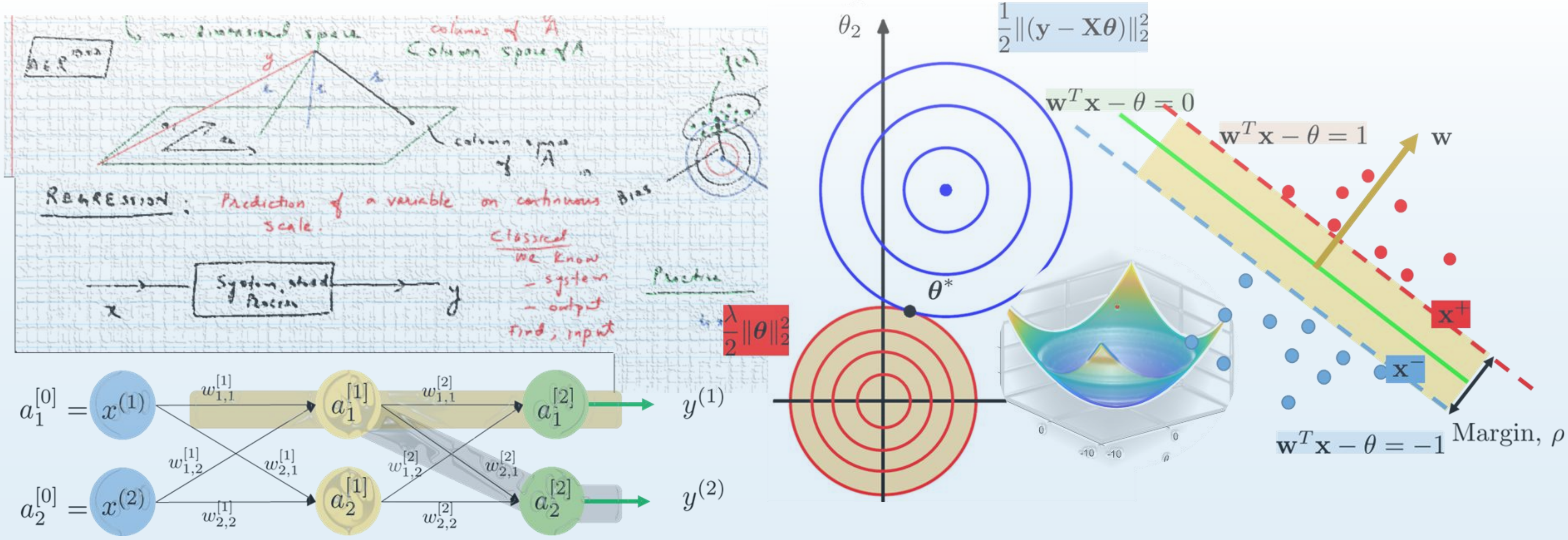


Machine Learning

Perceptron Classifier

School of Science and Engineering

https://www.zubairkhalid.org/ee514_2025.html

LUMS
A Not-for-Profit University

Outline

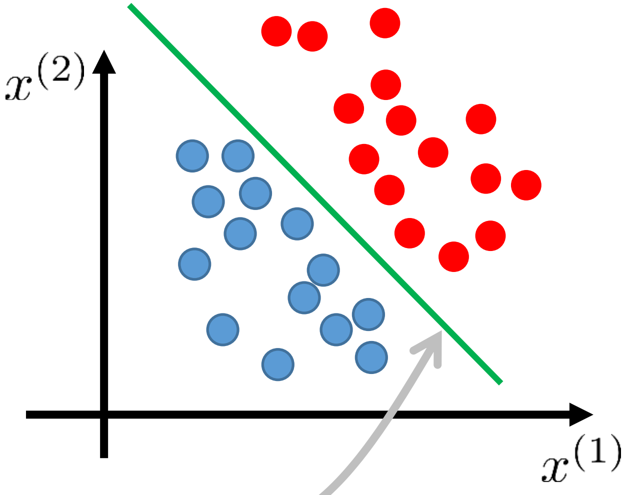
- Perceptron and Perceptron Classifier
- Perceptron Learning Algorithm
  - Geometric Intuition
- Perceptron Learning Algorithm Convergence

Linear Classifiers

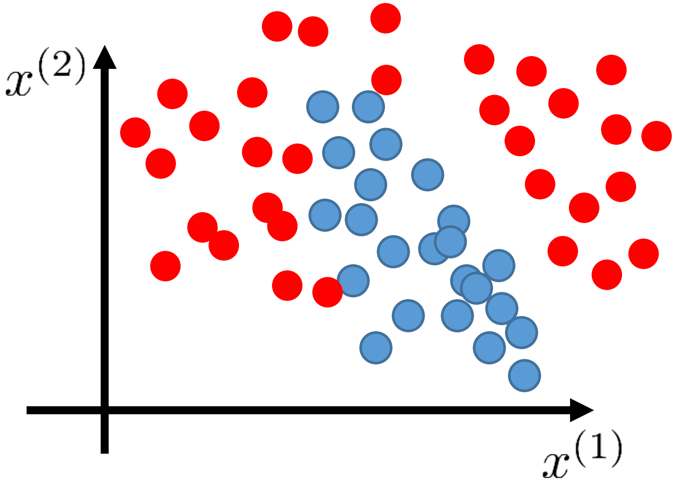
Overview:

- Linear Separability

Linearly Separable



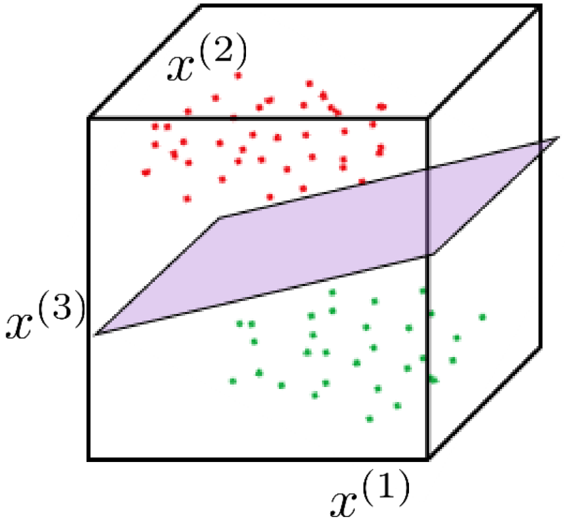
NOT Linearly Separable



- Linear Classifiers

$$h(\mathbf{x}) = \boldsymbol{\theta}^T \mathbf{x} + \theta_0$$

- line in 2D, plane in 3D, hyper-plane in higher dimensions.



We have studied three classifiers:

- kNN (Instance)
- Logistic Regression (Discriminative)

More Discriminative Classifiers:

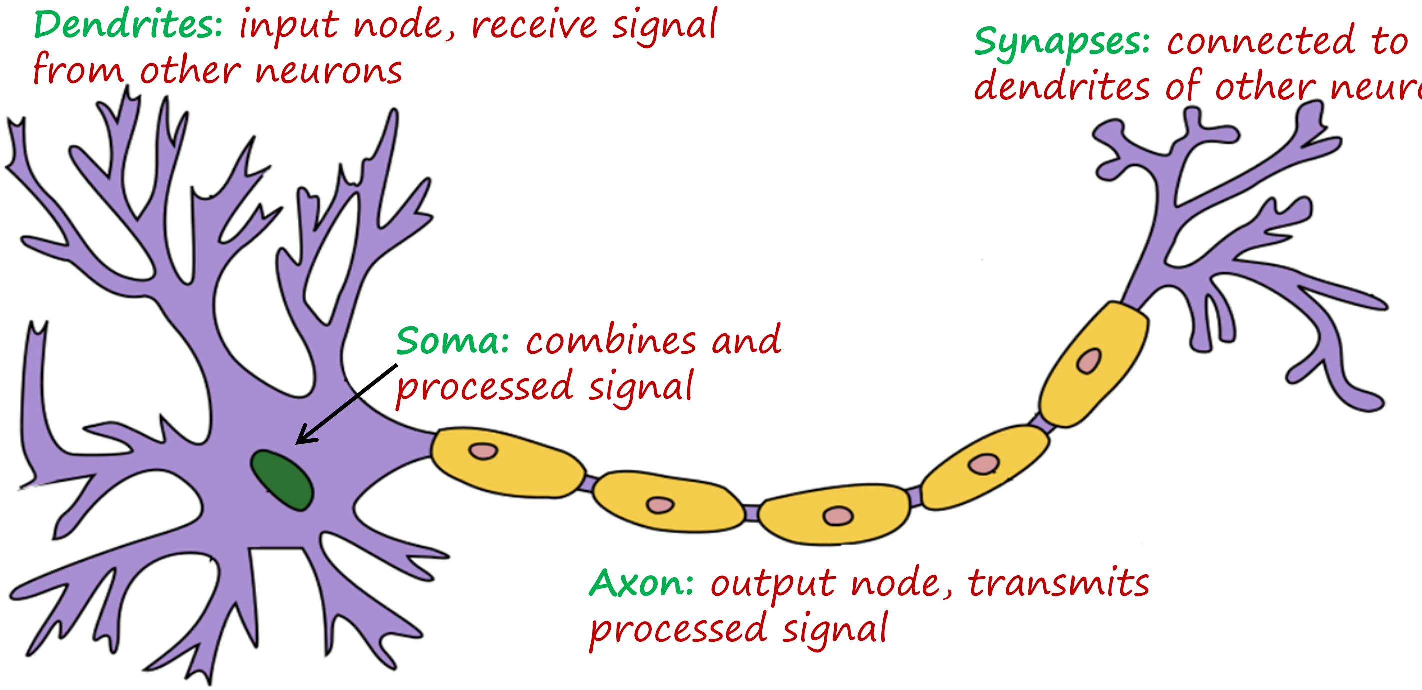
- Perceptron
- Support Vector Machines

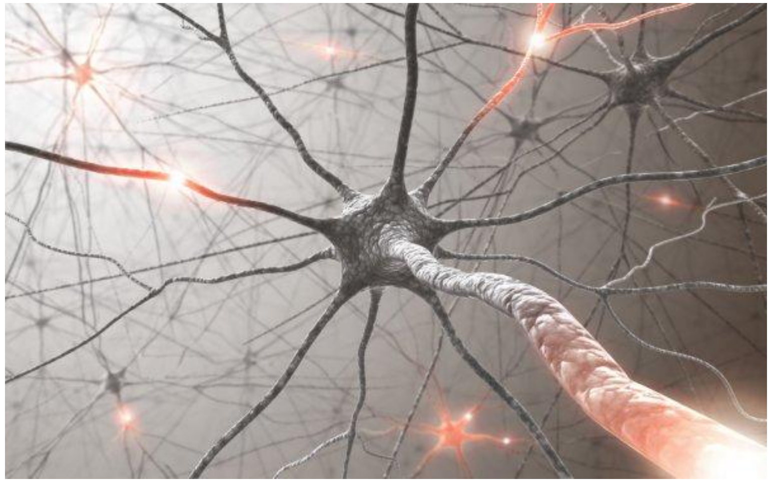
Perceptron Classifier

McCulloch-Pitts (MP) Neuron:

- McCulloch (neuroscientist) and Pitts (logician) proposed a computational model of the biological neuron in 1943.

Biological Neuron (Simplified illustration):





- *Neuron is fired or transmits the signal when it is activated by the combination of input signals.*

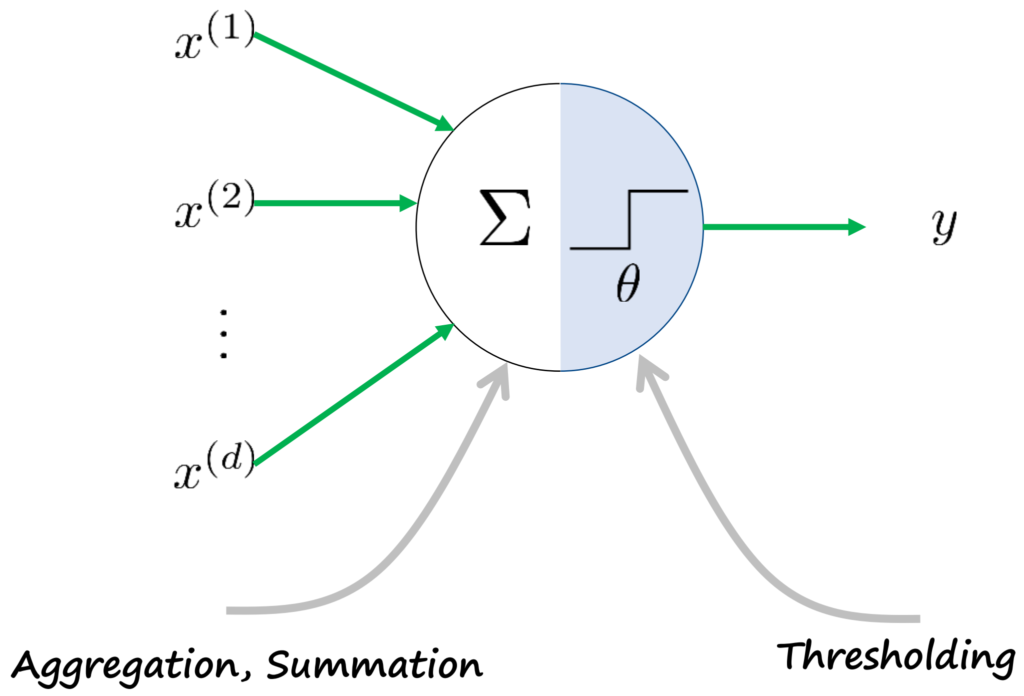
Source: McCulloch-Pitts Neuron — Mankind's First Mathematical Model Of A Biological Neuron | by Akshay L Chandra | Towards Data Science

Perceptron Classifier

McCulloch-Pitts (MP) Neuron:

- d number of boolean inputs $x^{(1)}, x^{(2)}, \ldots, x^{(d)} \in \{0, 1\}$.
- Boolean output, $y \in \{0, 1\}$.
- If sum of inputs is less than θ, the output is zero and one otherwise.
- θ is a thresholding parameter that characterizes the neuron.
- Mathematically;

$$y = \begin{cases} 1 & \text{if} \quad \sum_{i=1}^{d} x^{(i)} \geq \theta \\ 0 & \text{if} \quad \sum_{i=1}^{d} x^{(i)} < \theta \end{cases}$$



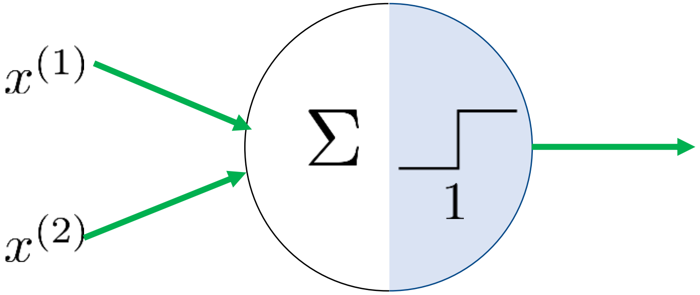
- Idea: Fire the nueron if at least θ number of inputs are active.

