

# Week 1

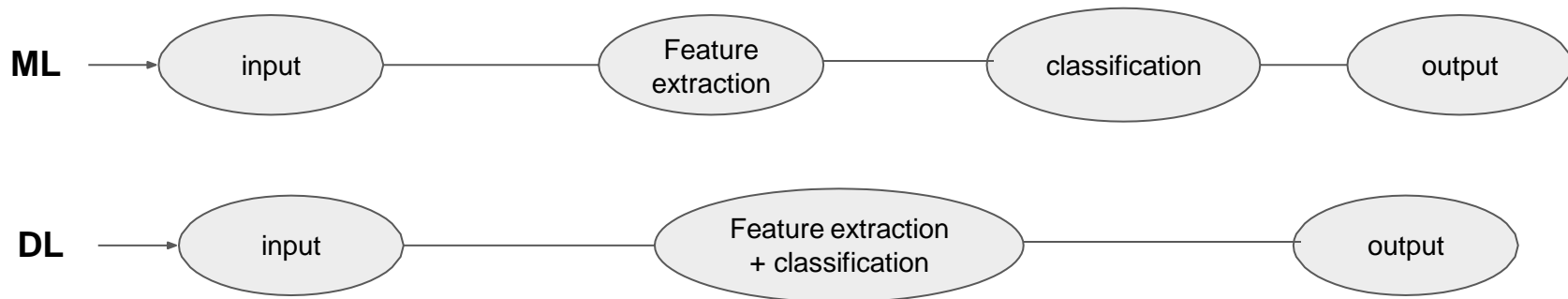
# Deep Learning: Overview

# Deep-Learning

- **Deep Learning is an important part of Machine Learning concerned with various algorithms which are inspired by the structure and function of brain called artificial neural networks.**
- **Deep Learning is considered over traditional machine learning in many instances. The main reason for this is that the deep learning algorithms tries to learn and process very high level of features from data and importantly unstructured data. So this is one of the greatest feature of deep learning over machine learning thus reducing the task of developing new feature extractors for every problem.**

# Machine Learning vs Deep Learning

Machine Learning(ML)	Deep Learning(DL)
<b>It can easily train on lesser data and requires lesser time to train</b>	<b>It requires large data and longer time to train</b>
<b>It trains on CPU</b>	<b>It trains on GPU</b>
<b>Some ML algorithms can be easily interpreted</b>	<b>Difficult to interpret.</b>



# Deep Neural Network TOC

- 1. Introduction to Deep Neural Network**
- 2. Boolean Gates and Artificial Intelligence**
- 3. Warren McCulloch and Walter Pitts Neuron**
- 4. Rosenblatt Neuron Perceptron**
- 5. Artificial Neural Network**
- 6. Application of ANN (Classification and Regression using Neural Networks)**
- 7. Activation Functions**
- 8. Softmax Activation Function**
- 9. Forward Propagation and Bias**
- 10. Loss Function**
- 11. Types of Loss Function (Classification)**
- 12. Types of Loss Function (Regression)**
- 13. BackPropagation and Learning rate**
- 14. Optimization**
- 15. Drawbacks of ANN**

# Introduction to DNN

## What is Deep Neural Network?

# Deep Neural Network / Artificial Neural Network

1. **Deep Neural Network is an Artificial Neural Network with more than one hidden layer.**
2. **An artificial Neural Network (ANN) is a model that employs a collection of artificial neurons to extract the patterns in the data that represent the relationship between independent and dependent variables.**
3. **Artificial Neurons are based on the observed behavior of neurons in biological brains. A collection of artificial neurons mimic the behavior of biological neural networks.**

# Deep Neural Network / Artificial Neural Network *(continued)*

- 4. Just as brain uses a network of interconnected neurons to parallelize the processing of input signals to trigger a response, ANN also make use of interconnected neurons to parallelly work on input signals and give an output.**
- 5. In the initial days of the journey towards modern Deep Neural Networks, the research in artificial neuron and ANN were closely tied to the research in neurology. The research in ANN borrowed ideas such as thresholds, linear summation, neuron firing or not etc from Neurology.**
- 6. However, the two fields today are delinked and one does not need to know anything about biological neurons to work on ANN / DNN.**

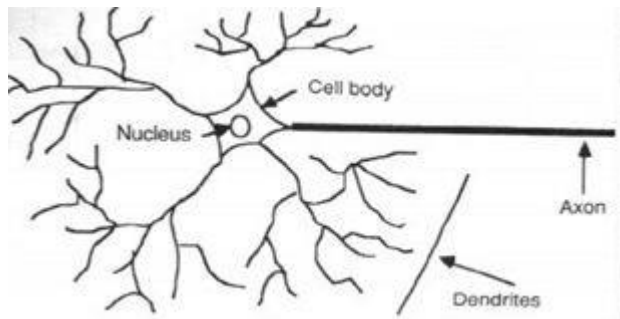


# How it evolved

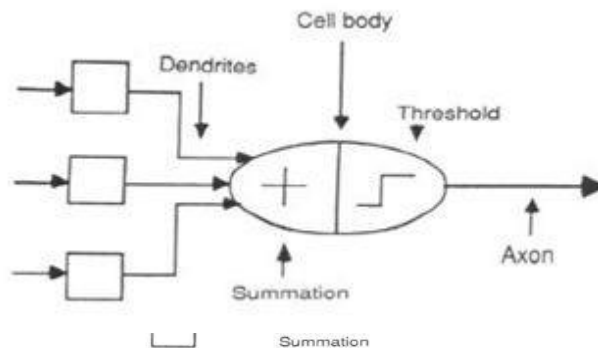
## (History of Artificial Neurons / Neural Networks)

