Course Name:Business Intelligence and Analytics
Professor Name:Prof. Saji.K.Mathew
Department Name:Department of Management Studies
Institute Name:Indian Institute of Technology Madras
Week:11
Lecture:40

ARTIFICIAL NEURAL NETWORK | BI&A | Prof. Saji K Mathew

Hello and welcome back to this session on artificial neural networks for financial time series modelling. So in this course, so far we have looked at methods or analytical techniques that is for cross sectional data including the session on cluster analysis. We try to make the data cross sectional when it was not. But in this session we are going to look at a different type of data, which is time series data, which is time series data meaning data collected at fixed time intervals, like your heart beats, or like the daily temperature, or the weather data or stock market data, which we are going to see, right. There are fixed time intervals which characterize this data.

And we are also going to study a different technique, which is unlike the type of techniques that we learnt. What is the unlikeliness of this method? Of course, decision tree was, of course biological like it was a tree but actually it is not a tree it is a metaphor. But here it is not just a metaphor but it is also some, the basis for the artificial neural networks is the biological neural networks. So artificial neural networks is an example of a machine learning technique which you can call as a artificial intelligent technique in itself.

It is an AI technique because human intelligence is imitated. So human intelligence is emulated or it is the basis for the design of artificial neural network. So the origins of artificial neural network was 1940s, okay, 1940s when two scholars, McCulloch and Pitts. Pitts was a mathematician, McCulloch was a expert in biology. So you can see what has a biology expert to do with a mathematician .

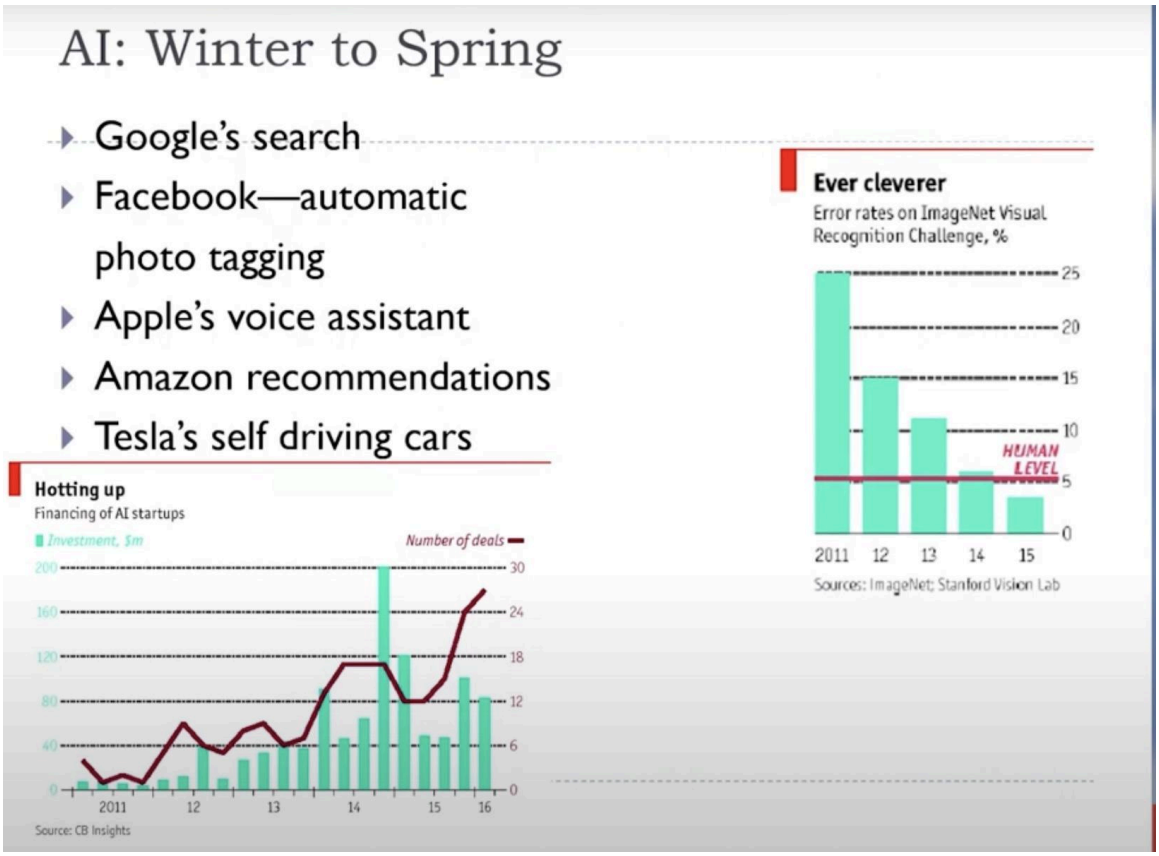
You do not go together for coffee or food with someone who is totally different from you, a zoologist or someone who is studying archaeology and then management, you know, you do not like them, right. That is a different world. And we, our research is different, my focus is different. You see the strength of unlike minds coming together or what we call as interdisciplinary research. So you see more insights or more valuable insights when people from different areas come together and start talking about and start getting to understand the others domain and that is when you know the, what you call

the, cross pollination is another example, in biology it happens and artificial neural network is the result of two such scholars working together and in 40s itself they actually showed a rudimentary model of a ANN.

