Arbres de décision

Cours d'analyse de données
Université Paris I

Motivation

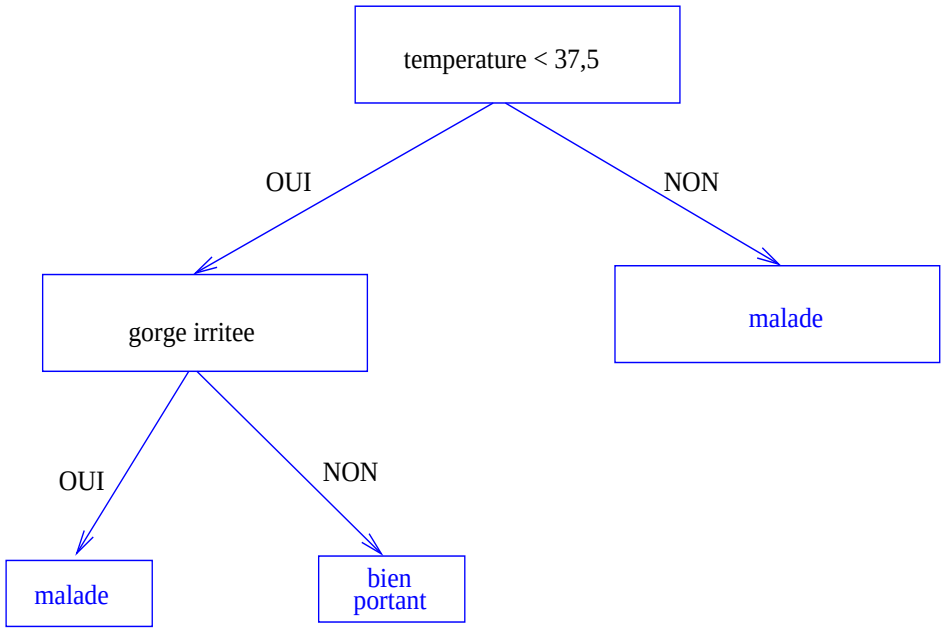
- Pour certains domaines d'application, il est essentiel de produire des classifications compréhensibles pour l'utilisateur
- Dans les méthodes classiques (hiérarchique, *k*-means, Kohonen, perceptron multi-couches), l'information est perdue dans les classes

Exemple

Décider si un patient est malade ou bien portant selon sa température et s'il a la gorge irritée

Arbre de décision

- 2 classes (malade, bien portant)
- 2 variables (température, gorge irritée)

