

Discriminants linéaires

Introduction à l'apprentissage automatique – GIF-4101 / GIF-7005

Professeur : Christian Gagné

Semaine 5



5.1 Modèles discriminatifs

Modèles génératifs et modèles discriminatifs

- Modèles génératifs de classement

- Classement basé sur la vraisemblance (densités de probabilité)

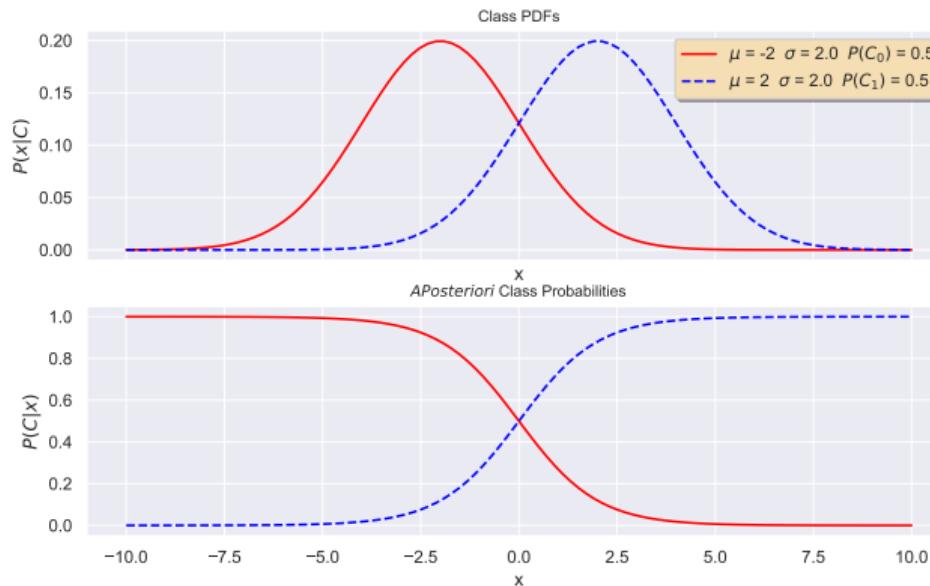
$$h_i(\mathbf{x}) = \log \hat{P}(C_i|\mathbf{x})$$

- Approches paramétriques (incluant densité-mélange) et non paramétriques

- Modèles discriminatifs

- Philosophie : résoudre uniquement le problème de discrimination, l'estimation des densités est une étape « superflue »
 - Obtenir fonction discriminante $h_i(\mathbf{x}|\Phi_i)$ selon une paramétrisation Φ_i
- « When solving a given problem, try to avoid solving a more general problem as an intermediary step. » (Vladimir Vapnik)

Modèles génératifs et modèles discriminatifs



- Modèle génératif : si $P(C_1|x) \geq P(C_0|x)$ alors C_1 , sinon C_0
- Modèle discriminatif : si $x \geq 0$ alors C_1 , sinon C_0

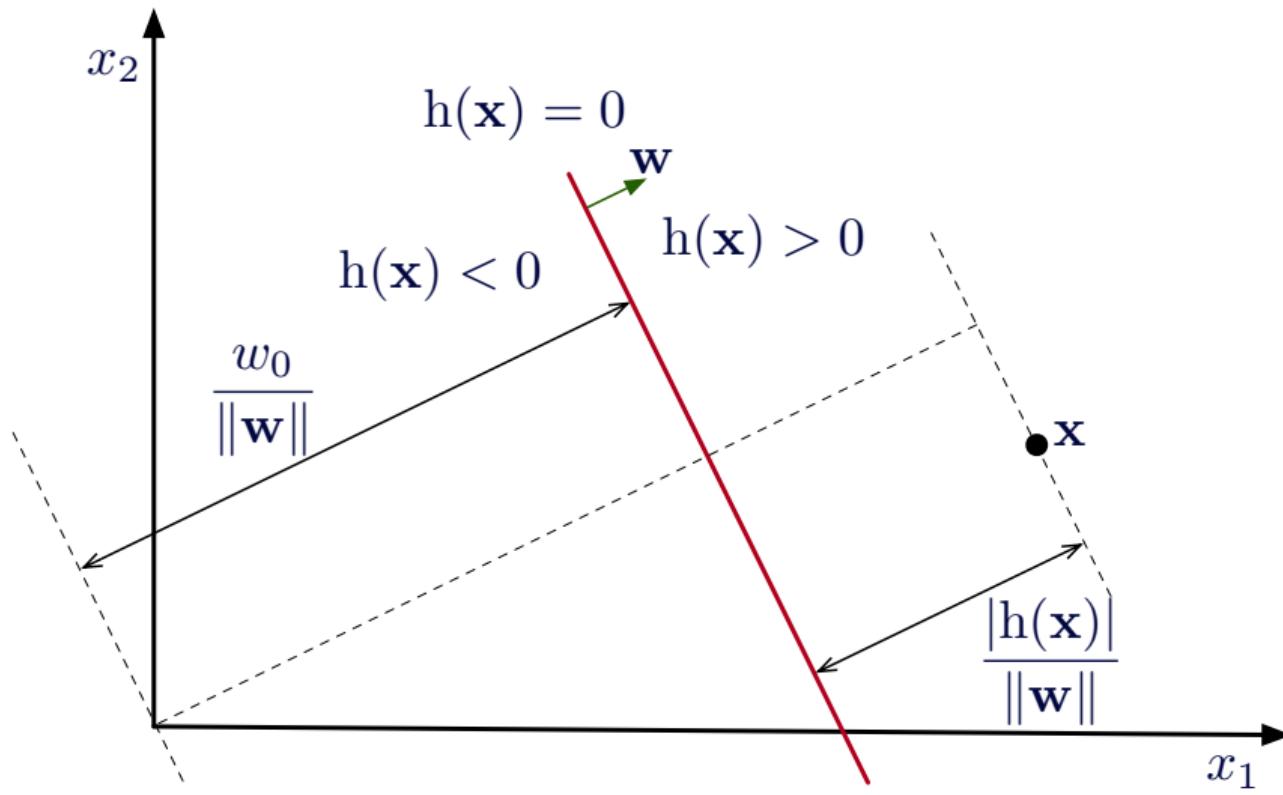
Discriminants linéaires

- Équation d'un discriminant linéaire

$$h_i(\mathbf{x}|\mathbf{w}_i, w_{i,0}) = \sum_{j=1}^D w_{i,j}x_j + w_{i,0}$$

- Modèle à deux classes
 - Une seule équation $h(\mathbf{x}|\mathbf{w}, w_0)$
 - \mathbf{x} appartient à C_1 si $h(\mathbf{x}) \geq 0$
 - Autrement (lorsque $h(\mathbf{x}) < 0$) \mathbf{x} appartient à C_2
 - Poids \mathbf{w} détermine l'orientation de l'hyperplan séparateur
 - Biais w_0 détermine la position de l'hyperplan séparateur dans l'espace d'entrée

Géométrie des discriminants linéaires



5.2 Perceptron

Perceptron

- Perceptron
 - Proposé en 1957 par Rosenblatt
 - Considéré comme le réseau de neurones le plus simple
 - La classe est assignée selon le signe de la fonction discriminante $h(\mathbf{x}|\mathbf{w}, w_0)$

$$h(\mathbf{x}|\mathbf{w}, w_0) = \mathbf{w}^\top \mathbf{x} + w_0, \quad \mathbf{x} \in \begin{cases} C_1 & \text{si } h(\mathbf{x}|\mathbf{w}, w_0) \geq 0 \\ C_2 & \text{autrement} \end{cases}$$

- Optimisation selon le critère du perceptron ($r^t \in \{-1,1\}$)

$$E_{percp}(\mathbf{w}, w_0 | \mathcal{X}) = - \sum_{\mathbf{x}^t \in \mathcal{Y}} r^t h(\mathbf{x}^t | \mathbf{w}, w_0)$$

- \mathcal{Y} représente les données de \mathcal{X} mal classées par $h(\mathbf{x}^t | \mathbf{w}, w_0)$

$$\mathcal{Y} = \{\mathbf{x}^t \in \mathcal{X} \mid r^t h(\mathbf{x}^t | \mathbf{w}, w_0) \leq 0\}$$

Descente du gradient

- Minimisation itérative d'un critère d'erreur $E(\mathbf{w}, w_0 | \mathcal{X})$ selon un jeu de données \mathcal{X}

$$\{\mathbf{w}^*, w_0^*\} = \underset{\{\mathbf{w}, w_0\}}{\operatorname{argmin}} E(\mathbf{w}, w_0 | \mathcal{X})$$

- Résolution par dérivées partielles, $\nabla_{\mathbf{w}} E$

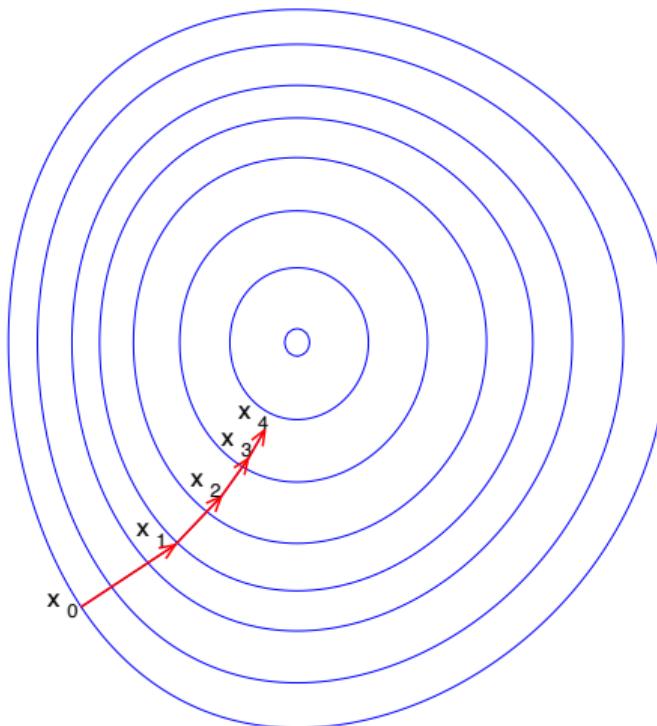
$$\nabla_{\mathbf{w}} E = \left[\frac{\partial E}{\partial w_0} \quad \frac{\partial E}{\partial w_1} \quad \frac{\partial E}{\partial w_2} \quad \cdots \quad \frac{\partial E}{\partial w_D} \right]$$

- Modification des poids w_i dans la direction opposée au gradient (on descend le gradient)

$$w_i = w_i + \Delta w_i, \quad \Delta w_i = -\eta \frac{\partial E}{\partial w_i}, \quad i = 0, \dots, D$$

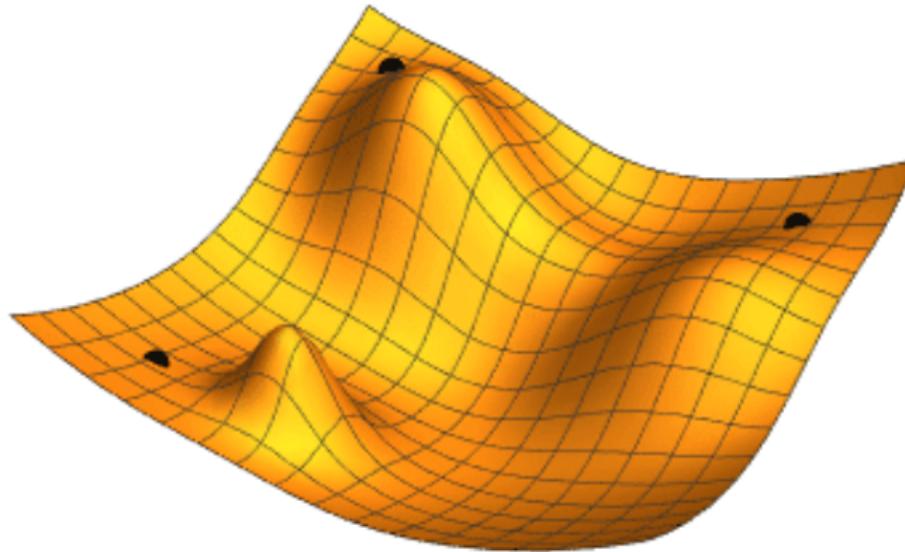
- $\eta \in [0, 1]$ est le pas ou *taux d'apprentissage*

