Machine Learning 101

SVMs y Métodos Kernel

Introducción

- Máquinas de vectores (de) soporte, del inglés, *Support Vector Machines*

- Inicialmente concebidas para problemas de clasificación, y posteriormente extendidas a regresión.
  - SVC: *Support Vector Classification*
  - SVR: *Support Vector Regression*

- Se definen como **clasificadores lineales de máximo margen**

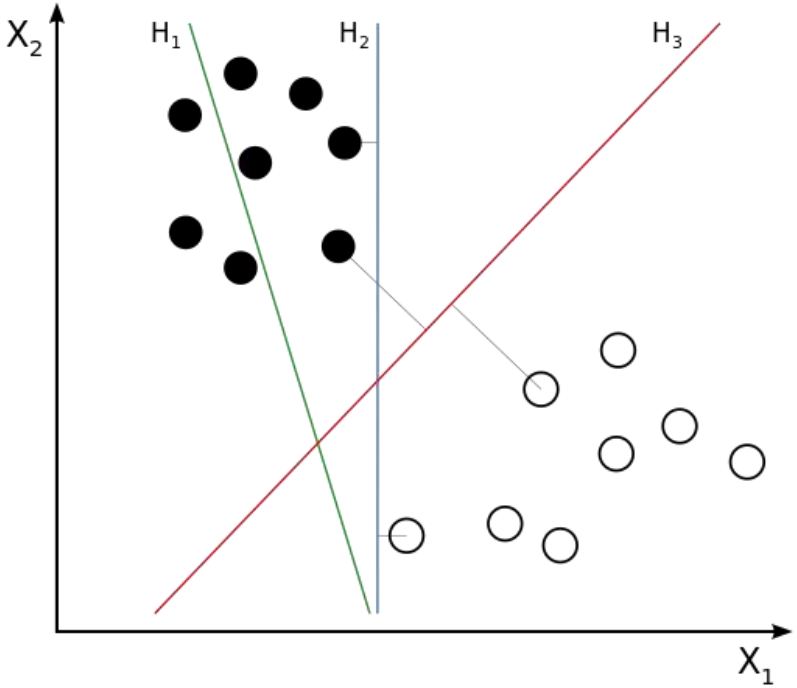
- Propuestas a mediados-finales de los 90s, con mucho auge en los 2000s
  - Grandes prestaciones en aprendizaje supervisado
  - Métodos Kernel

Índice

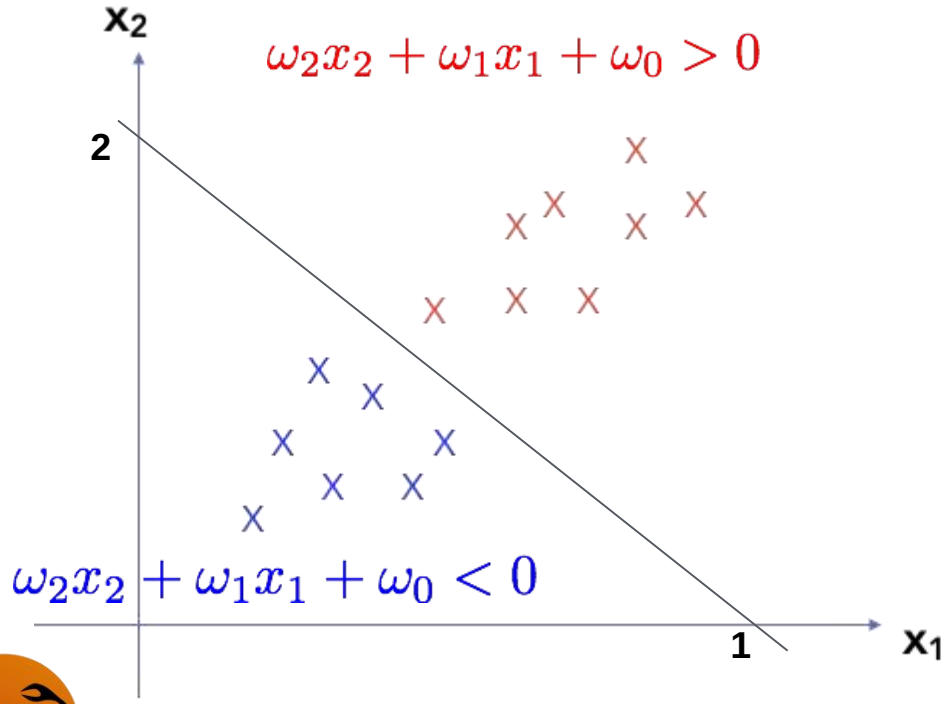
1. **Intuición: el (hiper)plano separador**
2. ¿Por qué *Support Vector*?
3. Caso no linealmente separable
4. SVMs en regresión
5. SVMs y selección de características
6. K-nn en regresión
7. Kernels y SVM
8. Otros algoritmos con Kernels

Intuición

- Clasificador lineal definido por un hiperplano separador de máximo margen



Plano separador



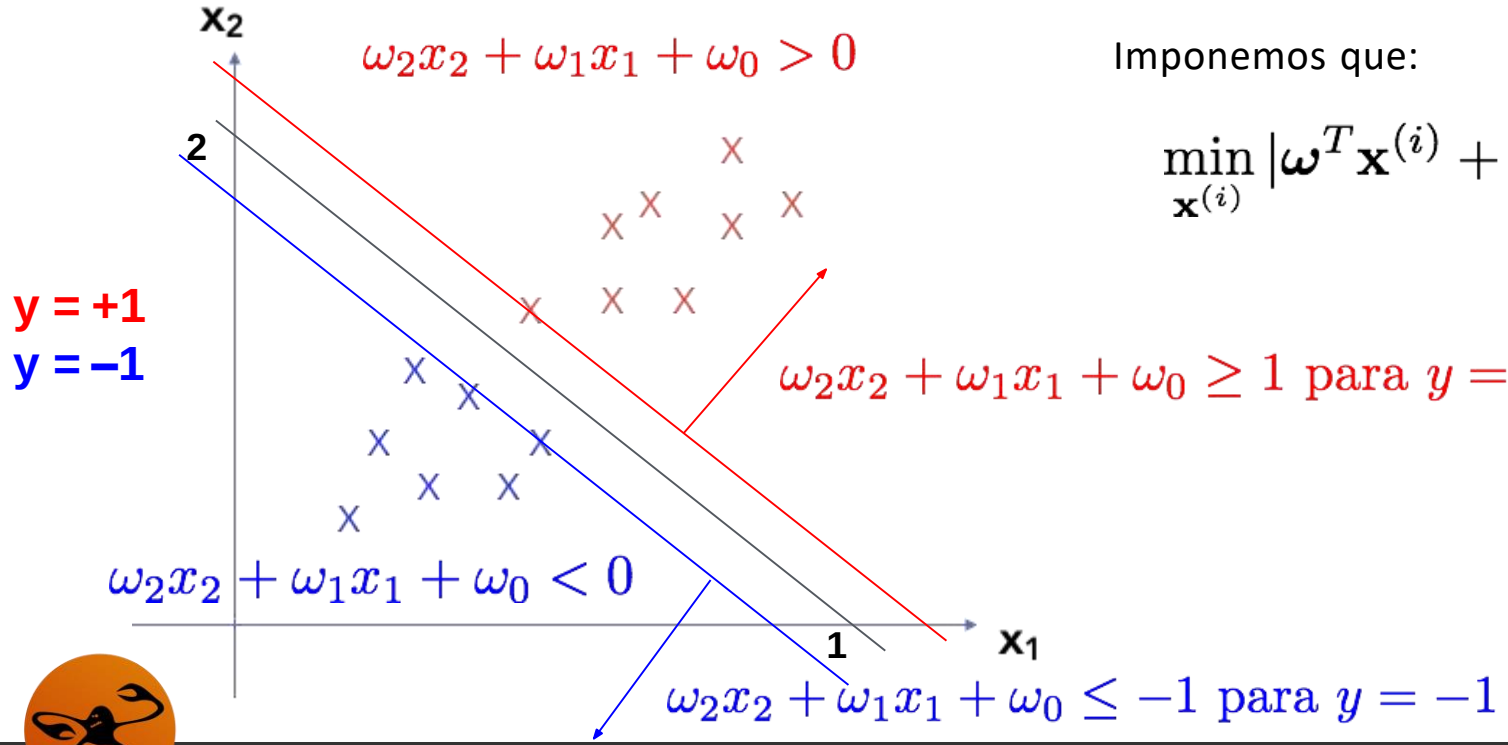
$$x_2 = mx_1 + n$$
$$x_2 = -2x_1 + 2$$
$$x_2 + 2x_1 - 2 = 0$$

$$\omega_2 x_2 + \omega_1 x_1 + \omega_0 = 0$$
$$\boldsymbol{\omega}^T \mathbf{x} + \omega_0 = 0$$

¿y cómo fuerzo a que sea de máximo margen?

