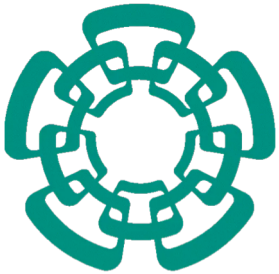
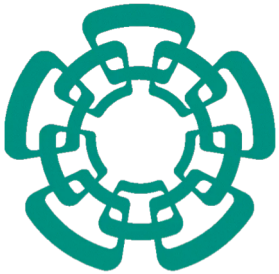


Introducción al Aprendizaje Computacional

Hugo Jair Escalante

hugo.jair@cs.cinvestav.mx

https://ccc.inaoep.mx/~hugojair/



Aprendizaje computacional supervisado

Hugo Jair Escalante

hugo.jair@cs.cinvestav.mx

https://ccc.inaoep.mx/~hugojair/

Preliminares

Aprendizaje computacional

- Subcampo de la IA que se enfoca en el desarrollo de programas de computadora que:
  - Adapten su comportamiento automáticamente a partir de datos
  - Sean capaces de aprender sin ser programados explícitamente

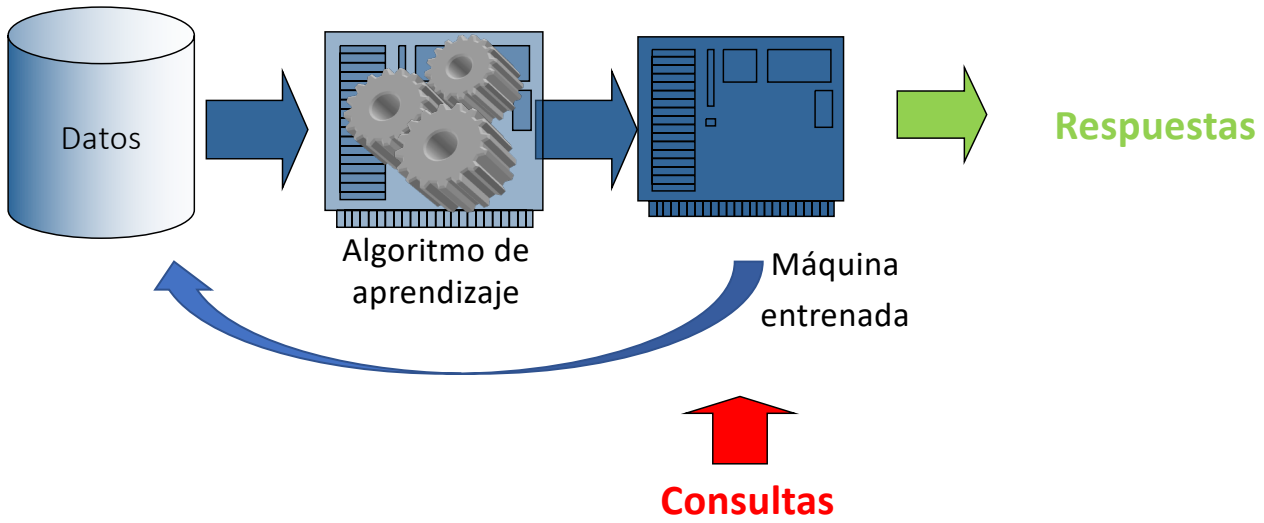
Cómo podemos construir programas de computadora que mejoren con la experiencia?

Cuáles son las leyes fundamentales que rigen los procesos de aprendizaje?

T. M. Mitchell. **The Discipline of Machine Learning.** Tech. Report. CMU-ML-06-108, 2006

Aprendizaje computacional

ML = representación + evaluación + optimización



Pedro Domingos. **A Few Useful Things to Know about Machine Learning**. Communications of the ACM, 55(10):78--87, 2012

Isabelle Guyon. **A Practical Guide to Model Selection.** In Jeremie Marie, editor, Machine Learning Summer School 2008, Springer Texts in Statistics, 2011. (slide from I.Guyon's)

Principales variantes

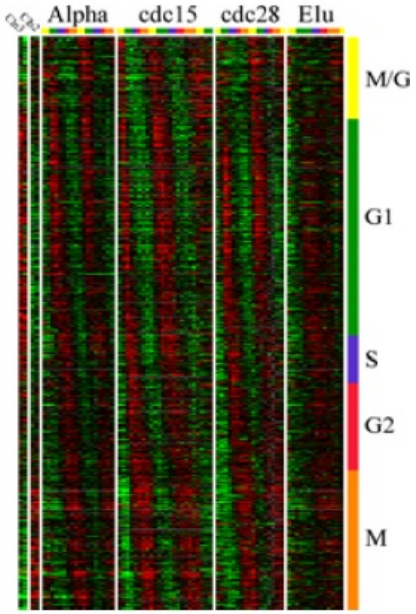
- **Aprendizaje supervisado:** hacer predicciones con respecto a una o más variables

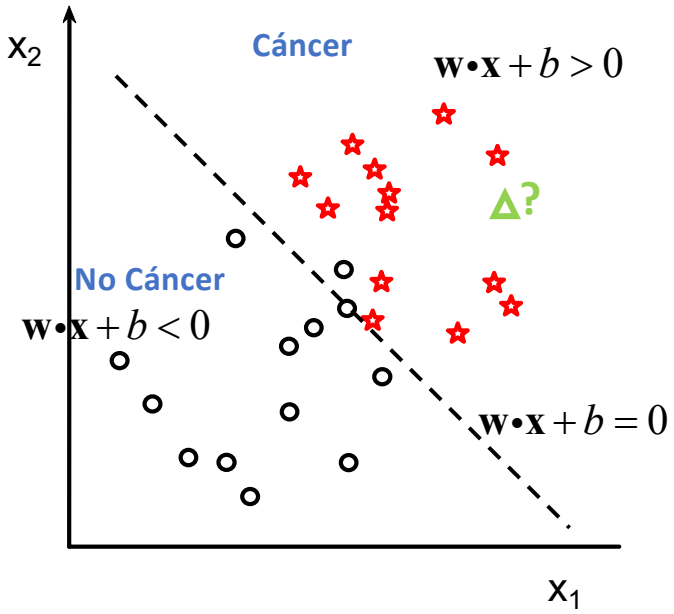
- **Aprendizaje semi-supervisado:** Usar técnicas supervisadas y la estructura de los datos para tareas heterogéneas

- **Aprendizaje no-supervisado:** hallar la mejor estructura (si la hay) para los datos

Aprendizaje supervisado

- Clasificación de Micro-arreglos de ADN



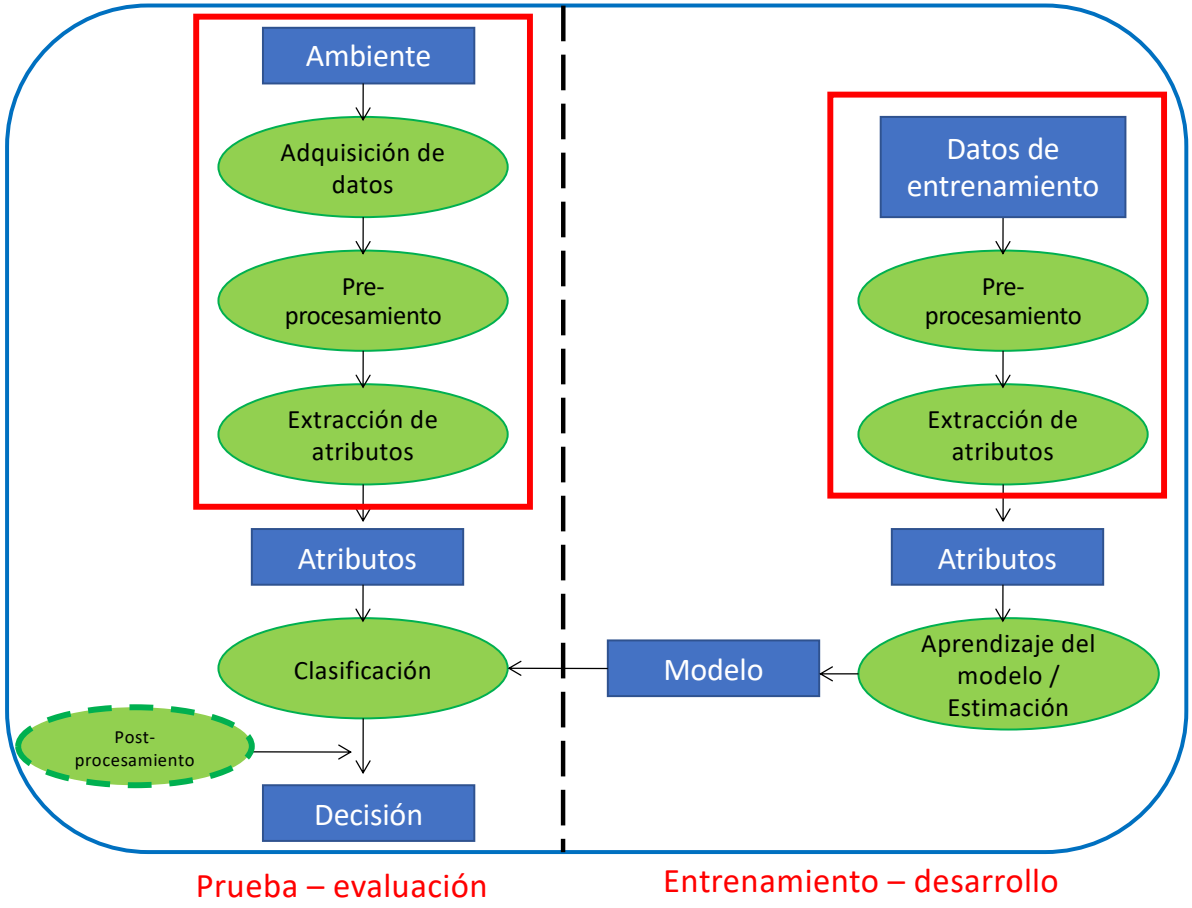


$$\mathbf{x} = \langle \mathbf{x}_1, \mathbf{x}_2 \rangle$$

$$f(\mathbf{x}) = \mathbf{w} \bullet \mathbf{x} + b$$

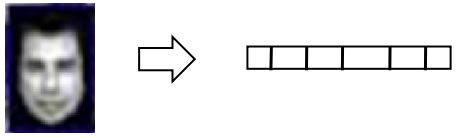
El proceso de diseño de un sistema de reconocimiento de patrones

R. O. Duda, P. Hart, D. Stork.
Pattern Classification.
Wiley, 2001.

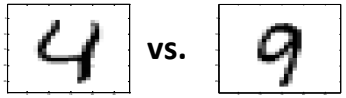


Aprendizaje supervisado

- Un patrón es una entidad vagamente definida, a la cual se le puede dar nombre e.g.:
  - La imagen de un rostro
  - Una secuencia de ADN
  - Un número escrito a mano



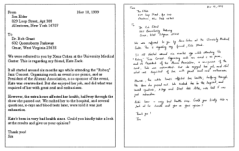
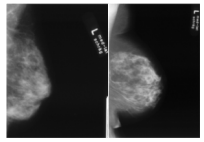
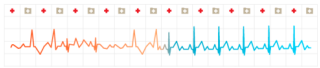
- Reconocimiento de patrones es el estudio de métodos automáticos para:
  - Aprender a distinguir patrones de interés
  - Tomar decisiones razonables de acuerdo a las categorías de los patrones



Datos etiquetados

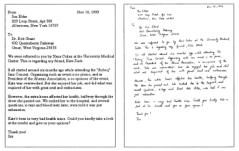
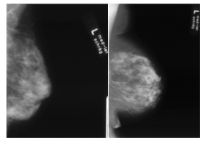
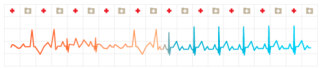
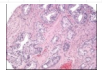
- Cuando los datos están asociados a categorías o clases se denominan datos etiquetados

Datos etiquetados

Dato	Etiqueta
	John
	Economía
	Saludo
	Tumor
	Saludable
	Saludable
	Auto

Usualmente los datos se representan como vectores numéricos y las etiquetas con números entre 1 y K

