BDA 5201
Machine Learning for Big Data

II Semester ME (BDA)

Evaluation Method

Marks for Grading

- Theory 100 Marks
  - Internal 50 + End Sem 50 : 100
- Lab 100 Marks
  - Internal 50 + End Sem 50 : 100

Example:

Relative Grading

< 40	F
40 – 49	E
50 – 58	D
59 – 67	C
68 – 76	B
77 – 87	A
88 – 100	A+

Topics Covered

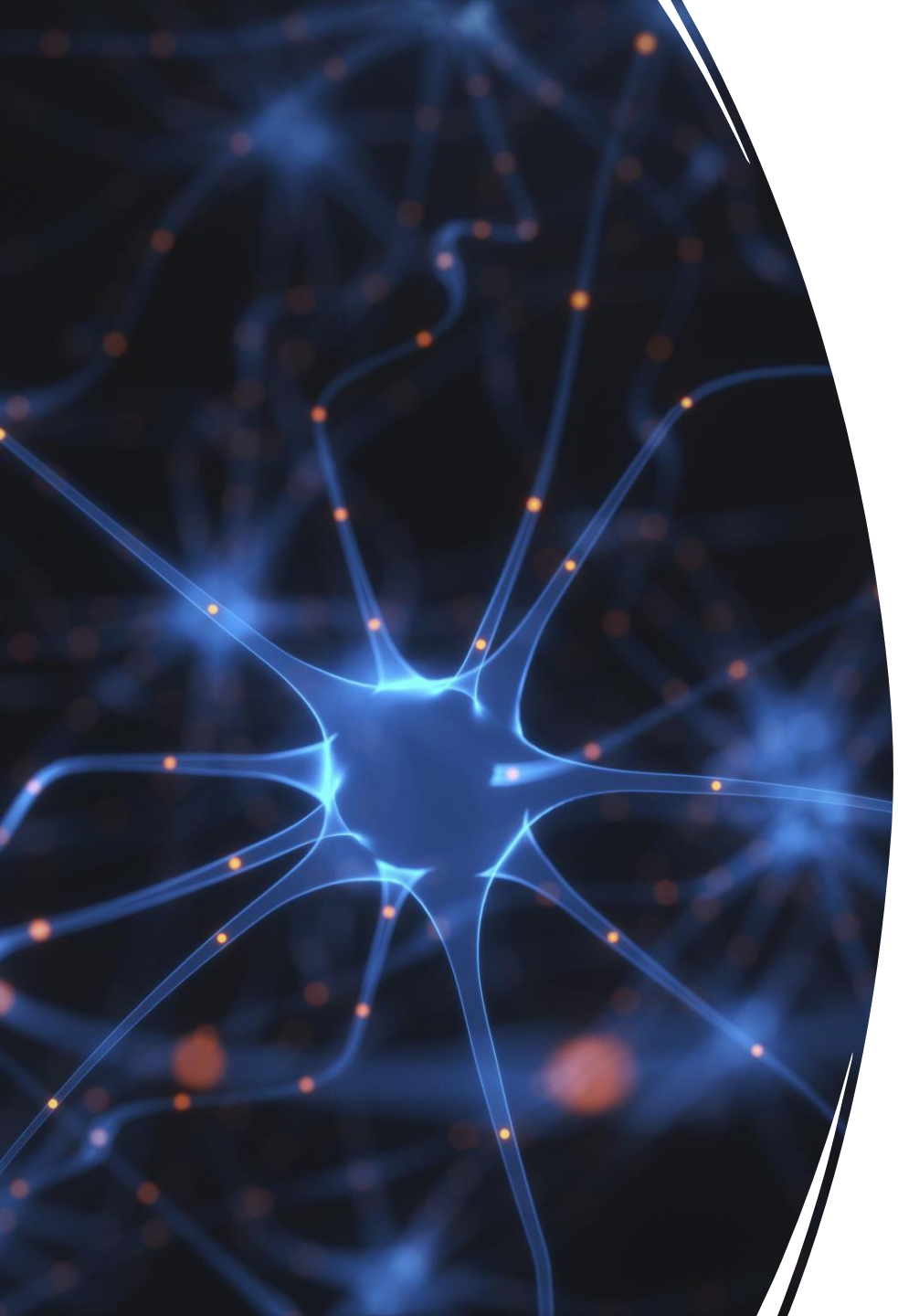
Artificial Neural Networks

Clustering

Support Vector Machines & Kernel Methods

Deep Learning

Reinforcement Learning



Artificial Neural Networks

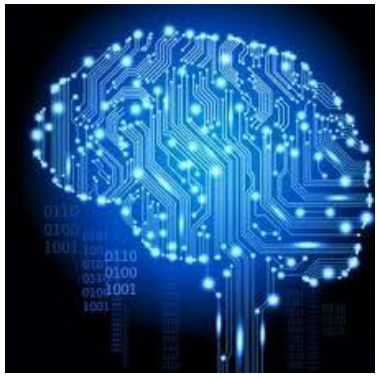
Lecture 1

Why Artificial Neural Networks?

ANN modeled (inspired by biological neurons)

- ***Create Intelligence to solve complex problems***

Modern Machine



not present in the modern computers

Massive parallelism, Low energy consumption

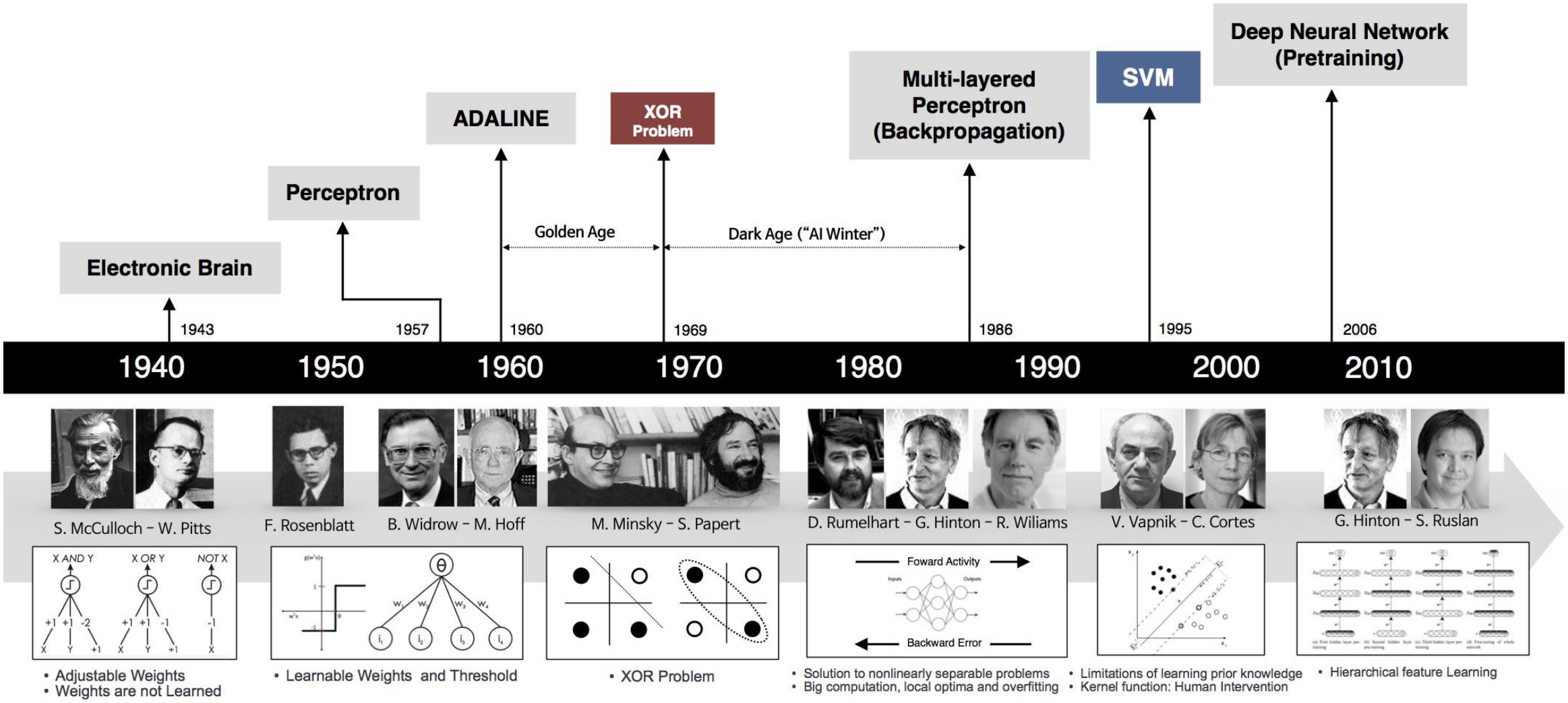
Learning ability, Fault tolerance

Characteristics of a human brain

Adaptivity (self-organization)

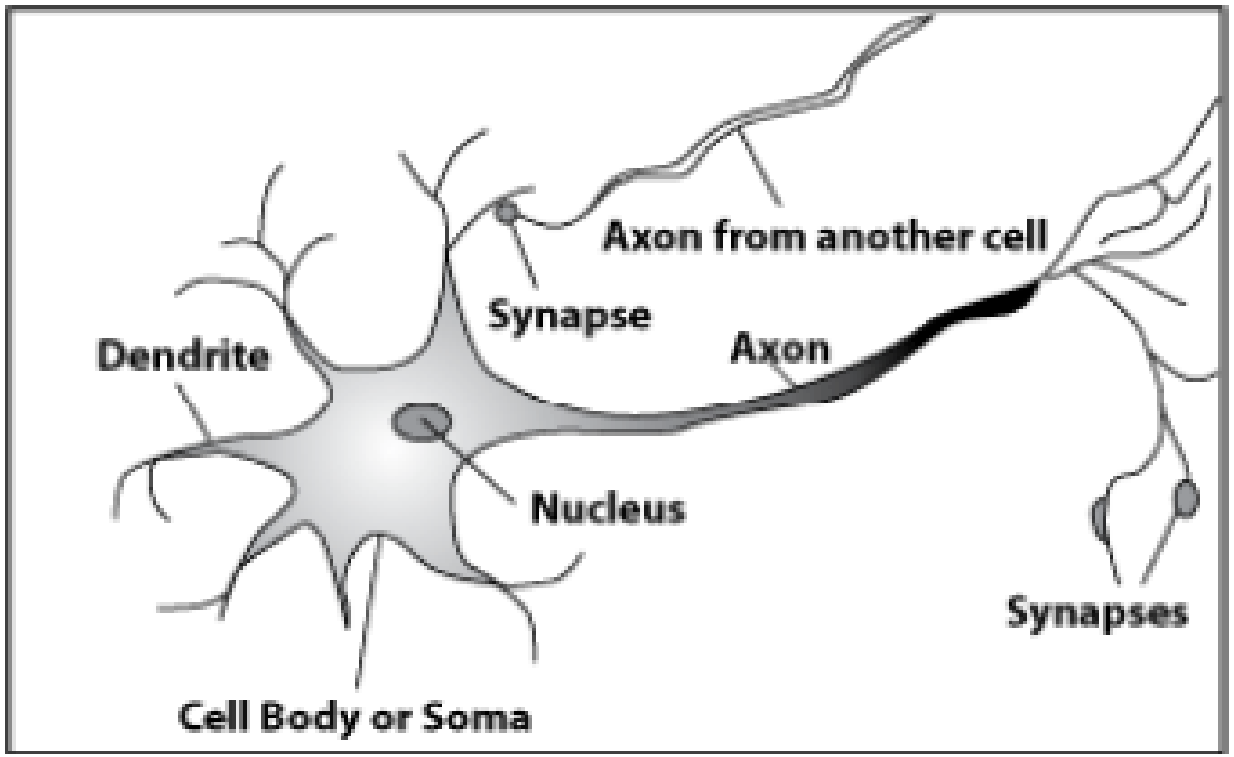
Generalization ability, Contextual information processing

History of Artificial Neural Networks



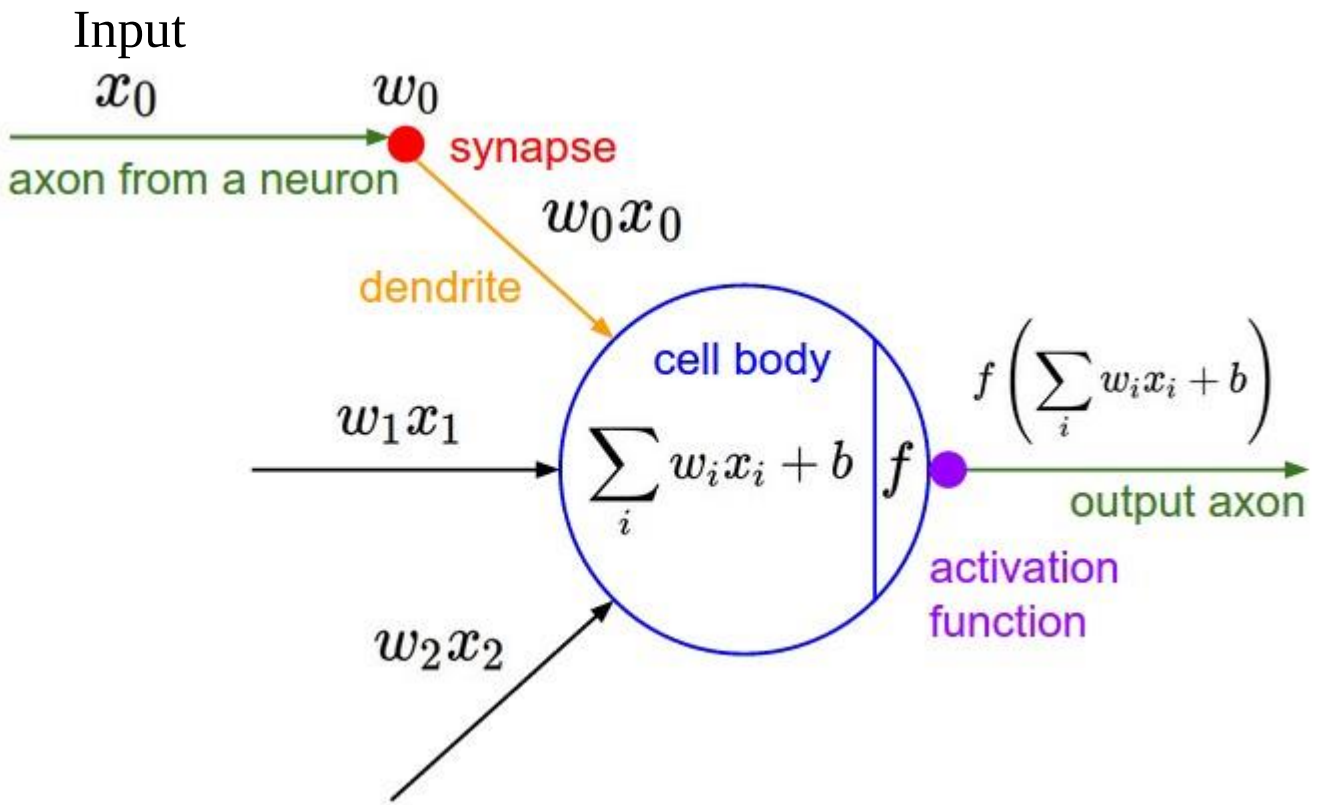
Biological Neural Networks

- In biology, a neuron is a cell that can transmit and process chemical or electrical signals.
- A neuron is connected with other neurons to create a network.
- Tens of billions of interconnected neuron structures – in human brain.
- Every neuron has an
  - Input called the **dendrite**
  - Cell body called **Soma**
  - Output called the **axon**

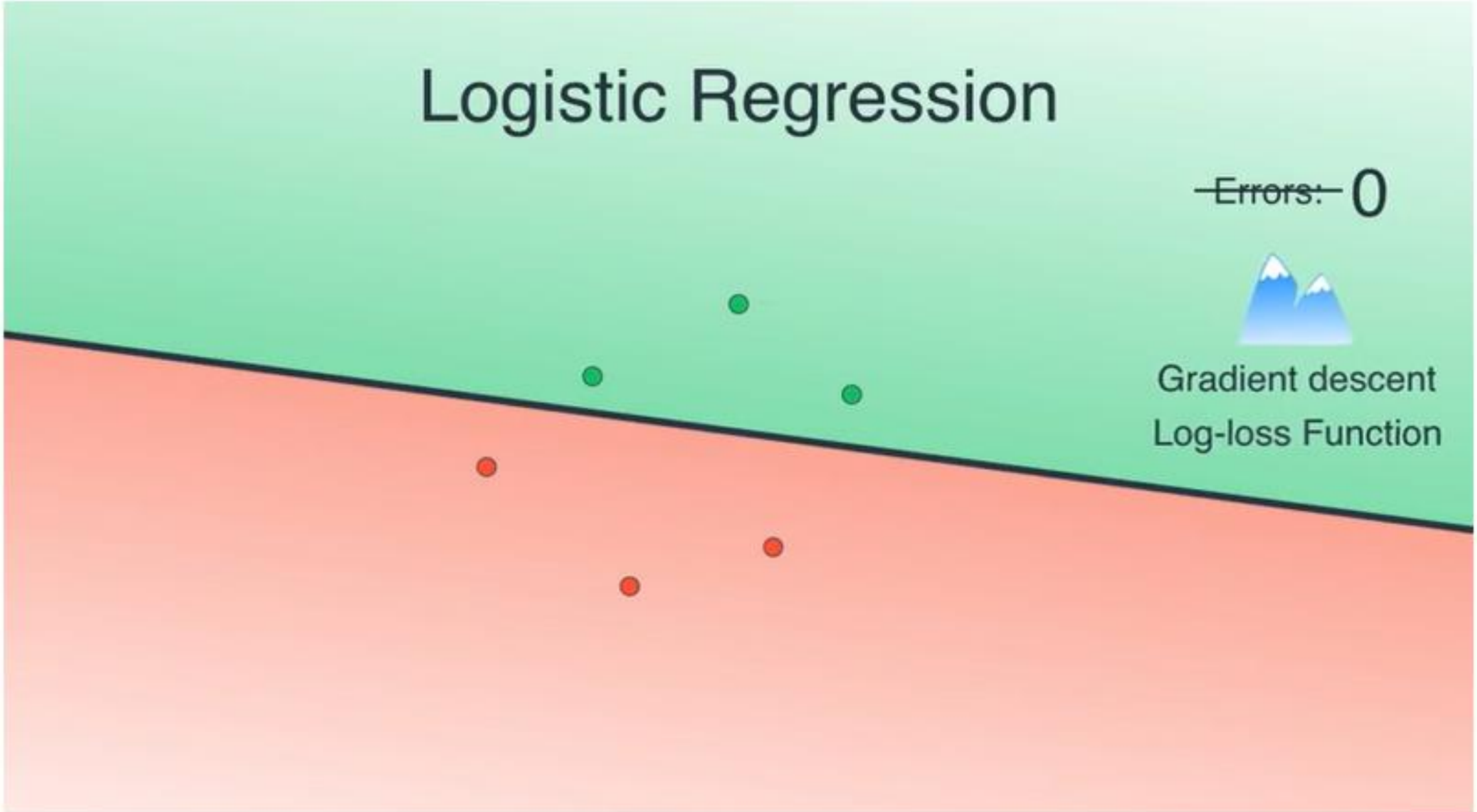


Computational Model of ANNs

- ANN is an information processing system
  - Consists of many nodes called **neurons** (processing units)
  - Signals are transmitted by connection **links**
  - Links possess an associated **weight**, which is multiplied with input signal (net input)
  - Output is obtained by applying **activations** to net input.

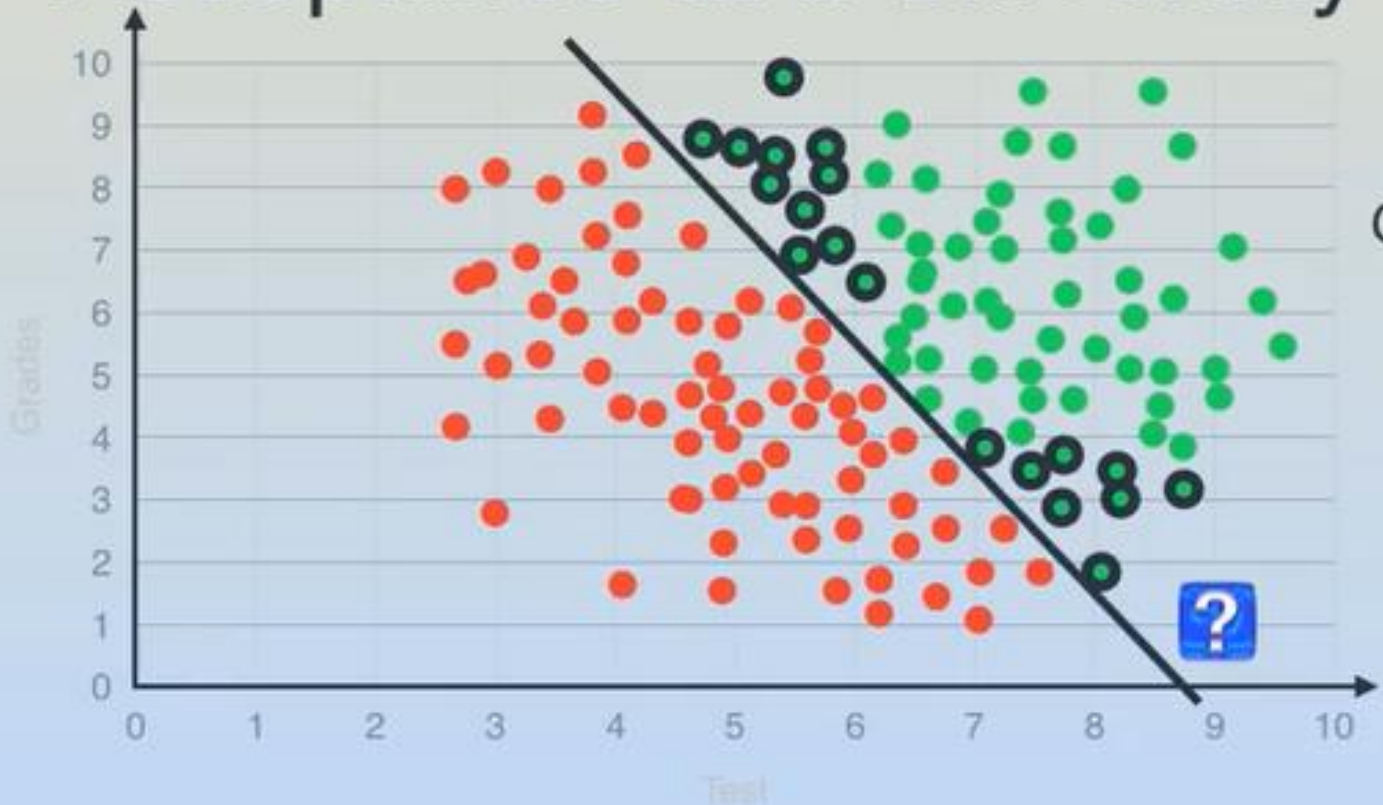


Logistic Regression



Log-loss function will give a large value for not classified data pints and a small value for correctly classified one

