Neural Network and Deep Learning

Deep Learning - Part 1

Dr.Sherif Saad

Learning Objectives

- Introduce the students to neural network
- Introduce the students to modern neural network techniques.

Outlines

Introduction Multilayer Neural Network

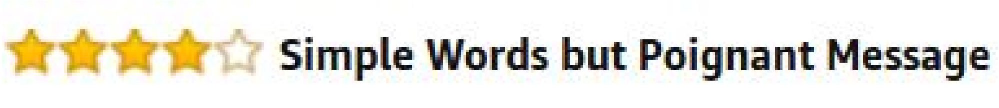
Backpropagation Neural Network

Deep Learning

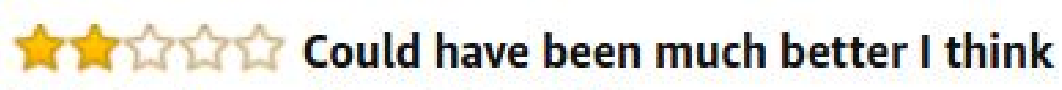
Introduction To Multilayer Neural Network

Grace wants to purchase a book **"Milk and Honey"** from Amazon. She trusts Alice and Bob opinions. She asked them since they read the book. Alice gave the book a rating score 4 stars out of 5. Bob gave the book 2 stars out of 5.

Alice vs Milk and Honey



Bob vs Milk and Honey



What should Grace do?

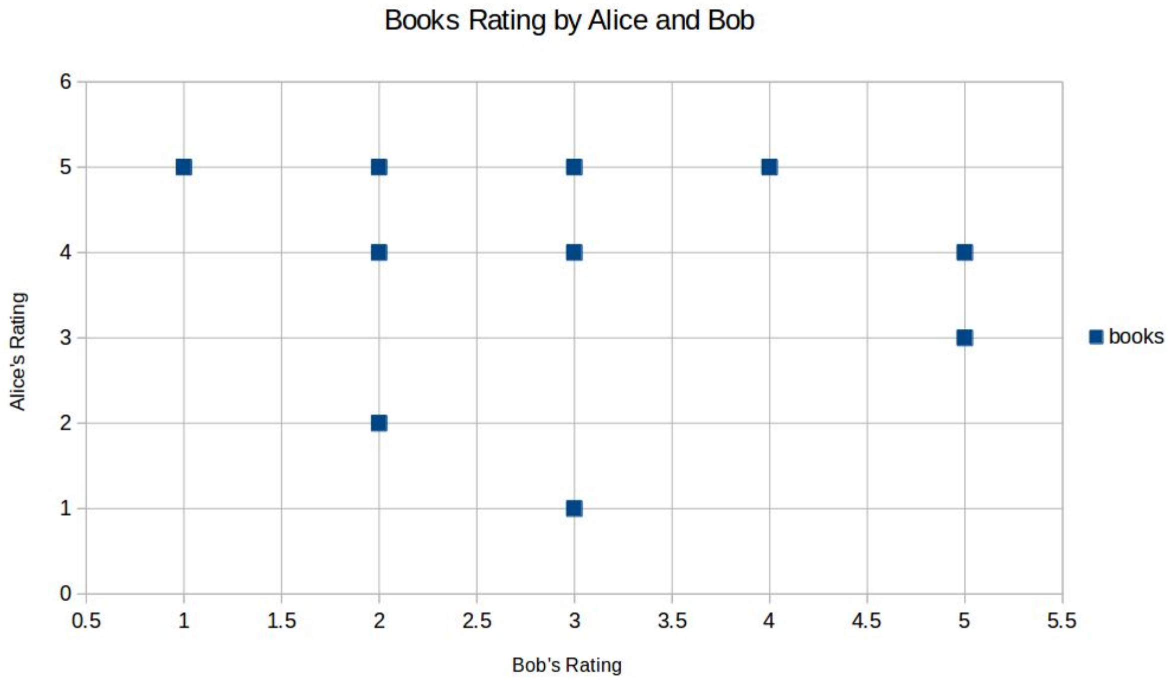
Introduction To Multilayer Neural Network

How could Grace decide if she buys the book or not?

