Módulo 3. IA y grandes volúmenes de datos

#2. El problema de clasificación

Clasificación binaria

- Disponemos de N pares de entrenamiento (observaciones)

$$\{(x_i, y_i)\}_{i=1}^{N} = \{(x_1, y_1), \cdots, (x_N, y_N)\}$$

con $x_i \epsilon \mathbb{R}^n$, $y_i \epsilon \{-1, +1\}$.

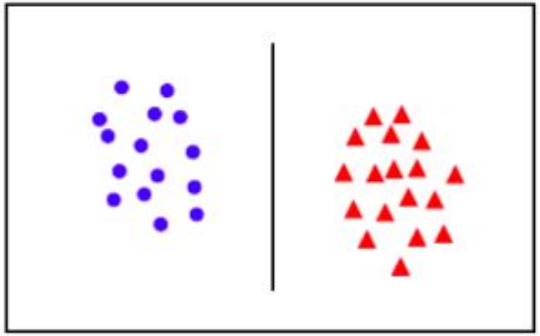
- Aprender una $f(x)$ tal que

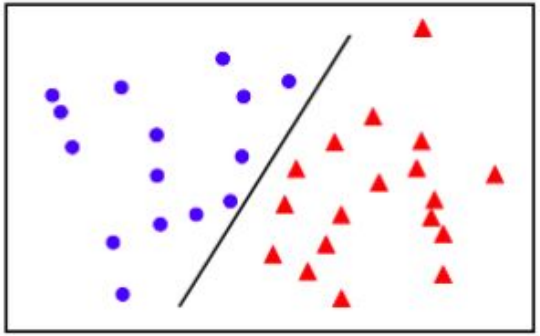
$$f(\mathbf{x}_i) \begin{cases} \geq 0 & y_i = +1 \\ < 0 & y_i = -1 \end{cases}$$

es decir: $y_i f(x_i) > 0$ para una clasificación correcta.

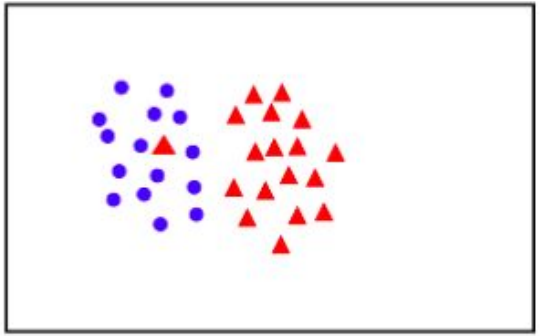
Separabilidad lineal

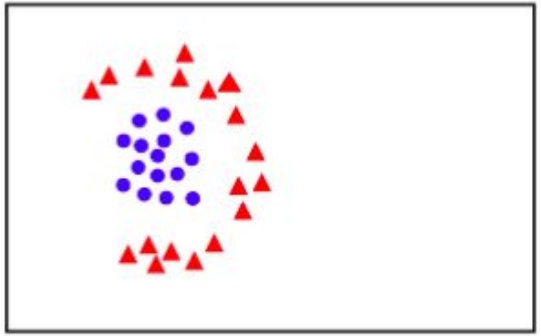
linealmente separable





no linealmente separable





Clasificadores lineales

- La entrada es un vector $\mathbf{x}_i$ de dimensionalidad n
- La salida es una etiqueta $y_i \epsilon \{-1, +1\}$
- Clasificador = función de predicción + función de decisión

$$g(f(\mathbf{x})) \rightarrow \{-1, +1\}$$

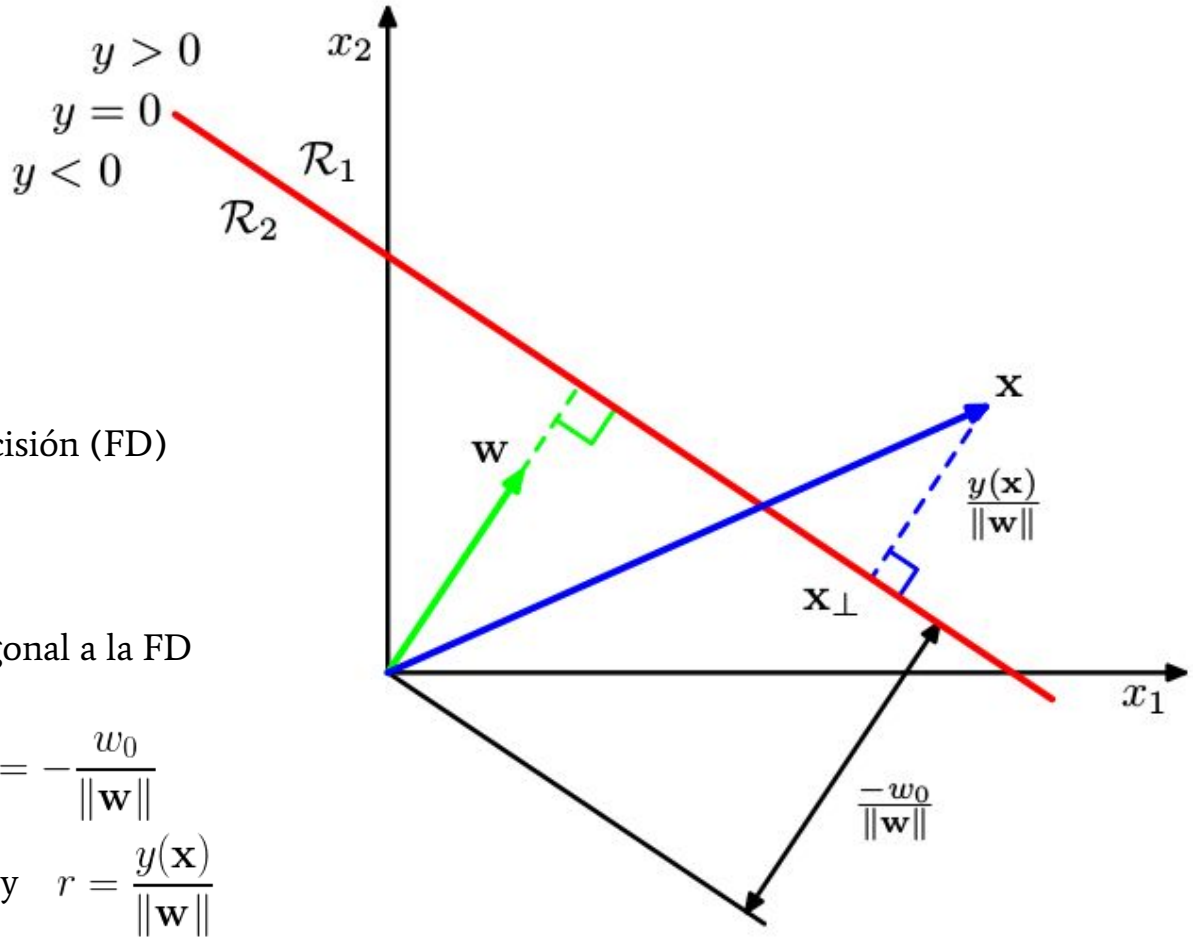
- Función de predicción **lineal**

$$f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0$$

- Función de decisión

$$g(z) = sign(z)$$

$$g(f(\mathbf{x})) = sign(\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0)$$



$$y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$$

$\mathbf{x}_A, \mathbf{x}_B$: puntos en la frontera de decisión (FD)

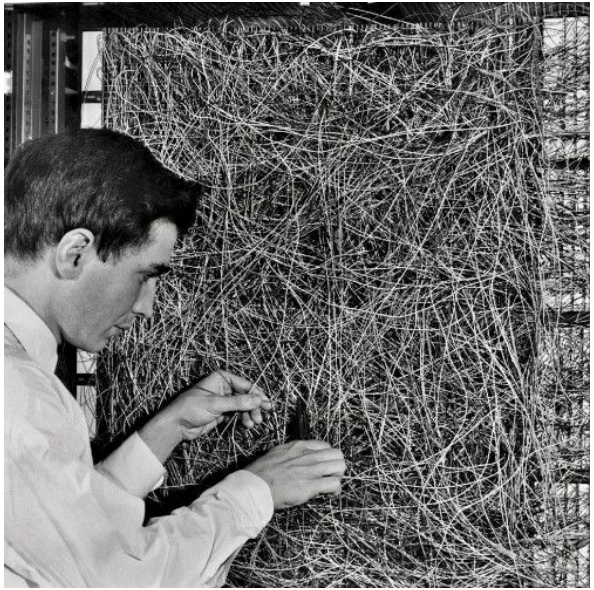
$$y(\mathbf{x}_A) = y(\mathbf{x}_B) = 0$$

$$\mathbf{w}^T(\mathbf{x}_A - \mathbf{x}_B) = 0 \implies \mathbf{w} \text{ es ortogonal a la FD}$$

si $\mathbf{x}$ en la FD, $y(\mathbf{x}) = 0 \rightarrow \dfrac{\mathbf{w}^T\mathbf{x}}{\|\mathbf{w}\|} = -\dfrac{w_0}{\|\mathbf{w}\|}$

si $\mathbf{x} = \mathbf{x}_\perp + r\dfrac{\mathbf{w}}{\|\mathbf{w}\|} \rightarrow y(\mathbf{x}_\perp) = 0$ y $r = \dfrac{y(\mathbf{x})}{\|\mathbf{w}\|}$

El algoritmo del "perceptrón"



- Propuesto por Rosemblatt en 1958
- El objetivo es encontrar un hiperplano de separación. **Si los datos son linealmente separables, lo encuentra.**
- Es un algoritmo *online* (procesa un ejemplo a la vez)
- Muchas variantes ...

