Introduction to Machine Learning

Outline

- Why Machine Learning (ML)?
- What is ML?
- Applications
- Neural networks

Why Machine Learning?

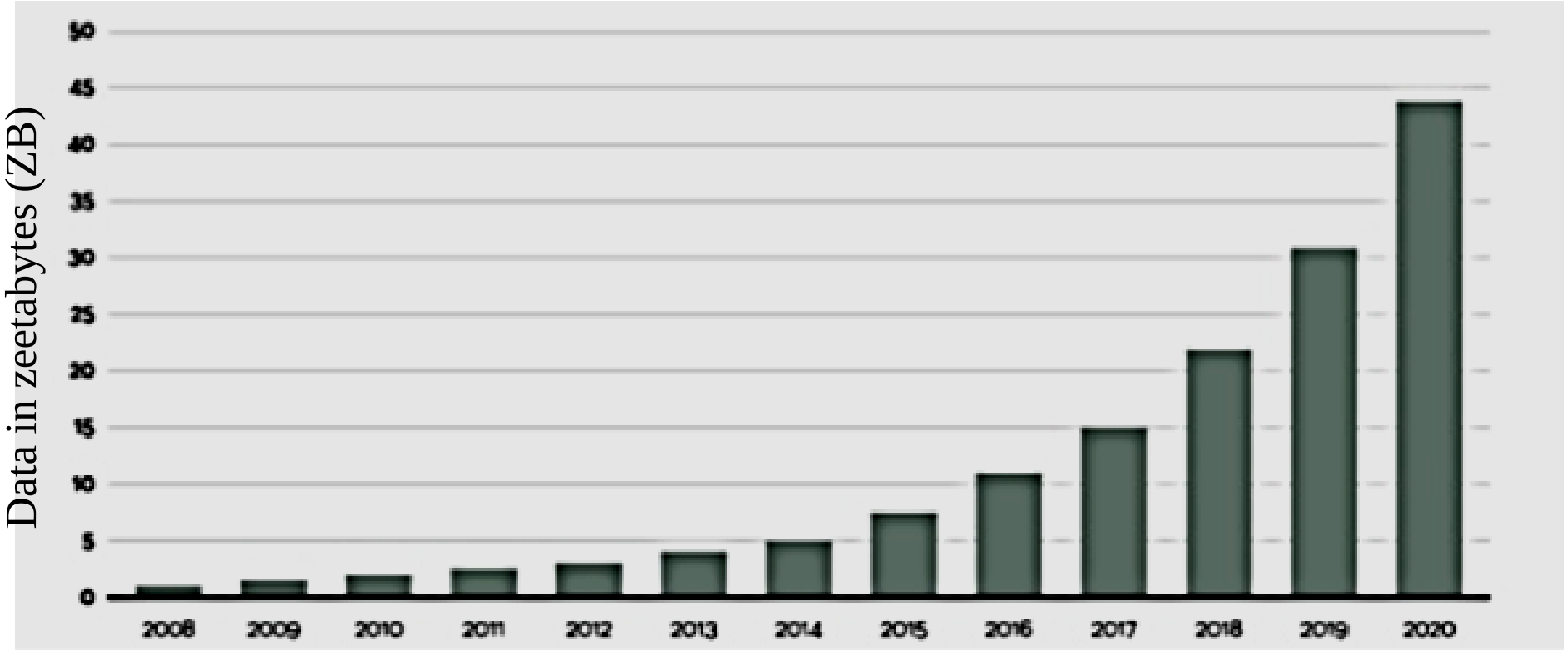


Figure. Growth of data [source: https://www.slideshare.net/machinepulse/machine-learning-and-realworld-applications]

Why Machine Learning? (contd…)

- Human expertise does not exist (navigating on Mars),
- Humans are unable to explain their expertise (speech recognition)
- Solution changes in time (routing on a computer network)
- Solution needs to be adapted to particular cases (user biometrics)

What is Machine Learning?

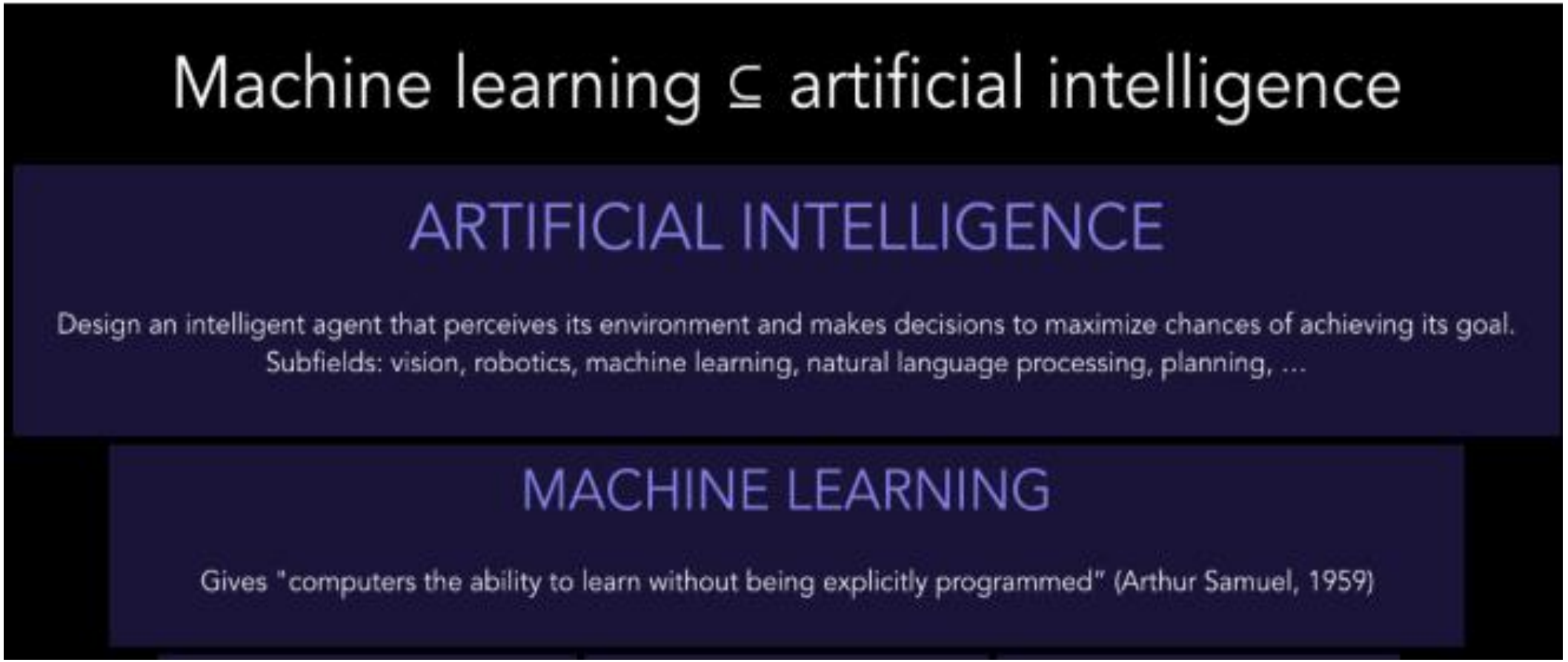
MACHINE + LEARNING

- Machine Learning is **programming computers**
  - Computers learn from data to obtain insight and
  - Help in knowledge discovery.

What is Machine Learning? (contd…)

- Tom M. Michell defines:
- A machine learns
  - With respect to a particular task T,
  - Performance metric P, and
  - Type of experience E,
  - if the **system reliably improves its performance P** at task T, following experience E.

Artificial intelligence is the study of agents that perceive the world around them, form plans, and make decisions to achieve their goals.



Machine learning is a subfield of artificial intelligence

What is Machine Learning? (contd…)

- **Example: Spam Detection**
  - Task T: recognizing and classifying an e-mail as good or spam.
  - Performance measure P: percent of e-mails **correctly classified**
  - Training experience E: a database of emails wherein each email is labeled (by spam or not spam) by users

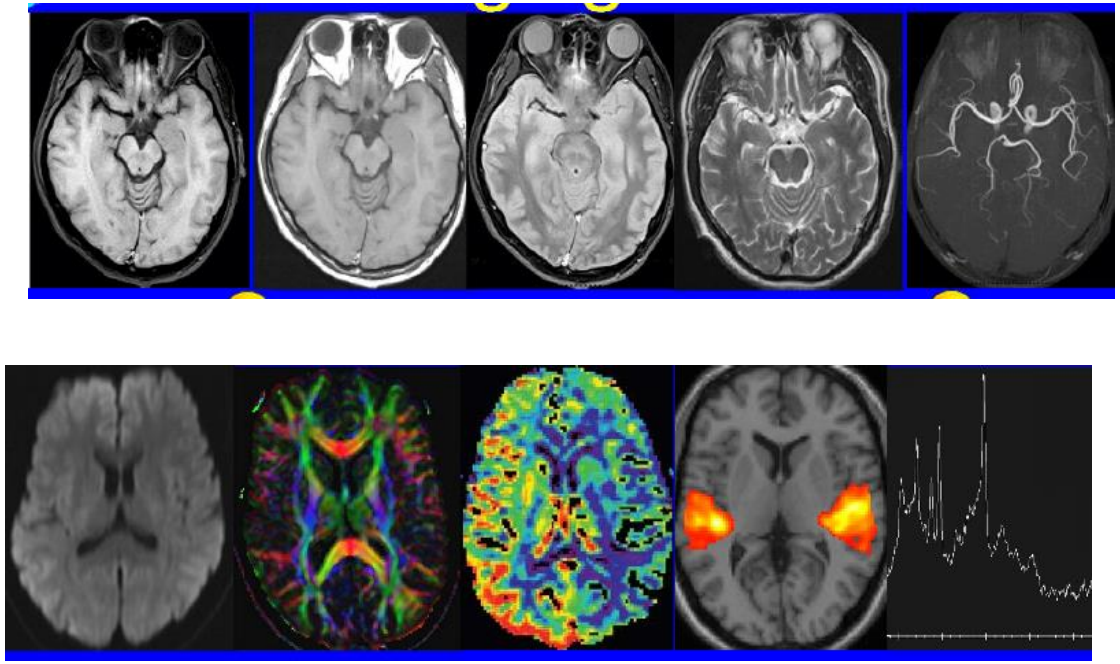
Applications

Spam Detection

Weather Forecasting

Source: Google Images

Applications (contd…)



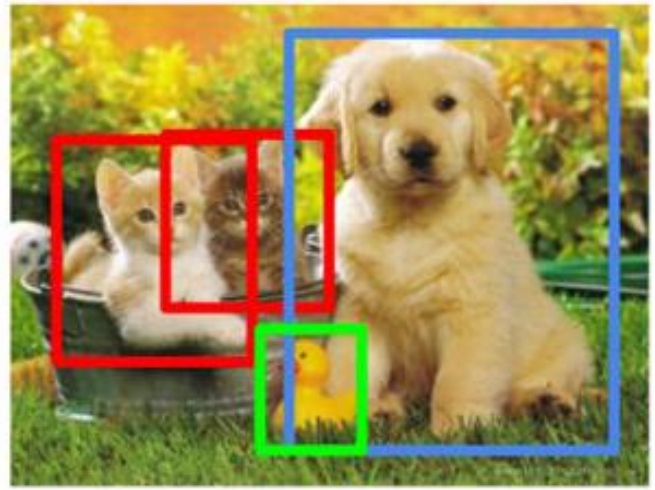
Medical Diagnosis

Text Summarization

Source: Google Images

Applications (contd…)

