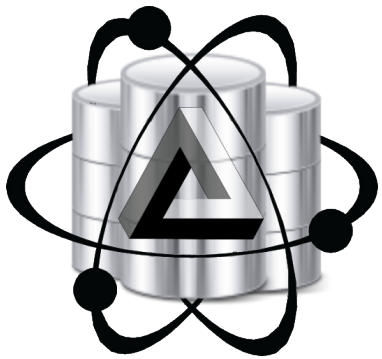
Fondements théorétiques en science des données

Arbres de décision

STT 3795

Guy Wolf
guy.wolf@umontreal.ca

Université de Montréal
Hiver 2020



Arbres de décision

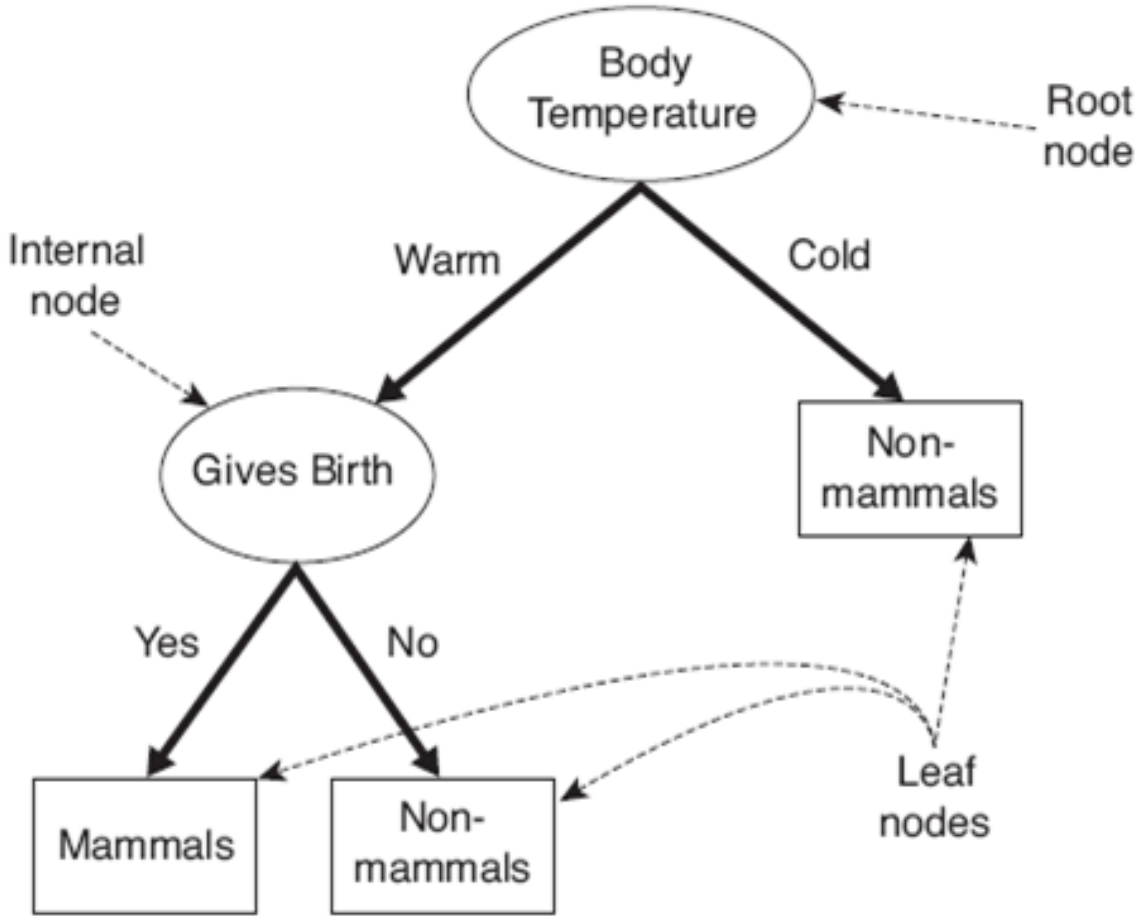
Un arbre de décision est un modèle simple, mais efficace, de classification.

L'étape d'**induction d'arbre** construit essentiellement **un ensemble de régles IF-THEN**, qui peut être visualisé comme un arbre, pour tester l'appartenance à une classe de points de données.

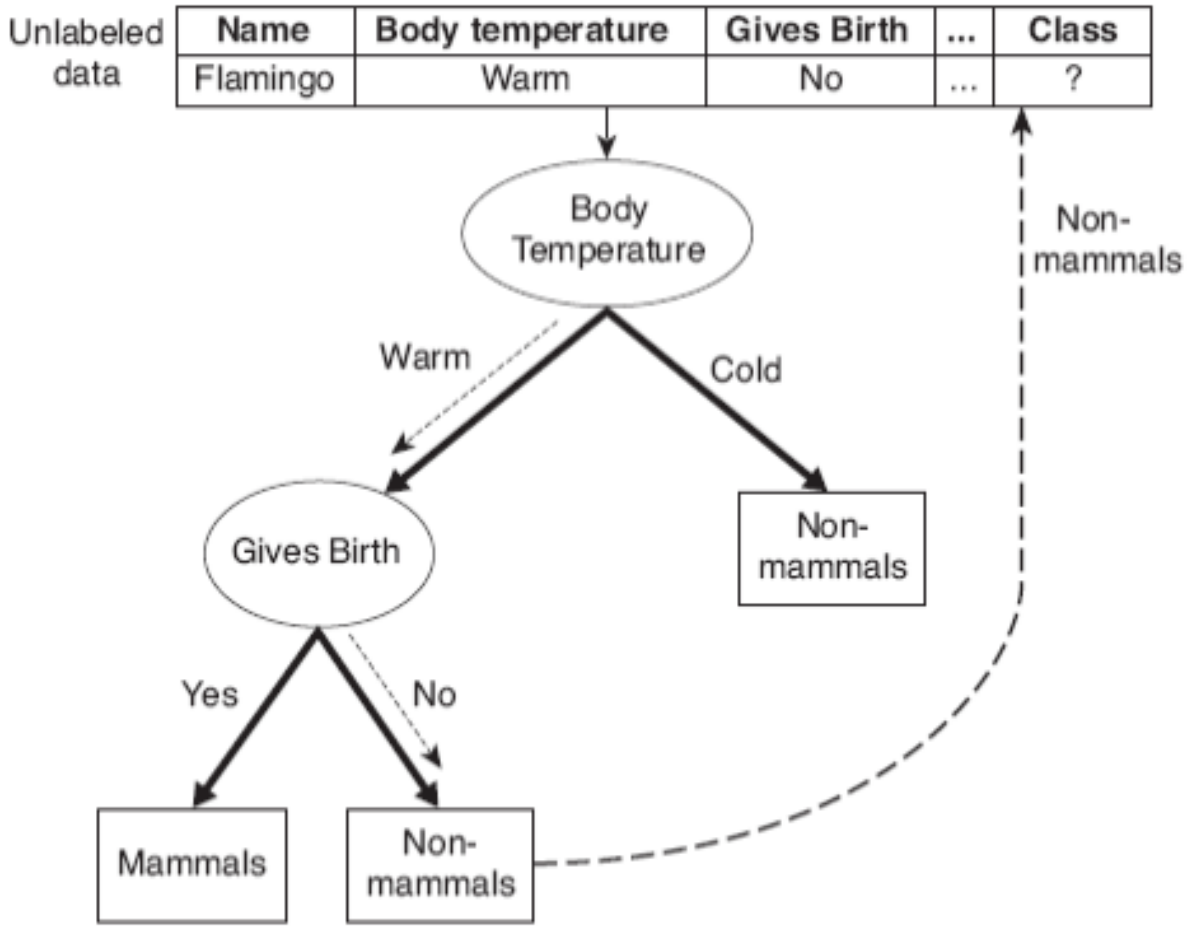
L'étape **déduction** teste ces conditions et **suit les branches de l'arbre** pour établir l'appartenance à une classe

Intuitivement, on peut y penser comme une **construction d'«interview»** pour estimer la classification de chaque point de données.

