Módulo 3. IA y grandes volúmenes de datos

#3. Optimización y aprendizaje

Regresión polinomial

- **Función de predicción lineal:**

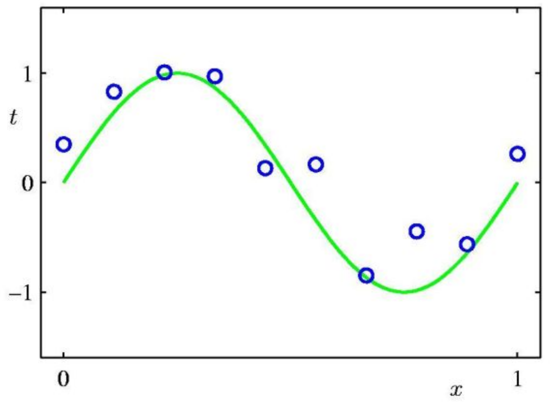
$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M$$

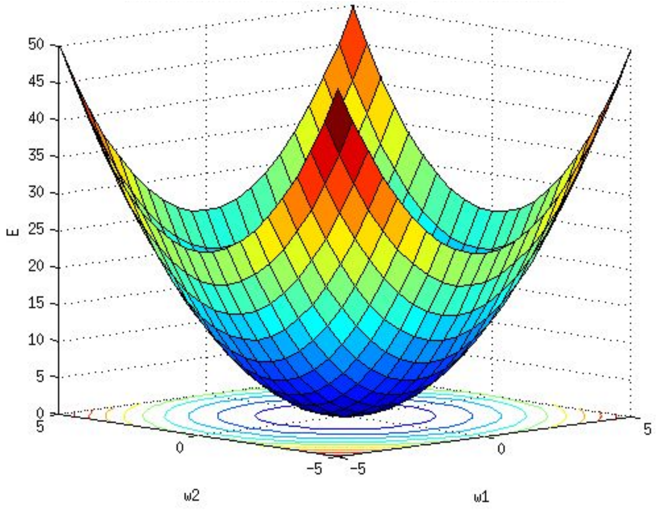
- **Función de costo**: error cuadrático

  medida del error en la predicción de t mediante $y(x; w)$

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$

- *Admite una solución en forma cerrada*



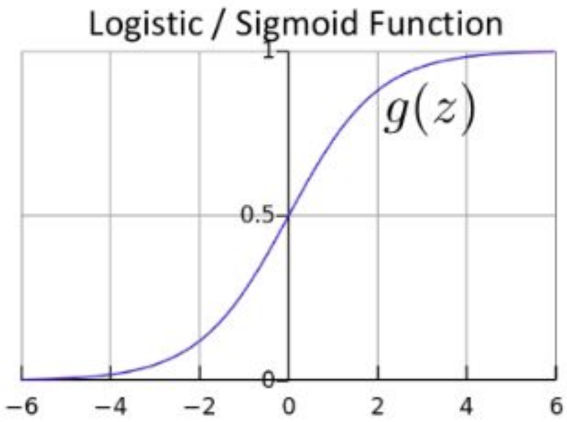


Regresión logística

- Dados $\{(x_1, y_1), \ldots, (x_N, y_N)\}$, con $x_i \in \mathbb{R}^n$, $y_i \in \{0, 1\}$

- Modelo: $p(y=1|x)=h_w(x)$

$$h_w(x) = \frac{1}{1 + exp(-w^T x))}$$



- Función de costo

$$L(w) = -\sum_{i=1}^{N} y_i \log\left(h_w(x_i)\right) + (1 - y_i)\log\left(1 - h_w(x_i)\right)$$

- $h_w(x)$ no lineal → ***no admite solución en forma cerrada***

Aprendizaje supervisado

Funciones $\mathcal{F}$

$f : \mathcal{X} \to \mathcal{Y}$

Datos de entrenamiento

$\{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}$

Aprendizaje

$$\boxed{\begin{aligned} &\text{find } \hat{f} \in \mathcal{F} \\ &\text{s.t. } y_i \approx \hat{f}(x_i) \end{aligned}}$$

