Módulo 4: Aprendizaje Automático

Temario de la Clase 4
30 de agosto del 2024
Máquinas de Soporte Vectorial

- Clasificador de margen suave
- Hiperplano
- Clasificación binaria
- Margen y optimización
- Máquinas de soporte vectorial
- Funciones kernel y truco de kernel
- Kernel polinómico y radial
- Clasificación multi-clase
- Regresión

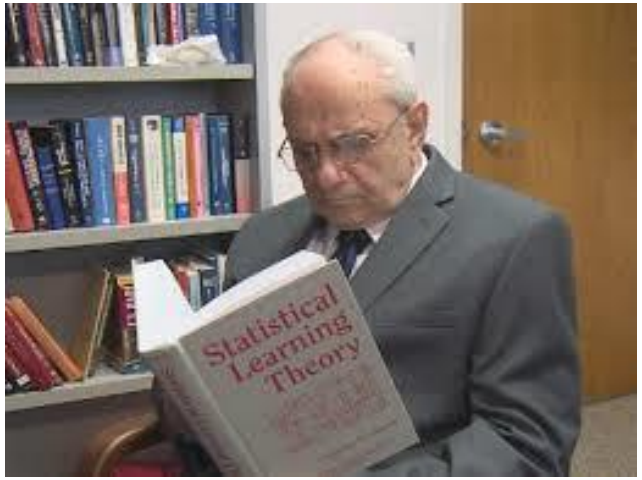
Introducción

Las máquinas de vector soporte o *Support Vector Machines* (SVM) son un tipo de *algoritmo de machine learning* **supervisado** aplicable a problemas de **regresión y clasificación**, aunque se usa más comúnmente como modelo de clasificación.

Se utilizan en varios dominios de aplicación, incluidos bioinformática, categorización de texto, visión por computadoras, detección de fraudes en tarjetas de crédito, etc.

Introducción

La popularidad de las máquinas de soporte vectorial creció a partir de los años 90 cuando los incorpora la comunidad informática.
Se considera una metodología muy flexible y con buen rendimiento en un amplio abanico de situaciones, aunque por lo general no es la que consigue los mejores rendimiento



Vladimir Vapnik

Referencias

Vapnik, V. (2000). Statistical Learning Theory. Willey.
Vapnik, V. (2013). The Nature of Statistical Learning Theory. Springer.

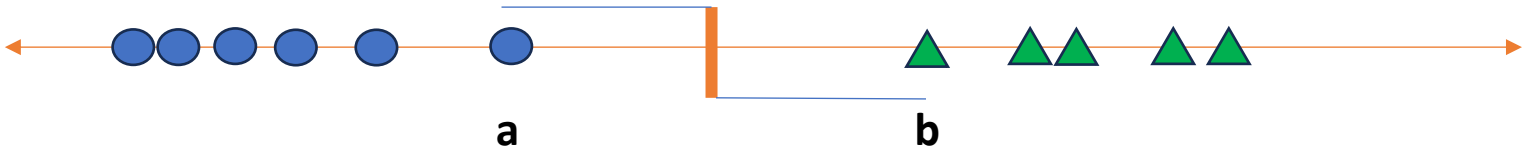
Introducción

Supongamos que tenemos un conjunto de datos correspondientes a dos categorías en una recta. Queremos encontrar un valor umbral que nos permita separar ambas categorías.

Puesto que nuestros datos están visiblemente separados, cualquier punto en el intervalo dado por a y b servirá como umbral para hacer la clasificación.

Maximal Margin Classifier

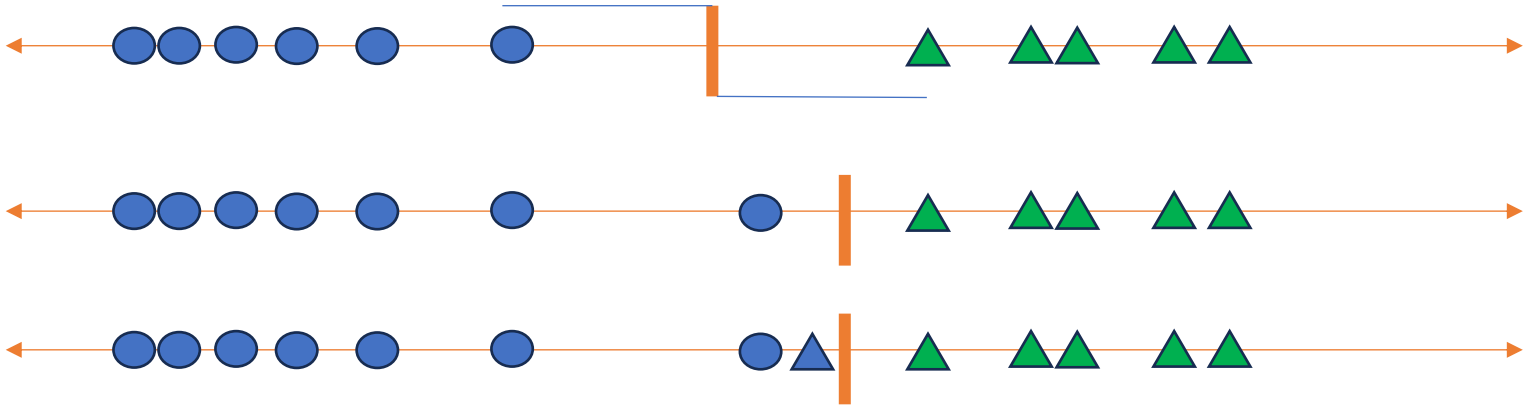
En este sencillo ejemplo, podemos usar el punto medio entre los bordes de cada grupo de datos para definir el umbral. La distancia entre esos bordes y el punto de umbral se denomina **margen**.



Si definimos un umbral que nos de el mayor margen tenemos un **Maximal Margin Classifier**, que nos va a permitir realizar la clasificación.

Maximal Margin Classifier

El problema con este tipo de clasificador, es que es muy sensible a los outliers:



Para reducir la rigidez de nuestro clasificador (y su sensibilidad a los outliers) podemos definir un umbral y un margen en el cual estén permitidas las clasificaciones erróneas.

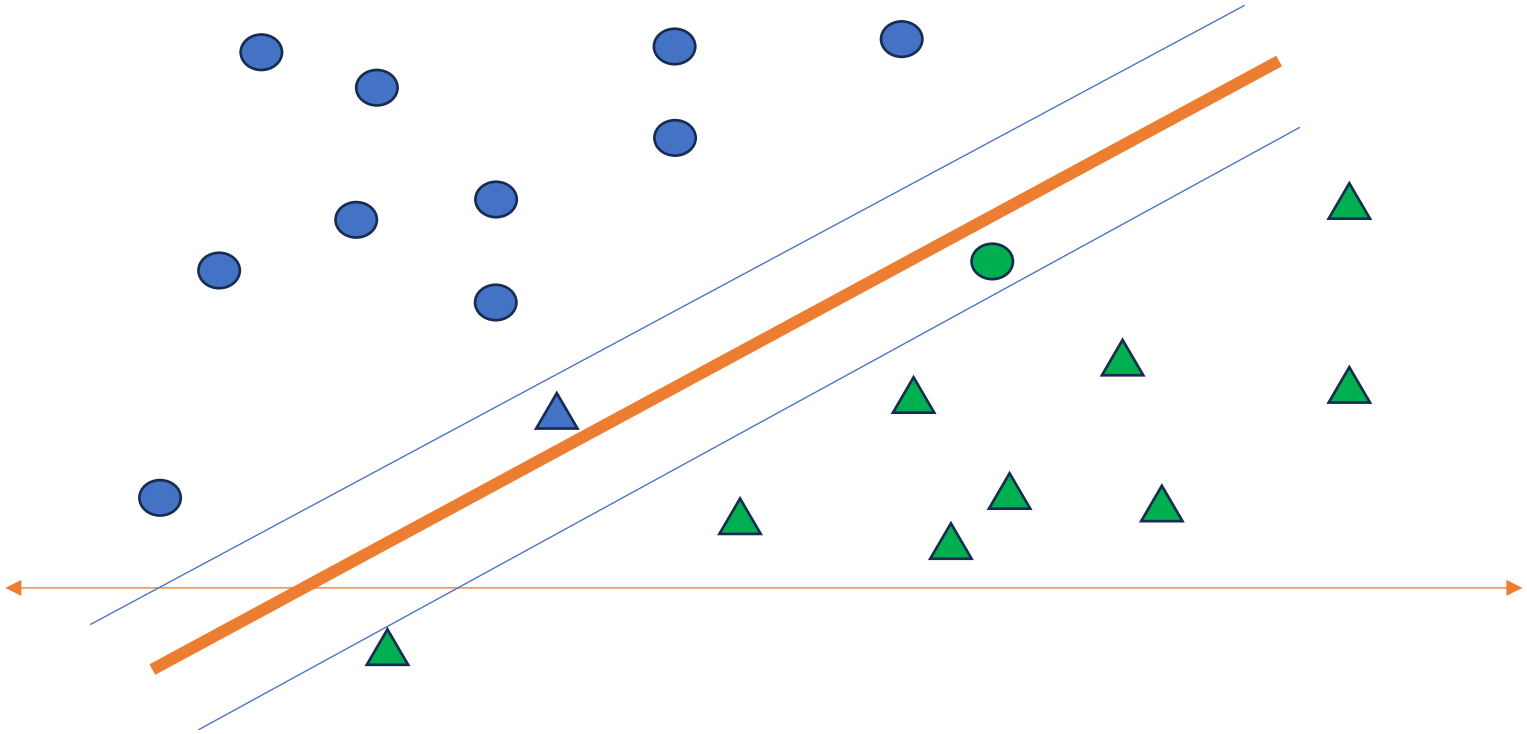
Soft Margin Classifier – Support Vector Classifier

