

Machine Learning pour le Big Data

Claude Barras

claude.barras@u-psud.fr

LIMSI / Université Paris-Sud - Polytech Paris-Sud

Bibliographie / source des exemples:
Pattern Classification,
R. Duda, P. Hart & D. Stork,
éd. Wiley-Interscience, 2000.

But de la 1^{ère} partie du cours

- Apprentissage automatique
- **Approches symboliques** :
 - collection de règles et concepts
- **Approche numériques** (statistiques)
 - Représentation numérique du monde
(photo, image scannée, signal audio, capteur de gestes, stylo électronique, données génériques, ...)
 - Utilisation d' **algorithme d'apprentissage**
 - L'être humain ne doit pas dire comment résoudre un problème, mais **annoter des exemples typiques**
 - Présentation de la plupart des algorithmes de base

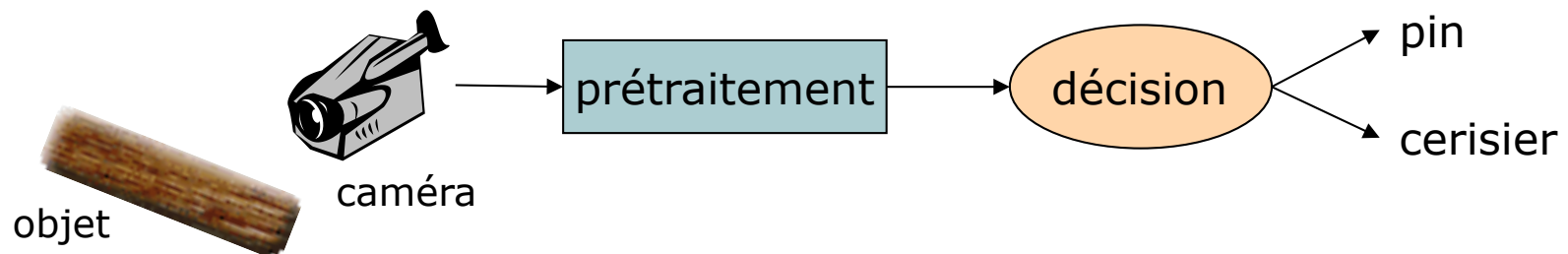
Introduction

- Comment développer des machines intelligentes?
- Perception de l'environnement
 - capteurs (lumière, images, température, son...)
 - ⇒ signaux électriques/binaires
 - interprétation des signaux?
 - signal audio ⇒ mots prononcés
 - image d'une enveloppe ⇒ code postal
 - image médicale 2D ⇒ détection d'une tumeur
 - ...
 - facile pour l'être humain (processus subconscient)
- Analyse d'un problème plus simple
 - associer des objets physiques à quelques catégories prédéfinies : **classification**

Introduction

Problème 1: classification de matériaux

- Tri de \neq types de bois sur une chaîne de fabrication



- Problème
 - les informations fournies par la caméra sont-elles toutes pertinentes pour la décision?
 - forme exacte, position, ...
- ➔ Prétraitement
 - réduction de la quantité de données pour extraire des valeurs pertinentes pour la décision : **codage**

Introduction

Problème 1: codage et décision

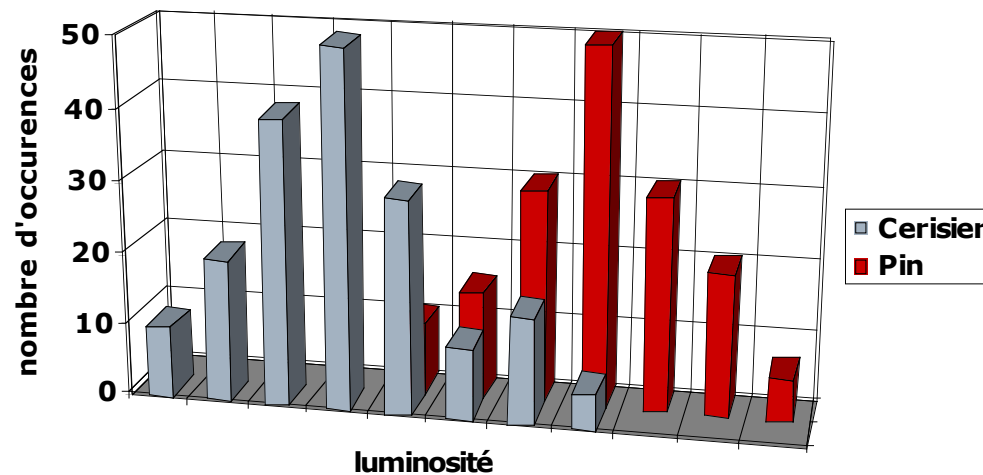
- Connaissance "experte": le pin est plus clair
 - module de codage calculant la luminosité de l'image
 - une seule valeur numérique
- Processus de décision
 - présenter l'objet à la caméra
 - calculer sa luminosité
 - si elle est plus élevée qu'un seuil s , alors on décide "pin", sinon "cerisier"
- Comment déterminer le seuil?

Introduction

Problème 1: choix du seuil

■ Histogramme

avec quelques morceaux de bois représentatifs

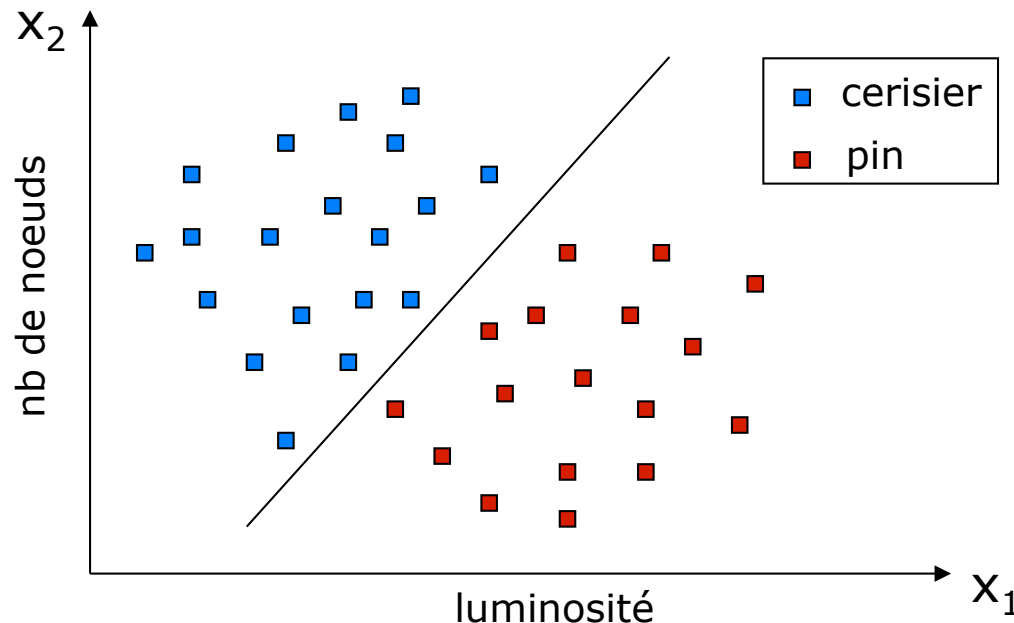


- + le pin est en moyenne plus clair que le cerisier
- il y a une zone de chevauchement
- ➔ il n'y a pas de seuil s qui permette de séparer parfaitement les 2 catégories de bois

Introduction

Problème 1: amélioration du codage

- ajout d'autres valeurs au codage
ex: nombre de nœuds par m²
⇒ codage $\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ avec x_1 : brillance
 x_2 : nombre de nœuds
- diagramme



Il semble possible
de séparer les 2
types de bois par
une droite

Introduction

Problème 1: qualité de la décision

- Questions
 - comment trouver la droite de séparation?
 - peut-on garantir que d'autres morceaux de bois seront correctement classés?
 - les échantillons étaient-ils
 - assez nombreux?
 - assez représentatif?
 - y a-t-il une mesure d'erreur ou de risque pour évaluer notre système?

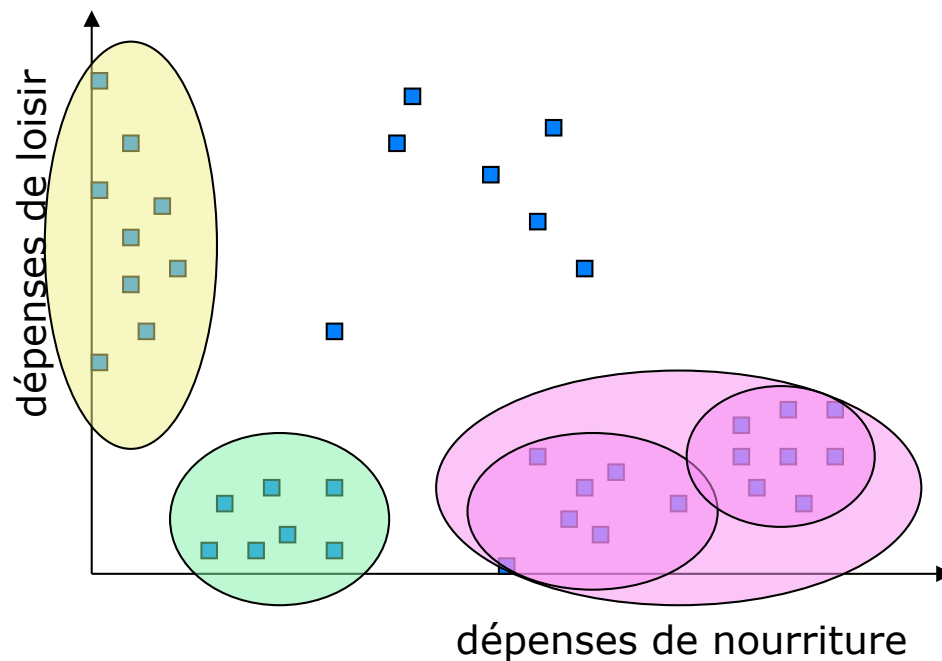
Introduction

Problème 2: étude commerciale

- grande surface:
 - base de données résumant les habitudes de consommation de ses clients
 - dépenses de nourriture, de loisir, ...
- objectif marketing:
 - cibler les campagnes de publicité
 - y a-t-il des groupes de clients aux comportements "proches"?

Introduction

Problème 2: regroupements possibles



a) petites courses ~ 20 €

b) courses hebdomadaires
(1 ou 2 catégories?)

c) clients achetant des articles
de loisir, et parfois tentés par
des achats de nourriture

■ Problèmes:

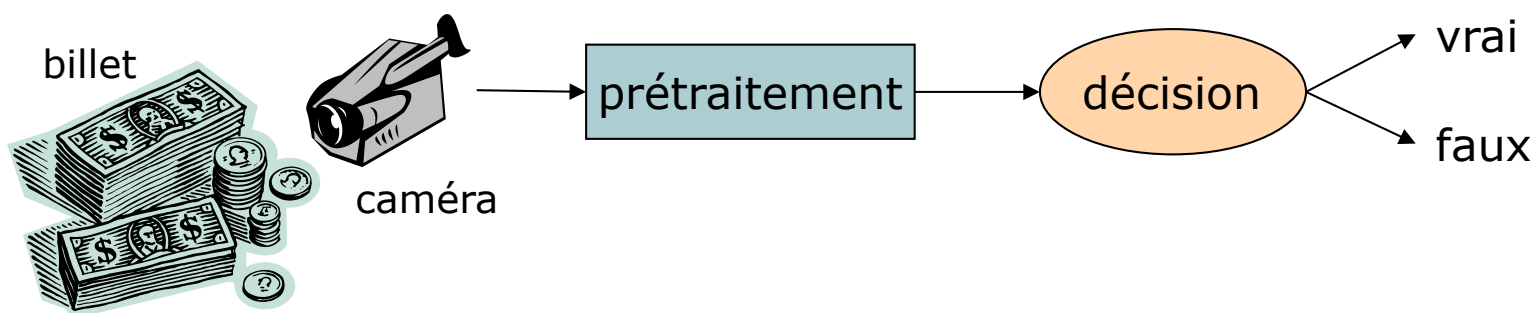
- nombre de classes?

(deux cas extrêmes: une seule classe pour tous,
ou autant de classes que de clients...)

- interprétation des regroupements?

Introduction

Problème 3: détection de faux billets



- Comparable au problème 1
(mesure de luminosité, épaisseur du papier,...)
- Particularités
 - il y a beaucoup moins de faux billets que de vrais
 - peut-on utiliser cette connaissance pour la décision?
 - cas extrême: on déclare toujours le billet vrai
(on fera moins d'erreurs qu'en utilisant des mesures fausses!)
 - les fausses décisions n'ont pas le même coût:
 - prendre un vrai billet pour un faux \Rightarrow on dérange le client
 - prendre un faux billet pour un vrai \Rightarrow on perd de l'argent!

Système de classification

- Architecture d'un système de classification
 - acquisition + numérisation des données
 - prétraitement/codage, spécifique au problème
 - réduction de la dimension des données
 - extraction d'un ensemble de valeurs pertinentes au problème de classification
 - ⇒ vecteur $\mathbf{x} = (x_1, \dots, x_d)^t \in \mathbf{R}^d$
 - classification (suivant plusieurs approches générales)
 - décision pour une classe en fonction du vecteur \mathbf{x}

Introduction

Rôle du codage

- La frontière prétraitement/classification est floue:
 - un codage inadapté rend la classification difficile
 - un codage sophistiqué peut simplifier la classification
- Exemple: orientation horizontale ou verticale d'un crayon sur une table?
 - acquisition: image 2D
 - codage 1: coordonnées (x,y) du début et de la fin du crayon $\Rightarrow \mathbf{x} \in \mathbf{R}^4$
 - codage 2: angle du crayon par rapport au bas de l'image
- Un codage peut être invariant par rapport à certaines propriétés
(ex: le codage 2 est invariant par rapport à la translation et l'agrandissement de l'objet)

Introduction

Approche statistique

- Ensemble d'exemples typiques du problème: **base d'apprentissage**
 - des **statistiques** sur ces données permettent de déterminer les paramètres du système, ex:
 - paramètres de la droite séparatrice
 - forme et position des regroupements
 - ...
- Deuxième ensemble d'exemples pour évaluer les performances du système: **base de test**
 - exemple: 1000 morceaux de bois (600 pin, 400 cerisier)
 - 900 pour l'apprentissage (dont 540 pins), 27 erreurs \Rightarrow 3%
 - 100 pour le test (dont 60 pins), 5 erreurs \Rightarrow 5%
 - en général, l'erreur sur la base de test est plus élevée, car les exemples n'ont pas servi à déterminer les paramètres

Introduction

Approche statistique ou structurelle

Exemple: reconnaissance des lettres A-Z à partir d'une image 32x32 points en niveaux de gris

- Approche statistique:
 - création d'une base d'apprentissage
 - choix du codage
 - détermination des paramètres
 - classification d'une nouvelle image comme le caractère le "plus probable", suivant les statistiques de l'apprentissage
- Avantage
 - pas de règles à définir
- Inconvénients
 - réponse non justifiée
 - améliorations difficiles
- Approche structurelle
 - classification en utilisant explicitement les caractéristiques de chaque caractère
 - ex: forme d'un 'A' \Rightarrow 2 traits diagonaux à 30° , et une barre horizontale au milieu
- Avantage
 - la décision est justifiée par le choix des règles
- Inconvénients
 - il faut un expert pour établir les règles
 - on risque de mal réagir avec des lettres déformées, des écritures atypiques...

