Universidad Nacional Autónoma de México
Facultad de Ingeniería

Máquinas de soporte vectorial

(SVM, Support Vector Machines)

Guillermo Molero-Castillo
guillermo.molero@ingenieria.unam.edu

Enero, 2021

Clasificación

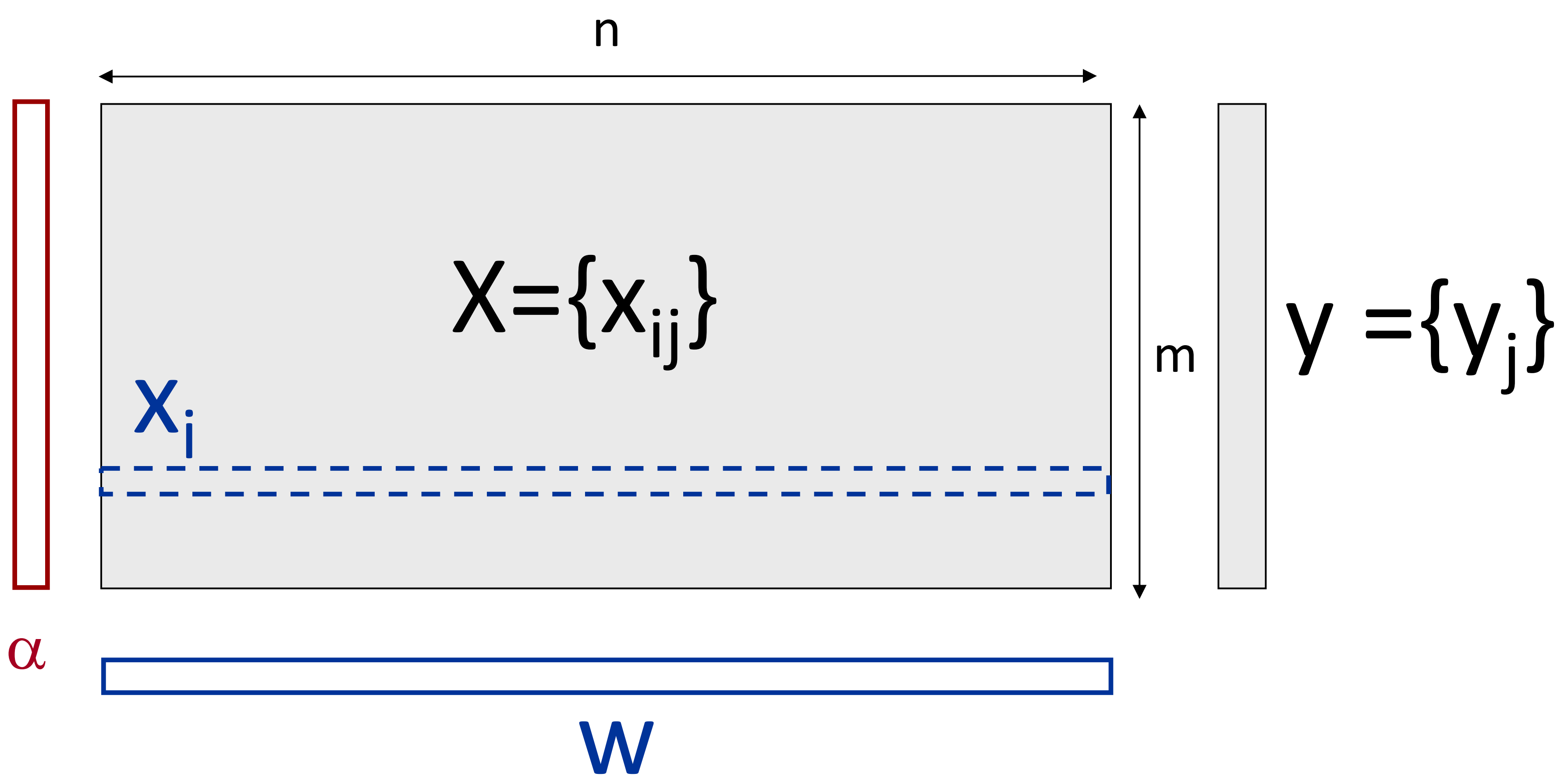
Binaria: Cada elemento puede pertenecer a una de dos clases.

$$f: \rightarrow \{-1,1\}$$

Multiclase : Cada elemento puede pertenecer a una de las K clases.

$$f: \rightarrow \{1,\ldots,K\}$$

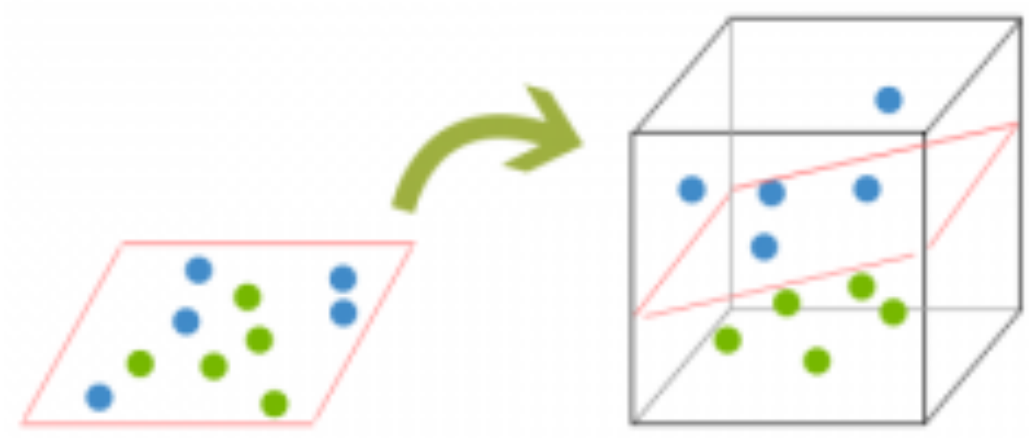
Clasificación



Proyección en el plano (vector)

