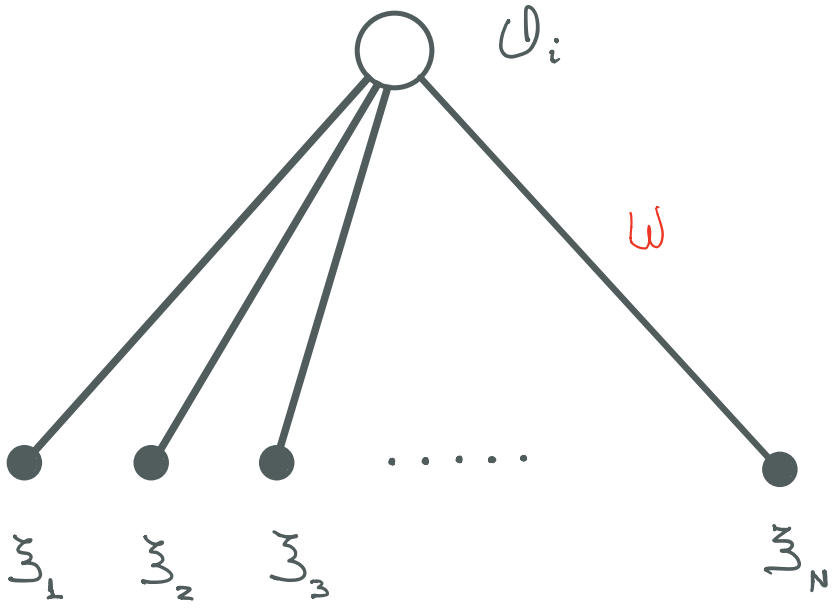
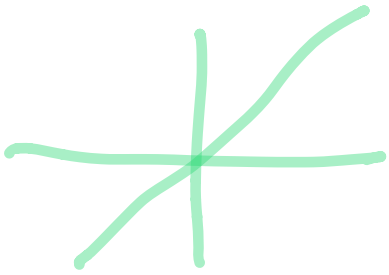
PERCEPTRON SIMPLE CON NEURONA LINEAL

Cuando describí las posibles funciones de activación $g(z)$, las cuales debemos recordar, definen la neurona de salida, no contemplé una función lineal como posibilidad, pero ahora analizaremos el caso del perceptrón simple con una neurona de salida lineal. Recuerden que si la capa de salida tiene más de una neurona, estas siempre se pueden desacoplar.



$$g(z) = z$$



$$\underline{V_i = g(h_i)} = h_i = \sum_k w_{ik} \xi_k$$

Como antes, suponemos que tenemos p entradas previamente etiquetadas

$$\vec{\xi}^\mu \longmapsto \zeta_i^\mu \quad \text{(valor de salida)}$$

(1) $$\zeta_i^\mu = \sum_k w_{ik} \xi_k^\mu \quad \text{(valor deseado)}$$

Nota: la salida puede tomar cualquier valor real, por lo cual no estamos haciendo una clasificación sino una REGRECIÓN

Existe una solución explícita llamada PSEUDO INVERSA que nos da los valores de los componentes del vector $\vec{w}_i = (w_{i1}, w_{i2}, \dots, w_{iN})$ y tiene la forma

$$w_{ik} = \frac{1}{N} \sum_{\mu\nu} \xi_i^\mu \left(Q^{-1}\right)_{\mu\nu} \xi_k^\nu$$

$$Q = \begin{pmatrix} q_{11} & q_{1p} \\ q_{p1} & q_{pp} \end{pmatrix} = \frac{1}{N} \left(\xi_i^1, \dots \xi_i^p\right) Q^{-1} \begin{pmatrix} \xi_i^1 \\ \vdots \\ \xi_i^p \end{pmatrix}$$

donde

$$Q_{\mu\nu} = \frac{1}{N} \sum_{k=1}^{N} \xi_k^\mu \xi_k^\nu = \frac{1}{N} \vec{\xi}^\mu \cdot \vec{\xi}^\nu$$

es la matriz de superposiciones entre los elementos del conjunto de entrenamiento.

Se puede probar que si usamos esta expresión y la reemplazamos en (1) se cumple la condición.

