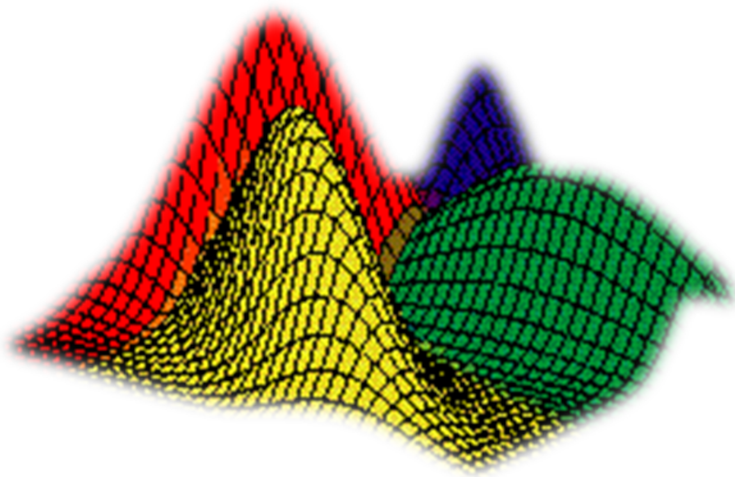


# CSE 473: Pattern Recognition

# Linear Classifier: Introduction

- Classifies linearly separable patterns
- we know the proper forms for the discriminant functions
- may not be optimal,
- very simple to use

# Linear discriminant functions and decisions surfaces

- Definition

Let a pattern vector $\mathbf{x} = \{x_1, x_2, x_3, \ldots,\}$
a weight vector $\mathbf{w} = \{w_1, w_2, w_3, \ldots,\}$

A discriminant function :

$$g(\mathbf{x}) = x_1 w_1 + x_2 w_2 + x_3 w_3 + \ldots$$

OR

$$g(\mathbf{x}) = w^t x + w_0 \qquad (1)$$

where w is the weight vector and $w_0$ the bias

# Linear discriminant functions and decisions surfaces

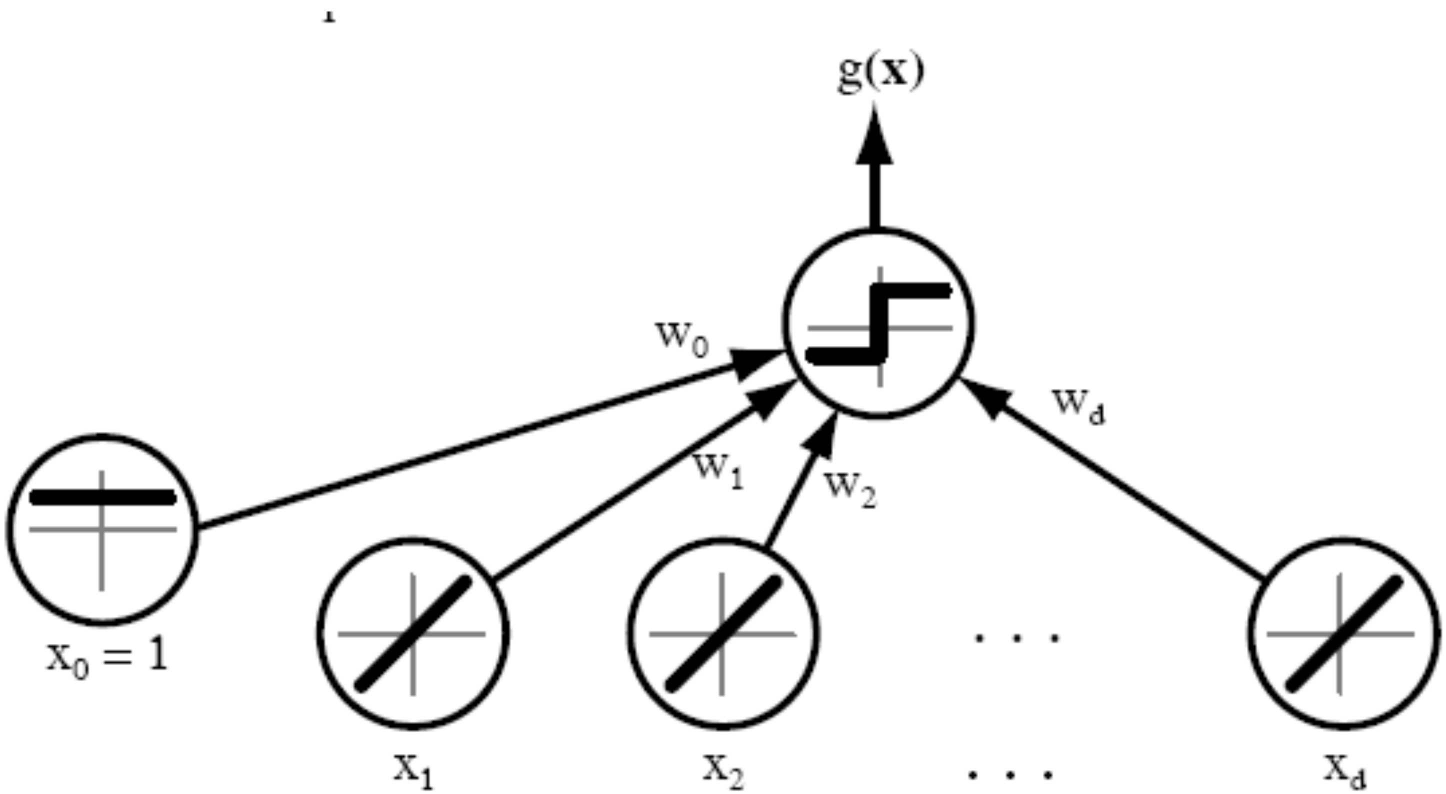
- Classify a new pattern **x** as follows