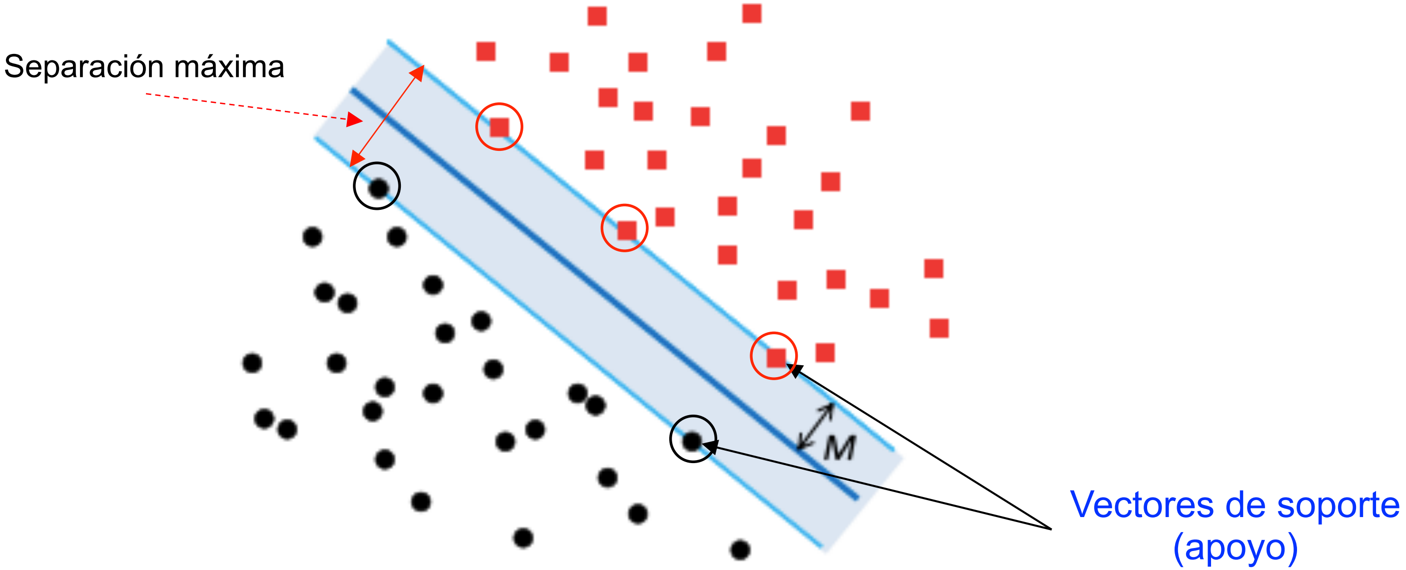
Máquinas de soporte vectorial

- **SVM**, tal como se entiende actualmente, fue presentado en la conferencia COLT (*COmputational Learnig Theory*) por Vapnik, Boser y Guyon en 1992.
- Posteriormente, fue descrito con mayor detalle en 1995 (Cortes y Vapnik) y 1998 (Vapnik) para pasar de la formulación teórica a su aplicación práctica en problemas reales de reconocimiento de patrones (*pattern recognition*).
- En la actualidad, el interés por este algoritmo no ha dejado de crecer en su aplicación a problemas reales. Se utiliza tanto para clasificación como para pronóstico (regresión).



Máquinas de soporte vectorial

- Hoy en día, **SVM** constituye un referente en el aprendizaje automático, sobre todo para encontrar la manera óptima de resolver problemas de clasificación.
- Intuitivamente, una SVM es un modelo que representa los puntos en el espacio, separando las clases en dos (espacios) mediante un hiperplano.
- Se busca la **separación máxima posible** entre los puntos más cercanos a cada clase (vectores de soporte).



Características

- Es un algoritmo de clasificación para datos lineales y no lineales.
- Utiliza un mapeo no lineal para transformar los datos de entrenamiento en una dimensión superior.
- Con la nueva dimensión, se busca el hiperplano de separación lineal óptimo.
- Con un mapeo no lineal apropiado, los datos pueden ser separados por un hiperplano.
- SVM encuentra este hiperplano usando **vectores de soporte** (datos de entrenamiento) para medir los **márgenes de separación**.

Principales ideas

- **Clasificador de margen máximo (Max-Margin Classifier)**

  Formalizar la noción del mejor separador lineal.

- **Multiplicadores de Lagrange (Lagrangian Multipliers)**

  Es la manera de convertir un problema de optimización en uno de posible solución.

