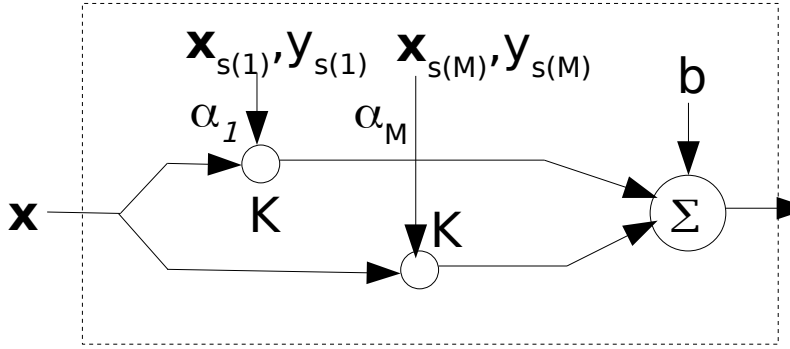
Tema 6. Máquinas de vectores de soporte (SVM)

SVM para clasificación (SVC) de 2 clases



$$y(\boldsymbol{x}) = sign\left\{\sum_{i=1}^{M} \alpha_i\, y_{s(i)}\, K[\boldsymbol{x}_{s(i)}, \boldsymbol{x}] + b\right\}$$

- Clasificador linear $y=\mathrm{sgn}(\mathbf{w}^T\mathbf{x}+b)$ no espazo oculto. Saída y=-1 para unha clase, y=+1 para a outra; $\mathbf{x}_{s(1)}...\mathbf{x}_{s(M)}$ son M vectores de soporte
- Parámetros entrenábeis: $\{\alpha_i\}_{i=1}^{M}$, b; hiper-parámetros sintonizábeis: regularización (λ), parámetro(s) do cerne (kernel) K
- A SVM combina dous ingredentes fundamentais:

1) Uso dun cerne K para aumentar a separabilidade linear dos datos
2) Selección do clasificador linear que minimiza a sobreaprendizaxe

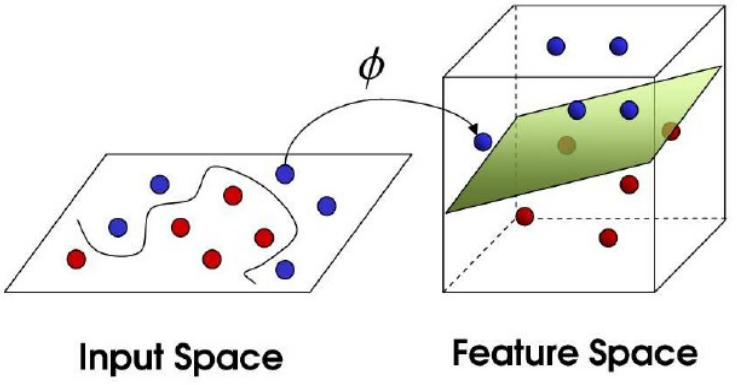
Cernes (I)

- Teorema de Cover (1965): a probabilidade de que N patróns no espazo n-dimensional sexan linearmente separábeis é 1 se n≥N-1 e:

$$\frac{1}{2^N}\sum_{i=1}^{n}\binom{N-1}{i}$$

  se n<N-1. Ésta é unha función crecente con n

- Aumentando a dimensionalidade n dos datos, aumenta a probabilidade de que sexan linearmente separábeis
- Aplicación cerne: mapea o espazo de entrada $\mathbb{R}^n$ a $\mathbb{R}^m$ con m>n: $\mathbf{x} \rightarrow \Phi(\mathbf{x})$
- A separabilidade linear é máis probábel neste segundo espazo, chamado espazo oculto ou de características



- Un cerne Φ usualmente cumpre que $\Phi(\mathbf{x})^T\Phi(\mathbf{y})=K(\mathbf{x},\mathbf{y})$: producto escalar xeralizado. Se existe expresión para $\Phi(\mathbf{x})$ en $\mathbb{R}^m$, o cerne é separábel