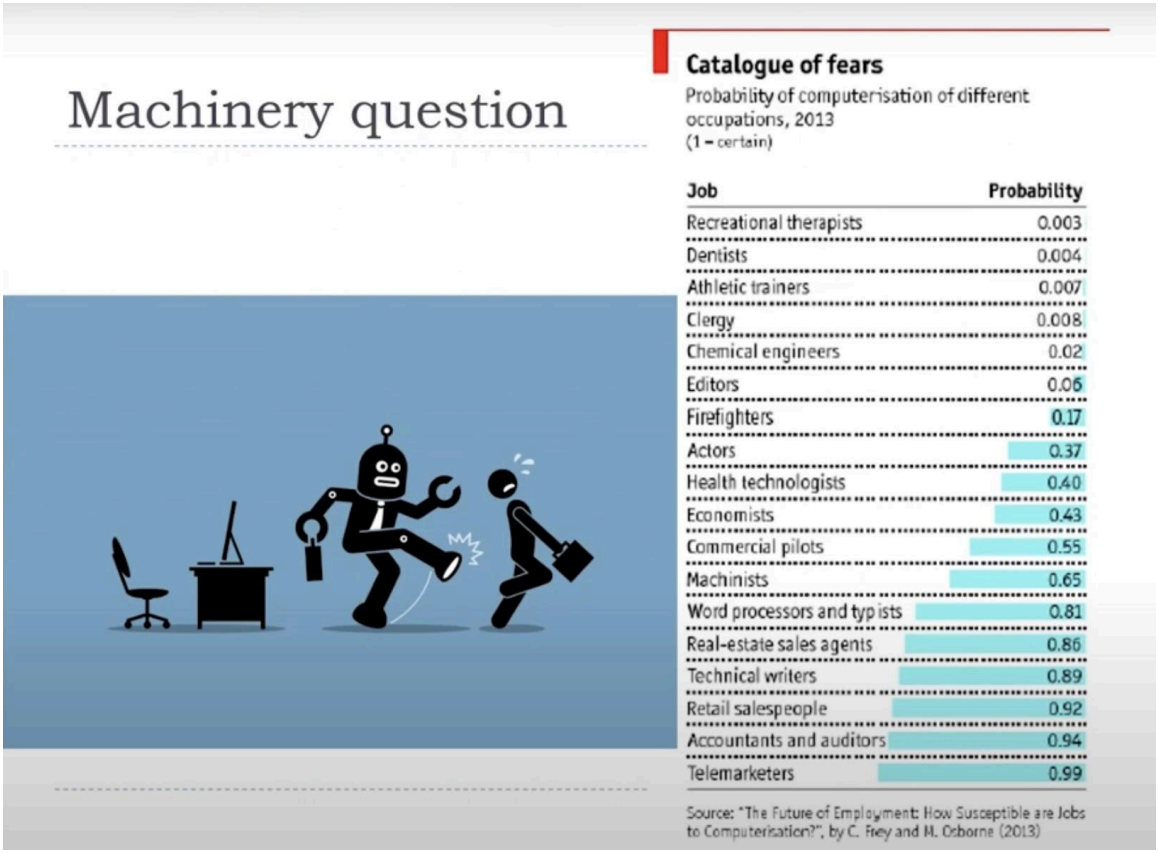
So I will show you early models as we go of neural networks and how it evolved and how it is useful for data analysis. Our aim is to explore the use of neural networks in data analysis, particularly modelling data and using it for prediction and our problem is related to capital markets or, you know stock market prediction. Okay today is the world of AI, so you know this is a more updated cartoon just for entertainment . Today there is discussion on future of work, future of work, what machines can today not only support decisions, but also take decisions. They are substituting humans and that has become a challenge and therefore there is discussion on future of job.

Who should be worried about their jobs? Whose jobs are very rule based that can be learned and not all jobs fall into that category probably today. So in order to motivate you for ANN and AI based techniques, you can see that AI is having great traction today, especially in analytic circles because they have been found useful, found to perform in certain applications very well, particularly with the emergence of social media. So you know, Facebook can identify you from your pictures, right that whether you like it or not