# Inspiration from Neurology

## Biological Neuron




## Artificial Neuron




# Boolean Gates and Artificial Intelligence

# Boolean Gates and Artificial Neurons

- 1. The objective of research in artificial neurons was to develop a computing system that could learn to do tasks on its own without instruction on how to do it.**
- 2. Most of the tasks that we do in our day-to-day life are classified. Hence, research was on to develop an artificial neuron that could classify.**
- 3. Since computing systems are based on Boolean gates which generated two classes, it was natural to check whether the artificial neurons can learn to mimic these gates such as the OR, AND gates.**

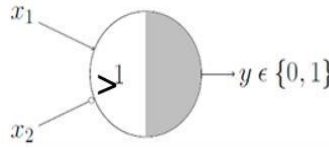
<b>AND</b>		<b>INPUT</b>		<b>OUTPUT</b>	
		A	B	Q	
		0	0	0	
		0	1	0	
		1	0	0	
		1	1	1	

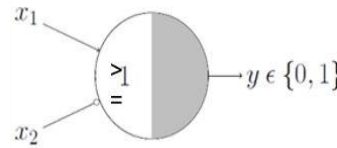
<b>OR</b>		<b>INPUT</b>		<b>OUTPUT</b>	
		A	B	Q	
		0	0	0	
		0	1	1	
		1	0	1	
		1	1	1	

# Warren McCulloch & Walter Pitts Neuron

# Warren McCulloch & Walter Pitts “AND”, “OR” Neuron



AND



OR

1. In McCulloch Pitt neurons inputs and outputs are binary. The output is only one but inputs can be many
2. All inputs have same positive weights (not shown in the figure)
3. The inputs multiplied with corresponding weights are summed up and the result sent to a step function.
4. The threshold of the step function is fixed (for e.g. 1 for “AND” gate with two inputs each with weight 1)
5. The threshold has to be modified for a gate, for e.g. ‘AND’ gate depending on the number of inputs. What would the threshold for the “AND” gate be if the number of inputs is  $x_1, x_2, x_3$ ?

# McCulloh Pitts Neuron

AND Gate

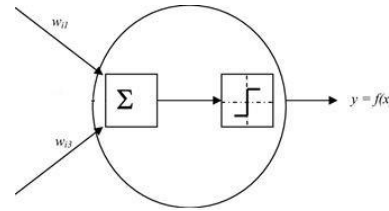
```
w[1] = 1
w[0] = 1
training_data = [
    (array([0,0]), 0),
    (array([0,1]), 0),
    (array([1,0]), 0),
    (array([1,1]), 1), ]
```

# Step function with threshold of >1. Anything below is 0  
step\_function = lambda x: 0 if x < 1 else 1

```
w[1] = 1
w[0] = 1
```

```
for x, _ in training_data:
    result = dot(x, w)
    print("{}: {}-> {}".format(x[:2], result, step_function(result)))
```

## McCulloh Pitts Neuron



McCullohPitt\_Ros  
enBlat\_Ne

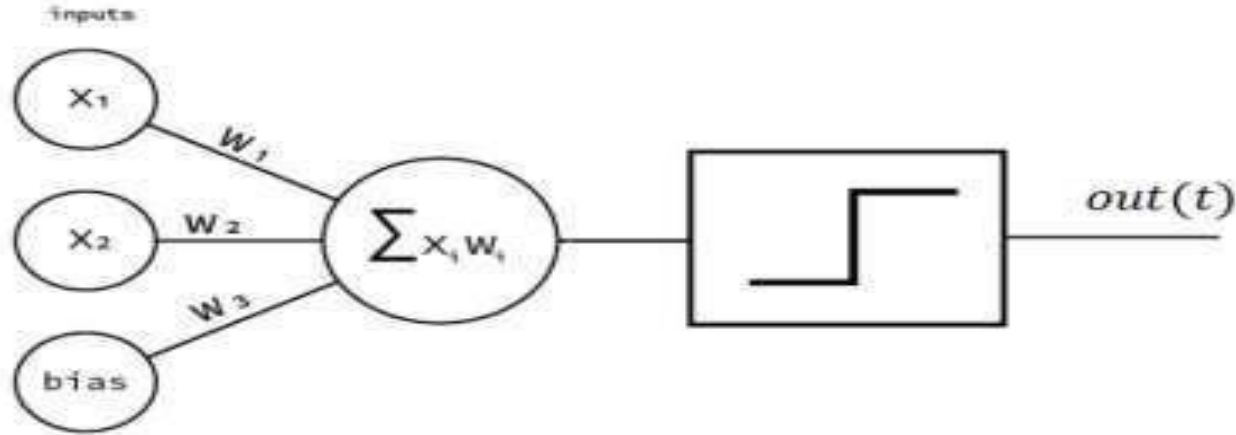
**The McCulloch-Pitts model of a neuron is simple. However, this model is so simplistic that it only generates a binary output and also the weight and threshold values are fixed.**

Proprietary content. © Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited.



# Rosenblatt Neuron Perceptron

# Rosenblatt Neuron / Perceptron



- **Uses weights for the inputs. A concept borrowed from William Hebb's rule**  
"neurons that wire together, fire together"
- **Other than the inputs they also get a special 'bias' input, which just has a value of 1**
- **Incorrect outputs for a given input are captured as errors**

# Rosenblatt Neuron / Perceptron (*continued*)

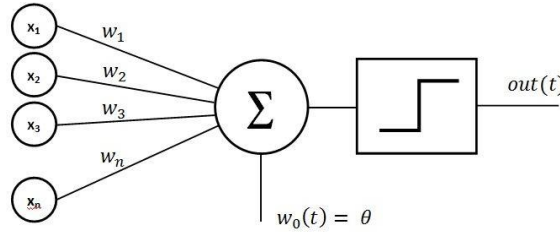
- **A learning rule modifies the weight to correct errors in output**
- **The neuron could behave like “AND” gate or “OR” gate with same threshold by adjusting weights.**
- **This neuron learns the patterns from the data. If the data is for “AND” gate, it learns to mimic “AND” gate.**
- **This concept of learning from data was a major improvement over McCulloch-Pitts neuron.**