Datos etiquetados

Dato	Etiqueta	Dato	Etiqueta
	John		1
	Economía		3
	Saludo		2
	Tumor		1
	Saludable		2
	Saludable		1
	Auto		24

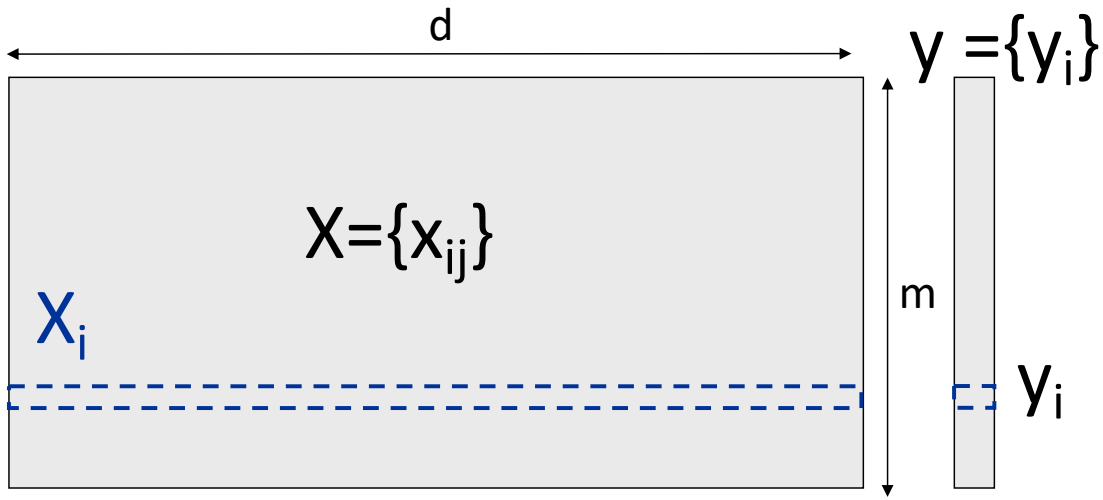
Datos etiquetados

- Los datos se representan por un conjunto de atributos o mediciones; ejemplos:
  - **Imágenes en color:** Contenido de color en RGB, textura, intensidad, etc.
  - **Textos:** Bolsa de palabras
  - **Imágenes en B/N:** Concatenación de filas de la matriz
  - **Señales:** valor de la señal

- Los atributos pueden ser:
  - Binarios
  - Enteros
  - Continuos
  - Categóricos

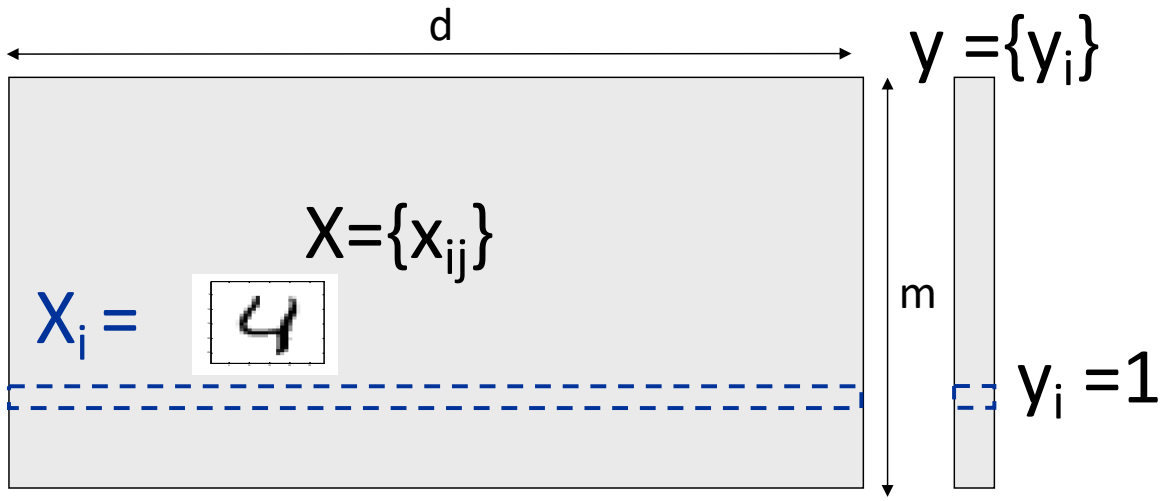
Convenciones

- Matriz de datos: **X**
  - **m** filas = patrones (datos): muestras, pacientes, documentos, imágenes, …
  - **n** columnas = atributos: (*features*, variables de entrada): genes, palabras, píxeles, …
- Vector de "salidas" **Y** con y_i en {-1, 1} (binario) o en {1,2, …, K} (multiclase)



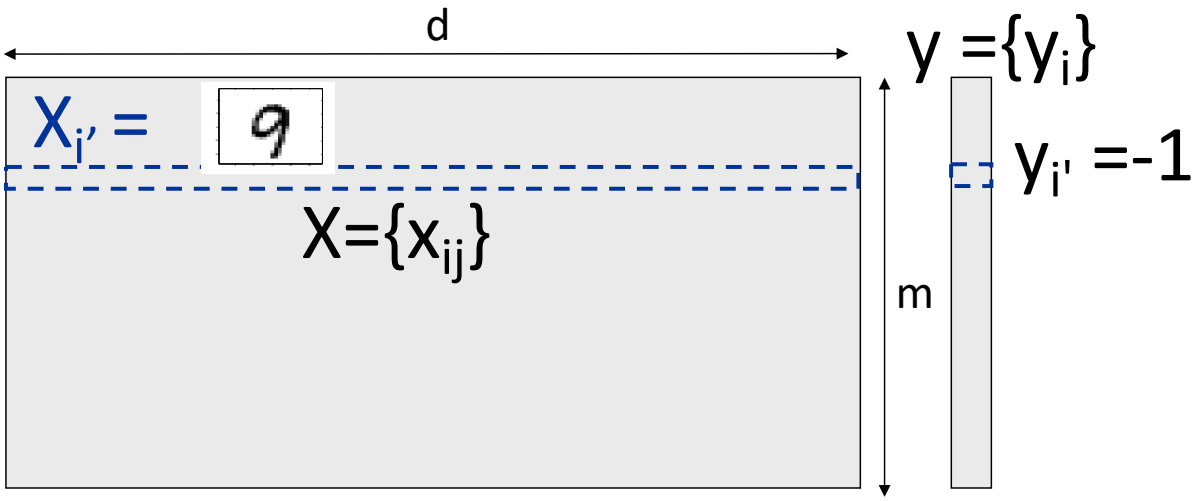
Convenciones

- Matriz de datos: **X**
  - **m** filas = patrones (datos): muestras, pacientes, documentos, imágenes, …
  - **n** columnas = atributos: (*features*, variables de entrada): genes, palabras, píxeles, …
- Vector de "salidas" **Y** con y_i en {-1, 1} (binario) o en {1,2, …, K} (multiclase)

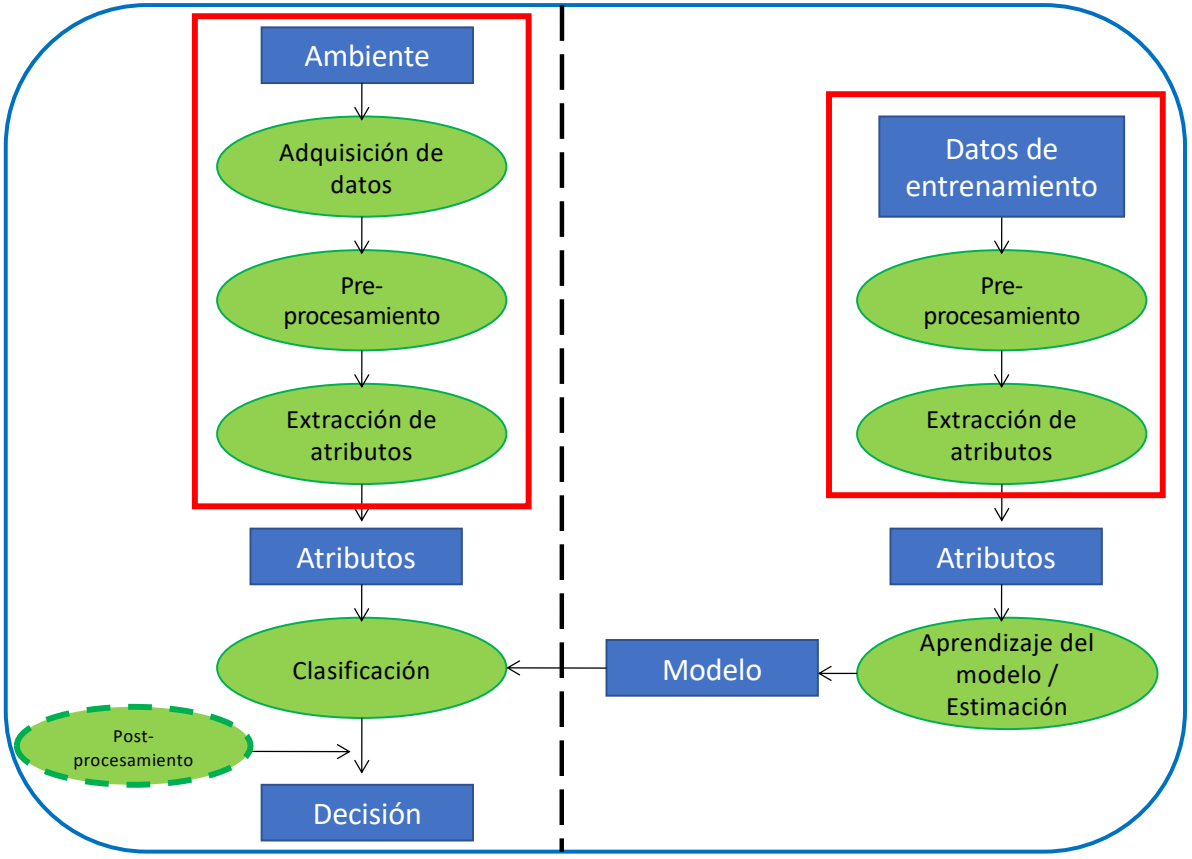


Convenciones

- Matriz de datos: **X**
  - **m** filas = patrones (datos): muestras, pacientes, documentos, imágenes, …
  - **n** columnas = atributos: (*features*, variables de entrada): genes, palabras, píxeles, …
- Vector de "salidas" **Y** con y_i en {-1, 1} (binario) o en {1,2, …, K} (multiclase)



El proceso de diseño de un sistema de reconocimiento de patrones



Más formalmente…

- Dado un conjunto de datos etiquetado:

$$D = \{(\mathbf{x_i}, y_i)\}_{1,\ldots m} \qquad \mathbf{x} \in \mathbb{R}^d; y \in \mathrm{C}$$

- En aprendizaje supervisado se busca encontrar una función:

$$f: \mathbb{R}^d \longrightarrow C$$

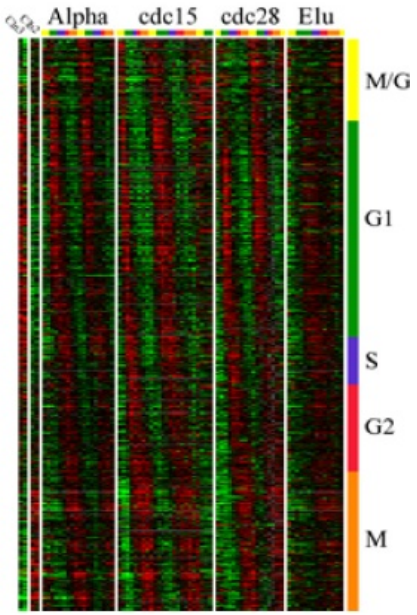
Más formalmente…

- Clasificación binaria: $f: \mathbb{R}^d \longrightarrow \{-1,1\}; \quad \mathbf{x} \in \mathbb{R}^d; y \in \{-1,1\}$
- Clasificación multi-clase: $f: \mathbb{R}^d \longrightarrow \{C_1, \dots, C_K\}; \quad \mathbf{x} \in \mathbb{R}^d; y \in \{C_1, \dots, C_K\}$
- Clasificación multi-etiqueta: $f: (\mathbb{R}^d, C_i) \longrightarrow \{0,1\}; \quad \mathbf{x} \in \mathbb{R}^d; y \in \{C_1, \dots, C_K\}$
- Regressión: $f: \mathbb{R}^d \longrightarrow \mathbb{R}^p; \quad \mathbf{x} \in \mathbb{R}^d; y \in \mathbb{R}^p$

