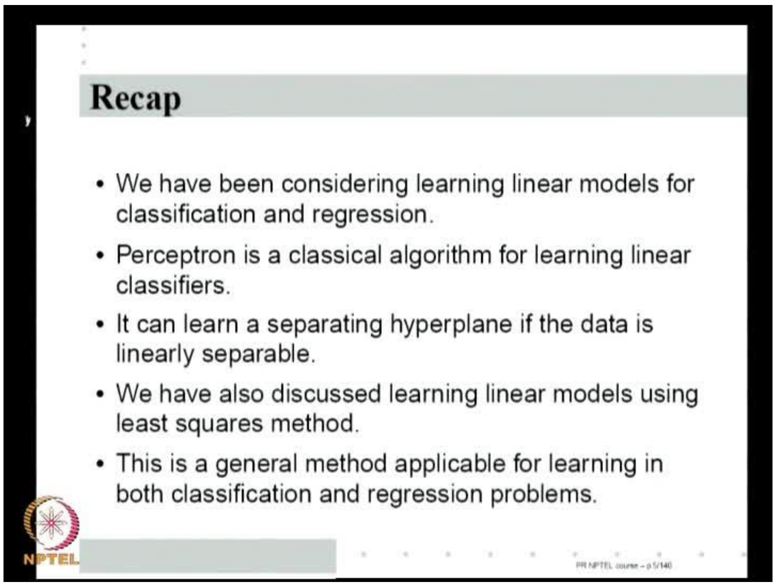
Pattern Recognition

Prof. P. S. Sastry
Department of Electronics and Communication Engineering
Indian Institute of Science, Bangalore

Lecture - 16
AdaLinE and LMS algorithm; General nonliner least-squares regression

Welcome to the next lecture on pattern recognition. To briefly recall what we have been doing in the last lecture we have we looked at learning linear models for classification and regression.

(Refer Slide Time: 00:43)



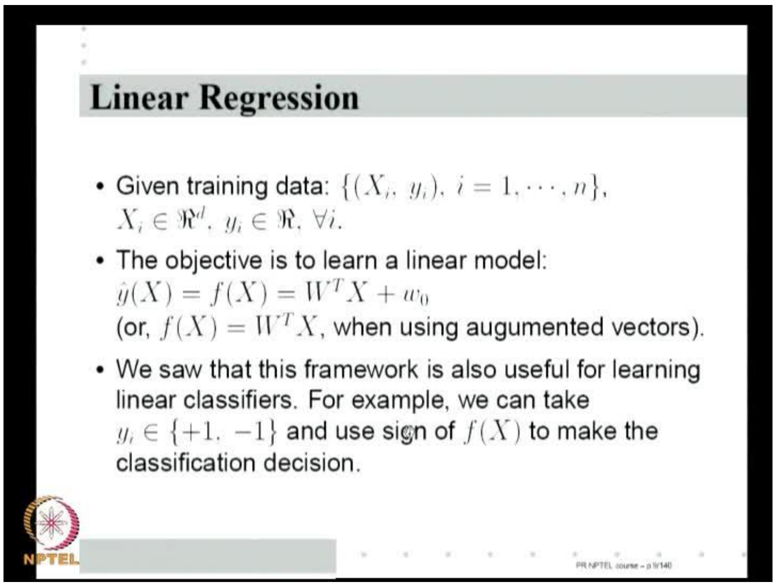
Recap

- We have been considering learning linear models for classification and regression.
- Perceptron is a classical algorithm for learning linear classifiers.
- It can learn a separating hyperplane if the data is linearly separable.
- We have also discussed learning linear models using least squares method.
- This is a general method applicable for learning in both classification and regression problems.

PR NPTEL course – p.5/146

We have been looking at linear discriminant functions as well as linear regression models; we started it a few classes ago. And the first linear classifier we considered is the perceptron algorithm, is a classical algorithm for learning linear classifiers, and we have seen all the details and as we showed it can learn a separating hyperplane if the data is linearly separable. Then for more general purposes, we have been looking at learning linear models using least squares method, linear least squares methods. Here we can learn both classifiers and regressors. So, this is what we started with last class and we will continue. This is a very general method applicable for both classification and regression problems. So, we have been treating both of them together.

(Refer Slide Time: 01:29)



Linear Regression

- Given training data: $\{(X_i, y_i), i = 1, \cdots, n\}$, $X_i \in \Re^d$, $y_i \in \Re$, $\forall i$.
- The objective is to learn a linear model: $\hat{y}(X) = f(X) = W^T X + w_0$ (or, $f(X) = W^T X$, when using augumented vectors).
- We saw that this framework is also useful for learning linear classifiers. For example, we can take $y_i \in \{+1, -1\}$ and use sign of $f(X)$ to make the classification decision.

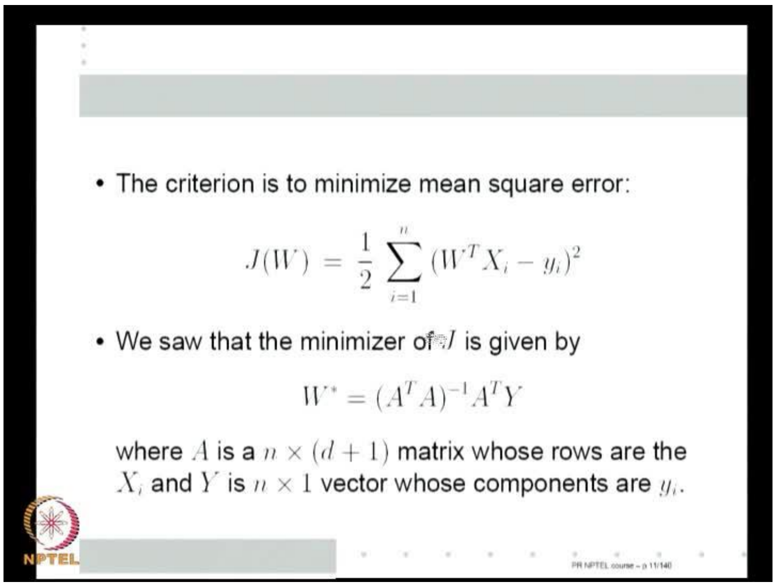
PR NPTEL course – p.9/140

So, to recall in a regression problem the given training data is X i y i where X i as usual is say d dimensional vector, y i now can be any real value real value. In a classification problem y i as that X binary values when you are considering two class problem, the binary values can be either 0, 1 or plus 1 minus 1 or it it will take finitely many values if I am considering M class problem. In a regression problem the only difference is that the targets y i can take any real value.

So, the idea is to learn a function relationship between x and y, so given this x i y i samples you want to learn y i the general function of x, so that given any nu x, we can predict y. And here we are interested in learning a linear model we are essentially interested in predicting the value of y, y hat I wrote there, which of course, is a function of x as f of x that is the function we want to learn, as w transpose x plus w naught. And this is the linear model that we have been looking at. As we have seen if we use augmented variables I can write it as w transpose x.

So, we will continue within the occasion when we want we will write the w 0 separately, otherwise we will just write w transpose X. But whenever we write w transpose X it is implicitly assumed that we are using augmented vector, so that this constant is also incorporated. Augmenting X essentially means I put one extra component in X whose value is 1 and I use w 0 at that first component in W. Essentially this is called an affine function this is quite linear because of the constant by augmenting it becomes a strictly linear function otherwise they are essentially affine functions.

(Refer Slide Time: 03:43)



- The criterion is to minimize mean square error:

$$J(W) = \frac{1}{2}\sum_{i=1}^{n}(W^T X_i - y_i)^2$$

- We saw that the minimizer of J is given by

$$W^* = (A^T A)^{-1} A^T Y$$

where A is a $n \times (d+1)$ matrix whose rows are the X_i and Y is $n \times 1$ vector whose components are y_i.

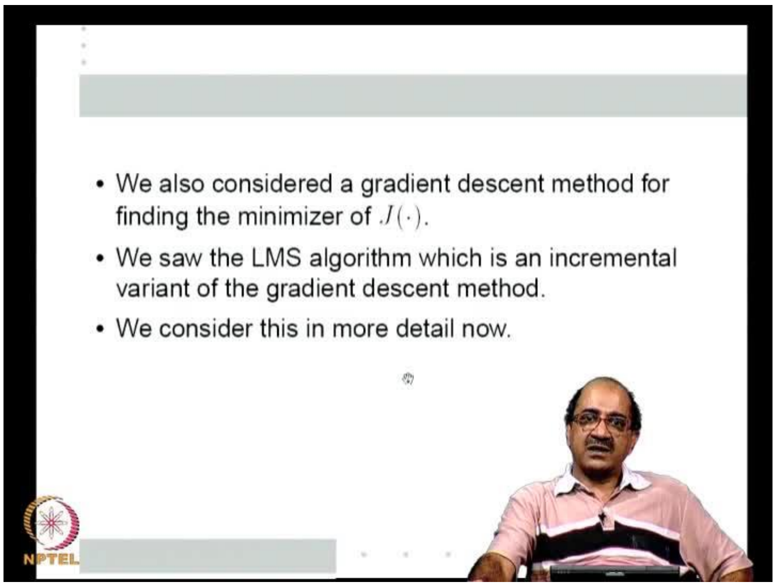
PR NPTEL course – p.11/140

So, our objective is to learn such a linear model and we have also seen that such a model can also be looked at as a classifier for example, we can use targets to be plus 1 minus 1 in a two class problem. And then if we learn affects to be close to the targets w 1, w naught are there w transpose x plus w naught is close to the targets, then we can simply use sign of f X as the classifier. We have been looking at the following criterion for our least squares method, the criterion is to minimize the mean square error. Essentially on an example X i, if I am currently using W, then W transpose X i is my prediction, y i is the target.

So, W transpose X i minus y i whole square is the error who we sum it up. So, that is why it is called a mean square error, essentially to make it strictly mean I should divide by n here. But really a because we are minimizing this constant here makes no difference and as we shall also see once again later on. Hence, we just take it to be half summation of the squares of the errors.

This is the criterion we are trying to minimise and we saw last class that the minimisers of this mean square criterion is given by W star is A transpose A whole inverse A transpose Y, where A is the n plus n by d plus 1 matrix whose rows are X i, X i's are the training data training vectors. There are n of them that is because I stack up X's at the rows of A, A will be A will have n rows and because each X i is an augmented vector there will be d plus 1 columns and capital Y is the n by 1 vector for all the targets namely y i.

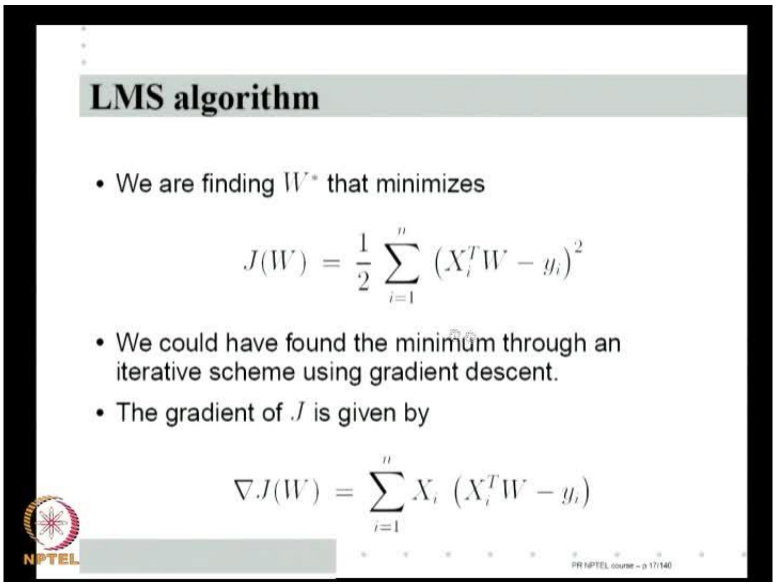
(Refer Slide Time: 05:13)



- We also considered a gradient descent method for finding the minimizer of $J(\cdot)$.
- We saw the LMS algorithm which is an incremental variant of the gradient descent method.
- We consider this in more detail now.

We have also seen that A transpose n inverse A transpose is called the generalised inverse of A, we have seen these relations with solving assets for linear equations. We also considered last class a gradient base method for minimising J and with that was the LMS algorithm, which is a incremental version of the gradient descent method. We that is what we ended the last class with we very hurriedly went through LMS algorithm. So, in this class we consider it in more detail, we will start with LMS and then discuss some more about the linear least squares method.

(Refer Slide Time: 05:41)



LMS algorithm

- We are finding W^* that minimizes

$$J(W) = \frac{1}{2}\sum_{i=1}^{n}\left(X_i^T W - y_i\right)^2$$

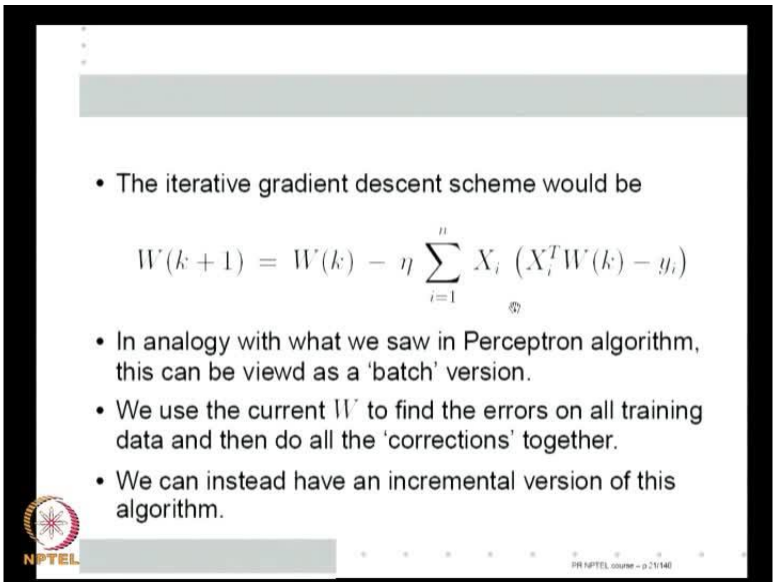
- We could have found the minimum through an iterative scheme using gradient descent.
- The gradient of J is given by

$$\nabla J(W) = \sum_{i=1}^{n} X_i \left(X_i^T W - y_i\right)$$

PR NPTEL course – p.17/140

So, essentially as I have said we our objective is to find a W star that minimizes this function of W, thus half i equal to 1 to n, X i transpose W minus y i whole square this is sum of squares of errors. And we have of course, seen you know straight linear algebra we are solving it, but we could have also used a gradient descent to find the minimum.

(Refer Slide Time: 06:34)



- The iterative gradient descent scheme would be

$$W(k+1) = W(k) - \eta\sum_{i=1}^{n} X_i \left(X_i^T W(k) - y_i\right)$$

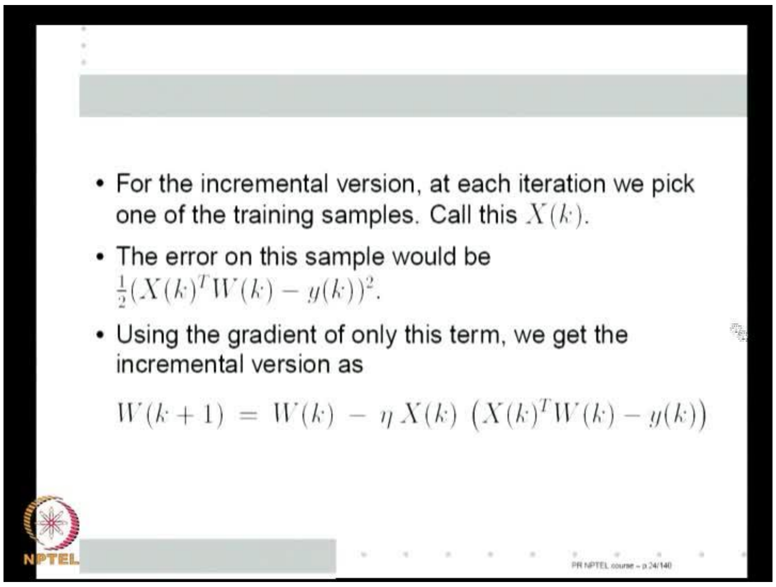
- In analogy with what we saw in Perceptron algorithm, this can be viewd as a 'batch' version.
- We use the current W to find the errors on all training data and then do all the 'corrections' together.
- We can instead have an incremental version of this algorithm.

PR NPTEL course – p.21/140

We can minimise any function by simply using an iterative gradient descent algorithm, to obtain the gradient descent we need the gradient of this function. So, differentially this function the only reason as I said we get this half here is that when you differentiate that two will cancel this half. So, the gradient becomes as equal to 1 to n, X i into X i transpose W minus y i. So, given this is the gradient the gradient descent could now be W k plus 1 is W k minus eta time's gradient that is the gradient.

