Introduction to Machine Learning

Prof. Mr. Anirban Santara

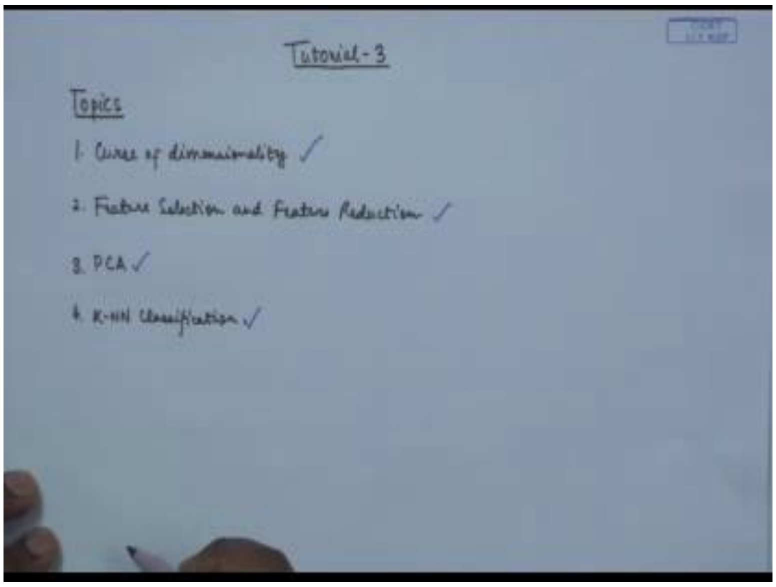
Department of Computer Science and Engineering

Indian Institute of Technology, Kharagpur

Tutorial III

Hello friends. This is Anirban. Welcome to the third tutorial session of this course. In this session, we are going to summarize the topics that have been taught during the week and we will discuss what all questions can be expected from this session in the exam and in the assignment; and we will also discuss how to solve them. The topics that we are going to cover today are curse of dimensionality; so, what it is and why it is at all important for consideration and how this leads to the concepts of features selection and feature reduction.

(Refer Slide Time: 00:41)



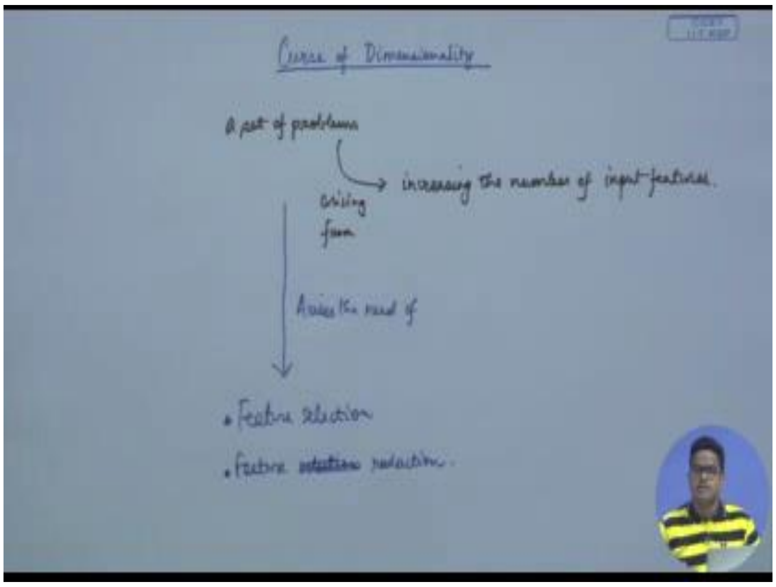
Tutorial-3

Topics

1. Curse of dimensionality ✓
2. Feature Selection and Feature Reduction ✓
3. PCA ✓
4. K-NN Classification ✓

And in the context of feature reduction, we will consider principle components analysis. We will study the process of finding the principle components and how to use them in the final machine learning tasks. And finally, we will take up k nearest neighbor classification and I will also hint how k nearest neighbors can be used for regression problems. Without further ado, let us take up the first topic of today, which is the curse of dimensionality.

(Refer Slide Time: 01:17)



Curse of Dimensionality

A set of problems

arising from

$\xrightarrow{}$ increasing the number of input features.

$\downarrow$ Arises the need of

- Feature selection
- Feature ~~extraction~~ reduction.

Curse of dimensionality is a set of problems which arise from increasing the number of input features. As the input, number of input features increases, as we are using, as we introduce more and more sophisticated and more and more diverse features to describe the input of, to our learning system, what happens is, the system becomes highly prone to noise that is present within the universe; maybe when, maybe this an example could be the use of, say you have a robot, which tries to predict the kind of discussions that are happening in a particular email thread.

