Sciences des données

Un voyage initiatique

Cécile CAPPONI, Rémi EYRAUD, Hachem KADRI

LIS, Aix-Marseille Université, CNRS
Equipe QARMA

M1 Informatique

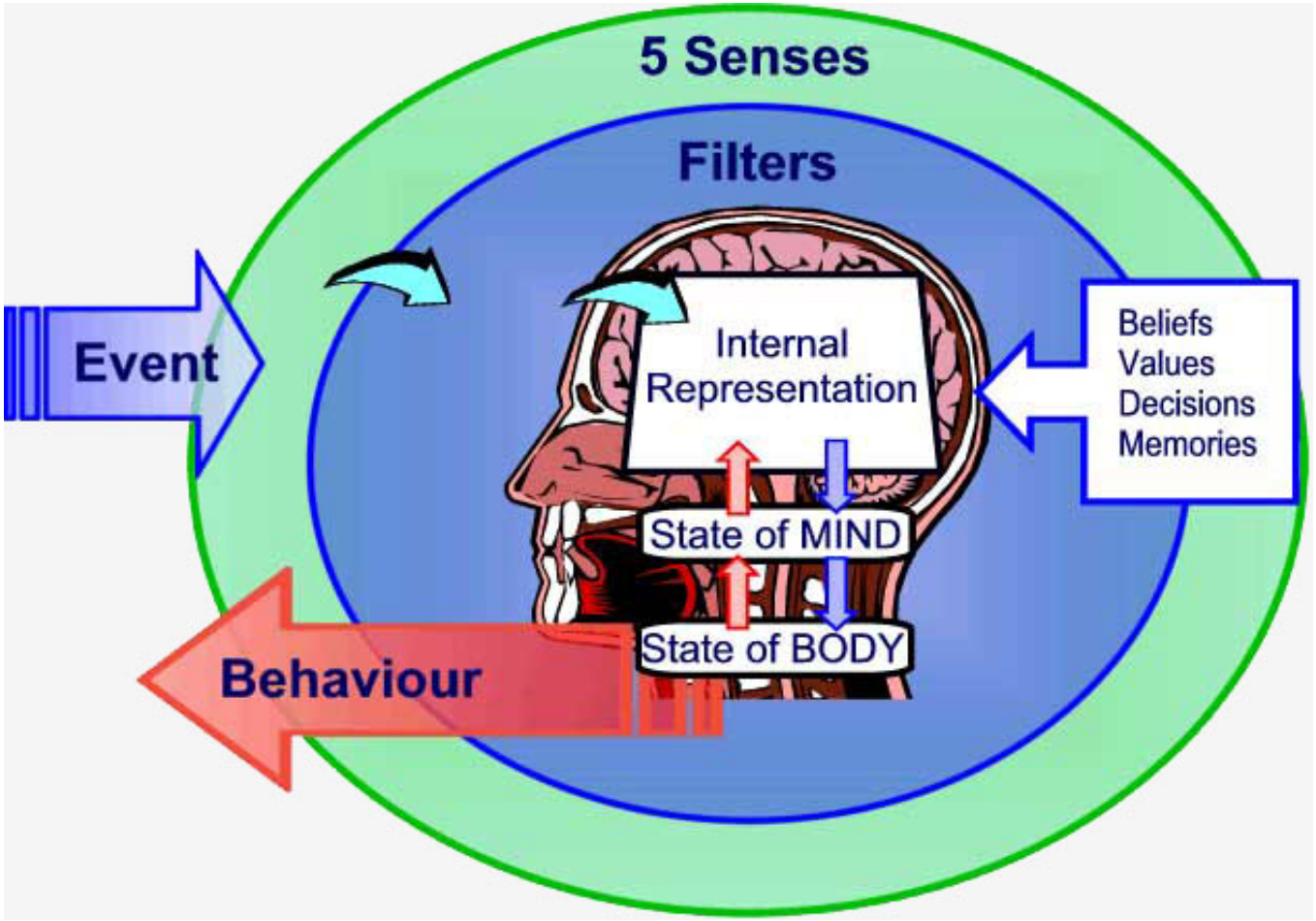
Outline

Outline

La classification : au coeur de l'apprentissage automatique

La classification : au coeur de l'apprentissage automatique

Construction de modèles de prédiction *automatique*

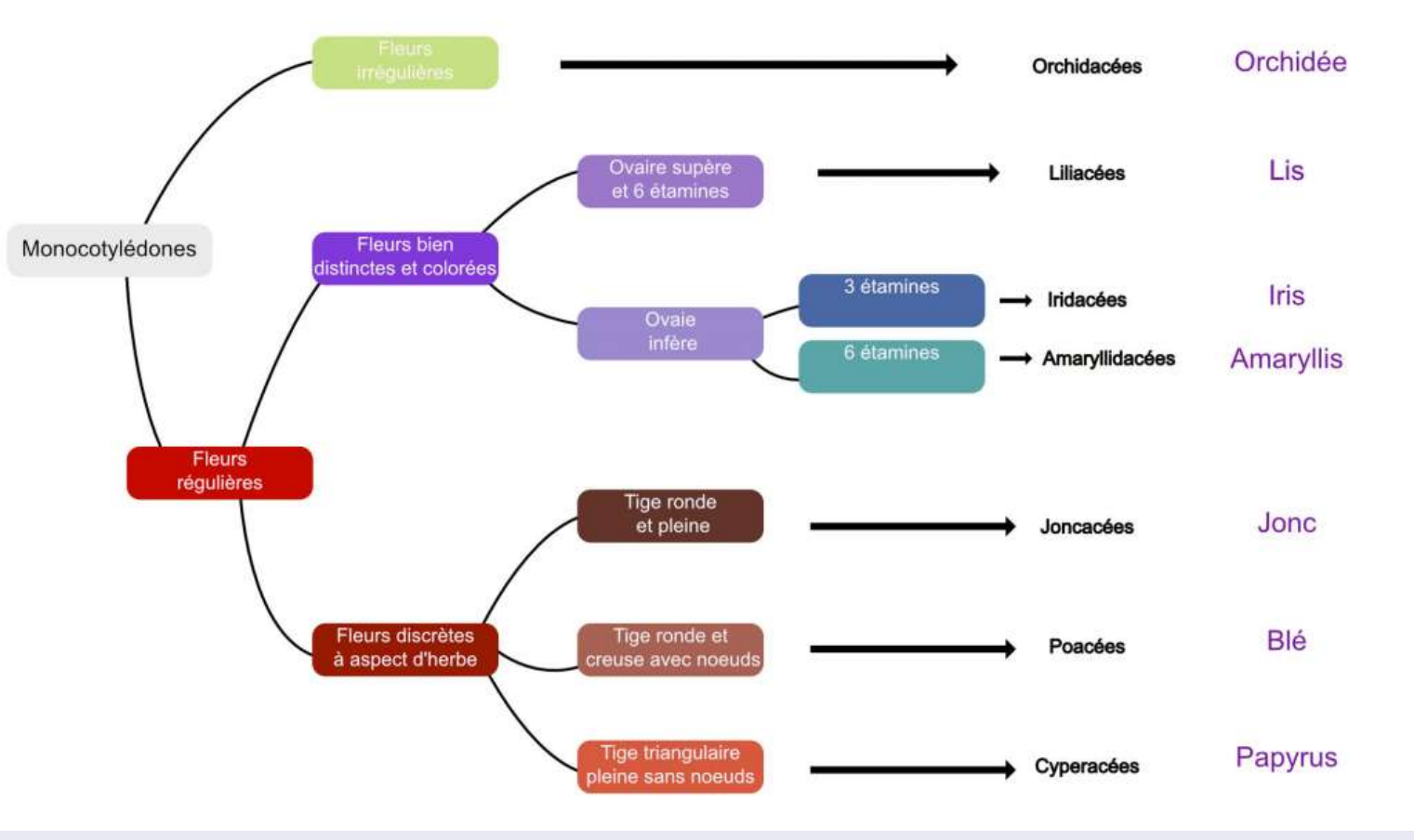


(source : i-change.biz)

La classification : au coeur de l'apprentissage automatique

Construction de modèles de prédiction *automatique*

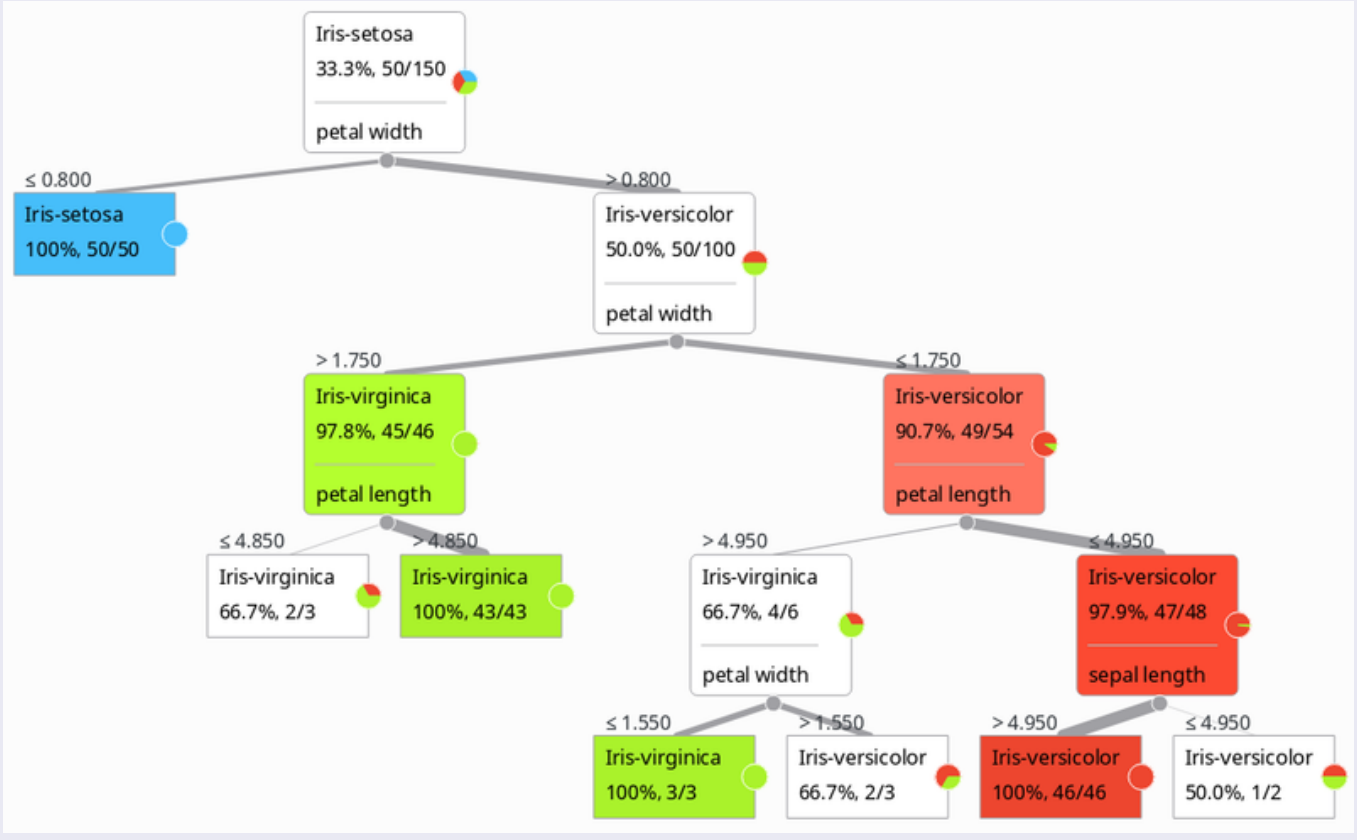
Construction manuelle de règles programmables (expertise)



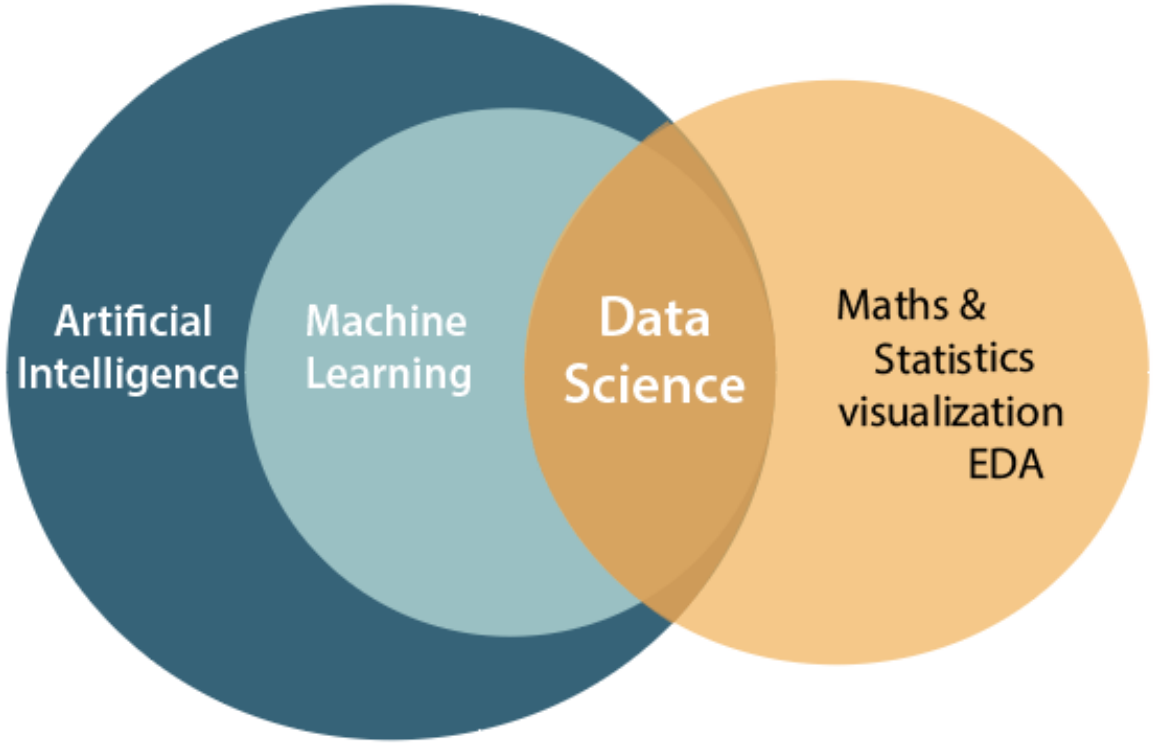
La classification : au coeur de l'apprentissage automatique