Predicción

$\boxed{y = \hat{f}(x)} \Leftarrow \boxed{x}$

Dato nuevo

Optimización y aprendizaje

- Un problema típico en ML se puede escribir como:

$$L(w) = \sum_{i=1}^{N} \ell(y_i, f_w(x_i)) + \lambda R(w)$$

costo de predicción del par (x_i, y_i)

regularización para controlar el *overfitting*

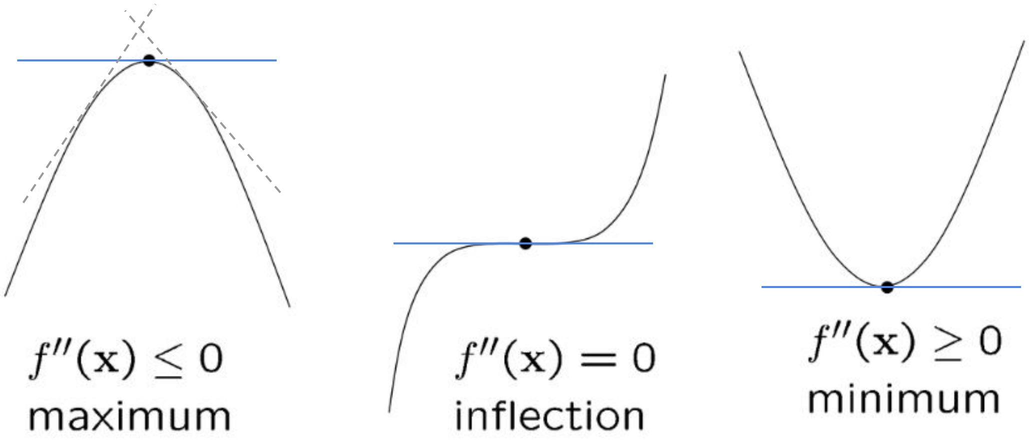
- "Aprender" significa resolver:

$$w^* = \arg\min_{w} L(w)$$

… y que $f_{w*}(.)$ pueda generalizar a ejemplos no vistos.

Optimización sin restricciones

- Recordemos. Caso de funciones 1D, $f : \mathbb{R} \longrightarrow \mathbb{R}$
  - f tiene un **punto estacionario** en x_0 cuando $\frac{\partial f}{\partial x}(x_0) = 0$
  - la derivada segunda determina el **tipo** de punto estacionario

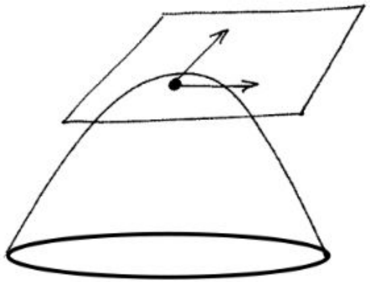


Optimización sin restricciones

- Recordemos. Caso de funciones nD, $f : \mathbb{R}^n \longrightarrow \mathbb{R}$
  - f tiene un **punto estacionario** en x_0 cuando