So, it reads the emails that have been sent within a thread and it tries to predict what kind of, what is the topic of discussions happening in this thread. So, if we ask the robot to look at, say, just the words within that email and then figure out on the basis of the, of what kind of words are more frequently being used in the emails of that thread, trying to predict what kind of topic is getting discussed in that particular thread, if you just ask the robot to look at, say 10 such words, then each email will be described in terms of the presence, or absence, of those 10 words. So this robot will be, will make some prediction. Now, to make it more sophisticated, a bit more capable of identifying better topics, maybe we can ask the robot to look at 100 words, 100 chosen words.

In this case, what happens is the number of dimensions of the input data increases. So, the input data over here, in this case is an email. And it is being described by a set of words. So, number of words that are being used to describe the email is the dimensionality of the feature vector of the email. So this is increasing. And as a result of this, what happens is, you do not have all of these 100 words in all the emails that are coming to you, right. So, 10 words could, could be, could co-occur more in an email, than all the 100 words that have been specified. So, what happens is, as you increase the number of features to describe your data that number, amount of training data you have, falls, falls short.

So, say, you have n dimensions of your input feature space. So, these, as the value of n increases, alright then, number of examples that would be required to modular function in this n dimensional space increases exponentially; and it goes like this that, to model statistics in a feature space efficiently, so, which are sound, you need enough training examples, you need enough samples, from that particular space, right. So, when you are trying to model some statistical phenomenon, you should have an adequate number of examples in support of that; then only, you can have a proper, like legitimate modeling, legitimate estimation of the statistical parameters.

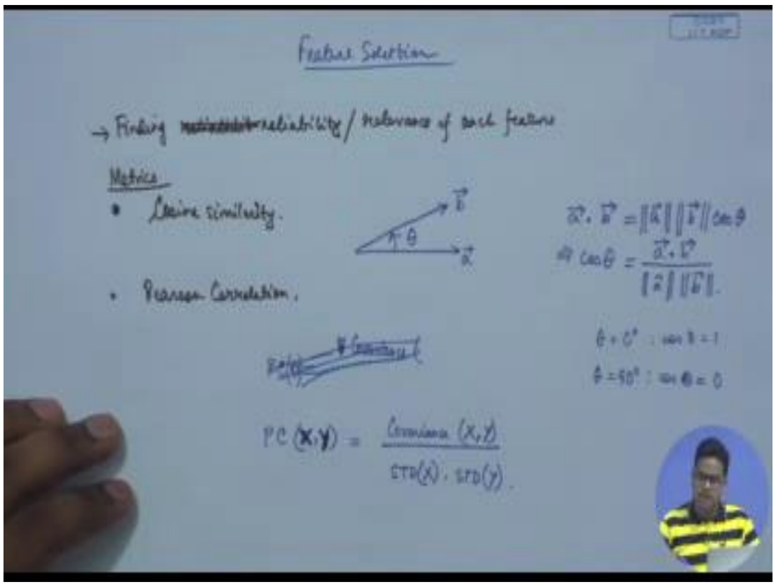
Now, if you have too much dimensionality of you feature space, alright, the number of examples which would be required to have a legitimate estimation of a statistical parameter becomes exponentially large, which is not always possible, because, the availability of clean and good and really valuable training data is costly. So that is why, we have to perform feature reduction, or dimensionality reduction, in some cases. And this is an important step in almost all machine learning applications.

We may select some 1000 features to start with, but all of these 1000 features may not be equally relevant, or equally effective, in making the prediction that we are looking forward to. So, we should be able to, like reject some of them; like, maybe, we have, if we start with 1000 features, maybe we would like to cut the number of relevant features down to, say 100. So, 900 features, which are less relevant, or less effective, for making the particular prediction, should be rejected. So this is all about feature reduction, or feature selection and which is our dimensional reduction in other terms, which is all about making sure that the amount of training data that you have, the amount of samples that you have collected, is adequate for having an, a reliable estimate of the statistical properties that you are trying to model. So this is the entire idea behind dimensionality reduction.

As a result of curse of dimensionality this set of problems, then arises the need of feature selection and feature reduction, which are interchangeably used as the same term, or you could. There are some little differences, subtle differences between feature reduction, these two things; these two terms. And feature selection, or reduction, as I said before, it helps in removing irrelevant or redundant features and if you have insufficient training data then, then you need to perform feature selection and if you have insufficient computational resources, you should use feature selection. Why, because, the complexity of your model becomes more and increases, as the number, feature dimensionality of your input increases. So, if you have say, 1000 dimensional input vector then, you should have, the model’s complexity, the number of parameters of the model will automatically increase.

