Laboratorio de datos, clase 14

Máquinas de soporte vectorial
(*support vector machines*, SVM)

Prof. Enzo Tagliazucchi

tagliazucchi.enzo@googlemail.com
www.cocucolab.org

El panorama de algoritmos de ML

- Los árboles de decisión son rápidos e interpretables
- Qué bien
- Pero pueden quedar en un mínimo de la función de pérdida
- Qué mal
- Los árboles pueden ser combinados en ensembles para atenuar este problema
- Qué bien
- Pero los ensembles dejan de ser tan interpretables como los árboles individuales

¿Qué problemas puede tener un algoritmo de ML?

Frontera de separación lineal
(clasificador lineal, regresión logística sin aumentación de features)

Converge a un mínimo local de la función de costo
(árboles de decisión, redes neuronales multicapa)

Tarda mucho en evaluar nuevas instancias
(KNN)

Resultados poco interpretables o intuitivos
(Random forest)

¿No sería lindo tener un clasificador que sea...?

... capaz de aprender una frontera no lineal prácticamente arbitraria?

... este garantizado a converger siempre a un mínimo global de la función de costo?

... sea rápido de entrenar y evaluar en nuevos datos?

... tenga una interpretación geométrica sencilla e intuitiva?

Machine Learning, 20, 273–297 (1995)

Support-Vector Networks

CORINNA CORTES corinna@neural.att.com
VLADIMIR VAPNIK vlad@neural.att.com
AT&T Bell Labs., Holmdel, NJ 07733, USA

Editor: Lorenza Saitta

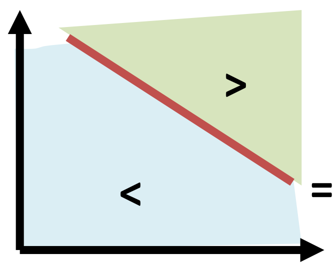
Abstract. The *support-vector network* is a new learning machine for two-group classification problems. The machine conceptually implements the following idea: input vectors are non-linearly mapped to a very high-dimension feature space. In this feature space a linear decision surface is constructed. Special properties of the decision surface ensures high generalization ability of the learning machine. The idea behind the support-vector network was previously implemented for the restricted case where the training data can be separated without errors. We here extend this result to non-separable training data.

High generalization ability of support-vector networks utilizing polynomial input transformations is demonstrated. We also compare the performance of the support-vector network to various classical learning algorithms that all took part in a benchmark study of Optical Character Recognition.

Clasificador de margen máximo

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$$

Hiperplano en R^2



Idea: como un hiperplano divide al espacio en dos mitades, puedo usarlo para clasificar puntos en dos categorías, dependiendo de qué lado del hiperplano queden.

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$

Hiperplano en R^p

Clasificador de margen máximo

Matriz de datos **X** (n samples, p features)

$$x_1 = \begin{pmatrix} x_{11} \\ \vdots \\ x_{1p} \end{pmatrix}, \ldots, x_n = \begin{pmatrix} x_{n1} \\ \vdots \\ x_{np} \end{pmatrix}$$

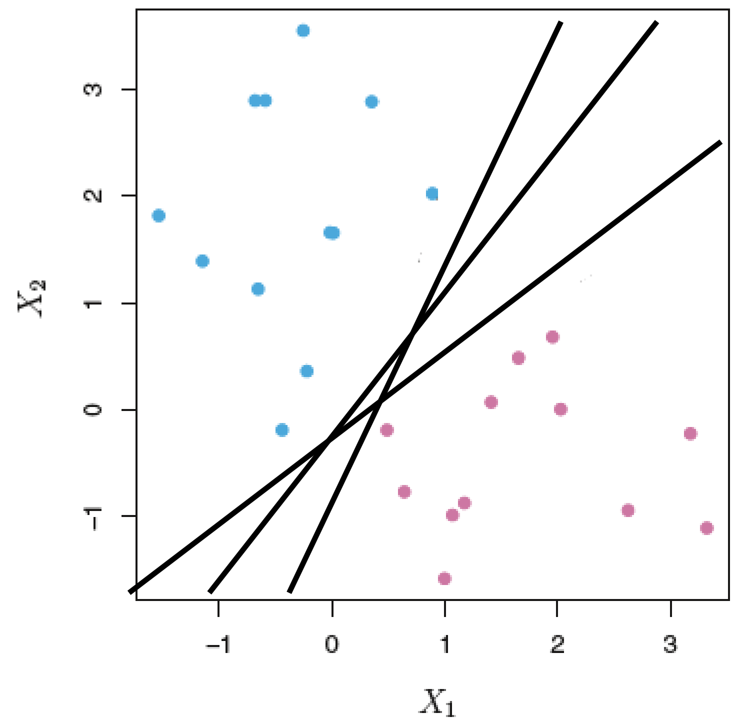
Cada uno de ellos con dos etiquetas posibles, $y_i = \pm 1$

Dado un nuevo ejemplo:

$$x^* = \begin{pmatrix} x_1^* & \ldots & x_p^* \end{pmatrix}^T$$

¿Qué etiqueta le corresponde?

