Week 2
Video Transcripts

Video 1 (08.30): Geometry of Least Squares Regression

Okay, so thinking geometrically about least squares is something that's very useful to develop intuitions about what it's doing. It's not necessary in order to use it or to even do mathematical things with it like analyze it, but intuitively it could be very helpful. So, in the next couple of slides I just want to build some intuition about what least squares regression is doing, geometrically. So, as we discussed before, we want to minimize something that looks like this where we take all of the outputs 'y' and put them into a n-dimensional vector if we have 'n' observations. And we approximate that with this matrix vector product, so we have the matrix 'X' where the i^{th} row corresponds to the 'i' input vector or some function of it. And the weights then are what we want to learn. So, we can think of 'y' as some point in 'R^n', a very high-dimensional space if we have many observations. And what we're trying to do is find a vector 'w' in order to get this matrix vector product physically close to the vector 'y' in 'R^n'. So, remember that this matrix vector product really entails taking the j^{th} column of the matrix 'x' written this way, multiplying it by the weight 'w_j', which is just a scalar.

So we take the vector 'X_j', j^{th} column of 'X'. We then scale it in some way according to 'w_j'. So if it's 'w_j's' positive, we stretch it or shrink it but keep the direction the same. If 'w_j' is negative, then we flip its direction and stretch it or shrink it. We do this for each column of the matrix 'X' and then add these vectors up. So, 'x_j' is pointing in one direction, 'x_j' plus one is pointing in another direction. We add those up and we're trying to now make the sum of these vectors get as close to 'y' as possible which is a point in 'R^n'. So what least squares is doing is it's basically saying the best vector 'w' is the one that gets these sum of these vectors as close to 'y' as possible, according to the Euclidean distance, which is the intuitive direct line. So, the vector constructed from 'X_w' gives me a point in 'R^n' and then I basically say what is the squared length of that point to the point I'm approximating which is 'y'. So, I draw a line between the two and I return the squared length of that line and that's equal to the sum of squared errors. I'm trying to minimize that so I'm literally trying to make the distance that I would have to travel in a direct line from my approximation to 'y' as small as possible. So, here's an illustration of that. In this case we have, because I'm trying to show it, we can only show it for three observations. So, 'y' is a vector in 'R^3', meaning that we have three data points. 'X_1' is the vector of ones, the first column of 'X'. So, this is a vector of one. So this point here would be one in our three according to some axis. And 'X_2' is then the second column of the matrix big 'X'. So, we're assuming that our data points are in 'R', our inputs are in 'R', and we have three pairs, 'x_1', 'y_1', through 'x_3', 'y_3'. And so, here's the first column of our matrix 'X'. The second column of our matrix 'X'. And we're now trying to weight these two vectors to get them as close to the output vector 'y' as possible. So if we have two columns to 'X', we basically have two vectors that we get to play with in 'R^3'. So of course unless, it's very rare that we could basically–we could take two vectors in 'R^3' and actually hit this vector 'y' in 'R^3'. We don't expect to be able to do that so what we're trying to do, is in this two-dimensional space spanned by these two vectors, find the point that is

physically the closest to 'y' as possible according to the everyday Euclidean sense.

So in this case, what we're saying is take 'y' and drop a line, a straight line, to the two-dimensional space spanned by these two vectors. So 90 degrees. This point here is physically the closest that we can get the sum of these two vectors to the vector 'y'. And this is the least squares solution. And so that's what we have here that when we take the least squares vector 'w_{LS}', it's a two-dimensional vector for this particular case, what we're saying is take the first dimension, multiply it by 'x_1', the second dimension and multiply it by 'x_2', sum those up and you'll hit this point. That's the approximation 'y' hat of 'y'. That's the closest that we can get the sum of those two vectors to be to the output 'y'. So there are two visual representations that we've discussed about least squares. I'll show them to you here just to differentiate between the two.

So, in the first representation what we showed was a two-dimensional input and the corresponding output 'y' and we showed them as points. So, here we have roughly 10, or so points where the input is two-dimensional, so this is the input, say this point here and then the output is that point there along this dimension 'y'. And, we're showing all of the points in this case and we're representing the linear regression function as a hyperplane like this. So this is the function that's represented in this picture. So something like this can be shown for all 'n', but only when the data is in 'R^2' or 'R^1', if we want to include the–so in r^2 where r–So, okay, so something like this can be shown for as many points as we want. The data is in 'R^2' and then the offset, the shift of this hyperplane is–corresponds to the weight 'w_0'. In this case, we're representing all of the 'y's' in one vector. So here we have different values for 'y', the output according to the input. Here we stack all of the outputs together and then we stack each–we represent each dimension separately for all of the points. And so in something like this, we can only show three points and one-dimensional data if we want to also represent the 'DC' offset. So, these are two ways that we can visually represent them. They're fundamentally different. Each have their own intuitions that help you think about the problem, and both are very useful but they should be kind of separated in your mind when you think about these problems.

Video 2 (04.33): Least Squares Linear Regression

So, let's remind ourselves what the problem is. So again, we have as data measured pairs 'x' 'y', where, 'x' is the input, and we'll assume that's in dimension 'R^d' plus one. The plus one because we take each input 'x' and we attach a one to the first dimension of it. So, we extend it by one dimension for the offset. And the output 'y' is in 'R'. And so we're predicting a real valued output, which is what makes it a regression problem. So the goal is, again, to find a function, 'f' that takes the input 'x' and maps it to the output 'y'. So we're taking an input in 'R^d' plus one and we're mapping it to an output in 'R'. And we're doing it in a way where we can approximate 'y', well by this function, 'f', where we input the corresponding 'x' and also a weight vector 'w', which is, the parameters that we're going to be able to change or modify in order to change the function and change how it maps the input to the output. And again, the definition of linear regression is that the function 'f' is a linear function of the unknown parameters 'w'.

So, we saw last time that this doesn't actually mean that we have a linear function of the inputs 'x'. We could perform some nonlinear function of the inputs 'x' and then send that to 'f'. And as long as it's linear in 'w', we still would have linear regression, and the solution for 'w' is solved by exactly the same equation.
So, let's continue our discussion on least squares. So again, to review, the least squares objective function finds the value of 'w' that minimizes the sum of squared errors. So, what we have is, the output 'y_i' is approximated as a dot product between the input 'x' and the coefficient vector, 'w' in the simplest case.

Last time we saw how this could actually be a nonlinear function of 'x'. But for notation let's just talk about vector 'x'. And we take this error squared and sum it up over all of the data points that we have. We can write this in a nicer matrix vector notation by taking these vectors 'x_i' and putting them along the rows of a matrix 'X'...capital 'X'. So this matrix is 'n' by 'd' plus one. And then we take the vector 'y' formed by taking each of these outputs—'y_1' through 'y_n'— and constructing a vector out of them. And then we take the error—we have the error vector, which is the point vector 'y' minus its approximation 'X_w'. This is the vector of errors for all 'n' of our data points. And when we take the dot product of that error vector with itself, we're summing the square of those errors. So taking this objective function 'L', finding the vector 'w' that minimizes it—because we're seeking to minimize the squared—sum of squared errors. So, we take the gradient of 'L' with respect to the vector 'w'. We get this solution, as its gradient.

And then because we want to find the minimum, we find the vector 'w' at which the gradient is equal to zero. So we find the point 'w' at which this function is minimized. Another way of saying this is, find the vector 'w' at which the gradient of this function is equal to zero. We solve and we get the least square solution, which is equal to this.

Video 3 (10.55): A Probabilistic View

In this part of the course I want to discuss the probabilistic interpretation of least squares, which can help us think about our model, it can help us think about what kind of assumptions we're making with our model, and whether those assumptions are reasonable, and also develop even more intuitions about what it is, that we're doing, and what our model is assuming. So let's recall from previously, the Gaussian distribution, Multivariate Gaussian in n dimensions.

So assuming that the covariance matrix is diagonal, so we write the covariance as, this variance, Sigma squared times an identity matrix. We can write an n-dimensional Gaussian in this way, where we take the dot product of the output of the random variable 'y' minus its mean mu, and then the dot product of that, and pass it through this density function. So now let's ask, what happens if we put a restriction on the mean of this n-dimensional Multivariate Gaussian. What if we define the mean vector mu to be equal to the matrix 'X', which we formed by taking our inputs and putting them along the rows of the

matrix as usual, times the vector 'w', which is something that we don't know. So 'x' we know, 'w' we don't know, and we want to learn it. And we're now going to say that mu is restricted to have this form.

So we plug this matrix vector product into the Gaussian for mu, and then we say let's find the maximum likelihood solution to 'w'. So let's find, in other words, the value of 'w', that maximizes the log likelihood of the output vector 'y', given a mean that has this form, and given a covariance that has diagonal covariance where each dimension has variance Sigma squared. So when we do this, when we take this Multivariate Gaussian, and take its log, we get this function. And now our goal is to find the vector 'w' that maximizes this function, which is the vector that maximizes the likelihood of the data outputs 'y' that we see, given the corresponding inputs 'x'. And so immediately we can see that least squares and maximum likelihood in this case share exactly the same solution. So remember, what's the least squares problem?