Para que exista la solución, debe existir Q^{-1}, y para esto los vectores $\vec{\xi}^\mu$ $(\mu = 1, 2, \dots, p)$ deben ser

linealmente independientes. O sea, si existe una relación lineal de la forma

$$a_1 \xi_k^1 + a_2 \xi_k^2 + \dots + a_p \xi_k^p = 0 \qquad \text{para todo } k \ (k = 1, 2, \dots, N)$$

el problema no tiene solución.

Descenso por el gradiente

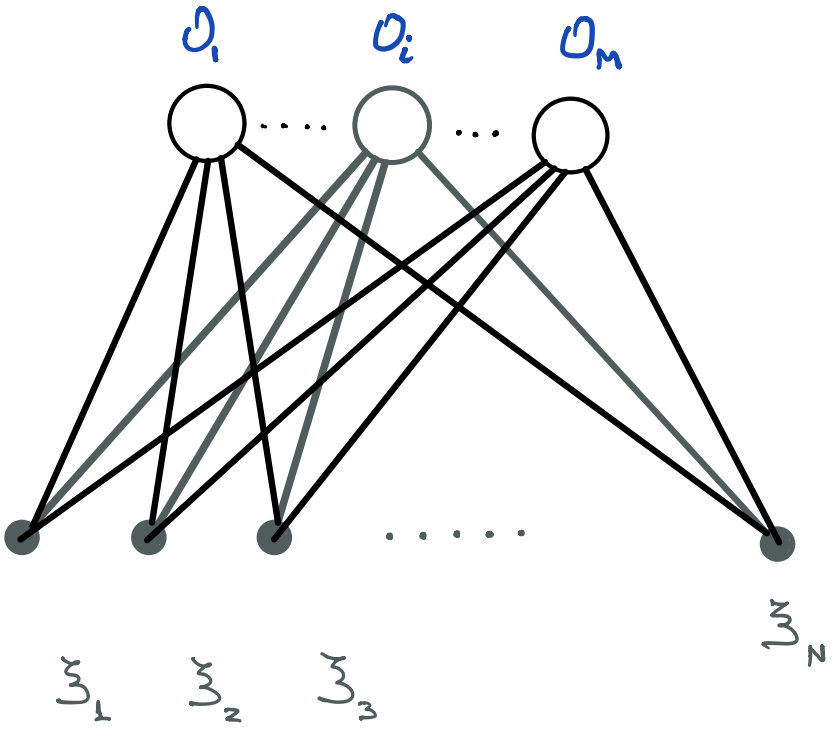
Hoy presentamos este método por primera vez pero nos acompañará SIEMPRE en los modelos neuronales de aprendizaje automático.

Definimos una función error o función costo de la forma

$$E(\vec{w}) = \frac{1}{2} \sum_{i}^{M} \sum_{\mu=1}^{p} (\zeta_i^\mu - O_i^\mu)^2$$

suma sobre todas las neuronas de la capa de salida

suma sobre los elementos del conjunto de entrenamiento



Cuando le presentamos la entrada μ obtenemos el resultado dado por la regla del perceptron lineal

$$O_i^\mu = \sum_k w_{ik} \xi_k^\mu \qquad g(z) = z$$

La idea es comenzar con un vector inicial arbitrario $\vec{w}$ y presentarle el ejemplo μ

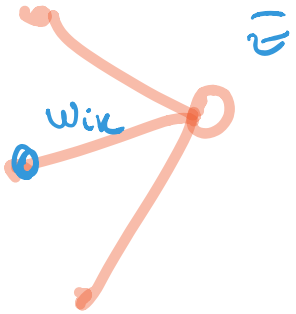
$$E : \mathbb{R}^N \longrightarrow \mathbb{R}$$

$$\nabla E : \mathbb{R}^N \longrightarrow \mathbb{R}^N \qquad \nabla E_{ik} = \frac{\partial E}{\partial w_{ik}}$$

$$w_{ik}^{\text{nuevo}} = w_{ik}^{\text{anterior}} + \Delta w_{ik}$$

$$\nabla E = \left(\frac{\partial E}{\partial w_{i1}}, \frac{\partial E}{\partial w_{i2}}, \dots, \frac{\partial E}{\partial w_{iN}} \right)$$