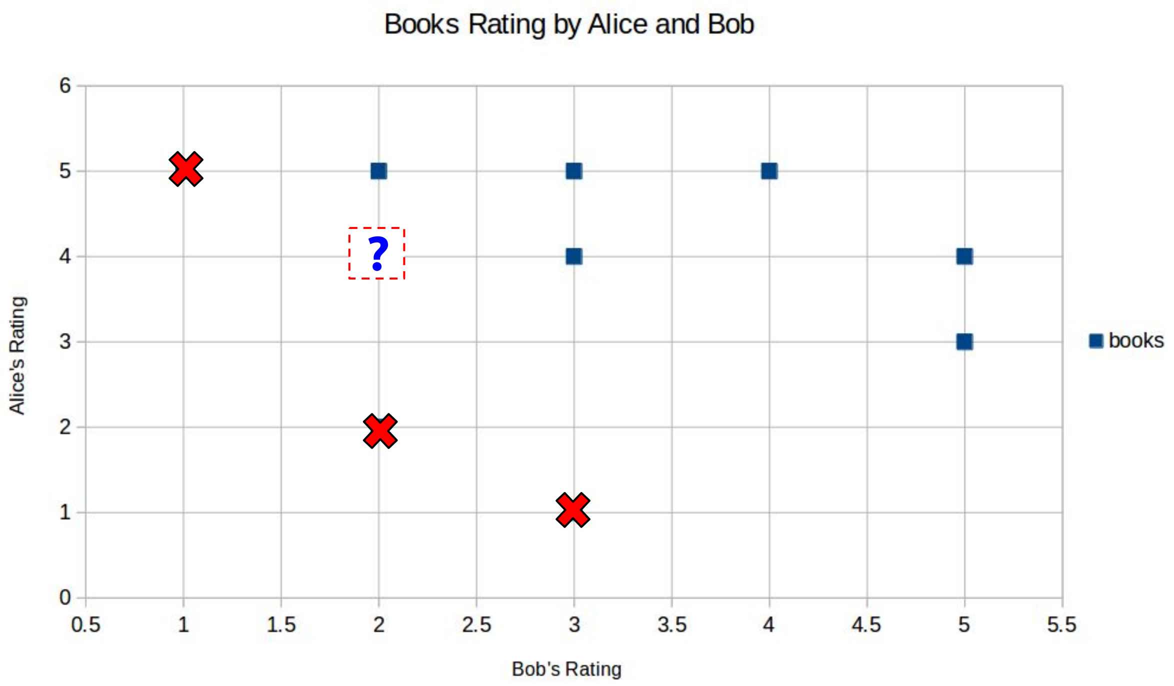
Thankfully, Grace has kept notes of Bob and Alice ratings for some books in the past. For each book, She also noted whether She liked the book or not.

Book Name	Bob's Rating	Alice's Rating	Grace Like it?
Q&A a Day: 5-Year Journal	3	4	Yes
The Shack	5	3	Yes
Secret Garden	1	5	No
Dark Matter	2	5	Yes
The Maze Runner	3	1	No
The Secret Life of Bees	4	5	Yes
The Couple Next Door	2	2	No
All the Missing Girls	5	4	Yes
Camino Island	3	5	Yes
Milk and Honey	2	4	?

Introduction To Multilayer Neural Network



Introduction To Multilayer Neural Network



Introduction To Multilayer Neural Network

Grace's problem could be modeled as a simple perceptron classification problem. Each book rating by Alice and Bob is an observation of two features x_1 (Bob' Rating) and x_2 (Alice's Rating). Then we have a our decision or mapping function as a as simple as a weighted linear combination of Alice's and Bob's ratings

$$z = x_0 w_0 + x_1 w_1 + x_2 w_2$$

$$f(z) = \begin{cases} 1 & z \geqslant 0 \\ -1 & z \leqslant 0 \end{cases}$$

Introduction To Multilayer Neural Network

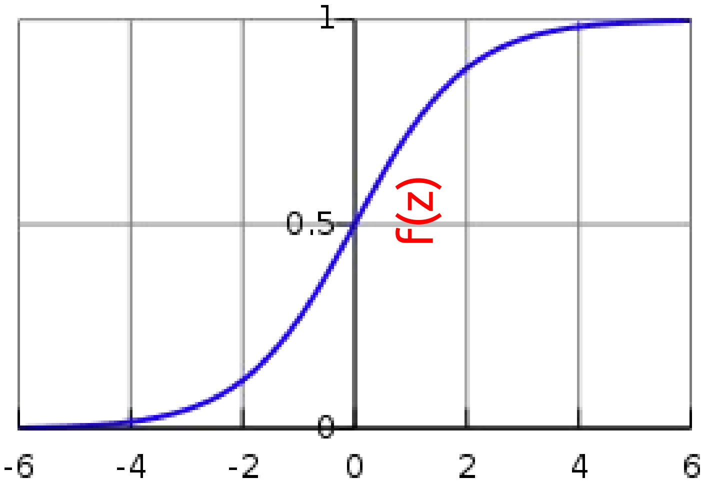
We can replace the piecewise unit step function with the sigmoid function to map all the outputs (decisions) to a value between zero and one. Then using booking rating data and stochastic gradient descent we can learn the weight by minimizing the error rate in f(z)

