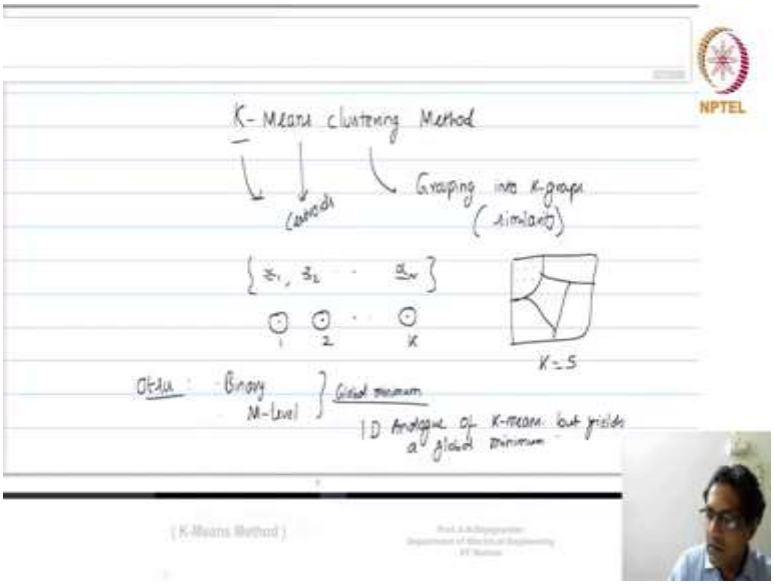
Image Signal Processing
Professor A. N. Rajagopalan
Department of Electrical Engineering
Indian Institute of Technology, Madras
Lecture 61
K-Means Method

(Refer Slide Time: 00:20)



So, in this class we go to look at what is called K-Means Clustering Method, K-Means Clustering Method. So, when somebody says that no that he wants to do a kind clustering, so this is some kind of grouping. So, grouping into K groups, that is what it means K groups, which also means that you need some kind of notion of similarity. We will say that right you going to group things together; we need to know how to actually group them.

So, which means that we need to have some idea about what do we mean by two objects being similar and so on. And these means they are also are referred to as centroids and this method assumes that there are actually K centroids. This K-Means you have to assume that there are K centroids. Now, what exactly is this problem? So, what does it means is that suppose you have a set of data samples.

Let us call them x1, x2 right up to xn you know large number all of them. Let us say n is very large and you want to actually group these data samples such that they all fall within K groups. And they should satisfy something in the sense that you like the most similar things to occupy

one group. And similarly those samples that look again very similar together should occupy another group and so on.

You are like one group, two groups and then you got you got K groups. The average of each of these groups that each has a mean value and that is why we refer to this as K-Means. There are Kth centroids and this could be scalars or on the other hand this could also be vectors. And this could range all the way from pure gray level intensities to occur to color vectors. So, it could be an RGB kind of a color vector, or it could also be a feature vector for all we care.

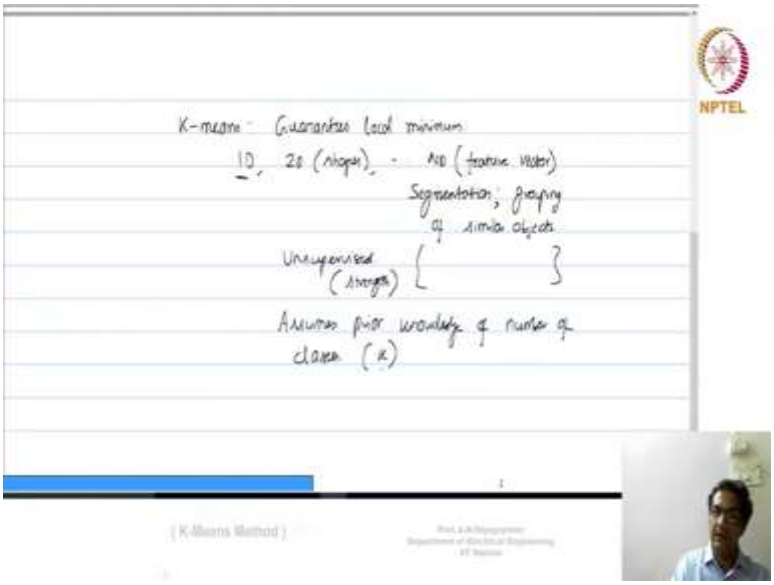
For example, you might want to think of an image like this; wherein let us you got one portion is land, another portion is water, another portion is vegetation. Something else let say then there might maybe you got something else here. And so right, your idea could be that you want to divide the dataset here into 1, 2, 3, 4, 5 groups. So, it say your K is 5. So, now what you could do is you could, so each of this each of these values that you have here would be your x1, x2 and so on; and you got a kind of right group them together into be K number of classes.

