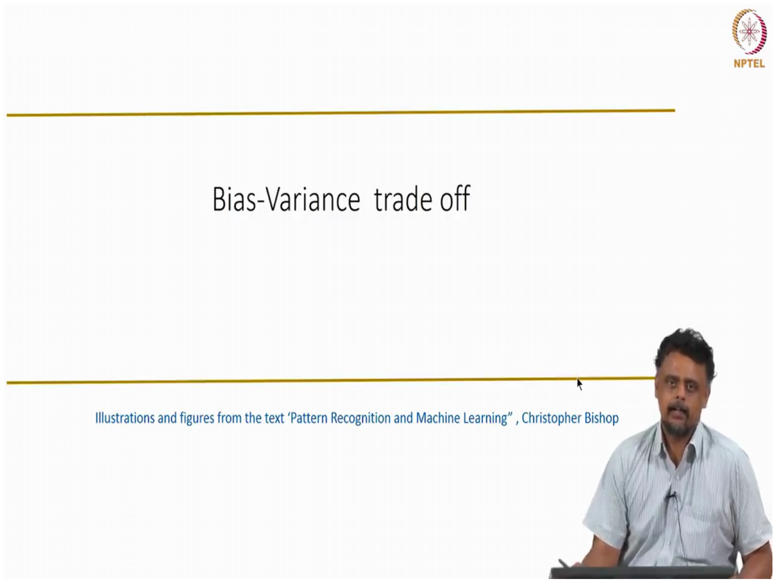
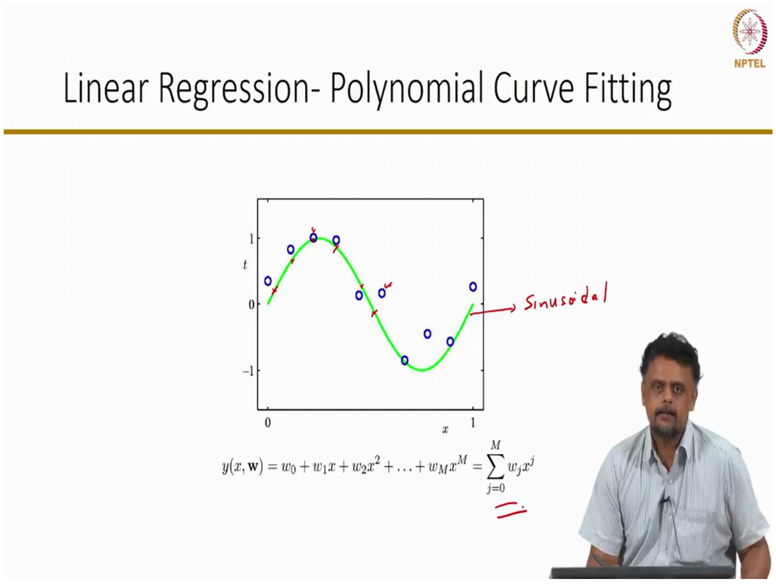
Machine Learning for Engineering and Science Applications
Professor Dr Ganapathy Krishnamurthi
Department of Engineering Design
Indian Institute of Technology, Madras
Bias-Variance Trade-off

(Refer Slide Time: 0:19)



Bias-Variance trade off

Illustrations and figures from the text 'Pattern Recognition and Machine Learning", Christopher Bishop

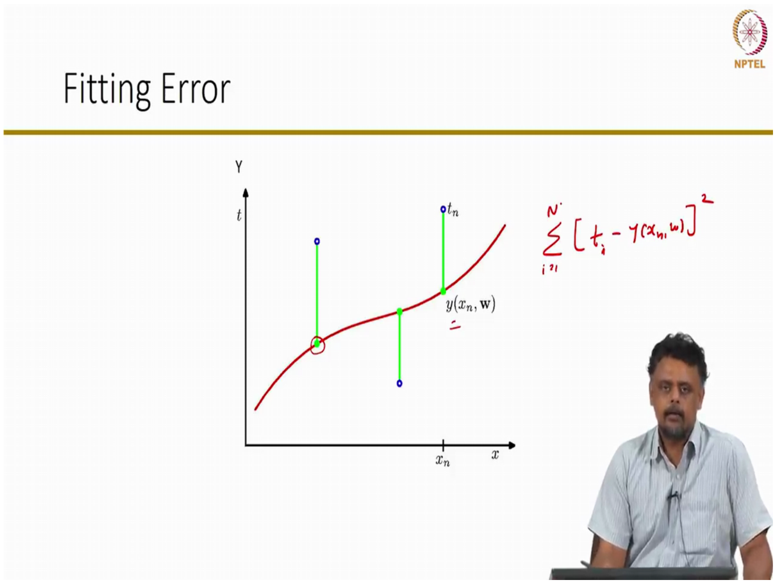


Linear Regression- Polynomial Curve Fitting

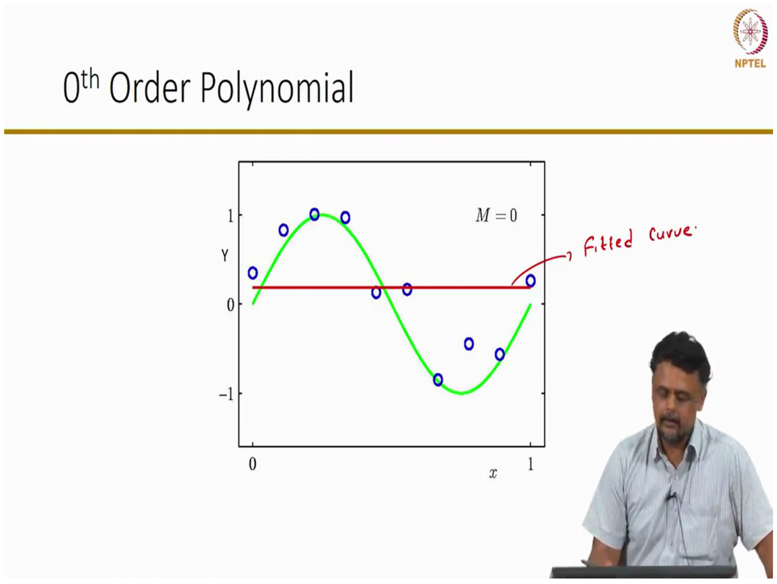
$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

Hello and welcome back, with this video, we will about the bias variance trade-off, which is an important component when you try to train different machine learning algorithms. Most of the examples and illustrations are taken from textbook by Christopher Bishop, Pattern Recognition in Machine Learning. And we will use those illustrations and material to understand bias variance trade-off. So let us consider this simple example, where we are shown this green curve, this is sinusoidal curve, from which we draw some data points at some specific intervals, in this case right on top here, and we just add noise to it to obtain this dataset which is given by the blue dot, blue circles.

