Deriving Rules From Data

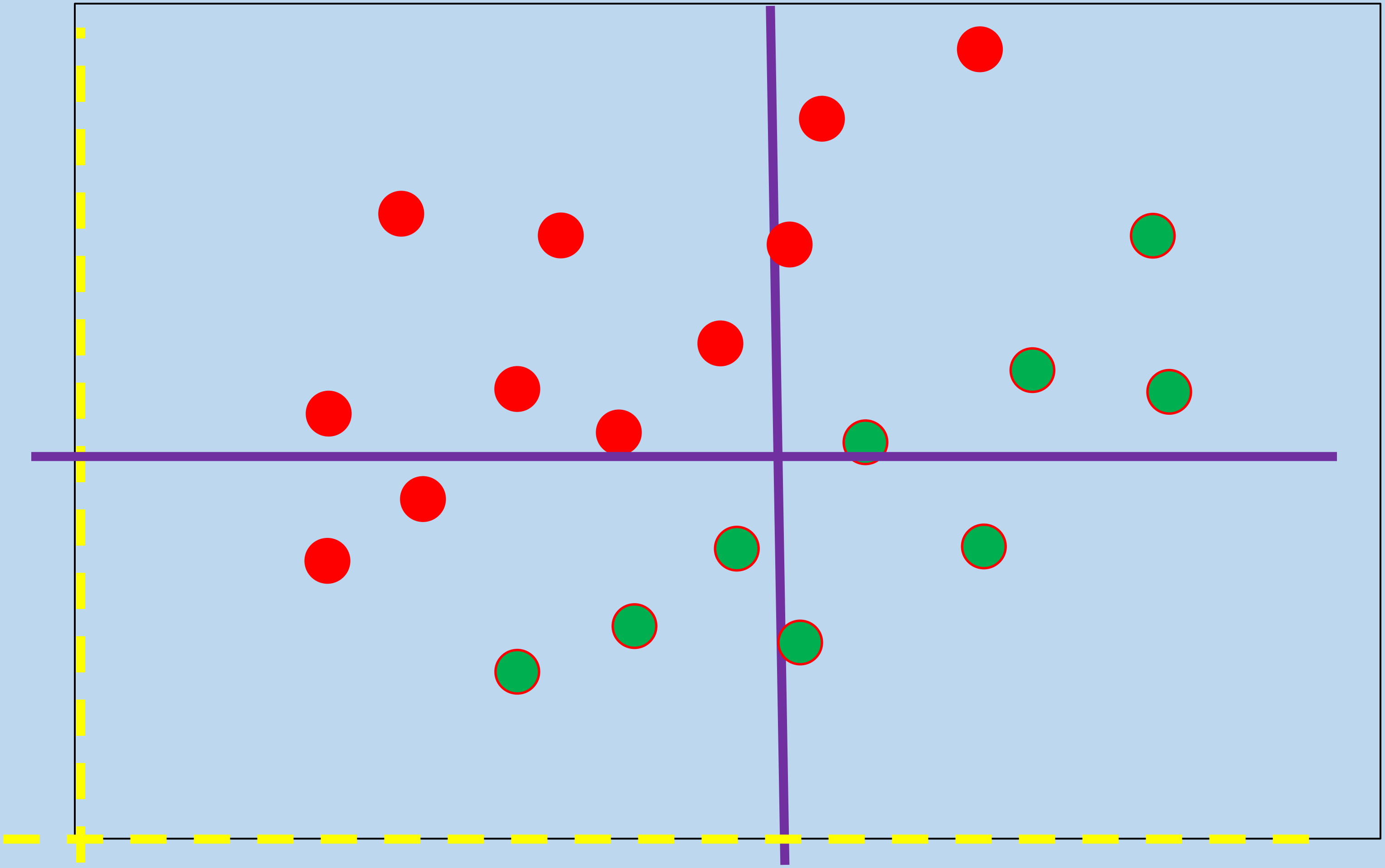
Deriving Rules from Data
Machine Learning Algorithms

