

INF5081

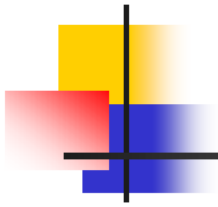
---

# Forage de données – Analyse prédictive

## ■ Arbre de décision

Mohamed Bouguessa

---



# Plan

---

- 1- Analyse prédictive
- 2- Classification
- 3- Arbre de décision
  - Construction de l'arbre
  - Élagage de l'arbre



# Classification des algorithmes d'apprentissage

---

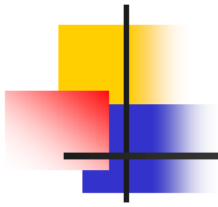
- Les algorithmes d'apprentissage sont classés en deux catégories :
  1. **Les algorithmes prédictifs**
    - **Inférence sur les données pour faire des prédictions**
  2. Les algorithmes descriptifs
    - Description des propriétés des données



# Algorithmes prédictifs

---

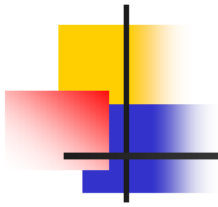
- **Principe**
  - Diviser / grouper les instances dans des **classes** spécifiques pour des prédictions futures
- **Exemples**
  - Les clients loyaux / les clients non loyaux
  - Les transactions frauduleuses / les transactions générales
- **Approche :**
  - Analyse prédictive
  - **Le forage de données prédictif c'est**
    - **l'apprentissage artificiel (Machine learning)**
    - Dans le reste de ce document, le forage de données prédictif et l'apprentissage artificiel désignent le même concept



# Apprentissage artificiel - Généralités

---

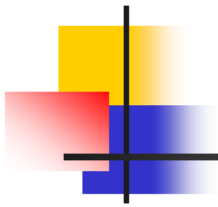
- **Apprentissage artificiel (Machine Learning) :**
  - Est une tentative de comprendre et reproduire cette faculté d'apprentissage dans des systèmes artificiels.
  - Il s'agit, très schématiquement, de concevoir des algorithmes capables, à partir d'un nombre important d'exemples (les données correspondant à « l'expérience passée »), d'en assimiler la nature afin de pouvoir appliquer ce qu'ils ont ainsi appris aux cas futurs.



# Apprentissage artificiel

---

Format des données utilisées dans  
l'analyse prédictive



# Données en entrée

Attributs				Catégorie/ classe
A1	A2	A3	A4	Y
-0.69	-0.72	Y	0.47	Healthy
-2.3	-1.2	N	0.15	Disease
0.32	-0.9	N	-0.76	Healthy
0.37	-1	Y	-0.59	Disease
-0.67	-0.53	N	0.33	Healthy
0.51	-0.09	Y	-0.05	Disease

Enregistrements / Objets

Données d'apprentissage

- Tâche : trouver un modèle qui « *approxime et généralise au mieux* » la relation entre les objets et leurs catégories;



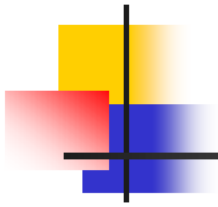
# But

---

La prédiction  
**classification** de nouvelles données.

A1	A2	A3	A4	Y
0.83	-0.54	T	0.68	Healthy
-2.3	-1.2	F	-0.83	Disease
0.08	0.63	F	0.76	Healthy
0.06	-0.29	T	-0.57	Disease
-0.98	-0.18	F	-0.38	Healthy
-0.68	0.82	T	-0.95	Disease
0.92	-0.33	F	-0.48	?





# Plan

---

1- Analyse prédictive

**2- Classification**

3- Arbre de décision

- Construction de l'arbre
- Élagage de l'arbre

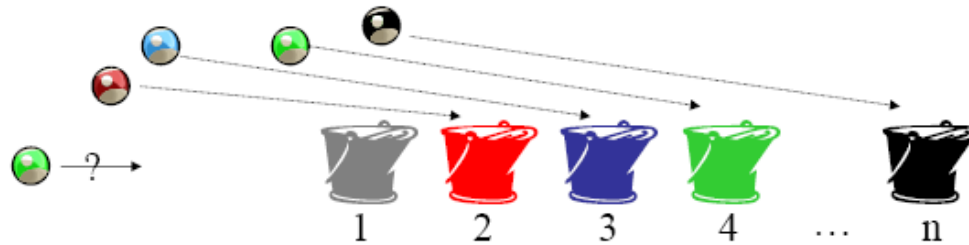


# Classification

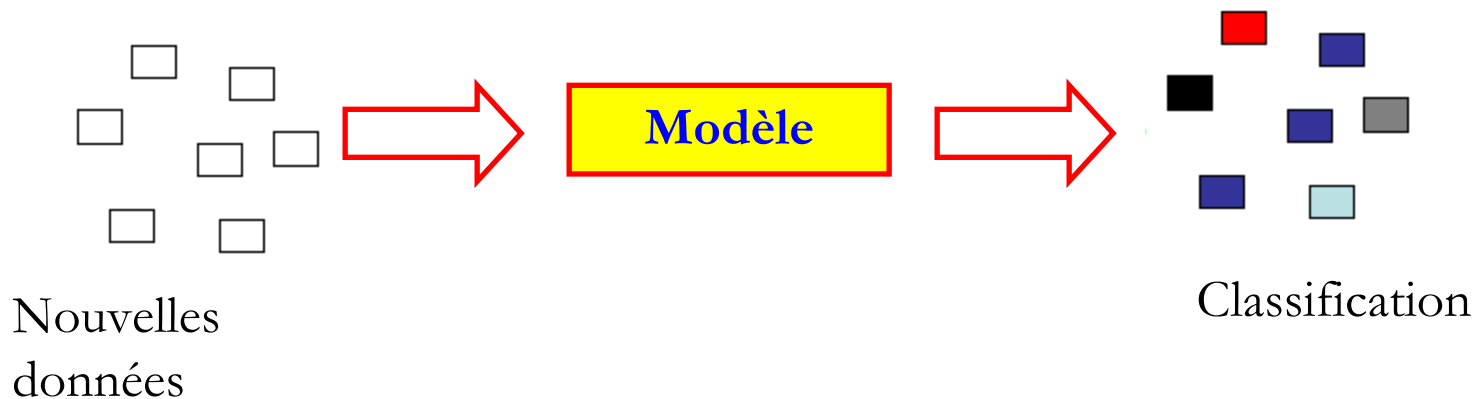
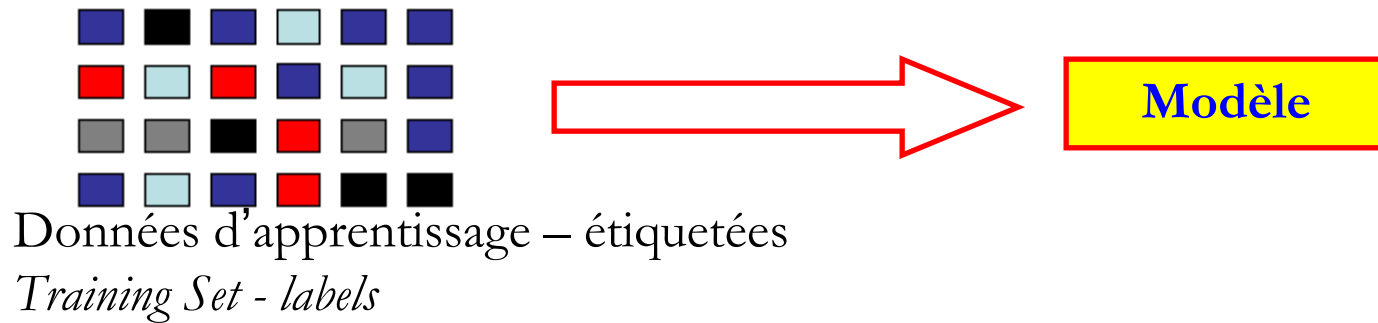
---

- ❑ En quelques termes, la classification:
  - Prédit la catégorie (la classe) d'un objet
  - Construis un modèle basé sur un jeu d'apprentissage et des labels (nom des catégories/classes) et l'utilise pour classer des données nouvelles.
  
- ❑ Applications :
  - Accord pour un crédit.
  - Marketing ciblé.
  - Diagnostic médical.
  - Prédiction de la faillite.

# Processus de classification



Classification = définition d'un modèle





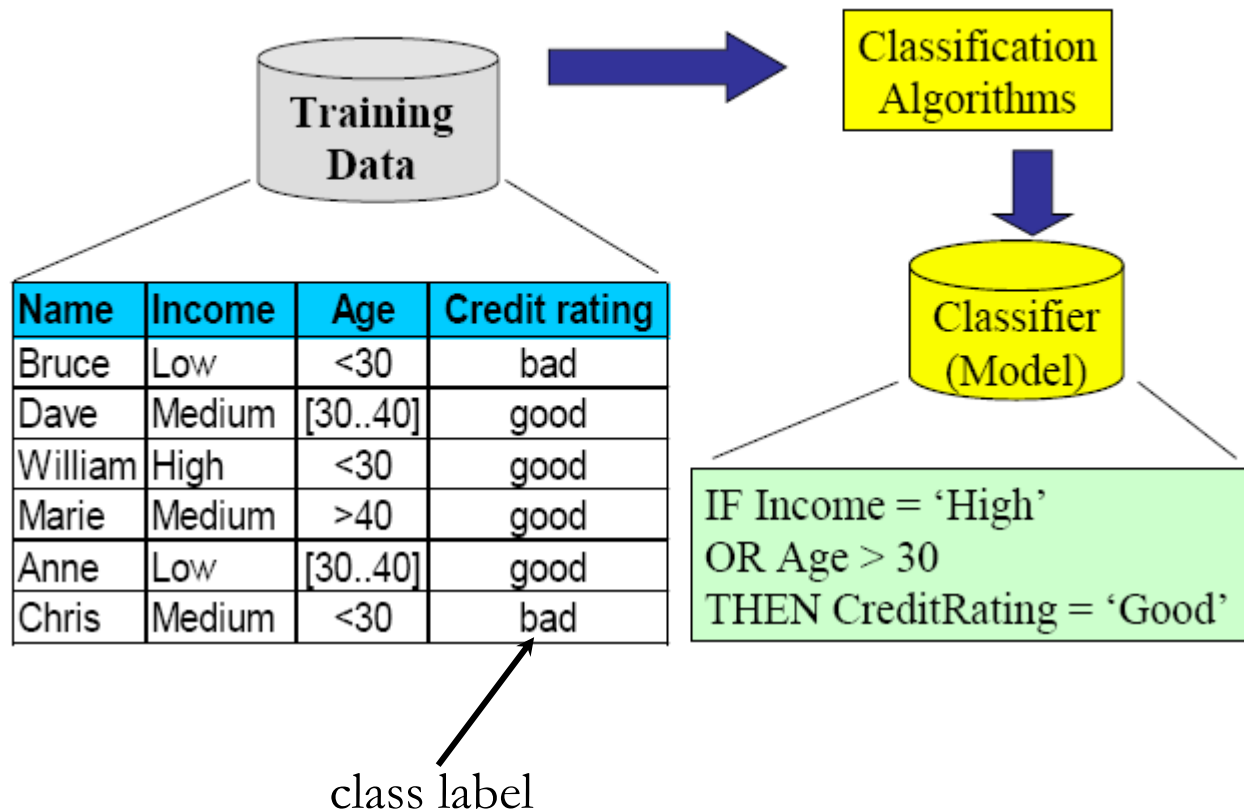
# Processus de classification – 3 étapes

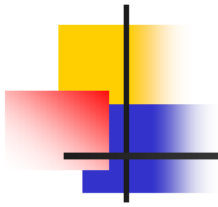
## 1 – Construction du modèle - Apprentissage

- **Jeu de données d'apprentissage (Training data)** : ensemble des objets utilisés pour la construction du modèle. Dans cet ensemble, chaque objet est censé appartenir à une classe prédéfinie selon l'attribut « **class label** »
- Le modèle est représenté selon l'une des formes suivantes:
  - ✓ Règles de classification (SI condition(s) ALORS action);
  - ✓ Arbres de décisions;
  - ✓ Réseaux de neurones;
  - ✓ Machines à vecteurs de support (*support vector machines SVM*);
  - ✓ Classificateur Bayésien.
  - ✓ Formules mathématiques.
  - ✓ ...

# Processus de classification

## 1 – Construction du modèle : Schéma





# Processus de classification

---

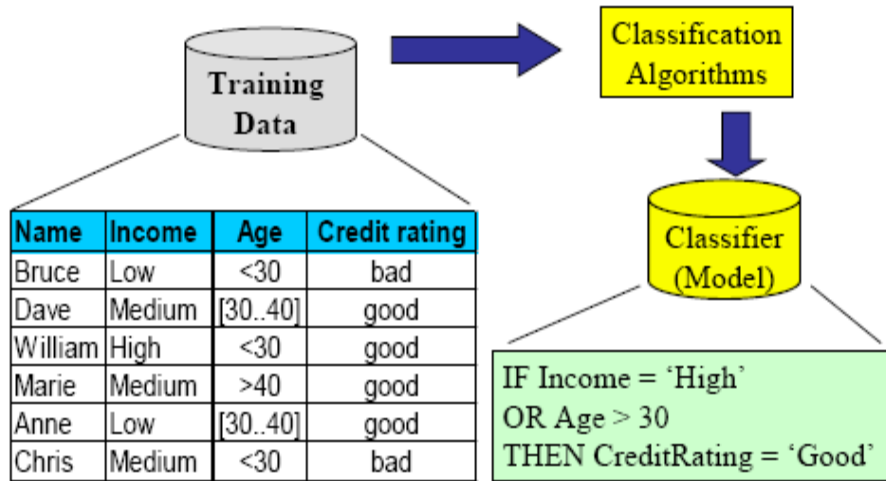
## 2 – Évaluation du modèle

- Estimation de la précision de la classification du modèle à l'aide des **données de test**.
- On doit définir un ensemble de données de test (les données de test doivent être différentes des données d'apprentissage);
  - ✓ Avec les données de tests, on connaît l'appartenance d'un objet à une classe spécifique;
- On va utiliser les données de test pour évaluer la performance du classificateur.  
Contrairement aux données d'apprentissage, on doit fournir au classificateur les données de test sans les labels des classes. Le classificateur va ainsi procéder à la classification.
- Pour valider les résultats, on doit maintenant utiliser les labels qui sont disponibles pour les données de tests pour calculer la précision de classification.

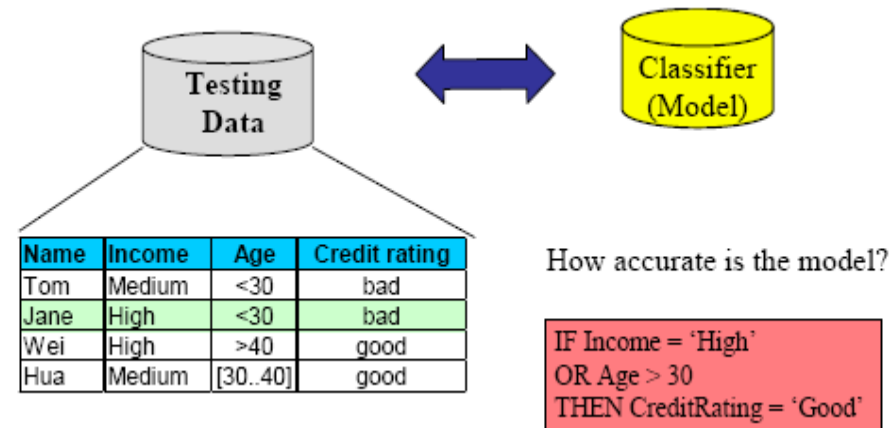
## 3 – Utilisation du modèle pour classer les nouveaux objets - Classification

# Processus de classification

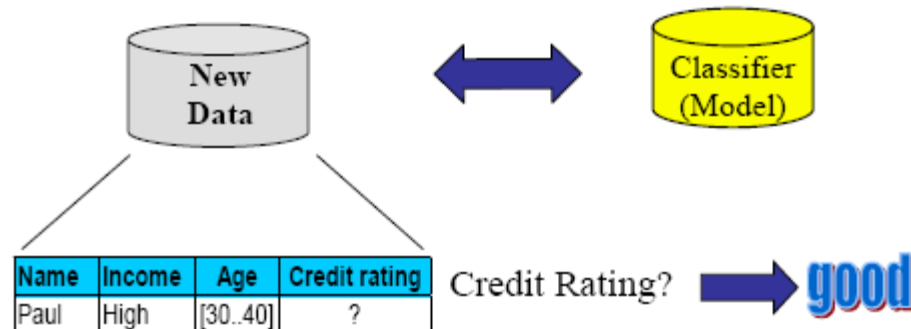
## 1- Apprentissage



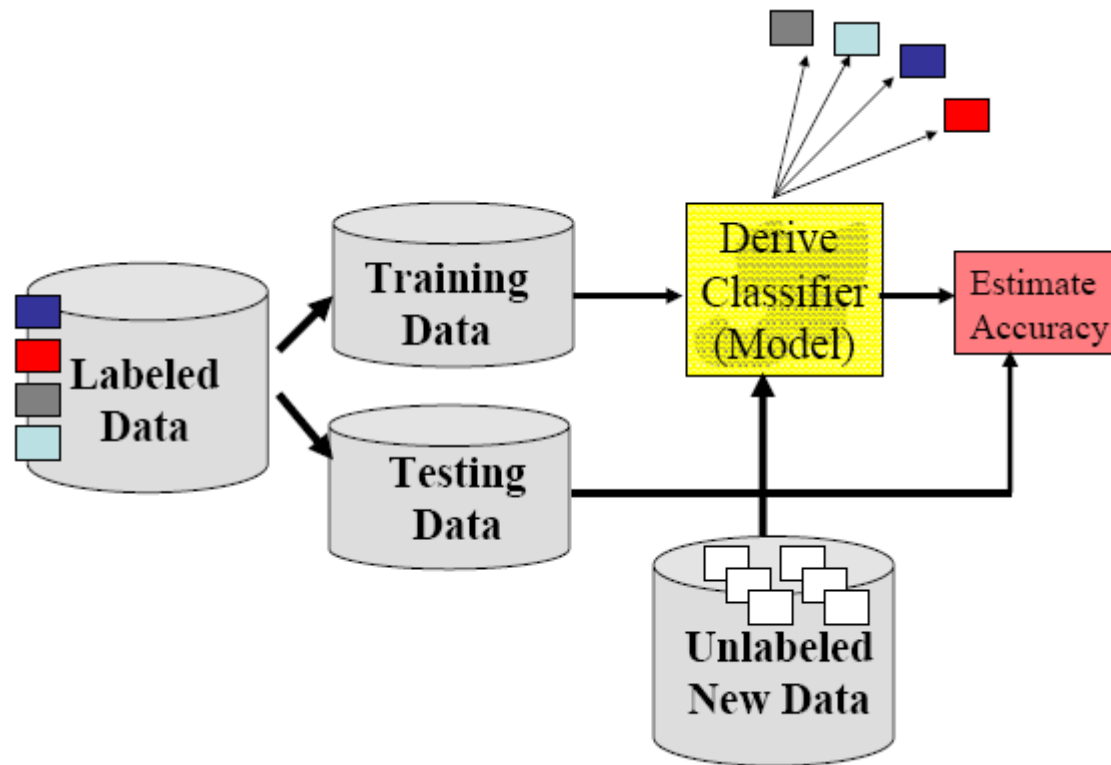
## 2- Évaluation



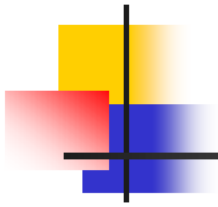
## 3- Classification



# Processus de classification







# Plan

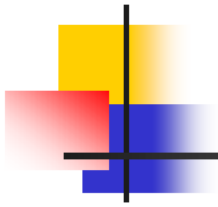
---

1- Analyse prédictive

2- Classification

**3- Arbre de décision**

- **Construction de l'arbre**
- Élagage de l'arbre



# Principe

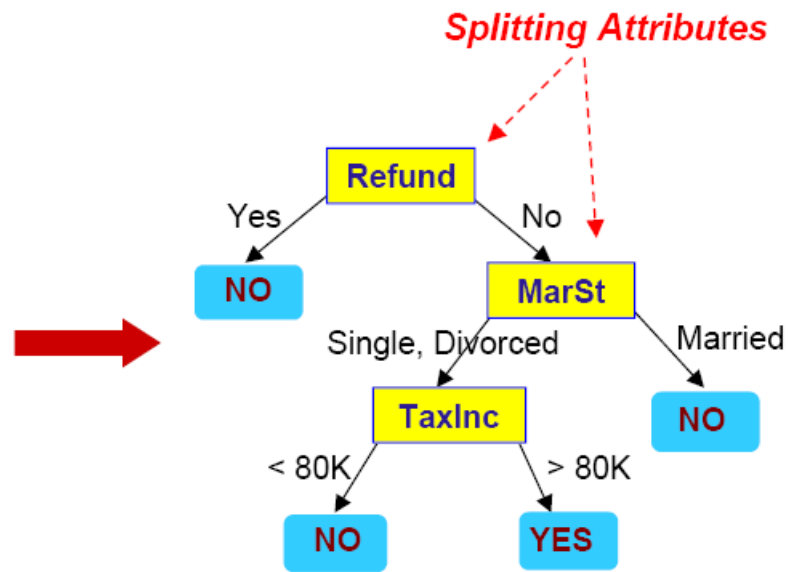
---

- La technique des arbres de décision est fondée sur l'idée simple de réaliser la classification d'un objet par une suite de test sur les attributs qui le décrivent.
- Ces tests sont organisés de telle façon que la réponse à un test indique à quel prochain test auquel on doit-on soumettre l'objet en question.

