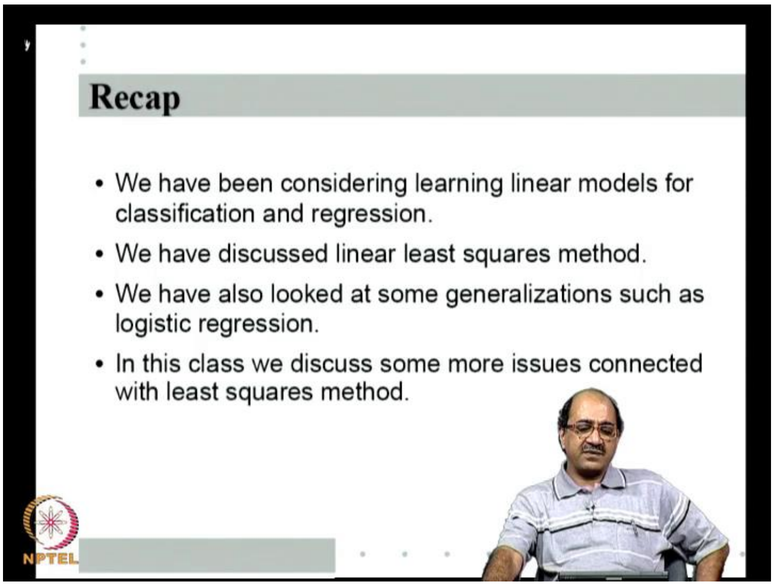
Pattern Recognition
Prof. P. S. Sastry
Department of Electronics and Communication Engineering
Indian Institute of Science, Bangalore

Lecture - 17
Logistic Regression; Statistics of Least Squares Method; Regularized Least Squares

Hello, good afternoon. Welcome to the next lecture in this pattern recognition course. To recall, we have been looking at learning linear models essentially, linear classifiers and linear regressors. So, linear models for both classification regression, we have been looking them together. We first looked at basically learning linear discriminant functions that is a hyper plane classifier, we looked at the special case of linearly separable classes, we looked at perceptron, and various ramifications of perceptron. Then we, we were considering linear least squares method.

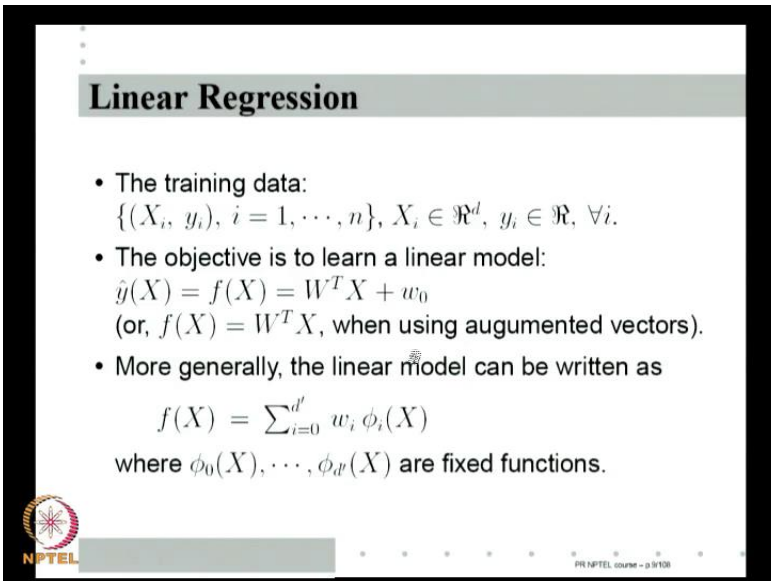
This is a general method for learning linear model for both classification and regression, essentially the idea is to minimize the sum of squares of errors. We have, we have looked at the linear least squares method, how one can find the minimizer using standard result from linear algebra? We also looked at how we can find the minimizer using gradient descent and we also looked at some small generalizations of the linear model such as logistic regression.

(Refer Slide Time: 01:01)



So, what we will do in this class is, we will, we will just briefly review this, and there are few interesting aspects in which linear least squares method can be viewed. So, we will, we will look at many small, small variations on the basic linear least squares method, and tight up with M L estimation, tight up with Bayesian estimation. Look at what is called Regularized Least Square, and so on. And end the class, with just a hint of another different method of learning, the linear classifier called the Fisher linear discriminant.

(Refer Slide Time: 02:26)



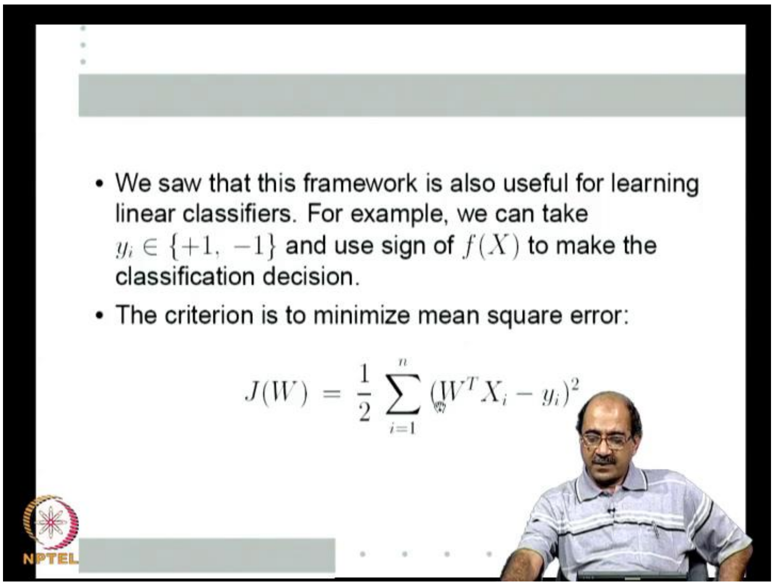
So, let us start by recalling linear Regression. So, for the regression, the training data is of the form, X i y i X i's are still some d-dimensional vectors, but y i are real valued targets now. So, we are given X i y i X i n R d y i n R, and the idea is to learn a functional directional period in X i and y i, as you are all learning to predict y given X, normally y i are called the targets. So, the targets are real valued.

So, that as we saw is the main difference between the classification, the regression problems. In the regression problem, the target is real valued as the classification problem, target is bind the valued in two class or finitely many valued, but essentially we have been looking at boundary valued currently. So, in a regression problem given data like this, the objective is to fit a linear Model. So, we want to predict y given any X, using some linear affine function W transpose X plus w naught. As we seen of course, we can absorb that constant, whenever the constant is not particularly important by simply assuming augmented vectors. All means is that, we take the X and put an extra component of 1 in it, so that the new X is now in d plus 1 dimensional space, and we assume the W vector has w naught as its first component.

So, that we can always represent the linear Model as W transpose X. So, essentially the idea is to learn a linear Model in this notation, W transpose X, actually we also saw that the the we do not have to only use X in general. A linear Model can be written as, f X as a sum of some linear combination, of some fixed basis functions. So, if phi 0 X phi 1 X, and phi d prime X are some fixed basis functions, then my linear Model can also be written as f X, is equal to some w i phi i X.

So, as long as all the phi's are fixed functions, we have seen the same methods of linear least squares regression, linear Least squares with instead of learning classifiers, we will work. So, even though most of the time, we will be only considering W transpose X, as I said it will work equally well for phi i X. One example, we saw is say just curve fitting in when X is also real valued, we will come back to that problem to introduce some more concepts this class, but in general a linear model is can be also written as w i phi i X and where phi 0 to i, just put d prime here because it does not have to be same as the dimension of X here, it can be any arbitrary number.

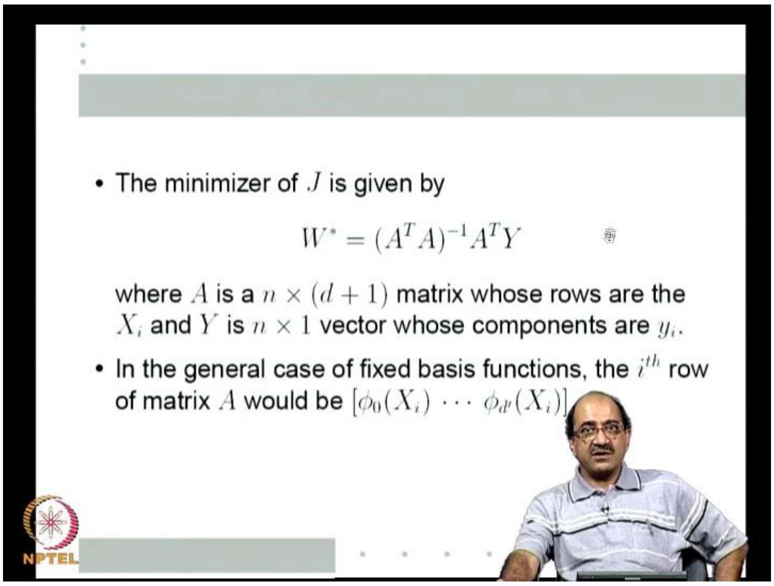
(Refer Slide Time: 05:17)



So, all such models are also learnt using the same method. Further as we saw, we can also learn classifiers using this method. We, we will still learn a W transpose X, we take the targets y to be plus 1 minus 1 in a two class case and then after learning f of X as W transpose X, given a new X we calculate f X and threshold it at 0 to decide the classification decision.

So, the same model can also be used for classifiers. In all cases, our criterion has been to minimize the mean square error. Essentially, somehow square error mean should have, should mean that we could have put 1 by n. Also as we seen last class by the way, we put that 1 by n or not makes no difference, because we are only interest in the minimizer of this function.

(Refer Slide Time: 06:56)

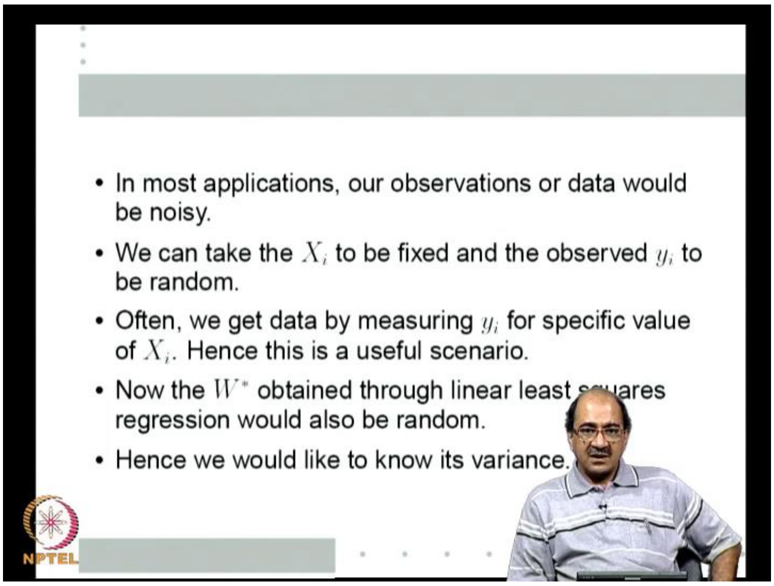


So, the criterion function is sum of squares W transpose X i, is what my model would have set on X i Y i, is the actual target in that example W transpose X i minus Y I, whole square is the other, and I am trying to find a W to minimize sum of squares of errors that is the reason, if I put Y i plus 1 or minus 1, I would learn something so that for all plus 1 patterns, W transpose X is closer to plus 1 all minus 1 pattern, W transpose X i is minus closer to minus 1. And that is why thresholding, W transpose X would give us a good way of learning linear classifiers also. We have seen that the, the minimizer of this can be directly written as W star is a transpose A whole inverse A transpose Y and our notation. We will always putting a star for anything that is the optimization solution of some criterion function.