Now, we have already seen a gradient descent version of perceptron algorithm. So, in analogy with that we can call this a batch mode algorithm. Here we are have 1 W k with which we are predicting the target on each of the examples, finding all the errors and then summing up all the corrections and doing them together. So, essentially this is the batch version because we are using the current W to find the errors on all the training data and then do all the corrections together.

(Refer Slide Time: 07:54)



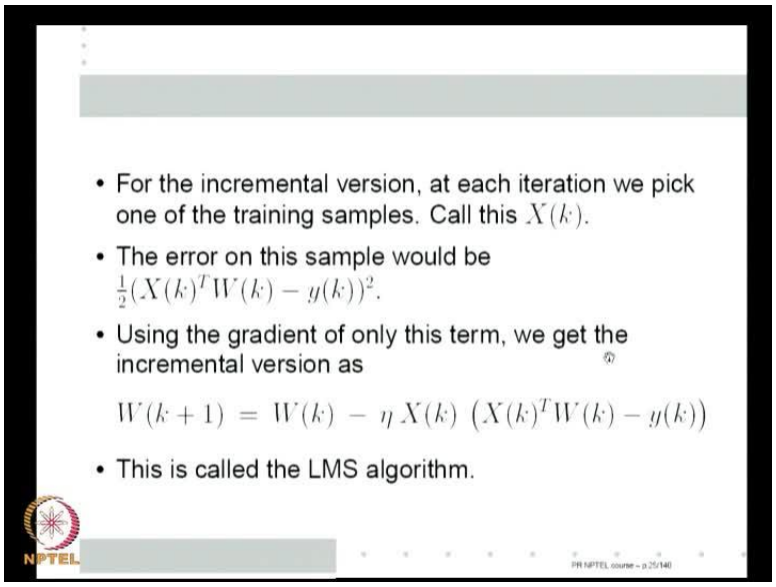
- For the incremental version, at each iteration we pick one of the training samples. Call this $X(k)$.
- The error on this sample would be $\frac{1}{2}(X(k)^T W(k) - y(k))^2$.
- Using the gradient of only this term, we get the incremental version as

$$W(k+1) = W(k) - \eta X(k) \left(X(k)^T W(k) - y(k)\right)$$

PR NPTEL course – p.24/140

And just like in perceptron case we can consider a incremental variant of this gradient descent. So, what will be such an incremental of variant be so instead of with the current W predicting targets on all the samples and then during the correction like this. At each instant we pick up a random example X k, Y k and just do correction for that. So, in the incremental version there will be one term here. So, for the incremental version let us say we pick the training sample that is picked at k let us call that X k.

(Refer Slide Time: 08:40)



- For the incremental version, at each iteration we pick one of the training samples. Call this $X(k)$.
- The error on this sample would be $\frac{1}{2}(X(k)^T W(k) - y(k))^2$.
- Using the gradient of only this term, we get the incremental version as

$$W(k+1) = W(k) - \eta\, X(k)\, \left(X(k)^T W(k) - y(k)\right)$$

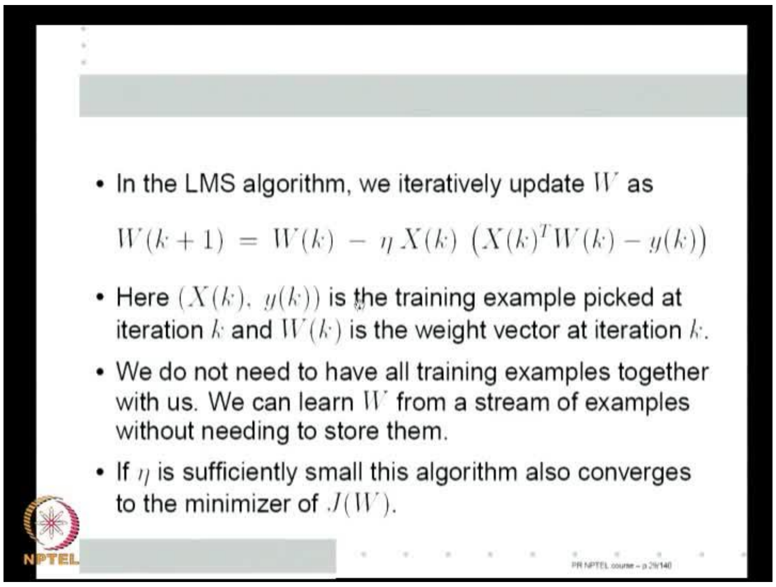
- This is called the LMS algorithm.

PR NPTEL course – p.25/140

So, at the each iteration we pick one training sample, then on that training sample the error would be X k transpose W k minus y k whole square of course, half will be there. So, in the incremental version we use only gradient of this term to correct that gives us the following incremental version of the gradient descent algorithm, W k plus 1 is W k minus eta X k into X k transpose W k minus y k.

So, this is the error and multiplied by X that is the update we do. So, in contrast we have this entire summation of the batch the batch version, here we will look at only one at a time, so even that we are looking at k th iteration is given by this given by X k. So, the example of X k, y k at k th iteration, so this is how I update W k in the gradient version. This is called the LMS algorithm, least mean square algorithm.

(Refer Slide Time: 09:00)



- In the LMS algorithm, we iteratively update W as

$$W(k+1) = W(k) - \eta X(k) \left(X(k)^T W(k) - y(k)\right)$$

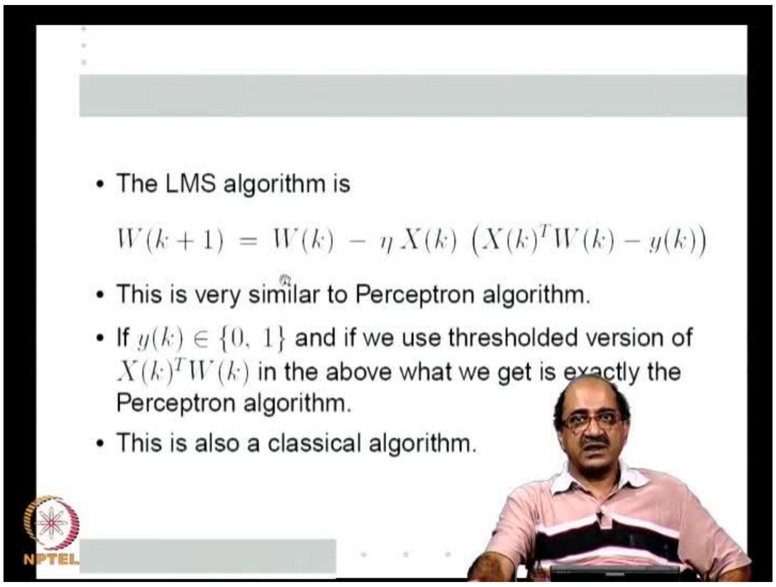
- Here $(X(k), y(k))$ is the training example picked at iteration k and $W(k)$ is the weight vector at iteration k.
- We do not need to have all training examples together with us. We can learn W from a stream of examples without needing to store them.
- If η is sufficiently small this algorithm also converges to the minimizer of $J(W)$.

PR NPTEL course – p.29/140

So, the LMS algorithm is given by this as we just now saw, W k plus 1 is W k minus eta times X k into X k transpose W k minus y k, where X k, y k is the training example picked at iteration k and W k is the weight vector at iteration k. So, like in the case of perceptron the basic idea about the incremental version is that we do not have to have all the training examples together with us. We can actually learn W from a stream of example so to say if I have got X k, y k coming at the stream without storing all of them we can simply keep updating W k like this.

So, that is essentially what characterise incremental version, one can show that if eta is sufficiently small, then this algorithm also converges to the minimise of J W, which is same as the earlier solution we got. So, instead of doing calculating that matrix A and inverting a transpose A and all that we can simply run an iterative algorithm like this and that will also converts if the step size eta is sufficiently small.

(Refer Slide Time: 10:16)



- The LMS algorithm is

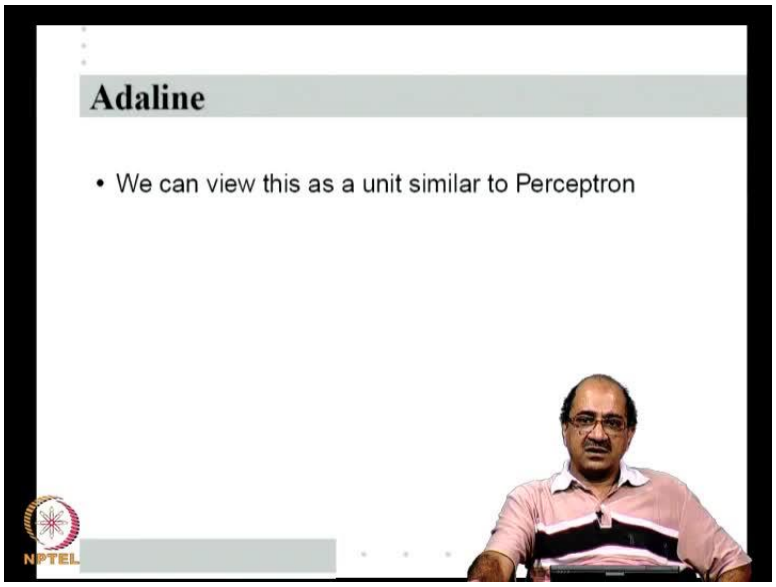
$$W(k+1) = W(k) - \eta X(k) \left(X(k)^T W(k) - y(k)\right)$$

- This is very similar to Perceptron algorithm.
- If $y(k) \in \{0, 1\}$ and if we use thresholded version of $X(k)^T W(k)$ in the above what we get is exactly the Perceptron algorithm.
- This is also a classical algorithm.

Once again this is the LMS algorithm, this is very similar to perceptron algorithm. Now, we can actually see the similarities like this let us say y k belongs to 0 and 1, like in the perceptron algorithm. So, here because we are using a linear model the predictor is X k transpose y k. If I am using a perceptron the predictor will be thresholded version of X k transpose y k at 0. So, suppose this algorithm we shall using X k transpose y k at the predictor we use the threshold.

Then what does that mean, if what I predict is same as y k of course, there is no correction like in the perceptron algorithm. Suppose I should have predicted one whereas, I predicted 0, so this is 0 this is 1 then it becomes the term in brackets becomes minus 1. So, which essentially means I am adding X k to W k, W k plus eta times X k is just a step size. Similarly, I should have predicted 0, but I have predicted 1, then I would have subtracted X k from W k. So, in that sense this exactly similar to the perceptron algorithm. So, like perceptron LMS algorithm is also a fairly classical algorithm more than 50 years old now, it is been used extensively in adaptive filtering and many other signal crossing and pattern recognition applications.

(Refer Slide Time: 11:40)

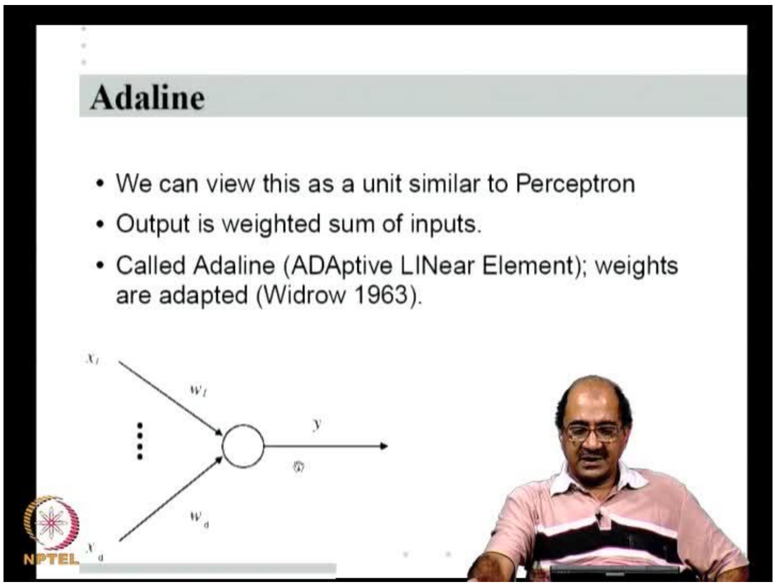


Adaline

- We can view this as a unit similar to Perceptron

Like perceptron we can view this also as a unit is often called adaline, it is also originally thought of as modelling that can referred up to units.

(Refer Slide Time: 11:53)

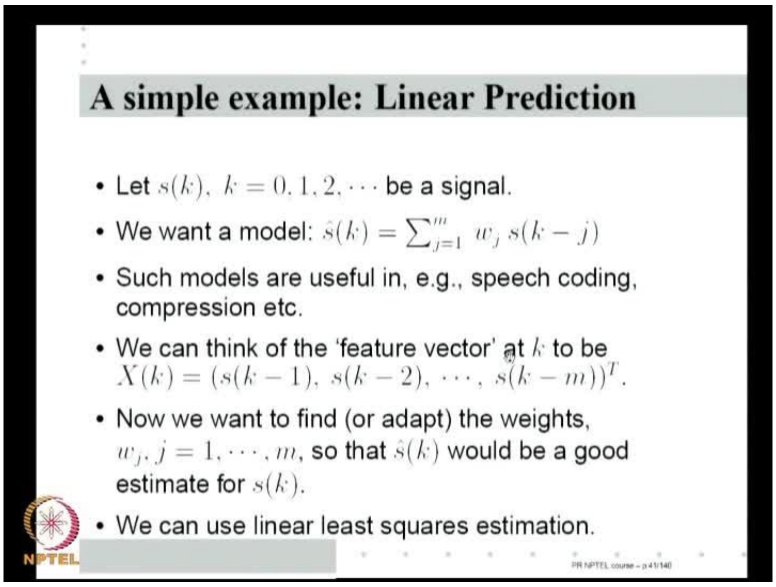


Adaline

- We can view this as a unit similar to Perceptron
- Output is weighted sum of inputs.
- Called Adaline (ADAptive LINear Element); weights are adapted (Widrow 1963).

So, essentially we have the same unit except that now the output is simply the weighted sum of the inputs, there is no thresholding anymore of this, such a unit was called adaline. Adaline standing for adaptive linear element, adaptive in the sense weights are adapted and this was first proposed by Widrow in 1963. Once again those days it was it was realised using op amps instead of having a algorithm in the system on a computer system. And it is it has been one of the one of the main algorithms for much of adapter signal crossing and many other linear model predictions.

(Refer Slide Time: 12:39)



A simple example: Linear Prediction

- Let $s(k)$, $k = 0, 1, 2, \cdots$ be a signal.
- We want a model: $\hat{s}(k) = \sum_{j=1}^{m} w_j\, s(k-j)$
- Such models are useful in, e.g., speech coding, compression etc.
- We can think of the 'feature vector' at k to be $X(k) = (s(k-1),\ s(k-2),\ \cdots,\ s(k-m))^T$.
- Now we want to find (or adapt) the weights, w_j, $j = 1, \cdots, m$, so that $\hat{s}(k)$ would be a good estimate for $s(k)$.
- We can use linear least squares estimation.

PR NPTEL course – p.41/140

So, before we go to more detail discussion of various extensions of linearly squares and its relationship with other algorithm. Let us just quickly run through one simple example, which is also a problem on which linear least squares very often used it is called linear prediction. Let us say we have a signal s k, we are assumed a discrete time signal, because we are working in discrete time here, all these algorithms are for discrete time case. So, let us say we have a signal s k, k is equal to 0, 1 so on and what we want is a model that s hat k by our notation of when we put hat is a estimate of s k.

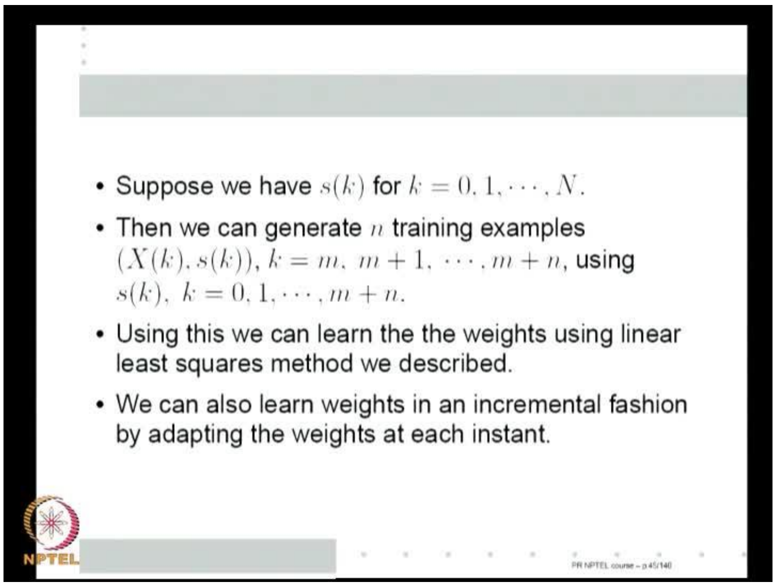
So, I want to predict the signal value at k based on the signal value at the previous m time distance, j goes from 1 to m. So, I am using s of k minus 1 k minus 2 all the way up to k minus m. So, using the m past values of the signal I want to predict the next value of the signal. So, we want a model that write s hat k as summation j is equal to 1 to m, w j s k minus j, here we are looking for a prediction model that is a linear w are the w j are the weights in the linear model that is why it is called a linear prediction.