Arbres de décision



Arbres de décision



Au fil des ans, de nombreux algorithmes d'induction d'arbres de décision ont été proposés.

Exemples (Algorithmes d'induction des arbres de décision)

- CART (Classification And Regression Trees)
- ID3 (Iterative Dichotomiser 3) & C4.5
- SLIQ & SPRINT
- Rainforest & BOAT

La plupart suivent un paradigme top-down de base connu comme «Hunt's algorithm», bien que certains utilisent des approches alternatives (p.ex., des constructions bottom-up) et des étapes d'implémentation particulières pour améliorer les performances.

Arbres de décision

Approche de base (Hunt's algorithm)

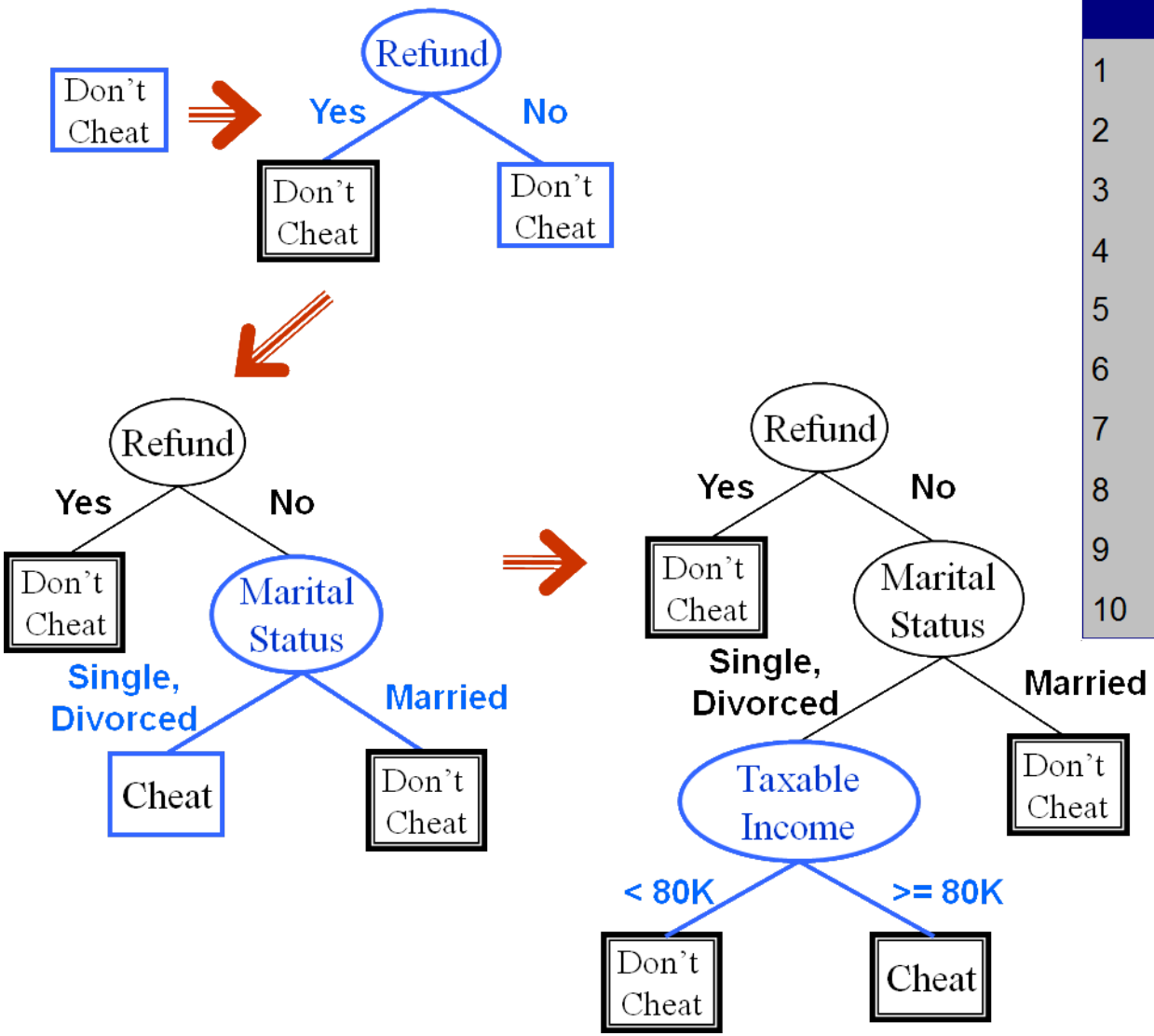
Un arbre est construit de haut en bas en utilisant une approche gloutonne récursive:

1. Start with all the training samples at the root
2. Choose the best attribute & split into several data subsets
3. Create a branch & child node for each subset
4. Run the algorithm recursively for each child node and associated subset
5. Stop the recursion when one of the following conditions are met:
   - All the data points in the node have the same class label
   - There are no attributes left to split by
   - The node is empty

Si un nœud de feuille contient plus d'une étiquette de classe, utilisez un vote à la majorité/pluralité pour définir sa classe.

Arbres de décision

Approche de base (Hunt's algorithm)



Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Arbres de décision

Division de nœuds

Chaque nœud interne de l'arbre prend en compte :

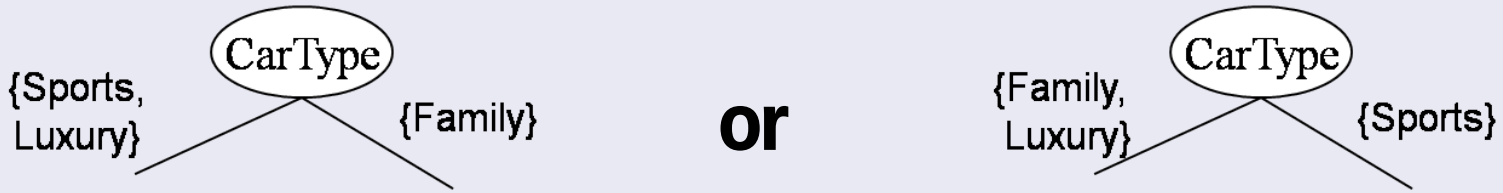
- un sous-ensemble de données, basé sur le chemin qui y mène
- un attribut permettant de tester et de générer des sous-ensembles plus petits à transmettre aux nœuds enfants

La division d'un nœud en nœuds enfants dépend du type de l'attribut testé et de la configuration de l'algorithme.

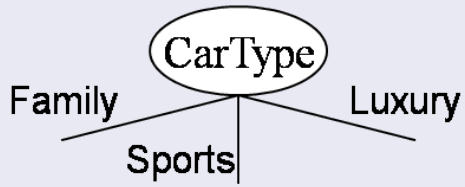
Par exemple, certains algorithmes forcent des divisions binaires (p.ex., CART), tandis que d'autres permettent des divisions multivoies (p.ex., C4.5).

