Perceptrón simple

Laboratorio de Machine Learning
Julián Ganzabal
Carlos Selmo

Redes Feed-Forward

- Llamaremos a las entradas y salidas de una Red Feed-Forward ξ_k y O_i respectivamente.
- La salida O_i puede ser calculada como:

$$O_i = g(h_i) = g\left(\sum_k w_{ik}\xi_k\right) \qquad (5.1)$$

  - Donde $g(h)$ es la función de activación similar al caso de las redes de Hopfield. Puede ser la función sígno, una sigmoidea continua o una unidad estocástica.
- Su nombre "Feed-Forward" se debe a que la salida de la red se obtiene como una propagación directa de la entrada hacia la salida, sin realimentaciones ni bucles.
- Omitiremos el uso de bias o thresholds ya que son un caso particular en el que cada unidad tiene una entrada fija en -1 y un peso correspondiente w_{i0}.

$$O_i = g\left(\sum_{k=0}^{N} w_{ik}\xi_k\right) = g\left(\sum_{k=1}^{N} w_{ik}\xi_k - \theta_i\right) \qquad (5.2)$$

Redes Feed-Forward

- Llamaremos ζ_i^μ a la salida deseada cuando ponemos un patrón ξ_k^μ. Si la red funciona correctamente queremos que se cumpla:

$$O_i^\mu = \zeta_i^\mu \qquad \text{(desired)} \qquad (5.3)$$

  - Para cada valor de i y μ, donde:

$$O_i^\mu = g(h_i^\mu) = g\left(\sum_k w_{ik}\xi_k^\mu\right). \qquad (5.4)$$

- Para este tipo de redes las entradas y salidas pueden ser binarias (+1 y -1) o continuas.
  - Particularmente para la salida, dependerá del tipo de función de activación que elijamos.
- Si elegimos el número de entradas igual al número de salidas y que la salida deseada sea igual a la entrada, podemos reducir este tipo de redes a una memoria autoasociativa.

Redes Feed-Forward

- En general, este tipo de redes pueden implementar memorias heteroasociativas (en las cuales los patrones de entrada ξ_k^μ son distintos a los patrones de salida ζ_i^μ).
- $p = 1,2,3 \ldots, \mu$ sigue siendo la cantidad de patrones del training set.
- Un caso particular de la memoria heteroasociativa es el problema de **clasificación** en el cual a un patrón de entrada ξ_k^μ le corresponde una clase determinada ζ_i^μ.
- Demostraremos para perceptrones simples que si existe un conjunto de pesos w_{ik} que solucionan el problema planteado, los podremos encontrar de forma sencilla.
- Sin embargo hay una gran cantidad de problemas que no pueden ser solucionados con una red de una sola capa, por eso estudiaremos sus posibilidades y sus limitaciones y mas adelante veremos como expandir estas posibilidades con el uso de una red multicapa.

Función de activación signo

- Con este tipo de función de activación la salida puede valer {+1,-1}, por lo tanto los vectores de salida deseada deben estar compuestos de valores {-1,+1}.
- Queremos que se cumpla:

$$\text{sgn}(\mathbf{w} \cdot \boldsymbol{\xi}_\mu) = \zeta^\mu \qquad \text{(desired)} \qquad (5.5)$$

  - Para todo μ.
- La función $sgn(\boldsymbol{w}.\boldsymbol{\xi}^{\boldsymbol{\mu}})$ dará +1 a la salida si la proyección del vector $\boldsymbol{\xi}^{\boldsymbol{\mu}}$ sobre $\boldsymbol{w}$ es positiva, y -1 en caso contrario.
- El límite entre proyecciones positivas y negativas sobre $\boldsymbol{w}$ está dado por la condición: $\boldsymbol{w}.\boldsymbol{\xi} = 0$**.** Por lo que la condición para que el perceptrón opere correctamente es que los pesos $\boldsymbol{w}$, determinen un hiperplano que divida en dos el espacio vectorial de entradas, de forma que de un lado queden los vectores cuya salida es +1 y del otro los vectores cuya salida es -1.