Este tipo de clasificador, denominado **Soft Margin Classifier,** tendrá mayor error sobre el set de datos (mayor bias), pero menor error sobre datos futuros (menor varianza).

En este caso, un margen mas o menos pequeño determinará la relación bias/varianza que tendrá nuestro clasificador. Cuando se utiliza un margen suave para determinar el umbral, también se dice que estamos ante un **Support Vector Classifier**. El nombre radica en el hecho de que las observaciones en el borde y en el interior del margen se denominan vectores de soporte.

Support Vector Classifier

El mismo problema puede extenderse a 2 dimensiones, donde podemos usar una recta para determinar el umbral y dos rectas paralelas para determinar los márgenes.



En 3 dimensiones, un plano podría dividir el espacio en 2 subespacios y así realizar clasificaciones.

Hiperplano

En un espacio euclídeo n-dimensional, un hiperplano es un subespacio plano y afín (no tiene por qué pasar por el origen) de dimensión n–1, que divide el espacio en dos mitades o subespacios.

Por ejemplo, en un espacio de 2 dimensiones un hiperplano es un subespacio plano de una sola dimensión, o lo que es lo mismo, una línea, definida por la ecuación de la recta.

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0$$

donde cualquier *X = (x1, x2)* para los que se cumple la ecuación es un punto en el hiperplano.

En 3 dimensiones, un hiperplano sería un plano, en 1 dimensión un hiperplano sería un punto.

Hiperplano

Para escenarios n-dimensionales, la ecuación anterior puede ser extendida a :

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n = 0$$

donde cualquier X = (x1, x2, ...,xn) para los que se cumple la ecuación es un punto en el hiperplano.

Hiperplano

En el supuesto que X no satisfaga la ecuación, dándose uno de estos dos casos :

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n > 0$$

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n < 0$$

el punto X estará situado a uno u otro lado del hiperplano, no sobre él.

Hiperplano para clasificación binaria

Suponiendo que contamos con una matriz de datos m x n con m observaciones y n predictores (donde en la variable respuesta son distinguibles dos clases distintas y1, ..., ym ∈ {-1, 1})

El objetivo será el de desarrollar un clasificador en base al subgrupo de datos de entrenamiento que clasifique correctamente nuevas observaciones en base a los valores de los predictores en función de un hiperplano de separación con la propiedad

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_n x_{in} > 0 \; ; si \; y_i = 1$$

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_n x_{in} < 0 ; si \; y_i = -1$$

Hiperplano para clasificación binaria

Equivalentemente:

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_n x_{in}) > 0$$

para cada i = 1, ..., m.

β_0, β_1, ..., β_n son los coeficientes del hiperplano.

Hiperplano para clasificación binaria

Una nueva observación **x*** se asignará a un grupo u otro dependiendo de en qué lado del hiperplano se localice (en función del signo de $f(x^*) = \beta_0 + \beta_1 x^*_1 + ... + \beta_n x^*_n$).

Si f(x*) es positiva, se asigna la nueva observación a la clase 1, y si es negativa, a la clase -1.

Hiperplano para clasificación binaria

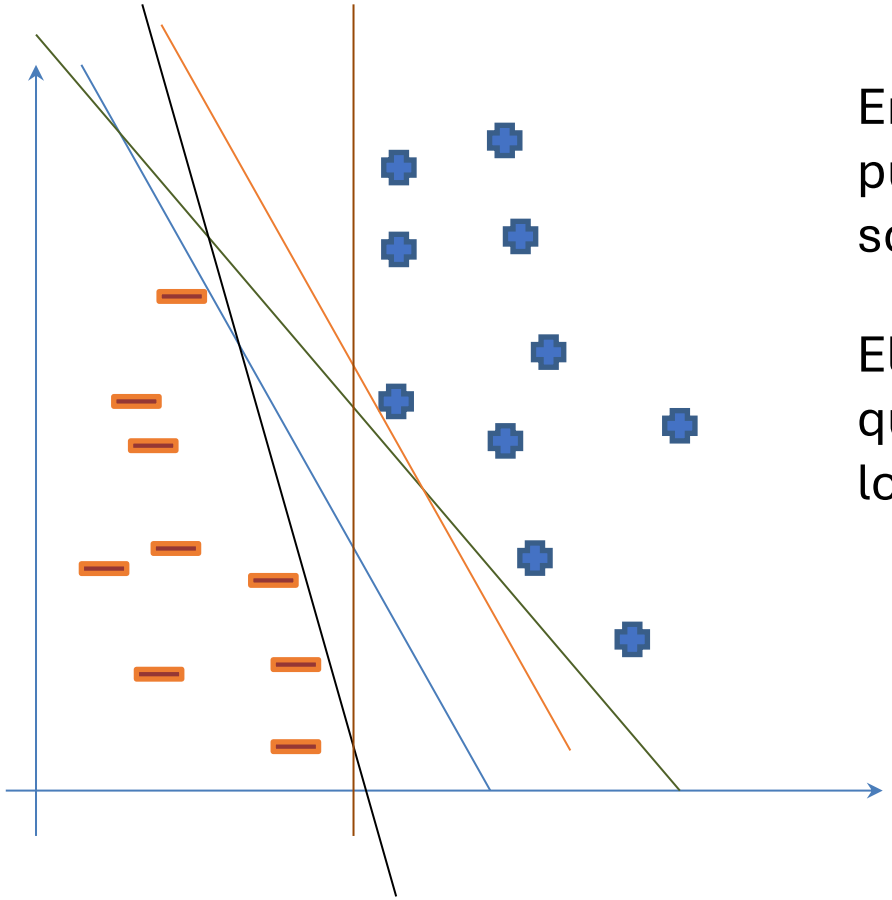
La magnitud de f(x*) también es informativa:

- si f(x*)tiene un valor muy lejano a 0, significa que x* se encuentra muy lejos del hiperplano, lo cual aporta más seguridad a la clasificación de dicha observación.

- un valor de f(x*) próximo a 0 significa que la observación está cerca del hiperplano, con lo que estaremos menos seguros acerca de la clase asignada a esta observación.