# Rosenblatt's Artificial Neuron / Perceptron...

- **Rosenblatt defined a Perceptron as a simple mathematical model of biological neurons**
- **It takes inputs as a set of binary values, every input is multiplied by a weight ( synaptic strength to each nearby neuron),**
- **A threshold to evaluate the sum of weighted inputs against. Output a 1 (neuron firing) if the weighted sum is more than the threshold, else a zero (neuron not firing)**
- **Other than the inputs from sensors / neighboring neurons, they also get a special 'bias' input, which just has a value of 1**
- **The bias is like an intercept in a linear equation. Useful in generating more functions with the same inputs**

- **This was an improvement of the work of Warren McCulloch and Walter Pitts, McCulloch-Pitts neuron.**
- **McCulloch Pitts neuron had a fixed set of weights associated with inputs and as a result, did not have the ability to learn.**
- **They had no bias term either Rosenblatt's neuron was inspired by a rule coined by Donald Hebb (researcher studying biological neurons), according to which learning in brain is stored in form of changes to the strength of relationships between two connected neurons "Neurons that fire together, wire together".**
- **Rosenblatt's neuron can model the basic OR/AND/NOT functions (building blocks of computing systems)**
- **This was a big step towards the belief that making computers able to perform formal logical reasoning would essentially solve AI.**

# Rosenblatt Neuron / Perceptron



1. **Weights for the inputs are not the same, can be positive or negative**
2. **The output can be -1, 0, 1 unlike MCP neuron where the output is only 0 or 1**
3. **The neuron is associated with a learning rule that modifies the weight to ensure with the same threshold, the neuron can behave like an “AND” gate or “OR” gate with no need for any threshold modification**
4. **This neuron learns from the data. It has the intelligence to learn the pattern from the data**

# Machine Learning (Artificial Neural Network) / Perceptron

## Rosenblatt's Neuron Functioning

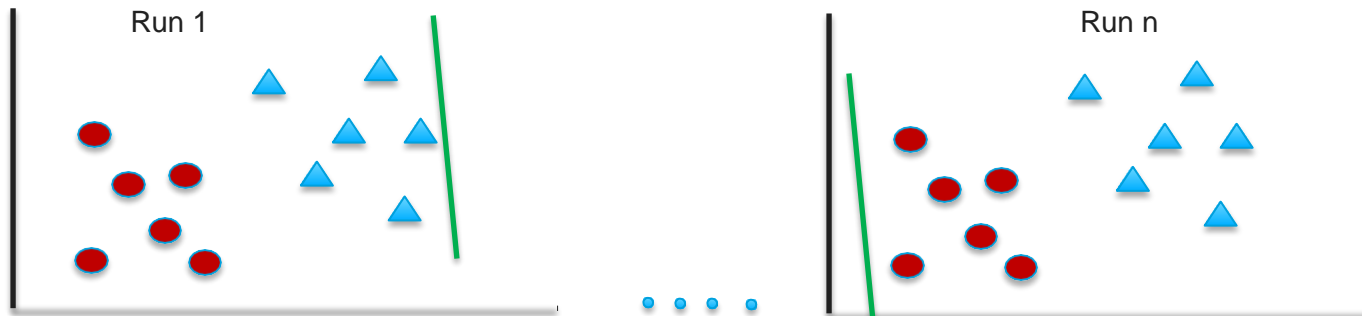
1. Start with a random set of weights for all the input variables, and the bias.
2. For the input data point, compute the output using the weights and the bias.
3. If the calculated output does not match the expected output (from training data) modify the weights \*\*
4. Go to the next example in the training set and repeat steps j - k until the Perceptron makes no more mistakes.

**\*\* The relation between weight adjustment and errors and a learning rate will make the learning possible. Learning is the process finding the right combination of weights that minimizes the errors in the training data set**

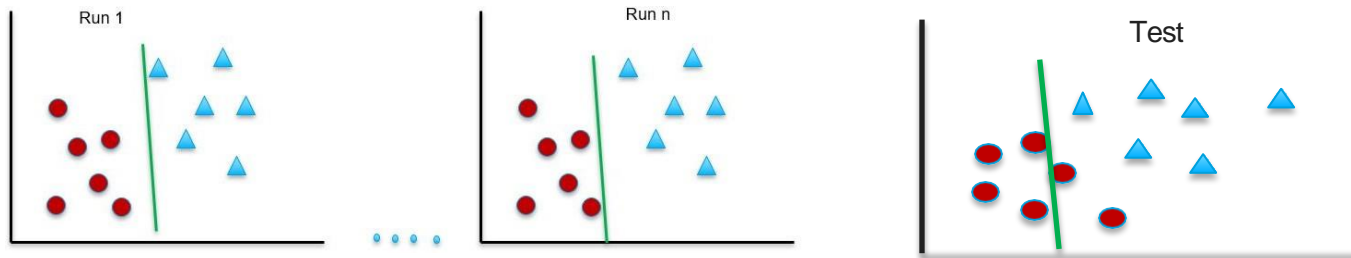
# Perceptron

## Perceptron Learning Algorithm–

1. **Select random sample from training set as input. Draw the first random line (green) such that blue triangles lie above it and red circles ones below**
2. **If the classification is correct, do nothing. But the first time many blue triangles are on the wrong side!**
3. **If the classification is incorrect, modify the weight vector  $w$  and shift the green line**
4. **Repeat this procedure until the entire training set is classified correctly**
5. **Howsoever times we run this algorithm, it will find a surface that separates the two classes**



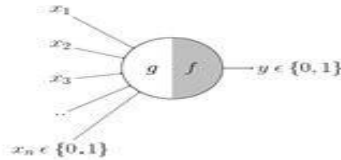




6. **The Convergence theorem guarantees that when the classes are linearly separable in the training set, the perceptron will find that surface that separates the two classes correctly**
7. **The perceptron algorithm does not guarantee it will be able to separate the two classes correctly even when the classes are linearly separable**
8. **Why? Because it does not look for an optimal plane. It stops the moment it finds the separator plane (a.k.a dichotomizer).**
9. **Since the planes are passing very close to the data points in the training set, they may not perform well in the test set where the distribution of the data will be different**