El algoritmo del "perceptrón"

Entrada:

- una secuencia de pares de entrenamiento $(\mathbf{x}_1, y_1)$, $(\mathbf{x}_2, y_2)$ ...
- Una tasa de aprendizaje r (número pequeño y menor a 1)

Algoritmo:

- Inicializar $\mathbf{w}^{(0)} \epsilon \mathbb{R}^n$
- Para cada ejemplo $(\mathbf{x}_i, y_i)$
  - Predecir $y_i' = sign(\mathbf{w}^T \mathbf{x}_i)$
  - Si $y_i' \neq y_i$:

    $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + r\,(y_i\,\mathbf{x}_i)$

El algoritmo del "perceptrón"

Entrada:

- una secuencia de pares de entrenamiento $(\mathbf{x}_1, y_1)$, $(\mathbf{x}_2, y_2)$ ...
- Una tasa de aprendizaje r (número pequeño y menor a 1)

Algoritmo:

- Inicializar $\mathbf{w}^{(0)} \epsilon \mathbb{R}^n$
- Para cada ejemplo $(\mathbf{x}_i, y_i)$
  - Predecir $y_i' = sign(\mathbf{w}^T \mathbf{x}_i)$
  - Si $y_i' \neq y_i$:

    $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + r\,(y_i\,\mathbf{x}_i)$

Actualiza solo cuando comete un error

Error en positivos:

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + r\,\mathbf{x}_i$$

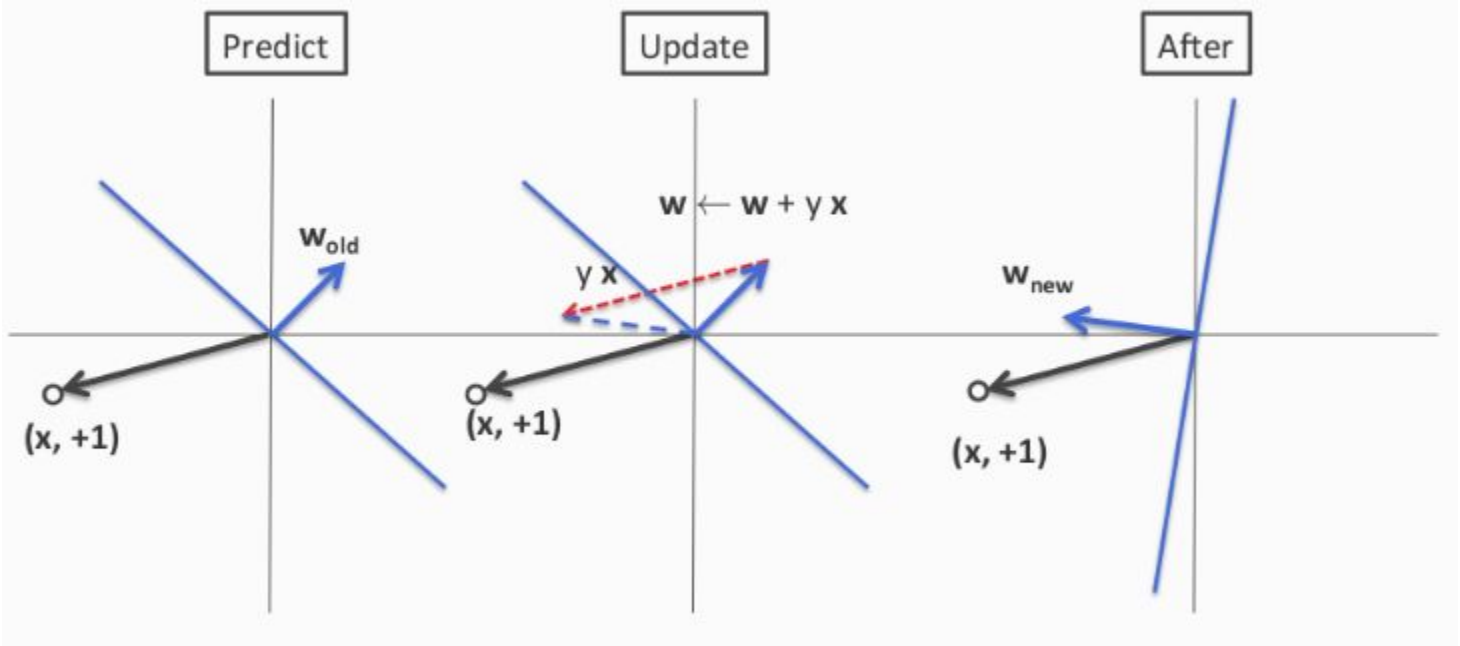
Error en negativos:

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - r\,\mathbf{x}_i$$

Si $y_i \mathbf{w}^T \mathbf{x}_i \leq 0 \rightarrow$ error

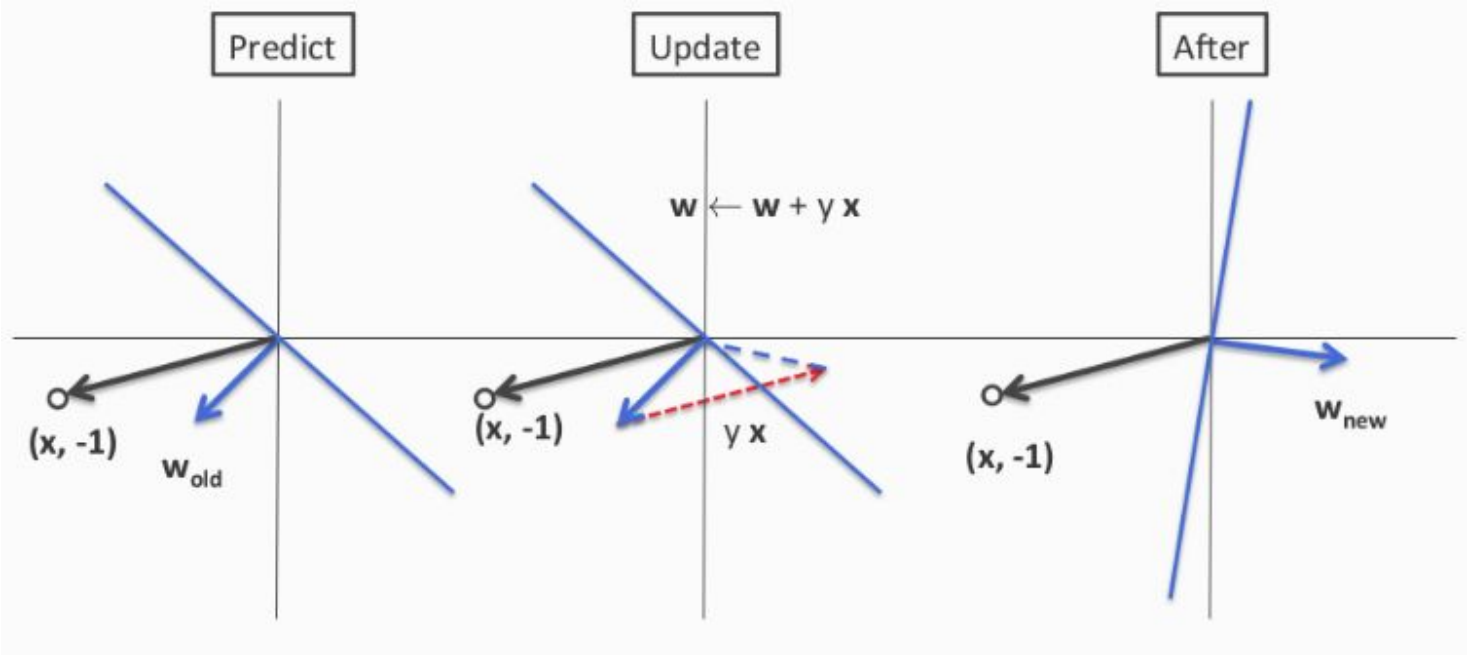
Dinámica de actualización

Error en ejemplo **positivo**:



Dinámica de actualización

Error en ejemplo **negativo**:



El algoritmo estándar

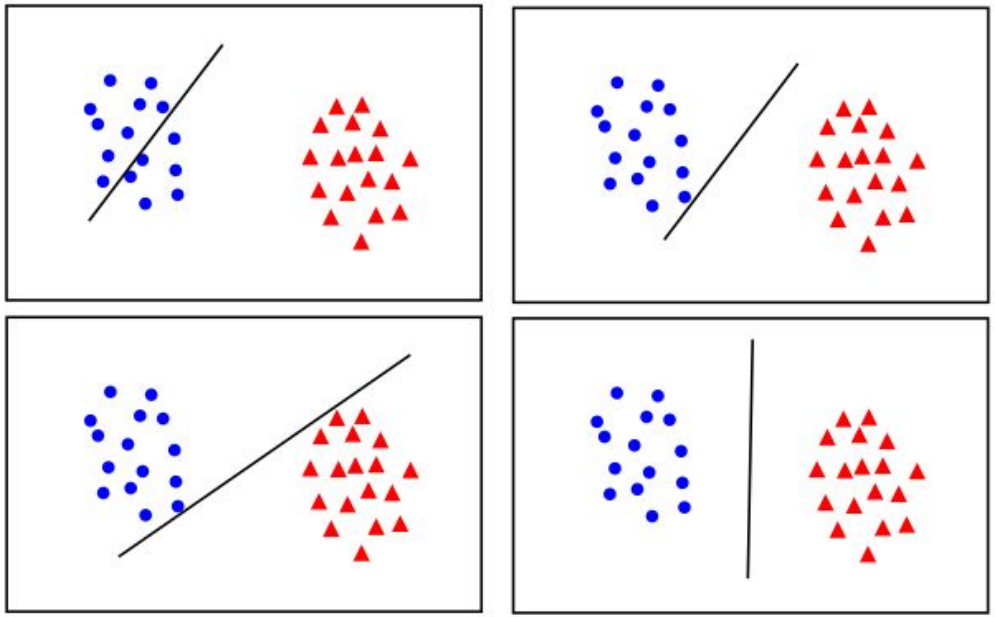
Dado un conjunto $\mathrm{D}=\{(\mathbf{x}_i, y_i)$, i=1, ..., N$\}$, $y_i \in \{-1, +1\}$, taza de entrenamiento r y número de épocas T

1. Inicializar $\mathbf{w}^{(0)}$
2. Para época t=1, …, T
   a. *barajar* el conjunto de entrenamiento D
   b. Para cada muestra de entrenamiento $(\mathbf{x}_i, y_i) \in \mathrm{D}$
      - si $y_i \mathbf{w}^{(t)\mathrm{T}} \mathbf{x}_i \leq 0$, actualizar $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + r\,(y_i\,\mathbf{x}_i)$
3. Retornar $\mathbf{w}^{(\mathrm{T})}$

r, T : hiperparámetros

Predicción: $sgn(\mathbf{w}^{\mathrm{T}}\mathbf{x})$

¿Cuál es el mejor w?



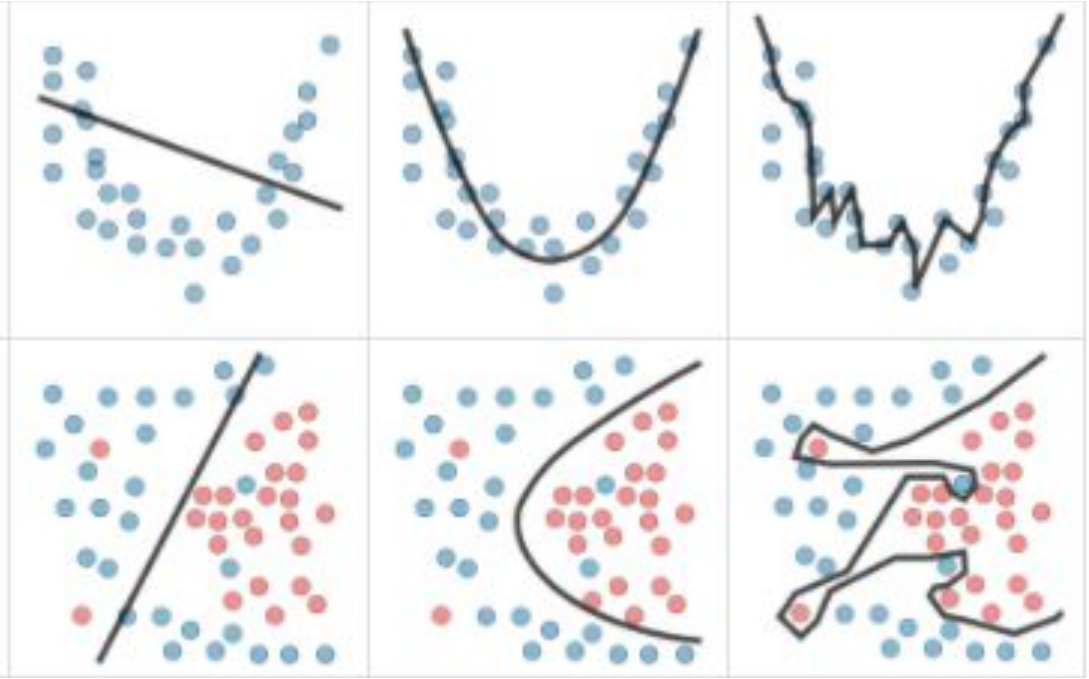
Solución de **margen máximo**: el hiperplano más estable ante perturbaciones de la entrada

Generalización en clasificación

- Complejidad del modelo ⇔ complejidad de la frontera de decisión

Regresión

Clasificación



Regresión logística

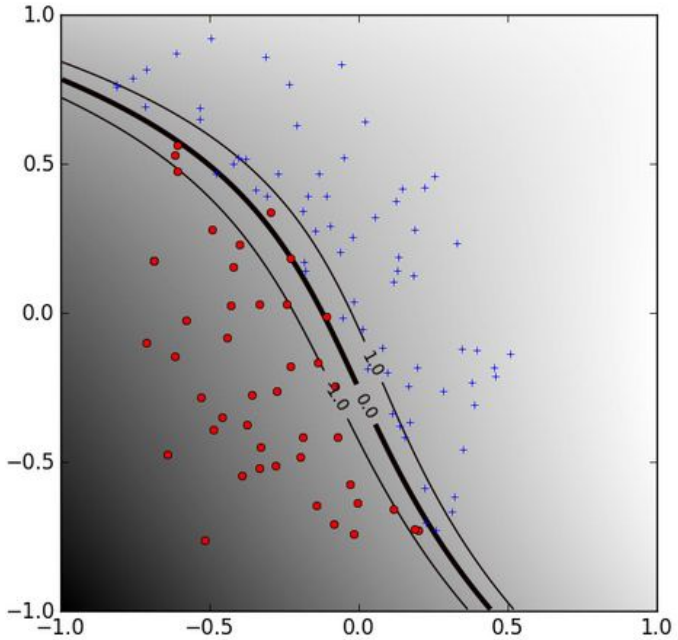
Clasificación basada en probabilidades

- Objetivo: dar una estimación de probabilidad de que una instancia x sea de una clase y, es decir, $\mathrm{p}(y|x)$

- Recordar:

$$0 \leq \mathrm{p}(evento) \leq 1$$

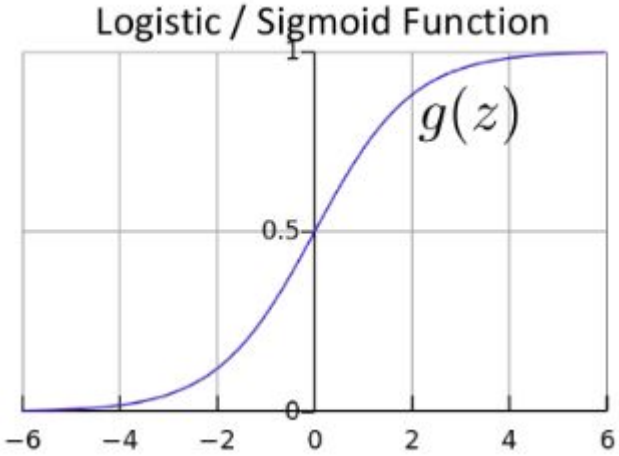
$$\mathrm{p}(evento) + \mathrm{p}(\neg evento) = 1$$