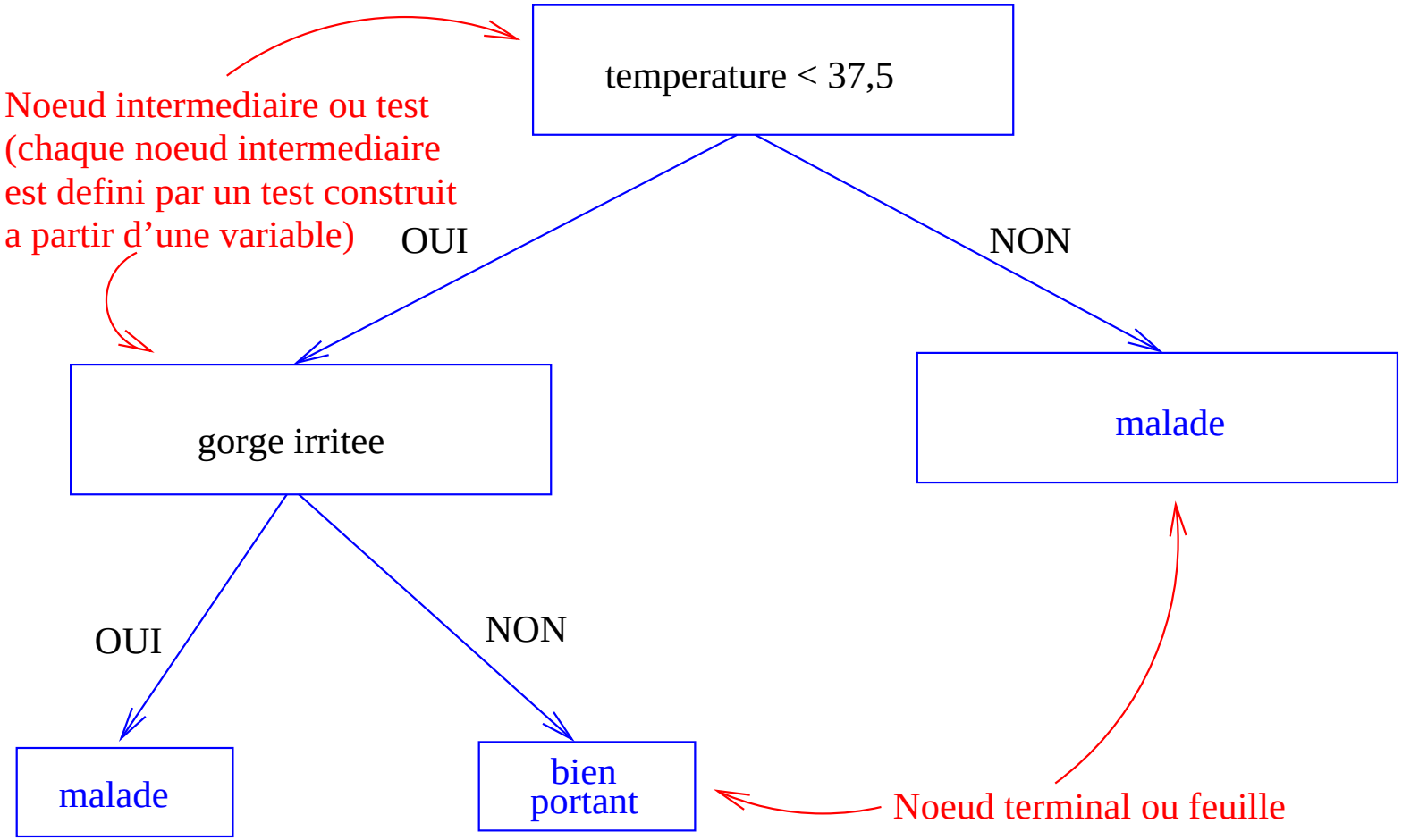


Généralités

Arbre de décision :

- Algorithme de classification supervisée
- Méthode statistique non-paramétrique
- Permet de classer un ensemble d'individus décrits par des variables *qualitatives* et *quantitatives*
- Produit des classes les plus homogènes possibles

Un peu de vocabulaire



Données et notations

Algorithme CART $\longleftrightarrow$ Algorithme d'apprentissage

- Entrées :
  - n individus
  - p variables continues ou discrètes
  - une variable supplémentaire contenant la classe de chaque individu (c classes)
- Sortie :
  - l'arbre de décision T

Données et notations

$N(p)$	$=$	nombre d'individus associés à la position (noeud) p
$N(k\|p)$	$=$	nombre d'individus appartenant à la classe k en sachant qu'ils sont associés à la position p
$P(k\|p) = \frac{N(k\|p)}{N(p)}$	$=$	proportion des individus appartenant à la classe k parmi ceux de la position p

Remarque :

Un noeud est *pur* si tous les individus associés appartiennent à la même classe !

Construction de l'algorithme - exemple

But $\rightarrow$ construire un arbre de décision qui classe et détermine les caractéristiques des clients qui consultent leurs comptes sur internet

Variables

- M : moyenne des montants sur le compte
- A : âge du client
- R : lieu de résidence du client
- E : le client à des études supérieures ?
- I : le client consulte ses comptes sur internet ?

Construction de l'algorithme - exemple

Client	M	A	R	E	I
1	moyen	moyen	village	oui	oui
2	élevé	moyen	bourg	non	non
3	faible	âgé	bourg	non	non
4	faible	moyen	bourg	oui	oui
5	moyen	jeune	ville	oui	oui
6	élevé	âgé	ville	oui	non
7	moyen	âgé	ville	oui	non
8	faible	moyen	village	non	non