Hiperplano para clasificación binaria

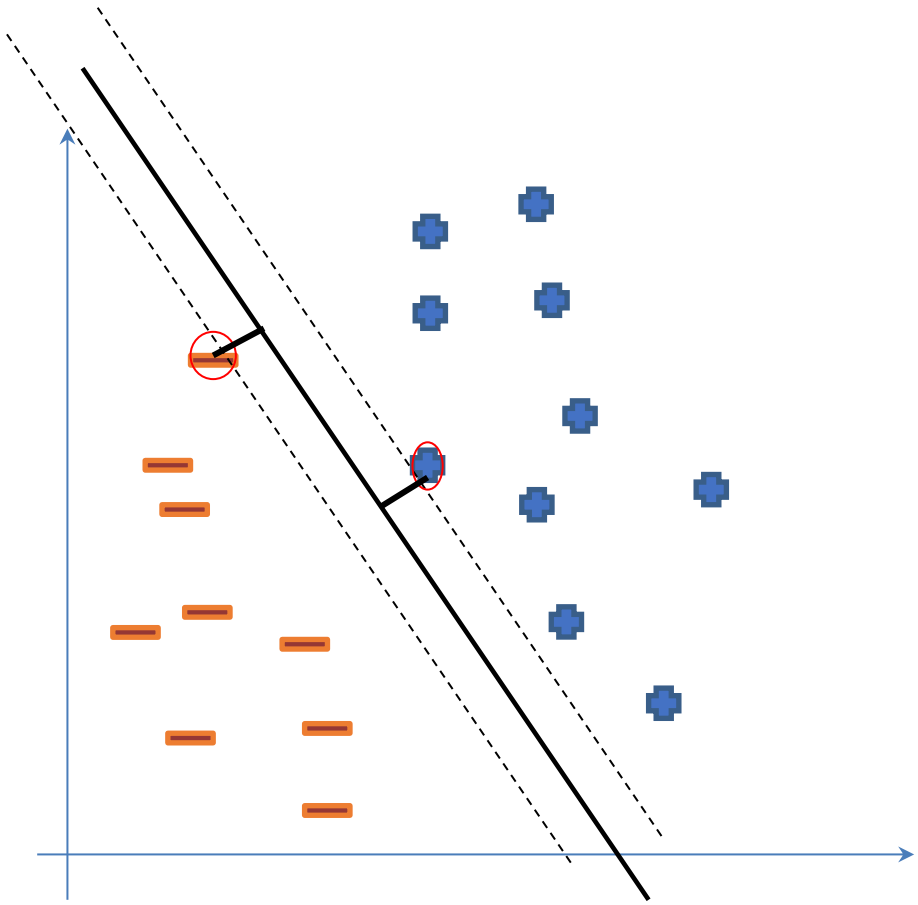
Conceptualmente



En un problema de clasificación puedo encontrar infinitas soluciones.

El reto es encontrar aquel hiperplano que más separado se encuentre de los eventos.

Hiperplano para clasificación binaria



En un problema de clasificación puedo encontrar infinitas soluciones.

El reto es encontrar aquel hiperplano que más separado se encuentre de los eventos.

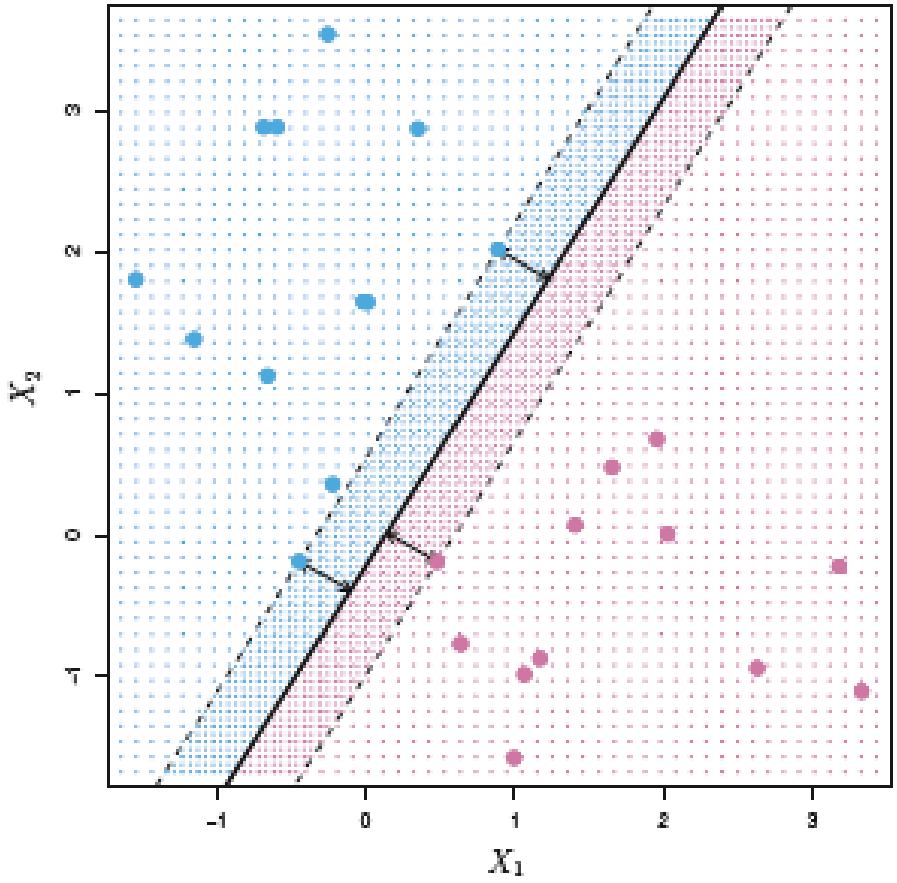
Se trata del hiperplano que maximice la distancia entre las clases. Se denomina **hiperplano óptimo de separación**

Esto define un *margen* formado por otros dos planos paralelos determinados por los vectores soporte de cada clase.

Maximal Margin

Se obtiene calculando las distancias perpendiculares de cada observación a un hiperplano dado.

La distancia más pequeña se corresponde con la distancia mínima de las observaciones al hiperplano, espacio conocido como margen.

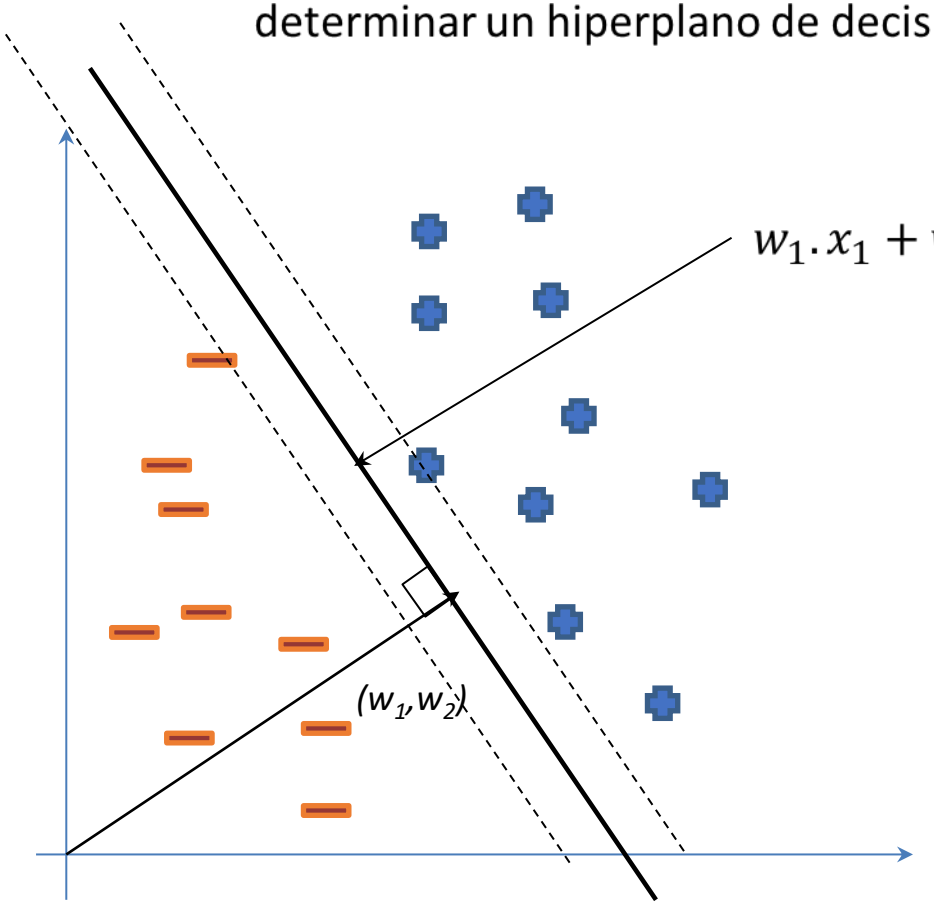


Con esto, el hiperplano óptimo de s
distancia mínima de las observaciones al hiperplano, o lo que es lo mismo, el mayor margen (***M***).