So, this K means given that it is kind of a grouping method and therefore not finds a lot of applications. It is actually a powerful method which is why I thought I mean I will speak about this particular scheme. And also and that has relation with the also thresholding that we saw earlier. Now, if you if you recollect Otsu, Otsu when we did the when we did a thresholding we actually derive the equations for the case when we have to binary thresholding.

And also (())(03:39) indicated how you could extend it through and to an M level kind of a kind of a thresholding. Ofcourse, we realize that both would require exhaustive search, and in fact M level would make it very-very tight complex. But then right this guarantee a global minimum, once you hit each of those values, threshold values you are at least guaranteed a global minimum.

But this is was an exhaustive search. The K-Means clustering and so in that sense it also works with histogram that the (I mean) which is way also is actually concede. So, that sense you can look up on Otsu as some kind of a one dimensional analog of, right one dimensional analog of K-Means. However there is kind of a difference with respect to Otsu, its one dimensional analog K-Means; but, yields a global minimum.

(Refer Slide Time: 04:51)



However, K-Means guarantees local minimum. However, you know as you will see K-Means is far more efficient; you know in the sense that it is far more quick it is far more general because it can deal with 1D data, it can deal with 2D data in the sense that you could have shapes and so on which you could incorporate. It can be some n dimensional data which could be just a transform, some feature vector and so on.

So in that sense we did not be hung on 1D, we can have whatever kind of data rate which we want to do grow which is why I said that x1, x2 up to xn this could be vector, this could be scalar, this could be vectors and so on. Now, one more thing that we should realize is that is that so in that sense you can kind of think upon K-Means is doing a kind of a segmentation, grouping; whatever you might want to call it segmentation, grouping of similar objects.

So, in the sense what we mean by segmentation is grouping of similar objects. Now, one other thing that we should realize is that K-Means ofcourse as I said guarantees local minimum; and K-Means is unsupervised. It is an unsupervised method in the sense that once you kind of give those data samples it automatically figures out what should go where. Ofcourse has some kind of an automatic criterion which I will explain, along the way.

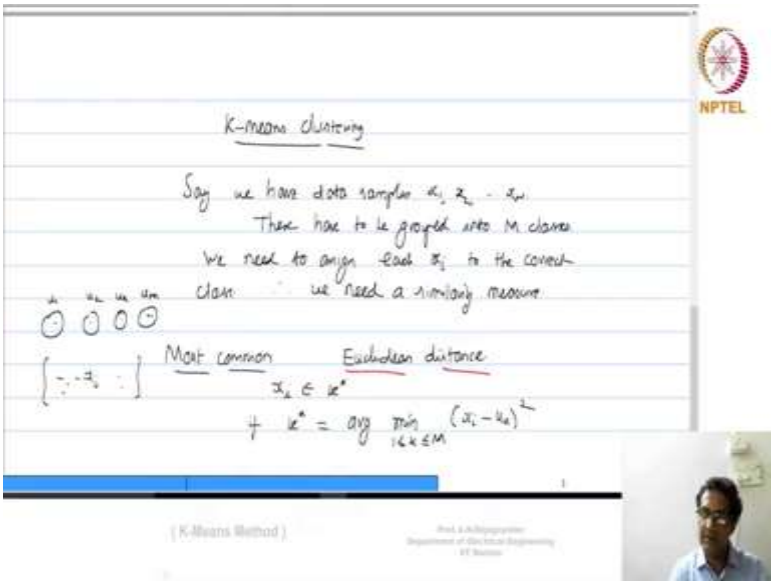
Ofcourse, the one weakness of K-Means is that it assumes prior knowledge of number of classes, so assumes. So, it is unsupervised which is actually its strength that is its strength. However, at

