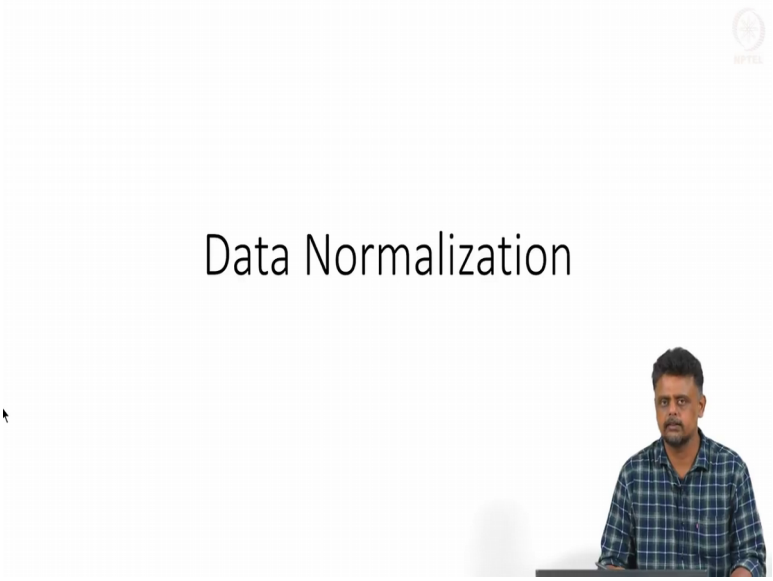
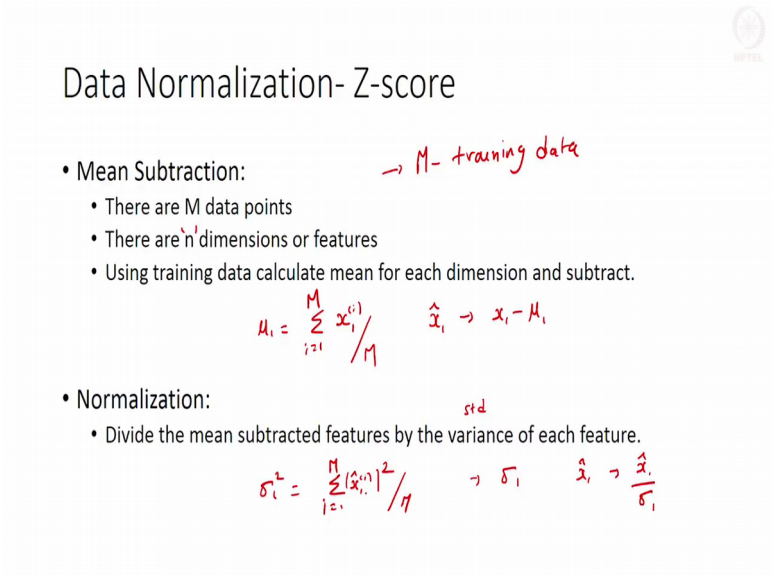
Machine Learning for Engineering and Science Applications
Professor Ganapathy Krishnamurthi
Department of Engineering Design
Indian Institute of Technology Madras
Data Normalization

(Refer Slide Time: 0.13)



Hello and welcome back, so in this video we will look at data pre-processing or data normalization and techniques, before you start using than as a input to some of the common machine learning algorithms okay.

(Refer Slide Time: 0:29)



So what do I will describe now is most often use the commonly used technique and is called the Z score normalization, I just go through what is actually done and then we came see why

we actually have to do it. Okay, so let say we have M data points, which is for a training data points, so this is, so it has to be M training data and there are N dimensions, so which is basically N dimensions or features, so incidentally these needs also known as features scaling techniques right.

So what we typically do is that we make them all 0 mean, so which means that if you have, let say we denote our data points by X right and look at, let say we are considering a features X1 right, so then we calculate the mean for that feature mew 1 and we have this I which is the data point index, so why equal to 1 to M, so we have mew 1 and we just set X1 to X1 that is mew 1 okay.