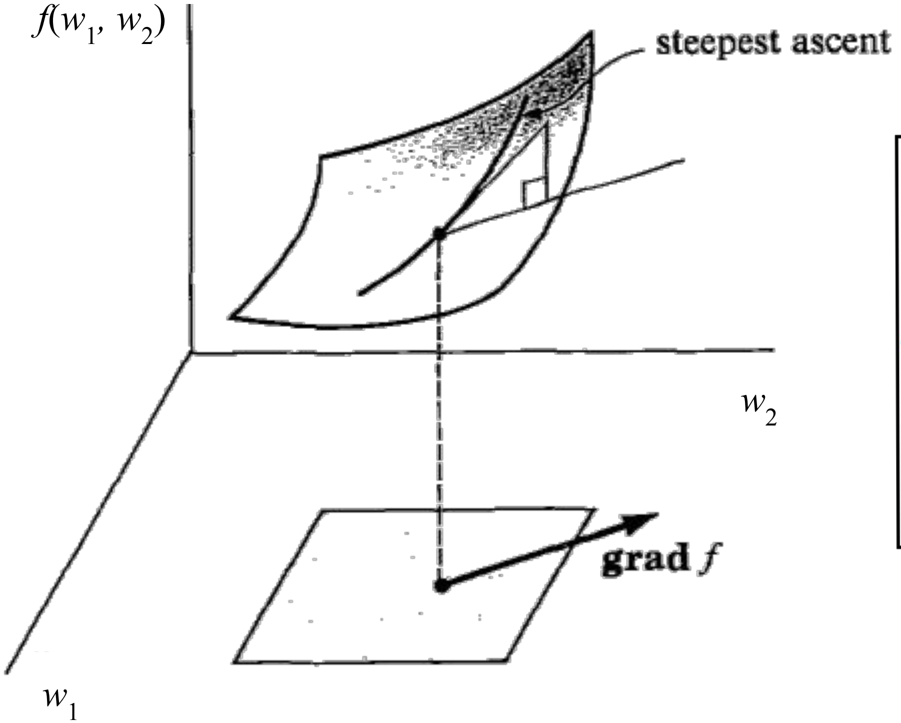
$$\nabla f(\mathbf{x}_0) \equiv \left(\frac{\partial f}{\partial x_1}, \cdots, \frac{\partial f}{\partial x_n} \right) (\mathbf{x}_0) = \mathbf{0}$$

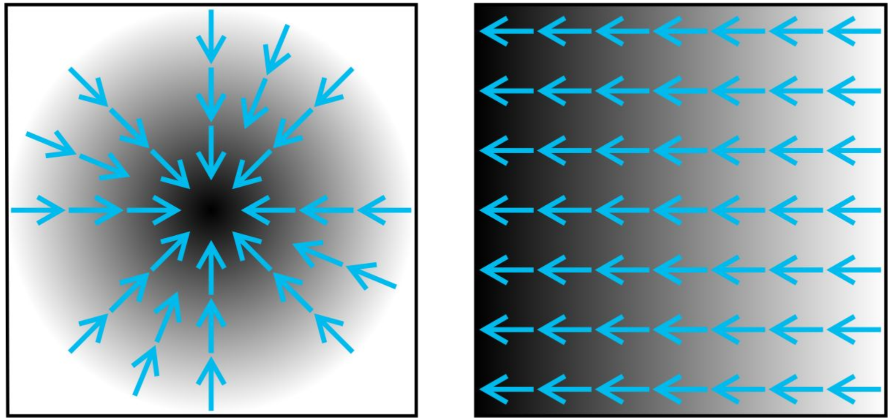
  - el **tipo** de punto estacionario lo determina la matriz Hessiana



$$\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right)^\top = \mathbf{0}$$

Optimización sin restricciones





https://es.wikipedia.org/wiki/Gradiente

- el gradiente en un punto da la dirección de máximo crecimiento

Optimización sin restricciones

- Idea general de los métodos (iterativos) de descenso: partir de algún w, avanzar en direcciones de f decrecientes

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + \Delta\mathbf{w}^{(t)}$$

- Variantes:
  - **Descenso de gradiente:** $\Delta\mathbf{w}^{(t)} = -\eta_t \nabla L(\mathbf{w}^{(t)})$
  - Descenso de gradiente estocástico: $\Delta\mathbf{w}^{(t)} = -\eta_t \nabla \ell_n(\mathbf{w}^{(t)})$
  - Newton-Raphson (segundo orden), etc.

Descenso de gradiente

Entrada:

- Conjunto de pares entrenamiento $D=\{(\mathbf{x}_i, y_i)\}$
- Una tasa de aprendizaje η

Algoritmo:

- Inicializar $\mathbf{w}^{(0)} \in \mathbb{R}^n$
- Repetir hasta converger
  - $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta \nabla_{\mathbf{w}} \mathbf{L}(\mathbf{w}^{(t)}; D)$
- Retornar último $\mathbf{w}$

Descenso de gradiente

Entrada:

- Conjunto de pares entrenamiento $D=\{(\mathbf{x}_i,y_i)\}$, $i=1,...,N$
- Una tasa de aprendizaje η

Algoritmo:

- Inicializar $\mathbf{w}^{(0)} \in \mathbb{R}^n$
- Repetir hasta converger
  - $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta \nabla_{\mathbf{w}} \mathbf{L}(\mathbf{w}^{(t)}; D)$
- Retornar último $\mathbf{w}$

Convergencia:

- no hay cambio (p.ej. norma del gradiente menor a ε)
- número de iteraciones (*early stopping*)

Ejemplo

- Datos de entrenamiento $D=\{(x_i, y_i)\}$, i=1, …, N, $x \in \mathbb{R}^n$, $y \in \mathbb{R}$

- **Modelo lineal** (parametrización): $f_w(x) = w^T x$

- **Función de costo error cuadrático**: $\ell(f_w(x), y) = ½(f_w(x) - y)^2 = ½(w^T x - y)^2$

- Problema a resolver:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} L(\mathbf{w}) = \arg\min_{\mathbf{w}} \frac{1}{N} \sum_{i=1}^{N} \ell(f_{\mathbf{w}}(\mathbf{x}_i), y_i)$$

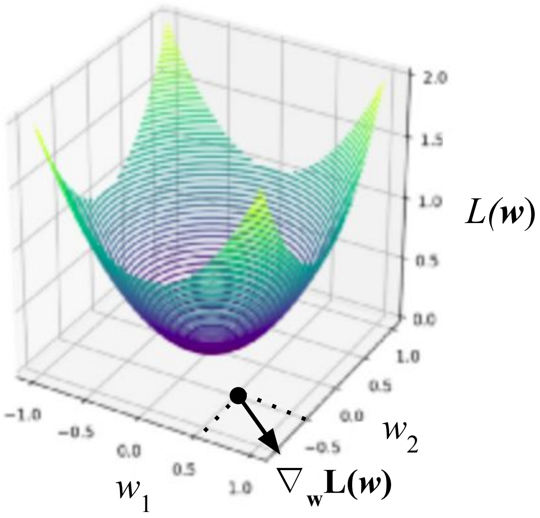
- Paso de actualización: $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta \underbrace{\nabla_{\mathbf{w}} \mathbf{L}(\mathbf{w}^{(t)}; \mathrm{D})}_{\text{gradiente}}$

Ejemplo

- Gradiente(*): $$\nabla_{\mathbf{w}} L(\mathbf{w}; D) \equiv \begin{pmatrix} \frac{\partial L(\mathbf{w};D)}{\partial w_1} \\ \vdots \\ \frac{\partial L(\mathbf{w};D)}{\partial w_n} \end{pmatrix} = \nabla_{\mathbf{w}} \frac{1}{N} \sum_{i=1}^{N} \ell(f_{\mathbf{w}}(\mathbf{x}_i), y_i)$$

$$= \frac{1}{N} \sum_{i=1}^{N} \nabla_{\mathbf{w}} \left[\frac{1}{2} \left(\mathbf{w}^T \mathbf{x}_i - y_i \right)^2 \right]$$

$$= \frac{1}{N} \sum_{i=1}^{N} \underbrace{\left(\mathbf{w}^T \mathbf{x}_i - y_i \right)}_{\text{error de predicción (salida)}} \mathbf{x}_i$$



variable (entrada)

(*) The Matrix Cookbook, https://www2.imm.dtu.dk/

Ejemplo

Entrada:

- Conjunto de pares entrenamiento $D=\{(\mathbf{x}_i,y_i)\}$, $i=1,\ldots,N$
- Una tasa de aprendizaje η

