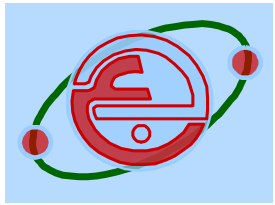
Université Badji Mokhtar - Annaba
Département d'Informatique



جامعة باجي مختار – عنابـــــة

Introduction à l'apprentissage automatique

Cours d'Apprentissage Automatique
Master 1 STIC, 2014-2015

Présenté par
Pr Souici – Meslati L.

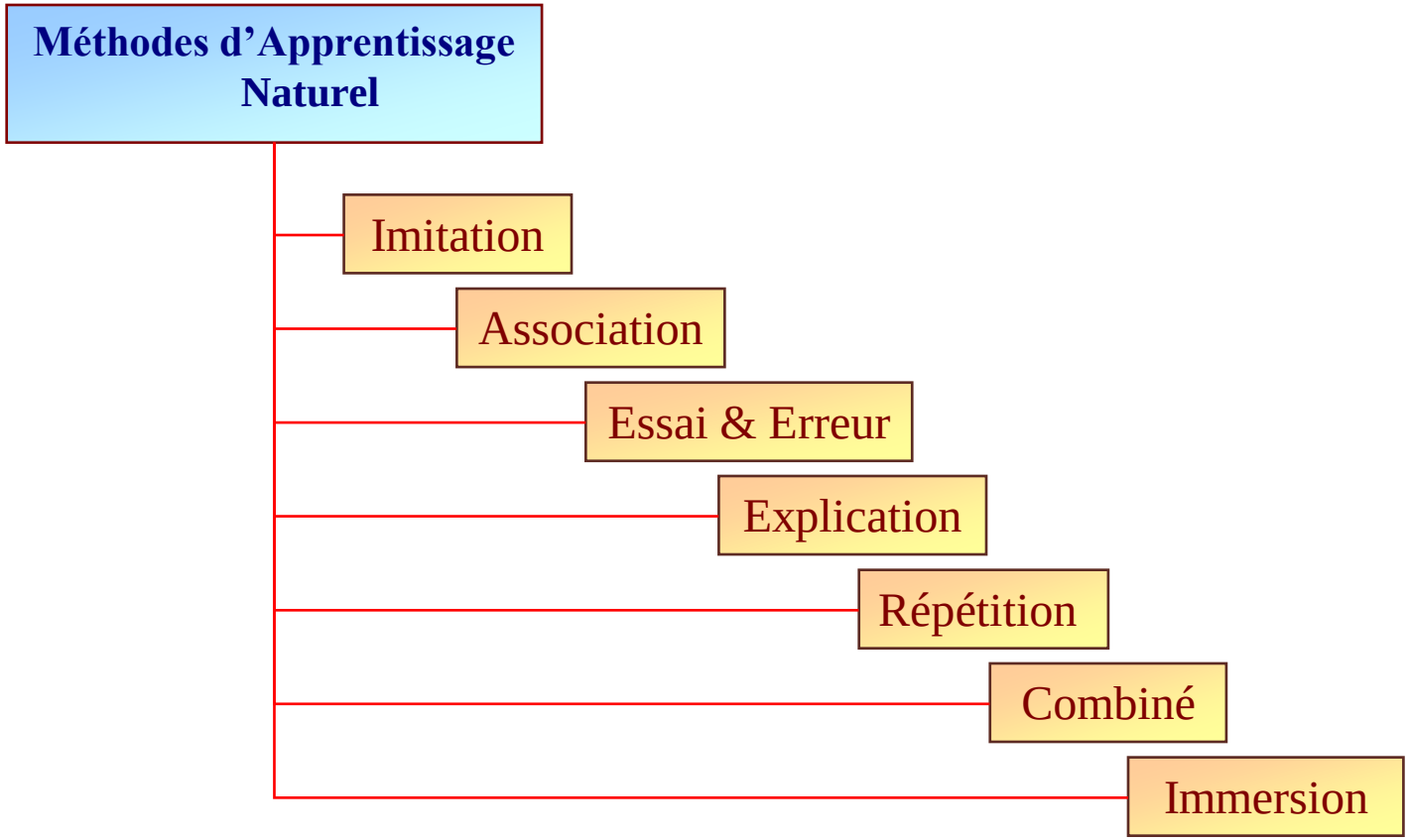
Plan

1. Apprentissage naturel
2. Définition de l'apprentissage automatique
3. Exemples d'apprentissage
4. Historique de l'apprentissage automatique
5. Domaines connexes
6. Applications de l'apprentissage automatique
7. Paramètres d'un problème d'apprentissage automatique
8. Classifications des méthodes d'apprentissage automatique
9. Directions de recherche actuelles et futures
10. Sources bibliographiques et URL

Apprentissage Naturel Définition

- Chez tous les êtres vivants, à des degrés divers et à l'aide de mécanismes très variables (génétiques, chimiques ou culturels), on observe une aptitude à acquérir de nouveaux comportements à partir des interactions avec leur environnement. L'apprentissage est une modification durable des potentialités de comportement résultant d'une interaction répétée avec l'environnement.
- L'apprentissage humain est l'acquisition de savoir-faire, c'est-à-dire le processus d'acquisition de pratiques, de connaissances, compétences, d'attitudes ou de valeurs culturelles, par l'observation, l'imitation, l'essai, la répétition…
- Pour la psychologie comportementale, l'apprentissage se fait par association, il est vu comme la mise en relation entre un événement provoqué par l'extérieur (stimulus) et une réaction adéquate du sujet, qui cause un changement de comportement qui est persistant, mesurable, et spécifique ou permet à l'individu de formuler une nouvelle construction mentale ou réviser une construction mentale préalable.

Apprentissage naturel
Méthodes (1)



Apprentissage naturel
Méthodes (2)

- **Apprentissage par imitation**

  Le plus courant: il suppose de la part de l'enfant la valorisation d'un modèle et la volonté de le posséder, de le prendre. C'est par l'imitation que se font tous les apprentissages "spontanés" de la petite enfance: parole, gestes, mimiques, etc.., ainsi que ceux de la dimension esthétique des activités: ton, grâce, style, manière, etc. Le rôle du pédagogue est de montrer l'exemple ou de proposer des modèles, sans devoir faire appel à la rationalité expérimentale et à sa systématisation. Abandonné par la pédagogie scolaire, il reste utilisé pour l'enseignement des arts: qu'il s'agisse de l'équitation, du violon, de la cuisine, du dessin ou de la danse.

- **Apprentissage par association**

  On associe un stimulus nouveau à un mécanisme déjà appris, pour créer un nouveau savoir. Par exemple, si une réaction à une odeur est déjà apprise, on peut faire apprendre la même réaction à un son en faisant systématiquement précéder l'odeur par le son comme c'est le cas célèbre du chien de Pavlov. Pavlov démontra ainsi qu'outre le réflexe non conditionné (salivation "normale" devant la nourriture), il est possible de déclencher, par un processus d'apprentissage - ou conditionnement -, un réflexe conditionné (salivation liée au stimulus). Par exemple, si l'on présente sa nourriture au chien en même temps que retentit une sonnerie, on constate que, au bout d'un certain temps, la seule sonnerie déclenche le processus de salivation.

Apprentissage naturel
Méthodes (3)

- **Apprentissage par essais et erreurs**

  Le sujet est mis en situation, on ne lui donne aucun mode d'emploi (parfois même pas la condition de succès ou d'élimination). Pour fonctionner correctement, il faut que la solution soit assez facile à trouver, compte tenu de ce que le sujet sait déjà.
  Pour apprendre des choses complexes, il faut donc s'appuyer sur l'apprentissage par association pour enchaîner des situations de difficulté croissante et permettant de nombreuses répétitions. Cela rend cet apprentissage coûteux. Mais c'est le seul qui fonctionne encore quand la solution doit être découverte, on parle alors de démarche heuristique.

- **Apprentissage par explication**

  On explique au sujet, oralement ou par écrit, ce qu'il doit savoir (exemple : un manuel de secourisme). C'est le principe des cours magistraux.

- **Apprentissage par répétition**

  On fait faire au sujet ce qu'il doit apprendre, d'abord passivement, puis de plus en plus activement, jusqu'à ce qu'il puisse faire et refaire seul les opérations.

Apprentissage naturel
Méthodes (4)

- **Apprentissage combiné**
  C'est le plus efficace, et il très utilisé en matière d'enseignement de savoir-faire professionnel, car il combine les modalités précédentes : le sujet est mis en situation (en commençant par les plus simples), on lui montre quelques fois les bons gestes en lui expliquant les principes d'action ; on le laisse ensuite se perfectionner par une répétition de moins en moins supervisée.

- **Apprentissage par immersion**
  Les langues s'apprennent mieux en situation d'immersion totale. Par exemple, lorsque les cours ne sont donnés que dans la langue à apprendre et que le professeur ne parle avec les élèves que dans leur langue d'immersion. À défaut, il est conseillé de passer une année ou deux dans un pays parlant la langue souhaitée afin de mieux saisir les différences d'expressions orales et écrites.

Définition de l'apprentissage automatique (1)

- **Apprentissage automatique** est le terme académique utilisé le plus couramment pour désigner l'apprentissage lorsque l'apprenant n'est pas un être vivant mais un ordinateur. Ce terme est parfois remplacé par **apprentissage machine** pour traduire directement **machine learning**, par **apprentissage algorithmique** pour insister sur les aspects opérationnels ou encore par **apprentissage artificiel** qui, contrairement à l'apprentissage naturel, marque son appartenance aux sciences de l'artificiel et semble apporter quelque chose de plus profond que la simple idée d'*automatique*.

- La notion d'apprentissage automatique englobe toute méthode permettant de construire un modèle de la réalité à partir de données, soit en améliorant un modèle partiel ou moins général, soit en créant complètement le modèle.

Définition de l'apprentissage automatique (2)

- Ainsi, l'objectif de l'apprentissage automatique est d'étudier et de mettre en œuvre des mécanismes d'apprentissage naturel sur des systèmes artificiels. Les deux définitions suivantes balisent assez bien le domaine d'activités de l'apprentissage machine et ses objectifs:
  - « Apprendre consiste à construire et à modifier des *représentations* ou des modèles à partir d'une série d'expériences »
  - « Apprendre consiste à améliorer les *performances* d'un système sur une tâche donnée à partir d'expériences »

- L'apprentissage automatique, dans une définition très générale, consiste en l'élaboration de programmes qui s'améliorent avec l'expérience. Pour être plus précis, on dit qu'un programme apprend:
  - par l'expérience E par rapport à une classe de tâches T et une mesure de performances P
  - si sa performance pour les tâches de T mesurée par P s'améliore avec E

Définition de l'apprentissage automatique (3)

- Tout domaine scientifique est défini par la question centrale qu'il étudie. L'apprentissage automatique tente de répondre à la question:
  - "Comment concevoir des systèmes informatiques qui s'améliorent automatiquement avec l'expérience et quelles sont les lois fondamentales qui gouvernent tous les processus d'apprentissage?"

- Cette question couvre une grande variété de tâches d'apprentissage, telles que :
  - Comment concevoir un robot mobile autonome qui apprend à se déplacer ?
  - Comment explorer les dossiers médicaux d'anciens patients pour apprendre quels futurs patients répondront mieux à quels traitements ?
  - Comment concevoir des moteurs de recherche qui s'adaptent automatiquement aux intérêts de leurs utilisateurs ?

Exemples d'apprentissage (1)
Exemple d'induction

- A ses débuts, l'apprentissage artificiel était essentiellement basé sur l'induction. C'est le processus par lequel on tire des lois de portée générale en partant de l'observation de cas particuliers.

- Les tests de mesure du QI font souvent appel à la faculté d'induction avec des questions telles que :

  Quel est le chiffre a qui prolonge la séquence :

  1 2 3 5 ... a

- Quelques solutions possibles:
  - a=6
  - a=7
  - a=8
  - a=2π

Exemples d'apprentissage (2)
Exemple d'induction

Quelques réponses valides :

- a = 6. Argument : c'est la suite des entiers sauf 4.
- a = 7. Argument 1 : c'est la suite des nombres premiers.
- a = 7. Argument 2 : suite binaire 1(1), 10(2), 11(3), 101(5), 111(7), 1011(11), 1111(15), 10111(23), 11111(31)...
- a = 8. Argument : c'est la suite de Fibonacci.
- a = 2π (a peut être n'importe quel nombre réel supérieur ou égal à 5).

Argument : La liste ordonnée des racines du polynôme :
$P = x^5 - (11 + a)x^4 + (41 + 11a)x^3 - (61 - 41a)x^2 + (30 + 61a)x - 30a$
qui est le développement de $(x - 1).(x - 2).(x - 3).(x - 5).(x - a)$

Exemples d'apprentissage (3)
Exemple d'induction

Choix de la solution

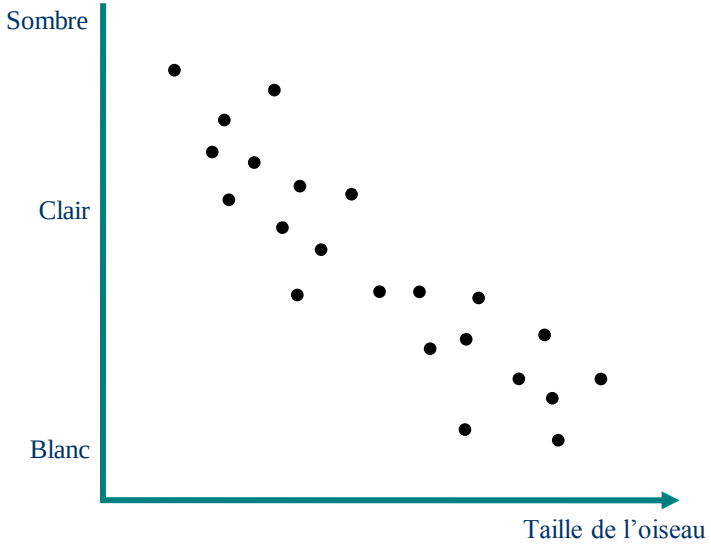
- La meilleure solution est celle qui fait intervenir le moins de concepts.
- Dans l'exemple, la solution a = 8 est préférable : elle ne nécessite que le concept d'addition.
- D'une manière générale, le principe du rasoir d'Occam conduit à choisir, pour une valeur explicative égale, la solution la plus simple.
- On peut toujours expliquer n'importe quelle solution si on se place dans un cadre assez complexe.
- On doit donc se fixer une famille de concepts à l'intérieur de laquelle on cherchera la meilleure explication des données. C'est ce qu'on appelle se donner un biais d'apprentissage.
- Le biais dépend de la représentation des données.
- Le compromis simplicité / efficacité devra guider le choix du biais.

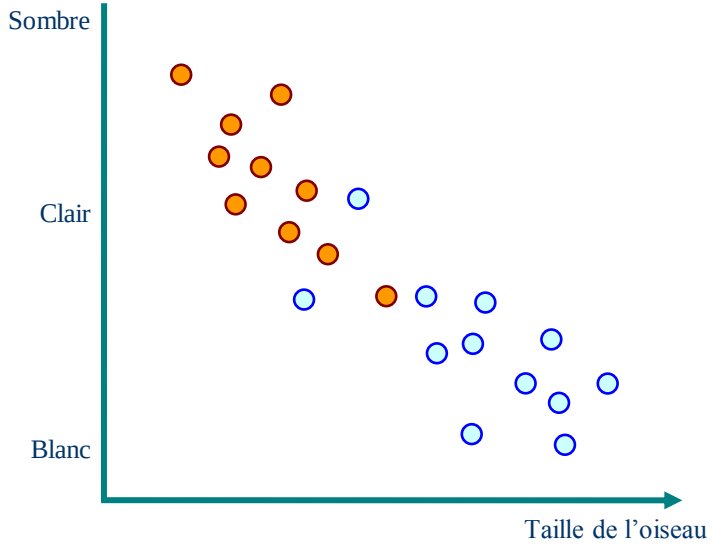
Exemples d'apprentissage (4)
Exemple de classification

Soit un étang sur lequel nagent des oies et des cygnes. Le brouillard est tombé quand arrivent deux avimateurs, l'un est débutant, l'autre est expert en ornithologie. Ils n'aperçoivent qu'une partie des animaux de manière peu distincte. L'expert arrive à les identifier alors que le débutant se contente de mesurer de manière approximative le niveau de gris du plumage et la taille de l'animal.

Exemples d'apprentissage (5)
Exemple de classification

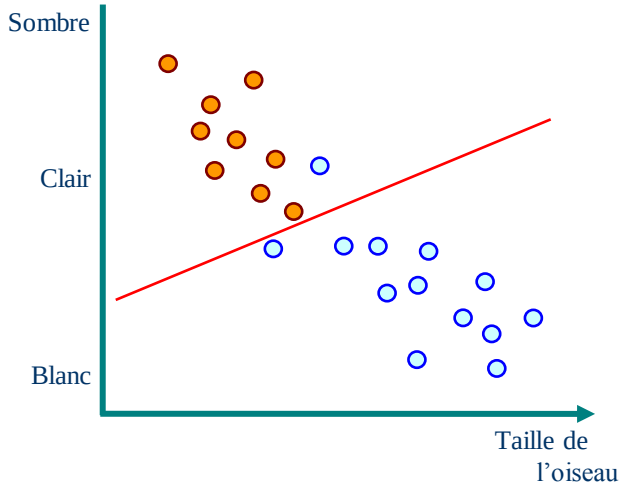
- Le premier graphique (à gauche) représente les oiseaux observés par le débutant placés dans l'espace de représentation.
- Le second graphique (à droite) représente les mêmes oiseaux, mais il est étiqueté par l'expert.
- Le cercle jaune signifie que l'oiseau est une oie, le cercle bleu qu'il est un cygne.

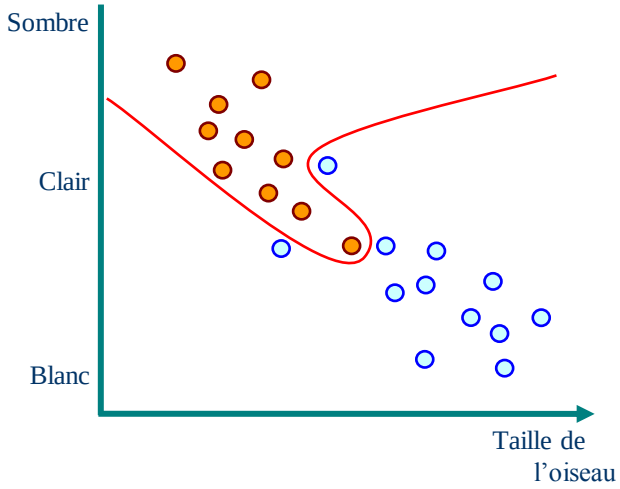




Exemples d'apprentissage (6)
Exemple de classification

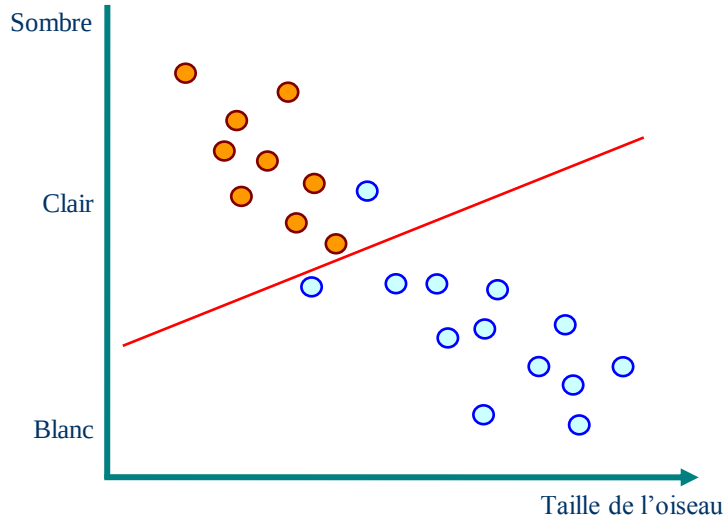
Le problème d'apprentissage est de trouver une règle qui décide, dans l'espace de représentation choisi, avec le moins d'erreurs possibles, quel oiseau est une oie et lequel est un cygne. Cette règle doit posséder de bonnes propriétés de généralisation pour fonctionner aussi sur les oiseaux non encore observés. L'apprenant doit imaginer de tracer dans le plan de représentation une ligne (droite ou courbe) qui sépare les cygnes des oies. A partir des exemples connus, il aura induit une loi générale : tout oiseau observé qui se place sous cette ligne sera classé comme un cygne, il sera classé comme une oie dans le cas contraire. Voici une règle de décision simple et une règle de décision complexe pour séparer les oies des cygnes.

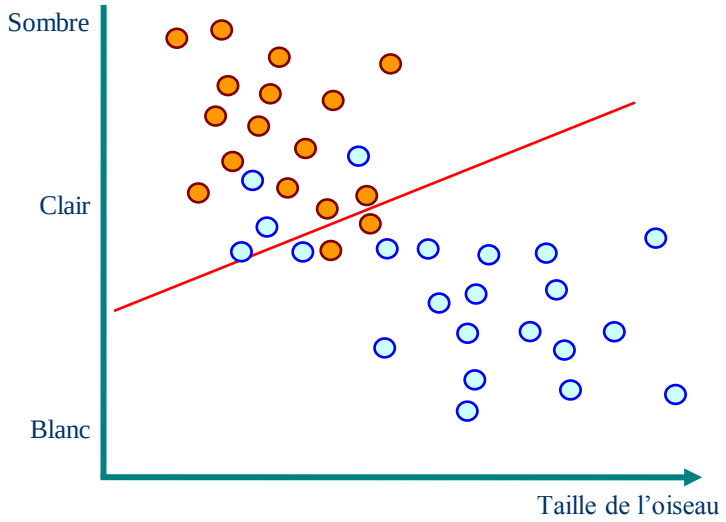




Exemples d'apprentissage (7)
Exemple de classification

Si le choix est porté sur une ligne droite, on ne peut en trouver aucune qui sépare parfaitement les exemples, mais c'est le prix à payer pour un concept aussi simple.
Quand le brouillard se lève, d'autres oiseaux deviennent visibles. Le graphique de droite illustre le test de la règle simple sur ces oiseaux. L'erreur constatée est (6/35).
Cet exemple est caractéristique de ce que font les programmes de reconnaissance de formes. C'est un apprentissage par généralisation d'une règle de décision sous forme d'une équation de droite dans le plan.





Exemples d'apprentissage (8)
Extraction de connaissances

- Une compagnie d'assurances cherche à lancer un nouveau produit, destiné à couvrir le risque de vol d'objets de valeur à domicile. Elle veut faire une campagne de publicité ciblée auprès d'une partie de ses clients. Cette compagnie ne dispose que de peu de produits du même type et par conséquent sa base de données ne comporte qu'une petite proportion d'enregistrements où un client est déjà associé à une assurance contre le vol à domicile. De plus, comme ces clients possèdent déjà un produit analogue, ce n'est pas vers eux qu'il faut principalement cibler la campagne.

- Comment savoir si un client qui n'a pas encore d'assurance de ce type sera intéressé par le nouveau produit?

Exemples d'apprentissage (9)
Extraction de connaissances

- Une solution est de chercher un profil commun aux clients qui se sont déjà montrés intéressés par un produit de ce type pour viser parmi tous les clients ceux qui ont un profil analogue. Que sera un tel profil?

- Dans la base de données, chaque client est décrit par un certain nombre de champs, que l'on peut supposer binaires. Par exemple : « âge inférieur à trente ans », « possède une maison », « a des enfants », « vit dans une zone à risque de vol », « a un système d'alarme dans sa maison », etc. Certains champs peuvent être non remplis : les clients qui ont seulement une assurance automobile n'ont pas été interrogés à la constitution de leur dossier sur l'existence d'un système d'alarme dans leur appartement.

Exemples d'apprentissage (10)
Extraction de connaissances

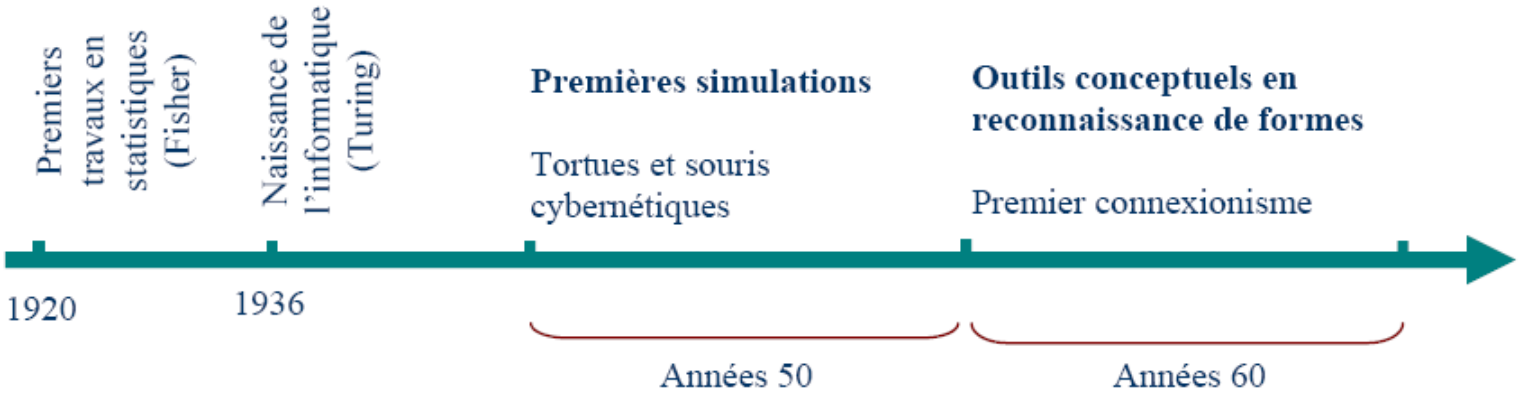
- Une façon de constituer un profil consiste à découvrir des associations dans les données, c'est-à-dire des implications logiques approximatives. Disons par exemple que la plupart des clients qui possèdent déjà une assurance contre le vol d'objets de valeur à domicile sont plutôt âgés et ont en général une voiture haut de gamme.

- Il semble raisonnable de démarcher parmi tous les clients ceux qui répondent au même profil. L'hypothèse est donc que posséder une voiture de luxe et être d'âge mûr est un profil qui implique sans doute la possession à domicile d'objets de valeur.

- Dans ce type de problème, l'expertise est présente dans les données, mais d'une manière implicite. C'est au programme de la découvrir et de l'utiliser. Il s'agit d'un apprentissage non supervisé pour l'extraction de connaissances.

Historique (1)
1ère période de l'apprentissage automatique

- Des principes préliminaires théoriques de l'apprentissage sont posés dès les premiers travaux en statistiques dans les années 1920 et 1930, cherchant à déterminer comment inférer un modèle à partir de données et surtout comment valider une hypothèse par rapport à un jeu de données (Fisher).
- Naissance de l'informatique avec les travaux de Gödel, Church et surtout Turing en 1936.
- Premières simulations de tortues ou souris cybernétiques qu'on place dans des labyrinthes en espérant les voir apprendre à en sortir de plus en plus vite.
- Deux courants d'apprentissage dans les années 1960 : un premier connexionnisme avec le Perceptron de Rosenblatt et un développement d'outils conceptuels sur la reconnaissance de formes.
- Annonce des limitations du perceptron en 1969 et arrêt, pour une quinzaine d'années de presque toutes les recherches dans le domaine.

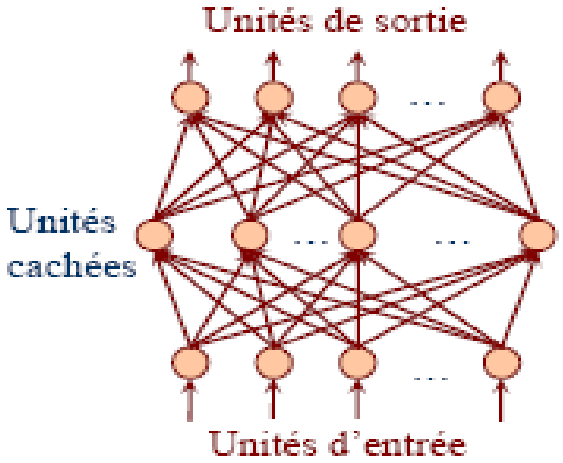


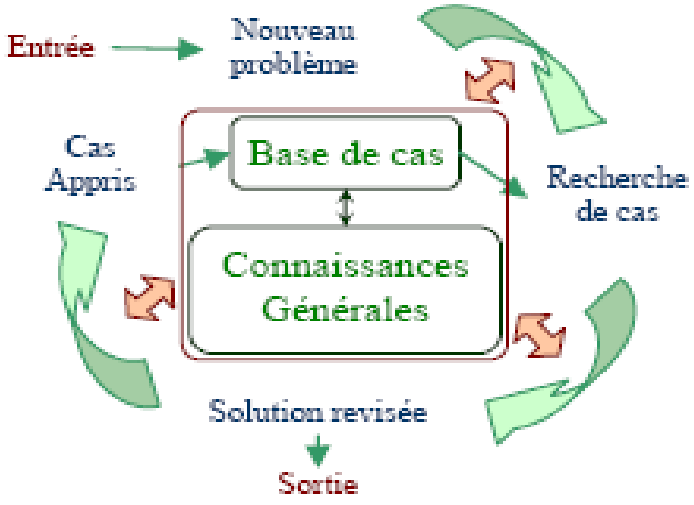
Historique (2)
2ème période de l'apprentissage automatique

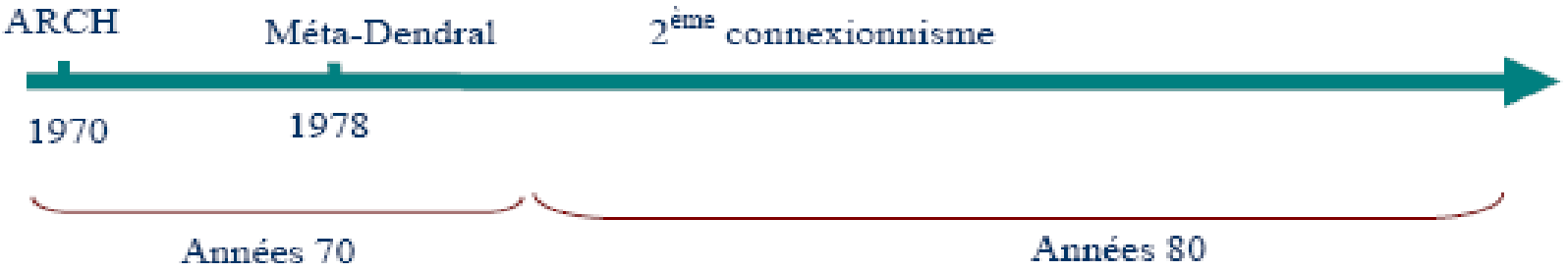
- Années 1970, l'accent est mis sur l'intelligence artificielle, c'est la période des systèmes experts. Recherches sur la représentation des connaissances et les règles d'inférence sophistiquées.
- Triomphe de systèmes impressionnants réalisant des tâches d'apprentissage spécifiques en simulant plus ou moins des stratégies mises en jeu dans l'apprentissage humain. Par exemple, ARCH qui apprend à reconnaître des arches dans un monde de blocs à partir d'exemples et de contre-exemples et META-DENDRAL qui apprend des règles dans un système dédié à l'identification de molécules chimiques.
- Facilité du dialogue entre psychologues et praticiens de l'apprentissage automatique.
- Réémergence très puissante du connexionnisme en 1985 avec la découverte de l'argorithme de rétropropagation par descente du gradient pour l'apprentissage des perceptrons multicouches.
- Recherches sur une nouvelle vision de l'apprentissage comme un processus de recherche dans un espace d'hypothèses défini a priori d'une hypothèse cohérente avec les données
- Développement, en parallèle, par les praticiens de l'apprentissage, d'algorithmes plus simples mais plus généraux que ceux de la décennie précédente : arbres de décision, algorithmes génétiques….

