Machine Learning

Support Vector Machines

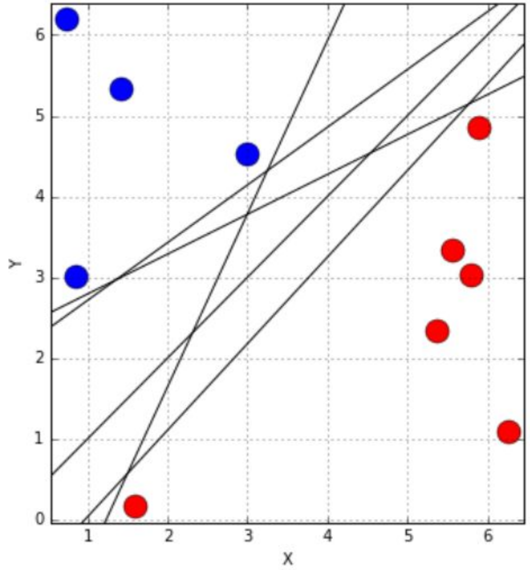
Christian Oliva Moya
Pedro Ramón Ventura Gómez

Support Vector Machines - Fundamentos

- Quiero hacer una **clasificación** de un problema **separable linealmente**

  Todas las soluciones son válidas

- Sin embargo, ¿cuál es el hiperplano óptimo?



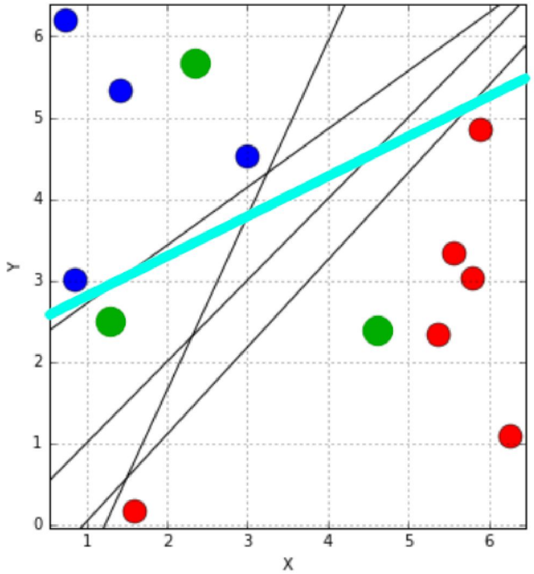
Support Vector Machines - Fundamentos

- Quiero hacer una **clasificación** de un problema **separable linealmente**

  Todas las soluciones son válidas

- Sin embargo, ¿cuál es el hiperplano óptimo?

- ¿Qué pasa en el ejemplo?



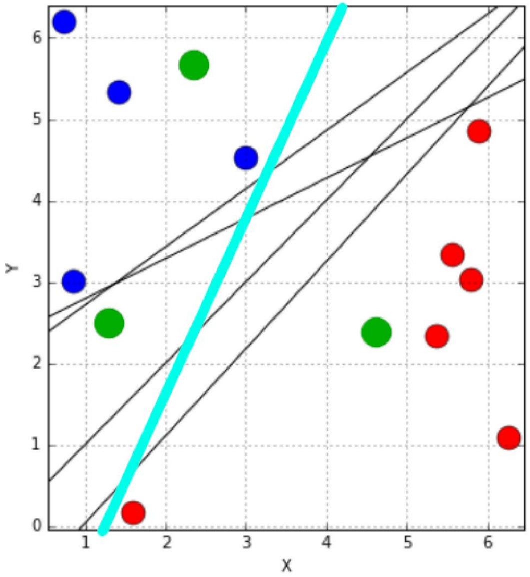
Support Vector Machines - Fundamentos

- Quiero hacer una **clasificación** de un problema **separable linealmente**

  Todas las soluciones son válidas

- Sin embargo, ¿cuál es el hiperplano óptimo?

- ¿Y ahora? ¿Qué pasa en el ejemplo?



