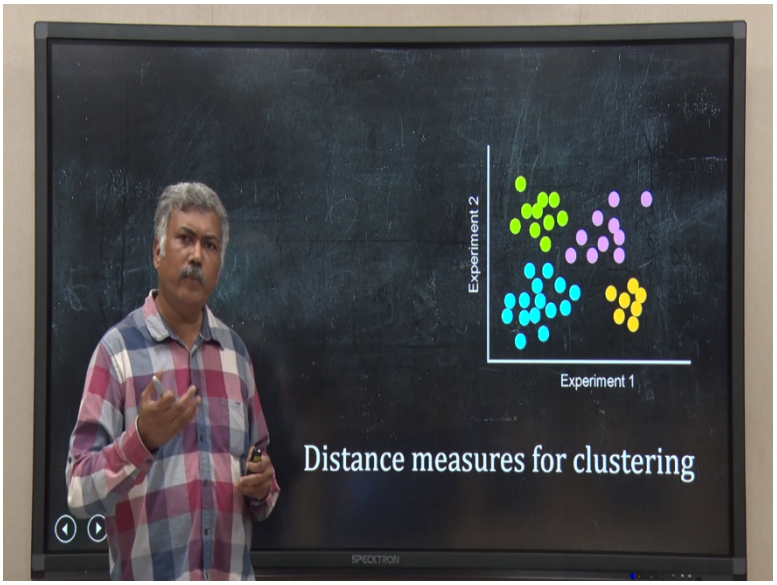
Data Analysis for Biologists
Professor Biplab Bose
Department of Biosciences and Bioengineering
Indian Institute of Technology Guwahati
Lecture 37
Distance measures for clustering

(Refer Slide Time: 00:29)



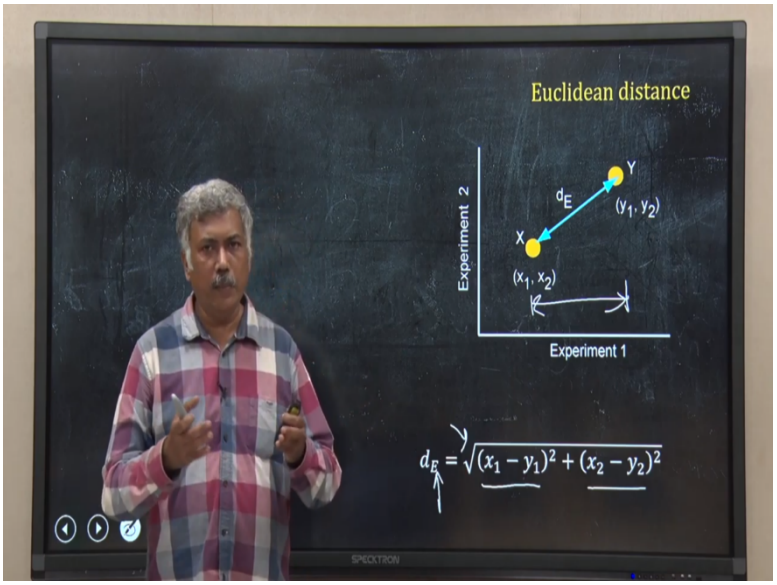
Clustering and classification of data are two key component of statistical learning. In clustering, what we do we have a heterogeneous population of data points and we break them or we group them in some clusters, small homogeneous to some extent homogeneous groups.

(Refer Slide Time: 00:49)

As we have discussed earlier, suppose, I have a gene expression data form coming from two experimental condition. The clustering algorithm will try to find out which of these genes in

my data set have similar or close behaviour in these two experimental conditions. When we say that two data point has a close behaviour or similar behaviour, that means, I need some sort of measurement of closeness or similarity between two data points. So, in this lecture, I will discuss different types of distance measure that are usually used in machine learning particularly in biology.

(Refer Slide Time: 01:34)

