UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escola Tècnica Superior d'Enginyeria **Informàtica**

etsinf

Tema 4. Representación basada en Kernels

Percepción (PER)

Curso 2017/2018

Departamento de Sistemas Informáticos y Computación

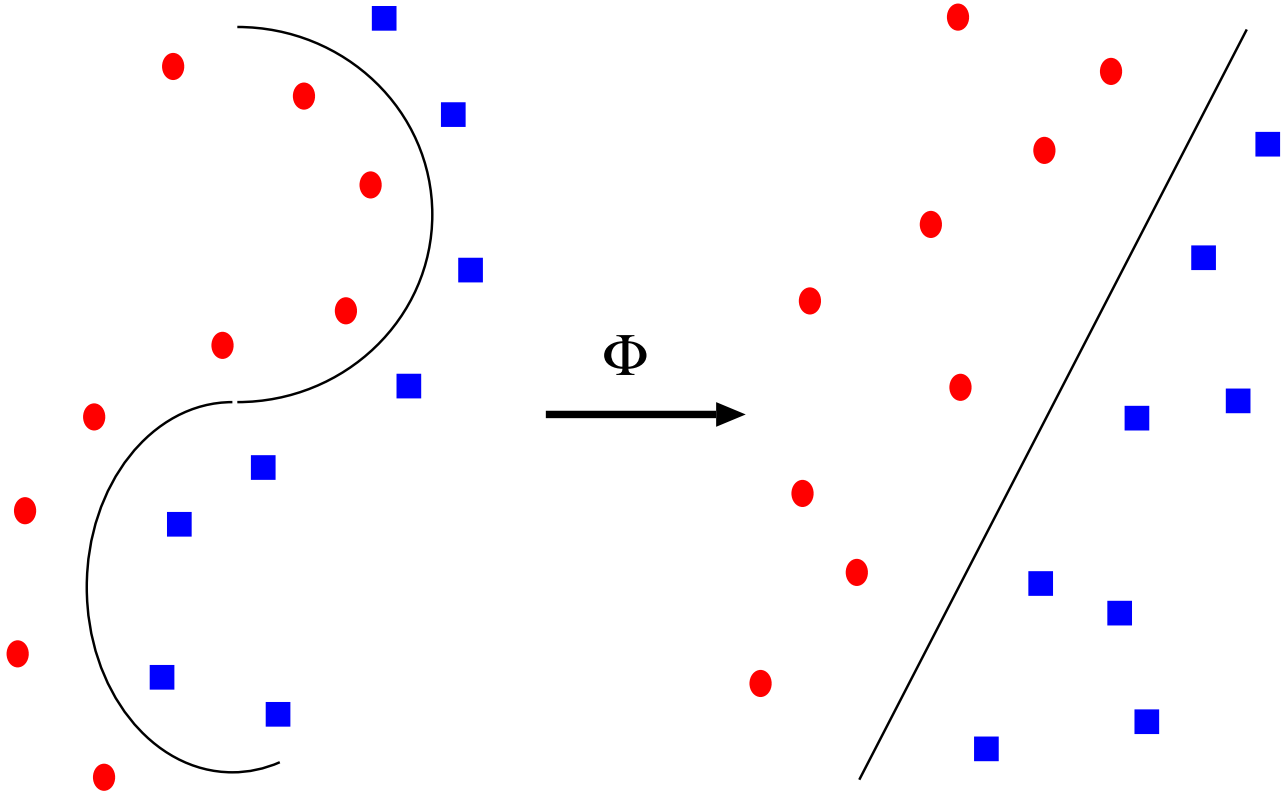
Índice

Índice

Introducción

- Objetivo principal de la representación basada en kernels es:

  Cambio de espacio de representación para obtener separabilidad lineal



Índice

Clasificación binaria y Kernels

- Usualmente se estudian los métodos kernel en problemas de clasificación binarios
- Conjunto de entrenamiento

$$X = \{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), \cdots, (\mathbf{x}_n, c_n)\} \quad \text{con} \quad c_i \in \{-1, +1\}$$

