University Mohamed Boudiaf of M'sila
Faculty of Mathematics and Computer Science

Introduction to Artificial Intelligence

Dr. SAID KADRI

Associate Professor "Class A"

Department of Computer Science, Faculty of Mathematics and Informatics,

University Mohamed Boudiaf of M'sila

E-mail: kadri.said28@gmail.com

Website: https://kadrisaid28.wixsite.com/sgadri

2020 – 2021

1. Introduction à l'Intelligence Artificielle

Préface

1. Qu'entendez-vous du terme Intelligence ?

(Prendre quelques définitions données par les étudiants)

2. Est-ce que l'ordinateur est intelligent ? pour quoi ?

3. Et l'être humain ?

1. Qu’entendez-vous du terme Intelligence ?

- L’intelligence est la capacité de résoudre des problèmes.

- L’intelligence est la capacité de résoudre des problèmes complexes avec efficacité et dans un temps restreint.

- L’intelligence est la capacité d’innover et de prendre de décisions à tout moment vis-à-vis un problème donné.

2. Est-ce que l’ordinateur est intelligent ? pour quoi ?

- L’ordinateur n’est pas intelligent !!! Il ne fait rien qu’exécuter les instructions du programme préparé par un humain.
- Il ne peut pas innover ou prendre des nouvelles décisions.

3. Et l’être humain ?

- L’intelligence est l’une des caractéristiques les plus importantes de l’être humain.
- Capable de prendre des nouvelles décisions à tout moment et s’adapter avec les variations de l’environnement.

Comparaison entre l'humain et l'ordinateur

Caractéristique	Homme	Ordinateur
1. Rapidité		
2. Fatigue		
3. Oublie		
4. Précision		
5. Mémorisation		
6. Apprentissage		
7. Innovation et adaptation		

Caractéristique	Homme	Ordinateur
1. Rapidité	Relativement lent	Très rapide
2. Fatigue	Il peut se fatiguer ou s'ennuyer après un certain temps de travail	jamais
3. Oublie	Il peut oublier des connaissances au cours du temps	Jamais (sauf dans le cas de pannes)
4. Précision	Il peut se tromper lors de calcul	Très précis
5. Mémorisation	Relativement illimitée	Limitée par la capacité de ses dispositifs de stockage
6. Apprentissage	D'un jour à l'autre, il peut apprendre des nouvelles connaissances	Mauvais (demande de l'IA)
7. Innovation et adaptation	Il peut prendre des nouvelles décisions à tout moment	Très faible, il ne peut rien faire d'autre que suivre les instructions du programme

Quelques définitions de l'IA (spécialisées)

- Le terme intelligence artificielle (IA) signifie : choisir parmi plusieurs (décider), comprendre, percevoir (acquérir des connaissances) et savoir (apprentissage).

- L'IA est le domaine qui étudie comment exécuter par l'ordinateur des taches pour lesquelles l'humain parait le meilleur.

- Une discipline visant à comprendre la nature de l'intelligence en construisant des programmes ordinateurs simulant l'intelligence humaine.

Définition générale

- A la lumière de définitions précédentes, on peut dire que l'IA traite des problèmes pour lesquels l'être humain parait plus efficace et performant. Ces problèmes font intervenir les différents sens : vision, écoute, parole, intuition.
- L'IA vient pour compléter le manque qui figure sur la machine en terme : innovation, prise de décision, adaptation avec la variabilité de l'environnement, apprentissage.
- Le but de l'IA est d'avoir des machines et des programmes plus au moins intelligents.

Observation

- L'informatique classique traite ***les données*** (informations numériques).
- L'intelligence artificielle traite ***les connaissances*** (informations symboliques).

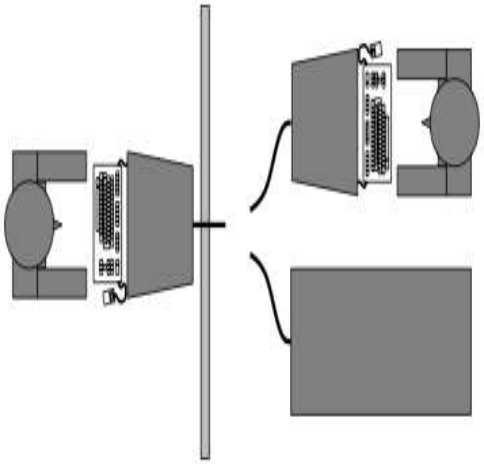
Comparaison des méthodes de l'intelligence artificielle et des méthodes de l'informatique classique.

Méthodes classiques	Méthodes de l'IA
Près du fonctionnement de la machine	Près du fonctionnement humain
Traitement de nombres ou de textes	Traitement de symboles
Utilisent beaucoup de calculs	Utilisent beaucoup d'inférences.
Suivent des algorithmes rigides et exhaustifs	Font appel à des heuristiques et des raisonnements incertains
Ne sont généralisables qu'à une classe de problèmes semblables	Sont généralisables à des domaines complètement différents.

Comment tester l'intelligence d'une machine

1. *Machine de Turing (1950)*

Un être humain interroge à la fois un agent humain (personne) et un agent artificiel (une machine) sans les voir au travers d'une interface : si les réponses données ne lui permettent pas de distinguer l'agent artificiel de l'agent humain alors l'agent artificiel est déclarée «intelligent».



L'homme:

Question : "What is 35,076 divided by 4,567?"

Answer : ????

La machine:

Question : "What is 35,076 divided by 4,567?"

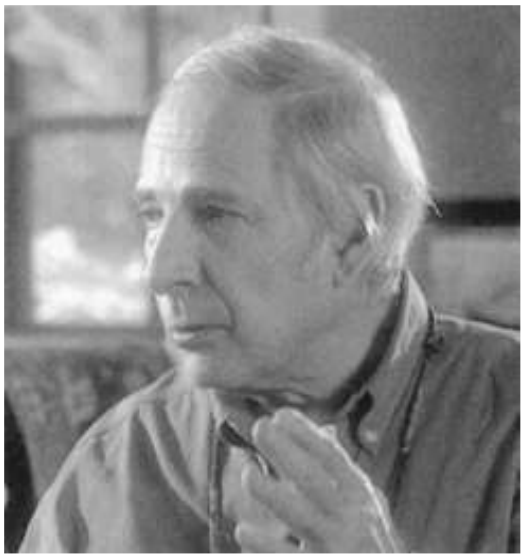
Answer : 7.6803153

Turing Test Homepage :***http://cogsci.ucsd.edu/~asaygin/tt/ttest.html***

2. *Chambre chinoise (Minds, Brains and programs/John Searle, Prof en Philosophy/Berkeley; 1980)*

John Searle est enfermé dans une pièce ne communiquant avec l'extérieur que par un guichet et contenant un très gros livre dans lequel est écrit une succession de questions et leurs réponses pertinentes (convenable), et rédigées en chinois.

Un expérimentateur lui transmet des messages par le guichet, tantôt en anglais, tantôt en chinois. Searle répond directement aux messages rédigés en anglais, alors que ceux rédigés en chinois, il est obligé de consulter le livre jusqu'à trouver une question identique au message, il recopie alors la réponse associée.
La même chose pour être faite en remplaçant l'humain qui répond aux questions par une machine pour tester son degré d'intelligence.



Conclusion :

- Une machine sera considérée comme intelligente si elle reproduit le comportement d'un être humain dans un domaine spécifique ou non.
- Une machine sera considérée comme intelligente, si elle modélise le fonctionnement d'un être humain.