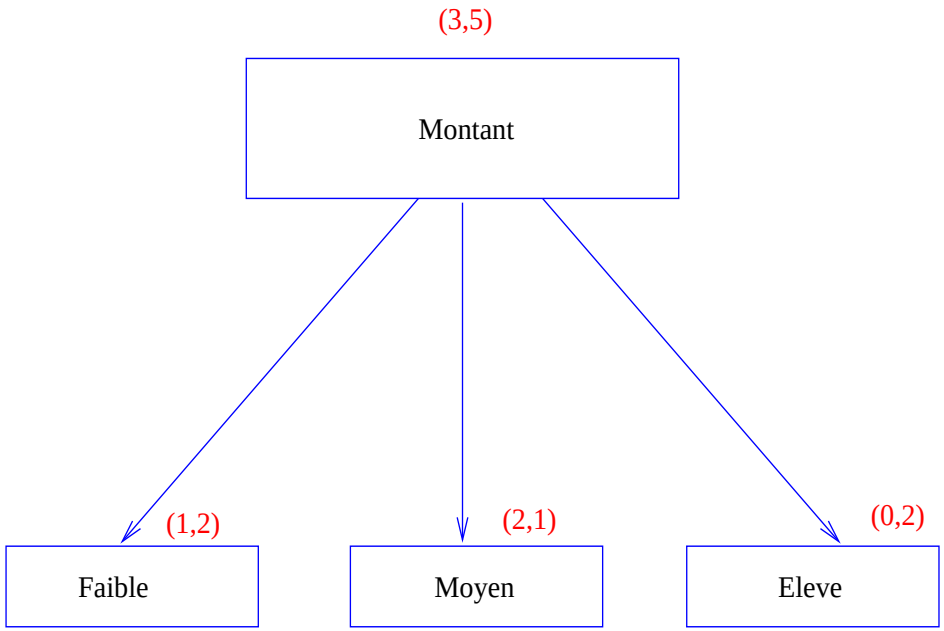
Construction de l'algorithme - exemple

- La construction est descendante
- Au début tous les individus sont regroupés
- Est-ce que le noeud initial $(3, 5)$ c'est un noeud terminal ou est-ce qu'on peut construire un test sur une variable qui permettra de mieux discriminer les individus ?
- Quatre constructions possibles, suivant les variables Montant (M), Age (A), Résidence (R) et Etudes (E)

Construction de l'algorithme - exemple

1. Construction selon la variable Montant (M)

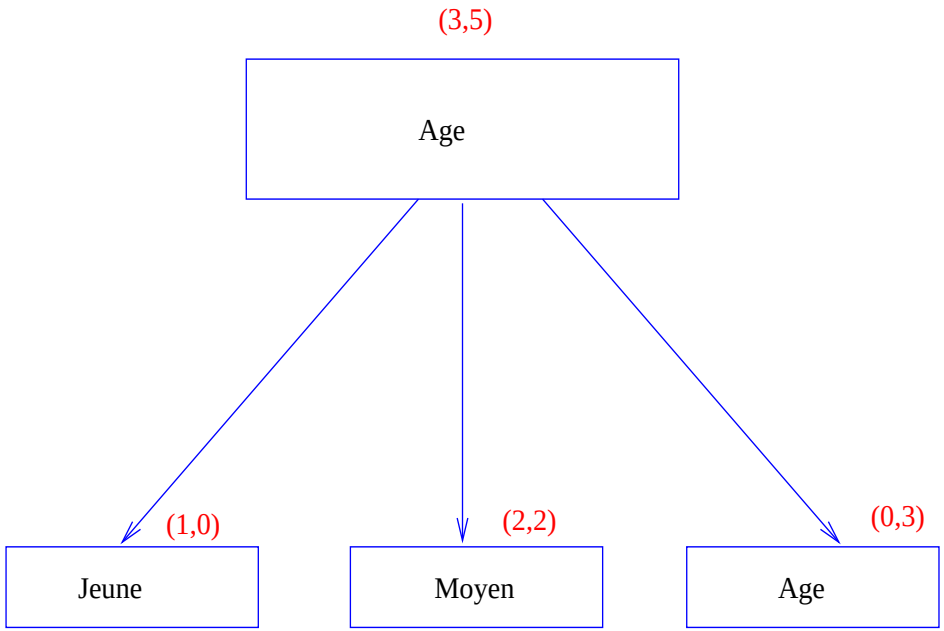
Client	M	I
1	moyen	oui
2	élevé	non
3	faible	non
4	faible	oui
5	moyen	oui
6	élevé	non
7	moyen	non
8	faible	non



Construction de l'algorithme - exemple

2. Construction selon la variable Age (A)

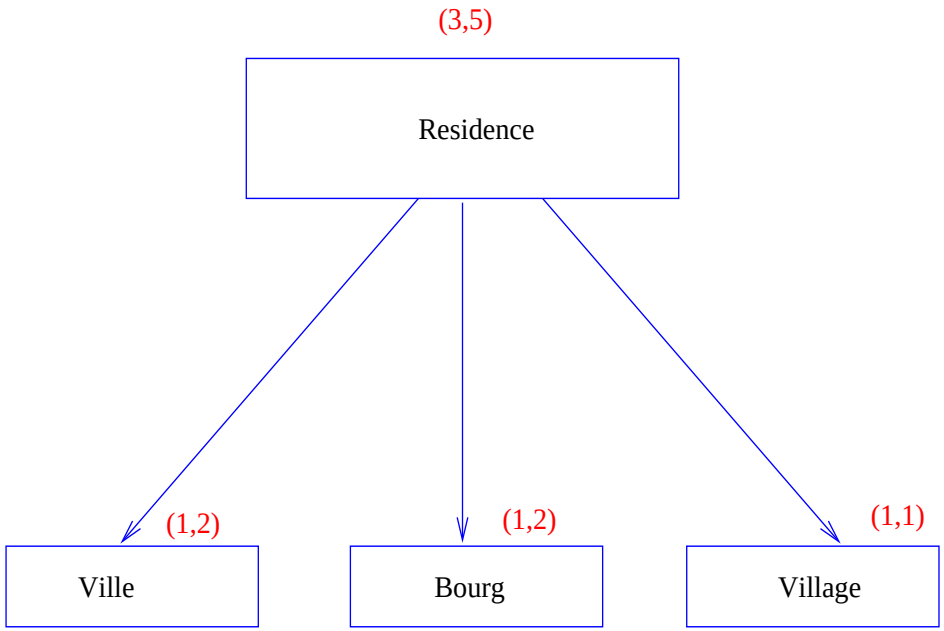
Client	A	I
1	moyen	oui
2	moyen	non
3	âgé	non
4	moyen	oui
5	jeune	oui
6	âgé	non
7	âgé	non
8	moyen	non