So, the W that optimizes this J W is called the W star, that is given by A transpose A whole inverse A transpose Y, where A is the n by d plus 1 matrix, whose rows are the given patterns X i. So, of course X i's are augmented that is why they are d plus 1 dimensional. So, in stack of all the given, X i as the rows of this matrix. Yes, that is why the A will be n by d plus 1 matrix and capital Y is the, is a n vector obtained by stacking of all the targets y 1 y 2 y n.

So, capital Y is an n by 1 vector, and then this is the optimal solution as we already seen A transpose A inverse A transpose is called the generalized inverse. Essentially, what this is trying to do is to project Y on to the column space of the A matrix. In the more general case, when we use fixed basis functions the solution is still given by this. Basically, in the more general case this instead of becoming W transpose X i, it become W transpose capital phi X i whose components are phi 0 X i phi 1 X i so on up to phi b prime X i. When we use that more general model, the only difference in the solution is that the i th row of a matrix instead of being the, the i th data sample X i, it becomes phi 0 X i phi 1 X i phi d prime X i, that is the only difference when we use the case of fixed basis functions.

(Refer Slide Time: 09:01)



Now, let us look at a few few generalizations of this. So, far we have just being taking our X i and Y i, as given data we are not worried about any probability model of generating the data. So, let us look at some, some of these issues. The only reason we want to fit a model is that obviously the data is not exact. So, there is no they may not be any specific function which exactly gives for Y i for each X i, specifically linear function that gives Y i exactly for each X i. This often happens because of observations of noisy.

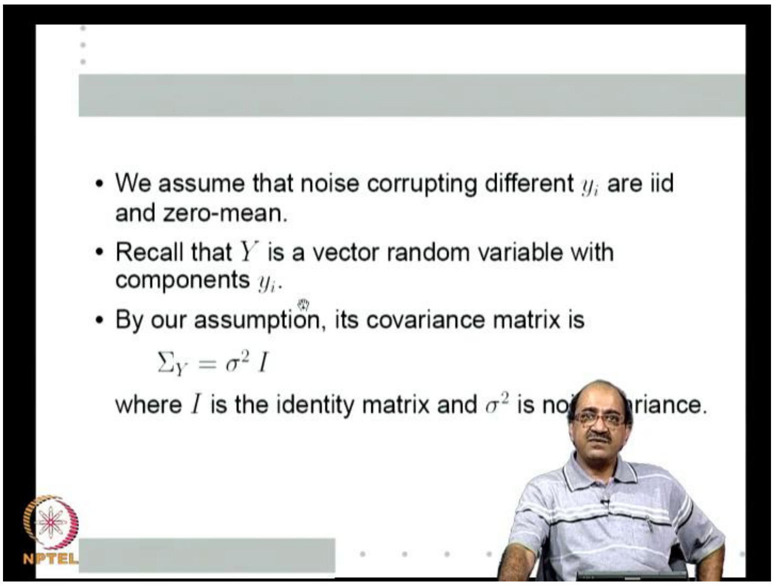
So, we want to get a good fit for noisy observations, and that is, that is the reason why we are summing this squares of errors as you seen is like expectation. So, let us suppose we take X i fixed, but the observations Y i to be random, random in the sense. They are, they are actually observations or their noise cap for observations of some function, f of X i, so it is like f of X i plus noise. So, observations Y i have noise, so they are random. But, we take X i to be fixed. This is a very convenient model is a useful scenario because very often the way, we get data is we, we take some X i, and measure the Y i that is how we generate the example data X i Y i, for which we, we fit a function. The most of our first experience with curve fitting is in physics experiments.

So, there is one control variable, and then you measure the value of the dependent variable, and then asking is there a functional relationship change. So, we pick some X i's and then measure Y i. So, it is good to think of Y i as where all the randomness is. So, because Y i are random, and we are, we are learning a W based on the Y i's. The W star that we obtained through linear least squares regression would also be random.

So, essentially what we want is how much is the variance in this W because, that tells us the amount of error in our fit. So, it is like the, the actual Y i given in the data set are not the true values. There are error bars, which Y i could be you know plus minus J of Y i. So, if there is such noise and I do not know how much the noise, but I take whatever is the actual measured value as the given value on fit the function, then how close is my W to the actual underlying W that is that is your question.

And, while we do not know the underlying W, we can certainly look at the variance of the fitted W. We do not know if indeed there is a true linear model, if it is then we can think of our linear least squares method. As an unbiased estimator of that linear model, and if you can look at is as an unbiased estimator, as you know its variance gives me least mean square error. So, in that sense, it is nice to be able to calculate the variance of W. So, how do we calculate the variance of W?

(Refer Slide Time: 12:17)

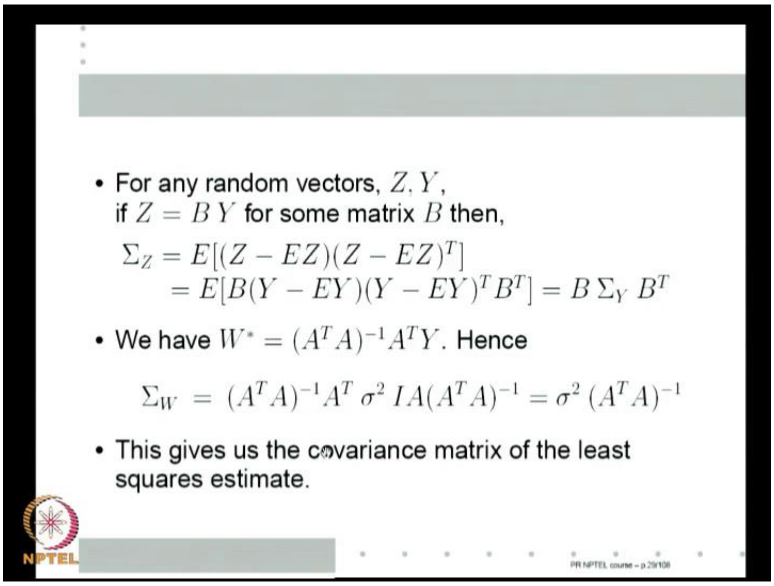


So, we we look at a very simple case this can be extended, but this is good enough for to get an idea of how these things are done. Let us assume the noise corrupting different Y i are IID, and zero mean, that is each observation is corrupted by similar noise, that is what IID stands for independent identical distributor. So, IID noise means similar noise. And, we assume 0 mean because you know, there should not be any bias in the noise. Now, capital Y is a vector whose components are y i, if y i's are random variables capital Y is a random vector, n-dimensional random vector, whose components are y i and this assumption means that each y i has zero mean, and fixed variance because they are IID, and they are independent.

So, the covariance between any two components of y 0, which means that given these assumptions the covariance matrix of the vector random variable is sigma square i, where i is the identity matrix, because it will be diagonal matrix, because there is no covariances all covariance express covariance between any two components of the capital Y vector is 0.

So, the, the covariance matrix of the capital Y vector would be diagonal and in addition, because we assuming that each is the noise in each y i is IID all of, all of them will have the same variance. So, we are writing the covariance matrix as sigma square i. Of course, assuming that it is sigma square i makes our makes our final calculation easier. As you will see, but even if it is not sigma square i even if it is diagonal and is still good enough even otherwise we can, we can get an expression, but this will give us simpler expression.

(Refer Slide Time: 14:32)



So, given that this is the covariance matrix. Let us for the, for the moment assume that we know sigma square, of course we are not saying who gives us sigma square, but let us assume that we have sigma square. So, if I know that we have sigma square, then can we calculate what is the variance in the fitted W star. For that, just recall a simple useful identity in random variables. Let us say, we have two random vectors Z and Y and Z is written as B times Y, some for some matrix B. Z and Y need not even have to be of the same dimension, because there is a matrix here.

But, Z is written as B times Y, for some matrix Y, then the question is can I calculate the covariance matrix of Z given the covariance matrix of Y. Because W star is given as some matrix multiply by capital Y this is an important question for us. How do we do this? So, the covariance matrix of Z by definition is expectation of Z minus expectation of Z into Z minus expectation Z transpose.

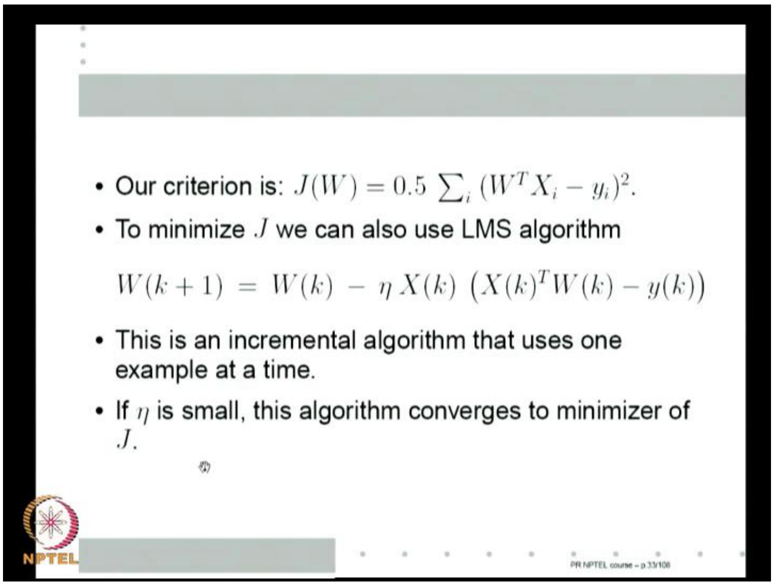
I hope you all of you remember that we are by our notation all vectors are column vectors. So, both Z and Y, we think of as column vectors. So, Z minus expectation Z is a column vector, this is a row vector. So, this is an auto product this gives my matrix. So, this is the definition of covariance matrix under vectors being column vectors. Now, I can substitute Z to be B Y and expected value of Z to be B expected value of Y, because

B is a constant. So, Z minus expected value of Z will be B into Y minus expected value of Y. So, if I substitute that I get B into Y minus expected value of Y because of the transport, transpose becomes Y minus expect value of Y transpose B transpose. Now, since B is constant expectation can go inside, then it becomes B expectation of Y minus expectation Y into Y minus expectation Y transpose, that is nothing but the covariance matrix of Y.

