

- Classification:
  - prédit la catégorie/classe d'un objet
  - construit un modèle basé sur un jeu d'apprentissage et des valeurs (nom des catégories) et l'utilise pour classer des données nouvelles
- Prédiction:
  - modélise des données numériques pour prédire des valeurs inconnues ou manquantes
- Applications
  - diagnostique médical
  - séquences codantes
  - structure secondaire, tertiaire
  - ...

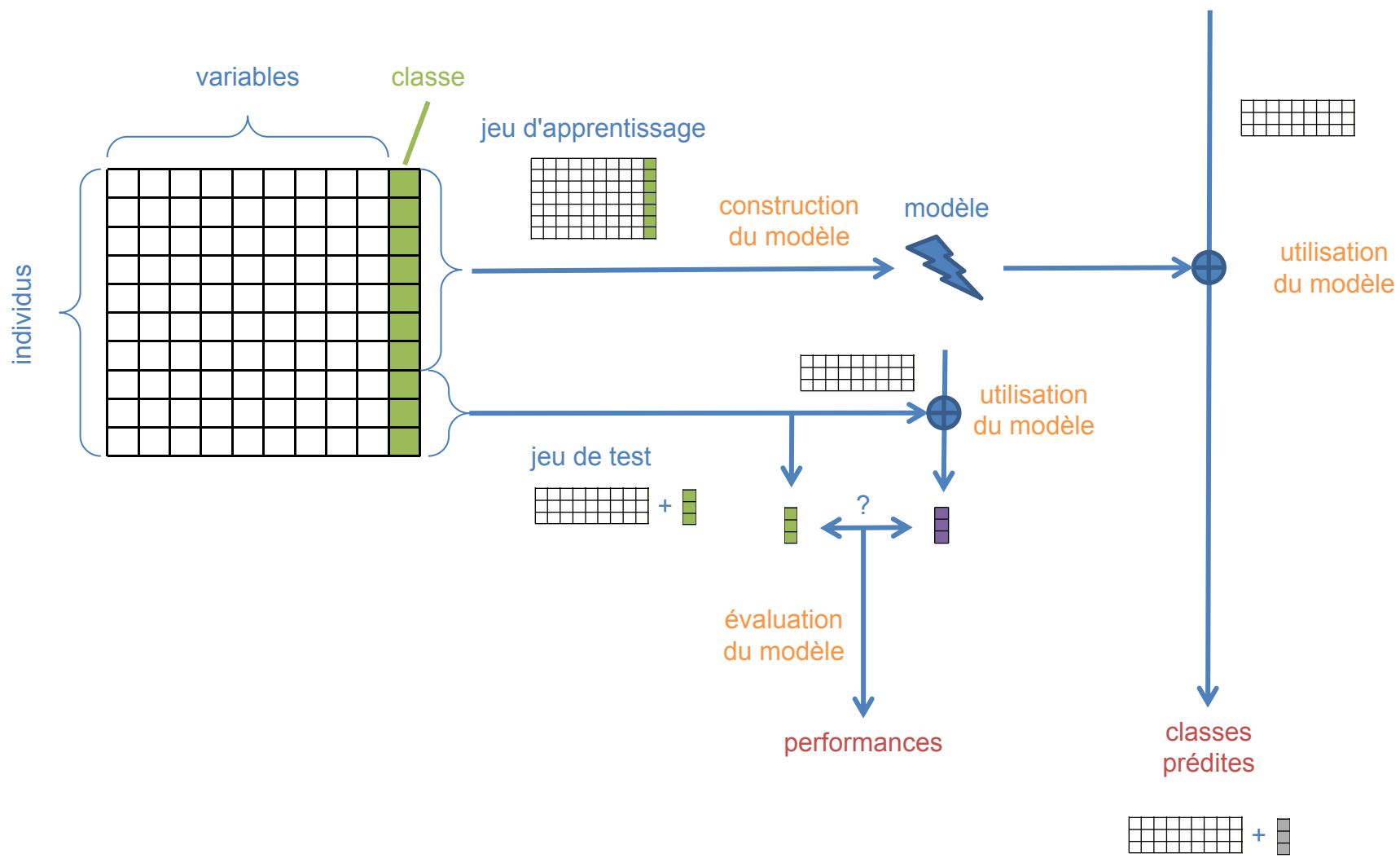
- Méthodes
  - arbre de décision et forêt aléatoire
  - classificateur bayésien (naïf ou réseau bayésien)
  - analyse discriminante
  - plus proches voisins
  - règles d'association
  - réseau de neurones (perceptron et deep learning)
  - algorithme génétique
  - machine à vecteur de support
  - modèle de Markov
  - ...
- Evaluation des performances

- Construction du modèle
  - chaque objet appartient à une classe connue
  - jeu de données d'apprentissage : ensemble des objets utilisés pour la construction du modèle
- Utilisation du modèle pour classer des objets
  - classe connue : estimation de la précision du modèle
    - les classes connues du jeu d'apprentissage sont comparées à celles prédites
    - précision : pourcentage d'objets de jeu de test correctement classés
    - le jeu de test est indépendant du jeu d'apprentissage sinon risque de biais et de sur-apprentissage
  - classe non connue (objets nouveaux)

