Méthodes et Outils de Traitements Avancés

Clustering, Classification and Fuzzy Sets

Vincent Bombardier
(MdC HC 61ème Section)

Centre de Recherche en Automatique de Nancy -UMR CNRS 7039-
Département: Ingénierie des Systèmes Eco-Technique
Projet Systèmes Intelligents Ambiants

ISET

MOTA: Présentation du module

- Tatouage d'images (JMM-8h)
- Clustering et Classification (VB-8h)
  - Clustering
    - K-means
  - Classification
    - Arbres de Décision (ID3, C4.5, CART)
  - Extension vers Théorie des Ensembles Flous
    - Fuzzy C-means
    - FDT

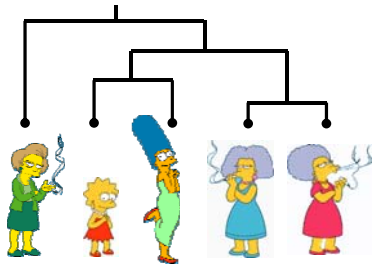
MOTA: What is Clustering?

- **Clustering** is the classification of objects into different groups, or more precisely, the partitioning of a data set into subsets (clusters), so that the data in each subset (ideally) share some common trait - often according to some defined distance measure.
- Also called unsupervised learning, sometimes called classification by statisticians and sorting by psychologists and segmentation by people in marketing
- Attach label to each observation or data points in a set
- Intuitively, if you would want to assign same label to a data points that are "close" to each other
- Sometimes, it is said that the for clustering, the distance metric is more important than the clustering algorithm

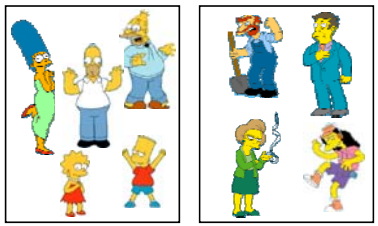
MOTA: Two Types of Clustering

- **Partitional algorithms:** Construct various partitions and then evaluate them by some criterion (k-means / Fuzzy C-means)
- **Hierarchical algorithms:** Create a hierarchical decomposition of the set of objects using some criterion (dendogramme)

Hierarchical



Partitional



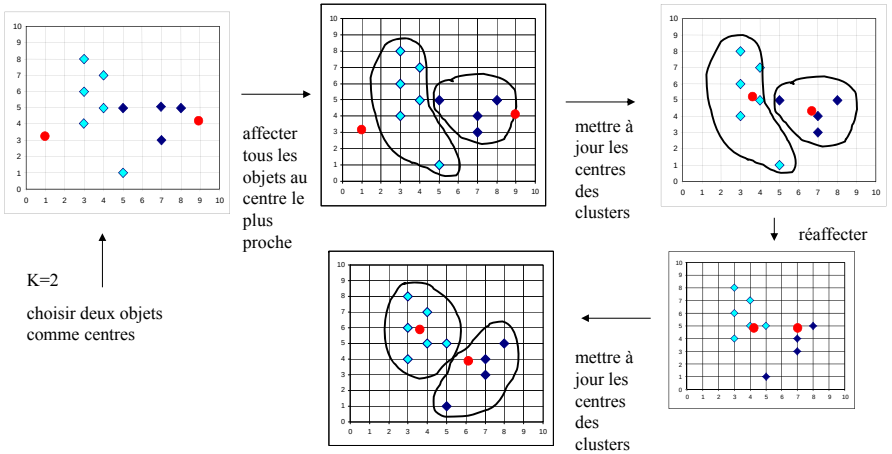
Desirable Properties of a Clustering Algorithm

- Scalability (in terms of both time and space)
- Ability to deal with different data types
- Minimal requirements for domain knowledge to determine input parameters
- Able to deal with noise and outliers
- Insensitive to order of input records
- Incorporation of user-specified constraints
- Interpretability and usability

Concepts de base

- But des algorithmes de cette section : Construire une partition d'une base de données de n objets en K clusters.
  - Étant donné K, trouver une partition de K clusters qui optimise une fonction objectif (un critère d'évaluation du clustering).
    - Pour trouver l'optimum global, il n'y a pas de méthode générale efficace
      - il faut souvent examiner toutes les partitions possibles
    - k-moyennes et k-médoïde sont des méthodes permettant de trouver des optimums locaux.
- k-moyennes
  - MacQueen 1967
  - Chaque cluster est représenté par son centre.
- k-medoïdes or PAM (Partition around medoids)
  - Kaufman & Rousseeuw, 1987
  - Chaque cluster est représenté par un objet du cluster.

Trace d'exécution de k-moyennes



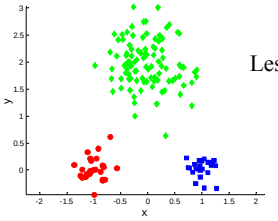
L'algorithme des K-moyennes

- Sélectionner K points pour être les centroïdes initiaux
  - Peut-être fait aléatoirement ou en partitonnnant la base de données en K partitions dont on choisit le centroïde
- Tant que les centroïdes changent d'une itération à l'autre
  - Former les K clusters en affectant chacun des points au centroïde le plus proche
  - Recalculer le centroïdes de chaque cluster
- Caractéristiques
  - L'idée de base de l'algorithme est très simple
  - Segmentation par partitionnement
  - Les clusters sont définis par leurs prototypes
  - Le nombre de clusters doit être spécifié
  - Il faut choisir
    - une fonction distance
    - une définition de centroïde

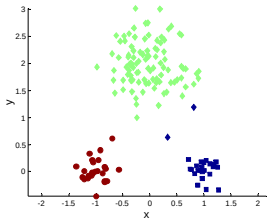
Commentaires généraux sur l'algorithme

- Les centroïdes initiaux peuvent varier d'un essai à l'autre
  - Les clusters produits peuvent différer d'un essai à l'autre
- Le centroïde est typiquement le centre des points
- La proximité est typiquement mesurée par la distance euclidienne, la similarité du cosinus, …
- L'algorithme converge en général pour les mesures typiques de similarité
- Souvent l'algorithme converge en quelques itérations
  - la condition d'arrêt peut être "seul l'appartenance d'un petit nombre de points est modifié d'une itération à l'autre"
- Complexité est O(n * K * I * d)
  - n = # de points, K = # de clusters, I = # d'itérations, d = # d'attributs

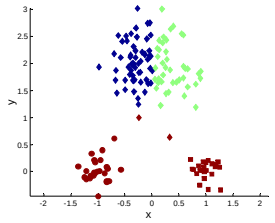
2 partitions différentes produites par 2 exécutions de l'algorithme



Les données



Partitionnement optimal



Partitionnement non- optimal

Description des symboles

- x : Un objet
- C_i : le $i^{ème}$ cluster
- c_i : Le centroïde du cluster C_i.
- c : Le centroïde de l'ensemble des données.
- m_i : # d'objets dans le $i^{ème}$ cluster.
- m : # d'objets dans la base de données
- K : # clusters

Remarque : le centroïde n'est pas nécessairement celui de la physique, i.e. la moyenne

Définition de la fonction objective et du centroïde

- La fonction « objectif » d'un algorithme de fouille de donnée est une fonction que l'on cherche à maximiser ou minimiser afin de se rapprocher du résultat voulu.
- Les algorithmes sont conçus pour itérer tant que l'évaluation de la fonction objectif n'a pas atteint un seuil déterminé.
- Pour l'algorithme K-moyennes, cette évaluation est cachée sous l'étape « Tant que les centroïdes changent d'une itération à l'autre »
  - Si les centroïdes changent, c'est que toutes les données n'avaient pas été associées au centroïde le plus proche et que les centroïdes idéaux selon cet algorithme n'avaient pas été trouvés.
- Un centroïde idéal celui qui minimise sa distance aux points qui lui sont associés
  - La notion de distance peut varier avec la nature du problème

Les mesures de distances

$$dist\ \text{Minkowski} = (\sum_{k=1}^{n} | p_k - q_k |^r)^{\frac{1}{r}},\ \text{c-moyennes floues}$$

$$dist\ euclidienne = (\sum_{k=1}^{n} | p_k - q_k |^2)^{\frac{1}{2}},\ \text{k-moyennes}$$

$$dist\ \text{Manhattan} = \sum_{k=1}^{n} | p_k - q_k |,\ \text{fuzzy ART}$$

$$\text{sup\_distance} = \max_{k \in [1...n]} | p_k - q_k |,$$

$$dist\ Mahalanobis = \sqrt{(p-q)\Sigma^{-1}(p-q)^T},\ \text{Ellipsoidal ART}$$

$$\text{Pearson correlation} = \frac{1 - r_{pq}}{2},\ \text{beaucoup utilisé en génétique}$$

...

Fonction objectif et centroïde pour un espace euclidien

- La distance utilisée est la distance euclidienne.
- La fonction objective à minimiser est SSE (Sum of Squared Error).

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(c_i, x)$$

- Le c_i qui minimise le SSE est la moyenne des $x_i \in C_i$

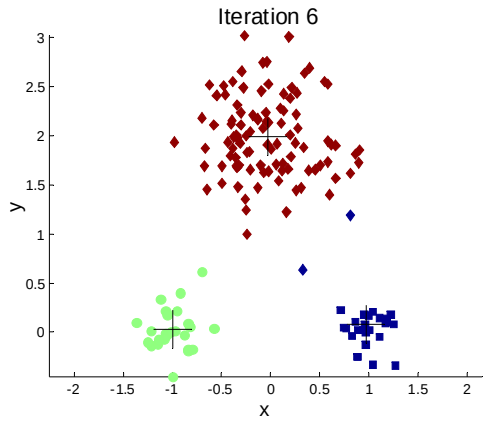
$$c_i = \frac{1}{m_i} \sum_{x \in C_i} x$$

- SSE décroit avec l'accroissement du nombre de clusters (K) et la compacité des clusters
- L'algorithme s'arrête sur un minimum local de SSE qui est fonction du choix des centroïdes initiaux

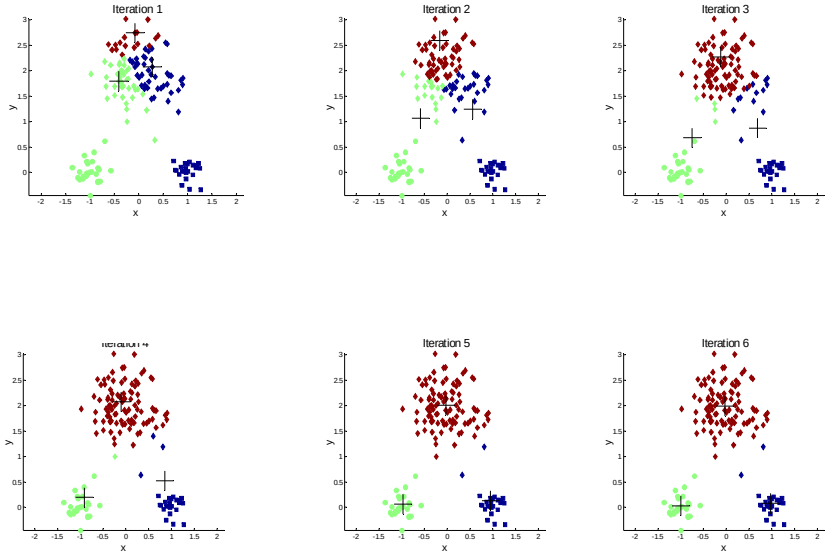
L'impact du choix des centroïdes inititiaux

- S'il y a autant de centroïdes qu'il y a de vrais clusters et qu'ils se trouvent chacun dans un cluster alors l'algorithme converge vers les vrais clusters
  - Mais on ne connait pas au départ où se trouvent les clusters et on ignore parfois le nombre de clusters.
- Les prochaines diapositives illustrent ce qui se passe lorsque les centroïdes ne se trouvent pas dans les clusters

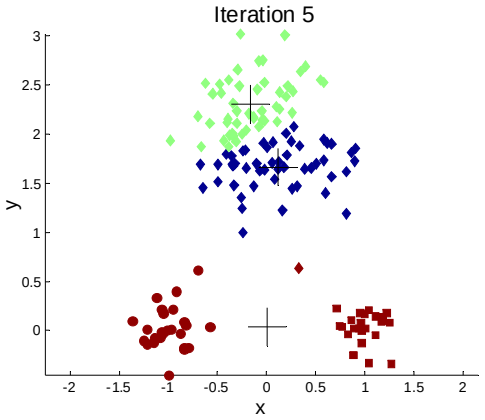
L'impact du choix des centroïdes inititiaux



MOTA: L'impact du choix des centroïdes inititiaux

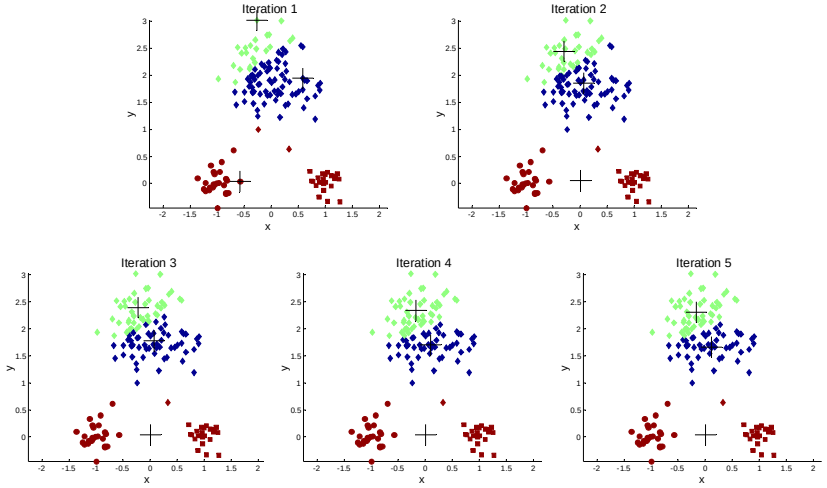


MOTA: L'impact du choix des centroïdes inititiaux



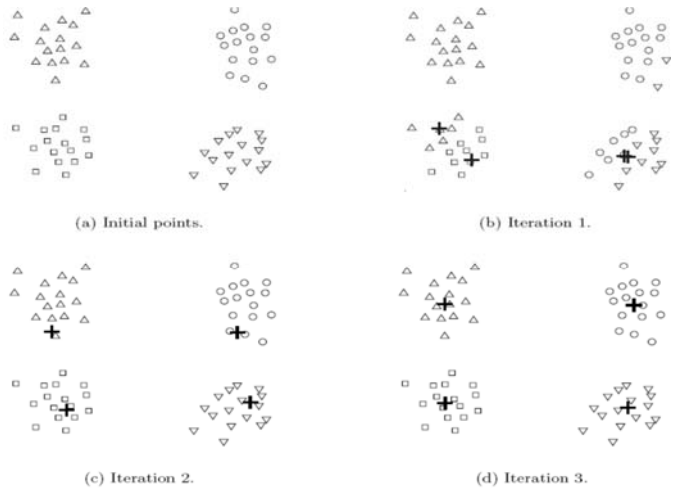
Comment est-on arrivé à un tel résultat ? (prochaine diapositive)

