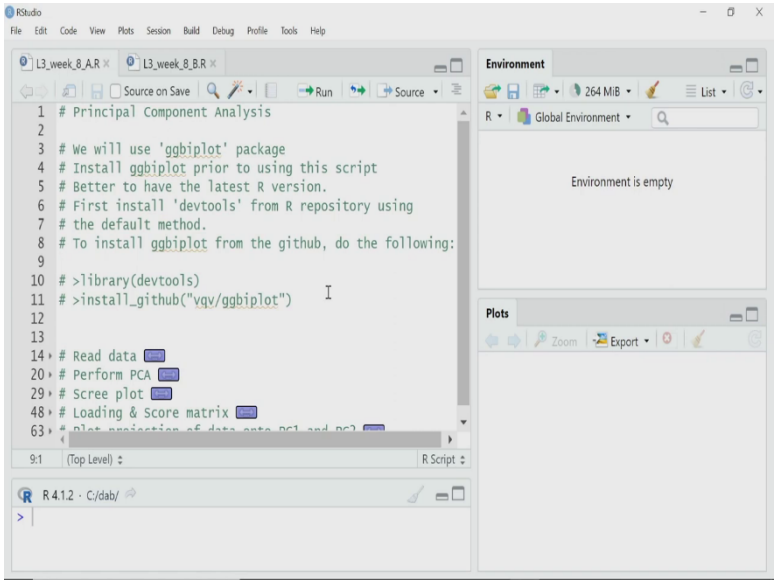
Data Analysis for Biologists
Professor Biplab Bose
Department of Biosciences & Bioengineering
Mehta Family School of Data Science and Artificial Intelligence
Indian Institute of Technology, Guwahati
Lecture: 46
Principal Component Analysis Using R

Hello everyone, welcome back. In this lecture, I will demonstrate how to use R to perform principal component analysis. As we have studied earlier, principal component analysis is a method to reduce the dimension of a data. Just to remind you, suppose you are doing an experiment where you have hundreds and thousands of variables, for example, you may be doing a gene expression studies, for different tumour samples.

So, you have hundreds of genes and their gene expression data. So, the dimension of your data is 100 in scale of 100, neither you can visualize that data in 2 dimension and also sometime it become very difficult to analyze that data, that such higher dimensional data. So, that is why we use principal component analysis to reduce the dimensionality of the problem.

So, today I will demonstrate PCA for 2 problems. The first 1 is a synthetic data, I just cooked up the data a small data set, is not really a very high dimension but that will allow us to understand how we will perform PCA for a particular data set and the tricks of that. And then I will move into a large data set which has really large dimension.

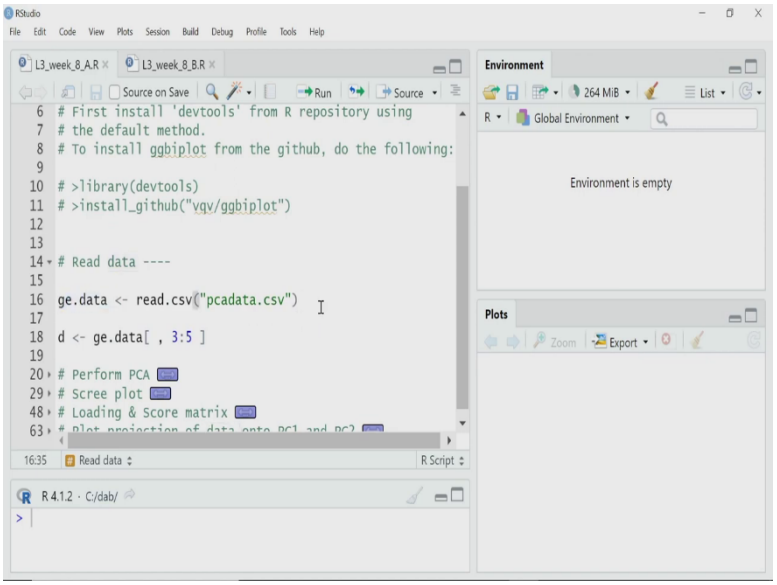
(Refer Slide Time: 1:47)



So, to perform PCA, I will use a inbuilt function PCA in R, but at the same time to visualize the PCA result, we will use a package called ggbiplot. So, here in the script, here which I will

say share with everyone these instructions are written how to install ggbiplot from github. So, before you start using this script and practice this problem, please install ggbiplot in your machine.

(Refer Slide Time: 2:24)

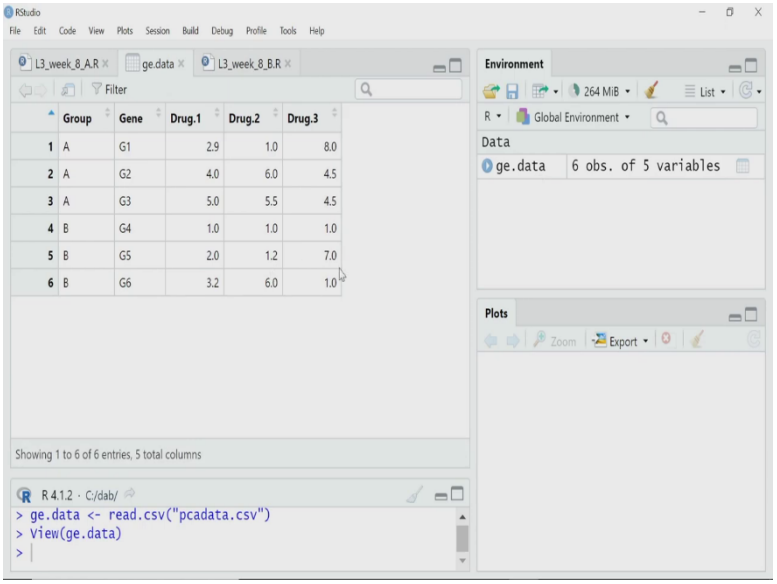


ge.data ← read.csv("pcdata.csv")

d ← ge.data[, 2:5]

So, the first step is obviously to read the data and to ease our work, I have created a CSV file. So, I will read that as that data and get that data and store it into ge.data variable. So, I am using a read dot CSV again. So, I have read the data, before I move into doing all this PCA. Let us look into the data to understand what we have at hand.

(Refer Slide Time: 2:43)



So, if I double click what I get is that. I have 5 columns. The first column is for groups, second column is for genes and the rest of the 3 columns are Drug 1, Drug 2, Drug 3. So, you can imagine maybe different cell lines or treated with, suppose a particular cell line is treated with a particular Drug at different dose.

So, Drug 1, Drug 2, Drug 3 can be different doses or these 3 can be completely different drugs and you are treating some cell with that and then you have checked the expression gene expression of certain gene starting from 1 to six. So, for 6 genes I have the data here, gene expression data just imagine their gene expression that although the values are very high sometimes 5.5 something like that.

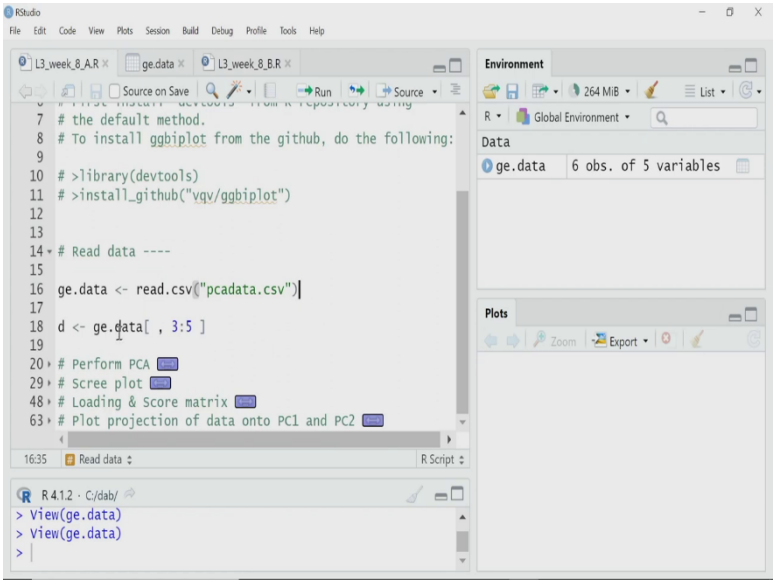
So, imagine these are gene expression data. So, for example gene one, when treat, I treat cells with Drug 1 has 2.9 reading, whereas if a Drug 3 it has 8. So, in expression of gene 1 has increased. Now, what is group, suppose from biology I know this g1, g2, g3 belongs to a particular function for example may be lipid metabolism. So, I call them group a, these are, these group belong to group a, group a is lipid metabolism group.

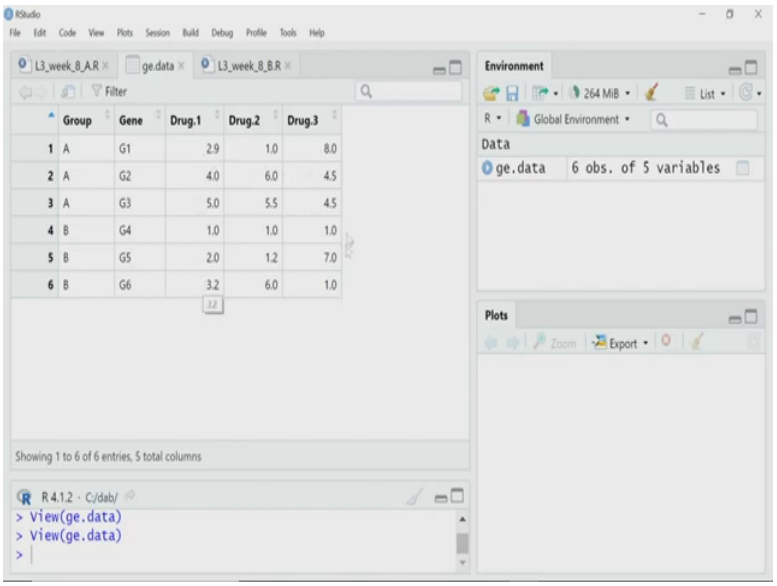
Whereas, suppose g4, g5, g6 are involved in, the motion locomotion of a cell. Or something else you imagine. So, there is group b. So, I have 2 groups and out of these 2 groups, I have 3 genes each and they are gene expression data in different in total 3 gene experimental condition, I have this data. So, it is a 3-dimensional data, I have 3 different experimental conditions, 3 variables, Drug 1, Drug 2, Drug 3, and I have observation for 6 genes.

