

Introduction aux réseaux de neurones

Fabrice Rossi

Projet Axis, INRIA Rocquencourt

2007

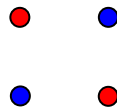
Plan

- 1 Méthodes linéaires sur des données transformées
 - Linéarité
 - Transformation des données
 - Réseaux de neurones RBF
 - Machines à vecteurs de support

Problèmes non linéaires

Trois sources de non linéarité :

- 1 le bruit : mauvais label associé à un exemple
- 2 les variables manquantes
- 3 le problème lui-même (intrinsèquement non linéaire)
 - Exemple du ou exclusif :

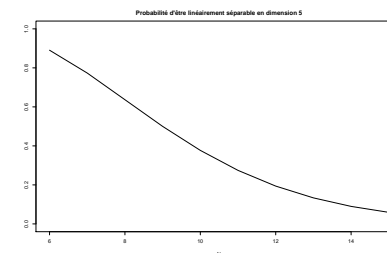


- Généralisable en dimension quelconque (problème de parité).

Linéarité et dimension

Résultats de Thomas Cover (1965)

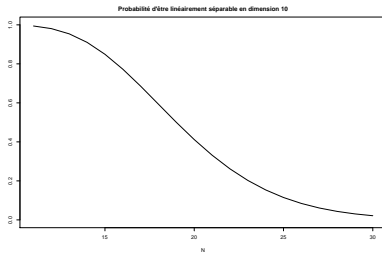
- la « linéarité » d'un problème dépend de la dimension
- l'espérance du nombre maximum de points linéairement séparable en dimension p est $2p$
- l'espérance du nombre minimal de variables nécessaires pour séparer linéairement N point est $\frac{N+1}{2}$
- distribution de plus en plus « piquée » :



Linéarité et dimension

Résultats de Thomas Cover (1965)

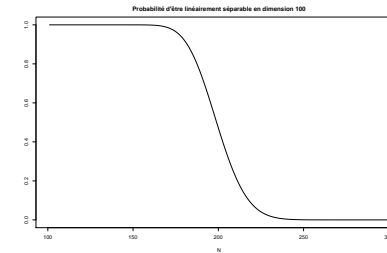
- la « linéarité » d'un problème dépend de la dimension
- l'espérance du nombre maximum de points linéairement séparable en dimension p est $2p$
- l'espérance du nombre minimal de variables nécessaires pour séparer linéairement N point est $\frac{N+1}{2}$
- distribution de plus en plus « piquée » :



Linéarité et dimension

Résultats de Thomas Cover (1965)

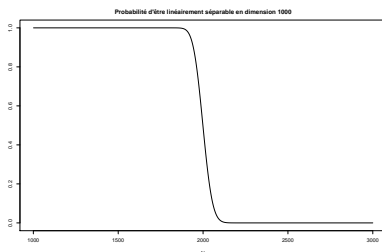
- la « linéarité » d'un problème dépend de la dimension
- l'espérance du nombre maximum de points linéairement séparable en dimension p est $2p$
- l'espérance du nombre minimal de variables nécessaires pour séparer linéairement N point est $\frac{N+1}{2}$
- distribution de plus en plus « piquée » :



Linéarité et dimension

Résultats de Thomas Cover (1965)

- la « linéarité » d'un problème dépend de la dimension
- l'espérance du nombre maximum de points linéairement séparable en dimension p est $2p$
- l'espérance du nombre minimal de variables nécessaires pour séparer linéairement N point est $\frac{N+1}{2}$
- distribution de plus en plus « piquée » :



Conséquences

- problèmes « simples » :
 - $\frac{p}{N} \gg 2$: beaucoup de variables pour peu d'observations
 - classifieur linéaire :
 - généralement une infinité de choix possibles
 - critère de choix très important
- problèmes « difficiles » :
 - $\frac{p}{N} \ll 2$
 - pas de séparateur linéaire
 - peu de variables et/ou beaucoup d'observations
 - données « contradictoires » (classes partiellement superposées)

Deux stratégies

- classifieur « complexe » sur les données d'origines :
 - combinaisons de neurones de McCulloch et Pitts (perceptron multi-couches)
 - neurones d'ordre supérieur (par exemple quadratiques)
- classifieur simple (linéaire) sur les données transformées :
 - image des données par une fonction non linéaire
 - modèle linéaire dans l'espace image
- une différence fondamentale : adaptation (ou non) de la partie non linéaire du modèle

Transformation explicite des données

- Principe :
 - fonction de transformation $\phi : \mathbb{R}^p \rightarrow \mathcal{H}$
 - \mathcal{H} peut être, par exemple, \mathbb{R}^q
 - on construit un classifieur linéaire dans \mathcal{H} sur les $\phi(\mathbf{x}_i)$
 - idée sous-jacente : choisir q grand pour obtenir un problème linéairement séparable
- Difficultés :
 - choix de ϕ
 - travail en dimension élevée :
 - temps de calcul
 - stabilité numérique
 - choix du séparateur linéaire :
 - analyse discriminante de Fisher
 - régression linéaire

Dimension un

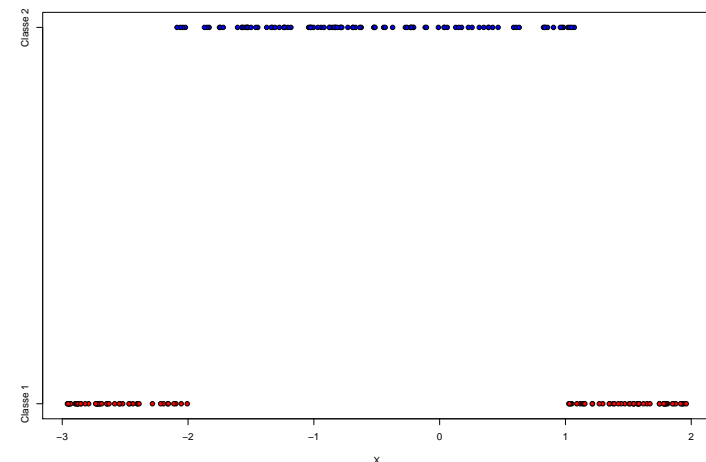
Solutions possibles pour $p = 1$:

- modèle linéaire : $\text{signe}(x - b)$
- modèle polynômial : $\text{signe}\left(\sum_{k=0}^d w_k x^k\right)$
- série de Fourier :

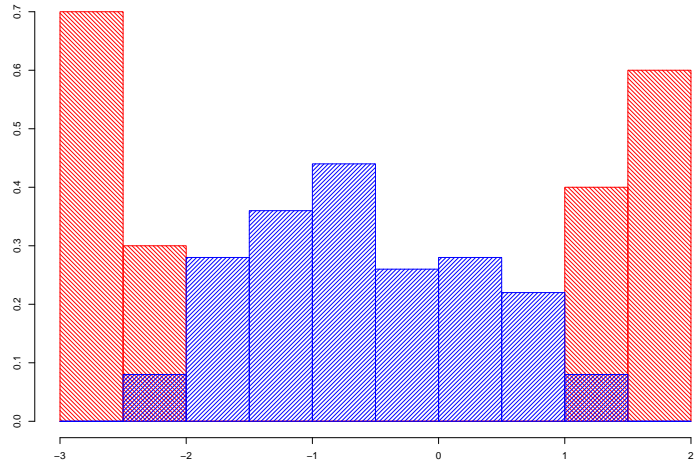
$$\text{signe}\left(w_0 + \sum_{k=1}^d \left(w_k \sin\left(\frac{2\pi k}{T} x\right) + w'_k \cos\left(\frac{2\pi k}{T} x\right)\right)\right)$$

- nombreuses autres solutions (splines, ondelettes, etc.)

Exemple (régression linéaire)



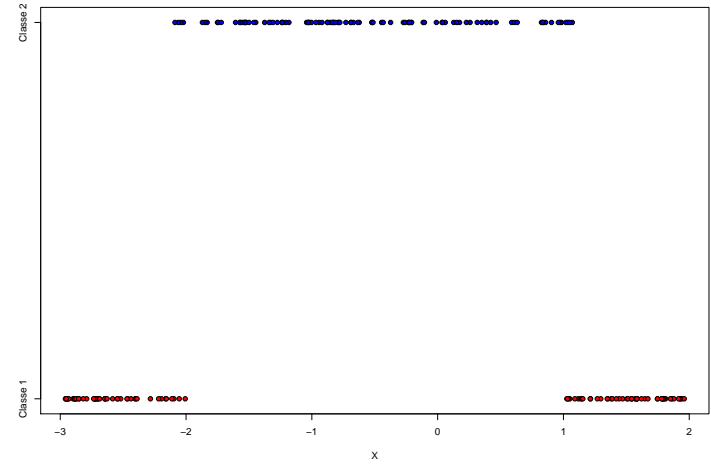
Exemple (régression linéaire)



Navigation icons

Exemple (régression linéaire)

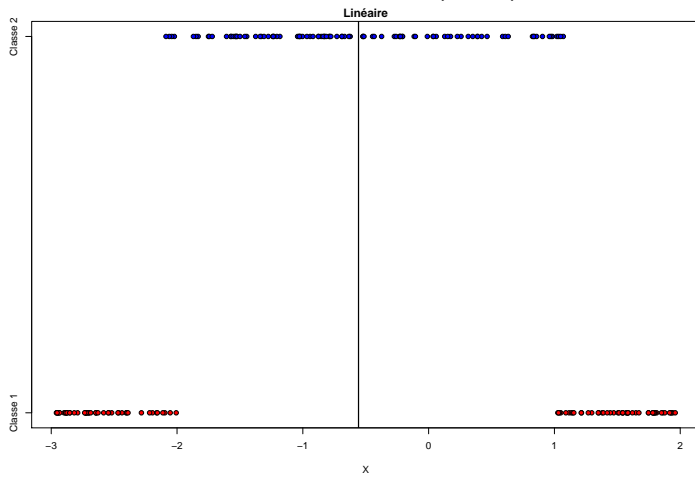
Modèle linéaire : $\text{signe}(x + c)$



Navigation icons

Exemple (régression linéaire)

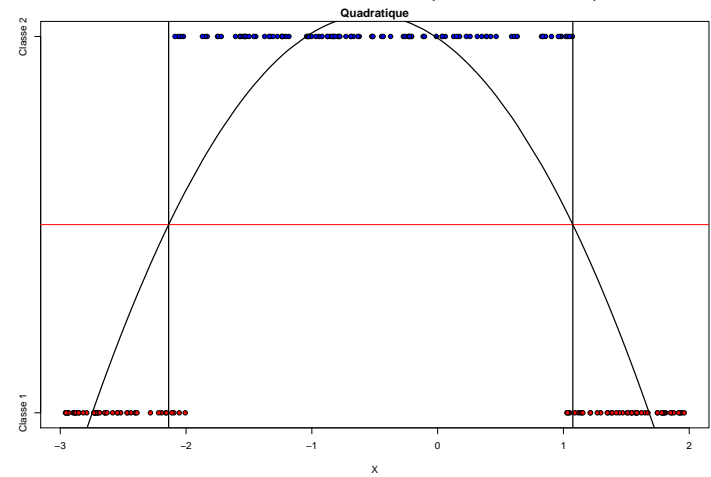
Modèle linéaire : $\text{signe}(x + c)$



Navigation icons

Exemple (régression linéaire)

Modèle quadratique : $\text{signe}(ax^2 + bx + c)$



Navigation icons

Dimension un

- polynômes :
 - degré d : d racines et donc $d + 1$ « zones »
 - d élevé : instabilité numérique
 - ne pas utiliser la base des X^k mais plutôt les polynômes de Lagrange ou Bernstein
- splines :
 - fonctions polynomiales par morceaux
 - cas classique : degré 3 sur chaque morceau
 - base : B-splines
 - très stables numériquement

Navigation icons

Dimension faible

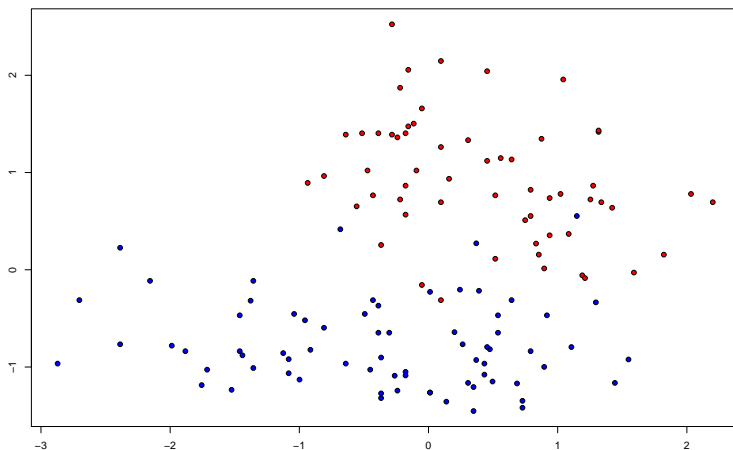
- pour $p = 2$ et $p = 3$, on peut étendre les solutions acceptables pour $p = 1$
- polynômes :
 - nombre de monômes de degré inférieur à d : $\frac{(p+d)!}{p!d!}$
 - acceptable pour p ou d très faible
- Exemple :
 - cas du où exclusif
 - $\phi(x_1, x_2) = (x_1, x_2, x_1 x_2)$

x_1	x_2	$x_1 x_2$	y
0	0	0	0
1	1	1	0
0	1	0	1
1	0	0	1

- linéairement séparable : $y = \phi(\mathbf{x})_1 + \phi(\mathbf{x})_2 - 2\phi(\mathbf{x})_3$

Navigation icons

Exemple



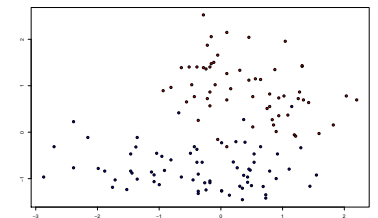
Navigation icons

Exemple

Méthode quadratique :

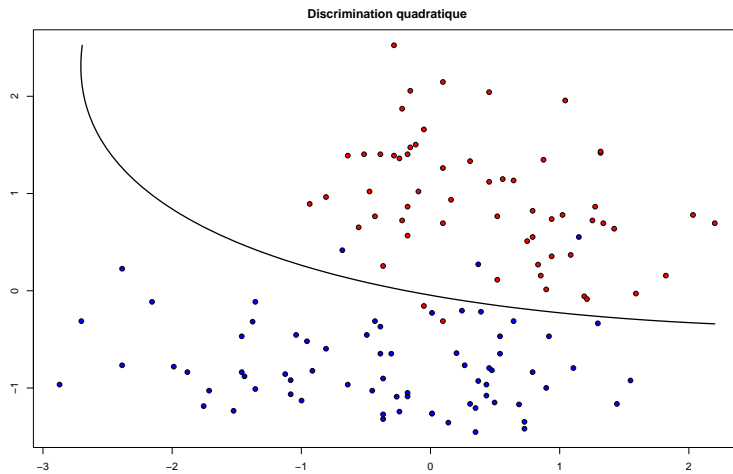
- $\phi(x_1, x_2) = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2)$
- séparateur « linéaire » : $\langle \mathbf{w}, \phi(x_1, x_2) \rangle$
- solution quadratique :

$$(x_1, x_2) \mapsto a + bx_1 + cx_2 + dx_1^2 + ex_1 x_2 + fx_2^2$$



Navigation icons

Exemple



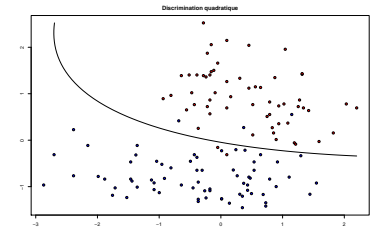
Exemple

Méthode quadratique :