Division par attributs nominaux:

Divisions binaires: utilisez un ensemble de valeurs possibles sur une branche et son complément sur l'autre:

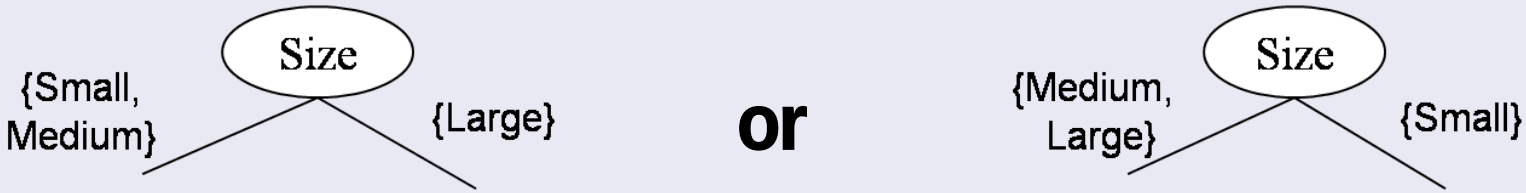


Divisions multivoies: utilisez une branche distincte pour chaque valeur possible:

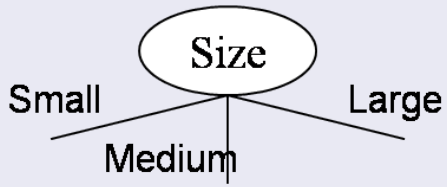


Division par attributs ordinaux:

Divisions binaires: trouvez un seuil et partitionnez en valeurs supérieures et inférieures:

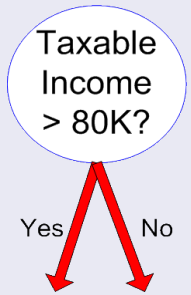


Divisions multivoies: utilisez une branche distincte pour chaque valeur possible:

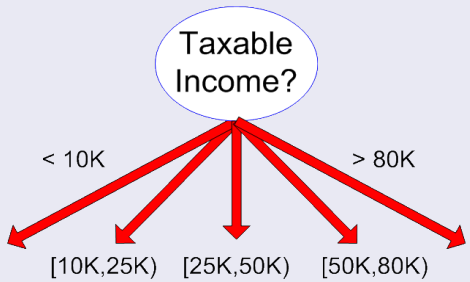


Division par attributs numériques:

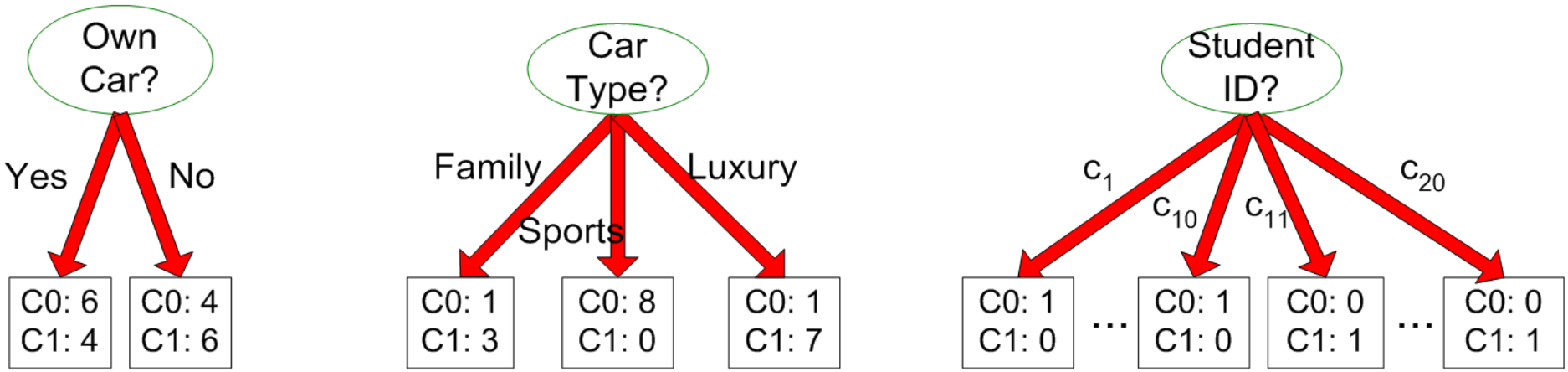
Divisions binaires: trouvez un seuil et partitionnez en valeurs supérieures et inférieures:



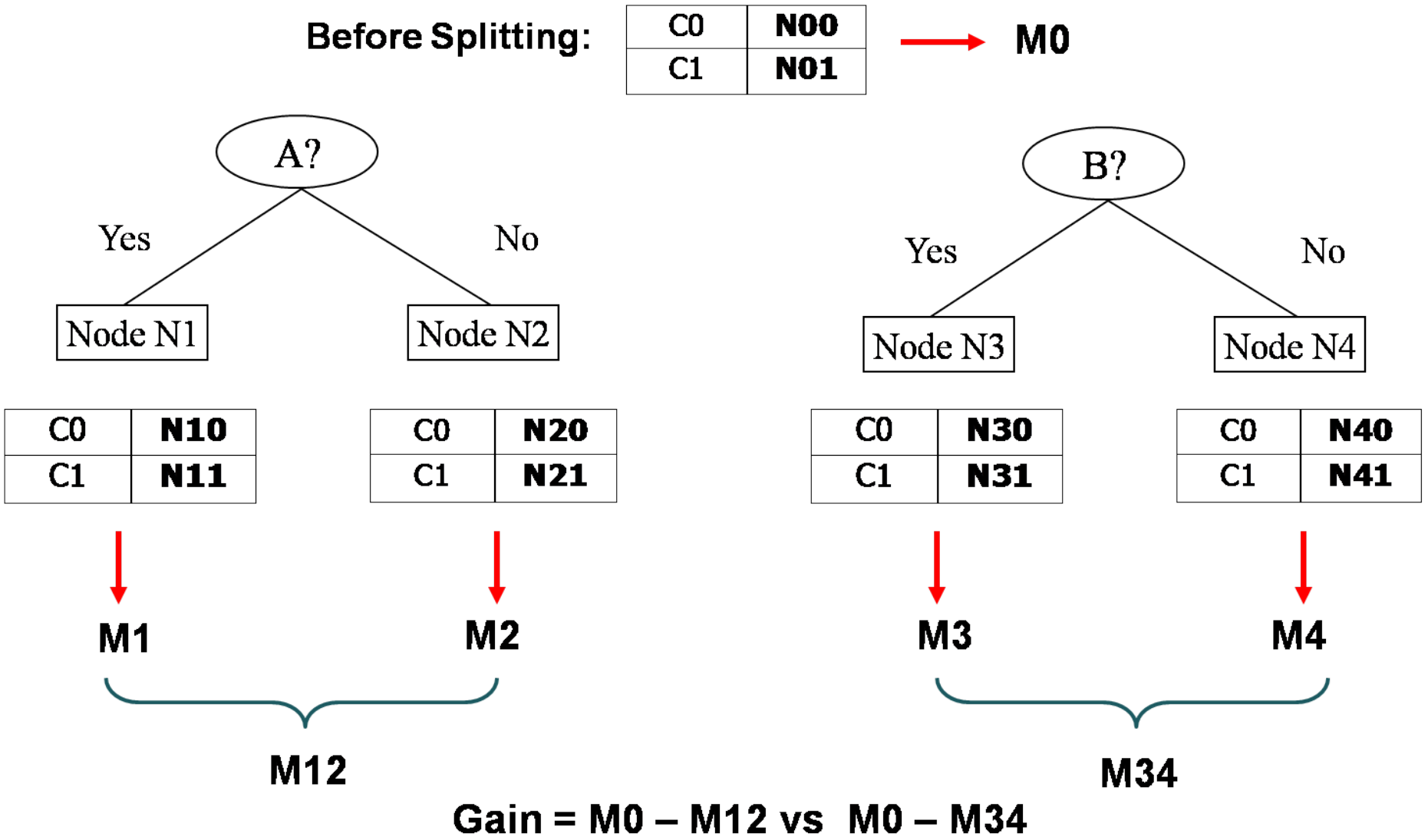
Divisions multivoies: discrétisez les valeurs (statiquement comme un prétraitement ou dynamiquement) pour former des valeurs ordinales :



Comment choisit-on le meilleur attribut (et la meilleure division) à utiliser à chaque nœud?