- **Kernels**

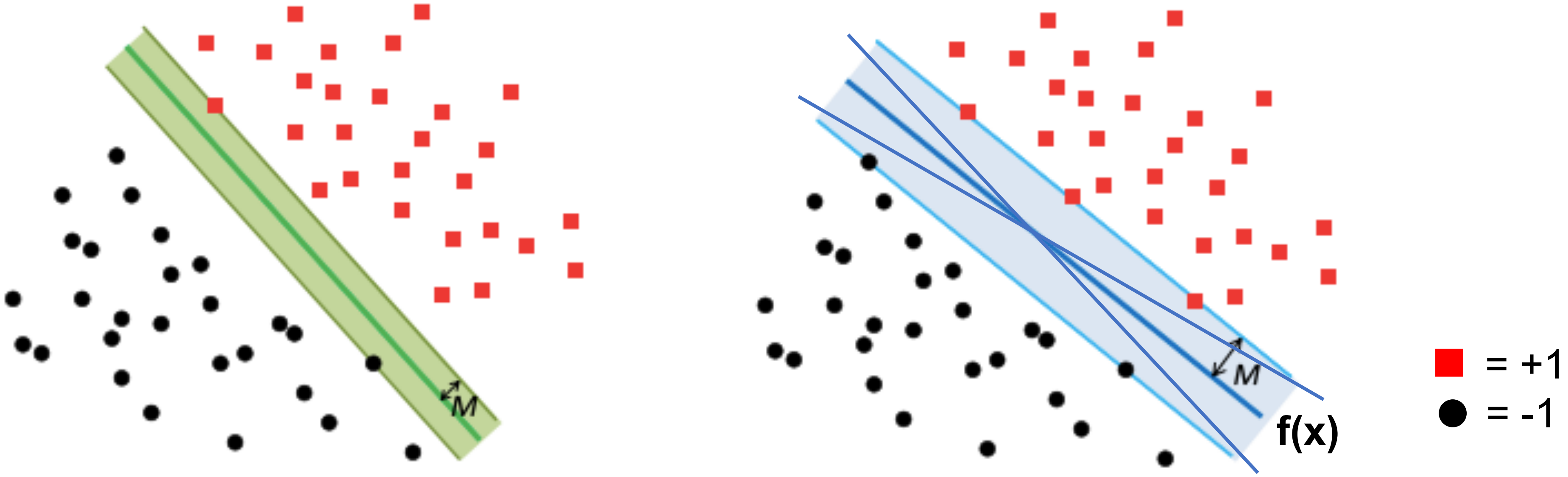
  Proyectar los datos en un espacio de dimensiones superiores para hacerlos separables.

- **Complejidad**

  Depende de la cantidad de ejemplos de entrenamiento, no de la dimensionalidad del espacio del kernel.

Fundamentos

Asumiendo un conjunto de datos **D $(\mathbf{x}_i, \boldsymbol{y}_i)_{i=1...n}$** asociados a una etiqueta de clase $y_i \in \{-1, 1\}$, separables mediante un hiperplano.