Descente du gradient

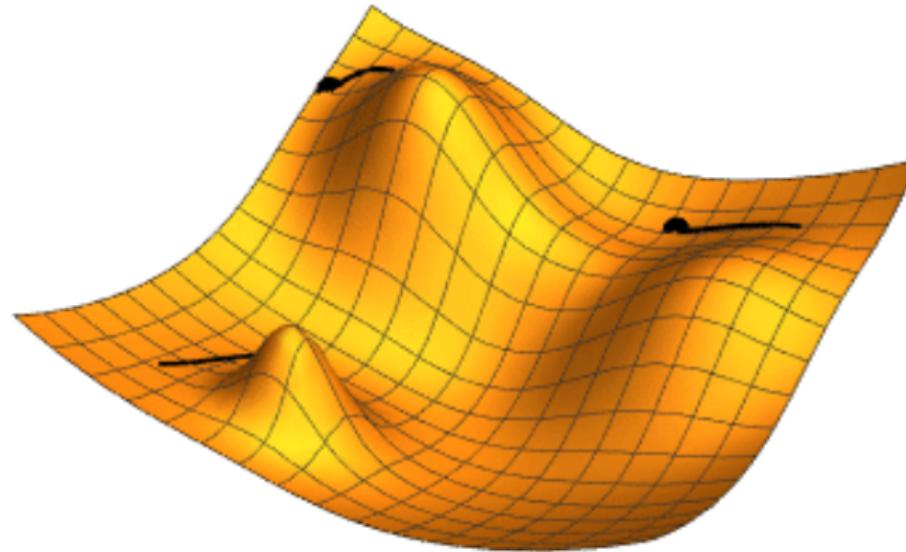


Domaine public, https://commons.wikimedia.org/wiki/File:Gradient_descent.svg

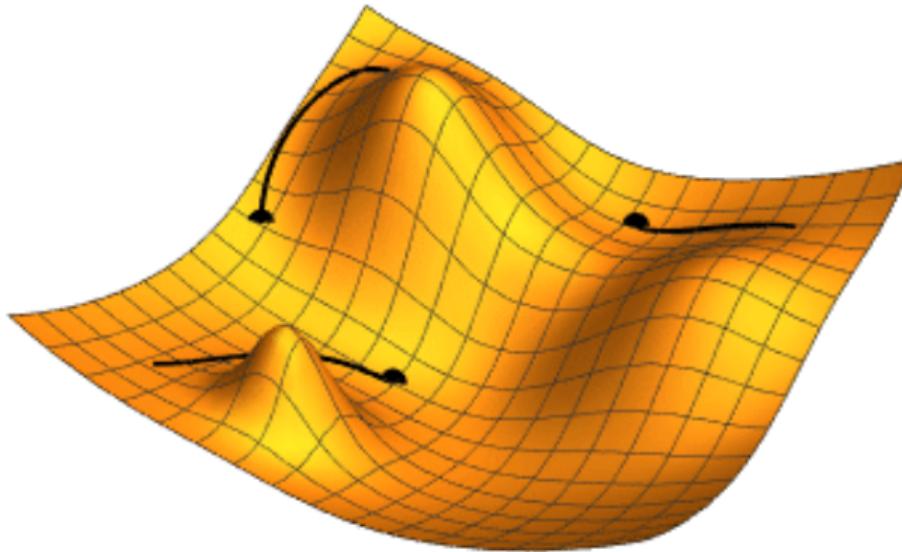
Descente du gradient



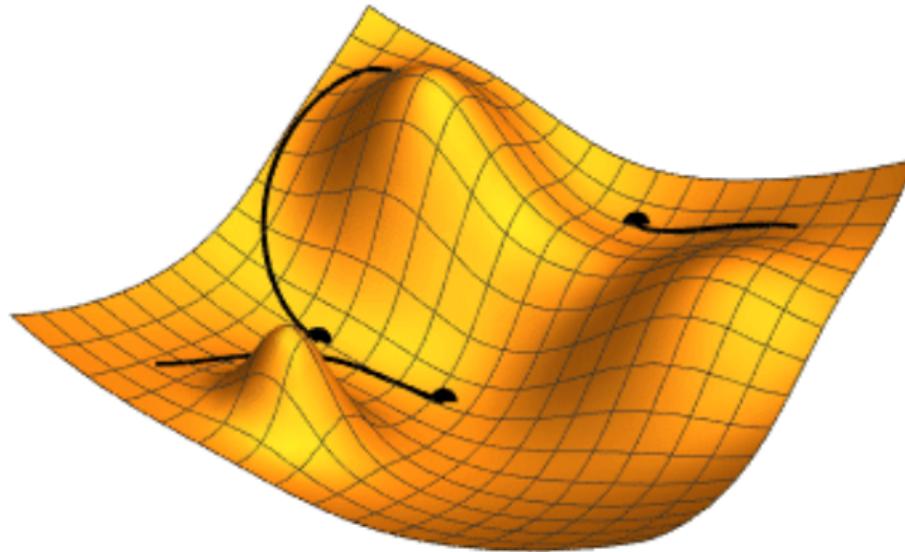
Descente du gradient



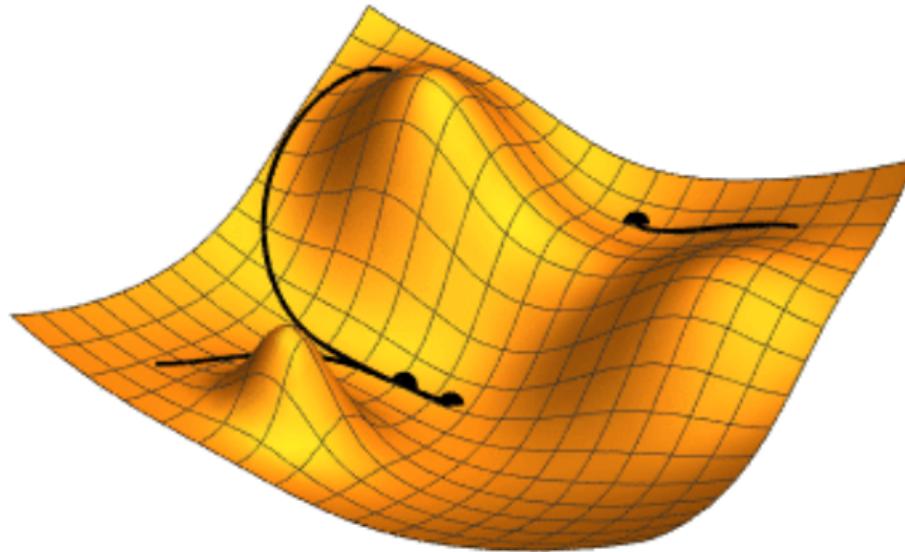
Descente du gradient



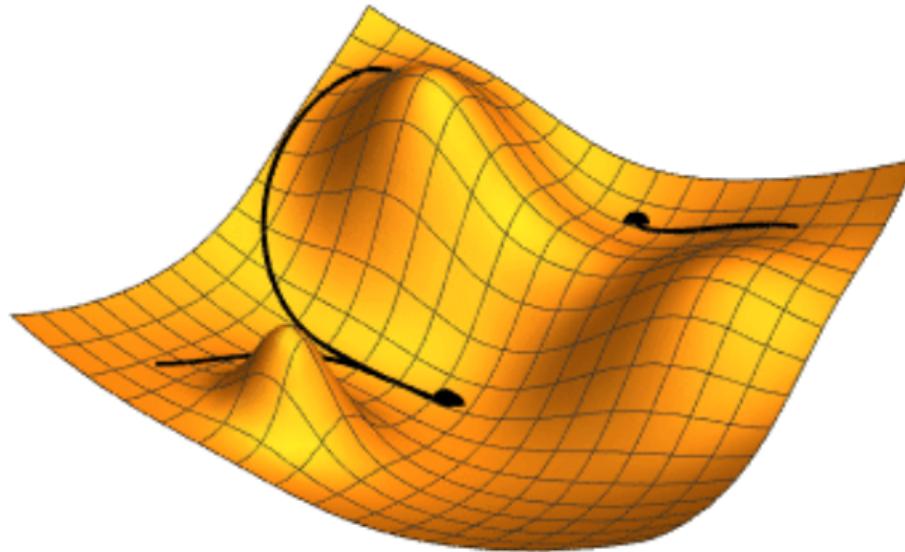
Descente du gradient



Descente du gradient



Descente du gradient



Descente du gradient avec perceptron

- Critère d'erreur du perceptron

$$E_{percp}(\mathbf{w}, w_0 | \mathcal{X}) = - \sum_{\mathbf{x}^t \in \mathcal{Y}} r^t h(\mathbf{x}^t | \mathbf{w}, w_0)$$

- \mathcal{Y} est l'ensemble des données de \mathcal{X} mal classées par $h(\mathbf{x}^t | \mathbf{w}, w_0)$
- Calcul du gradient $\nabla E_{percp}(\mathbf{w}, w_0 | \mathcal{X})$

$$\frac{\partial E}{\partial w_i} = \frac{\partial (-\sum_{\mathbf{x}^t \in \mathcal{Y}} r^t (\mathbf{w}^\top \mathbf{x} + w_0))}{\partial w_i} = - \sum_{\mathbf{x}^t \in \mathcal{Y}} r^t x_i^t$$

$$\frac{\partial E}{\partial w_0} = \frac{\partial (-\sum_{\mathbf{x}^t \in \mathcal{Y}} r^t (\mathbf{w}^\top \mathbf{x} + w_0))}{\partial w_0} = - \sum_{\mathbf{x}^t \in \mathcal{Y}} r^t$$

- Descente du gradient $w_i = w_i + \Delta w_i, i = 0, \dots, D$

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i} = \eta \sum_{\mathbf{x}^t \in \mathcal{Y}} r^t x_i^t, \quad \Delta w_0 = -\eta \frac{\partial E}{\partial w_0} = \eta \sum_{\mathbf{x}^t \in \mathcal{Y}} r^t$$

Algorithme du perceptron

1. Initialiser les poids \mathbf{w} et w_0 arbitrairement

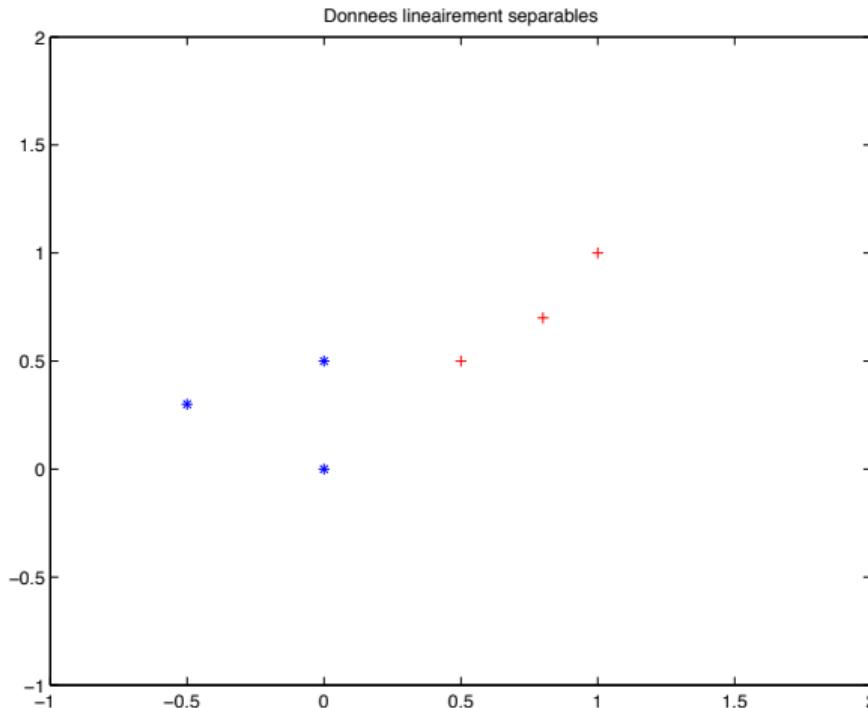
$$w_j = 0, \quad j = 0, \dots, D$$

2. Répéter jusqu'à convergence ou épuisement des ressources :

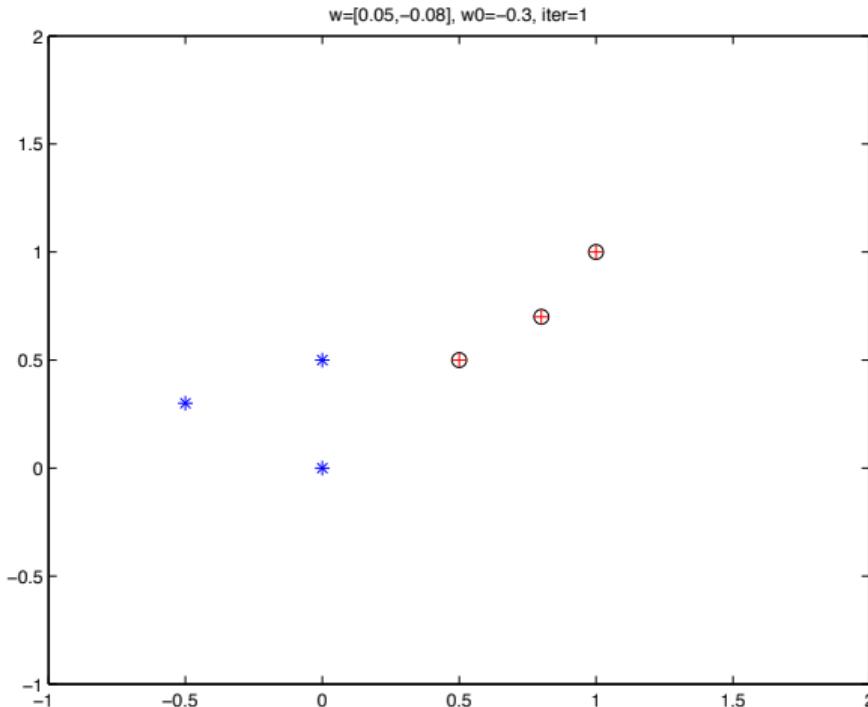
$$w_j = w_j + \eta \sum_{\mathbf{x}^t \in \mathcal{Y}} r^t x_j^t, \quad j = 1, \dots, D$$

$$w_0 = w_0 + \eta \sum_{\mathbf{x}^t \in \mathcal{Y}} r^t$$

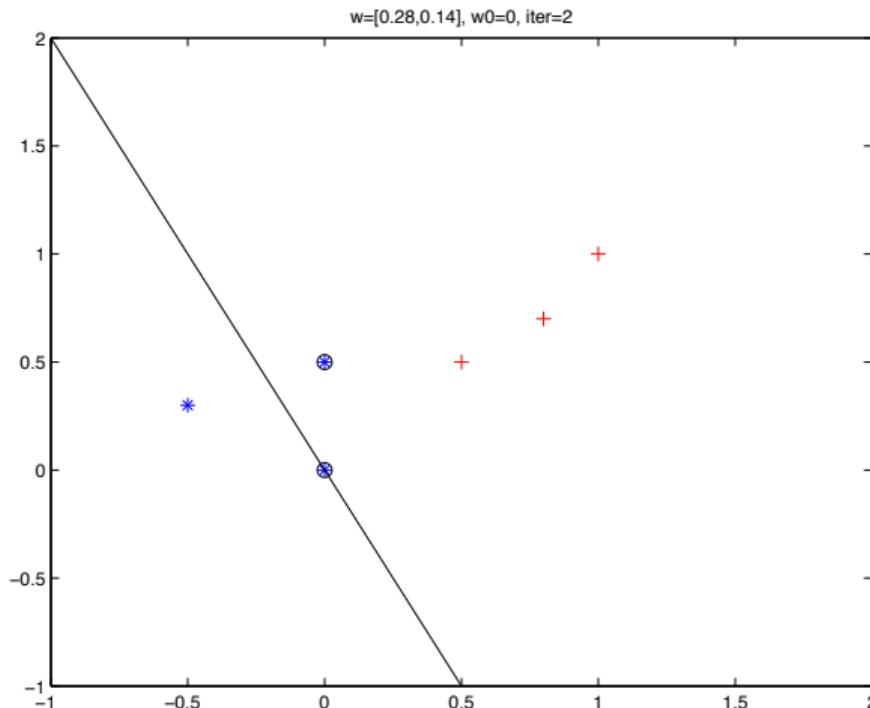
Exemple avec le perceptron (linéairement séparable)



