

Medical Image Analysis
Professor Ganapathy Krishnamurthi
Department of Engineering Design
Indian Institute of Technology, Madras

Lecture 38
Linear Regression

Hello, and welcome back. So, we start off with linear regression. We will eventually get to artificial neural networks and subsequently convolutional neural networks.

(Refer Slide Time: 00:27)

BREAST CANCER DATA SET

id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean
842002	M	17.99	10.38	122.8	1001	0.1184	0.2776	0.3001	0.1471
842017	M	20.57	17.77	132.9	1326	0.08474	0.07864	0.0989	0.07017
8430003	M	19.69	21.25	132	1203	0.1096	0.1589	0.1574	0.1279
8434001	M	11.42	20.38	77.58	386.1	0.1425	0.2839	0.2414	0.1262
8438042	M	20.29	14.34	135.1	1297	0.1003	0.1328	0.198	0.1543

B

• Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image.
 • Described in: [K. P. Bennett and O. L. Mangasarian: "Robust Linear Programming Discrimination of Two Linearly Inseparable Sets", Optimization Methods and Software 1, 1992, 23-34].

$$y \in \{0, 1\} \rightarrow \begin{bmatrix} 1 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

We start off by looking at this breast cancer data set. It was taken from Kaggle and the appropriate citation is given here, part of a scientific study. The idea here is to classify a certain breast cancer lesion as either it is benign or malignant. So, which is, basically the categories here. If you look, M is malignant, and of course, other category is benign, which is B, not shown here in the snapshot of the data set.

So, each of the columns that you see here would correspond to the so-called feature of the breast mass and each row corresponds to that of a patient. So, how are these so-called features computed? They are computed from a digitized image of a breast mass biopsy. It is called the fine needle aspirate. And what they describe are the characteristics of a cell nuclei present in an image.


So, you have a picture of the cell nuclei and you measure these properties of the cell nuclei from the image, and you list them against every patient. Now, the idea is, using these so-called measurements or features, you classify the breast cancer as either benign or as malignant. What is interesting of course, is that these features are hand calculated by, from the images of the biopsy tissue or biopsy breast cells.

So, this is one, what you call, category of data that you can use in linear regression. Even though this actually, we will see later, it is probably not a good candidate for linear regression. So, here there, are the diagnosis is either M or B. So, your target y , as you call, Y is your target would be either 0 or 1 or you can even say -1 or 1. So, -1 being benign or malignant, 1 being benign.

So, either or, other way. So, if you test positive then, you are malignant, if you test negative, you are benign. So, -1 would benign, 1 would be malignant. Same thing for 0,1. You assign benign and malignant accordingly. So, that is your target, that is what you want to estimate, whether a particular breast leash mass that was found may be based on a test is either benign or malignant.


So, that is, this is one task that you can accomplish with this data. And this data we, we assume that all these features or some of these features have a direct bearing on this particular diagnosis of either benign or malignant.

(Refer Slide Time: 03:17)




Housing Price Data set

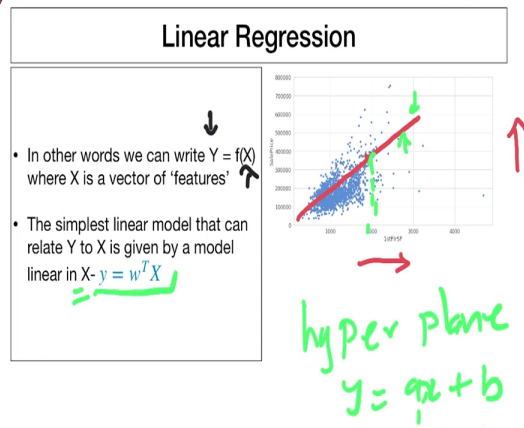
id	latitude	longitude	lotsize	bedrooms	bathrooms	fullbath	halfbath	fireplaces	garage	pool	quality	price
1	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
2	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
3	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
4	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
5	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
6	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
7	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
8	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
9	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
10	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
11	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
12	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
13	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
14	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
15	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
16	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
17	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
18	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
19	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
20	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
21	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
22	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
23	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
24	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
25	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
26	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
27	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
28	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
29	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
30	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
31	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
32	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
33	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
34	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
35	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
36	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
37	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
38	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
39	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
40	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
41	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
42	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
43	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
44	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
45	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
46	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
47	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
48	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
49	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000
50	41.75	-87.55	1600	3	1	1	0	0	0	0	10	160000



Another data set that is often used to illustrate linear regression is the housing price statistics. Again, this is taken from Kaggle challenge. So, you can go online, search for this data set, and there are quite a few versions of this, it seems. So, the idea here is to predict the sale price of a house given some of its features.

