Apprentissage automatique/artificiel

Julien Grosjean

M1 GIL
Année 2016-2017 S2

Plan du cours

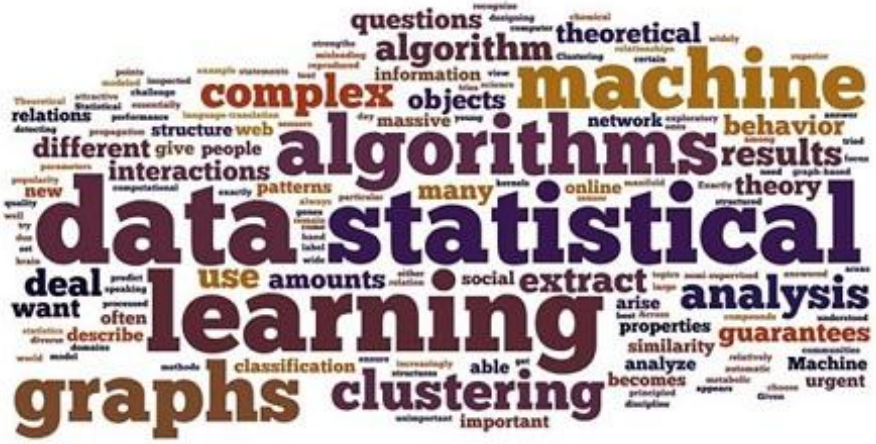
- Introduction

- L'apprentissage supervisé
- L'apprentissage non supervisé
- Autres types d'apprentissage

- Synthèse

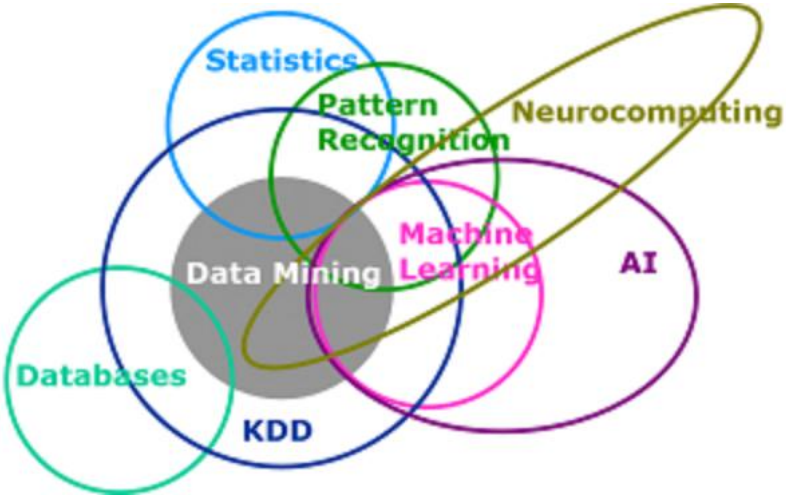
Axes du cours

- Connaitre et comprendre les différentes approches et dans quels contextes les utiliser (avantages, inconvénients, quelle approche pour quelles données ?)
- Connaitre les méthodes et outils (algorithmes, langages et outils statistiques) présentés
- S'approprier le vocabulaire spécifique



Définition(s)

- Automatique ⇔ Artificiel
- *Machine Learning*



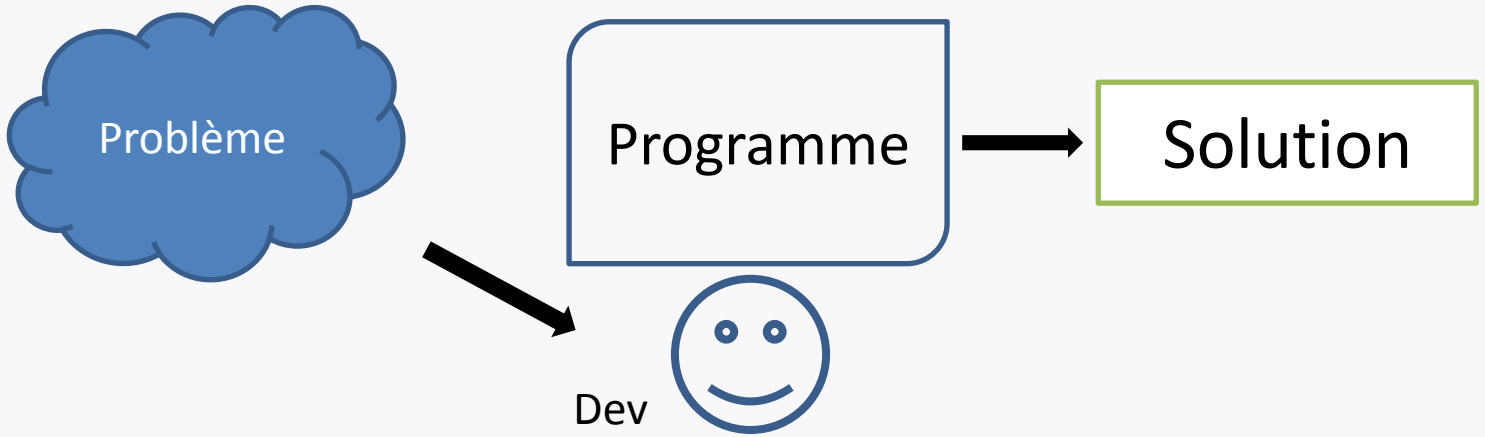
- Acquisition des connaissances ?
- Suite de règles (inférence) ?
- Processus systématique permettant de résoudre des problèmes trop complexes via des méthodes et algorithmes classiques (NP complexes, modèles inconnus, etc.)
- …

But

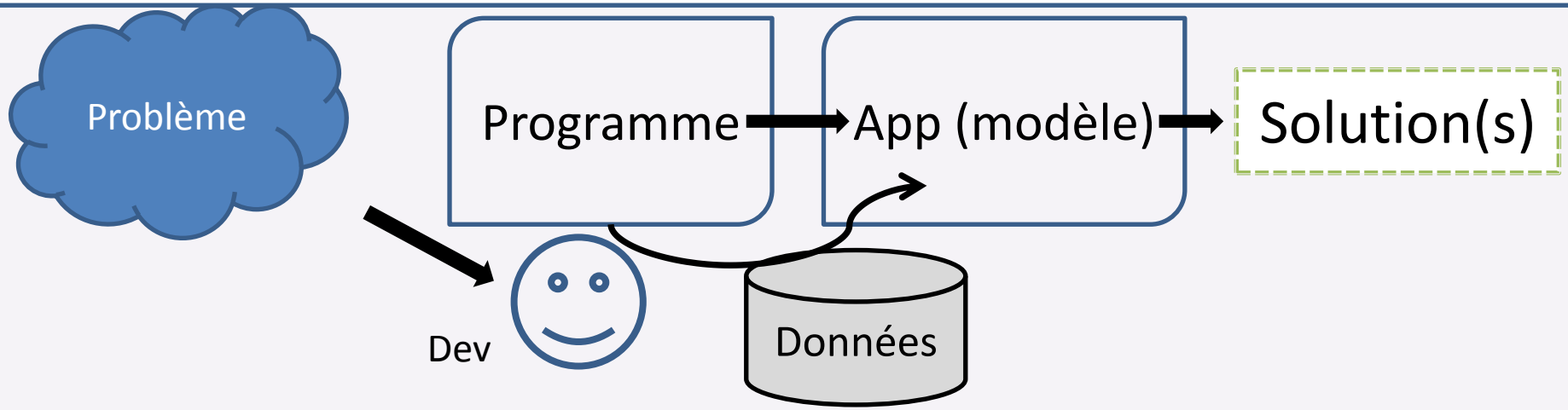
- Remplacer les experts humains (systèmes experts ++) : raisonnement
- Produire automatiquement des règles (modèle) à partir de données avec ou sans expertise humaine (les 2 grandes approches d'apprentissage)
- Éventuellement réutiliser ces règles
- Éventuellement comprendre ces règles

Démarche

Algo connu



Pas d'algo connu



Contraintes

- Les données (étiquetage, « interprétabilité », bruit)
- Les algorithmes
- Les outils
- La complexité (mémoire, temps) ~ technique

Applications

- Processus de décision ou de découvertes
- Répondre à des questions comme :
  - mon patient aura-t-il un accident cardio-vasculaire dans les cinq ans à venir ?
  - quel sera le résultat du prochain Rouen – Angers ?
  - la molécule que je désire commercialiser est-elle cancérigène ?
  - quel est l'auteur de cette page HTML ?
  - à quelle espèce appartient cet oiseau ?
  - cette phrase est-elle grammaticalement correcte ?
  - qui a gagné cette partie de morpion ?
  - quelle sera la taille de cet enfant à l'âge adulte ?

L’apprentissage

- Les « Cinq questions définitives » (Mergel) :
  1. Comment l’apprentissage se produit-il ?
  2. Quels facteurs influent sur l’apprentissage ?
  3. Quel est le rôle de la mémoire
  4. Comment le transfert du savoir se produit-il ?
  5. Quelles pratiques d’apprentissage sont mieux expliquées par cette théorie ?

Les méthodes d'apprentissage

- Par imitation
- Par induction (le « bon sens »)
- Par association (action -> réaction)
- Par essais et erreurs
- Par explication (algorithme)
- Par répétition (renforcement)
- Combiné
- Par immersion

Apprentissage supervisé

Apprentissage supervisé : principe

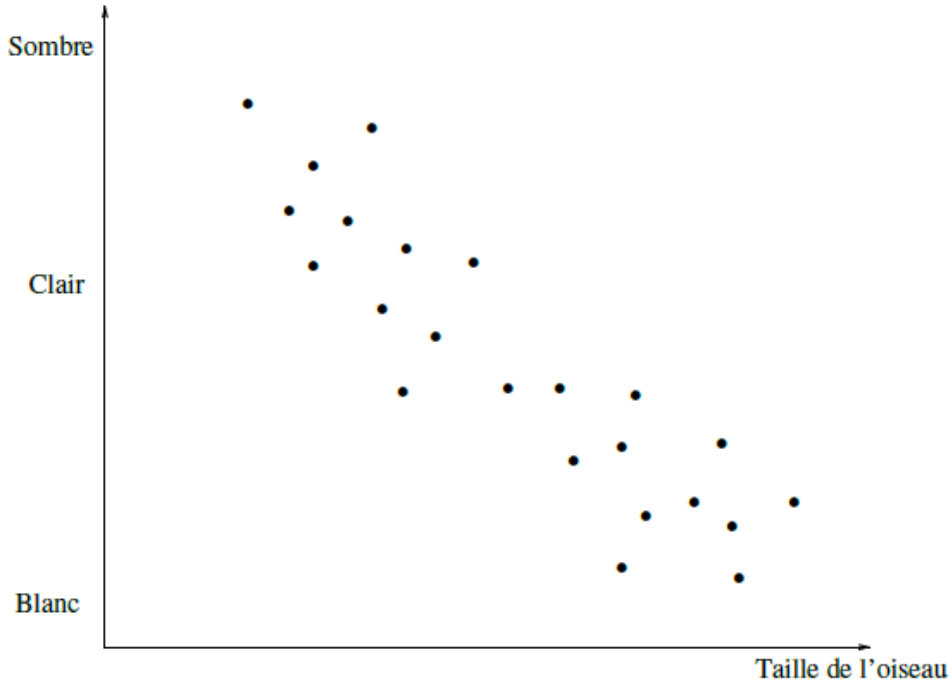
- *Classification*, classement
- Classer des instances à partir d'un jeu d'exemples étiquetés par leurs classes (par un humain) = prédiction
- Apprendre la fonction f à partir d'un ensemble de paires $(x, f(x))$ = modèle
- Régression : prédiction de valeur(s) numérique(s)
- Classification : prédictions de valeurs appartenant à un ensemble discret

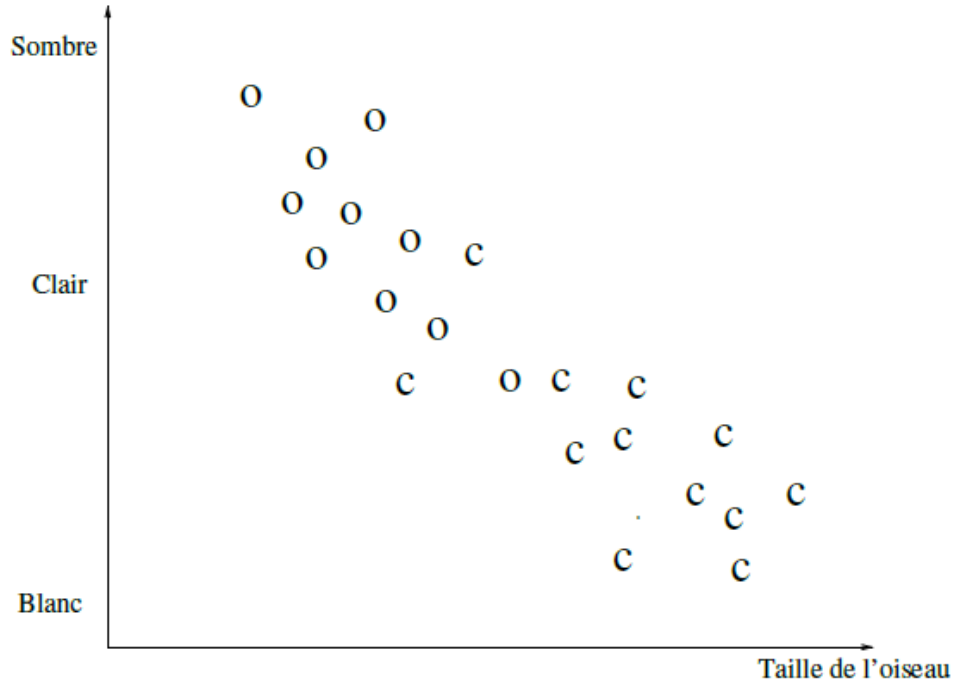
Les types de réponse

- Binaires : *oui* ou *non* (paire de valeurs)
- Discrète (plus de 2 valeurs)
  - Classe à prédire
- Continues
  - Régression => discrétiser ?