MOTA: L'impact du choix des centroïdes inititiaux



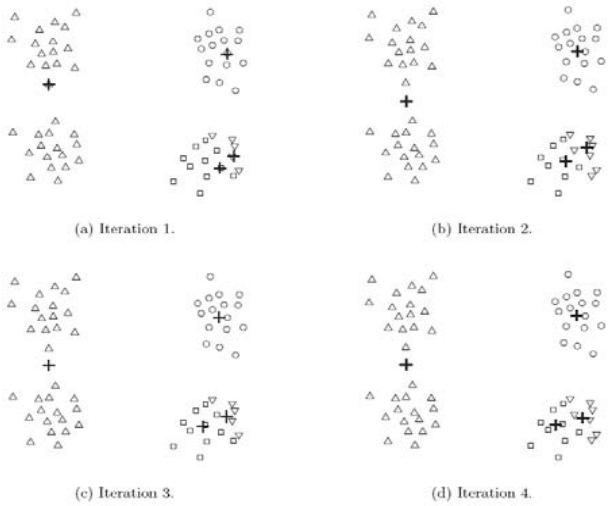
La capture de deux clusters par un centroïde est possible si 1) deux clusters sont plus proches entre eux que entre chacun de ceux-ci et les autres clusters ; 2) il y a un seul centroïde proche de ces 2 clusters.

MOTA: L'impact du choix des centroïdes inititiaux



Chaque paire de clusters rapprochés ont deux centroïdes

L'impact du choix des centroïdes inititiaux



Les paires de clusters rapprochés n'ont pas deux centroïdes

La sélection des centroïdes initiaux

- La probabilité d'avoir un seul centroïde dans chaque cluster (K est le nombre de clusters réels) diminue rapidement avec la valeur de K et la taille des clusters

$$p = \frac{\#\text{ de façons de choisir un centroide pour chaque segment}}{\#\text{ de façons de choisir K centroides}}$$

si les K segments ont chacun n objets alors

$$p = \frac{n^K}{(K\,n)!/(k\,n-k)!}$$

et pour K =10 et n=20 alors $p = 1.26 * 10^{-10}$

Solution au problème des centroïdes initiaux

- Multiples exécutions
  - Les probabilités ne sont pas de notre côté
- À partir d'un échantillon, déterminer les centroïdes initiaux par une méthode de clustering hiérarchique
- Choisir un premier centroïde aléatoirement ou comme étant le centre des données et ensuite choisir sucessivement k-1 centroïdes comme étant les points les plus éloignés des centroïdes déjà choisis.
- Traiter le problème une fois le clustering terminé
  - voir diapositive un peu plus loin
- Choisir une variante moins susceptible au problème du choix des centroïdes initiaux
  - Mise à jour incrémentale des centroïdes
    - exemple : K-moyenne par bisection

Que faire des clusters vides

- L'algorithme de base peut donner naissance à des clusters vides
- Plusieurs stratégies pour choisir un autre centroïde
  - Choisir le point qui se trouve le plus loin de tous les autres centroïdes
    - au minimum, on enlève le point qui contribue le plus à la valeur du SSE
  - Choisir un point à l'intérieur du cluster ayant le plus grand SSE
    - Cela entraînera la division de ce cluster en deux
  - S'il y a plusieurs clusters vides alors il faut réappliquer l'une des 2 statégies précédentes.

MOTA: Réduction du SSE lors du post-traitement

- Comme le SSE total est la somme des SSE des clusters, les stratégies consistent à diminuer globalement le SSE des clusters.
- Pour conserver K constant, les stratégies alternent entre deux phases :
  1. division d'un cluster ou introduction d'un nouveau centre
  2. fusion de deux clusters ou dispersion d'un cluster
- division d'un cluster
  - Celui qui contribue le plus au SSE total ou celui dont l'écart type d'un de ses attributs est le plus grand
- introduction d'un nouveau centre
  - souvent le point le plus éloigné de tous les centres ou un point du cluster contribuant le plus au SSE total
- fusion de deux clusters
  - souvent les clusters les plus rapprochés, ou les deux dont la fusion accroît le moins le SSE total
- dispersion d'un cluster
  - il s'agit de réaffecter tous les points du cluster contribuant le plus au SSE total.

MOTA: Mise à jour incrémentale des centroïdes

- Dans l'algorithme k-moyennes de base, tous les centroïdes sont recalculés après que tous les points ont été affectés à des clusters.
  - Une alternative est de recalculer les centroïdes après chaque affectation (approche incrémentale)
- Les caractéristiques de la mise à jour incrémentale
  - Chaque affectation entraîne le calcul de 0 (affecter au même cluster) ou 2 centroïdes (affecter à un autre cluster)
  - Possible de calculer l'affectation qui optimise une fonction objective arbitraire (différente de SSE total) ;
  - Plus coûteux
    - Mais comme la convergence est souvent rapide, le nombre de points réaffectés baisse rapidement
  - Introduit une dépendance sur l'ordre des points choisis
  - N'introduit jamais de clusters vides

MOTA: Le prétraitement des données

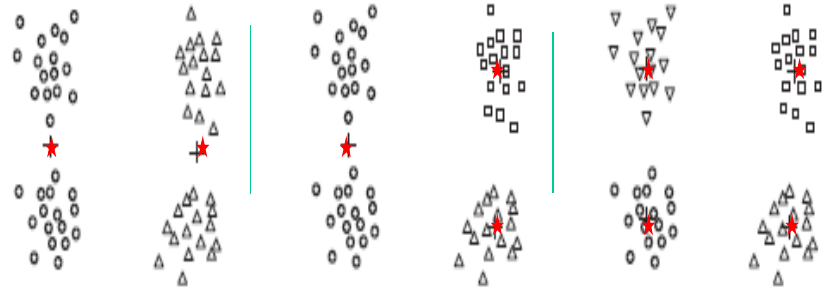
- La normalisation des données permet de rendre les clusters indépendants des unités de mesure des attributs
- L'élimination des données marginales permet d'obtenir des centres de cluster plus significatifs et mènent à une réduction du SSE total.
  - Trouver les données marginales n'est pas toujours évident

MOTA: Algorithme des k-moyennes par bisection

1. ListeDesClusters ← [<tousLesPoints>]
   - ; un seul cluster contenant tous les points
2. Tant que ListeDesClusters ne contient pas K clusters faire
   1. Choisir un cluster de ListeDesClusters et le supprimer de la liste
      - Plusieurs stratégies pour choisir le cluster à diviser
        - le plus gros
        - celui qui contribue le plus au SSE total
        - une combinaison des deux stratégies précédentes
   2. Appliquer n fois l'algorithme k-moyenne avec K=2 sur ce cluster
   3. Ajouter la paire de clusters qui minimise le SSE à ListeDesClusters
3. Appliquer K-moyenne en prenant comme point de départ ListeDesClusters
   - les centroïdes menant à un minimum local pour chaque paire ne sont pas ceux qui menant à un minimum local pour les K clusters

- Si on mémorise les bissections choisies de clusters, on obtient une segmentation hiérarchique

K-moyenne par bisection

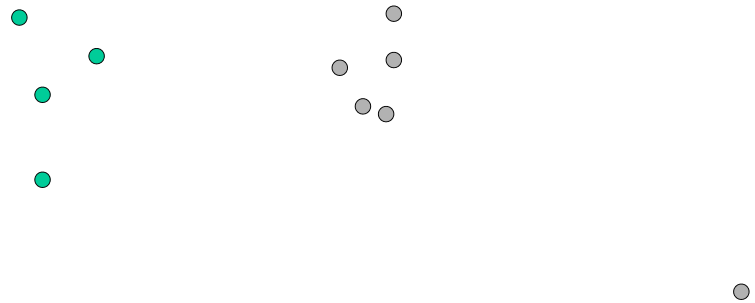


(a) Iteration 1. (b) Iteration 2. (c) Iteration 3.

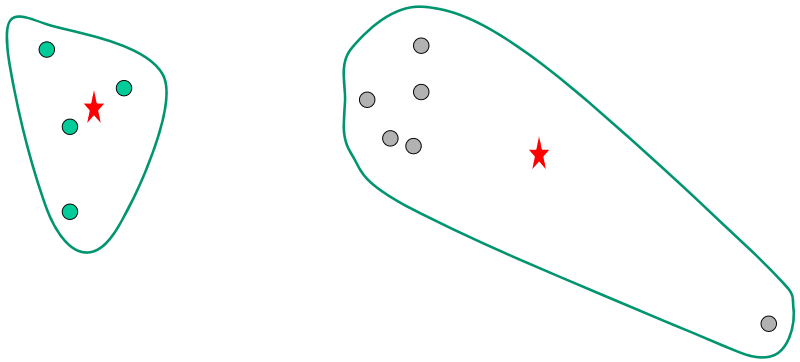
Les limites de l'algorithme des k-moyennes

- k-moyennes donne parfois une solution qui ne se rapproche pas des clusters naturels si ceux-ci sont
  - de différentes tailles
  - de différentes densités
  - n'ont pas une forme globulaire
- k-moyenne peut donner de mauvais résultats s'il y a des données marginales
- Le centre d'un cluster peut ne ressembler à aucun point réel.

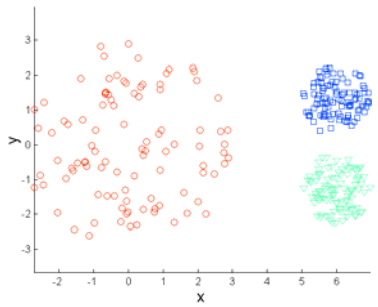
Effets des données marginales



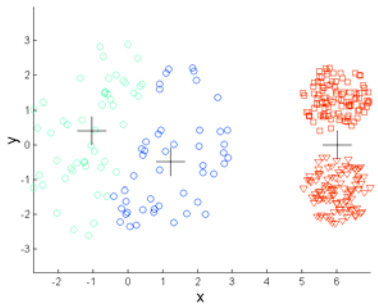
Effets des données marginales



MOTA: Effets de clusters ayant des densités différentes

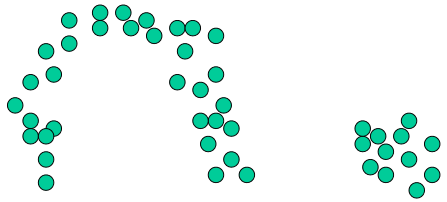


la base de données



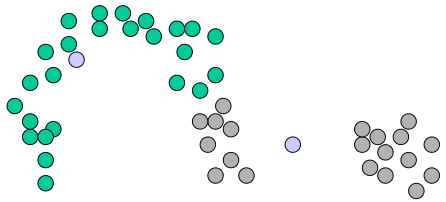
la classification avec k-moyennes (K=3)

MOTA: Effets des clusters non convexes



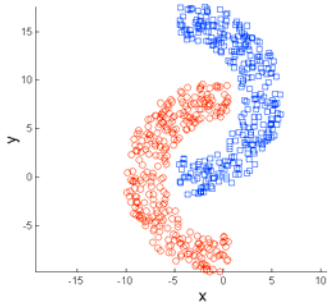
la base de données

MOTA: Effets des clusters non convexes

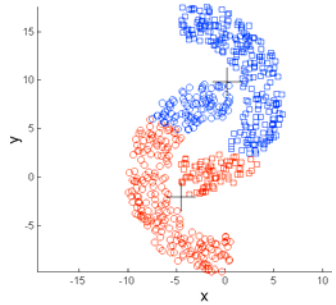


la segmentation avec k-moyennes (K=2)

MOTA: Effets des clusters non convexes

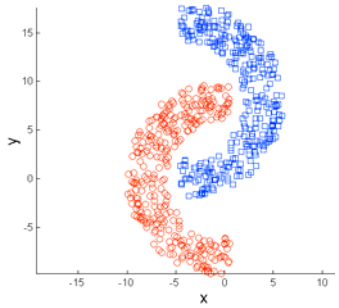


la base de données

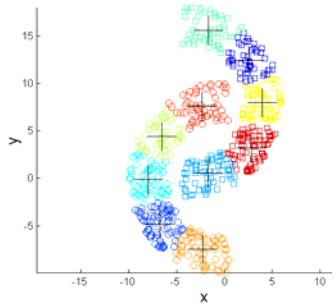


la segmentation avec k-moyennes (K=2)

Aller au-delà des limites de K-moyennes



La base de données



Les clusters produits par k moyenne

Il s'agit ici de choisir K > # clusters réels pour obtenir des sous-clusters des clusters réels et ensuite de combiner ces sous-clusters.

Comments on the K-Means Method

- Strength
  - *Relatively efficient*: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, k, $t << n$.
  - Often terminates at a *local optimum*. The *global optimum* may be found using techniques such as: *deterministic annealing* and *genetic algorithms*
- Weakness
  - Applicable only when *mean* is defined, then what about categorical data?
  - Need to specify *k*, the *number* of clusters, in advance
  - Unable to handle noisy data and *outliers*
  - Not suitable to discover clusters with *non-convex shapes*

CONCLUSION

- *K-means algorithm is* useful for undirected knowledge discovery and is relatively simple. K-means has found wide spread usage in lot of fields, ranging from unsupervised learning of neural network, Pattern recognitions, Classification analysis, Artificial intelligence, image processing, machine vision, and many others.