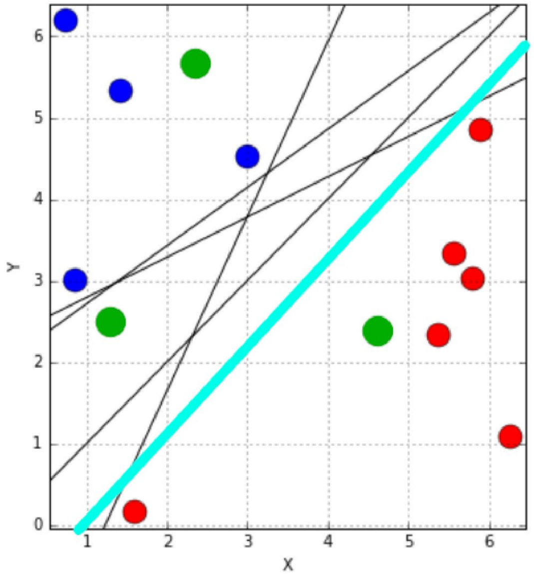
Support Vector Machines - Fundamentos

- Quiero hacer una **clasificación** de un problema **separable linealmente**

  Todas las soluciones son válidas

- Sin embargo, ¿cuál es el hiperplano óptimo?

- ¿Y ahora? ¿Qué pasa en el ejemplo?



Support Vector Machines - Fundamentos

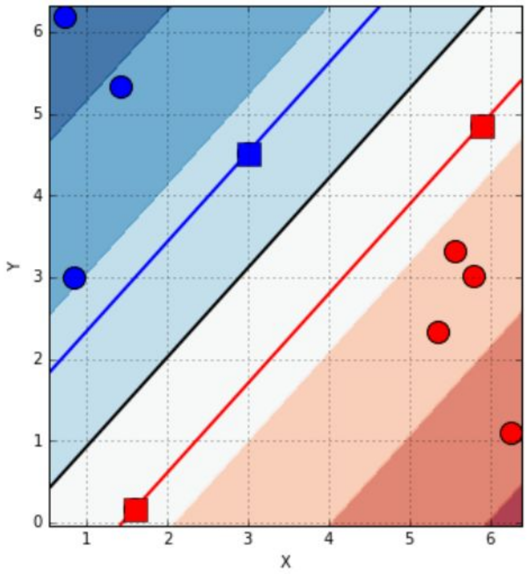
- Quiero hacer una **clasificación** de un problema **separable linealmente**

  Todas las soluciones son válidas

- Sin embargo, ¿cuál es el hiperplano óptimo?

- Parece razonable **maximizar el margen**

  (maximizar la mínima distancia desde cada punto al umbral)

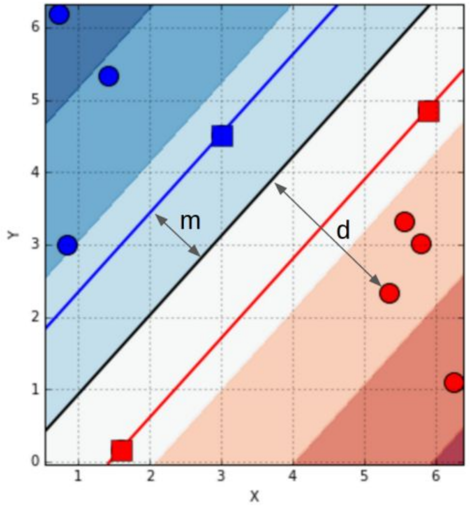


Support Vector Machines - Fundamentos

- Queremos encontrar el hiperplano **xw** + b = 0 que maximiza el margen

- La distancia de cada punto $\mathbf{x}_i$ al hiperplano es:

$$d = \frac{|\mathbf{x}_i \mathbf{w} + b|}{||\mathbf{w}||}$$



- Si utilizamos el hiperplano canónico |**xw** + b| = 1

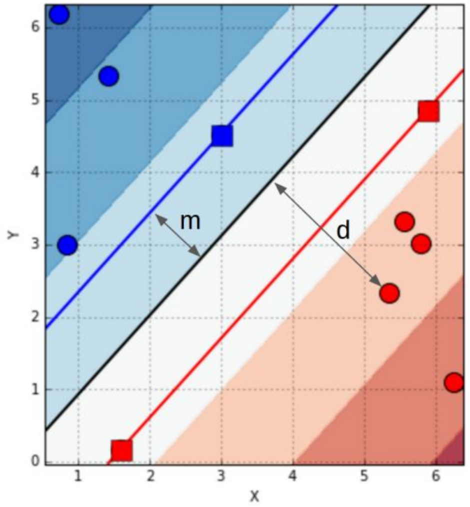
- Tenemos que la distancia al margen para los puntos más cercanos es: $$m = \frac{1}{||\mathbf{w}||}$$