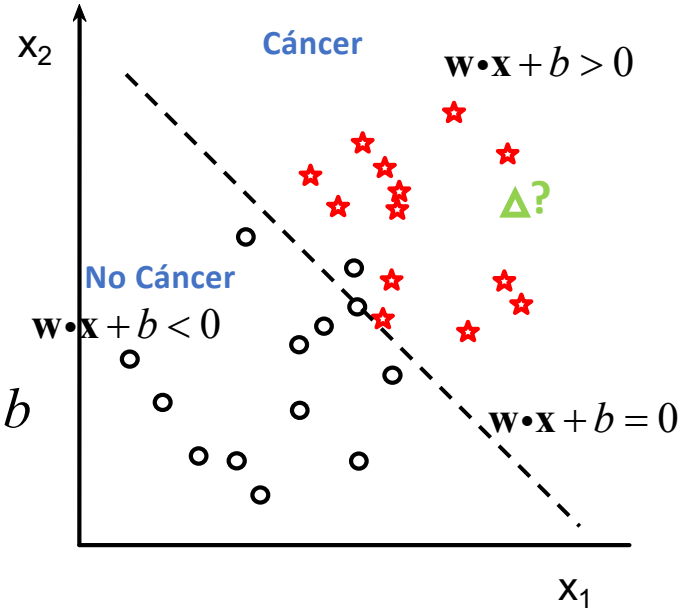
Clasificación binaria : 2 clases

$$f: \mathbb{R}^d \longrightarrow \{-1,1\}; \qquad \mathbf{x} \in \mathbb{R}^d; y \in \{-1,1\}$$

$$\mathbf{x} = \langle \mathbf{x}_1, \mathbf{x}_2 \rangle$$



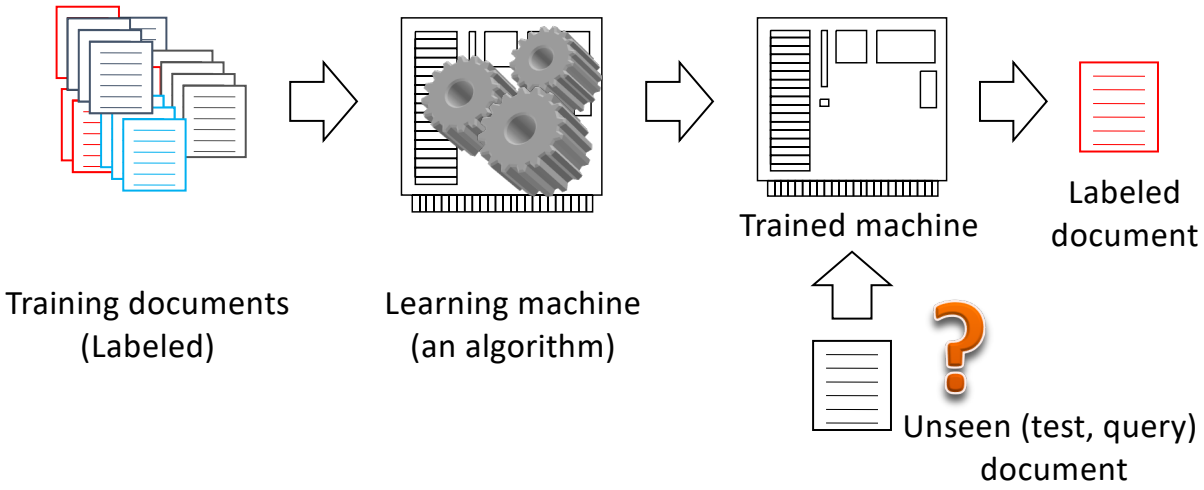
$$f(\mathbf{x}) = \mathbf{w} \bullet \mathbf{x} + b$$



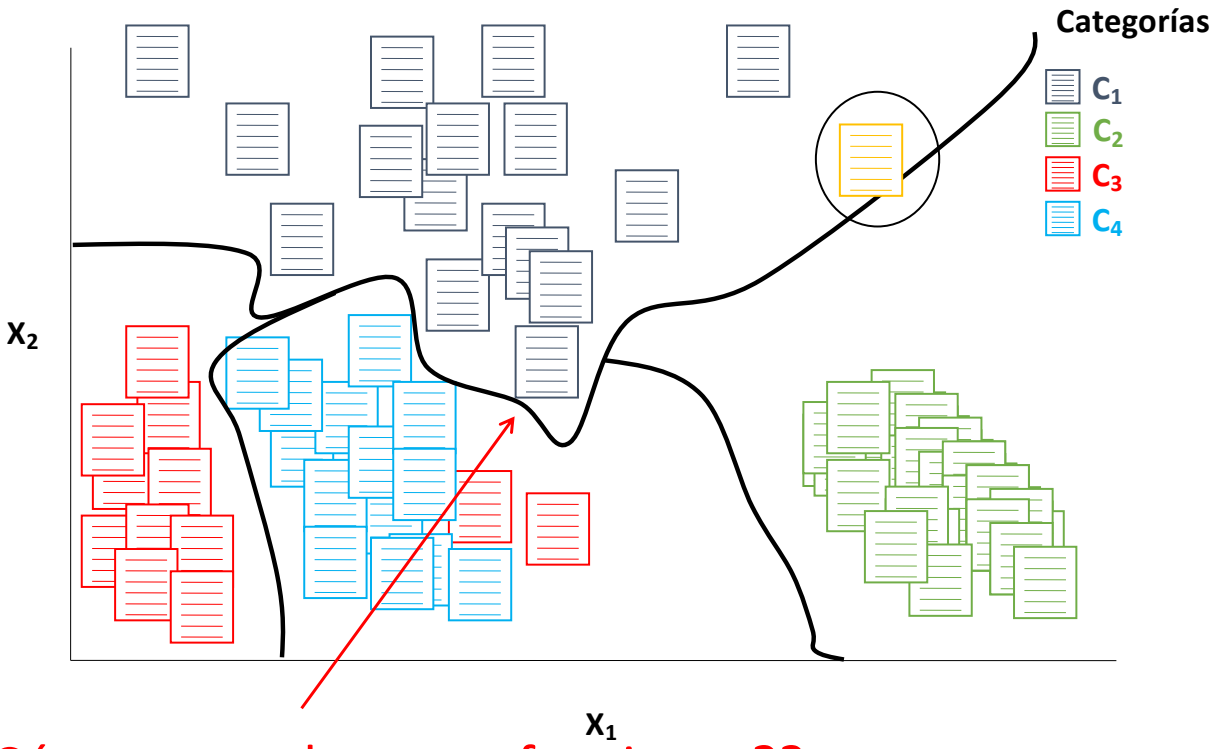
Clasificación multi-clase : k-categorías

$$f: \mathbb{R}^d \longrightarrow \{C_1, \ldots, C_K\};\ \mathbf{x} \in \mathbb{R}^d; y \in \{C_1, \ldots, C_K\}$$

- Desarrollo de métodos automáticos para clasificar documentos en categorías predefinidas



Ejemplo: clasificación de documentos

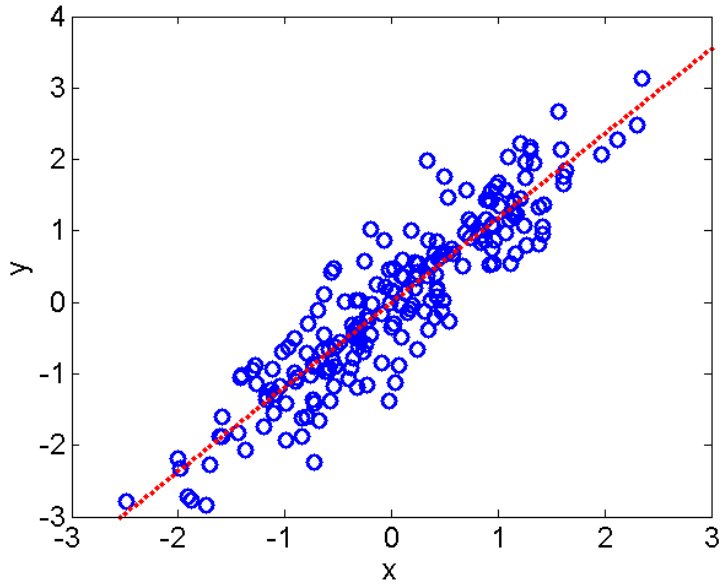


Cómo aprender estas funciones??

Regresión

$$f: \mathbb{R}^d \longrightarrow \mathbb{R}^p; \qquad \mathbf{x} \in \mathbb{R}^d; y \in \mathbb{R}^p$$

- Estimación de una variable continua



Resumiendo

- Tratamos de aprender *algo* que nos permita mapear entradas a salidas deseadas
- Las entradas serán vectores en R^d y las salidas pueden ser diversas
- Para aprender contamos con ejemplos etiquetados (supervisión)
- Podemos distinguir dos fases: desarrollo (entrenamiento) y evaluación (prueba)
- El objetivo es obtener un modelo que generalice más allá de la muestra

Regresión

Regresión

- Estimación de una variable continua

- Encontrar:

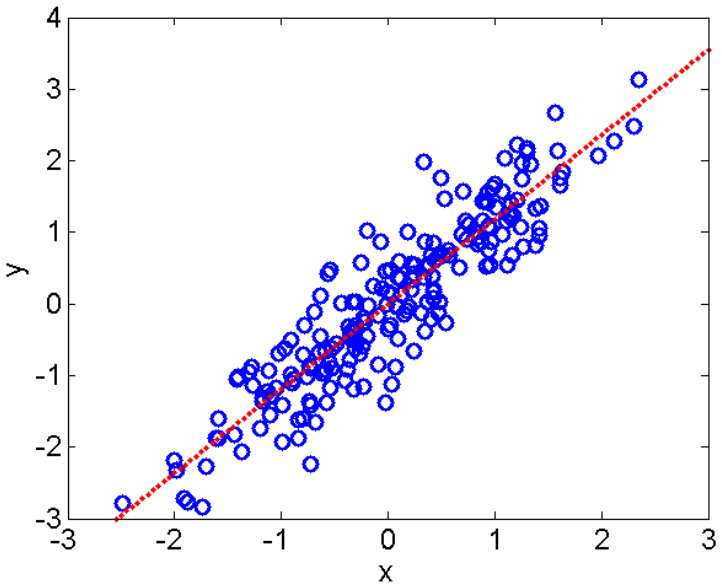
$$f: \mathbb{R}^d \longrightarrow \mathbb{R}^p;$$