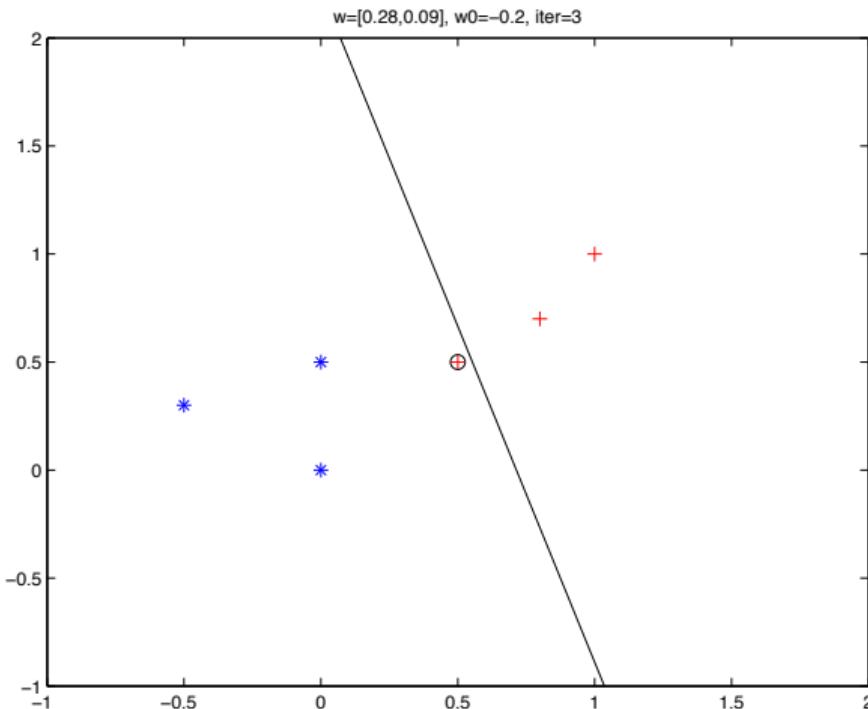
Exemple avec le perceptron (linéairement séparable)



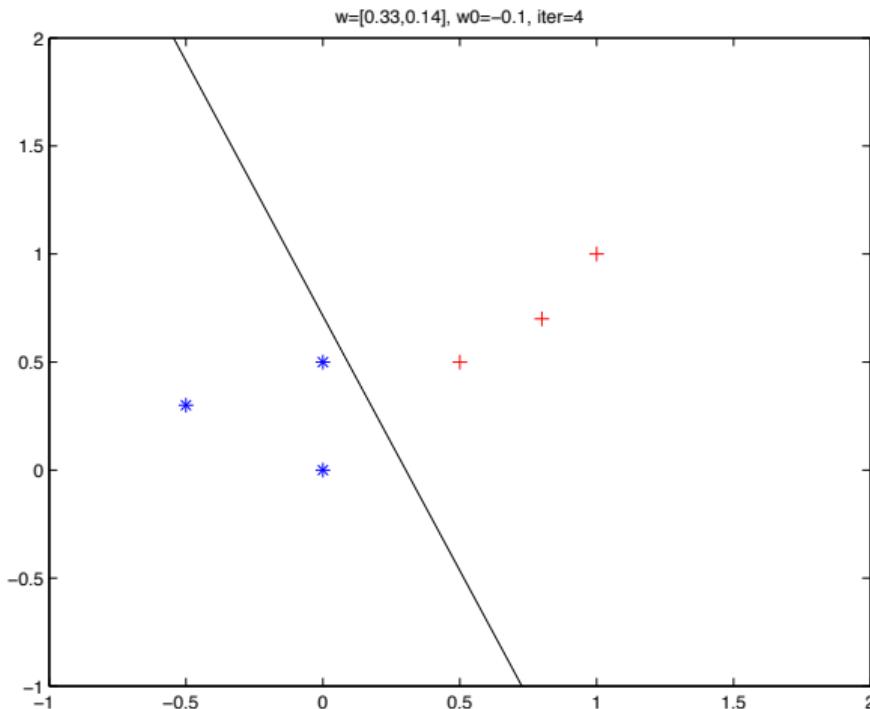
Exemple avec le perceptron (linéairement séparable)



Exemple avec le perceptron (linéairement séparable)



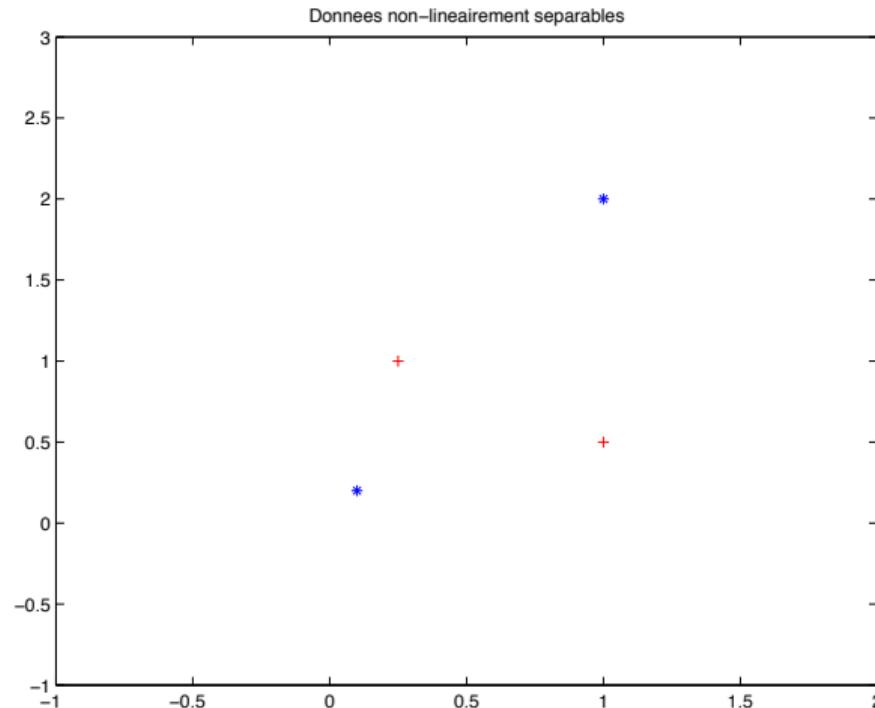
Exemple avec le perceptron (linéairement séparable)



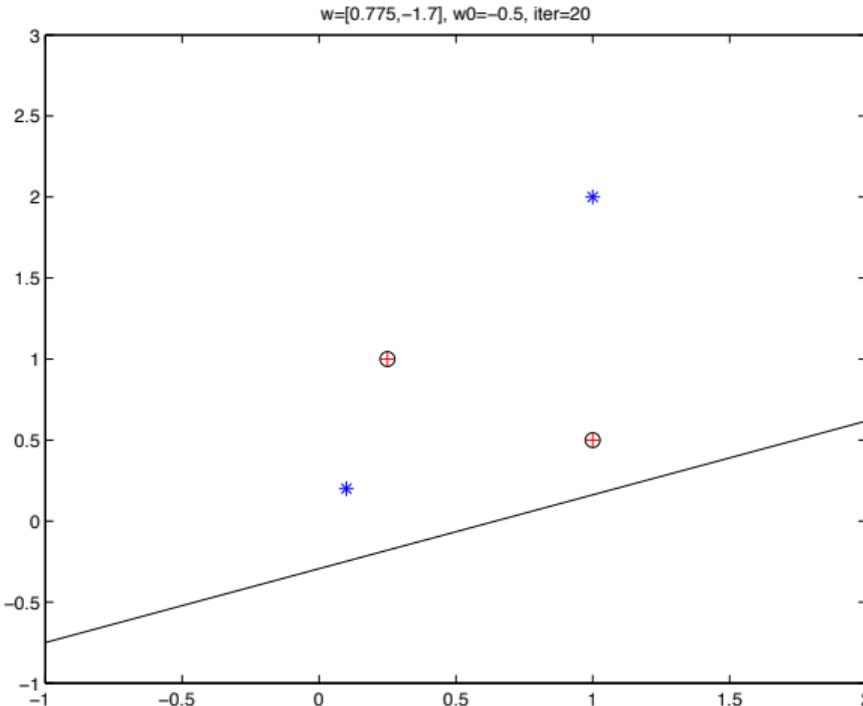
Convergence du perceptron

- Convergence sur données linéairement séparables
 - Preuve mathématique de convergence existe pour données linéairement séparables
 - Convergence vers une position quelconque du discriminant qui sépare les données
 - Pour données non linéairement séparables, pas de convergence
- Critère d'erreur en faible lien avec la nature des erreurs

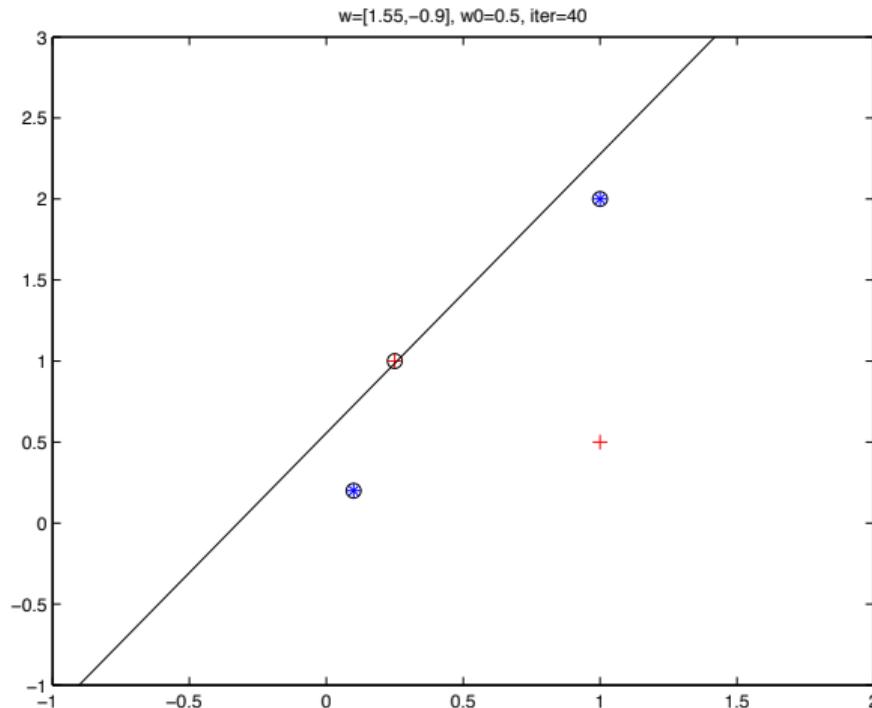
Exemple avec le perceptron (non linéairement séparable)



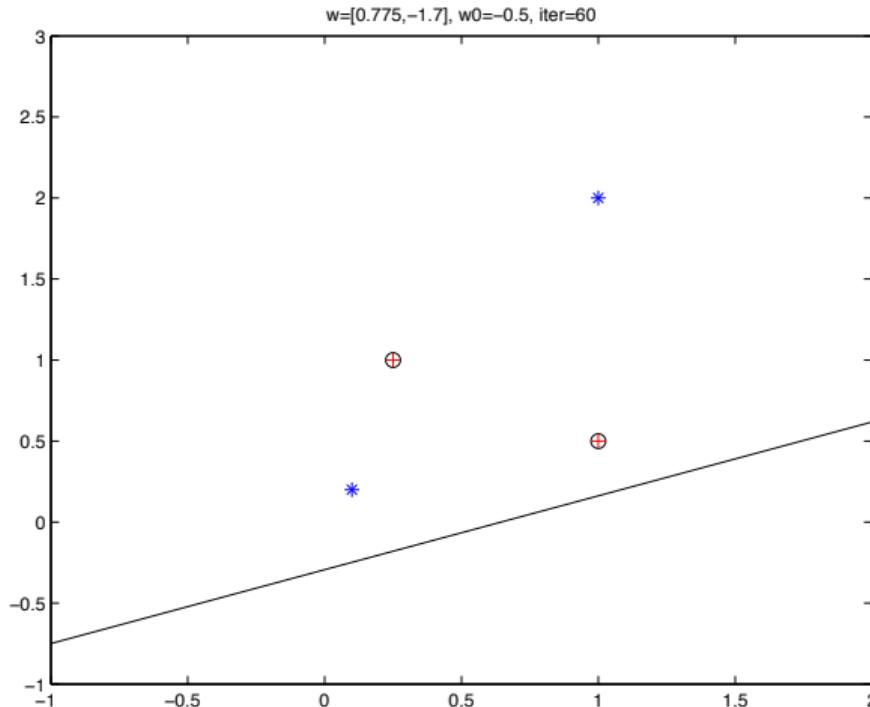
Exemple avec le perceptron (non linéairement séparable)



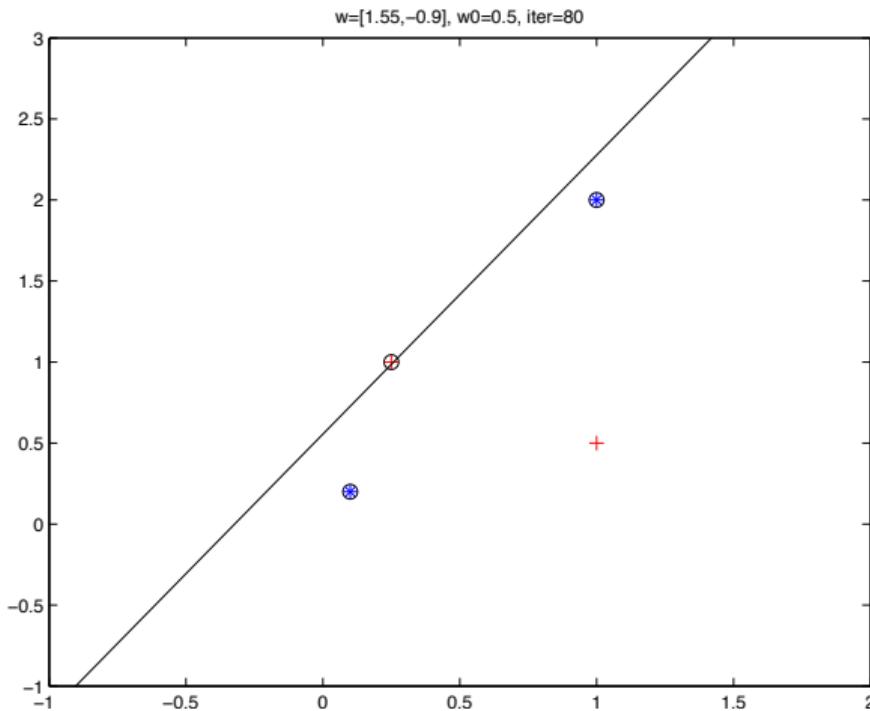
Exemple avec le perceptron (non linéairement séparable)