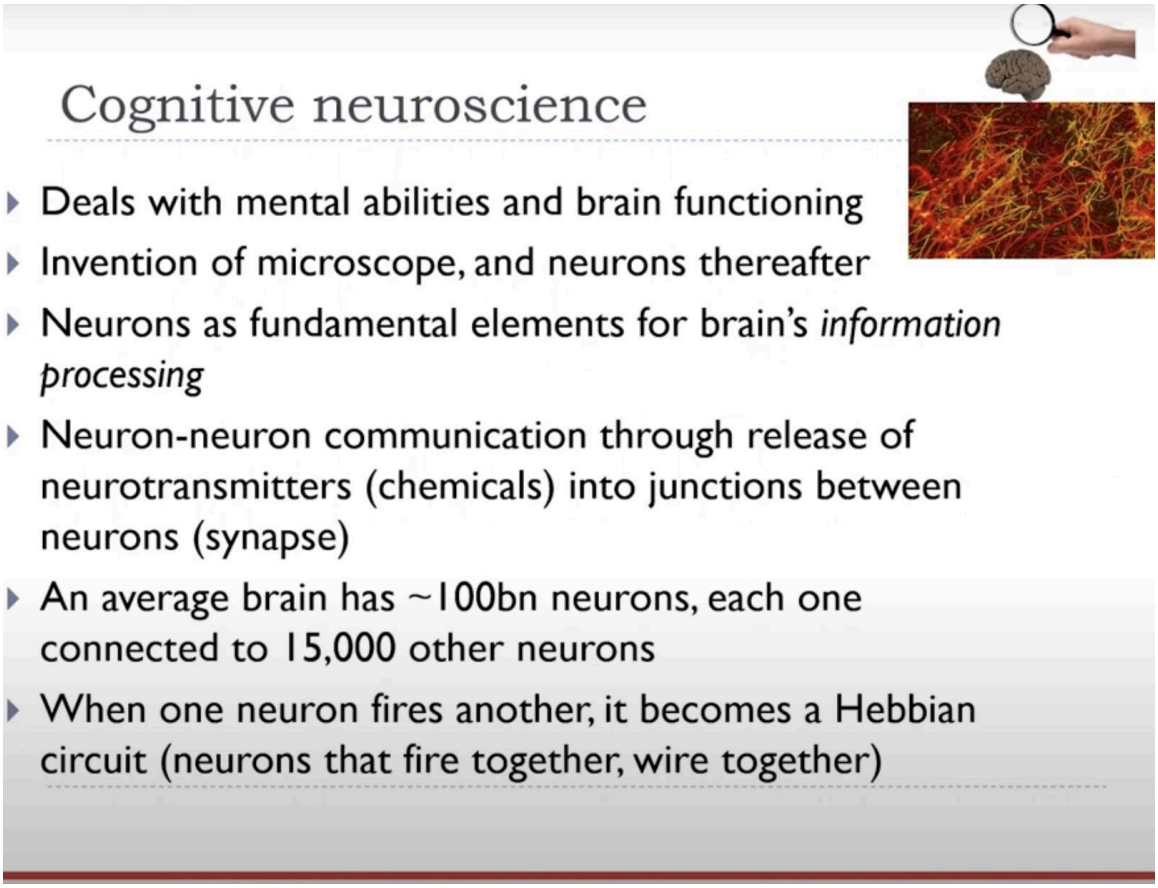
they do, okay if somebody else uploads your picture, you are identified with that person or group, because of your picture or the pattern recognition. Okay, these capabilities of AI is something that these large organizations acquired, acquired. So AI startups were acquired at high prices and they have become users of AI for business strategy.



You can also see that how performance of AI based solutions have been growing. Look at some of the reports, how accuracy of translation has in several cases is better than that of human translation. Those are the trends that you see in this. Just some data from reports that is available in public domain, and there is today the machinery question, that is this question did exist whenever automation actually happened in the history of modern world, when industrial revolution happened, you know the lot of activities performed by humans, by carrying a person from one place to the other. But when cars came, it was actually substituting or replacing them.



So that is a, it is a problem that did exist along with us but this seems to be having future impact. And you can look at this report, I think by researchers Frey and O'Shawn who concludes that certain jobs are less likely to be taken away and for example, a dentist job or a priest job etc. So do not switch your jobs so fast, these are very interesting examples. Alright and you also know that the recent importance of a discipline called cognitive neuroscience, cognitive neuroscience. Lot of people specialize in this area because cognitive neuroscience is a science that informs AI, because it is patterns from the neurology of human beings that is used for designing useful artificial neural networks.



And therefore neural networks features in artificial neural networks. And in the human brain, we all have our brains, and we evolved because our brains evolved. How many neurons you had when you were born or when you were conceived? From one cell, we actually grew, our number of neurons multiplied and how did we learn as children? How did we learn? We did lot of experiments to learn, right. Do you know that? We touched a hot surface and learned that we should not touch a hot surface.