- La clasificación de una nueva muestra $\mathbf{x}$ se realiza por el signo de una función discriminante $g(\mathbf{x})$:

$$c(\mathbf{x}) = \begin{cases} +1 & \text{si} \quad g(\mathbf{x}) \geq 0 \\ -1 & \text{si} \quad g(\mathbf{x}) < 0 \end{cases}$$

- El algoritmo Perceptron se puede reescribir para la clasificación en dos clases

Aprendizaje - Perceptron de 2 clases

- Entrada: $\{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), \cdots, (\mathbf{x}_n, c_n)\}$ y factor de aprendizaje α
- Salida: $\mathbf{w}$ y w_0 // vector de pesos entrenados y término independiente
- Algoritmo:

$\mathbf{w} = \mathbf{0}$; $w_0 = 0$ // vector de pesos iniciales y peso umbral nulos
do
    $m = 0$; // número de muestras bien clasificadas
    for $(i = 1; i \leq n; i{+}{+})$
        $g(\mathbf{x}_i) = \mathbf{w}^t \cdot \mathbf{x}_i + w_0$
        if $c_i \cdot g(\mathbf{x}_i) \leq 0$ **then** // Si hay un error de clasificación
            $\mathbf{w} = \mathbf{w} + \alpha\, c_i\, \mathbf{x}_i$; $w_0 = w_0 + \alpha\, c_i$
        else
            $m = m + 1$
while $(m < n)$

Aprendizaje - Perceptron de 2 clases

- Entrada: $\{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), \cdots, (\mathbf{x}_n, c_n)\}$ y factor de apr. $\alpha \in \mathbb{R}^n \wedge \alpha_i = \alpha_j \; \forall i, j$
- Salida: $g(\mathbf{x})$ // función de clasificación
- Algoritmo:

$g(\mathbf{x}) = 0$
do
    $m = 0$; // número de muestras bien clasificadas
    for $(i = 1; \; i \leq n; \; i{+}{+})$

        if $c_i \cdot g(\mathbf{x}_i) \leq 0$ **then** // Si hay un error de clasificación
            $g(\mathbf{x}) = g(\mathbf{x}) + \alpha_i \, c_i \, (\mathbf{x}_i^t \cdot \mathbf{x}) + \alpha_i \, c_i$
        else
            $m = m + 1$;
while $(m < n)$

Clasificación binaria y Kernels

- $g(\mathbf{x}) = \mathbf{w}^t \cdot \mathbf{x} + w_0$ es un clasificador con vector de pesos $\mathbf{w}$ y peso umbral w_0:

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i \, c_i \, \mathbf{x}_i \qquad w_0 = \sum_{i=1}^{n} \alpha_i \, c_i$$

- $g(\mathbf{x})$ relaciona $\mathbf{x}$ con algunas muestras de entrenamiento por el producto escalar y α_i pasa de factor de aprendizaje al peso de cada muestra:

$$g(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \, c_i \, (\mathbf{x}_i^t \cdot \mathbf{x}) + \alpha_i \, c_i$$

- Generalizar producto escalar para resolver tareas que no son linealmente separables
- Reemplazar producto escalar por una función ***kernel*** $K(\mathbf{x}_i, \mathbf{x})$, que proyecta a un espacio donde las muestras son linealmente separables y realiza el producto escalar.
- La proyección es **implícita** y se obtiene al calcular la función kernel:

$$g(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \, c_i \, K(\mathbf{x}_i, \mathbf{x}) + \alpha_i \, c_i = \sum_{i=1}^{n} \alpha_i \, c_i \, (\Phi(\mathbf{x}_i^t) \cdot \Phi(\mathbf{x})) + \alpha_i \, c_i$$