Historique (3)
2ème période de l'apprentissage automatique

Induction Supervisée
Arbres de décision, Algorithmes génétiques
Explanation based learning, Raisonnement basé cas

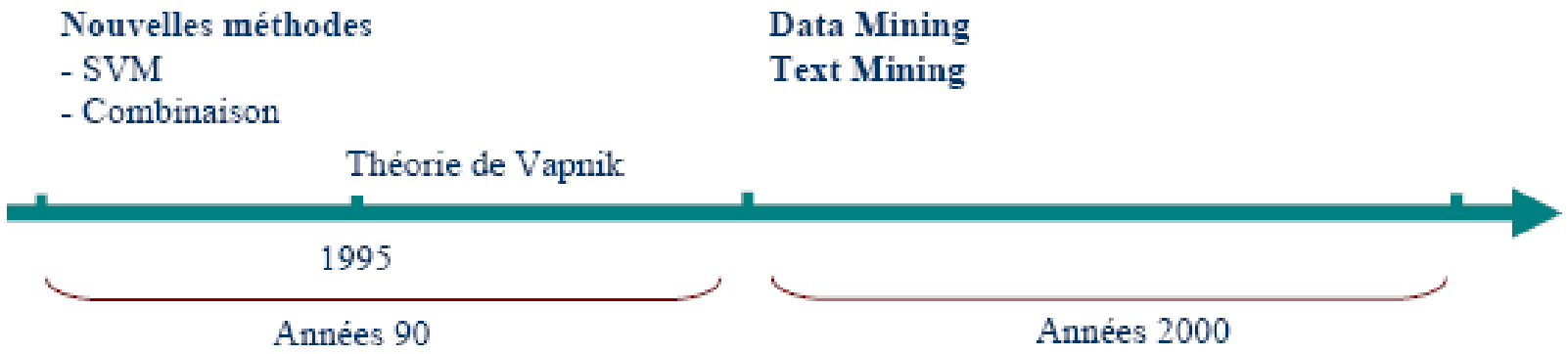






Historique (4)
3ème période de l'apprentissage automatique

- Ce n'est que dans les années 1990 que la théorie statistique de l'apprentissage a vraiment influencé l'apprentissage automatique en donnant un cadre théorique solide à des constats et interrogations empiriques.
- Effort théorique vigoureux par les théoriciens de l'approche statistique.
- Mise à l'épreuve des techniques développées sur de grandes applications à finalité économique (fouille de données) ou socio-économique (génomique).
- Nous vivons actuellement un âge d'or de l'apprentissage automatique (théorisation et mise à l'épreuve).

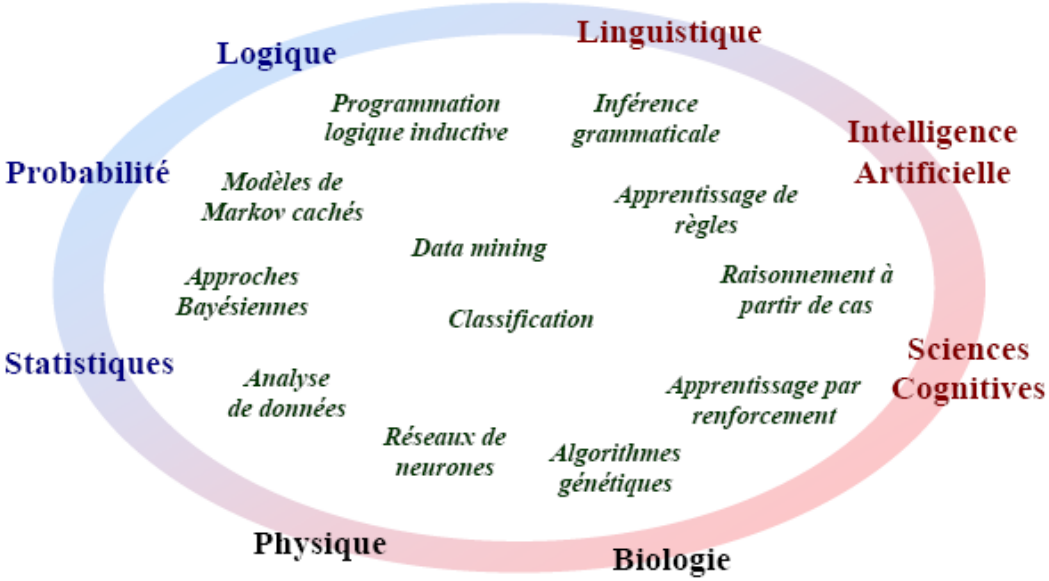


Domaines connexes (1)

- L'apprentissage automatique est à l'intersection de l'informatique et des statistiques.
- L'étude de l'apprentissage humain et animal en psychologie et en neurosciences est aussi l'un des domaines connexes de l'apprentissage automatique malgré que ce dernier n'a pas vraiment bénéficié d'un grand apport de la part des études de l'apprentissage naturel. Une synergie croissante est attendue dans les années à venir.
- D'autres domaines tels que la biologie et l'économie s'intéressent à l'adaptation des systèmes à leur environnement et un échange croissant d'idées va probalement avoir lieu entre ces disciplines et celle de l'apprentissage automatique.

Domaines connexes (2)

- Ainsi, de part la diversité des approches produites et la complexité des problèmes traités, l'apprentissage machine se trouve au carrefour de nombreuses disciplines tels les *mathématiques*, les *sciences humaines* et bien sûr l'*intelligence artificielle* ... En outre, des domaines à priori plus éloignés de l'informatique comme la *Biologie* et la *Physique* sont également une source importante d'idées et de méthodes. La carte ci-dessous propose une topologie approximative des liens entre les types d'apprentissage et les domaines scientifiques apparentés



Applications de l'apprentissage automatique (1)

- Ces 20 dernières années, l'intérêt pour l'apprentissage s'est notablement accru à partir du constat qu'il est de plus en plus difficile de prévoir dans un programme tous les cas possibles de données qu'il sera amené à traiter. En effet, il est fréquent que les programmes doivent s'adapter à leur fonction à partir de données empiriques issues de leur environnement. D'autre part la quantité et la complexité des données à traiter implique l'usage de processus d'analyse automatiques et intelligents. Le domaine de l'apprentissage artificiel regroupe toutes les méthodes qui permettent à un système (programme) de s'adapter et donner des réponses pertinentes, à partir de l'exploration de son environnement ou de l'extraction d'informations à partir d'exemples.

Applications de l'apprentissage automatique (2)

- Les méthodes d'apprentissage automatique sont très bien adaptées pour le développement de certains types d'applications telles que :
  - Des applications trop complexes pour concevoir manuellement l'algorithme correspondant mais il est relativement facile de collecter des données étiquetées pour l'apprentissage. C'est le cas des tâches perceptuelles telles la reconnaissance de la parole ou des images. Chacun de nous peut facilement dire, parmi un ensemble de photographies, quelles sont celles où se trouve sa mère mais personne ne peut écrire un algorithme pour accomplir cette tâche.
  - Des applications qui nécessitent que le logiciel s'adapte à son environnement opérationnel après leur mise en œuvre. C'est le cas typique des systèmes de dictée vocale qui doivent s'adapter à la façon de parler de leur utilisateur. L'apprentissage automatique fournit dans ce cas un mécanisme pour l'adaptation.

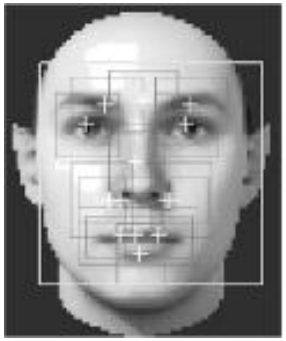
Applications de l'apprentissage automatique (3)

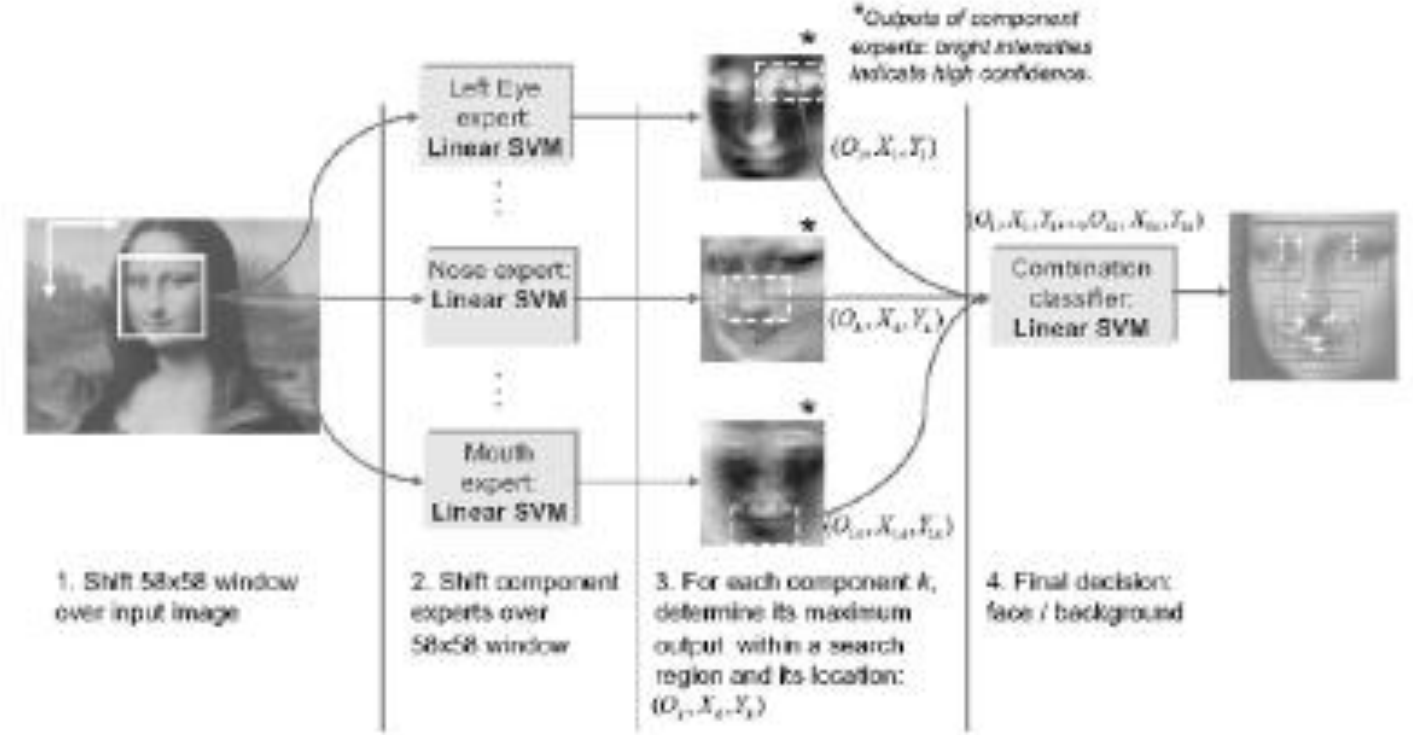
- Les applications sont nombreuses et concernent des domaines très variés. On peut citer, par exemple, la reconnaissance de formes avec, en particulier, la reconnaissance des visages, de la parole et du texte écrit, le contrôle de processus et le diagnostic de pannes, les programmes de jeu.

- Les méthodes d'apprentissage à partir d'exemples sont très utilisées dans la recherche d'informations dans de grands ensembles de données. En effet, l'évolution de l'informatique permet de nos jours de manipuler des ensembles de données de très grande taille (datawarehouse ou entrepôt de données). Par exemple, les chaînes de magasin peuvent mémoriser de grandes quantités de données concernant les consommateurs et ce qu'ils achètent.

Applications de l'apprentissage automatique (4)

- Apprendre à étiqueter des images (reconnaissance de visages)





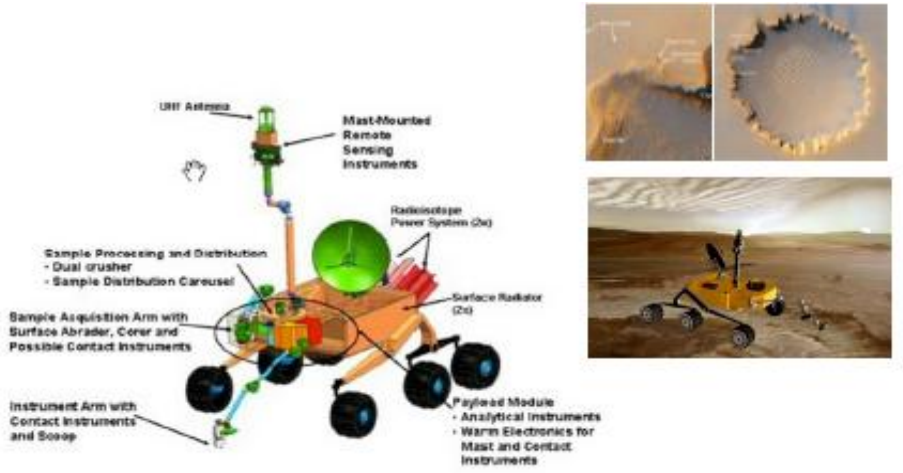
Applications de l'apprentissage automatique (5)

- Apprendre à mieux jouer
  - S'adapter à l'adversaire
  - Ne pas répéter ses fautes
  - Apprendre à jouer en équipe (équipes de robots)

- Apprentissage de comportement
  - Apprendre à marcher (insectoïdes de Brooks)

Applications de l'apprentissage automatique (6)

- Apprendre à se comporter sur une planète: Robot sur Mars



- Systèmes autonomes avec apprentissage

Applications de l'apprentissage automatique (7)

- Le développement des technologies Internet et Intranet font que de nombreuses données issues de sources diverses et dans des formats variés deviennent accessibles. Le processus de recherche d'informations dans de grands ensembles de données (KDD : Knowledge Discovery in Databases) comporte différentes étapes : la sélection des données (extraction des informations de l'entrepôt) ; la préparation des données (suppression des doublons, élimination des données aberrantes, ...); le codage des données (normalisation des données, choix de codage, ...); la phase d'extraction proprement dite appelée fouille de données (Data mining); la sortie des résultats.

Applications de l'apprentissage automatique (8)

- La phase d'extraction d'information utilise les outils usuels d'interrogation tels que les requêtes SQL standard et les requêtes multidimensionnelles, mais aussi, pour l'extraction d'informations cachées, les algorithmes d'apprentissage à partir d'exemples.
- Les algorithmes les plus utilisés sont : les *k*-plus proches voisins, les arbres de décision, les systèmes à base de règles (programmation logique inductive), les réseaux de neurones et les algorithmes génétiques.
  Citons, parmi d'autres, quelques applications et leur domaine :
  - analyse financière : prévision d'évolution de marchés,
  - marketing : établir un profil client
  - banque : attribution de prêts,
  - médecine : aide au diagnostic,
  - telecom : détection de fraudes
- Vues de cette manière, les méthodes d'apprentissage automatique vont jouer un rôle crucial en informatique mais il y aura toujours des types d'applications où elles ne risquent pas de devenir utiles telles que l'écriture des programmes de multiplication matricielle.

Paramètres d'un problème d'apprentissage (1)

- L'objectif du processus d'apprentissage
- Le protocole d'apprentissage
- Le critère de succès
- La nature des entrées (l'espace de représentation)
- La nature des résultats (l'espace des fonctions cibles)

Paramètres d'un problème d'apprentissage (2)

L'objectif

- Par rapport à la connaissance
  - Modifier le contenu de la connaissance (acquisition, révision, oubli)
  - Rendre la connaissance plus efficace (par réorganisation, optimisation ou compilation)
- Par rapport à l'environnement
  - Apprendre à reconnaître (des formes)
  - Apprendre à prédire
  - Apprendre à être plus efficace
- Par rapport à des classes abstraites de problèmes
  - Il est possible de caractériser l'apprentissage par une classe générale et abstraite de problèmes et de processus de résolution qui leurs sont liés (compression d'information, approximation, généralisation)
- Par rapport aux structures de données ou types d'hypothèses visées
  - Le choix du type de structures de données permet de guider la détermination de l'algorithme d'apprentissage à utiliser ainsi que les données qui seront nécessaires pour que l'apprentissage soit possible

Paramètres d'un problème d'apprentissage (3)

Le protocole

- Supervisé vs. non supervisé
- Degré d'interaction avec le superviseur
- Présentation des exemples à l'apprenant
  - Tous d'un seul coup (apprentissage hors-ligne ou batch learning)
  - En séquence avec réponse de l'apprenant après chaque entrée ou groupe d'entrées (apprentissage séquentiel ou en-ligne)

Paramètres d'un problème d'apprentissage (4)

Le critère de succès

- Relatif à un observateur externe
  - Taux d'erreur en classification (reconnaissance des formes)
  - Nombre de documents pertinents (recherche documentaire)
  - Simplicité du résultat, compréhensibilité et intelligibilité par un expert
  - Coût d'obtention du résultat
- Interne, relatif à l'apprenant
  - Mesure de coût ou de perte qui guide la modification des paramètres pour faire converger le modèle d'apprentissage

Paramètres d'un problème d'apprentissage (5)

Les entrées

- Qualité
  - Bruit
  - Origine, représentativité
- Nature
  - Numériques
  - Symboliques (binaires, nominales, séquencielles, logiques, etc.)
  - Mixtes

Paramètres d'un problème d'apprentissage (6)

Les résultats

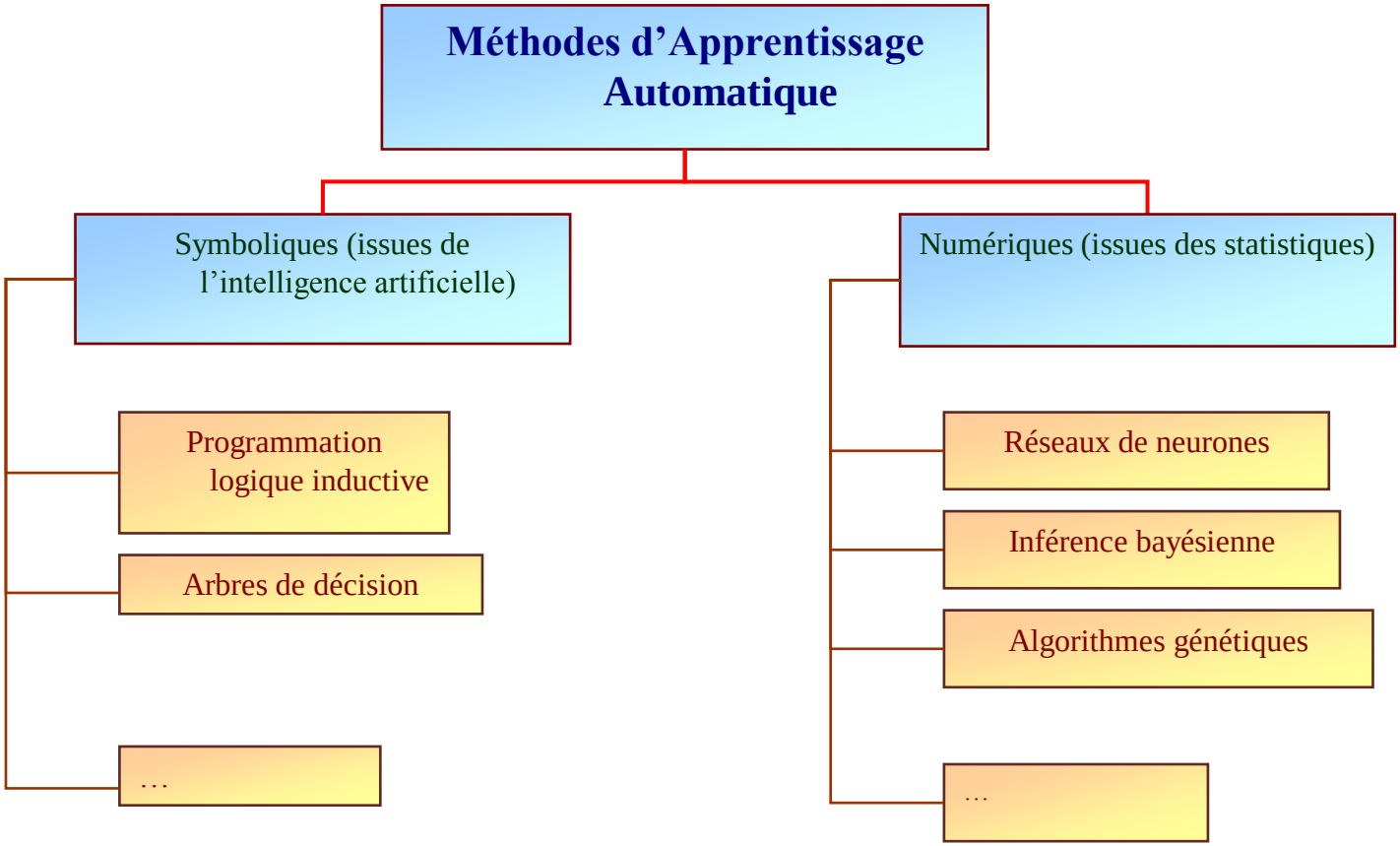
- Décision dans l'espace de représentation des entrées
  - Plus proches voisins
  - Apprentissage par analogie
- Optimisation des paramètres d'une fonction ou d'un algorithme
  - Hyperplans séparateurs, réseaux connexionnistes
  - Modèles de markov cachés
- Optimisation de la structure et des paramètres
  - Réseaux bayésiens
- Construction d'un concept
  - Arbres de décision
  - Programmation Logique Inductive

Classification des méthodes d'apprentissage automatique (1)

- Les méthodes utilisées en apprentissage automatique peuvent être classées de différentes manières, selon le point de vue sur lequel se base la classification établie.

- L'une des classifications les plus simplifiées propose d'affecter les méthodes d'apprentissage à l'une deux catégories principales, l'apprentissage symbolique et l'apprentissage numérique. Cette classification se base sur l'existence de deux tendances principales en apprentissage automatique, celle issue de l'intelligence artificielle dite symbolique (avec des méthodes telles que la programmation logique inductive, les arbres de décision…) et celle issue des statistiques, qualifiée de numérique (avec des méthodes telles que les réseaux de neurones, l'inférence bayésienne, les algorithmes génétiques)

Classification des méthodes d'apprentissage automatique (2)

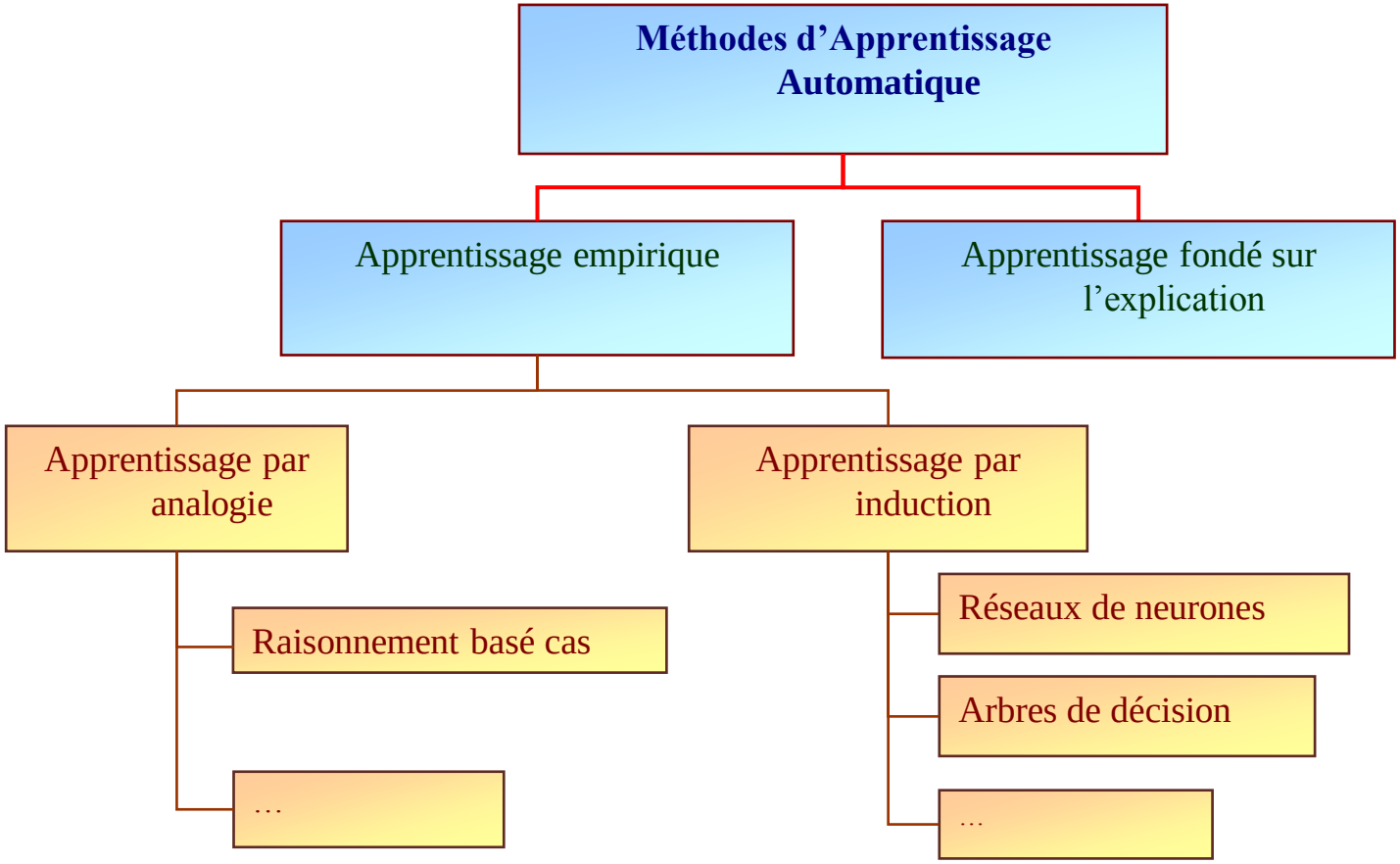


Classification des méthodes d'apprentissage automatique (3)

Selon le point de vue acquisition de connaissances en IA

- L'acquisition de connaissances en intelligence artificielle peut être faite par explicitation de connaissances d'un domaine avec l'aide d'un ingénieur de connaissances et de l'expert, ou à travers des processus semi-automatiques ou totalement automatiques d'acquisition de connaissances. Ces méthodes automatiques sont connues comme *méthodes d'apprentissage automatique* ("machine learning methods").
- Dans ce contexte, l'apprentissage automatique se présente comme une alternative prometteuse pour améliorer le processus d'acquisition de connaissances. Les systèmes experts de première génération, du fait des problèmes liés à l'acquisition de connaissances, ont évolué vers des systèmes dits de deuxième génération qui ont intégré, parmi d'autres techniques, des outils d'apprentissage automatique.
- Les différentes méthodes d'apprentissage automatique sont classées en deux groupes : les méthodes d'apprentissage empirique (*Empirical Learning*) et les méthodes d'apprentissage fondées sur l'explication (*Explanation Based Learning*).

Classification des méthodes d'apprentissage automatique (4)



Classification des méthodes d'apprentissage automatique (5)

Méthodes d'Apprentissage Empirique

Les méthodes d'apprentissage empirique sont fondées sur l'acquisition de connaissances à partir d'exemples. Elles incluent aussi des méthodes connues sous les noms d'*Instance based learning* ou *Exemplar based learning*. On peut citer comme exemples de méthodes d'apprentissage empirique les techniques suivantes : le raisonnement basé sur des cas, les réseaux de neurones artificiels, les arbres de décision, les algorithmes génétiques d'induction de règles etc.
Ces méthodes se divisent entre les méthodes d'apprentissage par analogie et les méthodes d'apprentissage par induction.

Apprentissage par Analogie

Les approches fondées sur l'analogie essayent de faire le transfert des connaissances sur une tâche bien connue vers une autre moins connue. Ainsi, il est possible d'apprendre des nouveaux concepts ou de dériver des nouvelles solutions à partir de concepts et solutions similaires connues. Ainsi, deux notions deviennent très importants dans la définition de l'apprentissage par analogie : le *transfert* et la *similarité*. Ce type d'approche est aussi dénommé apprentissage fondé sur la similarité.
Les approches empiriques fondées sur l'analogie, telles que les systèmes de raisonnement fondé sur des cas (CBR), peuvent être considérées plutôt comme des méthodes de raisonnement que comme des méthodes d'apprentissage.

Classification des méthodes d'apprentissage automatique (6)

- **Apprentissage par Induction**

L'apprentissage par induction reste toujours une des principales méthodes étudiées dans le domaine de l'apprentissage automatique. Dans cette approche, on cherche à acquérir des règles générales qui représentent les connaissances obtenues à partir d'exemples. Les règles ainsi obtenues peuvent être représentées d'une façon explicite (facilement interprétables) ou d'une façon implicite avec un codage qui n'est pas toujours facile à interpréter.

L'algorithme d'apprentissage par induction reçoit un ensemble d'exemples d'apprentissage et doit produire des règles de classification, permettant de classer les nouveaux exemples. Le processus d'apprentissage cherche à créer une représentation plus générale des exemples, selon une *méthode de généralisation de connaissances*. Ce type de méthodes est aussi appelé apprentissage de concepts ou bien acquisition de concepts. Parmi les approches d'apprentissage empirique par induction les plus connues, on trouve les réseaux de neurones artificiels et les arbres de décision. L'algorithme d'apprentissage par induction peut fonctionner de façon *supervisée* ou *non supervisée*

Classification des méthodes d'apprentissage automatique (7)

- **Apprentissage supervisé (supervised learning)**

Les exemples d'apprentissage sont étiquetés afin d'identifier la classe à laquelle ils appartiennent. Le but de l'algorithme de classification est de correctement classifier les nouveaux exemples dans les classes définies dans la phase d'apprentissage.

- **Apprentissage non-supervisé (unsupervised learning, clustering)**

L'algorithme d'apprentissage cherche à trouver des régularités dans une collection d'exemples, puisque dans ce type d'apprentissage on ne connaît pas la classe à laquelle les exemples d'apprentissage appartiennent. Une technique employée consiste à implémenter des algorithmes pour rapprocher les exemples les plus similaires et éloigner ceux qui ont le moins de caractéristiques communes.

Classification des méthodes d'apprentissage automatique (8)

- **Méthodes d'Apprentissage fondées sur l'Explication**

  Les méthodes inductives, telles que les réseaux de neurones ou les arbres de décision, ont besoin d'un nombre significatif d'exemples pour pouvoir bien généraliser les connaissances (induire des règles ou des concepts). Ceci restreint les possibilités d'application de ces méthodes, puisqu'on n'a pas toujours une base d'exemples assez grande et complète sur le domaine traité.

  Les méthodes d'apprentissage fondées sur l'explication (*Explanation-Based Learning Methods - EBL*) utilisent des connaissances (théoriques) préexistantes et un raisonnement déductif pour augmenter l'information fournie par des ensembles d'exemples. Ces méthodes sont connues sous le nom d'apprentissage par analyse (*Analytical Learning*).

  Dans les méthodes EBL, les connaissances sont dérivées à partir d'un simple cas par explication des raisons pour lesquelles il représente un exemple du concept appris. La méthode EBL utilise les connaissances préexistantes pour analyser, ou expliquer, comment chaque exemple observé lors de l'apprentissage satisfait les concepts existants. Ensuite, cette explication est utilisée pour différencier les attributs pertinents de l'exemple d'apprentissage de ceux qui ne le sont pas. De cette façon, l'exemple pourra être généralisé par un raisonnement logique, à la place des raisonnements statistiques souvent utilisés par les autres méthodes. Ces méthodes servent donc à améliorer les performances du système, grâce à des traitements qui rendent l'utilisation des connaissances plus efficace.

Classification des méthodes d'apprentissage automatique (9)

Selon la stratégie d'apprentissage utilisée

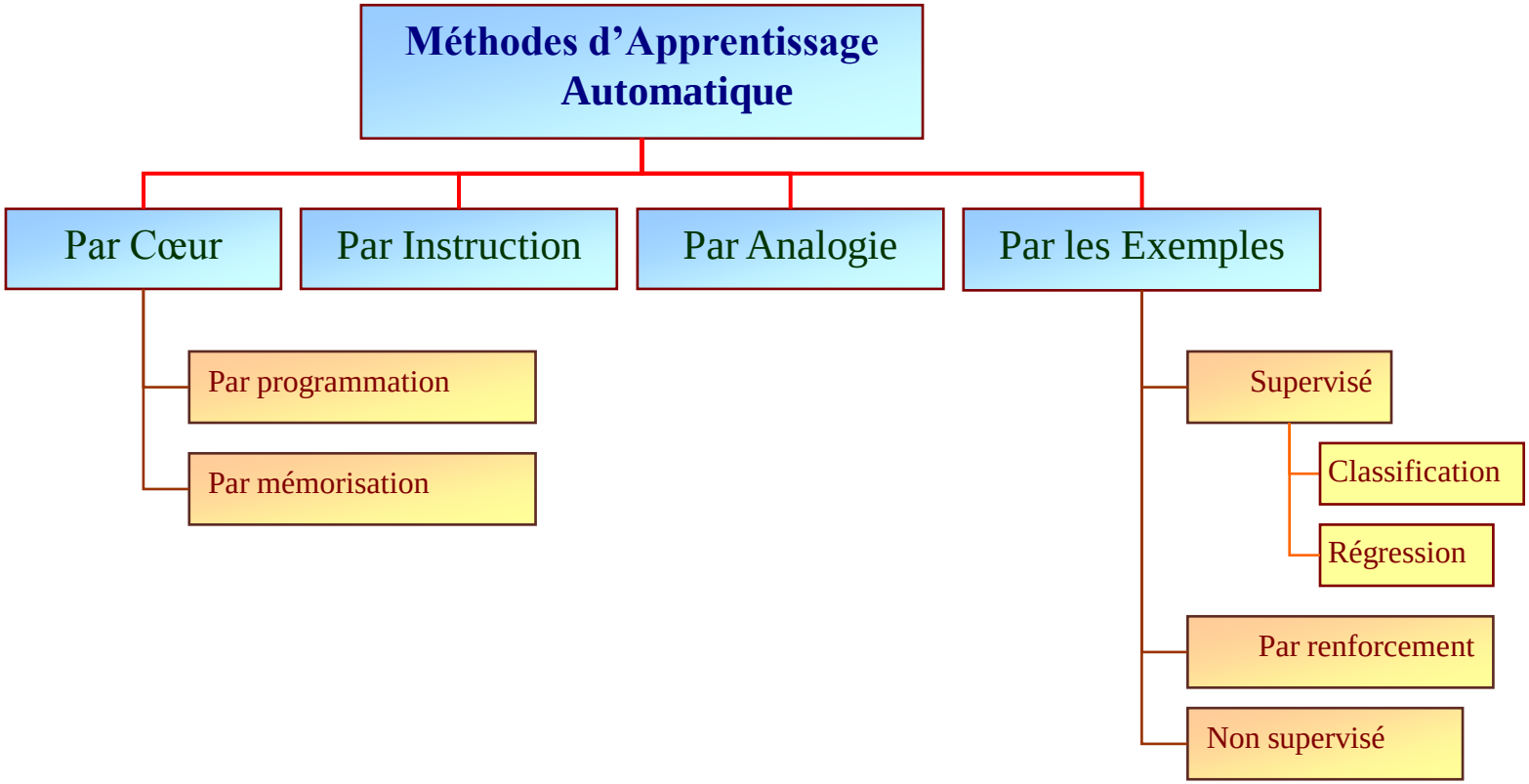
De manière générale, quand il s'agit d'apprentissage, on distingue deux entités: le professeur et l'apprenant.