Plano separador: condición

$$\omega_2 x_2 + \omega_1 x_1 + \omega_0 = 0$$



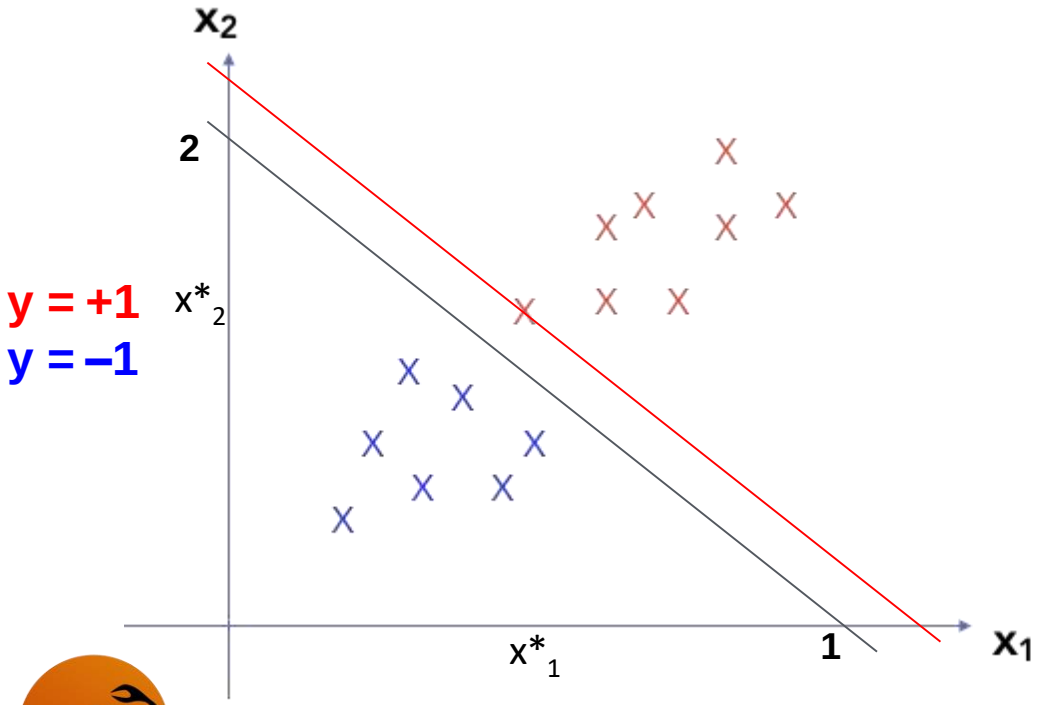
Imponemos que:

$$\min_{\mathbf{x}^{(i)}} |\boldsymbol{\omega}^T \mathbf{x}^{(i)} + \omega_0| = 1$$

$$\omega_2 x_2 + \omega_1 x_1 + \omega_0 \geq 1 \text{ para } y = +1$$

$$\omega_2 x_2 + \omega_1 x_1 + \omega_0 \leq -1 \text{ para } y = -1$$

Plano separador: justificación



Si

$$\omega_2 x_2 + \omega_1 x_1 + \omega_0 = 0$$

es un plano separador, entonces

$$c \cdot (\omega_2 x_2 + \omega_1 x_1 + \omega_0) = 0$$

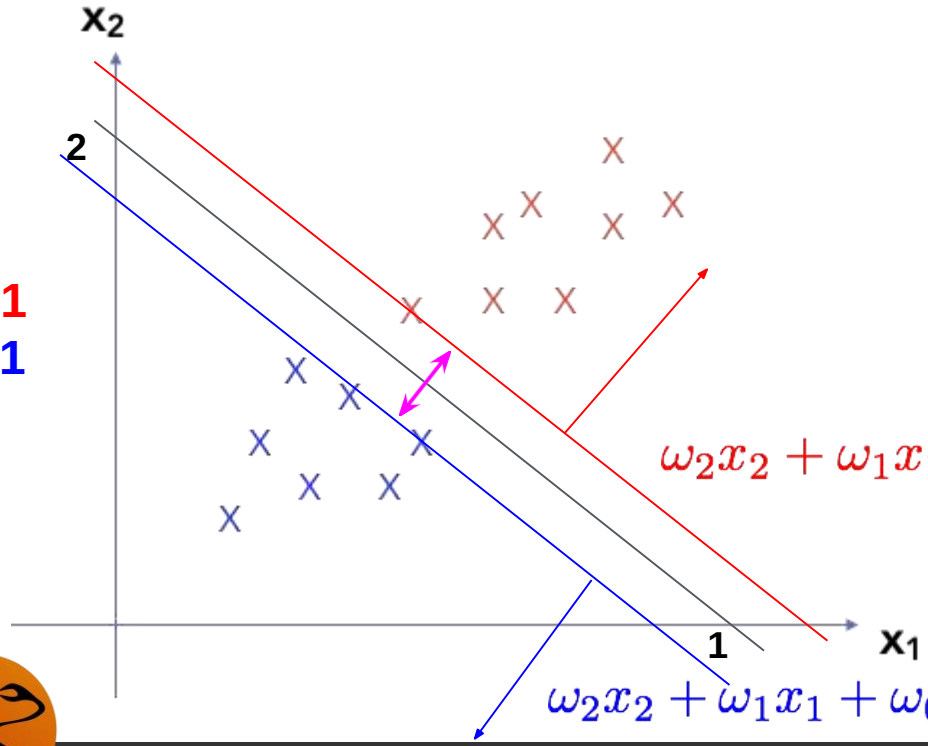
también lo es. Así, escojo *c* para que se cumpla la condición que me interesa.

Ej: supongamos que x^*_1=0.5, y x^*_2= 1.1, para el que quiero que

$$c \cdot (\omega_2 x_2^* + \omega_1 x_1^* + \omega_0) = 1$$

Entonces ¿cuánto vale c?

Plano separador: margen



El objetivo es **maximizar** el margen

$$\text{margen} = \frac{2}{||\boldsymbol{\omega}||}$$

bajo las condiciones anteriores, que pueden expresarse como

$$y\left(\omega_2 x_2 + \omega_1 x_1 + \omega_0\right) \geq 1$$

$$y\left(\boldsymbol{\omega}^T \mathbf{x} + \omega_0\right) \geq 1$$

$$\omega_2 x_2 + \omega_1 x_1 + \omega_0 \geq 1 \text{ para } y = +1$$

$$\omega_2 x_2 + \omega_1 x_1 + \omega_0 \leq -1 \text{ para } y = -1$$

Función de coste

- Clasificador lineal definido por un hiperplano separador de máximo margen
- Así, dado un conjunto de datos etiquetados

$$\{\mathbf{x}^{(i)}, y^{(i)}\} \text{ con } \mathbf{x}^{(i)} \in \mathbb{R}^d, y^{(i)} \in \{-1, +1\}$$

- El funcional a **minimizar** es

$$\min_{\boldsymbol{\omega}, \omega_0} \frac{1}{2}||\boldsymbol{\omega}||_2^2 \text{ s.to } y^{(i)}\left(\boldsymbol{\omega}^T\mathbf{x}^{(i)} + \omega_0\right) \geq 1, i = 1, \ldots, N$$

  - Problema de optimización **convexa** (solución única) ...
  - … con restricciones (Lagrangiano)

SVMs vs Regresión logística

- Clasificadores lineales los dos
- Máximo margen vs Mínimo error
- Optimización convexa vs Máxima verosimilitud
- ...