Regresión logística

- Aproximación probabilística al problema de clasificación
- La función de predicción $h_w(x)$ debe dar una aproximación de $p(y=1|x,w)$
- $0 \leq h_w(x) \leq 1$

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



Regresión logística

- Dados $\left\{ \left(\boldsymbol{x}^{(1)}, y^{(1)} \right), \left(\boldsymbol{x}^{(2)}, y^{(2)} \right), \ldots, \left(\boldsymbol{x}^{(n)}, y^{(n)} \right) \right\}$

  donde $\boldsymbol{x}^{(i)} \in \mathbb{R}^d, \; y^{(i)} \in \{0, 1\}$

- Modelo: $h_{\boldsymbol{\theta}}(\boldsymbol{x}) = g\left(\boldsymbol{\theta}^\intercal \boldsymbol{x}\right)$

$$g(z) = \frac{1}{1 + e^{-z}}$$

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} \qquad \boldsymbol{x}^\intercal = \begin{bmatrix} 1 & x_1 & \ldots & x_d \end{bmatrix}$$

Regresión logística. Función de costo

- Conjunto de entrenamiento $\{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\}, \quad \mathbf{x} \in R^M, \quad y \in \{0, 1\}$

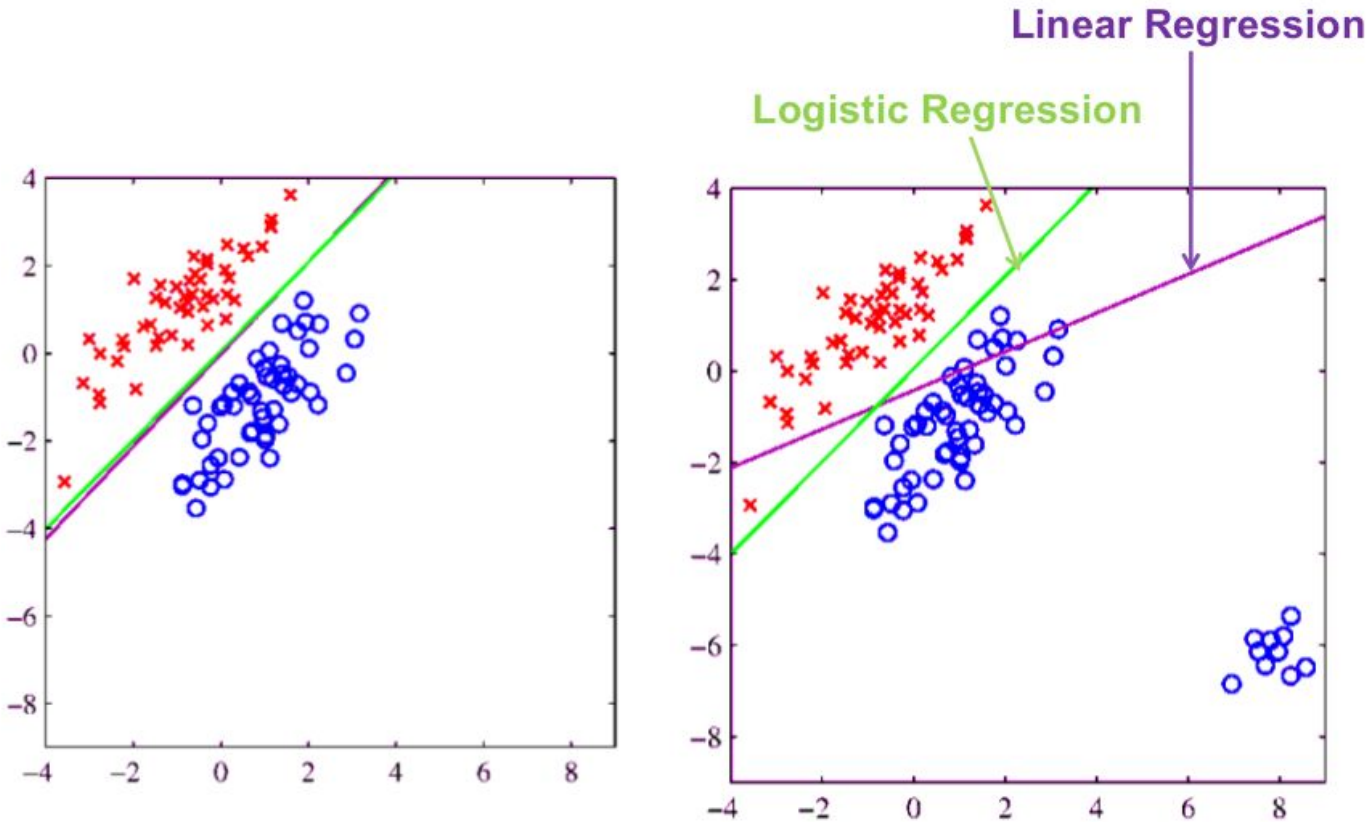
- y: observaciones discretas → muestras de una distribución Bernoulli

$$
\begin{aligned}
P(y = 1|\mathbf{x}, \mathbf{w}) &= f(\mathbf{x}, \mathbf{w}) \\
P(y = 0|\mathbf{x}, \mathbf{w}) &= 1 - f(\mathbf{x}, \mathbf{w}) \\
P(y|\mathbf{x}) &= (f(\mathbf{x}, \mathbf{w}))^y \, (1 - f(\mathbf{x}, \mathbf{w}))^{1-y}
\end{aligned}
$$

- Encontrar el $\boldsymbol{w}$ que maximice la verosimilitud de las etiquetas en el conjunto de entrenamiento

$$
\begin{aligned}
-L(\mathbf{w}) = C(\mathbf{w}) = \quad & \log P(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \sum_{i=1} \log P(y^i|\mathbf{x}^i, \mathbf{w}) \\
= \quad & \sum_i y^i \log f(\mathbf{x}^i, \mathbf{w}) + (1 - y^i) \log(1 - f(\mathbf{x}^i, \mathbf{w}))
\end{aligned}
$$

Regresión lineal vs. regresión logística



Problemas multiclase

Clasificación multiclase

- Una muestra puede pertenecer a 1 (o más) de K clases
  - Datos de entrenamiento $\{(\mathbf{x}_i, y_i)\}$, y_i=1,…, K

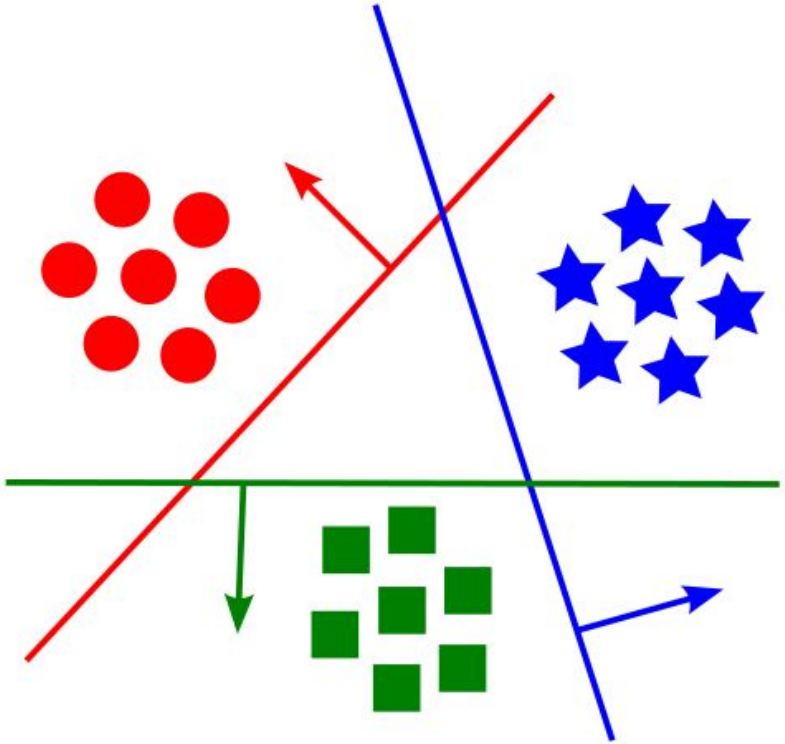
- Distintos tipos de problemas:
  - multiclase: $\mathbf{x}$ pertenece solo a una categoría
  - multietiqueta: $\mathbf{x}$ puede pertenecer a más de una categoría