- Le professeur (teacher) ou superviseur est l'entité qui détient les connaissances nécessaires pour accomplir une tâche spécifique.
- L'apprenant (learner) est l'entité qui doit apprendre (l'ordinateur, dans le cas de l'apprentissage automatique) pour acquérir les connaissances nécessaires pour accomplir une tâche spécifique

On peut distinguer les stratégies d'apprentissage selon la quantité d'inférences effectuées par l'apprenant sur l'information fournie par le professeur. Si un ordinateur est programmé directement, tout l'effort cognitif est accompli par le programmeur (professeur) et l'apprenant n'effectue aucune inférence. A l'opposé, si le système découvre de nouvelles théories ou invente de nouveaux concepts, il effectue beaucoup d'inférences pour dériver des connaissances à partir d'observations et d'expérimentations. Un cas intermédiaire correspondrait à un étudiant qui résoud un problème mathématique par analogie à des problèmes résolus dans un manuel. Ce processus nécessite moins d'inférence que de découvrir un nouveau théorème en mathématiques.

Classification des méthodes d'apprentissage automatique (10)

La taxonomie suivante de l'apprentissage automatique reflète une quantité croissante d'inférence effectuée par l'apprenant. Quatre types d'apprentissage y sont identifiés : Apprentissage par cœur, par instruction, par analogie et à partir d'exemples.



Classification des méthodes d'apprentissage automatique (11)

- **Apprentissage par cœur**
  Consiste en l'implantation directe de nouvelles connaissances à l'apprenant qui n'effectue aucune inférence ou transformation sur ces connaissances. Des variantes de ce type de méthodes incluent :
  - L'apprentissage par programmation : style usuel de la programmation des ordinateurs
  - L'apprentissage par mémorisation de faits et de données sans effectuer d'inférences (les systèmes primitifs de bases de données)
- **Apprentissage par Instruction**
  L'apprentissage par instruction (ou *learning by being told*) consiste en une acquisition de connaissances à partir d'un professeur ou d'un manuel, nécessitant que l'apprenant transforme les nouvelles connaissances en une représentation interne et les intègre graduellement avec des connaissances préalables pour en permettre l'utilisation. Ce type d'apprentissage correspond à celui utilisé dans l'enseignement.
- **Apprentissage par Analogie**
  Consiste à acquérir de nouvelles connaissances ou capacités par la transformation et l'adaptation de connaissances existantes qui ont une grande similarité avec les concepts désirés. Ce type d'apprentissage nécessite plus d'inférence que ses prédécesseurs.

Classification des méthodes d'apprentissage automatique (12)

- **Apprentissage à partir d'exemples**

A partir d'un ensemble d'exemples correspondant à un concept, l'apprenant induit une description générale de concept qui décrit les exemples. La quantité d'inférence effectuée par l'apprenant est largement supérieure à celle nécessaire dans les cas précédents. L'apprentissage à partir d'exemples est devenu si populaire qu'on le nomme simplement Apprentissage. On y trouve trois grandes familles de techniques d'apprentissage : supervisé, par renforcement et non supervisé.

- **Apprentissage supervisé:**

Selon le type de sortie, on distingue la classification et la régression

  - **Classification**

Si l'espace de sortie contient un nombre discret de classes ou catégories, on parle de classification. Les problèmes de reconnaissance de formes en sont un exemple typique.

  - **Régression**

Si l'espace de sortie contient des valeurs de variables continues, on parle de régression ou d'apprentissage de fonction. Des exemples typiques de régression correspondent à la prédiction ou l'estimation de valeurs de parts de marché ou de mesures physiques (pression, température…)

- **Apprentissage par Renforcement**

L'algorithme apprend un comportement étant donné une observation. L'action de l'algorithme sur l'environnement produit une valeur de retour qui guide l'algorithme d'apprentissage. Un exemple typique est l'apprentissage du jeu d'échecs.

- **Apprentissage non supervisé**

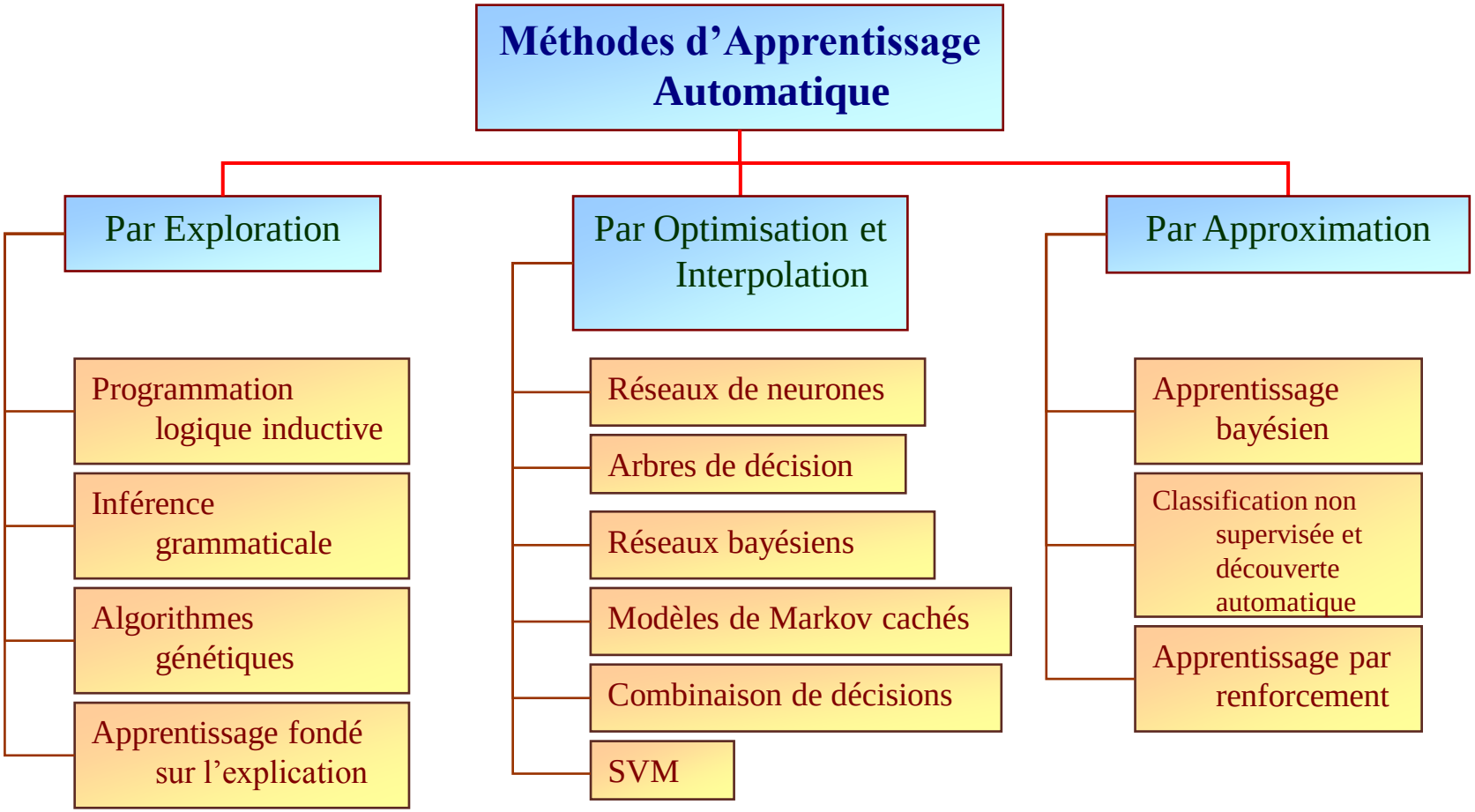
- Des exemples typiques incluent les problèmes de segmentation d'images

Classification des méthodes d'apprentissage automatique (13)

Selon le degré de structuration de l'espace des concepts à apprendre

- L'apprentissage est envisagé comme une recherche, dans un espace d'hypothèses défini a priori, d'une hypothèse cohérente avec les données. Lorsque les représentations des concepts à apprendre forment des ensembles fortement structurés, on effectue un apprentissage par exploration. L'apprentissage est dit par optimisation et interpolation lorsque les connaissances sur la structure des espaces d'hypothèses sont plus faibles et on dispose plutôt d'une notion de voisinage et d'une mesure de performance. On parle d'apprentissage par approximation lorsque l'information a priori sur la nature des concepts à apprendre est très faible ou nulle.
- Plus la connaissance préalable est faible, plus l'apprentissage nécessite de données pour aboutir. En contrepartie, les méthodes développées pour les tâches dans lesquelles on dispose de peu d'informations préalables sont aussi celles qui sont d'usage le plus général, s'adaptant à tous les contextes. C'est pourquoi ces méthodes telles que les réseaux neuronaux ou les algorithmes génétiques sont les plus populaires.

Classification des méthodes d'apprentissage automatique (14)



Evaluation d'un Apprentissage

Problème de l'évaluation des modèles d'apprentissage

- Comment on peux mesurer la qualité d'un modèle d'apprentissage ?
- L'erreur sur les exemples d'apprentissage n'est pas un bon indicateur pour des nouvelles données
- Solution : diviser les données en plusieurs ensembles d'échantillons

La représentation des exemples

- Un exemple est un couple (x,c), où $x \in \mathcal{X}$ est la description de l'objet à classer et $c \in C$ représente la classe de x.
- D'une manière plus générale, un problème d'*apprentissage inductif* consiste à retrouver une fonction entre un espace d'entrée et un espace de sortie grâce à un ensemble fini d'exemples, c'est à dire de couples entrées-sorties.
- L'espace $\mathcal{X}$ est appelé l'espace de représentation ou l'espace des instances.

Classe, concept

La **classe** de l'exemple est un entier c dans

$$\mathcal{C} = \{\omega_1, \omega_2, ..., \omega_C\}$$

C désigne le nombre de classes possibles.

Quand $C = 2$, on utilise le mot : **concept**

Un concept partage de l'espace de représentation en deux parties, l'une où il est vérifié, l'autre où il est invalidé.

On note alors $\mathcal{C} = \{VRAI, FAUX\}$ (ou parfois $\mathcal{C} = \{+, -\}$)

On appele **contre-exemples** les données classées $FAUX$ (on garde le mot d'**exemples** pour les autres).

Types d'échantillons

Un **échantillon** est un ensemble fini d'exemples.

On fait l'hypothèse indispensable que les exemples sont tirés de manière aléatoire et indépendante selon les C distributions de probabilités $P(\mathcal{X},\mathcal{C})$.

On distinguera dans la suite trois sortes d'échantillons :

- d'apprentissage
- de validation
- et de test.

Apprentissage d'une règle de classification

- Une **règle de classification** ou de **décision** h est une application définie sur $\mathcal{X}$ à valeurs dans $\mathcal{C}$.
- L'**apprentissage** d'une règle de classification consiste d'abord à choisir un ensemble $\mathcal{H}$ de règles possibles ; puis, à partir de l'examen d'un échantillon d'**apprentissage**, à trouver dans $\mathcal{H}$ une règle h.
- L'échantillon de **validation** est parfois utilisé comme "contrôleur" dans l'algorithme d'apprentissage.
- L'échantillon de **test** sera utilisé pour vérifier la qualité de l'apprentissage réalisé.

Taux d'erreur apparent et réel

Soit m la taille de l'ensemble d'apprentissage

Soit m_{err} le nombre d'exemples de cet ensemble qui sont mal classés par une certaine règle h choisie dans $\mathcal{H}$.

Le taux d'erreur apparent ou **risque empirique** de h est :

$$f_{err}(h) = m_{err}/m$$

Il est possible de définir une mesure plus fine du taux d'erreur apparent par une **matrice de confusion.**

La probabilité d'erreur ou **taux d'erreur réel** ou **risque réel** de h, que l'on note $P_{err}(h)$, est la probabilité que h classe mal un exemple tiré selon $P(\mathcal{X},\mathcal{C})$.

Ensemble de test, matrice de confusion

On divise l'ensemble des exemples en deux parties : le premier est utilisé pour l'apprentissage de la règle h et le second sert à sa validation a posteriori. Ce second ensemble est appelé **ensemble (d'exemples) de test.**

La **matrice de confusion** $M_h(i,j)$ d'une règle de classification est une matrice $C \times C$ dont l'élément générique donne le nombre d'exemples de test de la classe i qui ont été classés dans la classe j.

Si toutes les erreurs sont considérées comme également graves, la somme des termes non diagonaux de M, divisée par la taille de l'ensemble de test, sera l'estimation $\widehat{P_{err}}$ de la probabilité d'erreur exacte P_{err} du classificateur appris.

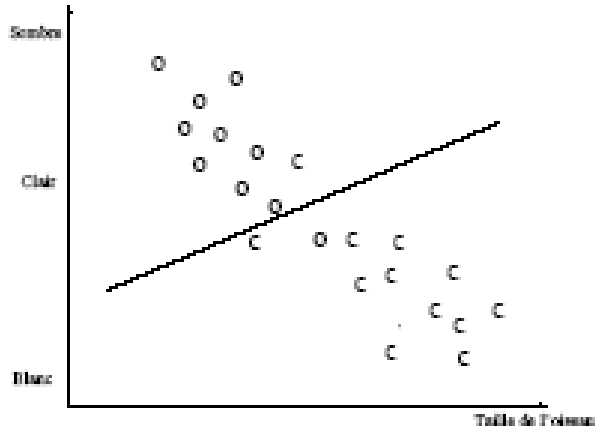
Exemple de matrice de confusion

Matrice de confusion

Classe prédite

Classe observée	FLOWER_TYPE	IRIS-VERSICOLOR	IRIS-SETOSA	IRIS-VIRGINICA	Total
	IRIS-VERSICOLOR	47	0	3	50
	IRIS-SETOSA	0	50	0	50
	IRIS-VIRGINICA	1	0	49	50
	Total	48	50	52	150

Matrice de confusion d'apprentissage et calcul du risque empirique

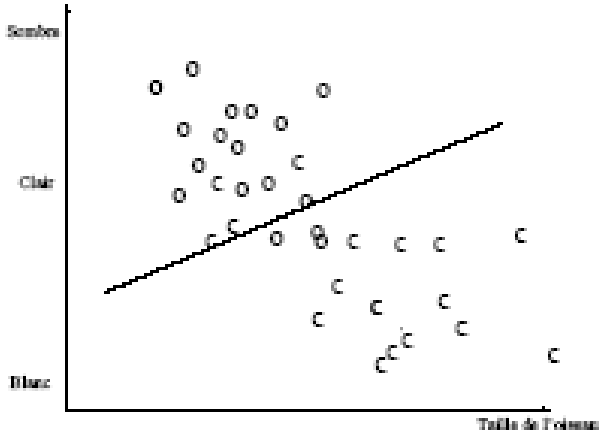


Une règle de décision simple pour séparer les oies des cygnes.

Matrice de confusion d'apprentissage. Erreur empirique : $\frac{1+1}{9+1+1+11} \simeq 9\%$

	O	C
O	9	1
C	1	11

Matrice de confusion de test et estimation du risque réel



Le test de la règle simple sur d'autres oiseaux.

Matrice de confusion de test. $\widehat{P_{err}} = \frac{3+4}{14+3+4+13} \simeq 24\%$.

	O	C
O	14	3
C	4	13

Vocabulaire utilisé pour l'apprentissage d'un concept

Pour un concept A et la matrice de confusion :

Classés \ Exacts	A	$\overline{A}$
A	a	b
$\overline{A}$	c	d

Le concept A est $VRAI$ exactement $(a+c)$ fois dans les données et $FAUX$ $(b+d)$ fois.

Le classificateur trouve $(a+b)$ fois A comme $VRAI$ et $(c+d)$ fois A comme $FAUX$.

Vocabulaire utilisé pour l'apprentissage d'un concept

a nombre de *bonnes acceptations* ou de *vrais positifs* ou de *détections* ou de *reconnaissances*

b nombre de *fausses alarmes* ou de *faux positifs*

c nombre de *faux rejets* ou de *faux négatifs* ou de *non-détections* (miss)

d nombre de *rejets corrects* ou de *vrais négatifs*

$\frac{b}{a+b}$ taux de *fausse alarme* ou *taille* ou *probabilité d'erreur de type 1*

$\frac{c}{c+d}$ taux de *faux rejet* ou *probabilité d'erreur de type 2*

$\frac{b+c}{a+b+c+d}$ taux d'*erreur*

$\frac{a+d}{a+b+c+d}$ taux de *reconnaissance* ou *puissance*

$\frac{a}{a+b}$ *précision*, ou *spécificité*

$\frac{a}{a+c}$ *rappel*

$\frac{d}{b+d}$ *sensibilité*

Classés \ Exacts	A	$\overline{A}$
A	a	b
$\overline{A}$	c	d

Mesures générales de la qualité d'une classification

- Taux de succès : nombre d'individus bien classés divisé par le nombre total d'individus (T_Succès_IRIS=0,973)

FLOWER_TYPE	IRIS-VERSICOLOR	IRIS-SETOSA	IRIS-VIRGINICA	Total
IRIS-VERSICOLOR	47	0	3	50
IRIS-SETOSA	0	50	0	50
IRIS-VIRGINICA	1	0	49	50
Total	48	50	52	150

Mesures générales de la qualité d'une classification

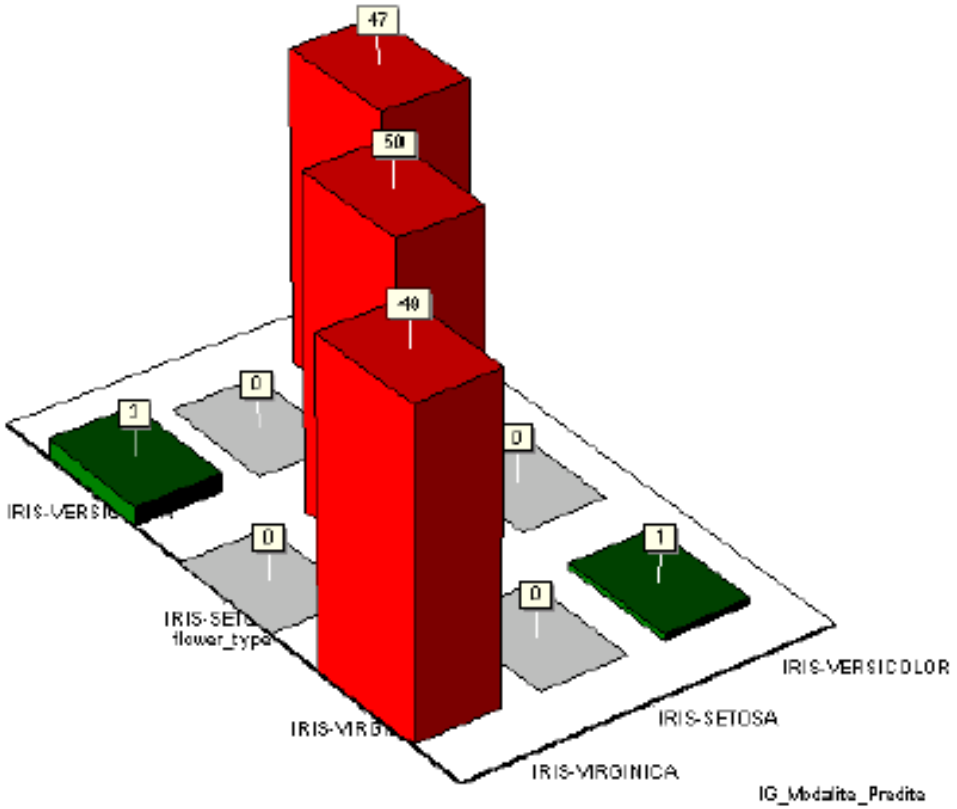
- Taux d'erreur : nombre d'individus mal classés divisé par le nombre total d'individus (T_Erreur_IRIS=0,026)

FLOWER_TYPE	IRIS-VERSICOLOR	IRIS-SETOSA	IRIS-VIRGINICA	Total
IRIS-VERSICOLOR	47	0	3	50
IRIS-SETOSA	0	50	0	50
IRIS-VIRGINICA	1	0	49	50
Total	48	50	52	150

Qualité de la classification -IRIS

Observation :
le taux d'erreur sur les données d'apprentissage est optimiste

T_Erreur_IRIS=0,026



Evaluation des résultats

- Ensemble de test : ensemble d'individus indépendants qui n'ont pas été utilisés dans la construction du modèle

- Important : ne pas utiliser les données de test pour la création du modèle !
- Certains modèles dépendent d'un ou plusieurs paramètres $\Rightarrow$ il faut les optimiser
- Il ne faut pas utiliser les données de test pour optimiser les paramètres

Evaluation des résultats

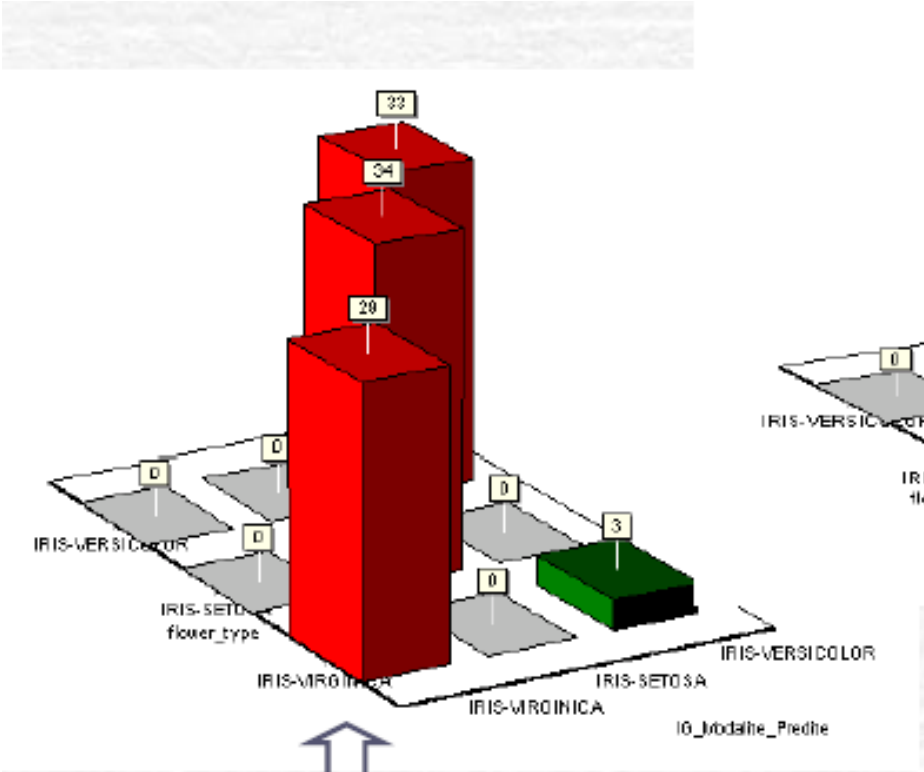
- Démarche : utiliser trois ensembles de données:
  - Données d'apprentissage : pour la création du modèle
  - Données de validation : pour optimiser les paramètres
  - Données de test : pour vérifier la qualité du modèle
  - Remarque: parfois les termes test et validation sont utilisé de manière inverse
- Beaucoup de données d'apprentissage $\Rightarrow$meilleur modèle
- Beaucoup de données de test $\Rightarrow$ précision dans l'estimation de l'erreur
- L'idéal : avoir les deux!

Evaluation des résultats

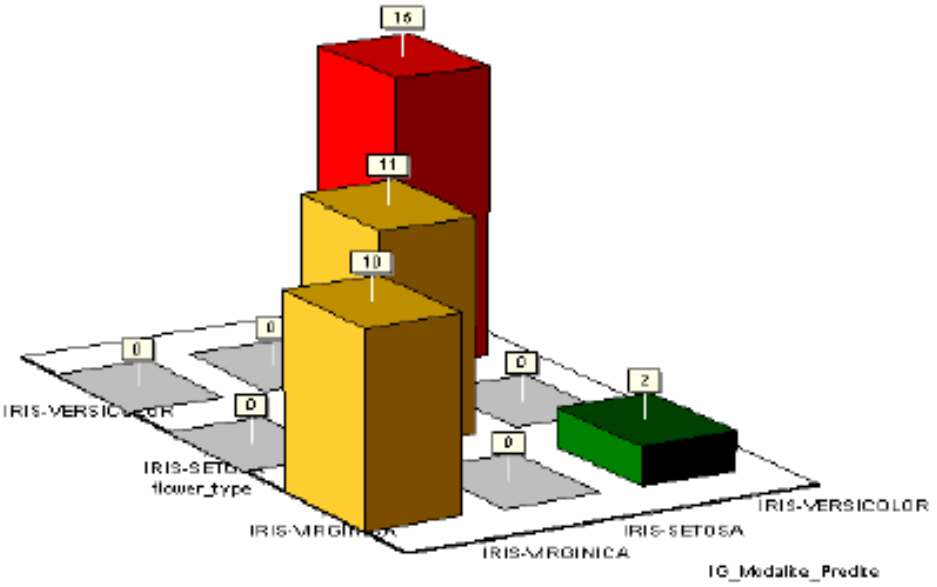
- Problème : la quantité de données est limitée
- Solution : utiliser 2/3 pour l'apprentissage et 1/3 pour des tests : T_Erreur_IRIS=0,030

FLOWER__TYPE	IRIS-VERSICOLOR	IRIS-SETOSA	IRIS-VIRGINICA	Total
IRIS-VERSICOLOR	33	0	0	33
IRIS-SETOSA	0	34	0	34
IRIS-VIRGINICA	3	0	29	32
Total	36	34	29	99

Evaluation des résultats



Taux d'erreur sur l'échantillon d'apprentissage : 3/99 = 0,030



Taux d'erreur sur l'échantillon de test 2/51=0,039

Evaluation des résultats

- Problème : les échantillons ne sont pas représentatifs (ex. données de test : manque une classe)
- Solution : s'assurer que chaque classe est représentée dans des proportions égales dans les deux échantillons

- Problème : fiabiliser le choix de l'échantillon de test
- Solution : répéter l'échantillonnage
  - A chaque étape sélectionner aléatoirement l'échantillon de test
  - Calculer la moyenne des taux d'erreur
  - Mais : chevauchement des différents échantillons de test

Validation croisée (N-fold cross validation)

- Évite le chevauchement des échantillons de test et permet d'obtenir une mesure précise du taux d'erreur réel mais nécessite de faire l'apprentissage N fois
- Les données sont divisées en N sous-ensembles (en général, N entre 5 et 10) de tailles égales
- Retenir l'un des échantillons (i) est utilisé pour tester et les autres (N-1) pour apprendre
- Estimer le risque réel sur l'échantillon i
- Recommencer N fois en faisant varier l'échantillon i de 1 à N
- Le taux d'erreur global est estimé avec la moyenne des taux d'erreur ainsi obtenus (moyenne des estimations de risque réel)

Calcul du rappel

Le **rappel**= la proportion d'individus pertinents retournés par le système comparé à la totalité des individus pertinents existants dans l'ensemble de données.

FLOWER__TYPE	IRIS-VERSICOLOR	IRIS-SETOSA	IRIS-VIRGINICA	Total
IRIS-VERSICOLOR	46	0	4	50
IRIS-SETOSA	0	50	0	50
IRIS-VIRGINICA	3	0	47	50
Total	49	50	51	150

Exp: 0.92=46/50 (nbre d'individus retournés/total ligne)

Rappel

FLOWER__TYPE	IRIS-VERSICOLOR	IRIS-SETOSA	IRIS-VIRGINICA	Total
IRIS-VERSICOLOR	0,9200	0,0000	0,0800	1,0000
IRIS-SETOSA	0,0000	1,0000	0,0000	1,0000
IRIS-VIRGINICA	0,0600	0,0000	0,9400	1,0000
Total	0,3267	0,3333	0,3400	1,0000

Calcul de la précision

La **précision**= la proportion d'individus pertinents parmi les individus retournés par le système. Le rappel et la précision sont souvent utilisés pour évaluer les résultats d'une classification

FLOWER__TYPE	IRIS-VERSICOLOR	IRIS-SETOSA	IRIS-VIRGINICA	Total
IRIS-VERSICOLOR	46	0	4	50
IRIS-SETOSA	0	50	0	50
IRIS-VIRGINICA	3	0	47	50
Total	49	50	51	150

Exp:
0.9388=46/49
(nbre d'individus retournés/total colonne)

Précision

FLOWER__TYPE	IRIS-VERSICOLOR	IRIS-SETOSA	IRIS-VIRGINICA	Total
IRIS-VERSICOLOR	0,9388	0,0000	0,0784	0,3333
IRIS-SETOSA	0,0000	1,0000	0,0000	0,3333
IRIS-VIRGINICA	0,0612	0,0000	0,9216	0,3333
Total	1,0000	1,0000	1,0000	1,0000

Discussion des critères d'appréciation

- Il ne suffit pas de faire confiance à la performance en apprentissage car le risque empirique est intrinsèquement optimiste mais n'est pas forcément un bon indicateur de la vraie performance (risque réel).

- Le phénomène de surapprentissage (over-fitting) montre que le risque empirique seul ne peut pas servir de base à l'estimation de la performance de l'apprentissage réalisé. En effet, le risque empirique diminue avec l'accroissement du nombre d'exemples d'apprentissage présentés (ou leur répétition) alors que le risque réel, d'abord décroissant, se met à augmenter après un certain stade

- Une évaluation a priori du risque réel peut être effectuée en fonction du risque empirique en utilisant des calculs théoriques fournissant des bornes en probabilité sur le risque réel en fonction du risque empirique.

Discussion des critères d'appréciation

- La façon la plus naturelle d'estimer le risque réel est d'utiliser un échantillon de test, statistiquement indépendant de celui de l'apprentissage pour faire cette mesure, mais il existe d'autres méthodes.

- La meilleure démarche est d'utiliser trois ensembles d'échantillons (apprentissage, validation et test).

- Il existe plusieurs mesures pour comparer:
  - le même algorithme d'apprentissage sur deux ensembles de test différents
  - deux algorithmes d'apprentissage sur le même ensemble de test
  - deux algorithmes d'apprentissage sur deux ensembles de test différents

- En plus des critères numériques, il existe d'autres critères tels que l'intelligibilité des résultats produits (un petit ensemble de règles extraites compréhensibles est préférable à un fouillis de règles sophistiquées), la simplicité des hypothèses produites...

Directions de recherche actuelles et futures

Le domaine de l'apprentissage automatique avance dans plusieurs directions en explorant plusieurs types de tâches d'apprentissage et en développant la théorie sous-jacente. Voici un échantillon de questions de recherches actuelles :

- Les données non étiquetées peuvent-elles être utiles pour un apprentissage supervisé ?
- Comment transférer ce qui a été appris dans une tâche pour améliorer l'apprentissage d'autres tâches ?
- Quelle est la relation entre les différents algorithmes d'apprentissage, lequel doit être utilisé et à quel moment ?
- Pour les apprenants qui collectent leurs propres données d'apprentissage, quelle est la meilleure stratégie ?
- Jusqu'à quel degré pouvons nous bénéficier conjointement de la sécurité des données et des avantages de la fouille de données ?