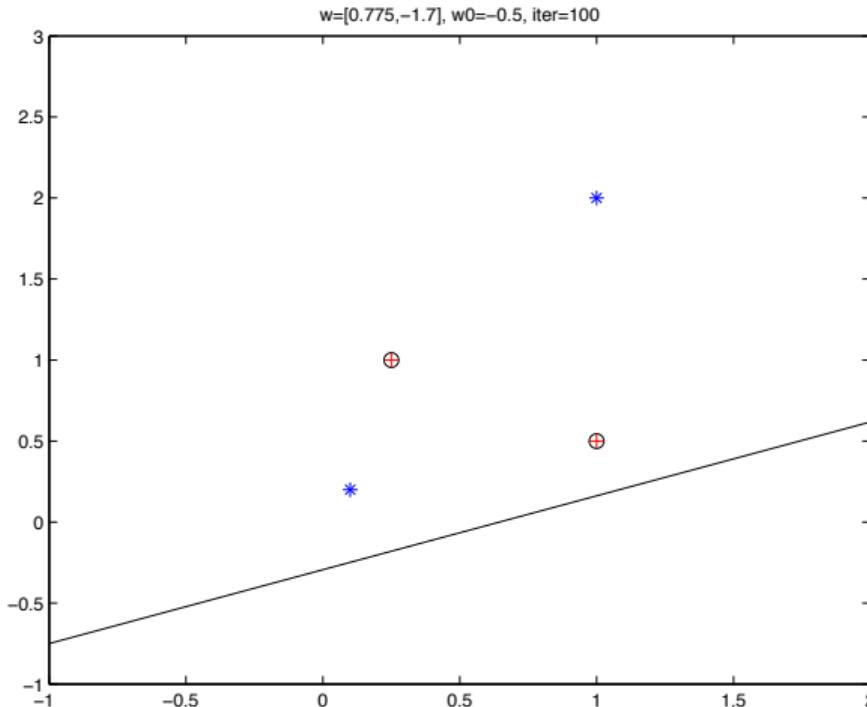
Exemple avec le perceptron (non linéairement séparable)



Exemple avec le perceptron (non linéairement séparable)



Exemple avec le perceptron (non linéairement séparable)



5.3 Méthode des moindres carrés

Régression pour le classement

- En régression, l'information sur les erreurs est riche
 - Différences fines entre valeurs cibles r^t et valeurs obtenues par $h(\mathbf{x}^t)$
 - Valeurs cibles $r^t \in \mathbb{R}$ en régression plus générales que valeurs cibles discrètes $r^t \in \{-1,1\}$ en classement
 - Erreur quadratique pour la régression linéaire

$$E_{quad}(\mathbf{w}, w_0 | \mathcal{X}) = \frac{1}{2} \sum_{\mathbf{x}^t \in \mathcal{X}} (r^t - h(\mathbf{x}^t | \mathbf{w}, w_0))^2$$

- Régression pour le classement
 - Optimiser l'hyperplan séparateur en traitant r^t et $h(\mathbf{x}^t | \mathbf{w}, w_0)$ comme des réels

Méthode des moindres carrés

- Descente du gradient selon l'erreur quadratique ($r^t \in \{-1,1\}$)

$$E_{quad}(\mathbf{w}, w_0 | \mathcal{X}) = \frac{1}{2} \sum_{\mathbf{x}^t \in \mathcal{X}} (r^t - (\mathbf{w}^\top \mathbf{x}^t + w_0))^2$$

$$\frac{\partial E_{quad}}{\partial w_i} = \frac{1}{2} \sum_{\mathbf{x}^t \in \mathcal{X}} (-2x_i^t)(r^t - (\mathbf{w}^\top \mathbf{x}^t + w_0))$$

$$\frac{\partial E_{quad}}{\partial w_0} = \frac{1}{2} \sum_{\mathbf{x}^t \in \mathcal{X}} (-2)(r^t - (\mathbf{w}^\top \mathbf{x}^t + w_0))$$

- En posant $e(\mathbf{x}^t) = r^t - h(\mathbf{x}^t | \mathbf{w}, w_0)$, alors

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i} = \eta \sum_{\mathbf{x}^t \in \mathcal{X}} e(\mathbf{x}^t) x_i^t$$

$$\Delta w_0 = -\eta \frac{\partial E}{\partial w_0} = \eta \sum_{\mathbf{x}^t \in \mathcal{X}} e(\mathbf{x}^t)$$

Algorithme pour classement avec moindres carrés

1. Initialiser les poids \mathbf{w} et w_0 arbitrairement

$$w_j = 0, \quad j = 0, \dots, D$$

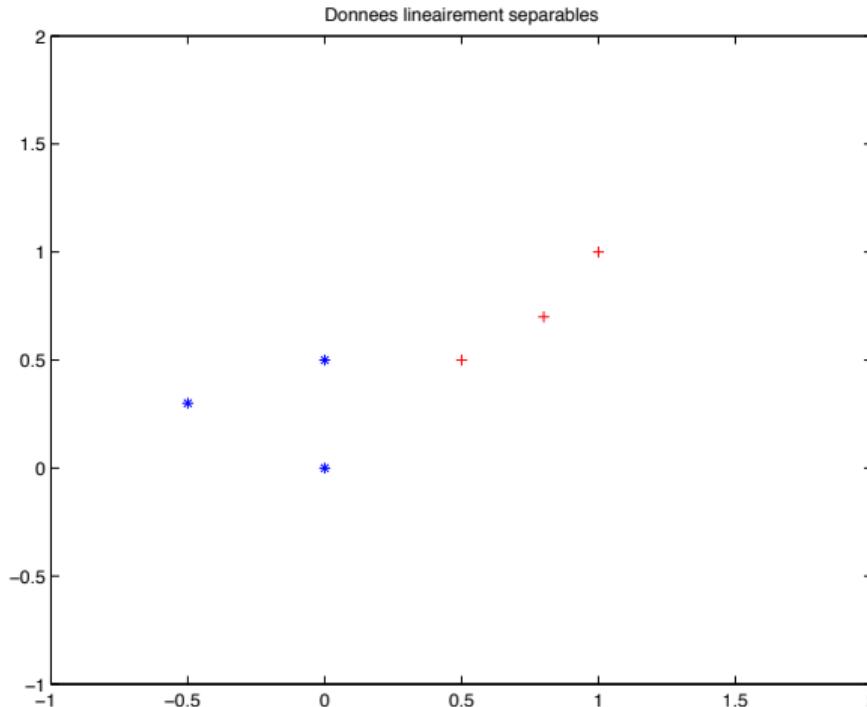
2. Répéter jusqu'à convergence ou épuisement des ressources :

$$e(\mathbf{x}^t) = r^t - (\mathbf{w}^\top \mathbf{x}^t + w_0)$$

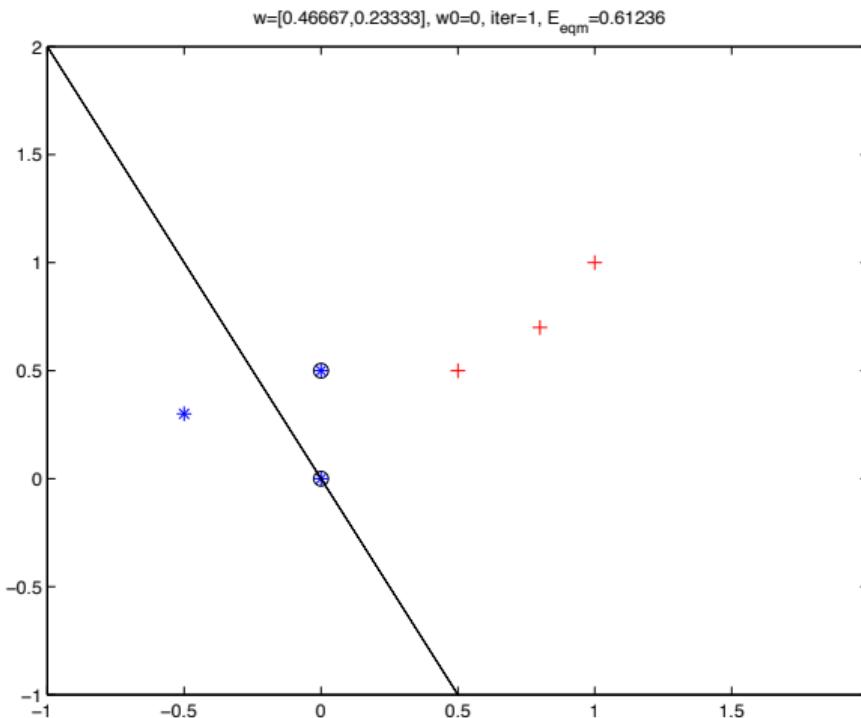
$$w_j = w_j + \eta \sum_{\mathbf{x}^t \in \mathcal{X}} e(\mathbf{x}^t) x_j^t, \quad j = 1, \dots, D$$

$$w_0 = w_0 + \eta \sum_{\mathbf{x}^t \in \mathcal{X}} e(\mathbf{x}^t)$$

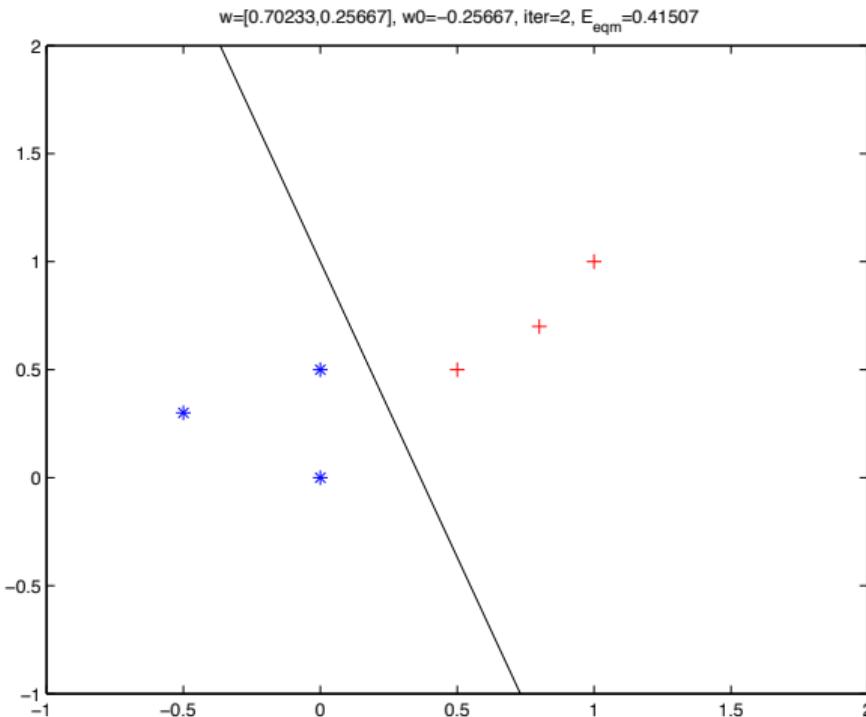
Exemple moindres carrés (linéairement séparable)



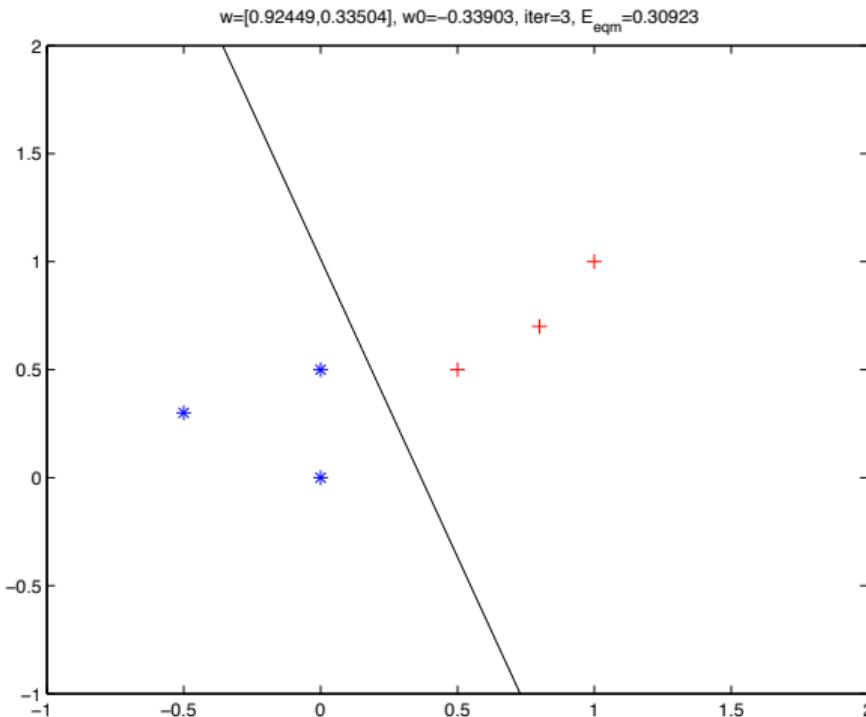
Exemple moindres carrés (linéairement séparable)



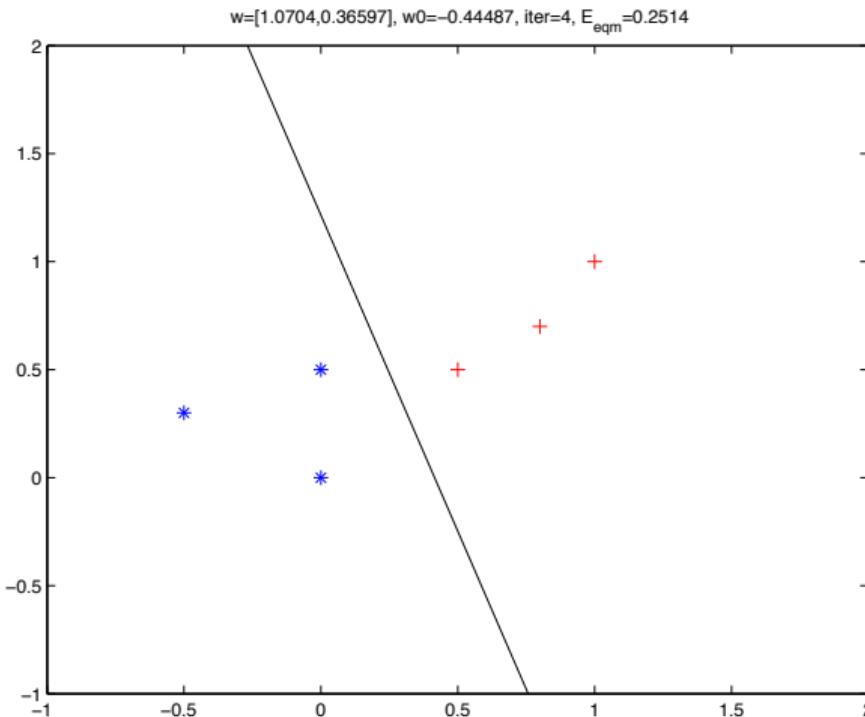
Exemple moindres carrés (linéairement séparable)