- A veces es más fácil descomponer el problema multiclase en una serie de problemas binarios. **Distintas estrategias: OVA, AVA, ...**

Estrategia uno contra todos (OVA)

- **Asumimos que cada clase es separable del resto**
- Dado un conjunto de entrenamiento $D=\{(\mathbf{x}_i, y_i)\}$, $y_i = 1,...,K$
  - Descomponer el problema en K problemas binarios. Para la clase *k*, crear un problema tal que:
    - Ejemplos cuya etiqueta es $y_i = k$ son ejemplos positivos
    - Ejemplos cuya etiqueta es $y_i \neq k$ son ejemplos negativos
  - Generar K clasificadores binarios con **función de predicción** $f_k(\mathbf{x})$, k=1,...,K.
- Predicción (*winner takes all*): $k^* = \text{argmax}\, f_k(\mathbf{x})$

Estrategia uno contra todos (OVA)

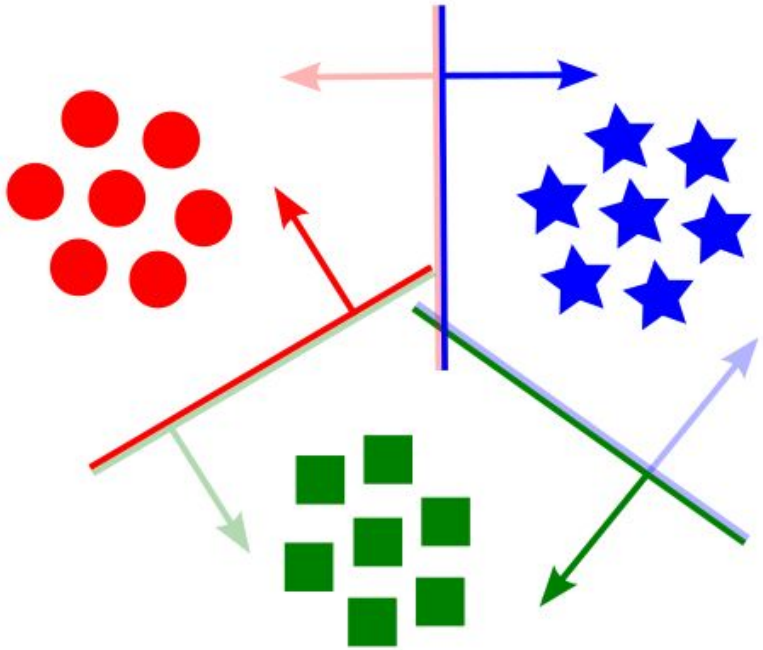


Estrategia todos contra todos (AVA)

- **Asumimos que cada clase par de clases es separable**
- Dado un conjunto de entrenamiento $D=\{(\mathbf{x}_i, y_i)\}$, $y_i = 1,...,K$
  - Descomponer el problema en K(K-1)/2 problemas binarios.

    Para el par de clases *(i, j), i≠j*, crear un problema tal que:
    - Ejemplos cuya etiqueta es $y_i = i$ son ejemplos positivos
    - Ejemplos cuya etiqueta es $y_i = j$ son ejemplos negativos
  - Generar K(K-1)/2 clasificadores binarios con **función de decisión** $g_{(i, j)}(\mathbf{x})$
- Predicción (*voting*): cada clase recibe K-1 “votos”

Estrategia todos contra todos (AVA)



Regresión logística multiclase

- Para dos clases:

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{1+\exp(-\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{x})} = \frac{\exp(\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{x})}{1+\exp(\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{x})}$$

peso asignado a y=0

peso asignado a y=1

- Para C clases (c=1,...,C):

$$p(y=c \mid \boldsymbol{x}; \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_C) = \frac{\exp(\boldsymbol{\theta}_c^{\mathsf{T}}\boldsymbol{x})}{\sum_{c=1}^{C}\exp(\boldsymbol{\theta}_c^{\mathsf{T}}\boldsymbol{x})}$$

(función **softmax**)

Métricas en clasificación

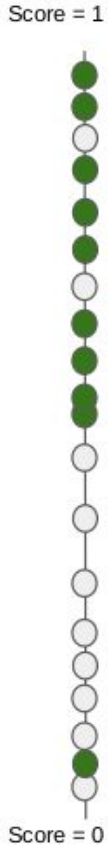
Importancia de las métricas

- La función de costo es solo un *proxy* al problema en el mundo real
- Las métricas ayudan a capturar objetivos reales en forma cuantitativa (no todos los errores son iguales)
- Ayudan a la organización el trabajo de los equipos en función de los requerimientos del problema
- Permiten cuantificar diferencias en:
  - performance deseada vs modelo base
  - performance deseada vs actual
  - evolución en el tiempo
- Deberían ser el objetivo del entrenamiento, pero a veces es difícil.

Clasificación binaria

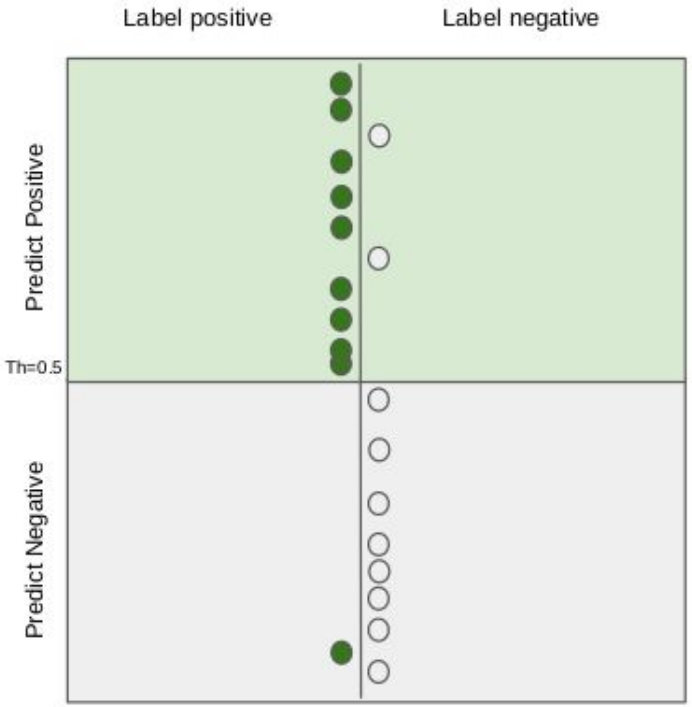
- Entrada: x, salida: y (valores 0/1 o -1/+1)
- Predicción del modelo: $\hat{y}=h(x)$
- Dos tipos de modelos:
  - Modelos que predicen directamente una variable categórica (kNN, árboles de decisión)
  - Modelos que predicen un puntaje (*score*) (SVM, regresión logística)
    - Se necesita elegir un umbral (func. de decisión)
    - Nos enfocaremos en esta última clase de modelos. Los anteriores se pueden ver como un caso especial.