Índice

1. Intuición: el (hiper)plano separador
2. **¿Por qué *Support Vector*?**
3. Caso no linealmente separable
4. SVMs en regresión
5. SVMs y selección de características
6. K-nn en regresión
7. Kernels y SVM
8. Otros algoritmos con Kernels

Support Vectors

- Se puede demostrar que la solución es

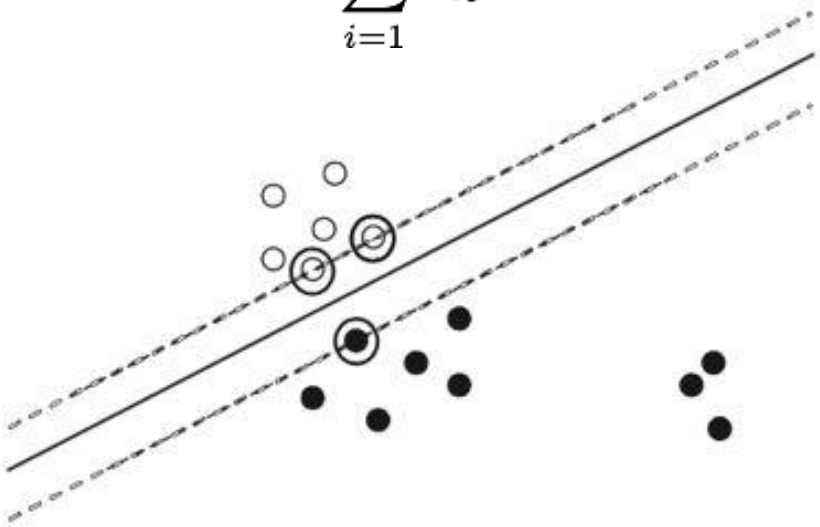
$$\hat{\omega} = \sum_{i=1}^{N} \alpha_i y^{(i)} \mathbf{x}^{(i)}$$

  - Una combinación lineal de las muestras de entrenamiento
  - $\alpha_i \geq 0$ pero para muchas muestras se cumple que $\alpha_i = 0$ (**solución dispersa**)
  - Vectores soporte: muestras para las que $\alpha_i \neq 0$

Support Vectors

- Se puede demostrar que la solución es

$$\hat{\boldsymbol{\omega}} = \sum_{i=1}^{N} \alpha_i y^{(i)} \mathbf{x}^{(i)}$$

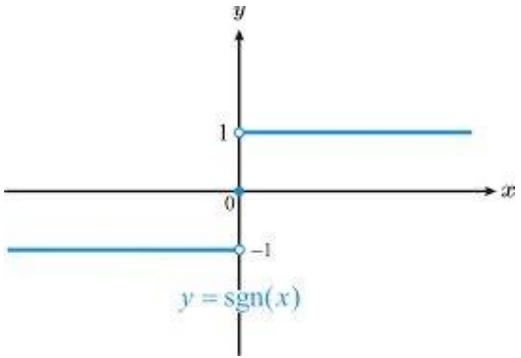


Frontera de separación

- Conocidos los pesos, la predicción se realiza a través de la fórmula

$$\hat{y} = f(\mathbf{x}) = \text{sign}\left(\hat{\boldsymbol{\omega}}^T \mathbf{x} + \hat{\omega}_0\right) = \text{sign}\left(\sum_{i=1}^{N} \alpha_i y^{(i)} \mathbf{x}^T \mathbf{x}^{(i)} + \hat{\omega}_0\right)$$

  - Producto escalar entre las muestras de entrenamiento y la muestra sobre la que quiero realizar la predicción

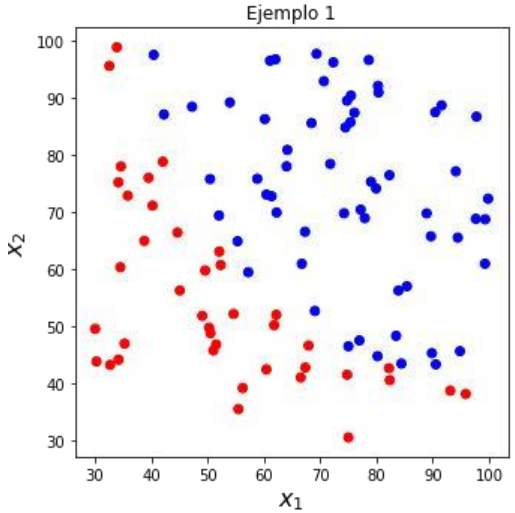


Índice

1. Intuición: el (hiper)plano separador
2. ¿Por qué *Support Vector*?
3. **Caso no linealmente separable**
4. SVMs en regresión
5. SVMs y selección de características
6. K-nn en regresión
7. Kernels y SVM
8. Otros algoritmos con Kernels

Caso linealmente no separable

- Hasta ahora hemos trabajado con un caso en el que las clases son claramente separables, esto es, no hay solapamiento entre ellas
- No hablamos de fronteras no lineales, seguimos considerando que existe un hiperplano capaz de separar las clases, aunque con errores

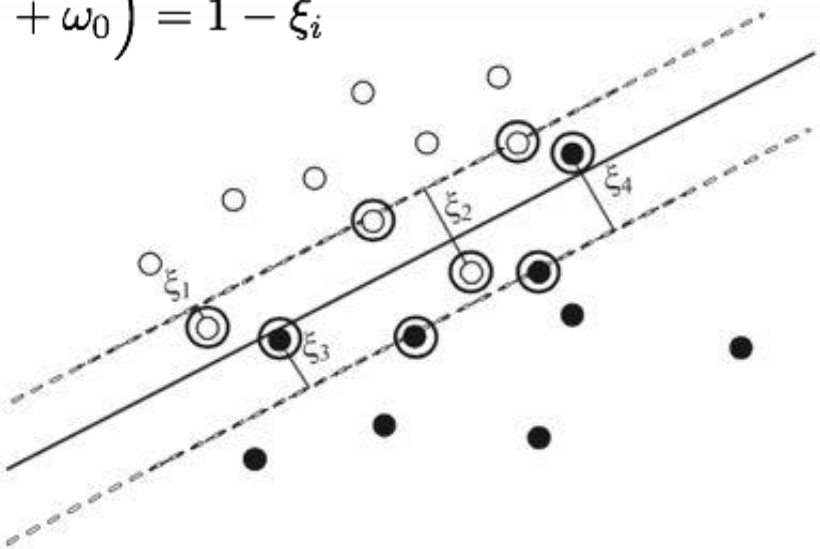


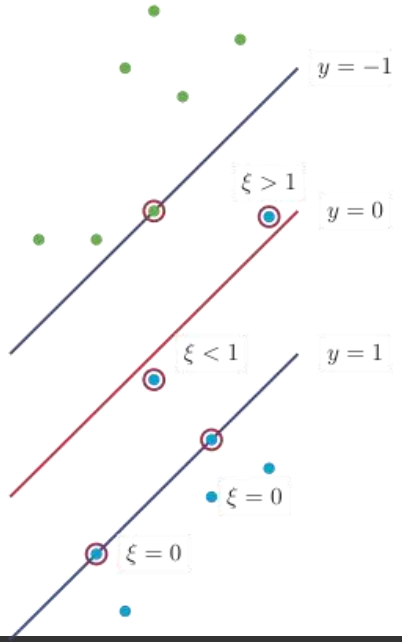
Caso linealmente no separable

- Voy a permitir errores: muestras **dentro del margen o mal clasificadas**
- Exclusivamente esas muestras les asigno un error (*slack* variable)

$$y^{(i)}\left(\boldsymbol{\omega}^T\mathbf{x}^{(i)}+\omega_0\right)=1-\xi_i$$

$$\text{con } \xi_i \geq 0$$





Caso linealmente no separable

- … pero penalizo los errores, con un coste C, ¿os suena?

$$\min_{\boldsymbol{\omega},\omega_0,\xi_i} \frac{1}{2}||\boldsymbol{\omega}||_2^2 + C\sum_{i=1}^{N}\xi_i$$