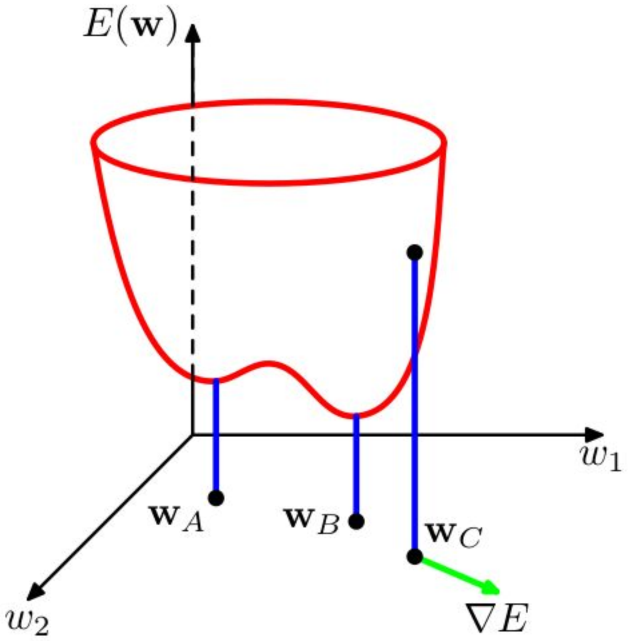
Algoritmo:

- Inicializar $\mathbf{w}^{(0)} \in \mathbb{R}^n$
- Repetir hasta converger:

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta \frac{1}{N} \sum_{i=1}^{N} \left({\mathbf{w}^{(t)}}^T \mathbf{x}_i - y_i \right) \mathbf{x}_i$$

- Retornar último $\mathbf{w}$

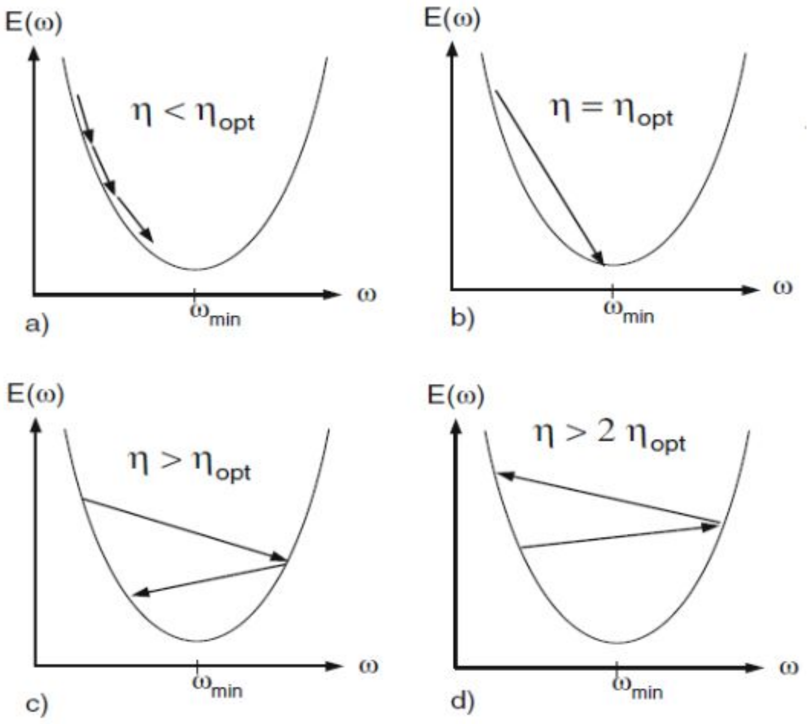
Descenso de gradiente



- ¿La solución es única?
- ¿Depende del punto de inicio?

Elección de la tasa de aprendizaje

- La tasa de aprendizaje determina la velocidad de convergencia.
- Muchas estrategias de adaptación (*momentum*, *scheduling*, etc).
- El hiperparámetro más importante junto con el número máximo de iteraciones / épocas.



El trabajo en aprendizaje supervisado

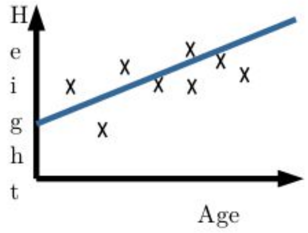
1. Obtener un conjunto de pares de entrenamiento $D=\{(\mathbf{x}_i, y_i)\}$, $i=1, ..., N$
2. Elegir una parametrización para el modelo predictivo, $f_{\mathbf{w}}(x)$
3. Elegir una función de costo, $\ell(y, f_{\mathbf{w}}(x))$
4. Resolver el problema de aprendizaje

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \frac{1}{N} \sum_{i=1}^{N} \ell(y_i, f_{\mathbf{w}}(\mathbf{x}_i)) + \lambda R(\mathbf{w})$$

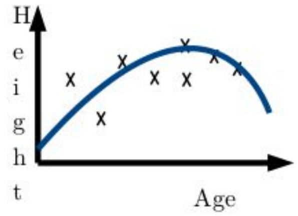
5. Evaluar, ajustar parámetros (λ, η, T_{max}, etc.), volver a iterar ...