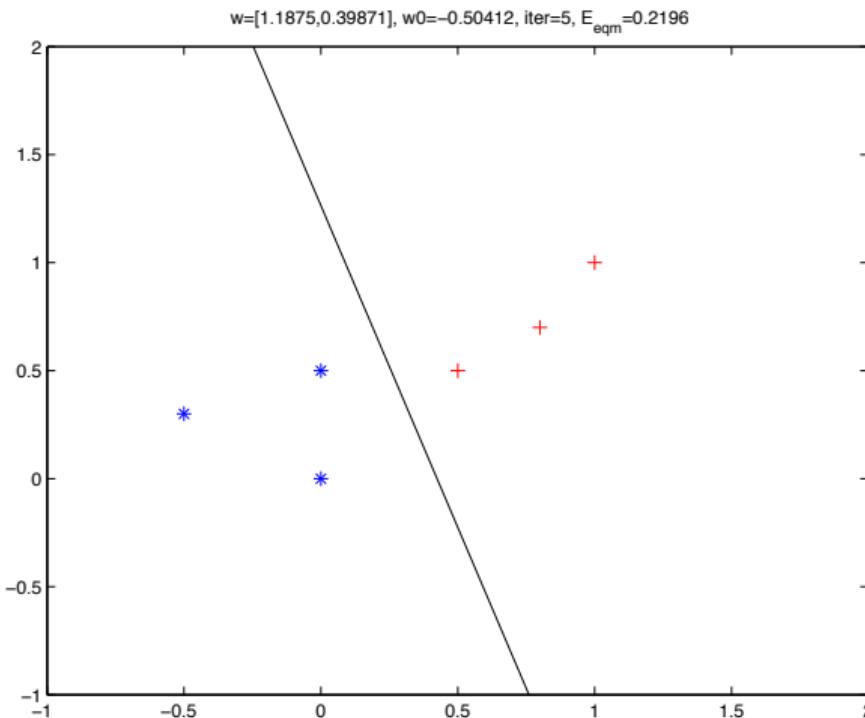
Exemple moindres carrés (linéairement séparable)



Exemple moindres carrés (linéairement séparable)



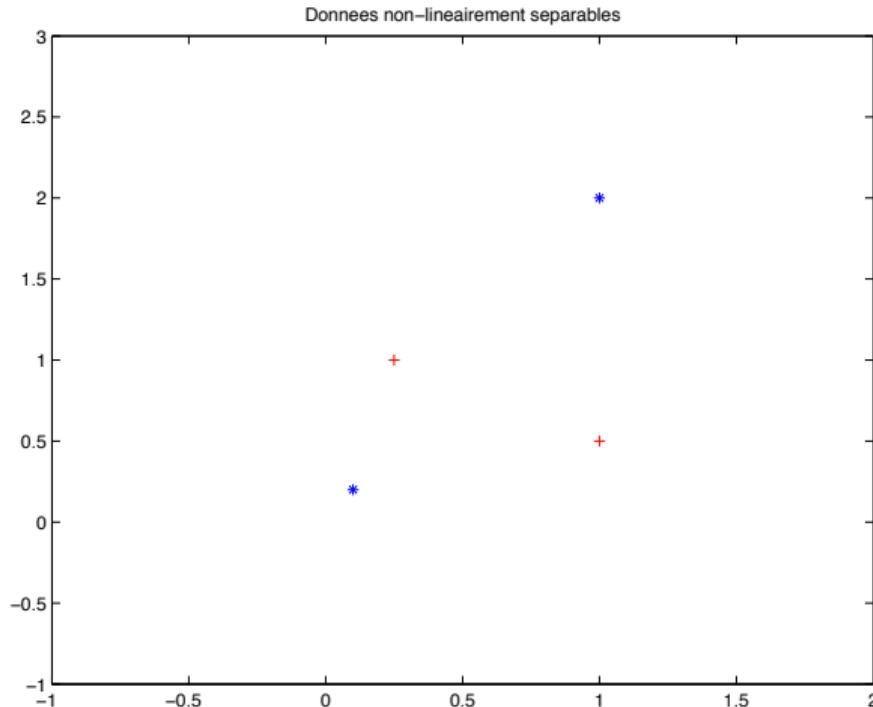
Exemple moindres carrés (linéairement séparable)



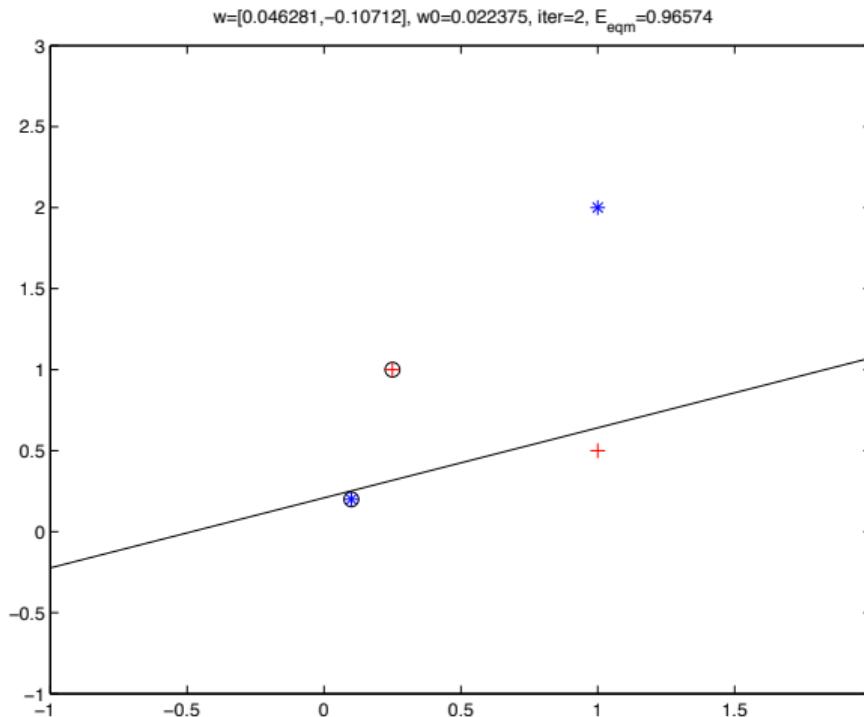
Convergence de la méthode des moindres carrés

- Convergence sur données linéairement séparables
 - Placement de l'hyperplan séparateur à une position minimisant l'erreur quadratique
 - Emphase sur les données avec l'erreur la plus grande
 - Forte influence des données bien classées, mais éloignées de l'hyperplan séparateur
- Données non linéairement séparables
 - Meilleur placement possible de l'hyperplan selon l'erreur quadratique

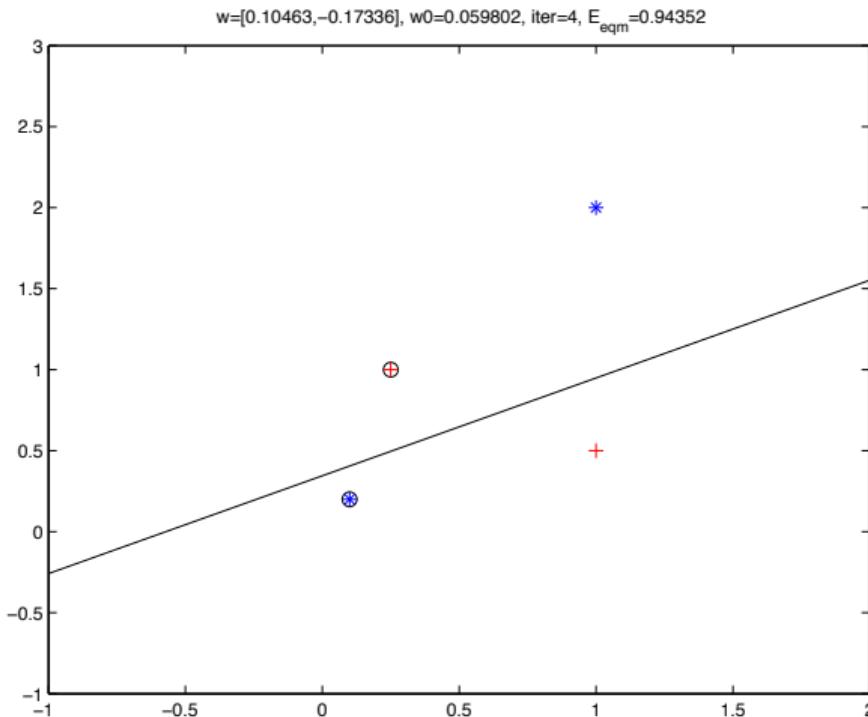
Exemple moindres carrés (non linéairement séparable)



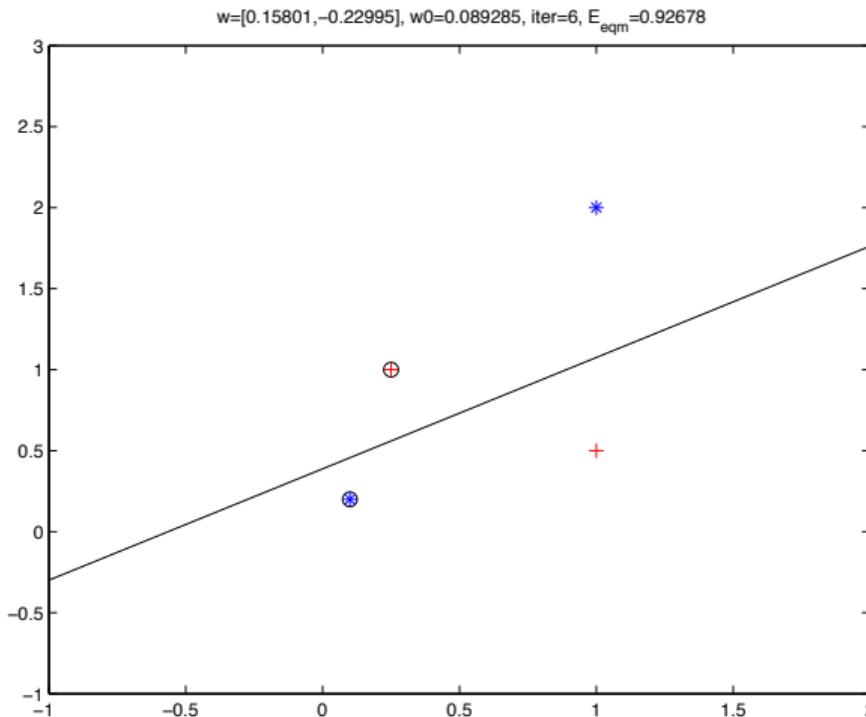
Exemple moindres carrés (non linéairement séparable)



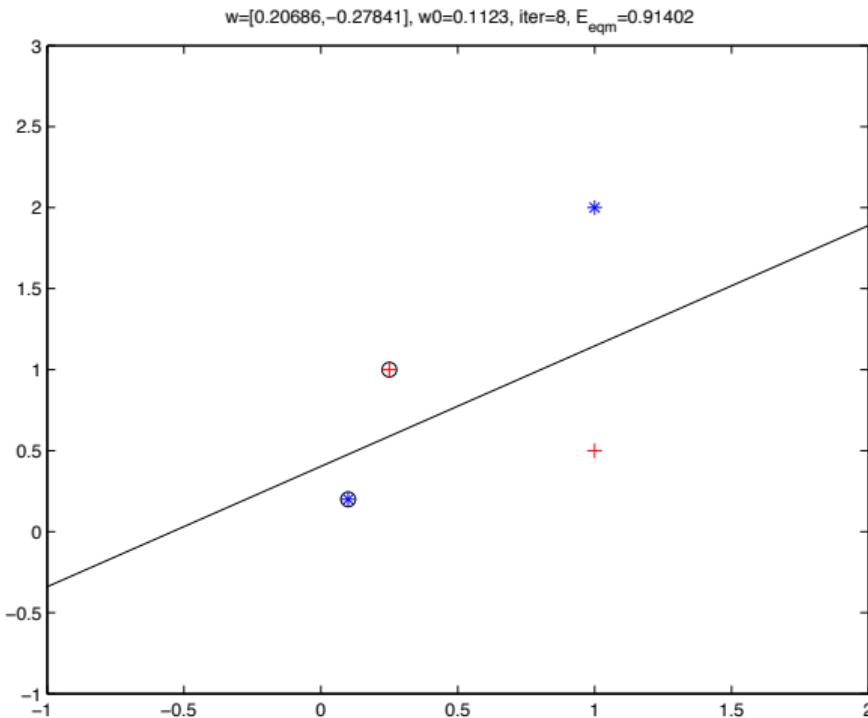
Exemple moindres carrés (non linéairement séparable)