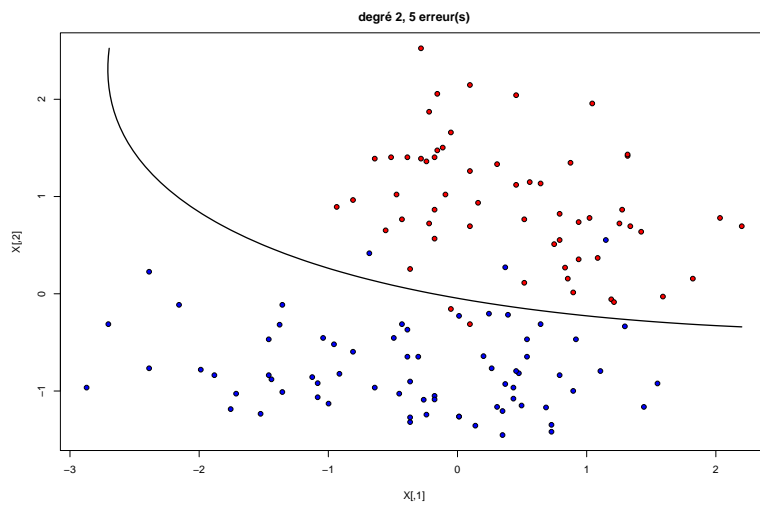
- $\phi(x_1, x_2) = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2)$
- séparateur « linéaire » : $\langle \mathbf{w}, \phi(x_1, x_2) \rangle$
- solution quadratique :

$$(x_1, x_2) \mapsto a + bx_1 + cx_2 + dx_1^2 + ex_1x_2 + fx_2^2$$

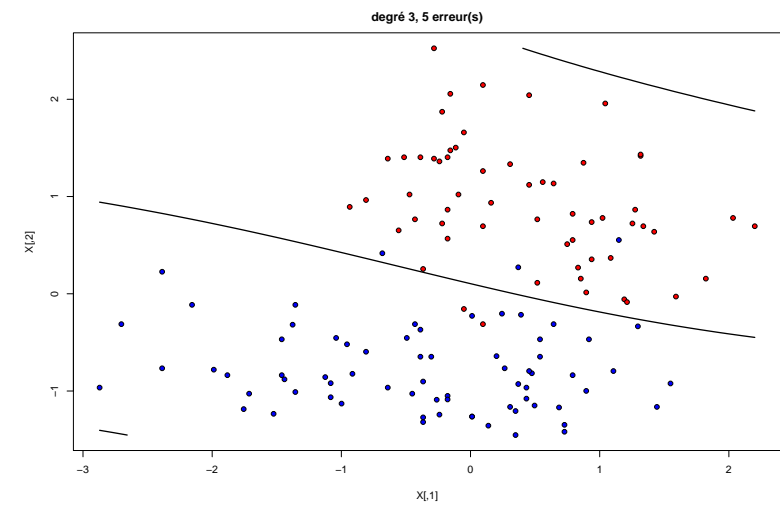
- linéaire : 5 erreurs
- quadratique : 5 erreurs