So, these features could be continuous values, like for instance lot area, or it could be categorical variables like features like LotConfig. So, it is a mixture of categorical and continuous features and these features, you use to predict the sale price of a house and this is a classical linear regression problem.

(Refer Slide Time: 04:03)



So, let us just come back to get back to what you mean by linear regression. So, the idea is we want, given some feature values, we should be able to predict the price of a, for instance in the case of what we saw earlier, for the housing price challenge, given some features, we should be able to estimate the price of the house. So, the price of the house is what we call the target here, that you want to estimate, Y . So, and X are the features.

So, what is this $f(X)$ that you have written there that gives you this Y . So, this function f is the function of features, X . So, if X is a vector of features like we saw, every house has a row corresponding to the table I showed you. So, the X is a vector of features and we give that as input to this function X , and its output would be 1 scalar variable which is the price of the house. So, that is the kind of problems we will typically look at in linear regression wherein the input is a vector of features and we call it X , and output is a scalar value Y .

So, then again, you might say, what is this $f(X)$ that we are trying to estimate? So, for instance in this case, I have drawn, plotted the first floor square footage on the X axis, first floor square footage on the X axis, and the Y axis is basically the price of the house, the sale price of the house. So, what we have is this very nice scatter plot. And, so it seems that, there seems to be like a linear correlation, almost, between the first floor square footage and the price.

So, as the first floor square footage increases, the sales price also seem to increase. Now, if we draw a line through this like this. Now, this kind of captures the trend in this plot. So, for a particular value of the square footage, you can kind of read off the value of the price of the house for this thing, you can read off right here. So, this red line here, this is what I call $f(X)$. So, in this case, I am considering only one variable, and X in this case is also a scalar, and output Y is also one scalar.

And it passes through the origin, so $Y = mX$, where 'm' is some constant, would be the answer to this, would be the solution for this line. So, so the idea is now, I hope it is clear, is to estimate this line, the equation of this line. This is what we refer to by $f(X)$ here. Now, in 2D, it is like this, on the other hand, if you have multiple dimensions, you are just trying to estimate a hyper plane. In multiple dimensions, you are going to estimate a hyper plane.

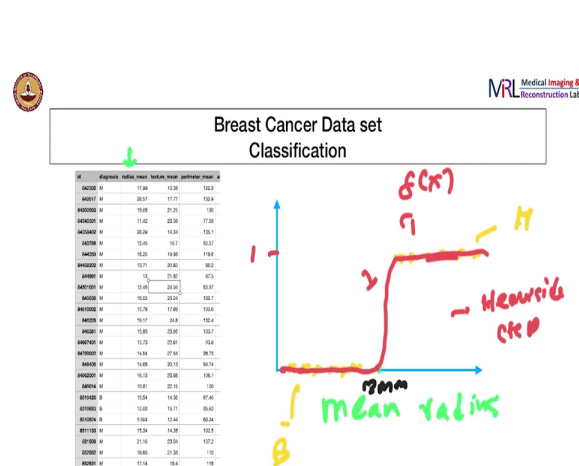
So, if you have, let us say, you are using, thinking of two features and you are trying to estimate the price, then you are just doing a plane. But if you have multiple features, here you have tens of features, we saw for housing price prediction, and, so you are trying to do this linear regression which basically estimates a hyper plane. So, if you just have one feature so you are just trying to estimate a straight line, you have multiple features, then you are trying to estimate a hyper plane.

So, what is the simplest linear model that can relate Y to X , in the sense, how do we estimate Y as a function of X and you want to find that $f(X)$. And we need to model that $f(X)$. So, we need a model for $f(X)$ because we currently make a best case, we do not have access to information that will tell you exactly what $f(X)$ is. Only these correlated features and the target.

So, the simplest model that can relate Y to X is given by a model which is linear in X , it is given by this expression $Y = w^T X$. So, you might be put off by this notation but you have seen some weird version of this in your school when you are trying to fit something to a straight line, so then you might have used some equations like $y = ax + b$. So, this a and b are nothing but components of w . So, this X is just one, this is in one variable, X is one variable, Y is some scalar output.

