risk function

$$R = \int L(d, h(x)) \, dP(x, d)$$

d: target; P: prob. distribution

approximate to

$$R_{emp} = \frac{1}{\ell} \sum_{1}^{\ell} (d_p - h(x_p))^2 \qquad \ell = \text{card}(TR)$$

VC-bounds: it holds with probability $1-\delta$

$$R \leq R_{emp} + \mathcal{E}\left(1/\ell, VC, 1/\delta\right)$$

VC-confidence

dim dataset

VC-dim: misura della complessità di H ⇒ flessibilità del modello

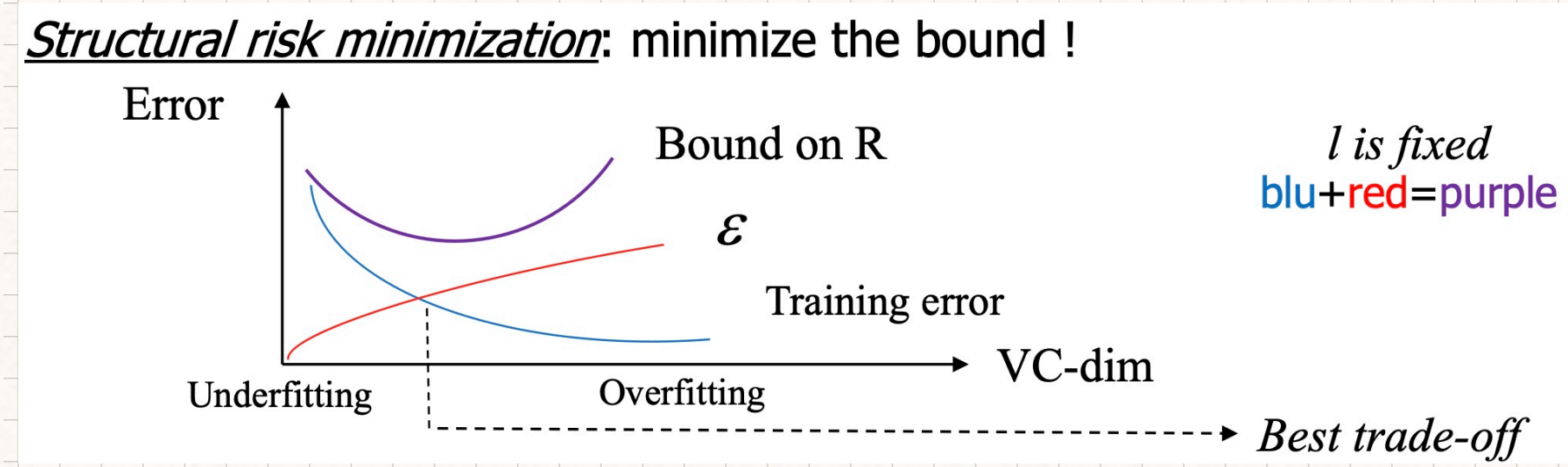
- ε is a function that grows with VC (VC-dim), that decreases with (higher) l and delta.
- We know that Remp decreases using complex models (with high VC-dim) (e.g. the

polynomial degree in the example)

- delta is the confidence, it rules the probability that the bound holds (e.g. low delta 0.01 → the bound holds with probability 0.99)

Se Vc aumenta, ma Remp diminisce ⇒ rischio overfitting

- Higher l (data) → lower VC confidence and bound close to R
- Too simple model (low VC-dim) can be not suff. due to high Remp (underfitting)
- Higher VC-dim (fix l) → lower Remp but VC-conf. and hence R may increase (overfitting)



Loss con Tikhonov

$$L(w) = \sum_{p=1}^{l} (y_p - x_p^T w)^2 + \lambda \|w\|^2 \rightarrow \sum_i w_i^2$$

– Small lambda value → minimizing the loss the focus is on obtaining a small error data term (first term, minimize just the training error) with a too complex model (high norm of the weights), the risk is of overfitting,

– High lambda value → minimize the loss the focus is on the second term, hence the data error (first term) could grow too much, i.e. moving to underfitting

Perceptron learning algorithm

1 Inizializzo i pesi con valori prossimi a zero

2 scelgo learning rate η

3 fino a condizione di stop non soddisfatta:

Delta rule $w_{new} = w + \frac{1}{2}\eta (d - out) x$ $\quad d = \{-1, +1\}$, $out = sign(w^T \cdot x)$

↳ vento 1/2

Perceptron Convergence Theorem

Th: For linearly separable tasks, the perceptron algorithm converges after a finite number of steps.

idea: find lower & upper bound to $|w|^2$ as a function of q^2 steps ✓ and q steps ✓ (lower b. / upper b.)

assume $w(0) = 0$, $\eta = 1$, q number of patterns misclassified (all false negatives)

