The Neural Network Model

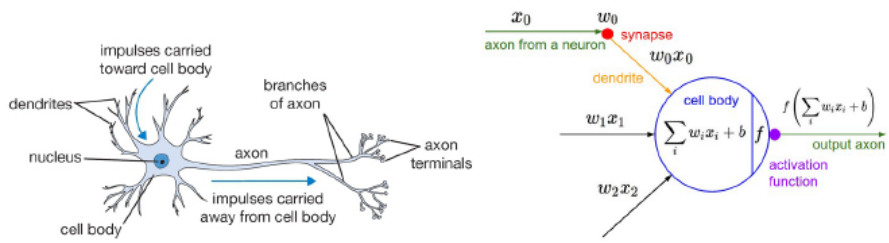

Figure 1: (Left) A biological neuron. (Right) An artificial neuron model.

A neuron is the basic computational unit of the brain. It has a sophisticated structure, with many parts (see figure above). However its working can be explained by simplifying it, and considering a mathematical model called the artificial neuron instead. The artificial neuron is defined by specifying an activation function and a set of parameters called weights (a vector w) and bias (which is a real number b). It takes an input vector x, and computes an output y (a real number) according to the formula $y = f(\sum_i w_i x_i + b)$.

It is similar to the linear classifier except having an activation function. The artificial neuron is studied using various activation functions like sign (1 for +ve, -1 for -ve inputs), sigmoid function $1/(1+e^{-x})$ etc.The advantage of sigmoid function over the sign function is that it is differentiable, which is required for the gradient descent algorithm (which we will see later) to work properly. The activation function makes the artificial neuron a non linear function.Though the artificial neuron was first proposed for studying the brain, it can also be used as a binary classifier in machine learning.

Introduction to Perceptron

In machine learning, the perceptron is an algorithm for supervised learning of binary classifiers. It is a type of linear classifier, i.e. a classification algorithm that makes all of its predictions based on a linear predictor function combining a set of weights with the feature vector.

The linear classifier says that the training data should be classified into corresponding categories such that if we are applying classification for two categories, then all the training data must be lie in these two categories.

The binary classifier defines that there should be only two categories for classification.

The basic perceptron algorithm is used for binary classification and all the training examples should lie in these categories. The term comes from the basic unit in a neuron, which is called the perceptron.

Origin of the Perceptron

According to Wikipedia:

The perceptron algorithm was invented in 1957 at the Cornell Aeronautical Laboratory by Frank Rosenblatt, funded by the United States Office of Naval Research. The perceptron was intended to

be a machine, rather than a program, and while its first implementation was in software for the IBM 704, it was subsequently implemented in custom-built hardware as the Mark 1 perceptron. This machine was designed for image recognition. It had an array of 400 photocells randomly connected to the neurons. Weights were encoded in potentiometers, and weight updates during learning were performed by electric motors.

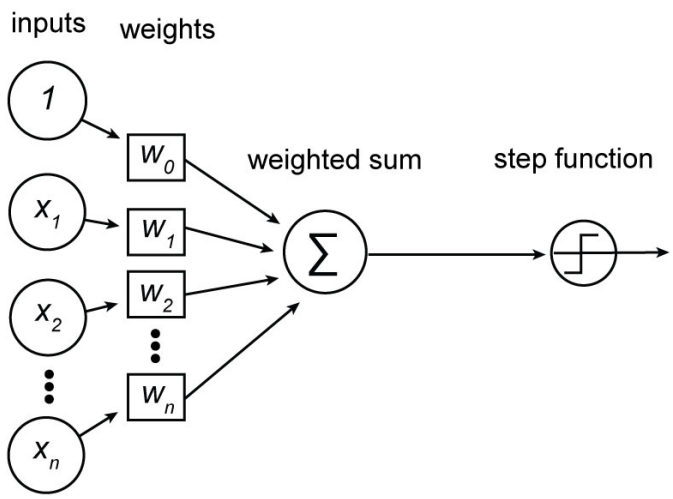

Figure 2

Following are the major components of a perceptron:

Input: All the features become the input for a perceptron. We denote the input of a perceptron by $[x_1, x_2, \ldots, x_n]$, where x represents the feature value and n represents the total number of features. We also have special kind of input called the bias. In the image, we have described the value of the BIAS as w_0.

Weights: The values that are computed over the time of training the model. Initially, we start the value of weights with some initial value and these values get updated for each training error. We represent the weights for perceptron by $[w_1, w_2, \ldots, w_n]$.

Bias: A bias neuron allows a classifier to shift the decision boundary left or right. In algebraic terms, the bias neuron allows a classifier to translate its decision boundary. It aims to "move every point a constant distance in a specified direction." Bias helps to train the model faster and with better quality.

Weighted summation: Weighted summation is the sum of the values that we get after the multiplication of each weight $[w_n]$ associated with the each feature value $[x_n]$. We represent the

weighted summation by $\sum w_i x_i$ for all $i -> [1 \text{ to } n]$.

Step/activation function: The role of activation functions is to make neural networks nonlinear. For linear classification, for example, it becomes necessary to make the perceptron as linear as possible.

Output: The weighted summation is passed to the step/activation function and whatever value we get after computation is our predicted output.

References

1. Neural Networks Course Notes / Blog
   - `http://cs231n.github.io/neural-networks-1/`
   - `https://ujjwalkarn.me/2016/08/09/quick-intro-neural-networks/`
2. Perceptron (with Math Proofs)
   - `https://www.cs.cmu.edu/~avrim/ML10/lect0125.pdf`
3. Perceptron (slides)
   - `http://aass.oru.se/~lilien/ml/seminars/2007_02_01b-Janecek-Perceptron.pdf`