¿Cuál elegir?

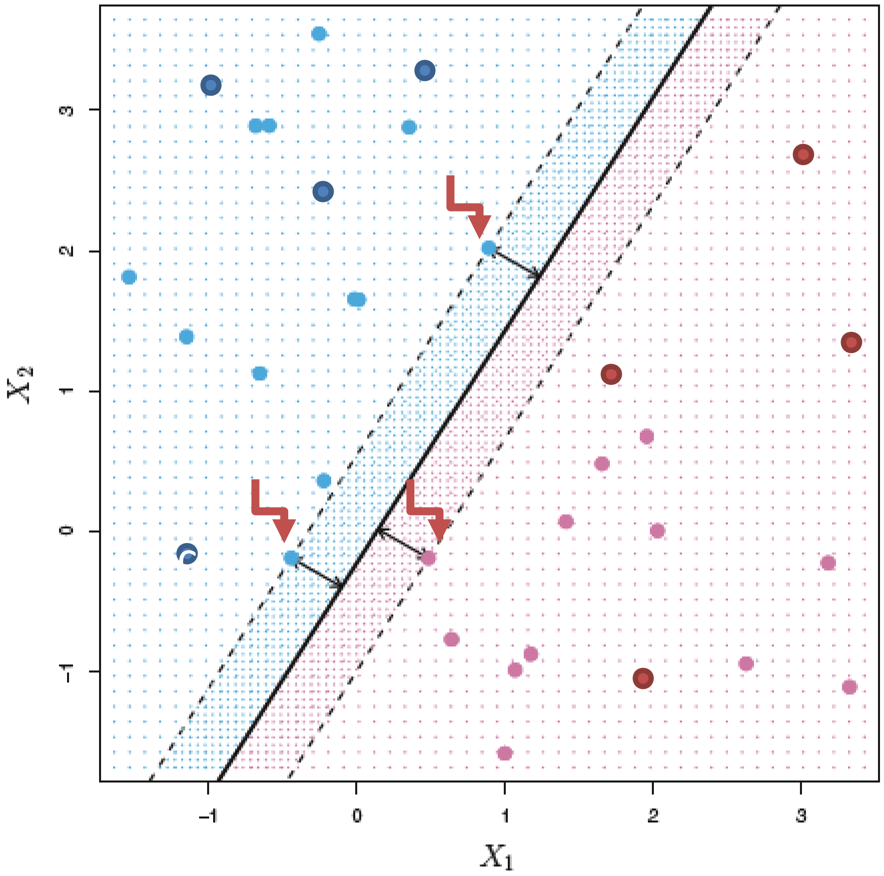


El signo de:

$$f(x^*) = \beta_0 + \beta_1 x_1^* + \beta_2 x_2^* + \ldots + \beta_p x_p^*$$

nos da la respuesta

Clasificador de margen máximo



Para cada recta puedo calcular su distancia a los puntos de entrenamiento. Esta distancia se denomina ***margen***

Dentro de las rectas hay una especial: ***La que maximiza el margen***

Los ejemplos para los cuales se realiza esta distancia mínima con el hiperplano se llaman ***soporte***

La ecuación que define al hiperplano óptimo depende únicamente de su suporte; los otros ejemplos pueden cambiarse sin que cambie el hiperplano

Clasificador de margen máximo

Observación 1 (álgebra lineal del CBC)

Sea un hiperplano dado por $\beta_0 + \langle \boldsymbol{w}, \boldsymbol{x} \rangle = 0$ y con $\boldsymbol{w}$ su normal y $\boldsymbol{x}$ un punto del espacio. Entonces, la distancia de un punto cualquiera del espacio $\boldsymbol{x}_0$ al plano está dada por:

$$d = \frac{|\beta_0 + \langle \boldsymbol{w}, \boldsymbol{x_0} \rangle|}{\|\boldsymbol{w}\|}$$

Observación 2 (corolario de lo anterior)

Si la normal cumple $\|\boldsymbol{w}\| = 1$, entonces la distancia de un punto al hiperplano está dada por, $d = |\beta_0 + \langle \boldsymbol{w}, \boldsymbol{x_0} \rangle|$

Observación 3

Dado un ejemplo $\boldsymbol{x}_i$ con etiqueta $y_i = \pm 1$ dependiendo de si $\beta_0 + \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle < 0$ o $\beta_0 + \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle > 0$, entonces el producto $y_i(\beta_0 + \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle)$ es siempre > 0, y si $\|\boldsymbol{w}\| = 1$, también es la distancia del punto $\boldsymbol{x_i}$ al hiperplano.