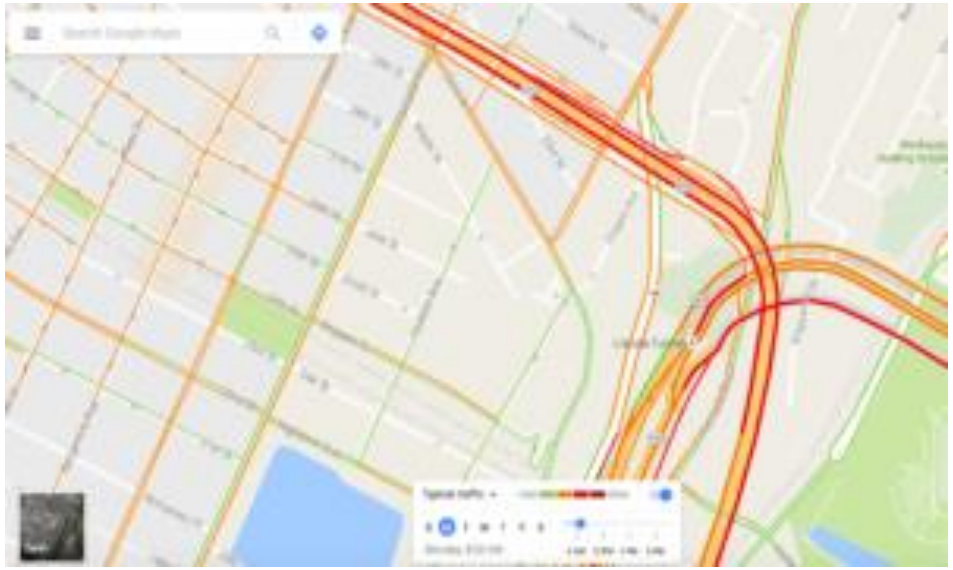
Object Classification



Object Detection

Source: Google Images

Applications (contd…)



Traffic Detection

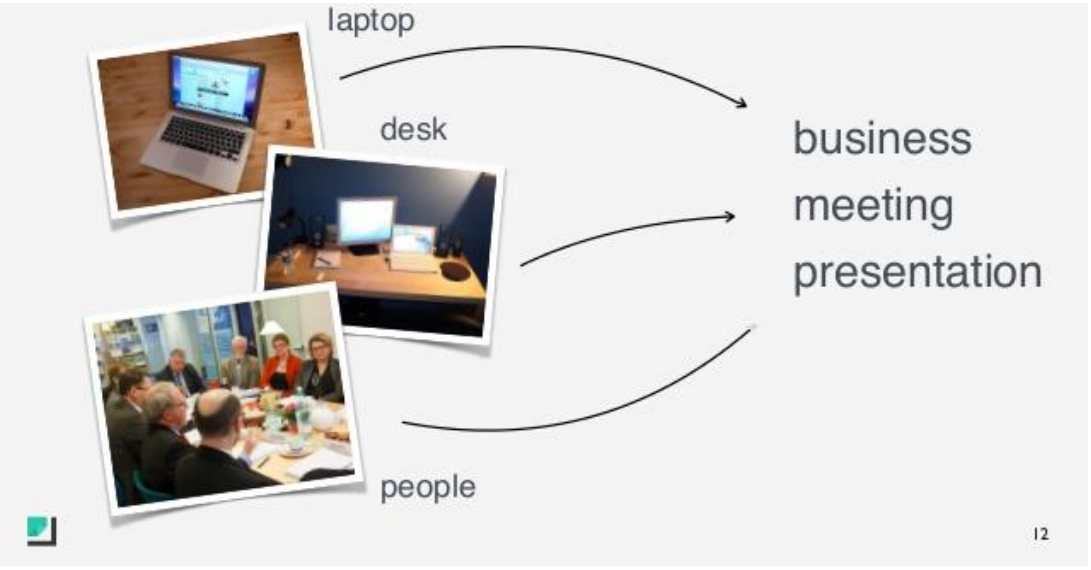


Photo Tagging

Source: Google Images

Applications (contd…)

Speech Recognition/ Speech-to-text

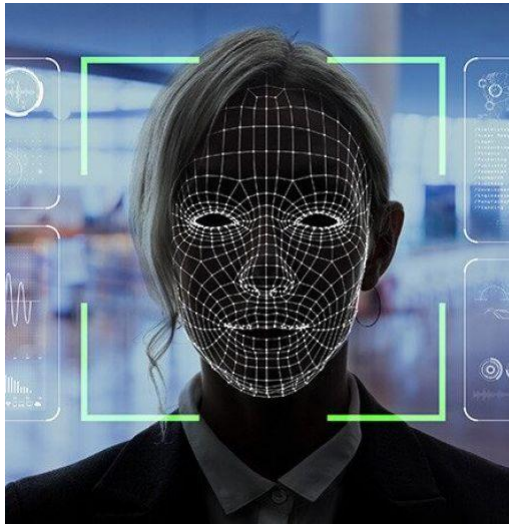
Online Fraud Detection

Source: Google Images

Applications (contd…)

Fingerprint Detection

Face Recognition



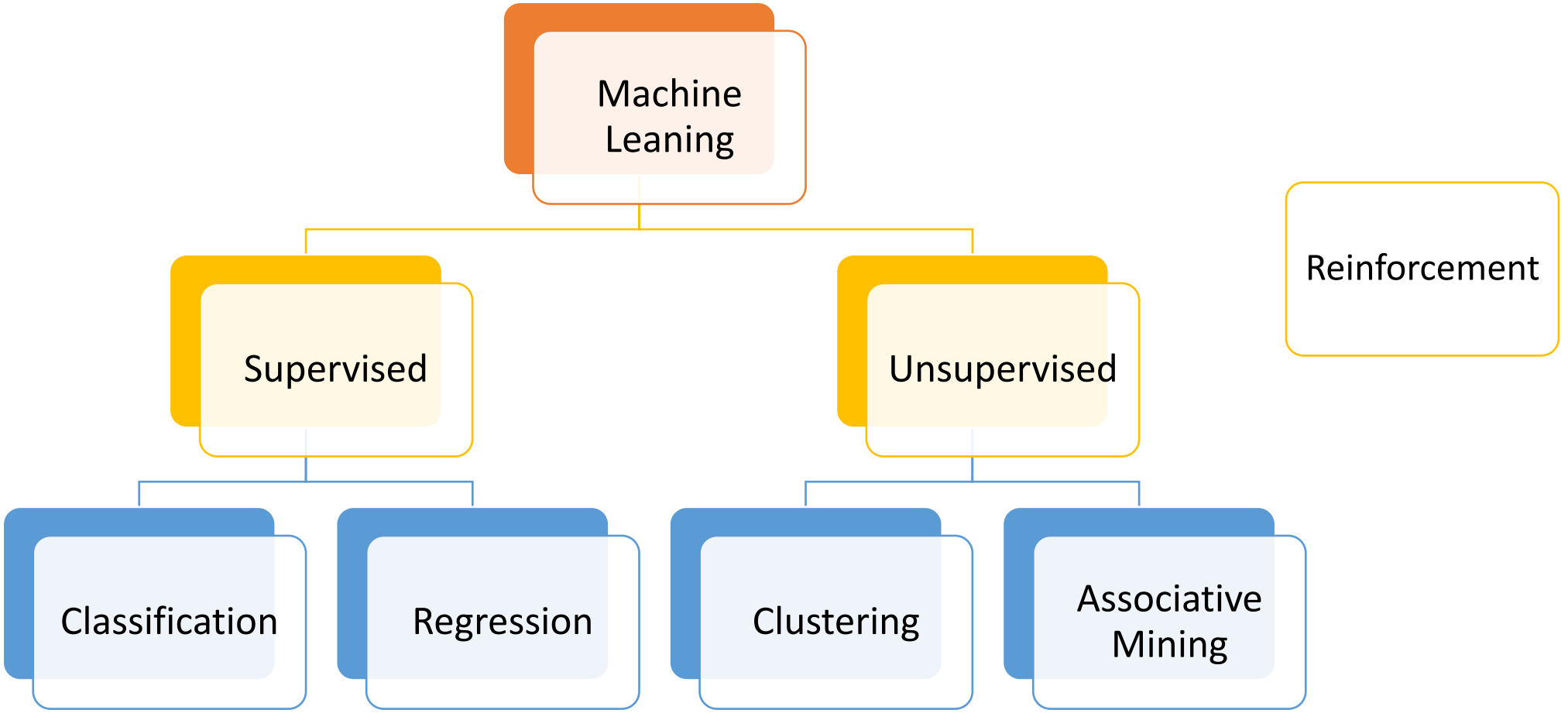
Self-driving Cars

Source: Google Images

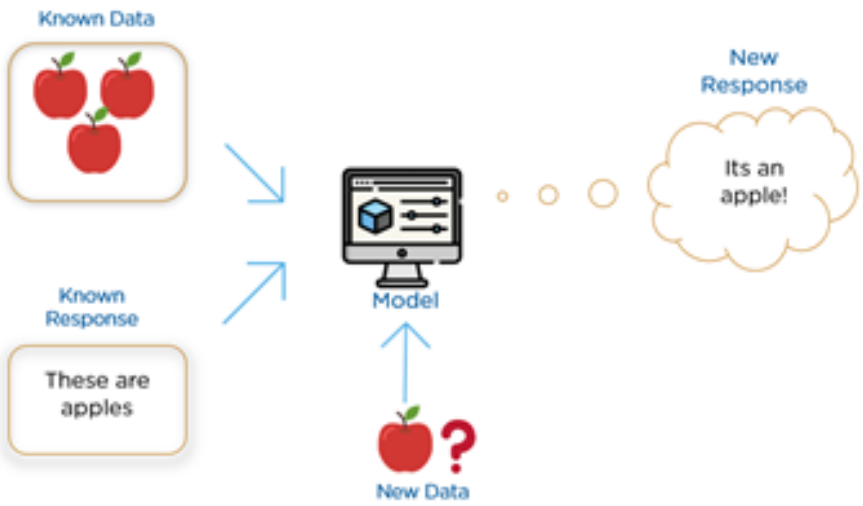
Applications (contd…)

Recommender System

Categories of Machine Learning Algorithms



Supervised learning



Somebody is telling you exactly what's right and what's wrong.