Introduction

Approche supervisée ou non supervisée

■ Classification **supervisée**

- on connaît la bonne réponse pour chaque exemple de la base d'apprentissage
- ⇒ l'algorithme d'apprentissage essaye de trouver les meilleurs paramètres pour que le système donne la bonne réponse pour tous les exemples de la base d'apprentissage
- Exemples: reconnaissance du bois, de faux billets, des caractères...

■ Classification **non supervisée**

- on ne connaît pas la "bonne" réponse pour les exemples de la classe d'apprentissage
- ⇒ l'algorithme d'apprentissage essaye de trouver un regroupement de plusieurs exemples qui "se ressemblent" (critère à définir!)
- Exemples: étude commerciale

■ Approches combinées

A. Classification supervisée

■ Problème

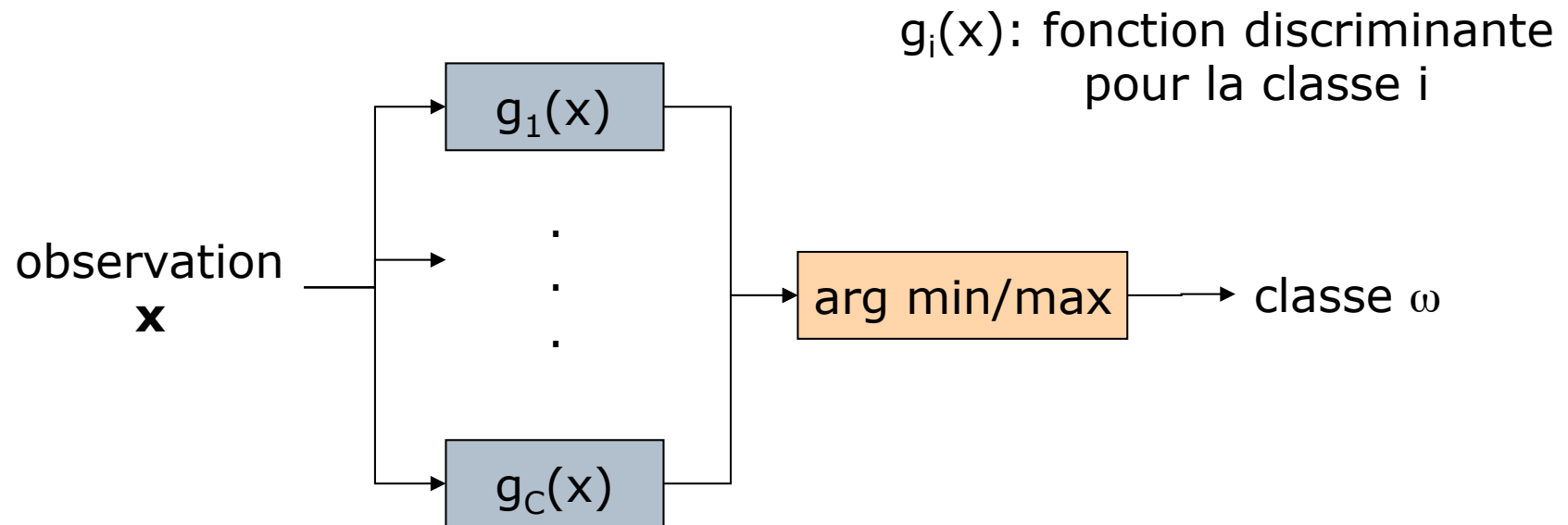
- a) reconnaissance des catégories "naturelles" d'objets (morceaux de bois, faux billets)
- b) attribuer des catégories **prédéfinies** à des collections de données

■ Formalisation

- en entrée:
 - vecteur d'observation $\mathbf{x} = (x_1, \dots, x_d)^t$ de dimension d
- en sortie:
 - classe de l'observation $\omega \in \Omega = \{\omega_1, \dots, \omega_C\}$ parmi C classes
- base d'apprentissage:
 - n paires associant un vecteur d'entrée \mathbf{x} + sa classe
 $\{(\mathbf{x}^{(1)}, \omega^{(1)}), \dots, (\mathbf{x}^{(n)}, \omega^{(n)})\}$
 - alternativement, un ensemble d'exemples par classe:
 $\omega_1 : \{\mathbf{x}^{(1,1)}, \dots, \mathbf{x}^{(1,n_1)}\}$
...
 $\omega_C : \{\mathbf{x}^{(C,1)}, \dots, \mathbf{x}^{(C,n_C)}\}$

A. Classification supervisée

Architecture générale



A. Classification supervisée

Approches envisagées

1. Déterminer un représentant μ_i pour chaque classe
fonction discriminante à minimiser:
distance entre l'observation et le représentant

$$g_i(\mathbf{x}) = D(\mathbf{x}, \mu_i)$$

2. Modéliser les fonctions discriminantes $g_i(\mathbf{x})$ ou les frontières de décision, en supposant une forme paramétrique – par ex. une droite
3. Modéliser la probabilité **a posteriori** de la classe i (après avoir observé \mathbf{x})

$$g_i(\mathbf{x}) = P(\omega_i / \mathbf{x})$$

- forme paramétrique des probabilités – par ex. une gaussienne
 - apprentissage des paramètres avec la base d'apprentissage
- ⇒ classifieur Bayésien

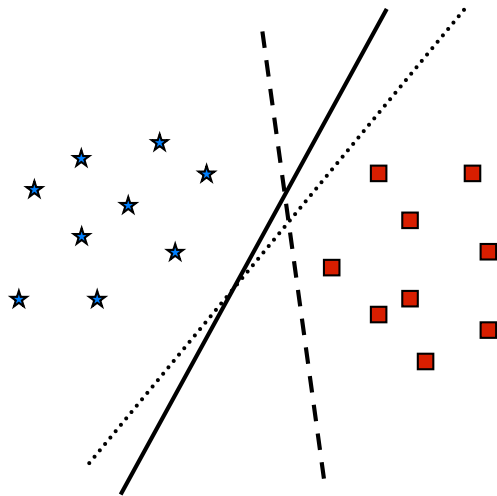
A. Classification supervisée

Frontières de décision (1)

- Tout classifieur peut être caractérisé par les frontières de décision:

$$g_i(\mathbf{x}) = g_j(\mathbf{x})$$

- Ex: 2 classes

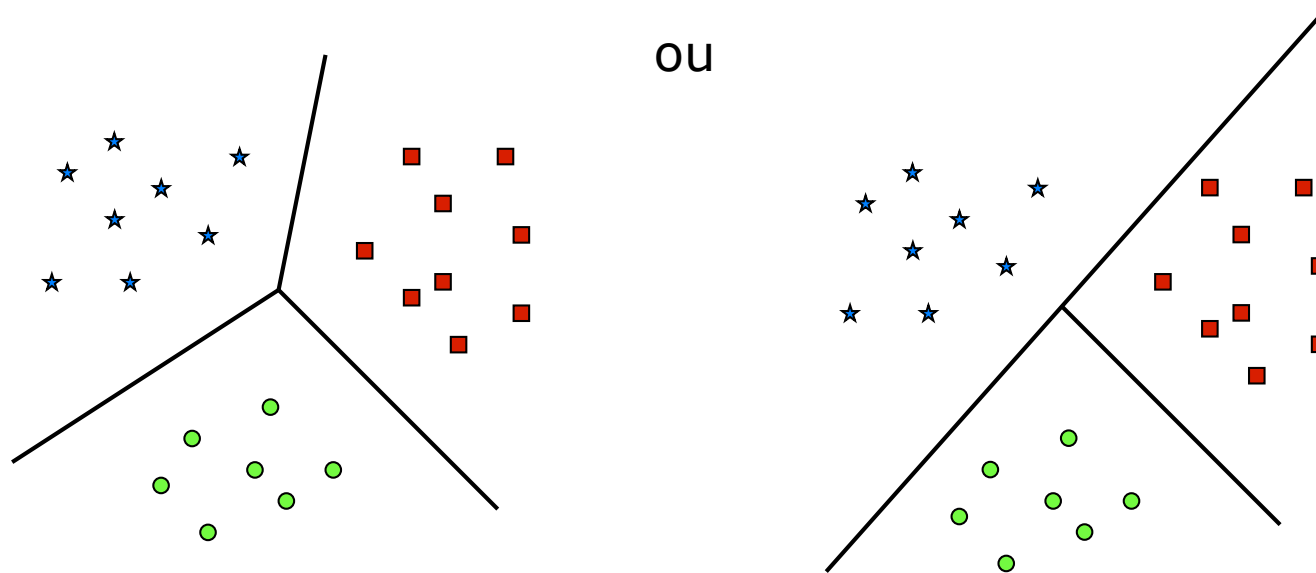


plusieurs droites
séparatrices possibles
⇒ critère d'optimalité?

A. Classification supervisée

Frontières de décision (2)

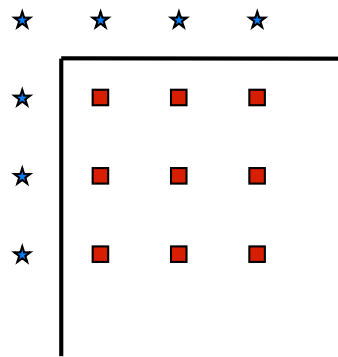
- ex: 3 classes



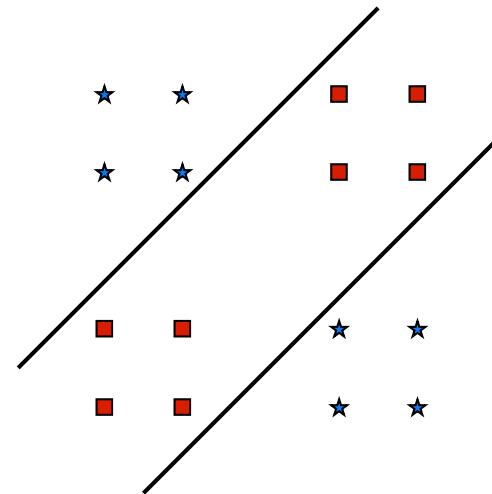
A. Classification supervisée

Frontières de décision (3)

- Peut-on toujours séparer 2 classes par une droite?



classes imbriquées



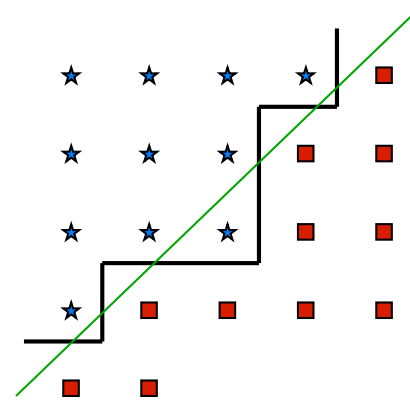
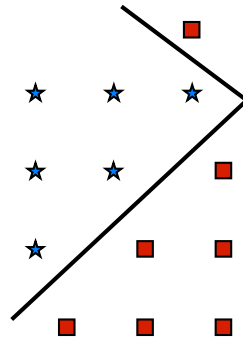
classes non-contiguës

⇒ Ces problèmes ne sont pas linéairement séparables!
Une solution: utiliser plusieurs droites

A. Classification supervisée

Frontières de décision (4)

- Autres exemples:



- Il peut être préférable de commettre des erreurs sur les exemples de la base d'apprentissage, au lieu de chercher à les séparer parfaitement:
 - on cherche les caractéristiques "générales" du problème, afin que les exemples non utilisés pendant l'apprentissage soient correctement classés: capacité de **généralisation**
 - problème des exemples atypiques, bruités, faussement étiquetés...

A.1. Classifieur à distance minimum

- Fonction discriminante = distance entre l'observation et un représentant de la classe
 - ⇒ choix de la classe pour laquelle la distance est minimale
- Variantes possibles :
 - calcul des représentants
 - nombre de représentants par classe
 - mesure de distance
- Cas de base :
 - représentant = moyenne de tous les exemples de la classe
 - Distance euclidienne

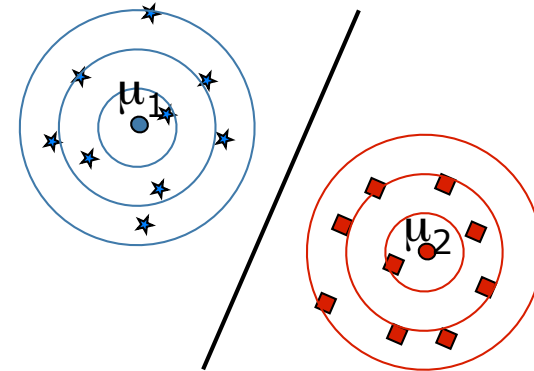
$$\mu_i = \frac{1}{n_i} \sum_{\mathbf{x}_j \in \omega_i} \mathbf{x}_j$$

$$\begin{aligned} D(\mathbf{x}, \mu) &= (\mathbf{x} - \mu)^t (\mathbf{x} - \mu) \\ &= \sum_{k=1}^d (x_k - \mu_k)^2 \end{aligned}$$

A.1. Classifieur à distance minimum

Cas euclidien

- Exemple
 - points de distance constante = cercles
 - frontière de décision = droite



- Fonctions discriminantes

$$\begin{cases} g_1(\mathbf{x}) = (\mathbf{x} - \mu_1)^t (\mathbf{x} - \mu_1) \\ g_2(\mathbf{x}) = (\mathbf{x} - \mu_2)^t (\mathbf{x} - \mu_2) \end{cases} \quad \text{on décide} \begin{cases} \text{classe 1, si } g_1(\mathbf{x}) < g_2(\mathbf{x}) \\ \text{classe 2, si } g_2(\mathbf{x}) < g_1(\mathbf{x}) \end{cases}$$