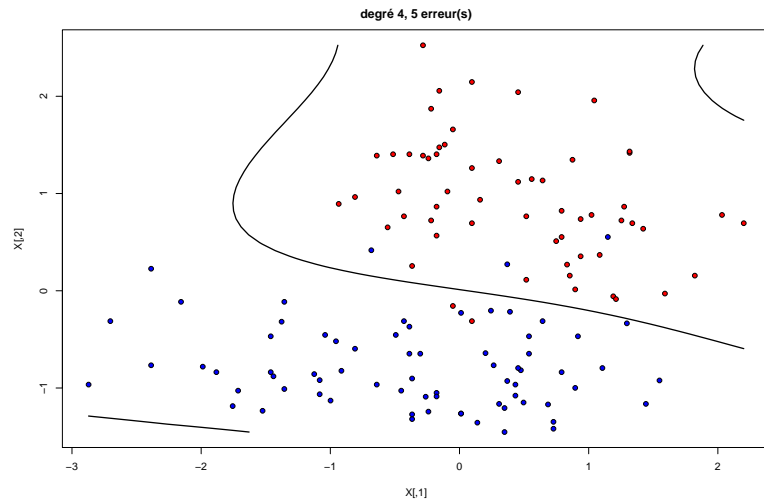
Complexité et degré



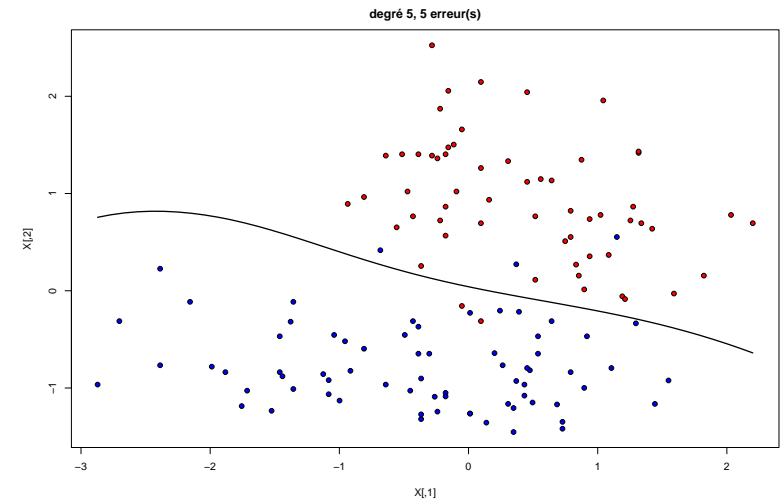
Complexité et degré



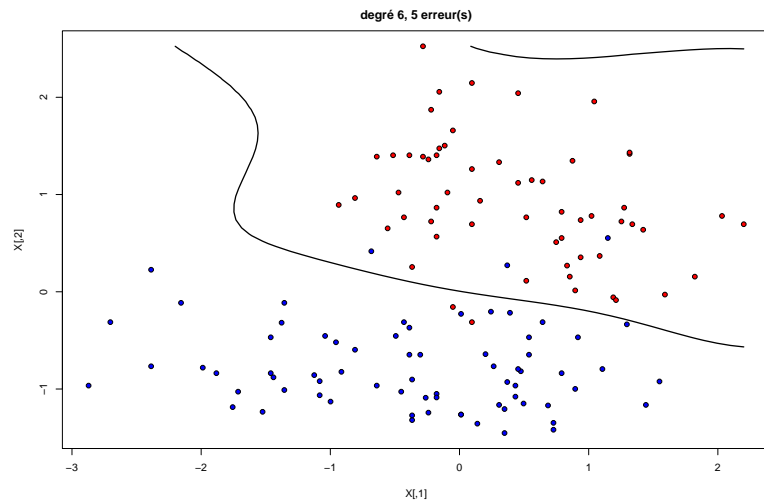
Complexité et degré



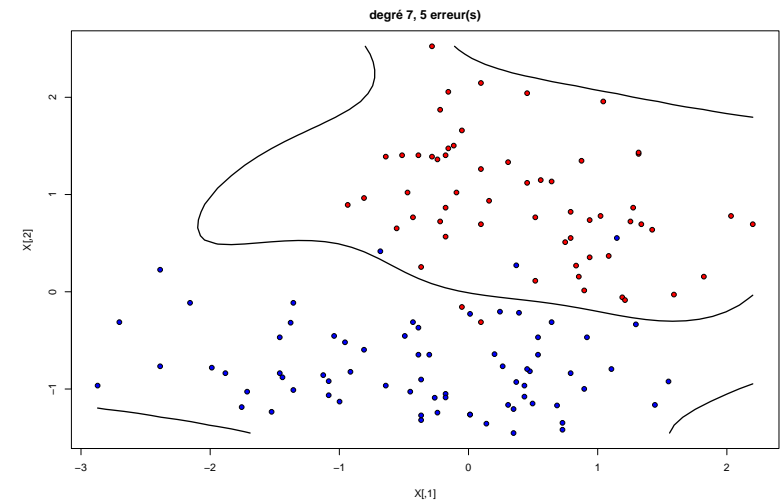
Complexité et degré



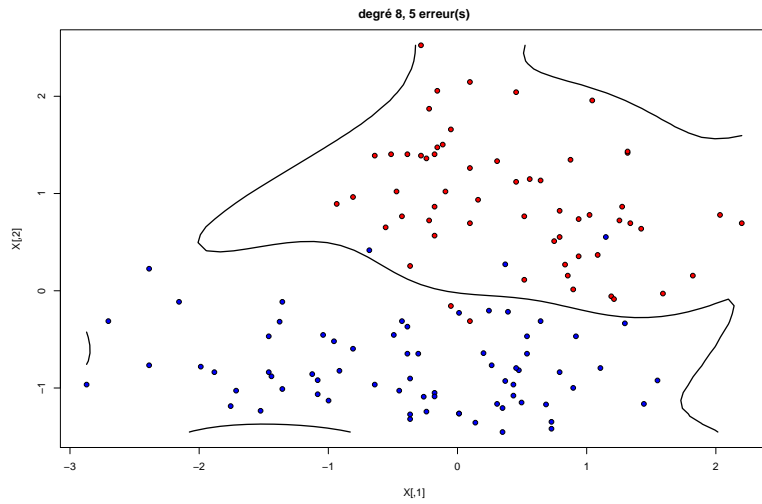
Complexité et degré



Complexité et degré

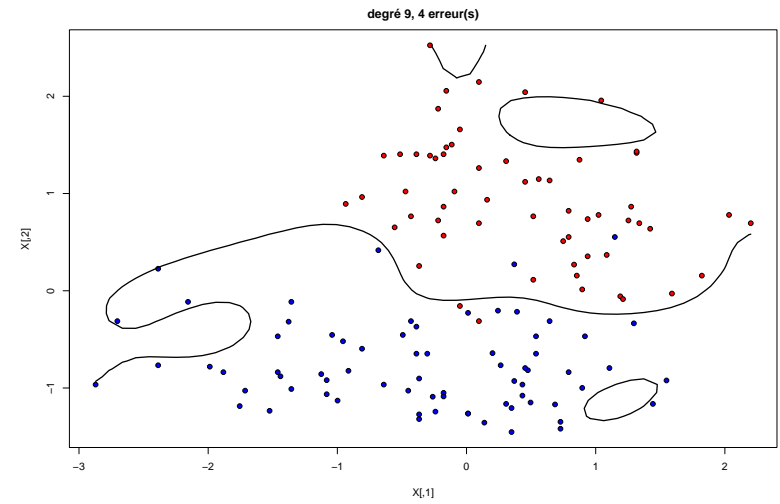


Complexité et degré



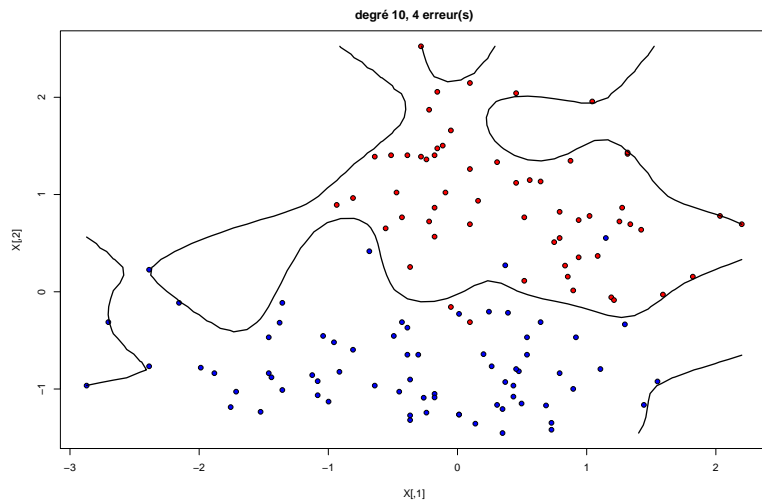
Navigation icons

Complexité et degré



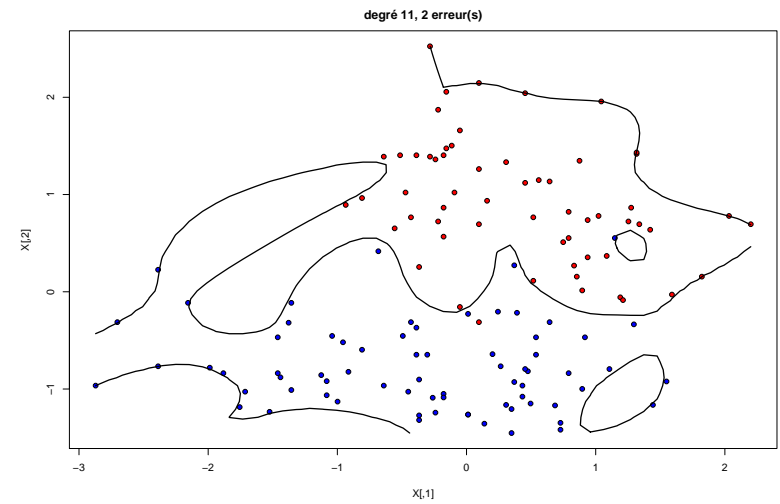
Navigation icons

Complexité et degré



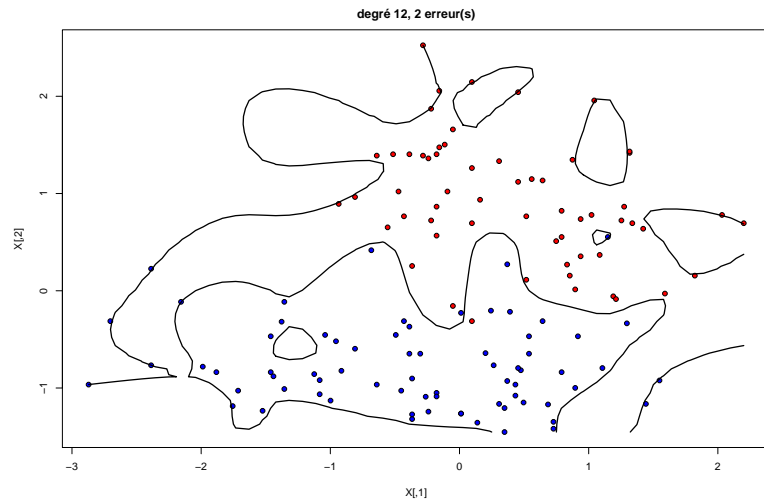
Navigation icons

Complexité et degré

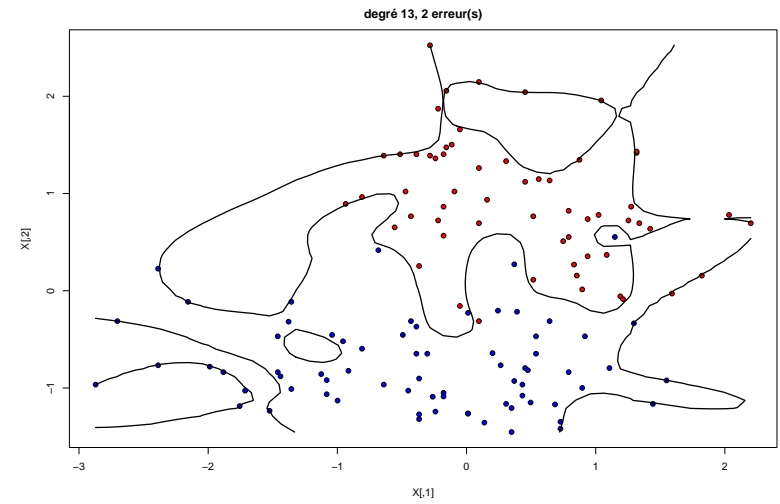


Navigation icons

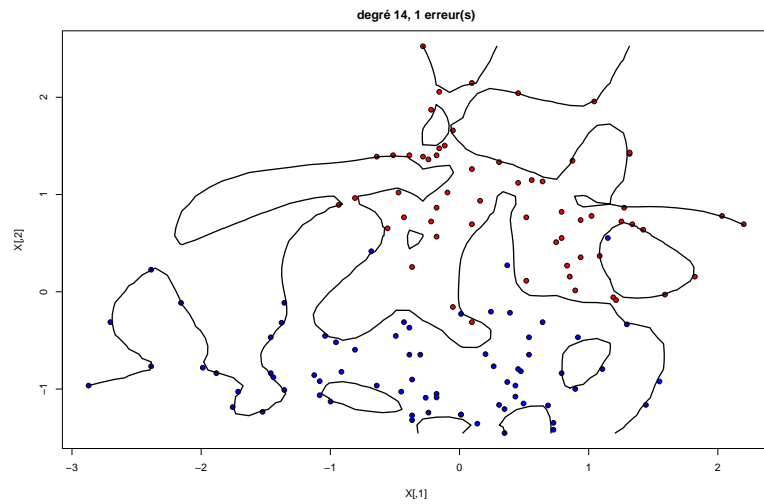
Complexité et degré



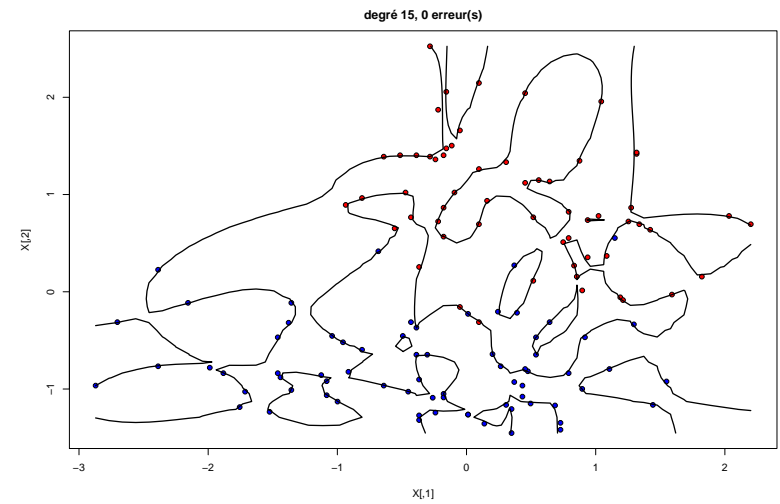
Complexité et degré



Complexité et degré



Complexité et degré



« Grandes » dimensions

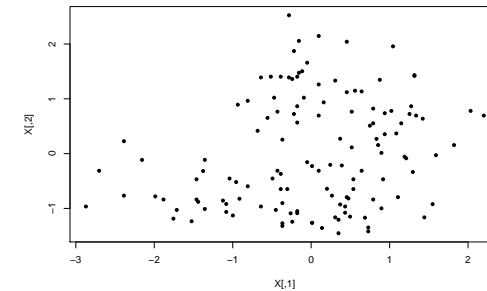
- les solutions polynomiales simples ne sont pas acceptables : trop de coefficients (sauf quadratique)
- une solution, neurones à base radiale :
 - une fonction $\zeta : \mathbb{R}^+ \rightarrow \mathbb{R}^+ :$
 - décroissante
 - $\lim_{u \rightarrow \infty} \zeta(u) = 0$
 - des points dans \mathbb{R}^p , ($\mathbf{c}_1, \dots, \mathbf{c}_q$) (les **centres**)
 - transformation

$$\phi(\mathbf{x}) = \left(\zeta \left(\|\mathbf{x} - \mathbf{c}_1\|^2 \right), \dots, \zeta \left(\|\mathbf{x} - \mathbf{c}_q\|^2 \right) \right)$$

- en général $\zeta(u) = \exp\left(-\frac{u}{2\sigma^2}\right)$ (σ : paramètre de la transformation)
- choix délicats :
 - nombre et positions des ($\mathbf{c}_1, \dots, \mathbf{c}_q$)
 - valeur de σ (peut dépendre du point \mathbf{c}_k)

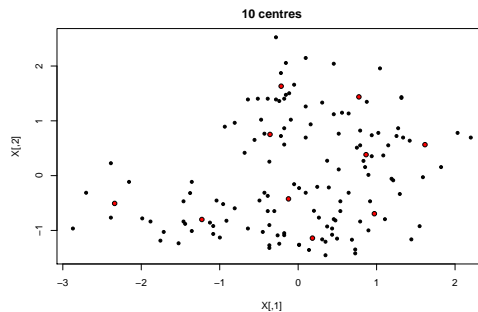
Choix des centres

- nombre : puissance de discrimination (même principe que le degré)
- position :
 - points « bien répartis »
 - problème de *quantification* : bien représenter un ensemble de points autre ensemble de points en moins grand nombre



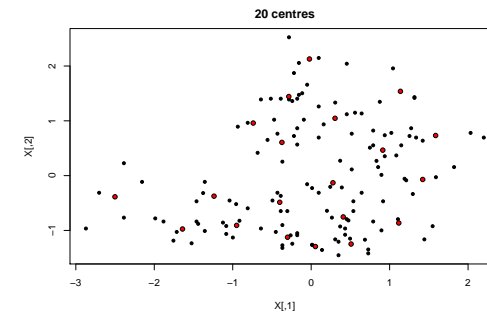
Choix des centres

- nombre : puissance de discrimination (même principe que le degré)
- position :
 - points « bien répartis »
 - problème de *quantification* : bien représenter un ensemble de points autre ensemble de points en moins grand nombre



Choix des centres

- nombre : puissance de discrimination (même principe que le degré)
- position :
 - points « bien répartis »
 - problème de *quantification* : bien représenter un ensemble de points autre ensemble de points en moins grand nombre



K-means

Une solution simple pour le choix des centres :

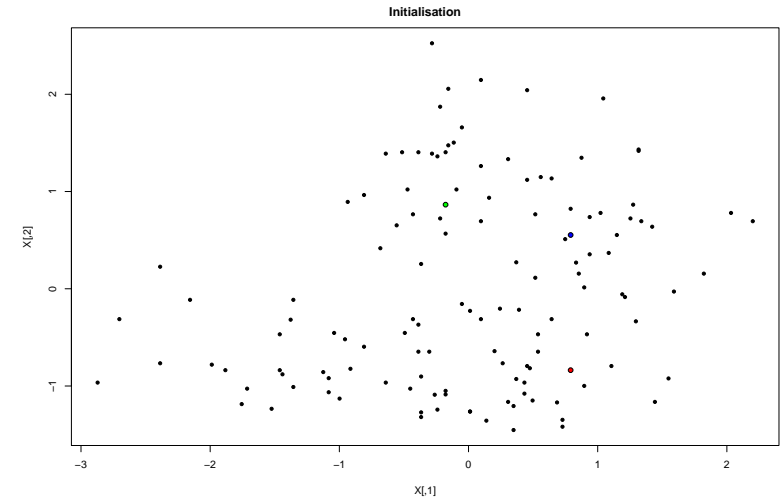
- **méthode de classification**
- erreur de quantification :

$$\sum_{i=1}^N \min_{1 \leq k \leq q} \|\mathbf{x}_i - \mathbf{c}_k\|^2$$

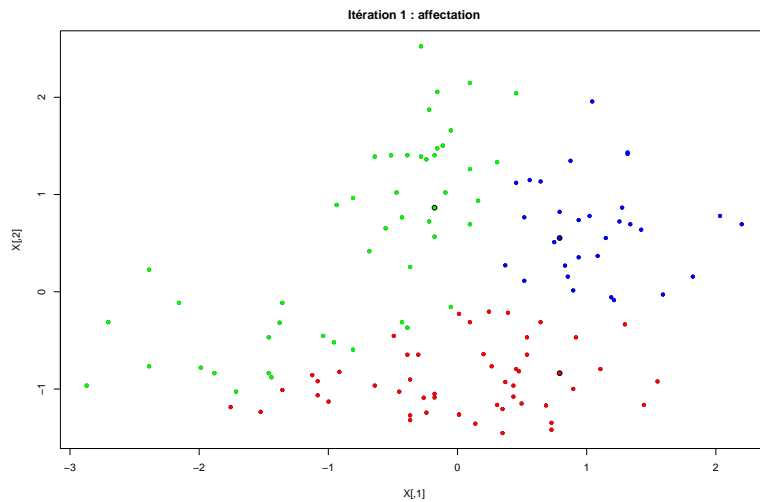
• algorithme :

- 1 valeurs initiales aléatoires pour les \mathbf{c}_k
- 2 $C_k = \{ \mathbf{x}_j \mid \|\mathbf{x}_j - \mathbf{c}_k\|^2 \leq \|\mathbf{x}_j - \mathbf{c}_j\|^2, j \neq k \}$
- 3 $\mathbf{c}_k = \frac{1}{|C_k|} \sum_{\mathbf{x}_j \in C_k} \mathbf{x}_j$
- 4 retour en 2 jusqu'à convergence