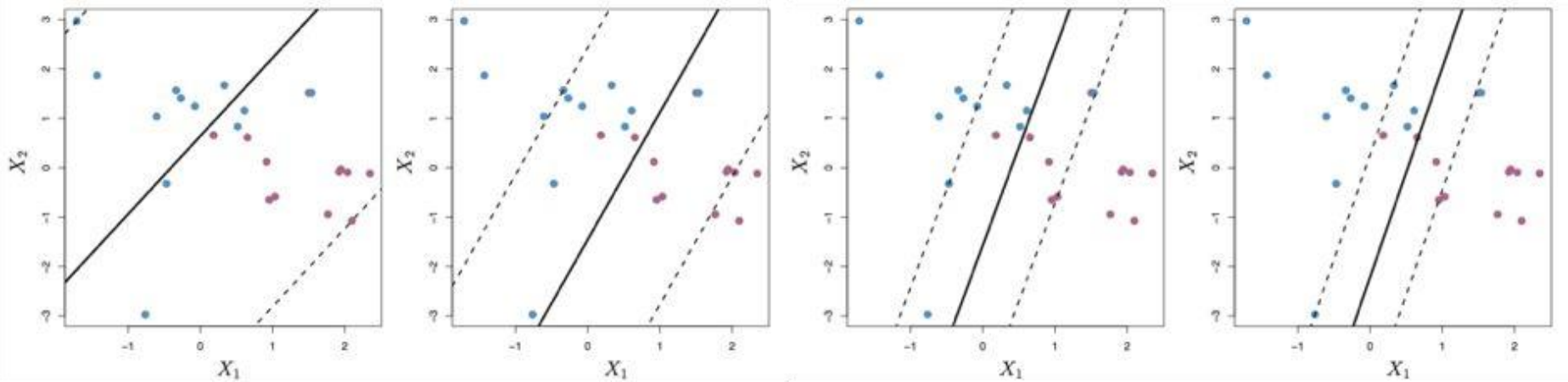
$$\text{s.to } y^{(i)}\left(\boldsymbol{\omega}^T\mathbf{x}^{(i)} + \omega_0\right) \geq 1 - \xi_i, i = 1, \ldots, N$$

$$\xi_i \geq 0.$$

- … regularización

Parámetro de regularización C

- C: cota superior al número de errores
- Compromiso entre margen y errores en la solución



- Si C elevado, margen estrecho, más peso a los errores. Alta complejidad
- Si C pequeño, margen ancho, menos peso a los errores. Baja complejidad

Caso linealmente no separable

- La solución no cambia con respecto al caso anterior

$$\hat{\boldsymbol{\omega}} = \sum_{i=1}^{N} \alpha_i y^{(i)} \mathbf{x}^{(i)}$$

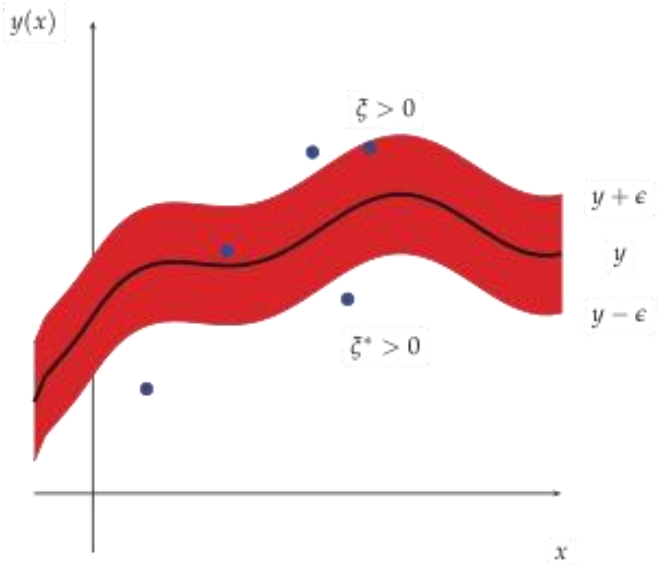
$$\hat{y} = f(\mathbf{x}) = \text{sign}\left(\hat{\boldsymbol{\omega}}^T \mathbf{x} + \hat{\omega}_0\right) = \text{sign}\left(\sum_{i=1}^{N} \alpha_i y^{(i)} \mathbf{x}^T \mathbf{x}^{(i)} + \hat{\omega}_0\right)$$

Índice

1. Intuición: el (hiper)plano separador
2. ¿Por qué *Support Vector*?
3. Caso no linealmente separable
4. **SVMs en regresión**
5. SVMs y selección de características
6. K-nn en regresión
7. Kernels y SVM
8. Otros algoritmos con Kernels

SVR: intuición

- Buscar el hiperplano que mejor se ajuste a los datos y permita un tolerancia a los errores
  - En otras palabras, regresión lineal, con restricciones



SVR: formulación

- Queremos que nuestra solución sea de la forma

$$y = f(\mathbf{x}) = \boldsymbol{\omega}^T \mathbf{x} + \omega_0$$

y permitir errores dentro del "margen": $[y - \epsilon, y + \epsilon]$, así que el funcional a minimizar es similar al problema de clasificación

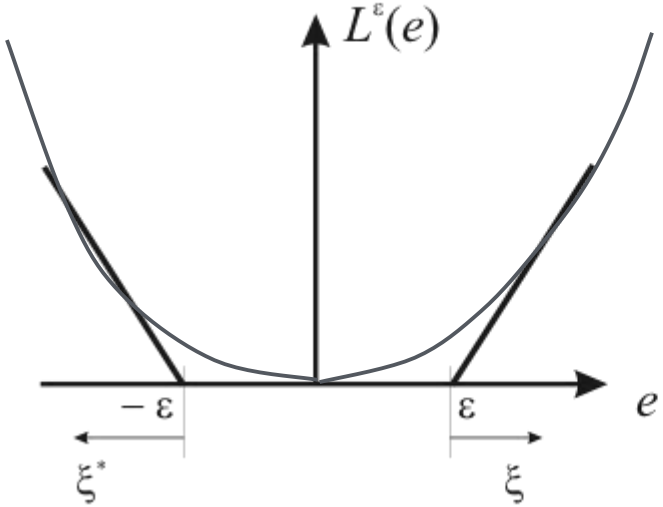
$$\min_{\boldsymbol{\omega}, \omega_0} \frac{1}{2} ||\boldsymbol{\omega}||_2^2$$

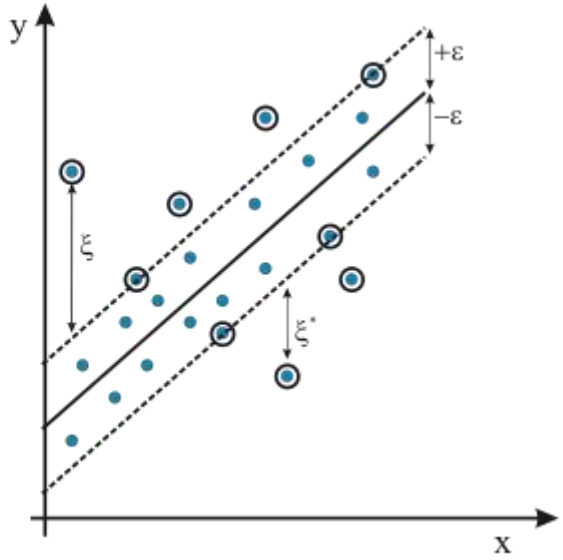
$$\text{s.to } y^{(i)} - \left(\boldsymbol{\omega}^T \mathbf{x}^{(i)} + \omega_0 \right) \leq \epsilon$$

$$-y^{(i)} + \left(\boldsymbol{\omega}^T \mathbf{x}^{(i)} + \omega_0 \right) \leq \epsilon$$

SVR: formulación

- pero, ¿qué hago con las muestras que caen fueran del margen?
- Las penalizo





SVR: formulación

- Regresión lineal, con función de coste e-insensible

$$\min_{\boldsymbol{\omega},\omega_0,\xi_i,\xi_i^*} \frac{1}{2}||\boldsymbol{\omega}||_2^2 + C\sum_{i=1}^{N} L^{\epsilon}(y^{(i)} - \hat{y}^{(i)}) = \min_{\boldsymbol{\omega},\omega_0,\xi_i,\xi_i^*} \frac{1}{2}||\boldsymbol{\omega}||_2^2 + C\sum_{i=1}^{N} (\xi_i + \xi_i^*)$$

sujeto a

$$y^{(i)} - \left(\boldsymbol{\omega}^T \mathbf{x}^{(i)} + \omega_0\right) \leq \epsilon + \xi_i$$

$$-y^{(i)} + \left(\boldsymbol{\omega}^T \mathbf{x}^{(i)} + \omega_0\right) \leq \epsilon + \xi_i^*$$

$$\xi_i, \xi_i \geq 0$$

SVR: solución

- Coeficientes: $\hat{\boldsymbol{\omega}} = \sum_{i=1}^{N} \alpha_i \mathbf{x}^{(i)}$
  - Combinación lineal de las muestras de entrenamiento

- Regresión: $\hat{y} = f(\mathbf{x}) = \hat{\boldsymbol{\omega}}^T \mathbf{x} + \hat{\omega}_0 = \sum_{i=1}^{N} \alpha_i \mathbf{x}^T \mathbf{x}^{(i)} + \hat{\omega}_0$
  - Producto escalar entre las muestras de entrenamiento y la muestra sobre la que quiero realizar la predicción

Índice

1. Intuición: el (hiper)plano separador
2. ¿Por qué *Support Vector*?
3. Caso no linealmente separable
4. SVMs en regresión
5. **SVMs y selección de características**
6. K-nn en regresión
7. Kernels y SVM
8. Otros algoritmos con Kernels

Recursive Feature Elimination

- Método *wrapper*
- Originalmente propuesto para SVM, analizando los coeficientes del modelo

¿Entiendes el algoritmo?