K-médoïdes
PAM (Partitioning Around Medoids)

- PAM (Kaufman and Rousseeuw, 1987), mis en oeuvre en S+
- Utilise les objets de la base de données pour représenter les clusters

Algorithme de PAM (Partitioning Around Medoids)

1. Choisir K médoïdes arbitrairement parmi les objets
2. Affecter chaque objet au médoïde le plus proche
3. Pour chaque paire (o_h, o_k) où
   o_h est un non-médoïde
   o_k est un médoïde
   TC_{kh} le coût total de l'échange (swap o_h o_k)
   si $TC_{kh} < 0$, alors remplacer o_k par o_h
4. Répéter l'étape 2 et 3 tant qu'il y a des échanges

TCkh : le coût total de l'échange de ok par oh

$TC_{kh}=\sum_i C_{ikh}$

on considère l'échange de o_k par o_h où

o_k est le médoïde actuel,

o_h est choisi aléatoirement parmi les non-médoïdes

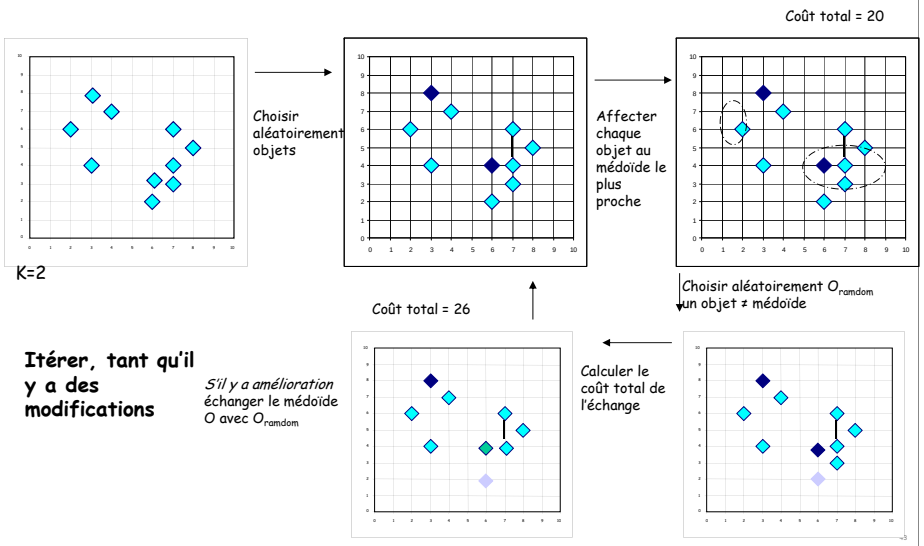
o_i est un non-médoïde quelconque $\neq o_h$

$o_{newmedi}$ le médoïde le plus proche de o_i après l'échange

$o_{oldmedi}$ le médoïde le plus proche de o_i avant l'échange

$C_{ikh} = dist(o_i, o_{newmedi}) - dist(o_i, o_{oldmedi})$

Illustration de l' algorithme PAM



Critique de l'algorithme PAM

- L'algorithme PAM donne de bons résultats sur de petites bases de données mais sa complexité ne lui permet pas de traiter de grandes bases de données

MOTA: CLARA (Clustering Large Applications)

- CLARA (Kaufmann and Rousseeuw in 1990)
  - mis en oeuvre en S+
- schéma de l'algorithme
  - applique PAM à plusieurs échantillons de la base de données
  - retourne les médoïdes de l'échantillon ayant la meilleure fonction objective
- s'applique à de grandes bases de données
- son efficacité dépend de la taille des échantillons
- si les médoïdes proviennent d'un échantillon biaisé alors ils ne sont pas similaires à ceux que l'on aurait trouvés avec toute la base de données

MOTA: CLARANS (A Clustering Algorithm based on Randomized Search)

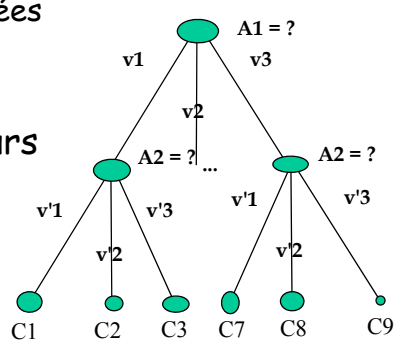
- CLARANS (Ng and Han, 1994)
  - "Randomized" CLARA
- CLARANS crée des échantillons de voisins dynamiquement
- Le processus de segmentation peut être présenter comme une fouille dans un graphe où chaque noeud est une solution potentielle
  - une solution = un ensemble de k médoïdes
- Si un optimum local est trouvé, CLARANS est réinitialisé avec un noeud choisi aléatoirement et continue sa fouille pour trouver un nouvel optimum local
- Il est plus efficace et évolutif (scalable) que PAM et CLARA
- Focusing techniques and spatial access structures may further improve its performance (Ester et al.'95)

MOTA: Arbres de décision: Définition

- Règles de classification basant leur décision sur des tests associes aux attributs organisés de manière arborescente.
- Les nœuds internes (nœuds de décision) sont étiquetés par des tests applicables à toute description d'un individu. En général, chaque test examine la valeur d'un unique attribut. Les réponses possibles correspondent aux arcs issus de ce nœud.
- Les feuilles sont étiquetées par une classe par défaut. Chaque nœud interne ou feuille est repéré par sa position : la liste des numéros des arcs qui permettent d'y accéder depuis la racine.
- Tout arbre de décision définit un classifier.
- Ce classifier a une traduction immédiate en terme de règles de décision.

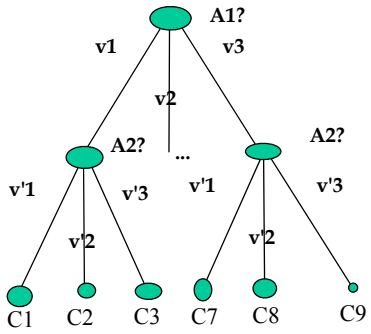
MOTA: Génération de l'arbre

- Objectif:
  - obtenir des classes homogènes
  - couvrir au mieux les données
- Comment choisir les attributs (Ai) ?
- Comment isoler les valeurs discriminantes (vj) ?



Arbre = ensemble de règles

- (A1=v1)&(A2=v'1) → C1
- (A1=v1)&(A2=v'2) → C2
- (A1=v1)&(A2=v'3) → C3
- ...
- (A1=v3)&(A2=v'1) → C7
- (A1=v3)&(A2=v'2) → C8
- (A1=v3)&(A2=v'3) → C9

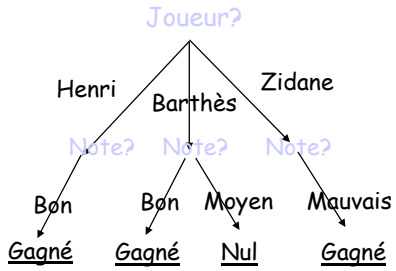


Exemple codant une table

Attributs ou variables

Joueur	Note	Résultat
Barthés	Bon	Gagné
Barthés	Moyen	Nul
Henri	Bon	Gagné
Henri	Bon	Gagné
Zidane	Mauvais	Gagné

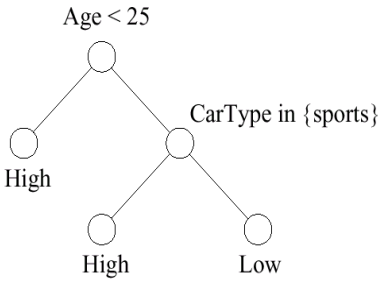
Classes cibles



Autre Exemple

rid	Age	Car Type	Risk
0	23	family	High
1	17	sports	High
2	43	sports	High
3	68	family	Low
4	32	truck	Low
5	20	family	High

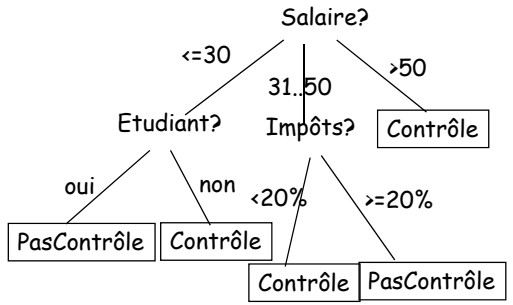
(a) Training Set



(b) Decision Tree

Autre Exemple

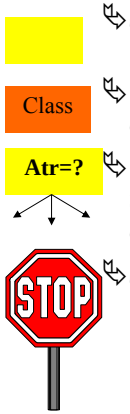
- Faut-il vous envoyer un contrôleur fiscal ?



Procédure de construction (1)

- Recherche à chaque niveau de l'attribut le plus discriminant
- Partition (nœud P)
  - si (tous les éléments de P sont dans la même classe) alors retour;
  - pour chaque attribut A faire
    - évaluer la qualité du partitionnement sur A;
  - utiliser le meilleur partitionnement pour diviser P en P1, P2, …Pn
  - pour i = 1 à n faire Partition(Pi);

Procédure de Construction (2)



- Processus récursif
  - L'arbre commence à un nœud représentant toutes les données
  - Si les objets sont de la même classe, alors le nœud devient une feuille étiqueté par le nom de la classe.
  - Sinon, sélectionner les attributs qui séparent le mieux les objets en classes homogènes => Fonction de qualité
  - La récursion s'arrête quand:
    - Les objets sont assignés à une classe homogène
    - Il n'y a plus d'attributs pour diviser,
    - Il n'y a pas d'objet avec la valeur d'attribut

Choix de l'attribut de division

- Différentes mesures introduites
  - il s'agit d'ordonner le désordre
  - des indicateurs basés sur la théorie de l'information
- Choix des meilleurs attributs et valeurs
  - les meilleurs tests
- Possibilité de retour arrière
  - élaguer les arbres résultants (classes inutiles)
  - revoir certains partitionnements (zoom, réduire)

Mesure de qualité

- La mesure est appelé fonction de qualité
  - Goodness Function en anglais
- Varie selon l'algorithme :
  - Entropie - Gain d'information (ID3/C4.5)
    - Suppose des attributs nominaux (discrets)
    - Peut-être étendu à des attributs continus
  - Gini Index
    - Suppose des attributs continus
    - Suppose plusieurs valeurs de division pour chaque attribut
    - Peut-être étendu pour des attributs nominaux

Indice de GINI

Si un ensemble de données T contient des éléments de N classes
$Gini(T) = \sum_i pi.(1-pi) = 1- \sum_i pi^2$ où pi est la fréquence relative de la classe i dans T

Indice de Twoing

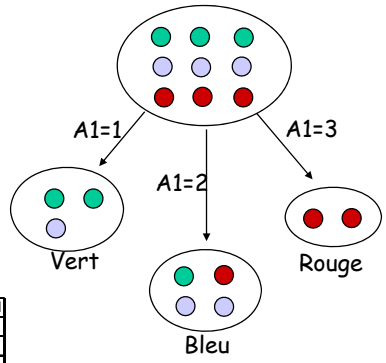
$G(t_g,t_d) = [((n_g/n)(n_d/n))/4][\sum_{i=1}^{m} | (n_{ig} / n_g) - (n_{id} / n_g) |]^2$

- t_g : Sommet gauche issu de t.
- t_d : Sommet droit issu de t
- n_d (resp (n_g)) = card {t_d } (resp card {t_g }).
- N : La taille de l 'échantillon d 'apprentissage.
- M : Le nombre de classe.
- n_{id} : (resp (n_{ig}): l 'effectif de la classe c_i dans t_d (resp (t_g)).

- Minimisation du désordre restant
  - pi = fréquence relative de la classe i dans le nœud N (% d 'éléments de la classe i dans N)
- Mesure d 'entropie d'un segment s
  - $E(N) = -\sum pi \, Log_2(pi)$
- Minimiser son évolution globale: Gain [Quinlan]
  - $\Delta N = E(N) - \sum_j Pj * E(Nj)$

- Mesure des mélanges de classes d'un nœud N
  - $i(N) = \sum i \sum j \{ pi * pj \}$ avec $i \neq j$
  - pi est la proportion d 'individus de la classe i dans N.
- La réduction d'impureté de chaque division du nœud N par la variable xj s'exprime par:
  - $\Delta N = i(N) - \sum_j pj * i(Nj)$
  - pj est la proportion d'individus du nœud dans le fils j
- Sur l 'ensemble des n variables, la division du nœud t est effectuée à l 'aide de la variable qui assure la réduction maximale de l'impureté ($\sum$ minimum)

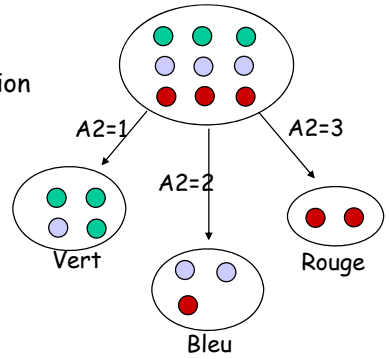
- Partition selon A1 (densité)
- Indice d'impureté :
  - $i(N) = \sum_i^k \sum_j^k \{ pi * pj \}$ avec $i \neq j$
  - Pi est la proportion d'individus de la classe i dans N.
- Entropie d'un segment s :
  - $E(N) = - \sum_i pi \log_2(pi)$



Proportion	C1	C2	C3	Total
Vert	0,67	0,25	0,00	
Bleu	0,33	0,50	0,00	
Rouge	0,00	0,25	1,00	
Entropie	0,92	1,00	0,00	0,75
N2 log2(N2)	-0,39	-0,50	0,00	
N3 log2(N3)	-0,53	-0,50	0,00	
N4 log2(N4)	0,00	0,00	0,00	
Impureté	0,44444444	0,625	0	0,43

Exemple: Partitions de boules (2)

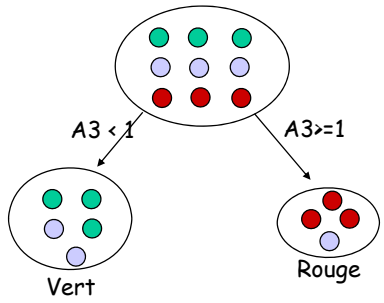
- Partition selon A2
  - Position et 4 au plus par partition



Proportion	C1	C2	C3	Total
Vert	0,75	0,00	0,00	
Bleu	0,25	0,67	0,00	
Rouge	0,00	0,33	1,00	
Entropie	0,81	0,39	0,00	0,49
N2 log2(N2)	-0,31	0,00	0,00	
N3 log2(N3)	-0,50	-0,39	0,00	
N4 log2(N4)	0,00	0,00	0,00	
Impureté	0,375	0,44444444	0	0,31

Exemple: Partitions de boules (3)

- Partition selon A3
  - Poids



Proportion	C1	C2	Total
Vert	0,60	0,00	
Bleu	0,40	0,25	
Rouge	0,00	0,75	
Entropie	0,97	0,50	0,76
N2 log2(N2)	-0,44	0,00	
N3 log2(N3)	-0,53	-0,50	
N4 log2(N4)	0,00	0,00	
Impureté	0,48	0,375	0,43

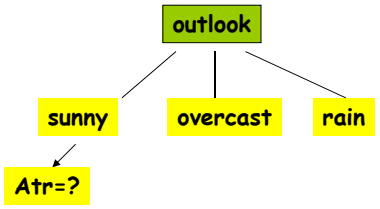
Exemple: Partitions de table (1)

Atr=?