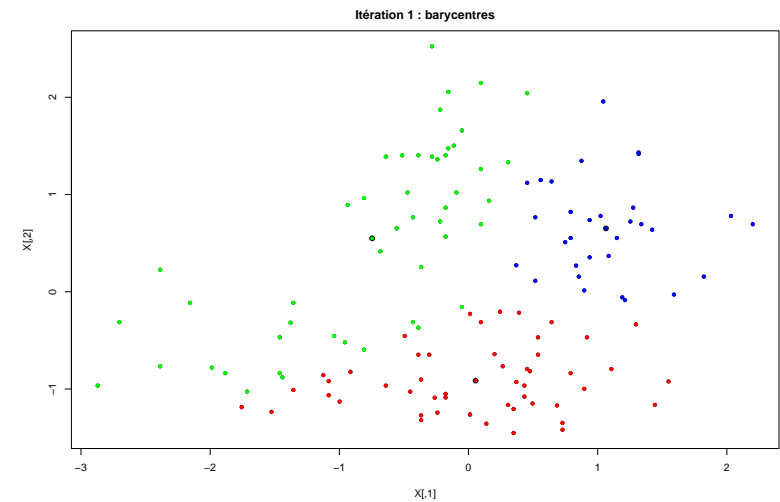
Exemple



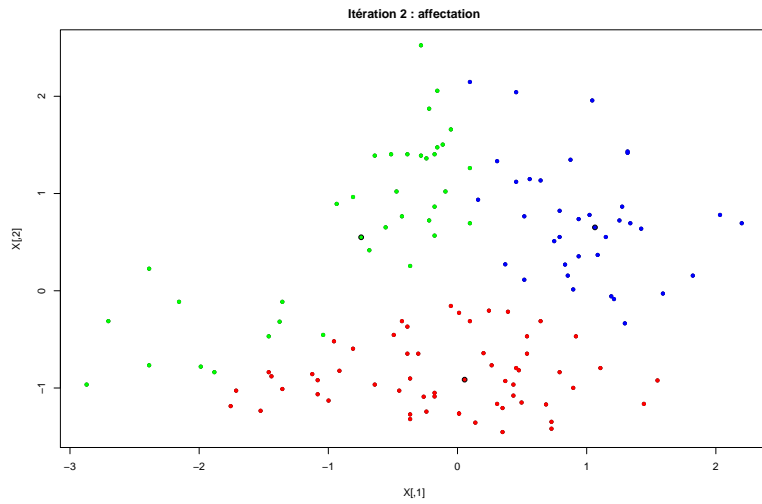
Exemple



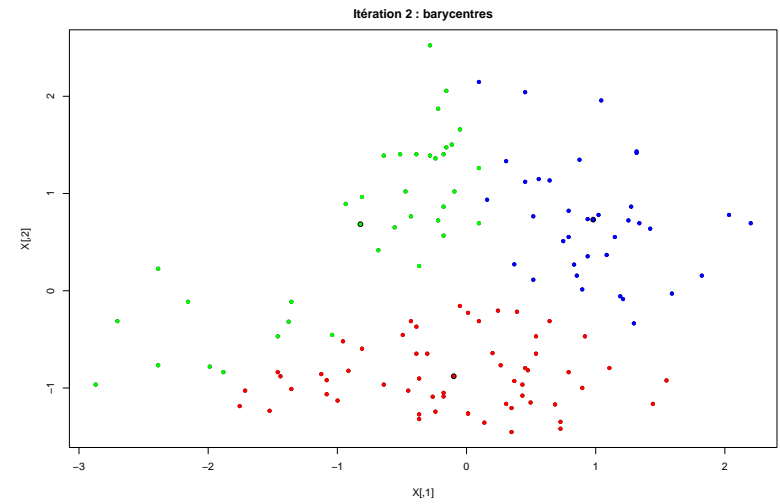
Exemple



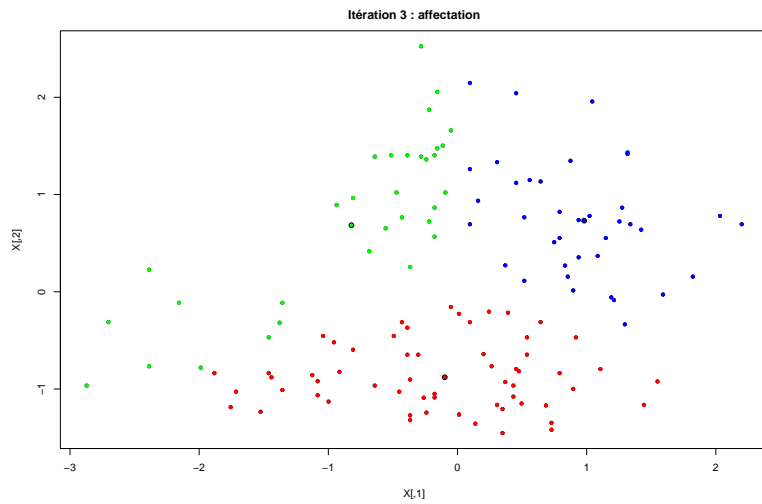
Exemple



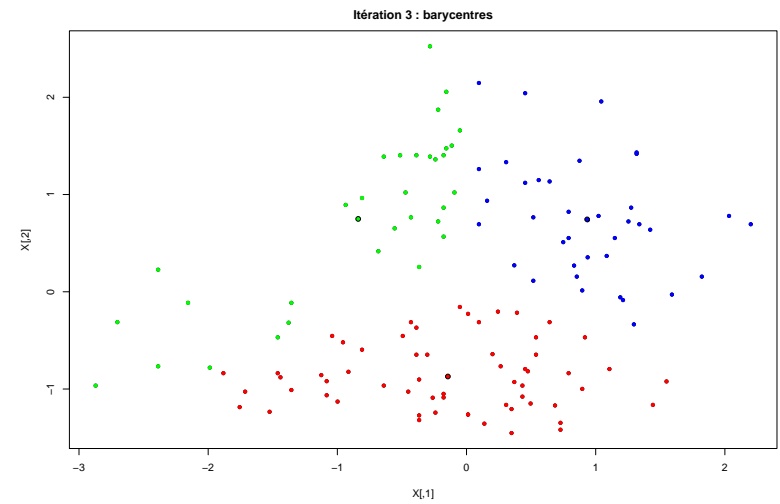
Exemple



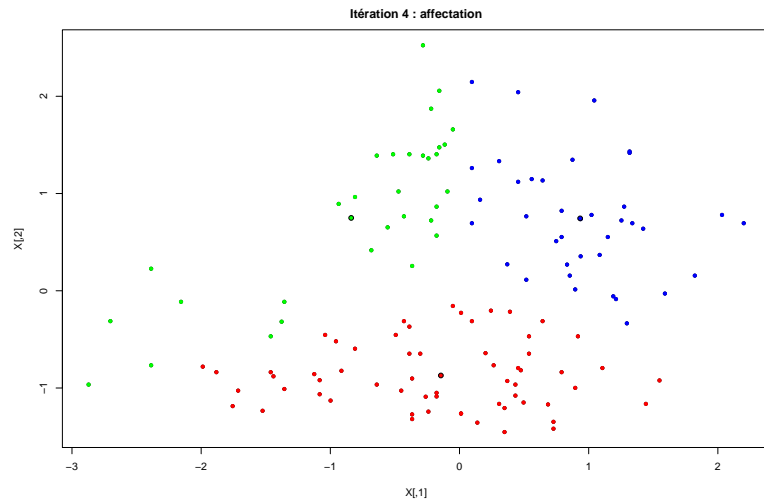
Exemple



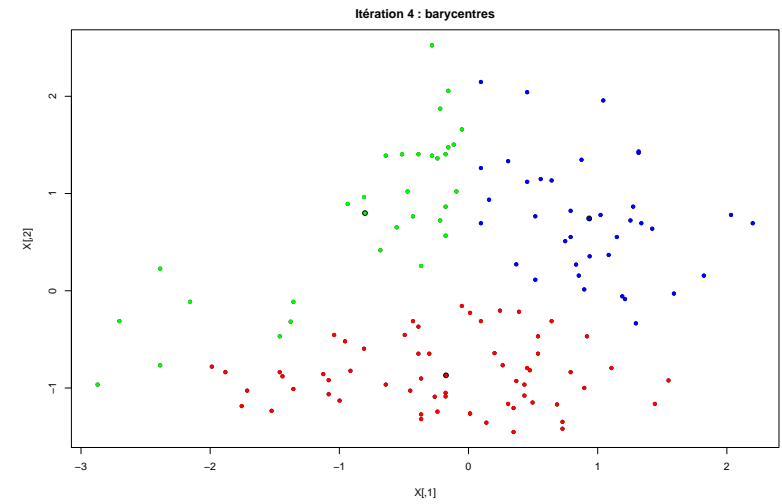
Exemple



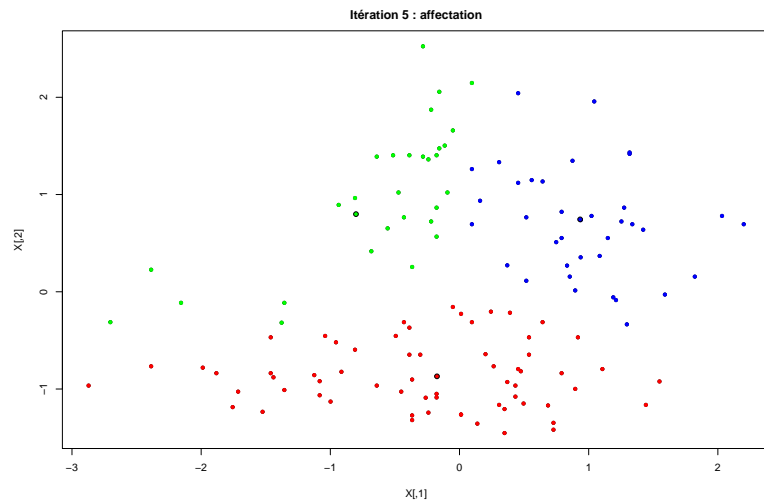
Exemple



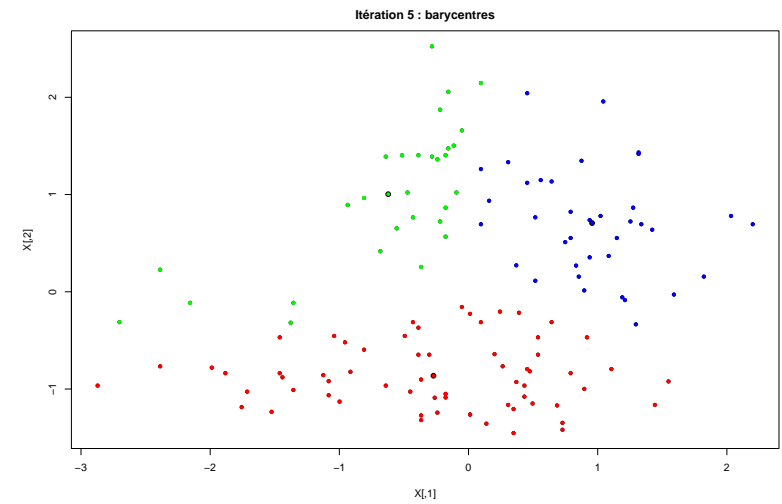
Exemple



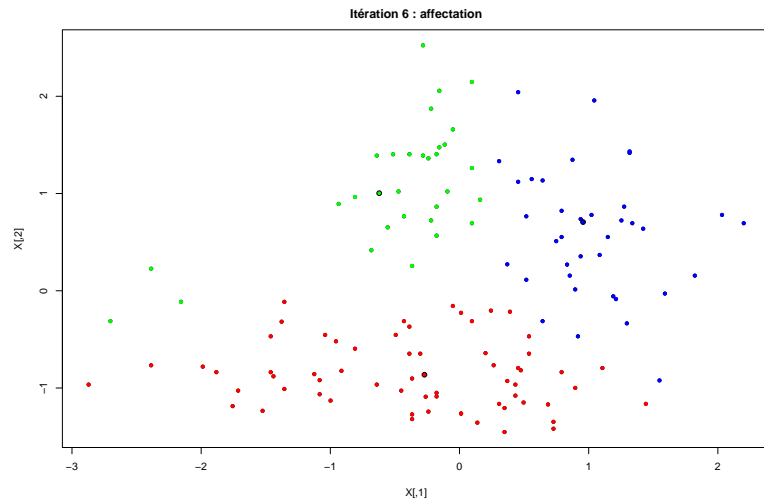
Exemple



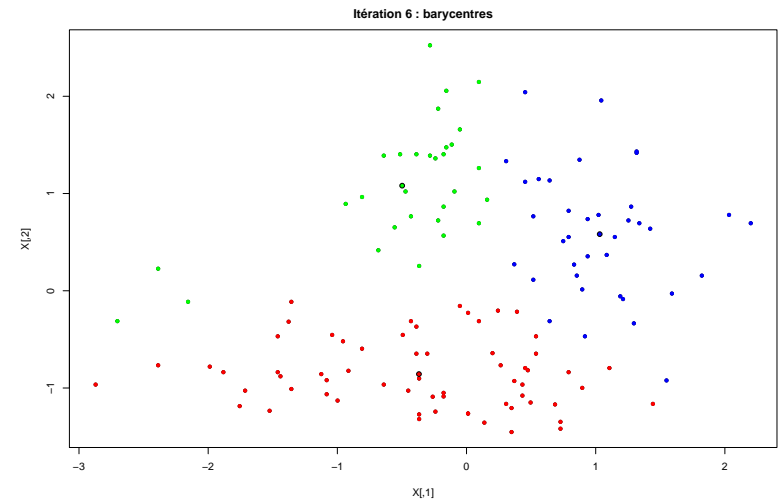
Exemple



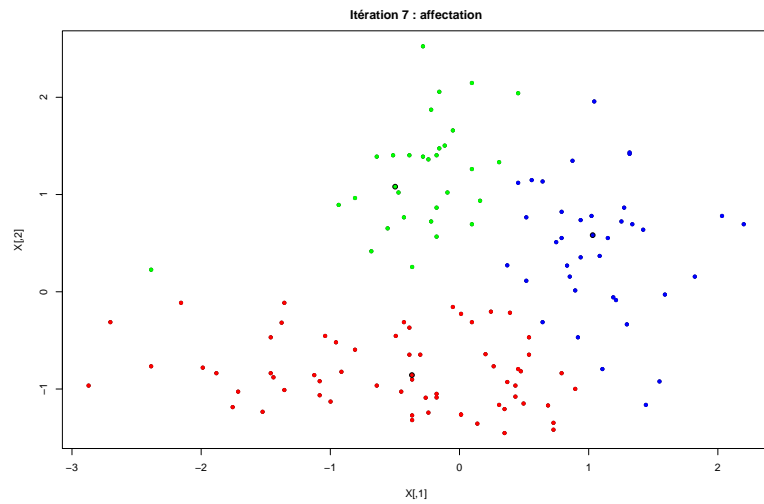
Exemple



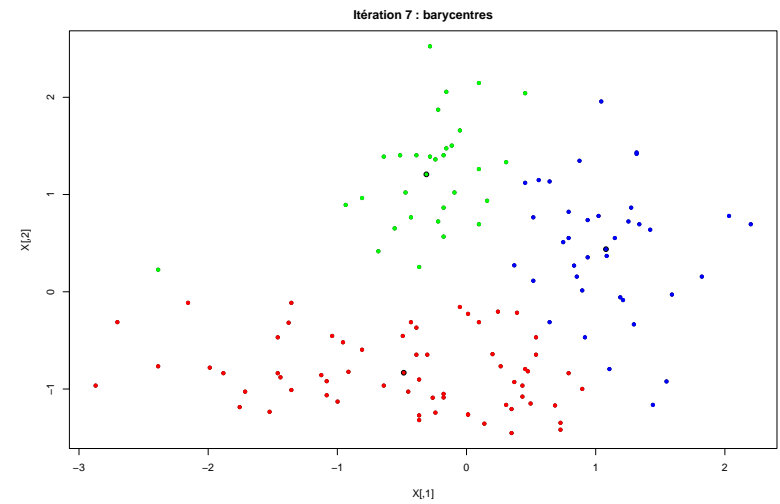
Exemple



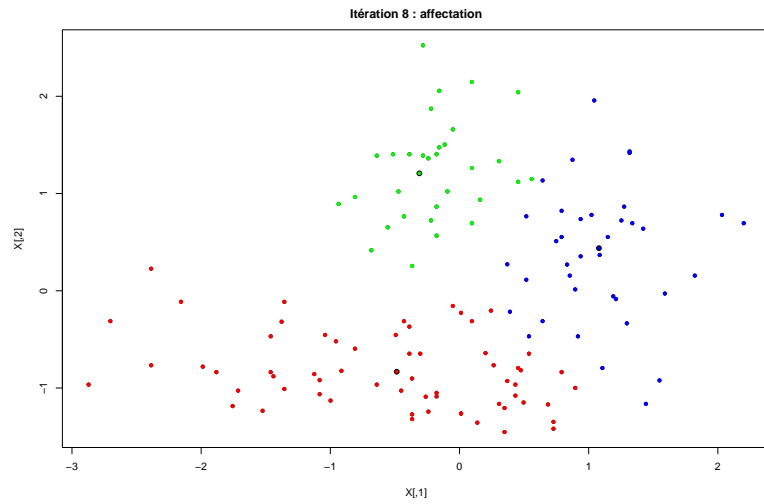
Exemple



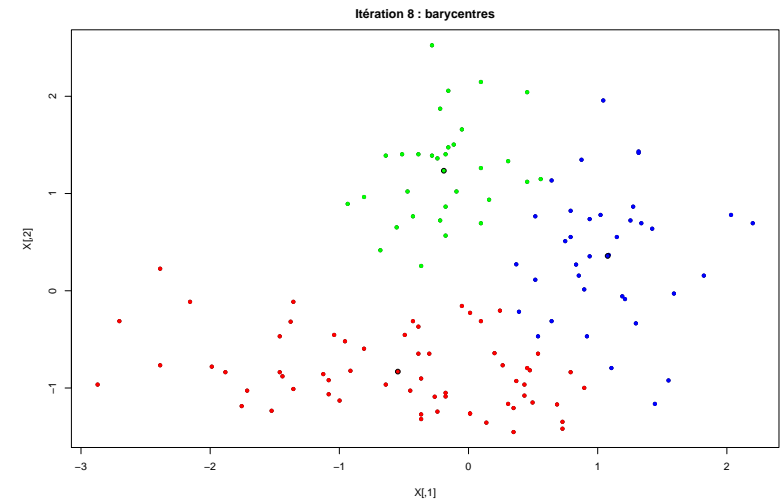
Exemple



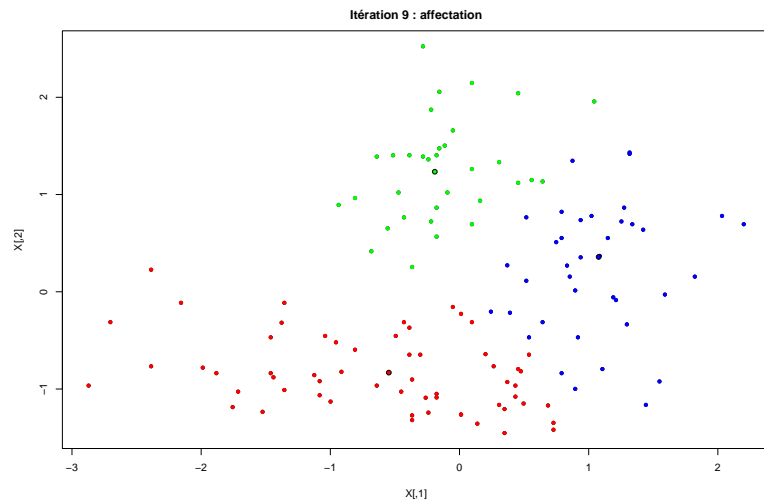
Exemple



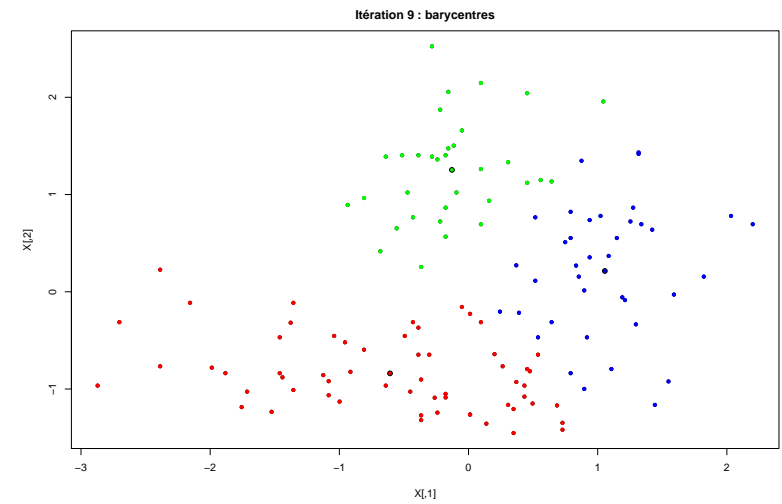
Exemple



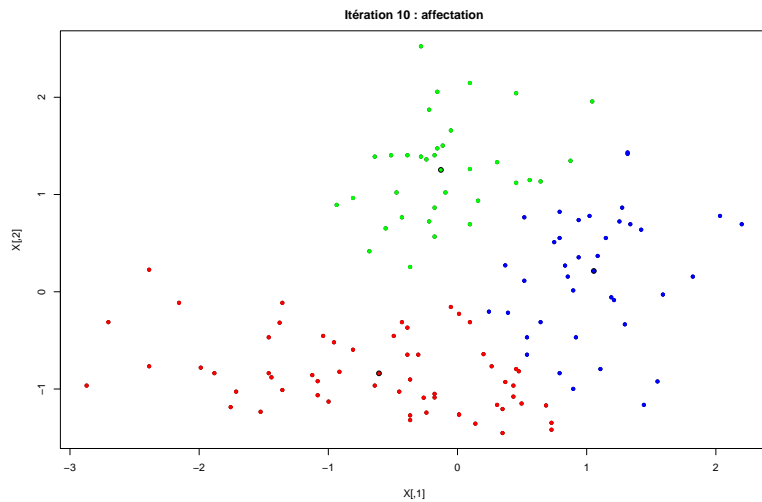
Exemple



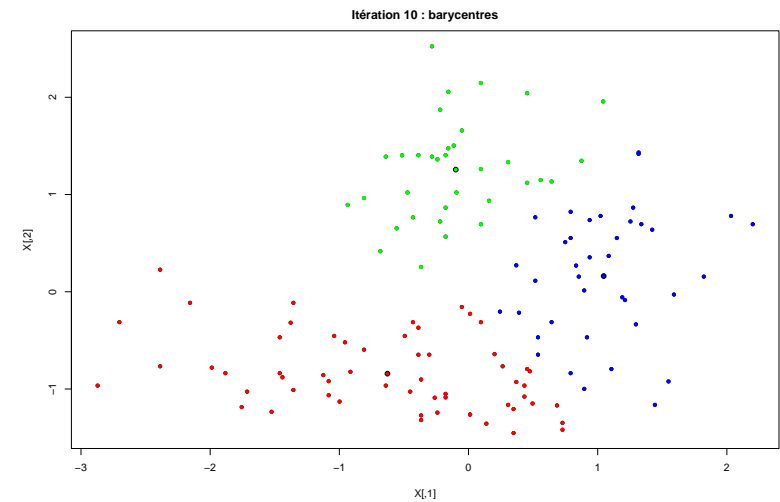