So, this becomes B sigma Y B transpose. So, if Z is equal to B Y, then covariance matrix of Z is given by B sigma Y B transpose. The sigma Y is the covariance matrix of Y. Now, we have, we have given that W star is some matrix into Y that particular matrix happens to be A transpose A whole inverse A transpose. So, know you take that to be B, and plug it in this formula. This is what we will get as our, this is what we will get as our covariance matrix of W sigma W is equal to B is a transpose A whole inverse A transpose sigma square I. Now, I need transpose of this that will be A into A transpose A whole inverse, realize that A transpose A is symmetric, and inverse transpose is same as transpose inverse.

So, now this is identity, so this drops off. So, I have got A transpose A into A transpose A whole inverse that becomes identity. So, ultimately I get the covariance matrix are W to be sigma square times A transpose A whole inverse. So, this gives us the covariance matrix to the least squares estimate. Essentially, W has components W 0 W 1 W n. So, in this matrix, the diagonal elements will be the covariances of the, the variance of the individual components, and the half diagonal elements will be covariance of the different pairs of components of W. So, this gives us all the information that we need about the variance in the least squares estimates. So, this is a useful formula to calculate the variance in the least squares estimate.

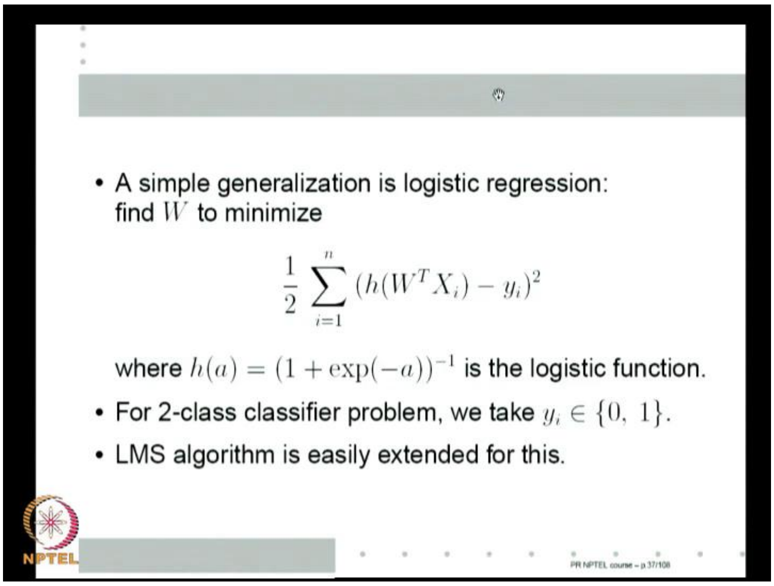
(Refer Slide Time: 18:13)



So, let us look at a come back to our criterion. This is our criterion least squares criterion, as you already seen instead of using this W star formula, we can also minimize it using the LMS algorithm which is nothing but the gradient descent. But, implemented in a incremental manner. So, at each iteration k, you pick up pick one of the training samples. Let us call them X k Y k, then you update it only based on that samples error. So, the update turns out to be W k plus 1 is equal to W k minus eta times X k into X k transpose W k minus Y k. This is actually the error multiply by X k. This is the LMS algorithm as we saw earlier.

This is an incremental algorithm use one example at a time, and as we already seen at least stated, if eta is small then the algorithm converges to a minimizer of J. So, this is the LMS algorithm. I am as we discussed earlier, there are some very interesting properties of this it is the, it is also classical algorithm. And one other reason for its attractiveness is that, it is easily generalize to some slightly mode general models than linear models.

(Refer Slide Time: 19:16)

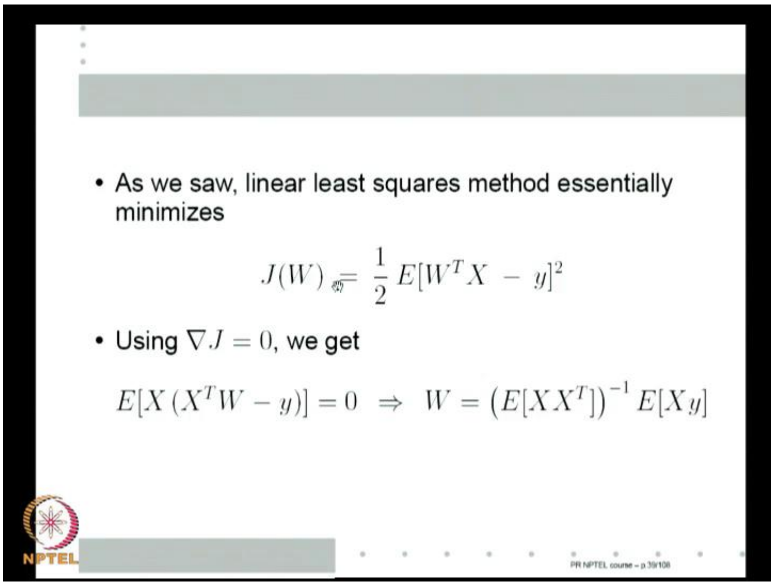


So, we considered one generalization earlier, which the logistic regression, what is logistic regression? Do instead of using W transpose X i at the model, we use h of W transpose X i at the model, where h is the h X is 1 by 1 plus exponential minus X is the logistic function. As we seen, this is a very good model for posterior probability, and we know the least squares. When we are doing expected value of f X minus Y whole square, the best function f is the conditional expectation of Y given X. Hence, in a in a classification context, this will allow us to estimate the posterior probability compared to linear function W transpose X h of W transpose X. The logistic function is a much better model for posterior probability.

So, for example we seen last class, that if the class conditional densities are Gaussian with the equal covariance, then the, the posterior probability actually is given by h of W transpose X. So, in such cases, this is a very nice model does not really linear model because of this h function here, but as we have seen, the LMS algorithm is easily extended for this. So, essentially for two class problem, we take y i to be 0 or 1. So, that h, h of W transpose X will give us the posterior probability estimate.

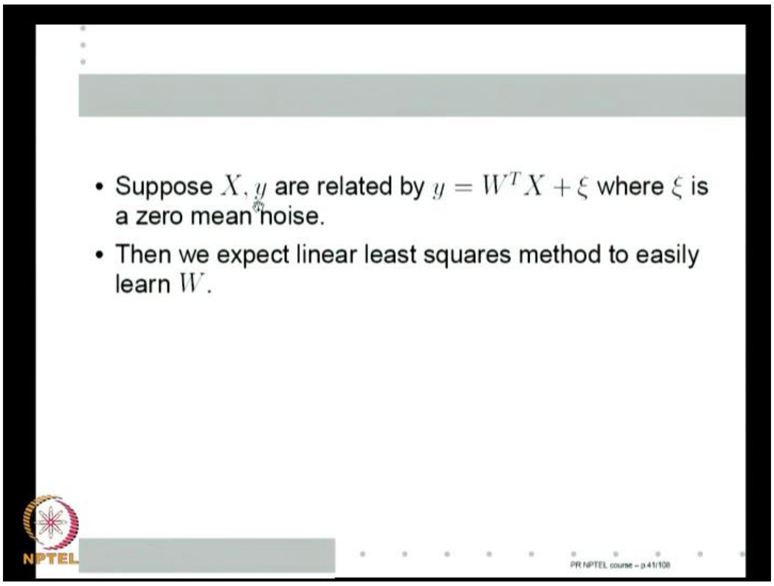
And, when we while, we with, with more difficult to put the projection thing into this frame work, if you are using LMS, we essentially need the gradient of this. The gradient of this is easy, because I get this error term anyway like in the previous LMS case, I get the error term instead of X k transpose W k minus y k, I get h of X k transpose W k minus y k, because the 2 cancels. Then I need the derivative of this term. So, there will give me one h prime term and then the X i X k term.

(Refer Slide Time: 21:52)



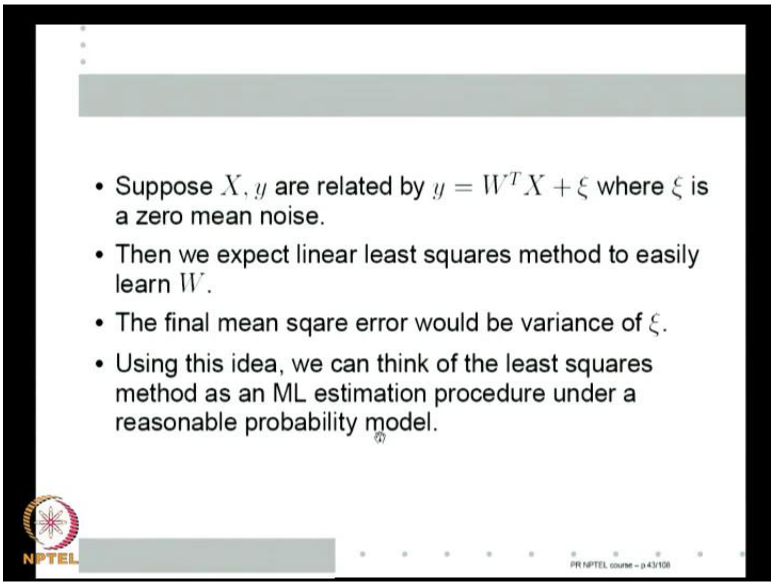
So, essentially if I use LMS for this, it will be same as this except that in this update, I will have one more term that is just h prime X k transpose W k. So, in that sense, LMS algorithm is easily extended for this and that is the Logistic Regression that we considered last class. Moving on, as you seen the least squares method is actually a good approximation to minimizing actual mean square error, that is why I put expectation here, and we seen that this is the solution for it.