Clasificador de margen máximo

El problema

Encontrar β_0 y $w = (\beta_1 \ldots \beta_p)$ tal que el M compatible con estas dos condiciones es el más grande posible:

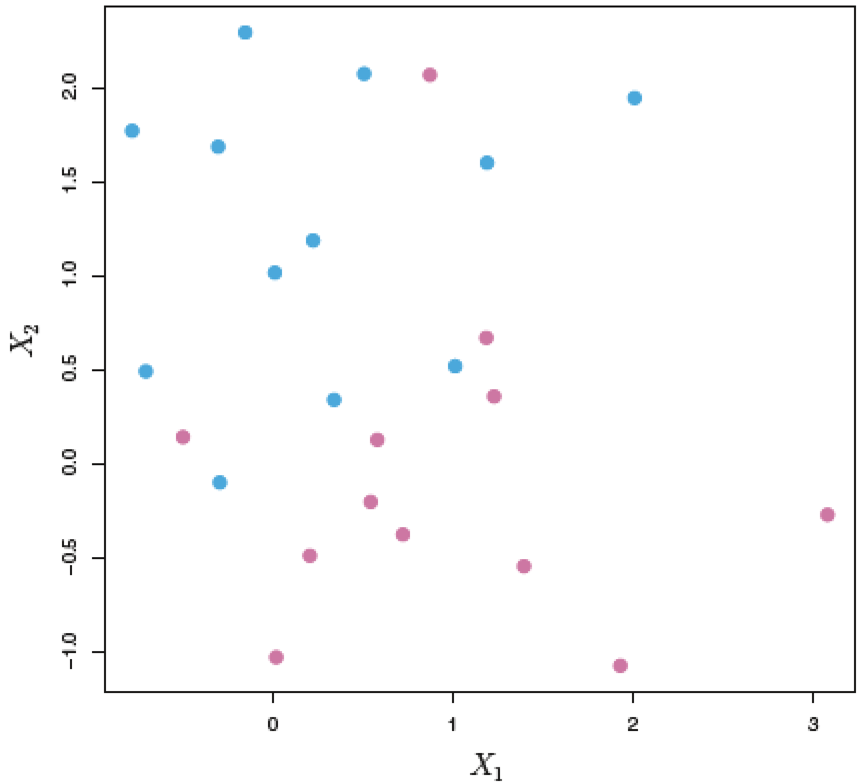
$$\sum_{j=1}^{p} \beta_j^2 = 1, \quad \rightarrow \quad \|w\| = 1$$

$$\underline{y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip})} \geq M \quad \forall\, i = 1, \ldots, n$$

por lo anterior, es igual a la distancia entre cada punto y el hiperplano

Formulación matemática de pedir que el margen (M) sea lo más grande posible

Clasificador de soporte vectorial



Clasificador de soporte vectorial

Encontrar γ, $\mathbf{w}$ tal que el M compatible con estas dos condiciones es el más grande posible:

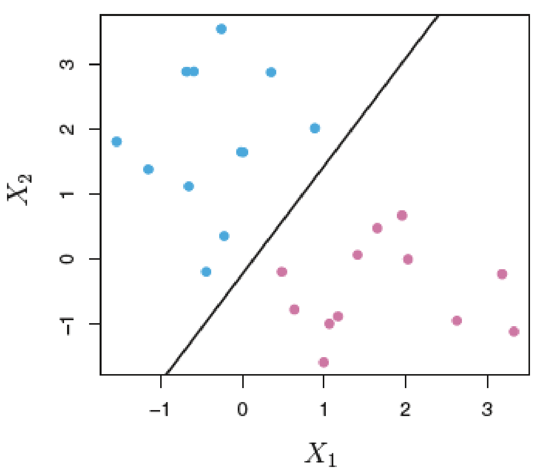
Encontrar β_0 y $\mathbf{w} = (\beta_1, \ldots, \beta_p)$ tal que el M compatible con estas dos condiciones es el más grande posible:

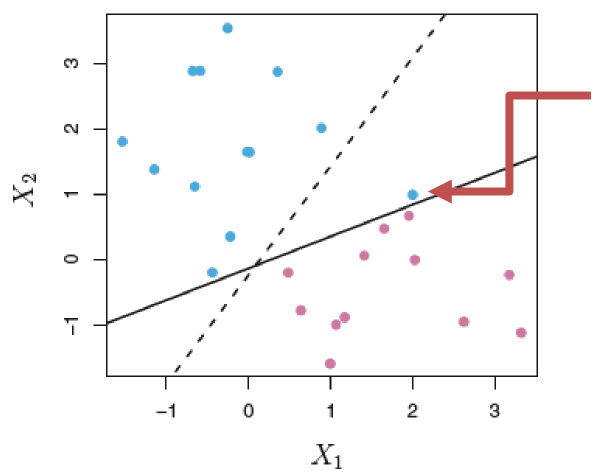
$$\sum_{j=1}^{p} \beta_j^2 = 1,$$

Esta condición no se puede cumplir en el caso no separable

$$\boxed{y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M} \ \forall\, i = 1, \ldots, n$$

... e incluso puede que no sea deseable cumplirla incluso en el caso separable:





Agregar este punto cambia todo...

Clasificador de soporte vectorial

Idea (margen suave): permitir que el clasificador cometa errores, dejando ejemplos del lado incorrecto, siempre y cuando la suma de esos errores esté acotada por alguna constante.