Separación reducida

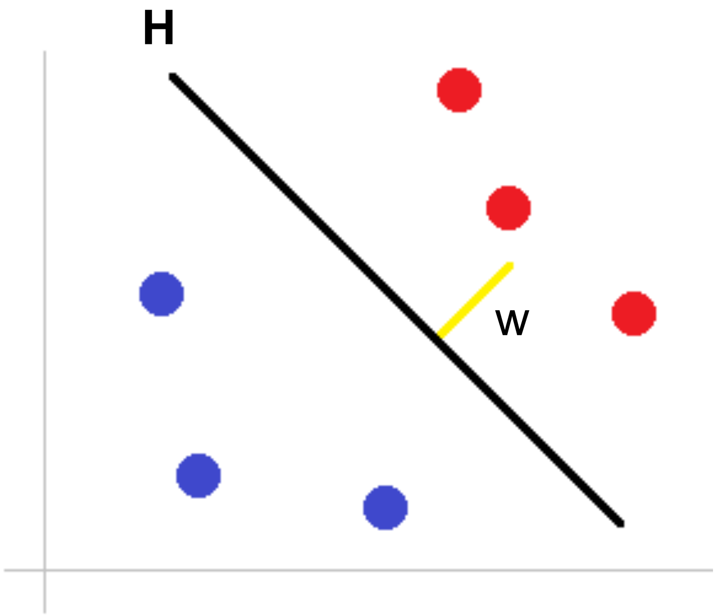
Separación máxima

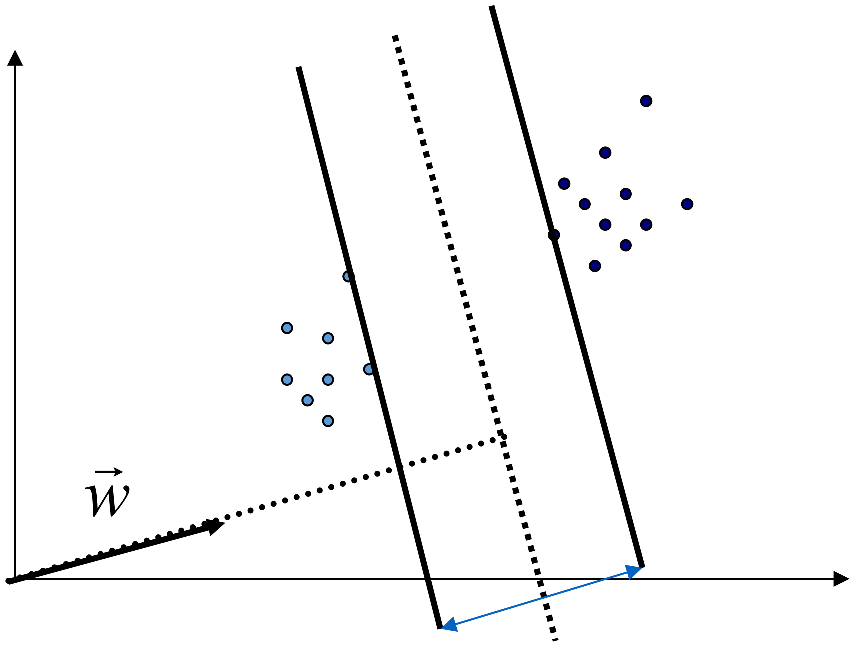
- Pueden existir diversas líneas (hiperplanos) que separan las dos clases, pero se debe encontrar el hiperplano óptimo (maximiza el margen de separación).

Fundamentos

Para caracterizar al hiperplano separador se utiliza el vector **(w)** y la ordenada al origen **(b)** – constante– (Recordar la ecuación de un hiperplano en álgebra):

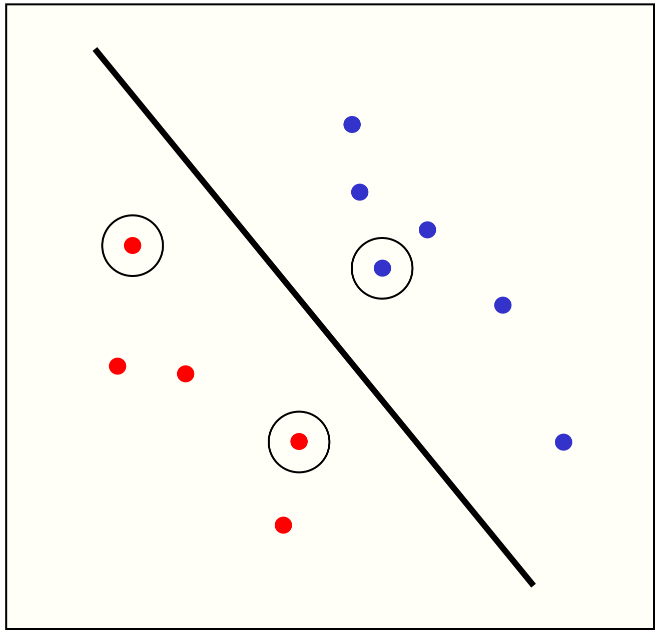
$$\boldsymbol{wx} + \boldsymbol{b} = \ \mathbf{0}$$





Vectores de soporte

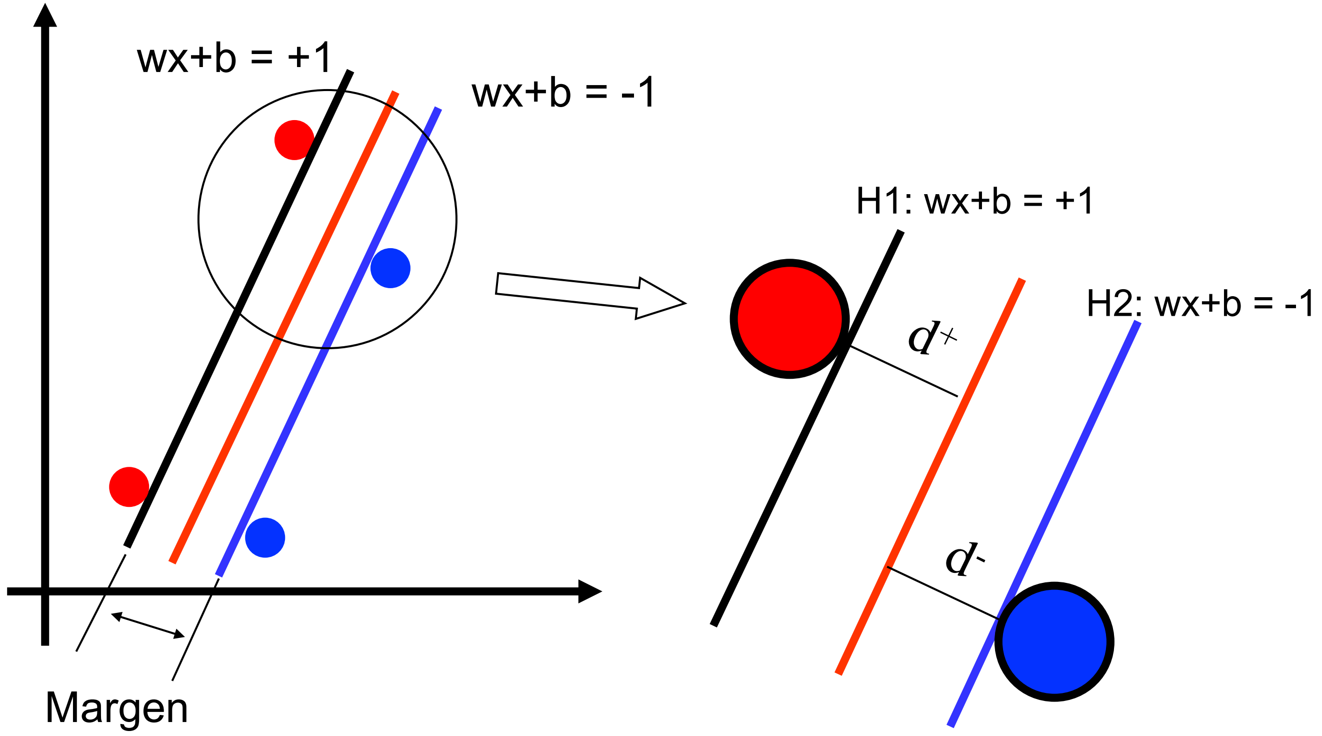
- El **separador de margen máximo** está determinado por un subconjunto de puntos de datos (vectores de soporte).
- Computacionalmente es útil una pequeña fracción de puntos (vectores de soporte), dado que éstos se utilizan para decidir en qué lado del separador se clasificará cada caso de prueba.