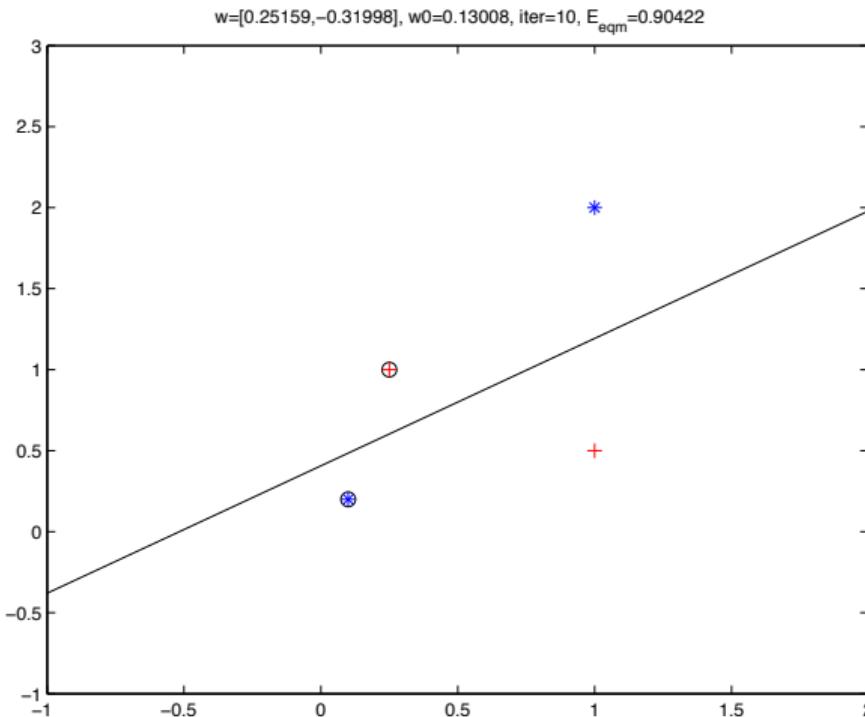
Exemple moindres carrés (non linéairement séparable)



Exemple moindres carrés (non linéairement séparable)



Exemple moindres carrés (non linéairement séparable)



Taux d'apprentissage η

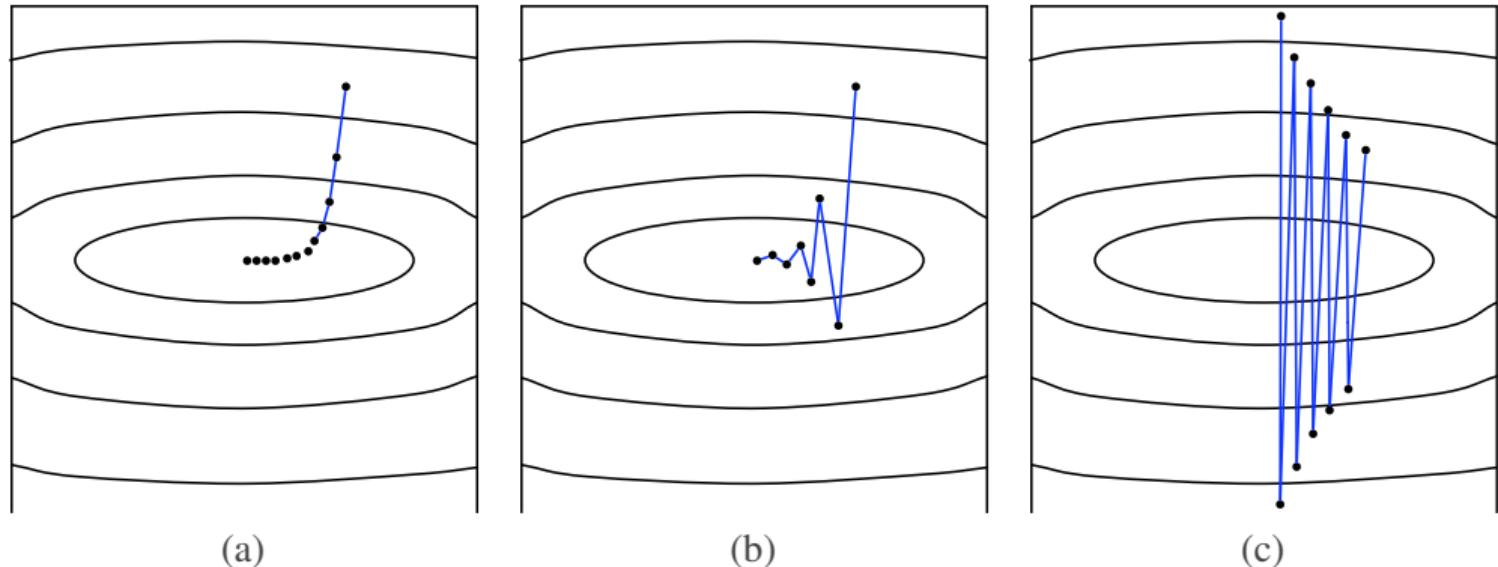


FIG. 5.6 – Trajectoire de la descente du gradient pour différents taux d'apprentissage : (a) taux faible ; (b) taux moyen ; (c) taux (trop) élevé.

5.4 Méthodes paramétriques linéaires

Classement paramétrique

- Fonction discriminante avec classement paramétrique

$$h_i(\mathbf{x}) = p(\mathbf{x}|C_i)P(C_i)$$

- Utilisation du log : $h_i(\mathbf{x}) = \log p(\mathbf{x}|C_i)P(C_i) = \log p(\mathbf{x}|C_i) + \log P(C_i)$
- Si $p(\mathbf{x}|C_i)$ correspond à une loi normale multivariée

$$p(\mathbf{x}|C_i) = \frac{1}{(2\pi)^{0,5D}|\Sigma_i|^{0,5D}} \exp \left[-0,5(\mathbf{x} - \boldsymbol{\mu}_i)^\top \Sigma_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i) \right]$$

$$h_i(\mathbf{x}) = -0,5 \log |\Sigma_i| - 0,5(\mathbf{x} - \boldsymbol{\mu}_i)^\top \Sigma_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i) + \log \hat{P}(C_i)$$

$$\hat{P}(C_i) = \frac{\sum_t r_i^t}{N}$$

$$\mathbf{m}_i = \frac{\sum_t r_i^t \mathbf{x}^t}{\sum_t r_i^t}$$

$$\mathbf{S}_i = \frac{\sum_t r_i^t (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^\top}{\sum_t r_i^t}$$

Classement paramétrique pour discrimination linéaire

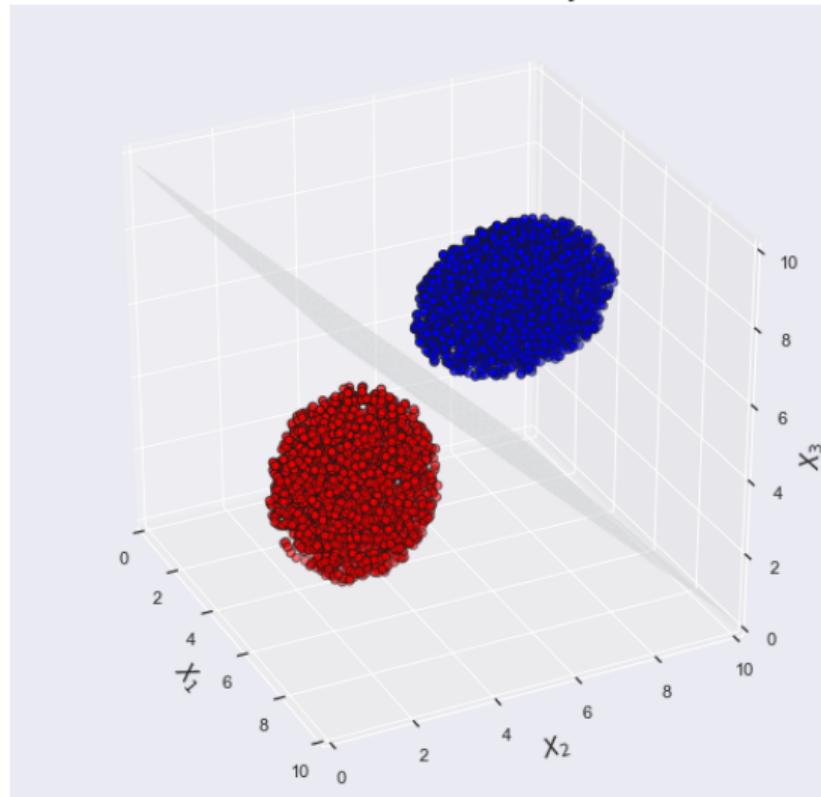
- Si l'estimation des matrices de covariance est partagée, $\mathbf{S} = \sum_i \hat{P}(C_i) \mathbf{S}_i$

$$\begin{aligned} h_i(\mathbf{x}) &= -0,5(\mathbf{x} - \mathbf{m}_i)^\top \mathbf{S}^{-1}(\mathbf{x} - \mathbf{m}_i) + \log \hat{P}(C_i) \\ &= -0,5 \left(\mathbf{x}^\top \mathbf{S}^{-1} \mathbf{x} - 2\mathbf{x}^\top \mathbf{S}^{-1} \mathbf{m}_i + \mathbf{m}_i^\top \mathbf{S}^{-1} \mathbf{m}_i \right) + \log \hat{P}(C_i) \\ &= \mathbf{x}^\top \mathbf{S}^{-1} \mathbf{m}_i + (-0,5 \mathbf{m}_i^\top \mathbf{S}^{-1} \mathbf{m}_i + \log \hat{P}(C_i)) \\ &= \mathbf{w}_i^\top \mathbf{x} + w_{i,0} \end{aligned}$$

$$\text{où } \mathbf{w}_i = \mathbf{S}^{-1} \mathbf{m}_i$$

$$w_{i,0} = -0,5 \mathbf{m}_i^\top \mathbf{S}^{-1} \mathbf{m}_i + \log \hat{P}(C_i)$$

Discrimination linéaire avec méthodes paramétriques



Fonction logit

- Classement paramétrique à deux classes (C_1 et C_2)
 - Choisir C_1 pour \mathbf{x} lorsque $P(C_1|\mathbf{x}) > P(C_2|\mathbf{x})$ et C_2 autrement
 - À deux classes $P(C_1|\mathbf{x}) + P(C_2|\mathbf{x}) = 1$, donc $P(C_2|\mathbf{x}) = 1 - P(C_1|\mathbf{x})$
 - Formulations équivalentes, en posant $y \equiv P(C_1|\mathbf{x})$

$$\begin{aligned}P(C_1|\mathbf{x}) &> P(C_2|\mathbf{x}) \quad \Rightarrow \quad y > (1-y) \\ \frac{y}{1-y} &> 1 \quad \Rightarrow \quad \log \frac{y}{1-y} > 0\end{aligned}$$

- $f_{logit}(y) = \log \frac{y}{1-y}$ est nommé fonction *logit*

Classement paramétrique et discriminant linéaire

- Deux classes suivant des lois normales multivariées avec matrice de covariance partagée : discriminant linéaire

$$\begin{aligned} f_{logit}(P(C_1|\mathbf{x})) &= \log \frac{P(C_1|\mathbf{x})}{1 - P(C_1|\mathbf{x})} = \log \frac{P(C_1|\mathbf{x})}{P(C_2|\mathbf{x})} \\ &= \log \frac{p(\mathbf{x}|C_1)}{p(\mathbf{x}|C_2)} + \log \frac{P(C_1)}{P(C_2)} \\ &= \log \frac{(2\pi)^{-0,5D} |\Sigma|^{-0,5} \exp[-0,5(\mathbf{x} - \boldsymbol{\mu}_1)^\top \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}_1)]}{(2\pi)^{-0,5D} |\Sigma|^{-0,5} \exp[-0,5(\mathbf{x} - \boldsymbol{\mu}_2)^\top \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}_2)]} + \log \frac{P(C_1)}{P(C_2)} \\ &= \mathbf{w}^\top \mathbf{x} + w_0 \end{aligned}$$

avec :

$$\begin{aligned} \mathbf{w} &= \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \\ w_0 &= -0,5(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2)^\top \Sigma^{-1}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2) + \log \frac{P(C_1)}{P(C_2)} \end{aligned}$$

Fonction sigmoïde

- Fonction logit

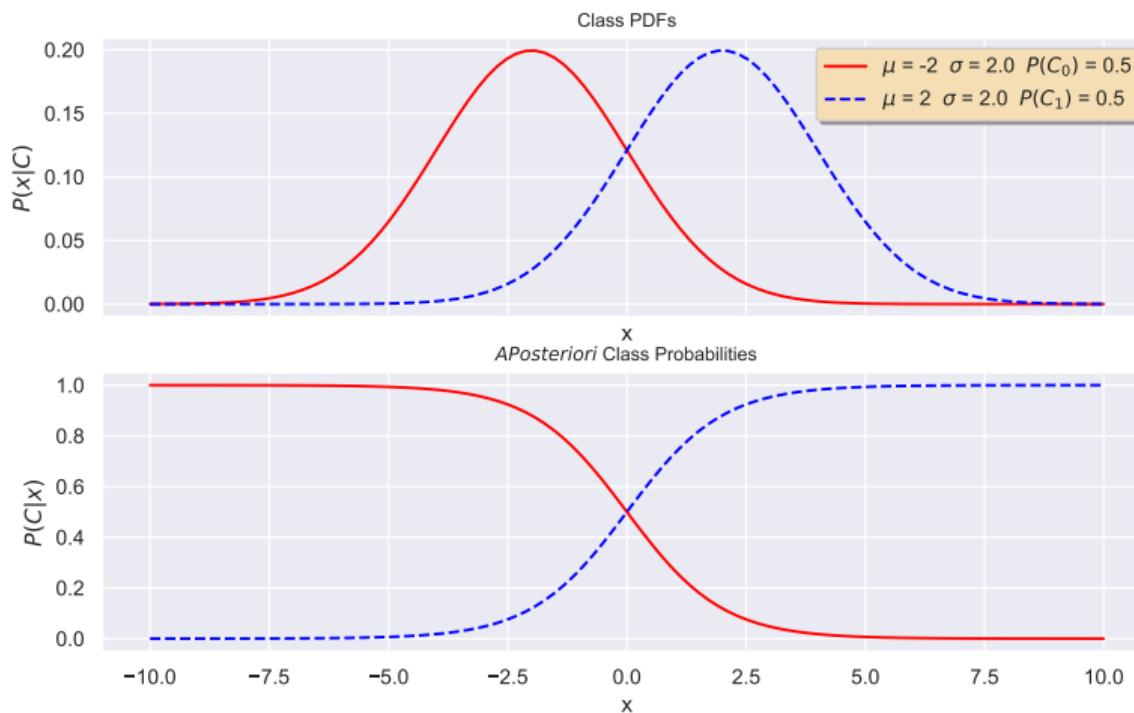
$$f_{logit}(P(C_1|\mathbf{x})) = \log \frac{P(C_1|\mathbf{x})}{1 - P(C_1|\mathbf{x})} = \mathbf{w}^\top \mathbf{x} + w_0$$

- Inverse de la fonction logit : fonction sigmoïde (aussi nommée fonction logistique)

$$f_{logit}(y) = \log \frac{y}{1 - y} = a \Rightarrow y = f_{sig}(a) = \frac{1}{1 + \exp(-a)}$$

$$P(C_1|\mathbf{x}) = f_{sig}(\mathbf{w}^\top \mathbf{x} + w_0) = \frac{1}{1 + \exp[-(\mathbf{w}^\top \mathbf{x} + w_0)]}$$