We want to find the vector 'w' that minimizes the sum of squared errors. What's the maximum likelihood problem, for our n-dimensional, Multivariate Gaussian assumption on the data?

We're trying to find the vector 'w' that maximizes, so previously minimizes, now maximizes, the log of the likelihood of the data over 'w'. Because this term here does not involve 'w', we don't even bother to write it. And so we're trying to find the vector 'w' that maximizes this function. What you notice is the negative sum of squared errors times some scalar. So, this scalar doesn't do anything.

So it should be clear, that these two problems share exactly the same solution. This is essentially the negative of this problem. But, because here we're trying to minimize and here we're trying to maximize, they have exactly the same solution. So, therefore, in a sense, what we can think of least squares as doing is making an independent Gaussian noise assumption on the error. So this is, just some intuition, that we can develop where we say that the least square solution corresponds to the maximum likelihood solution of this Multivariate Gaussian assumption, on our data. And so it's like saying that each of the errors are independent and identically distributed as a zero mean Gaussian with some variance Sigma squared. And so we can say, that we're effectively modeling our output as being equal to the dot product of the input with the weight vector 'w', plus this *iid*. Gaussian noise. Another way we can say this, is that we're modeling our outputs as independent Gaussian random variables where, the mean is equal to the dot product between the input 'x_i' and the coefficient vector 'w', and the variance is Sigma squared for each of the observations. Or, as on the previous slide, we can simply say that we're making an assumption that the vector 'y', is a Multivariate Gaussian with mean equal to 'X' times 'w', where this 'X' is formed by putting each observation input on the row, and covariance equal to Sigma squared times 'i'. So a diagonal covariance. So, this is like an assumption, that we're going to make. We're going to say that our noise is *iid*. Gaussian, essentially, and that maximum likelihood for this modeling assumption is the least square solution. But using this additional probabilistic line of analysis we can better understand the maximum likelihood solution and, therefore, the least squares solution.

Okay. So, let's look at some of the things that we can say if we make this modelling assumption. So,

again, we're given the modeling assumption, that our observations 'y', so the outputs that we observe in our dataset, were generated according to a Gaussian with mean equal to 'Xw', and then *iid*. Gaussian noise. So, this is going to be the modeling assumption that we make for how we observed the data, how we obtained the data 'y', given the corresponding inputs 'X'. So first, let's ask, what is the expected value of the maximum likelihood solution? And when I say maximum likelihood, you should also think least squares solution because they're the same. So what is the expected value of the maximum likelihood solution under this distribution? So what we're saying is what is the expected value of 'w_{ML}'? We know that the maximum likelihood solution is equal to this. So, we have this matrix-matrix product. There's nothing random there. But then, we also have the output–the observation vector 'y', and we're making a distribution assumption on what how 'y' is generated. So 'y' is random.

So, really, we can write this term like this, where we have the function times the distribution of 'y', and now we're integrating over 'y'. This is the expectation function. So, of course, every expectation you have some assumption of the distribution that you're taking, some distribution you're taking that expectation over. In this case, it's the distribution on 'y'. So, we have a function of 'y' times the distribution on 'y', the density of 'y', and now we integrate over 'y', and that gives us the expected value of this function of the data 'y'. So by basic manipulations of the expectation function, one rule is that because this term here does not involve 'y' at all, it's a constant, there's nothing random here, we can bring this outside of the expectation and obtain this. So, really, we want to take the expectation of 'y'. And because 'y' is a Multivariate Gaussian with this mean and this covariance, we know the expectation is just equal to whatever the mean is. So we plug in the mean, which is 'X' times 'w'. Again, we don't know 'w', but we're making an assumption that there exists some'w', and so, therefore, the data has this mean where the ground truth of what 'w' is, is plugged in, in order to generate the data.

So we input–we replace the expectation of 'y' with its mean, even though we don't necessarily have a value for 'w', and then we find that here we have 'X' transpose 'X', here we have 'X' transpose 'X' inverse, that matrix product cancels out, and we have 'y'. So what this is saying, intuitively, is that, if we have some ground truth value for 'y' for 'w', and we generate–we have some inputs 'x' that we construct in a matrix in this way, and then we generate an output 'y', according to this distribution, and then using that output 'y' we solve the maximum likelihood solution for 'w'. So we have the true 'w' and then we, using the random vector 'y', solve the maximum likelihood solution for 'w'. The expectation of our maximum likelihood solution is equal to the truth. In this case, what we're saying is that, the maximum likelihood solution for the vector 'w' is an unbiased estimate of the ground truth vector 'w', which we don't have access to. So this is good. Least squares or maximum likelihood for this model is going to, in expectation, give us the true parameter, which is what we're trying to learn.

Video 4 (09.54): Review: An Equality from Probability

Okay, so it's great that with maximum likelihood for this problem, the expected solution is the true solution. But even though the expected, in quotes, maximum likelihood solution is the correct one,

should we actually expect to see something near that value?

So, one way to instantly understand this is, if I have a Gaussian random variable with mean mu and Sigma, and variance Sigma squared, and Sigma squared is huge, then even though that random variable in expectation is equal to the mean, since the variance is so huge, I don't actually expect to see something close to the mean. I wouldn't be surprised, if I saw a value, random value very far away from the mean, because the variance is so huge. So, we've shown that the maximum likelihood solution is the correct one, but now we want to calculate the variance of that solution, under the same model and assumption.

So to do this, we should also look at the covariance. So let's recall that, if we have a vector 'y' that's from a Multivariate Gaussian with mean mu and covariant Sigma, then the variance of that vector 'y', the covariance of that vector 'y', is equal to the expectation of the outer product of 'y' with its mean subtracted off, and that's equal to the covariant matrix Sigma. So this is the covariance, the way to calculate the covariance of the random vector 'y'. And for the Gaussian, it's equal to Sigma. And now plugging in the value mu for the expectation, we can equivalently write this in this way, where we show that the variance of 'y' using this sequence where we actually take the outer product here, use the linearity of expectation to bring it, to calculate these three separate expectations, we find that the expectation of 'y' here is mu, and 'y' transpose, the expectation of 'y' transpose is mu transpose, so we have minus 2 mu times mu mu transpose plus mu mu transpose. So we have a minus mu, mu transpose, and then the expectation of 'yy' transpose, and that's equal to the variance. And so immediately, we also can say that with this Gaussian, the expectation of 'y' times 'y' transpose is equal to the covariance plus the mean times the mean transpose. So this is an equality that we'll find useful in a moment. And so let's return to the least squares linear regression problem and calculate the covariance, of the maximum likelihood solution under the Gaussian assumption.

So, again, to do this, we—what we're asking is the expectation of the maximum likelihood solution minus its expectation times that same thing transposed. Equivalently, what we're asking for is the expectation of the maximum likelihood solution times itself transpose, the expectation of this product, minus the expectation of the maximum likelihood solution times the transpose of that expectation. So this is from the previous slide where we have the expectation of 'yy' transpose, where with the variance of this random vector is equal to the expectation of the outer product subtracted by the outer product of its mean. And so let's use this sequence of equalities to calculate this, which can be done in closed form. So we've already shown that this expectation, of the maximum likelihood solution is equal to the truth.

Even though we don't know what it is, we can simply plug that value in there as a place holder. So we know that whatever 'w' is, the expectation of the maximum likelihood solution is equal to 'w'. And then, for the least squares solution here, or for the maximum likelihood solution here, actually plug it in. So this term here is equal to the maximum likelihood solution for 'w', and then this second term is equal to the transpose of that. And again, in this case, because these matrices involving 'X' are constant, there's

nothing random there, we can bring that outside of the expectation. And what we're really asking for is the expectation of 'yy' transpose.

And now because, 'y' is a Gaussian with mean equal to 'X' times 'w' and covariance equal to Sigma squared times I, from the previous slide, we know, that this is equal to the covariance of this outer—of 'y', which is Sigma squared 'I', plus the outer product of the mean of the vector 'y', which is equal to 'X' times 'w'. So this is the mean 'X' times 'w', and then 'w' transpose 'X' transpose, is the transpose of that mean. So we simply plug in, again, we're making this Gaussian assumption so we can plug in these values here even though we don't actually know what 'w' is. We assume that the vector 'w' has this relationship to the output, and so we can plug this function in here and still work with it. We then simplify by separating these two terms out like this, so we multiply these two matrices on the outside by this, and then separately add that same multiplication with this term here.

So here, this term is equal to this term where we get rid of this, and then this term is the same thing except for we now focus on this part here. And then again, we keep the 'ww' transpose like so. And now everything simplifies. So, for example, let's focus on this one here. We have 'X' transpose 'X' inverse times 'X' transpose 'X'. Whatever it turns out to be, this matrix product is equal to the identity matrix because they cancel each other out. And the same on this side. So this term is actually equal to 'ww' transpose, which will cancel that term. And then here, we have Sigma squared times the identity. And since Sigma squared is a scalar, we know that we can bring this term out front.

And then we get a cancellation here, so 'X' transpose times the identity is just 'X' transpose, and so we get a cancellation of 'X' transpose 'X' times the inverse of that matrix product. So when we simplify these three terms, we actually find that the covariance of the maximum likelihood solution is equal to Sigma squared times the inverse of 'X' transpose 'X'.