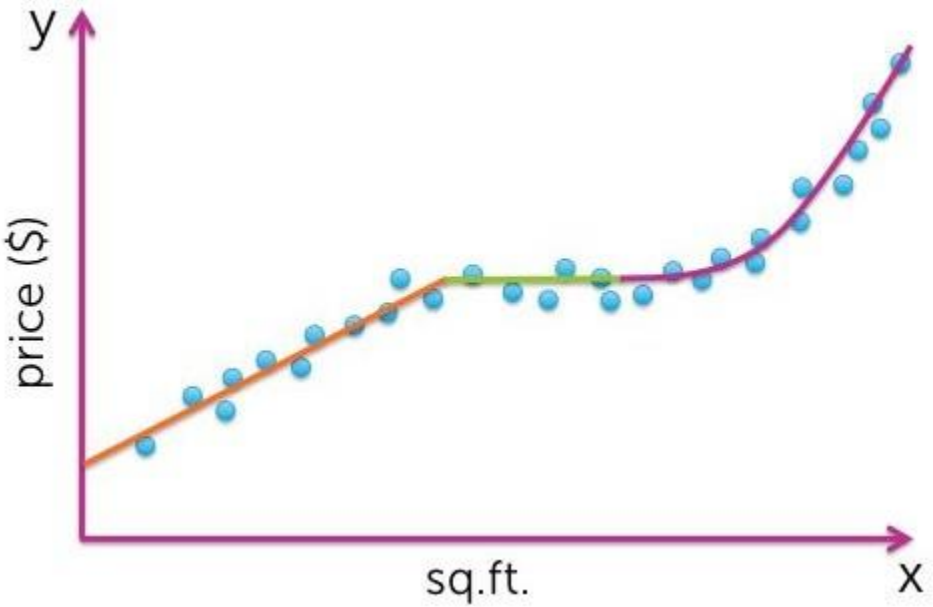
- En sklearn, extendido a otros algoritmos con indicadores de relevancia, como coeficientes o importancia de variables
  - Regresión lineal, logística, Ridge, Lasso
  - Algoritmos basados en árboles

Índice

1. Intuición: el (hiper)plano separador
2. ¿Por qué *Support Vector*?
3. Caso no linealmente separable
4. SVMs en regresión
5. SVMs y selección de características
6. **K-nn en regresión**
7. Kernels y SVM
8. Otros algoritmos con Kernels

Motivación

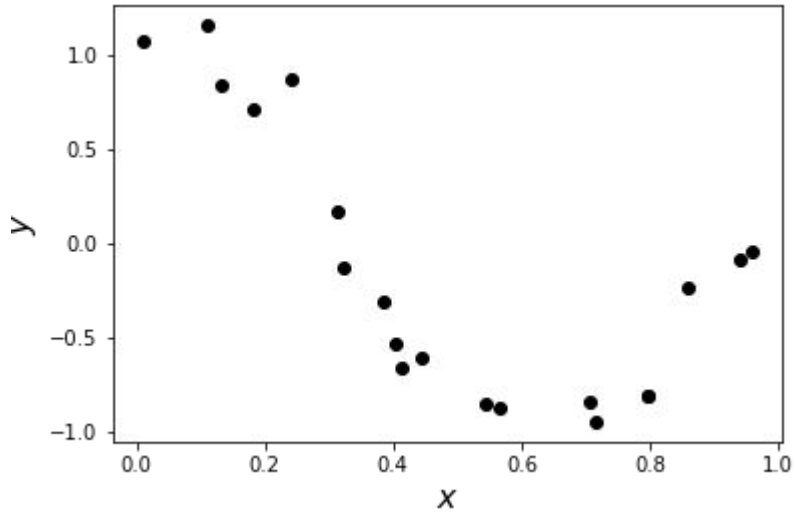
- Modelos paramétrico no siempre resultan adecuados
  - Modelo basado en datos, no paramétrico