Construction de l'algorithme - exemple

3. Construction selon la variable Résidence (R)

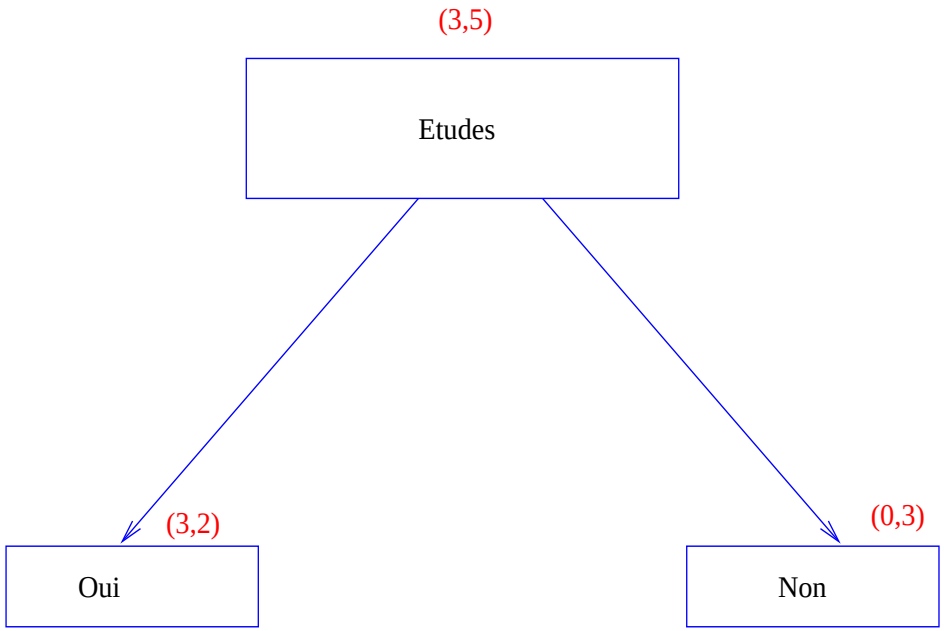
Client	R	I
1	village	oui
2	bourg	non
3	bourg	non
4	bourg	oui
5	ville	oui
6	ville	non
7	ville	non
8	village	non



Construction de l'algorithme - exemple

4. Construction selon la variable Etudes (E)

Client	E	I
1	oui	oui
2	non	non
3	non	non
4	oui	oui
5	oui	oui
6	oui	non
7	oui	non
8	non	non



Construction de l'algorithme - exemple

Quel test choisir ?

Variable test	Composition noeuds
Montant (M)	(1,2),(2,1),(0,2)
Age (A)	(1,0),(2,2),(0,3)
Résidence (R)	(1,2),(1,2),(1,1)
Etudes (E)	(3,2),(0,3)

- Sur R, aucune discrimination sur aucune branche $\Rightarrow$ On ne gagne rien avec ce test !
- Sur A, deux noeuds sur trois sont "*purs*" !
- Comment tout écrire mathématiquement ?

Degré de mélangeance

- On a besoin de comparer les différents choix possibles
- On introduit des fonctions qui permettent de mesurer le degré de mélangeance dans les différentes classes
- Propriétés des fonctions :
  - Le minimum est atteint lorsque tous les noeuds sont “*purs*”
  - Le maximum est atteint lorsque les individus sont *équirepartis* entre les classes