- lower bound on $|w(q)|^2$: $w^* \cdot w(q) = w^* \cdot \sum_{j=1..q} x_{ij} \geq q\delta$ $\quad \delta = \min_i (w^* x_i)$

via Swartz

$$|w^*|^2 \cdot |w(q)|^2 \geq |w^* \cdot w(q)|^2 \geq |q\delta|^2 \geq (q\delta)^2 / |w^*|^2$$

- upper bound

$$|w(q)|^2 = |w(q-1) + x_{iq}|^2 = |w(q-1)|^2 + 2 w(q-1) x_{iq} + |x_{iq}|^2$$

$\|w(q)\|^2 \leq \|w(q-1)\|^2 + |x_{iq}|^2$ ≤ 0 alla q^{th} iterazione

iterando

$$\|w(q)\|^2 < \sum |x_{ij}|^2 \leq q\beta \qquad \beta = \max_i |x_i|^2$$

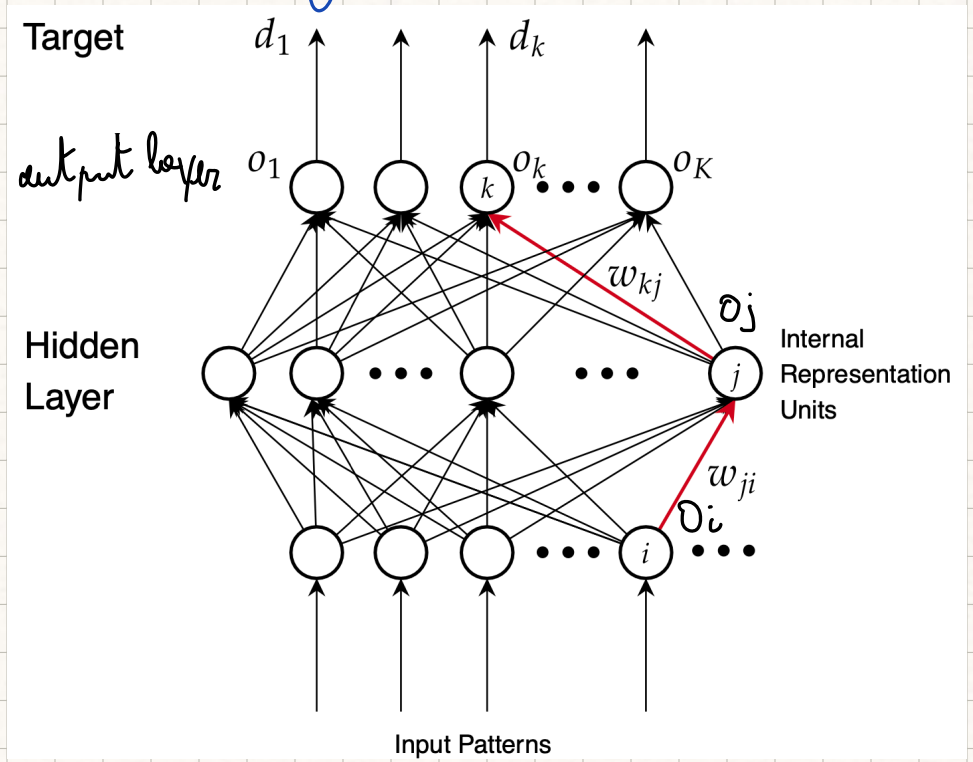
$$q\beta \geq \|w(q)\|^2 \geq (q\alpha)^2 / \|w^*\|^2$$

$$q\beta \geq q^2\alpha^2$$

$$\beta \geq q\alpha^2$$

$$q \leq \beta/\alpha^2$$

Backpropagation algorithm



- initialize all the weights w in the network and η
- while $E_{TOT} > \varepsilon$:

$\forall w \in \mathbf{W} : \Delta w = \frac{\partial E_{TOT}}{\partial w}$

$w_i^{new} = w_i + \eta \Delta w_i + \dots$

Compute E_{TOT} and out

$$E_{TOT} = \sum_p E_p, \quad E_p = \frac{1}{2} \sum_k (d_k - O_k)^2$$

$$\Delta w = -\frac{\partial E_{TOT}}{\partial w} = -\sum_p \frac{\partial E_p}{\partial w} = \sum_p \Delta_p w$$

$$\Delta_p w_{tu} = -\frac{\partial E_p}{\partial w_{tu}} = \left[-\frac{\partial E_p}{\partial net_t}\right] \cdot \left[\frac{\partial net_t}{\partial w_{tu}}\right] = \delta_t \cdot O_u$$

$$\begin{cases} net_t = \sum_j w_{tj} O_j \\ O_t = f_t(net_t) \end{cases} \Rightarrow \frac{\partial net_t}{\partial w_{tu}} = \frac{\partial \sum_j w_{tj} O_j}{\partial w_{tu}} = O_u \quad (0 \text{ per } j \neq u)$$