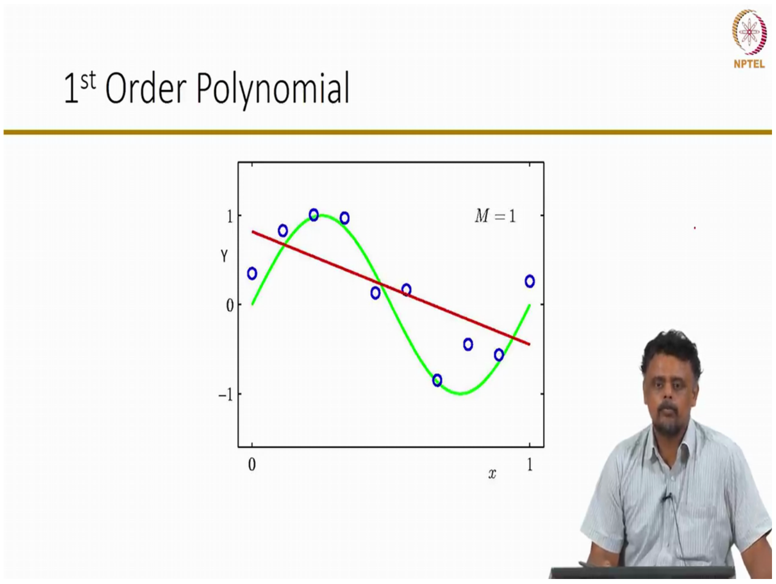
(Refer Slide Time: 1:22)



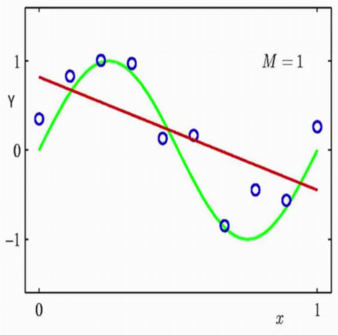
Fitting Error



0th Order Polynomial



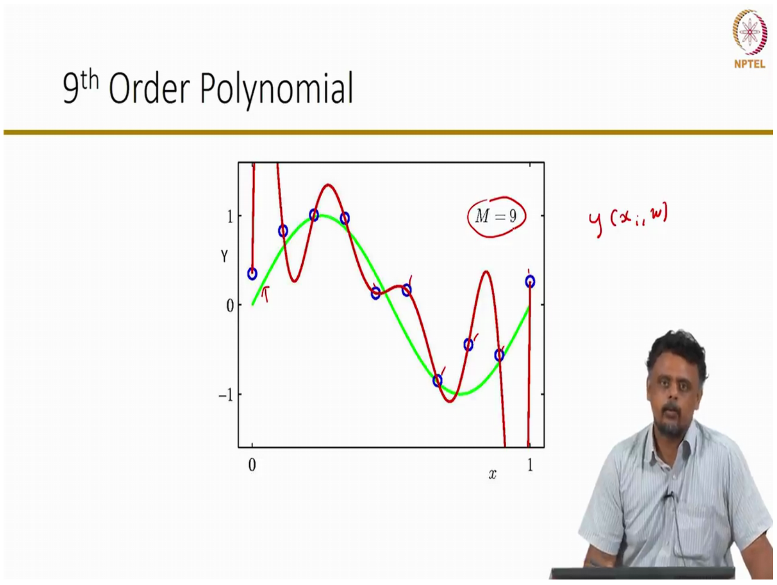
1st Order Polynomial



So the idea here is to, given that, given the data set consists of these blue circles, we want to fit into a polynomial of this form given here. So, what we are going to do is to vary the degree of the polynomial and see how the fix look like. So, how do we fit them, we use an error term to perform the regression and in this case, it is just the least squared error, which is given, which is illustrated in this figure. So t is the ground truth or the correct answer and whatever your polynomial outputs, and it is given by y, so your error is basically what you are going to use, this + this.

And we are going to sum it over all the data points that we get, so ti over here in this case it is n square and i will go from 1 to N data points. So given with his error, we are going to fit the given data set to a polynomial of varying degrees. And I will just what each of them look like. So we have a zero degree polynomial, which is nothing but a constant term, so as you can see there is a red line, which is actually the fit, the fitted curve, of course does not match the data that we have used.

(Refer Slide Time: 2:42)

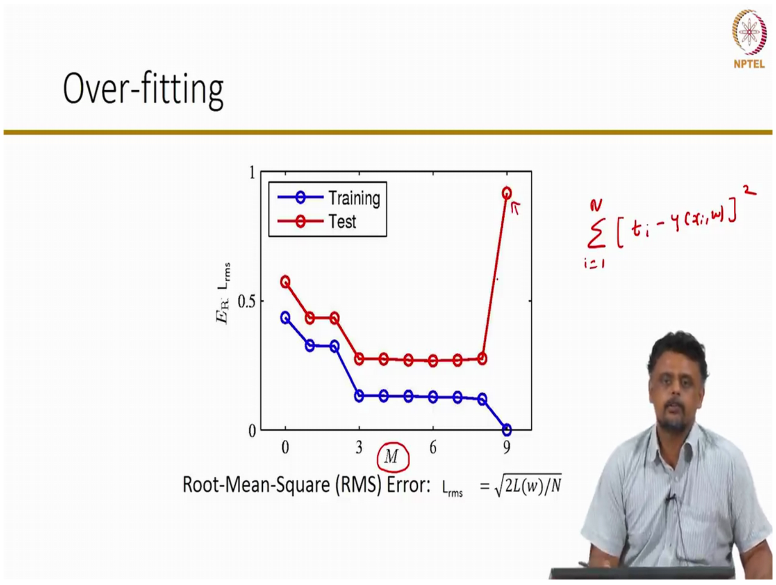


9th Order Polynomial

Again we use a polynomial of degree 1, which is nothing but a linear fit and once you start to go, get higher degree polynomial, in this case polynomial of degree 3 here and so on and so forth, till we come to the polynomial of degree 9. In this case you see that the red curve, which is the fit, that is the output y of xi W, fix goes through every one of these points. So it goes through all the blue circles. However in between the blue circles you see that it is off, what do you mean, what do I mean by this off, so we know that our new circles are drawn from this green curve.

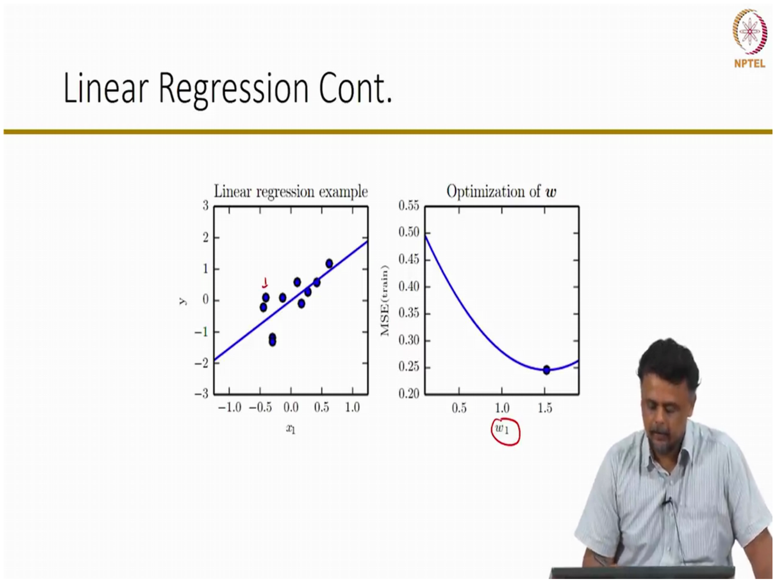
So ideally when you are done with the fit, you would expect the red curve to lie close to the green curve, but in this case, in between samples is actually off. So even the with higher degree polynomial is, we are able to fit every point exactly, so that our fitting error is very small. We see that in points, other than the blue circles, it is actually quite far from the ground truth, okay.