Gain(Outlook) = 0.246
Gain(Temperature) = 0.029
Gain(Humidity) = 0.151
Gain(Windy) = 0.048

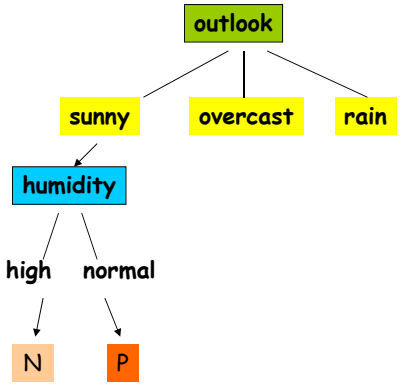
Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

Exemple: Partitions de table (2)



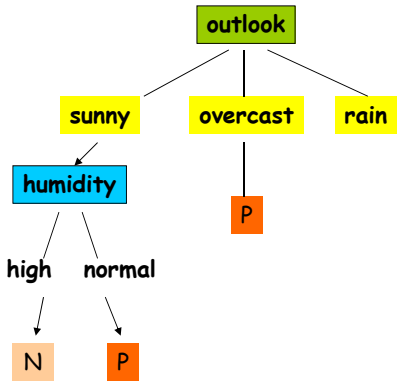
Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

Exemple: Partitions de table (3)



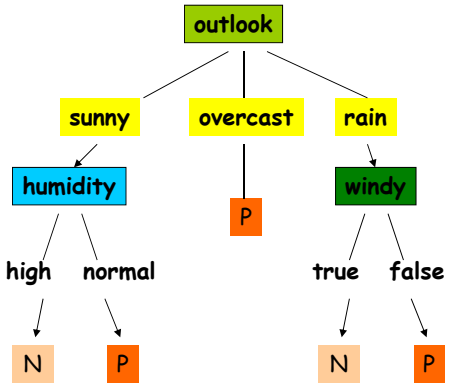
Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

Exemple: Partitions de table (4)



Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

Exemple: Partitions de table (5)



Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

Types de tests

- Binaire ou n-aire
  - plus ou moins large et profond
- Variable nominale
  - un prédicat par valeur ou par liste de valeurs ?
- Choix par niveau ou par classe
  - mêmes tests pour chaque nœud interne d'un niveau
  - arbres balancés ou non
- Élimination de classes
  - vides ou presque, peu représentatives

Types d'arbres

	Description	Critère de coupe
Segmentation	*binaire*	*variance si variable continue, Indice de pureté si variable nominale*
ID3	*n-aire*	*Gain informationnel par entropie de Shannon*
C4.5	*n-aire dérivé de ID3 + élagage, règles simplifiées, données manquantes,…*	*Ratio du gain informationnel*
CART	*binaire régression + élagage*	*Indice de Gini si 2 valeurs en conclusion Indice de Twoing sinon*

Méthodes ID3 et C4.5

- ID3
  - Le pouvoir discriminatoire (ou gain informationnel) d 'une variable <= une variation d '« entropie de Shannon » lors de la partition de S
- C4.5 (ID3++)
  - Support des variables continues
  - Introduit un facteur «Gain ratio » visant à pénaliser la prolifération des nœuds
- Critères d'arrêt :
  - Seuils de gain informationnel, d'effectif dans un nœud
  - Test statistique d'indépendance des variables (Ki2)

Méthode CART

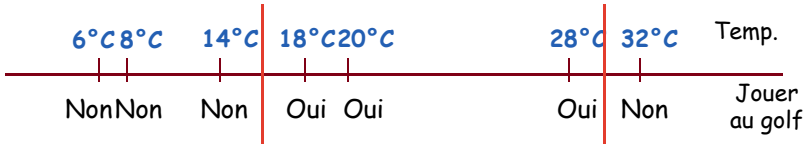
- Principes
  - si problème à 2 classes, cherche la bi-partition minimisant l'indice d'impureté de Gini
  - si problème à N classes, cherche celle maximisant le gain d'information donné par l'indice de Twoing
- Critères d 'arrêt :
  - Seuil de gain informationnel
  - Seuil d 'effectif dans un nœud
  - Procédure d'élagage

Les problèmes potentiels

1. Attributs à valeur continue
2. Attributs à facteurs de branchement différents
3. Valeurs manquantes
4. Sur-apprentissage
5. Recherche gloutonne
6. Le choix des attributs
7. Variance des résultats :
   - arbres différents à partir de données peu différentes

- Certains attributs sont continus
  - exemple : salaire
- découper en sous-ensembles ordonnés (e.g.,déciles)
  - division en segments [a0,a1[, [a1,a2[, …., [an-1,an]
- utiliser moyenne, médiane, … pour représenter
- minimiser la variance, une mesure de dispersion …
- investiguer différents cas et retenir le meilleur
  - exemple : 2, 4, 8, etc. par découpe d'intervalles en 2 successivement

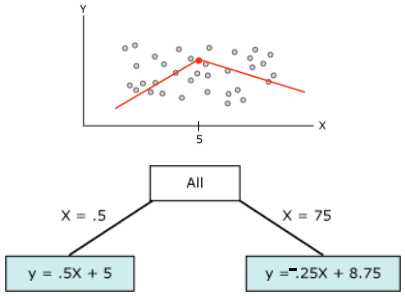
Discrétisation des attributs à valeur continue



Ici, deux seuils candidats : 16°C et 30°C

L'attribut $Temp_{>16°C}$ est le plus informatif, on le retient

- Partitionnement par droite de régression
- Chaque nœud est représenté par une formule de régression
- Séparation des données = point de non linéarité
- 1 ou plusieurs régresseurs
- Exemple :
  - salaire = a + b*tranche_age



- Soit un exemple $\langle x, c(x) \rangle$ dont on ne connaît pas la valeur pour l'attribut A
- Comment calculer $gain(S,A)$?

1. Prendre comme valeur **la valeur la plus fréquente dans S totale**
2. Prendre comme valeur **la valeur la plus fréquente à ce noeud**
3. **Partager l'exemple en exemples fictifs suivant les différentes valeurs possibles de A pondérés par leur fréquence respective**
   - **E.g. si 6 exemples à ce noeud prennent la valeur $A=a_1$ et 4 la valeur $A=a_2$**
     $A(x) = a_1$ avec prob=0.6 et $A(x) = a_2$ avec prob=0.4
   - **En prédiction, classer l'exemple par l'étiquette de la feuille la plus probable**

A-t-on appris un bon arbre de décision ?

- Ensemble d'apprentissage. Ensemble test.
- Courbe d'apprentissage
- Méthodes d'évaluation de la généralisation
  - Sur un ensemble test
  - Validation croisée
  - "Leave-one-out"

Sur-apprentissage : Effet du bruit sur l'induction

- Types de bruits
  - Erreurs de description
  - Erreurs de classification
  - "clashes"
  - valeurs manquantes
- Effet
  - Arbre trop développé : « touffus », trop profond

- Le sur-apprentissage (over-fitting)
- Risque empirique faible. Risque réel élevé.
- Le principe SRM (Minimisation du Risque Structurel)
  - Justification [Vapnik,71,79,82,95]
    - Notion de "capacité" de l'espace des hypothèses
    - Dimension de Vapnik-Chervonenkis

Il faut contrôler l'espace d'hypothèses

Il faut contrôler l'espace d'hypothèses

- Motivations :
  - Améliorer la performance en généralisation (SRM)
  - Fournir un modèle compréhensible des données (pour les experts)
- Stratégies :
  1. Contrôler directement la taille de l'arbre induit : *élagage*
  2. Modifier l'espace d'états (arbres) dans lequel se fait la recherche
  3. Modifier l'algorithme de recherche
  4. Restreindre la base de données
  5. Traduire les arbres obtenus dans une autre représentation

Procédure d'élagage

- Les arbres trop touffus sont inutiles et peu généraux
- Intérêt d'un élagage récursif à partir des feuilles
  - S'appuie sur un modèle de coût d'utilité
- Possibilité de l'appliquer sur l'ensemble des données ou sur un sous-ensemble réservé à la validation

Exemple d'élagage

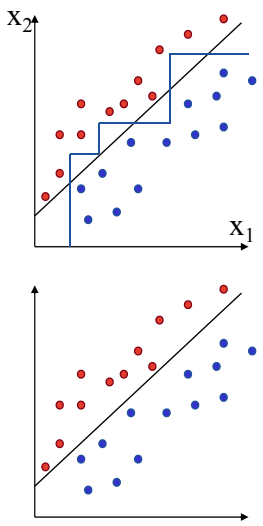
- Exemple :
  - arbres vus comme encodage de tuples
  - partition utile si gain supérieur à un seuil
  - coût d'un partitionnement
    - CP bits pour coder les prédicats de partition
    - Entropie_Après bits pour coder chaque tuple
  - partitionnement à supprimer si :
    - Gain = n * Entropie_Après + CP - n * Entropie_Avant < seuil
- Ce test peut être appliquer lors de la création

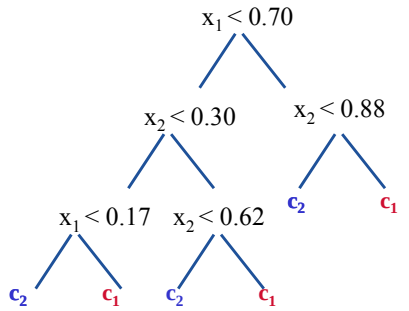
Induction d'arbres obliques

- Autre cause d'arbres touffus : une représentation inadaptée

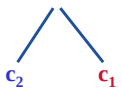
- **Solutions :**
  - Demander à l'**expert**
  - Faire préalablement une **ACP**
  - Autre méthode de **sélection d'attributs**
  - Appliquer de l'**induction constructive**
  - **Induction d'arbres obliques**

Arbres obliques





$1.1x_1 + x_2 < 0.2$



Conclusions

- Approprié pour :
  - Classification de formes décrites en attributs-valeurs
  - Attributs à valeurs discrètes
  - Résistant au bruit
- Stratégie :
  - Recherche par construction incrémentale d'hypothèse
  - Critère local (gradient) fondé sur critère statistique
- Engendre
  - Arbre de décision interprétable (e.g. règles de production)
- Nécessite contrôle de la taille de l'arbre

Évaluation d'un modèle de classification

- Métriques pour évaluer la performance du modèle
  - Comment mesurer la performance du modèle ?
- Méthodes pour obtenir des résultats fiables ?
- Méthodes de comparaison de modèle
  - Comment évaluer les performances relatives de plusieurs modèles proposés ?

Cluster Validation

- The number of clusters is not always previously known.
- In many problems the number of classes is known but it is not the best configuration.
- It is necessary to study methods to indicate and/or validate the number of classes:
  - User inspection
    - Study centroids, and spreads
    - Rules from a decision tree.
    - For text documents, one can read some documents in clusters.

Cluster evaluation: ground truth

- We use some labeled data (for classification)
- Assumption: Each class is a cluster.
- After clustering, a confusion matrix is constructed. From the matrix, we compute various measurements, entropy, purity, precision, recall and F-score.
  - Let the classes in the data D be $C = (c_1, c_2, \ldots, c_k)$. The clustering method produces k clusters, which divides D into k disjoint subsets, $D_1, D_2, \ldots, D_k$.

Evaluation measures:

true positive (TP)
eqv. with hit
true negative (TII)
eqv. with correct rejection
false positive (FP)
eqv. with false alarm, Type I error
false negative (FII)
eqv. with miss, Type II error

sensitivity or true positive rate (TPR)
eqv. with hit rate, recall

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN}$$

specificity (SPC) or true negative rate (TNR)

$$SPC = \frac{TN}{N} = \frac{TN}{FP + TN}$$

precision or positive predictive value (PPV)

$$PPV = \frac{TP}{TP + FP}$$

negative predictive value (NPV)

$$NPV = \frac{TN}{TN + FN}$$

fall-out or false positive rate (FPR)

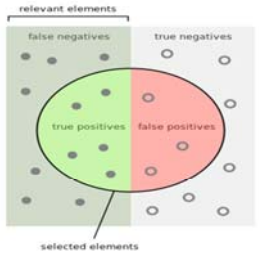
$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN} = 1 - SPC$$

false discovery rate (FDR)

$$FDR = \frac{FP}{FP + TP} = 1 - PPV$$

miss rate or false negative rate (FNR)

$$FNR = \frac{FN}{P} = \frac{FN}{FN + TP}$$



accuracy (ACC)

$$ACC = \frac{TP + TN}{P + N}$$

F1 score
is the harmonic mean of precision and sensitivity

$$F1 = \frac{2TP}{2TP + FP + FN}$$

Matthews correlation coefficient (MCC)

$$\frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

Informedness = Sensitivity + Specificity - 1
Markedness = Precision + NPV - 1

Sources: Fawcett (2006) and Powers (2011).[83]

Métriques pour évaluer la performance

- L'important est la capacité de prédiction du modèle
  - La vitesse de classification, d'induction du modèle et sa maintenance sont des éléments secondaires
- La matrice de confusion

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	TP	FN
	Class=No	FP	TN

TP (true positive)
FN (false negative)
FP (false positive)
TN (true negative)

Métriques pour évaluer la performance

		Classe prédite	
		Class=Yes	Class=No
Classe réelle	Class=Yes	TP	FN
	Class=No	FP	TN

$$Accuracy(Justesse) = \frac{TP + TN}{TP + TN + FP + FN}$$

Limites de la mesure de précision

- La justesse ne tient pas compte de la répartition des données dans les classes

		Classe prédite	
		Class=Yes	Class=No
Classe réelle	Class=Yes	9990	0
	Class=No	10	0

- Si le modèle classe toutes les données dans la classe yes,
  - sa justesse est (9990+0) / (9990 + 0 +10 + 0) = 99.9 %
  - mais il est incapable de reconnaître une donnée de la classe No
- Est-ce un classificateur utile ?