So, as the dimensionality of the input increases, the number of parameters of the model increases; and so, chances of over-fitting also increase. So that is why, you should look forward to, you should, like, plan some kind of feature reduction and feature selection if possible, in any machine learning application. So, some, some feature selection methods are using.

(Refer Slide Time: 08:33)



Feature Selection

→ Finding reliability / relevance of each feature

Metrics

- Cosine similarity.
- Pearson Correlation.

$\vec{a} \cdot \vec{b} = \|\vec{a}\| \|\vec{b}\| \cos\theta$

$\Rightarrow \cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \|\vec{b}\|}$.

$\theta = 0^\circ : \cos\theta = 1$

$\theta = 90^\circ : \cos\theta = 0$

$PC(X,Y) = \frac{\text{Covariance}(X,Y)}{STD(X) \cdot STD(Y)}$.

So, let us go to feature selection. So, you can do feature selection by finding reliability, or relevance, of each feature. So, how do you do it? So, there are some metrics, like cosine similarity; then, Pearson correlation. So, cosine similarity is metric which can be used for finding the similarity of orientations of 2 vectors in a inner product space. So, an inner product space is a kind of, is a special kind of vector space, in which an inner product is defined. So you should look up the web and figure and find out the exact definition of an inner product space. So, in an inner product space, what we can do, we can use cosine similarity this particular metric, to find out the similarity in orientation of 2 vectors. So, in simple terms, say, you have 2 vectors a and b.

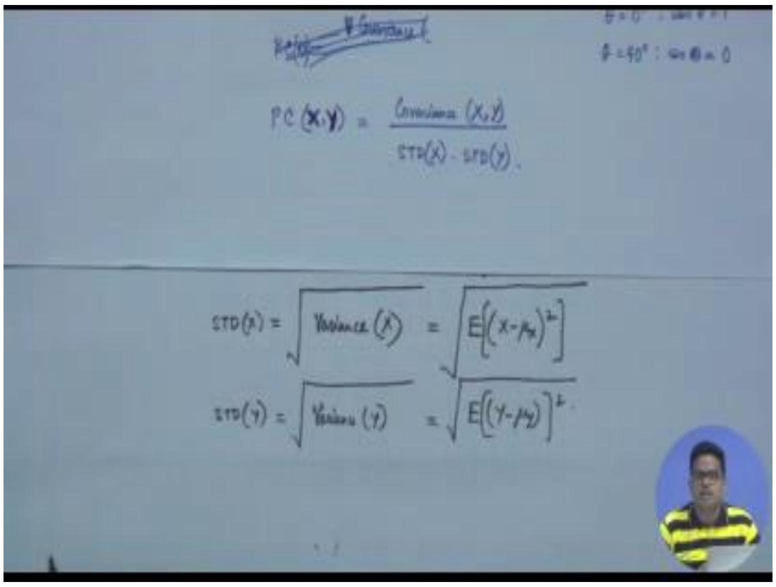
So, all we are trying to find out, is the cosine of this angle, between the two vectors a and b; and as you all know that, a dot product with b, which is the inner product, in the generic case, is equal to this mod of a, which is actually the absolute value of a, or the magnitude, alright; then, or rather, I would write, the norm of a, norm of b, times cosine of the angle between a and b. And this cosine theta that is that is the cosine similarity index. So, it is equal to a, inner product with b, divided by norm of a, norm of b. So this is the formula for calculating. And as you know that for theta equal to 0 degree, cos theta is equal to 1.

So, the cosine similarity is maximum, if the 2 vectors are pointing in the same direction; and if theta is equal to 90 degree that is, the 2 vectors are perpendicular to each other, in that case, cos of theta will be equal to 0. So, if the 2 vectors are orthogonal to each other then, there is no cosine similarity; there is no similarity of these 2 vectors.

So this is how you can this is a metric that can be used for feature selection. Another metric is called the Pearson correlation. In other words, you could also call it; it is also sometimes called the correlation coefficient. So this Pearson correlation, I will write it as PC, it is equal to, it is defined as the covariance of. And Pearson correlation, you calculate, you define this quantity in terms of, like, you calculate the Pearson correlation of 2 features. Say, you have n features and you want to find which of the features is linearly dependant on each other.

So, if you have, say, 10 features and the 10th feature can be calculated as a simple linear combination of the first 9 features; then, the tenth feature is kind of useless. So, it is good to get rid of the 10, of the tenth feature. And how to calculate, how to estimate, how linearly correlated 2 features are? So, Pearson correlation is one of those metrics. So, Pearson correlation of 2 variables x and y; so I will write it more clearly; Pearson correlation of 2 features x and y is equal to covariance of. So, I should write capital letters, because, these are random variables of, 2 random variables x and y, is equal to the covariance of x and y; so, divided by the standard deviation of x, times the standard deviation of y. So, how do you calculate the covariance of x and y?

