AA - TEMA 1: INTRODUCCIÓN

PARADIGMA

+ Aprender a partir de {datos de problemas, respuestas} → responder preguntas "parecidas".
+ Para ello → Obtener parte "general" aplicable siempre (inducción).
+ Dos fases { - Entrenamiento: aprender, a partir de datos → responder.
  - Test: probar si lo aprendido resuelve casos nuevos.
+ Dos tipos de problemas { - Regresión (predecir valores reales). Eligiendo 1.
  - Clasificación (predecir categorías) ⇄ Probabilística (%).

PRINCIPALES MÉTODOS DE APRENDIZAJE

+ Machine Learning { + Objetivo: Algoritmos ⇄ Precisos (↓Error en casos nuevos). ⇄ Eficientes → Para problemas a gran escala.
  - Puede generar malos modelos.
+ Aprendizaje Estadístico: aprender infiriendo con distrib. de probabilidad.
  + Resultados buenos si cumplen hipótesis.
  - Incapaz de resolver problemas a gran escala.
+ Aprendizaje Bayesiano: aproxima estadísticamente con conoc. previo.
  + Resultados buenos si la hipótesis se cumple.
  - Muy complejo y costoso.

APRENDIZAJE VS DISEÑO ---> información del problema?

+ Diseño: Uso de datos para fijar parámetros de un algoritmo de un problema bien especificado (tenemos mucha info. y conocemos funcionamiento).
  * Ejemplo: Calcular media/desviación de un modelo de prob. definido.
+ Aprendizaje: No conocemos (el) cómo resolver el problema, por lo que metemos los datos en un algoritmo de aprendizaje que nos ~~(ajuste)~~ de una hipótesis para clasificar/predecir las respuestas.

MACHINE LEARNING

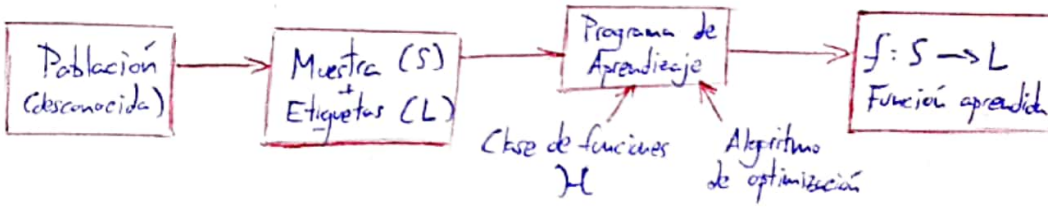
+ **Def.** Un programa aprende de la experiencia E con respecto a una clase de tareas T y medida de desempeño P_T, si:
  ↑↑ E → P_T también aumenta.
  * Cuantos más datos tengamos, mayor experiencia.

PARADIGMAS DE MACHINE LEARNING

+ Aprendizaje supervisado: Aprender una función con { - muestra de datos (entrada)
  - etiquetas. (salidas)
+ Aprendizaje no supervisado: Aprender propiedades de datos sin etiquetar, transformando las características observadas. (↓dimensionalidad, = info).
  * Auto-supervisado: Transforma datos observados en otros (más útiles) resolviendo "tareas de pretexto" (que es supervisada).
+ Conocimiento de transferencia: cómo compartir conocimiento entre tareas.
+ Aprendizaje por refuerzo: Compuesto por datos y recompensas. Se obtiene una política de actuación, que trata de maximizar las recompensas.

APRENDIZAJE SUPERVISADO

+ Asume { - Única información disponible → muestra.
  - Obtenemos, a través de un Criterio de Aprendizaje → f.
  - f calcula una predicción (opcn.) según su entrada.
+ Objetivo: elegir {muestra, programa} → f que prediga correctamente toda la población
* Problema: ¿qué datos son relevantes para el problema? (p.e. árbol ≤ SI/NO)
  → No lo sabemos, pero existe un patrón (ya que nosotros lo vemos).
  Por tanto, trataremos de aprenderlo a partir de los datos.
  → Si elegimos buena representación, las técnicas funcionarán.



APRENDIZAJE NO SUPERVISADO

+ Objetivo: transformar muestra a un espacio de representación, donde se introduce una nueva medida de distancia entre muestras.
  → Encontrar patrones. → facilitar posterior procesamiento.
+ Tipos
  - Clustering: usa la estructura geométrica.
  - Descubrimiento de dependencias → patrones.
  - Reducción de dimensionalidad → quitar atributos no relevantes.
  - Auto-supervisado.
+ Problema: encontrar características no relevantes → ↓↓ aprendizaje.

PROCESO DE APRENDIZAJE SUPERVISADO

+ Recibimos X, Y y D como punto de partida.

1. Elegimos $\mathcal{H}$ con los tipos de funciones que probablemente represente a f. $\mathcal{H}$ = "Conjunto de hipótesis".
2. Extraemos una muestra de D. Cada elemento de la muestra irá marcado con X e Y.
3. Elegimos, a través de un algoritmo de aprendizaje A y la muestra, una hipótesis $g \in \mathcal{H}$.
4. Comprobamos si $g \approx f$.