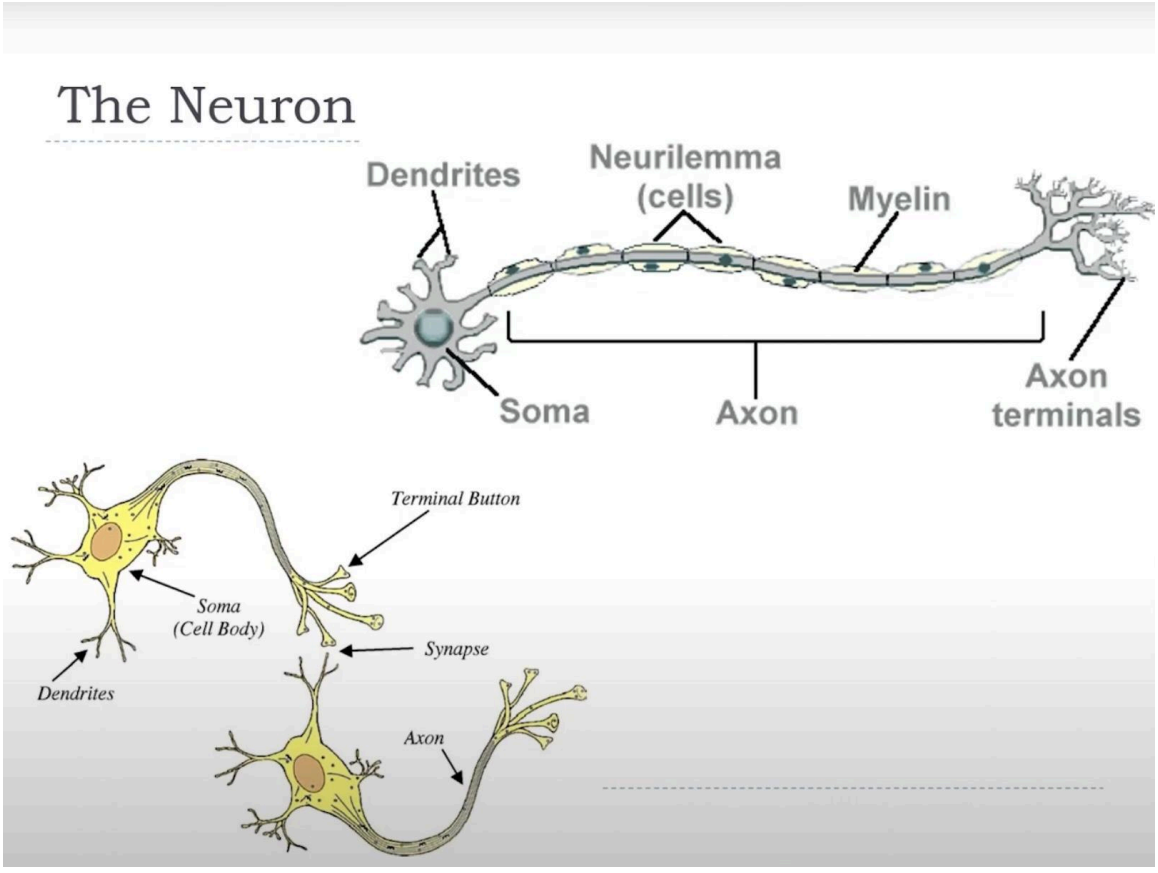
We disobey parents because we want to experiment, create that data for learning. It is through experimental data that actually our neurons got the connections that they have today. The learning is basically embedded in the connections of the neural networks. Those connections were established through a large sums or amounts of data constantly taught or trained our neural networks. It is that that is mimicked in artificial neural networks.

Here is some data about biological neurons. An average brain, I do not know whose brain is that, but an average brain has 100 billion neurons and each one is connected to about 10^4 other neurons, or 15,000 other neurons. So you see one point here, every neuron is connected to many other neurons. There is a connection or so called, it is

called synapse, you know, you must have heard about this term. So there is a synapse of one neuron with the other neuron.

That connection is what defines our memory. Our recall actually happens from connections. It is not like, you know, electronics where, you know, there is a cell which store a 0 or 1. There is a particular pattern in which the 0 or 1 is stored. But in neural networks it is not such state of a transistor or a device that stores a memory a bit.

It is the connections that actually establishes patterns of memory in the human brain. So that is better learned in cognitive neuroscience or cognitive psychology than in AI. This, how the brain functions, how, what is memory, what is intelligence is better explained by cognitive neuroscience. And artificial neural networks actually function based on those principles. Hebbian circuit which says neurons that fire together, wire together or the connections are established through synapse and then those connections defines your memory.