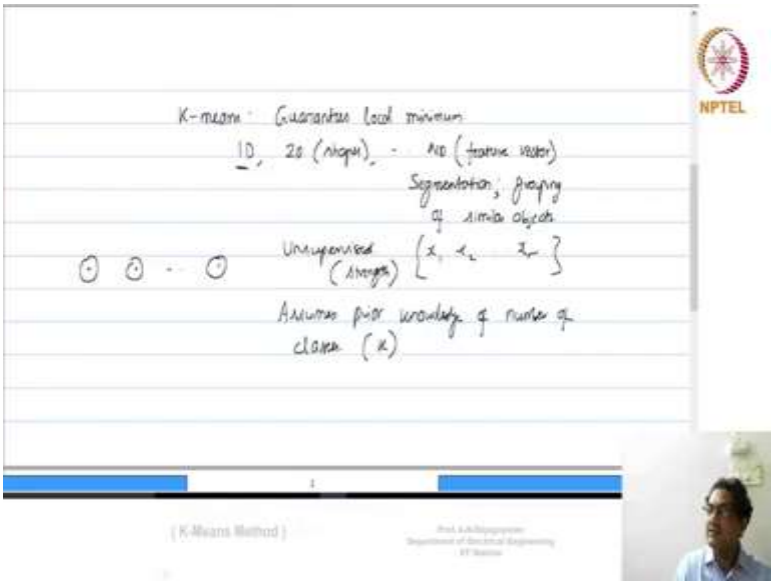
the same time it assumes prior knowledge of number of classes, knowledge of number of classes and therefore so because of the fact that it assumes prior number of classes, which is K.

So, in that sense you might say well how do I know precisely what my K is and so on. So, right there is an extension of K-Means called ISODATA, which I will explain later. So, basically there is there is a way out of this, so even if you do not know K exactly. You could start with an approximate K and then you could decrease increase and so on along the way.

But, that is not K-Means that is called ISODATA; I will talk about it later. That is the kind of built along K-Means, which has some additional flexibility; so, as of now we just realize that we need to know the number of classes. Now, how does this K-Means clustering work? Now let us kind of go through what it does.

(Refer Slide Time: 07:51)





So, as I said one of the first things that which you really need is a notion of similarity. Assumes that say we have a data samples I will start with the scalar case and then we can actually see what will happen. Here vector x1, x2 all the way up to xn indicate as we said earlier; we want to group them into these have to be grouped into let us say M classes; these have to be grouped into M classes that is our goal. So we need to assign, so well I just want to also mention that this can also be looked upon as a labeling problem.

So, some people even can look upon this is some kind of labeling problem. So, by labeling what we really mean is each of these groups that you have, you can think of each of it as a label. And

therefore when you have this data x1, x2 all the way to xn you can get a think upon x1 being assigned label right assigned the label okay that is the mean of the…that is the centroid of this cluster.

And second guy gets another label, then the then another for another data sample; sample gets another label whatever and so on. So, the sense it can this also called a labeling problem. Now, we need to assign each xi to the true class or what is called a correct class. And therefore we need a similarity measure, in order to this we would need, but let us say it is really a similarity measure.

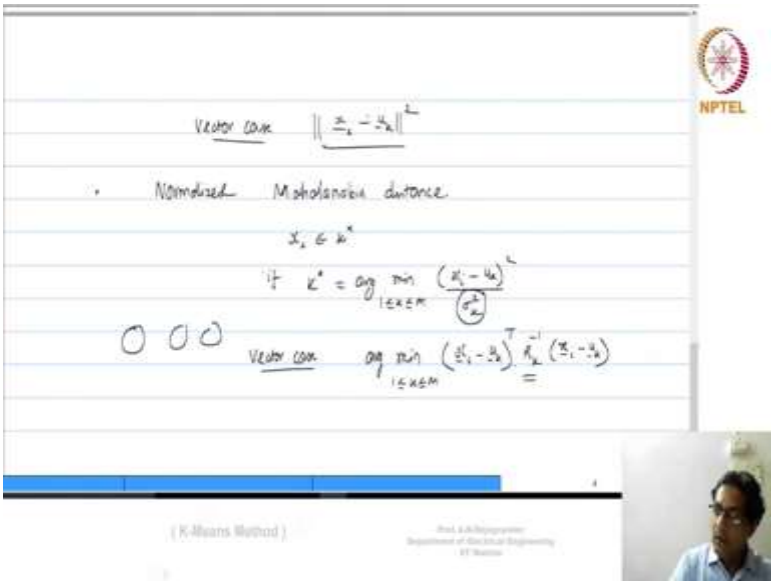
Now, I will go through a few of them but then the most common and the most effective in fact in almost all cases a simply a Euclidian distance. So, we still have not seen the algorithm is yet; but just to know as to what kind of similarity measures it can use. So, we say that so in a sense this is the most common Euclidian distance; because it is very simple to compute.

So, xi we will say belongs to K star that means it belongs to group K star; if K star is equal to arg min 1 less one less than or equal to K less than or equal to M and xi minus mu K square so what this means is that you got like K-Means. So, you got like mu 1 to be the mean and this mu 2 to be the mean of this and let say mu K to the to be the mean of okay.

