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*

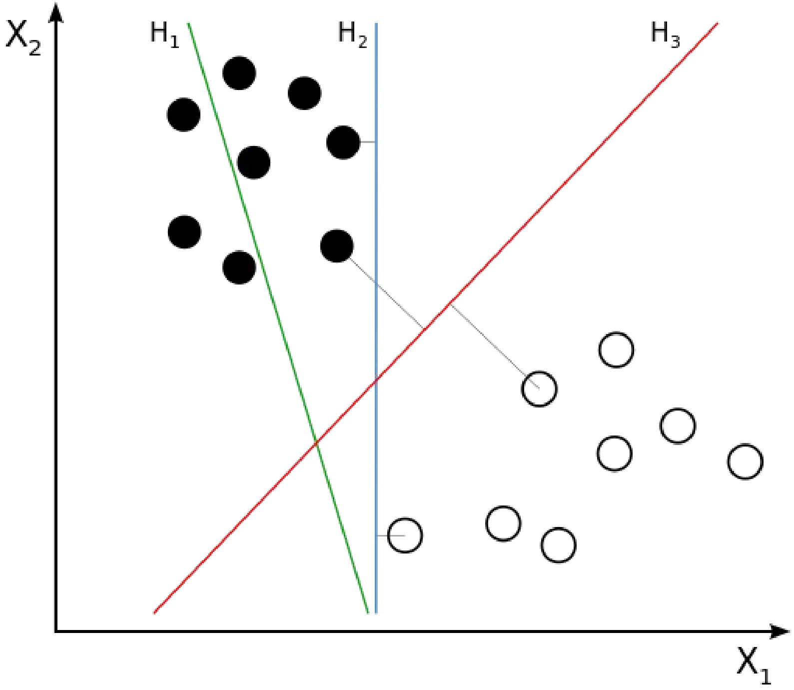
- Se definen como **clasificadores lineales de máximo margen**

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

Índice

Intuición

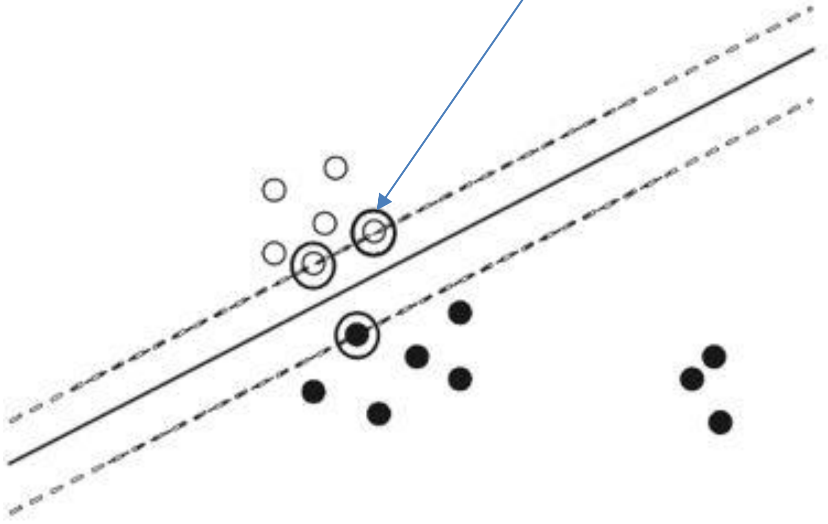
- Clasificador lineal definido por un hiperplano separador de máximo margen



Support Vectors

La solución viene dada por las muestras en el margen (de ahí “support vectors”)