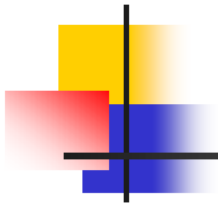
# Exemple

<i>Tid</i>	<i>Refund</i>	<i>Marital Status</i>	<i>Taxable Income</i>	<i>Cheat</i>
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

Données  
d'apprentissage



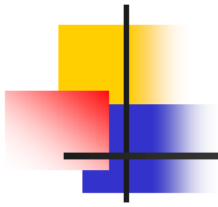
Modèle : Arbre de  
décision



# Arbre de décision

---

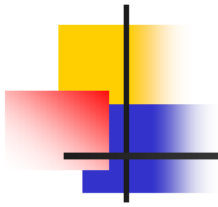
- L'exemple précédent illustre comment résoudre un problème de classification en posant une série de questions pertinentes sur les valeurs d'attributs de chaque enregistrement/objet/tuple à tester.
- À chaque fois on reçoit une réponse à une question, d'autres questions seront posées jusqu'à l'obtention d'une décision relative à l'appartenance d'un objet à une classe spécifique.
- Les séries des questions posées ainsi que les réponses peuvent être organisées sous la forme d'un arbre.



# Arbre de décision

---

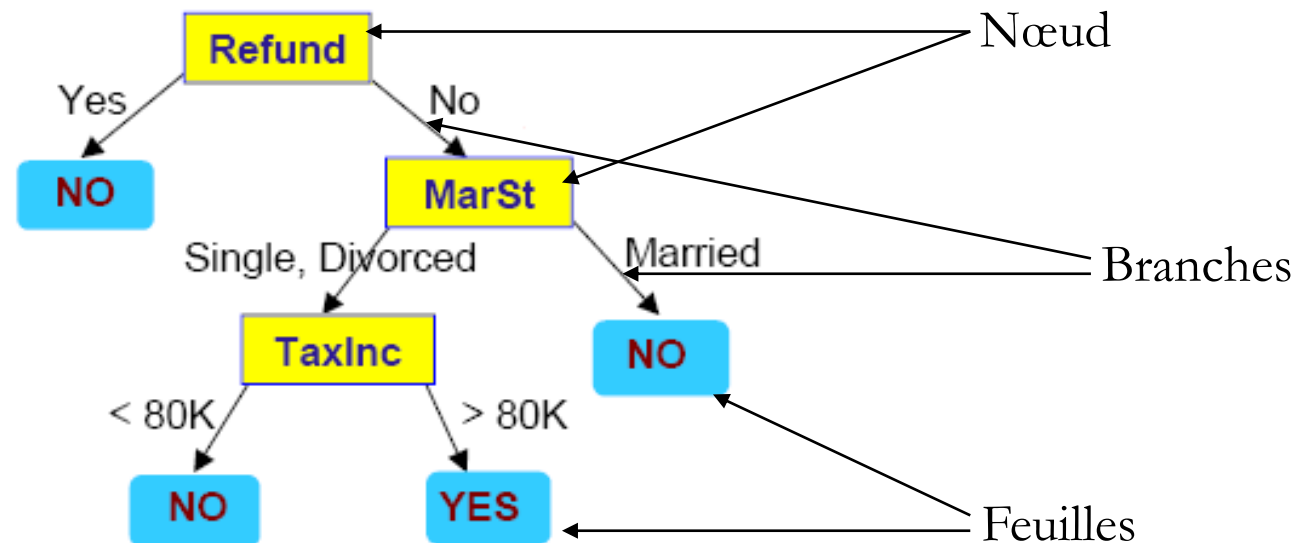
- Un arbre de décision est une structure de données utilisée comme modèle pour la classification.
- Elle est construite à l'aide d'une méthode récursive basée sur la stratégie divisée pour régner afin de créer des sous-groupes purs (un sous-groupe est pur lorsque tous les éléments du sous-groupe appartiennent à la même classe).
- L'objectif est de déterminer l'appartenance d'un objet à une classe en fonction de ses caractéristiques/attributs.
- Les attributs apparaissant dans l'arbre sont les attributs **pertinents** pour le problème considéré



# Arbre de décision

Un arbre de décision est composé de

- Nœud = test sur un attribut
- Branches = correspond à une valeur d'attribut
- Feuilles = désignent la classe de l'objet à classer





# Exemple

Prédire si un client sera un client qui rembourse son prêt (classe OUI) ou un client qui va avoir des difficultés de remboursement (classe NON)

Code	Label	Type	Inc. ?	Étendue (Code)
Raison	Raison	Symbolique	Non	TV, Autre, Courrier
Statut marital	Statut marital	Symbolique	Non	Marié, Célibataire, Divorcé, Séparé, Veuf
Titre	Titre	Symbolique	Non	Mme, Mlle, M.
Titre CoEmp	Titre CoEmp	Symbolique	Non	M., Aucun, Mme, Mlle
Garantie	Garantie	Symbolique	Non	Oui, Non
Assurance	Assurance	Symbolique	Non	Oui, Non
Logement	Logement	Symbolique	Non	Propriétaire, Locataire, Famille
Type logement	Type logement	Symbolique	Non	Appartement, Maison
Profession	Profession	Symbolique	Non	Employé, Ouvrier, Représentant, Enseignant
Enfants	Enfants	Entier	Non	[0, 6]
Ancienneté bancaire	Ancienneté bancaire	Entier	Non	[0, 60]
Salaire	Salaire	Entier	Non	[0, 59870]
Salaire CoEmp	Salaire CoEmp	Entier	Non	[0, 62078]
Nb Pers	Nb Pers	Entier	Non	[1, 8]
Revenu / tête	Revenu / tête	Entier	Non	[0, 59870]
Loyer	Loyer	Entier	Non	[0, 12800]
Âge	Âge	Entier	Non	[20, 87]
Emprunts courants	Emprunts courants	Entier	Non	[0, 6014]
Durée emprunt	Durée emprunt	Entier	Non	[1, 4]
Remboursement	Remboursement	Symbolique	Non	Lent, Normal, Rapide
Succès	Succès	Symbolique	Non	N, O



# Exemple

## Base de données

ALICE - [pretv6.alp - Enregistrements]

Fichier Edition Affichage Champ Arbre Outils Fenêtre ?

Details... Tous les enregistrements

	Nb Pers	Revenu / tête	Loyer	Âge	Emprunts courants	Durée emprunt	Remboursement	Succès
	1	14976	1628	33	0	4	Lent	N
	4	2300	2720	40	0	4	Lent	N
	3	6281	1760	36	1829	4	Lent	N
	2	6525	3200	45	0	4	Lent	N
	2	3646	1162	33	0	2	Normal	N
	2	6996	2080	31	1240	4	Lent	N
	2	3145	1296	33	0	2	Normal	N
	2	3279	1215	36	0	2	Normal	N
	5	2514	1363	41	0	4	Lent	N
	2	5666	1423	72	790	4	Lent	N
	4	2525	1600	39	930	4	Lent	N
	4	6150	960	41	1302	1	Rapide	N
	4	0	0	40	915	4	Lent	0
	2	8006	0	25	0	4	Lent	0
	2	5123	0	32	0	2	Normal	0
	2	5280	0	63	1853	4	Lent	0
	2	8659	1600	61	1240	3	Lent	0
	3	11620	1280	46	0	4	Lent	0

Pour l'aide, taper F1

NUM



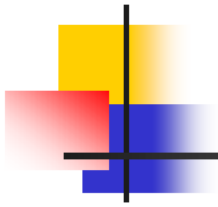


# Exemple

On veut construire l'arbre de décision pour une partie de la base de données « banque »

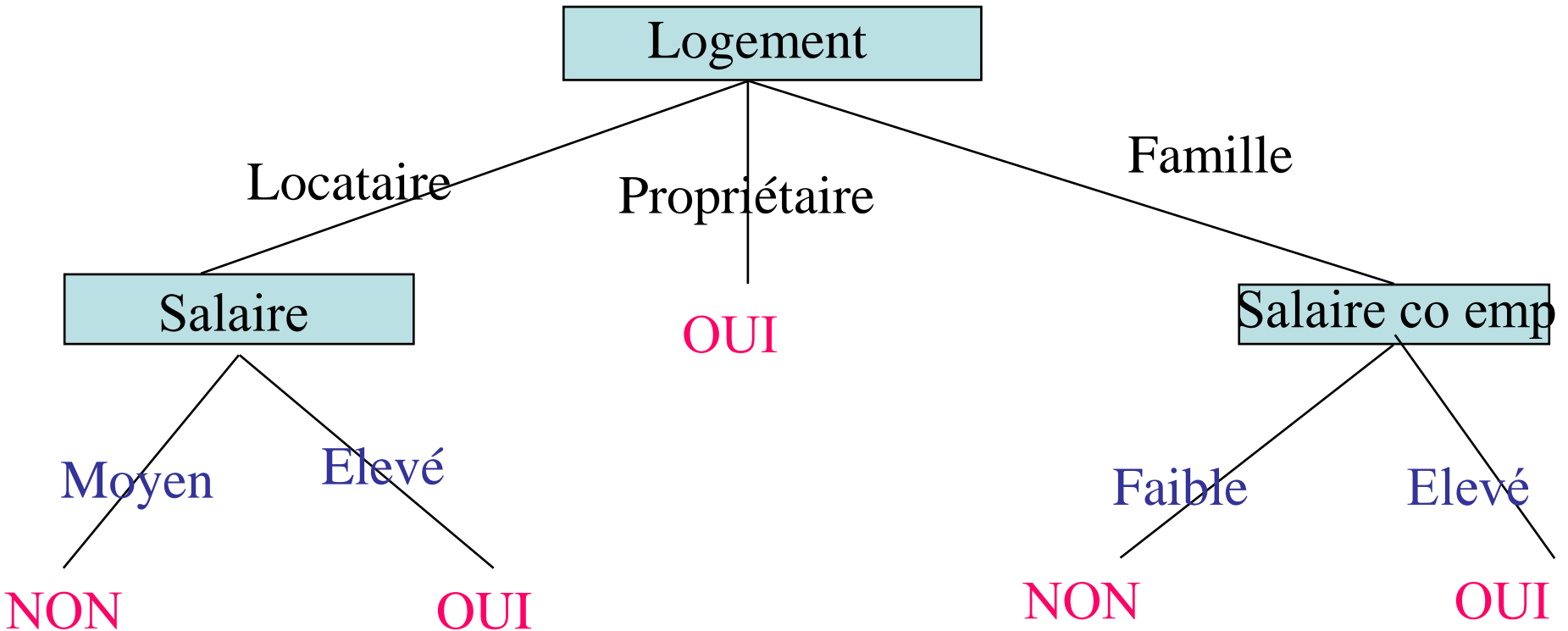
Soit un ensemble de cas E1-E14 décrivant des clients et leur classe OUI/NON

Client	Logement	salaire	Salaire	S. co-emp	<b>Succès</b>
E1	locataire	A	Moyen	Elevé	NON
E2	locataire	A	Moyen	Faible	NON
E3	propriétaire	A	Moyen	Elevé	OUI
E4	famille	B	Moyen	Elevé	OUI
E5	famille	C	Elevé	Elevé	OUI
E6	famille	C	Elevé	Faible	NON
E7	propriétaire	C	Elevé	Faible	OUI
E8	locataire	B	Moyen	Elevé	NON
E9	locataire	C	Elevé	Elevé	OUI
E10	famille	B	Elevé	Elevé	OUI
E11	locataire	B	Elevé	Faible	OUI
E12	propriétaire	B	Moyen	Faible	OUI
E13	propriétaire	A	Elevé	Elevé	OUI
E14	famille	B	Moyen	Faible	NON



# Exemple

Arbre de décision



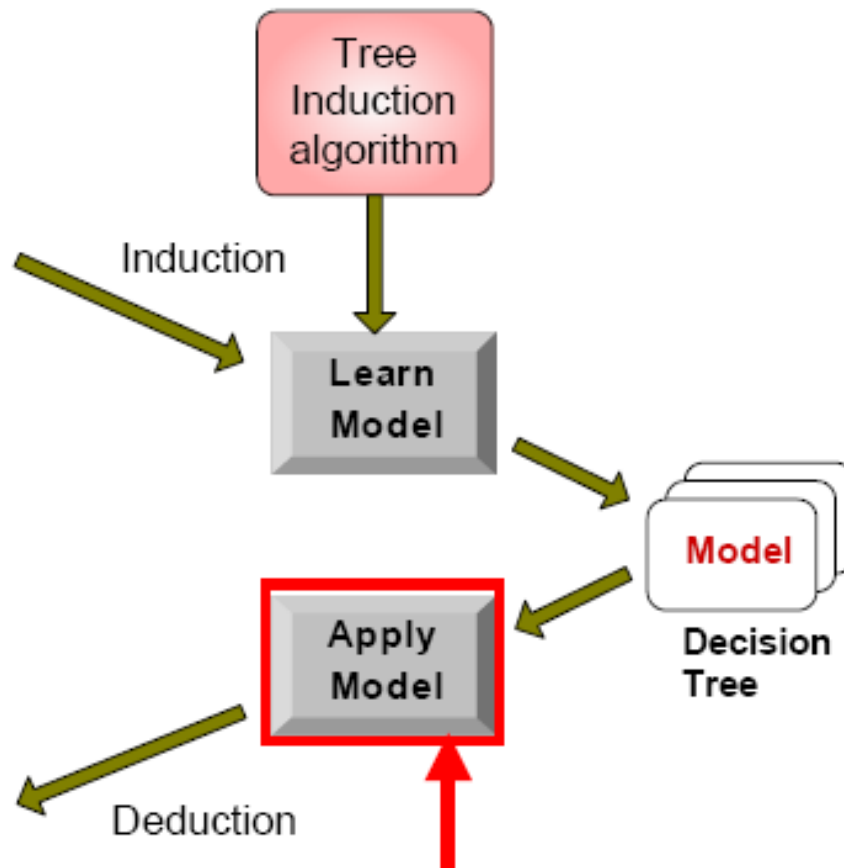
# Classification avec les arbres de décisions