$$\delta_t = -\frac{\partial E_p}{\partial net_t} = \left[-\frac{\partial E_p}{\partial O_t}\right] \cdot \frac{\partial O_t}{\partial net_t} \overset{O_t = f_t(net_t)}{=} \left[-\frac{\partial E_p}{\partial O_t}\right] \cdot f_t'(net_t)$$

distinguere tra O_k che appartiene all'output layer o hidden layer

$t = k$ (output level)

$$-\frac{\partial E_p}{\partial O_k} = -\frac{\partial \frac{1}{2}\sum_{z=1}^{K}(d_z - O_z)^2}{\partial O_k} = (d_k - O_k) \qquad \delta_k = (d_k - O_k) f_k'(net_k)$$

$t = j$ (hidden unit)

$$-\frac{\partial E_p}{\partial O_j} = \sum_{k=1}^{K} -\frac{\partial E_p}{\partial net_k} \cdot \frac{\partial net_k}{\partial O_j} = \sum_{k=1}^{K} \delta_k \cdot w_{kj}$$

$$\frac{\partial \sum_s w_{ks} \cdot O_s}{\partial O_j} = w_{kj}$$

$$\delta_j = \left[\sum_{k=1}^{K} \delta_k \cdot w_{kj}\right] \cdot f_j'(net_j)$$

$$w_{tm}^{new} = w_{tm} + \eta \cdot \delta_t \cdot O_m$$

Momentum

$$\Delta w_{new} = -\eta \frac{\partial E(w)}{\partial w} + \alpha \Delta w_{old}$$

$$w_{new} = w_{old} + \Delta w_{new}$$

Nesterov Momentum

1. $\tilde{w} = w + \alpha \Delta w_{old}$
2. evaluate the new gradient at this interim point ($\tilde{w}$)

$$\Delta \tilde{w} = -\eta \frac{\partial E(\tilde{w})}{\partial w} + \alpha \Delta w_{old}$$

3. $w_{new} = w + \Delta \tilde{w}$

• Shown to improve the rate of convergence for the batch mode (not for the stochastic case!)

merging Weight decay and Momentum

$$\Delta w_{tu} = -\eta \frac{\partial Loss(w)}{\partial w_{tu}} + \alpha \Delta w_{old\,tu} = \eta \delta_t o_u - \lambda w_{tu} + \alpha \Delta w_{old\,tu}$$

Possiamo tenere staccati i iperparametri scrivendo

$$\Delta w_{tu} = \eta \delta_t o_u + \alpha \Delta w_{old\ tu}$$

$$w_{new\ tu} = w_{tu} + \Delta w_{tu} - \lambda w_{tu}$$

The Cascade Correlation algorithm

• Start with N0 network, a network without hidden units: Training of N0. Compute error for N0. If N0 cannot solve the problem: go to N1.

• In the N1 network a hidden unit is added such that the correlation between the output of the unit and the residual error of network N0 is maximized (by training its weights, see next slide).

• After training, the weights of the new unit are frozen (they cannot be retrained in the next steps) and the remaining weights (output layer) are retrained.

• If the obtained network N1 cannot solve the problem, new hidden units are progressively added which are connected with all the inputs and previously installed hidden units.

• The process continues until the residual errors of the output layer satisfy a specified stopping criteria (e.g the errors are below a given threshold).

$$S_k = \left| \sum_p (O_p - \bar{O})(\sigma_{p,k} - \bar{\sigma}) \right| \qquad O_p = f(net_p)$$

$$\frac{\partial |S_k|}{\partial w_j} = sign(S_k) \frac{\partial S_k}{\partial w_j} = sign(S_k) \sum_p (\sigma_{p,k} - \bar{\sigma}) \frac{\partial}{\partial w_j}(O_p - \bar{O})$$