Source: McCulloch-Pitts Neuron — Mankind's First Mathematical Model Of A Biological Neuron | by Akshay L Chandra | Towards Data Science

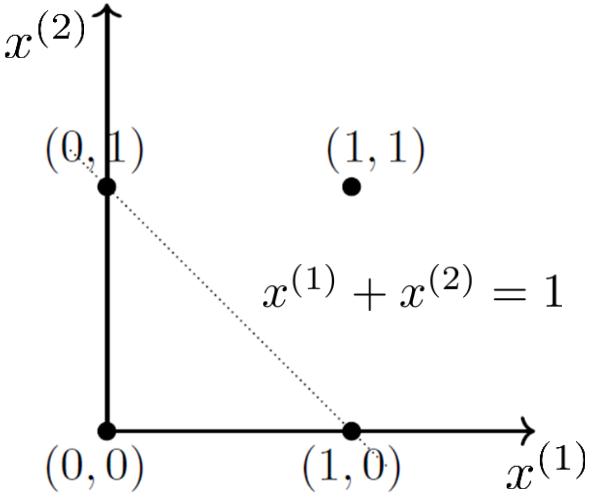
Perceptron Classifier

McCulloch-Pitts Neuron (MP) - Examples:

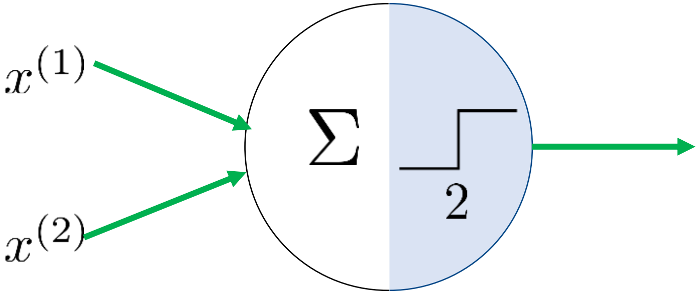
- OR of two inputs.



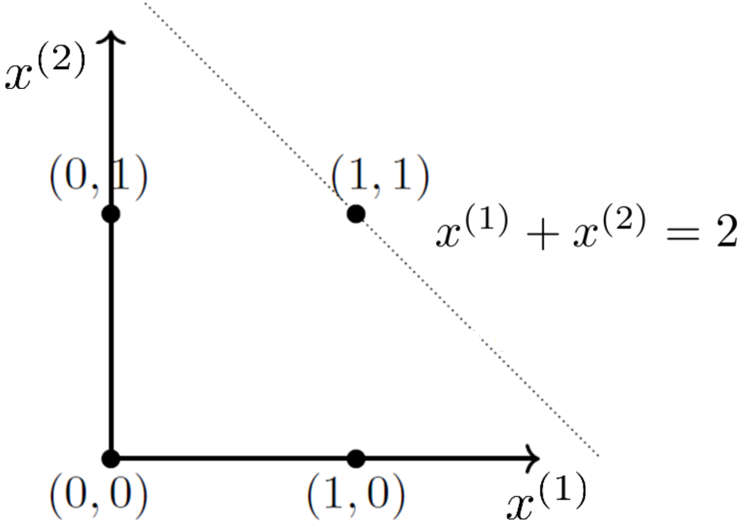
$$y = \begin{cases} 1 & \text{if} \quad x^{(1)} + x^{(2)} \geq 1 \\ 0 & \text{if} \quad x^{(1)} + x^{(2)} < 1 \end{cases}$$



- AND of two inputs.



$$y = \begin{cases} 1 & \text{if} \quad x^{(1)} + x^{(2)} \geq 2 \\ 0 & \text{if} \quad x^{(1)} + x^{(2)} < 2 \end{cases}$$

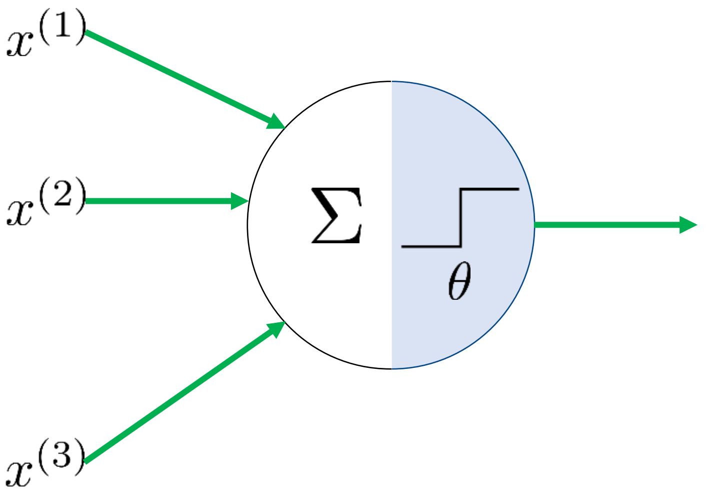


Source: McCulloch-Pitts Neuron — Mankind's First Mathematical Model Of A Biological Neuron | by Akshay L Chandra | Towards Data Science

Perceptron Classifier

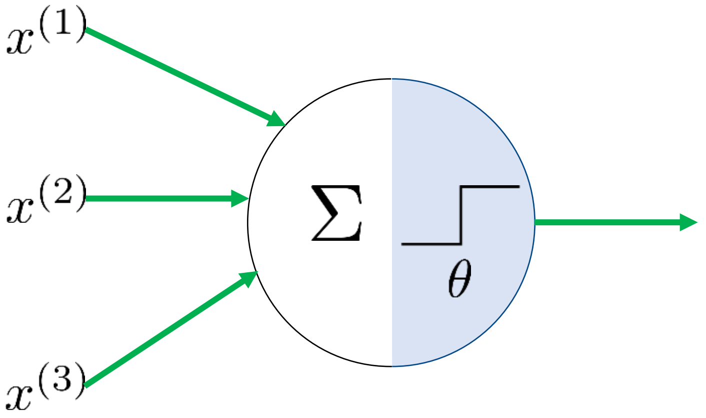
McCulloch-Pitts Neuron (MP) - Examples:

- OR of three inputs.



$$y = \begin{cases} 1 & \text{if} \quad x^{(1)} + x^{(2)} + x^{(3)} \geq 1 \\ 0 & \text{if} \quad x^{(1)} + x^{(2)} + x^{(3)} < 1 \end{cases}$$

- AND of three inputs.



$$y = \begin{cases} 1 & \text{if} \quad x^{(1)} + x^{(2)} + x^{(3)} \geq 3 \\ 0 & \text{if} \quad x^{(1)} + x^{(2)} + x^{(3)} < 3 \end{cases}$$

Source: McCulloch-Pitts Neuron — Mankind's First Mathematical Model Of A Biological Neuron | by Akshay L Chandra | Towards Data Science

Perceptron Classifier

McCulloch-Pitts (MP) Neuron – Limitations:

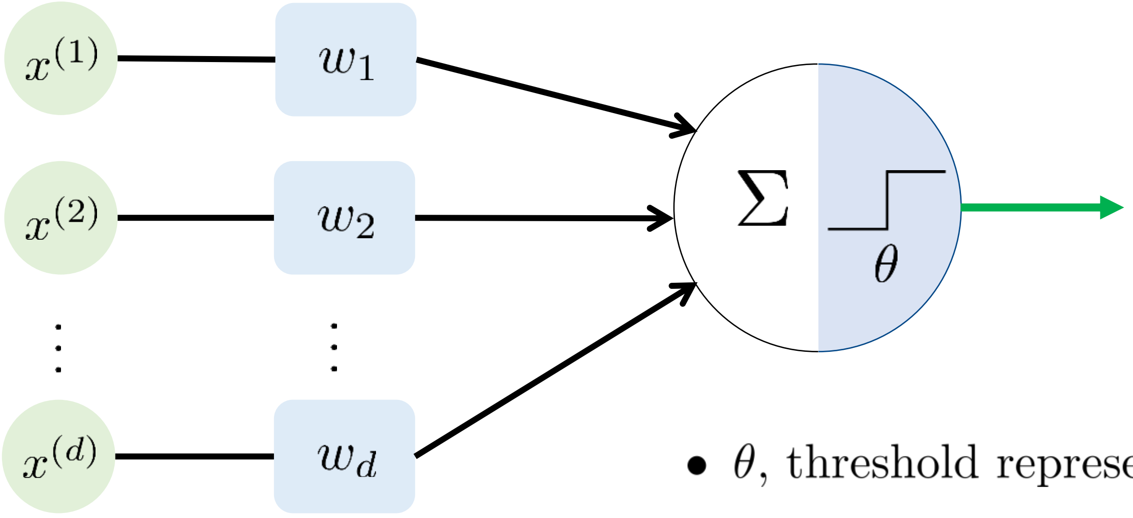
- Can classify if inputs are linearly separable with respect to the output.
  - How to handle the functions/mappings that are not linearly separable e.g., XOR?
- Can handle only boolean inputs.
  - Gives equal or no weightage to the inputs
  - How can we assign different weights to different inputs?
- We hand-code threshold parameter
  - Can we automate the learning process of the parameter?