- Dado:

$$D = \{(\mathbf{x_i}, y_i)\}_{1,\dots m}$$

- Con:

$$\mathbf{x} \in \mathbb{R}^d; y \in \mathbb{R}^p$$

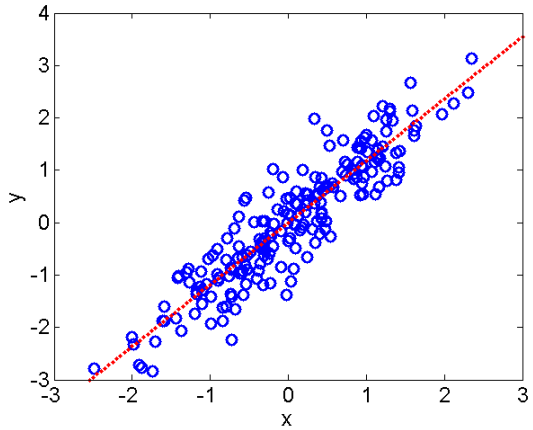


Regresión lineal

- Uno de los modelos más sencillos para regresión es un modelo lineal de la forma:

$$f(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 \ldots + + w_d x_d$$

- Se asume que los datos **D** son generados de acuerdo a *f*



Regresión lineal

- Uno de los modelos más sencillos para regresión es un modelo lineal de la forma:

$$f(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 \ldots + + w_d x_d$$

Con d=2 $\quad \mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix} \quad \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}$

$$f(\mathbf{x}, \mathbf{w}) = \mathbf{w}\mathbf{x}$$

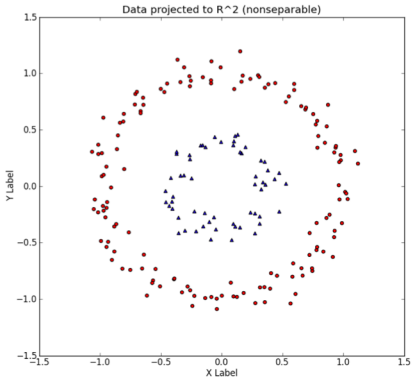
Regresión lineal

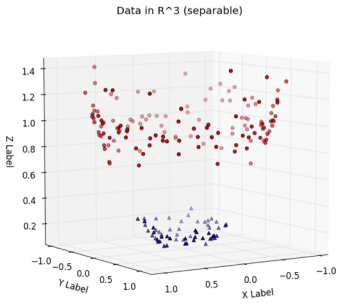
- La restricción lineal limita al modelo, pero es posible hacer un mapeo no lineal de las variables antes de aplicar el modelo lineal:

$$f(\mathbf{x}, \mathbf{w}) = w_0 + w_1\phi(x_1) + \cdots + w_d\phi(x_d)$$

$$f(\mathbf{x}, \mathbf{w}) = \mathbf{w}\boldsymbol{\phi}(\mathbf{x})$$

Qué forma tiene $\boldsymbol{\phi}$?





Regresión lineal

- La restricción lineal limita al modelo, pero es posible hacer un mapeo no lineal de las variables antes de aplicar el modelo lineal:

$$f(\mathbf{x}, \mathbf{w}) = w_0 + w_1\phi(x_1) + \cdots + w_d\phi(x_d)$$

$$f(\mathbf{x}, \mathbf{w}) = \mathbf{w}\boldsymbol{\phi}(\mathbf{x})$$

Qué forma tiene $\boldsymbol{\phi}$?

- Asumamos que $\boldsymbol{\phi}$ es la función identidad: $\boldsymbol{\phi}(\mathbf{x}) = \mathbf{x}$
- Cómo encontrar los pesos **w** qué definen al modelo?

Regresión lineal: mínimos cuadrados

- Una forma de abordar el problema es tratando de encontrar un aproximador de f, f* que minimice el cuadrado de los *residuos*:

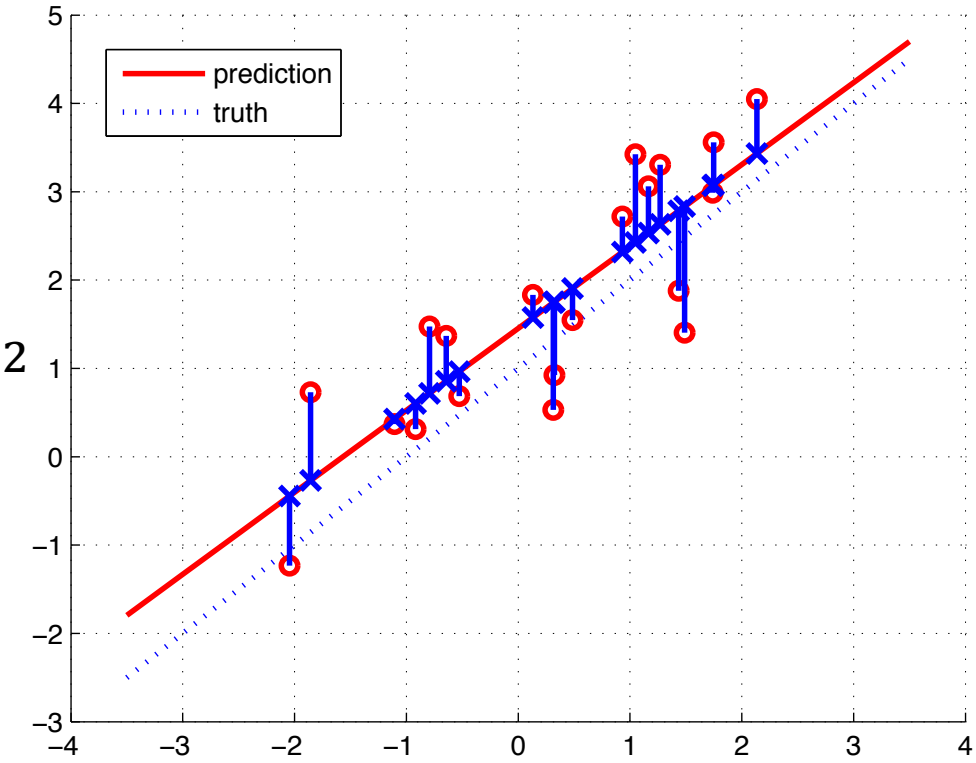
$$\epsilon^2 = (f(\mathbf{x}, \mathbf{w}) - \boldsymbol{f}^*(\mathbf{x}, \mathbf{w}))^2$$

- Para una muestra de m-datos, queremos encontrar la función f* que minimiza:

$$\epsilon^2 = \sum_{i=1}^{m} (f(\mathbf{x}, \mathbf{w}) - f^*(\mathbf{x}, \mathbf{w}))^2 \qquad = \sum_{i=1}^{m} (y_i - \mathbf{w}^* \mathbf{x}_i)^2$$

Regresión lineal: mínimos cuadrados

$$\epsilon^2 = (f(\mathbf{x}, \mathbf{w}) - \boldsymbol{f}^*(\mathbf{x}, \mathbf{w}))^2$$



http://people.cs.ubc.ca/~murphyk/MLbook/figReport-16-Aug-2012/pdfFigures/linRegResiduals.pdf

Regresión lineal: mínimos cuadrados

- Una forma de abordar el problema es tratando de encontrar un aproximador f* de f que minimice el cuadrado de los *residuos*:

$$\epsilon^2 = (f(\mathbf{x}, \mathbf{w}) - \boldsymbol{f}^*(\mathbf{x}, \mathbf{w}))^2$$

- Para una muestra de m-datos, queremos encontrar la función f* que minimiza

$$\epsilon^2 = \sum_{i=1}^{m} (f(\mathbf{x}, \mathbf{w}) - f^*(\mathbf{x}, \mathbf{w}))^2 \qquad = \sum_{i=1}^{m} (y_i - \mathbf{w}^* \mathbf{x}_i)^2$$

Regresión lineal: mínimos cuadrados

- Queremos encontrar **w** tal que se minimiza E,

$$E(\mathbf{w}^*) = \sum_{i=1}^{m} (y_i - \mathbf{w}^* \mathbf{x}_i)^2$$

- Por conveniencia:

$$E(\mathbf{w}^*) = (\mathbf{y} - \mathbf{XW})^T (\mathbf{y} - \mathbf{XW})$$

Regresión lineal: mínimos cuadrados

- Queremos encontrar **w** tal que se minimiza E,

$$E(\mathbf{w}^*) = \sum_{i=1}^{m} (y_i - \mathbf{w}^* \mathbf{x}_i)^2$$

- Derivando E con respecto a **w**:

$$\frac{dE(\mathbf{w}^*)}{d\mathbf{w}^*} = \mathbf{X}^T(\mathbf{y} - \mathbf{X}\mathbf{W})$$

Regresión lineal: mínimos cuadrados

- Queremos encontrar **w** tal que se minimiza E,

$$E(\mathbf{w}^*) = \sum_{i=1}^{m} (y_i - \mathbf{w}^* \mathbf{x}_i)^2$$

- Igualando a ceros:

$$\mathbf{X}^T(\mathbf{y} - \mathbf{X}\mathbf{W}) = \mathbf{0}$$

$$\mathbf{W} = (\mathbf{X}^T\mathbf{X})^{-1}\,\mathbf{X}^T\mathbf{y}$$

Regresión lineal: mínimos cuadrados

- Queremos encontrar **w** tal que se minimiza E,

$$E(\mathbf{w}^*) = \sum_{i=1}^{m} (y_i - \mathbf{w}^* \mathbf{x}_i)^2$$

- Si $\mathbf{X}^T\mathbf{X}$ es invertible, se tiene que la solución única está dada por:

$$\mathbf{W} = (\mathbf{X}^T\mathbf{X})^{-1}\,\mathbf{X}^T\mathbf{y}$$

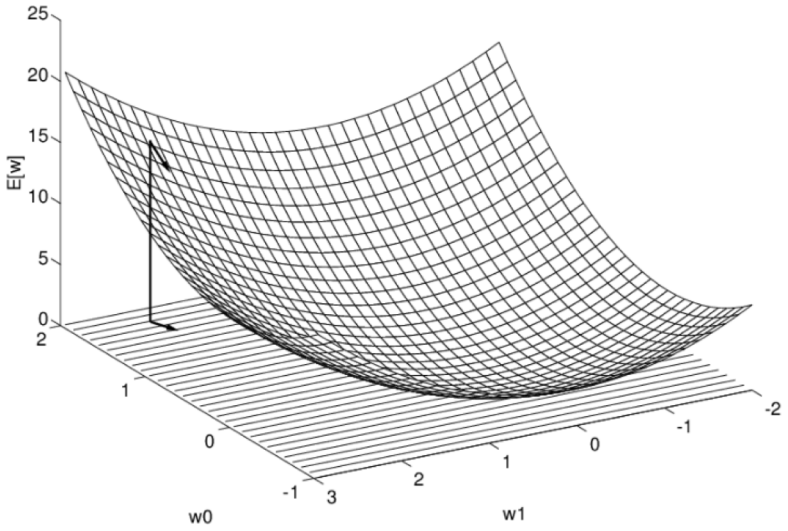
Regresión lineal

- La solución de mínimos cuadrados es óptima, ideal cuando es posible invertir la matriz, qué pasa cuando esto no es posible?

Regresión lineal: Gradiente descendente

- Alternativa:
  - Minimize $E(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(y_i - o_i)^2$ w.r.t **w**

- Idea: explorar el espacio de posibles valores que w puede tomar. Iniciando con un w aleatorio y actualizándolo cierta magnitud en la dirección que disminuye el error.
  - El gradiente de *E*(**w**) indica la dirección que produce el mayor incremento en **w**



Regresión lineal: Gradiente descendente

- Regla de actualización:

$$W \leftarrow W + \Delta W$$

$$\Delta W = -\alpha \nabla E$$

- Estimando delta

$$\begin{aligned} \frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i} \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2 \\ &= \sum_{d \in D} (t_d - o_d) \frac{\partial}{\partial w_i} (t_d - \vec{w} \cdot \vec{x}_d) \\ &= \sum_{d \in D} (t_d - o_d)(-x_{i,d}) \end{aligned}$$

$$\Delta w_i = \alpha \sum_{d \in D} (t_d - o_d) x_{i,d}$$

Regresión lineal: Gradiente descendente

- Estimación de **w** mediante gradiente descendente:

1. **w** ← randomly initialize weights
2. Repeat until stop criterion meet
   - *I.* $\Delta\mathbf{w} \leftarrow$ initialize to 0
   - II. For each $\mathbf{x}_i \in D$
     - *a)* $o_i \leftarrow \mathbf{w}\mathbf{x}_i + b$ // estimate the prediction
     - b) For each weight j estimate
       - *1.* $\Delta\mathbf{w}_j \leftarrow \Delta\mathbf{w}_j + (y_i - o_i)^2\, x_{i,j}$ // estimate the rate of change
   - **III.** $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{w}$ //Update *w*
3. *Return* **w**

Regresión lineal: Gradiente descendente

- SGD: en la práctica una version estocástica del algoritmo es utilizada, los pesos se actualizan después de procesar cada dato.