$$z = x_0 w_0 + x_1 w_1 + x_2 w_2$$

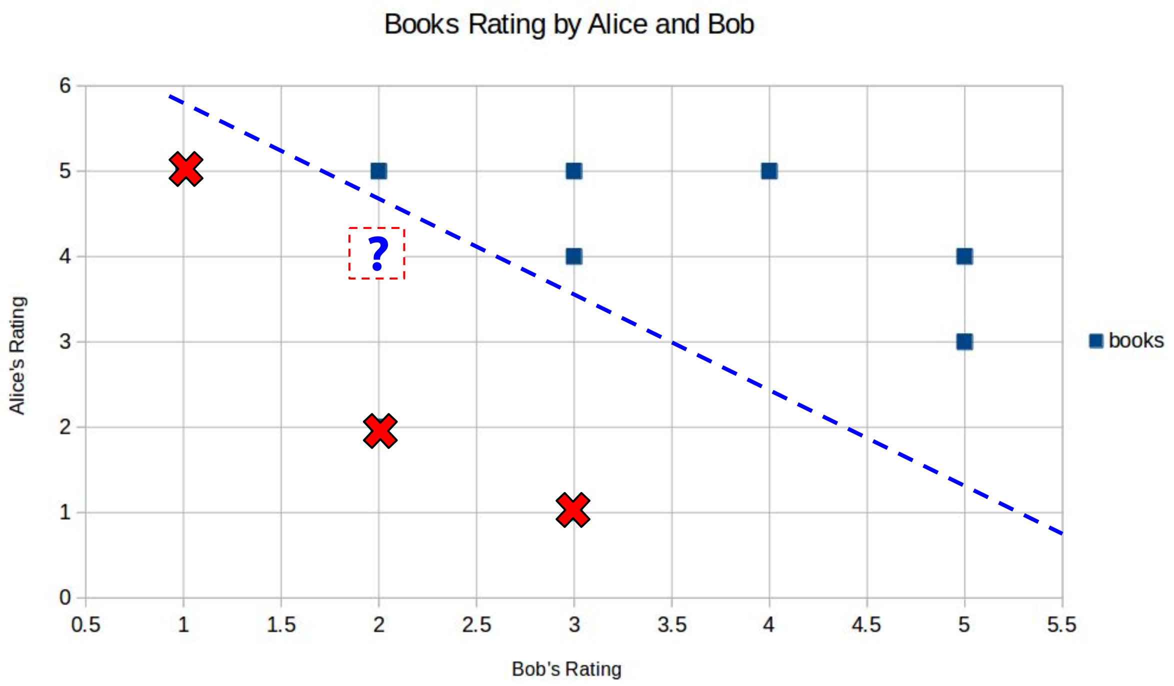
$$f(z) = \frac{1}{1 + e^{-z}}$$

$$new(w_i) = wi + \Delta w_i$$

$$\Delta w_i = \alpha(l - p)x_i$$

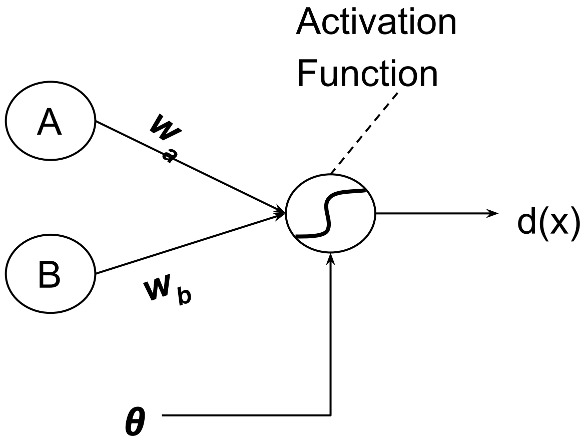


Introduction To Multilayer Neural Network



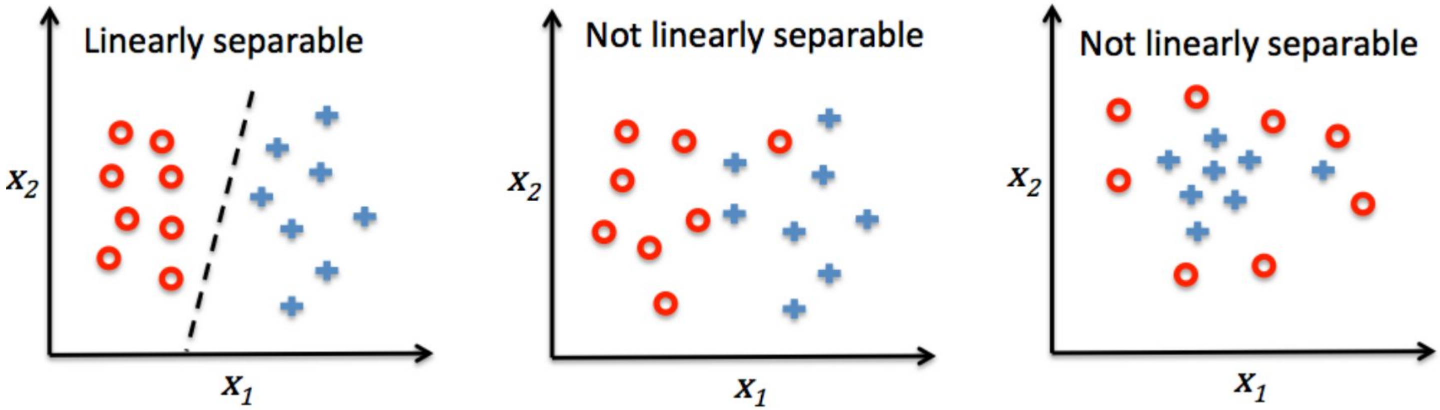
Introduction To Multilayer Neural Network

