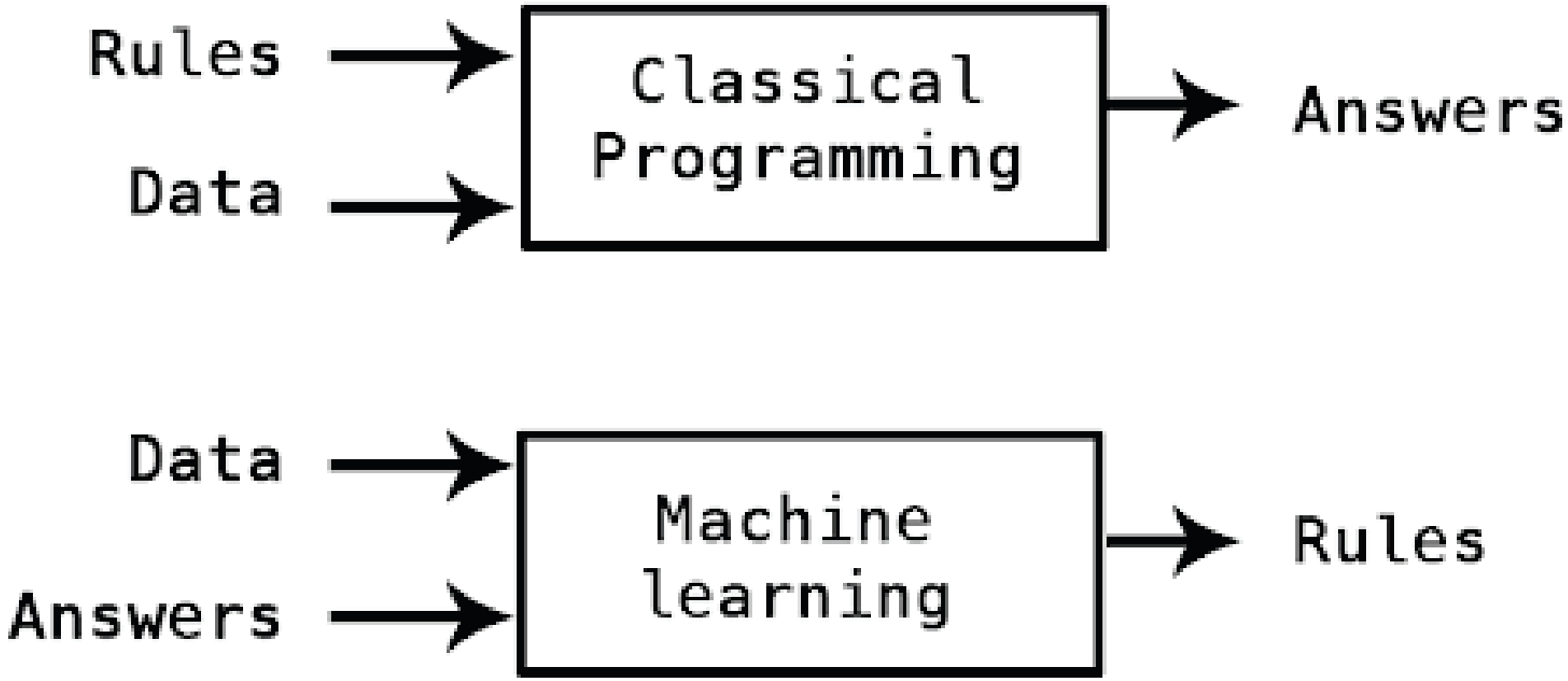
Artificial Intelligence

GETTING INSIDE

History of AI

- It was born in 1950s.
- *"the effort to automate intellectual tasks normally performed by humans".*
- For a fairly long time, many experts believed that human-level artificial intelligence could be achieved by having programmers handcraft a sufficiently large set of explicit rules for manipulating knowledge. This approach was known as *symbolic AI.*
- Symbolic AI – intractable to figure out explicit rules for tasks such as – speech recognition, NLP, Image classification etc.
- Hence came – ML.
- Alan Turing – 1950 – Turing Test.

History of AI



History of AI

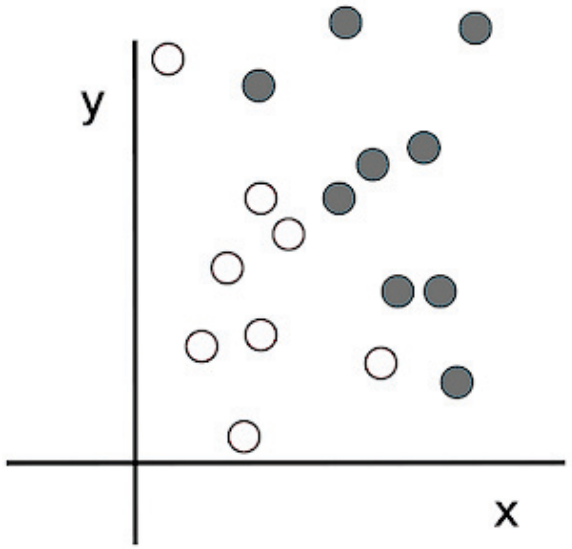
- Trending in 1990s – large datasets and hardware (CPU to GPU to TPU)
- A lot of statistical analysis and rules
- The birth of Deep Learning
- 3 parameters required for Deep Learning –

  - ***Input data points***—For instance, if the task is speech recognition, these data points could be sound files of people speaking. If the task is image tagging, they could be picture files.
  - ***Examples of the expected output***—In a speech-recognition task, these could be human-generated transcripts of sound files. In an image task, expected outputs could tags such as "dog", "cat", and so on.
  - ***A way to measure whether the algorithm is doing a good job***—This is necessary in order to determine the distance between the algorithm's current output and its expected output. The measurement is used as a feedback signal to adjust the way the algorithm works. This adjustment step is what we call *learning*.

History of AI

- The objective is to learn useful *representations* of the input data at hand—representations that get us closer to the expected output.
- But wait! What is a representation?
- A different way to look at data – to represent or to encode.
- Example –
  - Colors – RGB or HSV
  - Select all red pixels – RGB
  - Reduce saturation - HSV

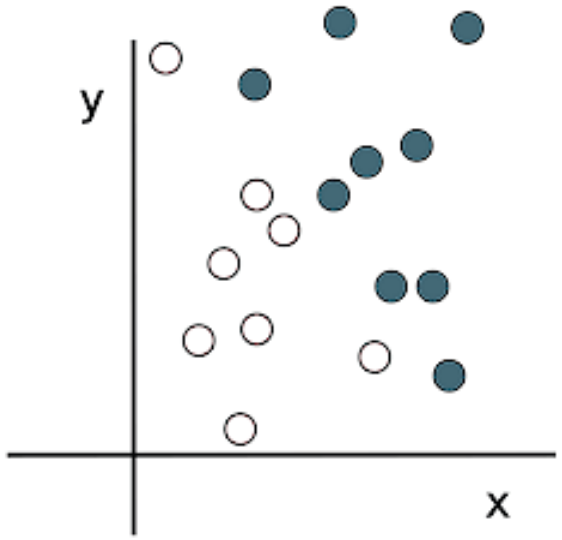
Let’s classify



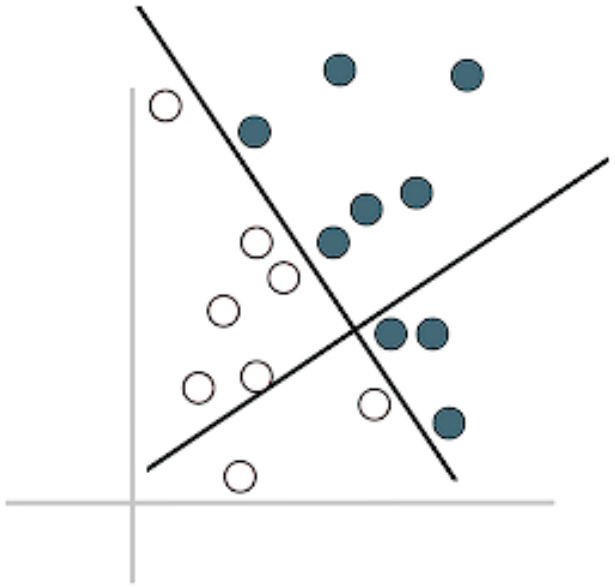
Input – Coordinates of the points
Expected Output – Colors
Measurement - % of points that are correctly classified

Let’s Classify

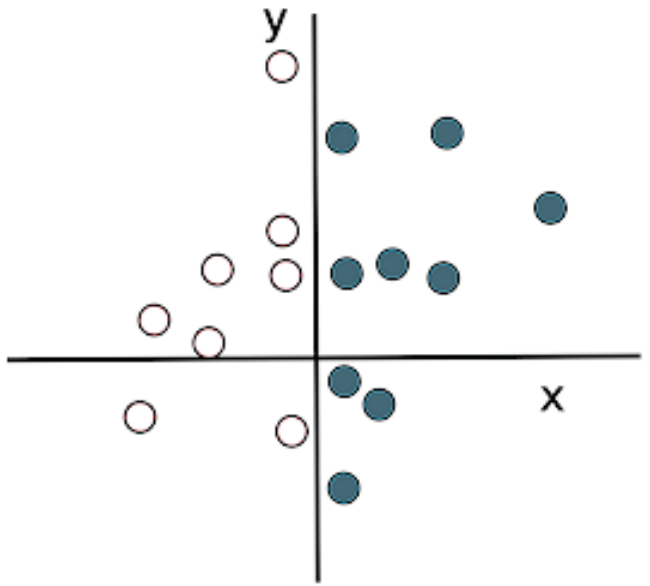
1: Raw data



2: Coordinate change



3: Better representation



So what is Machine Learning?

- All machine-learning algorithms consist of automatically finding such transformations that turn data into more useful representations for a given task.
- *Learning*, in the context of machine learning, describes an automatic search process for better representations.

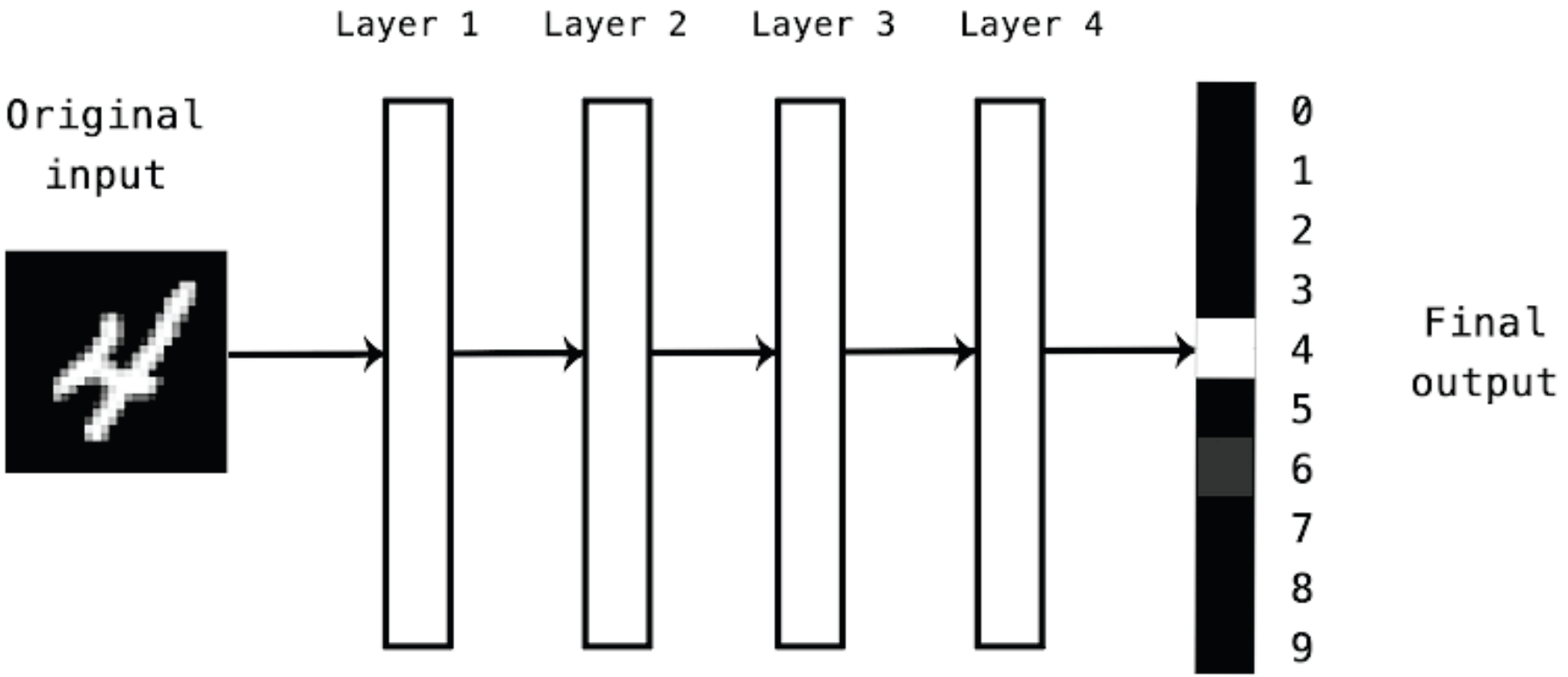
Model vs Network

- Set of algorithms defined to perform a task – model.
- Computational data graphs – network.

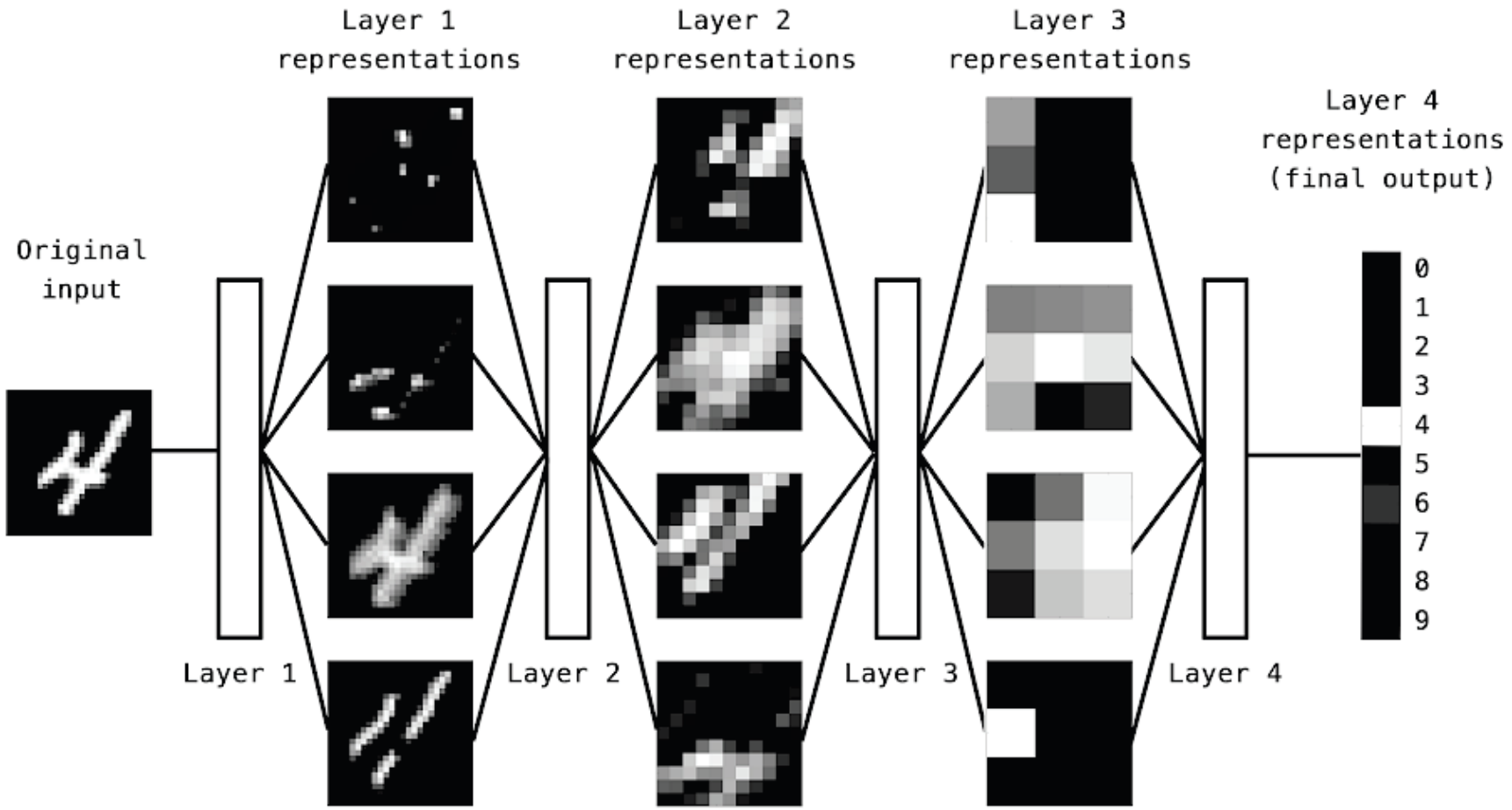
Then what is Deep Learning?

- It is not deeper understanding of the system.
- It is not the depth to which a system goes to understand.
- It is – the number of layers in a network (neural network).
- Simply, it is the answer to this question –
  - How many layers contribute to a model of the data?