(Refer Slide Time: 3:32)



Over-fitting

Root-Mean-Square (RMS) Error: $L_{rms} = \sqrt{2L(w)/N}$

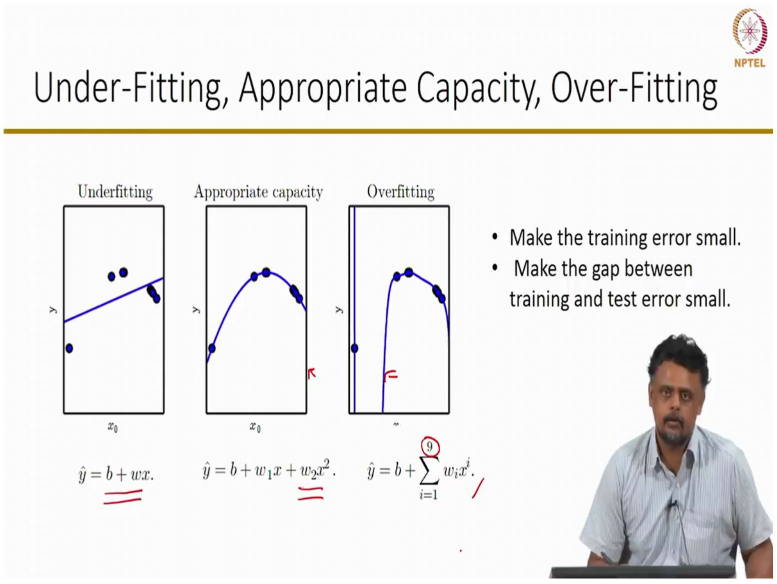


Linear Regression Cont.

So, if we actually plot the error, the error as we Define, in the previous slide I showed this as your ground truth - y, which is the polynomial which you are sitting into and W is the parameters and the number of terms is N. So as you fit the error 4 different values of n that is really of the polynomial, see that as we hit the higher degree polynomials, the error on the training data is very small, which is what the blue circle indicates. However the test data error, this starts to diverge.

So, this phenomenon is referred to as overshooting. Similarly has become back here, we see that again, there is quite high when we are using a polynomial of degree zero, that is we are just fitting it to a constant function. So in both the cases, we have a fairly large error, one in this end of the spectrum, we can call this under fitting and at this end of the spectrum, we will call it over fitting. So, just to summarise, we will see the similar plots, we will look together just to get an idea of what is going on. So we have the selection of these blue points, which we try to fit to different polynomials, polynomials of different degrees.

(Refer Slide Time: 5:43)



Under-Fitting, Appropriate Capacity, Over-Fitting

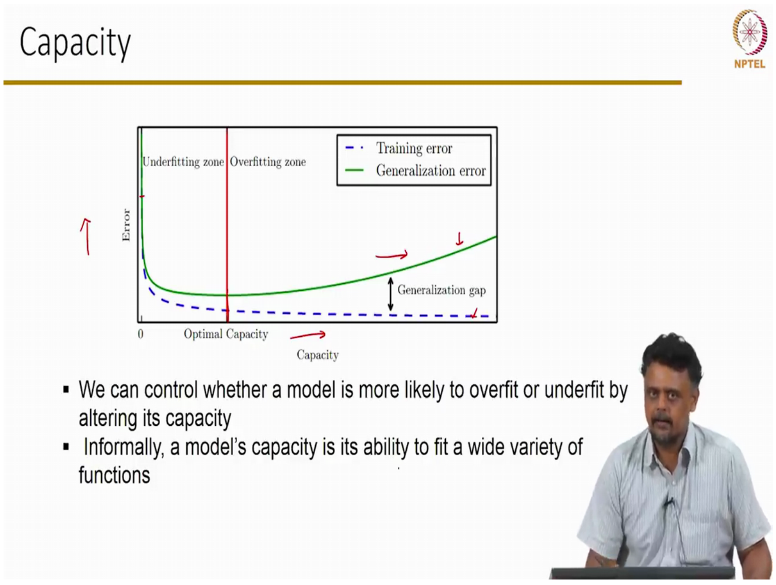
- Make the training error small.
- Make the gap between training and test error small.

$\hat{y} = b + wx.$ $\hat{y} = b + w_1 x + w_2 x^2.$ $\hat{y} = b + \sum_{i=1}^{9} w_i x^i.$

And we do that by estimating the parameters, in this case the Ws are the parameters, in this case it is a linear fit, so there is only one parameter W1, there is W0, which was not chosen the figure. So what we are seeing here is basically, here we have this one model, which is a polynomial of degree 1 and this is polynomial of degree 2, because it has 2 more terms x and x square and this is a polynomial of degree 9, okay. So as you can see the polynomial of degree 2 seems to fit properly in the sense that it goes through all the blue points and this is when we visualize the green curve superimposed on it, it is actually close to the green curve also.

While in this case, when you use higher degree polynomial, it actually does again fit through the blue points correctly but in between the blue points, where there is more data, which is not shown here, you see that the blue curve is actually off. But what is the fit here is basically obtain with polynomial of degree 2, which among these 3 models that we see here. So this is referred to as a model with appropriate capacity. So when you say capacity, you can say basically the number of parameters in your model.

(Refer Slide Time: 6:18)



Capacity

- We can control whether a model is more likely to overfit or underfit by altering its capacity
- Informally, a model's capacity is its ability to fit a wide variety of functions

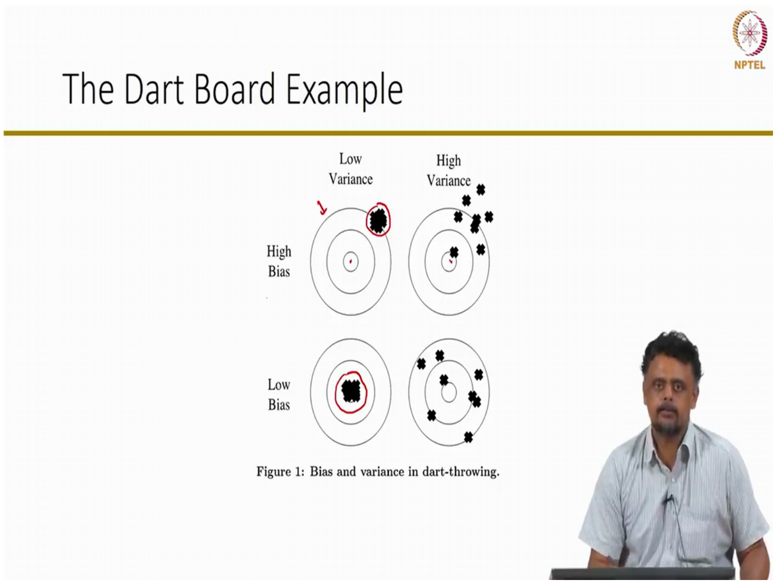
So if you have a polynomial of degree 9, at least 9+ the W0 parameter, which is 10. So, given this situation, we want to figure out what the bias various trade-offs does is to give us an idea how to figure out the appropriate capacity for a given problem. Okay. So we again, once again we look at this plot here, which shows the capacity on the x-axis and the error, that is the training error, the training and testing error that you get for a given model. So just to make, for ease of understanding you can think of capacity is a degree of the polynomial.

So we can see as you increase the capacity beyond a certain point, the training error has go down, right, it is much smaller. However the green curve which shows through the generalisation error in that sense, generalisation error is when you use testing data which are not part of the training data. So, we saw some of those blue circles in the plots are like because the data used to train our models. Suppose we choose some other points which do not coincide with the blue circles and then give it as input to our model, the output is actually quite far from the ground truth, which leads to a very high error.