(Refer Slide Time: 22:12)



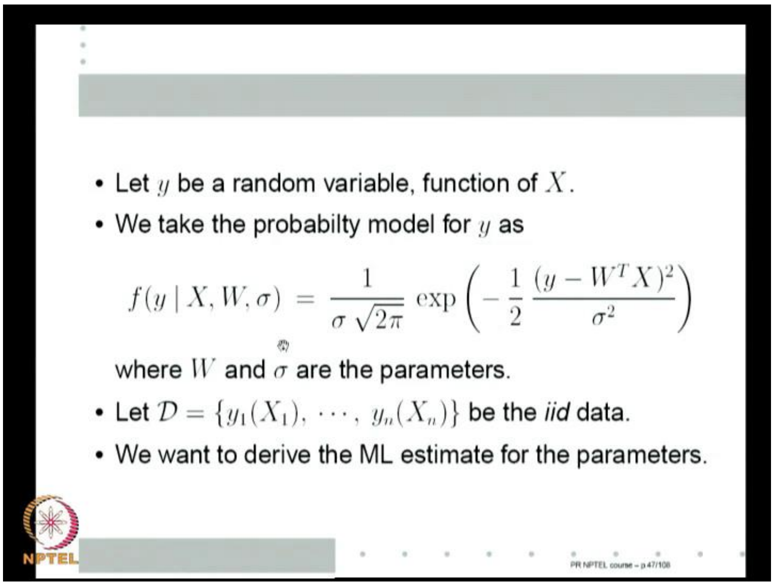
Now, we are if we are actually minimizing expectation like this. Let us suppose, X and y are related actually related by y is equal to W transpose X plus x i, where x i is a 0 mean noise. If for each X y is given by this, then in this expectation essentially they are able to transpose X will cancel. And, I will get only x i i here for each I, so what I should get ultimately. So, if I use the same W with which X and y are related. Then for that W J of W turns out to be nothing more than expected value of x i square. If x i is a 0 mean noise expect value of x i square is the variance

(Refer Slide Time: 22:53)



So, what it tells us is that if actually X and y are related by this, then we can expect the linear least squares method to true learn W very easily, and further that the final mean square error would be the variance of xi. This is if you actually did the expectation. What this means is the following, we can give a probabilistic interpretation to the linear Least Squares, by thinking of it as a, as a method as a, as a estimation method. Specifically, as a maximum likelihood estimation procedure for the parameters in a probability model governing y.

(Refer Slide Time: 23:56)



So, what we will do is we will, we will think of a reasonable probability model, to capture this idea and then show that the Least Squares Method is nothing but an M L estimate of the parameters of that model. So, let us look at the probability model. So, the idea is y is random variable, which a function of X, that is what we want to model. So, we take the model as follows.

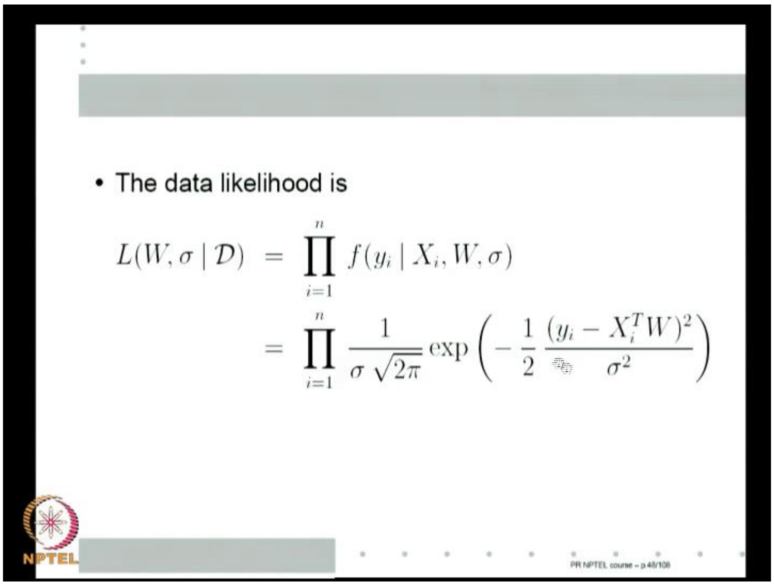
So, this is I am, we have been using f for many different things, during when we did M L and Bayesian estimation. I said f whenever needed stands for density function of any random variable, that you want conditional anything else, only when necessary we put any subscript, superscript so from context, you should know that this is a density function. Earlier, we are using f as our model function. So, because we are back to in estimation context this time, this f represents a density function.

So, this is the density f y. These are the model parameters of the model W and sigma in addition y depends on X, as we thinking of X as fixed, X is not part of the probability model. So, what we are saying is the density of y, with the parameters W and sigma, and an X, and a given X is Gaussian, whose mean is W transpose X and whose variance is sigma square.

So, essentially what it means is that we are modeling y as W transpose X plus a zero mean Gaussian, Gaussian noise whose variance is sigma square. So, that is same as saying, the probability model for y condition on X, and W on sigma is Gaussian with mean W transpose X, and variance sigma square, so W and sigma the parameters. Now, if I got many IID samples from this model.

So, sample from this model will be I put an X, I get a y. So, we will think of the models as y 1 at X 1, y 2 at X 2, y n at X n. This is the IID data that I have, and using this IID data I want to estimate the model parameters W and sigma more importantly W for us. So, I want to estimate the parameters W in the maximum likelihood convection. So, what does that means? Given, this IID data I will calculate the likelihood function, and find the W that maximizes the likelihood function. That is what we have done. We have done this M L estimate for many different models, earlier in this course.

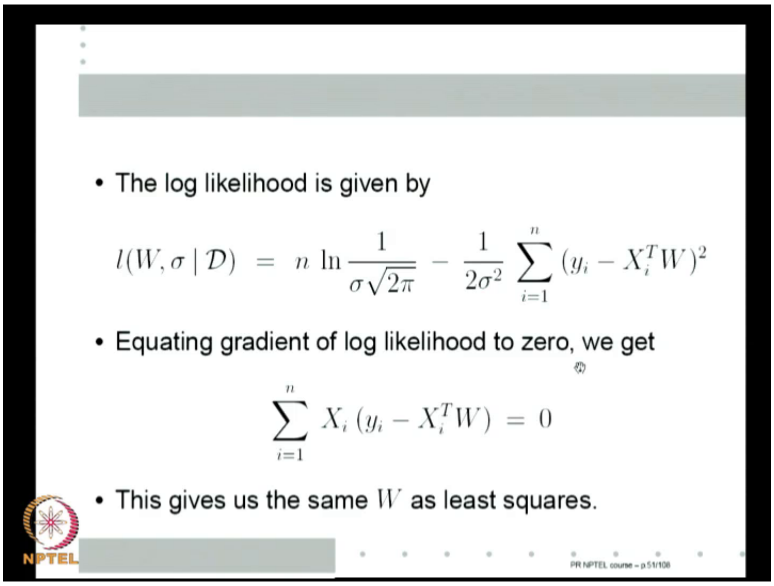
(Refer Slide Time: 26:23)



So, let us do the same thing again with this probability model, we want to find the M L estimate for W, and may be also for sigma. So, what is the data likelihood? The likelihood function, which is a function of the parameters condition on the data, is product i is equal to 1 to n f of y i condition on X i W sigma. This is the, the probability model we are using. So, inside the product it becomes 1 by sigma root 2 pi exponential minus half, because this X i y i is y i minus X i transpose W whole square by sigma square.

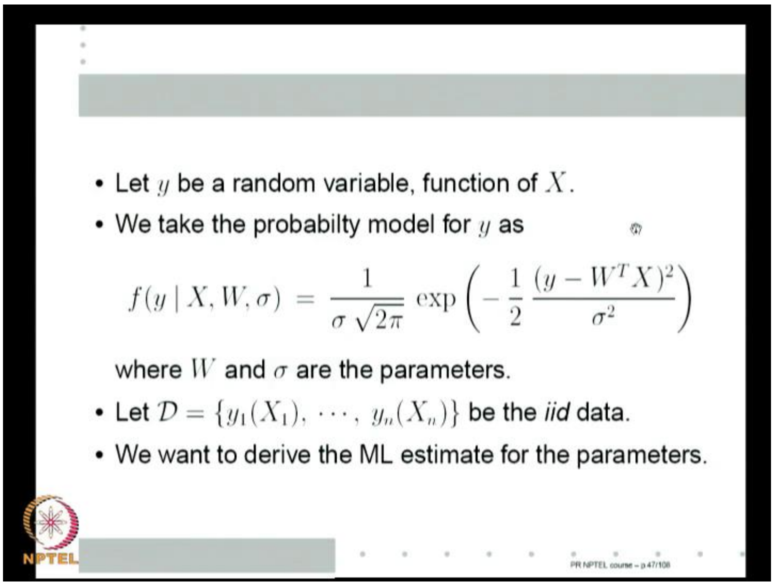
So, this is my data likelihood. So, we want to maximize this as we know in the ML context, we often take the log likelihood, and maximize that. So, let us take the log likelihood. So, if I take the logarithm, this becomes sum over i. So, this term is not dependent on i, the first term. So, that becomes n times sigma root 1 by sigma root 2 pi. So, its n times l n 1 by sigma root 2 pi, and this sum goes inside. So, I get exponential sum, so that is the second term. The exponential drops off because of l n. So, I get minus 1 by 2 sigma square y i minus X i transpose W whole square. That is my log likelihood. I want to maximize this with respect to parameter. Let us say in particular with respect to W. So, to maximize it with respect to W, we take the gradient with respect to W, and equate it to 0.

(Refer Slide Time: 27:41)



So, if we equate the gradient of the log likelihood to 0, the gradient this term, of course is not function of W. The gradient of this is back to what we are getting earlier. So, forget the sigma square because equate it to 0, the 2 anyway cancels. So, we get i is equal to 1 to n X i into y i minus X i transpose W is equal to 0. This is exactly the set of equations, we have for solving our linear least Squares estimate. So, this gives us the same W, as the linear least squares estimate.

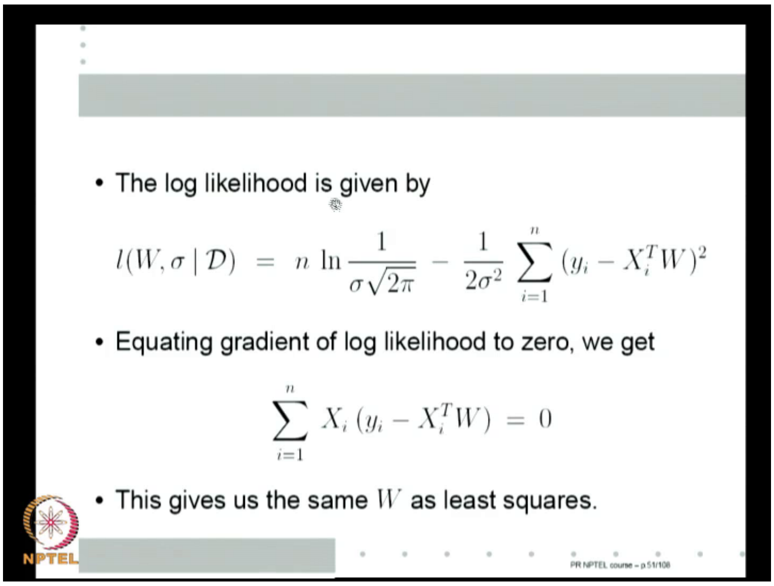
(Refer Slide Time: 23:56)



So, essentially I can think of the linear least squares estimate or linear least square solution W. As, as a maximum likelihood estimate of the parameters of this model, for y and this model, for y is very nice, we are essentially assuming, that the relationship in X and y is random. But in such a way that, y can be written as a linear function of X plus additive Gaussian noise.