Neural Networks

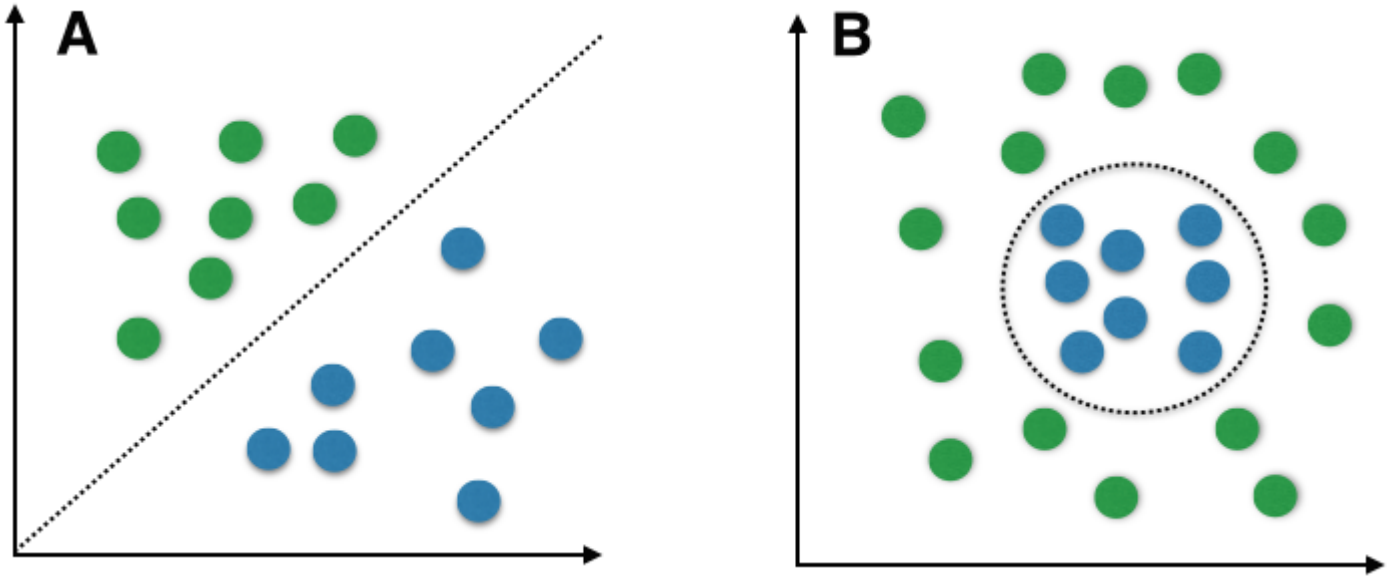


Neural Networks



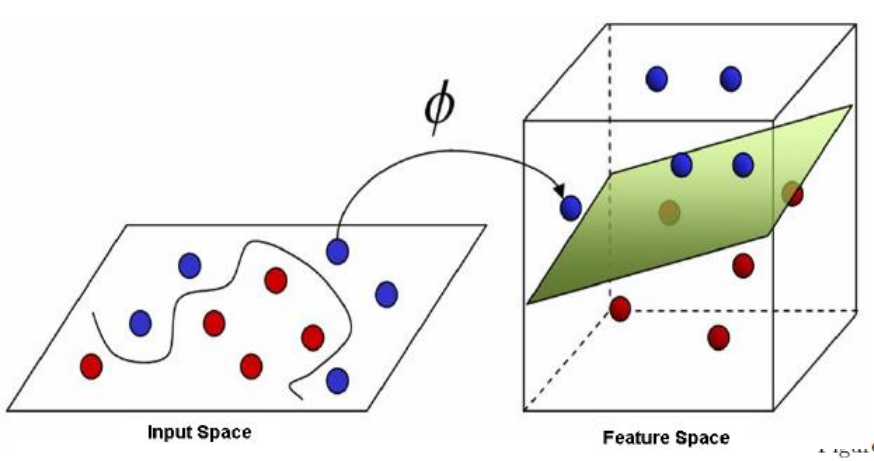
Algorithms of AI

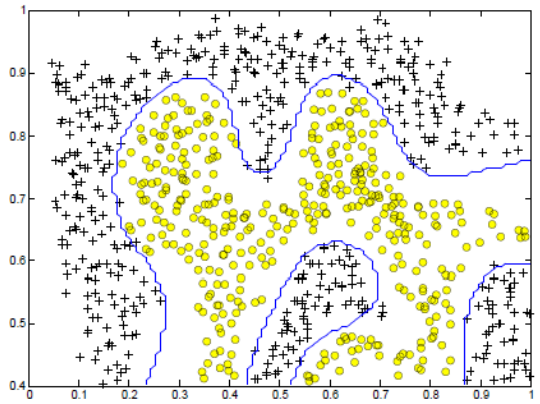
- Naïve Bayes
  - A type of Probabilistic Modeling – for classification.
  - Not many classification targets.

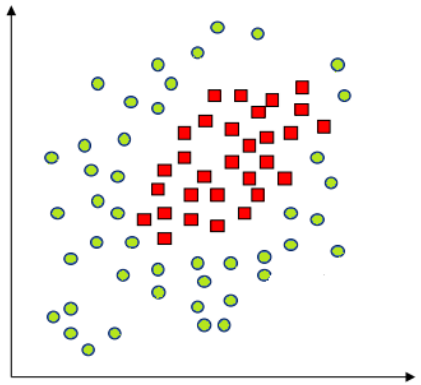


Algorithms of AI

- Kernel Methods
  - SVM (Support Vector Machines) – Creating decision boundary
  - Brittle on large datasets
  - Manual determination of boundary, separation is created by the system.







kernel

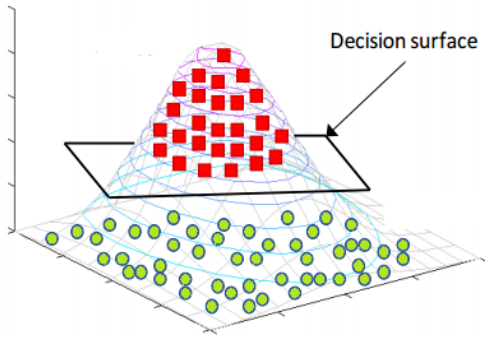
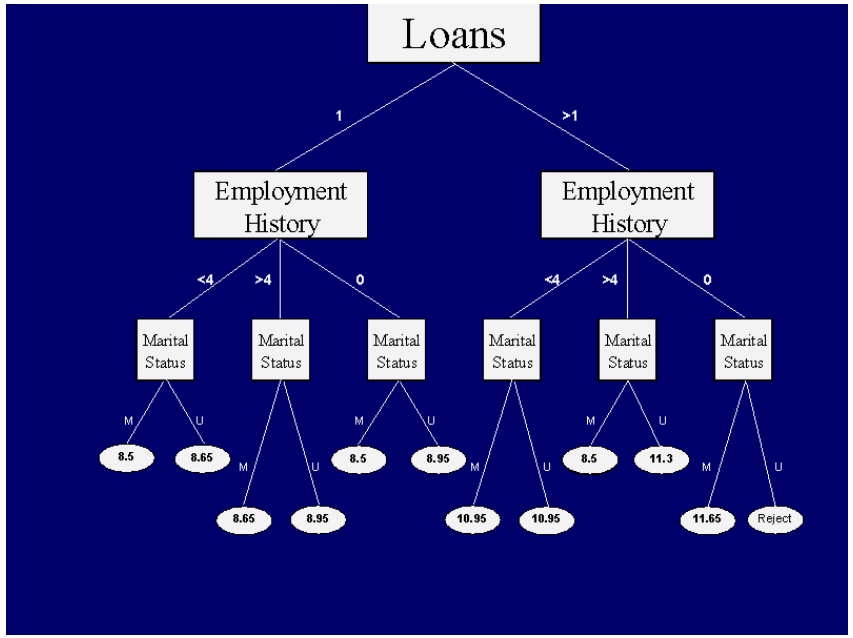
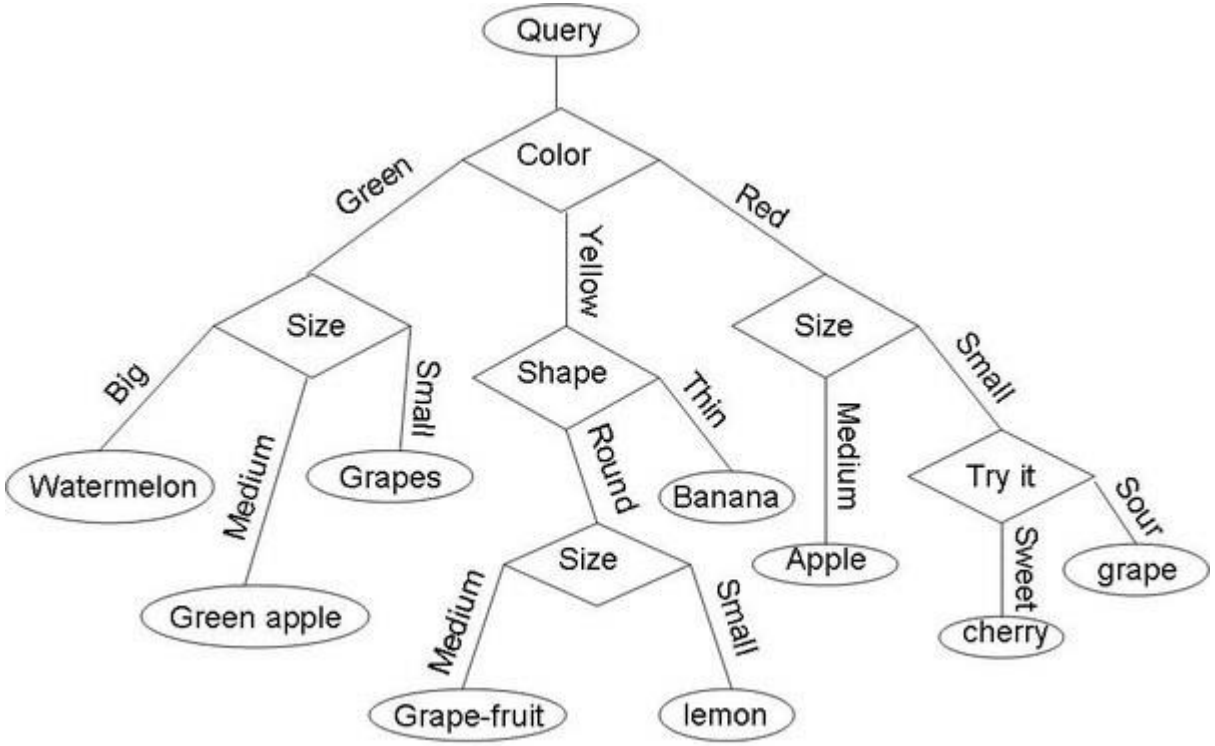


Figure 5: SVM (Gaussian Kernel) Decision Boundary (Example Dataset 2)

Algorithms of AI

- Decision Trees, Random Forests, Gradient Boosting Machines
  - Shallow Learning (not many aspects)
  - Mostly rules are pre-determined





Algorithms of AI

- Neural Networks
  - Noticed in 2010.
  - ImageNet Challenge – 2012 – Classify high resolution images into 1000 categories by learning from 1.4 Million images!
  - Machine Learning Accuracy – 74.3%.
  - Neural Networks Accuracy – 83.6%.
  - Neural Networks Accuracy in 2015 – 96.4%.

Why Deep Learning / Deep Nets?

- Better Performance.
- Feature Engineering Elimination (determination of layers of representation).
- Example – Cat – Ears, Nose, Skin, Fur, Color, Texture, Size, Shape etc.
- Instead – look and learn – all at once.

Hardware Evolution

- CPU
- GPU – 350 times a CPU
- TPU – 10 times a GPU
- In both computation and energy efficiency
- NVIDIA Titan X, a gaming GPU, can deliver a peak of 6.6 TLOPS in single precision: that is, 6.6 trillion float32 operations per second.

Data Evolution

- Big Data
- Volume
- Veracity
- Velocity
- Example on how much data is generated today.

AI Evolution

- 2011 – USD 19 Million.
- 2013 – USD 394 Million.
- Very fast growing.
- Earlier – C++ and CUDA expertise.
- Today – Simple R and Python scripts.

Understanding Neural Networks

- Jargons

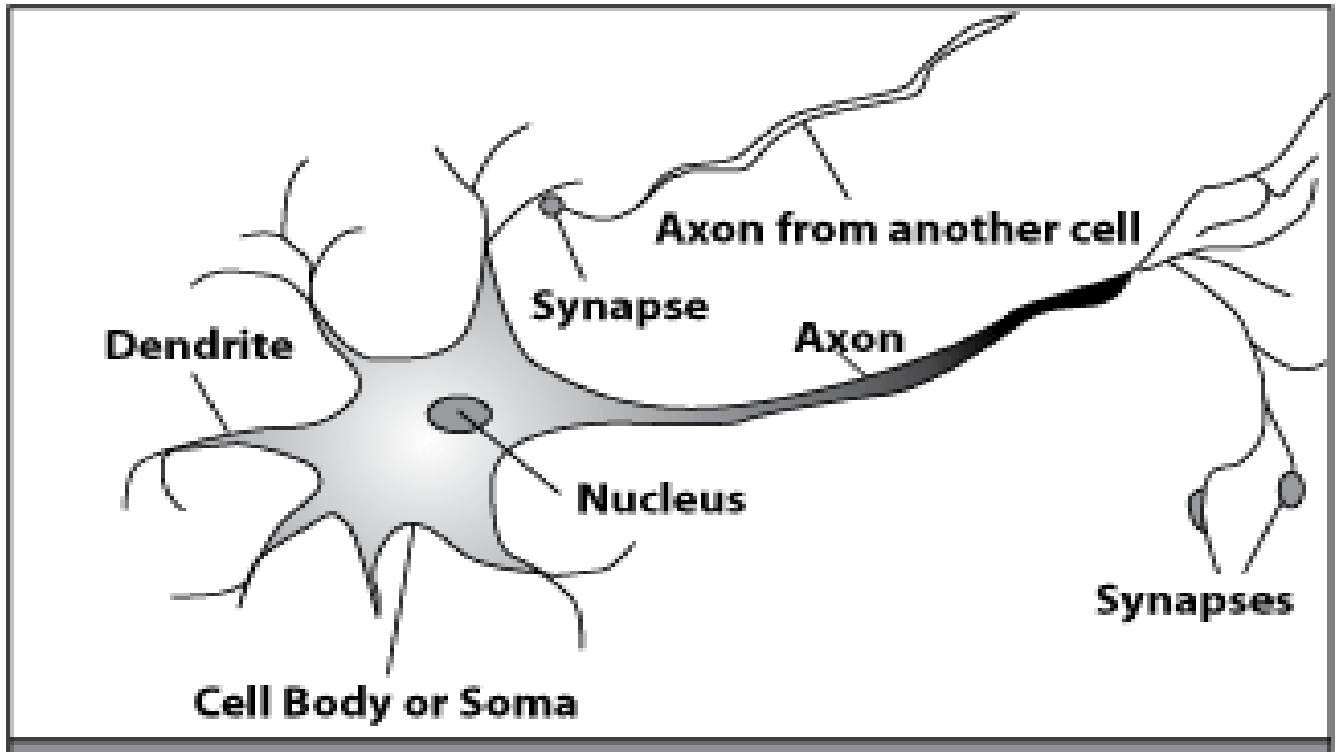
  - Inputs
  - Weights
  - Biases
  - Perceptrons
  - Hidden Layers
  - Activation Functions
  - Back Propagation
  - Output

How do our brains work?

- The Brain is A massively parallel information processing system.
- Our brains are a huge network of processing elements. A typical brain contains a network of 10 billion neurons.

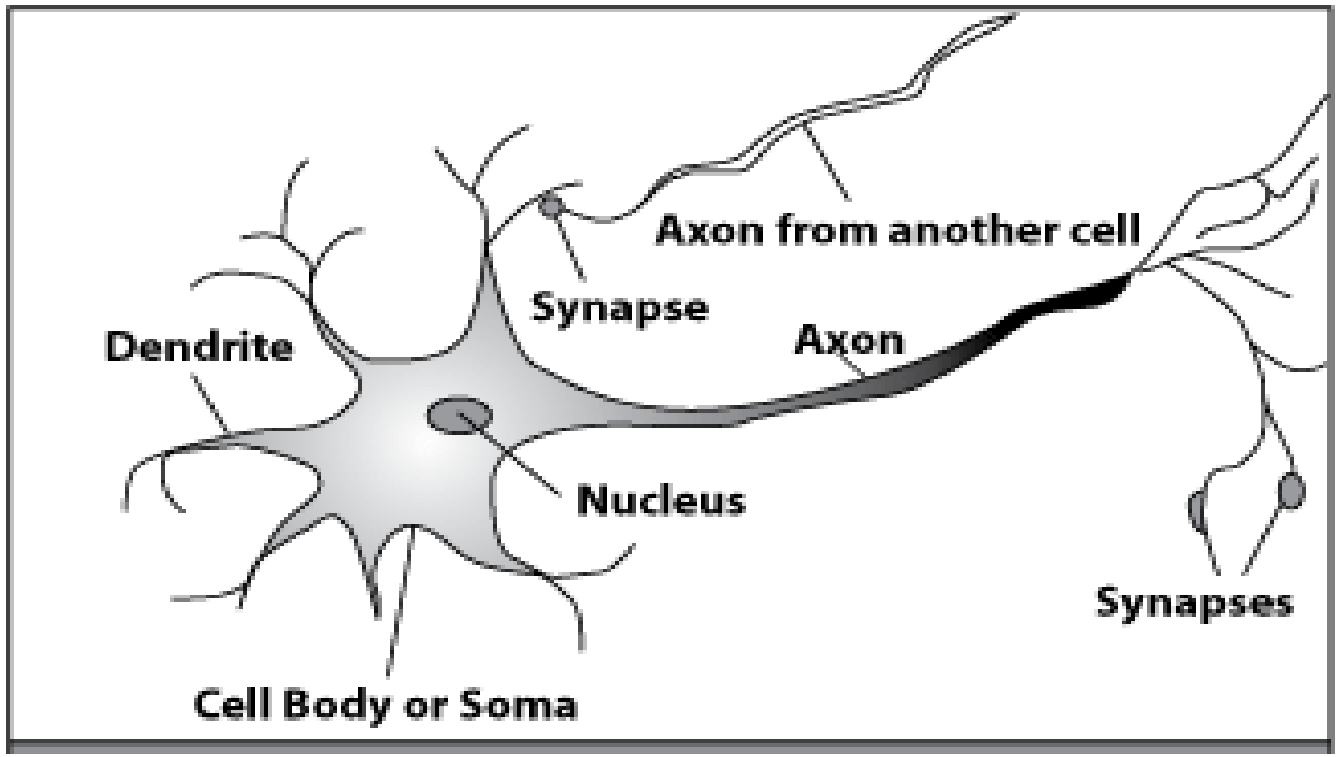
How do our brains work?

- A processing element



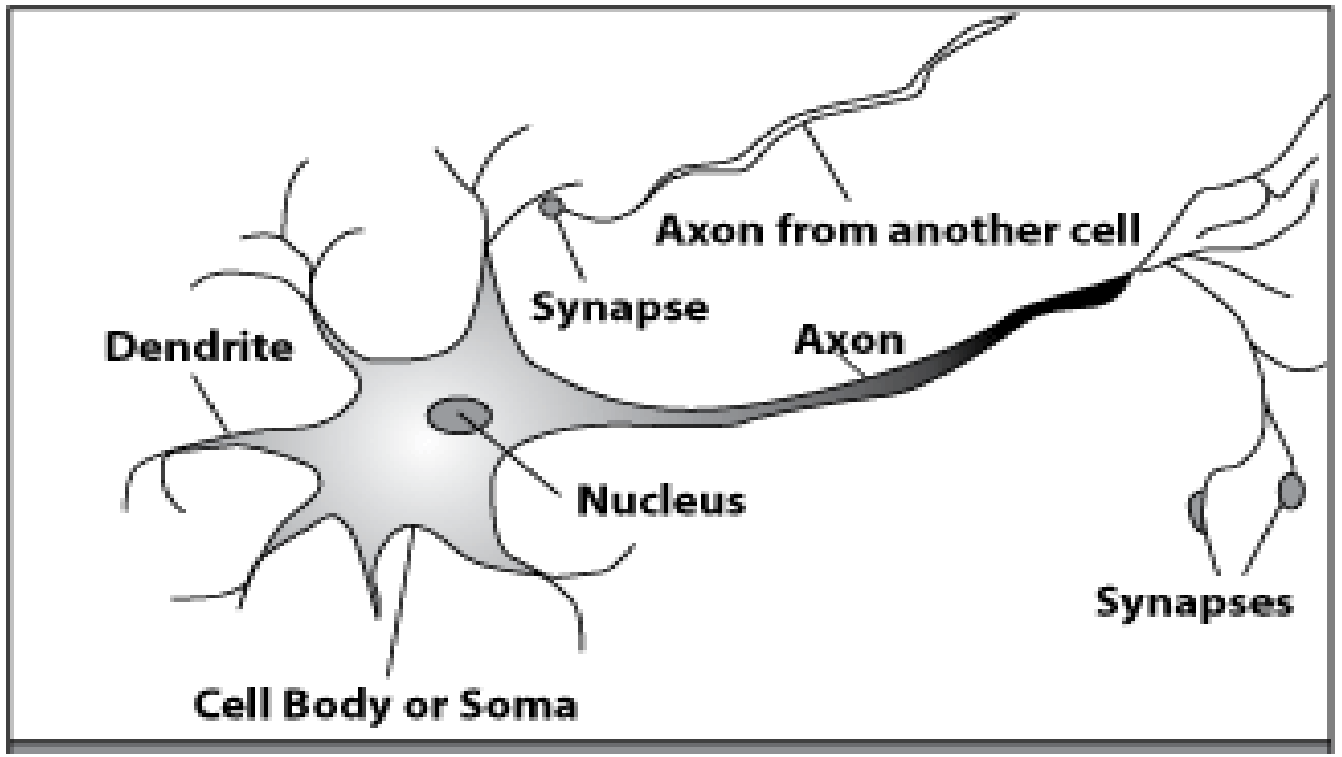
How do our brains work?

