

GPU Architectures and Programming
Prof. Soumyajit Dey
Department of Computer Science and Engineering
Indian Institute of Science Education and Research-Kharagpur

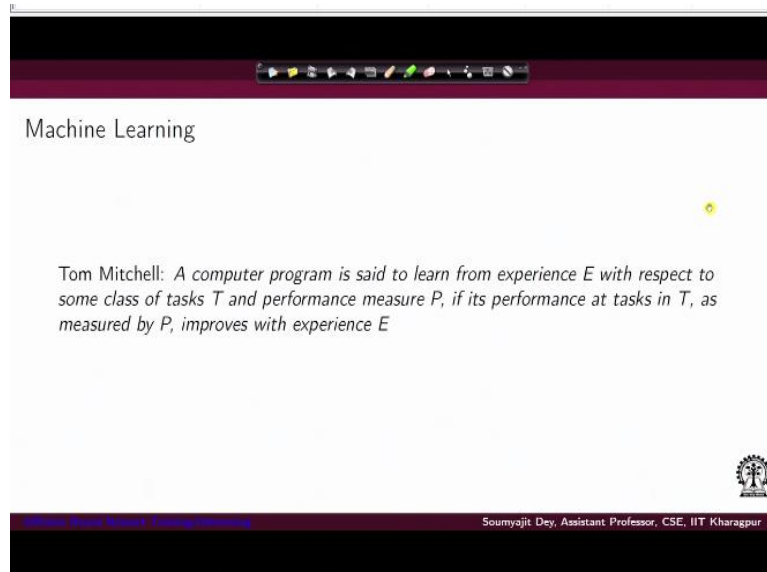
Lecture-54
Efficient Neural Network Training/Inferencing

Hi, welcome back to the lecture series on these GPU architectures and programming. So we will start with one very relevant topic right here, which is like how this paradigm of GPU programming or better to say in general assembly style programming, how does that really help in accelerating inhale workloads and to be more specific, we will focus on how this kind of architectures help to accelerate the execution of several I mean, both neural networks training as well as inferencing passes.

Because as we all know these are the most relevant workloads, at least in the MLCS community and also CS community in general. And also in I mean, it is finding a lot of applications in several cross disciplinary areas. And in terms of system execution, we can always say that this is the basic reason why these kind of neural network based systems are finding popularity is that there is a lot of data available nowadays point 1.

And point 2 is of course, we have got this kind of large scale systems where I have the facility of GPUs to accelerate parallel programs run times by and several orders of magnitude itself and that really helps in realizing this kind of large scale deep neural networks and they provide us significant and more and better to say usable accuracy in many application domains.

(Refer Slide Time: 02:06)



So, with that background, I mean, we like to quote the famous researcher Tom Mitchell here where he says that a computer program is said to learn from experience E with respect to some class of tasks T and performance measure P. If the performance it provides that tasks in T as it is and the performance measure of course is P, it improves with the experience. So the more experience you provide be it is learning better how to execute this class of tasks with respect to a performance measured that is defined by P right.

So we can say that it is learning from experience, provided it excels in future executions of such stacking task instances or maybe newer instances of data. I will just like to notify one very important thing that preparation of this set of slides have been significantly helped by my wonderful teaching assistant and even course who really has done a great job in helping me prepare this slide sets. I mean, of course, the others were also there. But in this slides, he has got a huge contribution, fine.

(Refer Slide Time: 03:14)

Learning Paradigms

Figure: Types of Learning

Souryajit Dey, Assistant Professor, CSE, IIT Kharagpur

So just the way we will approach it is that we will just give a brief overview of the basic computation that is performed while training a newer network for some tasks. So I will just put in a note here, that this is not going to be a ML kind of lecture, where we go into the nitty gritty of what is learning how is he done and all that, but rather, we will just take a specific workload, which is neural network training and inferencing process.

We will just identify what are the computational parts of that. And then we will map those computational parts to GPU architectures and see how they can be accelerated. That would be our approach here to be very clear. So doing a quick brush up of the learning techniques as we all know that so, these learning paradigms can be divided roughly into 3 aspects like you have supervised learning, unsupervised learning.

And also there is this other paradigm of reinforcement based learning. In strictly speaking if we just consider supervised and unsupervised learning, in the first case, we consider a set of label data that is already provided we there is a label like you have a data set and you have tags which actually said whether which are the positive as well as the negative or some other kind of classification tags are also there, which give you some information about the dataset.