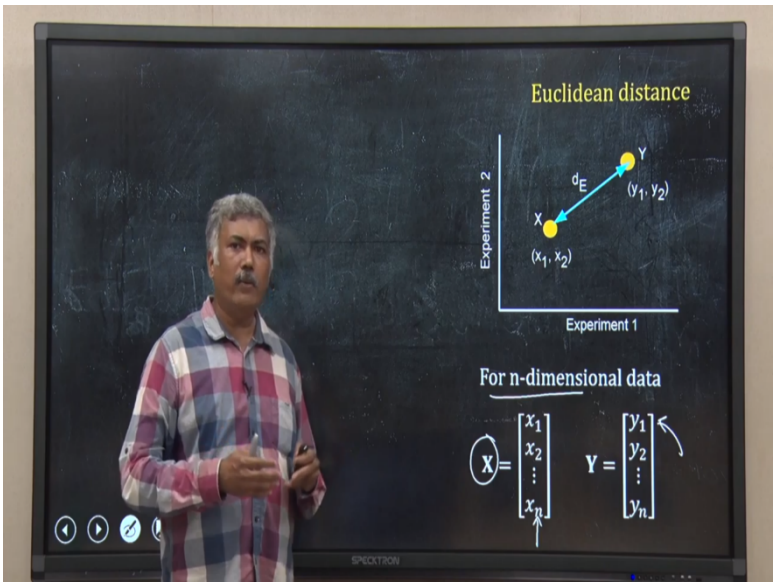


$$d_E = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$

I will start with the simplest and the well-known one that all of us know from school is called Euclidean distance. So, same experimental condition suppose, we have we have two genes X and Y suppose and I want to measure the distance between them and I want to use Euclidean distance as a measure of distance between them. So, in experiment 1 and 2 X has x1 and x2 level of expression whereas, in experiment 1 Y has y1 and experiment 2 y has y2. So, x1, x2 is the coordinate for X gene and y1 y2 is the coordinate for the Y gene. You want to calculate the distance between these and you will use the simple rule we learn in geometric class.

What I will do, the distance I have written E to represent Euclidean distance will be equal to square root of inside the square root we have x1 minus y1. So, this distance x1 minus y1 whole square plus x2 minus y2 whole square and you are doing the square root of that. This is essentially Euclidean distance and we have all learned that in school and we can use this as a measure for distance between two genes in my experiment or any other two objects in my clustering problem.

(Refer Slide Time: 02:49)

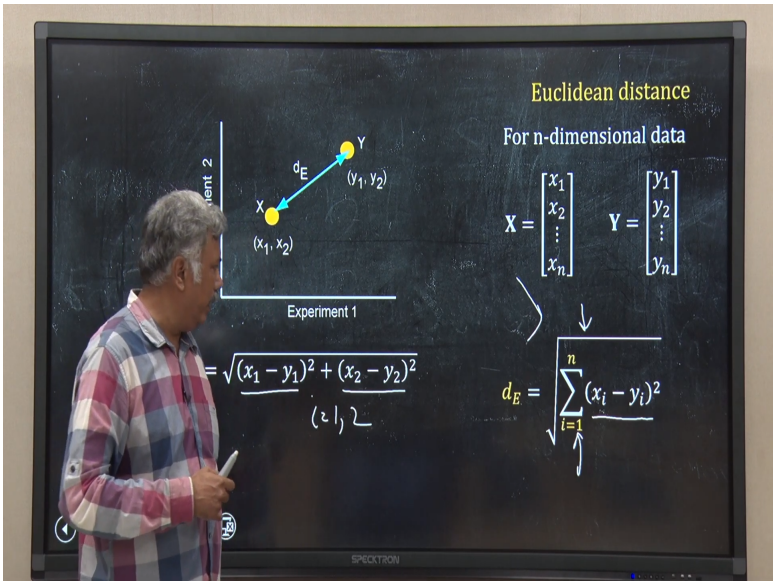


$$X = \begin{vmatrix} x1 \\ x2 \\ .. \\ .. \\ xn \end{vmatrix}$$

$$Y = \begin{vmatrix} y1 \\ y2 \\ : \\ : \\ xn \end{vmatrix}$$

Now usually, as I said your experimental condition will be large right not just two experimental condition, you may have 10-12 experimental condition, you may have taken sample from 50 sample 50 samples like 50 tumour samples, so, you have a multi-dimensional data. So, suppose I have n dimensional data that means for X gene, I can represent that data as a vector x1, x2, xn are the measurement of that in different experiment or different samples. And for Y also I can stack this data to create a vector y1, y2 up to yn. Now, I can use the same rule that just we discussed for two-dimensional problem to calculate the Euclidean distance.

(Refer Slide Time: 03:32)