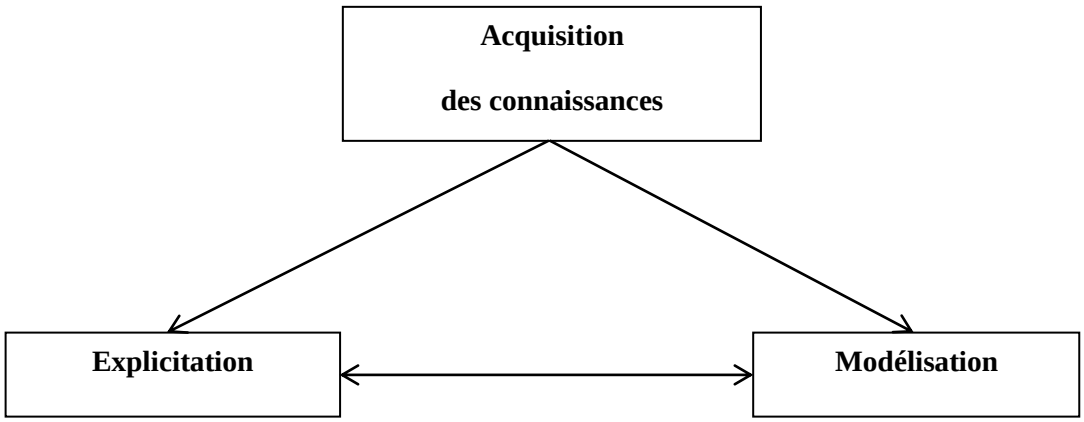
Quelques domaines de l'IA

1. Apprentissage machine
2. Représentation de connaissances
3. Traitement du Langage Naturel TLN/TALN
4. Reconnaissance des formes
5. Reconnaissance de l'écriture
6. Reconnaissance de la parole
7. Calcul formel
8. La simulation du raisonnement humain
9. Résolution de problèmes complexes
10. Enseignement assisté par ordinateur EAO
11. Systèmes experts
12. Robotique et FAO (Fabrication Assistée par Ordinateur)
13. Réalité virtuelle et Réalité augmentée
14. Vie artificielle

Exemples d'applications de l'IA

1. Diagnostic médical : thérapie, surveillance d'appareils
2. Synthèse d'images : vision par ordinateurs (robots).
3. Classification naturelle : (biologie, minéralogie, ...) (les SE).
4. Planification de taches : prédictions financières, …etc.
5. Architecture : CAO, DAO
6. Détection de pannes: le système Sherlock pour les avions F16
7. Education : e-learning
8. Génie : vérification de règles de conception.
9. Prospection géologique : gisements miniers.
10. Centrales nucléaires, feux de forêts : systèmes à temps réel.
11. Simulation de vols : les systèmes CAE, Bombardier
12. Jeux vidéo.

Processus d’acquisition de connaissances



Types de connaissances

a) Connaissances déclaratives (le savoir)

La population de L’Algérie était de 38.000 000 habitants en 2011.

b) Connaissances procédurales (le savoir-faire)

Une recette de cuisine =>des étapes pour faire un gâteau au chocolat.

c) Connaissances conceptuelles (combine les deux)

Le concept de référendum (élections) fera appel aux connaissances procédurales : installation des bureaux de votes et constitution des listes électorales ;

La connaissance déclarative est : un bulletin nul n'est pas compté.

Évolution de processus d'acquisition de connaissances

Techniques mannuelles	Techniques semi-automatiques	Apprentissage machine

Formalismes de représentation de connaissances

1. Logique des propositions (pas de quantificateurs et pas de variables)
2. Logique des prédicats de premier ordre (introduction de variables et de quantificateurs)
3. Règles de production
4. Réseaux sémantiques
5. Objets structurés
6. Logiques terminologiques KL-ONE
7. Graphes conceptuels
8. Réseau de Pétri
9. Réseaux de neurone
10. Autres, …

3. Apprentissage Automatique (Machine Learning)

- L'apprentissage automatique (machine learning) est de prédire le futur en se basant sur le passé. Par exemple, on peut prédire comment l'utilisateur Omar choisira un film qu'il n'a pas déjà vu, en se basant sur ses avis relatifs à des films qu'il a vu (le passif).
- Alors, prédire le futur en se basant sur le passé (un ensemble d'exemples) est dans le cœur de tous les algorithmes de l'apprentissage automatique.

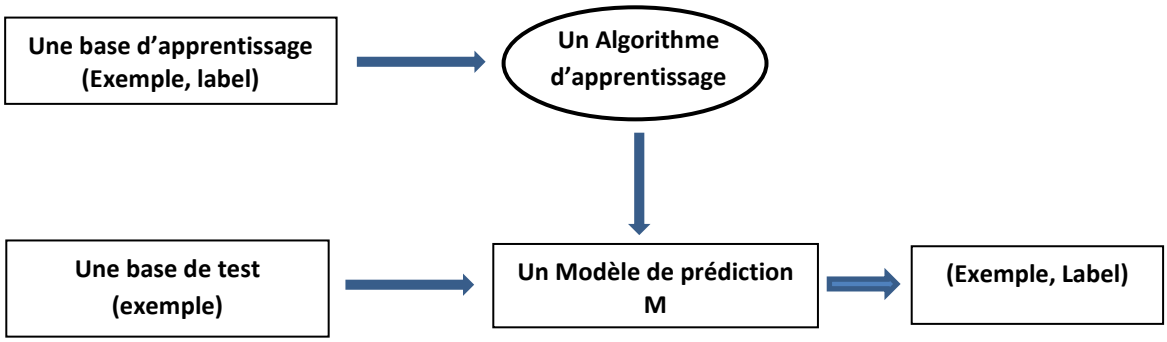


Figure 1.6. Principe de l'apprentissage automatique

- L'objective de l'apprentissage inductive est de prendre quelques données d'apprentissage et les utiliser pour construire le modèle de prédiction (une fonction f). Cette fonction sera évaluée sur la base de test. L'algorithme d'apprentissage est considéré valide et efficace si sa performance sur la base de test est élevée.

Exemples de problèmes typiques de prédiction

1. ***Régression :*** essayer de prédire une valeur réelle. Par exemple, prédire la valeur du stock demain en se basant sur son ancienne performance (statistiques).

2. ***Classification binaire:*** essayer de prédire une simple réponse (Oui/non). Par exemple, prédire si Omar préfère un module ou non.
3. ***Classification Multi-classes:*** essayer de classer un exemple dans un parmi plusieurs classes. Par, exemple, prédire le theme d'un article (Politique, Sport, Economie, Religion, Cinéma, Loisirs, etc).

Types d'apprentissage

1. ***Apprentissage supervisé :*** le processus de classification est effectué sur des classes prédéfinies (catégorisation)
2. ***Apprentissage non-supervisé :*** la classification est faite sur des classes non prédéfinies mais crées par le système de classification lui-même (segmentation/clustering).