  Decide class $\omega_1$ *if* $g(\mathrm{x}) > 0$

  and class $\omega_2$ if $g(\mathrm{x}) < 0$

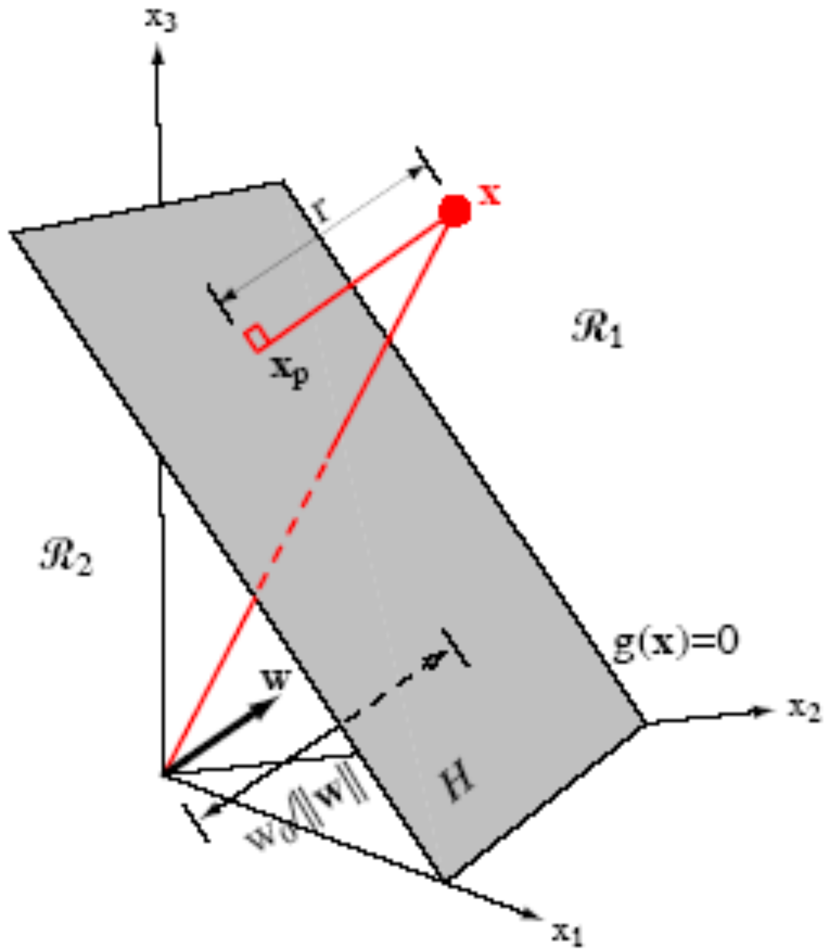
  If $g(x) = 0 \Rightarrow x$ is assigned to either class

# Linear discriminant functions and decisions surfaces



- The equation $g(x) = 0$ is the decision surface that separates patterns

- When $g(x)$ is linear, the decision surface is a hyperplane


x3
r
x
R1
xp
R2
g(x)=0
w
x2
H
w0/||w||
x1

# A little bit mathematics

- The Problem: Consider a two class task with $\omega_1$, $\omega_2$
  - $g(\underline{x}) = \underline{w}^T \underline{x} + w_0 = 0 =$
    $w_1 x_1 + w_2 x_2 + \ldots + w_l x_l + w_0$
  - Assume $\underline{x}_1, \underline{x}_2$ on the decision hyperplane:
    $0 = \underline{w}^T \underline{x}_1 + w_0 = \underline{w}^T \underline{x}_2 + w_0 \Rightarrow$
    $\underline{w}^T (\underline{x}_1 - \underline{x}_2) = 0 \;\; \forall \underline{x}_1, \underline{x}_2$

- Hence:

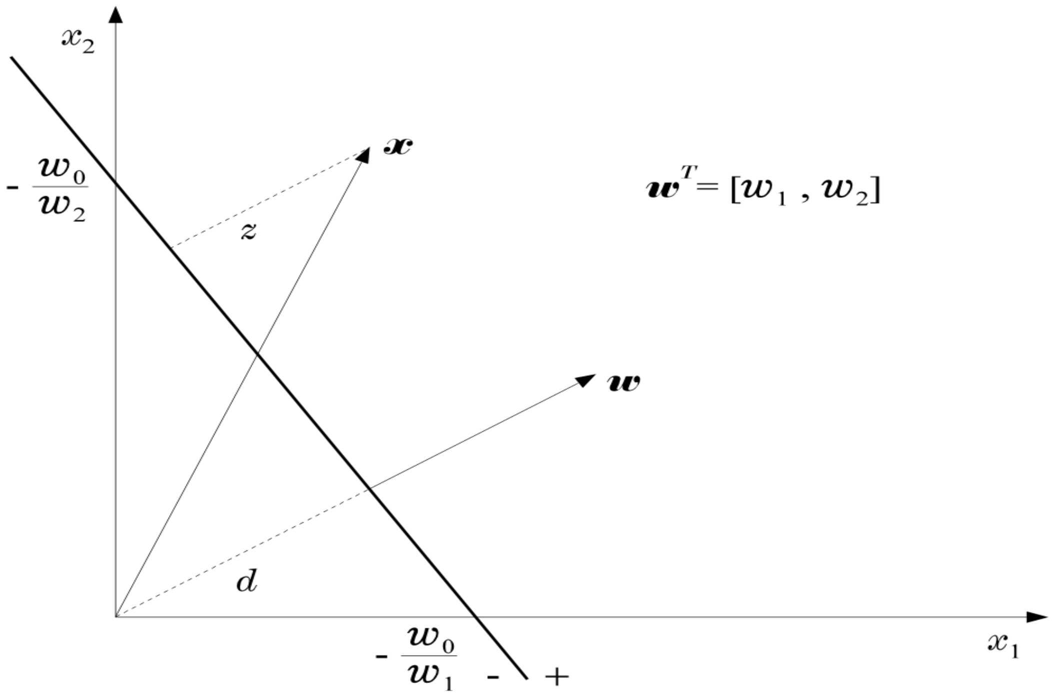
$$\boxed{\underline{w} \perp \text{ on the hyperplane}}$$

$$g(\underline{x}) = \underline{w}^T \underline{x} + w_0 = 0$$

## ➢ Hence:

$$\boxed{\underline{w} \perp \text{ on the hyperplane}}$$

$$g(\underline{x}) = \underline{w}^T \underline{x} + w_0 = 0$$




$$d = \frac{|w_0|}{\sqrt{w_1^2 + w_2^2}}, \quad z = \frac{|g(\underline{x})|}{\sqrt{w_1^2 + w_2^2}}$$

- The Perceptron Algorithm

  - Assume linearly separable classes, i.e.,

$$\exists\ \underline{w}^*:\ w^{*T}\ \underline{x} > 0\ \ \forall \underline{x} \in \omega_1$$
$$\underline{w}^{*T}\ \underline{x} < 0\ \ \forall \underline{x} \in \omega_2$$

- The Perceptron Algorithm

  - Assume linearly separable classes, i.e.,

$$\exists\ \underline{w}^*:\ \underline{w}^{*T}\underline{x} > 0\ \ \forall \underline{x} \in \omega_1$$
$$\underline{w}^{*T}\underline{x} < 0\ \ \forall \underline{x} \in \omega_2$$

  - The case $\underline{w}^{*T}\underline{x} + w_0^*$
    falls under the above formulation, since

    - $\underline{w}' \equiv \begin{bmatrix} \underline{w}^* \\ w_0^* \end{bmatrix},\ \underline{x}' = \begin{bmatrix} \underline{x} \\ 1 \end{bmatrix}$

    - $\underline{w}^{*T}\underline{x} + w_0^* = \underline{w}'^T\underline{x}' = 0$

– Our goal: Compute a solution, i.e., a hyperplane $\underline{w}$, so that

$$\underline{w}^T \underline{x} (><) 0 \quad \underline{x} \in \begin{cases} \omega_1 \\ \omega_2 \end{cases}$$

- The steps
  – Define a cost function to be minimized
  – Choose an algorithm to minimize the cost function
  – The minimum corresponds to a solution

– The Cost Function

$$J(\underline{w}) = \sum_{\underline{x} \in Y} (\delta_x \underline{w}^T \underline{x})$$

- Where $Y$ is the subset of the vectors wrongly classified by $\underline{w}$.

- $$\delta_x = -1 \text{ if } \underline{x} \in Y \text{ and } \underline{x} \in \omega_1$$
  $$\delta_x = +1 \text{ if } \underline{x} \in Y \text{ and } \underline{x} \in \omega_2$$

– The Cost Function

$$J(\underline{w}) = \sum_{\underline{x} \in Y} (\delta_x \underline{w}^T \underline{x})$$

- Where $Y$ is the subset of the vectors wrongly classified by $\underline{w}$.
- when $Y$=(empty set) a solution is achieved and