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set

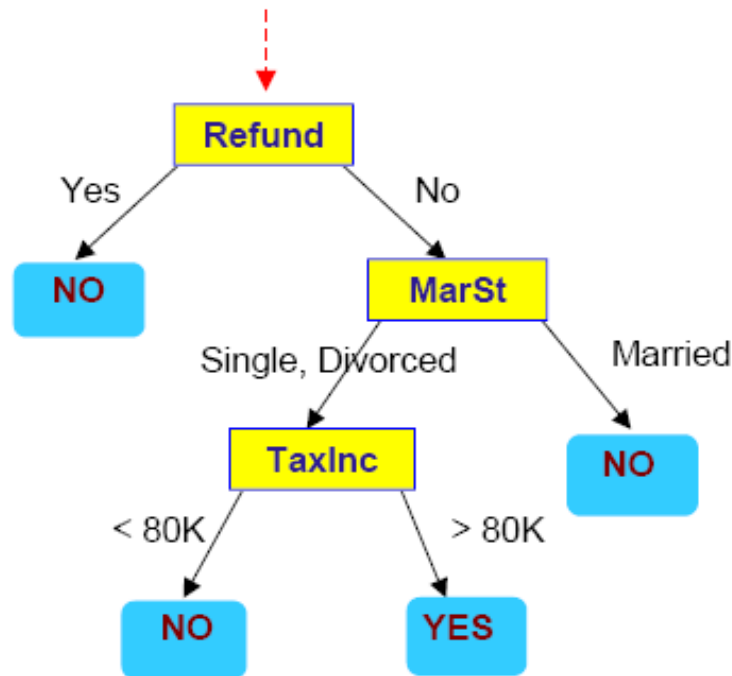


# Application du modèle

Données  
d'apprentissage

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

On commence à  
partir de la racine



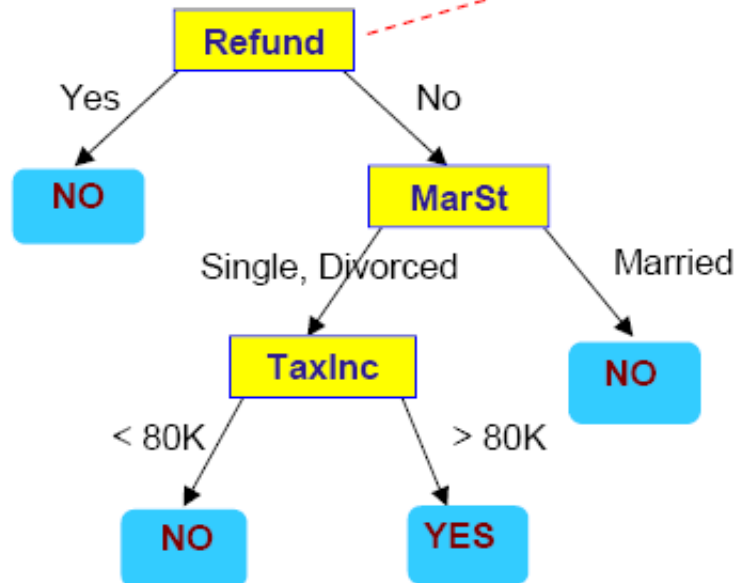
Donnée de test

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

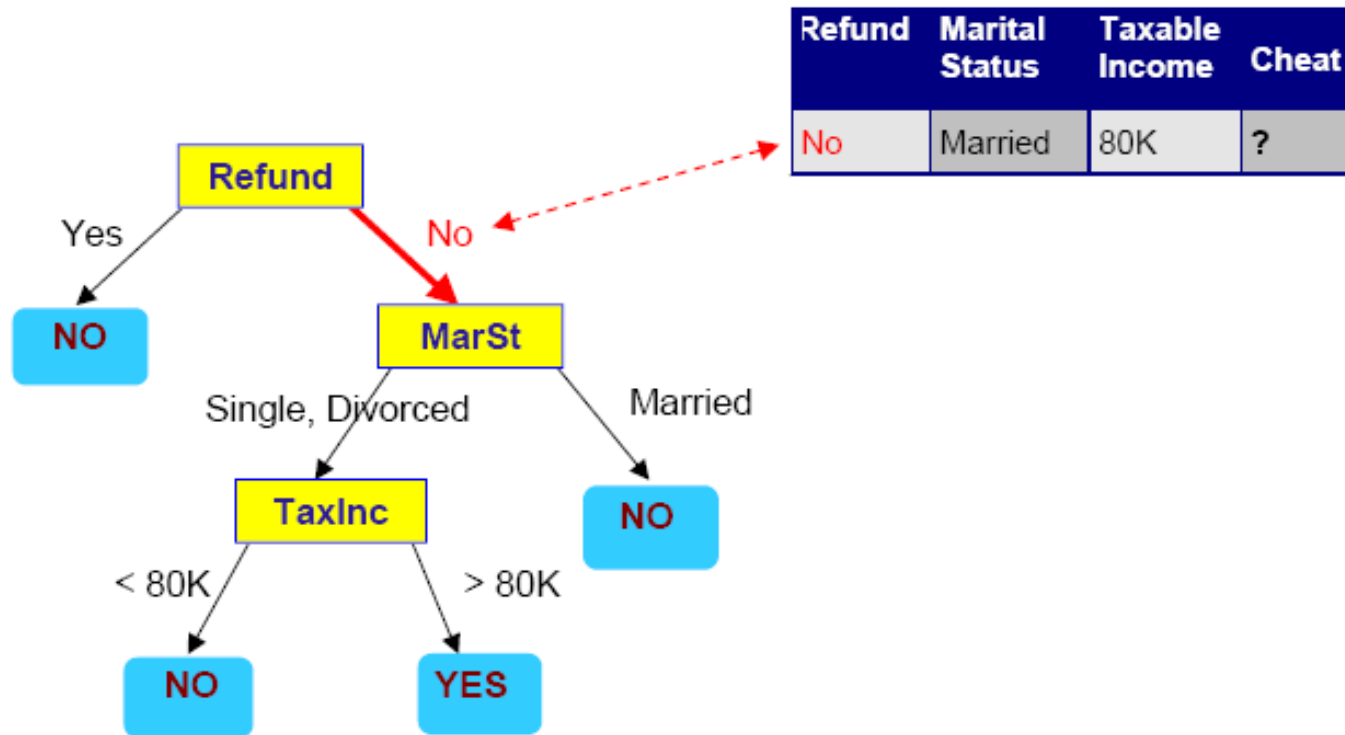
# Application du modèle

Donnée de test

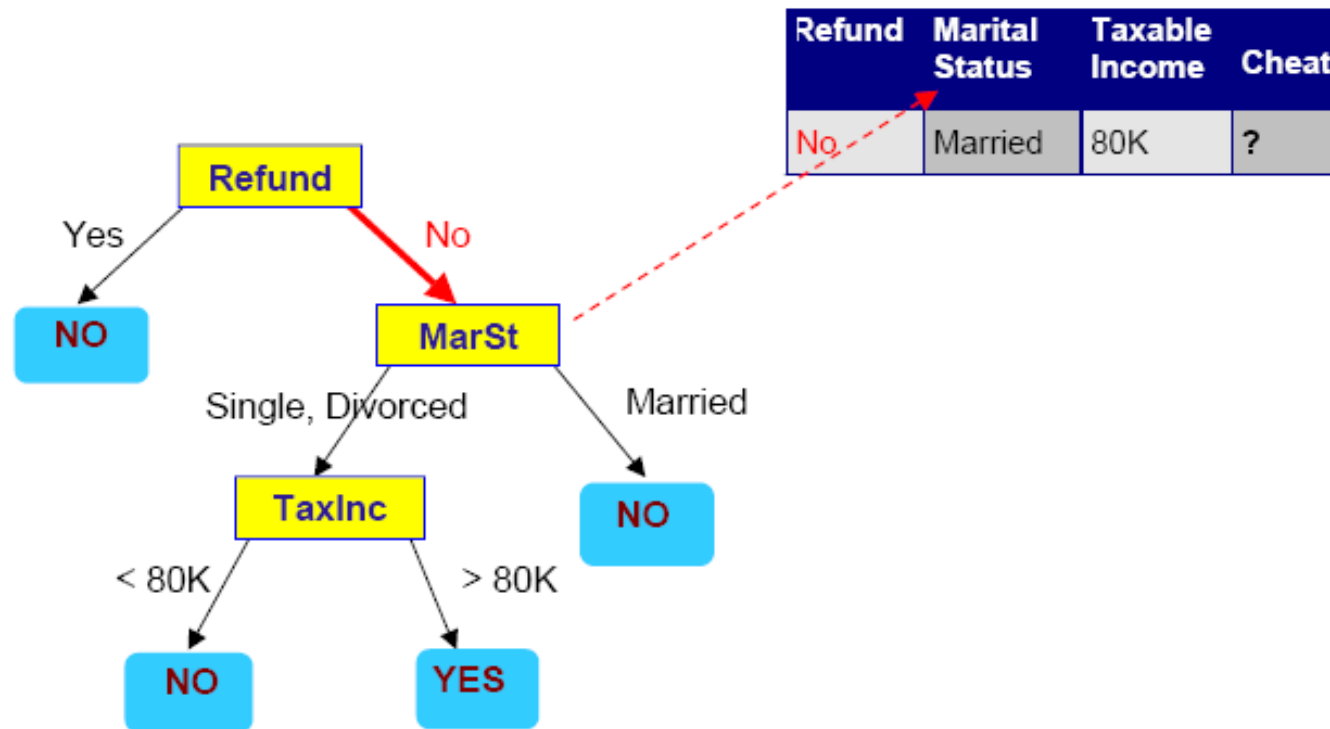
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



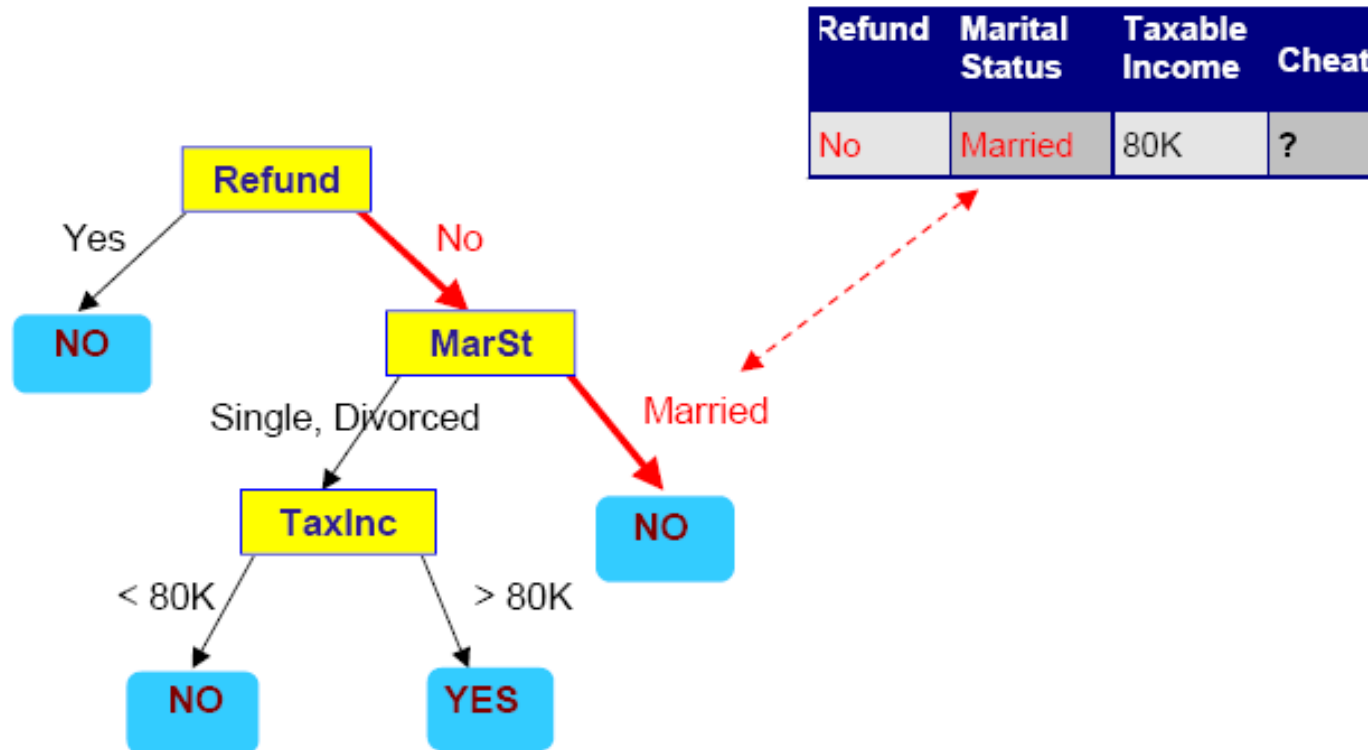
# Application du modèle



# Application du modèle

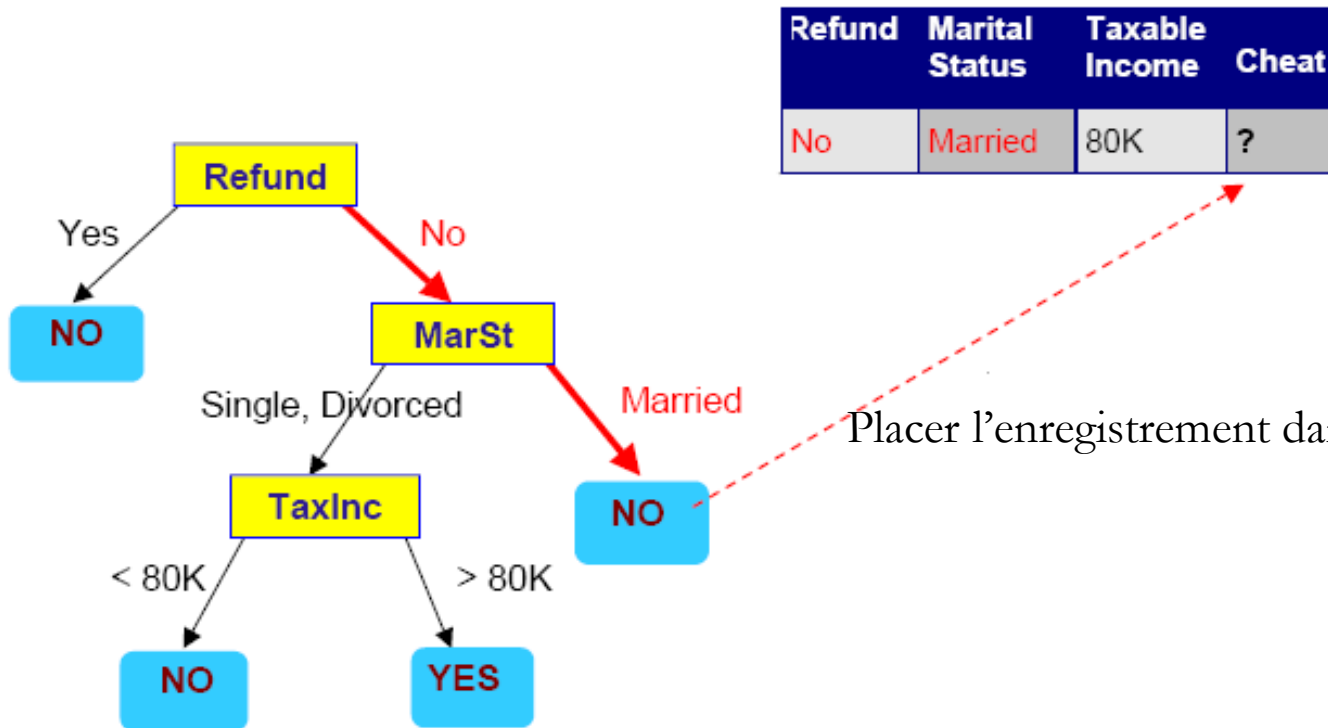


# Application du modèle





# Application du modèle



Placer l'enregistrement dans la classe « No »



# Construction d'arbre de décision

---

- La construction d'un arbre de décision se base généralement sur l'algorithme de **Hunt** qui représente à son tour la base de plusieurs algorithmes comme ID3, C4.5 et CART.
- On dispose donc d'un ensemble d'apprentissage  $S$  de  $n$  exemples  $(x, y)$  décrits par  $d$  attributs et une classe  $y = \{y_1, y_2, \dots, y_c\}$

Algorithme de Hunt → approche récursive qui adopte une stratégie « top-down » pour construire l'arbre



# Algorithme de Hunt

- 1 - L'arbre commence avec un nœud qui représente tout l'ensemble  $S$
- 2 - **Si** toutes les enregistrements de  $S$  appartiennent à la même classe  
**alors** le nœud devient une feuille étiqueté par le label de la classe.
- 3- **Sinon**
  - 3.1- Choisir le **meilleur** attribut de décision pour le nœud courant
  - 3.2- Placer cet attribut dans l'arbre
  - 3.3- Pour chaque valeur de l'attribut, créer une nouvelle branche de l'arbre
  - 3.4 Partitionnement des objets dans ces branches : segmentation**Si** chaque feuille satisfait **un critère d'arrêt**  
**alors** Fin  
**Sinon** Itérer sur les feuilles

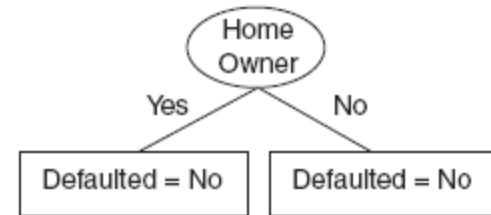
- Exemple de critères d'arrêt :
  - Les objets dans le nœud appartiennent à la même classe ;
  - Il ne reste plus d'attribut sur lesquels on va faire le découpage.

# Construction de l'arbre - Exemple

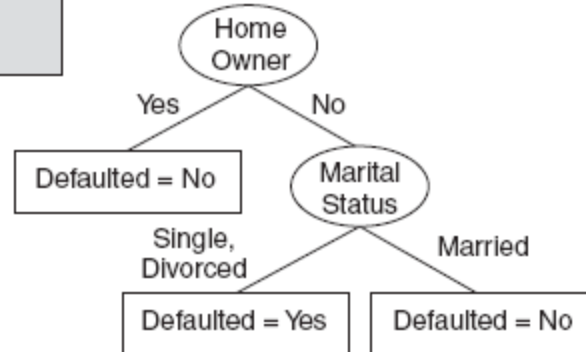
Tid	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Defaulted = No

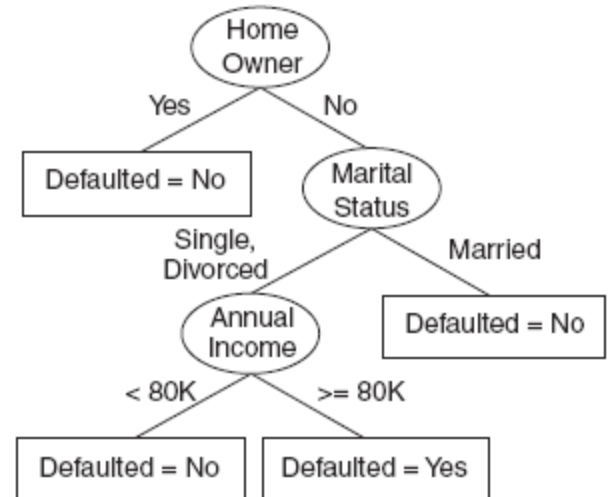
(a)



(b)



(c)



(d)

➤ Comment choisir le meilleur attribut pour précéder à la division ?



# Choix du meilleur attribut

---

- Il y a plusieurs mesures qui sont utilisées pour identifier le meilleur attribut pour la division. Dans ce cours nous nous focalisons sur deux mesures :
  1. **Gain d'information** : utilisé par les algorithmes ID3 et C4.5
  2. **Indice de Gini** : utilisé par l'algorithme CART



# Gain d'information (Information Gain)

Pour mesurer le Gain d'information, on doit estimer  
premièrement **l'entropie**

$$Entropie(S) = - \sum_{i=1}^c p_i \log_c(p_i)$$

- $S$  : l'ensemble des données d'apprentissage
- $C$  : le nombre de classes
- $P_i$  : le nombre d'objets dans la classe  $i$  divisé par le nombre total d'objets dans  $S$ .



# Entropie\*

- Le concept d'entropie est apparu en thermodynamique, pour mesurer le désordre moléculaire : l'entropie est proche de zéro lorsque les molécules sont immobiles et bien ordonnées.
- Cette notion s'est ensuite étendue à un grand nombre de domaines, parmi lesquels celui de la théorie de l'information de Shannon, où elle mesure l'information moyenne contenue dans un message : l'entropie est nulle lorsque tous les messages sont identiques.
- En apprentissage automatique, on utilise fréquemment l'entropie comme mesure d'impureté : l'entropie d'un jeu de données est nulle lorsque toutes ses observations appartiennent à une seule classe.
- Une réduction de l'entropie est souvent appelée gain d'information.

\*Source : Aurélien Geron. Machine Learning avec Scikit-Learn : Mise en œuvre et cas concrets. Dunod, 2019.



# Exemple de calcul de l'entropie

Données d'apprentissage

Tid	Home Owner	Marital Status	Annual Income	Defaulted Borrower
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

$$Entropie(S) = - \left[ \left( \frac{7}{10} \times \log_2 \left( \frac{7}{10} \right) \right) + \left( \frac{3}{10} \times \log_2 \left( \frac{3}{10} \right) \right) \right]$$

$$Entropie(S) = 0.88$$





# Entropie

---

- Il convient de noter que la valeur de l'entropie est 0 si tous les objets de  $S$  appartiennent à la même classe.
  - Par exemple, pour un problème de deux classes,  
$$\text{Entropie}(S) = - [ (1 * \log(1)) + (0 * \underline{\log(0)}) ] = 0.$$
- Lorsque l'entropie est égale à 1 cela signifie que les objets sont répartis équitablement dans les deux classes.
- Une valeur de l'entropie entre 0 et 1 indique que le nombre d'objets dans les deux classes n'est pas le même.

# Calcul du gain d'information pour un attribut spécifique $A$

$$Gain(S, A) = Entropie(S) - \sum_{v \in valeur(A)} \frac{|S_v|}{|S|} \times Entropie(S_v)$$

- $valeur(A)$  : l'ensemble des valeurs possibles pour l'attribut  $A$ .
- $S_v$  un sous-ensemble de  $S$  qui contient la valeur  $v$  de l'attribut  $A$ .
- $|S_v|$  : cardinalité de  $S_v$ .
- $|S|$  : cardinalité de  $S$ .

➤ On sélectionne l'attribut avec la plus grande valeur de Gain d'information



## Exemple

Construire l'arbre de décision de l'ensemble d'apprentissage suivant en utilisant le **gain d'information**

Outlook	Tempreature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N



## Exemple (suite)

Gain(Outlook) = ?