Clasificación binaria y Kernels

- ***Función kernel***: función que dado un par de objetos del espacio de representación original nos devuelve un valor real:

$$K : E \times E \to \mathbb{R}$$

- Usualmente la representación es vectorial, entonces:

$$K : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$$

- Dicho valor real modela el producto escalar de esos dos objetos en un nuevo espacio de representación:

$$K(\mathbf{x}, \mathbf{y}) = \Phi(\mathbf{x}) \cdot \Phi(\mathbf{y})$$

- ***La representación alternativa no se llega a producir***, sólo se necesita el ***resultado*** del producto escalar en esa representación para usarlo en un clasificador lineal

Índice

Aprendizaje - Kernel Perceptron

- El algoritmo Kernel Perceptron aprende la siguiente función:

$$g(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \, c_i \, K(\mathbf{x}_i, \mathbf{x}) + \alpha_i \, c_i$$

- Es decir, **una función lineal en un espacio de representación alternativo**:

$$g(\mathbf{x}) = \mathbf{w} \, \Phi(\mathbf{x}) + w_0$$

con

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i \, c_i \, \Phi(\mathbf{x}_i) \qquad w_0 = \sum_{i=1}^{n} \alpha_i \, c_i$$

- Los únicos parámetros a aprender son los α_i
- En fase de aprendizaje la función kernel se representa por una matriz $\mathbf{K} \in \mathbb{R}^{n \times n}$ tal que $\mathbf{K}_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$ (***matriz Gramm***)

Aprendizaje - Kernel Perceptron

Desde el punto de vista de la función a aprender:

- Entrada: $\{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), \cdots, (\mathbf{x}_n, c_n)\}$
- Salida: $g(\mathbf{x})$
- Algoritmo:

$g(\mathbf{x}) = 0$;
do
    $m = 0$; // número de muestras bien clasificadas
    for $(i = 1;\ i \leq n;\ i{+}{+})$
        if $c_i \cdot g(\mathbf{x}_i) \leq 0$ **then** // Si hay un error de clasificación
            $g(\mathbf{x}) = g(\mathbf{x}) + c_i\, K(\mathbf{x}_i, \mathbf{x}) + c_i$
        else
            $m = m + 1$
while $(m < n)$

Aprendizaje - Kernel Perceptron

Desde el punto de vista de los parámetros α:

- Entrada: $\{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), \cdots, (\mathbf{x}_n, c_n)\}$
- Salida: $\alpha \in \mathbb{R}^n$
- Algoritmo:

$\alpha = \mathbf{0}$;
do
    $m = 0$; // número de muestras bien clasificadas
    for $(i = 1; i \leq n; i{+}{+})$
        if $c_i \cdot g(\mathbf{x}_i) \leq 0$ **then** // Si hay un error de clasificación
            $\alpha_i = \alpha_i + 1$
        else
            $m = m + 1$
while $(m < n)$

Índice

Kernel Polinomial

- Kernel polinomial:

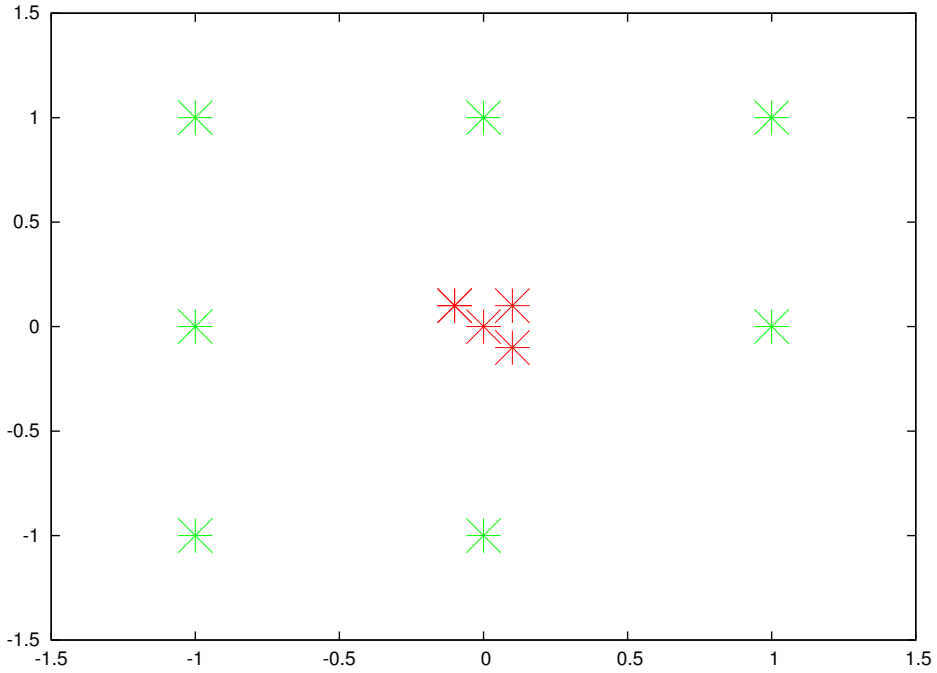
$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^t \cdot \mathbf{y} + c)^d$$