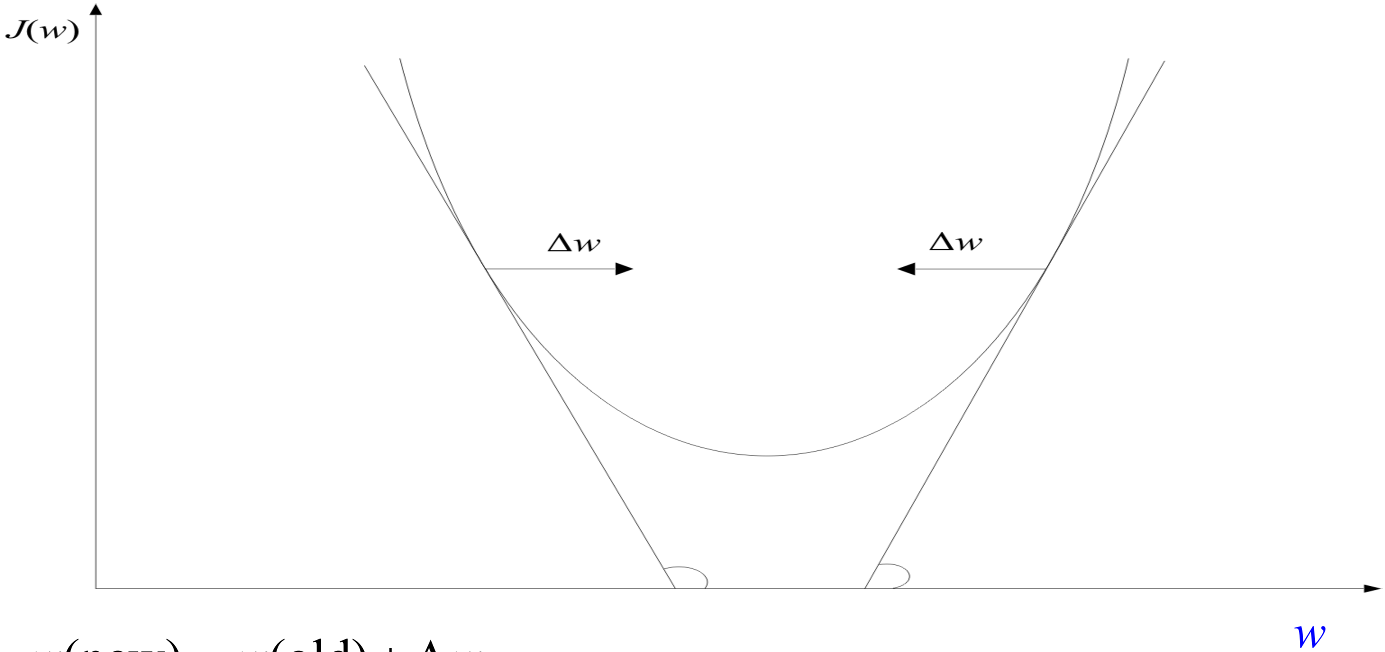
$$J(\underline{w}) = 0$$

otherwise

$$J(\underline{w}) \geq 0$$

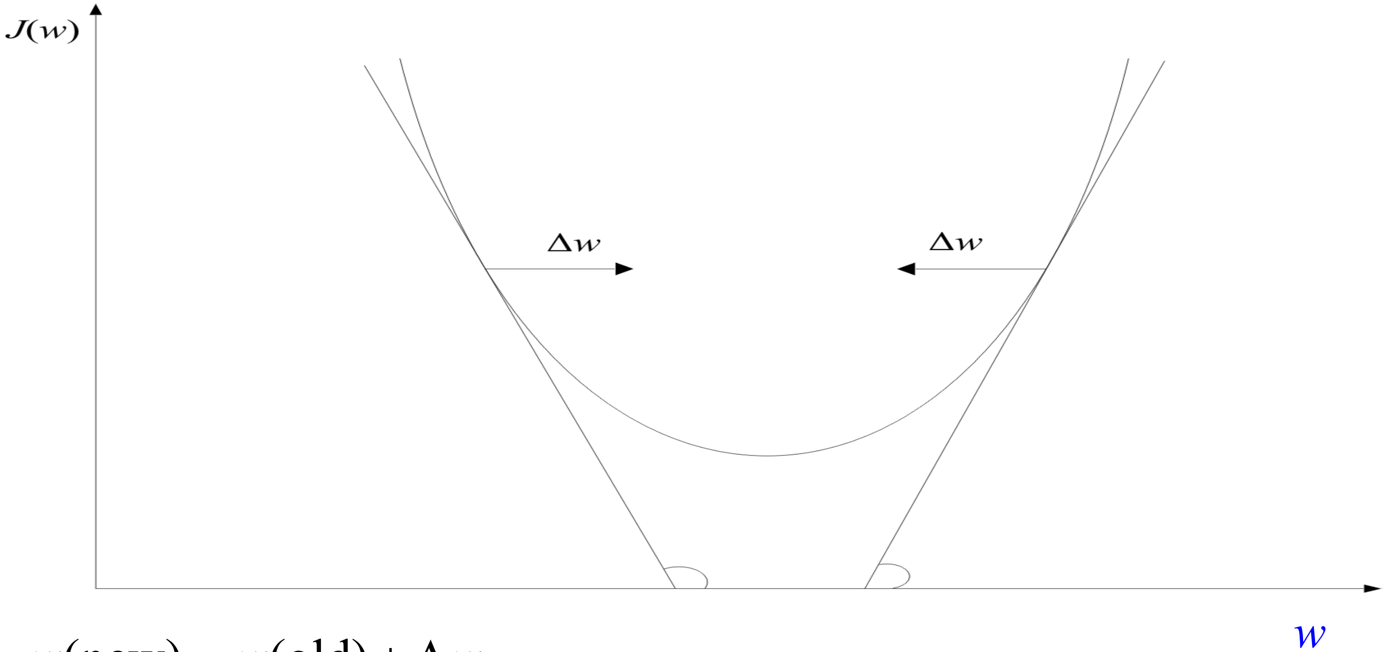
- $J(\underline{w})$ is piecewise linear (WHY?)