Un exemple

Des oies ou des cygnes ?

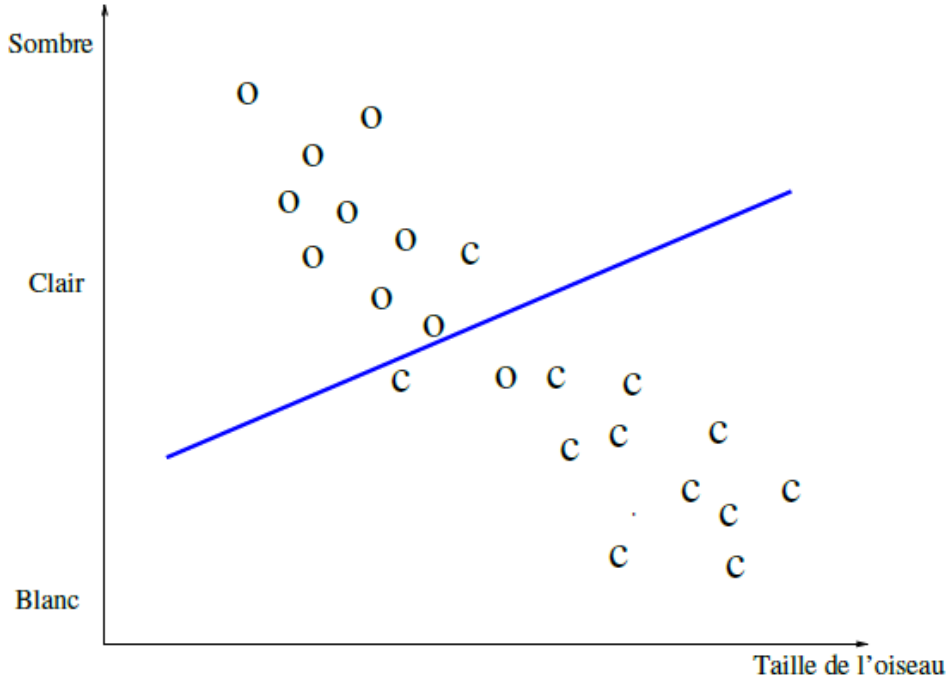


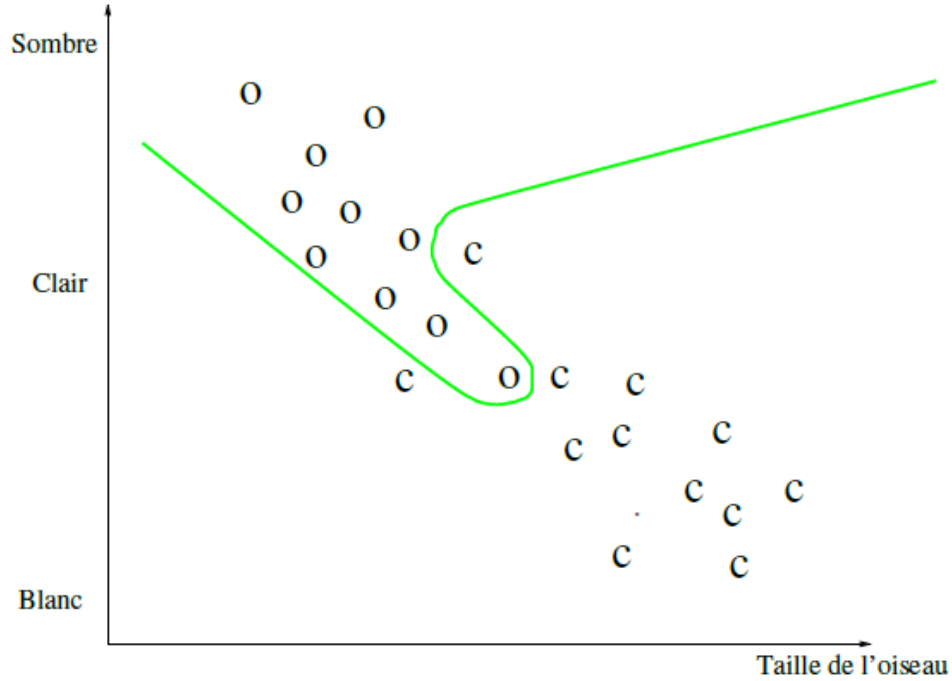


Aide de l'expert

Un exemple

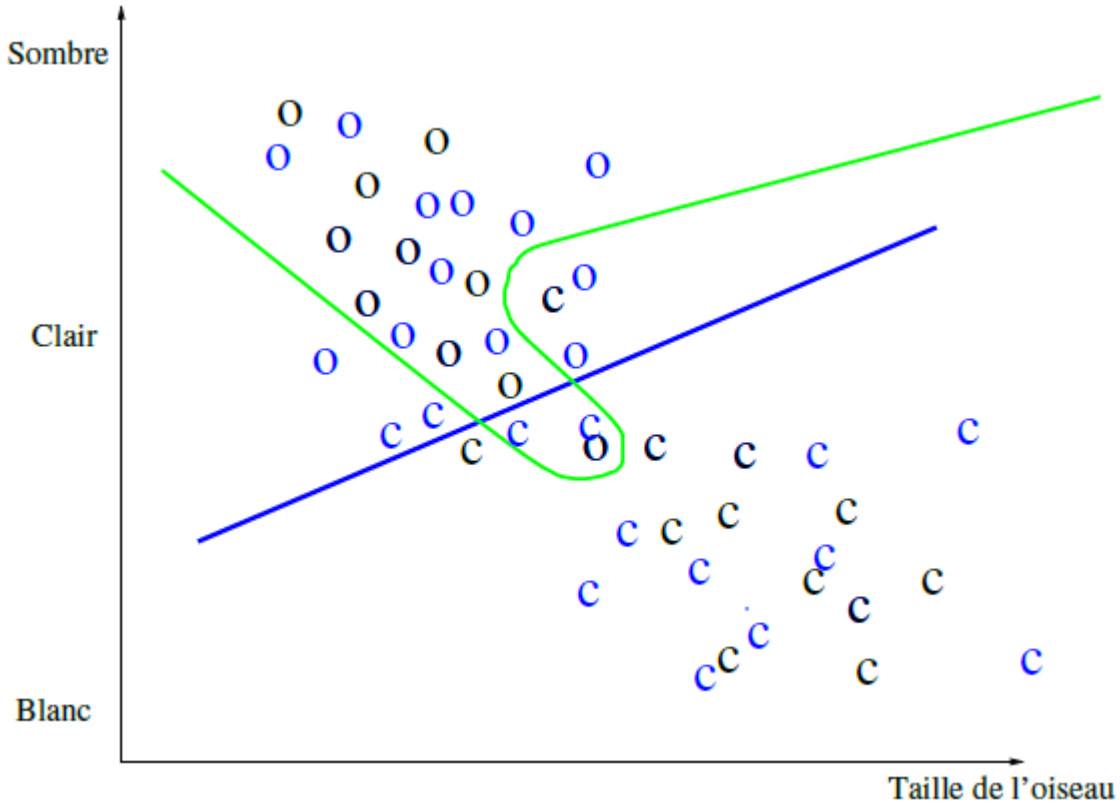
Deux hypothèses pour classer





Un exemple

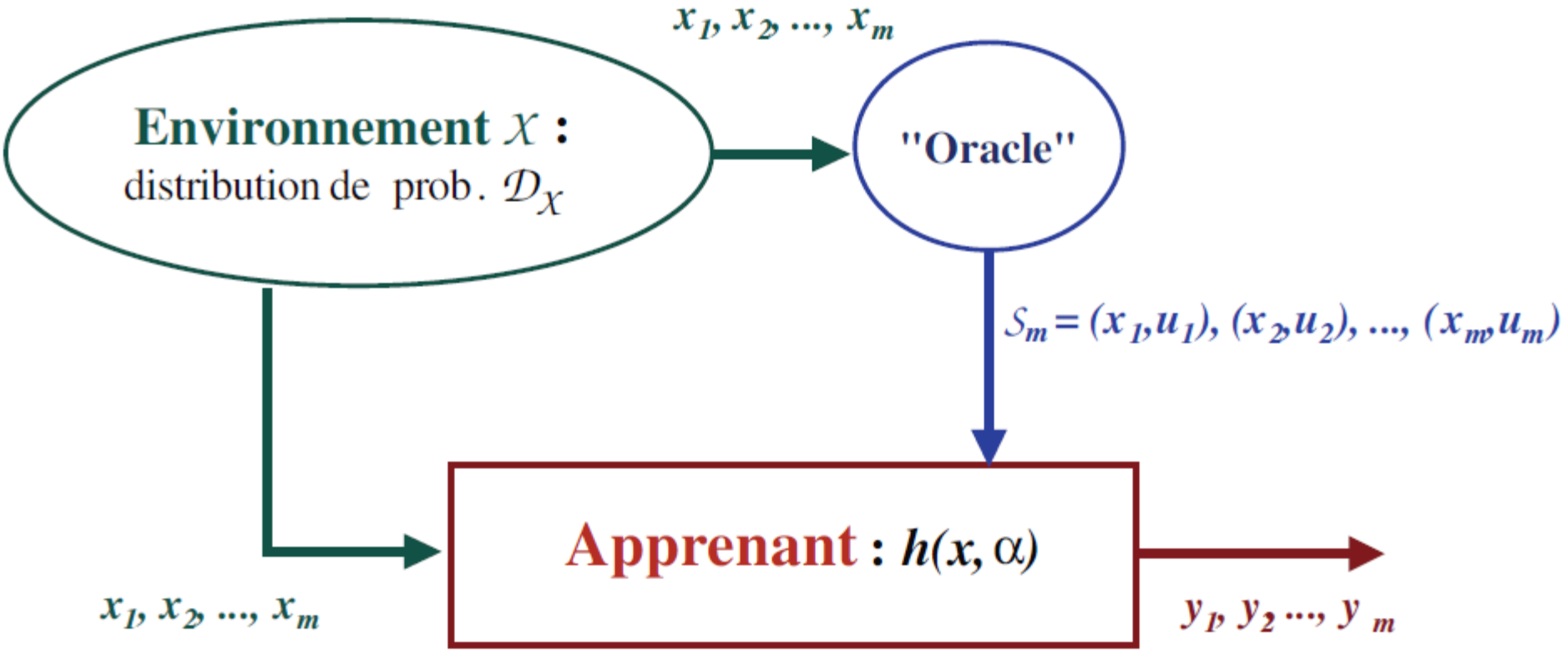
Nouveaux spécimens



Principaux algorithmes/outils

- Arbres de décision
- Classification naïve bayésienne
- Réseaux de neurones
- Méthode des K-NN
- SVM
- Analyse discriminante linéaire

Scénario d'apprentissage supervisé



Aspects formels

- Multi-ensemble $E \subset A_1 \times \ldots \times A_n \times C$
- A_i sont les attributs
- C est l'attribut cible $\Rightarrow$ classe
- Chaque attribut A a un domaine *dom*(A) de valeurs possibles

- Trouver une fonction $f : A_1 \times \ldots \times A_n \rightarrow C$
- Fonction idéale ou non

Aspects formels : exemple

- Détecter si un courriel est un spam ou non

	Auteur	MotClés	HTML	Majuscule	Spam
1	inconnu	true	true	false	false
2	inconnu	false	false	true	true
3	inconnu	true	false	false	false
4	inconnu	true	true	true	true
5	connu	false	false	false	false
6	inconnu	false	true	true	true
7	connu	false	false	true	false
8	inconnu	true	true	false	true
9	connu	true	false	false	false
10	inconnu	true	false	true	true
11	connu	true	false	false	false
12	inconnu	false	false	false	false

Aspects formels : exemple

- Quatre attributs
  - *Auteur* { *connu*,*inconnu*}
  - *MotsClés* {*true*,*false*}
  - *HTML* {*true*,*false*}
  - *Majuscule* {*true*,*false*}
- Attribut cible
  - *Spam* {*true*,*false*}

Évaluation

- ~ Recherche d'Information

$$\text{Rappel} = \frac{\#\text{ documents pertinents trouvés}}{\#\text{ documents pertinents}}$$

$$\text{Précision} = \frac{\#\text{ documents pertinents trouvés}}{\#\text{ documents trouvés}}$$

Évaluation

- Matrice de confusion

Prédiction / Classe	-1	+1
-1	VN	FN
+1	FP	VP

- À partir de cette matrice :
  - « Exactitude » *(accuracy)* = $\frac{VP + VN}{VP+VN+FP+FN}$
  - Rappel (*recall*) : $R = \frac{VP}{VP + FN}$ et précision $P = \frac{VP}{VP + FP}$
  - F-mesure : $F = 2 \times \frac{P \times R}{P + R}$
  - Sensibilité $Se = R$ et spécificité $Sp = \frac{VN}{VN + FP}$

Validation croisée

- *Cross-validation*
- Technique d'auto-évaluation
- Séparation des données : apprentissage + test
- Exemple : 10 validations croisée
  - 90% données apprentissage, 10% données test (aléatoire)
- Mesure le taux d'erreur (taux d'observations mal classées) : force du modèle en « circuit fermé »

Arbre de décision

- Construction à partir des données : hiérarchie de tests
- Simple (humains)
- Nécessite peu de données d'entrainement
- Utilisation de l'arbre produit sur des nouvelles données non classées = prédiction
- « Forêt d'arbres de décisions »