At this end of the spectrum, we once again see that both the training as the generalisation error are quite high. The plot can be little bit misleading, it seems to overlap there but all you have to understand is that the error is very high, so that is not also a desirable thing. So, somewhere in between is the optimal capacity which in our, in the case of our example, it is basically a polynomial of degree 2 or 3, which seems to give the best fit, then it goes through all the low points are very close to them. It is also close to the green curve, which is our ground truth, okay.

So by altering the capacity we can decide whether, we can make the model under fit or over fit. Under fitting is when you have very large error in terms of accuracy of the model. It is far away from the ground truth, the research that you get. And hear this as overshooting, wherein it actually fits the trade data perfectly but it does not generalise very well, so new data gives the very large error, okay. So just to summarise this, this is the usual way in which this bias and variance are visualised.

(Refer Slide Time: 8:15)

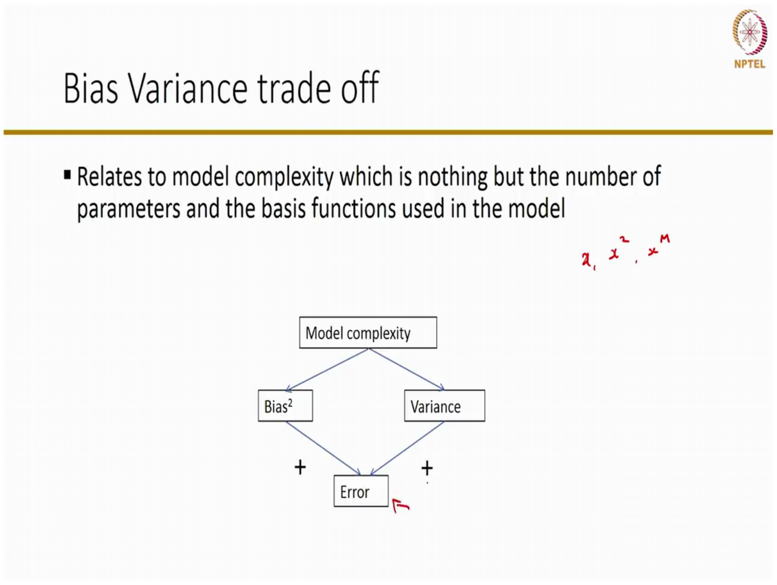


The Dart Board Example

Figure 1: Bias and variance in dart-throwing.

So in this case look at this model here, this is called the high, this is equivalent of the high bias model, that is, we want to be close to, this is the dartboard example, so we will like all the darts to hit the bull's-eye, which is the centre. What all the darts have actually hit quite far away from the bull's-eye, this is that they are not accurate. So it is high buyers, but they are all very close together, okay, so that is low variance, okay. Again if we look at 4 in the case of high bias, and high variance, once again, most of the darts has fallen far away from the bull's-eye, but again they are not very close together, they are highly dispersed, so that is high bias and high variance.

(Refer Slide Time: 9:11)



Bias Variance trade off

- Relates to model complexity which is nothing but the number of parameters and the basis functions used in the model

x, x^2, x^M

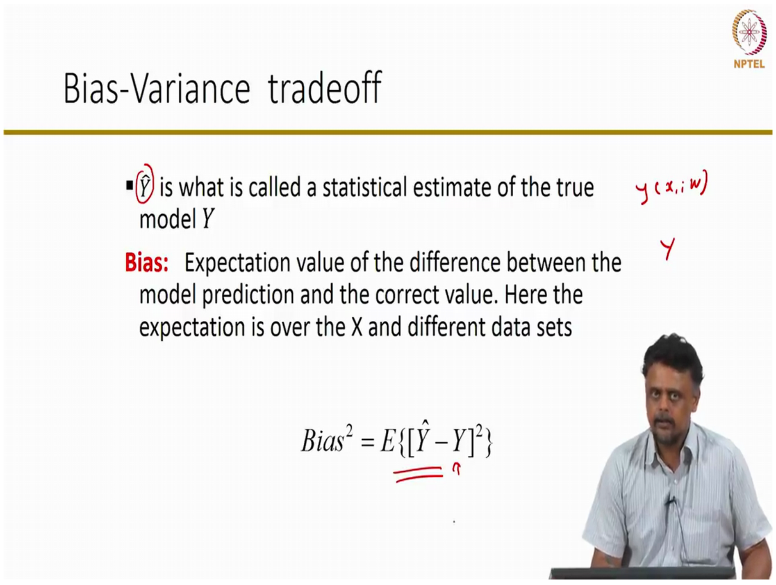
$$\begin{tikzcd} & \text{Model complexity} \arrow[dl] \arrow[dr] & \\ \text{Bias}^2 \arrow[dr, "+"'] & & \text{Variance} \arrow[dl, "+"] \\ & \text{Error} & \end{tikzcd}$$

The ideal model which is what we would like is the low bias and low variance, that is most of them are close to the bull's-eye in and around it with very small dispersion. And the other case is when we have the low bias and the high various model, wherein they are simply close but then again this was far away from each other, okay. So, this bias various trade-offs is basically, it relates to model complexity or crudely we can think of them as our number of parameters and the basis functions.

When we see the basis functions, so for instance, in the case of the example we saw some other basis functions are nothing but the x, x square, so on so forth through x to the power m by m is, where m indicates the model complexity. We have a lot more parameters, so general it is a complex model. The error that we get in training, so you can think of this as the fitting error or the training error, in this case it can be decomposed into 2 components, one is the bias squared error and the variance.

And this depends on the model complexity. This, the component of the error, the bias contributes to other contribution of the bias to the error and the contribution of the variance of their depends on the model complexity. So we what actually go through the derivation for that, so you can actually start with a squared error and decompose it into 2 sums, won the bias and variance. But we will just look at the terms themselves to understand what they mean by that.

(Refer Slide Time: 10:45)



Bias-Variance tradeoff

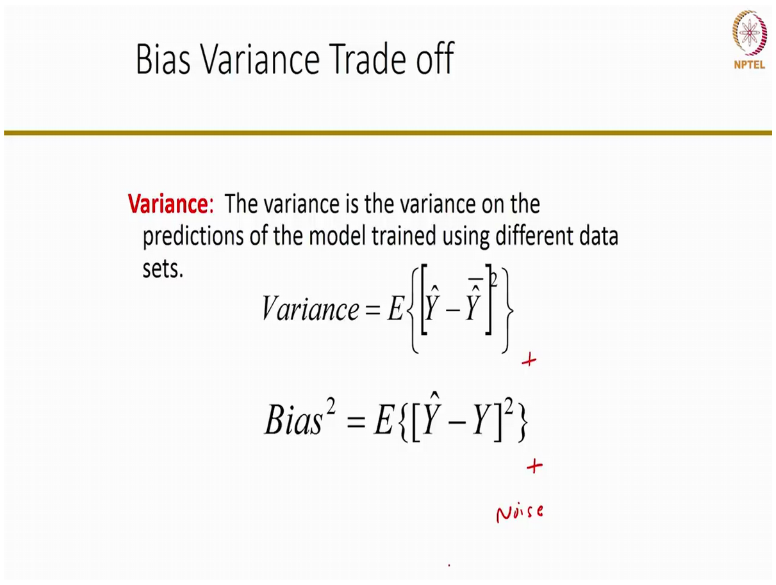
- $\hat{Y}$ is what is called a statistical estimate of the true model Y