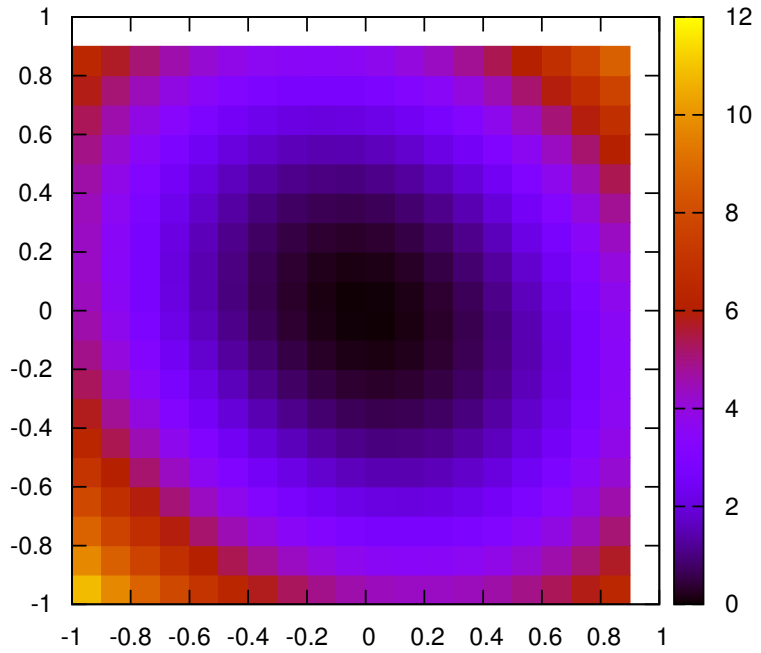
- Ejemplo $d = 2$

$$\begin{aligned}
K(\mathbf{x}, \mathbf{y}) &= \left(\begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} + c \right)^2 \\
&= (x_1\, y_1 + x_2\, y_2 + c)\,(x_1\, y_1 + x_2\, y_2 + c) \\
&= x_1^2\, y_1^2 + x_2^2\, y_2^2 + 2\, x_1\, y_1\, x_2\, y_2 + 2\, x_1\, y_1\, c + 2\, x_2\, y_2\, c + c^2 \\
&= \begin{bmatrix} x_1^2 & x_2^2 & \sqrt{2}\, x_1\, x_2 & \sqrt{2}\, x_1 & \sqrt{2}\, x_2 & c \end{bmatrix} \begin{bmatrix} y_1^2 \\ y_2^2 \\ \sqrt{2}\, y_1\, y_2 \\ \sqrt{2}\, y_1 \\ \sqrt{2}\, y_2 \\ c \end{bmatrix} \\
&= \Phi(\mathbf{x}) \cdot \Phi(\mathbf{y})
\end{aligned}$$

Kernel Polinomial

- Conjunto de entrenamiento (ver figura izquierda)
- Representación de $g(\mathbf{x})$ con $\mathbf{x} \in [-1, 1]$ y $\alpha_i = 1, \forall i$ (ver figura derecha)