- Frontière de décision

$$g_1(\mathbf{x}) = g_2(\mathbf{x}) \quad (\text{pas de décision possible})$$

A.1. Classifieur à distance minimum

Equation de la droite séparatrice

$$g_1(\mathbf{x}) = g_2(\mathbf{x})$$

$$\Leftrightarrow (\mathbf{x} - \mu_1)^t (\mathbf{x} - \mu_1) = (\mathbf{x} - \mu_2)^t (\mathbf{x} - \mu_2)$$

$$\Leftrightarrow \mathbf{x}^t \mathbf{x} - \mu_1^t \mathbf{x} - \mathbf{x}^t \mu_1 + \mu_1^t \mu_1 = \mathbf{x}^t \mathbf{x} - \mu_2^t \mathbf{x} - \mathbf{x}^t \mu_2 + \mu_2^t \mu_2$$

$$\Leftrightarrow -2\mu_1^t \mathbf{x} + 2\mu_2^t \mathbf{x} + \mu_1^t \mu_1 - \mu_2^t \mu_2 = 0$$

$$\Leftrightarrow 2(\mu_2 - \mu_1)^t \mathbf{x} - (\mu_2 - \mu_1)^t (\mu_1 + \mu_2) = 0$$

$$\Leftrightarrow (\mu_2 - \mu_1)^t \left(\mathbf{x} - \frac{\mu_1 + \mu_2}{2} \right) = 0$$

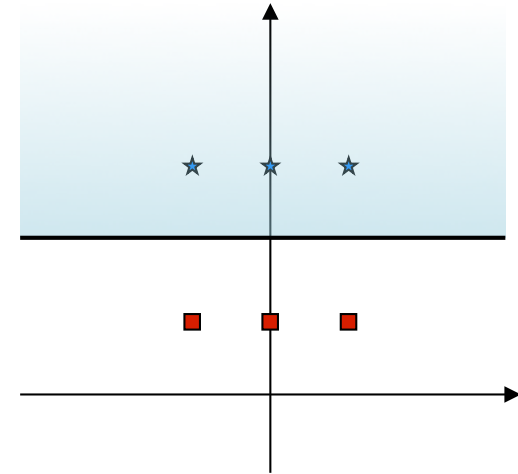
équation de la droite séparatrice = médiatrice de (μ_1, μ_2)

$$\text{et } (\mu_2 - \mu_1)^t \left(\mathbf{x} - \frac{\mu_1 + \mu_2}{2} \right) \begin{matrix} < \\ > \end{matrix} \begin{matrix} \omega_1 \\ \omega_2 \end{matrix}$$

A.1. Classifieur à distance minimum

Exemple

$$\begin{aligned} \omega_1 : \begin{Bmatrix} -1 & 0 & 1 \\ 1 & 1 & 1 \end{Bmatrix} &\Rightarrow \mu_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} & \mu_2 - \mu_1 = \begin{pmatrix} 0 \\ 2 \end{pmatrix} \\ \omega_2 : \begin{Bmatrix} -1 & 0 & 1 \\ 3 & 3 & 3 \end{Bmatrix} &\Rightarrow \mu_2 = \begin{pmatrix} 0 \\ 3 \end{pmatrix} & \frac{\mu_1 + \mu_2}{2} = \begin{pmatrix} 0 \\ 2 \end{pmatrix} \end{aligned}$$



■ Equation

$$(0 \ 2) \left(\begin{pmatrix} x \\ y \end{pmatrix} - \begin{pmatrix} 0 \\ 2 \end{pmatrix} \right) = 0 \Leftrightarrow 2(y - 2) = 0 \Leftrightarrow y = 2$$

■ Bilan

- droite séparatrice $y=2$
- classement de l'observation

$$\begin{pmatrix} x \\ y \end{pmatrix} \text{ classé } \begin{cases} \omega_1 & \text{si } y < 2 \\ \omega_2 & \text{si } y > 2 \end{cases} \quad (\text{pas de décision si } y = 2 !)$$

- test pour tous les exemples
 \Rightarrow tous correctement classés

$$\begin{cases} \forall \mathbf{x} \in \omega_1, y < 2 \\ \forall \mathbf{x} \in \omega_2, y > 2 \end{cases}$$

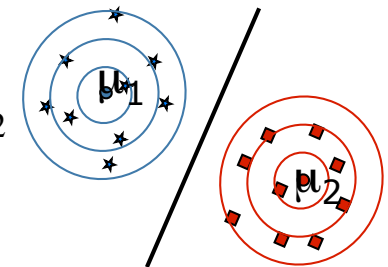
A.1. Classifieur à distance minimum

Choix de la mesure de distance (1)

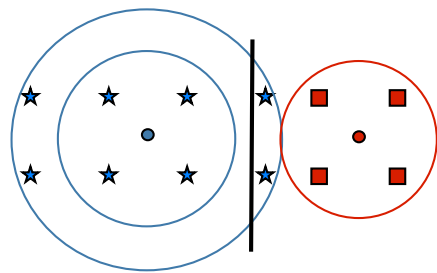
- Comparaison de plusieurs distances, performances?
 - points à distance constante (cercles, ovales...)
 - type de frontière de décision (droite, ...)

- Distance euclidienne

$$D(\mathbf{x}, \boldsymbol{\mu}) = \|\mathbf{x} - \boldsymbol{\mu}\|^2 = (\mathbf{x} - \boldsymbol{\mu})^t (\mathbf{x} - \boldsymbol{\mu}) = \sum_{k=1}^d (x_k - \mu_k)^2$$



- avantage: simplicité
- inconvénient: fonctionne mal avec une dispersion différente sur les différents axes



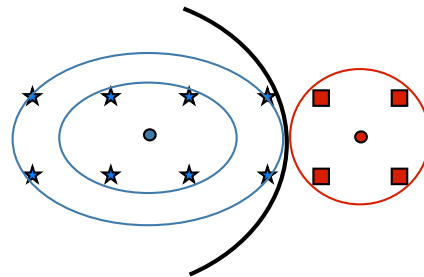
A.1. Classifieur à distance minimum

Choix de la mesure de distance (2)

- Amélioration: pondérer la distance par la variance des données dans chaque direction

$$D(\mathbf{x}, \boldsymbol{\mu}) = (\mathbf{x} - \boldsymbol{\mu})^t \begin{pmatrix} 1/\sigma_1^2 & & 0 \\ & \ddots & \\ 0 & & 1/\sigma_d^2 \end{pmatrix} (\mathbf{x} - \boldsymbol{\mu}) = \sum_{k=1}^d \frac{1}{\sigma_k^2} (x_k - \mu_k)^2$$

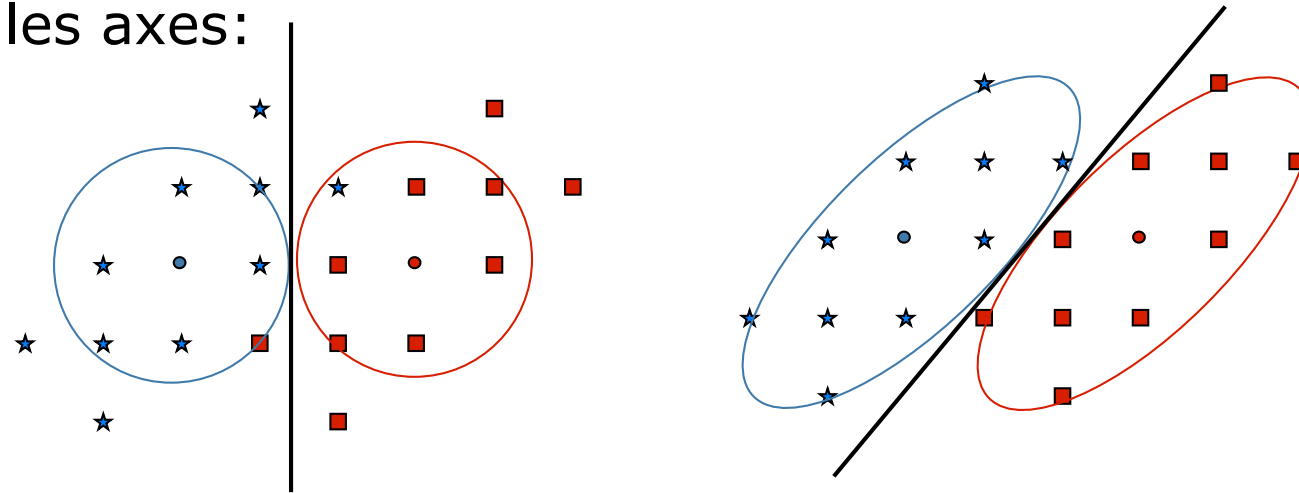
- points équivalents: ellipses alignées aux axes
- frontière de décision: paraboles



A.1. Classifieur à distance minimum

Choix de la mesure de distance (3)

...il faudrait que les ellipses ne soient pas forcément alignées avec les axes:



Distance de Mahalanobis: $D(\mathbf{x}, \mu) = (\mathbf{x} - \mu)^t \Sigma^{-1} (\mathbf{x} - \mu)$

avec Σ^{-1} inverse de la matrice de covariance

estimée par $\hat{\Sigma} = \frac{1}{n} \sum_i (\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T$

(par classe ou global)

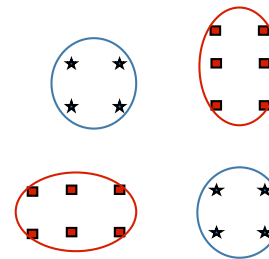
A.1. Classifieur à distance minimum

Nombre de représentants

- La distance de Mahalanobis n'est pas assez générale pour traiter tous les problèmes séparables linéairement



- ⇒ utiliser plusieurs références
 - combien par classe? (choix délicat)
 - comment les déterminer? (moins simple que la moyenne)
- Permet aussi de traiter des problèmes non séparables linéairement:



A.2. Classifieur à fonction discriminante linéaire

- Estimation directe des fonctions discriminantes ou des frontières de décision
 - hypothèse: forme paramétrique
 - paramètres estimés avec la base d'apprentissage
 - critère:
 - nombre d'erreurs commises sur les exemples de la base d'apprentissage
- Problème à 2 classes
 - on pose: $g(\mathbf{x}) = g_1(\mathbf{x}) - g_2(\mathbf{x})$
 - soit x un observation, on décide:
$$\begin{cases} \omega_1 & \text{si } g(\mathbf{x}) > 0 \\ \omega_2 & \text{si } g(\mathbf{x}) < 0 \end{cases}$$
(pas de décision si $g(\mathbf{x})=0$)

A.2. Classifieur à fonction discriminante linéaire

Droite séparatrice (1)

- Fonction discriminante linéaire

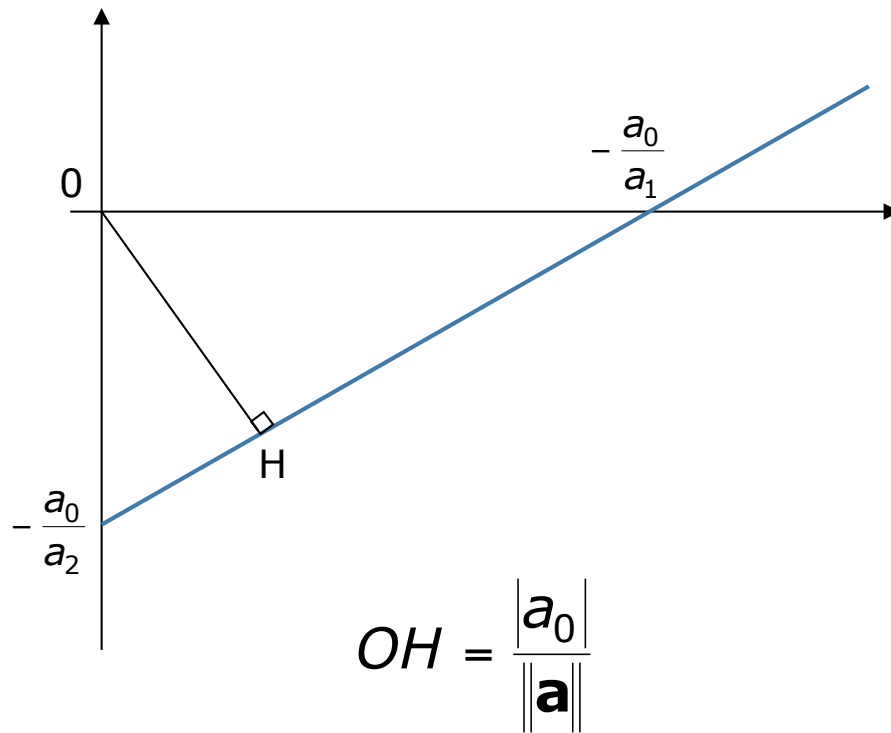
$$g(\mathbf{x}) = \mathbf{a}^t \cdot \mathbf{x} + a_0, \text{ avec } \begin{cases} \mathbf{a} : \text{vecteur des poids} \\ a_0 : \text{le seuil} \end{cases}$$

- Calcul de la droite séparatrice (en 2 dimensions)

$$\begin{aligned} \begin{pmatrix} a_1 \\ a_2 \end{pmatrix}^t \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + a_0 &= 0 \\ \Leftrightarrow a_1 x_1 + a_2 x_2 + a_0 &= 0 \\ \Leftrightarrow x_2 &= -\frac{a_1}{a_2} x_1 - \frac{a_0}{a_2} \end{aligned}$$

A.2. Classifieur à fonction discriminante linéaire

Droite séparatrice (2)



- le vecteur \mathbf{a} donne l'orientation de la droite - il est **normal** à la surface de décision
- le seuil a_0 détermine la position de la droite ($a_0 = 0 \Rightarrow$ la droite passe par l'origine)

A.2. Classifieur à fonction discriminante linéaire

Exemple

$$\omega_1 : \left\{ \begin{pmatrix} 1 \\ 2 \end{pmatrix} \quad \begin{pmatrix} 2 \\ 3 \end{pmatrix} \right\} \quad g(\mathbf{x}) = \mathbf{a}^t \cdot \mathbf{x} + a_0 \quad \text{avec} \quad \mathbf{a} = \begin{pmatrix} -1 \\ 2 \end{pmatrix} \text{ et } a_0 = -2$$
$$\omega_2 : \left\{ \begin{pmatrix} 2 \\ 1 \end{pmatrix} \quad \begin{pmatrix} 3 \\ 1 \end{pmatrix} \right\} \quad \Rightarrow g(\mathbf{x}) = (-1 \quad 2) \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - 2 = -x_1 + 2x_2 - 2$$

■ Test des points

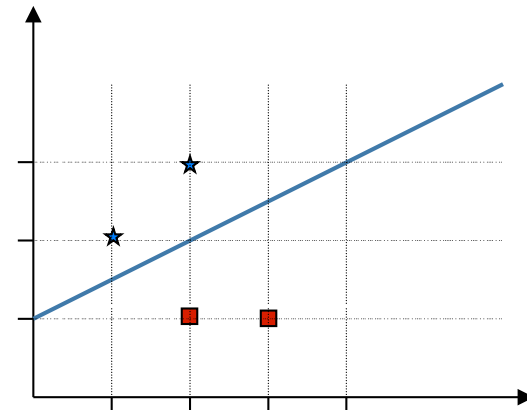
$$g(\mathbf{x}) > 0, \forall \mathbf{x} \in \omega_1$$

$$g(\mathbf{x}) < 0, \forall \mathbf{x} \in \omega_2$$

$$g(\mathbf{x}) = 0 \Rightarrow x_2 = \frac{1}{2}x_1 + 1$$

■ Comment interpréter la valeur de $g(\mathbf{x})$?

