Reconnaissance de Formes

Classification Statistique

Laurent HEUTTE

Laurent.Heutte@univ-rouen.fr
http://www.univ-rouen.fr/psi/heutte

Plan du Cours

1- Introduction

définitions, notion de système, applications

2- Processus de RdF

pré-traitements, extraction de caractéristiques, représentation, classification, combinaison

3- Classification « statistique » des formes

- théorie bayésienne de la décision
- méthodes paramétriques ou non / bayésiennes ou non
- arbres de décision
- réseaux de neurones

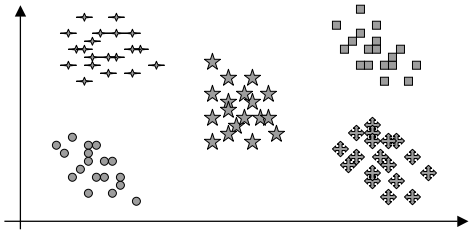
Approche Vectorielle

Une forme est représentée par un vecteur de R^d (ensemble des *d* mesures effectuées sur la forme)

$\Rightarrow$ Une forme est un point dans l'espace de représentation R^d

Contraintes: regroupement des classes dans des régions compactes de R^d

Exemples:



Outils mathématiques utilisés pour l'approche vectorielle:

proba, stat, analyse de données, géométrie, topologie

➔ outils développés et « bien maîtrisés »

Apprentissage / Décision Statistique

Apprentissage:

Etablissement par le système des frontières de séparation entre classes

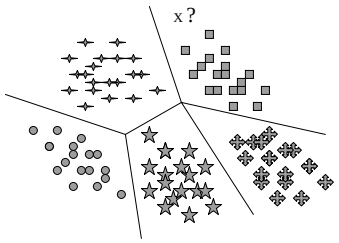
classes $\Rightarrow$ régionnement dans R^N

Décision:

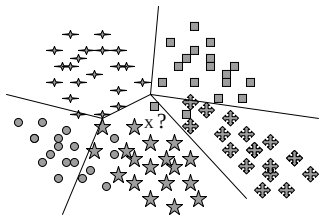
Affectation d'une classe d'appartenance en fonction de la position par rapport aux frontières

régionnement dans R^N $\Rightarrow$ classe

Cas facile



Cas difficile



Théorie Bayésienne de la Décision (1)

Notations:

$w \in \Omega$: ensemble des classes à reconnaître

$x \in R^N$: espace de représentation

$p(x)$: probabilité d'observer x

$p(w/x)$: probabilité d'appartenance à la classe w conditionnellement au fait d'être en x

$d : R^N \rightarrow \Omega$: règle de décision

$x \rightarrow d(x)=w_k$

(avec $\int_{R^N} p(x)dx = 1$ et $\forall x, \sum_k p(w_k/x) = 1$)

Règle de décision bayésienne (d1):

fonction qui à chaque objet x associe la classe la plus probable au point x:

$d1(x)=w \Rightarrow \forall w' \in \Omega, p(w/x) \geq p(w'/x)$

Rmq: si les caractéristiques sont suffisamment descriptives alors:

$p(w/x)=1$ ou $p(w/x)=0$

Théorie Bayésienne de la Décision (2)

Propriété: La fonction de décision bayésienne minimise la probabilité globale d'erreur du système

Démo: La probabilité d'erreur d'une règle est :

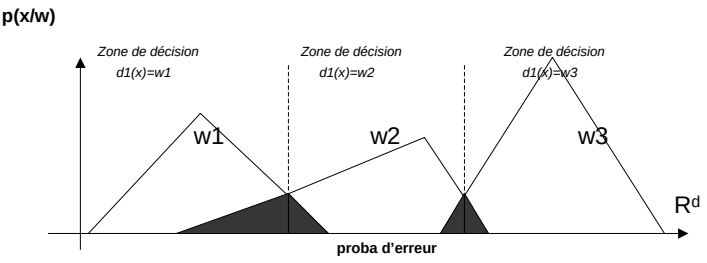
$$Err(d) = \int_{R^N} [1-p(d(x)/x)]p(x)dx$$

et $\forall x, \forall w, p(d1(x)/x) \geq p(d(x)/x)$

ainsi : $Err(d) - Err(d1) = \int_{R^N} [p(d1(x)/x)-p(d(x)/x)]p(x)dx$

d'où : $Err(d) \geq Err(d1), \forall d$

Règle: A chaque point x, associer la classe dont la densité de probabilité en x est la plus forte (maximiser la proba de ce qu'on observe)



