EL ALGORITMO PERCEPTRÓN

Dr. Jorge Hermosillo
Laboratorio de Semántica Computacional

Centro de Investigación en Ciencias

Universidad Autónoma del Estado de Morelos

PERCEPTRON

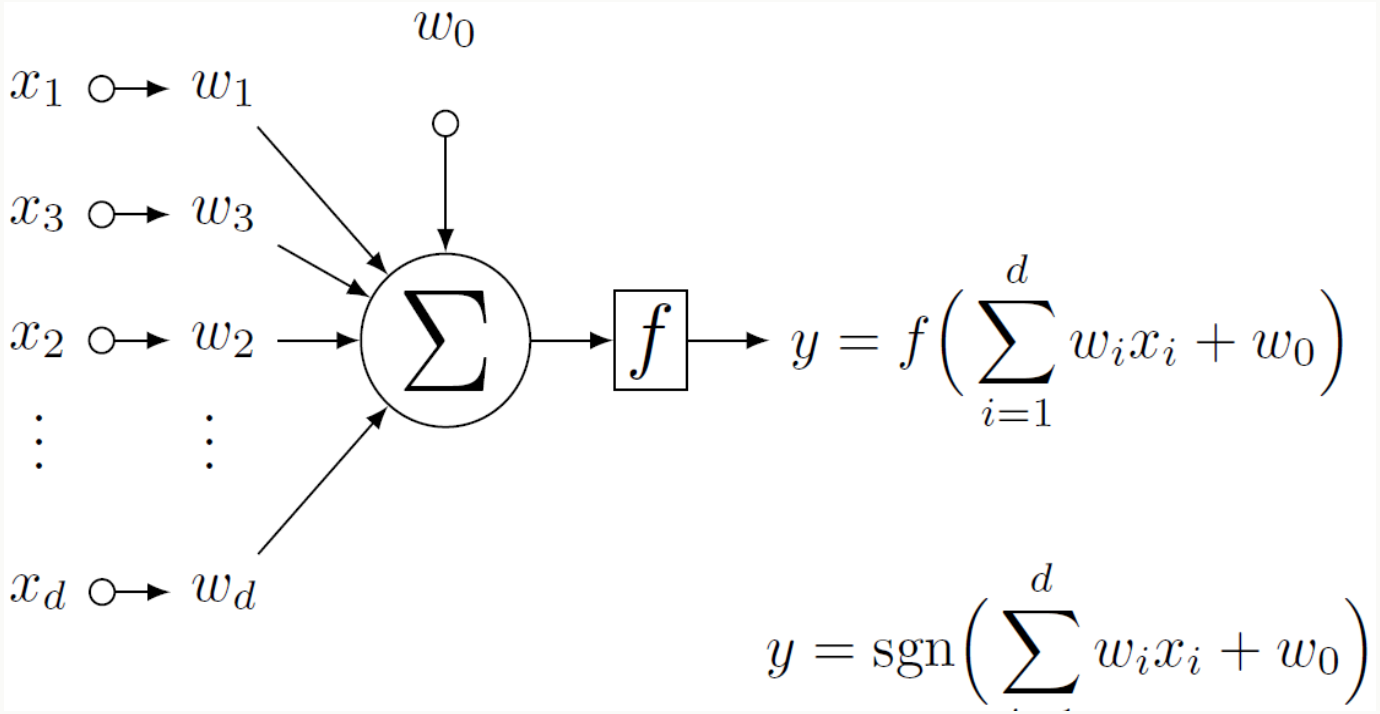
Modelo Lineal de clasificación entrenado por descenso de gradiente