$$\Delta w_{ik} = -\eta \frac{\partial E}{\partial w_{ik}}$$



$$= -\eta \frac{\partial}{\partial w_{ik}} \left(\frac{1}{2} \sum_{\mu=1}^{P} \left(\xi_i^\mu - O_i^\mu \right)^2 \right)$$

$$= -\eta \sum_{\mu=1}^{P} \frac{1}{2} \frac{\partial}{\partial w_{ik}} \left(\xi_i^\mu - \sum_j w_{ij} \xi_j^\mu \right)^2$$

$$= -\eta \sum_{\mu=1}^{P} \frac{2}{2} \left(\xi_i^\mu - \sum_j w_{ij} \xi_j^\mu \right) \frac{\partial}{\partial w_{ik}} \left(- \sum_j w_{ij} \xi_j^\mu \right)$$

$$= \eta \sum_{\mu=1}^{P} \left(\xi_i^\mu - O_i^\mu \right) \xi_k^\mu$$

$$\Delta w_{ik} = \eta \sum_{\mu=1}^{P} \left(\xi_i^\mu - O_i^\mu \right) \xi_k^\mu$$

$$\Delta \bar{w}_i = \eta \sum_{\mu} \left(\xi_i^\mu - O_i^\mu \right) \bar{\xi}_k$$

$$w_{ik}^{\text{nuevo}} = w_{ik}^{\text{anterior}} + \Delta w_{ik}$$

$$\bar{w}_i^{\text{nuevo}} = \bar{w}_i^{\text{anterior}} + \Delta \bar{w}_i$$

Si definimos

$$\delta_i^\mu = \zeta_i^\mu - O_i^\mu \qquad \text{(delta } i\,\mu\text{)}$$

entonces

$$\bar{w}_i^{\text{nuevo}} = \bar{w}^{\text{anterior}} + \eta \sum_\mu \delta_i^\mu \bar{\xi}^\mu$$

$$w_{ik}^{\text{nuevo}} = w_{ik}^{\text{anterior}} + \eta \sum_\mu \delta_i^\mu \xi_k^\mu$$

Esta regla de descenso por el gradiente se conoce como

- Regla delta o ADELINE (Widrow y Hoff) (1960)
- Regla least mean square (LMS) (1986)
- Regla Rescorla - Wagner (1972)

Tenemos M vectores sinápticos $\vec{w}_i$ con

$i = 1, 2, \ldots, M$

$$\vec{w}_1 = (w_{11}, w_{12}, \ldots w_{1N})$$

$$\vec{w}_2 = (w_{21}, w_{22}, \ldots w_{2N})$$

$$\vdots$$

$$\vec{w}_M = (w_{M1}, w_{M2}, \ldots w_{MN})$$

Entonces la función costo depende de $M.N$ coeficientes para un dado problema.

Recordemos que el problema es encontrar los $M.N$ coeficientes con los cuales el perceptron lineal resuelve bien todos los ejemplos del conjunto de entrenamiento

$$\{(\xi^\mu, \zeta^\mu) / \mu = 1, 2, \ldots, p\}$$

El gradiente de $E(\{\bar{w}\})$ es un vector en

$\mathbb{R}^{M \times N}$

$$\nabla E = \overbrace{\left(\frac{\partial E}{\partial w_{11}} , \ldots\ldots , \frac{\partial E}{\partial w_{MN}} \right)}^{M \times N}$$

$\leftarrow$ $M \cdot N$ elementos $\rightarrow$

El gradiente es un vector que apunta en la dirección de mayor crecimiento de $E(\{\bar{w}\})$ en $\mathbb{R}^{M \times N}$.

Comenzamos dijiendo los $M \times N$ vectores $\bar{w}_i$ $(i = 1 \ldots M)$ al azar.