- A neuron is connected to other neurons through about 10,000 synapses



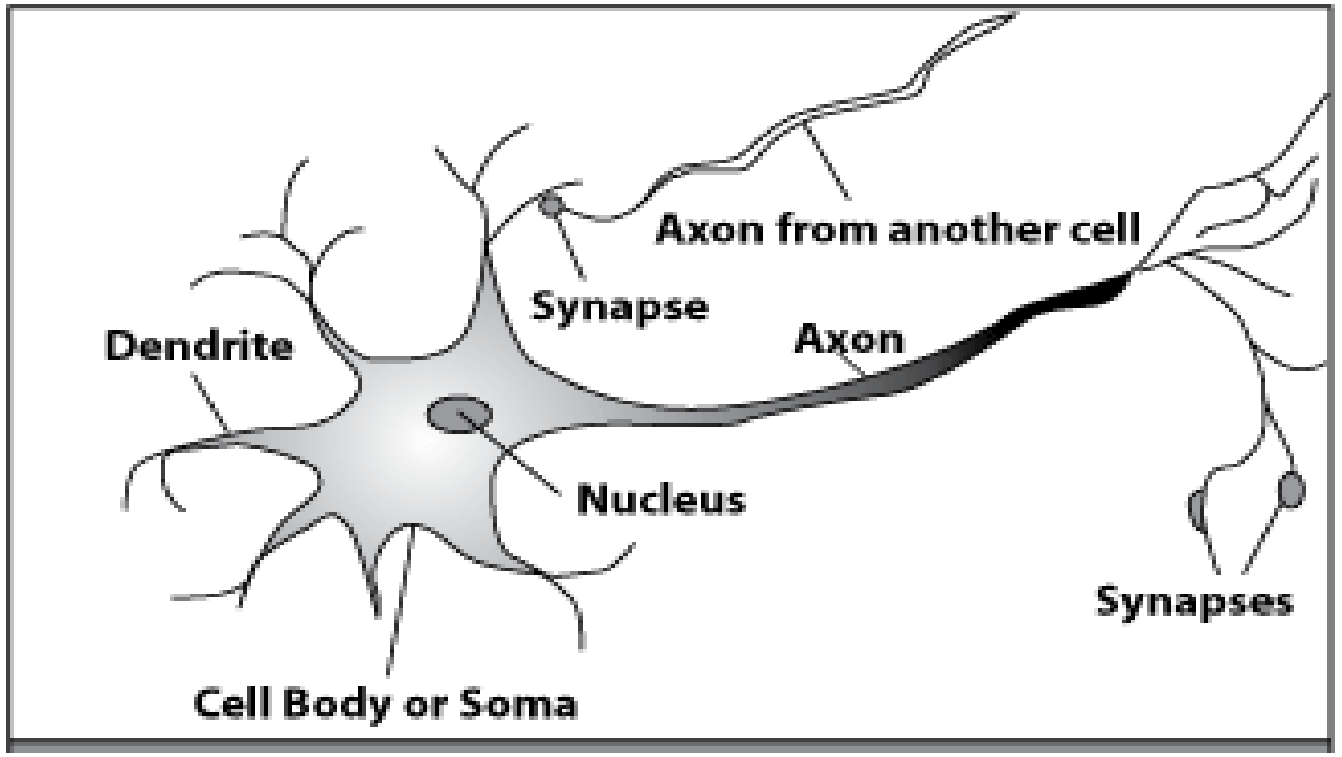
How do our brains work?

- A neuron receives input from other neurons. Inputs are combined.



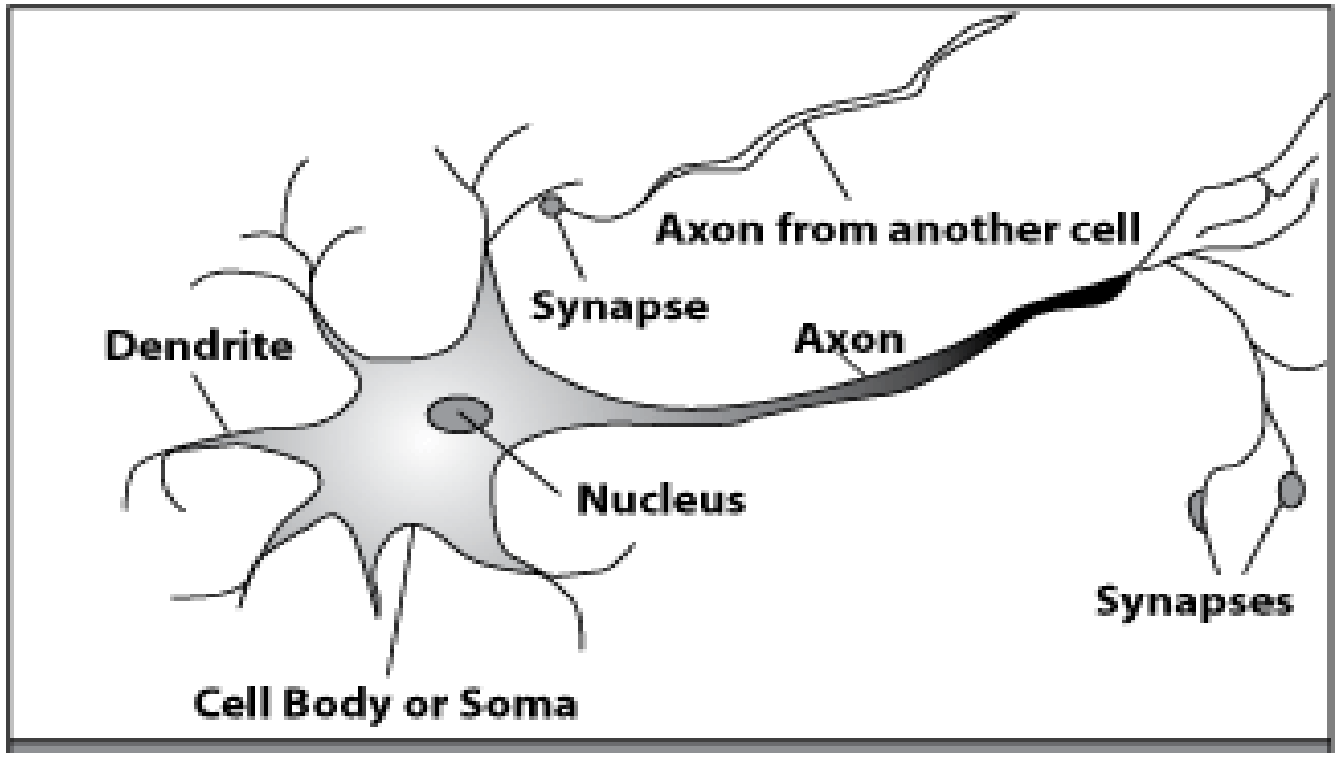
How do our brains work?

- Once input exceeds a critical level, the neuron discharges a spike - an electrical pulse that travels from the body, down the axon, to the next neuron(s)



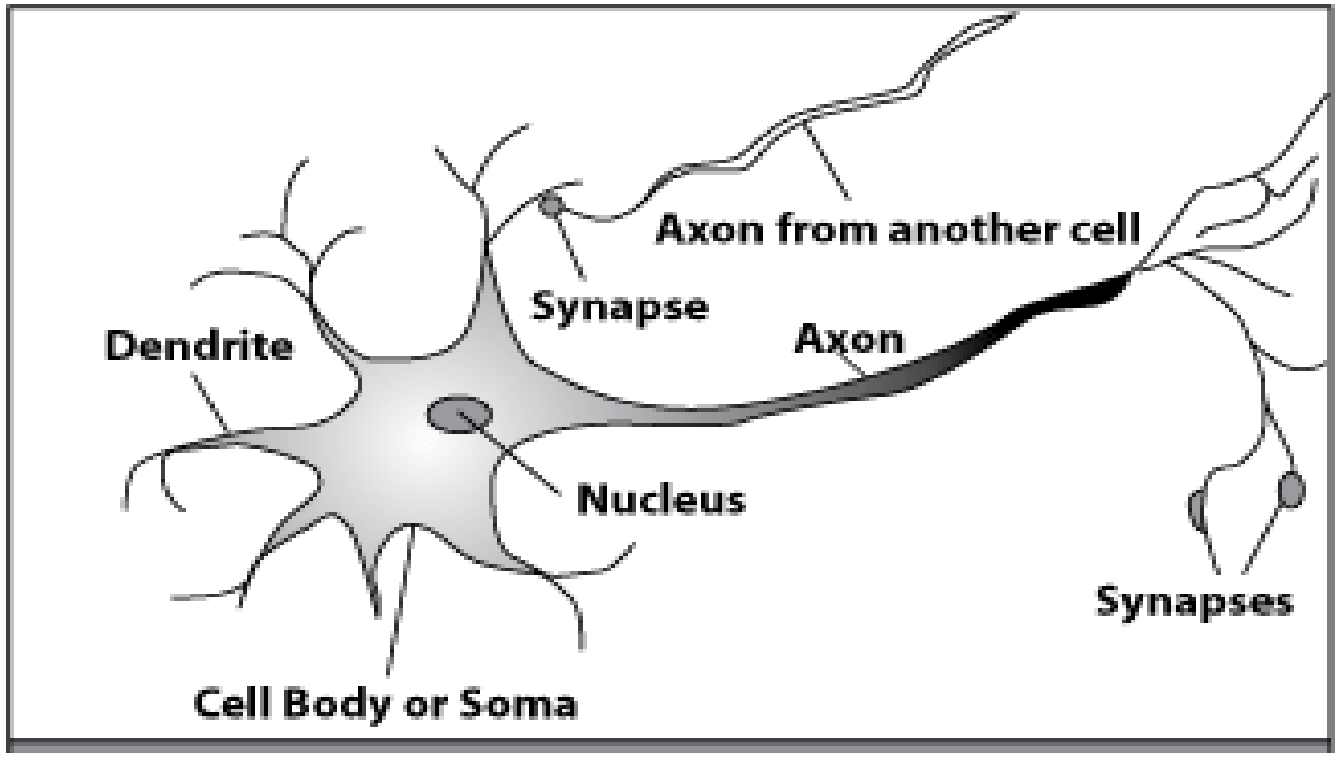
How do our brains work?

- The axon endings almost touch the dendrites or cell body of the next neuron.



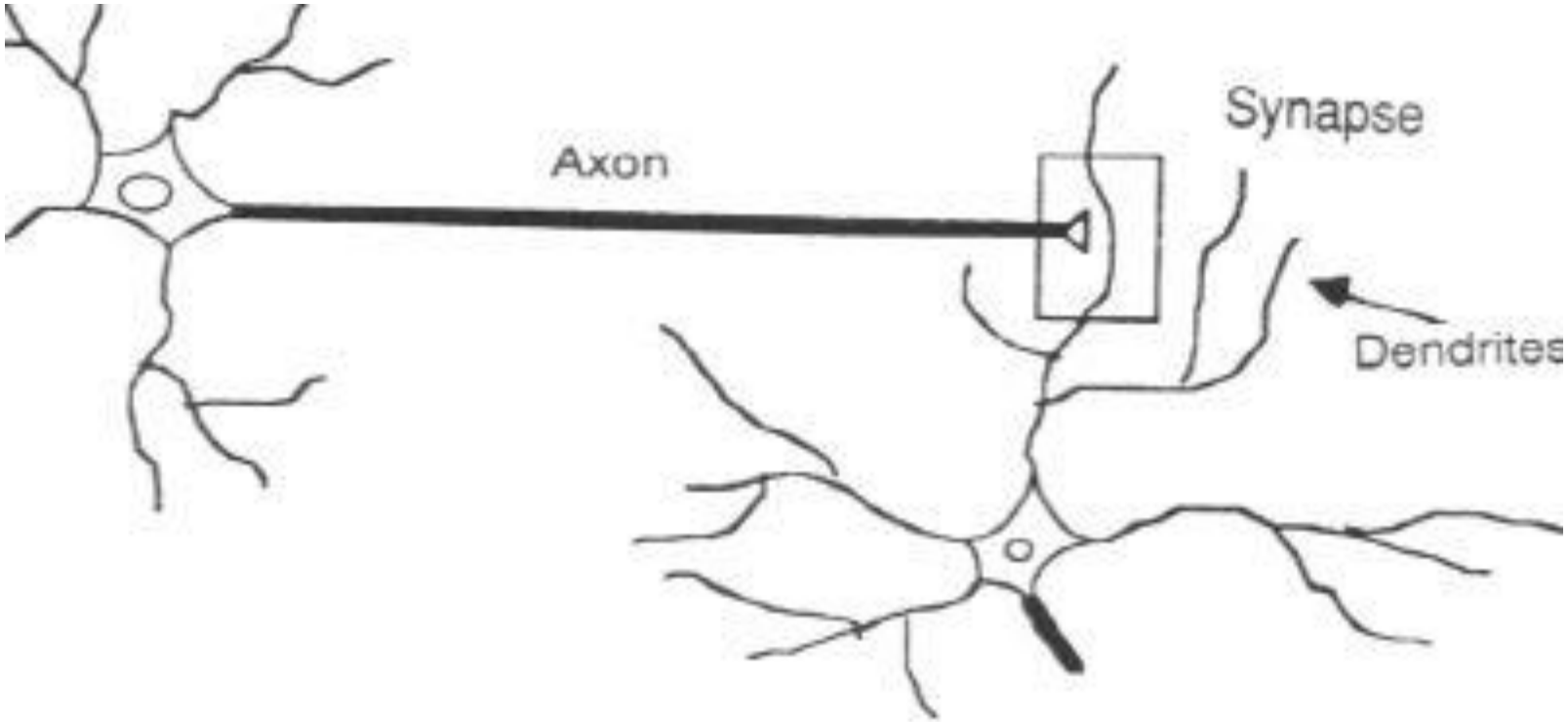
How do our brains work?

- The Neurotransmitters are chemicals which are released from the first neuron and which bind to the Second creating a spike.



How do our brains work?

- This link is called a Synapse.



Hawkins' Quotes

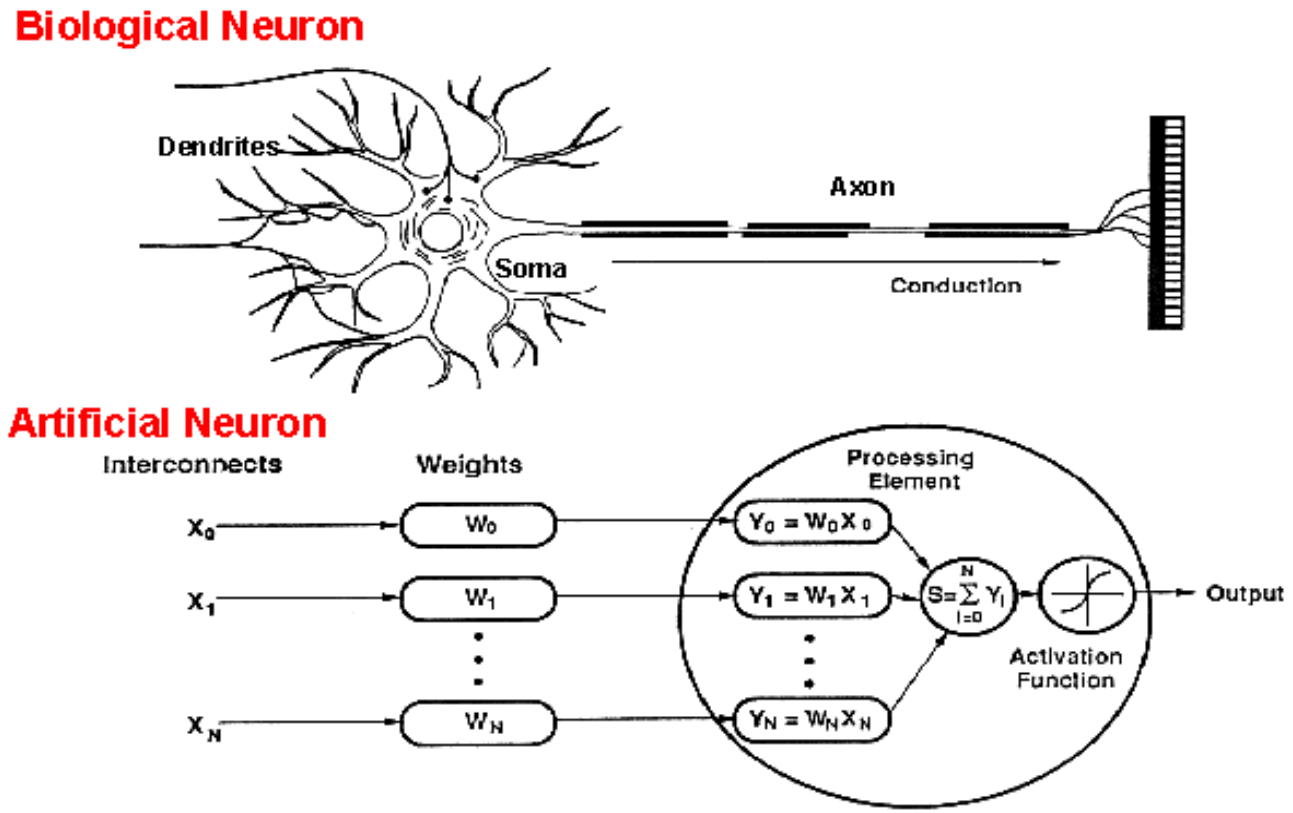
"Your brain constantly makes predictions about the very fabric of the world we live in, and it does so in a parallel fashion. It will just as readily detect an odd texture, a misshapen nose, or an unusual motion. It isn't obvious how pervasive these mostly unconscious predictions are, which is perhaps why we missed their importance."