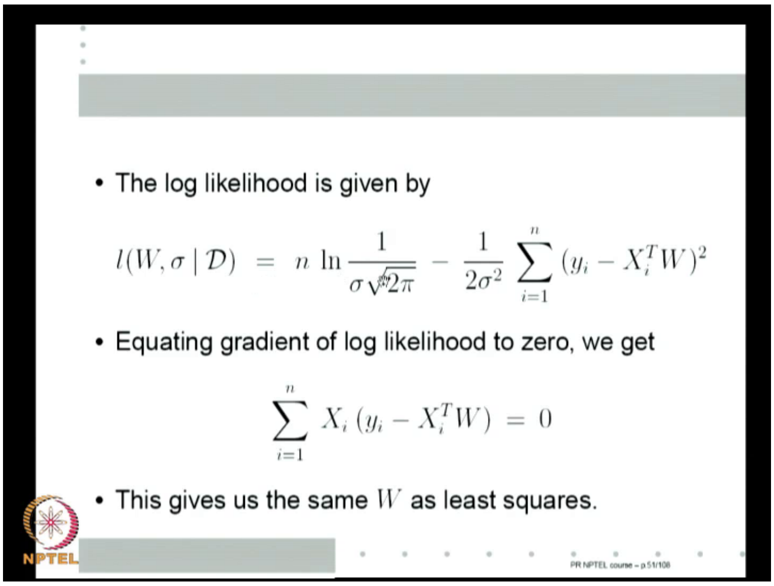
So, if I, if I think that, that is the underlying relation between y and X, y is a linear function of X plus additive Gaussian noise, then my least square solution is nothing but the M L estimate of model of that parameter. Actually, my my probability model for y has two parameters, W and sigma as we seen. The M L estimate W is same as what I get out of least square.

(Refer Slide Time: 29:27)



So, you can also ask what will be the ML estimate for sigma. So, to find the ML estimate for sigma, we have to maximize the log likelihood again with respect to sigma. So, I have to differentiate this with respect to sigma.

(Refer Slide Time: 27:41)



So, let us differentiate this with respect to sigma. This is what we get, this time is minus n l n sigma root 2 pi, that I can write as minus n l n sigma plus minus n l n root 2 pi. So, that is one term that is dependent on sigma the first and this term is also dependent on sigma.

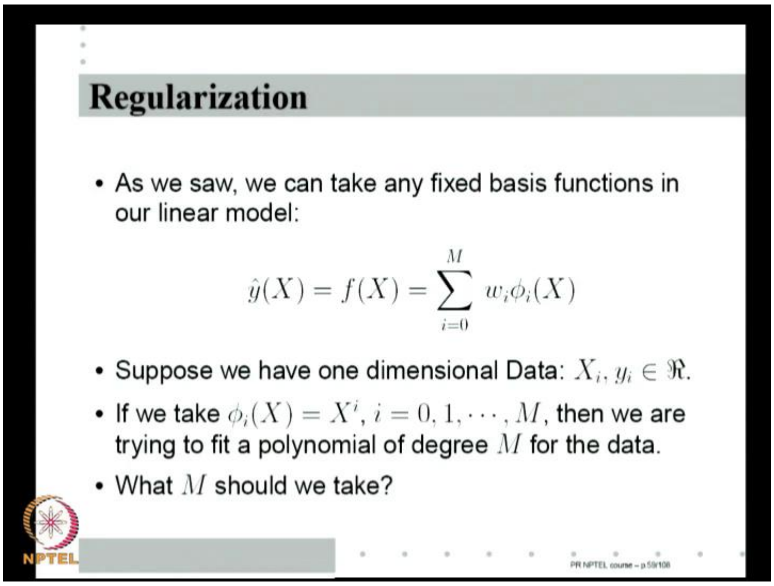
So, this will give me minus n by sigma because minus n l n sigma differentiates this minus n by sigma. This is a constant. This entire sum and this term will give me minus 2 by 2 sigma cube. So, that is my derivative minus n by sigma minus 2 by minus 2 sigma cube into this. So, this minus, this minus cancels take this n by sigma on that side, and bring n this side, and sigma cube that side.

And that gives me sigma square is 1 by n i is equal to 1 to n y i minus X i transpose W whole square, of course I have to simultaneously solve del l by del sigma is equal to 0 on del l by del W is equal to 0. So, the final M L estimate for sigma will be given by this equation, where this W is the W that satisfies del l by del W is equal to 0, which is my least square solution.

So, then X i transpose W here, is the actual fitted least square solution and in that sense this is nothing but the final average squared error that I get on the on the data. Because if this is the final least square solution. This is the square of the final error I get on the fitted model for the i th sample. So, this is the average error I get with the i th sample. So, my M L estimate for sigma square is the residual average error as we have seen in our sigma estimate earlier.

So, essentially if y is actually a linear function of X plus additive Gaussian noise, then linear least squares is the, is the best thing we can do. Then the W the linear least square gives us, is the M L estimate. For that W under that assumed model and the M L estimate for sigma square the noise corrupting the observations y i is well captured by the final average square error in the fitted model. So, this is one way in which we can look at linear least squares as a M L estimation procedure under a simple additive noise models. That is why very often we talk about this method, as also as linear least squares estimate.

(Refer Slide Time: 32:41)



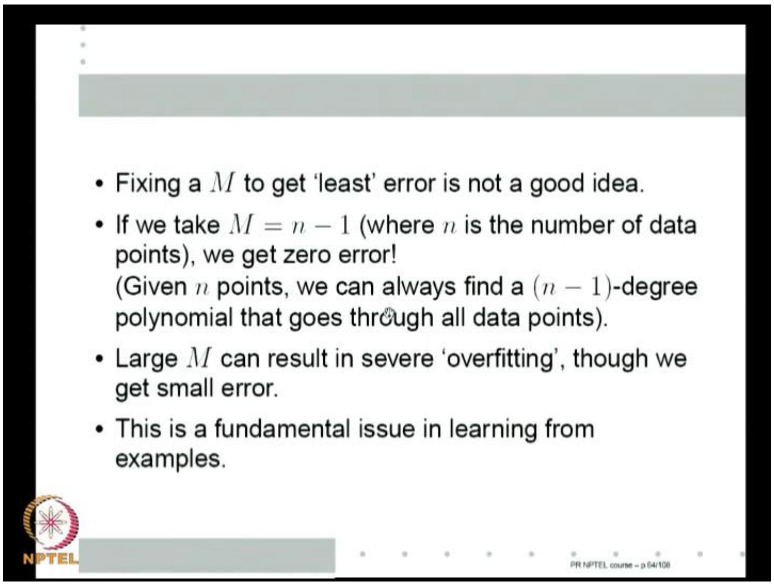
Let us look at another aspect of the, the linear least squares method, as we said in the beginning of this class, all these techniques are applicable for more general models, which are essentially used fixed basis functions. So, I can take my linear model to be sum over i equal to 0 to M w i phi i X. Earlier, we were calling it d prime let us call it M. Now, for some reason I will give you.

Now, let us go back to our old example to illustrate this. Let us say we have one dimensional data that is X i y i both belong to R, then let us take phi i X to be X power i, then this becomes w 0 plus w 1 X plus w 2 X square plus w 3 X cube and so on. So, this will be an m th degree polynomial expression in X say, essentially if I use that kind of phi i and I have one dimensional data, what I am doing is I am trying to fit a polynomial of degree M for the data. And this is the standard curve fitting problem though many of many of you may have only done it for straight lines.

In general given data X i y i in R, I can fit a polynomial of degree M. By simply using this method earlier, of course we looked at it as a nice generalization of linear, nice way of illustrating the generality in what we called linear models, but let us say we actually want to use it for fitting a polynomial. Now, I am given only data X i y i. So, this M is my choice, this M is the choice of the learning algorithm or the designer of the learning algorithm.

So, a question is if I want to use it what M should I take? How do I decide what M should I take? So, I have given points X i y i. Should I fit a straight line to them? Should I put a fit, a quadratic curve through them? Should I fit a cubic curve through them? After all, I can chose any M, and then write this expression. And my linear least squares method gives me all the W's that is the best. So, I can have a best fit straight line. I can have a best fit quadratic function, and so on, which one should I use?

(Refer Slide Time: 35:15)



Now, this though it may look. Look this may look deceptively simple say very, very deep issue. So, deep issue because I cannot fix M to get low error that is to, that is to say suppose, I fitted the best straight line and let us say the final residual mean square error is some number call it is a 2 point 1.Then, let us say I using the same method. I fitted the best quadratic line and let us say the final residual error 1 point 9 5. Does it really mean that the data has a quadratic relationship rather than linear relationship? How do I know?

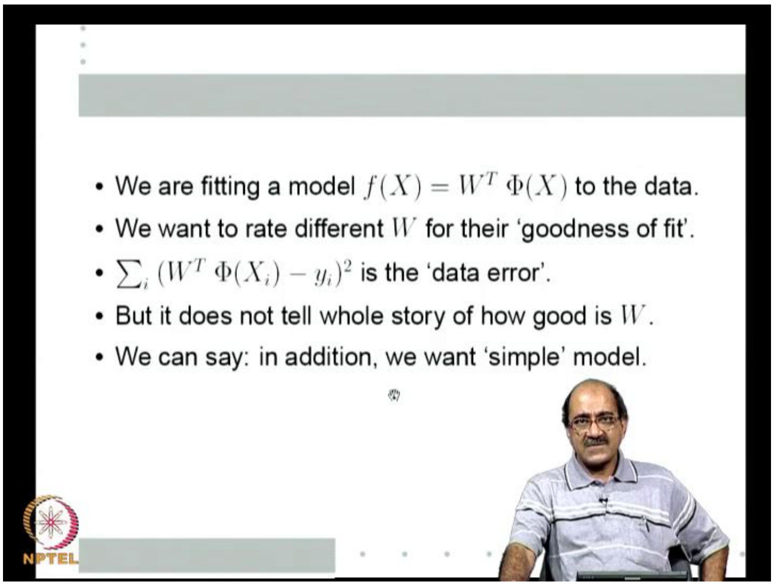
The first issue that all of us can immediately see, is that fixing M to get least error is not a good idea. Why it is not a good idea, not a good idea? Because, if I take M to be n minus 1, then we always get 0 error, why do we get 0 error? If I take n to be n minus 1, essentially if you give me n points I can always find a n minus 1 degree polynomial, that goes through all the data points give me any two points, there will be an straight line give me any three points, I can find a quadratics on which they will be and so on.