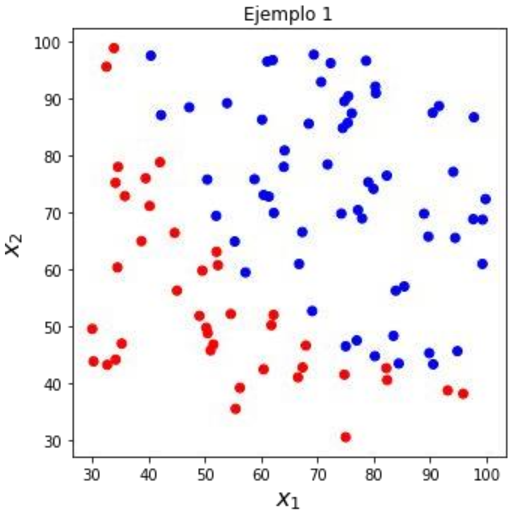
El plano “se apoya” en estas muestras



Índice

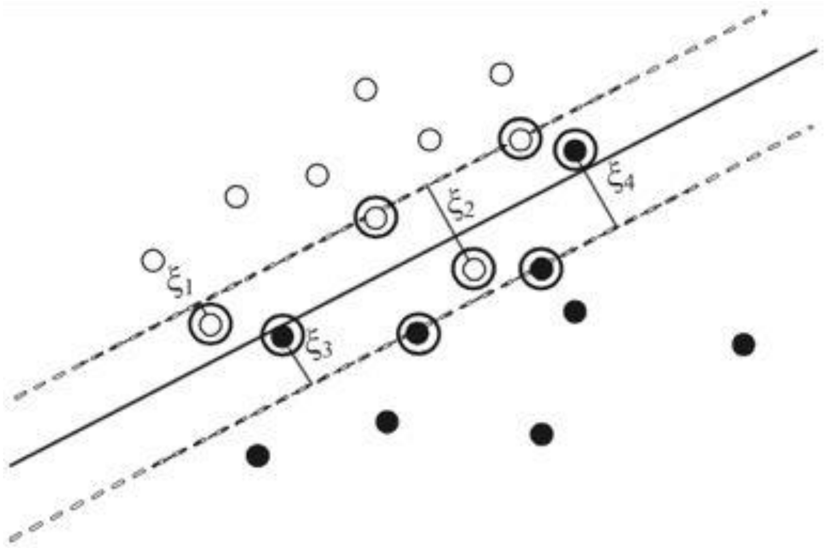
Caso linealmente no separable

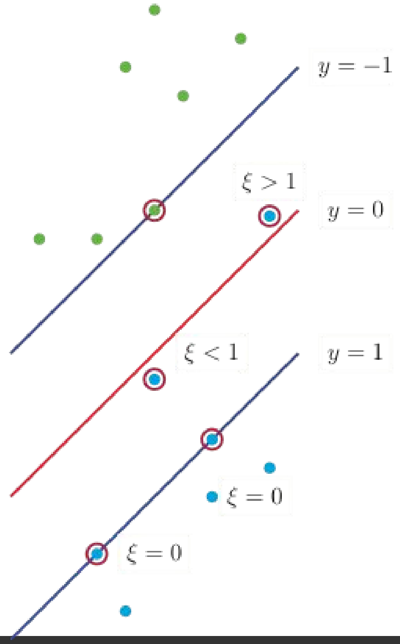
- Hasta ahora hemos trabajado con un caso en el que las clases son claramente separables, esto es, no hay solapamiento entre ellas
- No hablamos de fronteras no lineales, seguimos considerando que existe un <u>hiperplano</u> capaz de separar las clases, aunque con <u>errores</u>



Caso linealmente no separable

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





Caso linealmente no separable

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

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

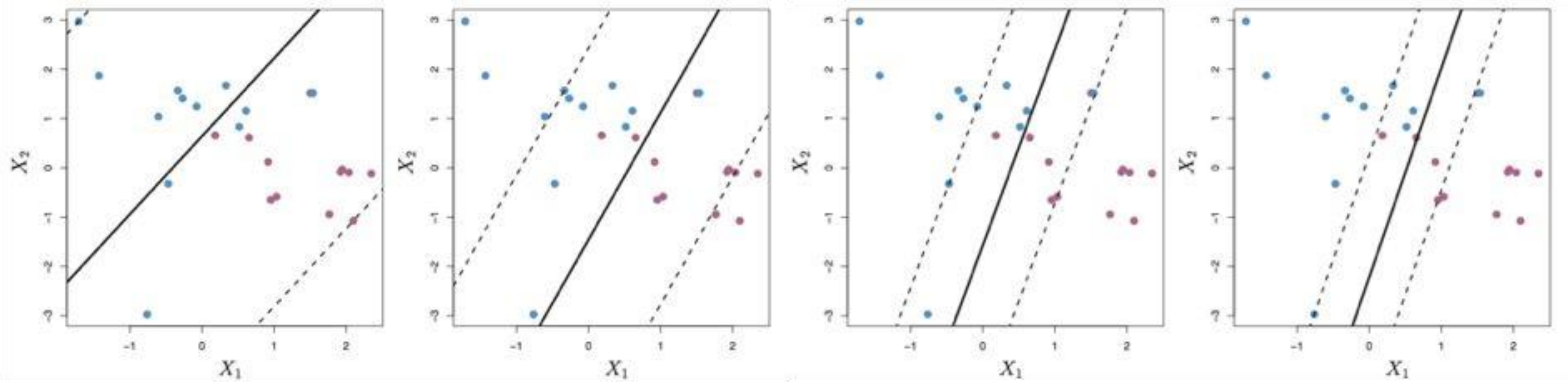
$$\text{s.to } y^{(i)}\left(\boldsymbol{\omega}^T\mathbf{x}^{(i)} + \omega_0\right) \geq 1 - \xi_i, i = 1, \ldots, N$$