1. $\mathbf{w} \leftarrow$ randomly initialize weights
2. Repeat until stop criterion meet
   - *I.* $\Delta\mathbf{w} \leftarrow$ initialize to 0
   - II. For each $\mathbf{x}_i \in D$
     - *a)* $o_i \leftarrow \mathbf{w}\mathbf{x}_i + b$ // estimate the prediction
     - b) For each weight j estimate
       - *1.* $\Delta\mathbf{w}_j \leftarrow \Delta\mathbf{w}_j + (y_i - o_i)^2\, x_{i,j}$ // estimate the rate of change
     - **c)** $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{w}$ //Update *w*
3. *Return* **w**

Interpretación probabilista

- Supongamos que y está dada por una función determinista más un ruido Gausiano aditivo:

$$y = f(\mathbf{x}, \mathbf{w}) + \epsilon$$

$$p(y|\mathbf{x}, \mathbf{w}, \sigma) = \mathcal{N}(y|f(\mathbf{x}, \mathbf{w}), \sigma^{\mathbf{2}})$$

$$p(y|\mathbf{x}, \mathbf{w}, \sigma) = \mathcal{N}(y|\mathbf{w}^T\mathbf{x}, \sigma^{\mathbf{2}})$$

Regresión: recapitulando

- Diversas formas para resolver el mismo problema,
- Diferentes soluciones
- Los modelos son definidos por los vectores de pesos w
- Aprendizaje supervisado que existe desde el siglo XVIII
- Beneficios de regresión lineal: simpleza, eficiencia, versiones online, performance satisfactorio cuando los datos siguen una distribución lineal, interpretabilidad del modelo

Regresión: extensiones

- Muchas otras formas de abordar el problema: árboles de decisión, redes neuronales, regresión por vectores de soporte, procesos Gaussianos, etc. (Algunas de éstas se verán en el curso)

- Extensión para datos no linealmente separables

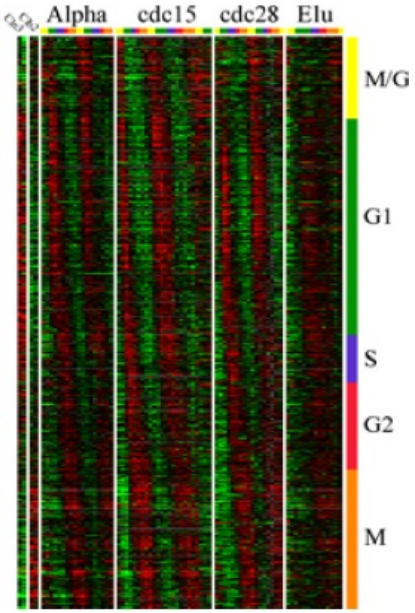
Clasificación

Modelos lineales para clasificación

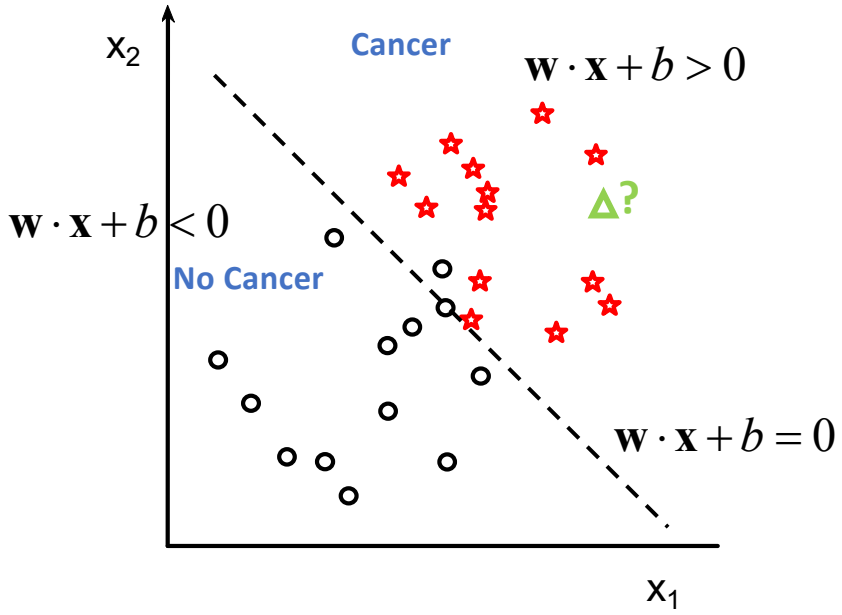
- **Idea:** aprender una función lineal (en los paramétros) que nos permita separar los datos:
  - $f(\mathbf{x}) = \mathbf{w} \bullet \mathbf{x} + b = \sum_{j=1:n} w_j x_j + b$ *(linear discriminant)*
  - $f(\mathbf{x}) = \mathbf{w} \bullet \Phi(\mathbf{x}) + b = \sum_{j} w_j \phi_j(\mathbf{x}) + b$ *(the perceptron)*
  - $f(\mathbf{x}) = \sum_{i=1:m} \alpha_i k(\mathbf{x}_i, \mathbf{x}) + b$ *(Kernel-based methods)*

Linear Discriminants and Support Vector Machines, *I. Guyon and D. Stork,* In Smola et al Eds. Advances in Large Margin Classifiers. Pages 147--169, MIT Press, 2000.

Modelos lineales para clasificación



$$\mathbf{x} = \langle \mathbf{x}_1, \mathbf{x}_2 \rangle$$



$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$$

Modelos lineales para clasificación

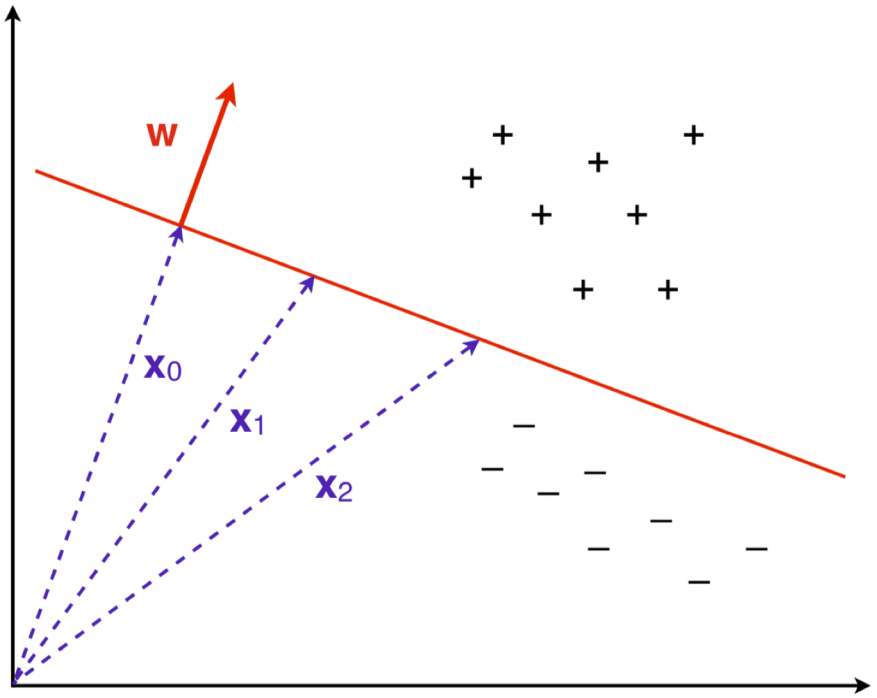
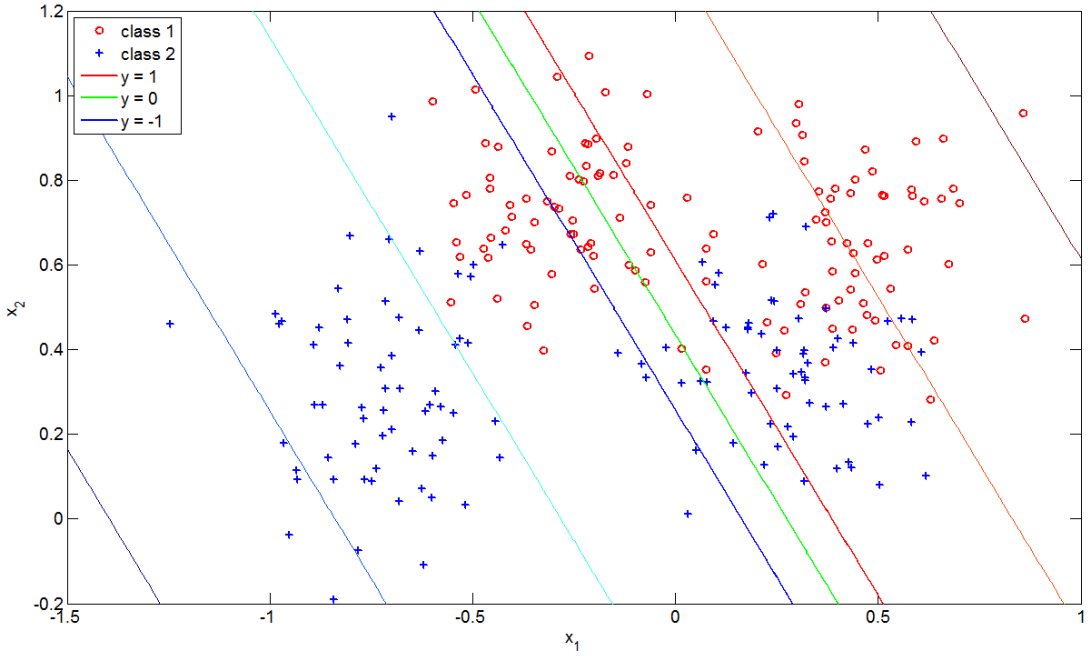


Figura de P. Flach. Machine Learning. The Art and Science of Algorithms that Make Sense of Data, Cambridge University Press, 2012