– The Algorithm
  - The philosophy of the gradient descent is adopted.




$$\underline{w}(\text{new}) = \underline{w}(\text{old}) + \Delta \underline{w}$$

$$\Delta \underline{w} = -\mu \frac{\partial J(\underline{w})}{\partial \underline{w}} \Big| \underline{w} = \underline{w}(\text{old})$$




$$\underline{w}(\text{new}) = \underline{w}(\text{old}) + \Delta \underline{w}$$

$$\Delta \underline{w} = -\mu \frac{\partial J(\underline{w})}{\partial \underline{w}} \Big| \underline{w} = \underline{w}(\text{old})$$

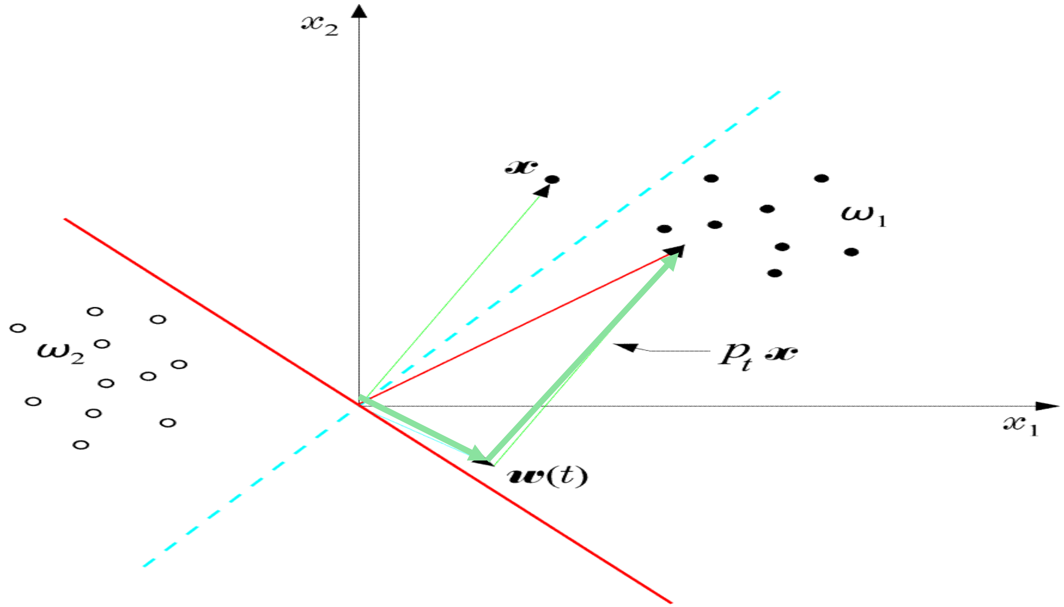
- Wherever valid

$$\frac{\partial J(\underline{w})}{\partial \underline{w}} = \frac{\partial}{\partial \underline{w}} \left( \sum_{\underline{x} \in Y} \delta_x \underline{w}^T \underline{x} \right) = \sum_{\underline{x} \in Y} \delta_x \underline{x}$$

- $$\boxed{\underline{w}(t+1) = \underline{w}(t) - \rho_t \sum_{\underline{x} \in Y} \delta_x \underline{x}}$$

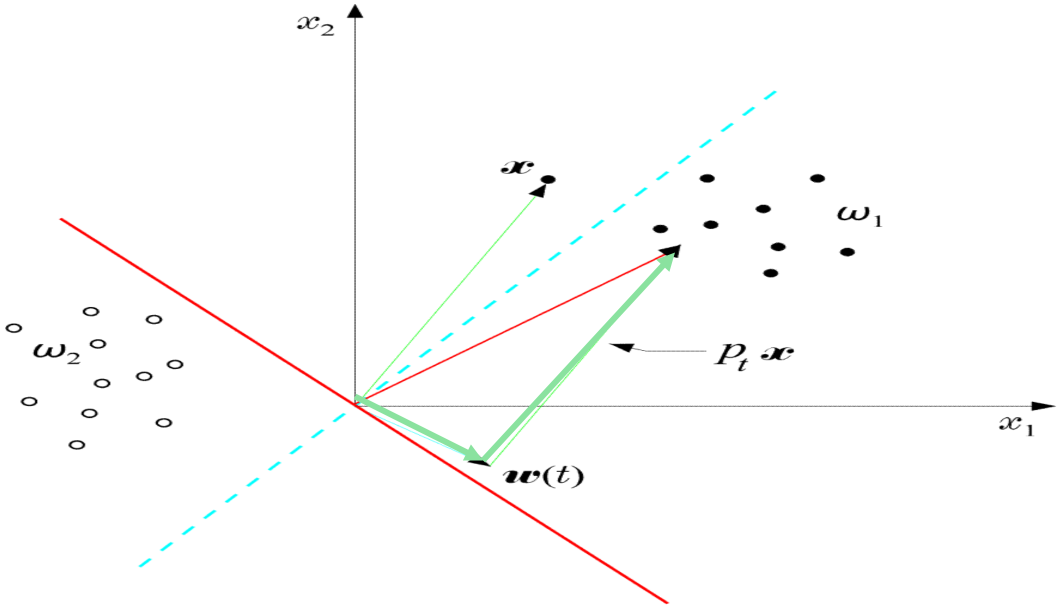
This is the celebrated Perceptron Algorithm