Por tanto, los parámetros del hiperplano β0, β1, ..., βp se optimizan para maximizar ***M***.

Hiperplano para clasificación binaria

Sea $\boldsymbol{x} \in \mathbb{R}^{\boldsymbol{n}}$ y $\boldsymbol{w} \in \mathbb{R}^{\boldsymbol{n}}$ los parámetros del sistema, es posible determinar un hiperplano de decisión representado por $\boldsymbol{w}.\boldsymbol{x} + \boldsymbol{b} = \boldsymbol{0}$



$$w_1.x_1 + w_2.x_2 + b = 0$$

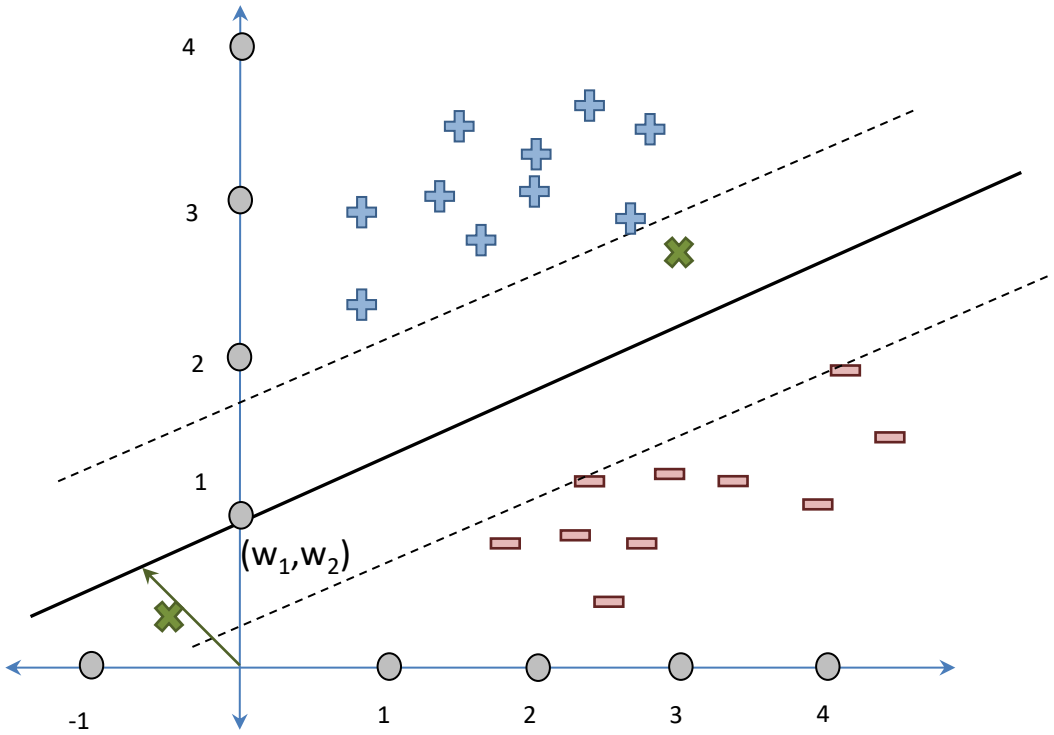
Número escalar

Fronteras de decisión

$$\mathrm{w}.x + b < 0$$

$$\mathrm{w}.x + b > 0$$

Ejemplo



$(w_1,w_2) = (-0.8, 0.7)$
$b=1$

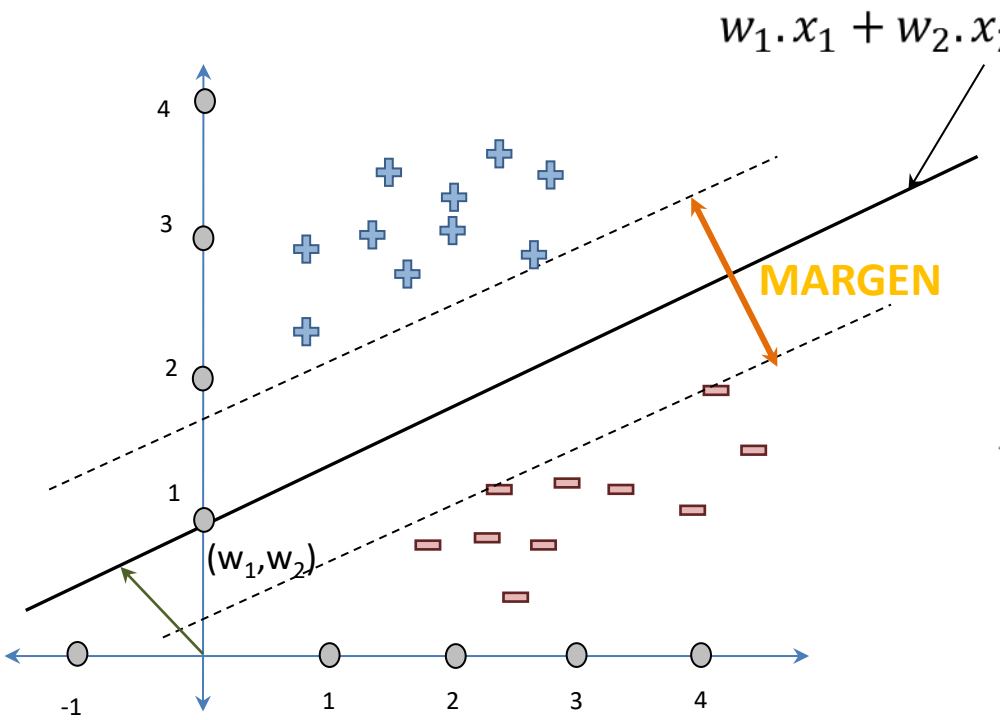
$$-0.8x_1 + 0.7x_2 + 1 = 0$$

$(x_1,x_2) = (3,2.5) \rightarrow$ $\quad -0.8(3) + 0.7(2.5) + 1 = 0.35$

$(x_1,x_2) = (-0.5,0.5) \rightarrow$ $\quad -0.8(-0.5) + 0.7(0.5) + 1 = -0.68$

Margen

El objetivo es encontrar un margen equidistante al hiperplano



Fronteras de decisión con margen

$w.x + b \geq 1$ $\quad$ $w.x + b \leq 1$

Clase 1: ✚ $\quad$ Clase -1: ▬

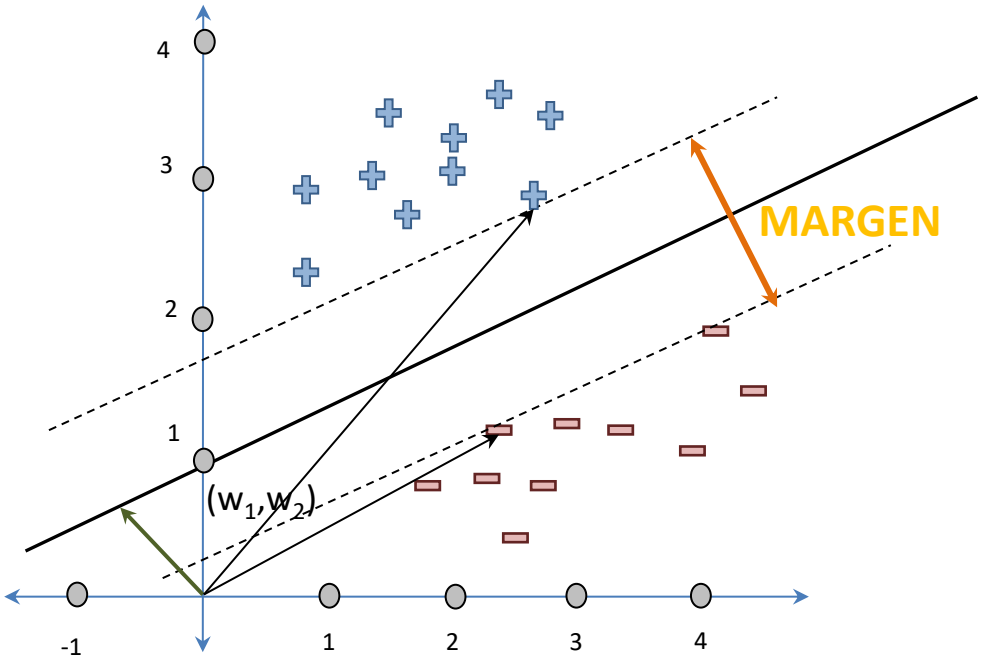
Expresión equivalente

$$y_i(w.x + b) \geq 1 \qquad \begin{cases} y_i = 1\ ;\ x_+ \\ y_i = -1\ ;\ x_- \end{cases}$$