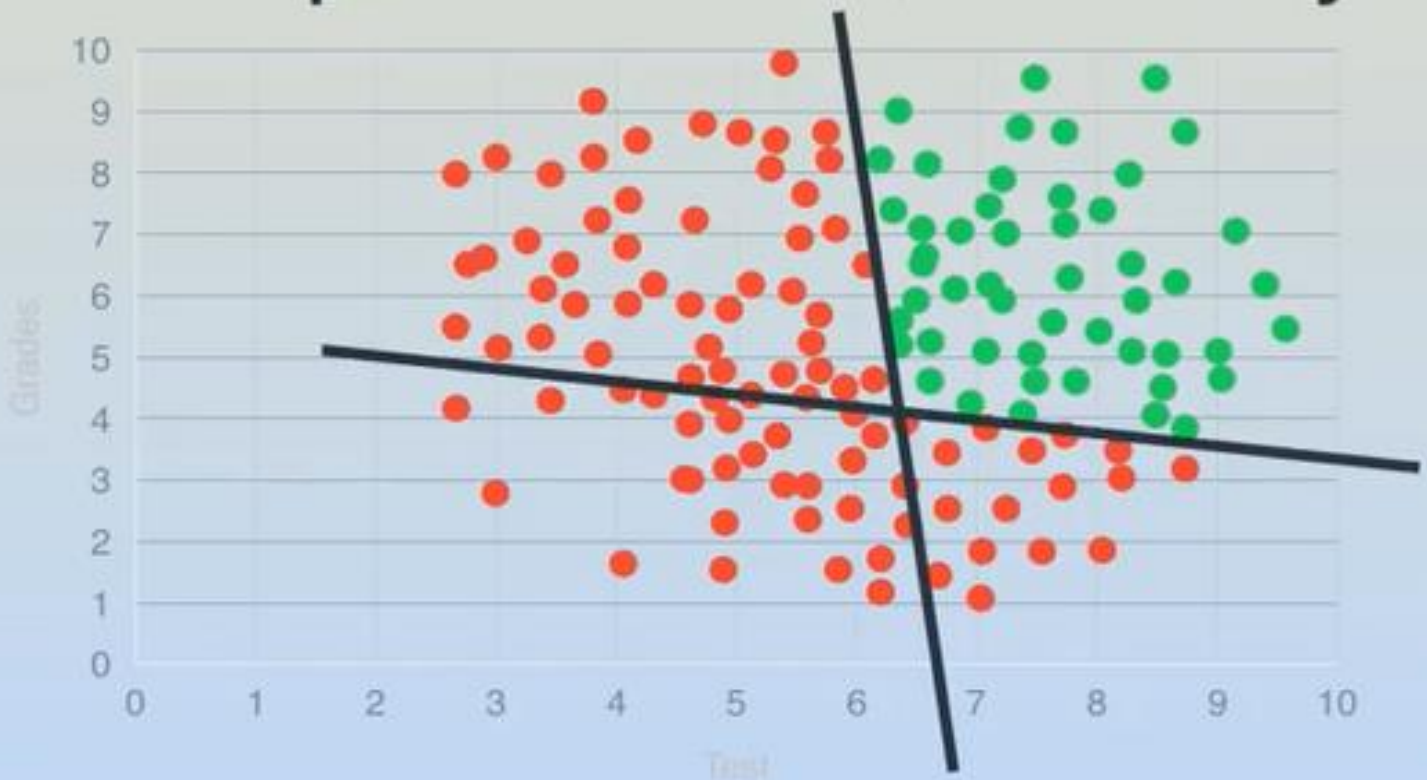
Acceptance at a University



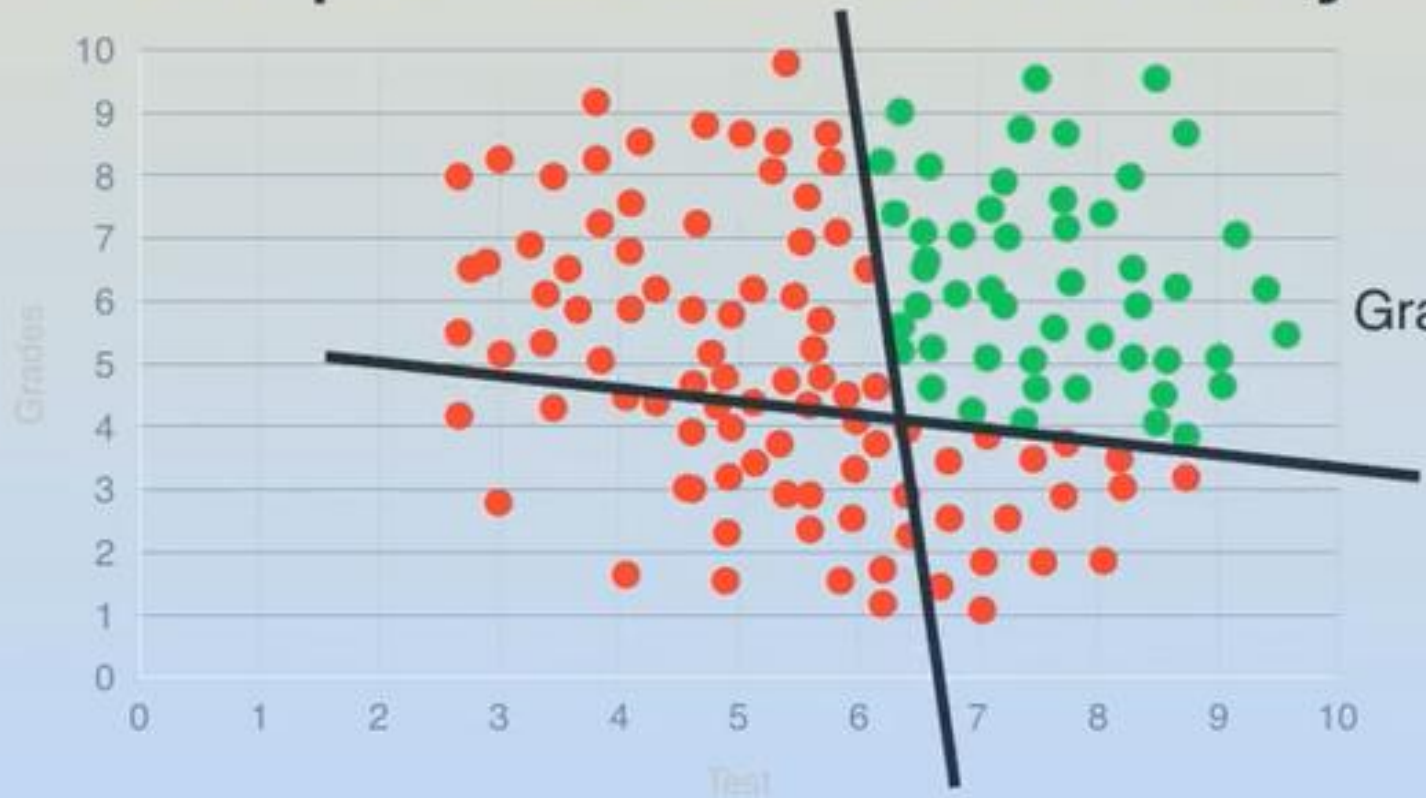
Student 4
Test: 9/10
Grades: 1/10

Acceptance at a University

Acceptance at a University

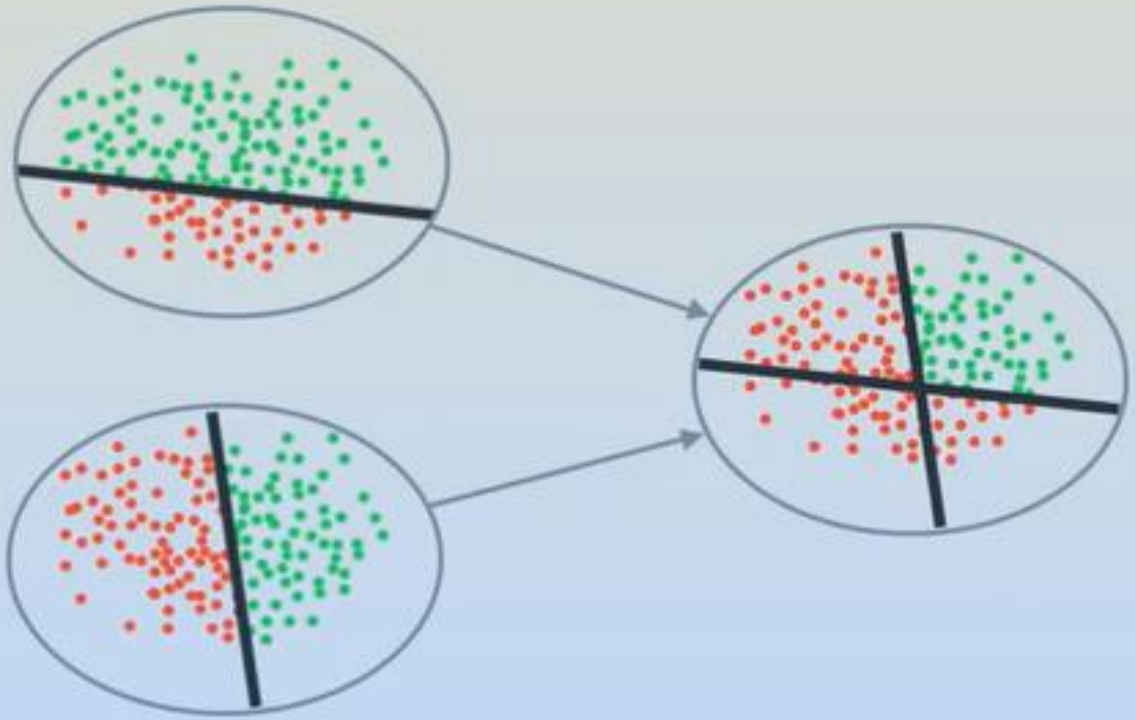


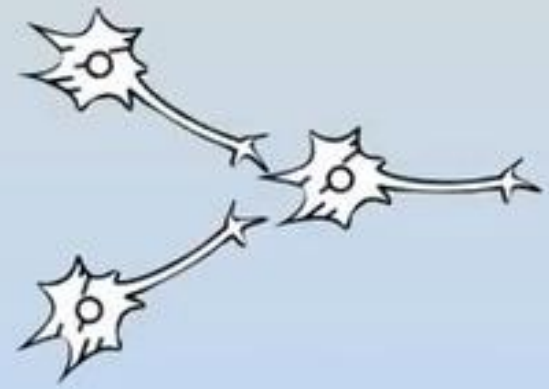
Acceptance at a University



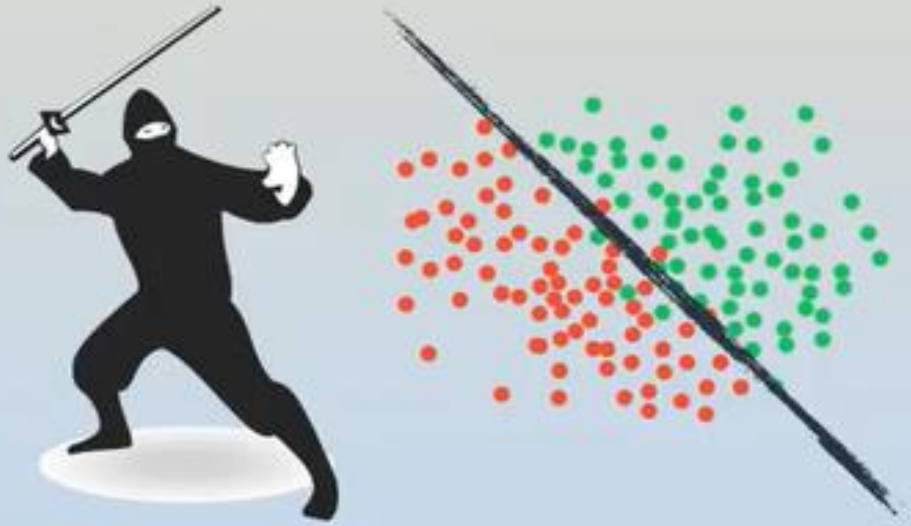
Gradient descent

Neural Network

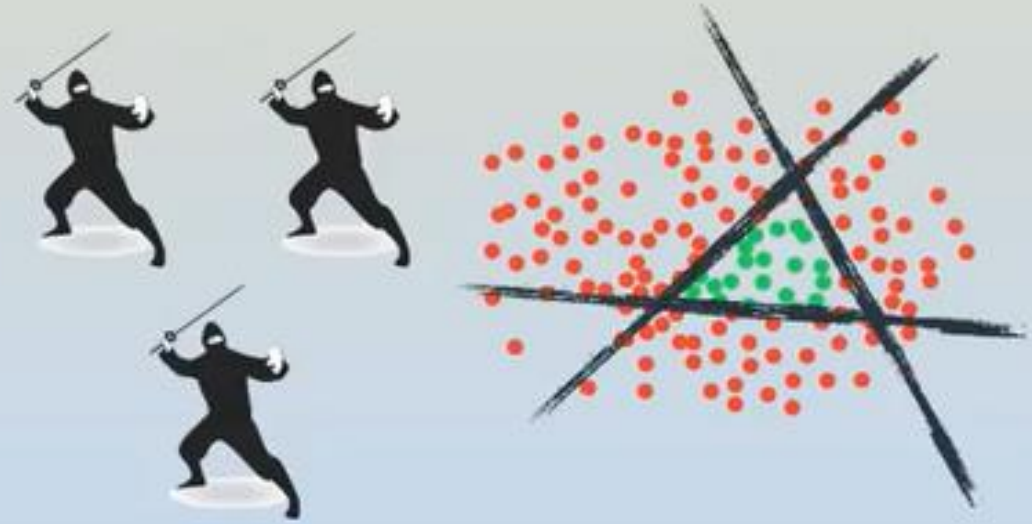




Logistic Regression & Neural Networks

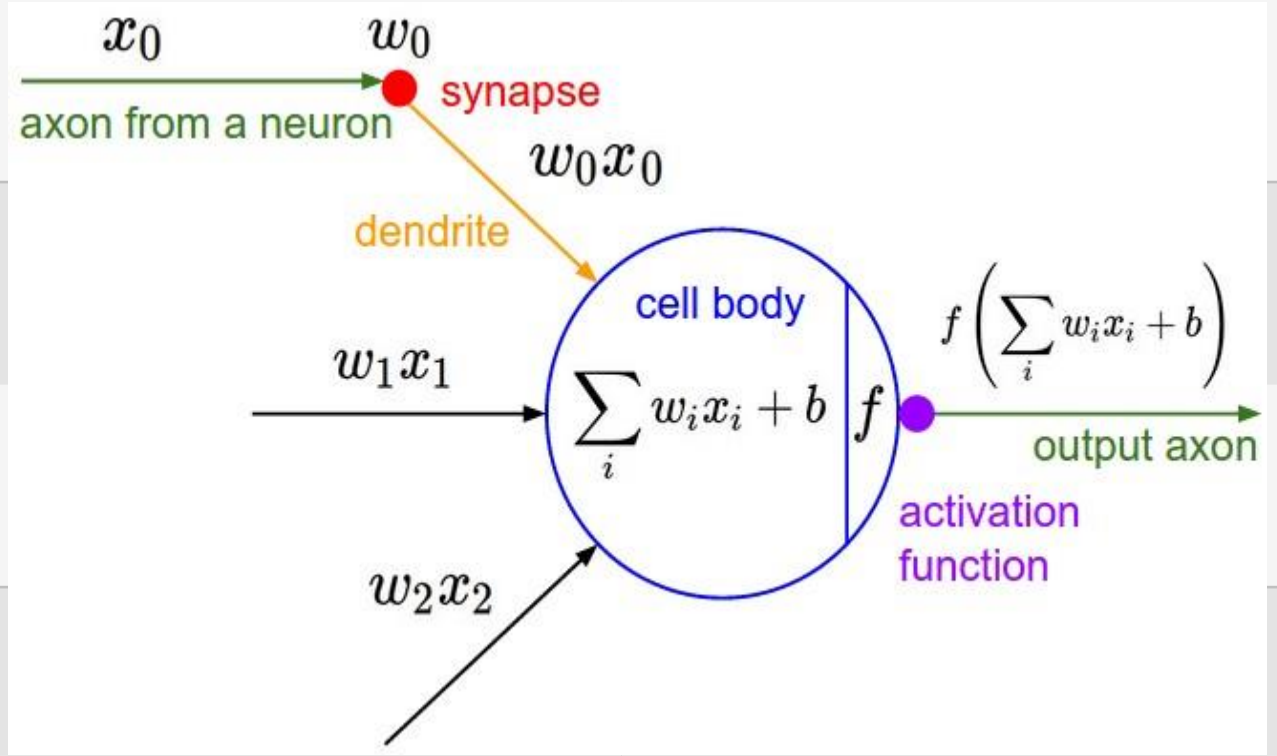


Logistic Regression



Neural Network

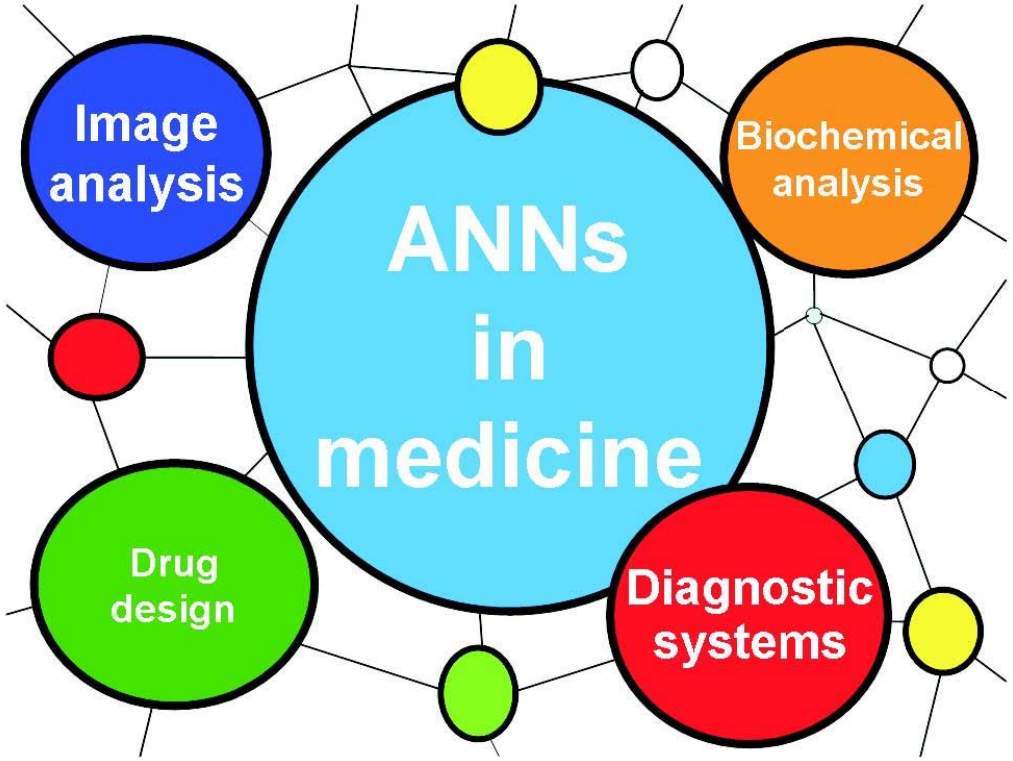
Biological Neural Network	Artificial Neural Network
Dendrites	Inputs
Cell nucleus	Nodes
Synapse	Weights
Axon	Output



Applications of ANN

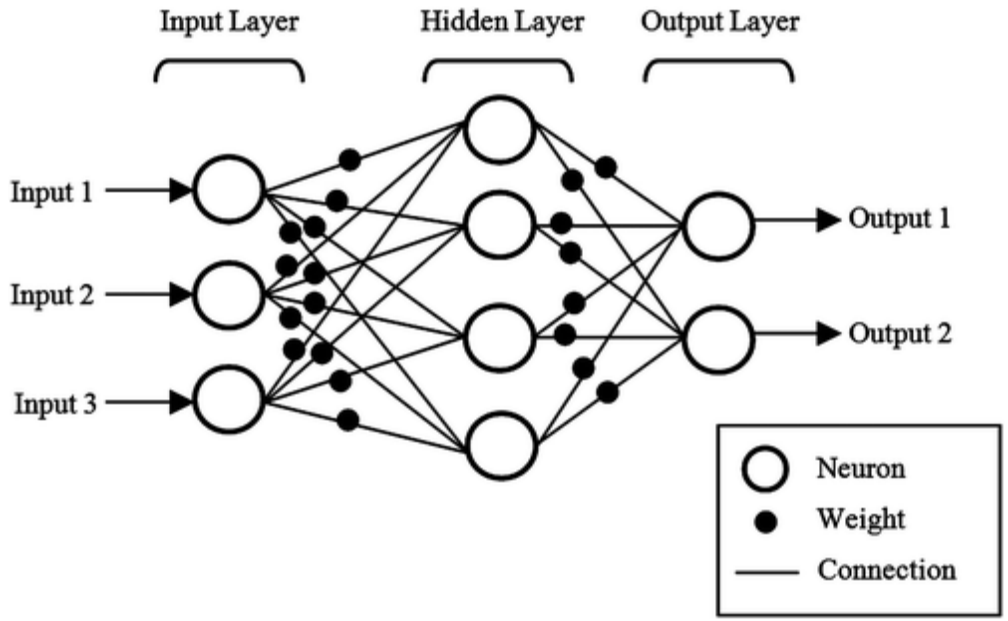
- Inspired by biological neural networks
  - Numerous advances have been made in developing intelligent systems.
- ANNs developed to solve a variety of problems in
  - pattern recognition
  - prediction
  - optimization
  - associative memory

Example: ANN in medical Applications



Artificial Neural Network

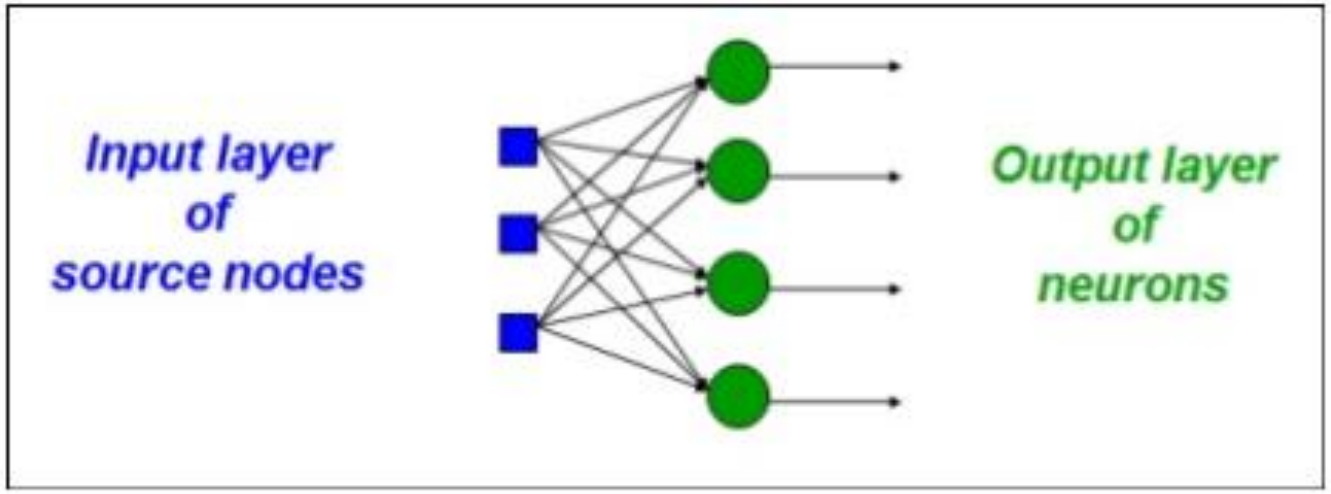
- Artificial Neural Network
  - Pool of simple processing units
  - Communication to each other over a large number of weighted connections.



ANN Models

- ANN Models - Classified
  - Single Layer ANN
  - Multi-layer ANN

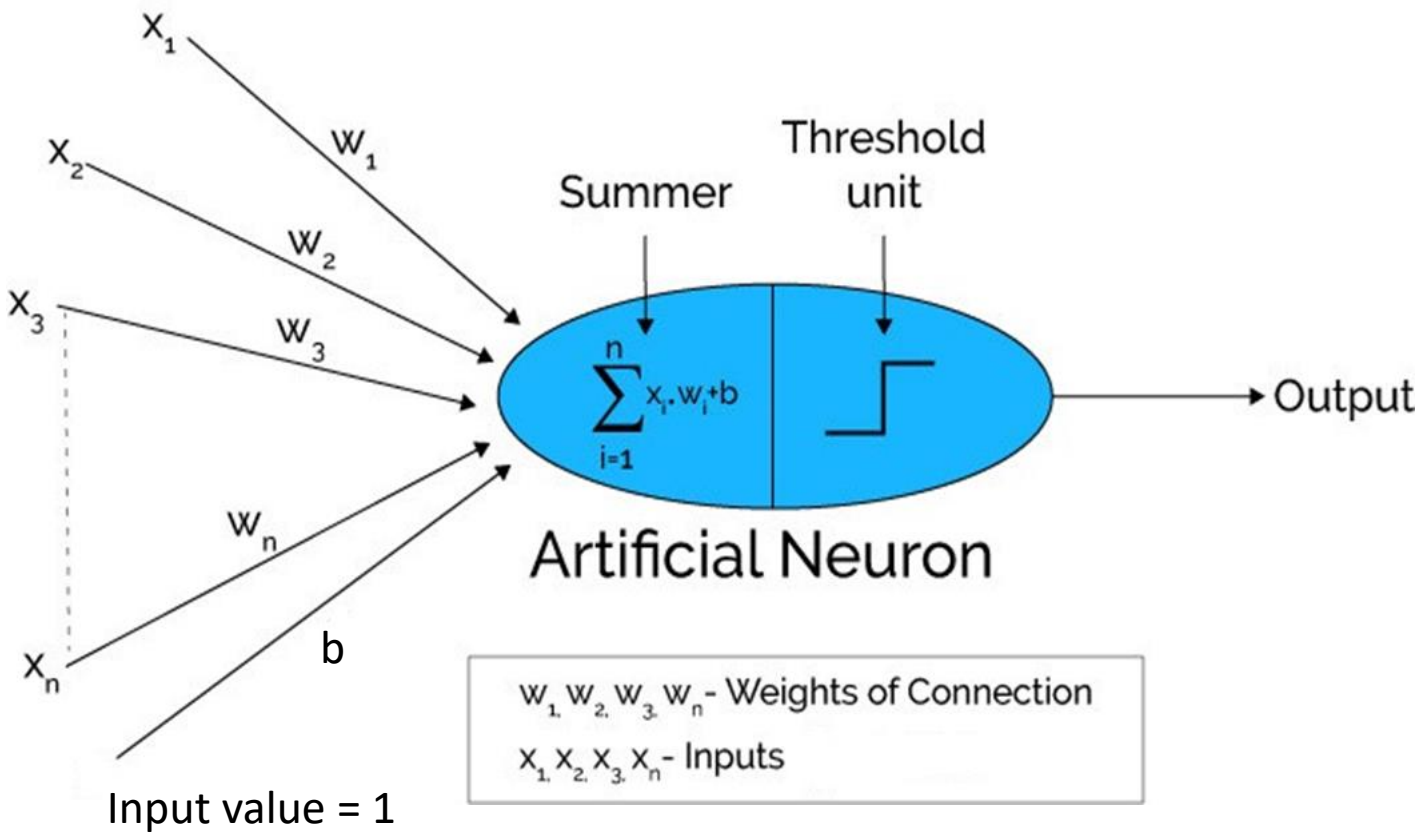
Single Layer ANN



- Only one layer of weighted interconnections
- Weighted input(s) are processed by only one layer and provide output(s)

ANN Models

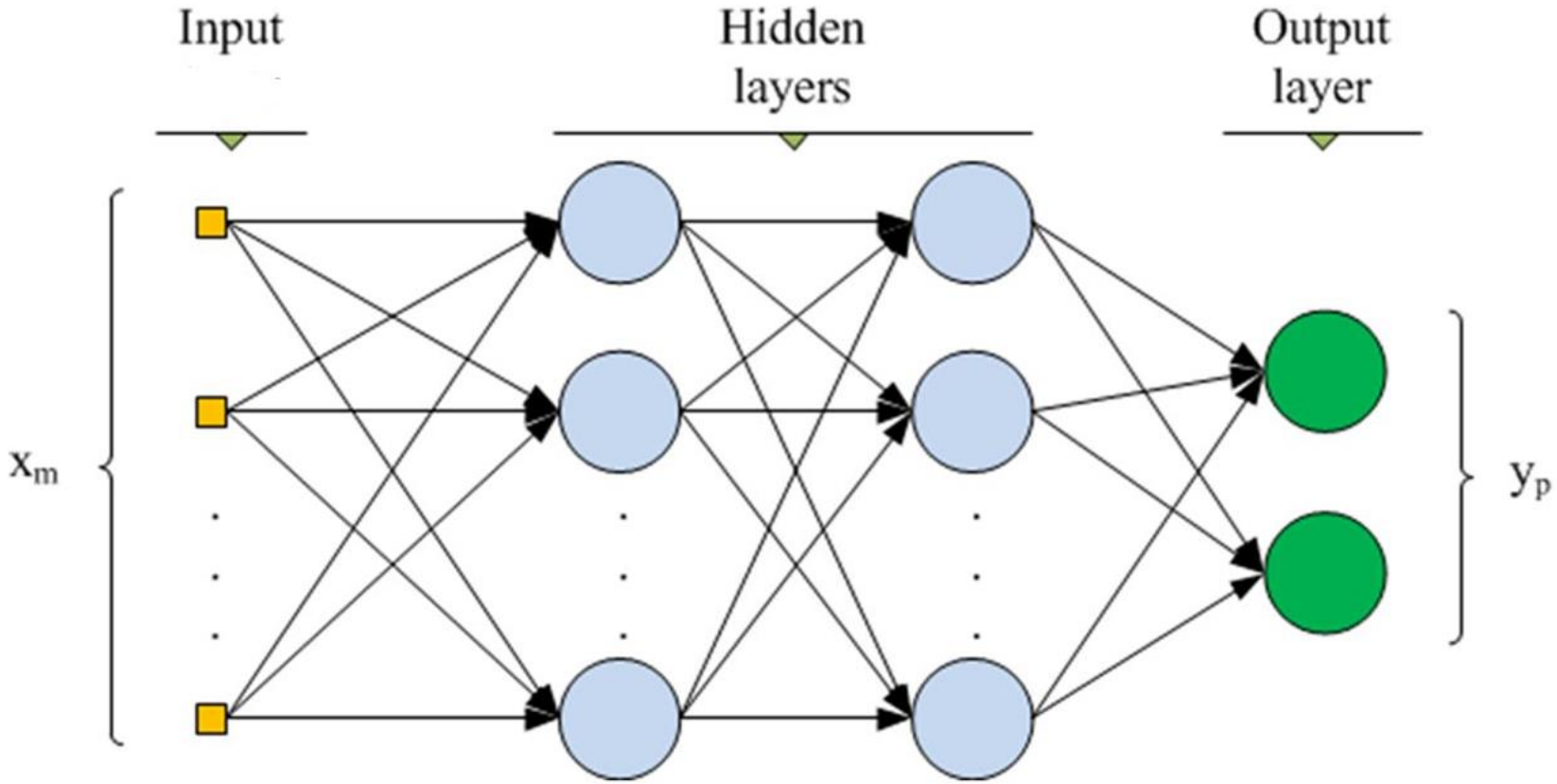
- Single Layer ANN - Example



X_1, x_2 … are input
b – bias weight
w_1, w_2 … are interconnecting weights

Multilayer ANN

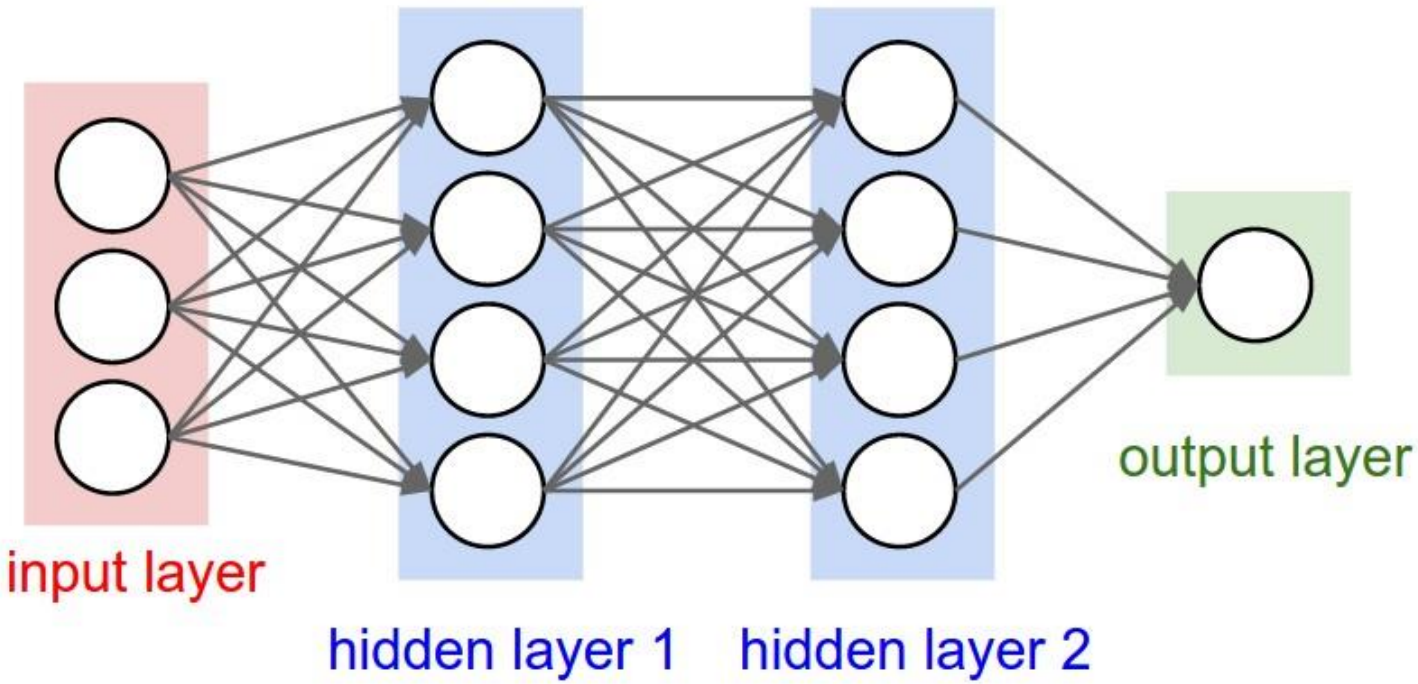
- Multilayer ANN are called layered networks
  - Input layer
  - Hidden layer(s)
  - Output layer



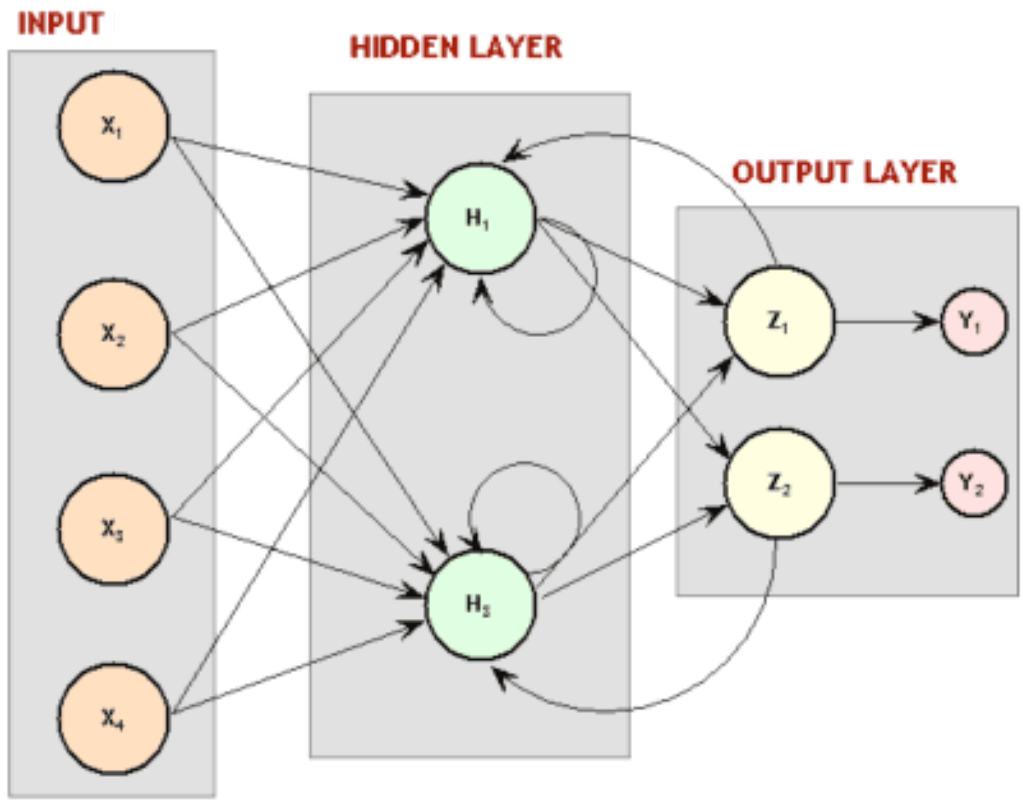
Basic building blocks of ANN

- Network Architecture
- Setting weights
- Activation function

- Network Architecture
  - Arrangement of neurons into layers
  - Connection pattern between neurons



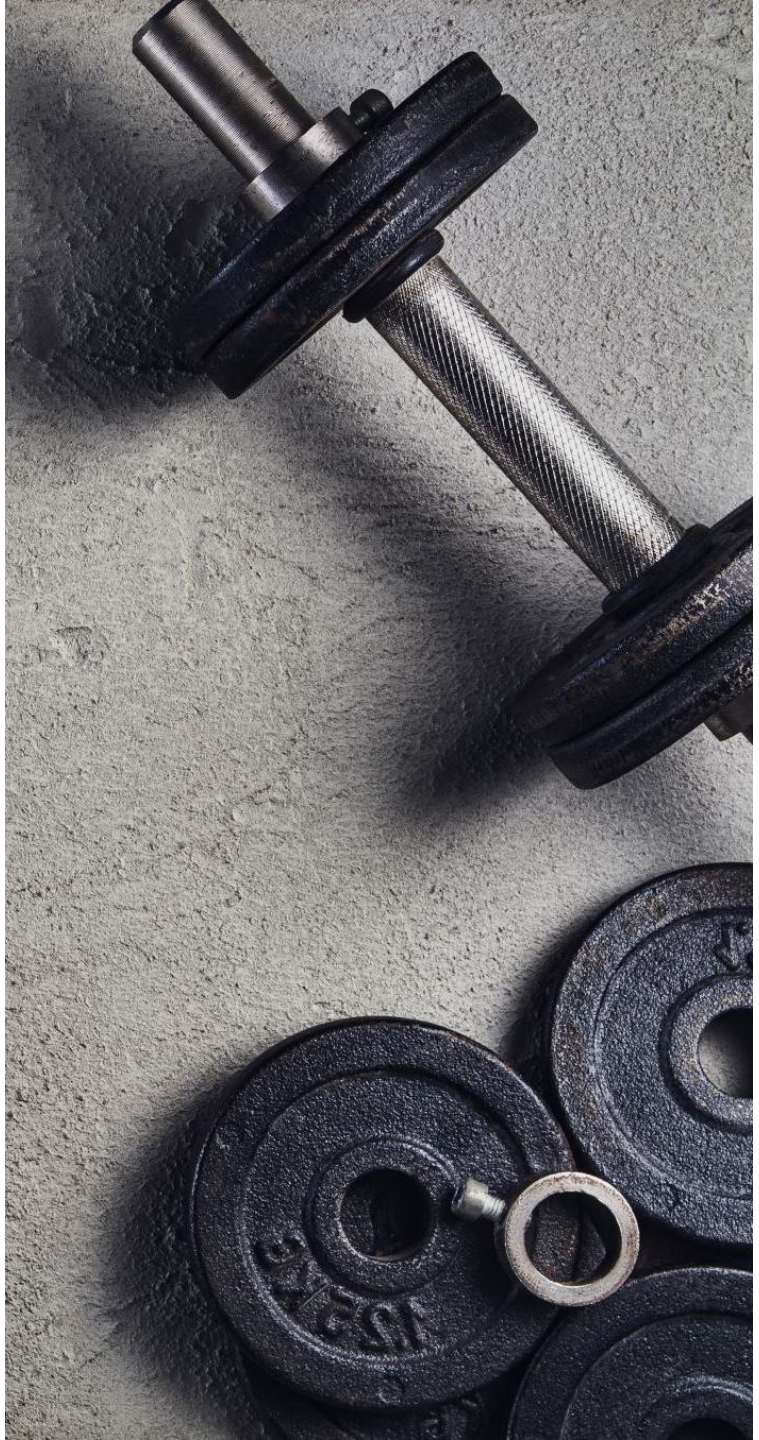
- The information is propagated from the inputs to the outputs
- Time has no role (NO cycle between outputs and inputs)



- All nodes are connected to all other nodes
- Every node is both input/ output node
- Delays are associated
- Training is more difficult
- Performance may be problematic
  - Stable Outputs may be more difficult

Setting weights

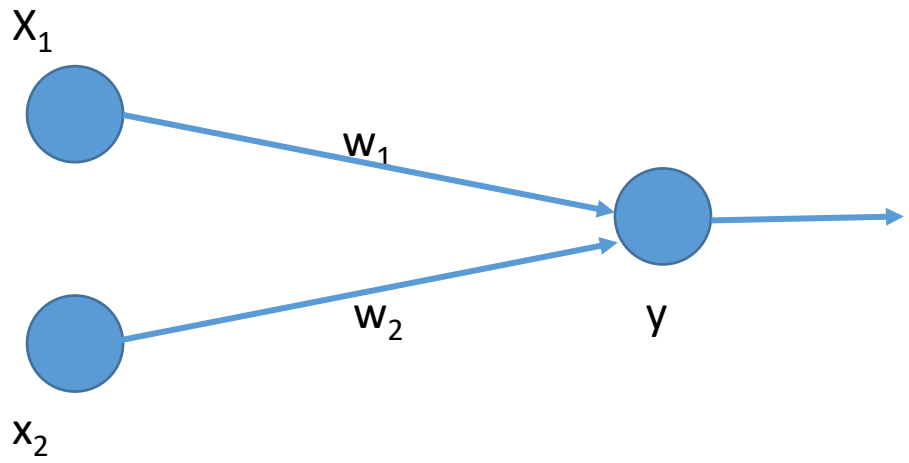
- Why weight setting in ANN?
  - Setting the values for weights – enable learning/ training
- Training
  - Enable ANN model to learn
  - Process of modifying the weights in the network to achieve the expected output
- Learning
  - Internal process when the network is trained



Artificial Neural Network (ANN) Terminologies

1. Weights

- Weight is an information used by the neural net to solve the problem
- Weights can set to zero or can be calculated by some methods



Net input = Net = $x_1 w_1 + x_2 w_2$

Net input = Net = $\sum_{i=1}^{n} x_i w_i$

- x_1 – input signal-1
- x_2 – input signal-2
- y – neuron
- w_1 – weight connecting X_1 to y
- w_2 –weight connecting X_2 to y

Artificial Neural Network (ANN) Terminologies

2. Activation functions

- Used to calculate the output response of a neuron.
- Sum of the weighted input signal is applied with an activation to obtain the response.
- For neurons in the same layer same activation functions are used.

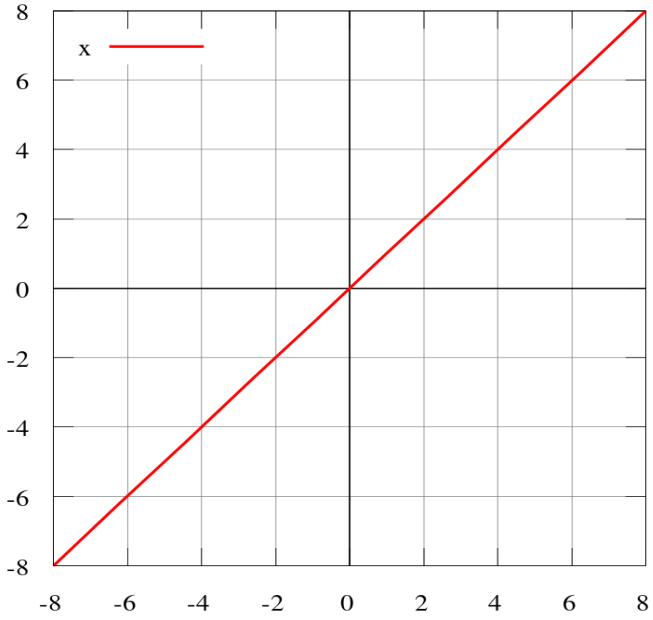
- Activation function
  - Linear
  - Non-linear (used in multilayer net)

Artificial Neural Network (ANN) Terminologies

2. Activation functions

Identity (Linear) Function:

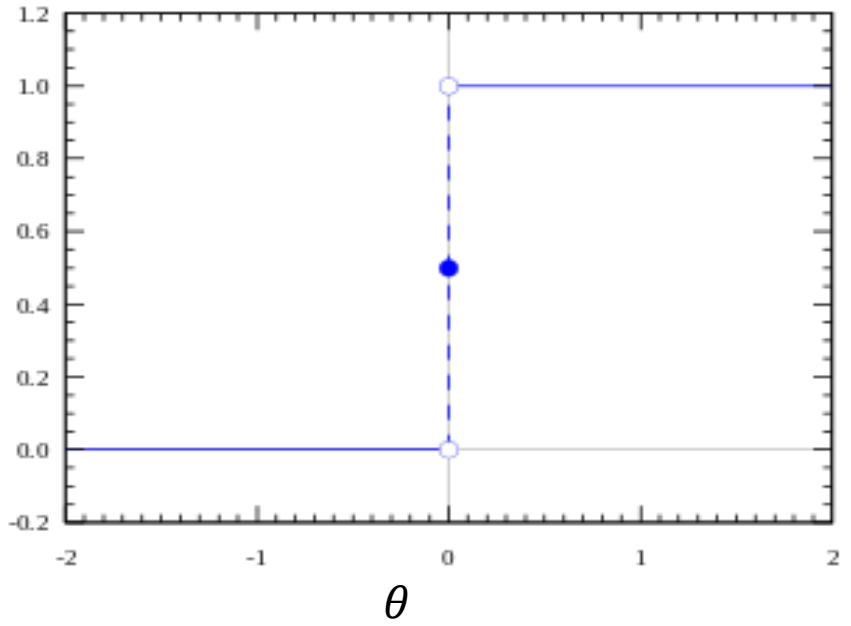
f(x) = x, for all x



Binary Step Function

$$f(x) = \begin{cases} 1; if\ f(x) \geq \theta \\ 0; if\ f(x) < \theta \end{cases}$$

- $where\ \theta\ is\ threshold$
- Single layer nets uses binary step (threshold) function.



Artificial Neural Network (ANN) Terminologies

2. Activation functions

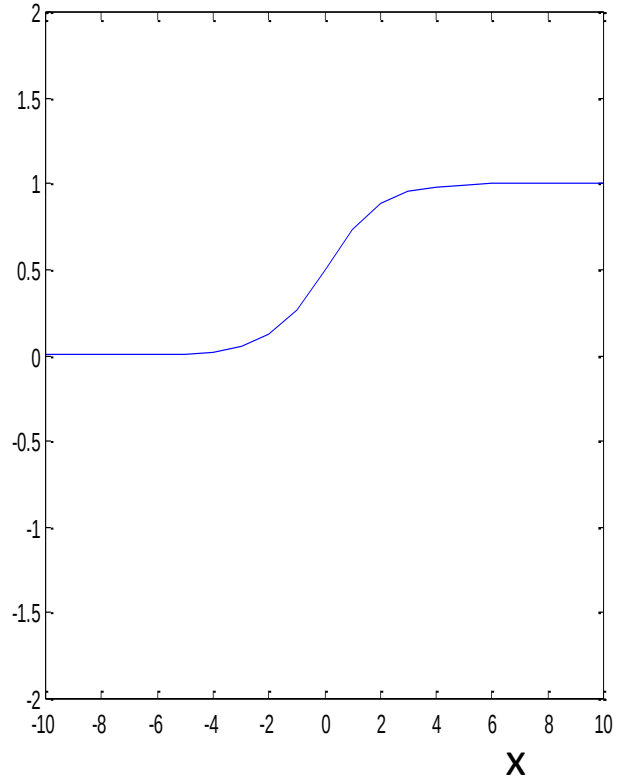
- **Sigmoidal function**
  - S-shaped curves
  - Hyperbolic & logistic functions
  - Used in multi layer nets
  - Example: back propagation net
  - Types
    - Binary Sigmoidal Function
    - Bipolar Sigmoidal Function

Binary Sigmoidal (Logistic) Function

It ranges between 0 to 1.

$$f(x) = \frac{1}{1+\exp(-\sigma x)}$$

Where σ – steepness parameter.