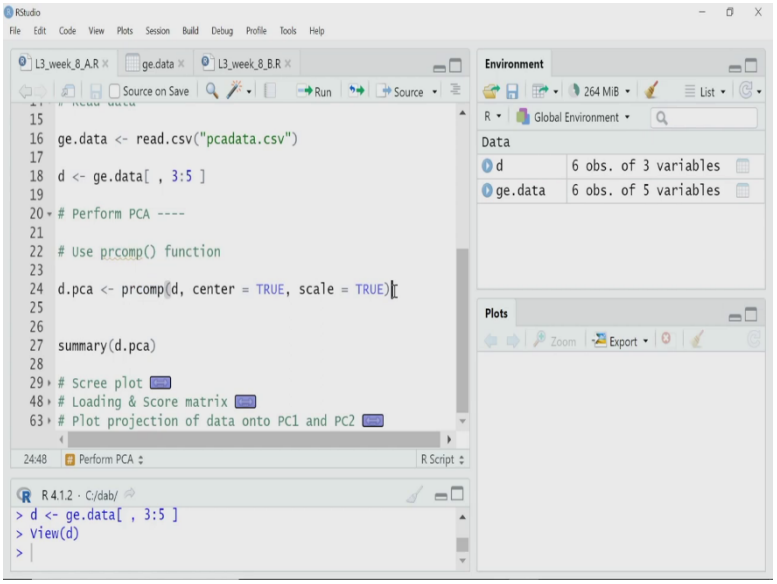
So now, if I want to visualize this data, I can create a 3-dimensional, 3-dimensional plot, definitely, but you we all know visualizing 3-dimensional data and then make a meaning out

of it is always not very easy. So, and also, we have to learn PCA in with this example. So, what we will do we will try to reduce it and make it 2 dimensional. So, I want to visualize the expression of these 6 genes in 2-dimensional space and that is why I will be doing PCA.

(Refer Slide Time: 5:11)







d.pca ← prcomp(d, center = TRUE, scale = TRUE)

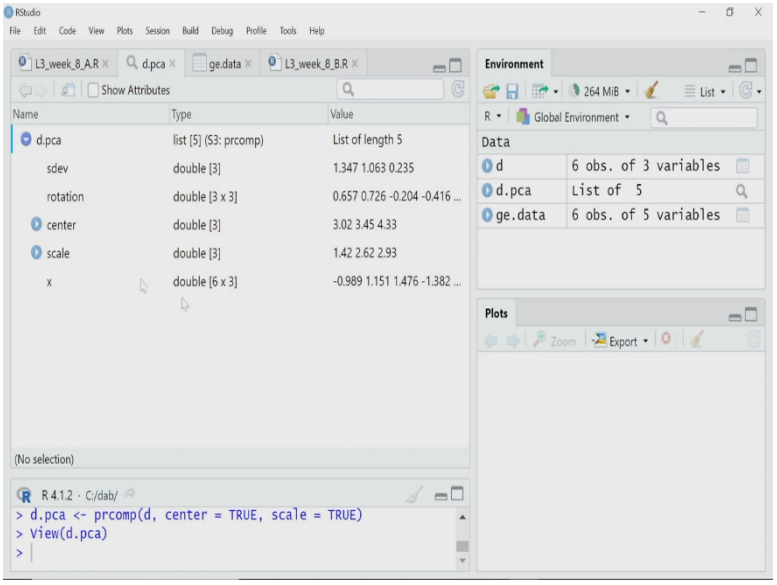
summary(d.pca)

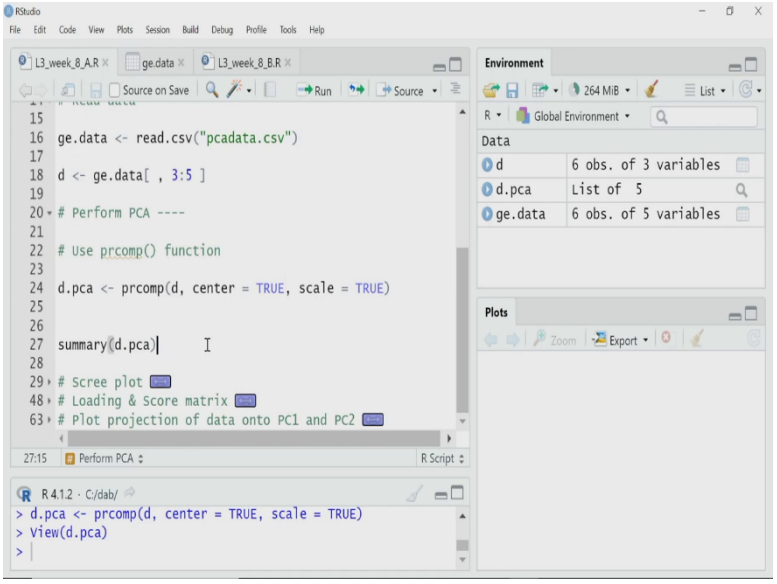
So, let us do, perform the that PCA. Now, in if you see the data, my numerical data which I have to reduce the dimension is from column 3 to column 5. So, I will extract that data first. So, what I am using, I am using the indexing. So, I am leaving the row index empty that means I want all the row of the data for columns 3 to 5, 3:5, and I will store that data in variable d. So, if you see, this is my data now.

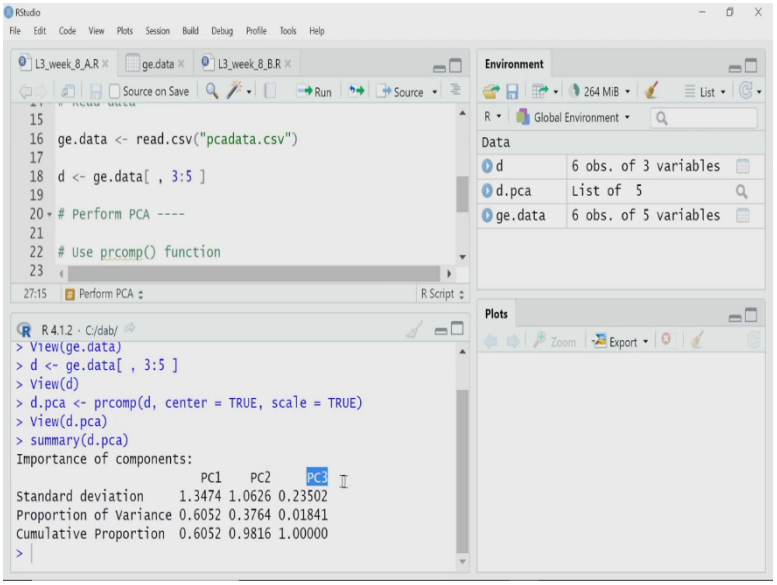
So, I will perform these PCA on this data set, to perform PCA I will be using inbuilt prcomp function. And what will be the arguments for that, obviously this data will be the first argument. So, d is the data and there are 2 other argument which are useful for PCA and quite important is that I want to center the data before you perform the PCA and also, I want to scale the data, that means I want to make the variance of each of these variables to 1.

So, that is why I have said scale equal to true, center equal to true. And I am using the prcomp, prcomp function with these arguments and the result of this PCA will be stored in a variable d.pca. So, PCA is done with a single click, now I want to see what it has created. I will first what I will do, I will first click into this environment pin d.pca to see what it is storing.

(Refer Slide Time: 6:49)







So, it is showing lots of information, for example, it is storing sdev, possibly this standard deviation, I will come back to it, it has something called rotation, center, scale, those things. Now, to get a brief of this PCA what I can do is I can actually use the summary function to get the summary of the PCA and that is quite useful. So, I will use summary function for d.pca and let us see in this console what is the summary.

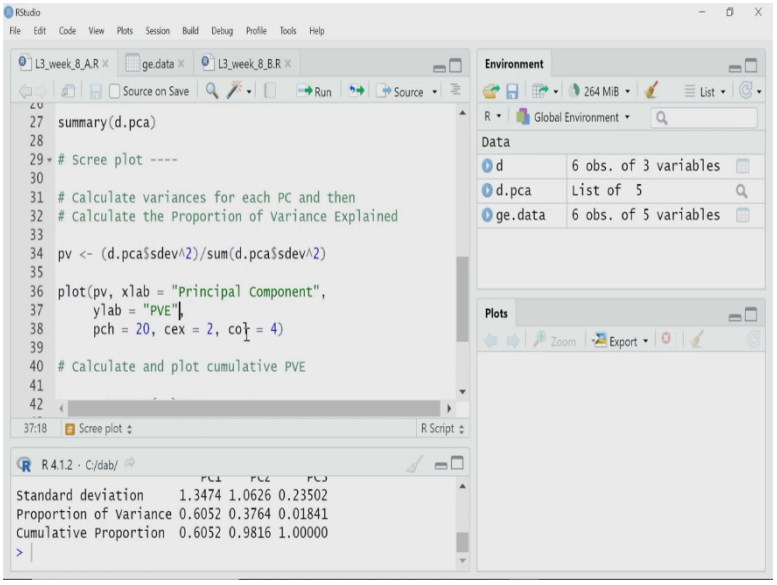
So, the summary says that okay, I have the 3 principal component PC1, PC2, and PC3, the standard deviation, remember, we want the principal components are such that along those direction, along those vectors, the variances are high highest. So, they capture the highest variance of the in the data.

So, the PCA or PC1 should have the highest variance. So, it should have the highest standard deviation. So, it is 1.34 for PC2 standard deviation is 1.0626 and for PC3, the standard deviation is very less. So, from looking at this, it seems this PC1 and PC2 together are actually capturing most almost possibly 90 of the variance in my data and also it has given the proportion of variance it has calculated.

So, 60 percent variance is captured by PC1, whereas 37 percent is captured by PC2, and PC3 capture only 0.018 that means 2 percent of the variance. So, you can easily now see that PC1 and PC2 is the most important principle component, I can actually chuck up possibly PC3. So, we will proceed and will project the data and analyze the data along this principle components.

So, once I have done that, already I have got the proportions or proportion of variances for each of this principle component, but you may be remembering that we have studied something called Scree plot, in Scree plot what we do? We represent the proportion of variance, proportion of variance explained by a particular principle component the vertical axis and the principal components in the horizontal axis, so I want to plot that.