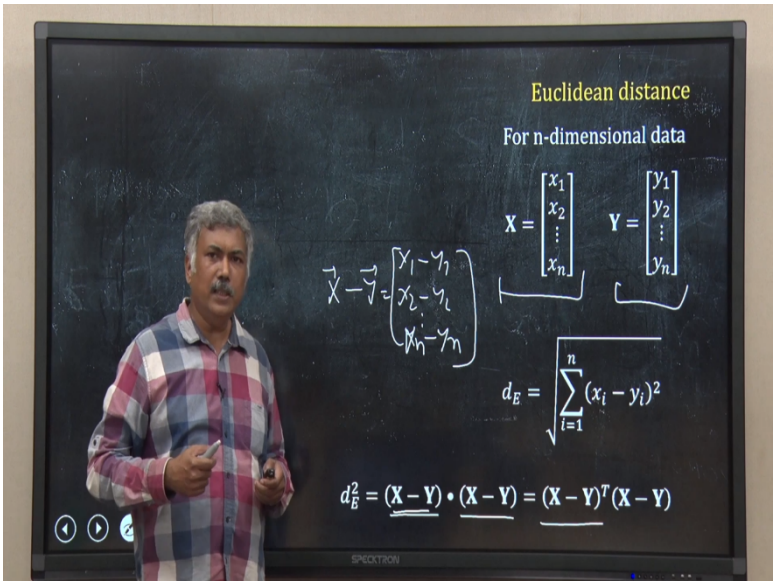


$$d_E = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$

$$d_E = \sqrt{\sum_{i=1}^{n} (x_i - y_i)2}$$

What I will do? In two-dimensional problem I have two squares inside that square root. Now, I will have n summation, so, what I have to do, I have to calculate xi minus yi where i varies from 1 to n and then square that. I will sum all those values. In this case, i has varied from 1 to 2 actually. So, in this particular case, where I have n dimensional data, I will vary i from 1 to n and I will get the square of this difference and sum all those n terms together and calculate the square root. That will be my Euclidean distance between two objects or two data points, or in this case, two genes in n dimensional data set.

(Refer Slide Time: 04:27)



$$\vec{X} - \vec{Y} = \begin{vmatrix} x1\text{-}y1 \\ x2\text{-}y2 \\ .. \\ .. \\ xn\text{-}yn \end{vmatrix}$$

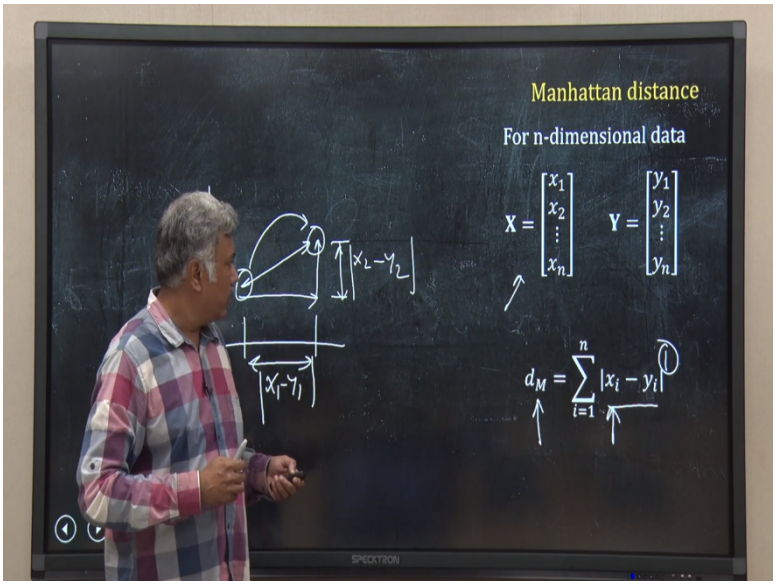
$$d_E = \sqrt{\sum_{i=1}^{n} (x_i - y_i)2}$$

$$d_E^2 = (X - Y) \bullet (X - Y) = (X - Y)^T (X - Y)$$

Now, actually, we can use linear algebra to very easily to calculate this Euclidean distance. Let me show that what we will do. So, usually what we will do, we will represent these data as column vectors and then the dot product between X vector minus Y vector. So, you have X vector minus Y vector, so, that will be something like that x1 minus y1 x2 minus y2 something like that xn minus yn.

So, this is the X minus Y vector. So, the dot product between X minus Y vector will give you the square of the Euclidean distance. And you can actually represent the dot product in this form, you take the transpose of X minus Y and multiply with X minus Y. So, these are easy to do using R and you can easily, or any other programming language. And rather than doing the sum by iteration in a loop and then adding them and then sticking the square root, this method is much faster to implement. Now, Euclidean distance measure actually the diagonal distance is it not.

(Refer Slide Time: 05:33)