So, if I take M to be n minus 1. I will always get 0 error, but that is ridiculous is like saying if I have ten points I will put a 10 degree polynomial or 9 degree polynomial through them, which of course would have a, would be a perfect fit. But, anybody who is played around with exponential data points knows that is very, very unlikely to be a good fit. It is it is essentially highly over treatment. It is like determining a straight taking two points in a, in an experiment, and say that this shows me that the relationship is linear, which is, which is ridiculous. Two points will not tell you that the relationship is linear.

So, generalizing this if I increase M, I may get lower and lower error, but does that mean that I am getting better and better fit. That is not true. Large M, of course gives me small error, but it results in what is called over fitting. So, because y i's are most probably noise corrupted. I would be fitting the noise rather than the trend in y i, if I increase M. So, large, M results in over fitting, though we get small error. Hence, we cannot really fix M based on the error we are getting.

Now, this is a very fundamental issue in learning from examples. We will come back to this question and in its most general version, the question is not even answerable, but is this is the first? First time in this course, we will come into this question. So, it is good to pond around this at least a little bit. Basically, in this scenario I cannot fix the degree of polynomial that I want to fit, based on which, degree polynomial gives me low error because I can get a ridiculous polynomial that gives me 0 error.

(Refer Slide Time: 38:19)



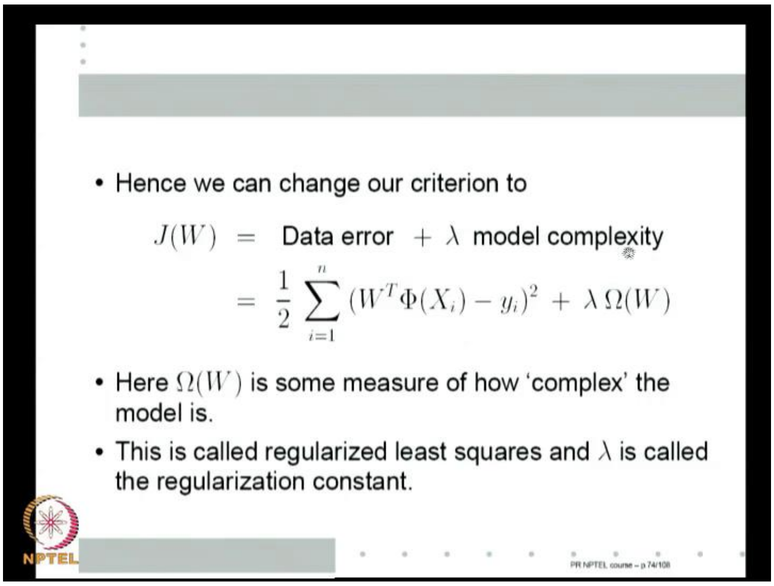
So, we can actually generalize this example. We are fitting a model f X is equal to W transpose capital phi of X on the data. And we want to rate different W for the goodness of fit. What we know now is this is the data error on the sample I have W transpose phi X i, is what my model will say y i is what the actual targets says and square this, and sum over i, that is the data error that we can call the data error.

We of course, have been just trying to minimize this data error. We know that, y is noisy and hence we do not want to exactly match, but we are still trying to minimize this data error. But, what we now saw is said it does not tell the whole story of how good W is? We can get in this kind of error row by simply putting more and more basis function. We can easily get 0 data error, but that does not mean that I am actually learning the underlying functional relationship, so what else can I ask?

So, somehow we do not want to fit too complicated model, it is like if I have 9 experimental points, and I will show you that a straight line as a, is a fairly good fit. May be I am inclined to believe that the relationship is linear, but if you tell me is that I can put eighth degree polynomial. Obviously, it is very difficult to believe. So, somehow we want a good error with a simple model, whatever that simple model means.

So, when we are asking how good a fitted model is, we should not blindly go only by the data error, but we should also ask. Are we fitting a very complicated model to get low data error? In this class, we look at it at a very simple level to just introduce what I called regularized least squares. We will come back to this question at least, at least at a preliminarily level. We will, we will discuss this question in more detail, when we take up our discussed terms statistical learning theory.

(Refer Slide Time: 40:27)



So, based on just what have we said just now, we one can say we want to change your criterion. The old J W not to just the error term, but in addition there is one more term, that somehow tells me the model complexity, and I want a W that is simultaneously minimizes both. Now, because I do not know how to simultaneously minimize both, I just added data error and model complexity. I cannot directly add them because it is like adding apples and oranges. This might be in one units, units in a, in a, in a general sense. Numerically, this can be in one range because I do not know, on what scale I want to measure model complexity. And I also do not know how much a model complexity I want to trade for how much of data error.

So, we just put some arbitrary constant here called lambda, which is kind of an exchange rate between my model complexity and data error trade off. So, in particular, in this in this linear model case, this is my data error half W transpose phi X M as the whole square. I have some model complexity term currently. Let us just called it capital omega of W, but chose what function W would be a nice model complexity term.

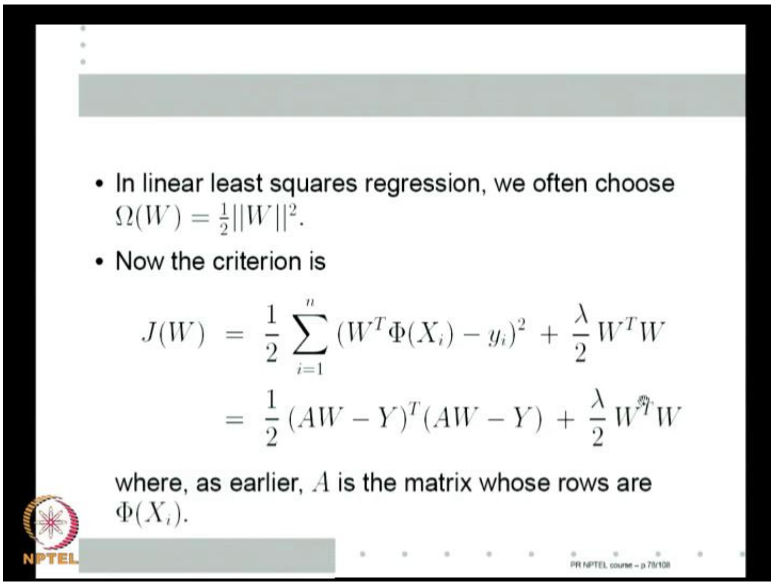
And we use this exchange rate lambda, so to say to decide how to add them, and then find a W to minimize it. This omega of W is some measure of how complex the model is. I put that complex in codes. So, it is, it is not easy to define what is complexity of model, but there are various measures in this class. We will just consider one of them without giving much reasons, but we will come back later on, on this issue. Now, this kind of a method is called regularized least squares, and the lambda is called a regularization parameter, regularization constant, omega is called the regularization function, and lambda is called the regularization constant.

So, instead of just minimizing data error and hence artificially getting low data error, and high confidence on a model that is unnecessarily complicated. Hence, it is not really good at predicting. That is the problem that happens, if I chose too many terms in my file. I am sure to say in some sense take any more complex model, then can be justified. But, I may not know because I will get very low data error and hence I am very confident about my model.

So, to avoid that kind of an error, that kind of over fitting error, we add a regularizing term, so this omega is called a regularization function; lambda is called the regularization constant. These kinds of things will be coming with us again and again in this course. We will, we will look at them in more detail, when we consider some other techniques of classification regression. This is the simplest and the first experience for you with this, so called regularization the idea is that not just data error.

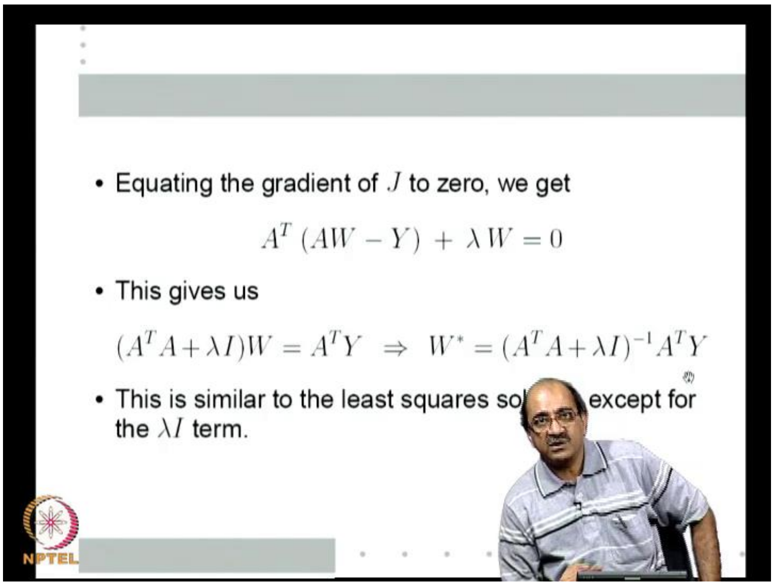
But, some level of model complexity should also be taken in to account. In linear least squares, we often choose the model complexity term to be norm W square. Later on, when we look at SVM's and so on, I will come back and tell you why this could be a good model complexity term? Now, let just take it to be a good model complexity term, then the criterion becomes J W is this plus lambda by 2 W transpose W.

(Refer Slide Time: 43:34)



Now, I want a W that minimizes this whole thing not just this, but the whole thing. Now, once again this data error, of course can be rewritten in the matrix form like we did earlier, A W minus Y transpose A W minus Y, where what will be, A will be the matrix whose rows are phi of X i capital phi of X, is the i th row of A. So, using that matrix I can always write this, this squared error as A W minus Y transpose A W minus Y. I just got one extra term, now finding gradient of this term is very simple. So, we can once again find the gradient of J, equate it to 0 to find our best W.

(Refer Slide Time: 44:52)



So, let us do that, so if we equate the gradient of J to 0, first term will be same A transpose into A W minus Y. Second term is just lambda by 2 W transpose W, its gradient is nothing, but lambda W. That is what we get, so if we simplify this I get a transpose A plus lambda I whole times W is equal to A transpose Y. Earlier, this lambda term was not there. It is simply A transpose A W is equal to A transpose Y, that is how we got W is A transpose A whole inverse into A transpose Y.