D'autres questions restent posées à plus long terme :

- Est-il possible de construire des systèmes d'apprentissage continu ?
- Les théories et algorithmes d'apprentissage automatique peuvent-elles aider à expliquer certains aspects de l'apprentissage humain ?
- Pouvons-nous concevoir des langages de programmation contenant des primitives d'apprentissage automatique ?
- La perception par ordinateur va-t-elle fusionner avec l'apprentissage automatique?

Sources bibliographiques et URL

LIVRES

Mitchell T. "Machine Learning", McGraw-Hill, 1997.
Cornuéjols A., Miclet L. "Apprentissage artificiel : Concept et algorithmes", Eyrolles, 2002
Alpaydin E. "Introduction to machine learning", MIT Press, 2004
Bishop C. M. "Pattern recognition and machine learning", Springer, 2006
Dreyfus G. et al., " Apprentissage statistique", Eyrolles, 2008 (Réseaux de neurones - Cartes topologiques - Machines à vecteurs supports)
Cornuéjols A., Miclet L. "Apprentissage artificiel : Concept et algorithmes", Eyrolles, 2010 (2ème édition)

SITES ET BASES DE DONNEEES

http://cml.ics.uci.edu/
Site du Center for Machine Learning and Intelligent Systems at the University of California, Irvine (UCI), contenant des bases de données, des documents et des liens

http://archive.ics.uci.edu/ml/ UCI Machine Learning Repository

Sources bibliographiques et URL

Conférences et revues
complètement dédiées à l'apprentissage automatique

International Conference on Machine Learning (ICML) : conférence annuelle internationale dominée par les américains

European Conference on Machine Learning (ECML) : conférence annuelle européenne (internationale)

Annual Conference on Learning Theory (COLT).

Conférence francophone d'apprentissage (CAP) conférence annuelle francophone qui a pris la suite des journées françaises d'apprentissage (JFA) depuis 1999

Journal of Machine Learning Research (JMLR). articles téléchargeables gratuitement à l'adresse : www.jmlr.org.

Machine Learning journal publié par Springer.

Université Badji Mokhtar - Annaba
Laboratoire de Recherche en Informatique

جامعة باجي مختار – عنابـــــة
مخبر البحث في الإعـلام الآلـــي

Chapitre 2
Les Arbres de Décision

Cours d'Apprentissage Automatique
Master 1 STIC, 2011-2012

Présenté par
Pr Souici – Meslati L.

Plan

1. Introduction
2. Concepts de base
   - Principe, Exemples, Définition, Représentation formelle, Traduction de la position des nœuds, Représentation textuelle, Extraction de règles, Avantages, Inconvénients
3. Apprentissage des arbres de décision
   - Principe, Exemple illustratif, Algorithme d'apprentissage
4. Elagage des arbres de décision
   - Préélagage, Post-élagage, Algorithme d'élagage
5. Systèmes fondés sur les arbres de décision
6. Arbres de décision flous
7. Exemple de construction d'arbres de décision

Introduction

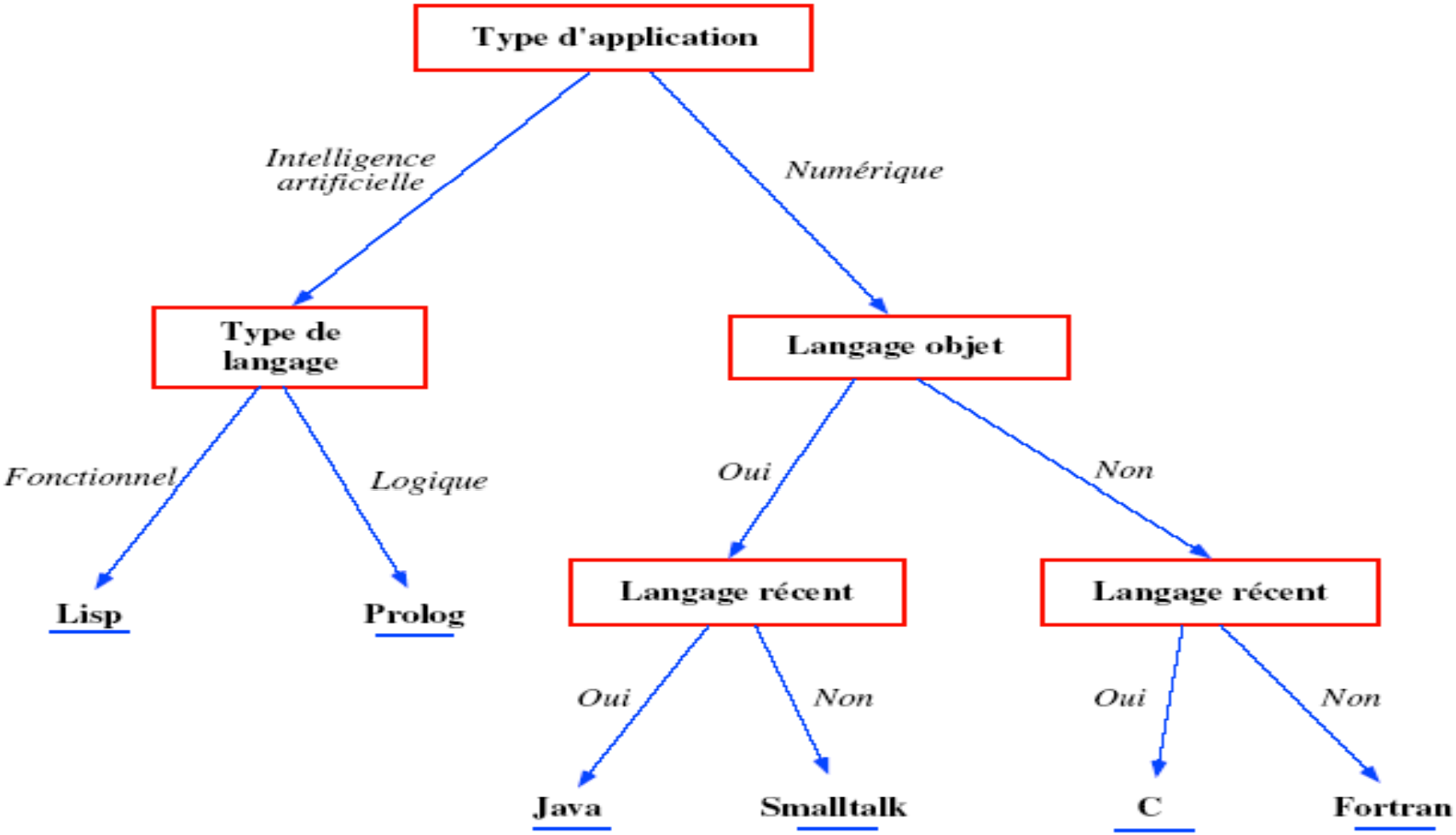
- Les arbres de décision sont l'une des méthodes d'apprentissage automatique symbolique les plus connues dans le domaine de l'intelligence artificielle. Ils sont parmi les techniques de classification les plus anciennes et les plus intuitives.
- Leur modélisation repose sur une représentation (structuration) graphique d'un ensemble de règles, qui les rendent aisément interprétables, rapides, apparemment simples, et donnant souvent des résultats convenables.
- C'est pourquoi cette approche a connu un vif succès dans différents domaines comme la fouille de données, ou encore les systèmes d'expertise (médicale, financière,… etc.) ainsi que l'aide au diagnostic médical où le médecin doit pouvoir interpréter les raisons du diagnostic.

Principe des arbres de décision

- Il s'agit de diviser la population successivement en sous-groupes selon les valeurs prises par des attributs (catégories d'une variable qualitative ou intervalles continus de valeurs pour une variable quantitative) qui, à chaque étape, discrimine le mieux la variable à modéliser (sommet de l'arbre).
- Cette représentation permet donc non seulement de réaliser des prévisions mais aussi de visualiser quelles sont les variables qui discriminent le plus.

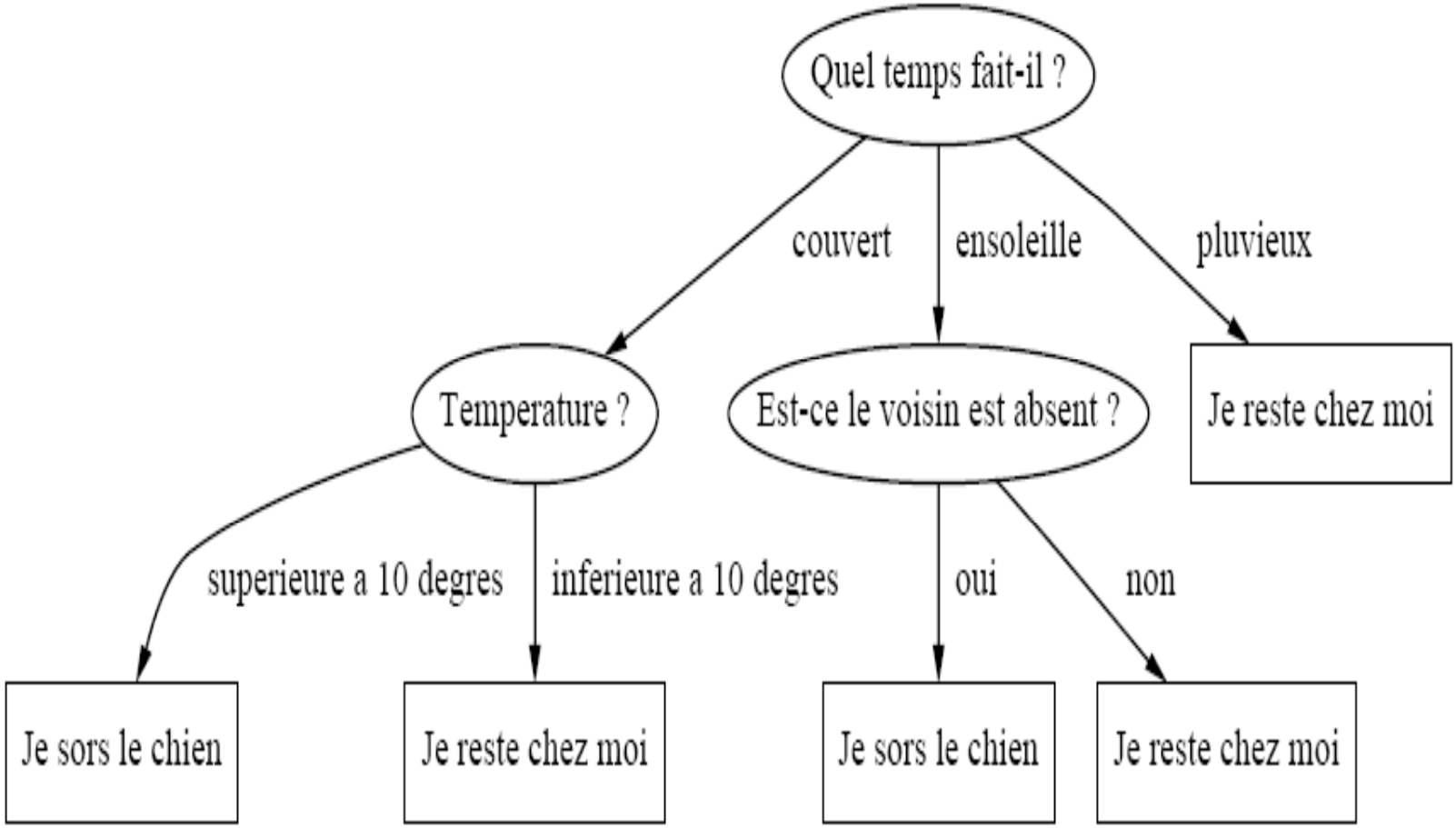
Exemples d'arbre de décision

Langages de programmation



Exemples d'arbre de décision

Promenade du chien



Définition d'un arbre de décision

- Un arbre de décision (decision tree) est un enchaînement hiérarchique de règles logiques, sa structure est construite grâce à des méthodes d'apprentissage **par induction à partir d'exemples**, en cherchant à chaque niveau, l'attribut le plus discriminant pour classifier un exemple. L'arbre ainsi obtenu représente une fonction (qui a une traduction immédiate en terme de règles de décision mutuellement exclusives) qui fait la classification d'exemples, en s'appuyant sur les connaissances induites à partir d'une base d'apprentissage. En raison de cela, ils sont aussi appelés arbres d'induction *(Induction decision tree*s).
- L'objectif des arbres de décision est de mettre en relation un ensemble de connaissances relatives aux propriétés des objets à classer dans l'espace de représentation de façon à pouvoir discriminer les classes auxquelles ils appartiennent. Ces connaissances se présentent sous la forme de conditions sur la valeur des attributs et leurs relations sont décrites par une arborescence facilement interprétable.

Représentation Formelle

- De manière formelle, un arbre de décision est un graphe orienté sans cycle *G = (N, A)* composé d'un ensemble de *noeuds* (sommets) *N* et d'*arcs A*. Les noeuds terminaux, appelés *feuilles* ne possèdent pas de fils. Un *chemin* (ou branche) dans l'arbre est une suite d'arcs qui relie deux sommets. Un noeud non terminal *NId* est étiqueté par un des attributs *Fk*, *k* =1,…, *n* de l'espace de représentation. Les arcs sont, quant à eux, étiquetés par une condition *Fkl* portant sur les valeurs de *Fk* et qui permet de passer du noeud père *Id* au nœud fils *Id.l* si elle est satisfaite. Cette condition représente en général une modalité de l'attribut ou un ensemble de valeurs. Enfin, les feuilles sont étiquetées par une classe $\omega i \in S$ représentant la décision qui doit être prise si une forme satisfait l'ensemble des conditions le long du chemin allant de la racine à la feuille.

Traduction de la position des noeuds

- Chaque nœud *NId* de l'arbre possède un identifiant que nous notons *Id* d'une façon générale. Cet identifiant est en fait une séquence d'indices repérant la position du noeud dans l'arbre. Chaque arc *(NId, NId.l)* est orienté et relie un noeud *père NId* et un noeud *fils NId.l* où *Id.l* est la concaténation (.) de l'identifiant *Id* du *père* avec l'indice *l* du *fils* (son numéro parmi l'ensemble des *fils* du noeud *père*). Chaque noeud possède exactement un père à l'exception de la racine de l'arbre qui n'en possède aucun. Son identifiant est 1 et elle est notée *N1*.
- Si on considère l'arbre de la promenade du chien, la traduction des positions de ces nœuds sera :
  - *N1* : étiqueté par le test : Quel temps fait-il?
  - *N1.1* : étiqueté par le test : Température?
  - *N1.1.1* : étiqueté par : Je sors le chien

Représentation textuelle

- L'arbre de décision peut être affiché sous forme textuelle en effectuant un décalage vers la droite pour chaque « niveau » de l'arbre, par exemple l'arbre de la promenade du chien devra être affiché textuellement sous la forme suivante :

| Quel temps fait-il? = Couvert :
| | Température? = >10 degrés : Je sors le chien
| | Température? = <10 degrés : Je reste chez moi

| Quel temps fait-il? = Ensoleillé :
| | Est-ce que le voisin est absent? = oui : Je sors le chien
| | Est-ce que le voisin est absent? = non : Je reste chez moi

| Quel temps fait-il? = Pluvieux : Je reste chez moi

Extraction de règles

- Grâce à la structuration des connaissances sous forme d'arbre, l'interprétation, sous forme de règles est rendue relativement aisée et améliore considérablement la lisibilité et la flexibilité. Ainsi sur l'exemple de la promenade du chien, on peut extraire des règles telles que :
  - **Si** (le temps est pluvieux) **alors** je reste chez moi
  - **Si** (le temps est couvert et que la température est supérieure à 10 degrés) **alors** je sors le chien
  - **Si** (le temps est couvert et que la température est inférieure à 10 degrés) **alors** je reste chez moi
- Ce type de structure a été utilisé dans de nombreux domaines applicatifs où son interprétabilité ainsi que son habileté à manipuler des attributs à la fois numériques et symboliques en a fait une approche privilégiée par rapport à d'autres approches de classification. Comme la construction d'un arbre de décision permet la génération de systèmes à base de règles, il nous faut faire le lien avec l'approche Systèmes Experts. Les deux approches à partir de données et par expertise peuvent être considérées comme concurrentes et complémentaires.

Avantages des arbres de décision

- Leur lisibilité, car ils peuvent être traduits sous forme de règles
- Plusieurs domaines d'application ont montré l'intérêt de l'utilisation des arbres de décision: la prédiction, la classification d'images, la reconnaissance de caractères manuscrits…
- L'expressivité des arbres de décision leur permet l'approximation de fonctions à valeurs discrètes et les rend capables d'apprendre des expressions disjonctives.
- Le nombre moyen de tests à effectuer sur une forme peut être réduit (si les *d* attributs sont tous binaires, ce nombre est limité par *d*).
- La structure de décision est globale à toutes les classes: on n'a pas de problème pour traiter directement C classes.
- Il n'est pas nécessaire de tester tous les attributs de chaque objet à chaque nœud, dans la plupart des cas pratiques, on se limite même à un seul (sélecteur).
- Il existe actuellement des solutions pour les problèmes d'apprentissage par les arbres de décision tels que le sur-apprentissage.
  - Arrêter la croissance de l'arbre quand la division des données n'est plus statistiquement significative ou de générer l'arbre entier, puis élaguer.
  - Générer plusieurs arbres et de sélectionner le meilleur d'entre eux en mesurant les performances sur un ensemble distinct de données de validation, en mesurant les performances sur l'ensemble d'apprentissage et en effectuant des tests statistiques, ou encore en minimisant la taille de l'arbre ainsi que ses erreurs de classification.

Inconvénients des arbres de décision

- La difficulté de manipulation de variables quantitatives (valeurs continues). La discrétisation des données reste une solution à considérer, mais elle ne résout pas tous les problèmes de traitement d'informations non symboliques.
- Le choix des attributs à considérer dans les divisions (noeuds de l'arbre) ne garantit pas une solution toujours parfaite. De plus, la méthode la plus utilisée, fondée sur la théorie de l'information de Shannon, impose de fortes contraintes aux méthodes incrémentales (on est obligé de tout reconstruire).
- Les arbres de décision n'utilisent pas de connaissances théoriques disponibles sur le problème posé. Ils exploitent uniquement des connaissances empiriques.
- Les arbres de décision ont tendance à être instables, c'est à dire que de petites fluctuations dans la base d'exemples utilisée peuvent considérablement modifier la topologie de l'arborescence.

Apprentissage des arbres de décision
Principe

- Traditionnellement, un arbre de décision se construit à partir d'un ensemble d'apprentissage par raffinements de sa structure.
- Un ensemble de questions sur les attributs est construit afin de partitionner l'ensemble d'apprentissage en sous-ensembles qui deviennent de plus en plus petits jusqu'à ne contenir à la fin que des observations relatives à une seule classe.
- Les résultats des tests forment les branches de l'arbre et chaque sous-ensemble en forme les feuilles.
- Le classement d'un nouvel exemple se fait en parcourant un chemin qui part de la racine pour aboutir à une feuille : l'exemple appartient à la classe qui correspond aux exemples de la feuille.

Apprentissage des arbres de décision
Exemple illustratif

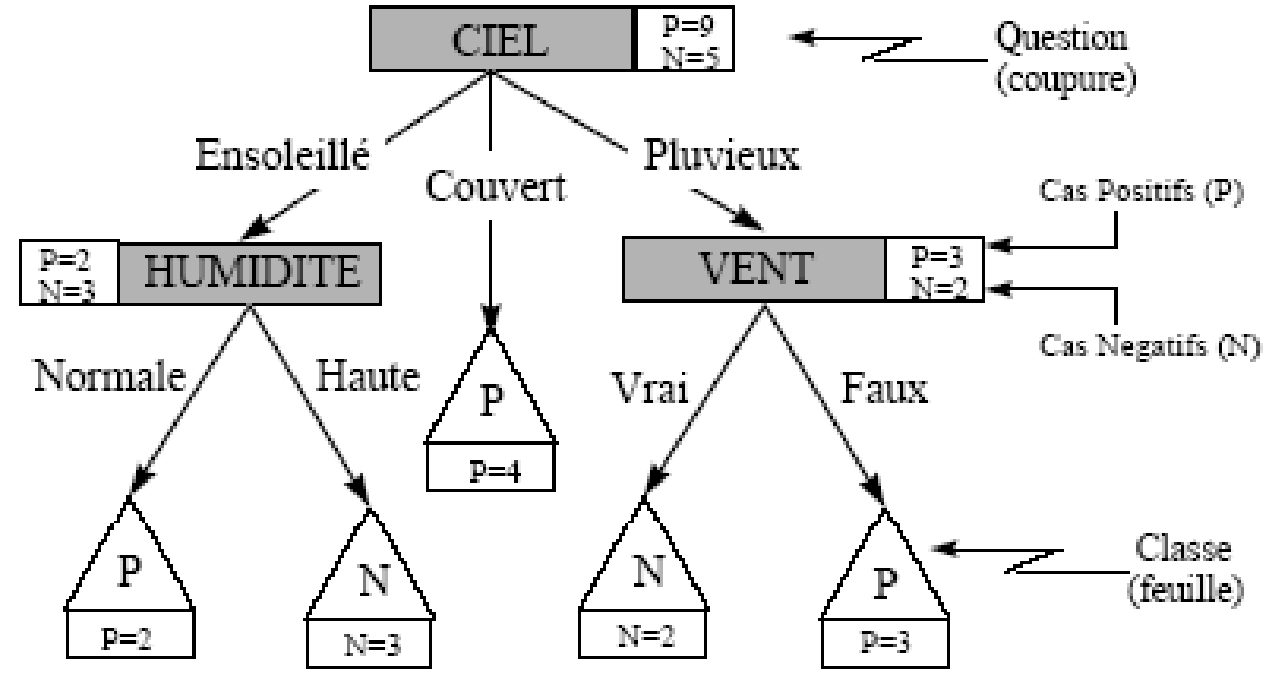
- Soit une base d'exemples de conditions météorologiques qui possèdent un certain nombre d'attributs significatifs, e.g. la température, le vent, l'humidité, etc. Chaque exemple associe des valeurs particulières à chaque attribut, et comme cette méthode est une *méthode d'apprentissage supervisé*, chaque exemple est associé à une classe particulière.On veut construire un arbre de décision pour le classement d'un ensemble de cas. Les cas dits *positifs* sont ceux qui appartiennent à la classe et les cas dits *négatifs* sont ceux qui n'y appartiennent pas.

NUMERO	*CIEL*	*TEMPERATURE*	*HUMIDITE*	*VENT*	*CLASSE*
1	ensoleillé	élevé	forte	non	N
2	ensoleillé	élevé	forte	oui	N
3	couvert	élevé	forte	non	P
4	pluvieux	moyenne	forte	non	P
5	pluvieux	basse	normale	non	P
6	pluvieux	basse	normale	oui	N
7	couvert	basse	normale	oui	P
8	ensoleillé	moyenne	forte	non	N
9	ensoleillé	basse	normale	non	P
10	pluvieux	moyenne	normale	non	P
11	ensoleillé	moyenne	normale	oui	P
12	couvert	moyenne	forte	oui	P
13	couvert	élevé	normale	non	P
14	pluvieux	moyenne	forte	oui	N

Apprentissage des arbres de décision
Exemple illustratif

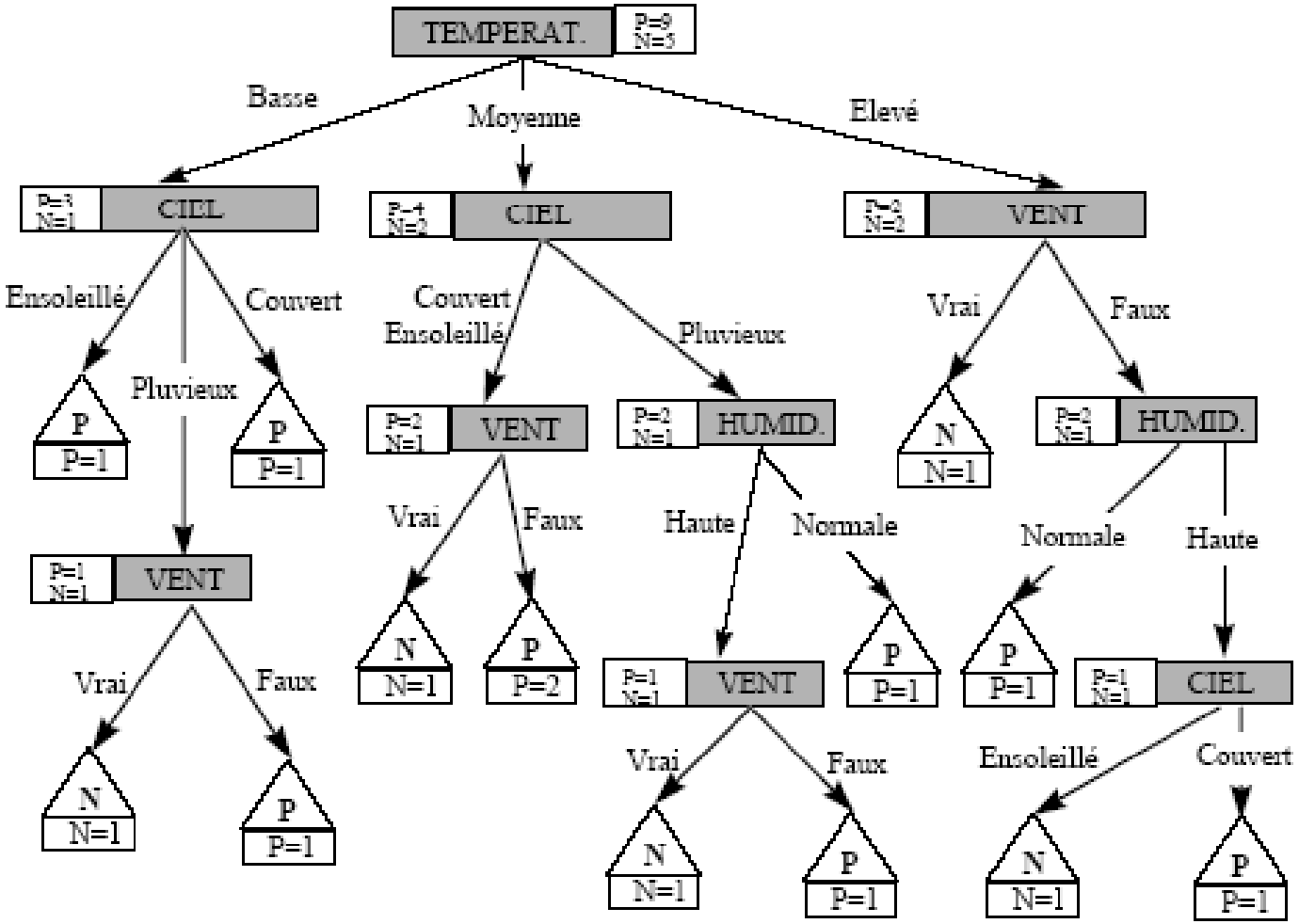
- Le principe de construction des arbres est le suivant : on choisit un attribut parmi les attributs non sélectionnés, et on crée un noeud portant un test sur cet attribut.
- Pour chaque classe d'équivalence ainsi induite, on opère le traitement suivant : si tous les exemples de cette classe d'équivalence appartiennent à la même classe (les classes météorologiques décrites dans le tableau précédent), alors on crée une feuille correspondante à cette classe, reliée au test précédent par un arc étiqueté par la valeur de l'attribut correspondant.
- Si tous les exemples de la classe d'équivalence considérée ne sont pas dans la même classe, alors on réitère ce processus en enlevant l'attribut précédemment considéré des attributs à sélectionner.
- Par exemple, on peut construire l'arbre de la manière suivante : le premier attribut sélectionné est le 'ciel'; cet attribut sépare la base en trois ensembles : "ciel ensoleillé", "ciel pluvieux" et "ciel couvert". Ensuite, pour les exemples qui ont l'attribut "ciel ensoleillé", on prend un autre attribut qui va permettre de les distinguer, par exemple l'humidité. Une fois que tous les exemples qui ont l'attribut "humidité forte" appartiennent à la même classe et qu'il en est de même avec ceux qui ont l'attribut "humidité normale", on peut arrêter le processus de division des branches de l'arbre à ce niveau. Il faut alors terminer la construction de l'arbre pour les autres branches qui n'ont pas encore été traitées jusqu'à la fin.

Apprentissage des arbres de décision
Exemple illustratif



IF ((CIEL=Ensoleillé *Et* HUMIDITE=Normale) *Ou*
(CIEL=Couvert) *Ou*
(CIEL=Pluvieux et VENT=Faux))
ALORS Classe = P

Apprentissage des arbres de décision
Exemple illustratif



Apprentissage des arbres de décision

- On peut remarquer que l'ordre dans lequel ces attributs sont sélectionnés est primordial pour la construction de l'arbre. On peut avoir différents arbres résultant de l'apprentissage d'une même base d'exemple, comme le montre les figures précédentes. En effet, si l'on commence par prendre la 'température', alors l'arbre considéré sera plus grand que si on prend 'Ciel'.
- Plus généralement, si on suppose que le nombre d'attributs est ***d*** et que ***a*** est le nombre moyen de valeurs possibles par attributs, le nombre d'arbres possibles sera:

$$\sum_{i=0}^{d-1} (d-i)^{a^i}$$

- Pour 4 attributs à 3 valeurs chacun, cela donne 544 arbres possibles. Il faut donc un moyen pour sélectionner l'attribut à tester à un point donné de la construction de l'arbre.

Apprentissage des arbres de décision

- On suppose que les arbres qui généralisent le mieux sont les arbres de petite taille, puisque ils vont tout d'abord sélectionner les décisions plus importantes et qui séparent le plus grand nombre de cas différents.
- De plus, la préférence pour la construction d'arbres plus simples est une application du principe du "rasoir d'Occam"
- Quinlan (1986) a proposé une méthode pour traiter ce problème du choix de l'attribut. Il utilise une technique fondée sur la théorie de l'information de Shannon. L'idée consiste à appliquer une méthode qui permet de maximiser le gain d'information apporté par chaque test.

Apprentissage des arbres de décision
Algorithme (attributs binaires)

```
Procedure CONSTRUIRE-ARBRE(X)
    début
        si tous les points de X sont de la même classe
        alors créer une feuille de cette classe
        sinon
          choisir le meilleur attribut pour créer un nœud
          le test associé sépare X en Xd et Xg ;
          CONSTRUIRE-ARBRE(Xd)
          CONSTRUIRE-ARBRE(Xg)
        fin si
    fin
```

Apprentissage des arbres de décision
Algorithme (cas général)

La construction de l'arbre s'effectue à partir d'une base d'apprentissage *Bapp* initiale. L'objectif est d'extraire de cette base les connaissances pertinentes permettant de partitionner la base d'apprentissage en sous-ensembles d'individus représentatifs d'une unique classe opérant ainsi une discrimination des classes. Les partitionnements s'effectuent de manière récursive sur une base d'exemples *B* (*B = Bapp* au départ) à partir de conditions sur les valeurs des attributs *Fk, k=1,…, n* de l'espace de représentation des formes. Ces attributs prennent un ensemble de valeurs (ou modalités), *Fkl, l= 1…., Lk.* L'algorithme d'apprentissage se résume alors de la manière suivante:

Apprentissage des arbres de décision
Algorithme (cas général)

INITIALISATION :
B = Bapp.
CONSTRUIRE ARBRE *(B)*
DEBUT
    SI le critère d'arrêt est vérifié
    ALORS créer une feuille
    SINON
      Choisir un attribut Fk et créer un nœud
      Partitionner *B* en *Lk* sous- ensembles *Bkl, l = 1,..., Lk*
    Pour *l =1, ..., Lk* CONSTRUIRE- ARBRE (*Bkl*)
    FINSI
FIN

Apprentissage des arbres de décision
Algorithme (cas général)

Cet algorithme récursif est caractérisé par quatre notions fondamentales qui permettent de distinguer les différentes approches :

- Le choix d'un attribut pour partitionner B.
- Une stratégie de partitionnement.
- Un critère d'arrêt.
- La création d'une feuille.

Algorithme d'apprentissage
Choix d'un attribut pour partitionner (1)

- Lors de la construction d'un noeud non terminal, il faut déterminer l'attribut qui discrimine le mieux la base courante B d'individus. Rappelons que les connaissances discriminantes ne sont pas toutes pertinentes puisqu'elles sont caractérisées par leur pouvoir de discrimination.
- De plus, cette pertinence est fortement dépendante du contexte dans lequel elle est évaluée. Choisir les attributs les plus discriminants localement à chaque noeud permet donc d'obtenir une modélisation plus robuste mais aussi plus compacte et lisible.
- Pour évaluer la pertinence d'un attribut relativement à la distribution en classes d'un ensemble d'individus on utilise le plus souvent une mesure de discrimination. De nombreux travaux ont été effectués sur l'élaboration de ces mesures et les propriétés qu'elles doivent posséder la mesure d'impureté de Gini ou encore le critère de χ^2.
- La mesure la plus utilisée est probablement ***l'entropie*** de Shannon (proposée en 1948) qui repose sur les principes de la théorie de l'information.