In the other case, which is unsupervised, there is no such better supervision available that is that for the data, you do not have any such tags that are available. So, all ML techniques which work on the first kind of data are known or kind of classified under supervised learning paradigm, whereas the other case we call it as unsupervised learning.

(Refer Slide Time: 05:00)

What is Performance P ?

- ▶ Performance P refers to some metric for assessing the quality of the rule/function learned.
- ▶ We restrict our discussion to Supervised Learning Problems.
- ▶ Supervised learning problems can be Classification (The target labels are discrete or categorical) or Regression (The target labels are continuous values).

Sourmyajit Dey, Assistant Professor, CSE, IIT Kharagpur

So, in general, when we are talking about a machine learning system, so, it has a got a set of tasks, where you are trying to learn rules or functions based on such examples, right. So the examples can be characterized by a vector of real numbers or Boolean variables, depending on what kind of tasks we are talking about. Each example may have a target level like we discussed supervised learning in that case, there is a target level.

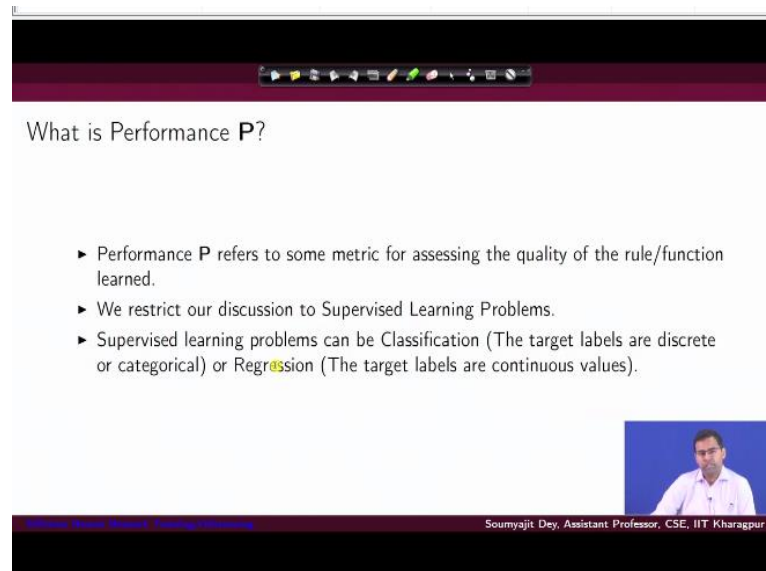
In case it does not have any kind of such levelling, we call it an unsupervised learning system. Examples are not available explicitly and some agent environment interaction must be envisaged to extract meaningful examples, which is the case for reinforcement learning, which is gaining a lot of traction nowadays. So like we say here that the examples are not readily available.

But you set up an agent environment interactions, so you kind of model the environment or you kind of put in a learning agent which is trying different possible moves in the environment and finding out what I mean how good is that move based on some runtime evaluation. And that data is kind of used to figure out well, what is a meaningful example and then make good use of that example to learn a classifier or a similar kind of system.

So and then we have this issue of experience. So of course, more experience means you have a bigger data set with reach examples. Now, of course, this is something which will be well known in the email communicating that of course, you can have a big data set, but that does not mean that it is reached there has to be enough information about the state space that you want to learn.

So the amount of data in that data set may not be that important, as like the richness of the data set in terms of information that can be mined from that data set about the characteristics of the status.

(Refer Slide Time: 06:59)



What is Performance P?

- ▶ Performance P refers to some metric for assessing the quality of the rule/function learned.
- ▶ We restrict our discussion to Supervised Learning Problems.
- ▶ Supervised learning problems can be Classification (The target labels are discrete or categorical) or Regression (The target labels are continuous values).