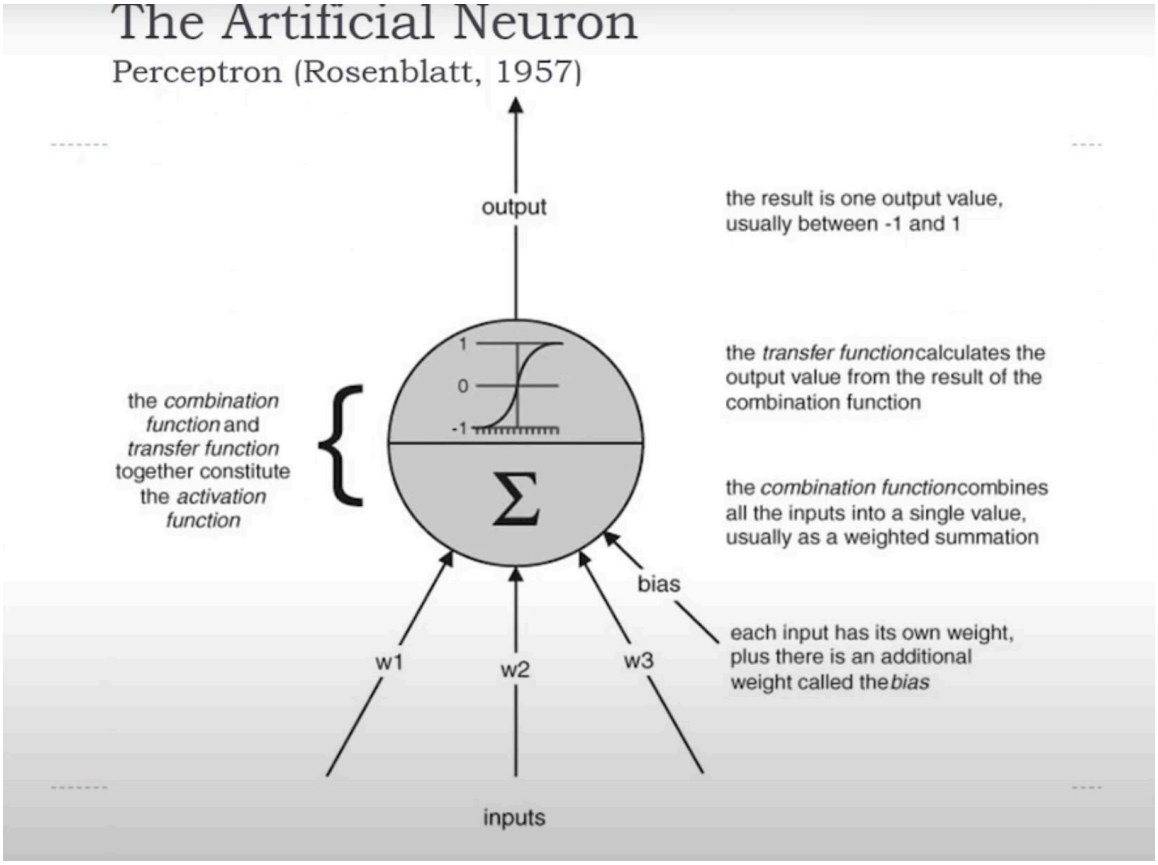


Now here is the depiction of a biological neuron. This is not neural network that you are learning but this is how biological neuron, one neuron looks like. And you can see that a neuron has a origin on dendrites, origin on dendrites. This these are, these are sort

of wires through which a neuron get connected to another neuron. So this is the origin, this is the terminal.

The terminal is called the axon, the end. So the end of one neuron connects with the dendrites of another neuron and that is how connection gets established. So, and you can also see that the biological neuron has different parts and its protective shield and so on. The long stem of the neuron is called the axon and axon terminal at the end. And the other diagram shows how neurons, biological neurons connect together.

You can see the terminal nodes here connects with the dendrites here and here is the point of connection, called synapse. Synapse is when a connection is made, when a connection is established and this is through chemical medium. It is called chemical transmitter. It is a chemical medium through which synapse happens. And signal is transmitted after synapse from the dendrites to the axon through electrical signal, that is electrical medium whereas connection between neurons is through chemical medium. And this is important in neuro treatments.



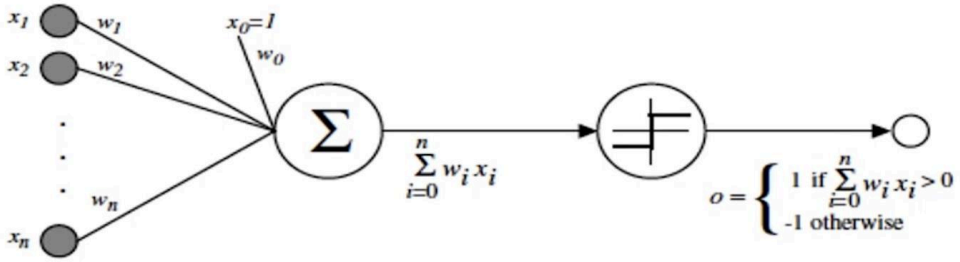
Now from biological neuron to artificial neuron. As I said, it is McCulloch and Pitts in the 1940s first attempted to represent an artificial neuron in a more mathematical way, or in a way that describes its function. A functional representation including mathematical symbols were attempted by McCulloch and Pitts and in McCulloch and Pitts representation there were no weights. But this representation is attributed to Rosenblatt in 1957, almost 10 years plus, after the first neuron, artificial neuron was represented and this particular representation is known as a perceptron.

A perceptron has weights, perceptron has weight, whereas the original artificial neuron did not have weights. So it is a more advanced representation wherein you can see that a neuron has inputs, you can call them as x1, x2, x3, etc, okay. It has a constant called bias. And then this symbol represents a sum, summing. And since weights are there, it obviously shows at one point of the neuron a weighted sum of the inputs is calculated, weighted sum of the input that is w1x1+ w2x2+ w3x3+ wpxp+ a constant, okay, that is what is computed here in a neuron.

And it is, the output of a neuron is not the weighted sum of the inputs, but it has one more stage, it has one more stage, that is called a transfer function. There is a transfer function associated with a neuron. The output from the first stage goes to the second stage of a transfer function wherein the intermediary output is transferred to the final output using a function. It is not the exact output but it is a transfer function that will determine what would be the exact output. A transfer function could be a linear function, a transfer function could be a step function, a transfer function could be a log function. A transfer function could be a sinusoid function. It could be multiple functions one could choose, in designing neurons and neural networks. So there is a stage of transfer function or what kind of a function is this, that is depicted here? It looks like an S, right. What do you call it? It is a sigmoid function or a logit. You are familiar with logistic regression I believe.

It is a logit. And divided by $(1+ e)^{-x}$. So this is a sigmoid, a sigmoid function. And sigmoid function is widely used, especially when you use data which has outliers. Can you look at the shape of the curve and interpret why it is so? It is very linear within a range of x but when the value of x goes to the other bounds, it becomes less responsive. So it is very less responsive to outliers.