- le signe suffit pour choisir la classe



A.2. Classifieur à fonction discriminante linéaire

Interprétation de $g(\mathbf{x})$

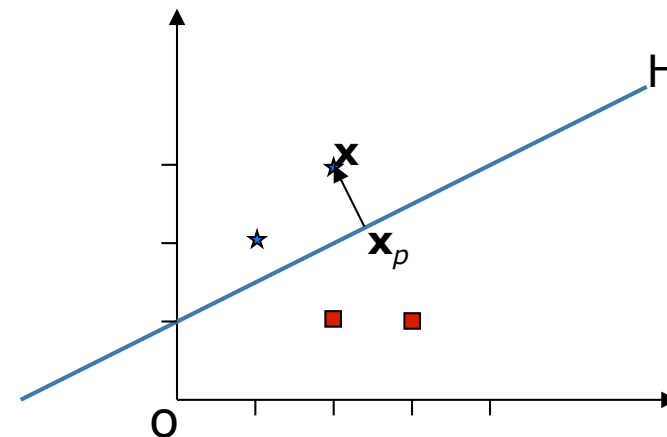
- Soit \mathbf{x}_p la projection de \mathbf{x} sur la droite séparatrice H
Comme \mathbf{a} est normal à H , on peut écrire

$$\mathbf{x} = \mathbf{x}_p + r \frac{\mathbf{a}}{\|\mathbf{a}\|}$$

où r est une mesure de distance
du point \mathbf{x} à la frontière H

d'où

$$\begin{aligned} g(\mathbf{x}) &= \mathbf{a}^t (\mathbf{x}_p + r \frac{\mathbf{a}}{\|\mathbf{a}\|}) + a_0 \\ &= (\mathbf{a}^t \mathbf{x}_p + a_0) + r \frac{\mathbf{a}^t \mathbf{a}}{\|\mathbf{a}\|} \\ &= g(\mathbf{x}_p) + r \frac{\|\mathbf{a}\|^2}{\|\mathbf{a}\|} \end{aligned}$$

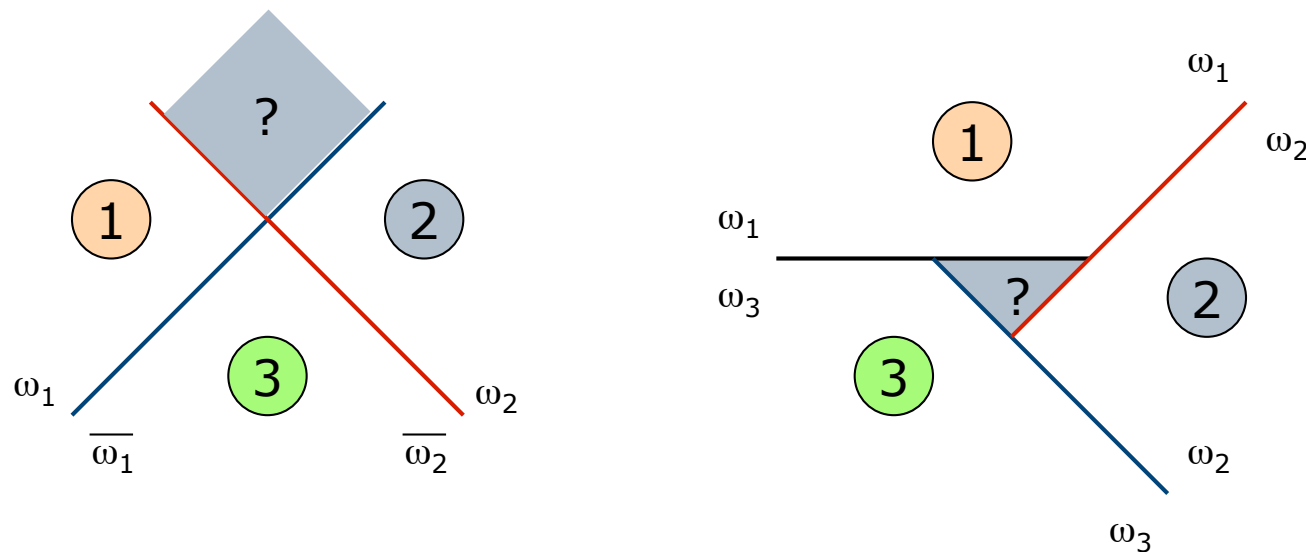


$$\boxed{g(\mathbf{x}) = r \cdot \|\mathbf{a}\|} \quad \text{ou } r = \frac{g(\mathbf{x})}{\|\mathbf{a}\|} \quad (\text{en particulier : } OH = \frac{|a_0|}{\|\mathbf{a}\|})$$

A.2. Classifieur à fonction discriminante linéaire

Cas multi-classes ($c > 2$)

- Approches possibles
 1. on considère $(c-1)$ décisions du type: $\omega_i / \text{pas } \omega_i$
 2. on utilise $c.(c-1)/2$ fonctions discriminantes linéaires pour séparer chaque paire de classes... mais les deux approches peuvent donner des régions pour lesquelles la classification n'est pas définie:



A.2. Classifieur à fonction discriminante linéaire

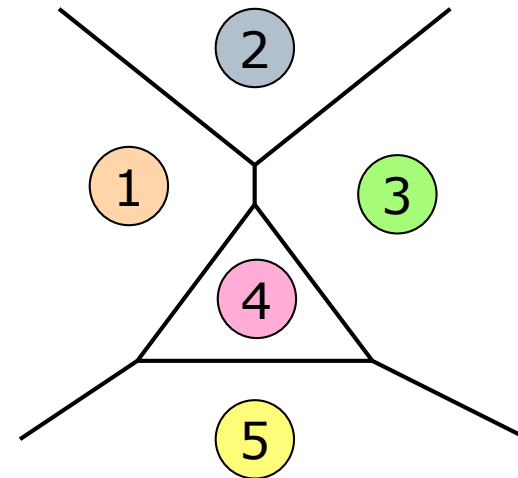
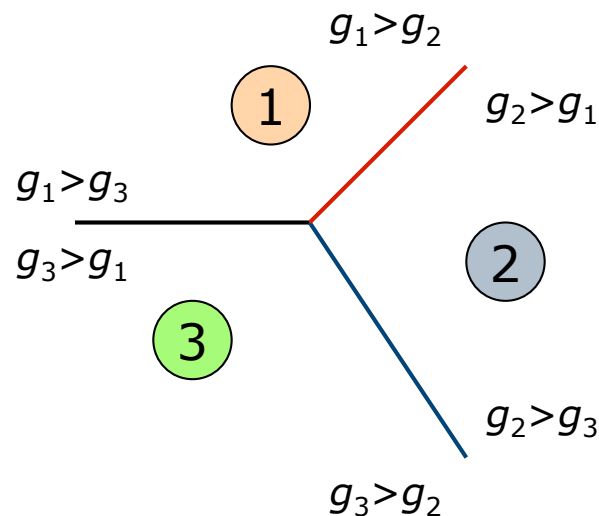
Cas multi-classes ($c > 2$)

3. définir c fonctions discriminantes linéaires

- décision pour la classe ω_i :

$$\text{si } g_i(\mathbf{x}) > g_j(\mathbf{x}), \forall j \neq i$$

- on peut montrer que les frontières des régions ainsi définies sont **convexes**



A.2. Classifieur à fonction discriminante linéaire

Formalisation

- En entrée:
 - n exemples $\mathbf{x} \in \mathbb{R}^d$,
avec attribut de classe (ω_1 ou ω_2)
 - on recherche

$$g(\mathbf{x}) = \mathbf{a}^t \cdot \mathbf{x} + a_0$$

$$\text{tel que } \begin{cases} g(\mathbf{x}) > 0 & \text{pour } \mathbf{x} \in \omega_1 \\ g(\mathbf{x}) < 0 & \text{pour } \mathbf{x} \in \omega_2 \end{cases}$$

A.2. Classifieur à fonction discriminante linéaire

Simplification des notations (1)

- Suppression du seuil

$$\begin{aligned} g(\mathbf{x}) &= \mathbf{a}^t \cdot \mathbf{x} + a_0 \\ &= (a_1 \cdots a_d) \begin{pmatrix} x_1 \\ \vdots \\ x_d \end{pmatrix} + a_0 \\ &= (a_0 \ a_1 \cdots a_d) \begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{pmatrix} \\ &= \bar{\mathbf{a}}^t \bar{\mathbf{x}} \end{aligned}$$

A.2. Classifieur à fonction discriminante linéaire

Simplification des notations (2)

- Suppression des labels de classe

$$\forall \mathbf{x} \in \omega_2, g(\bar{\mathbf{x}}) < 0$$

$$\Leftrightarrow \bar{\mathbf{a}}^t \cdot \bar{\mathbf{x}} < 0$$

$$\Leftrightarrow \bar{\mathbf{a}}^t \cdot (-\bar{\mathbf{x}}) > 0$$

$$\Rightarrow \forall \mathbf{x}, g(\tilde{\mathbf{x}}) > 0 \text{ avec}$$

$$\left\{ \begin{array}{l} \tilde{\mathbf{x}} = \begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{pmatrix} \text{ si } \mathbf{x} \in \omega_1 \\ \tilde{\mathbf{x}} = \begin{pmatrix} -1 \\ -x_1 \\ \vdots \\ -x_d \end{pmatrix} \text{ si } \mathbf{x} \in \omega_2 \end{array} \right.$$

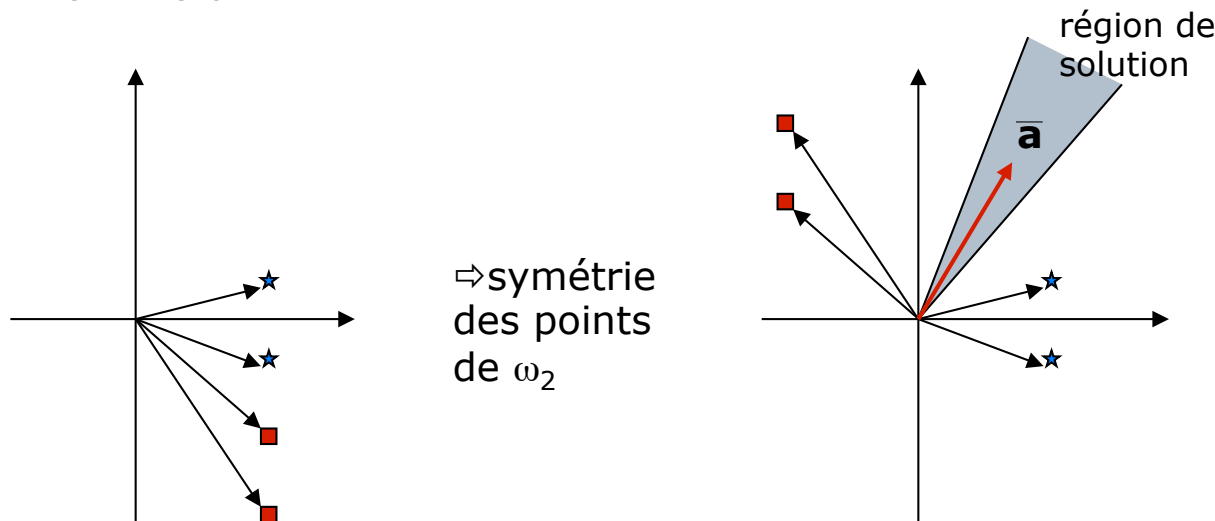
A.2. Classifieur à fonction discriminante linéaire

Recherche de la solution

■ Unicité de la solution $\bar{\mathbf{a}}$?

$$\forall \mathbf{x}, g(\tilde{\mathbf{x}}) = \bar{\mathbf{a}}^t \cdot \tilde{\mathbf{x}} > 0$$

⇒ la solution, si elle existe, doit être dans l'intersection des $\frac{1}{2}$ plans positifs construits avec tous les vecteurs normaux



⇒ il n'y a pas de solution unique (quel critère de qualité?)