Construction de modèles de prédiction *automatique*

Construction automatique à partir d'observations (induction)



Science des données et apprentissage automatique



Apprentissage machine = un moteur de la science des données

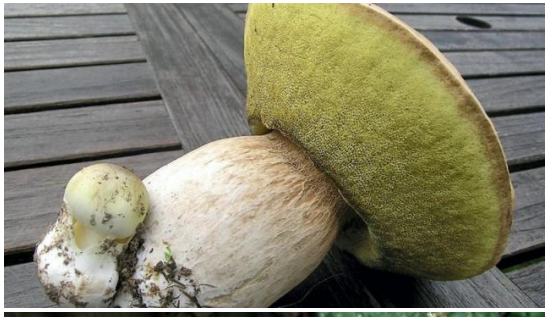
- Pour aller plus loin que les statistiques descriptives
- Objectifs : algorithmes pour construire des modèles numériques prédictifs à partir des données observées, qui généralisent bien à des données futures

ALGO – DONNÉES – MODÈLE – PRÉDICTION – GÉNÉRALISATION

Outline

Classer des champignons : comestibles ou vénéneux ?






p	x	s	n	t	...	k	s	u
e	x	s	y	t	...	n	n	g
e	b	s	?	t	...	n	n	m
...	...	...	...	...	...	...	...	...
p	x	y	w	t	...	k	s	u
e	x	y	y	t	...	k	n	g

Trouver la meilleure

$$h : \mathcal{X} \to \{\text{e}, \text{p}\}$$

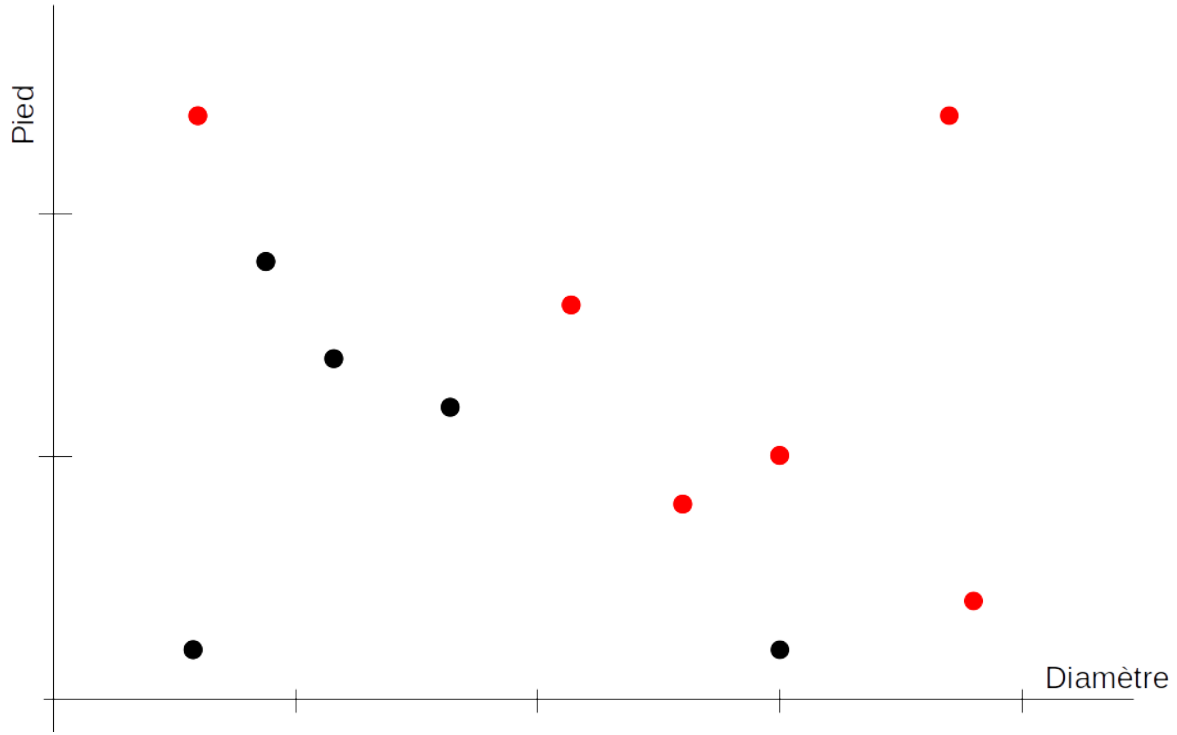
A partir de

$$S = \{(x_i, y_i)\}, x_i \in \mathcal{X}, y_i \in \{\text{e}, \text{p}\}$$

Classer des champignons : comestible ou vénéneux ?

Observations : comestibilité versus (Diamètre, Hauteur du pied)

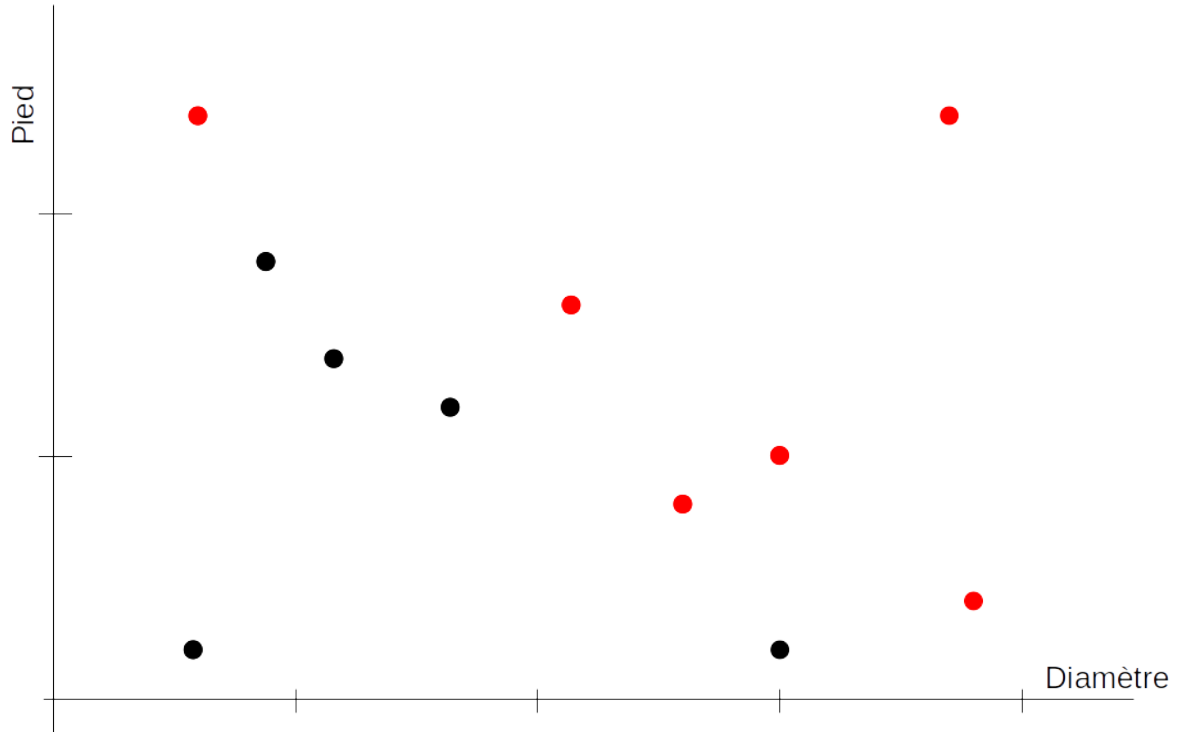
$$(176, 12), \cdots, (112, 8), (34, 1), \cdots$$



Classer des champignons : comestible ou vénéneux ?

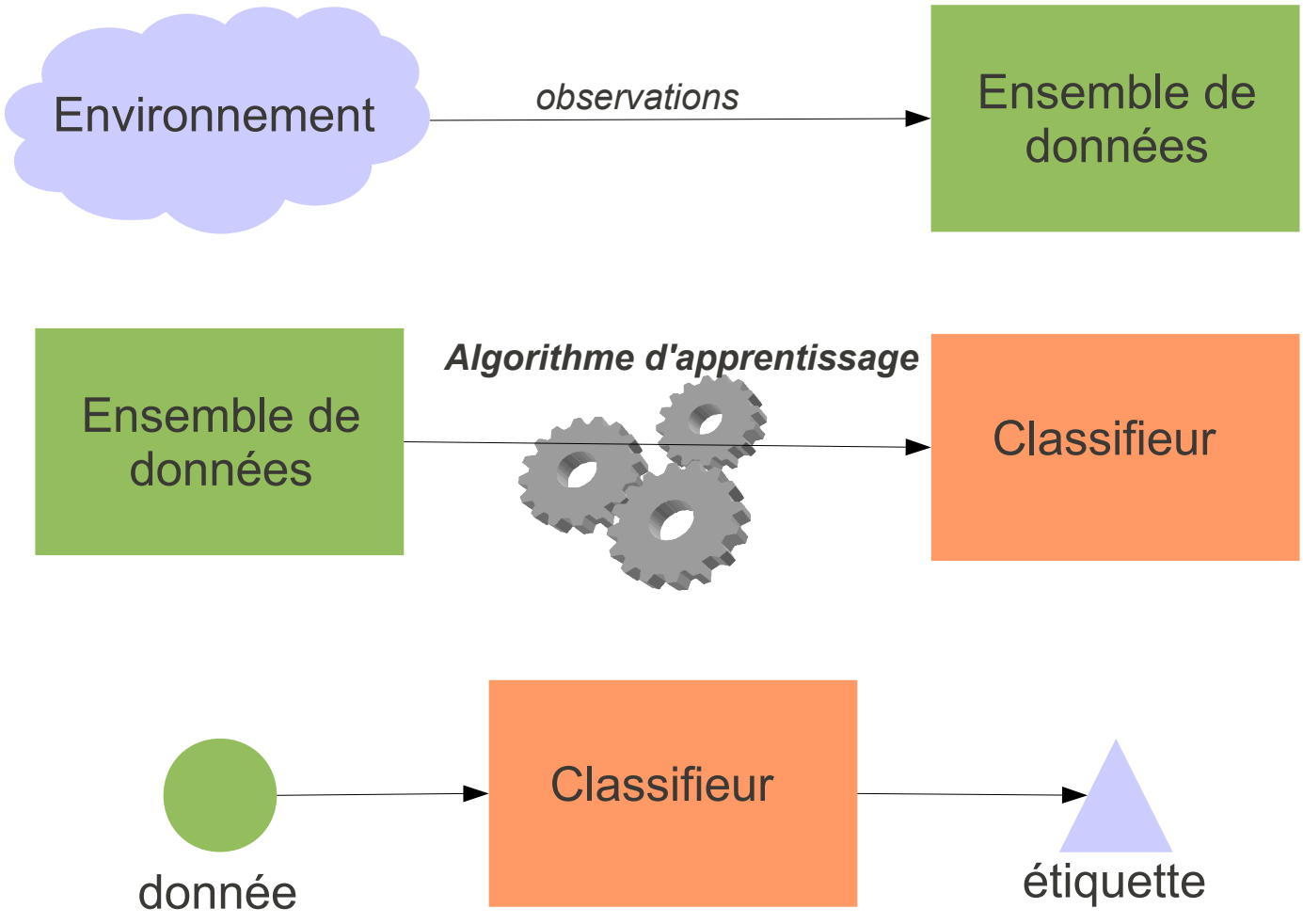
Observations : comestibilité versus (Diamètre, Hauteur du pied)