(Refer Slide Time: 9:07)



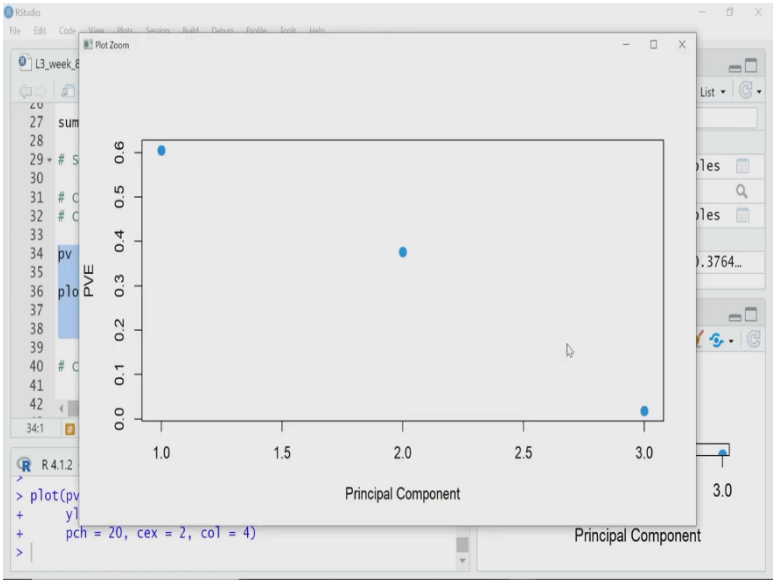

```
pv ← (d.pca$sdev^2) / sum(d.pca$sdev^2)
plot(pv, xlab = "Principal Component”,
     ylab= "PVE",
     pch = 20, cex = 2, col = 4)
```

So, to draw Scree plot, there is a Scree plot function default in R but I will not use that, I will calculate myself here and I will make a plot. So, what I will do? I have to calculate the variance captured by each of the principal component, variance associated with each of the principal component, but PCA output has given me standard deviation, only, now variance is square of standard deviation.

So, what I am doing here, I am capturing the standard deviation sdev from the PCA output. So, d.pca dollar sign sdev, so I am capturing the standard deviation for each of the principal component and then I am squaring it here. And then I am dividing the whole thing by sum of those square standard deviation. So, then it will become proportion, how much of the total variance is represented by a particular principal component.

And I am storing that data in PVE and I will plot that PVE. How I will plot, I will use the plot function, I am giving some name for the plot, and I am giving the labelling the x-axis as principal component y-axis as PVE the proportion of variance explained and I am using some symbol and colour that are given in the argument, so let me plot this.

(Refer Slide Time: 10:30)

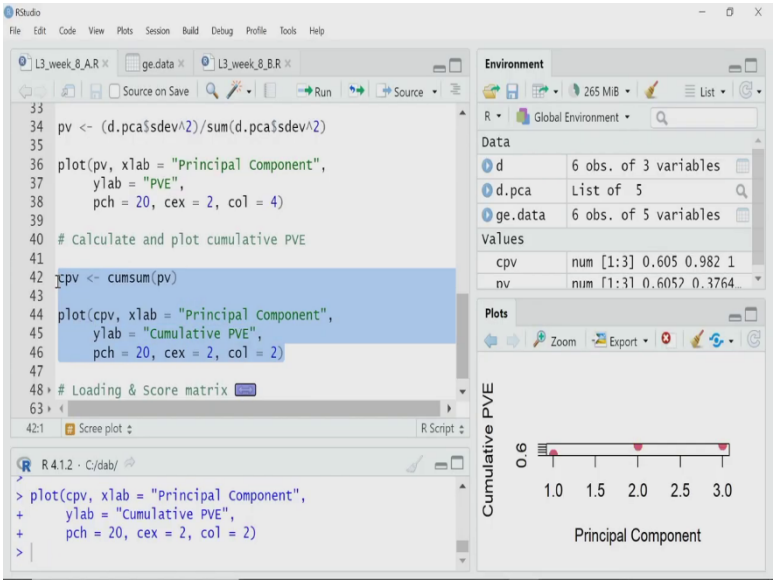


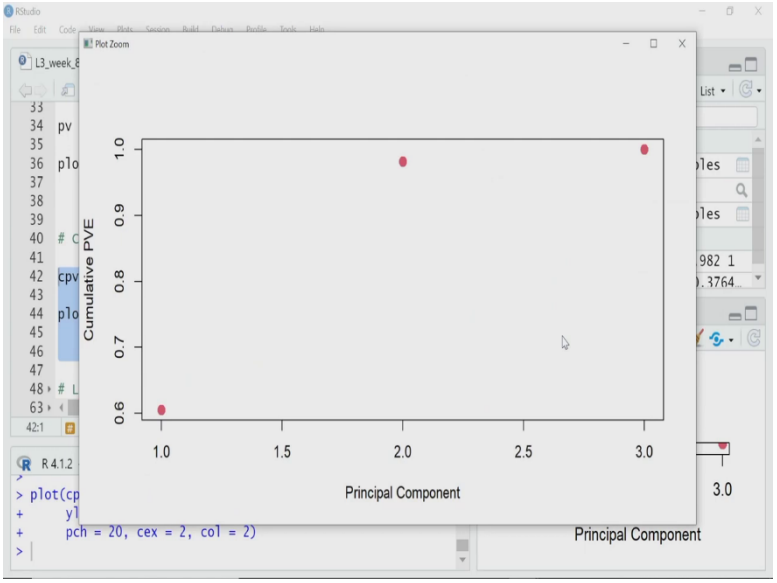
So, plot it, let us zoom, so now you can see, as you can see here these are the principal component, first principle component represent almost 60 percent of the variances. Whereas the second principal component represent around 40 percent of this. So, together both of them are representing almost 100 percent of the variance in the data, whereas the third component has very little contribution.

Now, in this case, it is very easy to identify that PC1 and PC2 should be the dimension where we should work because the third component has almost no contribution. So, we can simply remove the that third component from our subsequent analysis. But sometime it is not so easy to understand this type of plot, the PVE versus principal component plot, rather if I draw the cumulative plot then it becomes much easier to understand. So, I will do the cumulative plot for PVE and plot it.

So, to calculate, get the cumulative plot of PVE proportion of variance explained. What I have to do I have to calculate the cumulative sum. So, to do that I have the cum cumulative sum function, so I will use the PVE data as the argument. So, it will calculate the cumulative sum for the PVE data and I will store that in cpv and then I will plot that data in the same fashion. So, in this case now only the vertical axis will change, it will be the cumulative PVE.

(Refer Slide Time: 12:00)





cpv ← cumsum(pv)

plot(cpv, xlab = "Principal Component”,

ylab= "Cumulative PVE",

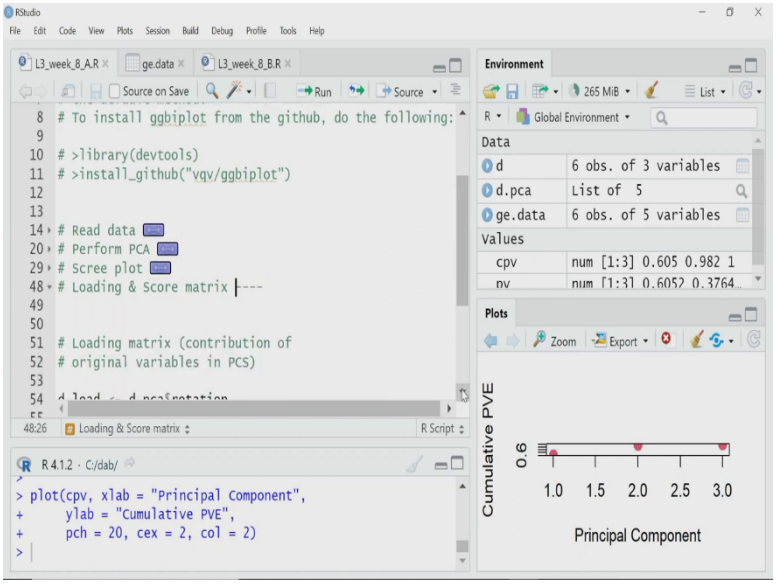
pch = 20, cex = 2, col = 4)

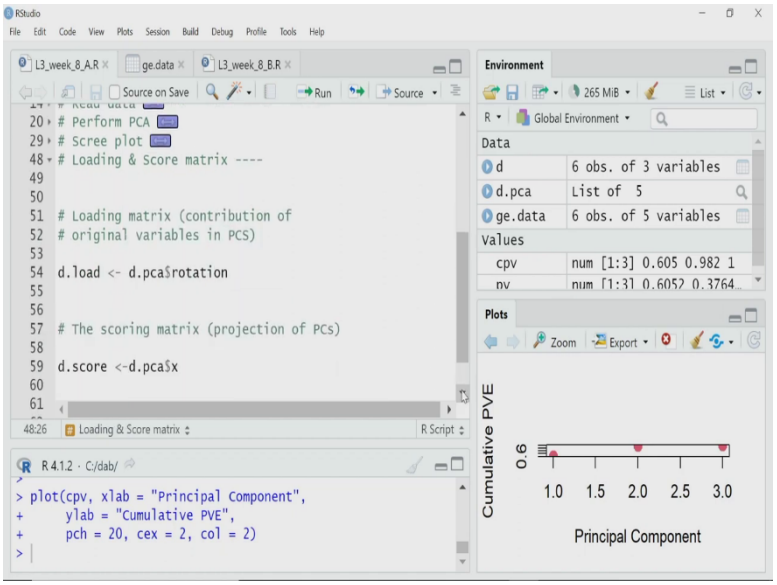
So, the plot is done, here is the plot, you can see. So, this is the cumulative plot, so it is increasing and then becoming 1, that because cumulative score will be value will be maximum only 1. So, by 2 by second component, second principal component I can see the cumulative PVE has reached almost 1.

Again, this plot says that I can actually remove principle component 3, I do not need it in my further analysis because it is not actually capturing much of the variance it is lead capturing very little of the variance. So, I have decided that I will work on PC1 and PC2 and remember that was my intention. I had 3-dimensional data I want to reduce the dimension.

So, that means if I plot my data now on this PC1 and PC2 space with horizontal axis PC1, vertical axis PC2 and I project my data in this space, that will capture the behaviour of the data, I will not lose much information and I can reduce the dimension from 3 to 2 and I can plot that also. So, before I plot let us see something like what we call we have learned earlier in PCA class that we have something called loading matrix and also scoring matrix.

(Refer Slide Time: 13:16)





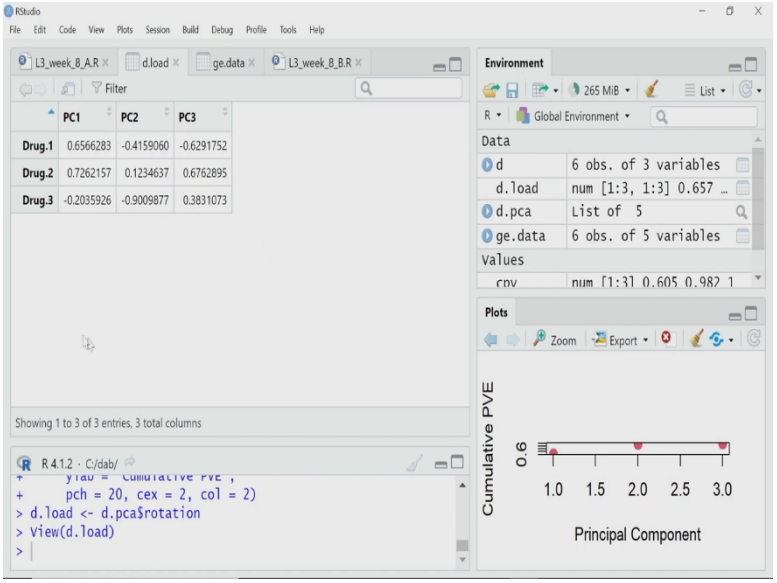
d.load← d.pca$rotation

d.score← d.pca$x

So, what is loading matrix? Loading matrix gives me the contribution of each of the original variable here Drug 1, Drug 2, Drug 3 are the original variable. So, their contribution to each of this principal component that I want to know, so that is called loading.