Gain(Temperature) = ?

Gain(Humidity) = ?

Gain(Windy) = ?

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

1- Calcule de l'entropie de tout l'ensemble de données :

$$Entropie(S) = - \left[ \left( \frac{5}{14} \times \log_2 \left( \frac{5}{14} \right) \right) + \left( \frac{9}{14} \times \log_2 \left( \frac{9}{14} \right) \right) \right] = 0.94$$

2- Calcule de gain d'information pour chaque attribut

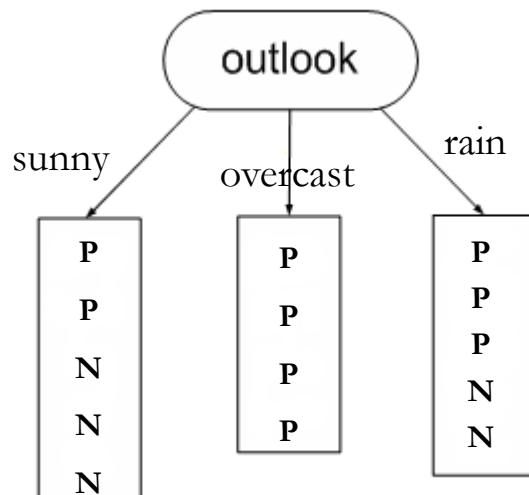
→ On commence par l'attribut « Outlook »

# Exemple (suite)

## ■ Calcul de : Gain(Outlook)

On à: 
$$Gain(S, Outlook) = Entropie(S) - \sum_{v \in valeur(Outlook)} \frac{|S_v|}{|S|} \times Entropie(S_v)$$

■  $valeur(Outlook) = \{ sunny, overcast, rain \}$



Outlook	Tempreature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

## Exemple (suite)

$$Gain(S, Outlook) = Entropie(S) - \sum_{v \in \text{valeur}(Outlook)} \frac{|S_v|}{|S|} \times Entropie(S_v)$$

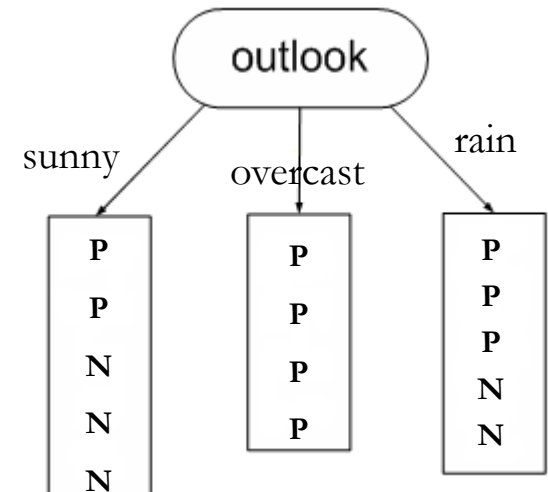
Maintenant on calcule cette partie pour chaque valeur  $v$

Pour  $v = \text{sunny}$

$$Entropie(S_{\text{sunny}}) = - \left[ \left( \frac{2}{5} \times \log_2 \left( \frac{2}{5} \right) \right) + \left( \frac{3}{5} \times \log_2 \left( \frac{3}{5} \right) \right) \right] = 0.97$$

$$|S_{\text{sunny}}| = 5 \text{ et } |S| = 14$$

$$\frac{5}{14} \times 0.97 = 0.347$$



## Exemple (suite)

Pour  $v = \text{overcast}$

$$\text{Entropie}(S_{\text{overcast}}) = - \left[ \left( \frac{4}{4} \times \log_2 \left( \frac{4}{4} \right) \right) + \left( \frac{0}{4} \times \log_2 \left( \frac{4}{4} \right) \right) \right] = 0$$

$$|S_{\text{overcast}}| = 4 \text{ et } |S| = 14_{S_{\{\text{rain}\}}}$$

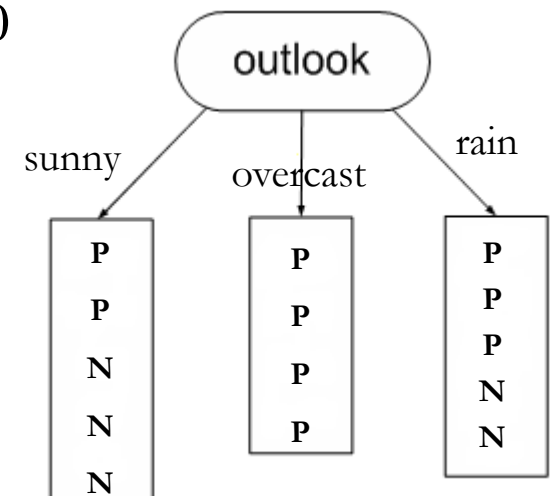
$$\frac{4}{14} \times 0 = 0$$

Pour  $v = \text{rain}$

$$\text{Entropie}(S_{\text{rain}}) = - \left[ \left( \frac{2}{5} \log_2 \left( \frac{2}{5} \right) + \frac{3}{5} \log_2 \left( \frac{3}{5} \right) \right) \right] = 0.97$$

$$|S_{\text{rain}}| = 5 \text{ et } |S| = 14$$

$$\frac{5}{14} \times 0.97 = 0.347$$





## Exemple (suite)

### ■ Calcul du Gain d'information

$$Gain(S, A) = Entropie(S) - \sum_{v \in valeur(A)} \frac{|S_v|}{|S|} \times Entropie(S_v)$$

$$Gain(S, Outlook) = 0.94 - (0.347 + 0 + 0.347) = \underline{0.246}$$

De même:

$$Gain(S, temperature) = 0.029$$

$$Gain(S, humidity) = 0.152$$

$$Gain(S, windy) = 0.048$$

Le max



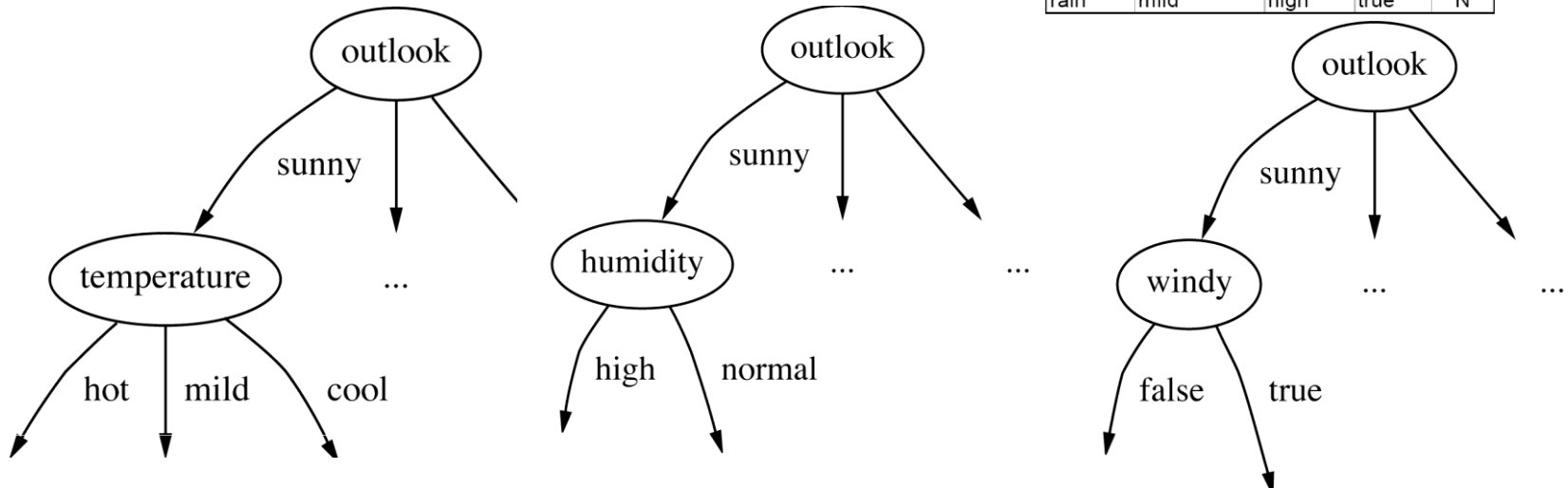
➤ On commence donc notre arbre avec l'attribut Outlook



## Exemple (suite)

### ■ Sélection du deuxième attribut

Outlook	Tempreature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N



On peut examiner:

Temperature, Humidity ou Windy pour « Outlook = sunny »

$\text{Gain}(S_{\langle \text{Outlook-sunny} \rangle}, \text{Température}) = 0.57$

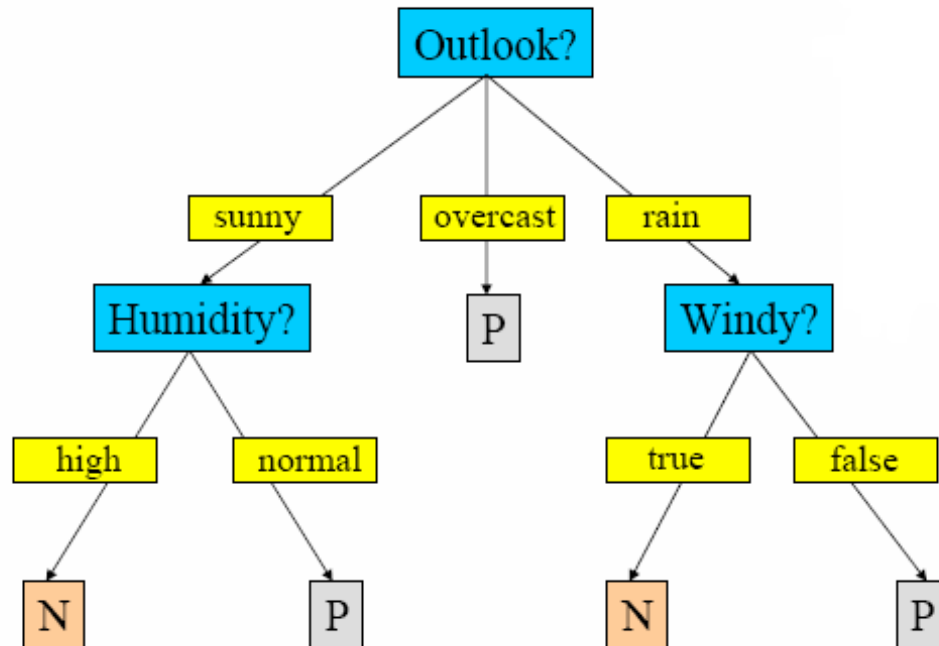
**$\text{Gain}(S_{\langle \text{outlook-sunny} \rangle}, \text{Humidity}) = 0.97$**

$\text{Gain}(S_{\langle \text{outlook-sunny} \rangle}, \text{Windy}) = 0.020$

Et on continue...

# Exemple (suite et fin)

## ■ Arbre identifié



Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N



## Indice de Gini (Gini Index)

---

- Pour un ensemble de données d'apprentissage  $S$  qui contient  $C$  classes, l'indice de Gini de l'ensemble  $S$  est défini comme suit :

$$Gini(S) = 1 - \sum_{j=1}^C P_j^2$$

$P_j$  est la proportion des objets qui appartiennent à la classe  $j$ .



# Indice de Gini

- L'indice de Gini d'un attribut  $A$  qui va être divisé en  $v$  branches (c.-à-d. l'ensemble de données d'apprentissage  $S$  est divisé en  $v$  sous ensembles  $\{S_1, S_2, \dots, S_v\}$  selon les valeurs de l'attribut  $A$ )

$$Gini(A) = \sum_{l=1}^v \frac{|n_{Av}|}{|n_A|} \times Gini(S_v)$$

- $n_{Av}$  le nombre d'objets associés à l'attribut  $A$  avec une la valeur  $v$
- $n_A$  le nombre d'objets associés à l'attribut  $A$

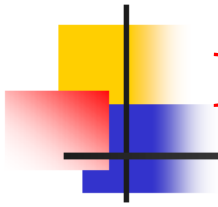
**→ On doit choisir l'attribut avec  
la plus petite valeur de l'indice de Gini**



## Exemple

Construire l'arbre de décision de l'ensemble d'apprentissage suivant en utilisant **l'indice de Gini**

Outlook	Tempreature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N



## Exemple (suite)

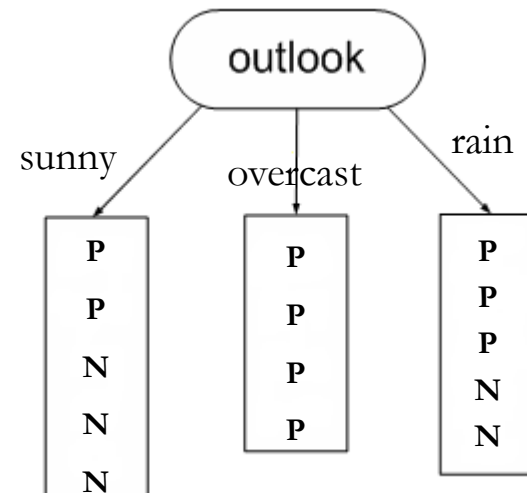
$$Gini(outlook) = \left( \frac{5}{14} \times Gini(sunny) \right) + \left( \frac{4}{14} \times Gini(overcast) \right) + \left( \frac{5}{14} \times Gini(rainy) \right)$$

$$Gini(sunny) = 1 - \left[ \left( \frac{2}{5} \right)^2 + \left( \frac{3}{5} \right)^2 \right];$$

$$Gini(overcast) = 1 - \left[ \left( \frac{4}{4} \right)^2 + \left( \frac{0}{4} \right)^2 \right];$$

$$Gini(sunny) = 1 - \left[ \left( \frac{3}{5} \right)^2 + \left( \frac{2}{5} \right)^2 \right]$$

$$Gini(outlook) = 0.34$$





## Exemple (suite et fin)

$$Gini(outlook) = 0.34$$

$$Gini(temperature) = 0.43$$

$$Gini(humidity) = 0.63$$

$$Gini(windy) = 0.42$$

- On doit choisir donc l'attribut *Outlook* comme racine de l'arbre
- **Exercice** : Compléter la construction l'arbre de décision de l'exemple « météo et match de foot » en utilisant l'indice de Gini.



# Gini ou Entropie?

---

Faut-il utiliser l'impureté Gini ou l'entropie ?

- À vrai dire, dans la plupart des cas cela ne fait pas une grande différence : on aboutit à des arbres similaires.
- L'impureté Gini est un peu plus rapide à calculer.
- Consulter l'analyse :  
<https://sebastianraschka.com/faq/docs/decision-tree-binary.html>



# Comment spécifier les conditions de test pour la création des branches?

❑ Cela dépend du type des attributs

- Attribut catégorique (nominal)
- Attribut numérique continu

❑ Dépend aussi de la façon de comment décomposer les attributs

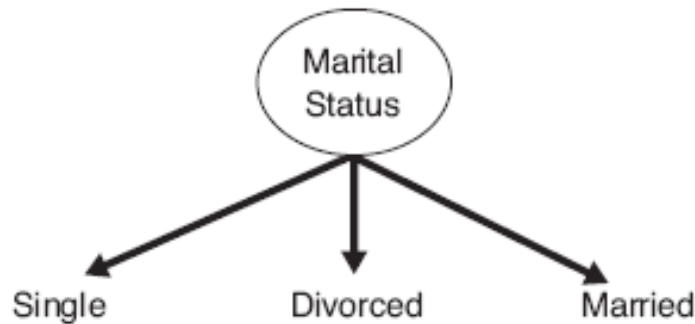
- Décomposition binaire
- Décomposition en plusieurs branches



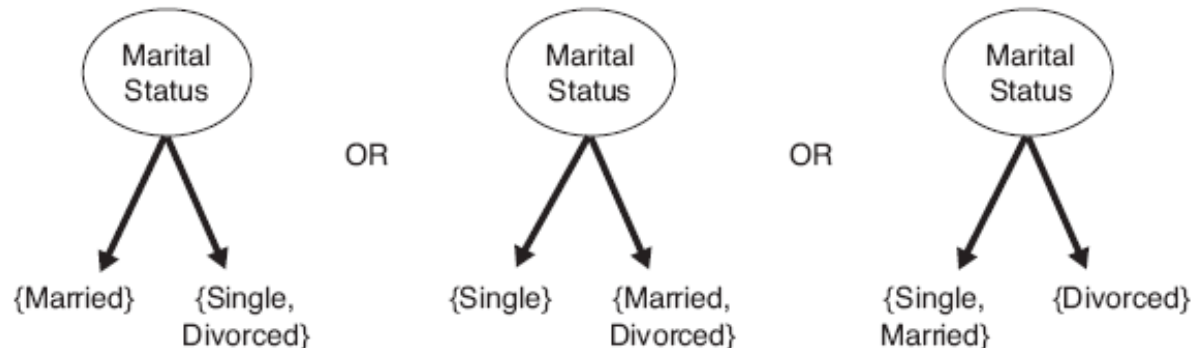
<i>Tid</i>	<b>Refund</b>	<b>Marital Status</b>	<b>Taxable Income</b>	<b>Cheat</b>
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

# Décomposition des attributs catégorique

❑ Décomposition en plusieurs branches selon toutes les valeurs de l'attribut → On crée une branche pour chaque valeur

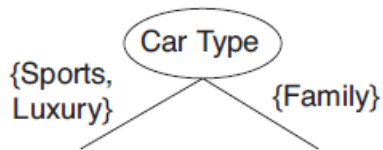


❑ Décomposition binaire → Regroupement des valeurs



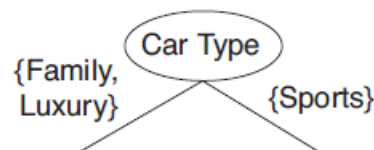
# Décomposition des attributs catégorique

## □ Exemple : Attribut avec valeurs catégorique

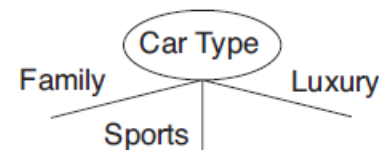


	Car Type	
	{Sports, Luxury}	{Family}
C0	9	1
C1	7	3
Gini	0.468	

(a) Binary split



	Car Type	
	{Sports}	{Family, Luxury}
C0	8	2
C1	0	10
Gini	0.167	



	Car Type		
	Family	Sports	Luxury
C0	1	8	1
C1	3	0	7
Gini	0.163		

(b) Multiway split



# Décomposition des attributs numérique

---

- Les attributs numériques sont transformés en catégorique. Ce processus est appelé **discrétisation**.
- Les valeurs des attributs sont divisées en intervalles
  - Les valeurs des attributs sont triées
  - Des séparations sont placées pour créer des intervalles / classes purs
- ➔ Nous choisissons l'intervalle qui optimise (maximise ou minimise) un critère spécifique (ex. indice de Gini)

# Décomposition des attributs numérique

La première valeur doit être  $<$  à la valeur min

La moyenne entre deux valeurs adjacentes

La dernière valeur doit être  $>$  à la valeur min

Cheat	No		No		No		Yes		Yes		Yes		No		No		No		No			
	Taxable Income																					
	60		70		75		85		90		95		100		120		125		220			
	55		65		72		80		87		92		97		110		122		172		230	
	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0	3	0
No	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
Gini	0.420		0.400		0.375		0.343		0.417		0.400		<u>0.300</u>		0.343		0.375		0.400		0.420	

➤ Choisir la position qui minimise l'indice de Gini

# Décomposition des attributs numérique

❑ La discrétisation peut aussi être faite en déterminant les valeurs d'attribut qui implique un changement de classe → la décomposition est faite entre les valeurs adjacentes avec deux classes différentes.