Support Vector Machines - Fundamentos

- Por tanto, maximizar el margen $m = \frac{1}{||\mathbf{w}||}$ es equivalente a

  Minimizar la función:

  $$L(\mathbf{w}) = ||\mathbf{w}||$$



- Por conveniencia para más adelante usaremos:

  $$L(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2$$

- Sujeto a la restricción:

  $$t_i(\mathbf{x}_i\mathbf{w} + b) \geq 1 \quad \forall i$$

Support Vector Machines - Fundamentos

- Para resolver este problema de minimización utilizamos un multiplicador de Lagrange para cada punto:

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{n} \alpha_i [t_i(\mathbf{x}_i \mathbf{w} + b) - 1]$$

- La solución se puede obtener optimizando la función sujeta a las condiciones Karush-Kuhn-Tucker (KKT):

$$\alpha_i \geq 0$$

$$t_i(\mathbf{x}_i \mathbf{w} + b) - 1 \geq 0$$

$$\alpha_i [t_i(\mathbf{x}_i \mathbf{w} + b) - 1] = 0$$

Support Vector Machines - Fundamentos

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{n} \alpha_i [t_i(\mathbf{x}_i \mathbf{w} + b) - 1]$$

$$\alpha_i \geq 0$$

$$t_i(\mathbf{x}_i \mathbf{w} + b) - 1 \geq 0$$

$$\alpha_i [t_i(\mathbf{x}_i \mathbf{w} + b) - 1] = 0$$

- Si $\alpha_i = 0 \Rightarrow t_i(\mathbf{x}_i \mathbf{w} + b) - 1 > 0$ (restricción **inactiva**)
- Si $\alpha_i > 0 \Rightarrow t_i(\mathbf{x}_i \mathbf{w} + b) - 1 = 0$ (restricción **activa**)

Support Vector Machines - Fundamentos

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{n} \alpha_i [t_i(\mathbf{x}_i \mathbf{w} + b) - 1]$$

- Si calculamos los gradientes respecto a **w** y *b* igualando a 0 tenemos:

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i t_i \mathbf{x}_i$$

$$\sum_{i=1}^{n} \alpha_i t_i = 0$$

- Y podemos simplificar la función para que dependa solo de alpha

Support Vector Machines - Fundamentos

$$L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j t_i t_j \mathbf{x}_i \mathbf{x}_j$$

- Sujeto a las restricciones:

$$\alpha_i \geq 0$$

$$\sum_{i=1}^{n} \alpha_i t_i = 0$$

Support Vector Machines - Fundamentos

- Resumiendo. Volviendo a las condiciones KKT:

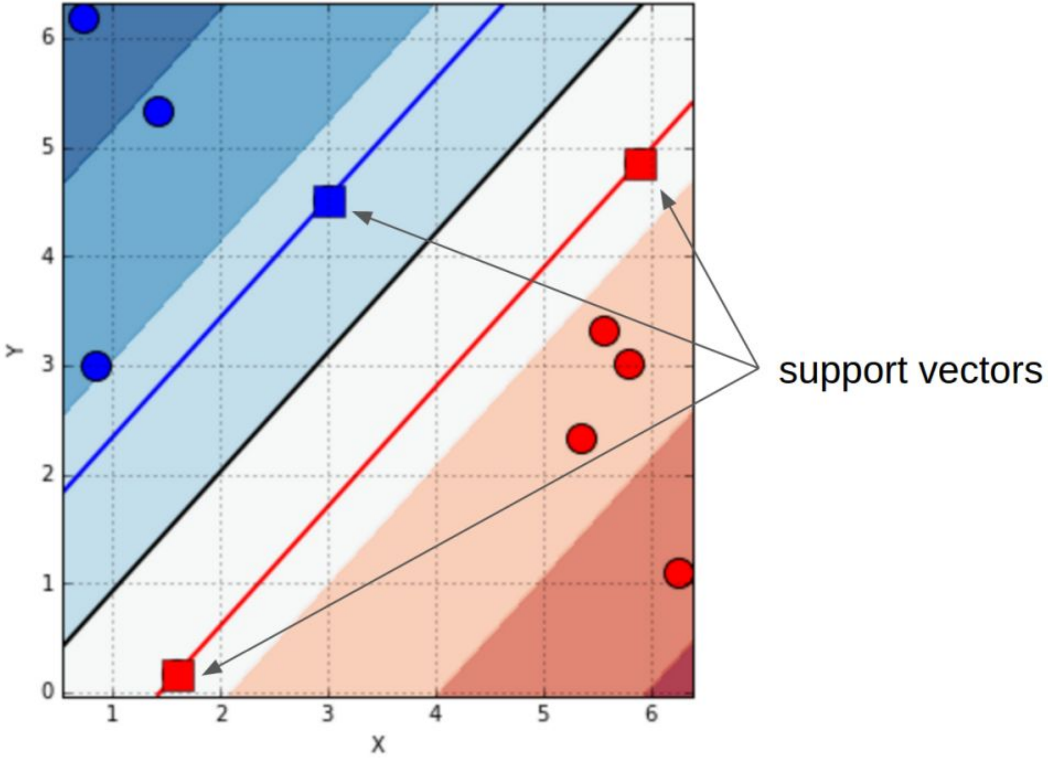
$$\alpha_i \geq 0$$

$$t_i(\mathbf{x}_i\mathbf{w} + b) - 1 \geq 0$$

$$\alpha_i[t_i(\mathbf{x}_i\mathbf{w} + b) - 1] = 0$$

- Si $\alpha_i = 0$ el punto $\mathbf{x}_i$ no contribuye en la separación del hiperplano
- Si $t_i(\mathbf{x}_i\mathbf{w} + b) = 1$ el punto $\mathbf{x}_i$ define el hiperplano (es un **support vector**)
- El vector $\mathbf{w} = \sum_{i=1}^{n} \alpha_i t_i \mathbf{x}_i$
- El parámetro *b* puede calcularse a partir de cualquier **support vector**

Support Vector Machines - Fundamentos

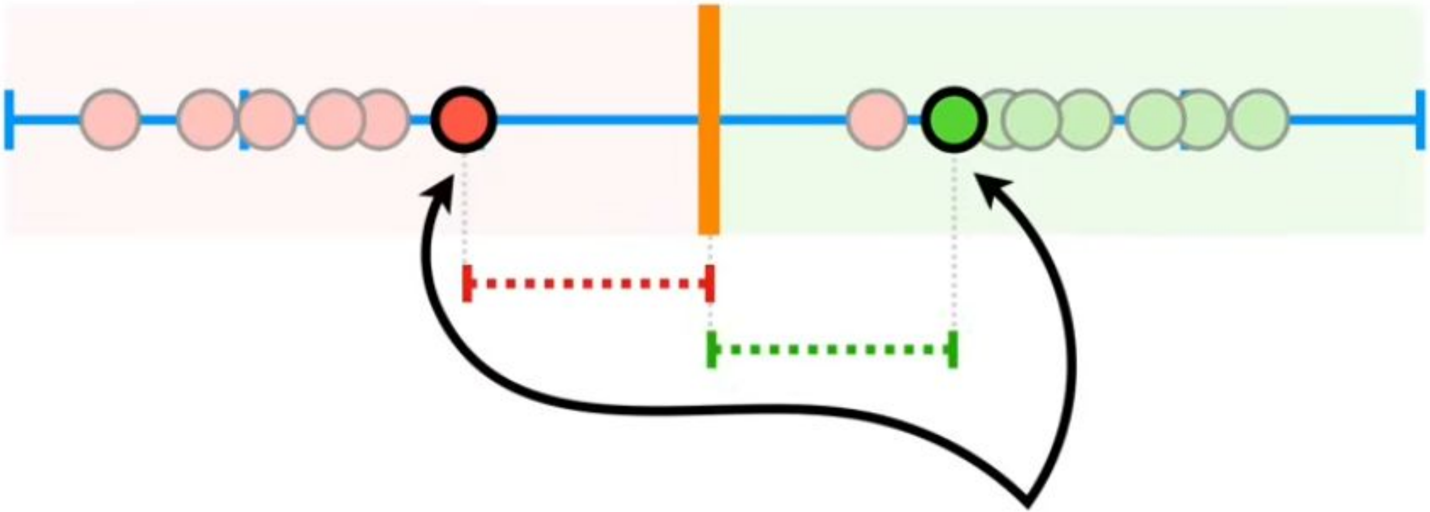


Support Vector Machines - Fundamentos

- Sin embargo, ¿qué sucede en este caso?