So, what is interesting here is actually in prcomp, in R they call it rotation, they do not call it loading, so let me calculate that. So, I am extracting the rotation matrix, which is essentially a loading matrix from the d.pca, PCA result and I am storing that in assigning that to d.load. So, let us open that d.load.

(Refer Slide Time: 13:59)



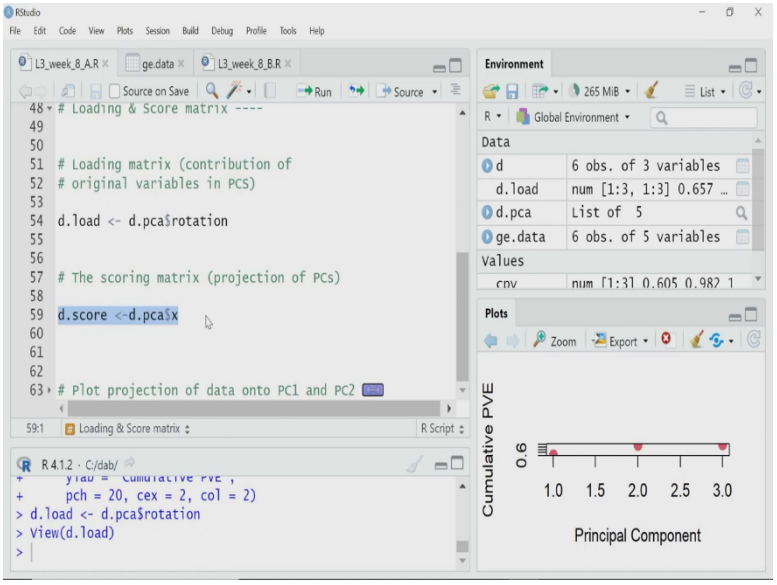
You see it is a matrix. So, PC1 PC2 PC3 are the columns and the rows are the Drug. So, now that it shows that Drug 1, I have a 0.65, so and Drug 2 also 0.72. So, Drug 1 and Drug 2 has positive contribution, good amount of positive contribution to PC1, principal component 1, whereas Drug 3 has negative contribution and that negative contribution is also weak.

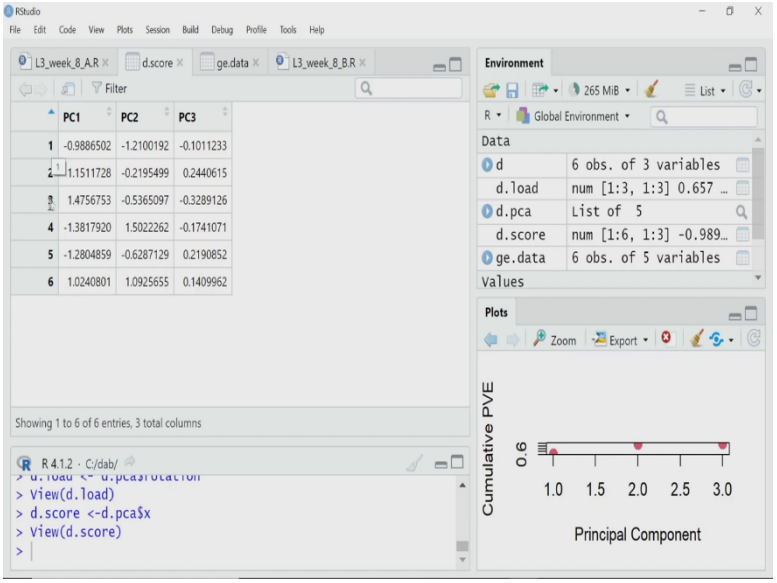
Similarly, in case of PC2 both Drug 1 and Drug 2 has negative contribution in fact Drug 3 has a strong negative contribution, whereas Drug 2 has a little positive contribution. So, this is the way I can understand from this loading matrix how the original variables are now hidden inside the PC1 and PC2, the principal components.

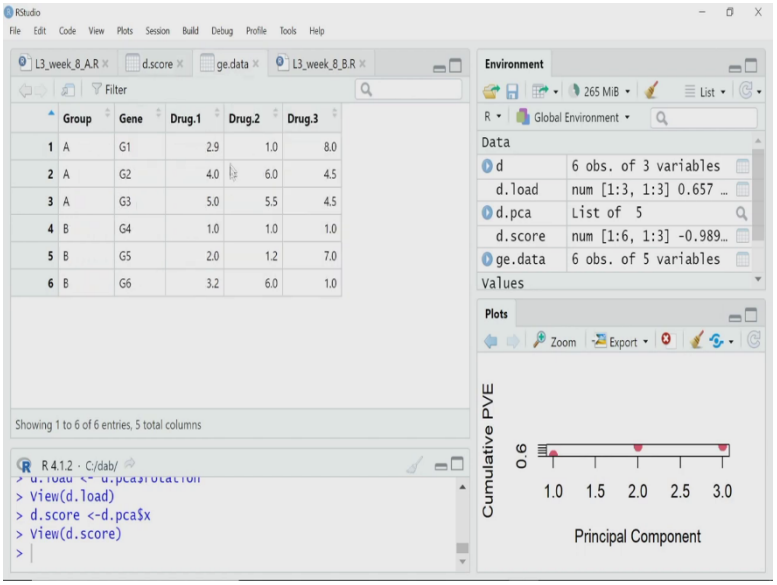
Now, once I know this the second thing, I have to know that. fine I know the principal components, I can, I want to now plot that data, to plot that I have to first project the data in this principle component unless I project it my dimensional reduction is not complete.

So, I have to project my data in this new dimension this 3 principle component and to do that I have to calculate the scoring matrix and it is very easy because the pr function has already calculated the scoring matrix and it is in the variable called x, so I will extract that and I will put it in d.score.

(Refer Slide Time: 16:23)



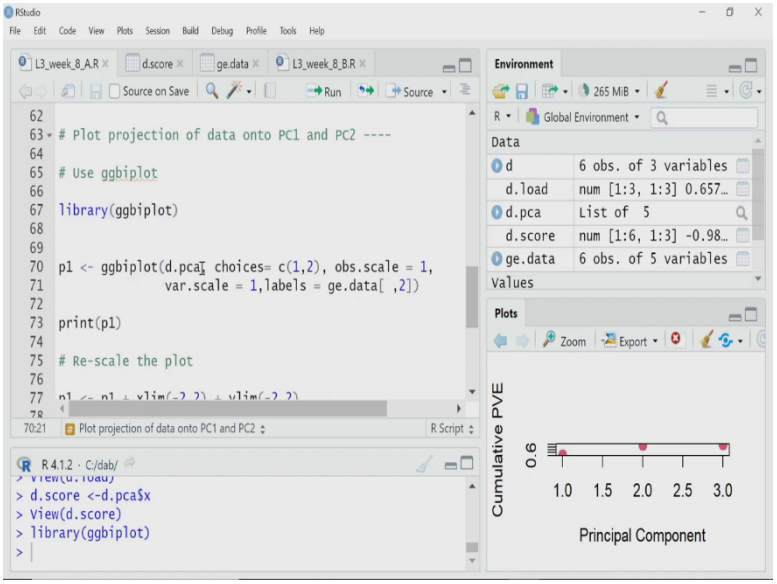




So, here I have the 3 principle component and 6 observation, if you remember 6 gene in this case. So, this matrix is the projected data, if you remember my original data I have 6 observations, g1 to g6 and Drug 1, Drug 2, Drug 3, are original dimension. In dimension 1, there is Drug 1 dimension, g1 has a value of 2.9, now I have 3 new dimension, 3 different transmissible component and the g1 is the 1.

So its value in that direction in the direction of PC1 is minus 0.99988, whereas for the second gene it is 1.15, similarly for gene 1 in PC2 it is minus 1.21. So, if I now use this data and plot it along the PC1 PC2 space I should be able to see all my 6 data point in this space of principle component 1 and principal component 2. But I will not do it, so using trivial plot function, what I will use, I will use the ggbiplot.

(Refer Slide Time: 17:02)



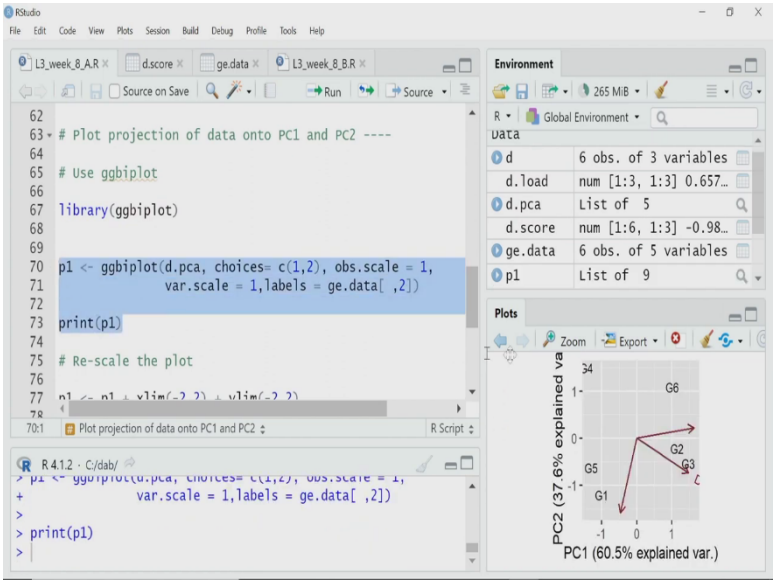

```
library(ggbiplot)
p1← ggbiplot(d.pca, choices = c(1, 2), obs.scale = 1,
                    var.scale = 1, labels = ge.data[ ,2])
print(p1)
```

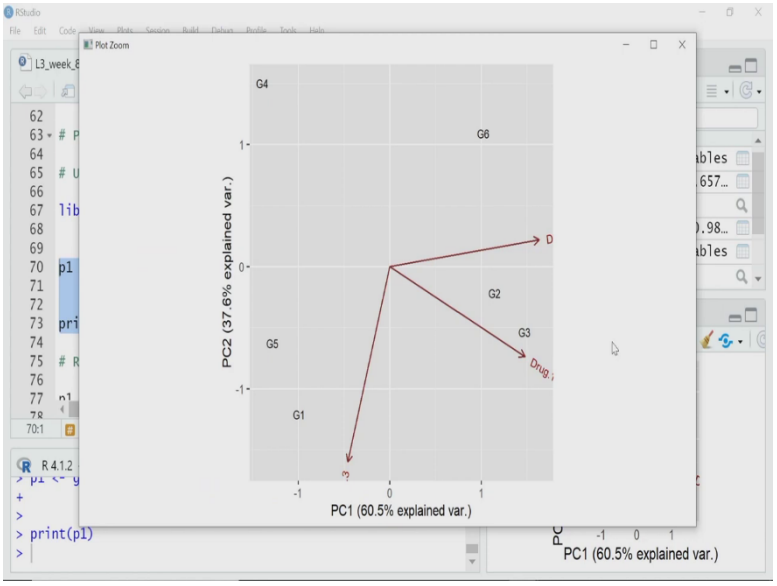
Plot the projected data I am using ggbiplot, so I have already installed ggbiplot in my machine. So, I will load it in my workspace. Now, I will be calling ggbiplot function to plot this projected data in PC1 and PC2, so how to do that? I have to assign certain arguments so for the ggbiplot, the first argument will be obviously the PCA data, that is the d.pca and then in choices I am choosing the principal component I want to plot.

