Université de Montréal
Département d'informatique et de recherche opérationnelle

IFT3395/6390
Fondements de l'apprentissage machine

Régression linéaire
et
Régression linéaire régularisée

Professeur: Pascal Vincent

LISA Laboratoire d'Informatique des Systèmes d'Apprentissage

Tâche supervisée

prédire t à partir de x

input $\mathbf{x} \in \mathbb{R}^d$ — target $\mathbf{t}$

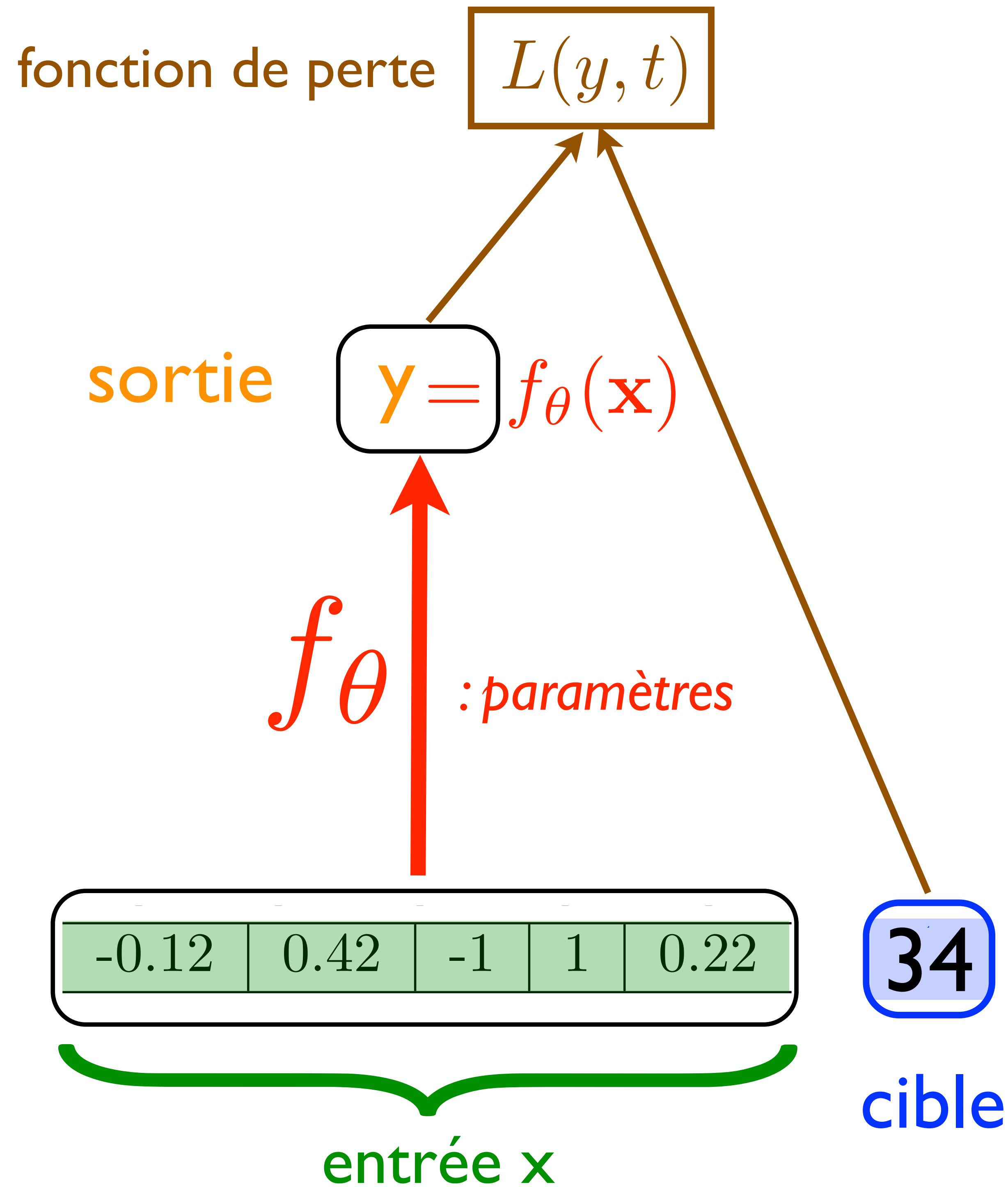
n examples

$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$	t
0.32	-0.27	+1	0	0.82	113
-0.12	0.42	-1	1	0.22	34
0.06	0.35	-1	1	-0.37	56
0.91	-0.72	+1	0	-0.63	77
...	...	...	...	...	...

Ensemble d'entraînement

$$D_n = \{(x^{(1)}, t^{(1)}), \ldots, (x^{(n)}, t^{(n)})\}$$

Apprendre une fonction f_θ qui minimise une perte (coût).



Minimization du risque empirique

On doit spécifier:

- Une forme paramétrée pour la fonction f_θ
- Une fonction de coût (ou perte) spécifique $L(y, t)$

On définit alors le risque empirique comme:

$$\hat{R}(f_\theta, D_n) \quad = \quad \sum_{i=1}^{n} L(f_\theta(\mathbf{x}^{(i)}), t^{(i)})$$

c.a.d. perte totale sur l'ensemble d'entraînement

L'apprentissage revient à trouver la valeur optimale des paramètres:

$$\theta^\star \quad = \quad \arg\min_\theta \hat{R}(f_\theta, D_n)$$

C'est le principe de minimisation du risque empirique

Ex: Régression linéaire

Un algorithme d'apprentissage très simple

On choisit

Une forme **lineaire** (affine) pour la fonction:

$$f_\theta(\mathbf{x}) = \underbrace{\langle \mathbf{w}, \mathbf{x} \rangle}_{\text{produit scalaire}} + b \qquad \theta = \{\mathbf{w}, b\}, \underbrace{\mathbf{w}}_{\text{vecteur de poids}} \in \mathbb{R}^d, \underbrace{b}_{\text{bias}} \in \mathbb{R}$$

Coût: erreur quadratique:

$$L(y, t) = (y - t)^2$$

Principe de minimisation du risque empirique:

On cherche les paramètres qui minimisent le risque empirique

$$\theta^\star = \arg\min_\theta \hat{R}(f_\theta, D_n)$$

Régression linéaire

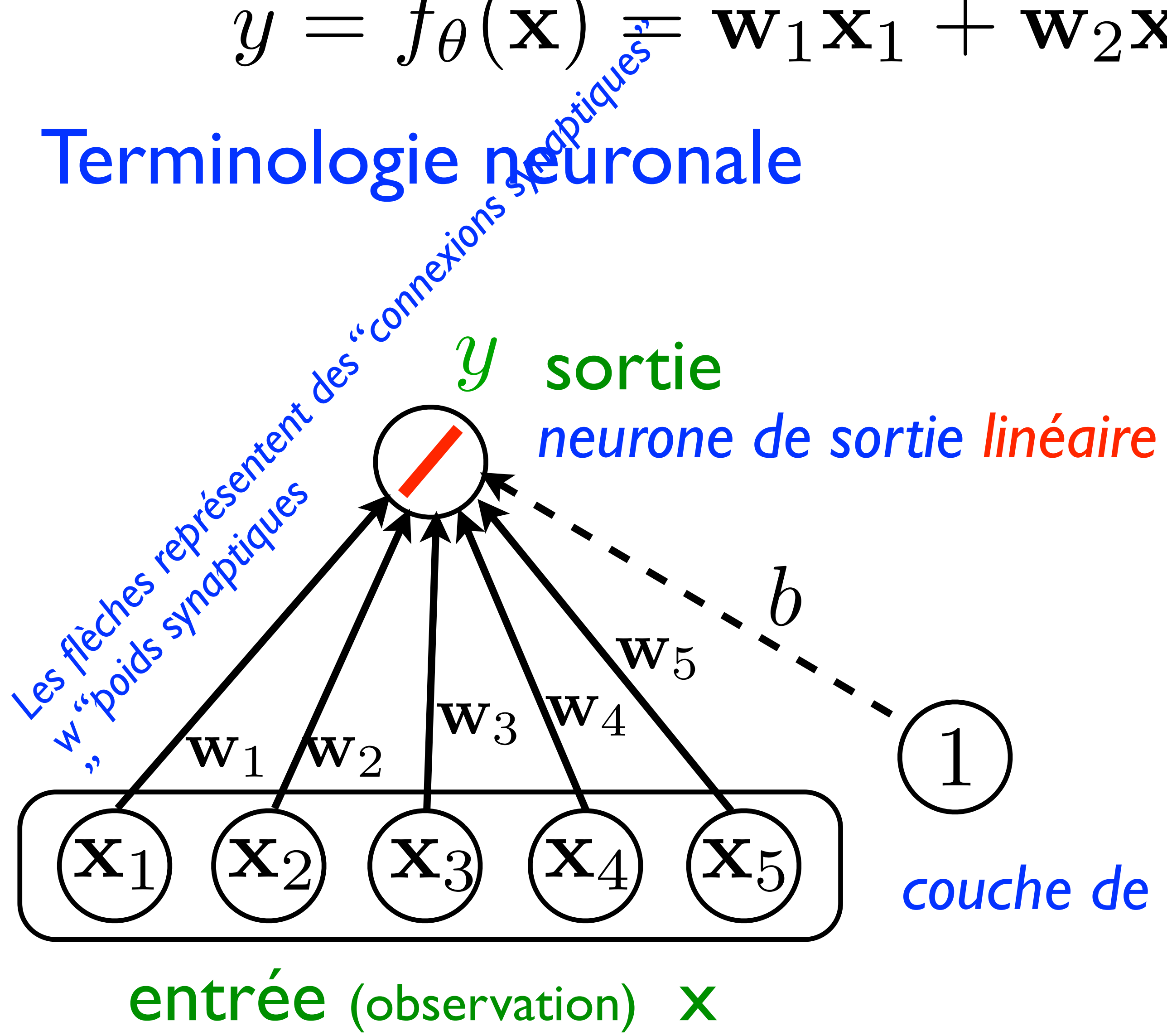
Vision neuronale

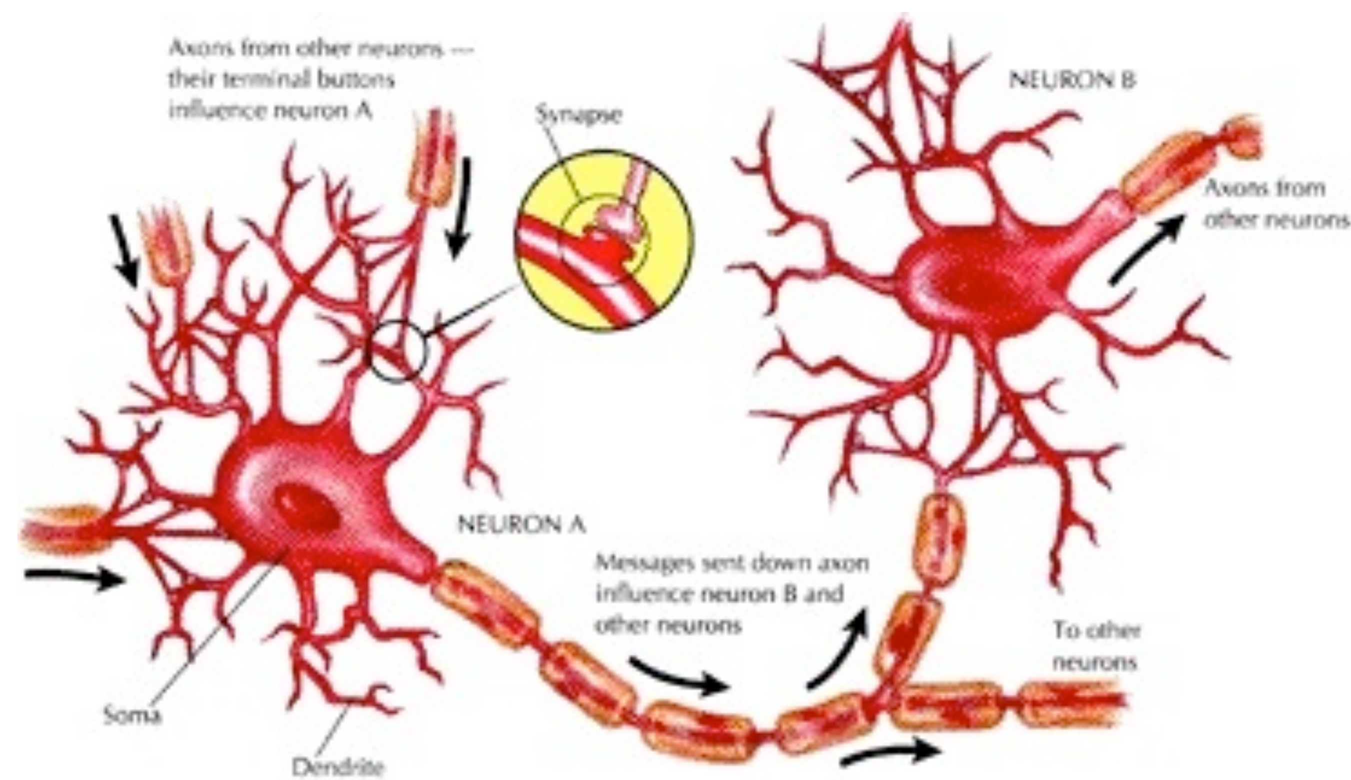
Compréhension intuitive du produit scalaire

chaque composante de x a une influence pondérée sur la sortie y

$$y = f_\theta(\mathbf{x}) = \mathbf{w}_1\mathbf{x}_1 + \mathbf{w}_2\mathbf{x}_2 + \ldots + \mathbf{w}_d\mathbf{x}_d + b$$

Terminologie neuronale





Risque empirique régularisé

Il est souvent nécessaire d'induire une «préférence» pour certaines valeurs des paramètres plutôt que d'autres pour éviter le «surapprentissage» (overfitting)

On définit un risque empirique régularisé comme:

$$\hat{R}_\lambda(f_\theta, D_n) \quad = \quad \underbrace{\left(\sum_{i=1}^{n} L(f_\theta(\mathbf{x}^{(i)}), t^{(i)}) \right)}_{\text{Risque empirique}} + \underbrace{\lambda \Omega(\theta)}_{\substack{\text{terme de régularization} \\ \text{(pénalité)}}}$$