(Refer Slide Time: 13:48)


$\theta = 0 : \cos\theta = 1$

$\theta = 90^\circ : \cos\theta = 0$

$$PC(x,y) = \frac{\text{Covariance}(x,y)}{STD(x) \cdot STD(y)}$$

$$STD(x) = \sqrt{\text{Variance}(x)} = \sqrt{E\left[(x-\mu_x)^2\right]}$$

$$STD(y) = \sqrt{\text{Variance}(y)} = \sqrt{E\left[(y-\mu_y)^2\right]}$$

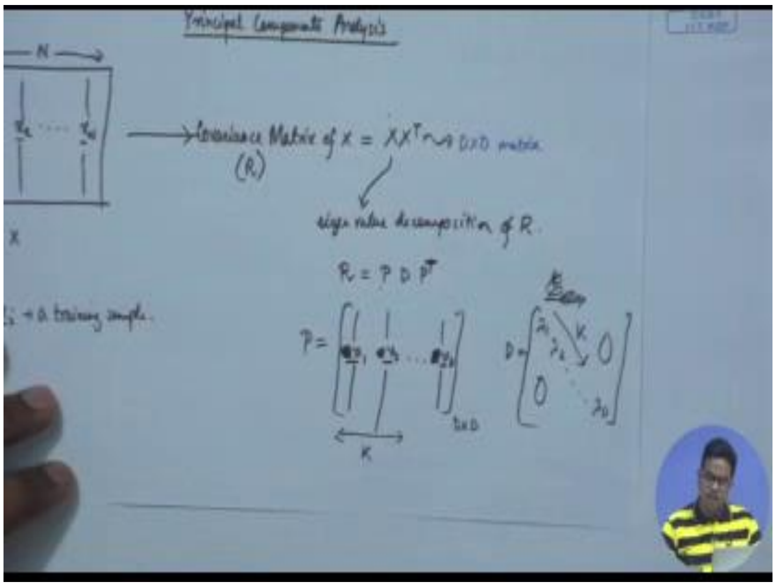
So, covariance of x and y, it is equal to the expected value that is, the mean value of x minus mu of x, times y minus mu of y, where mu of x is equal to the expected value of x and mu of y is equal to expected value of y. So, you just calculate them; then, you can subscribe them to this. So this (Refer Time: 14:27) is called mean normalization. So, you first mean normalize the random variables and then you take all the, like, joint instances of those, of both of those random variables and then, multiply them together and then, you take the expectation.

This is the, the way you take the expectation of 2, of a particular random variable. So, you first mean normalize the, the x, the y and then, you multiply those 2 mean normalized variables together to make another random variable and you take the expectation value, expectation of that. So that calculates the covariance of x and y. And standard deviation of x, as you all know, it is square root of the variance of x. So this is equal to expected value of x, minus mu of x, squared. So this is the variance and you take the square root of it. STD of y will be equal to square root of variance of y, which is equal to expected value of y minus mu of y.

So, the higher the Pearson correlation coefficient between 2 random variables, the more correlated these random variables are. And each feature is a random variable. So, each input feature is a random variable and if you; So, for given a set of input features, so, what you are going to do is, you are going to find pair-wise Pearson correlations between the random variables. And if the ones you find are highly correlated, you can just choose one of them; you can drop one of them. So, the higher the Pearson correlation, the higher is the amount of linear dependence. So, retaining such highly correlated variables would be redundant. So, you drop one of them.

So this is a feature selection method. And in the exam, you are going to face questions like this, in which you will be asked to calculate the Pearson correlation of random variables, given 2 input features and you have to calculate, find out the Pearson correlation and make a comment, whether they are highly linearly dependent, or not. So, if the value is close to 1, then they are highly linearly dependent and so, it is good to, it is like, wise to drop one of those random variables and just use one of those features. The next topic that we are going to take up is a method of feature reduction.

(Refer Slide Time: 17:13)


Principal Components Analysis
N
X
Covariance Matrix of X = XX^T ~ a D×D matrix
(R)
eigen value decomposition of R
R = P D P^T
P =
D×D
K
D =
λ1
λ2
0
λD
a training sample

Feature selection, definitely (Refer Time: 17:19) feature reduction, but this, they can be another kind of feature reduction, which is like, after finding, you know, after like squeezing the information within the given input features, which are stored within the given input features, into a smaller number of input features.

So, you are transforming the input feature space into another vector space, in which the same information is retained, but if you drop some dimensions, the amount of information loss is minimized. So this is one of this is like this falls within the category of feature reduction and principle components analysis is one of those feature reduction techniques. So, principle components analysis and this is one of the most widely used feature reduction techniques in the machine learning community.