– An example:




$$\underline{w}(t+1) = \underline{w}(t) + \rho_t\, \underline{x}$$
$$= \underline{w}(t) - \rho_t \delta_x \underline{x} \quad (\delta_x = -1)$$

- An example:




$$\underline{w}(t+1) = \underline{w}(t) + \rho_t \underline{x}$$
$$= \underline{w}(t) - \rho_t \delta_x \underline{x} \quad (\delta_x = -1)$$

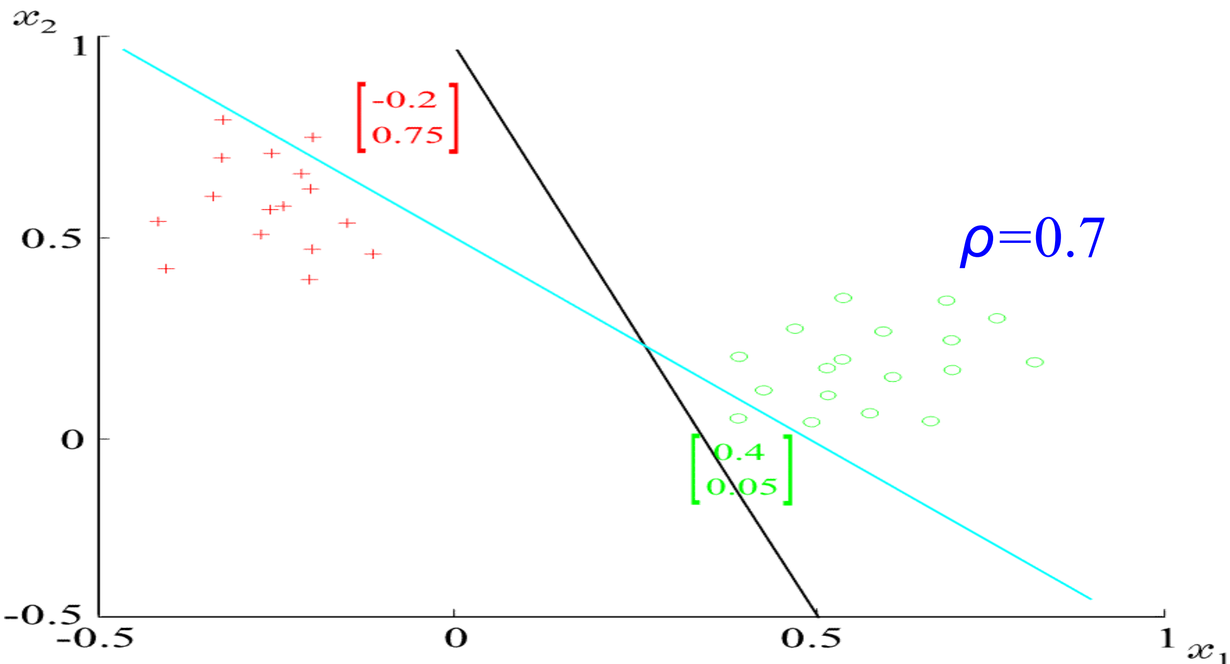
- The perceptron algorithm **converges** in a **finite** number of iteration steps to a solution if

– Example: At some stage $t$ the perceptron algorithm results in

$$w_1 = 1,\ w_2 = 1,\ w_0 = -0.5$$

$$x_1 + x_2 - 0.5 = 0$$

The corresponding hyperplane is




$$\underline{w}(t+1) = \begin{bmatrix}1\\1\\-0.5\end{bmatrix} - 0.7(-1)\begin{bmatrix}0.4\\0.05\\1\end{bmatrix} - 0.7(+1)\begin{bmatrix}-0.2\\0.75\\1\end{bmatrix} = \begin{bmatrix}1.42\\0.51\\-0.5\end{bmatrix}$$

# Convergence Proof of Perceptron Algorithm

Let , $\underline{w}^*$ be the optimal weight vector
$\underline{w}(t)$ be the weight vector at the $t$th iteration
$\underline{w}(t+1)$ be the weight vector at the $(t+1)$th iteration

We will prove that $||\underline{w}(t+1) - \underline{w}^*|| < ||\underline{w}(t) - \underline{w}^*||$

# Convergence Proof of Perceptron Algorithm

We know that $\boxed{\underline{w}(t+1) = \underline{w}(t) - \rho_t \sum_{\underline{x} \in Y} \delta_x \underline{x}}$

Let , α be a positive real number

Then, $$\boldsymbol{w}(t+1) - \alpha \boldsymbol{w}^* = \boldsymbol{w}(t) - \alpha \boldsymbol{w}^* - \rho_t \sum_{\boldsymbol{x} \in Y} \delta_x \boldsymbol{x}$$