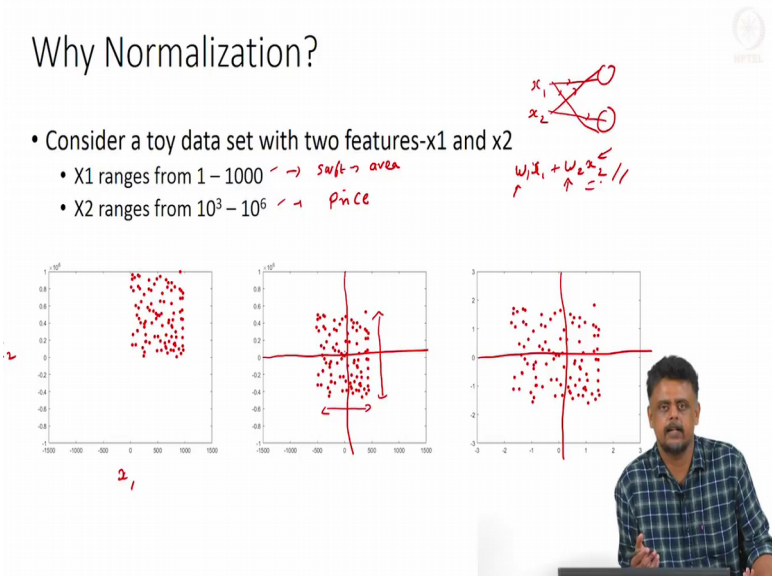
So we do that for every feature that is available in the dataset, so now the all our data points are 0 scented or 0 mean, the next step would you typically do is to divide this, so I denote this by X 1 cap, so that just to be different, divide the mean subtracted features by the variance, this actually standard deviation, square root of the variance is what you want to of each features, so if we can calculate, so once we have, then the 0 mean than the standard deviation we call them sigma 1 square nothing but the mean of the, so if we have, I equal to 1 to M then each of these I and then square.

So of course in the previous summation I left out the normalization by the total number of data points, of course, so that is am at the mean are, so I have to divide by M, divide by M, so this is the, since we have subtracted the mean out this will work, so we take the square root of that, so it will be just sigma 1 right, so we take sigma 1 and so we get X1 cap, cap goes to X1 cap divided by sigma 1 okay.

So that is the process, so you calculate the mean, so I left out that M in the first part, calculate the mean for each feature independently over the training data, not over the entire data set, now we have splitted data into training validation in this thing, so you calculate the mean over the entire training data, subtract the mean for each features from the individual data points and divide by these and deviation of each of the features okay.

So this is the general, the most often use the normalization technique, this is of course assuming that your data is goes in this distributed which might not be true most of the time, so what does this accomplish it, it brings your, make sure your data points 0 mean and units is standard deviation or unit variance right.

(Refer Slide Time: 4:00)



So why is it essential? Why did you want to do that? So, before we go that we just look at some what it looks like after the normalization, so I have just considering two features X1 and X2. Okay and X1 range from 1 to 1000 and X2 ranges from 10 raise to 3 to 10 raise to 6, so if you want solid example you can think of this as the square feet or the area of some living space, apartment or plot of land and this as price of that living space. Okay, very cheap by today's standard, price in rupees or whatever denomination want to, currency want to choose.

So if you plot this twos, so again X1 here and X2 on that axis for all 3 plots. That is what it looks like, it is a board drawn from random normal distribution, does not look anything spectacular but so I just want to show you what happens after the normalization, so if I subtract the mean out from X1, X2, you can see that they are 0 centred okay, so and so the zeros here kind of an, 0 centred but you can see that the X1is in this case, this is the span of X1 if you can think of it as the you know the scale of X1 is in this direction, the scale of X2 here.

So if you look at it, if you actually have look at the numbers, this is in the 1000 and this is of the order of 10 raise to 5 or 10 raise to 4, so this height the range over which the span is much higher than this range. Okay, so if we actually divide by the standard deviation for each one of them, you can see that ranges are also similar right, I think -2 to 2 or both the X1 and X2 parameters okay, this is after we divide by standard deviation.

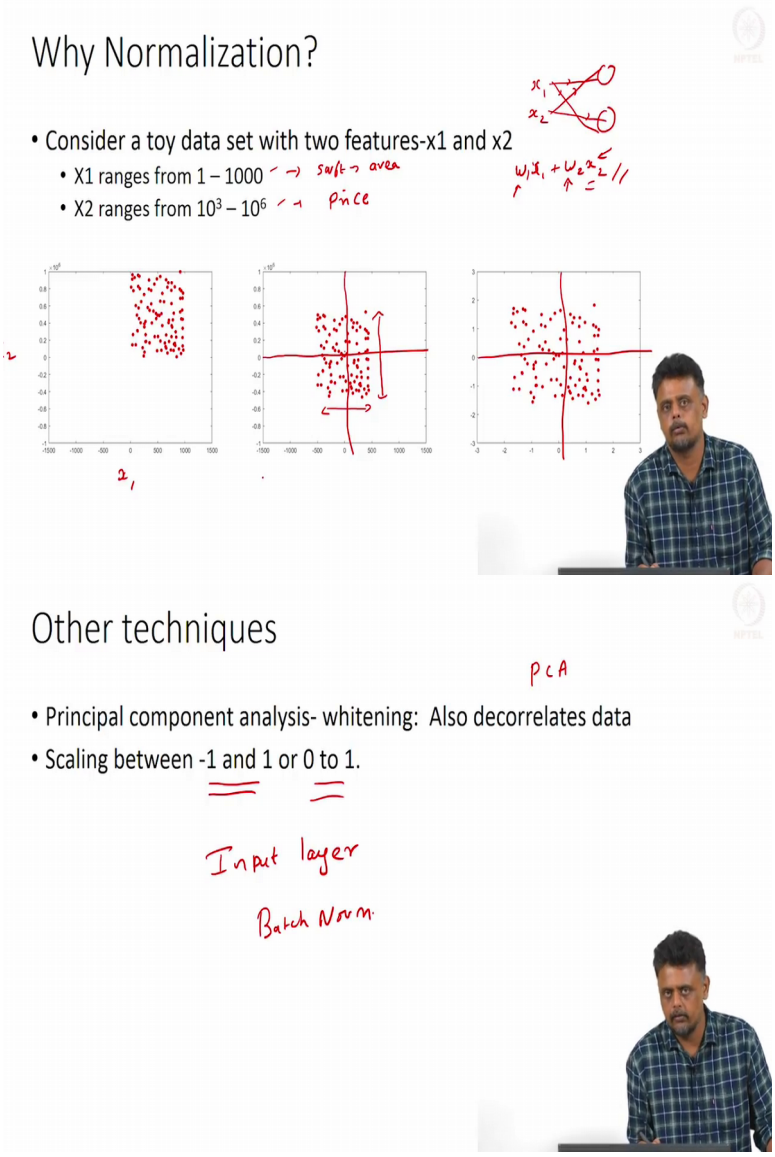
So why do we need to do this, so if you recall, let say if we are considering a neural network, let say I am just going to look at two neurons just to show you what will happen, let say the input is X1 and X2 right, so you have say like that right, so problem here is that initial we have terms like W1, X1 plus W2, X2 right this term will appear often right, so when you look at this kind of terms, if you are not normalizing, you will see that you know X1 has, X1 is much, much smaller than X2 of the factor of 10 raise to 3.