Para los vectores soporte

$$y_i(w.x + b) - 1 = 0$$

Margen máximo



Longitud del margen

$$d_- = \frac{x_-.w}{\|w\|} \qquad d_+ = \frac{x_+.w}{\|w\|}$$

$$d = d_+ - d_- = \frac{1}{\|w\|}(x_+ \cdot w - x_- \cdot w)$$

Para los vectores soporte

$$y_i(w.x + b) - 1 = 0$$

$$d = \frac{1}{\|w\|}(1 - b + 1 + b) \Longrightarrow$$

$$\boxed{d = \frac{2}{\|w\|}}$$

Optimización

Maximizar

$$d = \frac{2}{\|w\|} = \frac{2}{w^T w}$$

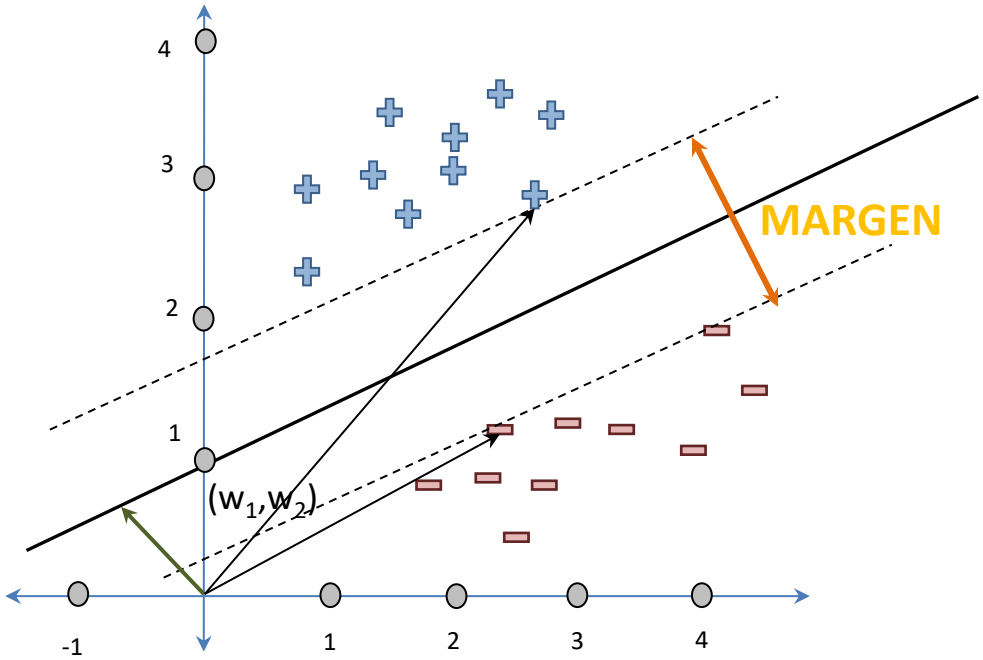
equivale a

Minimizar

$$w^T w$$

Restricciones

$$y_i(w.x + b) - 1 = 0$$



Multiplicadores de Lagrange

Optimización

$$f(w) = \frac{1}{2} w^T w$$

Función objetivo

$$g(w, b) = y_i(w.x_i + b) - 1 = 0$$

Restricciones de igualdad(una por cada vector soporte)

$$L(w, b, \alpha_i) = \frac{1}{2} w^T w - \sum_{i=1}^{l} \alpha_i [y_i(w \cdot x_i + b) - 1]$$

l es el número de vectores soporte

Al ser un problema de minimización se deriva y se iguala a cero

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{i}^{l} \alpha_i \cdot y_i \cdot x_i$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i}^{l} \alpha_i \cdot y_i \cdot = 0$$

$$\begin{cases} L(\alpha_i) = \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \end{cases}$$

Consideraciones

Se denominan “vectores soporte” porque “soportan” al hiperplano óptimo de separación.

Si estas observaciones cambiaran ligeramente, el hiperplano lo haría también.

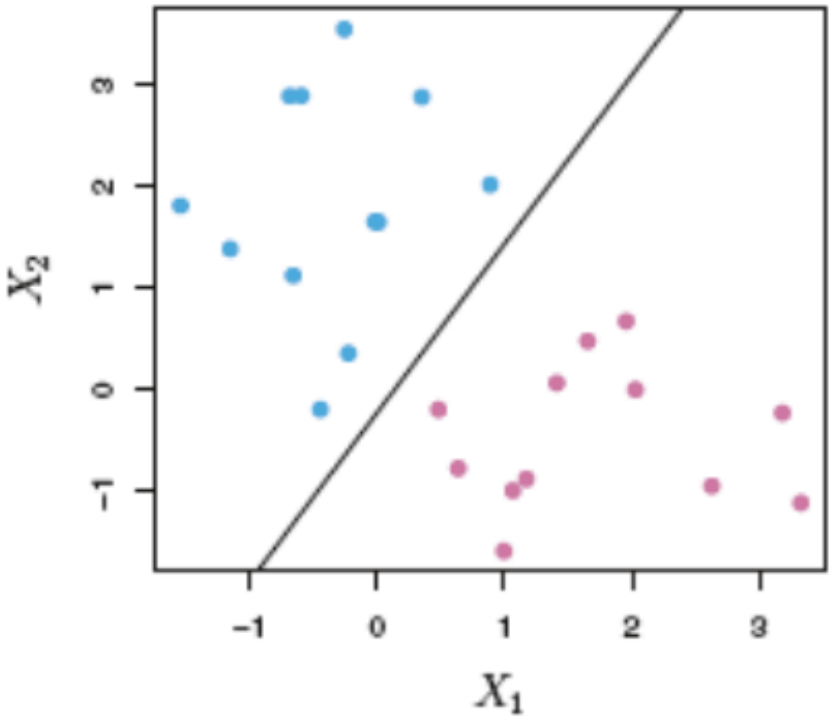
Se espera que un clasificador con un margen alto para las observaciones de entrenamiento lo tenga también para las observaciones de test, y pueda por tanto clasificarlas correctamente.

Un problema que puede surgir es que este clasificador sufra de overfitting cuando el número de dimensiones (n) es alto.

El método de los multiplicadores de Lagrange sólo es útil cuando las clases son linealmente separables.

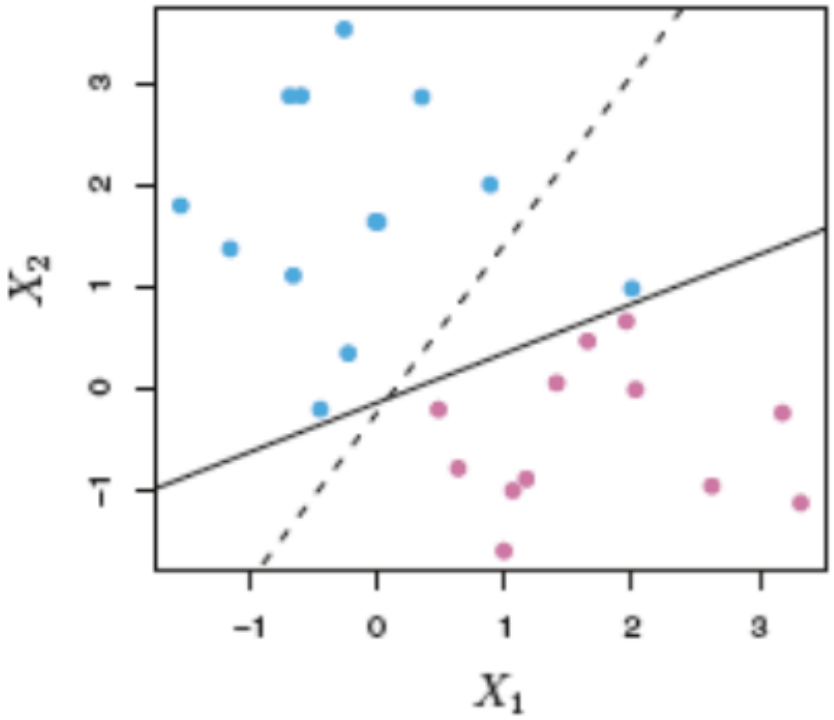
Clasificación binaria: cuasi-linealmente separable

Observaciones pertenecientes a dos clases no necesariamente serán separables mediante el uso de un hiperplano.