Artificial Neural Network (ANN) Terminologies

2. Activation functions

- Bipolar Sigmoidal Function
  - Range: +1 and -1
  - Called hyperbolic tangent function

$$b(x) = 2f(x) - 1$$

$$b(x) = 2 \times \frac{1}{1 + \exp(-\sigma x)} - 1$$

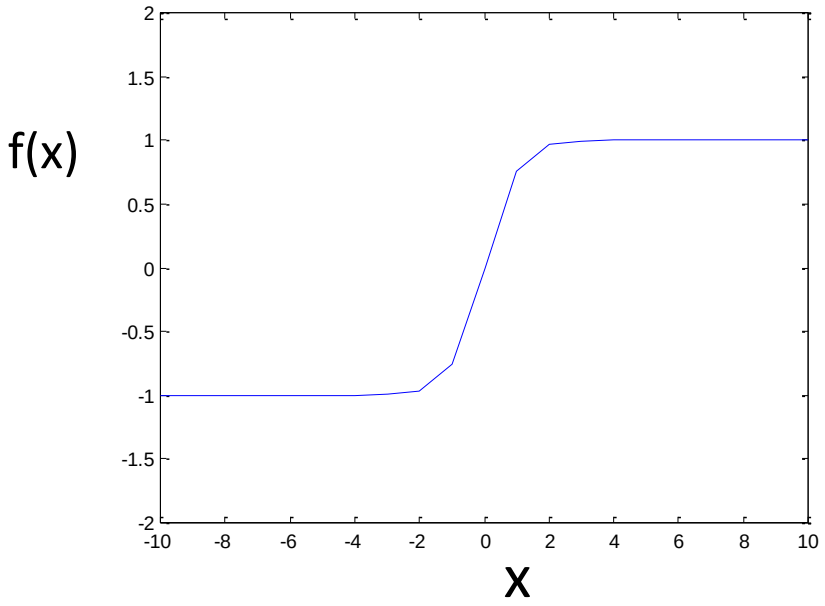
$$b(x) = \frac{2 - 1 - \exp(-\sigma x)}{1 + \exp(-\sigma x)} - 1$$

$$b(x) = \frac{1 - \exp(-\sigma x)}{1 + \exp(-\sigma x)}$$

In binary Sigmoidal Function

$$f(x) = \frac{1}{1 + \exp(-\sigma x)}$$

Where σ – steepness parameter.



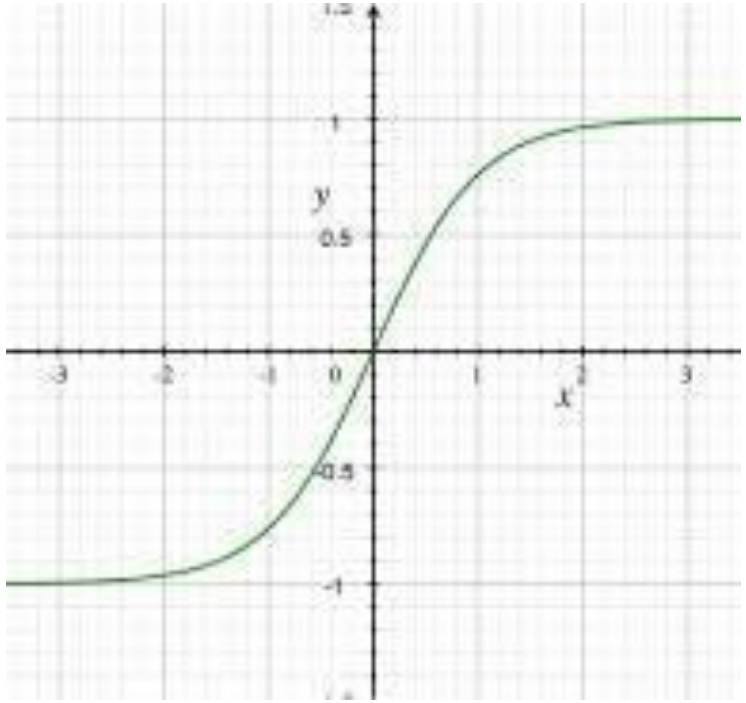
Vanishing Gradient

- Sigmoid activation function has a problem of “Vanishing Gradient”.
- Vanishing Gradient is a significant problem as a large number of inputs are fed to the neural network and the number of hidden layers increases, the **gradient or derivative becomes close to zero** thus leading to **inaccuracy** in the neural network.

Tanh

- tanh function is a non-linear and differentiable function similar to the sigmoid function, but output values range from -1 to +1.
- Problem with the tanh activation function: It is slow, and the vanishing gradient problem persists.

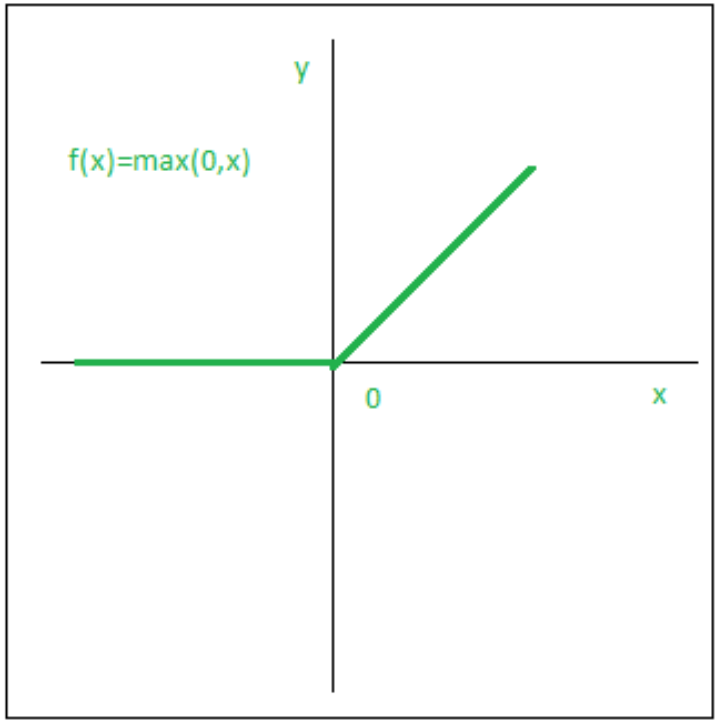
$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$



Rectified linear unit (**ReLU**)

- It is the most widely used activation function in CNN.

$$f(x) = \max(0, x)$$



Advantage

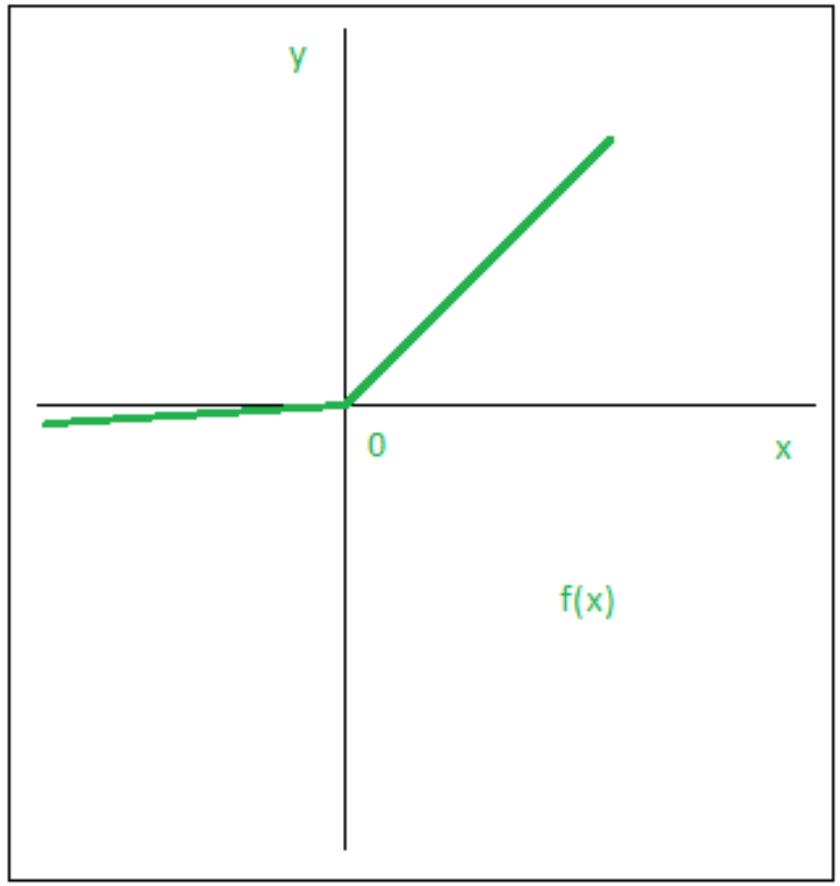
1. It does not activate all the neurons at the same time. i.e. if the input is negative, it will convert it to zero and the neuron does not get activated.

Leaky ReLU

- It is an improved version of the ReLU function.
- Instead of defining the ReLU function as 0 for x less than 0, we define it as a small linear component of x.

- It can be defined as:

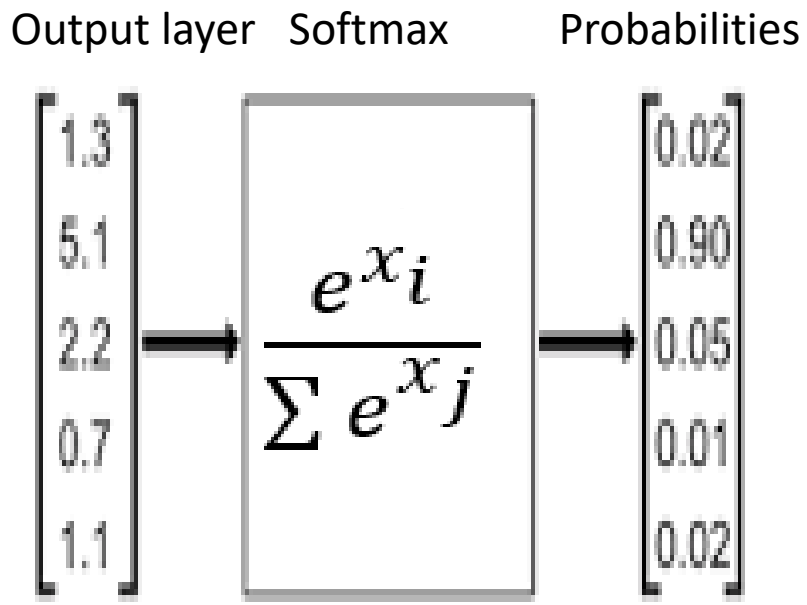
$$f(x) = \begin{cases} ax, & x < 0 \\ x, & \text{otherwise} \end{cases}$$



Softmax function

- The softmax function is different from other activation functions as it is placed at the last layer to normalize the output.
- We can use other activation functions in combination with Softmax to produce the output in probabilistic form.
- It is used in multiclass classification and generates an output of probabilities whose sum is 1.
- The range of output lies between 0 and 1.

$$softmax(x_i) = \frac{e^{x_i}}{\sum e^{x_j}} \text{ for all j}$$

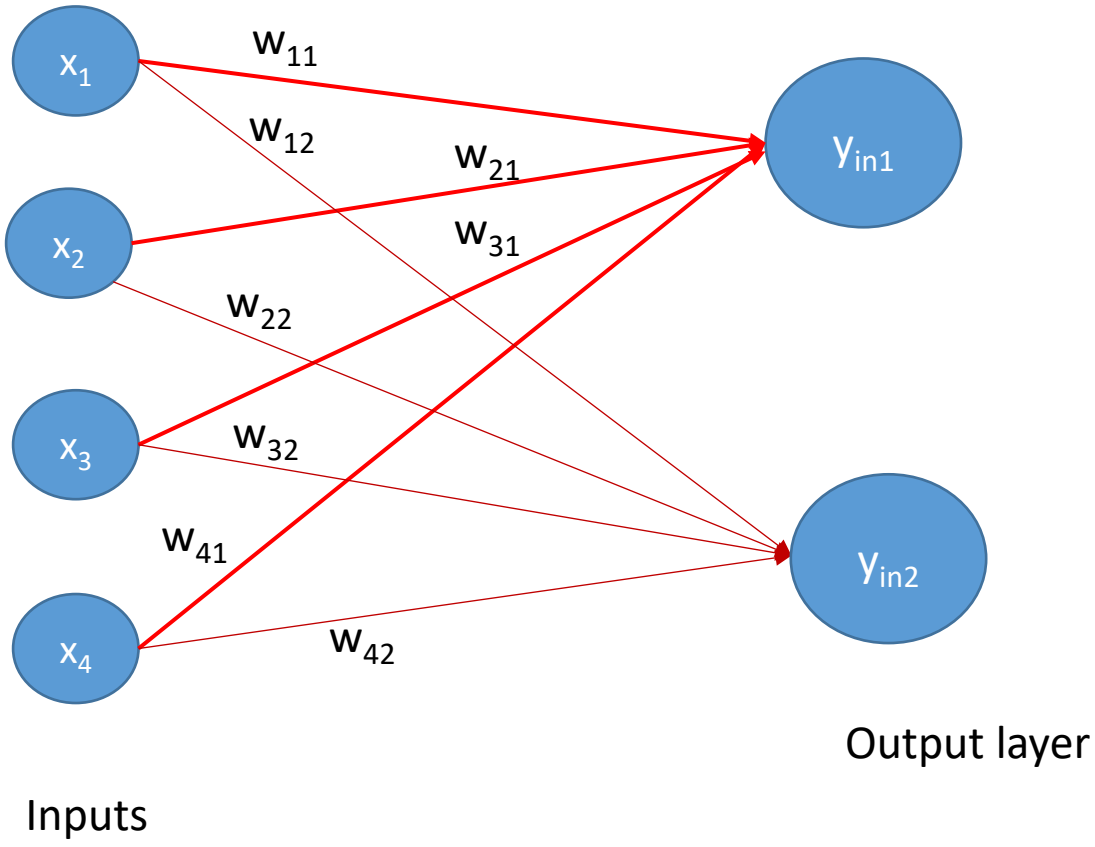


Artificial Neural Network (ANN) Terminologies
3. Calculation of Net Input using matrix multiplication method

- If the weights are given as $W = (w_{ij})$ in a matrix form

- The net input to output $y_{inj} = x_i * w_{ij}$

$$y_{inj} = \sum_{i=1}^{n} x_i w_{ij}$$

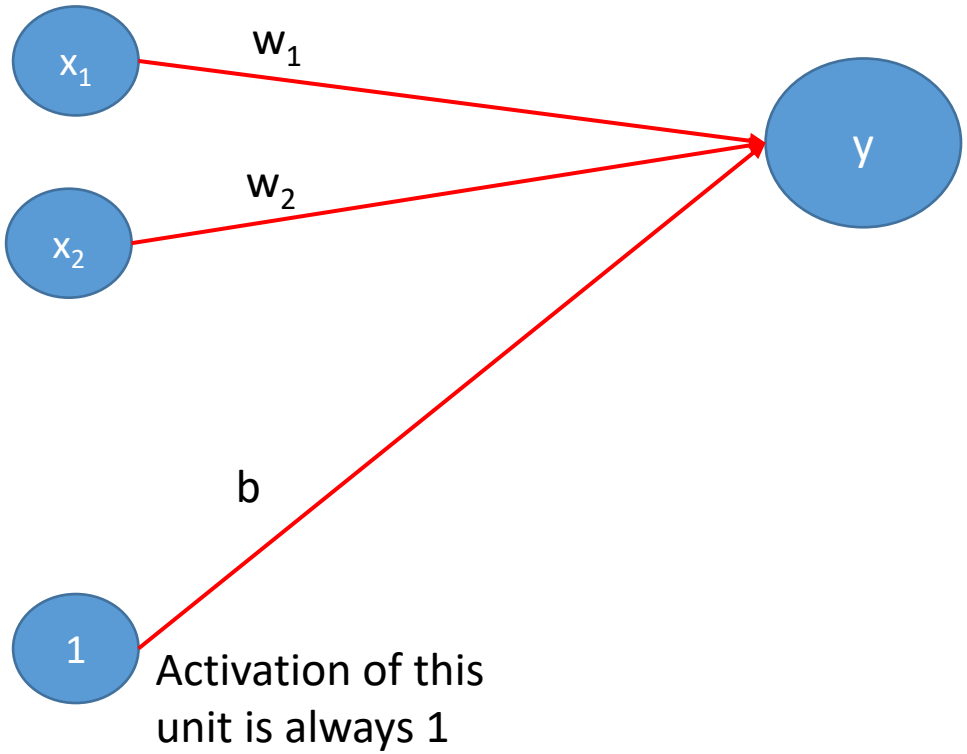


Artificial Neural Network (ANN) Terminologies

4. Bias

- Weight on a connection from a unit whose activation is always 1.
- Increasing bias increase net input.

$$\text{Net} = b + \sum_{i=1}^{n} x_i \, w_i$$



Artificial Neural Network (ANN) Terminologies

5. Threshold

- Used in calculating the activations of the given net
- Based on the threshold, output is calculated.
- Activation function is based on the value of θ

$$y = f(Net) = \begin{cases} +1;\ if\ Net \geq \theta \\ -1; if\ Net < \theta \end{cases}$$

$where\ \theta\ and\ \theta_j\ are\ thresholds$

McCulloch-Pitts Neuron Model

- The most fundamental unit of deep neural networks is a *perceptron*.
- But the very first step towards the *perceptron* we use today was taken in 1943 by McCulloch and Pitts, by mimicking the functionality of a biological neuron.

McCulloch-Pitts Neuron

- The first computational model of a neuron was proposed by Warren MuCulloch (neuroscientist) and Walter Pitts (logician) in 1943.

McCulloch-Pitts Neuron Model

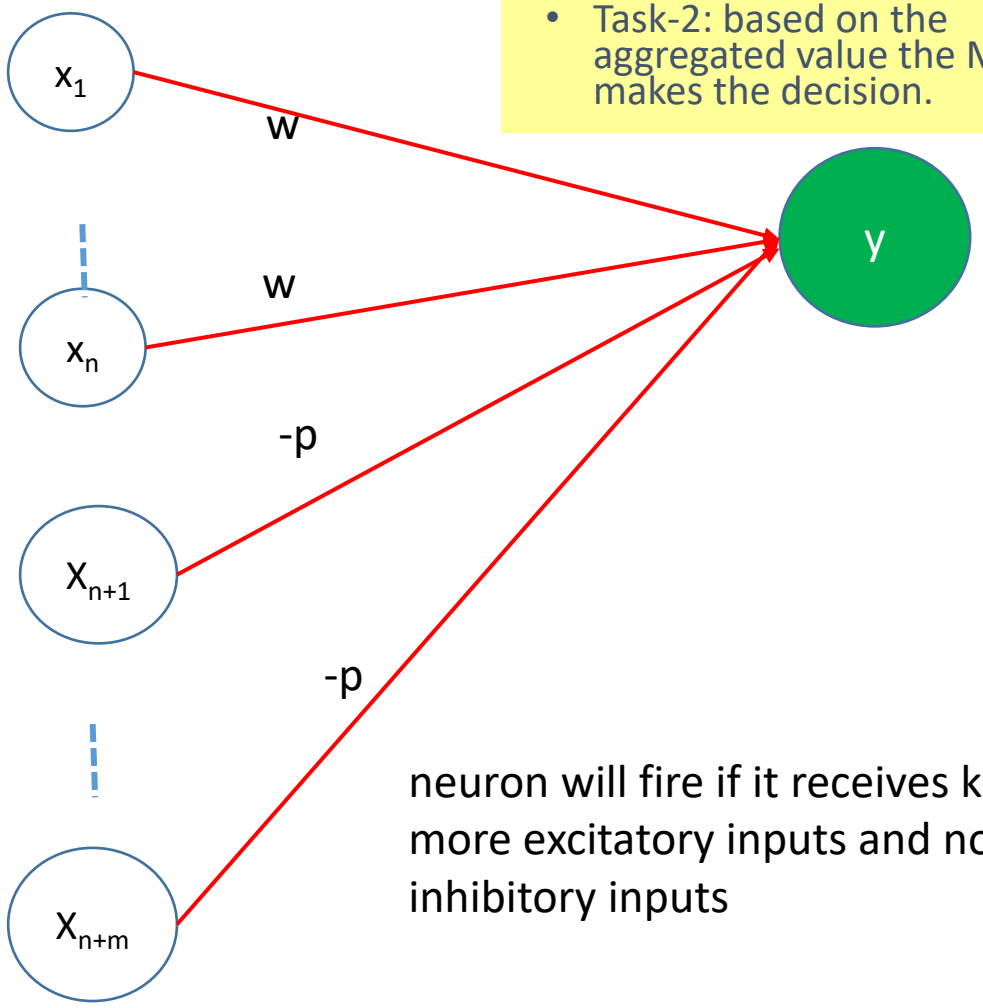
- Here, y is McCulloch-Pitts neuron
- Receives signal from any number of neurons
- Connection weights from x_1 .. x_n are excitatory, denoted by w.
- Connection weights from x_{n+1} .. X_{n+m} are inhibitory, denoted by –p.
- McCulloch-Pitts neuron y has the activation function

$$f(y_{in}) = \begin{cases} 1; if\ f(y_{in}) \geq \theta \\ 0; if\ f(y_{in}) < \theta \end{cases}$$

- Where, θ – threshold and y_{in} - net input to y

- *Excitatory* inputs have maximum effect on the decision making.
- An inhibitory input will always be 0.
- Excitatory inputs are NOT the ones that will make the neuron fire on their own, but they might fire it when combined.

- M-P performs two tasks.
  - Task-1: performs an aggregation of all the inputs
  - Task-2: based on the aggregated value the M-P net makes the decision.



neuron will fire if it receives k or more excitatory inputs and no inhibitory inputs

Examples

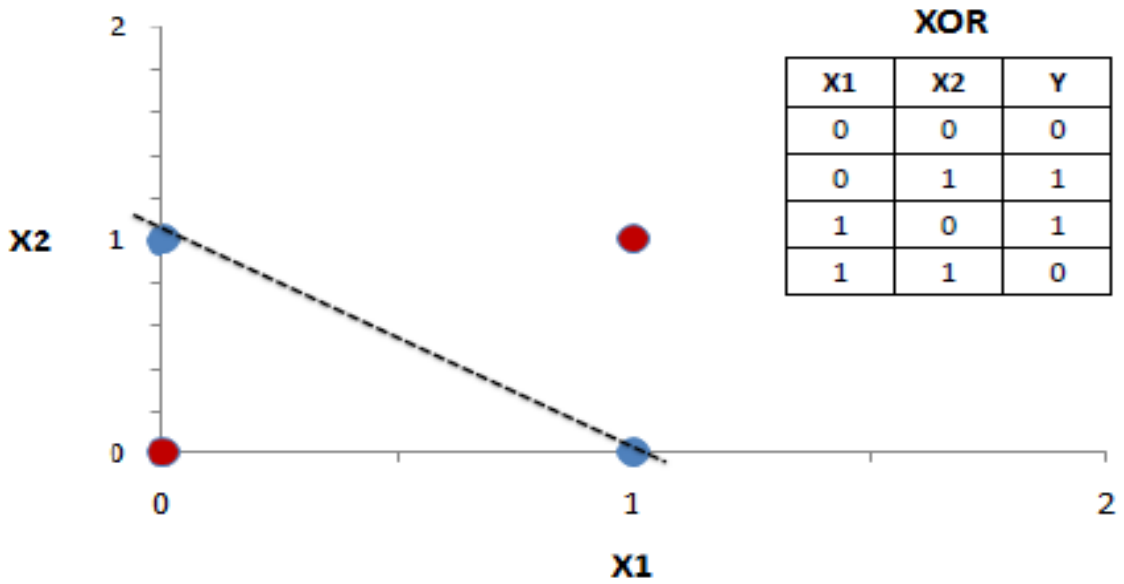
- Generate the output of logic AND function by McCulloch-Pitts neuron model.
- Generate the output of logic OR function by McCulloch-Pitts neuron model.
- Generate the output of logic NOT function by McCulloch-Pitts neuron model.
- Generate the output of logic NAND/ NOR function by McCulloch-Pitts neuron model.

Limitations Of M-P Neuron

- Applies only to Boolean inputs.
- We always need to hand code the threshold
- All inputs are holding equal priority. What if we want to assign more importance to some inputs?
- What about functions which are not linearly separable? Say XOR function.

Perceptron Networks

- Frank Rosenblatt - 1962
- Minsky and Papert -1988 - limitations
- Learning rule:
  - Iterative weight adjustment (powerful technique)
- Training in perceptron
  - Continue until no error (minimum) occurs.
- Perceptron Net
  - Solve – classification problem



XOR

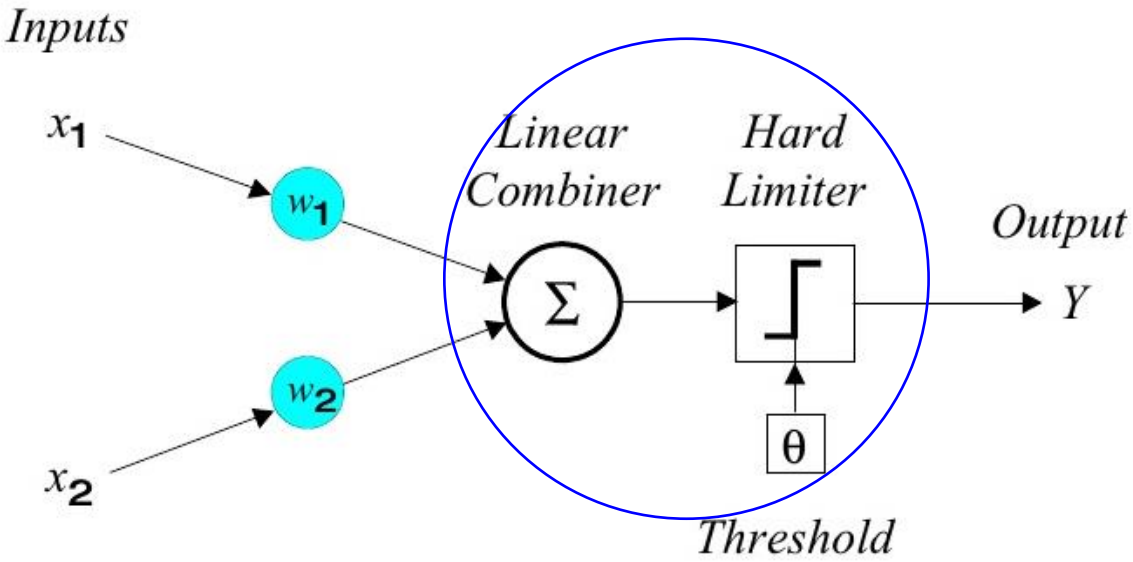
X1	X2	Y
0	0	0
0	1	1
1	0	1
1	1	0

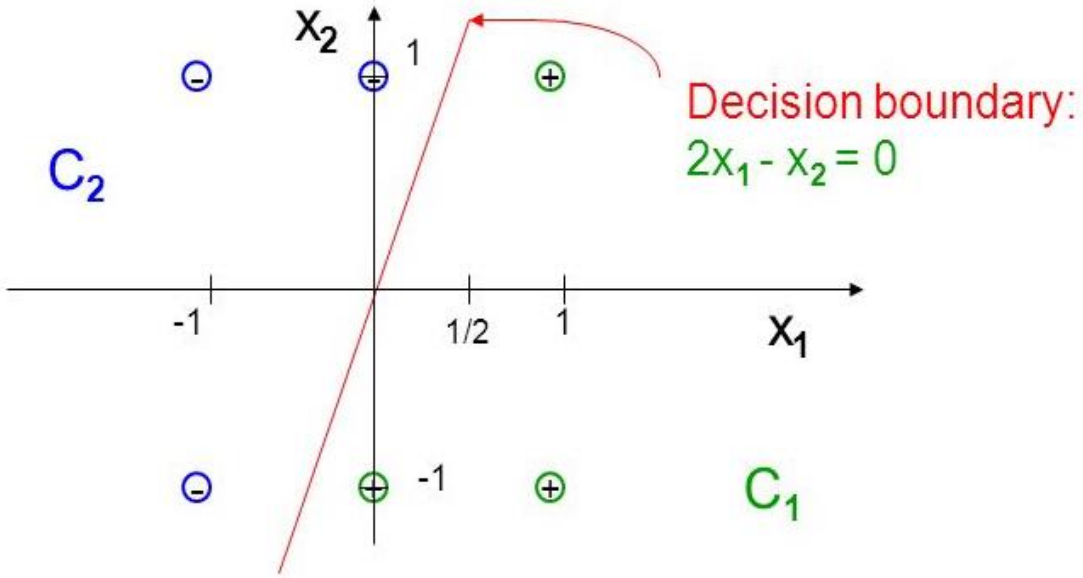
Perceptron Networks

- Types of perceptron:
  - Single Layer perceptron
  - Multilayer perceptron

Single Layer Perceptron

- Used for classification of patterns that are linearly separable.
- It consists of single neuron, that adjust the weights and bias.
- Rosenblatt found
  - If the patterns used to train the perceptron are drawn from the linearly separable class, the perceptron algorithm converges
  - Positions the decision surface in the form of a hyper plane between two classes.



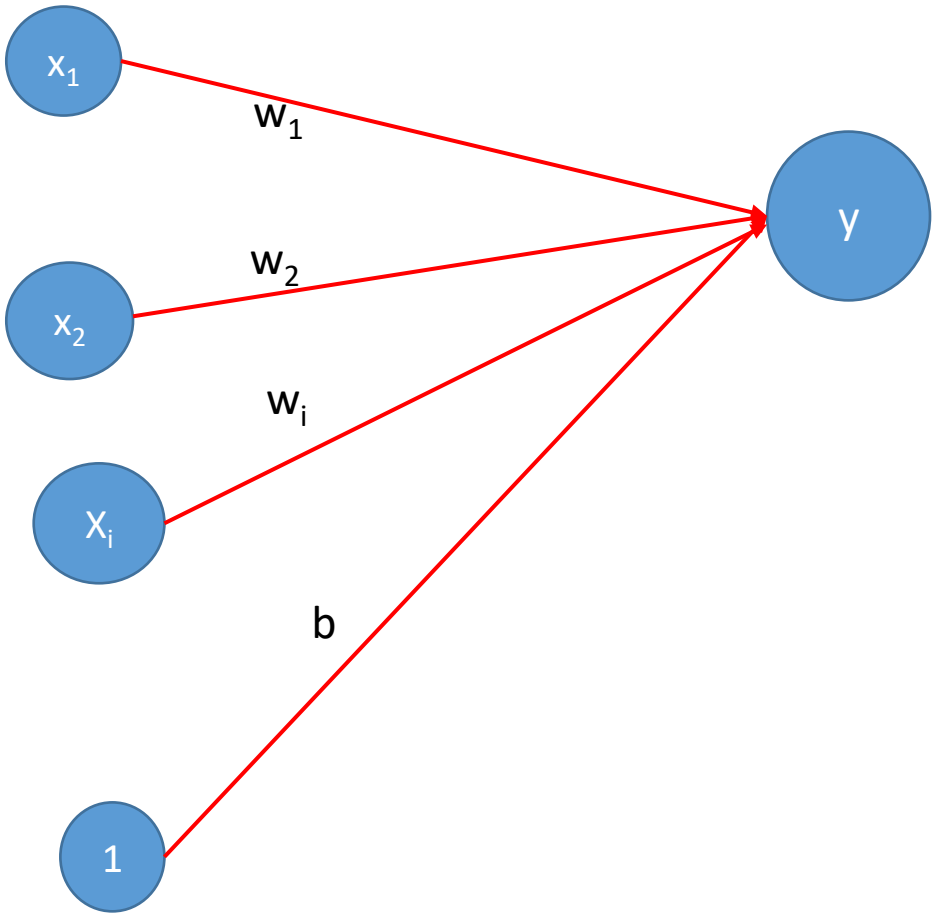


Single Layer Perceptron

Algorithm

1. Initialize weights and bias (set to ZERO) and set the learning rate α (0 to 1)
2. While stopping condition is not TRUE, repeat steps 3 to 7
3. For each training pair S: t do steps 4 to 6
4. Set activation of input layers $x_i = s_i$ for all i = 1 to n

Architecture



Single Layer Perceptron Architecture

5. Compute output response

$$y_{in} = b + \sum_{i=1}^{n} x_i \; w_i$$

Activation function

$$Y = f(y_{in}) = \begin{cases} +1 & if \; y_{in} > \theta \\ 0 & if \; y_{in} = \theta \\ -1 & if \; y_{in} < \theta \end{cases}$$

6. If the output response is not equal to target, then update the weights and bias

If $t \neq y$

$$w_{i(new)} = w_{i(old)} + \alpha t x_i$$
$$b_{(new)} = b_{(old)} + \alpha t$$

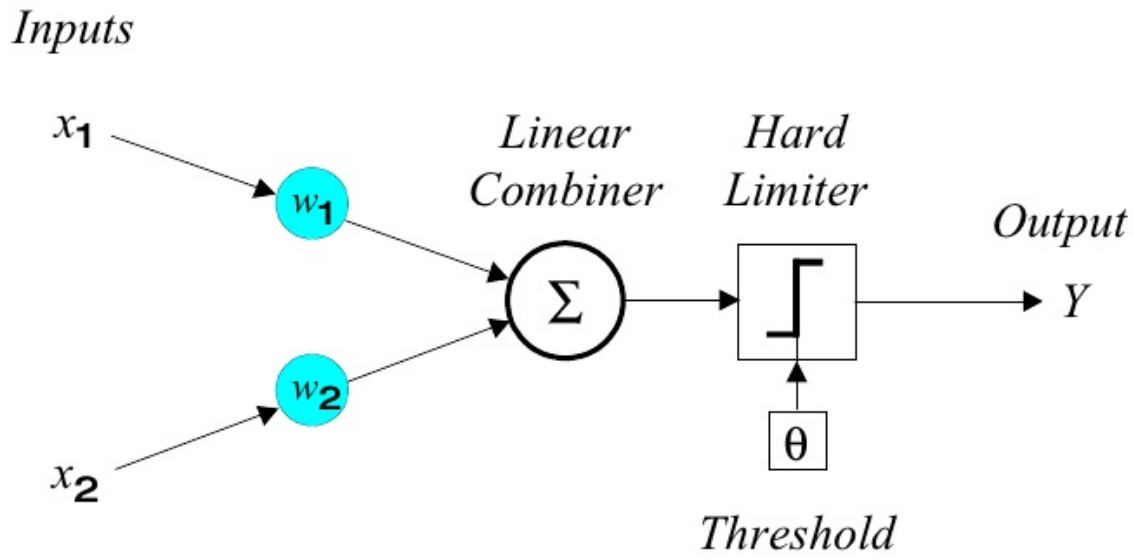
else

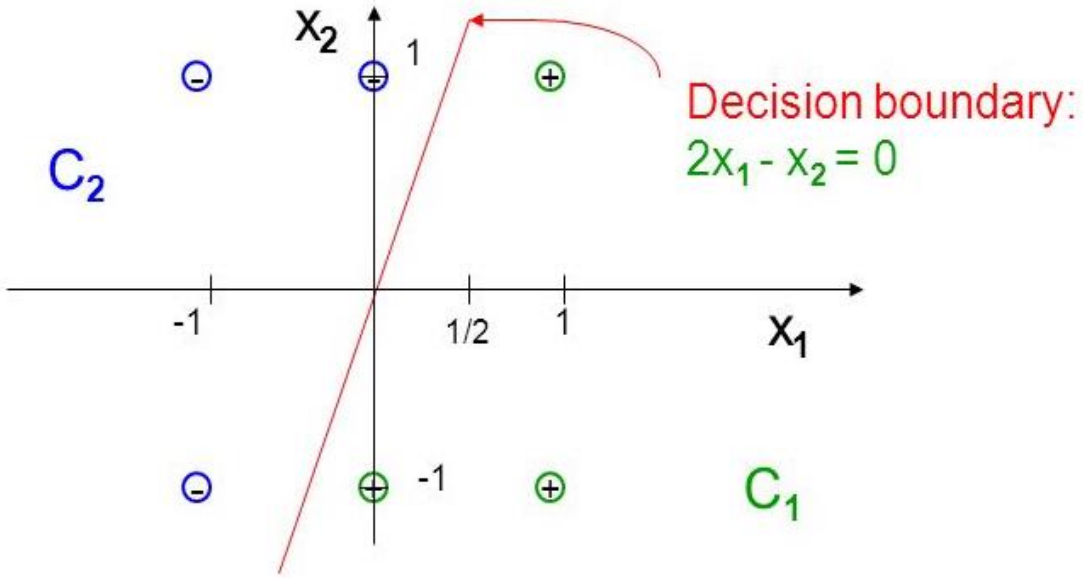
$$w_{i(new)} = w_{i(old)}$$
$$b_{(new)} = b_{(old)}$$

7. Test for stopping condition

Single Layer Perceptron Limitation

- This model is limited to perform the classification with only two classes.