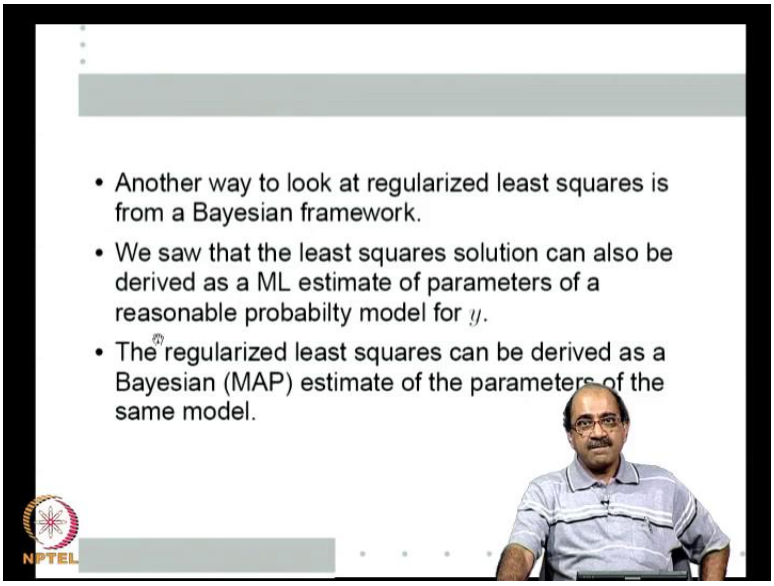
Now, I am adding this lambda. So, I get a A transpose A plus lambda I whole times W. So, my optimal W now transfers to be A transpose A plus lambda I whole inverse into A transpose Y. So, this exactly same as the earlier least squares except for this lambda I term. At this point, one simple thing you can notice, if some of you, if you have studied some optimization algorithms, you may have (( )) to add lambda times identity matrix to some other matrix.

Before taking inverse, many so called quasi neutron algorithms, if some of you know about them are based on this. A simple way of looking at this is, if A transpose A is not invertible or even if it is invertible, it has very poor condition number, then adding lambda times I will improve the condition number of A transpose A. So, in that sense the regularization is making this solution behave more smoothly and better.

So, with poor condition number what it means is even a small differences in your targets or in your examples can make large difference to W's. By adding lambda I, we can improve the condition number of this matrix which means the, the final solution obtained is somewhat robust to errors made in Y that is essentially what we want for relying only on the data error. We will might be giving too much importance to some noisy values of Y for fitting, where as using this regularization. I improve the condition number, so Y is much more robust. So, small perturbation W star is much more robust to small perturbations.

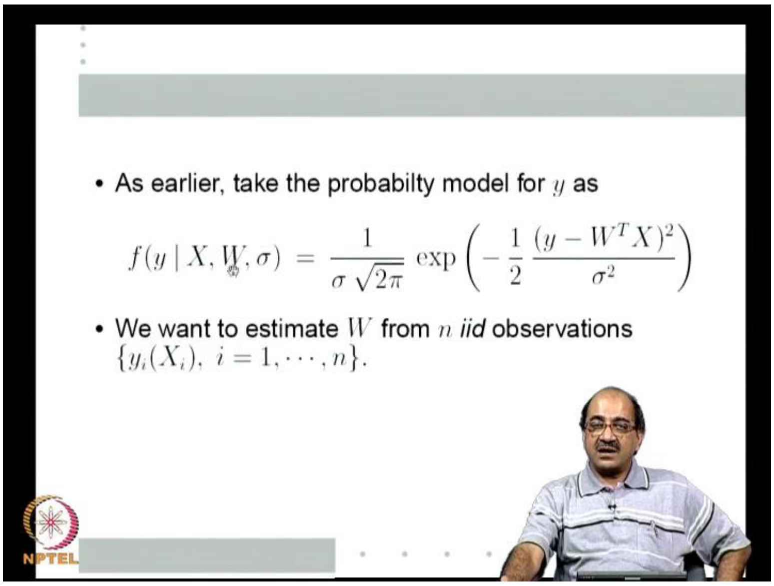
So, this is one way of looking at regularization. So, this is called the regularized least square solution. So, when you want to regularize the, the regularized least square means the same least squares thing, where the, in the criterion we add lambda time W transpose W as the regularizing term. Then this becomes the solution. Another way of looking at regularized least squares is from a Bayesian framework. We just now saw that the original least squares solution can be obtained as a ML estimate of the parameters of a reasonable probability model for y.

(Refer Slide Time: 47:15)



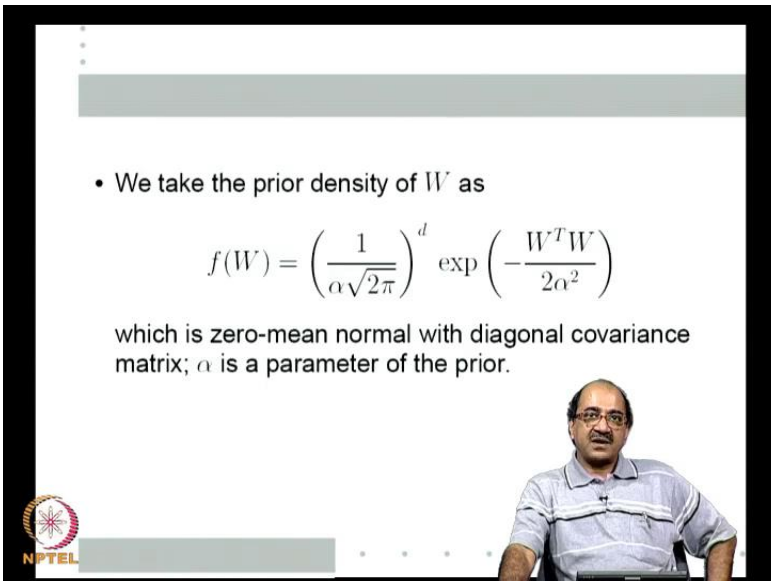
As it turns out the regularized least square solution, turns out to a Bayesian particularly specifically map estimate of the parameters of the same probability model. Let us quickly derive this, as earlier take the probability model.

(Refer Slide Time: 47:47)



For y, this is the same model that we used earlier, sigma root 2 pi exponential minus half y minus W transpose X sigma square W, and sigma are the parameters of the model. We want to estimate them. We are given IID data y 1 X 1, y 2 X 2, y n X n and we want to estimate. The only difference is that earlier we did an ML estimate, and now we want to do a Bayesian estimate. Recall from our earlier lectures in this course, that when you want to do a Bayesian estimate, we need a prior density on the parameters, prior density n W, because W is what we want to estimate.

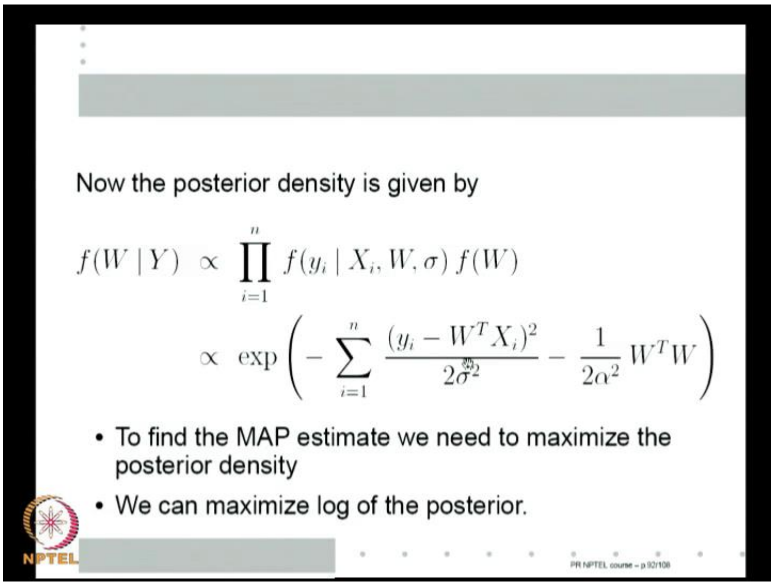
(Refer Slide Time: 48:54)



Now, W is the essentially, the parameter effecting the mean of a Gaussian distribution, Gaussian densities my probability conditional, my data model is a Gaussian density and my unknown parameter is what effects the mean. So, the conjugate prior would itself be a Gaussian. So, we will choose a Gaussian prior for W. So, we choose the prior n W as 1 by alpha root 2 pi whole to the power d exponential minus W transpose W by 2 alpha square. What is this W is I, I taken W to be d-dimensional error. Actually, I should have taken it to be d plus one dimensional. I am sorry, but really does not matter whether it is augmented or not. We just simply take it to be d dimensional for now. So, then the prior we are taking is a 0 mean normal distribution, which has a diagonal covariance matrix with all components having the same variance alpha squares.

So, different components of W have no covariance, and all components of W have the same variance, and it is 0 mean. So, we just choosing a 0 mean normal with the diagonal covariance matrix, and the variance alpha square is a parameter of the prior. As we seen the each prior density will have its own parameters. Sometimes in the Bayesian jargon, they are called the hyper parameters. So, we do not know what parameter to choose for alpha square prior choice, choice of the actual prior density is part of the art of Bayesian estimation, but anyway let us choose this as the prior.

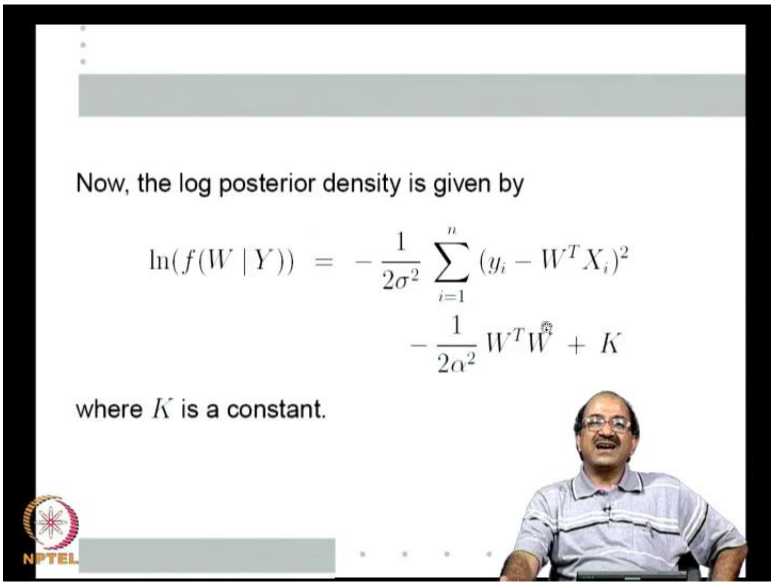
(Refer Slide Time: 50:17)



So, then in the Bayesian estimation, I have to calculate the posterior and because I want a MAP estimate, I have to find the maximum of the posterior. So, let us calculate the posterior. The posterior of Y given the data because capital Y is all the Y i and y i's are essentially the random part of the datas. So, that is our data is, this is the conditional the f y i given X i W sigma into the prior product over i is equal to 1 to n over the an IID observations proportional. Because we do not put the denominator, which in turn is proportional to see this is normal, y i given, this is y i minus W transpose X i whole square by 2 sigma square, this is also normal.