Théorie Bayésienne de la Décision (3)

Théorème de Bayes:

Utilité: Expérience (apprentissage) $\Rightarrow$ p(x/w) ou p(x,w)
Décision $\Leftarrow$ p(w/x)

Enoncé: p(w/x) = [p(x/w)p(w)] / p(x)
car: p(x,w)=p(x/w)p(w)=p(w/x)p(x)

Exemple: n=100 observations
5 occurrences possibles pour x: x1, x2, …, x5
4 classes possibles: w1, w2, w3, w4

	x1	x2	x3	x4	x5
w1	2	20	0	1	0
w2	0	2	0	12	13
w3	6	4	8	0	2
w4	20	0	8	0	2

p(w1)=23/100; p(x3)=16/100; p(x5,w4)=2/100

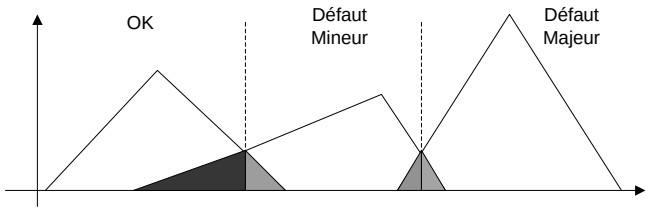
p(w1/x1)=2/28; p(w2/x1)=0;
p(w3/x1)=6/28; p(w4/x1)=20/28;
$\Rightarrow$ si j'observe x1, je décide w4

Théorie de la Décision avec Coût (1)

Principe:

la règle de décision bayésienne minimise la probabilité d'erreur globale mais dans la pratique, certaines erreurs peuvent porter plus à conséquence que d'autres!

$\Rightarrow$ non détection $\neq$ fausse alarme



Théorie de la Décision avec Coût (2)

Evaluation du risque de la décision prise

➔ Introduction d'une matrice de coûts (non détection ≠ fausse alarme)

Notations: c(wi,wj): coût de décider wi lorsqu'il s'agit en réalité de wj

d(x): règle de décision évaluée en x

$R_d(x)$: espérance de coût en x lorsque la décision est d(x)

$R_d(x) = \sum_j c(d(x),wj)p(wj/x)$

Exemple:

	x1	x2	x3	x4
w1	5	0	2	0
w2	3	4	2	1
w3	2	1	3	5

p(w1/x1)=5/10
p(w2/x1)=3/10
p(w3/x1)=2/10
décider w1

j \ i	w1	w2	w3	rejet
w1	0	10	60	10
w2	50	0	40	20
w3	100	30	0	10
Rd(x=x1)	35	11	42	13

décider w2

Méthodes Paramétriques / Non Paramétriques

Problème du modèle de base (TBD): difficile de mesurer les p(x/wi) !!!

➔ 2 types d'approches suivant les hypothèses retenues:

- méthodes paramétriques:

  on fait appel à une famille paramétrable de densités ou de surfaces et on optimise le(s) paramètre(s) pour minimiser la probabilité d'erreur

- méthodes non paramétriques:

  on développe un algorithme qui converge vers la densité ou la surface idéale quelle qu'elle soit (moyennant certaines hypothèses)

	Méthodes bayésiennes	Méthodes non bayésiennes
Méthodes paramétriques	Estimation de gaussiennes	Séparation linéaire
Méthodes non paramétriques	Fenêtres de Parzen	K plus proches voisins

Estimation de Gaussiennes (1)

Méthode paramétrique bayésienne

$p(x/w_k) = (2\pi)^{-d/2}(\det\Sigma_k)^{-1/2}\exp[-0.5(x-m_k)^t\ \Sigma_k^{-1}(x-m_k)]$

où: d dimension de l'espace de représentation
Σ_k matrice de variance-covariance de la classe w_k
m_k centre (moyenne) de la classe w_k