- To overcome these limitations, another model, known as perception model or perceptron, was proposed by Frank Rosenblatt (1958) and analysed by Minsky and Papert (1969).
  - Inputs real valued, weights used in aggregation
  - Learning of weights and threshold is possible.

Perceptron Classifier

Perceptron:

- d number of real-valued inputs $x^{(1)}, x^{(2)}, \ldots, x^{(d)} \in \mathbf{R}$. *(Difference from MP Neuron)*
- Boolean output, $y \in \{0, 1\}$.
- If sum of inputs is less than θ, the output is zero and one otherwise.
- Threshold θ and weights $w_1, w_2, \ldots, w_d$ are model parameters. *(Difference from MP Neuron)*



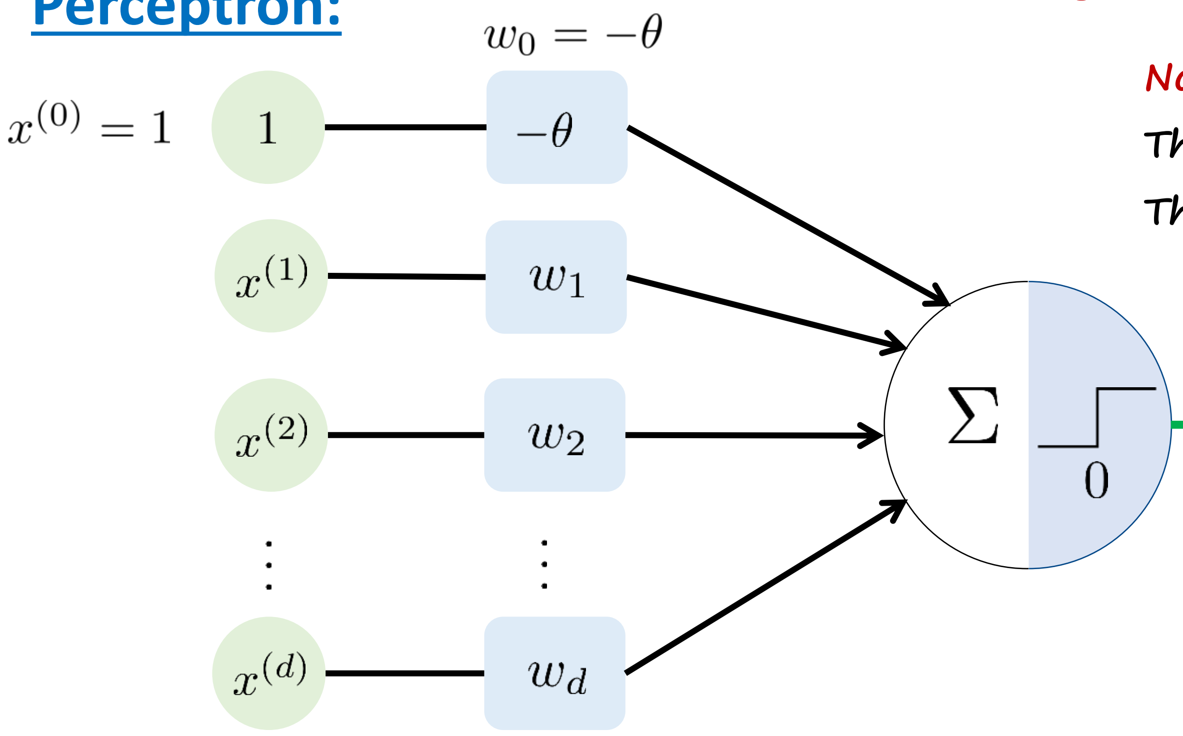
$$y = \begin{cases} 1 & \text{if} \quad \sum_{i=1}^{d} w_i\, x^{(i)} \geq \theta \\ 0 & \text{if} \quad \sum_{i=1}^{d} w_i\, x^{(i)} < \theta \end{cases} = \begin{cases} 1 & \text{if} \quad \sum_{i=1}^{d} w_i\, x^{(i)} - \theta \geq 0 \\ 0 & \text{if} \quad \sum_{i=1}^{d} w_i\, x^{(i)} - \theta < 0 \end{cases}$$

- θ, threshold represents a bias here.
- θ can be considered or absorbed as a weight.
- This will make aggregation/thresholding independent of any parameters.

Source: Perceptron: The Artificial Neuron (An Essential Upgrade To The McCulloch-Pitts Neuron) | by Akshay L Chandra | Towards Data Science

Perceptron Classifier

Perceptron:



Note:

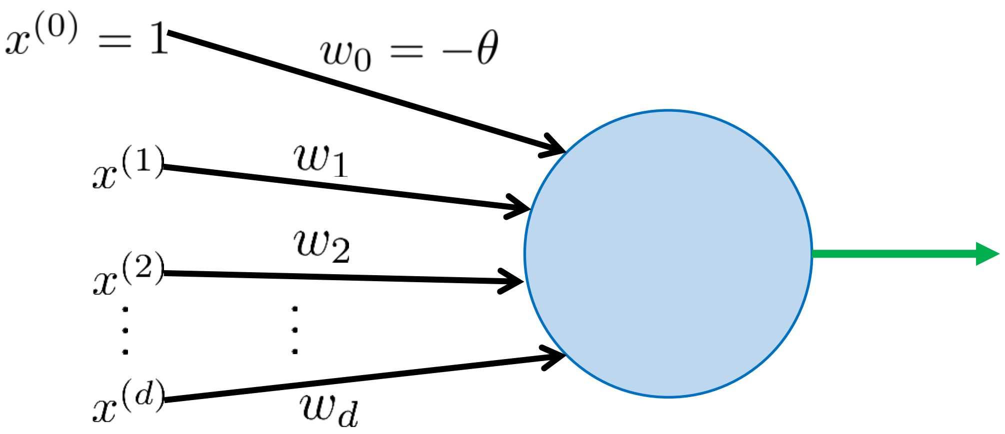
Threshold is zero here.

Threshold has been absorbed as a model weight.

$$y = \begin{cases} 1 & \text{if} \quad \sum_{i=1}^{d} w_i\, x^{(i)} - \theta \geq 0 \text{ or } \sum_{i=0}^{d} w_i\, x^{(i)} \geq 0 \\ 0 & \text{if} \quad \sum_{i=1}^{d} w_i\, x^{(i)} - \theta < 0 \text{ or } \sum_{i=0}^{d} w_i\, x^{(i)} < 0 \end{cases}$$

Alternative (Compact) Representation:

- $\mathbf{x} = [x^{(0)}, x^{(1)}, \ldots, x^{(d)}]$
- $\mathbf{w} = [w_0, w_1, \ldots, w_d]$



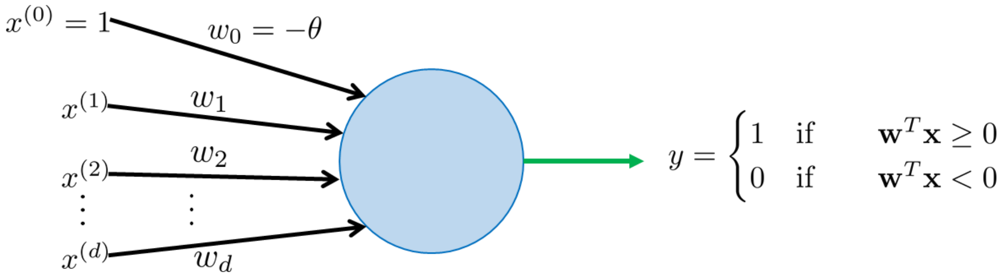
$$y = \begin{cases} 1 & \text{if} \quad \mathbf{w}^T\mathbf{x} \geq 0 \\ 0 & \text{if} \quad \mathbf{w}^T\mathbf{x} < 0 \end{cases}$$

Perceptron Classifier

Classification using Perceptron:

- Since $\mathbf{w}^T\mathbf{x} = 0$ represents a hyper-plane in the d-dimensional space, we can use perceptron as a binary classifier if the classes are linearly separable.