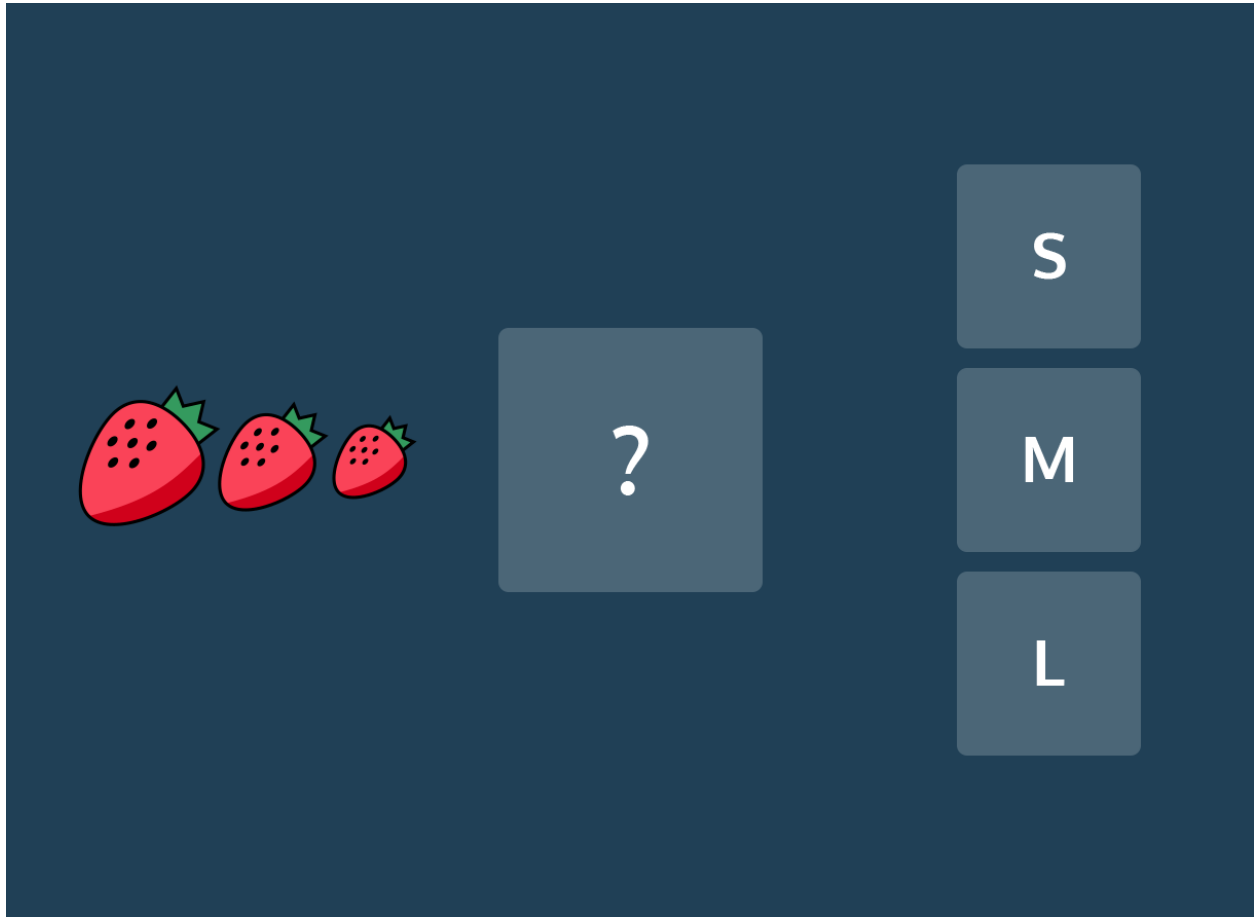
Applications: Face Recognition, Fingerprint recognition, Suggestions given by websites, Multi-touch gestures on gadgets and many more.

The two tasks of supervised learning: regression and classification

Regression:	Classification:
Predict a continuous numerical value. How much will that house sell for?	Assign a label. Is this a picture of a cat or a dog?

Supervised Learning

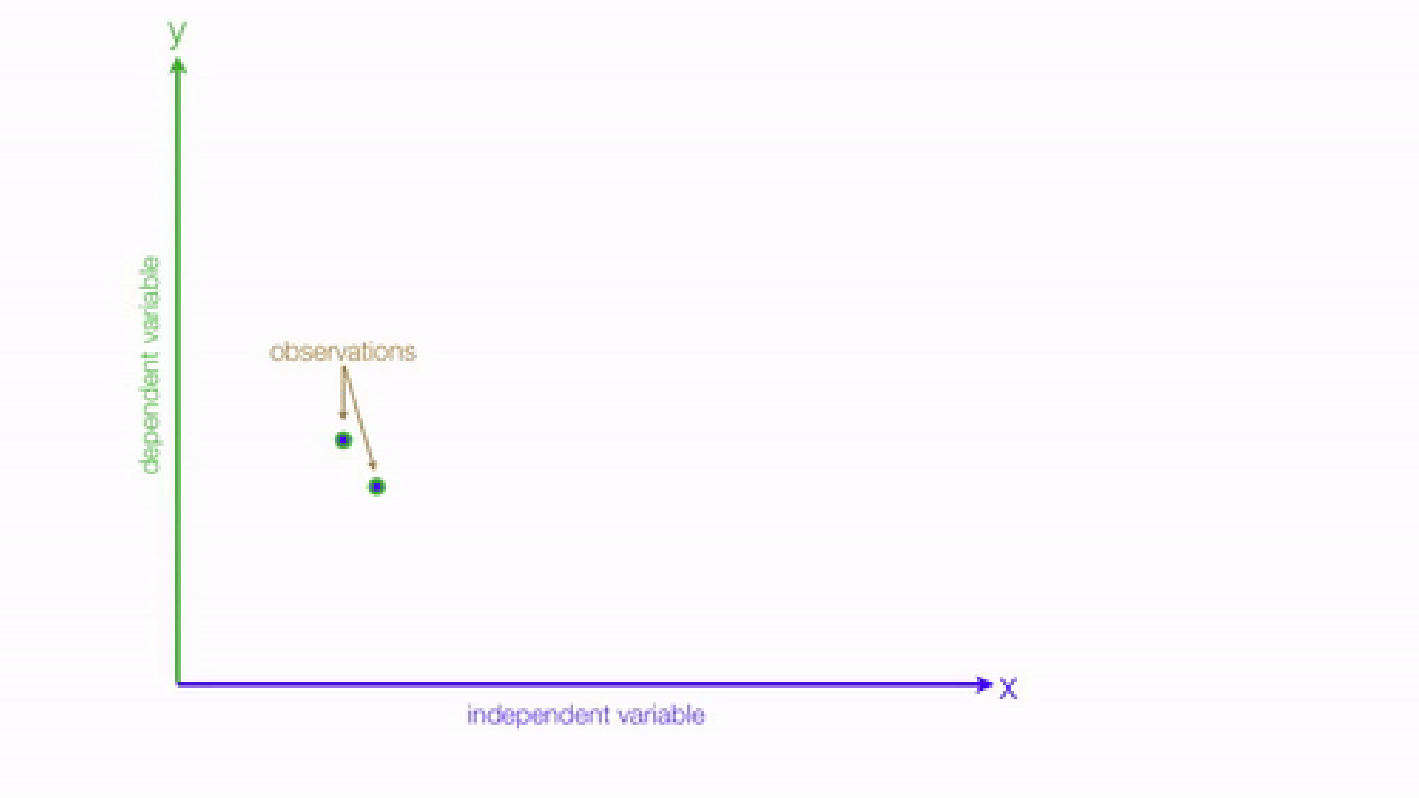
Classification



Source: https://www.codecademy.com/articles/regression-vs-classification

Supervised Learning

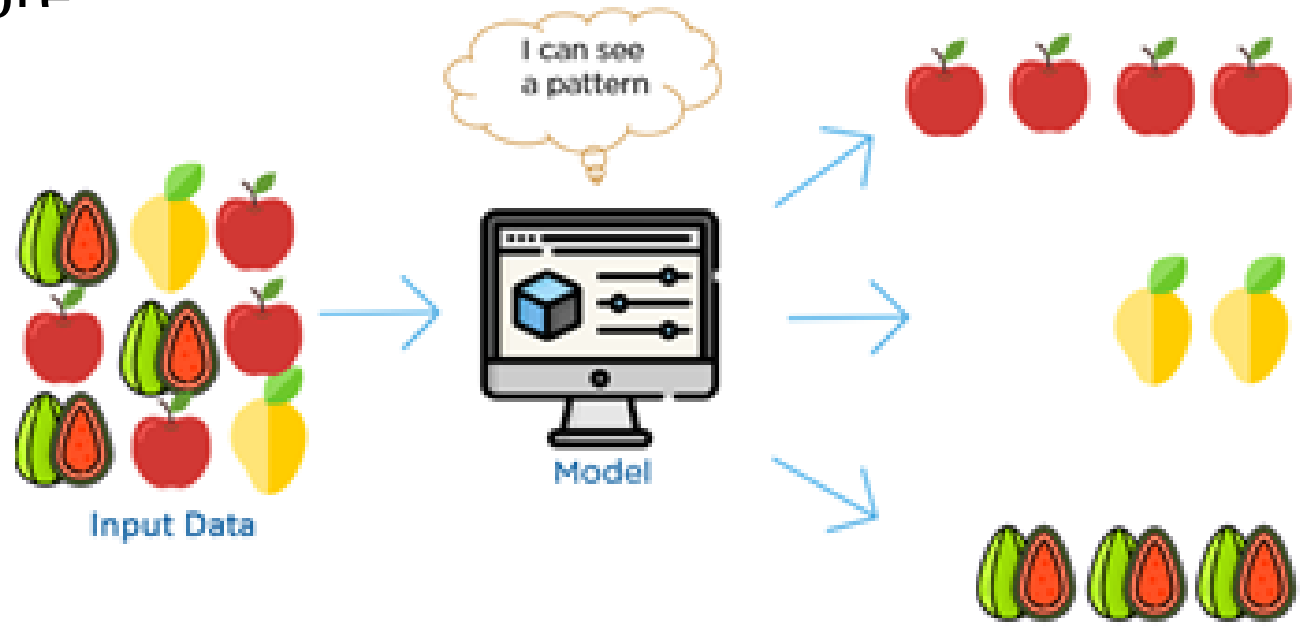
Regression



Source: https://gfycat.com/gifs/search/linear+regression

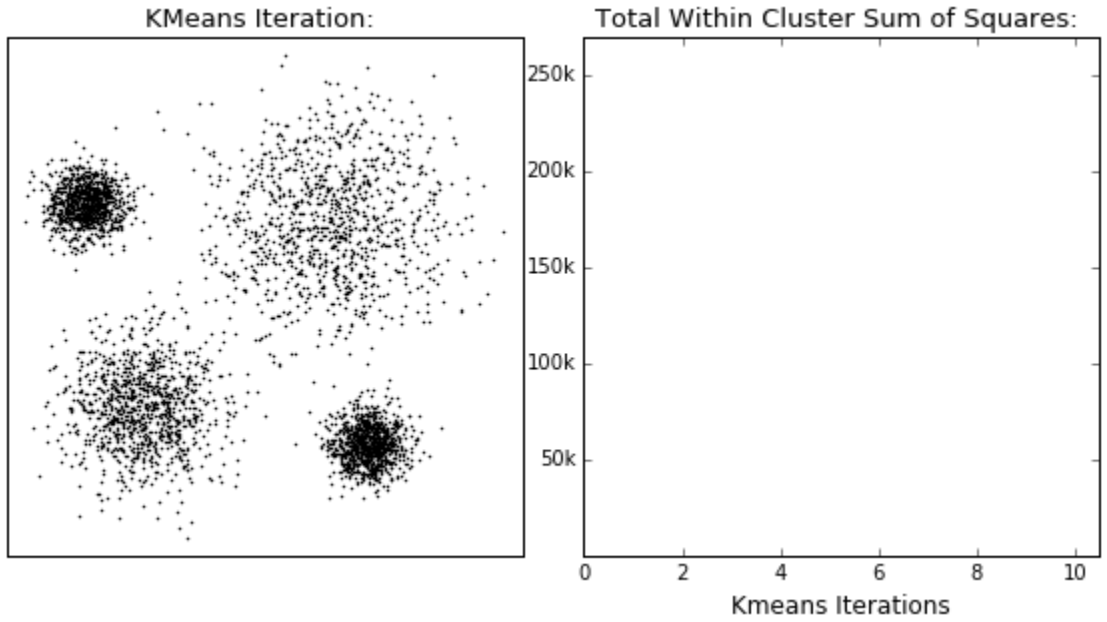
Unsupervised Learning

- Nobody is teaching you how to make your decisions.
- Grouping on the basis of inherent structure of data.
- Applications: Medical Imaging, Genetic Clustering, Market Analysis, Search Engines and many more