$$Y = sgn[wx + b > 0] = \begin{cases} +1 & wx + b > 0 \\ -1 & wx + b \leq 0 \end{cases}$$

Distancia

- Intuitivamente, el mejor hiperplano debe estar situado en la posición más neutra posible con respecto a las clases.
- Se considera los puntos que están en la frontera de la región de decisión, dado que es la zona donde puede haber dudas sobre a qué clase pertenece el elemento.

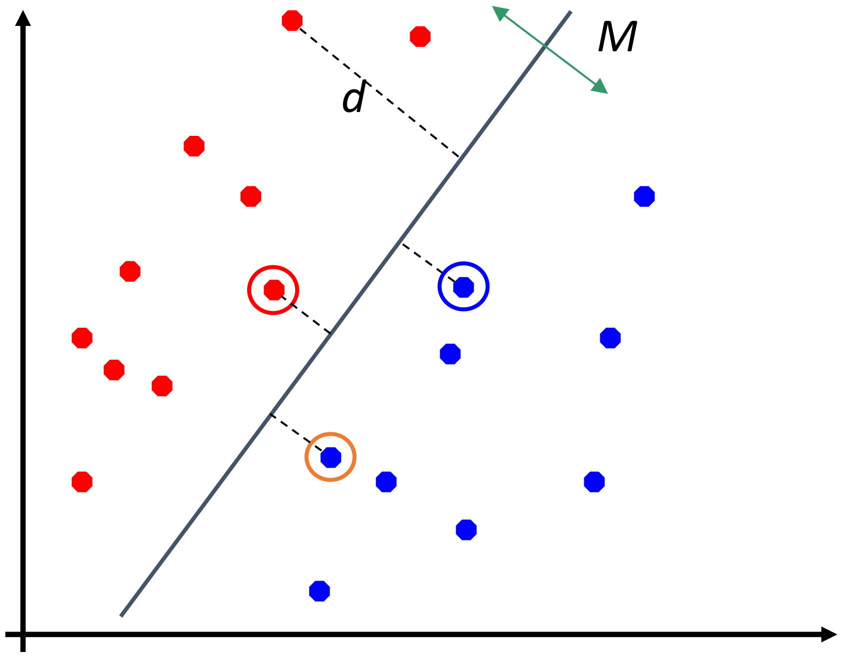


d^+ = la distancia más corta al punto positivo más cercano.
d^- = la distancia más corta al punto negativo más cercano.
El **margen** de un hiperplano de separación es **d^+ + d^-**.

Distancia

La distancia del ejemplo x_i al separador $\mathrm{H}(\boldsymbol{w}, \boldsymbol{b})$ es: $d = \frac{\mathbf{w}\boldsymbol{x_i} + b}{||\mathbf{w}||} = \frac{\pm\mathbf{1}}{||\mathbf{w}||}$ longitud euclidiana $\sqrt{\boldsymbol{w} \bullet \boldsymbol{w}}$

El **margen** del separador es la distancia entre los vectores de soporte: $M = \left|\frac{\mathbf{1}}{||\mathbf{w}||} - \frac{\mathbf{-1}}{||\mathbf{w}||}\right| = \frac{\mathbf{2}}{||\mathbf{w}||}$