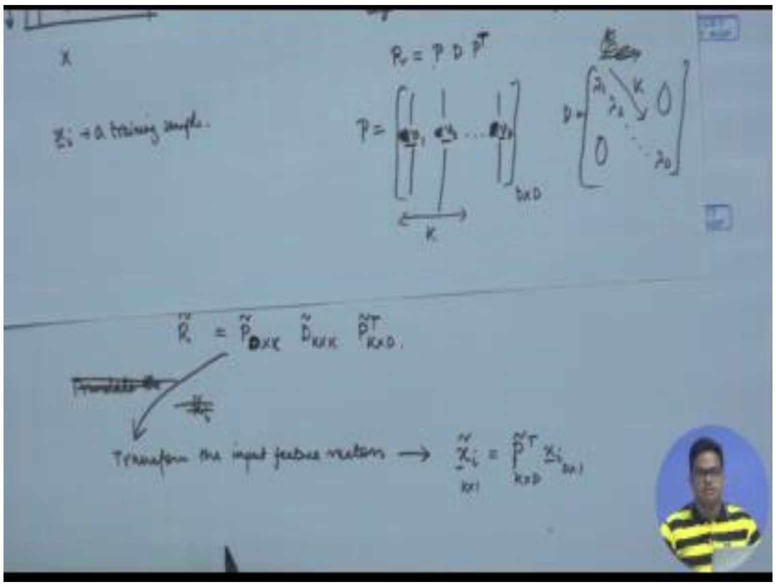
So, what happens in principle of component analysis? So, you are presented with an input matrix capital X, in which, each column represents an input feature vector, x 1, x 2, dot dot dot, till x n, say. So, each x i is a training sample. So, you arrange them as columns of a matrix capital X. Now, what you do in, while calculating the principle components analysis is, while doing this procedure is, you go about finding the covariance matrix of X, alright, which is equal to X into X transpose. So, as you have been taught in the class that, X into X transpose will calculate the column, the covariance matrix and you can figure it out why; just write out the expression of this matrix multiplication and you will figure out.

So, once this has been done, say, you have this, the input dimensionality is, say, capital D and the total number of samples is capital N, right. So, X into X transpose is going to be, a D cross D matrix, right. Now, what you do is, you take this matrix and do eigen value decomposition and this covariance matrix is usually called R. So, you make the, take the eigen value decomposition of R. So, what you do is, you represent R as a product of 3 matrices; and R is symmetric matrix. So, the decomposition, eigen value decomposition will produce, should have been written as PDP inverse, like and for a symmetric matrix this particular matrix that P will be.

So, I will come to it, what it is. So, it is going to be an orthogonal matrix, for which, the inverse is equal to the transpose. And now, let me tell about what these matrices will be. So, the matrix P will be another D cross D matrix, each column of which will be an eigen vector of x. So, eigen vector 1, eigen vector 2; this way, till eigen vector d; and D this matrix will be, let me drop the es in front; so, v 1, v 2, till v D. So, the eigen vectors and this matrix capital D will have the corresponding eigen values. So, I will write them as lambdas along the diagonal, in the corresponding positions. So this is a diagonal matrix, with the eigen values and in the hands-on tutorial class of this course, I had given a nice presentation and demonstration of, animation showing this procedure. So, you may refer to that lecture, if you want a better description of this procedure.

Now, if we can arrange these numbers, lambda 1 through lambda D in the decreasing order of their magnitudes and their corresponding locations of their eigen vectors are also properly shuffled, so that, the first K eigen vectors correspond to the top K eigen values. So, I should write this way; top K eigen values, according to the magnitude of the eigen values; then, what we can do is, we can have an approximation of R, I will write it as R tilde, which will be P tilde, I would write.

(Refer Slide Time: 22:42)



$\underline{x}_i \to$ a training sample.
$R = P D P^T$
$P = \begin{bmatrix} | & | & & | \\ v_1 & v_2 & \cdots & v_D \\ | & | & & | \end{bmatrix}_{D \times D}$
K
$D = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & 0 \\ 0 & & \ddots & \\ & & & \lambda_D \end{bmatrix}$
$\tilde{R} = \tilde{P}_{D \times K} \tilde{D}_{K \times K} \tilde{P}^T_{K \times D}$.
Transform the input feature vectors $\to \underset{K \times 1}{\tilde{\underline{x}}_i} = \underset{K \times D}{\tilde{P}^T} \underline{x}_{i\,D \times 1}$