Okay, so let's review, again, at a high level what is it that we've shown?

So we've shown that if we make a Gaussian assumption, we assume that the vector of response is 'y', is equal to the matrix of covariates, or future vectors 'X' times the weight vector 'w' plus independent Gaussian noise with this, with variance Sigma squared. So if we make this distribution assumption on the data that we observe, where, again, notice that we've only defined a distribution on 'y'. We haven't defined any distribution on 'X'. And then we solve the least squares, or the maximum likelihood solution for 'w', they're the same solution. So we assume there's some true 'w'. We don't observe it. But we do observe 'y', which is a random variable– a random vector, according to this distribution, and then we try to do the inverse problem of learning what is 'w' using maximum likelihood. Then the expected value of our maximum likelihood solution is equal to the truth. But the variance of our—the covariance of our maximum likelihood solution is equal to Sigma squared times 'X' transpose 'X' inverse. So when using this probabilistic assumption, can we say that maximum likelihood or least squares is not going to work, or where we don't necessarily trust the maximum likelihood solution that we're going to get? So that's simply when the values and the covariance are very large. So this is a relative, you know, large versus

small is all relative to the problem being considered. But if we want to be confident about our maximum likelihood vector 'w', when the values in this covariance are very, very large, we can't confidently say that our maximum likelihood solution is close to the truth. So let's look at a method that addresses this issue.

Video 5 (12.27): Ridge Regression

Next we're going to talk about a linear regression model called, Ridge Regression.

It's a very simple modification to least squares. So we saw with least squares that the values in our solution may be huge; they may deviate wildly from the ground truth, depending on what the matrix 'X' transpose 'X' inverse looks like. So in general, when we develop a model for data, we often wish to somehow constrain the model parameters. And so in this case, that constraint might take a form, the form of penalizing values of 'w', that we consider to be too large.

So there are many models of this form. But they all have this, many of them have this general form. Where we say the optimal value for the vector 'w', is equal to the vector, that minimizes the sum of squared errors. So this is exactly like before. The sum of squared errors, of our approximation. Plus some penalty term which is equal to, Lambda, which is a nonnegative regularization parameter that we would have to set somehow. Times some function of our model variables, or our model parameters, 'w'. So this is also a positive function that penalizes values of the vector 'w'. In some way, in order to encourage properties of 'w' that we might want in advance. So let's be more specific. Let's discuss Ridge Regression. So Ridge Regression is basically the name of a model, of a particular form.

Where we use a particular form of this regularization function, 'g'. So in particular, the Ridge Regression solution uses as the regularization function 'g', the squared magnitude of the vector 'w'. So with Ridge Regression, we're trying to find the vector 'w' that minimizes the sum of squared errors. Plus this additional penalty which penalizes the magnitude of the vector 'w', that we are going to return. So clearly there's a tradeoff between the first and second terms. And this is going to be controlled by the parameter, Lambda, that we set. So let's look at some extreme cases. So for example, when Lambda decreases to zero. This second term disappears. And we're simply left with the least squares solution, because we're trying to minimize the sum of squared errors.

So as Lambda goes to zero, the Ridge Regression solution that minimizes this function goes to the least square solution. And the opposite extreme, when Lambda blows up to infinity. We simply set 'w' to the vector of zeros, because any non-negative value in 'w' is penalized so heavily. Essentially, has infinite penalty whereas, the vector of zeros here essentially is penalizing you as the sum of squared values in the vector 'y'. Sowe basically, in the limit as Lambda goes to infinity will return the vector of all zeros. Because we're penalizing a nonnegative value infinitely much. Okay, so let's look at the Ridge Regression solution. It's very straightforward, given our previous discussion.

We can solve it using exactly the same procedures as we've looked at previously, for these squares. So we have our objective function, L, which is now the sum of both the squared errors and this additional regularization term which penalizes the magnitudes of the vector 'w'. Equivalently, we can write that as this vector product, plus this additional vector product. So these two lines are equal to each other. Two different ways of writing exactly the same thing. And now the solution, again, is the point of the objective, that the vector 'w' that minimizes this objective equivalently. It's the vector 'w', at which the gradient of the objective is equal to zero. So again, we have an objective that looks like this. And we're trying to find this point here. That's the point at which the gradient is equal to zero.
So we take the gradient to this function and we find that it's equal to this. We set it equal to zero. And we solve, or try to solve for 'w'. In this case, we can solve for 'w' in closed form. And we find that the Ridge Regression solution, the vector 'w' that minimizes this objective here, is equal to Lambda 'I' plus 'X' transpose 'X'. That matrix inverse times 'X' transpose 'y'. So, this is the Ridge Regression solution.

Clearly it looks a lot like the least squares solution. Previously, we saw that as Lambda goes to zero, we recover the least square solution. In that case this term disappears, and we're left with what we recognize to be the least square solution. In the other extreme, we saw how Lambda, as it goes to infinity, returns to the zero vector, and that's clear here, as this value Lambda goes to infinity.

This matrix, the inverse of that matrix is going to be a matrix of all zeros.

And that times any vector is going to give a vector of zeros. Okay, so like least squares, Ridge Regression also has an interesting geometric interpretation that's useful for developing intuitions of what's happening. So we can see that there's a tradeoff between the squared error penalty, which is this term here. And the regularization penalty that we add on the vector 'w', which is this term here. So the way that we're going to write this is in terms of what are called level sets. So we're going to write these two functions as their own independent level sets. Meaning they're curves, where, if you evaluate the function at any point along that curve. The solution is the same exact value. So for example, if we look at Lambda times 'W' transpose 'W', which is this term here. Then any point along this circle, if we take the vector, 'w', in this case, we're assuming 'w' is a two-dimensional vector. And evaluate it anywhere along this circle. That penalty is going to have the same exact value. And that's clear here because we're basically taking a–we're taking the magnitude of 'w', squared times some scalar. And any point along the circle is going to have the same magnitude squared.

So these circles here indicate the level sets of the penalty that we have on our vector 'w'. Similarly, what we can show, and I'm not going to derive it here. But what we can show is that we can write this the sum of squared errors, term as being equal to this term plus a constant allows us to write intuitive level set for this term here. That's in, that's given in terms of the least square solution. So here we have the least squares solution. We can put that here. And then we have this matrix which takes the level sets and puts them through and distorts them so that it becomes an ellipse. And we get something like this. And so what we ultimately get, is that the final penalty is the sum of level sets for this term, plus the level set for this term. And there's a give and take between the two. And so we're going to find a

solution that's somewhere along this part here. As Lambda goes, gets bigger and bigger, these penalty terms get more and more severe. And so we're dragged this way. As Lambda gets smaller and smaller, these penalty level sets kind of disperse and the solution is dragged this way. And then, as a function of Lambda, we sweep points in between these two extremes for the output vector 'w', which in this case is two-dimensional. I think I'll talk here quickly in one slide about some preprocessing. So Ridge Regression, penalizes each dimension of 'w' equally. Because the penalty is equal to Lambda times 'W' transpose 'W'.

So each dimension of 'w' is equally penalized. However, if the scale of the dimensions of the matrix 'X' is different. So if one dimension has values that range between zero and one, in general. While another one ranges between zero and a million. But you think that those two inputs are equally important. Then Ridge Regression is going to penalize weights on the smaller dimension that takes smaller values. So as some preprocessing for Ridge Regression, you notice that we didn't have this problem with least squares. Because we didn't put any restrictions on what the weight vector 'w' was equal to. But here, because we want the weight vector to have roughly equal dimensions, we preprocess the data. It's a very simple procedure. We take the matrix 'X'. In this case, we could think of taking the matrix 'X' and subtracting the mean off of each column, which is what this is saying. So take each 'x' in, in j^{th} dimension. Subtract the global mean of that dimension, and then divide by the variance of that dimension.

And similarly, we take the outputs 'y', and subtract off the mean. And then in this case, we're able to avoid having to add the dimension of ones. This is something that's nice that we don't have to worry about anymore. Because we've standardized and made everything zero mean. And also, we don't run into the issue of having different dimensions of our inputs be penalized differently and arbitrarily because of our Ridge Regression penalty.

Video 6 (04.33): More Analysis of Ridge Regression I

So, next I want to discuss some more analysis, go into more depth into some analysis of Ridge Regression, similar to the analysis that I did for least squares. So the solution to least squares and Ridge Regression are clearly very similar. Here on the left we see the least square solution that we've been discussing. And on the right we see the Ridge Regression solution. The only difference is in this additional term here, Lambda. And so, let's discuss what this Lambda is doing in a bit more detail. And so, the way that, we can compare these two, are using properties of–using techniques from linear algebra and also from probability in order to, better understand what Ridge Regression is doing, and how it relates and compares to what least squares is doing.