$$\hat{y} \neq \omega_0 + \omega_1 x + \omega_2 x^2$$

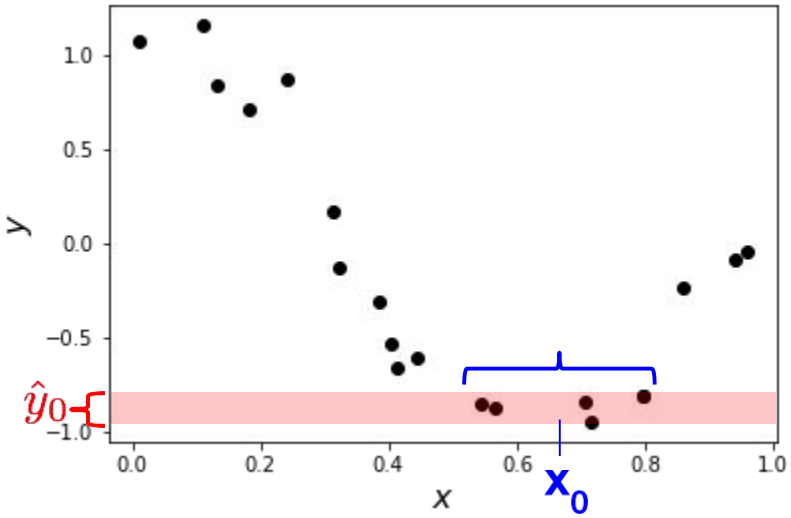
Opciones

- Solución 1: árboles de decisión en regresión
- Solución 2: Knn regresión



KNN regresión

- Queremos estimar el precio (y) a partir de sqm(x) en un nuevo punto x_0



- Buscamos vecinos de x_0

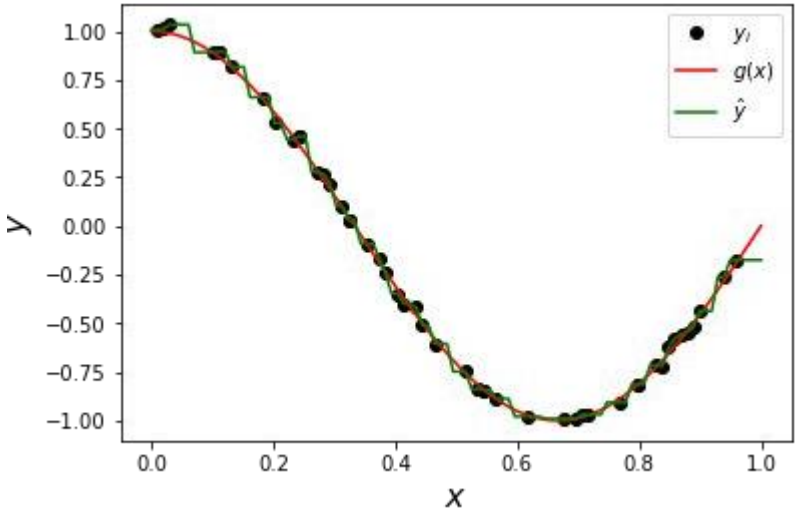
$$d_i = ||x_0 - x_i||_2$$

- El valor estimado es la media de los vecinos

$$\hat{y}_0 = \frac{1}{K} \sum_{i=1}^{K} y_i$$

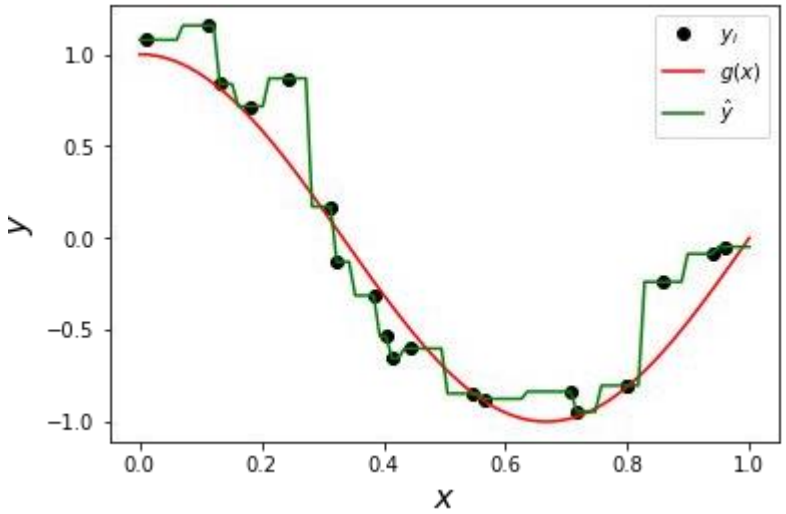
K = 1

- Buen ajuste si hay mucha densidad de puntos y bajo ruido



K = 1

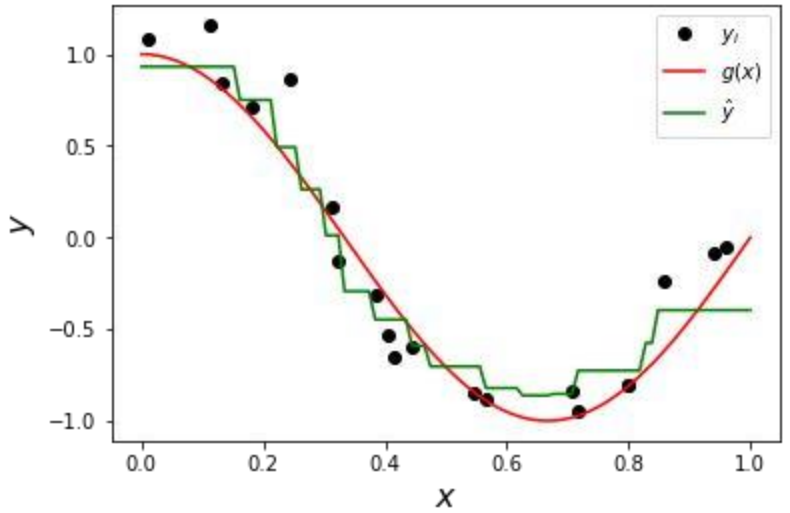
- Región sin ejemplos, mal ajustada
- Sensible al ruido



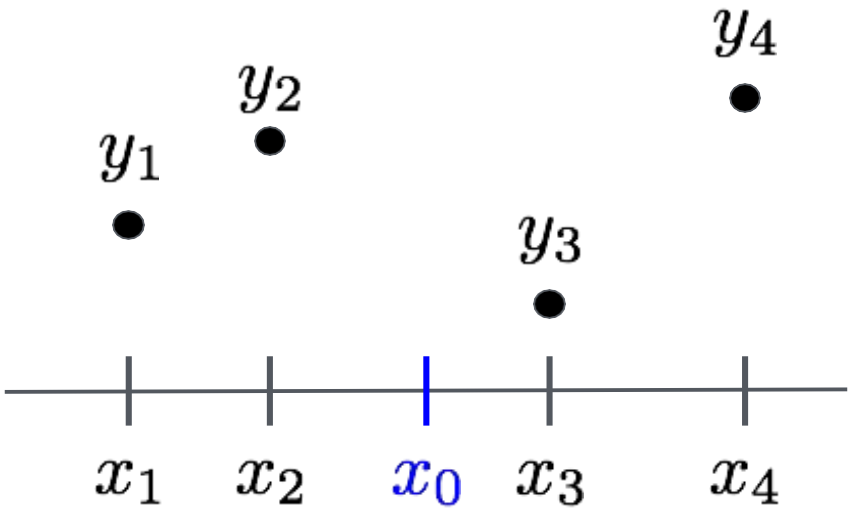
K = 5

- Como ya sabemos, deberíamos aumentar K
- Lo cual produce problemas de borde

- Solución: ponderar estimación por distancia entre vecinos
  - Dar más peso a los más cercanos
  - Reducir el peso de los más alejados



Ponderar la estimación (K = 4)



$$y_0 = \frac{\theta_1 y_1 + \theta_2 y_2 + \theta_3 y_3 + \theta_4 y_4}{\sum_{i=1}^{K} \theta_i}$$

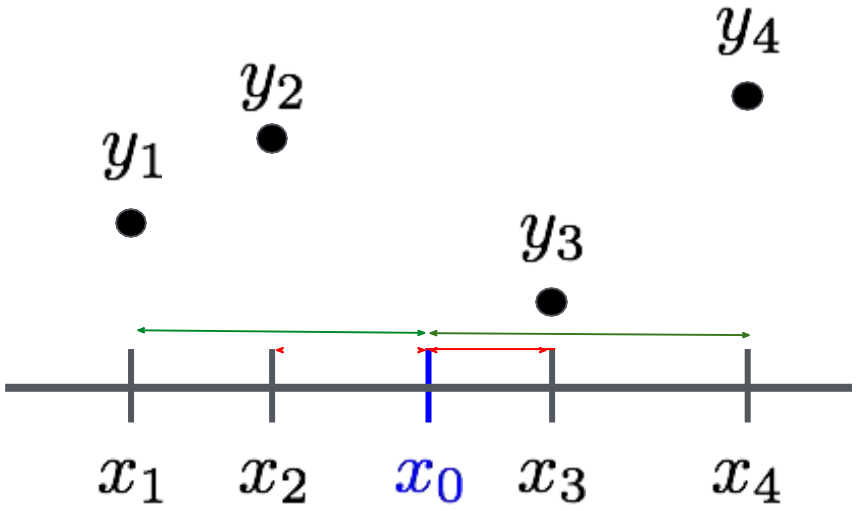
¿Cómo elegimos θ_i?

- En función de la distancia:

$$\theta_i = \frac{1}{d_i}$$

- donde:

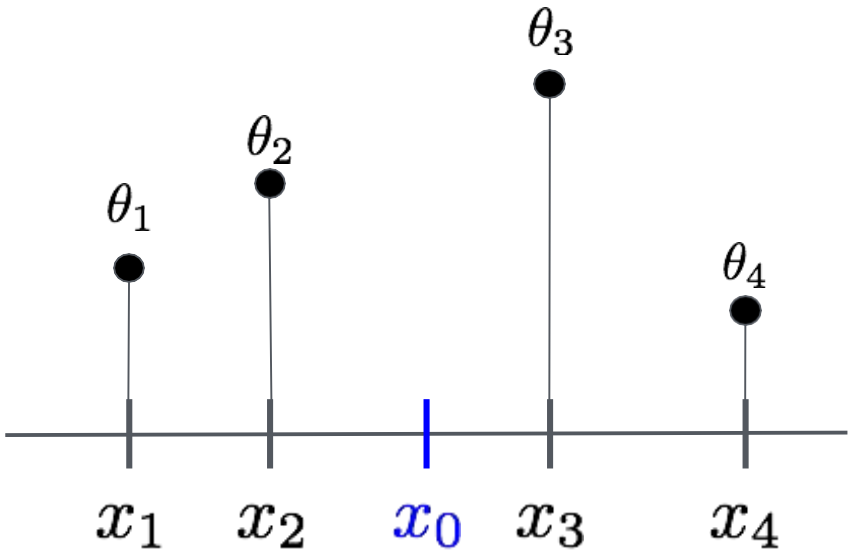
$$d_i = ||x_0 - x_i||_2$$



$$y_0 = \frac{\theta_1 y_1 + \theta_2 y_2 + \theta_3 y_3 + \theta_4 y_4}{\sum_{i=1}^{K} \theta_i}$$

¿Cómo elegimos θ_i?