$$(176, 12), \cdots, (112, 8), (34, 1), \cdots$$

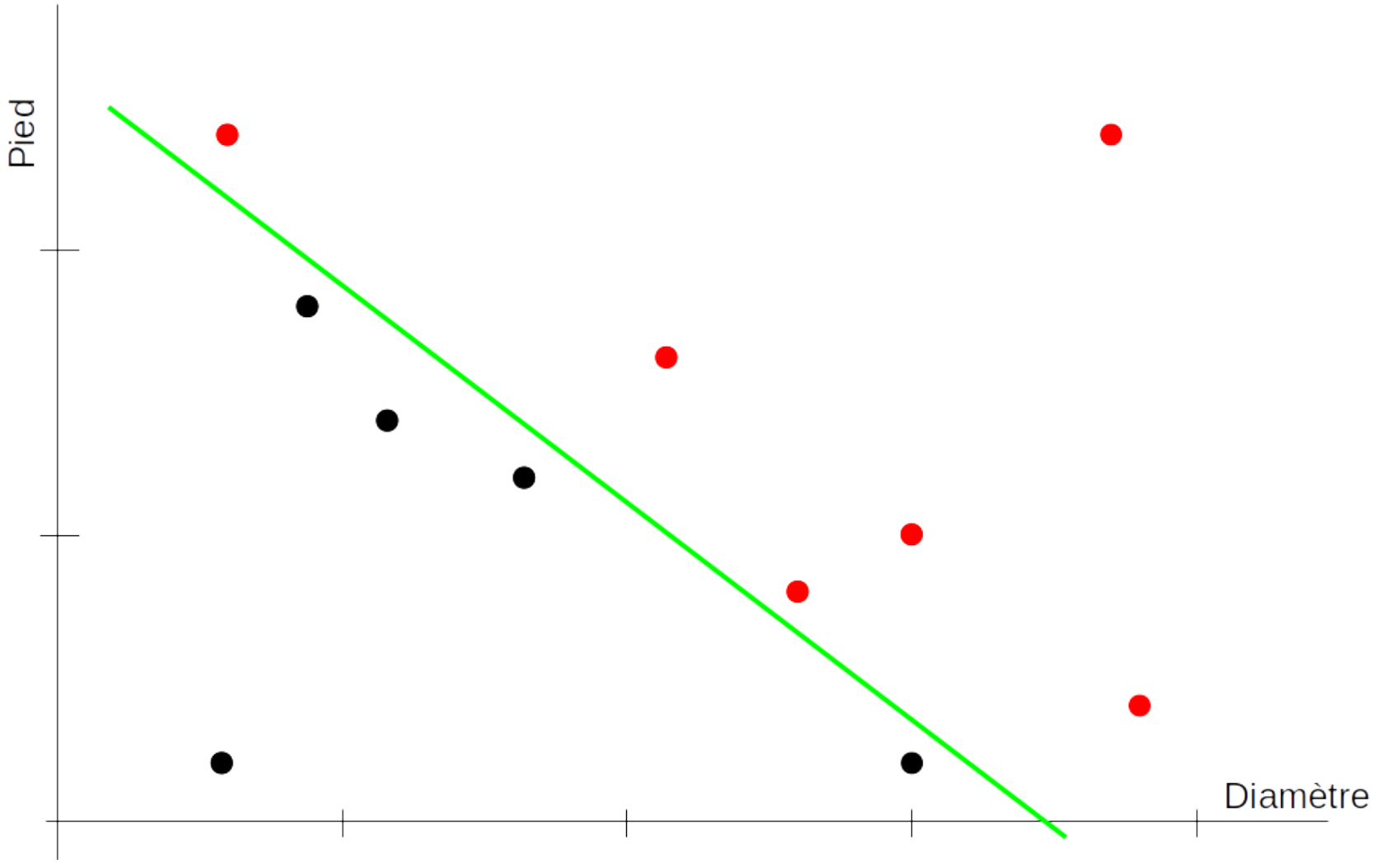


Construction automatique d'un modèle de prédiction à partir des données disponibles ?

Schéma général de la classification supervisée

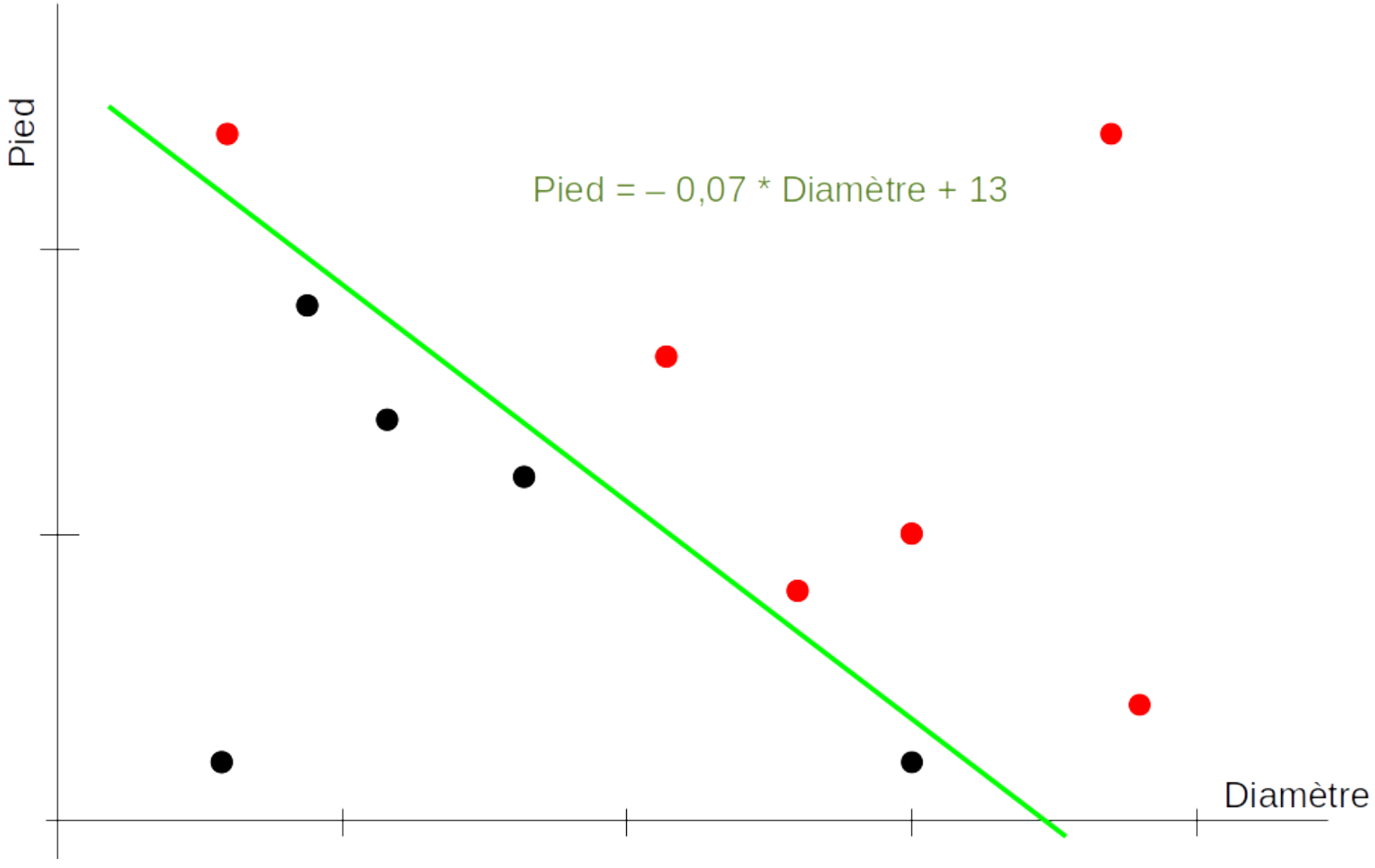


Un modèle simple : la droite (séparateur linéaire)



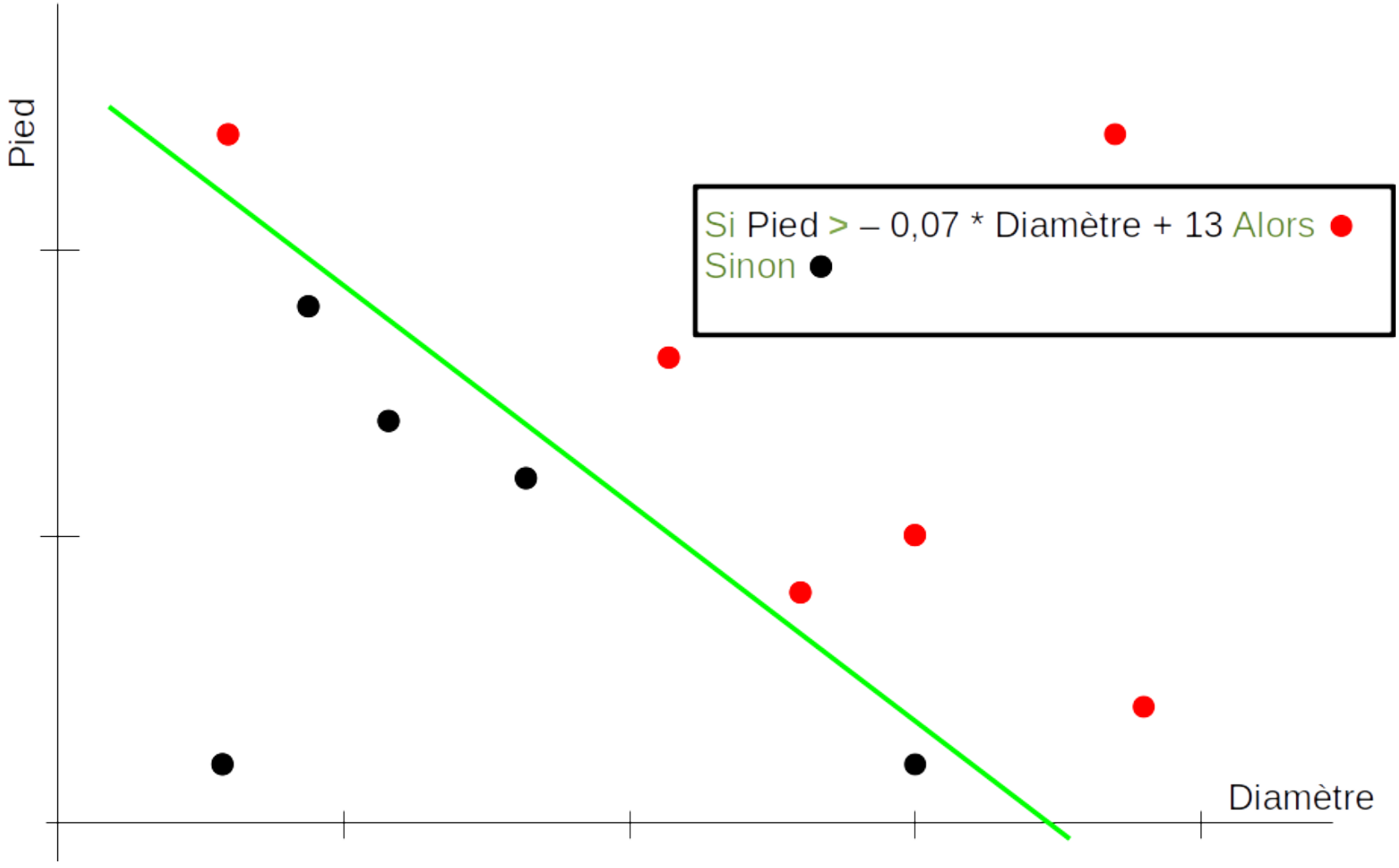
Un modèle simple : la droite (séparateur linéaire)

Apprentissage = Calcul de l'équation à partir des données



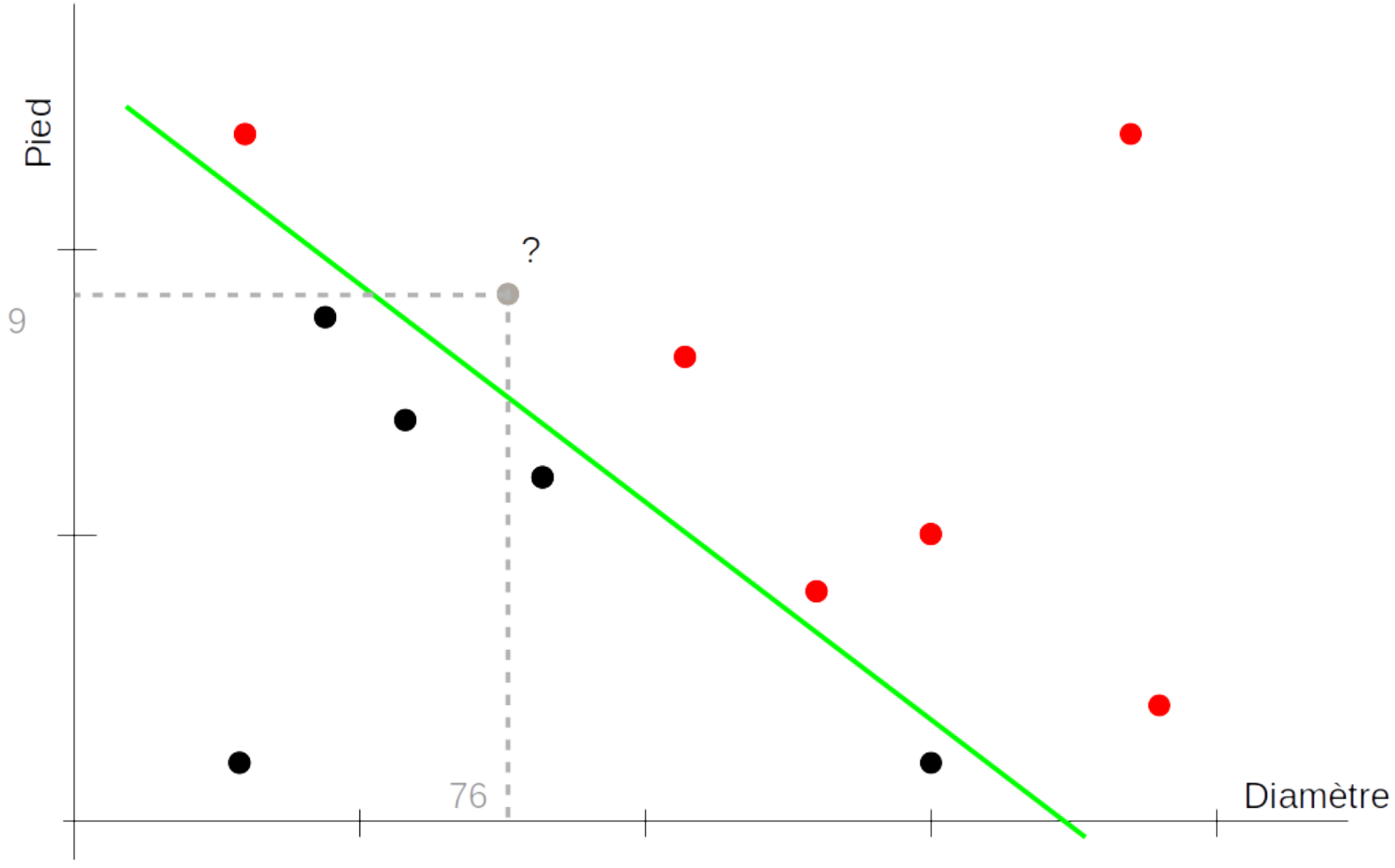
Un modèle simple : la droite (séparateur linéaire)