So, the way that we're going to do this, is using something called the singular value decomposition. And so, let's quickly go through a one-slide review of the singular value decomposition. So we know that we can write any 'n' by 'd' matrix 'X', where we are assuming that 'n' is greater than 'd', so we have more observations than dimensions in our problem. As this matrix product, 'US' times 'V' transposed, where

these are all three matrices, 'U', is an 'n' by 'd' matrix. We assume that it's orthonormal in the columns. What this means is that you transpose 'U' as the identity matrix. So each column of 'U' is unit length and is orthogonal to every other column in that same matrix. The matrix 'S', is a 'd' by 'd' non-negative diagonal matrix. So, the i^{th} value along the diagonal of 'S' is nonnegative, it can be zero. And the off diagonal values are all equal to zero. And the matrix 'V', is 'd' by 'd' and orthonormal. And so because it's square, we can say that 'V' transpose 'V' is also equal to 'V', 'V' transpose. And both of those are equal to the identity matrix. So the columns, of 'V' are orthonormal and the rows of 'V' are also orthonormal.

Okay, so from this we have some immediate equalities. If we have 'X' transpose 'X', which is of interest because it's in both the least squares and the Ridge Regression solution. So we have, 'X' transpose 'X' is equal to this matrix product, where we're simply replacing this singular value decomposition for 'X'. And then because we use the property that the transpose of a matrix product is equal to the transpose of each individual matrix in reverse order. So we get, 'U' transpose times 'U' here, which cancels out, because of the assumption. Because we're using a singular value decomposition. We have, 'S' transpose, which is 'S', because it's diagonal, times 'S'. So we have 'S' squared. And then, on the outside of this we have 'V' transpose, transpose, which is just V. And then 'V' transpose on the right. So, this matrix product is equal to this. And similarly, 'XX' transpose is equal to 'US' squared 'U' transpose.

So if we assume that the matrix 'X' is full rank– meaning that all diagonal elements of the matrix 'S' are strictly nonnegative, so greater than zero, then we also can write the inverse of 'X' transpose 'X' as follows. So we know 'X' transpose 'X' is equal to 'VS' squared 'V' transpose. The inverse of a matrix is–a matrix with this form where, 'V' is a square orthonormal matrix is simply taking the diagonal middle matrix and inverting it. And if you want a quick proof of that, you simply can take 'X' transpose 'X' times its inverse and replace that with its singular value decomposition and finally you get the identity matrix. So that's how you can show, that the inverse is equal to 'VS' to the negative two 'V' transpose.

Video 7 (15.16): More Analysis of Ridge Regression II

Okay so, let's use the SVD to analyze the least square solution. So when we looked at the variance of the least square solution, we saw that it was equal to this matrix–Sigma squared times the inverse of 'X' transpose 'X'. We now replace 'X' transpose 'X' inverse, with what we obtained from its singular value decomposition. And we notice that this inverse of 'S', is going to become huge when we have singular values that are very, very small. So when the the matrix 'X' has singular values that are close to zero, when we square that they become even closer to zero, if those values are less than one. And then when we invert that, we have very large values in this diagonal matrix. And so we can say that the least square solution is not going to be stable, when the singular values of the matrix 'X' when there are some singular values that are small. And so this is something that we can actually calculate, because we have the matrix 'X'. It's part of our dataset.

So we can construct it, we can, in our favorite coding platform, call the singular value decomposition function to get its singular values, and look at them, and see, are there singular values that are close to

zero. If there are, then there's a good chance that our solution to the least squares problem is not going to be stable. It's going to have values that aren't close to the ground truth. And so when does something like this happen? Intuitively, we can think of when there are columns of the matrix 'X' that are very highly correlated with each other, then we're going to have singular values that are very small.

So, for example, we might think that we're making many measurements of our data, and we're returning these high-dimensional vectors 'x' that are going to give us many things to help predict the output. But, if the different dimensions of those measurements are very highly correlated, then we're not actually helping ourselves solve the problem. So if we want to see how having small singular values are going to affect our prediction, we can simply look at how we make predictions for least squares. We take our new vector 'x', the dot product, with our least square solution, and say that that's our prediction of the corresponding response, or output. If we replace the least square solution with the singular value representation, we can see that, when there are small singular values, the inverse of that matrix of singular values will be very huge. And, depending on the value of the vector 'x', we could also have predictions that are wide off the mark, depending on how the vector 'x' correlates with those singular vectors, that have corresponding small singular values.

Okay, so in the next few slides I want to look more closely at the mathematical relationship between Ridge Regression and least squares. So, first, let's recall, that if we have two symmetric matrices, 'A' and 'B', they're both symmetric matrices, then the inverse of 'A' times 'B' can be calculated by taking the inverse of each of those matrices separately and then reversing the order. So 'AB' inverse is equal to, 'B' inverse times 'A' inverse. So now, if we want to look at the Ridge Regression solution, let's kind of work through it, and manipulate it in a certain way to relate it to the least square solution. So here in the first row, we have the Ridge Regression solution. This is what we solved on a previous slide. Now, let's do something a bit simplistic, and let's multiply and divide by the same thing.

So let's wedge in this part here, the matrix 'X' transpose 'X' times the inverse of 'X' transpose 'X'. So this matrix product is the identity. And so we haven't actually changed the solution by going from the first line to the second line, by adding a matrix times the inverse of itself. However, now we can look at this matrix-vector product and recognize this as the least squares solution. So let's write–instead of writing it in long form, let's just simply replace that with W_{LS}, the least square solution. Similarly here, let's do something a bit simplistic and write this, Lambda inverse–Lambda times the identity plus 'X' transpose 'X', as a product of 'X' transpose 'X', that matrix, by the matrix Lambda times 'X' transpose inverse plus the identity. So if we multiply through 'X' transpose 'X', multiply that through this term, here, we see that this term is, simply equal to this term.

However, here's where we're going to use the property at the top, where the inverse of these two symmetric matrices multiplied with each other is equal to the inverse of this matrix, times the inverse of this matrix. And so that's what I've written here. The inverse of this second matrix, which is a symmetric matrix, times the inverse of the first matrix, which is also a symmetric matrix, is equal to the inverse of the product of those two matrices. But now doing this we notice, that these two matrices cancel out.

And we're finally left with this relationship–that the Ridge Regression solution is equal to a matrix times the least square solution.

So they stand in this very specific relationship to each other, where we take the least square solution, and we multiply it by whatever this matrix turns out to be. It's calculated using things that we know so we can calculate it. And we can simply manipulate the least square solution, according to this vector, in order to get the Ridge Regression solution. And so immediately, we can see that we should be able to expect the Ridge Regression solution, to have smaller magnitude than the least square solution. This is the identity. And then we're adding something that's nonnegative to it, that has nonnegative singular values. And so you can imagine this as dividing by something greater than one. So we're going to be shrinking this. So I'll discuss this more specifically in a moment.

Okay, so let's now continue our this line of analysis where we're now going to replace the matrix 'X' by its singular value decomposition. And so, again, we can write a matrix 'X' as a product of these three matrices that have the properties discussed on the previous slide. And, therefore, this matrix 'X' transpose 'X' inverse can be written in this way. And so now if we continue from the previous slide, where this first line is the last line of the previous slide, and now replace this matrix 'X' transpose 'X' inverse by the matrices that we obtained from the singular value decomposition of 'X', we get this sequence.

So we get this term, by replacing this matrix, with this matrix. And now, because 'V' is a square orthonormal matrix, and because 'I' and Lambda 'S' inverse squared are both diagonal matrices, because those two things are true, we can write this matrix in this way. And so we can simply pull out the 'V's' on both sides, and then invert this matrix Lambda 'S' inverse squared plus the identity. If we want to prove why that's true, simply multiply this matrix by the matrix that we obtained from the term in between the brackets, and notice that the result is the identity matrix. And that's how you can show that it's true. Okay, so now let's call this inner matrix here, 'M'. Let's just define 'M' to be equal to the identity plus Lambda Sigma inverse squared. And then if we do that, we notice that 'M' is a diagonal matrix, and the i^{th} value on the diagonal of 'M' is equal to the i singular value squared, divided by Lambda plus the i^{th} singular value squared.

So we get a matrix 'M' that has diagonal values like this. And now let's take the solution of least squares, and also use the singular value decomposition of 'x', and we'll be able to show in a straightforward way that the least square solution can be written in this way, where 'U', 'S' and 'V' are calculated from the singular value decomposition of 'x'. So this is something that you could show very easily. And then we see that the Ridge Regression solution, knowing that it's equal to 'VM' 'V' transpose times the least squared solution, which is equal to this, plugging all of those things together in, and we find that the Ridge Regression solution can be written in this way–'V' times a matrix that we define 'S' Lambda inverse times 'U' transpose times 'y' 'UV', 'U' and 'V' are both the leftmost and rightmost matrices of the singular value decomposition of 'x', 'S' Lambda inverse is calculated in this way. It's equal to essentially 'M', where the numerator is now, doesn't have the squared term. So, 'S' Lambda inverse is equal to a

diagonal matrix where the i^{th} value along the diagonal is equal to the 'i' singular value divided by Lambda plus the 'i' singular value squared. And so, remember that as Lambda goes to zero, we get the least square solution back. And so we can kind of see from this matrix what's going on. Imagine that the d^{th} singular value is extremely small, it's very close to zero. Then when Lambda is equal to zero, we have that–this term is one divided by the d^{th} singular value, which is a massive number, which was the issue that we were running into, previously.