In this example, I have 3 principal component 1, 2, and 3 but I have already decided principal component 3 is useless, so I want to see it in only PC1 and PC2, so that is what I am written c12. So, I am, I want ggbiplot to plot the data in principle component 1 and principal component 2. And then you have some other arguments like observation scale equal to 1, variable scale equal to 1.

So, you are saying that do not change the scales of the observation and variables retain them these are by default. There is something label argument here, I will explain it but before that let me plot it then it will be easier to explain what this label is doing. So, what I will do, I will call the ggbiplot function with these arguments and the plot object will be created that will be stored in p1 and then I will print the p1, then only I will get the figure.

(Refer Slide Time: 18:22)



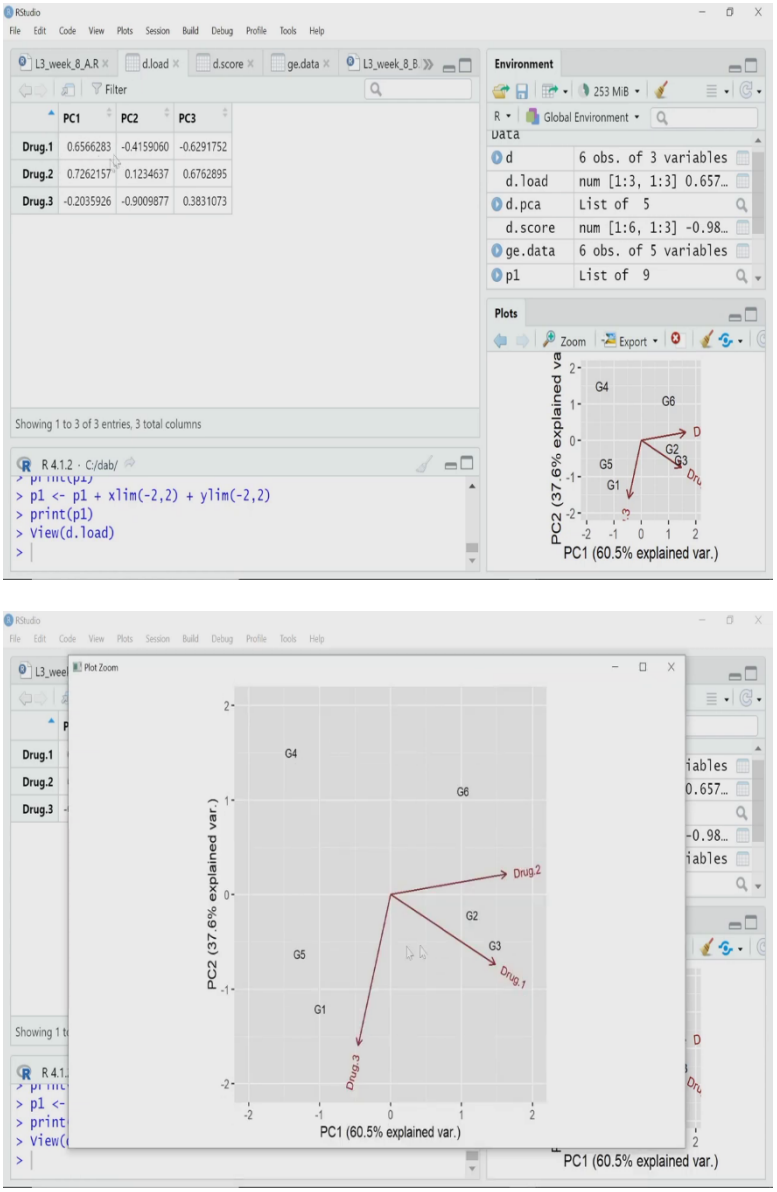


Let me zoom it. So, this is my plot, so I have PC1 in the horizontal axis, PC2 in the vertical axis and it has written that 60.5 percent of the variance in my data is explained by PC1. We also calculated that few seconds back and it is also saying 37.6 percent of the variation in my data is explained by PC2. And these arrows are actually the original variable.

Now, you may have noticed here that this Drug word has got cut here Drug has got cut here, so the labels had got cut, so what I will do, I will just change the scale of this plot a bit, so that I can accommodate all the labels. So, to do that what I am doing? I am saying that, you take the original plot object p 1 and then you change the limit for x that is x-axis from minus 1 to minus 2 and also change the y limit the y-axis limits from minus 2 to minus positive 2.

And you do it on this p1, so I am just adding them by summation sign. So, now I will print it, I hope it will be much better, let us see, now I can see the labels. So, what these arrows means? These arrows represent the original variables. Drug 1, Drug 2, Drug 3 and their contribution to the principal component, let me explain it.

(Refer Slide Time: 20:02)

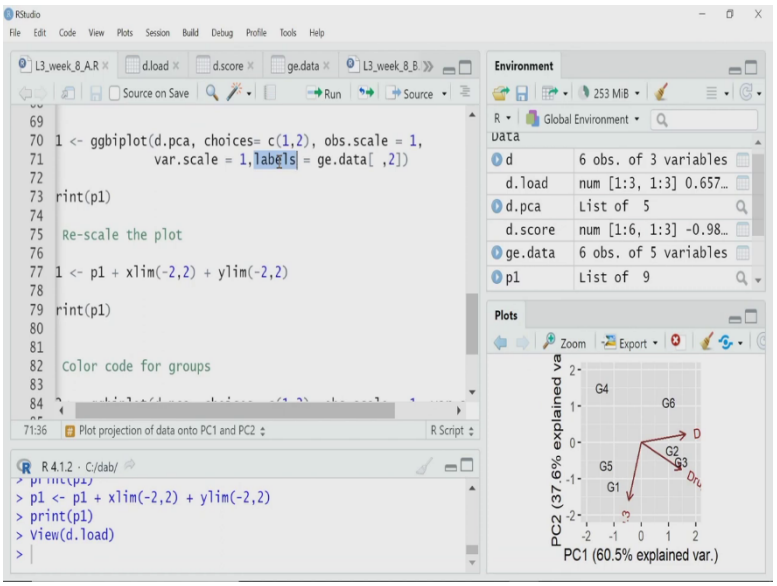


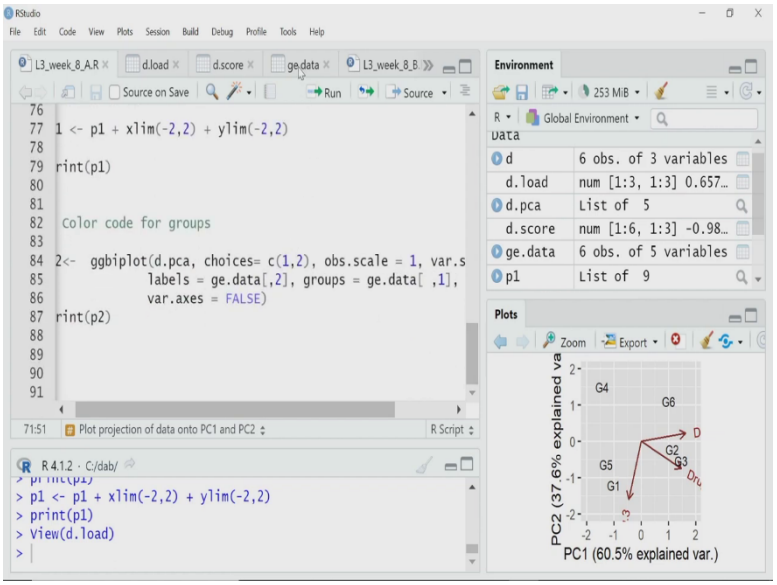
So, what I have I, if I take the loading matrix you can see PCA Drug 1 positively contribute to PC1 and negatively contribute to PC2. Let me look into the plot. So, Drug 1 arrow is pause on the positive side of PC1, 0 is here, so on this side. Hand side it will be positive. So, the arrow is pointing towards the positive side of PC1 but it is point also at the same time pointing towards the negative side of PC2.

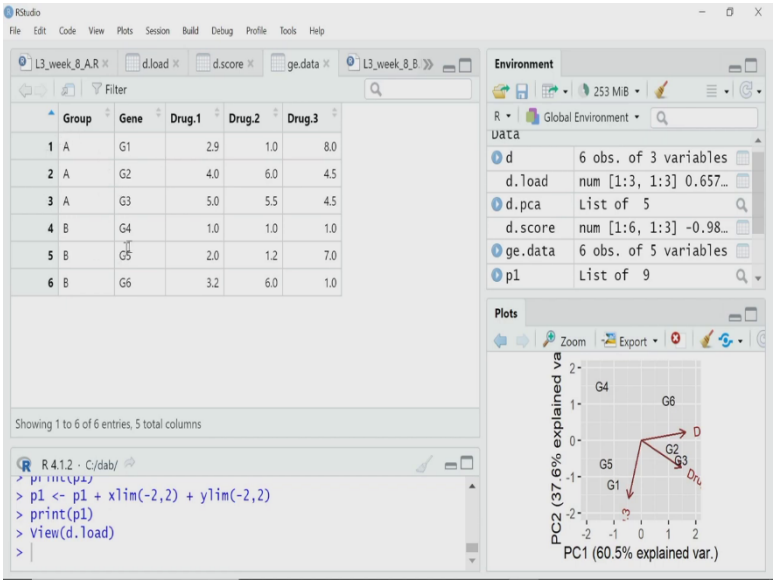
So, by this arrow I can understand that Drug 1 which is the origin 1 of the original variable has this type of contribution to PC1 and PC2. So, in a way these lines, these vectors drawn over here for Drug 1, Drug 2, Drug 3 which are the original variable are nothing but representation of the loading matrix.