Apprentissage = Règle de classification à partir de l'équation



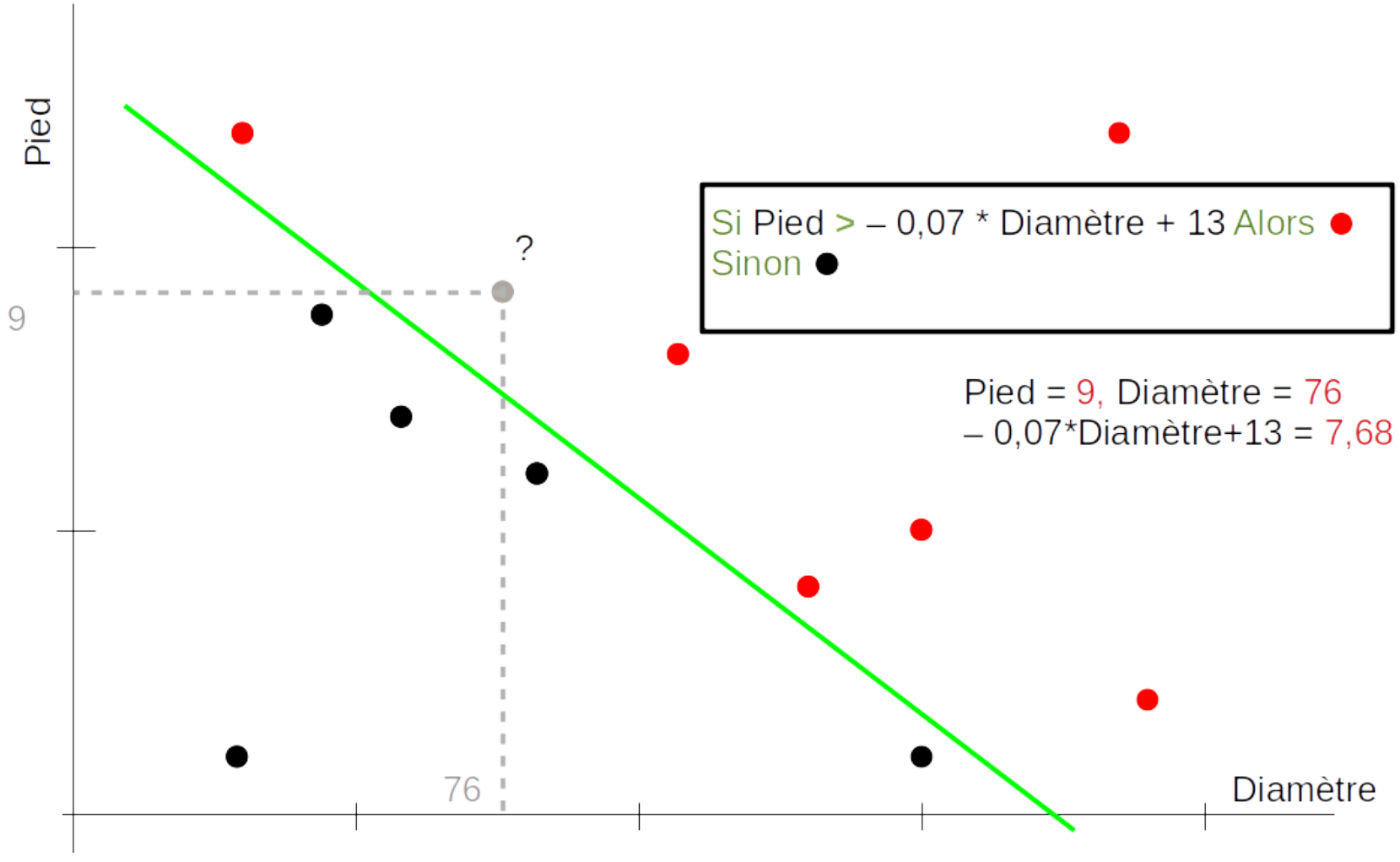
Un modèle simple : la droite (séparateur linéaire)

Prédiction = comment classer un nouveau champignon observé ?



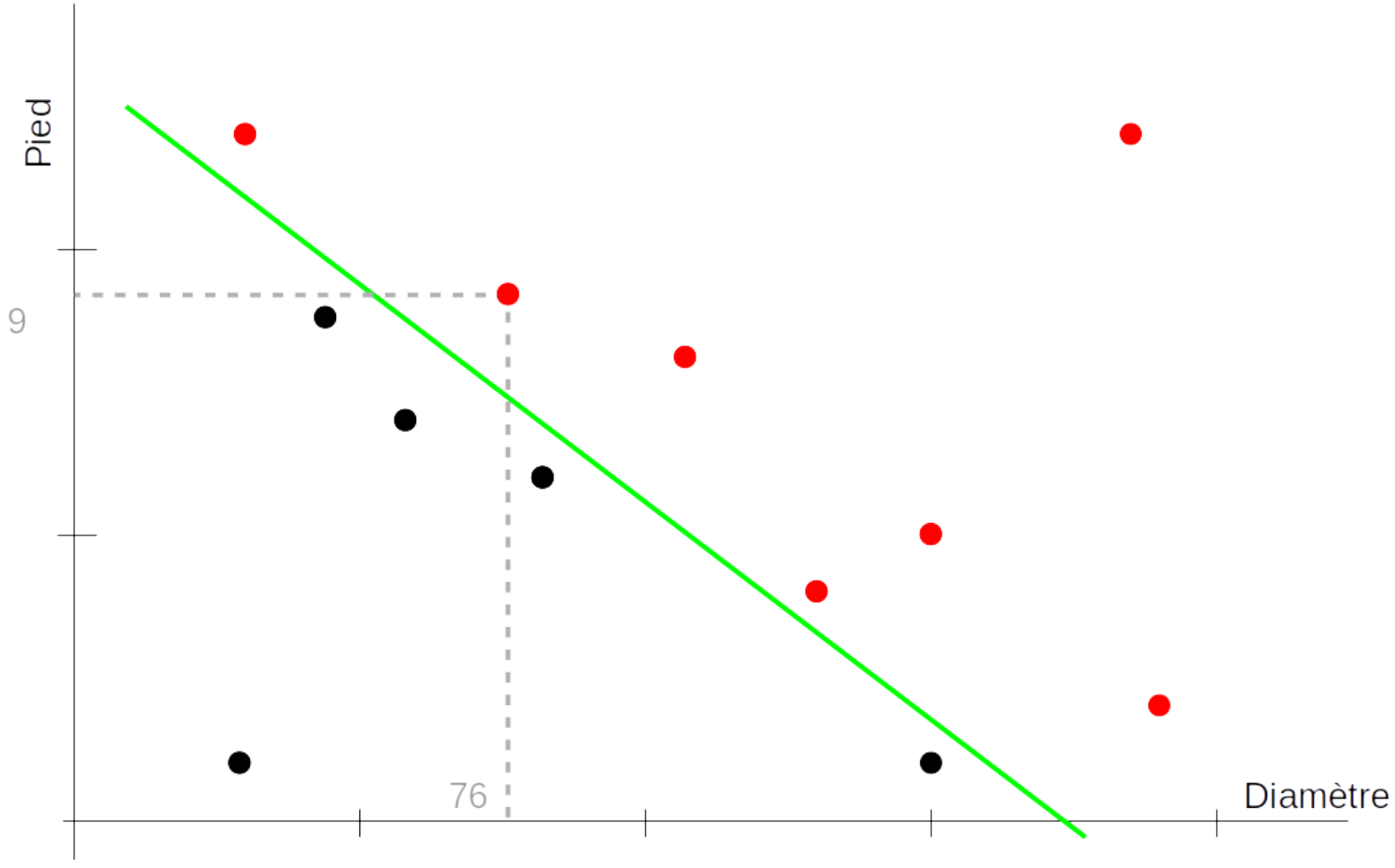
Un modèle simple : la droite (séparateur linéaire)

Prédiction = Appliquer la règle de classification



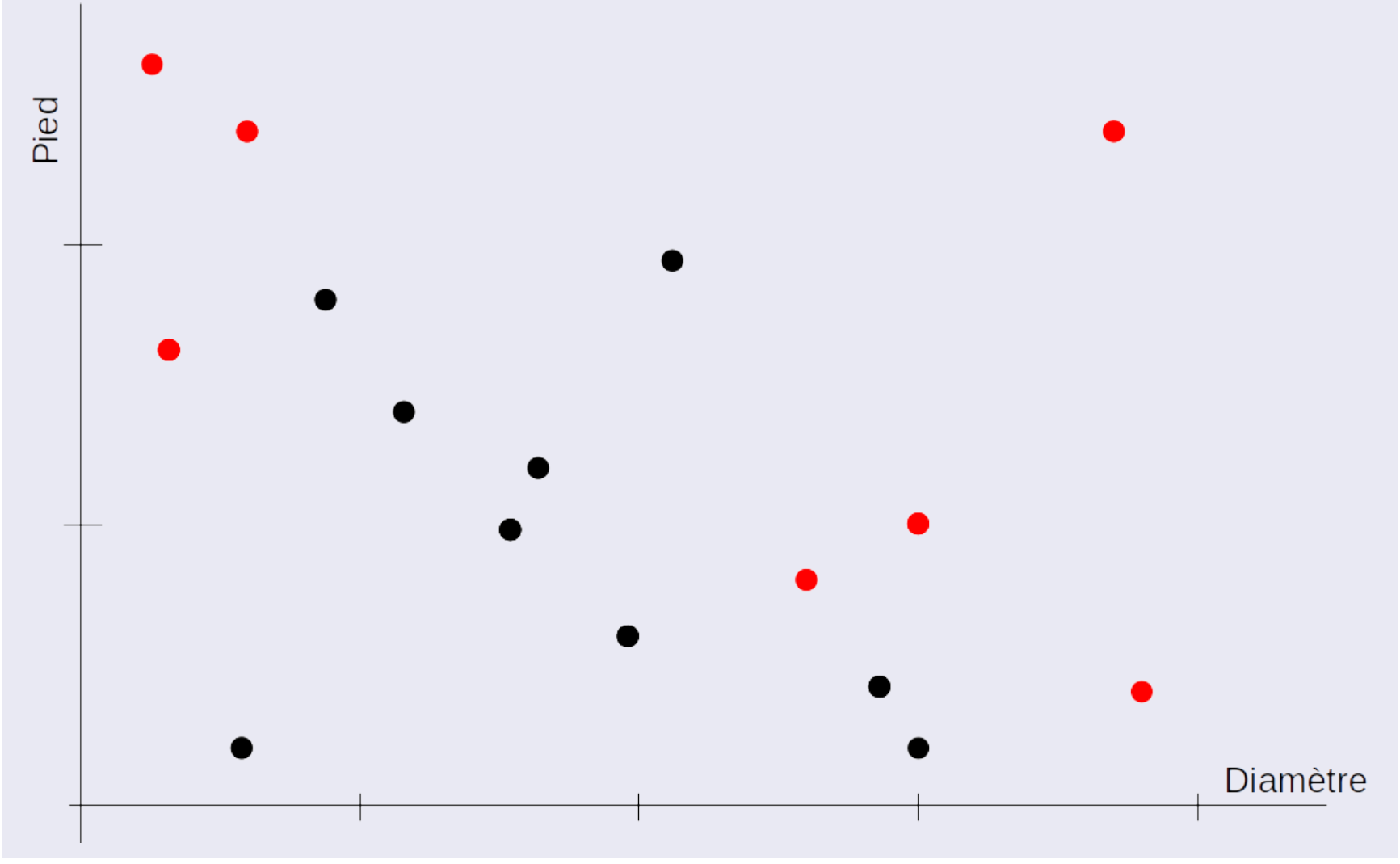
Un modèle simple : la droite (séparateur linéaire)

Quelle qualité de prédiction ?