Densités normales et probabilités a posteriori



5.5 Régression logistique

Régression logistique

- Régression logistique : estimer $P(C_1|\mathbf{x})$ par descente du gradient

$$y = \hat{P}(C_1|\mathbf{x}) = \frac{1}{1 + \exp[-(\mathbf{w}^\top \mathbf{x} + w_0)]}$$

- Apprendre \mathbf{w} et w_0 à partir de $\mathcal{X} = \{\mathbf{x}^t, r^t\}$, avec $r^t \in \{0, 1\}$
 - r^t pour un certain \mathbf{x}^t suit une distribution de Bernoulli selon une probabilité
 $y^t = P(C_1|\mathbf{x}^t)$

$$r^t | \mathbf{x}^t \sim \mathcal{B}(1, y^t)$$

- Vraisemblance de l'échantillonnage \mathcal{X} selon \mathbf{w}, w_0

$$I(\mathbf{w}, w_0 | \mathcal{X}) = \prod_t (y^t)^{(r^t)} (1 - y^t)^{(1 - r^t)}$$

- Erreur pour maximiser la log-vraisemblance

$$E_{\text{entr}}(\mathbf{w}, w_0 | \mathcal{X}) = -\log I(\mathbf{w}, w_0 | \mathcal{X}) = -\sum_t r^t \log y^t + (1 - r^t) \log(1 - y^t)$$

- Erreur $E_{\text{entr}}(\mathbf{w}, w_0 | \mathcal{X})$ également nommée *entropie croisée*

Minimisation de l'entropie croisée

- Dérivée de la fonction sigmoïde $y = f_{sig}(a) = \frac{1}{1+\exp(-a)}$

$$\begin{aligned}\frac{dy}{da} &= \frac{\exp(-a)}{[1 + \exp(-a)]^2} = \frac{1}{1 + \exp(-a)} \frac{\exp(-a) + 1 - 1}{1 + \exp(-a)} \\ &= \frac{1}{1 + \exp(-a)} \left(1 - \frac{1}{1 + \exp(-a)}\right) = y(1 - y)\end{aligned}$$

- Minimiser l'entropie croisée par descente du gradient

$$\Delta w_j = -\eta \frac{\partial E}{\partial w_j} = -\eta \frac{\partial E}{\partial y} \frac{\partial y}{\partial w_j} = \eta \sum_t \left(\frac{r^t}{y^t} - \frac{1 - r^t}{1 - y^t} \right) y^t (1 - y^t) x_j^t$$

$$= \eta \sum_t (r^t - y^t) x_j^t$$

$$\Delta w_0 = -\eta \frac{\partial E}{\partial w_0} = -\eta \frac{\partial E}{\partial y} \frac{\partial y}{\partial w_0} = \eta \sum_t (r^t - y^t)$$

Algorithme pour discrimination par régression logistique

1. Initialiser aléatoirement (uniformément distribués) les poids, $w_j \sim \mathcal{U}(-0,01, 0,01)$

$$w_j = \text{rand}(-0,01, 0,01), \quad j = 0, \dots, D$$

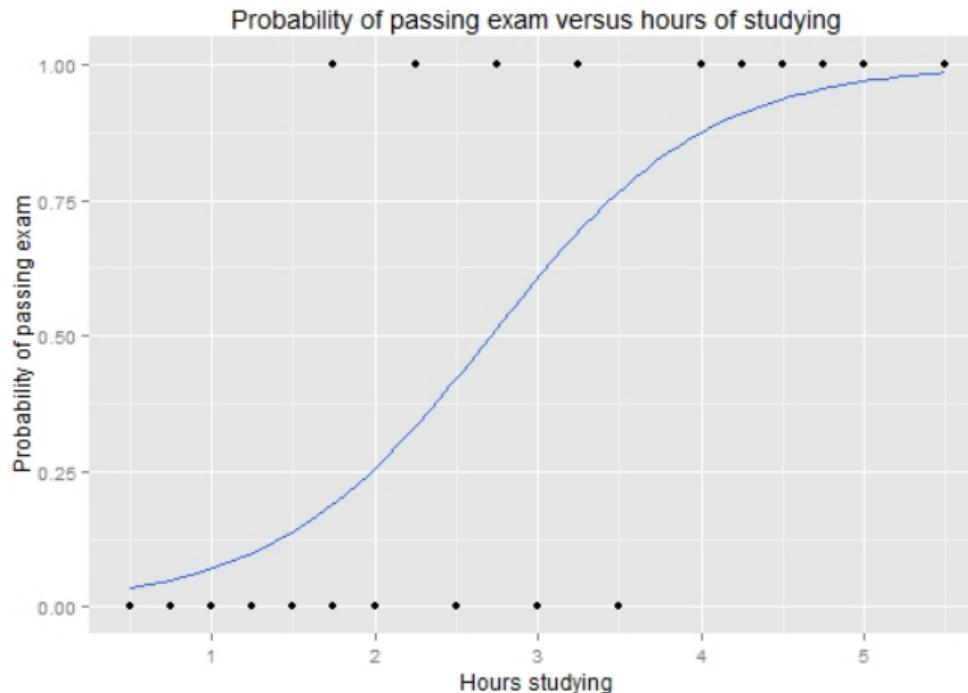
2. Répéter jusqu'à convergence :

$$y^t = \frac{1}{1 + \exp[-(\mathbf{w}^\top \mathbf{x}^t + w_0)]}, \quad t = 1, \dots, N$$

$$w_j = w_j + \eta \sum_t (r^t - y^t) x_j^t, \quad j = 1, \dots, D$$

$$w_0 = w_0 + \eta \sum_t (r^t - y^t)$$

Exemple de régression logistique



Critères de performance

- Différentes méthodes donnent différentes paramétrisations (\mathbf{w}, w_0)
 - Perceptron

$$E_{percp}(\mathbf{w}, w_0 | \mathcal{X}) = - \sum_{\mathbf{x}^t \in \mathcal{Y}} r^t h(\mathbf{x}^t | \mathbf{w}, w_0)$$

- \mathcal{Y} est l'ensemble des données de \mathcal{X} mal classées par $h(\mathbf{x}^t | \mathbf{w}, w_0)$
- Erreur quadratique

$$E_{quad}(\mathbf{w}, w_0 | \mathcal{X}) = \frac{1}{2} \sum_{\mathbf{x}^t \in \mathcal{X}} (r^t - h(\mathbf{x}^t | \mathbf{w}, w_0))^2$$

- Entropie croisée (régression logistique)

$$y = \frac{1}{1 + \exp[-h(\mathbf{x}^t | \mathbf{w}, w_0)]}$$

$$E_{entr}(\mathbf{w}, w_0 | \mathcal{X}) = - \sum_t r^t \log y^t + (1 - r^t) \log(1 - y^t)$$

Autres critères de performance

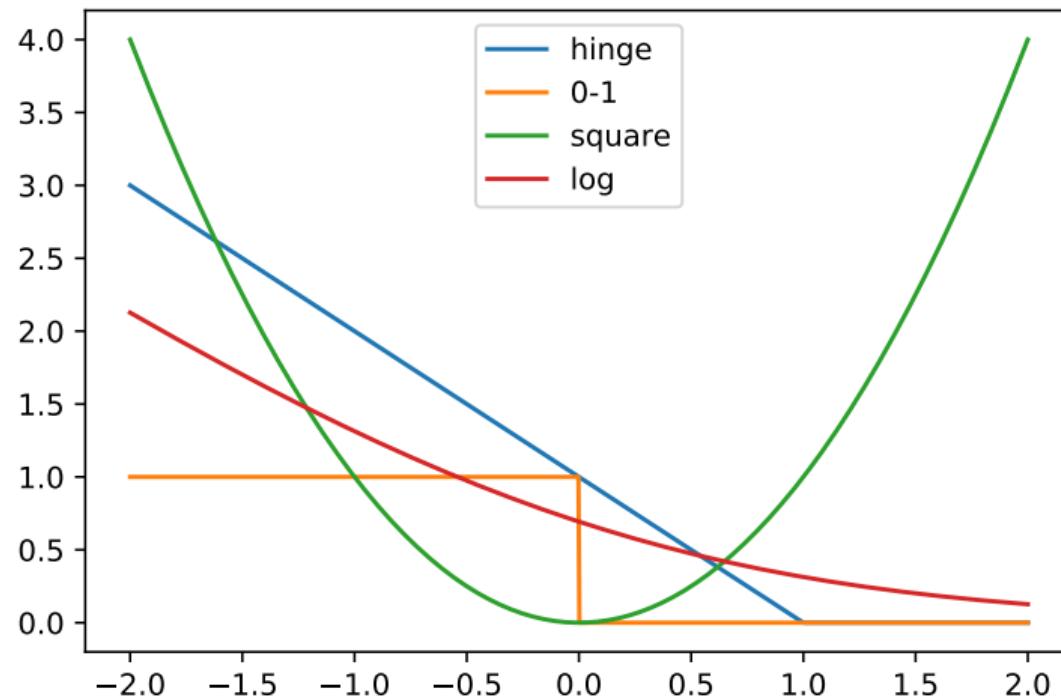
- Analyse discriminante linéaire : maximiser $J(w) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2}$
 - Séparer moyennes de classe m_i tout en réduisant variance de chaque classe s_i^2
- Fonction d'erreur *hinge*

$$E_{\text{hinge}}(\mathbf{w}, w_0 | \mathcal{X}) = \sum_{\mathbf{x}^t \in \mathcal{Y}} [1 - r^t h(\mathbf{x}^t | \mathbf{w}, w_0)]$$

- $\mathcal{Y} = \{\mathbf{x}^t \in \mathcal{X} \mid r^t h(\mathbf{x}^t | \mathbf{w}, w_0) \leq 1\}$
- \mathcal{Y} est l'ensemble des données de \mathcal{X} dans la *marge*
- Utilisée dans les SVM (présentés la semaine prochaine)
- Fonction de perte log

$$E_{\log}(\mathbf{w}, w_0 | \mathcal{X}) = \sum_{\mathbf{x}^t \in \mathcal{X}} \log [1 + \exp(-r^t h(\mathbf{x}^t | \mathbf{w}, w_0))]$$

Comparaison de différents critères d'erreurs



5.6 Normalisation et régularisation

Normalisation des poids

- Dans certaines circonstances, les valeurs des poids \mathbf{w} et w_0 peuvent exploser (ou imploser)
 - Plusieurs valeurs de poids donnent le même discriminant, seules valeurs relatives de \mathbf{w} et w_0 comptent pour le classement (signe de $h(\mathbf{x})$)
 - Corrections répétées ajoutent continuellement une valeur aux poids
 - Selon le critère, on obtient une erreur plus faible avec des poids faibles
 - Peut dépasser valeurs permises dans représentation des valeurs réelles sur un ordinateur (*overflow ou underflow*)
- Solutions possibles
 - Renormaliser les poids à chaque itération

$$\mathbf{w}' = \frac{\mathbf{w}}{\|[\mathbf{w} \ w_0]^\top\|}, \quad w'_0 = \frac{w_0}{\|[\mathbf{w} \ w_0]^\top\|}$$

- Normalisation dans critère d'erreur

$$E'(\mathbf{w}, w_0 | \mathbf{x}^t) = \frac{E(\mathbf{w}, w_0 | \mathbf{x}^t)}{\|\mathbf{x}^t\|^2}$$

Régression d'arête (régularisation l_2)

- Régression d'arête (*ridge regression* en anglais)
 - Limiter valeur des poids lors de l'optimisation

$$\text{minimiser} \quad E_{\text{quad}}(\mathbf{w}, w_0 | \mathcal{X}) = \frac{1}{2} \sum_{\mathbf{x}^t \in \mathcal{X}} (r^t - h(\mathbf{x}^t | \mathbf{w}, w_0))^2 \quad \text{sujet à} \quad \left(\sum_{i=1}^D w_i^2 \right) \leq \gamma$$

- Formulation équivalente (régularisation de Tychonoff)

$$E_{\text{ridge}} = \frac{1}{2} \sum_{\mathbf{x}^t \in \mathcal{X}} (r^t - h(\mathbf{x}^t | \mathbf{w}, w_0))^2 + \lambda \sum_{i=1}^D w_i^2$$

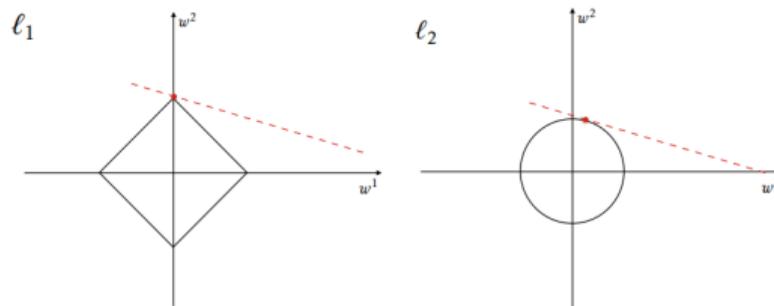
- Guide la recherche vers des modèles plus simples
 - Corrélation entre variables : valeur positive de w_i peut être annulée par valeur négative de w_j
- Exige que données soient normalisées et centrées à l'origine
- Peut être combiné à critères autres que l'erreur quadratique

LASSO (régularisation ℓ_1)

- LASSO : utiliser une régularisation ℓ_1 plutôt que ℓ_2

$$E_{LASSO} = \frac{1}{2} \sum_{\mathbf{x}^t \in \mathcal{X}} (r^t - h(\mathbf{x}^t | \mathbf{w}, w_0))^2 + \lambda \sum_{i=1}^D |w_i|$$

- N'est pas résoluble par dérivées partielles, exige méthodes telles que programmation quadratique
- Favorise élimination de variables (sélection de caractéristiques)



5.7 Modèles à plusieurs classes

Modèles à plusieurs classes

- À K classes, diverses stratégies possibles
 - Approche *un contre tous*
 - Approche *séparation par paires*
 - Autres approches (ex. code à correction d'erreur)
- Un contre tous
 - Une fonction discriminante par classe, $h_i(\mathbf{x}|\mathbf{w}_i, w_{i,0})$, $i = 1, \dots, K$
 - Option 1 : valeur maximale, $h(\mathbf{x}) = \underset{C_i=C_1}{\operatorname{argmax}} h_i(\mathbf{x})$
 - Option 2 : valeur positive uniquement

$$h(\mathbf{x}) = \begin{cases} C_i & \text{si } h_i(\mathbf{x}) \geq 0 \text{ et } h_j(\mathbf{x}) < 0, \forall i \neq j \\ \text{ambiguité} & \text{autrement} \end{cases}$$