Such models are very useful in many applications such as speech codings, speech compression many other signal compressions and so on. The idea is that if the signal is varying slowly over tens of signal I can learn to predict the signal using the past values. So, I can use some of the past values to learn the w j, once I learn the w j the rest of this signal I can predict by keep taking the previous values. So, I do not have to store the entire signal I can store the first few of values of the signal and the w weights.

So, that next few values can now be predicted on the fly that is how it is used in coding or compression. I will come back to this after running through the example. So, in our case say a linear model is w transpose X. So, we can think of the feature vector at k to be s k minus 1, s k minus 2, s k minus m. all vectors are column vectors, so that is why there is a transpose there. So, then if I take w components to be w 1, w m this is nothing but w transpose X. So, essentially I want a predictor that is w transpose X.

So, the idea is that we want to adapt the weights w j or length the weights of w j. So, that s hat k would be a good estimate for s k. Now, my target is s k. So, it is a very interesting thing what is happening I am getting s 0, s 1, s 2, s so on, so at any given time I take say s k minus 1, s k minus 2, s k minus m as my feature vector. And using a w i want to predict and the target for me will be s k. So, we can use the linear least squares algorithm that we have been talking about so far.

(Refer Slide Time: 15:59)



- Suppose we have $s(k)$ for $k = 0, 1, \cdots, N$.
- Then we can generate n training examples $(X(k), s(k))$, $k = m,\ m+1,\ \cdots, m+n$, using $s(k)$, $k = 0, 1, \cdots, m+n$.
- Using this we can learn the the weights using linear least squares method we described.
- We can also learn weights in an incremental fashion by adapting the weights at each instant.

PR NPTEL course – p.45/140

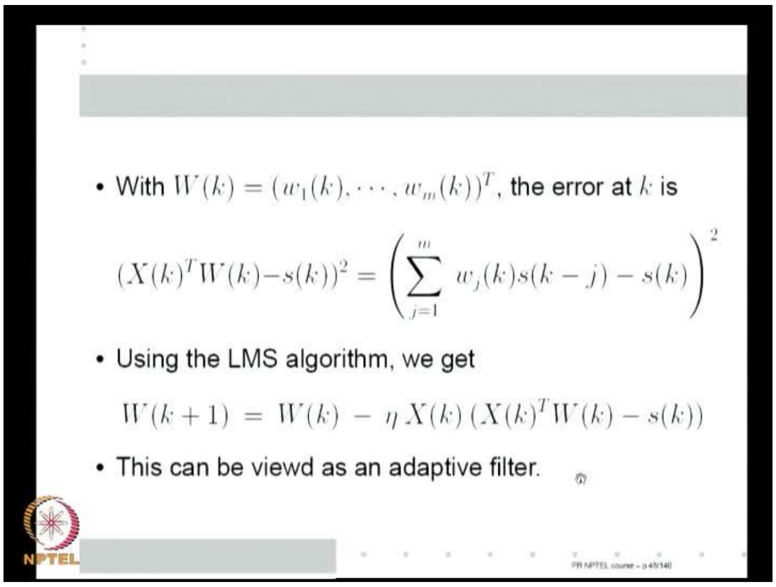
So, for example, let us say I have the signal for up to some capital N instance. Then we can generate say a small n training examples X k, s k, k going from m, m plus 1 m plus n why m, because to get X of m I need s m minus 1, s m minus 2 all the way as s m minus m. So, using from s 0 to s m minus 1 I can get X m by our notation. So, that is the feature vector and the target predictor for it will be s f and so on. So, X k, s k, k m plus 1 m, m plus 1 up to m plus n, to use this I will consume the signal from 0 to m plus n, consume the signal from 0 to m plus n I can generate such samples.

So, the basic idea is the following let us let us take P's signal of course, across the entire time duration I may not be able to predict this P signal as a linear combination of its past values, but over short periods of time it might be feasible. So, let us say for concreteness I am sampling at 10 sampling frequency, so that in one second I get 10 thousand samples. So, let us say in 1 in 100 milliseconds I get about 100 samples, 100 milliseconds might be a small enough chunk of time over which I can model the signal by this kind of a A R model, it is called A R model auto regressive model, which is which is essentially a linear combination of its past values.

So, let us say I want to represent a linear combination of past 10 values. So, I may use up to the first 100 values of the signal, if I use the first 100 values of signal, then I may get about 90 training samples, to learn the 10 unknown w's. Once I learn them from the sample 100 up to sample 1000. Now, I can predict using the length model. Now, I can do the same thing for the next chunk of 1000 samples. So, what happened is each chunk or as it is called a frame or the speech instead of storing all the 1000 samples I may need only the only the first 10 samples and then the 10 coefficients.

Then I can predict the remaining ones may be there will be some error, but may be useful in many signal compression and coding applications. So, all such coding for example, this kind of coding is called LPC, linear predictive code. So, once I got such training examples how do I learn of course, I can learn weights using our linear least squares, I can line up all these X k's into matrix A and then find the find the W optimal W or we can also learn it using the LMS algorithm. We can adopt weights at each instant.

(Refer Slide Time: 19:05)



- With $W(k) = (w_1(k), \cdots, w_m(k))^T$, the error at k is

$$(X(k)^T W(k) - s(k))^2 = \left(\sum_{j=1}^{m} w_j(k) s(k-j) - s(k) \right)^2$$

- Using the LMS algorithm, we get

$$W(k+1) = W(k) - \eta X(k) (X(k)^T W(k) - s(k))$$

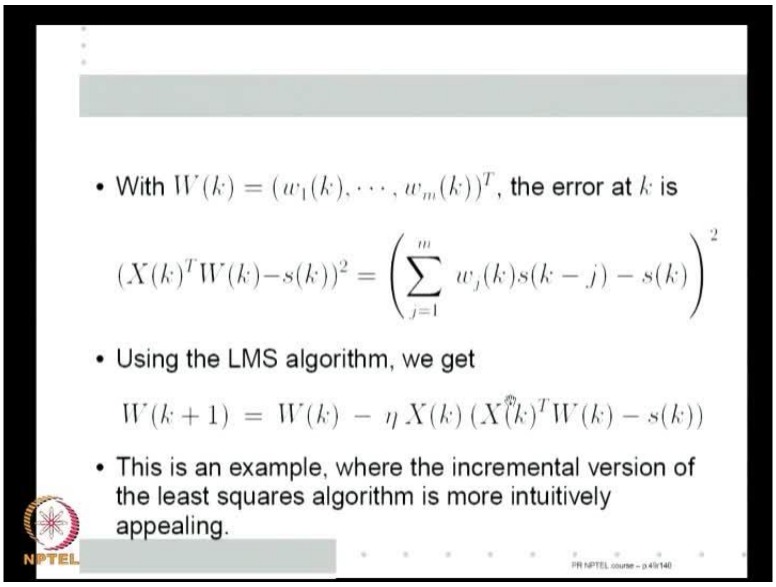
- This can be viewd as an adaptive filter.

PR NPTEL course – p.45/140

If I use the LMS algorithm, let us say weights at k th instant is given by W k that is w 1 k to w m k. Then error instant k is X k at transpose W k minus the target s k whole square, X k transpose at W k is nothing but w j k, s k minus j minus s k whole square. Now, this is simply a filter if I am getting s k on some line, I essentially have to keep putting the s and then multiply with weights and sum. So, this is a very simple F A R filter. So, essentially I am designing an F A R filter system, that is what my predictor is and this is the error.

Given this error of course, my algorithm will be W k plus 1 is W k minus eta X k, X k transpose X k into X k transpose W k that is this predictor minus s k that is the error term. So, as I said this can be thought of as a adaptive filter, essentially my predictor is an F A R filter and these w’s or the filter weights. So, at each instant I can keep modifying the filter (( )). So, this is a adaptive filter and if it has sufficiently small very quickly it converges and from then on I would be able to predict s k very accurately.

(Refer Slide Time: 20:28)



- With $W(k) = (w_1(k), \cdots, w_m(k))^T$, the error at k is

$$(X(k)^T W(k) - s(k))^2 = \left(\sum_{j=1}^{m} w_j(k) s(k-j) - s(k) \right)^2$$

- Using the LMS algorithm, we get

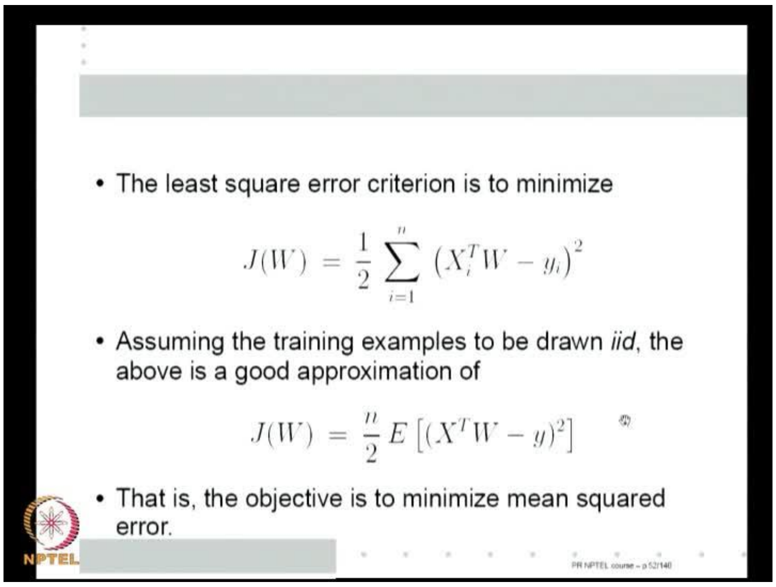
$$W(k+1) = W(k) - \eta X(k) (X(k)^T W(k) - s(k))$$

- This is an example, where the incremental version of the least squares algorithm is more intuitively appealing.

PR NPTEL course – p.49/140

So, this is for example, one situation where the incremental version of the least squares is much more intuitively appealing. Instead of taking all the signal then offline making so many training samples by stacking of X k's and then putting them in a matrix and inverting a transpose and all that, I can essentially do it online at each instant. I have a buffer to store the previous m value at the signal and I use that to calculate the current error and use that to keep changing my tapping weights in the in the filter. So, in this kind of adaptive filtering applications our incremental gradient descent version namely the LMS algorithm, is much more appealing than the batch version of variant descent or the actual matrix solution of the linear least squares. As I said with eta sufficiently small all of them give rise to the same final solution.

(Refer Slide Time: 21:32)



- The least square error criterion is to minimize

$$J(W) = \frac{1}{2}\sum_{i=1}^{n}\left(X_i^T W - y_i\right)^2$$

- Assuming the training examples to be drawn *iid*, the above is a good approximation of

$$J(W) = \frac{n}{2} E\left[(X^T W - y)^2\right]$$

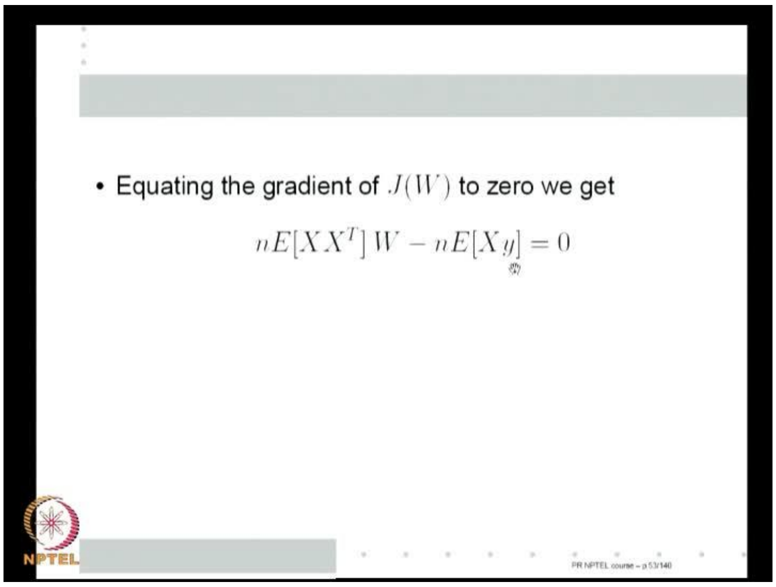
- That is, the objective is to minimize mean squared error.

PR NPTEL course – p.52/140

So, that is our example let us move on to now a few other issues about linear least squares. So, first let us consider the least squares error criterion is to minimize this error, if I assume that X i, y i are drawn IID from the underlying distributions. So, X n y have some stochastic relation between them and examples are IID. Then I can think of such a sum as essentially approximating the corresponding expectation. So, actually I can think of this as expected value of X transpose W minus y whole square, where X and y are the corresponding random variables, whose samples are X i and y i.

So, X comma y is a random vector and I got NIID samples X i, y i out of them. Of course, if it is expectation it has to be 1 by n here, we said n really does not matter, we will see currently y n does not matter, but because we did not put n there I will put n here. So, this J W is a very good approximation of n by 2 expected value of X transpose W minus y whole square. Now, I can differentiate this and find what will be the optimal W using the expectation operator. So, one thing we can see is that more clearly then in this version, this version clearly shows that the object is to minimise mean square error. Expectation operation is to minimize mean of the squared error. So, that is why this is called you know mean square error algorithms, minimize mean squared or least mean square estimates and so on.

(Refer Slide Time: 23:08)



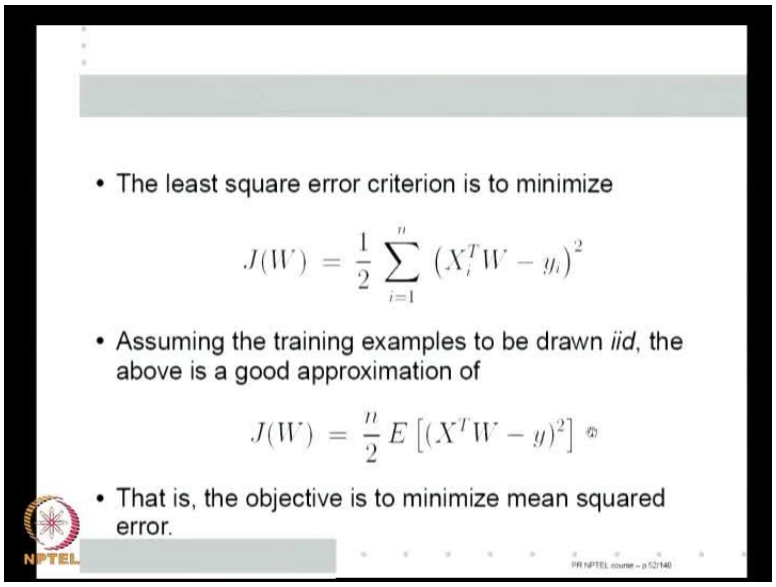
- Equating the gradient of $J(W)$ to zero we get

$$nE[XX^T]W - nE[Xy] = 0$$

PR NPTEL course – p.53/140

So, if we equate gradient of J W to 0, what is gradient of J W, this 2 will go away with this the half will go with the 2 that n will remain.

(Refer Slide Time: 23:12)



- The least square error criterion is to minimize

$$J(W) = \frac{1}{2}\sum_{i=1}^{n}\left(X_i^T W - y_i\right)^2$$

- Assuming the training examples to be drawn *iid*, the above is a good approximation of

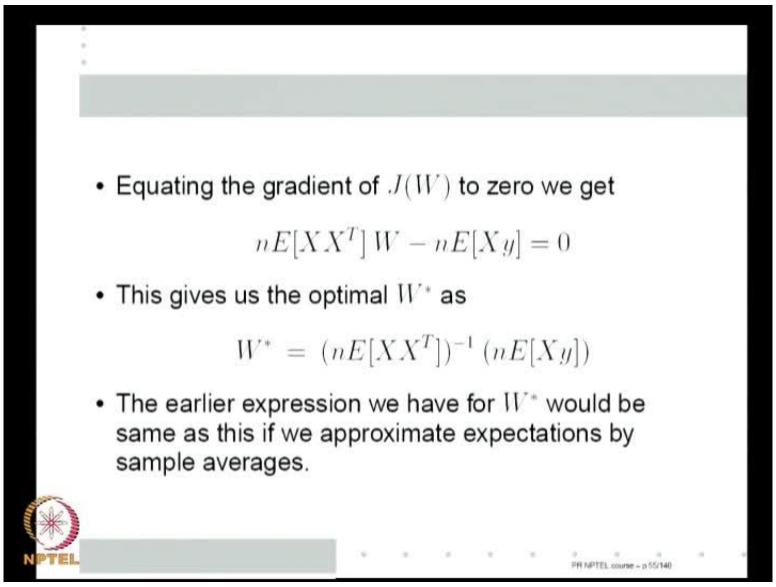
$$J(W) = \frac{n}{2}E\left[(X^T W - y)^2\right]$$

- That is, the objective is to minimize mean squared error.

