ALGONQUIN COLLEGE

CST8502
MACHINE LEARNING

Neural Networks

Professor : Dr. Anu Thomas
Email: thomasa@algonquincollege.com
Office: T315

Agenda

- What is a neuron?
- What is a neural network?
- Types of Neural Networks
  - Perceptron
  - Multi-layer perceptron or feedforward
    - Back Propagation
  - CNN, RNN

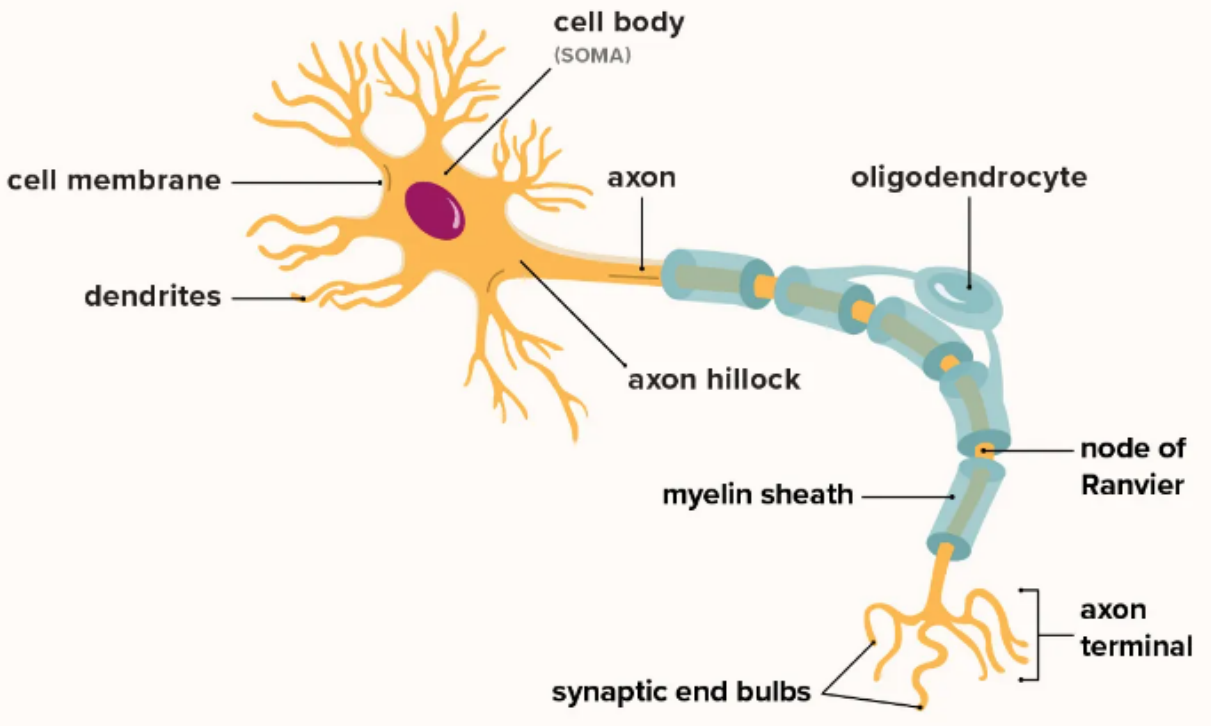
Why we need Neural Networks?

What is a Neuron?

- Fundamental units of the brain and nervous system
  - the cells responsible for receiving sensory input from the external world, for sending motor commands to our muscles, and for transforming and relaying the electrical signals at every step in between
- roughly 100 billion neurons do interact closely with other cell types