Gradient Descent

- It is a technique that update the weights in every iteration to minimize the error of a model on our training data.

  - Input: Training instances <0, 0>, <0, 1>, <1, 0>, <1, 1> one at a time.
  - Model make a prediction for the training instances
  - Calculate the error = $\alpha t x_i$
  - Update weights to reduce the error for next prediction

$$w_{i(new)} = w_{i(_{old})} + \alpha t x_i$$

Example

- Develop a perceptron for AND function with bipolar inputs and targets. (initial weights are 0; learning rate = 1; and threshold =1)

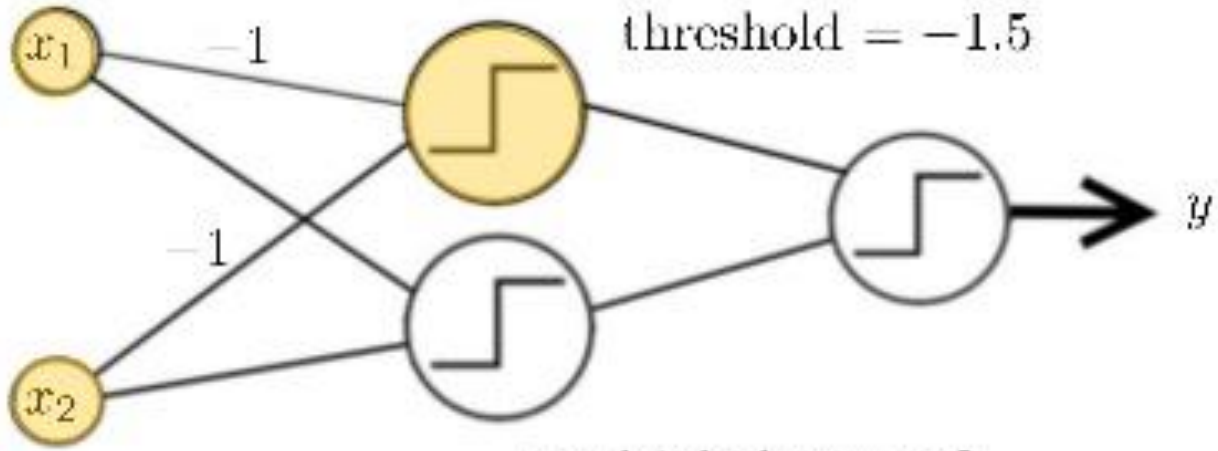
- Develop a perceptron for OR function with bipolar inputs and targets. (initial weights are 0; learning rate = 1; and threshold =1)

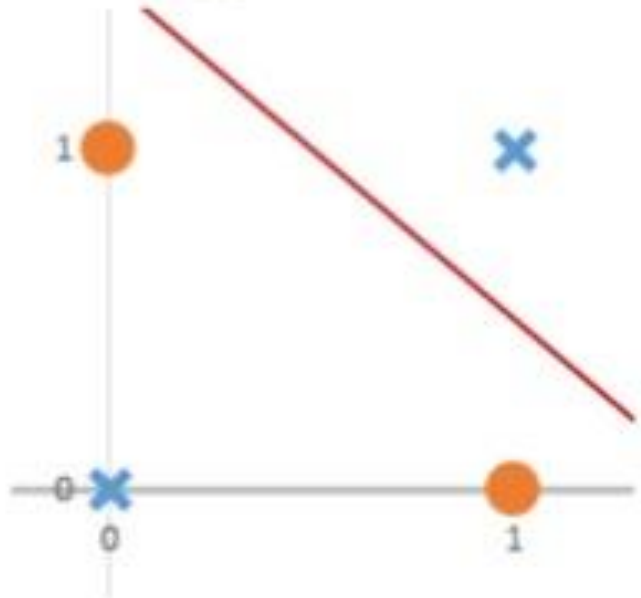
- Assignment:
  - NAND, NOR, and NOT functions (initial weights are 0; learning rate = 1; and threshold =1)

XOR Problem

Training Data

x_1	x_2	t
0	0	0
1	0	1
0	1	1
1	1	0





Multiple layers of perceptrons solve the XOR problem, but Rosenblatt did not have an learning algorithm to set the weights

Multilayer Networks

- Feed forward Networks
  - Output is calculated for every input to the network
  - No feedback

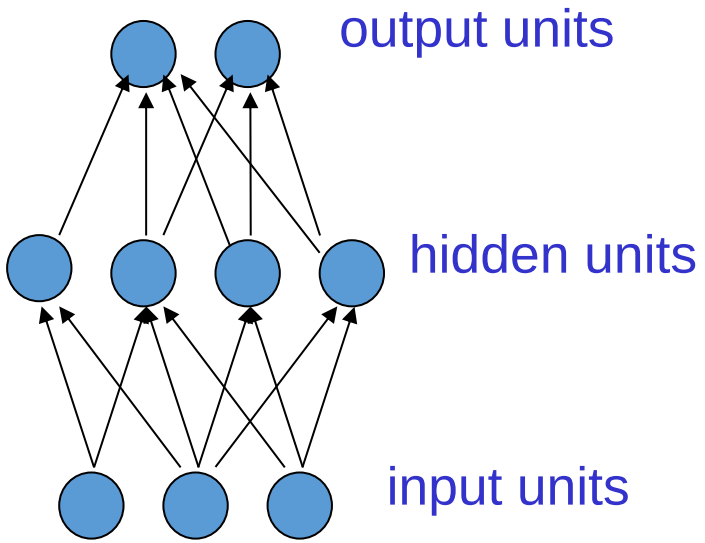
  - Example: Back Propagation Network

- Back Propagation Network (BPN)
  - Multilayer ANN
  - Using gradient descent delta learning rule known as back propagation (for errors)
  - Efficiently changing the weights with **differentiable activation function** units to learn a training set of input-output examples.

Back Propagation Network (BPN)

Architecture

- These are the commonest type of neural network in practical applications.
  - The first layer is the input and the last layer is the output.
  - If there is more than one hidden layer, we call them "**deep**" neural networks.
- They compute a series of transformations that change the similarities between cases.
  - The activities of the neurons in each layer are a non-linear function of the activities in the layer below.



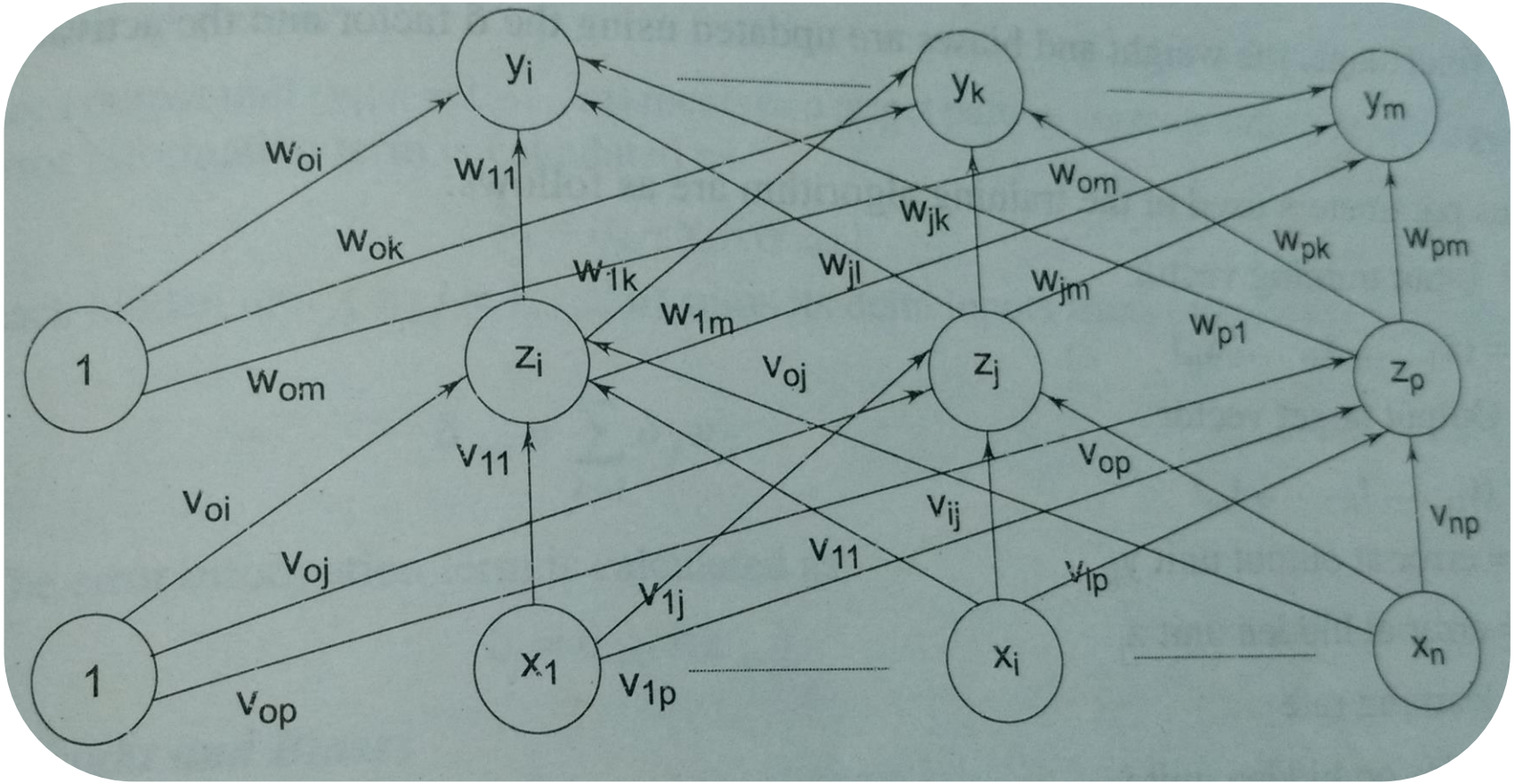
During back propagation learning

- Signals sent in reverse direction also.

Increase in number of hidden layer

- Minimize error
- Increase computational complexity
- Increase time taken for convergence.

Back Propagation Network (BPN)



Training Algorithm

Step 1: Initialize weight to small random values.

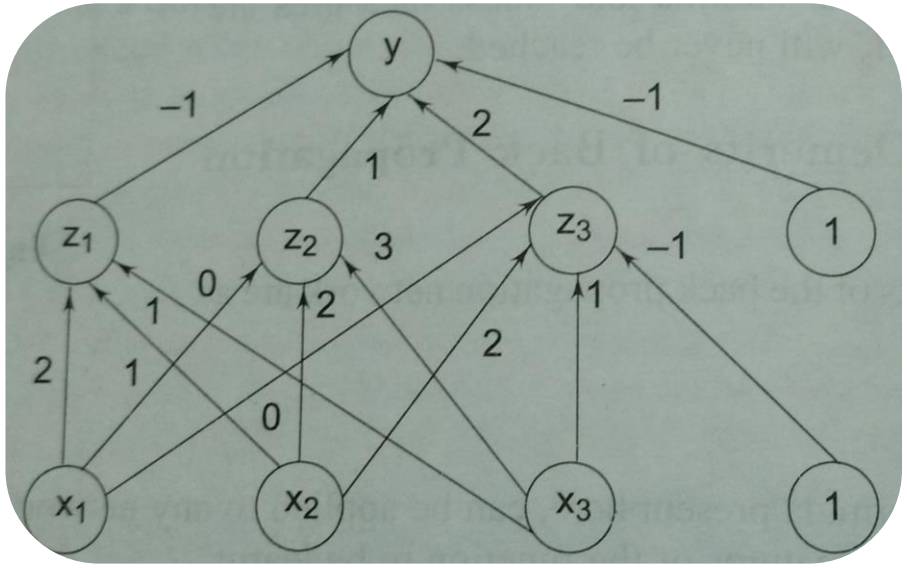
Step 2: While stopping condition is false, do steps 3 – 10

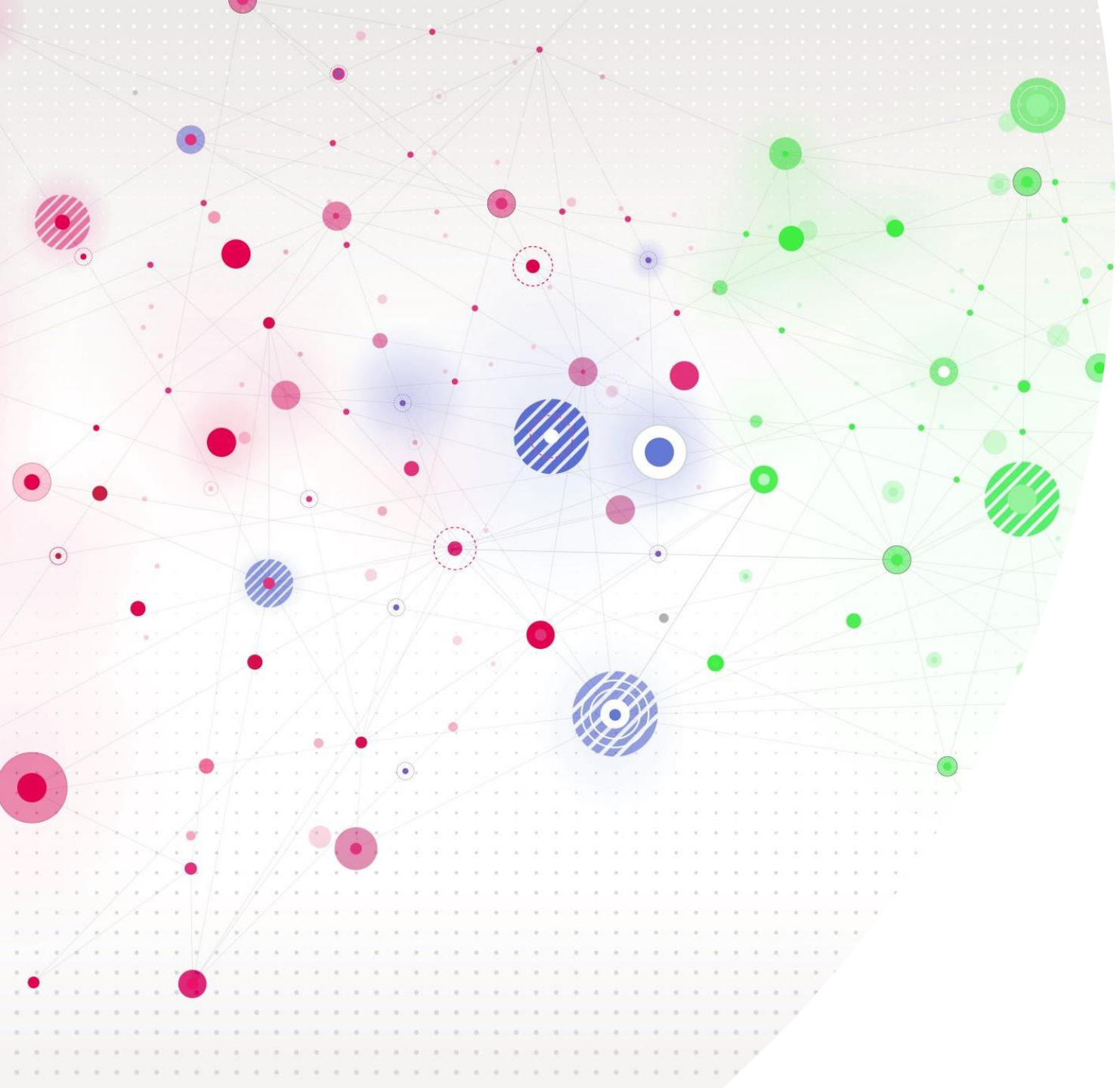
Step 3: For each training pair do steps 4 - 9

Example

- Find the new weights when the network given in the figure is presented the input pattern [0.6 0.8 0] and the target output is 0.9. use the learning rate $\alpha = 0.3$ and use binary sigmoid activation function.

- W = [-1 1 2]
- W_{01} = [-1]
- $V_{ij} = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 2 & 2 \\ 0 & 3 & 1 \end{bmatrix}$
- V_{01} = [0 0 -1]





Clustering in Machine Learning

Arockiaraj S

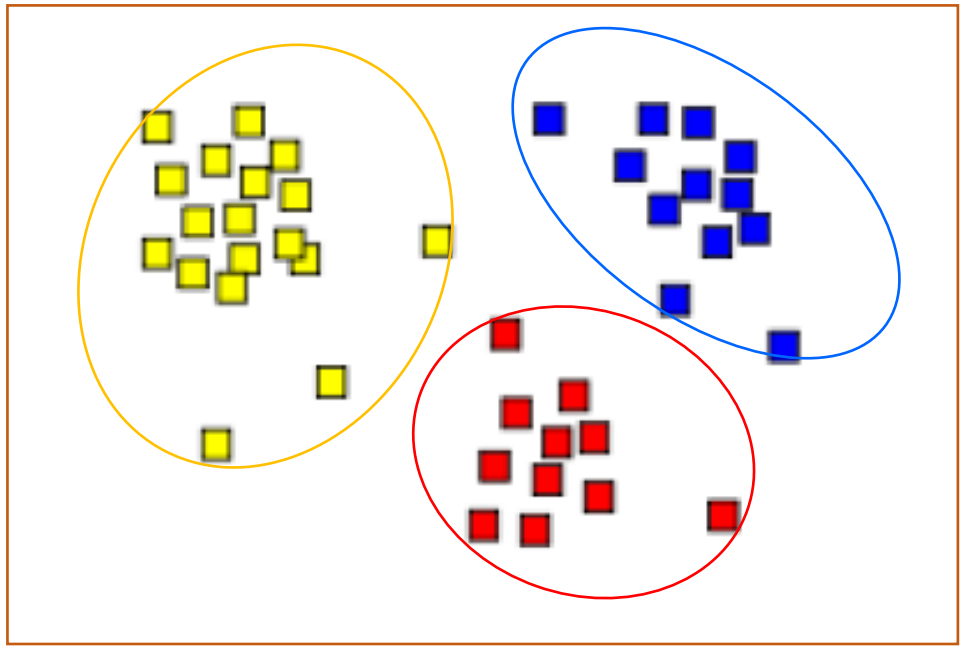
What is clustering?

The organization of unlabeled data into similarity groups called clusters.

Data items within a cluster are "*similar*"

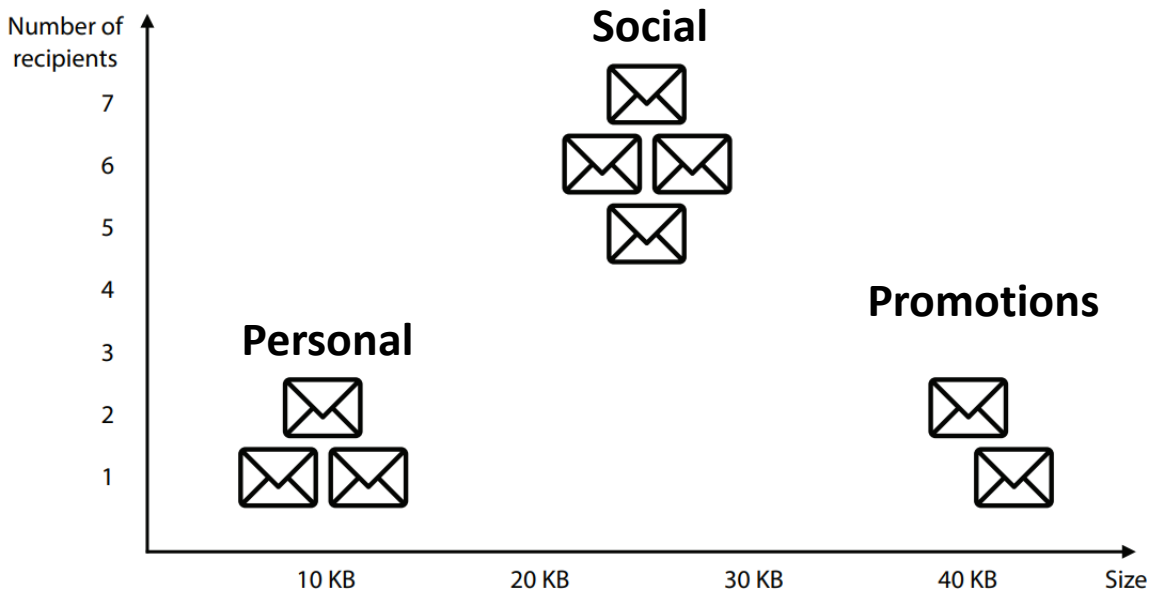
Data items between clusters are "*dissimilar*"

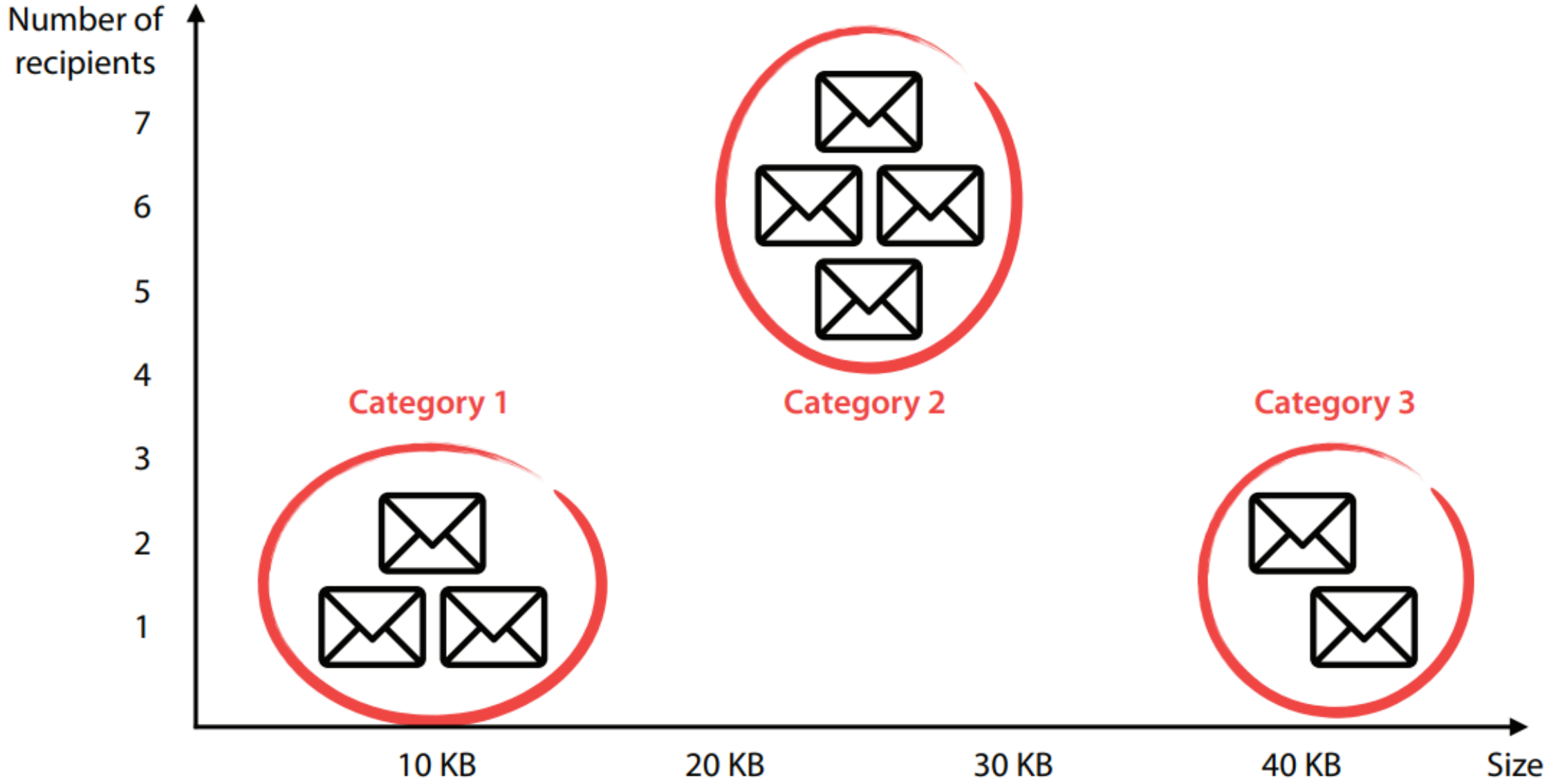
- Clustering is an unsupervised learning method
- Used for statistical data analysis in many fields.



cluster the emails into three categories based on size and number of recipients

Email	Size	Recipients
1	8	1
2	12	1
3	43	1
4	10	2
5	40	2
6	25	5
7	23	6
8	28	6
9	26	7




Number of recipients
7
6
5
4
3
2
1
Category 1
Category 2
Category 3
10 KB
20 KB
30 KB
40 KB
Size

Other applications of clustering

- **Market segmentation**: dividing customers into groups based on demographics and previous purchasing behavior to create different marketing strategies for the groups
- **Genetics**: clustering species into groups based on gene similarity
- **Medical imaging**: splitting an image into different parts to study different types of tissue
- **Video recommendations**: dividing users into groups based on demographics and previous videos watched and using this to recommend to a user the videos that other users in their group have watched

Types of Clustering

- **Soft Clustering**:
  - In soft clustering, instead of putting each data point into a separate cluster, a ***probability or likelihood*** of that data point to be in those clusters is assigned.
  - For example, each datapoint is assigned a probability to be in either in C1 or C2.

Data Points	Probability of C1	Probability of C2
A	0.91	0.09
B	0.3	0.7
C	0.17	0.83
D	1	0

Distance and Similarity measures in Clustering

Distance Measures:

- Used to determine the dissimilarity between two data points.

- There are several distance measures commonly used in clustering.

1. Minkowski distance family
    1. Euclidean distance
    2. Manhattan distance
2. Cosine similarity
3. Mahalanobis distance

1. Euclidean Distance:

- This is the most common distance measure used in clustering.
- Euclidean distance can be used for data with continuous variables, such as height, weight, or age.
- if $p = (p_1, p_2,\ldots, p_n)$ and $q = (q_1, q_2,\ldots, q_n)$ are two points in Euclidean n-space, then the distance (d) from p to q, or from q to p is given by the Pythagorean formula:

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

2. **Manhattan Distance:**

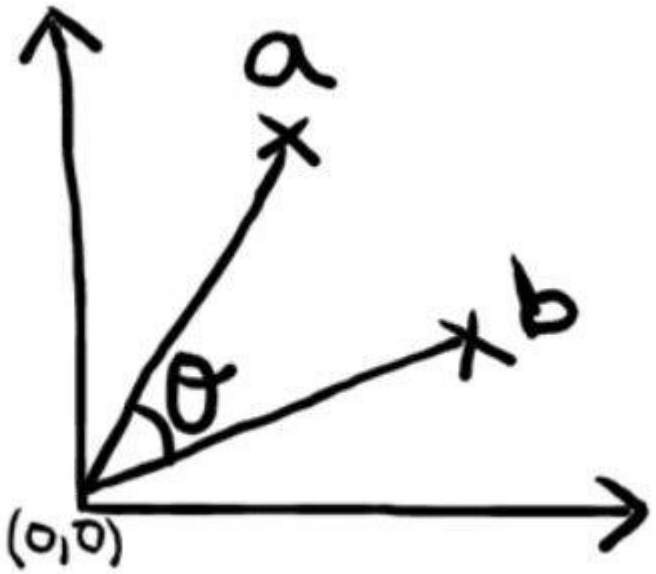
- This distance measure is also known as city block distance or taxicab distance.
- Manhattan distance can be used for data with discrete variables, such as the number of bedrooms in a house.

Manhattan Distance Formula

$$D_{mn}(x_i, x_j) = \sum_{l=1}^{d} |x_{il} - x_{jl}|$$

3. **Cosine Similarity:**

- This similarity measure is commonly used for text-based data or other high-dimensional data.
- It measures the cosine of the angle between two vectors in a multidimensional space.
- The cosine similarity ranges from -1 to 1, with 1 indicating that the vectors are identical a value of 0 indicating that they are orthogonal (perpendicular) to each other.



$$\cos\theta = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|}$$

K-Means Clustering

MacQueen, 1967

K-means is a Partitional Clustering algorithm – dividing a dataset into distinct groups or clusters.

The *k*-means algorithm partitions the given data into *k* clusters: Each cluster has a cluster **center**, called **centroid**.

The grouping is done by minimizing the sum of squares of distances between data and the corresponding cluster centroid.

K-Means Algorithm

Step-1: Choose the number of clusters (K)

Step-2: Select at random k points, the centroids (not necessarily from your dataset)

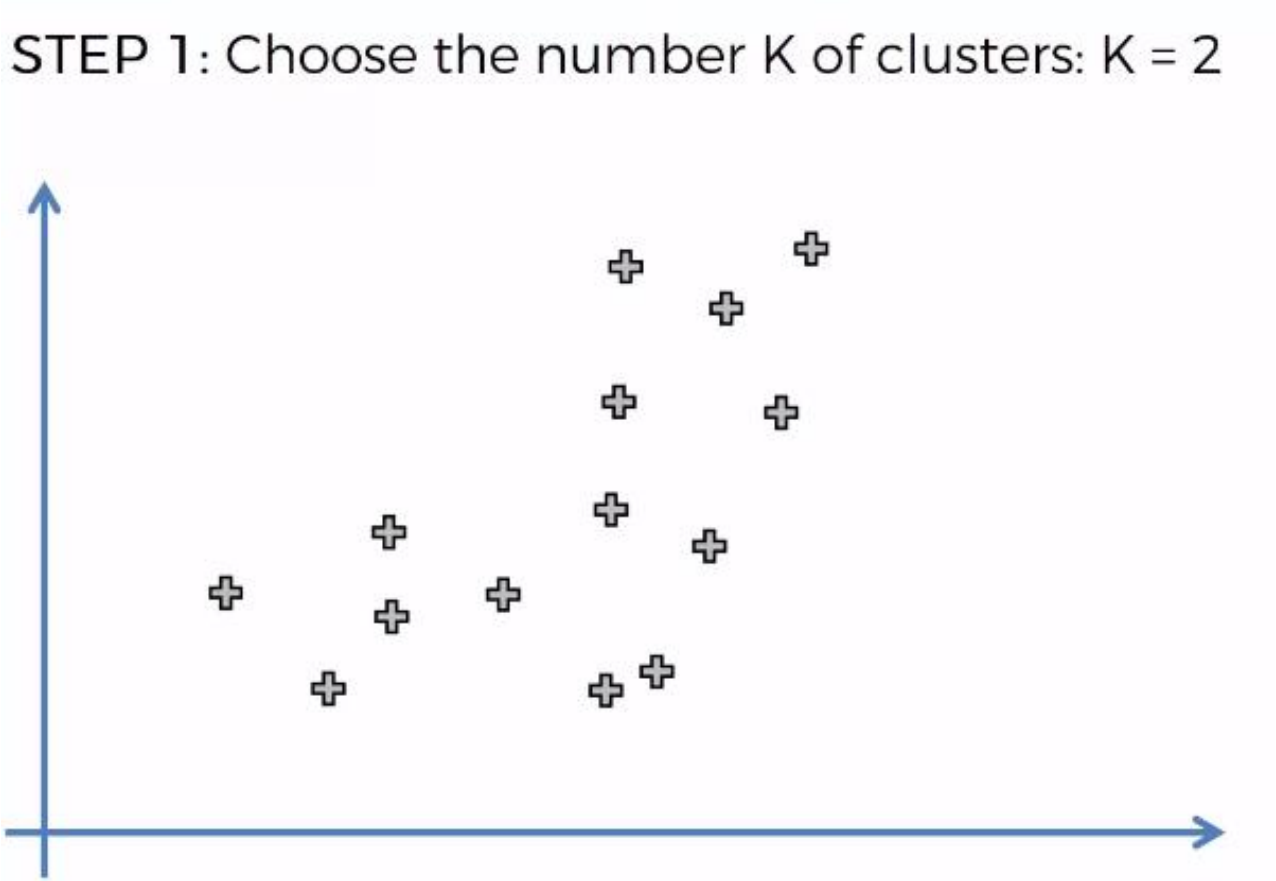
Step-3: Assign each data point to the closest centroid → that forms k clusters

Step-4: Compute and place the new centroid of each cluster

Step-5: Reassign each data points to the new closest centroid.
If any reassignment took place, go to Step-4, otherwise go to Stop.

Stop: your Model is ready

Example



K-Means Clustering: Example

- These objects belong to two groups of medicine.
- Determine which medicines belong to cluster 1 and which medicines belong to the other cluster 2.

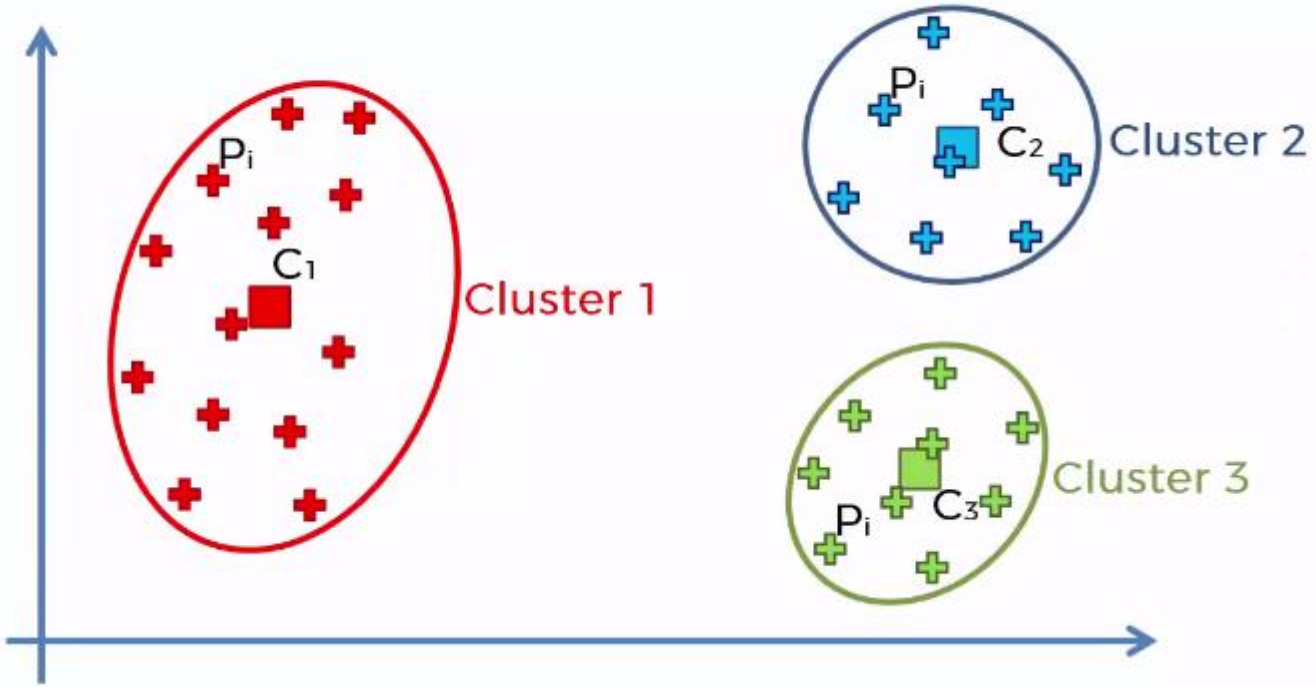
Objects	Attribute 1 (X):weight index	Attribute 2 (Y): pH
Medicine A	**1**	**1**
Medicine B	**2**	**1**
Medicine C	**4**	**3**
Medicine D	**5**	**4**

Random Initialization of Centroids

What will happen if we choose a bad random initialization?