Processus de classification

Deux étapes :

Etape 1 : Construction du modèle de prédiction à partir de l'ensemble d'apprentissage (training set) donné au départ (le passif)

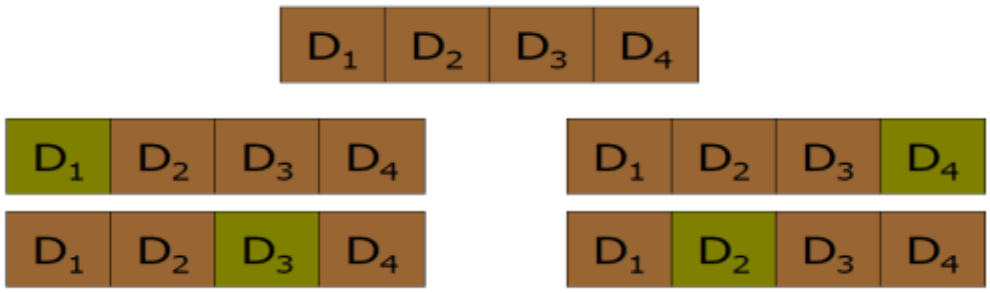
Etape 2 : Utilisation du modèle construit sur l'ensemble de test (test set) pour tester sa précision, puis l'utiliser dans la classification de nouvelles données (l'actuel/le présent).

Remarques :

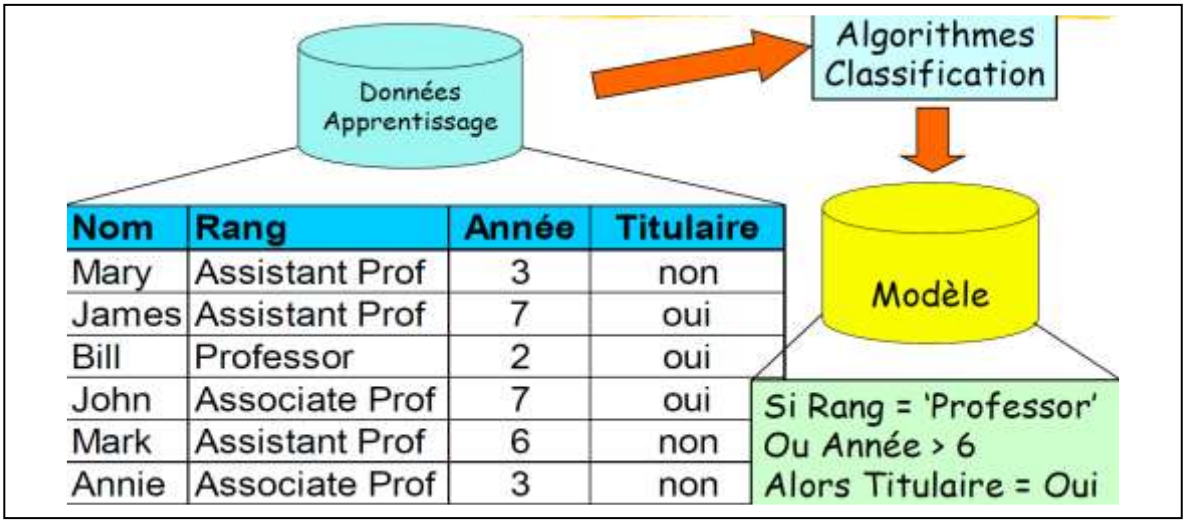
- Pour l'ensemble d'apprentissage chaque instance (exemple) est donnée sous forme d'un couple (instance, classe), c.à.d, la classe de l'élément est connue.

- Pour l'ensemble de test la classe de chaque instance (exemple) est inconnue ==> sera déterminée à la fin du processus de classification en se basant sur le modèle prédictif construit, et de même pour un nouvel élément.
- La classification est réalisée sur les nouvelles instances avec un éventuel taux d'erreur qui reflète la performance du modèle construit (taux d'erreur = taux des instances mal classées).
- Généralement, 2/3 des données sont utilisés pour l'apprentissage (construction du modèle) et 1/3 est utilisé pour le test (validation du modèle).
- L'une des méthodes utilisées pour la validation du modèle est la méthode de validation croisée qui consiste à :

- Diviser les données en k sous-ensembles
- Utiliser (k-1) sous-ensembles comme données d'apprentissage et un (01) sous-ensemble comme données de test.

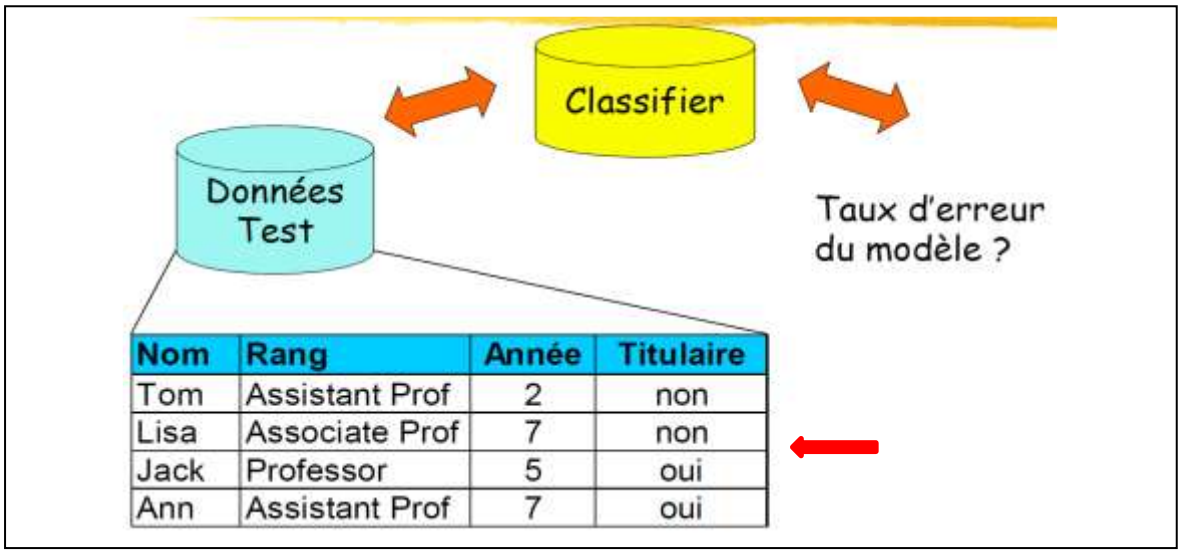


Exemple : Construction d'un modèle prédictif à partir de données



Nom	Rang	Année	Titulaire
Mary	Assistant Prof	3	non
James	Assistant Prof	7	oui
Bill	Professor	2	oui
John	Associate Prof	7	oui
Mark	Assistant Prof	6	non
Annie	Associate Prof	3	non