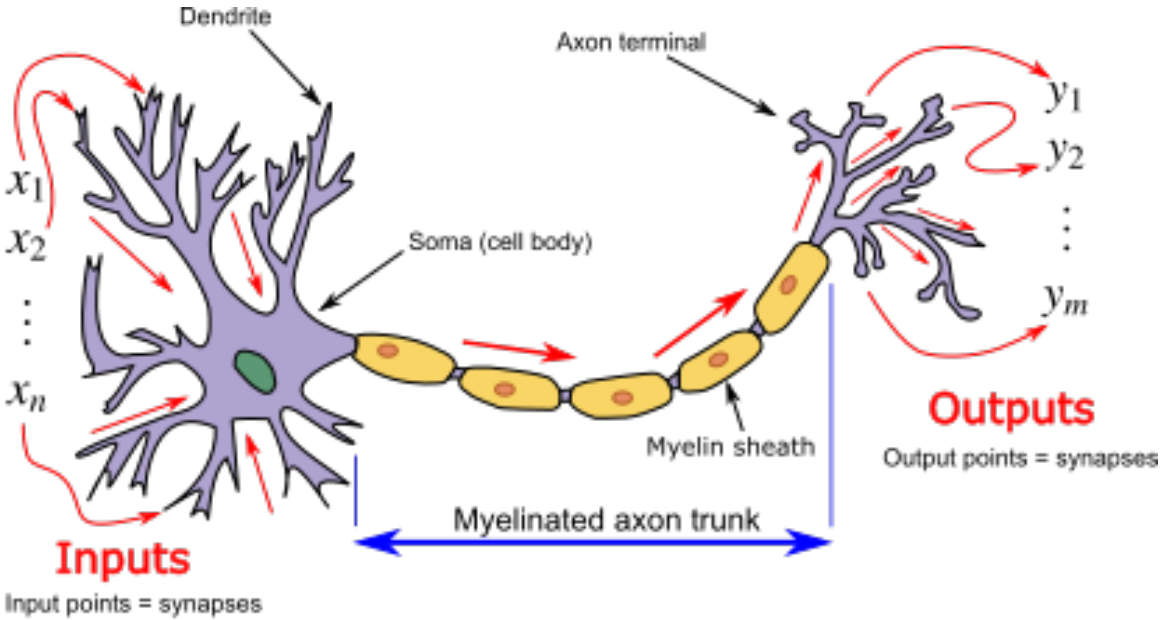
Taken from: https://qbi.uq.edu.au/brain/brain-anatomy/what-neuron

Structure of a Neuron



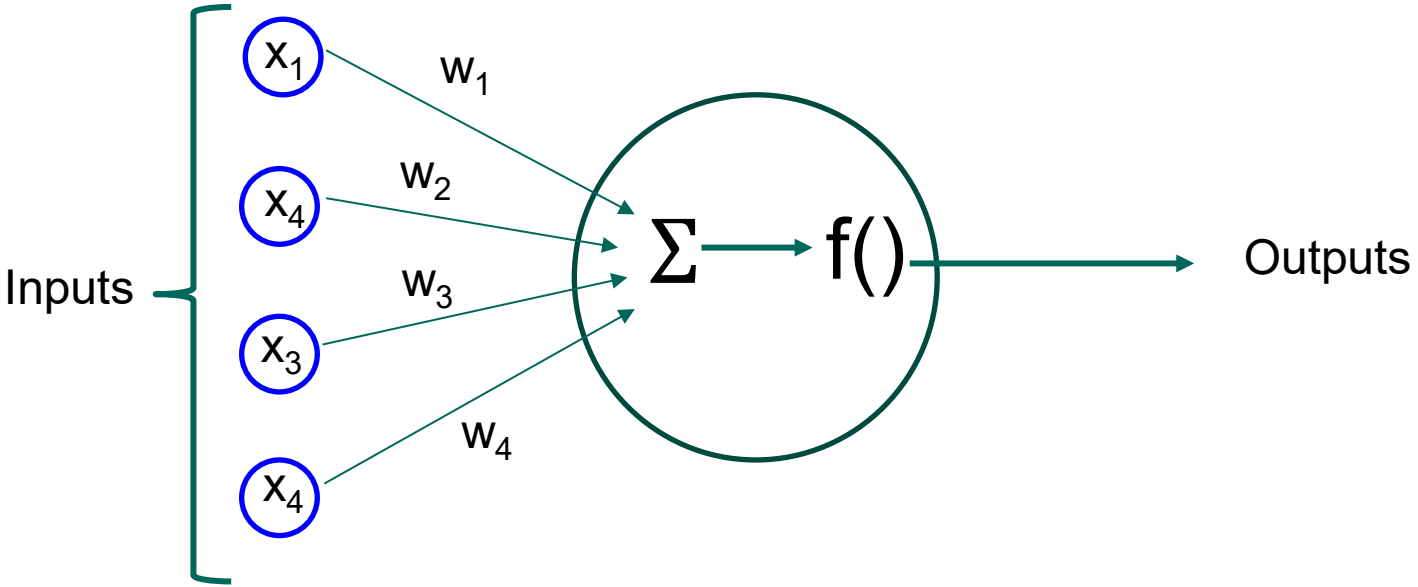
Taken from: https://www.healthline.com/health/neurons

Biological Neuron Model



Neuron and myelinated axon, with signal flow from inputs at dendrites to outputs at axon terminals. The signal is a short electrical pulse called action potential or 'spike'.