Autres mesures

#	Classe prédite	
	Class=Y	Class=N
Classe réelle — Class=Y	a	b
Classe réelle — Class=N	c	d

$$\text{Précision} = \frac{a}{a+c} = \frac{TP}{TP+FP}$$

$$\text{Rappel} = \frac{a}{a+b} = \frac{TP}{TP+FN}$$

$$F-\text{measure} = \frac{2 * \text{Rappel} * \text{Précision}}{\text{Rappel} + \text{Précision}} = \frac{2TP}{2TP+FP+FN}$$

$C\left({}^{prédit}/_{réel}\right)$

- Précision est sensible à C(Yes|Yes) & C(Yes|No)
- Rappel est sensible à C(Yes|Yes) & C(No|Yes)
- F-mesure (moyenne harmonique) n'est pas sensible à C(No|No)
  - plus proche de la valeur la plus faible (entre p et r)
  - donc une valeur forte signifie que la justesse et le rappel est bon

Evaluation measures: Entropy

Entropy: For each cluster, we can measure its entropy as follows:

$$entropy(D_i) = -\sum_{j=1}^{k} \Pr_i(c_j) \log_2 \Pr_i(c_j), \quad (29)$$

where $\Pr_i(c_j)$ is the proportion of class c_j data points in cluster i or D_i. The total entropy of the whole clustering (which considers all clusters) is

$$entropy_{total}(D) = \sum_{i=1}^{k} \frac{|D_i|}{|D|} \times entropy(D_i) \quad (30)$$

Evaluation measures: purity

Purity: This again measures the extent that a cluster contains only one class of data. The purity of each cluster is computed with

$$purity(D_i) = \max_j(\Pr_i(c_j)) \quad (31)$$

The total purity of the whole clustering (considering all clusters) is

$$purity_{total}(D) = \sum_{i=1}^{k} \frac{|D_i|}{|D|} \times purity(D_i) \quad (32)$$

An example

Example 14: Assume we have a text collection D of 900 documents from three topics (or three classes), Science, Sports, and Politics. Each class has 300 documents. Each document in D is labeled with one of the topics (classes). We use this collection to perform clustering to find three clusters. Note that class/topic labels are not used in clustering. After clustering, we want to measure the effectiveness of the clustering algorithm.

Cluster	Science	Sports	Politics		Entropy	Purity
1	250	20	10		0.589	0.893
2	20	180	80		1.198	0.643
3	30	100	210		1.257	0.617
Total	300	300	300		1.031	0.711

MOTA: A remark about ground truth evaluation

- Commonly used to compare different clustering algorithms.
- A real-life data set for clustering has no class labels.
  - Thus although an algorithm may perform very well on some labeled data sets, no guarantee that it will perform well on the actual application data at hand.
- The fact that it performs well on some label data sets does give us some confidence of the quality of the algorithm.
- This evaluation method is said to be based on **external data** or information.

MOTA: Evaluation based on internal information

- **Intra-cluster cohesion** (compactness):
  - Cohesion measures how near the data points in a cluster are to the cluster centroid.
  - Sum of squared error (SSE) is a commonly used measure.
- **Inter-cluster separation** (isolation):
  - Separation means that different cluster centroids should be far away from one another.
- In most applications, expert judgments are still the key.

MOTA: Compactness and Separation

- This a very complete validation measure.
- It validates the number of clusters and the checks the separation among clusters.
- From our experiments it works well even when the degree of superposition is high.

MOTA: Indirect evaluation

- In some applications, clustering is not the primary task, but used to help perform another task.
- We can use the performance on the primary task to compare clustering methods.
- For instance, in an application, the primary task is to provide recommendations on book purchasing to online shoppers.
  - If we can cluster books according to their features, we might be able to provide better recommendations.
  - We can evaluate different clustering algorithms based on how well they help with the recommendation task.
  - Here, we assume that the recommendation can be reliably evaluated.

MOTA: Méthodes pour obtenir des résultats fiables

- Comment obtenir une estimation fiable de la performance
- Pas de méthodes pré-établies, celle que vous créerez doit
  - prendre en ligne de compte
    - l'algorithme d'apprentissage,
    - la distribution des classes
      - On a vu que la précision est affectée par la distribution des données dans les classes
    - le coût d'une mauvaise classification
    - la taille de l'échantillon d'apprentissage et de test
  - utiliser une des métriques qui viennent d'être illustrées

MOTA TEF: Pourquoi ?

- Années 60 : généralisation des théories numériques
  - succès de la théorie du contrôle dans d'autres domaines tels que économie, biologie…
  - développement des techniques de simulation pour la prédiction

Les mêmes techniques vont pouvoir être utilisées avec un succès et une efficacité identiques sur les systèmes « humains »

MOTA TEF: Principe d'incompatibilité

- Zadeh (1973) :

…Exprimé de façon informelle, l'essence de ce principe est que lorsque la complexité d'un système augmente, notre habilité à émettre des jugements précis et néanmoins significatifs sur son comportement diminue jusqu'à atteindre un seuil au-delà duquel la précision et la pertinence deviennent des caractéristiques pratiquement mutuellement exclusives…

MOTA TEF: Conséquences

- Besoin d'un cadre formel permettant de :
  - se rapprocher de la manière dont l'Homme raisonne
  - modéliser les imperfections dont des connaissances peuvent être entachées :
    - incertitude : un doute existe quant à leur validité
    - imprécision : difficulté à les exprimer clairement
    - incomplétude: la connaissance est incomplète ou manquante

MOTA TEF: Sources d'imperfections

- Incertitude :
  - Gustave va probablement être reçu à son examen
  - Prendre le métro doit être une bonne solution
  - Si le malade tousse, il y a présomption de bronchite
- Imprécision :
  - Gus mesure entre 1m70 et 1m80
  - La température est d'environ 24°C
  - Si le malade tousse beaucoup, lui donner un peu de sirop
- Incomplétude (Vagueness)

MOTA TEF: Sources d'imperfections

- D'une manière générale un élément d'information est définit comme un quadruplet :

  Information=(attribut, objet, valeur, confiance)

  - Attribut renvoie à une fonction qui affecte une valeur à l'objet,
  - La valeur correspond à un sous ensemble du référentiel lié à l'attribut
  - Confiance est une indication sur la fiabilité

MOTA TEF: Sources d'imperfections

- On peut alors différencier clairement l'incertain de l'imprécis :
  - L'imprécision porte sur le contenu de l'information, i.e. « valeur »
  - L'incertain est relatif à la « confiance »
- On peut ajouter la gradualité et l'incomplétude:
  - La gradualité porte sur la définition de l'information, ie « attribut » ou « objet »
  - L'incomplétude concerne l'absence d'information.

MOTA TEF: Fuzziness : le flou (Zadeh, 1965)

- L'homme appréhende son environnement, raisonne et communique avec et sur des connaissances graduelles plutôt que tranchées
- La précision n'est pas toujours nécessaire et peut être « dangereuse » (Freine dans 24.5m pendant 2.34s…)
- Les sous-ensembles flous modélisent cette notion de gradualité :
  - on assigne à chaque élément d'un référentiel un degré d'appartenance.
  - L'ensemble des degrés d'appartenance des éléments du référentiel représente la fonction d'appartenance du sous-ensemble flou.

MOTA TEF: Historique

- 1965 : Théorie des ensembles flou introduite par L.A. Zadeh (UC Berkeley)
- En 1973, le Pr. Zadeh publie un article (dans l'IEEE Transactions on Systems, Man and Cybernetics) qui mentionne pour la première fois le terme de variables linguistiques (dont la valeur est un mot et non un nombre).
- En 1974, Mamdani (Université de Londres) réalise un contrôleur flou expérimental sur un moteur à vapeur.
- En 1980, F.L. Smidth & Co. A/S (au Danemark) met en application la théorie de la logique floue dans le contrôle de fours à ciment. C'est la première mise en oeuvre pratique de cette nouvelle théorie.
- Dans les années 80, plusieurs applications commencent à immerger (notamment au Japon).
- En 1987, « explosion du flou » au Japon (avec le contrôle du métro de Sendaï) et qui atteint son apogée en 1990 (fuzzymania).
- MAIS !!!! Ce que peut faire le flou, peut toujours être mieux fait avec les probabilités ????

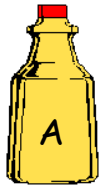
MOTA TEF: Historique

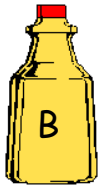
- 1990: Généralisation de l'utilisation de cette technique.
  - appareils électroménagers (lave-linge, aspirateurs, autocuiseurs,...etc) ,
  - systèmes audio-visuels (appareils de photos autofocus, caméscope à stabilisateur d'images, photocopieurs,...)
  - systèmes automobiles embarqués (BVA, ABS, suspension, climatisation,...etc.),
  - systèmes autonomes mobiles,
  - systèmes de décision, diagnostic, reconnaissance,
  - systèmes de contrôle/commande dans la plupart des domaines industriels de production.
- Il existe de processeurs dédiés et des interfaces de développement spécifiques (Cf 68HC12 de Motorola)
  - Ex: la famille des processeurs WARP (Weight Associative Rule Processor) de SGS-THOMSON dont les principales caractéristiques sont les suivantes :
    - Nombre de règles traitées : 256
    - Nombre d' entrées : 16
    - Nombre de sorties : 16
    - Méthode de composition des règles : Centre de gravité
    - Vitesse de traitement : 200 microsecondes pour 200 règles.

MOTA TEF: Probabilité / Flou (J. Bezdec)

ensembles flous = déguisement pour les statistiques ?

NON





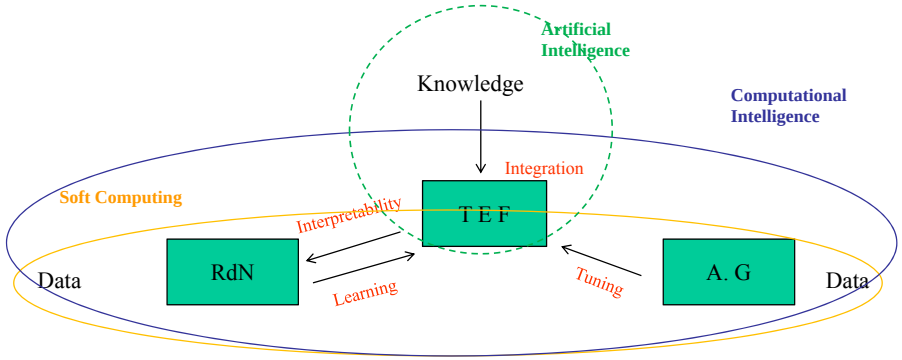
$\mu(A) = 0.9$ $p(B) = 0.9$

Quelle bouteille boirez-vous ?

MOTA TEF: Probabilité / Flou (J. Bezdec)

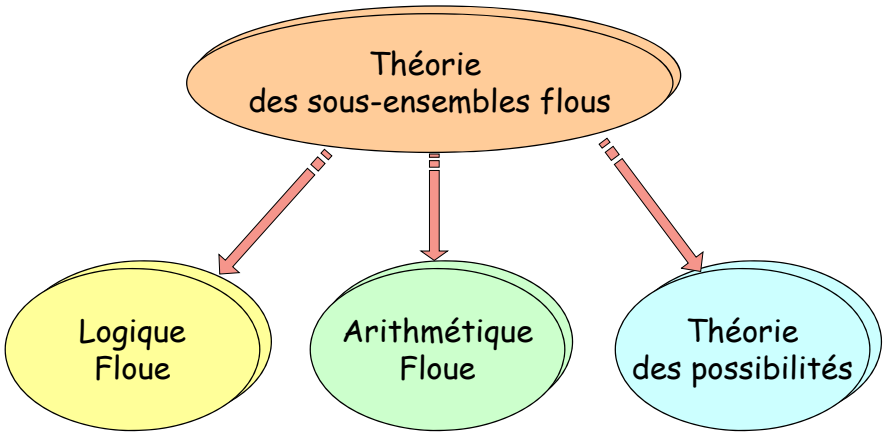
- A contient par exemple de l'eau vaseuse, pas de l'acide chlorydrique.
  - A est proche d'un liquide tout à fait potable.
- Sur 100 bouteilles B, 90 sont potables, 10 sont impropres voire fatales.
  - Il vaut mieux boire de l'eau vaseuse que de prendre le risque de mourir.
- IEEE Transaction on Fuzzy Systems Vol 2, N°1, 1994.
  - Fuzzziness vs. Probability (! ?)

- Soft Computing : From Data to Information
  - Informations non floues par nature, mais entachées d'imperfections (imprécisions ou/incertitudes) apportées par le processus de mesure et/ou son Environnement.

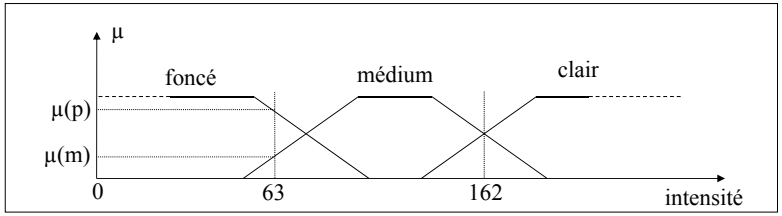


- Fuzzy Information Engineering: From Knowlegde to Information
  - l'information est floue par essence, soit parce que l'Objet est flou du fait de sa variabilité intrinsèque ou parce qu'on ne peut le définir de façon stricte (modèle linguistique), soit parce que les propriétés qui caractérisent l'Objet (net ou flou) sont floues.