Cernes (II)

- K(**x**,**y**) é unha medida de similaridade xeralizada entre **x** e **y**
- A SVM é un clasificador linear de vector **w** no espazo oculto **Φ**(**x**)
- Veremos que: $\boldsymbol{w}=\sum_{i=1}^{M}\alpha_i\, y_{s(i)}\boldsymbol{\Phi}\left[\boldsymbol{x}_{s(i)}\right]$

  sendo M<N o nº de vectores de soporte $\{\mathbf{x}_{s(i)}\}_{i=1}^{M}$
- Polo tanto: $y=sign\left[\boldsymbol{w}^T\boldsymbol{\Phi}\left(\boldsymbol{x}\right)+b\right]=sign\left\{\sum_{i=1}^{M}\alpha_i\, y_{s(i)}\boldsymbol{\Phi}\left[\boldsymbol{x}_{s(i)}\right]^T\boldsymbol{\Phi}\left(\boldsymbol{x}\right)+b\right\}$

  $$y\left(\boldsymbol{x}\right)=sign\left\{\sum_{i=1}^{M}\alpha_i\, y_{s(i)}K\left[\boldsymbol{x}_{s(i)},\boldsymbol{x}\right]+b\right\}$$
- Hai varios cernes distintos, con hiper-parámetros que se deben sintonizar para cada problema
- Cerne **gausiano** ou RBF: hai que sintonizar a anchura σ: Valores: $\{2^i\}_{-5}^{10}$ (espazo oculto de dimensión infinita)

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

Cernes (III)

- Cerne polinómico $K(\mathbf{x},\mathbf{y})=(\mathbf{x}^T\mathbf{y}+a)^b$. Hiperparámetros:

1) a: sesgo, $-\mu \leq a \leq \mu$ se $\mathbf{x},\mathbf{y}$ están estandarizados: $|\mathbf{x}^T\mathbf{y}|<\mu$

2) b: grao=1,2,3

- Hai outros cernes, p.ex. para cadeas de caracteres
- Sen cerne, usas un cerne linear $K(\mathbf{x},\mathbf{y})= \mathbf{x}^T\mathbf{y}$, clasificador linear no espazo de entrada, sen hiper-parámetros
- O cerne que mellor funciona normalmente é o gausiano, que so ten un hiperparámetro: a anchura σ
- A calidade da SVM (acerto en clasificación, RMSE en regresión) mostra un extremo para o mellor valor de σ
- A SVM con cerne linear funciona bastante pior que a SVM gausiana, agás con n moi alto, porque a separabilidade linear xa é moi elevada no espazo de entrada

Minimización da sobreaprendizaxe

- Teoría da aprendizaxe estatística (V. Vapnik, anos 60)
- Dimensión de Vapnik-Chervonenkis (h) dun clasificador de 2 clases é o máximo número de patróns que pode aprender sen erros, sexa cal sexa o etiquetado de clases dos patróns
- Mide a complexidade do clasificador: canto mais alta sexa h, máis sobreaprendizaxe. Débese minimizar
- Para un **clasificador linear** nun espazo n-dimensional $\mathbb{R}^n$: h=n+1
- Se os patróns verifican $|\mathbf{x}|<D$ e ρ é o **marxe** (mínima distancia entre **x** e o hiperplano que separa entre clases), entón:

$$h \leq min\left(\left\lceil \frac{D}{\rho^2} \right\rceil, n\right) + 1$$ Para un clasificador linear