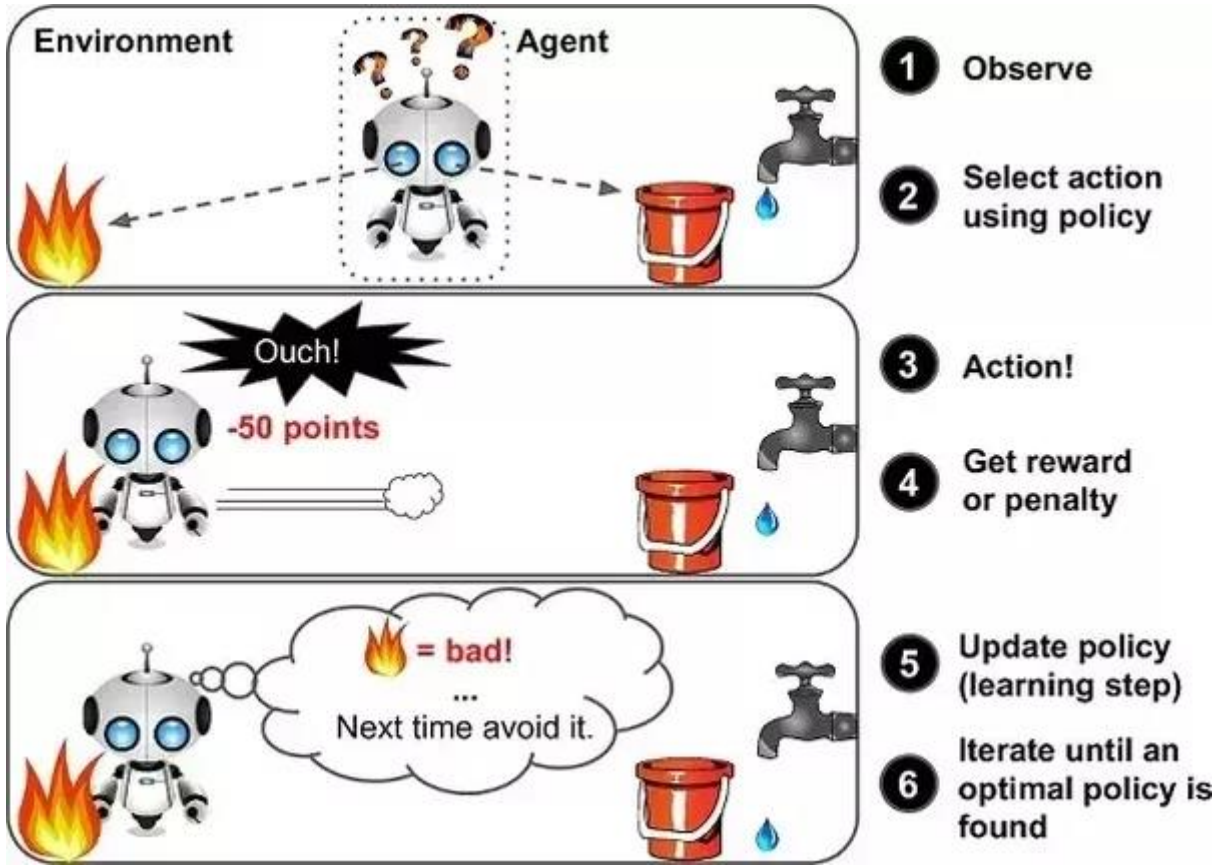
Unsupervised Learning

Clustering



Source: https://dashee87.github.io/data%20science/general/Clustering-with-Scikit-with-GIFs/

Reinforcement Learning



Learning with a critic
Feedback in the form of right or wrong category .

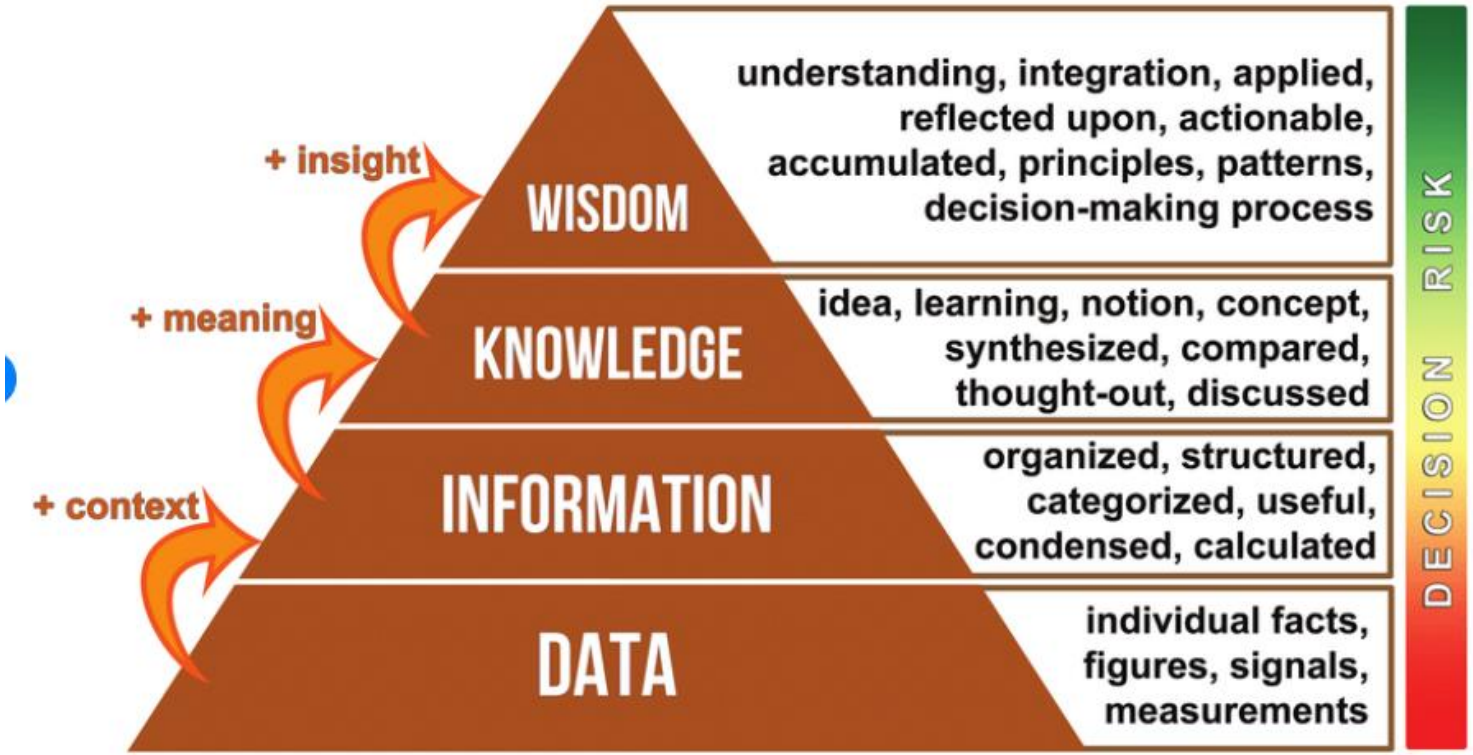
E.g. Autonomous machines exploring unknown terrains (like cars or robots), Telecommunication networks, Sensor Networks, Finance and many more.

Questions: Identify the nature of the problem

- Scenario 1
  - Given data of size of houses on the real estate market, try to predict their price. **Regression**
- Scenario 2
  - Given a picture of a person, identify whether the person is of high school or College or Graduated.

    Classification
- Scenario 3
  - Given a picture of a person, identify the age of a person. **Regression**
- Scenario 4
  - Consider a collection of 1000 easy written on Indian Economy. Try to find out a way to group these essays into a small number that are similar or related.

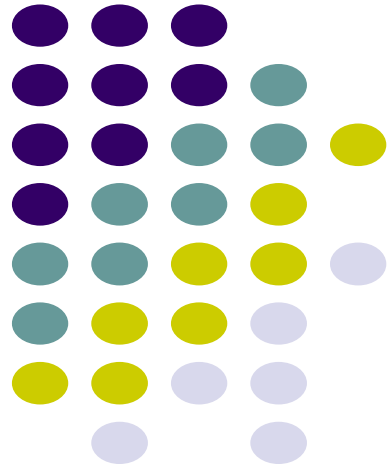
    Clustering



The data-information-knowledge-wisdom (DIKW) hierarchy as a pyramid to manage knowledge. Reproduced with permission from Tedeschi (2019).

Source: Google Images

Neural Networks

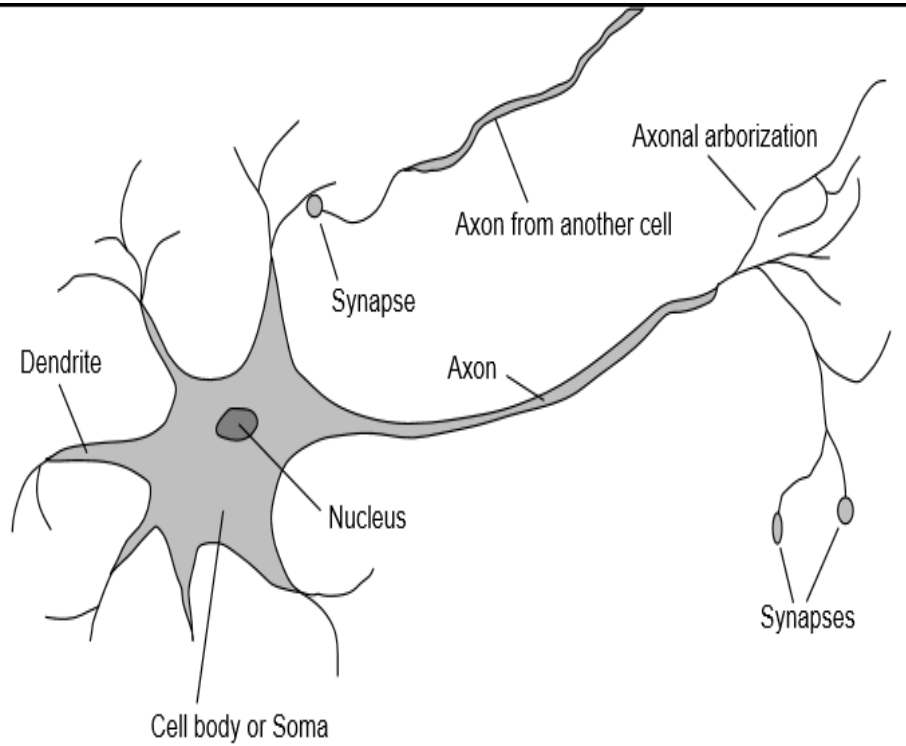


Biological neuron model

The brain consists of a densely interconnected set of nerve cells, or basic information-processing units, called neurons.

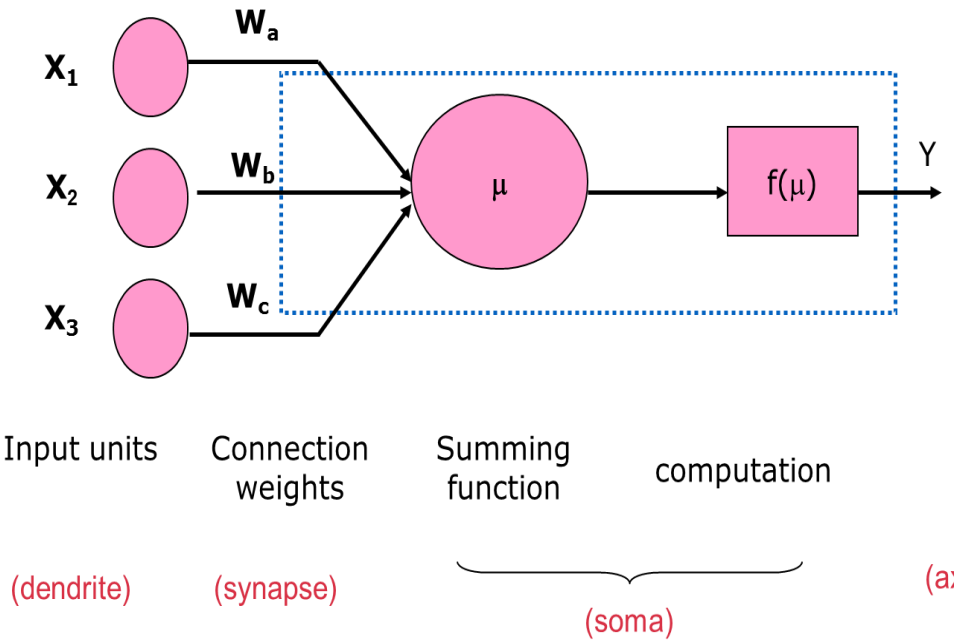
A neuron consists of:

- **Dendrites**: They are tree-like branches, responsible for receiving the information from other neurons it is connected to. In other sense, we can say that they are like the ears of neuron.
- **Soma**: It is the cell body of the neuron and is responsible for processing of information, they have received from dendrites.
- **Axon**: It is just like a cable through which neurons send the information
- **Synapses**: It is the connection between the axon and other neuron dendrites.