Right, mu K to be the some other Kth group and then all the way up to let us say mu m, so let us call this as mu M; so we have this, we have the set of samples, so let us say I pick some xi arbitrarily. I check is this thing closest to mu 1 or mu 2 or mu K or whichever one; I run through all of them and then the one with respect to whichever centroid. And this has the smallest Euclidian distance.

So we say that so we assign xi to that group. So, which is what we mean by xi actually belongs to K star; so we drop xi here. If let us say xi appears to be closest to mu 1; let us say xi appears to be closest to mu 2 then we will drop xi there. And then the same thing we would have to repeat with all data samples.

(Refer Slide Time: 11:57)



Well I mean so if it was a vector ofcourse then the vector case would have been simply that was a scalar case. And the vector case would have been simply norm xi, minus mu K norm square. We have to simply look at, look that this as a similarity measure and then search to over all K's for a given xi and then whichever mu K. This xi has the smallest it is kind of closest in the Euclidian distance; but set xi belongs to that group. Now, another thing what is called so another metric is called the normalized Mahalanobis distance.

So, it is nice that it is named after an Indian, normalized Mahalanob distance (())(12:47) Mahalanobis distance and this takes a variance into accounts. So the way so for the scalar case, what we will do is we will say xi belongs to K star; if K star turns out to be arg min 1 less than or equal to K less than or equal to M, xi minus mu K, the whole square by sigma square K.

So, it actually means that the so now so in addition you are trying to take variance into account which then means that if you had a vector case. If you instead of the scalar case, we had a vector case that would mean that you are going to look at arg min 1 less than K equal to M. Then you would have xi minus mu K, ofcourse this should all evaluate your scalar transpose R K inverse xi minus mu K.

So, this is a row and then this is a column and then in between your RK inverse, if this is a covariance. Now, but then as we can see it makes more sense to see variance also right it goes back to the joke on punches statistician that they said we would drown in a swimming pool,

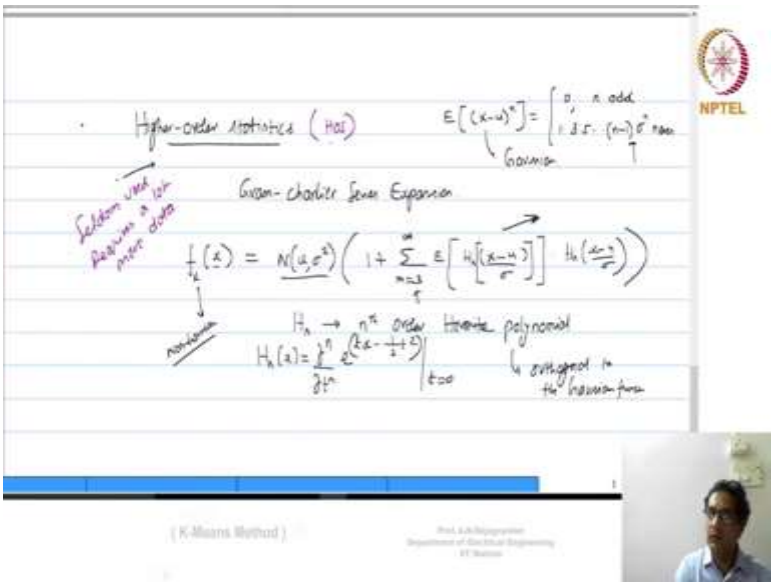
because they found because of whose average height was 5 feet but then it had a variance of actually 3 feet, so it is like that.

So if you ignore variance, it may not be a wise thing to do. And therefore this takes the variance into account, but then and also it puts a stress on your method because it also means that you need to have a good estimate of the variance. If it is scalar maybe it is still possible; or it is a vector you need a co-variance matrix. And especially for each group and do we have enough number of samples in every group in order to be able to that?

Could still be a question, my question mark. And therefore this kind of a matrix is used only when you are confident that you can get a robust estimate of R K. If you cannot get a robust estimate of R K, then we could not even attempt it. In fact we will end up doing worse than Euclidian if we have bad estimates of RK or mu or let us say sigma square K.

Therefore, this should be attempted only if you have enough number of data samples in every group. And you feel that we can actually make robust estimates of covariance mean of course you can do it, but then you need also good estimates of the covariance.

