Classification
Arbres de décision

Dr A. DJEFFAL

$2^{ème}$ année Master Systèmes d'Information, Optimisation et Décision

2018-2019

www.abdelhamid-djeffal.net

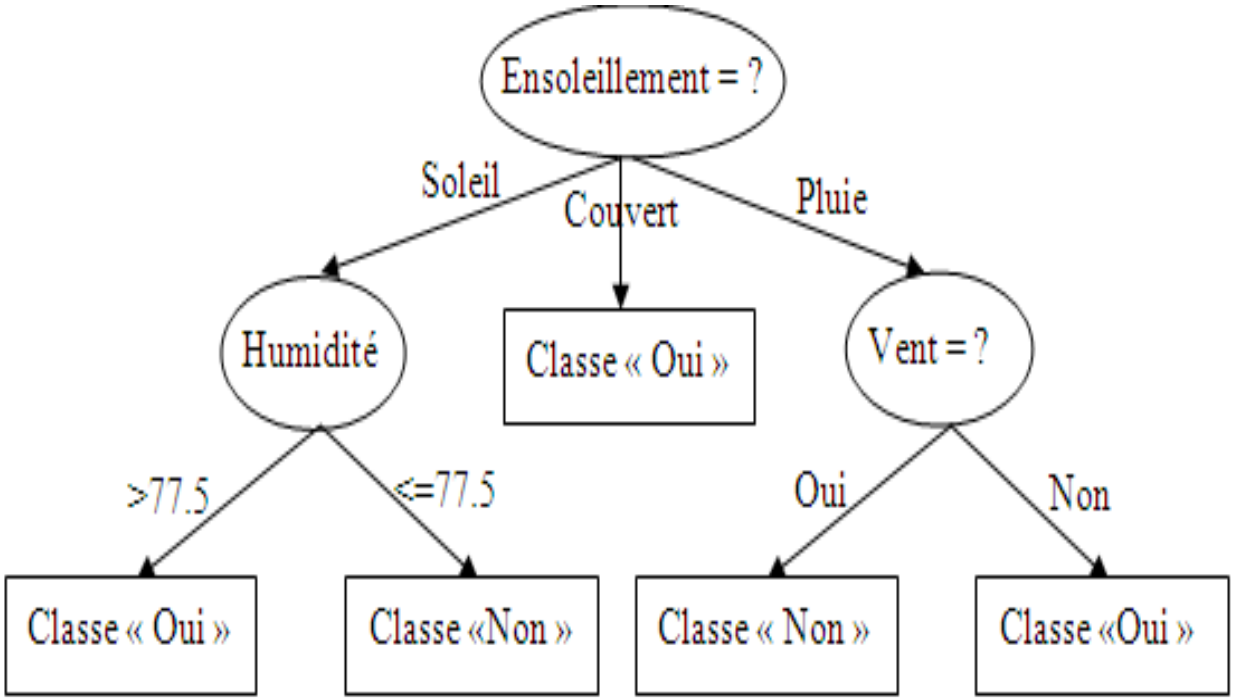
Principe
Définition

- Une méthode très efficace d'apprentissage supervisé.
- Partitionne un ensemble de données en des groupes les plus homogènes possible du point de vue de la variable à prédire.
- On prend en entrée un ensemble de données classées,
- On fournit en sortie un arbre où :
  - chaque nœud final (feuille) représente une décision (une classe)
  - chaque nœud non final (interne) représente un test.
  - Les branches représentent les résultats des tests
- Chaque feuille représente la décision d'appartenance à une classe des données vérifiant tous les tests du chemin menant de la racine à cette feuille.

- L'exemple suivant montre un ensemble de données avec quatre attributs : Ensoleillement, Température, Humidité, Vent et l'attribut à prédire Jouer.

N°	Ensoleillement	Température	Humidité	Vent	Jouer
1	Soleil	75	70	Oui	Oui
2	Soleil	80	90	Oui	Non
3	Soleil	85	85	Non	Non
4	Soleil	72	95	Non	Non
5	Soleil	69	70	Non	Oui
6	Couvert	72	90	Oui	Oui
7	Couvert	83	78	Non	Oui
8	Couvert	64	65	Oui	Oui
9	Couvert	81	75	Non	Oui
10	Pluie	71	80	Oui	Non
11	Pluie	65	70	Oui	Non
12	Pluie	75	80	Non	Oui
13	Pluie	68	80	Non	Oui
14	Pluie	70	96	Non	Oui

- L'arbre appris à partir de cet ensemble de donnée est le suivant :



- En effet, toutes les données ayant l'attribut Ensoleillement="Soleil" et l'attribut Humidité>77.5 appartiennent à la classe 1 ("oui").
- Toute nouvelle donnée peut être classée en testant ses valeurs d'attributs l'un après l'autre en commençant de la racine jusqu'à atteindre une feuille c'est-à-dire une décision.