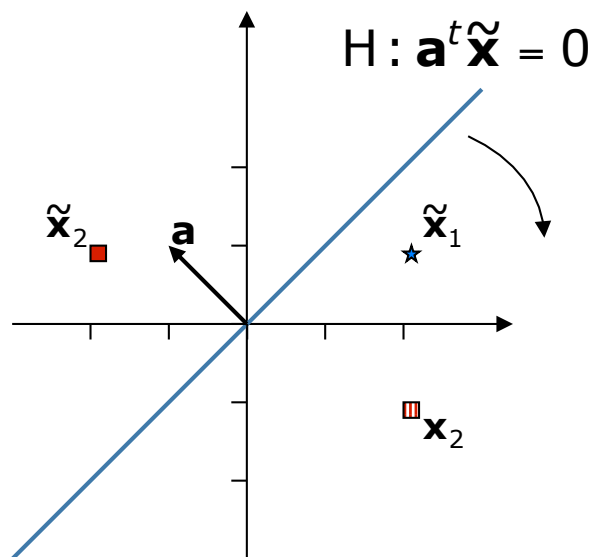
A.2. Classifieur à fonction discriminante linéaire

Algorithme du perceptron (1)

- Principe de l'algorithme ($a_0=0$ pour simplifier)

$$\omega_1 : \mathbf{x}_1 = \begin{pmatrix} 2 \\ 1 \end{pmatrix}, \omega_2 : \mathbf{x}_2 = \begin{pmatrix} 2 \\ -1 \end{pmatrix} \Rightarrow \tilde{\mathbf{x}}_1 = \begin{pmatrix} 2 \\ 1 \end{pmatrix}, \tilde{\mathbf{x}}_2 = \begin{pmatrix} -2 \\ 1 \end{pmatrix}$$

- 1^{ère} itération: $\mathbf{a} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$



$$\mathbf{a}^t \tilde{\mathbf{x}}_1 = -1 \Rightarrow g(\tilde{\mathbf{x}}_1) < 0$$

$$\mathbf{a}^t \tilde{\mathbf{x}}_2 = 3 \Rightarrow g(\tilde{\mathbf{x}}_2) > 0$$

- $\tilde{\mathbf{x}}_1$ est mal classé:
il faut tourner la droite H vers $\tilde{\mathbf{x}}_1$
- par exemple:

$$\mathbf{a}' = \mathbf{a}_{\text{initial}} + \tilde{\mathbf{x}}_1$$

A.2. Classifieur à fonction discriminante linéaire

Algorithme du perceptron (2)

■ 2^e itération: $\mathbf{a}' = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$

$$\tilde{\mathbf{x}}_1 = \begin{pmatrix} 2 \\ 1 \end{pmatrix}, \tilde{\mathbf{x}}_2 = \begin{pmatrix} -2 \\ 1 \end{pmatrix}$$

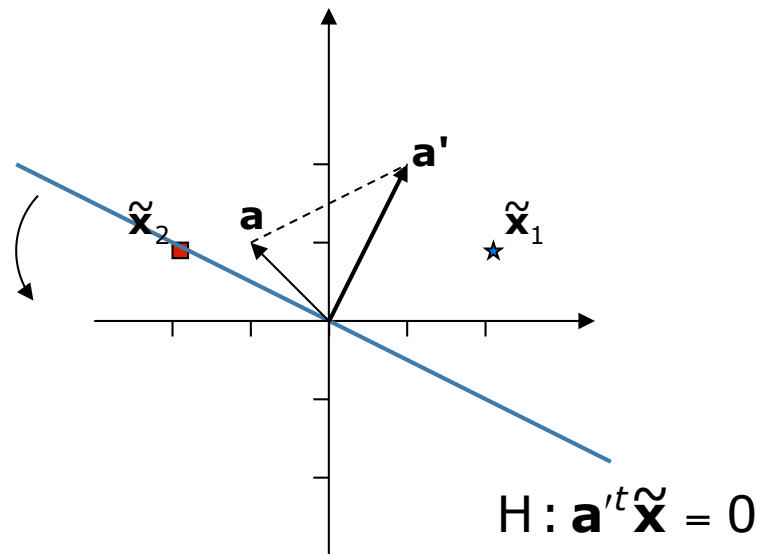
$$\mathbf{a}'^t \tilde{\mathbf{x}}_1 = 4 \Rightarrow g(\tilde{\mathbf{x}}_1) > 0$$

$$\mathbf{a}'^t \tilde{\mathbf{x}}_2 = 0 \Rightarrow g(\tilde{\mathbf{x}}_2) = 0$$

$\Rightarrow \tilde{\mathbf{x}}_1$ bien classé, $\tilde{\mathbf{x}}_2$ à la frontière

il faut tourner H vers $\tilde{\mathbf{x}}_2$

$$\mathbf{a}'' = \mathbf{a}' + \tilde{\mathbf{x}}_2$$



A.2. Classifieur à fonction discriminante linéaire

Algorithme du perceptron (3)

■ 3^e itération: $\mathbf{a}'' = \begin{pmatrix} -1 \\ 3 \end{pmatrix}$

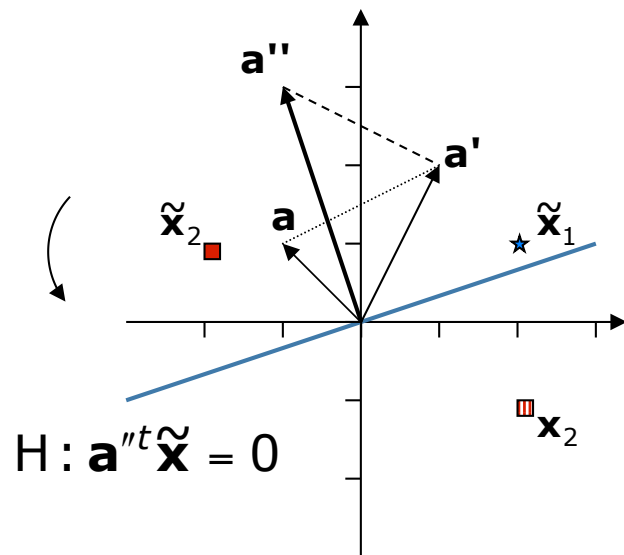
$$\tilde{\mathbf{x}}_1 = \begin{pmatrix} 2 \\ 1 \end{pmatrix}, \tilde{\mathbf{x}}_2 = \begin{pmatrix} -2 \\ 1 \end{pmatrix}$$

$$\mathbf{a}''^t \tilde{\mathbf{x}}_1 = 1 \Rightarrow g(\tilde{\mathbf{x}}_1) > 0$$

$$\mathbf{a}''^t \tilde{\mathbf{x}}_2 = 5 \Rightarrow g(\tilde{\mathbf{x}}_2) > 0$$

$\Rightarrow \tilde{\mathbf{x}}_1$ et $\tilde{\mathbf{x}}_2$ sont bien classés

et \mathbf{x}_1 et \mathbf{x}_2 sont chacun d'un côté de la frontière de décision H



A.2. Classifieur à fonction discriminante linéaire

Algorithme du perceptron (4)

■ Résumé

- normaliser les exemples
 - ajouter une composante $x_0=1$
 - inverser tous les exemples de la classe ω_2
- algorithme
 - \mathbf{a} initial arbitraire
 - tant qu'il existe \mathbf{x} tel que $g(\mathbf{x}) < 0$, faire
 $\mathbf{a}' \leftarrow \mathbf{a} + \mathbf{x}$

■ Variantes

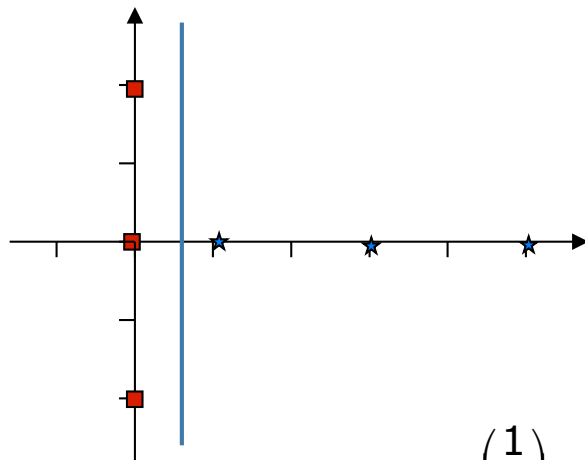
- présentation des exemples
 - cyclique (continuer après une erreur)
 - séquentielle (modifier \mathbf{a} et recommencer au début après une erreur)

A.2. Classifieur à fonction discriminante linéaire

Algorithme du perceptron: exemple

$$\omega_1 : \left\{ \begin{pmatrix} 0 \\ 2 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ -2 \end{pmatrix} \right\} \quad \bar{\mathbf{a}} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

$$\omega_2 : \left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 3 \\ 0 \end{pmatrix} \begin{pmatrix} 5 \\ 0 \end{pmatrix} \right\}$$



Frontière
de décision: $(1 \ -2 \ 0) \begin{pmatrix} 1 \\ x \\ y \end{pmatrix} = 0$

$$\Leftrightarrow 1 - 2x = 0 \quad \Leftrightarrow x = \frac{1}{2}$$

	$\begin{pmatrix} 1 \\ 0 \\ 2 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 0 \\ -2 \end{pmatrix}$	$\begin{pmatrix} -1 \\ -1 \\ 0 \end{pmatrix}$	$\begin{pmatrix} -1 \\ -3 \\ 0 \end{pmatrix}$	$\begin{pmatrix} -1 \\ -5 \\ 0 \end{pmatrix}$
$\bar{\mathbf{a}}_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$						
$\bar{\mathbf{a}}_1 = \begin{pmatrix} \\ \\ \end{pmatrix}$						
$\bar{\mathbf{a}}_2 = \begin{pmatrix} \\ \\ \end{pmatrix}$						
$\bar{\mathbf{a}}_3 = \begin{pmatrix} \\ \\ \end{pmatrix}$						
$\bar{\mathbf{a}}_4 = \begin{pmatrix} \\ \\ \end{pmatrix}$						

Règle de décision: $X < \frac{1}{2} : \omega_1 \quad X > \frac{1}{2} : \omega_2$

A.2. Classifieur à fonction discriminante linéaire

Algorithme du perceptron

■ Problèmes

- l'algorithme ne converge pas lorsque la solution n'existe pas! (problème non linéairement séparable)
- si une solution existe, l'algorithme converge après un nombre fini d'itérations, mais on n'a pas de critère de qualité de la solution

■ Critère proposé

- minimiser la fonction $J = \sum_{\substack{\text{exemple } \mathbf{x} \\ \text{mal classé}}} -\mathbf{a}^t \mathbf{x}$
- descente du gradient $\mathbf{a}' = \mathbf{a} - \varepsilon \frac{\partial J}{\partial \mathbf{a}} = \mathbf{a} + \varepsilon \sum_{\substack{\text{exemple } \mathbf{x} \\ \text{mal classé}}} \mathbf{x}$
- ⇒ déterminer l'ensemble des points mal classés à chaque itération, et incrémenter le vecteur \mathbf{a} de leur somme

A.2. Classifieur à fonction discriminante linéaire

Procédure à relaxation

- Contrainte de seuil minimum b

$$\mathbf{a}_{k+1} = \mathbf{a}_k + \rho \sum_{\substack{\mathbf{x} \text{ tel que} \\ \mathbf{a}_k^t \mathbf{x} \leq b}} \frac{b - \mathbf{a}_k^t \mathbf{x}}{\|\mathbf{x}\|^2} \mathbf{x} \quad (0 < \rho < 2)$$

- procédure à correction d'erreur (comme le perceptron)
 - ne converge que pour les cas linéairement séparables
 - pas forcément en un nombre fini d'itérations (\neq perceptron)
 - dépend de la dimension d vs. le nombre d'exemples n
 - $n < 2d \Rightarrow$ généralement linéairement séparable
- cas non séparable
 - détection de non convergence?
 - Perceptron: cycle des solutions dans le cas de valeurs entières
 - moyenne des derniers vecteurs de poids \mathbf{a}_k
 - évite de choisir une mauvaise solution

A.2. Classifieur à fonction discriminante linéaire

Autres algorithmes

- Minimisation de la distance quadratique

$$\sum_{\mathbf{x}} \|\mathbf{a}^t \mathbf{x} - b\|^2$$

- Procédure de Widrow-Hoff

$$\mathbf{a}_{k+1} = \mathbf{a}_k + \rho_k \sum_{\mathbf{x}} (b - \mathbf{a}^t \mathbf{x}) \mathbf{x} \quad (\rho_k = \frac{\rho_1}{k})$$

- prend en compte tous les exemples
- interruption lorsque $|\mathbf{a}_{k+1} - \mathbf{a}_k| < \varepsilon$
- cas linéairement séparable: séparation pas garantie
- cas non linéairement séparable: solution acceptable
- Procédure de Ho-Kashyap
 - combinaison des propriétés Perceptron + Widrow-Hoff
 - minimise la distance quadratique + séparation linéaire

A.2. Classifieur à fonction discriminante linéaire Support Vector Machines (SVM)

- Maximisation de la marge

$$\forall k, \bar{\mathbf{a}}^t \cdot \tilde{\mathbf{y}}_k \geq 1$$

- Les vecteurs support sont sur la marge

$$\bar{\mathbf{a}}^t \cdot \tilde{\mathbf{y}}_k = 1$$

- Dans les cas non linéaire

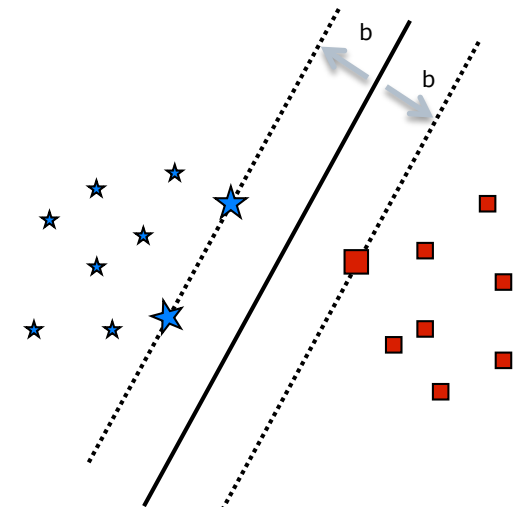
- projeter les données dans un espace de très grande dimension rend le problème linéairement séparable

$$\mathbf{y}_k = \varphi(\mathbf{x}_k)$$

- Il n'est pas nécessaire de calculer explicitement \mathbf{y} si on dispose d'une fonction noyau tel que

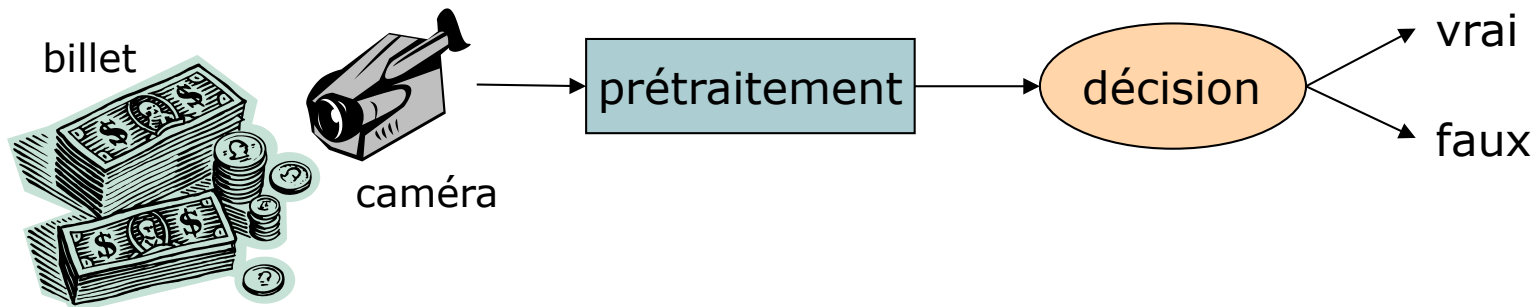
$$k(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)$$

- Classifieur à 2 classes très populaire



A.3. Classifieur Bayésien

- Approche plus générale en classification supervisée
 - caractérisation des classes
 - trouver la distribution de probabilité des observations pour chaque classe
 - règle de décision Bayésienne
 - choisir la classe qui **minimise le risque d'erreur**
- Exemple
 - détecteur de faux billets



A.3. Classifieur Bayésien

Connaissances *a priori*

- Probabilités a priori
 - le nombre d'exemples d'apprentissage peut varier d'une classe à l'autre
 - on peut utiliser ce nombre d'exemples pour déterminer les probabilités a priori $P(\omega_1) \dots P(\omega_C)$:
 - fréquence relative des classes dans base d'apprentissage:
 $P(\omega_i) = n_i / n$
 - utilisation pour un problème à 2 classes:
 - si on connaît uniquement $P(\omega_1)$ et $P(\omega_2)$?
(on a: $P(\omega_2) + P(\omega_1) = 1$)
 - choisir la classe la plus probable!
(surtout si $P(\omega_1) \gg P(\omega_2)$...)
- Prise en compte de l'observation ?