Perceptron



$$o(x_1, \ldots, x_n) = \begin{cases} 1 & \text{if } w_0 + w_1 x_1 + \cdots + w_n x_n > 0 \\ -1 & \text{otherwise.} \end{cases}$$

Sometimes we'll use simpler vector notation:

$$o(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \cdot \vec{x} > 0 \\ -1 & \text{otherwise.} \end{cases}$$

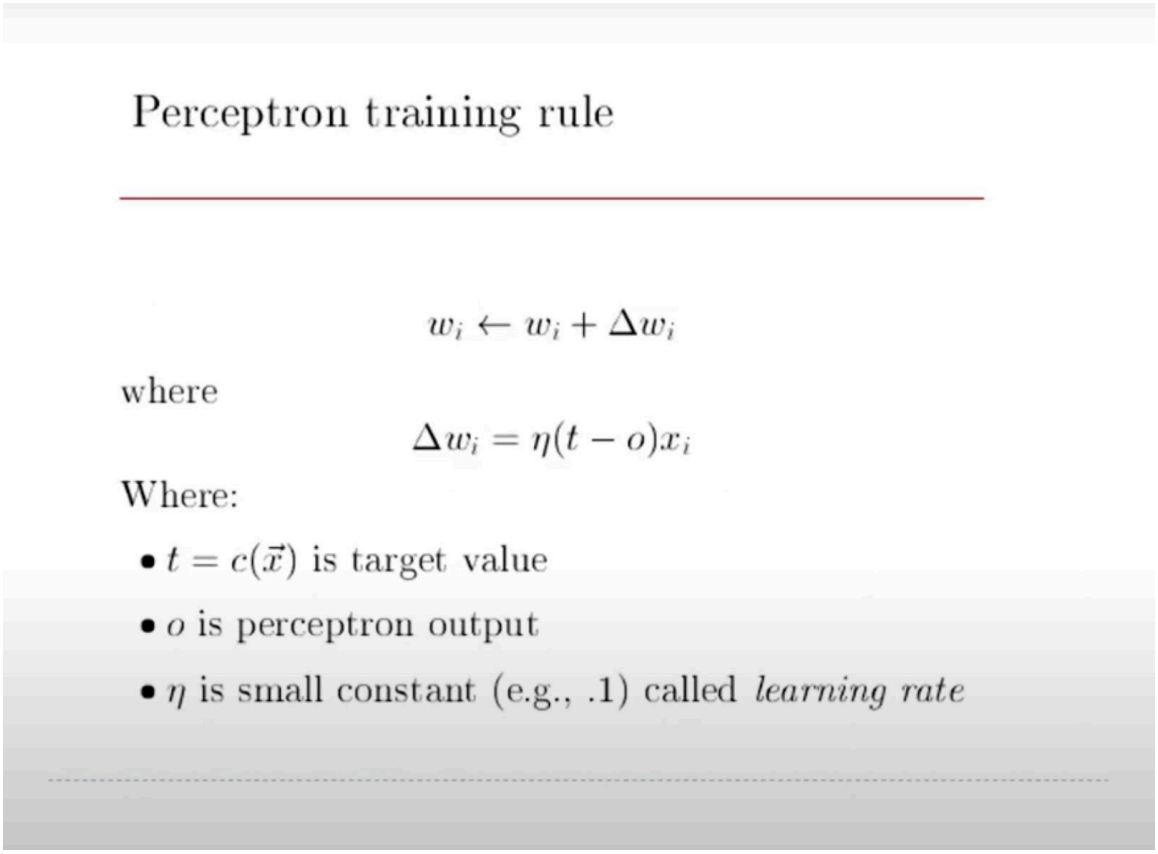
So sigmoid is one transfer function that you could use in a perceptron or in a neuron. This is the representation of one single neuron, one single neuron or one unit in a network of neurons which we will see how it gets developed. Now we have seen perceptron in a more pictorial way. But here is a more, it is a combination of both, you know, using diagram as well as using mathematical symbols to represent a neuron. And it is not difficult to understand, right.

It is the same representation in diagram but it is actually made horizontal than vertical here. So you can see different inputs and a bias summed up and weighted sum of the inputs here and instead of the sigmoid, you see a step function here. This is another perceptron with transfer function as a step function. What is a step function? Step function changes its state at one input value of the, at one input value. Its output changes at one value of the input.

And here you can see that when x changes from -1 to +1 or when it crosses 0 at 0th point, the output state changes. So that is what is represented in a more mathematical way here, that is the output is 1 if the weighted sum of inputs is greater than 0, otherwise it is -1, okay. That is a representation. And using vector notation, this is the expanded, output as a vector. In vector notation output using as a function of the x vector as input,

is 1 or -1 depending on the weighted sum of the input vector.

Input vector is the x vector. And there is also a weight vector which is, this is the weight vector, this is the input vector. So it is a product of the weight vector and the input vector, that determine the state of the output when you use step function, when you use step function as the transfer function. Very simple to understand.



So we build from here. Now, one aspect of a neural network is to understand how a neuron function, how it can be represented. It is sort of its description, or its internal dynamics is what actually you represent, in designing. So it shows, a perceptron only shows you the structure of a neuron, how is the internal structure of a neuron.