Encontrar β_0 y $\boldsymbol{w} = (\beta_1 \ldots \beta_p)$ tal que el M compatible con estas condiciones es el más grande posible:

Encontrar γ tal que el M compatible con estas condiciones es el más grande posible:

$$\|\boldsymbol{w}\| = 1$$

$$\sum_{j=1}^{p} \beta_j^2 = 1$$

$\epsilon_i = 0$ si está del lado correcto

sino, $\epsilon_i > 0$

si está del lado correcto

sino,

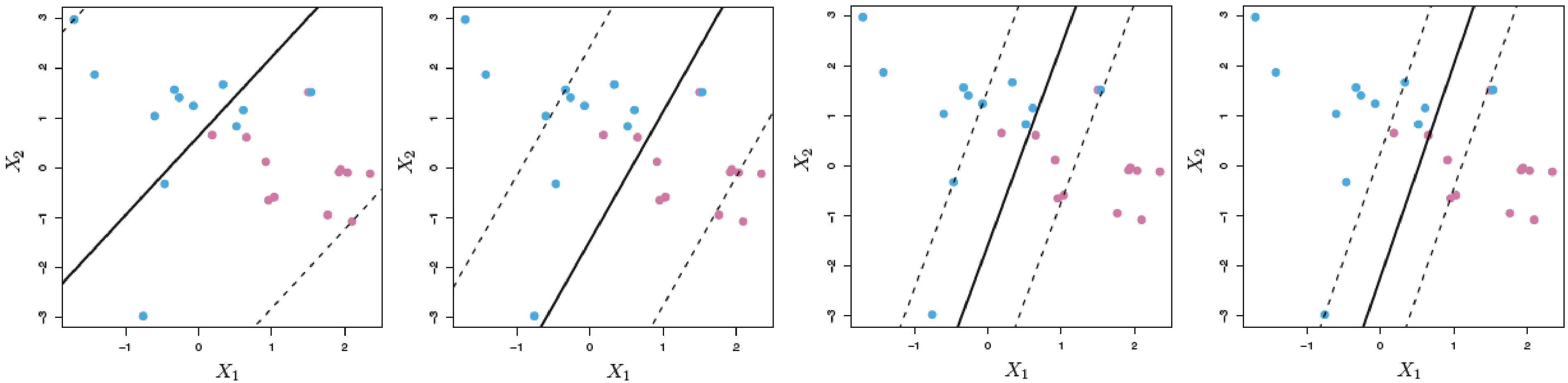
$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i)$$

$$\epsilon_i \geq 0, \boxed{\sum_{i=1}^{n} \epsilon_i \leq C}$$

Suma total de los errores acotada por C
(hiperparámetro que indica que tan tolerante soy con que los ejemplos de entrenamiento queden del lado equivocado)

Clasificador de soporte vectorial

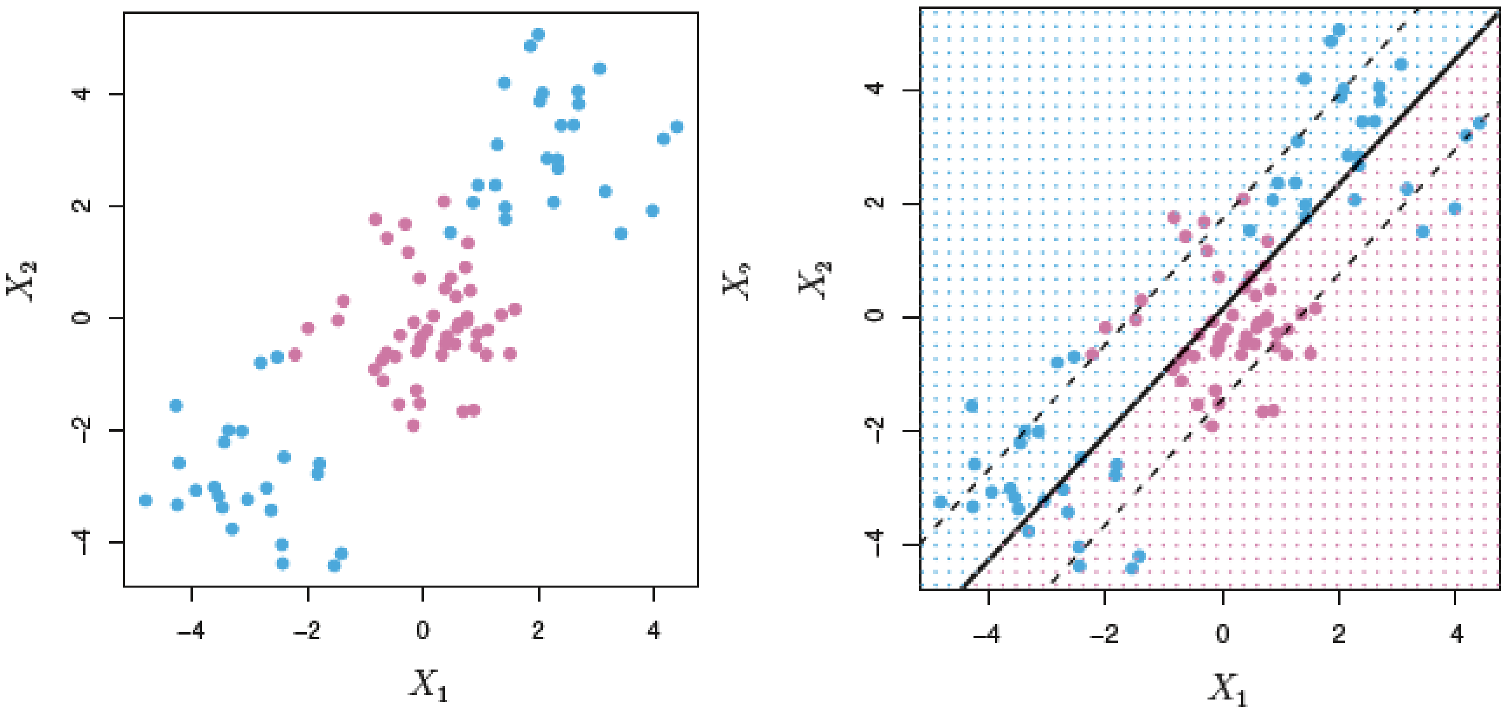
Al ir haciendo C más chico:



Intuitivamente, C sirve para regular el efecto que el ruido de los datos tiene en la frontera que encuentra el clasificador. Si C es chico (margen “duro”), el clasificador intenta separar todos los ejemplos y únicamente el soporte determina el hiperplano. En cambio, si C es más grande (margen “blando”) ya no importa únicamente el soporte, y se tienen en cuenta ejemplos más lejanos respecto del hiperplano.

Clasificador de soporte vectorial

Pero hay casos no separables que son problemáticos para una frontera lineal:



Clasificador de soporte vectorial

¿Qué hacíamos cuando pasaba lo mismo en regresión logística?

Agregar features mediante $\mathrm{x} \to \varphi(\boldsymbol{x})$, por ejemplo, $\varphi(\boldsymbol{x}) = \boldsymbol{x}^2$

$$X_1, X_2, \ldots, X_p \longrightarrow X_1, X_1^2, X_2, X_2^2, \ldots, X_p, X_p^2$$

La frontera es no lineal cuando la miro proyectada en los p features iniciales, pero es lineal cuando la miro en el espacio de 2p features.

Problema: ¿Qué pasa si no conocemos la forma de φ?

Máquinas de soporte vectorial

Llamamos $f(\boldsymbol{x}) = \beta_0 + \langle \boldsymbol{w}, \boldsymbol{x} \rangle$

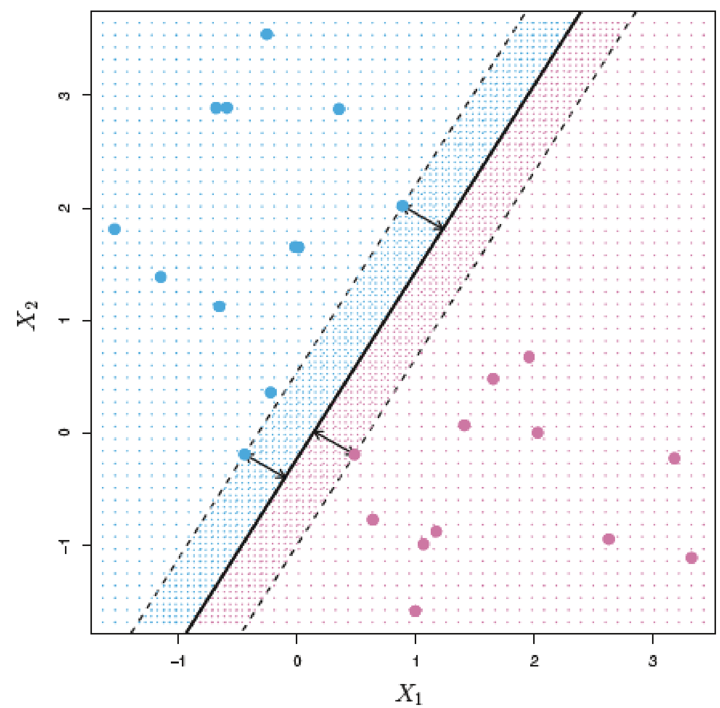
Luego, $f(x)$ me indica que tan cerca o lejos estoy del hiperplano, y me sirve para clasificar

clasificar $\boldsymbol{x}$

Resultado (sin demostración):

Resultado (sin demostración):

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i \langle x, x_i \rangle$$



donde **S es el conjunto de x** que están en el soporte y los α_i dependen únicamente de los β_i

Observación: para calcular nunca aparece solo, sino en producto con los del soporte

Observación: para calcular $f(\boldsymbol{x})$ nunca aparece solo, sino en producto con los $\boldsymbol{x}_i$ del soporte

Clasificador de soporte vectorial

Problema: ¿Qué pasa si no conocemos la forma de φ?