$$\xi_i \geq 0.$$

- … regularización

Parámetro de regularización C

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

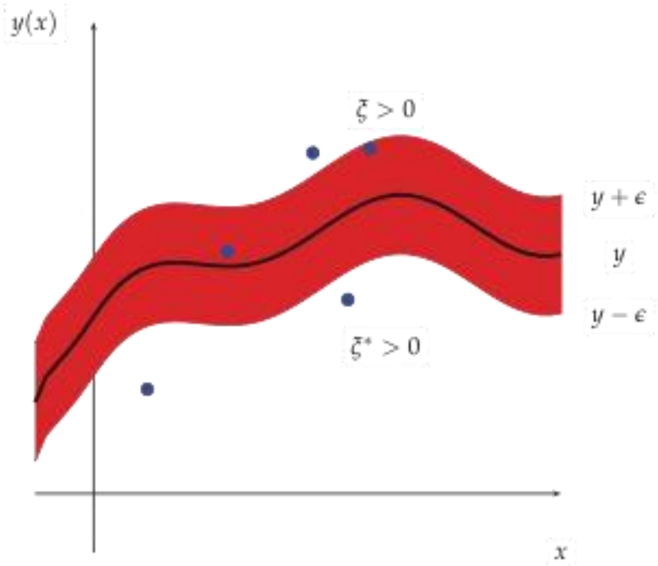


- Si C elevado, margen estrecho, más peso a los errores. Alta complejidad
- Si C pequeño, margen ancho, menos peso a los errores. Baja complejidad