Un modèle simple : la droite (séparateur linéaire)

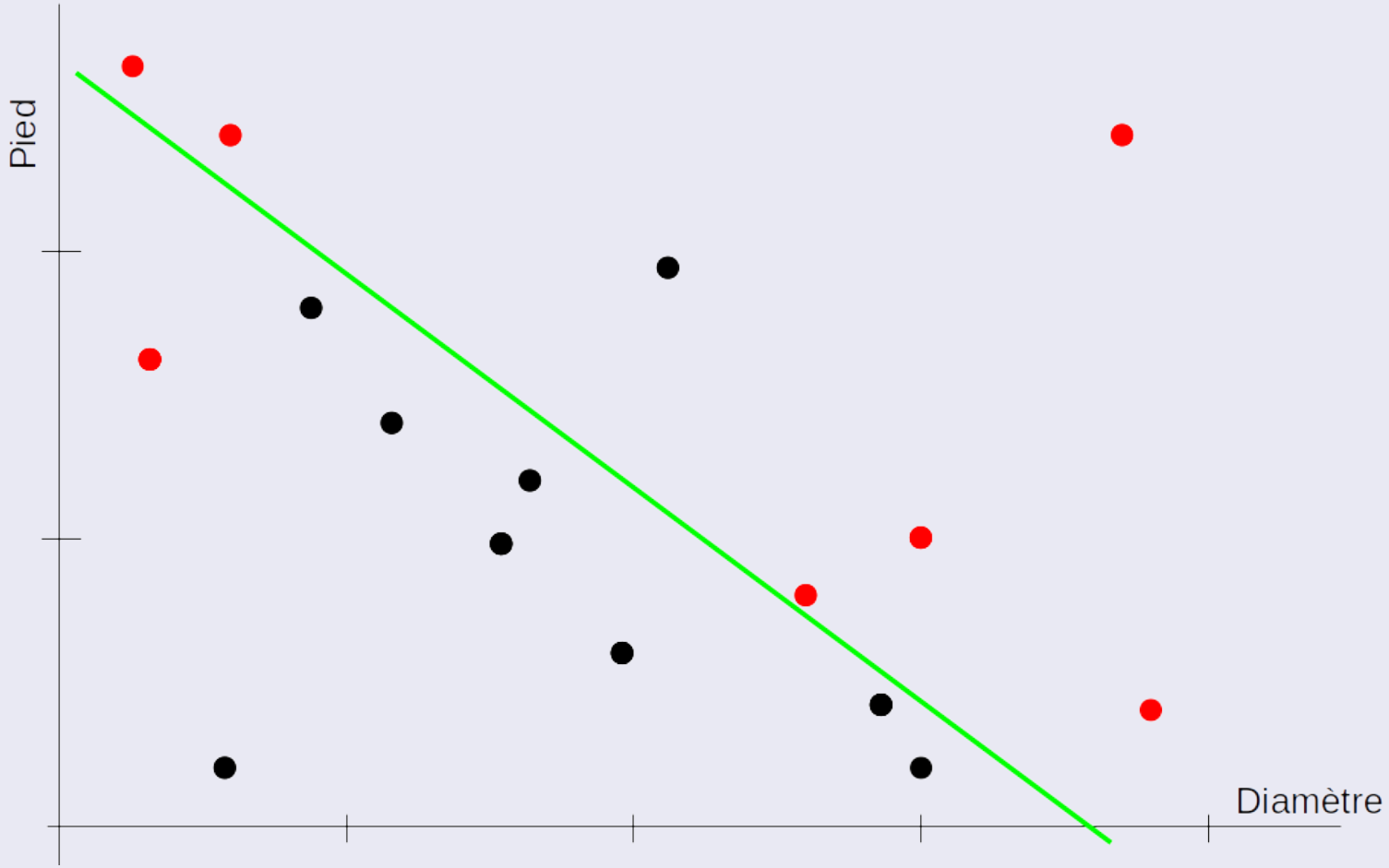
Pas toujours possible avec les données observées



Un modèle simple : la droite (séparateur linéaire)

Pas toujours possible avec les données observées

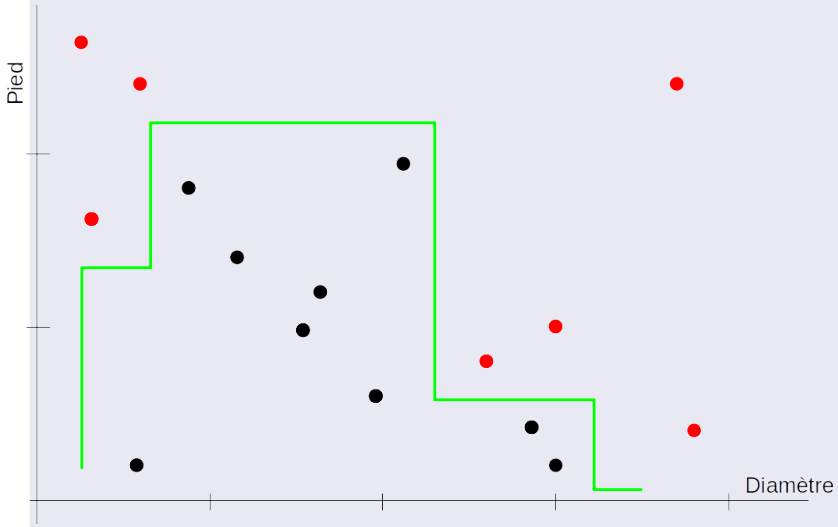
Tolérer quelques erreurs



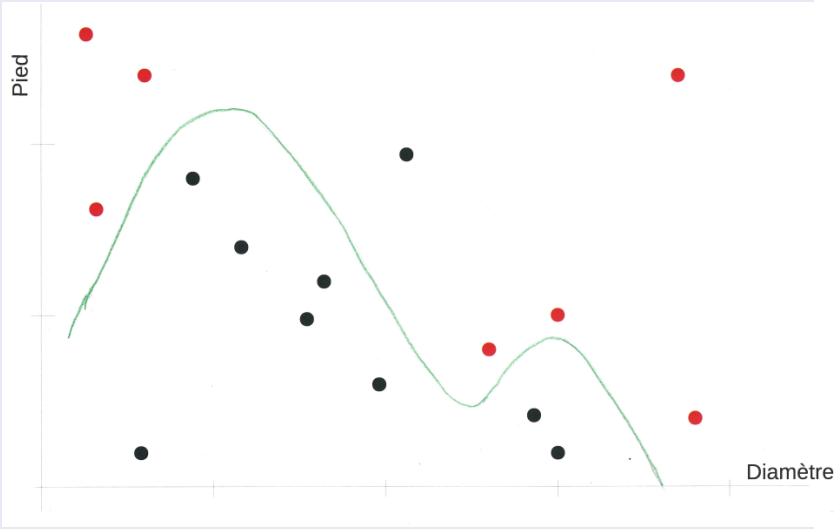
Un modèle simple : la droite (séparateur linéaire)

Pas toujours possible avec les données observées

Chercher un autre type de modèle plus expressif



Linéaire par morceaux



Polynomial $y = \sum_{i=0}^{n} a_i x^i$

Mais plus de valeurs à déterminer, donc plus complexe

Un modèle simple : la droite (séparateur linéaire)

Pas toujours possible avec les données observées

Principe du rasoir d'Occam (principe d'économie / parcimonie)

- Chercher le modèle le plus simple qui fasse le moins possible d'erreurs sur de nouvelles données
- Compromis complexité du modèle et quantité de données
- Fondements informatiques et mathématiques importants, ex. Valiant (prix Turing 2010), Vapnik (Médaille John von Neumann, 2017)

Un modèle simple : la droite (séparateur linéaire)

Pas toujours possible avec les données observées

Principe du rasoir d'Occam (principe d'économie / parcimonie)

- Chercher le modèle le plus simple qui fasse le moins possible d'erreurs sur de nouvelles données
- Compromis complexité du modèle et quantité de données
- Fondements informatiques et mathématiques importants, ex. Valiant (prix Turing 2010), Vapnik (Médaille John von Neumann, 2017)

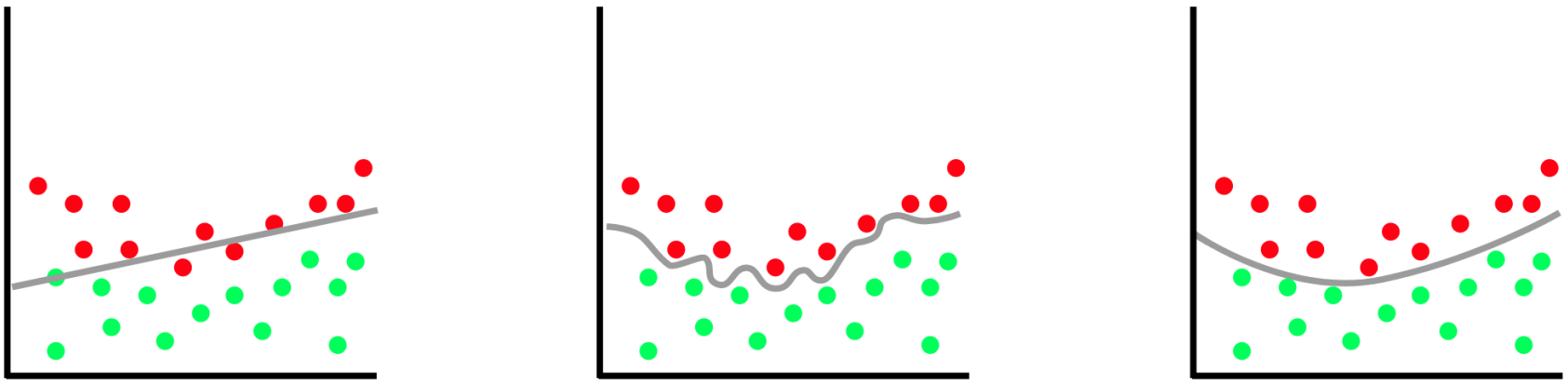
En pratique, données de dimensions plus grandes

Exemple : image couleur 300*300 = espace de 90 000 dimensions (au lieu de 2)

Proscrire l'apprentissage par coeur pour mieux généraliser

Chercher le modèle le plus correct et le plus simple

Pour savoir généraliser au mieux



- Accepter de faire quelques erreurs avec un modèle simple
- Tester plusieurs types de modèles, appris avec différents algorithmes (puis sélection de modèles, M1 option IAA)
- Lisser les modèles de prédiction (M2 IAAA)

Formalisation d'un problème de classification supervisée

Apprentissage supervisé – formalisation

- $\mathcal{X}$ espace d'entrée, $\mathcal{Y}$ espace des cibles
- D distribution sur $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$
- $S_{\text{train}} = \{(X_i, Y_i)\}_{i=1}^m$ échantillon de n v.a. Indépendamment et Identiquement Distribuées (IID) suivant D
- Fonction de perte $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$

But : minimisation du (vrai) risque

A partir de S_{train} trouver $f : \mathcal{X} \to \mathcal{Y}$ telle que $f = \text{argmin}_h R(h)$ avec

$$R(h) = \mathbb{E}_{XY}\ell(h(X), Y) = \int_{\mathcal{X}\times\mathcal{Y}} \ell(h(x), y) dD(x, y)$$

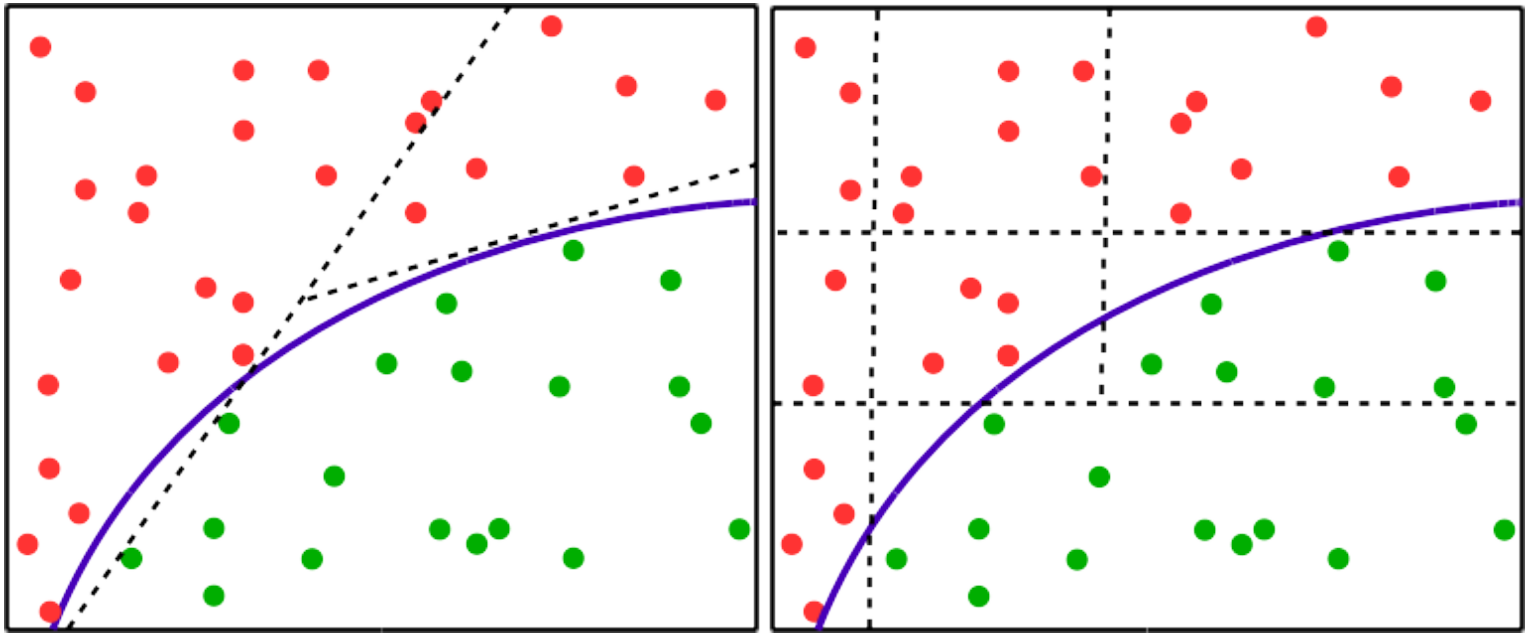
Pour la classification/catégorisation, $|\mathcal{Y}| < +\infty$, $\ell(y, y') = \mathbb{I}(y \neq y')$ et

$$R(h) = \mathbb{P}_{X,Y\sim D}(h(X) \neq Y)$$

Outline

Principe d'élaboration d'un arbre de décision appris sur S

- Processus récursif de division de l'espace des données en sous-régions de plus en plus parfaite en termes de classes



- Construction d'un arbre de décision = décomposition d'un problème de classification en une suite de tests imbriqués
  Chaque test porte sur une variable (parallèle aux axes) ou sur une combinaison linéaire de plusieurs variables (oblique).