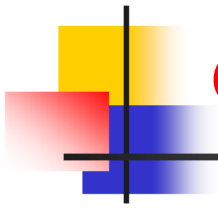
## ❑ Exemple

Class	No	No	No	Yes	Yes	Yes	No	No	No	No
Sorted Values →	60	70	75	85	90	95	100	120	125	220

Revenu < 80 → classe « No »

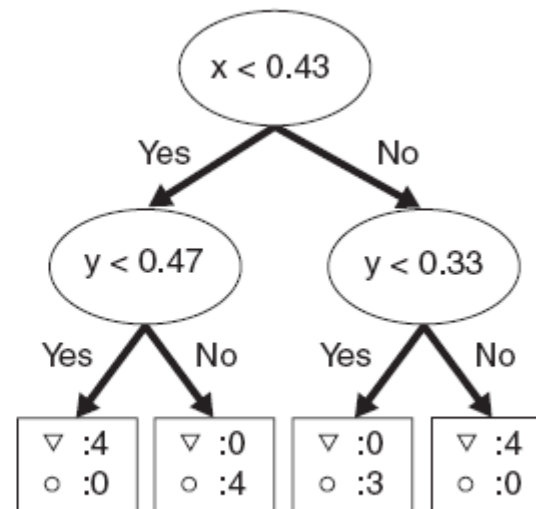
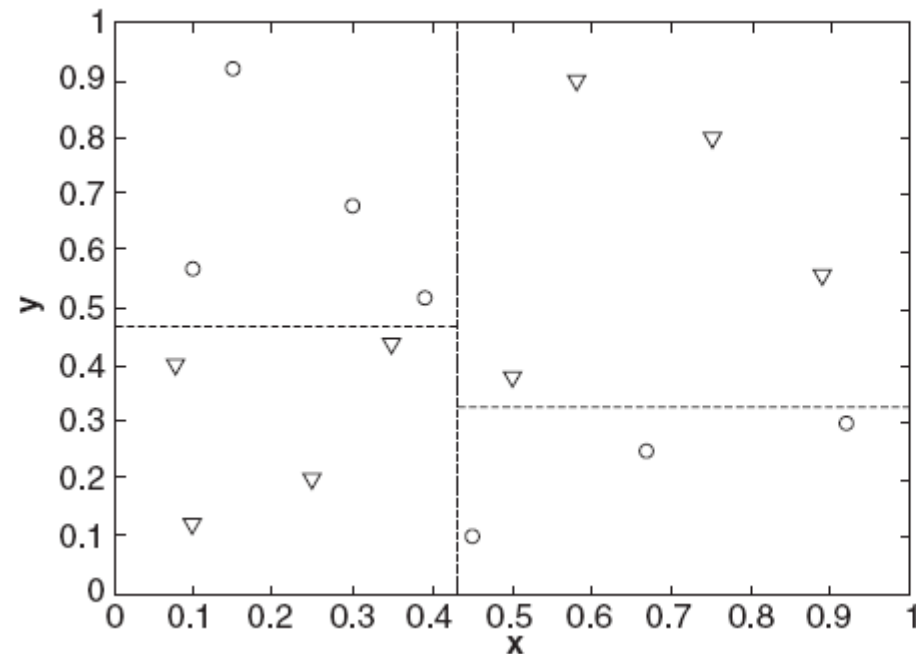
80 ≤ Revenu < 97 → classe « Yes »

Revenu ≥ 97 → classe « No »



# Classification des données numérique

## Exemple





# Overfitting (Surapprentissage)

- Un classifieur avec un taux d'erreur nul ou très faible n'est pas nécessairement un bon classifieur.
- Un bon classifieur est celui qui réussit à classer correctement les nouvelles données (bon pouvoir de généralisation).
- Sinon, un classifieur avec un taux d'erreur nul ou très faible, mais un faible pouvoir de généralisation souffre du problème de surapprentissage (overfitting).
- L'overfitting peut être due à :
  - la présence du bruit dans les données d'apprentissage;
  - ou un ensemble de données d'apprentissage très petit.





# Overfitting due à la présence du bruit

Soit l'ensemble des données d'apprentissage suivant qui représente un problème de classification en deux classes : mammifère désigné par « yes » et la classe non - mammifère désigné par « no ».

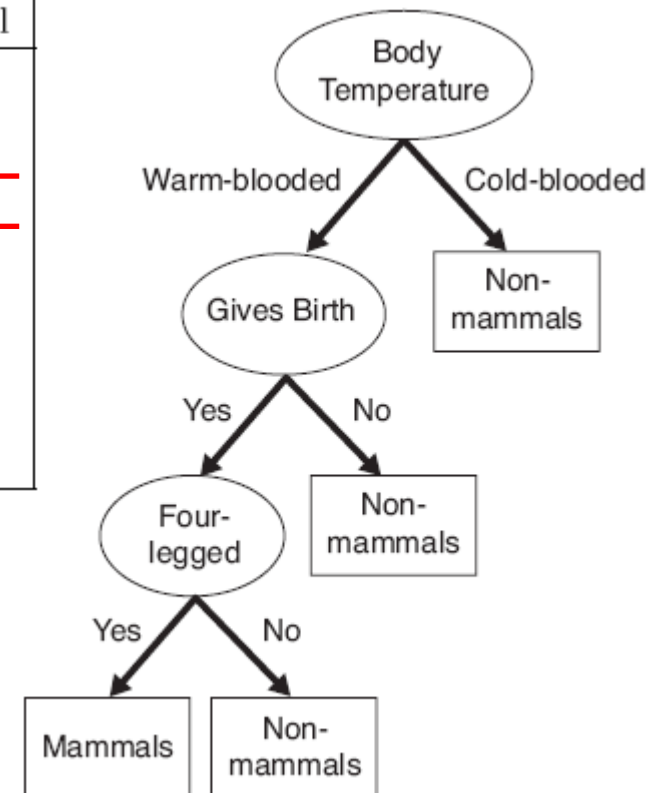
Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
porcupine	warm-blooded	yes	yes	yes	yes
cat	warm-blooded	yes	yes	no	yes
→ bat	warm-blooded	yes	no	yes	no*
→ whale	warm-blooded	yes	no	no	no*
salamander	cold-blooded	no	yes	yes	no
komodo dragon	cold-blooded	no	yes	no	no
python	cold-blooded	no	no	yes	no
salmon	cold-blooded	no	no	no	no
eagle	warm-blooded	no	no	no	no
guppy	cold-blooded	yes	no	no	no

Il existe deux enregistrements dont l'étiquetage est erroné (les deux animaux « bat » et « whale » sont considérés comme des animaux non mammifères).

# Overfitting due to the presence of noise

L'arbre de décision qui correspond à l'ensemble d'apprentissage précédent est :

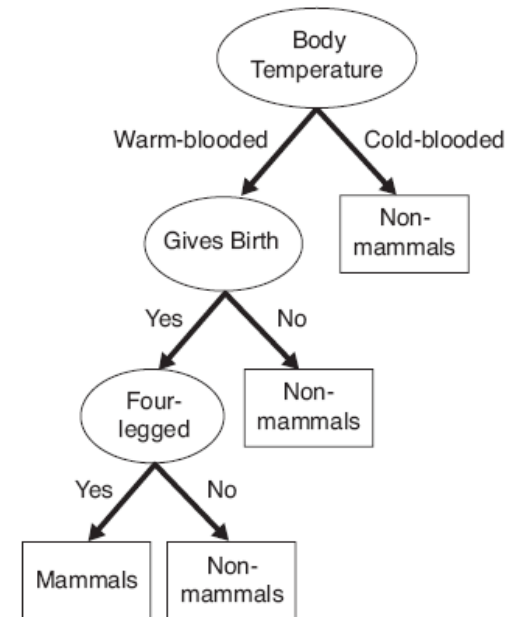
Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
porcupine	warm-blooded	yes	yes	yes	yes
cat	warm-blooded	yes	yes	no	yes
bat	warm-blooded	yes	no	yes	no*
whale	warm-blooded	yes	no	no	no*
salamander	cold-blooded	no	yes	yes	no
komodo dragon	cold-blooded	no	yes	no	no
python	cold-blooded	no	no	yes	no
salmon	cold-blooded	no	no	no	no
eagle	warm-blooded	no	no	no	no
guppy	cold-blooded	yes	no	no	no




# Overfitting due à la présence du bruit

Soit l'ensemble des données de test suivant

Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
human	warm-blooded	yes	no	no	yes
pigeon	warm-blooded	no	no	no	no
elephant	warm-blooded	yes	yes	no	yes
leopard shark	cold-blooded	yes	no	no	no
turtle	cold-blooded	no	yes	no	no
penguin	cold-blooded	no	no	no	no
eel	cold-blooded	no	no	no	no
dolphin	warm-blooded	yes	no	no	yes
spiny anteater	warm-blooded	no	yes	yes	yes
gila monster	cold-blooded	no	yes	yes	no



À partir de l'arbre de décision identifié, les humains et les dauphins sont classés comme non-mammifère !



# Overfitting due à un ensemble d'apprentissage très petit

- Soit l'ensemble d'apprentissage suivant qui contient 5 enregistrements qui sont étiquetés correctement.
- Cet ensemble contient deux classes : la classe mammifère désignée par « yes » et la classe non - mammifère désigné par « no ».

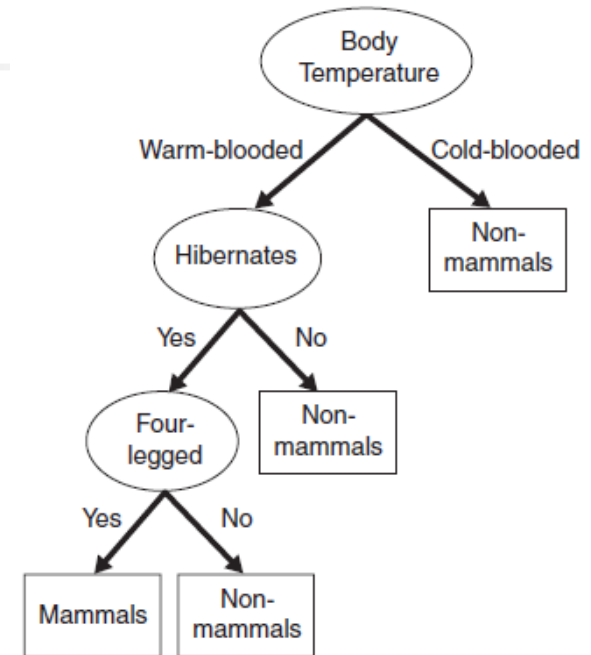
Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
salamander	cold-blooded	no	yes	yes	no
guppy	cold-blooded	yes	no	no	no
eagle	warm-blooded	no	no	no	no
poorwill	warm-blooded	no	no	yes	no
platypus	warm-blooded	no	yes	yes	yes

# Overfitting due à un ensemble d'apprentissage très petit

L'arbre de décision identifié →

Soit l'ensemble des données de test suivant :

Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
human	warm-blooded	yes	no	no	yes
pigeon	warm-blooded	no	no	no	no
elephant	warm-blooded	yes	yes	no	yes
leopard shark	cold-blooded	yes	no	no	no
turtle	cold-blooded	no	yes	no	no
penguin	cold-blooded	no	no	no	no
eel	cold-blooded	no	no	no	no
dolphin	warm-blooded	yes	no	no	yes
spiny anteater	warm-blooded	no	yes	yes	yes
gila monster	cold-blooded	no	yes	yes	no



À partir de l'arbre de décision identifié, les humains, les éléphants et les dauphins sont classés comme non-mammifère !



# Overfitting (Surapprentissage)

En général :

- Les arbres de décision font très peu d'hypothèses sur les données d'entraînement (par opposition à d'autres modèles qui supposent, par exemple, que les données sont linéaires).
- En l'absence de contraintes, la structure de l'arbre s'adaptera aux données d'entraînement en les ajustant le mieux possible, voire en les surajustant.



# Comment éviter l'overfitting

## Le pré-élagage (stop earlier)

- Une solution consiste à restreindre la liberté de l'arbre de décision durant l'entraînement → Limiter la profondeur maximale de l'arbre de décision.
- Solution implémentée dans Scikit-Lear → Utilisation des hyperparamètres de régularisation.
- Dans Scikit-Learn, c'est l'hyperparamètre `max_depth` qui permet de contrôler la profondeur (la valeur par défaut est `None`, qui signifie qu'il n'y a pas de limite). En réduisant `max_depth`, vous régularisez le modèle et réduisez par conséquent le risque de l'overfitting.
- Voir l'exemple de Iris dataset.



# Comment éviter l'overfitting

## Hyperparamètre de régularisation\*

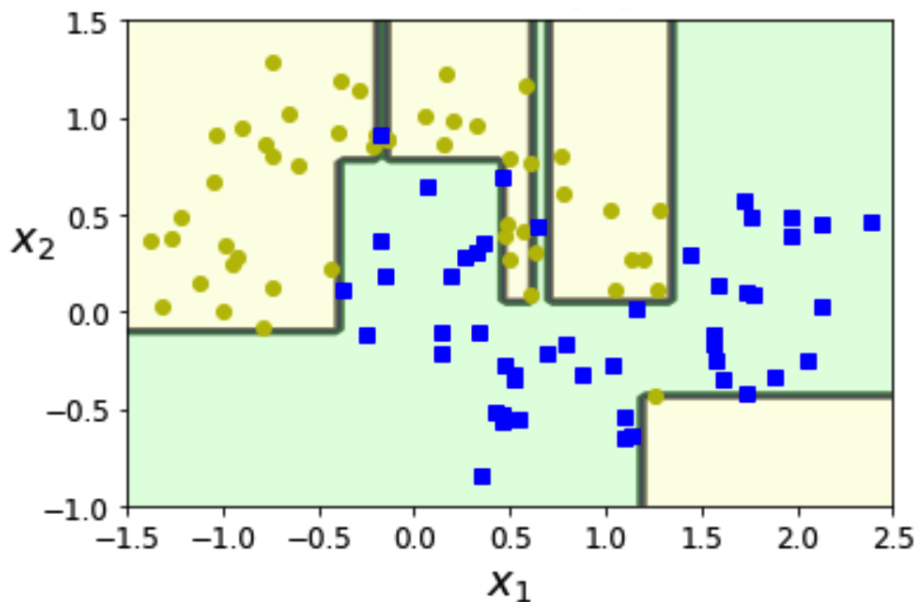
- La classe `DecisionTreeClassifier` possède quelques autres paramètres qui restreignent la forme de l'arbre de décision :
  - **`min_samples_split`** (nombre minimum d'observations que doit comporter un nœud avant qu'il puisse être divisé).
  - **`min_samples_leaf`** (nombre minimum d'observations qu'un nœud terminal doit avoir), **`min_weight_fraction_leaf`** (identique à **`min_samples_leaf`** mais exprimé sous forme d'une fraction du nombre total d'observations pondérées).
  - **`max_leaf_nodes`** (nombre maximum de nœuds terminaux) et **`max_features`** (nombre maximum de caractéristiques évaluées à chaque nœud en vue de sa division). Augmenter les hyper-paramètres `min_*` ou réduire les hyper- paramètres `max_*` va régulariser le modèle.



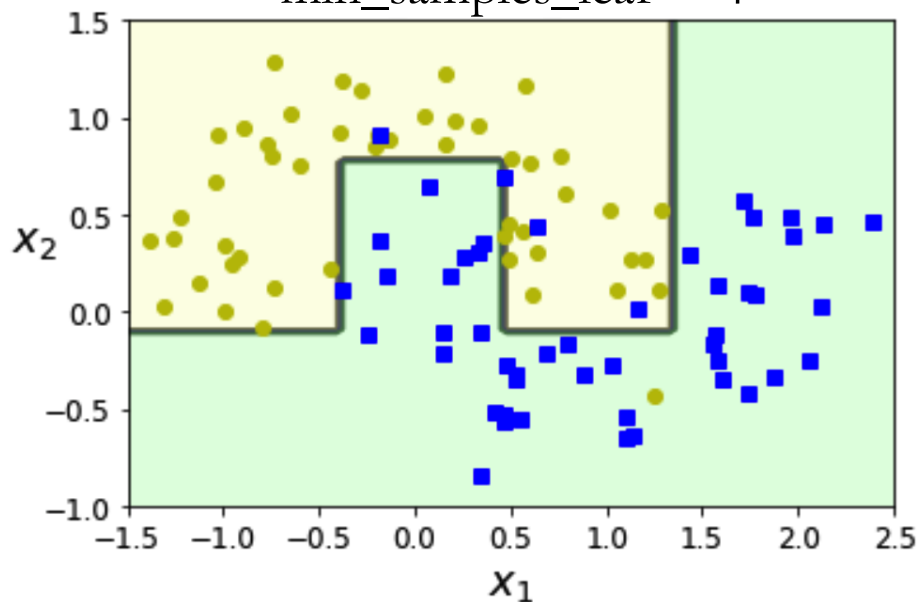
# Comment éviter l'overfitting

## Hyperparamètre de régularisation

Sans restriction



`min_samples_leaf = 4`



Régularisation utilisant `min_samples_leaf`

Figure tirée du livre : Machine Learning avec Scikit-Learn d'Aurélien Géron



# Le pré-élagage

---

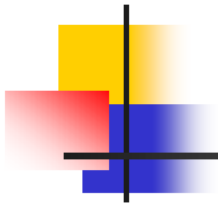
- L'avantage de cette approche est d'éviter de générer des arbres complexes.
- Cependant, cette méthode souffre du problème du choix des paramètres. En plus, cette approche de pré-élagage est « myopes » (puisque'elles ne prennent en compte qu'un critère local à la feuille examinée), et peut, de ce fait, manquer un développement de l'arbre qui serait excellent.
- Comme alternative, il y a les méthodes d'élagage *a posteriori*, une fois que l'arbre a été entièrement développé.

# Le post-élagage (post-pruning)

- Construire l'arbre au complet puis l'élaguer (trimming).
- L'élagage peut être fait en remplaçant un sous arbre de l'arbre de décision original par un nouveau nœud (type feuille) dont le label de la classe est déterminé à partir du label de la classe dominante des enregistrements (la classe qui représente le plus grand nombre d'enregistrements) associées au sous arbre éliminé.

```
depth > 1:  
| MultiAgent = 0:  
| | depth > 2: class 0  
| | depth <= 2:  
| | | MultiP = 1: class 0  
| | | MultiP = 0:  
| | | | breadth <= 6: class 0  
| | | | breadth > 6:  
| | | | | RepeatedAccess <= 0.322: class 0  
| | | | | RepeatedAccess > 0.322: class 1  
| MultiAgent = 1:  
| | totalPages <= 81: class 0  
| | totalPages > 81: class 1
```

```
depth > 1:  
| MultiAgent = 0: class 0  
| MultiAgent = 1:  
| | totalPages <= 81: class 0  
| | totalPages > 81: class 1
```



# Plan

---

1- Analyse prédictive

2- Classification

**3- Arbre de décision**

- Construction de l'arbre
- **Élagage de l'arbre**



# Le post-élagage (post-pruning)

## □ Principe

Le post-élagage est une technique plus valide théoriquement et plus efficace que le pré-élagage. Elle consiste à d'abord construire l'arbre de décision, puis chercher à le simplifier en l'élaguant progressivement en remontant des feuilles vers la racine. Pour juger quand il est bon d'arrêter d'élaguer l'arbre, on utilise un critère de qualité qui exprime souvent un compromis entre l'erreur commise par l'arbre et une autre mesure de sa complexité.