NEURONA DE McCULLOCH-PITTS

Linear Threshold Unit (LTU)
1943

Warren McCulloch

Walter Pitts

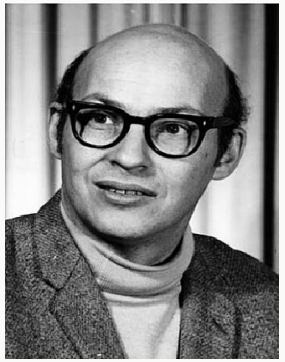


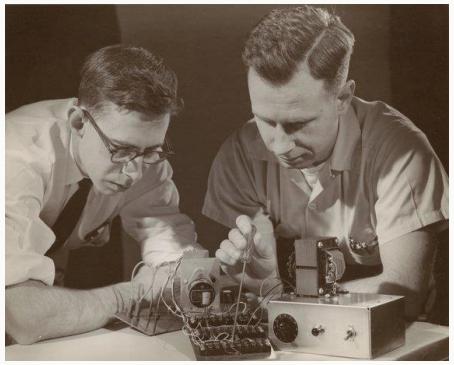
W. S. McCulloch y W. Pitts, "A logical calculus of the ideas immanent in neurons activity", Bull.Math. Biophys., 5, 115-133, (1943).

ALGORITMO PERCEPTRÓN

Frank Rosenblatt
1928 – 1971

Marvin Minsky
1927 – 2016





El **Algoritmo Perceptrón** (Rosenblatt, 1961) jugó un rol muy importante en la historia del Machine Learning. Inicialmente, Rosenblatt simuló el perceptrón en una computadora IBM 704 en Cornell en 1957, pero para principios de 1960, había diseñado un hardware de propósito específico, que implementaba directamente y de forma paralela el algoritmo de aprendizaje. Su trabajo fue duramente criticado por Marvin Minsky, quien publicó un libro mostrando las deficiencias de las redes de perceptrón de una sola capa para resolver problemas no separables. Desafortunadamente, Minsky conjeturó (equivocadamente) que modelos más generales de redes neuronales padecían de las mismas deficiencias, lo que propició un vacío en la investigación de la computación neural que duró hasta mediados de la década de 1980.

Rosenblatt, F. (1961). *Principles of neurodynamics. perceptrons and the theory of brain mechanisms* (No. VG-1196-G-8). Cornell Aeronautical Lab Inc Buffalo NY.

ALGORITMO PERCEPTRÓN (EJEMPLO)

Ejemplo: ***Aprobación de Tarjeta de Crédito***

CONCEPTO	DATO	Feature
EDAD	25	x_1
TIEMPO DE VIVIR EN DOM.	5	x_2
SALARIO	20,000.00	x_3
		$\vdots$
DEUDA	30,000.00	x_d

DECISIÓN (SALIDA) $y = \begin{cases} +1 \\ -1 \end{cases}$

HIPÓTESIS DE SEPARACIÓN LINEAL

Para una entrada $\mathbf{x} = (x_1, \ldots, x_d)$ "atributos de un cliente"

Aprobar la TC si $\sum_{i=1}^{d} w_i x_i > \text{umbral}$

Denegar la TC si $\sum_{i=1}^{d} w_i x_i < \text{umbral}$

Esta fórmula lineal $h \in \mathcal{H}$ puede escribirse como:

$$h(\mathbf{x}) = \text{sign}\left(\left(\sum_{i=1}^{d} w_i x_i\right) - \text{umbral}\right)$$

HIPÓTESIS DE SEPARACIÓN LINEAL

Para una entrada $\mathbf{x} = (x_1, \ldots, x_d)$ "atributos de un cliente"

Aprobar la TC si $\sum_{i=1}^{d} w_i x_i > \text{umbral}$

Denegar la TC si $\sum_{i=1}^{d} w_i x_i < \text{umbral}$

Esta fórmula lineal $h \in \mathcal{H}$ puede escribirse como:

$$h(\mathbf{x}) = \text{sign}\left(\left(\sum_{i=1}^{d} w_i x_i\right) + \boldsymbol{w_0}\right) = \text{sign}\left(\tilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}}\right)$$

PRINCIPIO DEL APRENDIZAJE

El perceptrón implementa:

$$h(\mathbf{x}) = \text{sign}\left(\mathbf{w}^{\text{T}}\mathbf{x}\right)$$

Dado el conjunto de entrenamiento:

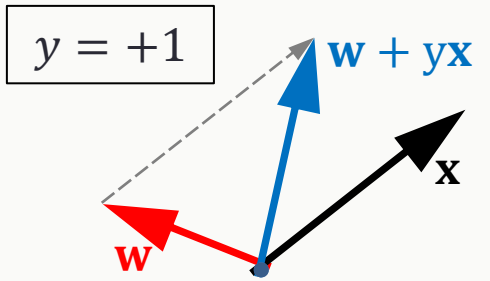
$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N)$$

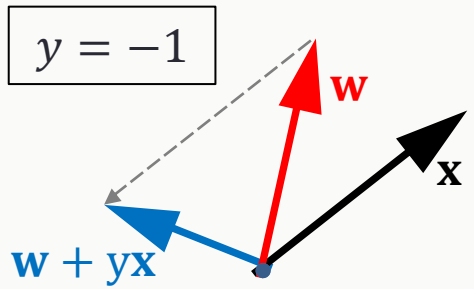
Toma un punto mal clasificado:

$$\text{sign}\left(\mathbf{w}^{\text{T}}\mathbf{x}_j\right) \neq y_j$$

y actualiza el vector de pesos:

$$\mathbf{w} \leftarrow \mathbf{w} + y_j\mathbf{x}_j$$





PRINCIPIO DEL APRENDIZAJE

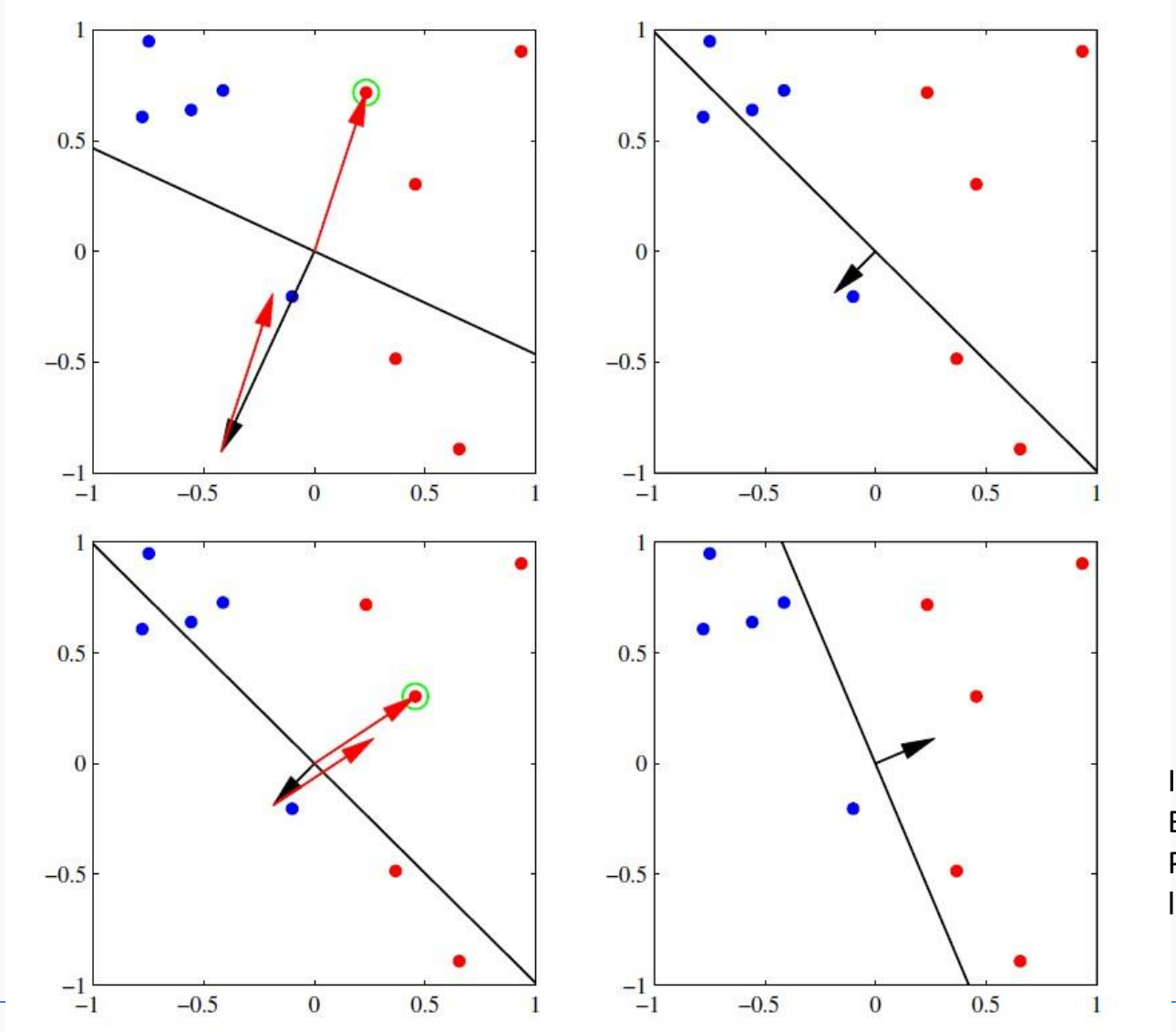


Imagen tomada de:
Bishop, Christopher M. (2006). Pattern recognition and machine learning. New York. Springer.

RESUMEN DEL ALGORITMO PERCEPTRON

- Datos de entrada:
  - $\mathbf{X} \coloneqq \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \subset \mathcal{X}(\text{datos}), \text{con } \mathbf{x}_i \coloneqq (x_1, x_2, \cdots, x_d) \subset \mathbb{R}^d$
  - $\mathbf{y} \coloneqq \{y_1, y_2, \cdots, y_N\} \subset \{\pm 1\}(\text{etiquetas})$
  - *En coordenadas homogéneas:*
    $$\mathbf{X} \coloneqq \{(1, \mathbf{x}_1), (1, \mathbf{x}_2), \cdots, (1, \mathbf{x}_N)\} \subset \mathbb{R}^N \times \mathbb{R}^{d+1}$$
- Parámetros:
  - $\mathbf{w} \coloneqq (w_1, w_2, \cdots, w_d) \subset \mathbb{R}^d$ *(vector de pesos)*
  - $w_0 = -u \in \mathbb{R}$ *(umbral)*
- Función (modelo lineal) en coord. homegéneas:
  - $y_i = \text{sign}(\langle \mathbf{w}, \mathbf{x}_i \rangle)$
  - $\mathbf{x}_i \coloneqq (1, x_1, x_2, \cdots, x_d) \subset \mathbb{R}^{d+1}$,
  - $\mathbf{w} \coloneqq (w_0, w_1, w_2, \cdots, w_d) \subset \mathbb{R}^{d+1}$

ALGORITMO DE APRENDIZAJE PERCEPTRON

Entrada: *Datos etiquetados de entrenamiento* $\mathbf{X}$ *en coordenadas homogéneas*

Salida: *Vector de pesos* $\mathbf{w}$ *que define al clasificador*

$\mathbf{w} = \mathbf{0}$ *#Otras inicializaciones son posibles*

$converge = \boldsymbol{Falso}$

mientras $converge == \boldsymbol{Falso}$:

    $converge = \boldsymbol{Verdadero}$

    para i **en** $|\mathbf{X}|$:

        si $y_i \langle \mathbf{w}, \mathbf{x}_i \rangle \leq 0$ **entonces**: #xi mal clasificado

            $\boldsymbol{w = w + y_i \eta x_i}$ #$0 < \eta \leq 1$ es la tasa de aprendizaje

            $converge = \boldsymbol{Falso}$

        fin

    fin

fin

Clasificador lineal “básico”