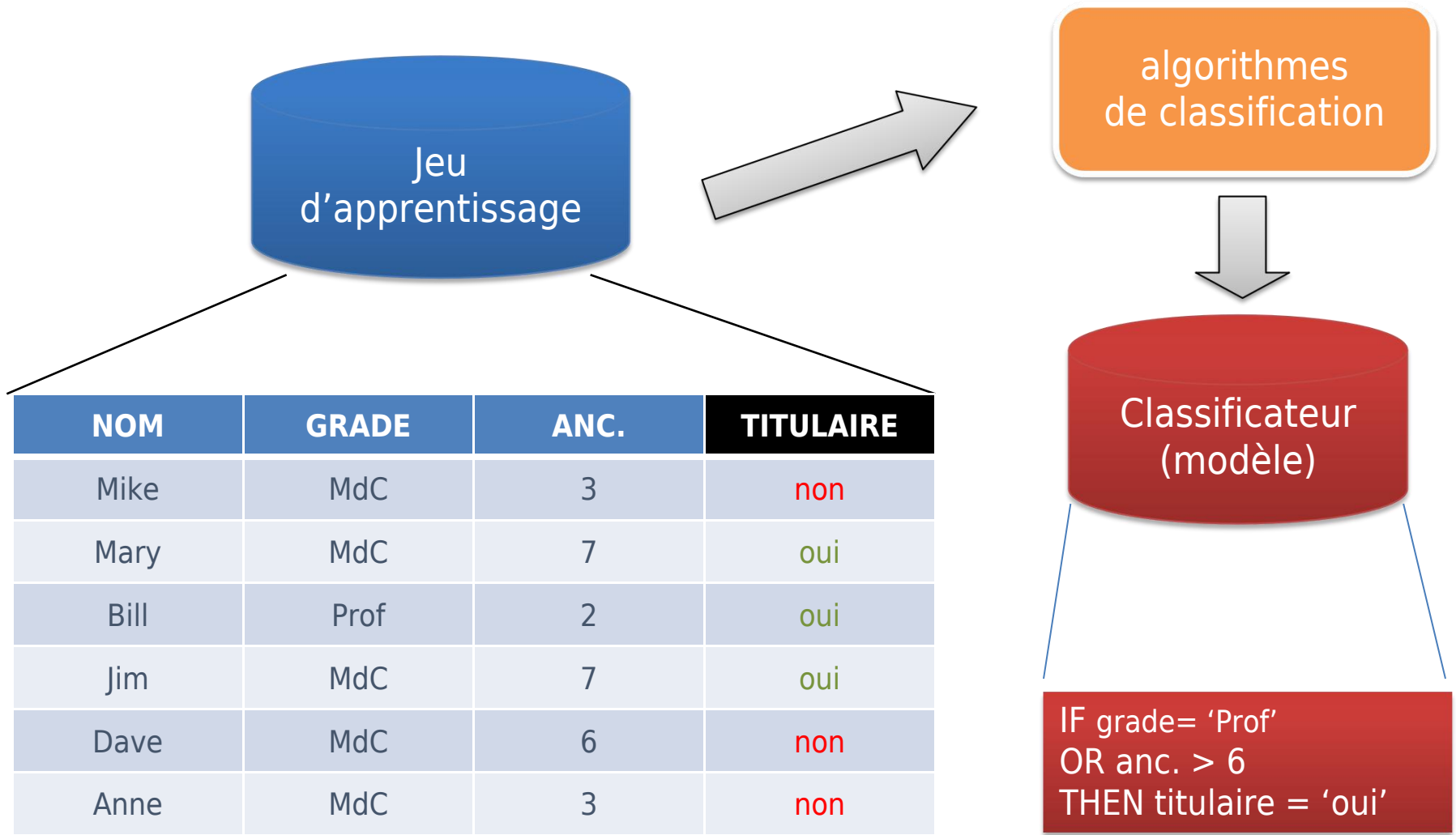
# Schéma

1. Construction : jeu de données pour lesquelles la classe est connue

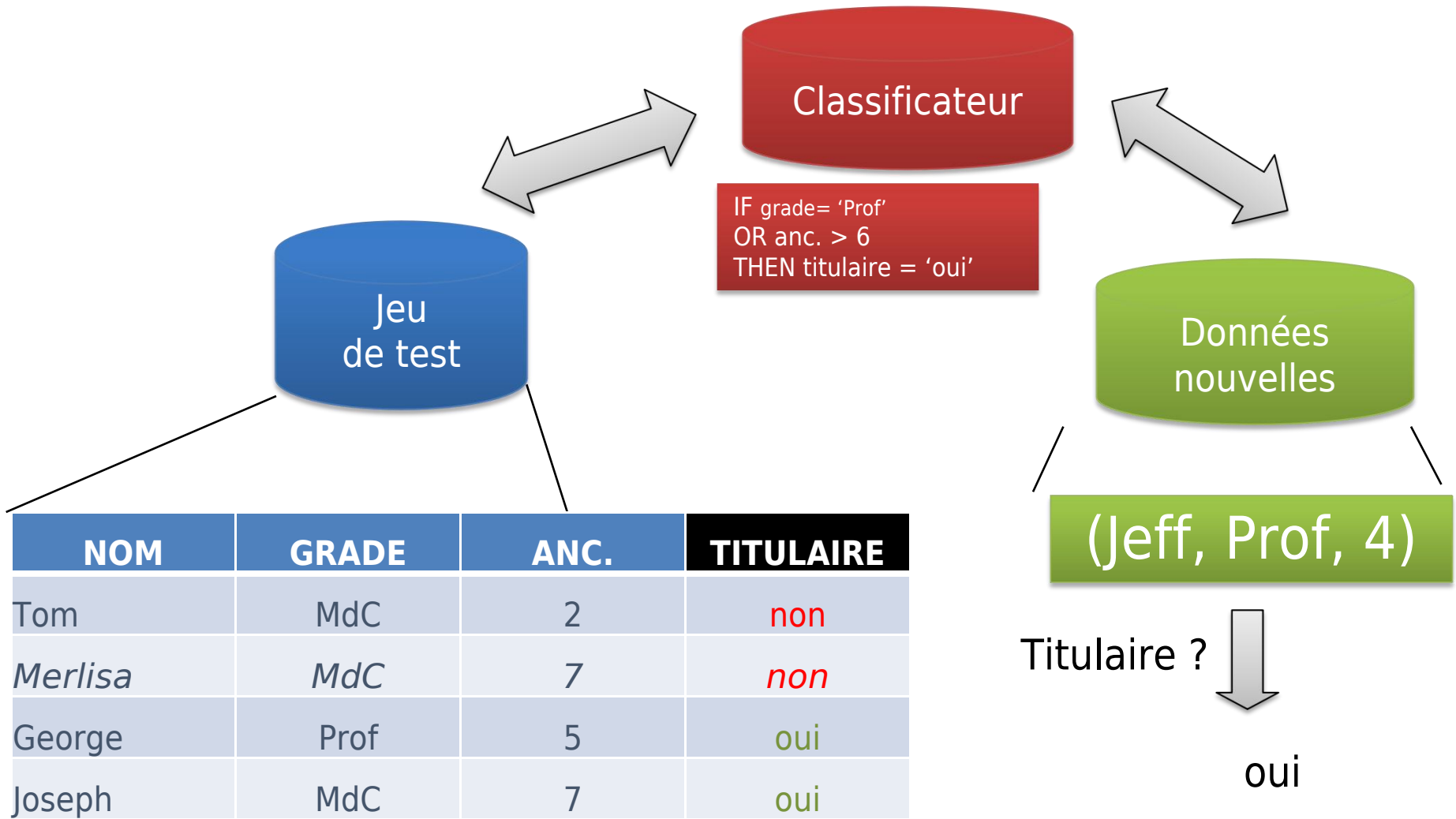
2. Utilisation : Données nouvelles



# Classification : construction du modèle



# Classification : utilisation du modèle



- Apprentissage supervisé (classification)
  - supervision : le jeu de données d'apprentissage fournit les classes des objets
  - les nouveaux objets sont classés en fonction du jeu d'apprentissage
  
- Apprentissage non supervisé (clustering)
  - Pas de classes définies
  - Étant donné un ensemble de mesures, d'observations, *etc.*, essayer d'établir l'existence de classes ou de clusters dans les données

## Considérations pour la classification

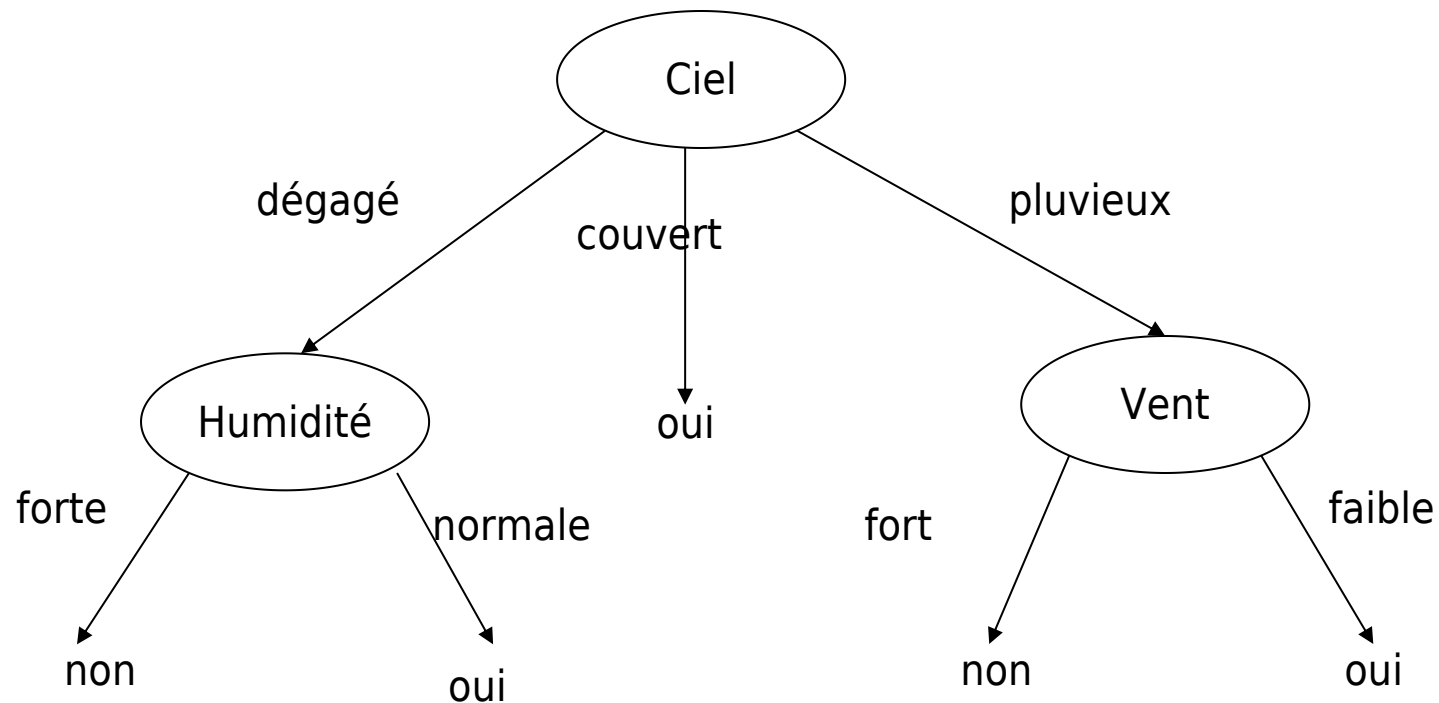
---

- Nettoyage des données
  - pré-traiter les données pour réduire le bruit et gérer les valeurs manquantes (si nécessaire)
- Analyse de pertinence
  - supprimer les attributs non pertinents ou redondants
- Transformation des données
  - généraliser ou normaliser les données
- Précision de la prédiction
- Efficacité et mise à l'échelle
  - pour construire le modèle
  - pour l'utiliser
- Robustesse
  - tolérance au bruit et aux données manquantes
- Interprétabilité
  - compréhension des données via le modèle
- Qualité des règles
  - taille de l'arbre de décision
  - règles de classification compactes



# Classification par arbre de décision

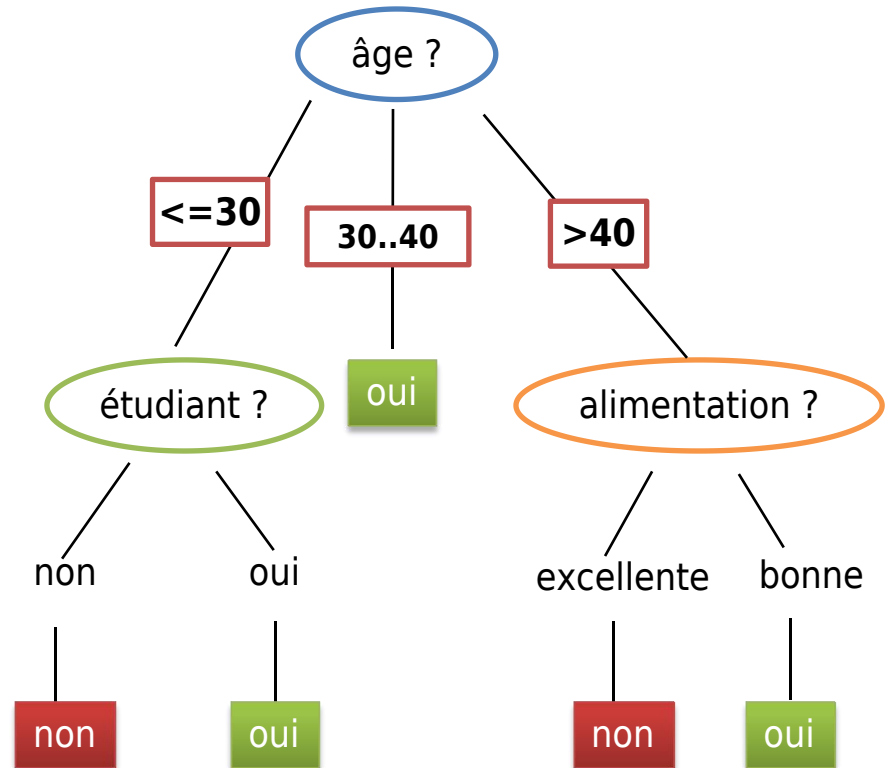
- Arbre de décision
  - nœuds internes : test sur un attribut
  - branches : résultat d'un test / valeur de l'attribut
  - feuilles : classe



- Génération de l'arbre en 2 étapes
  - Construction
    - au départ, tous les exemples du jeu d'apprentissage sont à la racine
    - partitionne récursivement les exemples en sélectionnant des attributs
  - Élagage
    - identification et suppression des branches correspondant à des exceptions ou du bruit
- Utilisation de l'arbre
  - teste les valeurs des attributs avec l'arbre de décision

# Exemple : pratique la méditation ?

âge	stress	étudiant	alimentation	médite
<=30	élevé	non	bonne	non
<=30	élevé	non	excellente	non
31...40	élevé	non	bonne	oui
>40	moyen	non	bonne	oui
>40	bas	oui	bonne	oui
>40	bas	oui	excellente	non
31...40	bas	oui	excellente	oui
<=30	moyen	non	bonne	non
<=30	bas	oui	bonne	oui
>40	moyen	oui	bonne	oui
<=30	moyen	oui	excellente	oui
31...40	moyen	non	excellente	oui
31...40	élevé	oui	bonne	oui
>40	moyen	non	excellente	non