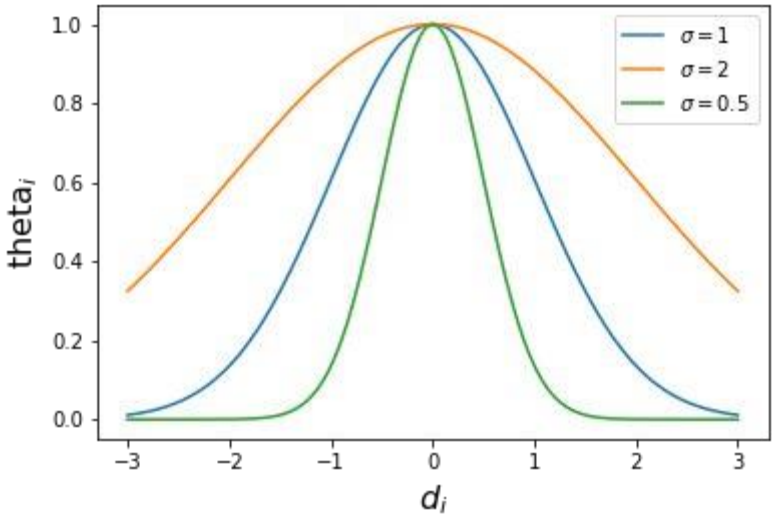
- Si $d_i \downarrow \Rightarrow \theta_i \uparrow$
- Si $d_i \uparrow \Rightarrow \theta_i \downarrow$



Otras opciones

- Podemos utilizar otras medidas de similitud

$$\theta_i(\sigma) = e^{-\frac{||\mathbf{x}_0 - \mathbf{x}_i||_2^2}{2\sigma^2}}$$



Kernel RBF

- RBF: Radial Basis Function

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{||\mathbf{x}_i - \mathbf{x}_j||_2^2}{2\sigma^2}}$$

- Expresada como

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma||\mathbf{x}_i - \mathbf{x}_j||_2^2}$$

Otros kernels

- Lineal

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) =< \mathbf{x}_i, \mathbf{x}_j >= \mathbf{x}_i^T \cdot \mathbf{x}_j$$

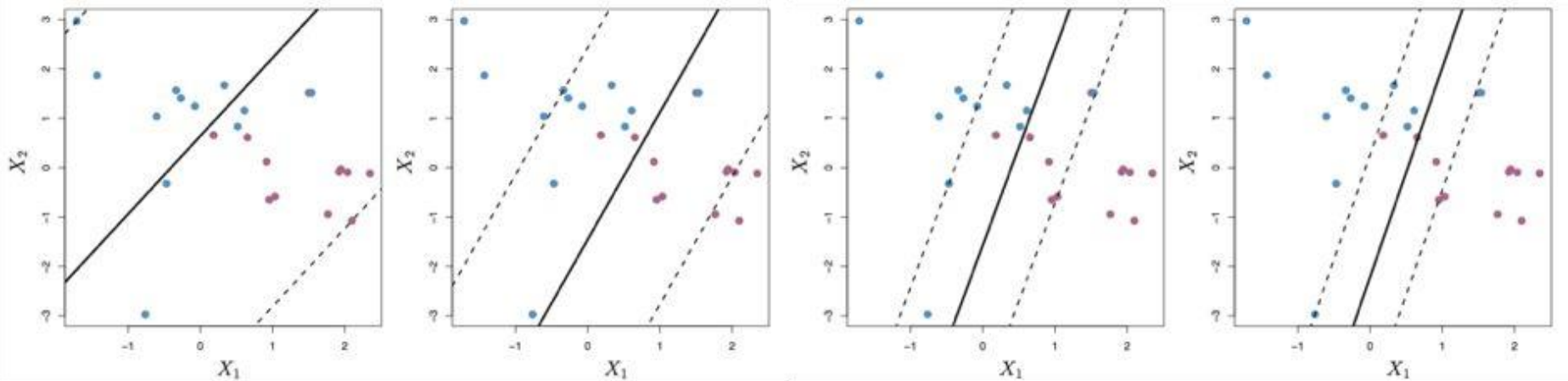
- Polinómico

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = (\gamma < \mathbf{x}_i, \mathbf{x}_j > +r)^d$$

Índice

1. Intuición: el (hiper)plano separador
2. ¿Por qué *Support Vector*?
3. Caso no linealmente separable
4. SVMs en regresión
5. SVMs y selección de características
6. K-nn en regresión
7. **Kernels y SVM**
8. Otros algoritmos con Kernels

SVMs (recap)

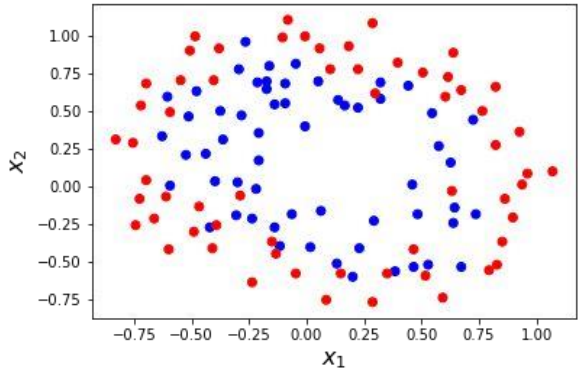


- Problemas lineales $\hat{\boldsymbol{\omega}} = \sum_{i=1}^{N} \alpha_i y^{(i)} \mathbf{x}^{(i)}$

$$\hat{y} = f(\mathbf{x}) = \text{sign}\left(\hat{\boldsymbol{\omega}}^T \mathbf{x} + \hat{\omega}_0\right) = \text{sign}\left(\sum_{i=1}^{N} \alpha_i y^{(i)} \mathbf{x}^T \mathbf{x}^{(i)} + \hat{\omega}_0\right)$$

SVMs: fronteras no lineales

- La formulación de las SVMs y LR es similar
- Si queremos definir fronteras de separación no lineal en LR, ¿qué habría que hacer?



$$x_1, x_2 \longrightarrow x_1, x_2, x_1^2, x_2^2, x_1 x_2$$

$D = 2$ $\qquad$ $D = 5$

Aumentamos la dimensionalidad del espacio de características

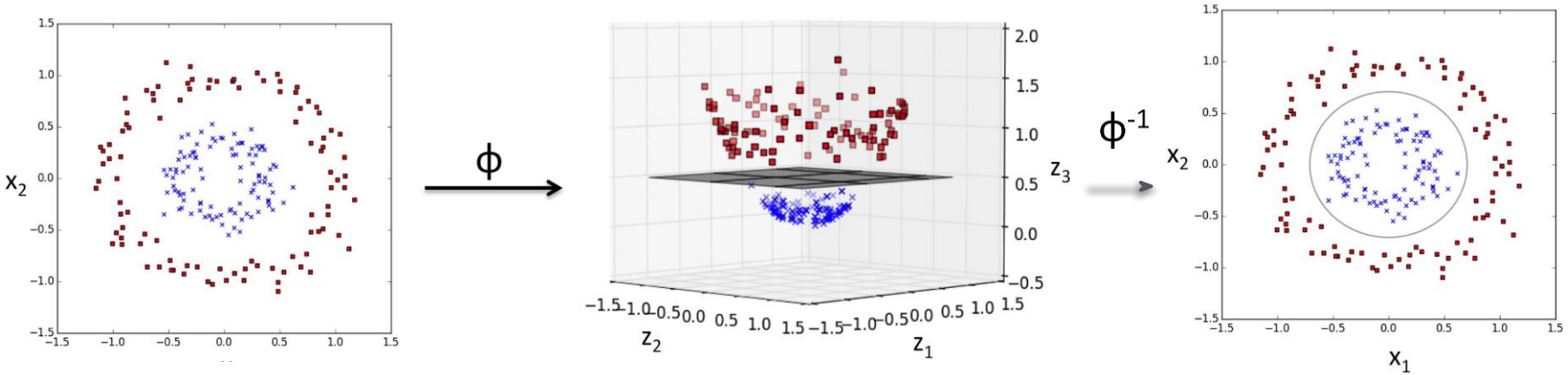
SVMs: fronteras no lineales

- En LR tenemos que elegir la función de transformación bajo nuestro mejor criterio

$$\mathbf{x}_1 \Rightarrow \phi(\mathbf{x}_1) \Rightarrow \mathbf{x}_1^2$$
$$\mathbf{x}_2 \Rightarrow \phi(\mathbf{x}_2) \Rightarrow \mathbf{x}_2^2$$

SVMs: fronteras no lineales

- ¿Qué buscamos con esta transformación?



SVMs: fronteras no lineales

- La formulación SVM permite no tener que conocer la transformación $\phi(x)$

- Truco del núcleo

Si tengo $< x^i, x^j >$ y quiero aplicar $< \phi(x^i), \phi(x^j) >$ no necesito conocer $\phi(x)$, aplico un Kernel.