# MP Neuron Vs Perceptron

## Warren McCulloch & Walter Pitts Neuron

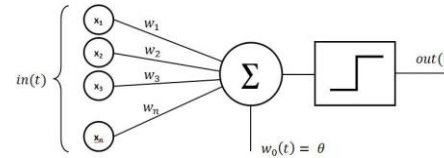


$$g(x_1, x_2, x_3, \dots, x_n) = g(\mathbf{x}) = \sum_{i=1}^n x_i$$

$$y = f(g(\mathbf{x})) = \begin{cases} 1 & \text{if } g(\mathbf{x}) \geq \theta \\ 0 & \text{if } g(\mathbf{x}) < \theta \end{cases}$$

1. It has an input layer that acts like dendrites.
2. It has two parts, the first part,  $g$  is weighted addition of inputs. The weights are manually initialized and all have same weight.
3. The weighted sum is passed through an activation function  $f$  which yields a 1 if threshold is crossed, else 0.

## Rosenblatt's Perceptron



1. It has an Input layer that acts as dendrites
2. It has two parts, first part is weighted addition  
Each input is multiplied with a weight (which is typically initialized with some random value)
3. The sum is then passed through an activation function which yields a 1 if threshold is crossed
4. The step function can be defined in such a way that output can range from -1 to +1
5. It captured the error and had a logic built in to adjust the weights automatically to reduce the error

# Weakness of Perceptron

1. **Papert and Minsky demonstrated that the perceptron was incapable of handling some of the binary gates such as XOR\*\***
2. **Given that it cannot represent all possible binary gates, it could not have been used for all possible computations hence the objective of computer based AI was a pipe dream!**
3. **XOR is an example of distribution of classes not linearly separable in two dimensions but is easily separable in higher 3 dimensional space. Ref: [http://www.mind.ilstu.edu/curriculum/artificial\\_neural\\_net/xor\\_problem\\_and\\_solution.php](http://www.mind.ilstu.edu/curriculum/artificial_neural_net/xor_problem_and_solution.php)**
4. **It was subsequently demonstrated that instead of making a neuron intelligent, a network of neurons can be used to do what a single neuron could not**
5. **This was the birth of a Artificial Neural Network**

**Lab - McCullohPitt\_RosenBlat\_Neurons.ipynb**

# Weakness of Perceptron

6. **Papert and Minsky demonstrated that the perceptron was incapable of handling some of the binary gates such as XOR\*\***
7. **Given that it cannot represent all possible binary gates, it could not have been used for all possible computations hence the objective of computer based AI was a pipe dream!**
8. **XOR is an example of distribution of classes not linearly separable in two dimensions but is easily separable in higher 3 dimensional space. Ref: [http://www.mind.ilstu.edu/curriculum/artificial\\_neural\\_net/xor\\_problem\\_and\\_solution.php](http://www.mind.ilstu.edu/curriculum/artificial_neural_net/xor_problem_and_solution.php)**
9. **It was subsequently demonstrated that instead of making a neuron intelligent, a network of neurons can be used to do what a single neuron could not**
10. **This was the birth of a Artificial Neural Network**

**Lab - McCullohPitt\_RosenBlat\_Neurons.ipynb**

# Artificial Neural Network

# Origins of Artificial Neural Networks

1. **Perceptrons were replaced with artificial neuron which not only had a weighted summation operation but also included a non-linearity function.**
2. **Multitude of such neurons working together could solve those problems where individual perceptrons failed**
3. **Multiple neurons in a single layer is akin to transforming data from lower dimensions to higher dimensions! (Cover's theorem in action)**
4. **Kolmogorov theorem\* states that any continuous function  $f(x_1, x_2, \dots, x_n)$  defined on  $[0, 1]$  with  $n > 2$  can be expressed in form of two carefully chosen functions**

$$f(x_1, x_2, \dots, x_n) = \sum_{j=1}^{2n+1} g_j \left( \sum_{i=1}^n \phi_{ij}(x_i) \right)$$

5. **Looks similar to layer of neurons with non-linear function  $g$  acting on summation of weighted inputs !  
No wonder NeuralNets form basis of complex processing**

\* [https://en.wikipedia.org/wiki/Universal\\_approximation\\_theorem](https://en.wikipedia.org/wiki/Universal_approximation_theorem) (another mathematical theorem)

# Origins of Artificial Neural Networks

1. **Perceptrons were replaced with artificial neuron which not only had a weighted summation operation but also included a non-linearity function.**
2. **Multitude of such neurons working together could solve those problems where individual perceptrons failed**
3. **Multiple neurons in a single layer is akin to transforming data from lower dimensions to higher dimensions! (Cover's theorem in action)**
4. **Kolmogorov theorem\* states that any continuous function  $f(x_1, x_2, \dots, x_n)$  defined on  $[0, 1]$  with  $n > 2$  can be expressed in form of two carefully chosen functions**

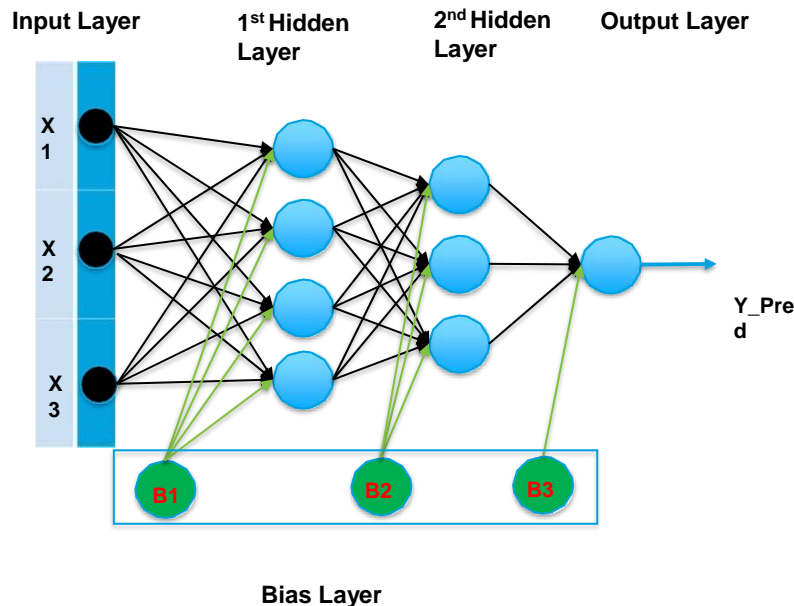
$$f(x_1, x_2, \dots, x_n) = \sum_{j=1}^{2n+1} g_j \left( \sum_{i=1}^n \phi_{ji}(x_i) \right)$$