- How is this different from MP neuron?
  - Inputs are real-valued.
  - We have real-valued weights in the process of aggregation.
  - We can learn the weights.



- We have seen this before. We obtained exactly same output in logistic regression case before mapping using sigmoid. We refer to logistic regression classifier as an elementary neural network.

- We used logistic function to have differentiable loss function and squishing the output in $(-\infty, \infty)$ to (0,1) giving us probabilistic view of the classifier.

- How can we learn the weights for the case of perceptron? We note here that we cannot use gradient descent.

- Remark:

  If classes are labeled as 1 and -1

$$y = \begin{cases} 1 & \text{if} \quad \mathbf{w}^T\mathbf{x} \geq 0 \\ -1 & \text{if} \quad \mathbf{w}^T\mathbf{x} < 0 \end{cases}$$

  We often write output as

$$y = \text{sign}(\mathbf{w}^T\mathbf{x})$$

  sign(.) returns sign of the argument.

Outline

- Perceptron and Perceptron Classifier
- Perceptron Learning Algorithm
  - Geometric Intuition
- Perceptron Learning Algorithm Convergence

Perceptron Classifier

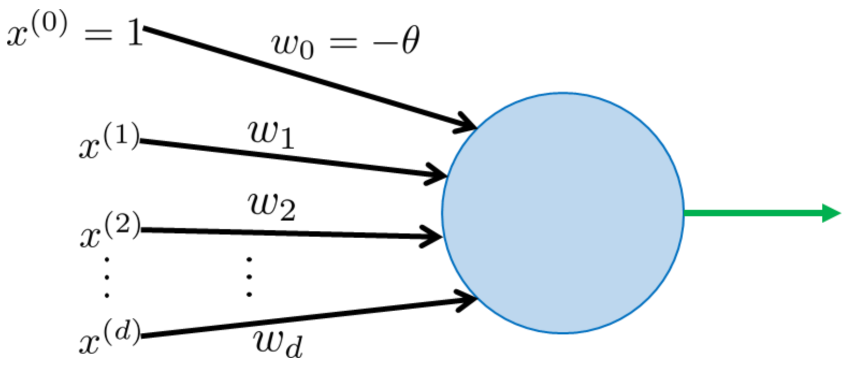
Perceptron Learning Algorithm:

- Assuming that the classes are **linearly separable**, we want to learn $\mathbf{w}$ given the data.

$$D = \{(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), \ldots, (\mathbf{x_n}, y_n)\} \subseteq \mathcal{X}^d \times \mathcal{Y}$$

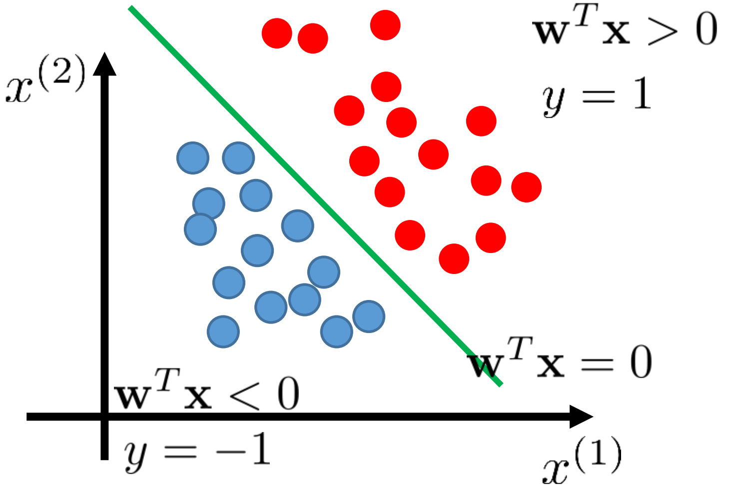
- $\mathcal{Y} = \{0, 1\}$ *(without loss of generality)*
- $\mathcal{Y} = \{-1, 1\}$

- Classifier:



$$y = \begin{cases} 1 & \text{if} \quad \mathbf{w}^T\mathbf{x} \geq 0 \\ -1 & \text{if} \quad \mathbf{w}^T\mathbf{x} < 0 \end{cases}$$

- Data: Linearly separable.



- Key idea: Learn/find a hyperplane characterized by $\mathbf{w}$ such that
  - $y_i(\mathbf{w}^T\mathbf{x}_i) > 0$ for every $(\mathbf{x_i}, y_i) \in D$
  - $y_i(\mathbf{w}^T\mathbf{x}_i) > 0$ implies $\mathbf{x}_i$ is on the correct side of hyperplane.

Perceptron Classifier

Perceptron Learning Algorithm:

Initialize $\mathbf{w} = 0$

while TRUE do

  $m = 0$ *(Count the number of misclassifications)*

  for $(\mathbf{x}_i, y_i) \in \mathcal{D}$ **do**

    if $y_i(\mathbf{w}^T\mathbf{x}_i) \leq 0$ *(misclassification for the chosen point)*

      $\mathbf{w} \leftarrow \mathbf{w} + y_i\mathbf{x}_i$ *(update weight vector: add a point if true label is 1 and subtract a point otherwise)*

      $m \leftarrow m + 1$

    end if

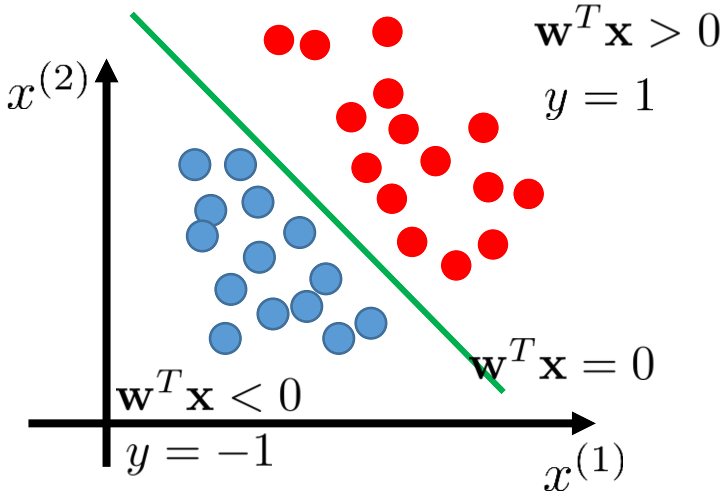
  end for

  if $m = 0$

    break

  end if

end while



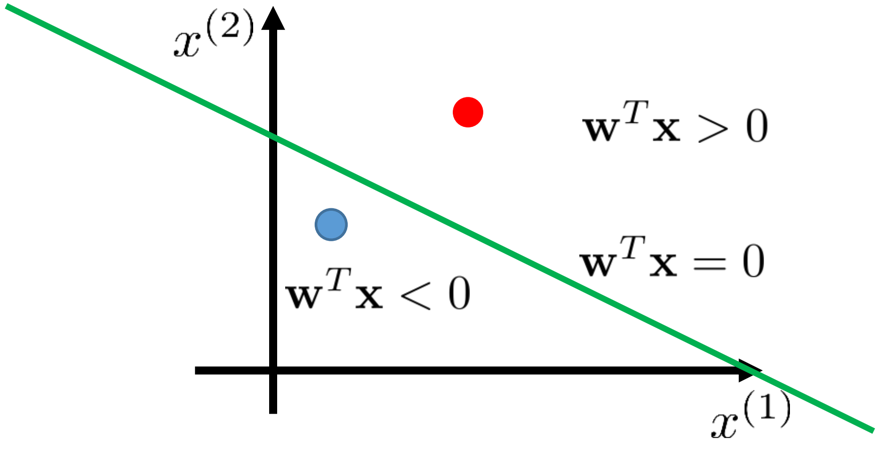
Perceptron Classifier

Perceptron Learning Algorithm – Intuition and Interpretation:

- Visualization of $\mathbf{w}^T\mathbf{x} = 0$:
  - $\mathbf{x} = [x^{(0)}, x^{(1)}, \dots, x^{(d)}]$ $\quad x^{(0)} = 1$
  - $\mathbf{w} = [w_0, w_1, \dots, w_d]$

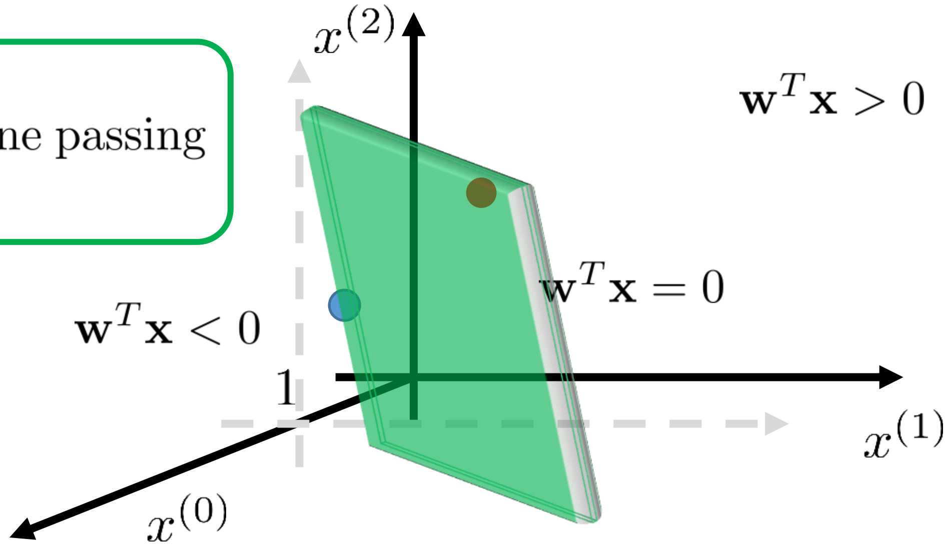
$$\sum_{i=1}^{d} w_i\, x^{(i)} = -w_0$$

(Hyperplane in d-dimensional space)



- Hyper-plane $\mathbf{w}^T\mathbf{x} = 0$ divides the space into two half-spaces.
  - Positive Half-space $\mathbf{w}^T\mathbf{x} > 0$
  - Negative Half-space $\mathbf{w}^T\mathbf{x} < 0$

- One more interpretation:
  Considering $x^{(0)}$ as a dimension, $\mathbf{w}^T\mathbf{x} = 0$ represents a hyperplane passing through origin in $d + 1$ dimensional space.



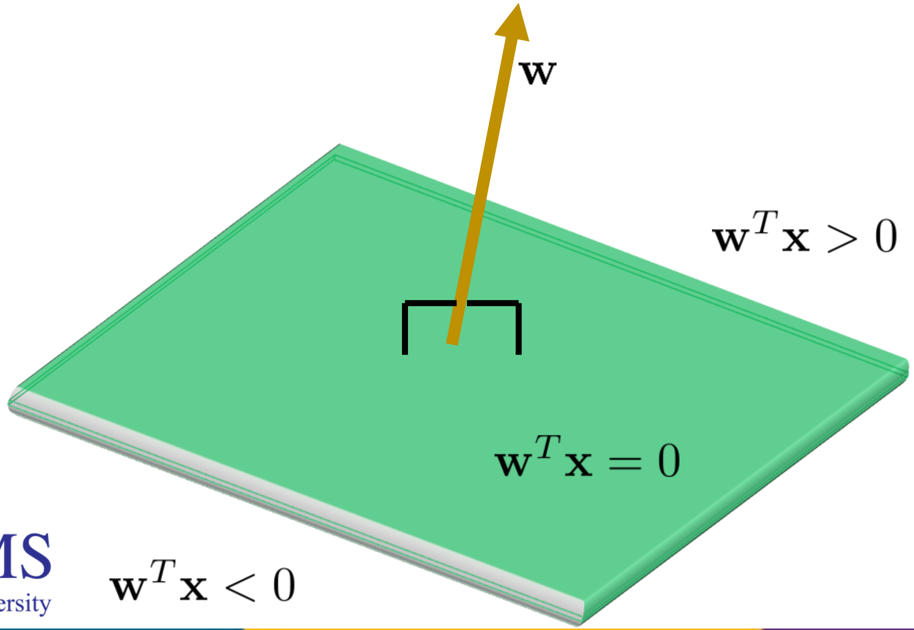
Perceptron Classifier

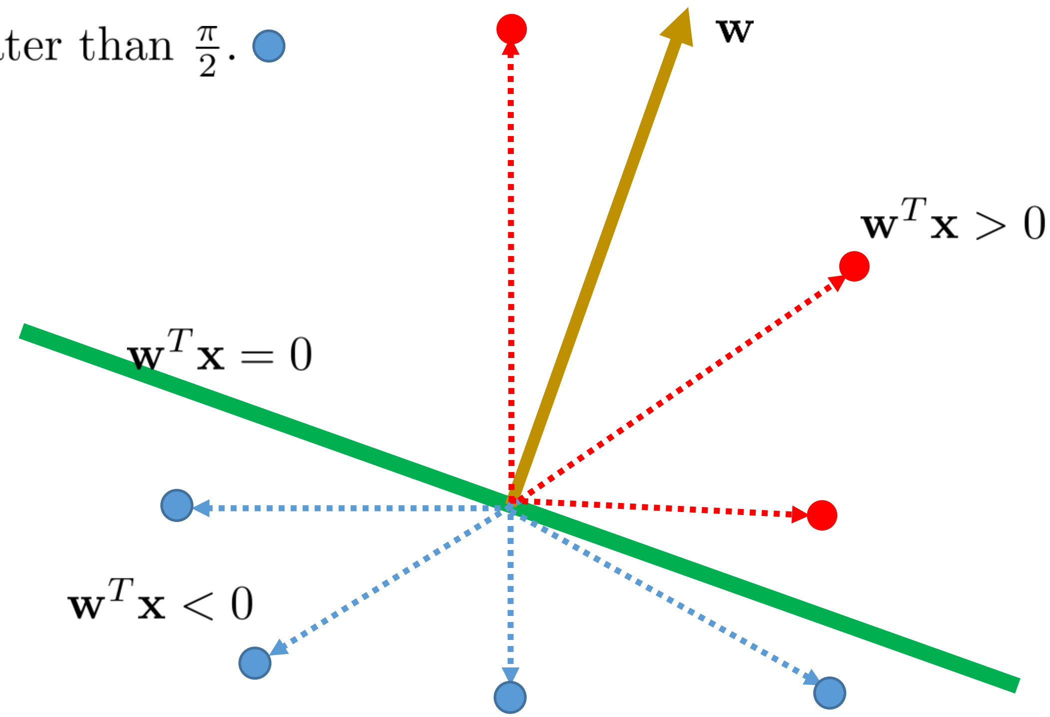
Perceptron Learning Algorithm – Intuition and Interpretation:

- For any point $\mathbf{x}_i$, $|\mathbf{w}^T\mathbf{x}_i|$ represents the distance of $\mathbf{x}$ from the hyper-plane.
- Since every point on the hyper-plane satisfies $\mathbf{w}^T\mathbf{x} = 0$, what is the angle, α between $\mathbf{w}$ and any $\mathbf{x}$?

$$\cos\alpha = \frac{\mathbf{w}^T\mathbf{x}}{\|\mathbf{w}\|_2\|\mathbf{x}\|_2} \Rightarrow \alpha = \frac{\pi}{2}.$$

- For any point in the psoitive half-space $\mathbf{w}^T\mathbf{x} > 0$: angle is less than $\frac{\pi}{2}$.
- For any point in the negative half-space $\mathbf{w}^T\mathbf{x} < 0$: angle is greater than $\frac{\pi}{2}$.