FORMALIZACIÓN DEL APRENDIZAJE SUPERVISADO

DATOS INICIALES

+ Obtendremos los datos de $P(D)$ { P es la probabilidad de extraer cada elemento → estática inmutable ($x \in D$); D es la población }
  * La muestra debe extraerse i.i.d (independiente, idénticamente distribuida).
+ X: características medidas en cada muestra.

FUNCIÓN OBJETIVO: $f: X \to Y$ (muestras)

+ f es la función a aproximar, que es desconocida.

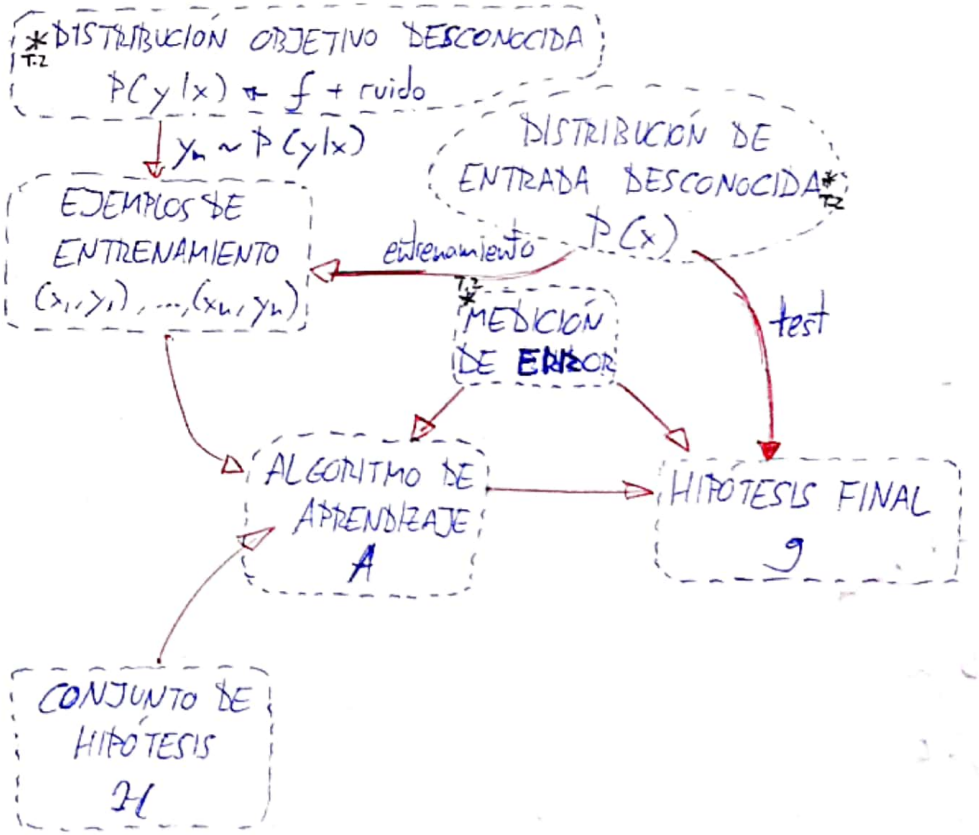
MODELO

+ $\mathcal{H}$: clase de funciones que usaremos para aproximar f.
+ Eligiremos $h \in \mathcal{H}$ que sea buena prediciendo la población.
  * Para encontrarla → Algoritmo de Aprendizaje.

ALGORITMO DE APRENDIZAJE

+ ¿Qué criterio optimizar para garantizar el aprendizaje? (función) { ERM, SRM, MDL, Similarity-Measure }
+ ¿Qué algoritmo usar para optimizar el criterio? A
  * Gradiente Descendente, SGD, Pseudo-inversa ...

(T.2) MODELO COMPLETO DE APRENDIZAJE (COMPLETO)



A - TEMA 2 : MODELOS LINEALES

+ $\mathcal{H}$ = clase de las f lineales de X.

$h(x) = \sum_{i=1}^{d} x_i w_i + \underbrace{x_0 \cdot w_0}_{\text{bias}} \;//\; h(x) = \vec{w}^T \vec{x}$

$(h \in \mathcal{H})$

$h, f : X \to Y$
$X = \{1\} \times \mathbb{R}^d$
$Y = \mathbb{R}$

+ Aplicable a todo, salvo a árboles.
+ Si un modelo lineal resuelve un problema → mejor modelo.
  * Nos da la influencia de cada característica.
+ Generalmente → Estrategia ganadora probar primero lineal.

REGRESIÓN

(de $y = f(x)$)

+ $h(x) = w^T x$ será la predicción de la regresión.
+ Usaremos el error cuadrático para medir la pérdida de $h_w(x)$ con f.
  * $Error(x) = (h_w(x) - f(x))^2$.
  * $E_{in}(h_w) = \frac{1}{N} \sum_{i=1}^{N} (h_w(x_i) - y_i)^2$, (error dentro de la muestra).
+ Eligiremos h_w tal que $\downarrow\downarrow\downarrow E_{in}$, aunque el objetivo es $\downarrow\downarrow E_{out}$.
+ $E_{out}(h) = \mathbb{E}_{(x,y)\sim P}[(h(x) - y)^2]$. (esperanza/media al elegir un elem. de forma aleatoria de la población).