Algorithme d'apprentissage

Choix d'un attribut pour partitionner (2)

- S = {ω1,…… ωi,…, ωs} est l'ensemble des classes
- B désigne un ensemble de N individus à partitionner au niveau d'un nœud.
- Parmi ces échantillons, Ni appartiennent à la classe ωi et la probabilité associée est :

$$p(\omega i) = Ni/N$$

- L'information relativement à la distribution en classes S sur B s'écrit:

$$H_B(S) = - \Sigma_{(i=1..s)} p(\omega i) \log_2 p(\omega i)$$

Algorithme d'apprentissage
Choix d'un attribut pour partitionner (3)

- Considérons maintenant un attribut Fk prenant les modalités *Fkl, l = 1..*L_k.
- Si cet attribut est choisi pour partitionner B, les N exemples sont partagés en L_k sous- ensembles *Bkl, l = 1,…,* L_k suivant la modalité à laquelle ils appartiennent. Les effectifs associés aux *Bkl* sont notés N_{kl}. Parmi eux N^i_{kl} appartiennent à la classe ωi. L'entropie associée à cette division relativement aux classes est appelée *entropie conditionnelle* :

$$H_B(S|F_k) = -\Sigma_{(i=1..Lk)} p(F_{kl}) \Sigma_{(i=1...s)} p(\omega i|F_{kl}) \log_2 p(\omega i|F_{kl})$$

$$= \Sigma_{(l=1...Lk)} p(F_{kl}) H_{Bkl}(S)$$

Avec : $p(\omega i \mid F_{kl}) = N^i_{kl} / N_{kl}$ et $p(F_{kl}) = N_{kl} / N$

- Prendre l'attribut *Fk* le plus discriminant revient alors à choisir celui qui maximise **le gain d'information:**

$$GB(F_k) = H_B(S) - H_B(S|F_k)$$

Algorithme d'apprentissage
Stratégie de partitionnement (1)

- Les partitionnements s'effectuent dans l'espace de représentation suivant les différentes modalités des attributs. Il existe alors différentes possibilités pour effectuer un partitionnement au niveau d'un noeud selon que les attributs soient nominaux ou continus.

Le cas nominal

- Dans le cas des attributs binaires, le test consiste à descendre dans un sous arbre si le test sur l'attribut choisi vaut VRAI, dans l'autre quand il vaut FAUX.
- Le cas où les attributs sont à valeurs discrètes se généralise facilement quand le test que l'on construit se réduit à opposer une valeur à toutes les autres, on est alors ramené au cas binaire. Par exemple, s'il existe un attribut couleur prenant ses valeurs dans l'ensemble {bleu, rouge, vert}, il est simple de l'éclater en trois attributs binaires, du type couleur-rouge, couleur-bleu couleur-vert qui est VRAI ou FAUX sur chaque donnée d'apprentissage.

Algorithme d'apprentissage
Stratégie de partitionnement (2)

- On se replace alors dans le cas exposé ci- dessus, avec la transformation d'un attribut nominal à k valeurs possibles en k attributs binaires que l'on traite indépendamment. La deuxième méthode, utilisée quand le nombre d'attributs est réduit, consiste à dériver un attribut pour chaque paire de modalités possible ({rouge, vert} {rouge, bleu} {vert, bleu} dans le cas de notre exemple).
- Cette technique a l'inconvénient d'oublier la signification globale de l'attribut ; en effet, si couleur- rouge est VRAI pour un attribut, couleur-bleu est automatiquement FAUX, mais cette propriété n'apparaît plus explicitement dans les données. Une autre solution est alors de calculer directement l'information mutuelle entre les deux variables à valeurs discrètes que sont d'une part cet attribut et d'autre part l'ensemble des classes. Si celle-ci se révèle la meilleure pour tous les attributs, on crée alors un nœud non binaire dans l'arbre de décision (dans l'exemple précédent, le test de l'attribut « couleur » donne quatre réponses possibles). Le seul inconvénient est qu'il faut gérer une structure de données plus complexe

Algorithme d'apprentissage
Stratégie de partitionnement (3)

Le cas continu

- Traiter un attribut continu peut paraître plus difficile, mais en pratique ce n'est pas fondamentalement différent, car le nombre de données d'apprentissage est fini, le nombre des valeurs que prend cet attribut sur les exemples est aussi fini. Mieux, ses valeurs sont ordonnées, contrairement au cas nominal. Le sélecteur consistera donc à comparer les valeurs à un seuil pour construire un nœud binaire.
- Pour un attribut ***a*** continu, on procède alors ainsi : on trie les points d'apprentissage selon la valeur de cet attribut, puis on cherche la seuil s(*a*) qui minimise l'un des critères précédents.
- Il est à noter que l'arbre de décision est le seul modèle permettant de gérer de manière homogène les attributs de natures variées, en particulier les mélanges continus et binaires.

Algorithme d'apprentissage
Critère d'arrêt (1)

- Le critère d'arrêt est un point essentiel de la procédure de construction de l'arbre car il détermine en grande partie son efficacité tant au niveau de sa compacité que de ces capacités de généralisation. Une première idée simple consiste à grossir l'arbre (en réitérant la procédure CONSTRUIRE-ARBRE) tant que les noeuds ne sont pas purs, c'est à dire tant qu'ils contiennent des exemples de classes différentes. L'arbre de décision ainsi obtenu décrit exactement l'ensemble d'apprentissage et ne fait aucune erreur de classification sur cette base. Cette approche possède deux inconvénients majeurs : d'une part les capacités de généralisation de l'arbre de décision seront très souvent mauvaises conduisant à un sur-apprentissage, d'autre part, la taille de l'arbre risque de devenir très importante si les classes sont difficilement séparables. L'impact sur la taille de l'arbre dépend en général du nombre d'exemples dans la base d'apprentissage, plus le nombre d'individus utilisé en apprentissage est important, plus l'arbre risque d'être de taille importante. Ce phénomène est accru par la présence de bruit dans les données. Outre les problèmes d'utilisation de ressources mémoire et CPU que cela suscite, un arbre trop grand est difficilement interprétable ce qui n'est pas souhaitable en général.

Algorithme d'apprentissage
Critère d'arrêt (2)

- Pour limiter ces effets néfastes, d'autres critères d'arrêt sont introduits dans la procédure de construction. On distingue principalement deux approches: l'arrêt « précoce » et « l'élagage ».
- La première agit pendant la construction de l'arbre tout comme le critère d'arrêt sur la pureté d'un nœud. Classiquement une feuille est construite lorsque le nombre d'échantillons de la base B d'un noeud est trop faible ou encore lorsque la pureté de ce sous-ensemble d'individus est jugée suffisante. Cette estimation peut se déduire directement de la mesure de discrimination utilisée pour les partitionnements. Cependant, ces mesures sont calculées localement à un noeud et sont souvent fortement dépendantes du nombre d'échantillons. Il est donc difficile d'établir un seuil unique qui puisse être valable pour tous les noeuds de l'arbre. La construction de l'arbre risquerait alors de s'arrêter prématurément pour certains noeuds ou, au contraire, tardivement pour d'autres. D'autres variantes ont été conçues pour pallier à ces problèmes comme par exemple une mesure d'impureté qui se calcule sur l'arbre complet.

Algorithme d'apprentissage
Critère d'arrêt (3)

- Au lieu d'arrêter la construction de l'arbre de manière « précoce », certains chercheurs suggèrent de construire l'arbre de décision de taille maximale puis de réduire sa taille dans une seconde phase par une méthode d'élagage. Le principe général consiste à supprimer des noeuds ou des sous-arbres peu représentatifs, ou encore à les regrouper. De nombreuses techniques ont été étudiées montrant pour la plupart une efficacité réelle par rapport à l'arbre non élagué. Certaines de ces techniques procèdent en deux étapes : la première consiste à construire une série d'arbres de plus en plus petits à partir de la base d'apprentissage en supprimant les noeuds en fonction d'un critère basé sur le rapport entre le taux d'erreur de l'arbre sur la base d'apprentissage et sa taille *(cost complexity*). La seconde utilise une base de validation pour choisir l'arbre qui a le taux d'erreur le plus faible sur cette base.

Algorithme d'apprentissage

Création d'une feuille

- Une feuille diffère d'un noeud non terminal par le fait qu'elle marque l'arrêt de l'expansion d'une branche parce qu'un des critères d'arrêt a été satisfait. La feuille signifie alors que l'on peut (doit) être capable d'identifier la classe la plus représentative dans la sous-base B locale. Lorsque cette sous-base est pure, il n'y a qu'une seule classe possible et c'est elle qui étiquette la feuille. C'est le cas idéal, mais il est rarement satisfait en pratique. Dans tous les autres cas, la règle générale consiste à étiqueter la feuille avec la classe possédant le plus de représentants.

Elagage des arbres de décision

- La poursuite de l'algorithme de construction jusqu'à son terme naturel produit un arbre Tmax avec des feuilles correspondant à des classes parfaitement homogènes ; il y a là un apprentissage par coeur et donc un risque de sous-estimation de la probabilité d'erreur par le taux d'erreur apparent (qui vaut 0).
- L'élagage consiste à minimiser la taille de l'arbre de décision en remplaçant un sous arbre par une feuille ou en regroupant plusieurs classes en une seule. Il peut se faire de deux manières: soit pendant la construction de l'arbre (préélagage), soit à posteriori, une fois que l'arbre aura été entièrement développé (post-élagage).

Le préélagage

- Il est effectué au fur et à mesure de la construction de l'arbre de décision où on cesse de diviser un noeud quand la pureté des points qu'il domine et non pas parfaite mais suffisante (si l'entropie calculée au niveau de ce noeud est inférieure à certain seuil), on le considère alors comme une feuille et on lui attribue la classe en question. Cette méthode ne prend en compte qu'un critère local à la feuille examinée et peut de ce fait manquer un développement intéressant de l'arbre.

Elagage des arbres de décision

Le post-élagage

- Une autre technique, plus valide théoriquement et plus efficace en pratique, consiste d'abord à construire l'arbre de décision complètement, puis seulement après à chercher à le simplifier en l'élaguant progressivement en remontant des feuilles vers la racine.
- Pour juger quand il est bon d'arrêter d'élaguer l'arbre, on utilise un critère de qualité qui exprime souvent un compromis entre l'erreur commise par l'arbre et une mesure de complexité. L'erreur commise est mesurée grâce à un ensemble de validation. On supposera donc dans ce paragraphe que l'ensemble d'apprentissage est assez important pour être coupé en deux parties: l'une (ensemble d'apprentissage proprement dit) pour construire l'arbre de décision Tmax, l'autre (ensemble de validation) pour choisir le meilleur parmi les élagages proposés.
- L'algorithme optimal consisterait à calculer le taux d'erreur de l'ensemble de validation sur tous les arbres qu'il est possible d'obtenir par élagage de Tmax. Mais leur nombre croit très rapidement avec la taille de Tmax, mesurée en nombre de nœuds. On utilise donc des solutions sous-optimales, dont la plus classique (un algorithme *glouton*) consiste à construire, sans retour en arrière, une séquence d'arbres par élagages successifs, en remontant des feuilles vers la racine. Cette séquence se note S= (Tmax,T1,…,Tk,…..,Tn). Tn est l'arbre constitué d'une seule feuille comprenant les m points d'apprentissage. C'est donc l'arbre élagué au maximum. Pour passer de Tk à Tk+1, il faut transformer un nœud dans Tk en feuille.

Elagage des arbres de décision

Le post-élagage

- Pour savoir si cet élagage serait bénéfique, l'idée générale est de comparer le «coût» de l'arbre élagué et celui de l'arbre non élagué, et d'arrêter l'élagage quand le coût du premier dépasse le coût du second. Pour évaluer ce coût, plusieurs critères ont été proposés qui prennent tous en compte à la fois l'erreur commise par l'arbre et une mesure de sa complexité.
- Nous examinons ici le critère consistant à choisir le nœud v qui minimise sur l'ensemble des nœuds de Tk la valeur suivante:

$$W(Tk,v) = (MCéla(v,k) - MC(v,k)) / (n(k) * (nt(v,k) - 1))$$

MCéla(v,k) est le nombre d'exemples de l'ensemble d'apprentissage mal classés par le nœud v de Tk dans l'arbre élagué à v.

MC (v,k) est le nombre d'exemples de l'ensemble d'apprentissage mal classés sous le nœud v dans l'arbre non élagué.

N(k) est le nombre de feuilles de Tk.

Nt (v,k) est le nombre de feuilles du sous- arbre de Tk situé sous le nœud v.

- Ce critère permet donc d'élaguer un nœud de Tk de façon à ce que Tk+1, l'arbre obtenu, possède le meilleur compromis entre taille et taux d'erreur apparent. Finalement, la suite S=(Tmax,T1,...,Tk,.....,Tn) possède un élément Tk0 pour lequel le nombre d'erreurs commises est minimal sur l'ensemble de validation: c'est cet arbre-là qui sera finalement retenu par la procédure d'élagage.

Elagage des arbres de décision

Algorithme d'élagage

Algorithme d'élagage d'un arbre de décision:

Procédure: élaguer (Tmax)

k←0

Tk←Tmax

Tant que Tk a plus d'un nœud faire

Pour chaque nœud v de Tk faire

Calculer le critère *W (Tk,v)* sur l'ensemble d'apprentissage

Fin pour

Choisir le nœud vm pour lequel le critère est maximum

Tk+1 se déduit de Tk en y remplaçant vm par une feuille

k←k+1

fin tant que

Dans l'ensemble des arbres {Tmax,T1,…..,Tk,….,Tn}, choisir celui qui à la plus petite erreur de classification sur l'ensemble de validation.

Systèmes fondés sur les arbres de décision

- Parmi les méthodes d'apprentissage fondées sur les arbres de décision les plus connues, on trouve l'algorithme ID3 proposé par Quinlan en 1979 et la méthode CART. Ces systèmes sont assez semblables, ils ont été développés par deux groupes de recherche séparés et presque à la même époque. La principale différence entre ces deux systèmes réside dans le choix de la mesure utilisée pour la sélection des attributs pendant la construction de l'arbre. Cette mesure est généralement fondée sur la théorie de l'information de Shannon (entropie et gain d'information, utilisés dans ID3).
- La méthode ID3 est à l'origine de plusieurs autres systèmes. L'un des plus connus est le système C4.5, développé plus récemment par Quinlan (en 1993). A l'origine du système C4.5 se trouve la volonté de résoudre les différents problèmes rencontrés dans son prédécesseur, l'algorithme ID3.

Systèmes fondés sur les arbres de décision

Ces problèmes sont assez caractéristiques des arbres de décision en général, à savoir:

- A) Les premiers systèmes ont été conçus pour traiter des attributs avec des valeurs nominales et discrètes (*variables qualitatives*) et ne pouvaient pas traiter des attributs avec des valeurs continues (*variables quantitatives*). Il manquait des méthodes bien adaptées pour trouver les bonnes questions à poser sur les valeurs continues et qui seront postérieurement associées aux noeuds de l'arbre.
- B) Le choix du meilleur attribut qui va diviser les exemples présente lui aussi certains problèmes. Comment peut-on nous assurer que la méthode employée va nous amener à la construction de l'arbre le plus simple possible ? Malheureusement, on ne peut pas être sûr de l'obtenir, puisque ce type de problème devient intraitable d'un point de vue calculatoire . De plus, il n'est pas difficile de trouver des exemples pour lesquels une variable a un très grand pouvoir de discrimination, mais n'aide pas beaucoup à la solution du problème (exp: l'âge ou le sexe d'une personne permet de discriminer les gens d'une façon très efficace, toutefois est-elle une bonne variable par rapport à tous les types de problèmes de classification?).

Systèmes fondés sur les arbres de décision

- C) Parfois, on ne connaît pas toutes les valeurs de chacun des attributs considérés pour la classification. Le fait que l'on doive travailler sur des domaines représentés par des informations incomplètes ouvre la problématique du traitement des *attributs avec valeurs manquantes.*
- D) Les arbres de décision souffrent du problème de surapprentissage (*overtrainning/ovefitting*). Le fait que la construction de l'arbre de décision ne s'arrête que lorsque tous les exemples de la feuille appartiennent à une même classe, suppose des ensembles d'apprentissage contenant uniquement des exemples "parfaits" (ils doivent être corrects et doivent bien représenter tout l'espace des entrées). L'arbre de décision finit par trop se spécialiser dans les exemples d'apprentissage, ce qui peut poser de graves problèmes au niveau de la généralisation aux nouveaux cas. Souvent, les bases d'apprentissage sont incomplètes et ont des exemples incorrects, et donc, cette méthode d'apprentissage ne fonctionnera pas correctement ;

Systèmes fondés sur les arbres de décision

- E) Sur un arbre de décision, les connaissances restent "cryptées". Bien qu'il s'agisse d'une représentation symbolique des connaissances, elle peut être parfois un peu difficile à lire et à interpréter.
- F) Les arbres de décision, tels que ceux construits par ID3, doivent être complètement reconstruits pour prendre en compte un nouvel exemple ajouté à la base d'apprentissage. Ce type d'arbre n'est pas incrémental du point de vue de l'acquisition de données, même si sa structure est construite d'une façon incrémentale (par ajout de noeuds). Il faut recommencer tout l'apprentissage pour prendre en compte un nouvel exemple.

Systèmes fondés sur les arbres de décision

- G) Malgré le fait que ces méthodes cherchent à construire des arbres simples, certaines branches peuvent être répliquées. La structure en arbre n'est pas toujours la forme la plus simple et économique pour représenter les informations. Les graphes de décision restent une alternative à la structuration en forme d'arbres.
- H) Enfin, ces arbres de décision n'exploitent pas de méthodes permettant l'utilisation des connaissances théoriques disponibles sur le problème. Dans leur forme initiale, les arbres d'induction travaillent uniquement sur les connaissances empiriques.

Systèmes fondés sur les arbres de décision

Plusieurs systèmes fondés sur les arbres de décision ont pris en compte ces différentes remarques décrites ci-dessus, en essayant d'apporter des solutions plus performantes:

- C4.5: c'est un système dérivé de l'ID3. Il présente des propositions pour traiter et améliorer les items A (discrétisation des variables quantitatives), B (prise en compte des coûts associés au choix de chaque attribut), C (prise en compte par la fonction de sélection d'attributs), D (élagage de l'arbre à la fin du processus d'apprentissage, à partir d'un ensemble d'exemples de test de généralisation) et E (explicitation de règles symboliques du type Si-Alors à partir des arbres de décision). De plus Ross Quinlan, le concepteur de ID3 et C4.5 propose, à travers sa compagnie Rulequest, plusieurs versions récentes de C4.5, comme C5 sous Unix et See5 sous Windows (consulter le site **www.rulequest.com**)
- Assistant Professional il utilise une autre fonction de choix d'attributs qui permet l'élagage de l'arbre de décision (*post-pruning*), et implémente une méthode de binarisation des attributs avec des valeurs continues.

Systèmes fondés sur les arbres de décision

- ID5R : Il s'agit d'une méthode de construction incrémentale d'arbres de décision qui permet d'apprendre des nouveaux exemples sans avoir besoin de recommencer tout l'apprentissage.
- ITI: système d'apprentissage incrémental d'arbres de décision, développé à partir de l'ID5R. Il apporte des améliorations par rapport à l'utilisation de variables continues et de valeurs manquantes, et il permet aussi d'élaguer les arbres.
- IDL: il s'agit d'un algorithme de construction incrémentale d'arbres de décision fondé sur la méthode ID5R.
- PRISM: méthode d'apprentissage fondée sur l'algorithme ID3 de Quinlan. Il se distingue d'ID3 par le type de méthode employée pour la sélection des attributs discriminants.
- SIPINA: système de construction de graphes de décision. Il utilise une fonction particulière de sélection d'attributs, implémente une méthode de discrétisation de variables continues ainsi qu'une méthode qui évite le sur-appentissage et permet aussi l'explicitation de règles représentées dans les graphes de décision.

Arbres de décision flous

- L'inconvénient principal des arbres de décision tels qu'ils ont été présentés jusqu'ici provient de la façon dont les conditions sur les attributs sont utilisées pour faire les partitionnements.
- Ces conditions sont strictes, ne laissant aucune alternative possible pour attribuer un individu à un des fils du noeud considéré. Les partitionnements résultants peuvent ainsi devenir arbitraires, rendant la modélisation moins robuste voire incohérente (et donc difficilement exploitable). Ce phénomène est encore accru si les données manipulées sont imprécises.
- Les arbres de décision flous apportent alors une solution intéressante à ces différents problèmes
- Afin d'améliorer la robustesse des arbres de décision classiques certains chercheurs ont modifié le mode de représentation de leurs arbres afin d'utiliser des conditions plus souples sur les attributs numériques. Les arbres de décision flous sont alors une extension directe reposant sur le formalisme bien établi de la théorie des sous-ensembles flous.

Arbres de décision flous

- L'idée principale consiste à décrire les attributs par des sous-ensembles flous appelées modalités floues et à utiliser celles-ci dans les conditions permettant de faire les partitionnements. Au niveau de la structure de l'arbre, les arcs sont alors étiquetés par les sous-ensembles flous (termes linguistiques) correspondants.
- Les méthodes liées à l'apprentissage des arbres de décision flous peuvent être considérées comme une extension de celles des arbres de décision classique. On peut distinguer trois évolutions majeures :
  - Une *fuzzification* de l'espace de représentation c'est à dire la détermination de sous-ensembles flous pour décrire les attributs qu'ils soient à l'origine symboliques ou numériques et continus.
  - L'adaptation de la méthode de partitionnement
  - L'adaptation de la mesure de discrimination.

Exemple de construction d'arbre de décision

	Ciel	Température	Humidité	Vent	Jouer au Tennis
J1	soleil	chaud	élevée	faible	non
J2	soleil	Chaud	élevée	fort	Non
J3	couvert	Chaud	élevée	Faible	oui
J4	pluie	doux	élevée	Faible	Oui
J5	Pluie	froid	normale	faible	Oui
J6	Pluie	Froid	normale	Fort	Non
J7	Couvert	Froid	normale	Fort	Oui
J8	Soleil	Doux	élevée	Faible	Non
J9	Soleil	froid	normale	Faible	Oui
J10	Pluie	doux	normale	Faible	Oui
J11	soleil	doux	normale	Fort	Oui
J12	Couvert	doux	élevée	Fort	Oui
J13	couvert	chaud	normale	Faible	Oui
J14	pluie	doux	élevée	fort	non

Formules simplifiées de calcul d'entropie et gain d'informations

Pour un problème à 2 classes (règle de décision binaire)

Entropie

Entropie(S)≡ -(p+) log_2 (p+) –(p-)log_2(p-).

S est un ensemble d'exemples

p+ est la proportion d'exemples positifs, p- est la proportion d'exemples négatifs

Mesure l'homogénéité des exemples

Entropie ([9+,5-])=-(9/14) log_2 (9/14)-(5/14) log_2 (5/14)=0.940

Gain d'information

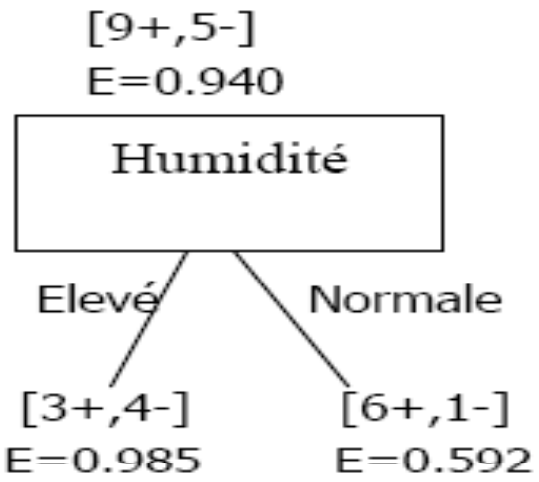
Gain(S, A)=Réduction d'entropie due à un tri suivant les valeurs de A

$$\textbf{Gain(S, A)} \equiv \textbf{Entropie(S)} - \sum_{v \in valeur(A)} (|\mathbf{S_v}| / |\mathbf{S}|) * \textbf{Entropie(S v)}$$

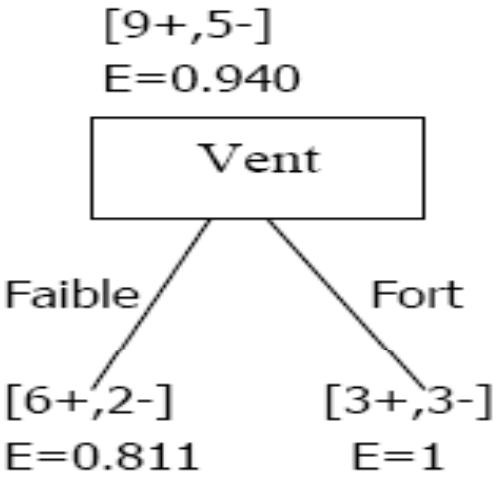
|S| désigne la cardinalité (nombre d'éléments) de l'ensemble S

Remarques:

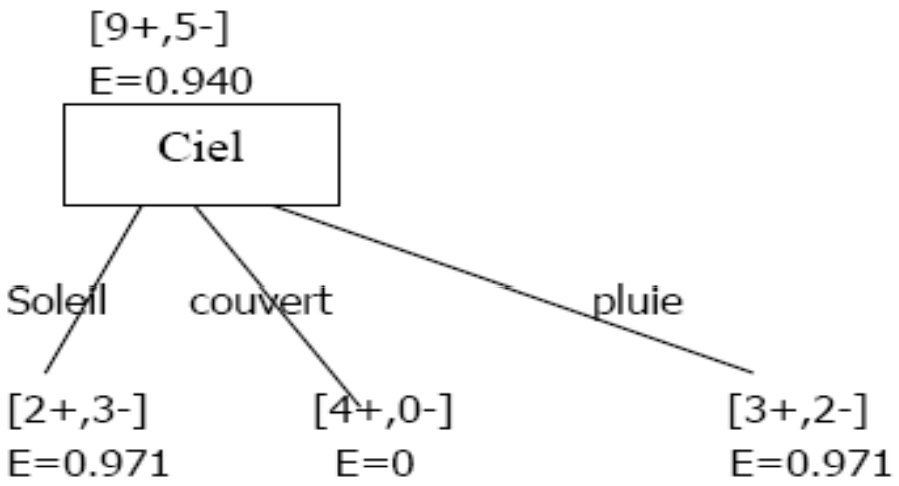
- Revoir aussi les diapositives 26 et 27 pour bien suivre les calculs.
- Pour le calcul du logarithme base a, $log_a(x)=ln(x)/ln(a)$ sachant que ln désigne fonction logarithme népérien, donc $log_2(x)=ln(x)/ln(2)$



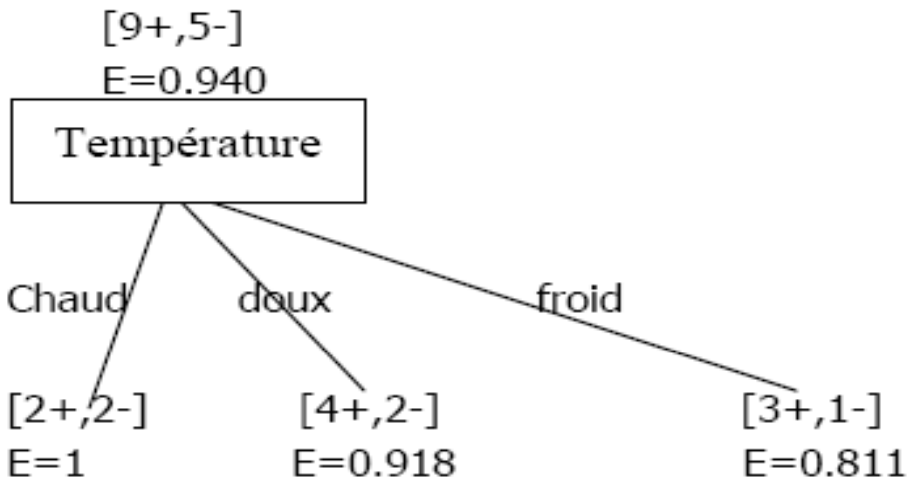
Gain(S,Humidité)=0.940-(7/14) 0.985-(7/14)0.592=0.151



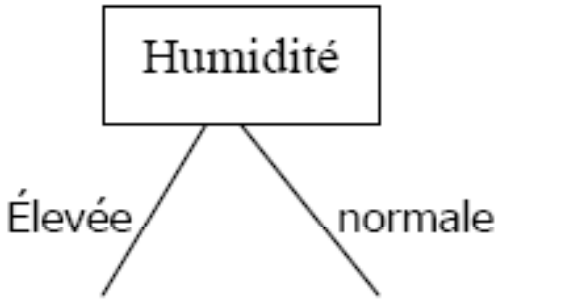
Gain (S, vent)=0.940-(8/14)0.811- (6/14)1=0.048



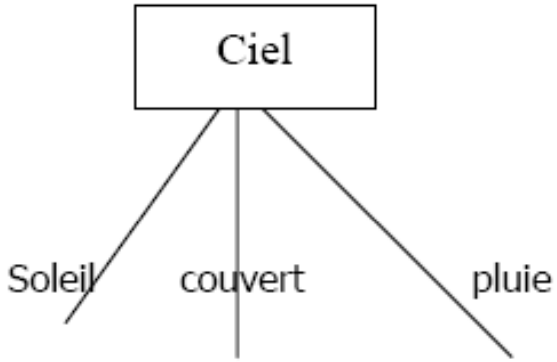
Gain(S,ciel)=0.940-(5/14)0.971-(5/14)0.971-0=0.246



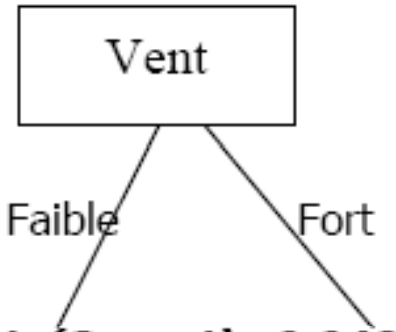
Gain(S, température)=0.940-(4/14)1-(6/14)0.918-(4/14)0.811=0.029