Observación: para calcular $f(x)$ nunca aparece solo, sino en producto con los x_i del soporte

Solución: No necesitamos conocer $\varphi(x)$ siempre que conozcamos como calcular el producto interno entre $\varphi(x)$ y $\varphi(x_i)$, es decir, una medida de cuanto se parecen $\varphi(\boldsymbol{x})$ y $\varphi(\boldsymbol{x}_i)$

La función que me manda x_i y $x_{i'}$ a $\langle\varphi(x_i),\varphi(x_{i'})\rangle$ se llama kernel, $K(x_i, x_{i'})$, **y es todo lo que necesito para determinar el espacio al que transformo mis features originales**

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

transformación identidad, clasificador de soporte vectorial lineal

$$K(x_i, x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d$$

transformación a polinomios de grado máximo d

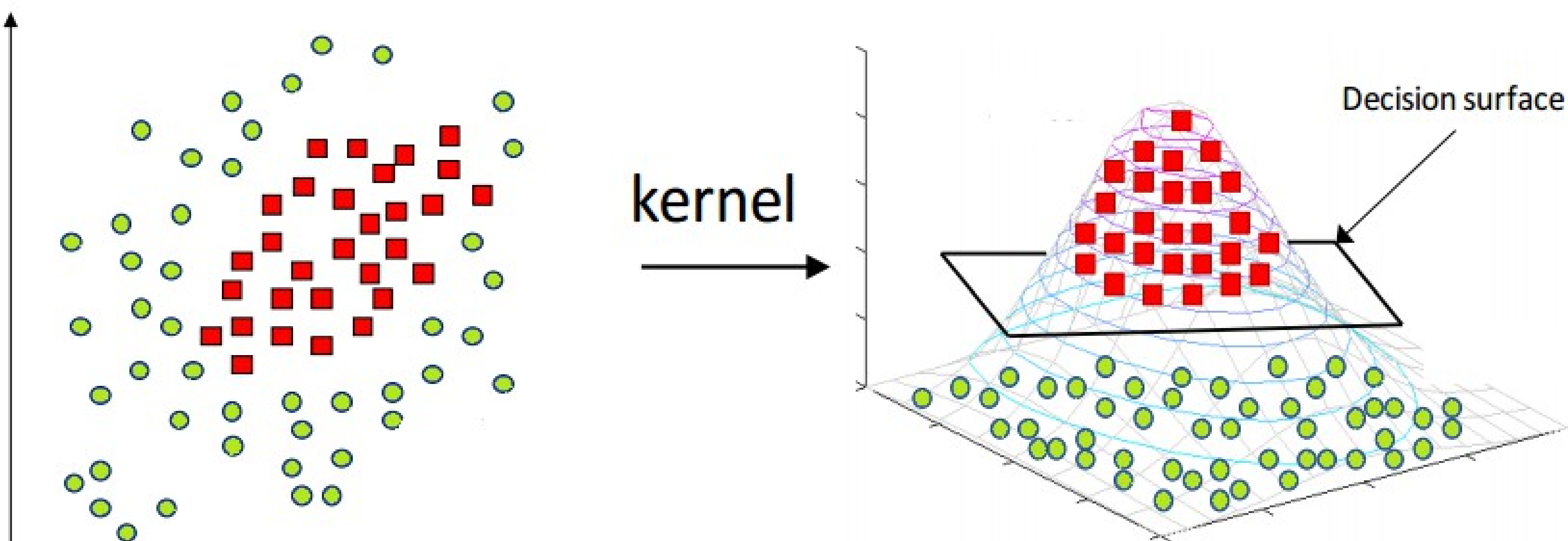
$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2)$$

corresponde a una transformación a un espacio vectorial de dimensión infinita, que se define *implicitamente*

Clasificador de soporte vectorial

Intuición: ¿Qué pasa si no conocemos la forma de φ?

Si no sabemos la transformación no lineal que le tenemos que aplicar a nuestros features, *cambiemos la geometría de nuestro espacio, de forma tal que una "recta" ahora pase a ser otra curva diferente*