So, this is the equation of straight line. So, this is more general form wherein you have more than multiple dimensions, and then you just have a weight matrix. So, this w is, you can think of them as coefficients or generally you can call them weights of your model. In this case, it is a linear model, linear index, and you are trying to, and the linear regression problem is basically all about figuring out this w which will best fit this data. So, we will see what that means in the subsequent slides.

(Refer Slide Time: 08:53)



So, before we go any further, so we also want to look at what have, in the case of classification, what is this, what kind of regression can we do? So, let us just plot one. So, one of the features in this breast cancer data set, it would be the mean radius of the cell. It is not visible, might not be visible to you, it does not matter. So, mean radius of this, of the mass or the cells that we have measured under the microscope. So, I am just going to write that down here, going to call this mean radius.

And then the Y axis is basically benign or malignant. So, we are going to say 0 if it is benign and it is 1 if it is malignant. So, once again so 0 if it is benign and 1 if it is malignant. And let us just do this simple rule that may be based on whatever little I see here, 13 millimeters, let us say here, is, these are different colors, maybe black so 13 millimeters is where anything about 13 millimeters is malignant, anything below,

anything, sorry, anything above 13 millimeters is malignant, anything below 13 millimeters is benign.

So, you have to draw a bunch of points here just to illustrate this. Try to use some colors which you have not seen before. So, let us just use some yellow here. So, for instance these are a bunch of points which are all less than 13, which is benign, and then there is a bunch of points here, maybe, again, here, bunch of points here, which are malignant. So, this is M for malignant, this is B for benign because it is below a certain thing. And we are going to use only one variable because we can always type more but easier to illustrate.

So, what is this, how would you do this classification? Ideally, the function we want has to look like this. Anything less than 13, it should always consistently give 0, I will put this as 1, and anything over 13, should immediately jump and give you 1 as this. But this is your Heavyside function, Heavyside step function. So, we are looking for a step function. So, this is your $f(X)$. This is what we want. And this is a classification problem. So, for classification problems we want some functions like this, approximate functions like this.

Here too, you can do similar analysis that I am going to describe for a linear regression, but then there could be some errors because of outliers. And that is where, that is why people generally do not use regression, the regression that I am going to describe for solving these kind of problems. These are called classification problems. And the $f(X)$ that we want is this shape. Whenever your X , which hits 13 millimeter, you want its output to jump to 1, whenever X is less than 13 millimeter, you want its output to be at 0. So, this is the function that we want to have, $f(X)$. So, that gives you the outcome that is the desired outcome.

(Refer Slide Time: 12:16)



Linear Regression

- A system that takes a vector X as input and outputs a scalar y
- The output is a linear function of the input - The model - $\hat{y} = w^T X$ where $w \in \mathbb{R}^n$
- The elements of X are called 'features' or 'predictors'
- The parameters of the model ' w ' are the weights. They determine the relative importance of each of the 'features'

Year	Model	Engine	Power	Price	Color	Options	Notes	Sale Price
1998	Model	Engine	Power	Price	Color	Options	Notes	2000

id	depart	nr_zona	nr_zona_prime	nr_zona_2	nr_zona_3	nr_zona_4	nr_zona_5	nr_zona_6	nr_zona_7
0002	1	1730	1130	1320	1020	1150	1070	1000	0

So, just to reiterate, the idea is that linear regression is a system that takes a vector x . So, I am going to say vector X because we usually have lots of features, not one feature, and it usually outputs a scalar Y . So, most of the problems you will run into, there will be one, one variable you will be predicting, one scalar you will be predicting, that is your Y , that is your target as you call it, and your input can be a vector, it will have multiple features in it.