5. **Looks similar to layer of neurons with non-linear function  $g$  acting on summation of weighted inputs !  
No wonder NeuralNets form basis of complex processing**

\* [https://en.wikipedia.org/wiki/Universal\\_approximation\\_theorem](https://en.wikipedia.org/wiki/Universal_approximation_theorem) (another mathematical theorem)

# Artificial Neural Network

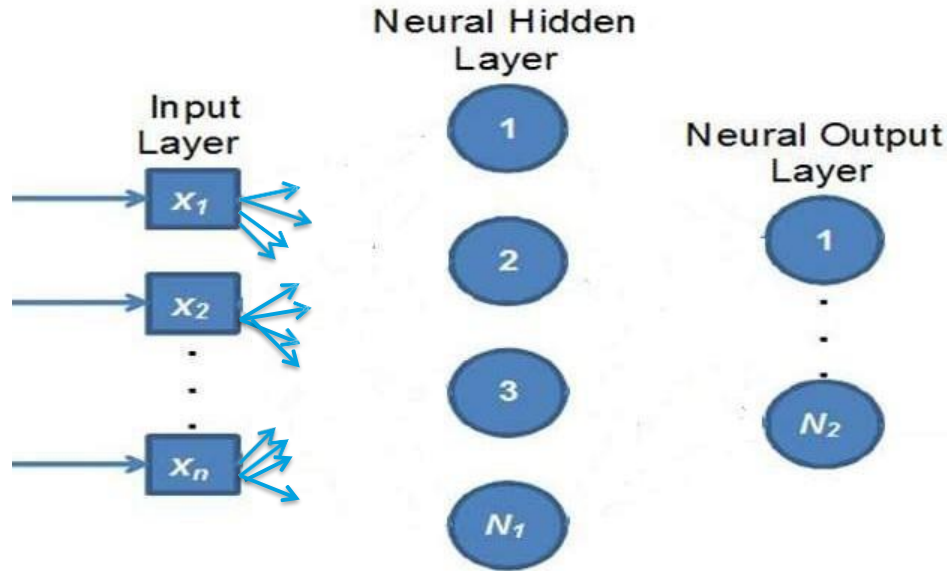
The processing element of an ANN is called *a node*, representing the artificial neuron. Each ANN is composed of a collection of nodes grouped in layers. A typical structure is shown. The initial layer is the input layer and the last layer is the output layer. In between, we have the hidden layers





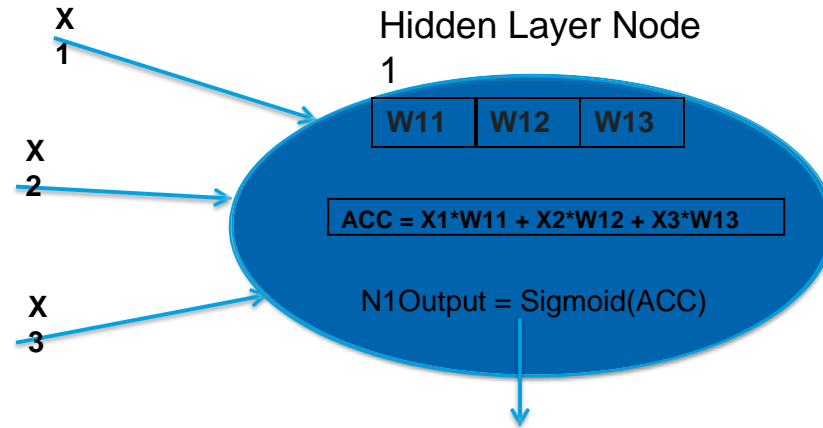
# Fully Connected Artificial Neural Network

**1. The input layer is passive, does no processing, only holds the input data to supply it to the first hidden layer.**



# Fully connected Artificial Neural Network

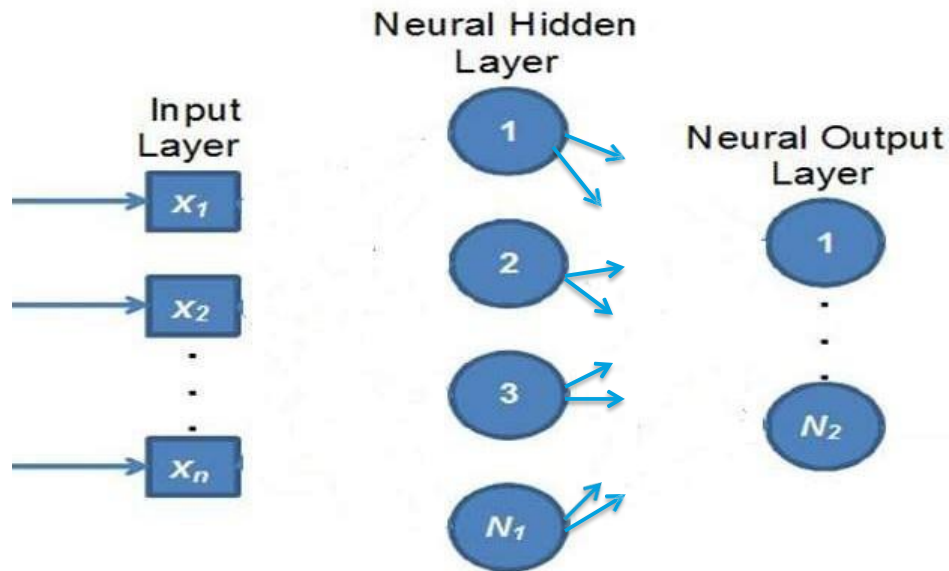
- Each node in the first hidden layer, takes all input attributes, multiplies with the corresponding weights, adds bias and the output is transformed using non\_linear function