PR NPTEL course – p.52/140

Then expect value of X into X transpose W minus y you equate it to 0, then I get n into expect value of X, X transpose W comes out of the expectation minus n times expected value of X y equal to 0.

(Refer Slide Time: 23:44)



- Equating the gradient of $J(W)$ to zero we get

$$nE[XX^T]\,W - nE[Xy] = 0$$

- This gives us the optimal W^* as

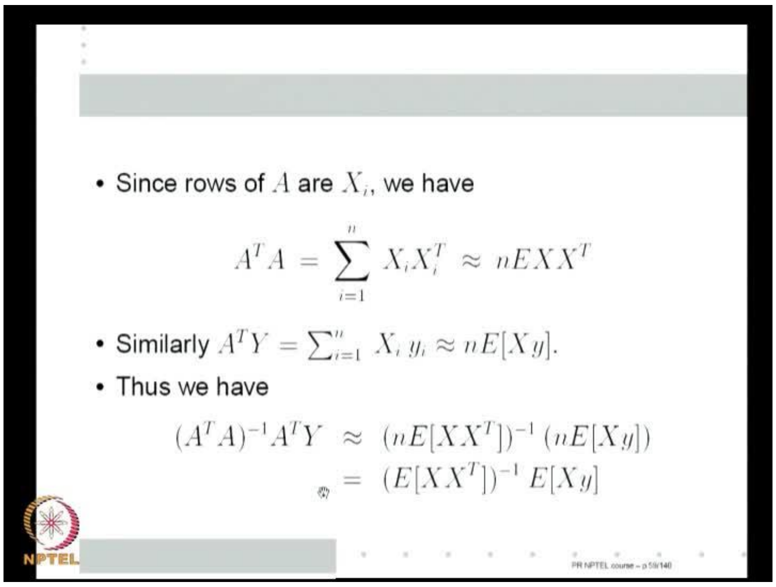
$$W^* = (nE[XX^T])^{-1}\,(nE[Xy])$$

- The earlier expression we have for W^* would be same as this if we approximate expectations by sample averages.

PR NPTEL course – p.55/140

It is expectation X, X transpose into W minus X y this of course, this two has come out when the derivative. So, this gives me w as n times expect value of X, X transpose whole inverse into n times expect value of X y. We can actually show that this the same expression we get for W star using our matrix relation, if we have once again approximate expectations by sample averages

(Refer Slide Time: 24:07)



- Since rows of A are X_i, we have

$$A^TA = \sum_{i=1}^{n} X_iX_i^T \approx nEXX^T$$

- Similarly $A^TY = \sum_{i=1}^{n} X_i\,y_i \approx nE[Xy]$.
- Thus we have

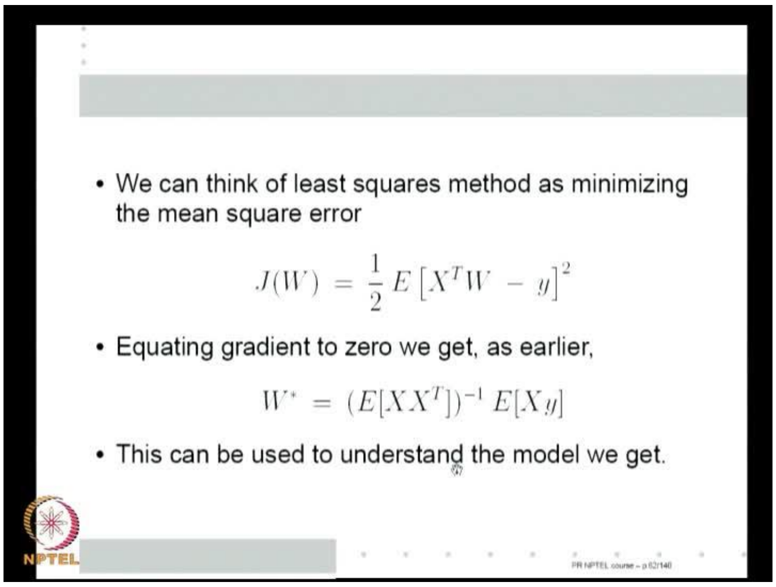
$$\begin{aligned}(A^TA)^{-1}A^TY &\approx (nE[XX^T])^{-1}\,(nE[Xy]) \\ &= (E[XX^T])^{-1}\,E[Xy]\end{aligned}$$

PR NPTEL course – p.56/140

That is easy to see let us quickly go through this; rows of A are X i, so A transpose will have columns as X i, so A transpose A will be X i, X i transpose some row. So, A at A transpose A is nothing but expect value of X, X transpose, but because there is input 1 by n here, that n comes here. Similarly, a transpose y because columns of A are X i will be summation i is equal to 1 to n X i times y i, which is approximately equal to n times expect value of X y.

So, if I do a transpose a whole inverse a transpose y, I get n times expect value of X, X transpose whole inverse into n times expect value of X y, this is same as the expression we have got earlier. And of course, this this n will become 1 by n because the inverse that will cancel with this n. So, is actually equal to expect value of X, X transpose whole inverse into expect value of X y that is the reason whether or not we use that n, the final solution is the same. As I said we are minimising J, so a constant in front of j does not change the minimiser, so that is the minimiser.

(Refer Slide Time: 25:20)



- We can think of least squares method as minimizing the mean square error

$$J(W) = \frac{1}{2} E\left[X^T W - y\right]^2$$

- Equating gradient to zero we get, as earlier,

$$W^* = (E[XX^T])^{-1} E[Xy]$$

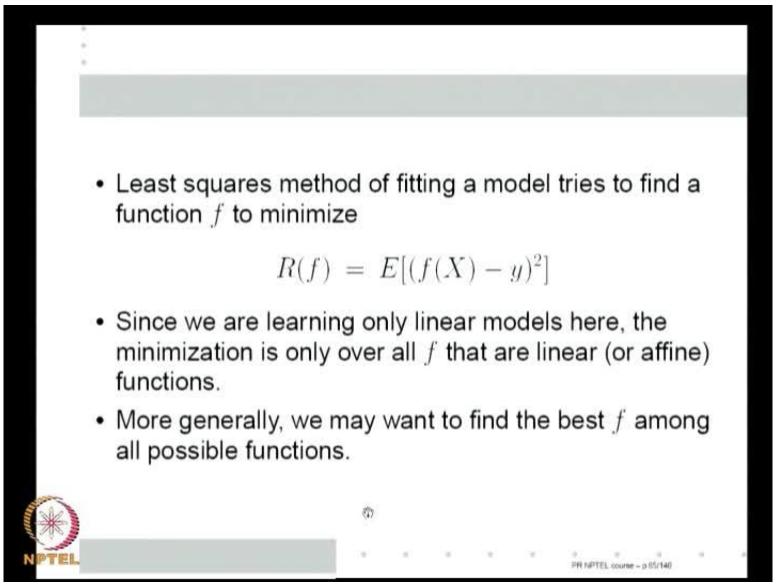
- This can be used to understand the model we get.

PR NPTEL course – p.62/140

So, to restate what I have been saying, we can think of least squares method as minimising the mean square error, essentially we are approximating the expectation by sample mean. But otherwise is actually minimising expect value of X transpose W minus y whole square, given IID samples of X comma y. And as we saw the minimiser is expectation of X, X transpose whole inverse into expect value of X y. Written in this style if I know something about the covariance matrix of X and so on so forth, I might be able to crunch this expression X, X transpose whole inverse into X y.

And hence, it allows to understand the kind of model we get. So, time permitting later on we will we will see some more examples of this, but it is good to remember that we can actually think of mean square, least mean square method as minimizing this expectation and this is the ideal solution.

(Refer Slide Time: 26:26)



- Least squares method of fitting a model tries to find a function f to minimize

$$R(f) = E[(f(X) - y)^2]$$

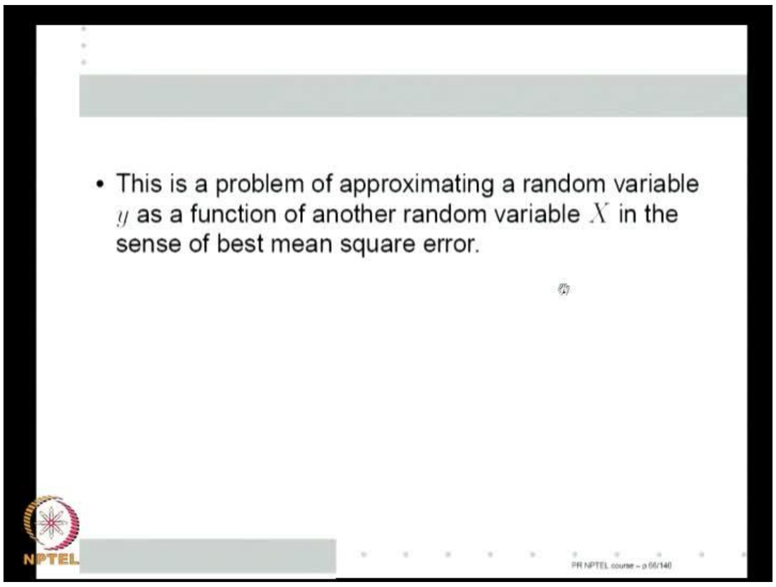
- Since we are learning only linear models here, the minimization is only over all f that are linear (or affine) functions.
- More generally, we may want to find the best f among all possible functions.

PR NPTEL course – p.85/140

Now, as you said least squares method is fitting a model that is a function of X, f of X, to minimise some expect value of f X minus y whole square. Since, I put some f here I changed the function to R of f, if you remember R is what we have using for the risk in the beginning we will come back to that later, but let us call this R of f now. So, if I put different functions I will get different values for R or for R of f and essentially the idea is which function is the best function. So, I am choosing a function f to predict y, so on an X I will predict f X, all the true value is y.

So, f X minus y whole square is the error and I am taking a mean of this other and trying to find R f that has the least error. So, there is what the kind of fit we are trying to make we are trying to fit a function. So, far in the method that we have considered we hare we are learning only a linear model here. So, the minimization is over not all possible f’s, but only the linear or affined functions f that is what our least square solution gives linear least square solution gives. But more generally we may actually want to ask, if I do not have this restriction, if I can somehow minimise this overall possible f. What is the best f among all possible functions? We can ask this question certainly whether or not we can computationally find such an f, we can ask that question what will be the best f if I do not put this linearity restriction, but say overall possible functions f find the find the best f.

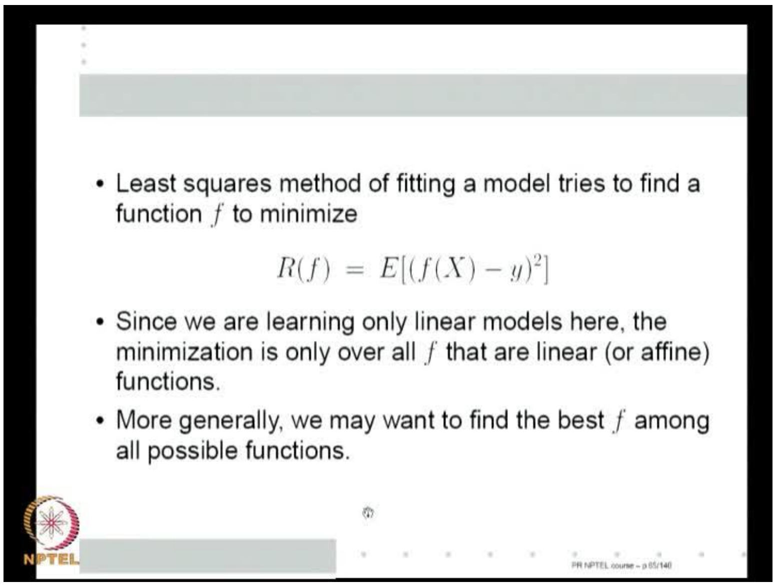
(Refer Slide Time: 28:08)


- This is a problem of approximating a random variable y as a function of another random variable X in the sense of best mean square error.

PR NPTEL course – p.66/148

Then there is a problem of approximating a random variable y as a function of another random variable X in the sense of best mean square error that is what this means.

(Refer Slide Time: 26:26)


- Least squares method of fitting a model tries to find a function f to minimize

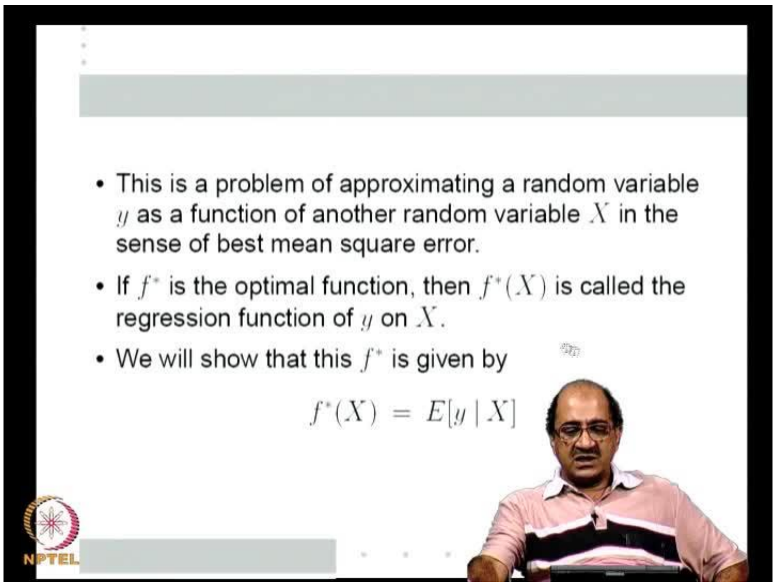
$$R(f) = E[(f(X) - y)^2]$$

- Since we are learning only linear models here, the minimization is only over all f that are linear (or affine) functions.
- More generally, we may want to find the best f among all possible functions.

PR NPTEL course – p.65/148

There are some random variables X and y and they are somehow related. So, y may not be a determinist function of f, but X and y have some join distribution, so y is stochastically determined by X. So, I want to find if I want to predict the value of the random variable y based on the random variable X that is I want a function of X, which is the best predicted for y, best in the sense of minimising the mean square error.

(Refer Slide Time: 28:45)



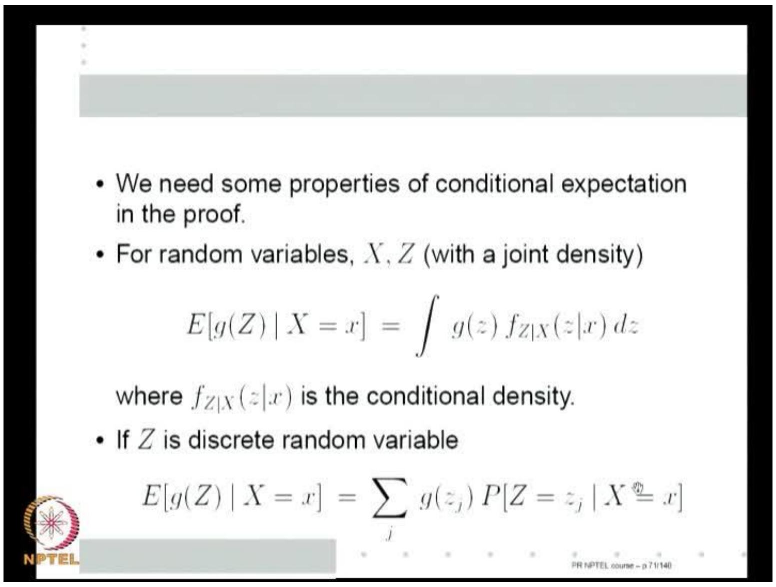
- This is a problem of approximating a random variable y as a function of another random variable X in the sense of best mean square error.
- If f^* is the optimal function, then $f^*(X)$ is called the regression function of y on X.
- We will show that this f^* is given by

$$f^*(X) = E[y \mid X]$$

So, our problem is approximating a random variable y as a function of another random variable X in the sense of best mean square error. If f star is such an optimal function, then such an f star is called the regression function of y on X. Regression function of y on X is the function of X that is the best predictor of y, where best is in the sense of minimum mean square error. We will currently show that this function the best function is given by expected value of y given X, this is a conditional expectation of the random variable y given the random variable X.