The following illustration explains the classification of any new book as the result of the rating of Alice times the weight associated with Alice Rate and the rating of Bob times the weight of Bob rating added with $\boldsymbol{\theta}$ (bias) and finally we apply to this value the sigmoid function



Introduction To Multilayer Neural Network

The two classes (like or dislike a book) are linearly separable (we can draw a linear decision function to separate the positive and the negative examples.). If the two classes can't be separated by a linear decision boundary then we need to use a non-linear model

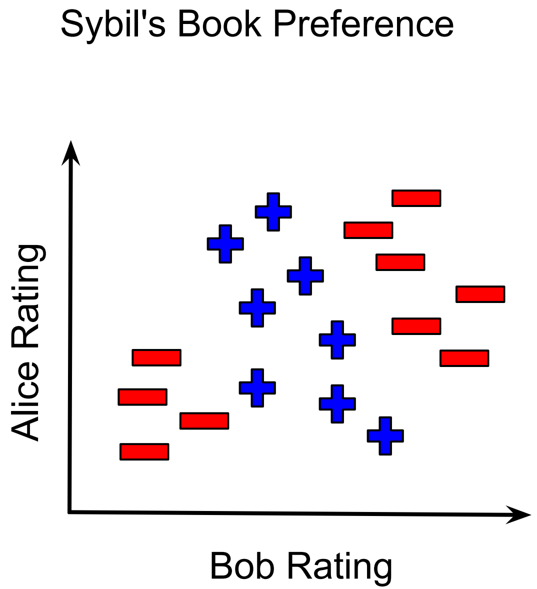


Multilayer Neural Network

It is clear that the linear decision boundary is limited to linearly separable data.

Sybil has book preference based on Bob and Alice rating. However, it is not linearly separable.

How can we define a mapping or decision function for Sybil based on the available data?

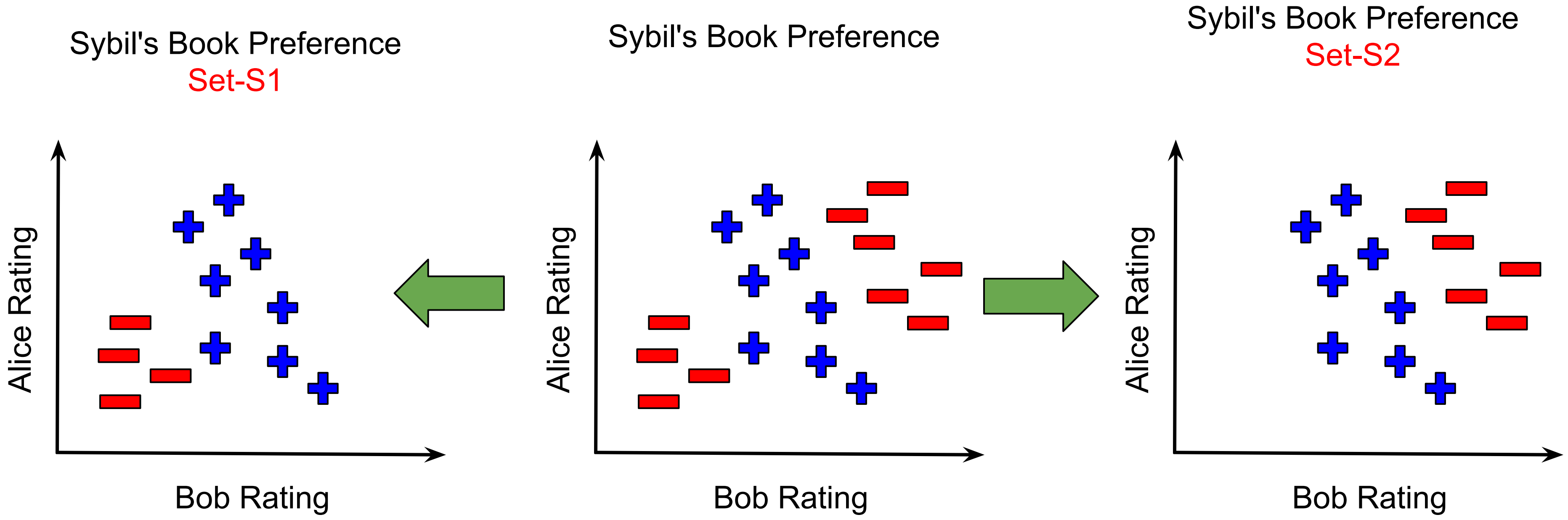


Multilayer Neural Network

One way to handle the nonlinearly separable data. Is to use a divide and conquer technique, as follows