Función de activación signo

- La siguiente figura muestra un ejemplo:

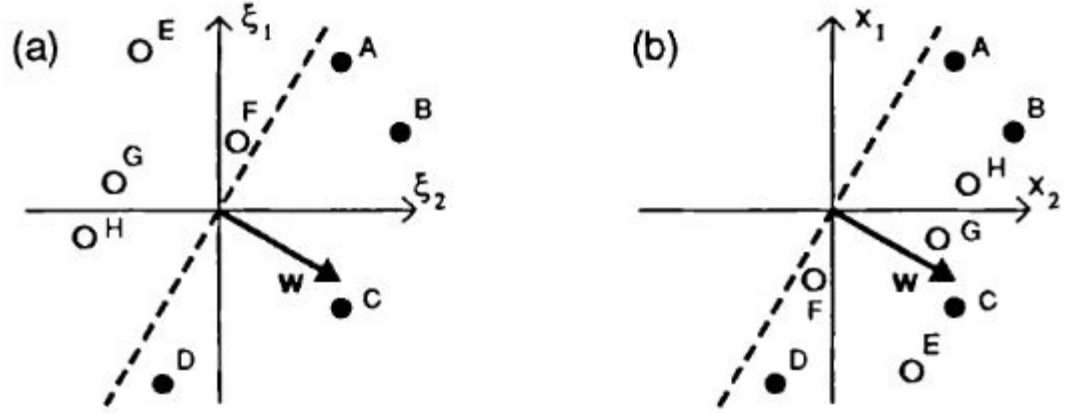


FIGURE 5.2 Geometrical illustration of the conditions (5.5) and (5.8). There are two continuous-valued inputs ξ_1 and ξ_2, and eight patterns ($\mu = 1 \ldots 8$) labelled A–H. Only one output is considered. The solid circles represent input patterns with $\zeta^\mu = +1$, whereas open circles mean $\zeta^\mu = -1$. In (a) the axes are ξ_1 and ξ_2 themselves, while in (b) they are x_1 and x_2; pattern μ has $\mathbf{x}^\mu = \zeta^\mu \boldsymbol{\xi}^\mu$. The condition for correct output is that the plane perpendicular to the weight vector $\mathbf{w}$ divides the points in (a), and lies on one side of all points in (b), as shown.

Función de activación signo

- En la figura de la derecha se muestra una representación alternativa, la cual está dada por:

$$x_k^\mu \equiv \zeta^\mu \xi_k^\mu \qquad (5.6)$$

or

$$\mathbf{x}^\mu \equiv \zeta^\mu \boldsymbol{\xi}^\mu \qquad (5.7)$$

- Lo cual traforma la condición (5.5) en:

$$\mathbf{w} \cdot \mathbf{x}^\mu > 0 \qquad \text{(desired)} \qquad (5.8)$$

- Lo cual significa que todos los vectores $\boldsymbol{x}$, deben estar del mismo lado del hiperplano perpendicular al vector **w**, para que el perceptrón funcione correctamente.

Separabilidad Lineal

- Si ese hiperplano no existe, entonces el problema no puede ser resuelto.
- Si ese hiperplano existe, se dice que el problema es **linealmente separable.**
- Si cada unidad tiene un threshold:

$$O_i = \mathrm{sgn}\Big(\sum_{k>0} w_{ik}\xi_k - w_{i0}\Big) \qquad (5.9)$$

  - O en notación vectorial:

$$O = \mathrm{sgn}(\mathbf{w}\cdot\boldsymbol{\xi} - w_0)\,. \qquad (5.10)$$

- Entonces el hiperplano que divide al espacio vectorial en dos es:

$$\mathbf{w}\cdot\boldsymbol{\xi} = w_0 \qquad (5.11)$$

  - Por lo que agregar un threshold a cada unidad nos permite que el hiperplano no necesariamente pase por el origen.