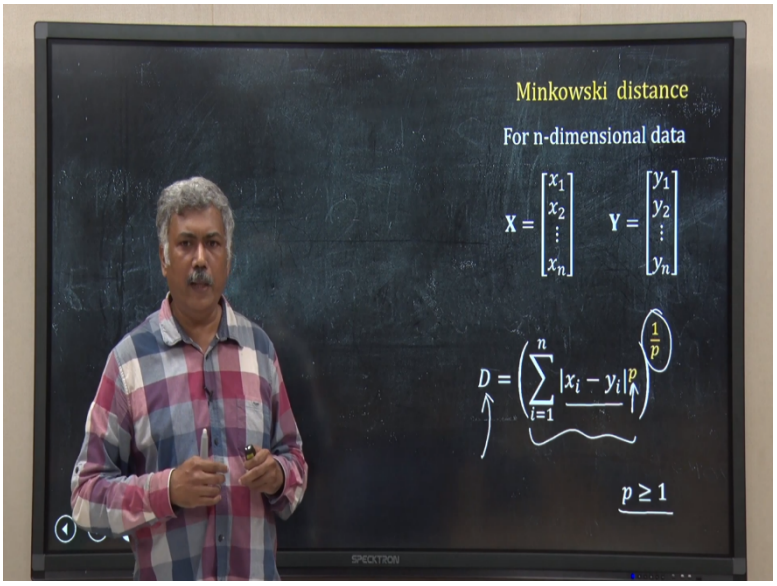
$$d_M = \sum_{i=1}^{n} |x_i - y_i|$$

For example, if I have two data point, two genes or two objects, you are calculating the diagonal distance between them. What if I do not want to measure the diagonal distance or something else that how many steps I have to move in the right and then up to reach from one data point to another one, that means I am saying how many steps I will go in this direction and then how many steps I should go up to move from this to this. And that is where the Manhattan distance comes. So, I have the same n dimensional data for X and Y two data points or two genes or two objects.

And the Manhattan distance will be equal to the difference between xi and yi in two-dimension, this is x1 minus y1 and this is x2 minus y2. So, you take the absolute value of these, you take the absolute value of this difference and you sum them. No squaring nothing, just the absolute value and you sum them. That is called Manhattan distance. Now you can easily see there is a pattern. In Euclidean distance, we have the squared term and then we sum them together and as we have taken a squared term, we are taking a square root.

In Manhattan distance, my power is here 1, so I do not need to take a square root or something like that.

(Refer Slide Time: 07:13)



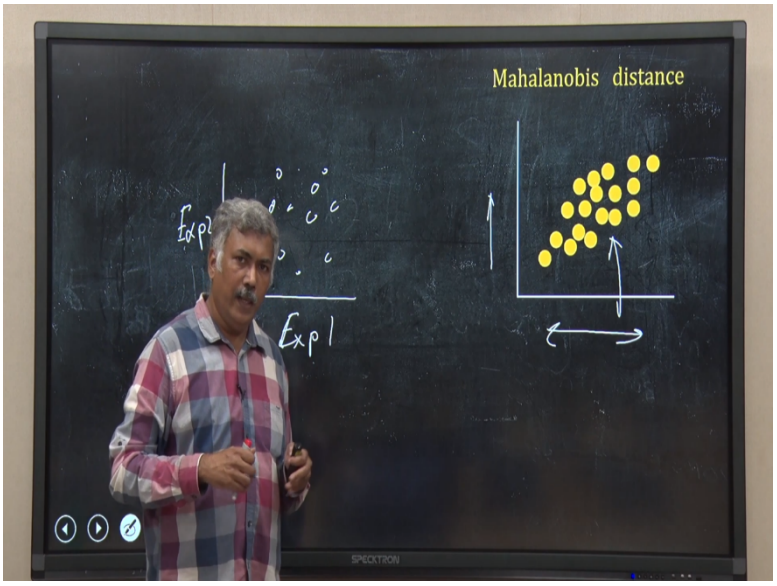
$$D = \left(\sum_{i=1}^{n} \left| x_i - y_i \right|^P \right)^{\frac{1}{p}}$$

$$p \geq 1$$

So, what we only try to generalize this, we get something called Minkowski distance. How do you define that? That Minkowski distance is equal to, you take the absolute difference between xi and yi and raise it to the power p and sum those value, you raise to the power p and sum those value and take the pth root that is why I written 1 by p. So, you take the pth root. So, when p equal to 1, it is Manhattan distance. When p equal to 2 it is Euclidean distance. So, this is a generalization and p should be always bigger equal to 1. So, this is called Minkowski distance.

Now, in all this distance measure, whether you are using Euclidean one or generalized Minkowski distance, we are not bothered about dispersion of data but dispersion of data can have effect on this measure starting from Euclidean to Minkowski distance. Let me explain what do I mean by that?

(Refer Slide Time: 08:16)

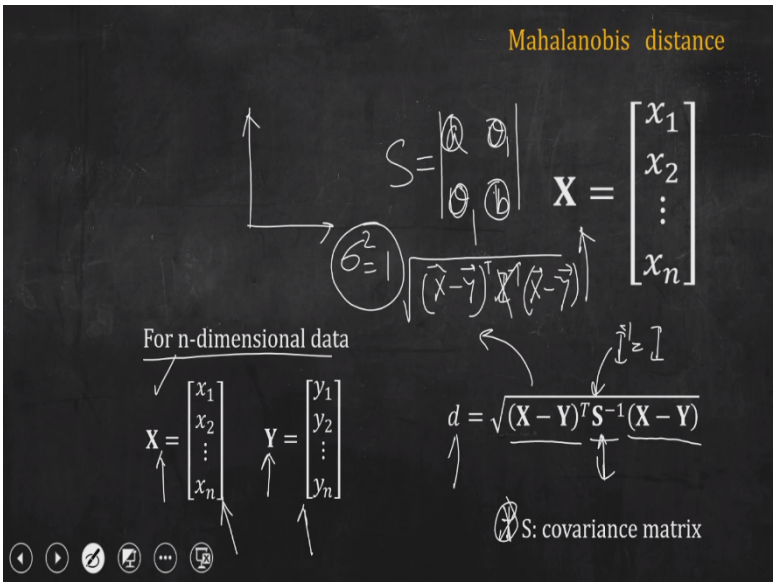


In general, if there are suppose, I have experimental condition 1 and I have experimental condition 2 and there is not much difference in the variance of the data in these two experimental conditions. And there is no correlation between or relation between a dependency between experimental condition 1 and experimental condition 2. Then what will happen my data points will be almost equivalently dispersed throughout this space, is it not. Something like this.