Neural Nets

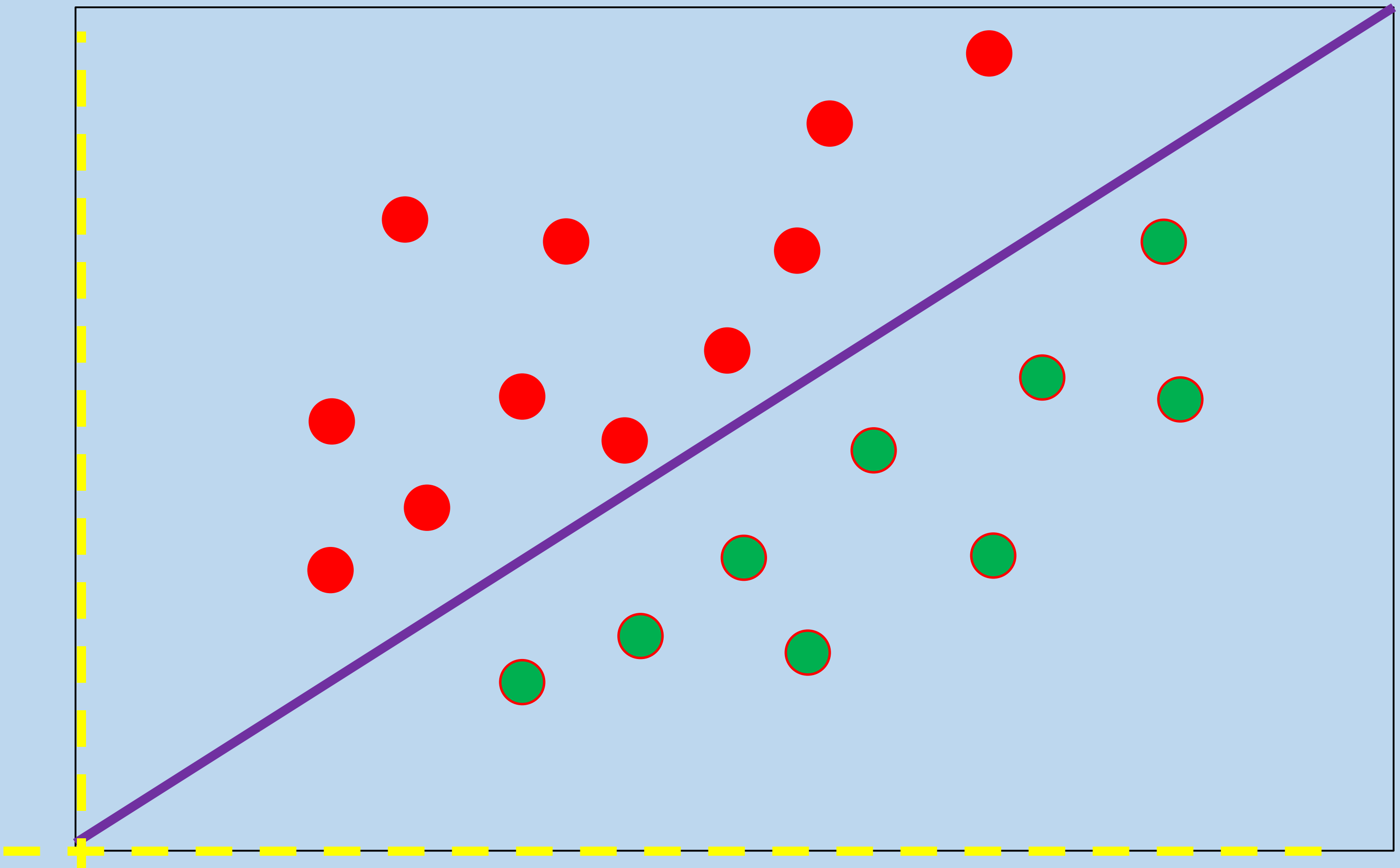
Class starts @ 6:35

Khasha Dehnad

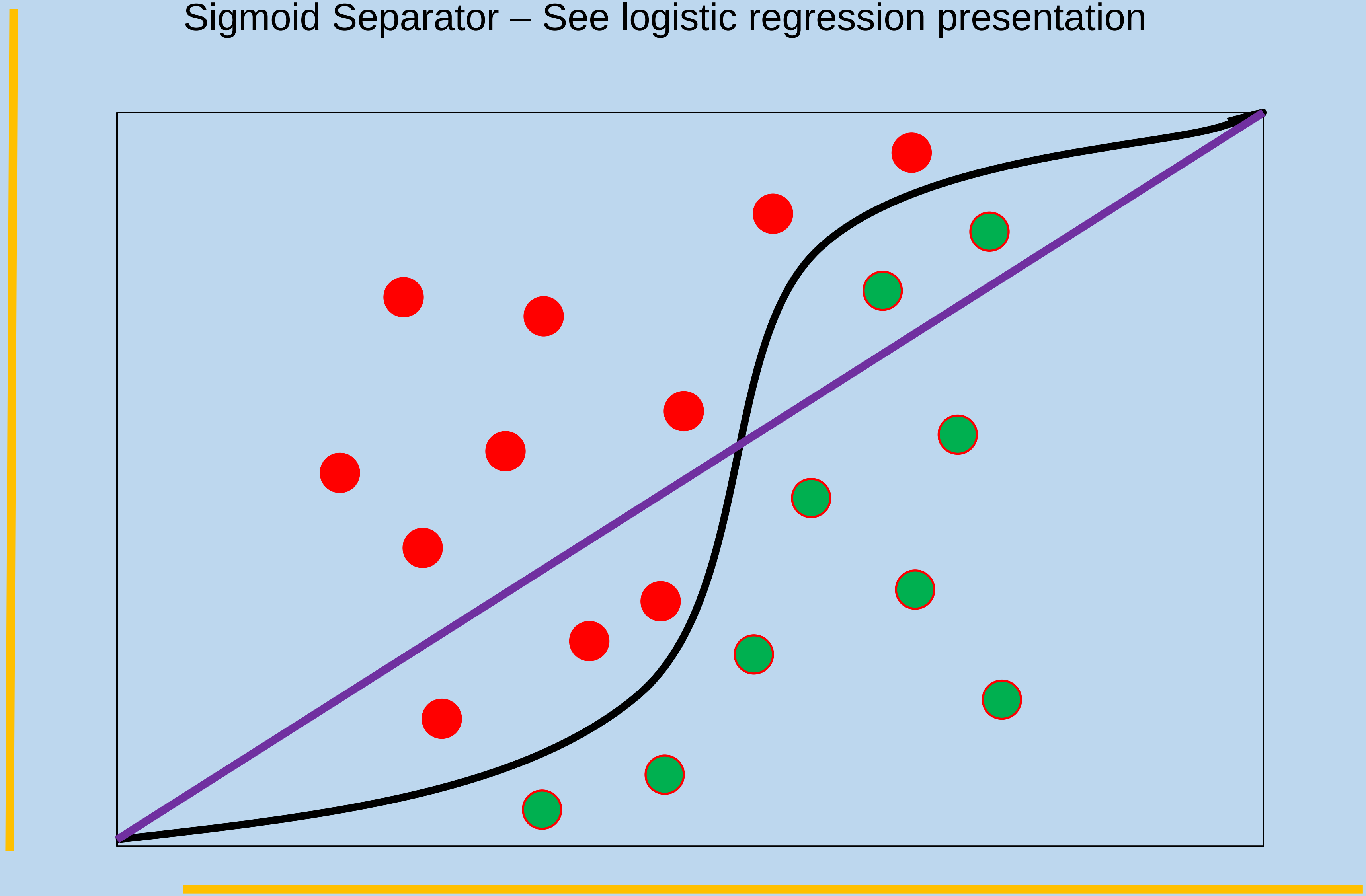
Decision Tree Classifiers



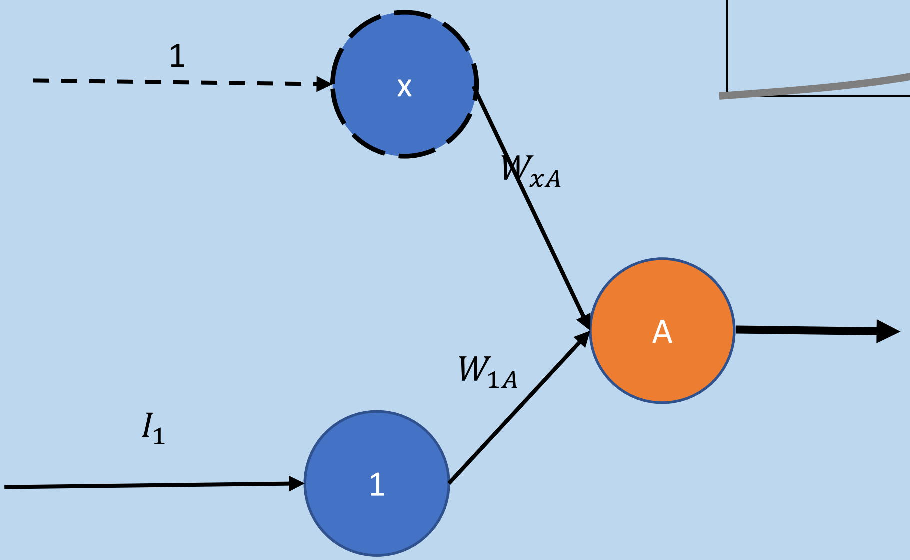
Non-vertical separator (SVM)- See SVM Presentation

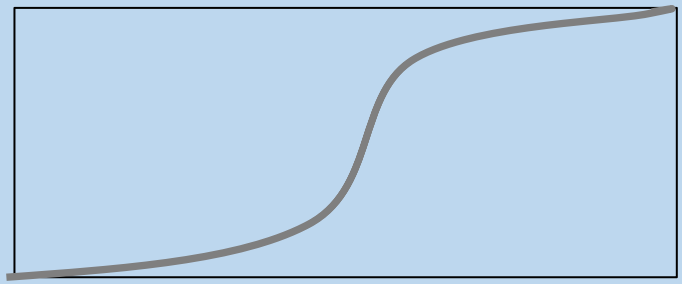


Sigmoid Separator – See logistic regression presentation



Perceptron-



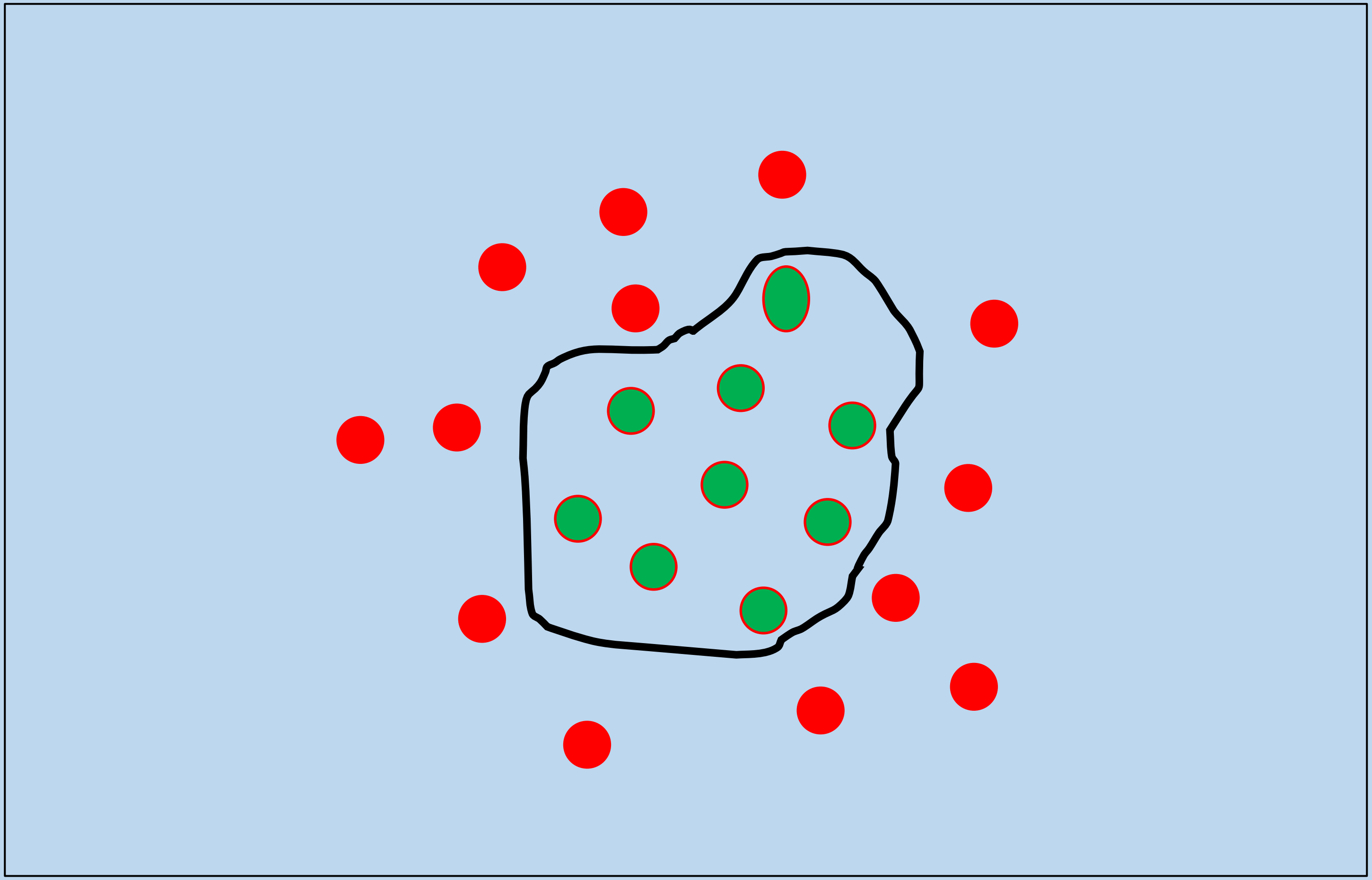


$$\sum_{i=0}^{n} W_{iA}\ I_i \longrightarrow \textbf{Sigmoid()}$$

Sigmoid(W_{xA} + W_{1A} I_1 **)**

$$f(\mathbf{net - z}) = 1/(1 + e^{-x})$$

Sigmoid Separator



Neural Networks

Simulating the Brain to Solve Problems
Artificial Neural Networks (ANN)

Overview

- **Computer emulation of biological neural systems for building models:**

  - **initially theorized in 1943 by McColloch and Pitts of University of Chicago,**
  - **simulates the brain's cognitive learning process,**
  - **"learns" patterns directly from the data,**
  - **searches for complex relationships,**
  - **automatically builds models,**
  - **predicts - compares - adjusts,**
  - **corrects the model's mistakes over and over again,**
  - **input: Data,**
  - **output: Prediction**
  - **tool: the Model "learned" from the Data.**

Neural Networks
Approximate Number of Neurons

Human Brain: 100 Billion (10 ^ 11) Neurons

Fruit fly: 100,000 Neurons

Nematode worm: 302 Neurons

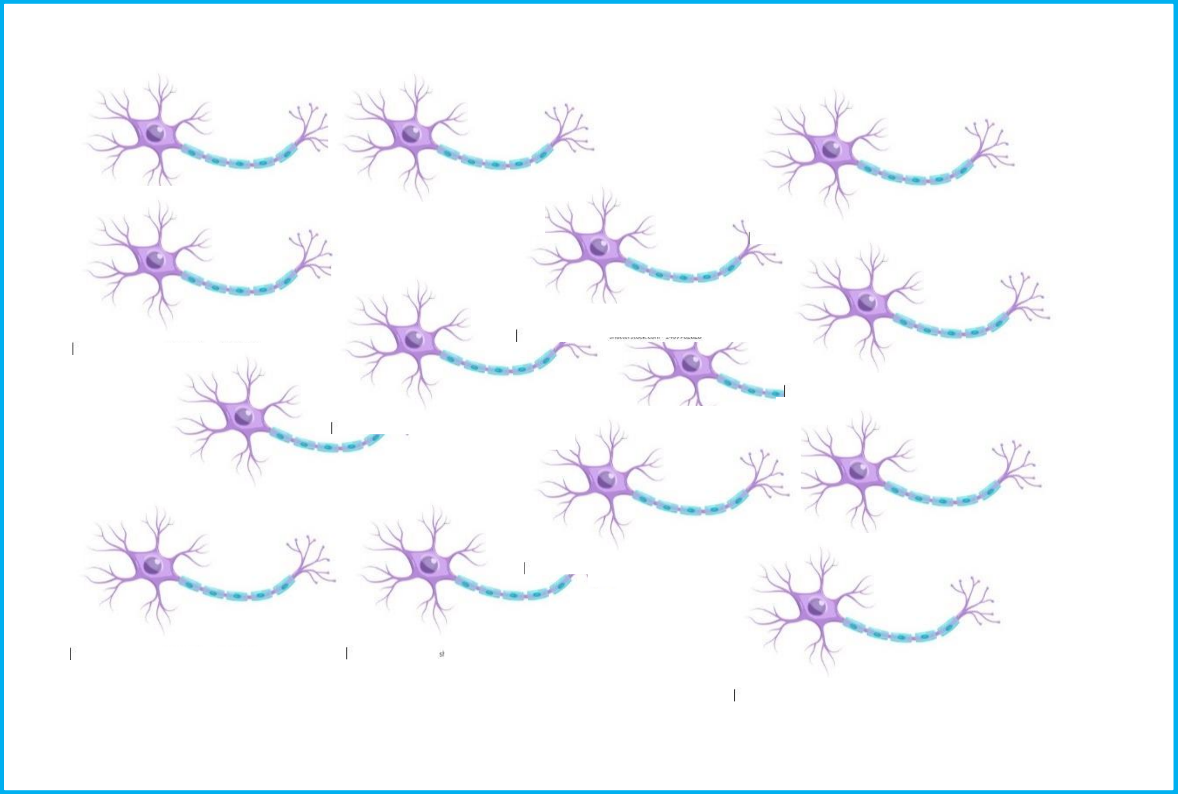
–

Neural Networks
Biological Principles Underlying the Neural Network Technology

The idea of neurons as the structural constituent of the brain was first introduced by Ramon y Cajal (French) 1911