Soumyajit Dey, Assistant Professor, CSE, IIT Kharagpur

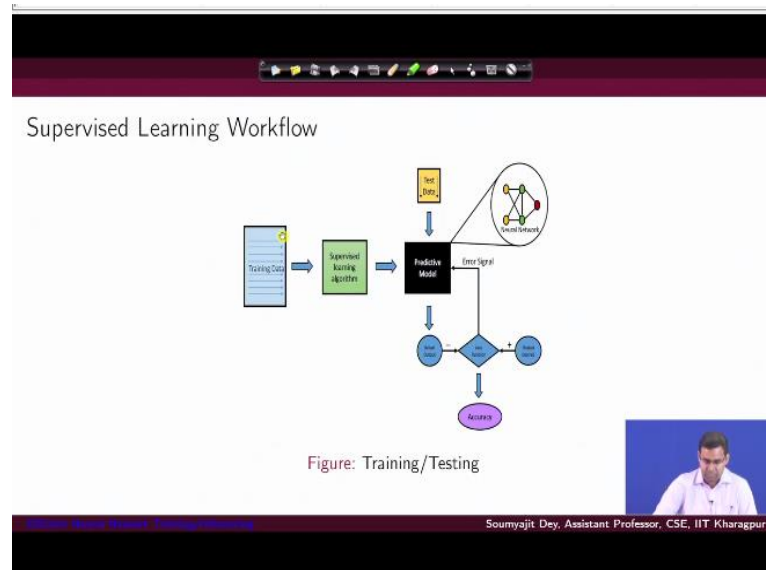
Then well, what is really performance, performance refers to some metric for assessing the quality of the function that has been learned. So well, at the end of doing all that training and everything based on the data that is provided you learn a function, but you have to calculate whether that function is good enough as for some deployment scenario or not. So do you need a measure, usually, we call it a cost function, which evaluates the performance of whatever function or rule you have learned right.

So in our lecture, we will discuss only supervised learning kind of problems and this kind of problem. So that means we are assuming there is a training data set with proper levelling that is available and that will be used to learn proper rules or functions. And for this kind of problems, the primary classification that is done is like, well, you can have a classification task or you can have a regression task.

These are the 2 kinds of systems that you can really learn. So, when you learn a classifier that means, given a new data input, you can infer the class of the data. So, basically you can kind of classify all the inputs to certain categories and you are trying to for a new input you are trying to find well which category it best fits in. So, that is called typically a classification task.

On the other case, where we have a regression class, so, regression would mean that you have this input data and you are trying to find out a fitness function like what function would fit that data pattern nicely. So that then again, I would say you are learning a continuous function, so that given any new data point, this continuous function will give you a possible value for as an output, taking the new data point as an input.

(Refer Slide Time: 08:54)



So in general, when we are speaking of supervised learning tasks, we already have a generic workflow like this, that means you have a training data set with levelled data, you have this supervised learning algorithm, which will work on this data. And at the end of the day what it will come up with is a predictive model. That model can either be a classifier or it can be a regression model.

Now, well how do I test the goodness of this model. I have trained it using the training data, I have figured out using some algorithm which ran on the training data that this is the model I am getting. Now to test the fitness of this model, I give you some other data as input called the test data set. And using this test data, I check with what is the inference that is done by the model, what is the classification of the new test data or what is the regression value that this predictive model computes.

Well, but this is test data that means for the input values, I also have the suitable output candidates available to me right. So, based on whatever the model is predicting, and whatever is the real output value that was supposed to be in the test data, I calculate some error and that

is measured by the loss function. So, I have this concept of loss function, I mean why do I have this function well.

There is an error right I have calculated something and using my predict predictor and there is something that test data it is that this should have been the real output. So, there is a difference, right. How do I really characterize this difference. Because we are speaking of data in a high dimensional space, it is not like a single tuple the value is 1 which is predicted, whereas the actual value must be 1.1.

It is not really like that, in general, we are talking about high dimensional data borrowing from linear algebra, we are supposed to bring in a possible fitness function, which characterizes this distance metric of in the high dimensional space that will, I am predicting this point in an N dimensional space, whereas the actual output is this point in that same N dimensional space.

So I need a suitable distance metric, which I characterized by choosing a loss function. And then, using this loss function, I generate something called an error signal, right, which I can typically use to refine this predictive model right. And that is how the loop will go on and finally, if I see that, whether I am happy with the accuracy that I am getting. So, if I see that well, the accuracy is not good enough.

Then I said that, well, this system is not learning enough from the training data, I may read some more data, or I may need to change the supervised learning algorithm itself or the other alternative can be that I need to change the parameters of the algorithm right. So this is the generic flow that I have for any supervised learning system. So you build a model, you tune you evaluate the quality of the model.