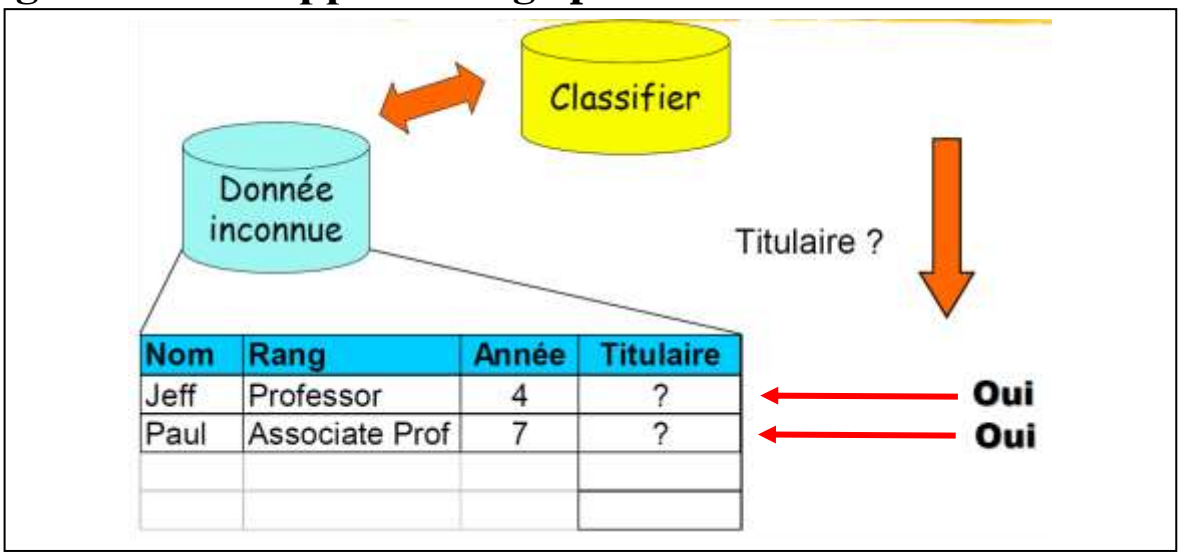
Exemple : Utilisation du modèle (validation du modèle)



Nom	Rang	Année	Titulaire
Tom	Assistant Prof	2	non
Lisa	Associate Prof	7	non
Jack	Professor	5	oui
Ann	Assistant Prof	7	oui

Exemple : Utilisation du modèle (classification de nouveaux éléments)

Algorithmes d'apprentissage pour classification



Nom	Rang	Année	Titulaire
Jeff	Professor	4	?
Paul	Associate Prof	7	?

Plusieurs algorithmes peuvent être envisagés, à noter :

- Méthode K-NN (k plus proches voisins)
- Arbres de décision
- Réseaux de neurones
- Méthode bayésienne
- Support Vector Machine SVM

1. Méthode K-NN :

- Son principe est le suivant : la classe d'un élément e_i est celle des *k* plus proches éléments à e_i.
- Modèle = échantillon d'apprentissage (beaucoup d'exemples) + une fonction de calcul de distance (similarité) + une fonction de choix de la classe en fonction des classes des voisins les plus proches (classe majoritaire).

Algorithme kNN (K-nearest neighbors)

Objectif : affecter une classe à une nouvelle instance

Début

Données d'entrée :

- Un ensemble d'apprentissage de *m* exemples classés sous forme de couple *(x, c(x))*
- Un nouvel exemple y

Traitement :

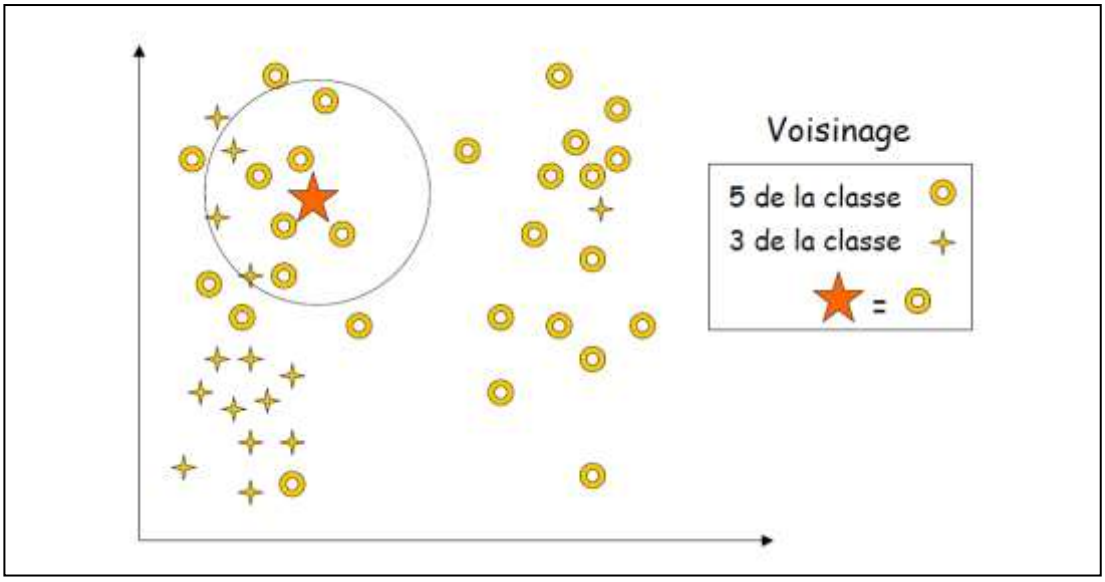
- Déterminer les k plus proches exemples voisins de y
- Combiner les classes de ces k exemples en une classe c

Résultat de sortie: la classe de y est *c(y) = c*

Fin

Remarques

- La solution la plus simple est de rechercher un seul plus proche voisin (k = 1) et affecter sa classe au nouvel élément x, on parle d'un algorithme 1-NN.
- Il est préférable de prendre une valeur impaire de k.
- Pour k > 1, on affecte la classe qui contient le maximum de voisins au nouvel élément (vote majoritaire).



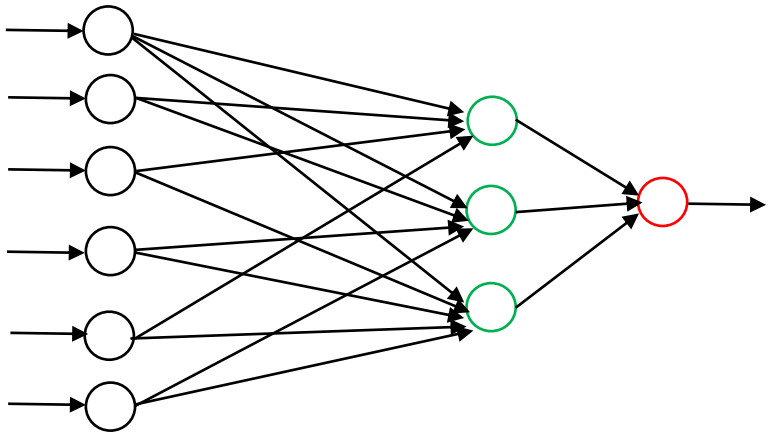
Critique de l'algorithme

- ***Pas d'apprentissage:*** introduction de nouvelles données ne nécessite pas la reconstruction du modèle.
- Clarté des résultats
- Valable pour tout type de données
- Nombre d'attributs : très important
- Temps de classification : couteux
- Stocker le modèle : couteux en mémoire
- Distance et nombre de voisins : dépend de la distance, choisie du nombre k de voisins et du mode de combinaison.

2. Réseaux de neurones artificiels

- Comme l'on a expliqué auparavant, un RNA simule le système nerveux biologique

- Un réseau de neurones est composé de plusieurs neurones interconnectés.
- Un poids est associé à chaque arc.
- A chaque neurone on associe une valeur de sortie.



Caractéristiques des RNA

- Capacité d'apprentissage : apprendre et changer son comportement en fonction de toute nouvelle expérience.
- Permettent de découvrir automatiquement des modèles complexes.
- Plusieurs modèles de réseaux de neurones : PMC (Perceptron Multi-Couches), RBF (Radial Basis Function), Kohonen, ...
- Méthode de classification: Ajuster les poids en utilisant l'erreur
- Erreur= Valeur désirée–Valeur actuelle.