Separabilidad Lineal

- Ejemplo AND:

TABLE 5.1 AND function

ξ_1	ξ_2	ζ
0	0	−1
0	1	−1
1	0	−1
1	1	+1

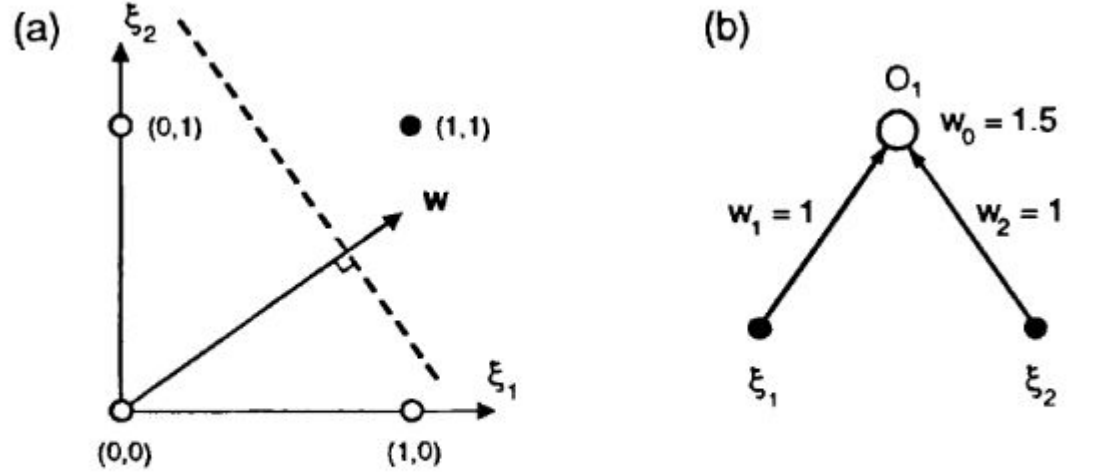


FIGURE 5.3 (a) The AND function is linearly separable (b) A perceptron that implements AND.

Separabilidad Lineal

- Ejemplo AND:

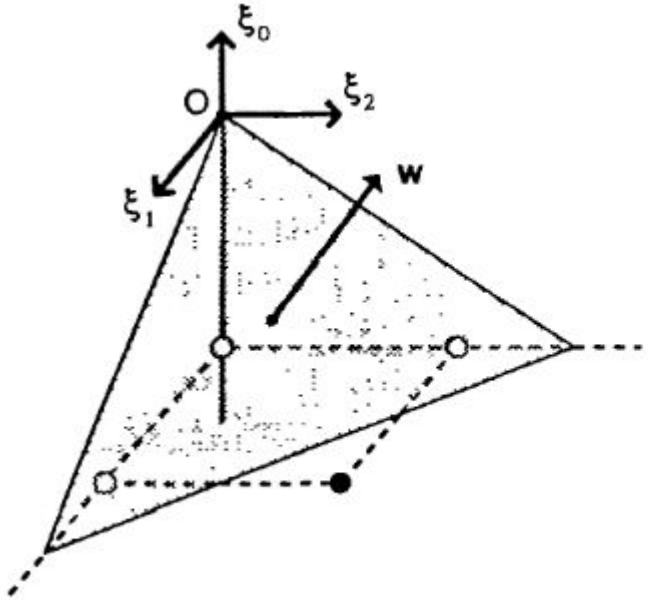


FIGURE 5.4 The AND problem with the threshold w_0 in an extra dimension ξ_0. Note that all the patterns have $\xi_0 = -1$ as in (5.2). The separating plane shown is perpendicular to the weight vector $\mathbf{w} = (1.5, 1, 1)$.