(Refer Slide Time: 15:29)



Now, a further extension of this is actually what is called higher order statistics based metric. Now, the one that I would like to mention that is a generalization of what we have seen is what does what was based on the Gram-Charlier series expansion, which is based on the Gram-

Charlier series expansion, and it goes as follows. Well, anyway this is only for information sake, this kind of metric is seldom used.

But, then the fact is that this is just interesting to know that arcade of good to know that such a thing exist and this Gram-Charlier series expansion, I will write it down for the scalar case, a vector case is very complex. So, I just write it down for the scalar case or what it says is that, if you can model fx.

Fx which is PDF of let us say kind of variable x; in this case I am taking a scalar as N mu sigma square. Where, let us say this is the mean of x and sigma square the variance of x into 1 plus summation m equal to 3 to infinity; expectation Hn of I think x minus mu by sigma. It is enough x minus mu by sigma and into Hn of x minus mu by sigma and then close bracket.

Now, if you are wondering what does Hn is? So, this Hn is the nth order Hermite polynomial; nth order Hermite polynomial, and this is just for information sake. Hermite polynomial and it was basically given as Hn of x, is e raised to tx minus half t square and dou n by dou tn in fact let me this raise and write it a little bit here; so this is the nth derivative partial derivative with respect to t. And evaluated at t equal to 0, that is your Hn of x.

And this is a Hermite polynomial; so, this whole thing is a bracket now. This is a Hermite polynomial and this expansion this kind of see interesting. So, it is like saying that it incorporates higher order statistics when needed to your similarity measure. So, if you feel that x is x is non-Gaussian, so right so this not really Gaussian which actually means that you would like to incorporate higher order statistics.

Like to bring them into the picture when you compute a similarity metric. So this term when it goes from n is, this goes from n equal to 3 to infinity and this indicate the higher order moments. If you had a vector then what it will indicate joint moments and so on; and then all this expression will become far more complex.

But, then it the key sort of take away is this, that in case x happens to be a Gaussian; let us say we do not have a priori, you have a dataset and then you try to make an estimate of this fx of x. If x happens to be a Gaussian, then you can show that all these terms, which owns each one of these terms will equivalently be 0. Because of the fact that Hermite polynomial is in fact

orthogonal to the Gaussian function. This orthogonal to be this can all be shown to the Gaussian function.

And therefore that equivalently the entire term will go to 0 and then what you will get is n mu sigma square which, now which can be shown to be a special case of Mahalanobis Euclidian that will turn out to be a special cases of this. On the other hand, I mean that if you had higher order statistics as you can see Mahalanobis and all ends with sigma square which is the second order statistics but then this you can go higher order.

But, at this stage let me also tell you that this higher order is not something like for example a Gaussian. If you had I mean you could argue that that that even let us say Gaussian has higher order statistics. For example, if you do it exists a Gaussian, and suppose I did x minus mu, if I if I computed the nth order central moment. We know that this is equal to 0 or if n is odd and if n is even it turns out and that will be 1 into 3 into 5 into all the way up to n minus 1 sigma power n, if n even.

