierarchical clustering using h clust where the data will be the distance data and the method will be complete. So, this part will be executed before plotting the data and the rest of the thing remains same except the title of the plot I have changed to complete linkage.

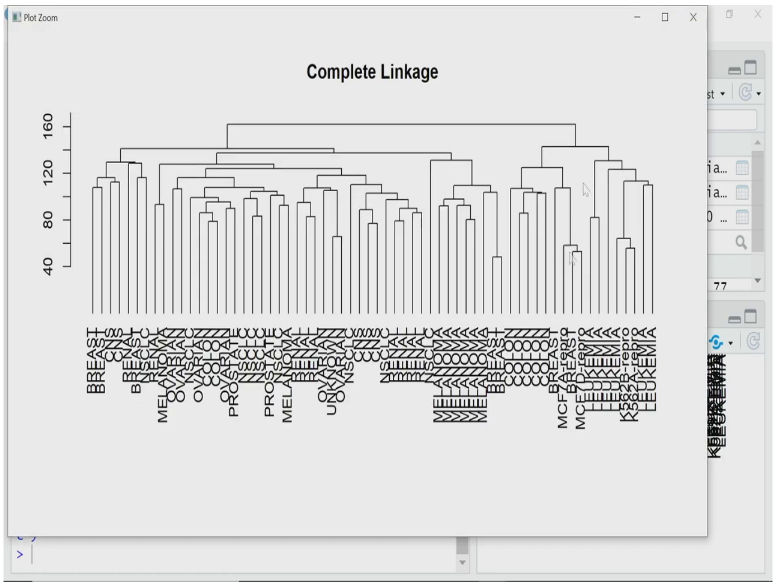
(Refer Slide Time: 12:14)



```
xlab = "",sub = "", ylab ="", main = "Complete Linkage")
```

```
g        64 obs. of 6831 varia…
g.data   64 obs. of 6830 varia…
g.sc     Large matrix (437120 …
hc       List of  7
```

```
> hc <- hclust(g.dist, method = "average")
> plot(hc, labels = g.lab, hang = -1, xlab = "",sub = "", ylab
="", main = "Average Linkage")
> plot(hclust(g.dist, method = "complete"), hang = -1,labels =
 g.lab, xlab = "",sub = "", ylab ="", main = "Complete Linkag
e")
>
```



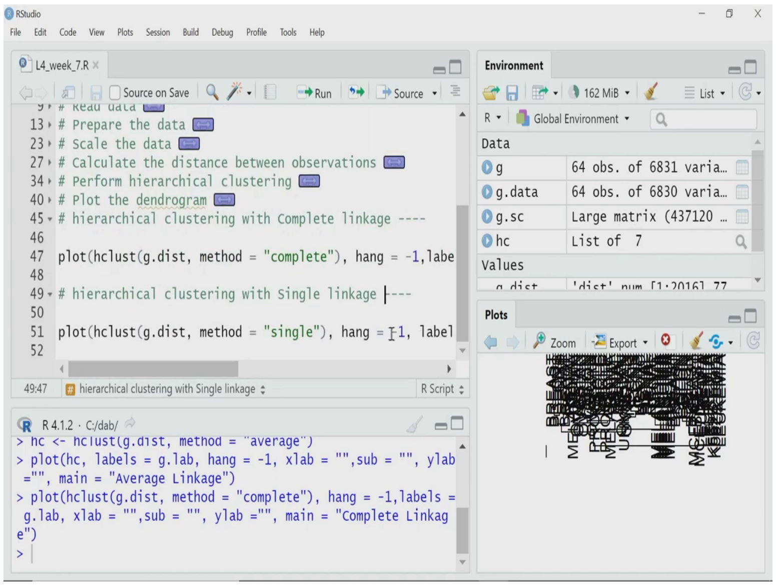
plot(hclust(g.dist, method = “complete”),

labels = g.lab, hang = -1, xlab = “”, sub = “”, ylab = “”,

main = “Complete Linkage”)

So, again, I will zoom make it full screen. Here also you can see nicely for example, this colon cancer cell all are stuck together melanoma are all stick together, leukemia is again cluster on the right-hand side. Obviously, this complete linkage dendrogram is quite different from what we got in the average linkage and that is quite expected, but broadly both complete linkage and single linkage has given me separation or clustering of cell type based upon the gene expression, same cell types are in same cluster. So, that way both complete linkage and average linkage is working for us in this particular data set.