⇒ Computing with Words



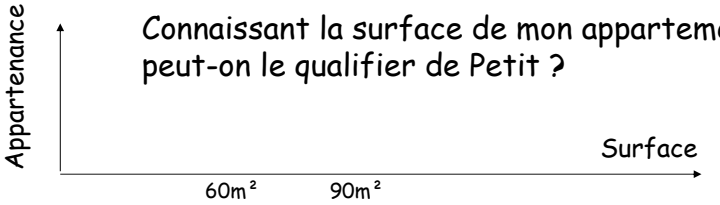
- Utilisation de variables linguistiques définies par un triplet (V, X, Tv)
  - *V est la variable (longueur, compacité, teinte, intensité, …)*
  - *X le référentiel (domaine de variation de V)*
  - *Tv le vocabulaire utilisé pour décrire de manière symbolique les valeurs de V (clair, médium, foncé, …)*
- Réalisée par des fonctions trapèzes, triangles, gaussiennes, équiréparties ou placées selon la distribution des variables d'entrée dans l'univers de discours.



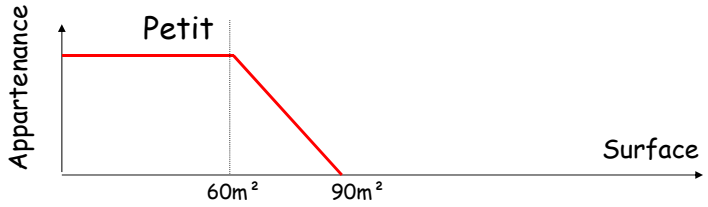
- Degré de similarité :
  - $\mu_A(x)$ exprime un degré de proximité, similarité…
    - $\mu_A(x)=1$ désigne un élément prototypique de A,
    - $\mu_A(x)=0$ désigne un élément dissemblable des prototypes,
    - $\mu_A(x)\in]0,1[$ indique la proximité de x du plus proche prototype.

- Utilisations :
  - Classification
  - Les mécanismes d'interpolation
- Applications :
  - Commande
  - Reconnaissance de Formes

- Degré de similarité :
  - *A* représente un concept, une caractérisation
  - $\mu_A(x)$ exprime le degré de proximité, similarité, ressemblance… de x à ce concept



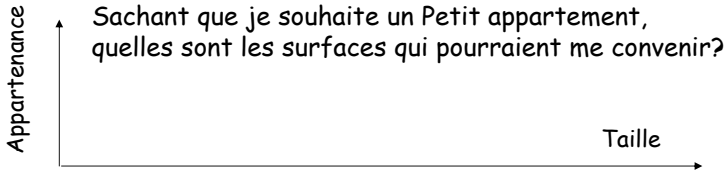
- Degré de similarité :
  - $\mu_A(x)=1$ désigne un élément prototypique de A,
  - $\mu_A(x)=0$ désigne un élément dissemblable des prototypes,
  - $\mu_A(x)\in]0,1[$ indique la proximité, la ressemblance de x avec le plus proche prototype,



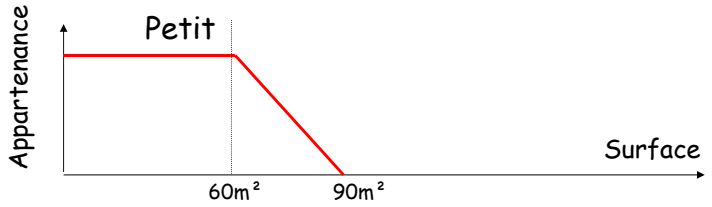
- Degré de préférence :
  - *A* représente une contrainte flexible sur une variable ou un ensemble d'objets plus ou moins préférés.
  - $\mu_A(x)$ caractérise alors l'intensité d'acceptation de la valeur *x* pour la variable ou de la préférence en faveur de l'objet *x*.

- Utilisation :
  - Optimisation
  - Programmation linéaire
  - Aide à la décision
- Applications :
  - Conception
  - Ordonnancement
  - Recherche opérationnelle

- Degré de préférence :



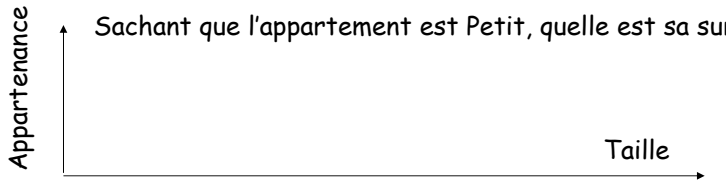
- Degré de préférence :
  - $\mu_A(x)=1$ désigne un élément complètement accepté,
  - $\mu_A(x)=0$ désigne un élément rejeté,
  - $\mu_A(x) \in]0,1[$ indique la préférence graduelle pour x .



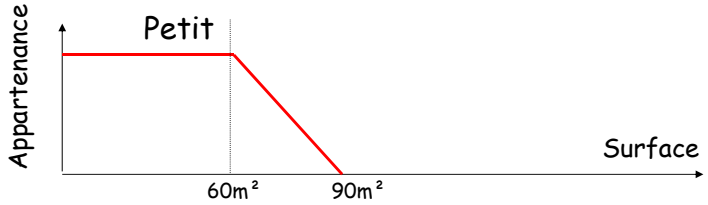
- Degré d'incertitude :
  - $\mu_A(x)$ représente le degré de possibilité qu'une variable u prenne la valeur x sachant uniquement que "u est A".

- Utilisation :
  - Raisonnement
  - Intelligence Artificielle
- Applications :
  - Systèmes experts

- Degré d'incertitude :
  - A représente une connaissance imparfaite sur un objet U
  - $\mu_A(x) = \pi_U(x)$ représente le degré de possibilité qu'une variable U prenne la valeur x sachant uniquement que "U est A".



- Degré d'incertitude :
  - $\mu_A(x)=\pi_U(x)=1$ signifie que U=x est possible,
  - $\mu_A(x)=\pi_U(x)=0$ signifie que U=x est impossible,
  - $\mu_A(x)\in]0,1[$ indique que des valeurs de U sont plus possibles que d'autres .



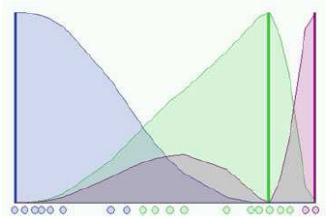
- Exemple : Soit une machine produisant des pièces.
  - Degré de similarité :
    Sachant qu'elle a produit 257 pièces aujourd'hui, comment peut-on qualifier la production ?
    Bonne, Moyenne, Mauvaise
  - Degré de préférence :
    On cherche une machine dont la production est bonne, le ssef BONNE caractérise les valeurs acceptables
  - Degré d'incertitude :
    On sait que la machine a eu une production qui est MAUVAISE, le ssef MAUVAISE caractérise cette connaissance

- Ontic Fuzzy Sets:
  - Conjunctive fuzzy sets represent objects originally construed as sets but for which a fuzzy representation is more expressive due to gradual boundaries.
- Epistemic Fuzzy Sets:
  - Disjunctive understanding of fuzzy sets refers to the representation of incomplete knowledge. Then, a fuzzy set does not model objects or quantities directly, but it models partial information about an underlying object or a precise quantity.

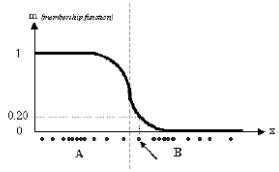
⇒ Sens attaché aux SEF

- L'algorithme des Fuzzy C-Means est une extension des K-Means dans le domaine de la théorie des Ensembles Flous (Zadeh -1965):
  - Prendre en compte l'incertain, l'imprécis.
  - Intégrer l'incomplétude et la gradualité.
- FCM a été crée par Dunn en 1973 et améliorée par J. Bezdeck en 1981
- Il permet une appartenance graduelle d'un point à plusieurs cluster
- Une fonction d'appartenance détermine le degré d'appartenance de chaque point à chaque cluster.
- Souvent utilisé dans la reconnaissance des motifs
  - (pattern recognition)

An Example



- With FCM, the same datum does not belong exclusively to one cluster, but it may belong to several clusters with different values of the membership coefficient



La matrice U pour l'algorithme k-moyennes

$$\begin{pmatrix} u_{11} & \cdots & u_{1k} \\ \vdots & \ddots & \vdots \\ u_{n1} & \cdots & u_{nk} \end{pmatrix}$$

sur chaque ligne, il y a que un seul 1
les autres valeurs sont des 0

La matrice U pour l'algorithme c-moyennes

n données
k clusters

$$\begin{pmatrix} u_{11} & \cdots & u_{1k} \\ \vdots & \ddots & \vdots \\ u_{n1} & \cdots & u_{nk} \end{pmatrix} \quad \forall i \; \forall j \left(0 < u_{ij} < 1\right) \quad \forall i \left(\sum_{j=1}^{k} u_{ij} = 1\right)$$

Suppose we have a data set $X = \{x_1, x_2, \ldots, x_n\} \subset R^p$.

A *c-partition* of X is a c × n matrix $U = [U_1 U_2 \ldots U_n] = [u_{ik}]$, where U_n denotes the *k-th* column of *U*.

There can be three types of *c*-partitions whose columns corresponds to three types of label vectors

Three sets of label vectors in R^c :

$N_{pc} = \{ y \in R^c : y_i \in [0\ 1]\ \forall i, \quad y_i > 0\ \exists i\}$ ***Possibilistic Label***

$N_{fc} = \{y \in N_{pc} : \Sigma y_i = 1\}$ ***Fuzzy Label***

$N_{hc} = \{y \in N_{fc} : y_i \in \{0, 1\}\ \forall i\}$ ***Hard Label***

The three corresponding types of c-partitions are:

$$M_{pcn} = \left\{ U \in R^{cn} : \mathbf{U}_k \in N_{pc}\ \forall k; 0 < \sum_{k=1}^{n} u_{ik}\ \forall i \right\}$$

$$M_{fcn} = \left\{ U \in M_{pcn} : \mathbf{U}_k \in N_{fc}\ \forall k \right\}$$

$$M_{hcn} = \left\{ U \in M_{fcn} : \mathbf{U}_k \in N_{fc}\ \forall k \right\}$$

These are the **Possibilistic**, **Fuzzy** and **Hard** c-partitions respectively

Let X = {x1 = peach, x2 = plum, x3 = nectarine}

Nectarine is a peach plum hybrid.

Typical c=2 partitions of these objects are:

$U_1 \in M_{h23}$

x_1	x_2	x_3
1.0	0.0	0.0
0.0	1.0	1.0

$U_2 \in M_{f23}$

x_1	x_2	x_3
1.0	0.2	0.4
0.0	0.8	0.6

$U_3 \in M_{p23}$

x_1	x_2	x_3
1.0	0.2	0.5
0.0	0.8	0.6

- Useful in Fuzzy Modeling
  - Identification of the fuzzy rules needed to describe a "black box" system, on the basis of observed vectors of inputs and outputs
- History
  - Isodata: Dunn, 1973
  - FCM: Bezdek, 1981
  - PCM: Krishnapuram - Keller, 1993
  - FPCM: N. Pal - K. Pal - Bezdek, 1997

Prof. Bezdek

Input, Output:

- Input: Unlabeled data set $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$

n is the number of data point i $\mathbf{X}$

$\mathbf{x}_k \in \Re^p$ is the number of features in each vector $\mathbf{v}_i$

- Main Output

A c-partition of X, which $c \times n$ matrix U

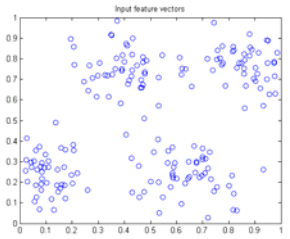
- Common Additional Output

Set of vectors $\mathbf{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_c\} \subset \Re^p$ is called "cluster center"

Sample Illustration:

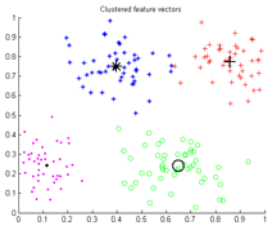
$\mathbf{X}$

$n=188$

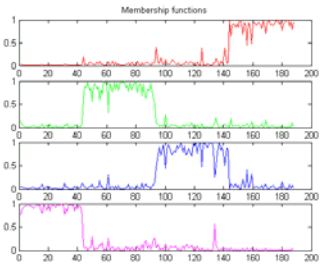


U and V

$c=4$



Rows of U (Membership Functions)



(FCM), Objective Function:

- Optimization of an "objective function" or "performance index"

$$\min_{(\mathbf{U},\mathbf{V})}\left\{ J_m(\mathbf{U},\mathbf{V}) = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^m D_{ik}^2 \right\}$$

Constraint $\sum_{i=1}^{c} u_{ik} = 1 \ , \forall k$

Distance $D_{ik}^2 = \left\| \mathbf{x}_k - \mathbf{v}_i \right\|_{\mathbf{A}}^2$

A-norm $\|\mathbf{x}\|_{\mathbf{A}} = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle_{\mathbf{A}}} = \sqrt{\mathbf{x}^T \mathbf{A} \mathbf{x}}$

Degree of Fuzzification $m \geq 1$

Minimizing Objective Function:

- Zeroing the gradient of J_m with respect to $\mathbf{V}$

$$u_{ik} = \left[\sum_{j=1}^{c} \left(\frac{D_{ik}}{D_{jk}} \right)^{\frac{2}{m-1}} \right]^{-1}, \forall \ i, k \qquad \mathbf{U}_t = F_\partial(\mathbf{V}_{t-1})$$

- Zeroing the gradient of J_m with respect to $\mathbf{U}$

$$\mathbf{v}_i = \left(\sum_{k=1}^{n} u_{ik}^m \mathbf{x}_k \Big/ \sum_{k=1}^{n} u_{ik}^m \right), \forall i \qquad \mathbf{V}_t = G_\partial(\mathbf{U}_{t-1})$$

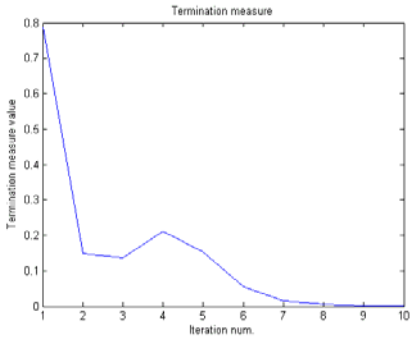
Note: It is the Center of Gravity

- Initial Choices
  - Number of clusters $1<c<n$
  - Maximum number of iterations (Typ.: 100) T
  - Weighting exponent (Fuzziness degree) m
    - m=1: crisp
    - m=2: Typical
  - Termination measure $E_t = \|\mathbf{V}_t - \mathbf{V}_{t-1}\|$ ← 1-norm
  - Termination threshold (Typ. 0.01) $0<\varepsilon$