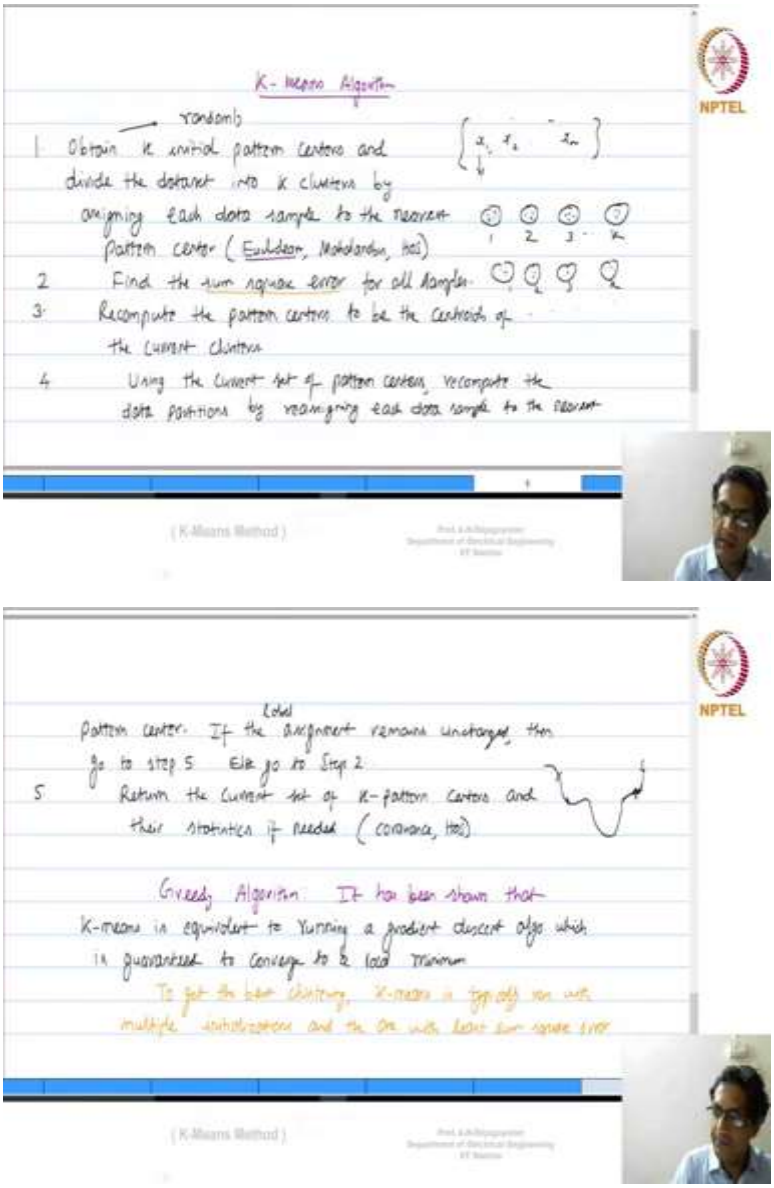
Now, this also ofcourse will be even if x is a Gaussian, we know that higher order moments are not 0. At least the even moments are not 0. But then the point is right these are again a function of your sigma square; so which is why it is called sufficient statistics. So, these higher order moments do not bring in any new information. This if you take n equal to 3, n equal to 5 and so on; they do not really n equal to 4 or 6 or whatever.

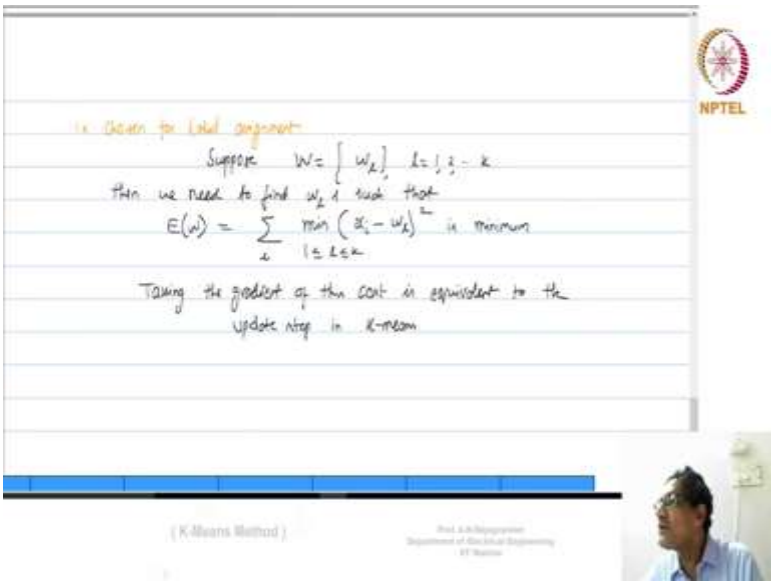
They do not really bring in begin anything new; because they are again back to bring they are again a function of sigma. And therefore it, therefore you do not you do not really gain anything by using higher order statistics and all in the in the Gaussian. But, this right whereas in this expression you actually get higher order statistical information.

So, if it is a Gaussian, then this will equally reduce to 0 so that you do not end up adding anything new. But, if it is non-Gaussian, then it will end up adding some new information which is coming from higher order statistics and that can be used. But, this let me just emphasize that this is seldom used and also it requires a lot more data; because now we right now are talking about higher order statistics.

So this also called HOS seldom used but then as you can see this is a neat generalization coming through a Gram-Charlier series expansion. And this generalization subsumes the Mahalanobis, subsumed Euclidian and all. So, that is it is just worth noting that such a things exists.

(Refer Slide Time: 21:43)





Now, at this point of time let us go and look at what is the K-Means method; K-Means algorithm or K-Means steps in K-Means or let us call this algorithm. And I am going to list these steps and then along the way, I will explain the walks with each of those steps mean. I will also explain what each of those steps mean, so the first thing so the steps in K-Means. So, the first step is so you have a dataset that is going from x1, x2 all the way to xn. The first step is obtain K initial pattern centers and divide the dataset into K clusters by assigning each data sample to the nearest pattern center.