By adding Lambda, we essentially put a floor on that. As the singular values go to zero for a Lambda greater than zero, this term actually goes to zero instead of infinity. When Lambda is equal to zero, as 'S_{dd}' goes to zero, this term goes to infinity. But now if Lambda is greater than zero, then this term goes to zero as 'S_{dd}' goes to zero, because this Lambda is kind of putting a floor on the smallest value that this denominator can take. So Lambda is essentially killing off these small singular values. And so here's a mathematical way that we can understand exactly how it's doing that. So the third relationship I want to look at briefly is how Ridge Regression relates to least squares, as a least square's problem.

So we can actually write Ridge Regression as a special case of least squares. And by doing this, what we do is, we take the original problem where we say that 'y', the vector of all the responses and-dimensional vector 'y', is approximately equal to the matrix 'X', that we construct by putting the inputs along the rows of 'x', times the vector 'w'. And then we extend this problem by adding 'd' or 'd' plus 1, more terms to it. So, I'll assume that we've standardized our data so we don't have the dimension of ones.

I'll assume that we're just simply looking at a 'd' column matrix here. And so if we take the original 'y', and then we attach these zeros, extend it by d dimensions and put zero below it, and then take the matrix 'X' and extend it by d dimensions along the first dimension, and then attach a diagonal matrix at the bottom of 'X' that has the value of square root of Lambda along the diagonal, and then solve the least squares solution for 'w', for this problem. So what we're calling this vector is 'y hat'. We'll call this matrix 'X hat'. And then 'w' is left unchanged. And then solve the least squares solution to the problem 'y hat' minus 'X hat' 'w' transpose times itself, So the sum of squared errors of this problem, then if we multiply that all through, what we find is that we're equivalently trying to minimize over 'w', this term, which is, if we rewrite it in another way, simply the original Ridge Regression term. So in a sense, Ridge Regression is almost like an augmented least squares problem.

Video 8 (06.04): The Regularization Parameter

So finally, I just want to briefly talk about selecting the value Lambda. So, we saw on the two extremes as Lambda goes to infinity, our Ridge Regression solution will be a vector of zeroes. And the other extreme as Lambda goes to zero, we get the least squares solution back. So, how do we select Lambda?

And I'll talk in more detail about these types of problems, later. Here, I just want to give a very simple heuristic with not so much justification, something that you can calculate using all of the data that you have very easily, by basically giving a trace of the solution as a function of Lambda. So, this isn't even necessarily a way to pick Lambda, it's a way to understand how Lambda changes our least squares solution or our regularized least squares Ridge Regression solution. So, this is something called the degrees of freedom. So on this case, what we're going to do is we're going to, as a function of Lambda, calculate the number of degrees of freedom in our solution. And so, this is equal to the trace of this matrix here, which is equal–equivalently equal to the sum of–among–the sum of the singular–for each of the 'd' the singular value, the 'i' singular value squared, divided by Lambda plus the 'i' singular value squared, and then sum that over the singular values.

So, if you'll recall, from a previous slide, that's simply equal to the trace of this matrix 'M'. This number of degrees of freedom is simply for plotting purposes. And so, what you'll see is that in the extreme, as the number of degrees of freedom equals zero, that's equivalent to saying Lambda is equal to infinity. Because then, you have this term is equal to zero for each 'i' and then the sum over 'i' is equal to zero. So, the number of degrees of freedom as Lambda goes to infinity is equal to zero. So that's one extreme. And then, as Lambda goes to zero, so in the limit, as Lambda goes to zero, this term is equal to one. And so, the sum from 'i' equals one to 'd', of one, is equal to 'd'. And so then, in the other extreme as Lambda goes to zero, the number of degrees of freedom is equal to 'd', which is the-dimensionality of the problem. So, in this example on the right, we have a problem that's originally eight dimensions, meaning that the vector 'x' is eight-dimensional. So, we have eight different inputs. For example, age could be an input, weight could be an input, among other possible measurements that we make, as inputs. And so, when we plot the–and so what we're showing here is a plot of the weight vector 'w' as a function of the degrees of freedom. So for example, let's imagine that age is the first dimension of our data 'x'. So, the weight for 'w_1' is the weight that we associate with the age of the person in making our prediction of some output.

And so, what this is showing is as a function of degrees of freedom, what is the weight for the first Ridge Regression solution for the first dimension, which corresponds to age in this example. So again, what we're doing here is we're starting Lambda at zero, and we're letting Lambda blow up to infinity as we move along the axis to the left. So, as the number of degrees of freedom goes to zero, Lambda is going to infinity. And so what we have here, for example, in this slice is one particular value of Lambda that gives the number of degrees of freedom roughly, five. So, this corresponds to a particular value of Lambda. And, what you can then do is read off the eight values in the vector 'w' for this particular value of Lambda. And so, what you can see here is that very clearly, the solution fundamentally changes, as Lambda goes from zero to infinity. For example, for a particular Lambda here, then the weight that we associate with age is positive.

So for this Lambda, this value of Lambda, we're saying that as age increases, we expect whatever response we're measuring to increase, whereas as we decrease Lambda, say here towards zero–so this would be a large value, this would be a small value, then the age is a negative weight. And so in this

case, we're saying that increasing age decreases the corresponding output.

So, we're saying that the age of the person in this example fundamentally changes, the relationship of the age, the output fundamentally changes as two different–for these two different values of Lambda. And so, what's making up the slack is the weights that we associate with everything else. And so here, as you can see, as Lambda goes to infinity, all the weights are going to zero. We get a zero vector. And, as Lambda goes to zero, in this case, we get degrees of freedom equal to eight because we have eight covariates. Then, the weights correspond to the least squares solution, and then we have a sweep of all values in between as a function of Lambda.

Video 9 (02.52): Regression With/Without Regularization

Okay! So, let's continue our discussion on linear regression using least squares and Ridge Regression techniques. So, to refresh your memories, we're working with a dataset that is of the form that comes in pairs. So, we have 'x_i', 'y_i', for 'i' equals one to n, so n observations. The data 'x' is in 'R^d', and the response 'y' is in 'R'. And, we're going to assume that we don't need the bias dimension in 'x', so we have standardized the data in some way so that we can work without the bias. Really, in the discussion that I'm going to have in this lecture whether the bias is there or not, it doesn't really impact the math, but it will impact the results that you have. So, I'm more interested in discussing the math techniques here. So, we're going to use a model of the form 'y' approximately equal to some function of its covariate vector 'x', and some model parameter vector 'w'. So, this is very generic. And, in particular, we're going to focus on the function where, 'f' evaluated at an 'x' and 'w' pair is just the dot product between 'x' and 'w'. So we're making predictions that 'y' should be approximately equal to the dot product between 'x' and 'w'. So, this is the function that we want to use. And now, we want to learn this function. We want to– by that we mean we want to learn the vector 'w'. And so, we do this by minimizing some sort of a 'loss' function that tells us what are good values of 'w' and what are bad values of 'w'. And so we've already discussed, a few 'loss' functions. The one that we're going to talk about now, or focus on is the Ridge Regression–corresponds to the Ridge Regression 'loss' function, also called L_2 regularized least squares. So, we have as the first term the sum of the squared errors. And as the second term, we add a penalty of the squared magnitude of the vector 'w' times some regularization parameter Lambda that we set. And so, we focused on two examples, one is where, Lambda is equal to zero in which case we only have the sum of squared errors, that's least squares. And when, Lambda is greater than zero, solving or minimizing this objective function gives the Ridge Regression solution.

Video 10 (05.47): Bias-Variance Trade-Off I

The next part of the lecture, I want to discuss something called the bias-variance trade-off. I'm going to discuss it in the context of comparing least squares with Ridge Regression. It's more general than that but, I mostly want to focus in some detail on how this general concept of trading-off between the bias

and the variance of a learning technique applies to what we've been talking about thus far. So, in order to give a bias-variance analysis, we are going to go one step further. And, we're going to continue with our hypothesis that we discussed last time of a generative model of the response vector 'y'. So, remember, from previously, the vector 'y' was now an n-dimensional vector, containing all the responses from our dataset. And we modeled that vector as being a Multivariate Gaussian random variable, where the mean is equal to the matrix 'X', where we have taken each covariate vector and put it along a row of the matrix 'X' and times a regression coefficient vector 'w'. And, we add to that zero mean Gaussian noise, having the covariance, Sigma squared 'i'. And so we believe that there's some true vector 'w', generating our data. We hypothesized this, but we don't know what it is. So we saw previously, how using least squares by minimizing the sum of the squared errors, the least squares solution, which has this form is unbiased, meaning that the expected value of our least squares solution under this hypothesis is equal to the true vector 'w'. However, the variance of our least squares solution can be very large depending on whether this inverse matrix has very large values. By contrast, we saw that the Ridge Regression solution is equal to this, where we've simply added a term, a value to the diagonal of this matrix 'X' transpose 'X' before inverting it. And we didn't show, but using exactly the same procedures as previously, we can show that the expected Ridge Regression solution, under this model hypothesis is equal to this. And, the variance of the Ridge Regression solution is equal to this, where we defined the matrix 'Z' to be like this. So, these are just terms that we can calculate. We have the matrix 'X'. We defined Lambda.