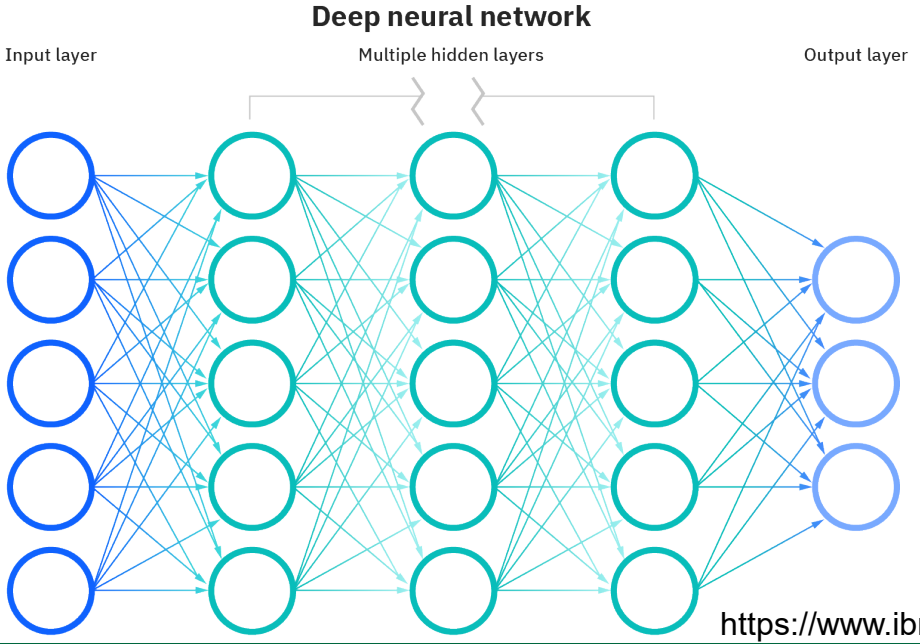
Artificial Neuron



As these neurons are artificial, NN is often called as Artificial NN

What is a Neural Network?

- Biologically motivated approach to ML
- Fundamental processing unit of a NN is a Neuron