- **Human Brain:**
  - **a network of individual but interconnected nerve cells called *neurons* (10^11 *neurons*)**
  - **neurons are connected to each other via huge number of so-called *synapses* (10^15 synapses or connections),**
  - **a given neuron is connected to 10 thousand other neurons by these synapses,**
  - **neurons can receive information from the outside world at various points in the network,**
  - **these pieces of information are called *stimuli,***
  - **a neuron transfers information on to other neurons by firing chemicals called *neurotransmitters,***
  - **these transfers occur over synapses like bursts of electricity,**
  - **the more important a particular stimulus is, the stronger the burst will be at the synapses,**
  - **the information received by a nerve cell at one of the synapses either *excite* or *inhibit* the cell,**
  - **if the receiving cell is excited, it will pass the information to other neurons,**
  - **if the receiving cell is inhibited, it will damp the impact of the information,**
  - **each nerve cell processes the raw input but passes it on only if it is important,**
  - **the information travels through the network by generating new internal signals,**
  - **the stimuli are processed by brain and nervous system and ultimately a response is produced.**


iStock
by Getty Images
511282379



Neural Networks
Artificial Neural Networks

–a system of neurodes (*nodes*) and *weighted connections* (synapses) inside the memory of a computer,

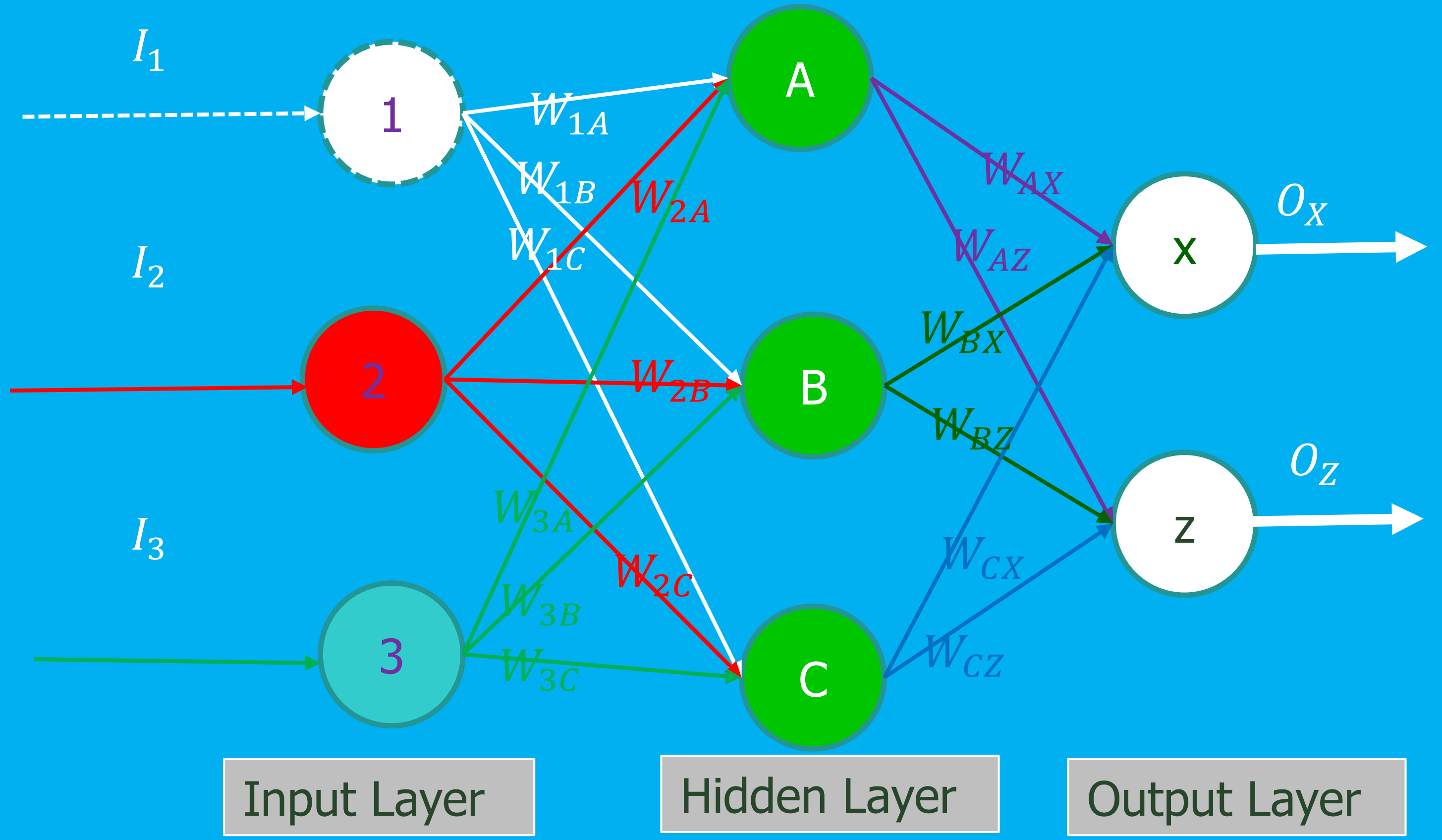
–nodes are data storage locations (like variables in a program, cells in a spreadsheet),

– nodes are arranged in *layers* with weighted connections running between layers,

– *balls* represent nodes and *lines* represent connection weights,

– *input* layer nodes receive the data,

– *output* layer nodes relay the response of the neural network out of the net,



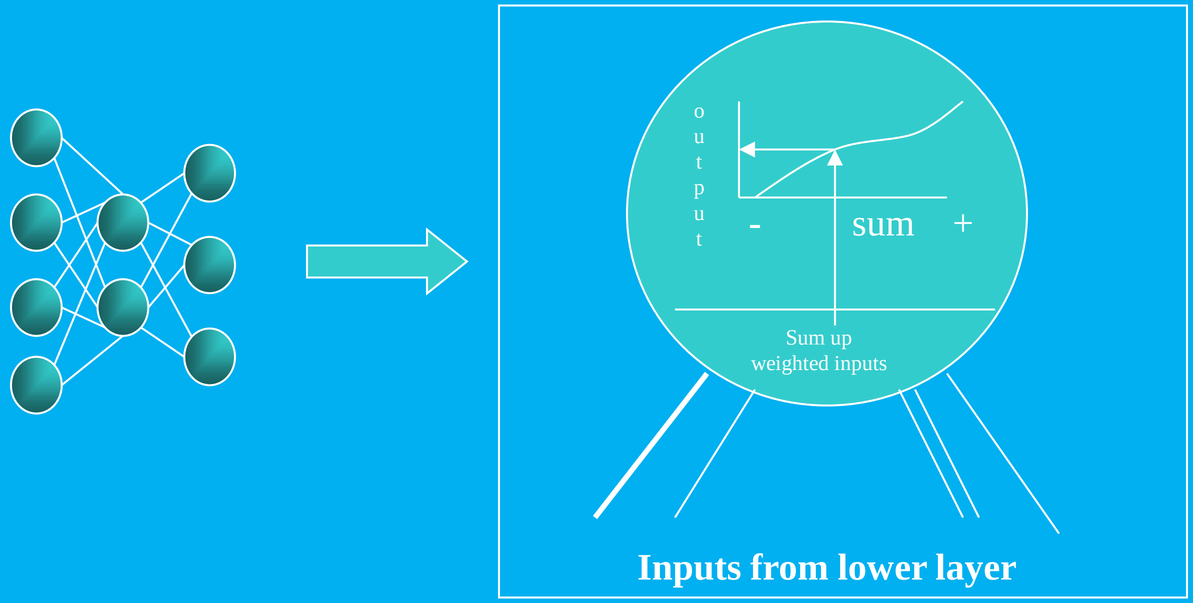
Neural Networks
ANN (Continued)

- *hidden* layer nodes (hidden from the outside world) conduct the internal processing,
- data are fed into the net through the input nodes,
- data are processed internally by hidden nodes, based on the inter-node connection weights,
- result are passed on to the outside world by output nodes,
- "learning" takes place through adjusting connection weights,
- a "learned" neural network has adjusted its weights properly