Degré de mélangeance

Exemples de fonctions

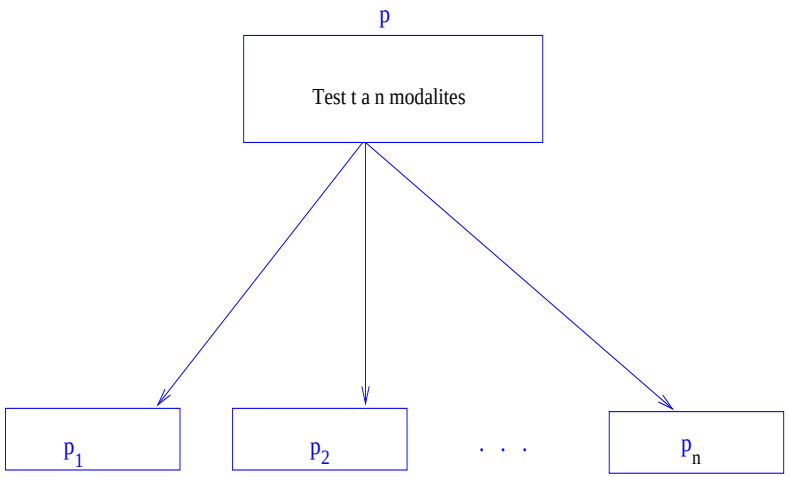
- Fonction d'entropie :

$$Entropie\,(p) = -\sum_{k=1}^{C} P\,(k|p) \ln P\,(k|p)$$

- Fonction de Gini :

$$Gini\,(p) = 1 - \sum_{k=1}^{C} P^2\,(k|p) = 2 \sum_{k<k'} P\,(k|p)\,P\,\left(k'|p\right)$$

Degré de mélangeance



- t = le test (la variable)
- n = le nombre de modalités de t
- i = la fonction pour mesurer le degré de mélangeance

On introduit la fonction de gain :

$$Gain\,(p,t) = i\,(p) - \sum_{j=1}^{n} P_j i\,(p_j)$$

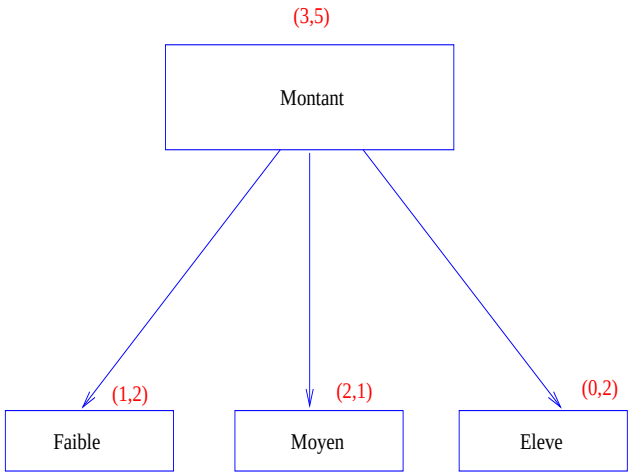
- P_j = la proportion des individus de la position p qui vont en position p_j
- La position p est fixée !

- On cherche le *test qui maximise le gain* !

Calcul du degré de mélangeance - exemple

Tester sur la variable Montant (M)

On considère le noeud 0 : (3,5)



$$Gain\,(0,M) = i\,(0) - \left(\tfrac{3}{8} i\,(1) + \tfrac{3}{8} i\,(2) + \tfrac{2}{8} i\,(3)\right)$$

On choisit i = l'entropie !

$$Entropie\,(1) = -\frac{1}{3}\ln\frac{1}{3} - \frac{2}{3}\ln\frac{2}{3} = 0.64$$

$$Entropie\,(2) = -\frac{2}{3}\ln\frac{2}{3} - \frac{1}{3}\ln\frac{1}{3} = 0.64$$