Méthodes d'apprentissage des RNA

- Méthode de rétropropagation du gradient (Back propagation)
- Méthode de Kohonen
- Méthode RBF (Radial Basis Function)
- Réseaux de neurones probabilistes
- ART (Adaptive Resonance Theory)

Quelques directives pour construire un modèle RNA

- Représentation des entrées
- Nombre de nœuds en entrée : correspond à la dimension des données du problème (attributs ou leurs codages).
- Déterminer le nombre de couches, nombre de nœuds dans chaque couche
- Nombre de couches cachées : Ajusté pendant l'apprentissage.
- Nombre de nœuds par couche : le nombre de nœuds par couche est au moins égal à deux et au plus égal au nombre de nœuds en entrée
- Nombre de nœuds en sortie : fonction du nombre de classes associées à l'application.
- Réseau riche ➔ pouvoir d'expression grand (Ex. 4-2-1 est moins puissant que 4-4-1)
- Attention : Choisir une architecture riche mais pas trop – chargé ==>Problème de sur-spécialisation (Overfitting)

Apprentissage d'un RNA

➢ ***Objectif principal :*** obtenir un ensemble de poids qui font que la plupart des instances de l'ensemble d'apprentissage sont correctement classées.

➢ ***Etapes :***

- Poids initiaux sont générés aléatoirement
- Les vecteurs en entrée sont traités en séquentiel par le réseau
- Calcul des valeurs d'activation des nœuds cachés
- Calcul du vecteur de sortie
- Calcul de l'erreur (sortie désirée –sortie actuelle).
- Les poids sont mis à jour en utilisant l'erreur.

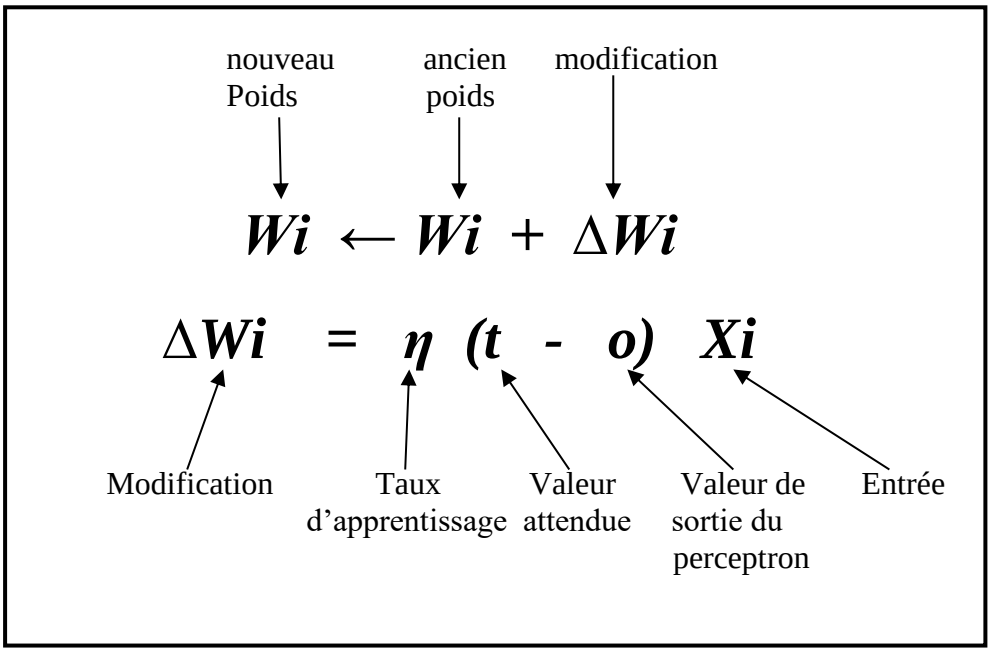
$$\boldsymbol{E(PMC) = \frac{1}{2}\sum_{x \in S}(d(x) - a(x))^2}$$

Avec :

x : un exemple de la base d'apprentissage S

d(x) : sortie désirée a(x) : sortie calculé par le RNA

- Les poids sont mis à jour en utilisant l'erreur.
- L'apprentissage se fait selon la loi de retro-propagation de Hebb vue précédemment :



- Le paramètre taux d'apprentissage $\eta \in [0,1]$ influe sur la modification des poids. (Valeur grande➔ modification forte ; Valeur petite ➔ modification minime)
- Critère d'arrêt : la tolérance définit l'erreur cible.
- et/ou Nombre d'instances bien classées (seuil)

Elagage du réseau

- *N* nœuds en entrée, *h* couches cachées, et *m* nœuds en sortie *h(m+n)* arcs (poids)

⇨ ***Elagage :*** Supprimer les arcs et les nœuds qui n'affectent pas le taux d'erreur du réseau. Eviter le problème de sur-spécialisation (over-fitting).

- Ceci permet de générer des règles concises et claires.

Réseaux de neurones RNA : Avantages

- Taux d'erreur généralement bon
- Outil disponible dans les environnements de data mining
- Résistance au bruit ➔ reconnaissance de formes (son, images sur une rétine, …)
- Classification rapide (réseau étant construit)
- Combinaison avec d'autres méthodes (ex : arbre de décision pour sélection d'attributs)

Réseaux de neurones RNA : Inconvénients

- Apprentissage très long
- Plusieurs paramètres (architecture, coefficients synaptiques, …)
- Pouvoir explicatif faible (boite noire)
- Pas facile d'incorporer les connaissances du domaine.
- Evolutivité dans le temps (phase d'apprentissage)

3. Arbre de décision (Decision Tree) :

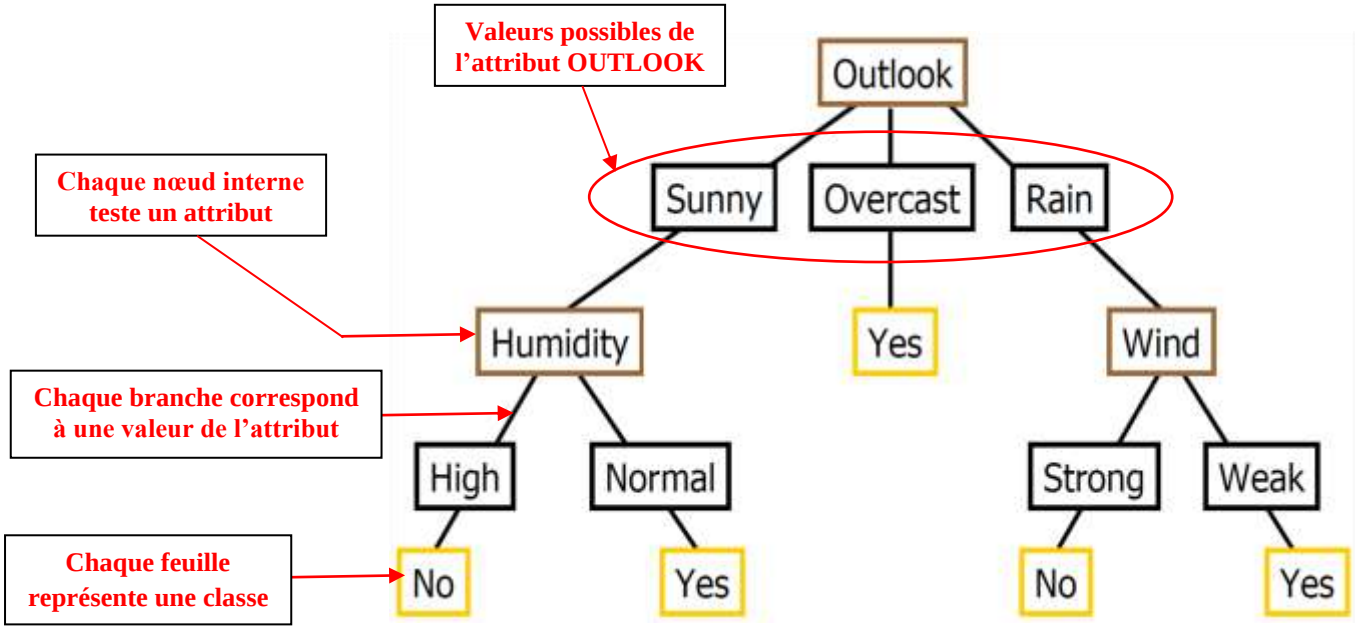
- L'arbre de décision est un modèle (algorithme) classique d'apprentissage. Il est lié à la notion fondamentale de l'informatique (diviser pour régner).
- Les arbres de décision peuvent être appliqués sur plusieurs problèmes d'apprentissage, le plus simple est la classification binaire.
- Arbre = Représentation graphique d'une procédure de classification
- La génération d'arbres de décision se fait à partir des données.