Exemple



Exemple



Exemple



Réseaux de neurones RBF

Principe simple :

- choisir q
- positionner q centres (par K-means)
- choisir un σ_k pour chaque centre : par exemple 2 fois la distance au centre le plus proche
- calculer

$$\mathbf{z}_i = \left(\exp\left(-\frac{\|\mathbf{x}_i - \mathbf{c}_1\|^2}{2\sigma_1^2}\right), \dots, \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{c}_q\|^2}{2\sigma_q^2}\right) \right)$$

- construire un modèle linéaire en minimisant :

$$\mathcal{E}(\mathbf{v}) = \sum_{i=1}^N \left(y_i - \mathbf{v}_0 - \sum_{k=1}^q \mathbf{v}_k z_{ik} \right)^2$$

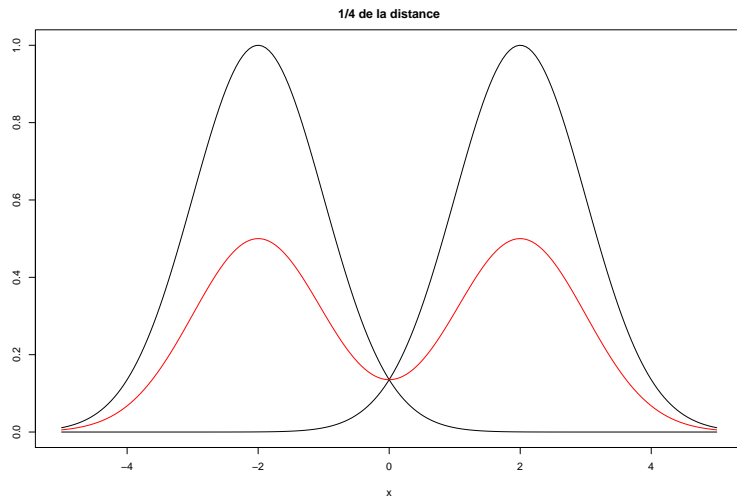
Remarque

- classifieur

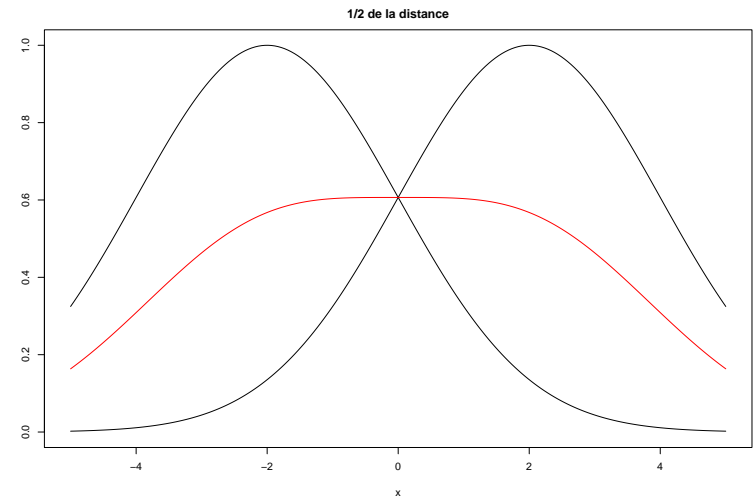
$$\psi(\mathbf{x}) = \text{signe} \left(\mathbf{v}_0 + \sum_{k=1}^q \mathbf{v}_k \exp\left(-\frac{\|\mathbf{x} - \mathbf{c}_k\|^2}{2\sigma_k^2}\right) \right)$$

- σ_k superposition des Gaussiennes :
 - trop petit : zones vides
 - trop grand : problèmes numériques

