Máquinas de soporte vectorial

Support Vector Machine - SVM

Machine Learning Lab
Sevilla

José Manuel Camacho Sosa - hello@josemazo.com

20 de mayo de 2016

Contenido.

Contenido.

1. Introducción.

Las máquinas de soporte vectorial (SVM) son, a priori, un **clasificador lineal biclase**, pero como veremos más adelante también puede realizar tareas más complejas.

Este método, tras entrenarlo, nos devolverá un **discriminante**, el vector normal al hiperplano que separa a las dos clases, además de otros atributos necesarios para clasificar entradas posteriores, la cual es un tarea instantánea.

Además, la complejidad de su solución sólo depende del número de entradas, y no del número de dimensiones del espacio de dichas entradas. Como veremos más adelante, esto nos será muy útil.

Contenido.

2. Hiperplano de separación óptima.

Supongamos que nuestras clases tienen las etiquetas *-1* y *+1*, y tenemos una muestra $\mathcal{X} = \{x^t, r^t\}$ donde $r^t = +1$ si $x^t \in C_1$ y $r^t = -1$ si $x^t \in C_2$. Tenemos que encontrar w y w_0 tal que:

$$r^t (w^T x^t + w_0) \geq +1$$

Además, aparte de querer que las instancias estén en el lado correcto del hiperplano representado por ***w***, la distancia del hiperplano a las instancias más cercana, llamada **margen**, deberá ser máxima para conseguir la mayor **generalización** posible, por lo que lo llamaremos el **hiperplano de máxima separación**. Para maximizar el margen tenemos que minimizar $\| w \|$, tarea que puede ser definida como:

$$\min (1/2) \| w \|^2 \quad \text{sujeto a} \quad r^t (w^T x^t + w_0) \geq +1, \ \forall t$$

2. Hiperplano de separación óptima.

Este es un problema de optimización cuadrática, cuya complejidad depende de *d*, el número de dimensiones del espacio de entrada.

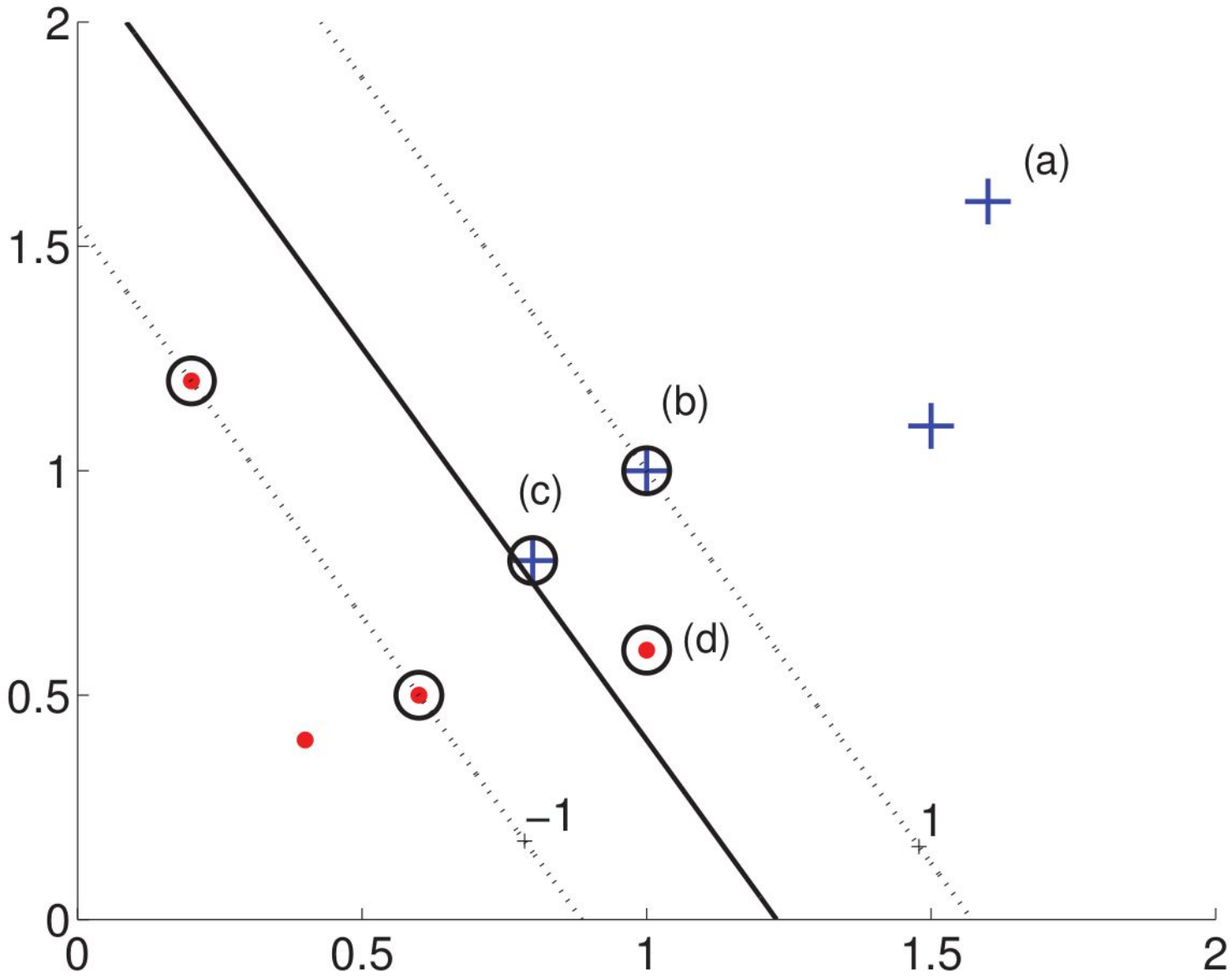
Pero reformulando la expresión como un lagrangiano (con sus restricciones y parámetros), la complejidad solo depende de *N*, el número de instancias, pudiéndose resolver aún con métodos de optimización cuadrática. Así, la complejidad temporal sería $O(N^3)$ y la espacial $O(N^2)$.