$\hat{h} = \min_{h \in \mathcal{H}} E_{out}(h)$

+ Para encontrar $\hat{h}$ → ERM (Minimizar Riesgo Empírico).

$E_{in}(w) = \text{media}[(w^T x_i - y_i)^2] \;//\; E_{in}(w) = \frac{1}{N} \|Xw - y\|^2$

error cuadrático

$w_{lin} = \min_{w \in \mathbb{R}^{d+1}} E_{in}(w)$

$X = \begin{pmatrix} - x_0 \to \\ \vdots \\ - x_n \to \end{pmatrix}$; $y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$

$w = \begin{pmatrix} w_i \\ \vdots \\ b \end{pmatrix}$

+ ¿Cuándo funciona ERM?
  → Modelo: y_i (etiquetas) $= f(x_i, \beta) + \underbrace{\text{ruido}}_{\epsilon_i}$ → Contemplamos ruido en y (no en X).
  → Estimador de los pesos w.
  * T.ª Gauss-Markov: El método de los mínimos cuadrados ordinarios alcanza $\hat{\beta}$ → estimador insesgado ($\mathbb{E}(\hat{\beta}_i) = \beta_i$) de varianza mínima (óptimo).
  * Condiciones $\begin{cases} - E(\epsilon_i) = 0 \text{ (Media ruido=0)}. \\ - E(\epsilon_i \epsilon_j) = 0 \text{ (Ruido incorrelado)}. \end{cases}$ $- \text{Var}(\epsilon_i^2) = \sigma^2 < \infty$. (Var. acotada)

PSEUDO-INVERSA : ECUACIONES NORMALES DE E_{in}

$E_{in}(w) = \frac{1}{N} \|Xw - y\|^2$ $\quad \nabla E_{in}(w) = 0$ en puntos críticos $\begin{cases} - \text{mínimo} \\ - \text{máximo} \\ - \text{p. silla} \end{cases}$

$\nabla E_{in}(w) = \frac{2}{N} X^T (Xw - y) \overset{!}{=} 0$

$X^T X w_{lin} = X^T y \to w_{lin} = \underbrace{(X^T X)^{-1} X^T}_{X^\dagger = \text{"pseudo-inversa de } X\text{"}} y$

* ¿Cómo hacer $(X^T X)^{-1}$? Descomposición en Valores Singulares.

$X = U D V^T$ → U, V^T son ortogonales ($U^{-1} = U^T$)
→ D es diagonal ($D = D^T$).

$(U^{-1})U = I$

$X^T X = (UDV^T)^T UDV^T = VD^T U^T \cdot UDV^T = VDDV^T$

$(X^T X)^{-1} = (VDDV^T)^{-1} = V^{-T} D^{-1} D^{-1} V^{-1} = V D^{-2} V^T$

$w_{lin} = (X^T X)^{-1} X^T y = V D^{-2} V^T \cdot VDU^T y = V D^{-1} U^T y$. ¡ siempre es calculable

* Si $(X^T X)$ tiene rango d → 1 solución
* Si $(X^T X)$ tiene rango $< d$ → ∞ soluciones (improbable).

PROPIEDADES DE REGRESIÓN

+ $\hat{y} = X w_{lin} = \underbrace{X (X^T X^{-1}) X^T}_{\hat{H}} y = \hat{H} y$
  * $\hat{H}$ es la "matriz de proyección". $\hat{H} : y \to \hat{y}$.
  * Es relevante en el estudio de E_{in} / E_{out} (si las respuestas son "y", crea pred. "$\hat{y}$").
  * Es idempotente ($\hat{H}^2 = \hat{H}$).
  * Su traza (suma elementos diagonal) $= d + 1$.
+ Generalización del error: existen fórmulas para calcular E_{out}:

$E_{out} = E_{in}(w_{lin}) + O\left(\frac{d}{N}\right)$ (d: Dimensión muestras (nº características). N: nº muestras)

+ Dado w_0, podemos encontrar $\hat{v}$: $E_{in}(w_0 + \eta \hat{v}) < E_{in}(w_0)$.
  * Obtenemos $\hat{v}$ con expansión de Taylor de grado 1.º (Aproxima función ΔE_{in}).

$\Delta E_{in} = E_{in}(w(0) + \eta \hat{v}) - E_{in}(w(0)) = \eta \nabla E_{in}(w_0)^T \hat{v} + O(\eta^2)$

$\eta \nabla E_{in}(w_0)^T \hat{v} = \eta \|\nabla E_{in}(w_0)\| \|\hat{v}\| \cos\theta \geq -\eta \|\nabla E_{in}(w_0)\|$ (Máximo que puede bajar)

* Si tomamos $\hat{v} = -\frac{\nabla E_{in}(w_0)}{\|\nabla E_{in}(w_0)\|}$ (vector unitario en dir. contraria a ∇) → $\Delta E_{in} = \downarrow$