You accordingly tune parameters of the model to figure out whether you are getting a better model or not. And once this step is complex as you can see that this is an iterative step you are building the model, you are checking the error that is generated based on that you may be tuning parameters of the model here. But finally, by after exploring all the options in the parameter space, you can report a maximum accuracy.

If that is good enough you are done. If that is not good enough, you change things here or you may say that well even changing here is not helping me, you increase the input data set by some way. So, even if by changing algorithms is not really helping that means, this training data set is not really rich, you are not covering enough aspects information theoretic aspects of the input state space of the model fine.

(Refer Slide Time: 12:58)

Supervised Learning Algorithm

- ▶ Characterize the model to be learned by some parameter θ
- ▶ Define a loss function between predicted output and actual output. (function of θ and inputs)
- ▶ Update θ so that the loss function is minimized.
- ▶ The more the loss function is closer to the minima, the more closer is the predicted output to the actual output
- ▶ This is also known as parameter estimation.

Handwritten notes on the slide include a graph showing a line with a negative slope and a vertical line, and the equation $y = f(x, \theta) = \theta x + \frac{\theta}{2}$. A range $\{0, 0.1, 1000\}$ is also written in pink.

So, now if we just enlist things step by step, so, any standard supervised learning algorithm would have the following characteristics, it will characterize the model to be learned by some parameter theta, it will define the loss function between the predicted output and the actual output, it will update this parameter theta that is, this kind of parameter tuning, as long as the loss function the value is not minimized.

So, of course, as you can see that the loss functions value depends on the model, right that I have trained. And now inside the model I have put in this unknown variable which I am calling a parameter of the model. So, typically what is a parameter. So, let us say I am defining a function $f(x)$ for y , instead of that I write well let the function be $f(x, \theta)$ of y . So this is a normal mathematical function.

We are used to x is the input generate the output instead of that all that we are saying is the output is y , this function is not only characterized by the input, there is a theta right. Well, what can that mean. Let me just take an example here. So let us say I am just trying to write a closed form expression to motivate the problem, do not think this will be the actual thing in general. So let us say the function is $5x$ plus theta by 2.

I am just writing one function here. All I am trying to say here is so you can see that the value of the function for certain input x , not only depends on the x itself, but also on what value of θ I am choosing it can be 0 to say that it is absent. It is not, it is not really influencing the function, $5x$, it can have a value 0.1 to say that well. I am not biasing the output of $5x$ by much it can have a value 1000, which can say that well I am playing, I am giving a big shift, I am giving a big positive shift.

In those cases, when the value of x is let us say, I mean less than some value like 200. I mean, I am just trying to motivate again, I have said there is nothing there is not a strict mathematical characterization, I am just trying to say that if you draw this our favourite coordinate system, and then if θ is really not there, you have a line like this slope right. If you put a small θ , you are just giving a small constant value to that slope.

And then again, if you have a big value already here, then initially will you are rising from here, but this there is already a significant value. But again, this is a very nice day example I would say, in general, θ having a very complex relationship, because it is maybe a high dimension in the high dimensional space and all that, right. So, where all that we are doing is we are saying that width in that model, there is a lot of information, I do not know. And that is what I am characterizing by the θ .

And I am asking the learning algorithm to figure out what would be a good value of θ , so that the loss function that is the error in my estimate is getting minimized, right. So, the more the loss function is closer to the minima, the more closer is the predicted output to the actual output. So in that case, I would say that well, in that case, I have chosen the θ in such a way that it approximates the actual functional relationship very nicely, right. So essentially, this is nothing but a kind of estimation of the parameter θ .

(Refer Slide Time: 16:53)

Linear Regression

- ▶ Given a dataset $\mathcal{D} = \{(x_i, y_i), x_i, y_i \in \mathbb{R}, i \in [1, n]\}$, learn a function $f_{\theta}(x) = y$ that can predict any unknown x_j not in the dataset, with a label y_j with reasonable accuracy.
- ▶ The function f_{θ} is of the form $f(x) = wx + b$.
- ▶ The parameters to be tuned are the slope w and the intercept b .

The slide is part of a video recording, as evidenced by the Windows taskbar at the bottom and a small video inset of a man in a white shirt in the bottom right corner.

So we will start with regression here, like as we discussed that regression is nothing but learning a continuous function. So you have a data set a label data set, given inputs x , you have outputs y , right. So you have this kind of values like that, and you are trying to figure out what kind of curve would approximate it nicely. As you can see, there is not going to be a good approximation because there are several values, which are lying at quite a distance, right.