# Algorithme pour l'induction d'arbre de décision

---

- Algorithme glouton
  - approche descendante réursive *diviser pour régner*
  - au départ, tous les objets sont à la racine
  - attributs catégoriels (les valeurs continues sont discrétisées à l'avance)
  - les exemples sont partitionnés récursivement par la sélection d'attribut
  - les attributs sont sélectionnés sur la base d'une heuristique ou d'une mesure statistique
- Conditions d'arrêt
  - tous les exemples pour un nœud appartiennent à la même classe
  - plus d'attribut pour partitionner, dans ce cas la classe attribuée correspond à celle la plus représentée
  - plus d'exemple à classer

## Mesure pour la sélection d'attribut. Exemple : gain d'information (ID3 et c4.5)

---

- Sélectionne l'attribut qui a le gain le plus élevé
- Soient 2 classes P et N
  - Soit un jeu d'apprentissage S qui contient p objets de classe P et n objets de classe N
  - La quantité d'information nécessaire pour décider si un objet de S appartient à P ou N est définie comme

$$I(p, n) = -\frac{p}{p+n} \log_2 \frac{p}{p+n} - \frac{n}{p+n} \log_2 \frac{n}{p+n}$$

- Les valeurs de A partitionnent S en  $\{S_1, \dots, S_v\}$ 
  - si  $S_i$  contient  $p_i$  exemples de P et  $n_i$  exemple de N, l'entropie ou l'information attendue nécessaire pour classer les objets dans tous les sous-arbres  $S_i$  est

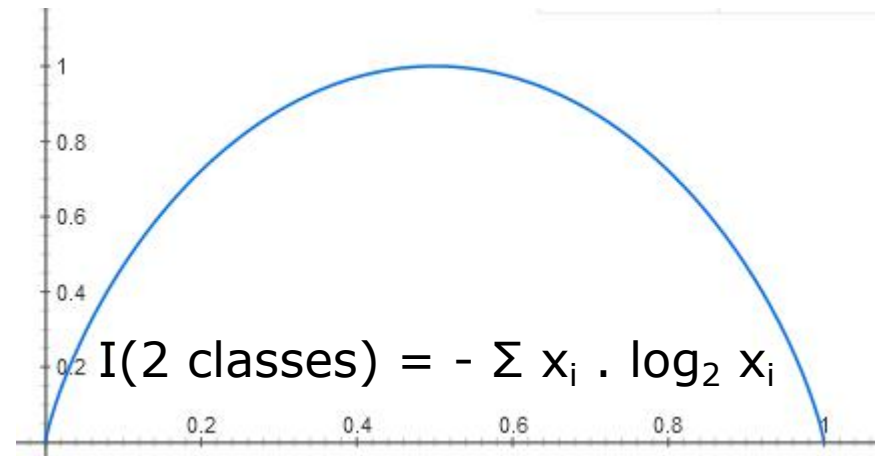
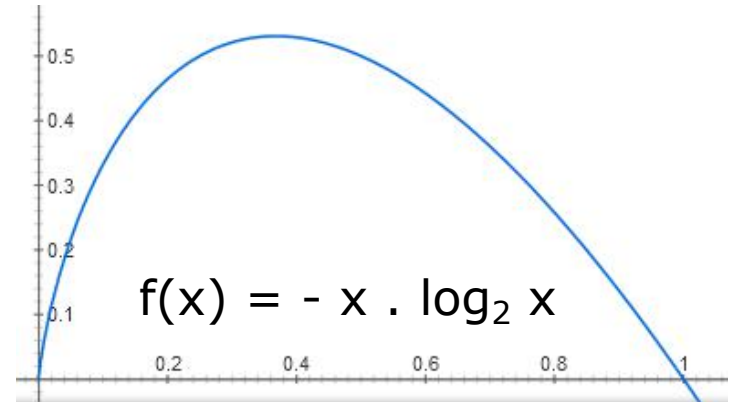
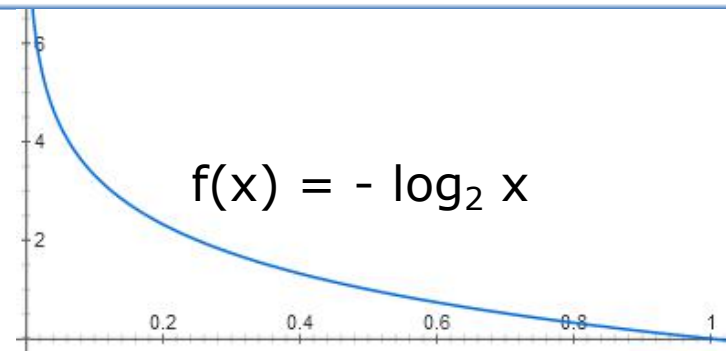
$$E(A) = \sum_{i=1}^v \frac{p_i + n_i}{p+n} I(p_i, n_i)$$

- Le gain d'information de l'attribut A est

$$Gain(A) = I(p, n) - E(A)$$

$$f(x) = -x \log x$$

- si  $f_1 = 0$  et  $f_2 = 1$   
 $I = 0$
- si  $f_1 = f_2 = 0.5$   
 $I = 0.5 + 0.5 = 1$
- si  $f_1 = 0.4$  et  
 $f_2 = 0.6$   
 $I = 0.53 + 0.44 = 0.97$



## Exemple : pratique la méditation ?

- ◆ Classe P : médite = oui
- ◆ Classe N : médite = non

âge	$p_i$	$n_i$	$I(p_i, n_i)$
$\leq 30$	2	3	0.971
30...40	4	0	0
$> 40$	3	2	0.971

- ◆  $I(p,n) = I(9,5) = 0.940$

- ◆ Calcul de l'entropie

$$E(age) = \frac{5}{14}I(2, 3) + \frac{4}{14}I(4, 0) + \frac{5}{14}I(3, 2) = 0.69$$

- ◆ Gain d'information

- ◆ De même

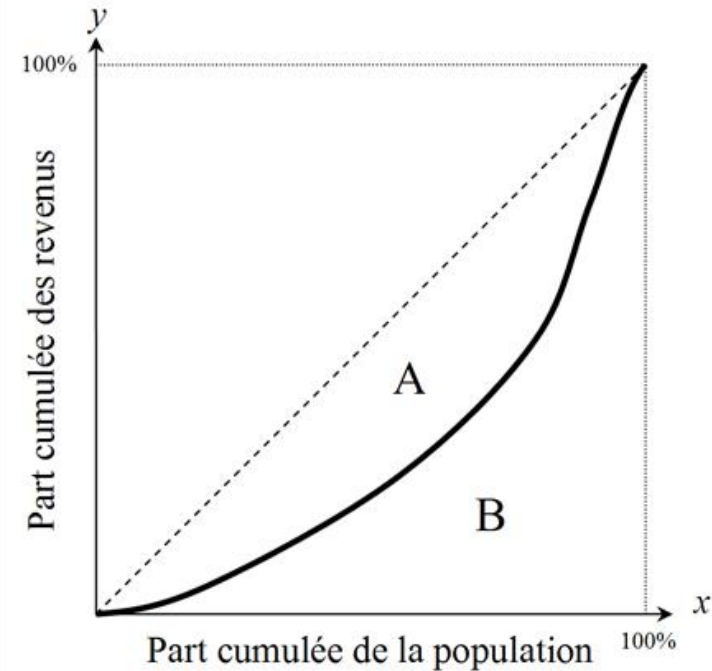
$$Gain(age) = I(p, n) - E(age) = 0.25$$

$$Gain(stress) = 0.029$$

$$Gain(etudiant) = 0.151$$

$$Gain(alimentation) = 0.048$$

- Coefficient de Gini
  - mesure la dispersion d'une distribution dans une population
  - 0 : uniforme, 1 : biaisée
  - utilisé pour mesurer l'inégalité de revenus
  - $G = A / (A+B)$
- $\chi^2$  : indépendance entre classe et modalités d'un attribut ?





## Éviter de trop modéliser le jeu d'apprentissage (overfitting)

---

- L'arbre généré risque de trop refléter le jeu d'apprentissage
  - trop de branches, certaines peuvent représenter des anomalies
  - précision faible pour des données nouvelles
- 2 approches
  - pré-élagage : arrêter la construction de l'arbre tôt = ne pas partitionner un nœud si la mesure de qualité dépasse un seuil
    - difficulté de fixer le seuil
  - post-élagage : supprimer des branches d'un arbre totalement construit = obtenir une séquence d'arbres progressivement élagués
    - utiliser un jeu de données différents pour décider du meilleur arbre élagué
    - principe MDL (minimum description length)
      - codage : règles associées à l'arbre + exceptions
      - supprimer la règle qui réduit le plus le codage tant que le codage diminue en taille

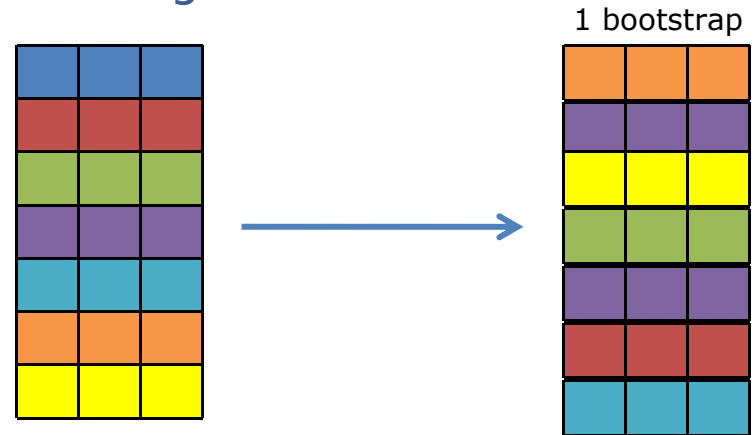
- Attributs définis sur des valeurs continues
  - définir dynamiquement les valeurs pour partitionner les données
  
- Tolérance aux données manquantes
  - attribuer la valeur la plus fréquente
  - attribuer une probabilité pour chaque valeur possible