$$Entropie\,(3) = -\frac{2}{2}\ln\frac{2}{2} = 0$$

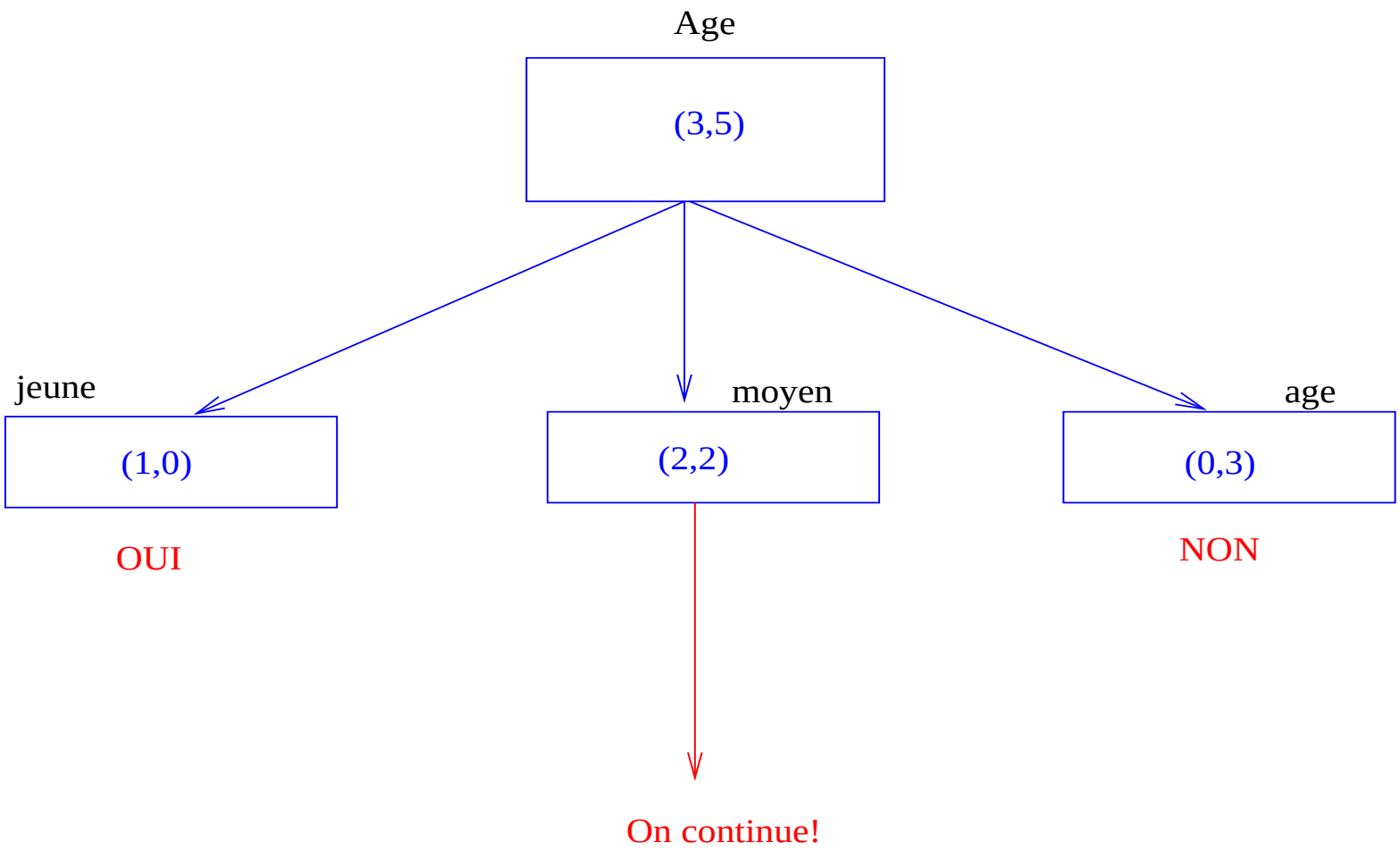
$$Gain\,(0,M) = Entropie\,(0) - 0.48$$

Calcul du degré de mélangeance - exemple

On considère le noeud 0 : (3,5)

Variable test	Gain
Montant (M)	$Gain\,(0, M) = Entropie\,(0) - 0.48$
Age (A)	$Gain\,(0, A) = Entropie\,(0) - 0.35$
Résidence (R)	$Gain\,(0, R) = Entropie\,(0) - 0.65$
Etudes (E)	$Gain\,(0, R) = Entropie\,(0) - 0.42$

Calcul du degré de mélangeance - exemple



Suite de la construction - exemple

Client	M	A	R	E	I
1	moyen	moyen	village	oui	oui
2	élevé	moyen	bourg	non	non
3	faible	âgé	bourg	non	non
4	faible	moyen	bourg	oui	oui
5	moyen	jeune	ville	oui	oui
6	élevé	âgé	ville	oui	non
7	moyen	âgé	ville	oui	non
8	faible	moyen	village	non	non

Client	M	R	E	I
1	moyen	village	oui	oui
2	élevé	bourg	non	non
4	faible	bourg	oui	oui
8	faible	village	non	non