$$L_p = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_t \alpha^t r^t (\boldsymbol{w}^T \boldsymbol{x}^t + w_0) + \sum_t \alpha^t$$

Una vez obtenido w y w_0, con el signo de $g(x) = w^T x + w_0$ podremos saber a qué clase pertenece la instancia.

Al discriminante w, se le llama **máquina de vectores soporte**, y las instancias que quedan en el borde de los márgenes o dentro de ellos son los **vectores soporte**.

2. Hiperplano de separación óptima.



2. Hiperplano de separación óptima.

A la hora de entrenar el modelo se añaden más parámetros para intentar buscar discriminantes que generalicen mejor. Estos se pueden dar como **variables flojas** o variables de prueba en **validación cruzada**.

Por ejemplo, uno de estas variables es el error de bisagra, ξ^t, que penaliza mucho a las instancias mal clasificadas, y también a las instancias en el margen del discriminante (aunque estén bien clasificadas), con mayor penalización mientras más cerca estén de éste.

Contenido.

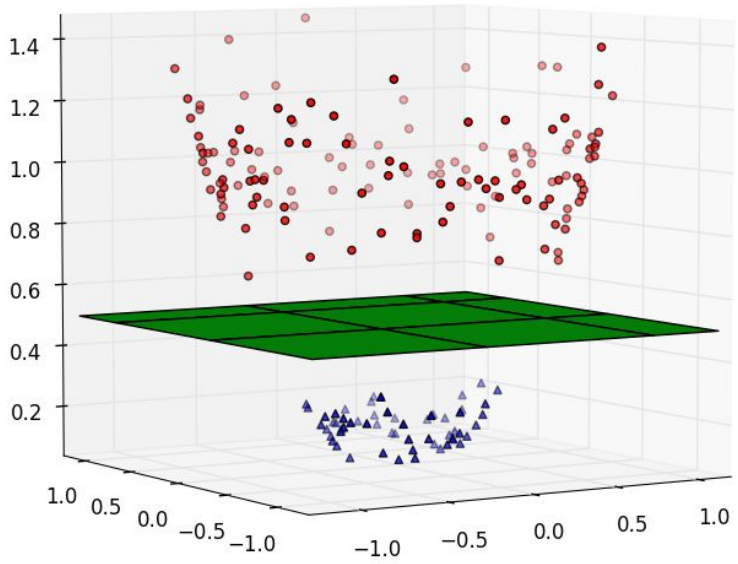
3. Truco del kernel.