Hawkins' Quotes

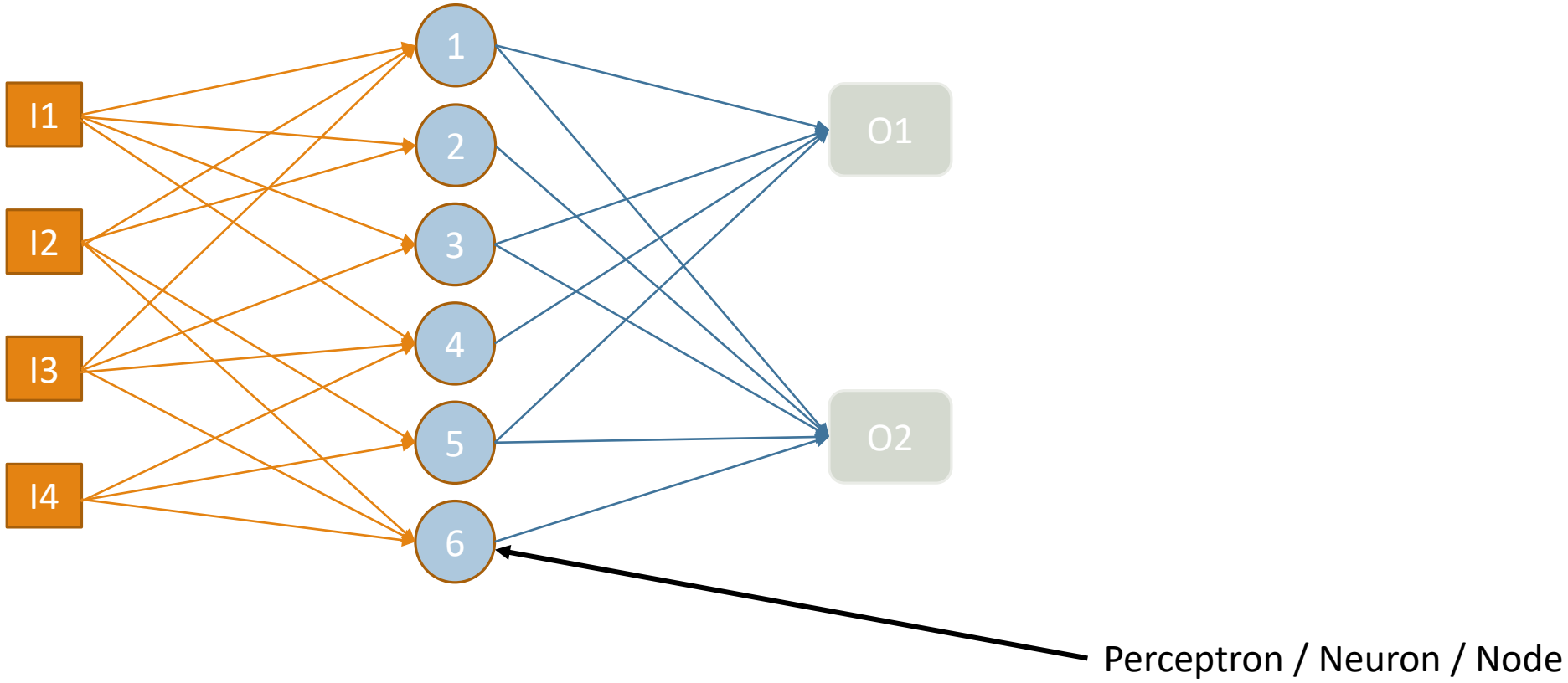
"Suppose when you are out, I sneak over to your home and change something about your door. It could be almost anything. I could move the knob over by and inch, change a round knob into a thumb latch, or turn it from brass to chrome….When you come home that day and attempt to open the door, you will quickly detect that something is wrong."

How do ANNs work?

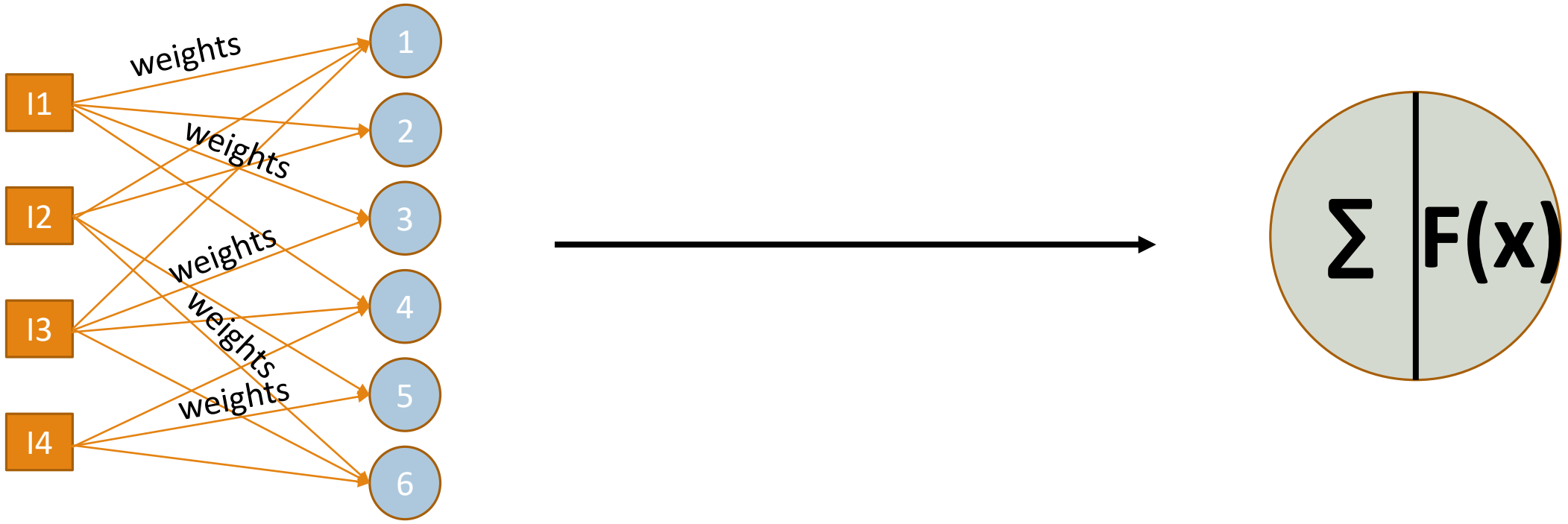
- Imitation of human neuron.



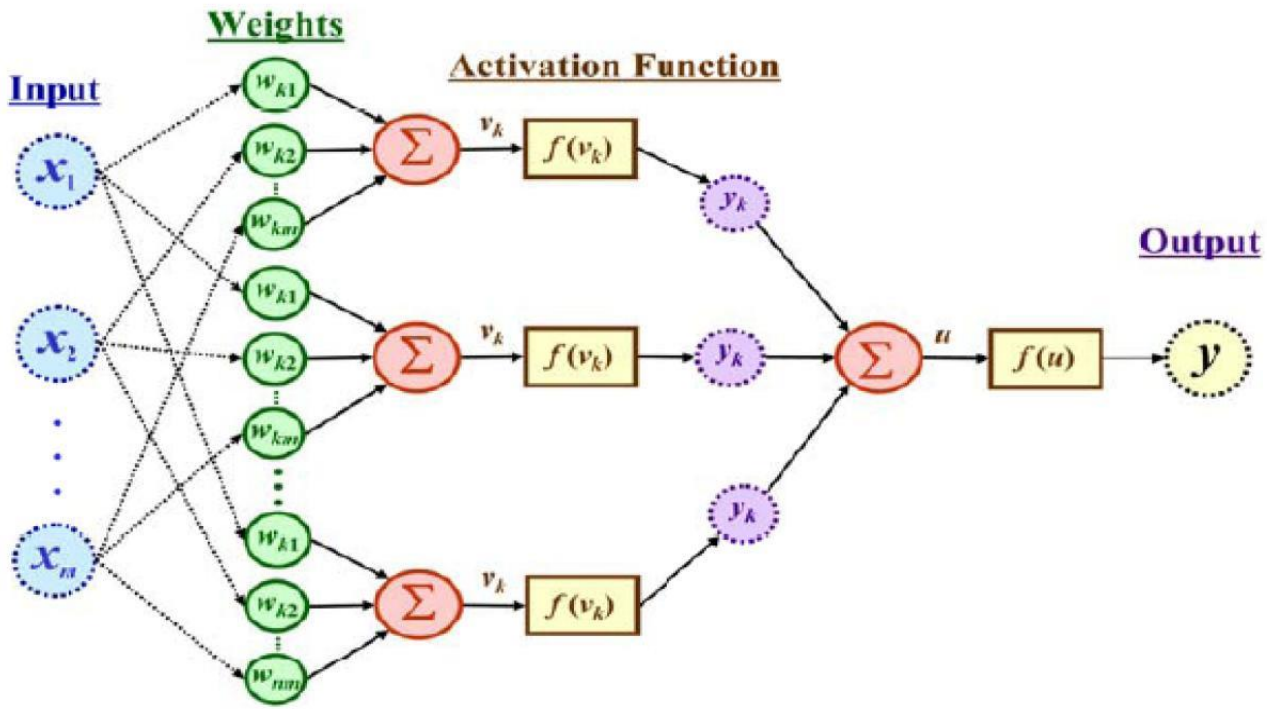
How does an ANN look like?



Inside an ANN



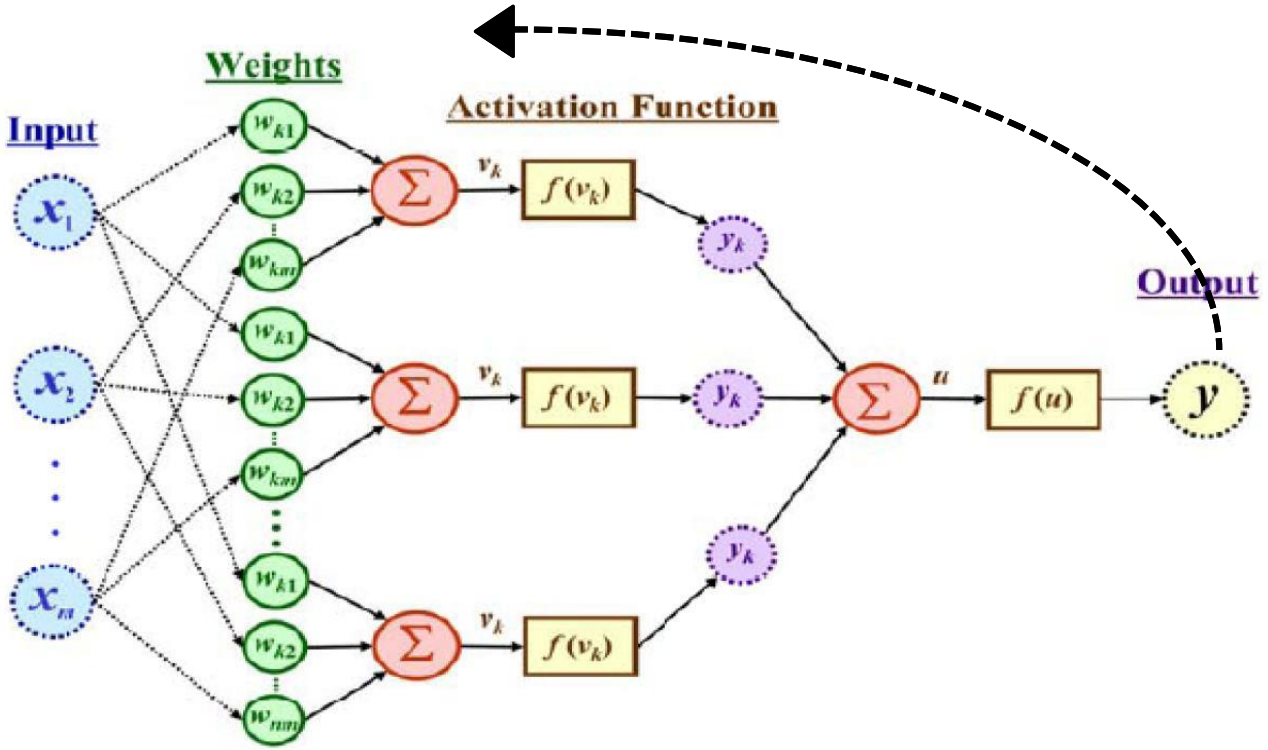
Inside an ANN



The simple ANN Equation

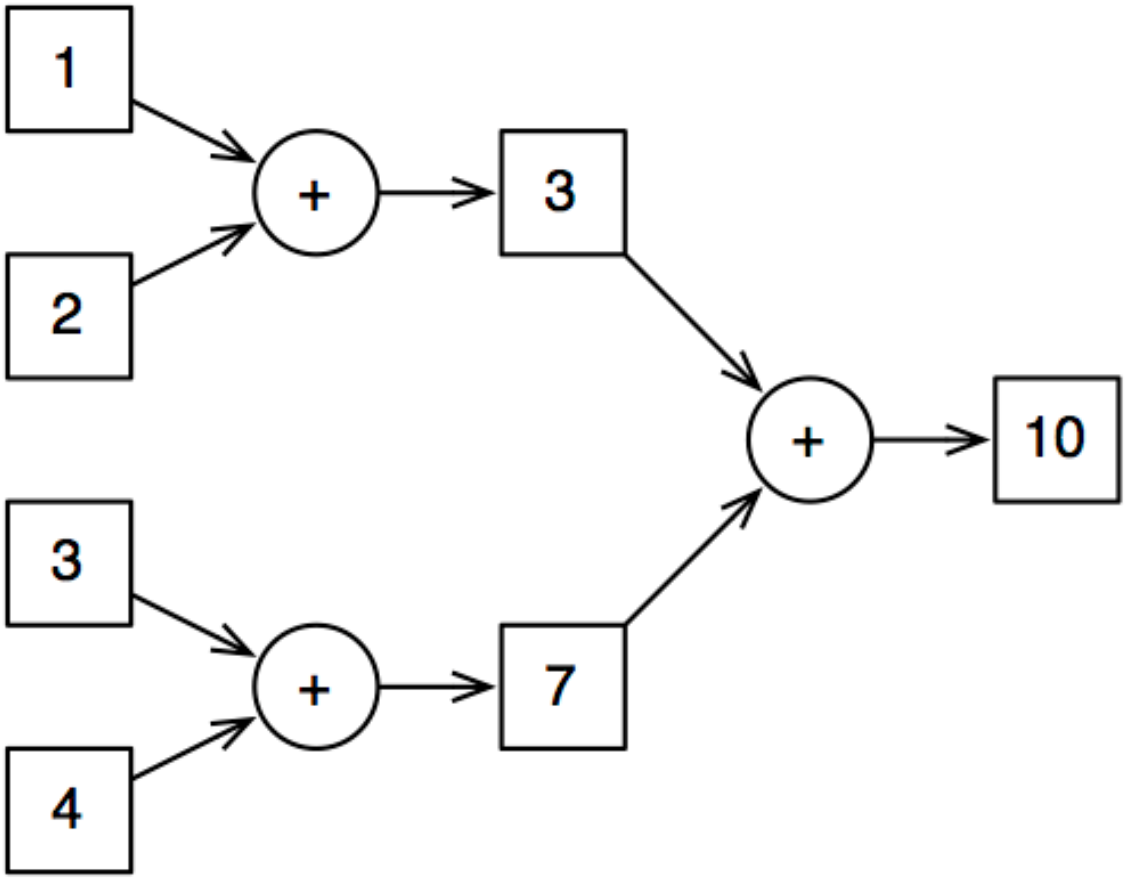
- Input X Weights = Output

Inside an ANN

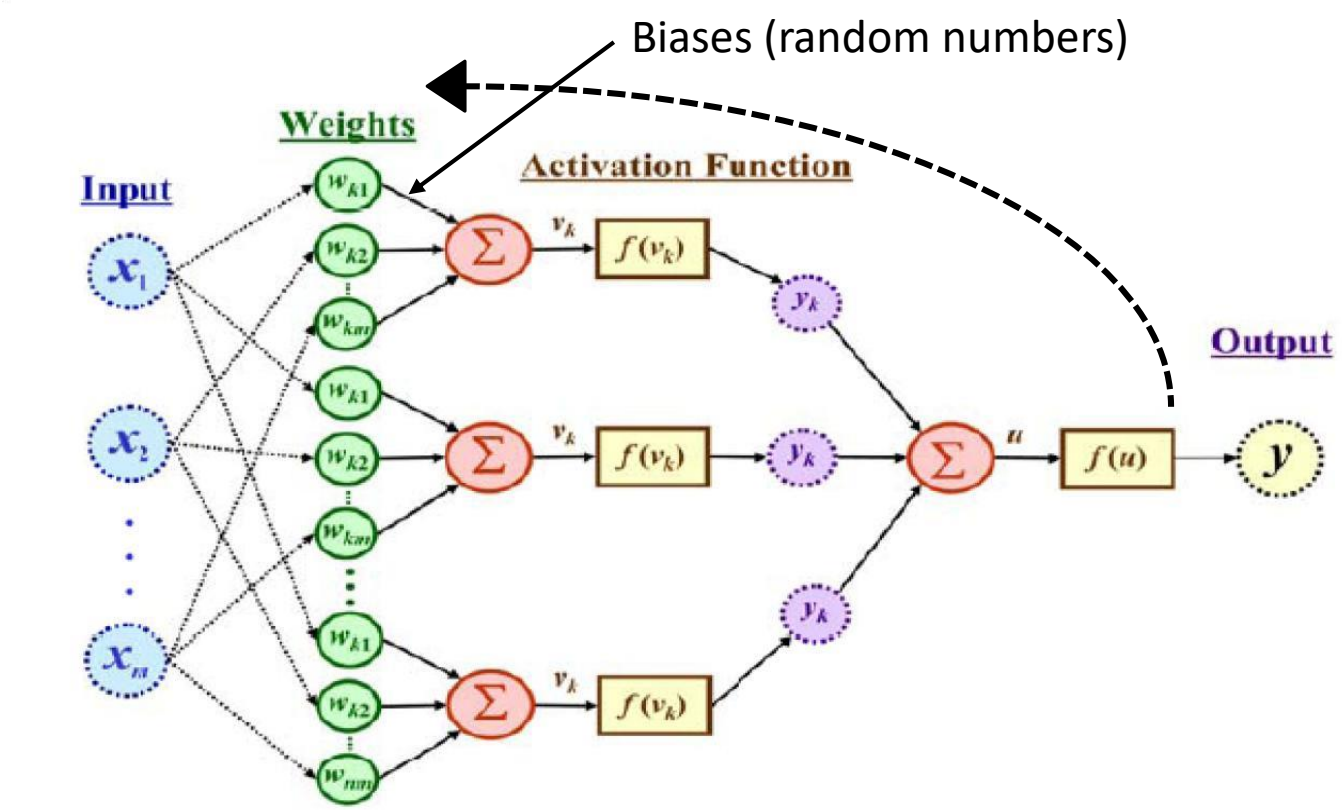


At each step, it checks if the desired output is achieved. If not …

Inside an ANN (backpropagation)



Inside an ANN

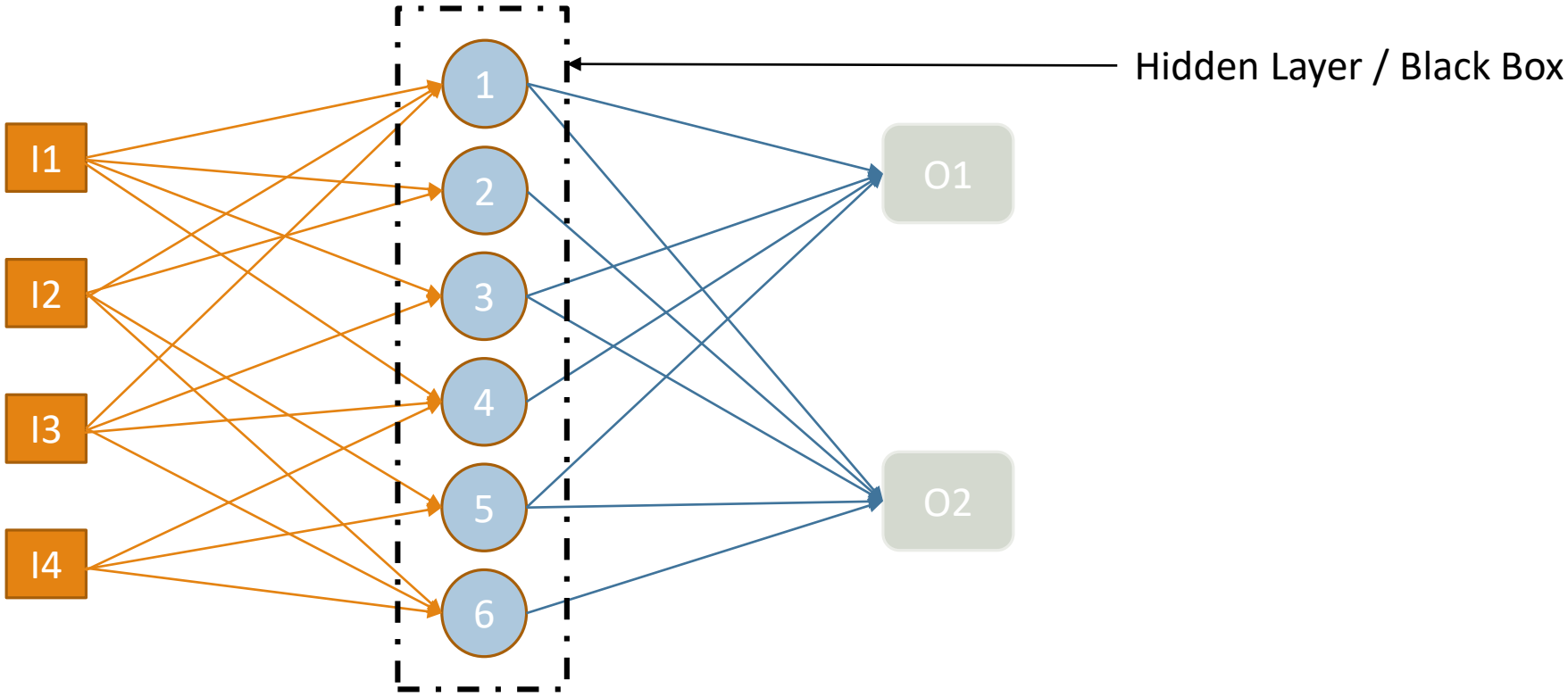


Sometimes weights can be 0. But maybe the input is important to get the desired output. Then what?