GRADIENTE DESCENDENTE

+ Técnica de optimización iterativa general → Mínimo local, siguiendo la dirección del vector de gradiente en cada punto.
+ Algoritmo: Comienza con w_0. De manera iterativa:
  * $w_j = w_j - \eta \frac{\partial E_{in}(w)}{w_j}$, $\forall j \in [0,n]$, actualizando todos a la vez.
    - η: Tasa de aprendizaje
    - $\frac{\partial E_{in}(w)}{w_j}$: Componente j del vector gradiente.
  * $\frac{\partial E_{in}(w_0)}{\partial w_j} = \frac{\partial}{\partial w_j} \frac{1}{N} \sum_{n=1}^{N} (w^T x_n - y_n)^2 = \frac{2}{N} \sum_{n=1}^{N} (x_{nj} (w^T x_n - y_n))$

~~* $w_{i+1} = w_i - \eta \cdot \left(\frac{2}{N} \sum_{i=1}^{N} (X^T (w^T x_i - y_i))^T \right)$~~

  * Cada punto (x_n, y_n) contribuye a la actualización proporcionalm. a su error de predicción.
+ ¿Qué valor asignar a η?
  * Si es muy pequeño, tarda en converger.
  * Si es muy grande, puede no converger y fluctuar permanentem.
  * Ideal: empezar con valores grandes, e ir descendiendo para converger.

GRADIENTE DESCENDIENTE ESTOCÁSTICO

+ En cada actualización de w, se utilizan solo un subconjunto de ejemplos (minibatch de $M << N$ elementos).
+ Ventajas
  * ↓↓ variabilidad en estimación del gradiente.
  * Muy rápido computacionalmente.
  * En funciones no convexas → Obtiene buen mínimo local.
+ Algoritmo: Comienza con un w_0. De manera iterativa:
  1. Reordenamos la muestra y la dividimos en minibatches.
  2. Para cada minibatch:
     for $(j=0 \rightarrow k)$ t_minibatch
     $w_j = w_j - \eta \sum_{i=0} x_{ij} (h(x_i) - y_i)$.

MÉTODO DE NEWTON

+ Realiza la expansión de Taylor de segundo nivel:

$$g(\Delta w) = E_{in}(w_0 + \Delta w) \approx E_{in}(w_0) + \Delta w^T \nabla E_{in}(w_0) + \frac{1}{2} \Delta w^T \nabla^2 E_{in}(w_0) \Delta w$$

$$\frac{\partial g(\Delta w)}{\partial \Delta w} = 0 \rightarrow \nabla E_{in}(w_0) + \underbrace{\nabla^2 E_{in}(w_0)}_{H \text{ (hessiana)}} \Delta w = 0$$

$$\Delta w = -H^{-1} \nabla E_{in}(w_0)$$

↳ Define el avance en dirección del gradiente.

- Es costoso calcular H^{-1}.

ANÁLISIS DE RESULTADOS

+ Análisis de residuos (función $r = y - h$). → detectar muestras fuera de la hipótesis de los mínimos cuadrados ordinarios.
  * Gráficos
    - Errores correlativos → Si los residuos muestran una tendencia.
      → Dependencias no controladas
      → Extracción de muestra no cumple i.i.d (p.e. series temporales)
    - Errores con varianza no constante:
      → Solución: transformar etiquetas, o dar pesos a observaciones.
    - Dependencia entre predictores
      → Multicolinealidad: 1 característica ⇔ combinación lineal.
      Problema → Inversión de matriz inestable, coef. imprecisos.
      → ↓↓ generalización.
    - Valores atípicos / aislados
+ En dimensiones muy altas → Escenario complejo → Hipótesis original pierde relevancia.

* Características categóricas → vector "one-hot" (un elemento = 1, resto = 0)
  → Modelos no lineales, como árboles, manejan categorías.
* Pérdida de valores → Faltan en la muestra de entrenamiento.
  → Solución: valores sustitutos ↗ interpolados de histograma.
  ↘ predecidos usando algún modelo.
  → Los árboles pueden también gestionar esto.

AA - TEMA 2: MODELOS LINEALES (II)

CLASIFICACIÓN

PERCEPTRÓN - DOS CLASES (aka. Perceptrón)

+ $\mathcal{H}$: hiperplanos separadores, con dimensión = nº características.
  * Si $h(x) \geq b \rightarrow x \in$ Clase 1
  * Si $h(x) < b \rightarrow x \in$ Clase 2

  $\Rightarrow h(x) = \text{signo}((\sum_{i=1}^{d} x_i w_i) + b)$, $h(x) \in \{-1, 1\}$

  * O, de forma equivalente: $\mathcal{H} = \{h \mid h(x) = \text{signo}(w^T x)\}$, con w cte.
    - $w^T = (b, w_1, w_2, \ldots, w_n)$
    - $x^T = (1, x_1, \ldots, x_n)$

+ Criterio de aprendizaje (ERM): $\min_{w \in \mathbb{R}^{d+1}} \frac{1}{N} \sum_{n=1}^{N} [\text{signo}(w^T x_n) \neq y_n]$.
  * Problema: no es derivable.