Definición del modelo predictivo (parametrización)

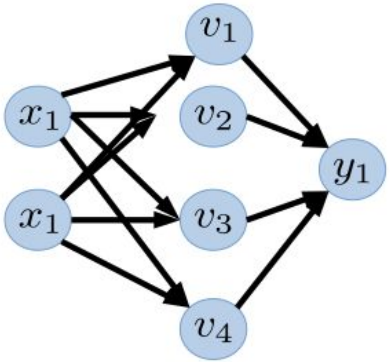
Linear: $$h_w(x) = \sum_{i=0}^{d} w_i x_i$$



Polinomial: $$h_w(x) = \sum_{i,j=0}^{d} w_{ij} x_i x_j$$



Neural Net:



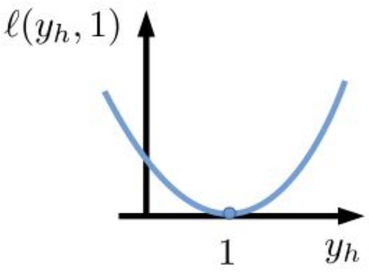
$exe:$

$$v_1 = \text{sign}(w_{11}x_1 + w_{12}x_2)$$

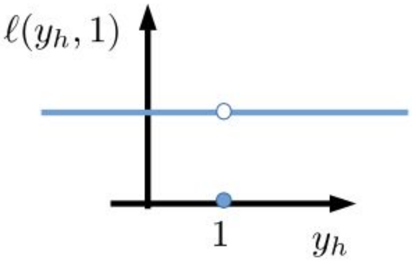
$$v_4 = 1/\left(1 + \exp(w_{41}x_1 + w_{42}x_2)\right)$$

Definición de función de costo

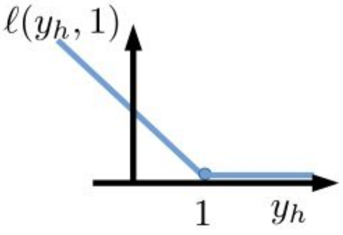
Quadratic Loss $\ell(y_h, y) = (y_h - y)^2$



Binary Loss $\ell(y_h, y) = \begin{cases} 0 & \text{if } y_h = y \\ 1 & \text{if } y_h \neq y \end{cases}$



Hinge Loss $\ell(y_h, y) = \max\{0, 1 - y_h y\}$



Ejemplo: regresión lineal

Linear hypothesis

$$h_w(x) = \langle w, x \rangle$$

L2 regularizor

$$R(w) = ||w||_2^2$$

L2 loss

$$\ell(y_h, y) = (y_h - y)^2$$

Ridge Regression

$$\min_{w \in \mathbf{R}^d} \frac{1}{n} \sum_{i=1}^{n} (y^i - \langle w, x^i \rangle)^2 + \lambda ||w||_2^2$$

Ejemplo: regresión lineal

Linear hypothesis

$$h_w(x) = \langle w, x \rangle$$

+

L2 regularizor

$$R(w) = ||w||_2^2$$

Logistic loss

$$\ell(y_h, y) = \ln(1 + e^{-y y_h})$$

Label encoding: $y \in \{-1, +1\}$

$$\mathbb{P}(y = 1|z) = \sigma(z) = \frac{1}{1 + e^{-z}}$$

$$\mathbb{P}(y = 0|z) = 1 - \sigma(z) = \frac{1}{1 + e^{z}}$$

↓

Logistic Regression

$$\min_{w \in \mathbf{R}^d} \frac{1}{n} \sum_{i=1}^{n} \ln(1 + e^{-y^i \langle w, x^i \rangle}) + \lambda ||w||_2^2$$