- Pour construire un tel arbre, plusieurs algorithmes existent : ID3, CART, C4.5,...etc.
- On commence généralement par le choix d'un attribut puis le choix d'un nombre de critères pour son nœud.
- On crée pour chaque critère un nœud concernant les données vérifiant ce critère.
- L'algorithme continue d'une façon récursive jusqu'à obtenir des nœuds concernant les données de chaque même classe.

- L'arbre est construit récursivement de haut en bas selon le principe "diviser pour régner"
- Au début tous les exemples sont dans la racine
- Les attributs sont catégoriels (si continus, il doivent être discrétisés)
- Les exemples sont partitionnés récursivement selon les attributs sélectionnés

- Les attributs sont sélectionnés selon des heuristiques ou des statistiques (gain d'informations) classe.
- Conditions d'arrêt
  - Tous les exemples d'un nœud appartiennent à la même classe
  - Il n y a plus d'attributs pour plus de partitionnement : la majorité est employée pour classer une feuille
  - Il n y a plus d'exemples restants.

Construction

Algorithme CONSTRUIRE-ARBRE(D : ensemble de données)

- Créer nœud N
- **Si** tous les exemples de D sont de la même classe C **alors** Retourner N comme une feuille étiquetée par C ;
- **Si** la liste des attributs est vide **alors** Retourner N Comme une feuille étiquetée de la classe de la majorité dans D ;
- Sélectionner l'attribut A du meilleur Gain dans D ;
- Etiqueter N par l'attribut sélectionné ;
- Liste d'attributs $\leftarrow$ Liste d'attributs - A ;
- **Pour** chaque valeur V_i de A **Faire**
  - Soit D_i l'ensemble d'exemples de D ayant la valeur de $A = V_i$;
  - Attacher à N le sous arbre généré par l'ensemble Di et la liste d'attributs
- **FinPour ;**
- **Fin ;**

En réalité ce n'est pas si simple, plusieurs problèmes doivent être résolus :

- Comment choisir l'attribut qui sépare le mieux l'ensemble de données ? On parle souvent de la variable de segmentation.
- Comment choisir les critères de séparation d'un ensemble selon l'attribut choisi, et comment ces critères varient selon que l'attribut soit numérique ou symbolique ?
- Quel est le nombre optimal du nombre de critères qui minimise la taille de l'arbre et maximise la précision ?
- Quels sont les critères d'arrêt de ce partitionnement, sachant que souvent l'arbre et d'une taille gigantesque ?

Choix d'attribut
Généralité

- Il s'agit de choisir parmi les attributs des données, celui qui les sépare le mieux du point de vue de leurs classes déjà connues.
- Pour choisir le meilleur attribut, on calcule pour chacun une valeur appelée "Gain" qui dépend des différentes valeurs prises par cet attribut.
- Cette mesure est basée sur les recherches en théorie d'informations menées par C.Shannon.

Par exemple :

- Gain d' information (ID3/C4.5)
  - Tous les attributs sont catégoriels
  - Peut être modifié pour les attributs numériques
- Gini index (IBM IntelligentMiner)
  - Tous les attributs sont continus
  - Supposons qu'il ya plusieurs splits possibles pour chaque attribut
  - Peut être modifié pour les valeurs catégoriels.

- Sélectionner l'attribut du gain le plus élevé
- Supposons qu'il y a deux classes P et N
- Soit l'ensemble d'exemples S contenant p exemples de la classe P et n exemples de la classe N
- La quantité d'information nécessaire pour décider qu'un exemple dans S appartienne à P ou N est définie par :

$$H(S) = -\frac{p}{p+n} log_2(\frac{p}{p+n}) - \frac{n}{p+n} log_2(\frac{n}{p+n})$$

Gain d'information
Principe (1)

- Supposons qu'en utilisant l'attribut A un ensemble S sera divisé en $\{S_1, S_2, ..., S_v\}$
- Si S_i contient p_i exemples de P et n_i exemples de N, l'entropie, ou l'information attendus nécessaire pour classifier les objets dans le sous arbre S_i est :

$$H(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} H(S_i)$$

- Le codage d'information qui peut être gagné en se branchant à A est

$$Gain(A) = H(S) - H(A)$$

- Soit un ensemble X d'exemples dont une proportion p_+ sont positifs et une proportion p_- sont négatifs.
- Bien entendu, $p_+ + p_- = 1$
- L'entropie de X est :

$$H(X) = -p_+ log_2(p_+) - p_- log_2(p_-)$$

- Biensur

$$0 \leq H(X) \leq 1$$

- Si $p_+ = 0$ ou $p_- = 0$, alors $H(X) = 0$.
- Ainsi, si tous exemples sont soit tous positifs, soit tous négatifs, l'entropie de la population est nulle.
- Si $p_+ = p_- = 0.5$, alors $H(X) = 1$.
- Ainsi, s'il y a autant de positifs que de négatifs, l'entropie est maximale.

$$Gain(X, a_j) = H(X) - \sum_{v \in valeurs(a_j)} \frac{|X_{a_j=v}|}{|X|} H(X_{a_j=v})$$

- $X_{a_j=v}$, est l'ensemble des exemples dont l'attribut considéré a_j prend la valeur v,
- la notation $|X|$ indique le cardinal de l'ensemble X.