A conditional expectation is the function of X, so this will be a function of X. So, the best function turns out to be the conditional expectation. In this course I have actually assumed that all the readers, how basic background on probability we have been using normal distribution, other concepts from probability, such as density functions, distribution function, joint densities, joint distributions, expectation movements and so on. But still conditional expectation is a little bit an advanced topic may not be covered in all first level probability courses.

(Refer Slide Time: 30:08)



- We need some properties of conditional expectation in the proof.
- For random variables, X, Z (with a joint density)

$$E[g(Z) \mid X = x] = \int g(z)\, f_{Z|X}(z|x)\, dz$$

where $f_{Z|X}(z|x)$ is the conditional density.

- If Z is discrete random variable

$$E[g(Z) \mid X = x] = \sum_j g(z_j)\, P[Z = z_j \mid X = x]$$

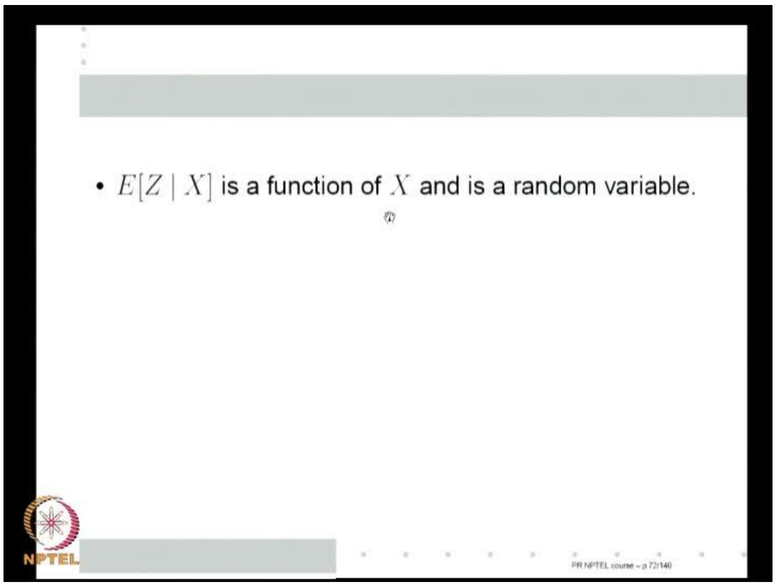
PR NPTEL course – p.71/140

So since, I had need some properties of conditional expectation as the proof, I will quickly give a short overview of conditional expectation of course, this is not teaching conditional expectation, it is just a short overview. So, given any two random variables X and Z, let us suppose they have a joint density at both they are continuous random variables. So, then the conditional expectation of any function of Z j of Z, conditioned on X at a value the value of the conditional expectation when X is equal to little x is given by this integral, g z, f of z given x z given x, d z, where f of Z given x is the conditional density.

As you know the conditional density Z given X is the joint density of Z and X divided by the marginal density of X. So, essentially conditional expectation is an expectation with respect to the conditional density of that given X. Normally if I am taking expectation of Z expectation of g Z, it is an expectation integral with respect to the marginal density of Z. Here instead of taking the expectation with respect to the marginal density we are taking expectation with respect to the conditional density of Z given X. Because, in particular of g is an identity function of g Z is Z, then this becomes conditional expectation of Z given Z is equal to x.

Of course, if Z is a discrete random variable then corresponding to this integral becomes a summation if it takes values Z 1, Z 2 so on. So, it will be g Z j and once again with respect to the conditional probabilities, probability Z is equal to Z j conditioned on X is equal to X. Of course, this integral if g Z is a real valued function, then this integral is over the real line otherwise, if g Z is a vector valued function this will be over an appropriate real Euclidean space, that is why I did not put any limits of course, nor it does not mean that it is a indefinite integral. This is over the entire space this integral.

(Refer Slide Time: 32:06)



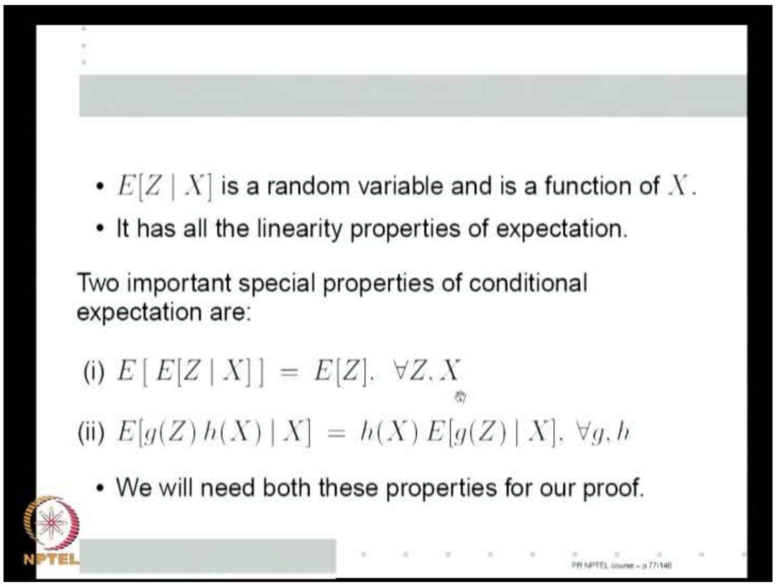
- $E[Z \mid X]$ is a function of X and is a random variable.

PR NPTEL course – p.72/140

So, in both cases we have defined conditional expectation of some function of Z conditioned on Z at a particular value of X. So, the first thing we have to understand is expect value of Z given X is a function of X, by that I mean if g suppose g Z is Z. So, conditional expectation of Z given X at a value little x taken by x is given by this expression, in this expression I am integrating over Z.

So, X will remain, so this will have be this will be some expression involving x. So, for every value X takes I get a value for conditional expectation, it is thus conditional expectation of Z given X is a function of X, for every value X takes there is a value for expected value of Z given X. So, conditional expectation Z given X is a function of X. Since, X is random any function of X will be random.

(Refer Slide Time: 33:20)



- $E[Z \mid X]$ is a random variable and is a function of X.
- It has all the linearity properties of expectation.

Two important special properties of conditional expectation are:

(i) $E\,[\,E[Z \mid X]\,] \;=\; E[Z],\;\; \forall Z, X$

(ii) $E[g(Z)\,h(X) \mid X] \;=\; h(X)\,E[g(Z) \mid X],\; \forall g, h$

- We will need both these properties for our proof.

PR NPTEL course – p.77/140

So, conditional expectation is a random variable that is the first important concept to note, unlike an ordinary expectation, which gives me a number conditional expectation gives me a random variable. Of course, it is an expectation integral so all linearity properties will satisfy, it is very easy to see conditional expectation, g 1 plus g 2 given X will be conditional expectation of g 1 given X and plus g 2 given X simply because of the linearity of the integral or the summation.

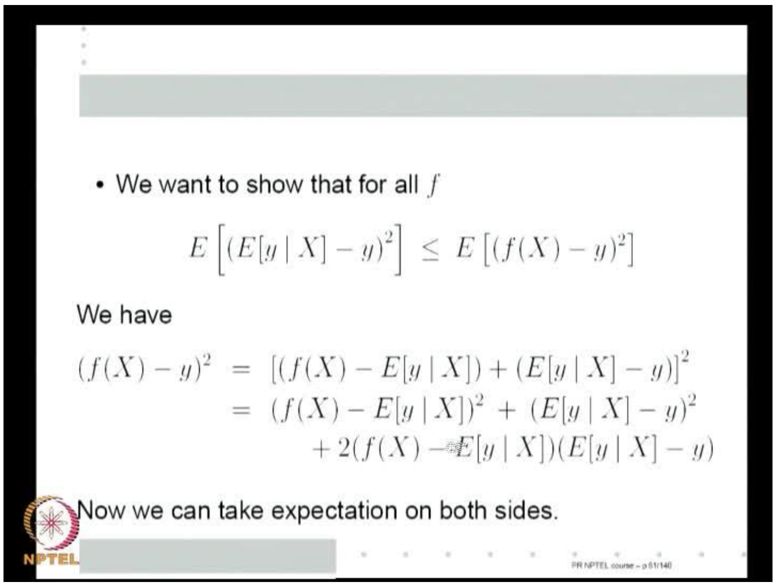
So, all the linearity properties are of course, satisfied. But in addition because of this special properties special nature of the conditional expectation, there are two very important special properties of conditional expectation. So, the first property is expectation of expectation Z given X is expectation Z, see the inner expectation is a conditional expectation that conditional expectation gives me a function of X.

So, the outer expectation is the expectation of that function of X. So, the outer expectation is with respect to the distribution of X, because what is inside that expectation is a function of X well conditional expectation is a function of X. This is of course, just expectation of Z, so this expectation with respect to the distribution of Z, this is true for all Z and X. So, I might have given any Z, one way of finding its expectation is to find the expectation of the conditional expectation, conditioned on any X that is convenient to us.

This often used in practice to find some certain expectation, but any way this is one property expectation or conditional expectation is the unconditional expectation. The second property because the conditional expectation with respect to the conditional distribution, anytime that is a function only of the conditioning random variable behaves like a constant inside the conditional expectation. So, if I am taking conditional expectation of g of Z into h of X conditioned on X, then it is same of as h of X into conditional expectation of g of Z given X.

So, anything inside the conditional expectation here h of X, which is a function only of the conditioning variable behaves like a constant and comes out of the expectation. In particular suppose g of Z is 1, then what does that mean of course, conditional expectation of any constant is the constant it is easy to seek in the definition. If g of Z is 1 for example, then conditional expectation of h of X given X will be simply h of X. These are two very important properties of conditional expectation, both of them we need for our proof.

(Refer Slide Time: 35:19)



- We want to show that for all f

$$E\left[(E[y \mid X] - y)^2\right] \le E\left[(f(X) - y)^2\right]$$

We have

$$\begin{aligned}(f(X) - y)^2 &= [(f(X) - E[y \mid X]) + (E[y \mid X] - y)]^2 \\ &= (f(X) - E[y \mid X])^2 + (E[y \mid X] - y)^2 \\ &\quad + 2(f(X) - E[y \mid X])(E[y \mid X] - y)\end{aligned}$$

Now we can take expectation on both sides.

PR NPTEL course – p.61/146

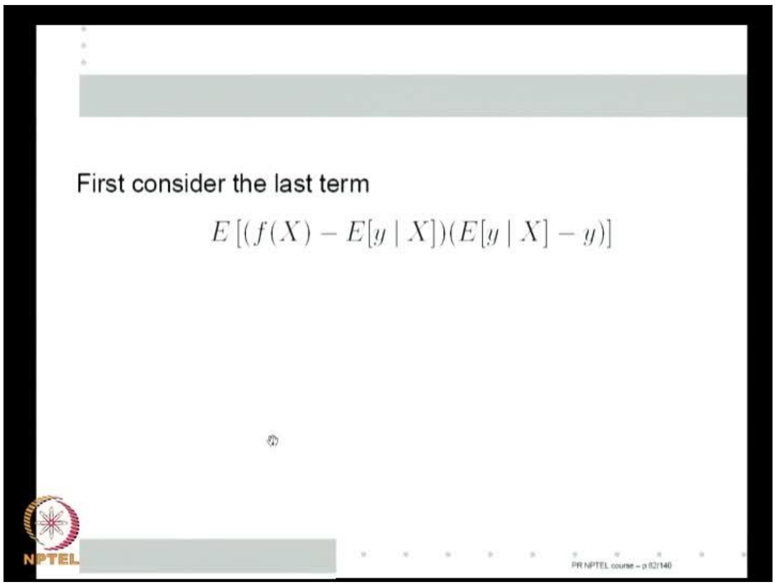
What is that we want to show? If I want to minimise expected value of f X minus y whole square, over all possible functions f, then that minimised function is the conditional expectation. So, what does that mean if I take expectation of conditional expectation of y given X minus y whole square that will be less than or equal to expectation of f of X minus y whole square for every f, this is what I want to show. To show that this expectation is minimised over all possible f, if I take f to be conditional expectation, I have to show if I plug the conditional expectation for f. Then what I get is less than what I get for any other f, this is what you want to show.

So, to do that first, let us write f X minus y whole square in an equivalent form which is convenient for the proof. So, f X minus y whole square, I can write as f X minus expectation of y given X plus expectation of y given X minus y whole square, I have done nothing we just added and subtracted a conditional expectation of y given X term. Now, I can I put brackets here to show how I intend to look at this, I look at this as the first term, this as the second term.

So, write a plus b whole square is a square plus b square plus two a b, to give me f X minus expected value of y given X whole square plus expected value of y given X minus y whole square plus the cross term the two a b term. The idea is that now, I will take expectations on both sides, because I will get an expectation of f X minus y whole square that is what I want here. One of these things here because expectation mean operated will expectation go inside, one of the terms here is expected value of y given X minus y whole square.

So, I get expectation of expectation of y given X minus y whole square that is the other term I want here. So, if I take expectation of both sides I get both the terms I want and then looking at the remaining terms I might be able to establish a relationship. To do that what we will do is we will first look at this cross term and show that if I take an expectation the cross term vanishes.

(Refer Slide Time: 37:19)

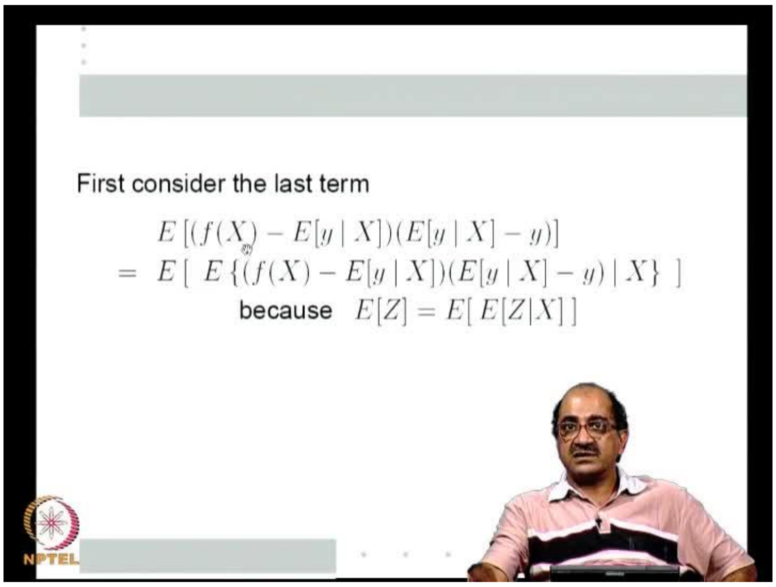

First consider the last term

$$E\left[(f(X) - E[y \mid X])(E[y \mid X] - y)\right]$$

PR NPTEL course – p.82/140

So, we will show it like this, first consider the last term that is expectation of f X minus expect value of y given X into expected value of y given X minus y that is the last term.

(Refer Slide Time: 37:40)


First consider the last term

$$\begin{aligned} & E\left[(f(X) - E[y \mid X])(E[y \mid X] - y)\right] \\ = \; & E\left[\; E\{(f(X) - E[y \mid X])(E[y \mid X] - y) \mid X\} \;\right] \end{aligned}$$

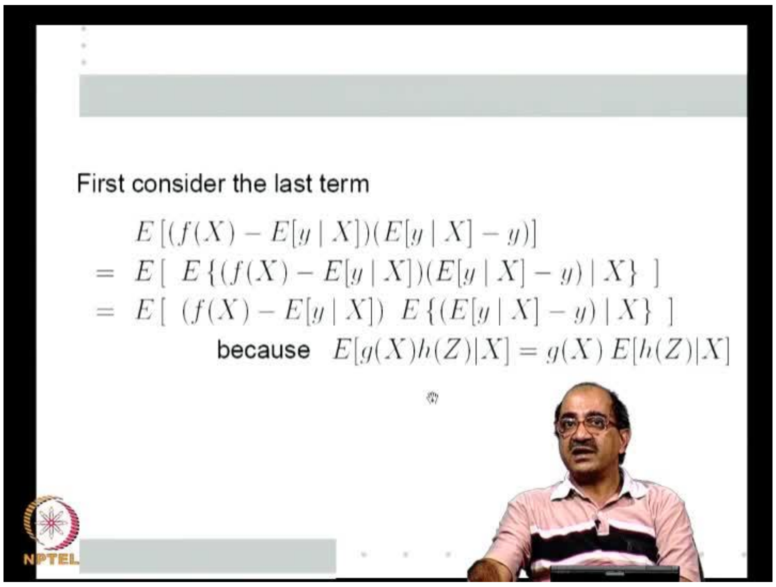
because $E[Z] = E[\,E[Z|X]\,]$

Further about the 2, 2 does not make any difference f X minus expected value of y given X into expected value of y given X minus y. I want to take expectation of this that is what it is. So, we can rewrite like this as we just now saw expectation of any random variable is expectation of a conditional expectation. So, I will write this expectation as expectation of expectation of this same thing f XX minus expected value of y given X into expected value of y given X minus y conditioned on X.