Arbre de décision

- Exemple de méthode : algorithme C4.5
- Notion d'entropie (« mélange des classes » dans un ensemble)
- Soit n classes différentes dans un ensemble D, l'entropie vaut :

$$H(D) = -(p_1 \log_2 p_1 + p_2 \log_2 p_2 + \ldots + p_n \log_2 p_n)$$

Où p_i donne la proportion d'éléments dans D étiquetés par la *i*-ème classe

Arbre de décision

- Le gain d'information d'un attribut par rapport à A : différence entre l'entropie avant et après le branchement sur A => plus le gain est important plus l'attribut est utile pour séparer les classes

$$G(A, D) = H(D) - \left(\Sigma_{x \in dom(A)} \frac{|D[A = x]|}{|D|} H(D[A = x])\right)$$

- Exemple Arbre = {Auteur, MotClés, HTML, Majuscule}, C = {Spam}, D l'ensemble des exemples cités précédemment, l'entropie de D est :

$$H(D) = -(\frac{5}{12} \log_2 \frac{5}{12} + \frac{7}{12} \log_2 \frac{7}{12}) = 0,98$$

Arbre de décision : exemple

- Gain de « Auteur »

$$H(D[\mathbf{Auteur} = connu]) = -(\frac{0}{5}\log_2\frac{0}{5} + \frac{5}{5}\log_2\frac{5}{5}) = 0$$

$$H(D[\mathbf{Auteur} = inconnu]) = -(\frac{5}{7}\log_2\frac{5}{7} + \frac{2}{7}\log_2\frac{2}{7}) = 0,86$$

$$G(\mathbf{Auteur}, D) = 0,98 - (\frac{5}{12} * 0 + \frac{7}{12} * 0,86) = 0,48$$

- Gain de « MotClés »

$$H(D[\mathbf{MotClés} = oui]) = -(\frac{3}{7}\log_2\frac{3}{7} + \frac{4}{7}\log_2\frac{4}{7}) = 0,99$$

$$H(D[\mathbf{MotClés} = non]) = -(\frac{2}{5}\log_2\frac{2}{5} + \frac{3}{5}\log_2\frac{3}{5}) = 0,97$$

$$G(\mathbf{MotClés}, D) = 0,98 - (\frac{7}{12} * 0,99 + \frac{5}{12} * 0,97) = 0$$

Arbre de décision : exemple

- Gain de « HTML »

$$H(D[\mathbf{HTML} = oui]) = -(\frac{3}{4}\log_2\frac{3}{4} + \frac{1}{4}\log_2\frac{1}{4}) = 0,81$$

$$H(D[\mathbf{HTML} = non]) = -(\frac{2}{8}\log_2\frac{2}{8} + \frac{6}{8}\log_2\frac{6}{8}) = 0,81$$

$$G(\mathbf{HTML}, D) = 0,98 - (\frac{4}{12} * 0,81 + \frac{8}{12} * 0,81) = 0,17$$

- Gain de « Majuscule »

$$H(D[\mathbf{Majuscule} = oui]) = -(\frac{4}{5}\log_2\frac{4}{5} + \frac{1}{5}\log_2\frac{1}{5}) = 0,72$$

$$H(D[\mathbf{Majuscule} = non]) = -(\frac{1}{7}\log_2\frac{1}{7} + \frac{6}{7}\log_2\frac{6}{7}) = 0,59$$

$$G(\mathbf{Majuscule}, D) = 0,98 - (\frac{5}{12} * 0,72 + \frac{7}{12} * 0,59) = 0,33$$

Arbre de décision : exemple

- Gains à partir de « Auteur (= inconnu) »

$$H(D[\textbf{Auteur} = inconnu][\textbf{MotClés} = oui]) = -(\frac{3}{4}\log_2\frac{3}{4}+\frac{1}{4}\log_2\frac{1}{4}) = 0,81$$

$$H(D[\textbf{Auteur} = inconnu][\textbf{MotClés} = non]) = -(\frac{2}{3}\log_2\frac{2}{3}+\frac{1}{3}\log_2\frac{1}{3}) = 0,92$$

$$G(\textbf{MotClés}, D[\textbf{Auteur} = inconnu]) = 0,86 - (\frac{4}{7} * 0,81 + \frac{3}{7} * 0,92) = 0,01$$

$$H(D[\textbf{Auteur} = inconnu][\textbf{HTML} = oui]) = -(\frac{3}{3}\log_2\frac{3}{3} + \frac{0}{3}\log_2\frac{0}{3}) = 0$$

$$H(D[\textbf{Auteur} = inconnu][\textbf{HTML} = non]) = -(\frac{2}{4}\log_2\frac{2}{4} + \frac{2}{4}\log_2\frac{2}{4}) = 1$$

$$G(\textbf{HTML}, D[\textbf{Auteur} = inconnu]) = 0,86 - (\frac{3}{7} * 0 + \frac{4}{7} * 1) = 0,29$$

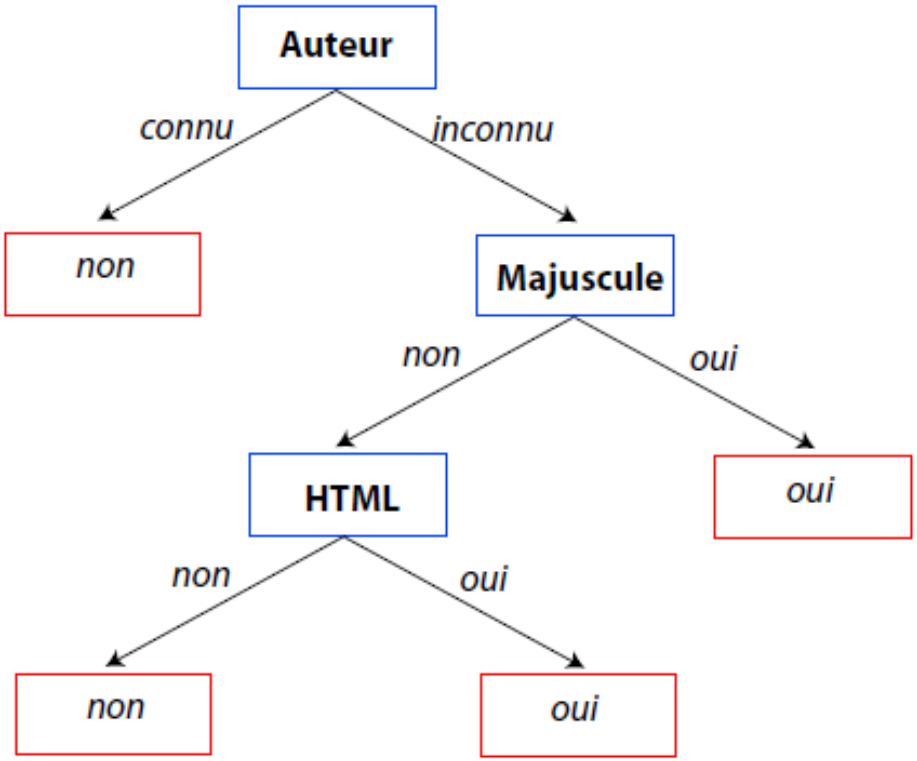
$$H(D[\textbf{Auteur} = inconnu][\textbf{Majuscule} = oui]) = -(\frac{4}{4}\log_2\frac{4}{4} + \frac{0}{4}\log_2\frac{0}{4}) = 0$$

$$H(D[\textbf{Auteur} = inconnu][\textbf{Majuscule} = non]) = -(\frac{1}{3}\log_2\frac{1}{3}+\frac{2}{3}\log_2\frac{2}{3}) = 0,92$$

$$G(\textbf{Majuscule}, D[\textbf{Auteur} = inconnu]) = 0,86 - (\frac{4}{7} * 0 + \frac{3}{7} * 0,92) = 0,47$$

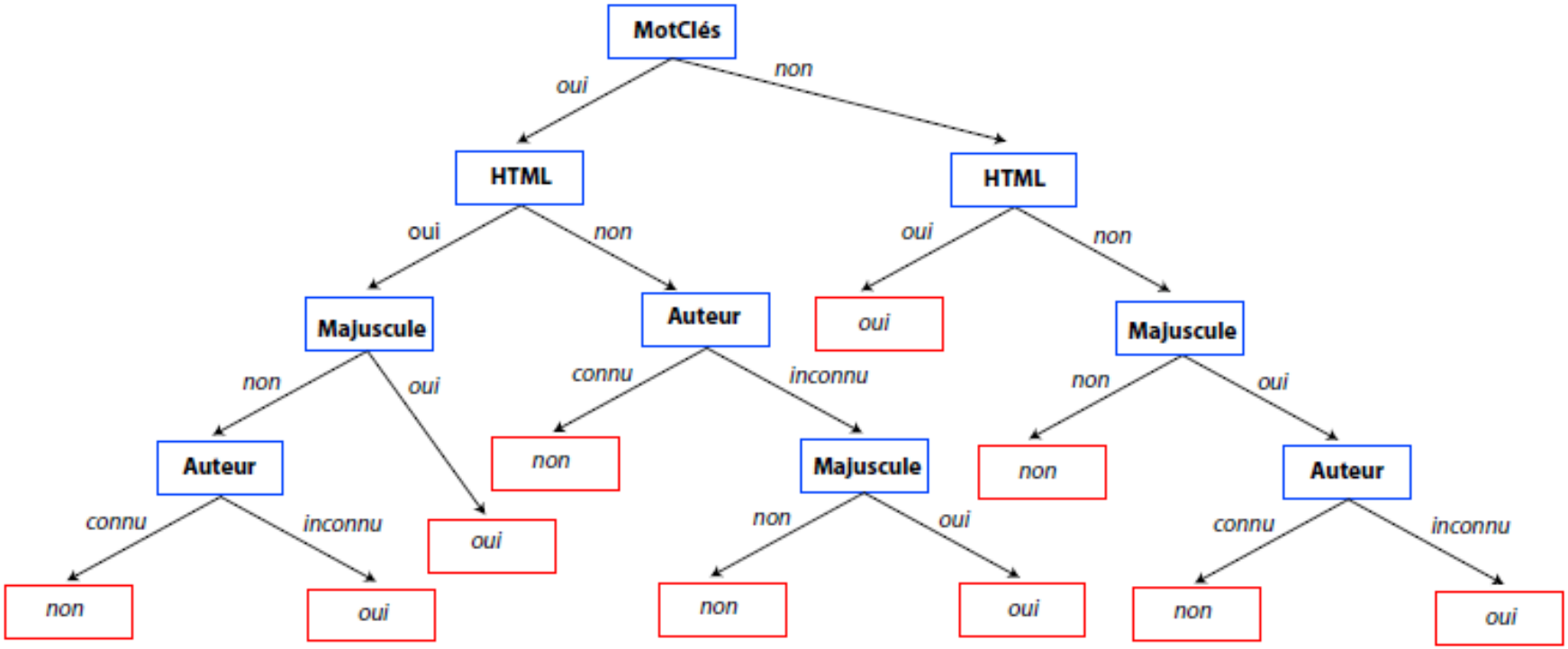
Arbre de décision : exemple

- Arbre final



Arbre de décision : exemple

- Mais pourquoi pas ?



Arbre de décision

- Si bruit ou trop peu d'exemples : tests peu performants : sur-apprentissage
- Technique d'élagage : précision -- , prédiction ++

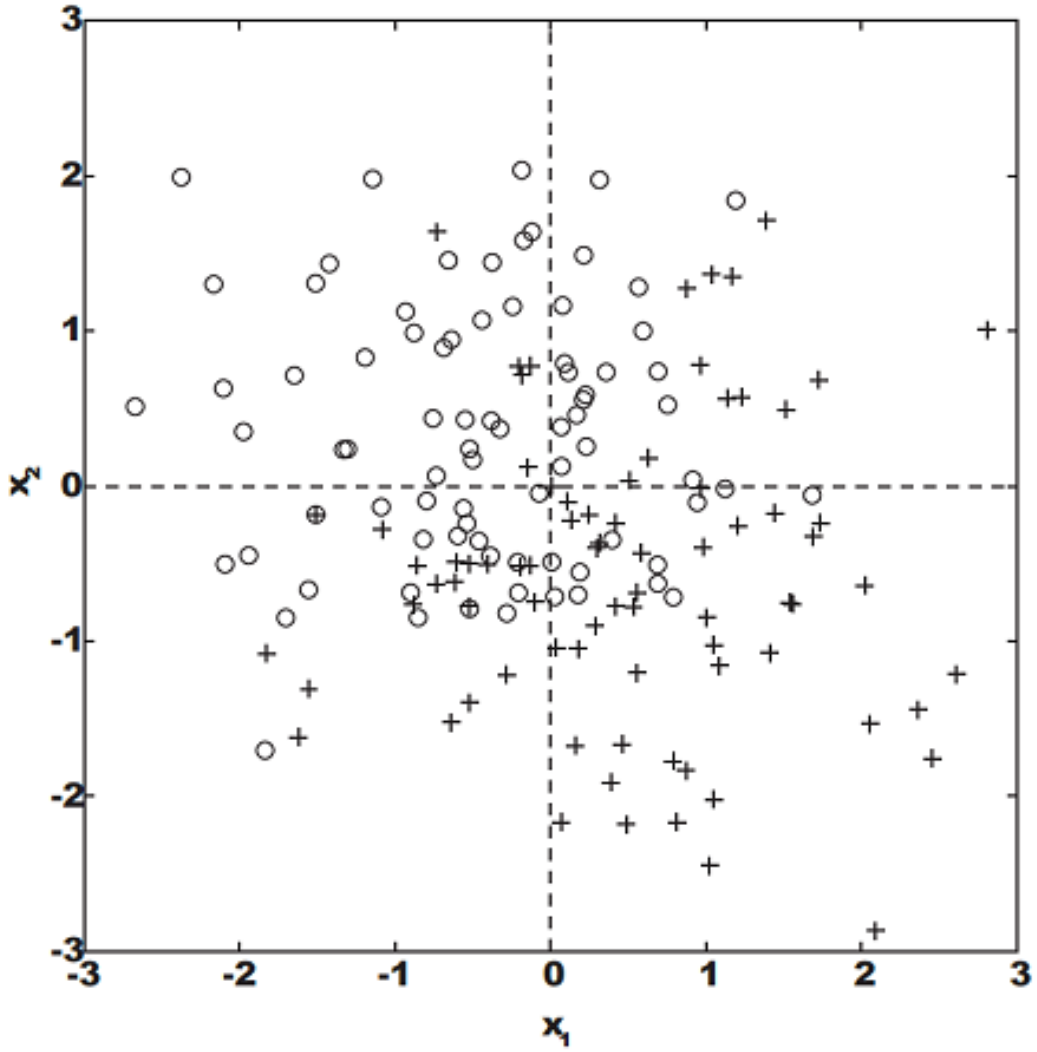
K-NN

- *K-Nearest Neighbors* : K-Plus Proches Voisins
- Données représentées par des nombres
- Prédire la classe de nouvelles données par leur proximité (distance) avec les données déjà étiquetées
- Distance classique : euclidienne