How working of the brain inspires?

- Inputs are received on dendrites, and if the input levels are over a threshold, the neuron fires, passing a signal through the axon to the synapse which then connects to another neuron.



- Human brain contains approx 10^{11} neurons and approx. 10^{14} connections

Brain	Neural Network
Neuron	Node
Connection of neurons	Connection weight

Information processing model that is inspired by the way biological nervous system (i.e) the brain, process information.

Artificial Neural Networks

Nodes – interconnected processing elements (units or neurons)

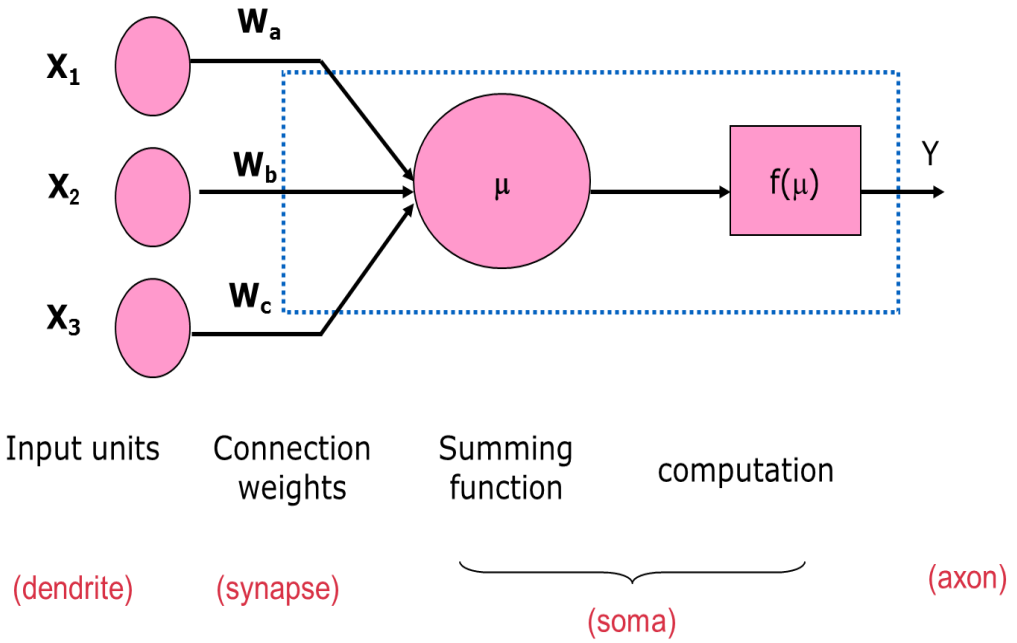
Neuron is connected to other by a *connection link.*

Each connection link is associated with *weight* which has information about the input signal.

ANN processing elements are called as *neurons or artificial neurons* , since they have the capability to model networks of original neurons as found in brain.

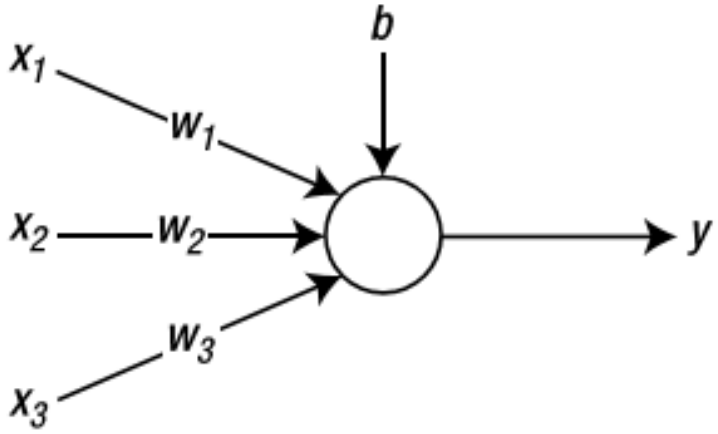
Internal state of neuron is called *activation or activity level* of neuron, which is the function of the inputs the neurons receives.

Neuron can send only one signal at a time.



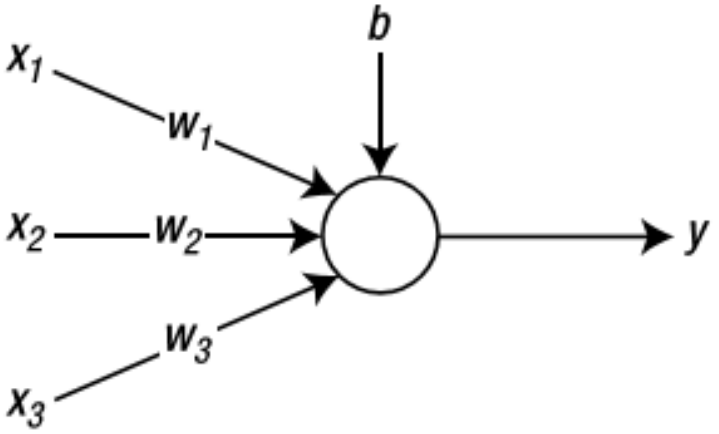
Artificial Neuron

- The circle and arrow of the figure denote the node and signal flow, respectively.
- $x_1 , x_2 , and\ x_3$ are the input signals.
- $w_1 , w_2 , and\ w_3$ are the weights for the corresponding signals.
- Lastly, b is the bias, which is another factor associated with the storage of information.
- In other words, the information of the neural net is stored in the form of weights and bias.



The weighted sum of the input signals is calculated.

$$v = w_1 x_1 + w_2 x_2 + w_3 x_3 + b$$
$$= wx + b$$



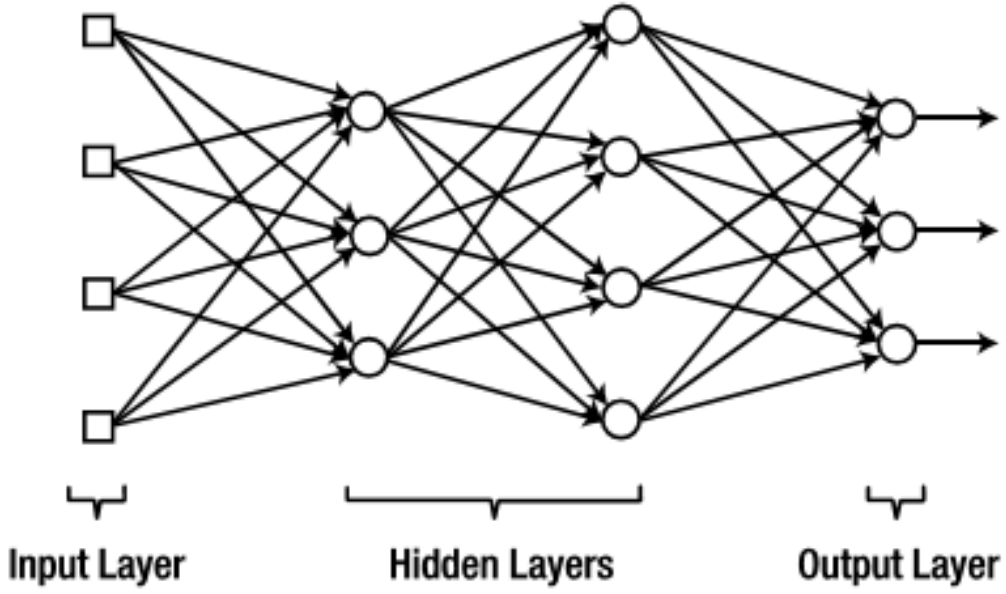
The output from the activation function to the weighted sum is passed outside.

$$y = \varphi(v) = \varphi(wx + b)$$

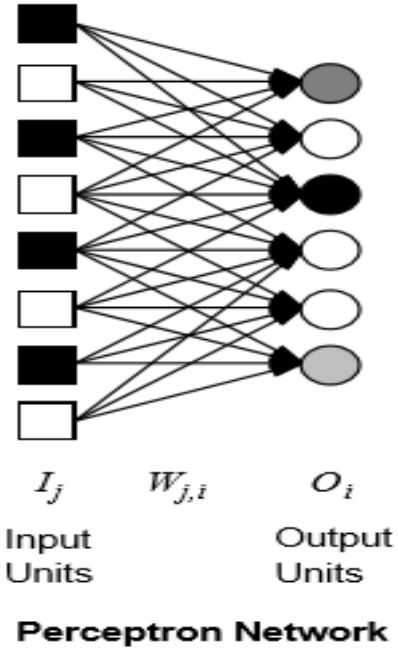
$$w = \begin{bmatrix} w_1 & w_2 & w_3 \end{bmatrix} \qquad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Layers of Neural Network

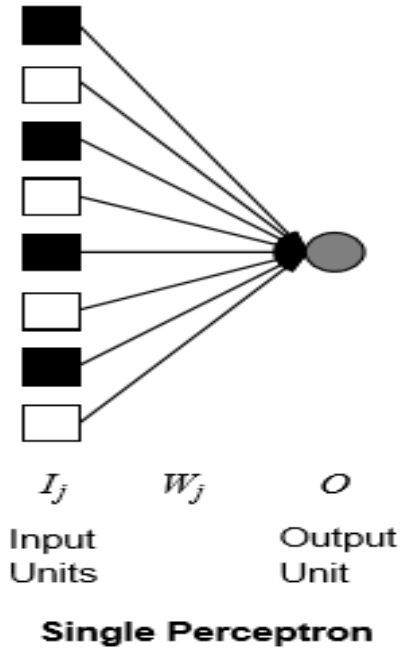
- As the brain is a gigantic network of the neurons, the neural network is a network of nodes



- Introduced in the late 50s – Minsky and Papert.
- A single layer perceptron (SLP) is the simplest feed-forward artificial neural network that can learn to classify any linearly separable set of inputs



Perceptron Network

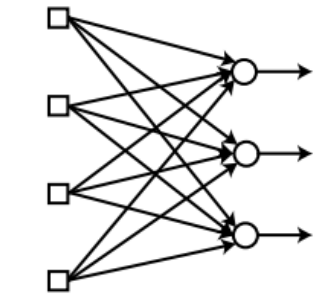


Single Perceptron

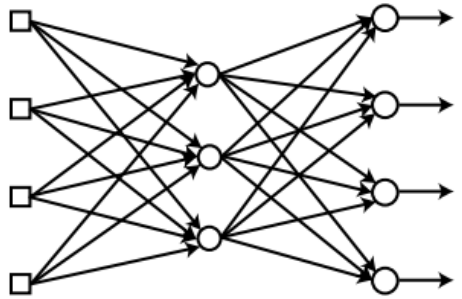
Network Architectures

- This describes the set of connections between the neurons, the number of layers, and the number of neurons in each layer.