# Le post-élagage (post-pruning)

- Soit  $T_{max}$  l'arbre obtenu à partir de l'ensemble d'apprentissage
- Construire une suite d'arbres  $\{T_{max}, T_1, \dots, T_k, \dots, T_n\}$  en partant des feuilles et en remontant vers la racine, et ce en transformant un nœud en feuille à chaque étape. Prendre note que  $T_n$  est l'arbre constitué d'une seule feuille comprenant les  $n$  points d'apprentissage.
- Pour passer de  $T_k$  à  $T_{k+1}$ , il faut transformer un nœud dans  $T_k$  en feuille. Pour savoir si cet élagage serait bénéfique, l'idée générale est de comparer le « coût » du nouvel arbre à celui du précédent et arrêter l'élagage si le coût est supérieur. La comparaison se fait à l'aide d'un critère de qualité.



## Critère pour l'élagage

La technique généralement utilisée est de choisir le(s) nœud(s) qui minimise(nt) sur l'ensemble des nœuds de  $T_k$  le critère suivant :

$$\overline{w}(T_k, v) = \frac{MC_{\text{éla}}(v, k) - MC(v, k)}{n(k)(nt(v, k) - 1)}$$

$MC_{\text{éla}}(v, k)$	Nombre d'exemples de l'ensemble d'apprentissage mal classés par le nœud $v$ de $T_k$ dans <b>l'arbre élagué à <math>v</math></b>
$MC(v, k)$	Nombre d'exemples de l'ensemble d'apprentissage mal classés sous le nœud $v$ dans <b>l'arbre non élagué</b>
$n(k)$	Nombre de feuilles de $T_k$
$nt(v, k)$	Nombre de feuilles du sous-arbre de $T_k$ situé sous le nœud $v$



## Critère pour l'élagage

Ce critère permet donc d'élaguer dans  $T_k$  un ou plusieurs nœuds de façon à ce que l'arbre obtenu, noté  $T_{k+1}$ , possède le meilleur compromis entre taille et taux d'erreur.

Finalement, la suite  $\{T_{max}, T_1, \dots, T_k, \dots, T_n\}$  que l'on construit ainsi possède un élément  $T_j$  pour lequel le nombre d'erreurs commises sur l'ensemble de validation est minimal. C'est cet arbre-là qui sera finalement retenu par la procédure d'élagage.





# Algorithme d'élagage

**Procédure : élaguer ( $T_{\max}$ )**

$k \leftarrow 0$

$T_k \leftarrow T_{\max}$

**tant que**  $T_k$  a plus d'un nœud **faire**

**pour** chaque nœud  $v$  de  $T_k$  **faire**

        calculer le critère  $w(T_k, v)$  sur l'ensemble d'apprentissage

**fin pour**

    choisi le nœud  $v_n$  pour lequel le critère est minimum

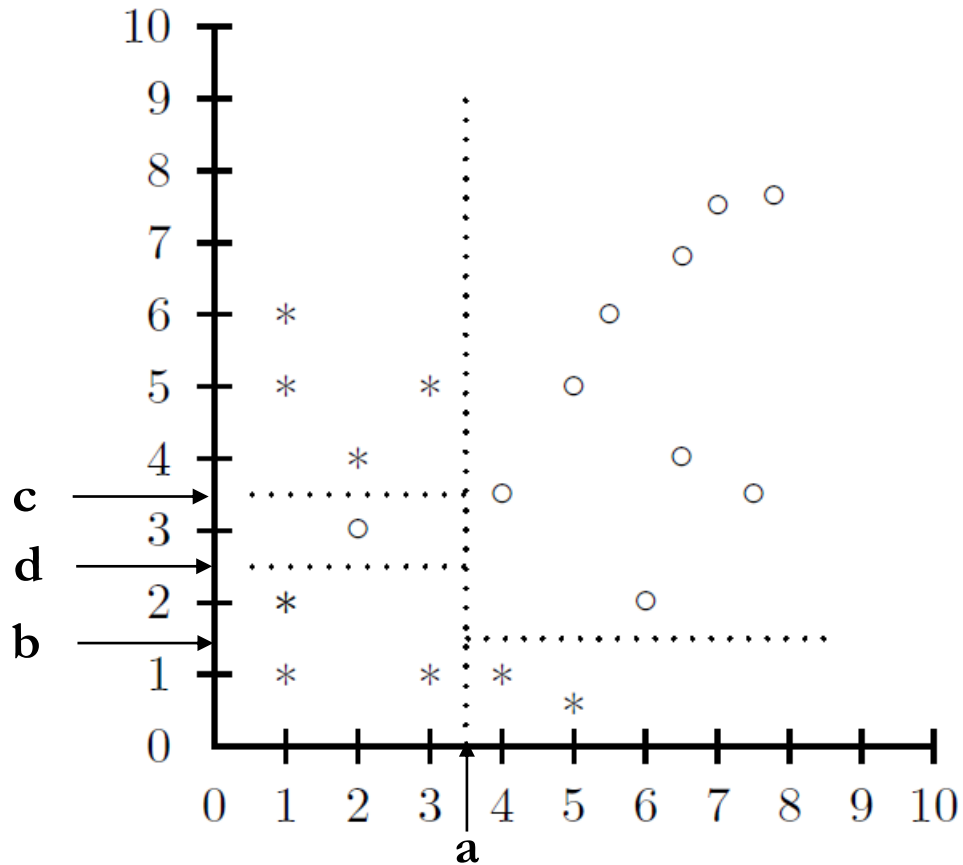
$T_{k+1}$  se déduit de  $T_k$  en y remplaçant  $v_n$  par une feuille

$k \leftarrow k+1$

**fin tant que**

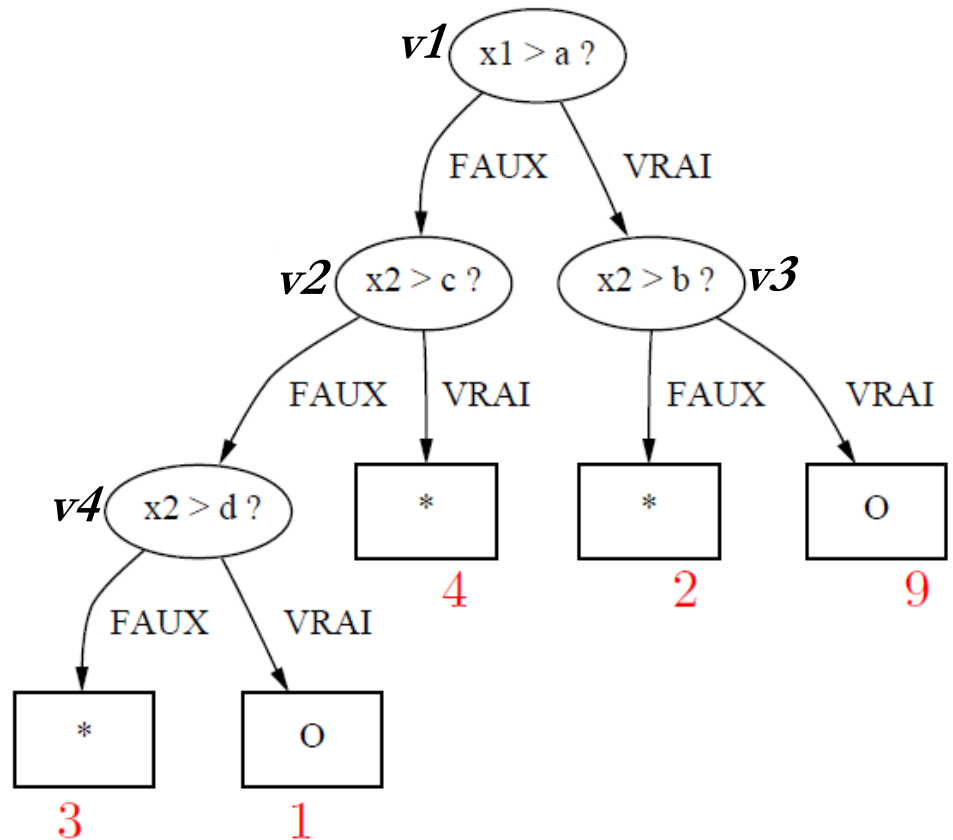
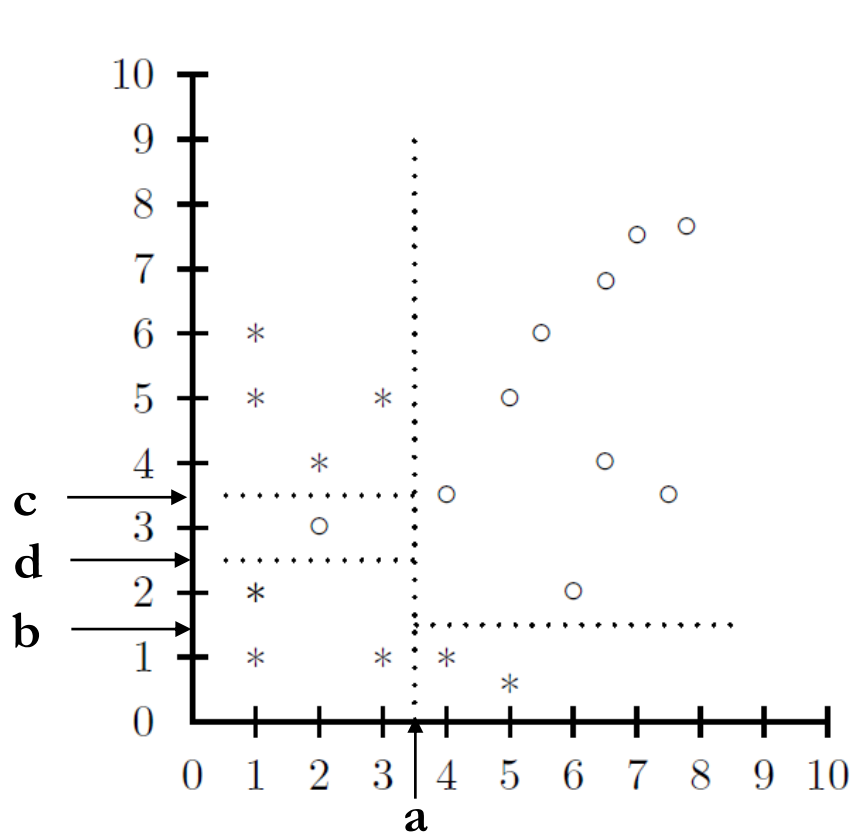
Dans l'ensemble des arbres  $\{T_{\max}, T_1, \dots, T_k, \dots, T_n\}$ , choisir celui qui a la plus petite erreur en classification sur l'ensemble de validation 81