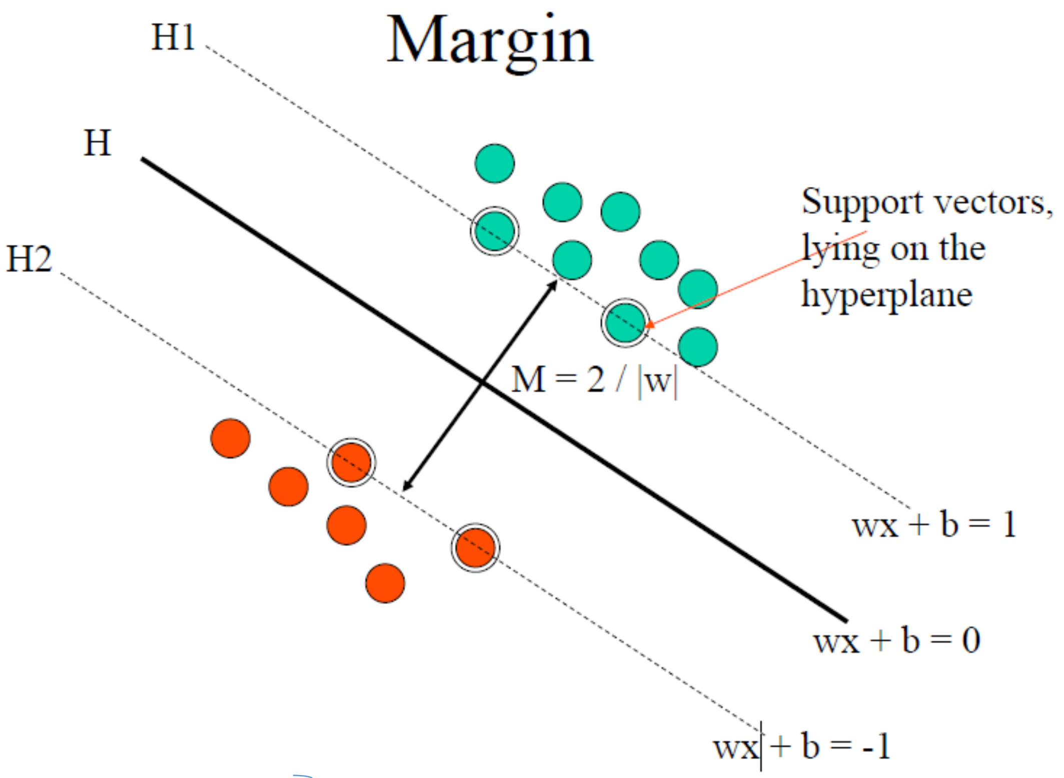
Formalización

- Todos los hiperplanos de separación están expresados por:

$$wx + b = 0$$

  donde: $w = \{w_1, w_2, \dots, w_n\}$ es es un vector y b una constante.

- El objetivo es encontrar el hiperplano $f(x) = wx + b$ que mejor clasifique los datos (la que minimiza el error de clasificación).
- Entonces, **SVM** busca el hiperplano con el margen más grande, conocido como hiperplano marginal máximo (MMH).



$y_i \in \{+1, -1\}$ Positivos / Negativos

Formalización

- Sea un conjunto de entrenamiento $\{(\boldsymbol{x_i}, y_i)\}_{i=1,\dots n}$, $\boldsymbol{x_i} \in \mathrm{R}^d$, $y_i \in \{-1, 1\}$ sea separado por un hiperplano con margen **M**. Entonces para cada elemento del entrenamiento $(\boldsymbol{x_i}, y_i)$:

$$\begin{aligned} &\mathrm{H}_1: \quad \mathbf{w}\boldsymbol{x_i} + b \geq +1 \quad \text{si } \boldsymbol{y_i} = +1 \\ &\mathrm{H}_2: \quad \mathbf{w}\boldsymbol{x_i} + b \leq -1 \quad \text{si } \boldsymbol{y_i} = -1 \end{aligned} \quad \Longleftrightarrow \quad y_i(\mathbf{w}\boldsymbol{x_i} + b) \geq +1$$

- Para cada vector de soporte $\boldsymbol{x_s}$, en la desigualdad anterior, se reescala $\mathbf{w}$ y b, obteniéndose la distancia entre cada $\boldsymbol{x_s}$ y el hiperplano:

$$d = \frac{\boldsymbol{y_s}(\mathbf{w}\boldsymbol{x_s}+b)}{||\mathbf{w}||} = \frac{\mathbf{1}}{||\mathbf{w}||}$$

- Entonces el margen se puede expresar mediante $\mathbf{w}$ y b como: $M = 2d = \dfrac{\mathbf{2}}{||\mathbf{w}||}$

Formalización

- Con base a lo anterior, se puede formular el problema de optimización cuadrática:

  Para $\mathbf{w}\ y\ b$, tales que: $M = \frac{\mathbf{2}}{\|\mathbf{w}\|}$ es maximizada, y para todo $(\boldsymbol{x_i},\ y_i), i = 1, \dots, n$:

$$y_i(\mathbf{w}\boldsymbol{x_i} + b) \ \geq\ +1$$

- La necesidad de optimizar una función cuadrática, sujeta a restricciones lineales, es un problema de programación matemática.

Formalización

- La solución implica construir un problema, donde un ***multiplicador de Lagrange*** $\boldsymbol{\alpha_i}$ se asocia con cada restricción de desigualdad en el problema original:

$$max. \qquad L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i x_j$$

$$donde$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \qquad ; \qquad \alpha_i \geq 0$$

Formalización

- Así, una solución α_i … α_n al problema dual es:

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x_i} \qquad b = y_k - \sum \alpha_i y_i \boldsymbol{x_i} \boldsymbol{x_k} \quad \text{para cualquier} \quad \alpha_k > 0$$

- Cada α_i distinto de cero indica que $\boldsymbol{x_i}$ es un vector de soporte. Entonces, la función de clasificación sería:

$$\boldsymbol{f}(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x_i} \boldsymbol{x} + b$$