On veut augmenter l'homogénéité et réduire l'hétérogénéité des sous-nœuds qui en résultent. En d'autres termes - on cherche des sous-ensembles aussi purs que possible p/r aux classes.


Before Splitting:
C0 N00
C1 N01
M0
A?
Yes
No
Node N1
Node N2
B?
Yes
No
Node N3
Node N4
C0 N10
C1 N11
C0 N20
C1 N21
C0 N30
C1 N31
C0 N40
C1 N41
M1
M2
M3
M4
M12
M34
Gain = M0 – M12 vs M0 – M34

L'impureté peut être quantifiée de plusieurs façons, qui varient d'un algorithme à l'autre :

Mesures d'impureté

- Erreur de classification
- Entropie (e.g., ID3 and C4.5)
- Gini index (e.g., CART, SLIQ, and SPRINT)

En général, ces mesures sont équivalentes dans la plupart des cas, mais il existe des cas spécifiques où l'une d'entre elles peut être avantageuse par rapport aux autres.

L'impureté peut être quantifiée de plusieurs façons, qui varient d'un algorithme à l'autre :

Mesures d'impureté

- Erreur de classification
- Entropie (e.g., ID3 and C4.5)
- Gini index (e.g., CART, SLIQ, and SPRINT)

Le gain d'impureté d'une division $t \rightarrow t_1, \ldots, t_k$ est la différence

$$\Delta_{\text{Impurity}} = \text{Impurity}(t) - \sum_{i=1} \frac{\#\text{pts}(t_i)}{\#\text{pts}(t)} \text{Impurity}(t_i)$$

entre l'impureté à t et la moyenne pondérée des impuretés enfantines.

Erreur de classification

Le taux d'erreur encouru en classant le nœud entier par vote de pluralité:

$$\text{Error}(t) = 1 - \max_c\{p(c|t)\}$$

où $p(c|t)$ est la fréquence de la classe c dans le noeud t.

- L'erreur minimale est de zéro - obtenue lorsque tous les points de données du nœud ont la même classe
- L'erreur maximale est de $1 - \frac{1}{\#\text{classes}}$ - obtenue lorsque les points de données dans le nœud sont également répartis entre les classes

Exemples (Erreur de classification)

C1	**0**
C2	**6**

P(C1) = 0/6 = 0 **P(C2) = 6/6 = 1**

Error = 1 – max (0, 1) = 1 – 1 = 0

C1	**1**
C2	**5**

P(C1) = 1/6 **P(C2) = 5/6**

Error = 1 – max (1/6, 5/6) = 1 – 5/6 = 1/6

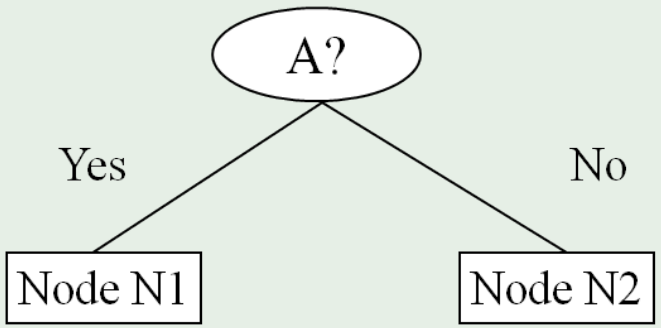
C1	**2**
C2	**4**

P(C1) = 2/6 **P(C2) = 4/6**

Error = 1 – max (2/6, 4/6) = 1 – 4/6 = 1/3

L'erreur de classification ne détecte pas toujours d'améliorations:

Exemple



	Parent
C1	7
C2	3
Error = 3/10	

Error(N1)
= 1 – 3/3 = 0

Error(N2)
= 1 – 4/7 = 3/7

	N1	N2
C1	3	4
C2	0	3
Error=3/10		

Error(Children)
= 3/10 • 0 + 7/10 • 3/7
= 3/10

Error doesn't improve!

Entropie

Concept standard de la théorie d'information qui mesure l'impureté d'un nœud en fonction du nombre de «bits» nécessaires pour y représenter les étiquettes de classe:

$$\text{Entropy}(t) = -\sum_{c} p(c|t) \log_2 p(c|t)$$

où $p(c|t)$ est la fréquence de la classe c dans le noeud t.

- L'entropie minimale est de zéro - obtenue lorsque tous les points de données du nœud ont la même classe
- L'entropie maximale est de log(#classes) - obtenue lorsque les points de données dans le nœud sont également répartis entre les classes

Exemples (Entropie)

C1	**0**
C2	**6**

P(C1) = 0/6 = 0 P(C2) = 6/6 = 1

Entropy = – 0 $\log_2 0$ –1 $\log_2 1$ = – 0 – 0 = 0

C1	**1**
C2	**5**

P(C1) = 1/6 P(C2) = 5/6

Entropy = – (1/6) $\log_2$ (1/6) – (5/6) $\log_2$ (1/6) = 0.65

C1	**2**
C2	**4**

P(C1) = 2/6 P(C2) = 4/6

Entropy = – (2/6) $\log_2$ (2/6) – (4/6) $\log_2$ (4/6) = 0.92

«Information Gain»

Pour un nœud t divisé en nœuds enfants $t_1, \ldots, t_k$, le gain d'information de cette division est défini comme suit :

$$\text{Info\_Gain}(t, t1, \ldots, t_k) = \text{Entropy}(t) - \sum_{i=1} \frac{\#\text{pts}(t_i)}{\#\text{pts}(t)} \text{Entropy}(t_i)$$

où $\#\text{pts}(\cdot)$ est le nombre de points de données dans un nœud.

- Il mesure la réduction de l'entropie obtenue par la scission - une division optimale maximiserait ce gain.
- Désavantage : il tend à préférer un grand nombre de petits nœuds enfants purs (p.ex., provoquant de surajustement)

«Gain Ratio»

Pour un nœud t divisé en nœuds enfants $t_1, \ldots, t_k$, le ratio de gain normalise le gain d'information par

$$\text{Split\_Info}(t, t_1, \ldots, t_k) = -\sum_{i=1}^{k} \frac{\#\text{pts}(t_i)}{\#\text{pts}(t)} \log_2 \frac{\#\text{pts}(t_i)}{\#\text{pts}(t)}$$

pour obtenir $\text{Gain\_Ratio} = \frac{\text{Info\_Gain}}{\text{Split\_Info}}$.

- Il pénalise les partitions à forte entropie (c-à-d, avec un grand nombre de petits nœuds enfants)
- Il est utilisé dans le C4.5 pour surmonter les désavantages du gain d'information brute.

Gini index

Un indice d'«inégalité sociale» (ou, plus formellement, de dispersion statistique) développé par le statisticien/sociologue Corrado Gini :

$$\text{Gini}(t) = 1 - \sum_c [p(c|t)]^2$$

où $p(c|t)$ est la fréquence de la classe c dans le noeud t.