So, what does this means is you know obtain K initial pattern centers; so this is obtained randomly. So, you do a random pick so randomly pick key of these guys; because you want to do, want to group them into k groups. So, what you do is so you randomly pick K centers so as you pick K centers you do not have anything in these groups as yet. Because, all that you have are, you have kind of randomly picked what you believed could be likely centers.

And then what you do is now you start, now once you have picked those K centers randomly then you start to revisit the entire all the samples now. Look at x1, see as to where it goes, is it closer to this mean that you picked up, that mean that you picked and so on, and then assign it. Then go to x2 see whether it goes here, it goes here it goes here and so on. So, basically that way you assign all of them into these K groups; they are like 1, 2, 3 all the way up to K groups.

Then 2; find the sum square error for all samples. So, basically what this means is that if let us x1 has a smallest error with respect to the second centroid then you would actually find out x1

minus mu 2 whole square. And similarly you would find out for x1, for x all of them and then sum up all the errors that you have. Because these errors need not be 0, so just find the sum square error for all the samples. So, I will tell you where this is useful, where we use this information.

3; re-compute the pattern centers to be the centroids of current clusters. So, what this means is that once you have once you have assigned, so once you have assigned all these; so you would have got some samples here, some samples here, some samples here, some samples here. Now, since you have some of setting inside each one of these groups, you kind of find to re-compute the average value.

Because initially this started randomly with some pattern centers, now you got actually data sample sitting in that. So, you can compute the average of all these guys and re-compute the pattern centers. So now what you have are some, so you have a new set of means 1, 2, 3 all the way up to k; so you have a new set of these pattern centers.

Now, re-compute the pattern centers, now 4. The fourth step is, so once you recomputed them using the current set of pattern centers that means using these pattern centers, using the current set of pattern centers, re-compute the data partition. So, you can kind of look upon this is a kind of data partitioning scheme. Re-compute the even data partition by assigning, by reassigning each data sample to the nearest pattern center.

Each data sample to the nearest pattern center, so what is that mean? So that actually means that mean have a new set of mean. Now, again you start with x1, x2 all the way up to so on; so it could so happen that x1 perhaps the first iteration it might have been assigned to x2, but now it could get reassigned to something else. It could come into x1, it will come to x3 or could stay in x2 and so you kind of reassign.

If the centroid assignment, if the label assignment of the centroid assignment, the assignment, label assignment remains unchanged let us call this label assignment remains unchanged that means all x1 to xn go back to the same groups from which they came, remains unchanged. Then we are kind of done, then go to step 5.

I mean which means you are actually done else go to step what is that, go to step 3. So, step 3 if we go back here, so kind of step, so it is relevant I mean should really go to step 2 okay and let me write it as step 2, so because right now what again; what we can again do is we can compute your error and then again go on.

Then again we compute what this means, after this right you might have you might we again have a new set of data samples coming in, new data samples coming in, new data samples coming and again re-compute the average and many more, until the label assignments is unchanged. What is step 5 right we neither said what is step 5; so return the current set of K patterns and there K patterns centers and their statistics if needed.

If needed get the sense that it just could probably you want the co-variance, if it is available HOS and so on. Only if needed otherwise you could ignore this, we just would be interested in knowing the centroids. Even here when I said assign to the nearest data sample we could use Euclidian as here as a simplest thing to use, which is also effective. Or, you could go to Mahalanobis or you could use HOS and so on.

But as I said the most common one that is used is Euclidian. Now, there now this is kind of a greedy algorithm, the K-Means is kind of a greedy algorithm. It is been shown that well we will not kind of go through the proof. But, it is been shown that K-Means is equivalent to running a gradient descent gradient algorithm which is guaranteed to converge to a local minimum.

So, what does actually means is that depending upon your initial assignment, so your cost right here because you are in some n dimensional space. Your cost could be something arbitrary like that. So if you start with your with an initial initialization from there you could actually get stuck here because it is a greedy algorithm. So, it will not be able to see that there is a global minimum there and so you will simply stop here, when the cost can have begins to rise.

And similarly if you now changed your initial assignment and probably start from here then it is very likely that whatever it, let me just redraw this maybe it is like this, then n then it could kind of right and get stuck there. Therefore depending upon what kind of initialization you do so initialization that means here; obtain when you random, due to this random initialization for each initialization where final label assignment need not be the same and what is normally done is and because of this sensitivity to the to the initialization of K-Means.