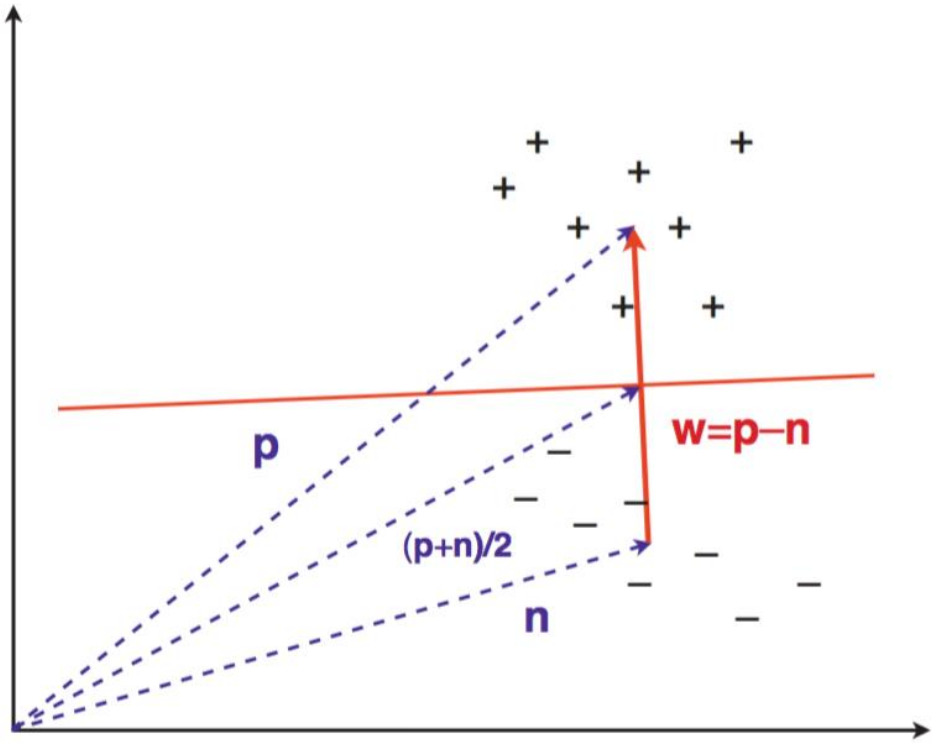


Figure 1.1. The basic linear classifier constructs a decision boundary by half-way intersecting the line between the positive and negative centres of mass. It is described by the equation $\mathbf{w} \cdot \mathbf{x} = t$, with $\mathbf{w} = \mathbf{p} - \mathbf{n}$; the decision threshold can be found by noting that $(\mathbf{p} + \mathbf{n})/2$ is on the decision boundary, and hence $t = (\mathbf{p} - \mathbf{n}) \cdot (\mathbf{p} + \mathbf{n})/2 = (||\mathbf{p}||^2 - ||\mathbf{n}||^2)/2$, where $||\mathbf{x}||$ denotes the length of vector $\mathbf{x}$.

Imagen tomada de:
Flach, Peter (2012). Machine Learning: The Art and Science of Algorithms that Make Sense of Data. Cambridge University Press.

PERCEPTRON DUAL

Modelo dual de ponderación de instancias de entrenamiento

ALGORITMO DE APRENDIZAJE PERCEPTRON

Nota que el perceptrón básico aprende los pesos de los atributos

$$\begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix} = y_1 \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}_1 + y_2 \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}_2 + \cdots + y_N \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}_N \Longrightarrow \mathbf{w} = \sum_{i=1}^{N} y_i x_i$$

Luego del entrenamiento cada ejemplo estuvo mal clasificado 0 o α veces (en función del número de épocas).

$$\mathbf{w} = \sum_{j=1}^{N} \alpha_j y_j \mathbf{x}_j$$

PERCEPTRON DUAL

- El vector de pesos es una combinación lineal de puntos de entrenamiento:

$$\mathbf{w} = \sum_{j=1}^{N} \alpha_j y_j \mathbf{x}_j$$

- En este caso la salida del clasificador $y_i = \text{sign}(\langle \mathbf{w}, \mathbf{x}_i \rangle)$ se escribe:

$$y_i = \text{sign}\left(\sum_{j=1}^{N} \alpha_j y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \right)$$

PERCEPTRON DUAL

$$y_i = \text{sign}\left(\sum_{j=1}^{N} \alpha_j y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle\right)$$

- Es decir que podemos aprender los pesos α de las instancias $\mathbf{x_i}$, en lugar de los pesos de los atributos.