Support Vector Machines - Fundamentos

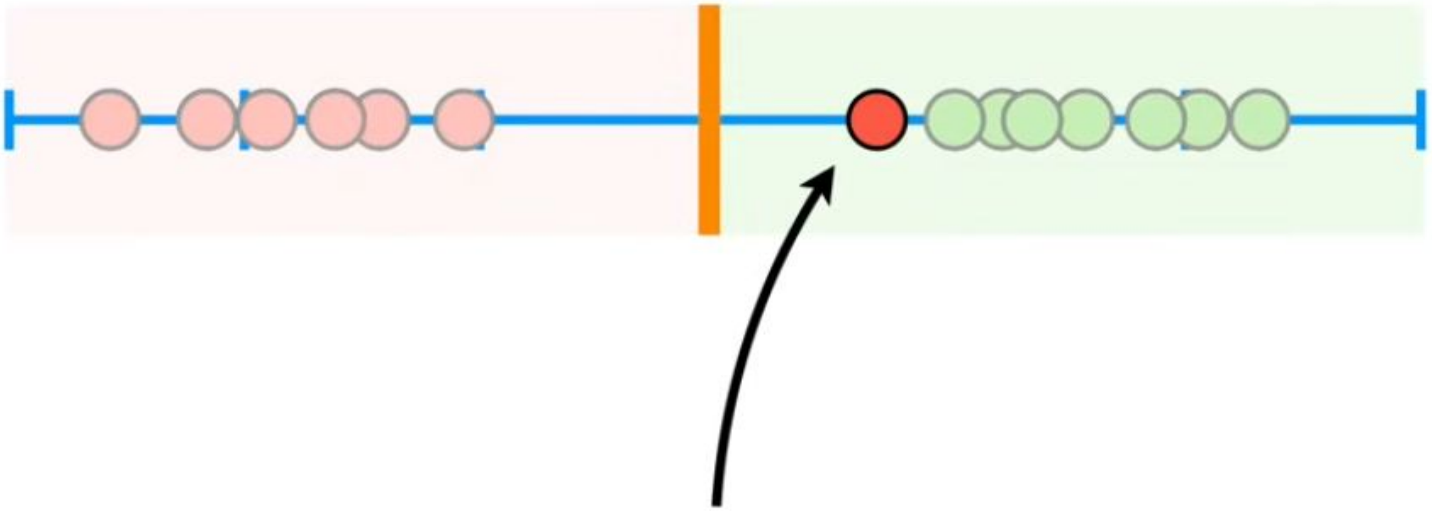
- Sin embargo, ¿qué sucede en este caso?



Querríamos algo así, ¿verdad?

Support Vector Machines - Fundamentos

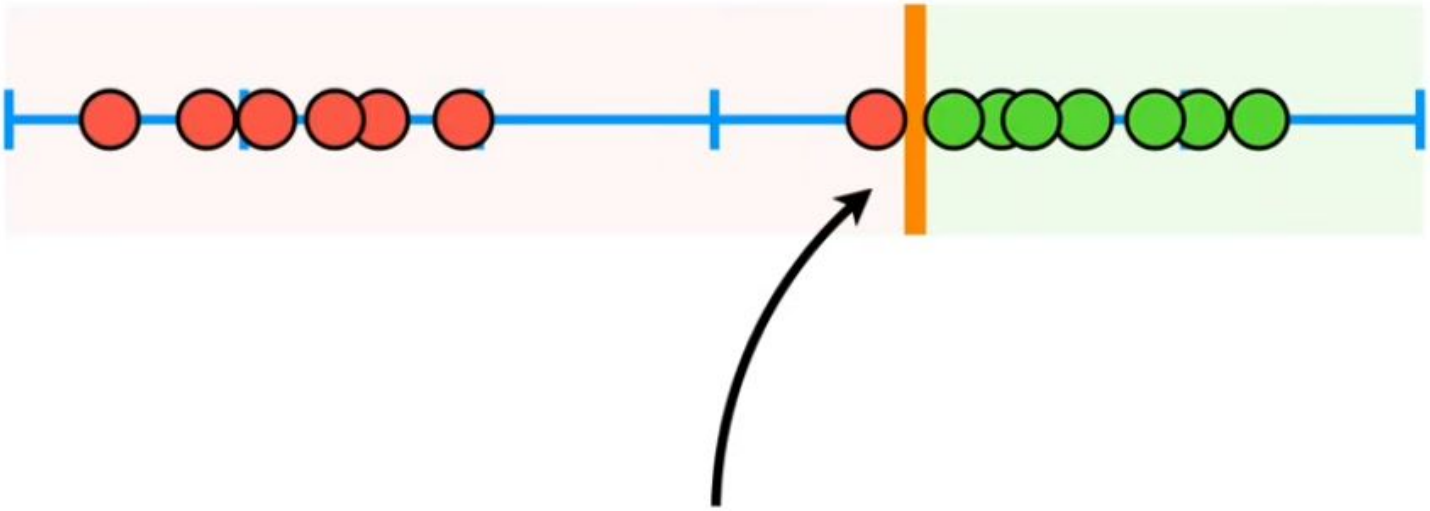
- Sin embargo, ¿qué sucede en este caso?