So in general, that is what I want to learn, I want to learn a function f , which is parameterized by x as well as θ . I mean, it takes us input x , it is parameterized by θ and is going to output y . And I want this function to fit the test data as nicely as possible right. So I want to learn this function, which can predict any new x , what is the output level y . And I want that to happen with reasonable accuracy, right.

So what we can do is let us say we make a linear approximation here. So let $f(x)$ be some $wx + b$, and I want to parameterize. So essentially, this w and b represents it is like a form by which I am saying that the parameter θ influences x , right, so in this case, I am saying that let θ be such that it is essentially appear w and b . The first item w is just killing x and the b is giving a constant dc shift, right. So I have a function of this form.

And I want to tune the slope and the intercept. So that I get a functional representation, a linear representation, which fits this input values as nicely as possible.

(Refer Slide Time: 19:02)

Linear Regression

- ▶ Linear Regression can also handle multidimensional linear models of the form $y = w_1x_1 + w_2x_2 + \dots + w_nx_n + b$ where there are multiple x values.
- ▶ Geometrically, this is equivalent to fitting a plane to points in three dimensions, or fitting a hyper-plane to points in higher dimensions.

$$[w_1, w_2, \dots, w_n] \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} + b$$

Now, of course, in general these input data can be high dimensional like we have been saying, right. So, x can be a vector comprising x_1, x_2, x_3 up to x_n like that. So, then I am not even I am also not saying that w is a scalar, but rather w is also a row vector. So, that essentially we are speaking like this that this is so, θ contains values w_1, w_2 like that up to w_n , which gets multiplied with x_1 and up to x_n .

And then I have the shift b , which is the value here, right. So, essentially in this case, I am saying that we are trying to figure out a function which fits in a multidimensional x .

(Refer Slide Time: 20:00)

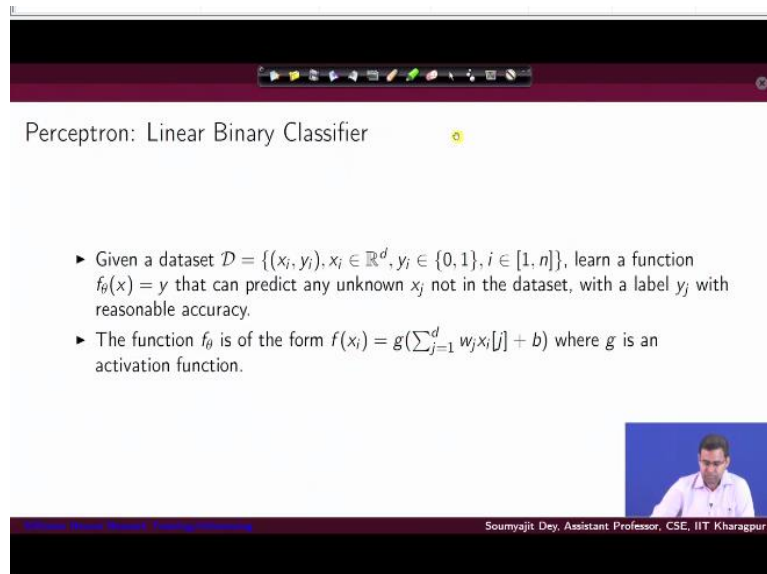
Perceptron: Linear Binary Classifier

- ▶ Given a dataset $D = \{(x_i, y_i), x_i \in \mathbb{R}^d, y_i \in \{0, 1\}, i \in [1, n]\}$, learn a function $f_{\theta}(x) = y$ that can predict any unknown x_j not in the dataset, with a label y_j with reasonable accuracy.
- ▶ The function f_{θ} is of the form $f(x_i) = g(\sum_{j=1}^d w_j x_i[j] + b)$ where g is an activation function.

$$\theta = (w_1, \dots, w_d, b)$$

So, this kind of linear binary classifiers or what we have been popularly knowing them in the literature perception right. So, consider a data set like this x_i, y_i these are the training data that are there for given x_i input the output level should be y_i . Now, we are saying that well,

they are all values where the dimension D right. So, there is y_i belongs to \mathbb{R} to the power d . And in this case, we are considering a binary classification that means, the outputs are either 0 or 1.