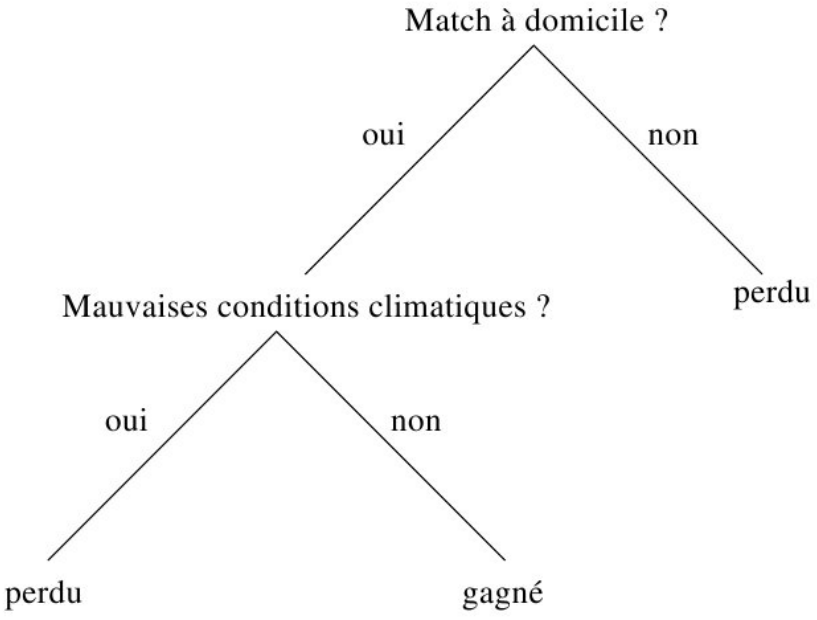
Introduction sur les arbres de décision

Notations

Soient $A_1, \ldots, A_m$: attributs binaires, n-aires, ou réels. L'espace de description est $\mathcal{X} = \prod_{j=1}^{m} \mathcal{X}_j$ où $\mathcal{X}_j$ est le *domaine* de A_j.

Définition d'un arbre de décision

Ensemble de règles de classification basant leur décision sur des tests associés aux attributs, organisés de manière arborescente.



Les arbres de décision (cont'd)

Structure

- Noeuds internes (noeuds de décision) : étiquetés par des tests applicables à toute description d'une instance. Généralement, un noeud interne = test sur un unique attribut.
- Arcs issus d'un noeud interne : réponses possibles au test du noeud.
- Feuilles de l'arbre : étiquetées par une classe.
- Chaque noeud interne ou feuille, est repéré par sa position (liste des numéros des arcs qui permettent d'y accéder en partant de la racine).

Arbre de décision et apprentissage

- Tout arbre de décision définit un classifieur
- Classifieur qui se traduit immédiatement en terme de règles de décision

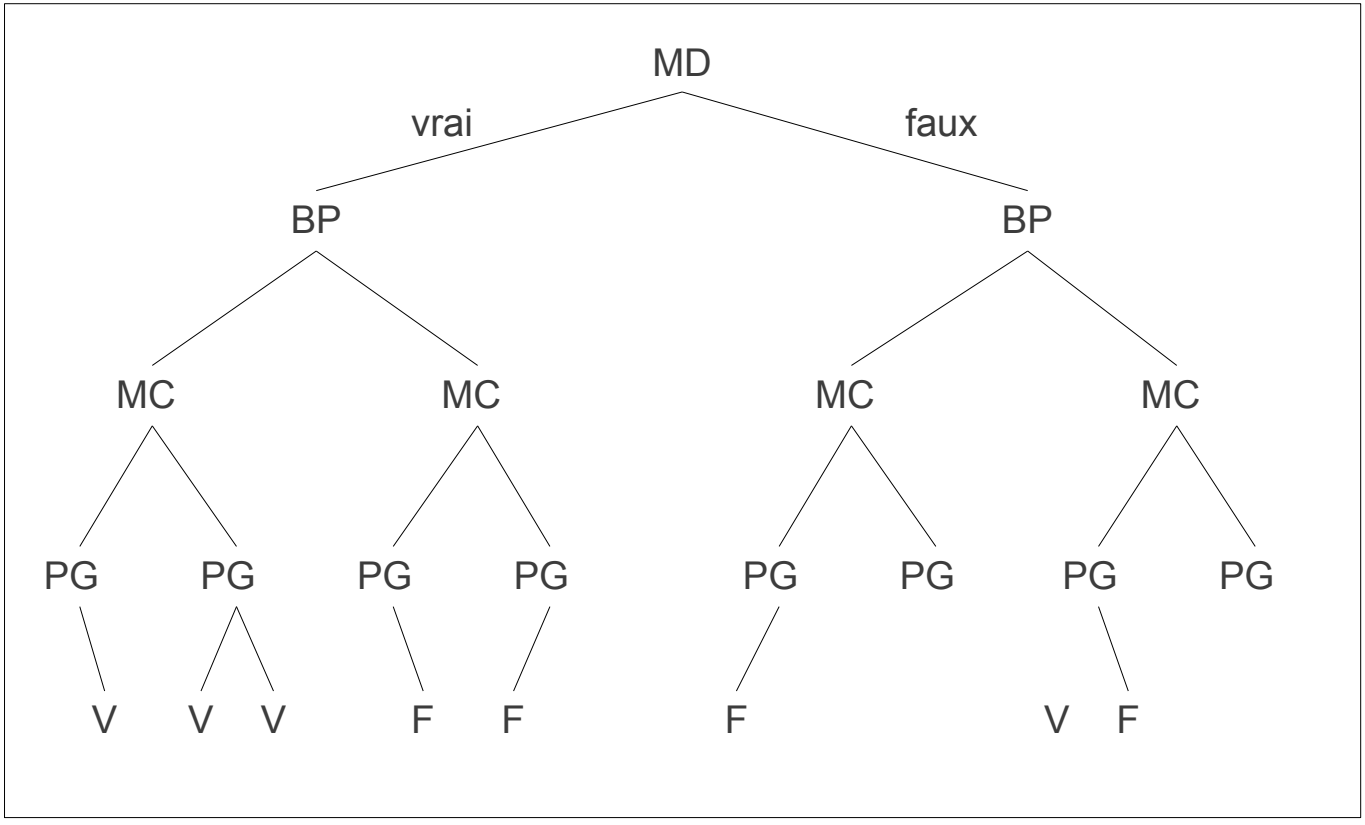
Gros avantages des arbres de décision

1. Décisions aisément interprétables,
2. classification très rapide

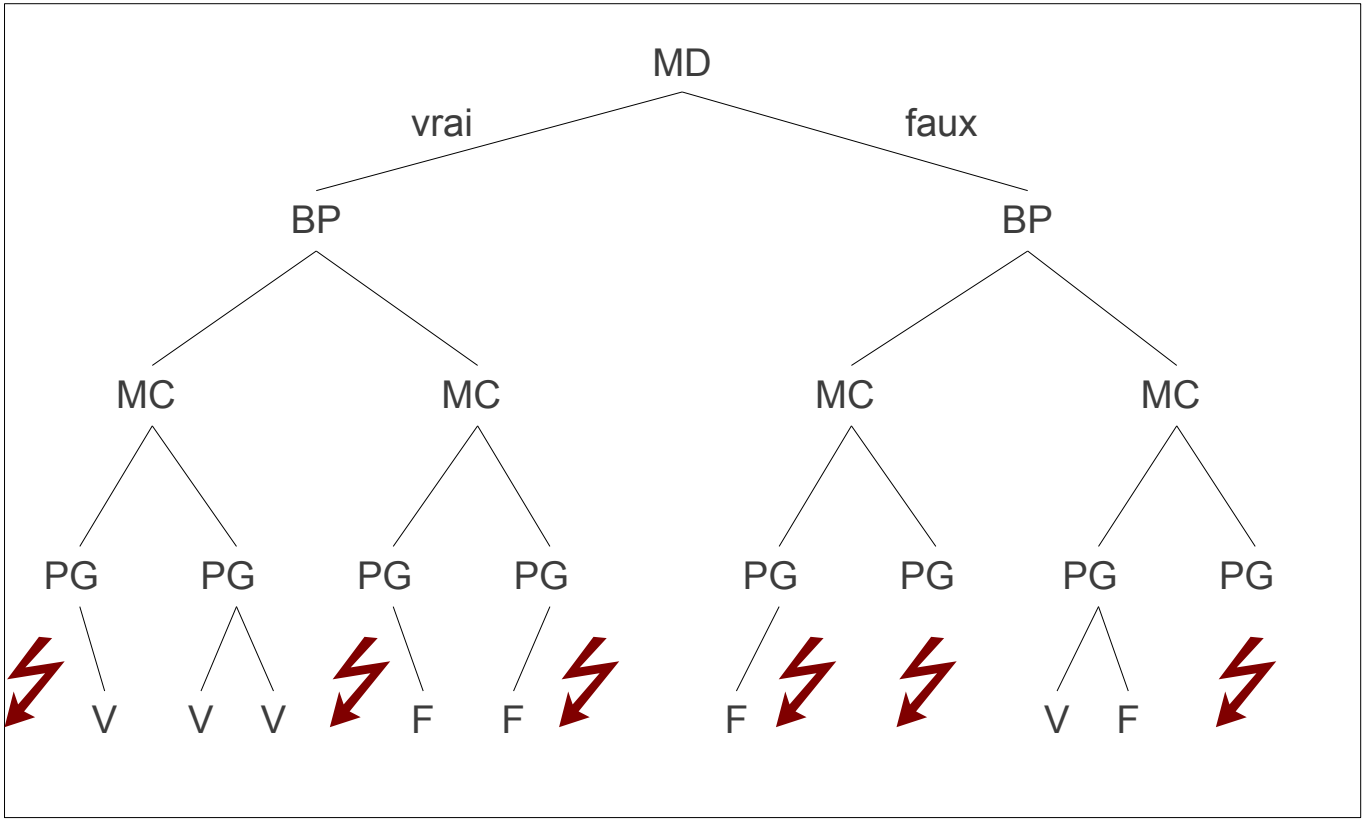
Construire un arbre de décision à partir de S

Match dom.	Balance pos.	Mauvais climat	Match préc. gagné	Victoire
V	V	F	F	V
F	F	V	V	V
V	V	V	F	V
V	V	F	V	V
F	V	V	V	F
F	F	V	F	F
V	F	F	V	F
V	F	V	F	F