# Extensions des arbres de décision

- Principal défaut : sensibilité et sur-apprentissage

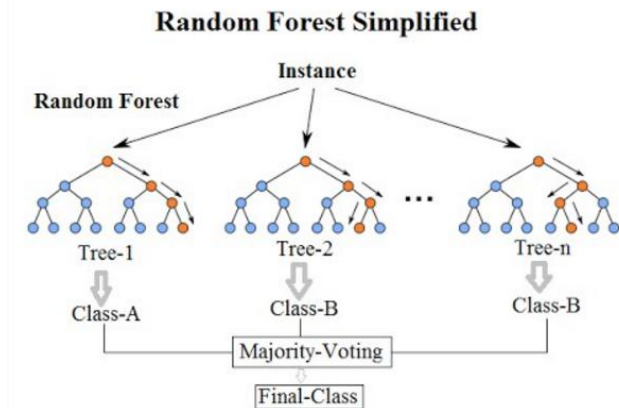
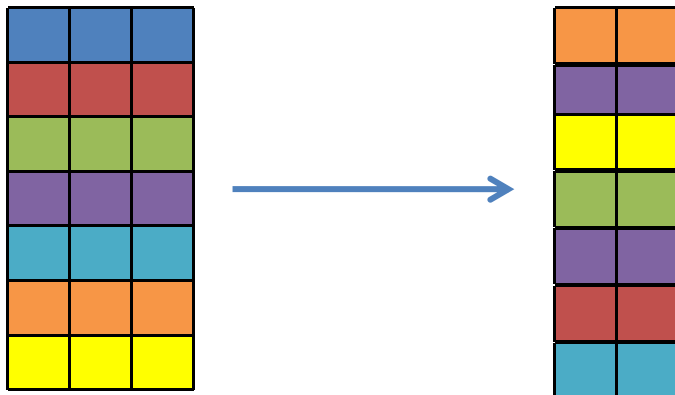
- Bagging (**bootstrap aggregating**)

- bootstrapping
- aggregating



- Forêts aléatoires

- échantillonnage des attributs pour décorréliser les arbres construits
- pas d'élagage : sur-apprentissage compensé par le bagging



source : wikipedia

# Évaluation d'un classificateur : performances

---

- **classificateur**
  - = modèle
  - = méthode + paramètres + jeu d'apprentissage
- **partitionnement**
  - utilisation de jeux indépendants :
    - apprentissage (2/3)
    - test (1/3)
- **validation croisée**
  - diviser les données en  $k$  partitions
  - utiliser  $k-1$  partitions pour l'apprentissage et la dernière pour le test
  - précision = nombre d'objets bien classés lors des  $k$  itérations / nombre d'objets
  - leave-one-out (validation croisée avec  $k = s$ )
- **bootstrapping**
  - tirage aléatoire avec remise des objets constituant le jeu d'apprentissage

# Performances du classificateur

Le plus simple : taux d'erreurs

2 classes : Positive (P) et Negative (N)

Taux d'erreurs =  $(FP+FN)/(P+N)$

Positif (P=●+●), Négatif (N=●+●),  
 Prédit Positif (PP=●+●), Prédit Négatif (PN=●+●),  
 Vrai Positif (TP=●), Faux Positif (FP=●),  
 Vrai Négatif (TN=●), Faux Négatif (FN=●)

**Sensibilité** =  $TP/P$  (eng: sensitivity, recall, TPR, ...)

**Spécificité** =  $TN/N$

**Précision** =  $TP / (TP+FP) = TP/PP = \mathbf{TPP}$

FDR =  $1 - TPP = FP / PP$

Val. Préd. Nég. =  $TN/PN$

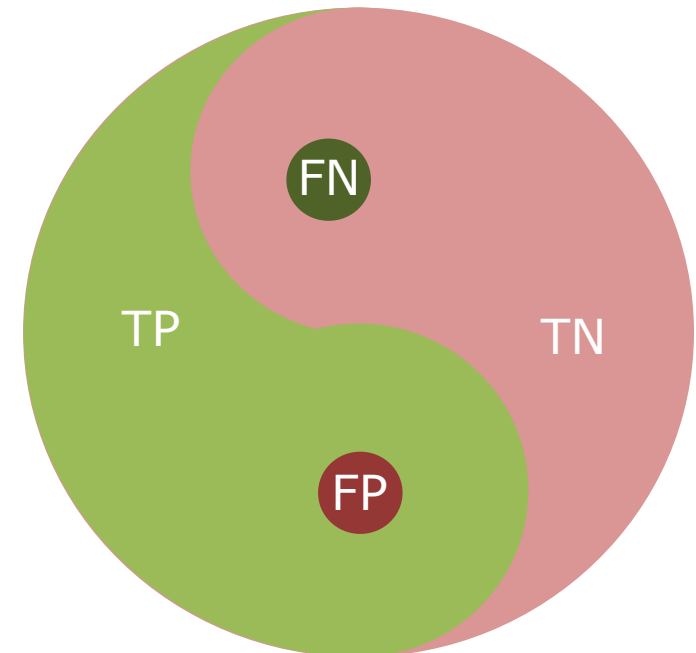
Exactitude (accuracy)

$$= \text{sensibilité} \cdot P / (P+N) + \text{spécificité} \cdot N / (P+N)$$

$$= \mathbf{(TP+TN)/(P+N)}$$

$$F_1 - \text{score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2TP}{2TP + FP + FN}$$

réalité → prédiction ↓	P	N	
PP	TP	FP	valeur prédictive positive
PN	FN	TN	
	sensibilité	spécificité	



# Performances du classificateur

3 classes (ou plus)

**Sensibilité** = TP/P (eng: sensitivity, recall, TPR)

**Précision** = TP / (TP+FP)

réalité → prédiction ↓	P	N
PP	TP	FP
PN	FN	TN

$$F_1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} = \frac{2TP}{2TP + FP + FN}$$

		réalité		
		banane	citron	orange
prédiction	banane	7	3	1
	citron	5	9	6
	orange	2	4	8

	Precision	Recall	F1-score	<i>n</i>
banane	0.64	0.5	0.56	14
citron	0.45	0.56	0.5	17
orange	0.57	0.53	0.55	15

bananes :

$$TP = 7 \quad ; \quad TN = 9 + 6 + 4 + 8 = 27$$

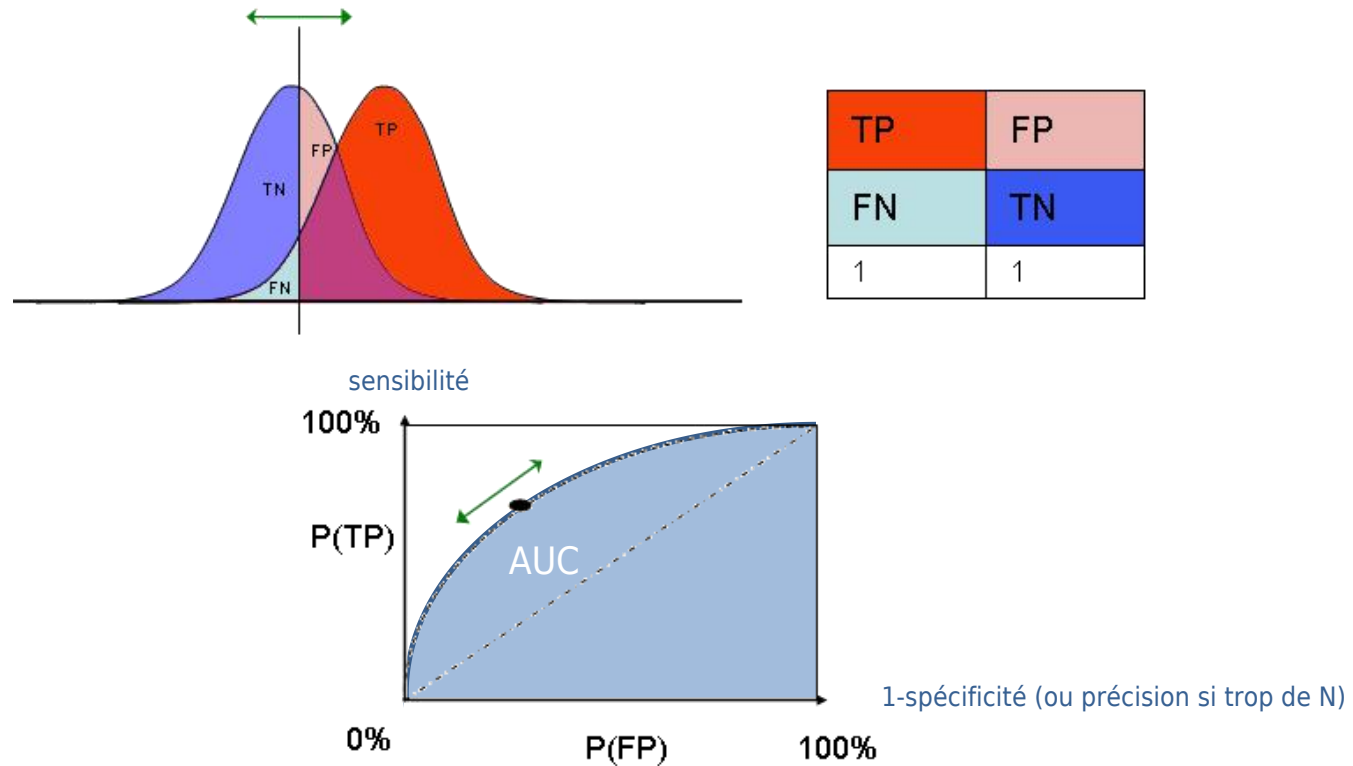
$$FP = 3 + 1 = 4 \quad ; \quad FN = 5 + 2 = 7$$

$$Precision = 7 / (7+4) = 0.64$$

$$Recall = 7 / (7+7) = 0.5$$

# Courbe ROC-AUC

- ROC pour Receiver operating characteristic (= courbe sensibilité/spécificité)
- AUC pour Area Under the Curve (= aire sous la courbe)
- 2 classes
- Principe : variation d'un seuil
  - score pour la présence d'un domaine
  - $\alpha$  pour un test statistique



- Apprentissage probabiliste : calcule explicitement les probabilités des hypothèses, une des approches les plus pragmatiques pour certains types d'apprentissage
- Incrémental : chaque exemple met à jour la probabilité qu'une hypothèse est correcte. Des connaissances *a priori* peuvent être combinées avec des données d'observation.
- Prédiction probabiliste : prédit plusieurs hypothèses, pondérées par leur probabilité