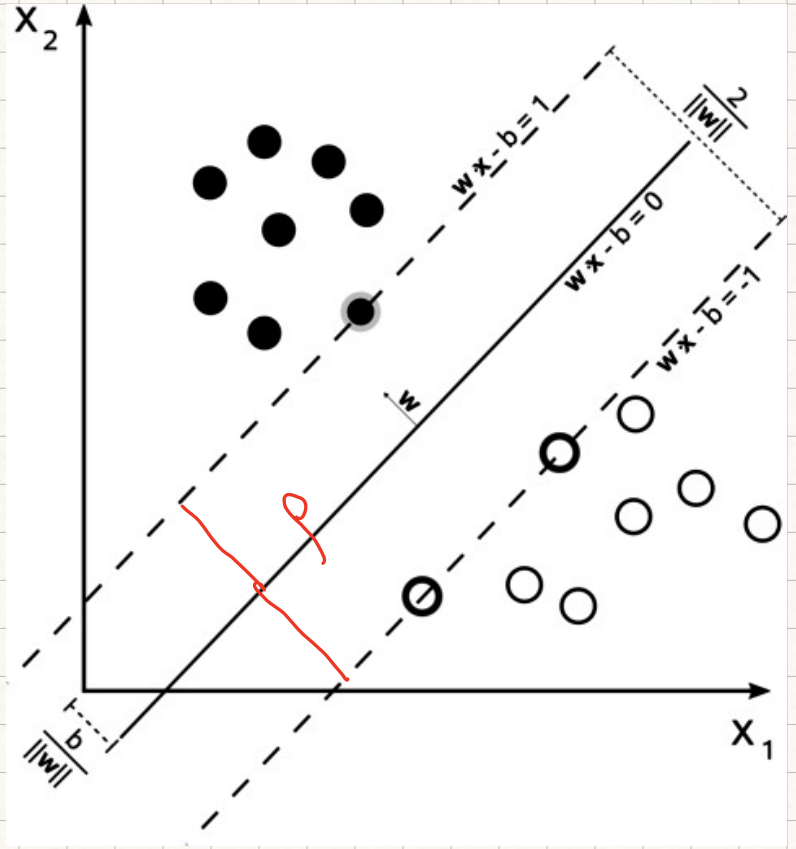
$$\frac{\partial}{\partial w_j} O_p = \frac{\partial}{\partial w_j} f(net_p) = \frac{\partial f}{\partial net_p} \cdot \frac{\partial net_p}{\partial w_j} = f'(net_p) \cdot x_{pj}$$

$$\text{sign}(\delta_k) \sum_p (E_{p,k} - \bar{E})\, f'(\text{net}_p)\, x_{pj}$$

• Definition: The VC dimension of a class of functions H is the maximum cardinality of a set of points (configuration) in X that can be shattered by H

$$R \leq R_{emp} + \varepsilon(1/\ell, V_c, 1/\delta)$$

SVM



Dato un punto x nello spazio: $x = x_p + r\frac{w_0}{\|w_0\|}$

proiezione punto su una retta

dist. origine - iperpiano

$g(x) = w_0^T x + b_0$

$$g\left(x_p + r\frac{w_0}{\|w_0\|}\right) = w_0^T\left(x_p + r\frac{w_0}{\|w_0\|}\right) + b_0 = w_0^T x_p + w_0^T r\frac{w_0}{\|w_0\|} + b_0 =$$

$$\underbrace{w_0^T x_p + b_0}_{g(x_p)=0} + r\|w_0\| \Rightarrow g(x) = r\|w_0\|$$

Ne pouze $g(x) = 1$

$$r = \frac{1}{\|w_0\|} = \frac{\rho}{2} \qquad \rho = \frac{2}{\|w_0\|}$$

pripom $w_0^T x + b_0 = 0 \Rightarrow$ maximizace $\rho = \frac{2}{\|w_0\|} \Rightarrow$ minimizace $\|w_0\|$

Hard margin SVM primal form

$T = \{x_i, y_i\}_1^N$

$$\min_{w,b} f(w) = \frac{1}{2} w^T w \;\; (\to \|w\|^2) \qquad y_i(w^T x_i + b) \geq 1 \qquad i = 1, \dots, N$$

Rilassamento Lagrangiano

Condizioni necessarie di Karush-Kuhn-Tucker

$$\nabla_x L(x^*, \lambda^*, \mu^*) = 0$$

$$g(x^*) \leq 0, \quad h(x^*) = 0$$

$$\lambda^{*T} g(x^*) = 0$$

$$\lambda^* \geq 0.$$

$\lambda = \alpha$

Duale Lagrangiano svolto

$$\min_{\lambda \geq 0} \frac{1}{2} \sum_i \sum_j \lambda_i \lambda_j y_i y_j x_i^T x_j - \sum_i \lambda_i$$

$$\sum_i \lambda_i y_i = 0 \qquad \lambda_i \geq 0 \quad \forall i$$

risolvendo il duale Lagrangiano

$$\mathcal{L}(\lambda^*, w^*, b^*) = \frac{1}{2} w^T w - \sum_1^N \alpha_i (1 - y_i (w^T x_i + b_i))$$

$$w^* = \sum_1^N \alpha_i y_i x_i$$

$\alpha > 0$, allora x_i vettore di supporto

x_i non è vettore di supporto, allora $\alpha = 0$

Alla fine basta trovare i λ per risolvere il problema

$$b^* = 1 - w^* x^{(s)} \qquad b^* = 1 - \sum_1^N \alpha_i y_i x_i x^{(s)}$$

Iperpiano di separazione ottimo: $w^{*T} x + b^* = 0 \iff \sum_1^N \alpha^* y_i x_i^T x + b^* = 0$