So, we can calculate this term, with what we have. We can't calculate this term, because we don't have the true vector 'w'. But, we know that our expected ridge solution has—relates to the true vector 'w' by pre-multiplying by this matrix. And, as a sanity check, we can let Lambda go to zero. And we would realize that then, the inverse of this matrix would cancel with this matrix. And so, we would have 'w'. And, similarly, letting lambda go to zero will cause the variance to revert to the least square variance.

Okay! So, the expectation in the covariance of the least squares and the Ridge Regression solutions, they give us some insight into how well we can hope to learn the true underlying vector 'w'. In the important case—so this is an important caveat in the case, where our modeling assumption is correct, meaning the case where this is the correct model assumption. So even though that's likely not going to be the case, it may seem reasonable to us, and so we can based on the assumption, perform the following analysis.

So we saw, how least squares is nice because it's unbiased, the expected solution is the truth, but, it potentially has a very high variance. Whereas the Ridge Regression solution is biased, so it's not—we don't expect it to be the truth, but it potentially has much lower variance than the least squares solution. So, which is preferable? So, in order to say which is preferable, we have to define some sort of a measure, as usual, of a quality. So, how good are each solutions? In order to say that we need to say what we're trying to do.

So, ultimately, what we really care about is how well our solution for 'w' is going to generalize to new data. So that's what we're going to, you know, ultimately do with least squares. We're going to learn the

least squares vector 'w', and then we're going to predict new data. So, for new incoming 'x_0', we're going to predict the response 'y_0', as the dot product between 'x_0' and the least squares solution. And similarly, with least—with Ridge Regression, we're going to replace the least squares solution with the Ridge Regression solution. So, this is what we care about, how well are we going to do with future data. So, we should try to come with our measure of quality that takes this into account.

Video 11 (14.34): Bias-Variance Trade-Off II

So previously, we saw that by minimizing the sum of squared errors, there was a relationship to maximum likelihood for the Gaussian model. So Gaussian, somehow, takes squared error into account when optimizing. So, if this is the model assumption that we're going to make for least squares and Ridge Regression, then the performance that we use on new data to measure their quality of these solutions should also take into consideration the squared error. So, that's what we're going to do. When we calculate the quality of the least squares solution versus the Ridge Regression solution, we're going to use this measure to do so. So, what this is saying is we want to calculate the expected squared error of our prediction of 'y_0', given 'x_0', and given all of the previous data that we've seen. So what we're integrating out here is the vector 'y'. So that's what I have written here. So this expectation can be written so, where we take the function that we're interested in and replace it here, and then we integrate over both the distribution on 'y', given 'X' and 'w'. So for a particular 'w', we calculate the distribution on 'y' and also the distribution on the new response, given the true but unknown vector 'w' and given the covariates, the new covariate vector 'x_0'. So, it's a bit weird the way I've written it here, but ' hat 'w'. is either the least squares or Ridge Regression solution, and we remember that both of those solutions are dependent on the vector 'y' and the matrix 'X' of covariates that are associated with 'y'. So when we're integrating out 'y' that integral is going to appear is going to impact how 'y' functions in this vector ' hat 'w'. So what we're really integrating out is all response variables, both what we've seen in the past and also what we're going to see in the future. And we're going to see how this expected squared error of our prediction changes, as a function of the data that we have, the covariates that we have 'x' and 'x_0', and also some true but under—but unknown—underlying regression coefficient vector. So finally, just to help with potentially confusing integral—in words what we can say that we are calculating here—this is just conceptually—we can say that imagine that I know—I have a data set of covariate vectors 'x'. I know those, and I have some new covariate vector 'x_0'. And I assume that there's some true underlying vector 'w'. But, I don't know what it is. I, then, generate the responses for my test set 'x', according to this hypothesis, this distribution which, I hypothesized. I, then, approximate 'w' using either least squares or Ridge Regression. And then I predict, a new 'y_0' for a new 'x_0' using either the least squares or the Ridge Regression solution. But, I don't get to see 'y' and I don't get to see 'y_0'.

What is the expected squared error of my prediction? So, it sounds confusing but this is something we can actually calculate. So let's go ahead and calculate this. So in these calculations, I'm implying that we're conditioning on 'x' and 'x_0'. And so, to calculate this expectation we simply multiply this term out through and, we then, by the linearity of expectation, can bring the expectation of those three terms

that you get by squaring these—this difference into the expectation of those three terms individually. And so, that's where this first expectation is coming from and the third one. And then, this second expectation originally should be of the product 'y_0' and ' hat 'w'. But, we can say that the expectation, given 'x' and 'x_0' of 'y_0' and ' hat 'w', is equal to the product of the expectation of 'y0' times the expectation of ' hat 'w'.

So, why can we do this?

Because again, remember what we're assuming here. We're assuming that we have the data 'x_0' and 'X', and conditioning on a ground truth 'w', we've made an independence assumption, not an *iid*. assumption, an independence assumption that the responses are independent of each other. So, if you tell me what 'x_1' is and what 'w' is, then 'y_1' is independent of 'y_2', given 'x_2' and 'w'. So let's also review, some facts that the expectation of the outer product of a vector is equal to the variance of that vector times the outer product of the expectation of that vector. So, this is something we discussed previously. And similarly, the expectation of 'y_0' squared, which we again assume that 'y_0' is a Gaussian with mean 'x_0' transpose 'w' Sigma squared. So, therefore, under this model hypothesis the expectation of 'y_0' squared is going to be equal to Sigma squared, plus the mean squared, and so that's what we have here. And so plugging in these two values into this equality, we have that the expected squared error on my new observation, given 'x_0' and given the previous covariates formed in the matrix 'X' and given some ground truth 'w', is equal to this first line. And so what have I done here, I have taken this term and replaced it with this term, which is what gives me these two terms. And here, I've taken 'y_0' and replaced it with its mean, because of the Gaussian assumption of how 'y_0' is generated, and then I've left this term here for now. And so notice that we can simplify this first line, as the second line by taking these three terms and writing them in this quadratic form. And so now, we just need to input the variance of our either the least squares or the Ridge Regression solution, and the expected value of the least squares or the Ridge Regression solution, depending on whether ' hat 'w'. is least squares.

We're using least squares or Ridge Regression.

Okay. So, where are we so far?

To review, we've shown that if we make the hypothesis that we generate a data set of N observations, according to this model, where we have our covariate matrix, where we take each corresponding covariate vector 'x_i' and put them along the rows of the matrix 'X', and some true but unobserved covariate, coefficient vector 'w'. And then, we generate 'y'—I'm sorry, there should be an identity matrix here. We generate 'y' from a Gaussian with this mean and with this covariance. And we also generate a new 'y' from the same exact model, except for restricted to a new observation. We then, approximate the true but unobserved coefficient vector 'w' with ' hat 'w', according to some algorithm, either least-squares or Ridge Regression or some other algorithm. Then, what we can say is that the expected squared error on our new observation, given the covariates associated with that new observation and given all of the covariates of the tests of the training set that we used to learn ' hat 'w', is equal to the

noise of the original problem. So we can never get rid of this noise.

It's equal to the noise of the observation, plus these two terms. And these two terms, correspond to our uncertainty about the ' hat 'w'. and its relationship to the true underlying 'w'. So, it corresponds to our uncertainty of how, for example, the least squares solution relates to the true solution, or the Ridge Regression solution relates to the true solution, or the solution found by some other algorithm relates to the true solution. So the first term is the squared bias. So notice that this term factors in how our expected solution differs from the true solution, or the true vector that we're approximating. So, for example, with least squares the expected—the expectation of the least squares solution is equal to the ground truth, and so this term will be equal to zero. Whereas for Ridge Regression because we have some bias the difference between the ground truth and our approximation of it, using Ridge Regression is nonzero, because it's a biased approximation. And so, this term will be nonzero and positive because of this form. The second term then, accounts for the variance of our solution.

So, for example, we would insert the variance of our least squares solution or the variance of the Ridge Regression solution here, and then take this quadratic—solve this quadratic term, and then add those two together. So it's very clear from this derivation that how well we expect to do on new data, given the algorithm that we used to learn to approximate 'w' with some old data is going to take into account these three terms. The first term is simply the model noise, which we can never get rid of. It's the noise of the sensor that we're using or something like that. The second term is how close we expect our solution to be to the truth, so how biased is our solution. And the third term is how much variance is there in our solution.

So a solution that has low bias, but very high variance, so no bias or—and very high variance, which least squares solution can have. We will have this term equal zero but this term might be massive. Whereas another solution like the Ridge Regression solution, which has some bias but, potentially much smaller variance can then trade-off some nonnegative, some positive value here for a very big reduction of the value here from the least squares solution. So that's the Bias-variance Trade-off. And so you could see how, with the Ridge Regression solution, hypothetically, it's possible that whereas we give up something by adding some bias, we can reduce the variance so much that the sum of the two is much less than the original variance term for least squares. And so, what these all end up being, these values, all depends on our data. So for the expectation, we can input this for least squares and the variance we can input this. Notice for the variance term we have everything necessary to calculate it. This is the data that we have. Whereas for Ridge Regression, this is our bias term and this is our variance term.