Now, what are these g1, g4, g6, all these things these are my original data point, my original data points are data for gene1, gene2, gene5 up to gene6, what I have done here, rather than using symbol I could have used like circles, square, triangle, for each data point, I have used the name of the genes the label of the gene itself, as the symbol that helps, because then I can easily understand, this one is g1, this one is this, g6 something like that, I not need to use a separate symbol for that.

(Refer Slide Time: 21:34)







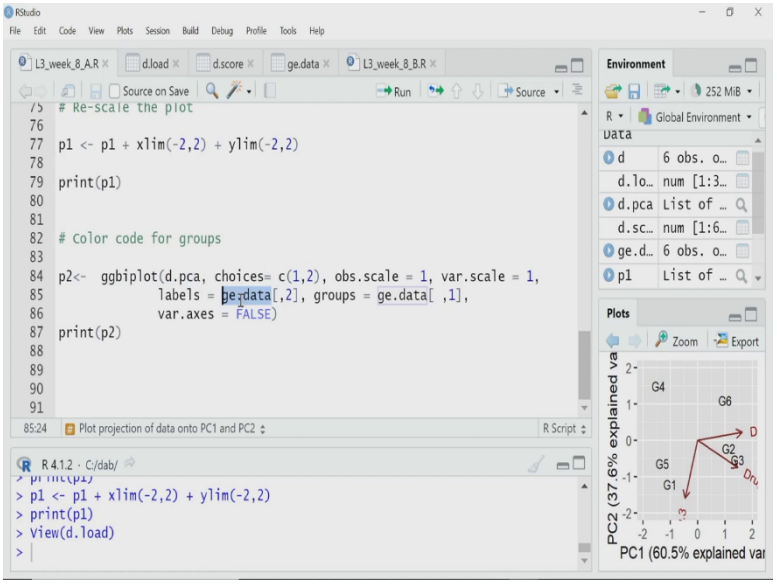
p1← p1 + xlim(-2, 2) + ylim(-2, 2)

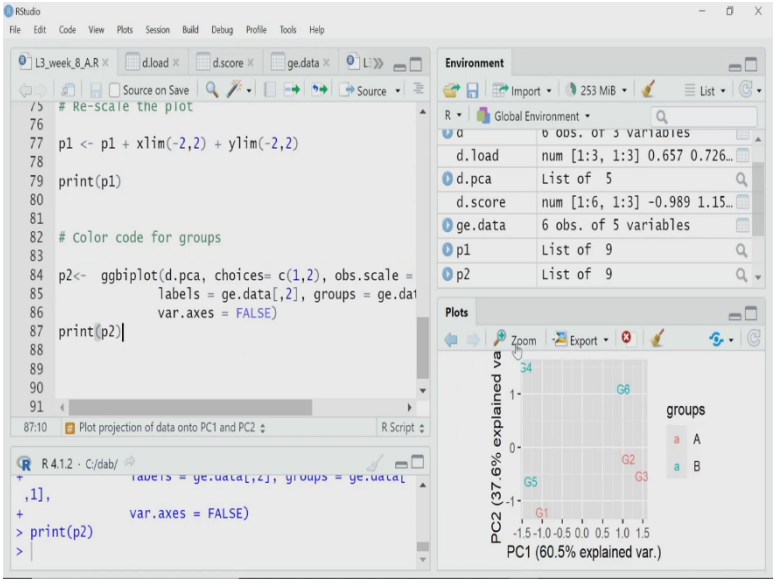
print(p1)

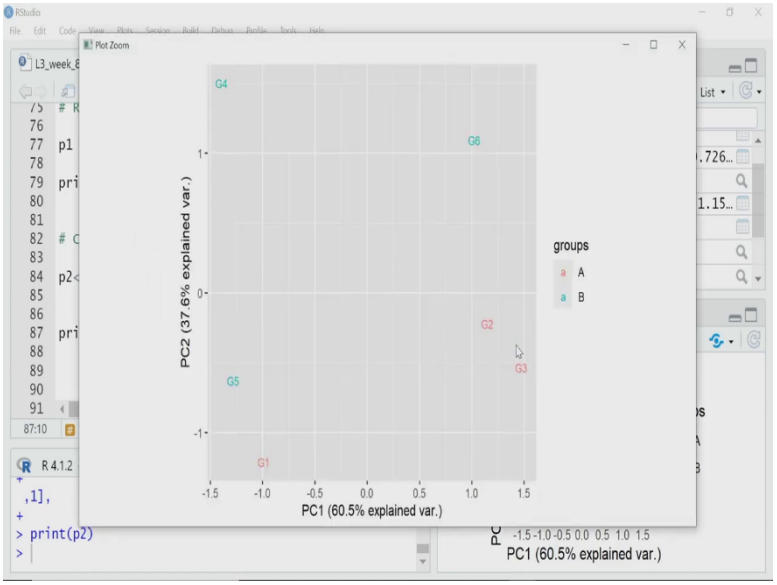
And to achieve that, what I have done? I have put an argument specifically, while calling ggbiplot. What I have said? I have said make the labels, make the labels equal to g.data column 2, what is in column 2? In column 2 I have the gene names g1 to genes 6. So, now ggbiplot has not used any symbol as a label, rather it has used the column 2 information that is the name of the genes as the label and has in place of symbol has put those num, gene numbers in g1, g2, up to g6.

Now, another important information was there in our data is that? Some of the genes belongs to group a, some of the genes belong to group b. Now, I if I want to see that grouping inside this my plot, this principle component plot, so how can I do that?

(Refer Slide Time: 22:31)







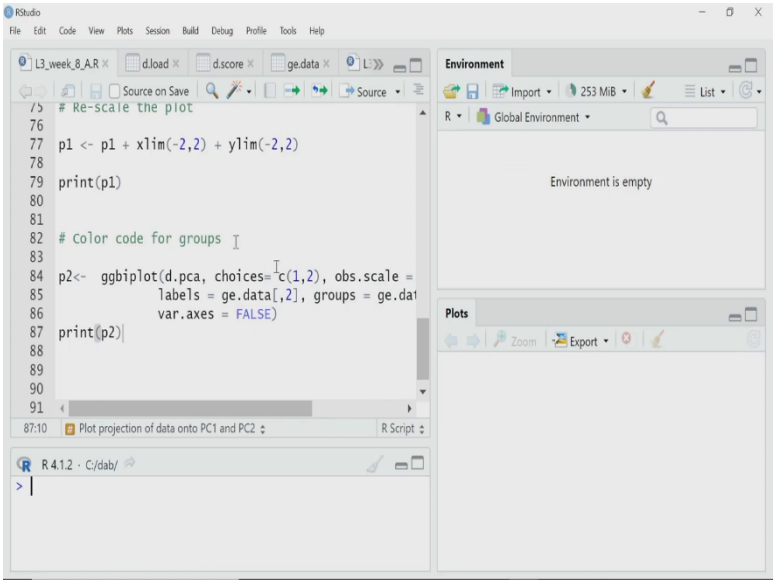
```
p2← ggbiplot(d.pca, choices = c(1, 2), obs.scale = 1, var.scale = 1,
                labels = ge.data[ ,2], groups = ge.data[ ,1],
                var.axes = FALSE)
print(p2)
```

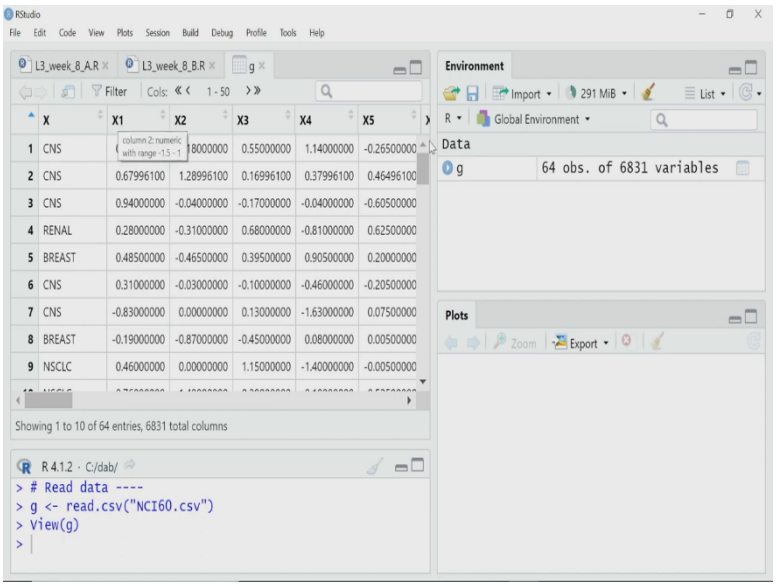
So, again I will use ggbiplot, but in this case I will use an option called groups. So, if you look into the script what I have here, same ggbiplot I am calling the same d.pca data, original data, I am choosing principal component 1 and 2, scales remain 1, I am labelling with the column 2 data as a label for the data points in the plot, and here I am adding something extra, I am saying group the data, so groups equal to g.data first column.

What is in first column? I have first column I have the group information. So, I am asking you group this data and colour code them based on this grouping. And another thing I have added sometimes, these arrows representing this vector, representing the original variable are quite disturbing that does not add to our observation understanding of the plot, so I am setting it as false.

So, var.axes equal to False. So, now let us create the object run it and then plot it. Now, I get a neat plot you can see gene 1, gene 2, gene 3 are now in pink and here in the legend they are saying it is a group A, whereas 4, 5, and 6 belongs to group B, that is why they have blue colour. So, in this way I have colour coded the data in group and I can see where they lie in this new dimension PC1 and p versus PC2.

(Refer Slide Time: 24:25)





Now, this is with a very synthetic cooked, home cooked data to explain what a how to perform PCA, and how to visualize the data. Now, I will move into a large data set where actually dimension is very high and I will be using that nci 60 data set that we have used earlier for clustering problem. If you remember that has a data for 64 cell line, cancer cell line, and all those cancer cell line they have measured the ex-gene expression of something like 6830 to 30 genes or something like that.