- La valeur minimale de Gini est de zéro - obtenue lorsque tous les points de données du nœud ont la même classe
- L'indice de Gini maximal est de $1 - \frac{1}{\#\text{classes}}$ - obtenu lorsque les points de données dans le nœud sont également répartis entre les classes

Exemples (Gini index)

C1	0
C2	6

$P(C1) = 0/6 = 0 \quad P(C2) = 6/6 = 1$

$Gini = 1 - P(C1)^2 - P(C2)^2 = 1 - 0 - 1 = 0$

C1	1
C2	5

$P(C1) = 1/6 \quad P(C2) = 5/6$

$Gini = 1 - (1/6)^2 - (5/6)^2 = 0.278$

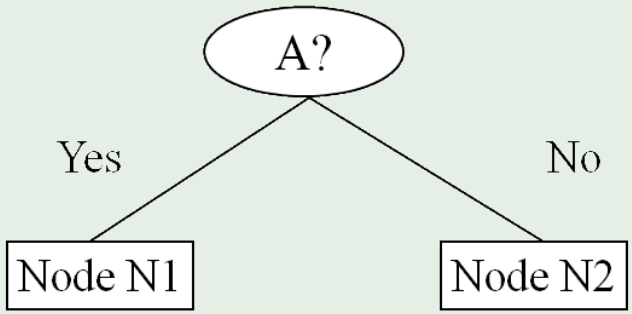
C1	2
C2	4

$P(C1) = 2/6 \quad P(C2) = 4/6$

$Gini = 1 - (2/6)^2 - (4/6)^2 = 0.444$

La valeur de Gini pour une division est calculée de la même manière que pour une erreur de classification, mais elle fonctionne mieux:

Exemple



	Parent
C1	7
C2	3
Gini = 0.42	

Gini(N1)
= 1 – (3/3)² – (0/3)²
= 0

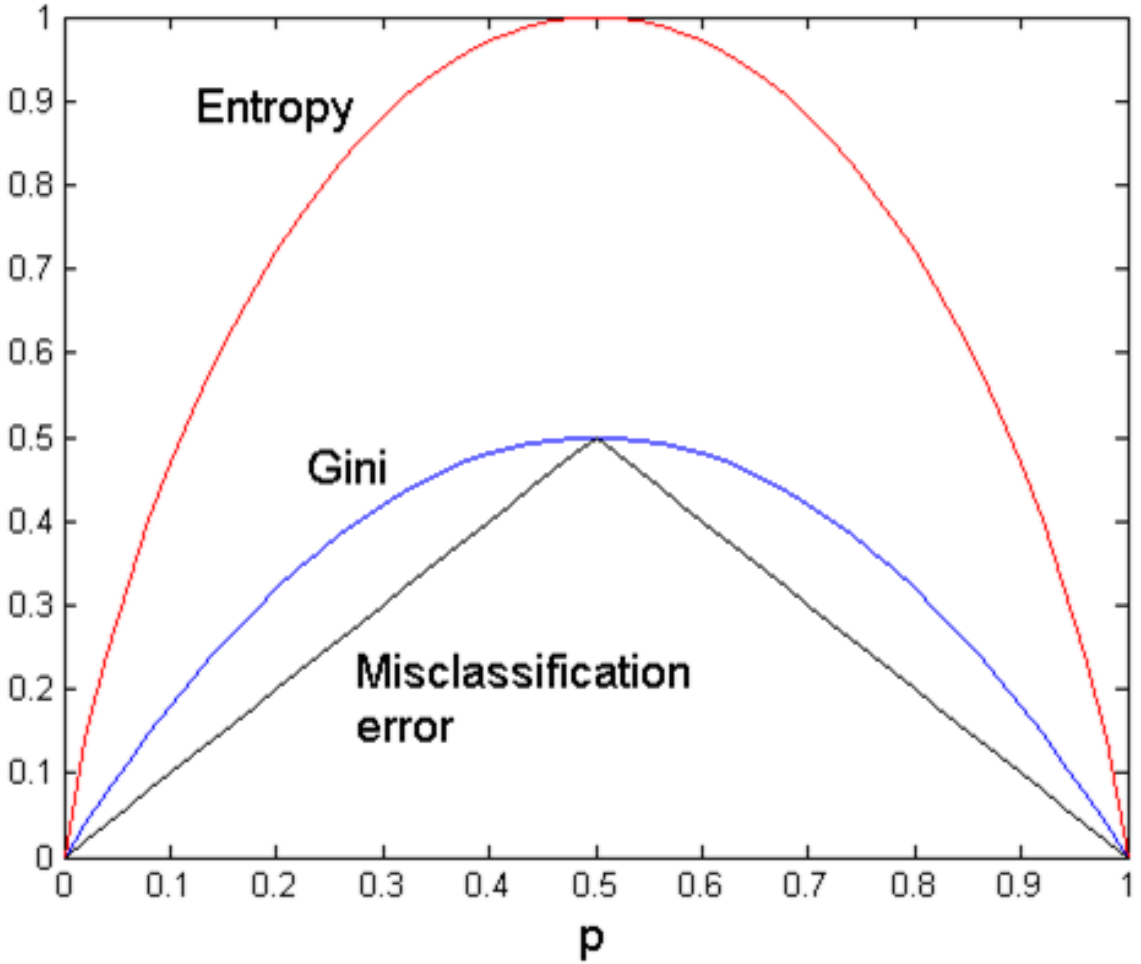
Gini(N2)
= 1 – (4/7)² – (3/7)²
= 0.489

	N1	N2
C1	3	4
C2	0	3
Gini=0.342		

Gini(Children)
= 3/10 * 0
+ 7/10 * 0.489
= 0.342

Gini improves!

Comparaison des trois mesures d'impuretés pour deux classes, où p est la portion de points dans l'une (et $1 - p$ dans l'autre):



Des études ont montré que le choix des mesures d'impuretés a peu d'effet sur la qualité de la classification. Néanmoins, chaque choix a son propre biais.

Information gain:

- biaisé en faveur des attributs multivalorisés

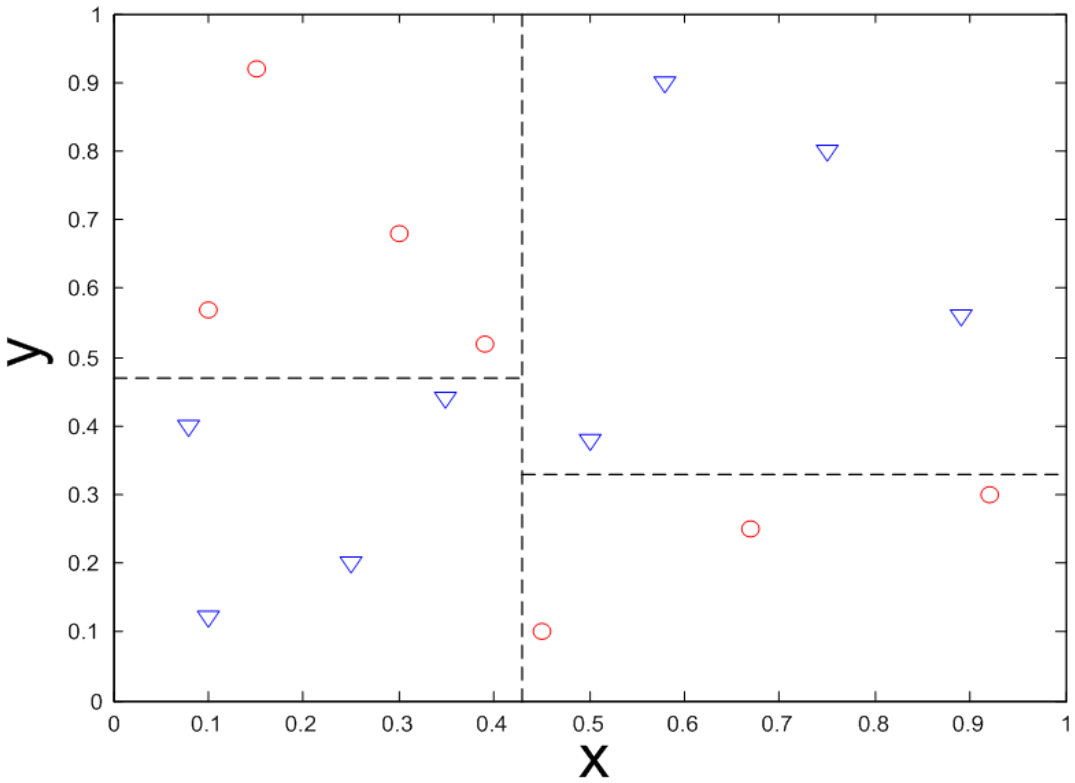
Gain ratio:

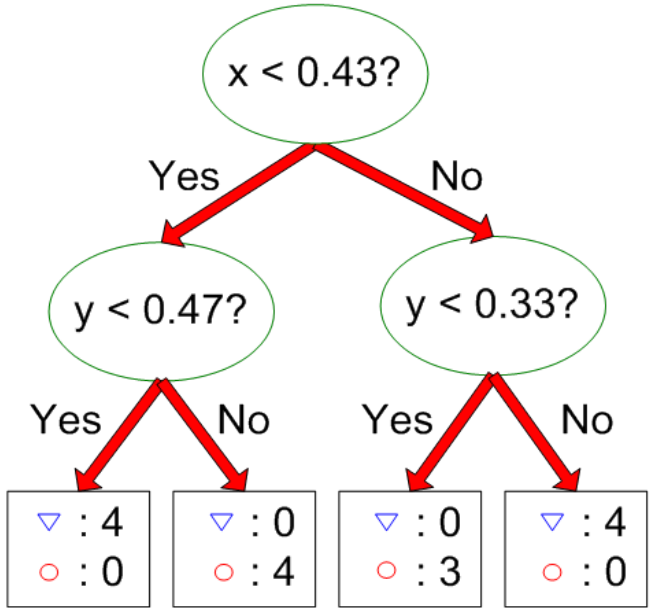
- a tendance à préférer les divisions déséquilibrées où une partition est nettement plus petite que les autres

Gini index:

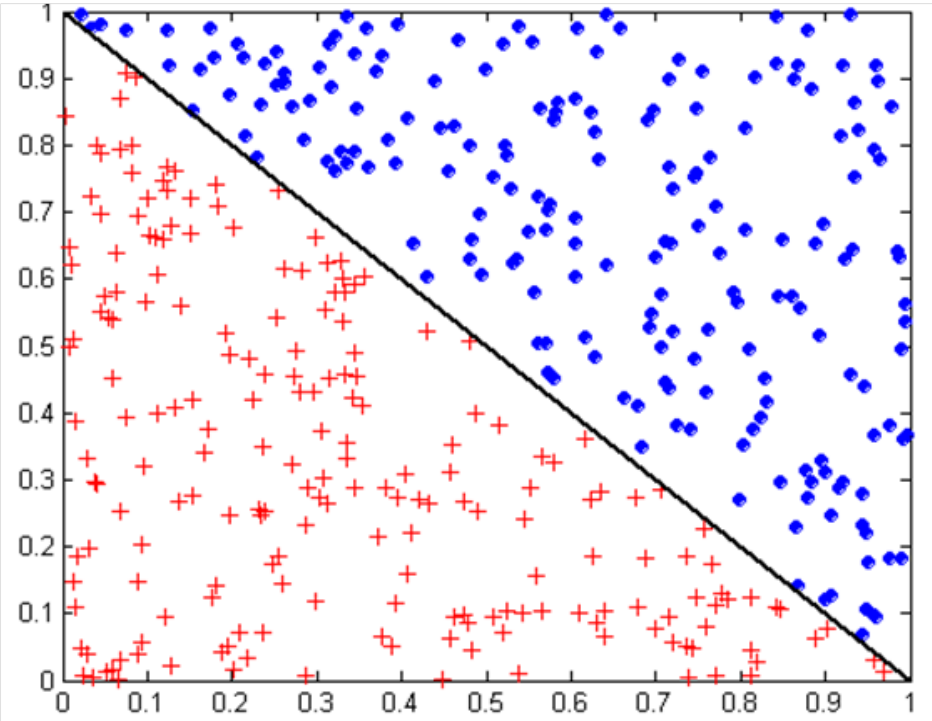
- biaisé en faveur des attributs multivalorisés
- a des difficultés quand la #classes est grande
- tend à favoriser des partitions de même taille et de même pureté

Les frontières des décisions indiquent quelles régions correspondent à quelle classe:

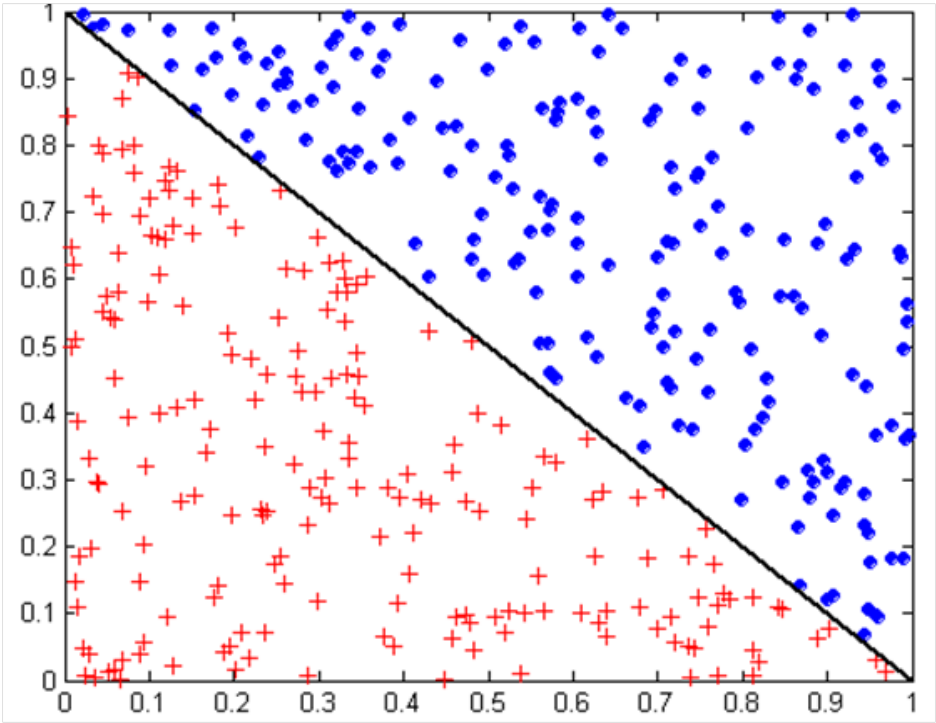


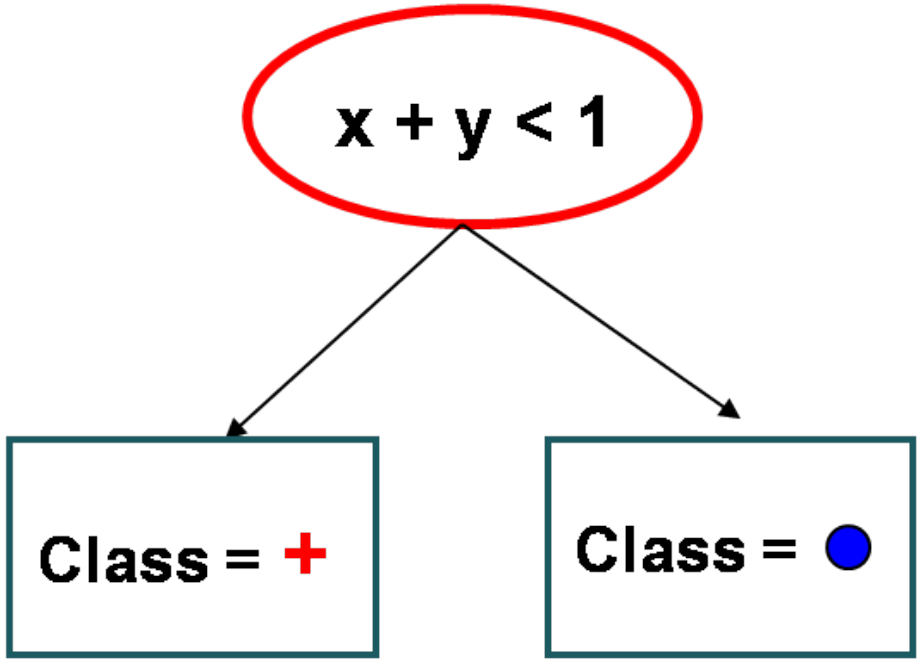


Et si nous voulons considérer les régions non rectangulaires?



Et si nous voulons considérer les régions non rectangulaires?





Les arbres de décision obliques prennent en compte les combinaisons linéaires d'attributs

Arbres de décision

L'élagage des arbres

Des arbres de décision trop grands peuvent facilement surajuster les données d'entraînement et ne pas bien généraliser.

Dans ce cas, la complexité du modèle peut être quantifiée par le nombre de nœuds dans l'arbre. Pour éviter le surajustement, la taille de l'arbre induit doit être limitée.

On peut aussi utiliser le principe de longueur minimale de description (LMD) de la théorie d'information:

LMD

Minimisez $\text{Cost}(\text{data}, \text{model}) = \text{Cost}(\textit{model}) + \text{Cost}(\text{data}|\text{model})$, où ce dernier coût ne tient compte que des étiquettes de classe pour les points mal classifiés.

La taille de l'arbre de décision est réduite par l'élagage:

Pre-élagage

Arrêtez l'algorithme d'apprentissage de l'arbre avant que l'arbre n'ait atteint sa pleine taille en utilisant des conditions d'arrêt restrictives, telles que:

- le gain en impuretés est inférieur à un seuil donné
- la taille du sous-ensemble considéré est inférieure à un seuil

Post-élagage

Apprenez d'abord un arbre complet, puis coupez-le en utilisant les opérations suivantes:

- Remplacement d'un sous-arbre - couper un sous-arbre et le remplacer par une feuille
- Soulèvement du sous-arbre - utiliser la branche la plus traversée à un nœud, élever son sous-arbre d'un niveau, et éliminer les sous-arbres frères.

Forêts aléatoires

Ensemble d'arbres de décision

Alors que les arbres de décision sont simples et efficaces, ils sont également sensibles aux variations d'entraînement et au suréquipement. Pour réduire la variance et augmenter la stabilité, plusieurs arbres de décision peuvent être combinés ensemble pour former une forêt:

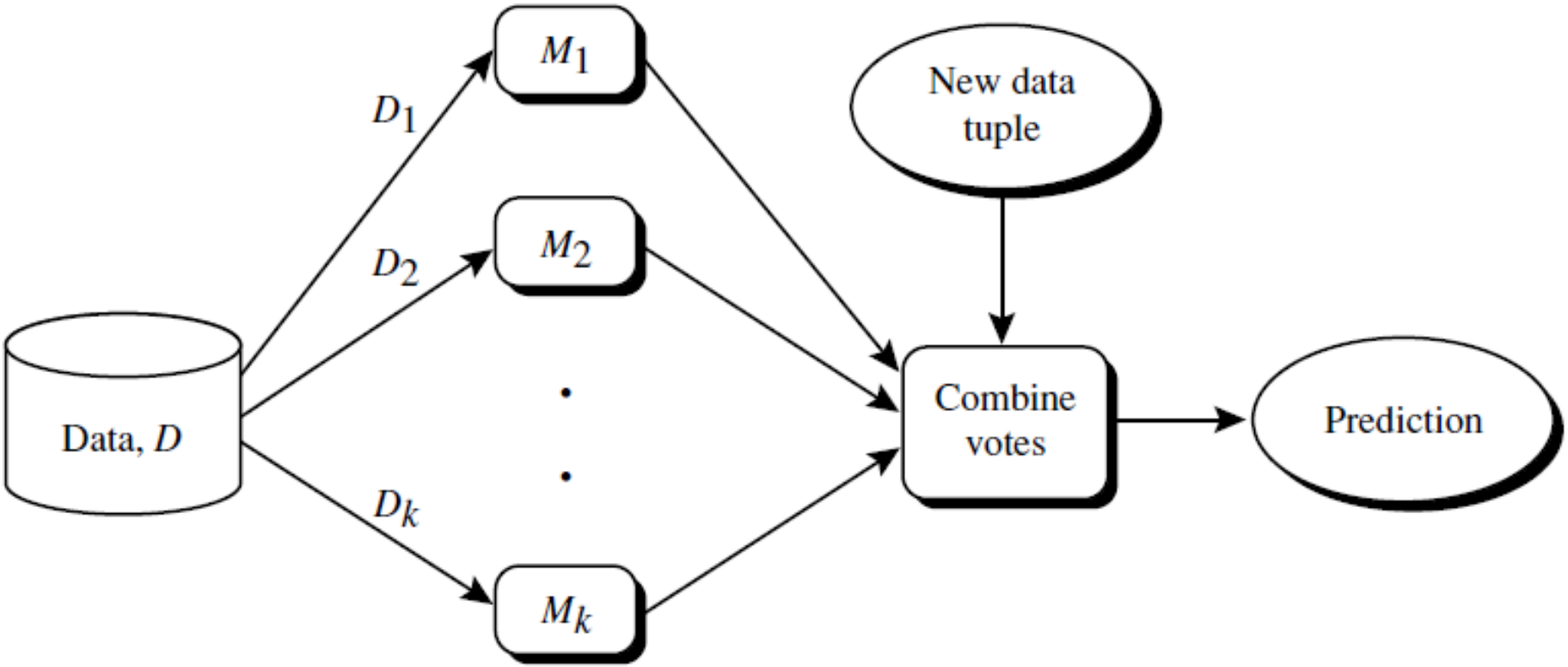
Forêt aléatoire

Une méthode d'ensemble qui combine plusieurs arbres de décision et agrège leurs résultats. Pour construire les arbres, plusieurs vecteurs aléatoires sont échantillonnés i.i.d. de la même distribution et chaque construction individuelle d'arbre de décision dépend des données et de l'un de ces vecteurs.

Les forêts aléatoires sont plus efficaces en termes de calcul que les autres méthodes d'ensemble, tout en ayant une précision comparable.