Modelos lineales para clasificación

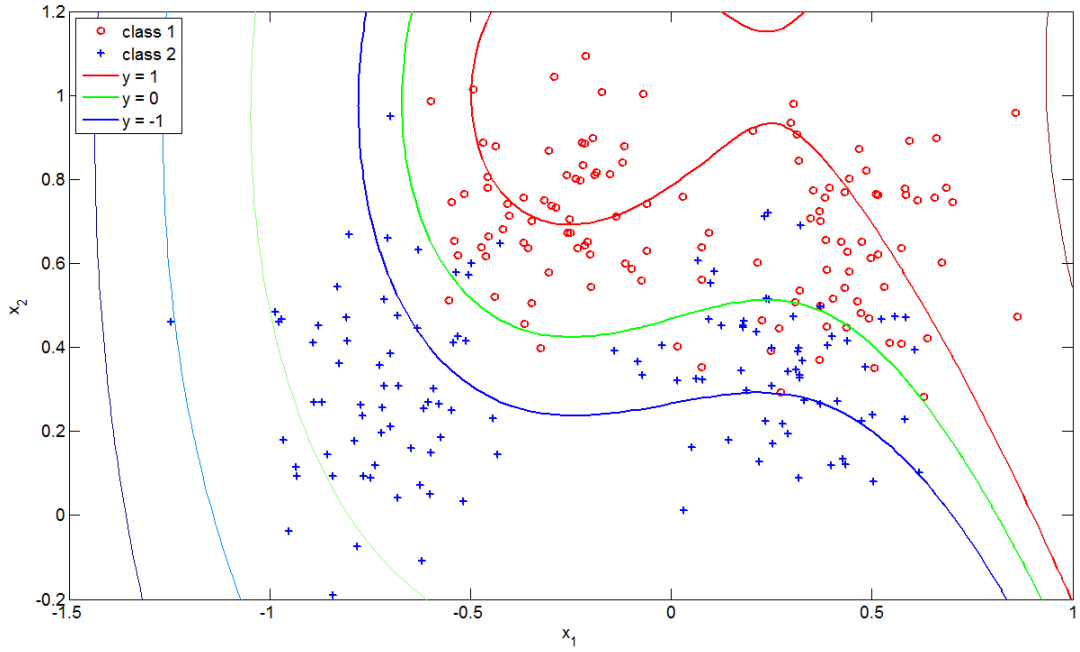
http://clopinet.com/CLOP



Linear support vector machine

Modelos lineales para clasificación

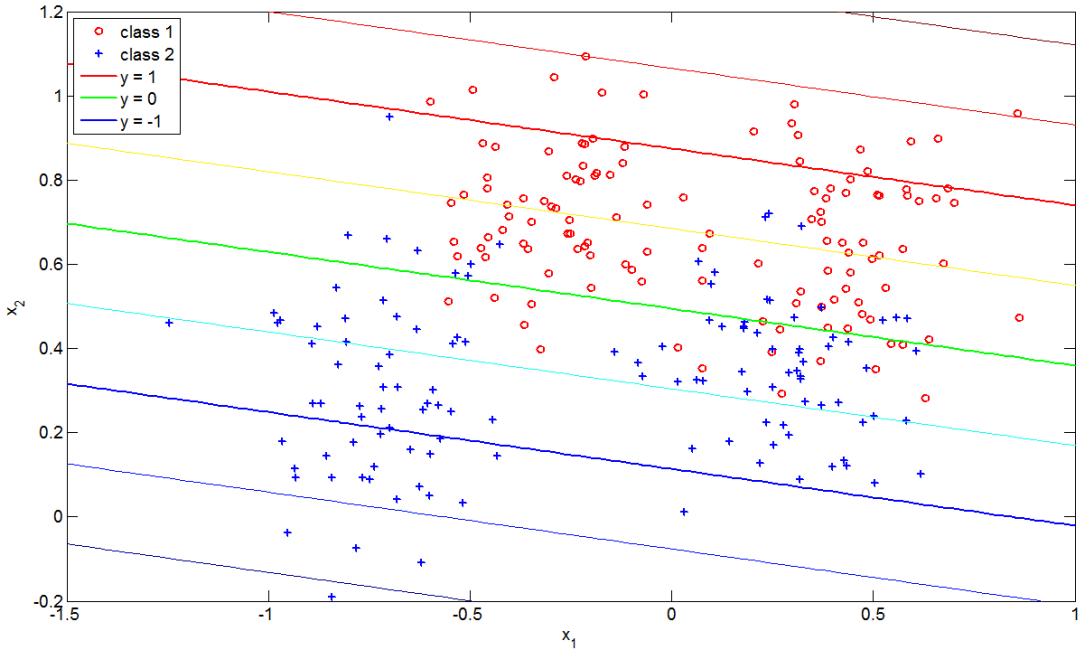
http://clopinet.com/CLOP



"Non-linear" support vector machine

Modelos lineales para clasificación

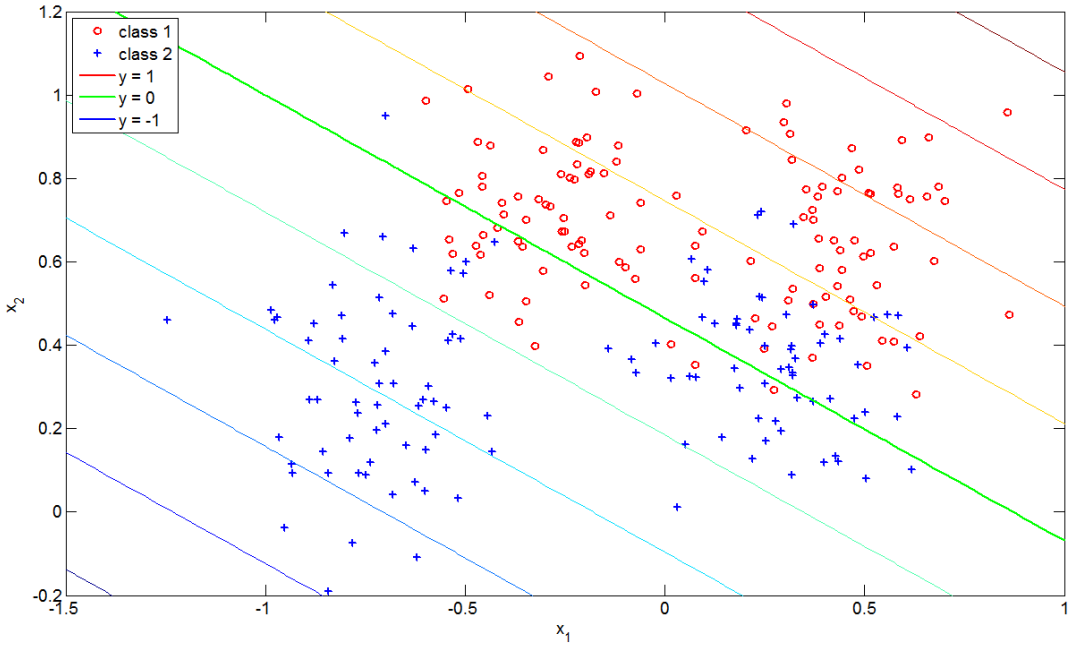
http://clopinet.com/CLOP



Kernel ridge regression

Modelos lineales para clasificación

http://clopinet.com/CLOP



Zarbi classifier

El perceptron

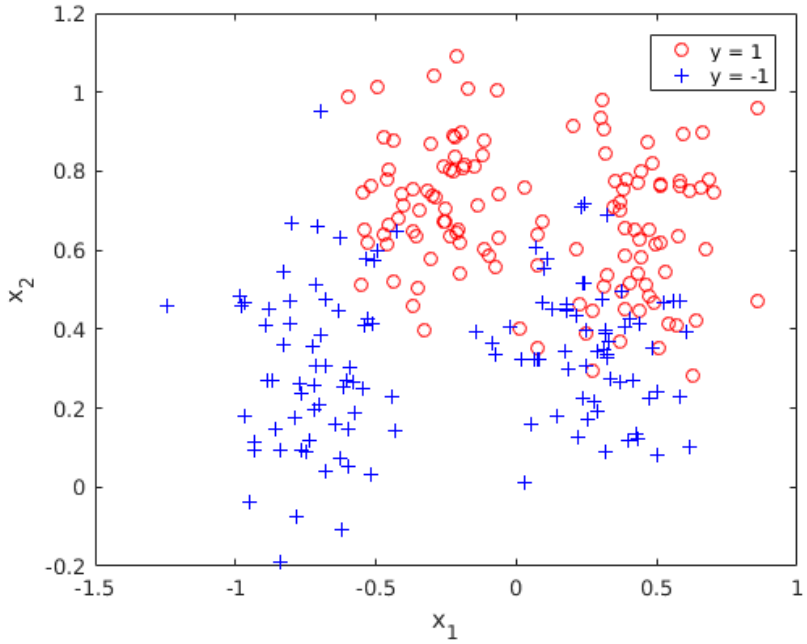
- **Perceptron:** Un clasificador lineal simple que puede resolver problemas de clasificación que son linealmente separables (abuelo de las redes neuronales y las máquinas de soporte vectorial)

- El perceptron aprende una función de decision de la forma:

$$f(\mathbf{x}) = sign(\mathbf{w}\mathbf{x} + b), \text{ with } \mathbf{w} \in \mathbb{R}^d$$

El perceptron

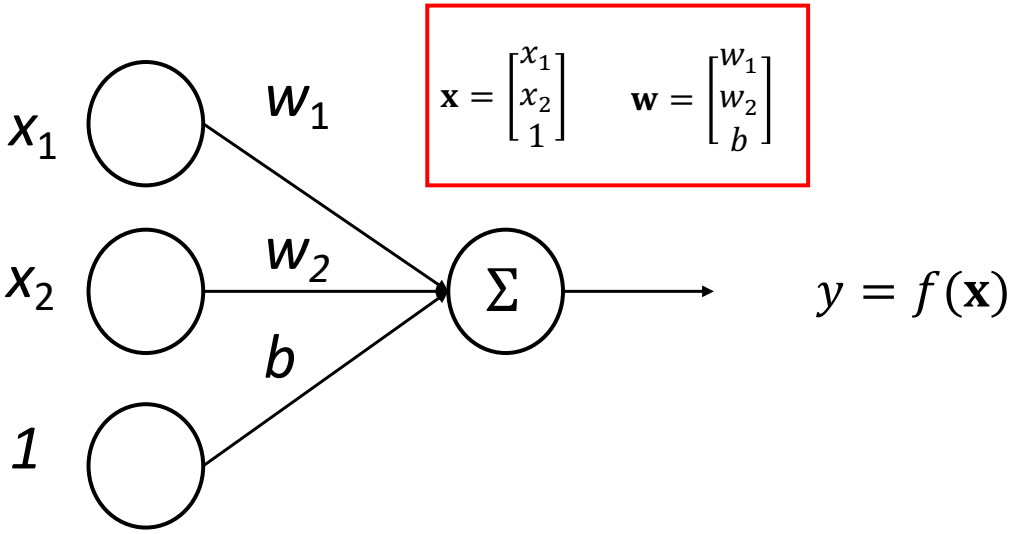
- Given: $D = \{(\mathbf{x}_i, y_i)_{1.....N}\}$, with $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$



El perceptron

- El perceptron aprende una función de decision de la forma:

$$f(\mathbf{x}) = sign(\mathbf{w}\mathbf{x} + b), \text{ with } \mathbf{w} \in \mathbb{R}^d$$

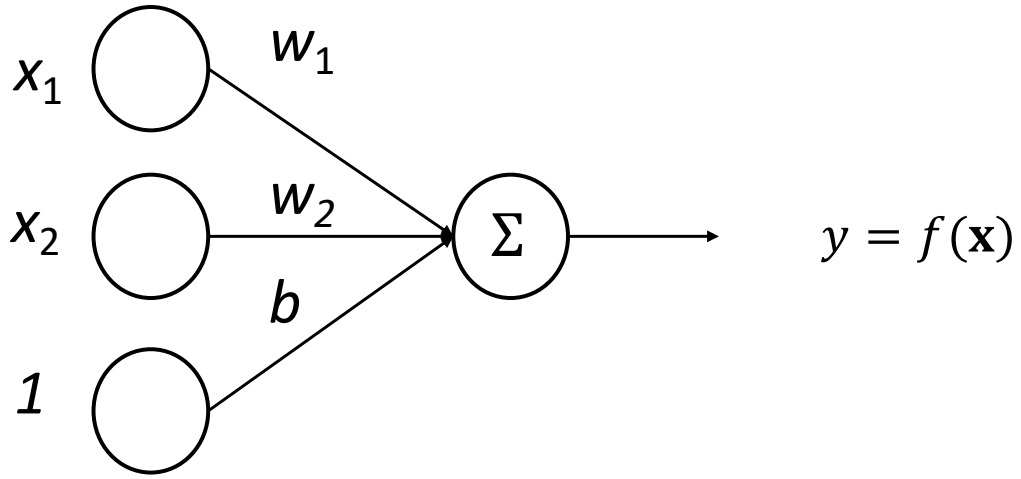


$$y = w_1 x_1 + w_2 x_2 + b$$

El perceptron

- El perceptron aprende una función de decision de la forma:

$$f(\mathbf{x}) = sign(\mathbf{w}\mathbf{x} + b), \text{ with } \mathbf{w} \in \mathbb{R}^d$$