But what is equally important is, well I have neurons does not mean that I become intelligent. What is the other thing that is required? Learning. I need to be trained. I need to learn. So there has to be some method or some way or some algorithm that receives data and trains the neurons systematically, or in a way that you become more intelligent. As far as biological neuron is concerned, neural network is concerned, it is to make you intelligent and survive in this world.

So, and in our case when we look at neural networks we will be designing neural

networks to solve our problems. So that is the next thing that we are getting into as to how neurons can be trained. One is the structure of it, other is the training of it. And so in order for neurons to be trained as a network of neurons, what is the control that you have? What can you actually do? Or what is the control parameters that can be adjusted to train the neural network. That is the important question. In the case of human brain we live in an environment, data comes from the environment. Data is not something that you can control to a great extent when we live our life. We get exposed to a lot of environments. So data keeps coming.

But if something is hot, I should not touch that hot surface. So there has to be some adjustment within the neural networks that informs me when there is an input which is hot, immediately triggers an action, well that is a hot surface. So somewhere an adjustment has to be done to train my neural networks. Using the same principle, what can be controlled is the weight. There are weights or as in regression, you know, there are coefficients, which are estimated. In a similar way, in neural networks there are weights which can be adjusted in accordance with the data.

Or in other words neural networks training basically involves adjustment of internal weights based on the patterns in the data and that process is called training. That process of adjusting the internal weights based on patterns in the data is called training. Now how can you design a training algorithm. That is a key question.

And how can it, on what principle it can function? That is what is explained here with a simple example of a perceptron. So perceptron weight adjustment. If the original weight is W_i, a delta is the adjustment that happens, a delta. But how do you calculate the delta? Suppose there is an original weight W_i, but when something happens, when a new input data comes, that weight need to be adjusted. How is that adjustment done? It is based on a function which is η which is called learning rate, which is a learning rate.

Take it as a constant or a value that you control, like a hyper parameter. But t- o, you just notice here, that I am referring to a textbook from computer science and the notations have changed. If it was statistics, you would have used y and x. But in CS, in algorithmic world, the t stands for target, and o stands for output.

xi is the same, input. o, instead of y we are using o here, saying that it is an output, from a unit, here a linear unit. So t - o is what? t - o is the error. t- o is the error. t is the target, meaning actual. o is the output produced by the neuron. And when a neuron produces an output o, it should have produced an output which is t. There is a difference between that. And that gets multiplied by the xi also. You see that in the perceptron training, it is not only the error multiplied by a learning rate, but it also gets multiplied by the input. So if the input is high, the impact of the error will be higher on the weight.

If the input is low, the impact of error on weight will be low. So it also depends on xi. That is what you see here.

Now perceptron training here, as depicted here is a method of adjusting weights, increasing or decreasing existing weight based on the error that is produced by a neuron, as a function of error and also a learning rate and the input.

So you can see that the weight gets adjusted. If prior weight was W_i, after you input a tuple or a new data, the weight gets adjusted to W_i because the new data, output produced by the neuron was not exactly the target data. There was an error. So adjustment happens. So that is how perceptron training rule functions. So this is again an early training rule, in the, maybe in the 50s.

Gradient Descent

To understand, consider simpler *linear unit*, where

$$o = w_0 + w_1 x_1 + \cdots + w_n x_n$$

Let's learn w_i's that minimize the squared error

$$E[\vec{w}] \equiv \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2$$

Where D is set of training examples

- Perceptron training may not converge if the linear unit is not linearly separable
- $E[\vec{w}]$ is half the squared difference between the target output t_d the linear unit output o_d, summed over all training tuples.
- Here E is a function of weight vector, it is dependent on the linear unit output (Od).

I cannot really tell you what is exact date but, or the time, but we come to the next type of algorithm, called gradient descent algorithm. Gradient descent algorithm is an improvement on the perceptron training rule. The key difference is something that you can notice the way error e is calculated. e stands for error. Error is a function of weight, which is given by half d, a member of d, means small d is a tuple or one data, and from a population of d, d is the collection of all data points that is going to train the neuron.

It is the sum of the squared errors, sum of the, half of the sum of the squared errors for all the tuples, all the data points that is available or that you use to train a neuron. So the difference you see that earlier, in the case of perceptron you used error, but it was not squared. It is error squared here. There is a reason for squaring the error, that is because the convergence, this is again an optimization problem. The convergence is actually, is better achieved in a square function and it is also said that the perceptron training rule requires linearly separable inputs.

Linearly separable means, if there are two classes they should be separated by a line or if there are multiple classes there should be a space that clearly separates these classes. So data should have those characteristics, in which if not, if the data is not linearly separable, then convergence does not happen. In order to address that problem you have a error, the square of the error is used in the gradient descent algorithm. And you can see the error is the basis for designing the differentials or the weight vector, the weight vector or the delta of the weights to readjust the weight.