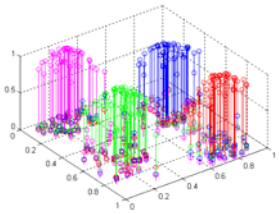
- Guess Initial Cluster Centers $\mathbf{V}_0 = (\mathbf{v}_{1,0} \cdots, \mathbf{v}_{c,0}) \in \Re^{cp}$
- Alternating Optimization (AO)
  - $t \leftarrow 0$
  - REPEAT
  - $t \leftarrow t+1$
  - $\mathbf{U}_t = F_\partial(\mathbf{V}_{t-1})$
  - $\mathbf{V}_t = G_\partial(\mathbf{U}_{t-1})$
  - UNTIL ($t = T$ or $\|\mathbf{V}_t - \mathbf{V}_{t-1}\| \leq \varepsilon$)
  - $(\mathbf{U}, \mathbf{V}) \leftarrow (\mathbf{U}_t, \mathbf{V}_t)$

Sample Termination Measure Plot: $m = 2.0$

Termination Measure Values



Final Membership Degrees



Implementation Notes

- Process could be shifted one half cycle
  - Initialization is done on $\mathbf{U}_0$
  - Iterates become $\mathbf{U}_{t-1} \rightarrow \mathbf{V}_t \rightarrow \mathbf{U}_t$
  - Termination criterion $\|\mathbf{U}_t - \mathbf{U}_{t-1}\| \leq \varepsilon$
- The convergence theory is the same in either case
- Initializing and terminating on V is advantageous
  - Convenience
  - Speed
  - Storage

- Advantages
  - Unsupervised
  - Always converges
- Disadvantages
  - Long computational time
  - Sensitivity to the initial guess (speed, local minima)
  - Sensitivity to noise
    - One expects low (or even no) membership degree for outliers (noisy points)

Optimal Number of Clusters:

- Performance Index

$$\min_{(c)} \left\{ P(c) = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^{m} \left(\|\mathbf{x}_k - \mathbf{v}_i\|^2 - \|\mathbf{v}_i - \bar{\mathbf{x}}\|^2 \right) \right\}$$

Average of all feature vectors $\bar{\mathbf{x}} = \frac{1}{n} \sum_{k=1}^{n} \mathbf{x}_k$

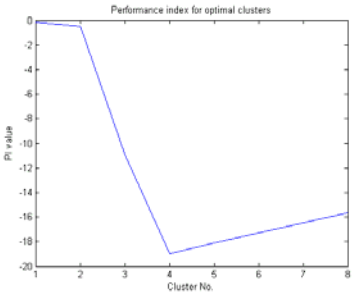
$\sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^{m} \left(\|\mathbf{x}_k - \mathbf{v}_i\|^2 \right)$ — $\sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^{m} \left(\|\mathbf{v}_i - \bar{\mathbf{x}}\|^2 \right)$

Sum of the within fuzzy cluster fluctuations (small value for optimal c)

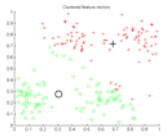
Sum of the between fuzzy cluster fluctuations (big value for optimal c)

Optimal Cluster No. (Example):

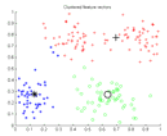
Performance index for optimal clusters (is minimum for c = 4)



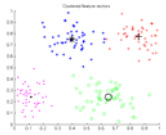
c = 2



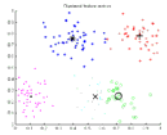
c = 3



c = 4



c = 5



- This method (GK) is fuzzy clustering method similar to the Fuzzy C-means (FCM).
- The difference is the way the distance is calculated:
  - FCM uses Euclidean distances
  - GK uses Mahalobis distances

- Mahalanobis distance is calculated as

$$d_{ik}^2 = (\mathbf{x}_k - \mathbf{v}_i)^T \mathbf{A}_i (\mathbf{x}_k - \mathbf{v}_i)$$

- The matrices $\mathbf{A}_i$ are given by

$$\mathbf{A}_i = \sqrt[p]{\det(\Sigma_i)} \cdot \Sigma_i^{-1}$$

- The Fuzzy Covariance Matrix is

$$\Sigma_i = \frac{\sum_{j=1}^{n} \mu_{ij}^m (\mathbf{x}_j - \mathbf{v}_i)(\mathbf{x}_j - \mathbf{v}_i)^T}{\sum_{j=1}^{n} \mu_{ij}^m}$$

- The clusters are hyperellipsoids on the $\Re^l$.
- The hyperellipsoids have aproximately the same size.
- In order to be possible to calculate S^{-1} the number of samples n must be at least equal to the number of dimensions l plus 1.

Limitations des arbres de décision binaire classiques

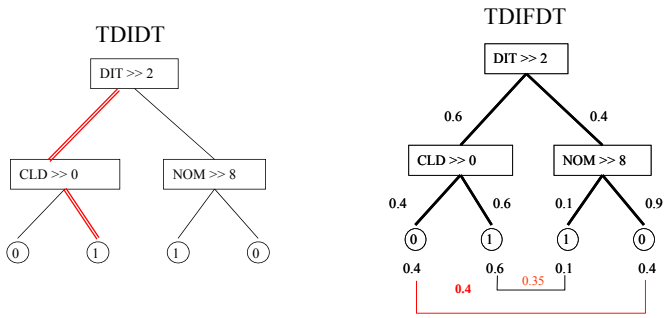
- Le critère de sélection pour diviser les données lors de la construction de l'arbre n'est pas toujours approprié.
- L'arbre est sensible au bruit dans l'ensemble d'apprentissage.
- *Le processus de décision dépend de valeurs seuils*
  - ↳NOM <= 20 -> class 1
  - ↳NOM > 20 -> class 0

D'où vient 20 ? Pourquoi pas 19.9 ou 20.1 ?

Limitations des arbres de décision binaire classiques

- Le processus de classification suit le premier chemin valide

  Exemple (classe avec DIT=3, CLD=0, NOM=4)



Avantages potentiels d'un arbre de décision flou

- L'usage de valeurs linguistiques élimine le problème des valeurs seuils
- Tous les chemins de l'arbre sont évalués lors du processus de classification
- Meilleur pouvoir de généralisation entre l'ensemble d'apprentissage et l'ensemble test
- Meilleure robustesse face au bruit
- Règles plus facilement interprétables

Induction d'un arbre de décision flou

Similaire à l'algorithm C4.5 de Quinlan

```
Function induce_fuzzy_tree (Example set E, Properties P)
Begin
 if all entries in E are in the same class
   then return a leaf node labeled with that class
 else if P is empty
   then return leaf node labeled with disjunction of all classes in E
   else begin
        fuzzify E;
        select a property pr of P, and make it the root of the current tree;
        for each fuzzy partition f, of pr,
        begin
          create a branch of the tree labeled with f;
          let partition pa be elements of E with f  for pr;
         call induce_fuzzy_tree (pa, P), attach result to branch f
        end
   end
end
```

Sélection de la propriété ayant le meilleur pouvoir de représentation

- *Dans TDIDT, on utilise l'entropie classique. Pour $A=\{a_i\}_{i=1,\dots,n}$:*

$$H(A) = -\sum_{i=1}^{n} P(a_i)\log_2\left(P(a_i)\right)$$

où $p(a_i)$ est la probabilité que $A=a_i$.

- On définit l'entropie floue, ou entropie-étoile de manière similaire :

$$H^*(A) = -\sum_{i=1}^{n_p} P^*(v_i)log_2\left(P^*(v_i)\right)$$

est la probabilité floue que $A=v_i$:

$$P_A^*(v_i) = \sum_{1\le l\le n} \mu_{v_i}(a_l)P(a_l)$$

$P^*(v_i)$

- $\mu_{v_i}(a_l)$: fonction d'appartenance d'une valeur a_l à v_i
- $P(a_l)$: fréquence de a_l dans l'ensemble d'apprentissage

Sélection de la propriété ayant le meilleur pouvoir de représentation

- Comme chaque attribut est commun à toutes les classes, ses différentes valeurs floues sont distribuées en conséquence :

$$E_{A_j}^* = -\sum_i P^*(v_i)\sum_j P^*(c_j \mid v_i)\log_2 P^*(c_j \mid v_i)$$

- $P^*(v_i) = \sum_{1\le l\le n} \mu_{v_i}(a_l)P(a_l)$
- $P^*(c_j \mid v_i) = \dfrac{P^*(c_j, v_i)}{P^*(v_i)}$
- $P^*(c_j, v_i) = \sum_{1\le l\le q}\sum_{1\le k\le n} min(\mu_{c_j}(c_l), \mu_{v_i}(a_k))P(c_l, a_k)$

- Choisir A_j ayant $E_{A_j}^*$ min. comme critère de division

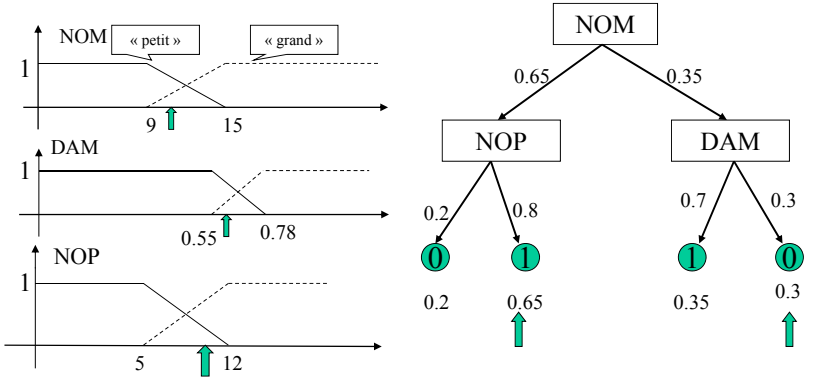
Procédure possible d'inférence pour la classification

- Pour chaque arbre :
  - Les données entrent par la racine de chaque arbre et sont propagées vers les feuilles .
  - Utiliser l'algorithme max-min* pour :
    - Déterminer les valeurs d'appartenance de chacune des feuilles au label associé (min)
    - En déduire la valeur floue de chaque label (max).
- Partant de tous les arbres, utiliser la méthode du vote majoritaire pour identifier la classe d'appartenance des données

*Max-min : min(μ) le long de chaque chemin, max(μ) pour chaque label de sortie

Exemple

- Données : (NOM=11, NOP=11, DAM=0.6)



- L'utilisation de max- min donne :
  - μ(1) = 0.65
  - μ(2) = 0.3

Exemple - suite

- Méthode du vote :

$$\mu(c_i) = \frac{\mu(\{c_i\}) + \frac{1}{|n-1|}\sum_{c_j \neq c_i} \mu_{c_j}(\{\overline{c_j}\})}{|n-1|}$$

- On prend la classe qui obtient le plus grand μ.

Ex : pour E1, $\mu(C_1) = (1 + 1/2(0.3+0.8))/2 = 0.775$

Ex.	Tree 1		Tree 2		Tree 3		Vote		
	C1	$\overline{C1}$	C2	$\overline{C2}$	C3	$\overline{C3}$	C1	C2	C3
E1	1	0	0	0.3	0	0.8	**0.78**	0.2	0.08
E2	0.55	0.6	0.3	0.7	0.32	0.48	**0.57**	0.42	0.48
E3	0.1	0	0.8	0.1	0	0.6	0.22	**0.55**	0.02

Partition Coefficient

- This coefficient is defined as

$$PCoef = \left(\sum_{i=1}^{c}\sum_{j=1}^{n}(\mu_{ij})^2\right)\Big/ n$$

$$1/c \leq F \leq 1$$

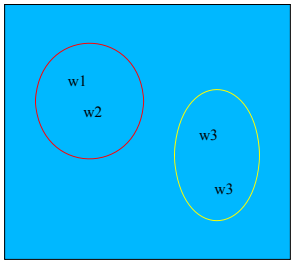
- When F=1/c the system is entirely fuzzy, since every element belongs to all clusters with the same degree of membership
- When F=1 the system is rigid and membership values are either 1 or 0
- F is inversely proportional to the number of clusters.
- F is more appropriated to validate the best partition among several runs of an algorithm

Partition Coefficient Example

- The Partition Matrix is

$$U = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

$$PCoeff = \frac{1^2 + 1^2 + 1^2 + 1^2}{4} = 1$$

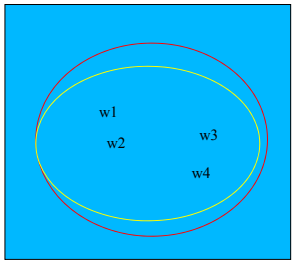


Partition Coefficient Example

- The Partition Matrix is

$$U = \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 \end{bmatrix}$$

$$PCoeff = \frac{8 \times 0.5^2}{4} = 0.5 = 1/2 = 1/c$$



MOTA: Partition Entropy

- Partition Entropy is defined as

$$H = -\left(\sum_{i=1}^{c}\sum_{j=1}^{n}\mu_{ij}\cdot\log(\mu_{ij})\right)\Big/ n$$

$$0 \le H \le \log c$$

  - When H=0 the partition is rigid.
  - When H=log(c) the fuzziness is maximum.
  - 0 <= 1-F <= H

- Partition Entropy (H) is directly proportional to the number of partitions.
- H is more appropriated to validate the best partition among several runs of an algorithm.

MOTA: Compactness and Separation

- CS is defined as

$$CS = \frac{J_m}{n\cdot(d_{\min})^m}$$

- J is the objective function minimized by the FCM algorithm.
- m is the fuzzy factor.
- d_{min} is minimum Euclidean distance between two the center of two clusters.