- Por lo tanto la única información necesaria es la matriz n-por-n $\mathbf{G} = \mathbf{X}\mathbf{X}^{\mathrm{T}}$, llamada *matriz Gram*, que contiene todos los productos punto.

ALGORITMO DEL PERCEPTRON DUAL

$\boldsymbol{\alpha} \coloneqq (\alpha_1, \alpha_2, \cdots, \alpha_N) = 0$

$converge = \boldsymbol{Falso}$

mientras $converge == \boldsymbol{Falso}$:

$converge = \boldsymbol{Verdadero}$

para i **en** $|\mathbf{X}|$:

toma $(\mathbf{x}_i, y_i)$ de los datos

si $\mathrm{y_i} \sum_{j=1}^{|\mathbf{X}|} \alpha_j y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \leq 0$ **entonces**: #xi mal clasificado

$\alpha_i = \alpha_i + 1$

$converge = \boldsymbol{Falso}$

fin

fin

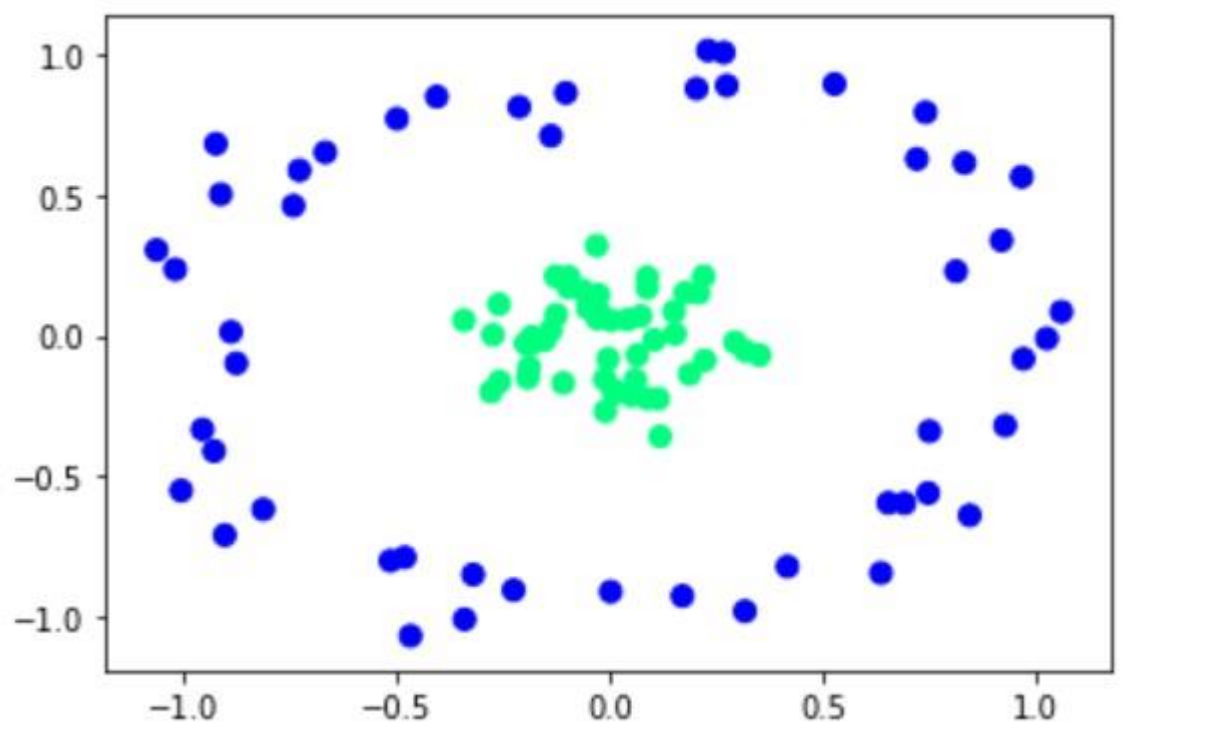
fin

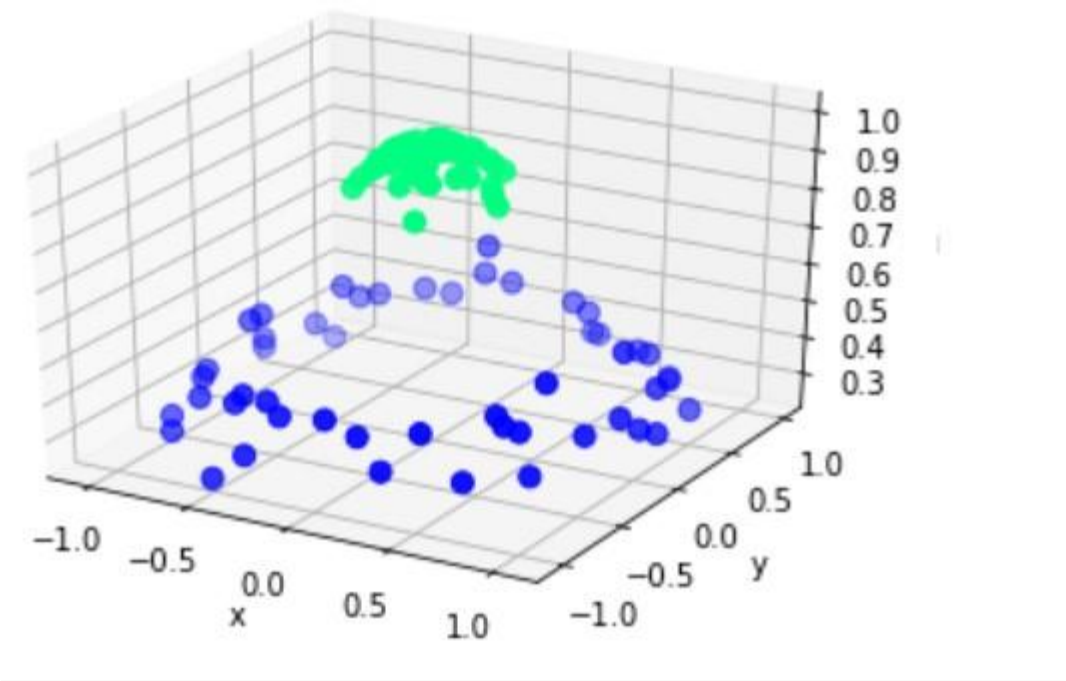
THE KERNEL TRICK

Transformación del espacio de features

Mas allá de la clasificación lineal: métodos de Kernel

$$Z = \exp(-\Sigma X^2)$$





TÉRMINOS EMPLEADOS