$y(x_i, w)$

Y

Bias: Expectation value of the difference between the model prediction and the correct value. Here the expectation is over the X and different data sets

$$Bias^2 = E\{[\hat{Y} - Y]^2\}$$



Bias Variance Trade off

Variance: The variance is the variance on the predictions of the model trained using different data sets.

$$Variance = E\left\{\left[\hat{Y} - \bar{\hat{Y}}\right]^2\right\}$$

$+$

$$Bias^2 = E\{[\hat{Y} - Y]^2\}$$

$+$

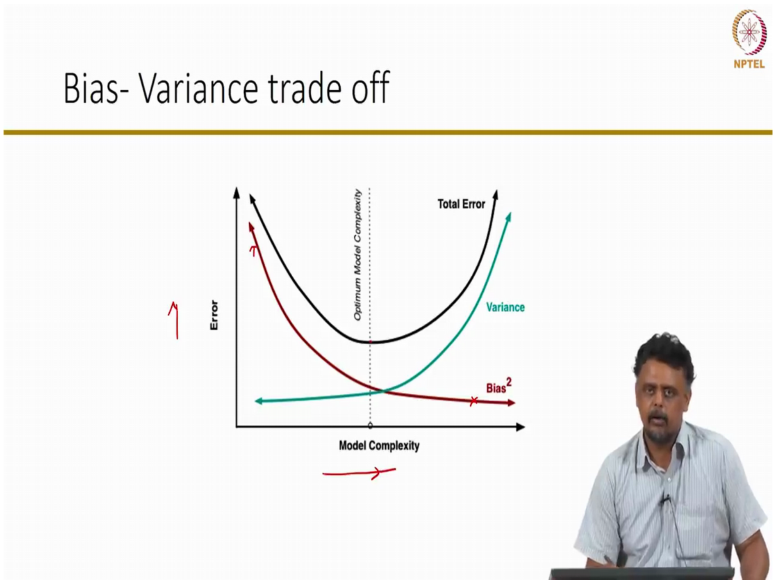
Noise

So if we consider dataset, right and we have seen that y is our model and Ws are the parameters of our model, xi is the input and you can think of capital Y as a ground truth or the correct answer that would like to get. So Y hat is the statistical estimate, we will go, we will see later that why we call this as statistical estimate, for now we can think of this as the answer that is a petition made by your model, or the output audio model, okay. So the bias is the expectation value of the difference between the model prediction and the correct value, okay.

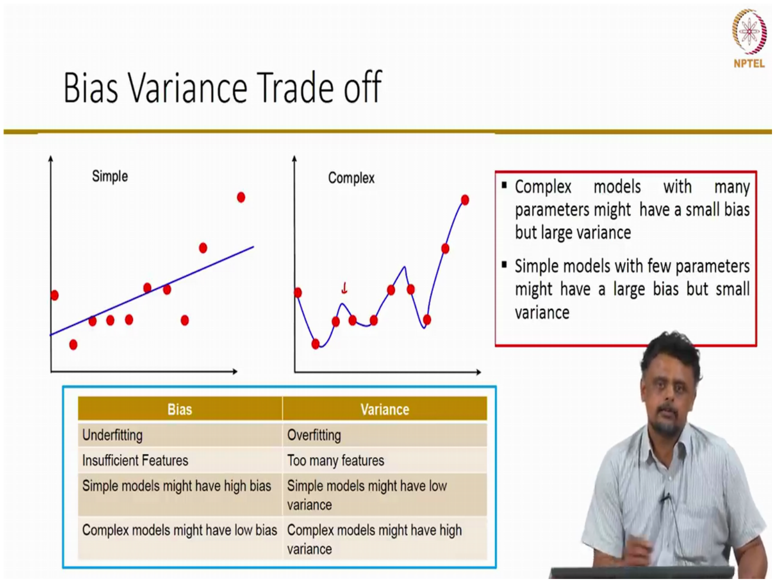
So we all look at, we all know what expectation value is, but here I am only talking about the single values, I will clarify later what we mean by expectation. So you can define bias to be this term, so which is nothing but the, this is a ground truth, the difference between the ground truth, which is the correct value that your model should predict and the answer that your model actually gives, the difference between them squared at the average of that is actually what we call the bias, the bias term.

The variance is the variance of your prediction itself. So you will have a range of predictions and the variance among them is what we call the various, obviously the name impacts variance. So the error that we get, the fitting error that we consider can be written as the sum of these 2 terms. There is also one more term called noise, this is noise which is inherent in your data, because all your answers are not, even your ground truth has some errors in it, there is always some noise, so which we are not taking into account in this model, okay.

(Refer Slide Time: 12:16)



Bias- Variance trade off



Bias Variance Trade off

Simple
Complex

- Complex models with many parameters might have a small bias but large variance
- Simple models with few parameters might have a large bias but small variance

Bias	Variance
Underfitting	Overfitting
Insufficient Features	Too many features
Simple models might have high bias	Simple models might have low variance
Complex models might have low bias	Complex models might have high variance

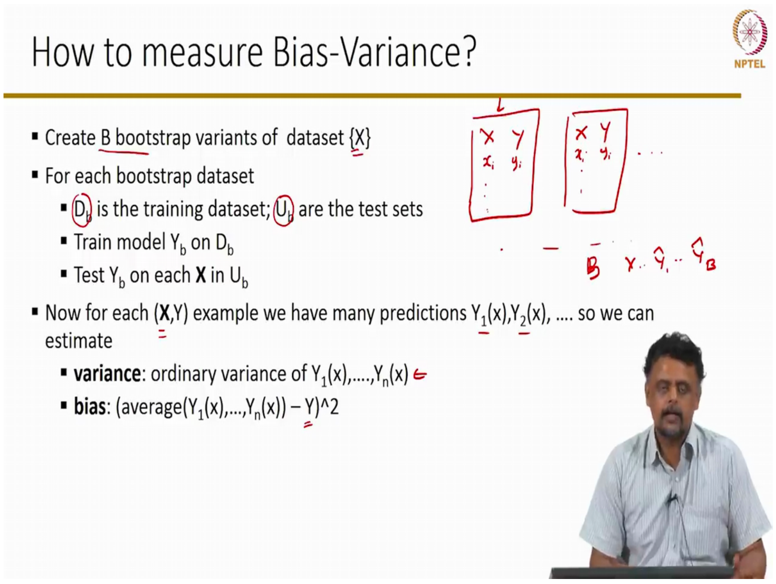
How do we, so the error that you get is the sum of these 2, so if you look at the plot again, when we consider model complicity along the x-axis and the actual fitting error along the y-axis, we see that as the model complexity increases, the bias error, the bias component of the error comes down, which we saw earlier, right. Because when you look at the polynomial of degree 9, it was able to, the curve that we finally got was able to go through all the blue circles, that is the data points that we used, okay.

Similarly as you look at the model complicity as you have a very simple model, in the case we have the polynomial of degree zero, then the error becomes very high, the contribution is very high, okay. And if you look at the various component of it, as you increase the complicity of the model, you have higher various and as you decrease, the variance decreases. Okay, so these 2 add up to give you the total error, so there is a point at which you have an optimum error, right capacity model, it strikes the right compromise between the buyers and the variance, okay.