- Le problème de classification peut être formulé en utilisant les probabilités *a posteriori* :
  - $P(\mathbf{C}|X)$  = probabilité que l'objet  $X = \langle x_1, \dots, x_k \rangle$  est de la classe  $C$
- Exemple :  $P(\mathbf{non}|\text{ciel=dégagé, vent=fort, ...})$
- Idée : attribuer à l'objet  $X$  la classe qui maximise  $P(C|X)$

- Théorème de Bayes 
$$P(C|X) = \frac{P(X|C) \times P(C)}{P(X)}$$

$P(\text{classe} | \text{observations}) = P(\text{observations} | \text{classe}) \cdot P(\text{classe}) / P(\text{observations})$

- Principe : attribuer la classe la plus probable
  - $C_i$  tel que  $P(C_i|X)$  est maximum
  - $C_i$  tel que  $P(X|C_i) \cdot P(C_i) / P(X)$  est maximum
- remarque :  $P(X)$  est constant pour toutes les classes donc pas besoin de la calculer pour trouver la classe qui aura la plus forte probabilité
- $P(C) =$  fréquence de la classe  $C$
- Problème : rarement faisable en pratique

- Assomption naïve : indépendance des attributs

$$P(x_1, \dots, x_k | C) = P(x_1 | C) \times \dots \times P(x_k | C)$$

- si le  $i$ -ème attribut est catégoriel :  
 $P(x_i | C)$  est estimée comme la fréquence des échantillons qui ont pour valeur  $x_i$  et qui sont de classe  $C$
- si le  $i$ -ème attribut est continu :  
 $P(x_i | C)$  est estimée avec une gaussienne
- calcul facile

## Exemple

Ciel	Température	Humidité	Vent	Classe
dégagé	chaud	forte	faux	N
dégagé	chaud	forte	vrai	N
couvert	chaud	forte	faux	P
pluvieux	moyenne	forte	vrai	P
pluvieux	frais	normale	faux	P
pluvieux	frais	normale	vrai	N
couvert	frais	normale	vrai	P
dégagé	moyenne	forte	faux	N
dégagé	frais	normale	faux	P
pluvieux	moyenne	normale	faux	P
dégagé	moyenne	normale	vrai	P
couvert	moyenne	forte	vrai	P
couvert	chaud	normale	faux	P
pluvieux	moyenne	forte	vrai	N

Un objet  $X = \langle \text{pluvieux, chaud, forte, faux} \rangle$  ?

$$\begin{aligned}
 P(X|p) \cdot P(p) &= P(\text{pluvieux}|p) \cdot P(\text{chaud}|p) \cdot P(\text{forte}|p) \cdot P(\text{faux}|p) \cdot P(p) \\
 &= \frac{3}{9} \cdot \frac{2}{9} \cdot \frac{3}{9} \cdot \frac{6}{9} \cdot \frac{9}{14} = 0.010582 \\
 P(X|n) \cdot P(n) &= P(\text{pluvieux}|n) \cdot P(\text{chaud}|n) \cdot P(\text{forte}|n) \cdot P(\text{faux}|n) \cdot P(n) \\
 &= \frac{2}{5} \cdot \frac{2}{5} \cdot \frac{4}{5} \cdot \frac{2}{5} \cdot \frac{5}{14} = \mathbf{0.018286}
 \end{aligned}$$

X est classé comme n

Ciel	
$P(\text{dégagé}   p) = 2/9$	$P(\text{dégagé}   n) = 3/5$
$P(\text{couvert}   p) = 4/9$	<b><math>P(\text{couvert}   n) = 0</math></b>
$P(\text{pluvieux}   p) = 3/9$	$P(\text{pluvieux}   n) = 2/5$
température	
$P(\text{chaud}   p) =$	$P(\text{chaud}   n) =$
$P(\text{moyenne}   p) =$	$P(\text{moyenne}   n) =$
$P(\text{frais}   p) =$	$P(\text{frais}   n) =$
humidité	
$P(\text{forte}   p) = 3/9$	$P(\text{forte}   n) = 4/5$
$P(\text{normale}   p) = 6/9$	$P(\text{normale}   n) = 1/5$
vent	
$P(\text{vrai}   p) = 3/9$	$P(\text{vrai}   n) = 3/5$
$P(\text{faux}   p) = 6/9$	$P(\text{faux}   n) = 2/5$
classe	
$P(p) = 9/14$	$P(n) = 5/14$

- ... rend le calcul possible
- ... mène à des classificateurs optimaux si elle est vérifiée
- ... mais c'est rarement le cas, car les attributs sont souvent corrélés
- tentative de pallier cette difficulté :
  - réseaux Bayésiens : combinent le raisonnement Bayésien avec la relation causale entre les attributs
  - arbres de décision : considère un seul attribut à la fois, en commençant par le plus important

- Théorème de Bayes

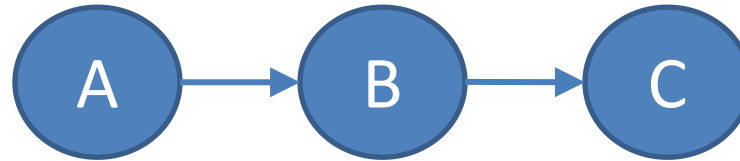
$$P(A|B) = \frac{P(B|A) P(A)}{P(B)}.$$

- $A = (a_1, a_2, \dots, a_n)$
- $B = (b_1, b_2, \dots, b_n)$
- Pas faisable avec des milliers de variables
- Hypothèse de l'indépendance **ou**
- prise en compte partielle par modèle réseau
  - modélisation de l'influence d'un facteur de transcription sur ces cibles

## Réseaux bayésiens

- Niveau d'expression de B dépend de celui de A (influence)

- $P(B/A)$



- Le niveau d'expression de C
  - dépend de celui de B
  - est indépendant de celui de A sachant B

	cond1	cond2	cond3	cond4	cond5
A	off	low	high	low	low
B	off	low	high	high	low
C	off	off	low	high	low

observations

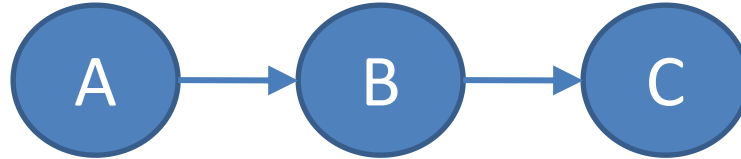


$P(B/A)$

		$P(B/A)$		
		A	off (1)	low (3)
B	off	1/1	0/3	0/1
	low	0/1	2/3	0/1
	high	0/1	1/3	1/1

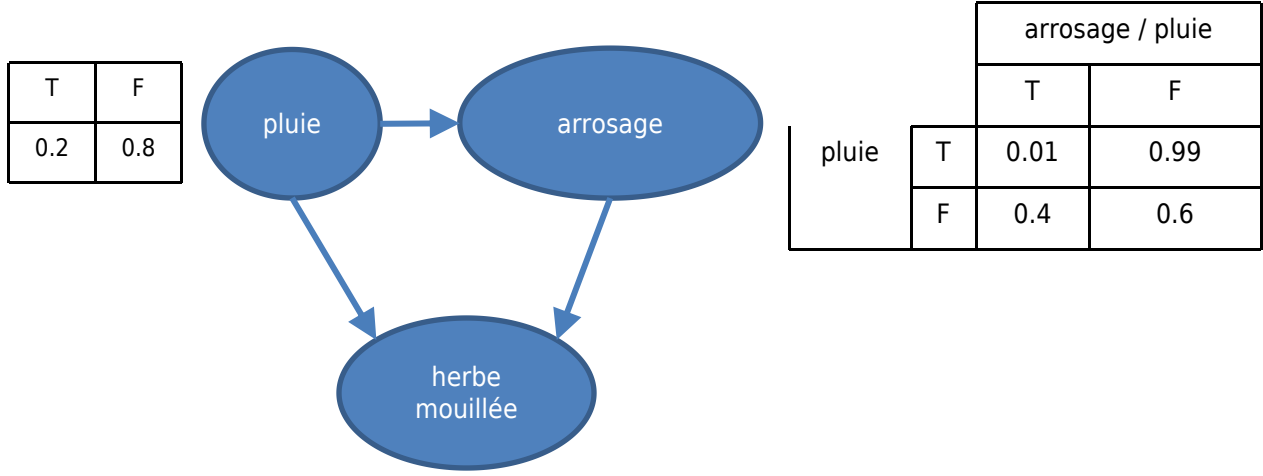
- Niveau d'expression de B dépend de celui de A (influence)

- $P(B/A)$



- Le niveau d'expression de C
  - dépend de celui de B
  - est indépendant de celui de A sachant B
- Avantages :
  - capture l'aspect stochastique de la régulation
  - possibilité d'intégrer des régulations connues
  - peu de sur-apprentissage et robustesse
  - quantitatif (niveau d'expression) ou qualitatif (on/off)
- Inconvénient : Pas de cycle donc pas de boucle d'auto-régulation