la soluzione dipende interamente dai vettori di supporto

Theorem - Vapnik

Let D be the diameter of the smallest ball around the data points $\mathbf{x_1}, \cdots, \mathbf{x_N}$. For the class of separating hyperplanes described by the equation $\mathbf{w}^T\mathbf{x} + b = 0$ the upper bound to the VC dimension is

dimensione dell'input space

$$VC \leq min(\lceil \frac{D^2}{\rho^2} \rceil, m_0) + 1$$

since $\frac{D^2}{\rho^2} = Radios^2 \|\mathbf{w}\|^2$, VC can be less than $m_0 + 1$ computed for general hyperplanes by constraining the search to the "regularized" one with maximum margin

Soft margin primal problem

$$\min_{w,b,\xi} f(w,\xi) = \frac{1}{2} w^T w + C \sum_1^N \xi_i$$

$$y_i (w^T x_i + b) \geq 1 - \xi_i \qquad \forall i = 1, \ldots, N$$

$$\xi_i \geq 0 \qquad \forall i = 1, \ldots N$$

- Low C → many TR errors allowed → possible underfitting
- High C → no TR errors allowed → smaller margin → possible overfitting

Lagrange dual:

$$\max_{\lambda\geq 0,\mu\geq 0}\min_{w,b,\xi}\frac{1}{2}w^Tw + C\sum_i \xi_i + \sum_i \lambda_i(1-\xi_i - y_i(w^Tx_i+b)) - \sum_i \mu_i\xi_i$$

Dual Problem

Given the training set $T = \{(\Phi(\mathbf{x_i}), y_i)\}_{i=1}^N$ find the optimal values of $\{\lambda_i\}_{i=1}^N$ which maximize the objective function

$$Q(\lambda) = \sum_{i=1}^N \lambda_i - \frac{1}{2}\sum_{i,j=1}^N \lambda_i\lambda_j y_i y_j \overbrace{k(\mathbf{x}_i,\mathbf{x}_j)}^{K_{i,j}}$$

satisfying the constraints

$$\sum_{i=1}^N \lambda_i d_i = 0$$

$$0 \leq \lambda_i \leq C \;\forall i = 1,\cdots,N$$

scelgo C, e la funzione Kernel. Calcolo la matrice Kernel K. Risolvo il problema duale e calcolo λ_i. Calcolo b (bias) usando K e λ_i.

Dato un nuovo input x calcolo $\Rightarrow w^T\phi(x) = \sum_1^{\tilde{N}} \lambda_i y_i K(x, x_i)$ e calcolo il segno di $w^T\phi(x)$

Polynomial learning machine $k(x, x_i) = (x^T x_i + 1)^p$

Radial Basis Function Net: $k(x, x_i) = e^{-\frac{1}{2\sigma^2} \|x - x_i\|^2}$

Two layer perceptron $k(x, x_i) = \tanh(\beta_0 x^T x_i + \beta_1)$ where $\beta_0 > 0$ and $\beta_1 < 0$

SVM per la regressione

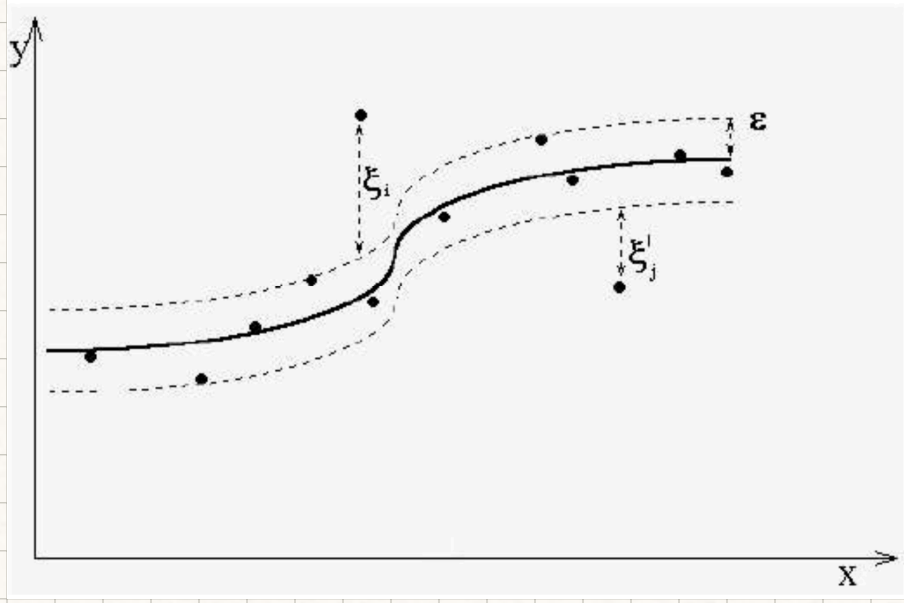
Primal Problem

Given the training set $T = \{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$ find the optimal values of $\mathbf{w}$ such that the following objective function is minimized