Allí donde tenga $< x^i, x^j >$ utilizaré un Kernel como

$$\mathcal{K}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = e^{-\gamma||\mathbf{x}^{(i)} - \mathbf{x}^{(j)}||_2^2}$$

Formulación

- Así podemos transformar la frontera lineal

$$\hat{y} = f(\mathbf{x}) = \text{sign}\left(\hat{\boldsymbol{\omega}}^T \mathbf{x} + \hat{\omega}_0\right) = \text{sign}\left(\sum_{i=1}^{N} \alpha_i y^{(i)} \mathbf{x}^T \mathbf{x}^{(i)} + \hat{\omega}_0\right)$$

- En una no lineal

$$\hat{y} = f(\mathbf{x}) = \text{sign}\left(\hat{\boldsymbol{\omega}}^T \mathbf{x} + \hat{\omega}_0\right) = \text{sign}\left(\sum_{i=1}^{N} \alpha_i y^{(i)} \mathcal{K}(\mathbf{x}, \mathbf{x}^{(i)}) + \hat{\omega}_0\right)$$

Ejemplo

- Supongamos la transformación

$$\phi(\mathbf{x}^{(i)}) = \left[\left(x_1^{(i)}\right)^2, \left(x_2^{(i)}\right)^2, \sqrt{2}x_1^{(i)}x_2^{(i)}\right]$$

- Con $\mathbf{x}^{(i)} = \left[x_1^{(i)}, x_2^{(i)}\right]$ vamos a demostrar que

$$< \phi(\mathbf{x}^{(i)}), \phi(\mathbf{x}^{(j)}) >= \mathcal{K}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \left(< \mathbf{x}^{(i)}, \mathbf{x}^{(j)} >\right)^2$$

Ejemplo

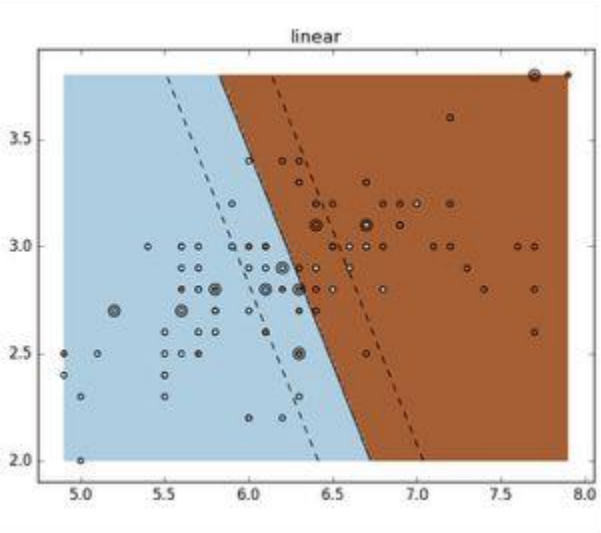
$$\mathbf{x}^{(i)} = \left[x_1^{(i)}, x_2^{(i)}\right]$$

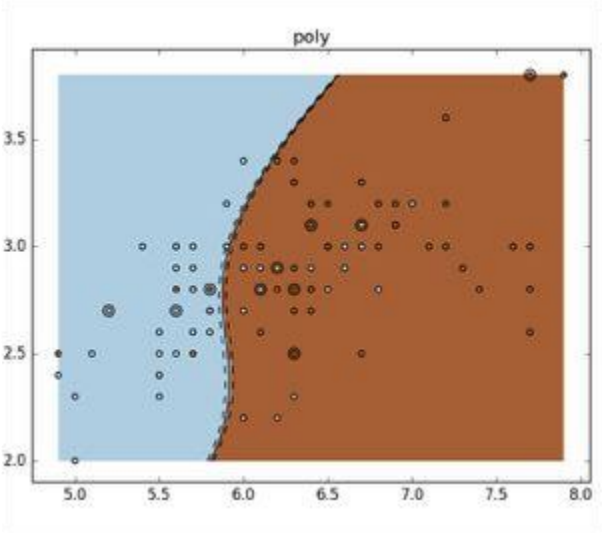
$$\phi(\mathbf{x}^{(i)}) = \left[\left(x_1^{(i)}\right)^2, \left(x_2^{(i)}\right)^2, \sqrt{2}x_1^{(i)}x_2^{(i)}\right]$$

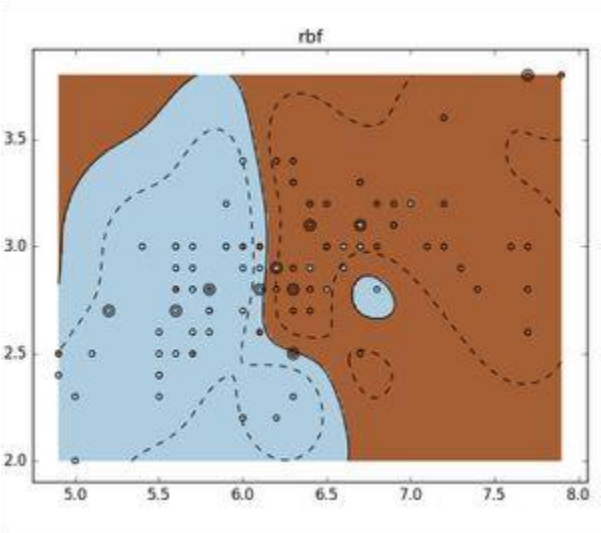
$$\phi(\mathbf{x}^{(j)}) = \left[\left(x_1^{(j)}\right)^2, \left(x_2^{(j)}\right)^2, \sqrt{2}x_1^{(j)}x_2^{(j)}\right]$$

$$\begin{aligned}
< \phi(\mathbf{x}^{(i)}), \phi(\mathbf{x}^{(j)}) > &= \left(x_1^{(i)}\right)^2 \left(x_1^{(j)}\right)^2 + \left(x_2^{(i)}\right)^2 \left(x_2^{(j)}\right)^2 + 2x_1^{(i)}x_2^{(i)}x_1^{(j)}x_2^{(j)} \\
&= \left(x_1^{(i)}x_1^{(j)}\right)^2 + \left(x_2^{(i)}x_2^{(j)}\right)^2 + 2x_1^{(i)}x_2^{(i)}x_1^{(j)}x_2^{(j)} \\
&= \left(x_1^{(i)}x_1^{(j)} + x_2^{(i)}x_2^{(j)}\right)^2 = \left(< \mathbf{x}^{(i)}, \mathbf{x}^{(j)} >\right)^2 = \mathcal{K}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})
\end{aligned}$$

Resultado







- Hay que fijar los parámetros libres del Kernel
  - Hiperparámetro adicional
  - CV

Índice

1. Intuición: el (hiper)plano separador
2. ¿Por qué *Support Vector*?
3. Caso no linealmente separable
4. SVMs en regresión
5. SVMs y selección de características
6. K-nn en regresión
7. Kernels y SVM
8. **Otros algoritmos con Kernels**

Métodos Kernel

- Allí donde tengamos un producto escalar de la forma $< x^i, x^j >$ o en forma matricial una expresión como XX^T podemos aplicar un Kernel

  - Ridge Regression, muy similar a SVR
  - PCA

Kernel Ridge Regression

- La solución en forma matricial del algoritmo Ridge es

$$\hat{\boldsymbol{\omega}} = \left(\mathbf{X}^T\mathbf{X} + \alpha\mathbf{I}_D\right)^{-1}\mathbf{X}^T\mathbf{y}$$

- Aplicando el matrix inversión lemma, puede expresarse como:

$$\hat{\boldsymbol{\omega}} = \mathbf{X}^T\left(\mathbf{X}\mathbf{X}^T + \alpha\mathbf{I}_N\right)^{-1}\mathbf{y}$$

- Esto permite aplicar la extensión Kernel

$$\hat{\boldsymbol{\omega}} = \mathbf{X}^T\left(\mathcal{K} + \alpha\mathbf{I}_N\right)^{-1}\mathbf{y}$$

Kernel Ridge Regression

- Misma solución que SVR

$$\hat{\omega} = \mathbf{X}^T \boldsymbol{\beta}$$

Donde $\boldsymbol{\beta} = (\mathcal{K} + \alpha \mathbf{I}_N)^{-1} \mathbf{y}$

Diferencia: la función de coste difiere, siendo en este caso el MSE.

Conclusiones sobre SVMs y Kernels

- Algoritmos muy potentes con grandes prestaciones
- El algoritmo no calcula la probabilidad, se estima a partir de heurística (no muy fiable)
- Computacionalmente intenso:
  - Si alta dimensionalidad (muchas variables), Kernel lineal
  - Valores de C elevados → cuesta mucho entrenar
  - Cálculo del Kernel cuando el problema tiene muchas muestras
- RBF es capaz de aprender casi todo, Kernel universal.
- Kernels usan medidas de distancia/similitud: ESCALADO

Referencias

- Machine Learning, a probabilistic perspective
  - Capítulo 14
- Hands On Machine Learning.
  - Capítulos 5, 8
- Felipe Alonso-Atienza, Tesis Doctoral (uc3m)

Let’s code!