So, that means you have a huge dimension, 6800 dimensions, that gene number of dimension, you can never visualize it. But we believe that based on this gene expression information I can actually identify different cell type, different cancer cell type and that is what we have got

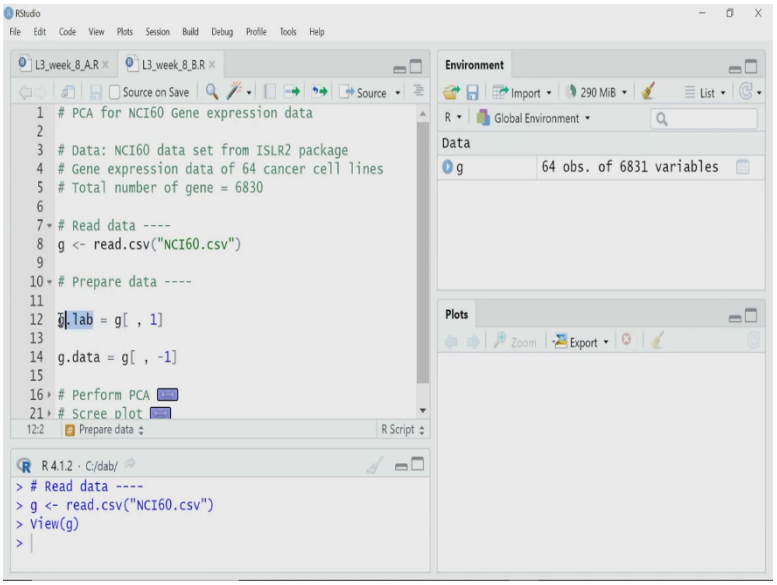
in clustering also. Some of those clusters are very homogeneous, where we observe that cell type of singular cancer are clubbed together in 1 cluster.

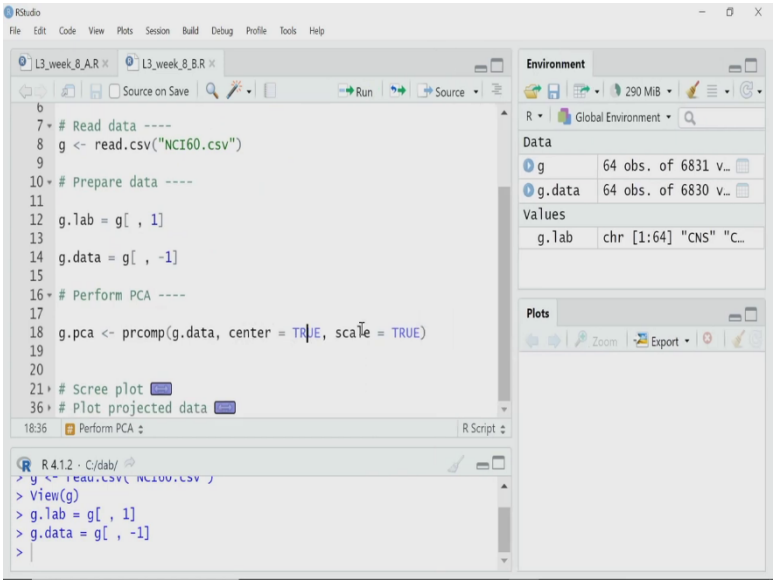
So, now many a time it is useful for visualization first of all to reduce the dimension from the 6000 dimensions to 2 dimension or 3 dimension, so I have to use PCA. At the same time sometime, it is much better to perform PCA fast and then perform clustering on the data projected on some selected PCA, that gives us better grouping, better separation of different objects in different clusters.

So, what I will do? I will perform the PCA on that nci 60 dataset. So, we already have that in CSV format, so I will read that data, it takes a few seconds to read. Now, I have to prepare the data. Just to see the data once again. So, I have the first column is for the name of the cell rather the type of the cell line, CNS, RENAL, BREAST.

So breast mean breast cancer cell line, something like that and these other columns are different genes and we have 600, 6830 such columns, such genes. So, that information is there, so now I, if to perform PCA I have to remove the first column because that does not give the gene expression data.

(Refer Slide Time: 26:29)





g ← read.csv("NCI60.csv")

g.lab ← g[, 1]

g.data← g[, -1]

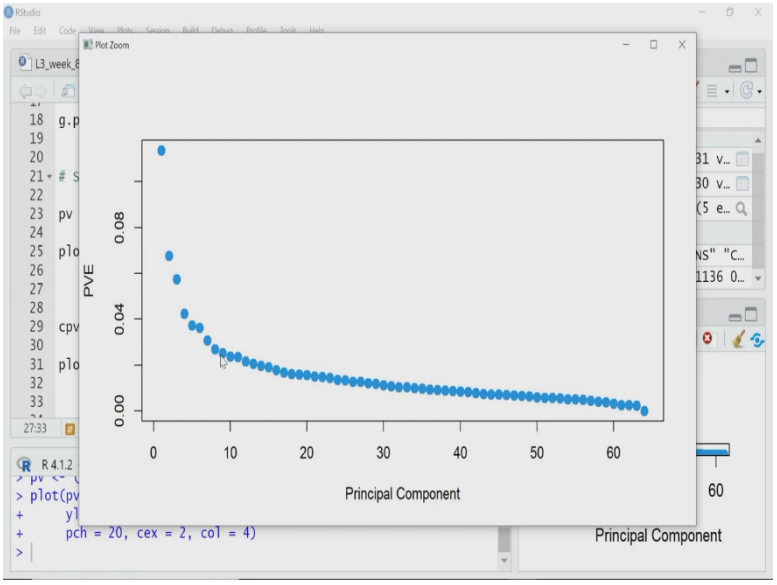
g.pca ← prcomp(g.data, center = TRUE, scale = TRUE)

So, what I am doing? I am extracting that column as label data. So, g.label I am extracting the first column here and then I am telling R that, get rid of the first column and save rest of the column as g.data on which I will perform PCA. So, now I have got g.data on which I will perform PCA, again I will use pr comp and I will keep the centre and scale true and I am performing on the data g.data, and that whole output will be stored in g.pca.

So, it is done, now I want to see the scree plot. So, I am not doing summary here because it will have large number of principal component and that may be that makes the console bit messy, so rather than that I will go directly for Scree plot.

Again, just we did for the earlier dataset, we will calculate the variance for each principal component from the standard deviation and then divide by some of the variance to get the proportion of variance explained and that will be stored in PVE. So, that is what we are doing in this line and then I will plot that PVE versus number of different principle components, from 1 to the rest. So, here it is the plot function I am calling, let us look into it.

(Refer Slide Time: 28:00)



pv ← (g.pca$sdev^2) / sum(g.pca$sdev^2)

plot(pv, xlab = "Principal Component”,

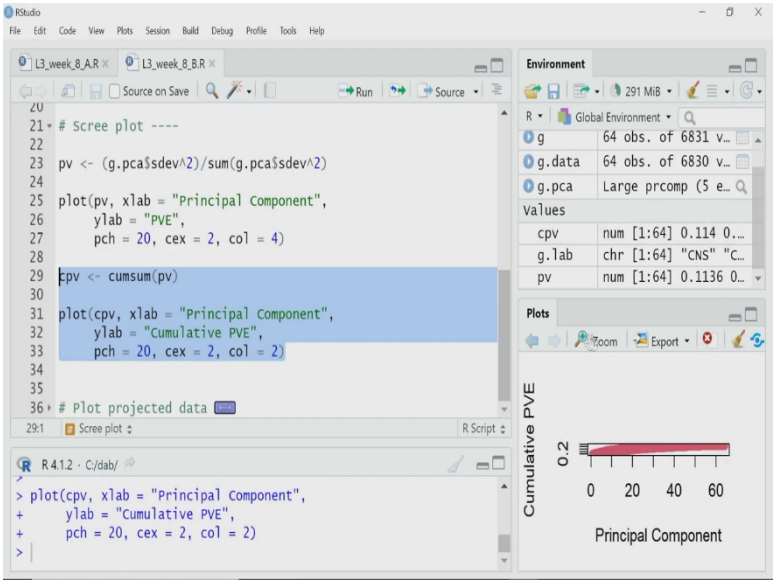
ylab= "PVE",

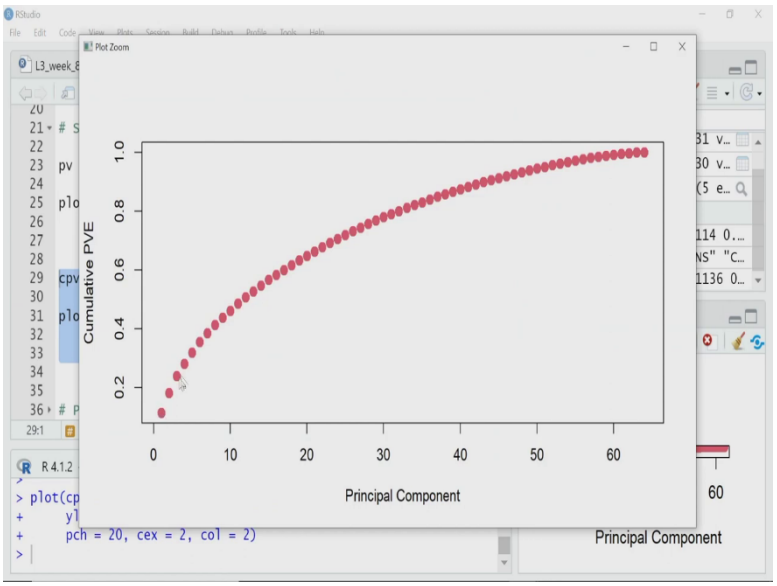
pch = 20, cex = 2, col = 4)

So, this is a neat scree plot, you can see maybe possibly up to around 8 or 9 principle component we have to consider because only after that it becomes very low. So, just principle component 1 and 2 will not actually capture all the variation in the data and all the information of the data. That is obvious because it is a quite a large dimensional system not like the earlier 1, we have just 3 variables, it has thousands of variables.

So, maybe we have to go up to 10 or 9 principal components to capture the overall behaviour reasonably. Now, as I said earlier it sometime will become difficult to understand the PVE versus principal component plot, rather if we plot the cumulative 1 that becomes much easier to understand. So, let me plot that 1 in the same fashion using the cumulative sum function and then plotting that data.

(Refer Slide Time: 28:50)





cpv ← cumsum(pv)

plot(cpv, xlab = "Principal Component”,

ylab= "Cumulative PVE",

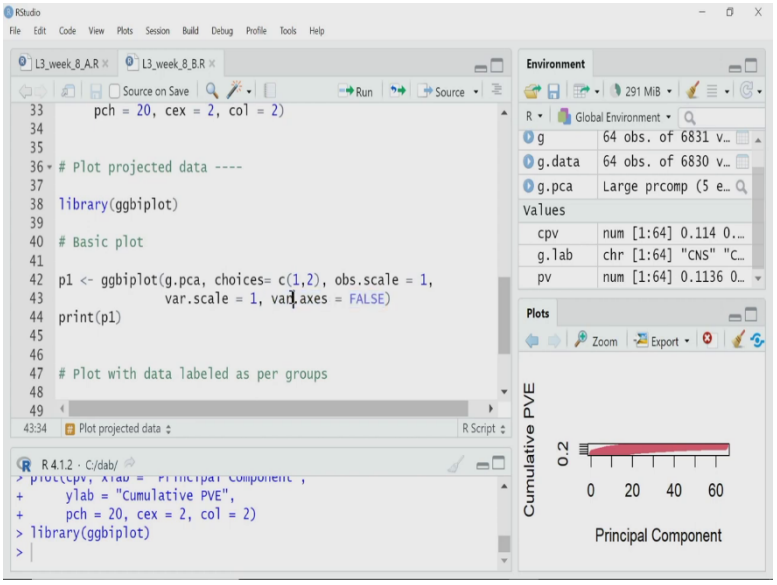
pch = 20, cex = 2, col = 4)

So, here is the cumulative PVE plot and as you can see, maybe up to not just 10, maybe up to you have to go up to 15 or something to capture the 60 percent of the variation. So, I may have to go up to principal component 15 or something to capture the 60 percent of the variation in the data. Now, that is a different case whether how far we will go.