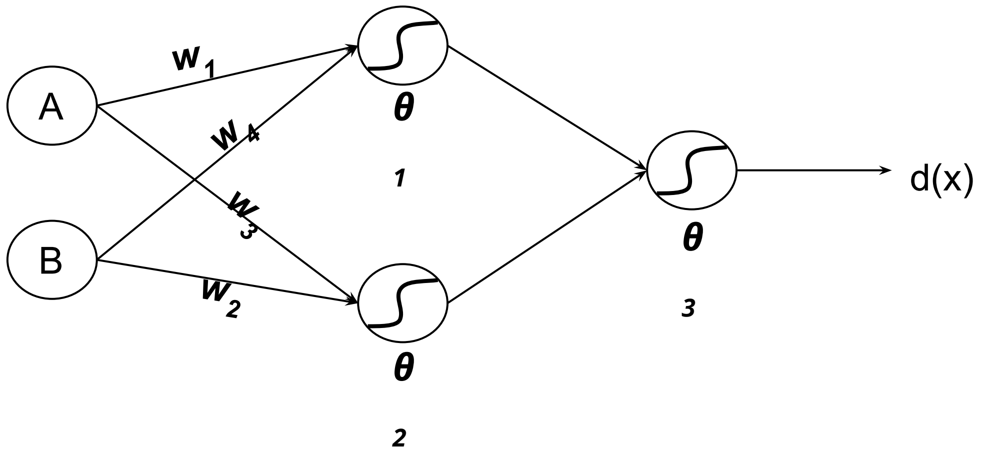
1. Divide the data into two or more sets. Each set can be simply classified by a linear decision. Learn the decision function for each set
2. Use the learned decision functions and compute the decision values for each example. Then treat these values as input to another decision function.
3. Use stochastic gradient descent to find the final decision function.

Multilayer Neural Network



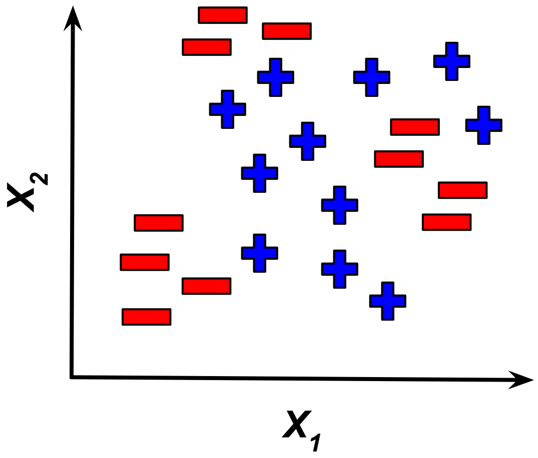
Multilayer Neural Network

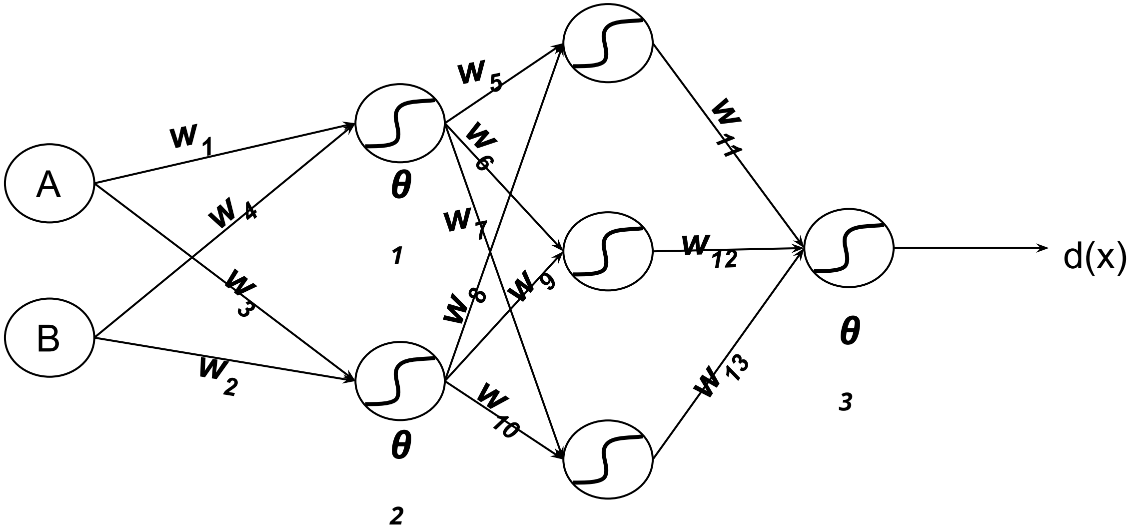
By breaking down the nonlinearly separable data into subsets of linearly separable data and learn the decision function for each set and combine them into one final decision function we constructed a neural network. It has one hidden layer, which has two “neurons and one output layer which compute the final decision.



Multilayer Neural Network

If we have more complex function to learn we need to have a multilayer neural network





Backpropagation Neural Network

The work in neural network started in 1940s by Warren McCulloch and Walter Pit.