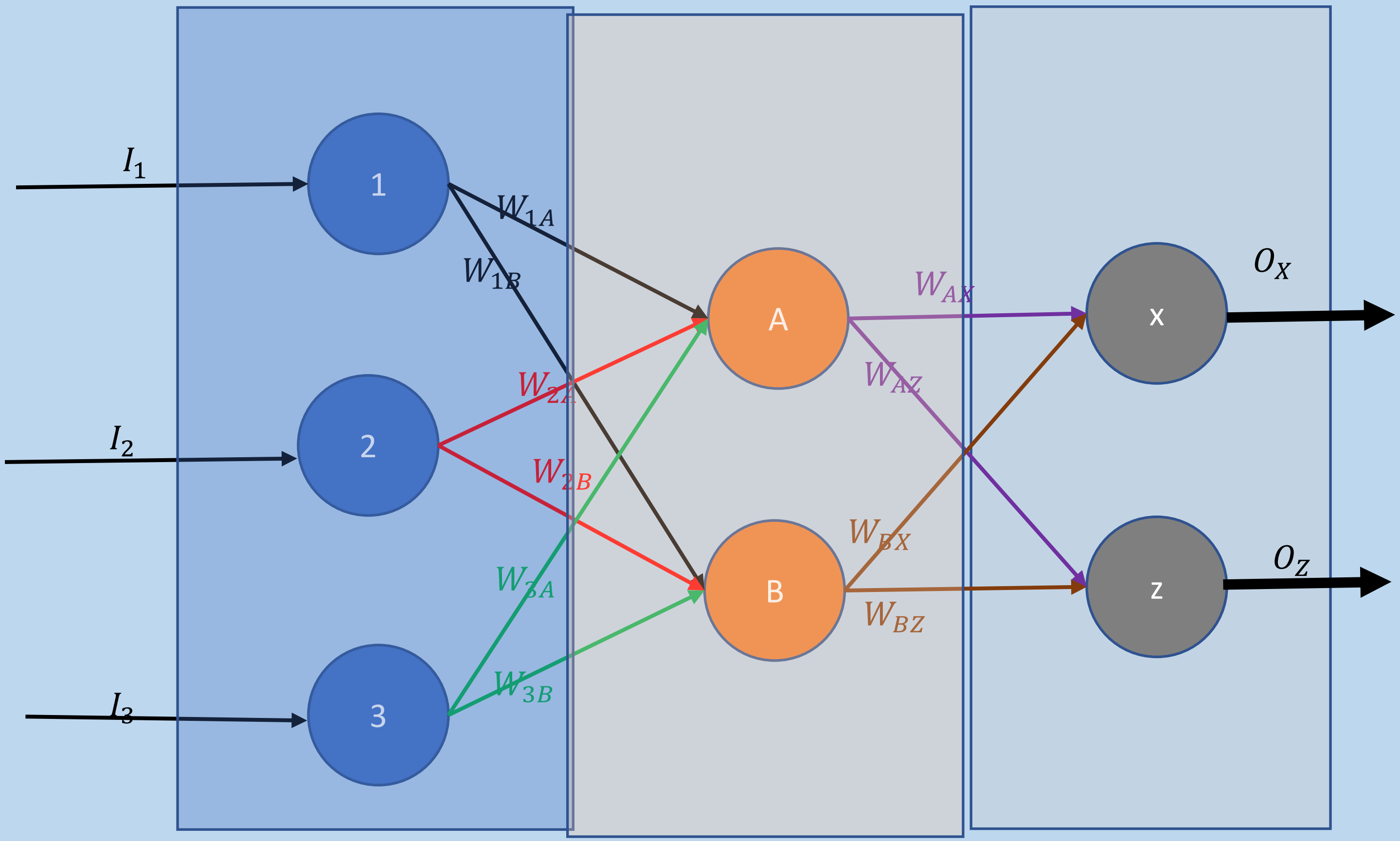
ANN operates in the same way as the biological model on which it is based.

Neural Networks
Application of a Learned Neural Network

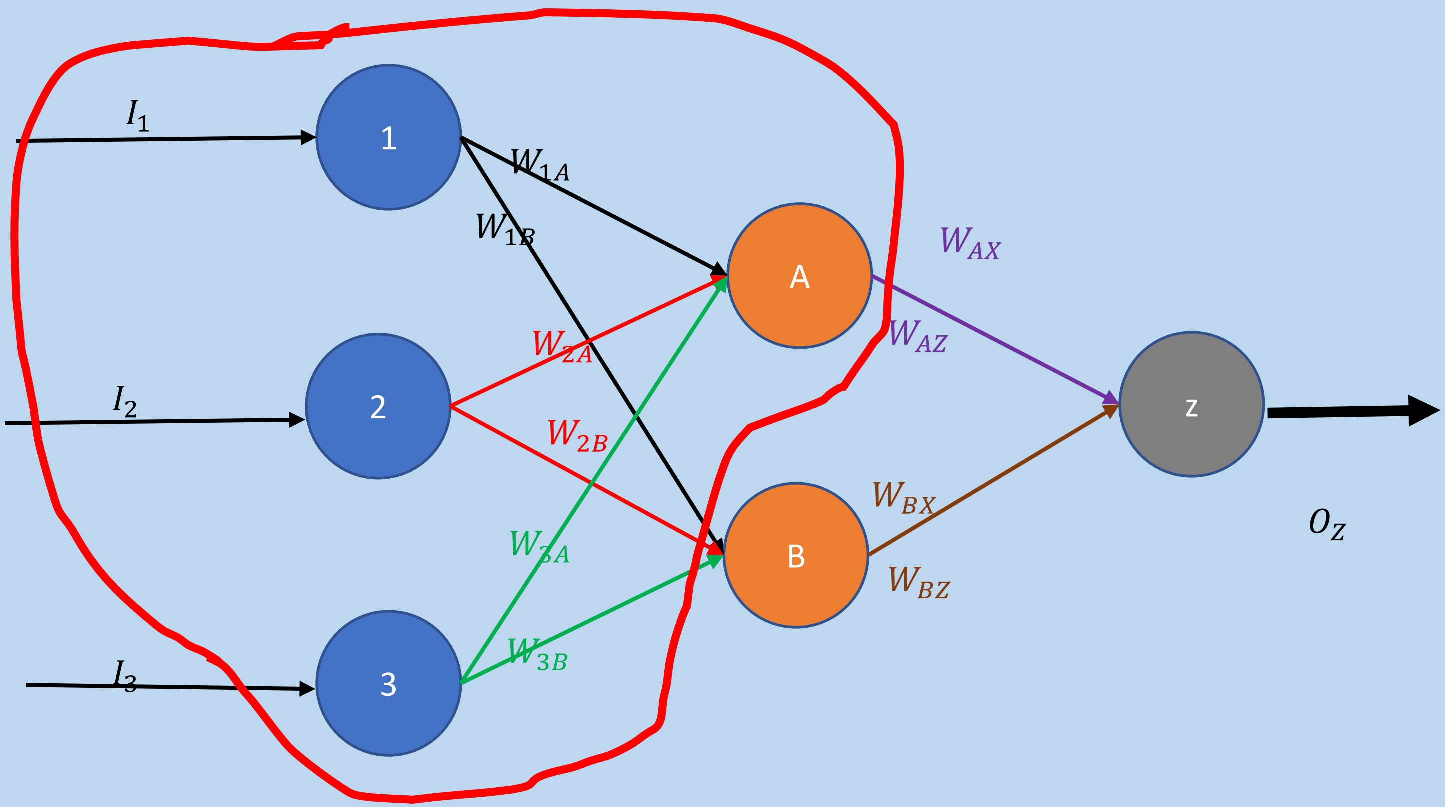
- **Integration Function:**
  - **each neuron receives a set of raw data (input),**
  - **the neuron multiplies each input by the connecting weight leading into it,**
  - **connection weight determines the importance of a given input in contribution to the output of the neuron,**
  - **more important inputs will have bigger weights and less important ones will have smaller weights,**
  - **the *integration function* of the neurode calculates a weighted sum of all inputs.**



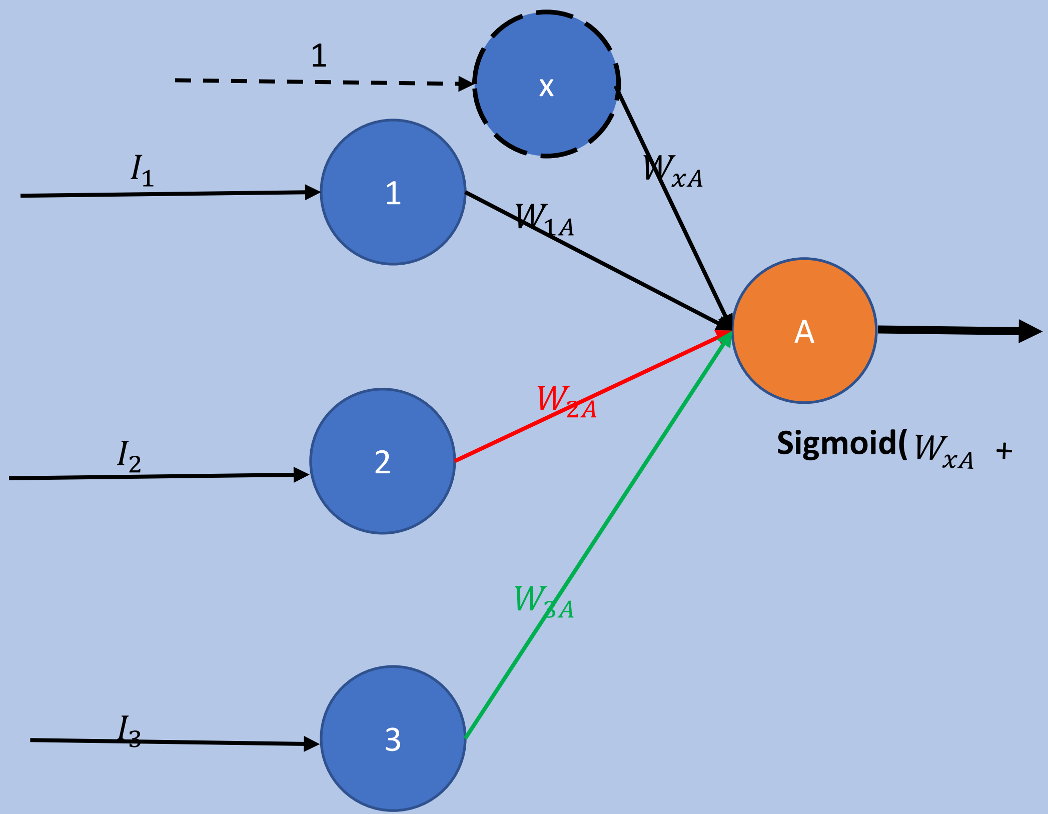
Neural Network with three Layers



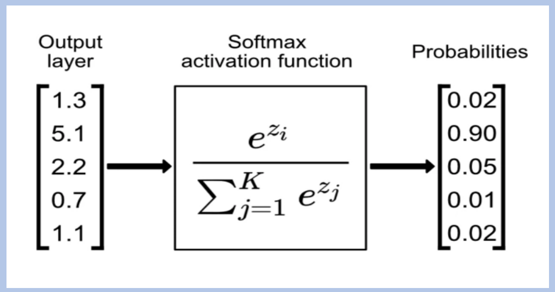
Neural Net with One Node in the Output Layer