Separabilidad Lineal

- Ejemplo XOR:

TABLE 5.2 XOR function

ξ_1	ξ_2	ζ
0	0	−1
0	1	+1
1	0	+1
1	1	−1

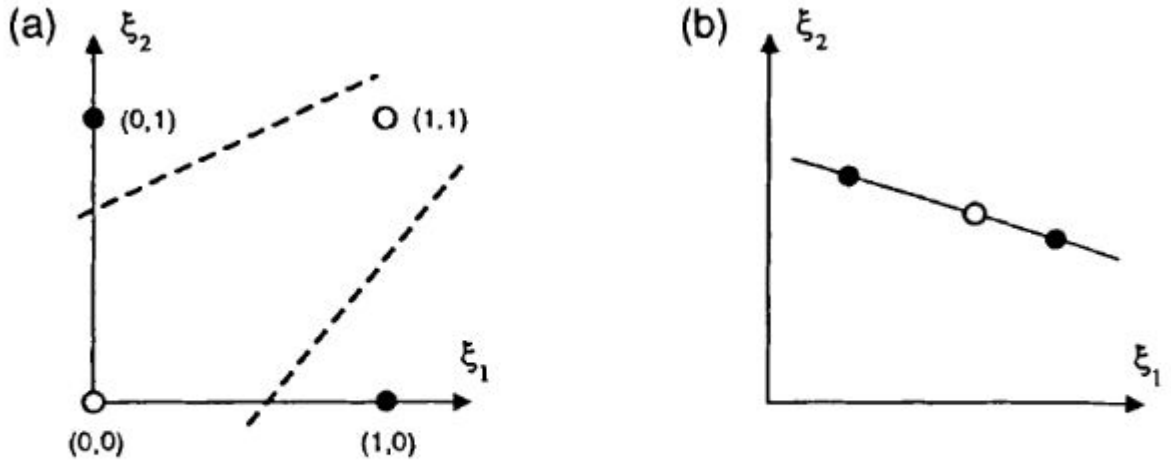


FIGURE 5.5 Problems that are *not* linearly separable. (a) The XOR problem. (b) Points that are not in "general position."

Separabilidad Lineal

- Ejemplo XOR:
  - En base a la table de verdad podemos escribir las siguientes inecuaciones:

$$w_1 + w_2 < w_0 \quad (5.12)$$
$$-w_1 - w_2 < w_0 \quad (5.13)$$
$$w_1 - w_2 > w_0 \quad (5.14)$$
$$-w_1 + w_2 > w_0\,. \quad (5.15)$$

  - De (5.12) y (5.15) obtenemos que $w_1 < 0$.
  - De (5.13) y (5.14) obtenemos que $w_1 > 0$
  - Por lo tanto no existe una solución que staisfaga todas las condiciones, por lo que el problema no es linealmente separable.

Separabilidad Lineal

- Ej. Puntos que no se encuentran en posición general:
  - Analicemos el caso de la figura (5.5b):

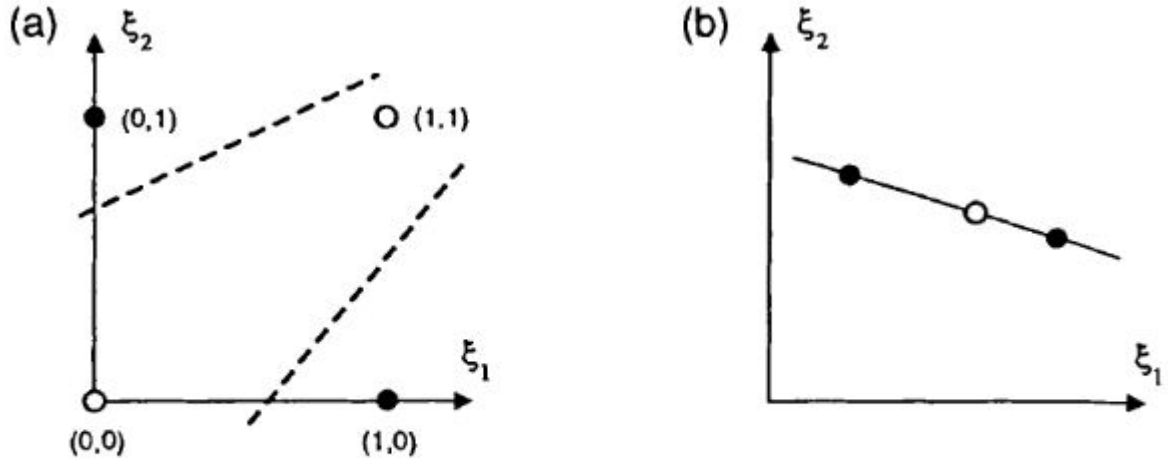


FIGURE 5.5 Problems that are *not* linearly separable. (a) The XOR problem. (b) Points that are not in "general position."