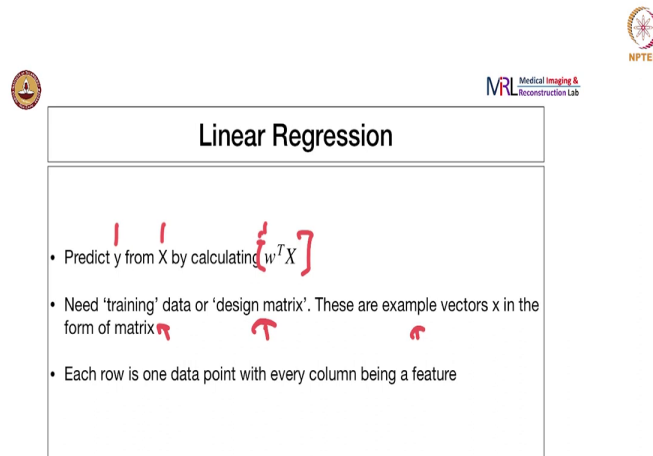
Once again, the way to do that, so let us see, if you have, if you have a bunch of features, the model that you would have is of this form, $w^T X$, where $w \in R^n$, basically it has n components and each for one feature value. So, the elements of X which I have been repeating is called features, and w is always referred to as the weight matrix or the weight vector.

And what does it signify? The pair of the weights are also known as the parameters of the model, and what they do is determine the relative importance of each of the features. So, now, you have an “explainable model”, quote unquote because if that, if a w for a corresponding X is very high, then you know for a fact that that particular feature is contributing a lot to a certain, the output.

So, this is why I have shown here in this example features for both the regression problem, housing price as well as the breast cancer problem, where we have a bunch of,

we have a bunch of numbers, which we call them vectors. And for housing price, actually it is mixed because it has both categorical as well as or continuous values in your X .

(Refer Slide Time: 14:02)



The slide is titled "Linear Regression" and contains the following bullet points:




- Predict y from X by calculating $w^T X$ (with handwritten red annotations above the equation)
- Need 'training' data or 'design matrix'. These are example vectors x in the form of matrix (with handwritten red arrows pointing to the word "matrix")
- Each row is one data point with every column being a feature (with handwritten red arrows pointing to the words "data point" and "feature")

Logos for MRL (Medical Imaging & Reconstruction Lab) and NPTEL are visible in the top right corner.

So, linear regression proceeds as follows. All you have to do is predict y given X as input by calculating this $w^T X$. Now, we can do this provided we know w . But actually, we will not. That is what we want to estimate. We want to estimate the weights. So, we need what is known as the training data or the design matrix. So, these are the examples x vectors x in the form of a matrix. Just like that table, if you look at this in excel table, it look like a matrix.

So, every row of that matrix corresponds to a data point, every column of that matrix would correspond to a particular feature. So, and with this, if you have a bunch of such examples, using that, we can actually estimate this w because once we have w , then a new X comes in, we always calculate $w^T X$ and predict y . So, then how do we estimate w ? So, that is the question, that is the core focus of the linear regression algorithm.

(Refer Slide Time: 15:11)



Linear Regression

- Need to estimate the weights so that the predictions from the model is close to the ground truth *- Training data*
- The 'closeness' is measured using least squares over the entire data set.
- $MSE = \frac{1}{m} ||\hat{y} - y||_2^2$ *(x_i, y_i) - m such pair*
Model output - $w^T x$

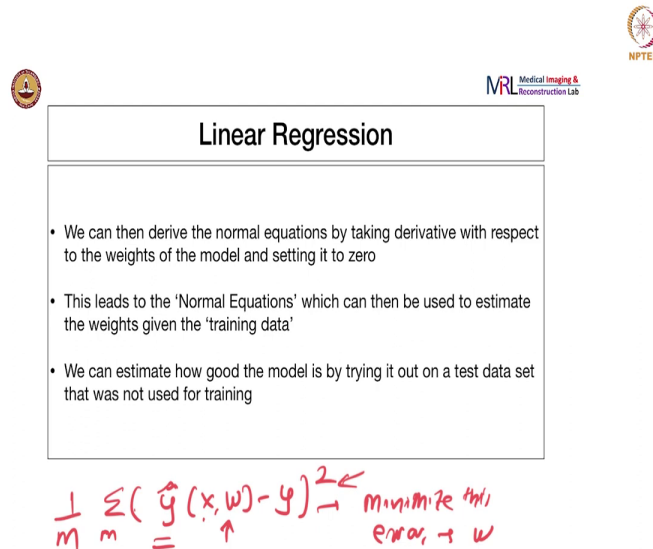
So, the way we go about estimating the weights is that we estimate the weight so that the prediction of the model is close to the ground truth. So, what I mean by the ground truth is the training data. So, training data is usually provided in pairs. So, you have a bunch of, if you have a X , I am going to use subscripts, but this is just to indicate, each subscript indicates a data point, not the component of the vector X . So, (x_i, y_i) you will have this pairs, let us say you have m such pairs.