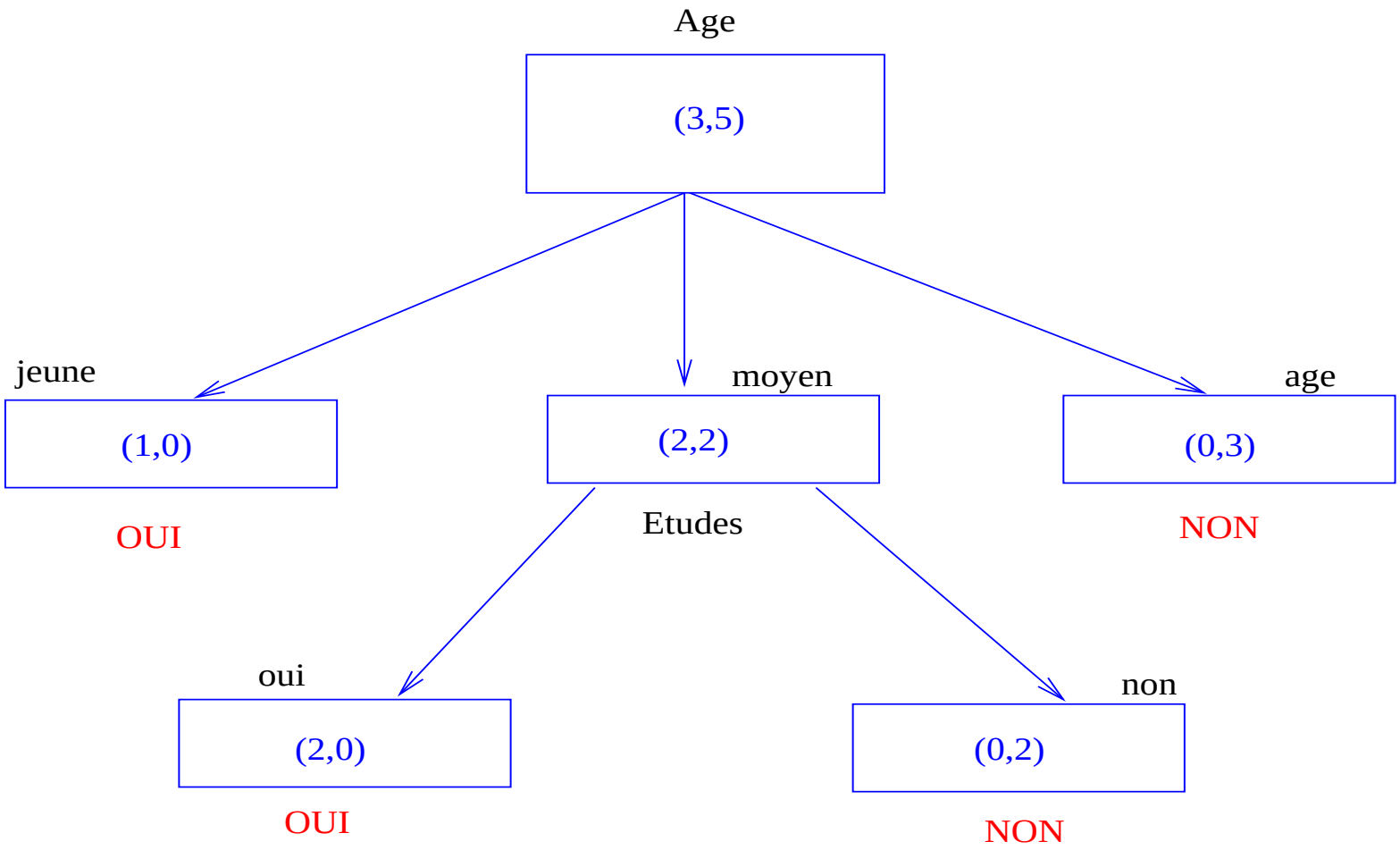
Suite de la construction - exemple

Quel test choisir ?

Variable test	Composition noeuds
Montant (M)	(1,1),(1,0),(0,1)
Résidence (R)	(1,1),(1,1)
Etudes (E)	(2,0),(0,2)

- Calculer le gain pour chaque test ?

Suite de la construction - exemple



Apprentissage des arbres de décision - Généralités

Idée

- Diviser récursivement et le plus efficacement possible les individus de l'ensemble d'apprentissage par des tests définis à l'aide des variables jusqu'à ce que l'on obtienne des sous-ensembles d'individus ne contenant (presque) que des exemples appartenant à une même classe !

Trois opérations

1. Décider si un noeud est terminal (tous les individus sont dans la même classe ou il y a moins d'un certain nombre d'erreurs)
2. Sélectionner un test à associer à un noeud (aléatoirement ou en utilisant des critères statistiques)
3. Affecter une classe à une feuille (noeud terminal) - la classe majoritaire !

Remarque

- Les différents algorithmes vont différer par ces trois opérations !

Apprentissage des arbres de décision - Algorithme

Pas 0

- échantillon S
- on initialise l'arbre vide
- le noeud courant est la racine

REPETER :
Pas 1

- décider si le noeud courant est terminal
- si *OUI*, Pas 2 : affecter une classe
- si *NON*, Pas 3 : sélectionner une décision (test) et créer un nouvel arbre
- revenir au Pas 1

Pas 4

- passer au noeud non-exploré suivant, s'il existe
- jusqu'à obtenir un arbre de décision

Arbres de décision - Eviter le surapprentissage

- On pourrait calculer un arbre “parfait” ou presque
- En pratique, un tel arbre n’existe pas toujours !

Objectif : construire un arbre avec la plus petite erreur de classification possible

Problèmes

- l’algorithme ne revient pas en arrière et ne remet pas en question ses choix !
- on peut obtenir une erreur faible sur l’ensemble d’apprentissage, mais aussi un faible pouvoir prédictif !