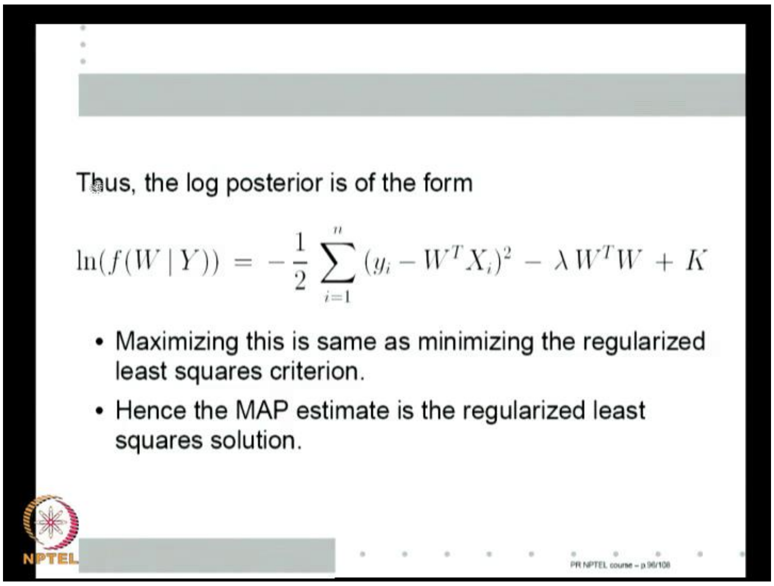
So, the exponential term is 1 by 2 alpha square W transpose W. There are some constant outside 1 by sigma root 2 pi 1 by alpha root 2 pi to the power d n all those things. So, forgetting about the constants, now this proportion to this, so we need to, to find MAP estimate, we need to maximize the posterior. So, instead of maximizing the posterior, we can maximize the log of the posterior. So, let us try, and maximize the log of the posterior because if I take log the exponential will go away, this proportional constant simply means I can write the posterior, to be some K times this exponential, if I take log, I get some log K term as a, as a additive constant.

(Refer Slide Time: 51:57)



So, with that this is my log posterior. So, will give me whatever inside the exponent. These two terms plus some constant, so minus 1 by 2 sigma square summation i is equal to 1 to n y i minus W transpose X i whole square minus 1 by 2 alpha square W transpose W. This is what is inside the exponent plus some constant K. This is the log posterior density, and this is what I want to maximize.

(Refer Slide Time: 52:55)



So, essentially of course this alpha square is some hyper parameter, I do not know it is value sigma square is also part of the model. I do not know it is value. So, bottling up all those unknown constants, we can rewrite this form as follows, can write this as half i is equal to 1 to n y i minus W transpose X i whole square minus some lambda times W transpose W.

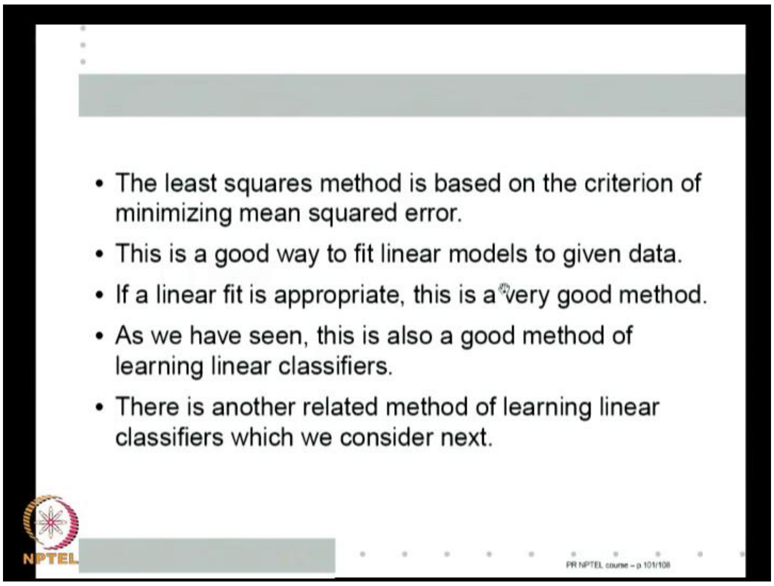
All I am doing is I am multiplying this whole thing by sigma square, and calling sigma square by 2 alpha square as lambda, K really does not matter it is some constant. It does not come into our maximization. So, I can write this as minus half i is equal to 1 to n y i minus W transpose X i whole square minus lambda times W transpose W plus some constant. This is what I want to maximize.

So, if you want to maximize this, this K does not make any difference a constant. I am, I have to maximize this both terms, I put minus here. So, it is same as minimizing if I put both terms plus. So, maximizing the log posterior will be same as minimizing the regularized least square. That is my regularized least squares criterion function, half i is equal to 1 to n y i minus W transpose X i whole square plus lambda times W transpose W.

So, maximizing the log posterior is same as minimizing the regularized least squares criterion, which essentially means that the MAP estimate is the regularized least square solution. So, just like we shown that for this reasonable probability model namely the targets y i, there are related to X i by y i W transpose X i plus a 0 mean additive Gaussian noise, then the ML estimate corresponds to the, corresponds to the regularly or normal least squares, and the Bayesian estimate corresponds to regularized least square. And that is also on hinge side, not very surprising because ML estimate is good, when we have large data, large relative to the dimension of the W vector.

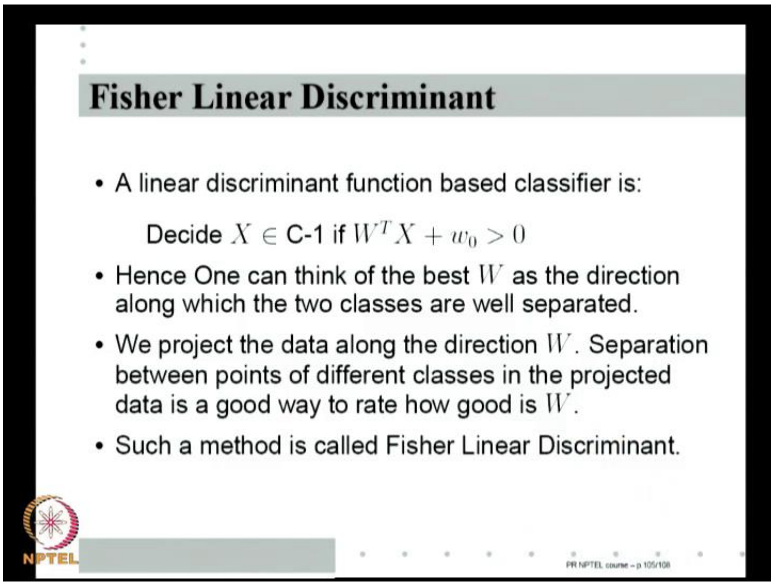
So, if I have large data then this problem of over fitting does not come. Over fitting comes if my degree of the polynomial is much higher compared to is, is large relative to the number of data points I have, but my number of data points are very large, then over fitting is not a not an issue. So, ML estimate is good enough, and on the other hand if my number of data points is small as we have seen, when we did estimation Bayesian estimation performs well. So, the regularized least squares would is particularly needed, if, if my model complexity is large as the number of data point is small both are essentially the same.

(Refer Slide Time: 55:13)



So, to sum up the standard least squares, is the ML estimate under, under this nice probability model on the regularized least squares, is the Bayesian MAP estimate. So, let us, let us sum this up. So, least squares method is based on the criterion of minimizing mean square error. It is a good way to fit linear models to given data, if through linear fit is appropriate. This is very good method as we seen if I can assume that the X and y are related by y is equal to W transpose X plus additive Gaussian noise, then you know essentially, if it least squares is an M L estimate, and regularized least square is a Bayesian estimate of the, of the model. This is also a good method of learning linear classifiers.

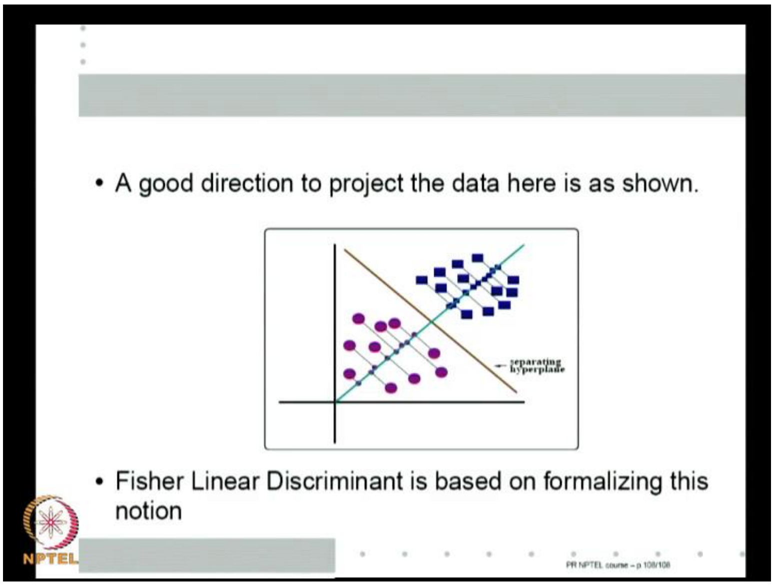
(Refer Slide Time: 55:58)



Let us now, we will move on to another related method of learning linear classifier that we consider next I just give you a brief overview of this. A discriminant function based classifier, we will, we will move it first. We will look at it again next class that discriminant function based classifier is to say X belongs to class 1, if W transpose X plus W not greater than 0, I can think a W transpose X as projecting X into the direction W. So, then essentially the feature vector no matter what its dimension is becomes a 1 dimensional feature vector.

So, one way of asking because W 0 is just a threshold, I am asking, which is a good direction along which the two classes are well separate? I am asking project all the things along W. So, and put a W 0 point somewhere in that direction all points on one side are one class, all points in the other side are another class. One, one can think of learning a linear discriminant function, as learning the best W direction along which to project different data. So, the separation between points of different classes is the projected data is large.

(Refer Slide Time: 57:08)



So, one way of telling what is a good W, is to look at projections of data along direction W and if the separation between different classes is good, then that is a good direction. Such a method is called Fisher linear discriminant. Let us look at a small example, let us say, this is the two class problem, so that will be the separating hyper plane. Now, if I project that on to X axis, they overlap if I project data onto Y axis, they overlap. But, if you project data along some line like this, then I can make one dimensional data point. For projecting along this direction, then all the data is well separated, if I project onto X axis and Y axis, then are well separated, but along this direction.

If you project along that one dimensional subspace, the two classes are well separated and that is the W direction, which I project. As you know, W is the normal to the separating hyper plane. So, that will be the separating hyper plane. So, a best way to ask where is the separating hyper plane is to ask, which is the direction along which I should project? This is the basic idea of Fisher linear discriminant. A Fisher linear discriminant is just a way of formalizing this notion. So, in the next class we look at the Fisher linear discriminant in more detail.

Thank you.