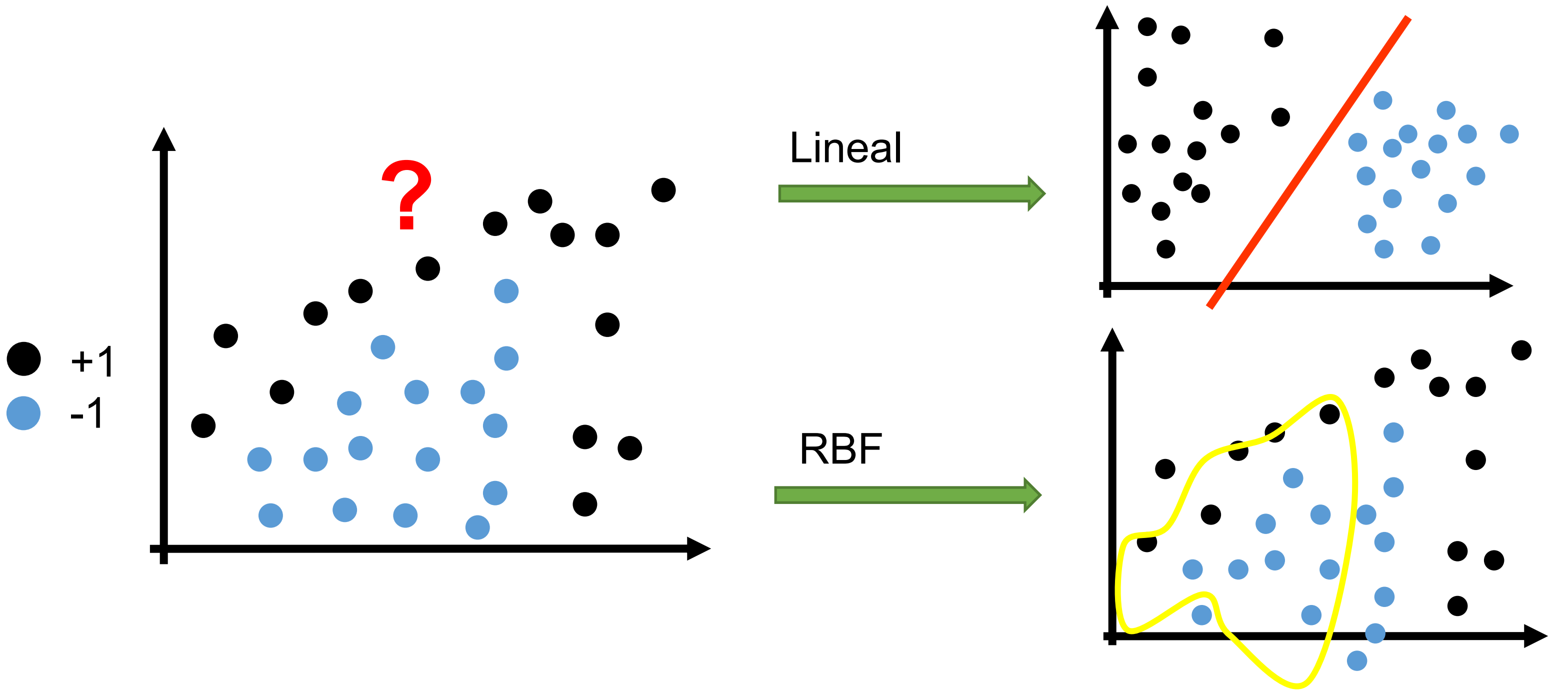
Kernels

- Son útiles en problemas de separación de clases.
- Se mapean los datos en un mejor espacio de representación por una determinada función, denominada ***kernel***.
- Los ***kernel*** comunes son:

Linear	$K(x, y) = x \cdot y$
Sigmoid	$K(x, y) = \tanh(ax.y + b)$
Polynomial	$K(x, y) = (1 + x \cdot y)^d$
KMOD	$K(x, y) = a\left[\exp\left(\frac{\gamma}{\lvert\lvert x-y \rvert\rvert^2+\sigma^2}\right) - 1\right]$
RBF	$K(x, y) = \exp(-a\lvert\lvert x - y\rvert\rvert^2)$
Exponential RBF	$K(x, y) = \exp(-a\lvert\lvert x - y\rvert\rvert)$

Kernels

Las funciones polinómicas y las funciones de base radial (RBF) pueden ser útiles para separar los datos, aún no siendo separables linealmente.