Ω pénalize plus ou moins certains valeurs des paramètres.
$\lambda \geq 0$ contrôle l'importance de la régularization
(par rapport au risque empirique)

Ex: la Régression Ridge

= Régression linéaire+ régularization quadratique (L2)

On pénalise les poids élevés

$$\Omega(\theta) = \Omega(\mathbf{w}, b) = \|\mathbf{w}\|^2 = \sum_{j=1}^{d} \mathbf{w}_j^2$$

Terminologie neuronale:

penalité de type "weight decay"

$$\hat{R}_\lambda(f_\theta, D_n) \quad = \quad \underbrace{\left(\sum_{i=1}^{n} L(f_\theta(\mathbf{x}^{(i)}), t^{(i)}) \right)}_{\text{Risque empirique}} + \underbrace{\lambda\Omega(\theta)}_{\substack{\text{terme de régularization} \\ \text{(pénalité)}}}$$

Ex: la Régression Ridge

= Régression linéaire+ régularization quadratique (L2)

Risque empirique résularisé:

$$\hat{R}_\lambda(f_\theta, D_n) \quad = \quad \underbrace{\left(\sum_{i=1}^{n} L(f_\theta(\mathbf{x}^{(i)}), t^{(i)}) \right)}_{\text{Risque empirique}} + \underbrace{\lambda\Omega(\theta)}_{\substack{\text{terme de régularization} \\ \text{(pénalité)}}}$$