+ Solución: usar una función parecida y derivable:

$$\text{error}(w^T x_n, y_n) = \begin{cases} -y_n w^T x_n & \text{si signo}(w^T x_n) \neq y_n \text{ (falla)} \\ 0 & \text{si signo}(w^T x_n) = y_n \text{ (acierto)} \end{cases}$$

$$\nabla \text{error} = \frac{\partial \max(0, -y_n w^T x_n)}{\partial w} = \begin{cases} -y_n x_n & \text{si fallamos} \\ 0 & \text{si acertamos} \end{cases}$$

+ Algoritmo Perceptrón (PLA).
  * Equivale a SGD con tam_minibatch = 1 y $\eta = 1$.
  * Regla de actualización:
    - Si falla: $w_{i+1} = w_i + y \cdot x$
    - Si acierta: $w_{i+1} = w_i$

    Hasta que clasifique todo bien → FIN
  * Si el conjunto es linealmente separable → solución que clasifica todos los ejemplos bien en t finito. Si no, no converge.
  * Interpretación geométrica: $y \cdot x = \|y\| \|x\| \cos(\theta)$
    1. Si $y = 1$, y $w \cdot x < 0 \rightarrow \theta > 90° \rightarrow (w + y \cdot x) \cdot x > 0$
    2. Si $y = -1$, y $w \cdot x > 0 \rightarrow \theta < 90° \rightarrow (w + y \cdot x) \cdot x < 0$

1) 

$\theta > 90°$ ✗
$\theta' < 90°$ ✓

2) 

$\theta < 90°$ ✗
$\theta > 90°$ ✓

MODELO DE CLASIFICACIÓN LINEAL

+ $X = \{1\} \times \mathbb{R}^d$; $Y = \{-1, 1\}$, $f: X \rightarrow Y$
+ $\mathcal{H} = \{h \mid h(x) = \text{signo}(w^T x)\}$
+ De acuerdo a ERM y la teoría de aprendizaje:

$$E_{out}(g) \leq E_{in}(g) + O\left(\sqrt{\frac{\log N}{N} \cdot d}\right) \rightarrow \text{Con } N \text{ suficientemente grande, } \boxed{E_{out} \approx E_{in}}$$

→ Reducción de E_{in} en datos linealmente separables.

+ Resultado PLA: dada una muestra lin. sep., PLA encuentra una separación tras $(R\beta)^2$ iteraciones (como máximo).
  * $R = \max_i \|x_i\|$
  * $\beta = \min \{\|w\| : \forall i \in [m], y_i w^T x_i \geq 1, w \in \mathbb{R}^d\}$.

  Por tanto, la convergencia depende de β (que puede ser exponencialm. grande).
+ Sin embargo, no tenemos por qué llegar tan lejos (la población puede ser no separable, aunque la muestra lo sea).

→ Reducción de E_{in} en datos lin. no separables.

+ Caso 1: Errores en f lin. sep.
  * Si queremos encontrar solución → ↑↑ E_{in}.

$$E_{in} = \min_{w \in \mathbb{R}^{d+1}} \frac{1}{N} \sum_{i=1}^{N} [\text{signo}(w^T x_i) \neq y_i] \rightarrow \text{Problema NP-Duro.}$$

  * Alternativa: PLA Pocket
    - En cada iteración, calcula E_{in}. Si es más bajo que el mejor E_{in} hasta ahora → Guarda solución.
    - ↓↓ eficiencia/iteración, pero acota nº iteraciones.
+ Caso 2: transformaciones no lineales → Hacer que sean sep.

REGRESIÓN LINEAL PARA CLASIFICACIÓN

+ Regresión lineal aprende una $f(x) \in \mathbb{R}$.
+ Podemos calcular w: $w^T x_n \approx y_n = \pm 1$.
+ Así, $\text{sign}(w^T x_n) \rightarrow y_n = \pm 1$.

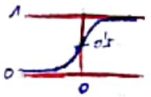
MUESTRAS CON RUIDO

+ Pueden haber muestras con $y_n \neq f(x_n)$ ← por falta de info. / ← f es no determinista" (influye punto vista).
+ En vez de una función, buscaremos distribución $P(y|x)$.
+ Cada muestra → $P(x,y) = P(x)\,P(y|x)$
  Info. de tal población ← % de x e y (a la vez) | % x (importancia de x) | % y, conociendo x (que es la entrada). Es lo que queremos conocer.
  → $P(x,y) = P(y)\,P(x|y)$
  % y (etiquetas + probables) | % x, conociendo su clase y.
+ Objetivo con ruido = Objetivo determinista + ruido
  $f(x) = E(y|x)$ $\quad y - f(x)$
+ Objetivo determinista = Objetivo con ruido: $P(y|x) = 0$, excepto en $y = f(x)$.
+ ¿$P(y|x)$?
  - Aprenderlo directamente (Discriminativa): si consideramos muestra i.i.d: $P(x,y) = P(y|x)\,P(x) \Rightarrow$ cte.
  - Aprenderlo a través de $P(x|y)$ con el T^a de Bayes (Generativa):
    $P(y|x) = \frac{P(x|y)\cdot P(y)}{P(x) \to \text{cte.}} \propto P(x|y)\cdot P(y)$ (proporcional)

CLASIFICADORES PROBABILÍSTICOS

REGRESIÓN LOGÍSTICA (LGR)