Perceptron Classifier

Perceptron Learning Algorithm – Intuition and Interpretation:

- We want to learn $\mathbf{w}$ such that $y_i(\mathbf{w}^T\mathbf{x}_i) > 0$ for each $(\mathbf{x}_i, y_i) \in \mathcal{D}$.
- In other words, we require $\mathbf{w}^T\mathbf{x}_i > 0$ for $y_i = 1$ and $\mathbf{w}^T\mathbf{x}_i < 0$ for $y_i = -1$

Algorithm:

```
Initialize w = 0
while TRUE do
    m = 0
    for (x_i, y_i) ∈ D do
        if y_i(w^T x_i) ≤ 0
            w ← w + y_i x_i
            m ← m + 1
        end if
    end for
    if m = 0
        break
    end if
end while
```

- We make update when $y_i(\mathbf{w}^T\mathbf{x}_i) \leq 0$.
- For example, consider a point $(\mathbf{x}_i, 1)$ for which

  we have $y_i(\mathbf{w}^T\mathbf{x}_i) \leq 0 \Rightarrow \mathbf{w}^T\mathbf{x}_i \leq 0$.
- Angle α between $\mathbf{w}$ and $\mathbf{x}_i$ is greater than $\pi/2$.
- But we require this angle to be less than $\pi/2$.
- Update: $\mathbf{w}_{\text{new}} = \mathbf{w} + \mathbf{x}_i$
- What about angle (α_{new}) bewteen $\mathbf{w}_{\text{new}}$ and $\mathbf{x}_i$?
- Since $\mathbf{w}_{\text{new}}^T\mathbf{x}_i = \mathbf{w}^T\mathbf{x}_i + \mathbf{x}_i^T\mathbf{x}_i \Rightarrow \mathbf{w}_{\text{new}}^T\mathbf{x}_i > \mathbf{w}^T\mathbf{x}_i$
- Since $\cos(\alpha_{\text{new}}) \propto \mathbf{w}_{\text{new}}^T\mathbf{x}_i$ and $\cos(\alpha) \propto \mathbf{w}^T\mathbf{x}_i$

$$\cos(\alpha_{\text{new}}) > \cos(\alpha)$$

- Consider a point $(\mathbf{x}_i, -1)$

  $y_i(\mathbf{w}^T\mathbf{x}_i) \leq 0 \Rightarrow \mathbf{w}^T\mathbf{x}_i \geq 0$.
- α less than $\pi/2$.
- Require α greater than $\pi/2$.
- Update: $\mathbf{w}_{\text{new}} = \mathbf{w} - \mathbf{x}_i$
- $\mathbf{w}_{\text{new}}^T\mathbf{x}_i < \mathbf{w}^T\mathbf{x}_i$

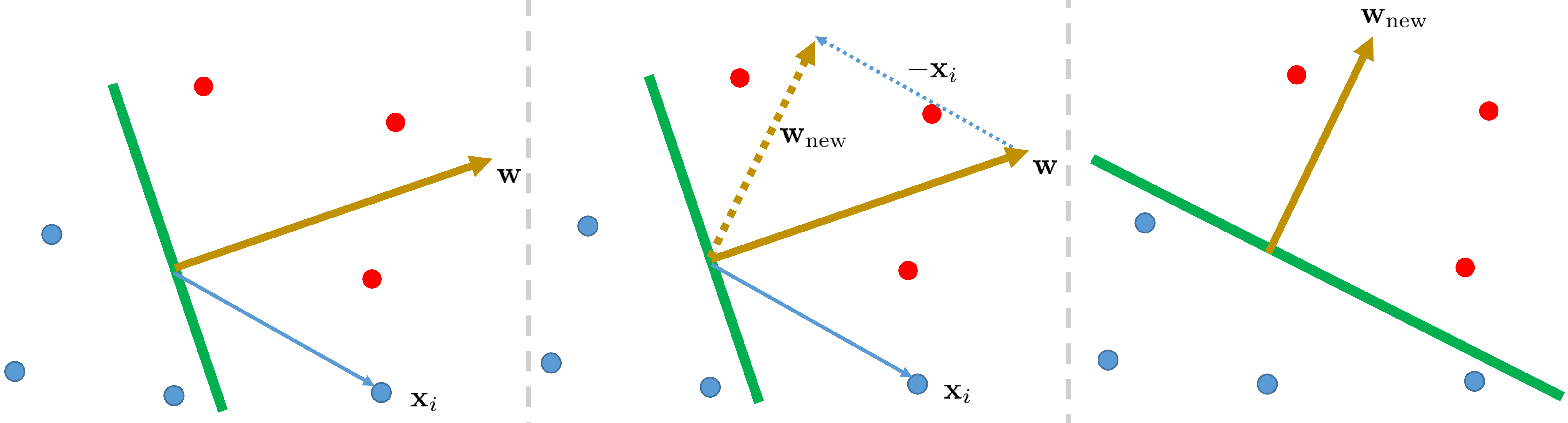
$$\cos(\alpha_{\text{new}}) < \cos(\alpha)$$

This is exactly we require!

Perceptron Classifier

Perceptron Learning Algorithm – Intuition and Interpretation:

Randomly chosen $\mathbf{w}$. | Update: $\mathbf{w}_{\text{new}} = \mathbf{w} - \mathbf{x}_i$



Misclassification for one point only | *Update Step* | *No misclassification*

Perceptron Classifier

Perceptron Learning Algorithm:

Initialize $\mathbf{w} = 0$

while TRUE do

$m = 0$ *(Count the number of misclassifications)*

for $(\mathbf{x}_i, y_i) \in \mathcal{D}$ **do**

if $y_i(\mathbf{w}^T\mathbf{x}_i) \leq 0$ *(misclassification for the chosen point)*

$\mathbf{w} \leftarrow \mathbf{w} + y_i\mathbf{x}_i$ *(update weight vector: add a point if true label is 1 and subtract a point otherwise)*

$m \leftarrow m + 1$

end if

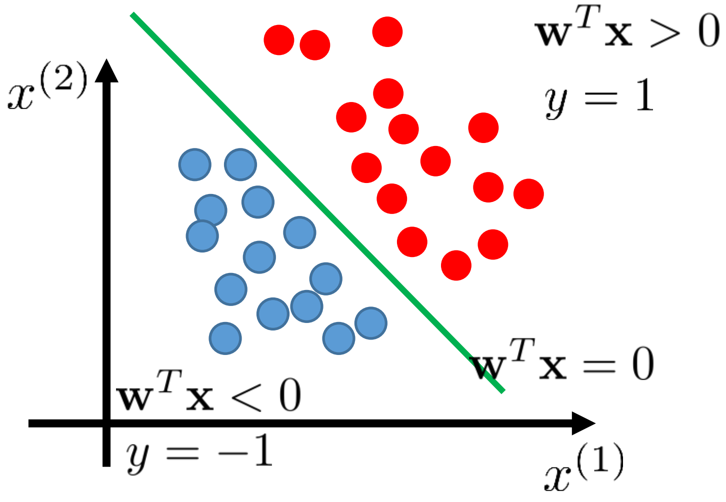
end for

if $m = 0$

break

end if

end while



Outline

- Perceptron and Perceptron Classifier
- Perceptron Learning Algorithm
  - Geometric Intuition
- Perceptron Learning Algorithm Convergence