Querríamos ignorar este punto, que debe ser un **outlier**

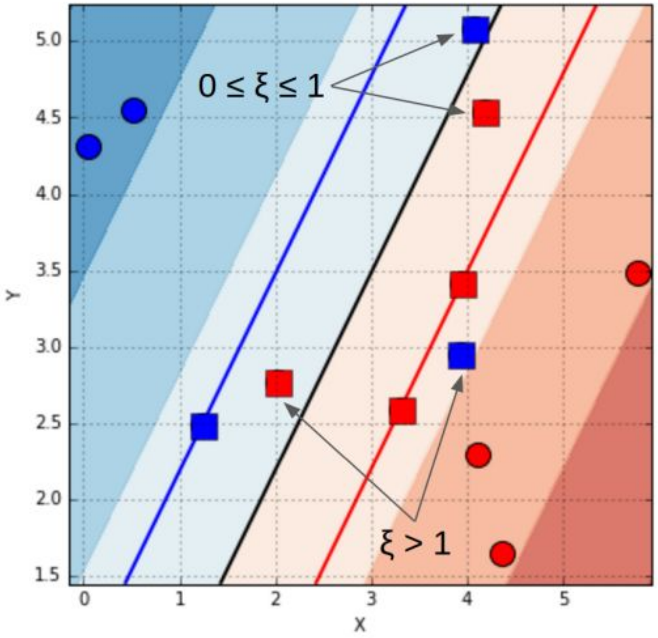
Support Vector Machines - Fundamentos

- Sin embargo, ¿qué sucede en este caso?



Pero si no hacemos nada, se obtendría este hiperplano…

Support Vector Machines - Fundamentos



- Necesitamos darle flexibilidad al modelo:
- Introducimos las **slack variables** $\xi_i \geq 0$
- Por tanto las restricciones ahora son:

$$t_i(\mathbf{x}_i\mathbf{w} + b) \geq 1 - \xi_i$$

- Si $\xi_i = 0$ el punto xi está fuera del margen y es correctamente clasificado
- Si $0 \leq \xi_i \leq 1$ el punto xi está dentro del margen y clasificado bien
- Si $\xi_i > 1$ el punto no se clasifica correctamente

Support Vector Machines - Fundamentos

- Si antes nuestro objetivo era minimizar $L(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2$

- Ahora nuestro objetivo es minimizar:

$$L(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i$$

- Sujeto a las restricciones:

$$t_i(\mathbf{x}_i\mathbf{w} + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

Support Vector Machines - Fundamentos

- Si antes nuestro objetivo era minimizar $L(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2$

- Ahora nuestro objetivo es minimizar:

$$L(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2 + \boxed{C}\sum_{i=1}^{n} \xi_i$$

C es un parámetro de regularización

C grande hace que haya menos errores

C pequeño reduce el overfitting

- Sujeto a las restricciones:

$$t_i(\mathbf{x}_i\mathbf{w} + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

Support Vector Machines - Fundamentos

- Lo más interesante es que nos queda la misma función a maximizar:

$$L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j t_i t_j \mathbf{x}_i \mathbf{x}_j$$

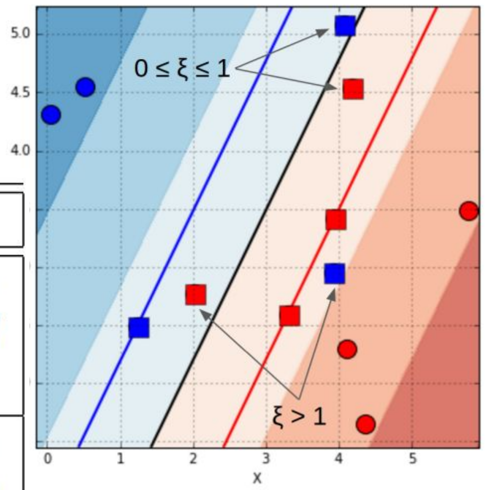
- Pero cambiando una de las restricciones:

ANTES	AHORA
$\alpha_i \geq 0$	$0 \leq \alpha_i \leq C$
$\sum_{i=1}^{n} \alpha_i t_i = 0$	$\sum_{i=1}^{n} \alpha_i t_i = 0$

Support Vector Machines - Fundamentos

- Tabla resumen:

α	$\mu = C - \alpha$	ξ	$t(\mathbf{w}^t\mathbf{x} + b)$	Type
$\alpha = 0$	$\mu > 0$	$\xi = 0$	$t(\mathbf{w}^t\mathbf{x} + b) > 1$	Well classified, <u>out</u> of the margin
$0 < \alpha < C$	$\mu > 0$	$\xi = 0$	$t(\mathbf{w}^t\mathbf{x} + b) = 1$	Well classified, <u>on</u> the margin
$\alpha = C > 0$	$\mu = 0$	$0 < \xi \leq 1$	$t(\mathbf{w}^t\mathbf{x} + b) \geq 0$	Well classified, <u>inside</u> the margin
		$\xi > 1$	$t(\mathbf{w}^t\mathbf{x} + b) < 0$	Wrongly classified point

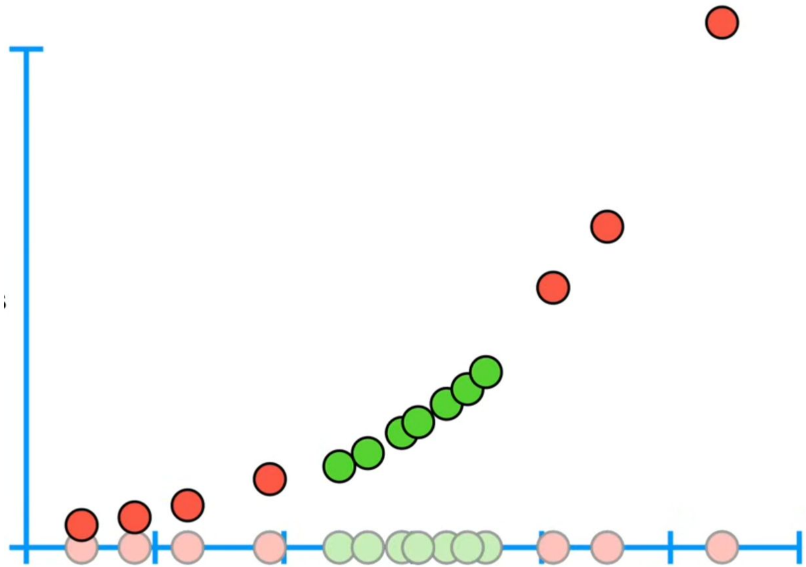


Support Vector Machines - Problemas no lineales

- ¿Toda esta historia para hacer un separador lineal?
- ¿Qué hago en una situación así?

Support Vector Machines - Problemas no lineales

- Gracias al **truco del Kernel** (Kernel trick), las SVMs pueden:
  - Realizar una proyección a un espacio dimensional mayor
  - Encontrar el separador lineal en ese espacio



Support Vector Machines - Problemas no lineales

- Métodos de Kernel: definimos una función de transformación tal que:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)\Phi(\mathbf{x}_j)^T$$

- Así, necesitamos encontrar el hiperplano óptimo en el espacio transformado:

$$\Phi(\mathbf{x})\mathbf{w} + b = 0$$

Support Vector Machines - Problemas no lineales

- Como antes, **w** está definido por los **support vectors** ($\alpha_i \neq 0$)

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i t_i \Phi(\mathbf{x}_i)$$

- El coeficiente *b* se obtiene utilizando cualquier **support vector** con $\alpha_i < C$

$$t_i(\Phi(\mathbf{x}_i)\mathbf{w} + b) = 1$$

- Para clasificar un nuevo patrón **x** debemos evaluar:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i t_i \Phi(\mathbf{x}_i)\Phi(\mathbf{x}) + b = \sum_{i=1}^{n} \alpha_i t_i k(\mathbf{x_i}, \mathbf{x}) + b$$

Support Vector Machines - Problemas no lineales

- Como antes, **w** está definido por los **support vectors** ($\alpha_i \neq 0$)

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i t_i \Phi(\mathbf{x}_i)$$

- El coeficiente *b* s**or** con $\alpha_i < C$

Gracias a esta propiedad del Kernel, podemos calcular la clasificación **sin hacer la transformación** al espacio de mayor dimensión

- Para clasificar ur

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \boxed{t_i \Phi(\mathbf{x}_i)\Phi(\mathbf{x})} + b = \sum_{i=1}^{n} \alpha_i t_i \boxed{k(\mathbf{x_i}, \mathbf{x})} + b$$