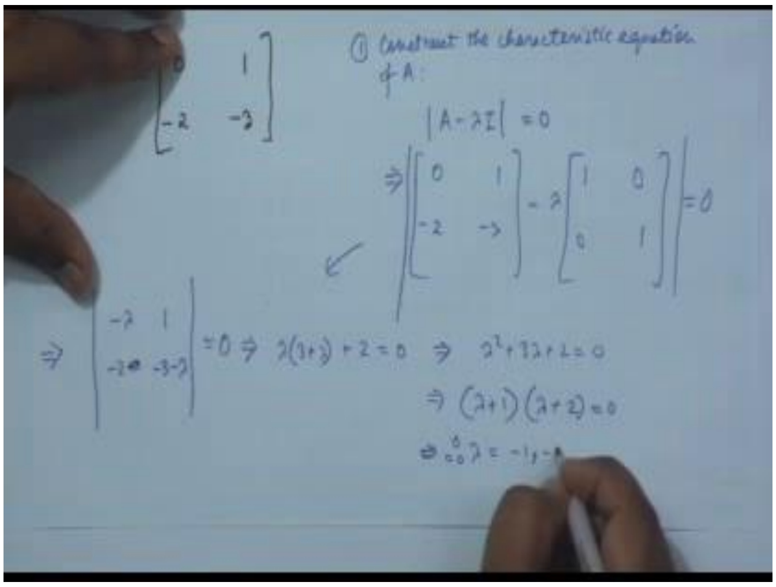
So this size will be now, D cross K; then again, D tilde, which is a K cross K matrix; and then, P tilde transpose, which is a K cross D matrix. So, what we did, we just shows the eigen vectors corresponding to the top K eigen values, according to their magnitudes and then, we approximated R as a product of these truncated matrices.

Now, once this has been done. So, the number, the higher the number of, higher the value of capital K chosen, the lesser is the error incurred in this approximation procedure. And it has been shown; it can be proven that, if you follow this procedure in reducing the dimensionality. So, say, you have a limitation of 10 features for describing your input. So this number of features is fixed. Now, if you try to find out different ways of combining the inputs, input features and squeezing the information of the input features, into 10 feature vectors. So then, the most, the method that preserves the maximum amount of information, which causes the least amount of information loss, in other words, is this method of principle components analysis. So this is the best method of doing the dimensionality reduction, into a fixed number of features.

Now, once this has been done, what we can do is, we can translate the feature vectors of the inputs, say, I will write. So, now, what you can do is, translate the, or transform, transform. After this, transform the input feature vectors. So, how do you do it? You transform vectors as P tilde transpose x i. So, P tilde transpose is going to be K cross D matrix alright. So this is D cross 1. And so, what you end up is K cross 1 dimensionality. So, once this has been done, so, what you have is, you have the transformations of the input vectors in a reduced dimensional space.

So, these are K dimensional vectors and you have a lesser set of features to describe the input with. So this is how feature reduction goes, using principle components analysis. And in the exam, you will be asked to solve problems regarding this. So, you will be given all the data; you have to find out the principle components and do the feature reduction. And you may also be asked to calculate the eigen values of a given matrix. So, how do you calculate the eigen values of a given matrix?

(Refer Slide Time: 26:20)



$$\begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}$$

① Construct the characteristic equation of A:

$$|A - \lambda I| = 0$$

$$\Rightarrow \left| \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right| = 0$$

$$\Rightarrow \begin{vmatrix} -\lambda & 1 \\ -2 & -3-\lambda \end{vmatrix} = 0 \Rightarrow \lambda(3+\lambda) + 2 = 0 \Rightarrow \lambda^2 + 3\lambda + 2 = 0$$

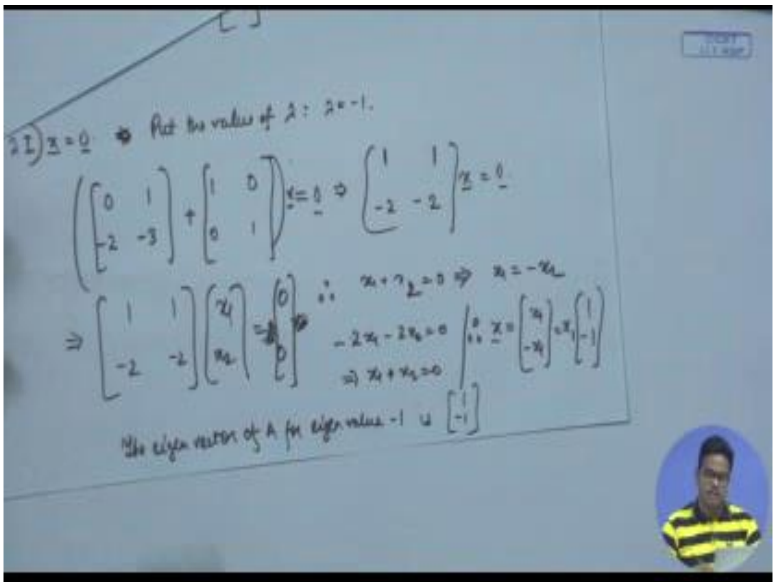
$$\Rightarrow (\lambda+1)(\lambda+2) = 0$$