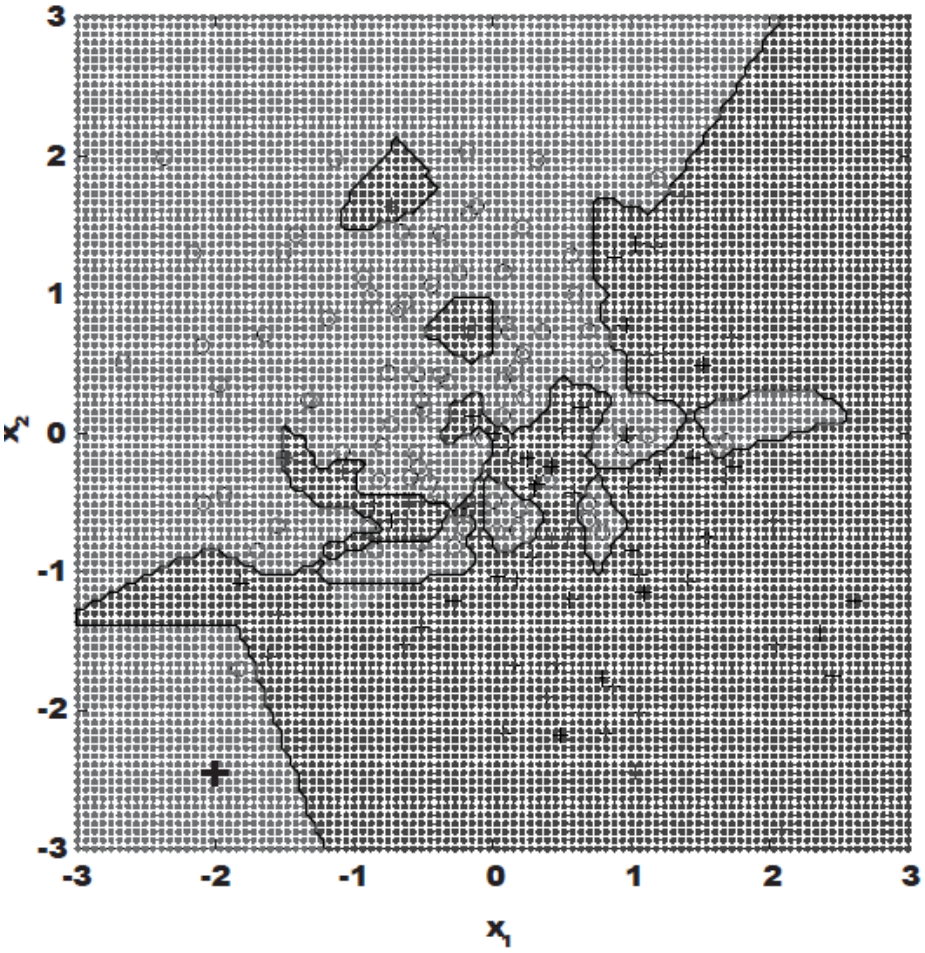
$$d(a,b) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2}$$