Gain d'information

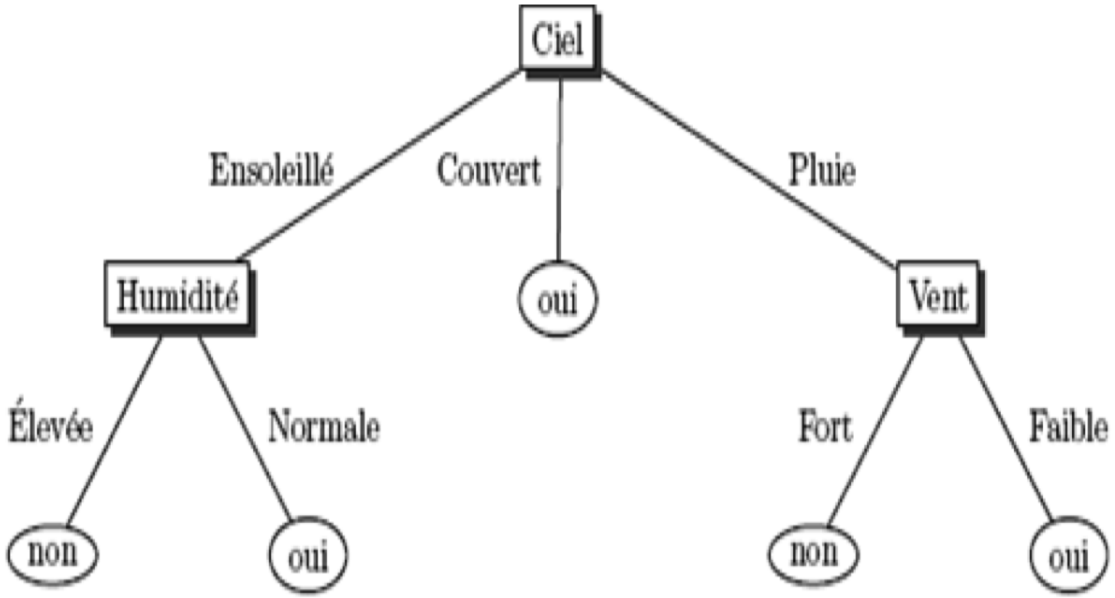
Exemple

Le Gain du champs "Vent" de la table précédente est calculé comme suit :

$$
\begin{aligned}
Gain(X, vent) &= H(X) - \tfrac{6}{14} H(X_{a=oui}) - \tfrac{8}{14} H(X_{a=non}) \\
On\ a : & \\
H(X) &= -\tfrac{5}{14} ln_2 \tfrac{5}{14} - \tfrac{9}{14} ln_2 \tfrac{9}{14} = 0.940 \\
H(X_{a=non}) &= -(\tfrac{6}{8} ln_2 \tfrac{6}{8} + \tfrac{2}{8} ln_2 \tfrac{2}{8}) = 0.811 \\
Et & \\
H(X_{a=oui}) &= -(\tfrac{3}{6} ln_2 \tfrac{3}{6} + \tfrac{3}{6} ln_2 \tfrac{3}{6}) = 1.0 \\
D'où : & \\
Gain(X, vent) &= 0.940 - \tfrac{8}{14} * 0.811 - \tfrac{6}{14} * 1.0 \\
&= 0.048
\end{aligned}
$$

Déterminer l'arbre de décision déduit de la table suivante :

Numéro	Ensoleillement	Température	Humidité	Vent	Jouer ?
1	Ensoleillé	Chaude	Elevée	Faible	Non
2	Ensoleillé	Chaude	Elevée	Fort	Non
3	Couvert	Tiède	Elevée	Faible	Oui
4	Pluie	Fraîche	Elevée	Fort	Oui
5	Pluie	Fraîche	Normale	Faible	Oui
6	Pluie	Fraîche	Normale	Fort	Oui
7	Couvert	Fraîche	Normale	Fort	Oui
8	Ensoleillé	Tiède	Elevée	Faible	Non
9	Ensoleillé	Fraîche	Normale	Faible	Oui
10	Pluie	Tiède	Normale	Faible	Oui
11	Ensoleillé	Tiède	Normale	Fort	Oui
12	Couvert	Fraîche	Elevée	Fort	Oui
13	Couvert	Chaude	Normale	Faible	Oui
14	Pluie	Tiède	Normale	Fort	Non



- Si une base T contient des exemples de n classes, gini index, gini(T) est défini par :

$$Gini(T) = 1\text{-}\sum_{j=1}^{n} p_j^2$$

où p_j est la fréquence de la classe j dans T.