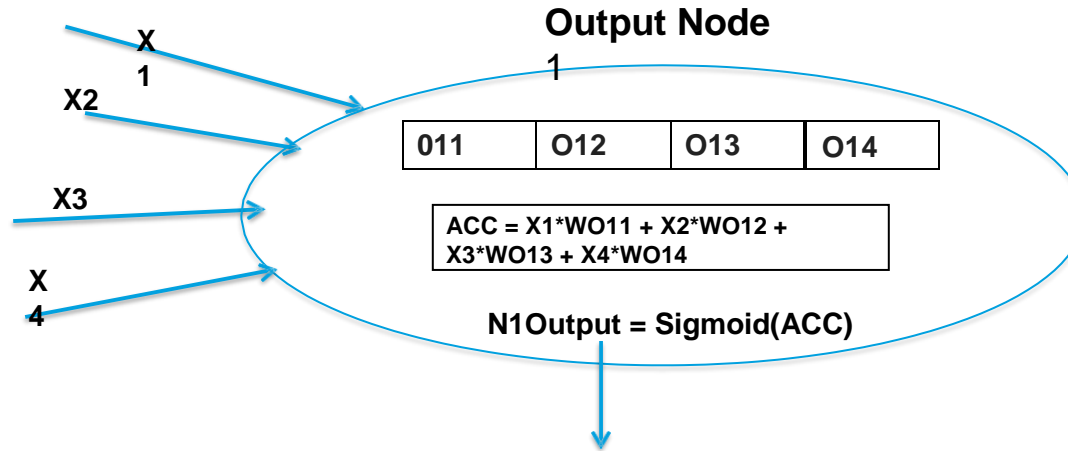
- The weights for a given hidden node is pre-fixed and all the nodes in the hidden layer have their own weights
- The output of each node is fed to output layer nodes or another set of hidden nodes in another hidden layer

# Fully connected Artificial Neural Network

**5. The output value of each hidden node is sent to each output node in the output layer**

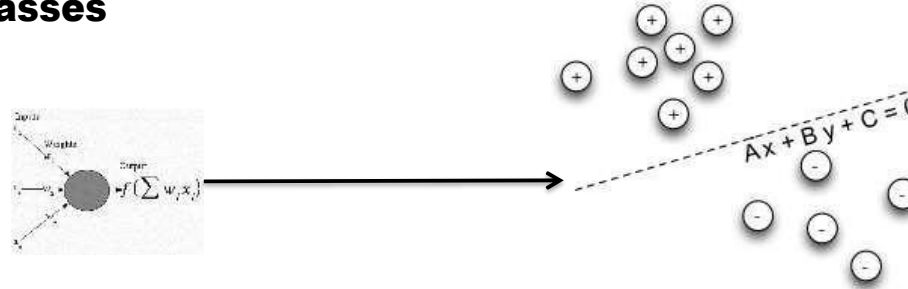


# Fully connected Artificial Neural Network



# Fully connected Artificial Neural Network

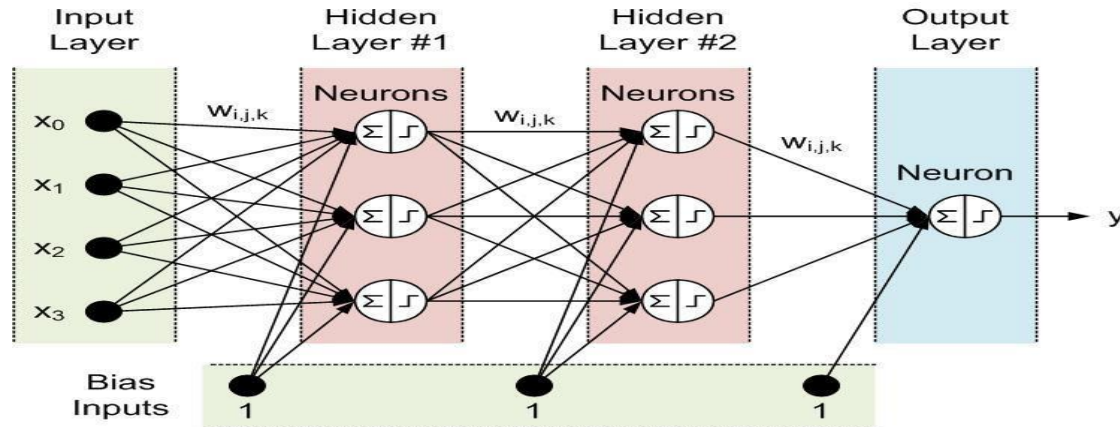
6. In a binary output ANN, the output node acts like a perceptron classifying the input into one of the two classes



7. Examples of such ANN applications would be to detect fraudulent transaction, whether a customer will buy a product given the attributes etc.
8. ANN can also be used for multi-class classification problems such as digit recognition. In fact, it is often used for multi-class classification