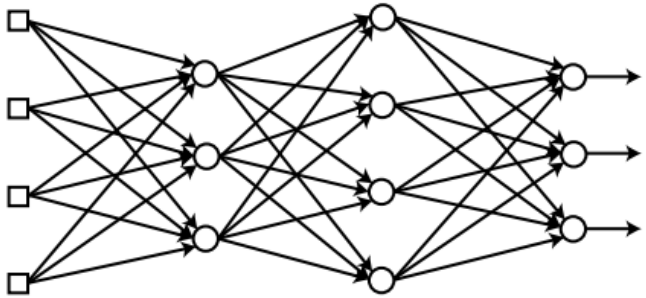
Single-Layer Neural Network		Input Layer - Output Layer
Multi-Layer Neural Network	Shallow Neural Network	Input Layer - Hidden Layer - Output Layer
	Deep Neural Network	Input Layer - Hidden Layers - Output Layers



Single-layer Neural Network



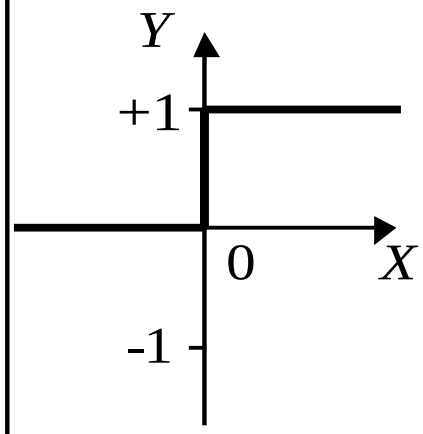
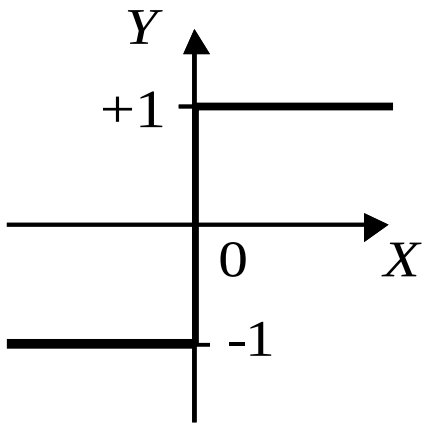
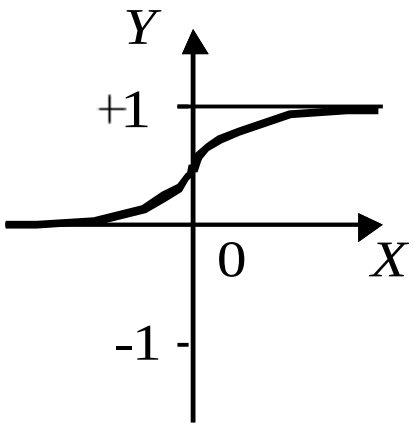
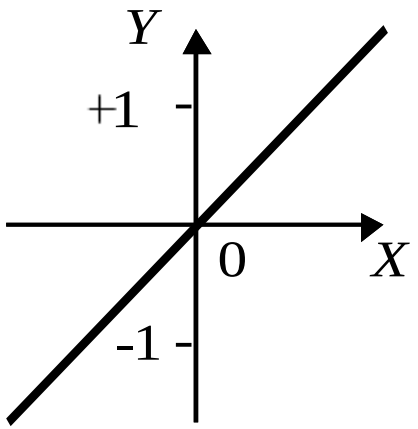
(Shallow) Multi-layer Neural Network



Deep Neural Network

Activation Functions

- Activation function is applied over the net input to calculate the output of an ANN.
- Information processing of processing element has two major parts: input and output.

Step function	*Sign function*	*Sigmoid function*	*Linear function*
			
$Y^{step} = \begin{cases} 1, & \text{if } X \geq 0 \\ 0, & \text{if } X < 0 \end{cases}$	$Y^{sign} = \begin{cases} +1, & \text{if } X \geq 0 \\ -1, & \text{if } X < 0 \end{cases}$	$Y^{sigmoid} = \frac{1}{1+e^{-X}}$	$Y^{linear} = X$

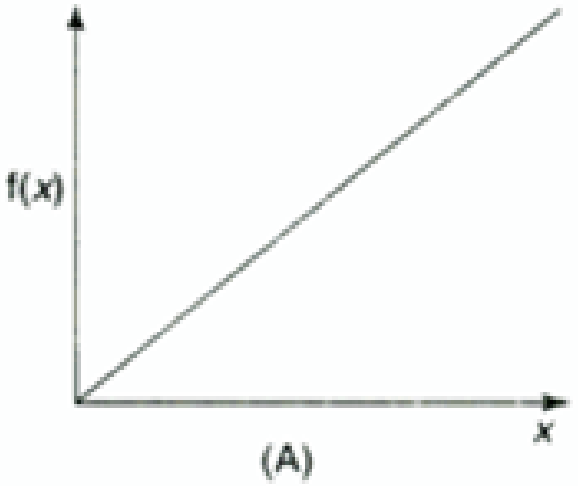
Activation Functions

- Activation function is applied over the net input to calculate the output of an ANN.
- Information processing of processing element has two major parts: input and output.

1. Identity function:
   - It is a linear function which is defined as

     $f(x) = x$ *for all* x

   - The output is same as the input.



Binary step function

It is defined as

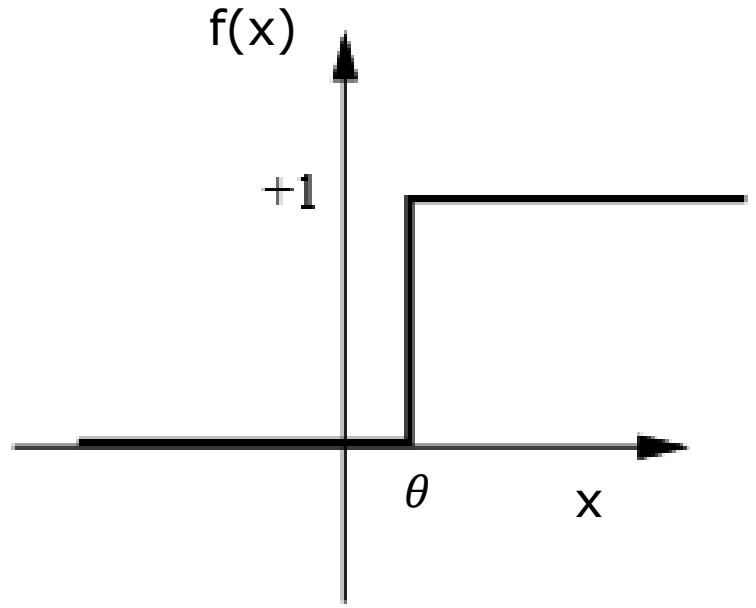
$f(x) = 1$ if $x >= \theta$

$= 0$ if $x < \theta$

where θ represents thresh hold value.

It is used in single layer nets to

convert the net input to an output that is binary.



3. Bipolar step function:

It is defined as

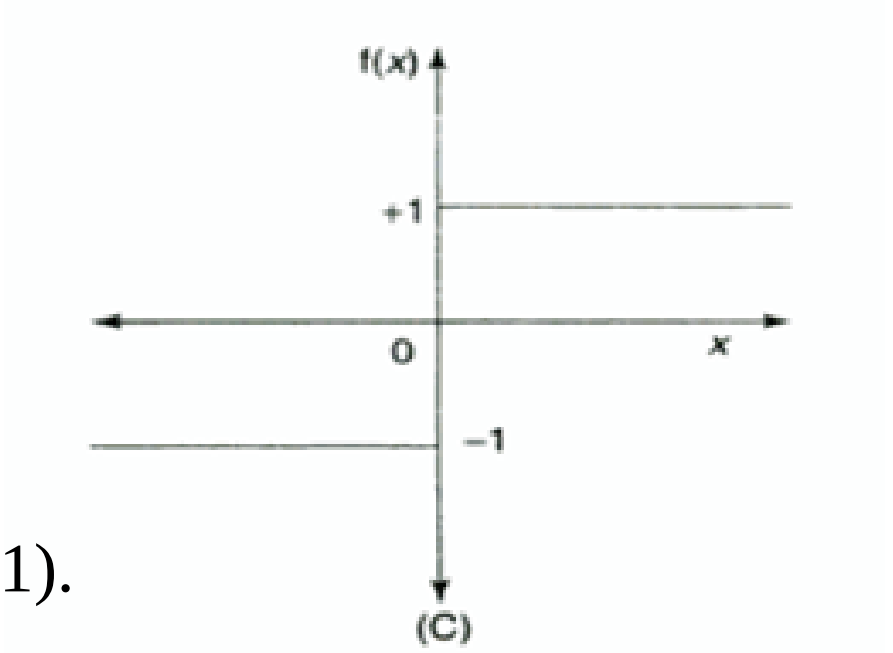
$f(x) = 1$ if $x >= \theta$

$= -1$ if $x < \theta$

where θ represents threshold value.

- It is used in single layer nets to convert

the net input to an output that is bipolar (+1 or -1).



4. Sigmoid function:

It is defined as

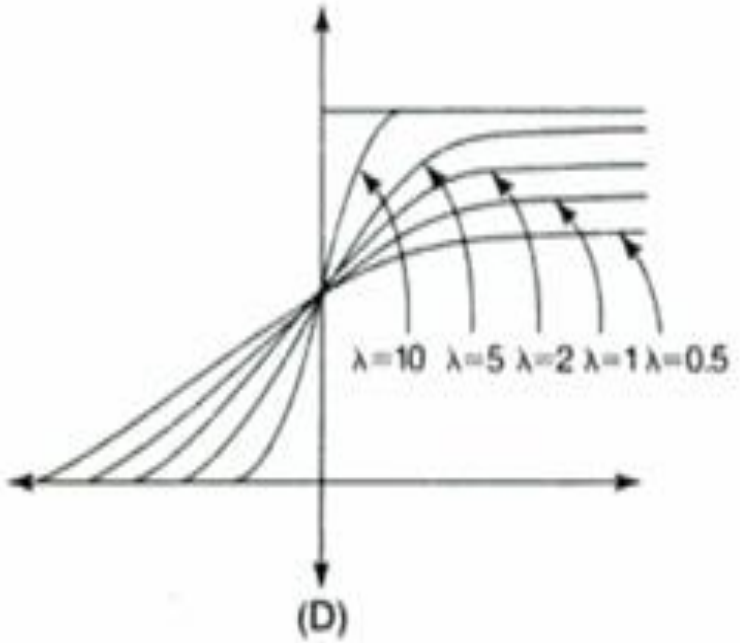
$$f(x) = \frac{1}{1 + e^{-\lambda x}}$$

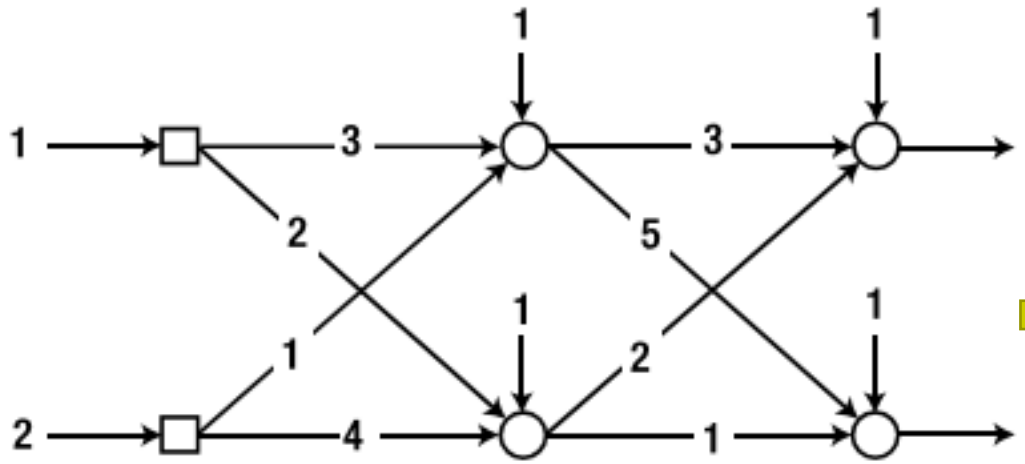
where λ – steepness parameter.

