APPRENTISSAGE AVEC ARBRE DE DÉCISION

Cours Master1 IAM

Introduction

- L’une des grandes familles d’approches pour la résolution de problèmes comme pour l’apprentissage est la technique consistant à « diviser pour régner » (divide and conquer ». Elle se résume à identifier des sous-problèmes, à leur trouver une solution, puis à combiner ces solutions pour résoudre le problème général.
- C’est sur ce principe que sont fondés les algorithmes d’apprentissage par arbres de décision.
- Certaines applications spécifiques comprennent le diagnostic médical, l'évaluation du risque de crédit des demandes de prêt, la classification des maladies, a classification de recherche Web, la reconnaissance de forme, etc.

Problème de classification

Chaque élément xi de la base d'apprentissage est représenté par un vecteur multidimensionnel (x1,x2,...xn) correspondant à l'ensemble de variables descriptives du point. Chaque nœud interne de l'arbre correspond à un test fait sur une des variables xi :

- Variable catégorielle : génère une branche (descendant) par valeur de l'attribut
- Variable numérique : test par intervalles (tranches) de valeurs

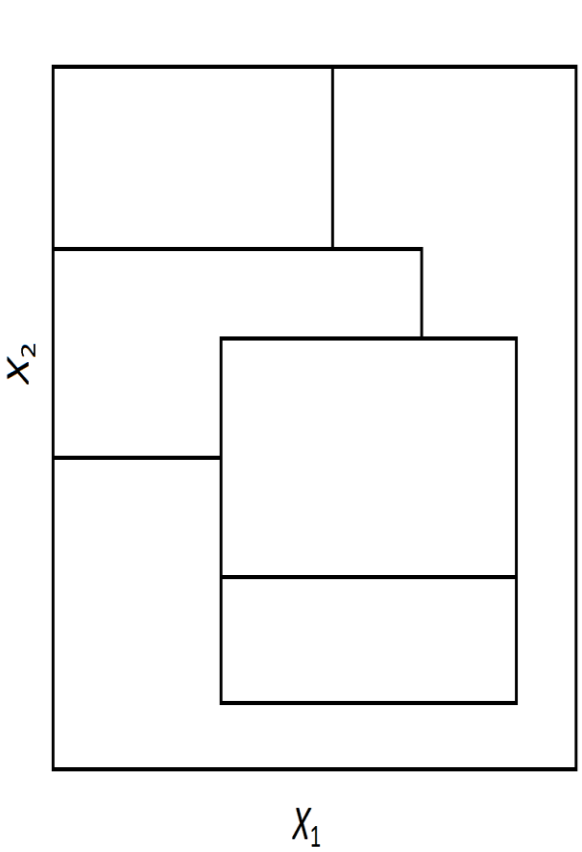
Les *feuilles* de l'arbre spécifient les classes.

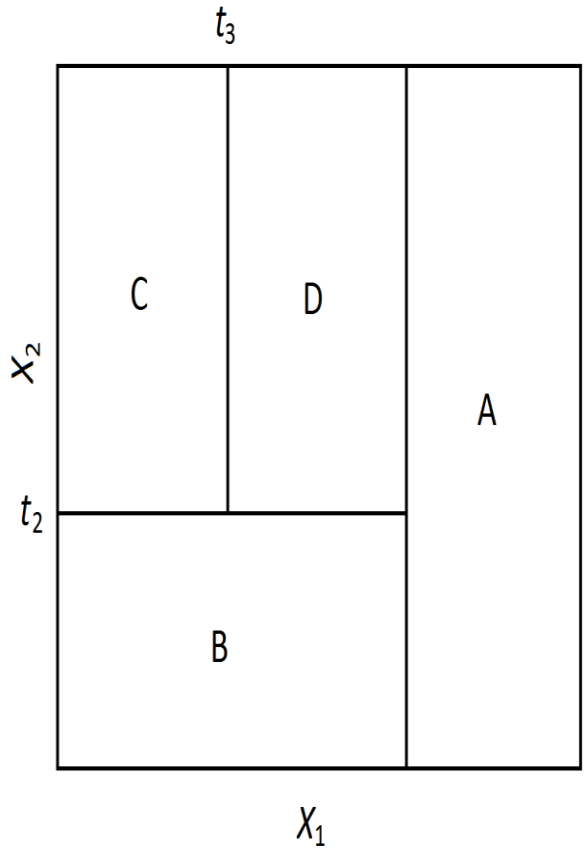
Une fois l'arbre construit, classifier un nouvel candidat se fait par une descente dans l'arbre, de la racine vers une des feuilles (qui encode la décision ou la classe). A chaque niveau de la descente on passe un nœud intermédiaire où une variable xi est testée pour décider du chemin (ou sous arbre) à choisir pour continuer la descente.