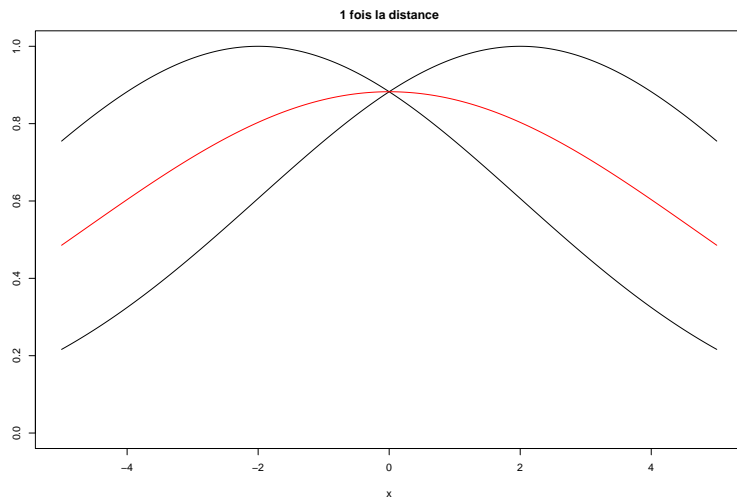
Exemple



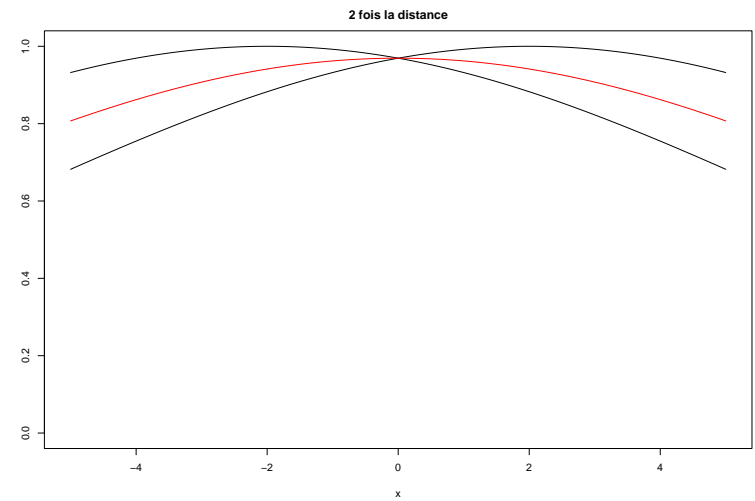
Exemple



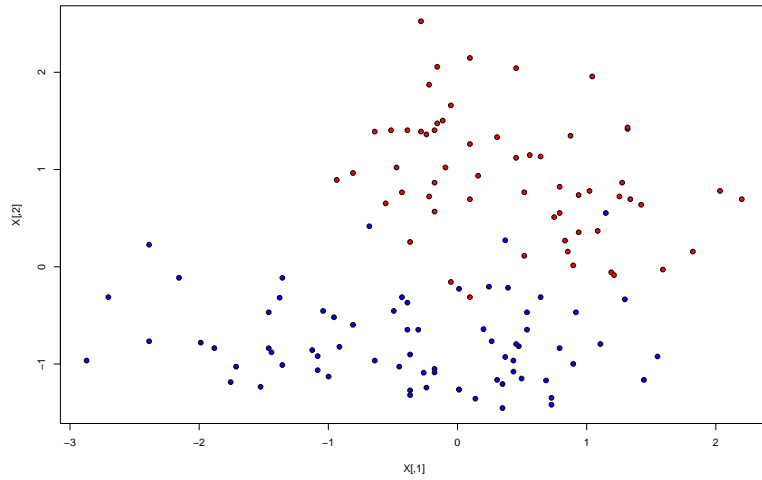
Exemple



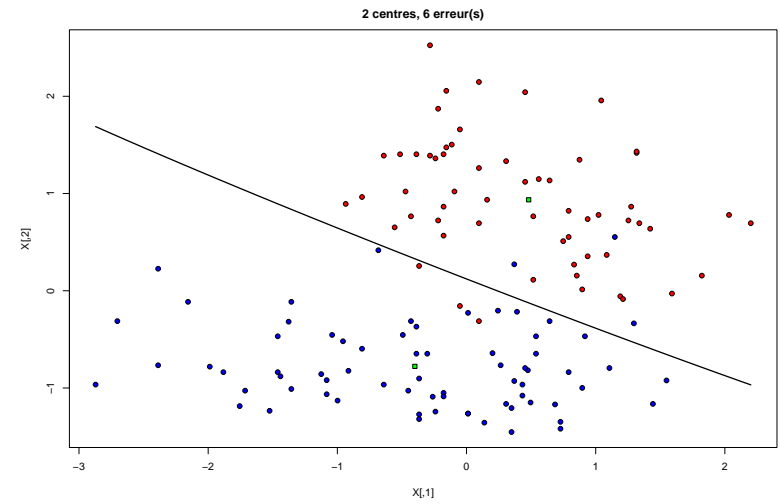
Exemple



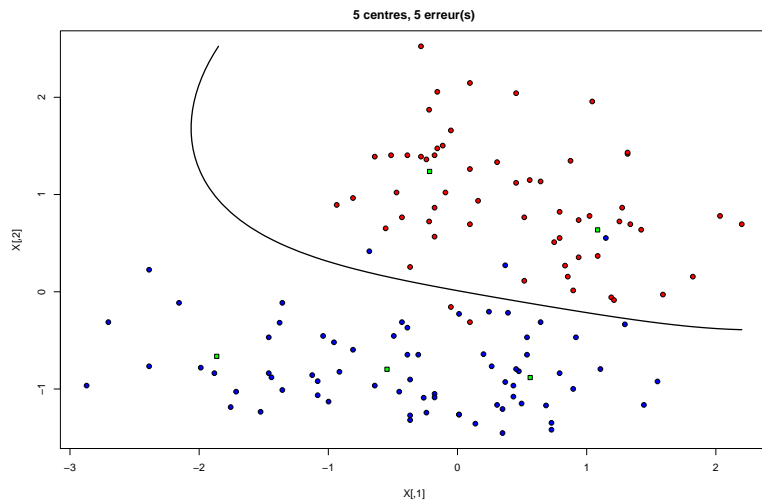
Application



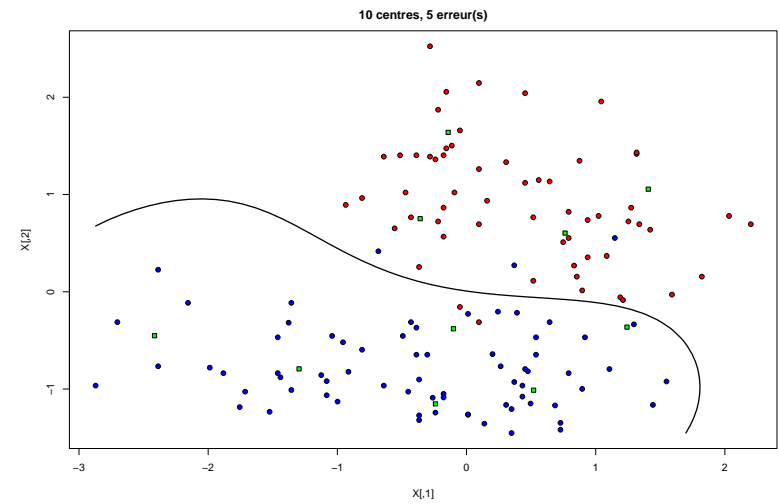
Application



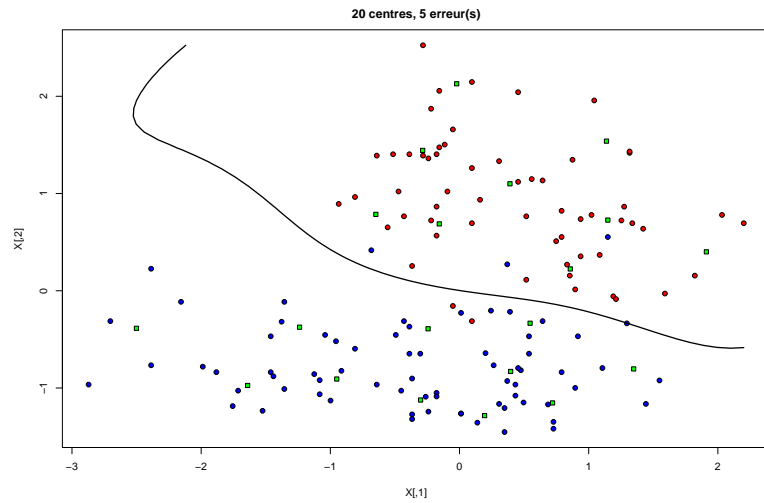
Application



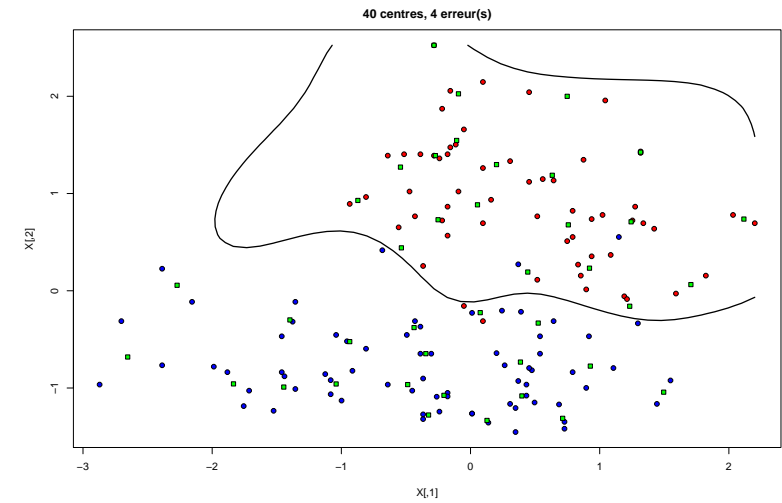
Application



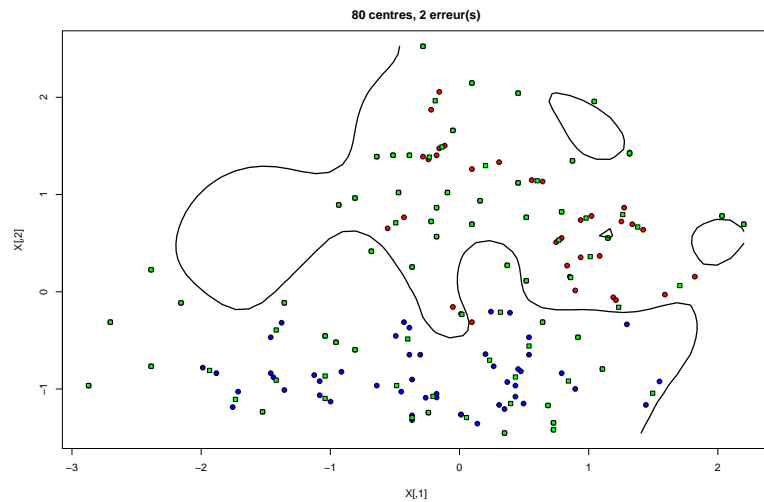
Application



Application



Application



Transformation explicite des données

Résumé

- transformation des données $\phi : \mathbb{R}^p \rightarrow \mathcal{H}$
- construction d'un modèle linéaire sur les données transformées
- en pratique :
 - polynômes (ou solutions proches) pour $p = 1$
 - quadratique pour p quelconque
 - neurones à base radiale (idem)
 - analyse discriminante ou moindres carrés
- difficultés :
 - choix des paramètres de la transformation
 - temps de calcul

Machine à vecteurs de support

- classifieur linéaire de marge maximale dans \mathcal{H} (produit scalaire noté $\langle \cdot, \cdot \rangle_{\mathcal{H}}$)
- problème à résoudre :

$$(P_C) \min_{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^N} \frac{1}{2} \langle \mathbf{w}, \mathbf{w} \rangle_{\mathcal{H}} + C \sum_{i=1}^N \xi_i,$$

avec $y_i (\langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle_{\mathcal{H}} + b) \geq 1 - \xi_i, 1 \leq i \leq N,$
 $\xi_i \geq 0, 1 \leq i \leq N.$

- sous cette forme, même problème de complexité que pour une régression (par exemple)

Problème dual

Astuce du noyau

- (P_C) est équivalent à

$$(D_C) \max_{\alpha} \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathcal{H}}$$

sous les contraintes $\sum_{i=1}^N \alpha_i y_i = 0$ et $0 \leq \alpha_i \leq C$

- on note $K(u, v) = \langle \phi(u), \phi(v) \rangle_{\mathcal{H}}$ (K est un noyau)
- si on peut calculer $K(u, v)$ directement, (D_C) se résout sans calculer les $\phi(\mathbf{x}_i)$
- classifieur

$$\mathbf{x} \mapsto \text{signe} \left(b + \sum_{\alpha_i \neq 0} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) \right)$$

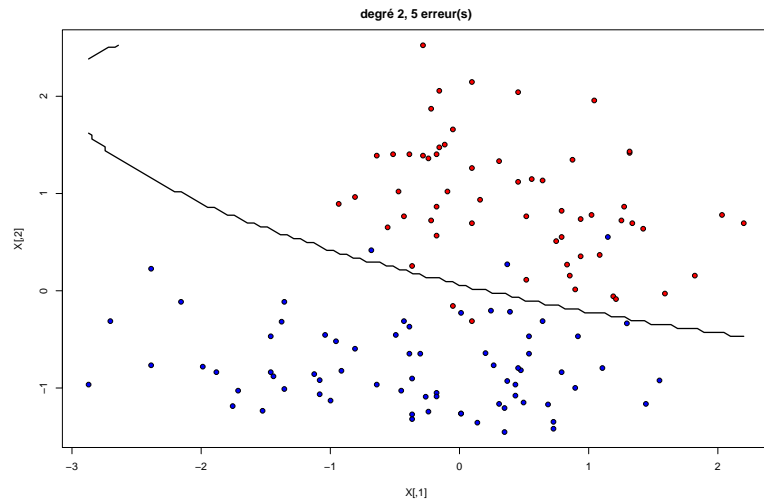
Transformation implicite des données

- se donner $\phi : \mathbb{R}^p \rightarrow \mathcal{H}$ et un produit scalaire $\langle \cdot, \cdot \rangle_{\mathcal{H}}$...
- est équivalent à se donner $K : \mathbb{R}^p \times \mathbb{R}^p \rightarrow \mathbb{R}^+$:
 - symétrique : $K(u, v) = K(v, u)$
 - positif : $\sum_{i,j} \lambda_i \lambda_j K(u_i, v_j) \geq 0$
- intérêt : calcul beaucoup plus simple et rapide de K (parfois)
- exemple :
 - $u = (u_1, u_2)$ and $v = (v_1, v_2)$
 - $K(u, v) = \left(1 + \sum_{i=1}^2 u_i v_i \right)^2$ (5 opérations)
 - $\Phi(u) = (1, \sqrt{2}u_1, \sqrt{2}u_2, \sqrt{2}u_1 u_2, u_1^2, u_2^2)$ (3 opérations)
 - $K(u, v) = \langle \Phi(u), \Phi(v) \rangle$ (3 + 3 + 6 opérations)

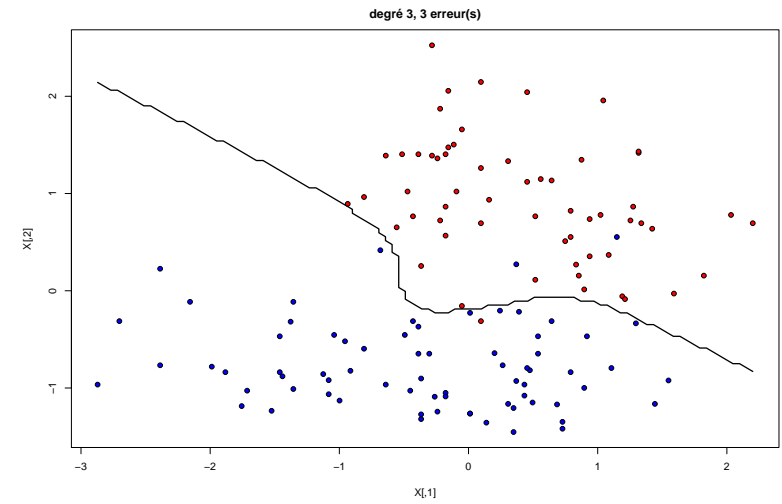
Exemples de noyau

- noyau polynôme :
 - $K(u, v) = (1 + \langle u, v \rangle)^d$
 - temps de calcul en $O(p + d)$
 - équivalent à l'utilisation de l'ensemble des monômes de degré inférieur à d
- noyau Gaussien :
 - $K(u, v) = \exp \left(-\frac{\|u - v\|^2}{2\sigma^2} \right)$
 - lien avec les réseaux de neurones RBF :
 - MVS : $\mathbf{x} \mapsto \text{signe} \left(b + \sum_{\alpha_i \neq 0} \alpha_i y_i \exp \left(-\frac{\|\mathbf{x}_i - \mathbf{x}\|^2}{2\sigma^2} \right) \right)$
 - RBF : $\mathbf{x} \mapsto \text{signe} \left(b + \sum_{k=1}^q \beta_k \exp \left(-\frac{\|\mathbf{c}_k - \mathbf{x}\|^2}{2\sigma^2} \right) \right)$
 - critères d'erreur différents
- nombreuses autres possibilités (en particulier pour des données « spéciales » : textes, graphes, etc.)

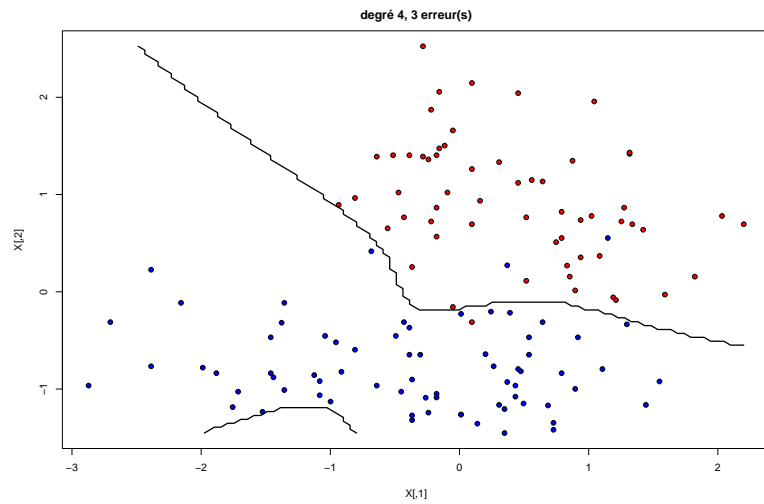
Exemple (noyau polynôme)



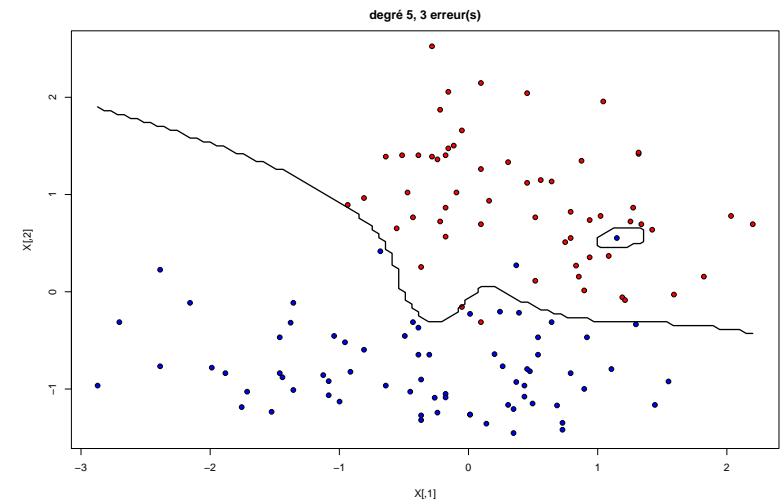
Exemple (noyau polynôme)



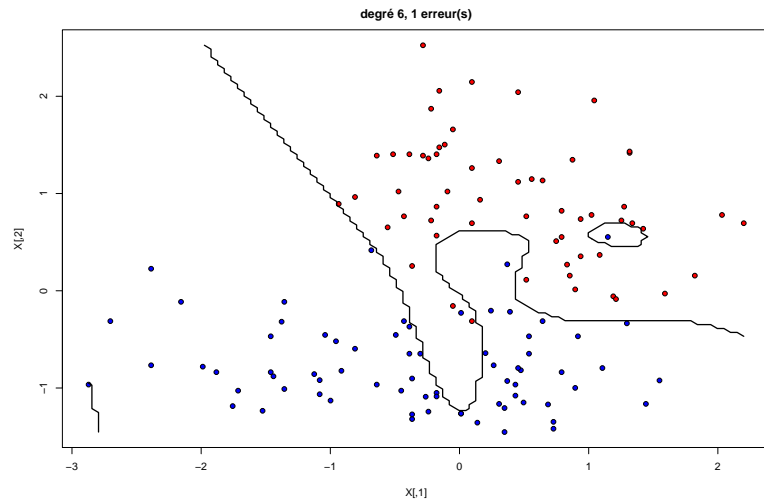
Exemple (noyau polynôme)