# Convergence Proof of Perceptron Algorithm

$$\boldsymbol{w}(t+1) - \alpha \boldsymbol{w}^* = \boldsymbol{w}(t) - \alpha \boldsymbol{w}^* - \rho_t \sum_{x \in Y} \delta_x \boldsymbol{x}$$

Squaring both sides,

$$\|\boldsymbol{w}(t+1) - \alpha \boldsymbol{w}^*\|^2 = \|\boldsymbol{w}(t) - \alpha \boldsymbol{w}^*\|^2 + \rho_t^2 \|\sum_{x \in Y} \delta_x \boldsymbol{x}\|^2 - 2\rho_t \sum_{x \in Y} \delta_x (\boldsymbol{w}(t) - \alpha \boldsymbol{w}^*)^T \boldsymbol{x}$$

# Convergence Proof of Perceptron Algorithm

$$\|\boldsymbol{w}(t+1) - \alpha \boldsymbol{w}^*\|^2 = \|\boldsymbol{w}(t) - \alpha \boldsymbol{w}^*\|^2 + \rho_t^2 \|\sum_{x \in Y} \delta_x \boldsymbol{x}\|^2 - 2\rho_t \sum_{x \in Y} \delta_x (\boldsymbol{w}(t) - \alpha \boldsymbol{w}^*)^T \boldsymbol{x}$$

However, $-\sum_{\boldsymbol{x} \in Y} \delta_x \boldsymbol{w}^T(t)\, \boldsymbol{x} < 0$

Hence,

$$\|\boldsymbol{w}(t+1) - \alpha \boldsymbol{w}^*\|^2 \le \|\boldsymbol{w}(t) - \alpha \boldsymbol{w}^*\|^2 + \rho_t^2 \|\sum_{x \in Y} \delta_x \boldsymbol{x}\|^2 + 2\rho_t \alpha \sum_{x \in Y} \delta_x \boldsymbol{w}^{*T} \boldsymbol{x}$$

# Convergence Proof of Perceptron Algorithm

$$\|\boldsymbol{w}(t+1) - \alpha \boldsymbol{w}^*\|^2 \le \|\boldsymbol{w}(t) - \alpha \boldsymbol{w}^*\|^2 + \rho_t^2 \|\sum_{x \in Y} \delta_x \boldsymbol{x}\|^2 + 2\rho_t \alpha \sum_{x \in Y} \delta_x \boldsymbol{w}^{*T} \boldsymbol{x}$$

Now, define

$$\beta^2 = \max_{\tilde{Y} \subseteq \omega_1 \cup \omega_2} \|\sum_{x \in \tilde{Y}} \delta_x \boldsymbol{x}\|^2$$ and $$\gamma = \max_{\tilde{Y} \subseteq \omega_1 \cup \omega_2} \sum_{x \in \tilde{Y}} \delta_x \boldsymbol{w}^{*T} \boldsymbol{x}$$

# Convergence Proof of Perceptron Algorithm

$$\|\boldsymbol{w}(t+1)-\alpha\boldsymbol{w}^*\|^2 \le \|\boldsymbol{w}(t)-\alpha\boldsymbol{w}^*\|^2+\rho_t^2\|\sum_{x\in Y}\delta_x\boldsymbol{x}\|^2+2\rho_t\alpha\sum_{x\in Y}\delta_x\boldsymbol{w}^{*T}\boldsymbol{x}$$

Now, define

$$\beta^2=\max_{\tilde{Y}\subseteq\omega_1\cup\omega_2}\|\sum_{x\in\tilde{Y}}\delta_x\boldsymbol{x}\|^2 \quad \text{and} \quad \gamma=\max_{\tilde{Y}\subseteq\omega_1\cup\omega_2}\sum_{x\in\tilde{Y}}\delta_x\boldsymbol{w}^{*T}\boldsymbol{x}$$

Here, $\delta_x\boldsymbol{w}^{*T}\boldsymbol{x}$ is always negative, so is $\gamma$

We can write,

$$\|\boldsymbol{w}(t+1)-\alpha\boldsymbol{w}^*\|^2 \le \|\boldsymbol{w}(t)-\alpha\boldsymbol{w}^*\|^2+\rho_t^2\beta^2-2\rho_t\alpha|\gamma|$$

# Convergence Proof of Perceptron Algorithm

$$\|w(t+1) - \alpha w^*\|^2 \le \|w(t) - \alpha w^*\|^2 + \rho_t^2\beta^2 - 2\rho_t\alpha|\gamma|$$

If we choose, $\alpha = \dfrac{\beta^2}{2|\gamma|}$

We can write,

$$\|w(t+1) - \alpha w^*\|^2 \le \|w(t) - \alpha w^*\|^2 + \rho_t^2\beta^2 - \rho_t \beta^2$$

# Convergence Proof of Perceptron Algorithm

$$\|w(t+1) - \alpha w^*\|^2 \leq \|w(t) - \alpha w^*\|^2 + \rho_t^2 \beta^2 - \rho_t \beta^2$$

Here, $\rho_t^2 \beta^2 - \rho_t \beta^2 < 0$

How?