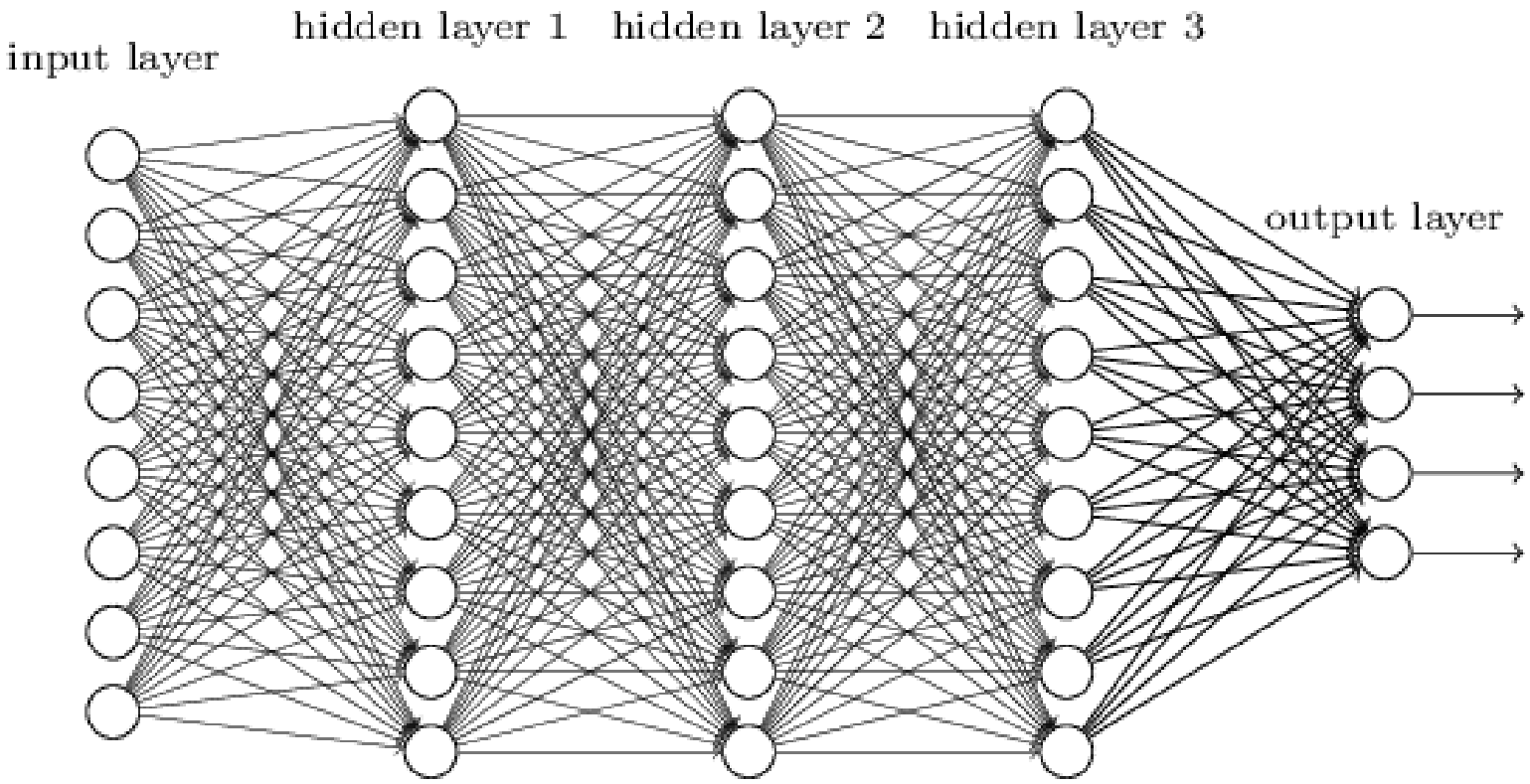
The simple ANN Equation … perfected

- Input X Weights + Biases = Output

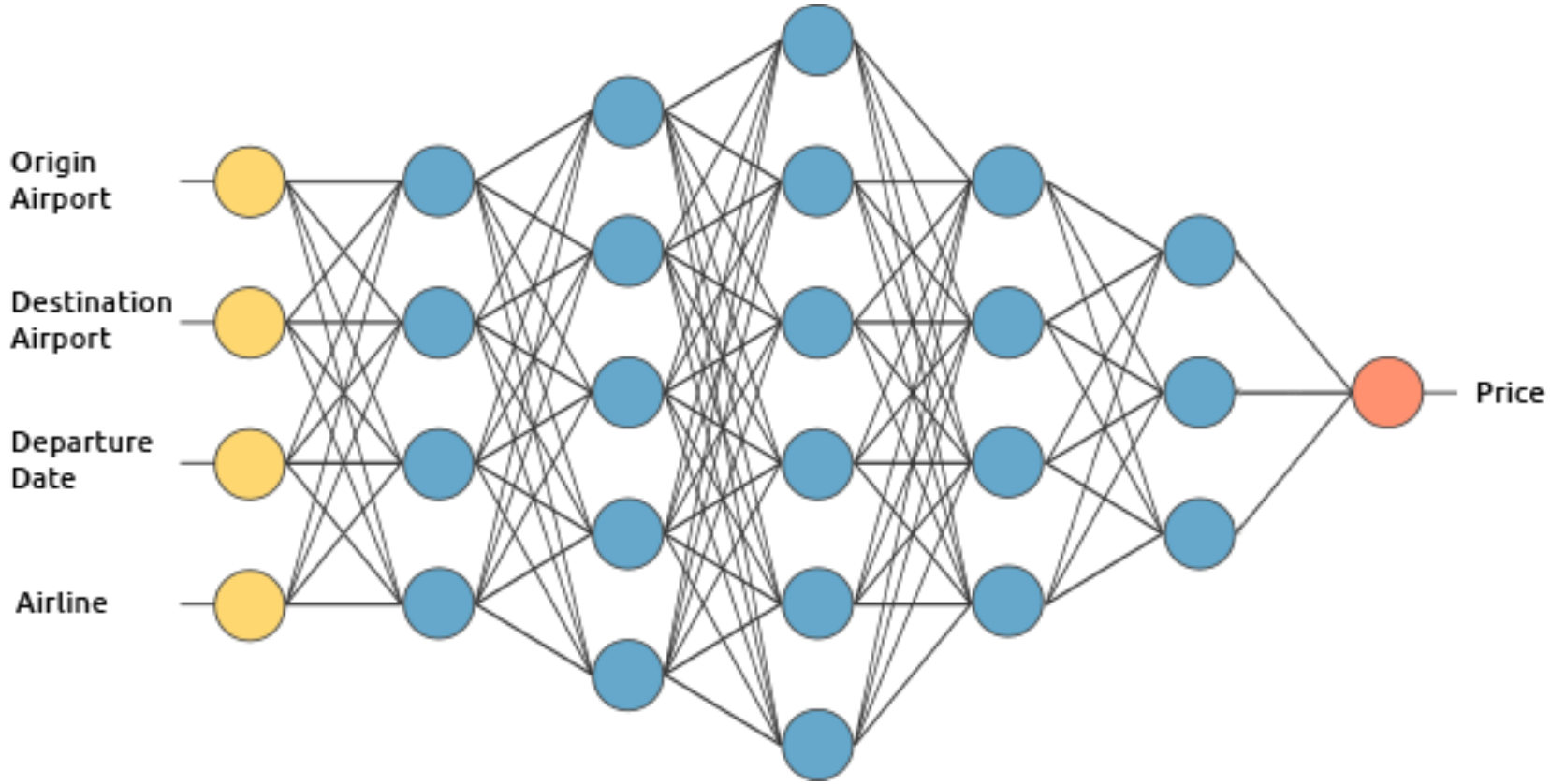
Hidden Layers



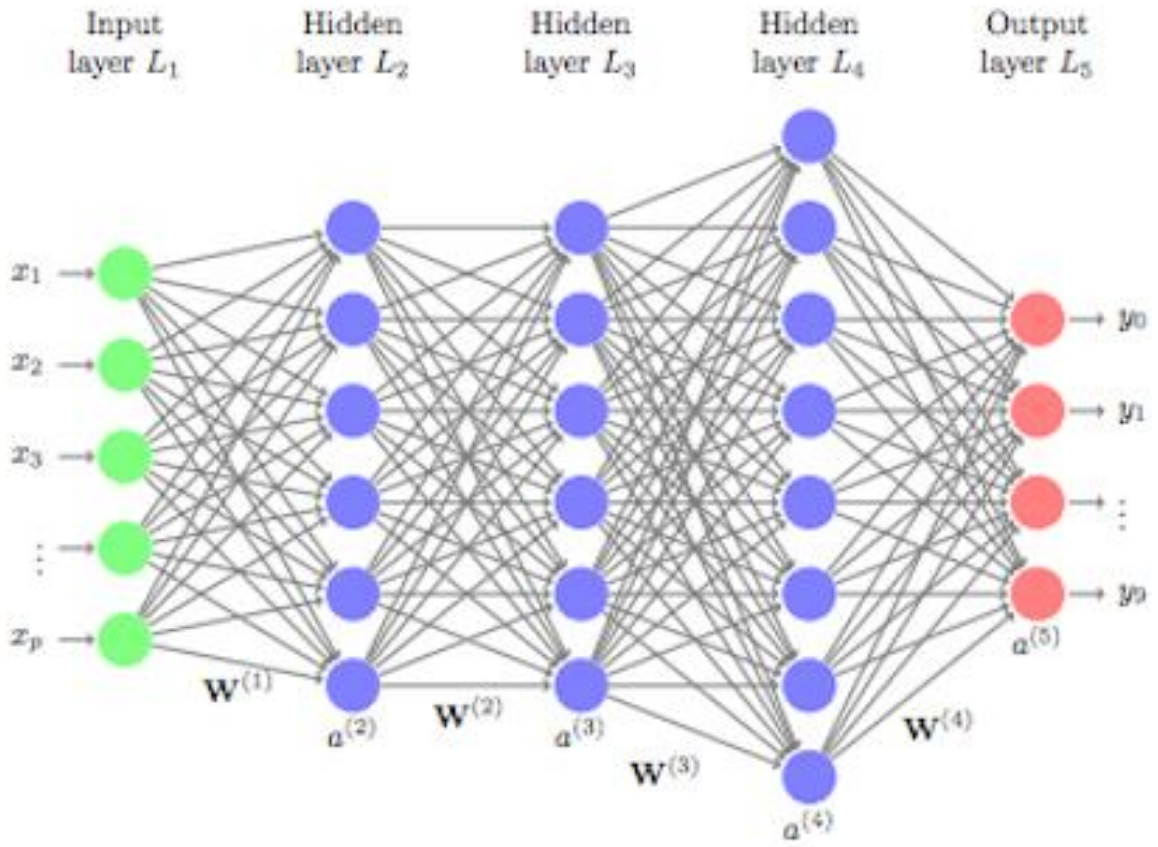
Hidden Layers



Hidden Layers



Hidden Layers



Key functions of any NN

- Activation Function – to get as close to the output as possible – mathematical equation
- Cost or Loss Function – minimizing the loss or penalty
- Optimization Function - to identify best representation of the data – pattern

Hyperparameters

- Number of nodes
- Epochs (1 feed forward + 1 backpropagation)
- Batch Size
- Cross Entropy
- Number of Hidden Layers

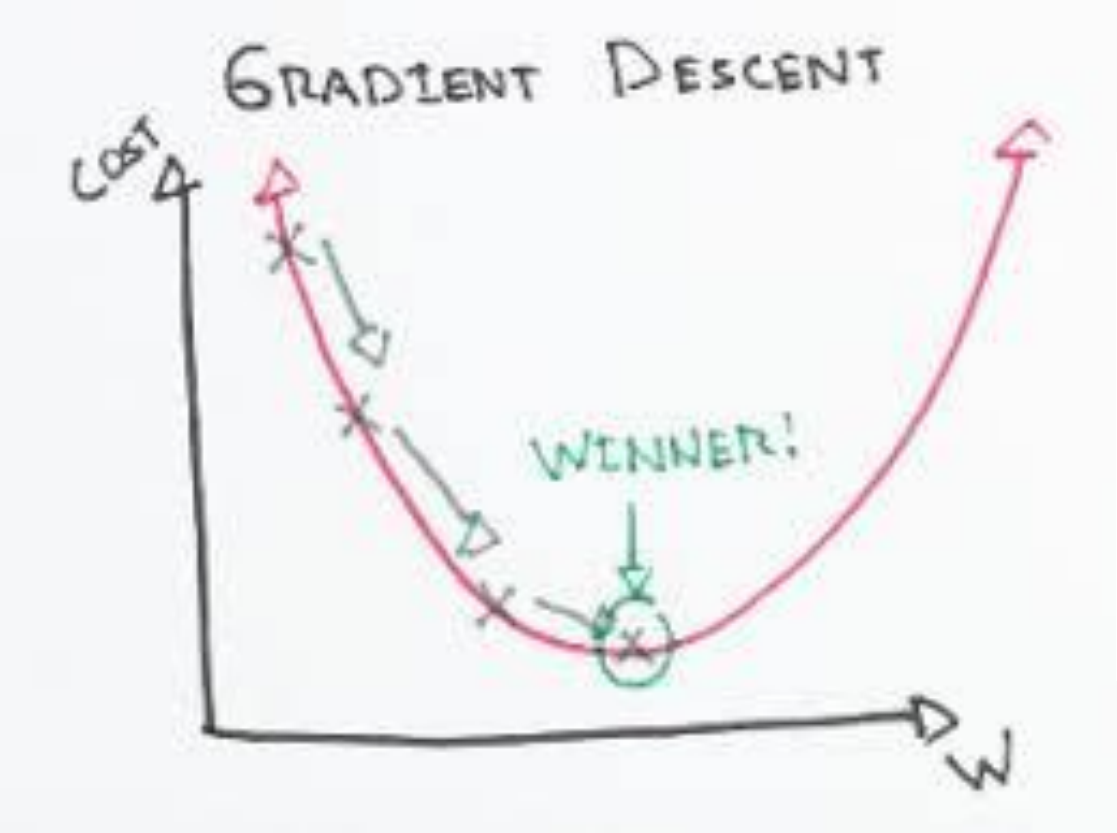
Let’s look at a sample code

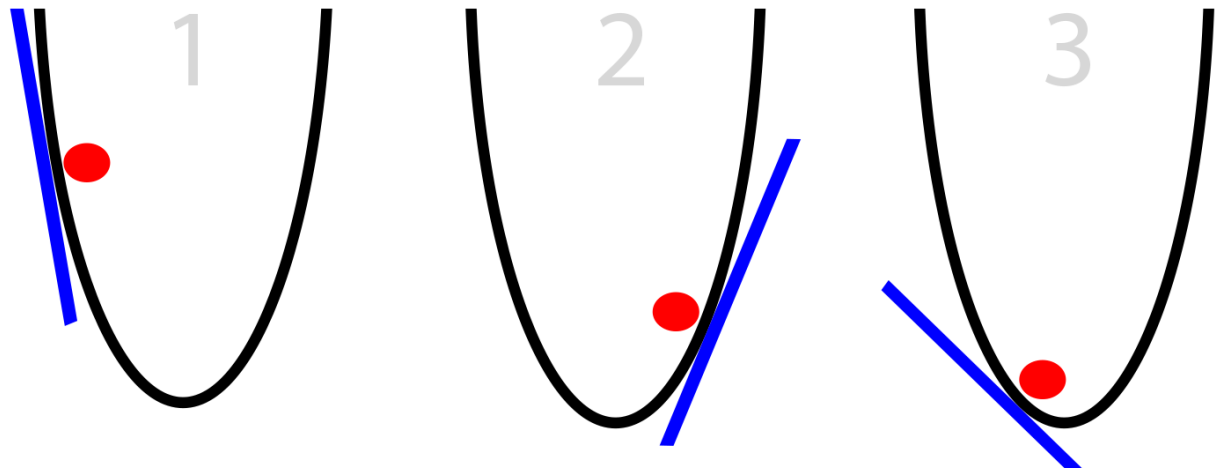
Gradient Descent & Overfitting

- **Gradient descent** is a first-order iterative optimization algorithm for finding the minimum of a function.
- To find a local minimum of a function using **gradient descent**, one takes steps proportional to the negative of the **gradient** (or approximate **gradient**) of the function at the current point.