It is taken care off by actually taking, by actually running this algorithm with several initializations. And then picking the one that actually gives you the smallest sum square error; that is a reason why the sum square error is actually computed. Because when you try with different-different initializations and then run the K-Means algorithm you will get, you can get assignments that keep changing depending upon initial assignment.

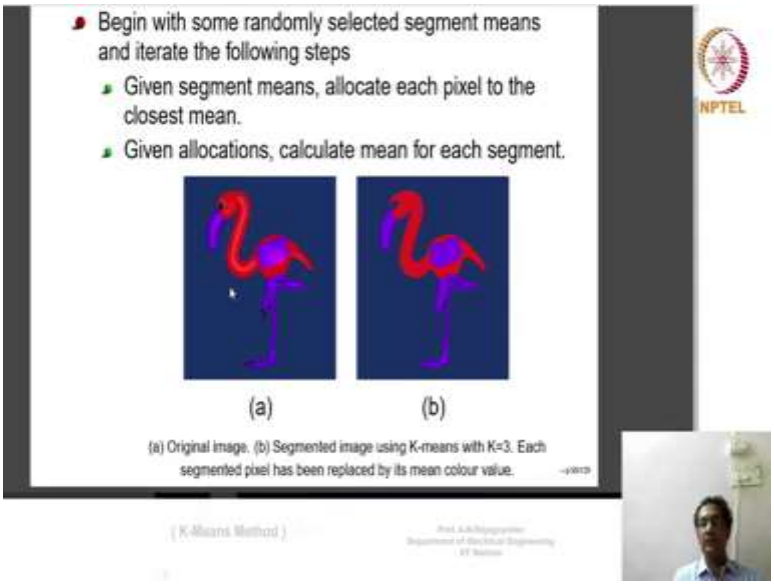
But then you finally pick that set of labels for which the sum square error is the smallest. So, let me just write this down as was a point to get the best clustering; to get the best or whatever label assignment or clustering results. K-Means typically run with multiple K-Means is typically run with multiple initializations and the one with smallest mean square error. The one with the smallest, the one with least error, in the sense the least sum square error is chosen for label assignment which makes sense.

Now, as I said I mean so it has been shown that this equivalent to a kind of data, ideal but descent algorithm. So suppose W is some set of prototypes Wl where l equal to 1, 2, 3 up to K; then then we need to find Wl's such that it is a E of W. Suppose I call some quantity if W is equal to sum over all I min where I is I goes over the number of samples runs over the number of samples min, xi minus Wl whole square, 1 less than or equal to what is that l less than or equal to K is actually minimum.

So, you need to find out so because the goal is to find out Wl such that xi minus Wl when it is computed over compared with all Wl is when then you compute this sum square error that happens to be the minimum. So, it is been shown that taking the gradient of this cost is; we will not enter into the proof of this. But, then by taking the gradient of this cost is equivalent to the update step in K-Means; going to the update step in K-Means.

Now right I mean so the point is ready fewer, if you were to run K-Means and suppose we want to see what kind of a result it can actually give you. So, I will show you an example for that; let us just go and go and take a look at one example.

(Refer Slide Time: 36:12)



Let me say now this is actually a segmentation problem and so you have this crane and you want to be able to and you and you would like to segment it into some K number of colors or K number of regions.

In this case we will see which is a color, so right so in this case if you see this; so this segmentation result using K-Means is for let me see what is this K. So, K is equal to 3, so you can see that is one color here, one color here, and there is one color here. Now we can see that you can see all these intensities have been either clubbed much with this or this or with that.

Now, some people might just like an output like this, and they will say this is fine. But, then this is this being an image enhancement kind of task; there is no reason to believe that K equal to 3 is optimal. Somebody might actually want to go with K equal to 4 and they might say that I would like these colors to emerge, emerge kind of separately. They should probably be assigned a separate class and so on; and simply even here okay. It looks like there is some sort of a different color that is sitting there.

And therefore you might want to get (())(37:20) as a fourth color. In this case it says one more sort of a class or color. Now that is again up to you. So, some people might be happy with this, some people might want to choose K equal to 4 and so on. But, then the idea is that here many even though this can only guarantee a local minimum but still because of the fact that you can run into multiple initializations.

And choose one which gives you the lowest error and go with the label assignment that you get out of that. I mean nice thing about this algorithm is that it is fast and you can do, I mean you can work with 1D, 2D whatever n dimensional feature vectors. And even a simple Euclidian distance, it works quite well. An extension of this is actually ISODATA, which does not assumed knowledge of K, and I will talk about it next.