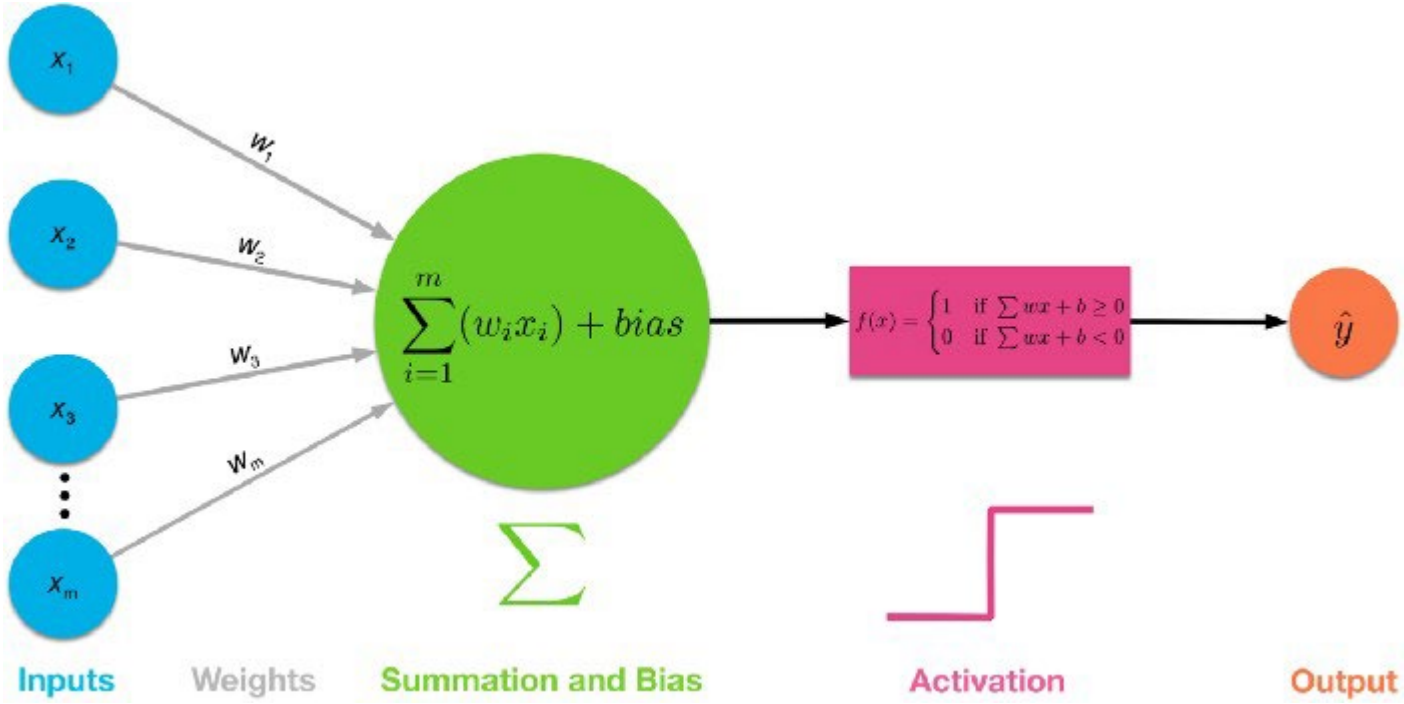


https://www.ibm.com/cloud/learn/neural-networks

Neural Network

- https://www.youtube.com/watch?v=ER2It2mIagI&list=PLeo1K3hjS3uu7CxAacxVndI4bE_o3BDtO&index=6&ab_channel=codebasics
- https://www.youtube.com/watch?v=bfmFfD2RIcg&t=223s

Single-layer Perceptron Network



Taken from: https://towardsdatascience.com/multi-layer-neural-networks-with-sigmoid-function-deep-learning-for-rookies-2-bf464f09eb7f

Activation Function

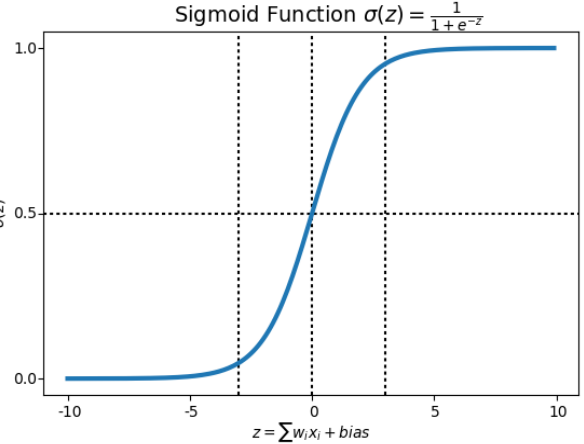
Sometimes, activation function is as simple as:

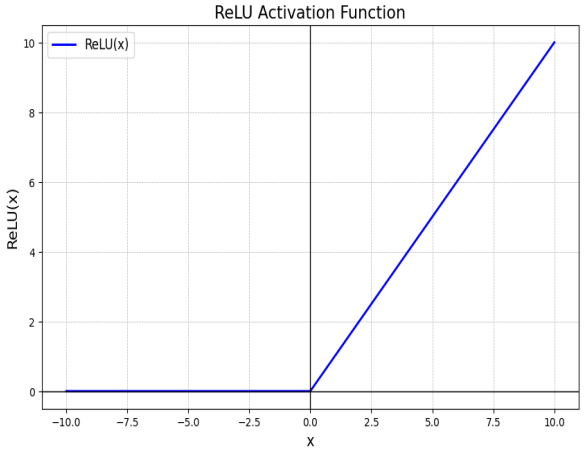
$$Output = \begin{cases} 1 \quad if \sum w_i x_i + b \geq 0 \\ 0 \quad if \sum w_i x_i + b < 0 \end{cases}$$