The screenshot shows a presentation slide with a dark blue header and footer. The title is "Perceptron: Linear Binary Classifier". The main content consists of two bullet points:

- ▶ Given a dataset $D = \{(x_i, y_i), x_i \in \mathbb{R}^d, y_i \in \{0, 1\}, i \in [1, n]\}$, learn a function $f_{\theta}(x) = y$ that can predict any unknown x_j not in the dataset, with a label y_j with reasonable accuracy.
- ▶ The function f_{θ} is of the form $f(x_i) = g(\sum_{j=1}^d w_j x_i[j] + b)$ where g is an activation function.

In the bottom right corner, there is a small video inset showing a man speaking. The footer contains the text "Soumyajit Dey, Assistant Professor, CSE, IIT Kharagpur".

So, as you can see we are making a jump here. This is our model for regression which we have spoken about. So, in that case we come trying to approximate the data by a continuous function, the other problem as we discussed earlier, this classification, classification means given some input, I want to see the membership of the resulting output to a class right. Now, in a simplest possible case, the class can be binary.

So in that case, I will say that we will have class 0 and class 1. So for a given input, just see whether it goes maps to a 0 or a 1. So that is why this is a simple linear binary classifier or in other words, we call it as perception. So simple perception, so we will learn a function for this problem, where we are trying to figure out what is the suitable settings of theta such that I have, I mean the most accurate possible binary classification.

So we are trying to figure out $f_{\theta}(x) = y$, that can predict any unknown x_j , which is not in the data set with a label y_j , which with reasonable accuracy, of course, in this case also we consider this f_{θ} to be of this form. Given an x_i I am trying to figure out first, what is this in I mean, what is going to be the linear scaling here. That means, in this case, we are saying that well you have this set of widths, w_1 to w_d .

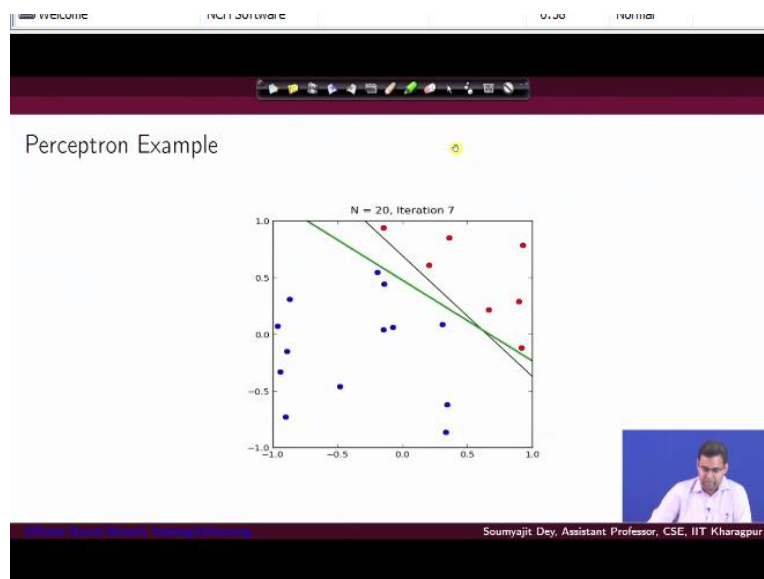
I am considering that x is a vector of size D , and they are all getting multiplied by the x_1 to x_d . And so, kind of I am trying to see what is theta here right. And then also we have another

parameter that is b . But wait there is something more here right g . Now, what is that. So, in this case, what we are doing is, we are saying well, let us first apply our linear scaling idea which we discussed for the regression problem, right.

So essentially, we have a linear function map which is taking the input vector x generating a single value by multiplying it with a row vector w_1 to w_d , and then adding a constant should be, right. But does that solve our requirement. No, what is the requirement, that requirement is not to figure out a real value y , but rather to figure out a Boolean value y which is taking either a classification 0 or 1.

So, what we do here is, whatever we get out of this calculation, we map it to something called an activation function, which will perform the final classification. So, it will have a thresholding nature, so that for up to some values, it may map it to 0 up to some values beyond that to 1, it will have some kind of thresholding feature, which will be useful to get the initial computation of the linear function map to a corresponding 0 or 1 partition class.

