Knowledge Share

Introduction to Neural Networks

entelect
everything is possible

Privolin Naidoo

November 2017

Contents

- Overview
- History of Neural Networks
- Why use Neural Networks
- Perceptron
- Feed-Forward Neural Networks
- Training and Visualizing Neural Networks

Overview

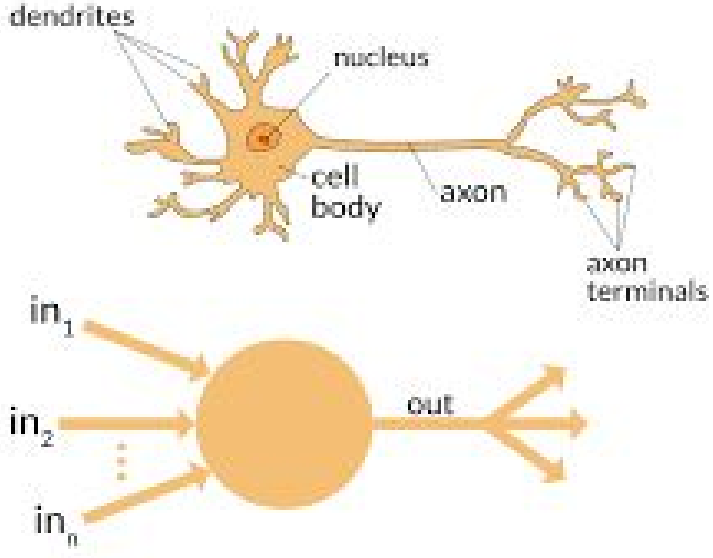
- Neural Networks are a machine learning technique modelled around the human brain.
- Specifically the neurons in the brain.
- It is used for a variety of tasks:
  - Image Processing
  - Speech Recognition
  - Machine Translation
  - Medical Diagnosis
  - ‘Artificial Intelligence’
  - Prediction, Classification, Regression, etc.

Brief History of Neural Networks

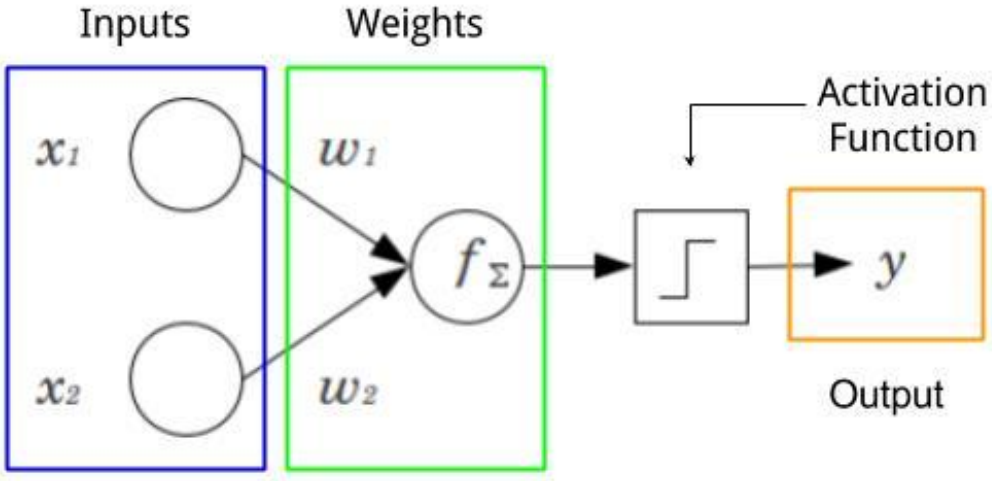
- Neural Networks have a unique history.
- They have been around since the 1950's
- When they were initially introduced, they were immediately popular, but fell away shortly after.
- Decades after its creation, Neural networks are now one of the most popular machine learning algorithms being used in the world today.

Perceptrons/Artificial Neurons

- These are the building blocks of the neural network.
- Modelled on the neurons in our brain.