Gradient Descent & Overfitting

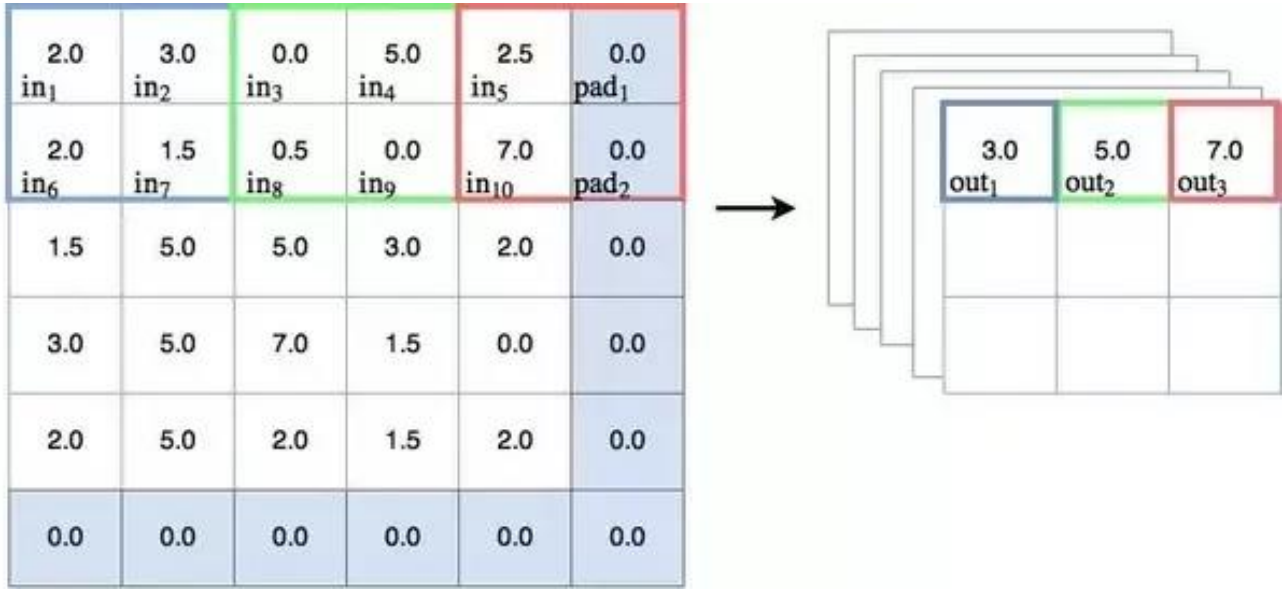
Gradient Descent & Overfitting





Maxpooling

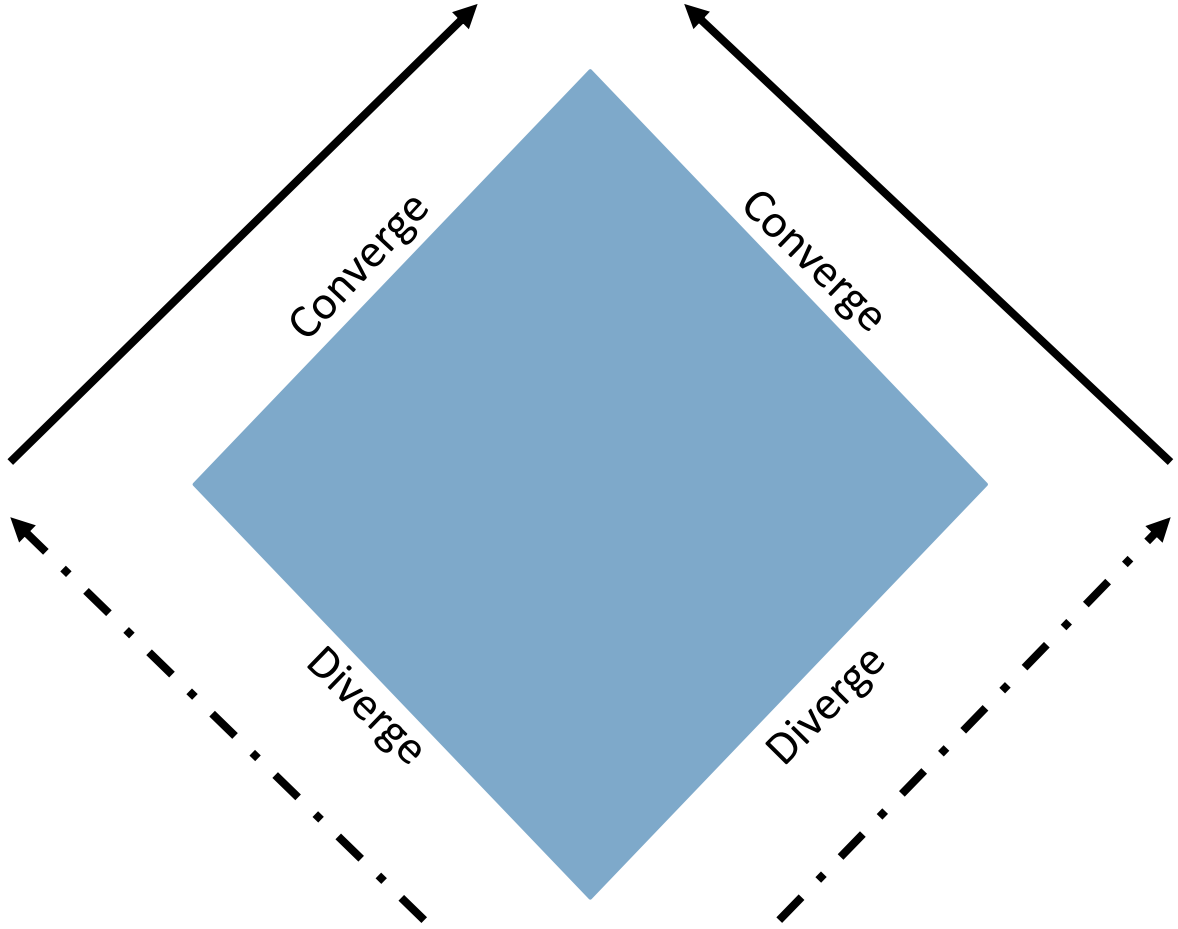
- Sample based discretization process
- Objective is to down sample the representation by reducing its dimensionality
- Avoids over fitting
- Reduces computational cost (time)



Maxpooling

0	0	0	0	0
0	0	0	0	0
0	-1	1	0	0
0	0	0	0	0
0	0	0	0	0

The Diamond Model of Learning



Activation Functions

- They add non-linear properties to the model to get the desired output.
- Simply putting, activation functions help the networks understand patterns better.

- So after Input X Weights + Biases – the activation function is applied.

- Why non-linear properties?
- We need a Neural Network Model to learn and represent almost anything and any arbitrary complex function which maps inputs to outputs.
- Neural-Networks are considered ***Universal Function Approximators***. *It means that they can compute and learn any function at all.*

Neural Mantra

- Input times Weights,
- add Bias and then
- Activate!

Activation Functions - Types

- Sigmoid (Logistic)
  - Range between 0 and 1
  - Slow convergence
  - Vanishing gradient problem
- Tanh (Hyperbolic Tangent)
  - Range between -1 and 1
  - Easy optimization
  - Vanishing gradient problem

Used for binary classification of output layer.

- ReLu (Rectified Linear Units)
  - Higher convergence rate
  - Resolves vanishing gradient problem

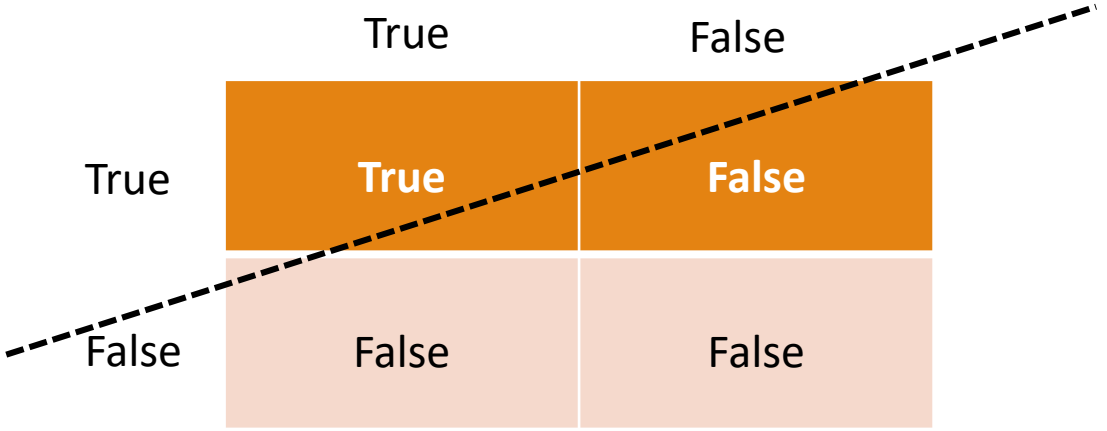
→ Ideal for hidden layers.

Activation Functions - Types

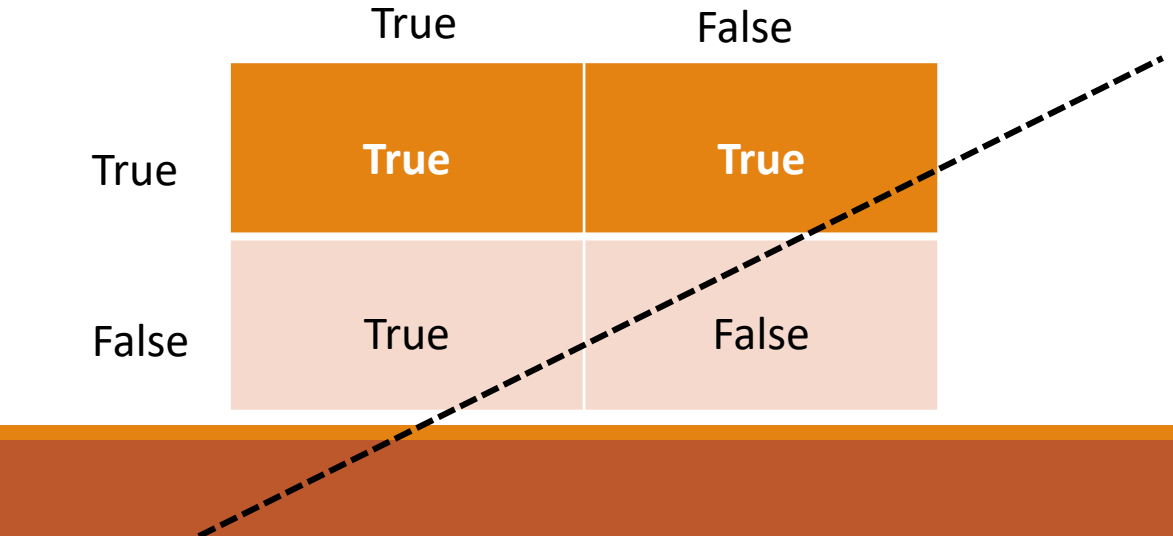
- Softmax Function
  - Deals well with multi-classification problem
  - Calculates probabilities between 0 and 1
  - Easy to interpret

Why MLP? Why not Single Perceptron?

- AND (A&&B)

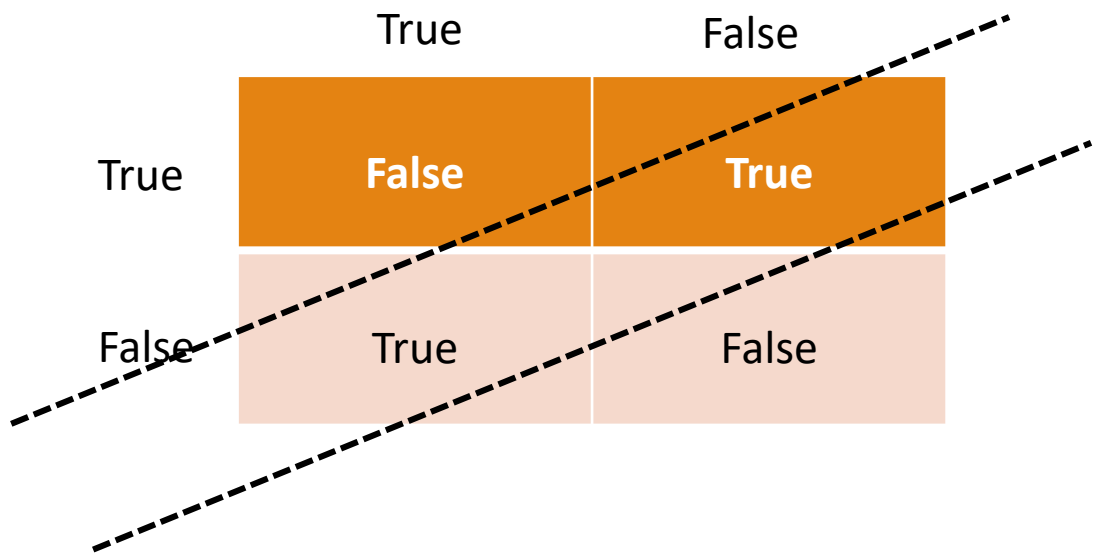


- OR (A||B)

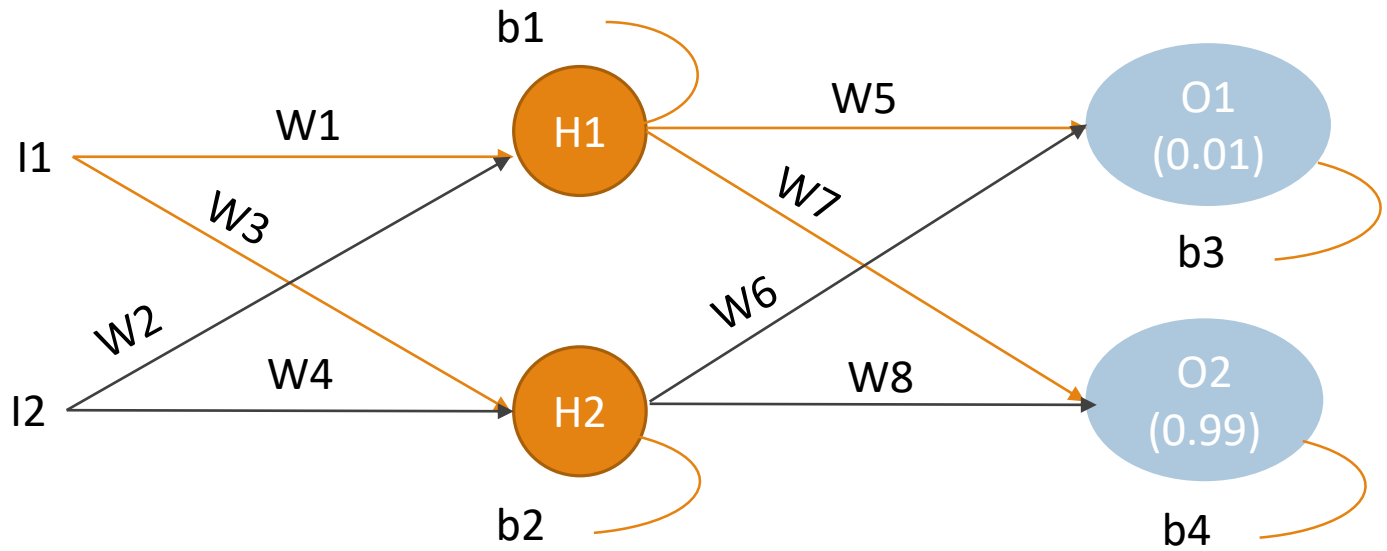


Why MLP? Why not Single Perceptron?

- XOR (Exclusive OR)



MLP Calculations (with backpropagation)

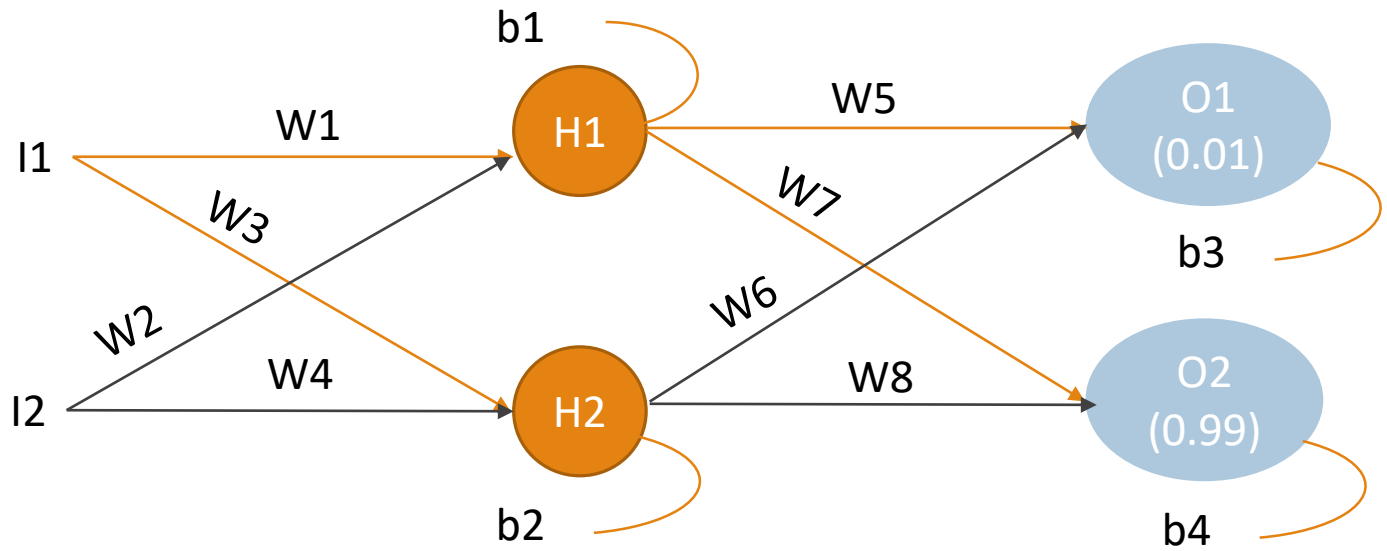


Variable	Value
I1	0.05
I2	0.10
B1,B2	0.35
B3,B4	0.60
W1	0.15
W2	0.20
W3	0.25
W4	0.30
W5	0.40
W6	0.45
W7	0.50
W8	0.55

H1 = I1*W1 + I2*W2 + b1 = 0.3775
H2 = I1*W3 + I2*W4 + b2 = 0.3925

Out H1 = sigmoid(H1) = 0.593269992
Out H2 = sigmoid(H2) = 0.596884378

MLP Calculations (with backpropagation)

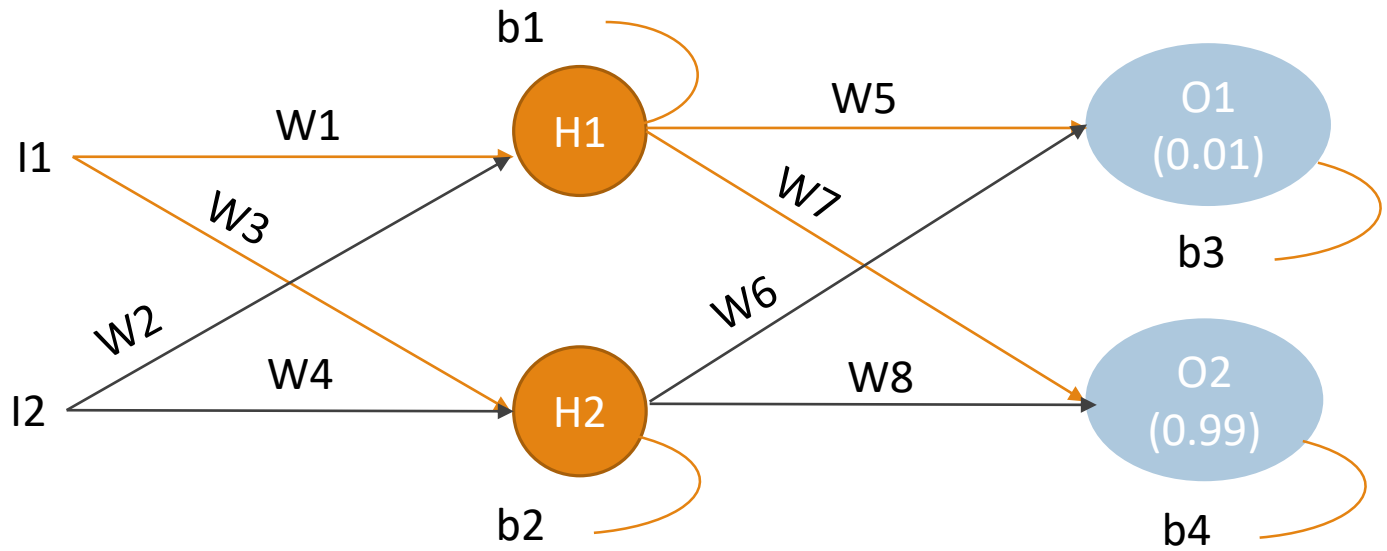


Variable	Value
I1	0.05
I2	0.10
B1,B2	0.35
B3,B4	0.60
W1	0.15
W2	0.20
W3	0.25
W4	0.30
W5	0.40
W6	0.45
W7	0.50
W8	0.55