- Kernel: en términos generales un Kernel es un factor multiplicativo en una sumatoria o integral.
- En nuestro contexto un Kernel es una transformación de los datos que debe cumplir con ciertas propiedades específicas.
- El espacio de nuevos datos (transformados) se le llama comúnmente espacio de atributos ($feature\ space$), en contraste con el espacio original o espacio de entrada ($input\ space$).

PRODUCTO PUNTO EN EL ESPACIO CUADRÁTICO

$$\mathbf{x}_i = (x_i, y_i) \; ; \; \mathbf{x}_j = (x_j, y_j) \quad \rightarrow \langle \mathbf{x}_i, \mathbf{x}_j \rangle = x_i x_j + y_i y_j$$

- Instancias en el espacio de atributos al cuadrado:

$$\mathbf{x}_i^2 = (x_i^2, y_i^2) \; ; \; \mathbf{x}_j^2 = (x_j^2, y_j^2)$$

- Producto punto de estas instancias:

$$\langle \mathbf{x}_i^2, \mathbf{x}_j^2 \rangle = x_i^2 x_j^2 + y_i^2 y_j^2$$

- Casi igual a:

$$\langle \mathbf{x}_i, \mathbf{x}_j \rangle^2 = x_i^2 x_j^2 + y_i^2 y_j^2 + 2 x_i x_j y_i y_j$$