On cherche la valeur des paramètres qui minimisent cet objectif:

$$\{\mathbf{w}^*, b^*\} = \theta^* = \underset{\theta}{\operatorname{argmin}} \hat{R}_\lambda(f_\theta, D_n)$$

Ex: la Régression Ridge

= Régression linéaire+ régularization quadratique (L2)

- Pour la régression linéaire ou ridge un peu d'algèbre linéaire nous donne une **solution analytique**

on résout pour $\theta = \{b, \mathbf{w}\}$: $$\frac{\partial \hat{R}_\lambda(f_\theta, D_n)}{\partial \theta} = 0$$

on obtient: $$\boxed{\begin{pmatrix} b^* \\ \mathbf{w}^* \end{pmatrix} = (\check{X}^T \check{X} + \lambda \check{I})^{-1} \check{X}^T \mathbf{t}}$$

$$\text{où } \check{X} = \begin{pmatrix} 1 & \mathbf{x}_1^{(1)} & \dots & \mathbf{x}_d^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \mathbf{x}_1^{(n)} & \dots & \mathbf{x}_d^{(n)} \end{pmatrix}, \mathbf{t} = \begin{pmatrix} t^{(1)} \\ \vdots \\ t^{(n)} \end{pmatrix} \qquad \check{I} = \begin{pmatrix} 0 & 0 & & & \\ 0 & 1 & 0 & & \\ & 0 & 1 & 0 & \\ & & 0 & \ddots & 0 \\ & & & 0 & 1 \end{pmatrix}$$

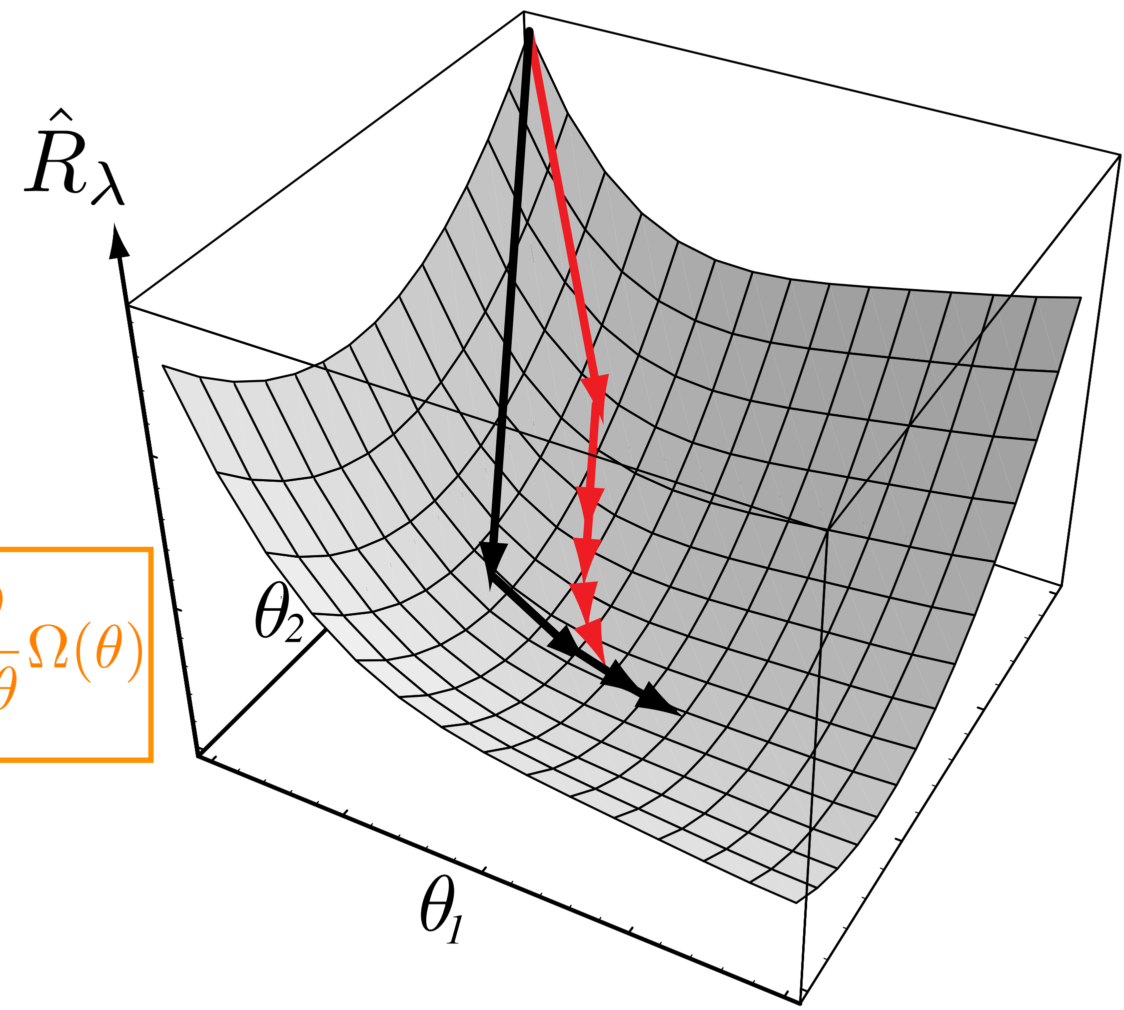
- La plupart du temps (autres choix pour f ou L) on n'a pas de solution analytique.
- D'une manière plus générale, on peut utiliser une technique de descente de gradient.