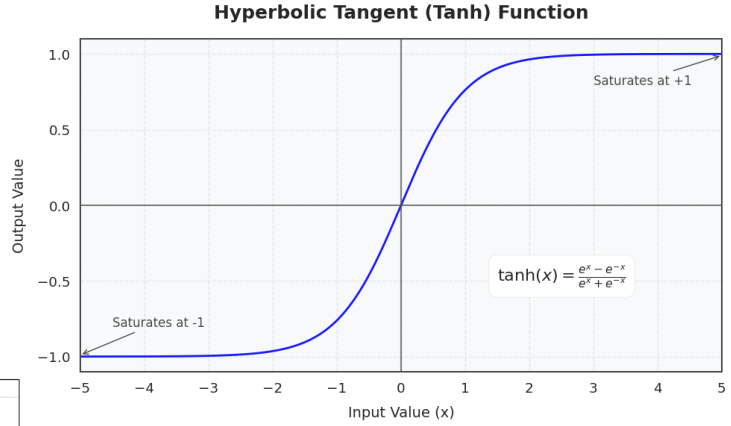
Common Activation Functions

- Sigmoid
  - $f(x) = \frac{1}{1+e^{-x}}$
  - Maps values between 0 and 1, often used in binary classification
- Tanh
  - $f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$
  - Maps values between -1 and 1, often used in hidden layers
- ReLU (Rectified Linear Unit)
  - $f(x) = \max(0, x)$
  - If $x > 0$, $f(x) = x$, if $x \leq 0$, $f(x) = 0$
- Softmax
  - Used in multi-class classification, it outputs probabilities summing up to 1

Activation Functions - Graphs







Math behind Artificial Neurons

- Inputs $x_1, x_2, \ldots, x_n$
- Weights $w_1, w_2, \ldots, w_n$
- Net input function – input matrix * weight matrix
- Threshold function – normalizes the result given by the net input function within a numeric range
- Error – how much is off from the actual result. This will be used to readjust the weights accordingly

Math behind Artificial Neurons

- Learns by adjusting the weights until it can correctly classify the training data
- Takes a lot of time for training
- Has high tolerance to noisy data

Types of Neural Networks

- Perceptron
  - single neuron
  - Single layer perceptron can only learn linearly separable problems
- Multi-layer Perceptron
  - Input layer, one or more hidden layers, output layer
  - Feed-forward NN
    - Data flows only in one direction, without any feedback loops or recurrent connections.
  - Uses back propagation for training
    - Repeatedly adjust the weights to minimize the difference between actual output and the desired output
- Convolutional NN
  - For image processing (2D/3D spatial data)
- Recurrent NN
  - For text/speech processing (sequential data)

Multi-layer Perceptron

- trains on two arrays:
  - array X which holds the training samples represented as floating point feature vectors;
  - array y which holds the target values (class labels) for those training samples
- Can set hidden layer sizes. Example: hidden_layer_sizes = (5, 3)
  - Number of elements corresponds to the number of layers
  - Each number corresponds to the number of neurons in that layer
  - In this example, there are 2 layers, first layer has 5 neurons, and the second one has 3 neurons

https://scikit-learn.org/stable/modules/neural_networks_supervised.html

How to implement?

- Load dataset
- Split into train and test
- Preprocessing, if applicable
- Initialize classifier (model)
  - mlp = MLPClassifier(hidden_layer_sizes=(50,), activation='relu', solver='adam', max_iter=500, random_state=42)
  - Solver
    - used to optimize the network's weights and biases by minimizing the loss function
    - responsible for **updating the parameters of the model** during training to improve performance.
- Fit the model using train set (training)
  - mlp.fit(X_train, y_train)
- Predict for test/new instances (testing)
  - y_pred = mlp.predict(X_test)

References

- https://www.ibm.com/cloud/learn/neural-networks
- https://www.youtube.com/watch?v=ER2It2mIagI&list=PLeo1K3hjS3uu7CxAacxVndI4bE_o3BDtO&index=6&ab_channel=codebasics
- https://www.youtube.com/watch?v=bfmFfD2RIcg&t=223s
- https://python-course.eu/machine-learning/neural-network-digits-dataset.php