Índice

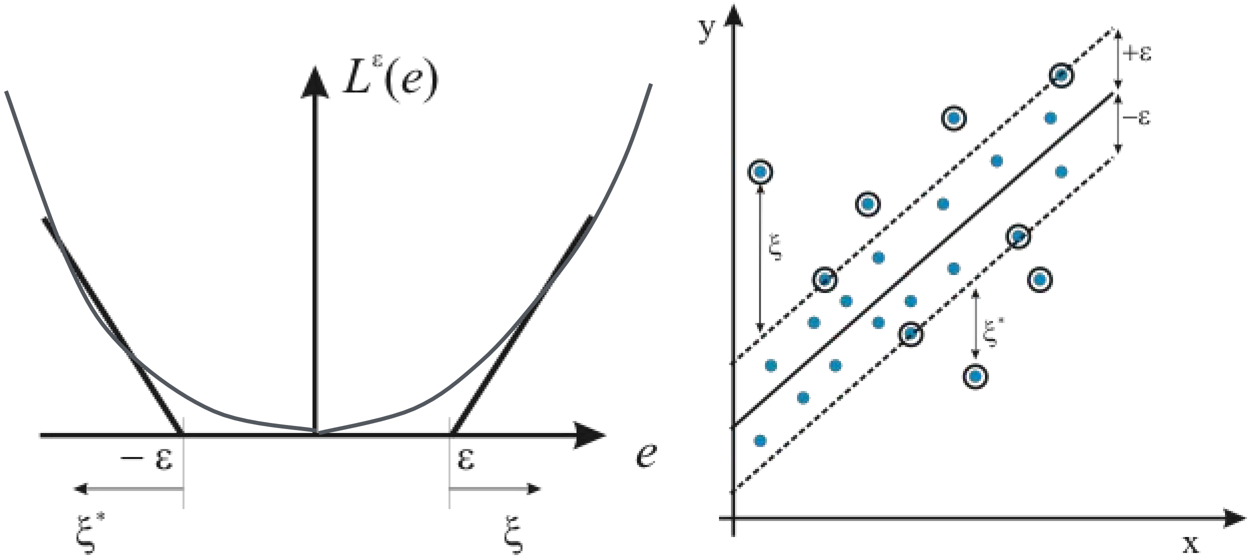
SVR: intuición

- 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

Pero ¿qué hago con las muestras que caen fuera del margen? Las penalizo



Índice

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

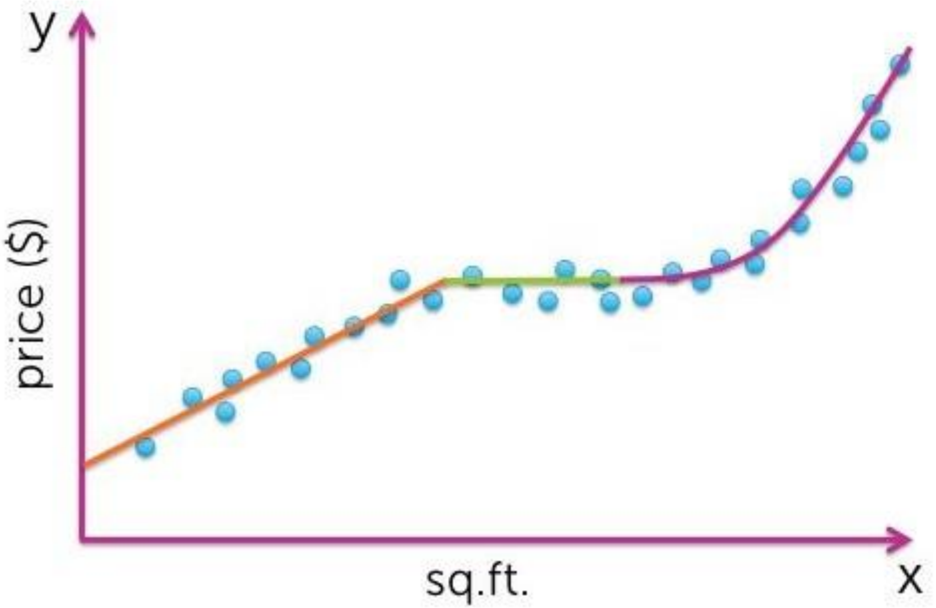
Let’s code!

Índice

1. Intuición: el hiperplano separador
2. Caso no linealmente separable
3. SVMs en regresión
4. SVMs y selección de características
5. **K-nn en regresión**
6. Kernels y SVM
7. Otros algoritmos con Kernels

Motivación

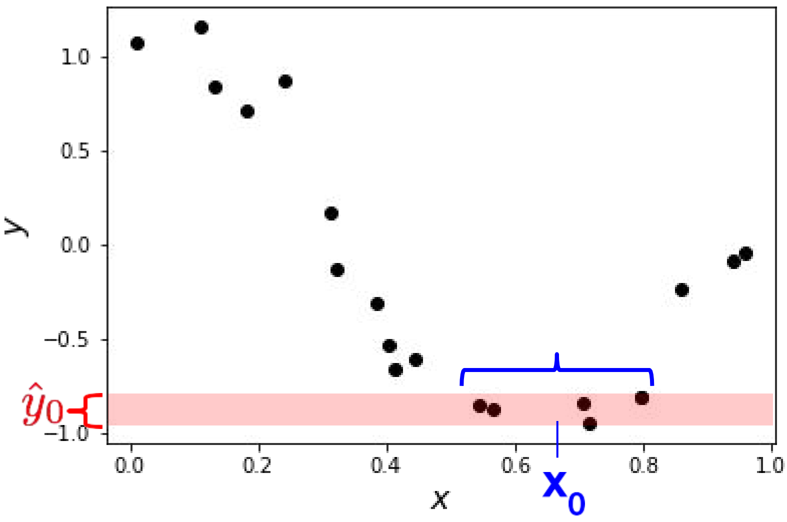
- Modelos paramétrico no siempre resultan adecuados
  - Modelo basado en datos, no paramétrico