APRENDIZAJE EN BATCH

- Sea $\mu = 1, 2, 3, \ldots P$
- le presentamos $\bar{\xi}^\mu$ a la red y obtenemos los M valores O_i^μ.
- Con estos M valores calculo cada una de las componentes del gradiente de E $(M \times N)$

- con los elementos del gradiente calculo las variaciones que aplicaremos a cada w_{ik}

$$\Delta w_{ik} = -\eta \frac{\partial E}{\partial w_{ik}} = \eta \sum_{\mu} \delta_i^{\mu} \xi_k^{\mu}$$

$$\delta_i^{\mu} = \left(\xi_i^{\mu} - \theta_i^{\mu} \right)$$

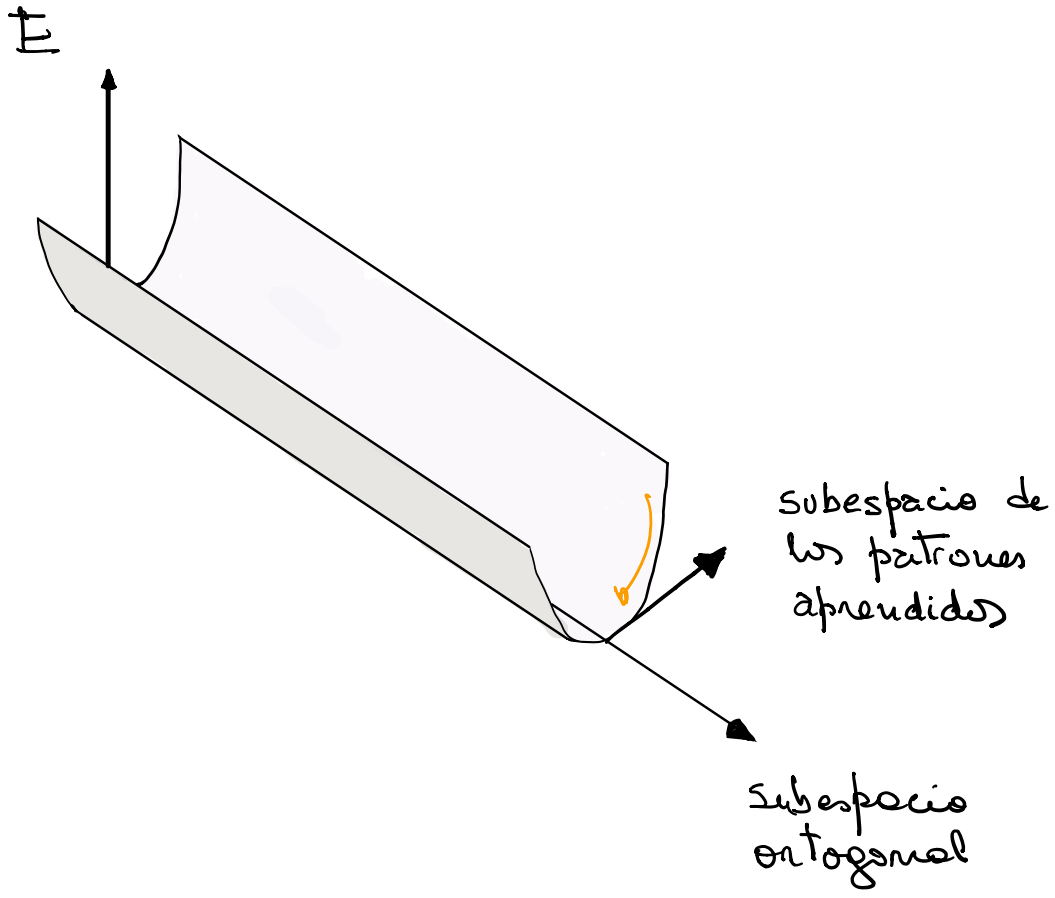
- con los Δw_{ik} actualizamos TODOS los acoplamientos sinápticos