Apprentissage: estimer pour chaque classe w_k : m_k et Σ_k
Estimateur du Maximum de Vraisemblance

$m_k = (1/n_k)\Sigma x_i$

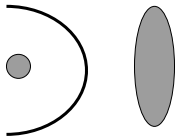
$\Sigma_k = (1/n_k\text{-}1)\Sigma(x_i-m_k)(x_i-m_k)^t$ (non biaisé)

Décision: appliquer la règle de Bayes
Choisir w_k qui maximise $p(w_k/x)=p(x/w_k)p(w_k)/p(x)$

Estimation de Gaussiennes (2)

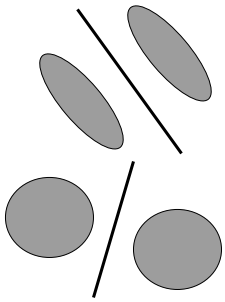
Choisir w_k qui maximise $p(w_k/x)$ revient à <u>minimiser</u> la distance du point au centre m_k (il suffit de changer de fonction discriminante!)

• cas général:
distance de Mahalanobis généralisée
$d^M_k(x) = (x-m_k)^t\ \Sigma_k^{-1}(x-m_k)$
$+ [\ \ln(\det\Sigma_k) - 2\ln p(w_k)\ ]$



• même covariance Σ:
distance de Mahalanobis
$d^m_k(x) = (x-m_k)^t\ \Sigma^{-1}(x-m_k)$

• gaussiennes isotropes:
distance euclidienne
$d^e_k(x) = (x-m_k)^t\ (x-m_k)$



Séparation Linéaire (1)

Méthode paramétrique non bayésienne

But : rechercher la meilleure surface séparatrice entre classes parmi une famille paramétrable de surfaces (ici: hyperplans)

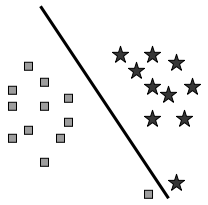
Hypothèse : séparabilité entre les classes

Apprentissage: Déterminer un hyperplan défini par un vecteur normal a et un scalaire a_0 tel que:

$\forall x \in w_1, \Sigma_i a_i x_i + a_0 > 0$

$\forall x \in w_2, \Sigma_i a_i x_i + a_0 < 0$

$(a_0, \ldots a_n)$ vecteur de paramètres à optimiser

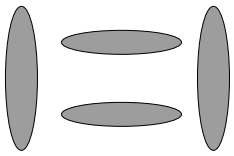


Décision : positionnement du point inconnu par rapport aux surfaces

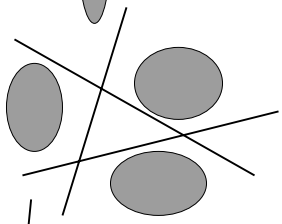
Séparation Linéaire (2)

Théoriquement, s'il y a $K=2^p$ classes, il faut p hyperplans pour les discriminer!

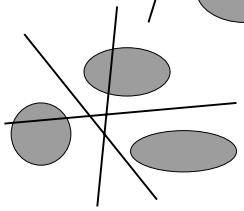
En pratique: difficile!!!



K classes $\Rightarrow$ K hyperplans:
(une classe contre toutes les autres)



K classes $\Rightarrow$ K(K-1)/2 hyperplans:
(classes deux à deux)



K Plus Proches Voisins (1)

Méthode non bayésienne non paramétrique

Principe : faire voter les K plus proches voisins de x pour savoir à quelle classe x appartient (K fixé)

Exemple :

Propriété remarquable : $Err^* \leq \lim_{N \to \infty} Err_{1ppv} \leq 2\, Err^*$

Avantages / inconvénients :

+ possibilité de faire du rejet ($K/2 \leq m \leq K$)
+ mise en œuvre très simple
+ convergence indépendante de la distance
+ possibilité de tenir compte des coûts
- complexité de l'algorithme de décision proportionnel à N
- performances mal connues quand N est fini

K Plus Proches Voisins (2)