So, there is an inner expectation that is condition on x, so for the inner expectation I used the flower bracket, so that we can see and then that the outer expectation. So, I simply wrote this using expect value of Z is expect value of expect value of z given x. Now, what I can do is if I look at the let us look at the inner conditional expectation without cluttering the thing.

If I had look at the first term in the inner conditional expectation, f of X is a function of X, so it depends only on X, as we just now saw expected value of y given X is a function of X. So, this also depends only on X. So, this entire term f X minus expect value of y given X is a some function of X. So, that term behaves like a constant in the inner conditional expectation, so it comes out.

(Refer Slide Time: 38:48)



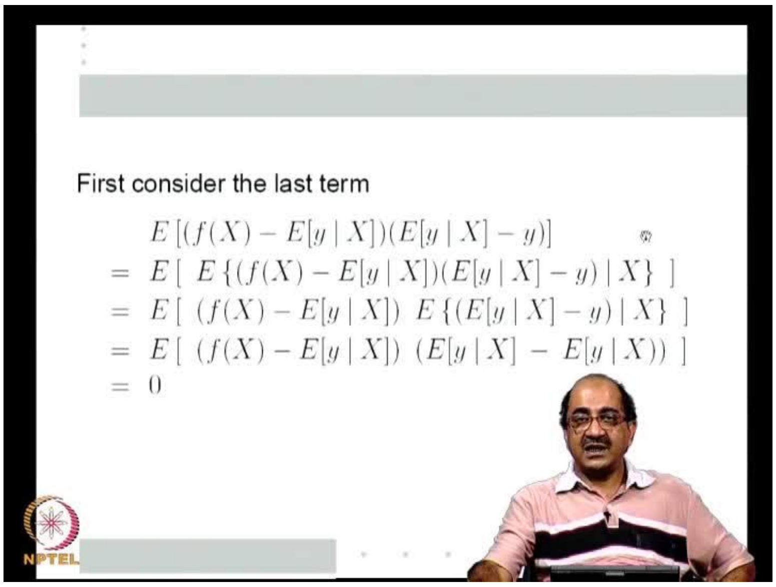
First consider the last term

$$
\begin{aligned}
& E\left[(f(X)-E[y \mid X])(E[y \mid X]-y)\right] \\
= \; & E\left[\; E\{(f(X)-E[y \mid X])(E[y \mid X]-y) \mid X\} \;\right] \\
= \; & E\left[\; (f(X)-E[y \mid X]) \; E\{(E[y \mid X]-y) \mid X\} \;\right]
\end{aligned}
$$

because $E[g(X)h(Z)|X] = g(X)\, E[h(Z)|X]$

So, I can write this as, so I have taken that f X minus expect value of y given X out of the inner conditional expectation and wrote expected value of the second term expected value of y given X minus y given X. So, here we used the second property of the conditional expectation. Now, you look at the inner conditional expectation this is expectation of some a minus b given X, there will be expectation of a given X minus b given X. a given X is expectation of expectation of y given X, given X expected value of y given X is a function of X. So, it will come out by itself, the second term is simply expected value of y given X.

(Refer Slide Time: 39:27)

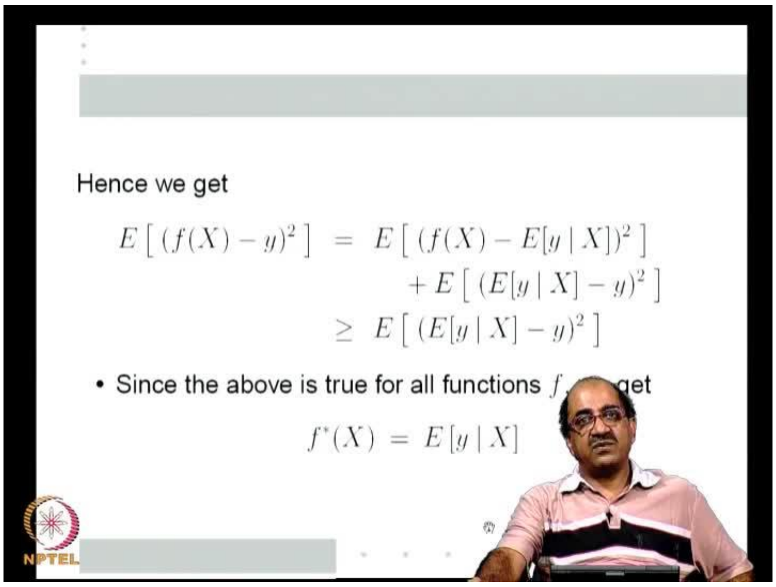


First consider the last term

$$
\begin{aligned}
& E\left[(f(X) - E[y \mid X])(E[y \mid X] - y)\right] \\
= \; & E\left[\; E\{(f(X) - E[y \mid X])(E[y \mid X] - y) \mid X\} \;\right] \\
= \; & E\left[\; (f(X) - E[y \mid X]) \; E\{(E[y \mid X] - y) \mid X\} \;\right] \\
= \; & E\left[\; (f(X) - E[y \mid X]) \; (E[y \mid X] \; - \; E[y \mid X)) \;\right] \\
= \; & 0
\end{aligned}
$$

So, if I push this inner conditional expectation inside I will get expected value of y given X minus expected value of y given X that gives me 0. So, using the two special properties of conditional expectation, we can show that the expectation of the third term in our expression goes to 0. Using this now what we have see this is what we had earlier, if I take expectations I will put expectation here that is expectation of this term, that is expectation of this term plus expectation of this term, this term will go to 0. So, expectation of this is expectation of this plus expectation of this.

(Refer Slide Time: 40:15)



Hence we get

$$E\left[\,(f(X)-y)^2\,\right] = E\left[\,(f(X)-E[y\mid X])^2\,\right] + E\left[\,(E[y\mid X]-y)^2\,\right]$$
$$\geq E\left[\,(E[y\mid X]-y)^2\,\right]$$

- Since the above is true for all functions f, we get

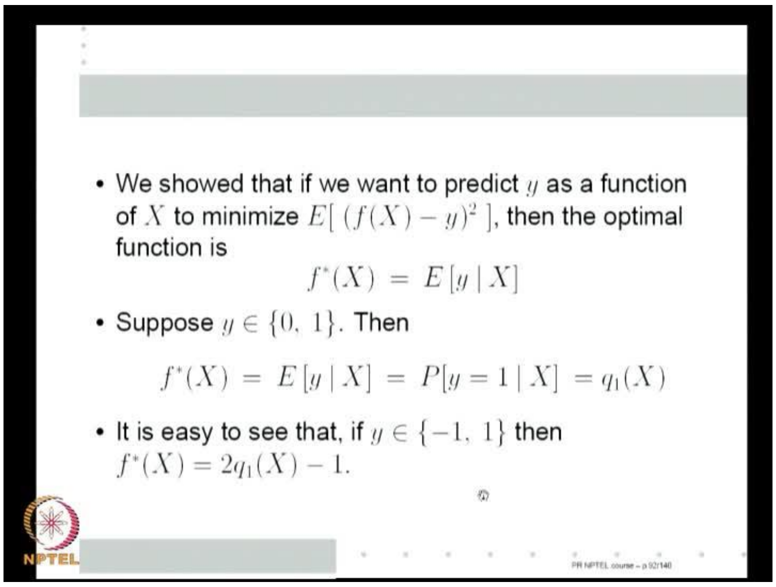
$$f^*(X) = E[y\mid X]$$

So, let us write that, so expected value of f X minus y whole square the expected value of f X minus expected value of y given X whole square plus expectation of expected value of y given X minus y whole square. Now, the right hand side is expectation of conditional expectation minus whole square, this is the term we are interested in plus expectation of some quantity, which is always positive. The expectation of square of some random variable expectation of square of some random variable is always greater than or equal to 0.

So, if I drop this term this will be greater than this, because this is obtained by adding some positive term to this term. So, expected value of f X minus y whole square is greater than or equal to expected value of expected value of y given X minus y whole square. This is true for every f, no matter of what function f i take, expect value of f X minus y whole square is always greater than or equal to expected value of conditional expectation of y given X minus y whole square.

So, this is since this is true for all f, we have we finally, prove the results that the optimal function to approximate y in a mean square sense is the conditional expectation of y given X. It is very important result about mean square estimation of course, the conditional expectation of y given X depending on the joint distribution of y and X, may be a linear function of X, may not be a linear function of X. We have not made any restriction on the functions over which you found the minimum or all possible functions if you find the best approximation then that is the conditional expectation of y given X.

(Refer Slide Time: 41:56)



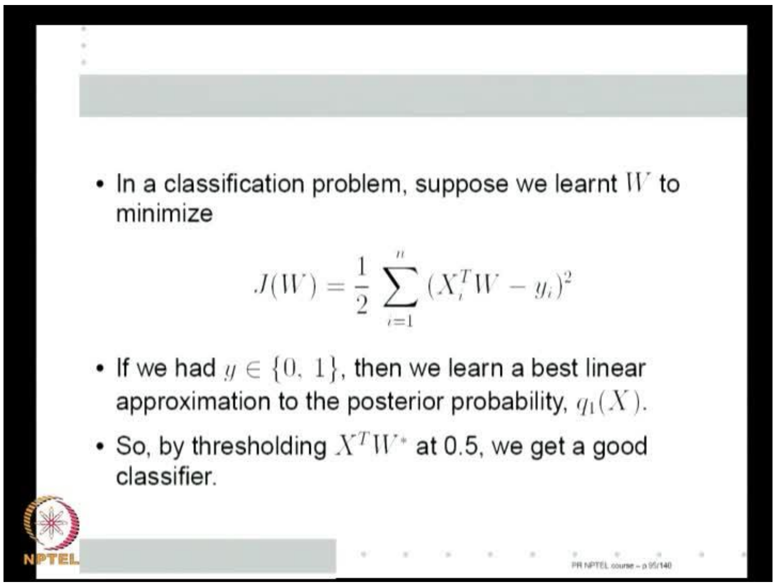
- We showed that if we want to predict y as a function of X to minimize $E[\ (f(X)-y)^2\ ]$, then the optimal function is
$$f^*(X) = E[y \mid X]$$
- Suppose $y \in \{0, 1\}$. Then
$$f^*(X) = E[y \mid X] = P[y = 1 \mid X] = q_1(X)$$
- It is easy to see that, if $y \in \{-1, 1\}$ then $f^*(X) = 2q_1(X) - 1$.

PR NPTEL course – p.92/140

So, what is that we have showed we show that if you want to predict y as the function of X to minimise the mean square error, then the optimal function is the conditional expectation of y given X. Now, suppose y is a binary variable, y to X only value is 0 1 then what is the conditional expectation, f star X, which is conditional expectation of y given X. From our definition y takes only two values 0 into probability of y is equal to 0 given X plus that is 1 into probability y is equal to 1 given X. So, that is only probability of y is equal to 1 given X and we already have a name prior probability y is equal to 1 given X in a classification situation. This is the posterior probability of class 1, so that is q 1 of X.

Similarly, suppose we take y to be minus 1 plus 1, then conditional expectation of y given X will be plus 1 into probability of y is equal to 1 given X plus minus 1 into probability of y is equal to minus 1 given X, that will be q 1 X minus q 0 x or q minus 1 X and that is q 1 X minus 1 minus q 1 X, so that is two q 1 X minus 1. So, if we if we are using in this prediction model in this mean square problem, if we are using if in a classification problem, if we are taking y to be 0 or 1. Then the optimal f star would be q 1 X, using y to be minus 1 or plus 1, then the optimal f star will be 2 q 1 X minus 1.

(Refer Slide Time: 43:26)



- In a classification problem, suppose we learnt W to minimize

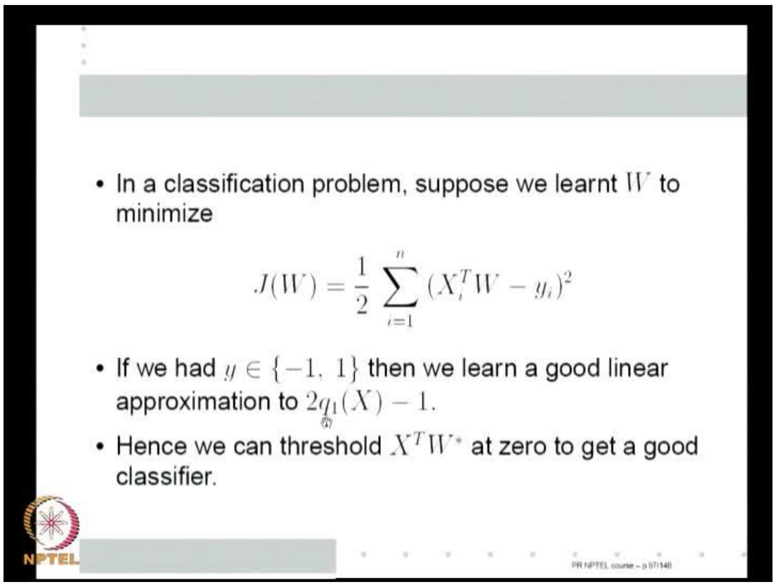
$$J(W) = \frac{1}{2} \sum_{i=1}^{n} (X_i^T W - y_i)^2$$

- If we had $y \in \{0, 1\}$, then we learn a best linear approximation to the posterior probability, $q_1(X)$.
- So, by thresholding $X^T W^*$ at 0.5, we get a good classifier.

PR NPTEL course – p 95/140

What does it mean? Suppose in a classification problem I have learnt W to minimise this X i transpose W minus y i whole square. So, if I think of this as f of X, the best function would be the conditional expectation, which I have given to be q 1 x or to q 1 x minus 1 depending on what is the coding for y is. So, we can say if we are using y is equal to 0 1, then we learnt the best linear approximation to q 1 x. If we had minimised our all functions, we would have learnt q 1 X at the optimal function. But since, you are not doing it to all functions, but only over linear functions we can say roughly hand wavingly that we learnt a best linear approximation to the posterior probability that is q 1 X.

(Refer Slide Time: 44:30)



- In a classification problem, suppose we learnt W to minimize

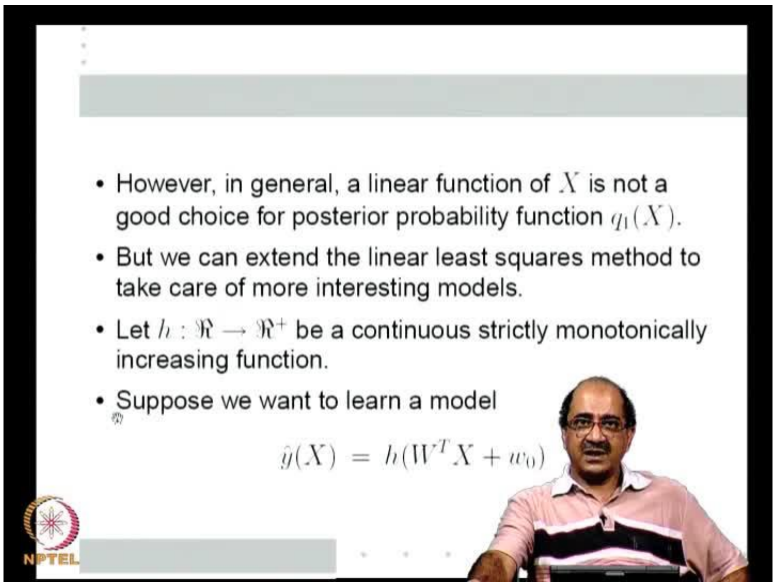
$$J(W) = \frac{1}{2}\sum_{i=1}^{n}(X_i^T W - y_i)^2$$

- If we had $y \in \{-1, 1\}$ then we learn a good linear approximation to $2q_1(X) - 1$.
- Hence we can threshold $X^T W^*$ at zero to get a good classifier.

PR NPTEL course – p.97/140

So, essentially if we solve this least square problem in a classification problem using y is equal to 0 1. Then after learning given any X, X transpose W is a good estimate of the posterior probability, which means a good classifier is obtained by thresholding X transpose W star at 0.5. Similarly, if I have taken y to be minus 1, 1 then we learn a good linear approximation to 2 q 1 X minus 1, which means we can threshold X transpose W star at 0 to get a good classifier. Earlier we said this hand wavingly in the sense, y is plus 1 minus 1 I am trying to match y with X, X transpose W that is why I can threshold it 0, but now, we exactly know y. In the mean square estimation, if we use targets as minus 1 plus 1, then we would be trying to approximate 2 q 1 x minus 1 and hence it.