$$\Rightarrow \therefore \lambda = -1, -$$

The eigen values of a matrix, say, you have been given a matrix A, which is 0, 1, minus 2, minus 3. So, calculate the eigen values of this matrix. So, how am I going to, going to solve this problem? First, step one is, construct the characteristic equation of the matrix A. So, what is it going to look like? It is A minus lambda I, determinant of that is equal to 0. So this is called the characteristic equation of a matrix, alright, of a square matrix.

Now, what it is going. So this works down to, like, 0, 1, minus 2, minus 3, minus lambda times. So, lambda is an eigen value; or yes, 0, 1; we are going to solve this; I will just shift here. So this is going to result in this matrix; minus lambda, 1, minus 2, minus 2; and then, minus 3, minus lambda, determinant is equal to 0; which gives, yes. So, it will be lambda times 3 plus lambda, minus, no, plus 2 equal to 0, which will produce, lambda square, plus 3 lambda, plus 2 equal to 0; which is equal to lambda plus 1, into lambda plus 2, equal to 0; or therefore, the solutions for lambda minus 1 and minus 2. So, these are going to be your eigen values. Now, how do you calculate the eigen vectors of A? There will be 1 eigen vector, per eigen value.

(Refer Slide Time: 28:45)



$\lambda I)\underline{x} = \underline{0}$ Put the value of λ : $\lambda = -1$.

$$\left(\begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)\underline{x} = \underline{0} \Rightarrow \begin{bmatrix} 1 & 1 \\ -2 & -2 \end{bmatrix}\underline{x} = \underline{0}$$

$$\Rightarrow \begin{bmatrix} 1 & 1 \\ -2 & -2 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$\therefore x_1 + x_2 = 0 \Rightarrow x_1 = -x_2$

$-2x_1 - 2x_2 = 0$

$\Rightarrow x_1 + x_2 = 0$

$\therefore \underline{x} = \begin{bmatrix} x_1 \\ -x_1 \end{bmatrix} = x_1\begin{bmatrix} 1 \\ -1 \end{bmatrix}$

The eigen vector of A for eigen value -1 is $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$

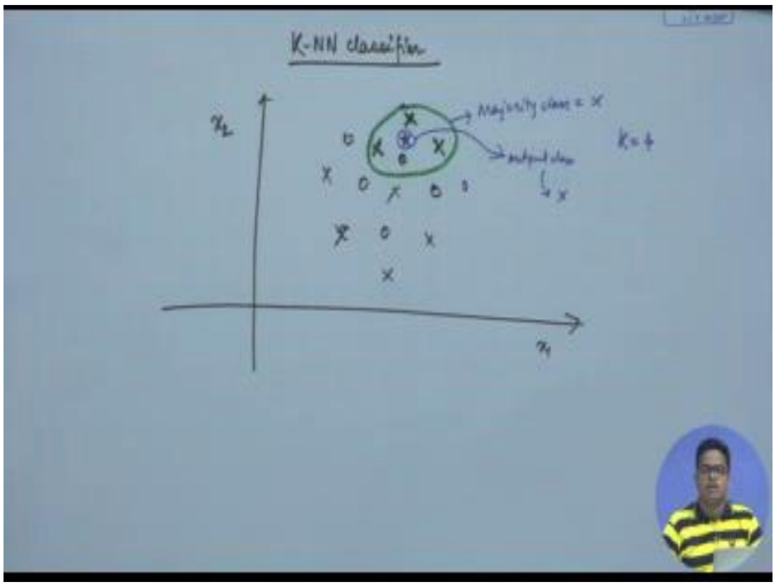
So, calculating eigen vectors; now, for calculating the eigen vector, write down the equation A x. So, say, let x be an eigen vector; x is equal to x 1, x 2, as this is a 2 plus 2 matrix, the eigen vectors will be 2 cross 1. Next, you do A x equal to lambda x. So this is how this is the condition for an eigen vector that the translinear, the transmission of the eigen vector with this matrix, is equal to a scaling, with the eigen value, corresponding eigen value.