Variantes accélérées sub-optimales:

- Kppv avec élimination (édition)
  prétraitement de la base d'apprentissage: élimination de tous les éléments dont le ppv appartient à une autre classe
- Condensation
  réduction de l'espace de représentation à un sous-espace ayant les mêmes propriétés (par ACP par exemple)
- Pavage
  l'espace est divisé en cellules et la recherche est limitée à quelques cellules: décision en 2 passes
- Tri sur ou plusieurs coordonnées
  limite la recherche à quelques dimensions
- Kppv hiérarchiques
  organisation des points sur plusieurs niveaux de finesse

Fenêtres de Parzen

Méthode bayésienne non paramétrique

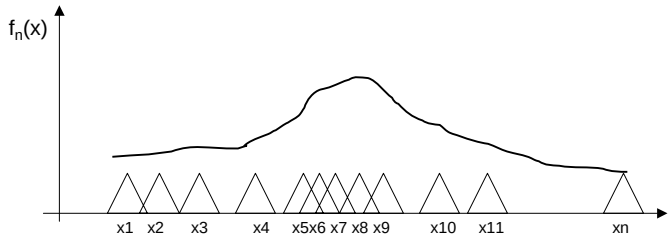
But : approximer p(x,w) par une somme de fonctions φ centrées autour des points d'apprentissage puis appliquer la règle de Bayes

Exemples de fonctions φ : gaussienne, uniforme, triangle, …

Principe: $x_1, x_2, \ldots, x_n \Rightarrow f_n(x) = 1/n \sum_i \varphi(x-x_i) = \hat{p}(x,w)$

n représentations d'éléments de w (apprentissage)

x : représentation d'un élément inconnu (décision)



➔ Gros volume de calcul lors de la reconnaissance

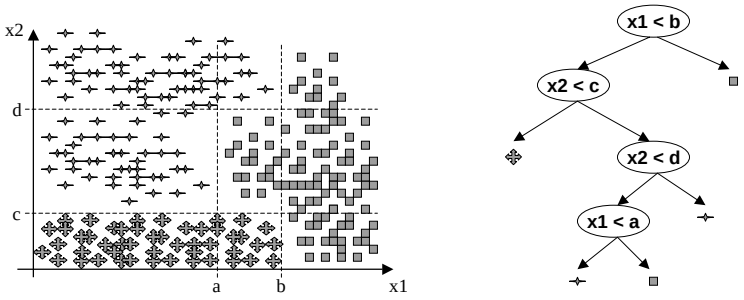
Laquelle Choisir ???

Méthode	Champs d'application	Convergence	Vol. de calcul (décision)	Vol. à stocker (apprentissage)
Parzen	Toute distribution	Lente, sûre	Énorme N*I	Énorme N*I
K-ppv	Toute distribution	Lente, sûre	Énorme N*I	Énorme N*I
Paramétrique gaussien	Très limité	Rapide (si…)	Faible $J*I^2$	Faible $J*I^2$
Séparation linéaire	Limité	Rapide (si…)	Très faible $1.5*J*I$	Faible $I*J(J-1)/2$

I: nombre de caractéristiques; J: nombre de classes;

N: nombre d'éléments en apprentissage

Arbres de Décision

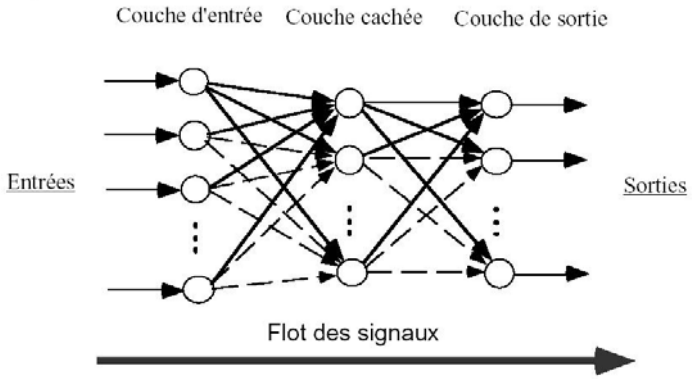


+ construction automatique / ordonnancement des tests
+ possibilité de pondérer les arcs par des probas
+ combinaison de caractéristiques
+ interprétation simple
- structure trop proche des données (sensibilité)
- difficultés à « balancer » l'arbre