# Convergence Proof of Perceptron Algorithm

$$\|w(t+1) - \alpha w^*\|^2 \leq \|w(t) - \alpha w^*\|^2 + \rho_t^2 \beta^2 - \rho_t \beta^2$$

Applying the above equation successively for steps *t*, *t*-1, . . ., 0, we get

$$\|w(t+1) - \alpha w^*\|^2 \leq \|w(0) - \alpha w^*\|^2 + \beta^2 \left( \sum_{k=0}^{t} \rho_k^2 - \sum_{k=0}^{t} \rho_k \right)$$

However, $\lim_{t \to \infty} \sum_{k=0}^{t} \rho_k = \infty$ and $\lim_{t \to \infty} \sum_{k=0}^{t} \rho_k^2 < \infty$

# Convergence Proof of Perceptron Algorithm

$$\|w(t+1) - \alpha w^*\|^2 \leq \|w(0) - \alpha w^*\|^2 + \beta^2 \left( \sum_{k=0}^{t} \rho_k^2 - \sum_{k=0}^{t} \rho_k \right)$$

However, $\lim_{t \to \infty} \sum_{k=0}^{t} \rho_k = \infty$ and $\lim_{t \to \infty} \sum_{k=0}^{t} \rho_k^2 < \infty$

This means,
After some constant tine $t_0$ the R. H. S. will be non-positive

But, the L. H. S. cannot be negative

Therefore, $0 \leq \|w(t_0 + 1) - \alpha w^*\| \leq 0$

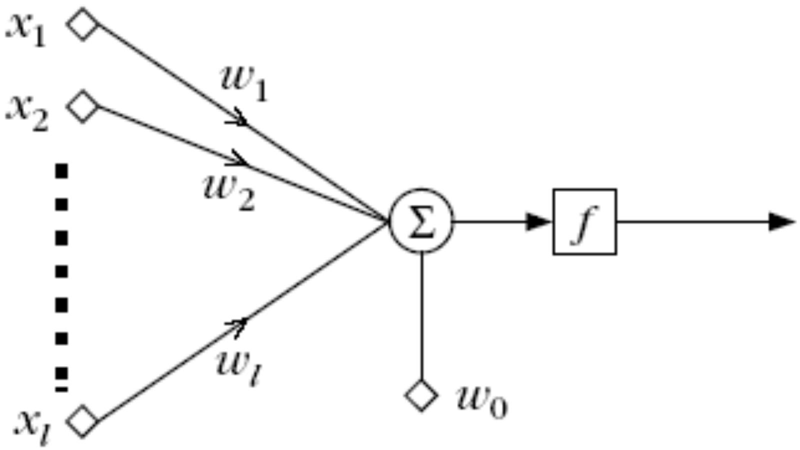
# Convergence Proof of Perceptron Algorithm

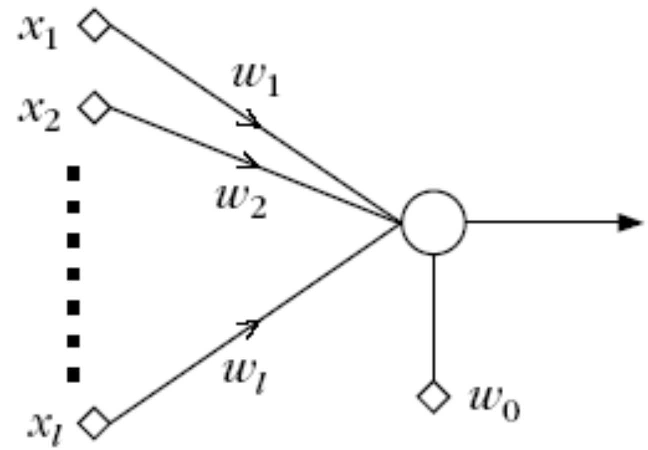
$$0 \leq \| \boldsymbol{w}(t_0 + 1) - \alpha \boldsymbol{w}^* \| \leq 0$$

is equivalent to

$$\boldsymbol{w}(t_0 + 1) = \alpha \boldsymbol{w}^*$$

# The Perceptron




$w_i\,'s$ synapses or synaptic weights

$w_0$ threshold

- The network is called perceptron or neuron
- a learning machine that learns from the training vectors

# Variants of Perceptron Algorithm

$$\underline{w}(t+1) = \underline{w}(t) + \rho \underline{x}_{(t)}, \quad \begin{matrix} \underline{w}^T(t)\underline{x}_{(t)} \leq 0 \\ \underline{x}_{(t)} \in \omega_1 \end{matrix}$$

$$\underline{w}(t+1) = \underline{w}(t) - \rho \underline{x}_{(t)}, \quad \begin{matrix} \underline{w}^T(t)\underline{x}_{(t)} \geq 0 \\ \underline{x}_{(t)} \in \omega_2 \end{matrix}$$

$$\underline{w}(t+1) = \underline{w}(t) \quad \text{otherwise}$$

– It is a reward and punishment type of algorithm

# Variants of Perceptron Algorithm

- initialize weight vector $\mathbf{w}(0)$
- define pocket $\mathbf{w}_s$.and history $h_s$
- generate next $\mathbf{w}(t+1)$. if it is better than $\mathbf{w}(t)$, store $\mathbf{w}(t+1)$ in $\mathbf{w}_s$ and change the $h_s$

  – It is $\boxed{\text{pocket}}$ algorithm

# Generalization of Perceptron Algorithm for M- Class case