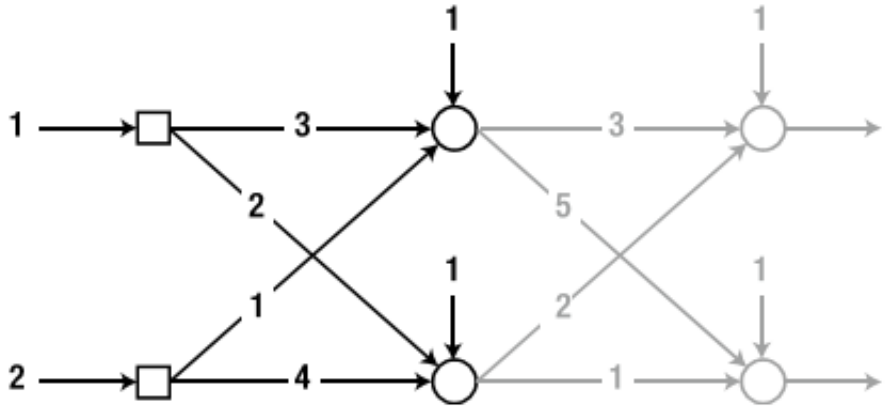
The derivative of this function is

$$f'(x) = \lambda f(x)[1-f(x)].$$

The range of sigmoid function is 0 to 1.







The activation function of each node is assumed to be a linear function

$$\varphi(x) = x$$

The first node of the hidden layer calculates the output as:

Weighted sum: $v = (3\times1) + (1\times2) + 1 = 6$

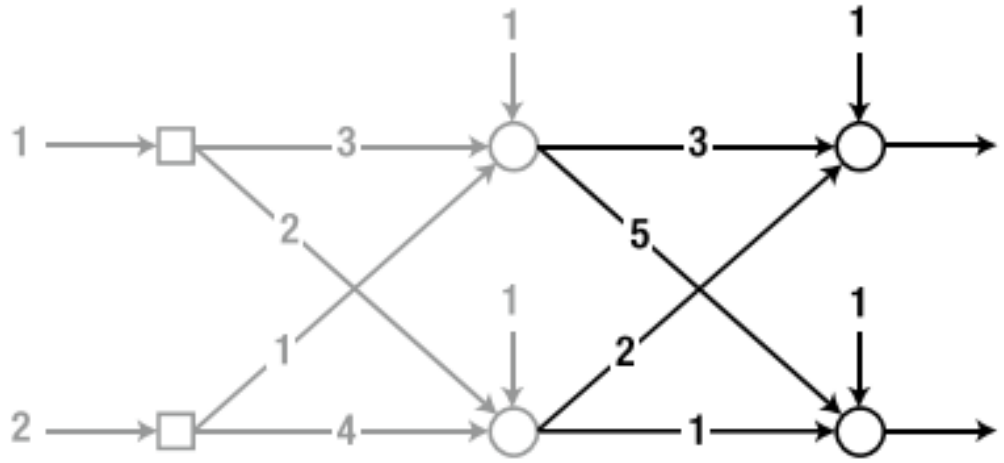
Output: $y = \varphi(v) = v = 6$

In a similar manner, the second node of the hidden layer calculates the output as:

Weighted sum: $v = (2\times1) + (4\times2) + 1 = 11$

Output: $y = \varphi(v) = v = 11$

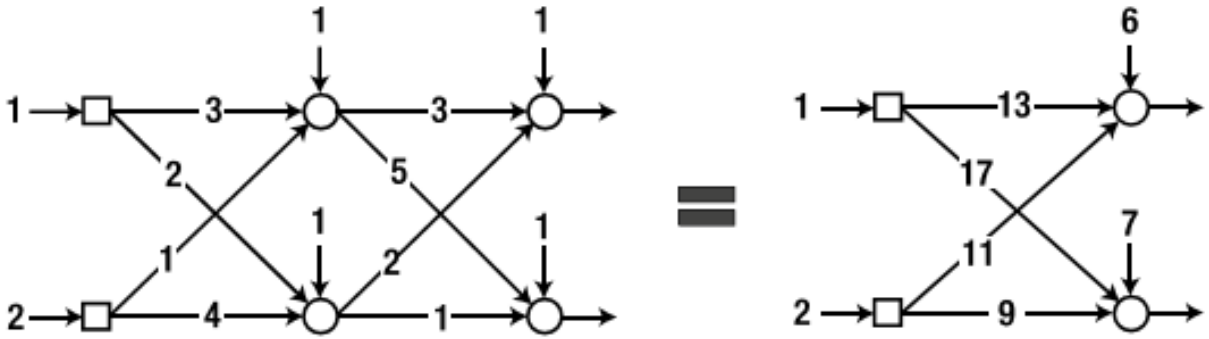
$$v = \begin{bmatrix} 3\times1+1\times2+1 \\ 2\times1+4\times2+1 \end{bmatrix} = \begin{bmatrix} 3 & 1 \\ 2 & 4 \end{bmatrix}\begin{bmatrix} 1 \\ 2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ 11 \end{bmatrix}$$



Weighted sum: $v = \begin{bmatrix} 3 & 2 \\ 5 & 1 \end{bmatrix} \begin{bmatrix} 6 \\ 11 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 41 \\ 42 \end{bmatrix}$

Output: $y = \varphi(v) = v = \begin{bmatrix} 41 \\ 42 \end{bmatrix}$

- Effect of linear activation function



Training Method or Learning :

Training Method	Training Data
Supervised Learning	{ input, correct output }
Unsupervised Learning	{ input }
Reinforced Learning	{ input, some output, grade for this output }

Supervised Learning:

Application: Recognizing hand-written digits, pattern recognition and etc.
Examples: Perceptron, feed-forward neural network, radial basis function

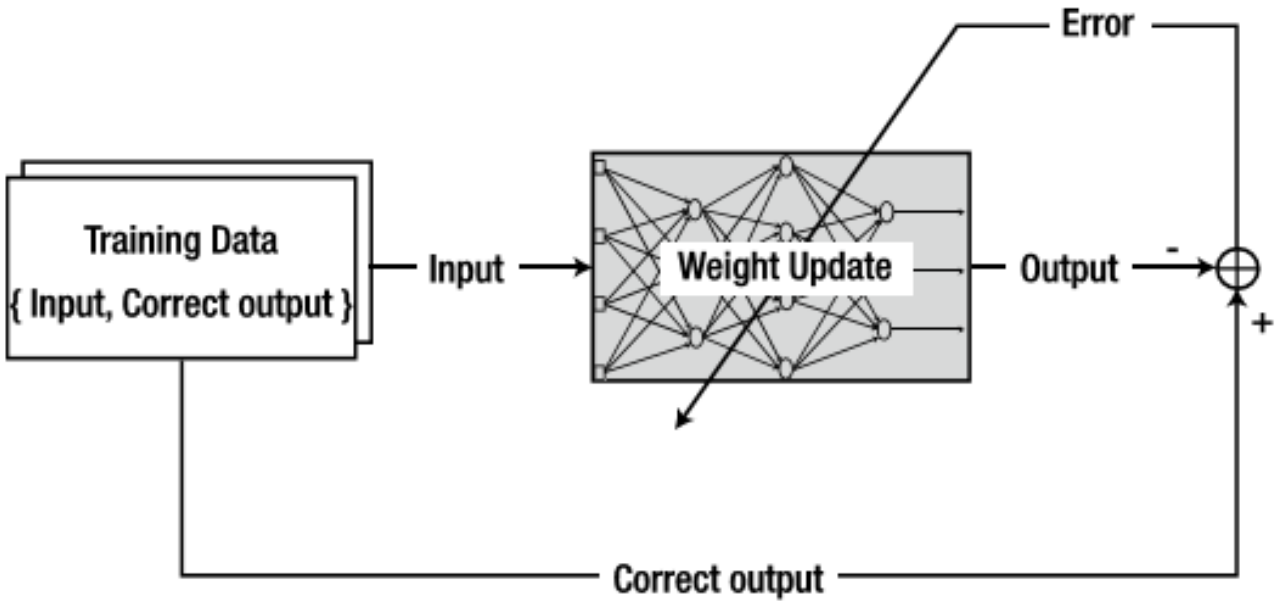
Reinforcement Learning:

Application: Text Prediction
Examples: Recurrent neural networks

Unsupervised Learning:

Application: Clustering
Examples: Kohonen, SOM , Hopfield networks.

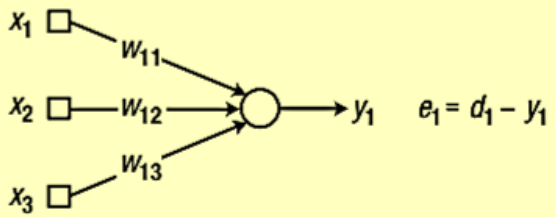
Supervised Learning



1. Initialize the weights with adequate values.
2. Take the "input" from the training data, which is formatted as { input, correct output }, and enter it into the neural network. Obtain the output from the neural network and calculate the error from the correct output.
3. Adjust the weights to reduce the error.
4. Repeat Steps 2-3 for all training data

Training of a Single-Layer Neural Network: Delta Rule

- It's a single-unit network.
- Let d_i is the correct output of the output node i
- Change the weight by an amount proportional to the difference between the desired output and the actual output



where

x_j = The output from the input node j, (j = 1, 2, 3)

e_i = The error of the output node i

w_{ij} = The weight between the output node i and input node j

α = Learning rate ($0 < \alpha \leq 1$)

Delta Rule

1. Initialize the weights at adequate values.
2. Take the "input" from the training data of { `input, correct output` } and enter it to the neural network. Calculate the error of the output, y_i, from the correct output, d_i, to the input.

$$e_i = d_i - y_i$$

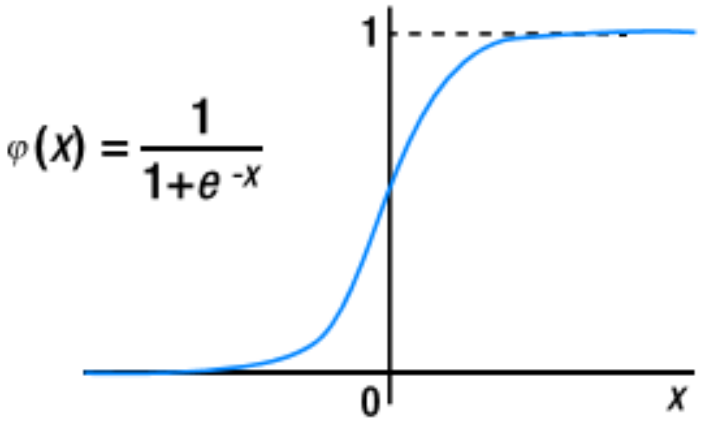
3. Calculate the weight updates according to the following delta rule:

$$\Delta w_{ij} = \alpha e_i x_j$$

4. Adjust the weights as:

$$w_{ij} \leftarrow w_{ij} + \alpha e_i x_j$$

5. Perform Steps 2-4 for all training data.
6. Repeat Steps 2-5 until the error reaches an acceptable tolerance level.



We need the derivative of this function, which is given as:

$$\varphi'(x) = \varphi(x)(1-\varphi(x))$$

$$\delta_i = \varphi'(v_i)e_i = \varphi(v_i)(1-\varphi(v_i))e_i$$

$$w_{ij} \leftarrow w_{ij} + \alpha\varphi(v_i)(1-\varphi(v_i))e_i x_j$$

$$\delta_i = \varphi'(v_i)e_i$$

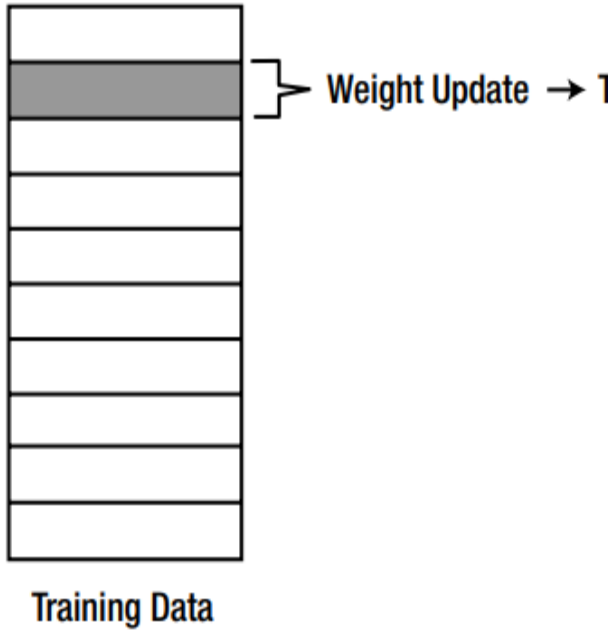
$$w_{ij} \leftarrow w_{ij} + \alpha\delta_i x_j$$

v_i = The weighted sum of the output node i

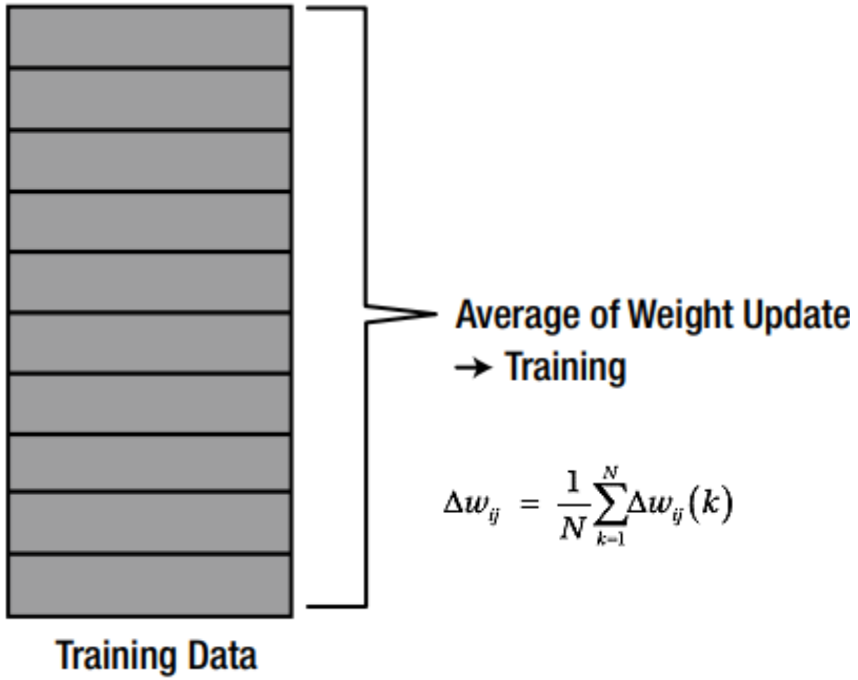
φ' = The derivative of the activation function φ of the output node i

Error Calculations in ANN

- **Stochastic Gradient Descent**

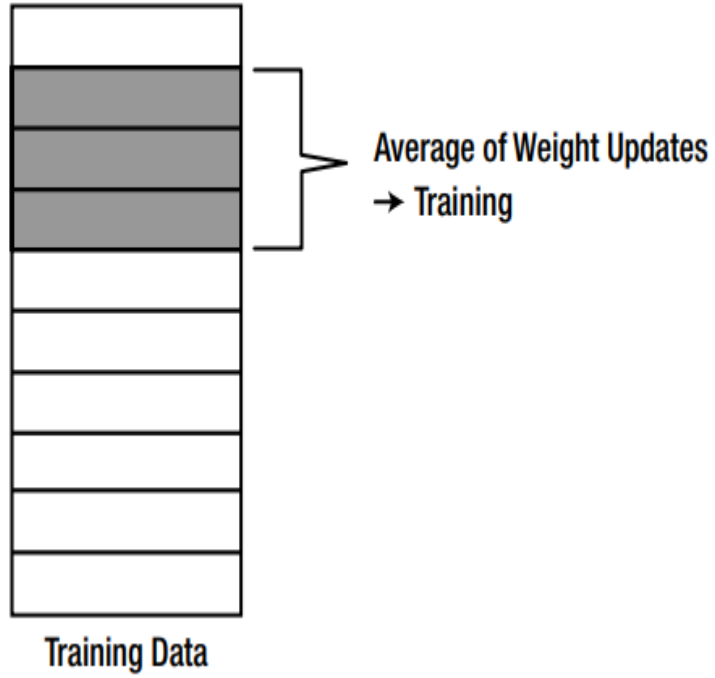


Batch



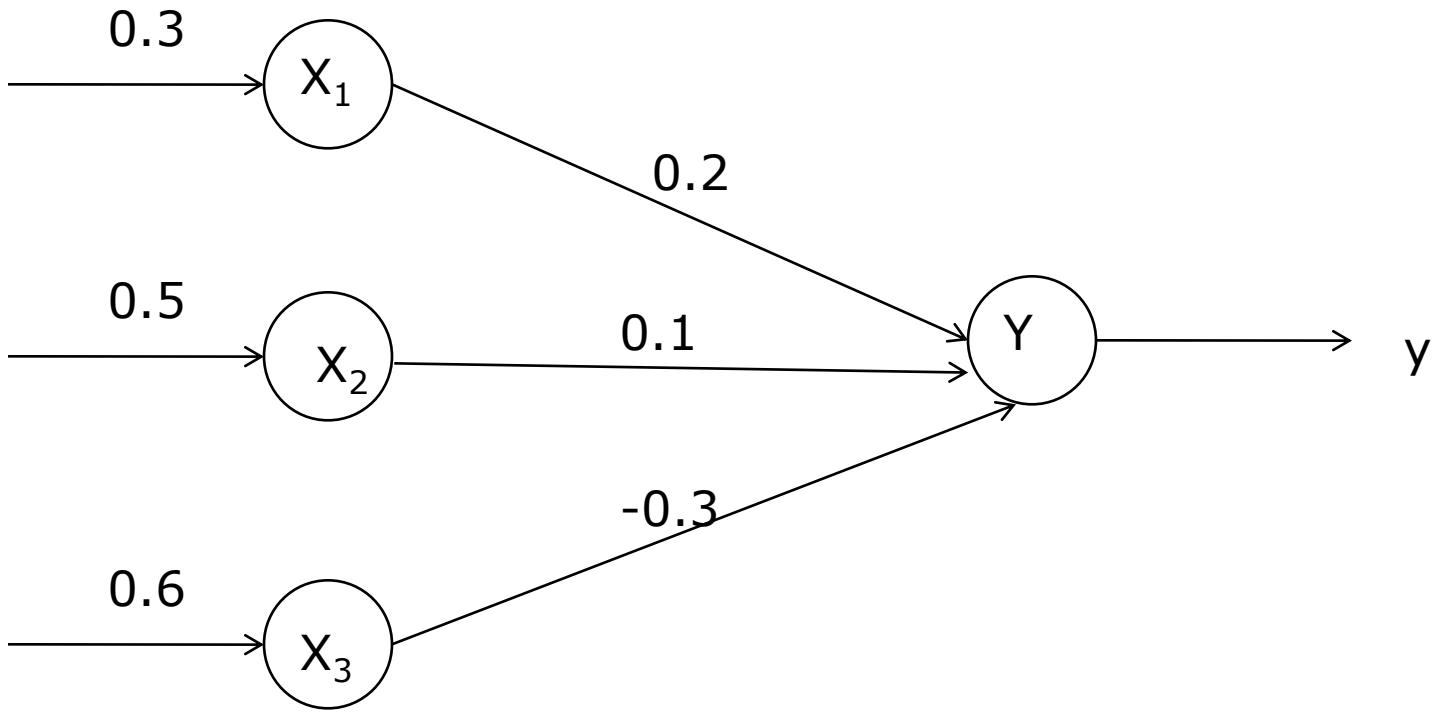
$$\Delta w_{ij} = \frac{1}{N}\sum_{k=1}^{N}\Delta w_{ij}(k)$$

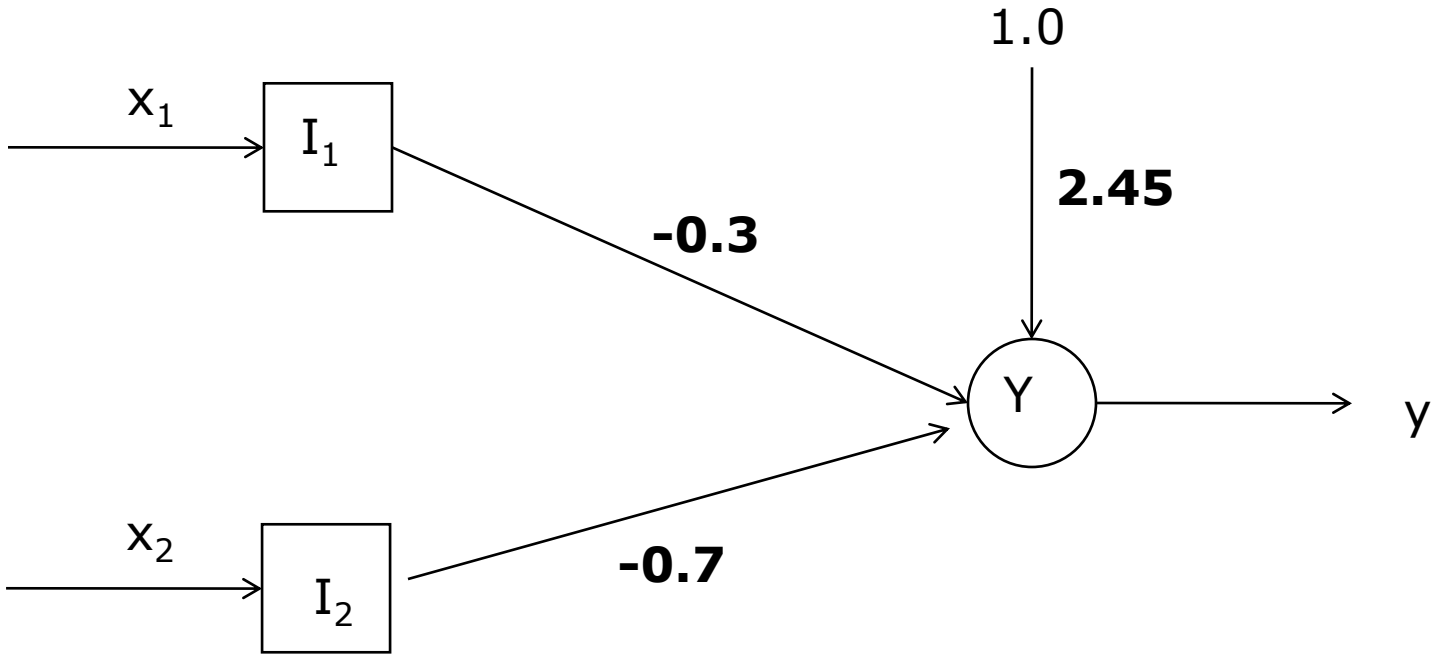
Mini batch.



Examples

- Calculate the net input to the output neuron




1.0
x_1
I_1
2.45
-0.3
Y
y
x_2
I_2
-0.7

Calculate the output of the neuron Y using activation function

1) Step function with threshold 0.5
2) Sign function with threshold 0.5
3) Linear function