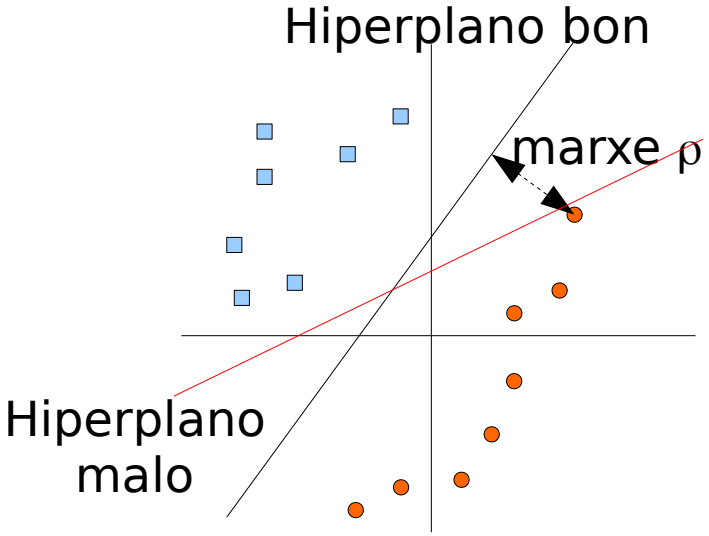
- Débese maximizar a marxe ρ para minimizar a cota superior de h, e polo tanto a sobreaprendizaxe (risco estructural)

Saída da SVM (I)

Distancia entre $\Phi(\mathbf{x}_i)$ e o plano

- A marxe ρ no espazo oculto é: $\rho = \min_{i=1\ldots N} \frac{|w^T \Phi(x_i) + b|}{|w|}$

- Imponse a condición $|\mathbf{w}^T\Phi(\mathbf{x}_i)+b|=1$, para o $\mathbf{x}_i$ máis cercano, así que $\rho=1/|\mathbf{w}|$



- O erro de entrenamento avalíase como $\xi_i=\max(0,1-y_i(\mathbf{w}^T\Phi(\mathbf{x}_i)+b))$ para $\Phi(\mathbf{x}_i)$

- $\xi_i>0 \leftrightarrow \Phi(\mathbf{x}_i)$ clasifícase mal ($\xi_i>1$) ou ben pero moi cerca de estar mal ($0<\xi_i<1$)

- Hai que maximizar (λ=hiperparámetro de regularización):

$$J(w,b,\vec{\xi}) = |w|^2 + \lambda \sum_{i=1}^{N} \xi_i$$

Erro de xeralización (risco estructural)

Erro de entrenamento (risco empírico)

coas condicións: $\xi_i \geq 0,\ y_i(w^T \vec{\Phi}(x_i)+b) \geq 1-\xi_i, i=1\ldots N$

Saída da SVM (II)

- Método dos multiplicadores de Lagrange: $\{\alpha_i,\beta_i\}_{i=1}^{N}$

$$L(\boldsymbol{w},b,\vec{\xi},\vec{\beta},\vec{\alpha})=|\boldsymbol{w}|^2+\lambda\sum_{i=1}^{N}\xi_i+\sum_{i=1}^{N}\beta_i\xi_i+\sum_{i=1}^{N}\alpha_i[y_i(\boldsymbol{w}^T\vec{\Phi}(\boldsymbol{x}_i)+b)-1+\xi_i]$$

- Derivando a respeito de **w** e igualando a **0**: $\boldsymbol{w}=\sum_{i=1}^{N}\alpha_i y_i \boldsymbol{\Phi}(\boldsymbol{x}_i)$
- Vector **w**: combinación linear dos patróns de entrenamento. Isto non é escalábel para N grandes
- Solución: como veremos, $\alpha_i=0$ para moitos i (solución dispersa)
- Derivando a respeito de b, ξ_i, α_i e β_i transfórmase o problema a maximizar $\vec{\alpha}^T\mathbf{1}-\frac{\vec{\alpha}\boldsymbol{K}\vec{\alpha}}{2}$, sendo $K_{ij}=K(\mathbf{x}_i^T,\mathbf{x}_j)$, coas condicións:

$$\vec{\alpha}^T\boldsymbol{y}=0, 0\le\alpha_i\le\lambda, \beta_i\xi_i=0, \alpha_i\left\{y_i\left[\sum_{j=1}^{N}\alpha_i K(\boldsymbol{x}_j^T,\boldsymbol{x})+b\right]\right\}=0, i=1,\dots N$$