MOTA: Compactness and Separation

- The minimum distance is defined as

$$d_{\min} = \min_{i,j}\left\|c_i - c_j\right\|$$

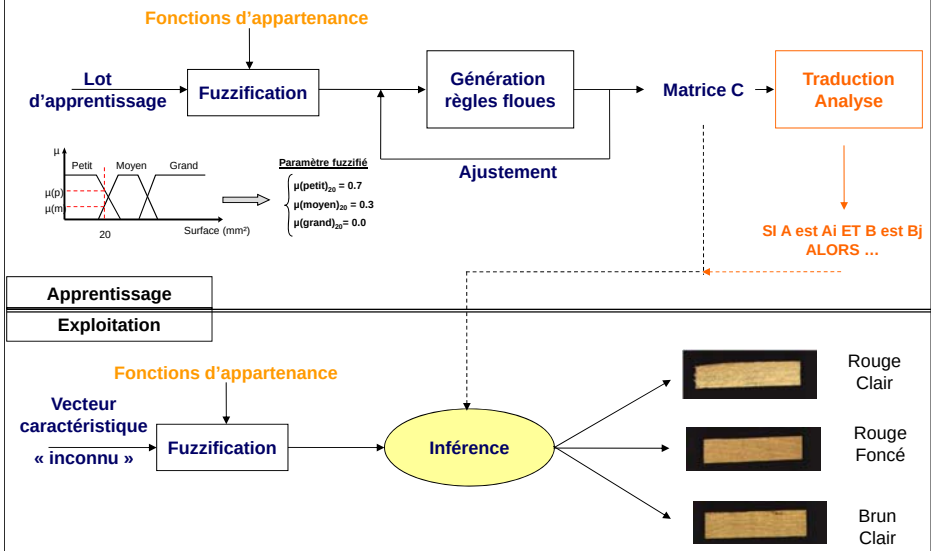
- The complete formula is

$$CS = \frac{\sum_{i=1}^{c}\sum_{j=1}^{n}\mu_{ij}^{m}\left\|\mathbf{v}_i - \mathbf{x}_j\right\|^2}{n\cdot\min_{i,j}\left\|\mathbf{v}_i - \mathbf{v}_j\right\|^2}$$

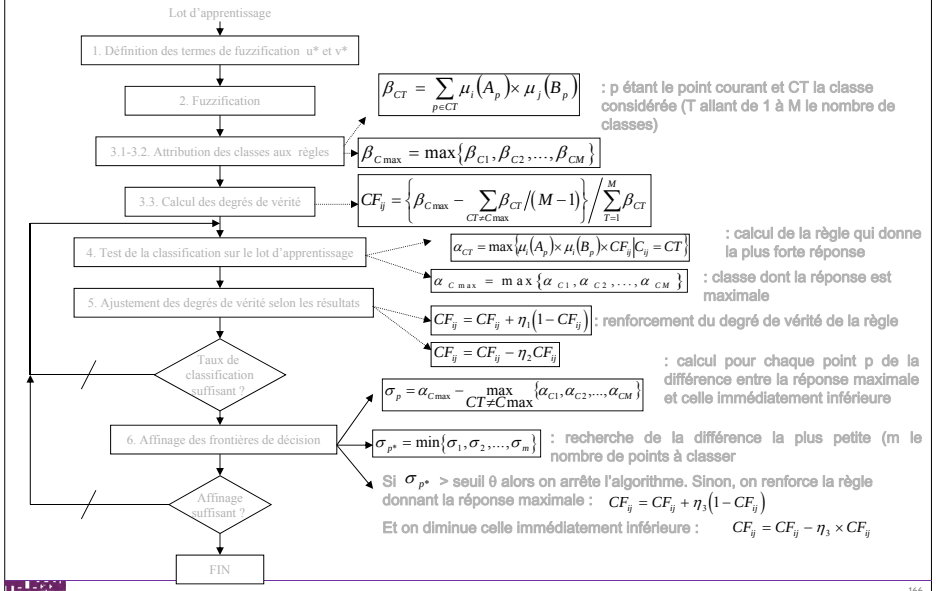
MOTA TEF: Fuzzy Rule Classifier (FRC)

- *V. BOMBARDIER, E. SCHMITT: "Fuzzy rule classifier: Capability for generalization in wood color recognition", Engineering Applications of Artificial Intelligence, Vol 23 (2010) n°6, pp 978-988.*
- Trois étapes pour la phase d'apprentissage :
  - Fuzzification automatique des paramètres d'entrée (Analyse des Typiqualités Schmitt-07)
  - Génération automatique des règles (Algorithme Ishibuchi-92)
  - Ajustement du modèle numérique (version Itérative de l'algorithme Ishibuchi-96)
- Règles Conjonctives à Possibilité (data learning) (Dubois-96)
- Modèles Larsen (Max / Produit - Multi Input) (Berthold-03)
- Sortie Floues (Pas de defuzzification)
  - La décision locale est prise en fonction du maximum de possibilité)

MOTA TEF: Fuzzy Rule Classifier (FRC)

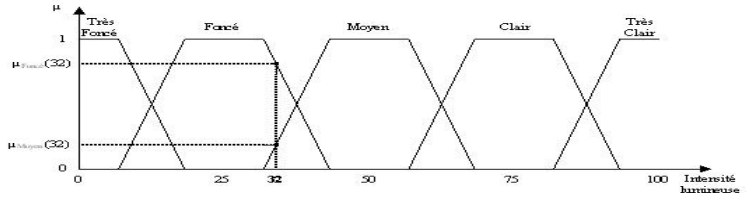


MOTA TEF: FRC: Algorithme



MOTA TEF: F.R.C. : Fuzzification équirépartie Fuzzification adaptée

- Plusieurs techniques de fuzzification (numérique → symbolique)
  - Fuzzification équirépartie



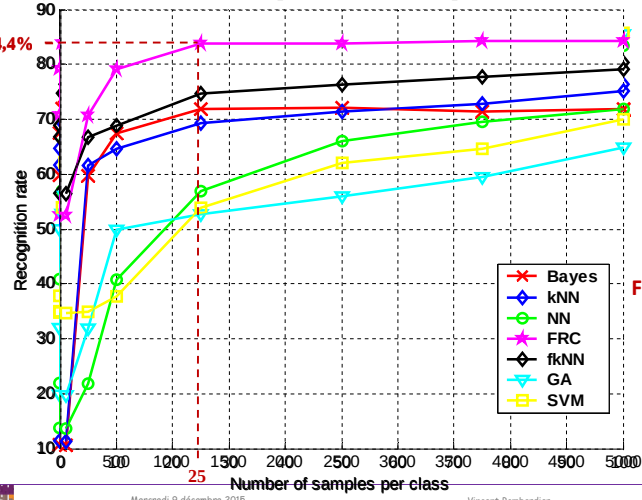
➔Choix des courbes triangulaires-trapézoïdales

- Fuzzification adaptée aux connaissances (entrées / sorties)

Base de données	Taux de reconnaissance	Nombres de règles générées
Base 1	83.25%	48
Base 2	84.45%	108

MOTA TEF: F.R.C. : Generalization Capability

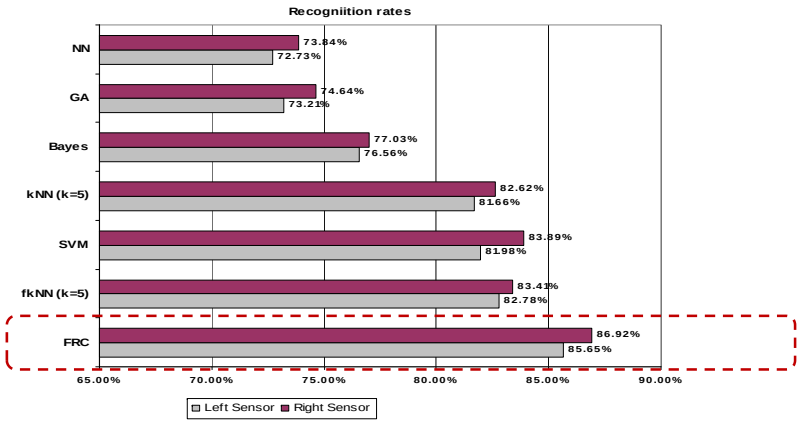
- Using Artificial Database composed of 5000 samples (Gaussian white noise)



Similar maxima : SVM, NN, GA and FRC

FRC needs less samples to reach the maximum

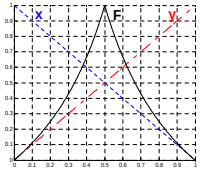
Classification Rates Left and Right Sensors



Fuzzy Merging : F Operator Definition

- Three kinds of operators (Bloch-96)
  Strict: $F(x,y) \le \min(x,y)$
  Indulgent: $F(x,y) \ge \max(x,y)$
  Wise: $\begin{cases} \text{if } x \le y \text{ then } x \le F(x,y) \le y \\ \text{if } y \le x \text{ then } y \le F(x,y) \le x \end{cases}$
- Proposed Operator (Perez-Oramas 2000)

$$F(x,y) = \min\left[1, \frac{\min(x,y)}{1-\min(x,y)}\right]$$

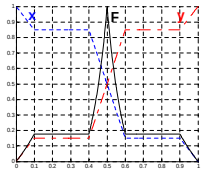


- If x and y > μ_S=0.5 then

$$F(x,y) = \min\left(1, \frac{x}{1-x}\right) \text{ si } x < y < \mu_S \text{ ou } x < \mu_S < y$$

$$F(x,y) = \min\left(1, \frac{y}{1-y}\right) \text{ si } y < x < \mu_S \text{ ou } y < \mu_S < x$$

$$F(x,y) = 1 \text{ si } x \ge \mu_S \text{ et } y \ge \mu_S$$



Fuzzy Merging vs. Symbolic Merging: advantages

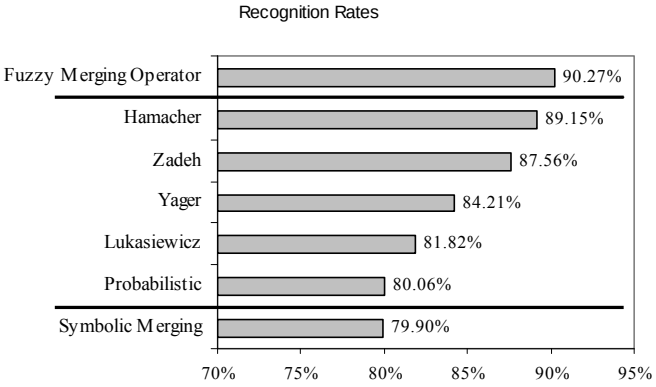
- Symbolic multi face merging tables used by the industrialist:

	Dark Brown	Brown	Light Brown	Dark Red	Red	Light Red
Dark Brown	DB	*Rejected*	*Rejected*	*Rejected*	*Rejected*	*Rejected*
Brown	*Rejected*	B	*Rejected*	*Rejected*	*Rejected*	*Rejected*
Light Brown	*Rejected*	*Rejected*	LB	*Rejected*	*Rejected*	*Rejected*
Dark Red	*Rejected*	*Rejected*	*Rejected*	DR	*Rejected*	*Rejected*
Red	*Rejected*	*Rejected*	*Rejected*	*Rejected*	R	*Rejected*
Light Red	*Rejected*	*Rejected*	*Rejected*	*Rejected*	*Rejected*	LR

	Left Sensor	Right Sensor
μ_{max}	μ (DR) = 0.8	μ (DB) = 0.6
μ_{max-1}	μ (DB) = 0.2	μ (DR) = 0.4
μ_{max-2}	μ (B) = 0.1	μ (B) = 0.1
Max	Dark Red	Dark Brown

Merging	Dark Red	Brown	Dark Brown
F(c1,c2)	0.67	0.11	0.25

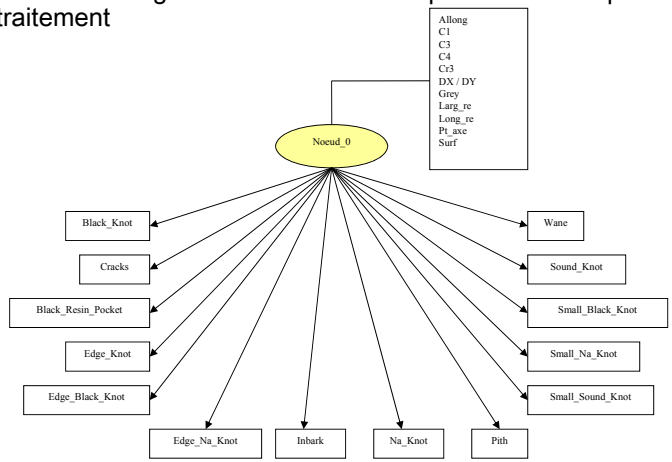
Fuzzy Merging: Results



➔**Classification Rate increasing up to 10% vs Symbolic Merging (industrial use)**

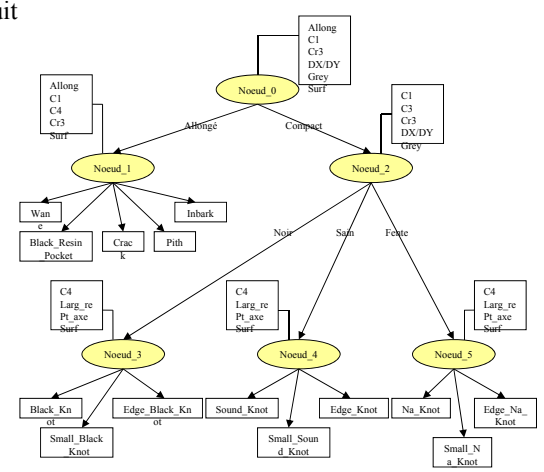
MOTA TEF: F.R.C. : Inférence Unique

- **Système d'Inférence Floue Unique :**
  - 11 paramètres en entrée / 14 classes de défauts en sortie
  - Nombre de règles très élevé : Pb interprétation et temps de traitement



MOTA TEF: F.R.C. : Inférences Multiples

- **Système d'Inférences Floues Arborescent :** ***FRC Arborescent***
  - Temps de traitement réduit
  - Interprétabilité des S.I.F.



BOMBARDIER V., MAZAUD C., LHOSTE P., VOGRIG R.
« Contribution of Fuzzy Reasoning Method to knowledge Integration in a wood defect Recognition System », Computers in Industry Journal, Elseviers, 12 pages, vol 58, issue 4 (mai 2007), pp 355-366