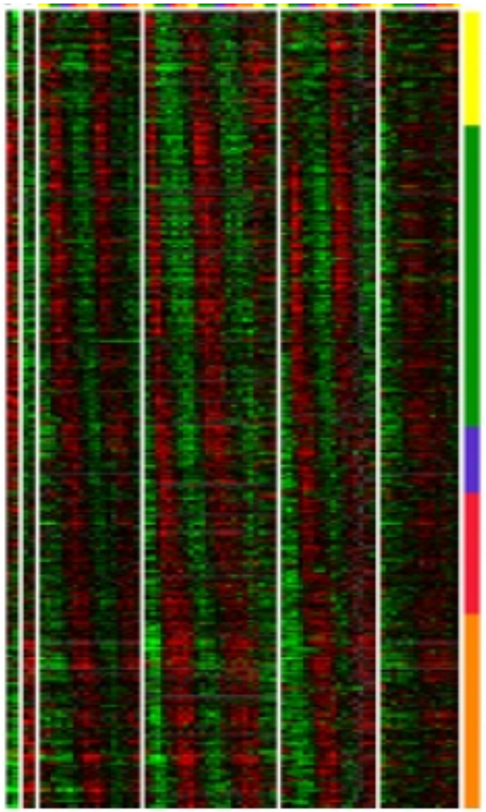
Consideraciones

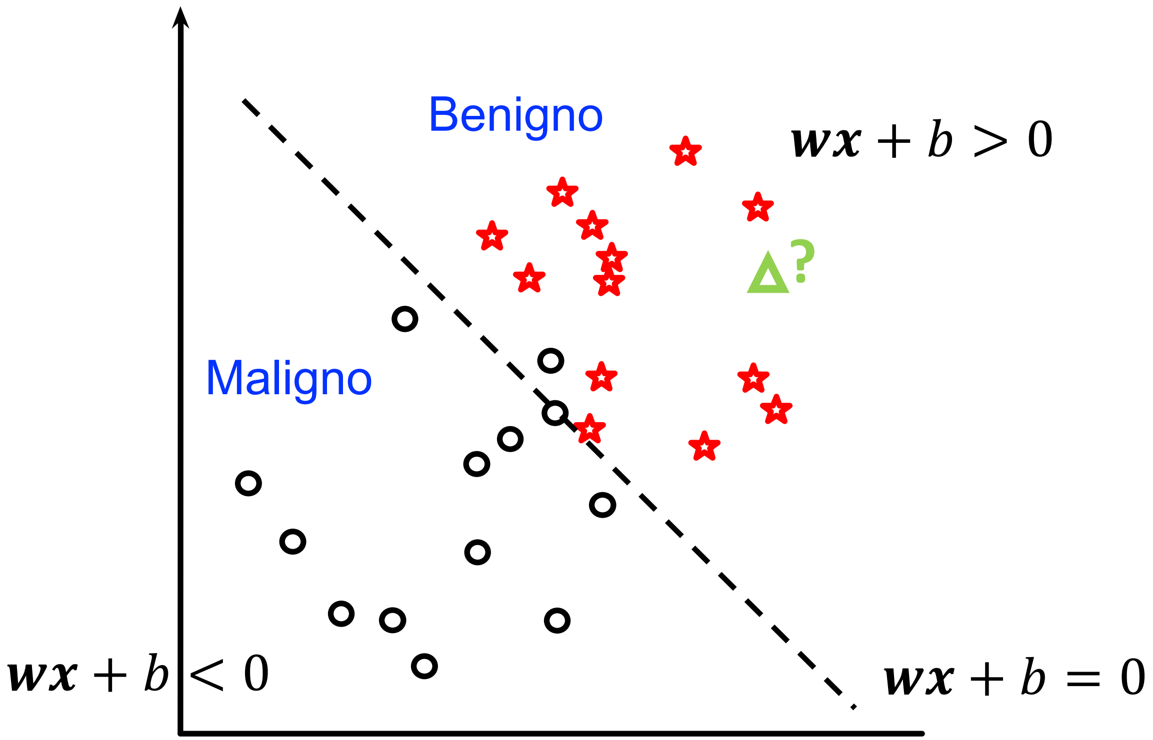
- Las SVM se encuentran actualmente entre los algoritmos que mejor desempeño tienen en problemas de clasificación.
- Se pueden aplicar a tipos de datos complejos, más allá de los vectores de características, mediante varios tipos de funciones kernel.
- Los algoritmos de optimización más populares para SVM utilizan un proceso de descomposición, como los multiplicadores de Lagrange.
- El ajuste de las SVM sigue siendo un arte negro: la selección de un kernel y parámetros específicos generalmente se hace de manera prueba y error.

Máquinas de soporte vectorial

Ejemplo

$X = [x_1, x_1, \ldots, x_n]$ $\quad Y$





$$f(x) = wx + b$$

Regresión Logística

```
Exactitud 0.9298245614035088
              precision    recall  f1-score   support

           0       0.95      0.87      0.91        45
           1       0.92      0.97      0.94        69

    accuracy                           0.93       114
   macro avg       0.93      0.92      0.93       114
weighted avg       0.93      0.93      0.93       114
```

SVM Lineal

```
#Se declara el tipo de kernel
ModeloSVM_1 = SVC(kernel='linear')
```

```
Exactitud 0.9210526315789473
              precision    recall  f1-score   support

          -1       0.97      0.82      0.89        45
           1       0.89      0.99      0.94        69

    accuracy                           0.92       114
   macro avg       0.93      0.90      0.91       114
weighted avg       0.93      0.92      0.92       114

Elementos de validación: 114
```

Máquinas de soporte vectorial

SVM RBF

```
#Se declara el tipo de kernel
ModeloSVM_2 = SVC(kernel='rbf')
```

```
Exactitud 0.8947368421052632
              precision    recall  f1-score   support

          -1       0.95      0.78      0.85        45
           1       0.87      0.97      0.92        69

    accuracy                           0.89       114
   macro avg       0.91      0.87      0.89       114
weighted avg       0.90      0.89      0.89       114

Elementos de validación: 114
```

Máquinas de soporte vectorial

SVM RBF

```
#Se declara el tipo de kernel
ModeloSVM_3 = SVC(kernel='sigmoid')
```

```
Exactitud 0.45614035087719296
              precision    recall  f1-score   support

          -1       0.05      0.02      0.03        45
           1       0.54      0.74      0.62        69

    accuracy                           0.46       114
   macro avg       0.29      0.38      0.33       114
weighted avg       0.35      0.46      0.39       114

Elementos de validación: 114
```