Separabilidad Lineal

- Ej. Puntos que no están en posición general:
  - El caso 5.5b es un caso muy particular ya que la probabilidad de que tres puntos estén perfectamente alineados en una recta es cero.
  - Se dice que una serie de puntos está en posición general cuando este tipo de condición no ocurre.
  - Es decir, en un espacio N-dimensional un conjunto de puntos está en posición general si no mas de d+1 puntos caen exactamente en un hiperplano d-dimensional para d<N. (d puntos definen el hiperplano y uno mas estaría contenido en este hiperplano).
  - Si no tenemos una entrada de threshold el plano pasará siempre por el origen, por lo que la condición queda que no mas de d puntos caigan exactamente por un hiperplano que pasa por el origen.
    - Esta condición puede ser expresada como: Cualquier subconjunto de N puntos o menos, debe ser linealmente independiente.

Separabilidad Lineal

- Se puede demostrar que el tipo de falla en la separabilidad lineal que se presenta en el caso de la XOR solo puede ocurrir cuando hay mas patrones p que cantidad de entradas N.
- El tipo de falla en el que un conjuntos no se encuentra en posición general, se puede presentar para cualquier p.

Algoritmo de aprendizaje

- Consideraremos problemas linealmente separables.
- Para este tipo de problemas, un algoritmo de aprendizaje puede ser:
  - Ingresar a la red con los patrones de entrenamiento uno a uno.
  - Si la salida obtenida es igual a la deseada ($O_i^\mu = \varsigma_i^\mu$), los pesos no se cambian.
  - Caso contrario:

$$w_{ik}^{\text{new}} = w_{ik}^{\text{old}} + \Delta w_{ik} \tag{5.16}$$

$$\Delta w_{ik} = \begin{cases} 2\eta\zeta_i^\mu \xi_k^\mu & \text{if } \zeta_i^\mu \neq O_i^\mu; \\ 0 & \text{otherwise;} \end{cases} \tag{5.17}$$

$$\Delta w_{ik} = \eta(1 - \zeta_i^\mu O_i^\mu)\zeta_i^\mu \xi_k^\mu \tag{5.18}$$

$$\Delta w_{ik} = \eta(\zeta_i^\mu - O_i^\mu)\xi_k^\mu . \tag{5.19}$$

Algoritmo de aprendizaje

- El parámetro η es llamado velocidad de aprendizaje.
- Para dar por correcta la salida, se tiene que cumplir:

$$sgn\left(h_i^\mu\right) = \zeta_i^\mu$$

  - Que es lo mismo que pedir que:

$$h_i^\mu \zeta_i^\mu > 0$$

  - Pero para no estar en una condición muy ajustada, también podemos pedir:

$$\zeta_i^\mu h_i^\mu \equiv \zeta_i^\mu \sum_k w_{ik} \xi_k^\mu > N\kappa \qquad \text{(desired)}. \qquad (5\ 20)$$

    - Como la sumatoria tiende a ser mas grande cada vez que aumenta N, entonces escalamos la constante κ por un factor de N.
  - Para implementar esta condición de aprendizaje, podemos escribir:

$$\Delta w_{ik} = \eta \Theta(N\kappa - \zeta_i^\mu h_i^\mu) \zeta_i^\mu \xi_k^\mu \qquad (5.21)$$

    - Donde $\ominus$ es la función escalón.

Algoritmo de aprendizaje

- Por lo que (5.21) se reduce a (5.18) para el caso $\kappa = 0$.
- La expresión (5.21) se conoce como regla de aprendizaje del perceptrón.
- Se puede demostrar que (5.21) converge a una solución en un número finitos de pasos, siempre que la solución exista.