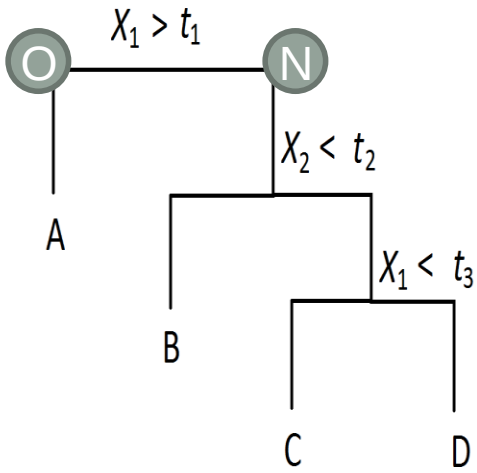
Principe de construction

- Initialement, les points des la base d'apprentissage sont tous placés dans le nœud racine. Une des variables de description des points est la classe est dite « variable cible ».
- La variable cible peut être catégorielle (problème de classification) ou valeur réelle (problème de régression).
- Chaque nœud est coupé (opération *split*) donnant naissance à plusieurs nœuds descendants. Un élément de la base d'apprentissage situé dans un nœud se retrouvera dans un seul de ses descendants.
- L'arbre est construit par partition récursive de chaque nœud en fonction de la valeur de l'attribut testé à chaque itération (*top-down induction*).
- Le critère d'optimisation est la homogénéité des descendants par rapport à la variable cible. La variable qui est testé dans un nœud sera celle qui maximise cette homogénéité.
- Le processus s'arrête quand les éléments d'un nœud ont la même valeur pour la variable cible (*homogénéité*).

Représentation de la base d'exemples



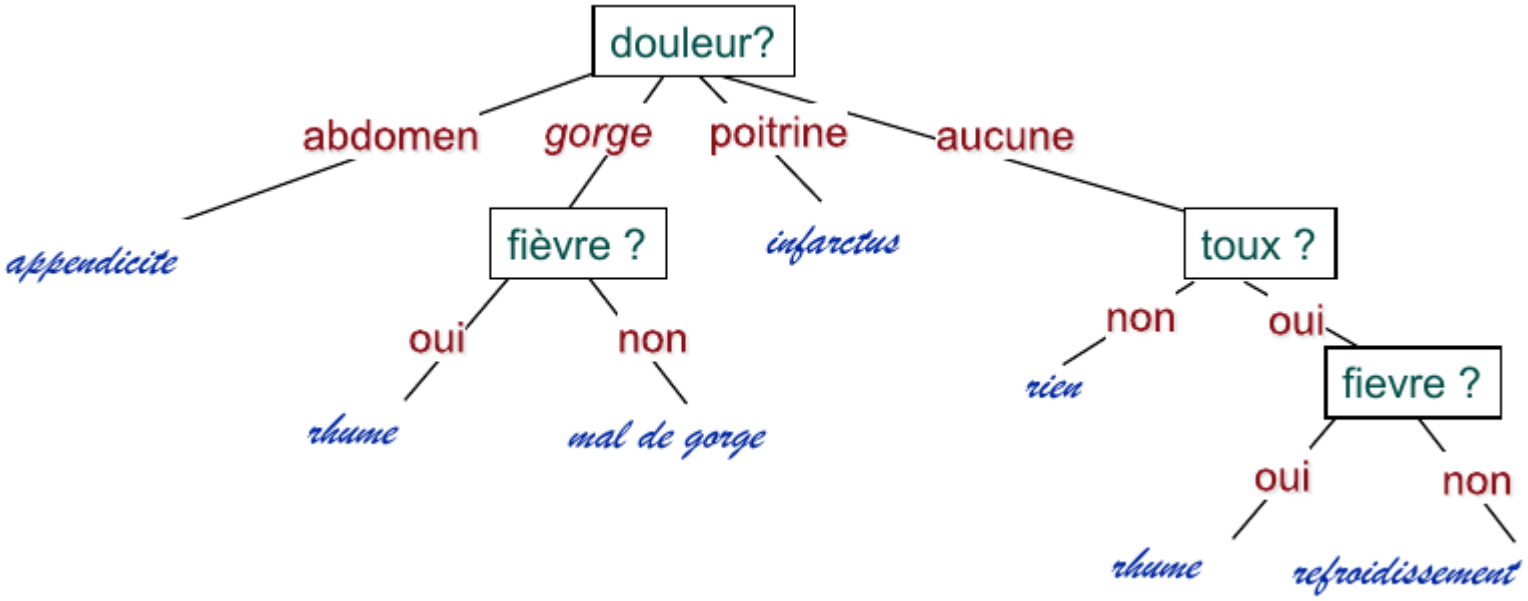




Séparation de classes par partition itérative des variables. A chaque étape le but est de couper le nœud en deux régions les plus homogènes possible