$$w_{ik}^{\text{nuevo}} = w_{ik}^{\text{anterior}} + \Delta w_{ik}$$

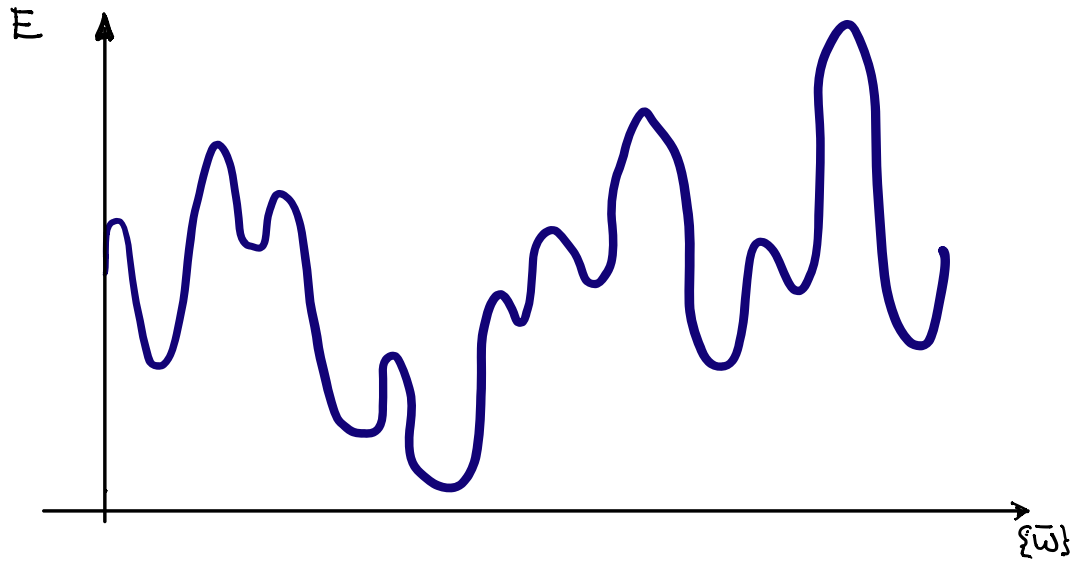
$$= w_{ik}^{\text{anterior}} + \eta \, \delta_i^{\mu} \xi_k^{\mu}$$

- empezamos de nuevo

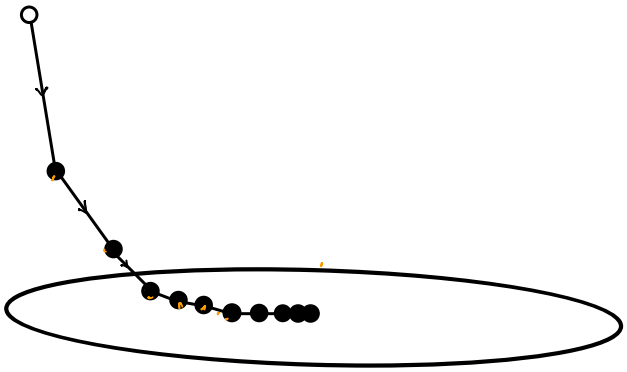
y volvemos al inicio a menos que $\mu = P$.



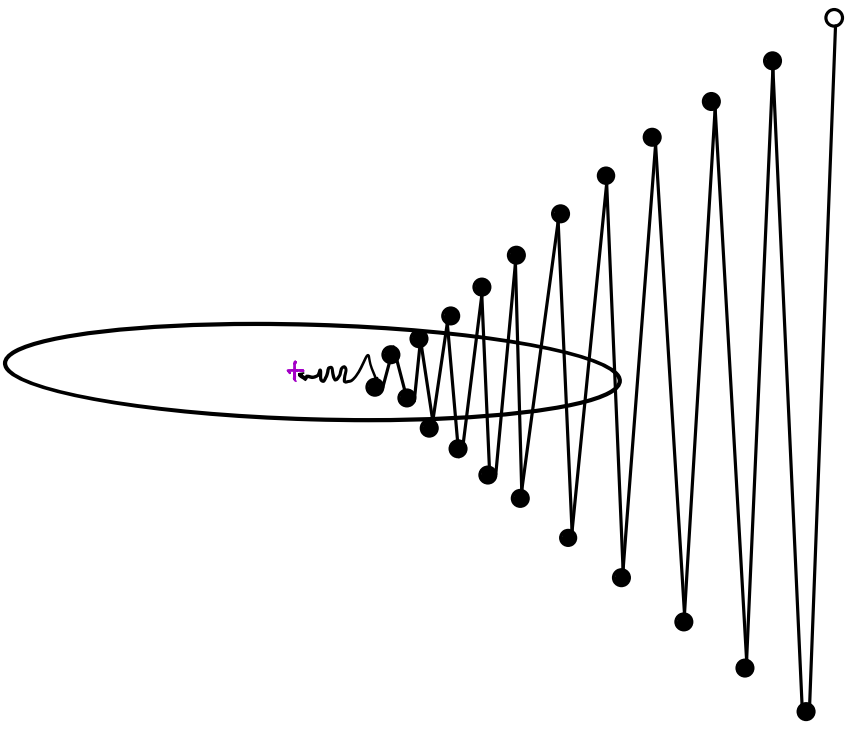
Ahora no vale la separabilidad lineal. Tenemos que intentar aprender descendiendo por la función error, pero E es una función muy rugosa



