Aprendizaje automatizado

Kernels

Gibran Fuentes-Pineda

Mayo 2023

Representación dual

- Problema de optimización

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y^{(i)} y^{(j)} \underbrace{\left[\left(\mathbf{x}^{(i)} \right)^\top \mathbf{x}^{(j)} \right]}_{k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})}$$

$$\text{sujeto a } 0 \leq \alpha_i \leq C, \forall i, \sum_{i=1}^{n} \alpha_i y^{(i)} = 0$$

donde $\mathbf{w} = \sum_{i=1}^{n} \alpha_i \cdot y^{(i)} \cdot \mathbf{x}^{(i)}$

- Para predecir la clase de una nueva instancia $\tilde{\mathbf{x}}$

$$\tilde{y} = \text{signo} \left(\sum_{i=1}^{n} \alpha_i y^{(i)} \underbrace{\left[\left(\mathbf{x}^{(i)} \right)^\top \tilde{\mathbf{x}} \right]}_{k(\mathbf{x}^{(i)}, \tilde{\mathbf{x}})} + b \right)$$

¿Qué es una función de kernel?

- Función evaluada en los reales $k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \in \mathbb{R}$
  - Simétrica: $k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = k(\mathbf{x}^{(j)}, \mathbf{x}^{(i)})$
  - No negativa: $k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \geq 0$

¿Qué es una función de kernel?

- Función evaluada en los reales $k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \in \mathbb{R}$
  - Simétrica: $k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = k(\mathbf{x}^{(j)}, \mathbf{x}^{(i)})$
  - No negativa: $k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \geq 0$

- Puede ser vista como una medida de similitud (aunque no necesariamente debe ser una)

¿Qué es una función de kernel?

- Función evaluada en los reales $k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \in \mathbb{R}$
  - Simétrica: $k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = k(\mathbf{x}^{(j)}, \mathbf{x}^{(i)})$
  - No negativa: $k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \geq 0$

- Puede ser vista como una medida de similitud (aunque no necesariamente debe ser una)

- Para mapeos a espacios no lineales $\phi(\mathbf{x}^{(i)})$, el kernel está dado por

$$k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \left(\phi(\mathbf{x})^{(i)}\right)^{\top} \phi(\mathbf{x}^{(j)})$$

Ejemplos de funciones de kernel

- Lineal

$$k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \left(\mathbf{x}^{(i)}\right)^{\top} \mathbf{x}^{(j)}$$

Ejemplos de funciones de kernel

- Lineal

$$k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \left(\mathbf{x}^{(i)}\right)^{\top} \mathbf{x}^{(j)}$$

- Gaussiana

$$k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp\left(-\frac{1}{2}\left[\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right]^{\top} \Sigma^{-1} \left[\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right]\right)$$

Ejemplos de funciones de kernel

- Lineal

$$k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \left(\mathbf{x}^{(i)}\right)^\top \mathbf{x}^{(j)}$$

- Gaussiana

$$k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp\left(-\frac{1}{2}\left[\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right]^\top \Sigma^{-1}\left[\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right]\right)$$

- Función de base radial (RBF)

$$k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp\left(-\frac{\|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2}{2\sigma^2}\right)$$

Ejemplos de funciones de kernel

- Lineal

$$k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \left(\mathbf{x}^{(i)}\right)^\top \mathbf{x}^{(j)}$$

- Gaussiana

$$k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp\left(-\frac{1}{2}\left[\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right]^\top \Sigma^{-1}\left[\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right]\right)$$

- Función de base radial (RBF)

$$k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp\left(-\frac{\|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2}{2\sigma^2}\right)$$

- Similitud coseno

$$k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \frac{\left(\mathbf{x}^{(i)}\right)^\top \mathbf{x}^{(j)}}{\|\mathbf{x}^{(i)}\| \cdot \|\mathbf{x}^{(j)}\|}$$

El truco del kernel

- Proyectamos el espacio de entrada a un espacio de más alta dimensionalidad en la que sea posible separar las clases linealmente