Choosing the Right Number of Clusters

→ Within Cluster Sum of Square (WCSS)



$$\text{WCSS} = \sum_{p_i \text{ in cluster 1}} \text{distance}\,(P_i, C1)^2 + \sum_{p_i \text{ in cluster 2}} \text{distance}\,(P_i, C_2)^2 + \sum_{p_i \text{ in cluster 3}} \text{distance}\,(P_i, C_3)^2$$

Strengths of K-means

- **Simple**: easy to understand and to implement
- **Efficient:** Time complexity: $O(tkn)$
  - n-is the number of data points
  - *k-is the number of clusters*
  - *t- is the number of iterations*
  - Since both k and t are small. k-means is considered a linear algorithm.
- K-means is the most popular clustering algorithm.

Weaknesses of K-means

- The user needs to specify number of clusters (K).
- The algorithm is sensitive to **outliers**
  - Outliers are data points that are **very far away from other data points**.
  - Outliers could be **errors** in the data recording or some **special data points** with very different values.

Hierarchical Clustering

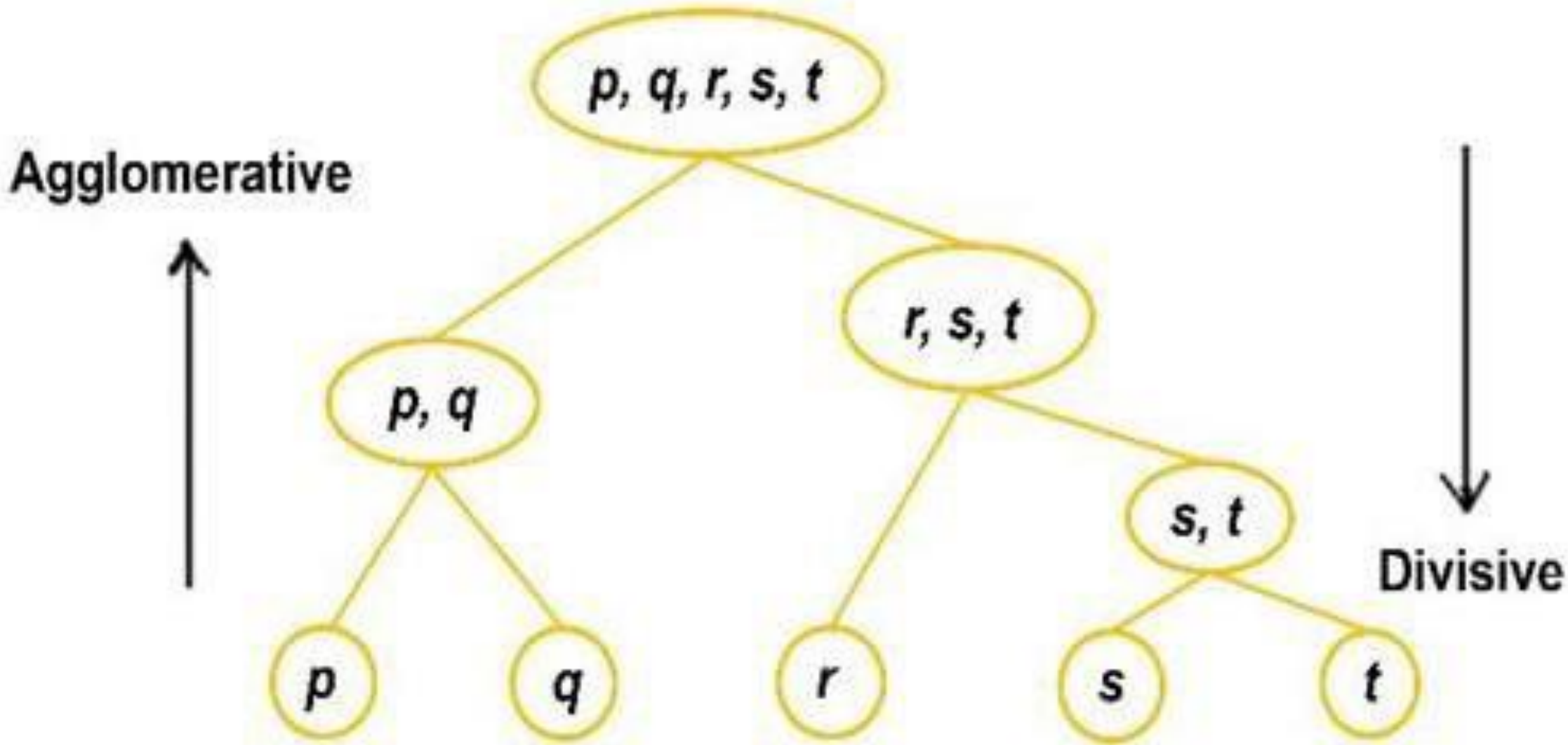
In Machine Learning

Hierarchical Clustering

- Two types
  - Agglomerative (bottom-up)
  - Divisive (top-down)

- Agglomerative Algorithm
  - Begin with each data element as a separate cluster and merge them into successively larger cluster

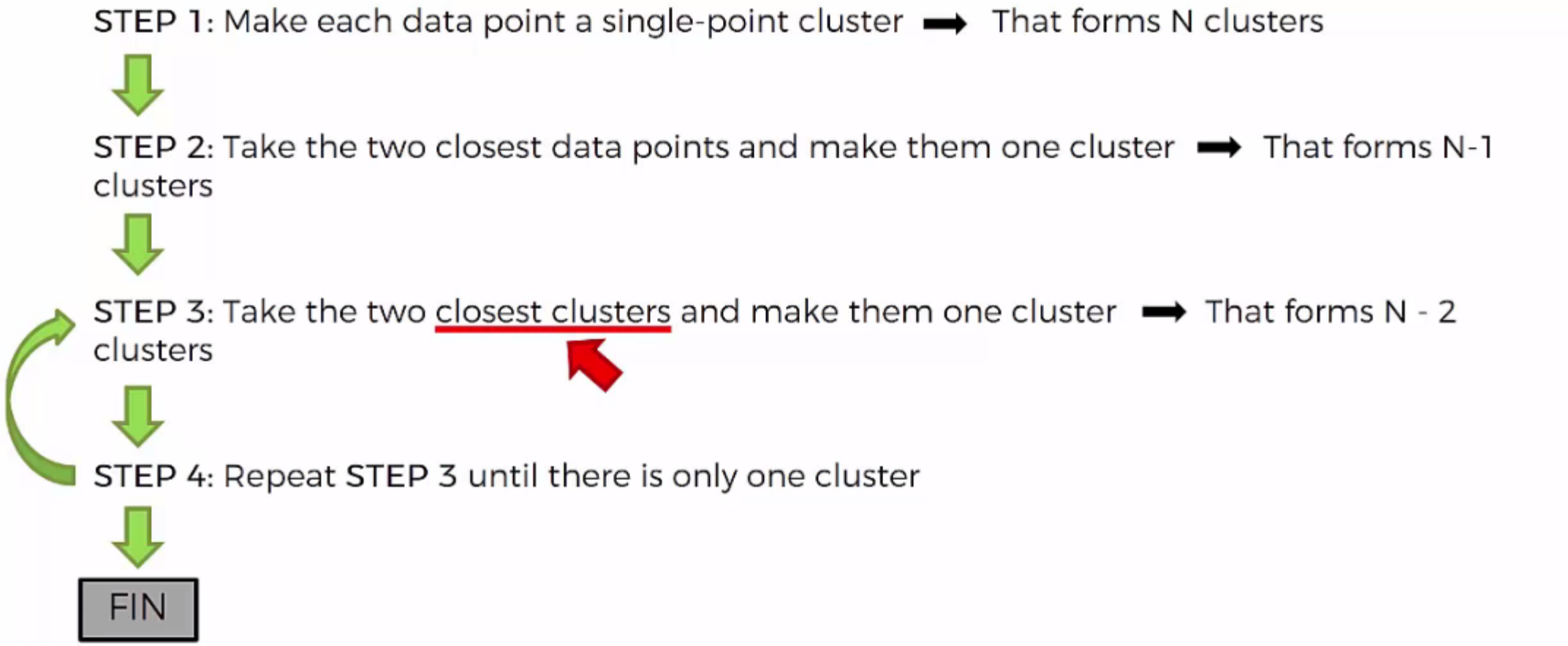
- Divisive Algorithm
  - Begins with the whole set and proceed to divide it into successive smaller clusters


Agglomerative
p, q, r, s, t
p, q
r, s, t
s, t
p
q
r
s
t
Divisive

Applications

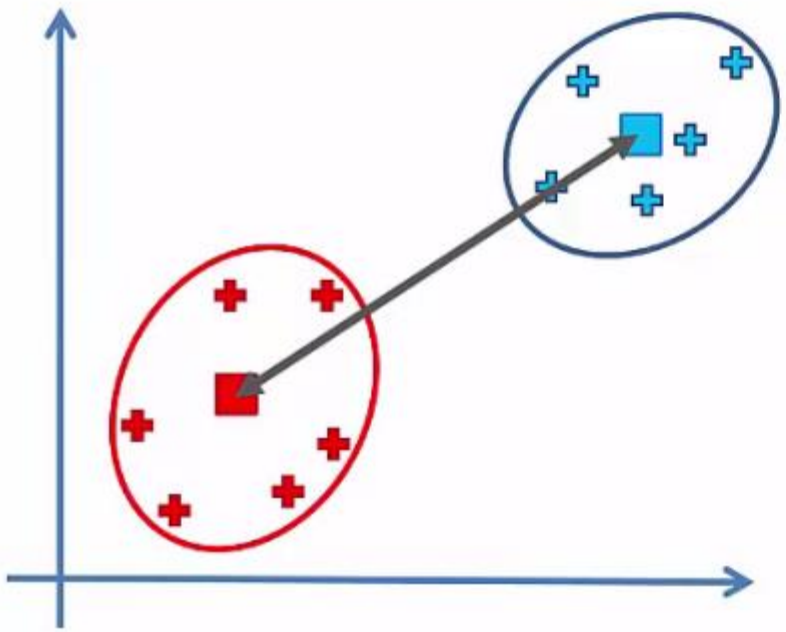
- **Bioinformatics**: grouping animals according to their biological features to reconstruct phylogeny trees
- **Business**: dividing customers into segments or forming a hierarchy of employees based on salary.
- **Image processing**: grouping handwritten characters in text recognition based on the similarity of the character shapes.
- **Information Retrieval**: categorizing search results based on the query

Agglomerative HC



- Closest clusters - Euclidean distance or Manhattan distance

Distance Between Clusters



Distance Between Two Clusters:

- Option 1: Closest Points
- Option 2: Furthest Points
- Option 3: Average Distance
- Option 4: Distance Between Centroids

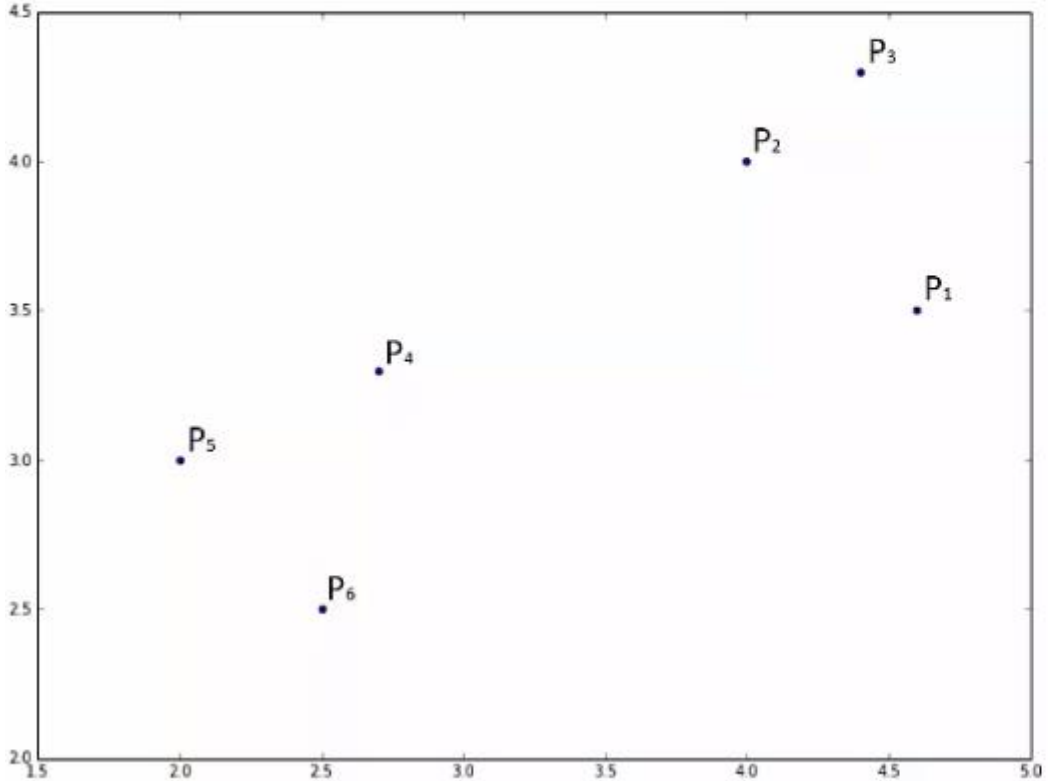
Centroid Linkage

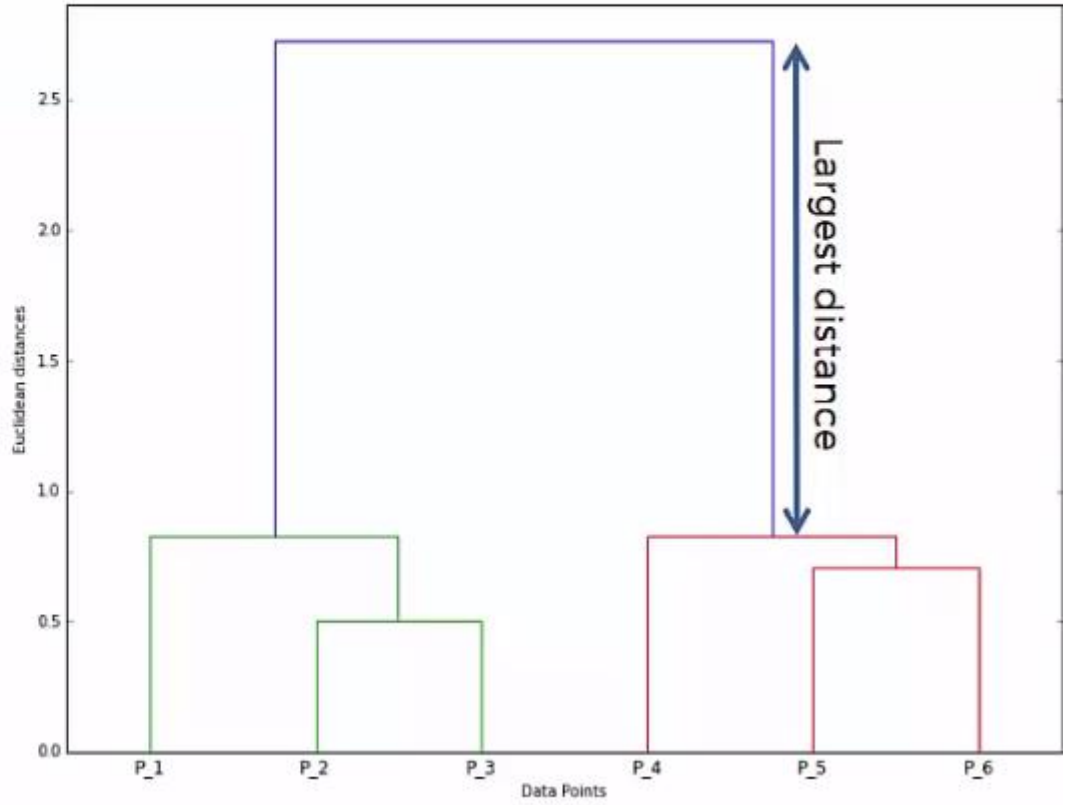
Ward Linkage

How do Dendrograms Works?

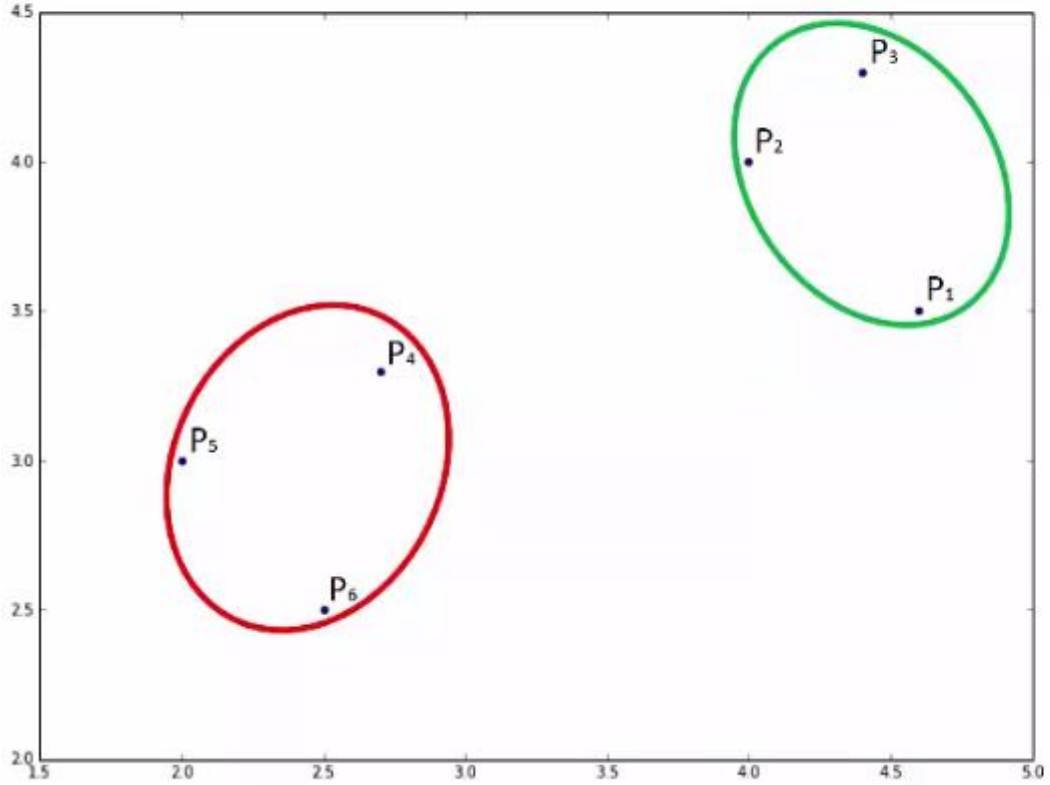
Using Dendrograms

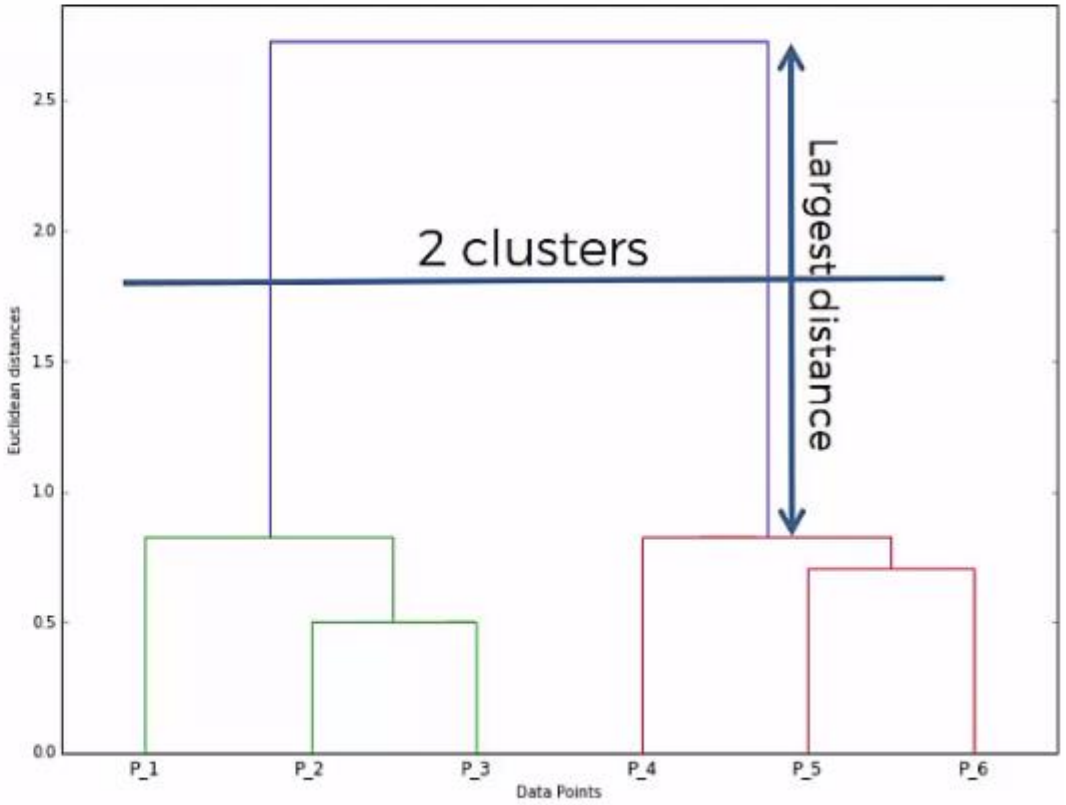
Dendrograms – Optimal # of Clusters





Dendrograms – Optimal # of Clusters





Example: Distance metrics used in hierarchical clustering

- **Single Linkage**, **Complete Linkage**, and **Average Linkage**.

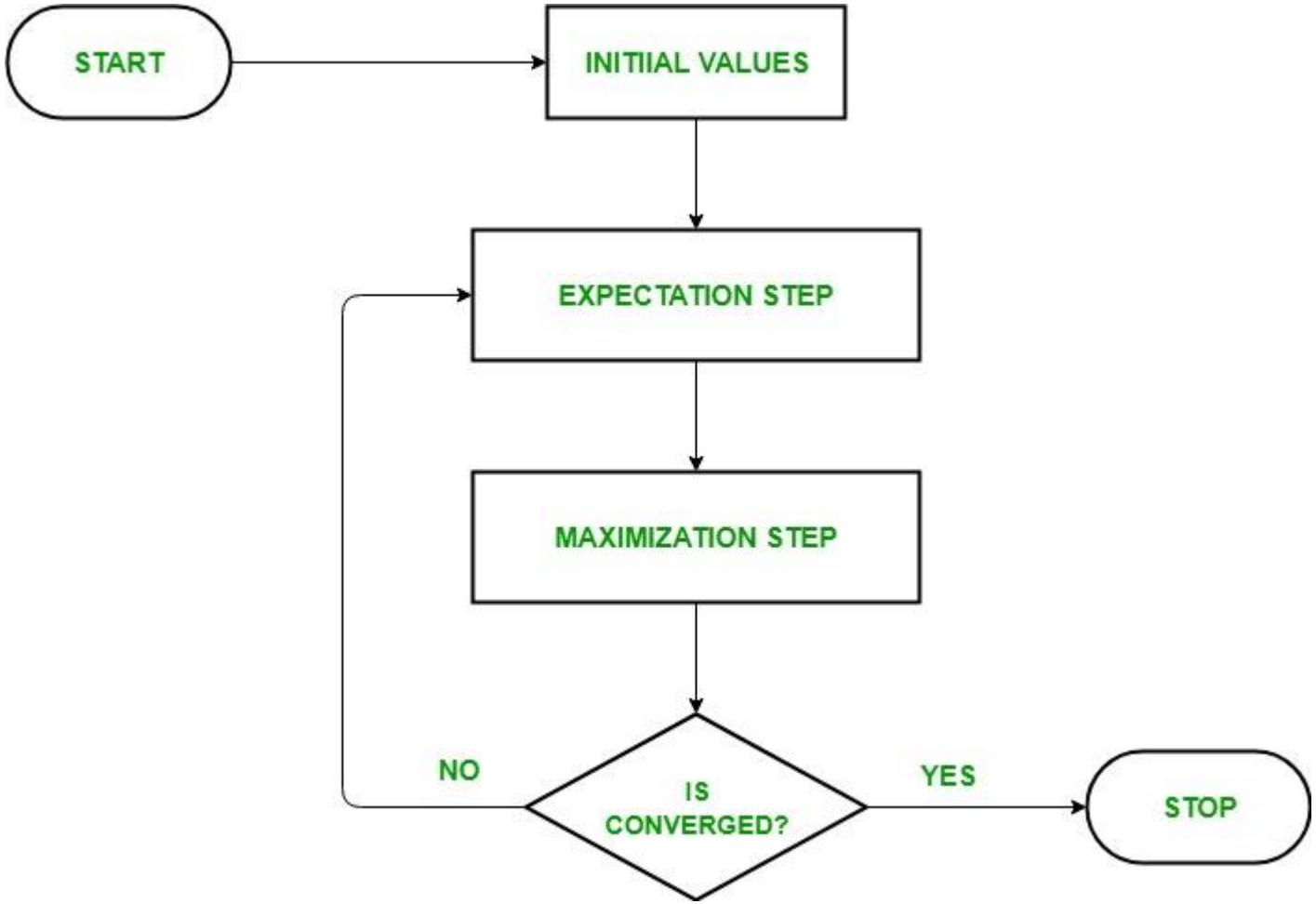
Point	x	y
A	1	2
B	2	3
C	3	1
D	5	4
E	6	5

Clustering

Expectation-Maximization Algorithm

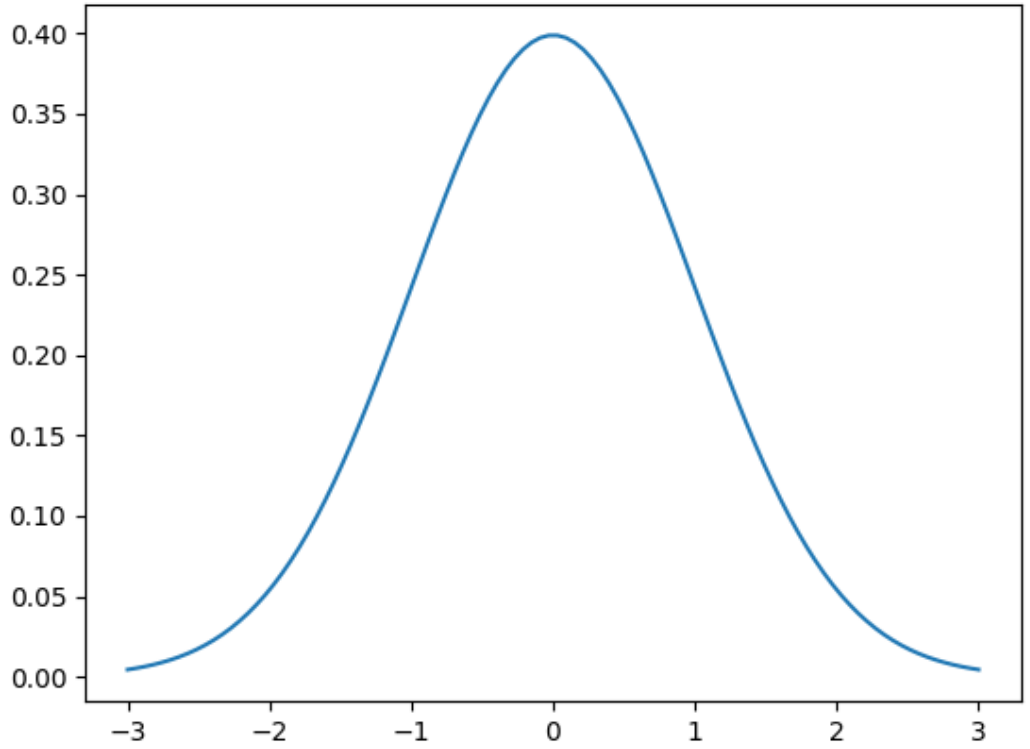
Expectation-Maximization (EM) Algorithm Works:

- The Expectation-Maximization algorithm use the available observed data of the dataset to estimate the missing data and then use that data to update the values of the parameters.



Gaussian Distribution

- **Gaussian** or **normal distribution**
- It is a statistical distribution.
- A Gaussian centered at 0 with a standard deviation of 1 shown in figure.
- It is also called a **bell curve** sometimes



x is an input

Two parameters in the equation:

μ is a mean & σ is a standard deviation.

σ^2 – variance

$$\mathcal{N}(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Function that describes the normal distribution

Advantages of EM algorithm

It is always guaranteed that likelihood will increase with each iteration.

The E-step and M-step are often easy for many problems in terms of implementation.

Disadvantages of EM algorithm

It has slow convergence.

It makes convergence to the local optima only.

It requires both the probabilities, forward and backward.

Clustering Metrics in Machine Learning

Clustering Metrics in Machine Learning

- Clustering - unsupervised machine-learning approach
- Group similar data points based on specific attributes.
- It is critical to evaluate the quality of the clusters created when using clustering techniques.
- These metrics are quantitative indicators used to evaluate the performance and quality of clustering algorithms.

Clustering Metrics in Machine Learning

- Silhouette score – will discuss
- Davies–Bouldin Index - self study
- Calinski-Harabasz Index – self study

Support Vector Machine (SVM)

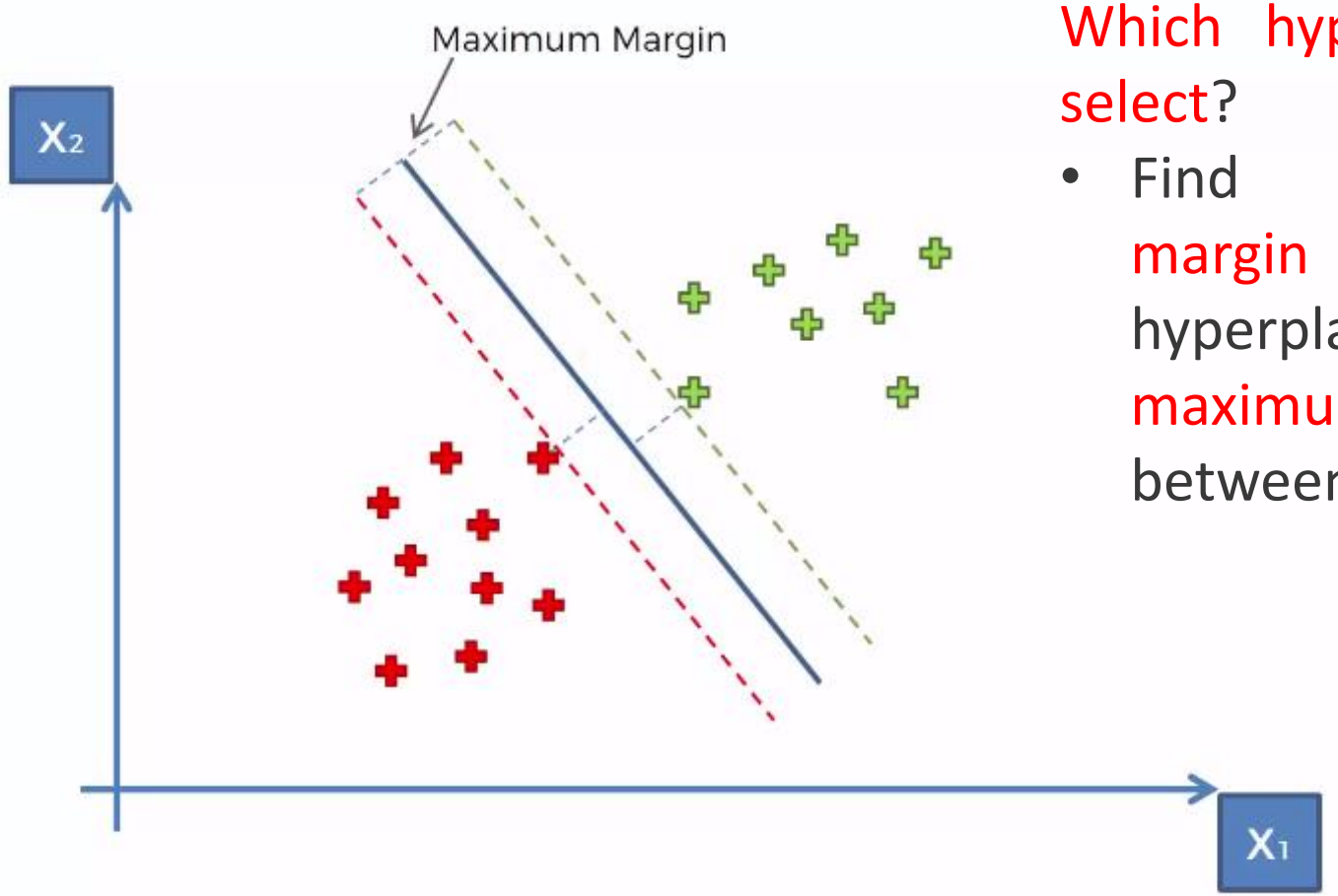
Machine Learning Approach

What is a Support Vector Machine(SVM)?

- It is a supervised machine learning problem where we try to find a hyperplane that best separates the two classes.

Maximum Margin

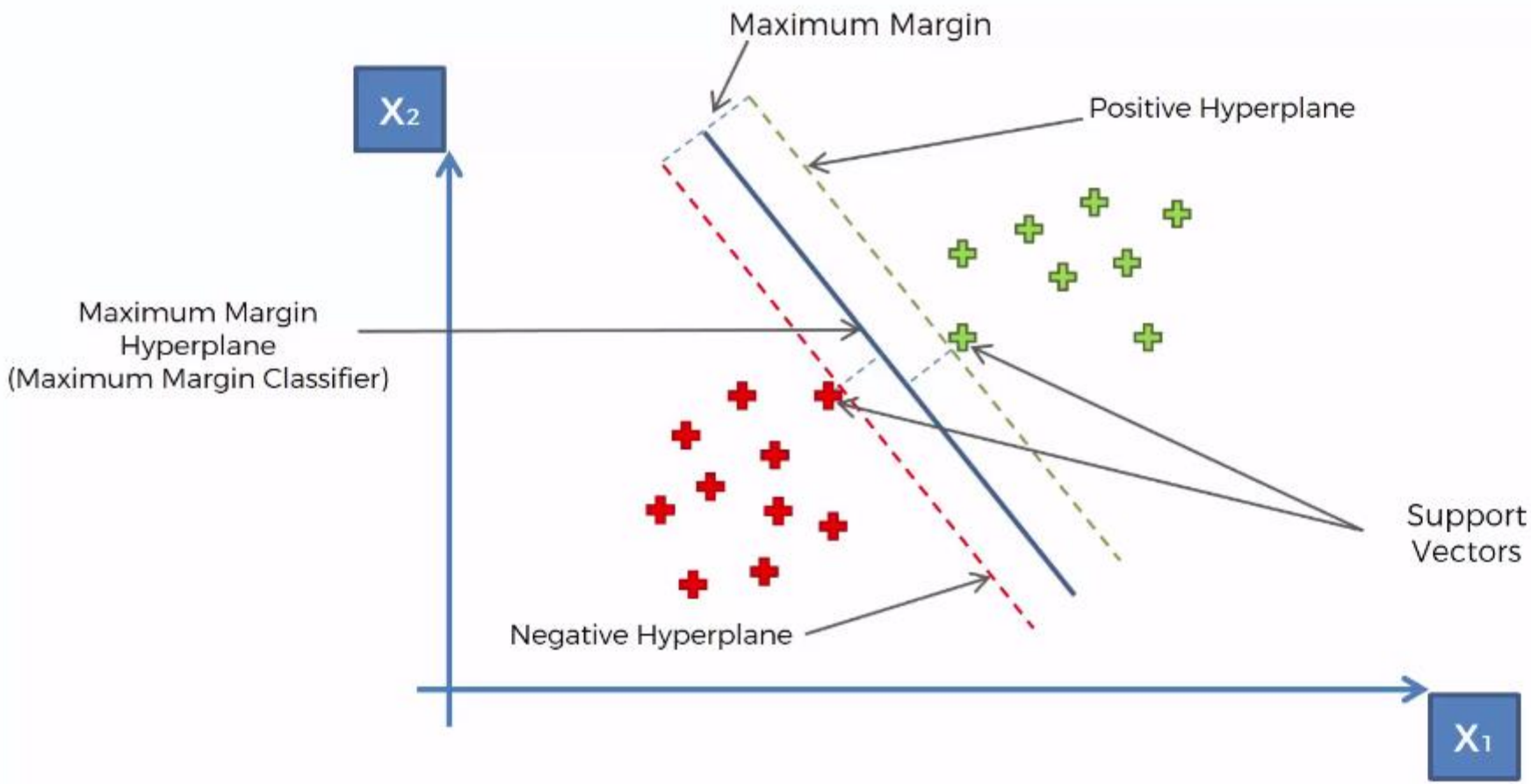
One with maximum margin is selected.