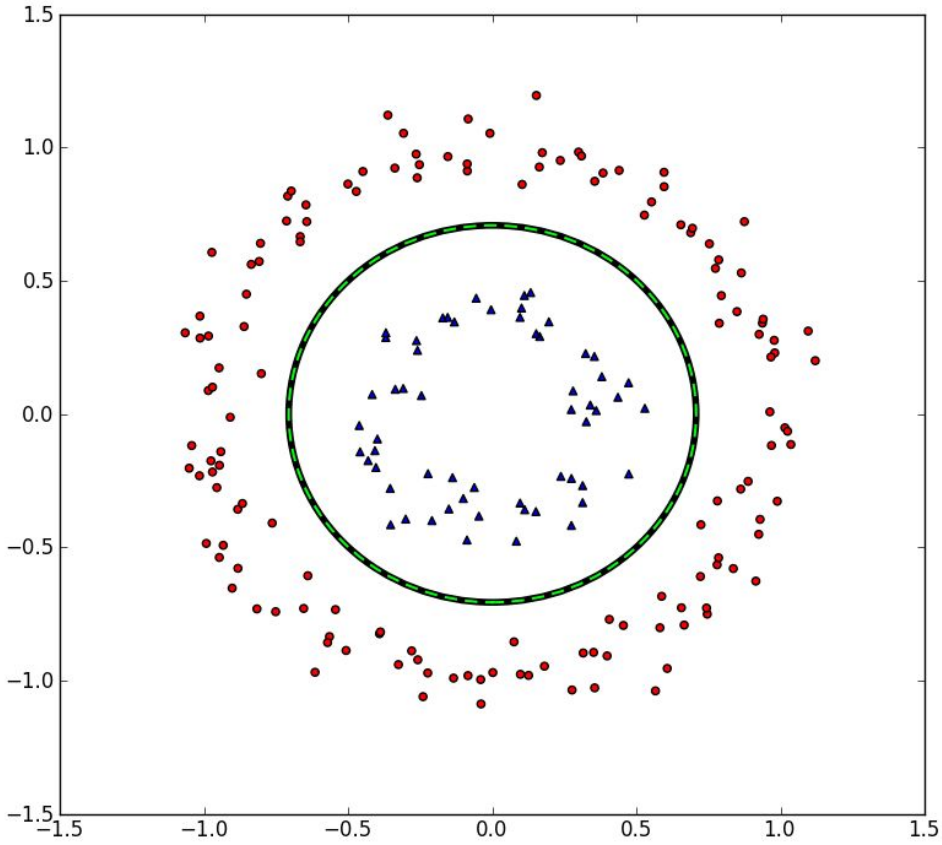
Un problema donde las entradas sean linealmente separables es muy idílico en el mundo real. Para esto existe una solución, mapear el problema a un nuevo espacio utilizando **transformaciones no lineales**, $z = \Phi(x)$, donde en este espacio el problema si es linealmente separable.

El nuevo espacio generalmente tiene un **número de dimensiones mayor** que el original, pero esto nos da igual, ya que nuestro método para resolver el lagrangiano sólo depende el número de elementos de entrada, *N*.

Por la naturaleza del lagrangiano, una forma fácil de hacer este cambio de espacio en la expresión es añadir los valores de una matriz de Gram con los valores del kernel, de la forma ***K(x, y)***.