- Détermination de K => modèle

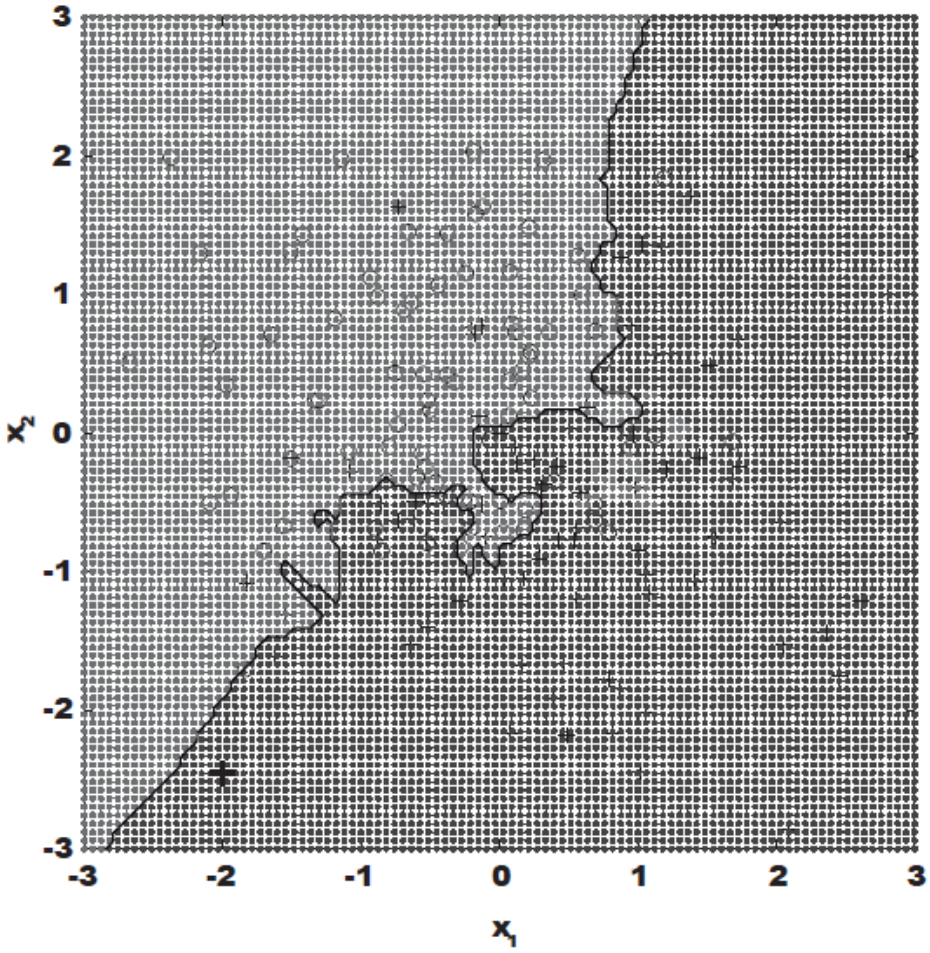
K-NN : exemple et analyse



K-NN : exemple et analyse

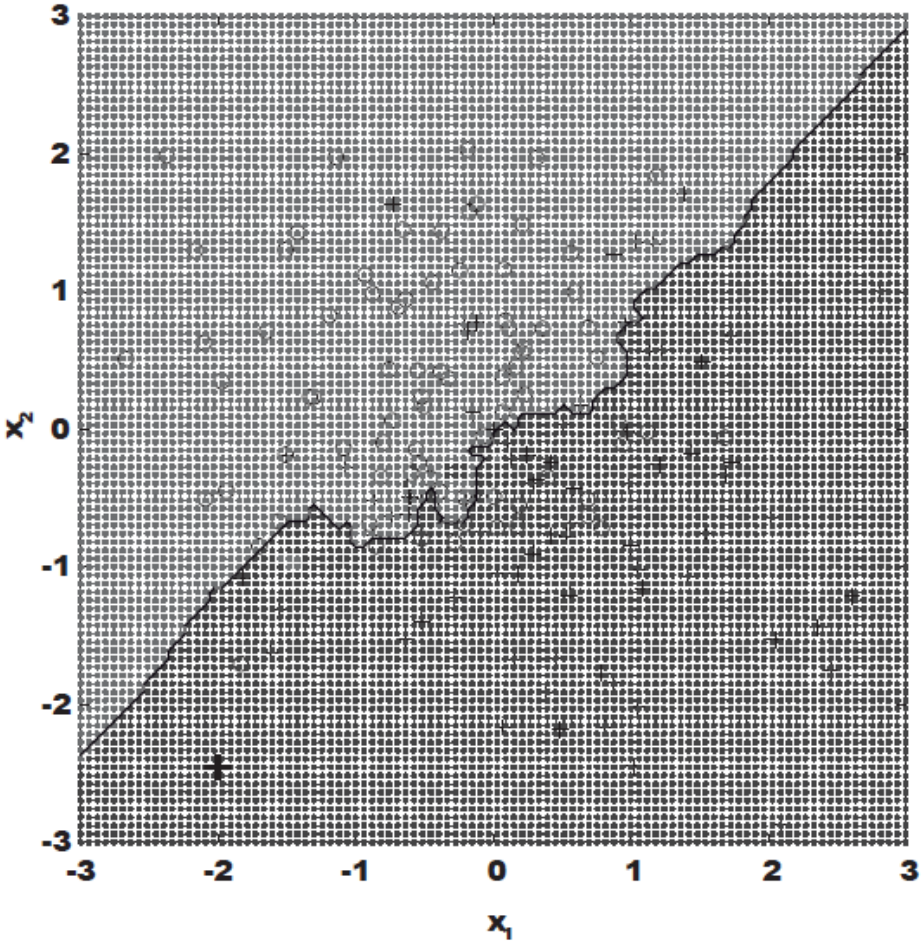


$k = 1$

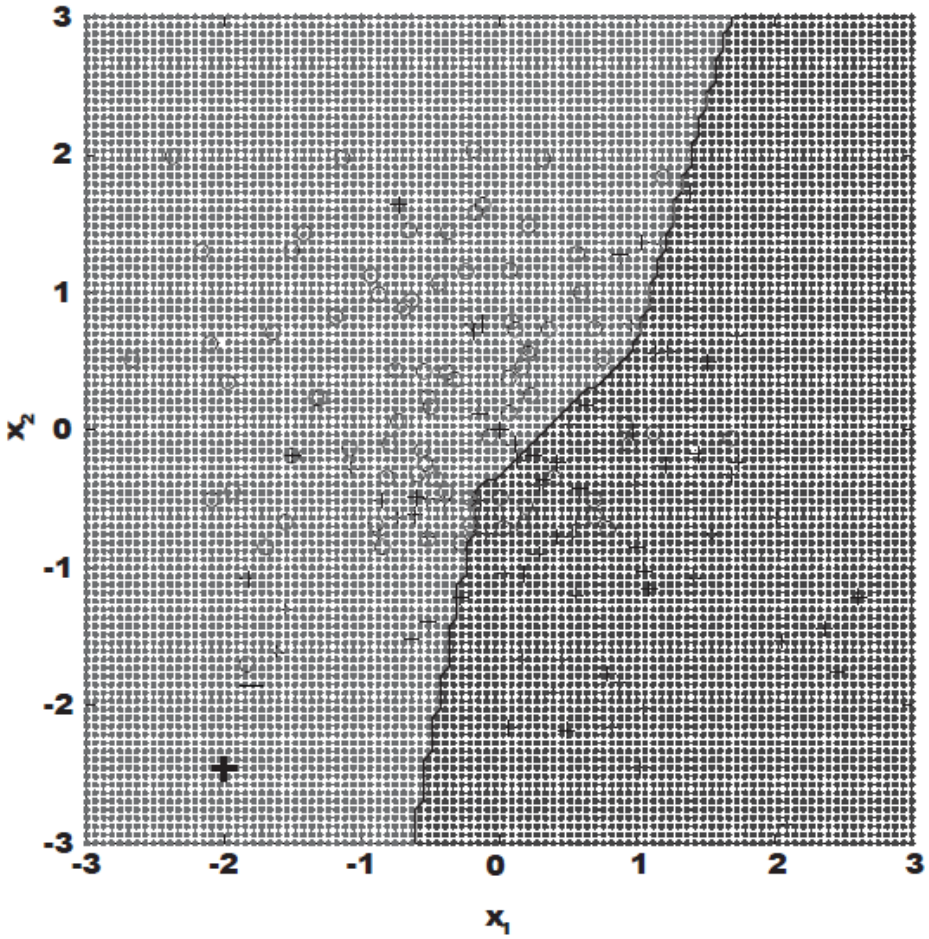


$k = 7$

K-NN : exemple et analyse

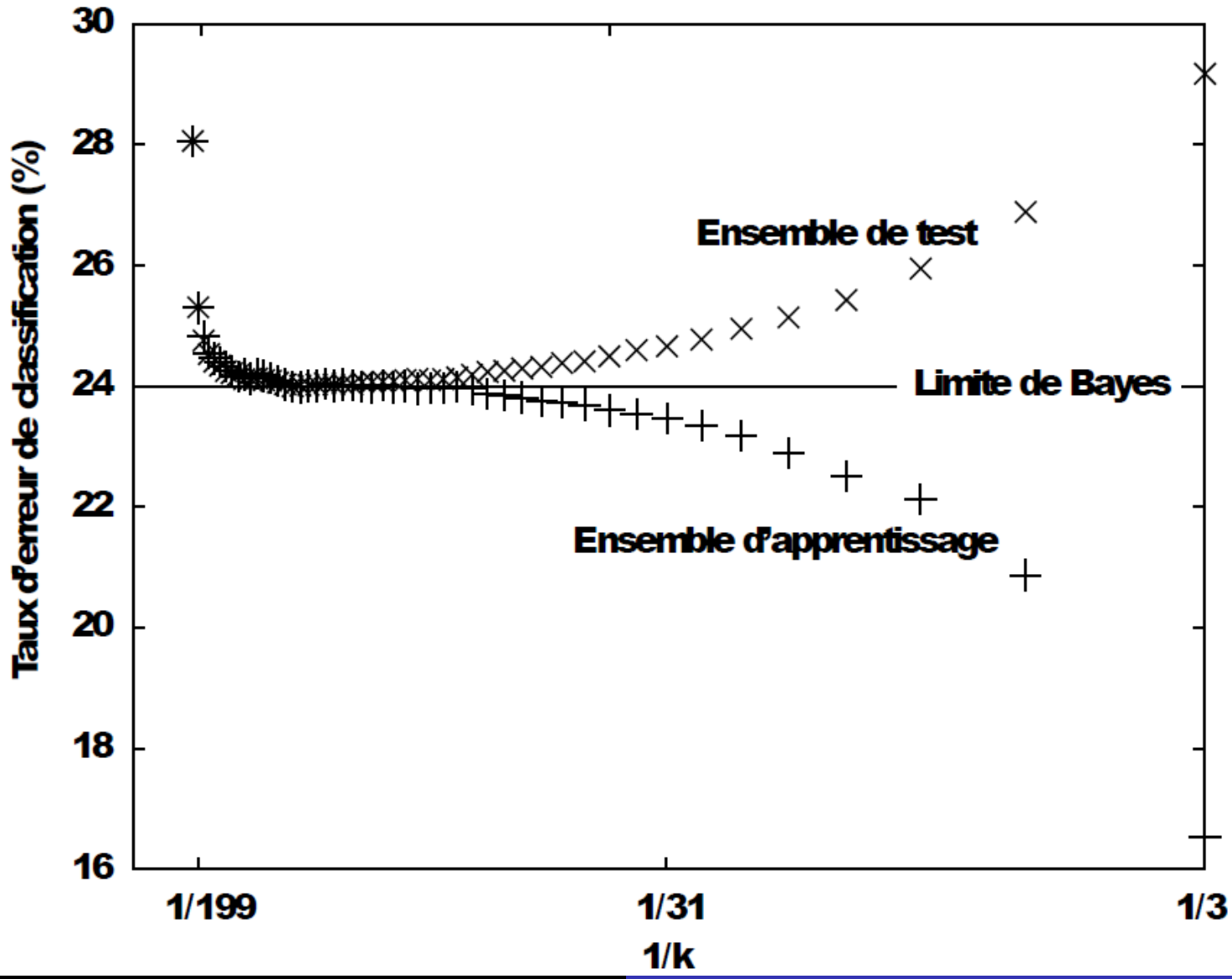


$k = 21$



$k = 159$

K-NN : exemple et analyse



K-NN

- Avantages : très simple, un seul paramètre
- Inconvénients : un seul paramètre (!), nécessite de calculer la distance avec tous les points (pas toujours vrai) : coûteux si beaucoup de données

Classification naïve bayésienne

- Thomas Bayes
- Probabiliste
- Lois de probabilités indépendantes entre attributs (« naïf ») : modèle à caractéristiques indépendantes
- Très satisfaisant cependant
- Nécessite peu de données d'entrainement

Classification naïve bayésienne : principe

- L'hypothèse : probabilité d'une donnée d'être d'une classe C sachant les attributs A_1, A_2, …, A_n
- Attention :
  - Fréquences : estimation de la probabilité d'occurrence d'un évènement
  - Bayésienne : estimation de la probabilité d'occurrence d'un évènement sachant qu'une hypothèse préliminaire est vérifiée (=connaissance)

Classification naïve bayésienne : principe

- Probabilité d'un évènement A : P(A)
- Entre 0 et 1
- P(A) = 1 : évènement certain
- P(A) = 0 : évènement impossible
- P(non A) = 1 – P(A)

Classification naïve bayésienne : principe

- P(A|B) = Probabilité que l'évènement A survienne si l'évènement B survient
- Théorème de Bayes

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

- Et

$$P(B \mid A) = \frac{P(A \cap B)}{P(A)}$$

- Donc

$$P(A \mid B) = \frac{P(B \mid A) * P(A)}{P(B)}$$

Classification naïve bayésienne : problématique

- Quelle est l'hypothèse le plus probable au vu de l'ensemble d'apprentissage ?
- Pour une instance donnée, au vu de l'ensemble d'apprentissage, quelle sera sa classe la plus probable ?

Classification naïve bayésienne : application

$$P(C_k \mid A_1,\ldots, A_n) = \frac{P(A_1,\ldots, A_n \mid C_k) * P(C_k)}{P(A_1,\ldots,A_n)}$$

- $P(C_k)$: proportion d'instances de la classe C_k
- $P(A_1,\ldots,A_n)$: proportion d'instances d'attributs $(A_1,\ldots, A_n)$
- $P(A_1,\ldots, A_n \mid C_k)$: nombre de fois où l'on rencontre $(A_1,\ldots, A_n)$ dans les instances de la classe C_k (vraisemblance)

Classification naïve bayésienne : application

- $C = (C_1,\ldots, C_k)$: ensemble de classes (à chaque classe, une probabilité)
- $(A_1,\ldots,A_n)$: ensemble d'attributs à valeurs discrètes
- Au final, prédire la classe revient à choisir la classe donc la probabilité est la plus forte parmi C

- Or, $P(A_1,\ldots,A_n)$ est constant (jeu d'apprentissage)
- On définit alors :
  - $h_{MAP} = \text{argmax}[C_k \in C]\ P(A_1,\ldots, A_n \mid C_k) * P(C_k)$
    Hypothèse Maximale A Posteriori
  - $h_{ML} = \text{argmax}[C_k \in C]\ P(A_1,\ldots, A_n \mid C_k)$
    Maximum de vraisemblance

Classification naïve bayésienne : application

$$P(C_k \mid A_1,\ldots, A_n) = \frac{P(A_1,\ldots, A_n \mid C_k) * P(C_k)}{P(A_1,\ldots,A_n)}$$

- Hypothèse naïve : indépendance d'occurrence des attributs décrivant l'exemple

$$P(A_1,\ldots, A_n \mid C_k) = \prod_{i=1}^{n} P(A_i \mid C_k)$$

- Au final, il faut estimer pour chaque classe

$$\prod_{i=1}^{n} P(A_i \mid C_k) * P(C_k)$$

- Et prendre la probabilité la plus grande

Classification naïve bayésienne : exemple

- Contrôle fiscal…
- Valeurs numériques et non numériques
- Faut-il effectuer un contrôle fiscal (Classes « true » ou « false ») ?
- Données :

Revenus (k€)	Impôts	Étudiant	Contrôle
< 30	< 20 %	True	False
30 – 50	< 20 %	False	True
30 – 50	< 20 %	True	False
30 – 50	> 20 %	False	False
> 50	< 20 %	False	True

- Prédire un contrôle :

35	**6 %**	**True**	**?**

Classification naïve bayésienne : exemple

- Classer X = (rev=35, imp=6%, etu=true)

- Il faut donc calculer

  P(cont=true|X)

  P(cont=false|X)

  => À vous !

Classification naïve bayésienne : exemple

- P(cont=true|X)
  = P(rev=30-50|true)*P(imp<20%|true)*P(etu=true|true)*P(true)
  = (2/3 * 1 * 1/3) * 3/5 = **0.13**

- P(cont=false|X)
  = P(rev=30-50|false)*P(imp<20%|false)*P(etu=true|false)*P(false)
  = (1/2 * 1/2 * 1/2) * 2/5 = 0.05

Classification naïve bayésienne : conclusion

- Méthode très répandue
- Simple
- Robuste
- Peut être couteux si beaucoup de données
- Hypothèse naïve souvent fausse mais cela marche tout de même très bien !
- Flexible : possibilités d'adapter le modèle

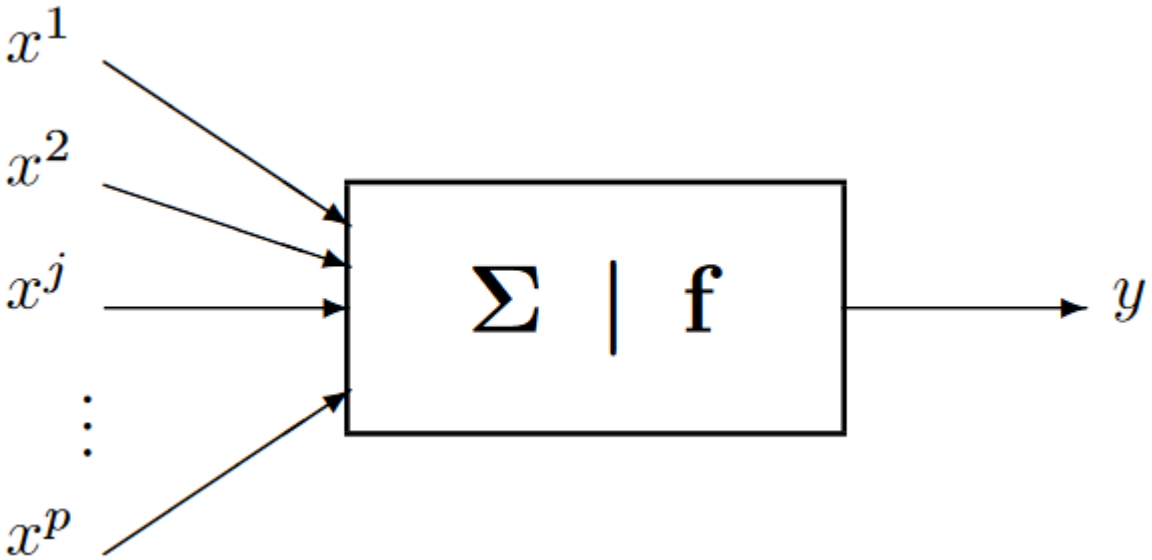
Réseau de neurones

- Supervisé ou non
- Système qui imite les neurones vivants
- 1 neurone = 1 bloc de code
- *n* entrées et *m* sorties : graphe
- Connexion entre neurones = synapse : poids synaptique (pertinence de la liaison ⇔ effet mémoire)
- Affinage des connexions lors de l'apprentissage (implication ++ => poids ++)

Réseau de neurones : la biologie

- Environ 10^{11} neurones
- Environ 10^{15} connexions
- Transmission de l'information : environ 100 m/s
- Neurones et développement

Réseau de neurones

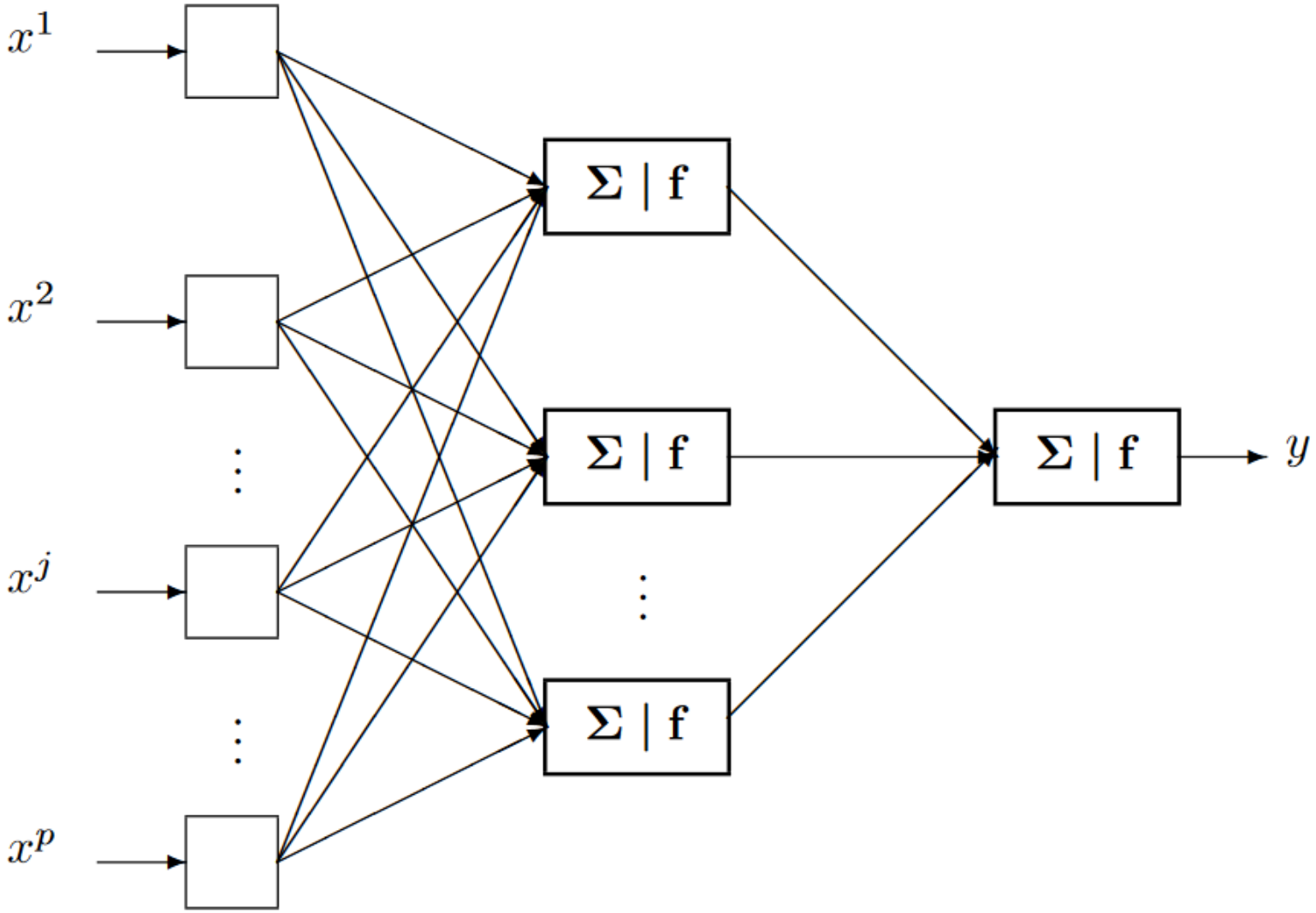


- Neurones d'entrée : couche d'entrée
- Neurones de sortie : couche de sortie
- Entre deux : neurones cachés