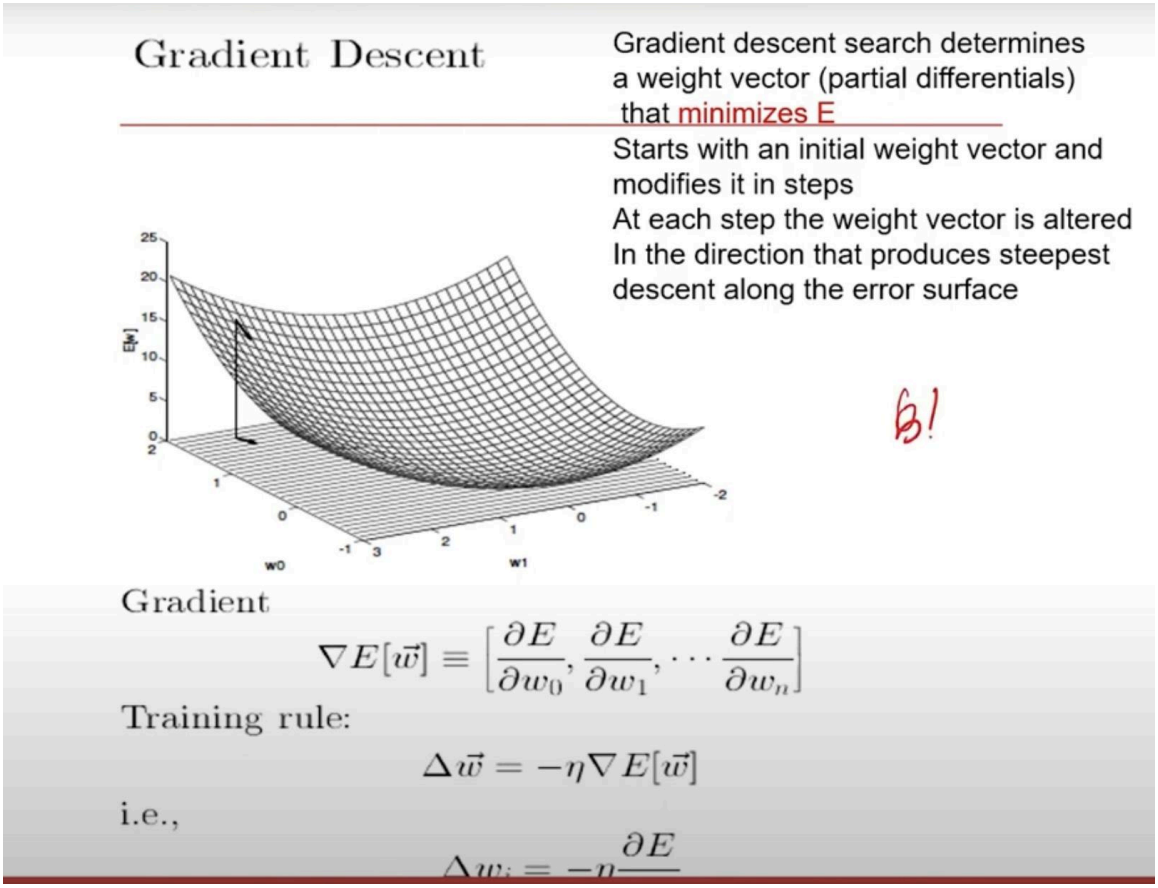
The next slide will explain this. Here, so E is a function of the weight vector, E is a function of the weight vector. Why you do not, do you see weight anywhere on the right side of the equation? As such it is not represented. You know that T d is a given tuple. This is the actual value, target is the actual data.

But there is O_d, O_d is what? O_d is the output of the neuron. And the output is a function of W_i. So and therefore, error is a function of the weights. Error is a function of weights because output is a function of weights. Sir, where is the weight? So that again. Where is the weight? Weights are in, very good question actually, you asked a good question.

Because if you see here, you know in our equations, you know it is like a weighted sum of the inputs. But actually when you design a neural network, we are going next there. You cannot actually express a neural network as a function of weights as we express in a multiple linear equation. This is often called a black box. Black box, it uses algorithms to train and recompute the weights based on algorithms like gradient descent algorithms.But they cannot be expressed in the form of a formula to understand or interpret the model. This model cannot be interpreted, using a equation. These are adjustment that happens inside the neural network.

But interpreting the model is not possible. It is not easy. And therefore they are called black box models. So if you ask me how the whole equation looks like, how the model looks like represented in a mathematical model, it is not, you know, it is not useful or it is not easy. So gradient descent algorithm, as we saw the principle in the last slide, it

uses a squared error function. And since there is a weight vector, not just one weight, there are multiple weights and in order to find the gradient, gradient means what? Gradient is the equivalent of a slope when you have multiple variables, right. When you have multiple variables, it becomes a surface float and it is the slope of the surface.



So the idea here is that the error which is a function of the weights should be descending. Descend means slowly reducing. Error should be reducing slowly as you adjust the weights, by calculating a differential, weight vector or partial differential. Here it is partial differential because there are multiple weights, okay. It is multiple weights and therefore it is a vector of partial differentials of error with respect to weights which is used to adjust the weights. It is used to adjust the weights with η as a learning rate. Essentially meaning with each tuple, you compute a delta weight vector to adjust the weights. So when you start, there are certain random weights. You input one input. The weights are adjusted based on the error. And you input the next input or next tuple, the weights are again adjusted by calculating the delta weight vector. So this is the training algorithm, okay. This is the training algorithm that constantly trains or iteratively trains a neural network and till the error actually becomes very low or becomes acceptable. Not that it becomes zero but the error should be decreasing or descending when you

constantly adjust the weights as a function of the error. So this is an explanation of how a gradient descent algorithm works for training neural networks.