(Refer Slide Time: 45:30)



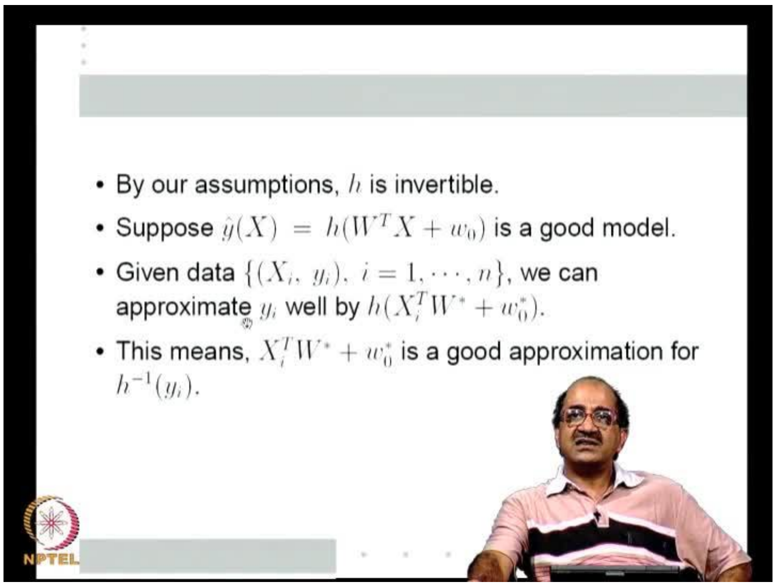
- However, in general, a linear function of X is not a good choice for posterior probability function $q_1(X)$.
- But we can extend the linear least squares method to take care of more interesting models.
- Let $h : \Re \rightarrow \Re^+$ be a continuous strictly monotonically increasing function.
- Suppose we want to learn a model

$$\hat{y}(X) = h(W^T X + w_0)$$

So, it does not matter they put minus 1 plus 1 or 1, 1 and 0, both cases I essentially learn some information about the posterior probability. So, mean square estimation is one way of learning posterior probability right. Of course, we said if we use 0 1, we will learn the best linear approximation of posterior probability of function. But of course, in general a linear function is not a good model for posterior probability, if the q 1 function X is a linear function of X that means q 1 of X 1 plus X 2 is q 1 X 1 plus q 1 X 2 probabilities do not add like that.

Because, X 1 plus X 2 is another X that will also have a posterior probability, but if you keep adding probabilities like that it will easily go beyond 1. So, in general a linear function of x is not a good choice for posterior probability. But linear least square method is very easily extendable to extendable; so that more interesting models than just such linear models for posterior probability can be learnt. Let us look at it a little more carefully, let us say h is some function continuous strictly monotonically increasing function R to R plus. So, we wanted to be positive because we wanted to model the posterior probability. By assume continuous strictly monotonically increasing so that h becomes invertible. And let us say instead of saying we learned a linear model W transpose X plus w naught, we say we actually learning a model, which is h of W transpose X plus w naught.

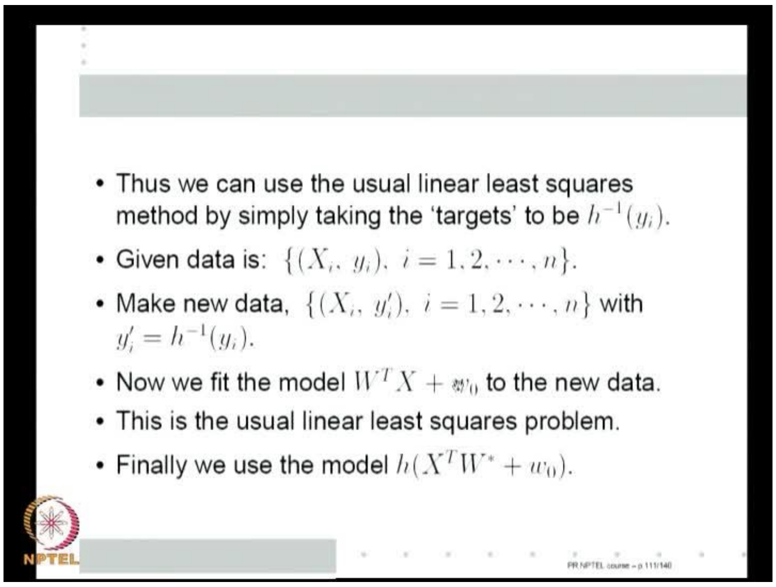
(Refer Slide Time: 46:40)



- By our assumptions, h is invertible.
- Suppose $\hat{y}(X) = h(W^T X + w_0)$ is a good model.
- Given data $\{(X_i, y_i), i = 1, \cdots, n\}$, we can approximate y_i well by $h(X_i^T W^* + w_0^*)$.
- This means, $X_i^T W^* + w_0^*$ is a good approximation for $h^{-1}(y_i)$.

By our assumption h is invertible, so suppose h of W transpose X plus w naught is a good model. What does that mean, given some data like this, essentially if W star and w naught star are the are the optimal values. Then h of X i transpose W star plus w naught star is a very good approximation for y i, that is what this model is good means. With since, h is invertible this is same as saying X i transpose W star plus w naught star is a good approximation for h inverse of y. So, if instead of predicting y if I am trying to predict h inverse of y, then I would get a linear model the usual linear model. So, putting a h there makes no difference.

(Refer Slide Time: 47:21)

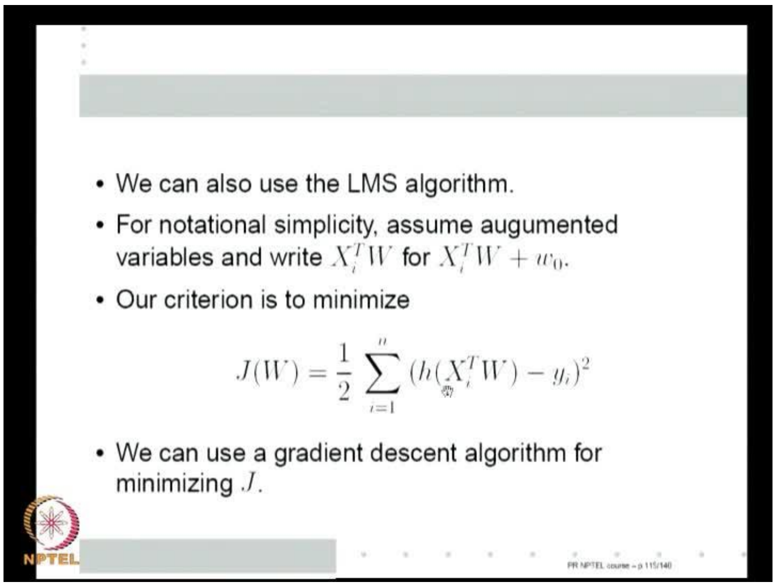


- Thus we can use the usual linear least squares method by simply taking the 'targets' to be $h^{-1}(y_i)$.
- Given data is: $\{(X_i, y_i), i = 1, 2, \cdots, n\}$.
- Make new data, $\{(X_i, y_i'), i = 1, 2, \cdots, n\}$ with $y_i' = h^{-1}(y_i)$.
- Now we fit the model $W^T X + w_0$ to the new data.
- This is the usual linear least squares problem.
- Finally we use the model $h(X^T W^* + w_0)$.

PR NPTEL course – p.111/140

We can use the usual linear least squares method by simply taking the targets to be h inverse of y i, rather than y i. What does that mean, suppose that is the given data X i, y i I make new data, where y i prime is h inverse of y i. So, for X i, y i prime I fit a linear model, W transpose X plus w naught is fitted to the new data. And then the usual linear least squares problems we will work now and finally, we can use the model h f, h f X transpose W star plus w naught. So, in this sense, I can easily generalise linear least squares not to just model such as X transpose W plus w naught, but also models, which are h f X transpose W plus w naught, where h is any strictly monotonically increasing continuous function.

(Refer Slide Time: 48:11)



- We can also use the LMS algorithm.
- For notational simplicity, assume augumented variables and write $X_i^T W$ for $X_i^T W + w_0$.
- Our criterion is to minimize

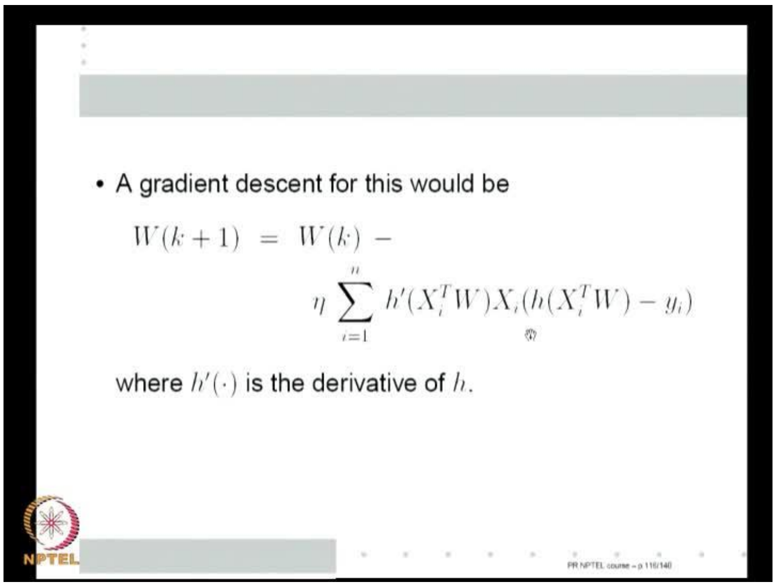
$$J(W) = \frac{1}{2} \sum_{i=1}^{n} (h(X_i^T W) - y_i)^2$$

- We can use a gradient descent algorithm for minimizing J.

PR NPTEL course – p.115/140

Of course, when learning this in the usually least square sense we could have used the matrix method we could have used LMS method. If we use the LMS method it is particularly simple I do not even have to rework the targets. To do this let us say to make my job easier instead of writing X i transpose W plus w naught, I will simply revert back to X i transpose W by using augmented variables.

(Refer Slide Time: 48:59)



- A gradient descent for this would be

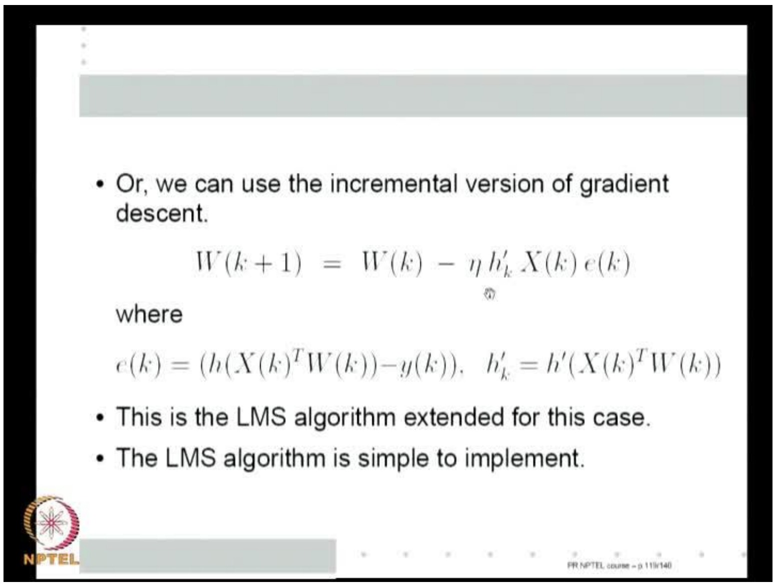
$$W(k+1) = W(k) - \eta \sum_{i=1}^{n} h'(X_i^T W) X_i (h(X_i^T W) - y_i)$$

where $h'(\cdot)$ is the derivative of h.

PR NPTEL course – p.116/140

So, this is what we want to minimise h of X i transpose W minus y i whole square summed over i. We can use the gradient descent or do the gradient descent, essentially the h was not there, then that will become X i transfer W minus y i into and the left multiplied with X i. Because, there is h the only thing I will get is an additional h prime term. So, that will be the gradient descent the extra term will be h prime of X i transpose W. If I differentiate this 2 will come out, then I need derivative of this with respect to W. So, that becomes h prime of this and then gradient of X i transpose W, which is the old term. So, my gradient descent will be the gradient term will be h prime of X i transpose W and to X i into h of X i transpose W minus y i, where h prime is the derivative of h.

(Refer Slide Time: 50:00)



- Or, we can use the incremental version of gradient descent.

$$W(k+1) = W(k) - \eta\, h'_k\, X(k)\, e(k)$$

where

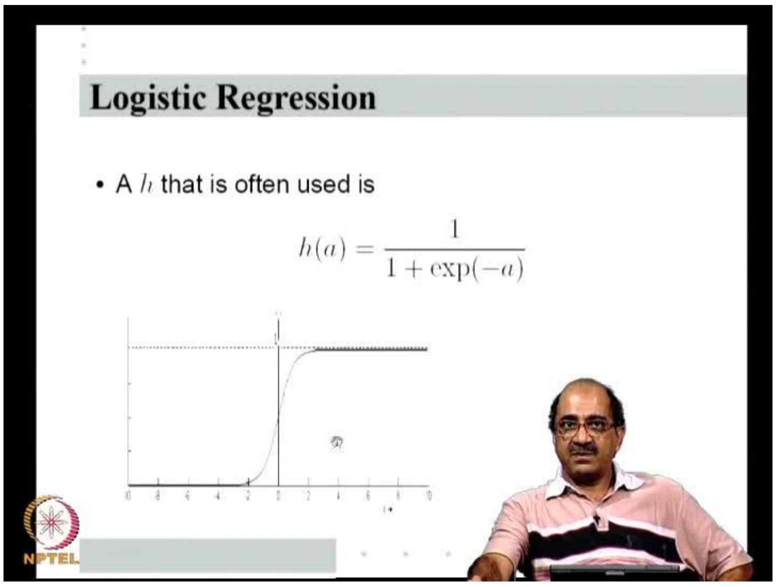
$$e(k) = (h(X(k)^T W(k)) - y(k)), \quad h'_k = h'(X(k)^T W(k))$$

- This is the LMS algorithm extended for this case.
- The LMS algorithm is simple to implement.

PR NPTEL course – p.119/140

So, earlier instead of h of X i transpose W minus y i, I had X i transpose W minus y i, but that is natural, because my new error now is h of X i transpose W minus y i, the X i was anyway there, only thing extra there is added is h prime. So, I do not have to do any recoding of targets or nothing I can just run the whole gradient descent with this extra term. Matter of fact if I write it in terms of incremental version is even simpler, W k plus 1 W k minus eta times, I wrote it as h prime k X k e k, e k is the error, usually it is X k transpose W k minus y k. Now, it is h of X k transpose W k minus y k, h prime k is nothing but the derivative of h evaluated at the current prediction.

(Refer Slide Time: 50:39)



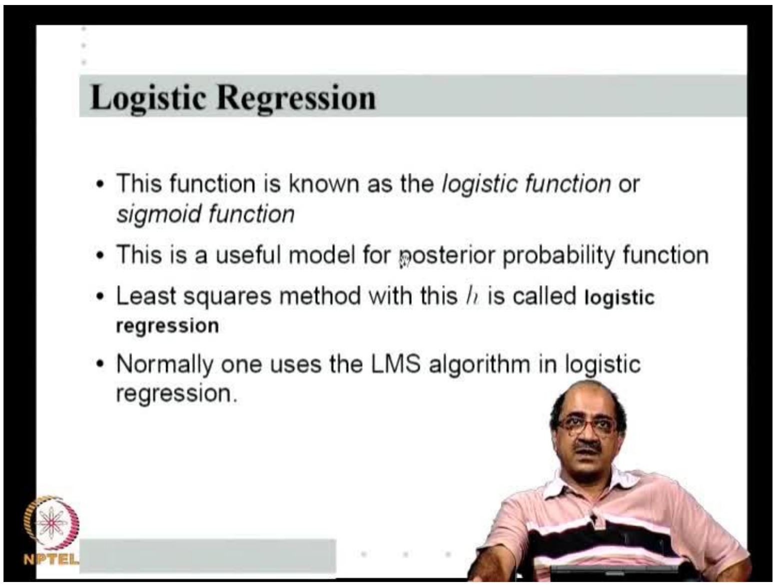
Logistic Regression

- A h that is often used is

$$h(a) = \frac{1}{1+\exp(-a)}$$

So, that will be the incremental version and the LMS algorithm in this case is easily extended to this case and it is very simple to implement. I do not have to do anything I just add a h prime term, the same error term earlier X k term I just add a derivative term. A h that is often used is this form h of a is 1 by 1 plus exponential minus a, we have seen this in our one of first few lectures, but anyway you might have forgotten. So, this has this generic structure. So, essentially for a positive large it becomes 1, because exponential minus a goes to 0 for a negative large this goes to infinity, so actual go to 0. So, in between it sharply transits and always takes value half h 0.