So, but during optimization than in order, from order to from network to converge then the weights have to be adjusted accordingly, so you would expect that the weight multiplying the larger value would be much smaller, let say by same order of magnitude, so it turns out that this kind of optimization is much harder to do or typically will not converge or would be very slow to converge, so because when you are inputting saying that X2 because it is working with numbers you are saying that X2 is more important than X1 because it takes larger values.

Now if you think that all your features are equally important than its important to do this normalization because it brings them in the same range of numerical values and it helps converge and faster okay, optimization is easier this way using gradient descent right, so because we have sums of this form where you know if of this will be the input to your their sigmoid function or the nonlinearity, so then the larger term will tend to dominate all the times, so to prevent that from happening, of course during optimization W2 would be adjusted so that X2 does not dominate, but of course that is a difficult problem for optimization.

If X1, X2 are the same range than the problem is easily solved. Okay, so that is the reason why the to this normalization, if all red line points are in the same range, you need not sometimes do it, again, but then also depends on what your output is like and all that, so, for instance, if you are doing a conditional output, sorry probabilistic output, let say it is a classification task, it makes sense to squash all your inputs in the range 0 to 1 or within this or do this kind of normalization, same thing with regression okay, so if you are inputs are bounded then you can have a bound on your output also, so it makes sense to do this normalization for many classification regression tasks.

(Refer Slide Time: 8:34)



So the other techniques which will look at, the something called the principal component analysis where you can reduce the dimensionality of data and also decorrelates the data, so, for instance, in the previous example I said okay, you can think of the smaller feature as the square footage and the larger variable as the rise, obviously they have not here correlated right, so there be a lot of such features in your data, which might be correlated and the principal component analysis techniques helps decorrelates also.

Now we will look at this at a later lecture, in the coming weeks where will talk about what PCA is and then I will show you the how we can use for pre-processing data for machine learning algorithms and of course there is other scaling we can do here, we did the score wherein we subtracted the main divided by standard deviation, you can scale your data to lie

between -1 and 1 or between 0 to 1. Okay, it is depending on your application, this can be done, these are also valid pre-processing techniques, I would not call them normalization but pre-processing techniques that will help you. Okay.

So these are the, so what we have look at is the most fundamental or the often use pre-processing technique and it is typically done for many any kinds of input you can do that for imaging inputs or any kind of real value inputs okay, so from here on we will look at, so this is again for the input layer, so this is for the input layer okay, so this is your data, which is your, this is goes to as input to the first layer of your algorithm in this case, the planning algorithm, so what happens in the intermediate layers right?

So this kind of techniques applicable thereto and how do you go what implementing them? So this will be the topic of our next lecture, where will look at a very, at this point in time and often used technique called batch normalization okay and that seems to help quite a bit for faster training and convergence and gets of many of the problems okay, so one of the things before you go that, one other thing that I pointed out but this created is that if we have very large features value, it is quite possible you use it, the saturation of you are nonlinearity okay, especially if you are using sigmoid that can easily happen, if the weights also blow up. Okay, that is a possibility.

So let say X2 it is a very high weight then you will end up in the saturation well the G mean there be no learning, so that is typically why want to have all your inputs to be scale with the certain ranges okay, so how do we address this when it happens in a hidden layer okay, that can think of it, there were is a problem to, so we will look at this batch normalization or batch norm is called which axis a layer, one of the layer in a deep network and how it helps in faster convergence, that will be the topic of our next video. Thank you.