- Muchos algoritmos se pueden *kernelizar* usando la representación dual
  - Substituímos producto punto en representación dual por una llamada a un kernel

- Es necesario definir funciones de kernel válidas
  - Elegir un mapeo $\phi(\mathbf{x}^{(i)})$ y definir el kernel en base a este.
  - Definir directamente funciones, sin conocer $\phi(\mathbf{x}^{(i)})$
  - Ciertas operaciones sobre funciones válidas producen otras funciones válidas

Kernels positivos definidos (Mercer)

- Si la matriz de Gram es positiva definida, se conoce como kernel de Mercer

$$\mathbf{K} = \begin{pmatrix} k(\mathbf{x}^{(1)}, \mathbf{x}^{(1)}) & \cdots & k(\mathbf{x}^{(1)}, \mathbf{x}^{(n)}) \\ & \vdots & \\ k(\mathbf{x}^{(n)}, \mathbf{x}^{(1)}) & \cdots & k(\mathbf{x}^{(n)}, \mathbf{x}^{(n)}) \end{pmatrix}$$

- La eigendescomposición de $\mathbf{K}$ está dada por

$$\mathbf{K} = \mathbf{U}^\top \Lambda \mathbf{U}$$

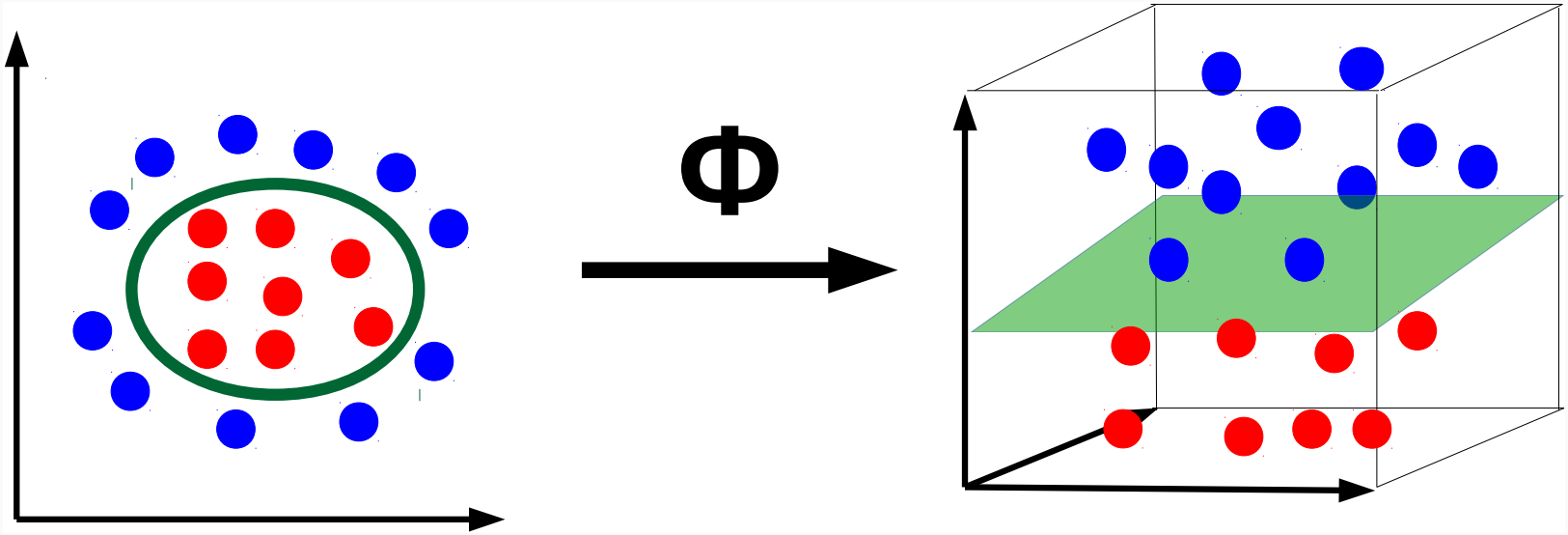
donde

$$k_{ij} = \left(\Lambda^{\frac{1}{2}} \mathbf{U}_{:,i}\right)^\top \left(\Lambda^{\frac{1}{2}} \mathbf{U}_{:,j}\right)$$