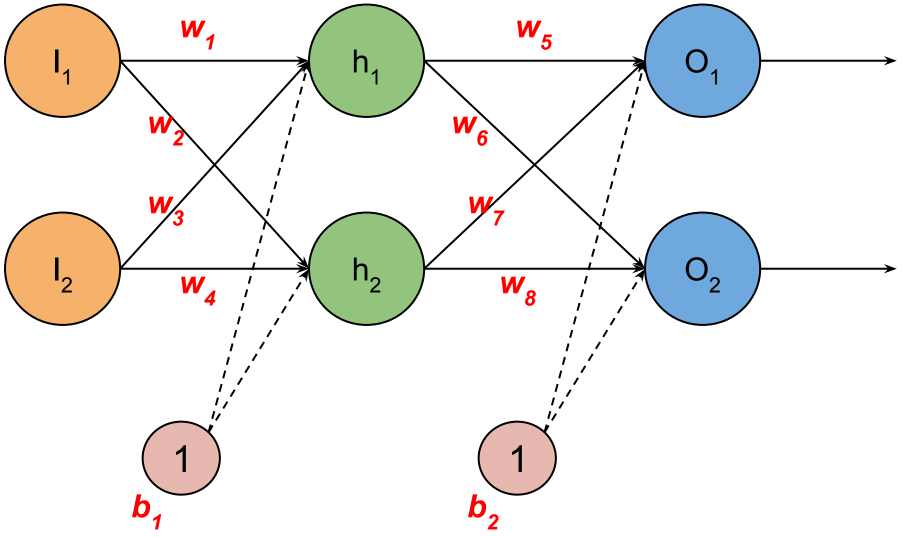
In 1950s Rosenblatt's proposes the perceptron algorithm.

Machine learning practitioners slowly began to lose interest in neural networks since no one had a good solution for training a neural network with multiple layers.

Interest in neural networks was rekindled in 1986 when D.E. Rumelhart, G.E. Hinton, and R.J. Williams were involved in the (re)discovery and popularization of the backpropagation algorithm to train neural networks more efficiently

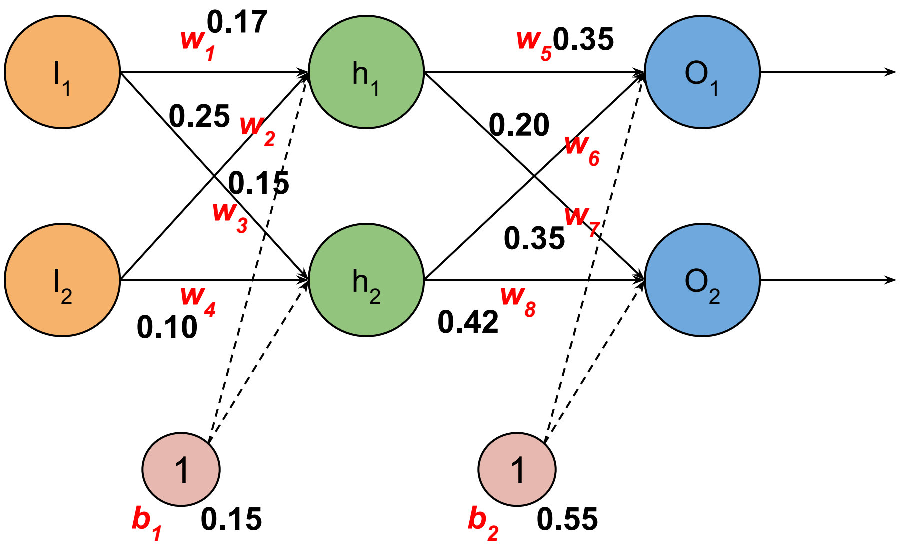
Backpropagation Neural Network

How could we learn the weights for a multilayer neural network?



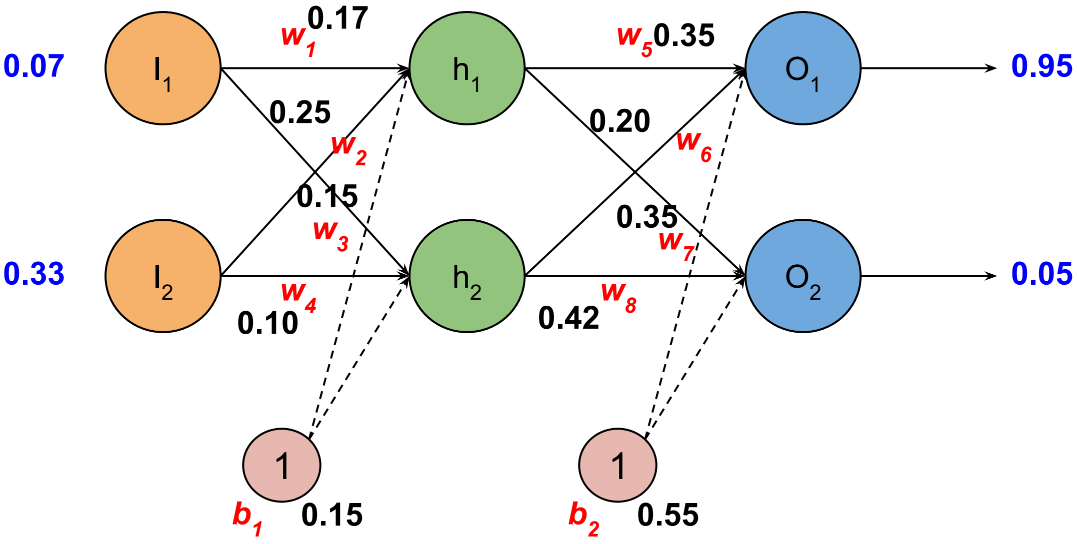
Backpropagation Neural Network

Step 1: Initialize the weights using some random number



Backpropagation Neural Network

Step 2: start online learning by feeding an input sample and expect a known output