In all of these formulas, x is implicit and you do not mention it. So, we want to estimate, how do we estimate w , we want to, we take w so that the output of the model so \hat{y} is the model output, which is, you calculate as $w^T X$, is close to your ground truth or what is, what comes with your training data as the target y . So, this particular notation is basically is called the mean square error.

So, you subtract for every case in your training data or examples, you predict your \hat{y} , subtract it from the real, or you subtract the real y from it, square it and you do that for every case in your or every data point in your training data and you add them and then basically you estimate the average because 1 over m will give you the average error, that is a mean square error.

So, once you estimate this error, you can use this somehow to adjust w , that we will see later in the form of gradient design. But the idea is how do we, when we estimate w , what is the criterion that we have to consider. w has to be estimated so that this error is minimized or very low. So, this is just using the training data that is provided to us.

(Refer Slide Time: 17:10)



The slide is titled "Linear Regression" and contains the following bullet points:

- We can then derive the normal equations by taking derivative with respect to the weights of the model and setting it to zero
- This leads to the 'Normal Equations' which can then be used to estimate the weights given the 'training data'
- We can estimate how good the model is by trying it out on a test data set that was not used for training

Below the bullet points, there is a handwritten equation in red ink:

$$\frac{1}{m} \sum_{n=1}^m (\hat{y}(x, w) - y)^2 \leftarrow \text{minimize this error} \rightarrow w$$


So, how do we estimate w . So, we had the expression $(y - \hat{y})^2$, the sum of squared, mean square. So, we want to minimize w . Now, like I said, implicit but I did not really indicate is, if you just look at the expression, it is actually $(\hat{y}(x, w) - y)^2$ is what we are looking at. So, let me write this and use a different so for instance we are looking at $\frac{1}{m} \sum (\hat{y}(x, w) - y)^2$, the summation over all the m examples 1 over m .



So, the \hat{y} is implicit dependence on x and w . Now, we are trying to estimate w , and how do we estimate w ? So, we know that we have to minimize this expression, minimize this error, minimize this error for a choice of w . So, the easiest way to do that is to take the derivative of this loss function with respect to w and set it to 0 to obtain the minimum. When you do that, you get into a closed form expression like your analytical expression for estimate w by estimating w which you can use.

So, this is one way of solving the linear regression problem. We also see in some cases that we cannot, this is probably not the most efficient and then we will then start using the so-called gradient descent algorithm, but this is the general idea. So, we want to minimize the errors between what we predict using the w 's with the, our model $w^T X$ and the correct output, y . So, we do the sum of squares, average sum of squares or mean square error, we want to reduce that.

And we want to reduce that with respect to, by twiddling with the parameters w or the weights w . And the way to do that, of course, in mathematics, is to set the last, this is called the last function or the cost function, and we take the derivative of this with respect to w , set it to 0 or the gradient with respect to w and set it to 0, and then when we do that, we can actually obtain an analytical expression for w .

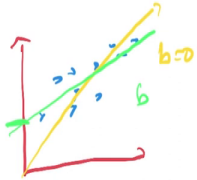
(Refer Slide Time: 19:40)



Linear Regression

- In order to minimize the MSE wrt to the 'weights' w , we take the derivative wrt w i.e. the gradient wrt w and set it to zero for minimum
- $w = (X^T X)^{-1} X^T y$ ←
- These are known as the normal equations if any
- Usually there is a 'bias' term, making the ~~the~~ making an affine function
- $\hat{y} = w^T x + b$ ←




So, if we do the math then we end up with this particular expression for w and these are known as the normal equations. And of course, I remember that (x, w) in all matrices or vectors, but there is also, in some cases, a biased term making it an affine function. So, $y = w^T X + b$.

In case you are wondering where that came from, so for instance, you, if you consider usual regression, you can, for instance think about a bunch of points, you fit a straight

line through it so you can fit a line which goes through the origin. Here, this, for this line, $b=0$, but you can also fit a line for instance, which is a more appropriate, maybe something like this, here b is some value b is number. So, this is the intercept. So, X equal to 0, that is what this is.