Réseau de neurones

- Types de fonctions d'activation :
  - Linéaire
  - Sigmoïde
  - Seuil
  - Radiale
  - Stochastique (probabiliste)
  - …

Réseau de neurones



Réseau de neurones : but

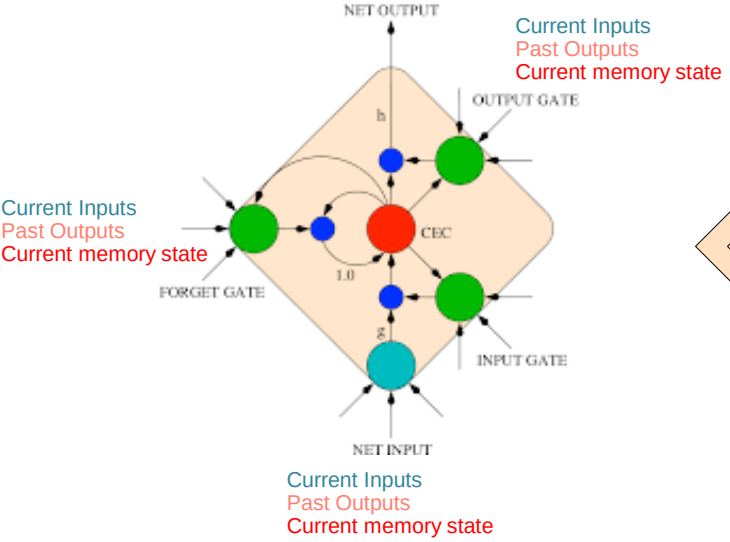
- Obtenir une configuration optimale = opérationnelle (stabilisation des poids synaptiques et/ou de la topologie)
- Réutilisation sur des situations nouvelles

Réseau de neurones : applications

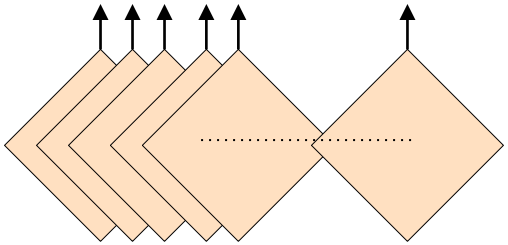
- Réseaux de neurones et compréhension du fonctionnement du cerveau : reconnaissance de la parole, d'objets, de visages, …
- Science cognitive : réponse de la machine au langage naturel
- Linguistique statistique, technologie du langage (traduction automatique)
- Traitement des *Big Data*
- Jeux

Exemple réseau de neurones

LSTM Cell
Long Short Term Memory

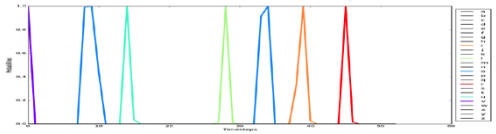


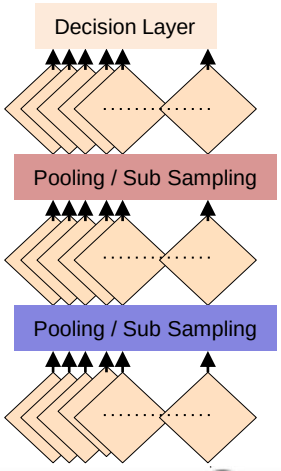
LSTM Layer



Staked / Deep reccurent NN

v o u l o i r

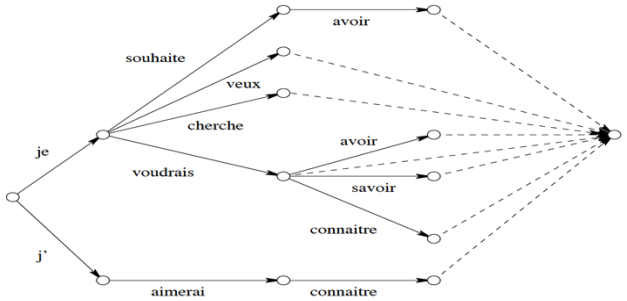


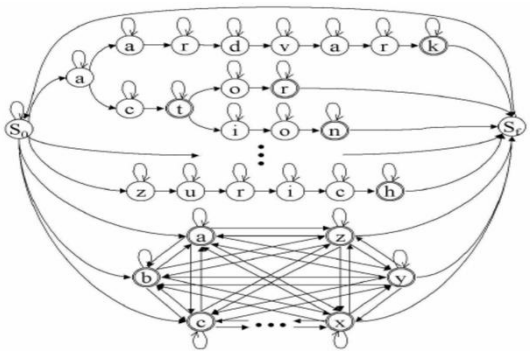


C. Chatelain , S. Thomas, Y. Kessentini , T. Paquet , L. Heutte, A Deep HMM model for multiple keywords spotting in handwritten documents, Pattern Analysis and Applications, vol. 18, n° 4, pp. 1003-1015, 2015.
Bruno Stuner, Clément Chatelain, Thierry Paquet, Cascading BLSTM Networks For Handwritten Word Recognition, accepted ICPR 2016, Cancun.

Exemple réseau de neurones

Word sequences follow some statistical rules.
Statistical language model (n-gram)
Weighted finite states automata WFSA

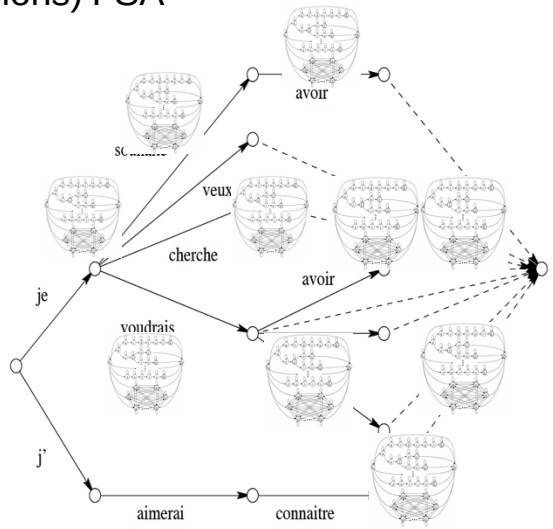




Characters sequences follow some other syntactical rules (lexicon) FSA

Encoding regular expressions as well (alphanumeric expressions) FSA

WFSA composition provides the global model / automata of the admissible solutions



W. Swaileh, T. Paquet, Un modèle syllabique du Français et de l'Anglais pour la reconaissance de l'écriture, revue Document Numérique, 2016.

Réseau de neurones : problèmes

- Minimum local vs. minimum global
- Nouvelles méthodes
- Solutions techniques : très grand nombre de neurones & synapses
- Possibilité d'élaguer
- Rétropropagation

Réseau de neurones

- Surface de décision linéaire ou non linéaire
- Peu coûteux
- Nécessite assez de données
- Souvent complexe à régler et à mettre en œuvre

SVM

- Machine à Vecteurs de Support
- = Séparateur à Vaste Marge
- *Support Vector Machine*
- Vapnik (60’s => 1998)
- Problèmes de discrimination ou régression
- Chercher l’hyperplan séparateur optimal puis regarder de quel côté se trouve les instances à prédire

SVM : principe

- Distance maximale entre la frontière de séparation et les échantillons les plus proches (vecteurs de supports)
- La frontière est celle qui maximise cette marge
- Problème : comment permettre la définition d'une frontière ?
- Étape de transformation non-linéaire puis trouver l'hyperplan le plus discriminant

SVM : transformation non-linéaire

- Projection des données d'apprentissage dans un espace où elles sont linéairement séparables (espace à grande dimension par une transformation basée sur un noyau)

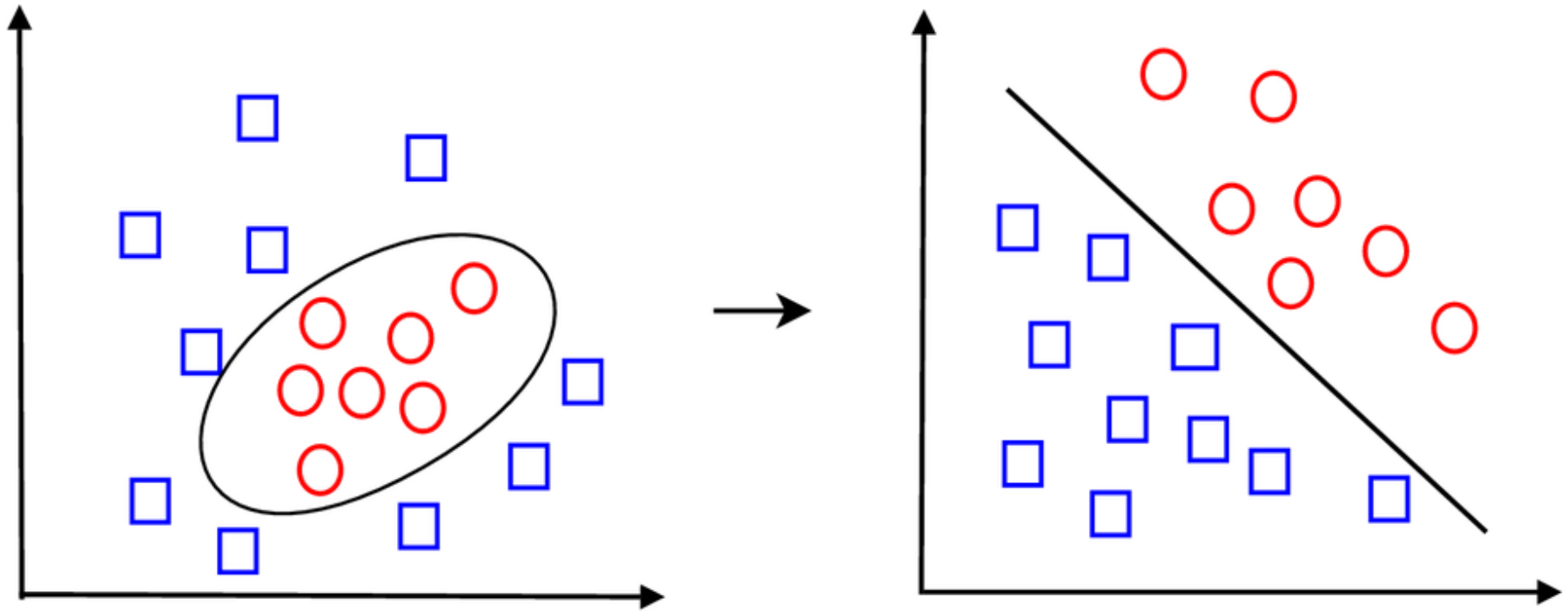


Illustration : Christian Raymond

SVM : hyperplan optimal

- Séparation correcte des données d'apprentissage et maximisation de la marge : optimisation quadratique

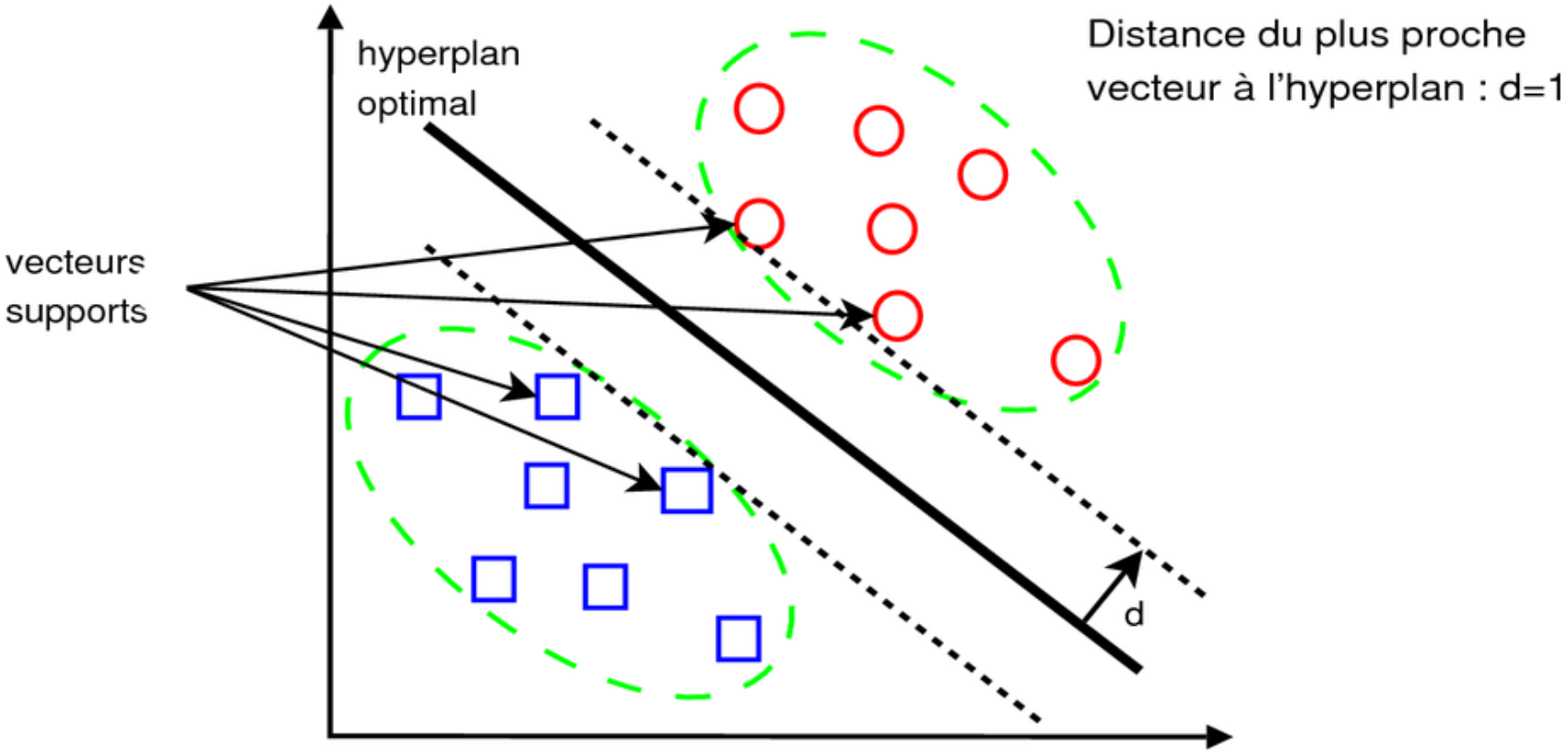


Illustration : Christian Raymond

SVM

- Bonnes performances
- Grands volumes de données
- Attributs indépendants ou non
- Classe binaire seulement mais adaptations possibles (1 classe vs toutes les autres ou tous les cas 1 classes vs 1 classes via probabilités)
- Modèles dépendants de l'hyperplan séparateur (noyaux) et de son potentiel à séparer les données