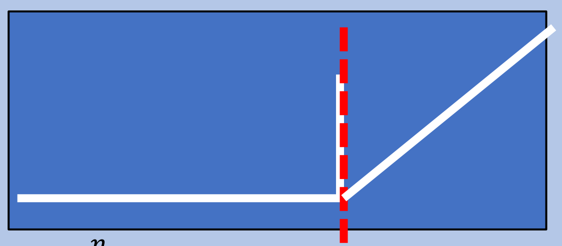
Perceptron-



Softmax

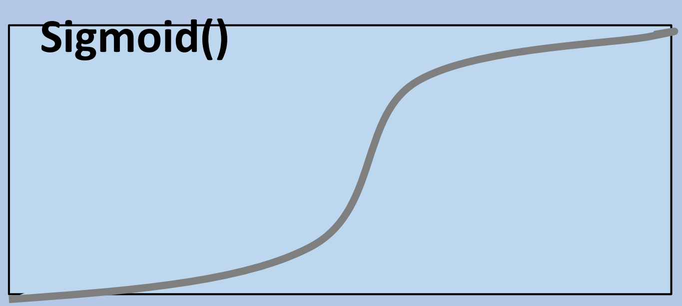


ReLU(x)=max(0,x)



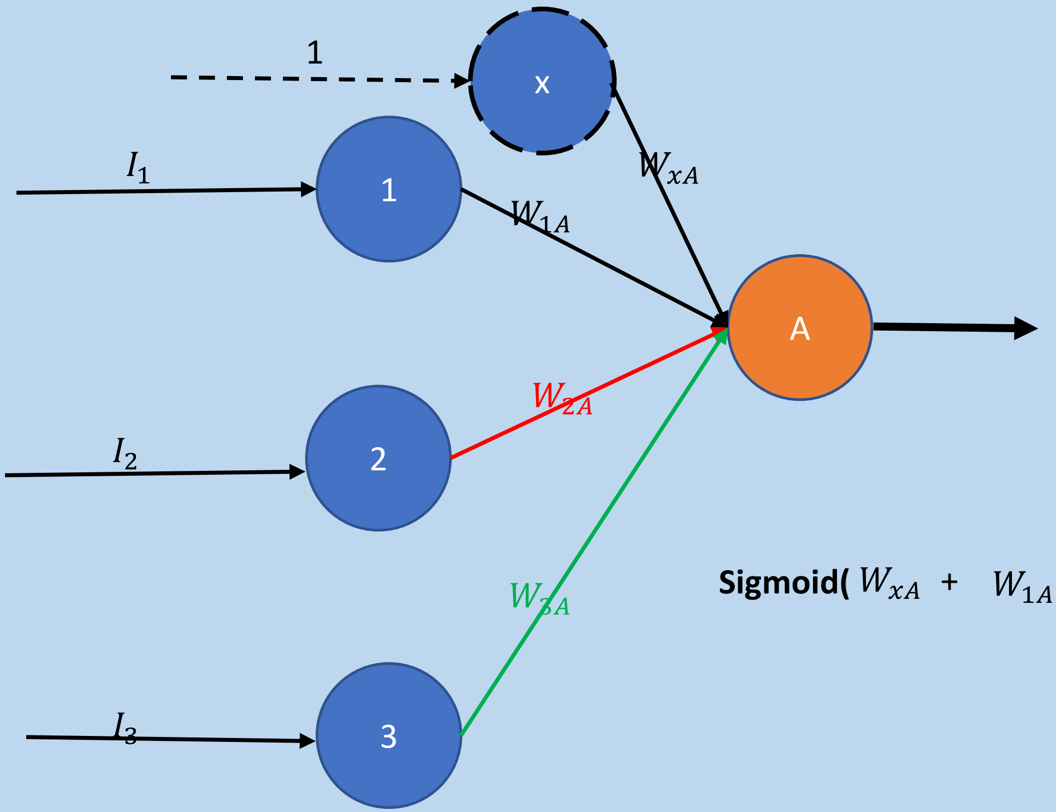
$$\sum_{i=0}^{n} W_{iA}\; I_i \longrightarrow \textbf{Sigmoid()}$$

Sigmoid($W_{xA} + W_{1A}\; I_1 + W_{2A}\; I_2 + W_{3A}\; I_2$ **)**



$$f\,(\mathbf{net - z}) = 1/(1+e^{-x})$$

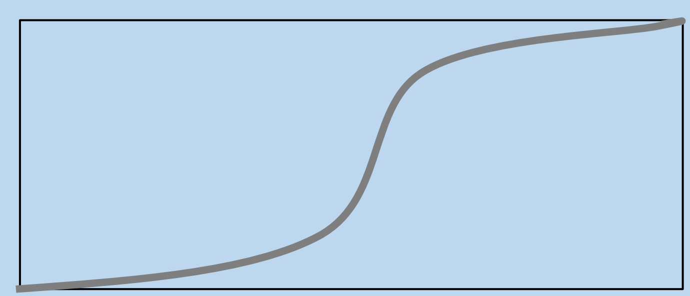
Perceptron-



$$\sum_{i=0}^{n} W_{iA}\ I_i \longrightarrow \textbf{Sigmoid()}$$

$$\textbf{Sigmoid(}\ W_{xA} + W_{1A}\ I_1 + W_{2A}\ I_2 + W_{3A}\ I_2\ \textbf{)}$$

$$f(\mathbf{net\text{-}z}) = 1/(1+e^{-x})$$

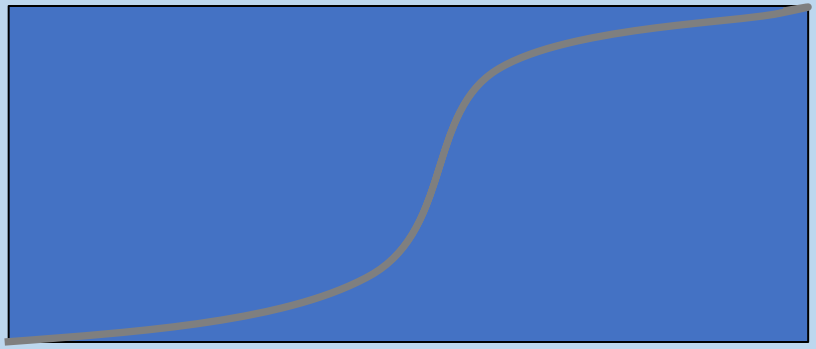


Neural Networks
Application of a Learned Neural Network (Continued)

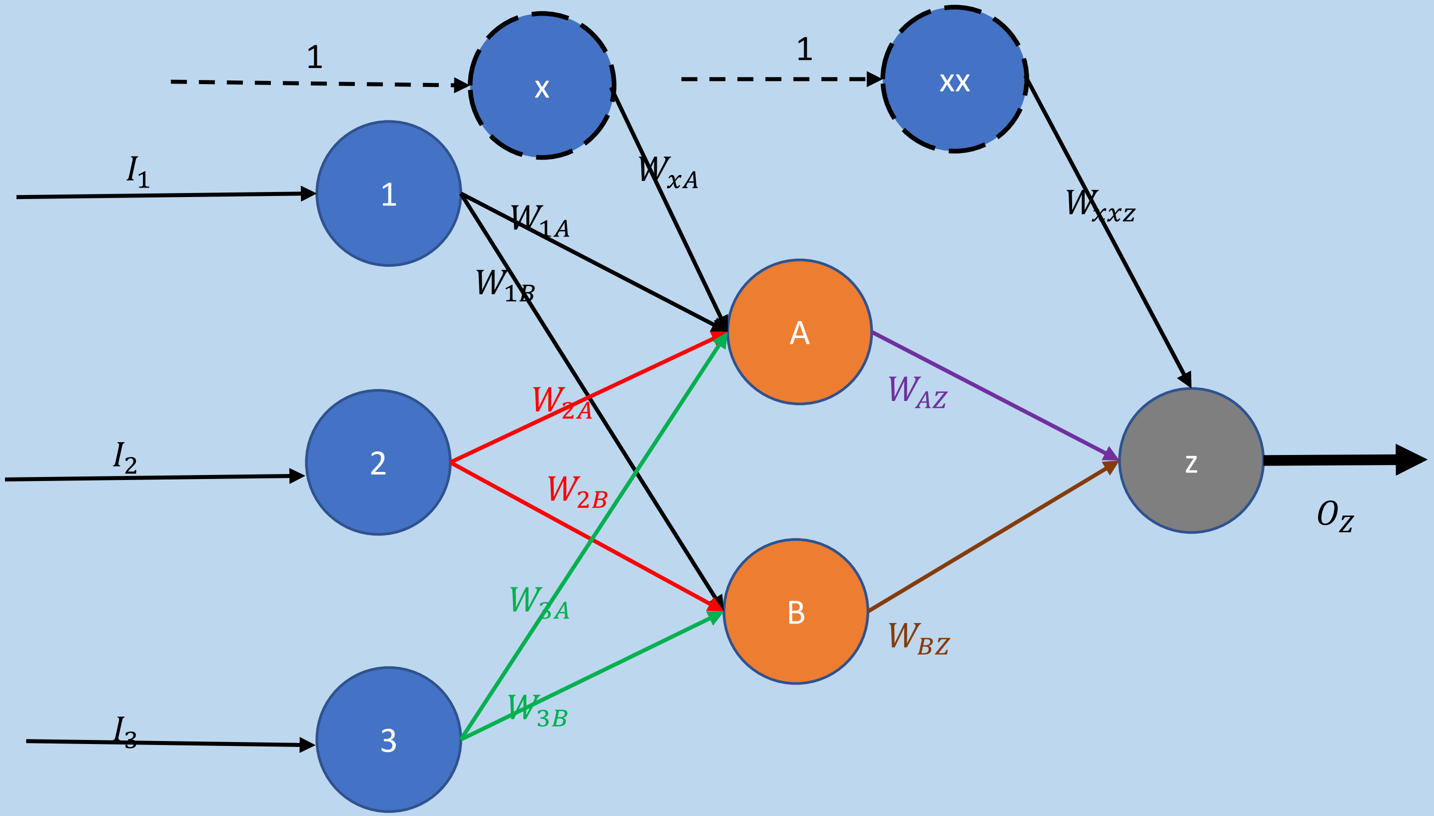
- **Transfer Function:**
  - **the weighted sum is converted into an output value using a mathematical function called *transfer function,***
  - **transfer function normalizes the output into the range of [0,1],**
  - **it serves as a kind of “dimmer” switch for turning the neuron “on” and “off ”,**
  - **the transfer function’s value will be *high* (excited) when the sum of the inputs is large & positive; and *low* (inhibited) when the sum is large negative,**
  - **the transfer function determines the degree at which a given sum will cause a neurode to fire.**