Which hyperplane does it select?

- Find the maximum margin between the hyperplanes that means maximum distances between the two classes.

Hyperplanes

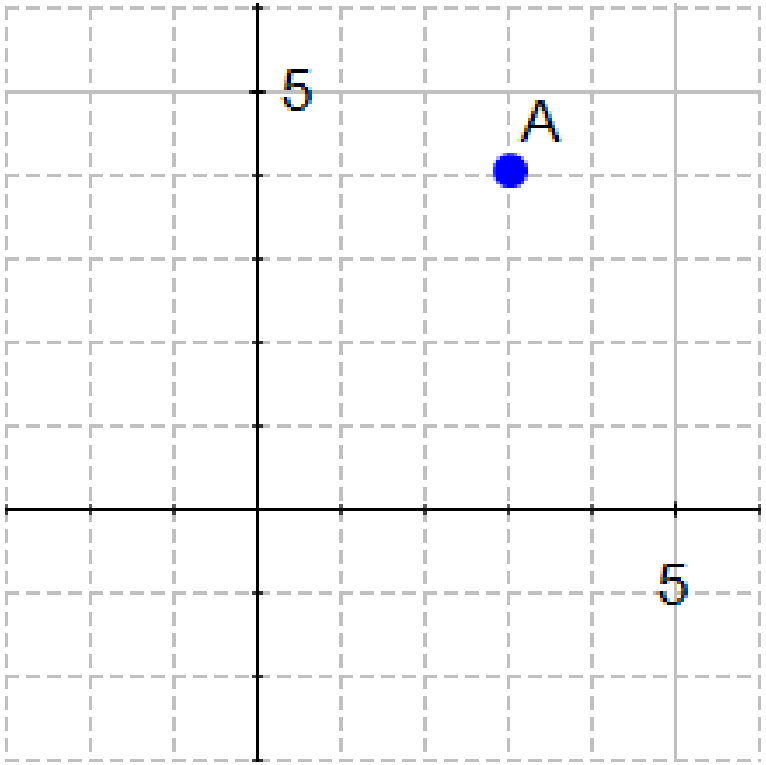


Mathematical Intuition Behind Support Vector Machine

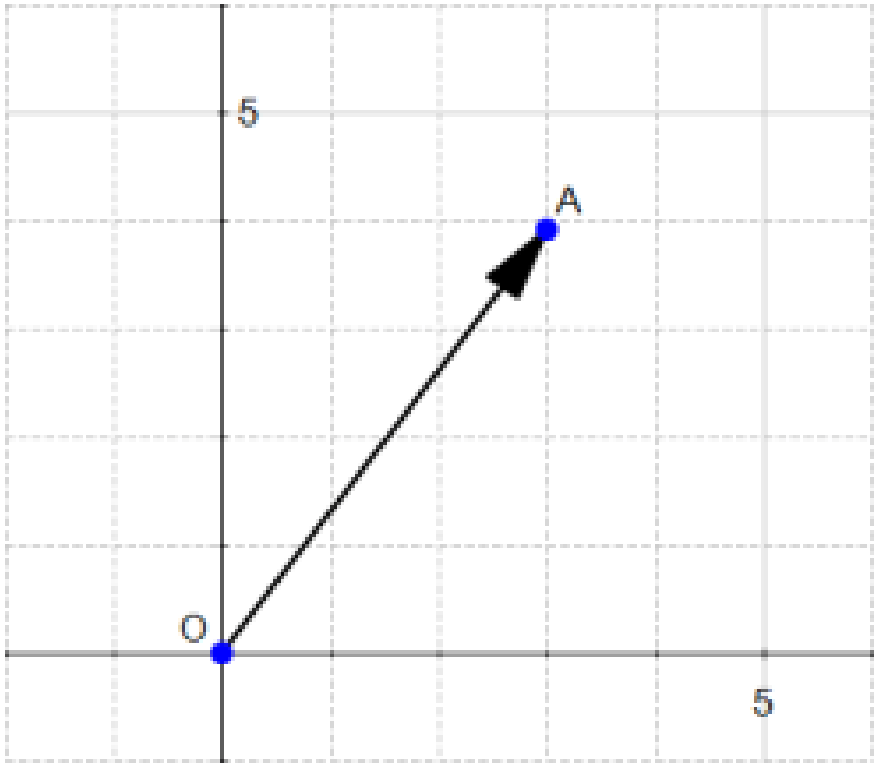
Vector

A vector is a quantity that has magnitude as well as direction and just like numbers we can use mathematical operations such as addition, multiplication.

If we define a point $A(3,4)$ in $\mathbb{R}^2$ we can plot it like this.



A vector in the plane, starting at the origin and ending at A



The magnitude

The magnitude or length of a vector X is written $\|X\|$ and is called its norm.

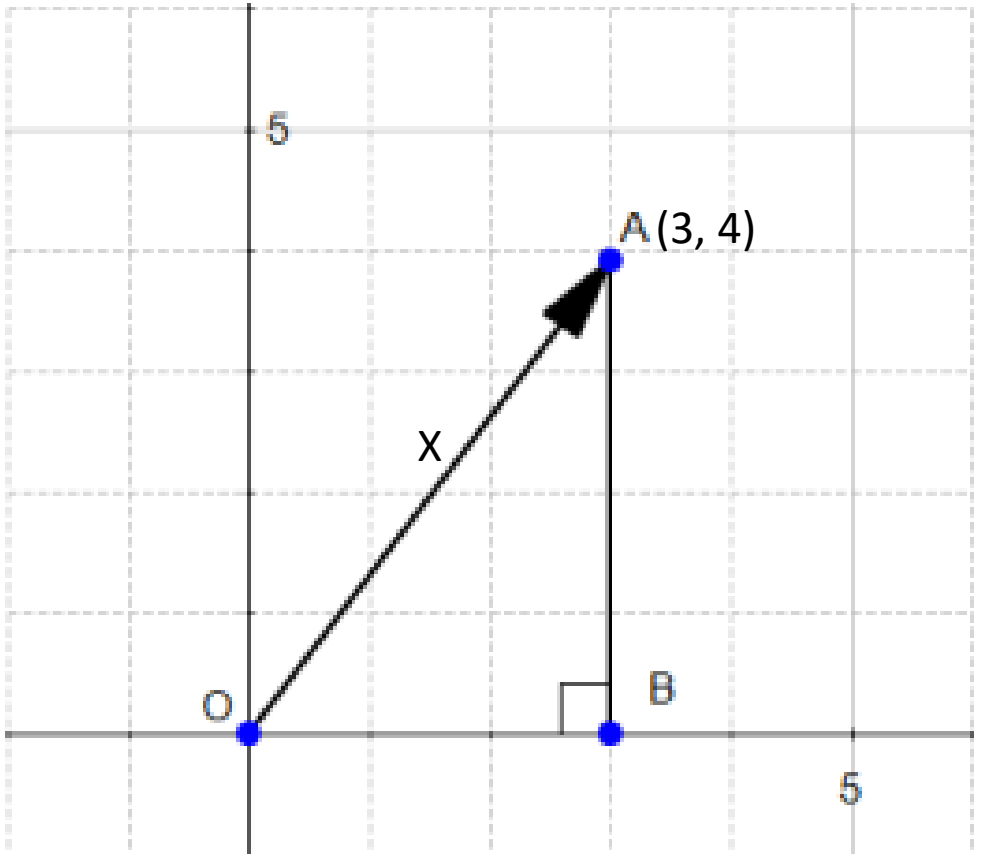
$$OA^2 = OB^2 + AB^2$$

$$OA^2 = 3^2 + 4^2$$

$$OA^2 = 25$$

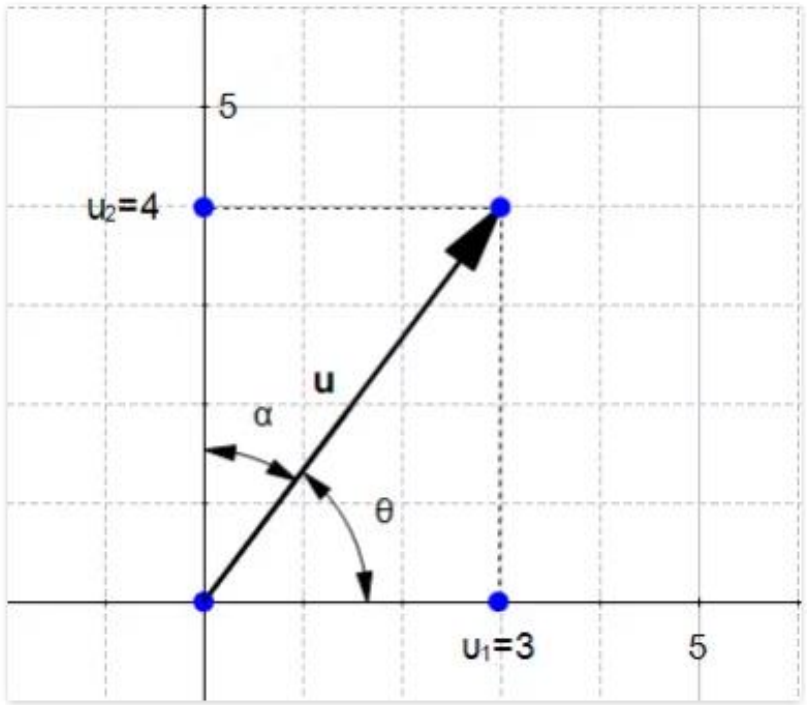
$$OA = \sqrt{25}$$

$$\|OA\| = OA = 5$$



The direction

*The **direction** of a vector* $\mathbf{u}(u_1, u_2)$ *is the vector* $\mathbf{w}\left(\frac{u_1}{\|u\|}, \frac{u_2}{\|u\|}\right)$



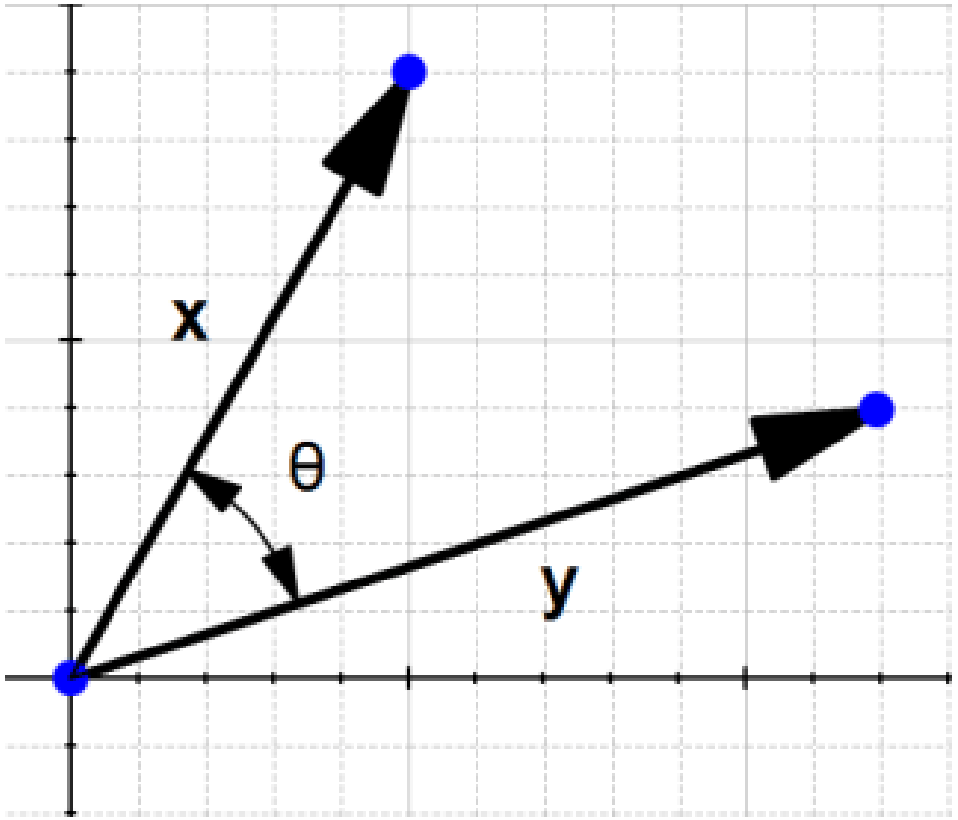
The dot product

- One **very** important notion to understand SVM is the dot product.

- If we have two vectors **x** and **y** and there is an angle **θ** between them, their **dot** product is :

$$x \cdot y = \|x\| \, \|y\| \cos(\theta)$$

$$\cos(\theta) = \frac{adjacent}{hypotenuse}$$



Classification of Data points

Classification of Data points: How to classify the points as +ve or –ve.

$y = mx + c$

$m = -1$ & $c = 0$

$y - mx - c = 0$

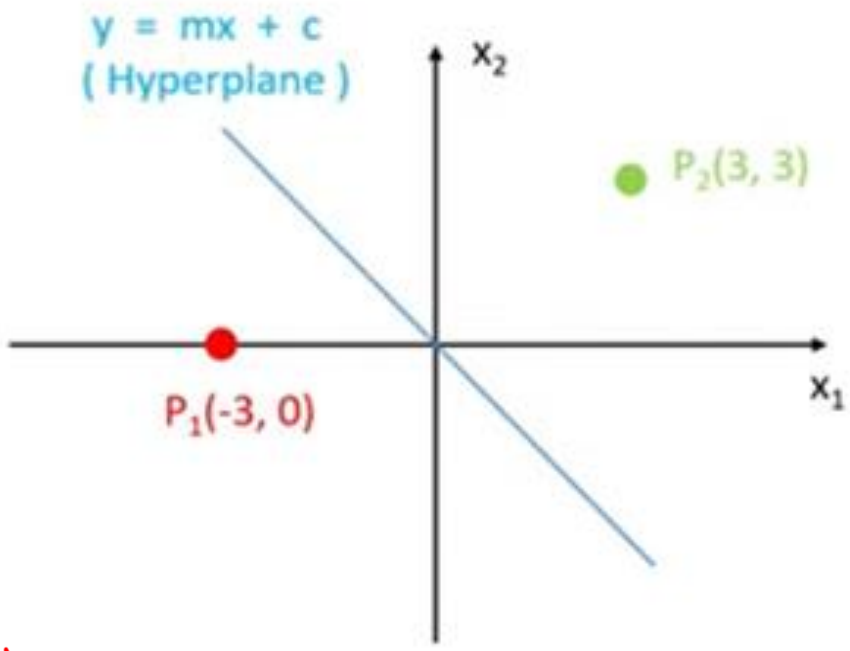
$y + x = 0$

$w = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$

$x = \begin{pmatrix} x \\ y \end{pmatrix}$

$w^T x = (1, 1) \begin{pmatrix} x \\ y \end{pmatrix}$ → data point value P_1, P_2

$= x + y$



Consider $P_1(-3, 0)$

$w^T x \Rightarrow w^T P_1$

$= (1, 1) \begin{pmatrix} -3 \\ 0 \end{pmatrix}$

$= -3$

$w^T x$ for $P_1(-3, 0)$ is **−ve**

For $P_2[3, 3]$

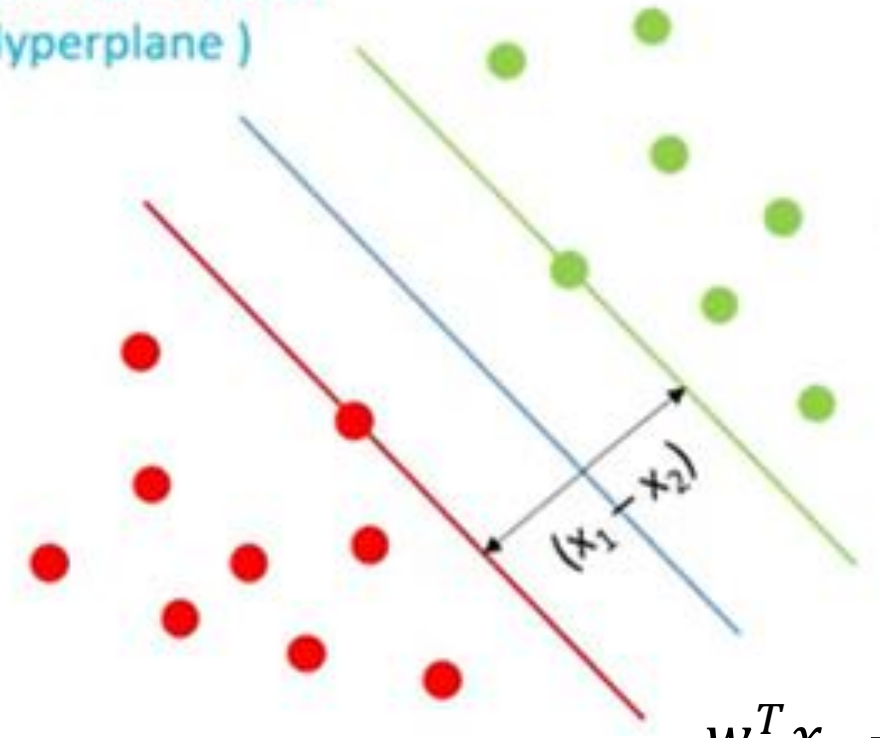
$w^T x = (1, 1) \begin{pmatrix} 3 \\ 3 \end{pmatrix} = 6$

= +ve

Optimization for Maximum margin:

$$y_i = \begin{cases} -1, & w^T x_1 + b \leq -1 \\ 1, & w^T x_1 + b \geq 1 \end{cases} \quad \text{(Label)}$$

$w^T x + b$ = Label
(Hyperplane)



$w^T x_1 + b = 1$

$w^T x_2 + b = -1$

Soft Margin SVM

- In real-life applications, we rarely encounter datasets that are perfectly linearly separable.
- Instead, we often come across datasets that are either nearly linearly separable or entirely non-linearly separable.
- Unfortunately, the trick demonstrated above for linearly separable datasets is not applicable in these cases.
- This is soft margin classifier - Support Vector Machines (SVM) come into play.

Linear Example

Suppose we are given the following positively labeled data points in $\Re^2$:

$$\left\{\begin{pmatrix} 3 \\ 1 \end{pmatrix}, \begin{pmatrix} 3 \\ -1 \end{pmatrix}, \begin{pmatrix} 6 \\ 1 \end{pmatrix}, \begin{pmatrix} 6 \\ -1 \end{pmatrix}\right\}$$

and the following negatively labeled data points in $\Re^2$ (see Figure 1):

$$\left\{\begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \end{pmatrix}\right\}$$

Linear Example

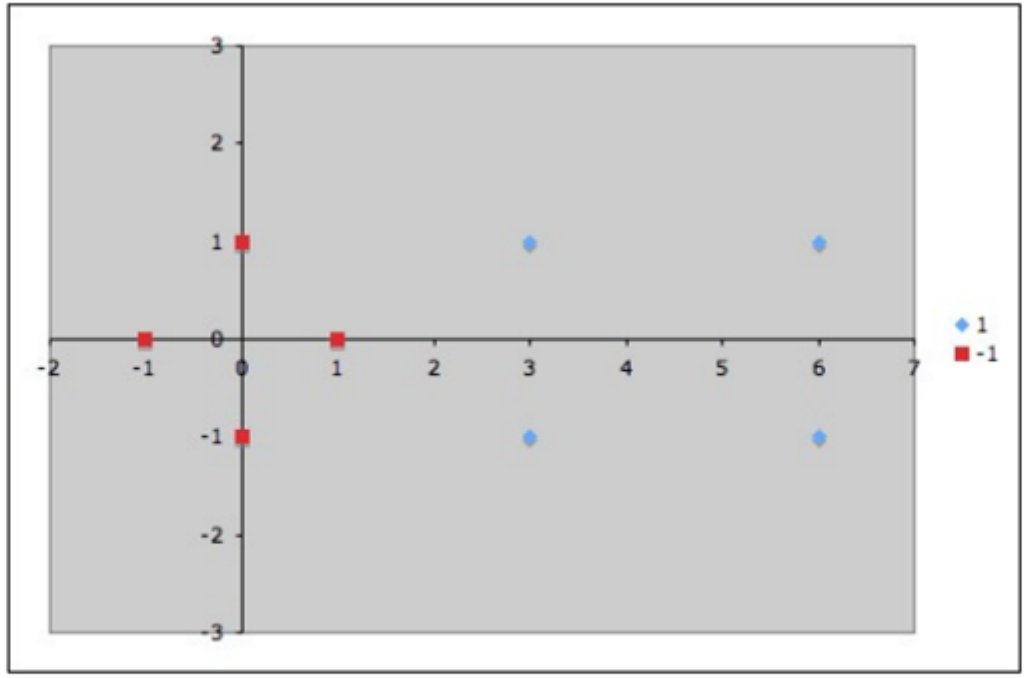
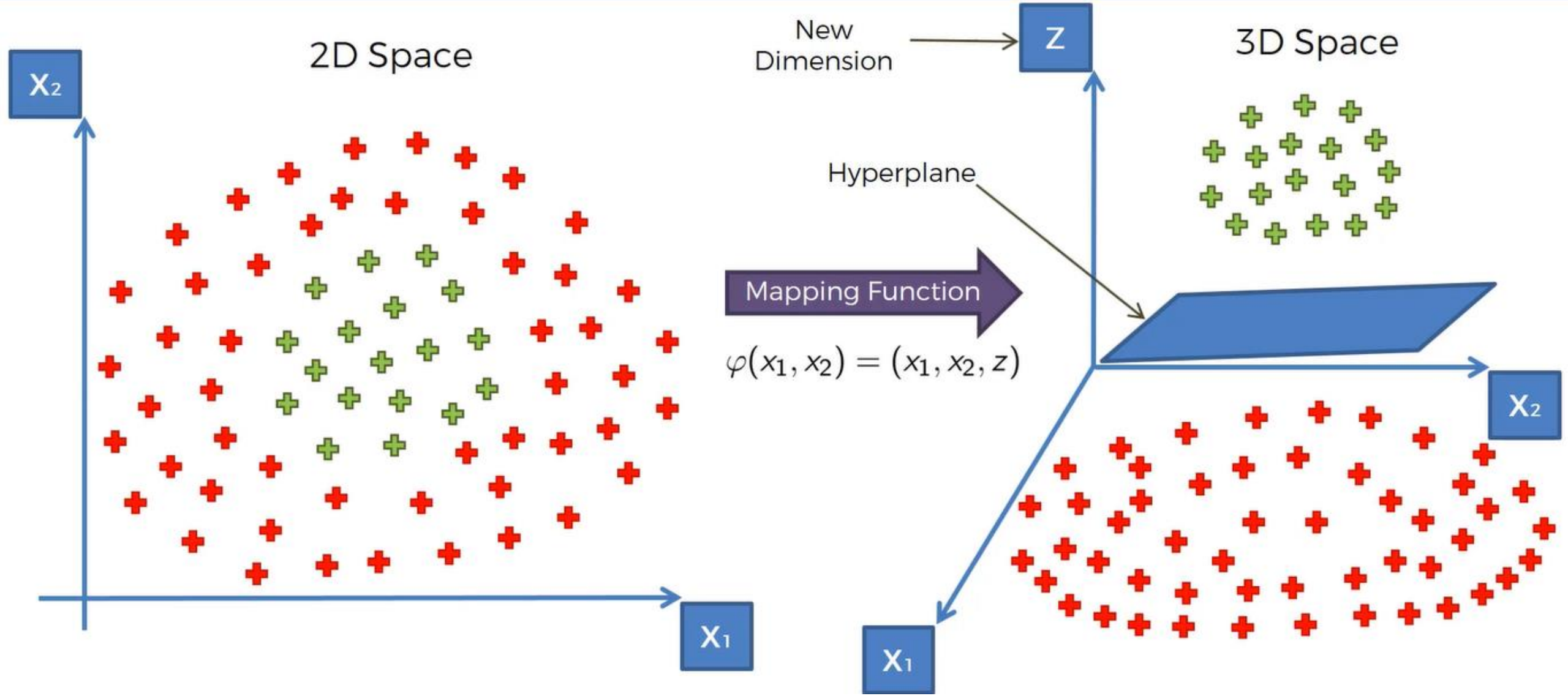


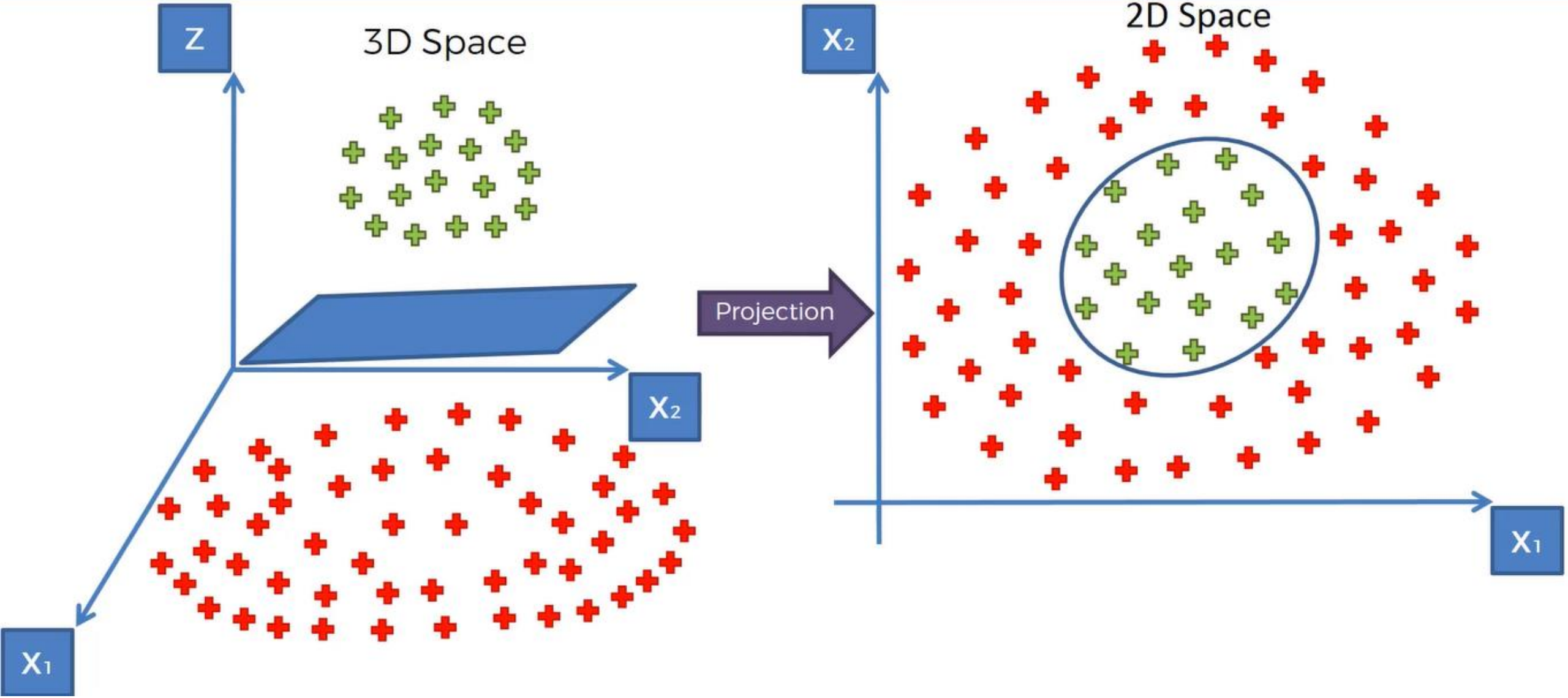
Figure 1: Sample data points in $\Re^2$. Blue diamonds are positive examples and red squares are negative examples.

Kernel SVM Intuition

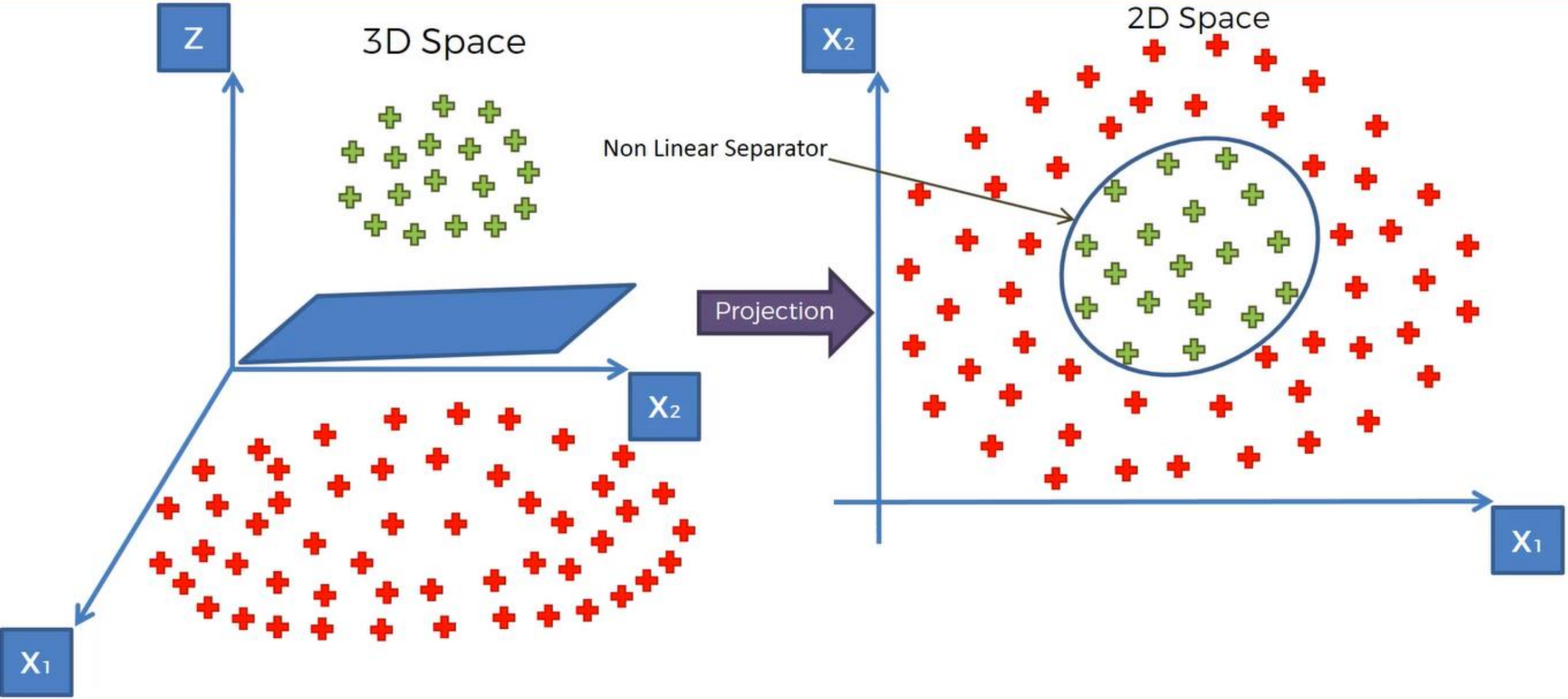
Mapping to a Higher Dimension



Projecting back to 2D Space



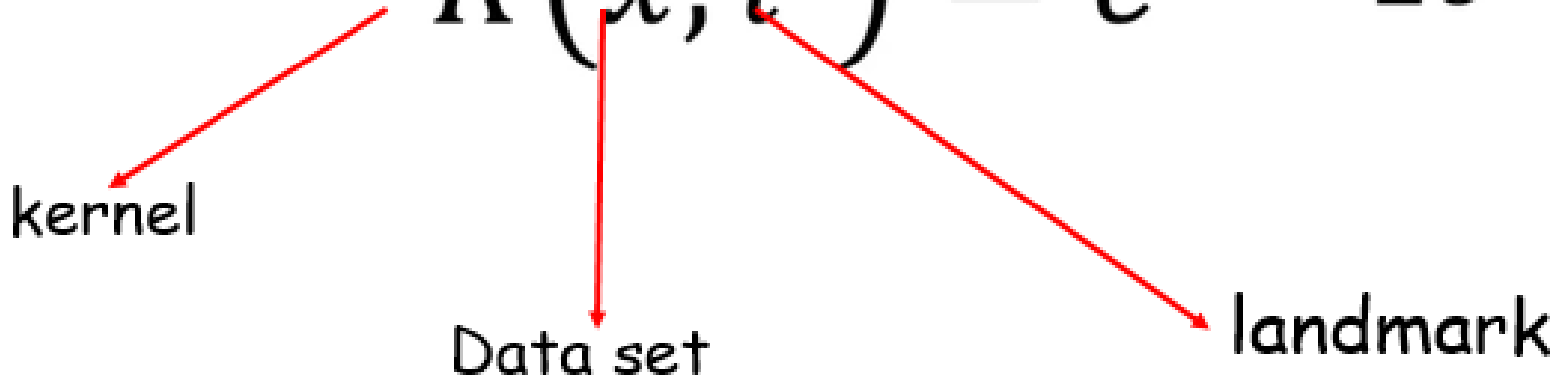
Projecting back to 2D Space



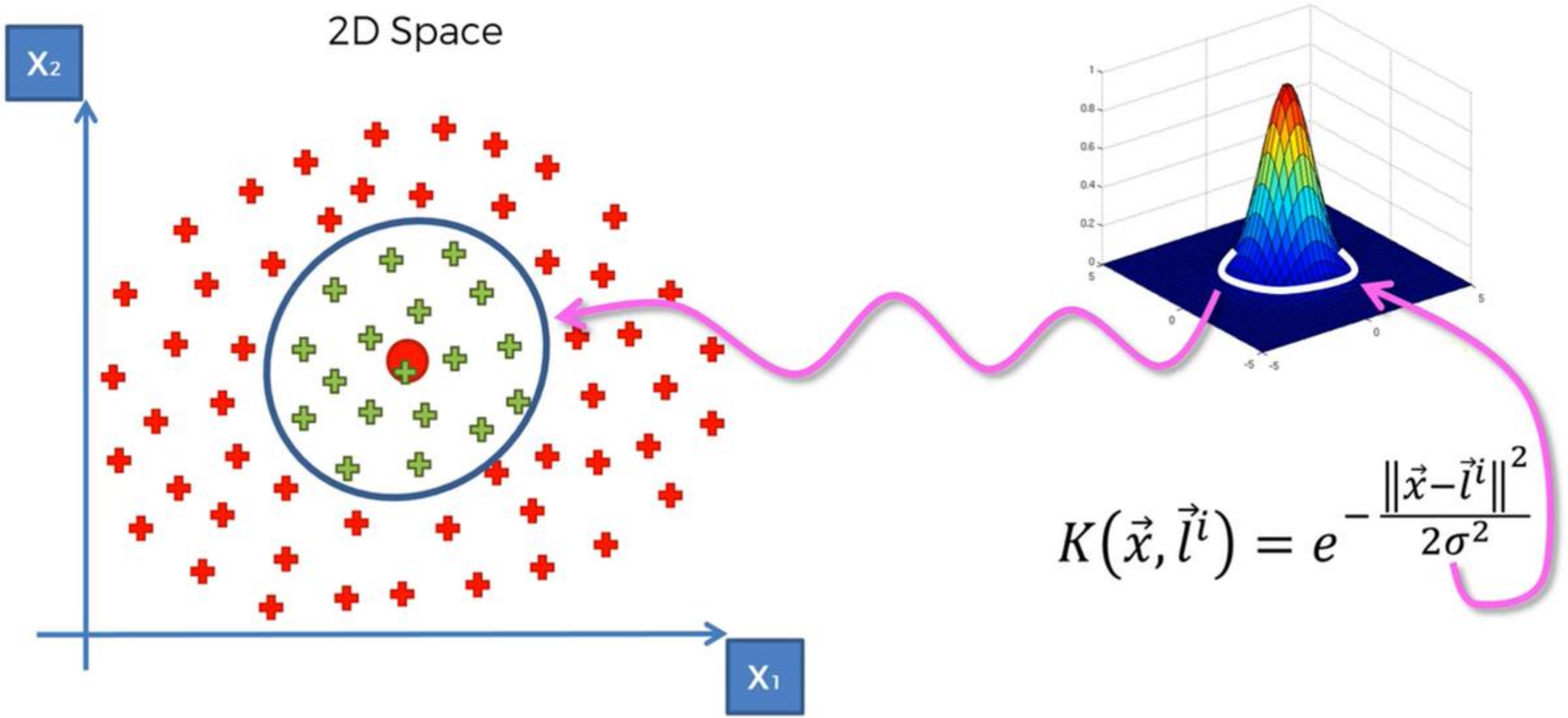
The Kernel Trick

The Gaussian RBF Kernel

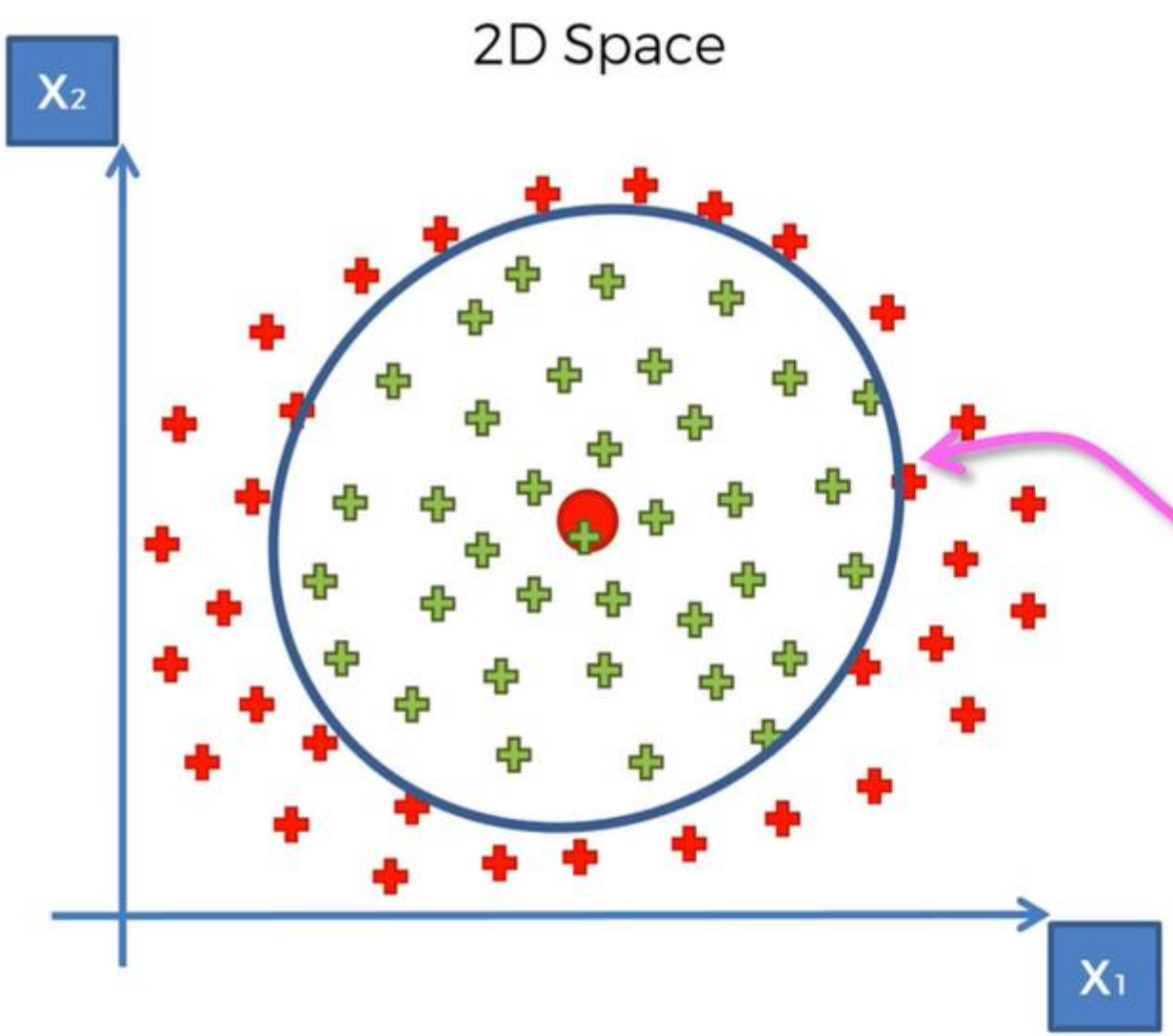
$$K(\vec{x}, \vec{l}^{i}) = e^{-\frac{\|\vec{x}-\vec{l}^{i}\|^2}{2\sigma^2}}$$