# Machine Learning (Artificial Neural Network)

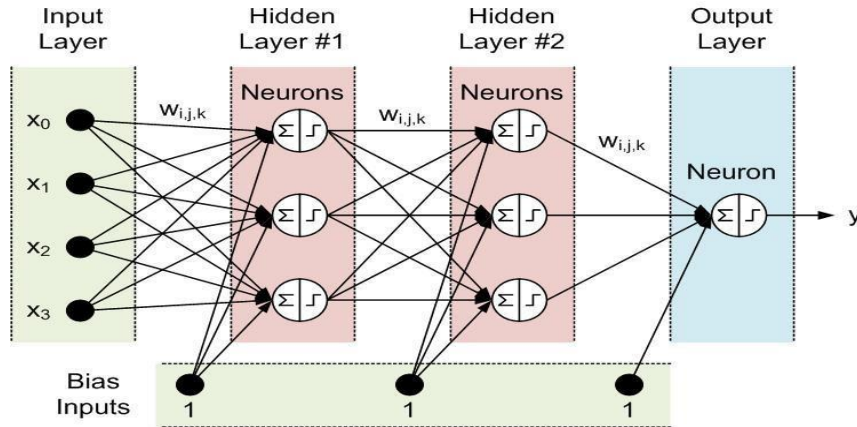
10. **The processing elements of a ANN is called a *node*, representing the artificial neuron. Each ANN is composed of a collection of nodes grouped in layers. A typical structure is shown The initial layer is the input layer and the last layer is the output layer. In between we have the hidden layers**



# Machine Learning (Artificial Neural Network)

## 11. Mathematical foundations for artificial neural networks

- a. **Kolmogorov theorem** – any continuous function  $f$  defined on  $n$ -dimensional cube is representable by sums and superpositions of continuous functions of only one variable

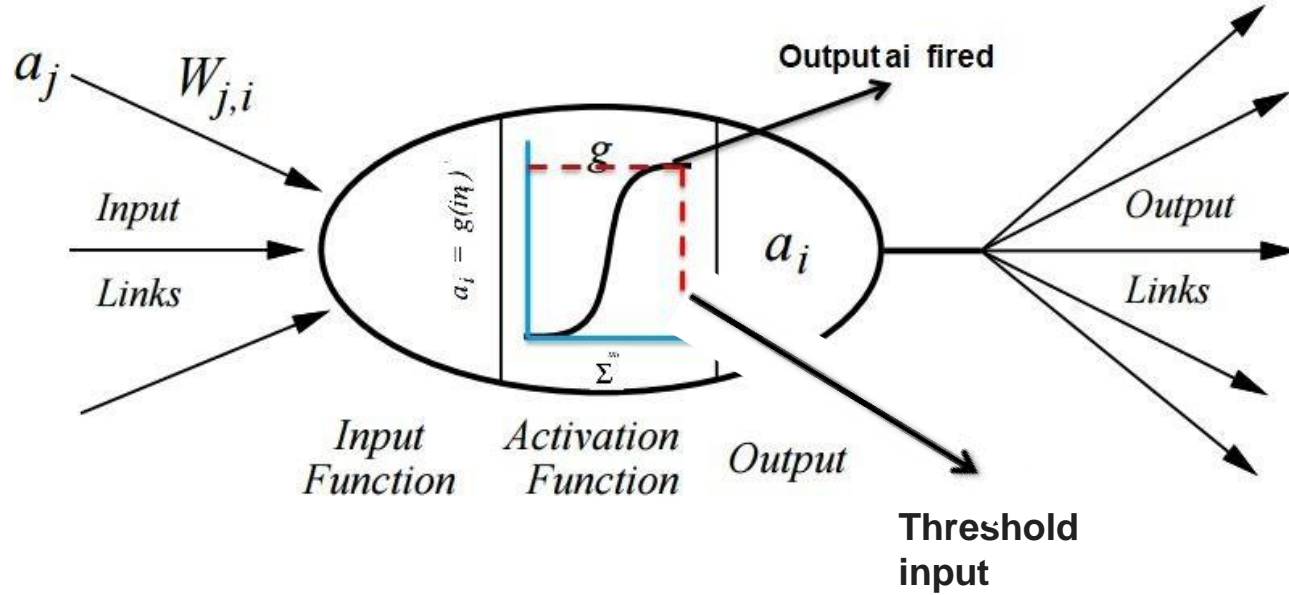


$$f(x_1, x_2, \dots, x_n) = \sum_{q=1}^{2n+1} g \left( \sum_{p=1}^n \lambda_p \phi_q(x_p) \right)$$

- b. **Cover's theorem** - states that given a set of training data that is not linearly separable, one can with high probability transform it into a training set that is linearly separable by projecting it into a higher-dimensional space via some non-linear transformation.

# Machine Learning (Artificial Neural Network)

12. A given node will fire and feed a signal to subsequent nodes in next layer only if the non-linear function it implements reaches a threshold.
13. In ANN use of Sigmoid function is more common than step function.