But if you imagine that the experimental 1 and experimental condition 2 has some sort of correlation between them, then what may happen? My data may look like this, it has a skew. At the same time the variance of these two experimental, of data in these two experimental condition may also vary. So, that will give another type of skew. So, there is a hidden relation in this data, the data is not equivalently or uniformly distributed throughout the space. So, this bias distribution, this hidden relation between different dimension can actually give error to your calculation for Euclidean distance or any other similar distance measure.

Now, to correct this error, what we have to use? We have to give some weightage for different dimension. For example, in this direction, we have more dispersion whereas in this direction you have less dispersion. So, you have to give differential weightage between data for these dimensions and that is where Mahalanobis distance comes. How we define Mahalanobis distance?

(Refer Slide Time: 10:02)



$$d = \sqrt{(X - Y)^T S^{-1} (X - Y)}$$

So, this is my data. I have n dimensional data here for X and Y. Now, the Mahalanobis distance is defined as the square root of X minus Y transpose into inverse of S into X minus Y. What is X minus Y? X minus Y is the difference between X vector and Y vector. X vector is for one gene, one object. Y vector is for the other gene or other data point or other object. What is S? S is the co-variance matrix. We take the inverse of that and we put it in the middle and the whole thing is multiplied and you take a square root of that. That is Mahalanobis distance.

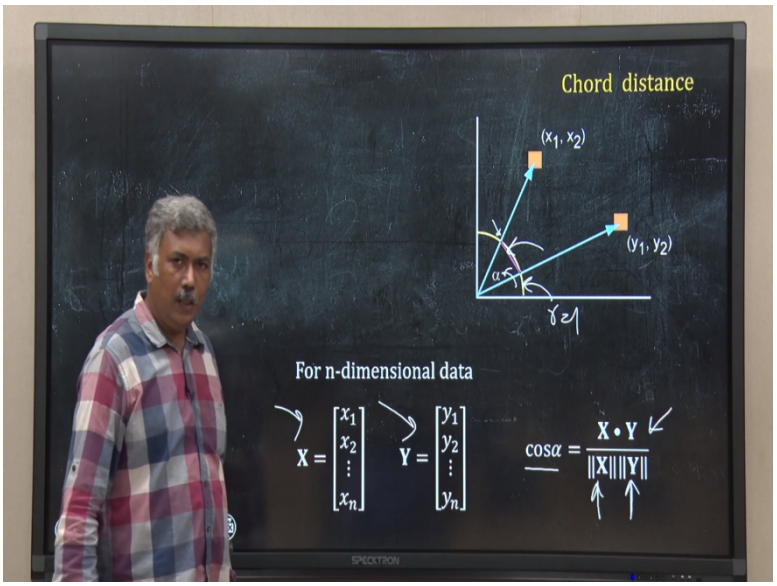
Look at this definition very carefully. If I consider that in these two-dimension in this direction and this direction, there is no correlation, there is no connection between these two linear some sort of dependence, there is no dependence between these two dimension and in both the dimension both the experimental condition your variance, variance is equal to 1. So, then if you take X data and Y data then how will the S look like? The covariance will look like? Covariance will look like something like this. You will have 1 here 1 here, because in covariance matrix your diagonal element gives the variance.

So, as I have considered the variance in both the direction is 1 so, I will have 1 here 1 here and as I have considered there is no correlation between the two-dimensional data then I should have covariance is zero. So, this is a identity matrix. So, if I put that here so it will be inverse of identity matrix and that will be also identity matrix and then this whole thing becomes same to Euclidean distance.

It will become X minus Y transpose of that into inverse of identity matrix into X minus Y and the square root of that. You can easily do the multiplication. You will realize this is nothing but a Euclidean distance because we can simply forget about it is equivalent to 1. So, when data is uniformly distributed, there is no bias and there is no correlation between these two experimental condition then Mahalanobis distance is essentially Euclidean distance.

But when there is some bias there is some value these are not zero, I have v1 something there and these are also not equal suppose this is A and this is B, then this covariance matrix will correct my error in my Euclidean distance calculation and it will be a better measurement of the distance between two objects or two data points. Now, till now we are working on essentially Euclidean distance, there is some sort of another distance called chord distance.