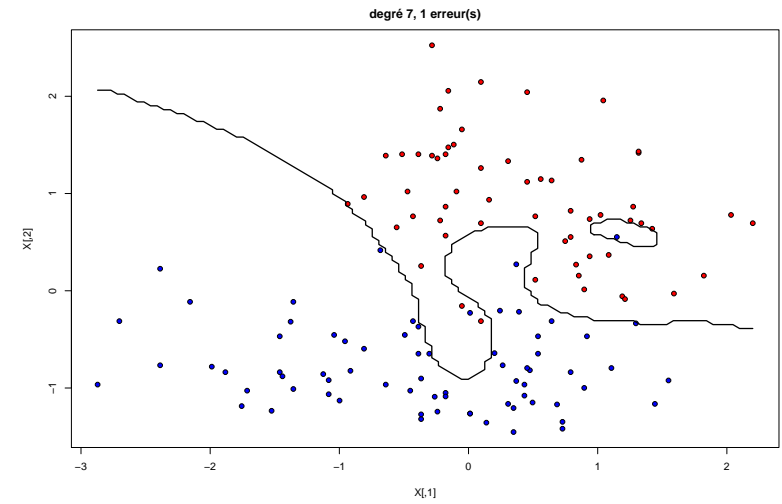
Exemple (noyau polynôme)



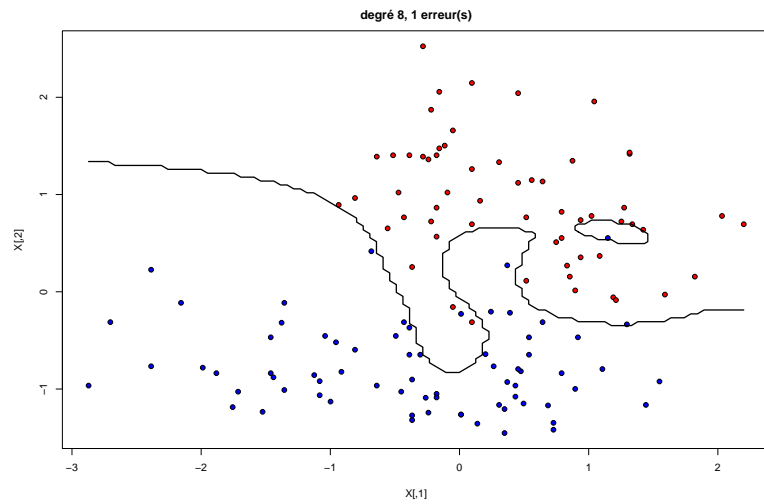
Exemple (noyau polynôme)



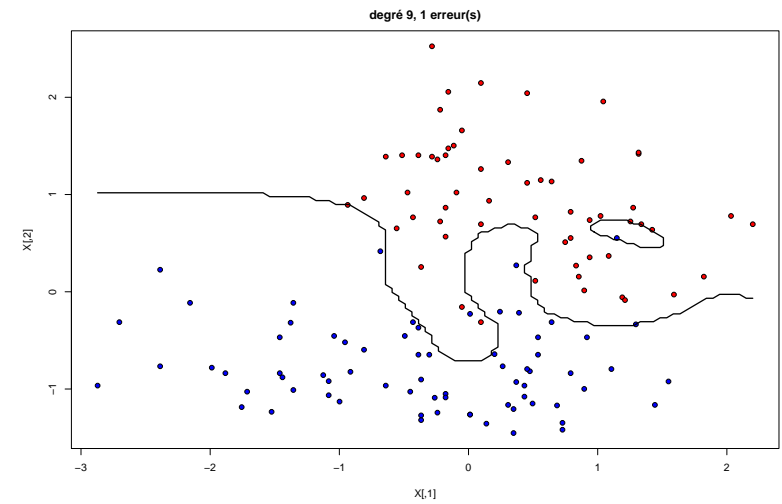
Exemple (noyau polynôme)



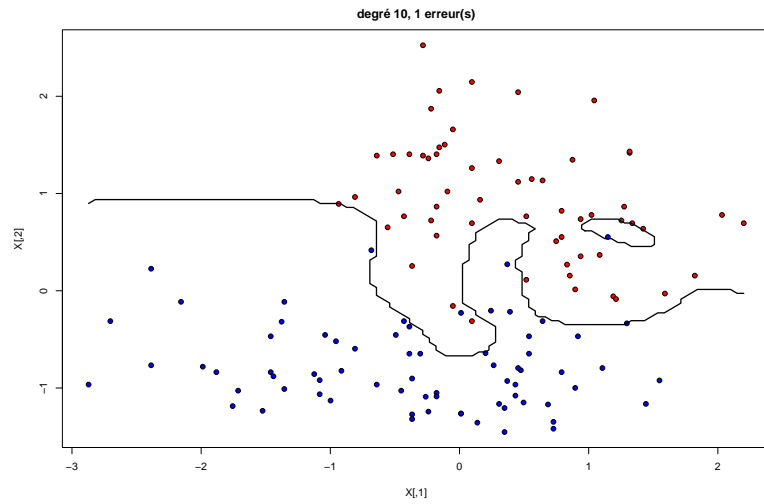
Exemple (noyau polynôme)



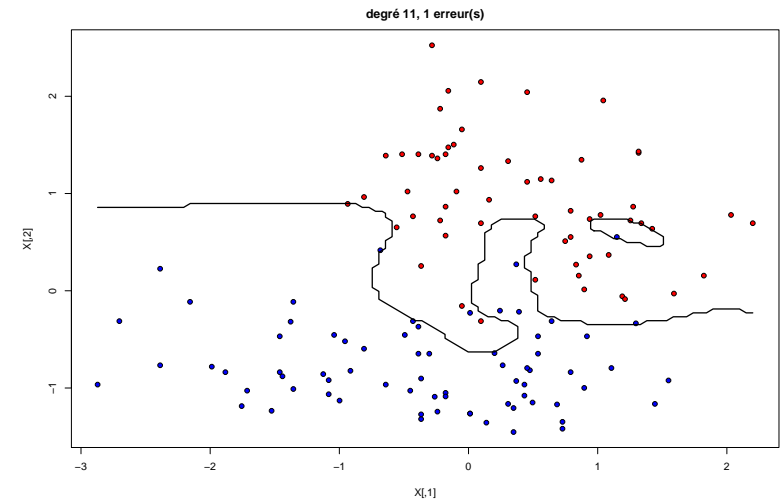
Exemple (noyau polynôme)



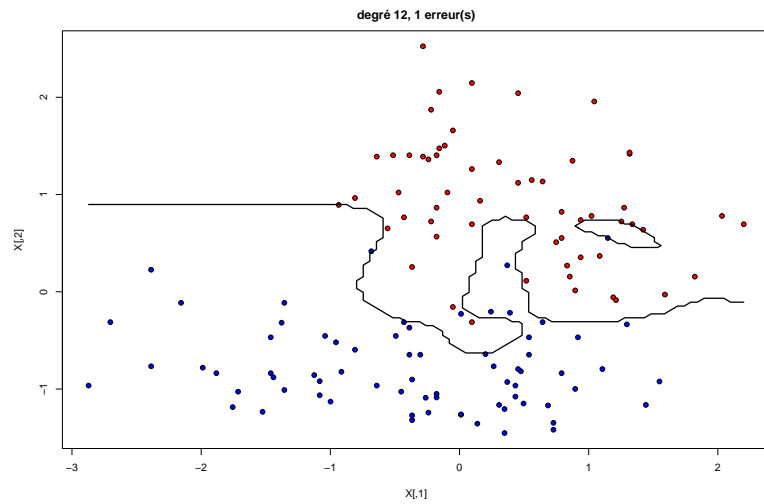
Exemple (noyau polynôme)



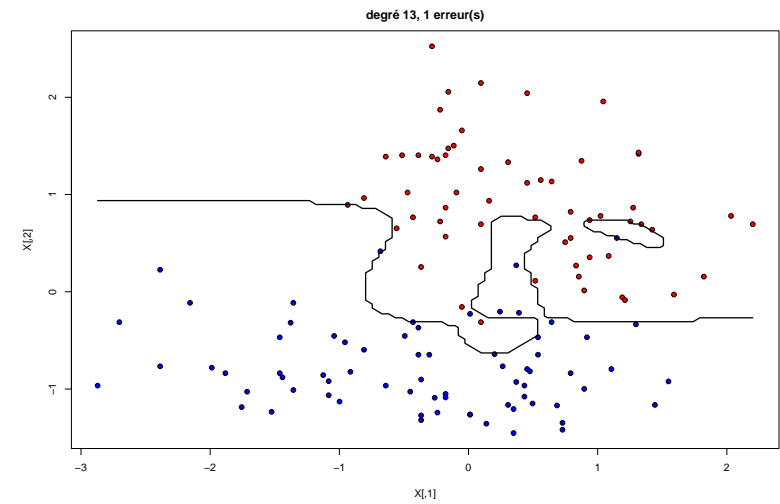
Exemple (noyau polynôme)



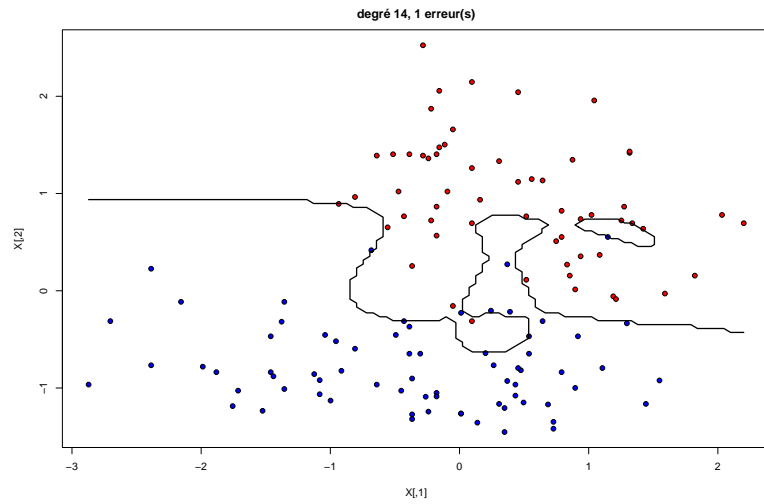
Exemple (noyau polynôme)



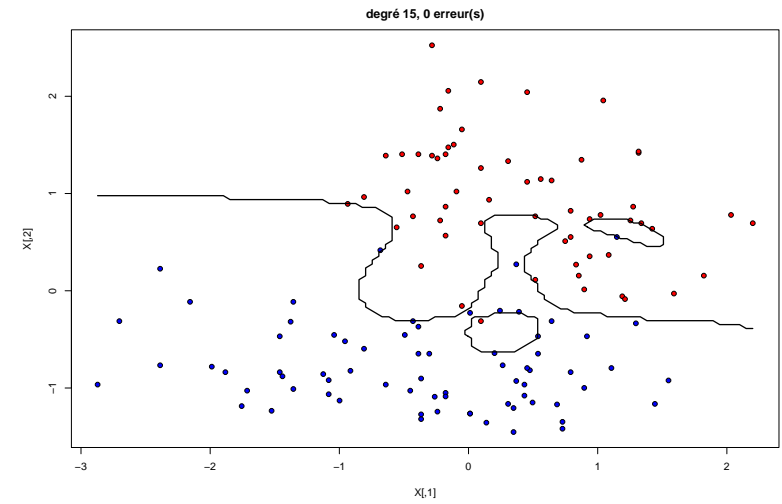
Exemple (noyau polynôme)



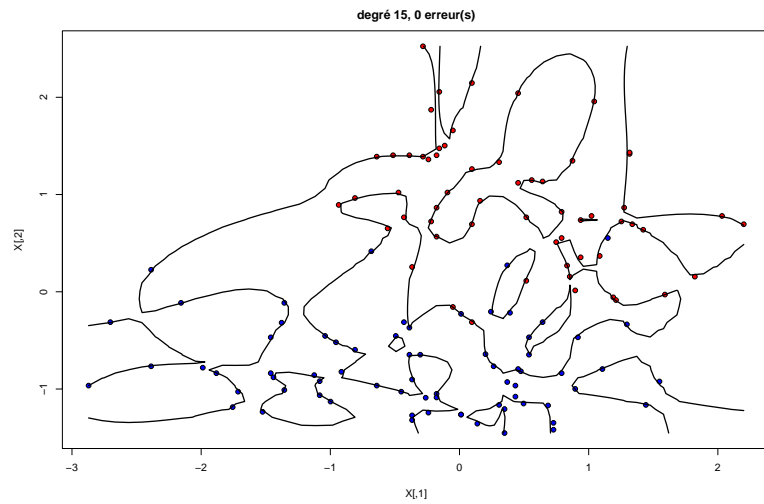
Exemple (noyau polynôme)



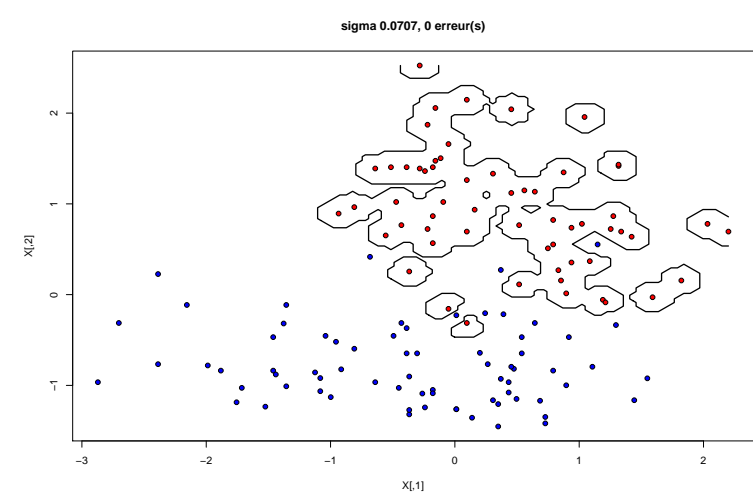
Exemple (noyau polynôme)



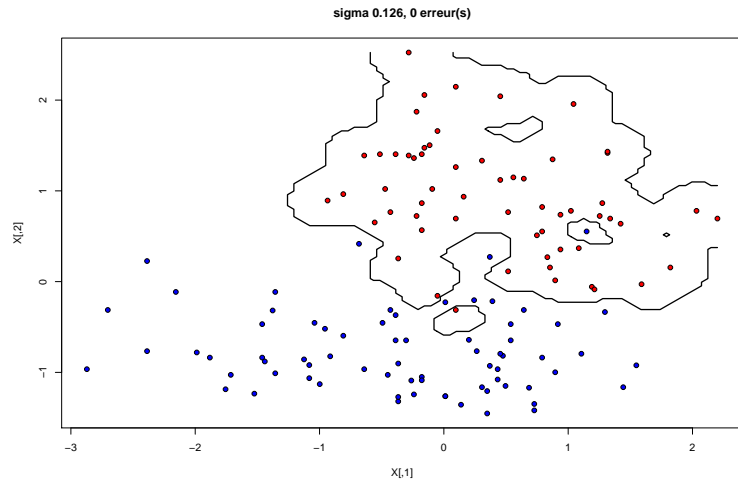
Exemple (noyau polynôme)