T	F
0.2	0.8

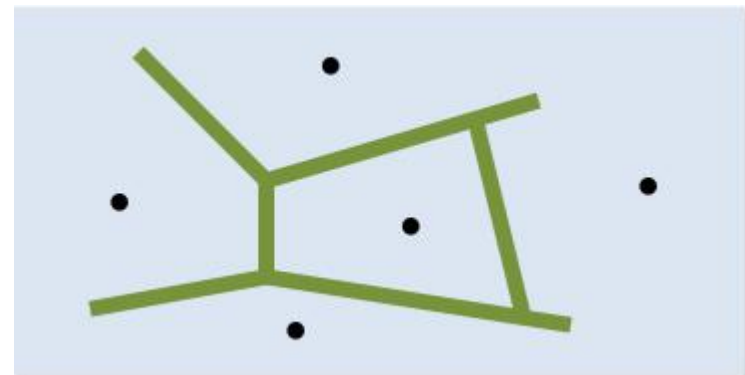
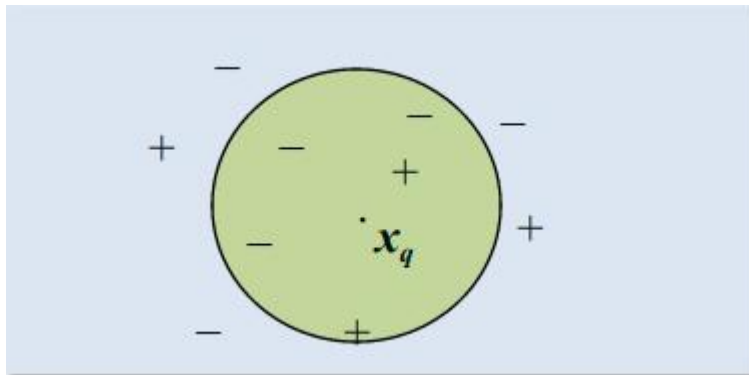
		arrosage / pluie	
		T	F
pluie	T	0.01	0.99
	F	0.4	0.6

		mouillée	
		T	F
arrosage	pluie		
	F	0	1
	T	0.8	0.2
	F	0.9	0.1
T	0.99	0.01	

- Apprentissage basé sur des instances :
  - Stocke le jeu d'apprentissage et effectue le traitement quand une nouvelle instance doit être classée
- Approches typiques
  - $k$  plus proches voisins ( $k$  nearest neighbors)
    - chaque objet représente un point dans l'espace
  - régression
    - loess, approximation locale
- Inférence des valeurs manquantes

## Algorithme des $k$ plus proches voisins

- Chaque objet est représenté par un point dans un espace à  $n$  dimensions
- Les plus proches voisins sont définis en terme de distance (euclidienne, manhattan, ...) ou dissimilarité
- La fonction de prédiction peut être discrète/nominale ou continue
  - discrète : valeur la plus fréquente
  - continue : moyenne
- Diagramme de Voronoï : surface de décision induite par le plus proche voisin



- Pondération en fonction de la distance
  - pondère la contribution de chacun des  $k$  voisins en fonction de sa distance à l'objet à classer
  - plus on est proche, plus on a de poids
- Robuste dans le cas de données bruitées
- Problème de dimensionnalité : la distance peut être dominée par des attributs non pertinents
  - solution : normalisation des dimensions ou élimination des attributs non pertinents

# Types de données

---

- qualitatives ou nominales :
  - qualitatives
  - binaires, logiques
  - énumérations, facteurs
- numériques :
  - quantitatives
    - discrètes : entiers
    - continues
      - continues sur un intervalle
      - échelle linéaire, logarithmique, exponentielle
    - nombres complexes
  - ordinales, temporelles
  - géométriques, spatiales
- textuelles, sémantiques, ontologies
- mixtes

- table de contingence

		Objet $j$	
		1	0
Objet $i$	1	$a$	$b$
	0	$c$	$d$

- coefficient simple d'appariement (invariant, si la variable est symétrique)

$$d(i, j) = \frac{b + c}{a + b + c + d}$$

- coefficient de Jaccard (non invariant, si la variable est asymétrique)

$$d(i, j) = \frac{b + c}{a + b + c}$$

# Dissimilarité de valeurs binaires

- Exemple

Nom	Sexe	Fièvre	Tousse	Test-1	Test-2	Test-3	Test-4
Jacques	M	O	N	P	N	N	N
Marie	F	O	N	P	N	P	N
Jean	M	O	P	N	N	N	N

- sexe est symétrique
- les autres sont asymétriques
- soit O et P = 1, et N = 0

$$d(\text{jacques}, \text{marie}) = \frac{0 + 1}{2 + 0 + 1} = 0.33$$

$$d(\text{jacques}, \text{jean}) = \frac{1 + 1}{1 + 1 + 1} = 0.66$$

$$d(\text{jean}, \text{marie}) = \frac{1 + 2}{1 + 1 + 2} = 0.75$$

- généralisation des valeurs binaires : plus de 2 états
- méthode 1 : appariement simple
  - $m$  : nombre d'appariements,  $p$  : nombre total de variables

$$d(i, j) = \frac{p - m}{p}$$

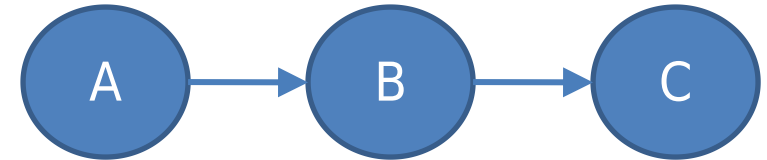
- méthode 2 : utiliser un grand nombre de variables binaires
  - création d'une variable binaire pour chacun des états d'une variable nominale
- Information mutuelle

$$I(X, Y) = \sum_{y \in Y} \sum_{x \in X} \log \frac{p(x, y)}{p(x)p(y)}$$



## Information mutuelle

$$I(X, Y) = \sum_{y \in Y} \sum_{x \in X} \log \frac{p(x, y)}{p(x)p(y)}$$



observations

	cond1	cond2	cond3	cond4	cond5
A	high	low	high	low	low
B	high	low	high	high	low



	b = low	B = high	total
a = low	2 / 5	1 / 5	3 / 5
A = high	0	2 / 5	2 / 5
total	2 / 5	3 / 5	1



```

ab  ab  a b  aB  aB  a B  Ab  AB  AB  A  B
> .4*log( .4 / (.6*.4) ) + .2*log( .2 / (.6*.6) ) + 0 + .4*log( .4 / (.4*.6) )
[1] 0.2911032

```

• Profils phylogénétiques

Strain

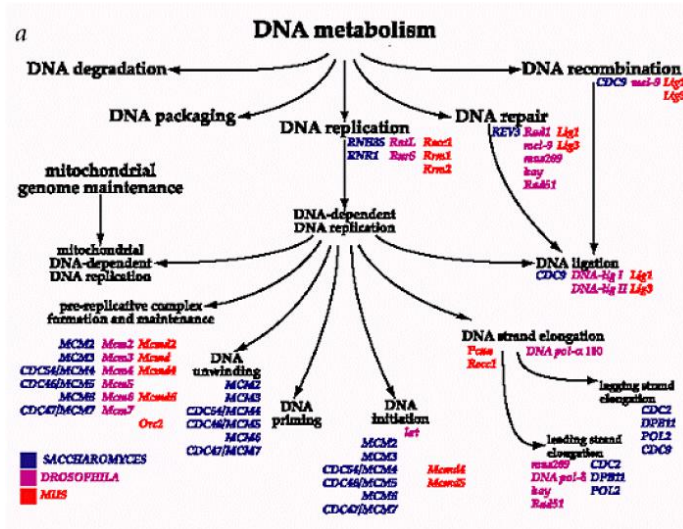
Gene(s)

gènes



génomés

• Annotations, ex: Gene Ontology



gene x term matrix

	term 1	...	term m
gene 1	✓	✓	✓
⋮	✓		
gene n		✓	✓

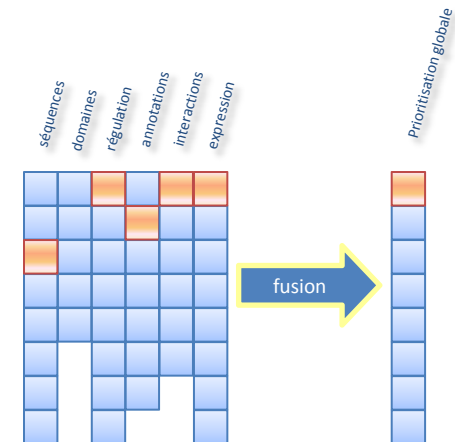
## Variable ordinale

- l'ordre est important : rang
- peut être traitée comme une variable continue sur un intervalle
  - remplace  $x_{if}$  par son rang
  - transforme chaque variable sur  $[0,1]$  en remplaçant le  $i$ -ème objet de la  $f$ -ème variable

$$r_{if} = \{1, \dots, M_f\}$$

- calcule la dissimilarité en utilisant les méthodes de valeurs continues sur un intervalle

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$



# Valeurs continues sur un intervalle, Fonction de distance

- Distance de Minkowski :

$$d(i, j) = \sqrt[q]{|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + \dots + |x_{ip} - x_{jp}|^q}$$

avec  $i = (x_{i1}, x_{i2}, \dots, x_{ip})$  et  $j = (x_{j1}, x_{j2}, \dots, x_{jp})$  deux objets à  $p$  dimensions, et  $q$  un entier positif

- si  $q = 1$  : distance de Manhattan (ou city block distance)

$$d(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \dots + |x_{ip} - x_{jp}|$$

- si  $q = 2$  : distance euclidienne

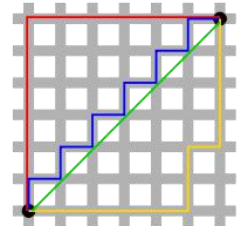
$$d(i, j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \dots + (x_{ip} - x_{jp})^2}$$

- Propriétés

- $d(i, i) = 0$
- $d(i, j) \geq 0$  (positive)
- $d(i, j) = d(j, i)$  (symétrique)
- $d(i, j) \leq d(i, k) + d(k, j)$  (inégalité triangulaire)

- Dissimilarité basée sur un coefficient de corrélation

- Pearson, Spearman (rangs)
- $d(x, y) = 1 - \text{corr}(x, y)$



source : wikipedia

- Distance de Canberra ( $\sim$  Manhattan pondérée)

$$d(p, q) = \sum_{i=1}^n \frac{|p_i - q_i|}{|p_i| + |q_i|}$$

- Similarité = cosinus de l'angle formé par les 2 vecteurs
- Distance de Mahalanobis
  - distance d'un point à un ensemble
  - $x$  : vecteur/point
  - $S$  : matrice de variance-covariance

$$d(x) = \sqrt{(x - \mu)^T S^{-1} (x - \mu)}$$

- mesure positive sur une échelle non linéaire, échelle exponentielle qui suit approximativement  $Ae^{BT}$  ou  $Ae^{-BT}$
- Méthodes
  - les traiter comme des variables continues sur un intervalles : mauvais choix
  - appliquer une transformation logarithmique puis les traiter comme des variables continues sur un intervalle

$$y_{if} = \log x_{if}$$

- les traiter comme des variables ordinales en traitant leur rang

- Les objets peuvent être décrits avec tous les types de données
  - binaire symétrique, binaire asymétrique, nominale, ordinale, ...
- Utilisation d'une formule pondérée pour combiner leurs effets

$$d(i, j) = \frac{\sum_{k=1}^p w_k d_k(i, j)}{\sum_{k=1}^p w_k}$$

- **min-max** : transformation depuis l'intervalle  $[\min, \max]$  vers l'intervalle  $[a, b]$

$$v' = \frac{v - \min}{\max - \min}(b - a) + a$$

- **z-score** :
- $$v' = \frac{v - \mu}{\sigma}$$

- **mise à l'échelle décimale**

$$v' = \frac{v}{10^n} \quad \text{avec } n \text{ le plus petit entier tel que } \max(|v'|) < 1$$