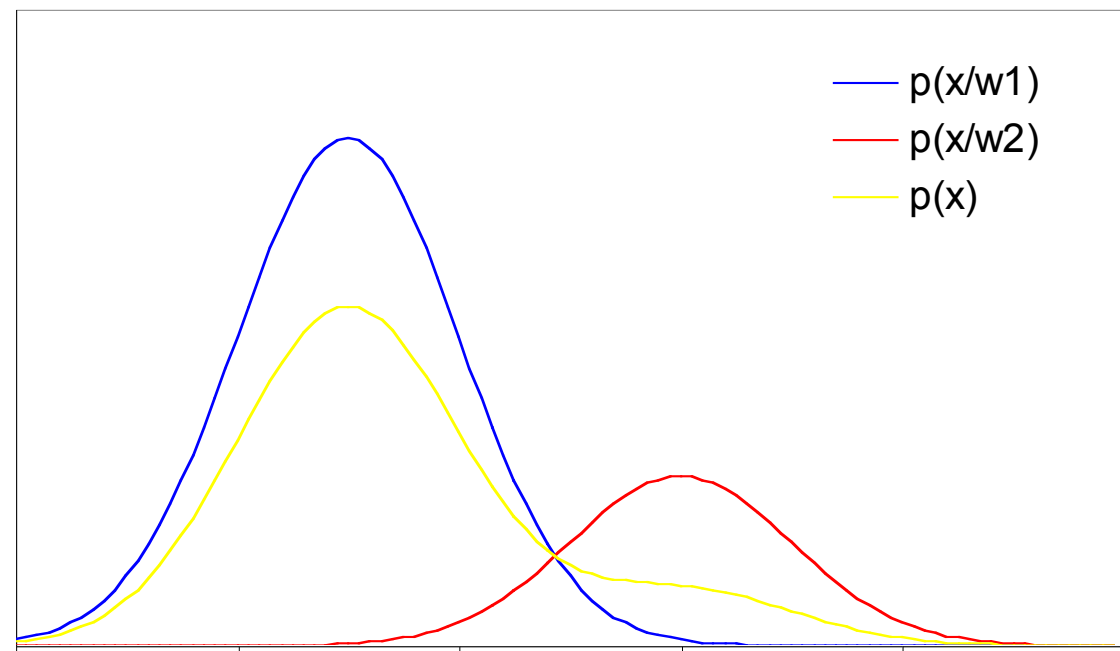
A.3. Classifieur Bayésien

Vraisemblance conditionnelles

- La valeur de l'observation x dépend de la classe
 - densité de probabilité de x conditionnelle à la classe ω
 - $p(x/\omega_1)$: probabilité d'une mesure x pour un vrai billet
 - $p(x/\omega_2)$: " " " " pour un faux billet

- densité de probabilité de x quelque soit la nature du billet

$$p(x) = p(x/\omega_1)P(\omega_1) + p(x/\omega_2)P(\omega_2)$$



A.3. Classifieur Bayésien

Probabilités *a posteriori*

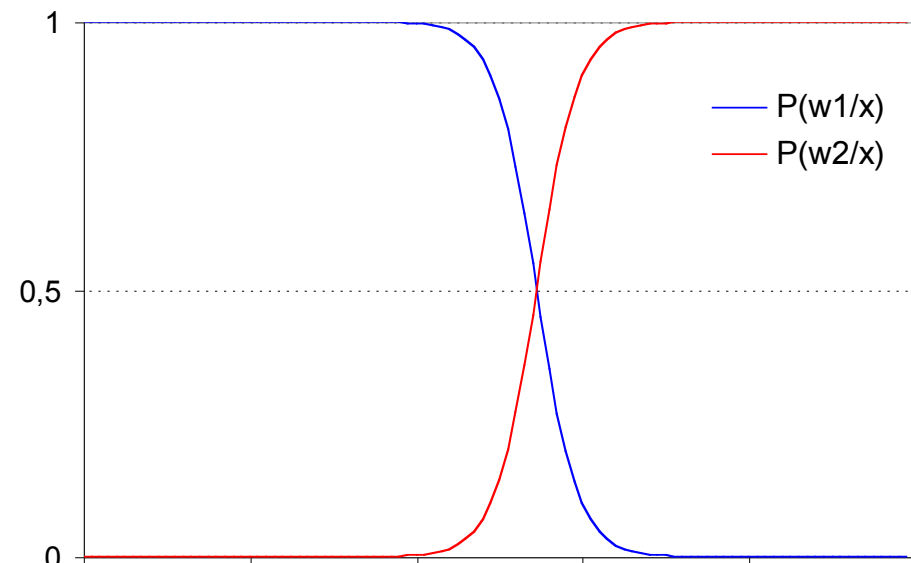
- Prise en compte de la valeur de x dans la décision
 - Probabilité *a posteriori* de ω_1 et ω_2
 - $P(\omega_1/x)$: probabilité que le billet soit vrai en sachant la mesure x
 - $P(\omega_2/x)$: " " " faux " " "

- Calcul de $p(\omega_i/x)$?
 - formule de Bayes

$$P(\omega_1/x) = \frac{p(x/\omega_1)P(\omega_1)}{p(x)}$$

$$P(\omega_2/x) = \frac{p(x/\omega_2)P(\omega_2)}{p(x)}$$

$$\forall x, P(\omega_1/x) + P(\omega_2/x) = 1$$



A.3. Classifieur Bayésien

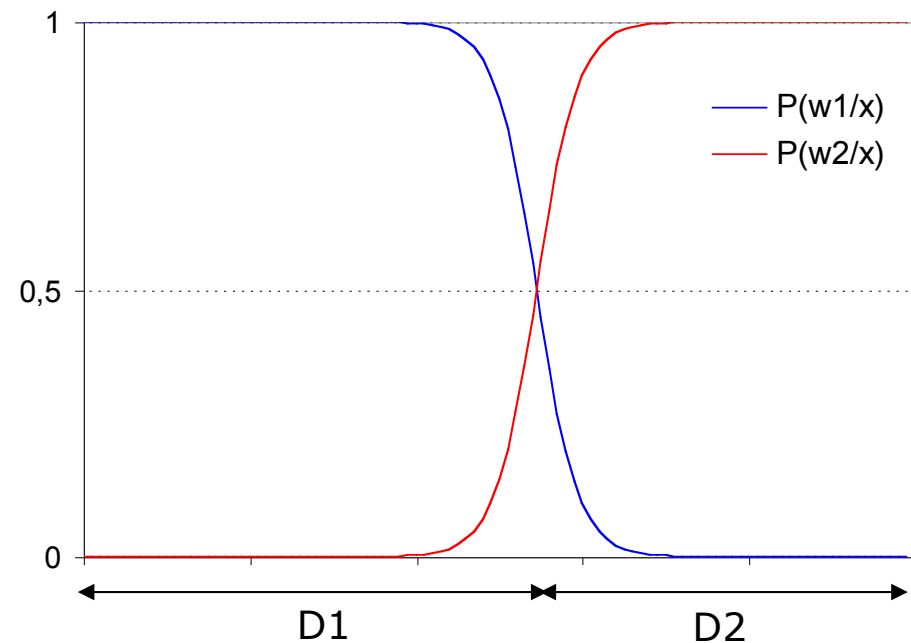
Règle de décision

- Choix de la classe la plus probable, dans l'état des connaissances

- Décider ω_1 si $P(\omega_1/x) > P(\omega_2/x)$
 ω_2 sinon

- Règle équivalente:
 ω_1 si $p(x/\omega_1)P(\omega_1) > p(x/\omega_2)P(\omega_2)$
 ω_2 sinon

⇒ Régions de décision
 D_1 et D_2

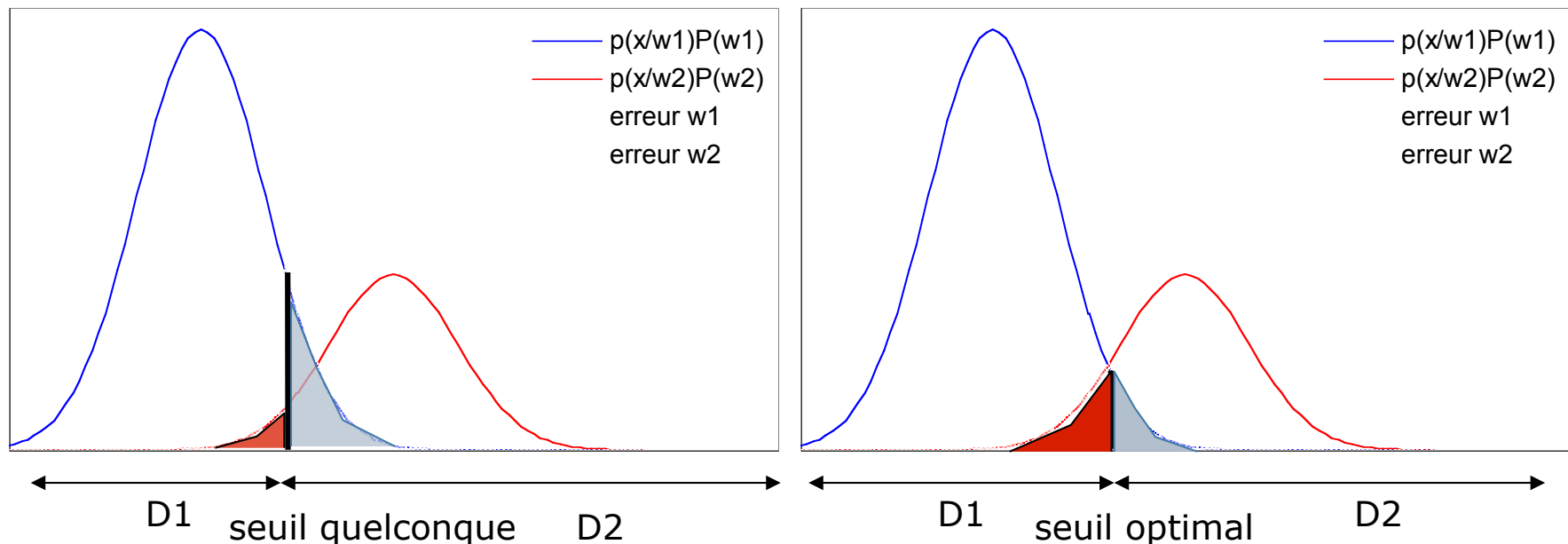


A.3. Classifieur Bayésien

Probabilité d'erreur

$$\begin{aligned} P(\text{erreur}) &= P(\omega_1, \text{décide } \omega_2) + P(\omega_2, \text{décide } \omega_1) \\ &= P(\omega_1)P(x \in D_2 / \omega_1) + P(\omega_2)P(x \in D_1 / \omega_2) \\ &= \int_{D_2} p(x / \omega_1)P(\omega_1)dx + \int_{D_1} p(x / \omega_2)P(\omega_2)dx \end{aligned}$$

Le classifieur Bayésien est optimal dans le sens où il **minimise le taux d'erreur**



A.3. Classifieur Bayésien

Cas général

- Probabilité *a posteriori* de la classe i , en appliquant la règle de Bayès:

$$P(\omega_i / \mathbf{x}) = \frac{p(\mathbf{x} / \omega_i) \cdot P(\omega_i)}{p(\mathbf{x})} \quad \text{avec} \quad p(\mathbf{x}) = \sum_{j=1}^c p(\mathbf{x} / \omega_j) \cdot P(\omega_j)$$

- Choix de la classe la plus probable au vu de l'observation

$$\omega = \operatorname{argmax}_{\omega_i} P(\omega_i / \mathbf{x}) = \operatorname{argmax}_{\omega_i} \frac{p(\mathbf{x} / \omega_i) \cdot P(\omega_i)}{p(\mathbf{x})}$$

or, la probabilité a priori de \mathbf{x} est indépendante de ω_i , donc:

$$\omega = \operatorname{argmax}_{\omega_i} p(\mathbf{x} / \omega_i) \cdot P(\omega_i)$$

- Probabilité d'erreur: $P(\text{erreur}) = 1 - \sum_{i=1}^c \int_{D_i} p(\mathbf{x} / \omega_i) P(\omega_i) d\mathbf{x}$

A.3. Classifieur Bayésien

Décision avec coûts (1)

- Plusieurs actions possibles $A = \{\alpha_1, \alpha_2, \dots, \alpha_s\}$
 - décider une classe ω_i , rejeter (pas de prise de décision)
- Fonction de coût $\lambda_{ij} = \lambda(\alpha_i / \omega_j)$
 - coût de l'action α_i si la classe est ω_j
- Risque conditionnel $R(\alpha_i / \mathbf{x})$
 - risque associé à l'action α_i sachant \mathbf{x}
 - si la classe est ω_j , le coût est λ_{ij}
 - sur l'ensemble des classes le risque de l'action α_i est
$$R(\alpha_i / \mathbf{x}) = \sum_{j=1}^c \lambda(\alpha_i / \omega_j) P(\omega_j / \mathbf{x})$$
- Règle de décision
 - choisir l'action α_i si $R(\alpha_i / \mathbf{x}) < R(\alpha_j / \mathbf{x}), \forall j \neq i$
 - action qui **minimise le risque conditionnel**

A.3. Classifieur Bayésien

Décision avec coûts (2)

- Cas particulier

- action $\alpha_i \equiv$ choisir la classe ω_i
- fonctions de coût

$$\begin{cases} \lambda(\alpha_i / \omega_i) = 0 & \text{bonne décision} \\ \lambda(\alpha_i / \omega_j) = 1, \forall j \neq i & \text{mauvaise décision} \end{cases}$$

$$\Rightarrow R(\alpha_i / \mathbf{x}) = \sum_{j \neq i} P(\omega_j / \mathbf{x}) = 1 - P(\omega_i / \mathbf{x})$$

- minimiser $R(\alpha_i / \mathbf{x})$ revient à maximiser $P(\omega_i / \mathbf{x})$
- on retrouve le classifieur à taux d'erreur minimum