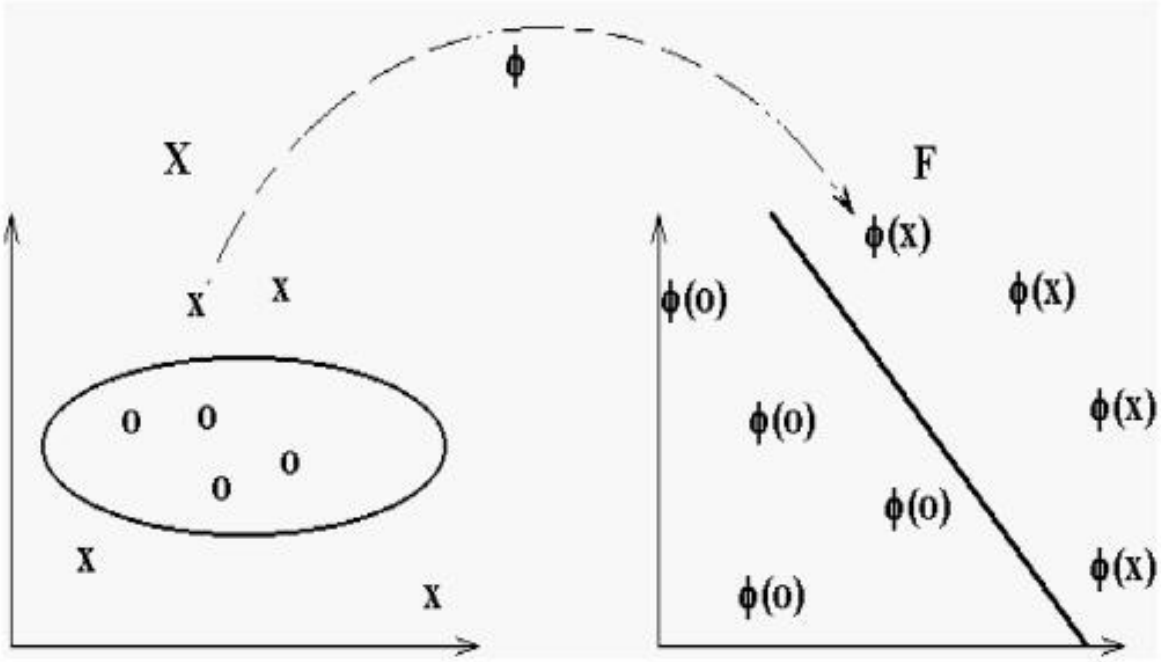




- Recordad $g(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \, c_i \, K(\mathbf{x}_i, \mathbf{x})$ siendo $K(\mathbf{x}_i, \mathbf{x})$ un kernel polinómico

Kernel Polinomial

- En el ejemplo previo hay una proyección implícita a un nuevo espacio donde las muestras de entrenamiento son linealmente separables:



http://upload.wikimedia.org/wikipedia/commons/b/b1/Svm_8_polinomial.JPG

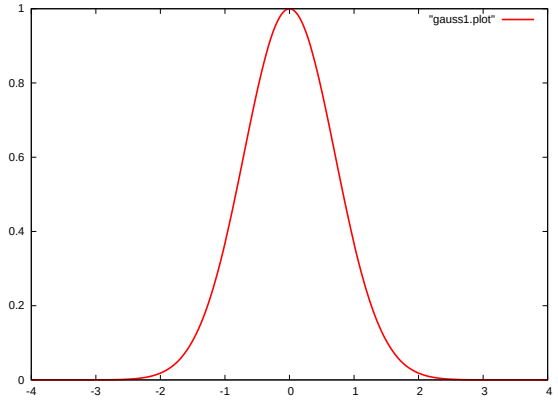
Índice

Kernel Gaussiano

- Kernel Gaussiano:

$$K(\mathbf{x}, \mathbf{y}) = \exp\left(\frac{- \mid\mid \mathbf{x} - \mathbf{y} \mid\mid^2}{2\sigma^2}\right)$$