El agregado de una sola observación tiene la capacidad de cambiar drásticamente el hiperplano óptimo de separación.

Clasifica todas las observaciones de entrenamiento perfectamente → sensibilidad a nuevas observaciones.



Margen reducido → distancia supone una medida de la confianza con la que una observación es correctamente clasificada.

Soft Margin Classifier: Support Vector Classifier

Basado en un hiperplano pero no separa perfectamente las dos clases.
Está permitido que algunas observaciones se encuentren en el lado incorrecto del margen.

- **Mayor robustez a observaciones individuales.**
- **Mejor clasificación de la mayoría de las observaciones de entrenamiento y test**.

Estas serán las observaciones de entrenamiento mal clasificadas por el modelo.

El support vector classifier clasifica cada nueva observación en función de a qué lado de hiperplano pertenezca.
Sólo las observaciones que se encuentran sobre el margen o que lo violan (vectores soporte) afectarán al hiperplano

Los **vectores soporte** se corresponden con las observaciones que se encuentran sobre el margen y también las que lo violan.

Proceso de optimización

Parámetro de regularización o tunning parameter ***C***

Controla la severidad permitida de las violaciones de las ***n*** observaciones sobre el margen e hiperplano y el equilibrio sesgo-varianza

Casos:

C > 0 no más de ***C*** observaciones pueden encontrarse en el lado incorrecto del hiperplano.

Si **C** es pequeño, los márgenes serán estrechos pues muy pocas observaciones podrán estar en el lado incorrecto del mismo.

Al aumentar ***C***, mayor es la tolerancia → margen será más ancho; más vectores soporte (modelo más flexible y con mayor sesgo pero menor varianza).

C = 0 el clasificador es equivalente al maximal margin classifier, pues no están permitidas violaciones sobre el margen.

Proceso de optimización

En la práctica el parámetro **C** se escoge u optimiza por validación cruzada.

- **Poca varianza y alto sesgo: márgenes anchos y mayor número de vectores soporte.**
- **Mucha varianza y bajo sesgo: márgenes estrechos y menor número de vectores soporte.**

Clasificación binaria: caso no linealmente separable

En general, las aplicaciones complejas del mundo real requieren un espacio de hipótesis más expresivo que las funciones lineales simples.

Por lo tanto, necesitamos introducir funciones más complejas para manejar datos del mundo real.

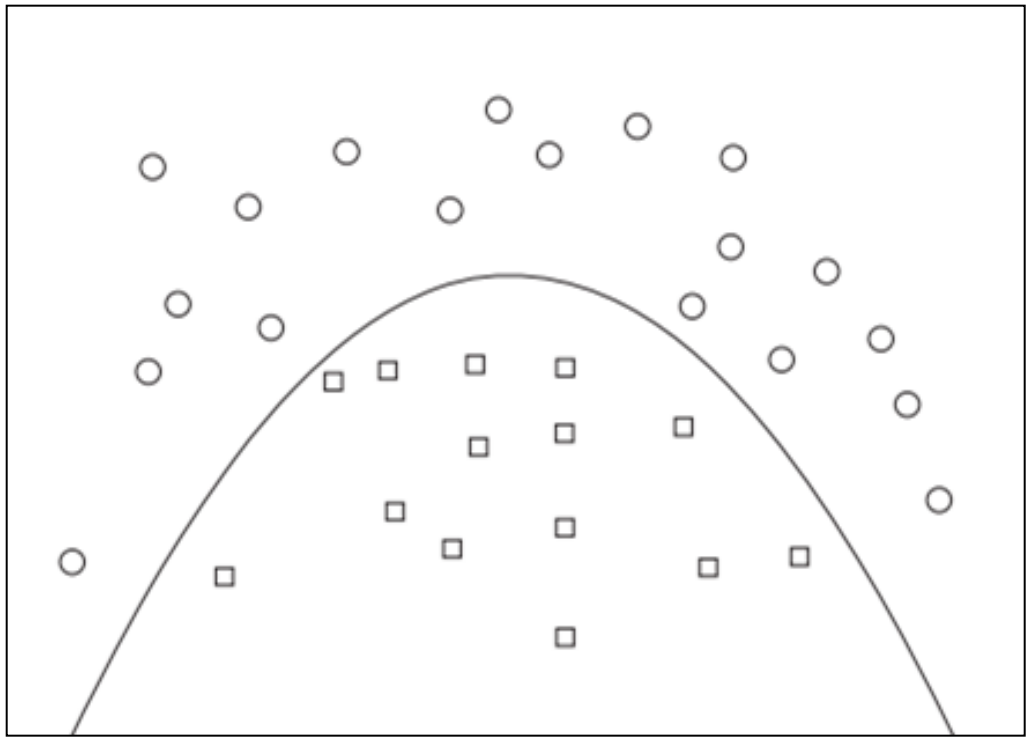
Si observamos, vemos que la única manera en que los datos aparecen en el problema de optimización descrito se presenta en forma de producto interno <xi, xj>.

Clasificación binaria: caso no linealmente separable

Obtener una representación potencialmente mejor de los datos, puede mapear los puntos de datos en un espacio alternativo de *dimensiones superiores* llamado espacio de características a través de un reemplazo:

$$<x_i, x_j> \rightarrow \phi(x_i), \phi(x_j) .$$

No es necesario conocer la forma funcional del mapeo $\phi(x)$



Puntos de datos bidimensionales que no pueden ser linealmente separados.

Los círculos y los cuadrados, respectivamente, representan puntos de datos en las clases −1 y +1.

Support Vector Machine

Al reemplazar el producto interno con una ***función kernel*** (que realiza el mapeo) elegida apropiadamente, los datos pueden volverse linealmente separables en el espacio de características a pesar de no ser separables en el espacio de entrada.

Al utilizar una función kernel para transformar los datos hacia una dimensión mayor donde los datos son linealmente separables, y por lo tanto, donde es posible realizar una clasificación basada en un Support Vector Classifier, se dice que estamos ante una **máquina de soporte vectorial / máquina de vectores de soporte** o **Support Vector Machine (SVM)**.

Funciones Kernel

Para hacer que este proceso sea matemáticamente posible, las SVM utilizan las funciones kernel para encontrar de manera sistemática Support Vector Classifiers en dimensiones mayores.

El número de cómputos requeridos se reduce gracias al truco de Kernel (Kernel trick) que calcula las relaciones de alta dimensionalidad sin transformar los datos a una mayor dimensión: se calculan las relaciones entre puntos como si ya estuvieran en una dimensión más alta.

Esto es especialmente útil cuando el número de características (dimensiones) es muy alto o incluso infinito, lo cual sería computacionalmente prohibitivo.

Kernel trick (truco de Kernel)

El primer paso del ***kernel trick*** es cambiar la representación de los datos:

$$x = (x_1, \dots, x_n) \rightarrow \phi(x) = (\phi_1(x), \dots, \phi_N(x))$$

Este paso es equivalente a embeber el espacio de entrada $\boldsymbol{X \subseteq R^n}$ en un nuevo espacio de características que se puede definir como:

$$F = \{\phi(x) : x \in X\}$$

$\boldsymbol{\phi : X \rightarrow F \subseteq R^N}$ es el nuevo mapeo

El segundo paso es detectar relaciones en el espacio de características, utilizando un *soft margin classifier*.

Las relaciones lineales se pueden representar usando productos internos $\phi(x), \phi(z)$ entre todos los pares de puntos observados $\boldsymbol{x, z \in X}$.

La función que devuelve el producto interno entre las imágenes de dos puntos de datos cualesquiera en el espacio de características se llama ***kernel***.