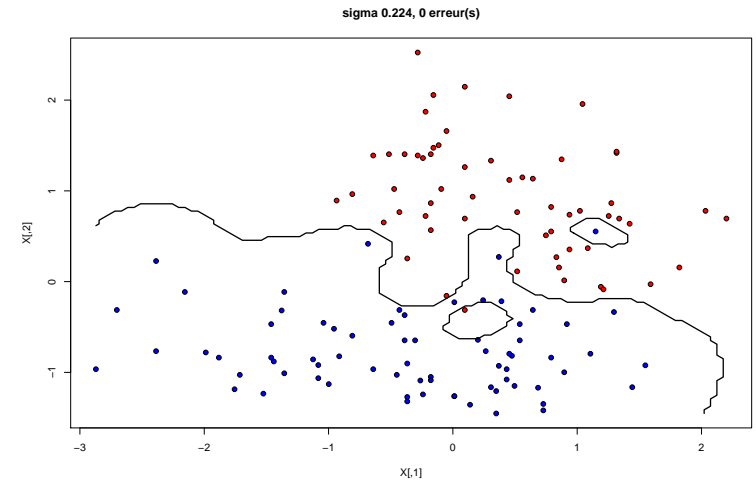
Exemple (noyau gaussien)



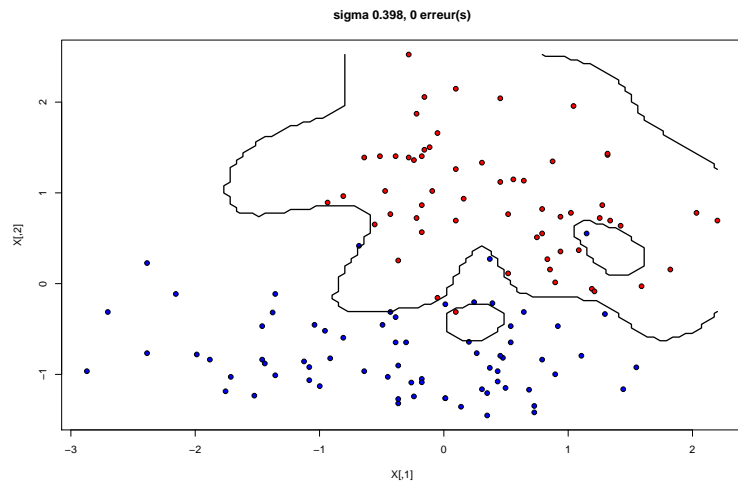
Exemple (noyau gaussien)



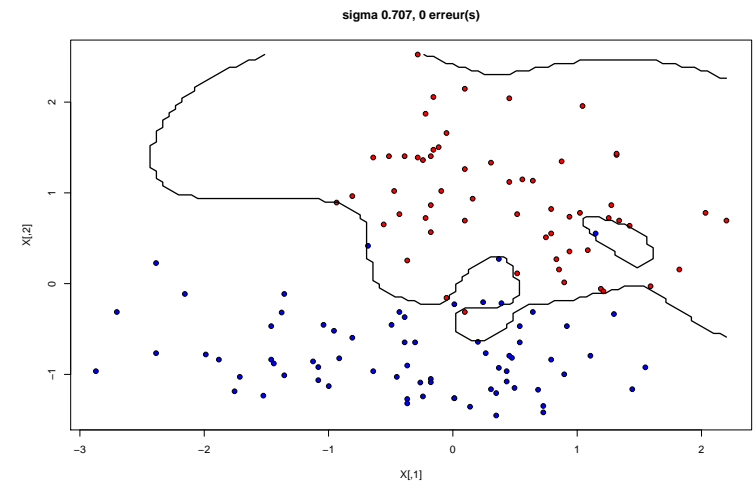
Exemple (noyau gaussien)



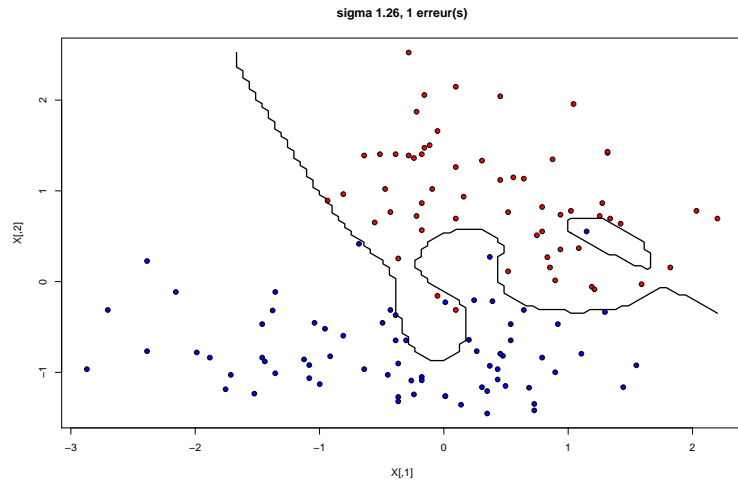
Exemple (noyau gaussien)



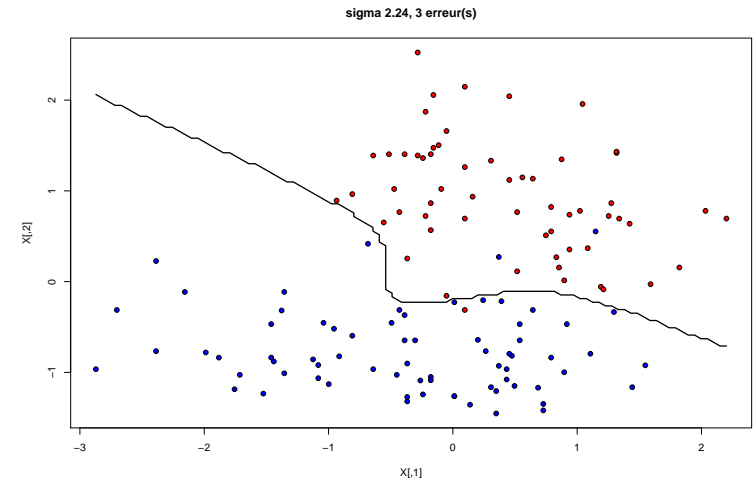
Exemple (noyau gaussien)



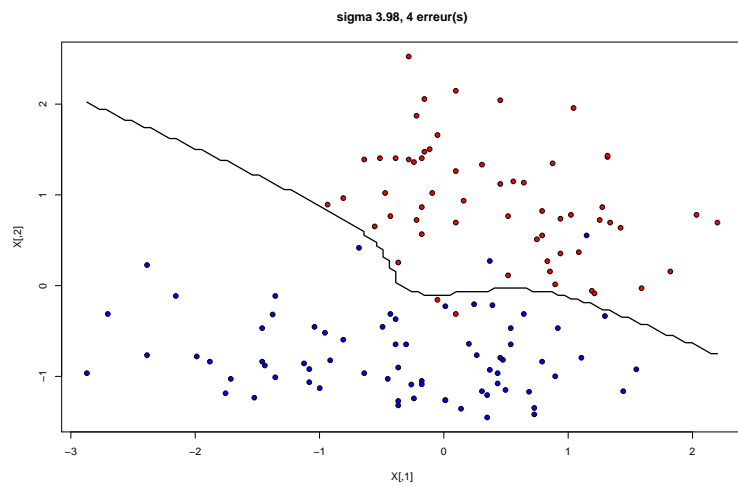
Exemple (noyau gaussien)



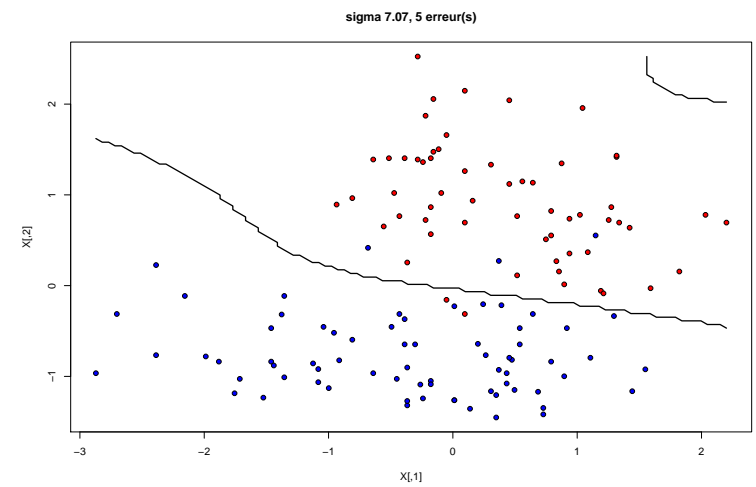
Exemple (noyau gaussien)



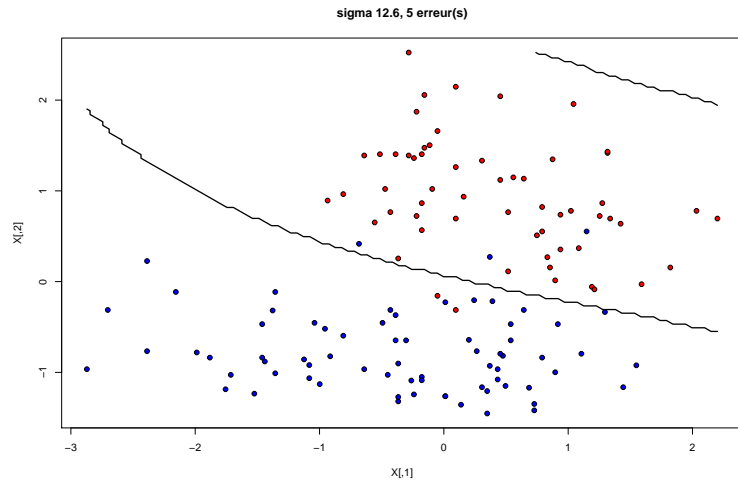
Exemple (noyau gaussien)



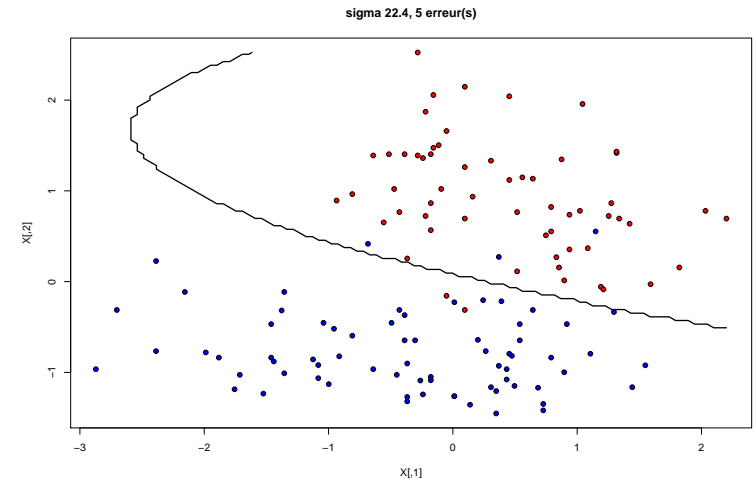
Exemple (noyau gaussien)



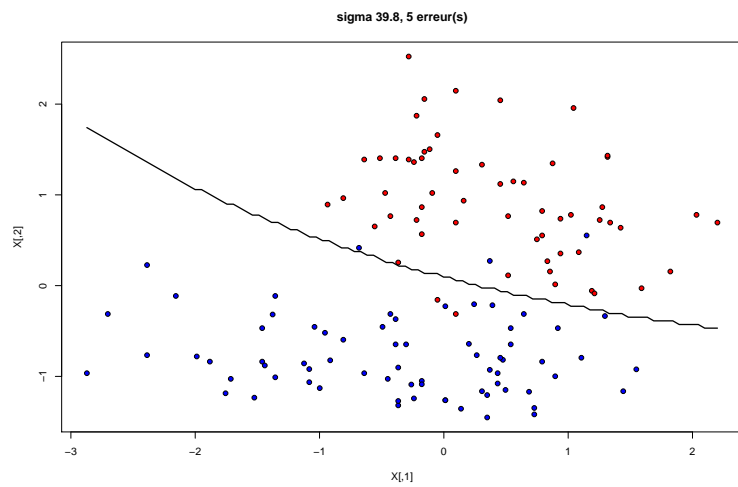
Exemple (noyau gaussien)



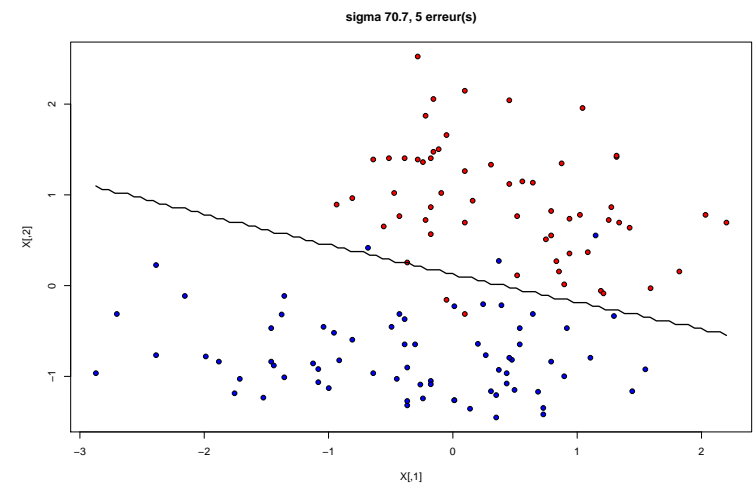
Exemple (noyau gaussien)



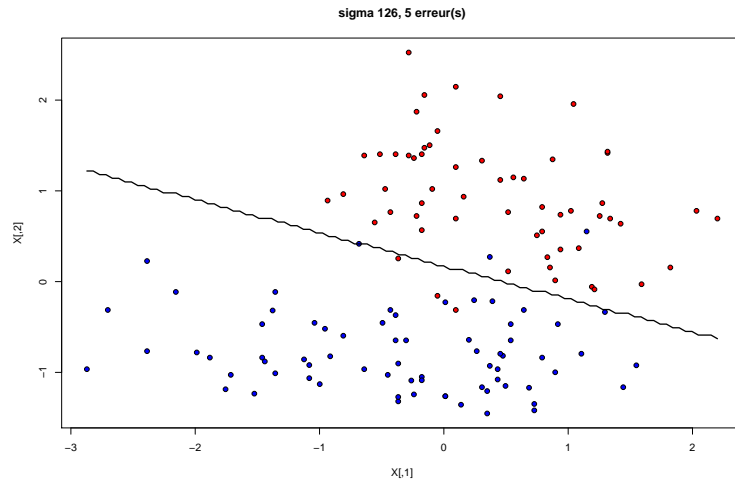
Exemple (noyau gaussien)



Exemple (noyau gaussien)



Exemple (noyau gaussien)



Transformation implicite des données

Résumé

- noyau de « comparaison » $K : \mathbb{R}^p \times \mathbb{R}^p \rightarrow \mathbb{R}^+$
- construction d'une machine à vecteurs de support s'appuyant sur le noyau
- équivalent à la construction d'une MVS sur les données transformées
- difficultés :
 - choix du noyau
 - choix du paramètre de la MVS C et de ceux du noyau
- avantages :
 - temps de calcul maîtrisés
 - grande souplesse (noyaux sur des données non vectorielles)