Modelos basados en *scores*



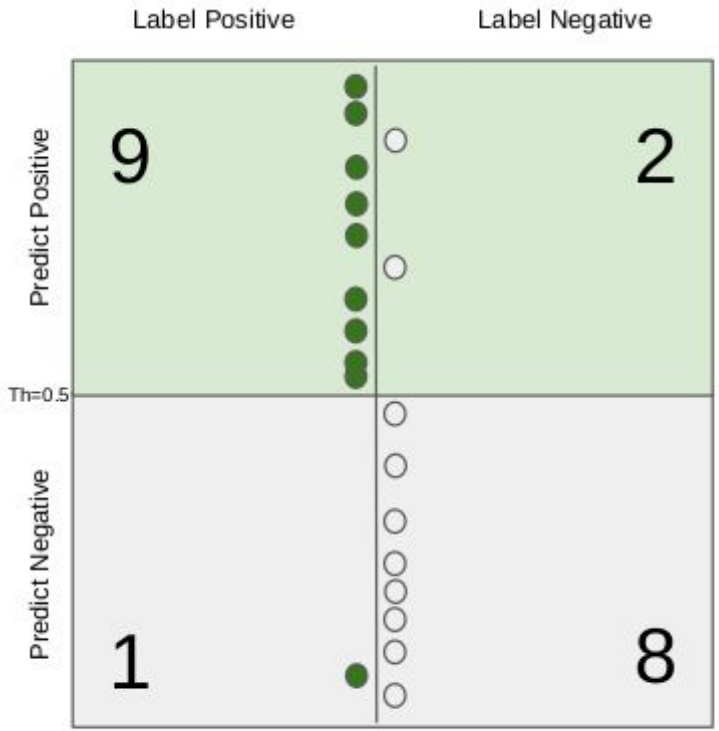
●	Positive example
○	Negative example

Umbral → clasificador → métrica puntual



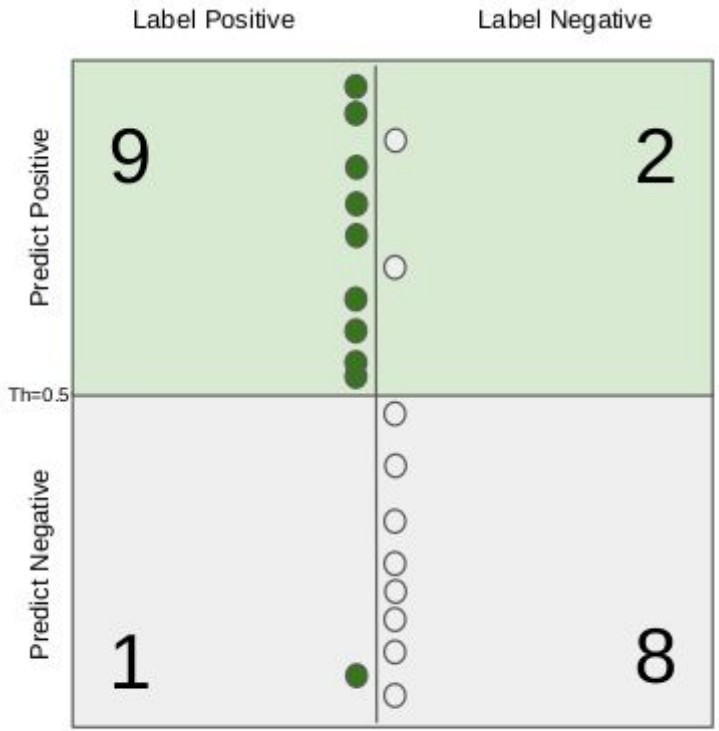
Th
0.5

Matriz de confusión



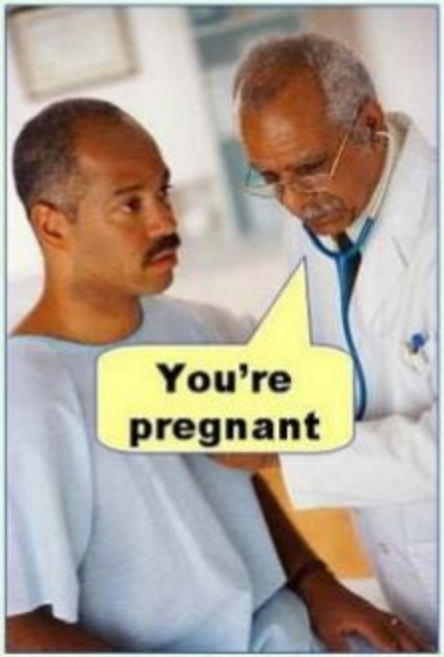
- la suma total es fija (muestra)
- la suma por columnas es fija (muestras por clase)
- la calidad del modelo y el valor de umbral deciden el agrupamiento de filas
- queremos que los elementos diagonales tengan valores grandes y los no diagonales valores chicos

Matriz de confusión

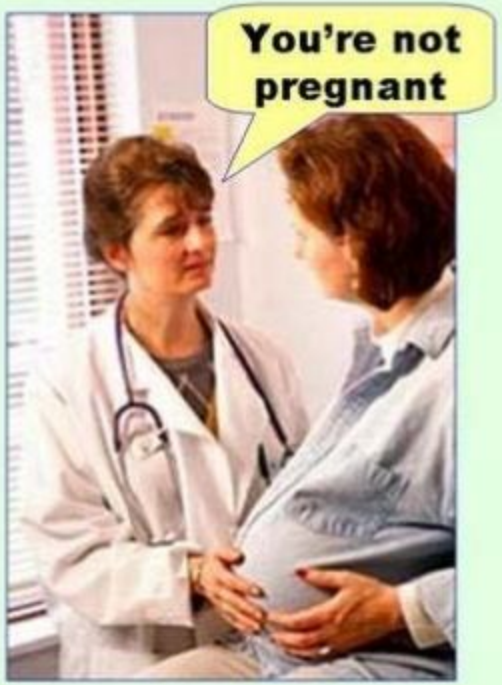


- *true positives* (TP) = 9
- *true negatives* (TN) = 8
- *false positives* (FP) = 2
- *false negatives* (FN) = 1

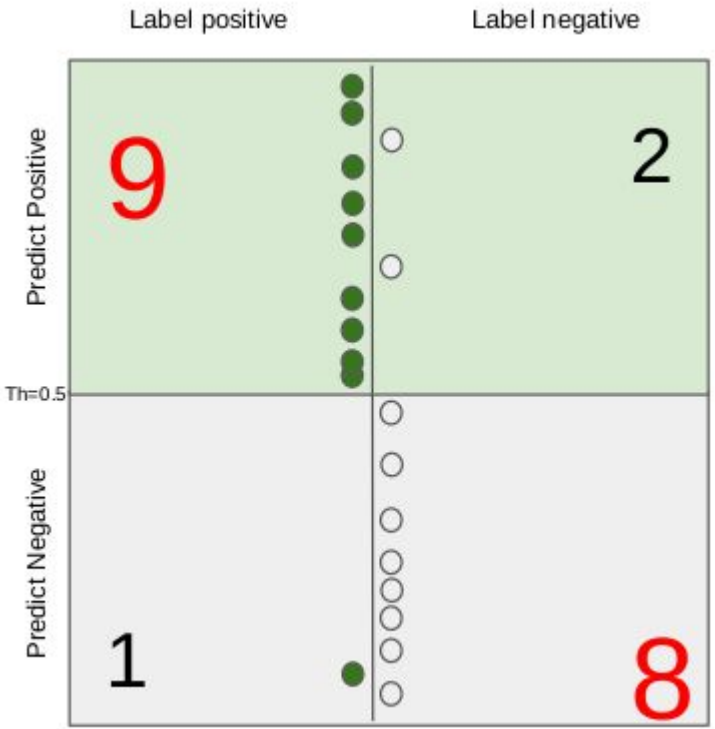
Type I error
(false positive)



Type II error
(false negative)



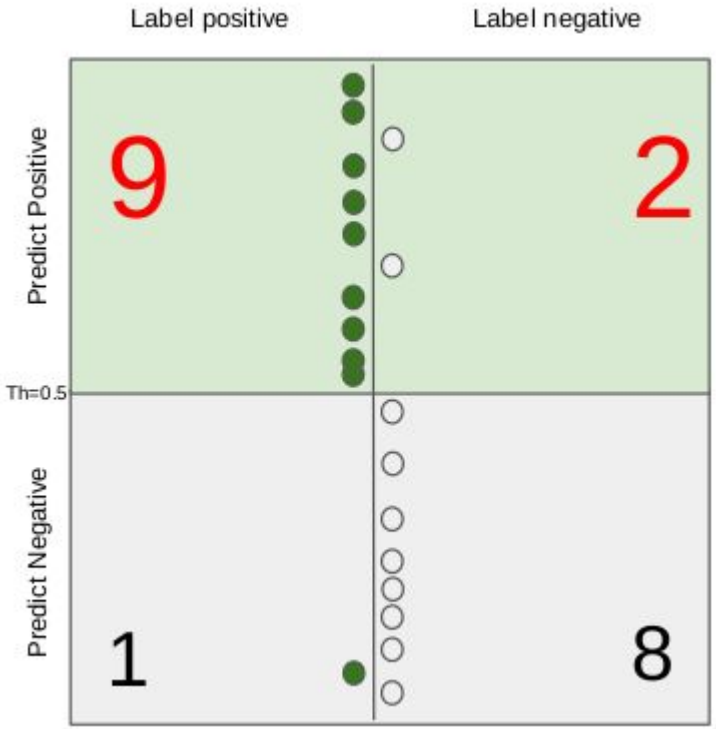
Métricas puntuales: exactitud (*accuracy*)