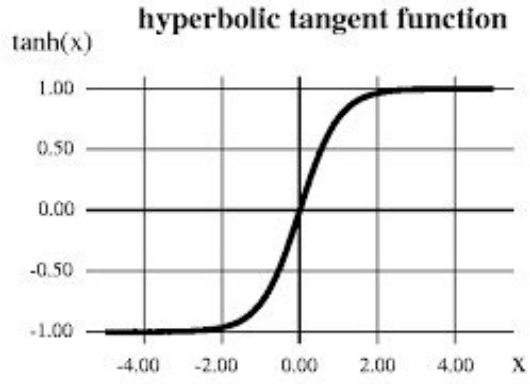
Perceptrons/Artificial Neurons

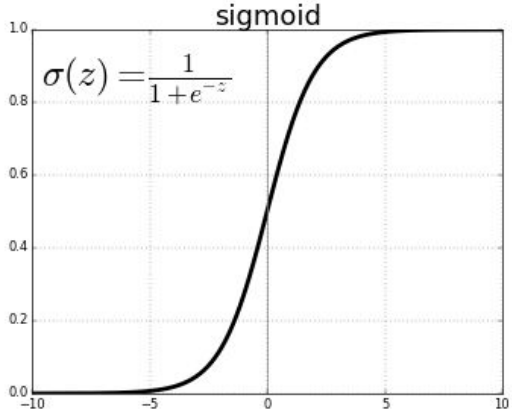


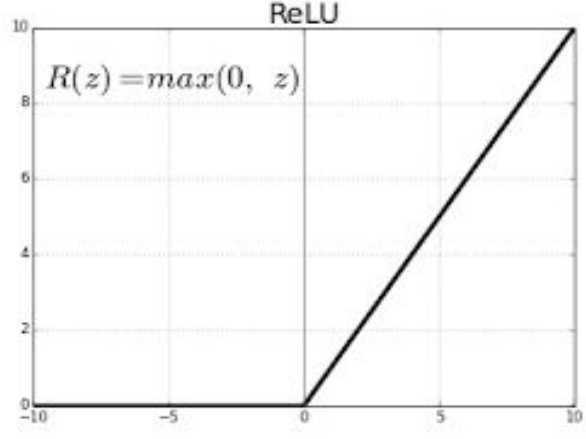
Activation Functions

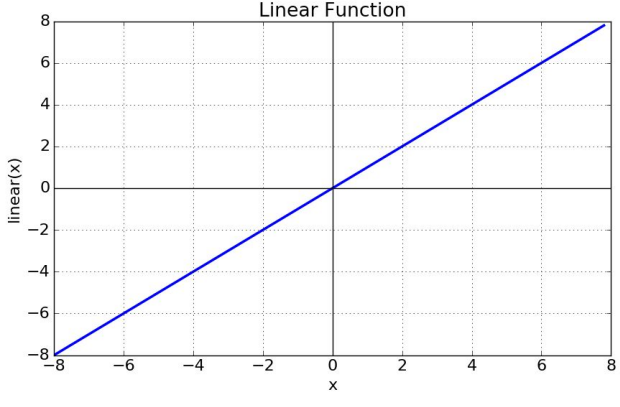
- The activation functions are simple calculations which are used to transform the inputs.
- These are loosely modelled on what researchers think our neurons do.
- One of the reasons of using simple functions, is that we also require their derivatives during training.
- Examples of Activation Functions
  - Rectifier Linear Unit (ReLU)
  - tanh (hyperbolic tangent function)
  - sigmoid
  - Linear Function

Activation Functions

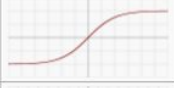
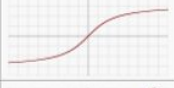






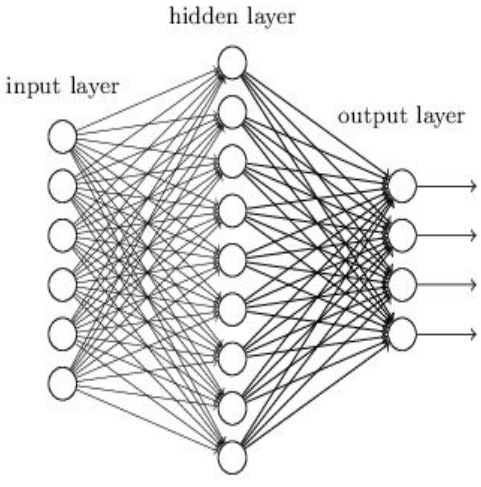


Activation functions

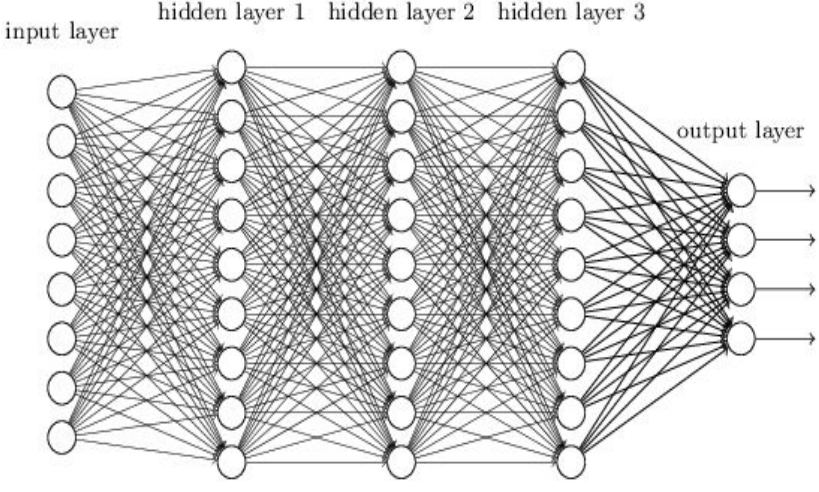
Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1-f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2+1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU) [2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) [3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1+e^x)$	$f'(x) = \frac{1}{1+e^{-x}}$

Topology of the FF Neural Network

"Non-deep" feedforward neural network



Deep neural network



Feed Forward Neural Networks

Input Layer

- Stream of input features, each node represents a different feature

Hidden Layer

- A layer of Perceptrons
- Outputs of a hidden layer perceptrons are inputs for the next layer

Output Layer

- Final layer of perceptrons in the network.
- Nodes represents the different classes or objectives of the neural Network

Training FF Neural Networks

- The weights of the inputs to each layer are being estimated/trained.
- Regularization inputs to each layer, to avoid overfitting.
- Uses Gradient Descent in the training process on output layer
- Back propagates the changes in weight using output of gradient descent.

Training and Visualising Neural Networks

playground.tensorflow.org/

R Implementation of a FF Neural Network

github.com/Privolin/Neural_Networks_R