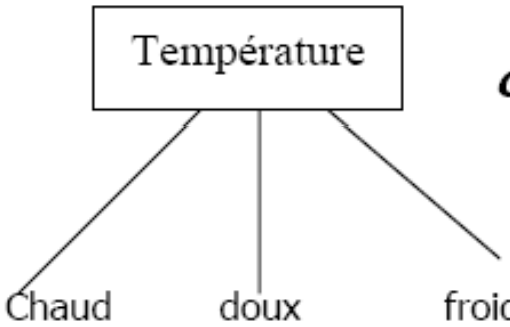
Gain(S, Humidité)=0.151



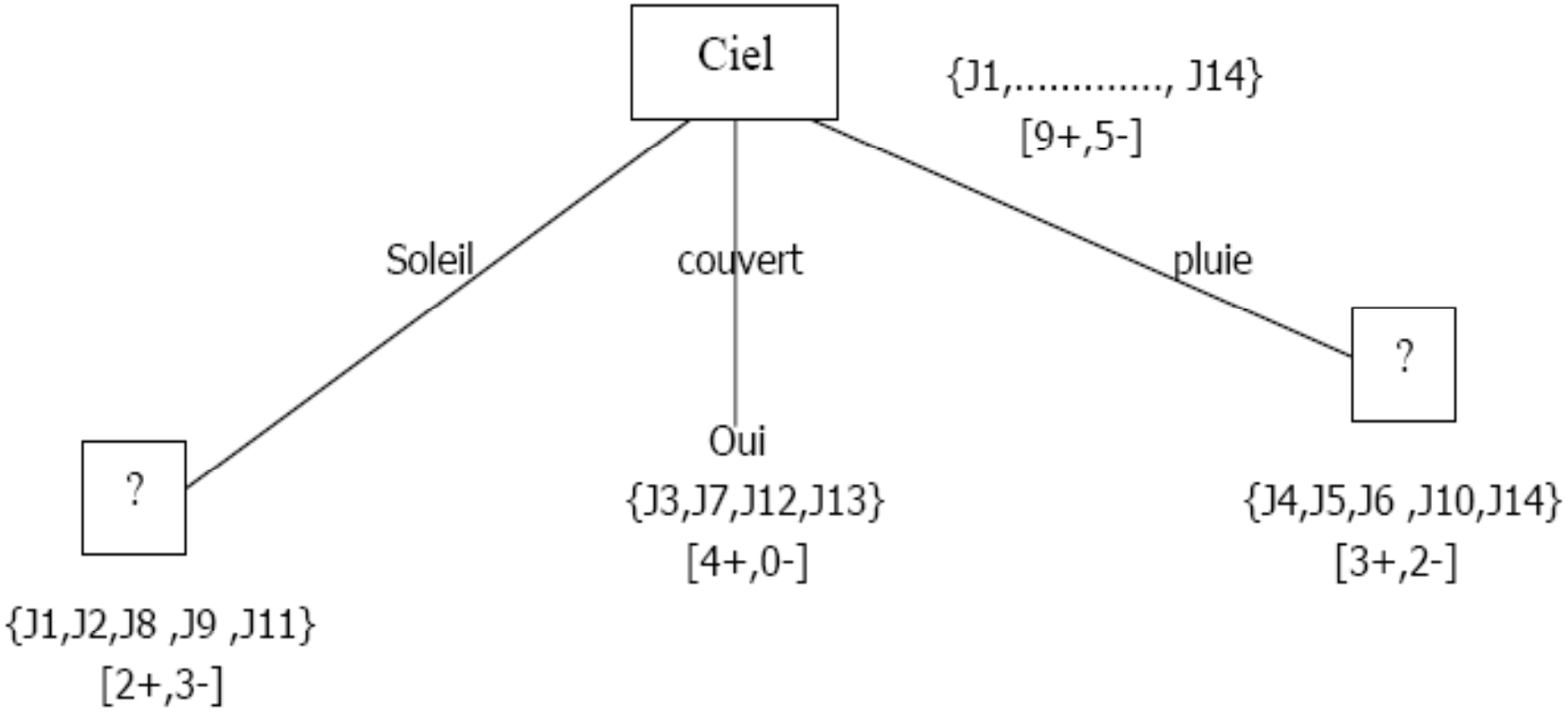
Gain(S, ciel)=0.246



Gain(S, vent)=0.048



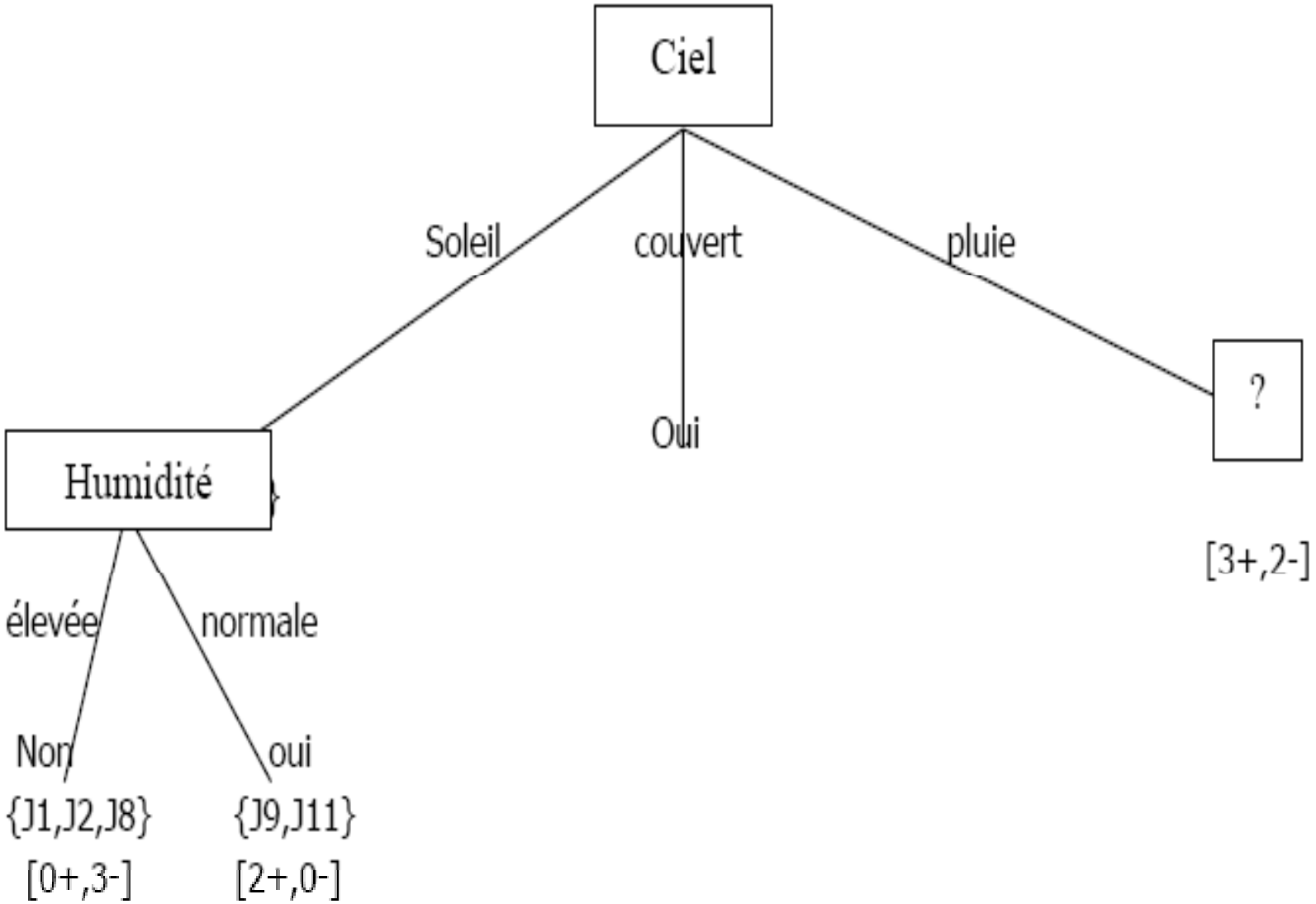
Gain(S, température)=0.029

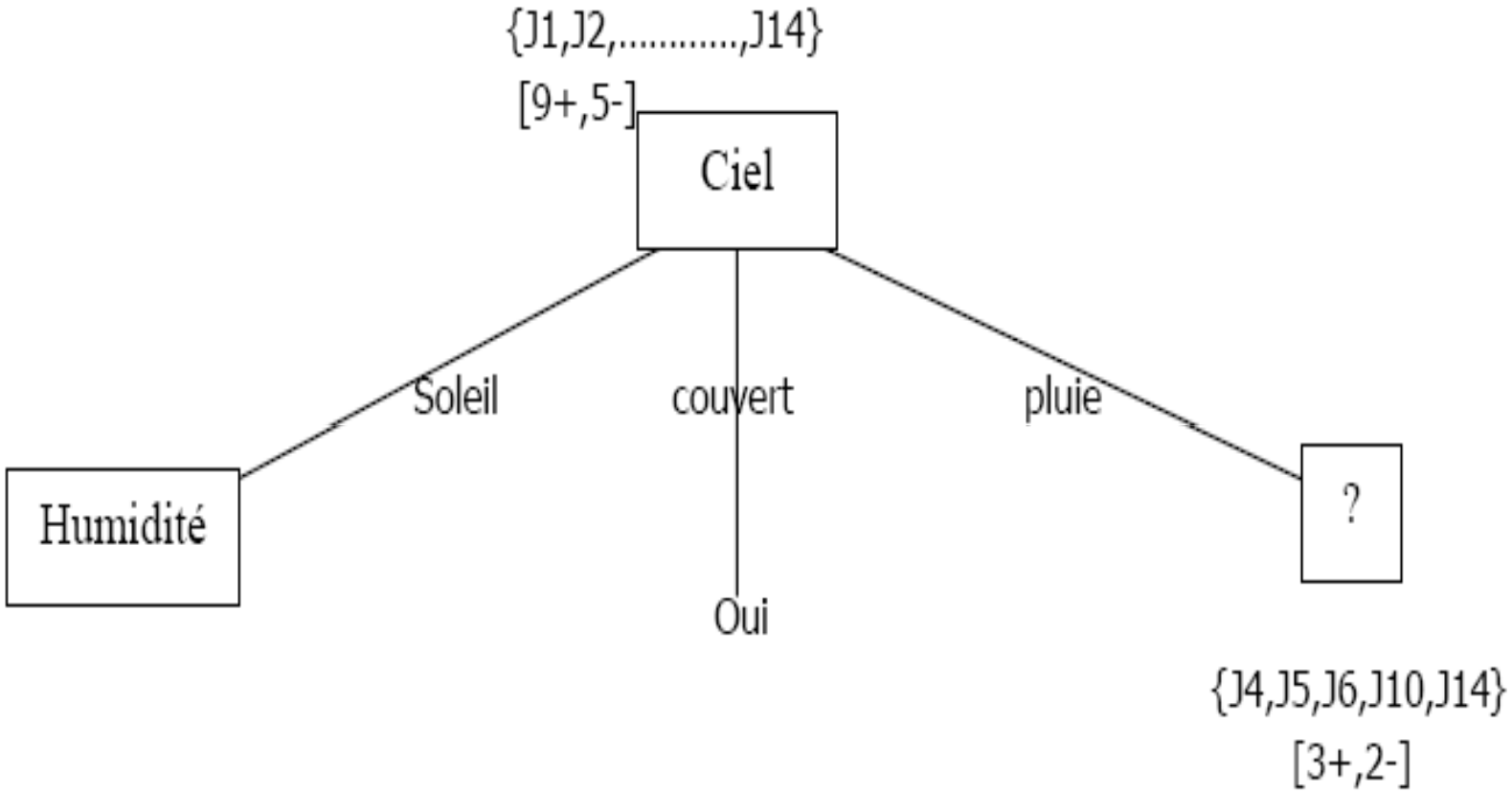


Gain (S soleil,Humidité)=0.970-(3/5)0-(2/5)0=0.970

Gain (S soleil,température)=0.970-(2/5)0-(2/5)1-(1/5)0=0.570

Gain (S soleil, vent)= 0.970-(2/5)1-(3/5)0.918=0.019


Ciel
Soleil
couvert
pluie
Oui
?
[3+,2-]
Humidité
élevée
normale
Non
{J1,J2,J8}
[0+,3-]
oui
{J9,J11}
[2+,0-]

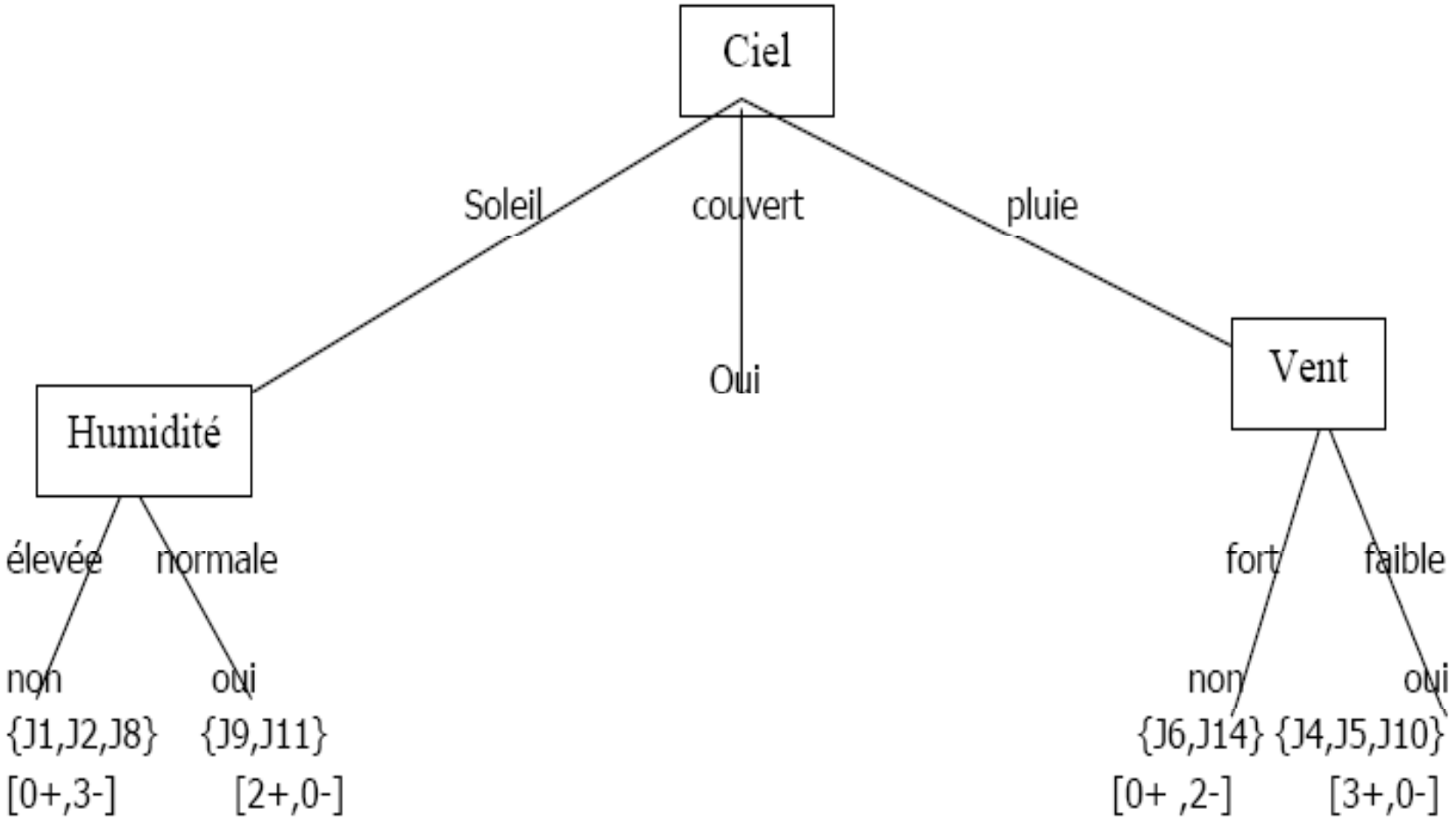


Gain (S pluie,Humidité)=0.970-(2/5)1-(3/5)0.918=0.019

Gain (S pluie,température)=0.970-(0/5)-(3/5)0.918-(2/5)1=0.019

Gain (S pluie, vent)= 0.970-(2/5)0-(3/5)0=0.970

L'arbre de décision obtenu est alors :



Exercice

Les diapositives suivantes représentent un extrait du livre: Cornuéjols A., Miclet L. "Apprentissage artificiel : Concepts et algorithmes", Eyrolles, 2002

- Dans l'exemple qui suit, le problème d'apprentissage consiste à trouver une règle de décision binaire à partir de huit exemples sur quatre paramètres binaires.
- Le problème qui se pose à un enfant qui revient de l'école est le suivant : *peut-il aller jouer chez son voisin ou pas?*
- L'expérience, qu'il a acquise par punition-récompense sur les huit jours d'école précédents, est résumée dans le tableau des huit exemples d'apprentissage.

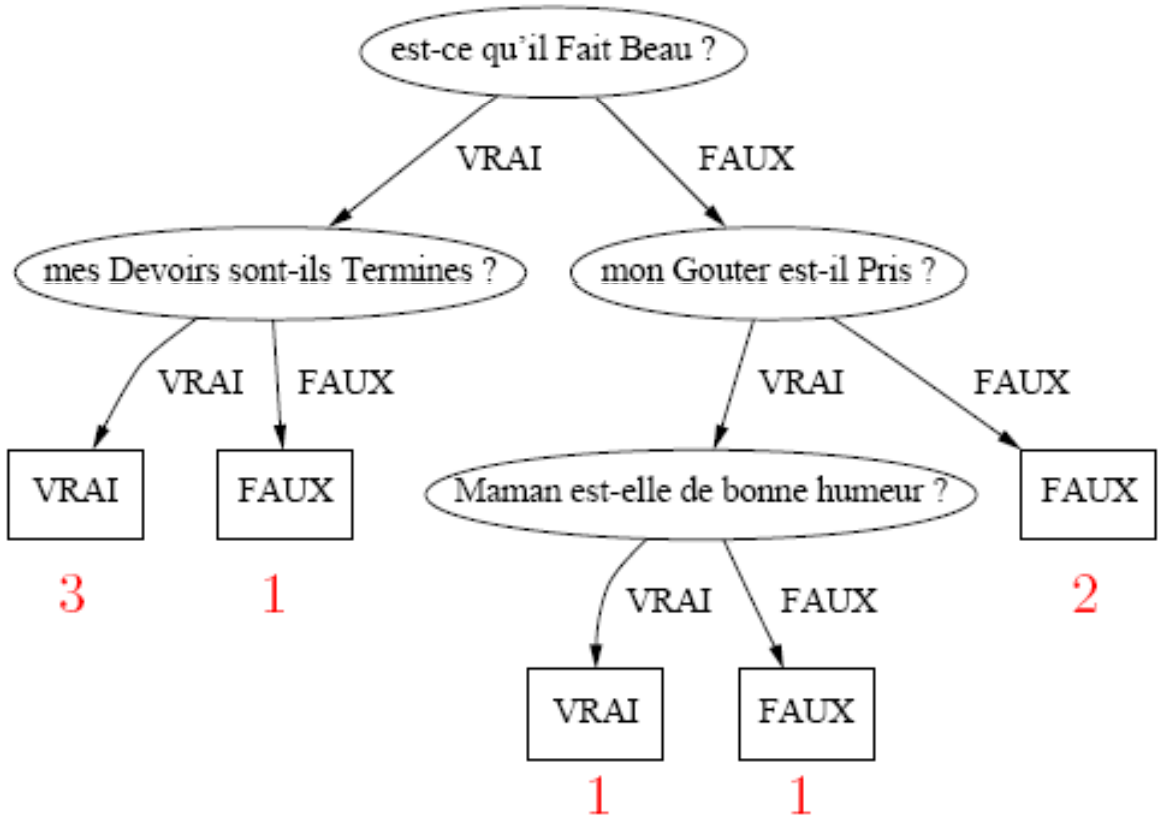
	mes Devoirs sont-ils Finis?	Maman est-elle de Bonne Humeur?	Est-ce qu'il Fait Beau?	Mon Goûter est-il Pris?	**DECISION**
1	VRAI	FAUX	VRAI	FAUX	**OUI**
2	FAUX	VRAI	FAUX	VRAI	**OUI**
3	VRAI	VRAI	VRAI	FAUX	**OUI**
4	VRAI	FAUX	VRAI	VRAI	**OUI**
5	FAUX	VRAI	VRAI	VRAI	**NON**
6	FAUX	VRAI	FAUX	FAUX	**NON**
7	VRAI	FAUX	FAUX	VRAI	**NON**
8	VRAI	VRAI	FAUX	FAUX	**NON**

Travail demandé

De la même manière que dans l'exemple de construction d'arbre présenté dans les diapositives 48-56, construire l'arbre correspondant à l'ensemble des huit exemples présentés dans le tableau précédent.

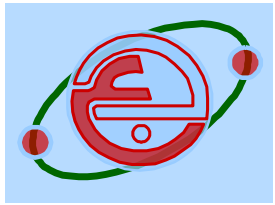
Le résultat final est illustré par l'arbre obtenu (voir diapositive 60).

Solution



L'arbre de décision construit sur les exemples précédents.

Université Badji Mokhtar - Annaba
Département d'Informatique



جامعة باجي مختار – عنابـــــة

Concepts de Base des Algorithmes Génétiques

Cours d'Apprentissage Automatique
Master 1 STIC, 2014-2015

Présenté par
Pr Souici – Meslati L.

Plan

1. Génétique et évolution naturelle
2. Définitions et historique
3. Applications
4. Principe de base d'un AG
5. Organigramme général d'un AG
6. Mise en œuvre d'un AG
7. Exemple illustratif

Génétique et évolution naturelle

L'évolution dans la nature survient quand des entités ont la capacité de se reproduire, qu'il existe une population de ces entités, qu'il existe une variété (diversité) à travers ces entités, et que la survie des entités dépend des différences entre elles.

Chaque organisme vivant possède un ensemble unique de chromosomes qui décrit comment le construire. Un chromosome n'est autre qu'une longue molécule d'ADN dont les différentes parties sont appelées **gènes**.

Quand un organisme se reproduit, il fait passer une copie de ses chromosomes à sa progéniture. Des erreurs, pouvant surgir au moment de la copie, peuvent éventuellement donner une progéniture meilleure (*mutations*). Dans la nature et lors d'une reproduction, les chromosomes de deux parents sont combinés produisant ainsi une progéniture avec un mélange des chromosomes des deux parents. Ceci peut mener à des combinaisons des bons gènes ; ce qui permet lors d'une autre reproduction à améliorer encore la progéniture. Ce processus clé est connu sous le nom de *«croisement»(crossover)*,

Définitions et Historique

Les Algorithmes Evolutionnaires (AE) sont inspirés du concept de sélection naturelle élaboré par Charles Darwin.

Dans The Origin of Species (1859), Darwin montre que l'apparition d'espèces distinctes se fait par le biais de la sélection naturelle de *variations individuelles*. Cette sélection naturelle est fondée sur la lutte pour la vie, due à une population tendant naturellement à s'étendre mais disposant d'un espace et de ressources finis. Il en résulte que les individus les *plus adaptés* (*the fittest* en anglais) tendent à survivre plus longtemps et à se reproduire plus aisément. Le terme " adapté " se réfère à l'environnement, que l'on peut définir comme étant l'ensemble des conditions externes à un individu, ce qui inclut les autres individus. Les lois de variation (croisements et mutations) furent expliquées plus tard par Mendel, puis par la génétique moderne.

Définitions et Historique

Les AE font partie du champ de l'Intelligence Artificielle (IA). Il s'agit d'IA de bas niveau, inspirée par " l'intelligence " de la Nature. Intelligence que l'on peut définir de la façon suivante : *" the capability of a system to adapt its behaviour to meet its goals in a range of environments "*.

Trois types d'AE ont été développés isolément et à peu près simultanément, dans les années 60, par différents scientifiques : les *Algorithmes Génétiques* , les *Stratégies d'Evolution* , et la *Programmation Evolutionnaire* .

Présentant des différences marquées à l'origine, ils tendent de plus en plus à se confondre. Dans les années 90, ces trois champs ont commencé à sortir de leur isolement et ont été regroupés sous le terme anglo-saxon d' *Evolutionary Computation* .

Définitions et Historique

Les algorithmes génétiques sont des méthodes robustes de recherche, fondées par John H.Holland dans les années 60 à l'université de Michigan. Les algorithmes génétiques ont été popularisés par J. Holland en 1975 grâce à son livre «Adaptation in natural and artificial systems». Dans son travail, il cherche à utiliser la métaphore de l'adaptation naturelle des espèces par variation et sélection dans le but de réaliser une adaptation optimale à l'environnement.

J. Holland est l'auteur de plusieurs travaux concernant les fondements théoriques des algorithmes génétiques. Plusieurs autres auteurs ont eu des contributions remarquables sur ce sujet (Goldberg, De Jong, Baker, Grefenstette et Davis).

Avant la fin des années 80, les travaux de ces derniers se sont concentrés beaucoup plus sur les concepts théoriques des algorithmes génétiques.

Définitions et Historique

Dès la fin des années 80, les recherches sur les algorithmes génétiques se sont accentuées et se sont orientées vers l'amélioration de la qualité des résultats et des performances. En particulier, l'implémentation des algorithmes génétiques sur des architectures parallèles a pris une part considérable des efforts des chercheurs. Par conséquent, plusieurs techniques et variantes des algorithmes génétiques sont apparues. Ces techniques ont été mises au point du fait que certains problèmes s'adaptent mal aux algorithmes génétiques de base et nécessitent la modification du sens de certains paramètres et l'introduction de nouveaux concepts.

Le vocabulaire employé est directement calqué sur celui de la théorie de l'évolution et de la génétique.

Nous parlerons donc d'individus (solutions potentielles), de population , de gènes (variables), de chromosomes , de parents , de descendants, de reproduction, de croisement , de mutations , etc. Et nous nous appuierons constamment sur des analogies avec les phénomènes biologiques.

Applications

Les applications des AG sont multiples : optimisation de fonctions numériques difficiles (discontinues, multimodales, bruitées…), traitement d'image (alignement de photos satellites, reconnaissance de suspects…), optimisation d'emplois du temps, contrôle de systèmes industriels, apprentissage des réseaux de neurones...

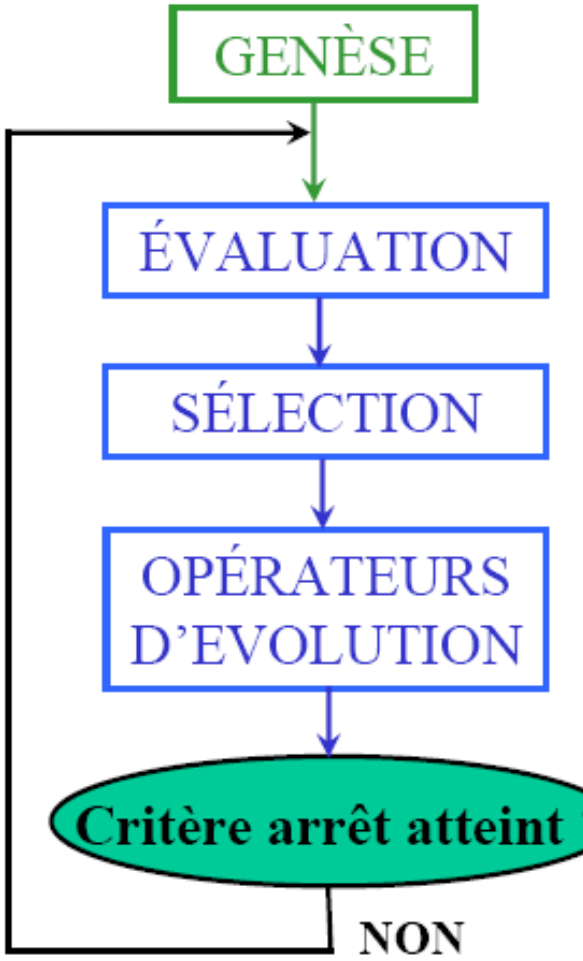
Les AG peuvent être utilisés pour contrôler un système évoluant dans le temps (chaîne de production, centrale nucléaire…) car la population peut s'adapter à des conditions changeantes. En particulier, ils supportent bien l'existence de bruit dans la fonction à optimiser. Ils peuvent aussi servir à déterminer la configuration d'énergie minimale d'une molécule ou à modéliser le comportement animal.

Les AG sont également utilisés pour optimiser des réseaux (câbles, fibres optiques, mais aussi eau, gaz…), des circuits VLSI, des antennes… On envisage l'intégration d'AG dans certaines puces électroniques afin qu'elles soient capables de se reconfigurer automatiquement en fonction de leur environnement *(Evolving Hardware* en anglais *)* .

Principe de base d'un AG

Le principe de base qui gouverne le fonctionnement d'un algorithme génétique (AG) est simple. En effet, celui-ci maintient une population d'individus codés, de laquelle il sélectionne les meilleurs en fonction d'un certain critère (par exemple, les individus dont le codage maximise une fonction définie). L'algorithme effectue ensuite le croisement entre les individus sélectionnés à la manière des systèmes génétiques naturels. Toutefois, si le codage, la fonction ou le croisement est effectué au hasard, il est très probable que les résultats obtenus ne soient pas intéressants.

Organigramme général d'un algorithme génétique



Création (généralement aléatoire) d'une population initiale

Notion de « degré d'adatation » (« fitness ») des individus

On favorise les « plus aptes »

Nouvelle génération (ou modif. de la population), avec reproduction/*modifications/combinaisons* des individus de la génération n-1

Récupérer le (voire les) meilleur(s) individu(s) de la population finale

Mise en œuvre d'un algorithme génétique

Population initiale

Elle constitue l'ensemble des individus-solutions avec lesquels l'algorithme commence, elle représente la première génération et est générée aléatoirement. Cependant, rien ne nous empêche d'utiliser des résultats ou des solutions existantes pour former celle-ci.

Une population « pure » produit, au bout de quelques générations, des individus très fortement apparentés qui risquent de devenir tous identiques. Lorsque cela se produit, l'évolution s'arrête. Pour pallier ce problème, l'initialisation aléatoire de cette population est très recommandée.

La taille de la population initiale détermine la taille des populations des générations à venir, traitées par l'AG qui peut aussi traiter des populations de taille variable. Le choix de cette taille est primordial car la convergence de l'AG en dépend fortement.

Mise en œuvre d'un algorithme génétique

Etape 1: Choix du codage des chromosomes

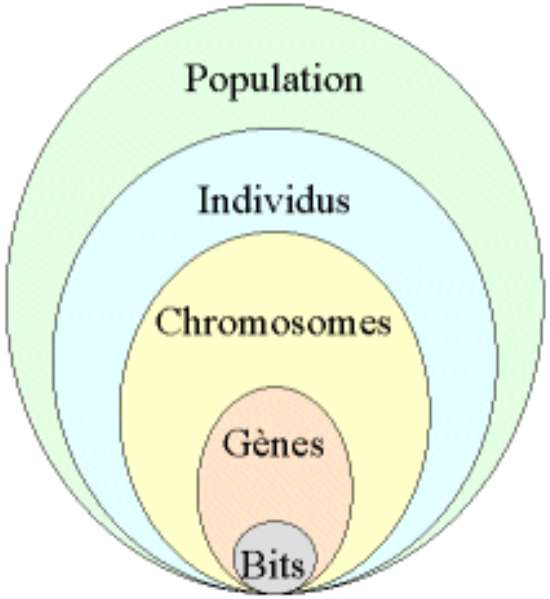
La première étape est de définir et de coder convenablement le problème. A chaque variable d'optimisation *xi* (à chaque paramètre du dispositif), nous faisons correspondre un *gène* . Nous appelons *chromosome* un ensemble de gènes. Chaque dispositif est représenté par un *individu* doté d'un génotype constitué d'un ou plusieurs chromosomes. Nous appelons *population* un ensemble de *N* individus que nous allons faire évoluer.

Buts

Pouvoir coder toutes les solutions

Faciliter la mise en oeuvre
des opérations de reproduction.

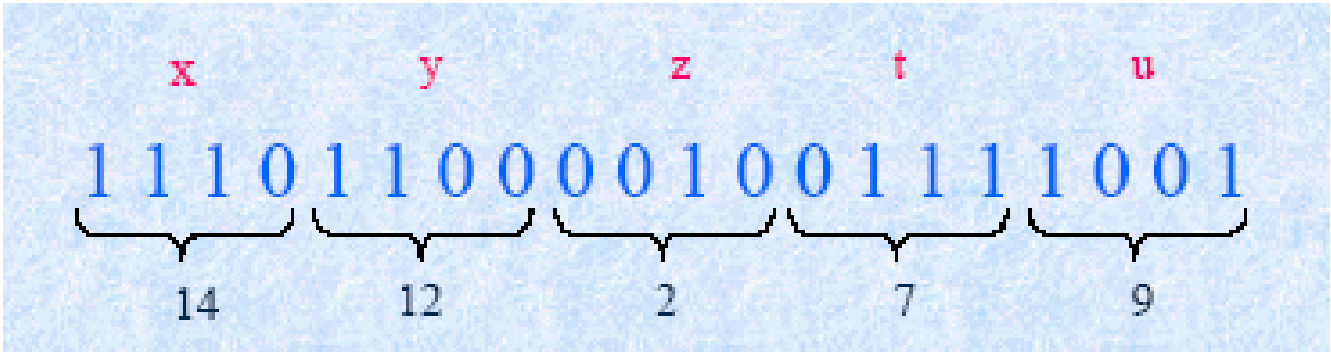
Choix en fonction du problème.



Mise en œuvre d'un algorithme génétique

Exemple1: Optimisation d'une fonction à 5 variables (x,y,z,t,u)

- Codage binaire d'un chromosome: chaque paramètre est codé sur 4 bits (valeurs entre 0 et 15)
- Taille du chromosome: 20 bits



Mise en œuvre d'un algorithme génétique

Exemple 2:

Le voyageur de commerce (10 villes)

- codage par valeurs entières
  - 1 2 9 0 3 5 8 6 7 4
- taille du chromosome: 10

- codage binaire (4 bits par valeur :0..9)
  - 0001001010010000001101011000011001110100
    - taille du chromosome: 40
    - Problème des valeurs *possibles* à gérer

Mise en œuvre d'un algorithme génétique

Etape 2: Définition d'une fonction d'adaptation (fitness)

Plaçons-nous, par exemple, dans le monde d'une espèce carnivore, un animal qui arrive à capturer ses proies facilement et qui se défend bien s'adapte parfaitement à son environnement, c'est à dire qu'il arrive à en profiter pleinement. En revanche, un animal assez faible pour se défendre ou ne pouvant pas attraper facilement ses proies finira par mourir. Il est évident que si nous avions à *noter* ces deux animaux, la note du premier sera nettement plus élevée que celle du deuxième.

Dans les AGs, l'environnement est simulé par une fonction. Les solutions dites plus adaptées maximisent cette fonction tandis que les moins adaptées la minimisent. La différence majeure avec ce qui se passe dans la nature est que celle-ci n'attribue pas de notes aux individus, et même si cette note existe, elle dépendra d'un nombre incalculable de paramètres.