So we can again calculate the variance. So we can compare the variance of the Ridge Regression solution to the least squares solution. However, for the Ridge Regression solution we don't know what 'w' is, and so even though we can calculate this entire term for least squares, because 'w' cancels here, for Ridge Regression we still have 'w' in this term. And so, the bias really depends on what the value of 'w' is. So we can, make some statements potentially about values for 'w' that—where it works out very well or values of 'w', where this term blows up. And also, the relationship of our solution to the new

observation that we're trying to make a prediction for, is factored in by using 'x_0' here.

Video 12 (03.49): Cross-Validation

High-variance tradeoff is a nice theoretical way of motivating, why adding some bias to our model might reduce the variance or might result in a model that is somehow better or preferable to a model with no bias, depending on how their variances relate to each other. But, we saw with ridge aggression that we need the true underlying vector 'w', in order to even be able to calculate the bias term. So, that's often the case that we need access to something that we don't have access to, in order to actually calculate the sum of the bias and the variance of our model.

And so, let's look at a simpler and much more practical method for doing either model selection or determining a particular parameter. So for example, with ridge aggression, we have this parameter, Lambda, which penalizes the squared magnitude of our coefficient vector 'w'. We don't know what to set this to. So, how do we decide what to set Lambda to? One way is to use, what I'll discuss called cross-validation, where we pick a value for Lambda.

We then, randomly partition the data set into K roughly equal groups where, K is the number that we decide, for example 10. We then, learn the model on K minus one of those groups, and try to predict the held-out K^{th} group. We then do this K times, where each time we hold out, one of the K groups and then, learn the model on the other K minus one groups. And then, predict on the held-out group. So every group gets held out one time. We evaluate the performance for each held-out group, and then the cumulative performance is just the sum of the performances on each held-out group. So, intuitively, what this is doing is this is like saying that instead of having a data set of size, for example 100, we pretend that we have really a data set of size 90. We learn our model. And then, we treat the remaining 10 observations that we've held out, as if they're a new set of 10 observations that we want to somehow make predictions for. So we treat those, as if we don't know what the ground truth is, and see, because we actually have the ground truth, how well would we have done if that set, held-out set of 10 observations really were something we tried to predict. And so, intuitively it makes sense that cross-validation is something that can really give us a sense of how our model will perform on unseen data.

So, we do K-fold cross-validation for a particular parameter setting of Lambda. And then, we sweep over multiple settings of Lambda, and we get our performance measure for each one. And then, we simply pick the value of Lambda that performs the best. And this, we can think of as being the value of Lambda that will translate to the new, unseen test data that we observe in the future, because the way that we calculated the performance measure, treated the data that we do have as if it was test data that we didn't know the ground truth for.

Video 13 (11.54): Bayes Rule

Okay. So in the next part of this lecture, I want to discuss something new called, Bayes Rule. This is a new technique that we haven't discussed thus far. It's very general, very useful for quantifying our uncertainty in model parameters. And I want to discuss it in the context of the linear regression problems that we've been discussing so far.

So let's motivate, Bayes Rule through Ridge Regression. So we've discussed the Ridge Regression objective function, where we saw that we were trying to minimize an objective of this form, where we summed squared errors. And then–this is a measure of basically of how well a particular value of the vector 'w' can predict the data that we have. So by minimizing this, we're finding a 'w' that performs very well on our data. And then this second term, Lambda times the squared magnitude of 'w', in a sense, is like a prior belief about what 'w' should be. So in a sense it constrains 'w'. It gives us-it imposes a prior belief on, in this case, the idea that the magnitudes of 'w' shouldn't be too large. So this is a prior belief that we're using.

And so, the question is, is there a mathematical way to formalize this idea of imposing a prior belief on a model parameter or a model variable.

And the answer is, using, or one answer is, using the probability-framework of probability and Bayes Rule, we can make some coherent rigorous statements about what's going on. And so before I return to, extending the Ridge Regression solution into a fully Bayesian realm, I want to quickly, very briefly, give a review of probability, what statements we make with probabilities, and also what we can do with probabilities. So the next two slides, I think of as being mostly review. But I want to go through them to set the context for what we'll discuss later. So, imagine that we have two events 'A' and 'B' that may or may not be related. So for example, 'A' could be the event that it's raining outside and 'B' the event that the ground is wet. We can talk about probabilities of these events. So we can say that 'P' of 'A', the probability of 'A' is the probability that it's raining outside. And the 'P' of 'B', the probability of 'B' is the probability that the ground is wet. We can also talk about conditional probabilities. So these are probabilities of events, given knowledge of other events.

So the probability of 'A' given 'B', which is what this is saying, is the probability that it's raining, given the fact that I know that it's wet–so the probability of 'A' given 'B'. And the reverse of that is the probability of 'B' given 'A', where we're saying that the probability that the ground is wet, given that it's raining outside. So these are conditional probabilities. Finally, we can talk about joint probabilities. So the probability of 'A' and 'B', which is the probability that both it is raining and the ground is wet.

So the joint probability with this notation is the probability that both 'A' and 'B' occur. So now, let's look at how these probabilities all relate to each other. So there's some, simple rules for moving from one probability to another. For example, the joint probability of 'A' and 'B' is equal to the product of the conditional probability of 'A' given 'B', times the probability of 'B'. Or, we can reverse it and say that it's equal to the conditional probability of 'B' given 'A', times the probability of 'A'. Similarly, we can represent the probability of 'A' as a marginalization, where we sum over all of the possible values that

another event can take within a joint probability. So the probability of 'A' can also be represented as the joint probability of 'A' and 'B', where we simply sum over all possible values of 'B', getting rid of any impact that it could have. And so, in this case we call the probability of 'A' the marginal probability.

So, you can call this two different things. You can say it's simply the probability of 'A' or, you could say it's the marginal probability of 'A'. When you add that extra word 'marginal', you're implying that there's some additional event that you're integrating or summing out. And similarly, with 'B' we could say the marginal probability of 'B' is equal to the joint probability of 'A' and 'B', where we summed or integrated over all possible values of 'A'. Where-when 'A' is discrete, it can take a finite number of values we sum. When 'A' is continuous, the sum becomes an integral. Okay. So, these three simple equations allow us to say some interesting things and even derive Bayes Rule. So for example, the first line alone is enough to say the following two lines–the first equality in the following two lines.

So for example, these three terms are equal to each other. If we focus on this equality here, using simple algebra, we can say that the conditional probability of 'A' given 'B' is equal to the product of the conditional probability of 'B' given 'A', times the probability of 'A', divided by the probability of 'B'. And, exactly the same down here, except 'A' and 'B' is reversed. And also, because of the representation of the probability of 'A' and 'B' as marginal probabilities, we can say that the probability of 'B' is a marginal probability of the joint probability of 'A' and 'B', where we've integrated or summed over 'A'. And then the joint probability again, up here, can be written as a conditional probability, times a marginal probability.

And so we have these-this sum in the denominator. And so, simply notice that for both of these terms the denominator is equal to the sum of the numerator over all possible values of 'A'. And so what this is doing is, it's simply normalizing the numerator, so that it's a probability distribution. And so this is known as Bayes Rule. So Bayes Rule is nice because in practice it allows us to quantify what we don't know, using probability distributions. And so imagine that we wanted to say something about the probability of 'B', given knowledge that the event 'A' happened. So Bayes Rule says that the probability of the event 'B', after knowing the event 'A', in this case we would call that the posterior probability because it's the probability after obtaining some data 'A', is equal to the probability of 'A' given 'B'. In this case, this is the likelihood.

So, it's the likelihood of observing what we saw, given a certain setting for 'B', times a prior on 'B'–so a probability of 'B' that we have to define a priori-so this is a prior probability on the event 'B', divided by the marginal probability of the event 'A'.

And so notice that when we think about what we do and don't know that these manipulations that I've discussed on the previous slide, which was just simply conditional and joint and marginal probabilities, take on new meaning. So we can now, use these terms of posterior, likelihood, and prior, whereas in reality the probability of 'B' given 'A' is just a conditional probability, and the probability of 'A' given 'B' is also just a conditional probability. But, the significance is different.

The probability of 'B' given 'A' is our posterior belief of the unknown 'B', given the observation 'A'. Whereas the probability of 'A' given 'B' is the likelihood of seeing what we saw, given a certain setting for 'B'. So they're both conditional probabilities. But we think of them in different ways. So we've discussed Bayes Rule in the context of discrete variables. It simplifies–it can be extended to continuous random variables and continuous parameters in a very simple and straightforward way. So in this case, instead of probabilities, we'll talk about densities. So, let's think of an example where, Theta is a continuous-valued model parameter, and 'X' is some data that we possess.

So then, using Bayes Rule, the posterior probability of the model parameter or the model variable, given the data that we've observed, is equal to the likelihood of the data, given the model variable times the prior probability of the model variable, divided by the integral of the numerator over all possible values of the model variable, which is just the marginal probability of the data. So in this case, using Bayes Rule for model learning, we possess–we usually–we possess the knowledge of everything necessary, in principle. So for example, the probability of the data, given the model variable, which is the likelihood, is known from the model definition.