Saída da SVM (III)

- Este problema de optimización resólvese usando cálculo numérico iterativo.
- Canto máis difícil é o problema, máis tempo tarda en resolverse a optimización. Pode ser máis rápida en problemas máis grandes
- Só se requiren M patróns de entrenamento (vectores de soporte) $\mathbf{x}_{s(1)}...\mathbf{x}_{s(M)}$, para os que $0 \leq \alpha_i \leq \lambda$, sendo $\alpha_i = 0$ para os restantes patróns (solución dispersa):

w no espazo oculto $\longrightarrow \boldsymbol{w} = \sum_{i=1}^{M} \alpha_i \, y_{s(i)} \boldsymbol{\Phi}\left[\boldsymbol{x}_{s(i)}\right]$

Cálculo do sesgo

- Sendo $(\mathbf{x}_m, y_m)$ un vector de soporte: $b = y_m - \sum_{i=1}^{M} \alpha_i \, y_{s(i)} K\left[\boldsymbol{x}_m, \boldsymbol{x}_{s(i)}\right]$
- Sustituíndo **w** en $z=\text{sign}(\mathbf{w}^T\Phi(\mathbf{x})+b)$ e usando que $\Phi(\mathbf{x})^T\Phi(\mathbf{y})=K(\mathbf{x},\mathbf{y})$ obtemos a expresión da SVM:

$$y(\boldsymbol{x}) = sign\left\{\sum_{i=1}^{M} \alpha_i \, y_{s(i)} K\left[\boldsymbol{x}_{s(i)}, \boldsymbol{x}\right] + b\right\}$$

Hiperparámetros e clasificación multi-clase (I)

- **Hiperparámetros sintonizábeis**, cerne gausiano, grid-search:

  λ (regularización): $2^{-5}..2^{15}$: os resultados non son moi sensíbeis ao seu valor, é pouco importante

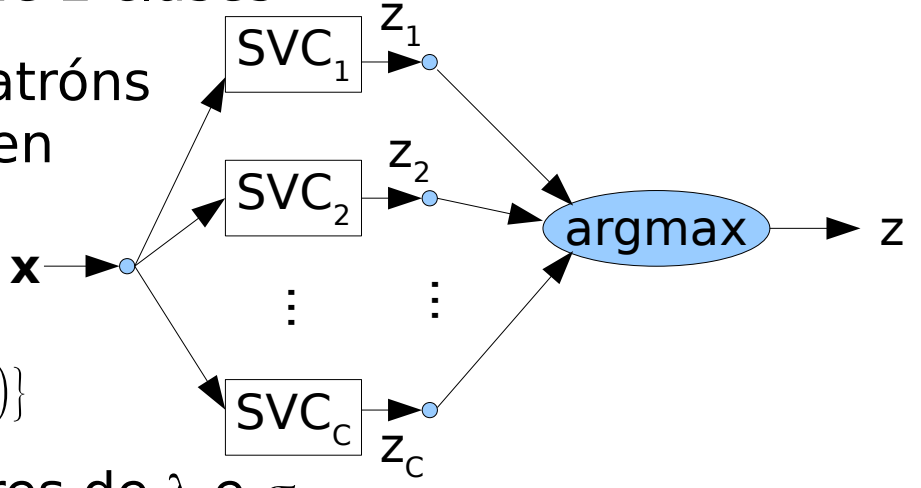
  σ (ancho do cerne gausiano): $2^{-3}..2^{7}$: moi importante nos resultados: o mellor valor está a miúdo no medio dos extremos

- **Clasificación multiclase**: se C>2 é elevado, usar o esquema one-vs-all (**OVA**), de O(C), onde a SVM i-ésima clasifica os patróns entre a clase i e as restantes, usa C SVMs de 2 clases

- A SVM i-ésima entrena con tódolos patróns (y_k=1 se $\mathbf{x}_k$ pertence á clase i, y_k=-1 en caso contrario)

$$z_i(\boldsymbol{x}) = \sum_{j=1}^{M} \alpha_j y_{s(j)} K[\boldsymbol{x}_{s(j)}, \boldsymbol{x}] + b$$