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

Ejemplo: KNN regresión

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



- Buscamos vecinos de x_0

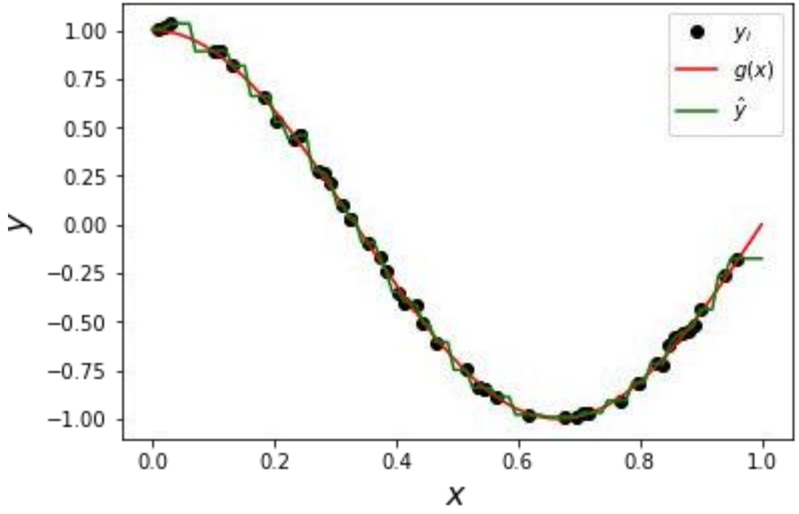
$$d_i = ||x_0 - x_i||_2$$

- El valor estimado es la media de los vecinos

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

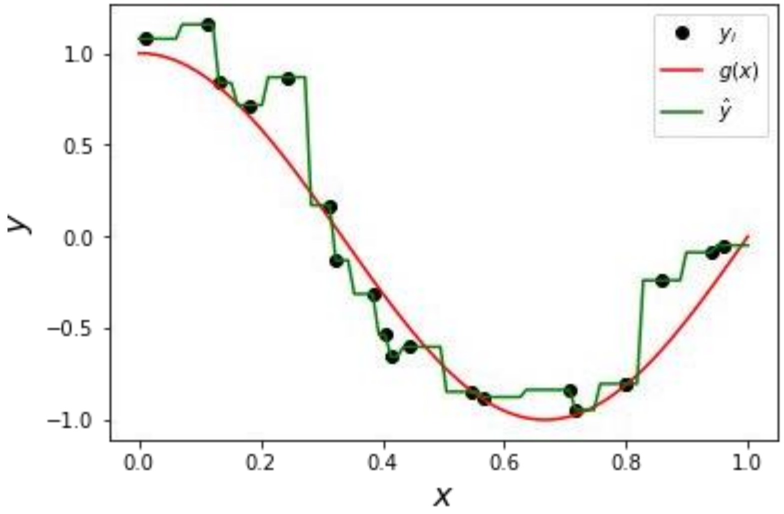
K = 1

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



K = 1

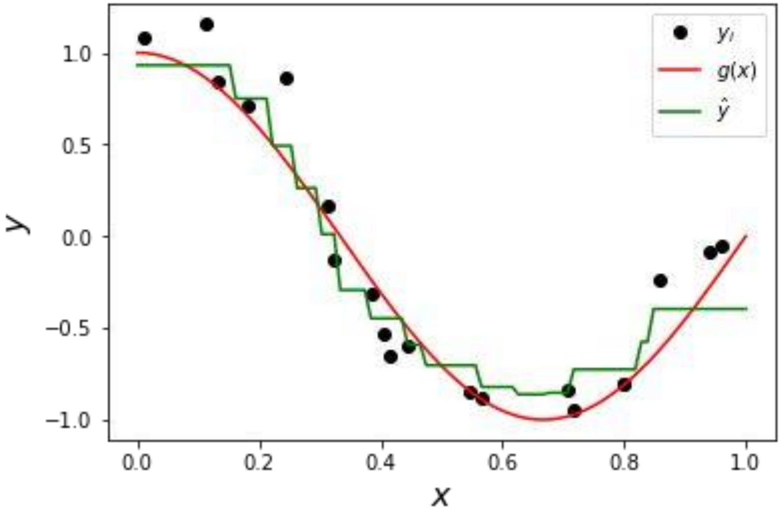
- Región sin ejemplos, mal ajustada
- Sensible al ruido