Exemple 01: Construction d'un arbre de décision à partir d'une base d'apprentissage

Base d' apprentissage

Outlook	Temperature	Humidity	Windy	Class Play tennis? (yes/No)
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rain	Mild	High	False	Yes
Rain	Cool	Normal	False	Yes
Rain	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rain	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	HIgh	True	Yes
Overcast	Hot	Normal	False	Yes
Rain	Mild	High	True	No

A partir de la base d'apprentissage précédente on peut construire l'arbre de décision suivante :



Passage de l'arbre de décision aux règles de classification

Exemple :

Si (outlook=sunny) Et (humidity=normal) Alors (Yes : play tennis)

Remarques

- Une règle est générée pour chaque chemin de l'arbre (de la racine à une feuille)
- Les paires (attribut-valeur) d'un chemin forment une conjonction
- Le nœud terminal représente la classe prédite
- Les règles sont généralement plus faciles à comprendre que les arbres

Pour l'arbre de décision précédent, on aura l'ensemble des règles suivantes :

R1: If (Outlook=Sunny) ∧(Humidity=High) Then PlayTennis=No

R2: If (Outlook=Sunny) ∧(Humidity=Normal) Then PlayTennis=Yes

R3: If (Outlook=Overcast) Then PlayTennis=Yes

R4: If (Outlook=Rain) ∧(Wind=Strong) Then PlayTennis=No

R5: If (Outlook=Rain) ∧(Wind=Weak) Then PlayTennis=Yes

Passage des règles de classification à un arbre de décision

R1 : (Ristourne= Oui) ⇒ Non

R2 : (Ristourne= Non)

et (Situation in {Célibat., Divorcé})

et (Impôt< 80K) ⇒Non

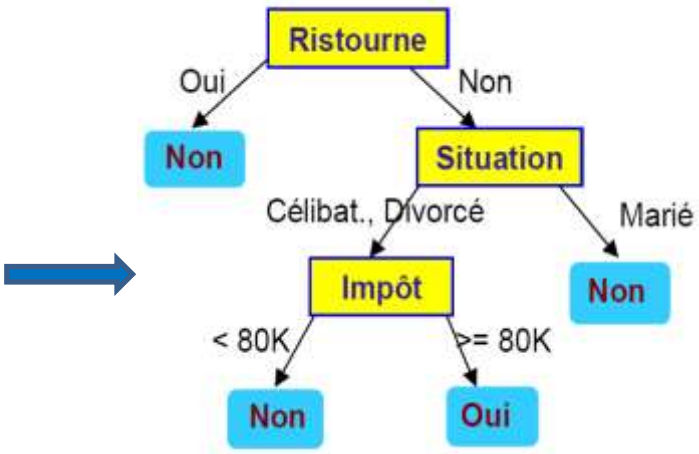
R3 : (Ristourne= Non)

et (Situation in {Célibat., Divorcé})

et (Impôt>= 80K) ⇒Oui

R4 : 'Ristourne= Non)

et (Situation in {Marié}) ⇒Non



Construction de l'arbre de décision

Deux phases peuvent figurer :

Phase 1 : Construction de l'arbre

- Arbre peut atteindre une taille élevée

Phase 2 : Elaguer l'arbre (Pruning)

- Identifier et supprimer les branches qui représentent du "bruit" ➔Améliorer le taux d'erreur

Démarche de construction de l'arbre de décision (Phase 1)

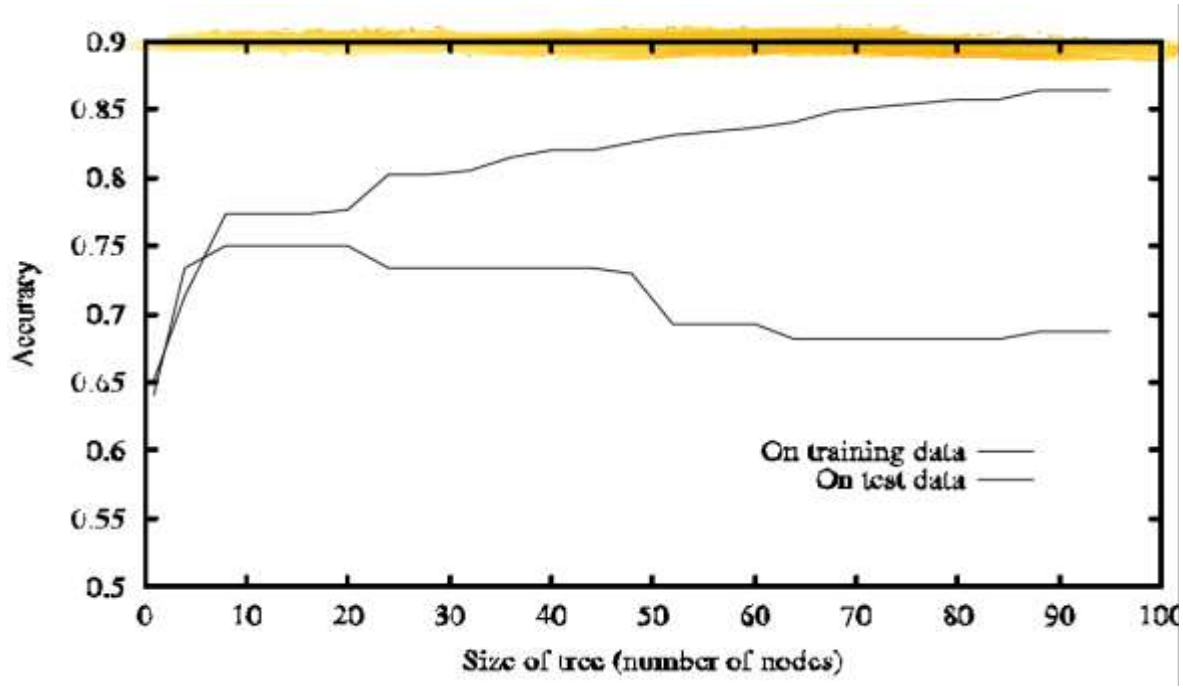
- Au départ, toutes les instances d'apprentissage sont à la racine de l'arbre.
- Sélectionner un attribut et choisir un test de séparation (split) sur l'attribut, qui sépare le "mieux" les instances (ex : Outlook, Humidity, Wind).
- La sélection des attributs est basée sur une heuristique ou une mesure statistique (Gain d'information, Indice Gini, la loi khi-2, …).
- Partitionner les instances entre les nœuds fils suivant la satisfaction des tests logiques
- Traiter chaque nœud fils de façon récursive
- Répéter jusqu'à ce que tous les nœuds soient des terminaux. Un nœud courant est terminal si :
  - Il n'y a plus d'attributs disponibles
  - Le nœud est "pur", i.e. toutes les instances appartiennent à une seule classe,
  - Le nœud est "presque pur", i.e. la majorité des instances appartiennent à une seule classe (Ex : 95%)
  - Nombre minimun d'instances par branche (Ex : algorithme C5 évite la croissance de l'arbre, k=2 par défaut)
- Etiqueter le nœud terminal par la classe majoritaire