Analyse discriminante linéaire

- Fisher (1936)
- Variante des probabilités conditionnelles de Bayes et SVM
- Basée sur les matrices de co-variance : regroupement des données par « densité »

- Chaque classe supposée Gaussienne
- Un paramètre : le seuil de discrimination

Apprentissage non supervisé

Apprentissage non supervisé : principe

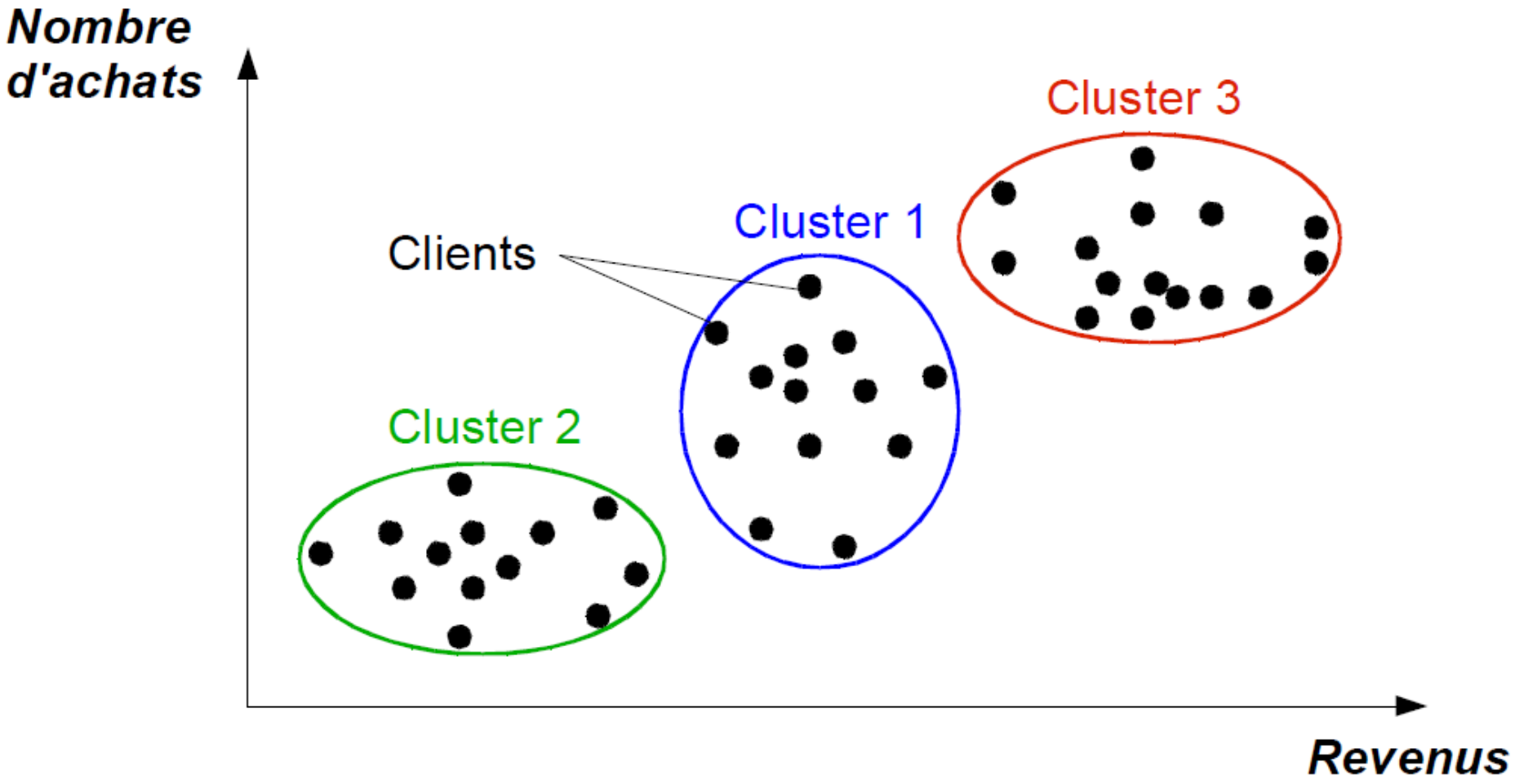
- *Clustering*
- Données non-étiquetées (la « vraie vie » ?)
- Données brutes souvent
- Objectif : trouver des points communs entre ces données = description et/ou structuration
- Recherche des groupes (*clusters*) dans un ensemble de données :

  - avec la plus grande similarité possible intra-groupe
  - et la plus grande dissimilarité possible inter-groupe
- Applications diverses : séismologie, santé, commerce, etc.

Exemple

- On dispose de données sur des clients (âge, nombre d'enfants, revenus, nombre d'achats, etc.)
- On regroupe en clusters les clients ayant des caractéristiques communes
- Pour chaque cluster, on définit une offre commerciale adressée aux clients de ce cluster

Exemple


Nombre d'achats
Cluster 3
Cluster 1
Clients
Cluster 2
Revenus

Exemple

- Objets « suffisamment similaires » regroupés en clusters
- Définition du seuil de similarité difficile
- Évaluation des clusters
  - Distance entre objets à l'intérieur du cluster
  - Distance avec les objets des autres clusters
- Les données bruitées et les déviations nuisent à la qualité du clustering

Proximité et distance

Notion de proximité

- Mesure de dissimilarité DM : plus la mesure est faible, plus les points sont similaires (distance)
- Mesure de similarité SM : plus la mesure est grande, plus les points sont similaires

Comment mesurer la distance entre 2 points d(x1; x2) ?

- Distance de Minkowski :

$$d(i,j) = \sqrt[q]{(|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + ... + |x_{ip} - x_{jp}|^q)}$$

  - Distance euclidienne : $d^2(x1; x2) = \Sigma_i (x1_i - x2_i)^2$ (norme L_2)
  - Distance de Manhattan : $d(x1; x2) = \Sigma_i |x1_i - x2_i|$ (norme L_1)
- Distance de Sebestyen : $d^2(x1; x2) = (x1 - x2)W\ ^t(x1 - x2)$ avec W= matrice diagonale
- Distance de Mahalanobis :$d^2(x1; x2) = (x1 - x2)C\ ^t(x1 - x2)$, avec C=covariance

Distance : variables binaires

- Une table de contingence pour données binaires

		Objet *j* = 1	Objet *j* = 0	sum
Objet *i*	1	a	b	$a+b$
	0	c	d	$c+d$
	sum	$a+c$	$b+d$	p

a = nombre de positions où i à 1 et j à 1

- Exemple o_i =(1,1,0,1,0) et o_j =(1,0,0,0,1)

  a=1, b=2, c=1, d=1

Distance : variables binaires

- Coefficient d'appariement (*matching*) simple (invariant pour variables symétriques):

$$d(i,j) = \frac{b+c}{a+b+c+d}$$

- Exemple o_i =(1,1,0,1,0) et o_j =(1,0,0,0,1)

  $d(o_i , o_j)$=3/5

- Coefficient de Jaccard

  $d(o_i , o_j)$=3/4

$$d(i,j) = \frac{b+c}{a+b+c}$$

Distance : variables binaires

- Variable symétrique: exemple le sexe d'une personne => coder masculin par 1 et féminin par 0 c'est pareil que le codage inverse

- Variable asymétrique: exemple test HIV. Le test peut être positif ou négatif (0 ou 1) mais il y a une valeur qui sera plus présente que l'autre. Généralement, on code par 1 la modalité la moins fréquente

  2 personnes ayant la valeur 1 pour le test sont *plus similaires* que 2 personnes ayant 0 pour le test

Distance : variables binaires, exemple

- Exemple

Nom	Sexe	Fièvre	Toux	Test-1	Test-2	Test-3	Test-4
Rick	M	Y	N	P	N	N	N
Maggie	F	Y	N	P	N	P	N
Glenn	M	Y	P	N	N	N	N

- Sexe est un attribut symétrique
- Les autres attributs sont asymétriques
- La distance n'est mesurée que sur les asymétriques

Distance : variables binaires, exemple

Nom	Sexe	Fièvre	Toux	Test-1	Test-2	Test-3	Test-4
Rick	M	Y	N	P	N	N	N
Maggie	F	Y	N	P	N	P	N
Glenn	M	Y	P	N	N	N	N

$$d(\text{Rick, Maggie}) = \frac{0 + 1}{2 + 0 + 1} = 0.33$$

$$d(\text{Rick, Glenn}) = \frac{1 + 1}{1 + 1 + 1} = 0.67$$

$$d(\text{Glen, Maggie}) = \frac{1 + 2}{1 + 1 + 2} = 0.75$$

Distance : variables nominales

- Une généralisation des variables binaires, ex: rouge, vert et bleu
- Méthode 1: « Matching simple », m: nb d'appariements, p: nb total de variables

$$d(i,j) = \frac{p-m}{p}$$

- Méthode 2: utiliser un grand nombre de variables binaires. Créer une variable binaire pour chaque modalité (ex: variable rouge qui prend les valeurs vrai ou faux)

Évaluation

- Métrique pour la similarité: la similarité est exprimée par le biais d'une mesure de distance
- Une autre fonction est utilisée pour la mesure de la qualité
- Les définitions de distance sont très différentes que les variables soient des intervalles (continues), catégories, booléennes ou ordinales
- En pratique, on utilise souvent une pondération des variables

Principaux algorithmes/outils

- Regroupement hiérarchique (agglomération ou division)
- K-moyennes (partitionnement)
- Méthodes par densité
- Méthodes probabilistes (modèles)
- Réseaux de neurones

Regroupement hiérarchique : principe

- Regroupement Hiérarchique Ascendant (*bottom-up*) : chaque point ou cluster est progressivement « absorbé » par le cluster le plus proche.
- Méthode hiérarchique descendante (*top-down*) : départ avec un cluster contenant tous les objets puis séparation par dissemblances

=> + Conditions d'arrêts

RHA : algorithme

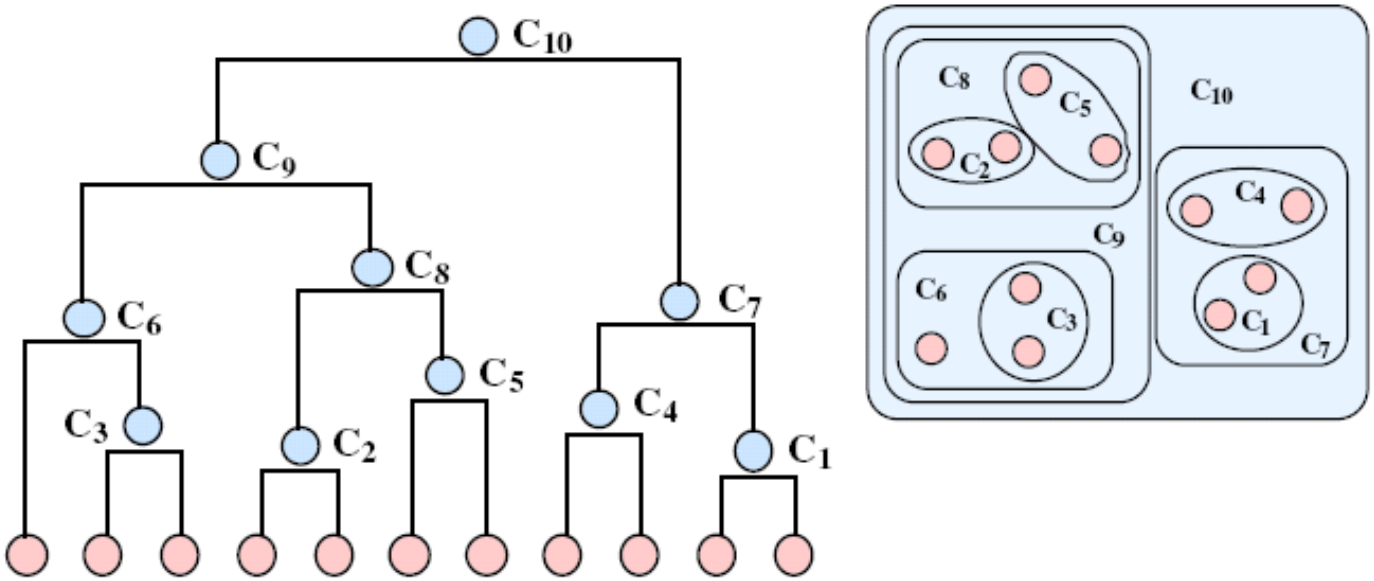
- Initialisation :
  - Chaque individu est placé dans son propre cluster
  - Calcul de la matrice de ressemblance M entre chaque couple de clusters (par ex des points sur un plan)
- Répéter
  - Sélection dans M des deux clusters les plus proches Ci et Cj
  - Fusion de Ci et Cj par un cluster Cg plus général
  - Mise à jour de M en calculant la ressemblance entre Cg et les clusters existants

Jusqu'à la fusion des 2 derniers clusters

RHA : différentes techniques

- Plusieurs techniques (variantes de RHA)
  - plus proche voisin (ppv) : min(d(i;j); i$\in$ C1; j $\in$ C2)
  - distance maximum : max(d(i;j); i $\in$ C1; j $\in$ C2)
  - distance moyenne : ($\Sigma_{i;j}$ d(i;j))/(n1*n2)
  - distance des centres de gravité : d(b1;b2)
  - distance de Ward : sqrt(n1n2/(n1+n2))*d(b1;b2)

RHA : dendrogramme



- Dendrogramme = représentation des fusions successives
- Hauteur d'un cluster dans le dendrogramme = similarité entre les 2 clusters avant fusion (sauf exceptions avec certaines mesures de similarité...)