$$\boldsymbol{\sigma}' = \boldsymbol{\sigma} \quad (1 - \boldsymbol{\sigma})$$

$$f(\mathbf{net - z}) = 1/\left(1 + e^{-x}\right)$$



Neural Net with Dummy nodes



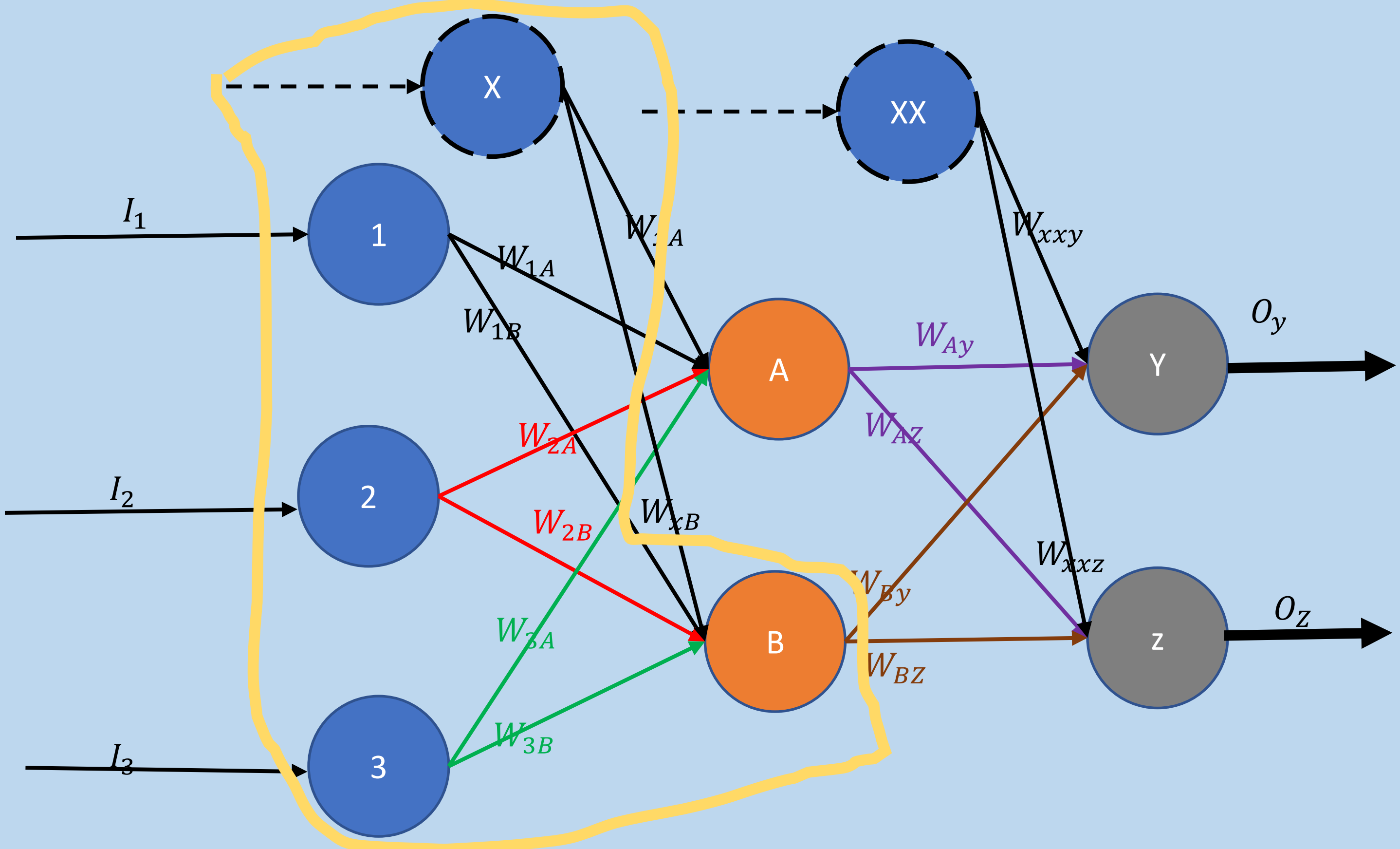
Matrix Representation of the Input-Hidden Layer

$$\textbf{Sigmoid}\left(\begin{bmatrix} 1 & I_1 & I_2 & I_3 \end{bmatrix} \begin{pmatrix} W_{xA} & W_{xB} \\ W_{1A} & W_{1B} \\ W_{2A} & W_{2B} \\ W_{3A} & W_{3B} \end{pmatrix} \right)$$

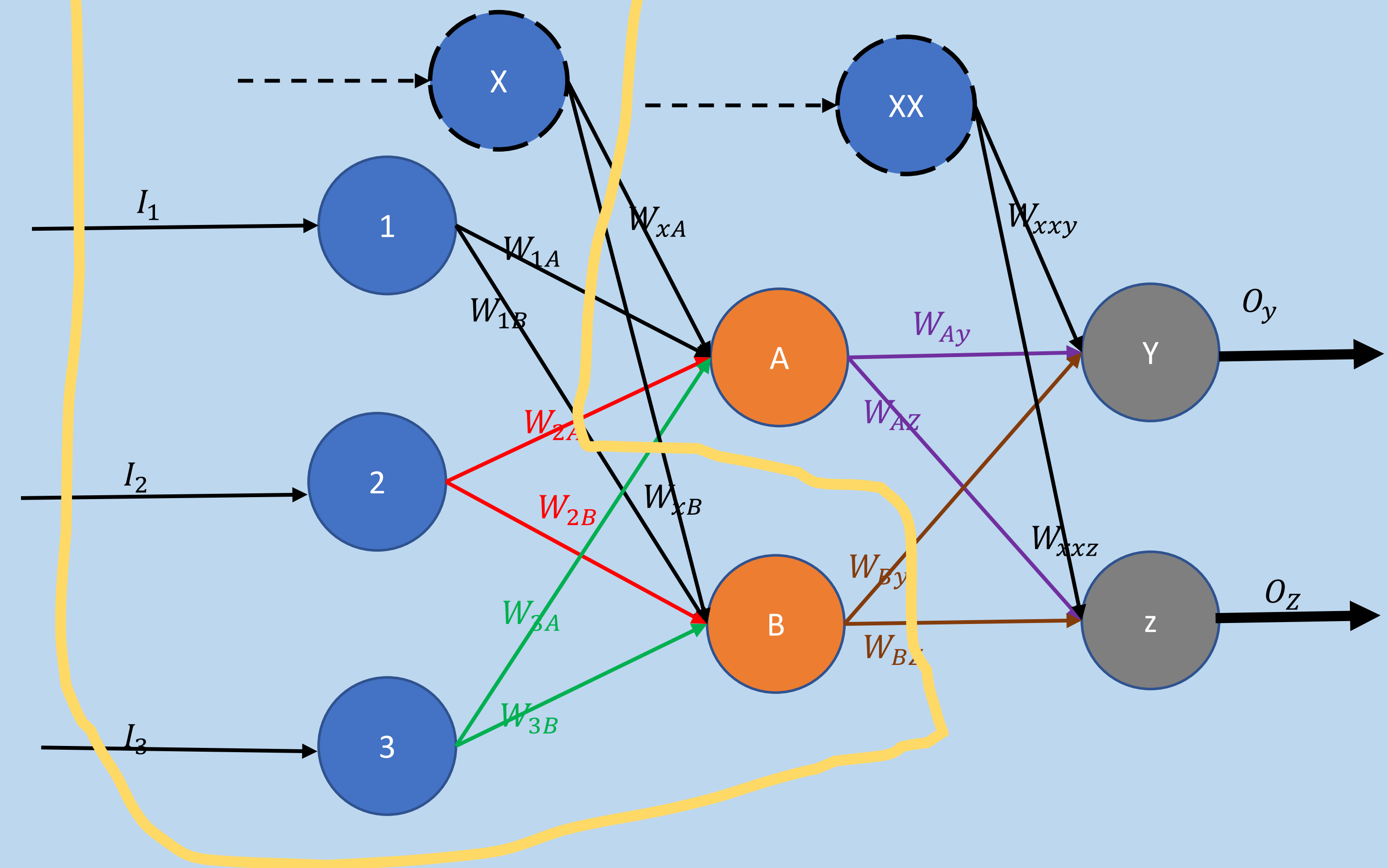
Matrix Representation of the Hidden-Output Layer

$$\textbf{Sigmoid}\left(\begin{bmatrix} 1 & O_A & O_B \end{bmatrix} \begin{bmatrix} W_{xxz} \\ W_{Az} \\ W_{BZ} \end{bmatrix} \right)$$

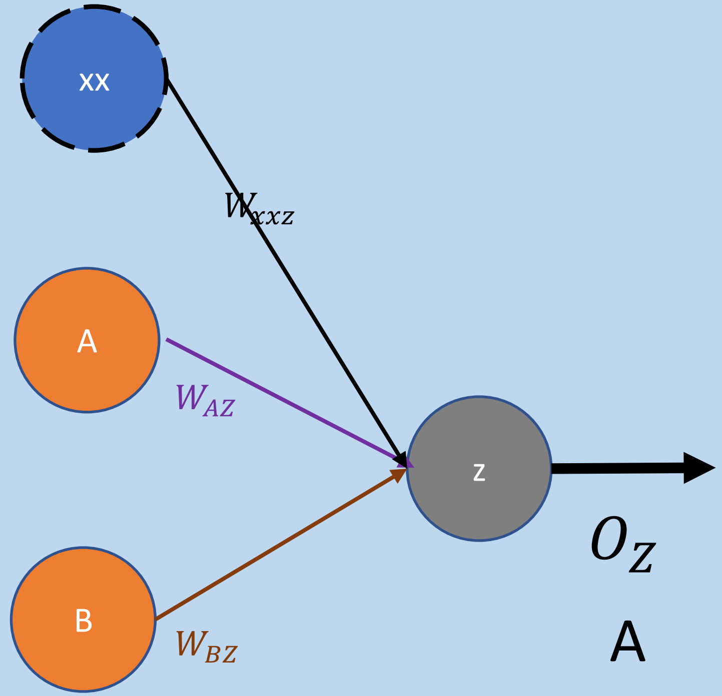
Neural Network with three Layers



Neural Network with three Layers



Neural Net with Dummy nodes



$$e^{2} = (A - O_Z)^2$$

Neural Net- Weight Adjustments

$$\frac{\partial e^2}{\partial W_{AZ}} = \frac{\partial e^2}{\partial O_Z} * \frac{\partial O_Z}{\partial \Sigma} * \frac{\partial \Sigma}{\partial W_{AZ}}$$