- Si un kernel es Mercer, existe un mapeo $\phi(\mathbf{x}^{(i)})$ tal que

$$k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \left(\phi(\mathbf{x})^{(i)}\right)^\top \phi(\mathbf{x}^{(j)})$$

Intuición de clasificación con kernels



SVM con kernel lineal

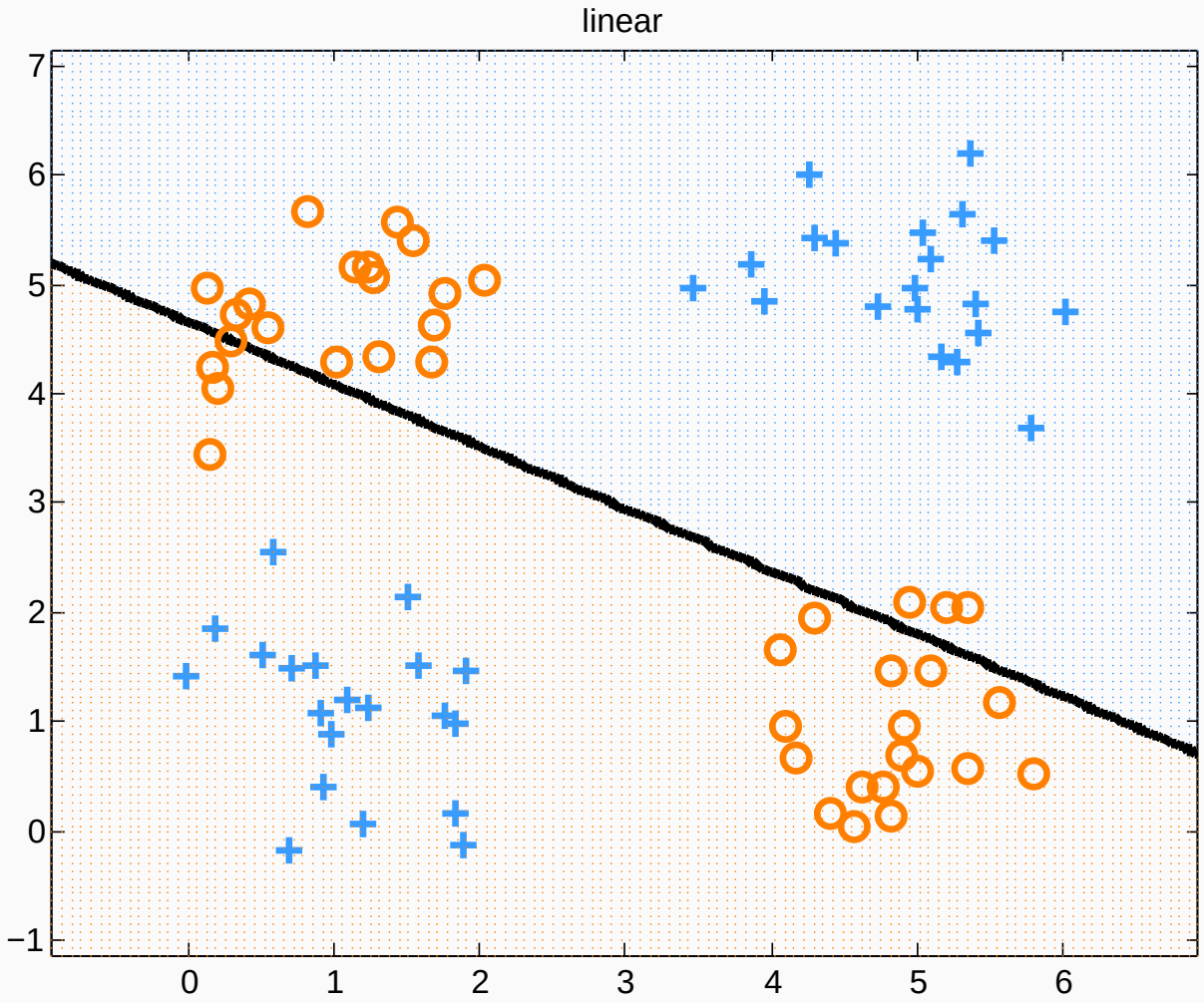


Imagen generada usando ejemplo de `https://github.com/probml/pmtk3`

SVM con función de base radial

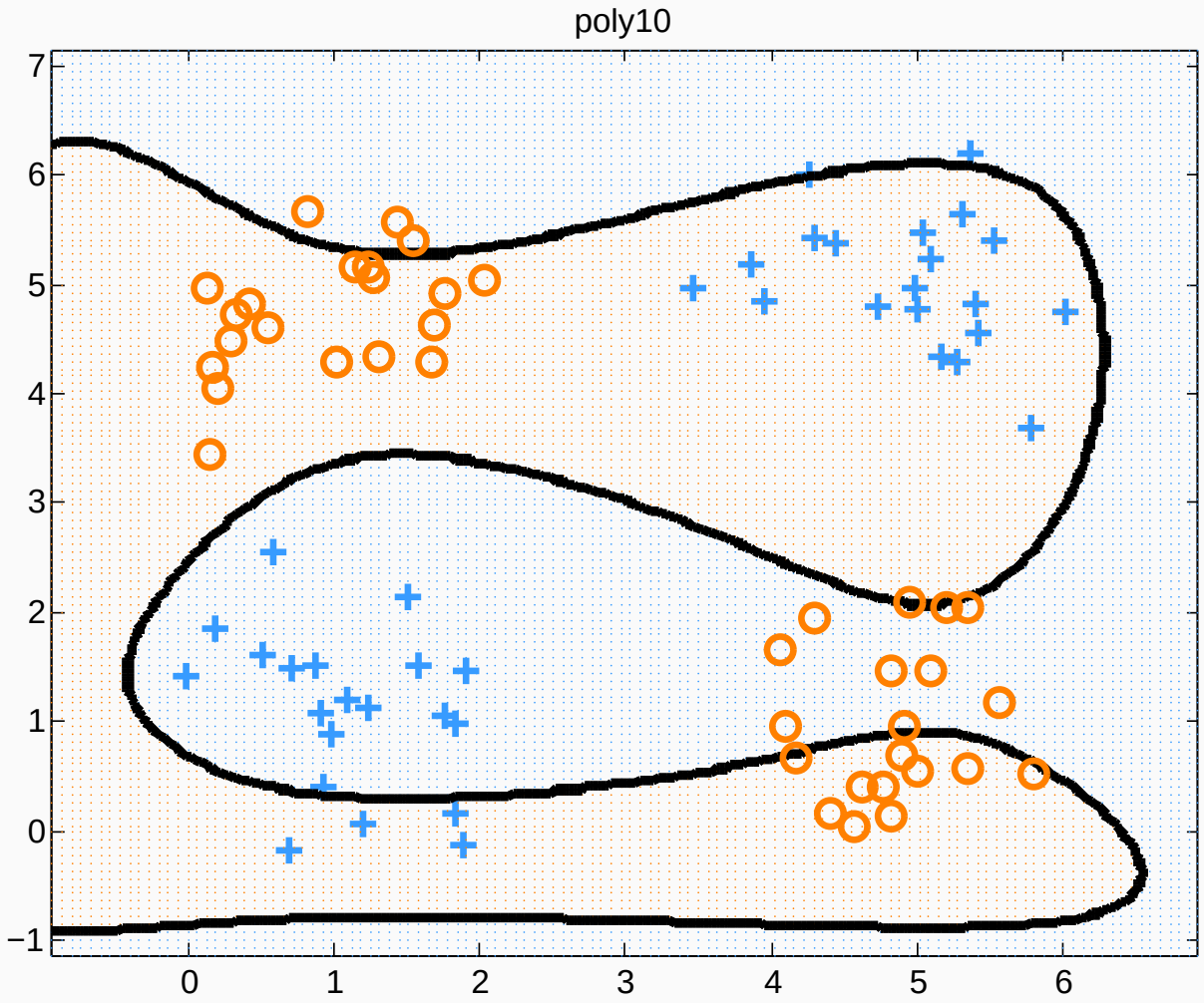


Imagen generada usando ejemplo de `https://github.com/probml/pmtk3`

SVM con kernel polinomial

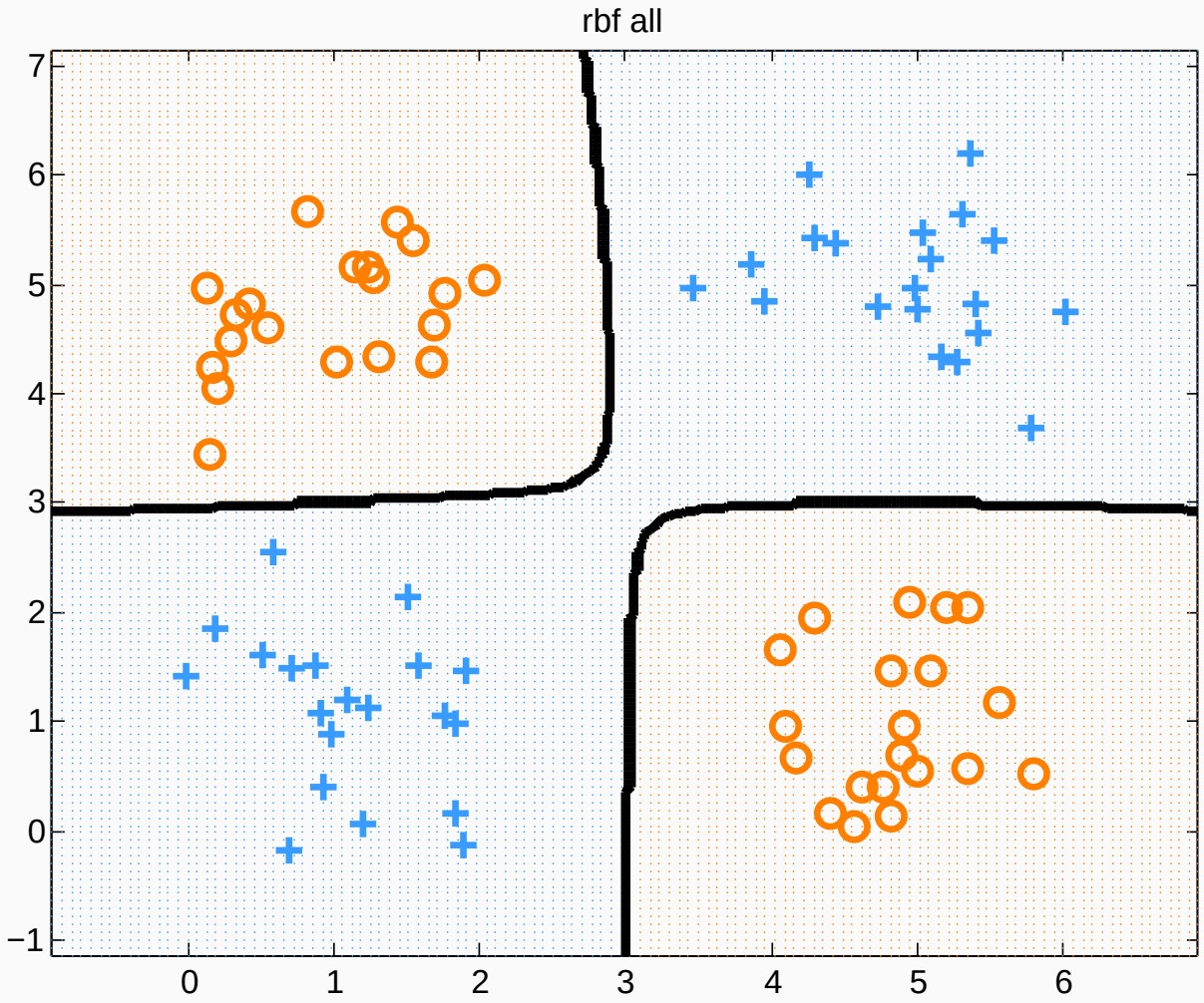


Imagen generada usando ejemplo de `https://github.com/probml/pmtk3`