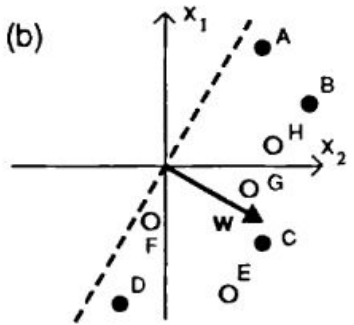
- Podemos expresar la condición de estabilidad como:

$$\mathbf{w} \cdot \mathbf{x}^{\mu} > N\kappa \qquad \text{(desired)}. \qquad (5\ 22)$$

  - Recordemos que $x^{\mu} = \xi^{\mu}\zeta^{\mu}$
  - Esto equivale a que todos los puntos en el espacio **x** deben estar a una distancia $N\kappa/|\boldsymbol{w}|$ del hiperplano perpendicular a **w.**

Algoritmo de aprendizaje

- Para la figura 5.2b, podemos ver que existe una solución siempre que κ sea lo suficientemente chico o $|\boldsymbol{w}|$ lo suficientemente grande:



- Podemos expresar el algoritmo de aprendizaje en notación vectorial para una interpretración geométrica:

$$\Delta \mathbf{w} = \eta \Theta(N\kappa - \mathbf{w}\cdot\mathbf{x}^{\mu})\mathbf{x}^{\mu} \qquad (5.23)$$

Algoritmo de aprendizaje

- Lo cual nos dice que los pesos **w** son modificados levemente en la dirección de $\boldsymbol{x^{\mu}}$ si la proyección de $\boldsymbol{wx^{\mu}}$ es menor que $N\kappa/|\boldsymbol{w}|$.
- Por ejemplo para $\kappa = 0$:

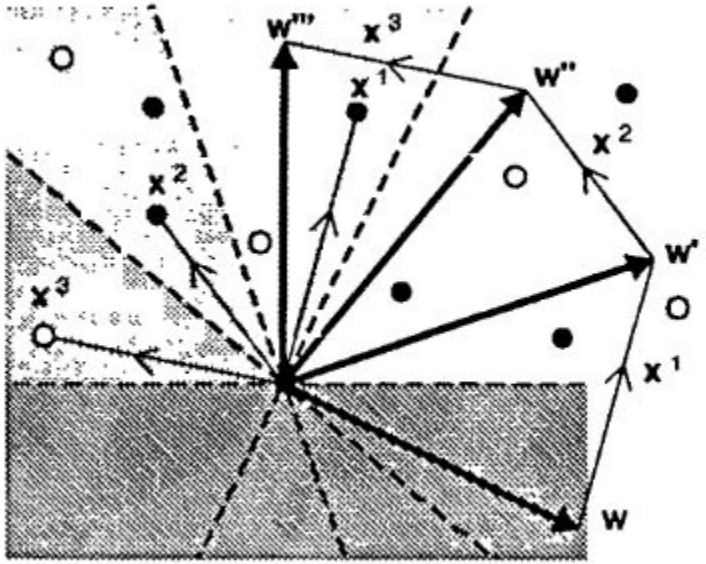


FIGURE 5.6 How the weight vector evolves during training for $\kappa = 0$, $\eta = 1$. Successive values of the weight vector are shown by **w**, **w′**, **w″**, and **w‴**. The darker and darker shading shows the "bad" region where $\mathbf{w} \cdot \mathbf{x} < 0$ for the successive **w** vectors. Each **w** is found from the previous one (e.g., **w′** from **w**) by adding an $\mathbf{x}^{\mu}$ from the current bad region.