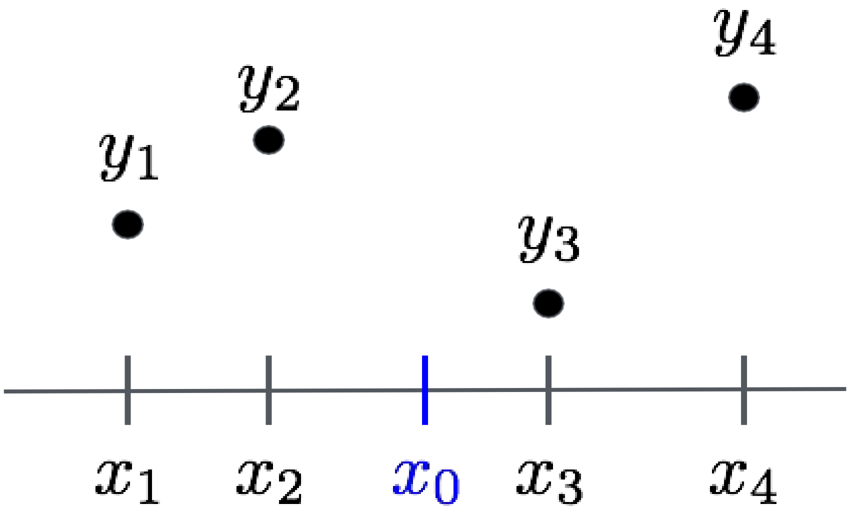
K = 5

- 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)



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

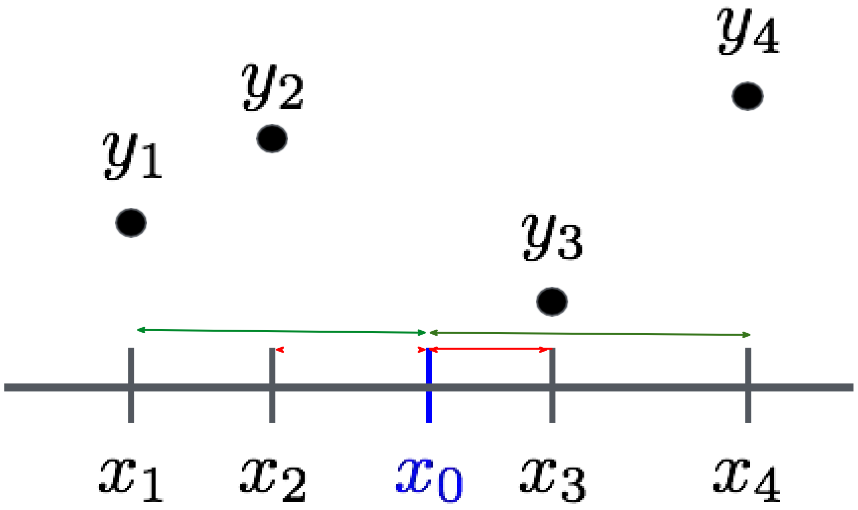
¿Cómo elegimos θ_i?

- En función de la distancia:

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

- donde:

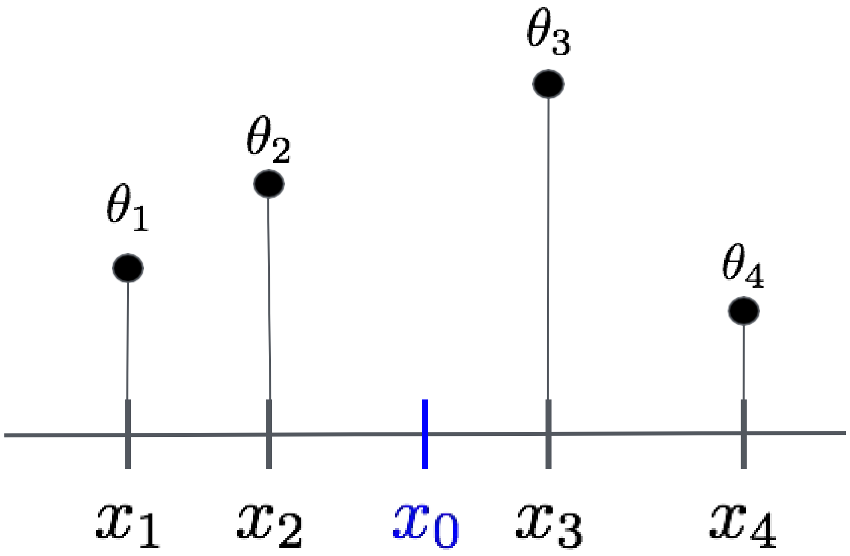
$$d_i = ||x_0 - x_i||_2$$



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

¿Cómo elegimos θ_i?

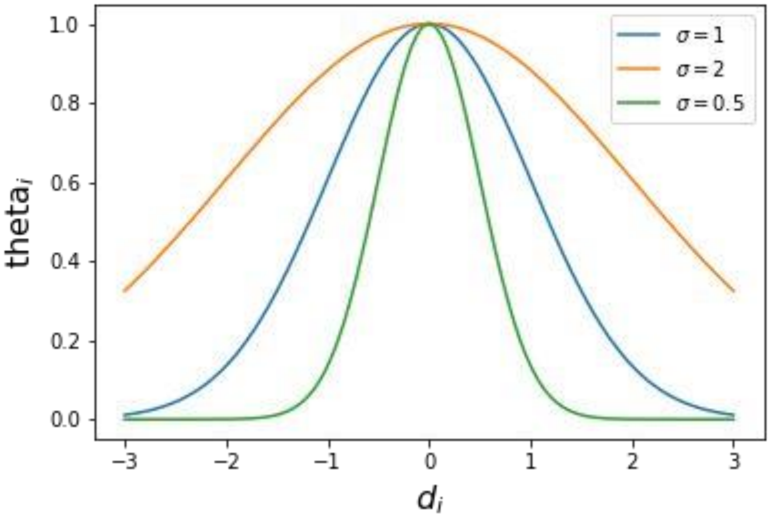
- Si $d_i \downarrow \Rightarrow \theta_i \uparrow$
- Si $d_i \uparrow \Rightarrow \theta_i \downarrow$



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

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

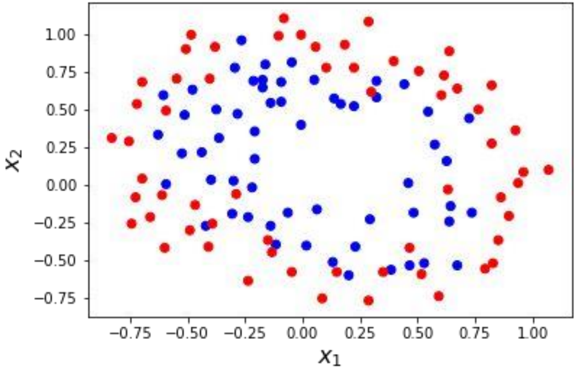
- Polinómico

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

Índice

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?