Kernel lineal

Entre los kernels más populares para usar con SVMs se encuentran:

$$K(x_i, x_{i'}) = \sum_{j=1}^{n} x_{ij} \cdot x_{i'j}$$

El ***kernel lineal*** cuantifica la similitud de un par de observaciones usando la correlación de Pearson.

Con un kernel lineal, el clasificador obtenido es equivalente a un ***support vector classifier.***

Kernel polinómico

Para dos puntos xi, xj se tiene:

$$K(xi, xj) = (x_i x_j + r)^d$$

Donde d determina el grado del polinomio y r tendrá influencia en los coeficientes. Si r = 0.5 y d = 2:

$$K(x_i, x_j) = (x_i x_j + 0.5)^2 = x_i x_j + {x_i}^2 {x_j}^2 + 0.25$$
$$K(x_i, x_j) = (x_i, x_i^2, 0.5).(x_j, x_j^2, 0.5)$$

Obtengo dos vectores de 3 componentes. Puesto que las terceras componentes son iguales, las puedo obviar. Entonces obtuve una nuevo mapeo de puntos de coordenadas (x_i, x_i^2), (x_j, x_j^2).

Kernel polinómico

Para dos puntos xi, xj se tiene:

$$K(xi, xj) = (x_i x_j + r)^d$$

Donde d determina el grado del polinomio y r tendrá influencia en los coeficientes. Si r = 0.5 y d = 2:

$$K(x_i, x_j) = (x_i x_j + 0.5)^2 = x_i x_j + {x_i}^2 {x_j}^2 + 0.25$$
$$K(x_i, x_j) = (x_i, x_i^2, 0.5).(x_j, x_j^2, 0.5)$$

La función kernel es equivalente a un producto punto, que determina las coordenadas de alta dimensión para los datos. Solamente es necesario calcular el producto punto para todos los pares de datos del dataset.

Kernel radial (RBF kernel)

El radial basis function kernel se define como:

$$K(xi, xj) = e^{-\gamma(x_i - x_j)^2}$$

donde γ es una constante positiva: cuanto mayor sea, mayor la flexibilidad del SVM.

Este kernel encuentra SVC en infinitas dimensiones, por lo que no es posible visualizarlo.

El ***kernel radial*** tiene un comportamiento muy local → solo las observaciones de entrenamiento cercanas a una observación de test tendrán efecto sobre su clasificación.

Kernel radial (RBF kernel)

Si tomamos un kernel polinómico donde r=0 tenemos:

$$K(xi, xj) = (x_i x_j)^d = \left(x_i^d\right).(x_j^d)$$

Con lo cual tenemos una transformación en la misma dimensión de los datos. Combinando varios de estos kernel de distintas dimensiones obtenemos un producto puntos de varias dimensiones:

$$K(xi, xj) = (x_i x_j)^1 + (x_i x_j)^2 + (x_i x_j)^3$$
$$= \left(x_i, x_i^2, x_i^3\right).(x_j, x_j^2, x_j^3)$$

Kernel radial (RBF kernel)

Volviendo a la expresión del kernel radial, podemos tomar γ= ½ y reescribir:

$$K(xi, xj) = e^{-\gamma(x_i - x_j)^2} = e^{-1/2(x_i - x_j)^2}$$

$$K(xi, xj) = e^{-1/2(x_i + x_j)^2} e^{x_i x_j}$$

Desarrollando la exponencial por la serie de Taylor:

$$e^x = 1 + \frac{1}{1!}x + \frac{1}{2!}x^2 + \frac{1}{3!}x^3 + \cdots + \frac{1}{\infty!}x^\infty$$

$$e^{x_i x_j} = 1 + \frac{1}{1!}x_i x_j + \frac{1}{2!}(x_i x_j)^2 + \frac{1}{3!}(x_i x_j)^3 + \cdots + \frac{1}{\infty!}(x_i x_j)^\infty$$

Kernel radial (RBF kernel)

Desarrollando la exponencial por la serie de Taylor:

$$e^{x_i x_j} = 1 + \frac{1}{1!} x_i x_j + \frac{1}{2!} (x_i x_j)^2 + \frac{1}{3!} (x_i x_j)^3 + \cdots + \frac{1}{\infty!} (x_i x_j)^\infty$$

$$e^{x_i x_j} = \left(1, \sqrt{\frac{1}{1!}} x_i, \sqrt{\frac{1}{2!}} {x_i}^2 + \sqrt{\frac{1}{3!}} {x_i}^3 + \cdots \right).$$

$$\left(1, \sqrt{\frac{1}{1!}} x_j, \sqrt{\frac{1}{2!}} {x_j}^2 + \sqrt{\frac{1}{3!}} {x_j}^3 + \cdots \right)$$

Por lo que el kernel radial contiene proyecciones de los datos en un número infinito de dimensiones.

Kernel radial (RBF kernel)

$$K(xi, xj) = e^{-\gamma(x_i - x_j)^2}$$

Debido a la relación cuadrática entre la diferencia de xi e xj, dicha relación disminuirá a medida que ellos estén distantes entre sí, por lo que la clasificación de un punto se verá mayormente influenciada por puntos cercanos.

Para resumir

SVM Básico: El objetivo principal de SVM es encontrar el hiperplano que mejor separa las clases en un espacio de características de alta dimensión. Este hiperplano se elige de manera que maximice el margen, que es la distancia entre el hiperplano y los puntos de datos más cercanos de cualquier clase, conocidos como *support vectors*.

LSVM (Linear SVM): Es una versión simplificada del SVM que se aplica cuando los datos son linealmente separables o casi linealmente separables. Es decir, LSVM busca un hiperplano lineal que separe las clases de la mejor manera posible. Al ser lineal, se utiliza cuando se asume que una separación lineal es suficiente para clasificar los datos correctamente.

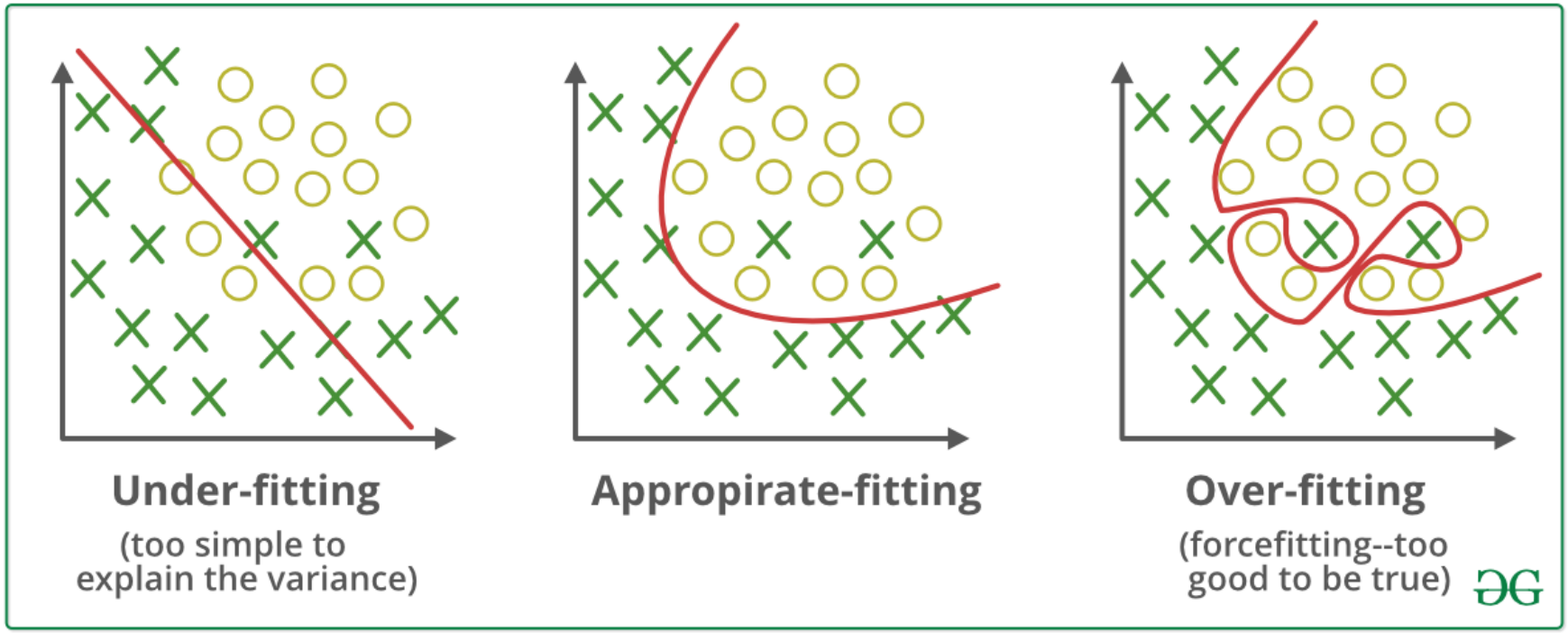
Función de Costo: En LSVM, la función de costo intenta minimizar la suma de las violaciones del margen (en caso de que algunos puntos estén dentro del margen o mal clasificados) y el error de clasificación, manteniendo al mismo tiempo un margen lo más grande posible.