K-moyennes

- *K-means*
- Construire une partition à **K** clusters d'une base **D** de **n** objets
- Chaque cluster est représenté par son centre (barycentre)

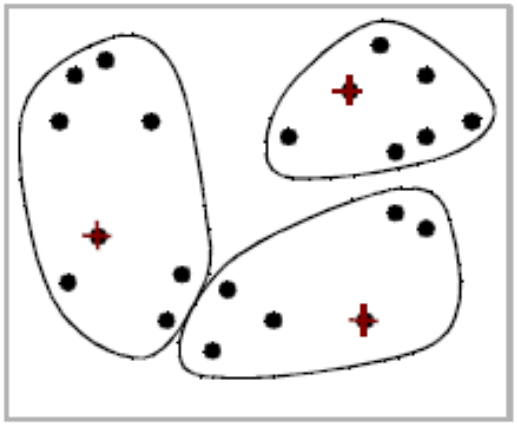
K-moyennes : algorithme

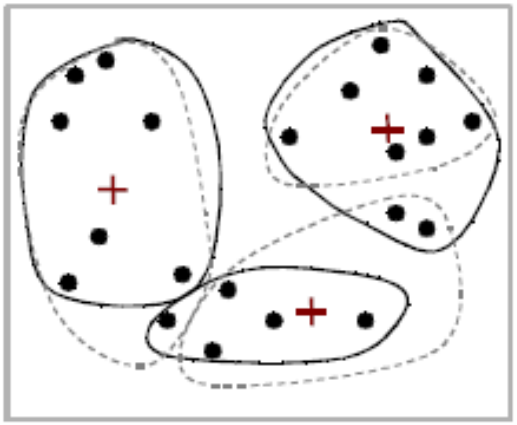
1. Sélectionner aléatoirement K objets comme centroïdes des clusters initiaux
2. *Répéter*
    3. Assigner chaque objet x au cluster dont le centroïde est le plus proche de x
    4. *Pour chaque cluster C*
        5. Recalculer son centroïde comme moyenne arithmétique (barycentre) des objets de C

Jusqu'à ce que les clusters soient stables

K-moyennes : principe

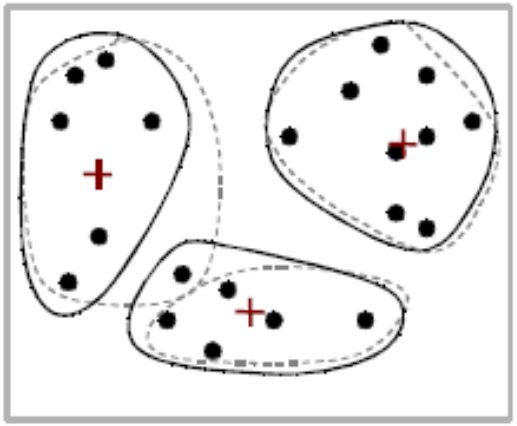
Choix aléatoire de *k* objets centres initiaux et calcul des clusters

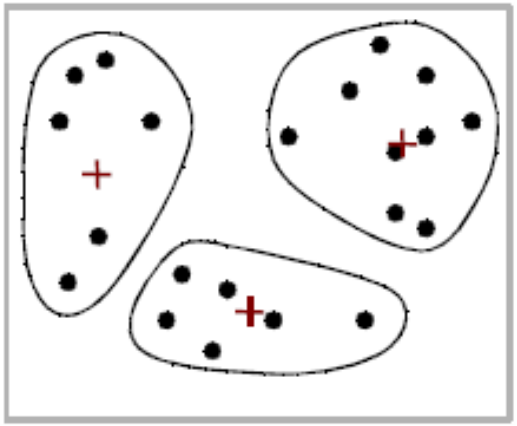




Calcul des centres des clusters et mise-à-jour des clusters

Mise-à-jour des centres des clusters et mise-à-jour des clusters





Arrêt lorsque les clusters sont stables (critère stable)

K-moyennes

Avantages

- Efficace : complexité en O(*knt)*

– *k : nbr clusters, n : nbr objets, t : nbr itérations*

– En général *k << t << n*

- Interprétation aisée des résultats : centroïde caractérise le cluster

Inconvénients

- Nécessité de fixer *k* (empirique, modèle, … ?)
- Sensible aux exceptions et aux données bruitées
- Variables numériques seules

K-moyennes : variantes

Variante: K-médoïdes

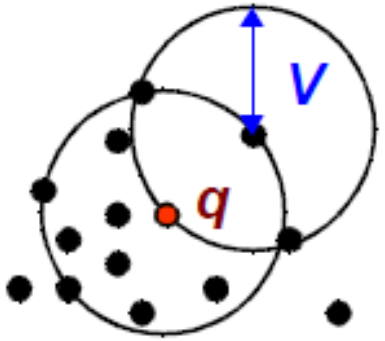
- Utilise un objet « central » au lieu du barycentre (médiane vs. moyenne)
- Moins sensible aux données bruitées
- Chaque cluster est représenté par un objet réel : son médoïde

Nuées dynamiques

- Extension des K-médoïdes
- Chaque cluster est représenté par un ensemble d'objets centraux
- Plus stable et moins dépendant de K
- Plus coûteux (temps de calculs)

Méthodes basées sur la densité

- Objets : points dans l'espace des données, Clusters : régions denses séparées par des régions peu denses
- Paramètres :
  - V = distance maximale de voisinage
  - N = nombre minimal d'objets dans le voisinage d'un objet cœur (q)
  - Centre d'une zone dense

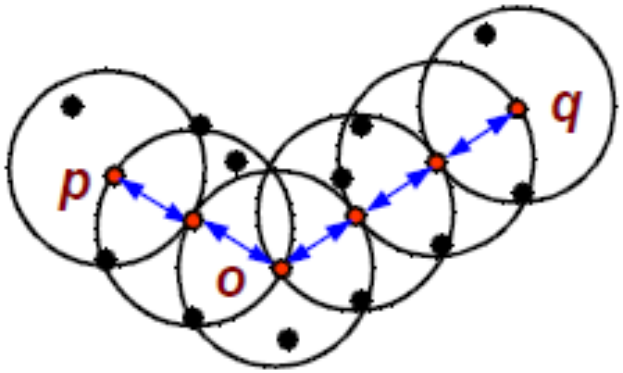


$N = 5$

Méthodes basées sur la densité : algorithme

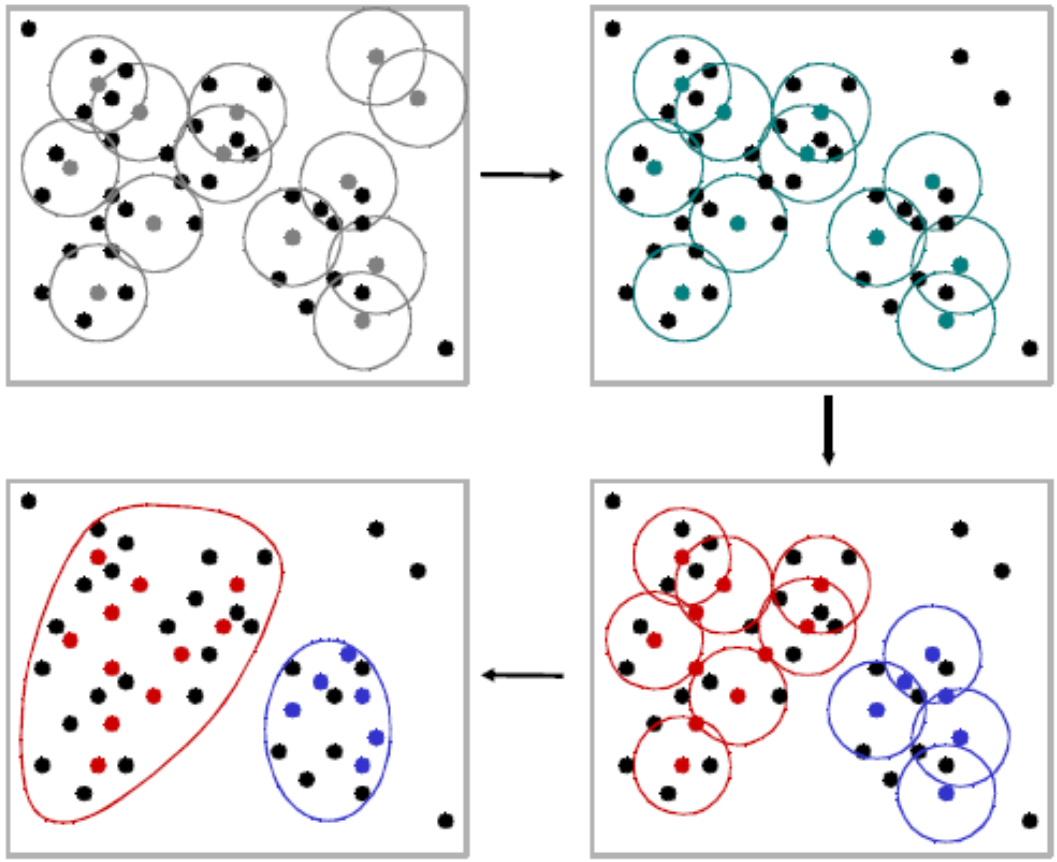
1. Choisir aléatoirement un ensemble d'objets et calculer leur voisinage
2. Identifier les objets cœurs
3. Construire un cluster pour chaque objet cœur
4. Fusionner les clusters d'objets cœurs mutuellement atteignables

Cluster de points mutuellement atteignables



Méthodes basées sur la densité : exemple

Choix aléatoire de points de départ et calcul de la taille de leur voisinage



Détermination des *objets cœurs*

Fusion des clusters dont les objets cœurs sont mutuellement atteignables

On obtient des clusters de tailles et de formes différentes

Méthodes basées sur la densité

Avantages

- Robustes aux données bruitées (points isolés)
- Clusters non convexes (formes quelconques)

Inconvénients

- Peu adaptées aux attributs symboliques
- Complexité $O(n^2)$

Autre outil

- Analyse en Composante Principale (ACP) : « dé-corréler » des variables en « composantes principales » en perdant le moins d'information possible

Autres approches

Apprentissage semi-supervisé

- Données étiquetées et non étiquetées
- Utile pour la prédiction (classement) et/ou le clustering
- Étiqueter une partie des données => moins coûteux

Apprentissage profond

- *Deep Learning*
- Depuis années ’80 mais vraiment utilisé depuis 2012
- Série de modules chaînés (couches)
- Chaque module est entrainable et paramétrable
- Calcul d’un gradient par rétro-propagation
- Équivalent à un réseau neuronal multicouches

Apprentissage profond : exemple

- Graphe de flot : calcul décomposé par nœuds
- Le graphe de flot de l'expression $\sin(a^2 + b/a)$ peut être représenté par un graphe avec :
  - deux nœuds d'entrée a et b
  - un nœud pour la division, b/a, dont les entrées (les enfants) sont a et b
  - un nœud pour le carré, prenant seulement a comme entrée
  - un nœud pour l'addition, dont la valeur serait $a^2 + b/a$, prenant comme entrées les nœuds a^2 et b/a
  - un nœud de sortie calculant le sinus, dont la seule entrée est le nœud d'addition
- Profondeur = chemin le plus long depuis l'entrée jusqu'à la sortie

Apprentissage par renforcement

- Observation : action <-> réaction (effet)
- « Récompense » ou « punition »
- Particulièrement adapté pour les réseaux de neurones : rétro-propagation, …
- Applications : jeux, robots/automates, …

Apprentissage par renforcement

1. L'agent observe un état d'entrée
2. Une action est déterminée par une fonction de prise de décision (politique=*policy*)
3. L'action est effectuée
4. L'agent reçoit un résultat en fonction de son environnement
5. Informations sur le résultat donné pour cette état (récompense ou punition) ou l'action est enregistrée

Apprentissage par renforcement : exemple

- Les ascenseurs de Robert Crites et Andrew Barto (1996)
- 4 ascenseurs dans une tour de 10 étages
- Deux boutons à l'extérieur (sauf RDC et $9^{ème}$)
- En tous, environ 10^{22} états possibles (complexité bien trop élevée, surtout en 1996)
- Utilisation d'un réseau de neurones : 47 neurones d'entrée, 20 neurones cachés, 2 neurones de sortie

Apprentissage par renforcement : exemple

- Entrées : boutons appuyés, occupations des ascenseurs et temps écoulé
- Utilisation ici d'une punition (et non pas d'une récompense) : rétro-propagation d'une erreur quand le temps d'attente est trop grand
- Résultats : réduction significative du temps d'attente ; 60 000 heures de simulation (calculs sur 4 jours sur une station puissante de l'époque)

Synthèse

- Approches complémentaires
- Nécessité de comprendre/prétraiter les données
- Choix en fonction d'un problème