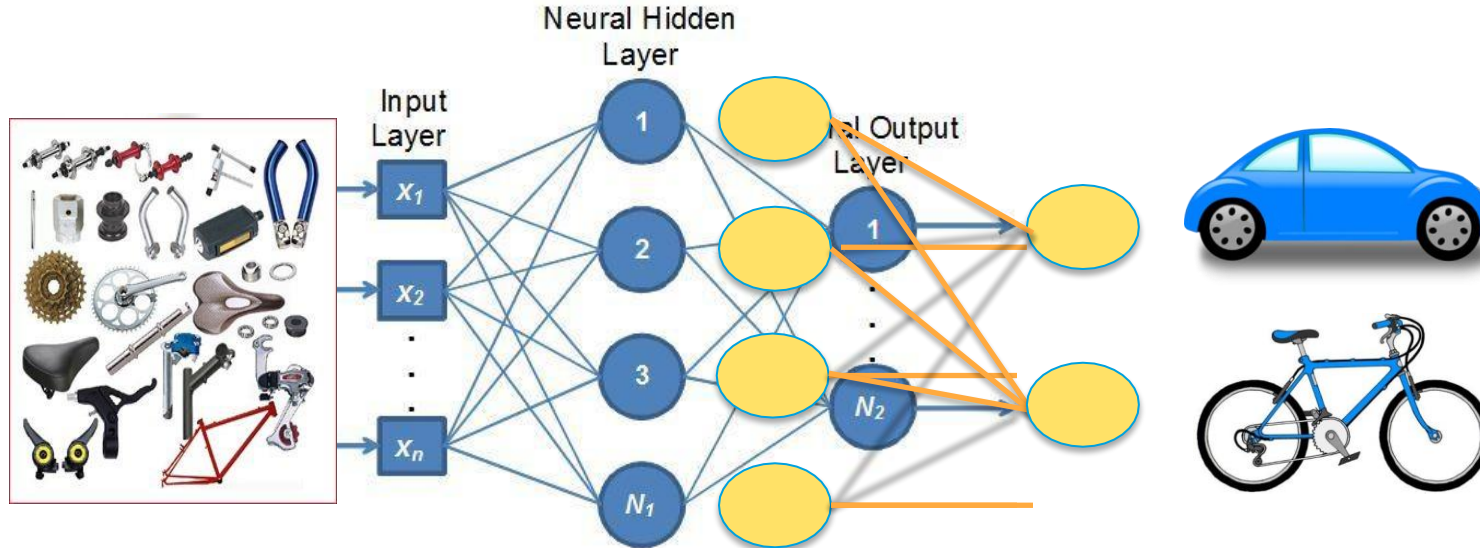




# Application of ANN

# Machine Learning (Artificial Neural Network)

- We can have a ANN with multiple output nodes where a given output node may or may not get triggered given the input and the weights.



- We can have a ANN with multiple output nodes where a given output node may or may not get triggered given the input and the weights.

# Machine Learning (Artificial Neural Network)

- The weights required to make a neural network carry out a particular task are found by a learning algorithm, **together with** examples of how the system *should* operate
- The examples in vehicle identification could be a large hadoop file of several millions sample segments such as bicycle, motorcycle, car, bus etc.
- The learning algorithms calculate the appropriate weights for each classification for all nodes at all the levels in the network
- If we consider each input as a dimension then ANN labels different regions in the n-dimensional space. In our example one region is cars, other region is bicycle

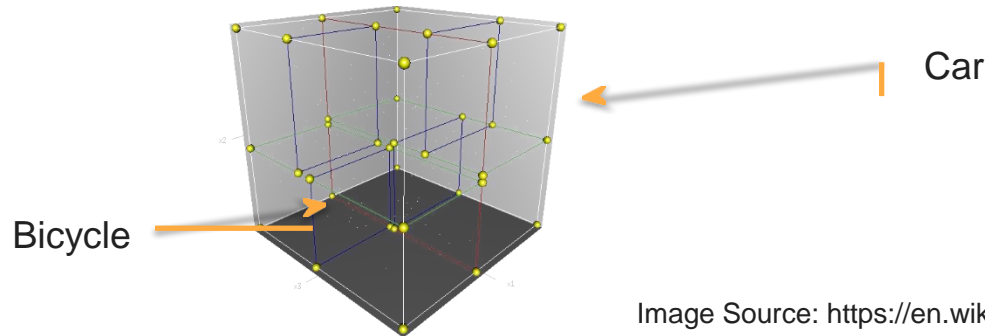


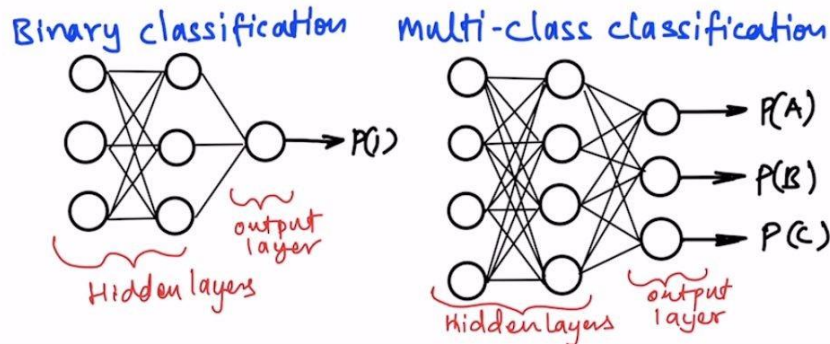
Image Source: [https://en.wikipedia.org/wiki/K-d\\_tree#/media/File:3dtree.png](https://en.wikipedia.org/wiki/K-d_tree#/media/File:3dtree.png)

# ANN for Classification

**Classification always refers to a predictive modeling problem, where the input data is classified into one of the predefined existing classes. In Supervised machine learning the output labels are always known. For example: predicting a patient having cancer or not is an example of binary classification problem.**

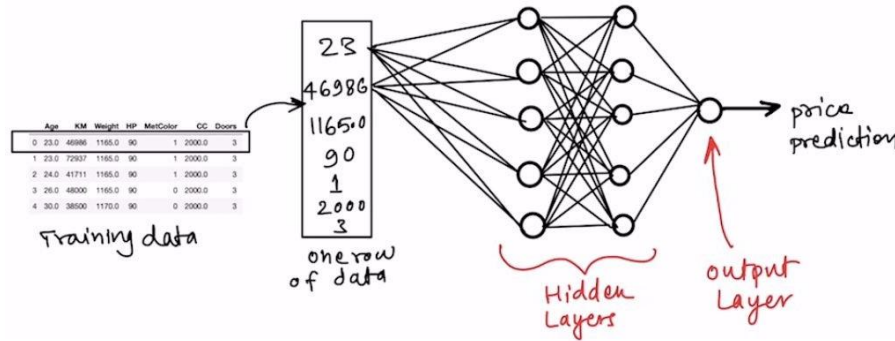
**We have come across many Machine Learning algorithms used for classification but Neural networks can mainly be used when we have many output classes or labels and we have high amount of data for the model.**

**ANN has many advantages when used for classification but it even has some disadvantages like higher computational costs and longer training time.**



# ANN for Regression

- Regression always refers to a predictive modeling problem, where the output values are predicted as a continuous value as a function of inputs.



- Regression models mostly work well when the regression equations fits well on the data. And using ANN is similar to classification but we need to replace the final layer used in the classification model with a fully connected layer with a single node with a single activation function.