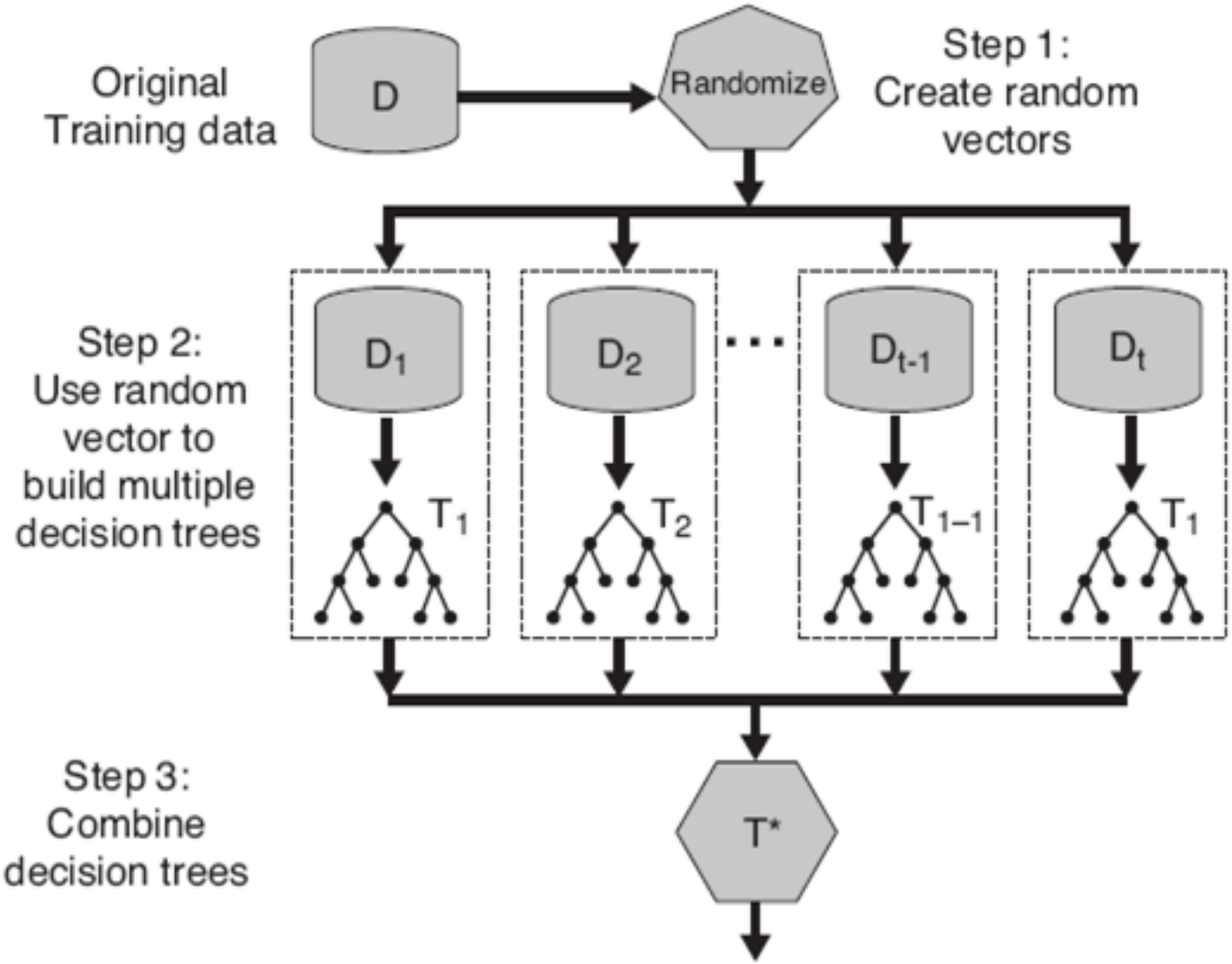
Forêts aléatoires

Ensemble d'arbres de décision



Forêts aléatoires

Ensemble d'arbres de décision



Forêts aléatoires

Des approches de randomisation des arbres

Les entrées de l'arbre de décision peuvent être randomisées de deux façons:

Sélection aléatoire des entrées (Forest-RI):

Sélectionnez aléatoirement un sous-ensemble d'attributs comme candidats à la division des nœuds au lieu d'utiliser tous les attributs des données. Cette sélection aléatoire est généralement effectuée séparément à chaque nœud.

Combinaisons linéaires aléatoires (Forest-RC):

Créez de nouveaux attributs qui sont une combinaison linéaire (aléatoire) d'attributs existants. Ce qui réduit la corrélation entre les classificateurs individuels, mais permet également de définir des limites de décision non rectangulaires.

Ce dernier peut être lié à des projections aléatoires, qui sont également utilisées dans d'autres tâches (p.ex., la recherche de kNN et la réduction de la dimensionnalité).

Projections aléatoires

Random projections (RP) are a popular an efficient way to project (numerical) data into a space of arbitrary dimensions, without having to learn the embedding function from the data.

For a dataset of n data points in d dimensions, a typical construction uses the following steps:

- Choose target dimension $k > 0$, draw $k \cdot d$ samples i.i.d. from a given distribution, and organize them in a matrix $R \quad \mathbb{R}^{d \times k}$
- Scale R by some factor to get matrix A (e.g., $A = k^{-1/2}R$)
- Organize the data points as rows of a matrix $X \quad \mathbb{R}^{n \times d}$ and embed via matrix multiplication $XA \quad \mathbb{R}^{n \times k}$

Random forests (FC) can be regarded as building decision trees on multiple instantiations of RP.

La motivation et le raisonnement qui sous-tendent le RP proviennent du célèbre lemme JL et de sa preuve.

Lemme (Johnson-Lindenstrauss, 1984)

Soit $X \quad \mathbb{R}^d$ *un ensemble de données fini de taille* $|X| = n$. *Pour tout* $0 < \varepsilon < 1$ *et* $k > 8\frac{\ln(n)}{\varepsilon^2}$ *arbitraires, il existe un plongement linéaire de* X *dans* $\mathbb{R}^k$ *tel que* $(1 - \varepsilon) \leq \frac{f(x)-f(y)^2}{x-y^2} \leq (1 + \varepsilon)$.

Lemme (Version alternative - lemme JL distributionnel)

Étant données des $0 < \varepsilon, \delta < \frac{1}{2}$ *et* $k > \frac{C}{\varepsilon^2}\ln(\frac{1}{\delta})$ *arbitraires, où* C *est une constante, soit* R *une matrice* $k \times d$ *t.q.* $R_{ij} \overset{i.i.d.}{} N(0,1)$. *Alors, pour* $A = \frac{1}{\sqrt{k}}R$ *et pour tout* $x \quad \mathbb{R}^d$ *unitaire,*
$\Pr[|\ Ax\ ^2 - 1| \leq \varepsilon] \geq (1 - \delta)$.

Il existe plusieurs façons de construire une matrice de projection aléatoire, p.ex.:

Exemple (Achlioptas 2001)

Un schéma simple mais efficace de la matrice de projection consiste à tirer aléatoirement (i.i.d.) la matrice R (rappel $A = \sqrt{k}^{-1} R$) de:

$$R_{ij} = \begin{cases} +\sqrt{s} & \text{with probability } \frac{1}{2s} \\ 0 & \text{with probability } (1 - \frac{1}{s}) \\ -\sqrt{s} & \text{with probability } \frac{1}{2s} \end{cases}$$

où $s > 0$ contrôle la sparsité de la matrice construite.

D'autres constructions permettent des matrices plus clairsemées, ou bien imposent l'orthogonalité pour une projection propre.

Récapitulation

Les arbres de décision sont des classificateurs simples et populaires

- L'algorithme d'induction conçoit un partitionnement hiérarchique des données en sous-ensembles homogènes qui se composent pour la plupart d'une seule classe
- La classification des nouveaux points de données se fait en suivant les branches de l'arbre et le vote majoritaire dans les nœuds des feuilles

Le surajustement et la variance élevée sont courants avec les arbres de décision, mais peuvent être atténués par l'élagage et la randomisation

Les forêts aléatoires sont un exemple populaire de «ensemble classification»

- Ils utilisent des arbres multiples pour obtenir des frontières de décision solides et flexibles
- La randomisation est réalisée par la sélection ou la projection d'éléments aléatoires avant la construction de chaque arbre de décision