Algoritmo de aprendizaje

- En el caso de que $\kappa > 0$ es posible encontrar un vector **w** con un módulo lo suficientemente grande de forma que las proyexiones de los vectores **x** sean todas positivas.
- Dependiendo de los patrones de entrenamiento $\boldsymbol{x}^{\boldsymbol{\mu}}$, puede ser que haya un rango bastante amplio de direcciones de **w** que sean una solución válida, que el rango sea acotado, o que no haya ninguna solución en absoluto:

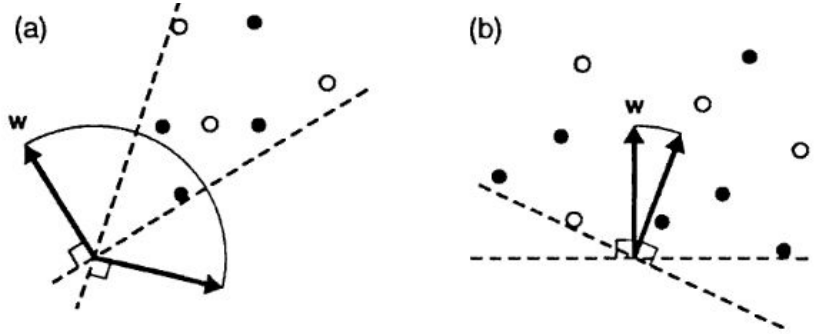


FIGURE 5.7 (a) An easy problem: any weight vector in the 135° angle shown will have positive pattern projections. (b) A harder problem where the weight vector must lie in a narrow cone.

Algoritmo de aprendizaje

- Podemos usar esta apreciación para medir qué tan fácil o difícil es encontrar una solución utilizando:

$$D(\mathbf{w}) = \frac{1}{|\mathbf{w}|} \min_{\mu} \mathbf{w} \cdot \mathbf{x}^{\mu} \qquad (5.24)$$

  - Que es la distancia del "peor" de los $\boldsymbol{x}^{\boldsymbol{\mu}}$,sobre el hiperplano perpendicular a **w**.
- Si D(**w**) es positivo, entonces todos los patrones están del lado "correcto" del hiperplano y por lo tanto hay una solución para un valor válido para un valor de $|\boldsymbol{w}|$ los suficientemente grande.
- Si maximizamos D(**w**) de forma tal que tengamos la dirección óptima de **w** para un $|\boldsymbol{w}|$ mínimo, decimos que tenemos un perceptrón óptimo.
- Es decir, D(**w**) para el perceptrón óptimo vale:

$$D_{\max} \equiv \max_{\mathbf{w}} D(\mathbf{w}) \qquad (5.25)$$

Algoritmo de aprendizaje

- El valor D_{max} nos da una pauta de que tan difícil es encontrar una solución para un problema dado.
- Si D_{max}<0 entonces el problema no puede ser resuelto.
- La siguiente figura ilustra el concepto:

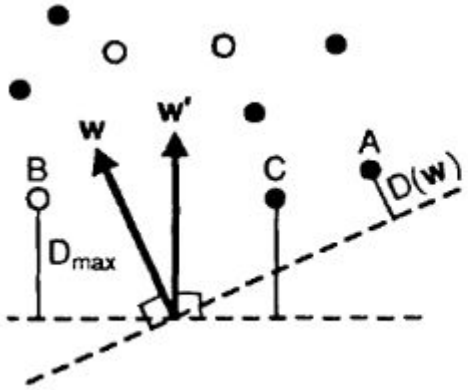


FIGURE 5 8 Definitions of $D(\mathbf{w})$ and $D_{\max}$ Pattern A is nearest to the plane perpendicular to weight vector $\mathbf{w}$, so the distance to A gives $D(\mathbf{w})$. Maximizing $D(\mathbf{w})$ with respect to $\mathbf{w}$ gives $\mathbf{w}'$, with $D(\mathbf{w}') = D_{\max}$. Note that both B and C are distance $D_{\max}$ from the plane.

Convergencia del algoritmo de aprendizaje

- Supongamos que existe un D(**w***)>0 tal que soluciona el problema que queremos resolver con el perceptrón simple.
- Veremos si podemos garantizar la convergencia utilizando el algoritmos de aprendizaje propuesto en (5.20).
- Si partimos de **w**=0, supongamos que M^{μ} es la cantidad de veces que se utilizó el patrón μ para actualizar los pesos, podemos expresar **w** como:

$$\mathbf{w} = \eta \sum_{\mu} M^{\mu} \mathbf{x}^{\mu} \qquad (5.26)$$

- La idea de la prueba de convergencia consiste en demostrar que $\frac{\boldsymbol{w}.\boldsymbol{w}^*}{|\boldsymbol{w}|}$ puede aumentar arbitrariamente a medida que incrementamos el valor de M, donde $M = \sum_{\mu} M^{\mu}$.