3. Truco del kernel.





3. Truco del kernel.

Gaussian Radial Kernel: $K(x, y) = e^{-c^2||x-y||^2}$

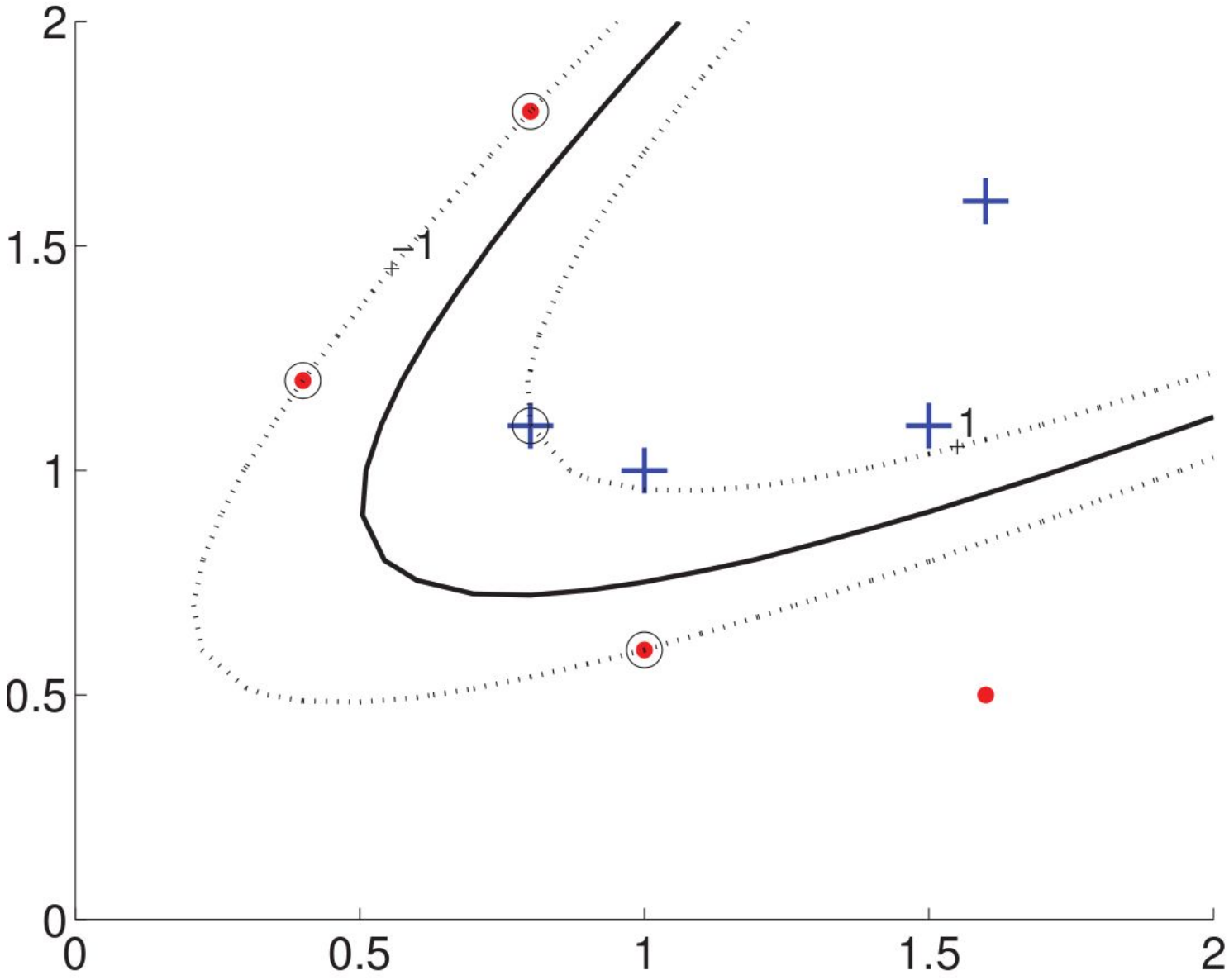
Polynomial Kernel: $K(x, y) = (x'y + c)^d$