Th	TP	TN	FP	FN	Acc
0.5	9	8	2	1	.85

- acc=(TP+TN)/(TP+FP+TN+FN)
- equivalente al costo 0/1

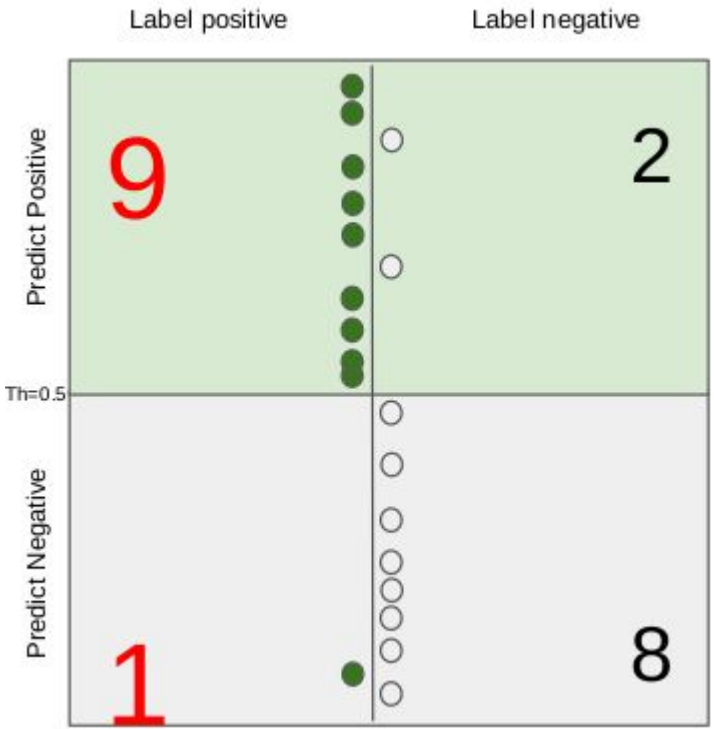
Métricas puntuales: precisión



Th	TP	TN	FP	FN	Acc	Pr
0.5	9	8	2	1	.85	.81

- Prec=TP/(TP+FP)
- Prec 100% = todos bajo el umbral salvo el de score más alto (siempre que sea correcto)

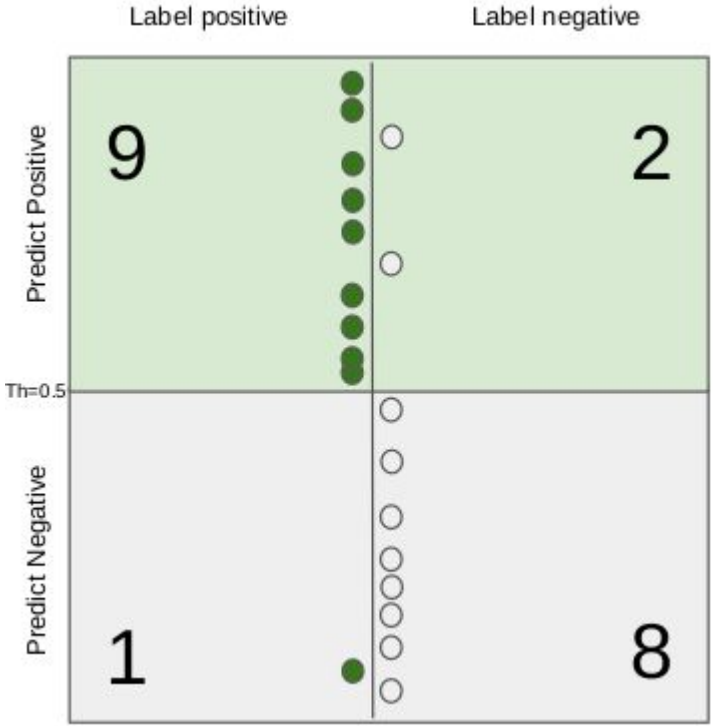
Métricas puntuales: sensitividad (*recall*)



Th	TP	TN	FP	FN	Acc	Pr	Recall
0.5	9	8	2	1	.85	.81	.9

- Recall=TP/(TP+FN)
- Recall 100% = todos los puntos por encima del umbral

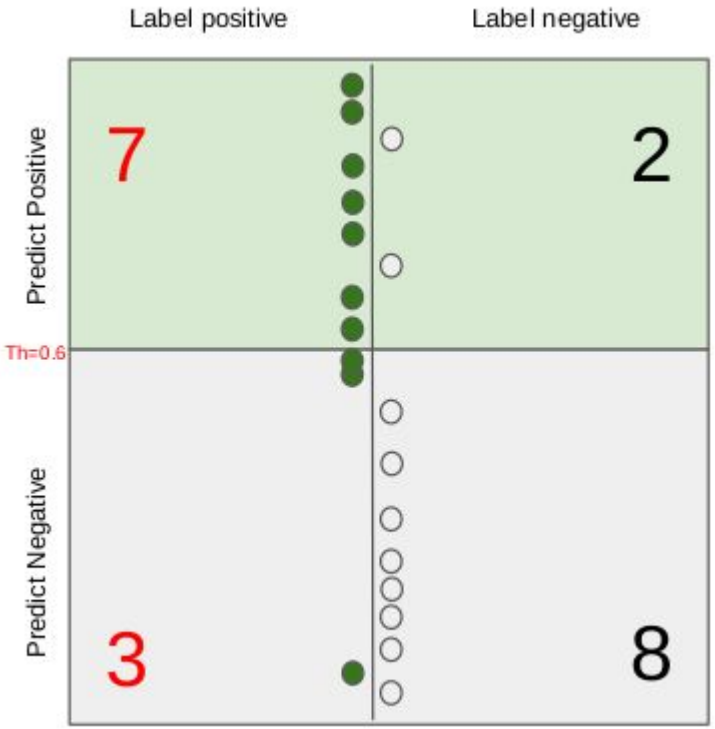
Métricas puntuales: F1-score



Th	TP	TN	FP	FN	Acc	Pr	Recall	F1
0.5	9	8	2	1	.85	.81	.9	.857

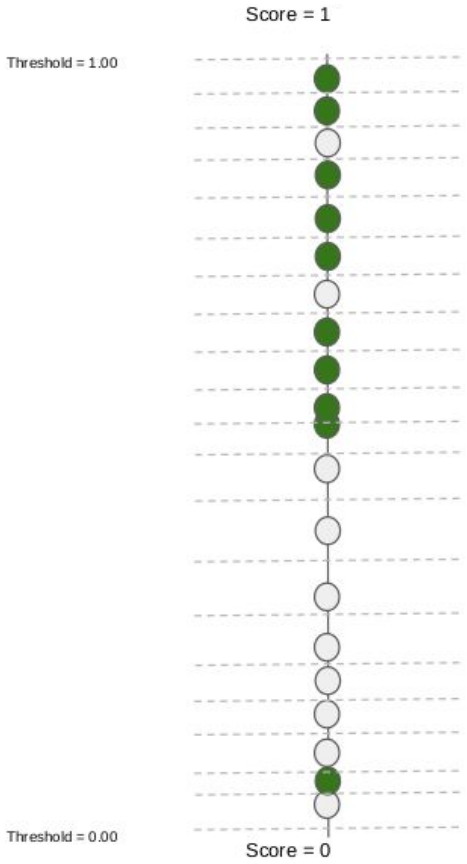
$$F_1 = \left(\frac{2}{\mathrm{recall}^{-1} + \mathrm{precision}^{-1}} \right) = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