So, this is the most general model. In one variable, it is easy to illustrate. So, using this form, you can actually estimate w given your training data set. But this is probably not the most efficient way when you have very large data sets or when you are trying to do real time estimation.

(Refer Slide Time: 21:10)



MRI Medical Imaging & Reconstruction Lab

Linear Regression

- We are trying to minimize $MSE = \frac{1}{m} ||w^T X - y||_2^2$, this is called the training error Training data
- When an unseen set of X , X^{test} is given as input we would still like the error to be low, i.e. we would like
 - Low training error ←
 - Testing error and training error are close ←

Overfitting
Underfitting

So, we are trying to minimize this error and what is this error, this is called the training error because this is done on, this is capital X that are typically using, this is done on the training data, the example data that was given to us, we used to do this. So, this is called the training error. So, when an unseen set of X applies, so X test comes in, where this is data you have not used for this process, it is given as input, you would still like the error to be low.




So, the idea is the following, we need load training error and we also want the unseen data, that is, the testing error should be comparable to the training error. So, this is where two important things arise, one is called overfitting, other one is called underfitting. See

overfitting is when we have extremely low training error, and you will see at some point in next lecture, next two lectures where that comes from.

So, extremely low training error and then, but then when the new data set comes in, your, the testing and training error are not close, they are so far apart, testing error is much, much higher than the training error. The underfitting is when the training error is larger. Anyways, large because your model is not good enough, maybe a linear model is not good enough to fit the data.

So, then your training error would be large and that is called under fitting. So, typically in, if you go to the point where you talk about deeptool networks, overfitting will be more common.

(Refer Slide Time: 22:55)



Linear Regression Regularization
<ul style="list-style-type: none">• Regularization is done to decrease the gap between the training and test error• To obtain better fits typically we allow for more complicated models or functions for eg. polynomial instead of a linear relationship• However, we can also provide criterion that prefers a certain set of solutions over the other, for instance weight decay.• $J(w) = MSE + \lambda w^T w$, forces the weights to the low values when lambda is large <p><i>- $\ w\ _2^2$ Analytical expression</i></p>

Now, so, we were talking about training error and test error and how it would be great if they are very close to each other. So, typically, test error will always be slightly higher than the training error depending on the model because not all models quote unquote “generalized”. Generalization is when you have a new data comes in, your model performs as well as it did on the training data.

Now, one of the steps that you can take to decrease the gap between your training and test error is to inject some prior knowledge about the model. You better take some prior

knowledge into the model. So, one of the, there are many ways of doing that. So, one, one way is the most naive way is you should use more complicated models.

So, for instance, we have we are talking about linear regression where our model is linear, by itself is linear in the variables, in the features but we can have polynomial models, polynomial models, which are more complicated, in that case also, their fitting will be very good. So, your training error will be very small, but then overfitting can happen.

However, we can provide some prior knowledge about what are our expectations. So, one of the things we can do is for instance, add this kind of a term to the loss function. Here $w^T w$ is sometimes that you will see in this form, w^2 . The sum of squared values of your parameters of your model or in this case, weights of your model.

So, what this term does, since we are reducing the mean square error in general, we will try to minimize the sum. And but, but what this particular thing does is since you have w squared, it is, there is incentive for w to become small for a for, a lot of the w 's to become small. So, you want not only, do you want this linear model, but the forces the weights to low values, especially by tweaking the value of λ .

So, for instance, you think that some weights values are too high and so it is not stable, then you just make λ large enough when you try to do the, try to solve for it, and in that way you will force the w 's to be very small. So, basically what you are trying to do is to select a particular type of solution.

By regularization, you are enforcing, you are forcing the algorithm to select a particular type of solution. So, for instance, there are different sets of w 's that might satisfy this equation. Ideally, of course, because, you typically have lots of problems with noise etc.


So, we will not go into that in much detail, but if you if you look at this particular expression, by increasing λ is, what you are trying to do is making sure that this term is weighed very high in the loss function or $J(w)$ is the loss function or the error, or the error term. And when, and by doing that you are forcing w to become very small, in fact, sometimes you can drive it to 0 depending on the form of this regularization.



So, what it is forcing to do is to choose a solution for w wherein all the w s are very low values. So, lot of, for a lot of problems, this is typically used, especially when you actually do not have too much data. The number of weights will be very large compared to the number of data points. And at that time, you have a, the choice is to have this kind of regularizer, think of it as adding more, more equations when there is generally less equations than there are unknowns.