Exemple Illustratif

	Toux	Fièvre	Poids	Douleur
Marie	non	oui	normal	gorge
Fred	non	oui	normal	abdomen
Julie	oui	oui	maigre	aucune
Elvis	oui	non	obese	poitrine

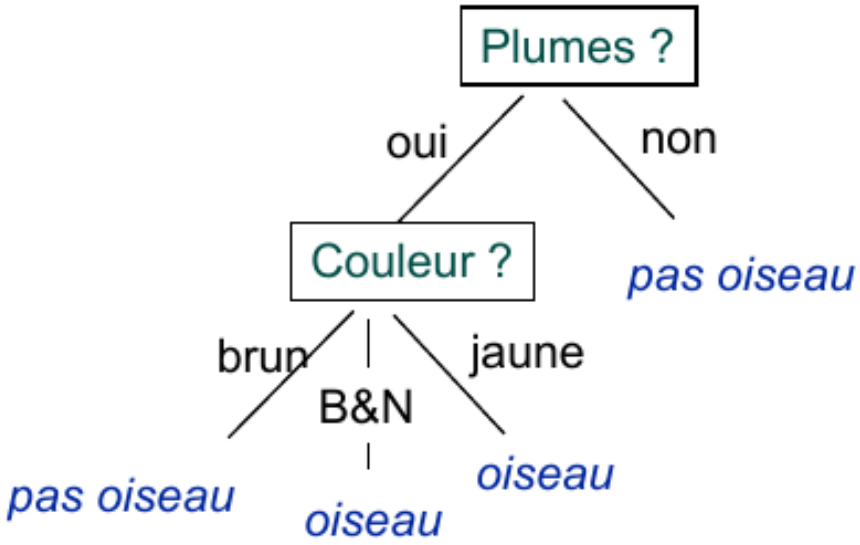


Un ensemble d'individus décrits par plusieurs variables catégorielles (Toux, Fievre, Poids, Douleur). Dans l'arbre de décision obtenu, la variable cible se retrouve dans les feuilles.

Représentation par des règles logiques

	Couleur	Ailes	Plumes	Sonar	**Concept**
Faucon	jaune	oui	oui	non	*oiseau*
Pigeon	B&N	oui	oui	non	*oiseau*
chauve-souris	brun	oui	non	oui	*pas oiseau*

Pour un ensemble d'éléments décrits par des variables catégorielles la classification résultante peut être vue comme une combinaison des règles booléennes.



(Si Plumes = « non » Alors Classe= « pas-oiseau)
ou (Si Plumes = oui & Couleur= brun Alors Classe= pas-oiseau)
ou (Si Plumes = oui & Couleu r= B&N Alors Classe= oiseau)
ou (Si Plumes = oui & Couleu r= jaune Alors Classe= oiseau)

Implémentation : ID3 (Iterative Dichotomiser 3)

L'algorithme ID3 introduit par Quinlan dans son article *Induction of Decision Trees* (Machine Learning Journal, Mars 1986)

1. Placement de tous les exemples d'apprentissage dans le nœud racine,
2. Chaque nœud est coupé sur un des attributs restants (qui n'a pas été encore testé),
3. Le choix de cet attribut se fait sur une mesure d'homogénéité (mesure d'entropie) par rapport à la variable cible. Cette mesure est le gain d'information obtenu par le découpage.

On suppose que la variable cible a *n* valeurs distinctes (les classes).

Pour un nœud *S* (interne ou feuille), on calcule son *entropie* par rapport à la cible,

Calcul de l'entropie

L'entropie de Shannon définit la quantité d'informations apportée par un évènement : plus la probabilité de l'évènement est faible(rare), plus la quantité d'informations qu'il apporte est grande,

1. Partitionner S sur les valeurs de la cible en n groupes: C1,...,Cm
2. Calculer *pi*, i=1...n : probabilité qu'un élément de S se retrouve dans Ci (pi ≈ |Ci|/|S|) où |Ci| est la taille du groupe Ci,
3. $\boldsymbol{Entropie\ de\ S : H(S) = -\sum_{i=1}^{m} p_i\, log_2 p_i}$

Pour calculer le gain d'information dans un nœud interne S sur l'attribut « A »:

1. Partitionner S sur les valeurs de l'attribut « A » en k sous-groupes: S1,...,Sk
2. pi : la probabilité qu'un élément de S appartient à Si (pi≈|Si|/|S|)
3. Calculer le ***gain d'information*** sur l'attribut « A » :

$$\boldsymbol{G_i(S,A) = H(S) - H(S|A),} \quad \boldsymbol{où}$$

$$\boldsymbol{H(S|A) = \sum_{v\ valeurs\ de\ A} p(A=v).H(S|A=v)}$$

Exécution de l'algorithme ID3

L'**algorithme ID3** commence par la racine. Ensuite pour le nœud S en train d'être testé :

1. Calculer le gain d'information pour chaque attribut pas encore utilisé,
2. Choisir l'attribut « A » de gain d'information maximal,
3. Créer un test (décision) sur cet attribut dans le nœud S et générer les sous-noeuds S_1,...S_k correspondant à la partition sur l'attribut choisi « A ».
4. Récurrence sur les nœuds qui viennent d'être crées.

Sortie de la récursivité :

- Tous les éléments de S sont dans la même classe (H(S)=0): S devient nœud feuille
- Pas d'attributs non utilisés : nœud feuille sur la classe majoritaire
- S=∅ : nœud feuille sur la classe majoritaire du parent (ce cas est nécessaire pour la classification de nouveaux échantillons)

Exemple: Classification d'un ensemble de jours (J1, …, J14) en deux catégories : « Jouer au golf », et « Ne pas jouer »

Exemple *[Quinlan,86]*

Attributs	**Pif**	**Temp**	**Humid**	**Vent**
Valeurs possibles	soleil,couvert,pluie	chaud,bon,frais	normale,haute	vrai,faux

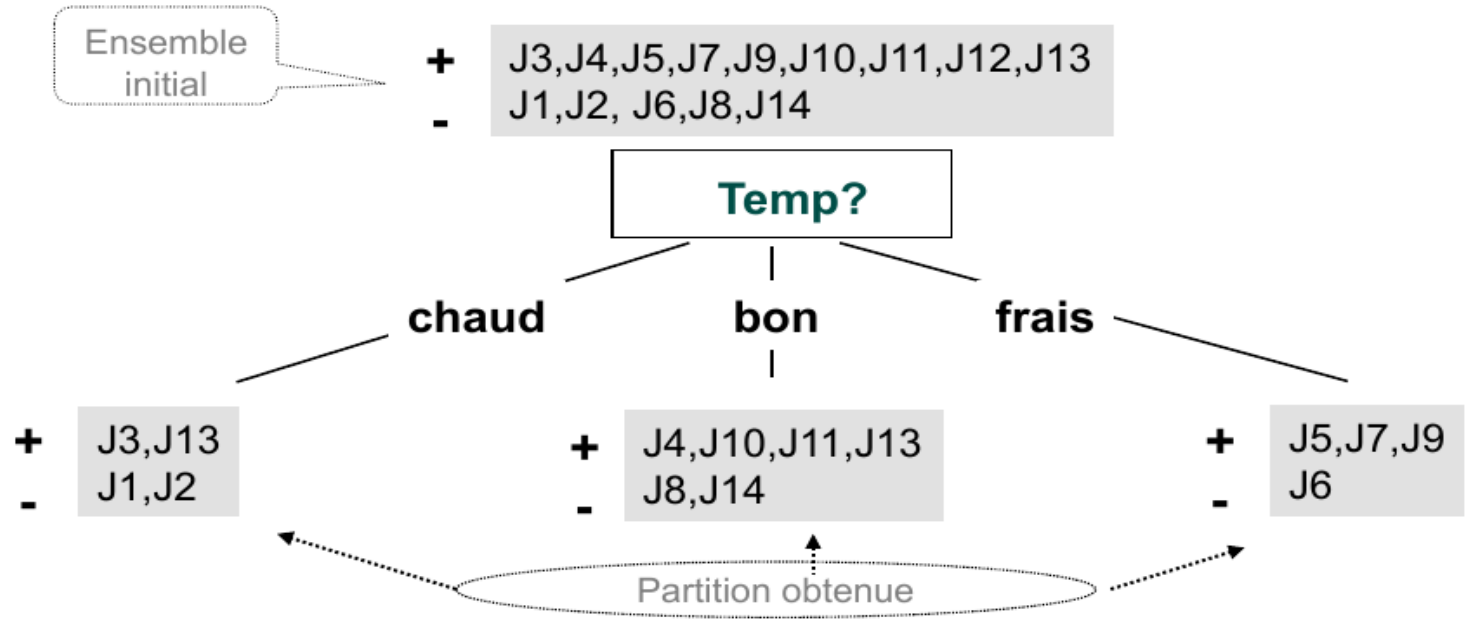
N°	Pif	Temp	Humid	Vent	Golf
1	soleil	chaud	haute	faux	NePasJouer
2	soleil	chaud	haute	vrai	NePasJouer
3	couvert	chaud	haute	faux	Jouer
4	pluie	bon	haute	faux	Jouer
5	pluie	frais	normale	faux	Jouer
6	pluie	frais	normale	vrai	NePasJouer
7	couvert	frais	normale	vrai	Jouer
8	soleil	bon	haute	faux	NePasJouer
9	soleil	frais	normale	faux	Jouer
10	pluie	bon	normale	faux	Jouer
11	soleil	bon	normale	vrai	Jouer
12	couvert	bon	haute	vrai	Jouer
13	couvert	chaud	normale	faux	Jouer
14	pluie	bon	haute	vrai	NePasJouer