- Normaliser les données : s'affranchir des unités de mesures

- écart absolu à la moyenne

$$s = \frac{|x_1 - \mu| + |x_2 - \mu| + \dots + |x_n - \mu|}{n}$$

- Calculer la mesure normalisée (z-score)

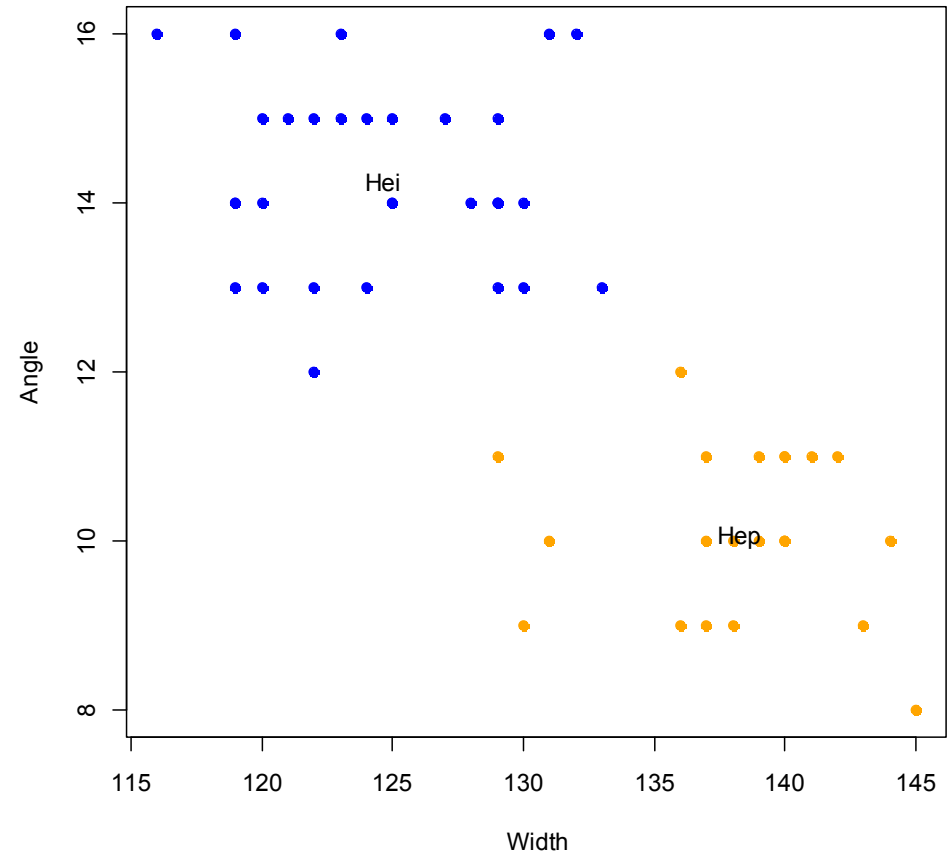
$$z_i = \frac{x_i - \mu}{s}$$

- L'utilisation de l'écart absolu est plus robuste que celle de l'écart type

## Analyse discriminante

2 familles de puces caractérisées par l'angle et la taille de leur édéage :

- Hei : Heikertingeri
- Hep : Heptapotamica

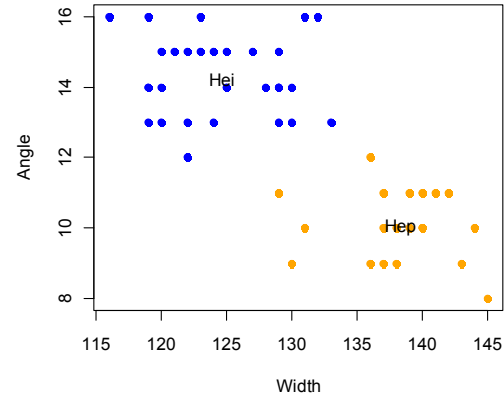


**Objectif :** combinaison linéaire des variables permettant de séparer au mieux les groupes dans un nouvel espace de représentation

# Analyse discriminante

2 familles de puces  
caractérisées par l'angle  
et la taille de leur édéage :

- Hei : Heikertingeri
- Hep : Heptapotamica



**Objectif** : combinaison linéaire des variables permettant de séparer au mieux les groupes dans un nouvel espace de représentation

**dispersion intra-groupe** : matrice de variance-covariance  $W_k$   
avec  $n$  le nombre d'individus et  $n_k$  le nombre d'individus de la classe  $k$

$$W = \frac{1}{n} \sum_k n_k \times W_k$$

**éloignement entre les groupes** : matrice de variance-covariance inter-groupes :  $B$   
avec  $\mu$  la moyenne globale et  $\mu_k$  la moyenne de la classe  $k$

$$B = \frac{1}{n} \sum_k n_k (\mu_k - \mu)^T (\mu_k - \mu)$$

**dispersion totale** :  $V = W + B$

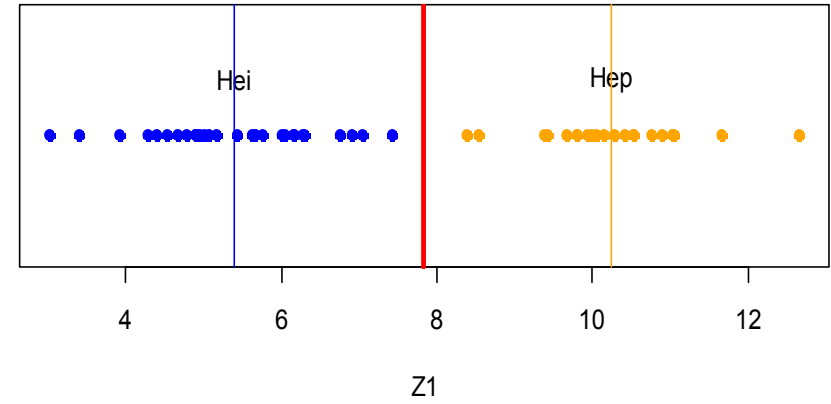
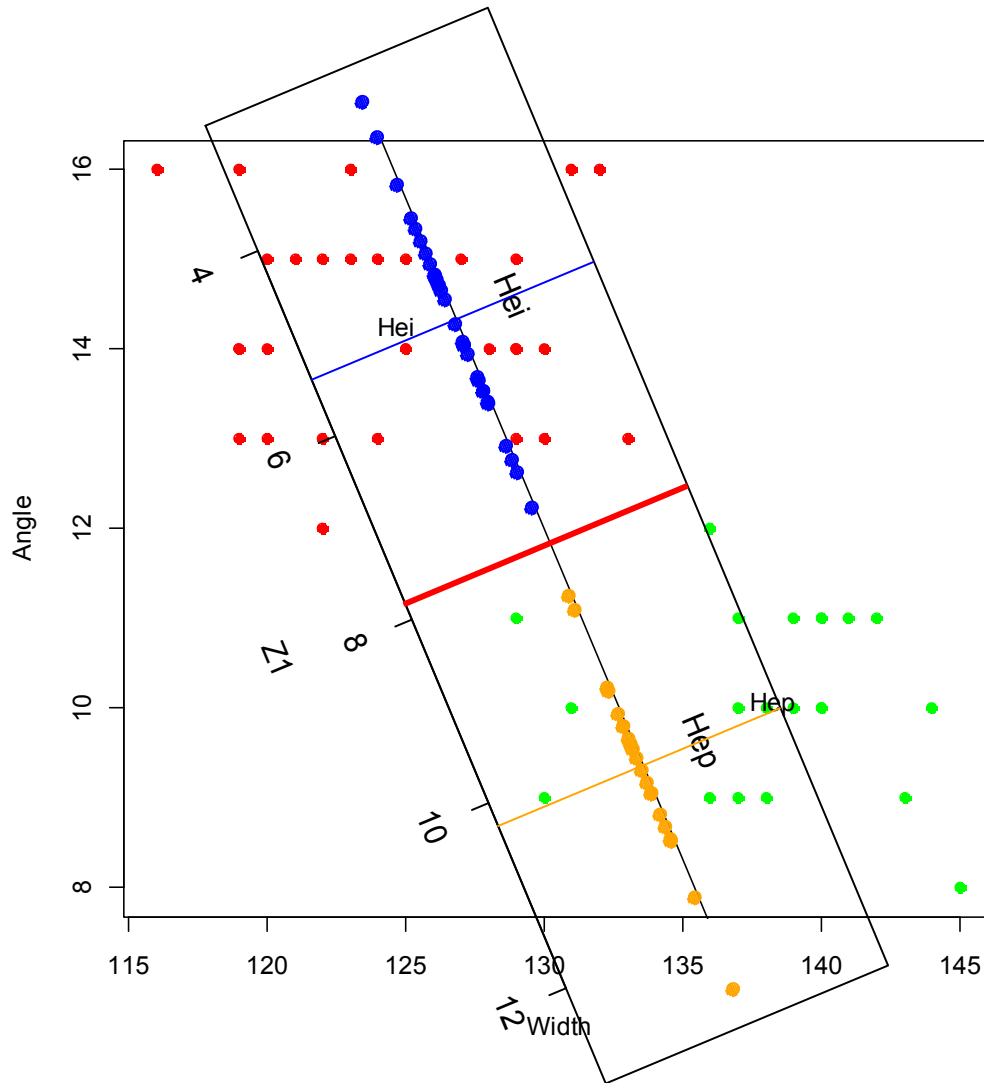
**principe** : trouver les axes factoriels qui maximisent l'éloignement des groupes par rapport à la dispersion totale