Arbres de décision - Eviter le surapprentissage

En général

- l'erreur d'apprentissage diminue à chaque étape
- l'erreur réelle (de validation) diminue, se stabilise et puis augmente ! (Phénomène de surapprentissage)

Eviter le surapprentissage

- on devrait pouvoir arrêter la croissance de l'arbre à tout moment $\leftrightarrow$ *pas de critère théorique pour l'instant !*
- le risque d'arrêter trop tôt » le risque d'arrêter trop tard !
- il existe des méthodes en deux phases
  1. construction de l'arbre
  2. élagage pour essayer de diminuer l'erreur réelle (de validation)

L’algorithme CART (Classification and Regression Tree)

L'algorithme CART

Arbre de décision binaire !

Des tests (variables, attributs) binaires :

1. Variables qualitatives à n modalités
   - autant de tests binaires que de partitions en deux classes
   - $2^{n-1} - 1$ tests possibles
2. Variables quantitatives
   - une infinité de découpages selon des seuils
   - le "meilleur" seuil est choisi par un expert (température<37.5) ou de manière automatique

Algorithme CART - Méthodologie

Notations

- $S \longrightarrow$ l'échantillon
- $A \longrightarrow$ l'ensemble d'apprentissage
- $T \longrightarrow$ l'ensemble de validation

Méthode en deux phases :

1. Phase d'expansion
   - construction de l'arbre
   - données A, ensemble d'apprentissage
   - fonction de Gini
2. Phase d'élagage
   - erreur testée sur l'ensemble T

Algorithme CART - Phase d'expansion

Pas 0

- échantillon A
- on initialise l'arbre vide
- le noeud courant est la racine

REPETER :
Pas 1

- décider si le noeud courant est terminal
- si *OUI*, Pas 2 : affecter une classe
- si *NON*, Pas 3 : sélectionner une décision (test) et créer un nouvel arbre
- revenir au Pas 1

Pas 4

- passer au noeud non-exploré suivant, s'il existe
- jusqu'à obtenir un arbre de décision

Algorithme CART - Phase d'expansion

Pas 1 : décider si le noeud courant est terminal

p est terminal si

- $Gini\,(p) \leq p_0$ ou $N\,(p) \leq n_0$
- p_0 et n_0 sont des paramètres fixés !

Algorithme CART - Phase d'élagage

1. Construction d'une suite d'arbres $t_0, t_1, \ldots, t_q$
2. Sélection de l'arbre final en fonction de l'erreur sur l'ensemble de validation

Remarque : si l'échantillon est trop petit, il n'y a pas de découpage en ensemble d'apprentissage et de validation $\Rightarrow$ la solution est la validation croisée !

Algorithme CART - Phase d'élagage

Construction d'une suite d'arbres $t_0, t_1, ..., t_q$

- $t_0 =$ l'arbre obtenu à la fin de la phase d'expansion
- $t_q =$ une feuille
- $t_{i+1} =$ un élagué de l'arbre t_i

Algorithme CART - Phase d'élagage

Construction de l'arbre t_{i+1} à partir de t_i

- Soit p une position dans t_i
- Soit u_p le sous-arbre de t_i en position p
- On définit

$$g(p) = \frac{\Delta_{app}(p)}{|u_p| - 1}$$

$$\Delta_{app}(p) = \frac{MC(p) - MC(u_p)}{N(p)}$$

où

- $|u_p|$ = la taille de l'arbre u_p
- $MC(p)$ = le nombre d'individus mal-classés à la position p lorsqu'on élague t_i en position p
- $MC(u_p)$ = le nombre d'individus mal-classés par le sous-arbre u_p

On choisit la position p pour laquelle $g(p)$ est minimale !

Algorithme CART - Variables continues

Sélection de la meilleure valeur de coupure pour la variable X

- On range les valeurs $X_{(1)}, X_{(2)}, ..., X_{(n)}$
- On considère $(n-1)$ bornes (en général, la moyenne de chaque segment $\frac{X_{(1)}+X_{(2)}}{2}, ..., \frac{X_{(n-1)}+X_{(n)}}{2}$)
- On teste chaque borne et on choisit celle qui minimise la fonction de Gini !

Algorithme CART - Variables continues

Remarques

- Ne pas compter deux fois les valeurs égales !
- Le temps de calcul risque d'être long si la taille de l'échantillon est importante et il y a beaucoup de variables continues

Algorithme CART - Avantages

- Pas d’hypothèses sur les variables
- Les variables peuvent être continues ou discrètes
- Méthode robuste par rapport aux valeurs abérrantes, les collinéarités ou l’hétéroscedasticité
- Facile à lire et à interpréter
- Facilement généralisable (algorithme Random Forest)