+ $h(x) = \sigma(w^T x) \in [0,1]$ // $h(x) = \tanh(w^T x) \in [-1,1]$
  → Función logística (sigmoide): $\sigma(x) = \frac{e^x}{e^x + 1}$.
+ Podemos considerar la salida ≈ probabilidad de clasificación.
+ Nos da flexibilidad al asignar muestras a las etiquetas.
+ Caso binario
  * Probabilidad C_1: $h(x) = \sigma(w^T x)$
  * Probabilidad C_2: $1 - h(x) = \sigma(-w^T x)$



+ Ratio "odds" = $\frac{h(x)}{1-h(x)} \in (0,+\infty)$ ($h(x)$: P.C$_1$, $1-h(x)$: P.C$_2$) → Función logit = $\ln(\text{odds}) = \ln(e^{w^T x}) = w^T x \in (-\infty,\infty)$

CRITERIO DE APRENDIZAJE DE MÁXIMA VEROSIMILITUD

+ ~~Máxima~~ Trataremos de maximizar la probabilidad de la muestra:
  $L(S) = \prod_{i=1}^{N} \mathbb{P}_w(y_i|x_i)$ { - Si clasificamos bien → ≈1 ; - Si clasificamos mal → ≈0.
+ $\mathbb{P}_w(y|x) = \begin{cases} f(x) & \text{para } y=1 \\ 1-f(x) & \text{para } y=-1 \end{cases} = \begin{cases} h_w(x) & \text{para } y=1 \\ 1-h_w(x) & \text{para } y=-1 \end{cases} = \sigma(y w^T x)$ (Nos da el signo)
  $h_w(x) = \sigma(w^T x) = 1 - \sigma(-w^T x)$
+ Por tanto, maximizaremos $L(w) = \prod_{i=1}^{N} \mathbb{P}_w(y_i|x_i) = \prod_{i=1}^{N} \sigma(y w^T x)$.
+ Equivalentemente, podemos calcular E_{in} y minimizarlo:
  * $E_{in}(w) \overset{①}{=} -\frac{1}{N}\ln(L(w)) = \frac{1}{N}\sum_{i=0}^{N} \ln\left(\frac{1}{\mathbb{P}(y_i|x_i)}\right) = \frac{1}{N}\sum_{i=0}^{N} \ln\left(1 + e^{-y_i w^T x_i}\right)$ ① Véase "Entropía Cruzada"
  * $\nabla_w E_{in}(w) = \frac{1}{N}\sum_{i=0}^{N} -y_i x_i\, \sigma(-y_i w^T x_i)$.

CLASIFICACIÓN MULTIETIQUETA

+ Podemos generalizar regresión logística → k etiquetas.
  * En vez de $\mathbb{P}(x=1) = p$ y $\mathbb{P}(x=0) = q$, tal que $p+q=1$.
  * $\mathbb{P}(x_1,\dots,x_N) = \mathbb{P}(x_1)\cdot\dots\cdot\mathbb{P}(x_n)$ ≠
  * $L(x_1,\dots,x_N) = \prod_{i=1}^{N} \sigma(w^T x_i)^{[x_i==1]}\, \sigma(-w^T x_i)^{[x_i==0]} = \prod_{i=1}^{N}\prod_{k=0}^{K} \sigma(w_k x_i)^{y_{ik}}$ (Itera sobre clases)

  (Reescritura de clasif. binaria)
+ Ahora, y = 1-of-k vector (1 elemento es 1, resto 0).
+ La verosimilitud se define ahora usando N clases. (asumimos i.i.d).
  * $P(y|w_1,\dots,w_K,x) = \prod_{k=1}^{K} \mathbb{P}(y_k|w_k,x) = \prod_{k=1}^{K} \sigma(w_k^T x)^{y_k}$
    Estimadores de cada clase | Etiqueta a 1 en la muestra | Clase=1 en y | Cant. muestra | Prob. en el modelo de que la muestra encontrada exista
  * Para cada muestra → N: $L(y|w_{1\dots k}) = \prod_{n=1}^{N}\prod_{k=1}^{K} (\sigma(w_k^T x_n))^{y_{nk}}$ → Función a maximizar. (con k w's, será k veces más divertido!)

AA – TEMA 2: MODELOS LINEALES (III)

CLASIFICACIÓN MULTIETIQUETA (CONTINUACIÓN)

+ SOFTMAX: algoritmo que crea distribución de probabilidad a partir de los nºs resultantes de los distintos $w_k^T x$.

$$P(y_j = 1 \mid x) = \frac{e^{w_j x}}{\sum_{i=1}^{k} e^{w_i x}}, \text{ para cada clase } j.$$

* $\sum P(y_j \mid x) = 1$.

+ Regla de Bayes: Dada una distribución P_{x} binaria, la mejor función de predicción será:

$$h_p(x) = \begin{cases} 1 & \text{si } \mathbb{P}[y=1 \mid x] \geq 0'5 \\ 0 & \text{en otro caso.} \end{cases}$$

* Por tanto, siempre asignaremos a una muestra x la clase + probable.

COSTES EN EL ERROR

+ No es lo mismo fallar en unos casos u otros.
+ Dos casos
  - Falso positivo: "Dejar a un intruso entrar en zona máxima seguridad".
  - Falso negativo: "No dejar al general entrar en cuartel".