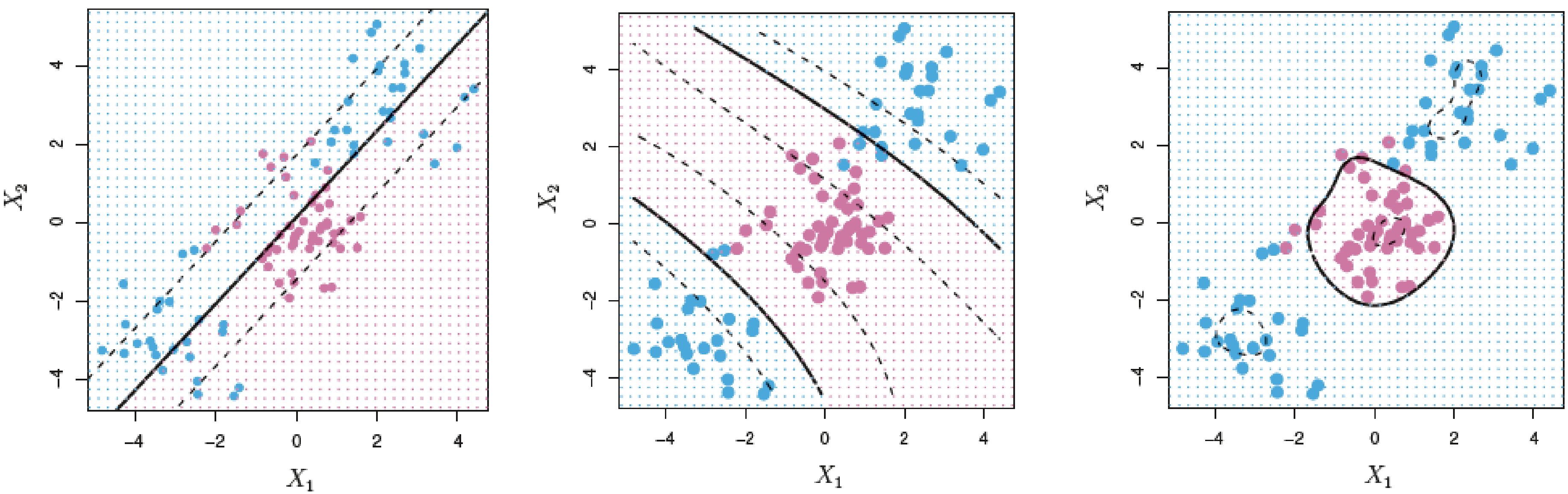


Clasificador de soporte vectorial

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

$$K(x_i, x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d$$

$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2)$$



Clasificador de soporte vectorial

Ventajas

- Funciona bien con pocos datos relativos a la dimensión del espacio de features
- Puede aprender cualquier frontera razonable con el *kernel* adecuado
- La constante de penalización *C* para el margen hace las veces de constante de regularización ridge
- El resultado de evaluar el clasificador depende únicamente de los vectores en el soporte, por lo tanto es rápido de computar, y también fácil de almacenar una vez ajustado
- El tamaño de f(**x**) es interpretable como una medida de distancia al hiperplano
- En general, está basado en intuiciones geométricas
- El problema de optimización es cuadrático, en otras palabras, hay un único mínimo global
- Se puede generalizar a regresión sin mayores problemas

Desventajas

- Para problemas multietiqueta hay que entrenar varios para los problemas binarios correspondientes y hacerlos votar
- Como todos los clasificadores basados en una noción de distancia, puede ser muy sensible a la normalización de los datos
- Hay que tener cuidado con datasets no balanceados
- Hay que trabajar para darle una interpretación probabilística