Kernel Trick: En SVM, a menudo se usa el "truco del kernel" para permitir que el algoritmo se adapte a separaciones no lineales. Sin embargo, LSVM no utiliza este truco, ya que se limita a clasificaciones lineales.

Consideraciones

Es importante tener en cuenta que una mayor flexibilidad no tiene por qué mejorar las predicciones.

Un modelo muy flexible puede ajustarse demasiado a los datos de entrenamiento.



Clasificación multi-clase

Existen varias extensiones de los SVMs para problemas de clasificación con más de dos clases ($K > 2$),

- One-versus-one
- One-versus-all

Clasificación one-versus-one

Este método construye $\binom{K}{2}$ SVMs, correspondiente a K(K – 1)/2, cada uno comparando un par de clases.

Una observación de test se clasifica usando cada uno de los SVMs, contando el número de veces que esta observación es asignada a cada una de las K clases.

La clase final predicha será aquella a la que la observación ha sido asignada en la mayoría de los SVMs.

Clasificación one-versus-all

Se ajustan K SVMs, cada vez comparándose una de las K clases (codificada como +1) con el resto de K -1 clases (codificadas como -1).

Siendo

$$\beta_{0k}, \beta_{1k}, \ldots, \beta_{pk}$$

los parámetros resultantes del ajuste de un SVM y x* una observación de test, la observación será asignada a la clase para la que:

$$\beta_{0k} + \beta_{1k}x^*_1 + \ldots + \beta_{nk}x^*_n$$

sea mayor.
La magnitud de f(x*) indica como de lejos está x* del hiperplano de separación.
Cuanto más lejos esté, mayor será el nivel de confianza de que la observación x* ha sido correctamente clasificada.

SVM vs. Regresión logística

Una característica interesante de los ***support vector classifiers*** es que solo los vectores soporte juegan un papel importante en la clasificación final obtenida: observaciones en el lado correcto del margen no afectan a las predicciones.

En contraposición, el término de penalización en regresión logística es muy pequeño para observaciones lejanas al límite de decisión, pero nunca es exactamente 0.

Ambos métodos estadísticos pueden dar con frecuencia resultados similares.

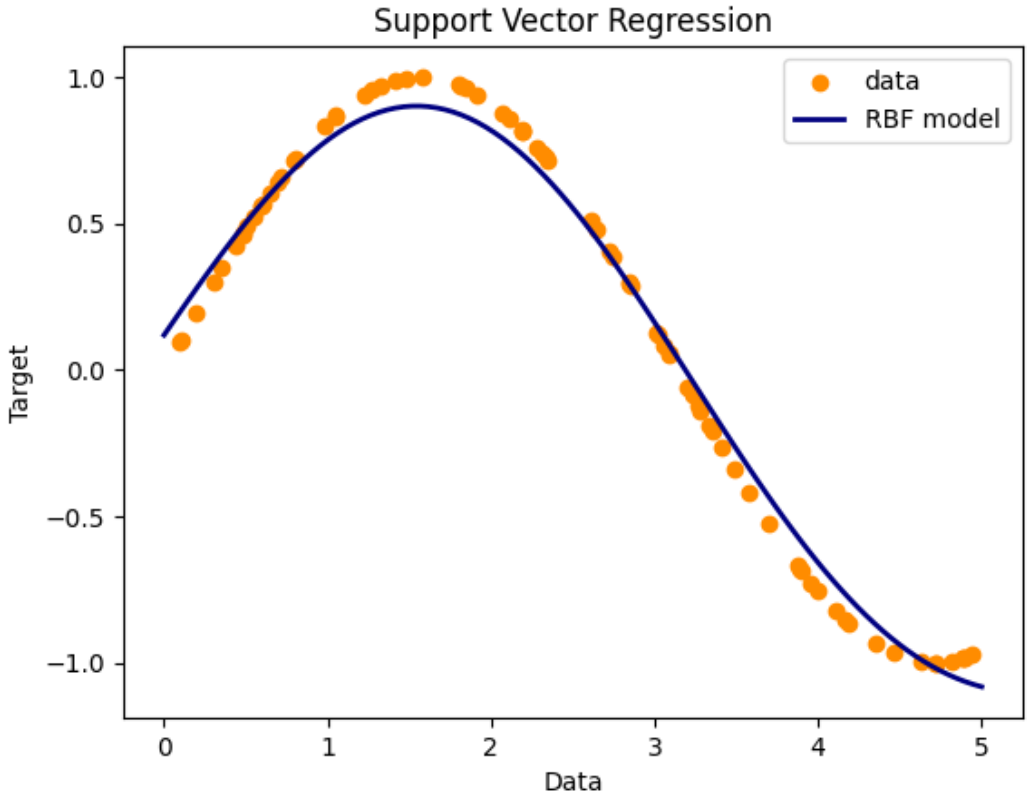
SVM vs. Regresión logística

Cuando las clases son fácilmente separables, los SVMs suelen superar a la regresión logística.

En situaciones donde las clases son más solapantes, la regresión logística suele ser la opción escogida.

El uso de kernels para aumentar la dimensionalidad no está limitado sólo al uso de SVMs (podrían usarse también para regresión logística), pero su uso está más extendido en estos casos.

Support Vector Regression (SVR)



Es una variante de las máquinas de vectores de soporte (SVM) y está diseñada para predecir valores numéricos continuos, lo que la hace adecuada para tareas como la previsión de series temporales, la predicción de precios de acciones y más.

Support Vector Regression (SVR)

SVR busca encontrar una función que prediga una variable objetivo continua mientras maximiza el margen entre los valores predichos y los puntos de datos reales.

Margen: SVR identifica un “margen” alrededor de la línea de regresión predicha y su objetivo es ajustar la línea dentro de este margen mientras se minimiza el error de predicción.

Vectores de soporte: en SVR, los puntos de datos que están más cerca de la línea de regresión y definen el margen se conocen como “vectores de soporte”. Estos puntos juegan un papel crucial en la determinación del modelo de regresión.

Support Vector Regression (SVR)

Truco kernel: SVR puede usar varias funciones de kernel (lineal, polinómica y radial) para transformar el espacio de características, lo que hace posible modelar relaciones no lineales entre las características de entrada y la variable objetivo.

Hiperparámetros: SVR requiere ajustar los hiperparámetros, como el parámetro de regularización (C) y los parámetros de kernel, para lograr el mejor rendimiento del modelo.

Función de pérdida: SVR normalmente utiliza una función de pérdida insensible a épsilon que permite algunos errores dentro de un rango definido (épsilon) y penaliza más severamente los errores fuera de este rango.

Support Vector Regression (SVR)

Robustez: SVR es robusto a los valores atípicos, ya que se centra principalmente en los puntos de datos cercanos al margen (vectores de soporte) y no depende en gran medida de todos los puntos de datos.

Conclusión

La disponibilidad de algoritmos de clasificación confiables, con claras propiedades computacionales y estadísticas, ha marcado un progreso significativo en el campo del análisis de patrones.

Los recursos de computación y memoria son consideraciones prácticas importantes al analizar datos extensos.

SVM solo necesita un pequeño subconjunto de puntos de entrenamiento (los vectores de soporte) para definir la regla de clasificación, lo que a menudo lo hace más eficiente en memoria y menos exigente desde el punto de vista computacional al inferir la clase de una nueva observación.