Métricas puntuales: cambio del umbral



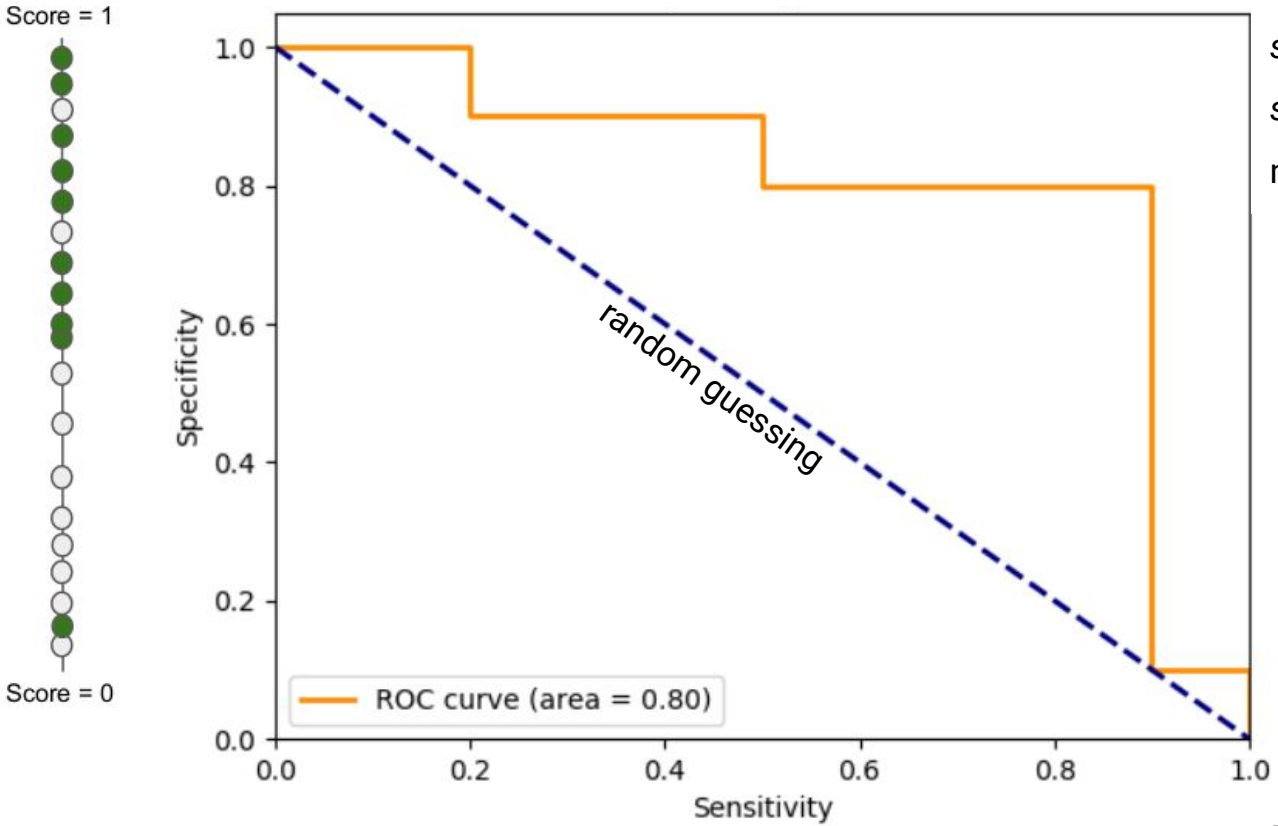
Th	TP	TN	FP	FN	Acc	Pr	Recall	F1
0.6	7	8	2	3	.75	.77	.7	.733

umbrales efectivos = # ejemplos + 1



Threshold	TP	TN	FP	FN	Accuracy	Precision	Recall	Specificity	F1
1.00	0	10	0	10	0.50	1	0	1	0
0.95	1	10	0	9	0.55	1	0.1	1	0.182
0.90	2	10	0	8	0.60	1	0.2	1	0.333
0.85	2	9	1	8	0.55	0.667	0.2	0.9	0.308
0.80	3	9	1	7	0.60	0.750	0.3	0.9	0.429
0.75	4	9	1	6	0.65	0.800	0.4	0.9	0.533
0.70	5	9	1	5	0.70	0.833	0.5	0.9	0.625
0.65	5	8	2	5	0.65	0.714	0.5	0.8	0.588
0.60	6	8	2	4	0.70	0.750	0.6	0.8	0.667
0.55	7	8	2	3	0.75	0.778	0.7	0.8	0.737
0.50	8	8	2	2	0.80	0.800	0.8	0.8	0.800
0.45	9	8	2	1	0.85	0.818	0.9	0.8	0.857
0.40	9	7	3	1	0.80	0.750	0.9	0.7	0.818
0.35	9	6	4	1	0.75	0.692	0.9	0.6	0.783
0.30	9	5	5	1	0.70	0.643	0.9	0.5	0.750
0.25	9	4	6	1	0.65	0.600	0.9	0.4	0.720
0.20	9	3	7	1	0.60	0.562	0.9	0.3	0.692
0.15	9	2	8	1	0.55	0.529	0.9	0.2	0.667
0.10	9	1	9	1	0.50	0.500	0.9	0.1	0.643
0.05	10	1	9	0	0.55	0.526	1	0.1	0.690
0.00	10	0	10	0	0.50	0.500	1	0	0.667

Métricas resumen: curvas ROC (rotada)



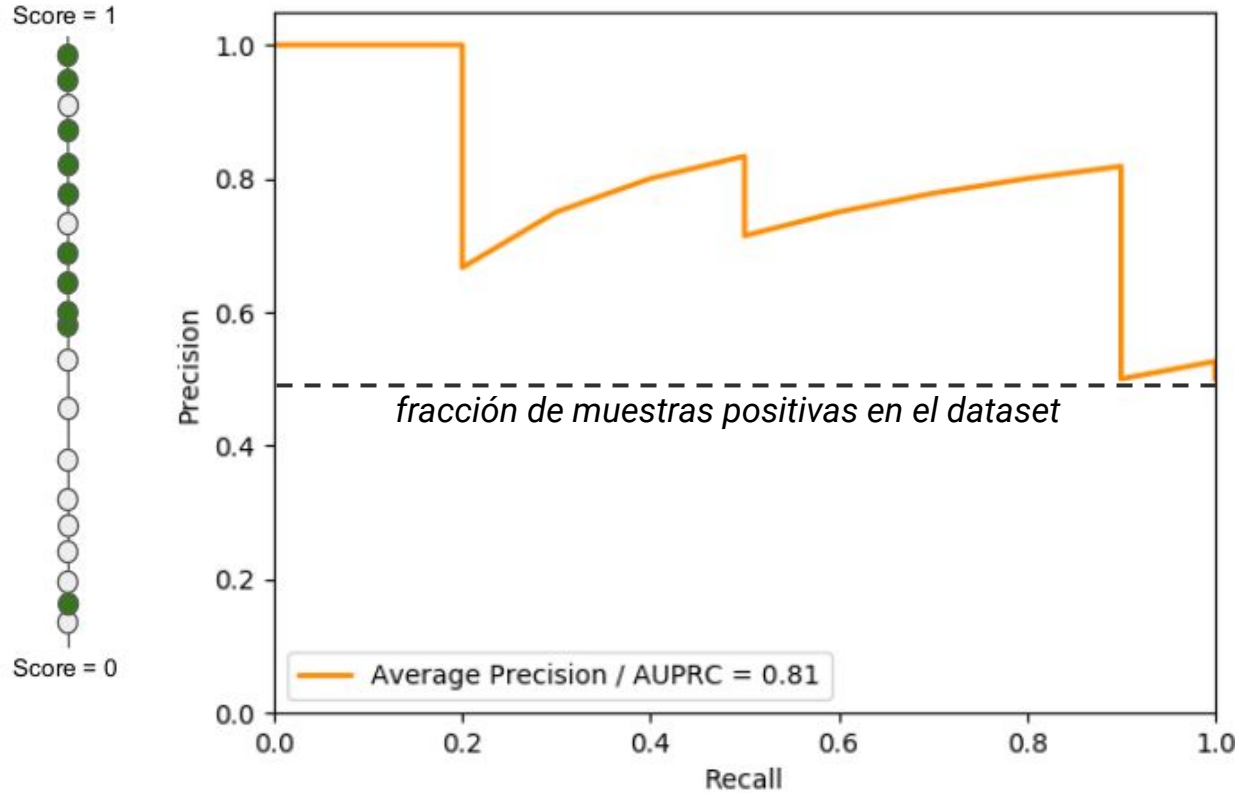
specificity=tnr=TN/Neg=TN/(TN+FP)

sensitivity=tpr=TP/Pos=TP/(TP+FN)

métrica AUC=área bajo la curva ROC

ROC=Receiver Operating Characteristic

Métricas resumen: curvas PR



precision=TP/(pos predichos)

recall=TP/(pos verdaderos)

AUPR=área bajo la curva PR

PR=Precision-Recall

Curvas ROC y PR en validación cruzada

Opción 1:

- Asumir que magnitudes de los *scores* son comparables entre corridas
- Acumular predicciones de todas las corridas
- Trazar la curva usando predicciones acumuladas

Opción 2:

- Trazar las curvas individuales para cada partición
- Considerar la “curva promedio”

Resumen curvas ROC y PR

- Permiten evaluación cuantitativa a distintos niveles de “confianza”
- Asumen problemas binarios
- Se pueden resumir en medidas del tipo “área bajo la curva”
- Las curvas ROC son insensibles a cambios en la distribución de clases en el conjunto de test
- Las curvas PR muestran la fracción de las predicciones que son FP
- Las curvas PR son útiles en problemas con una proporción de muestras negativas muy alta
- Permiten determinar umbrales óptimos para distintos puntos de operación

Problemas multiclase

- Problema con N clases => matriz de confusión de NxN
- La mayoría de las métricas se analizan como N problemas binarios (OVA)
  - El desbalance crece con el número de clases
- Variantes multiclase de métricas AUC
  - micro vs. macro average