So, this is one way of so-called regularization of the model which helps to decrease the gap between the training and the test error. Now once again, you have the $J(w) = (y - \hat{y})^2 + \lambda w^T w$, we can still take the derivative with, of the left hand of this expression with respect to w , set it to 0 and arrive at a closed form expression for x .

So, analytical expression still possible, is still possible by even with adding this regularization term. There are different regularization terms but each of them enforces some expectation on your solution.

(Refer Slide Time: 27:23)



Hyperparameters

- The hyper parameter in a regularised linear regression model is the lambda used in weight decay
- Machine learning algorithms typically have hyper parameters which have to be tuned separately from the algorithm
- A separate partition of the data 'validation' data is used to train the hyper parameters.

= $\lambda w^T w$

So, now we come to this so-called hyper parameters, which are also an essential part of all these models. So, we saw in the previous slides this parameter λ that we used by, to weight the $w^T w$ term. By tweaking λ , you can enforce either small, small values of w . So,

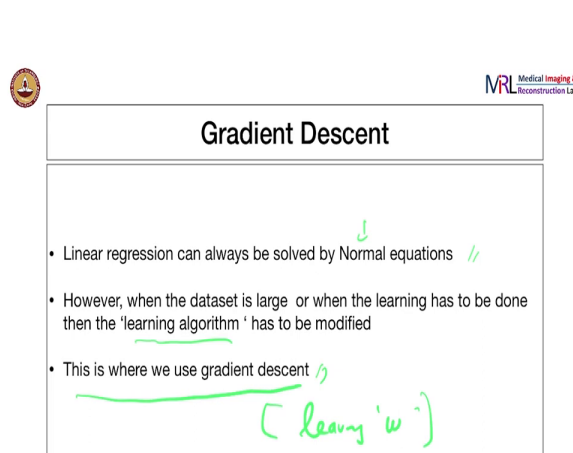
this λ is referred to as a hyper parameter. Almost all machine learning models have some hyper parameter that you have to tune.

Now how do you tune this hyper parameter, how will you make it work? Now you will be tempted to say I will put different values of λ , train my network and then of course, test it on test data. So, that is typically not how it works. So, what you will do is take a separate partition of the data called validation data, and you would tune your hyper parameter using that validation data.

So, after training with different λ , you will test it on some validation data, which is different from the test data in order to tune your hyper parameters λ . Once again, hyper parameters are there for pretty much a lot of the machine learning algorithms, even neural networks have this. And this is just an indication of how many of the concepts are common across these models.

So, one, this hyper parameter in this case is only one parameter, λ , but in some, many networks, for instance, many machine learning methods, there could be more than one parameter. So, you need to have a systematic way of evaluating a correct hyper parameter and for that, you would need the so-called validation data set. So, you would not tune your hyper parameters on the test set, rather, you turn it on the validation data set.

(Refer Slide Time: 29:06)



The slide is titled "Gradient Descent" and contains three bullet points. The first bullet point is "Linear regression can always be solved by Normal equations //", with a green arrow pointing down to the word "Normal". The second bullet point is "However, when the dataset is large or when the learning has to be done then the 'learning algorithm' has to be modified", with "learning algorithm" underlined in green. The third bullet point is "This is where we use gradient descent //", with "gradient descent" underlined in green. Below the third bullet point, there is a handwritten green note "(Leaving 'w')".

- Linear regression can always be solved by Normal equations //
- However, when the dataset is large or when the learning has to be done then the 'learning algorithm' has to be modified
- This is where we use gradient descent //

(Leaving 'w')

Now linear regression can also be solved with the normal equations. This can be done. However, when the data set is very large, millions of data points, or when the learning has to be done in real time, like you are getting one point at a time and you want to estimate, keep estimating your weight matrix or weight vector, then this, our learning algorithm, so I would say that this, solving this normal equations amounts to some learning algorithm.

But, so, then we have to change a different, to a different algorithm when we are trying to do only a few bunch of points at the time or even one point at a time. This is where we use gradient descent. So, we are going to take a brief look at the gradient descent algorithm and how it can be used for learning w , the weights of your model, in this case, especially for your linear model. We will see that in the next video. Thank you.