# Un exemple d'élagage



Un ensemble d'apprentissage qui contient deux classes

# Un exemple d'élagage (suite)

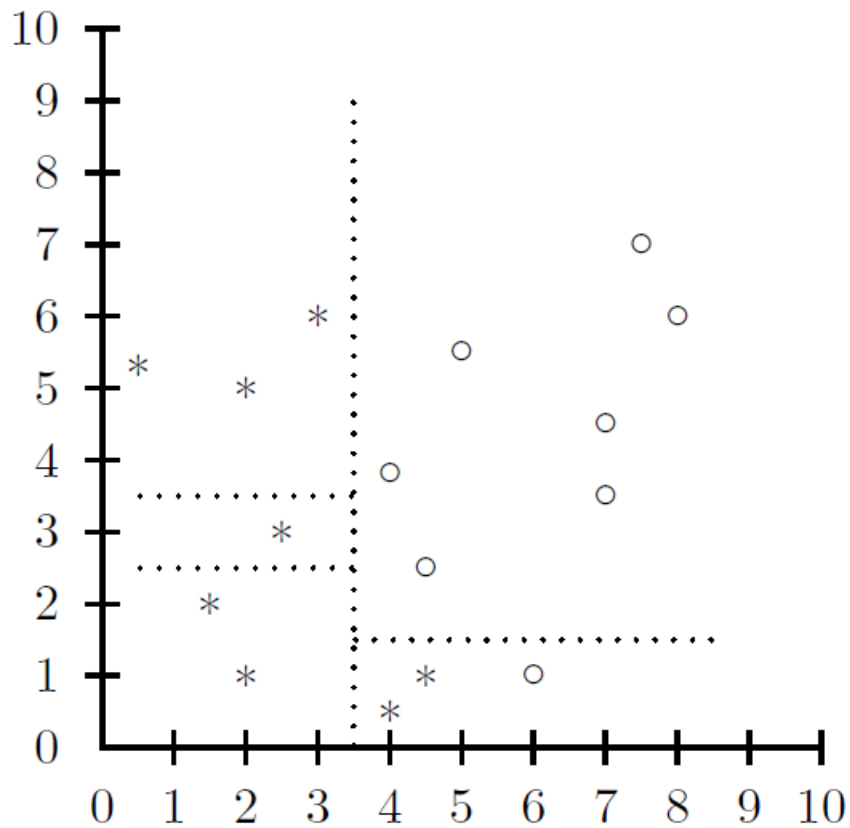


L'arbre de décision identifié

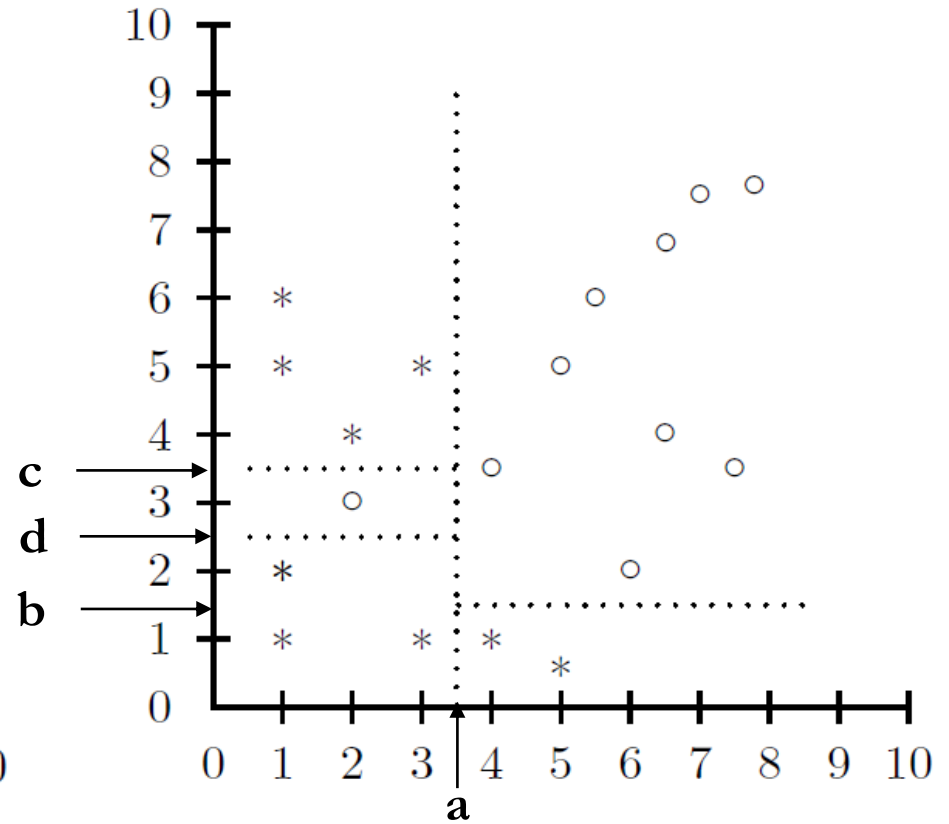
Taux d'erreur de l'arbre

sur les données d'apprentissage : **0/19**

# Un exemple d'élagage (suite)

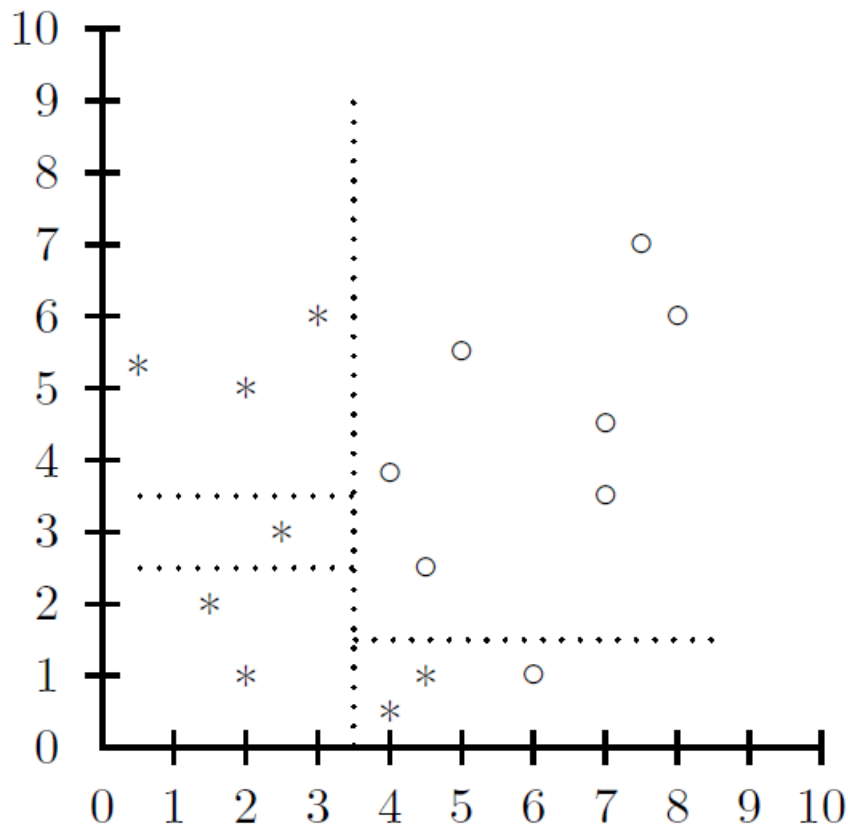


Données de validation

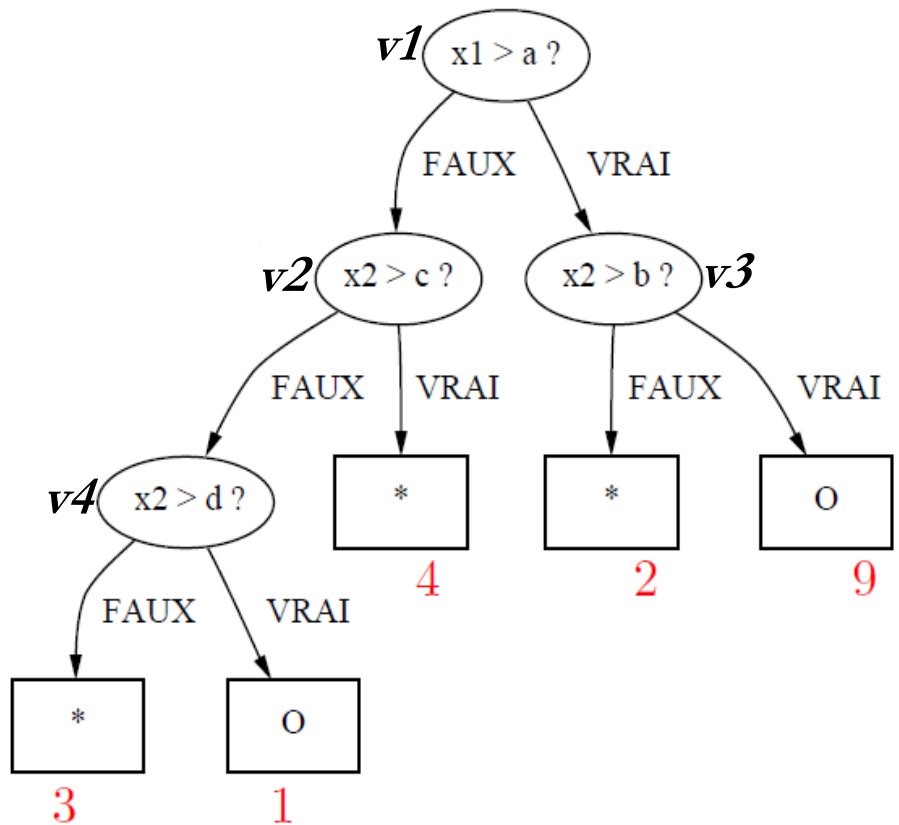


Données d'apprentissage

# Un exemple d'élagage (suite)

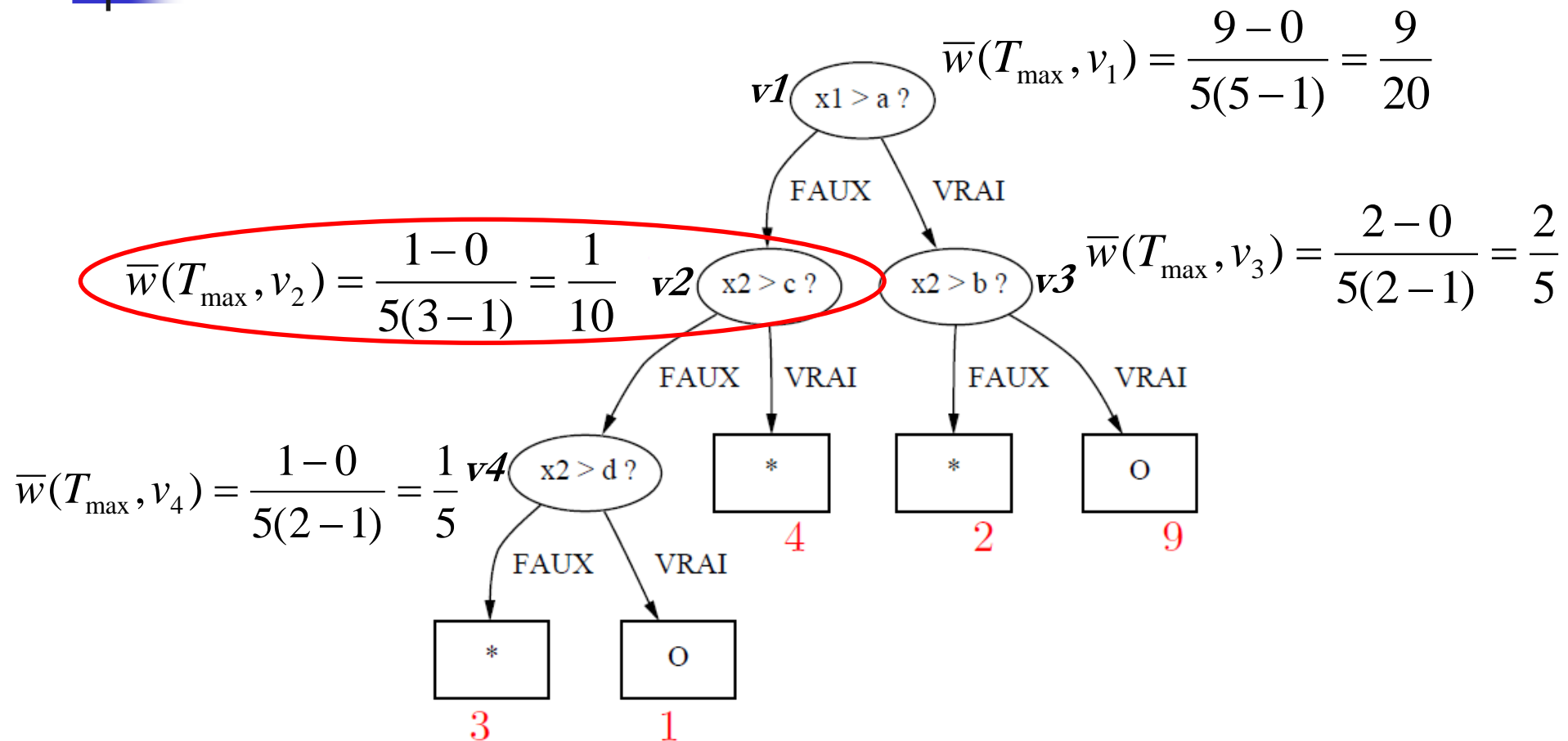


Données de validation



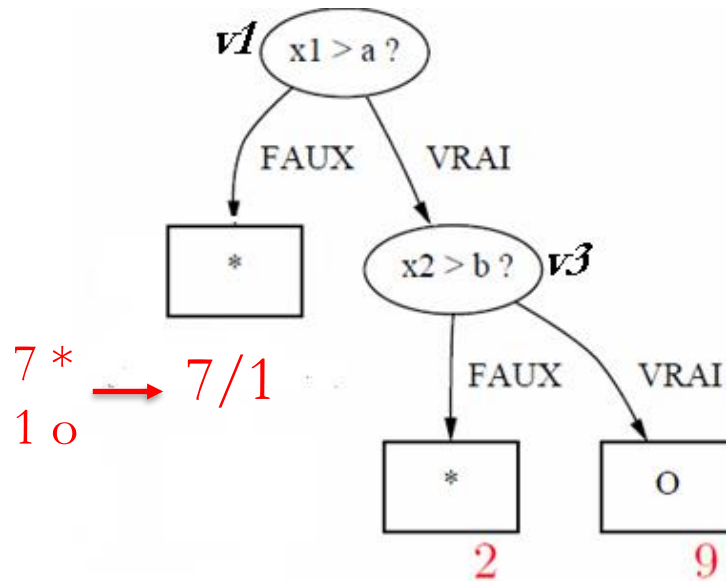
Taux d'erreur de l'arbre sur les données de validation : **2/16**

# Calcul du critère d'élagage pour chaque nœud



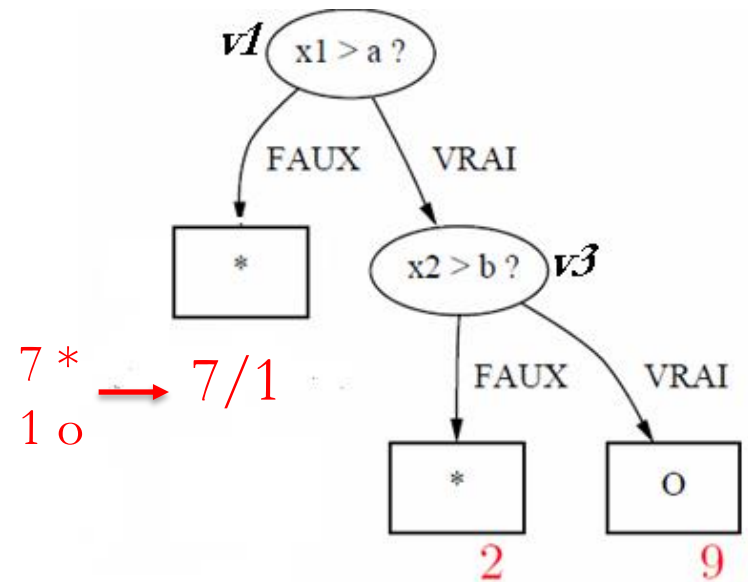
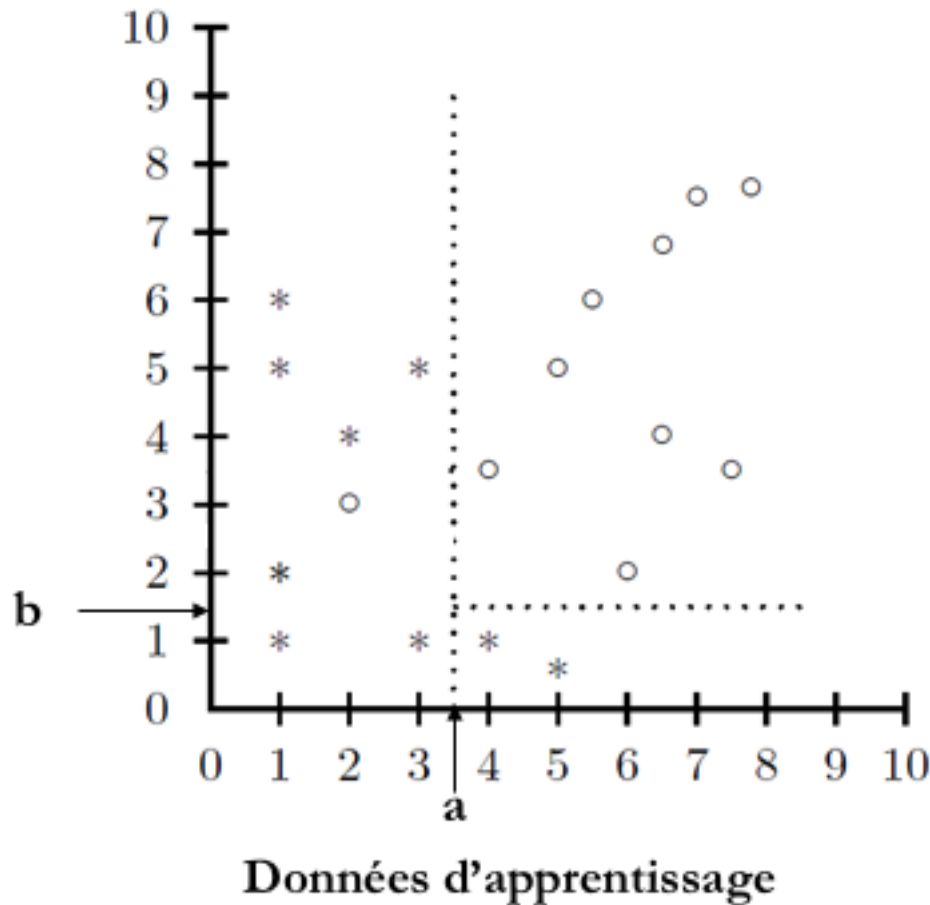
→ On doit donc remplacer le nœud «  $x2 > c ?$  » par une feuille

# Un exemple d'élagage (suite)



Arbre  $T_1$ : résultat de l'élagage de  $T_{max}$  au nœud  $v_2$

# Un exemple d'élagage (suite)

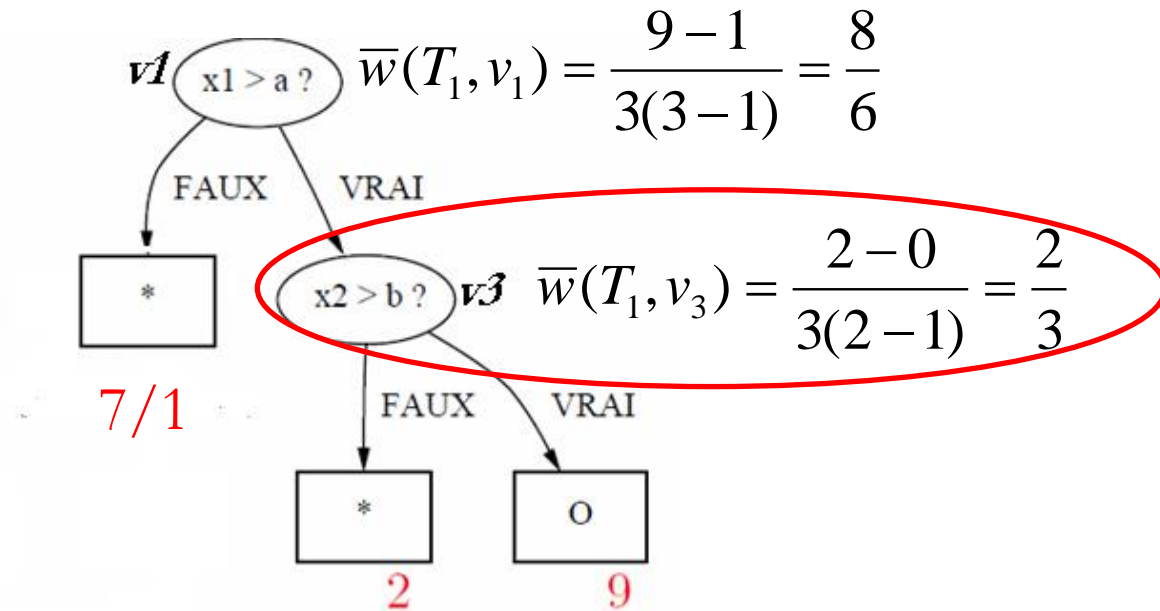


Arbre  $T_1$ : résultat de l'élagage de  $T_{max}$  au nœud  $v_2$



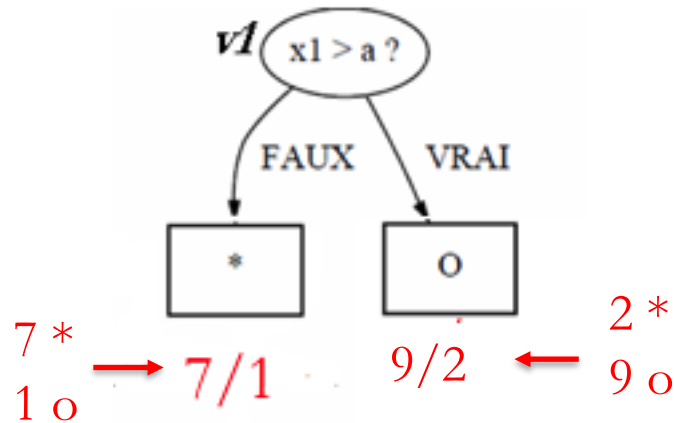
# Un exemple d'élagage (suite)

Élagage de l'arbre  $T_1$



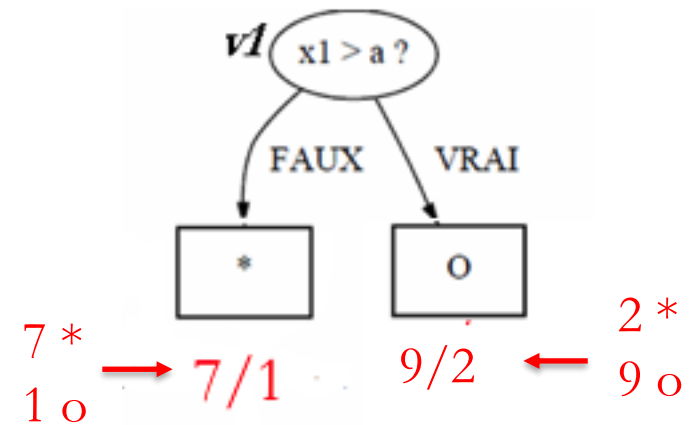
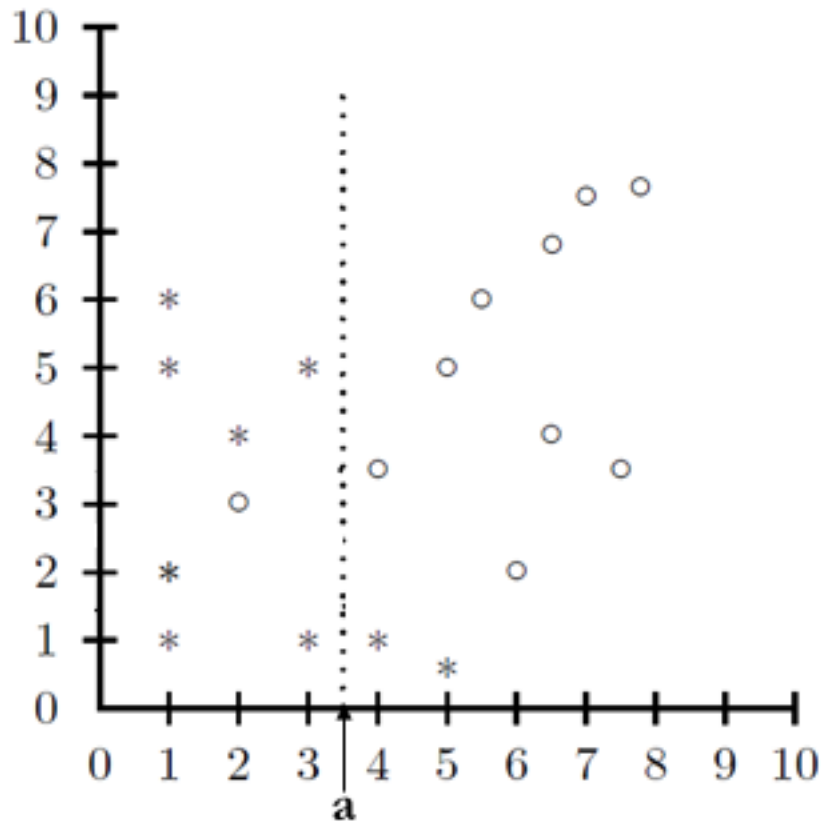
→ L'arbre  $T_2$  choisi résultera de l'élagage de  $v_3$  dans  $T_1$  ; il aura donc pour seul nœud la racine de  $T_{max}$ , avec une feuille pour chaque classe. On peut donc arrêter le processus d'élagage à ce niveau.