$$\Psi(\mathbf{w}, \xi, \xi') = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}(\xi_i + \xi_i')$$

under the constraints

$$\left.\begin{array}{l} d_i - \mathbf{w}^T\Phi(\mathbf{x}_i) \leq \epsilon + \xi_i \\ \mathbf{w}^T\Phi(\mathbf{x}_i) - d_i \leq \epsilon + \xi_i' \\ \xi_i \geq 0 \\ \xi_i' \geq 0 \end{array}\right\} \forall i = 1, \cdots, N$$



$w = \sum_{i}^{N} \underbrace{(\lambda_i - \lambda_i')}_{\gamma_i} \phi(x_i)$ formulazione duale

Dato un record di input x noi stimiamo il valore di $f(x)$ con

$$h(x) = \sum_{i}^{N} \gamma_i K(x_i, x)$$

Riassumendo

1. Important: we firstly have to select values for the user specified parameters C and ϵ
2. Important: we also have to choose an inner product kernel function k
3. We compute the kernel matrix K
4. We solve the optimization problem in the dual form and get the optimal values of the Lagrangian multipliers (we get the values of $\{\gamma_i\}_{i=1}^{N}$)
5. We compute the optimal value for the bias
6. We obtain the estimate function as a linear combiner of dot products in a feature space we can ignore –> $h(\mathbf{x}) = \sum_{i=1}^{N} \gamma_i k(\mathbf{x}_i, \mathbf{x})$

Bias variance decomposition

- Expected error or mean of z : $\bar{z} = E_p[z] = \sum_1^l z_i \, P(z_i)$
- Variance of z : $Var[z] = E[(z - \bar{z})^2] = E[z^2] - \bar{z}^2 \Rightarrow E[z^2] = Var[z] + \bar{z}^2$

$$E_p[(y - h(x))^2] = E_p[h(x)^2 - 2 y h(x) + y^2] = E_p[h(x)^2] + E_p[y^2] - 2 E_p[y] E_p[h(x)]$$

$$E_p[h(x)^2] = E_p[(h(x) - \bar{h}(x))^2] + \bar{h}(x)^2$$

$$E_p[y] = \bar{y} = E_p[f(x) + \varepsilon] = f(x) \Rightarrow E_p[y^2] = E[(y - f(x))^2] + f(x)^2$$

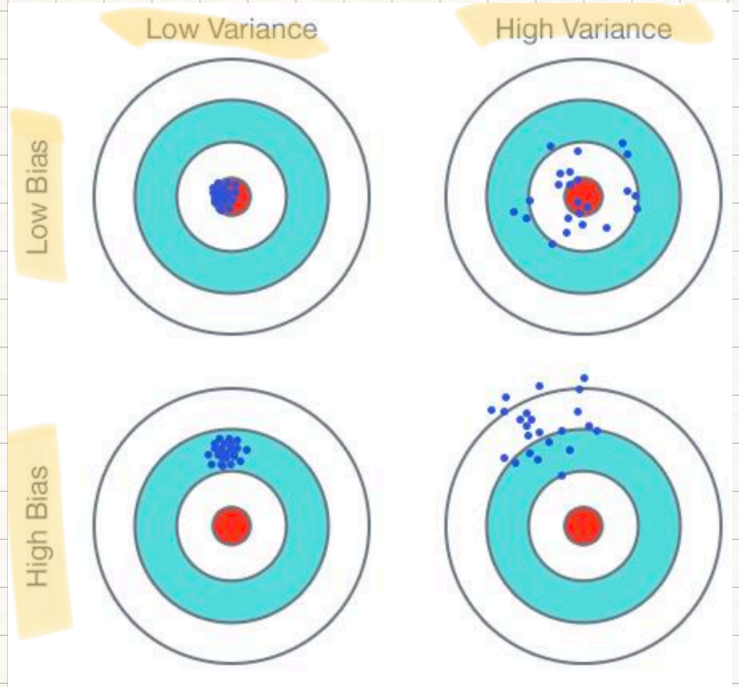
(unica realizzazione)

$$E_p[(y - h(x))^2] = E_p[(h(x) - \bar{h}(x))^2] + \bar{h}(x)^2 - 2 f(x) \bar{h}(x) + E_p[(y - f(x))^2] + f(x)^2$$

$$= E_p[(h(x) - \bar{h}(x))^2] + (\bar{h}(x) - f(x))^2 + E_p[(y - f(x))^2]$$

Variance + Bias^2 + E[noise^2]