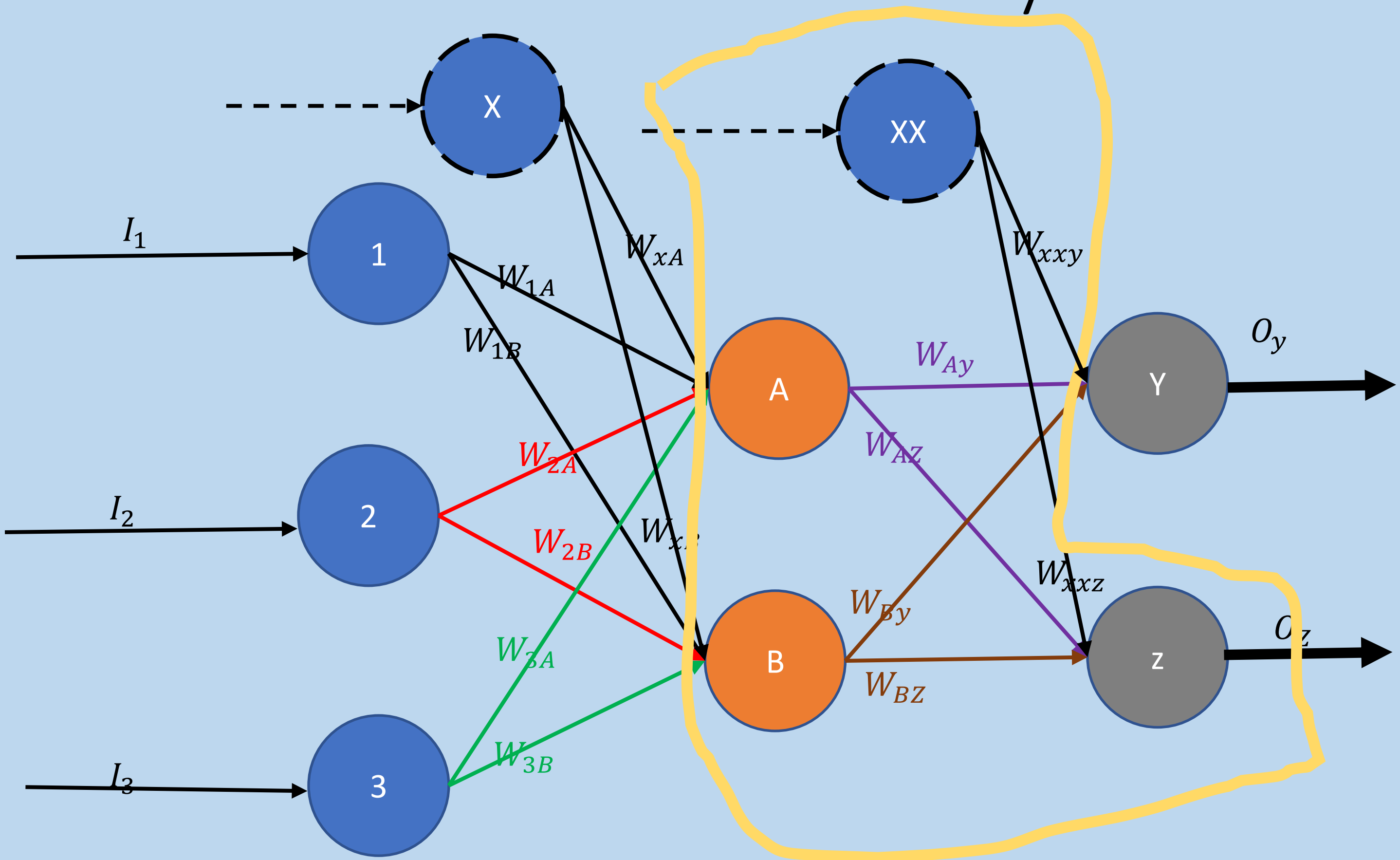
$$\frac{\partial (A - O_Z)^2}{\partial O_Z} = -2(A - O_Z) = -2e$$

$$\frac{\partial O_Z}{\partial \Sigma} = \frac{\partial \, \textbf{Sigmoid}(W_{xxZ} + W_{AZ} \; O_A + W_{BA} \; O_{BZ})}{\partial (W_{xxZ} + W_{AZ} \; O_A + W_{BA} \; O_{BZ})} = O_Z * (1 - O_Z)$$

$$\frac{\partial \Sigma}{\partial W_{AZ}} = \frac{\partial (W_{xxZ} + W_{AZ} \; O_A + W_{BA} \; O_{BZ})}{\partial W_{AZ}} = O_A$$

$$\Delta = \boxed{\frac{\partial e^2}{\partial W_{AZ}} = - \; e * O_Z * (1 - O_Z) * O_A} * -\eta$$

Neural Network with three Layers



$$= \frac{W_{AZ}}{W_{xxZ} \quad W_{AZ} \quad W_{BZ}} \delta_Z + \frac{W_{Ay}}{W_{xxy} \quad W_{Ay} \quad W_{By}} \delta_y$$

$$e_A = \frac{W_{AZ}}{W_{xxZ} + W_{AZ} + W_{BZ}} \delta_Z + \frac{W_{Ay}}{W_{xxy} + W_{Ay} + W_{By}} \delta_y$$

$$e_B = \frac{W_{BZ}}{W_{xxZ} + W_{AZ} + W_{BZ}} \delta_Z + \frac{W_{By}}{W_{xxy} + W_{Ay} + W_{By}} \delta_y$$

$$\begin{bmatrix} e_A \\ e_B \end{bmatrix} = \begin{pmatrix} W_{AZ} & W_{Ay} \\ W_{BZ} & W_{By} \end{pmatrix} * \begin{pmatrix} \delta_Z \\ \delta_y \end{pmatrix} = \begin{pmatrix} W_{AZ}\ \delta_Z + W_{Ay}\ \delta_y \\ W_{BZ}\ \delta_Z + W_{By}\ \delta_y \end{pmatrix}$$

Matrix Representation of the Hidden-Output Layer

$$W_{ij} New = W_{ij} Current + \Delta w_{ij}$$

$$\Delta W_{ij} = \eta \delta_j X_{ij}$$

$$\delta_j =$$

$$output_j (1 - output_j)(actual_j - output_j)$$ Output Nodes

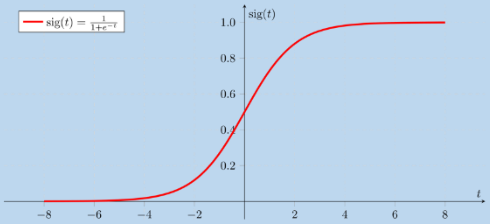
$$output_j (1 - output_j) \sum W_{jk} \delta_j$$ Hidden Nodes

$$\frac{\partial e^2}{\partial W_{AZ}} = \overbrace{- \; e * O_Z * (1 - O_Z)}^{\delta_j} * \overbrace{O_A}^{X_{ij}} * -\eta$$

$$W_{JK} \; \delta_j \;\; => \begin{pmatrix} W_{AZ} \; \delta_Z + W_{AX} \; \delta_y \\ W_{BZ} \; \delta_Z + W_{BX} \; \delta_y \end{pmatrix}$$

Appendix

Sigmoid function : First derivative



$$\sigma(x) = \frac{1}{1+e^{-x}} \quad \ldots(1)$$

$$= \frac{d}{dx}\frac{1}{1+e^{-x}} = \frac{d}{dx}(1+e^{-x})^{-1}$$

$$= \frac{d}{dx}(1+e^{-x})^{-1} = -(1+e^{-x})^{-2}.\frac{d}{dx}(1+e^{-x})$$

$$= -(1+e^{-x})^{-2}.(\frac{d}{dx}[1] + \frac{d}{dx}[e^{-x}]) = -(1+e^{-x})^{-2}.(0 + \frac{d}{dx}[e^{-x}])$$

$$= -(1+e^{-x})^{-2}.(e^{-x}.\frac{d}{dx}[-x]) = -(1+e^{-x})^{-2}.(e^{-x}.-\frac{d}{dx}[x])$$

$$= -(1+e^{-x})^{-2}.(e^{-x}.-1) = (1+e^{-x})^{-2}.e^{-x}$$

Source: https://towardsdatascience.com/derivative-of-the-sigmoid-function

First derivative of sigmoid function

$$= -(1+e^{-x})^{-2}.(e^{-x}.-1) = (1+e^{-x})^{-2}.e^{-x}$$

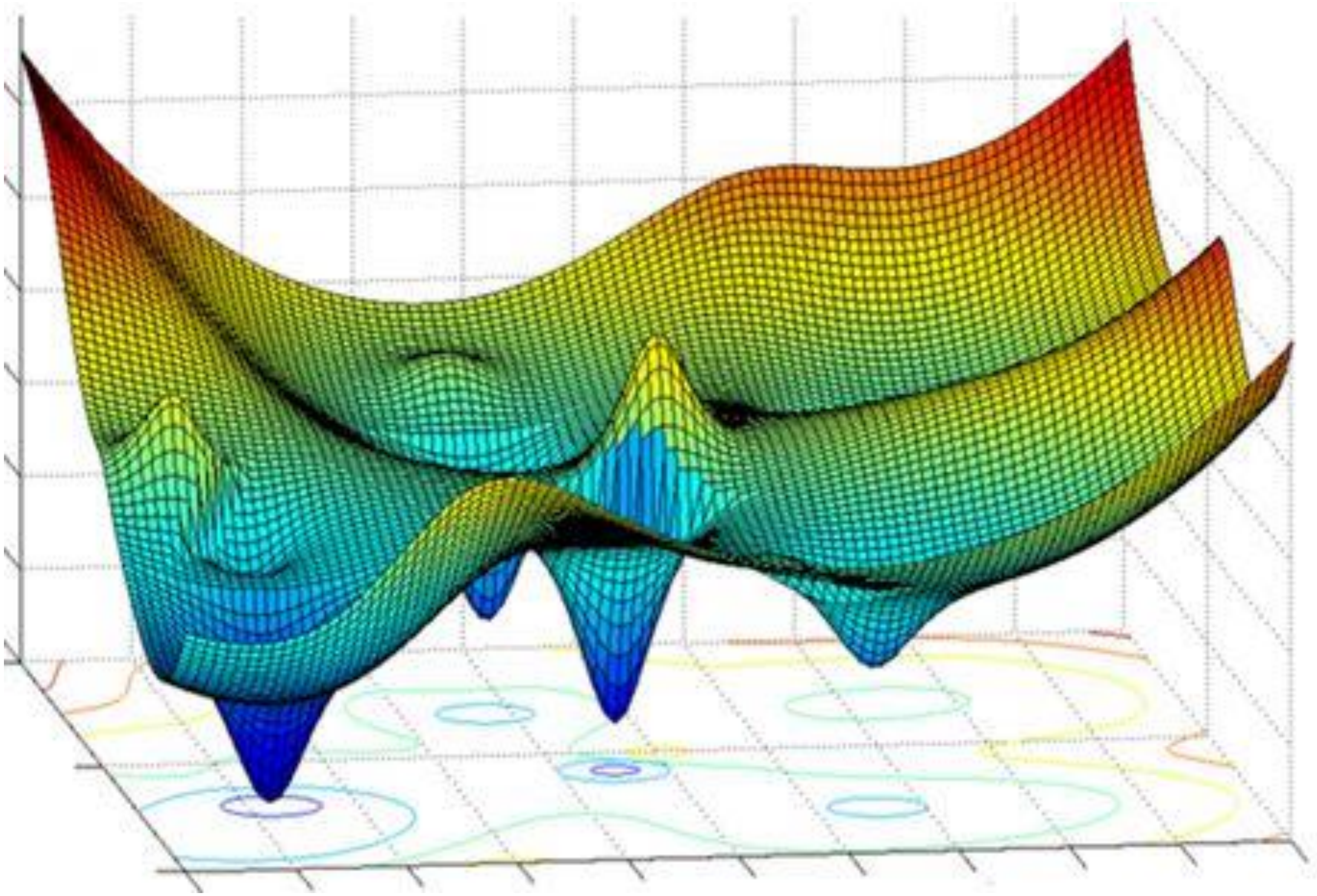
$$= \frac{1.e^{-x}}{(1+e^{-x}).(1+e^{-x})} = \frac{1}{(1+e^{-x})}.\frac{e^{-x}}{(1+e^{-x})}$$