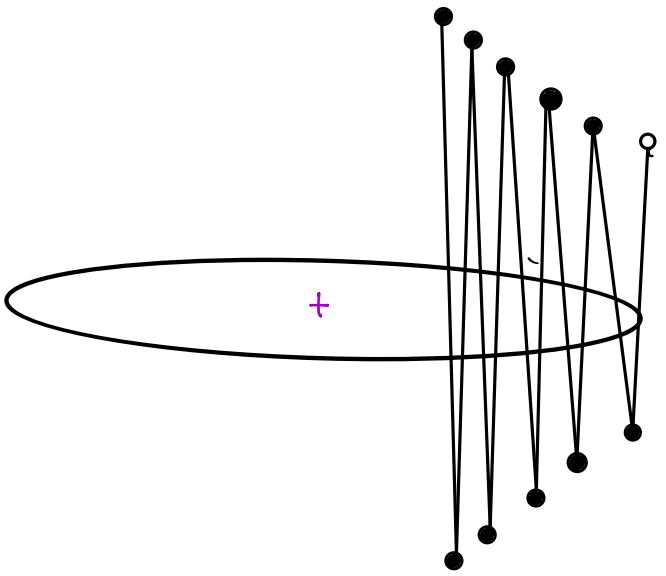
EJEMPLO : $E = x^2 + 20y^2$



$\eta = 0.02$



$\eta = 0.476$



$\eta = 0.505$

En el caso de neuronas de salida lineales, la forma cuadrática de la función E se puede diagonalizar si los ejemplos del conjunto de entrenamiento son linealmente independientes.

$$E = \sum_{\lambda=1}^{M \times N} a_\lambda \left(W_\lambda - W_\lambda^0 \right)^2$$

W_λ : combinación lineal de los w_{ik}.

Por la forma cuadrática de E los autovalores a_λ son positivos o nulos. Los autovectores asociados a los autovalores de valor cero definen la parte de E independiente de w_λ's.

Si hacemos un descenso por el gradiente en la base diagonalizada

$$\Delta w_\lambda = -\eta \frac{\partial E}{\partial w_\lambda} = -2\eta a_\lambda (w_\lambda - w_\lambda^0)$$

Tenemos que llegar en el menor número de pesos a w_λ^0 (un número que surge de la diagonalización)

$$\delta w_\lambda^{opt} = w_\lambda - w_\lambda^0 \quad \text{en dirección de } \lambda$$

$$\begin{aligned} \delta w_\lambda^{nuevo} &= \delta w_\lambda^{old} + \Delta w_\lambda \\ &= w_\lambda - w_\lambda^0 - 2\eta a_\lambda (w_\lambda - w_\lambda^0) \\ &= (w_\lambda - w_\lambda^0).(1 - 2\eta a_\lambda) \\ &= \delta_\lambda^{opt}.(1 - 2\eta a_\lambda) \end{aligned}$$

η es tal que $|1 - 2\eta a_\lambda| < 1$

El mayor λ_λ define, a través de su autovecto, la dirección de mayor curvatura de E

$$\eta < \frac{1}{\lambda_\lambda^{max}}$$

Por otro lado, el menor autovalor positivo define la dirección de más lenta aproximación al mínimo. Así, si $\lambda_\lambda^{max} / \lambda_\lambda^{min}$ es un número muy grande, estaremos en problemas y el método convergirá muy lentamente.

Nota:

* no pedimos que el problema sea linealmente separable
* no encontramos una solución perfecta
* igual que con el problema de clasificación, estudiamos el error vs. épocas. Solo que ahora la medida de error es la función E
* los computólogos llaman a la

función error como función de perdida

o esto