O1 = Out H1*W5 + Out H2*W6 + b3 = 1.105905967
O2 = Out H1*W7 + Out H2*W8 + b4 = 1.2249214

Out O1 = sigmoid(O1) = 0.75136507
Out O2 = sigmoid(O2) = 0.77292846

MLP Calculations (with backpropagation)



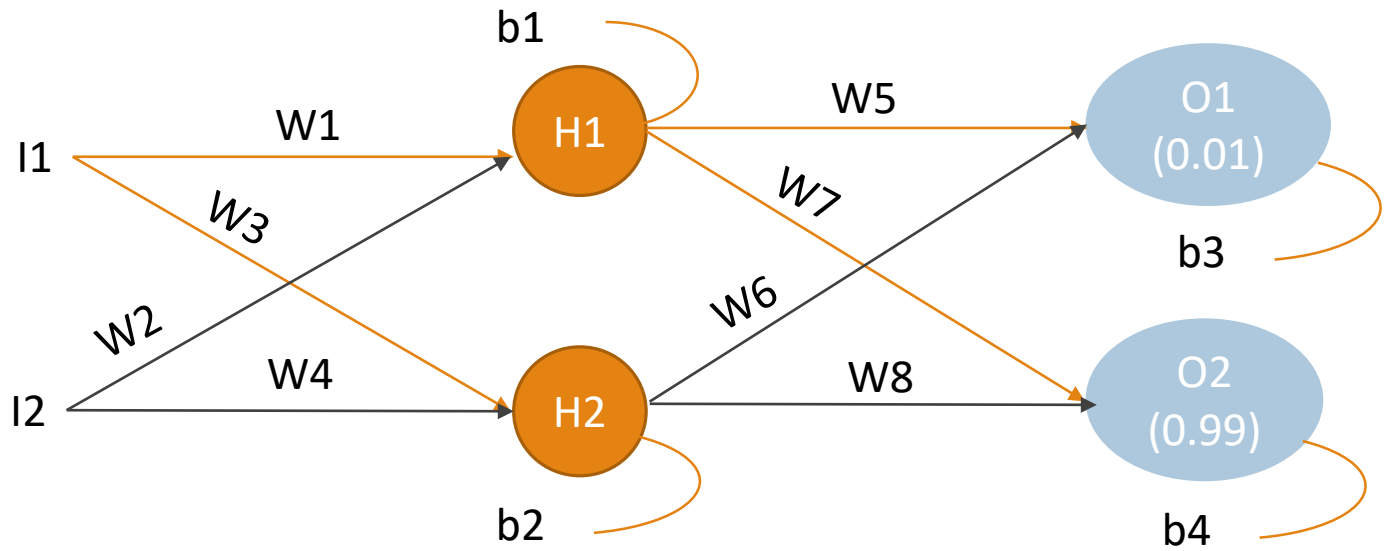
Variable	Value
I1	0.05
I2	0.10
B1,B2	0.35
B3,B4	0.60
W1	0.15
W2	0.20
W3	0.25
W4	0.30
W5	0.40
W6	0.45
W7	0.50
W8	0.55

$E1 = [(T1-O1)^2] / 2 = 0.274811083$

$E2 = [(T2-O2)^2] / 2 = 0.023560026$

$Etotal = E1 + E2 = 0.298371109$

MLP Calculations (with backpropagation)

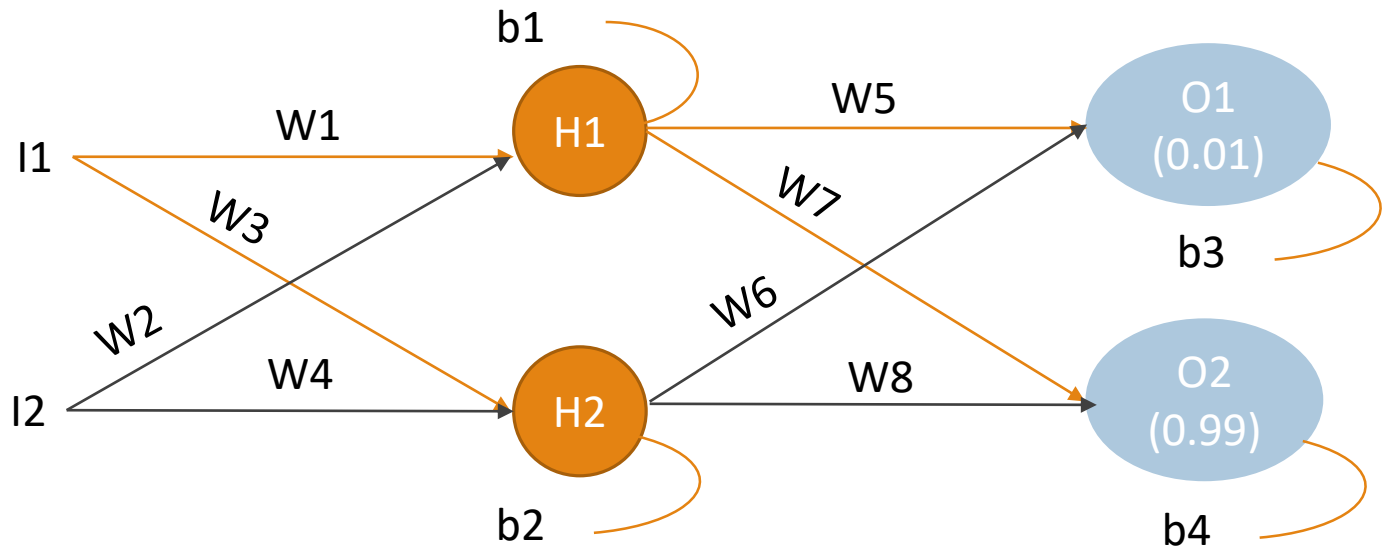


Variable	Value
I1	0.05
I2	0.10
B1,B2	0.35
B3,B4	0.60
W1	0.15
W2	0.20
W3	0.25
W4	0.30
W5	0.40
W6	0.45
W7	0.50
W8	0.55

Error at W5 = $\frac{\partial Etotal}{\partial W5} = \frac{\partial Etotal}{\partial OutO1} * \frac{\partial OutO1}{\partial O1} * \frac{\partial O1}{\partial W5} =$

0.082167041

MLP Calculations (with backpropagation)



Variable	Value
I1	0.05
I2	0.10
B1,B2	0.35
B3,B4	0.60
W1	0.15
W2	0.20
W3	0.25
W4	0.30
W5	0.40
W6	0.45
W7	0.50
W8	0.55

Updating W5

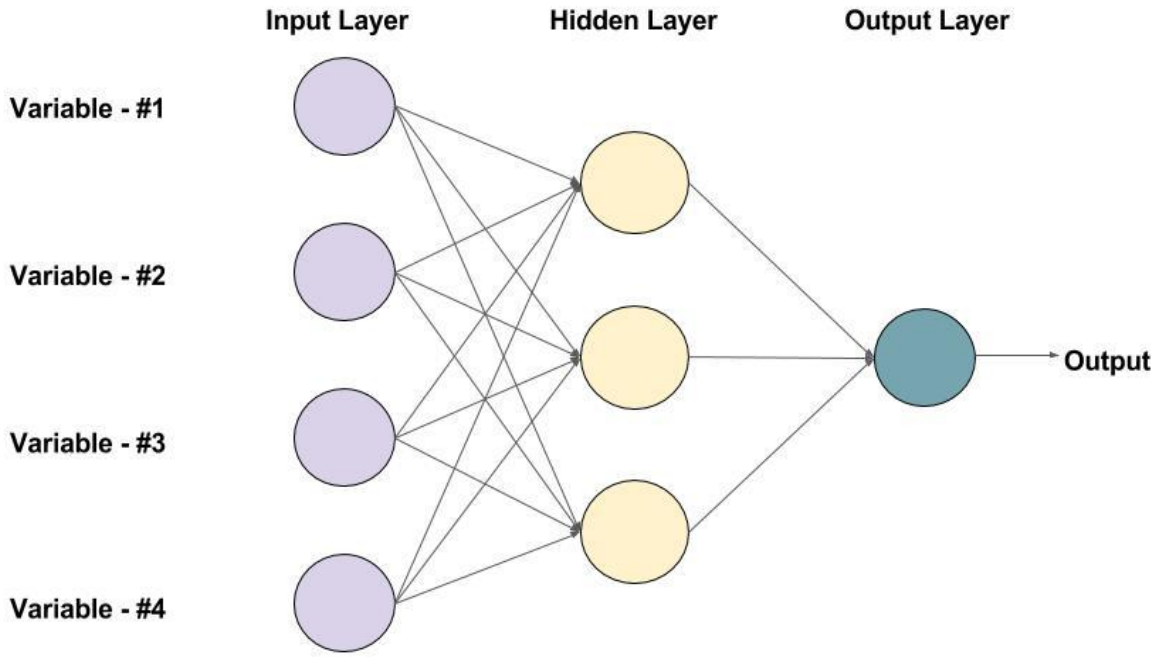
New W5 = W5 - $\eta * \frac{\partial Etotal}{\partial W5} = 0.35891648$

Similar backpropagation for all weights.

$\eta = 0.5$

Types of Neural Networks

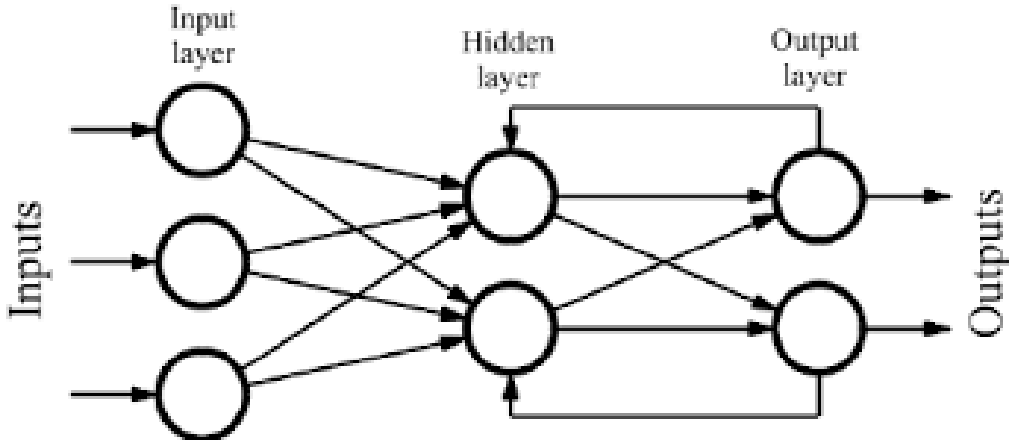
- Feed Forward NN (FF)
  - Simplest
  - Input travels in one direction only – forward
  - May or may not have multiple layers
  - No backpropagation
  - Mainly used for image classification
  - Simple to operate and maintain



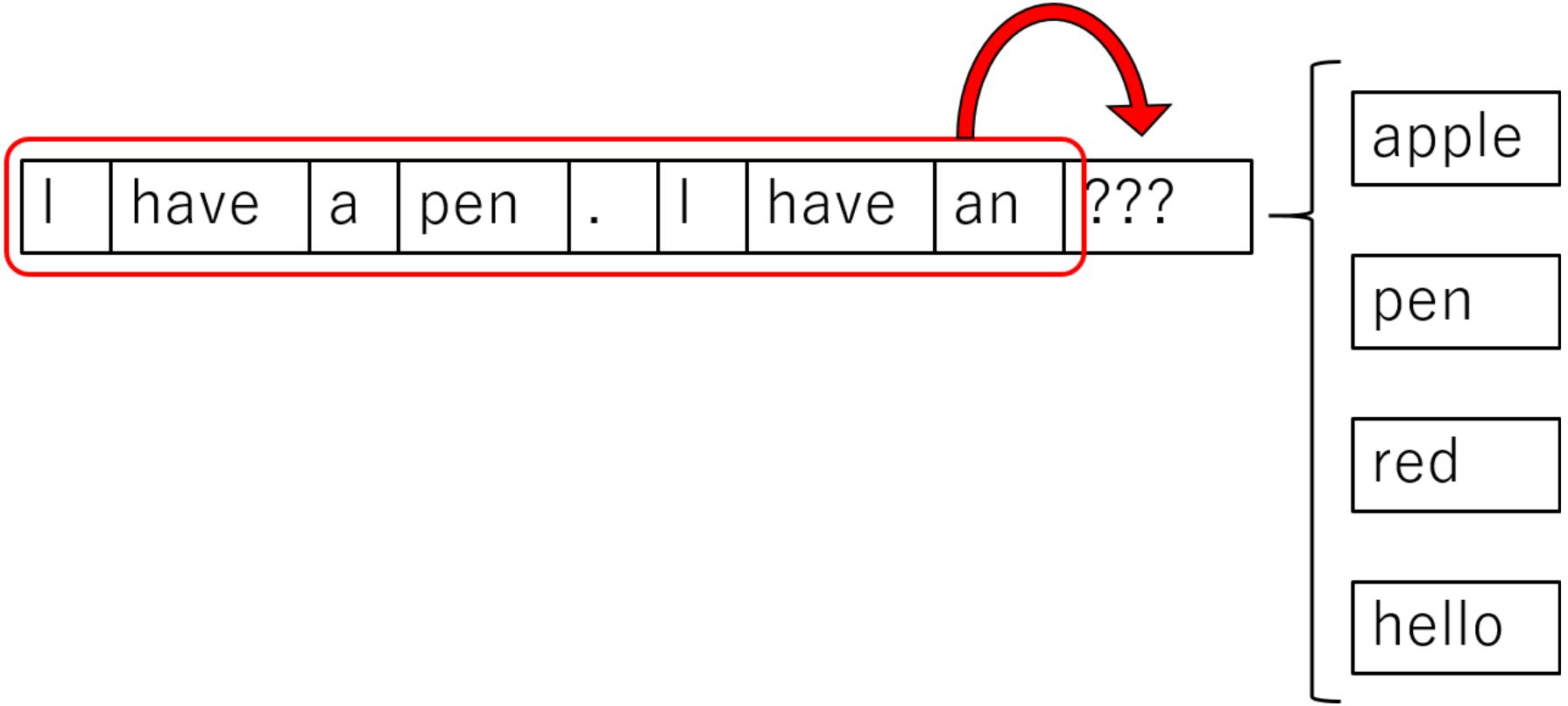
An example of a Feed-forward Neural Network with one hidden layer (with 3 neurons)

Types of Neural Networks

- Recurrent NN (RNN) – LSTM (Long Short Term Memory)
  - Has multiple layers
  - Remembers output of one layer
  - Feeds it back to the input layer
  - Acts like a memory shell – has some information to begin processing
  - Better backpropagation output
  - Used in Text to Speech, Human like speech, NLP (Summary generation)

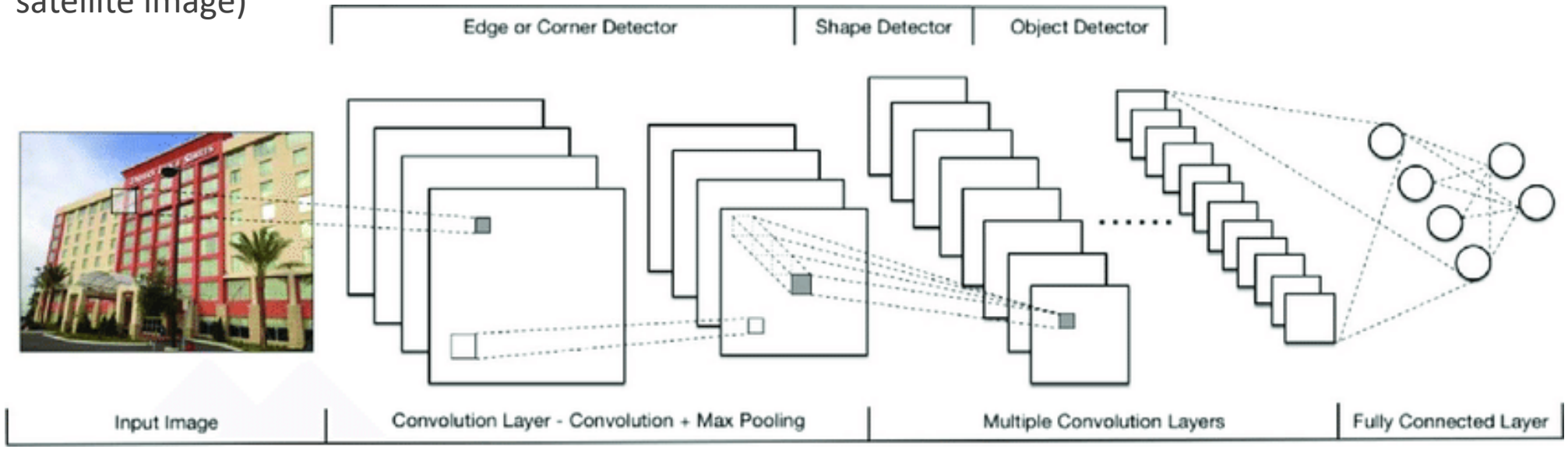


RNN (LSTM) Example



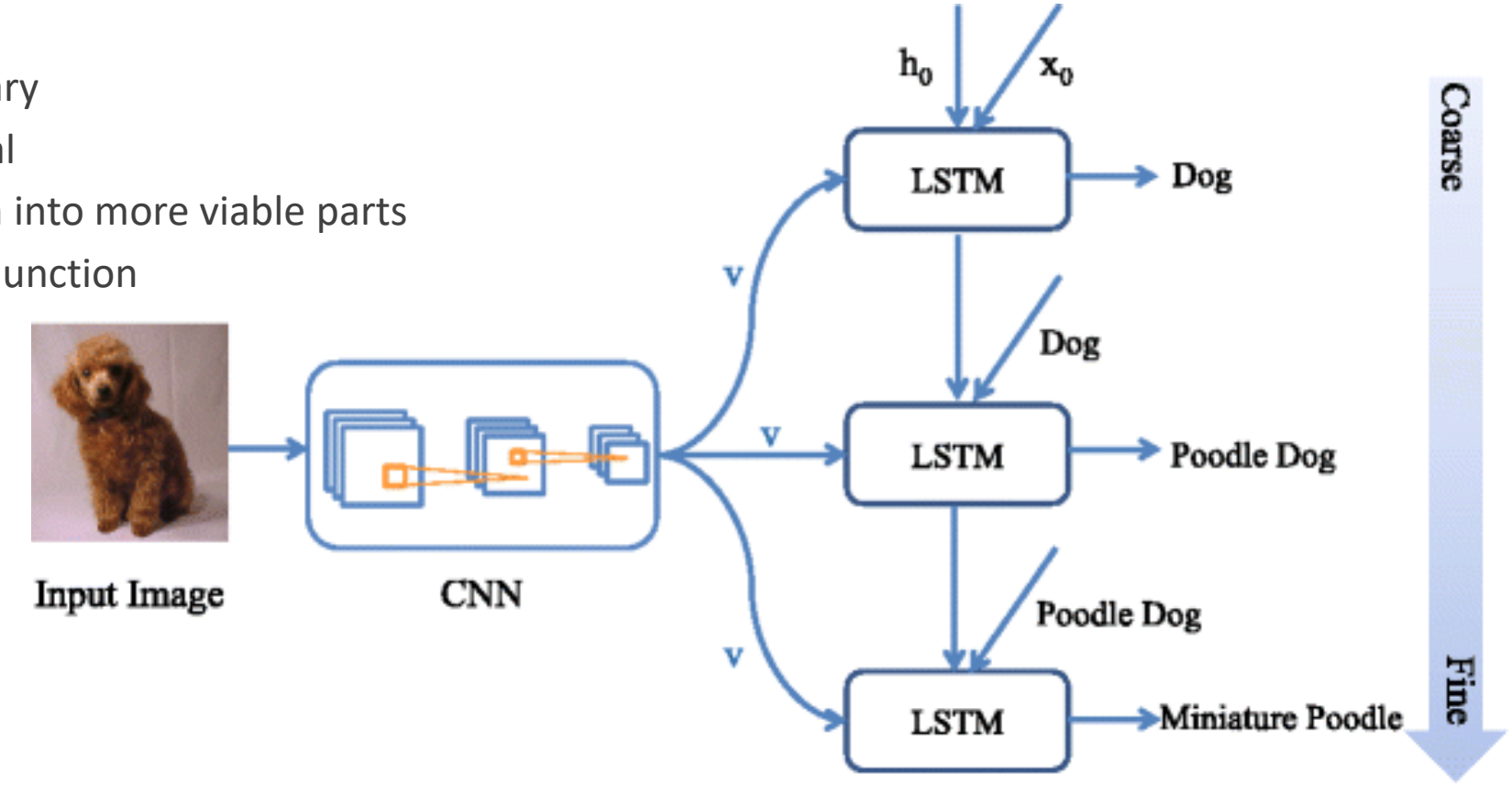
Types of Neural Networks

- Convolutional NN (CNN)
  - Similar to FF
  - Takes a chunk of input (batch input)
  - Higher accuracy
  - Used in Image Recognition, Computer Vision and Signal Processing (Prediction of agriculture yield using satellite image)