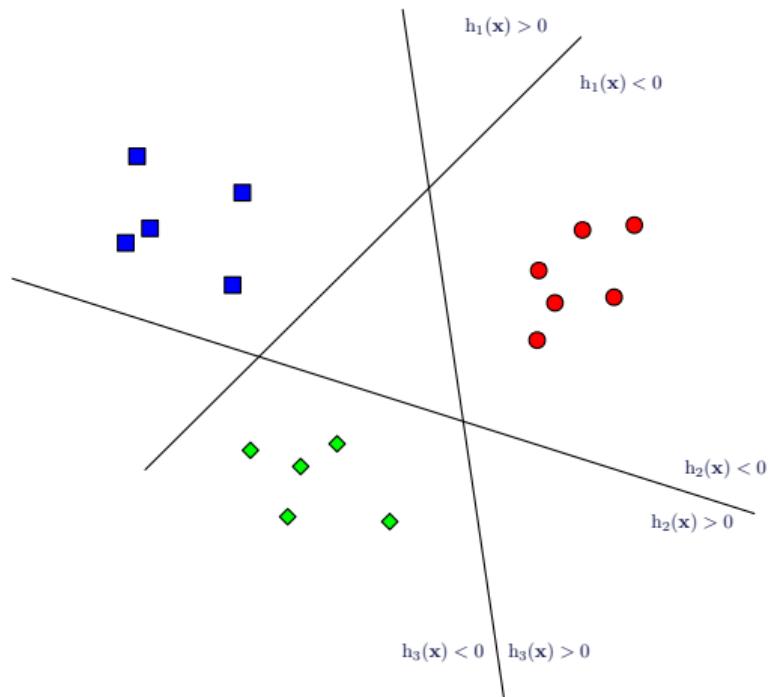
Modèles à plusieurs classes

- Séparation par paires
 - Un discriminant linéaire par chaque paire de classes,
 $h_{i,j}(\mathbf{x}|\mathbf{w}_{i,j}, w_{i,j,0})$, $i = 1, \dots, K - 1$, $j = i + 1, \dots, K$
 - Discriminant symétrique $h_{j,i}(\mathbf{x}) = -h_{i,j}(\mathbf{x})$, $j = 2, \dots, K$, $i = 1, \dots, j - 1$
 - Discriminants entraînés seulement sur les données de la classe C_i et C_j

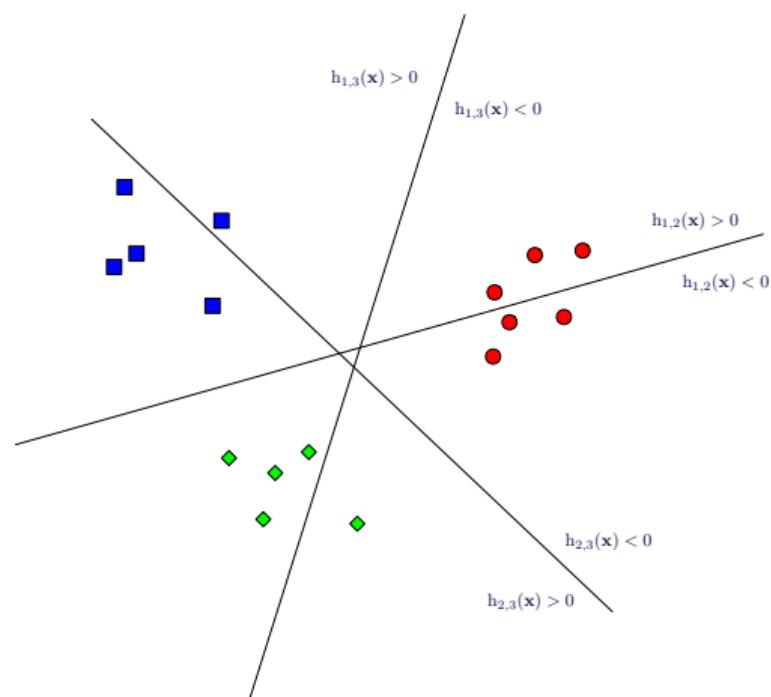
$$h_{i,j}(\mathbf{x}) = \begin{cases} \geq 0 & \text{si } \mathbf{x} \in C_i \\ < 0 & \text{si } \mathbf{x} \in C_j \\ \text{ignoré} & \text{autrement} \end{cases} \quad \text{avec } i = 1, \dots, K, j = i, \dots, K$$

- Évaluation de données : choisir C_i si $\forall j \neq i$, $h_{i,j} > 0$
- Relaxation possible : $h_i(\mathbf{x}) = \sum_{j \neq i} h_{i,j}(\mathbf{x})$

Frontières de décisions à plusieurs classes



Un contre tous



Séparation par paires

5.8 Apprentissage en ligne

Apprentissage en ligne et par lots

- Apprentissage par lots
 - Correction des poids une fois à chaque itération, en calculant l'erreur pour tout le jeu de données
 - Relative stabilité de l'apprentissage
- Apprentissage en ligne
 - Correction des poids pour chaque présentation de données, donc N corrections de poids par itération
 - Guidé par l'erreur sur chaque observation ($E(\mathbf{w}, w_0 | \mathbf{x}^t)$)
 - Requiert la permutation de l'ordre de traitement à chaque itération pour éviter les mauvaises séquences
 - Apprentissage en ligne est plus rapide que par lots, mais avec risque de plus grandes instabilités

Descente du gradient stochastique

- Descente du gradient stochastique
 - Aller plus loin que l'apprentissage en ligne : échantillonnage aléatoire avec remise de chaque donnée d'entraînement
 - Algorithme typique :
 1. Échantillonner aléatoirement (uniformément) une observation \mathbf{x}^t dans \mathcal{X} , $t \sim \mathcal{U}(1, N)$
 2. Déterminer la valeur du taux d'apprentissage, typiquement $\eta^t = 1/l$ où l est l'indice de la donnée actuelle selon son ordre de traitement
 3. Corriger les poids par descente du gradient
 - Requiert un ajustement décroissant du taux d'apprentissage pour chaque donnée
 - Intéressant pour traiter très grands jeux de données, en une passe
 - Permet aussi un arrêt de l'apprentissage à tout moment
 - Peut également être adapté au traitement de flux de données

5.9 Fonctions de base

Problème du XOR

- Problème du XOR

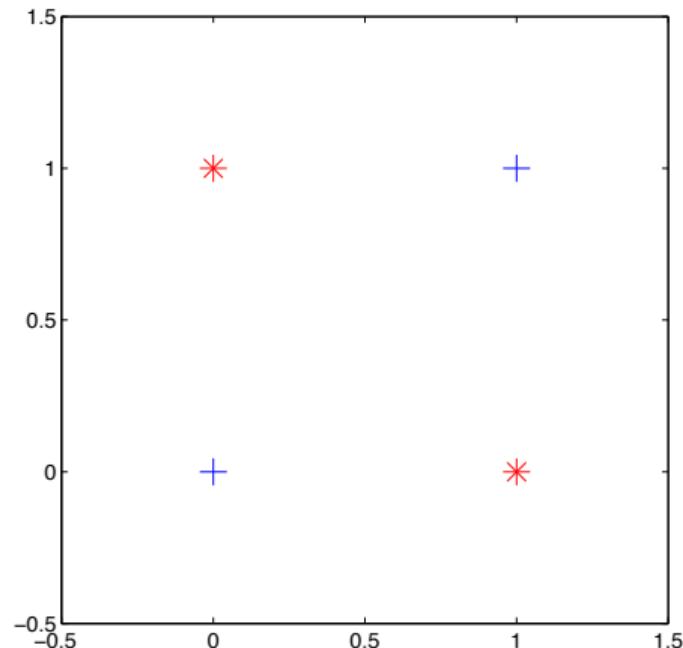
$$\mathbf{x}_1 = [0 \ 0]^\top \quad r_1 = 0$$

$$\mathbf{x}_2 = [0 \ 1]^\top \quad r_2 = 1$$

$$\mathbf{x}_3 = [1 \ 0]^\top \quad r_3 = 1$$

$$\mathbf{x}_4 = [1 \ 1]^\top \quad r_4 = 0$$

- Exemple de données non linéairement séparables



Fonctions de base

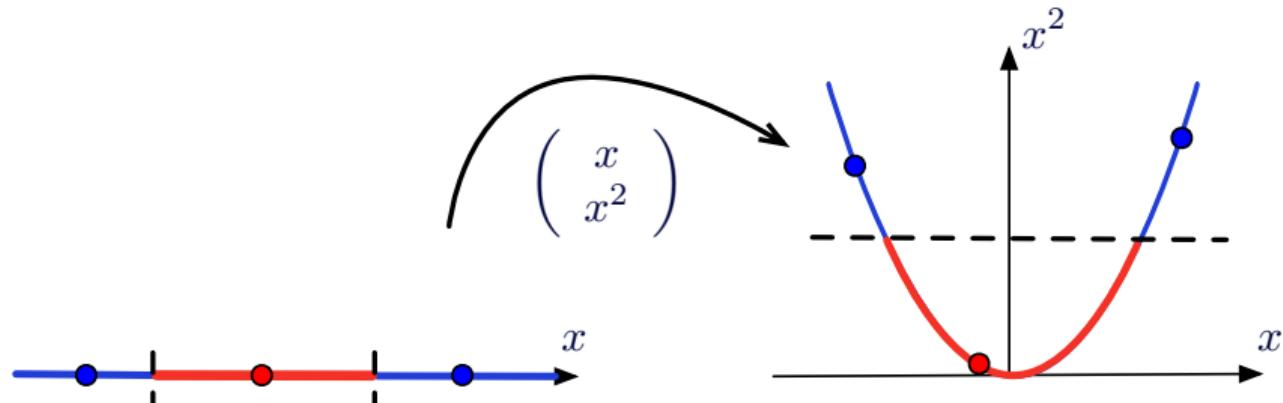
- Discriminant avec fonction de base
 - Transformation non linéaire $\phi : \mathbb{R}^D \rightarrow \mathbb{R}^K$ traitée sous une forme linéaire

$$h_i(\mathbf{x}) = \sum_{j=1}^K w_j \phi_{i,j}(\mathbf{x}) + w_0$$

- Exemple de fonctions de base

- $\phi_{i,j}(\mathbf{x}) = x_j$
- $\phi_{i,j}(\mathbf{x}) = x_1^{j-1}$
- $\phi_{i,j}(\mathbf{x}) = \exp(-(x_2 - m_j)^2/c)$
- $\phi_{i,j}(\mathbf{x}) = \exp(-\|\mathbf{x} - \mathbf{m}_j\|^2/c)$
- $\phi_{i,j}(\mathbf{x}) = \text{sgn}(x_j - c_j)$

Projection avec une fonction de base



- En 1D : non linéairement séparable
- Avec projection en 2D : linéairement séparable

Fonctions de base

- Résolution du XOR avec fonction de base $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$

$$\phi(\mathbf{x}) = [x_1 \ x_2 \ (x_1 x_2)]^\top$$

- Résultats de la transformation

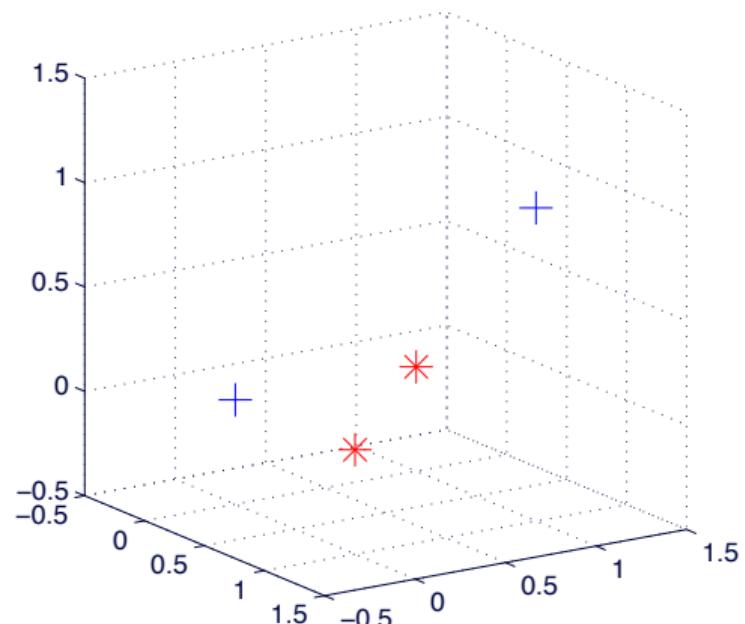
$$\mathbf{z}_1 = [0 \ 0 \ 0]^\top \quad r_1 = 0$$

$$\mathbf{z}_2 = [0 \ 1 \ 0]^\top \quad r_2 = 1$$

$$\mathbf{z}_3 = [1 \ 0 \ 0]^\top \quad r_3 = 1$$

$$\mathbf{z}_4 = [1 \ 1 \ 1]^\top \quad r_4 = 0$$

- Données linéairement séparables dans le nouvel espace !



5.10 Discriminants linéaires dans scikit-learn

Scikit-learn : modèles linéaires

- `discriminant_analysis.LinearDiscriminantAnalysis` : méthodes paramétriques pour générer discriminants linéaires
- `linear_model.LinearRegression` : régression linéaire par moindres carrés
- `linear_model.Ridge` : régression d'arête (moindres carrés + régularisation l_2)
- `linear_model.Lasso` : LASSO (moindres carrés + régularisation l_1)
- `linear_model.LogisticRegression` : régression logistique
- `linear_model.Perceptron` : discriminant linéaire entraîné par la règle du perceptron
- `linear_model.SGDClassifier` et `linear_model.SGDRegressor` : modèles linéaires entraînés par descente du gradient stochastique

Scikit-learn : gestion du multiclasse

- Classifieurs de scikit-learn gèrent le multiclasse *out-of-the-box*
- Modèles pour gestion particulière du multiclasse
 - `multiclass.OneVsRestClassifier` : approche un contre tous
 - `multiclass.OneVsOneClassifier` : séparation par paires
 - `multiclass.OutputCodeClassifier` : codes à correction d'erreur (vu dans présentation sur méthodes par ensemble)