Let me now make the projection plot, let me project this data on first component principle 1 and principle component 2, you can do that on other components also. So, to do that, again I will use ggbiplot. So, already I have loaded ggbiplot, so I do not need to load or rather let me run it once again, so I have loaded. Now, I want ggbiplot to project my 64 sample, I have 64 cell line data, on principal component 1 and 2.

So, what is my first argument? My first argument is the PCA data, g.pca, and then I am saying that my choice is principal component 1 and 2, in this case you may take 3, 4, something like that also because even up to principal component 15, they are quite important it seems.

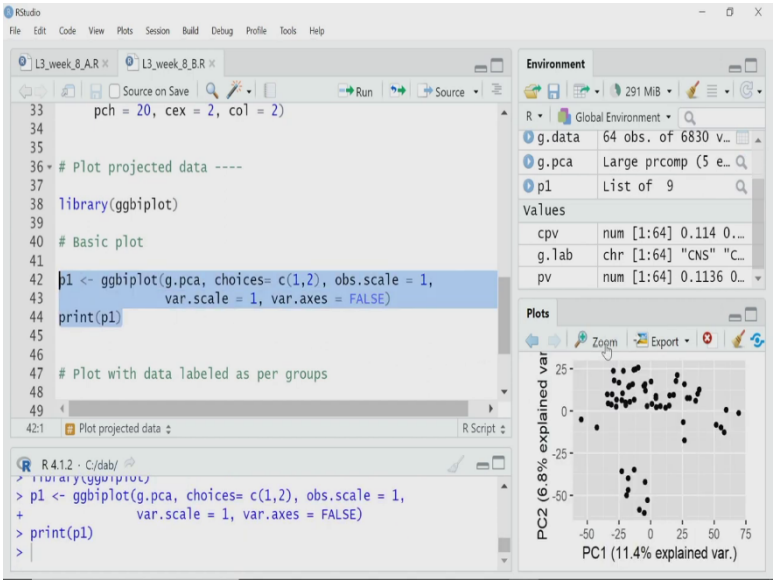
(Refer Slide Time: 30:12)

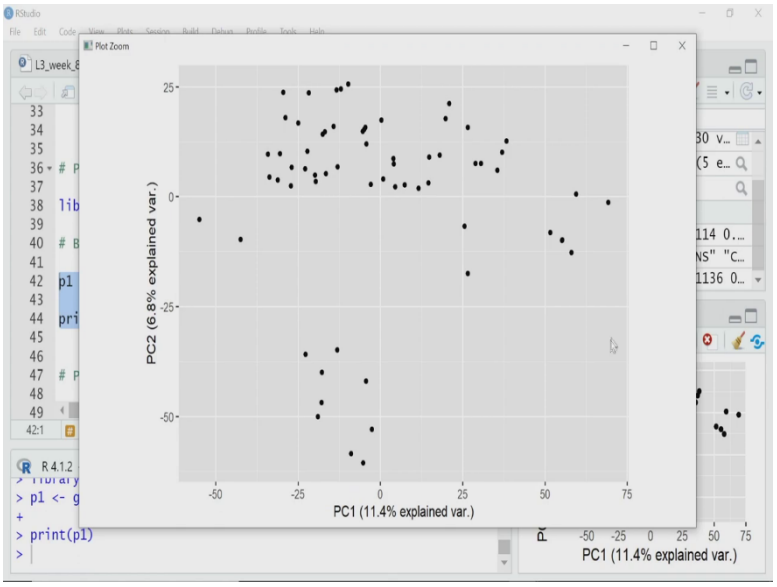



```
library(ggbiplot)
p1← ggbiplot(g.pca, choices = c(1, 2), obs.scale = 1,
                var.scale = 1, var.axes = FALSE)
print(p1)
```

And what I have done here, I have set the variance dot axis, sorry variable dot axis, the variable axis is the vectors as false, why I have done that? Because I have 6,830 variables, if I show those arrows the whole space will get covered by arrow, it does not make any sense. So, I am saying do not plot it, so I have made it false.

(Refer Slide Time: 30:40)



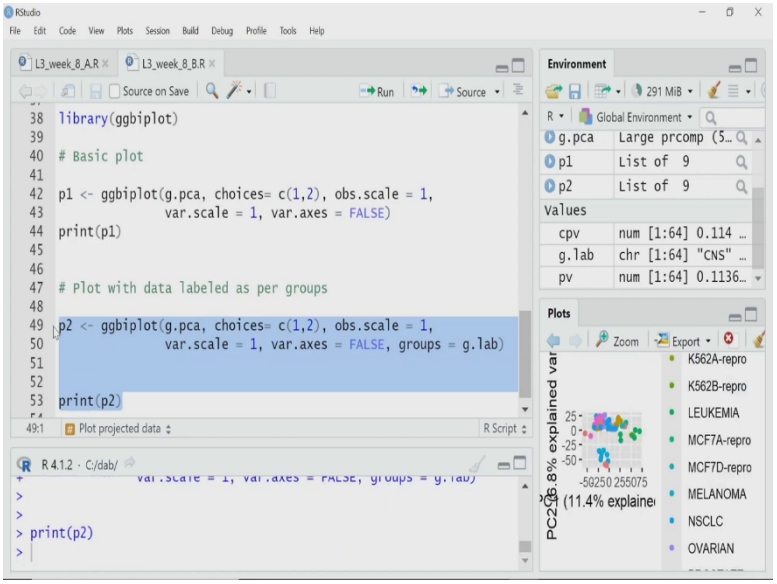


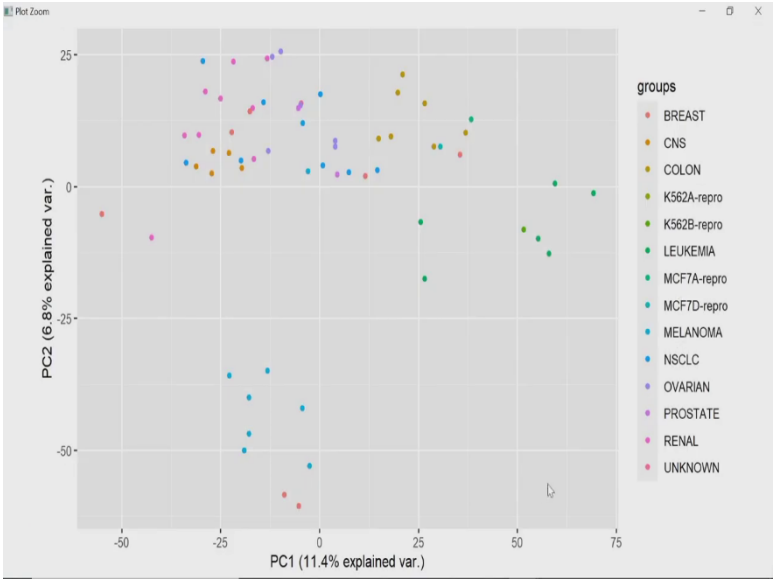
And let me create the plot, let us see. So, now this is the plot, p1 is representing only 11.4 percent, whereas PC2 is represent 6.8 percent. And all these data point, these black field dots are essentially those 64 cell lines. Now, you may be wondering why I have not labelled them with names or something like that, I have not done that because the name of each of these cell lines are not single letter but suppose renal or leukaemia, something like that quite big and if I write them on this plot, this plot will look ugly.

So, rather than that I have chosen, I have allowed ggbiplot to use symbol the default symbol it has used is the black field circles. So, how, what I have done actually, I have not used any label argument while calling ggbiplot and that is why it has used the default label for plotting the data and use the filled circle with black colour. Now, you may wonder that that does not give me clear picture.

It will be good that somehow if I can represent each of the dot of each of the cell line. So, yes that is what I will do now, what I will do? I will colour these dots based upon which cell line they belong to, which cell type they belong to. For example, all Renal cell line, cancer cell line will be given a particular colour, so all dots that belong to renal cell line data point will become that colour, will have that that colour.

(Refer Slide Time: 32:14)





p2← ggbiplot(g.pca, choices = c(1, 2), obs.scale = 1,

var.scale = 1, var.axes = FALSE, groups = g.lab)

print(p2)

How I will do that? I will use that option of groups that we use in the earlier example. So, now here I am saying calling ggbiplot again, and I am choosing principal component 1 and 2, everything is remaining same, variable axis is false, I am just adding this groups argument again, and now I am asking to take the label, if you remember, we have extracted the label, the first column of the data set as label for the data and I am saying use that label to group my data.

So, that is what it will do it will group the data based on that label and then put each group unique colour and it will visually, be visualized in the plot. So, here I plot it, let me zoom it, now you can see a nice plot here. So, the PC1, PC2 remains same, only thing it has changed is that the colour of each of this dot.

For example, this leukaemia is a green thing, so possibly these are the leukemia one, the nsa clc are here, and these unknowns are here. So, in this way by using the group option group argument I have coloured each of the cell lines based upon which group or which cell type belongs to. That bring this brings us to the end of this lecture.

But before I end let me point out 1 common mistake sometime people do in hurry, is that here looking at this data you can easily see some green points are forming some sort of cluster, some blue points are forming some sort of cluster, and you may get tempted to say, I have achieved clusters, no, principal component is not for clustering, what we have achieved?

We have reduced the dimension. I had 6,830 dimensions, from that I have reduced it to lower dimension, maybe I can go up to the 15 first principle component or up to 20 first principle component, even then the number 15 or 20 is much lesser than 6830.

So, PCA has allow us to reduce the dimension, it does not allow or does not tell us where the cluster exists, if there is cluster in the data or not, it does not do that. While plotting this PCA plot, by chance you may see some cluster but that is not the result of the algorithm, that is because of how the data are but that is not actually the cluster the way we, the as a cluster generated by clustering algorithms.

So, what you have to do? If you really want the cluster, whether you want to see, whether based on this gene expression I can get cluster of cell line or not, you have to use some clustering algorithm like hierarchical clustering algorithm, or k-means clustering algorithm, that we have learned earlier. Now, rather than using the original gene expression data, you may use the projected data, the scoring matrix data as input for this clustering algorithms, and then you can cluster the data. That is all thank you for learning with me today.