Mise en œuvre d'un algorithme génétique

Etape 2: Définition d'une fonction d'adaptation (fitness)

Il faut définir une *fonction d'adaptation* (évaluation), c'est la fonction à optimiser qui reflète la qualité des solutions pour résoudre le problème posé. Si le problème à résoudre consiste à optimiser une certaine fonction mathématique alors celle-ci constitue la fonction d'évaluation appelée aussi *fonction d'adaptation*. Cependant, si le problème consiste à trouver une bonne combinaison de paramètres pour la conception d'une pièce mécanique ou à trier un tableau, il n'est pas évident de définir cette fonction. Il faut alors trouver un artifice qui permet d'attribuer une note élevée aux solutions qui s'approchent le plus de la solution optimale et une note assez faible aux solutions qui s'en éloignent. Cette fonction est à définir en fonction du problème, elle doit atteindre son maximum pour les meilleures solutions et son minimum pour les plus mauvaises.

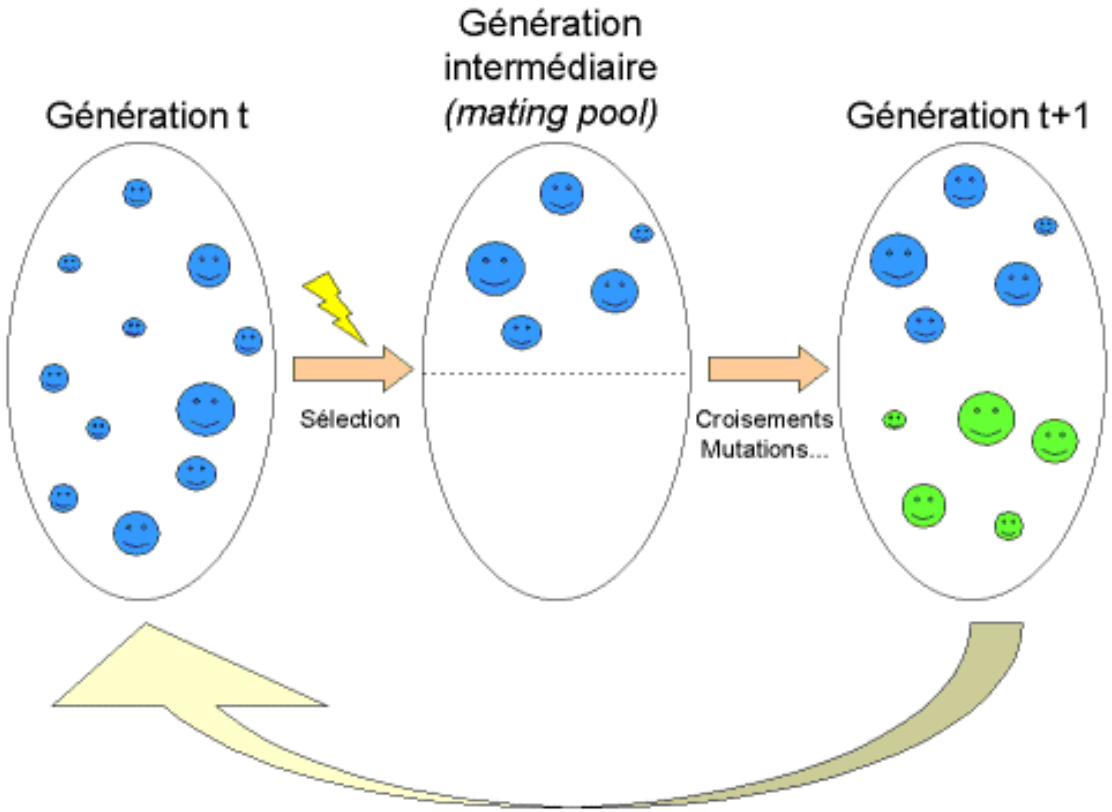
Mise en œuvre d'un algorithme génétique

Etape 2: Définition d'une fonction d'adaptation (fitness)

Le seul critère que doit remplir la fonction d'adaptation, est qu'elle doit être définie sur tout l'ensemble des solutions possibles pour éviter qu'un chromosome, résultant de la recombinaison, ne puisse avoir d'adaptation, ce qui empêche l'AG de continuer son exécution. La qualité des résultats des AGs dépend étroitement du choix de la fonction d'adaptation. Elle doit permettre, au maximum, à l'AG d'explorer l'espace de solutions sans se bloquer dans une de ses régions.

Mise en œuvre d'un algorithme génétique

On appelle *génération* la population à un instant *t* donné.
Une fois réalisée l'évaluation de la génération, on opère une sélection à partir de la fonction d'adaptation. Seuls les individus passant l'épreuve de sélection peuvent accéder à la *génération intermédiaire* (*mating pool*) et s'y reproduire.



Mise en œuvre d'un algorithme génétique

Etape 3: Choix de la méthode de sélection

A chaque itération de l'algorithme, un ensemble d'individus est sélectionné pour être candidat à la reproduction. La méthode de sélection a trois objectifs principaux:

- Elle permet de choisir les individus sur lesquels s'appliqueront les opérations de reproduction pour la création de la future génération (création du mating-pool).
- Elle doit "favoriser" les meilleurs individus. De ce fait, les individus qui maximisent la fonction d'adaptation auront plus de chances d'être sélectionnés que les autres tandis que les individus les moins adaptés risquent de ne jamais être sélectionnés.

- Elle doit permettre d'explorer les différentes parties de l'ensemble de recherche.

Mise en œuvre d'un algorithme génétique

Etape 3: Choix de la méthode de sélection

Il existe plusieurs méthodes de sélection:

1. Sélection proportionnelle

C'est la méthode de sélection la plus utilisée. Elle est aussi connue sous le nom de Méthode de la Roue (Roulet Wheel Selection-RWS). La roue est divisée en secteurs et chaque secteur est associé à un individu, la superficie du secteur étant proportionnelle à la valeur de l'adaptation de l'individu, les meilleurs individus auront une probabilité plus grande d'être choisis.

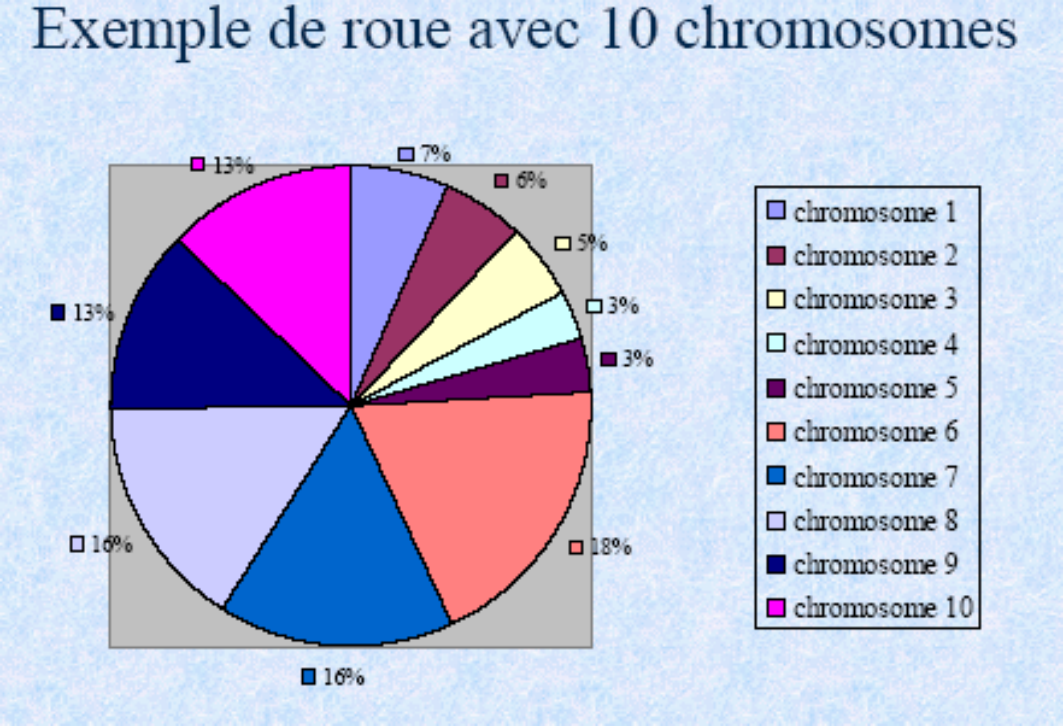
Soient:
n: nombre d'individus à remplacer
Np: nombre d'individus de la population
fj: valeur d'adaptation de l'individu j
ni: le nombre de sélections espérées de l'individu i

$$n_i = \frac{n}{\sum_{j=1}^{N_p} f_j} f_i$$

Mise en œuvre d'un algorithme génétique

Etape 3: Choix de la méthode de sélection

Ensuite, la bille est lancée et s'arrête sur un chromosome. Les meilleurs chromosomes peuvent ainsi être tirés plusieurs fois et les plus mauvais ne jamais être sélectionnés.



Mise en œuvre d'un algorithme génétique

Etape 3: Choix de la méthode de sélection

2. Sélection par rang

- La sélection précédente rencontre des problèmes lorsque les valeurs d'adaptation des chromosomes varient énormément. Si la meilleure fonction d'évaluation d'un chromosome représente 90% de la roulette alors les autres chromosomes auront très peu de chance d'être sélectionnés et on arriverait à une stagnation de l'évolution.
- La sélection par rang trie d'abord la population par fitness. Ensuite, chaque chromosome se voit associé un rang en fonction de sa position. Ainsi le plus mauvais chromosome aura le rang 1 , le suivant 2 , et ainsi de suite jusqu'au meilleur chromosome qui aura le rang N (pour une population de N chromosomes). La sélection par rang d'un chromosome est la même que par roulette, mais les proportions sont en relation avec le rang plutôt qu'avec la valeur de l'évaluation. Avec cette méthode de sélection, tous les chromosomes ont une chance d'être sélectionnés. Cependant, elle conduit à une convergence plus lente vers la bonne solution. Ceci est dû au fait que les meilleurs chromosomes ne diffèrent pas énormément des plus mauvais.

Mise en œuvre d'un algorithme génétique

Etape 4: Définition des opérateurs génétiques

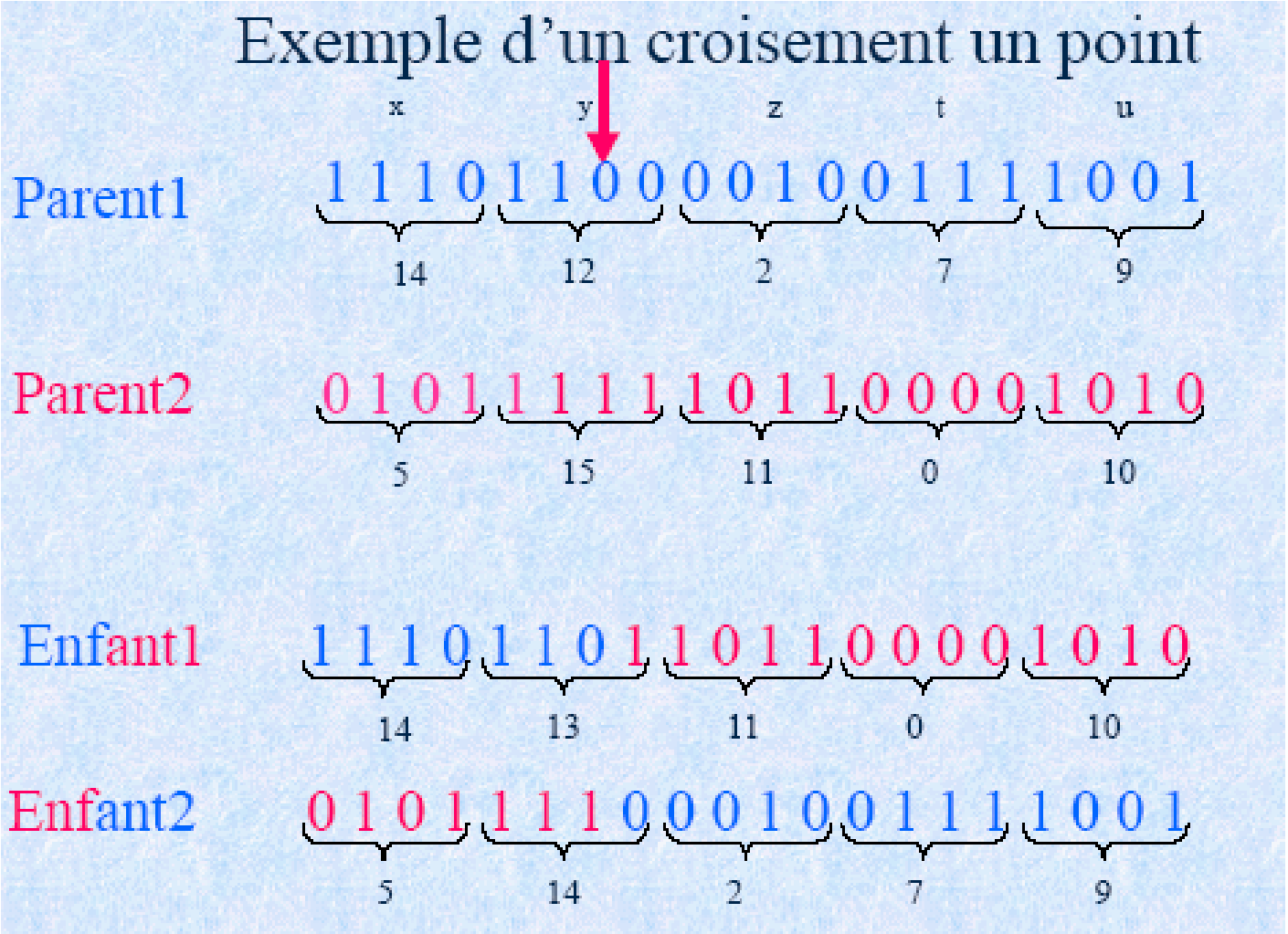
Croisement:

- S'applique sur deux individus différents.
- Résultat: chromosome formé à partir des gènes de ses deux parents.
- Deux enfants sont "produits" pour la génération suivante.
- Un pourcentage de croisement est fixé.

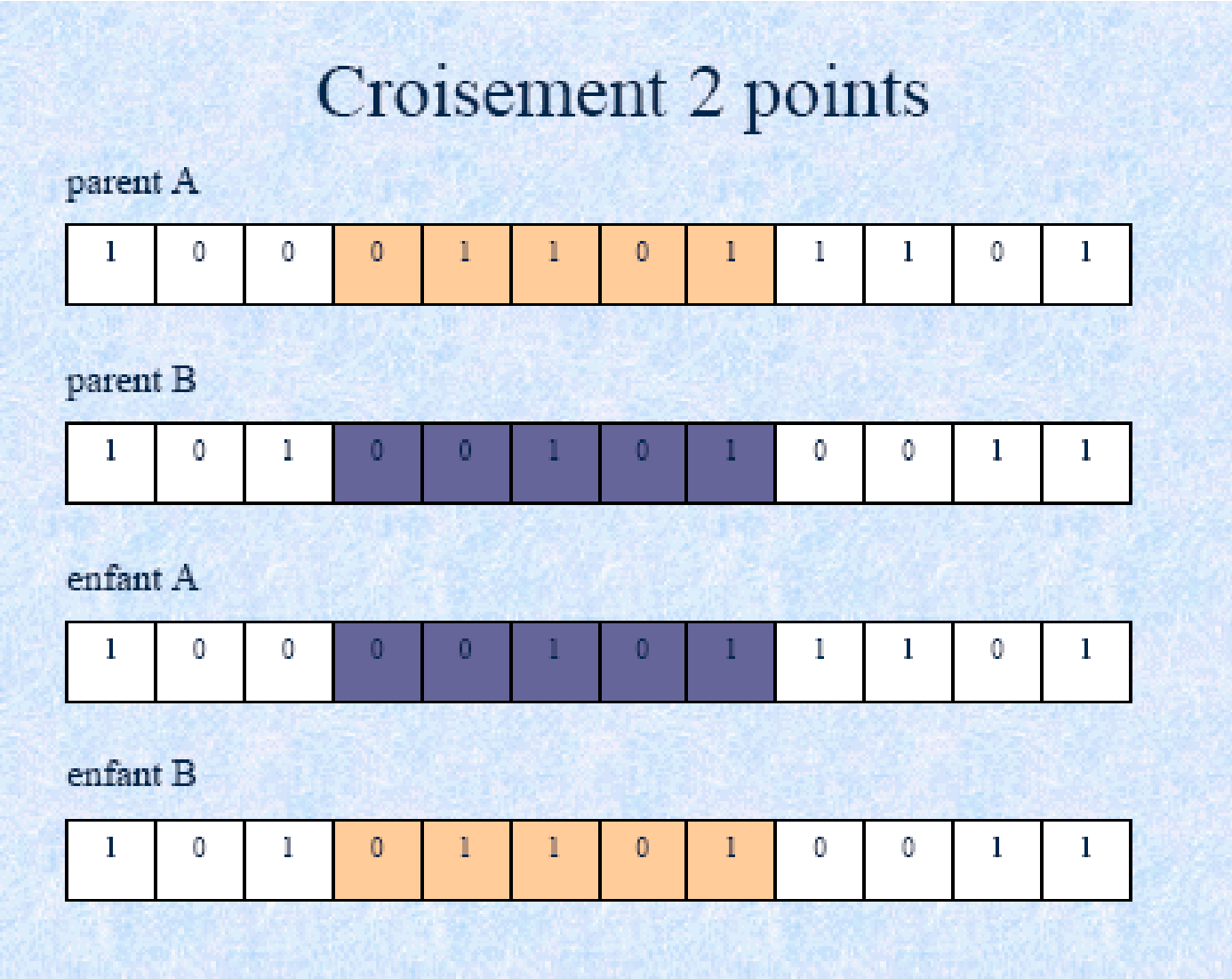
Mutation:

- S'applique sur un seul individu par la modification d'un ou plusieurs gènes du parent choisi(s) aléatoirement.
- Un seul nouvel enfant est fourni.
- Un pourcentage de mutation est fixé.

Mise en œuvre d'un algorithme génétique



Mise en œuvre d'un algorithme génétique



Mise en œuvre d'un algorithme génétique

Croisement logique

parent A

1	0	0	0	1	1	0	1	1	1	0	1

parent B

1	0	1	0	0	1	0	1	0	0	1	1

enfant A (AND)

1	0	0	0	0	1	0	1	0	0	0	1

enfant B (OR)

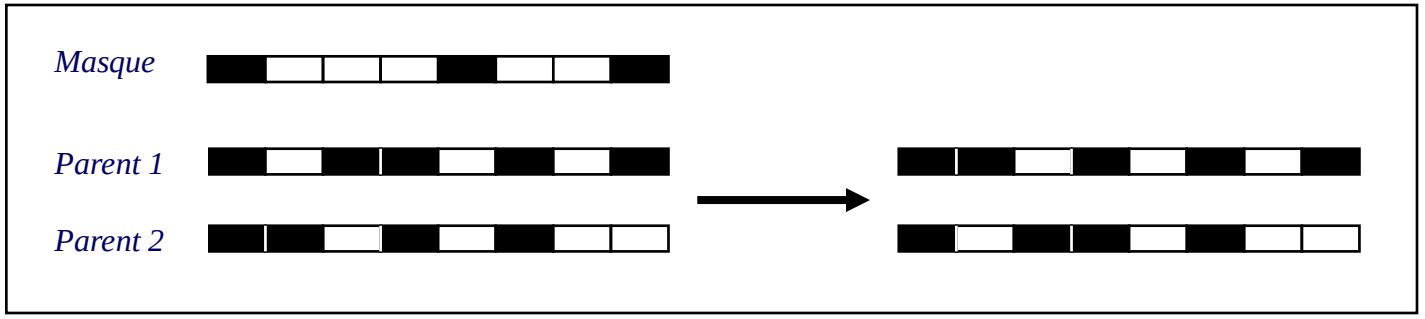
1	0	1	0	1	1	0	1	1	1	1	1

Mise en œuvre d'un algorithme génétique

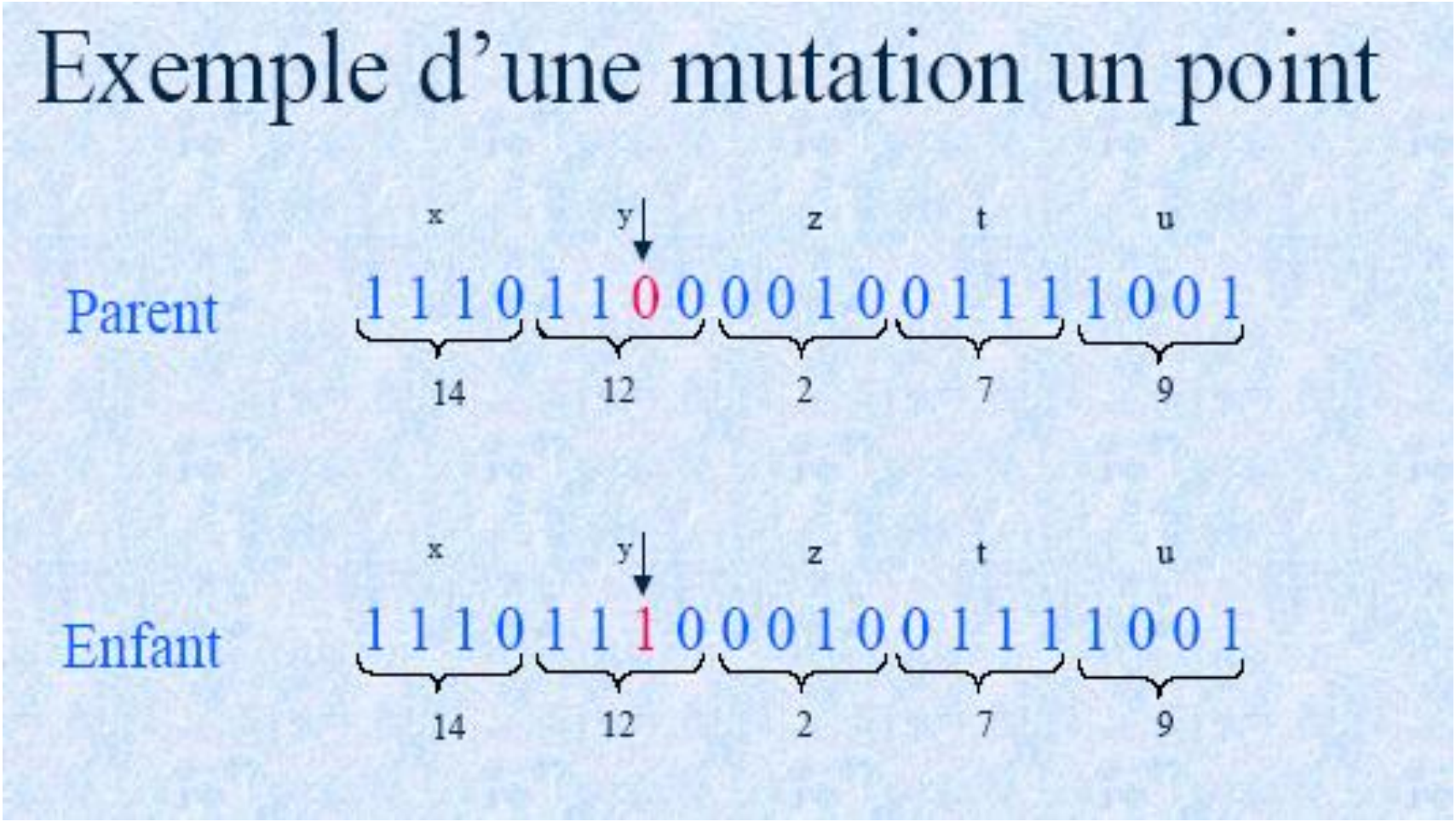
Croisement avec masque

Si la taille du codage utilisé est égale à *l* alors une chaîne binaire, appelée *masque*, de longueur *l* est générée aléatoirement. Après avoir choisi deux parents pour le croisement, l'algorithme suivant est appliqué

Pour *i* allant de 1 à *l* faire
Si le bit *i* du masque est à 1 alors
-le bit *i* du premier enfant reçoit le bit *i* du premier parent.
-le bit *i* du deuxième enfant reçoit le bit *i* du deuxième parent.
Sinon
-le bit *i* du premier enfant reçoit le bit *i* du deuxième parent.
-le bit *i* du deuxième enfant reçoit le bit *i* du premier parent.



Mise en œuvre d'un algorithme génétique

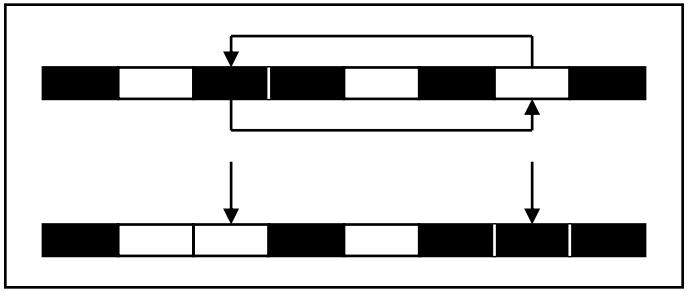


Mise en œuvre d'un algorithme génétique

Etape 4: Définition des opérateurs génétiques (fin)

- Mutation de plusieurs positions

- L'inversion



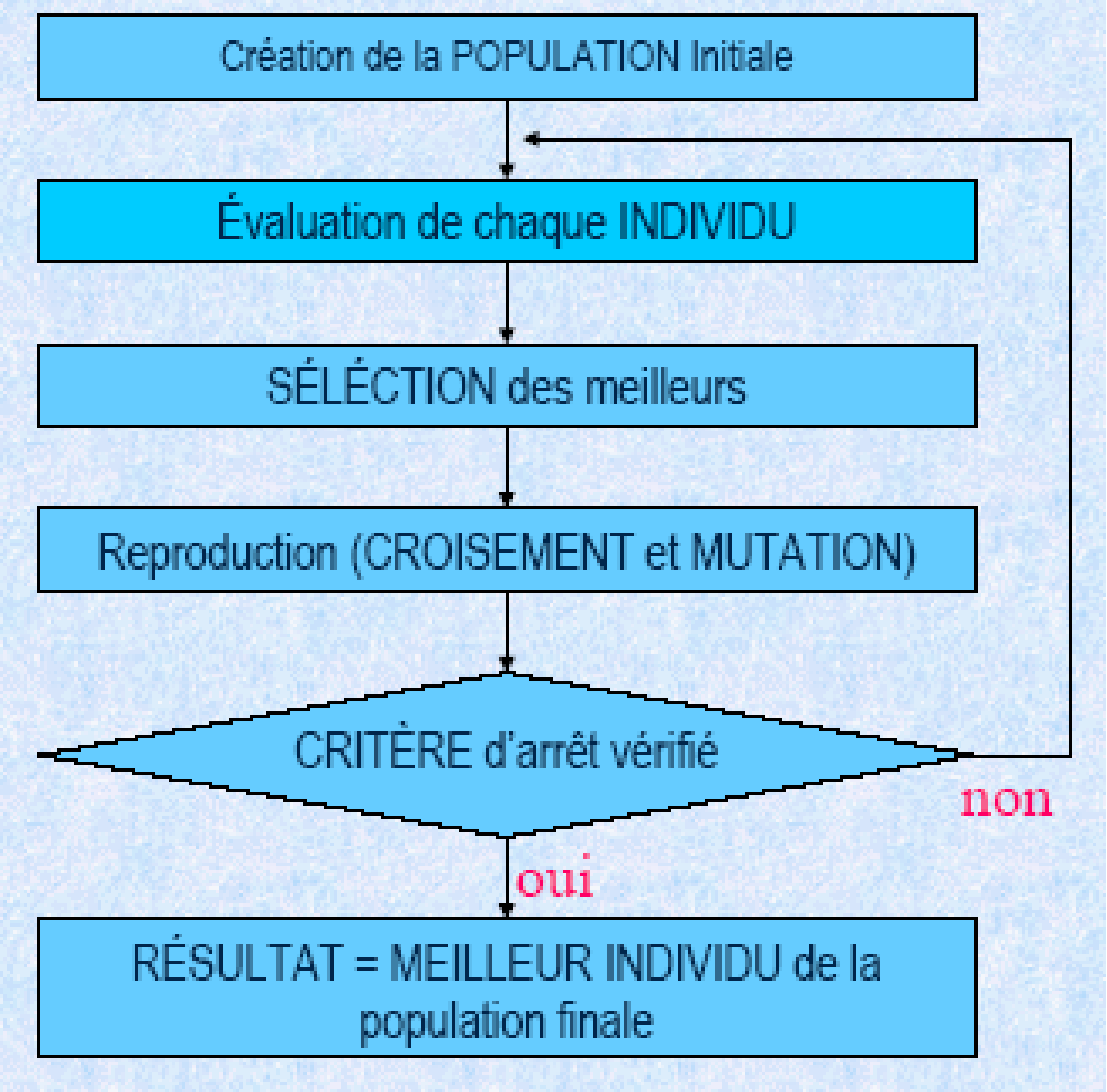
Mise en œuvre d'un algorithme génétique

Etape 5: Convergence(Arrêt du processus)

De générations en générations la fonction d'adaptation du meilleur chromosome et la moyenne de l'ensemble de la population vont évoluer et tendre vers l'optimum (global).

- La convergence est l'augmentation progressive vers de l'uniformité.
- Un gène a convergé quand 95% de la population possède la même valeur du gène.
- La population a convergé quand tous les gènes ont convergé.

Rappel: Organigramme d'un AG



Exemple illustratif

Une population initiale est composée de 4 individus. Chaque individu est caractérisé par 3 traits: ses capacités de jardinier, sa qualité de cuisinier et ses envies cleptomanes (de voler). Chaque trait peut prendre 2 valeurs (0 ou 1), suivant que l'individu ait ou n'ait pas la caractéristique. L'environnement est modélisé par une fonction d'adaptation définie pour chacune des combinaisons des 3 caractéristiques; par exemple :

l'individu (0;0;0), qui n'est ni cuisinier, ni jardinier, ni cleptomane, aura une qualité d'adaptation nulle, ce qui traduit le fait que, ne pouvant se fournir sa nourriture, il dépérira avant d'avoir pu se reproduire ;

l'individu (0;1;0) ne sera pas mieux favorisé car, malgré ses talents de cuisinier, il n'aura rien à se mettre sous la dent;

l'individu (1;0;0) aura néanmoins ses chances en se nourrissant de légumes crus. (0;0;1) survit en volant ;

l'individu (0;1;1) arrive à améliorer la qualité de ses vols en les cuisinant;

l'individu (1;0;1) combine ses vols avec les produits de son jardin ;

l'individu (1;1;0) est l'individu idéal ;

l'individu (1;1;1) perd du temps et de l'énergie à voler pour son propre plaisir, ce qui lui donne une capacité d'adaptation inférieure à celle de (l;1;0).

Exemple illustratif

Individus	qualité d'adéquation
(0;0;0)	1
(0;0;1)	3
(0;1;0)	1
(0;1;1)	4
(1;0;0)	2
(1;0;1)	4
(1;1;0)	5
(1;1;1)	4

Exemple illustratif

Supposons que la population initiale soit constituée des individus {(1;0;0), (0;1;0), (0;0;1), (1;1;1)}. Voyons comment agit un Algorithme Génétique pour trouver la solution optimale:

Première phase : *évaluation* des qualités d'adaptation

(1;0;0) se voit attribué un score de 2
(0;1;0) a un score de 1
(0;0;1) a 3, tandis que (1;1;1) a 4.

Deuxième phase : *sélection*

L'algorithme attribue à chaque individu une chance de participer à la phase de reproduction proportionnelle au rapport de sa qualité d'adaptation par rapport à la qualité moyenne. Ainsi, nous attribuerons à l'individu 1 la probabilité de 2/(2+1+3+4), soit 2 chances sur 10.

Exemple illustratif

Nous procéderons alors au tirage au sort, suivant le principe de « la roue de la Fortune »: 2/10 de la roue sont attribués à l'individu 1, 1/10 à l'individu 2, 3/10 à l'individu 3, et 4/10 à l'individu 4. La roue est lancée autant de fois qu'il y a d'individus dans la population. Supposons que le tirage ait donné: l'individu 4 (2 fois), l'individu 3 (1 fois) et l'individu 1 (1 fois). Nous sommes maintenant en présence d'une population intermédiaire: ((1;1;1), (1;1;1), (0;0;1), (1;0;0)}, respectivement appelés { 1bis, 2bis, 3bis, 4bis}.