* Dependiendo de qué tipo de error nos sea más tolerable, ajustamos:

$$f_{COSTE} = c_{-1} \cdot P(error_{-1}) + c_1 \cdot P(error_1)$$

MÉTRICAS DE EVALUACIÓN

MATRIZ DE CONFUSIÓN

Predicción \ Muestra	Pos.	Neg.
✓	True Positive	False Positive
X	False Negative	True Negative
	P=TP+FN	N=FP+TN

+ Accuracy: % aciertos $= \frac{TP+TN}{P+N}$
+ Sensibilidad (o recall) $= \frac{TP}{P}$
+ Especificidad (o fallout) $= \frac{TN}{N}$
+ Precisión $= \frac{TP}{TP+FP}$

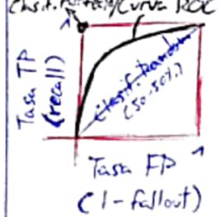
(aunque ↓pérdida → ↓métrica)

* Nos interesa reducir la métrica, no la pérdida!

CURVA ROC

+ Representa si un clasificador es mejor que otro.

Curva ROC ("Receiver Operator Characteristic")



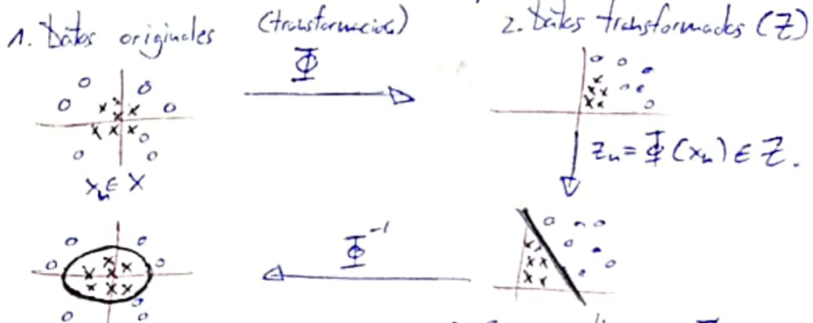
+ Cada punto de la curva equivale a un clasificador con cierto umbral.
+ Conforme ajustamos el umbral, ↑↑ tasa TP, pero también ↑↑ FP.

+ Área bajo la curva (AUC) = % de que el clasificador posicione TP sobre FP. ($\in [0,1]$).

AUC ≃ 0'5 → Clasif. Random. AUC ≃ 1 → Perfecto.

TRANSFORMACIONES NO LINEALES

+ Un modelo lineal solo tiene que tener w lineal.
  * Resto características → cualquier función.
+ Generalmente, añadir características no lineales mejora ajuste.
  * Recomendables cuando el residuo no es lineal.
  * Esto tiene su coste → Necesitaremos más muestras. (↑ dimensionalidad → ↑N).
* El uso de transformaciones no lineales no cambian $\mathcal{H}$, sino que se transforman características a espacio donde son lin. sep.



4. Obtenemos clasif. en X.

$$g(x) = \text{sign}(\tilde{w}^T \Phi(x))$$

($\tilde{w}$: Pesos en Z; $\Phi(x)$: características en Z.)

- Problema: debemos fijar la transformación sin ver los datos (por razones teóricas). ¿Qué Φ tomamos?

+ Función Riesgo: medida de la diferencia < predicciones / valores reales } respecto a P

↳ $L_p(h) \overset{def}{=} \mathbb{E}_{z \sim p}[\ell(h, z)] \overset{def}{=} E_{out}$ (ℓ: f de pérdida; $\mathbb{E}$: Esperanza (= media))

* Aplicable a cualquier modelo de aprendizaje.

+ Sea Φ_Q el polinomio de orden Q (y su transformación a Z).
  * ↑↑ Q → ↑↑ flexibilidad del modelo.
  * ↑↑ Q → ↑↑ $\tilde{d} = O(Q^d)$ → ↑↑ coste computacional.
  * ↑↑ Q → ↑↑↑ N (nº muestras) para llegar al mismo nivel de error de generalización $\left(O\left(\frac{\tilde{d}}{N} \log(N)\right)\right)$.

+ Por tanto → elegir Q que
  - separe bien los ejemplos dentro (↓E_{in}).
  - sea lo más pequeño posible (↓E_{out}).

AA - TEMA 2: MODELOS LINEALES (III)

CLASIFICACIÓN MULTIETIQUETA (CONTINUACIÓN)

+ SOFTMAX: algoritmo que crea distribución de probabilidad a partir de los nºs resultantes de los distintos $w_k^T x$.

$$P(y_j = 1 | x) = \frac{e^{w_j x}}{\sum_{i=1}^{k} e^{w_i x}}, \text{ para cada clase } j.$$

$* \sum P(y_j | x) = 1.$

+ Regla de Bayes: Dada una distribución $P_{x,y}$ binaria, la mejor función de predicción será:

$$h_p(x) = \begin{cases} 1 & \text{si } \mathbb{P}[y=1|x] \geq 0'5 \\ 0 & \text{en otro caso.} \end{cases}$$

* Por tanto, siempre asignaremos a una muestra x la clase + probable.