Phase 2 : Elagage de l'arbre obtenu (pruning)

- Supprimer les sous-arbres qui n'améliorent pas l'erreur de la classification (accuracy) ➔ avoir un arbre ayant un meilleur pouvoir de généralisation, même si on augmente l'erreur sur l'ensemble d'apprentissage.
- Eviter le problème de sur-spécialisation ou sur-apprentissage (over-fitting), i.e., on a appris "par coeur" l'ensemble d'apprentissage, mais on n'est pas capable de généraliser.

Raisons de la sur sur-spécialisation (sur-apprentissage /Over-fitting)

- Peu de données d'apprentissage
- Bruits et exceptions sur ces données



Comment éviter l'overfitting ?

Deux techniques peuvent être utilisées :

1. ***Pré-élagage :*** Arrêter de façon prématurée la construction de l'arbre
2. ***Post-élagage :***
   - Supprimer des branches de l'arbre complet ("fully grown")
   - Convertir l'arbre en règles ; élaguer les règles de façon indépendante (C4.5)

Construction d'un arbre de décision : Synthèse

- Evaluation des différents branchements pour tous les attributs
- Sélection du "meilleur" branchement" et du meilleur attribut
- Partitionner les données entre les fils
- Construction en largeur (C4.5) ou en profondeur (SPLIT)

Questions critiques :

- Comment formuler des tests de branchement ?
- Comment sélectionner des attributs (mesures de sélection) ?

Quelques algorithmes pour les arbres de décision

- Plusieurs algorithmes de construction des arbres de décision ont été développés, à noter : ID3, C4.5, C5 (CART), CHAID
- Différence principale : mesure de sélection d'un attribut–critère de branchement (split)

Mesures de sélection d'attributs lors de la construction d'un DT

1. Gain d'Information (pour les algorithmes : ID3, C4.5)
2. Indice Gini (CART)
3. Table de contingence statistique χ^2 (CHAID)
4. G-statistic

1.Gain d'information

Principe : Sélectionner l'attribut avec le plus grand gain d'information

- Soient P et N deux classes (Eq : Yes/No) et S un ensemble d'instances avec *p* éléments de P et *n* éléments de N
- L'information nécessaire pour déterminer si une instance prise au hasard fait partie de P ou N est on note I(p, n) ou aussi appelée l'entropie de classification *E* qui mesure l'homogénéité des exemples la formule :

$$\boldsymbol{Entropie\ (S) = Entropie\ (p,n) = I(p,n)}$$

$$\boldsymbol{= -\frac{p}{p+n} Log_2\left(\frac{p}{p+n}\right) - \frac{n}{p+n} Log_2\left(\frac{n}{p+n}\right)}$$

On note le suivant :

- S : l'ensemble des exemples |S| = N = $p+n$
- p : nombre d'exemples positifs (classés Oui)
- n : nombre d'exemples négatifs (classés Non)
- E(s) = 0 ==> tous les exemples appartiennent à la même classe
- E(s) = 1 ==> ∃ autant d'exemples (p) que des exemples (n) (50% pour chaque classe)

- Le gain d'information (Information Gain IG) calcule la réduction attendue sur l'entropie E(s) si un attribut A est utilisé.

⇨ Un algorithme de classification basé sur les DT calcule IG pour chaque attribut *Ai* puis choisit celui qui réduit le plus l'entropie, c.à.d., celui qui permettra le plus nettement possible de séparer les exemples restants.

$$Information\ Gain\ IG\ (S,A)$$
$$= E(S) - Somme\ des\ valeurs\ de\ l'attribut$$
$$= E(s) - \sum \left|S_{V_i}\right| * E(S_{V_i})/|S|$$

Avec : S : ensemble des exemples d'entrée

Ai : l'attribut Utilisé (en cours)

S_v : le sous-ensemble de S dont l'attribut *Ai* a la valeur V

Exemple :

S = {9^+, 5^-} Avec : + (jouer au tennis OUI)

- (ne pas jouer au tennis NON)

➔ Information Nécessaire pour classer un exemple est donnée par l'entropie E

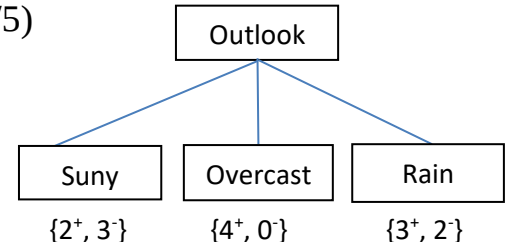
$$E(S) = -(p/N)*Log_2(p/N) - (n/N)*Log_2(n/N) \quad (1)$$

$$= - 9/14* Log_2(9/14)-5/14*Log_2(5/14)$$

$$= \mathbf{0.940}$$

E(sunny) = I(2, 3) = $-2/5*Log_2(2/5) – 3/5*Log_2(3/5)$

= 0.971



E(Overcast) = I(4, 0) = 0

E(Rain) = I(3, 2) = 0.971

➔ Gain (S, Outlook) = $E(s) - E(outlook) = E(s) - \sum|S_{V_i}| * E(S_{V_i})/|S|$

= 0.940 – (|Sv = sunny|*E(Sv = sunny)/|S| + |Sv = Overcast|*E(Sv = Overcast)/|S| +

|Sv = rain|*E(Sv = Rain)/|S| (2)

= 0.940 – (5/14 * 0.971 + 4 /14* 0 + 5 /14* 0.971) = 0.940 – (0.3468 + 0 + 0.3468)

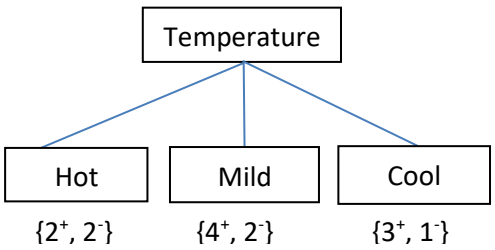
= 0.940 – 0.694 = **0.246** (E(Outlook) = 0.694)

De manière similaire on calcule : (en appliquant (1))

E(Hot) = I(2, 2) = ?

E(Mild) = I(4, 2) = ?

E(Cool) = I(3, 1) = ?