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

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

Aumentamos la dimensionalidad del espacio de características

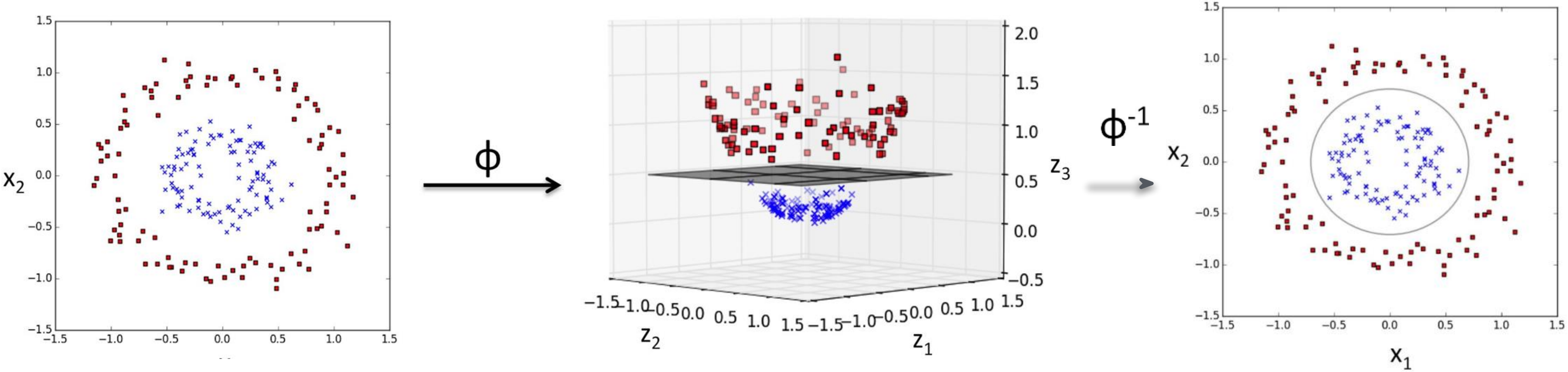
SVMs: fronteras no lineales

- 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?

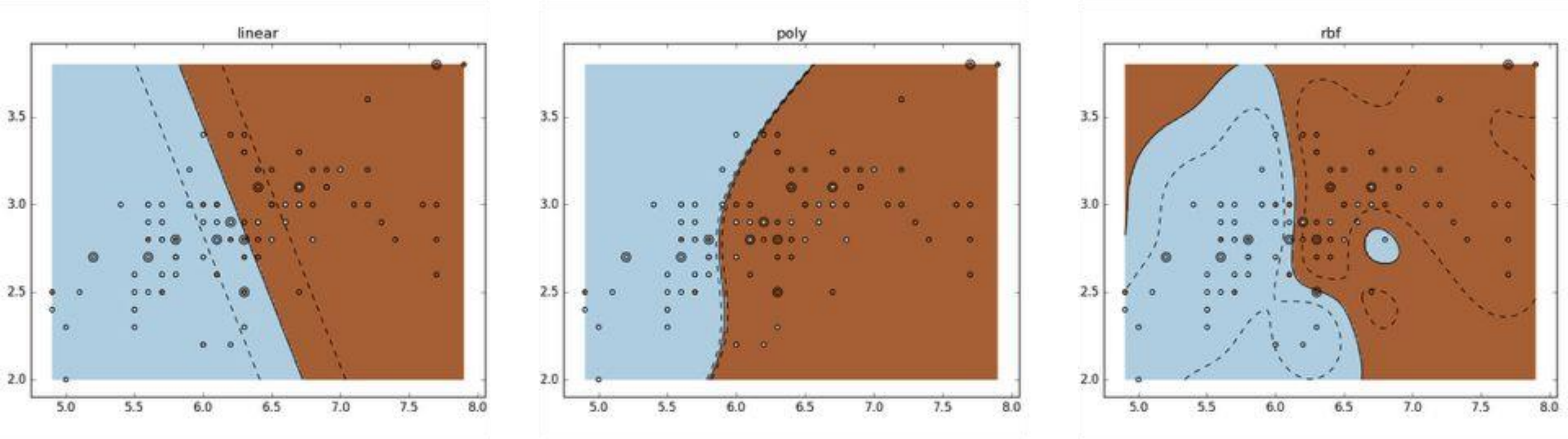


SVMs: fronteras no lineales

- La formulación SVM permite no tener que conocer la transformación.

- Truco del kernel (Kernel trick): No es necesario calcular explícitamente las coordenadas de los puntos en el nuevo espacio dimensional; basta con calcular las distancias entre parejas de puntos, aunque no sepamos las coordenadas. Ese cálculo puede realizarse de forma sencilla con un kernel.

Resultado



- Hay que fijar los parámetros libres del Kernel
  - Hiperparámetro adicional: degree para poly, gamma para RBF
  - Cross Validation

Índice

1. Intuición: el hiperplano separador
2. Caso no linealmente separable
3. SVMs en regresión
4. SVMs y selección de características
5. K-nn en regresión
6. Kernels y SVM
7. **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

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

Let’s code!