A.3. Classifieur Bayésien

Décision avec coûts - exemple

- 2 classes
 - ω_1 vrai billet, $P(\omega_1)=0,6$
 - ω_2 faux billet, $P(\omega_2)=0,4$
- Fonction de coût
 - $\lambda(\alpha_1 / \omega_1) = \lambda_{11} = 1\text{€}$ accepter un vrai billet (test)
 - $\lambda(\alpha_1 / \omega_2) = \lambda_{12} = 101\text{€}$ accepter un faux billet (test+perte)
 - $\lambda(\alpha_2 / \omega_1) = \lambda_{21} = 11\text{€}$ refuser un vrai billet (préjudice commercial)
 - $\lambda(\alpha_2 / \omega_2) = \lambda_{22} = 1\text{€}$ refuser un faux billet (test)
- Règle de décision
 - choisir α_1 si $R(\alpha_1 / \mathbf{x}) < R(\alpha_2 / \mathbf{x})$
 - $\Leftrightarrow \lambda_{11} P(\omega_1 / \mathbf{x}) + \lambda_{12} P(\omega_2 / \mathbf{x}) < \lambda_{21} P(\omega_1 / \mathbf{x}) + \lambda_{22} P(\omega_2 / \mathbf{x})$
 - $\Leftrightarrow (\lambda_{21} - \lambda_{11})p(\mathbf{x} / \omega_1)P(\omega_1) > (\lambda_{12} - \lambda_{22})p(\mathbf{x} / \omega_2)P(\omega_2)$
 - $\Leftrightarrow 10 \times 0,6 \times p(\mathbf{x} / \omega_1) > 100 \times 0,4 \times p(\mathbf{x} / \omega_2)$
 - modification des régions de décision

A.3. Classifieur Bayésien

Fonctions discriminantes

- Cas général

$$g_i(\mathbf{x}) = -R(\alpha_i/\mathbf{x})$$

- Taux minimal d'erreur

$$g_i(\mathbf{x}) = P(\omega_i/\mathbf{x})$$

- Fonctions équivalentes

- décision invariante par

- biais additif

- constante multiplicative

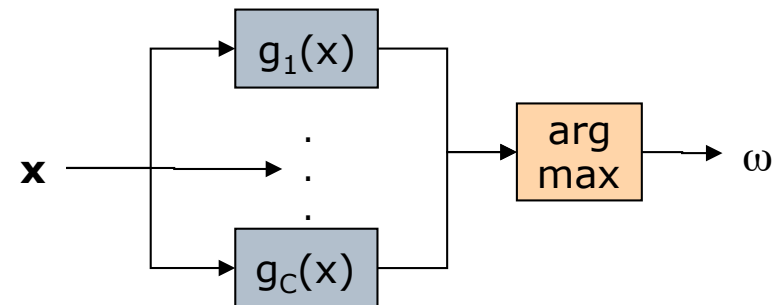
- composition par une fonction monotone

- exemples

$$g_i(\mathbf{x}) = P(\omega_i/\mathbf{x})$$

$$g_i(\mathbf{x}) = p(\mathbf{x}/\omega_i)P(\omega_i)$$

$$g_i(\mathbf{x}) = \log p(\mathbf{x}/\omega_i) + \log P(\omega_i)$$



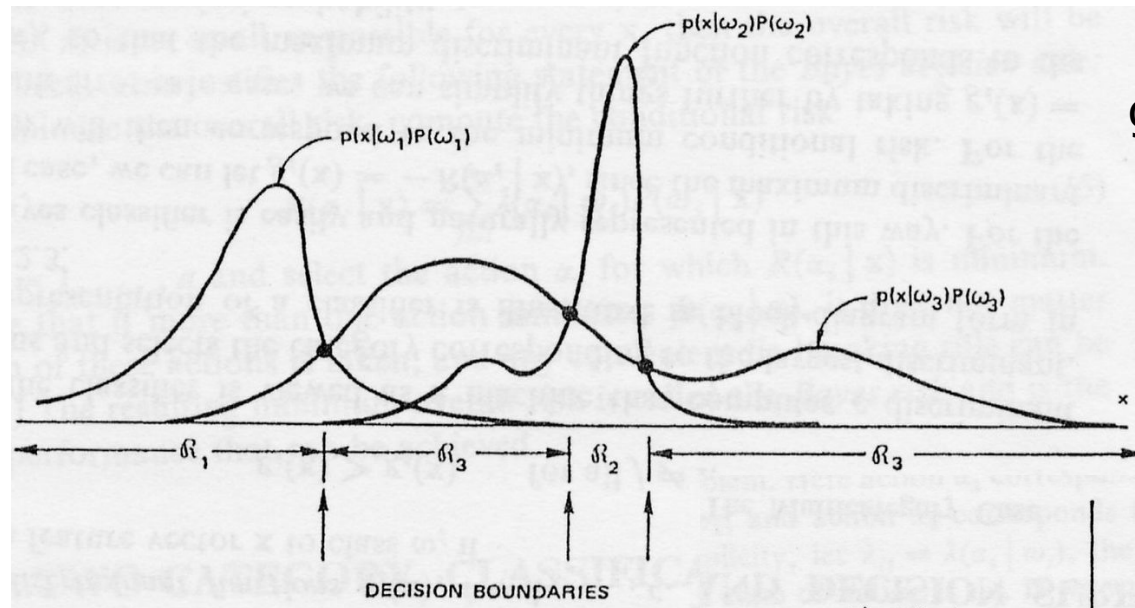
fonctions équivalentes

⇒ décisions identiques

⇒ régions de décision
identiques

A.3. Classifieur Bayésien

Frontières de décision



$$g_i(\mathbf{x}) = g_j(\mathbf{x}), i \neq j$$

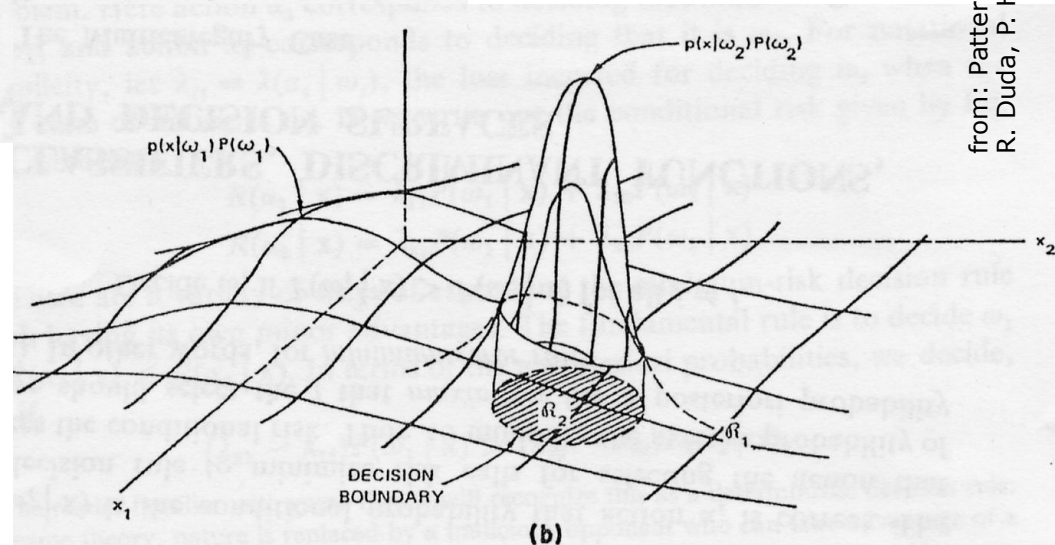


FIGURE 2.4. Examples of decision boundaries and decision regions.

from: Pattern Classification and Scene Analysis,
R. Duda, P. Hart, Wiley-Interscience

A.3. Classifieur Bayésien

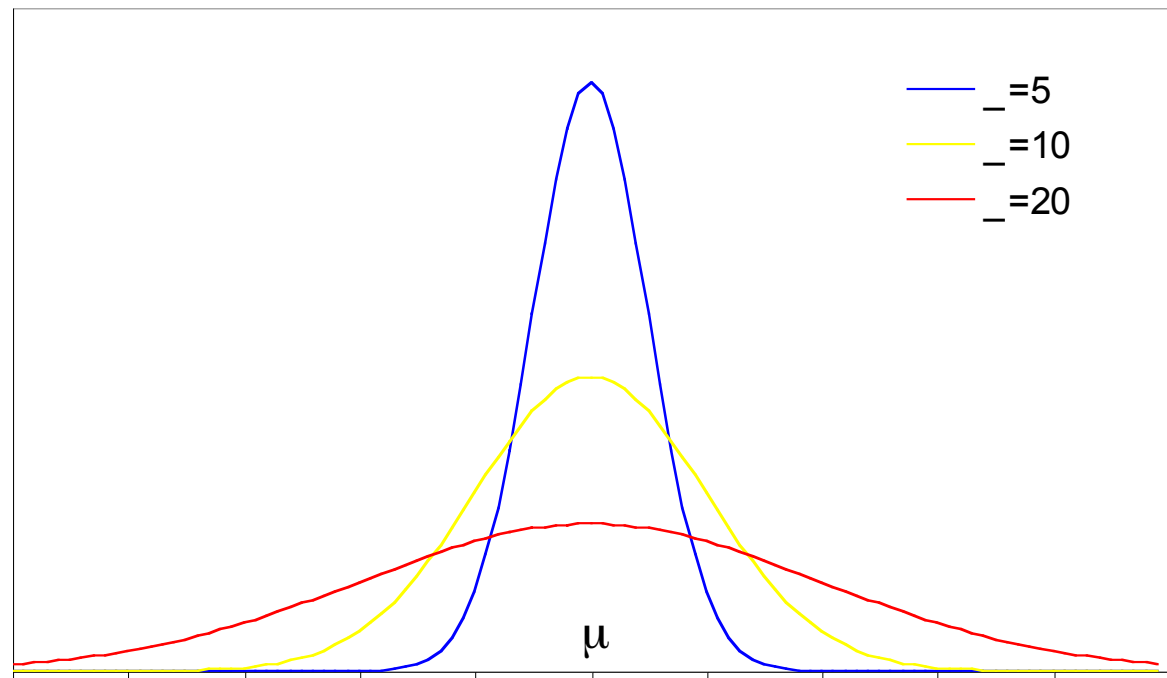
Loi normale

- En dimension 1:

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \quad \text{avec} \quad \begin{cases} \mu = E[x] & \text{moyenne} \\ \sigma^2 = E[(x-\mu)^2] & \text{variance} \end{cases}$$

$$p(x) \sim N(\mu, \sigma^2)$$

~95% de la courbe
entre $\mu-2\sigma$ et $\mu+2\sigma$



A.3. Classifieur Bayésien

Loi normale

- En dimension d:

$$p(\mathbf{x}) \sim N(\mu, \Sigma) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(\mathbf{x} - \mu)^t \Sigma^{-1} (\mathbf{x} - \mu)}$$

avec $\begin{cases} \mu = E[\mathbf{x}] \\ \Sigma = E[(\mathbf{x} - \mu)(\mathbf{x} - \mu)^t] \end{cases}$ vecteur moyenne
matrice de covariance

- Σ matrice de covariance symétrique, définie positive
- si les d dimensions sont indépendantes, alors Σ est diagonale

$$\Rightarrow p(\mathbf{x}) = \prod_{i=1}^d p(x_i)$$

produit de densités de probabilités normales de dimension 1

A.3. Classifieur Bayésien

Loi normale

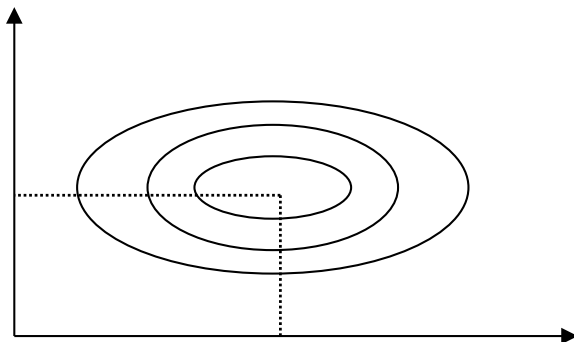
■ En dimension 2: $\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ $\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}$ $\Sigma = \begin{pmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{pmatrix}$

■ courbes d'équidensité

$$p(\mathbf{x}) = K \Rightarrow (\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu) = K'$$

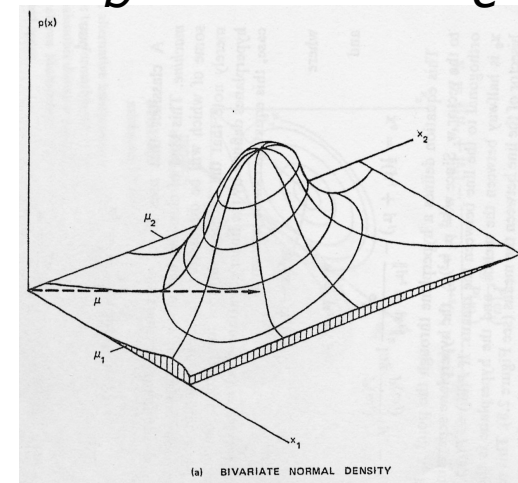
$$\sigma_{12} = 0$$

$$\frac{(x_1 - \mu_1)^2}{\sigma_1^2} + \frac{(x_2 - \mu_2)^2}{\sigma_2^2} = K''$$



$$\sigma_{12} \neq 0$$

$$\frac{(x_1 - \mu_1)^2}{a} + \frac{(x_2 - \mu_2)^2}{b} + \frac{(x_1 - \mu_1)(x_2 - \mu_2)}{c} = K'''$$



from: Pattern Classification and
Scene Analysis, R. Duda, P.
Hart, Wiley-Interscience

A.3. Classifieur Bayésien

Fonctions discriminantes pour loi normale

- Fonctions discriminantes pour une distribution normale des données

$$p(\mathbf{x} / \omega_i) \sim N(\mu_i, \Sigma_i)$$

- 1er cas particulier $\Sigma_i = \sigma^2 I = \begin{pmatrix} \sigma^2 & & 0 \\ & \ddots & \\ 0 & & \sigma^2 \end{pmatrix}$

$$\begin{aligned} g_i(\mathbf{x}) &= \log p(\mathbf{x} / \omega_i) + \log P(\omega_i) \\ &= K - \frac{\|\mathbf{x} - \mu_i\|^2}{2\sigma^2} + \log P(\omega_i) \end{aligned}$$

- si en plus $P(\omega_i) = P(\omega_j)$, on peut simplifier: σ^2 disparaît

$$g_i(\mathbf{x}) = -\|\mathbf{x} - \mu_i\|^2$$

classifieur à distance minimum (distance euclidienne)

A.3. Classifieur Bayésien

Fonctions discriminantes pour loi normale

- 2ème cas particulier $\Sigma_i = \Sigma$
 - matrice de covariance identique pour toutes les classes

$$g_i(\mathbf{x}) = \log P(\omega_i) - \frac{1}{2} (\mathbf{x} - \mu_i)^t \Sigma^{-1} (\mathbf{x} - \mu_i)$$