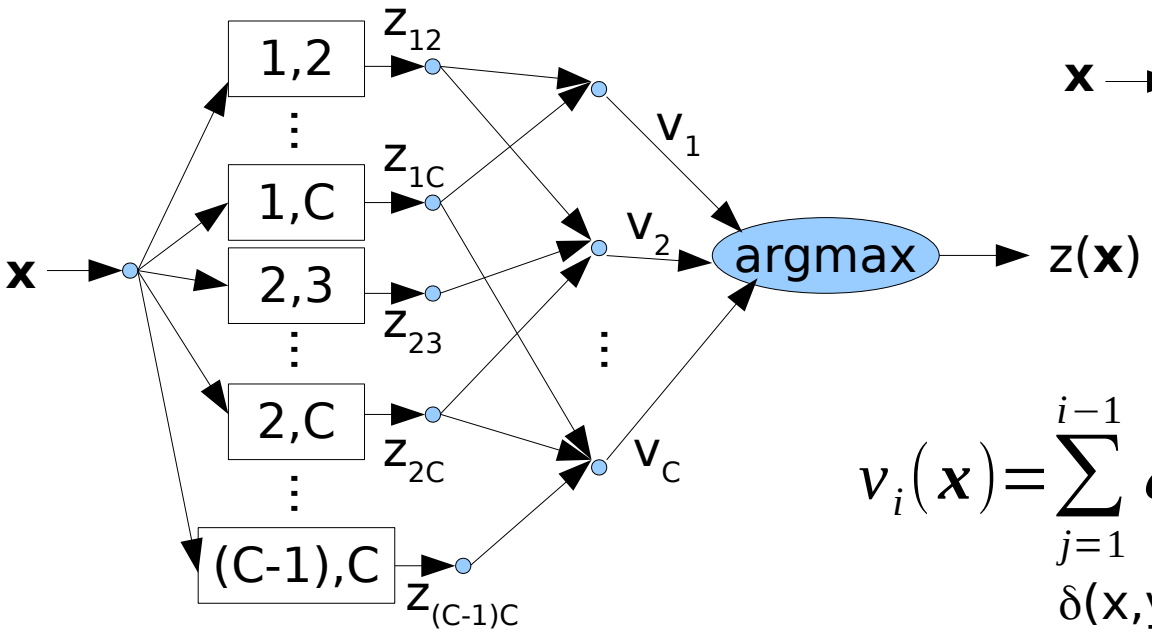
$$z(\boldsymbol{x}) = \underset{i=1\ldots C}{argmax} \{z_i(\boldsymbol{x})\}$$

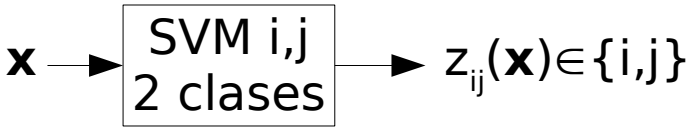


- Todas as SVMs usan os mesmos valores de λ e σ

Clasificación multi-clase (II)

- Con C>2 baixo, emprégase un esquema one-vs-one (**OVO**). Usa C(C-1)/2 SVMs de 2 clases. Acerta máis que OVA. É máis lento que OVA porque ten complexidade $O(C^2)$
- A SVM ij clasifica entre a clase i e a j, con i=1..C-1 e j=i+1..C, entrenando so cos patróns $\mathbf{x}_k$ destas clases (y_k=1 se $\mathbf{x}_k$ é da clase i e y_k=-1 se $\mathbf{x}_k$ é da clase j)





$$z(\boldsymbol{x}) = \underset{i=1\ldots C}{argmax} \{v_i(\boldsymbol{x})\}$$

$$v_i(\boldsymbol{x}) = \sum_{j=1}^{i-1} \delta[z_{ji}(\boldsymbol{x}), i] + \sum_{j=i+1}^{C} \delta[z_{ij}(\boldsymbol{x}), i]$$