So we define this term, here. We define some generative distribution on our data, given some model with its unknown variables. And then, the prior probability on the model variables is also something that we're going to define. So what we want is the posterior probability of all model variables. And we can write that in terms of things that we know. And then the only question that remains is whether we can calculate this integral and the denominator, in order to give a closed form, analytic expression for the posterior probability.

Video 14 (06.59): Coin Flipping Example

Let's look at one simple example, where we're trying to learn the bias of a coin. So, imagine that we have a coin with bias Pi towards 'heads'. What this means is that the probability of 'heads' is Pi and the probability of 'tails' is one minus Pi, if we flip this coin. And, we encode 'heads' as a one for the purpose of modelling and we code 'tails' as a zero.

So, we flip the coin many times, n times independently, so what happens on any flip doesn't impact any other flip. And, each time we flip it, it'll come up 'heads' in which case we write a one, or 'tails' in which case we write a zero. And, it'll come up 'heads' with probability Pi. And now, we've observed a sequence of n flips and we want to learn something about what's the underlying bias of the coin.

So first, by making the assumption that the flips are all independent of each other, what we're saying mathematically is that the probability of the sequence 'x', one through 'x_n', given a coin bias value, can be written as the product over each observation of the individual likelihoods of each event. So, this is, what it means to be independent that 'x_1' and 'x_2' through 'x_n' are all independent of each other, if the likelihood of all of the data is just equal to the product of the likelihood of each individual observation.

And so for this, problem, the probability of 'x_i', given Pi, is a Bernoulli random variable, which has this mathematical form. If 'x_i' is equal to one, then this term evaluates to be Pi. So, the probability of 'x_i' equaling one is Pi. If 'x_i' is equal to zero, this term evaluates to be one minus Pi, and so the probability that 'x_i' is zero is one minus Pi. And so then, we multiply all of these together. So next, we have to choose a prior for Pi, and so we're going to define that to be a Beta distribution, and this is an arbitrary choice. It's a choice that we make because things are going to work out nicely. I'm not going to go into detail on the background of why we would make this choice. We're just going to work with this choice for this example. And so, if Beta distribution on Pi takes two parameters that are nonnegative, that are strictly positive, 'a' and 'b'. And, has this form.

So, the density of a Beta distribution with parameters 'a' and 'b' can be written like this. And so now, the question is what is the posterior distribution of Pi, given the sequence that we've observed? So, we simply use Bayes rule to do this. Bayes rule says that the posterior probability of the coin bias, given the sequence of events, is equal to the likelihood of a particular sequence, given the coin bias times the prior on the coin bias, divided by the integral of the numerator over all possible values that Pi can take. In this case, Pi can take values between zero and one, because it's a bias.
It's a probability of a binary event. And so now, we simply plug in this likelihood. We plug in the prior and we see if we can solve Bayes rule. So for this example, I want to introduce a trick that is often very useful. Notice that the denominator normalizes the numerator so that it's a function of Pi that integrates to one. But the denominator doesn't actually depend on Pi, because it is completely removed by integrating it out. So in this case, we can write using this notation, which you should read to be proportional to the posterior probability of Pi, given the sequence 'x' is proportional to the likelihood of the sequence 'x', given the bias Pi times the prior probability of the bias Pi. So, the posterior is proportional to the likelihood times the prior. So, what we then do is we multiply the likelihood together with the prior. So in this case, the likelihood term from the previous slide is equal to this. The prior term from the previous slide is equal to this. We then, multiply these two things together, and again because we're working with proportionalities, we notice that this term that we multiplied on does not involve Pi at all. So when we normalize, the product of these two terms, this term is going to appear in both the numerator and the denominator. It's going to cancel out. And so, if we're only interested in representing the posterior as a proportionality, meaning the posterior is equal to this function times some number such that it integrates to one, we don't have to worry about any term like this that is multiplied onto a function of Pi that doesn't involve Pi. So, we have that the posterior is proportional to this term here. So we then, say do we recognize any distributions that are proportional to this function? And we immediately can see that the answer is yes. This function is proportional to a Beta, where the first parameter is equal to 'a' plus the sum of the number of successes and the second parameter is equal to 'b' plus the sum of the number of failures. Or, in the coin flipping case, 'a' plus the total number of heads and 'b' plus the total number of tails. And so here, we have a posterior probability distribution on the bias of the coin that takes into account the prior and the data, and gives us a measure of uncertainty of what Pi is–as represented by this function.

Video 15 (07.49): Maximum A Posteriori

At this point now, we can go back to Ridge Regression, and just like with least squares, we could relate it to maximum likelihood. We can now relate Ridge Regression to a probability distribution, and so this is something called, Maximum a Posteriori. So remember with the likelihood model for linear regression, we modeled data pairs 'x_i' and 'y_i' using this linear model. And we saw that if we put the values 'y_i' into an n-dimensional vector 'y' and we stack the vectors 'x_i' along the rows of a matrix capital 'X' that the least squares solution, which is the value of 'w' that minimizes the sum of squared errors, which is what this is equal to, was equivalent to the maximum likelihood solution for this model, where we assume that 'y' is generated from a Multivariate Gaussian with mean equal to 'X_w' and covariance equal to Sigma squared 'I'. So, these two problems share exactly the same solution. And so we can view the least squares solution, probabilistically as being the maximum likelihood solution for this model on the data vector 'y'. So, the question now is whether we can make a similar probabilistic connection for the Ridge Regression problem. So I'm going to step through that process. First, what I'm going to do is I'm going to make an assumption of a prior model for the vector 'w'. So, we see that the likelihood model is again that 'y' is generated from a Multivariate Gaussian like this. We'll make this hypothesis. And so, what about a prior on 'w'? Let's hypothesize a prior distribution on 'w'. And let's, assume that the prior on the vector 'w' is Gaussian with mean zero and covariance equal to Lambda inverse 'I', where this is now a d-dimensional Multivariate Gaussian, if 'w' is d dimensions. Then we can write, the prior density of 'w' in this way. So, this is simply the density function that corresponds to this probability distribution. And now, let's think about trying to find a value for the vector 'w' that satisfies at the same time–and this is vague but it will become more concrete in a minute, that satisfies both the data likelihood, this term and also our prior condition or our prior belief on the vector 'w'. So, this leads to something called Maximum a Posteriori estimation. So, the MAP-Maximum a Posteriori solution or MAP solution seeks to find the value for the vector 'w' that is the most probableunder the posterior distribution. So, what we say is that the MAP solution for 'w' is equal to the 'arg max' over 'w' of the log of the posterior distribution of 'w', given 'y' and given 'X'. And so, again, because the log function is-doesn't-is monotonic, the value of 'w' that maximizes this posterior is also equal to the value of 'w' that maximizes the log of that posterior. And so, we work with the log out of convenience. And so, the first step in trying to find this thing, this value for the vector 'w' is to use Bayes rule. So, we take the posterior of 'w' and we replace it with Bayes rule, which we know to be this.

So, this distribution is equal to this distribution. And then, using the fact that the log of a product of things is equal to the sum of the logs of those individual things, we can say that the log of this product is equal to the log of the likelihood, plus the log of the prior, minus the log of the normalizing constant, or the evidence term, or the marginal likelihood of 'y', given 'X'. So, this is what we ultimately want to maximize over 'w'. And so, let's contrast this with maximum likelihood, which simply tries to find the value of the vector 'w' that maximizes the log of this likelihood. Now we've added two terms.

We've added the log of the prior and we've added this additional term, which we observe doesn't actually involve 'w'. So, no matter what this value is changing 'w' is not going to change this value. So since we're not interested in the value of the maximum, we're only interested in the location of the

vector 'w' that maximizes it, meaning the 'arg max', we can eliminate this term. We don't even need to look at what this term is, which is useful. Because that means that we don't have to calculate this integral to find this term. And in many models that's going to help solve a problem, where we actually can't calculate this in integral in closed form. Okay. So, let's look at the MAP solution for the linear regression problem that we're discussing.

So, again, the MAP solution is equal to the value of the vector 'w' that maximizes the log of the likelihood plus the log of the prior on 'w', and because we've assumed a Gaussian likelihood, we can replace this term with this term, where we've again, removed the constants that don't impact the solution. And also, the log of the prior, which we've assumed to be Gaussian now is equal to this term. And again, we've removed some constants that don't impact the solution.

So, let's call this objective function 'L'. And then, try to maximize this objective over 'w'. And so again, how do we do this? We take 'L'. We find the gradient of 'L' and then we find the location of 'L'. We find the location 'w' at which this gradient is equal to zero. So we take the gradient of 'L'. It's equal to this. And we find the value of the vector 'w' such that this vector is equal to a vector of zeros. And using simple algebra, we can find that the MAP solution, therefore is equal to this, which we recognize as being the Ridge Regression solution, where we've simply redefined this-the original Lambda term to now be Lambda times Sigma squared. So, notice that just like least squares, the least squares solution maximizes the likelihood. So, it corresponds to the maximum likelihood solution.

Ridge Regression maximizes the posterior under a Gaussian prior assumption on 'w_0' mean Gaussian prior assumption on 'w'. And so, the Ridge Regression solution corresponds to the MAP solution. So, before we had the least squares solution corresponds to the maximum likelihood solution. Now, we have the Ridge Regression solution corresponds to the MAP solution.