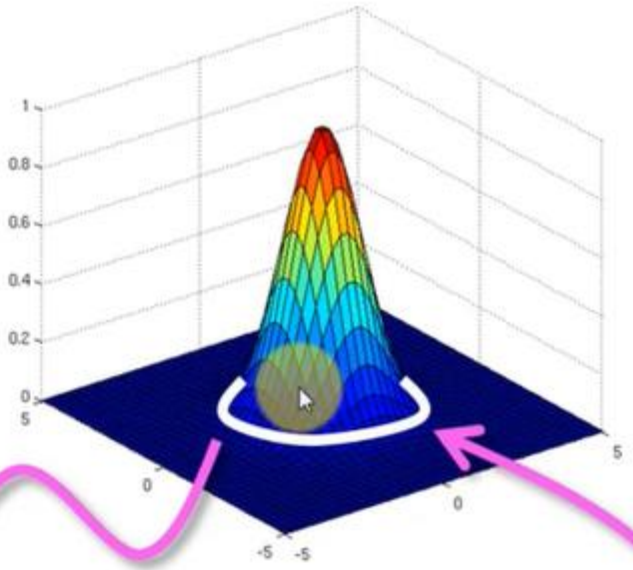
The Gaussian RBF Kernel



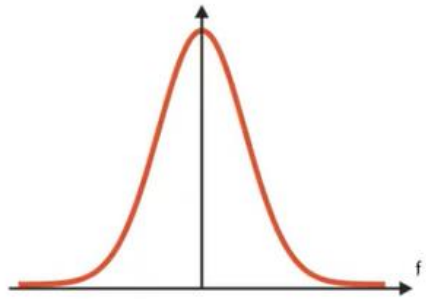
The Gaussian RBF Kernel

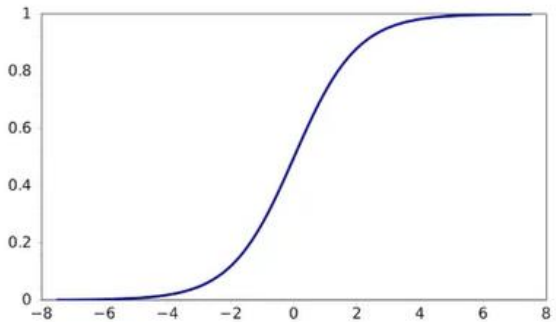


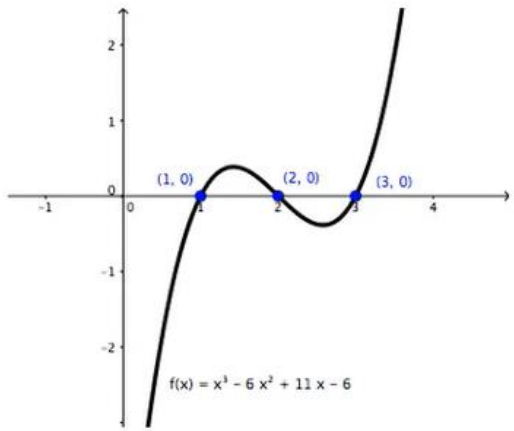
σ



$$K\left(\vec{x}, \vec{l}^{i}\right) = e^{-\frac{\left\|\vec{x}-\vec{l}^{i}\right\|^{2}}{2\sigma^{2}}}$$







Gaussian RBF Kernel

$$K\left(\vec{x}, \vec{l}^{i}\right) = e^{-\frac{\left\|\vec{x}-\vec{l}^{i}\right\|^{2}}{2\sigma^{2}}}$$

Sigmoid Kernel

$$K(X, Y) = \tanh(\gamma \cdot X^{T}Y + r)$$

Polynomial Kernel

$$K(X, Y) = (\gamma \cdot X^{T}Y + r)^{d}, \gamma > 0$$

SVM

Nonlinear Example

Example

We are given the following positively labeled data points in $\Re^2$:

$$\left\{\begin{pmatrix} 2 \\ 2 \end{pmatrix}, \begin{pmatrix} 2 \\ -2 \end{pmatrix}, \begin{pmatrix} -2 \\ -2 \end{pmatrix}, \begin{pmatrix} -2 \\ 2 \end{pmatrix}\right\}$$

and the following negatively labeled data points in $\Re^2$ (see Figure 5):

$$\left\{\begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ 1 \end{pmatrix}\right\}$$

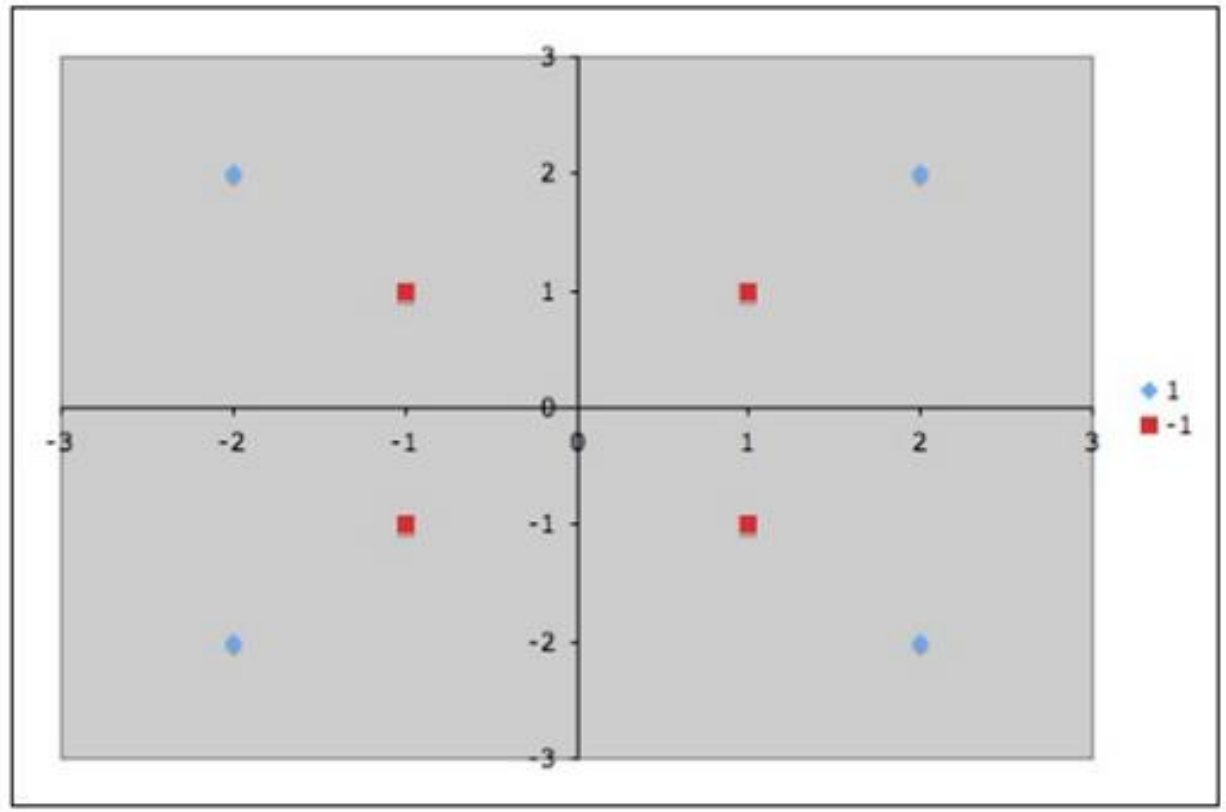


Figure 5: Nonlinearly separable sample data points in $\Re^2$. Blue diamonds are positive examples and red squares are negative examples.

Define

$$\Phi_1 \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{cases} \begin{pmatrix} 4 - x_2 + |x_1 - x_2| \\ 4 - x_1 + |x_1 - x_2| \end{pmatrix} & \text{if } \sqrt{x_1^2 + x_2^2} > 2 \\ \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} & \text{otherwise} \end{cases} \quad (1)$$

Example

- Discover a separating hyperplane that accurately discriminates the two classes.
- It is obvious that no such hyperplane exists in the input space.
- Therefore, we must use a nonlinear SVM.

Deep Learning

What is Deep Learning?

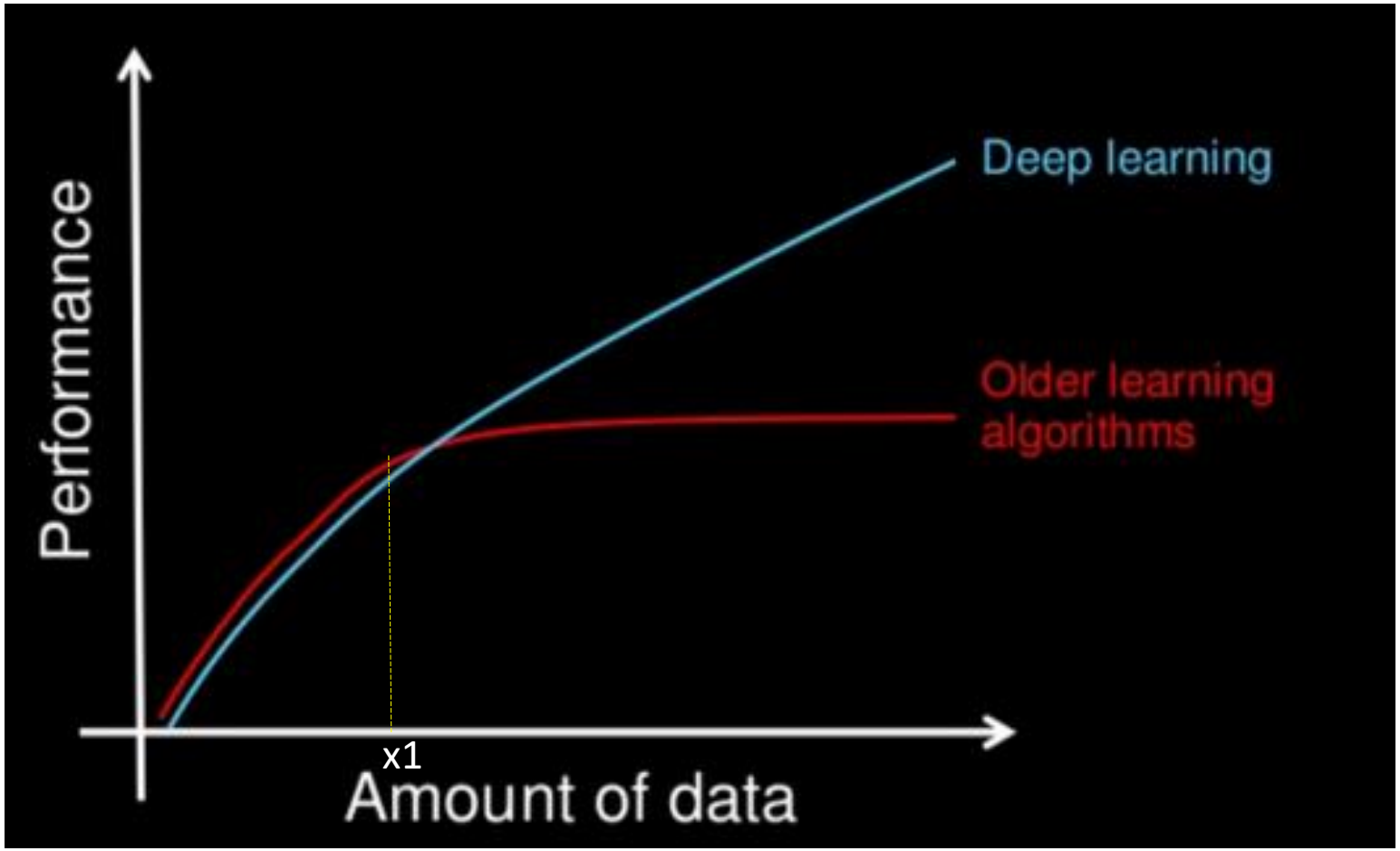
*Deep refers to the **number of layers***

Deep Learning is a deep neural networks generally.

"very large neural networks & huge amounts of data"

Why Deep Learning?

- Performance…



Types of deep learning models

- Convolutional neural networks (CNNs)
- Recurrent neural networks (RNNs)
- Autoencoders and variational autoencoders
- Generative adversarial networks (GANs)
- Diffusion models
- Transformer models

Applications (1): Sentiment Analysis

The process of identifying and categorizing **opinions expressed in a piece of text**, to determine whether the writer's attitude towards a particular topic, product, etc. is **positive, negative, or neutral**.

The best way to hope for any chance of enjoying this film is by lowering your expectations.

The show starts out as competent but unremarkable and gradually grows into something of considerable power.

Applications (2): Question Answering

Yesterday Joy travelled to school.

Yesterday Rai went back to the beach.

This morning Joy travelled to market.

Bill went back to the cinema yesterday.

Rai went to the Hotel this morning.

Joy went back to the dining room this afternoon.

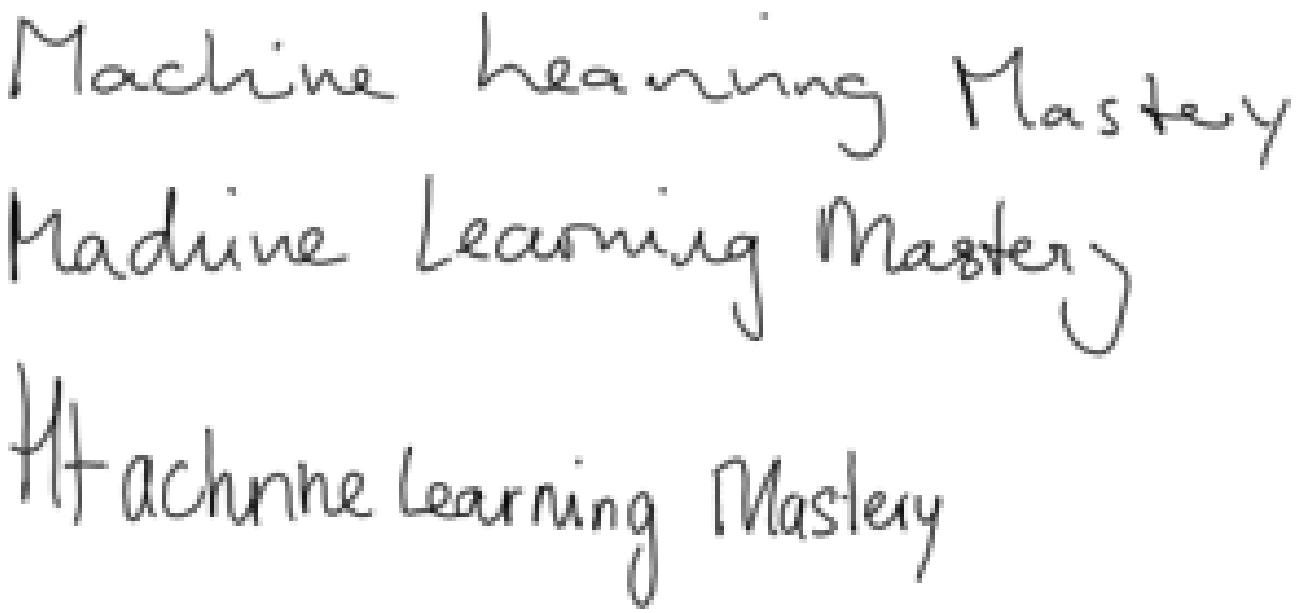
Question: Where was Rai before went to Hotel?

Applications (3): Language Translation

can a man live without food?

ಒಬ್ಬ ಮನುಷ್ಯನು ಆಹಾರವಿಲ್ಲದೆಯೇ ಜೀವಿಸಬಹುದೇ?

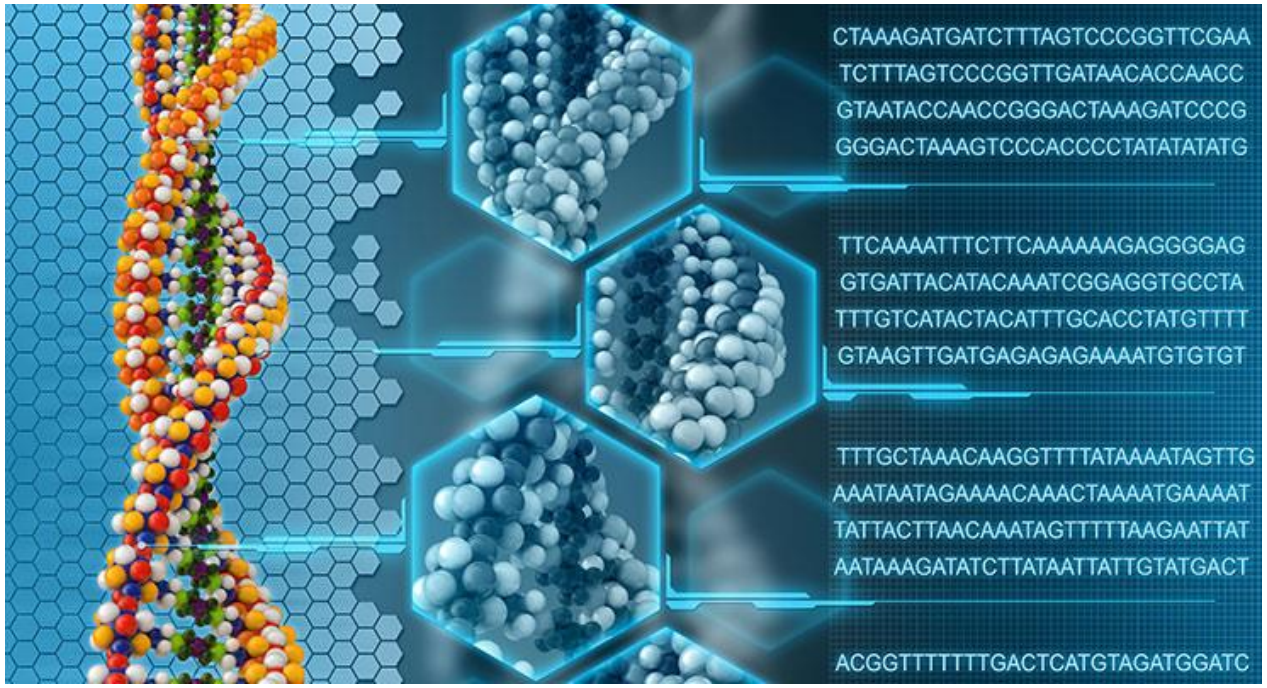
Applications (6): Automatic Handwriting Generation



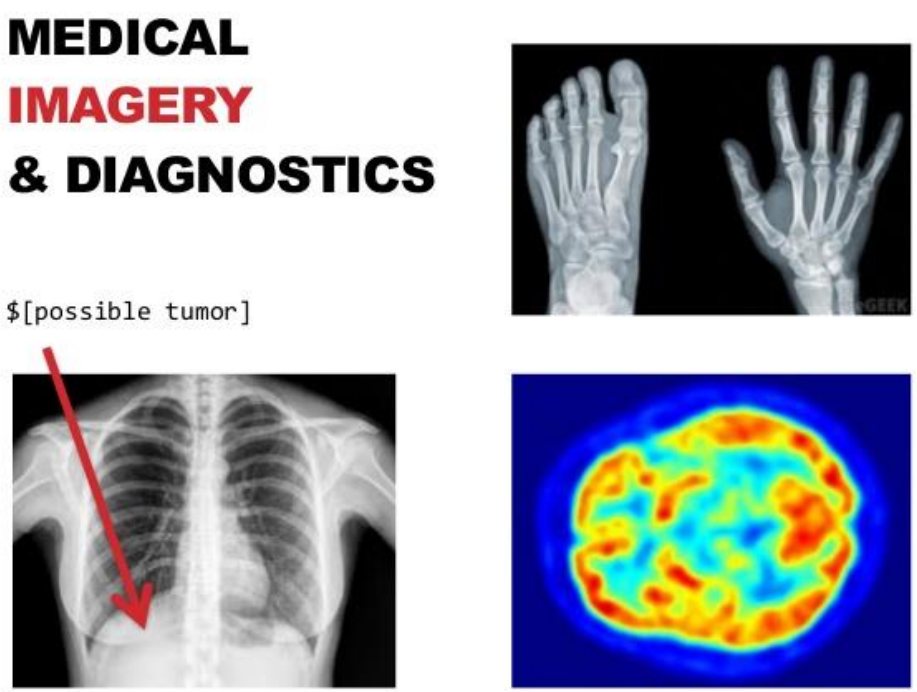
Machine Learning Mastery

Applications (7): Deep Learning in Medicine

Genomics for personalized medicine
Genome Sequencing for aging problem



Better and faster diagnoses –
MRIs, CT scans, X-rays



Applications (8): Recommender Systems

amazon

LinkedIn ebay YAHOO!

YouTube facebook

Applications (9): Self Driving Cars

+ Reduce traffic – communicating with other vehicles

+ Speed – within cities

+ Reach destination – with in short time

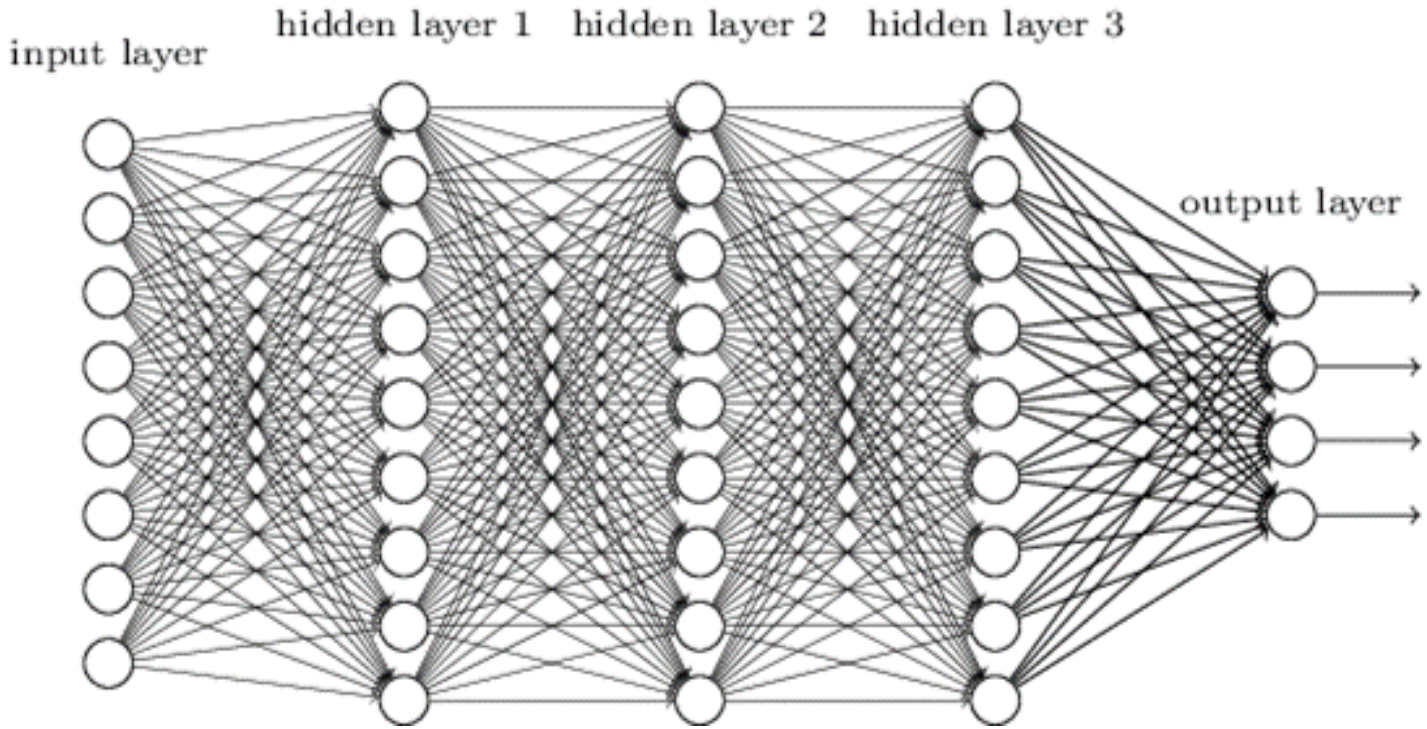
+ No road accidents

+ No traffic jams

- Drivers jobs ???
- Real estate business ???

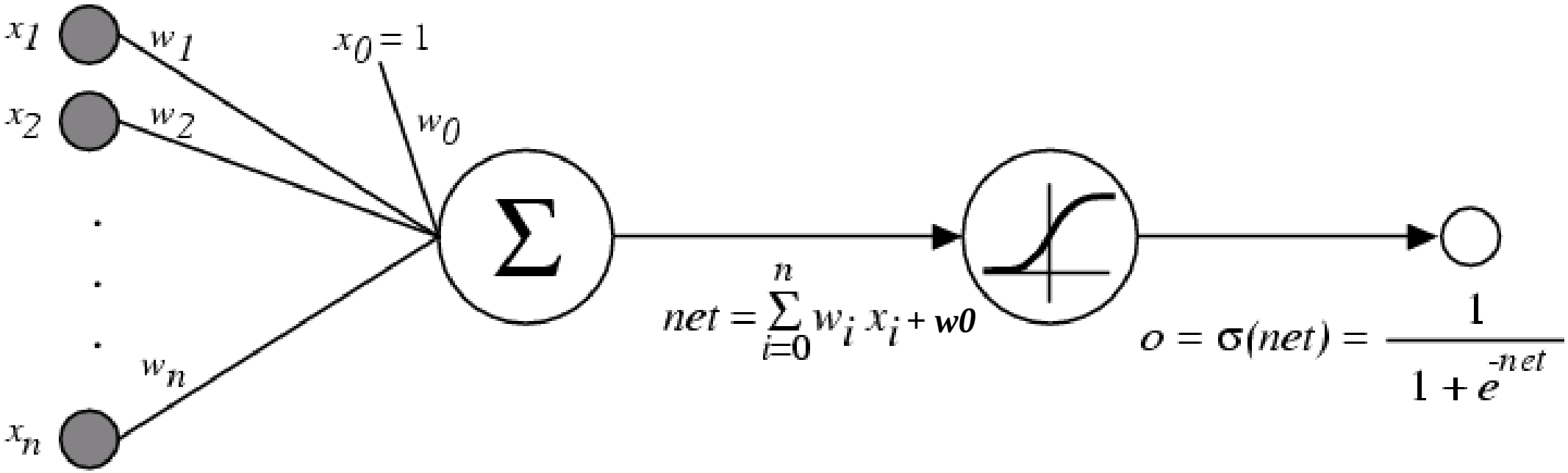
Architecture of Deep Neural Network

- Input layer
- Two or more hidden layers
- Output layer



- Deep architectures are composed of multiple levels of non-linear operations - neural nets with many hidden layers.

Architecture of Deep Neural Network



Challenges in Deep Learning

Data availability: It requires **large amounts of data** to learn from. For using deep learning it's a big concern to gather as much data for training.

Computational Resources: For training the deep learning model, it is **computationally expensive** because it requires specialized hardware like **GPUs and TPUs**.

Time-consuming: While working on sequential data depending on the computational resource it can take **very large** even in days or months.

Interpretability: Deep learning models are complex, it works like a **black box,** it is very difficult to interpret the result.

Overfitting: when the model is **trained again and again**, it becomes too specialized for the training data, leading to overfitting and poor performance on new data.

Hyperparameters in Neural Networks

Learning rate: This hyperparameter controls the **step size** taken by the optimizer during each iteration of training. **Too small** a learning rate can result in **slow convergence**, while **too large** a learning rate can lead to **instability and divergence**.

Epochs: This hyperparameter represents the **number of times the entire training dataset is passed through the model** during training. Increasing the number of epochs can **improve the model's performance** but may lead to overfitting if not done carefully.

Number of layers: This hyperparameter determines the depth of the model, which can have a significant impact on its **complexity and learning ability**.

Hyperparameters in Neural Networks

Number of nodes per layer: This hyperparameter determines the **width of the model**, influencing its capacity to represent **complex relationships in the data**.

Architecture: This hyperparameter determines the overall structure of the neural network, including the number of layers, the number of neurons per layer, and the connections between layers. The optimal architecture depends on the complexity of the task and the size of the dataset

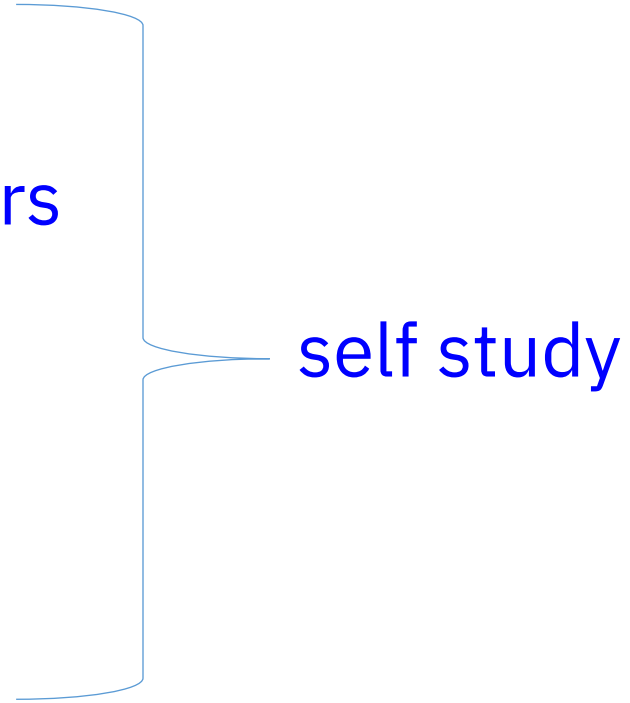
Activation function: This hyperparameter introduces non-linearity into the model, allowing it to learn complex decision boundaries. Common activation functions include sigmoid, tanh, and Rectified Linear Unit (ReLU).