Exponential Kernel: $K(x, y) = e^{-\alpha||x-y||}, \alpha > 0$

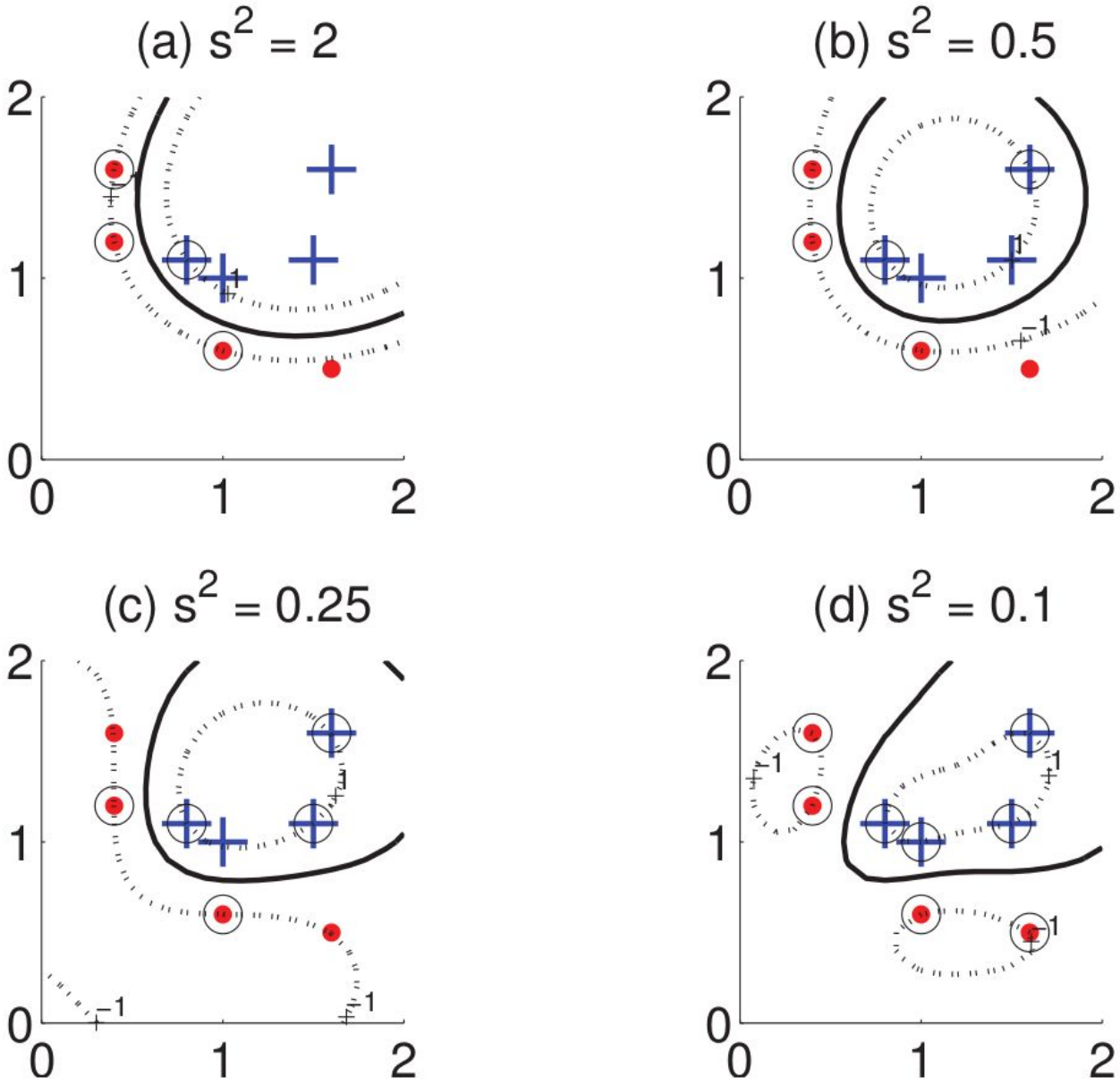
Sigmoid Kernel: $K(x, y) = \tanh(cx'y + d)$

Inverse Quadratic Kernel: $K(x, y) = \frac{1}{\alpha^2||x-y||^2+1}$

3. Truco del kernel.



3. Truco del kernel.



Contenido.

4. Otros usos.

Siendo lo visto hasta ahora la base de los SVM y su único caso, clasificación biclase, con pequeñas extensiones se pueden utilizar para:

- **Clasificación multiclase**: Utilizando varios discriminantes, cada uno en modo *one vs. all*.
- **Regresión**: Ajustando el resultado de la regresión al discriminante.
- **Ranking**: Utilizando pares de clases, determinamos si una instancia entre cual y cual.
- **Clustering**: Buscando esferas envolventes multidimensionales a través de kernels gaussianos.
- **Reducción de la dimensionalidad**: Haciendo una versión *kernelizada* de PCA.

Contenido.

5. Conclusiones

- Una vez entrenado el modelo clasifica instantáneamente, solo necesita calcula el signo de $g(x) = w^T x + w_0$, unido a que la eficiencia del entrenamiento solo está ligada al tamaña de la muestra hace que sea un algoritmo a tener en cuenta…

- … si no fuera tan difícil de parametrizar. La cantidad de parámetros, siendo uno de ellos un kernel vectorial, hace muy difícil encontrar el valor necesario en sus parámetros para cada caso.