Algoritmo de optimización mínima secuencial (SMO)

- Divide el problema de optimización en una serie de subproblemas con 2 multiplicadores de Lagrange (es el mínimo debido a la restricción de desigualdad lineal)

- Es posible optimizar cada subproblema de forma analítica

$$0 \leq \alpha_1, \alpha_2 \leq C$$
$$y^{(1)} \cdot \alpha_1 + y^{(2)} \cdot \alpha_2 = k$$

donde k es el negativo de la suma del resto de los términos de la restricción de igualdad

Algoritmo de descenso por subgradiente (PEGASOS)

- La función bisagra no es diferenciable, pero podemos usar el subgradiente

$$\tilde{\nabla} E(\mathbf{w}, b) = \begin{cases} 0, & y^{(i)} \cdot (\mathbf{w}^\top \mathbf{x}^{(i)} + b) \geq 1 \\ y^{(i)} \cdot \mathbf{x}^{(i)}, & y^{(i)} \cdot (\mathbf{w}^\top \mathbf{x}^{(i)} + b) < 1 \end{cases}$$

- Algoritmo
  1. Inicializamos $\mathbf{w}$ y b a 0
  2. Para $t = 1, \ldots, T$ realizar
     - 2.1 Elige ejemplo $\{x^{(i)}, y^{(i)}\}$ aleatoriamente
     - 2.2 $\eta^{\{t\}} = \frac{1}{\lambda \cdot t}$
     - 2.3 Si $y^{(i)} \left(\left(\mathbf{w}^{\{t\}} \right)^\top \mathbf{x}^{(i)} + b \right) < 1$

$$\mathbf{w}^{\{t+1\}} = (1 - \eta^{\{t\}} \cdot \lambda) \cdot \mathbf{w}^{\{t\}} + \eta^{\{t\}} \cdot y^{(i)} \cdot \mathbf{x}^{(i)}$$

     - 2.4 En caso contrario

$$\mathbf{w}^{\{t+1\}} = (1 - \eta^{\{t\}} \cdot \lambda) \cdot \mathbf{w}^{\{t\}}$$

PEGASOS kernelizado

- Es posible entrenar un SVM definido a partir de los parámetros

- Esta versión *kernelizada* es la siguiente:
  1. Inicializa $\alpha_i^{\{0\}}, i = 1, \ldots, n$ a 0
  2. Para $t = 1, \ldots, T$ realizar
     - 2.1 Elige el índice de un ejemplo aleatoriamente $s \in \{1, \ldots, n\}$
     - 2.2 $\eta^{\{t\}} = \frac{1}{\lambda \cdot t}$
     - 2.3 Si $y^{(s)} \cdot \left[\eta^{\{t\}} \cdot \sum_{i=1}^{n} \alpha_i \cdot y^{(i)} \cdot K(\mathbf{x}^{(i)}, \mathbf{x}^{(s)})\right] < 1$, entonces

$$\alpha_s^{\{t\}} = \alpha_s^{\{t-1\}} + 1$$

     - 2.4 En caso contrario

$$\alpha_s^{\{t\}} = \alpha_s^{\{t-1\}}$$