Troisième phase : *reproduction*

Au cours de cette phase, les couples sont choisis au hasard. Supposons que 1bis soit réunis avec 4bis et 2bis avec 3bis. Des points de croisement sont ensuite choisis au hasard:

1 1 | 1 et 1 | 1 1

1 0 | 0 0 | 0 1

ce qui donne:

1 1 0 et 1 0 1

1 0 1 0 1 1

Exemple illustratif

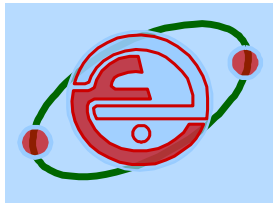
Les 4 nouveaux membres sont donc: { (1;1;0),(1;0;1);(1,0,1);(0;1;1) }

Ils subissent alors des mutations (en général, la probabilité de mutation est très faible). Supposons que seule la caractéristique 2 du 3ème individu soit mutée. Finalement, la nouvelle population formant la deuxième génération, sera constituée des 4 individus ((1;1;0),(1;0;1);(1,1,1);(0;1;1)}. Nous remarquerons que l'individu idéal y est maintenant présent.

Ensuite, en recommençant itérativement les 3 phases ci-dessus, nous tendrons vers une population constituée presque exclusivement de (1;1;0) (qui a une fonction d'adaptation supérieure aux autres)

Notons cependant qu'il y aura toujours quelques mutants pour préserver la diversité de la population et donc éventuellement réagir à un changement de la fonction d'adaptation suite à un changement quelconque de l'environnement.

Université Badji Mokhtar - Annaba
Département d'Informatique



جامعة باجي مختار – عنابـــــة
قسم الإعـلام الآلـــي

La classification non supervisée

Cours d'Apprentissage Automatique
Master 1 STIC, 2014-2015

Présenté par
Pr Souici – Meslati L.

Plan

- Introduction
- Principe et objectifs
- Applications
- Intérêts
- Etapes
- Classification non hiérarchique ou partitionnement: méthode des K-Means

Introduction

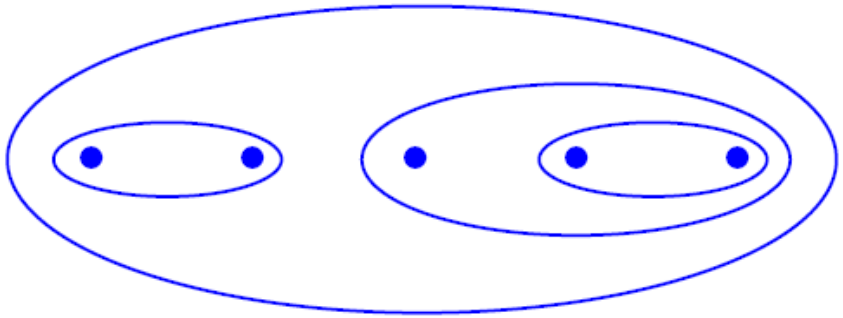
- La classification non supervisée (classification automatique ou clustering) est une méthode d'apprentissage non supervisé : les exemples sont dépourvus d'une étiquette (classe) fournie par un expert.

- La problématique de la classification automatique est simple : étant donné un certain nombre d'objets décrits par des attributs, est-il possible d'identifier les familles dans lesquelles se regroupent naturellement ces objets ?

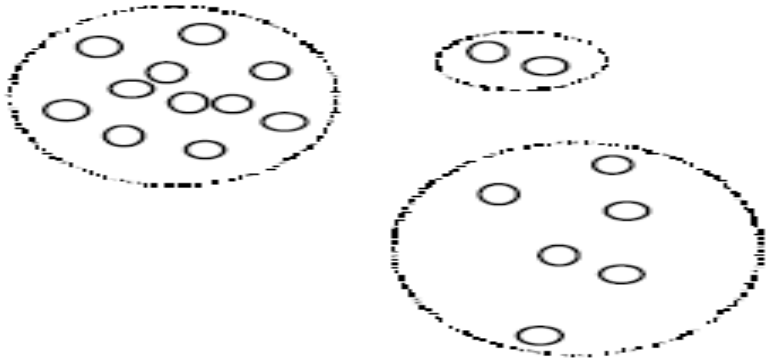
Introduction

- Techniquement, il y a 2 manières d'aborder cette problématique :

- Organiser les données selon une hiérarchie de classes ou de familles (comme c'est le cas en sciences naturelles)



- Faire l'hypothèse qu'il existe un certain nombre de classes dans les données et chercher à les partitionner le mieux possible en autant de sous-ensembles disjoints.



Principe et objectif

Principe : Constituer des groupes « naturels » de manière à ce que les individus dans un même groupe se ressemblent le plus possible , et les individus dans des groupes différents soient dissemblables, càd soient les plus différents possible.
L'objectif principal est donc de constituer des groupes d'objets *homogènes et différenciés*, càd des groupes d'objets tels que :

- les objets soient les plus similaires possibles au sein d'un groupe pour assurer une grande *homogénéité* de chaque classe (*critère de compacité*),
- les groupes soient aussi dissemblables que possible pour assurer une bonne *séparation* des classes (*critère de séparabilité*)

Remarque: la ressemblance ou la dissemblance est mesurée sur l'ensemble des variables descriptives.

Applications

Hypothèse

On suppose qu'une structure de classes existe au sein de la population étudiée, le but de l'analyse est de l'identifier et de la mettre à jour.

Exemples traités avec les méthodes de classification automatique:

- Classification des consommateurs de différents produits pour identifier des groupes d'individus ou de ménages ayant un comportement homogène vis-à-vis de la consommation de différentes marques ou variétés, de l'attitude par rapport à un produit…
- Classification de la clientèle d'une banque,
- Classification des communes algériennes,…

Remarques

- Une classification automatique obtenue sur un ensemble n'est pas **LA** classification de cet ensemble, mais **une** classification (parmi beaucoup d'autres) établie à partir de variables et de méthodes choisies.

- L'apprentissage non supervisé évite l'assistance d'un opérateur mais n'assure pas toujours une classification correspondant à la réalité (celle de l'utilisateur)

Intérêts de la classification

Pourquoi constituer les groupes?

- Identifier des groupes d'individus ayant un comportement (ou des caractéristiques) homogènes
- Mettre en évidence les principales structures dans les données (définir des « concepts »)
- Affecter de nouveaux individus à des catégories
- Identifier les cas totalement atypiques

Ainsi, la classification automatique a plusieurs intérêts:

- Les classes obtenues assurent une vue concise et structurée des données.
- Des regroupements inattendus apparaissent.
- Des regroupements attendus n'existent pas.
- Les classes significatives entraînent la définition de fonctions de décision permettant d'attribuer un nouvel individu à la classe dont il est le plus proche.

Etapes de classification automatique

Les principales étapes d'une classification automatique sont:

1. Choix des données et leur représentation.
2. Calcul des dissimilarités entre les *n* individus à partir du tableau initial.
3. Choix d'un algorithme de classification et son exécution.
4. L'interprétation des résultats :
   - évaluation de la qualité de la classification
   - description des classes obtenues

Etape 1. Choix des données

La classification obtenue est liée aux variables choisies pour décrire les individus. On distingue:

- les variables actives, celles sur lesquelles sera basée la classification des individus
- les variables illustratives (ou supplémentaires) qui serviront à décrire les classes constituées

Les données sur lesquelles la classification automatique est appliquée se présentent sous l'une des formes suivantes:

- *N* objets (ou individus) caractérisés par *p* descripteurs
- tableau carré symétrique de ressemblances ou de dissemblances (similarités, dissimilarités, distances).

Etape 2. Mesures de ressemblance ou dissemblance

Il existe un grand choix de mesures de ressemblance ou de dissemblance. Le tableau obtenu est un tableau carré de dimension *n*.

Variables quantitatives: La distance euclidienne est une mesure possible de la ressemblance.

Variables qualitatives: De nombreux indices de ressemblance ont été proposés. Dans le cas d'objets décrits par des variables binaires: indice de Jaccard, indice de Russel et Rao…

Etape 3. Les méthodes de classification automatique

Il y a 2 manières d'aborder la problématique de la classification non supervisée:

- Soit en construisant une hiérarchie de classes (méthodes de classification hiérarchique) de manière ascendante ou descendante.
- Soit directement en cherchant un nombre fixé de classes (méthodes non hiérarchiques ou de partitionnement direct).

Une classification hiérarchique est souvent préférable lorsqu'on désire effectuer une analyse détaillée des données tandis que les méthodes non hiérarchiques sont plus rapides et choisies lorsque le nombre de données est grand.

Etape 3. Les méthodes hiérarchiques

- La classification ascendante hiérarchique (CAH) conduit à la construction d'un **arbre de classification** (ou **dendrogramme**) montrant le passage des *n* individus au groupe «total »par une succession de regroupements.
- La classification descendante hiérarchique procède à l'inverse par subdivisions successives de l'ensemble à classer.

Remarque: On peut obtenir une partition à partir d'une hiérarchie (partitionnement indirect).

Etape 3. Les méthodes non hiérarchiques

Ces méthodes appelées aussi méthodes de partitionnement direct, cherchent à opérer, au sens d'un critère donné, le «meilleur »regroupement possible des individus en un nombre choisi a priori de classes.

Le principe général de ces méthodes est la constitution de *k* groupes (*k* étant un nombre choisi par l'analyste) à partir des *n* individus sur la base d'un algorithme itératif «Allocation/Recentrage» en essayant d'optimiser un indice global mesurant la qualité de la classification.

Etape 4. Interprétation des résultats

Il existe plusieurs manières d'interpréter les résultats obtenus par la classification automatique: par les individus, par les variables, par les groupes de variables…

Par les individus

Pour chaque classe, on examine :

- son effectif,
- son diamètre (distance entre les 2 points les plus éloignés)
- la séparation (distance minimum entre la classe considérée et la classe la plus proche) et le numéro de la classe la plus proche
- les identités des individus les plus proches du centre de gravité de la classe ou «parangons»
- les identités des l'individus les plus éloignés du centre de gravité de la classe ou «extrêmes».

Etape 4. Interprétation des résultats

Par les variables :une par une

On calcule un critère mesurant la pertinence de chaque variable de façon isolée pour interpréter la classe.

Exemple :

prix ..... critère fort pour cette classe

âge ..... critère faible pour cette classe

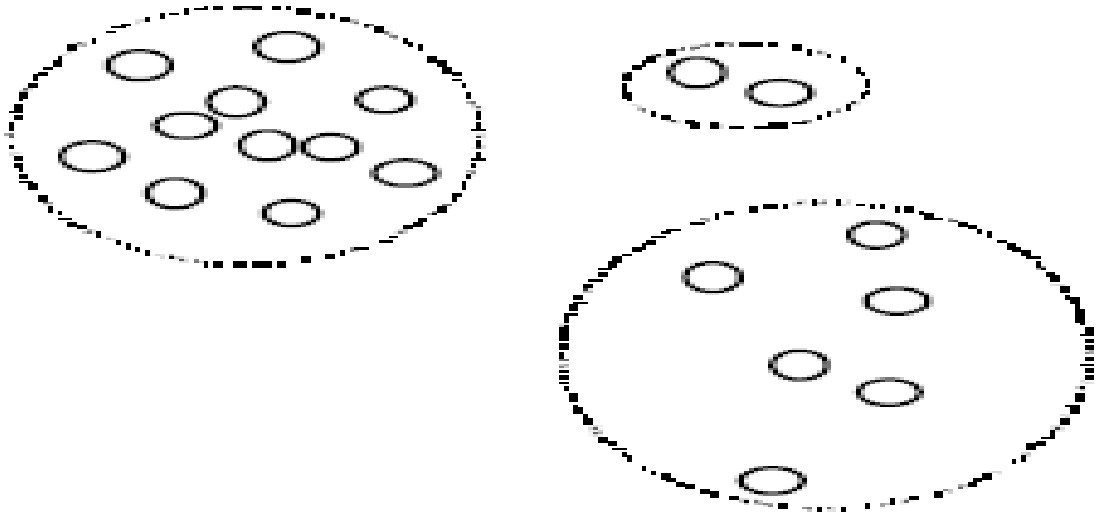
On se pose des questions telles que:

Est-ce que tous les éléments de la classe ont certaine(s) valeur(s)de cette variable (condition nécessaire d'appartenance à la classe) ?

Est-ce que certaine(s) valeur(s) de cette variable ne se rencontrent que dans cette classe (condition suffisante d'appartenance à la classe) ? ...

Méthodes de Classification non hiérarchique (partitionnement)

- Les méthodes de partitionnement (non hiérarchiques), permettent de subdiviser l'ensemble des données en un certain nombre (connu) de classes. Ces classes doivent être d'intersections vides et chacune doit être représentée par un noyau. Les manières de calculer la proximité d'un individu et d'une classe sont variables. Ces méthodes permettent de traiter assez rapidement des données de taille importante.



Méthodes de Classification non hiérarchique (partitionnement)

- Notons que le nombre réel de classes k est très souvent inconnu et qu'il a en outre une définition peu claire : on ne sait que très rarement déterminer le nombre de classes optimal.
- Bien souvent ce nombre sera prédéfini (il correspondra par exemple à un nombre de catégories déjà existantes) ou calculé en fonction de l'utilisation qui est faite de la classification (utilisée pour la recherche documentaire, le meilleur nombre de classes est par exemple celui qui permet d'obtenir la meilleure précision du système de recherche). L'estimation du nombre de classes suppose un certains nombre d'hypothèses ou de connaissances sur les données manipulées.
- La meilleure valeur de k peut être déterminée expérimentalement.

Méthodes de Classification non hiérarchique (partitionnement)

- Les méthodes de partitionnement sont itératives : à partir d'une partition initiale les individus sont affectés à la classe qui leur est la plus proche. Les difficultés dans l'implémentation de ces méthodes réside dans la manière de choisir une partition initiale et de définir la proximité entre un individu et une classe (choix du mode de représentation des classes et de la distance employée).

- Un des principaux inconvénients de ces méthodes réside dans le fait que **la partition finale dépend de la partition initiale**.

- On distingue plusieurs méthodes de partitionnement, la plus connue est la méthode des K-moyennes (K-means)

Méthode des k-Moyennes (k-Means)

La méthode des k-Means est l'une des méthodes non hiérarchiques les plus connues. Les centres des classes sont calculés après chaque affectation d'un individu dans une classe, les centres sont des **points de l'espace,** les distances entre les individus et les classes correspondent aux distances entre les individus et les centres.

La première partition est généralement obtenue en tirant au hasard *k* points *(k « centres ») c_1,..., c_k de E (l'ensemble des individus à classer). Ces k* points définissent une partition de E en *k* classes : C_1, ..., C_k *:* chaque classe contient les individus de E plus proches de son *centre* que de tout autre *centre*. Les k premiers points sont alors remplacés par les k centres de gravité (véritables *centres* cette fois, qui sont des points de l'espace et pas forcément des individus) des classes C_1, ..., C_k.

Lors d'une nouvelle itération chaque individu est alors affecté à la classe qui lui est la plus proche et on recommence. La vitesse de convergence de l'algorithme dépend des *k centres* initiaux.

Méthode des K-Moyennes (K-Means)

K-Means est applicable à des données numériques (réelles), son principe peut être résumé comme suit:

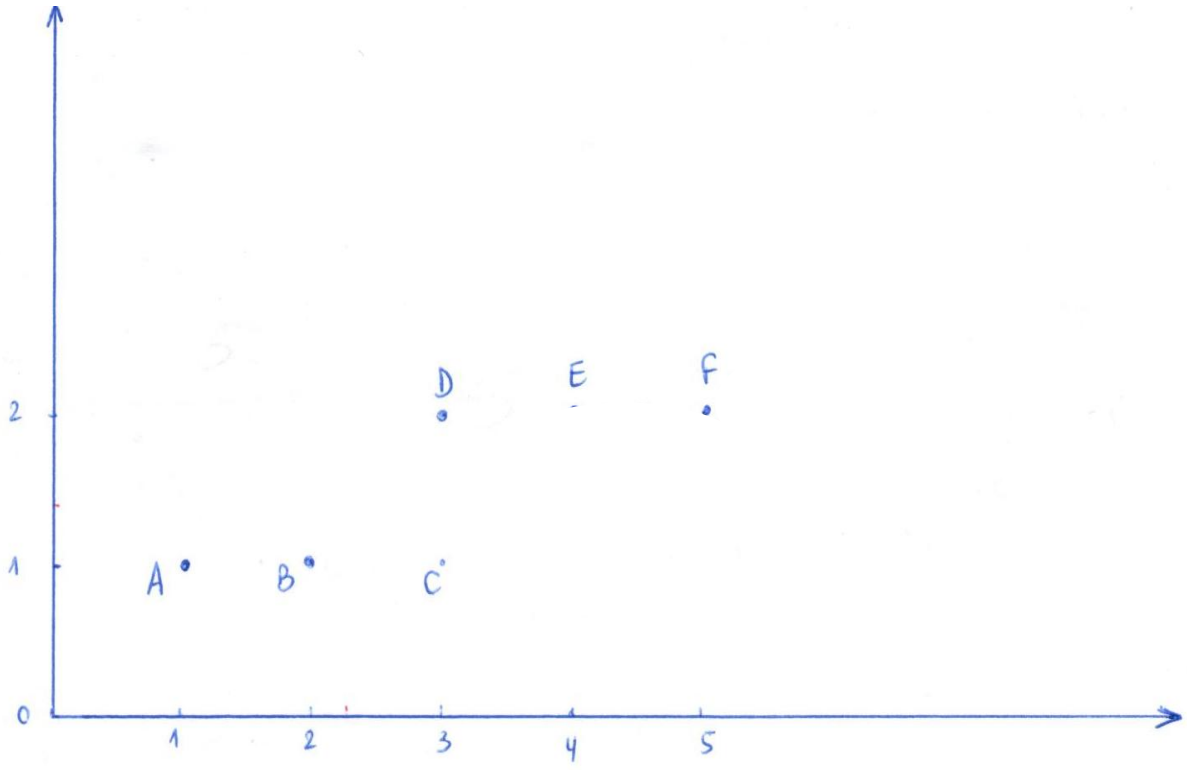
- Soient N objets numérotés de 1 à N. On cherche à les regrouper en k classes (faire une partition à k blocs). On commence par en tirer k au hasard et répartir les objets en blocs selon leur distance (euclidienne) aux k objets choisis au hasard.
- Le partitionnement se poursuit avec les centres de gravité des classes et le processus se répète jusqu'à la convergence (plus de changements dans les partitions).

Exercice sur la méthode de K-moyennes

Soient 6 objets représentés par les points suivants:
A(1,1), B(2,1), C(3,1),
D(3,2), E(4,2), F(5,2)

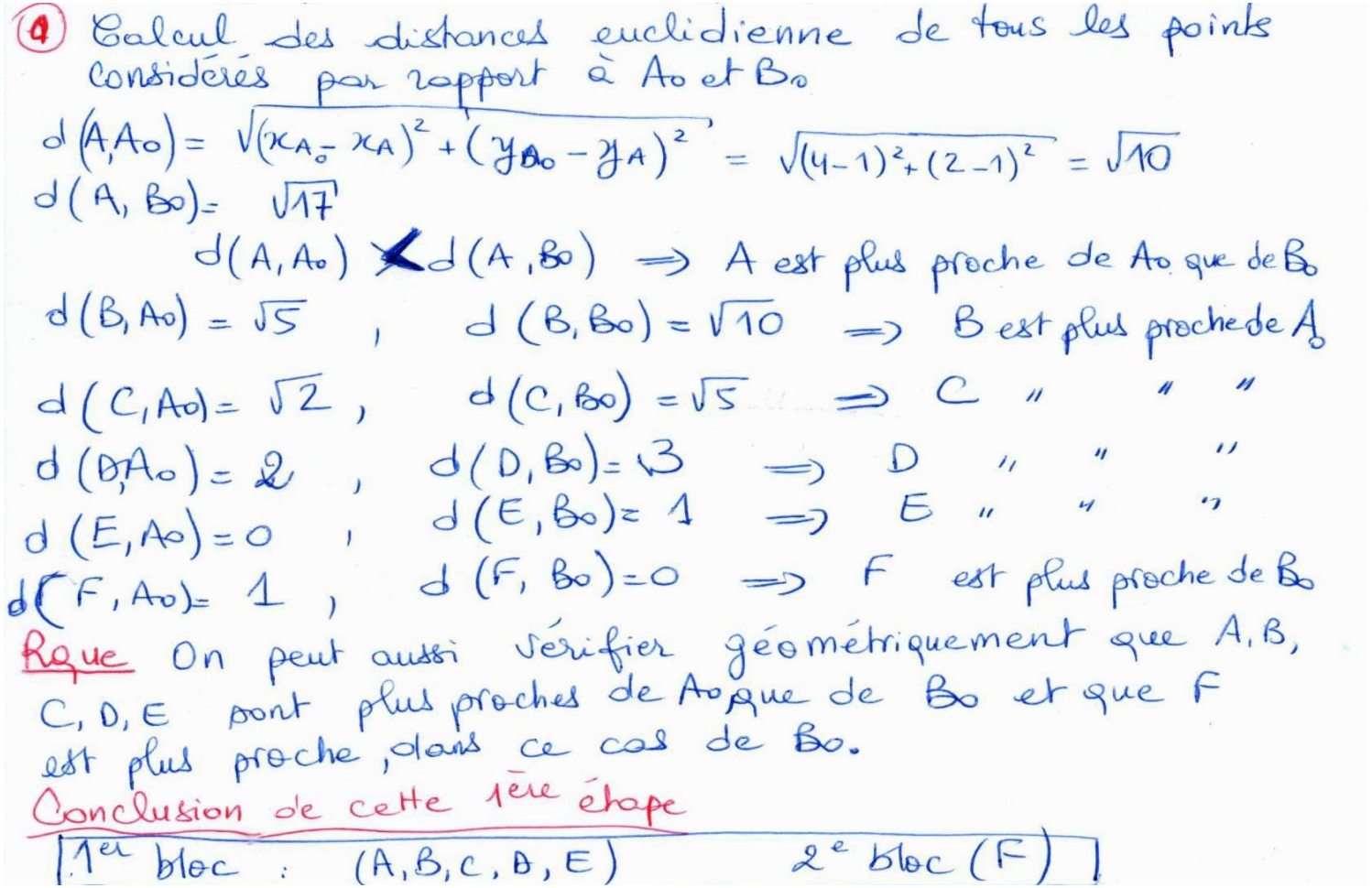
1. Appliquer l'algorithme des 2-moyennes (k-moyennes avec k=2, en utilisant la distance euclidienne) à cet ensemble de points en supposant que les deux objets tirés au hasard au départ sont A_0=E et B_0=F.

2. Même question si on suppose que A_0=B et B_0=E



Correction exercice sur la méthode des K-moyennes

1. A_0=E(4,2) et B_0=F(5,2)



④ Calcul des distances euclidienne de tous les points considérés par rapport à A_0 et B_0

$d(A,A_0) = \sqrt{(x_{A_0} - x_A)^2 + (y_{B_0} - y_A)^2} = \sqrt{(4-1)^2 + (2-1)^2} = \sqrt{10}$

$d(A, B_0) = \sqrt{17}$

$d(A,A_0) < d(A,B_0) \Rightarrow$ A est plus proche de A_0 que de B_0

$d(B,A_0) = \sqrt{5}$, $d(B,B_0) = \sqrt{10} \Rightarrow$ B est plus proche de A_0

$d(C,A_0) = \sqrt{2}$, $d(C,B_0) = \sqrt{5} \Rightarrow$ C " " "

$d(D,A_0) = 2$, $d(D,B_0) = 3 \Rightarrow$ D " " "

$d(E,A_0) = 0$, $d(E,B_0) = 1 \Rightarrow$ E " " "

$d(F,A_0) = 1$, $d(F,B_0) = 0 \Rightarrow$ F est plus proche de B_0

Rque On peut aussi vérifier géométriquement que A, B, C, D, E sont plus proches de A_0 que de B_0 et que F est plus proche, dans ce cas de B_0.

Conclusion de cette 1ère étape

1er bloc : (A, B, C, D, E)	2e bloc (F)

Correction exercice sur la méthode de K-moyennes

(b) Calcul des centres de gravité des 2 classes.

Pour la classe (A, B, C, D, E), le centre de gravité sera :

$$A_1\left(\frac{x_A + x_B + x_C + x_D + x_E}{5}, \frac{y_A + y_B + y_C + y_D + y_E}{5}\right) \Rightarrow A_1(1.6, 1.4)$$

Pour la classe (F), le centre de gravité sera $B_1 = F(5, 2)$

(c) On poursuit le partitionnement avec les nouveaux centres

$A_1(1.6, 1.4)$ et $B_1(5, 2)$

Après calcul des distances euclidiennes $d(A, A_1)$, $d(A, B_1)$, $d(B, A_1)$, $d(B, B_1)$ ..... $d(F, A_1)$, $d(F, B_1)$ de la même manière que précédemment, on obtient les nouveaux blocs : 1er bloc (A, B, C, D) 2e bloc (E, F)

dont les centres de gravité sont, respectivement,

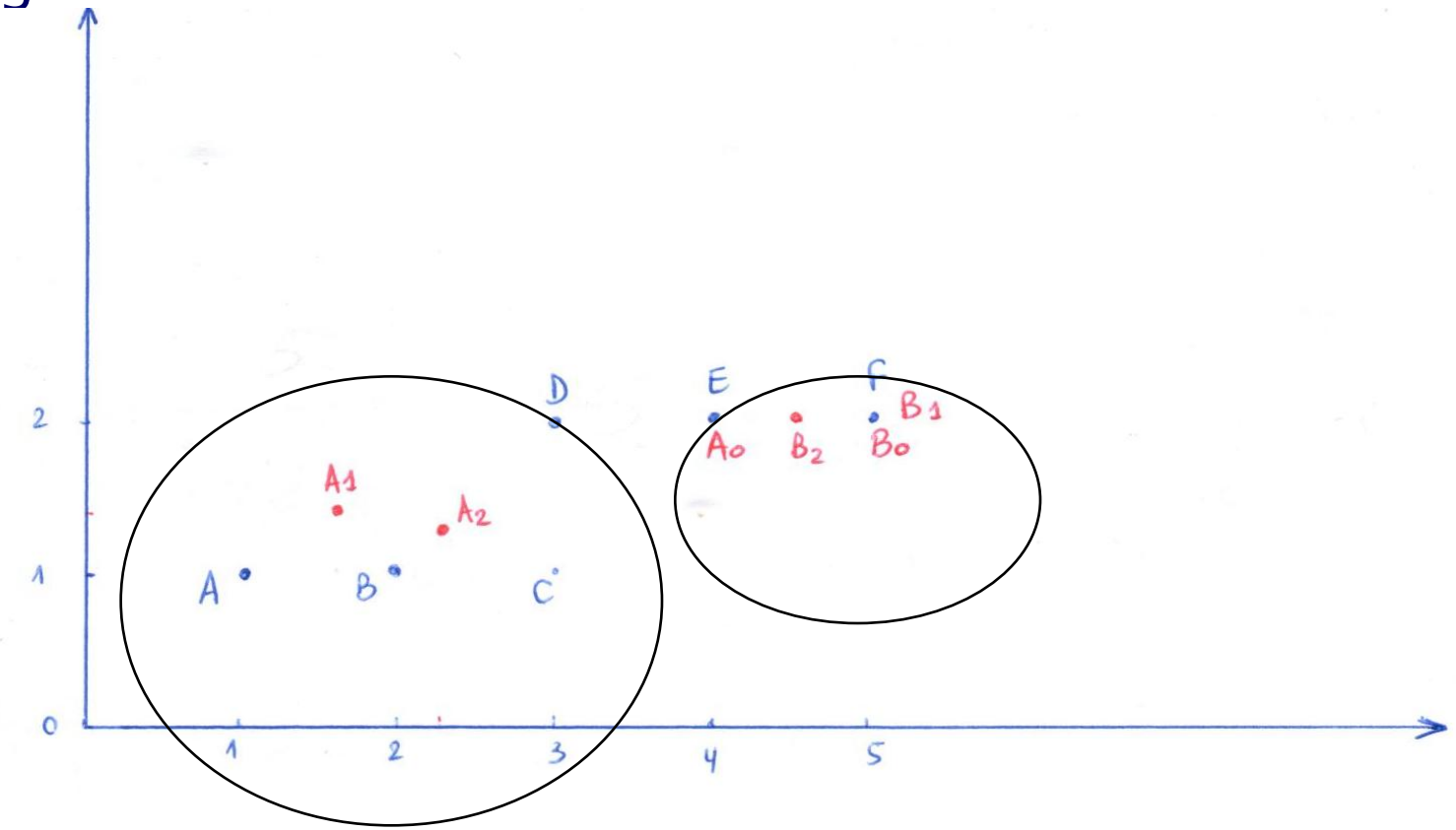
$A_2(2.25, 1.25)$ et $B_2(4.5, 2)$

Correction exercice sur la méthode de K-moyennes

(d) On poursuit le partitionnement avec A_2 et B_2.
Après calcul des distances euclidiennes $d(A,A_2)$, $d(A,B_2)$ ..... $d(F,A_2)$, $d(F,B_2)$, on remarque que A, B, C, D sont plus proches de A_2 et E, F sont plus proches de B_2
Donc, cette nouvelle itération ne change plus la partition (A, B, C, D) (E, F) : l'algorithme a convergé.

Correction exercice sur la méthode de K-moyennes

Donc, dans le 1er cas, avec A_0=E(4,2) et B_0=F(5,2), on obtient les classes C1=ABCD, C2=EF avec convergence de l'algorithme à la 3ème itération.



Correction exercice sur la méthode de K-moyennes

2ème cas. A_0=B(2,1) et B_0=E(4,2)

Après calcul des distances, on obtient la partition

(A, B, C) (D, E F)

Centres de gravités respectifs

A_1(2,1) B_1(4,2)

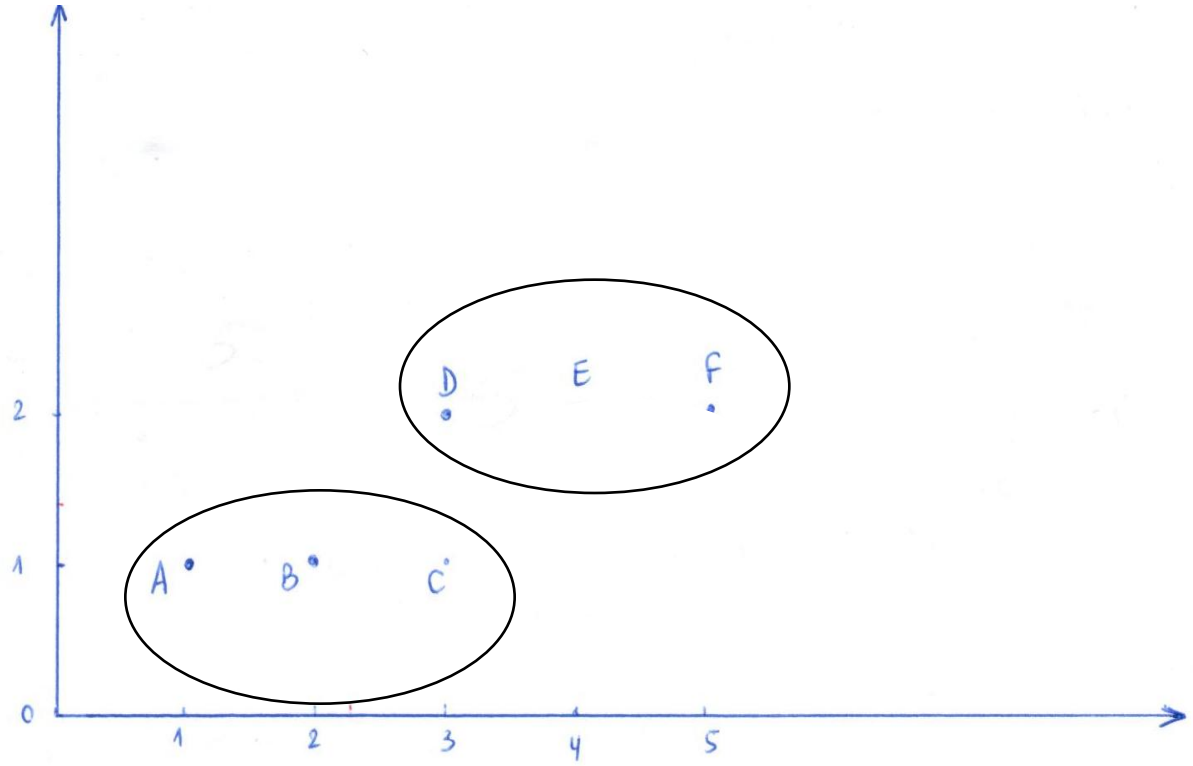
On remarque que :

$A_1 = A_0 = B$ et que $B_1 = B_0 = E$

Donc, avec cette initialisation (A_0=B et B_0=E), l'algo de partitionnement converge directement.

Correction exercice sur la méthode de K-moyennes

Donc, dans le 2ème cas, avec A_0=B(2,1) et B_0=E(4,2)



On obtient les classes C1=ABC, C2=DEF avec convergence de l'algorithme dès la première itération