So, typically, to understand it from the point of view of curve fitting, we have a simple model, which in this case, you know we have, let us say these red are the data points that we have drawn from some complex function and we are trying to fit it using a simple model, in this case the straight-line fit. A simple model will give you, it has insufficient number of parameters and features, but it will have higher bias, okay. On the other hand if you use a slightly complex model to fit the same vector, which is what we have seen in the right.

Then it will have a lot more features than you actually need. But it will have very high variance, you can say this will have very low bias, in the sense it will actually be able to fit through all the data points, but you are variance will be very large. On the other hand, simple model will have lower variance, okay. So this is how you understand it from the point of view of curve fitting.

(Refer Slide Time: 14:17)



How to measure Bias-Variance?

- Create B bootstrap variants of dataset {X}
- For each bootstrap dataset
  - D_b is the training dataset; U_b are the test sets
  - Train model Y_b on D_b
  - Test Y_b on each **X** in U_b
- Now for each (**X**,Y) example we have many predictions $Y_1(x)$,$Y_2(x)$, ... so we can estimate
  - **variance**: ordinary variance of $Y_1(x)$,....,$Y_n(x)$
  - **bias**: (average($Y_1(x)$,...,$Y_n(x)$) – Y)^2

Now, how do we crudely speaking how do we measure this bias variance? We will not do it this way but just for this, just to understand what these terms mean, we will have to go through this process, okay. So here some of these terms may not be familiar to you but I will make you make the understanding easier. We will consider B bootstrap samples of variance of dataset X. So, think of it as the 2 have access to, let us say 10 different or 20 different datasets drawn from the same data distribution. In this if we go back to our example, so we had this green which was a sinusoidal curve, you sample 10 points at a time from the sinusoidal curve and then you do that let us say 10 or 20 times, okay.

So you will have about 10-20 datasets, each dataset having 10 points, that is what we call bootstrapping. So we have B bootstrap variants, from where, from our data, okay, corresponding X and Y, okay. So for each of the bootstrap set, we call T, that is the data that we have extracted as the training set, okay. And for each of them we will have a corresponding test set also, okay. So the way to think about it is you have X and Y, you will have one set of xi, so this is one dataset that you draw from our cup, let us say the green cover that we saw, the sinusoidal curve.

And then you will make, you will draw different set of X and Y, again xi and the corresponding yi. So you will have in this case B datasets, so you do this, let us a small b times, right or capital B times, right. B can be 10, 20, 100, whatever you like. For each one of these datasets, we will fit to a model, which is, which can be, in our case for example can be a polynomial of degree n, we choose m, but of course you fit all of them to the same polynomial, okay. And then you test it on its separate held out dataset, each of them will have a different test data.

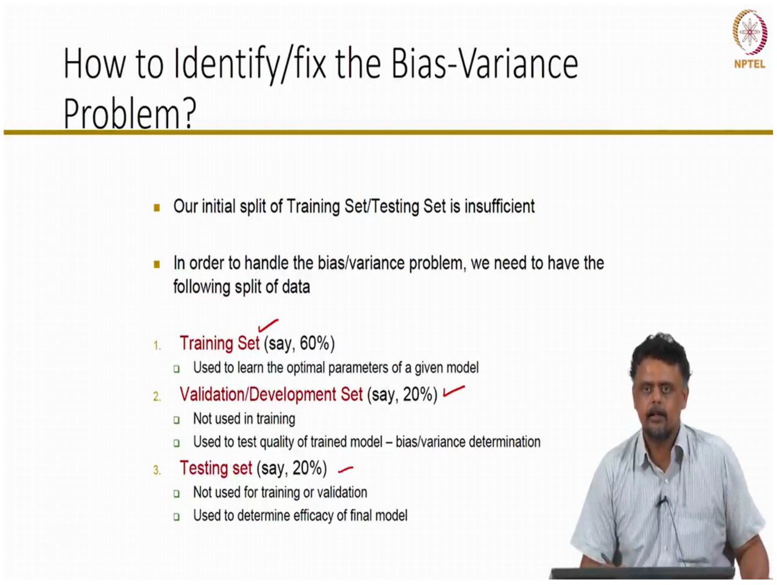
Now that we have B different datasets, for every model that we use, let us say we are using a model of degree 2, there is a polynomial of degree 2. For every X we will have many predictions, red, Y1, Y2, up to the number of data points we have, right. So, what we wanted to cover for each X, so remember that after we fit this model, we can evaluate that model for any X. So, if there are B models, so we will fit B models to the B bootstrap variance or the B samples that we have, B datasets that we have.

We will have, we can evaluate for single X, we can have B Ys, okay. So for one X, since you have B models, so for one X, we can have outputs up to B, right, B outputs. So the variance is nothing but the variance of those outputs. So that is the variance of our model, variance of those outputs. The bias is nothing but the average, for every X you can calculate an average of the Ys that we get and subtract it from the ground truth and take the square, that is a bias square.

So this is how you can actually calculate the bias and variance for the model you choose. Just to summarise, we have B datasets, all coming from the same distribution and with one of those datasets, you will fit the same model. So, in this case you decide to fit a model with a polynomial degree 2, okay. So corresponding you will get the model parameters. Now that you have a model parameters for each one of these B models, you will plug-in individual Xs so for each excuse will get B capital B Ys.

The variance of those Y is the variance of your model, the mean of those Y - the ground truth squared is the bias, okay. Of course you realise that doing this on a real dataset, especially when you have a large model is not going to be feasible, especially since we will not be having access to some many watches offer dataset. So how do we actually do it in a real scenario?

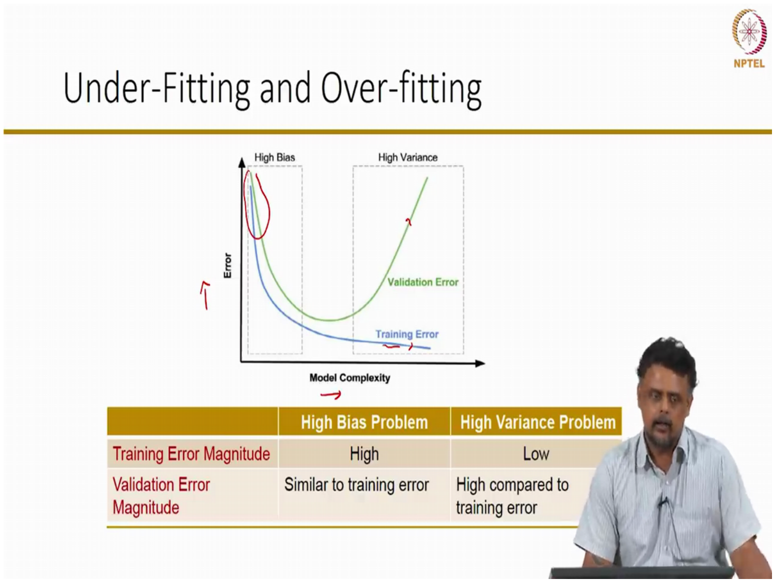
(Refer Slide Time: 18:23)



How to Identify/fix the Bias-Variance Problem?