Types of Neural Networks

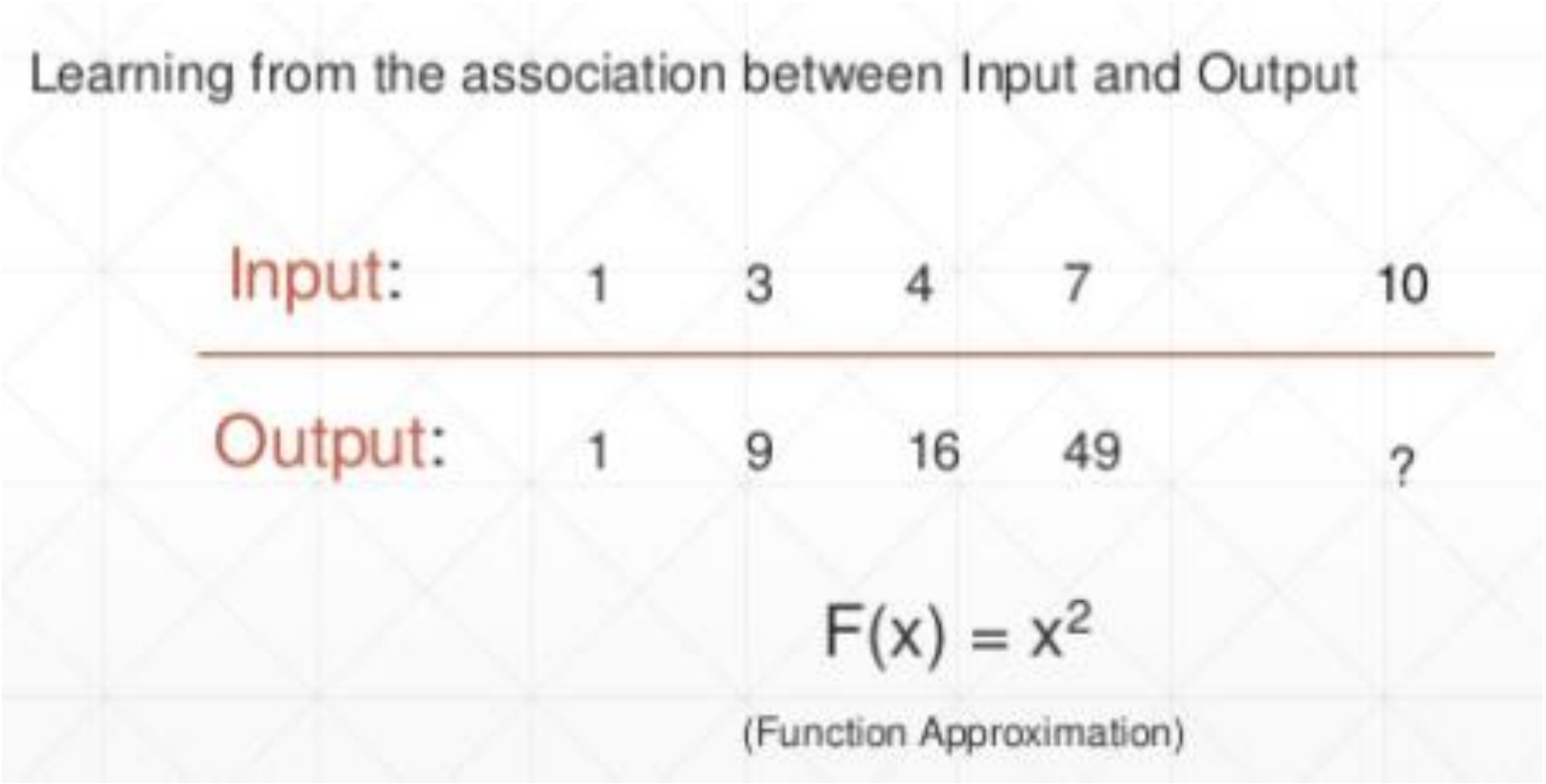
- Modular NN (MNN)
  - More than 1 NN
  - Connected by an intermediary
  - Divide and Conquer principal
  - Breaks large problems down into more viable parts
  - Simpler NNs handled in conjunction



Types of Learnings

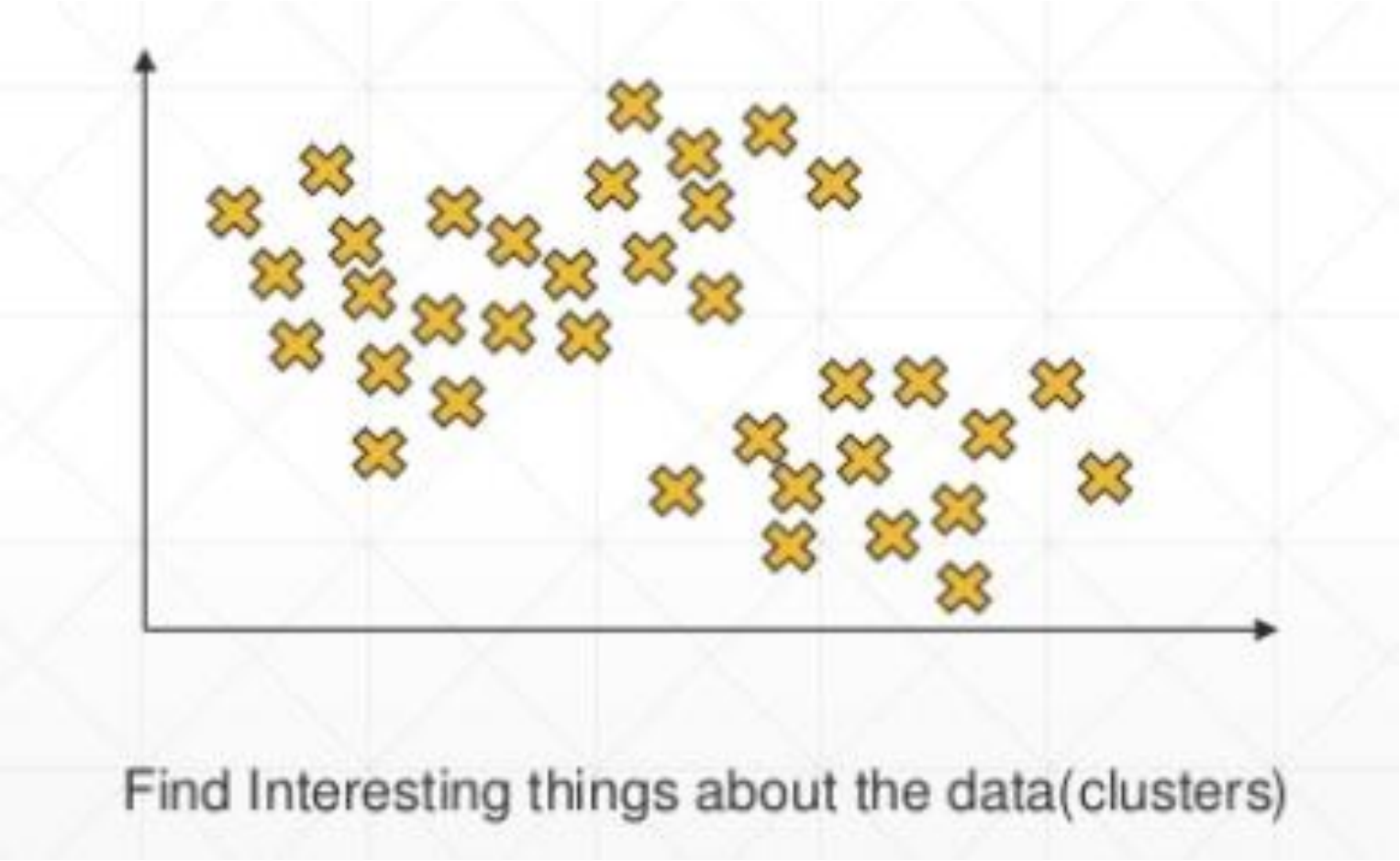
- Supervised Learning
  - Data has target values
  - We try to learn a function that can be used to predict further (Sales Prediction, Defaulter Classification)
- Unsupervised Learning
  - Data does NOT have target values
  - We try to learn what function created that data (Clustering)
- Reinforcement Learning
  - Sequential Decision situation
  - Making a decision now influences further decision
  - Learns by interacting with the environment
  - No prior information on environment available

Supervised Learning



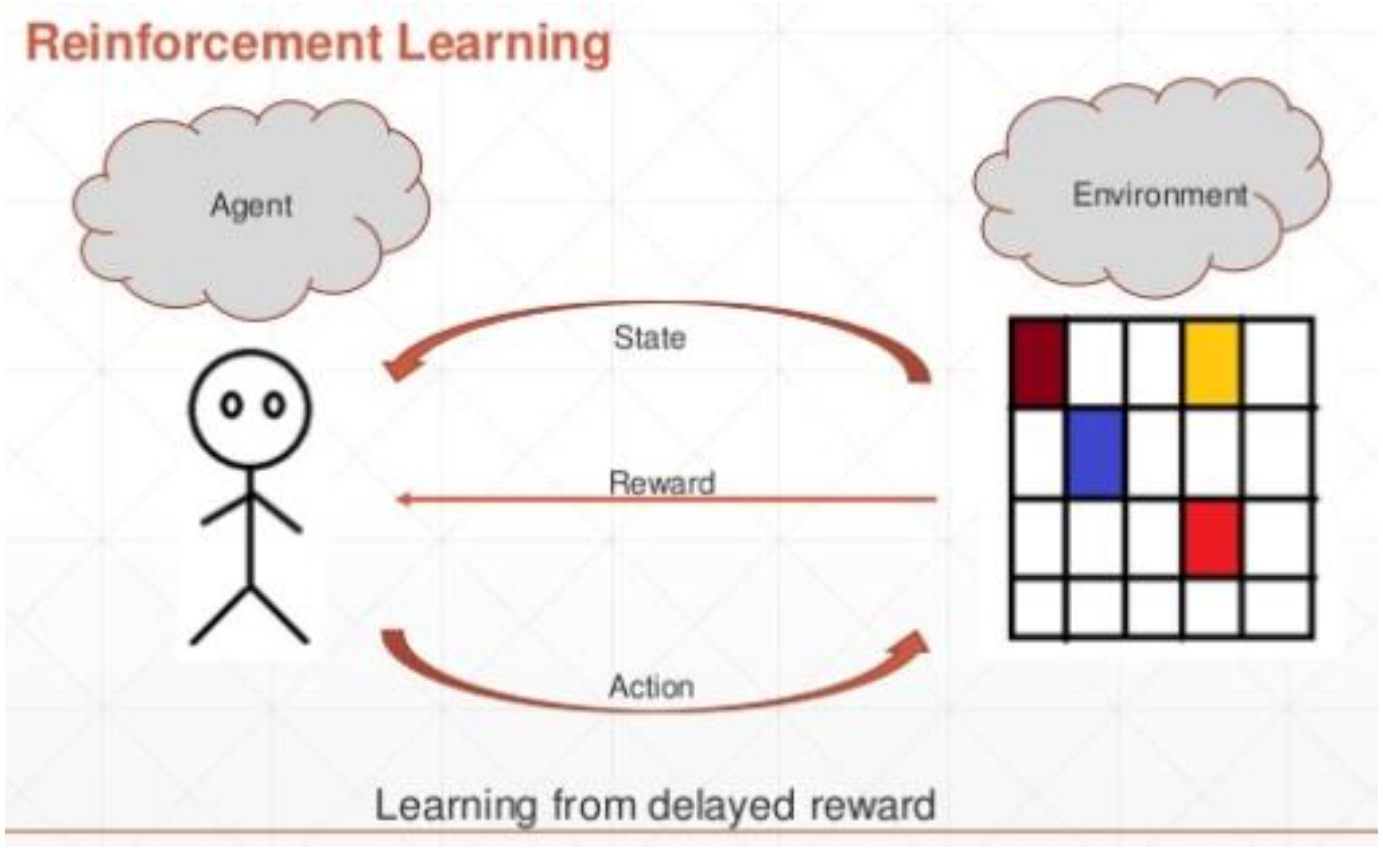
Supervised Learning

Unsupervised Learning

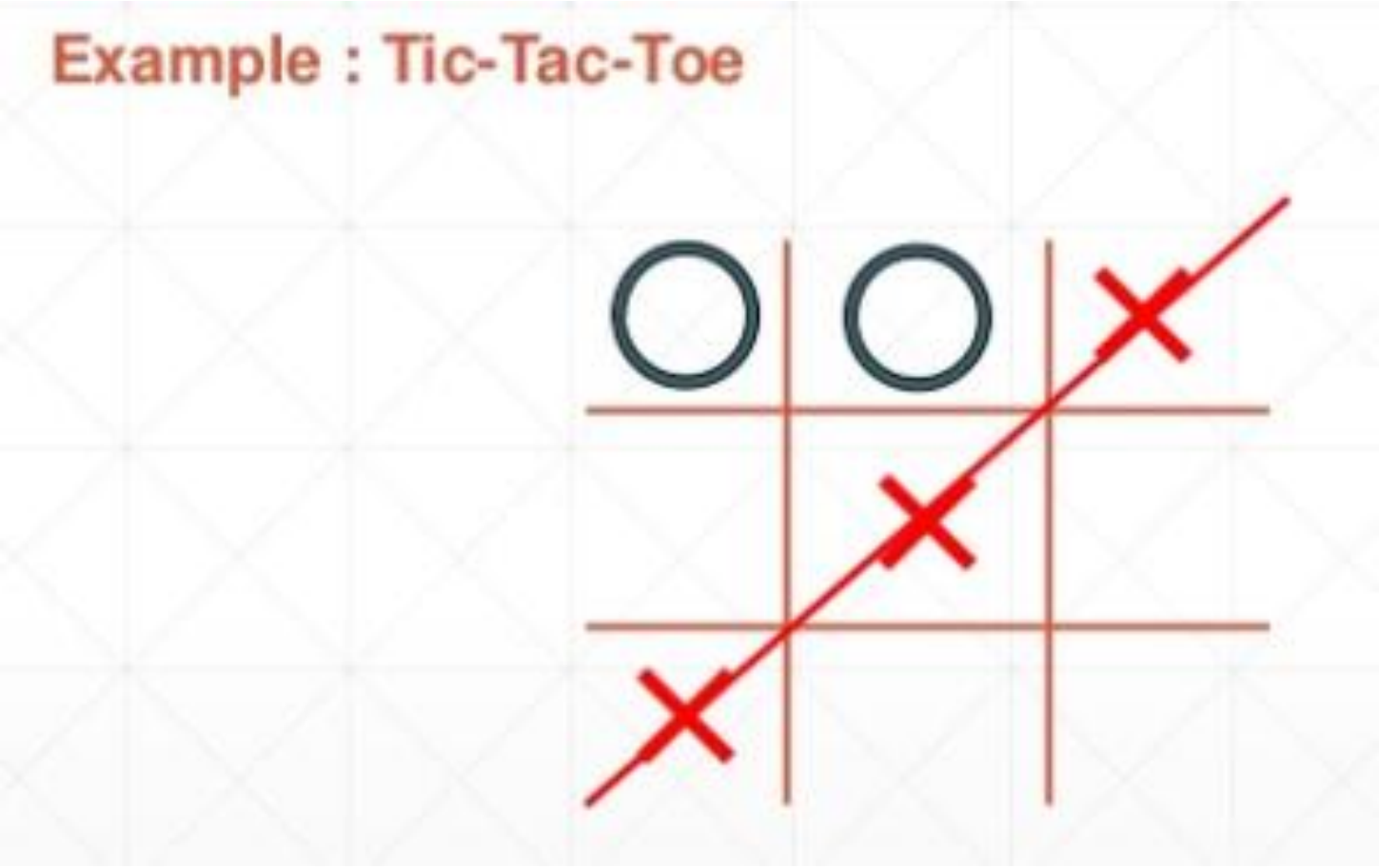


Unsupervised Learning

Reinforcement Learning



Reinforcement Learning



AI Libraries / APIs

- Keras
  - Keras is an open source high-level neural networks API, written in Python
  - It was developed by Francois Chollet with a focus on enabling fast experimentation
  - All other libraries integrate with Keras – fast, abstractive, intuitive

- TensorFlow
  - TensorFlow™ is an open source software library for high performance numerical computation
  - Developed by Google

- Theano
  - Python library that allows you to define, optimize, and evaluate mathematical expressions involving multi-dimensional arrays efficiently
  - Open Source and developed by University of Montreal, Canada

AI Libraries / APIs

- CNTK – Microsoft Cognitive Toolkit
  - Developed by Microsoft
  - Less popular
  - Complex to learn
  - Partly open source

Hands On!