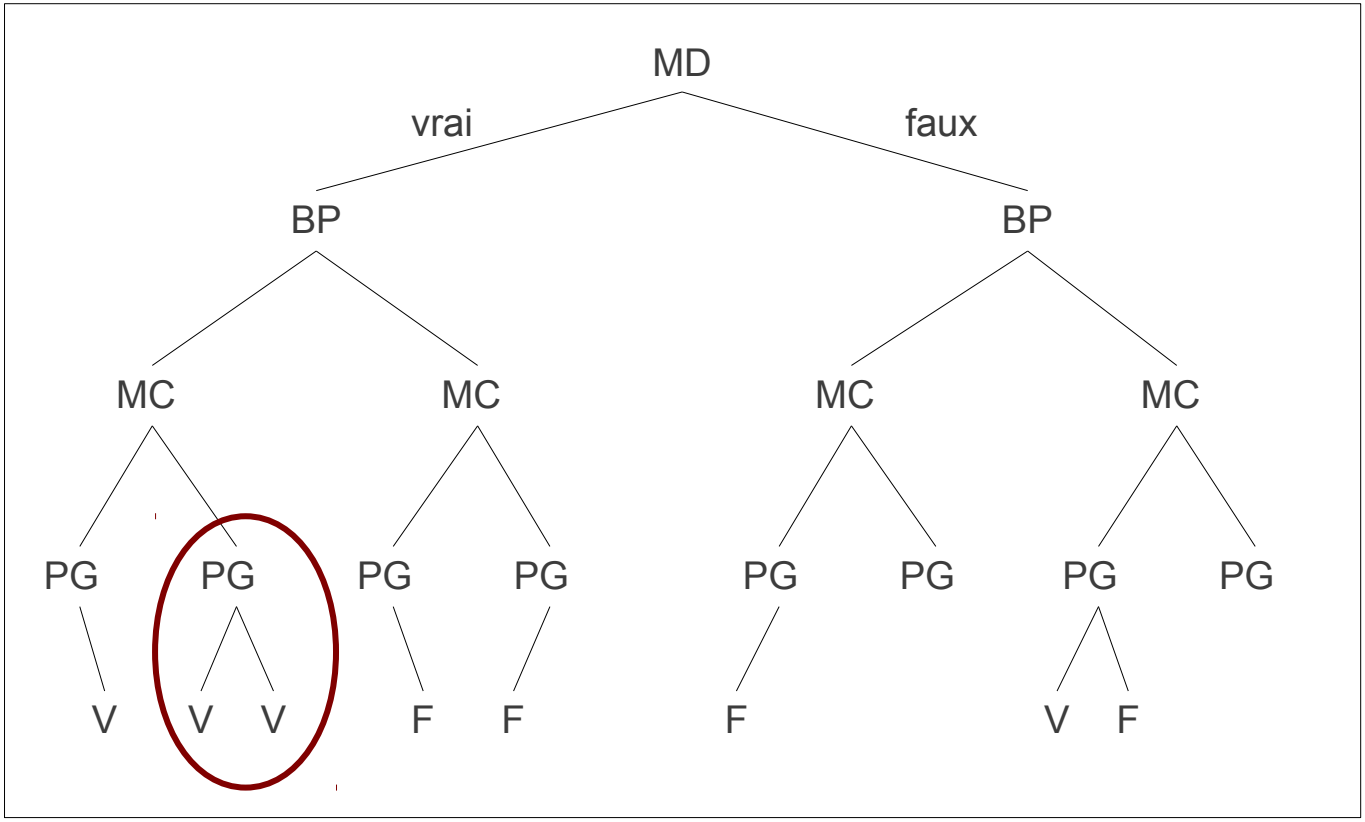
Construction aveugle



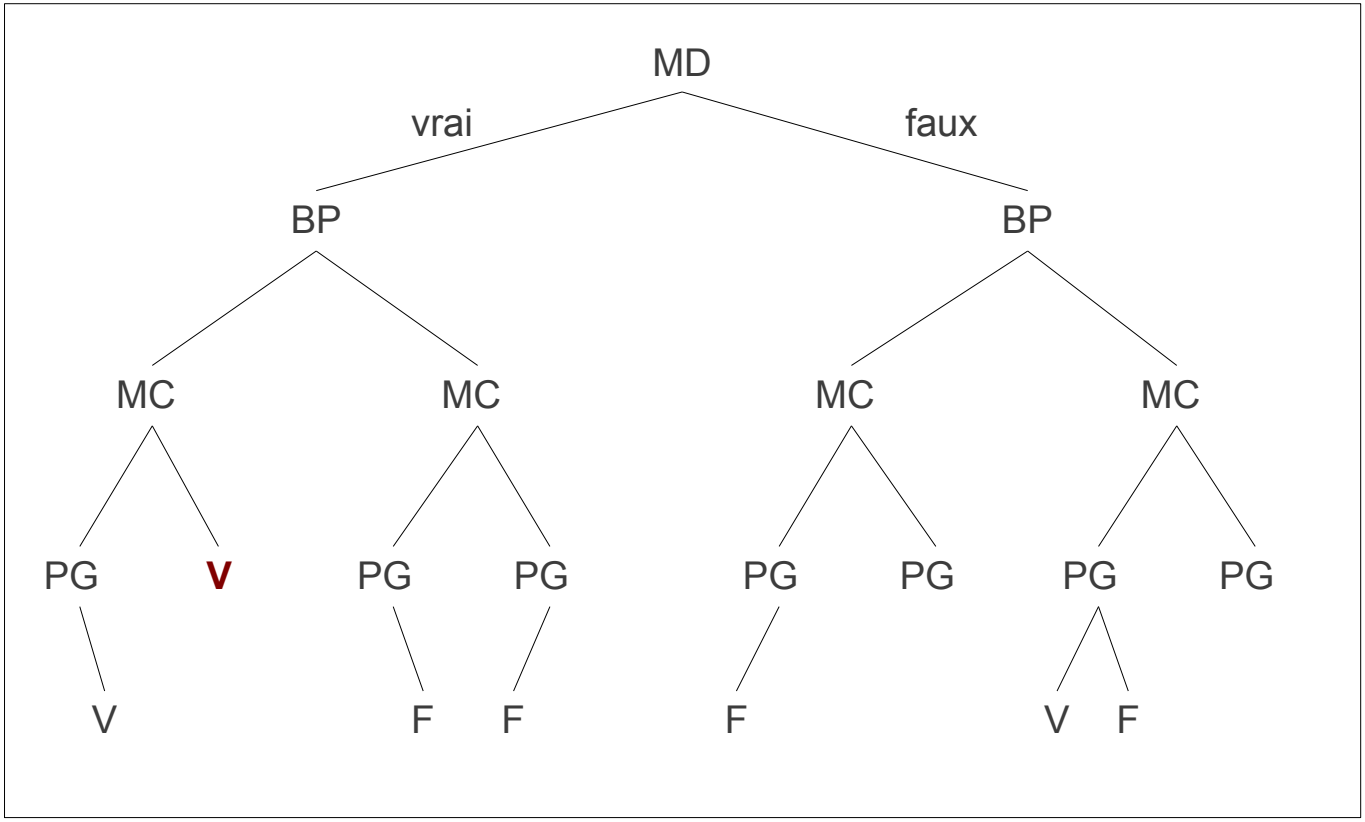
Construction aveugle



Construction aveugle



Construction aveugle

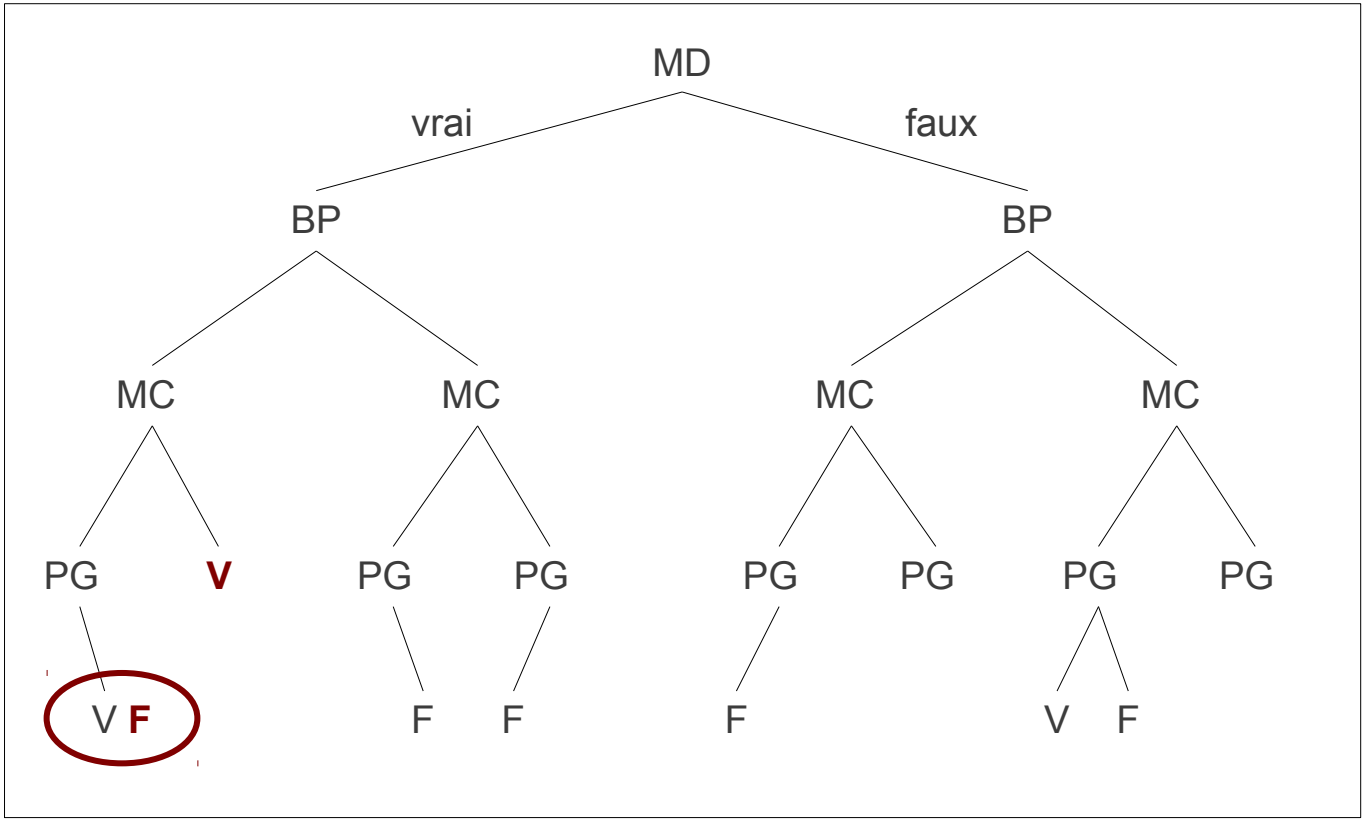


Construire un arbre de décision à partir de S

Match dom.	Balance pos.	Mauvais climat	Match préc. gagné	Victoire
V	V	F	F	V
F	F	V	V	V
V	V	V	F	V
V	V	F	V	V
F	V	V	V	F
F	F	V	F	F
V	F	F	V	F
V	F	V	F	F
V	V	V	F	F

Cas de non-déterminisme – Pas d'arbre *parfait*

Problème non-déterministe



Arbre de décision de risque empirique minimal ?

- Construire un arbre de décision de risque empirique minimal = possible pour tout S
- Mais très faible capacité prédictive (généralisation). pourquoi ?
- Est-ce que le plus petit arbre de décision compatible avec S aura de meilleures capacités de généralisation ?
- *cf.* théorie de l'apprentissage statistique (Vapnik)

Petit arbre de décision compatible

Intérêt

Conforme aux principes invoqués en apprentissage

- *Principe du rasoir d'Occam* : trouver l'hypothèse la plus courte possible compatible avec les données
- *Principe MDL* (minimum description length) : soit des données S, trouver l'hypothèse H telle que $|H| + |S/H|$ soit la plus petite possible (longueur d'un codage des données via H)

Mais...

Trouver le plus petit arbre de décision compatible avec S est un problème NP-complet.

Besoin d'algorithmes spécifiques

Plus petit arbre possible (mais pas forcément le plus petit), compatible au mieux avec les données, pour satisfaire le principe ERM.

Structure générique de l'algorithme d'apprentissage

Algorithmes de référence

- CART
- C4.5

Principe général

Construction top-down, gloutonne et récursive, d'un *petit* arbre consistant avec la *plupart* des données

Trois opérateurs majeurs

1. Décider si un noeud est terminal,
2. Si un noeud n'est pas terminal, lui associer un test
3. Si un noeud est terminal, lui associer une classe

Structure générique de l'algorithme (cont'd)

Require: échantillon S

Initialiser l'arbre courant à vide : racine = noeud courant
repeat
    Décider si le noeud courant est terminal
    if noeud terminal **then**
        Lui affecter une classe
    else
        Sélectionner un test et créer autant de nouveaux noeuds qu'il y a de réponses possibles au tests
    end if
    Passer au noeud suivant non-exploré (s'il existe)
until Plus de noeud sans classe
return Arbre de décision A

Structure générique de l'algorithme (cont'd)

Noeud terminal ?

- Lorsque (presque) tous les exemples de S en ce noeud sont dans la même classe,
- Lorsqu'il n'y a plus d'attributs à tester à ce niveau

Quelle classe à un noeud terminal ?

- Classe majoritaire
- Classe la plus représentée, si égalité

Sélection d'un test ?

Choix de l'attribut qui fait le mieux progresser la discrimination des données de S : gain en information.