KERNEL CUADRÁTICO COMO UN PRODUCTO PUNTO

- Para que sean iguales, podemos transformar los datos agregando un tercer atributo: $\sqrt{2}xy$

$$\phi(\mathbf{x}_i) = \left(x_i^2, y_i^2, \sqrt{2}x_i y_i\right) \; ; \; \phi(\mathbf{x}_j) = \left(x_j^2, y_j^2, \sqrt{2}x_j y_j\right)$$

- Calculando ahora el producto punto:

$$\left\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \right\rangle = x_i^2 x_j^2 + y_i^2 y_j^2 + 2x_i x_j y_i y_j = \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle^2$$

- Definimos ahora:

$$\kappa\left(\mathbf{x}_i, \mathbf{x}_j\right) = \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle^2$$

- Sustituimos $\kappa\left(\mathbf{x}_i, \mathbf{x}_j\right)$ en lugar de $\left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle$ en el algoritmo del perceptrón DUAL.

ALGORITMO DEL PERCEPTRON DUAL

$\boldsymbol{\alpha} \coloneqq (\alpha_1, \alpha_2, \cdots, \alpha_N) = 0$

$converge = \boldsymbol{Falso}$

mientras $converge == \boldsymbol{Falso}$ **:**

$converge = \boldsymbol{Verdadero}$

para i **en** $|\mathbf{X}|$:

toma $(\mathbf{x}_i, y_i)$ de los datos

si $\mathrm{y_i} \sum_{j=1}^{|\mathbf{X}|} \alpha_j y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \leq 0$ **entonces**: #xi mal clasificado

$\alpha_i = \alpha_i + 1$

$converge = \boldsymbol{Falso}$

ALGORITMO DEL PERCEPTRON DUAL

$\boldsymbol{\alpha} \coloneqq (\alpha_1, \alpha_2, \cdots, \alpha_N) = 0$; $\boldsymbol{\kappa(\mathbf{x_i}, \mathbf{x_j})}$: *entrada*

$converge = \boldsymbol{Falso}$

mientras $converge == \boldsymbol{Falso}$:

  $converge = \boldsymbol{Verdadero}$

  para i **en** $|\mathbf{X}|$:

    toma $(\mathbf{x}_i, y_i)$ de los datos

    si $\mathrm{y_i} \sum_{j=1}^{|\mathbf{X}|} \alpha_j y_j \boldsymbol{\kappa(\mathbf{x_i}, \mathbf{x_j})} \leq 0$ **entonces**: #xi mal clasificado

      $\alpha_i = \alpha_i + 1$

      $converge = \boldsymbol{Falso}$

KERNEL POLINOMIAL Y GAUSSIANO

- Con base en lo anterior podemos definir kernels de grado p > 2, por ejemplo:

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle^p$$

- En general podemos definir un kernel polinomial:

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\langle \mathbf{x}_i, \mathbf{x}_j \rangle + c)^p, c \geq 0$$

- No cualquier función puede ser utilizada como kernel
- Nota que: $\kappa(\mathbf{x}, \mathbf{x}) = \langle \phi(\mathbf{x}), \phi(\mathbf{x}) \rangle = \|\phi(\mathbf{x})\|^2$ para cualquier kernel que cumpla con ciertas propiedades referidas como transformaciones "semidefinidas positivas".

KERNELS BÁSICOS

- Aunque nuevos kernels aparecen en la literatura, los siguientes cuatro son básicos y ampliamente utilizados:

Lineal:	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$
Polinomial:	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\langle \mathbf{x}_i, \mathbf{x}_j \rangle + r)^p, r \geq 0$
Gaussiano (*Radial Basis Function* – RBF):	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) = \exp\left(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2\right)$
Sigmoide:	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\gamma \langle \mathbf{x}_i, \mathbf{x}_j \rangle + r)$

donde r, p, γ son parámetros de los modelos.

DESCENSO DE GRADIENTE EN EL PERCEPTRÓN: LA FUNCIÓN DELTA