Autre possibilité:

optimisation par descente de gradient

$$\hat{R}_\lambda(f_\theta, D_n) = \underbrace{\left(\sum_{i=1}^{n} L(f_\theta(\mathbf{x}^{(i)}), t^{(i)})\right)}_{\text{Risque empirique}} + \underbrace{\lambda\Omega(\theta)}_{\text{terme de régularization}}$$

- On initialise les paramètres aléatoirement
- On les mets à jour itérativement en suivant le gradient



Soit **descente de gradient batch** (par lot):

BOUCLE: $\theta \leftarrow \theta - \eta \frac{\partial \hat{R}_\lambda}{\partial \theta}$

$$= \boxed{\left(\sum_{i=1}^{n} \frac{\partial}{\partial \theta} L(f_\theta(\mathbf{x}^{(i)}), t^{(i)})\right) + \lambda \frac{\partial}{\partial \theta}\Omega(\theta)}$$

Ou **descente de gradient stochastique**:

BOUCLE:

Choisir i dans 1...n

$$\theta \leftarrow \theta - \eta \frac{\partial}{\partial \theta}\left(L(f_\theta(\mathbf{x}^{(i)}), t^{(i)}) + \frac{\lambda}{n}\Omega(\theta)\right)$$

Ou **d'autres méthodes de descente de gradient**

(gradient conjugué, Newton, gradient naturel, ...)

Diverses régularisations

«Ridge»: Régularisation L_2

En termes Bayesiens: correspond à un à priori *Gaussien* sur les poids

$$\Omega(\theta) = \Omega(\mathbf{w}, b) = \|\mathbf{w}\|_2^2 = \sum_{j=1}^{d} \mathbf{w}_j^2$$

«Lasso»: Régularisation L_1

En termes Bayesiens: correspond à un à priori *Laplacien* sur les poids

$$\Omega(\theta) = \Omega(\mathbf{w}, b) = \|\mathbf{w}\|_1 = \sum_{j=1}^{d} |\mathbf{w}_j|$$

=> sélection automatique de composantes (un certain nombre de poids seront nuls)

«Elastic net»: combinaison des 2

$$\Omega(\theta) = \Omega(\mathbf{w}, b) = \lambda_1 \|\mathbf{w}\|_1 + \lambda_2 \|\mathbf{w}\|_2^2$$

Etc...