# Un exemple d'élagage (suite)



Arbre  $T_2$ : résultat de l'élagage de  $T_1$  au nœud  $v_3$

# Un exemple d'élagage (suite)



Arbre  $T_2$ : résultat de l'élagage de  $T_1$  au nœud  $v_3$



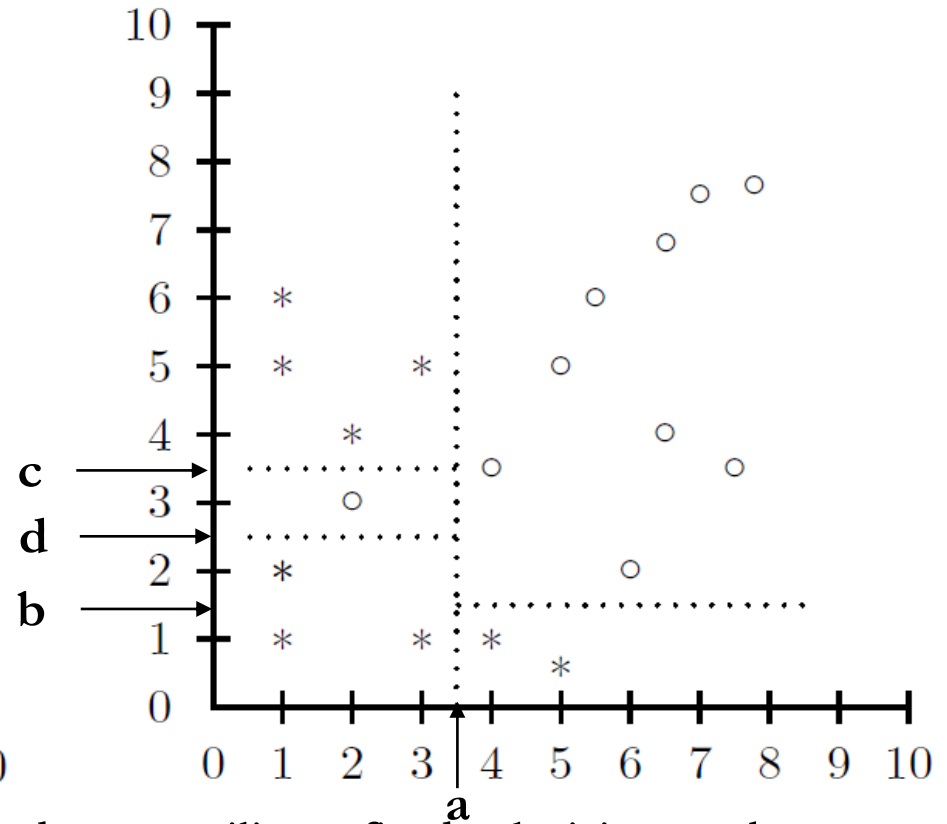
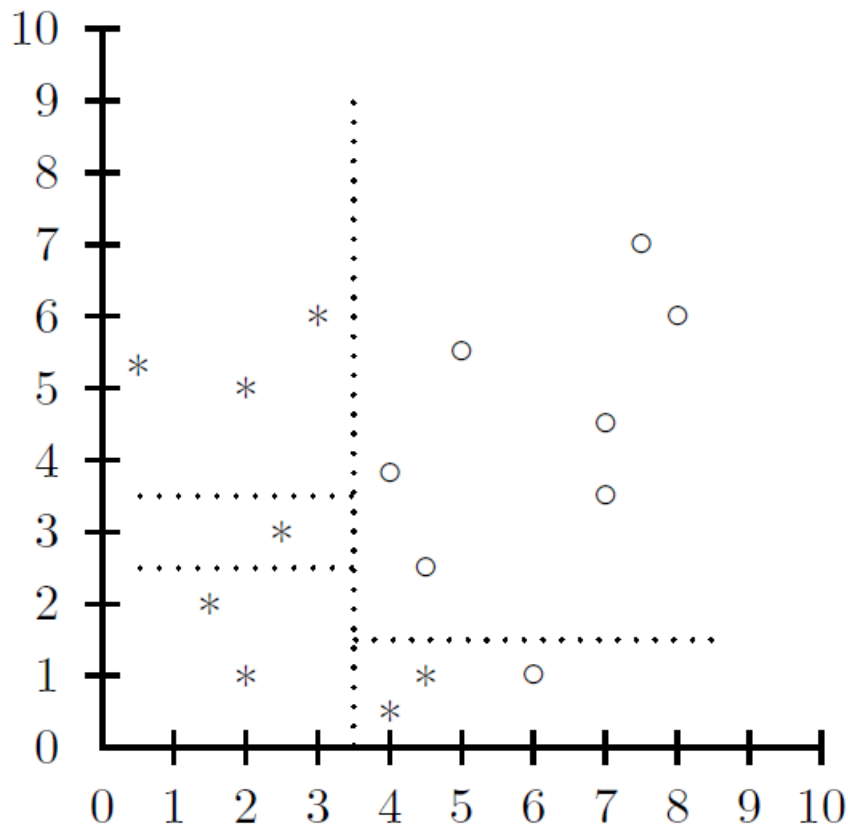
## Un exemple d'élagage (suite)

Puisqu'on suppose dispose d'un ensemble de validation, on va tester ce dernier sur les arbres  $T_{max}$ ,  $T_1$  et  $T_2$  que l'on vient de calculer.

On y ajoute l'arbre noté plus haut  $T_n$ , composé d'une seule feuille et qui ne représente que la probabilité *a priori* des classes dans l'ensemble d'apprentissage.

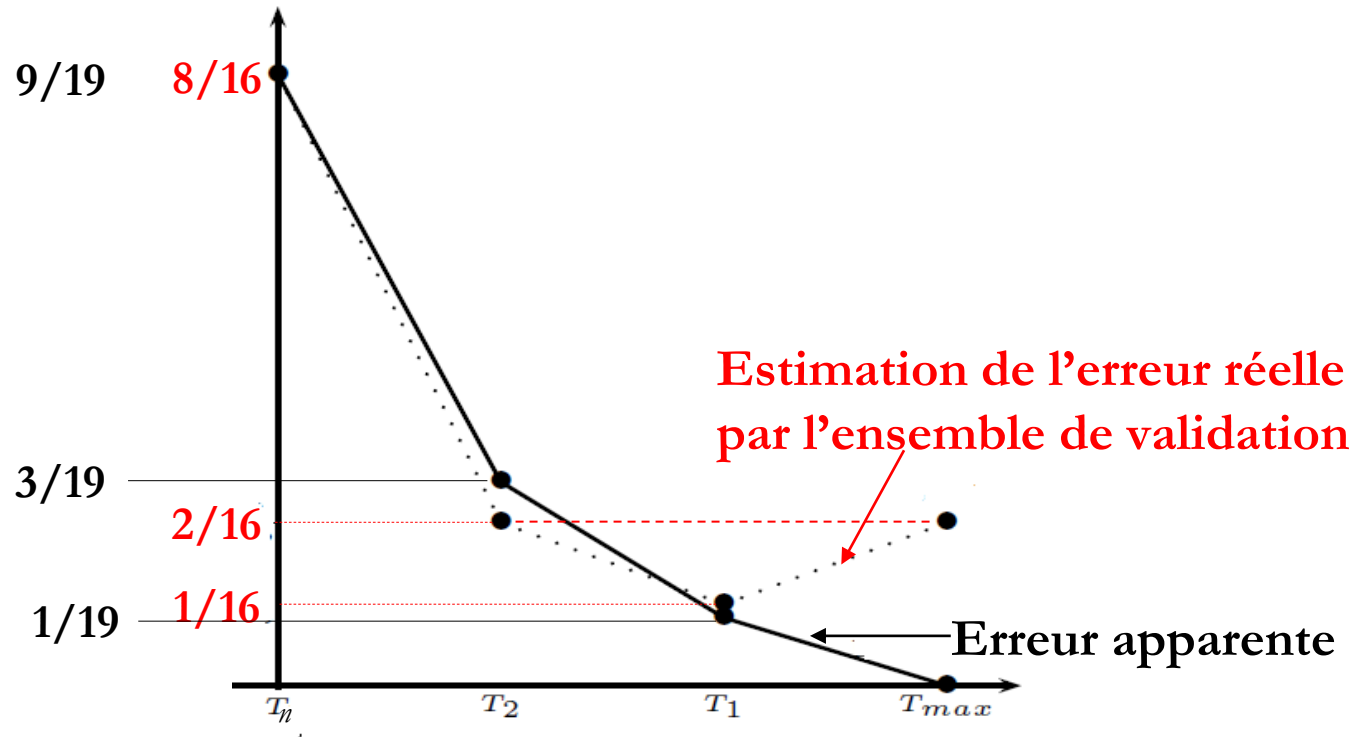
On choisira l'arbre que possède la meilleure estimation de taux d'erreur de classification. La procédure d'élagage sera alors terminée.

# Un exemple d'élagage (suite)



Les données de tests (validation) que nous devons utiliser afin de choisir un arbre parmi  $T_{max}$ ,  $T_1$ ,  $T_2$  ou  $T_n$

# Un exemple d'élagage (suite et fin)



- C'est l'arbre  $T_1$  qui a le plus petit taux d'erreur sur l'ensemble de données de validation → **on doit choisir  $T_1$  comme le meilleur modèle.**
- Selon le graphe illustré ci-dessus, on voit clairement que  $T_{max}$  souffre du problème de l'overfitting
- L'arbre  $T_n$  a une performance comparable à un classificateur aléatoire (taux d'erreur  $\approx 50\%$ )



# Arbre de décision - discussion

---

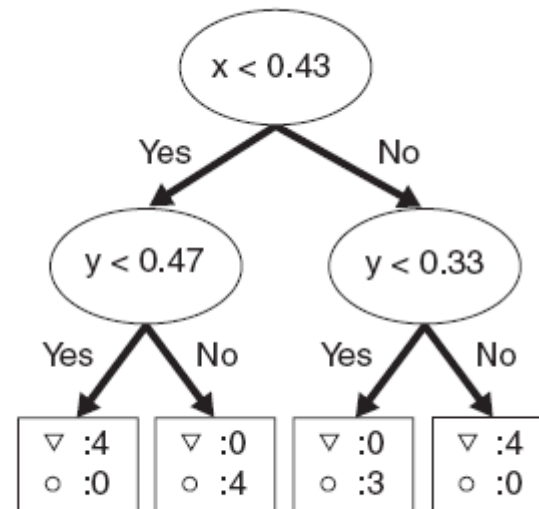
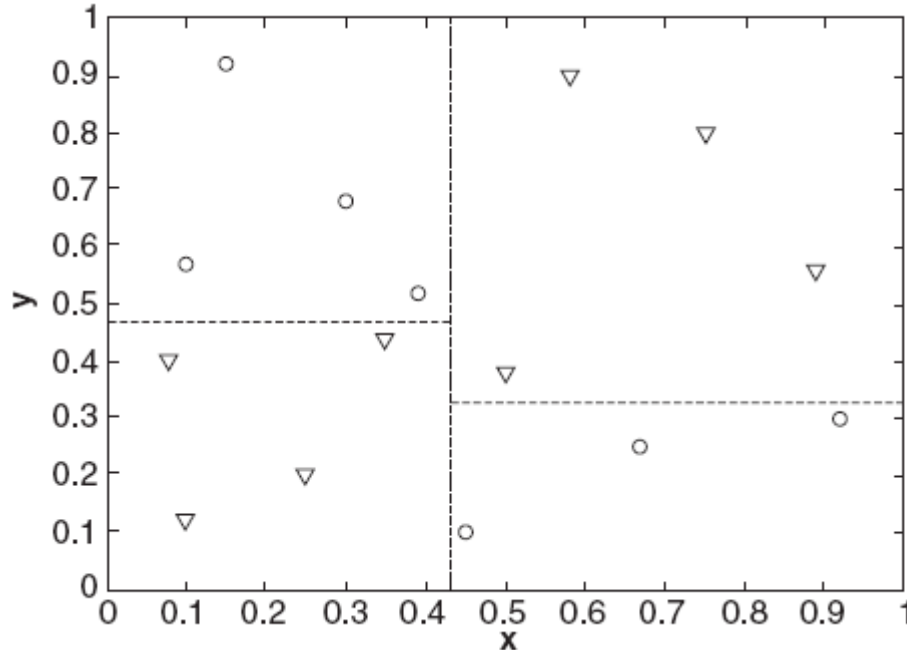
## Avantages

- Lisibilité du résultat.
- Pas de paramétrage difficile pour la construction de l'arbre.
- Efficacité et disponibilité (présent dans tous les logiciels de l'apprentissage automatique et de data mining).

# Arbre de décision - discussion

## Limites

- Non incrémental : on doit recommencer la construction de l'arbre si on veut intégrer de nouvelles données.
- Produit des frontières de décisions qui sont parallèles aux axes seulement.

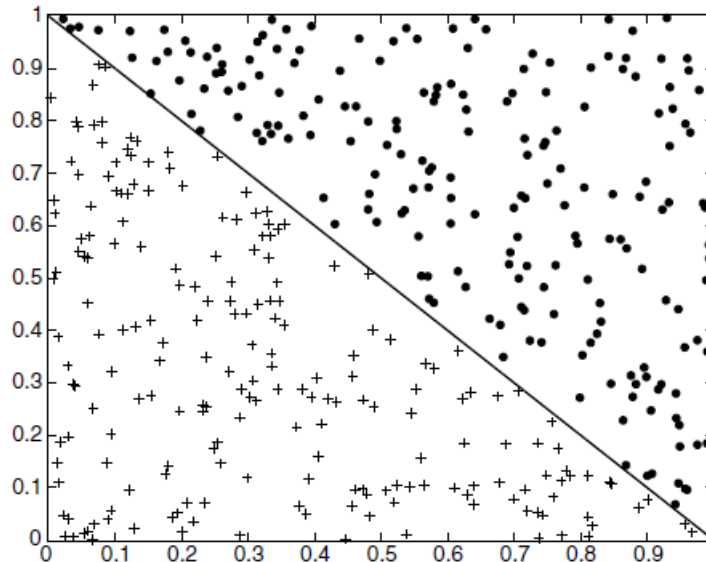




# Arbre de décision - discussion

## Limites

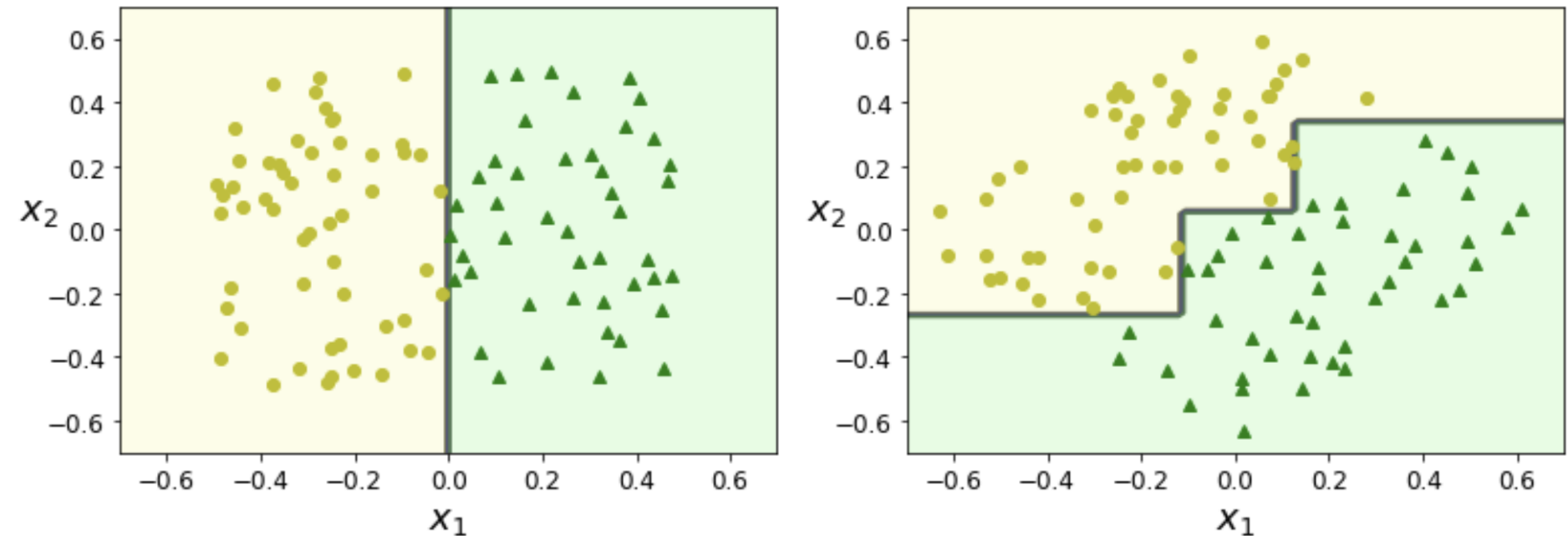
Voici un exemple ou la séparation des données à l'aide des arbres de décision ne sera pas optimale, car on teste un seul attribut à la fois



# Arbre de décision - discussion

## Limites

Sensibilité à la rotation des données d'entraînement



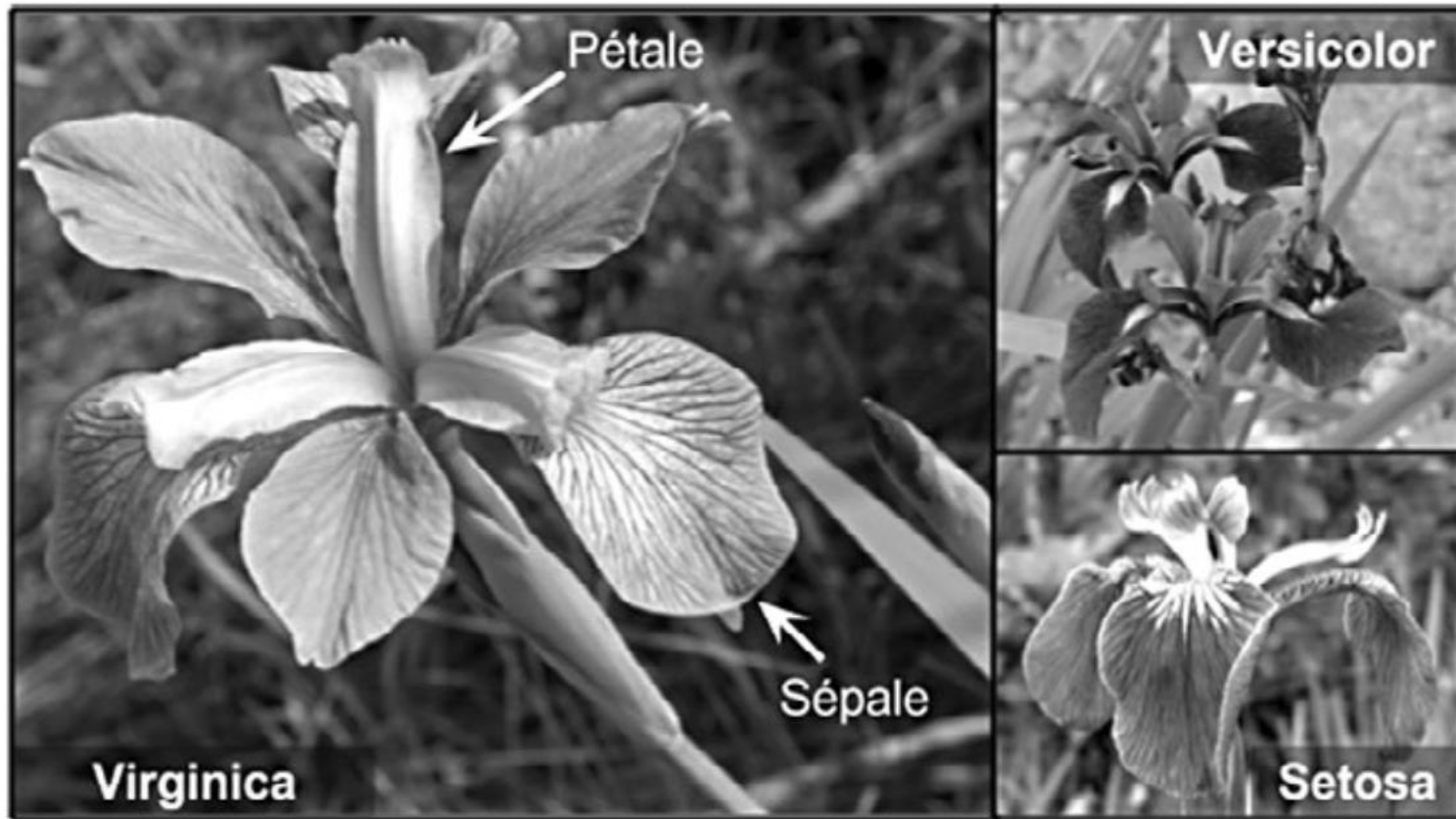


# Arbre de décision avec Scikit-Learn\*

- Scikit-Learn, l'une des bibliothèques open source les plus simples et néanmoins les plus puissantes disponibles aujourd'hui, que vous pourrez directement utiliser dans vos systèmes en production.
- Scikit-Learn fournit des valeurs par défaut raisonnables pour la plupart des paramètres, permettant ainsi de créer facilement et rapidement un système basique et opérationnel.
- Scikit-Learn utilise l'algorithme CART qui ne produit que des arbres binaires : les noeuds non terminaux ont toujours deux fils (c.-à-d. que les questions n'ont que les réponses oui ou non).

# Arbre de décision avec Scikit-Learn

Voir l'exemple sur le dataset Iris (Fleurs de trois espèces) – disponible dans Moodle



La photo provient du livre : Machine Learning avec Scikit-Learn d'Aurélien Géron<sup>100</sup>