Réseaux de Neurones (1)

A passe avant (e.g. Perceptron multi-couches)

- **Fonctionnement en reconnaissance**
- **Apprentissage ?**



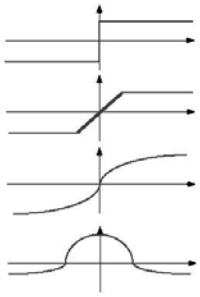
Réseaux de Neurones (2)

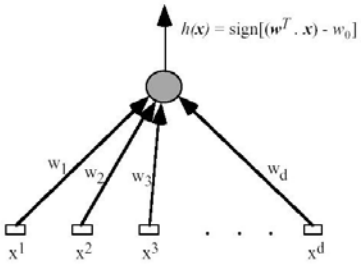
Neurone de base:

Calcule : $h(\boldsymbol{x}) = g(\sum_{1 \le i \le d} w_i x^i - w_0) = g(a)$

g : fonction d'activation

(Généralement non linéaire)





Réseaux de Neurones (3)

Apprentissage:

1. **Présentation d'un exemple parmi l'ensemble d'apprentissage**

   Séquentielle, aléatoire, en fonction d'un critère donné

2. **Calcul de l'état du réseau**
3. **Calcul de l'erreur = fct(sortie - sortie désirée)** (e.g. = $(y^k - t^k)^2$)
4. **Calcul des gradients**

   Par l'algorithme de rétro-propagation de gradient

5. **Modification des poids synaptiques**
6. **Critère d'arrêt**

   Sur l'erreur. Nombre de présentation d'exemples, ...

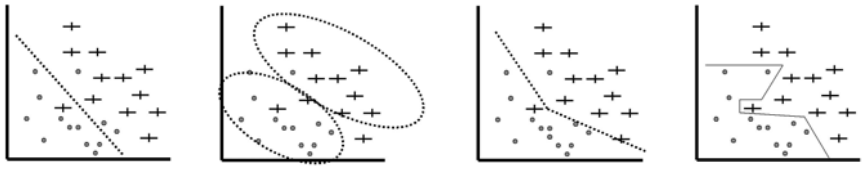
7. **Retour en 1**

Réseaux de Neurones (4)

Avantages:

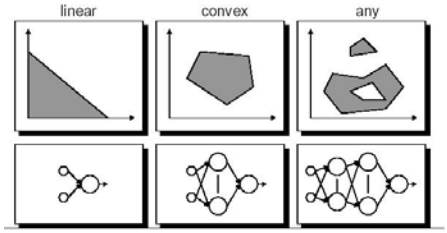
Aptitude naturelle à fournir des surfaces de décision complexes (par extraction de combinaison non linéaire des caractéristiques)

Capacités d'apprentissage fortement autonomes



Inconvénients:

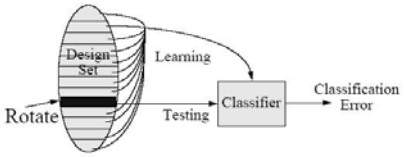
Effet « boite noire »



Conclusion

Evaluation des performances d'un classifieur: comment utiliser au mieux les données disponibles?

- A éviter: mesure de l'erreur apparente (sur la base d'apprentissage)
- Méthode Hold-Out:
  - Diviser l'ensemble des données disponibles en deux sous-ensembles (apprentissage et test)
  - Dilemme: avoir le plus de données possibles pour l'apprentissage / avoir le plus de données possibles pour le test (variance faible)
- Validation Croisée:



  - Biais (exactitude) / Variance (précision)
  - N grand: biais + petit; variance + grande; + coûteux en tps de calcul; à utiliser lorsque peu de données disponibles afin d'en utiliser au maximum pour l'apprentissage
  - N petit: biais + grand; variance + faible; - coûteux en tps de calcul; à utiliser lorsque l'ensemble des données disponibles est suffisamment important