(Refer Slide Time: 24:04)



(Refer Slide Time: 24:09)

Perceptron Algorithm

Let $x = [x_1, x_2, \dots, x_n, 1]$ and $w = [w_1, w_2, \dots, w_n, b]$
The value of the dot product $w \cdot x$ is the same as $\sum_{j=1}^n w_j * x_j$.

```
while convergence is not reached
  for i in range(len(y)):
    y_pred = heaviside(w.x)
    dw = (y[i] - y_pred) * eta * x[i]
    w = w + dw
```

Soumyajit Dey, Assistant Professor, CSE, IIT Kharagpur

So, if we try to use this idea and design an algorithm for performing such classification task one can be as follows. So, you have this kind of set of inputs x , and you have for them, the actual outputs known for them, right the y i's and you are trying to estimate what should be these parameters w 's and b 's. So, what you do is, you run an algorithm through which what you are trying to see you are trying to have some initial values for these w 's.

You use them to predict the y 's. And then you figure out what is the difference between the actual y and the predicted y . So, that is your error. You multiply that with a constant η called the learning rate. You multiply that with that positions x value. So, I mean, we will come and see why we are designing this like this that will come later on. Let us just try to assume that here you are scaling this error by these values.

And trying to figure out what can be an incrementally better estimate of w . So, this calculation gives you a modification in w , you add that modification here. So, this can be a positive or a negative quantity, you get a refined parameter w , which will again push into that in this loop again, let us understand all that we are doing, we are using a set of x , we are assuming a set of initial w 's.

We are using them to estimate a predicted value of y . Then we are calculating what is the difference between the predicted value and the original value right. Again, here we are making use of heaviside as a threshold in function like g and then we are getting an error, that error we are multiplying by a constant as well as the current x value for that component right. And then what we are doing is we are modifying the w .

Let us try and understand again. So this is a simplistic case, right, all we had done is we have removed the b here as you need to be 0, we are just playing with w simplistically, we are using one special function called heaviside. Check what it is, it is a simple thresholding function. And using heaviside we are thresholding it to get a predicted value of y . We calculate the error multiply by learning rate the value of the chosen x to get a possible shift in w and accordingly tuned w .

We keep on executing this loop as long as we are not really satisfied with the values that we are coming. That is what we will define as convergence later on. Now, we are just let us now try to give an example here that how really that that looks like consider a 2 dimensional space, you have x 's and y 's and we are trying to learn a classifier or linear binary classifier or a perceptron here.

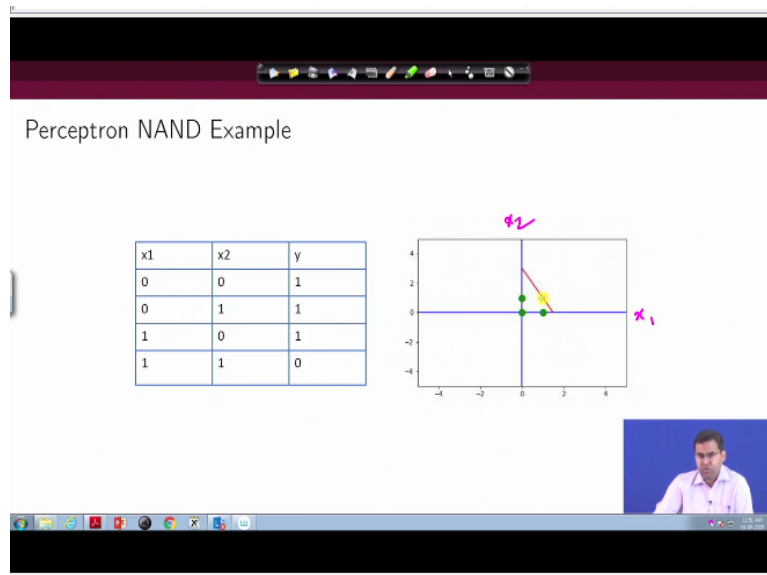
So, let us say these are your data points and in an ideal system, this is your classification line. So, these are the data points that should be in one class, these are the data points in blue that should be in the other class, right. If you start executing our algorithm that we just discussed, what may happen is you start with a test classifier like this. So this is the line that you are getting by assuming some initial values of w 's right.

Because what does really the initial values of w give you. It gives you a line right. Of course, here you also have a constant value b which is not 0, right. Now, as you keep on executing the algorithm, those values of w will keep on getting tune it will keep on changing and what you expect is that with certain time you are going to get possibly a good approximation. Now, why is this a good approximation for this classifier that will be hypothesized.