- Our initial split of Training Set/Testing Set is insufficient
- In order to handle the bias/variance problem, we need to have the following split of data

1. Training Set (say, 60%)
   - Used to learn the optimal parameters of a given model
2. Validation/Development Set (say, 20%)
   - Not used in training
   - Used to test quality of trained model – bias/variance determination
3. Testing set (say, 20%)
   - Not used for training or validation
   - Used to determine efficacy of final model



Under-Fitting and Over-fitting

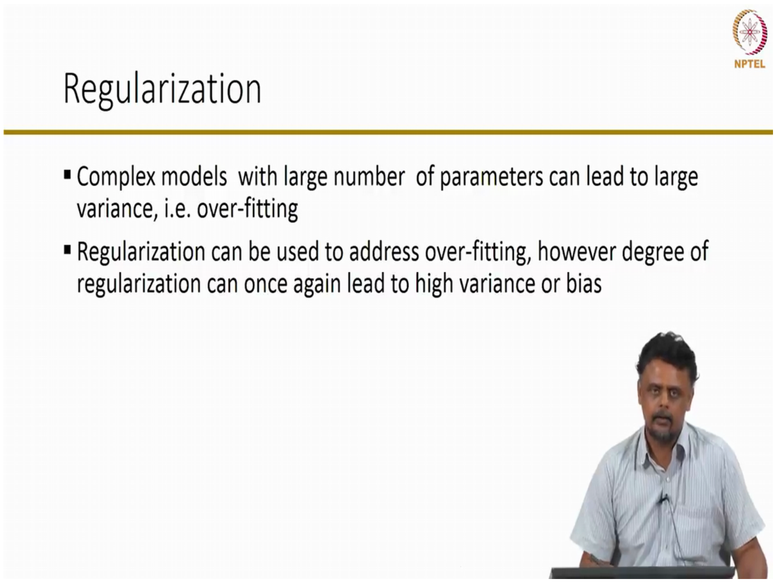
	High Bias Problem	High Variance Problem
Training Error Magnitude	High	Low
Validation Error Magnitude	Similar to training error	High compared to training error

One of the prescribed methods is to split your existing data into a training, which can be 60 to 70 percent, there is something called a validation or development set, this is what you will monitor, in order to determine whether you are model is a high bias or high variance. And you have a testing set in the end, just 2, once you figure out the correct model using the validation dataset, you will test it to see whether it is as good as you think it is. Okay. So, here is the problem, right, so then you have model capacity and the error, okay.

So you will plot both, the training error as well as the validation data error for different model complexities. So if the validation data and the training magnitude and the training data error, are both high, we saw that plot a few slides ago, then it means that it is a high bias problem, that your model has a high bias, right. Both your training error and, this is basically in this region, right. On the other hand, if the training error is low, if the training data has low variance, right, while you are validation has higher, and this is, sorry, let me repeat that, so if your training error is very low, but you are validation error is very high, then you have a high variance problem.

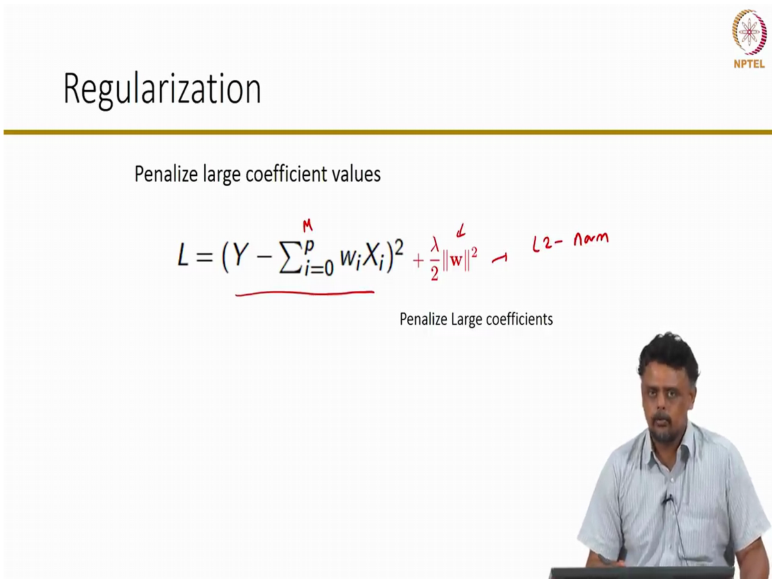
So this is what is in this case. So your training error became very low but your validation error is very high, okay. So, just to recap, the idea is to split your training data into 3, one is the training data, which we will use to figure out the parameters of your model. And validation data is that, is the one that you will use to check whether your model, whether the model has high bias or higher variant. If the model has very high training and validation data error, then you have high bias, then the model has high bias. If your training error is very low, while your validation error is very high, then it means that you have very high variance model, which means you have very complex model, it might not be necessary for the data that you have at your disposal.

(Refer Slide Time: 20:33)



Regularization

- Complex models with large number of parameters can lead to large variance, i.e. over-fitting
- Regularization can be used to address over-fitting, however degree of regularization can once again lead to high variance or bias



Regularization

Penalize large coefficient values

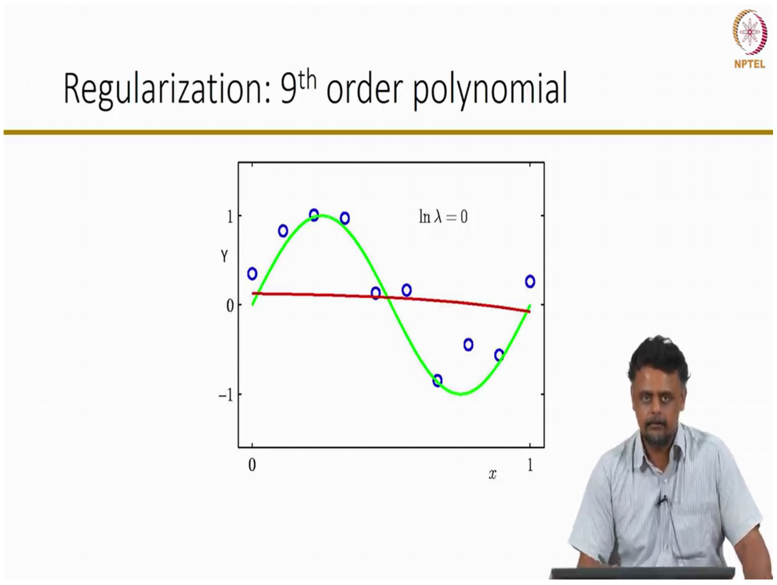
$$L = (Y - \sum_{i=0}^{\overset{M}{P}} w_i X_i)^2 + \frac{\lambda}{2}\|\mathrm{w}\|^2 \rightarrow L2\text{- Norm}$$

Penalize Large coefficients

Okay, so then, what are summarise here, so then how do we handle this problem. So, we have very complex model that can lead to over fitting, okay. Then, but then you would like to play it safe if you actually want to retain the complex model because it seems to generalise very well. So if you think that it generalises very well, then how do you address the high variance problem, how do you make sure that your fits are good? So this is accomplished using regularisation, okay. So what is regularisation?

Which is in this case, we have the least square error, here the symbols are slightly different, so we have used P instead of capital M, so it does not matter. Wis are your parameters, Xis are you input, Y is the ground truth, okay. So what we do is, we penalise large coefficients by adding a term to your fitting error. In this case it is lambda over 2 double squared, okay. And W is the 2 norm, you must have seen this 2 norm or the L2 norm. Add the L2 norm to your fitting error and then you do the fit as before. So what it does is that it is a very large coefficients, this will penalise it.