la classe

Développement de l'arbre de décision: exemple

Si on choisit l'attribut « Temp »
pour le nœud racine

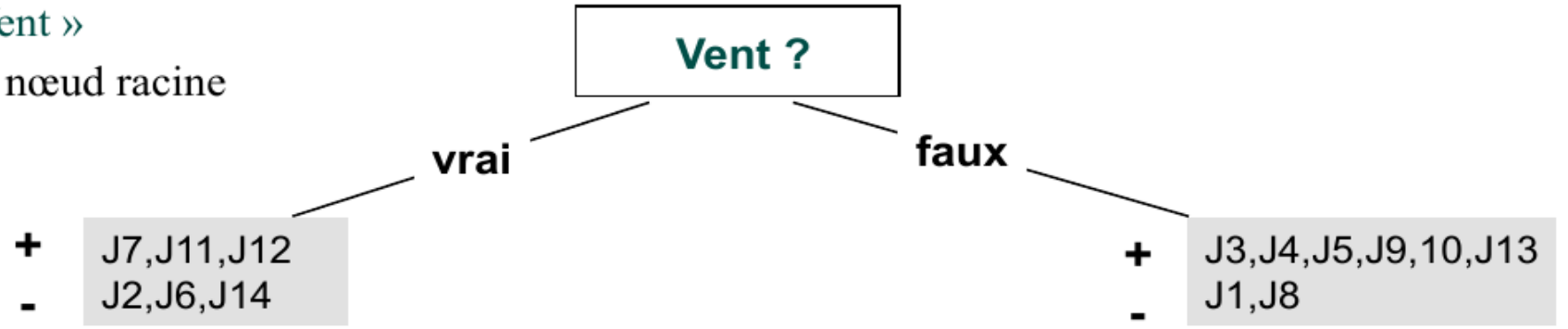
N°	Pif	Temp	Humid	Vent	Golf
1	soleil	chaud	haute	faux	NePasJoue
2	soleil	chaud	haute	vrai	NePasJoue
3	couvert	chaud	haute	faux	Jouer
4	pluie	bon	haute	faux	Jouer
5	pluie	frais	normale	faux	Jouer
6	pluie	frais	normale	vrai	NePasJoue
7	couvert	frais	normale	vrai	Jouer
8	soleil	bon	haute	faux	NePasJoue
9	soleil	frais	normale	faux	Jouer
10	pluie	bon	normale	faux	Jouer
11	soleil	bon	normale	vrai	Jouer
12	couvert	bon	haute	vrai	Jouer
13	couvert	chaud	normale	faux	Jouer
14	pluie	bon	haute	vrai	NePasJoue



Sélection de l'attribut

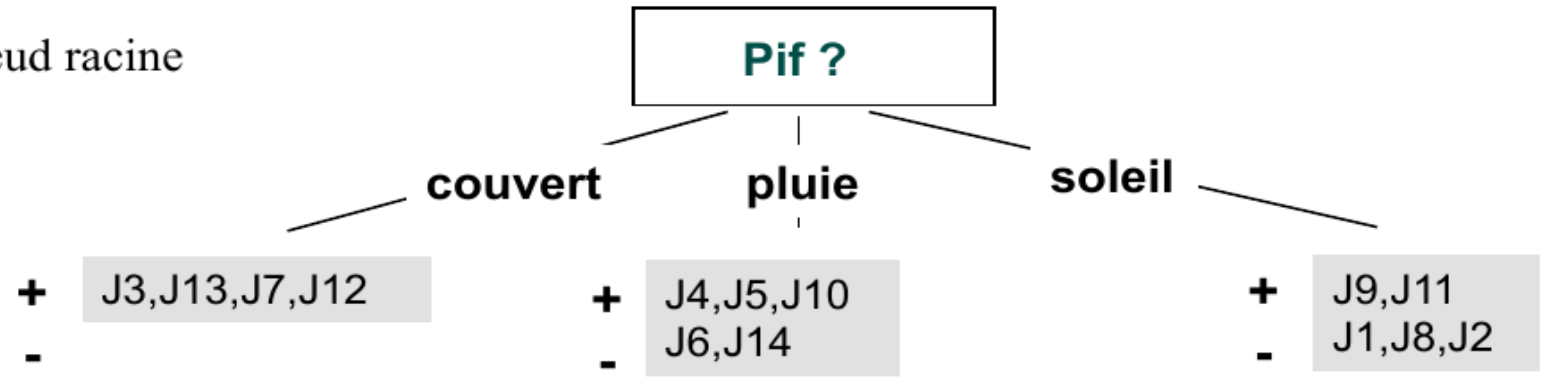
Si on choisit l'attribut « Vent » pour le nœud racine