$Z_1$  premier axe factoriel défini par le vecteur  $u_1$  tel que l'on maximise

$$\frac{u_1^T B u_1}{u_1^T V u_1}$$

**Solution** : résolution de  $V^{-1} B u = \lambda u$

# Analyse discriminante



	u1
Width	0.1280690
Angle	-0.7393077

## Règle bayésienne

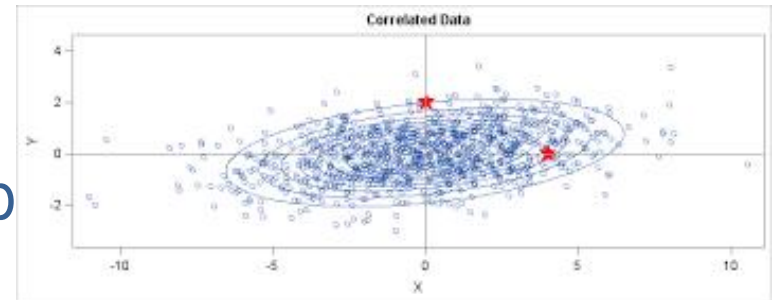
$$P(C = c|X) = \frac{P(C = c) \times P(X|C = c)}{\sum_{i=1}^k P(C = c_i) \times P(X|C = c_i)}$$

Hypothèse homoscedasticité :

$$W = \frac{1}{n} \sum_k n_k \times W_k$$

Fonction de classement linéaire :

~ distance au centre de gravité qui p  
variance

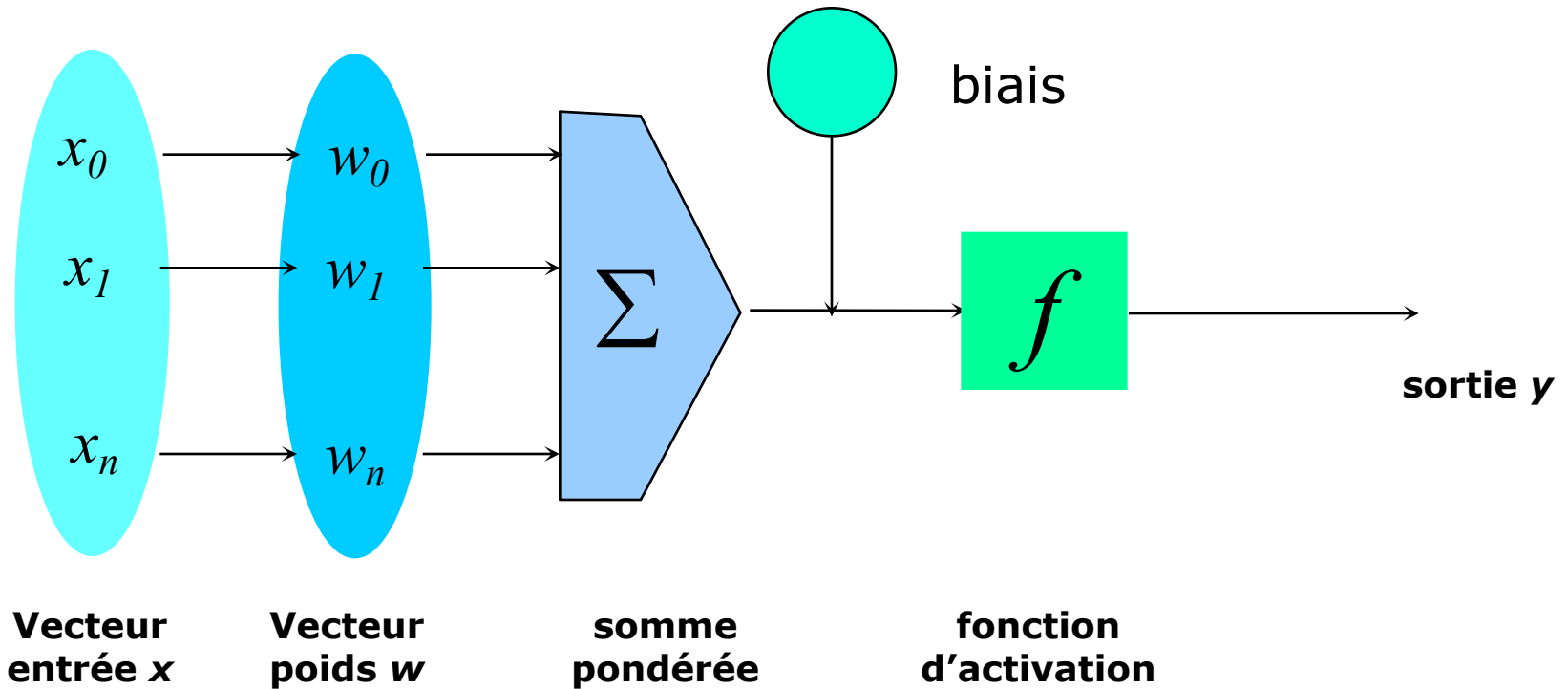


$$\text{score}(C = c, X) = -(X - \mu_c)^T W^{-1} (X - \mu_c)$$

- Pouvoir discriminant d'un axe factoriel
  - variance portée par l'axe / valeur propre
- Robustesse
  - peut fonctionner même si les hypothèses de départ de ne sont pas respectées

# Un neurone

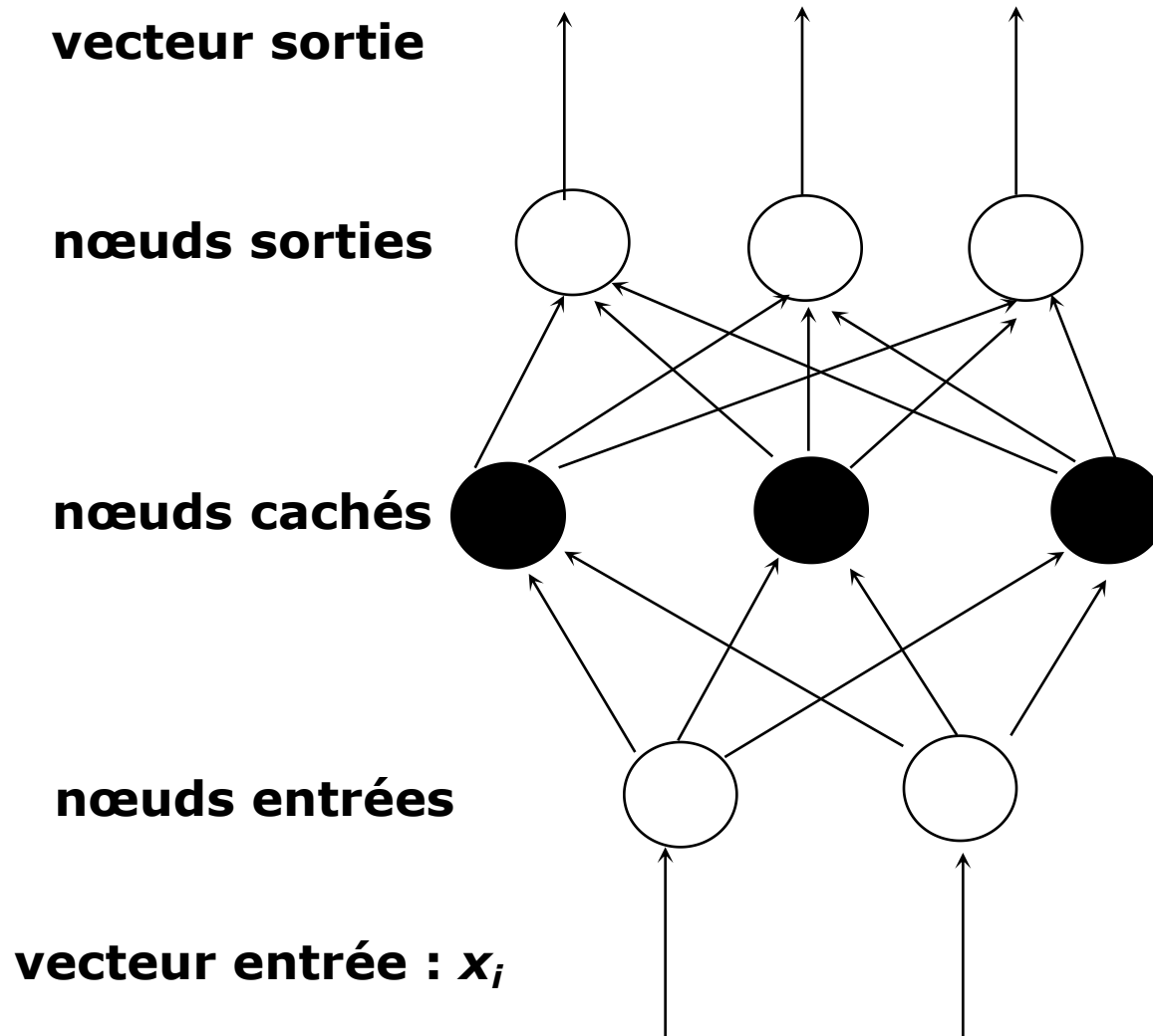
- vecteur  $x$   $n$ -dimensionnel est intégré en  $y$  par le produit scalaire ( $x_i \cdot w_i$ ), le biais et la fonction d'activation



- 
- Objectif
    - obtenir un ensemble de poids permettant de classer correctement (presque tous) les objets du jeu d'apprentissage
  - Étapes
    - Initialiser les poids avec des valeurs aléatoires
    - Passer les objets au réseau un par un
    - pour chaque unité (neurone)
      - calculer l'entrée (combinaison linéaire de toutes les entrées)
      - calculer la sortie en utilisant la fonction d'activation
      - calculer l'erreur
      - mettre à jour le biais et les poids



# Perceptron (multicouche) (eng: MLP pour multilayer perceptron)



1. calcul de l'erreur en sortie pour chaque neurone
2. augmentation ou diminution du biais et du poids pour obtenir une faible erreur locale
3. attribution d'une erreur au neurones précédents
4. retour à l'étape 2 pour la couche précédente

- Avantages
  - précision souvent élevée
  - robuste, marche lorsque le jeu d'apprentissage contient des erreurs
  - sortie peut être valeur discrète, continue, ou un vecteur de plusieurs valeurs discrètes ou continues
- Critiques
  - apprentissage long
  - difficile de comprendre le modèle
  - difficile d'incorporer des connaissances