Support Vector Machines - Kernels típicos

- **Kernel polinómico**

$$k(a, b) = (a \times b + r)^d$$

Donde r es el coeficiente del polinomio y d es el grado

Vamos a desarrollarlo con un ejemplo: r = ½ y d = 2

$$(a \times b + \frac{1}{2})^2 = (a \times b + \frac{1}{2})(a \times b + \frac{1}{2})$$

$$= ab + a^2b^2 + \frac{1}{4}$$

$$= (a, a^2, \frac{1}{2})(b, b^2, \frac{1}{2})$$

Support Vector Machines - Kernels típicos

- **Kernel polinómico**

$$k(a,b) = (a \times b + r)^d$$

Donde r es el coeficiente del polinomio y d es el grado

Vamos a desarrollarlo con un ejemplo: r = ½ y d = 2

$$(a \times b + \frac{1}{2})^2 = (a \times b + \frac{1}{2})(a \times b + \frac{1}{2})$$

$$= ab + a^2b^2 + \frac{1}{4}$$

$$= (a, a^2, \frac{1}{2})(b, b^2, \frac{1}{2})$$

La transformación es esta

$$\boxed{\Phi(x) = (x, x^2, \frac{1}{2})}$$

Support Vector Machines - Kernels típicos

- **Kernel RBF** (Radial Basis Function)

$$k(a,b) = e^{-\gamma(a-b)^2}$$

Donde gamma escala la distancia cuadrática entre *a* y *b*

- Este Kernel encuentra los **support vectors** en una dimensión infinita

$$k(a,b) = e^{-\gamma(a-b)^2} = e^{-\gamma(a^2+b^2-2ab)} = e^{-\gamma(a^2+b^2)}e^{\gamma 2ab}$$

Vamos a verlo con un ejemplo: $\gamma = \frac{1}{2}$

Support Vector Machines - Kernels típicos

- **Kernel RBF** (Radial Basis Function)

$$k(a,b) = e^{-\gamma(a-b)^2} = e^{-\gamma(a^2+b^2-2ab)} = e^{-\gamma(a^2+b^2)}e^{\gamma 2ab}$$

$$\gamma = \frac{1}{2}$$

- Nos queda:

$$k(a,b) = e^{\frac{1}{2}(a-b)^2} = e^{-\frac{1}{2}(a^2+b^2)}e^{ab}$$

Support Vector Machines - Kernels típicos

- **Kernel RBF** (Radial Basis Function)

$$k(a,b) = e^{-\gamma(a-b)^2} = e^{-\gamma(a^2+b^2-2ab)} = e^{-\gamma(a^2+b^2)}e^{\gamma 2ab}$$

$$\gamma = \frac{1}{2}$$

- Nos queda:

$$k(a,b) = e^{\frac{1}{2}(a-b)^2} = e^{-\frac{1}{2}(a^2+b^2)}\boxed{e^{ab}}$$

Aplicamos la expansión
en Serie de Taylor

Support Vector Machines - Kernels típicos

- **Kernel RBF** (Radial Basis Function)

$$k(a,b) = e^{-\gamma(a-b)^2} = e^{-\gamma(a^2+b^2-2ab)} = e^{-\gamma(a^2+b^2)} e^{\gamma 2ab}$$

$$\gamma = \frac{1}{2}$$

- Nos queda:

$$k(a,b) = e^{\frac{1}{2}(a-b)^2} = e^{-\frac{1}{2}(a^2+b^2)} \boxed{e^{ab}}$$

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

Support Vector Machines - Kernels típicos

- **Kernel RBF** (Radial Basis Function)

$$k(a,b) = e^{-\gamma(a-b)^2} = e^{-\gamma(a^2+b^2-2ab)} = e^{-\gamma(a^2+b^2)}e^{\gamma 2ab}$$

$$\gamma = \frac{1}{2}$$

- Nos queda:

$$k(a,b) = e^{\frac{1}{2}(a-b)^2} = e^{-\frac{1}{2}(a^2+b^2)}\boxed{e^{ab}}$$

$$\boxed{e^{ab} = 1 + \frac{1}{1!}ab + \frac{1}{2!}(ab)^2 + \frac{1}{3!}(ab)^3 + \ldots + \frac{1}{\infty!}(ab)^\infty}$$

Support Vector Machines - Ventajas

- Ventajas de usar SVMs:
  - **No hay mínimos locales** ya que es un problema cuadrático
  - La solución óptima se puede encontrar en **tiempo polinómico**
  - **Hay pocos hiperparámetros**: C, el tipo de Kernel y los parámetros del Kernel (se pueden buscar haciendo una validación cruzada típica)
  - **Solución estable** (no depende de inicialización aleatoria)
  - **Buena capacidad de generalización**