\+ J3,J4,J5,J7,J9,J10,J11,J12,J13
\- J1,J2, J6,J8,J14

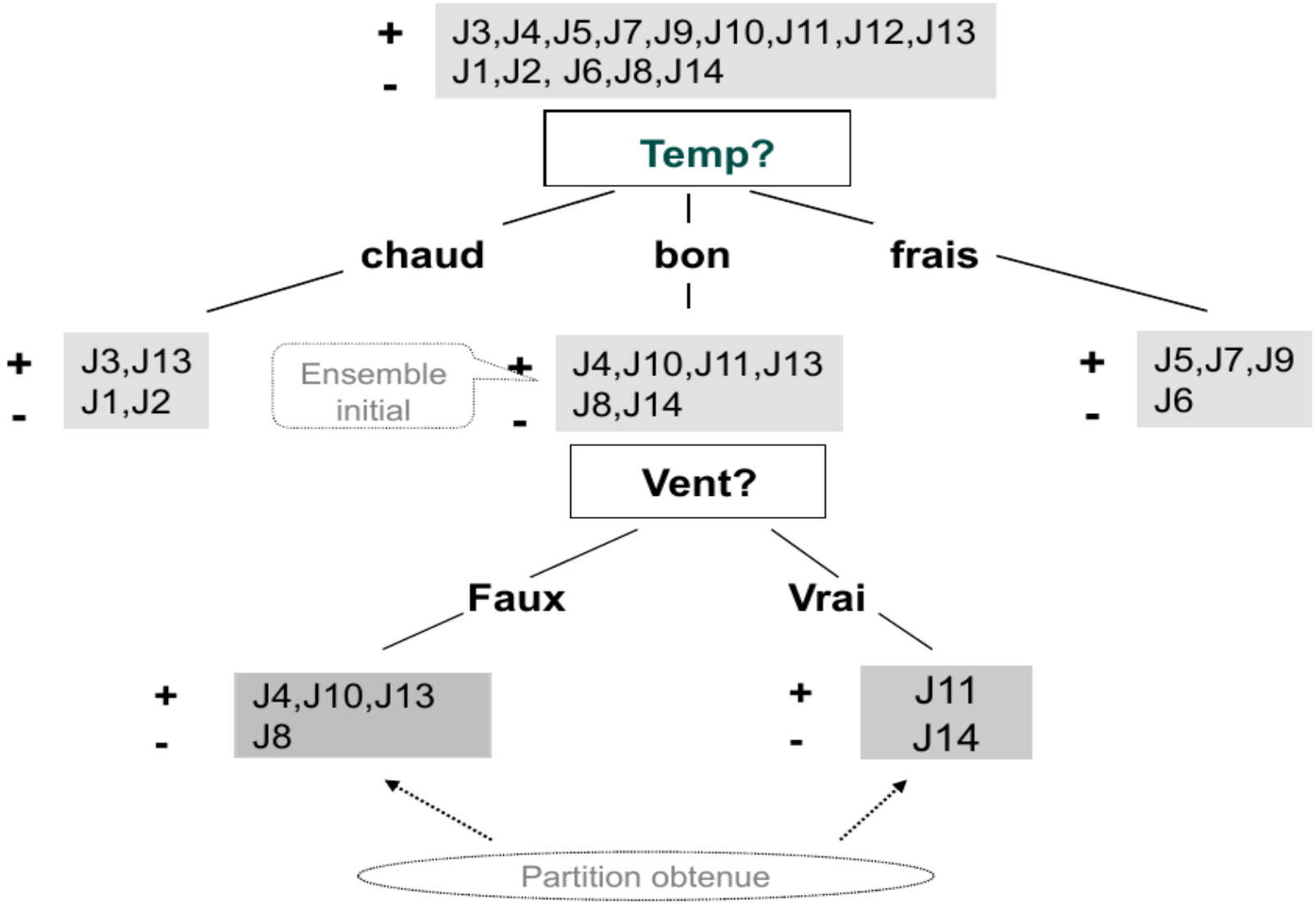


Si on choisit l'attribut « Pif » pour le nœud racine

\+ J3,J4,J5,J7,J9,J10,J11,J12,J13
\- J1,J2, J6,J8,J14



Développement de l'arbre



Exemple

- Calcul de l'entropie de l'échantillon initial:

$H(S) =$

$-\ p(classee = jouer) log_2 p(classe = jouer) -$

$p(classe = Nonjouer).\ log_2 p(classe = Nonjouer) = \frac{9}{14}.\ log_2\left(\frac{9}{14}\right) - \frac{5}{14}.\ log_2\left(\frac{5}{14}\right) =$

- Calcul du gain d'information pour chaque attribut A:

1. Gain(S, Pif) = $H(S) - H(S|Pif)$

$H(S|Pif) =$

$p(Pif = soleil).\ [-p(classe = j|Pif = soleil).\ log_2(classe = j|Pif = soleil) -$

$p(classe = nj|Pif = soleil).\ log_2(classe = nj|Pif = soleil)] +$

$p(Pif = couvert).\ [-p(classe = j|Pif = couvert).\ log_2(classe = j|Pif = couvert) -$

$p(classe = nj|Pif = couvert).$

$log_2\ (classe = nj|Pif = couvert)]\ +$

$p(Pif = pluie).\ [-p(classe = j|Pif = pluie).\ log_2(classe = j|Pif = pluie) -$

$p(classe = nj|Pif = pluie).$

$log_2(classe = nj|Pif = pluie)]\ =$

$\frac{5}{14}.\left[-\frac{2}{5}.\ log_2\left(\frac{2}{5}\right) - \frac{3}{5}, log_2\left(\frac{3}{5}\right)\right] + \frac{4}{14}.\left[-\frac{4}{4}.\ log_2\left(\frac{4}{4}\right) - \frac{0}{4}.\ log_2\left(\frac{0}{4}\right)\right] +$

$\frac{5}{14}.\left[-\frac{3}{5}.\ log_2\left(\frac{3}{5}\right) - \frac{2}{5}.\ log_2\left(\frac{2}{5}\right)\right]$, D'où Gain(S, Pif) = 0,940 – 0,694 = 0,246

Exemple suite

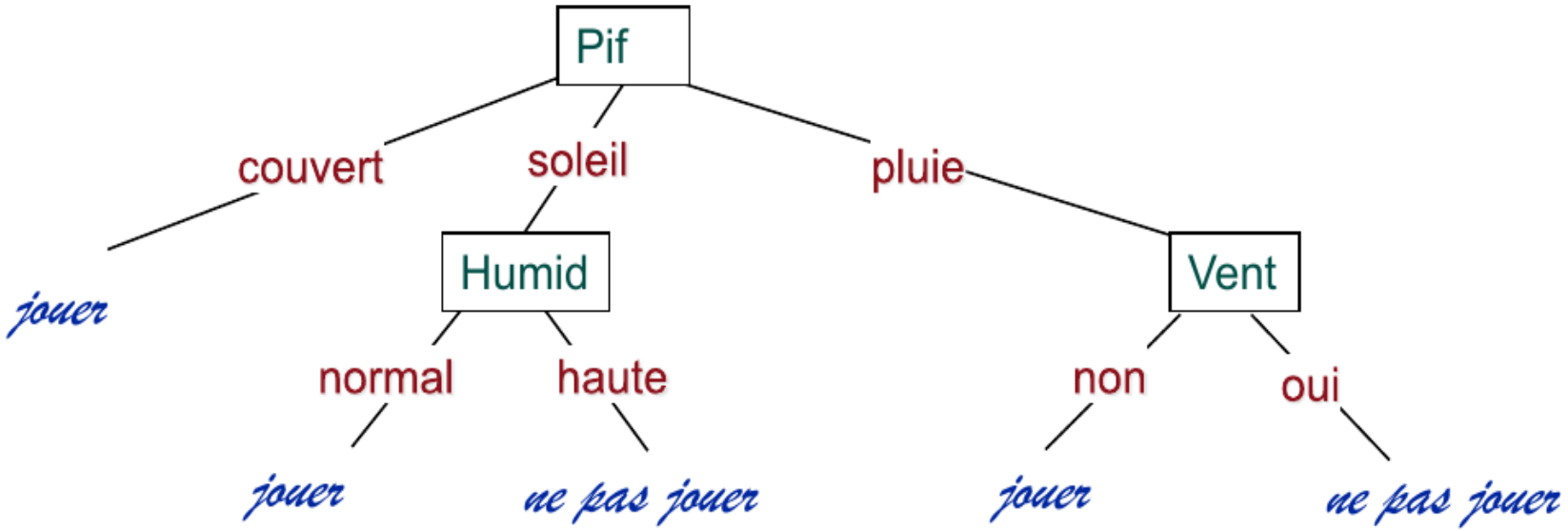
On opère le même calcul pour les attributs restants, on trouve:

- Gain(S, Temp) = 0,246
- Gain(S, Humid) = 0,151
- Gain(S, vent) = 0,048

L'attribut choisi pour le premier test (racine de l'arbre) est celui qui possède la valeur maximale de gain, soit l'attribut « Pif ».

On complète l'arbre en procédant de la même façon mais cette fois-ci on n'utilise pas l'échantillon initiale mais plutôt un sous–échantillon décrit par la valeur de l'attribut correspondant au sous-arbre à déterminer.

Arbre final


Pif
couvert
soleil
pluie
jouer
Humid
Vent
normal
haute
non
oui
jouer
ne pas jouer
jouer
ne pas jouer

Problèmes de l'algorithme ID3

- Solution globale non garantie (la solution trouvée est un optimum local; une amélioration possible : backtracking pour tenter d'obtenir une meilleure solution)
- Sur-apprentissage (overfitting) : pour éviter il faut préférer les arbres de taille réduite (problème d'élagage de l'arbre)
- Pas adapté pour des données numériques continues (ces données doivent être quantifiées avant d'être utilisées avec cet algorithme, mais le problème qui se pose est : « comment faire cette quantification de façon optimale ? »)
- Pour essayer de pallier ces manques deux extensions on été proposées, l'algorithme C4.5 et l'algorithme C5.0

Extensions ID3

C4.5 (Iterative Dichotomiser 4.5)

- Le critère de division est le gain d'information normalisé maximal (différence d'entropie avant et après la division)
- Traitement de variables continues en cherchant des seuils qui maximise le gain d'information
- Traitement de valeurs manquantes
- Étape d'élagage après la création pour remplacer des branches inutiles par des feuilles

C5.0 (Iterative Dichotomiser 5.0)

- Vitesse et utilisation mémoire
- Optimisé pour des bases de données de très grande taille
- Arbres plus petits
- Pondération des cas et erreurs de classification

Critères d'arrêt et élagage

- Règles « évidentes » :
  - tous les exemples du nœud sont de même classe
  - tous exemples du nœud ont mêmes valeurs de variables
  - l'hétérogénéité des nœuds ne diminue plus
  - nb d'exemples dans le nœud < seuil minimal
- Contrôle des performances de généralisation (sur base de validation indépendante)
- Elagage a posteriori : supprimer des branches peu représentatives et nuisant à la généralisation (parcours bottom up en « remontant » d'un niveau tant que cela diminue erreur en généralisation)

Elagage

Soit l'arbre T, et v un de ses nœuds, et :

- $M_C(T,v)$ = nb d'exemples Mal Classés par v dans T
- $M_{Cela}(T,v)$ = nb d'exemples Mal Classés par v dans l'arbre T élagué à v
- n(T) = nb de feuilles de T
- nt(T,v) = nb de feuilles du sous-arbre de T sous nœud v

ALORS on prend comme critère à minimiser :

$$w(T,v) = (MCela(T,v)-MC(T,v))/(n(T)*(nt(T,v)-1))$$

Algorithme d'élagage

Elaguer (Tmax):

K←0; T_k←Tmax

TANT QUE T_k a plus d'un nœud FAIRE

  POUR chaque nœud v de T_k FAIRE

    calculer w(T_k, v) sur exemples appris ou validés.

  FIN_POUR

  choisir v_m = tel que w(T_k, v) soit minimum

  T_{k+1}← T_k où v_m a été remplacé par une feuille

  k←k+1

FTQUE

Pour finir, on choisit parmi {Tmax, T1, … Tn} l'arbre qui a la plus petite erreur de classification sur l'ensemble de validation

Résumé

- ID3 (Inductive Decision Tree, Quinlan 1979) :
  - – arbres « de discrimination » (ie variables uniquement qualitatives)
  - – critère d'hétérogénéité = entropie
- C4.5 (Quinlan 1993) :
  - – Amélioration ID3, permettant notamment arbres « de régression » (gestion des variables continues) et valeurs manquantes
- CART (Classification And Regression Tree, Breiman et al. 1984) :
  - – critère d'hétérogénéité = Gini