(Refer Slide Time: 13:29)



$$\cos\cos\alpha = \frac{X.Y}{||X||||Y||}$$

Let us explain what is that. In case of Euclidean distance if I have these two data points, I will be interested to calculate these distances is it not. That will be my Euclidean distance. But suppose this distance, linear distance is not important. For us, it is important that what is the angle between two vectors representing these two data points and that is what I have done here, I have blue vector for representing this data point, I have these another blue vector for representing this data point.

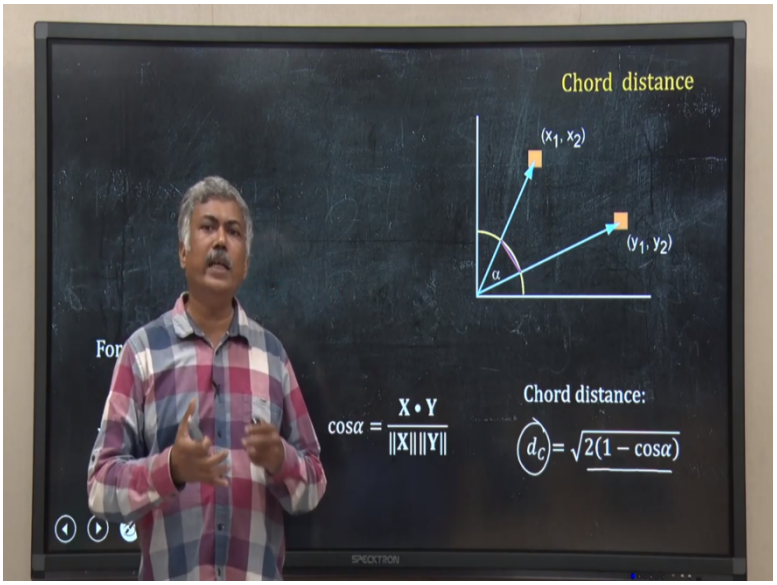
So, this is X and this is Y vectors. So, I do not want to know the distance, I want to know the angle between X and Y vectors. So, that will give me the angular relation between two data points. Remember in clustering, we are trying to find out some sort of relationship, some sort

of closeness. Here, we want to know how close these two data point in terms of the angle. So, how can I calculate the angle between X and Y vector? Very easy.

We have done vector multiplication in our one of our lecture. We have also calculated the angle between vectors in that lecture. So, if I have two vectors X and Y for two data points, then the angle alpha will be given by cos alpha equal to dot product of these two divided by the length of each of these vector. So, I know the angle. Now imagine these two vectors does not have any angle.

They are overlapping that means the angle alpha will be zero, then cos alpha will be equal to 1. So that means that's the closest, you cannot have more closer than this. So, I will define a distance in such a way that when the angle is zero, my distance measure will also be very small, the lowest one. So, that is where the chord distance come. We will define the distance between these two data points in terms of this chord shown by this pink line, this is a chord considering a circle of radius 1 unit circle. So, what do I get the, how do I get the length of that chord?

(Refer Slide Time: 15:43)



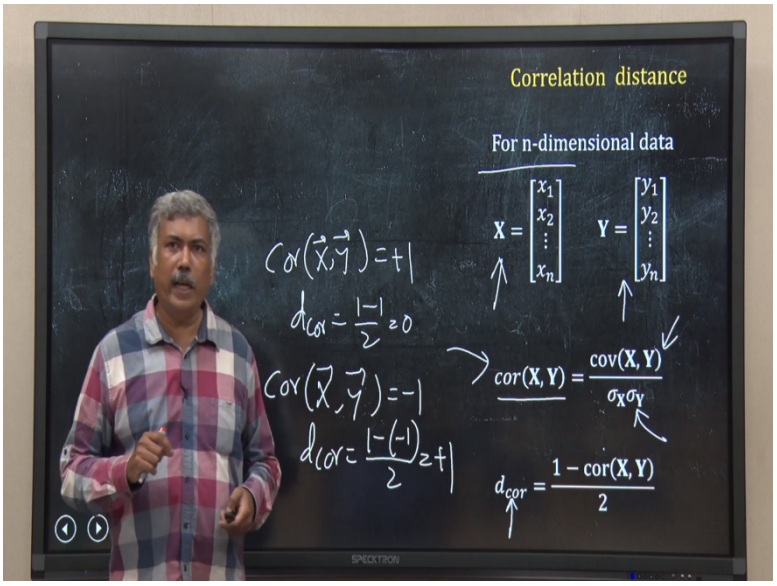
$$cos\alpha = \frac{X.Y}{||X||||Y||}$$

$$d_C = \sqrt{2(1 - cos\alpha)}$$

The length of that chord which we will call the chord distance is equal to 2 into 1 minus cos alpha. Alpha is the angle between these two vectors and the square root of that. You can easily check now, when the alpha will be 0 cos alpha will be 1 and the distance will be 0. So,

this is called the chord distance. There is another type of distance which does not again look into the linear distance in the space rather try to identify the statistical correlation between the data points and that is called correlation distance.