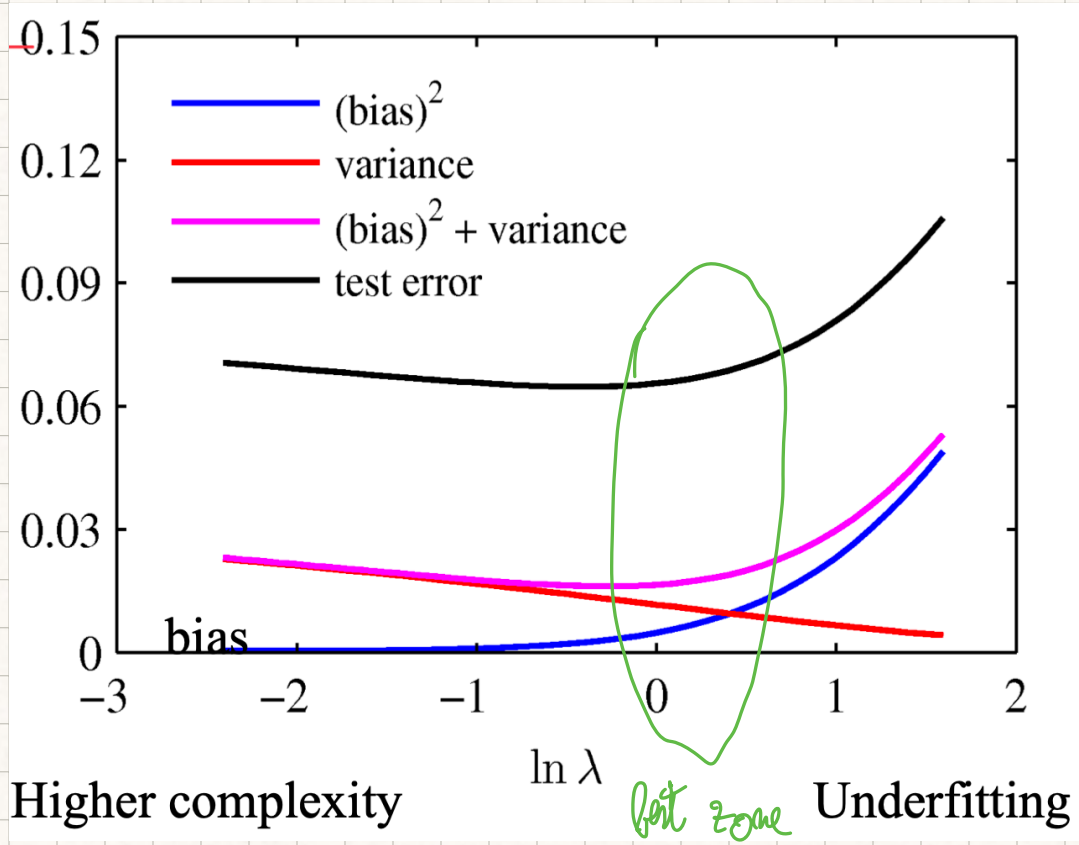
$Var[h(x)] + Bias[h(x)]^2 + \sigma^2$



Nel caso di regolarizzazione (weight decay)

all'aumentare di λ diminuisce la varianza e aumenta il bias.

Al diminuire di λ aumenta la varianza del modello predittivo e diminuisce il bias



SOM & k-means

Vector quantization

a submanifold $V \subseteq \mathbb{R}^m$, encoded by reference vectors $w = (w_1, \dots, w_K)$, $w_i \in \mathbb{R}^m$ $i = 1 \dots K$

subdivisible $V \Rightarrow$ $V_i = \{ x \in V \text{ s.t. } \|x - w_i\| \leq \|x - w_j\| \ \forall j \}$

↓

Voronoi polyhedra

each data vector x is described by the corresponding reference vector w_i

Quantization error function

$$E = \int f(\mathrm{d}(\boldsymbol{x}, \boldsymbol{w}_{i^*(x)}))\, p(\boldsymbol{x})\, d\boldsymbol{x} = \int \left\| \boldsymbol{x} - \boldsymbol{w}_{i^*(x)} \right\|^2 p(\boldsymbol{x})\, d\boldsymbol{x}$$

probability distribution

(squared error criterion)

$$E = \sum_i^l \sum_j^K \left\| \boldsymbol{x}_i - \boldsymbol{w}_j \right\|^2 \delta_{winner}(i, j)$$ Discrete version

$\delta_{winner}(i, j)$ characteristic function of the receptive field of $\boldsymbol{w}_j$
1 if j is the winner for $\boldsymbol{x}_i$, 0 otherwise

Lloyd and MacQueen's well-known K-MEANS clustering algorithm for any x_i

$$\frac{\partial E}{\partial w_{i^*}} = \Delta \boldsymbol{w}_{i^*} = \eta \delta_{winner}(i, i^*)(\boldsymbol{x}_i - \boldsymbol{w}_{i^*}) \cdot 2$$

Learning rate

- The set of reference vectors w that minimizes E is the solution of the VQ problem

SOM Competitive stage

- The winner is $i^*(x) = \arg\min_i \|x - w_i\|$

w_i: weight of unit i

i: index of the unit on the map

The current input vector x is compared with all the unit weights using an Euclidean distance criteria.