COSTES EN EL ERROR

+ No es lo mismo fallar en unos casos u otros.
+ Dos casos
  - Falso positivo: "Dejar a un intruso entrar en zona máxima seguridad".
  - Falso negativo: "No dejar al general entrar en cuartel".

* Dependiendo de qué tipo de error nos sea más tolerable, ajustamos:

$$f_{COSTE} = c_{-1} \cdot P(error_{-1}) + c_1 \cdot P(error_1)$$

MÉTRICAS DE EVALUACIÓN

MATRIZ DE CONFUSIÓN

Predicción \ Muestra	Pos.	Neg.
✓	True Positive	False Positive
X	False Negative	True Negative
	P=TP+FN	N=FP+TN

+ Accuracy: % aciertos $= \frac{TP+TN}{P+N}$
+ Sensibilidad (o recall) $= \frac{TP}{P}$
+ Especificidad (o fallout) $= \frac{TN}{N}$
+ Precisión $= \frac{TP}{TP+FP}$

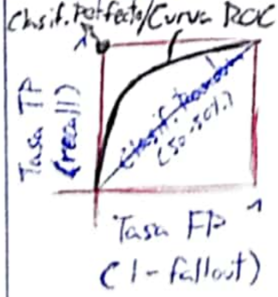
(aunque ↓ pérdida → ↓ métrica)

* Nos interesa reducir la métrica, no la pérdida!

CURVA ROC

+ Representa si un clasificador es mejor que otro.

Curva ROC ("Receiver Operator Characteristic").

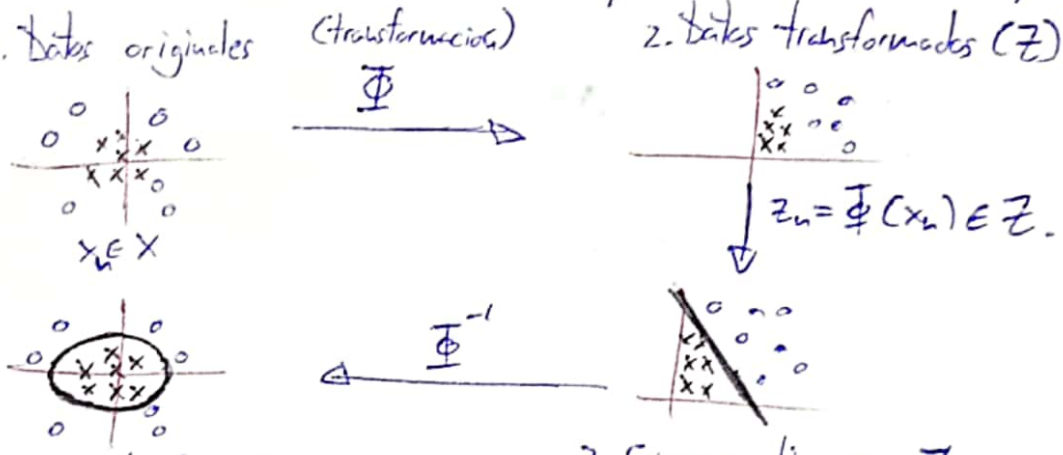


+ Cada punto de la curva equivale a un clasificador con cierto umbral.
+ Conforme ajustamos el umbral, ↑↑ tasa TP, pero también ↑↑ FP.

+ Área bajo la curva (AUC) = % de que el clasificador posicione TP sobre FP. ($\in [0,1]$).
AUC ≃ 0'5 → Clasif. Random. AUC ≃ 1 → Perfecto.

TRANSFORMACIONES NO LINEALES

+ Un modelo lineal solo tiene que tener w lineal.
  * Resto características → cualquier función.
+ Generalmente, añadir características no lineales mejora ajuste.
  * Recomendables cuando el residuo no es lineal.
  * Esto tiene su coste → Necesitaremos más muestras. (↑ dimensionalidad → ↑ N).
* El uso de transformaciones no lineales no cambian $\mathcal{H}$, sino que se transforman características a espacio donde son lin. sep.



$$g(x) = \text{signo}(\tilde{w}^T \Phi(x))$$

$\tilde{w}$: Pesos en Z — $\Phi(x)$: Características en Z.

- Problema: debemos fijar la transformación sin ver los datos (por razones teóricas). ¿Qué Φ tomamos?

+ Función Riesgo: medida de la diferencia ⇔ predicciones ⇔ valores reales } respecto a P
  ↳ $L_P(h) \overset{def}{=} \mathbb{E}_{z \sim P}[\underbrace{\ell(h,z)}_{\text{f de pérdida}}] \overset{def}{=} E_{out}$
  ! → Esperanza (= media)
  * Aplicable a cualquier modelo de aprendizaje.
+ Sea Φ_Q el polinomio de orden Q (y su transformación a Z).
  * ↑↑ Q → ↑↑ flexibilidad del modelo.
  * ↑↑ Q → ↑↑ $d = O(Q^3)$ → ↑↑ coste computacional.
  * ↑↑ Q → ↑↑↑ N (n^o muestras) para llegar al mismo nivel de error de generalización ($O(\frac{d}{N}\log(N))$).
+ Por tanto → elegir Q que separe bien los ejemplos dentro (↓ E_{in}).
  |- sea lo más pequeño posible (↓ E_{out}).