Because as you can see that this rightly classifies the data points of one class, which is marked in red from the data points in other class, which are marked in blue right. So that is a good classifier, then, as you can see, they are not identical, but this serves the same purpose at least in this case. So, with this, I will have great convergence, because since this is also correctly classifying all the data points just like the original hypothesis classifier function.

So, I do not have anything new to learn from this data set, right. So that I will have received convergence and I must have stopped here.

(Refer Slide Time: 28:48)



Consider certain examples here. For example, suppose I am trying to learn a NAND relationship. So what is NAND. So, this is the truth table of NAND here right. Consider that you have inputs x_1 and x_2 , right. So you have an input x , a 2 dimensional vector a 2D represent here mean you have input x , which is a row matrix, a column matrix of size 1 cross 2, and you are trying to use classified to binary classified to a 0 or 1.

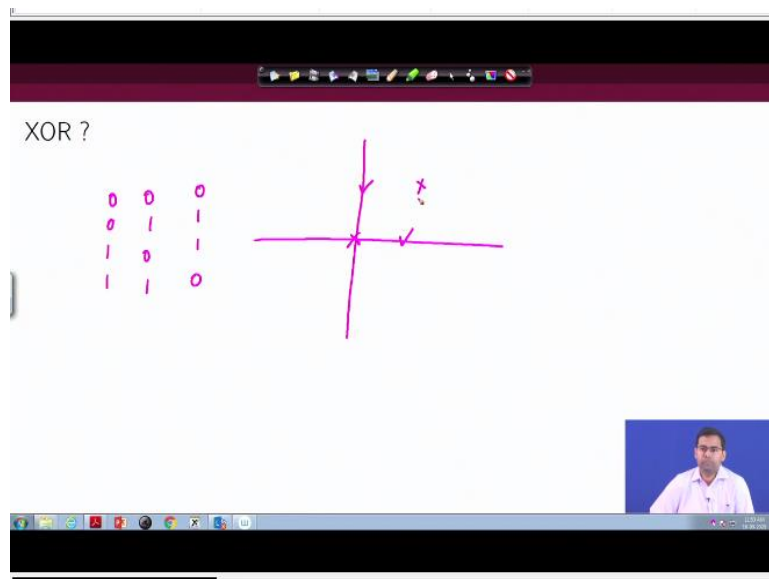
So, that is like learning any 2 input Boolean function, in this case, the choice is NAND, we have the truth table of NAND here, both 0s, you get a 1 0 1 you get a 1 1 0 you get a 1 only when you have both 1s. So, that will give you an end which is 1 and now you negate the end to get 0 which is an end value, right. If you plot these here in your coordinate system, so the positive cases are the green points right.

So and for all those cases, you have, so essentially we are just plotting here, the x 's and y 's x_1 s and x_2 s, right. So in this direction, you have x_1 , in this direction x_2 . And here, we are just trying to see what are my positive cases. So 0 0 is a positive case, because output classification is 1, I am just saying positive negative instance of point 0 and all that. And you have 1 0, and 0 1 1 0 and 0 1, both also as positive cases right.

Now for 1 1, that is a negative or 1 class classification, which is marked in red. If you apply the previous algorithm here, you will learn to slide right, so as you can see, the classifier does its job because it is classifying everything that is positive on one side, keeping the 1 value

here on it with suitable tuning or redefinition, I can put it I mean like this, so the classifier is doing his job.

(Refer Slide Time: 30:56)



And interesting question in this case can be what happens with XOR. First of all, let us understand that we are learning a linear classifier here. That means we are considering that these points are linearly separable, right. But in general, it may not be so. So let us see why. I mean, this method may not be working. So well we saw, what is the truth table absorbed by the way right.

So the next task is to figure out whether this is linearly separable or not. So if you use the previous method, then you have 0 0. So that is a 0 class classification, 1 0, and 0 1 as 1 class or positive classification, and this is again, 0 class or negative classification. As you can see, you cannot draw a line, which will put the 0's on 1's side and 1's on the other side, you cannot have such a relationship here.

That is why the XOR relationship will not be linearly separable. And this algorithm where I am assuming that the function approximation is going to be a linear function is not going to work, right. So we have to see how things can be managed at this point. With this, let us end this lecture. And we will take it further from here in the next lecture. Thank you.