- si en plus $P(\omega_i) = P(\omega_j)$

$$g_i(\mathbf{x}) = -\frac{1}{2} (\mathbf{x} - \mu_i)^t \Sigma^{-1} (\mathbf{x} - \mu_i)$$

- on retrouve le classifieur à distance minimum (associé à la distance de Mahalanobis)

- Cas général - Σ_i est quelconque

$$g_i(\mathbf{x}) = \log P(\omega_i) - \frac{1}{2} \log |\Sigma_i| - \frac{1}{2} (\mathbf{x} - \mu_i)^t \Sigma_i^{-1} (\mathbf{x} - \mu_i)$$

- fonctions discriminantes quadratiques

A.3. Classifieur Bayésien

Estimation paramétrique

- On suppose connu le type des lois de probabilité $p(\mathbf{x} / \omega_i)$
 - il faut estimer les paramètres de ces lois par des statistiques sur la base d'apprentissage
- Cas d'une distribution normale $p(\mathbf{x} / \omega_i) \sim N(\mu_i, \Sigma_i)$
 - il faut estimer la moyenne et la matrice de covariance
 - estimation par maximum de vraisemblance
 - (valeur maximisant la probabilité que les données d'apprentissage aient été générées suivant la densité)

dimension 1:

$$\begin{cases} \hat{\mu} = \frac{1}{n} \sum_{k=1}^n x_k \\ \hat{\sigma}^2 = \frac{1}{n} \sum_{k=1}^n (x_k - \hat{\mu})^2 \end{cases}$$

dimension d:

$$\begin{cases} \hat{\mu} = \frac{1}{n} \sum_{k=1}^n \mathbf{x}_k \\ \hat{\Sigma} = \frac{1}{n} \sum_{k=1}^n (\mathbf{x}_k - \hat{\mu})(\mathbf{x}_k - \hat{\mu})^t \end{cases}$$

A.3. Classifieur Bayésien

Estimation non paramétrique

- Principe: estimation directe de la densité de probabilité $p(\mathbf{x})$ à partir des données observées autour de \mathbf{x}
- Hypothèse:
 - densité $p(\mathbf{x})$ fonction continue
 - R un voisinage de \mathbf{x} de volume V
 - peu de variation de $p(\mathbf{x})$ dans R
- Alors
$$P(x' \in R) = \int_R p(x') dx' \approx p(x) \int_R dx'$$
$$\Rightarrow \hat{p}(x) = \frac{P(x' \in R)}{V}$$
- Soient n tirages suivant la densité $p(x)$
 - Probabilité que k échantillons appartiennent à R
$$E[k] = n.P(x' \in R)$$

A.3. Classifieur Bayésien

Estimation non paramétrique

- Il faut se rapprocher assez de \mathbf{x} pour que les hypothèses sur $p(\mathbf{x})$ soient valables, sans que le voisinage soit vide
 - soient
 - n : le nombre d'échantillons observés
 - V_n : le volume d'une région de \mathbf{R}^d centrée sur \mathbf{x}
 - k_n : le nombre de points dans le volume V_n
 - alors $\hat{p}_n(\mathbf{x}) = k_n / (n \times V_n)$
 - convergence
$$\hat{p}_n(\mathbf{x}) \xrightarrow{n \rightarrow \infty} p(\mathbf{x}) \quad \text{si} \quad \begin{cases} \lim_{n \rightarrow \infty} V_n = 0 \\ \lim_{n \rightarrow \infty} k_n = \infty \\ \lim_{n \rightarrow \infty} k_n / n = 0 \end{cases}$$
- Deux approches
 - contrôler le volume V_n ou le nombre de point k_n

A.3. Classifieur Bayésien

Estimation non paramétrique

- 1ère approche: estimateur de Parzen
 - on fixe le volume, par exemple $V_n = 1/\sqrt{n}$
 - puis on calcule k_n
- Voisinage défini comme un hyper-cube de côté h_n
 - volume $V_n = h_n^d$
 - fonction d'appartenance $\varphi(\mathbf{u}) = \begin{cases} 1 & \max_{k \leq d} |u_k| \leq \frac{1}{2} \\ 0 & \text{sinon} \end{cases}$
 - alors
$$\hat{p}_n(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^n \frac{1}{V_n} \varphi\left(\frac{\mathbf{x} - \mathbf{x}_i}{h_n}\right)$$
- Généralisation: fenêtres de Parzen
$$\varphi(u) \geq 0 \quad \text{et} \quad \int \varphi(u) du = 1$$
 - exemple de noyau $\varphi(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{u^2}{2}}$

A.3. Classifieur Bayésien

Règle des k plus proches voisins

- 2ème approche: estimation non-paramétrique directe de la densité de probabilité $P(\omega_i / \mathbf{x})$
 - soient
 - n : le nombre d'échantillons observés
 - V : le volume d'une région de \mathbf{R}^d centrée sur \mathbf{x}
 - k : le nombre de points dans le volume V
 - k_i : le nombre de points de la classe ω_i dans le volume V
 - alors $\hat{p}(\mathbf{x}; \omega_i) = k_i / (n \times V)$
$$\Rightarrow \hat{P}(\omega_i / \mathbf{x}) = \frac{\hat{p}(\mathbf{x}; \omega_i)}{\sum_j \hat{p}(\mathbf{x}; \omega_j)} = k_i / k$$
 - $P(\omega_i / \mathbf{x})$ est estimé par la proportion de points de la classe ω_i parmi les k plus proches voisins de x
 - Choix de k: ni trop grand, ni trop petit, typiquement $k \sim \sqrt{n}$
- Règle de décision des k-ppv ("k nearest neighbours"):
 - Décider ω_i si $k_i > k_j, \quad \forall i \neq j$

B. Classification non supervisée

- Classification supervisée
 - on dispose d'exemples, et on connaît leur classe
- Classification non-supervisée
 - on dispose d'exemples, sans connaître leurs classes
 - situation
 - étiquetage possible mais coûteux
 - système évolutif, apparition de nouvelles classes
 - meilleure connaissance de la structure des données
 - connaissances a priori ou hypothèses possibles sur:
 - le nombre de classes C
 - la probabilités a priori de chaque classe (?)
 - la forme paramétrique de la densité de probabilité de chaque classe (??)

B. Classification non supervisée

Principes

- Méthodes de coalescence ("clustering")
 - Idée: séparer les données en paquets de points similaires
 - mesure de similarité/dissimilarité entre points?
 - qualité de la partition des données entre paquets?
 - Mesure de similarité
 - distance euclidienne + seuil de distance
 - problème: sensibilité aux changements d'échelle x/y
 - normalisation préalable des données
 - moyenne et variance
 - analyse en composantes principales
 - autres distances: de Mahalanobis, de Minkovski
- $$d_{\lambda}(\mathbf{x}, \mathbf{y}) = \left[\sum_{k=1}^d |x_k - y_k|^{\lambda} \right]^{\frac{1}{\lambda}}$$
- $\lambda=1$: Manhattan/city bloc (valeur absolue)
 $\lambda=2$: euclidienne
 $\lambda \rightarrow \infty$: Chebyshev (max)
- toute mesure de similarité (symétrique)

B. Classification non supervisée

Critère de qualité

- Critère de qualité

- n échantillons $\{\mathbf{x}_1 \dots \mathbf{x}_n\}$, partition H en c paquets disjoints $H_1 \dots H_c$
- Qualité $Q(H)$ à maximiser (question de recherche ouverte!)

- Moindre carrés

- soit μ_i la moyenne du paquet H_i $\mu_i = \frac{1}{n_i} \sum_{\mathbf{x}_j \in H_i} \mathbf{x}_j$

alors la somme des erreurs au carré est $J = \sum_{i=1}^c \sum_{\mathbf{x}_j \in H_i} \|\mathbf{x}_j - \mu_i\|^2$

- partition à **variance minimum**

- adapté pour des nuages de points compacts
- problème si le nombre de point des nuages est déséquilibré

- reformulation

$$J = \frac{1}{2} \sum_{i=1}^c n_i S_i \quad \text{avec} \quad \begin{cases} S_i = \frac{1}{n_i^2} \sum_{\mathbf{x}_j, \mathbf{x}_k \in H_i} \|\mathbf{x}_j - \mathbf{x}_k\|^2 & \text{cas euclidien} \\ S_i = \frac{1}{n_i^2} \sum_{\mathbf{x}_j, \mathbf{x}_k \in H_i} s(\mathbf{x}_j, \mathbf{x}_k) & \text{cas général} \end{cases}$$

B. Classification non supervisée

Recherche de la partition

- Recherche directe
 - explosion combinatoire en fonction de n et c
 - $\sim c^n/c!$ possibilités...
- Recherche par optimisation itérative
 - partition initiale
 - modification de la partition en améliorant le critère de qualité

pb: atteinte d'un optimum local
- Cas du critère $J_{\text{euclidien}}$
 - Le réassignement des échantillons à la classe du centroïde dont il est le plus proche améliore le critère $J_{\text{euclidien}}$
 - l'initialisation est un problème critique
- Procédure des k-moyennes
- Généralisation: nuées dynamiques
- Variante ISODATA
 - regroupement/division pour avoir des homogènes classes

B. Classification non supervisée

Algorithme des k-moyennes

1. Choisir des valeurs initiales $\hat{\mu}_1^0 \dots \hat{\mu}_c^0$
2. Classifier les n échantillons dans la classe pour laquelle ils sont le plus proche de $\hat{\mu}_i$

$$\omega(\mathbf{x}_k) = \underset{i}{\operatorname{argmin}} \|\mathbf{x}_k - \hat{\mu}_i\|^2$$

3. Recalculer la moyenne à partir des points associés à la classe

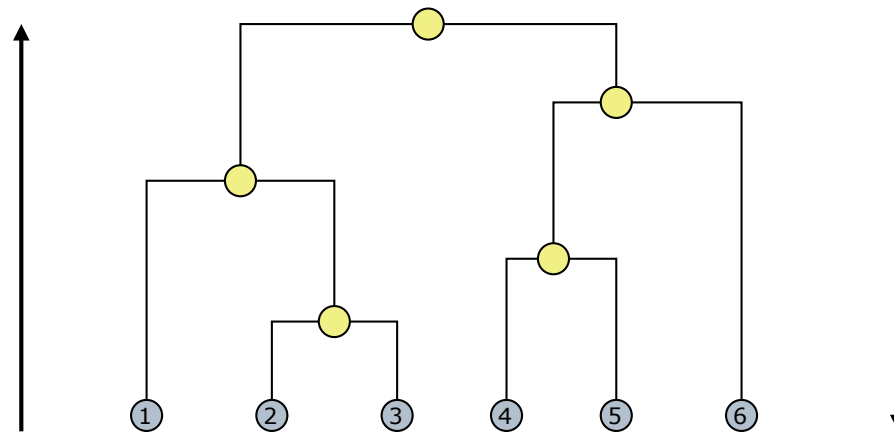
$$\hat{\mu}'_i = \frac{\sum_{\omega(\mathbf{x}_k)=i} \mathbf{x}_k}{\operatorname{card}\{\mathbf{x}_k / \omega(\mathbf{x}_k) = i\}}$$

4. Reboucler à l'étape 2 tant que:
 - il existe i tel que $\hat{\mu}'_i \neq \hat{\mu}_i$
 - le nombre maximal d'itération n'est pas atteint
 - la gain relatif du critère de qualité est trop faible

B. Classification non supervisée

Classification hiérarchique

- Au lieu d'une partition, on considère une séquence de partitions imbriquées:
 - niveau 1: 1 paquet de n éléments
 - niveau n : n paquets de 1 élément



- Approches ascendantes (par agglomération) et descendantes (par division)

B. Classification non supervisée

Méthode par agglomération

■ Principe

1. Initialement, un paquet par classe: $c=n$, $H_i=\{\mathbf{x}_i\}$
2. Choisir les 2 paquets les plus proches et les fusionner
3. Répéter l'étape 2 jusqu'à atteindre le nombre de classe désiré (ou $c=1$, ou autre critère d'arrêt)

■ Distances inter-clusters

$$d_{\min}(H_i, H_j) = \min_{\mathbf{x} \in H_i, \mathbf{y} \in H_j} \|\mathbf{x} - \mathbf{y}\|$$

$$d_{\max}(H_i, H_j) = \max_{\mathbf{x} \in H_i, \mathbf{y} \in H_j} \|\mathbf{x} - \mathbf{y}\|$$

$$d_{\text{avg}}(H_i, H_j) = \frac{1}{n_i n_j} \sum_{\mathbf{x} \in H_i, \mathbf{y} \in H_j} \|\mathbf{x} - \mathbf{y}\|$$

$$d_{\text{mean}}(H_i, H_j) = \|\mu_i - \mu_j\|$$

(équivalentes dans le cas euclidien)

- si le critère dérive de la distance inter-éléments:
 - tous les calculs sont faits à partir de la matrice des distances

■ Généralisation du critère

- Fusionner les 2 paquets tant que le critère de qualité croît

B. Classification non supervisée

Méthode par division

- Principe
 1. Initialement, tous les points dans une classe ($c=1$)
 2. Diviser un ou plusieurs paquets en sous-paquets
 3. Répéter jusqu'à satisfaction du critère d'arrêt
- Quantification vectorielle binaire
 - algorithme LBG (Linde, Buzo, Gray)
 - on part d'un centroïde
 - à chaque itération, le nombre de paquets est doublé en créant de nouveaux centroïdes par perturbation du centroïde initial de chaque classe
$$\begin{cases} \mu_i^+ = \mu_i(1 + \varepsilon) \\ \mu_i^- = \mu_i(1 - \varepsilon) \end{cases}$$
 - les centroïdes sont recalculés par les k-moyennes

B. Classification non supervisée

Densités de probabilité plus complexes

- Une densité de probabilité normale ne suffit pas à modéliser toutes les distributions observées...
- Un mélange de gaussiennes ("gaussian mixture") permet de modéliser des distributions plus complexes

$$p(\mathbf{x}) = \sum_{j=1}^c P(\omega_j) p(\mathbf{x}/\omega_j) \quad \text{avec } p(\mathbf{x} / \omega_j) \sim N(\mu_j, \Sigma_j)$$

- les poids des gaussiennes et les paramètres des gaussiennes sont inconnus: leur estimation est typiquement un problème d'apprentissage non supervisé
 - approche itérative maximisant la vraisemblance
 - pb de l'initialisation: utilisation de k-moyennes, approche par division
 - simplifications: Σ_i diagonale, Σ_i identique pour toutes les gaussiennes
- avec un nombre de gaussiennes "suffisant", on peut s'approcher de toute densité de probabilité
 - limité par la quantité des données d'apprentissage!
 - pas de règle simple pour (bien) choisir le nombre de gaussiennes...