Backpropagation Neural Network

Step 3: Execute the forward Pass

- In calculate the net input to each hidden layer neuron, apply to it the activation function (e.g. sigmoid) then repeat the process with the output layer neurons.
- Then calculate the total error of the output layer by calculating the squared error function for each output neuron. The total error will be the sum of error of all the neurons in the output layer.

Backpropagation Neural Network

Step 3: Execute the forward Pass

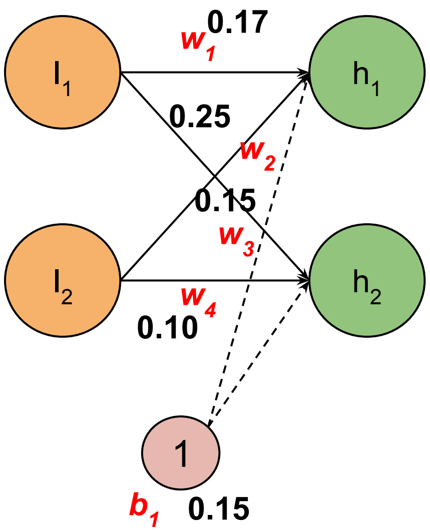
- In calculate the net input to each hidden layer neuron, apply to it the activation function (e.g. sigmoid) then repeat the process with the output layer neurons.

$$net(h_1) = w_1 * I_1 + w_2 * I_2 + b_1 * 1$$

$$act(h_1) = \frac{1}{1 + e^{-net(h_1)}}$$

$$net(h_2) = w_3 * I_1 + w_4 * I_2 + b_1 * 1$$

$$act(h_2) = \frac{1}{1 + e^{-net(h_2)}}$$



Backpropagation Neural Network

Step 3: Execute the forward Pass

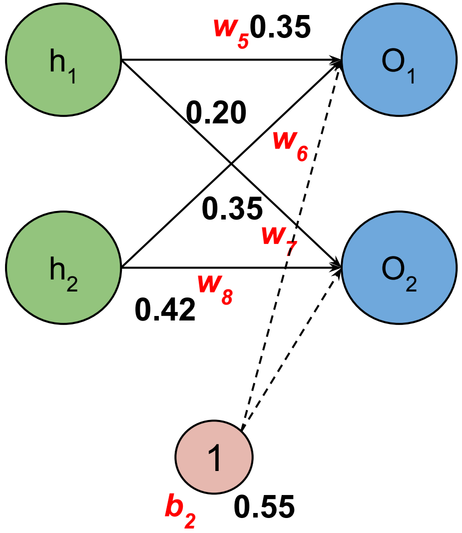
- In calculate the net input to each hidden layer neuron, apply to it the activation function (e.g. sigmoid) then repeat the process with the output layer neurons.

$$net(O_1) = w_5 * h_1 + w_6 * h_2 + b_2 * 1$$

$$act(O_1) = \frac{1}{1 + e^{-net(O_1)}}$$

$$net(O_2) = w_7 * h_1 + w_8 * h_2 + b_2 * 1$$

$$act(O_2) = \frac{1}{1 + e^{-net(O_2)}}$$



Backpropagation Neural Network

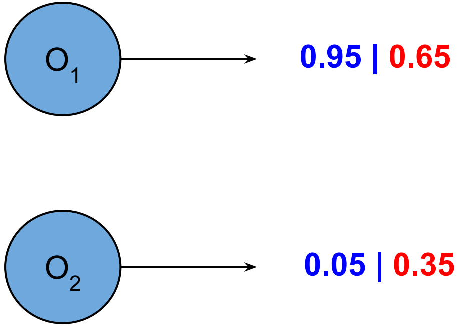
Step 3: Execute the forward Pass

- Then calculate the total error of the output layer by calculating the squared error function for each output neuron. The total error will be the sum of error of all the neurons in the output layer.