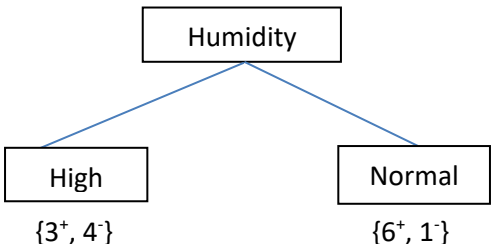


IG (S, Temperature) = **0.029** (en appliquant (2))

E(High) = I(3, 4) = 0.985 (en appliquant (1))

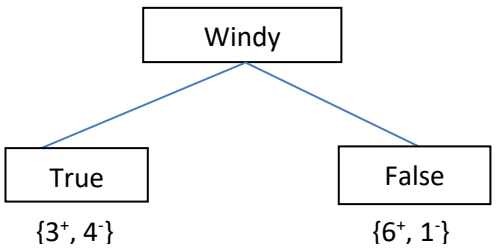
E(Normal) = I(6, 1) = 0.592

IG (S, Humidity) = **0.151** (en appliquant (2))



E(True) = I(3, 4) = 0.811 (en appliquant (1))

E(False) = I(6, 1) = 1.0



IG (S, Windy) = **0.048** (en appliquant (2))

==> le Meilleur attribut pour la sélection dans la première phase est l'attribut « Outlook » (ayant la valeur maximale de IG)

Remarque

- De la même manière on sélectionne d'autre attributs dans les prochaines phases de la construction de l'arbre.
- D'autres variantes de l'algorithme DT (C4.5) utilise d'autre mesures de sélection tels que : Gain Ratio avec :

$$GR(A) = \frac{IG(A)}{E(A)}$$

Exp :

$$GR(Outlook) = \frac{IG(Outlook)}{E(Outlook)} = \frac{0.246}{0.694} \approx 0.354$$

Arbres de décision - Avantages

- Compréhensible pour tout utilisateur (lisibilité du résultat –règles - arbre)
- Justification de la classification d'une instance (racine ➔feuille)
- Valable pour tout type de données
- Robuste au bruit et aux valeurs manquantes.
- Attributs apparaissent dans l'ordre de pertinence ➔tâche de pré-traitement (sélection d'attributs)
- Classification rapide (parcours d'un chemin dans un arbre).
- Outils disponibles dans la plupart des environnements de data mining

Arbres de décision - Inconvénients

- Sensibles au nombre de classes : performances se dégradent
- Evolutivité dans le temps : si les données évoluent dans le temps, il est nécessaire de relance la phase d'apprentissage

4.Méthodes bayésiennes (probabilistes)

Idée : Prédire des hypothèses multiples fixées au préalable, en se basant sur leurs probabilités.

Comment classer ?

- Calculer la probabilité P(C/X) : probabilité que le tuple (instance) (X=<x_1,…,x_k> est dans la classe C).
  Ex: *P(classe=N | outlook=sunny, windy=true,…)*
- Affecter à une instance X la classe C telle que P(C|X) soit maximale

Théorème de Bayes

$$\boldsymbol{P(C|X) = \frac{P(X|C).P(C)}{P(X)}}$$

Avec :

- P(X) : est une constante pour toutes les classes (probabilité d'apparition de l'instance X dans la base)
- P(C) = fréquence relative des instances de la classe C
- ***P(C\|X) est maximal ➔ P(X\|C)·P(C) est maximal***

Problème : *calculer P(X|C) est non faisable !*

Hypothèse Naïve :

$$P(x_1, \ldots, x_k|C) = P(x_1|C)\cdot\ldots\cdot P(x_k|C)$$

Avec :

- P(x_i|C) est estimée comme la fréquence relative des instances possédant la valeur x_i (i-ème attribut) dans la classe C.

Exemple : prenons l'exemple vu précédemment (avec les arbres de décision)

Num	Outlook	Temperature	Humidity	Windy	Class Play tennis? (yes/No)
1	Sunny	Hot	High	False	**No**
2	Sunny	Hot	High	True	**No**
3	Overcast	Hot	High	False	**Yes**
4	Rain	Mild	High	False	**Yes**
5	Rain	Cool	Normal	False	**Yes**
6	Rain	Cool	Normal	True	**No**
7	Overcast	Cool	Normal	True	**Yes**
8	Sunny	Mild	High	False	**No**
9	Sunny	Cool	Normal	False	**Yes**
10	Rain	Mild	Normal	False	**Yes**
11	Sunny	Mild	Normal	True	**Yes**
12	Overcast	Mild	HIgh	True	**Yes**
13	Overcast	Hot	Normal	False	**Yes**
14	Rain	Mild	High	True	**No**

Appliquons l'algorithme de Naïve Bayes comme suit :

- On a deux classes: p(Yes), n(No)
- On 04 attributs: Outlook, Temperature, Humidity, Windy
- On peut calculer les deux probabilités : P(p) = 9/14; P(n) = 5/14
- $\boldsymbol{P(v_i|c_j)}$: probabilité d'apparition de la valeur $\boldsymbol{v_i}$ dans les tuples appartenant à la classe c_i ➔ =

$$\boldsymbol{P(v_i|c_j} = Nb.Occ(vi) \in c_i/Total.Occ(c_i)$$

- Par application de la formule précédente, on obtient le tableau suivant :

Attribut 1 : Outlook {sunny, Overcast, Rain}		**Attribut 2 : Temperature {Hot, Mild, Cool}**		**Attribut 3 : Humidity {High, Normal}**		**Attribut 4 : Windy {True, False}**	
P(class=P)	**P(class=N)**	**P(class=P)**	**P(class=N)**	**P(class=P)**	**P(class=N)**	**P(class=P**	**P(class=N)**
P(sunny|p)=2/9	P(sunny|n)=3/5	P(hot|p)=2/9	P(hot|n)=2/5	P(high|p)=3/9	P(high|n)=4/5	P(true|p)=3/9	P(true|n)=3/5
P(overcast|p)=4/9	P(overcast|n)=0	P(mild|p)=4/9	P(mild|n)=2/5	P(normal|p)=6/9	P(normal|n)=1/5	P(false|p)=6/9	P(false|n)=2/5
P(Rain|p) = 3/9	P(Rain|n) = 2/5	P(cool|p)=3/9	P(cool|n)=1/5				

Soit l'instance X=<x_1, x_2, x_3, x_4> = <Rain, Hot, High, False>

Calculer : P(X\|classe=p) ; P(X\|classe=n) ?

P(X\|classe=p) =P(X|classe=p)·P(classe=p)

= P(<x_1, x_2, x_3, x_4>|classe=p)·P(classe=p)

= P(x_1|p). P(x_2|p). P(x_3|p). P(x_4|p). P(classe=p)

= P(Rain|p). P(Hot|p). P(High|p). P(False|p).P(classe=p)

= 3/9*2/9*3/9*6/9.9/14 = 1/3*2/9*1/3*2/3*9/14=$\frac{1*2*1*2*9}{3*9*3*3*14}$

= $\frac{1*2*1}{3*3*7} = \frac{2}{63} \approx 0.0317$ (1)

P(X\|classe=n) =P(X|classe=n)·P(classe=n)

= P(<x_1, x_2, x_3, x_4>|classe=n)·P(classe=n)

= P(x_1|n). P(x_2|n). P(x_3|n). P(x_4|n). P(classe=n)

= P(Rain|n). P(Hot|n). P(High|n). P(False|n).P(classe=n)

= 2/5*2/5*4/5*2/5.5/14 =$\frac{2*2*4*2*5}{5*5*5*5*14}$

= $\frac{2*2*4}{5*5*5*7} = \frac{16}{875} \approx 0.0183$ (2)

De (1) et (2), l'instance X= <Rain, Hot, High, False> peut être classifiée dans la classe C1 = p car :

P(X|classe=p)·P(classe=p) > P(X|classe=n)·P(classe=n) <==> 0.0317>0.0183