Now this boils down to A minus lambda I, x equal to 0; and now, you put the value of lambda. So, here, lambda equal to minus 1; say, we are trying to find out the eigen vector of minus 1, the eigen value minus 1. So, what will it become? It is going to be 0, 1, minus 2, minus 3, minus, minus 1 times. So this becomes plus I, is equal to 0, which is equal to, I am sorry, times x, is equal to 0, which is equal to. Now, we can see, yes. So, which is equal to 1, 1, minus 2; this is minus 2, times x equal to 0; and this is same as writing 1, 1, minus 2, minus 2, x 1, x 2, equal to 0. So this is going to produce 2 equations, right. Therefore, x 1 plus x 2 equal to 0. So this is what you get from the first row; and these 0es are 0 vectors. So, basically, they are 0, 0, in a column.

So, I should write it as this 0, 0. This is one equation and the second equation is going to be minus 2 x 1, minus 2 x 2 equal to 0, which actually boils down to the same thing, x 1, x 2 equal to 0. Now this relation x 1 equal to minus x 2 is what will make our eigen vectors. Therefore, the eigen vector x can be written as x 1 and minus x 1, which is equal to a constant x 1 times 1 minus 1. So, as you know, eigen vectors remain invariant upon scaling. We just, we can drop this number x 1 and we can say that, an eigen value, the eigen value, eigen vector of A, for eigen value minus 1 is 1, minus 1.

And this way, we can find the eigen vector for the eigen value minus 2 as well. So, you will be asked this kind of questions, to calculate the eigen values of a matrix. Now, we go to the next topic, which is K nearest neighbor classification.

(Refer Slide Time: 32:11)



So, how we will cover it very quickly, how does it go? Say, you have this kind of a scenario, in which, you have been given 2 categories of examples. So, another cross, cross, dot; say this is your x 1; this is your x 2, 2 feature dimensions. Now, if you are asked. Now, a new sample arises, which lies, say here, denoted by a star. So, now, the question is which class this star this sample belongs to. So, what the K nearest neighbor algorithm will do is, it will find out what are the K training examples, which are nearest to the point in question; say, in our case, K is equal to 4; it is going to find out what 4 training examples are closest to the point in question. So, the ones that are selected are this cross.

So, I will highlight them in green, cross, cross, cross and dot. So, now, it is going to do a voting that, which class is majority among the neighbors of the, of the point in question. So, over here, the majority class is cross. So that is why, the output class for the point in question is cross. So this is how K nearest neighbor classification algorithm works. Now, there are some questions; for example, what happens when the K, value of K increases.

So, what happens when you choose a larger area for your context? So, when you choose a larger value of K, what happens? So, several things can happen. So, the sensitivity to noise may be reduced. For example, if you just look at the first 5, the nearest 5 samples then, it may not be sound enough. So, maybe there is, there are some noisy points which do not give any real information, but if you look at a wider perspective, or wider neighborhood, alright, with a bigger value of K then, the amount, effect of noise may reduce. So, the number of valid points within that larger neighborhood is, could be larger. So, there is a possibility of better performance, better generalization.

Another thing is, there is better probability estimation for discrete random variables. So, for discrete random variables, you would like to calculate the probability of those random variables by, through the frequency definition; that is, the number of times that particular random variable happened to occur with a particular value, divided by the total number of occurrences of that variable.

For example, the number, total number of times, total number of people. So, if you want to find out, what is the probability of a girl attending this course, may and by the frequency definition, the probability will be equal to, the total number of girls enrolled in this course, divided by the total number of students. So this is the frequency definition. And when you are using frequency definition for estimation of probability and you are using the K nearest neighbor philosophy along with it that is, you are going to look at only certain neighborhoods of the, of the point and try to estimate some probability value through the frequency definition, then a bigger neighborhood value, bigger value of k, would give a better answer.

And then, however, a bigger value of K is possible to take, only when the total number of training samples that you have at your hand is large enough, right. So, if you are always looking for, say, K equal to 100 that is, 100 nearest neighbors and you happen to have just 50 samples in your training set then, every single point will always see the same population, right. So, always, those 50 points will always be taking care of, taken into consideration. So, 100, neighborhood size of 100, or the K value of 100 is not possible to use, in this case.

So, for using a large value of K and having some good answers, you should have a large training set size; however, all of those, all of these things that I said right now, are mere observations. So, there is no theoretical, you know, these are not theoretically authoritative commentary. So, as K nearest neighbor algorithm is a non parametric learning algorithm, you always, like, you are prone to the noise that is present in the data and no matter how big a K you find, you use, you are always, it is always possible that, your estimations of statistics will not be good enough. So, there are chances of failure, because you are completely dependent on the data.

And yes that is pretty much, pretty much for today and we have discussed all the topics that were, more or less all the topics that have been covered in this particular week and we have seen how to solve problems in the exam. So, best of luck for your assignment; it will come online this Sunday and the deadline will also be announced along with it; and be sure that, the deadline will not be postponed. So, start solving the assignment as soon as you complete the lectures. Good.

See you next time. Bye-bye.