- Representación gráfica de la gaussiana (unidimensional):



- Kernel muy empleado, pues asume que la función $\Phi(\cdot)$ implícitamente relacionada proyecta los puntos a un espacio de dimensionalidad infinita
- En dimensionalidad infinita los datos son ***siempre*** linealmente separables

Índice

Kernels Generalizados

- Objetivo: definir y evaluar diferentes funciones kernel
- La función kernel debe cumplir que:

$$\exists \Phi : \mathbb{R}^D \rightarrow \mathbb{R}^{D'} \colon K(\mathbf{x}, \mathbf{y}) = \Phi(\mathbf{x}^t)\, \Phi(\mathbf{y})$$

- Mercer condition: condición necesaria y suficiente para caracterizar que K sea un kernel válido:

  "La matriz Gramm $\mathbf{K}_{i,j}$ definida para el conjunto de entrenamiento $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ es semidefinida positiva $(\mathbf{z}^t \mathbf{K} \mathbf{z} \geq 0, \forall \mathbf{z} \in \mathbb{R}^{n \times 1}, \mathbf{z} \neq \mathbf{0})$"

- La matriz Gramm $\mathbf{K}$ es semidefinida positiva si se cumple:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} K(\mathbf{x}_i, \mathbf{x}_j)\, c_i\, c_j \geq 0 \quad \forall c_i, c_j \in \mathbb{R}$$

- Respetando esta propiedad podemos construir kernels desde kernels más simples

Kernels Generalizados

Si K_1 y K_2 son kernels, entonces K es un kernel:

$$K(\mathbf{x},\mathbf{y}) = \begin{cases} c \cdot K_1(\mathbf{x},\mathbf{y}) & c > 0 \\ f(\mathbf{x}) \cdot K_1(\mathbf{x},\mathbf{y}) \cdot f(\mathbf{y}) & \text{para cualquier función } f \\ q(K_1(\mathbf{x},\mathbf{y})) & q \text{ polinomio con coeficientes no negativos} \\ (c + K_1(\mathbf{x},\mathbf{y}))^d & d, c > 0 \\ K_1(\mathbf{x},\mathbf{y}) + K_2(\mathbf{x},\mathbf{y}) & \\ K_1(\mathbf{x},\mathbf{y}) \cdot K_2(\mathbf{x},\mathbf{y}) & \\ \exp\left(K_1(\mathbf{x},\mathbf{y})\right) & \\ \frac{K_1(\mathbf{x},\mathbf{y})}{\sqrt{K_2(\mathbf{x},\mathbf{y})}} & \end{cases}$$

Kernels Generalizados

- Sea $\mathcal{A}$ una matriz semidefinida positiva, entonces K es un kernel:

$$K(\mathbf{x}, \mathbf{y}) = \mathbf{x}^t \, \mathcal{A} \, \mathbf{y}$$

- Sean $\mathbf{x}, \mathbf{y} \in \mathbb{R}^D$, tales que $\mathbf{x} = (\mathbf{x}_a, \mathbf{x}_b)$, $\mathbf{y} = (\mathbf{y}_a, \mathbf{y}_b)$, con:
  - $\mathbf{x}_a, \mathbf{y}_a \in \mathbb{R}^{D_a}$
  - $\mathbf{x}_b, \mathbf{y}_b \in \mathbb{R}^{D_b}$
  - $D = D_a + D_b$

  Si K_a y K_b son kernels en $\mathbb{R}^{D_a}$ y $\mathbb{R}^{D_b}$, respectivamente, K es un kernel:

$$K(\mathbf{x}, \mathbf{y}) = \begin{cases} K_a(\mathbf{x}_a, \mathbf{y}_a) + K_b(\mathbf{x}_b, \mathbf{y}_b) \\ K_a(\mathbf{x}_a, \mathbf{y}_a) \cdot K_b(\mathbf{x}_b, \mathbf{y}_b) \end{cases}$$