Perceptron Classifier

Perceptron Learning Algorithm – Proof of Convergence:

Assumptions:

- Data is lineary separable: $\exists\, \mathbf{w}^*$ such that $y_i(\mathbf{x}_i^T \mathbf{w}^*) > 0\ \forall(\mathbf{x}_i, y_i) \in D$.
- We rescale each data point and the $\mathbf{w}^*$ such that

$$||\mathbf{w}^*|| = 1 \qquad \text{and} \qquad ||\mathbf{x}_i|| \leq 1 \quad i = 1, 2, \ldots, n$$

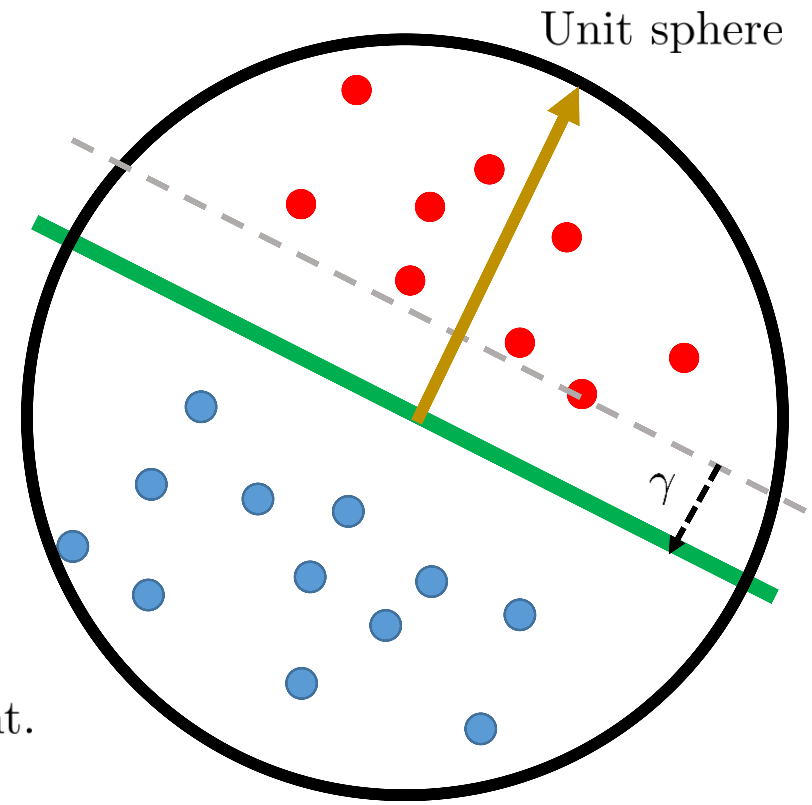
  - All inputs $\mathbf{x_i}$ live within the unit sphere
  - $\mathbf{w}^*$ lies on the unit sphere

- We define the margin of a hyper-plane, denoted by γ, as

$$\gamma = \min_{(\mathbf{x}_i, y_i) \in D} |\mathbf{x}_i^\top \mathbf{w}^*|$$

.

  - γ is the distance from the hyperplane to the closest data point.



Adapted from source: https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote03.html

Perceptron Classifier

Perceptron Learning Algorithm – Proof of Convergence:

Theorem: Under these assumptions, the perceptron algorithm makes at most $1/\gamma^2$ misclassifications.

Proof:

- In our algorithm, when $y_i(\mathbf{w}^T\mathbf{x}_i) \leq 0$, we update as: $\mathbf{w}_{\text{new}} = \mathbf{w} + y_i\mathbf{x}_i$
- Consider the effect of an update on $\mathbf{w}_{\text{new}}^T\mathbf{w}^*$:

$$\mathbf{w}_{\text{new}}^T\mathbf{w}^* = (\mathbf{w} + y_i\mathbf{x})_i^T\mathbf{w}^* = \mathbf{w}^T\mathbf{w}^* + y_i(\mathbf{x}_i^T\mathbf{w}^*) \geq \mathbf{w}^T\mathbf{w}^* + \gamma \qquad (1)$$

The inequality follows from the fact: $\mathbf{w}^*$, the distance from the hyperplane defined by $\mathbf{w}^*$ to $\mathbf{x}_i$ must be at least γ (i.e., $y_i(\mathbf{x}_i^T\mathbf{w}^*) = |\mathbf{x}_i^T\mathbf{w}^*| \geq \gamma$).

This means that for each update, $\mathbf{w}^T\mathbf{w}^*$ grows by at least γ.

- Consider the effect of an update on $\mathbf{w}_{\text{new}}^T\mathbf{w}_{\text{new}}$:

$$\mathbf{w}_{\text{new}}^T\mathbf{w}_{\text{new}} = (\mathbf{w}+y_i\mathbf{x}_i)^T(\mathbf{w}+y_i\mathbf{x}_i) = \mathbf{w}^T\mathbf{w}+2y_i(\mathbf{w}^T\mathbf{x}_i)+y_i^2(\mathbf{x}_i^T\mathbf{x}) \leq \mathbf{w}^T\mathbf{w}+1 \qquad (2)$$

The inequality follows from the fact: $2y_i(\mathbf{w}^T\mathbf{x}_i) \leq 0$ as we had to make an update. $y_i^2(\mathbf{x}_i^T\mathbf{x}_i) \leq 1$ as $y_i^2 = 1$ and all $\mathbf{x}_i^T\mathbf{x}_i \leq 1$ (because $\|\mathbf{x}_i\| \leq 1$).

This means that for each update, $\mathbf{w}^T\mathbf{w}$ grows by at most 1.

LUMS
A Not-for-Profit University

Perceptron Classifier

Perceptron Learning Algorithm – Proof of Convergence:

Proof (continued):

$$\mathbf{w}_{\text{new}}^T \mathbf{w}^* \geq \mathbf{w}^T \mathbf{w}^* + \gamma \quad (1)$$

$\mathbf{w}^T\mathbf{w}^*$ grows by at least γ.

$$\mathbf{w}_{\text{new}}^T \mathbf{w}_{\text{new}} \leq \mathbf{w}^T \mathbf{w} + 1 \quad (2)$$

$\mathbf{w}^T\mathbf{w}$ grows by at most 1.

After M updates, we have:

- $M\gamma \leq \mathbf{w}^T\mathbf{w}^*$ *(From (1); each update increases, at least, by gamma)*
- $M\gamma \leq \mathbf{w}^T\mathbf{w}^* = |\mathbf{w}^T\mathbf{w}^*| \leq \|\mathbf{w}\|\,\|\mathbf{w}^*\|$ *(By Cauchy-Schwartz inequality)*
- $M\gamma \leq \|\mathbf{w}\|\,\|\mathbf{w}^*\| = \|\mathbf{w}\|$ *(Unit sphere assumption)*
- $M\gamma \leq \|\mathbf{w}\| = \sqrt{\mathbf{w}^T\mathbf{w}}$
- $M\gamma \leq \|\mathbf{w}\| = \sqrt{\mathbf{w}^T\mathbf{w}} \leq \sqrt{M}$
- $M\gamma \leq \sqrt{M} \Rightarrow M^2\gamma^2 \leq M \Rightarrow M \leq \frac{1}{\gamma^2}.$

Theorem is proved since the number of updates is equal to the number of misclassifications!

Perceptron Classifier

Summary:

- As can train perceptron to classify given data but cannot be used to estimate the probability of x or generate x given y, Perceptron classifier is discriminative.

- Assumes that the classes are linearly separable.
  - Does not make any assumptions about the data such as feature independence (required for Naïve Bayes).

- We can update the weights (model parameters) using one training data point, and therefore the perceptron classifier is an online learning algorithm.

- Learning Algorithm is based on the principle that it uses mistakes during learning to iteratively update the weights.

- Under certain assumptions, we showed the convergence of the learning algorithm.