Self study

- **Difference between Machine Learning and Deep Learning**
- **Advantages of Deep Learning**
- **Disadvantages of Deep Learning**
- **Hyperparameters in Support Vector Machine**
- **Hyperparameters in Bagging and Boosting models**

Types of deep learning models

- Convolutional neural networks (CNNs)
- Recurrent neural networks (RNNs)
- Autoencoders and variational autoencoders
- Generative adversarial networks (GANs)
- Diffusion models
- Transformer models



Types of deep learning models

- Convolutional neural networks (CNNs)
- Recurrent neural networks (RNNs)
- Autoencoders and variational autoencoders
- Generative adversarial networks (GANs)
- Diffusion models
- Transformer models

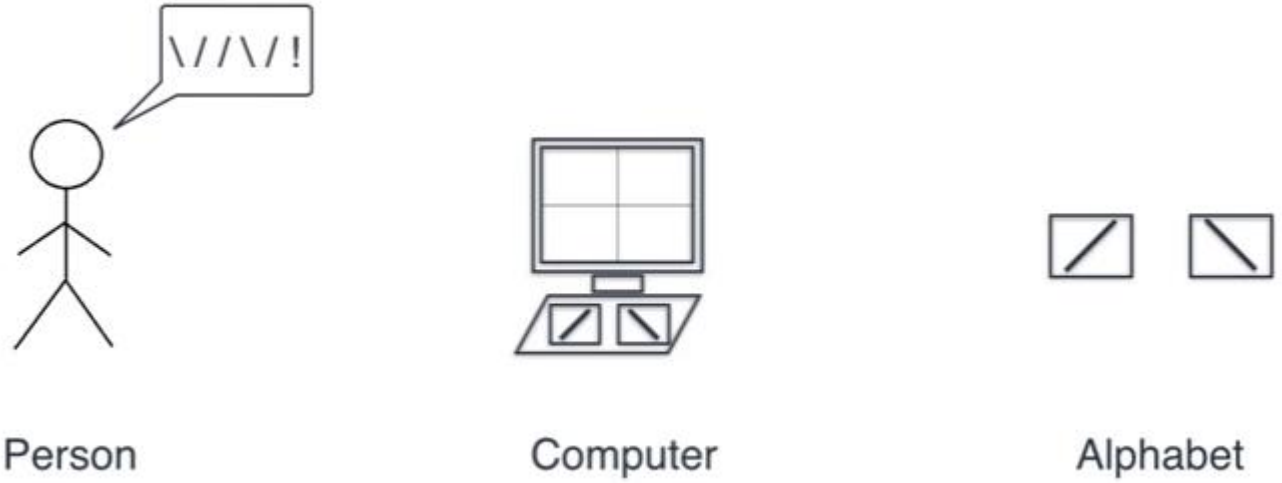
Convolutional Neural Networks

Convolutional Neural Networks

- Convolutional neural networks (CNNs) are widely used tools for deep learning.
- Suitable for applications such as
  - Images
  - Text
  - Signals, and
  - other continuous responses - as inputs

Convolutional Neural Networks

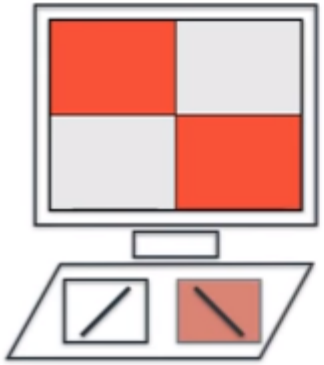
Consider a simple world scenario

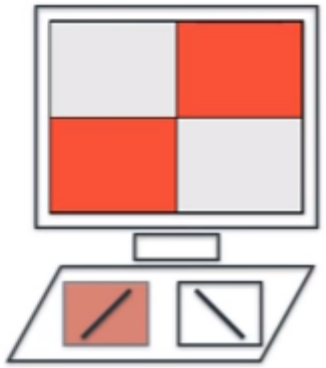


- Computer – 2 X 2 pixel resolution
- Keyboard with only two alphabets / and \

A simple image recognition system

- If you press the key in the right, you can see the image in the screen.

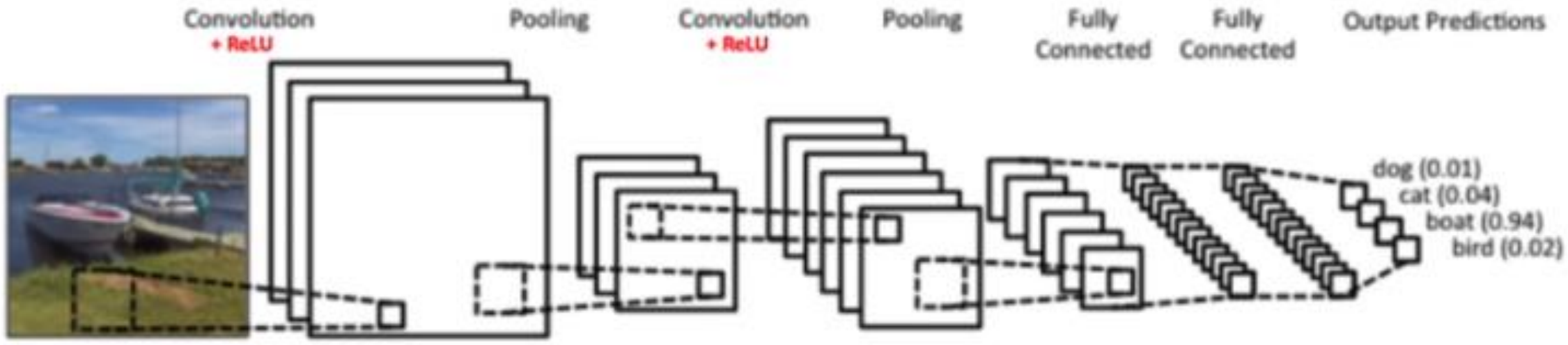




- If you press the key in the left, you can see the image in the screen.

Components of a Convolutional Network

- The convolutional layer
- The Pooling layer [optional]
- Fully Connected Layer (output layer)



The Convolution Layer

- Input image of size 6*6.
- Filter/ weight matrix 3*3, which extracts certain features from the images

INPUT IMAGE

18	54	51	239	244	188
55	121	75	78	95	88
35	24	204	113	109	221
3	154	104	235	25	130
15	253	225	159	78	233
68	85	180	214	245	0

Filter/weight

1	0	1
0	1	0
1	0	1

429

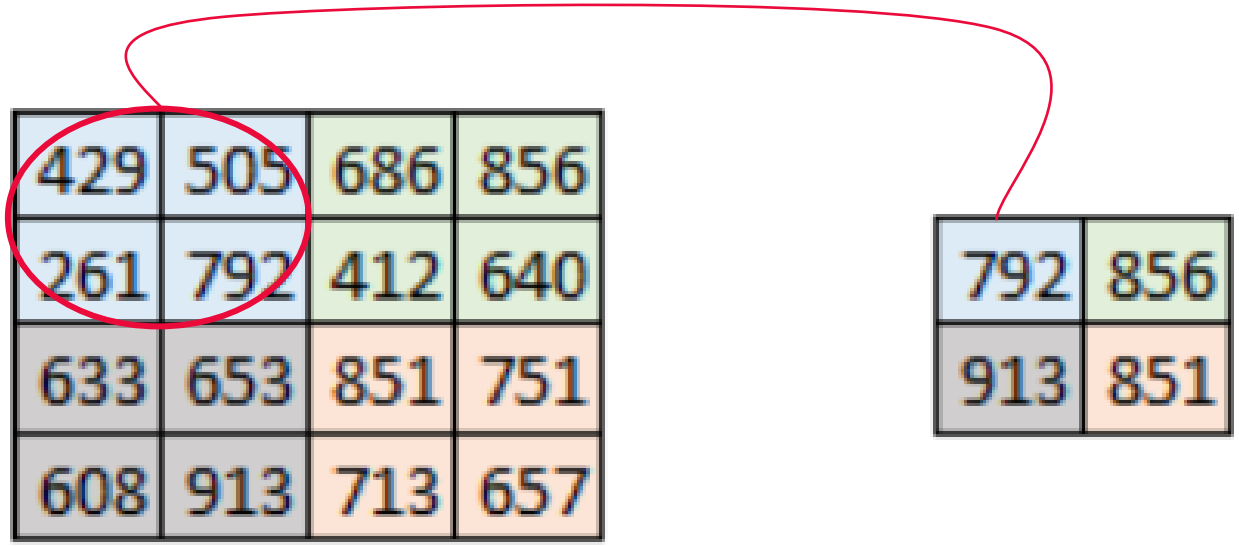
This weight shall now **slides across the image** such that all the pixels are covered at least once, to give a convolved output

The Pooling Layer

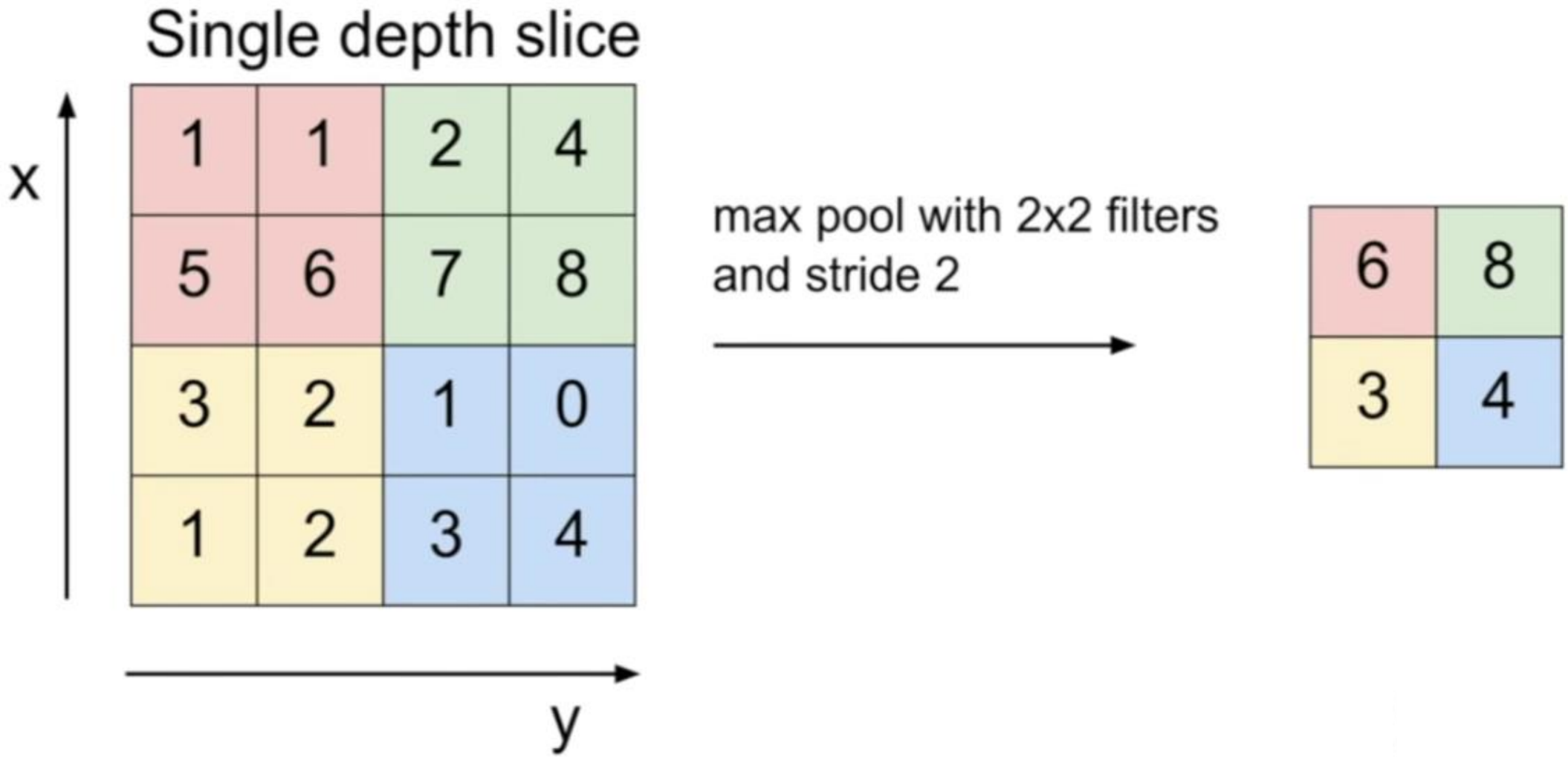
- When the images are too large, to reduce the number of trainable parameters, pooling layer is used.
- Periodically introduce pooling layers between subsequent **convolution layers**.
- Pooling is done for the purpose of reducing the spatial size of the image.
- The most common form of pooling layer generally applied is the max pooling.

Example

- We have stride as 2, while pooling size also as 2.
- The max operation is applied to each depth dimension of the convolved output.
- The 4*4 convolved output has become 2*2 after the max pooling operation.

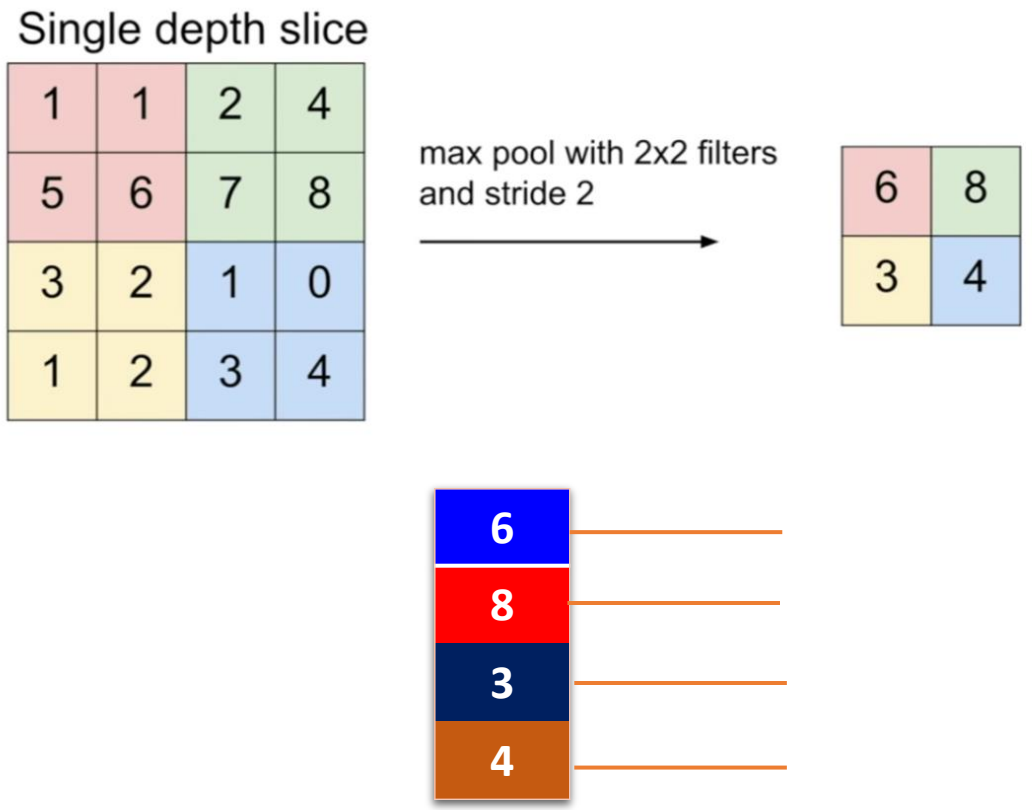


MAX POOLING

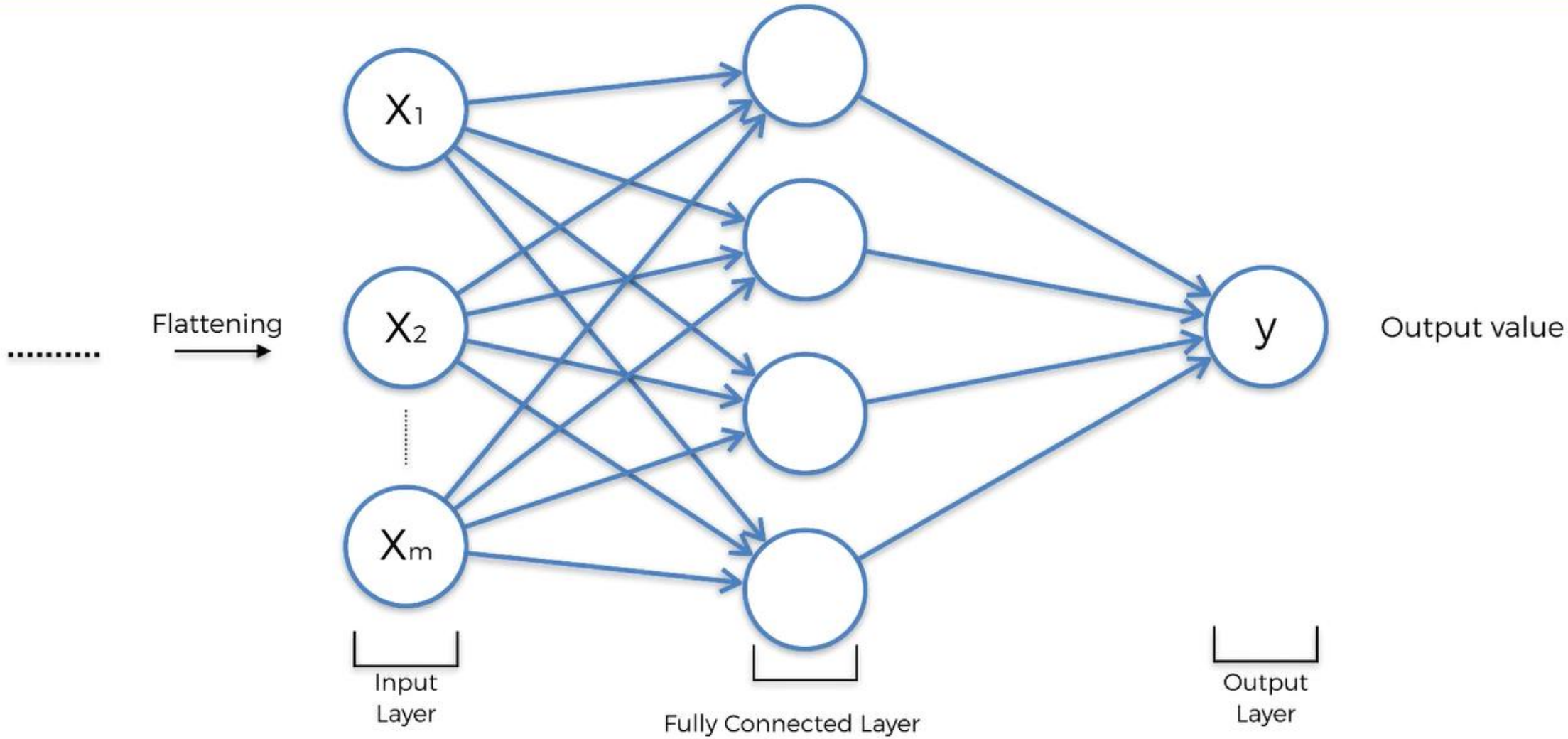

Single depth slice
x
y
1 1 2 4
5 6 7 8
3 2 1 0
1 2 3 4
max pool with 2x2 filters
and stride 2
6 8
3 4

Fully Connected & Output layer

- After multiple layers of convolution and pooling, we need the output in the form of a **class**.
  - **convolution layers** would only be able to extract features
  - **pooling layers** reduce the number of parameters from the original images
- **Fully connected layer** to generate an output equal to the number of classes we need.

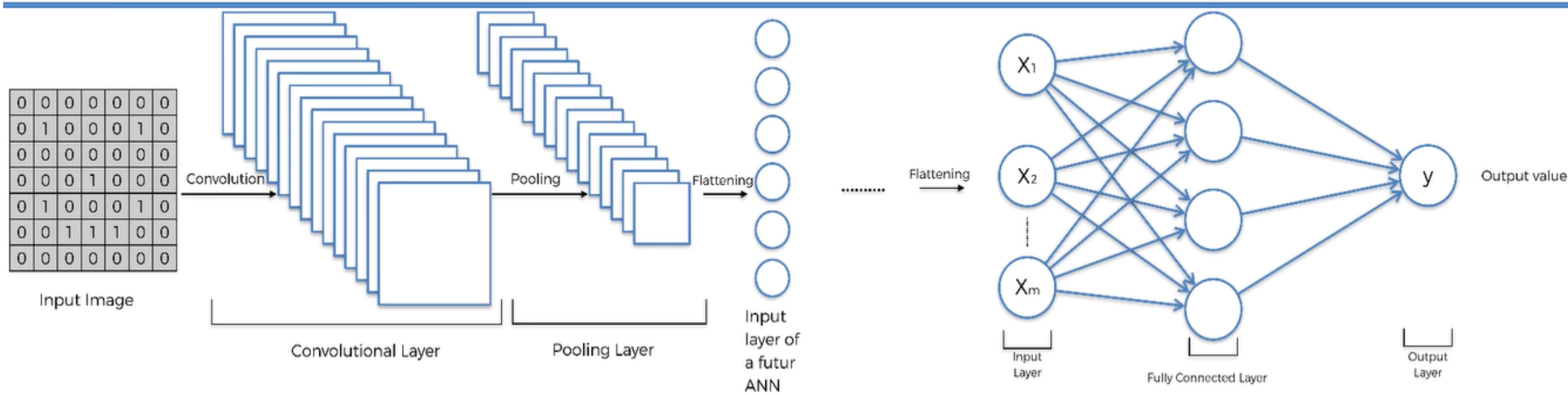


Step 4 - Full Connection



Step 3 - Flattening

Step 4 - Full Connection



Reinforcement Learning

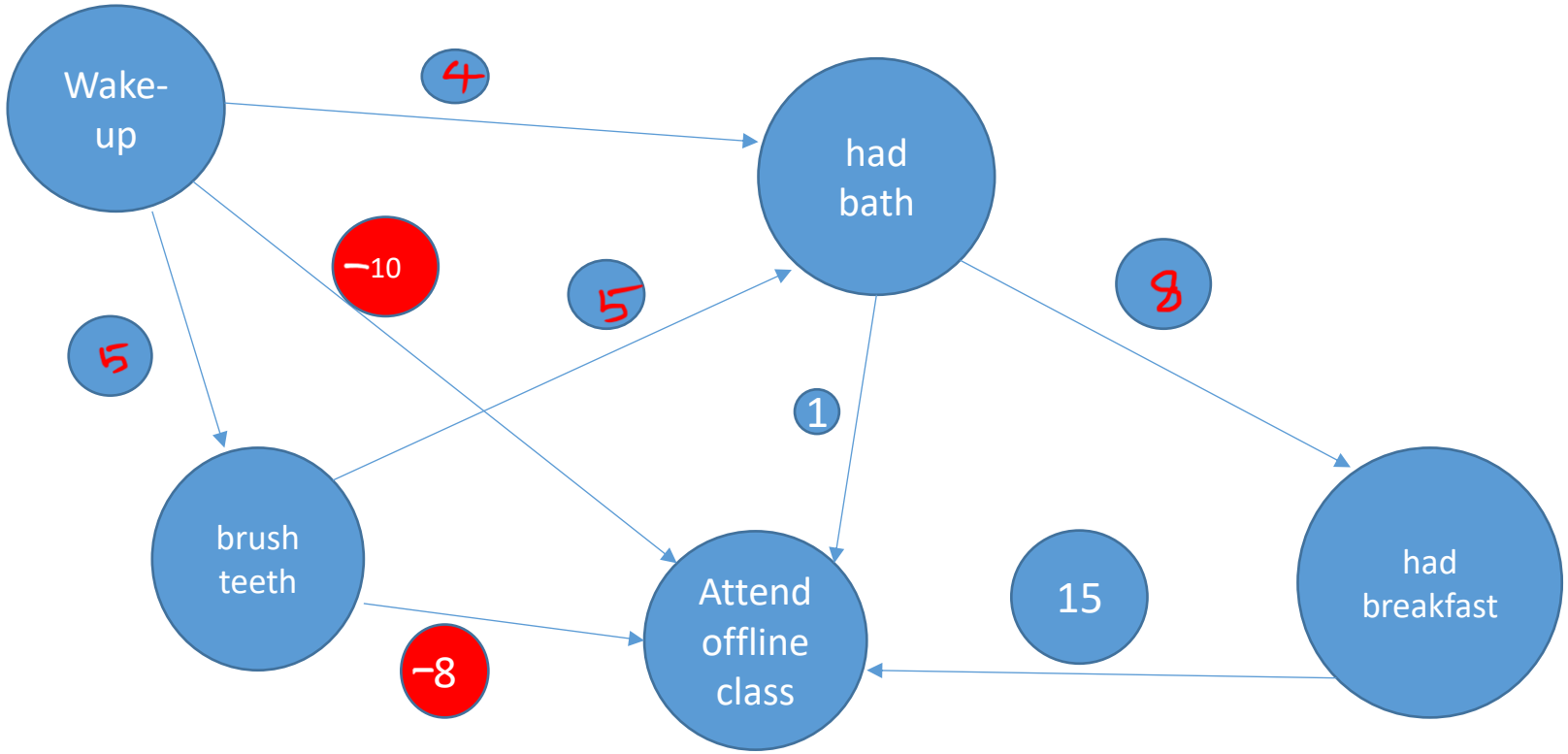
II semester BDA

What is Reinforcement Learning?

- **Reinforcement Learning** is concerned with how **software agent should take actions in an environment** that helps you to **maximize** some portion of the **cumulative reward**.

Reward based learning

- Reinforcement Learning
  - Rewards (+ve or –ve)
  - Wake-up, brushed teeth, had bath, had breakfast, reach college



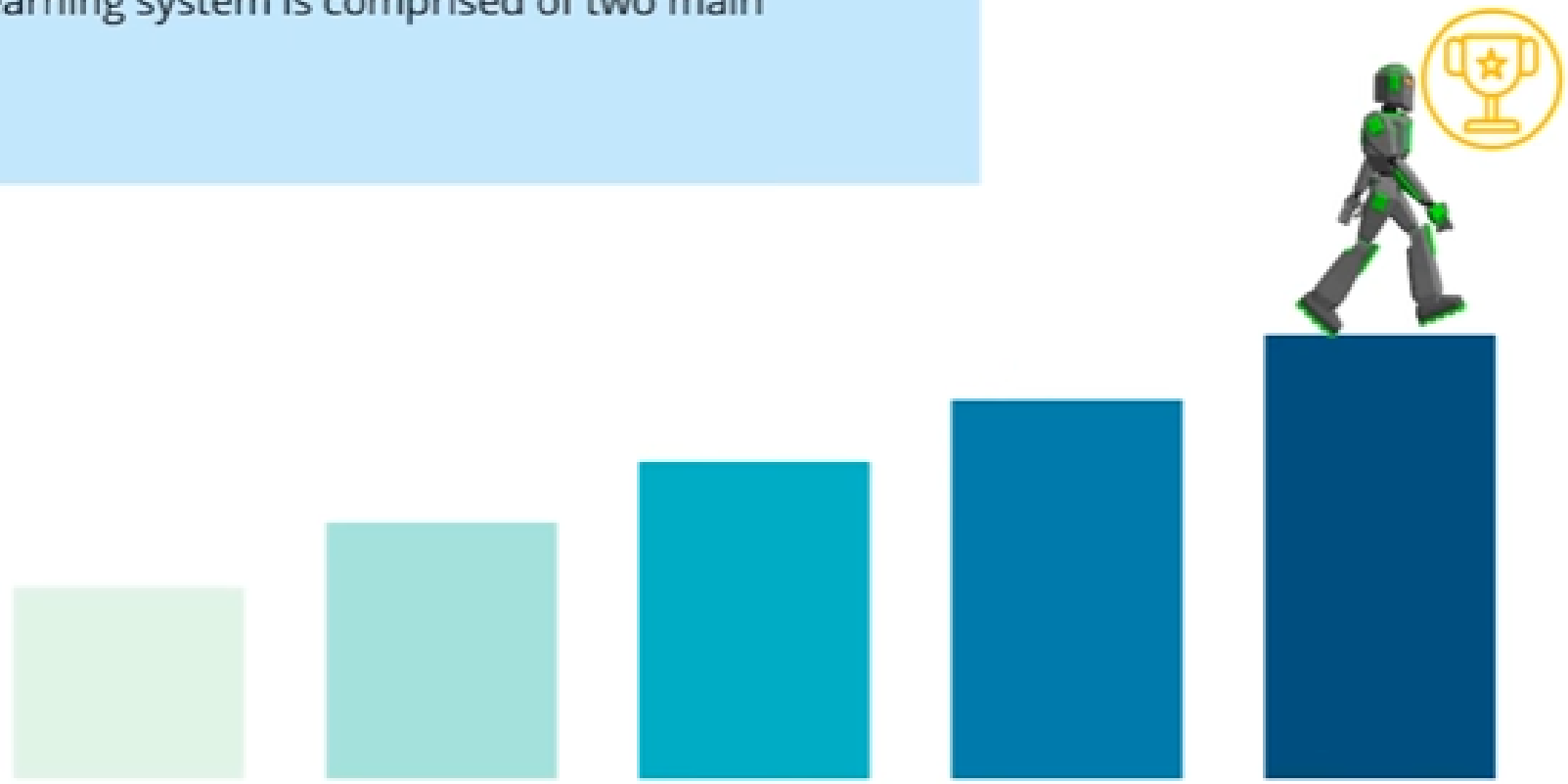
Reinforcement Learning vs. Supervised Learning

Parameters	Reinforcement Learning	Supervised Learning
Decision style	Take decisions sequentially.	Decision is made on the input given at the beginning.
Works on	Interacting with the environment.	Examples or given sample data.
Dependency on decision	Decision is dependent.	Decisions are independent of each other.
Best suited	Supports and work better in AI, where human interaction is prevalent.	It is mostly operated with an interactive software system or applications.
Example	Chess game	Object recognition

Reinforcement Learning Process

Reinforcement Learning system is comprised of two main components:

- Agent
- Environment



Reinforcement Learning Definitions

Agent: The RL algorithm that learns from trial and error

Environment: The world through which the agent moves

Action (A): All the possible steps that the agent can take

State (S): Current condition returned by the environment

Reinforcement Learning Definitions

Reward (R): An instant return from the environment to appraise the last action

Policy (π): The approach that the agent uses to determine the next action based on the current state

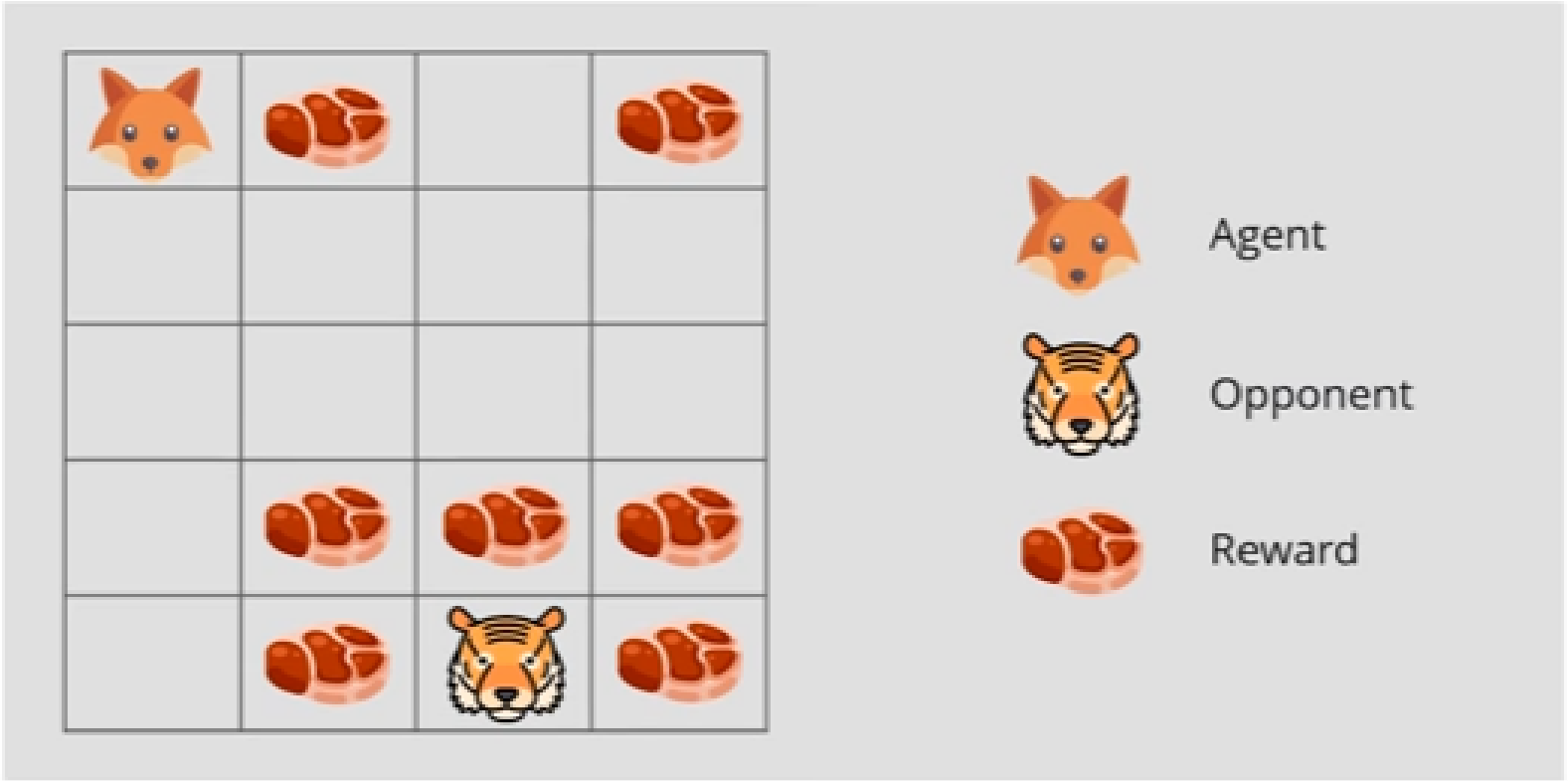
Value (V): The expected long-term return with discount, as opposed to the short-term reward R

Action-value (Q): This similar to Value, except, it takes an extra parameter, the current action (A)

Exploration & Exploitation

Exploitation is about using the already known exploited information to heighten the rewards

Exploration is about exploring and capturing more information about an environment

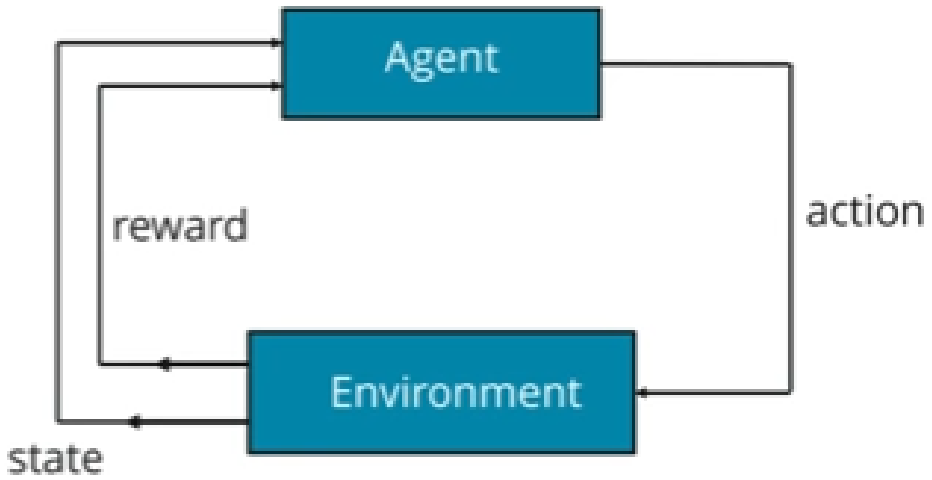


Markov Decision Process

The mathematical approach for mapping a solution in reinforcement learning is called *Markov Decision Process* (MDP)

The following parameters are used to attain a solution:

- Set of actions, A
- Set of states, S
- Reward, R
- Policy, π
- Value, V

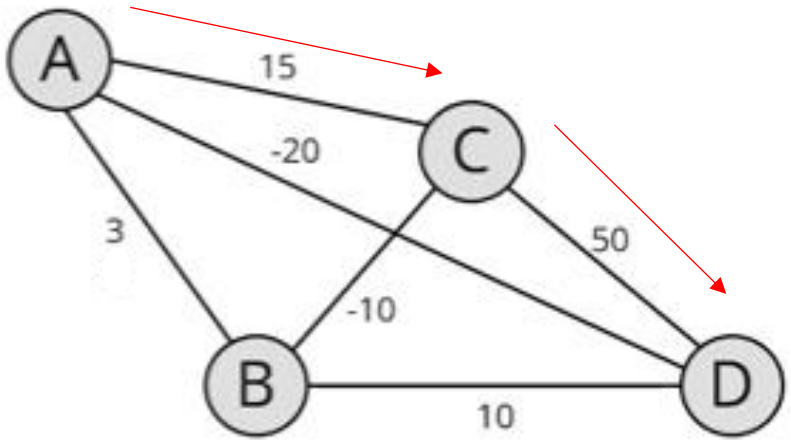


Markov Decision Process – Shortest Path Problem

Goal: Find the shortest path between A and D with minimum possible cost (maximum reward)

In this problem,

- Set of states are denoted by nodes i.e. {A, B, C, D}
- Action is to traverse from one node to another {A -> B, C -> D}
- Reward is the cost represented by each edge
- Policy is the path taken to reach the destination {A -> C -> D}



Q – Learning Algorithm

1. Set the gamma parameter, and environment rewards in matrix R
2. Initialize matrix Q to zero
3. Select a random initial state
4. Set initial state = current state
5. Select one among all possible actions for the current state
6. Using this possible action, consider going to the next state
7. Get maximum Q value for this next state based on all possible actions
8. Compute: Q(state, action) = R(state, action) + Gamma * Max[Q(next state, all actions)]
9. Repeat above steps until current state = goal state


Thank you