Contents lists available at SciVerse ScienceDirect

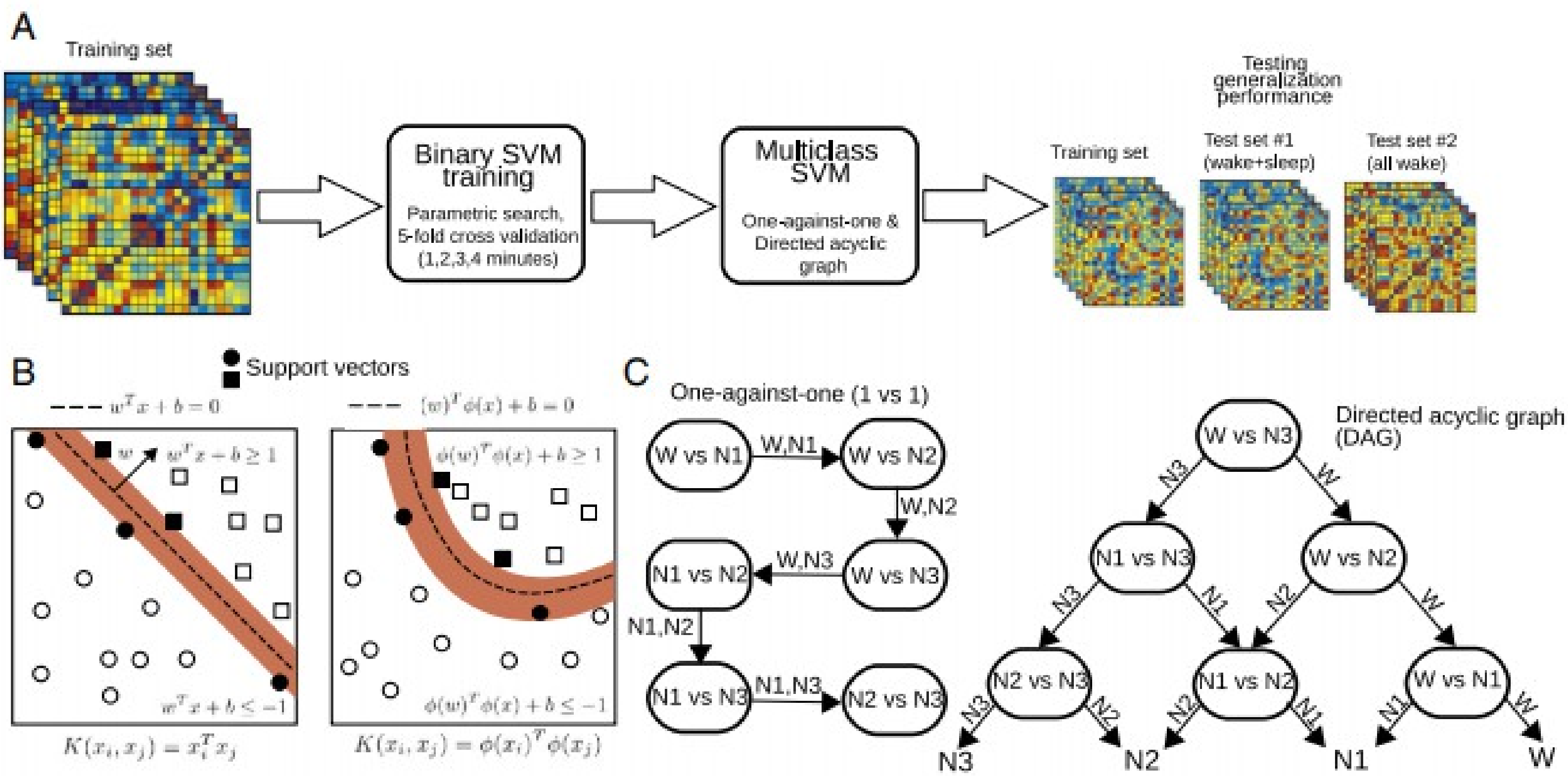
NeuroImage

journal homepage: www.elsevier.com/locate/ynimg

Automatic sleep staging using fMRI functional connectivity data

Enzo Tagliazucchi *, Frederic von Wegner, Astrid Morzelewski, Sergey Borisov, Kolja Jahnke, Helmut Laufs

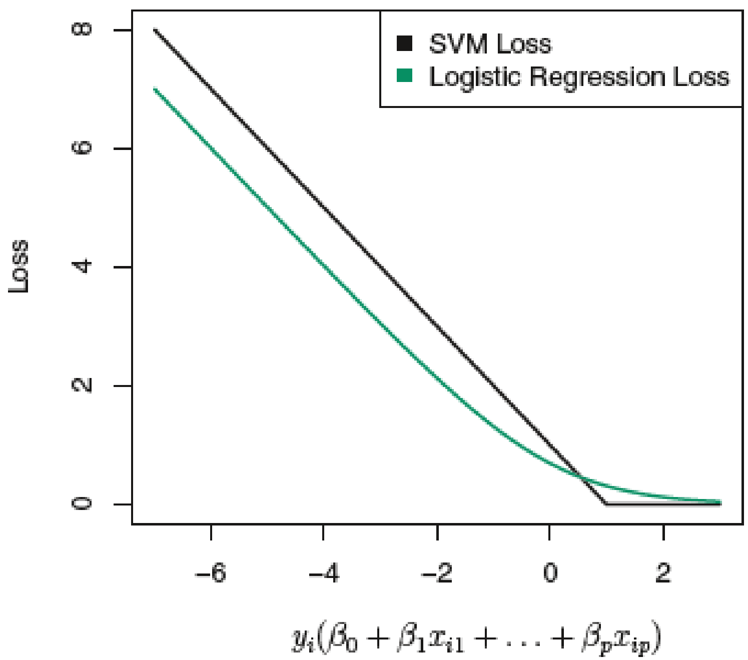
Department of Neurology and Brain Imaging Center, Goethe University Frankfurt am Main, Germany



Clasificador de soporte vectorial

¿Cuál es la relación entre SVM y regresión logística?

$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{minimize}} \left\{ L(\mathbf{X}, \mathbf{y}, \beta) + \lambda P(\beta) \right\}$$



When the support vector classifier and SVM were first introduced, it was thought that the tuning parameter C in (9.15) was an unimportant "nuisance" parameter that could be set to some default value, like 1. However, the "Loss + Penalty" formulation (9.25) for the support vector classifier indicates that this is not the case. The choice of tuning parameter is very important and determines the extent to which the model underfits or overfits the data, as illustrated, for example, in Figure 9.7.

C es equivalente a la constante de regularización ridge en regresión logística

sklearn.svm.SVC¶

class `sklearn.svm.SVC`(**, C=1.0, kernel='rbf', degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter=- 1, decision_function_shape='ovr', break_ties=False, random_state=None*) [source]

C = penalización de margen suave, regularización

kernel = linear, poly, rbf, sigmoid, precomputado

degree = grado del kernel polinómico (en caso de usarlo)

gamma = parámetro del kernel rbf

coef0 = término independiente en kernel poly o sigmoid

probability = hace un 5 fold cross validation para estimar las probabilidades de pertenecer a cada clase (costoso computacionalmente)

class_weight = pesa diferente los errores en el problema de optimización, de forma tal que se puede compensar por datos desbalanceados