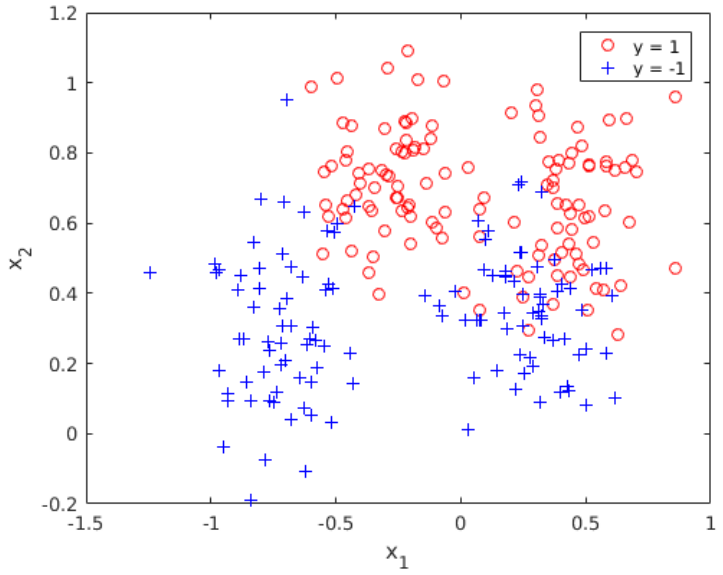


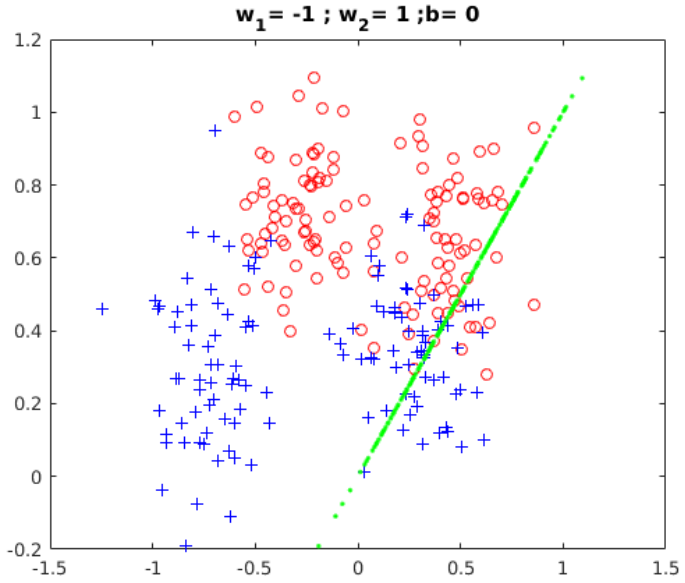
$$y = w_1 x_1 + w_2 x_2 + b$$

El perceptron

- El perceptron aprende una función de decision de la forma:

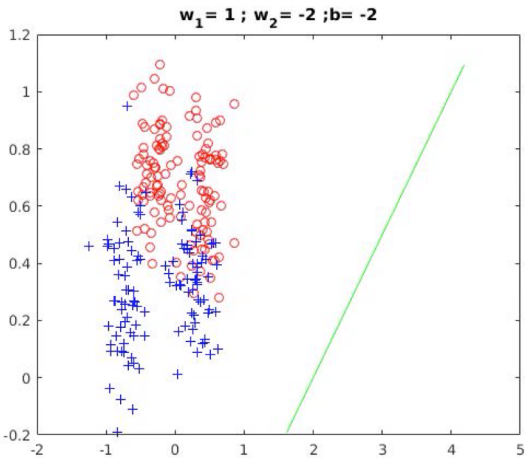
$$f(\mathbf{x}) = sign(\mathbf{w}\mathbf{x} + b), \text{ with } \mathbf{w} \in \mathbb{R}^d$$

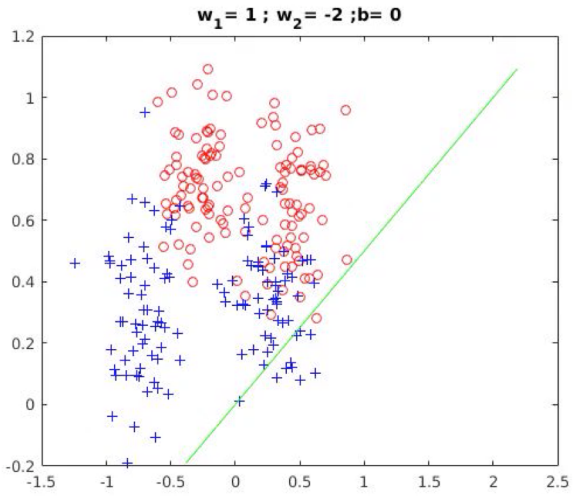


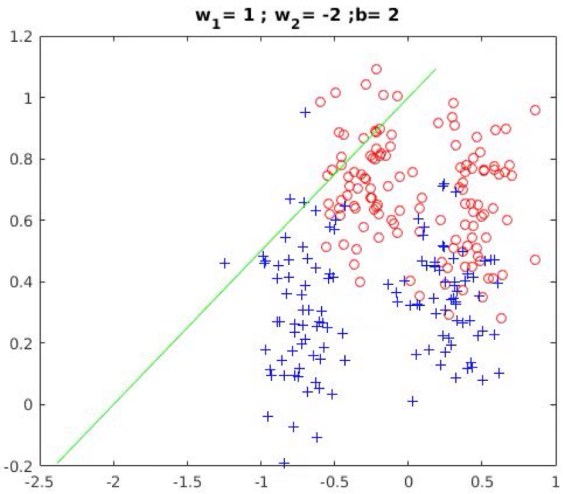


El perceptron

- Diferentes paramétros resultan en diferentes modelos







El perceptron

- Cómo estimar los pesos **w**?

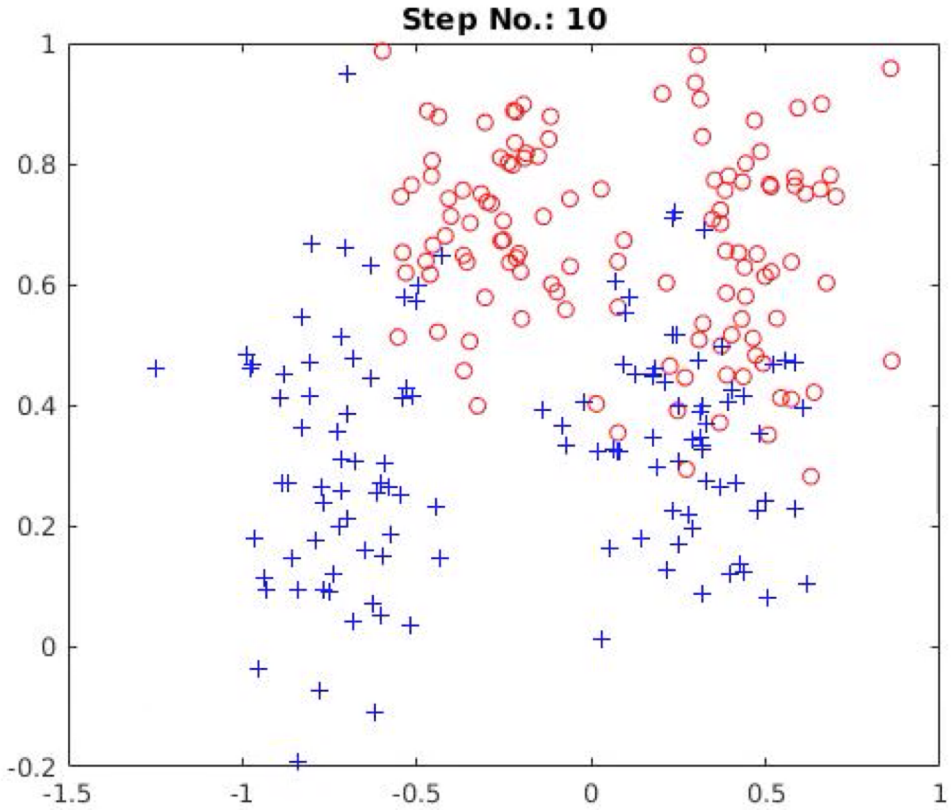
- El algoritmo de aprendizaje del perceptron

1. **w** ← randomly initialize weights
2. Repeat until stop criterion meet
   I. For each $\mathbf{x}_i \in D$
      a) $o_i \leftarrow \mathbf{w}\mathbf{x}_i + b$ // estimate perceptron's prediction
      b) $\Delta\mathbf{w} \leftarrow \eta(y_i - o_i)$ // estimate the rate of change
      c) $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{w}$ //Update *w*
3. *Return* **w**

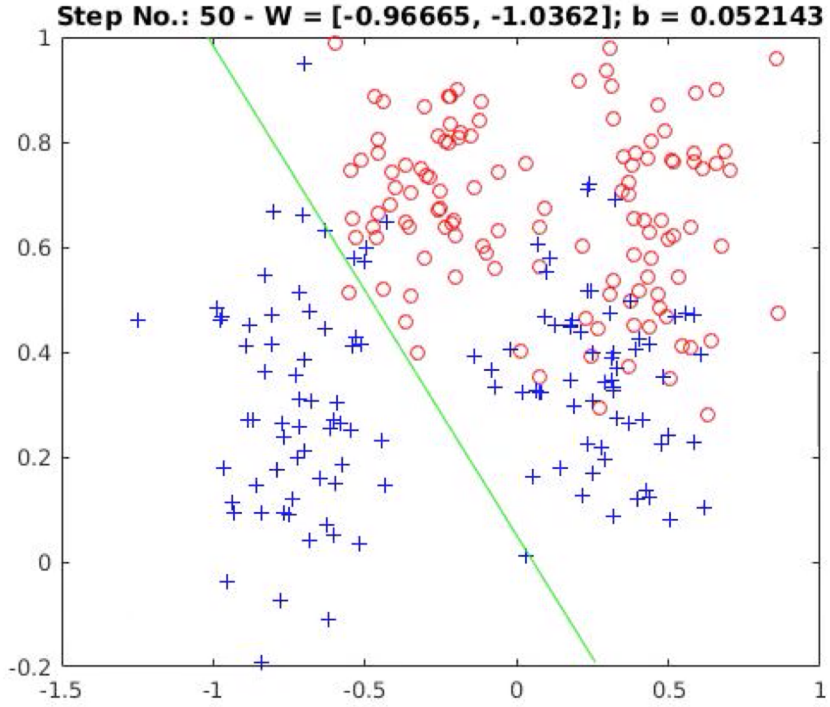
Intución ?

- Convergencia está garantizada (para problemas linealmente separables)

El perceptron en acción



El perceptron en acción

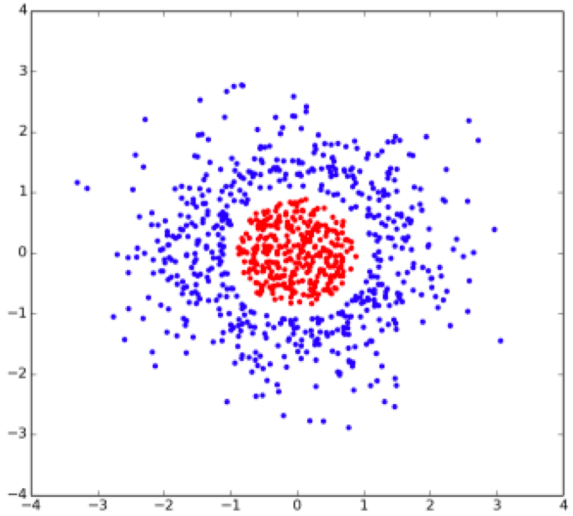


El perceptron

- Cómo estimar los pesos **w**?

- El algoritmo de aprendizaje del perceptron

1. **w** ← randomly initialize weights
2. Repeat until stop criterion meet
   I. For each $\mathbf{x}_i \in D$
      a) $o_i \leftarrow \mathbf{w}\mathbf{x}_i + b$ // estimate perc
      b) $\Delta\mathbf{w} \leftarrow \eta(y_i - o_i)$ // estimate the
      c) $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{w}$ //Update *w*
3. *Return* **w**

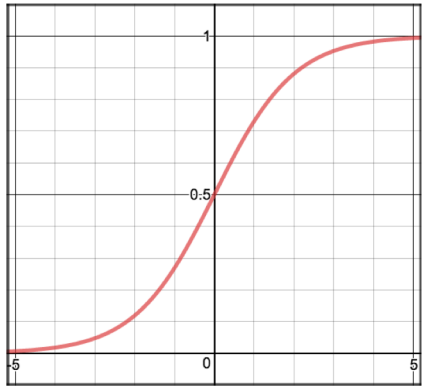


- Convergencia está garantizada (para problemas linealmente separables)

Qué pasa si el problema no es linealmente separable?