$\delta(x,y)=1$ se $x=y$ e $\delta(x,y)=0$ se $x \neq y$

Regresión regularizada con cernes (kernel ridge regression)

- Sexa un cerne K: $\mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$, p. ex. gausiano: $K(\boldsymbol{x},\boldsymbol{y}) = \exp\left(\frac{-|\boldsymbol{x}-\boldsymbol{y}|^2}{2\sigma^2}\right)$
- Regresión non linear: modela a saída como:

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} w_i K(\boldsymbol{x},\boldsymbol{x}_i) = \boldsymbol{w}^T \boldsymbol{k}(\boldsymbol{x}) \qquad \boldsymbol{k}(\boldsymbol{x}) = [K(\boldsymbol{x},\boldsymbol{x}_1) \ldots K(\boldsymbol{x},\boldsymbol{x}_N)]^T$$

- A función K é un cerne (producto escalar xeralizado):

$$\boldsymbol{w} = \underset{\boldsymbol{v}}{argmin}\{J(\boldsymbol{v})\} \quad J(\boldsymbol{w}) = \sum_{i=1}^{N}\left[y_i - \sum_{j=1}^{N} w_i K(\boldsymbol{x}_i,\boldsymbol{x})\right]^2 + \lambda\left|f(\boldsymbol{x})^2\right|$$

- J(**w**) pode rescribirse como J(**w**)=(**y**-**Kw**)^T(**y**-**Kw**)+λ**w**^{T}**K**^{T}**w**, onde **K** é a matriz NxN de cernes con $K_{ij}=K(\mathbf{x}_i,\mathbf{x}_j)$. Derivando e igualando a **0**:

$$(\boldsymbol{K}^T\boldsymbol{K} + \lambda\boldsymbol{K}^T)\boldsymbol{w} = \boldsymbol{K}^T\boldsymbol{y} \rightarrow \boldsymbol{w} = (\boldsymbol{K} + \lambda\boldsymbol{I})^{-1}\boldsymbol{y}$$

I é a matriz identidade

- Saída para un patrón de teste **x**: $z = \boldsymbol{y}^T(\boldsymbol{K} + \lambda\boldsymbol{I})^{-1}\boldsymbol{k}(\boldsymbol{x})$

Support vector regression ε-SVR (I)

- O RMSE non é moi axeitado para regresión porque os artefactos (valores moi alonxados da media) dan diferencias cadráticas altas
- Úsase a función linear ε-insensitiva: $\xi_i = \max(0, |y_i - z(\mathbf{x}_i)| - \varepsilon)$
- Minimizar a función de custe: $J = \frac{|\boldsymbol{w}|^2}{2} + \lambda \sum_{i=1}^{N} (\xi_i + \xi_i')$

  coas condicións:

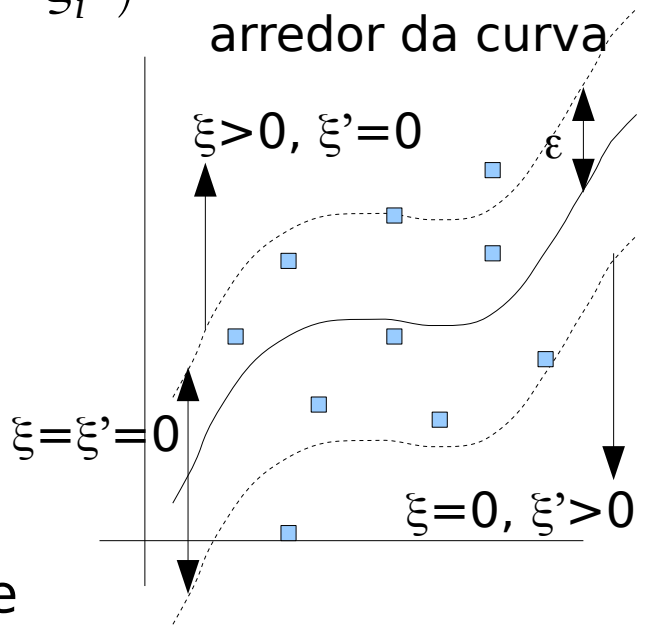
$$y_i - \boldsymbol{w}^T \boldsymbol{\Phi}(\boldsymbol{x}_i) - b \leq \varepsilon + \xi_i', i = 1 \ldots N$$