(Refer Slide Time: 22:00)



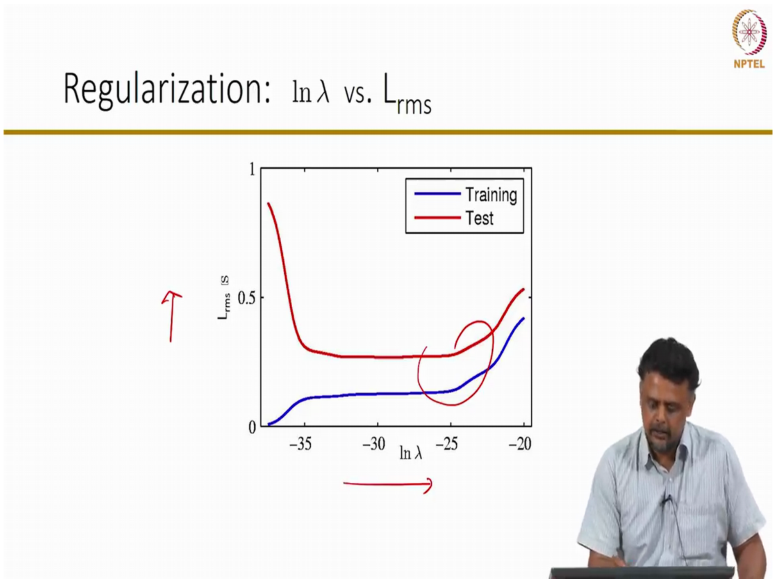
Regularization: 9th order polynomial

So, why do we have to penalise very large coefficients? We will show you why that is required and the will look after we have examined what the curves look like after you do the regularisation. So, if we choose, if we have a same, we go back to the polynomial example where we are now using M equal to 9. And we saw earlier that there was a huge error, the fitted curve which is a red curve was oscillating wildly. But now it is a little bit more boot, because we added the regularisation term, the log lambda is how you measure the strength of the regularisation term.

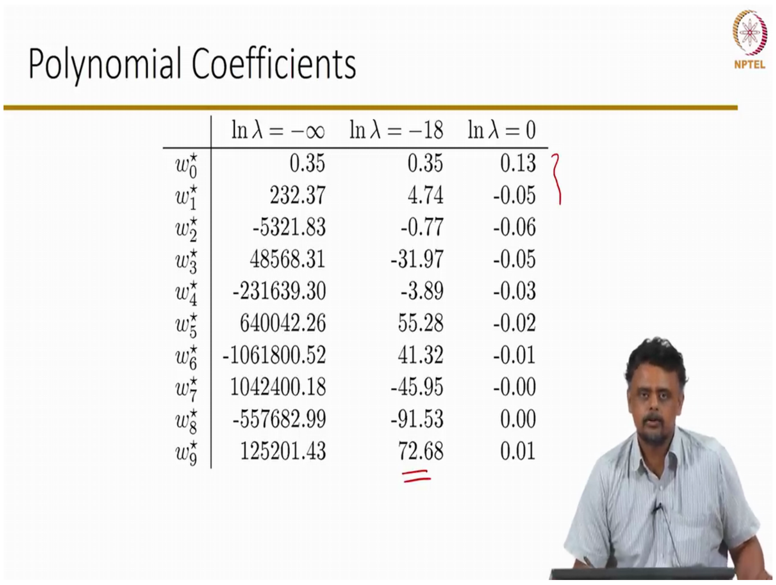
So, because typically lambda is a very small number between zero and one, so it can be like 0.01 or something, so expressing it in log lambda is more meaningful. So, when we use a small hyper parameter lambda, so lambda is a hyper parameter as we call it. So we have L2 regularisation, then even with polynomial of degree 9, we get a reasonably good fit. And when we use log lambda zero, this is, this is very strong regularisation, so then it actually becomes a very high bias model. This is very similar to what we saw when we had just a polynomial of degree zero, okay.

So by adjusting the strength of this lambda, we can control the bias variance trade-off, that is the idea behind having a regularisation. So, we have a very, this is a kind of, this lambda i is a very small number, so then we have a very smooth curve fitting. But when we make our regularisation very small, where lambda is close to 1, then you have a very, this case becomes a very high bias model, right.

(Refer Slide Time: 23:25)



Regularization: $\ln\lambda$ vs. L_{rms}



Polynomial Coefficients

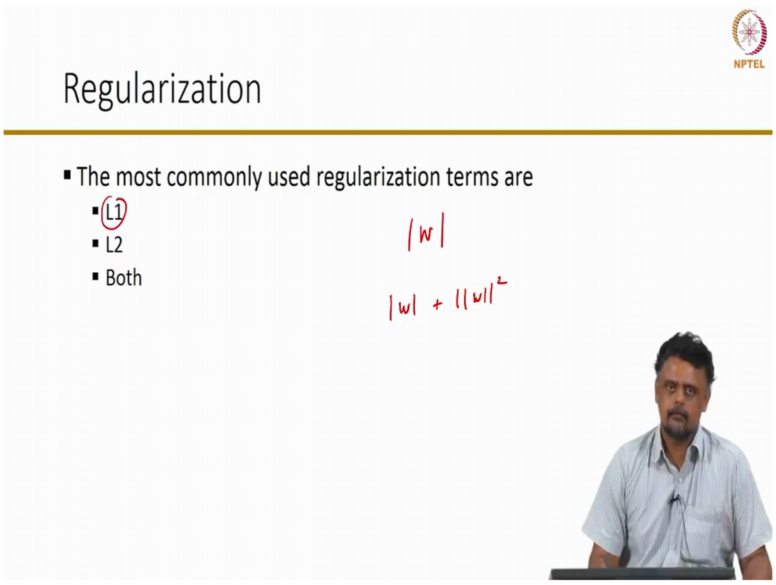
	$\ln\lambda = -\infty$	$\ln\lambda = -18$	$\ln\lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

So, then once we have the, once we have the regularisation in place, in this case the L2 regularisation, then we can look at the set of the regularisation strength of the regularisation versus the error, LRM versus the error, the fitting error that we measure, the root mean square error. And we see that beyond a point, it is a very good region here to operate, right. So we are okay here, by using a very highly complex model, polynomial of degree 9, we are going to, we are getting the training and test data to be close to each other by choosing an appropriate value for lambda, okay.

So, what do I mean by penalising the high bit. So, if we do not have regularisation in place and we fit using the M degree polynomial, you see that some of the weights that we estimate, the parameters of the model are very high, we see that. Of course this when you examine this, you see that this is meaningless, it should not be this way, right. So once we start, once we adding the regularisation, so this is without regularisation, which is log lambda is - infinity, this is some medium level of regular occasion, when we got very good results with a very nice fit.

And this is a very strong regularisation you see that most of them are going to a very small number, so we have only included 2 or 3 significant figures after the decimal place, so it does not show up. So, by adjusting lambda, so we can actually do the bias variants trade off, okay. So this we saw, this is for the L2 regularisation wherein we add the L2 norm of the parameter to your fitting error cross function.

(Refer Slide Time: 25:05)



Regularization

- The most commonly used regularization terms are
  - L1
  - L2
  - Both

$|w|$

$|w| + ||w||^2$

We can also do the L1 norm, which is nothing but the absolute value of your parameters. In fact some, many, you if you go to the deep learning, many of the network that you train will have both the combination of yours L1 and L2 norms, okay. So, typically we have a very complex models, you would end up with a very high variance model, in the sense, the generalisation error is very high. But will not predict out of, the data that comes, which is not part of the training data, if you give it the data, it would predict properly.

So in order to control that high various problems, you add in regularisation terms which will penalise very high values of your parameter estimate and smoothen your model to have reasonable variance and reasonable bias, thank you.