Regresión logística

- También podemos aprender clasificadores lineales con fundamento probabilista, estimando $p(y|\mathbf{x};\mathbf{w})$, con $y \in \{0,1\}$



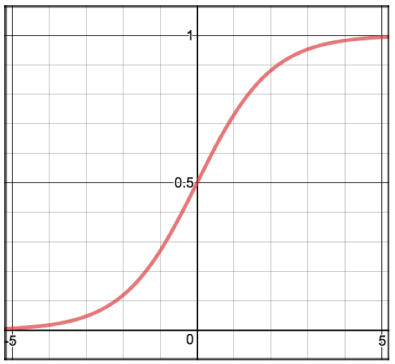
- Queremos estimar la probabilidad de que $p(y=1|\mathbf{x};\mathbf{w})$

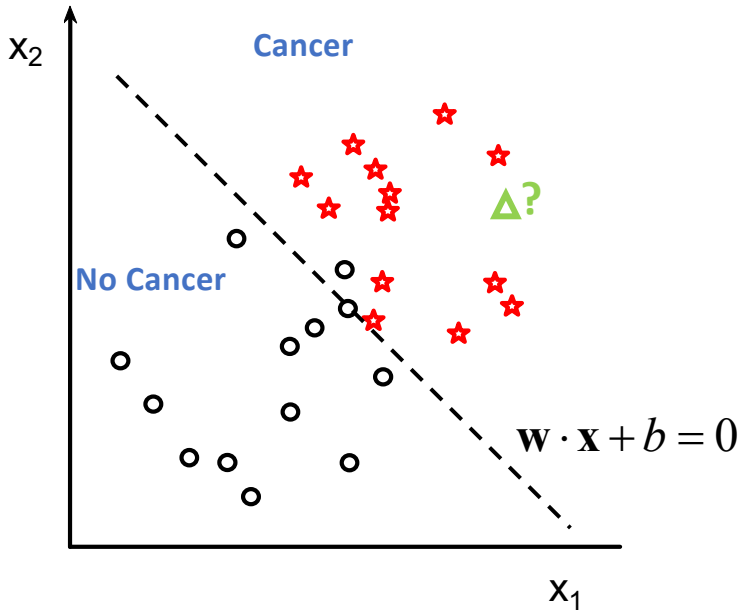
- Asumiendo un modelo lineal de regresión modificado por una sigmoide:
  - $p(y = 1|\mathbf{x}; \mathbf{w}) = \sigma(\mathbf{w}^T\mathbf{x})$

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

Regresión logística

- Asumiendo un modelo lineal de regresión modificado por una sigmoide:
  - $p(y = 1|\mathbf{x}; \mathbf{w}) = \sigma(\mathbf{w}^T\mathbf{x})$
- Regla de decisión: $f(\mathbf{x}, \mathbf{w}) = p(y|\mathbf{x}, \mathbf{w}) > 0.5$
- Asumiendo: $p(y = 1|\mathbf{x}, \mathbf{w}) = \text{Ber}(y|\sigma(\mathbf{w}^T\mathbf{x}))$

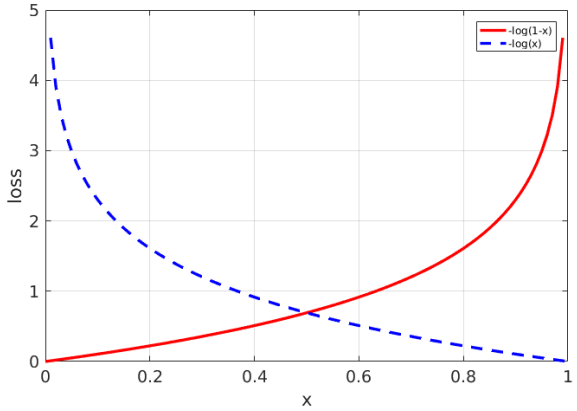




Regresión logística

$$f(k;p) = p^k(1-p)^{1-k} \quad \text{for } k \in \{0,1\}$$

- Aprendizaje de **w**?
- Se calcula el NLL de: $p(y|\mathbf{x},\mathbf{w}) = \text{Ber}\left(y\middle|\sigma\left(\mathbf{w}^T\mathbf{x}\right)\right)$
  - $\text{NNL}(\mathbf{w}) = -\sum_{i=1}^{m} \log[\sigma\left(\mathbf{w}^T\mathbf{x}\right)^{\mathbb{I}_{y_i=1}} \times (1-\sigma\left(\mathbf{w}^T\mathbf{x}\right))^{\mathbb{I}_{y_i=0}}]$
  - $\text{NNL}(\mathbf{w}) = -\sum_{i=1}^{m} \log\left(y_i \log\left(\sigma\left(\mathbf{w}^T\mathbf{x}\right)\right) + (1-y_i)\log\left(1-\sigma\left(\mathbf{w}^T\mathbf{x}\right)\right)\right)$



Regresión logística

$$f(k;p) = p^k(1-p)^{1-k} \quad \text{for } k \in \{0,1\}$$

- Aprendizaje de **w**?
- Se calcula el NLL de: $p(y|\mathbf{x},\mathbf{w}) = \mathrm{Ber}\left(y\middle|\sigma\left(\mathbf{w}^T\mathbf{x}\right)\right)$
  - $\mathrm{NNL}(\mathbf{w}) = -\sum_{i=1}^{m} \log[\sigma\left(\mathbf{w}^T\mathbf{x}\right)^{\mathbb{I}_{y_i=1}} \times (1-\sigma\left(\mathbf{w}^T\mathbf{x}\right))^{\mathbb{I}_{y_i=0}}]$
  - $\mathrm{NNL}(\mathbf{w}) = -\sum_{i=1}^{m} \log\left(y_i \log\left(\sigma\left(\mathbf{w}^T\mathbf{x}\right)\right) + (1-y_i)\log\left(1-\sigma\left(\mathbf{w}^T\mathbf{x}\right)\right)\right)$

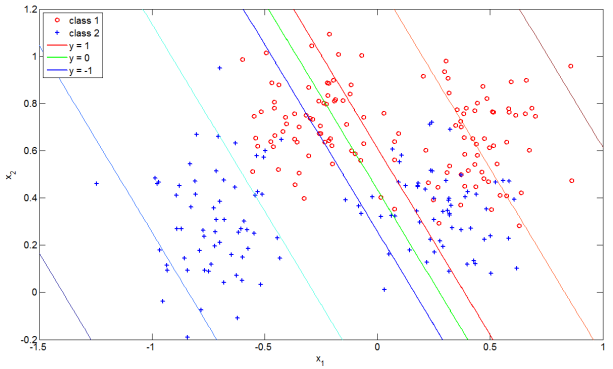
Cross entropy loss function

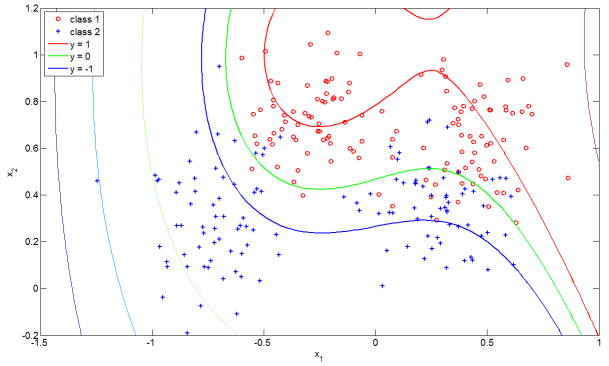
- Usar gradiente descendente u otro método para encontrar **w**

Unidades no lineales?

- Un problema con el perceptron es que solo pueden aprender funciones lineales. En datos no linealmente separables solo Podemos esperar un *buen ajuste!*

- Soluciones?
  - Mapear los datos a un espacio de mayor dimensionalidad donde el problema sea linealmente separable?
  - Construir modelos no lineales?
  - ?





Clasificadores lineales

- Únicamente aprenden funciones lineales (aunque pueden generalizarse a espacios no lineales, SVM, NNs)

- Eficientes, interpretables (algunos)

Extensiones?

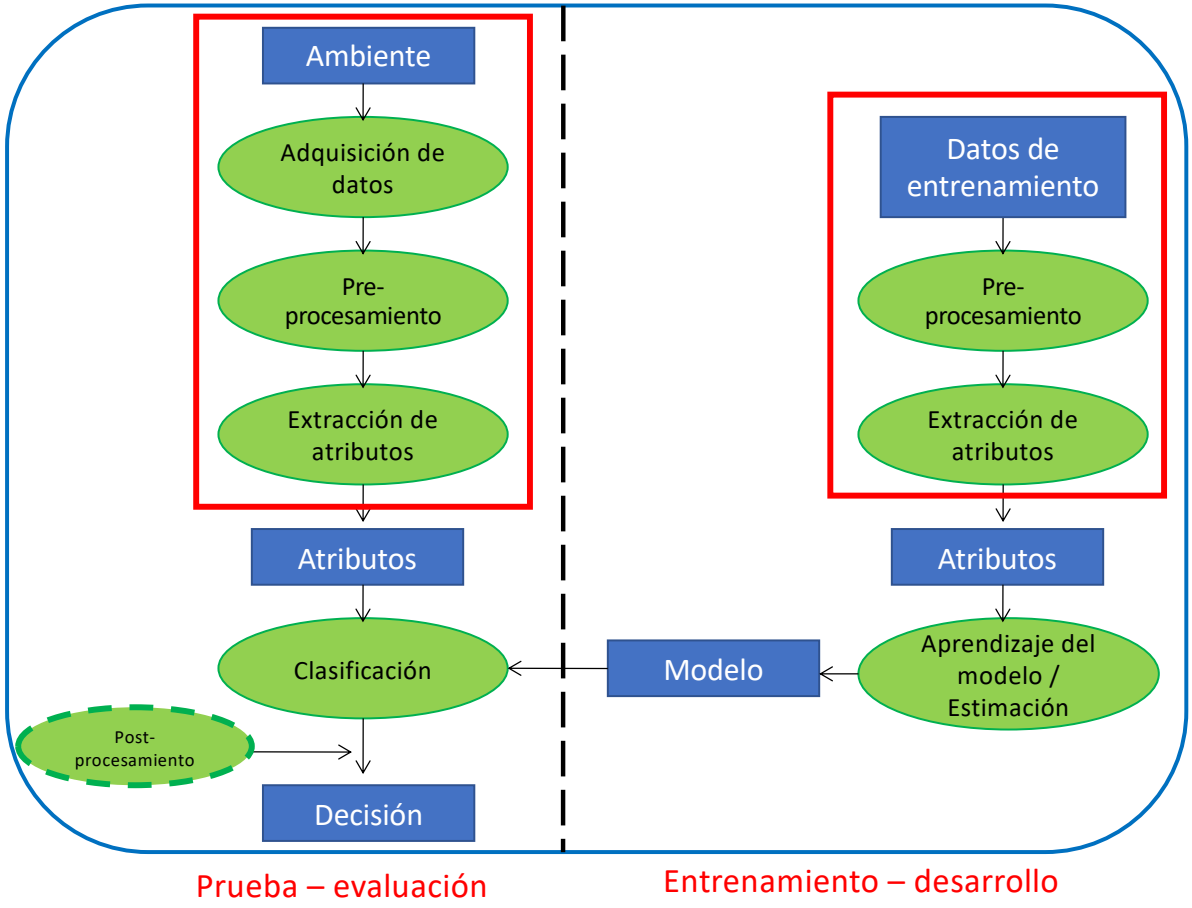
- Extensión multi clase? Multi etiqueta?
- Dimensionalidad y eficiencia?
- Ruido en los datos?

Principales métodos de clasificación

- Árboles de decisión
  - Intuitivo, simple, genera reglas
- Clasificador Bayesiano Ingenuo
  - Probabilista
- K-Vecinos más cercanos
  - Basado en similitudes, ejemplos
- Máquinas de soporte vectorial
  - Basado en kernels
- Ensambles
  - Combinación de clasificadores individuales
- Redes neuronales
  - SoTA

El proceso de diseño de un sistema de reconocimiento de patrones

R. O. Duda, P. Hart, D. Stork.
Pattern Classification.
Wiley, 2001.



Recap.

- Introducción al aprendizaje supervisado
  - Requieren ejemplos etiquetados
  - Aprenden a mapear entradas a salidas deseadas
- Tareas comunes: regresión, clasificación
- Objetos se describen por vectores
- Modelos lineales

Preguntas