(Refer Slide Time: 16:14)



$$cor(X, Y) = \frac{cov(X, Y)}{\sigma_X \sigma_Y}$$

$$d_{cor} = \frac{1 - cor(X, Y)}{2}$$

Again, what do we have? We have the n dimensional data suppose and I have X vector that is for one data point and Y vector for another data point x1 to xn, y1 to yn. Now, I take these X and Y vectors and calculate the Pearson correlation coefficient as simple as that. In one of our lecture, we have learned about Pearson correlation coefficient. So, we calculate the correlation between X and Y.

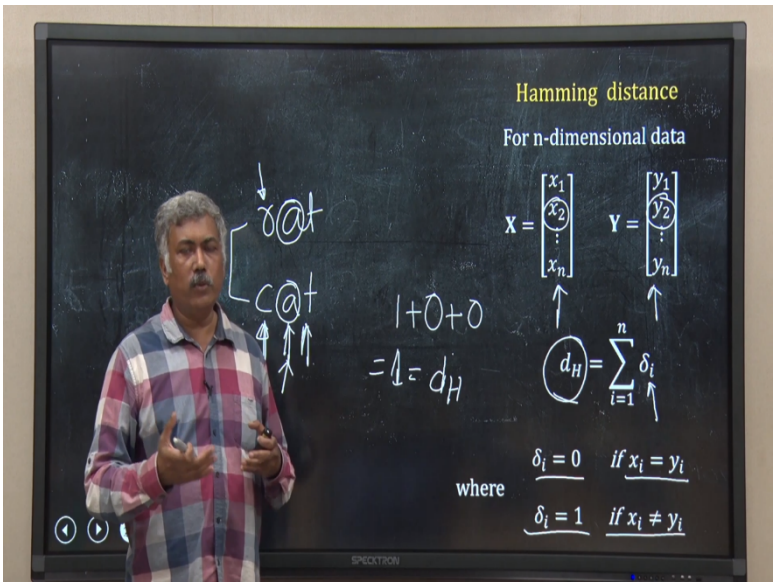
And if you remember from that lecture, Pearson correlation coefficient will be covariance between X and Y divided by standard deviation of X and standard deviation of Y. These are easy to calculate. In R you can easily calculate that. So, you can simply in one line command you can calculate the correlation between the, Pearson correlation between X and Y. So, we know Pearson correlation can vary from plus 1 to minus 1. Plus 1 mean we have a positive linear relationship, minus 1 means we have a negatively linear relationship.

0 means there is no relationship between these two, right they are independent. Now, that is how we define the correlation distance d chord is equal to 1 minus the correlation, the Pearson correlation between these two data points divided by 2. So, what will happen? When I have the correlation between X and Y is equal to plus 1 that means, they are positively correlated right then my distance should be low right.

So, then I will have 1 here. So, that means, I will get d correlation equal to 1 minus 1 by 2 I will have 0 whereas suppose I have negative correlation so they are not close but they have an association right. So, in case of that suppose correlation between X and Y is equal to minus 1, then my distance will be 1 minus minus 1 divided by 2, so, it will plus 1.

So, this is the another extreme, if I have a distance plus 1 that means they are oppositely connected right opposite from each other. So, I may keep them separate in separate clusters. Whereas, if the correlation is 0, that means they are independent right. So, the distance is 0.5 from 0 to 1 it is in the middle. So, this way based on the correlation, statistical correlation in your data set you can also measure the closeness between different data points and then use that to segregate data in different clusters.

(Refer Slide Time: 18:58)



$$d_H = \sum_{i=1}^{n} \delta_i$$

$$\delta_i = 0 \; if \; x_i = y_i$$

$$\delta_i = 1 \; if \; x_i \neq y_i$$

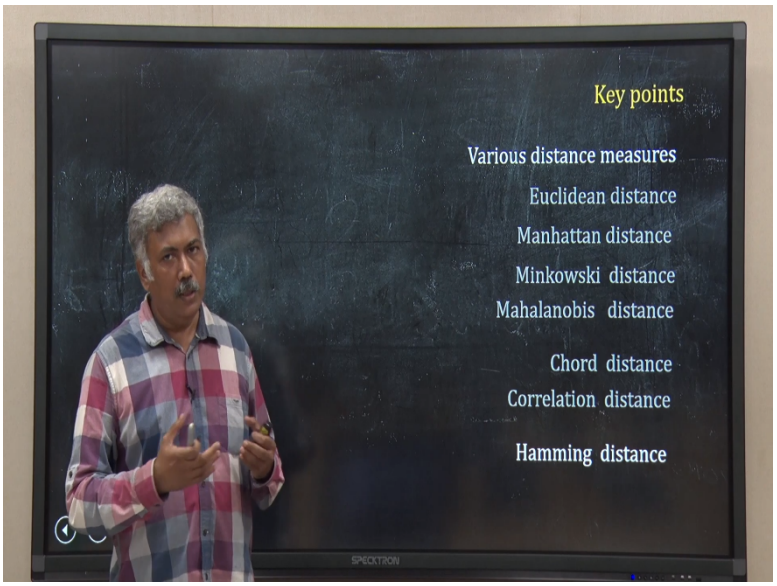
The last one that we will learn today in this lecture is quite different it is called hamming distance. And let me explain that using a something different not gene expression that or something like that. Suppose, I have two words we will call them strings right. So, I have a rat and I have cat. So, what do I do? I check each of these letters. Now both the word has three letters.