$$Error(O_1) = \sum \frac{1}{2}(target_{O_1} - output_{O_1})^2$$

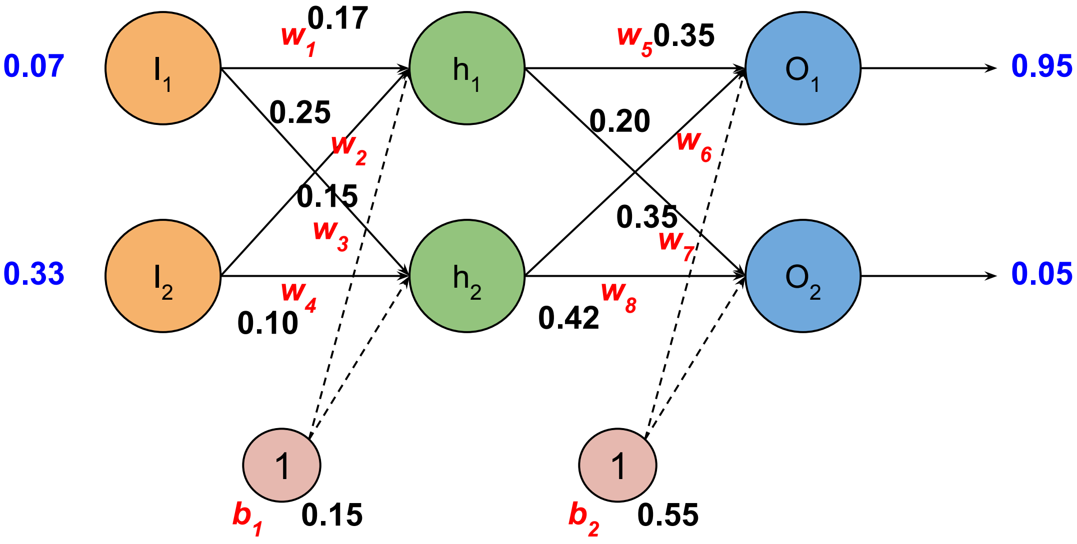
$$Error(O_2) = \sum \frac{1}{2}(target_{O_2} - output_{O_2})^2$$

$$Error_{total} = Error(O_1) + Error(O_2)$$



Backpropagation Neural Network

Now that we have the total error how can we adjust the weights to correct this error



Backpropagation Neural Network

Now that we have the error we got to step 4

Step 4: Execute the Backward Pass

We feed the output starting from the output layer and backward into each layer of the network.

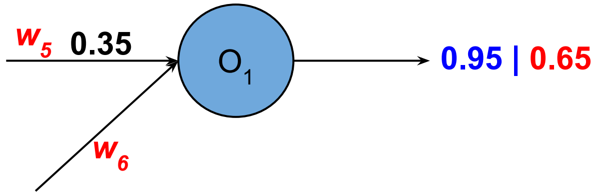
Here the backpropagation step update each of the weights in the network so that they cause the actual output to be closer to the target.

Backpropagation Neural Network

Step 4: Execute the Backward Pass

$$\frac{\partial Error_{total}}{\partial w_5} = \frac{\partial Error_{total}}{\partial act(O_1)} * \frac{\partial act(O_1)}{\partial net(O_1)} * \frac{\partial net(O_1)}{\partial w_5}$$

- How much does the total error change with respect to the output?
- How much does the O_1 change with respect to its total net input?
- How much does the total net input of O_1 change with respect to w_5?



To answer these questions we relies on some basic calculus concepts derivatives, partial derivatives, and the chain rule

Backpropagation Neural Network

Step 4: Execute the Backward Pass

We continue the backward pass by calculating the new values for each weight in each layer in the neural network

We only use the updated weight when we complete the backward pass. In other words we use the original weights, not the updated weights, when we continue the backpropagation

Deep Learning

What is Deep Learning?

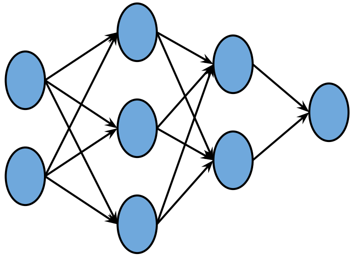
- We call any neural network with 2 or more hidden layers as deep neural network.
- We use the term deep learning to refer to algorithms we use to train deep neural network.
- To refer to new algorithms that make use of unlabeled data
- Set of algorithms to learn features that we can use to construct neural networks.
- The a set of algorithms that use neural networks as an architecture, and learn the features automatically.

Deep Vs Shallow Networks

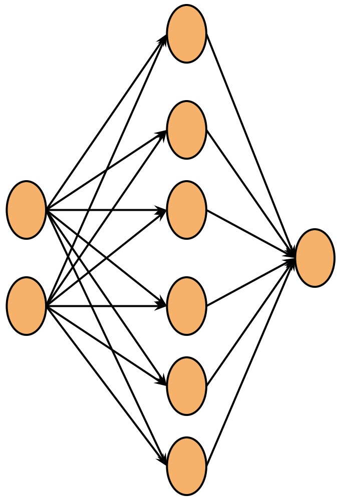
For nonlinear functions , deep networks seem computationally more attractive than shallow networks.

Experiments showed that in order to get to the same level of performances of a deep network, we need shallow network with many more connections (about 10X connections).

The layered structure in general has better performance and effectively model nonlinear functions



Deep Network



Shallow Network

When To Use Neural Networks

Neural Networks are good with problems that deal with natural data (text, image, video stream, speech, etc) because these data has highly nonlinear properties. Neural networks have a lot of parameters, and can approximate very nonlinear functions.

Need large or massive number of labeled data to train and reach good performance.

Sine the late 2000s , neural networks have recovered and become more successful thanks to the availability of inexpensive computing power and a massive amount of labeled data.

Questions