$$\boldsymbol{w}^T \boldsymbol{\Phi}(\boldsymbol{x}_i) + b - y_i \leq \varepsilon + \xi_i, i = 1 \ldots N$$

$$\xi_i \geq 0, \xi_i' \geq 0, i = 1 \ldots N$$

- Solución: $\boldsymbol{w} = \sum_{i=1}^{M} (\alpha_i - \alpha_i') \boldsymbol{\Phi}[\boldsymbol{x}_{s(i)}]$

  sendo α_i e α_i' os M multiplicadores de Lagrange asociados ás condicións e aos vectores de soporte $\mathbf{x}_{s(1)} .. \mathbf{x}_{s(M)}$



Support vector regression ε-SVR (II)

- Cálculo do sesgo: $b=\varepsilon-y_m+\sum_{i=1}^{M}\left(\alpha_i-\alpha_i'\right)K\left[\boldsymbol{x}_m,\boldsymbol{x}_{s(i)}\right]$

  onde $\mathbf{x}_m$ é un vector de soporte e y_m é a súa saída desexada
- Saída da ε-SVR: $z(\boldsymbol{x})=b+\sum_{i=1}^{M}\left(\alpha_i-\alpha_i'\right)K\left[\boldsymbol{x},\boldsymbol{x}_{s(i)}\right]$

- Hiperparámetros sintonizábeis con cerne gausiano: igual que en clasificación: regularización (λ) e ancho do cerne (σ)
- Os vectores de soporte $\mathbf{x}_i$ verifican $z(\mathbf{x}_i)\in[y_i-\varepsilon, y_i+\varepsilon]$, e $0\leq\alpha_i$, $\alpha_i'\leq\lambda$. Os restantes $\mathbf{x}_i$ verifican $\alpha_i=\alpha_i'=0$. O valor de ε non é sensíbel (ε=0.1 funciona ben)
- A ε-SVR é o mellor algoritmo de regresión que existe

Complexidade e implementacións

- O entrenamento da SVM é unha optimización cadrática con complexidade $O(N^3)$ e memoria $O(N^2)$; OVA é $O(C^2)$, OVO é $O(C)$
- Implementacións eficientes: $O(N^p)$ con $1 \leq p \leq 2.3$
- Lenta para N>10.000-50.000, dependendo do nº de entradas
- Con patróns moi anchos (n elevado), usa cerne linear porque non é necesario mapear a un espazo multi-dimensional

($\mathbf{x}_m$,y_m): vector de soporte

$$y(\boldsymbol{x}) = sign(\boldsymbol{w}^T \boldsymbol{x} + b), \boldsymbol{w} = \sum_{i=1}^{M} \alpha_i y_{s(i)} \boldsymbol{x}_{s(i)}, b = y_m - \sum_{i=1}^{M} \alpha_i y_{s(i)} \boldsymbol{x}_m^T \boldsymbol{x}_{s(i)}$$ Clasificación

$$y(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + b, \boldsymbol{w} = \sum_{i=1}^{M} (\alpha_i - \alpha_i') \boldsymbol{x}_{s(i)}, b = \varepsilon - y_m + \sum_{i=1}^{M} (\alpha_i - \alpha_i') \boldsymbol{x}_m^T \boldsymbol{x}_{s(i)}$$ Regresión

- Os cernes linear e gausiano dan calidades similares con n alto
- **LibSVM**: accesíbel dende C++, Octave/Matlab, Python, Weka/Java. Proporciona SVC multi-clase OVO, ε-SVR, e outras
- Función **ksvm** en paquete **kernlab** de R