SOM Cooperative stage

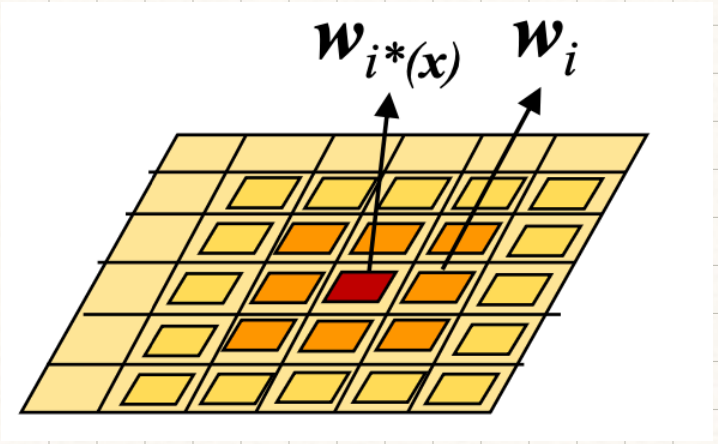
$$w_i(t+1) = w_i(t) + \eta(t)\, h_{i,i^*(x)}(t)\,[x - w_i(t)]$$

The weight of the winning unit and the weight of the units in its neighborhood are moved closer to the input vector (Hebbian learning). At iteration t:

$$h_{i,i^*}(t) = \exp\left(-\frac{\|r_i - r_{i^*}\|^2}{2\sigma_{nh}^2(t)}\right)$$

r_i: coordinates of unit i

$\sigma_{nh}(t)$: width



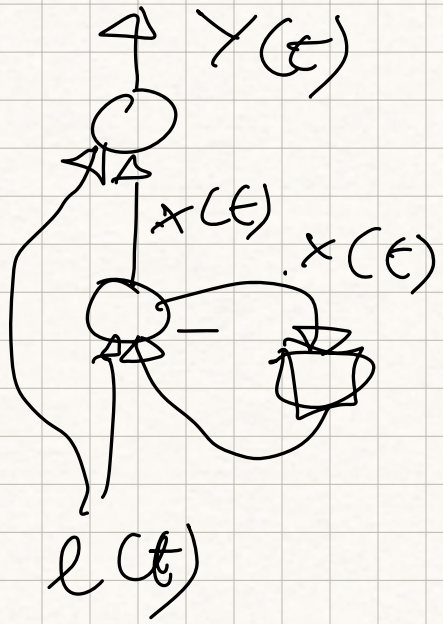
hi,j(t) is a function that decreases monotonically for increasing ||i-j||.

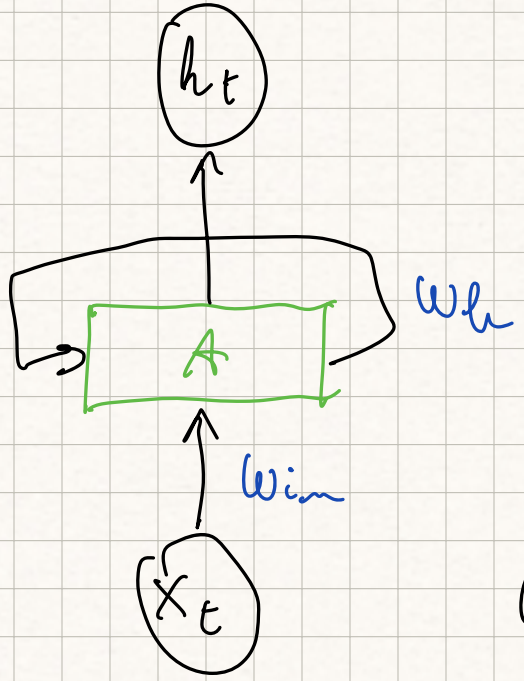
- Learning rate (η) and neighbourhood radius values (σ) are decreased as function of iteration t

RNN

State transition system

$$\begin{cases} x(t) = \tau(l(t), x(t-1)) \\ y(t) = g(x(t), l(t)) \end{cases}$$ ← modello RNN





$$f_t(h_{t-1}, x_t) = W_h h_{t-1} + W_{in} x_t + b_h$$

$$h_t = \tanh(f_t)$$

$$y_t = f(W_{out} h_t + b_{out})$$