Convergencia del algoritmo de aprendizaje

- Evaluemos el crecimiento de **w.w*** utilizando (5.24) y (5.26):

$$\mathbf{w}\cdot\mathbf{w}^* = \eta \sum_{\mu} M^{\mu}\mathbf{x}^{\mu}\cdot\mathbf{w}^* \geq \eta M \min_{\mu} \mathbf{x}^{\mu}\cdot\mathbf{w}^* = \eta M D(\mathbf{w}^*)|\mathbf{w}^*| . \qquad (5.27)$$

  - Por lo que **w.w*** crece proporcional a M.
- Ahora acotemos el crecimiento de $|\boldsymbol{w}|$. Analicemos cuánto cambia $|\boldsymbol{w}|$ en una iteración en la que se entrena con un patrón x^{α}:

$$\begin{aligned} \Delta|\mathbf{w}|^2 &= \left(\mathbf{w} + \eta\mathbf{x}^{\alpha}\right)^2 - \mathbf{w}^2 \\ &= \eta^2\left(\mathbf{x}^{\alpha}\right)^2 + 2\eta\mathbf{w}\cdot\mathbf{x}^{\alpha} \\ &\leq \eta^2 N + 2\eta N\kappa \\ &= N\eta(\eta + 2\kappa) . \end{aligned} \qquad (5.28)$$

- La inecuación es válida ya que $N\kappa \geq \boldsymbol{w}.\boldsymbol{x}^{\alpha}$ ya que de otra forma no estaríamos actualizando **w**.

Convergencia del algoritmo de aprendizaje

- Para la demostración se usó $x_k^\alpha = \pm 1$, por lo que $(\boldsymbol{x}^\alpha)^2 = N$. Se puede extender la demostración para $x_k^\alpha \in \mathbb{R}$.
- Si sumamos los incrementos M veces nos queda:

$$|\mathbf{w}|^2 \leq MN\eta(\eta + 2\kappa)\,. \qquad (5.29)$$

  - Por lo que $|\boldsymbol{w}|$ crece a lo sumo como $\sqrt{M}$ y de (5.27), $\frac{\boldsymbol{w}.\boldsymbol{w}^*}{|\boldsymbol{w}|}$ crece por lo menos a razón de $\sqrt{M}$. Pero esto no puede continuar mas allá de la condición de convergencia, en cuyo caso $\frac{\boldsymbol{w}.\boldsymbol{w}^*}{|\boldsymbol{w}|}$ deja de crecer.
- Recordemos que el producto escalar normalizado:

$$\phi \doteq \frac{(\mathbf{w}\cdot\mathbf{w}^*)^2}{|\mathbf{w}|^2|\mathbf{w}^*|^2} \qquad (5.30)$$

- Es el coseno cuadrado del ángulo entre **w** y **w***, por lo que será siempre menor que 1.

Convergencia del algoritmo de aprendizaje

- Si agregamos a esta cota las inecuaciones (5.27) y (5.29) nos queda:

$$1 \geq \phi \geq M \frac{D(\mathbf{w}^*)^2 \eta}{N(\eta + 2\kappa)} \qquad (5.31)$$

- De donde podemos deducir una cota superior para la cantidad de actualizaciones de pesos (usando el mejor **w*** posible):

$$M \leq N \frac{1 + 2\kappa/\eta}{D_{\max}^2}. \qquad (5.32)$$

  - Esta cota es proporcional a las unidades de entrada.
  - No depende de la cantidad de patrones p (aunque todos deben ser verificados antes de confirmar la situación de convergencia).
  - Dmax tiende a achicarse a medida que aumenta p, generando que M aumente.
  - La cota de M aumenta a medida que aumenta k ya que el módulo total de **w** tiene que ser aún mayor mas allá de que la dirección sea la correcta, y eso se logra con mas iteraciones