- Si la base T est partitionnée en deux bases T_1 et T_2 de tailles N_1 et N_2 respectivement, le gini index $gini(T)$ du partitionnement est défini par :

$$Gini_{split}(T) = \frac{N_1}{N} gini(T_1) + \frac{N_2}{N} gini(T_2)$$

- L'attribut de $Gini_{split}(T)$ minimum est choisi pour diviser le nœud

Taille de l'AD

Choix de la bonne taille de l'arbre

- AD construit peut être d'une taille très importante épuisant les ressources de calcul et de stockage.
- Solution $\Rightarrow$ élagage :éliminer de l'AD les branches les moins significatives (déduisant d'un min d'exemples ou de appartenant à diff classes).
- Élagage avant ou après l'apprentissage (pré et post-élagage)

Taille de l'AD

Pré-élagage

- Effectué lors de la construction de l'arbre,
- lorsqu'on calcule les caractéristiques statistiques d'une partie des données tel que le gain, on peut décider de l'importance ou non de sa subdivision,
- ainsi on coupe complètement des branches qui peuvent être générée.

Taille de l'AD

Post-élagage

- Effectué après la construction de l'arbre en coupant des sous arbres entiers et en les remplaçant par des feuilles représentant la classe la plus fréquente dans l'ensemble des données de cet arbre.
- On commence de la racine et on descend,
- pour chaque nœud interne (non feuille), on mesure sa complexité avant et après sa coupure (son remplacement par une feuille),
- si la différence est peu importante, on coupe le sous arbre et on le remplace par une feuille.

- **Algorithme ID3**
  - ID3 construit l'arbre de décision récursivement.
  - A chaque étape de la récursion, il calcule parmi les attributs restant pour la branche en cours, celui qui maximisera le gain d'information.
  - Le calcul ce fait à base de l'entropie de Shanon déjà présentée.
  - L'algorithme suppose que tous les attributs sont catégoriels ;
  - Si des attributs sont numériques, ils doivent être descritisés pour pouvoir l'appliquer.

- **Algorithme C4.5 (J48)**
  C'est une amélioration de l'algorithme ID3,
- Prend en compte les attributs numérique ainsi que les valeurs manquantes.
- L'algorithme utilise la fonction du gain d'entropie combiné avec une fonction *SplitInfo* pour évaluer les attributs à chaque itération.
- Attributs discrets : Gain et permet le regroupement,
- Attributs continus : Segmentés par un expert, sinon :
  - trier l'attribut
  - prendre les seuils $a_i + a_{i+1}/2$ (a_i et a_{i+1} deux valeurs consécutives de l'attribut)
  - prendre les compositions de meilleur gain
- Valeurs manquante :
  - pour le test : prendre la classe majoritaire
  - pour l'entrainement prendre la distribution des valeurs connues

- **Algorithme CART**
  - "Classification And Regression Trees",
  - analogue à l'algorithme ID3 mais arbre binaire et l'indice de Gini
  - À un attribut binaire correspond un test binaire.
  - À un attribut qualitatif ayant n modalités, on peut associer autant de tests qu'il y a de partitions en deux classes, soit $2^n - 1$ tests binaires possibles.
  - Enfin, dans le cas d'attributs continus : discrétiser puis revenir au cas qualitatif

Extraction des règles
Principe

- Représenter sous forme de règles IF-THEN
- Une règle est crée pour chaque chemin de la racine vers une feuille
- Chaque paire de valeurs d'attributs forme une conjonction
- Les feuilles représentent les classes prédites
- Les règles sont faciles à comprendre pour les humains
- Exemple
  - IF age = "$\leqslant$30" AND student = "no" THEN buys_computer = "no"
  - IF age = "$\leqslant$" AND student = "yes" THEN buys_computer = "yes"
  - IF age = "31..40" THEN buys_computer = "yes"
  - IF age = ">40" AND credit_rating = "excellent" THEN buys_computer = "yes"
  - IF age = ">40" AND credit_rating = "fair" THEN buys_computer = "no"

Avantages
Avantages

- Une bonne vitesse d'entrainement par rapport à d'autre méthodes
- Convertible à de simples et compréhensibles règles
- Possibilité d'utilisation des requêtes SQL pour accéder aux BDDs
- Une précision comparable à d'autres méthodes