So, we say these strings rat and cat these two strings have the same length of three characters. Now, I have aligned them and I start from the first character and check whether this first character C in one string matches with the first character in the other. No, it does not match. So, I say 1. Now, I go to the second one and check whether the second character in my string matches with the second character in the other string.

Yes, they match. So, I say zero. That means they are similar, exactly same. Similarly, for the third one also you get 0. So, the summation is 1 plus 0 plus 0 is 1, this is my hamming distance. Hamming distance between these two strings. So, that is what I have written mathematically here. So, hamming distance between two data sets, two data points or two object in an n dimensional space if I consider and remember both have the same dimension 1 to n, then it is equal to summation of del i.

What is del i? Del i is equal to 0 if xi and yi are same, if these two are same or identical, then I will say they del 2 will be 0 that way we have done for a and t in this case. If they are not equal, then I say del i equal to 1. So, it is the binary case and you are summing those binary digit 1, 0, 1, 0 and the whole summation, the result is actually called the hamming distance. Each of these distance measure has their limitations, has their specific uses. Based on that we have to use it, when we will use those for a particular problem we will discuss why we are using that particular type of distance measure for our work.

(Refer Slide Time: 21:38)

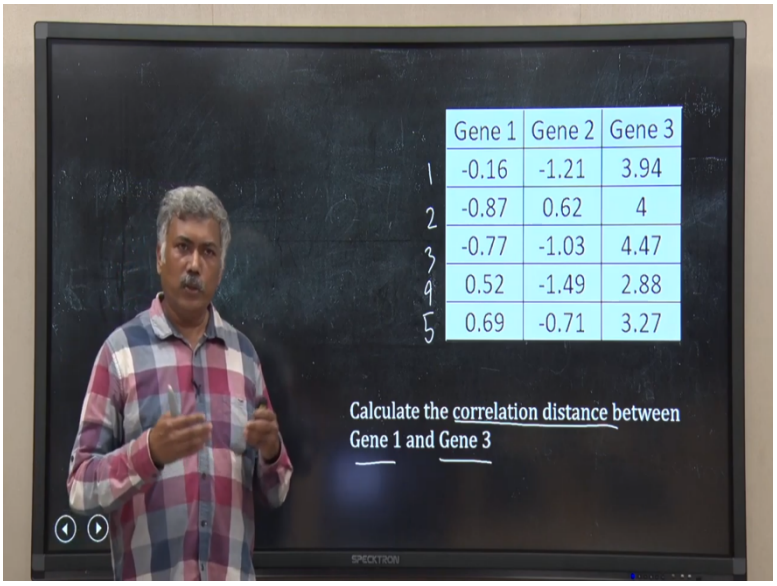


Let me jot down the key points of this lecture. The most common distance measure that we use for our data analysis in clustering is usually the Euclidean distance, generalized form of

Euclidean distance Manhattan distance is Minkowski distance. And in many cases, it is better to use a weighted distance measure and Mahalanobis distance is one of the common method which is a form of weighted Euclidean distance which is used for machine learning in clustering.

Then apart from this linear distance-based thing, we have also something called chord distance which measure the angle between two data points or objects. And also, we may use correlation distance which is based upon the statistical correlation in your data. And at the end, we have discussed our hamming distance which is essentially binary and in some cases, if you have some qualitative data, it is much better to use hamming distance as a measure of distance.

(Refer Slide Time: 22:44)



Gene 1	Gene 2	Gene 3
-0.16	-1.21	3.94
-0.87	0.62	4
-0.77	-1.03	4.47
0.52	-1.49	2.88
0.69	-0.71	3.27

That is all for this lecture, I will leave you with a particular problem to solve. Suppose, I have done a gene expression experiment and I have 1, 2, 3, 4, 5 experimental conditions and I have measured multiple genes expression, I have shown just part of it one gene 1, gene 2 and gene 3, what do you have to do? You have to calculate the correlation distance between gene 1 and gene 3, I hope you will try it. You can use R or any other calculator to calculate this one, you

know the definition of correlation distance by now. So, I hope you will be able to do it. Till then happy learning. See you in the next lecture.