$$= \frac{1}{(1+e^{-x})}.\frac{e^{-x}+1-1}{(1+e^{-x})} = \frac{1}{(1+e^{-x})}.(\frac{1+e^{-x}}{1+e^{-x}} - \frac{1}{1+e^{-x}})$$

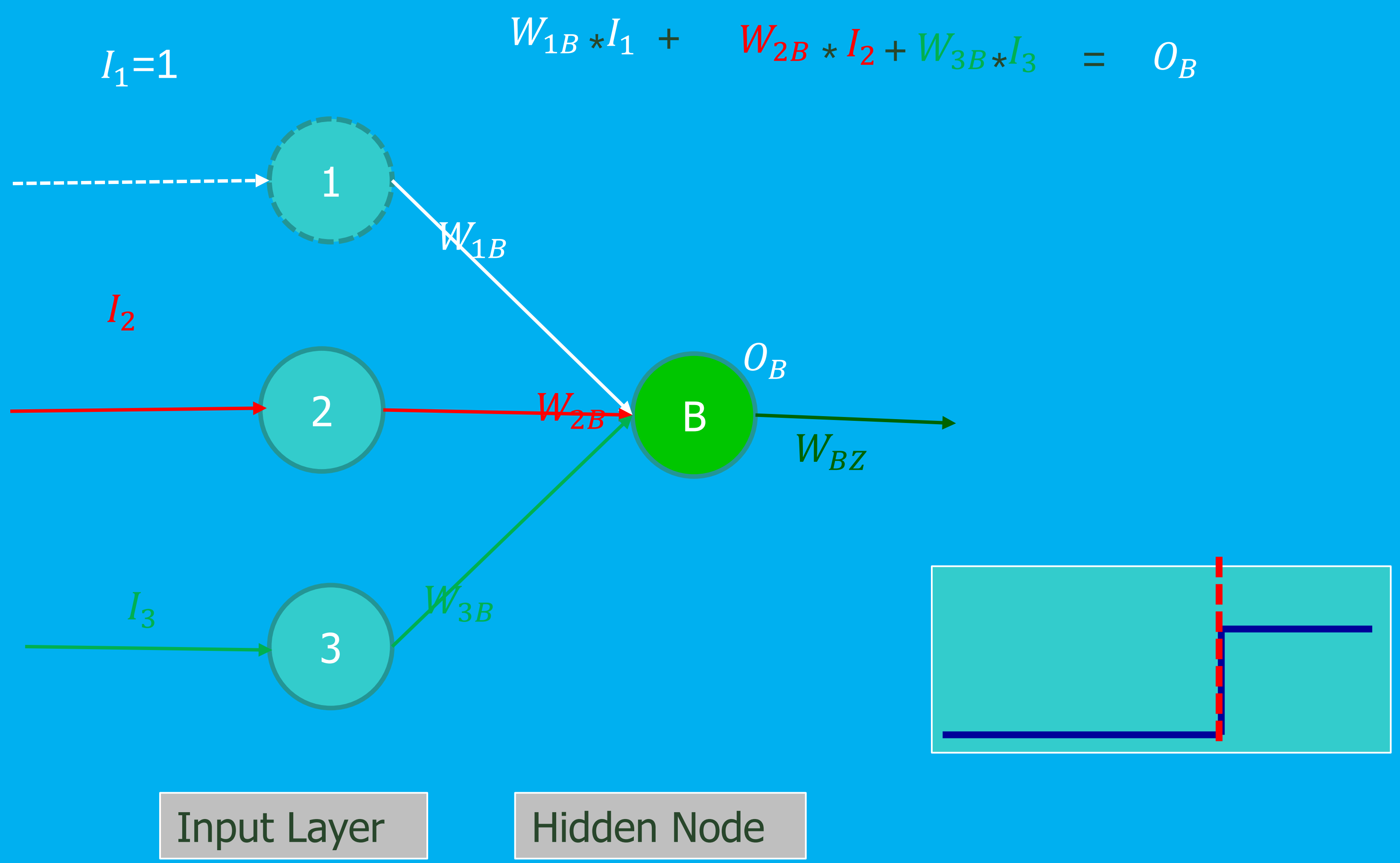
$$= \frac{1}{(1+e^{-x})}.(1 - \frac{1}{1+e^{-x}}) = \sigma(x).(1-\sigma(x))$$

Source: https://towardsdatascience.com/derivative-of-the-sigmoid-function

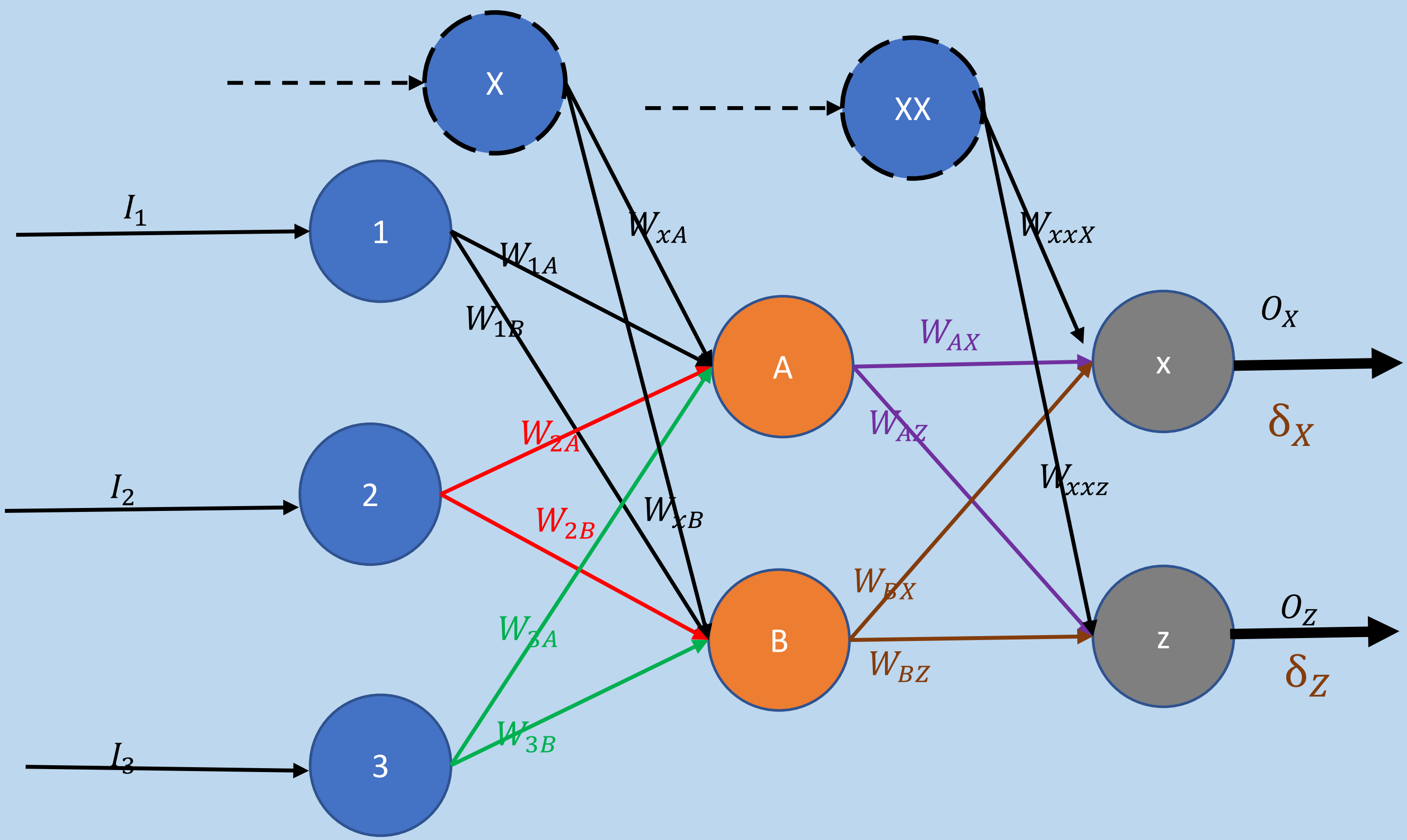
Gradient Descent



https://www.fromthegenesis.com/gradient-descent-part1/


$W_{1B} * I_1 + W_{2B} * I_2 + W_{3B} * I_3 = O_B$
I_1=1
1
W_{1B}
I_2
2
W_{2B}
O_B
B
W_{BZ}
I_3
3
W_{3B}
Input Layer
Hidden Node

Neural Network with three Layers



$$e_A = \frac{W_{AZ}}{W_{xxZ} + W_{AZ} + W_{BZ}}\,\delta_Z + \frac{W_{AX}}{W_{xxX} + W_{AX} + W_{BX}}\,\delta_X$$