(Refer Slide Time: 51:26)

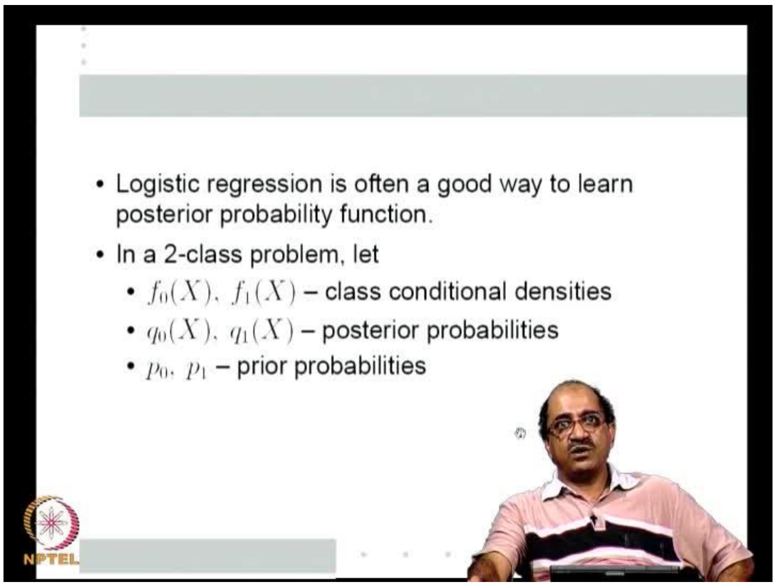


Logistic Regression

- This function is known as the *logistic function* or *sigmoid function*
- This is a useful model for posterior probability function
- Least squares method with this *h* is called **logistic regression**
- Normally one uses the LMS algorithm in logistic regression.

This function is called and as you can see this could be a probability function, at for various values of a this could be probability because it goes between 0 and 1. This function is known as the logistic function or the sigmoid function and this is useful model for posterior probability. Least squares method using this h, the modifier least square method that we have just now discussed that using this particular h is called logistic regression. Normally one uses the LMS algorithm as I told you in logistic regression.

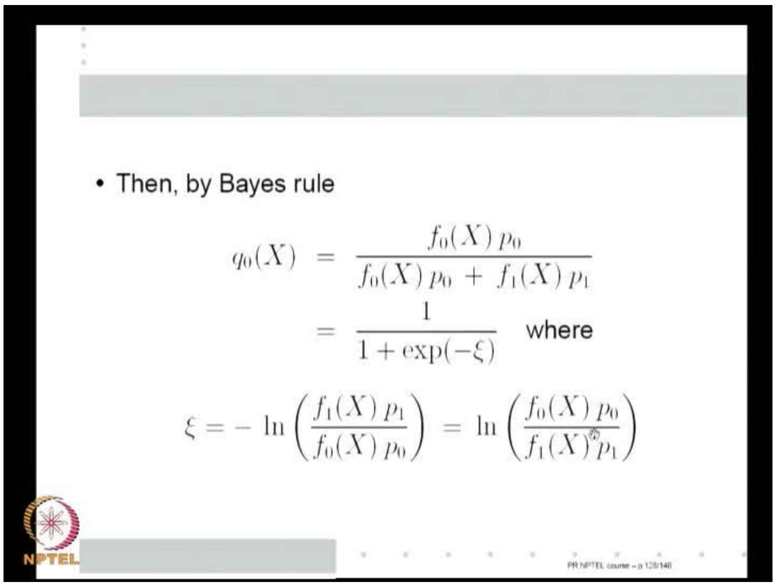
(Refer Slide Time: 51:47)



- Logistic regression is often a good way to learn posterior probability function.
- In a 2-class problem, let
  - $f_0(X)$, $f_1(X)$ – class conditional densities
  - $q_0(X)$, $q_1(X)$ – posterior probabilities
  - p_0, p_1 – prior probabilities

So, logistic regression is often a good way to learn posterior probability functions, we will just see in what way it happens. Let us take a two class problem and recall our notation f 0, f 1 are the class conditional densities, q 0, q 1 are posterior probabilities, p 0, p 1 are the prior probabilities.

(Refer Slide Time: 52:03)



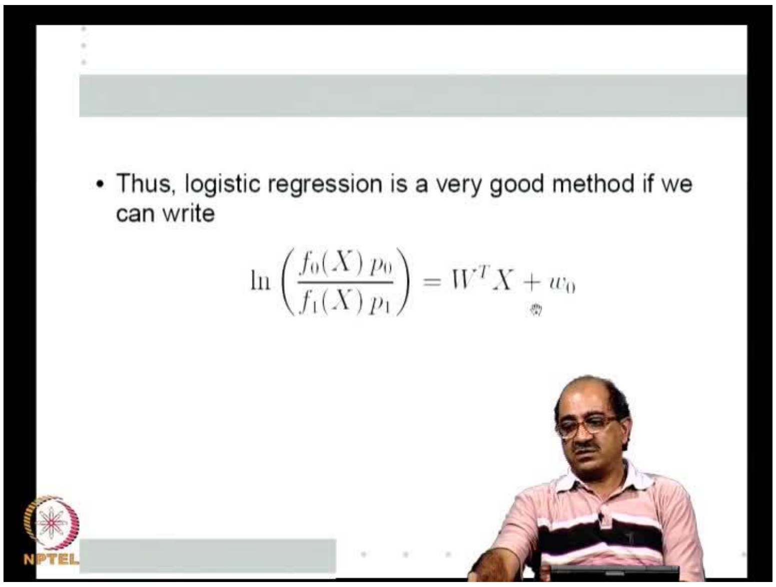
- Then, by Bayes rule

$$q_0(X) = \frac{f_0(X)\,p_0}{f_0(X)\,p_0 + f_1(X)\,p_1}$$
$$= \frac{1}{1+\exp(-\xi)} \quad \text{where}$$
$$\xi = -\ln\left(\frac{f_1(X)\,p_1}{f_0(X)\,p_0}\right) = \ln\left(\frac{f_0(X)\,p_0}{f_1(X)\,p_1}\right)$$

PR NPTEL course – p.128/140

Then what is the Bayes rule the posterior probability q 0 is given by f 0, p 0 by f 1, p f 0, p 0 plus f 1, p 1, this is the Bayes rule this is how the posterior probability is calculated. So, I can write this in this form, if I divide the numerator and denominator by f 0, p 0 I will get 1 by 1 plus f 1 X, p 1 by f 0 X p 0. Now, I can write any a as exponential l n a, so that is what I did, but I wanted a minus a, so I wrote xi itself as minus of l n, f 1 X p 1 by f 0 X p 0. So, if I write xi as minus of l n, f 1 X p 1 by f 0 X p 0. So, this becomes exponential l n, f 1 X p 1 by f 0 X p 0 exponential l n cancel each other, this becomes 1 by 1 plus f 1 X p 1 by f 0 X p 0, so that is what I want. And minus in l n is nothing you know you just because it is a fraction you just interchange the fractions simply becomes l n of f 0, p 0 by f 1, p 1.

(Refer Slide Time: 53:45)



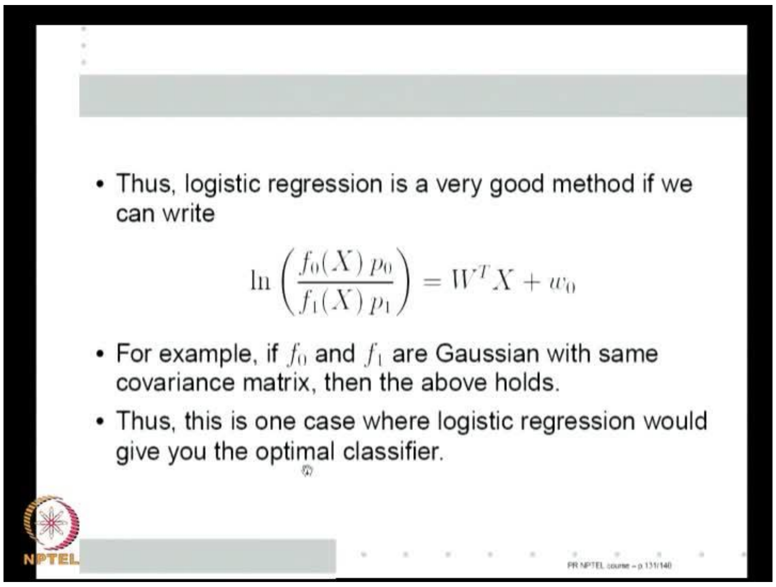
- Thus, logistic regression is a very good method if we can write

$$\ln\left(\frac{f_0(X)\,p_0}{f_1(X)\,p_1}\right) = W^T X + w_0$$

So, I can always write q 0 as 1 by 1 plus exponential minus xi, there is no problem writing it as a sigmoid the posterior probability, where the xi becomes this. So, posterior probability can always be written as sigmoid function, but the main question is can this xi be expressed as a linear function, because we wanted h of W transpose X plus w naught. So, asking can I write q 0 of x as 1 by 1 plus exponential minus W transpose X plus w naught.

So, can I write this l n f 0, p 0 by f 1, p 1 as a linear function? So, logistic regression is a very good method, if we can write l n, f 0, p 0 by f 1, p 1 as W transpose X plus w naught. If I can do that then it becomes 1 by 1 plus exponential minus W transpose X plus w naught. So, that is nothing but h of W transpose X plus w naught, where h is the sigmoid function. Hence, this perfectly fits the logistic regression to estimate the posterior probability.

(Refer Slide Time: 54:16)



- Thus, logistic regression is a very good method if we can write

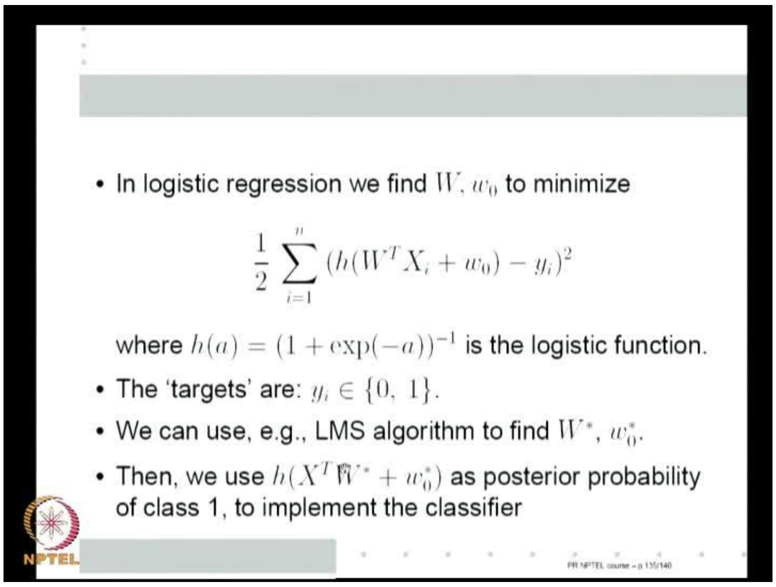
$$\ln\left(\frac{f_0(X)\,p_0}{f_1(X)\,p_1}\right) = W^T X + w_0$$

- For example, if f_0 and f_1 are Gaussian with same covariance matrix, then the above holds.
- Thus, this is one case where logistic regression would give you the optimal classifier.

PR NPTEL course – p.131/140

So, the question is can I the... So, hence we can conclude the logistic regression is a very good method if we can write l n f 0, p 0 by f 1, p 1. So, for all class conditional density is such that this expression can be approximated by a linear function logistic regression is a very good model. For example, if f 0 and f 1 are Gaussian, multidimensional Gaussians, but with the same covariance matrix, then as we saw when we studied the Bayes classifier for them. l n of this will be a linear function of X that is why the optimal classifier in Bayes case will be a linear discriminant function that you already seen. So, this will always be a linear function what it means is that, in this one situation, that is this is any example of one situation or many such situations, where logistic regression would give you the actual of bayes optimal classifier.

(Refer Slide Time: 55:12)



- In logistic regression we find W, w_0 to minimize

$$\frac{1}{2}\sum_{i=1}^{n}(h(W^T X_i + w_0) - y_i)^2$$

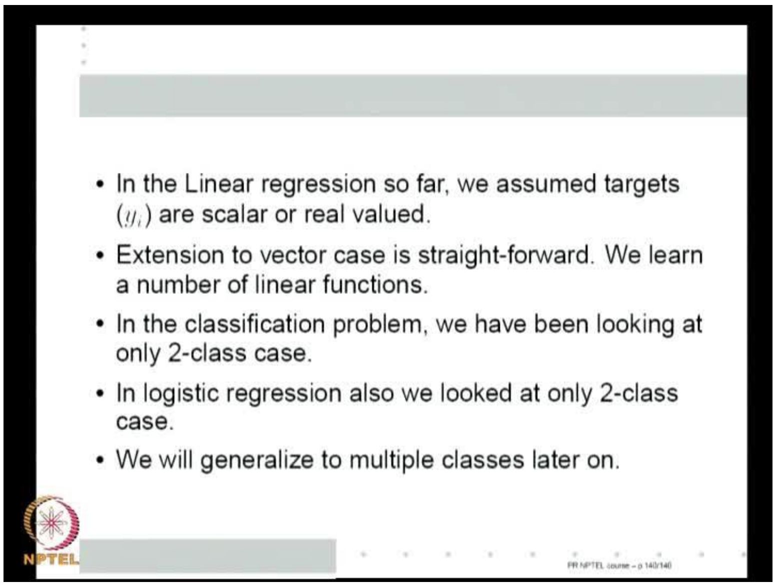
where $h(a) = (1+\exp(-a))^{-1}$ is the logistic function.

- The 'targets' are: $y_i \in \{0, 1\}$.
- We can use, e.g., LMS algorithm to find W^*, w_0^*.
- Then, we use $h(X^T W^* + w_0^*)$ as posterior probability of class 1, to implement the classifier

So, in that sense unlike the linear function logistic regression is a good model for posterior probability. So, to sum up in a in logistic regression we find W and w 0 to minimise, i is equal to 1 to n h of W transpose X i plus w naught minus y i whole square, where h is given by 1 by 1 plus exponential minus a that is the logistic function. We take the targets to be 0 on 1 for the two class case. We can use the LMS algorithm for finding W star and w naught, this is easier than because even the logistic function is strictly invertible beyond a level the inverse is inverse becomes very large.

But we do not have to find the new targets as I said, if we use the LMS algorithm it is simply adding 1 h prime term in the update. Then once we learn W one w naught we use h of X transpose W star plus w naught as the posterior probability estimate, which means given a nu X I calculate h of X transpose W star plus w naught and then threshold it at 0.5 to implement the classifier. Very often wherever linear classifiers are useful this logistic regression finds you the optimal classifiers.

(Refer Slide Time: 56:21)



- In the Linear regression so far, we assumed targets (y_i) are scalar or real valued.
- Extension to vector case is straight-forward. We learn a number of linear functions.
- In the classification problem, we have been looking at only 2-class case.
- In logistic regression also we looked at only 2-class case.
- We will generalize to multiple classes later on.

PR NPTEL course – p 140/146

So, to sum up in the linear regression so far we have assumed, so we have looked at linear regression linear classifiers, we have looked at how to look at it from the mean square error point of view, we looked at the regression function optimum mean square estimate. And arising out of it we looked at the logistic regression as a way to learn posterior probabilities. Now, so far we have considered linear regression when this targets are scalar we took our data to be X i can be any three dimensional vector, but y i is belong to R that by itself is not much different.

Extension to vector case is straight forward, essentially if y's are multidimensional then you have to learn a function from R d to say R 2 or R 3, then R d to R 2 or R 3 is simply three different functions each will be linear. So, I will just learn a number of linear functions. In the classification class also we will be looking at only two class case both for we regularly in a regression as well as in logistic regression. There is not quite clear what you would use in multiclass case, because using the 0 and 1 is what gave us this nice posterior probability estimates.

So, in general it is not so easy to look at multiclass case, but we will generalise the multiple classes later on. So, what we will do is next class we will look at some other connection between linear least squares estimate and the M L R Bayesian estimates that we considered earlier. And after that we will look at how to generalise this multi class case and look at multiclass linear discriminant functions. Thank you.