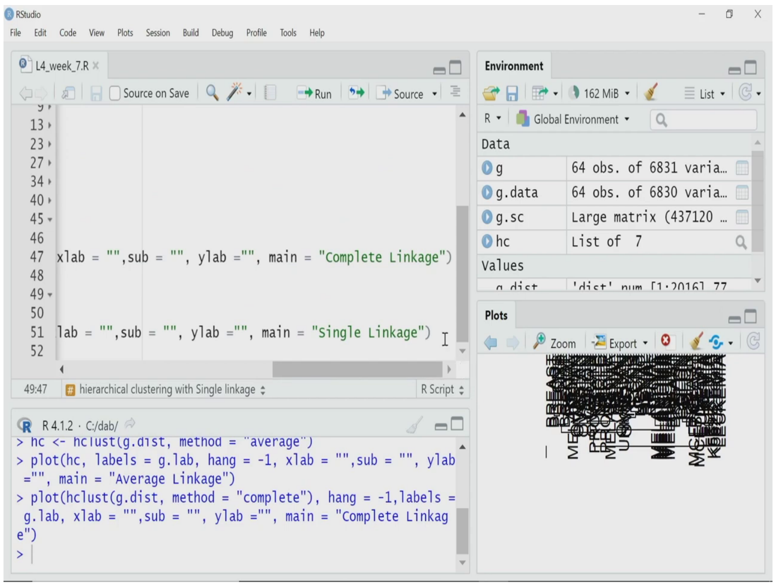
(Refer Slide Time: 13:03)



```
# Read data
# Prepare the data
# Scale the data
# Calculate the distance between observations
# Perform hierarchical clustering
# Plot the dendrogram
# hierarchical clustering with Complete linkage ----

plot(hclust(g.dist, method = "complete"), hang = -1,labe

# hierarchical clustering with Single linkage ----

plot(hclust(g.dist, method = "single"), hang = -1, label
```

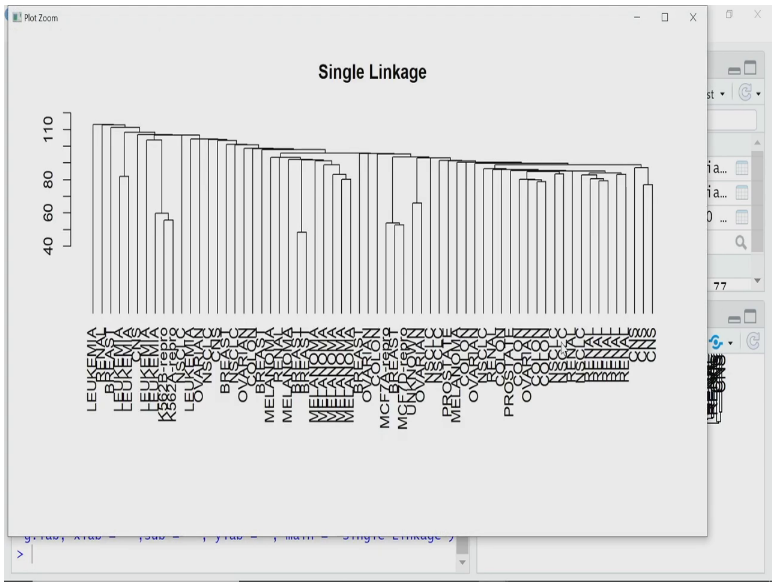
```
> hc <- hclust(g.dist, method = "average")
> plot(hc, labels = g.lab, hang = -1, xlab = "",sub = "", ylab
 ="", main = "Average Linkage")
> plot(hclust(g.dist, method = "complete"), hang = -1,labels =
 g.lab, xlab = "",sub = "", ylab ="", main = "Complete Linkag
e")
>
```

```
Data
g        64 obs. of 6831 varia…
g.data   64 obs. of 6830 varia…
g.sc     Large matrix (437120 …
hc       List of 7
Values
g.dist   'dist' num [1:2016] 77…
```



```
xlab = "",sub = "", ylab ="", main = "Complete Linkage")

lab = "",sub = "", ylab ="", main = "Single Linkage")
```

```
> hc <- hclust(g.dist, method = "average")
> plot(hc, labels = g.lab, hang = -1, xlab = "",sub = "", ylab
 ="", main = "Average Linkage")
> plot(hclust(g.dist, method = "complete"), hang = -1,labels =
 g.lab, xlab = "",sub = "", ylab ="", main = "Complete Linkag
e")
>
```



plot(hclust(g.dist, method = "single"),

labels = g.lab, hang = -1, xlab = "", sub = "", ylab = "",

main = "Single Linkage")

Now, let us perform the same thing with single linkage. Again, I will not perform the clustering separately while plotting I will do that let me see, now, we have some trouble. In some cases for example melanoma they are close they are in a tight cluster. But you can see lots of mix up has happened and what we have achieved we got a dendrogram which has something called trailing in a thing, you get a long chain of the cluster getting fused with one another. That is quite an aberration of clustering.

And we discussed this problem while discussing hierarchical clustering in our previous lecture, and that is why single linkage is usually not the preferred choice of distance measure while performing hierarchical clustering. Just like here we have seen the good dendrogram we have got a very balanced dendrogram when we are using average linkage and complete linkage for our data set most of the time, people try to use average linkage or complete linkage to create a more balanced dendrogram.

And in this case also I will go for either the average linkage or for the complete linkage. So, that brings me to the end of this lecture, what we have seen in this lecture is that we have performed hierarchical clustering using h clust the default function that we have and we have created the dendrogram. In this lecture, we have shown that all these three methods you can use you know all these three linkages average, complete and single, but as I said, it is better to use average linkage or complete linkage to get a better dendrogram.

You can extract the data for each of these clusters from these also from this hierarchical clustering also, I will not go into them they will be similar to how we have worked earlier for data extraction. So, that is all for this lecture. Thank you for learning with me today.