- Aunque se pueda hacer regresión o clustering, es intentar crear un navaja suiza en un ámbito donde casi no se puede realizar esta tarea, y así queda demostrado en temas como el ranking o el PCA kernelizado.

Contenido.

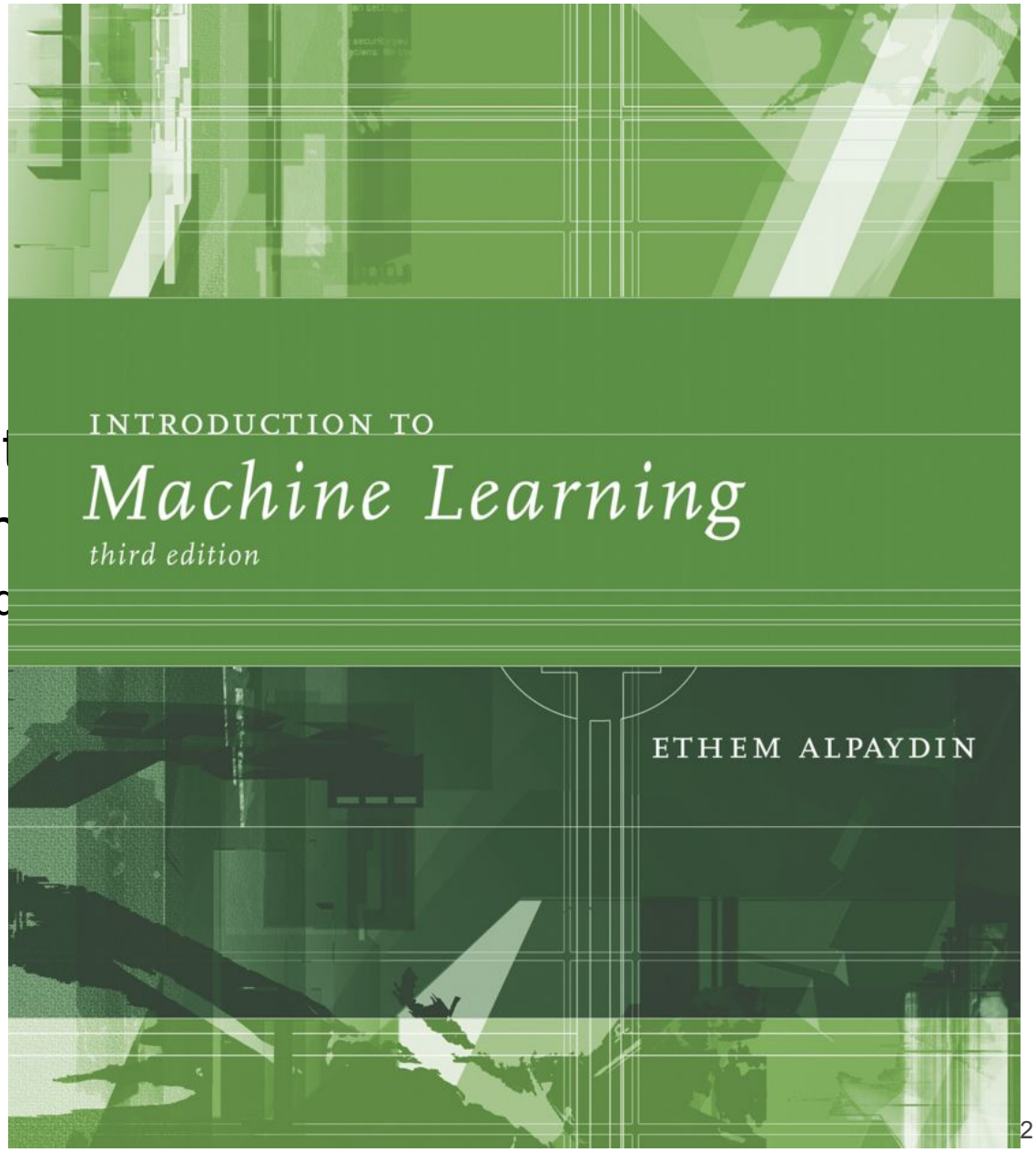
6. Referencias.

Introduction t

Machine Learnin

Ethem Alpayc

3rd Edition - 2014



Contenido.

7. Ejemplo con scikit-learn.