- Indice de Gini (CART)
- Critère d'entropie (C4.5)

Indice de Gini et entropie

Mesure du désordre (ou d'inégalité de répartition), pour choix d'un test à une position de l'arbre

Données

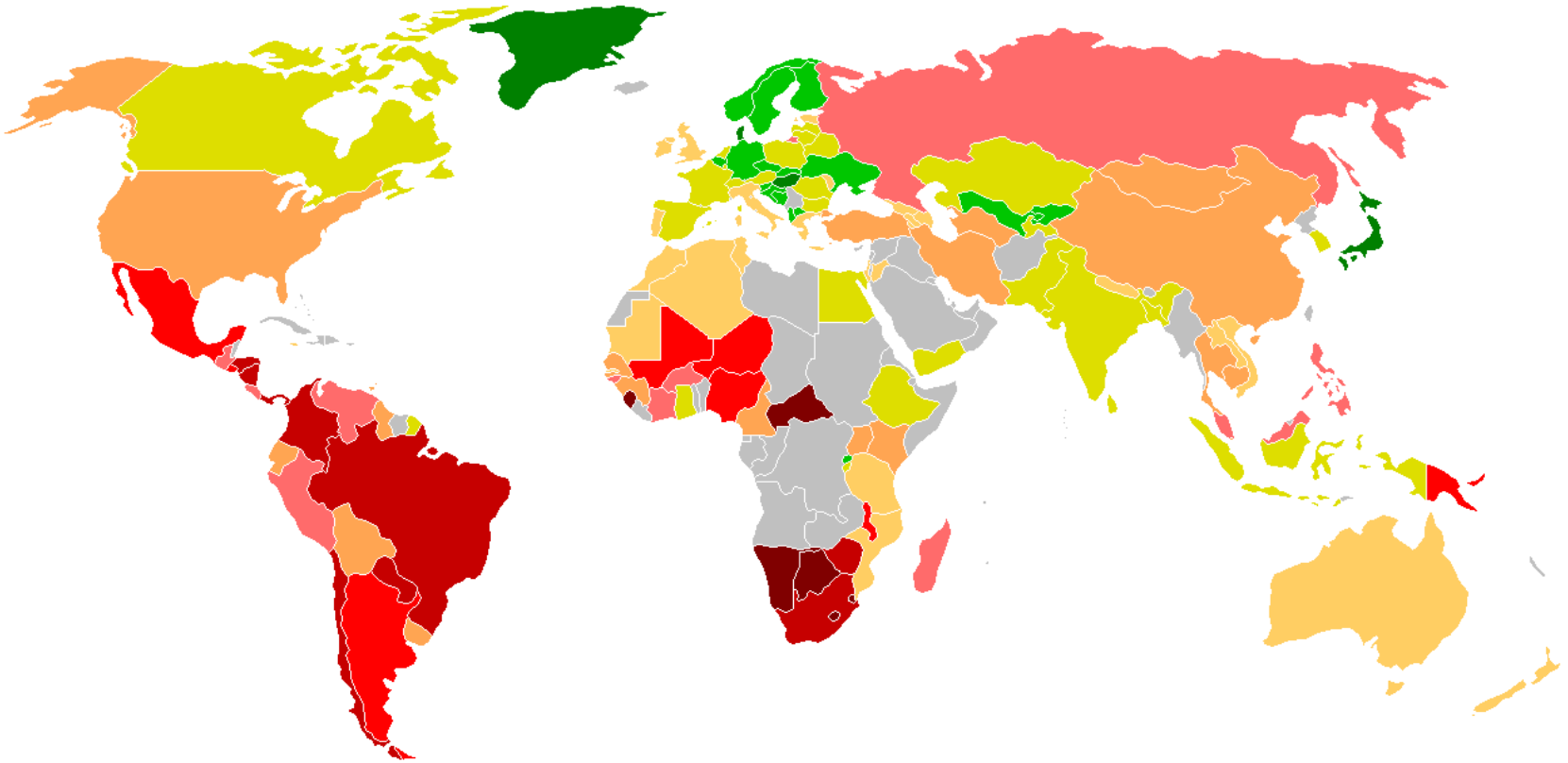
- S un échantillon, a l'attribut cible
- $S_1, \ldots, S_k$ partition de S selon les classes de a

Indice de Gini

$$\text{Gini}(S) = \sum_{i=1}^{k} \frac{|S_i|}{|S|}(1 - \frac{|S_i|}{|S|}) = \sum_{i \neq j} \frac{|S_i| \times |S_j|}{|S|^2}$$

Coefficient de Gini

(Inégalité des revenus – source : wikipedia)



Color	Gini coefficient				
	< 0,25		0,40 - 0,44		> 0,60
	0,25 - 0,29		0,45 - 0,49		NA
	0,30 - 0,34		0,50 - 0,54		
			0,35 - 0,39		0,55 - 0,59

Indice de Gini et entropie (cont'd)

Mesure du désordre (ou d'inégalité de répartition)

Données

- S un échantillon, a l'attribut cible
- $S_1, \ldots, S_k$ partition de S selon les classes de a

Indice de Gini

$$\text{Gini}(S) = \sum_{i=1}^{k} \frac{|S_i|}{|S|}(1 - \frac{|S_i|}{|S|}) = \sum_{i \neq j} \frac{|S_i| \times |S_j|}{|S|^2}$$

Entropie

$$\text{Ent}(S) = -\sum_{i=1}^{k} \frac{|S_i|}{|S|} \log(\frac{|S_i|}{|S|})$$

(variation d'un terme multiplicatif constant selon la base du log)

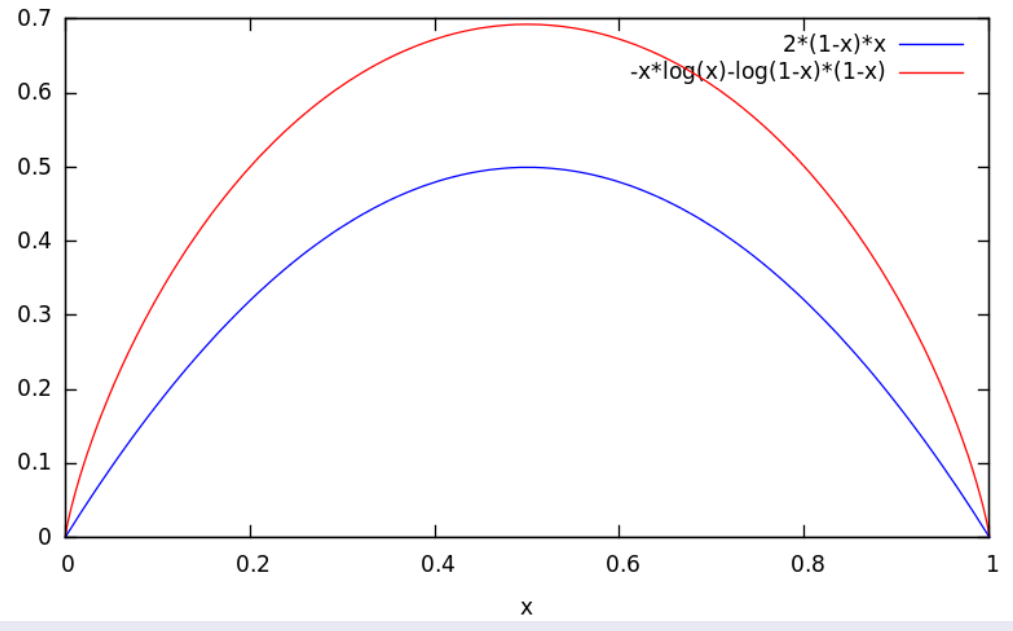
Indice de Gini et entropie (cont'd)

Exemple

Supposons $k = 2$, soit $x = \frac{|S_1|}{|S|}$

$$\text{Gini}(S) = 2x(1-x)$$

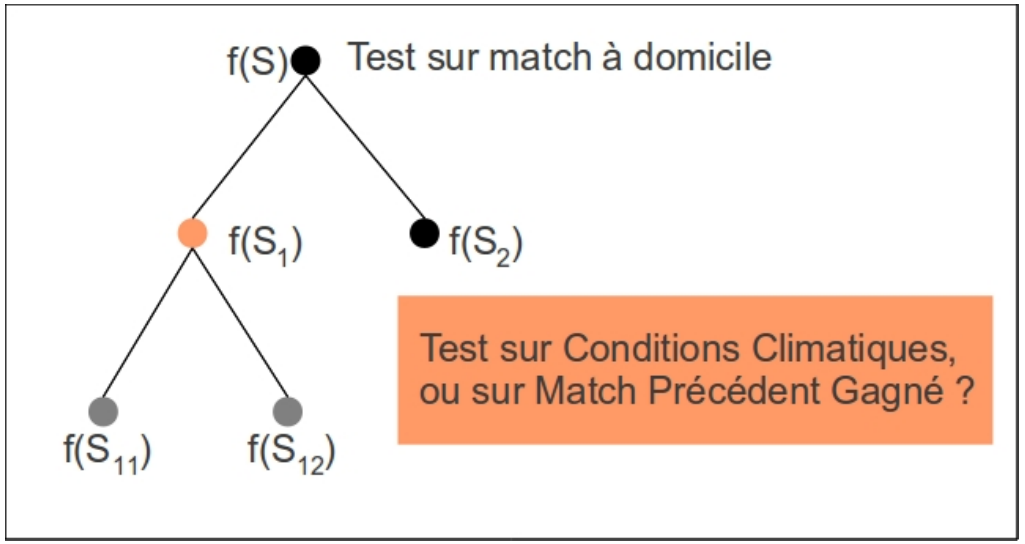
$$\text{Ent}(S) = -x\log x - (1-x)\log(1-x)$$



Comment choisir un test parmi les attributs disponibles ?

Cas des attributs binaires

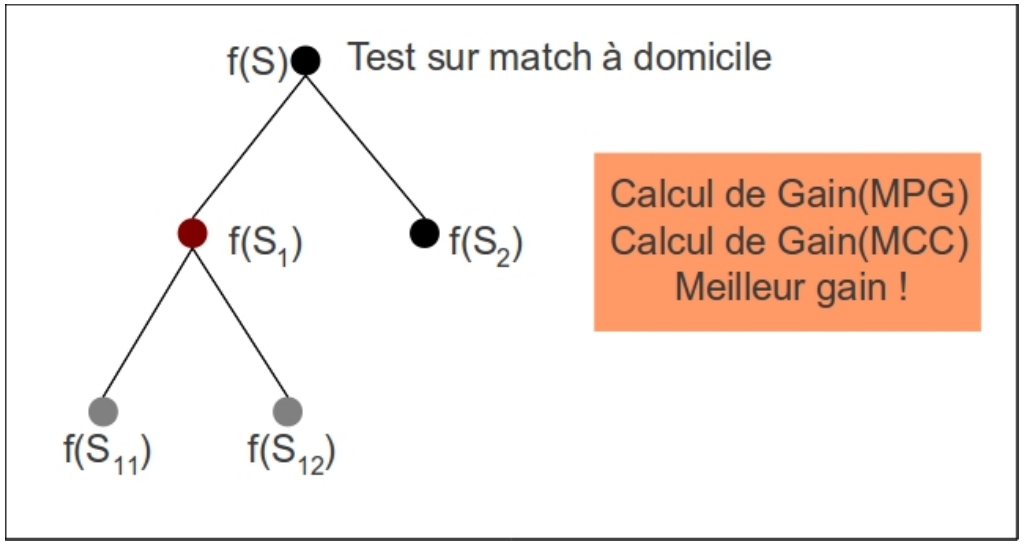
- p : position courante (ϵ à la racine) de l'arbre en construction
- T : un test (sur un attribut : T a deux réponses possibles !)
- f =Ent ou f =Gini, S_p échantillon associé à p
- S_{pi} : ensemble des exemples de S_p qui satisfont la i-ème branche de T
- P_i : proportion des éléments de S_p qui satisfont la i-ème branche de T



Comment choisir un test parmi les attributs disponibles ?

Cas des attributs binaires

- p : position courante (ϵ à la racine) de l'arbre en construction
- T : un test (sur un attribut : T a deux réponses possibles !)
- f =Ent ou f =Gini, S_p échantillon associé à p
- S_{pi} : ensemble des exemples de S_p qui satisfont la i-ème branche de T
- P_i : proportion des éléments de S_p qui satisfont la i-ème branche de T



Comment choisir un test parmi les attributs disponibles ?

Cas des attributs binaires

- p : position courante (ϵ à la racine) de l'arbre en construction
- T : un test (sur un attribut : T a deux réponses possibles !)
- f =Ent ou f =Gini, S_p échantillon associé à p
- S_{pi} : ensemble des exemples de S_p qui satisfont la i-ème branche de T
- P_i : proportion des éléments de S_p qui satisfont la i-ème branche de T

$$\text{Gain}_f(p, T) = f(S_p) - \sum_{j=1}^{2} P_j \times f(S_{pj})$$

Le gain d'un attribut à une position

$$\text{Gain}_f(p, T) = f(S_p) - \sum_{j=1}^{2} P_j \times f(S_{pj})$$

Propriétés (cas attributs binaires)

- Terme $f(S_p)$ ne dépend pas de T !
- Conséquence : Maximiser le gain revient à minimiser $\sum_{j=1}^{2} P_j \times f(S_{pj})$!
- Gain maximal lorsque le test sur un attribut permet de classer correctement toutes les données
- (Gain minimal si aucune information apportée par ce test au regard de la classification)
- Sélction de l'attribut qui maximise le gain : stratgégie *gloutonne*

Sur l'exemple des matchs

Choix du premier attribut (position racine)

Critère de Gini (DOM, BAL, MCC, MPG)

- $\text{Gain}(\epsilon, \text{DOM}) = \text{Gini}(S) - (\frac{5}{8}\text{Gini}(S_1) + \frac{3}{8}\text{Gini}(S_2)) =$
  $\text{Gini}(S) - 2 \times \frac{5\times2\times3}{8\times5\times5} - 2 \times \frac{3\times1\times2}{8\times3\times3} = \text{Gini}(S) - \frac{7}{15}$
- $\text{Gain}(\epsilon, \text{BAL} = \text{Gini}(S) - \frac{3}{8}$
- $\text{Gain}(\epsilon, \text{MCC} = \text{Gini}(S) - \frac{7}{15}$
- $\text{Gain}(\epsilon, \text{MPG} = \text{Gini}(S) - \frac{1}{2}$

Gain Maximum pour BAL (idem pour Entropie) : c'est l'attribut choisi à la racine pour une première phase de classification.

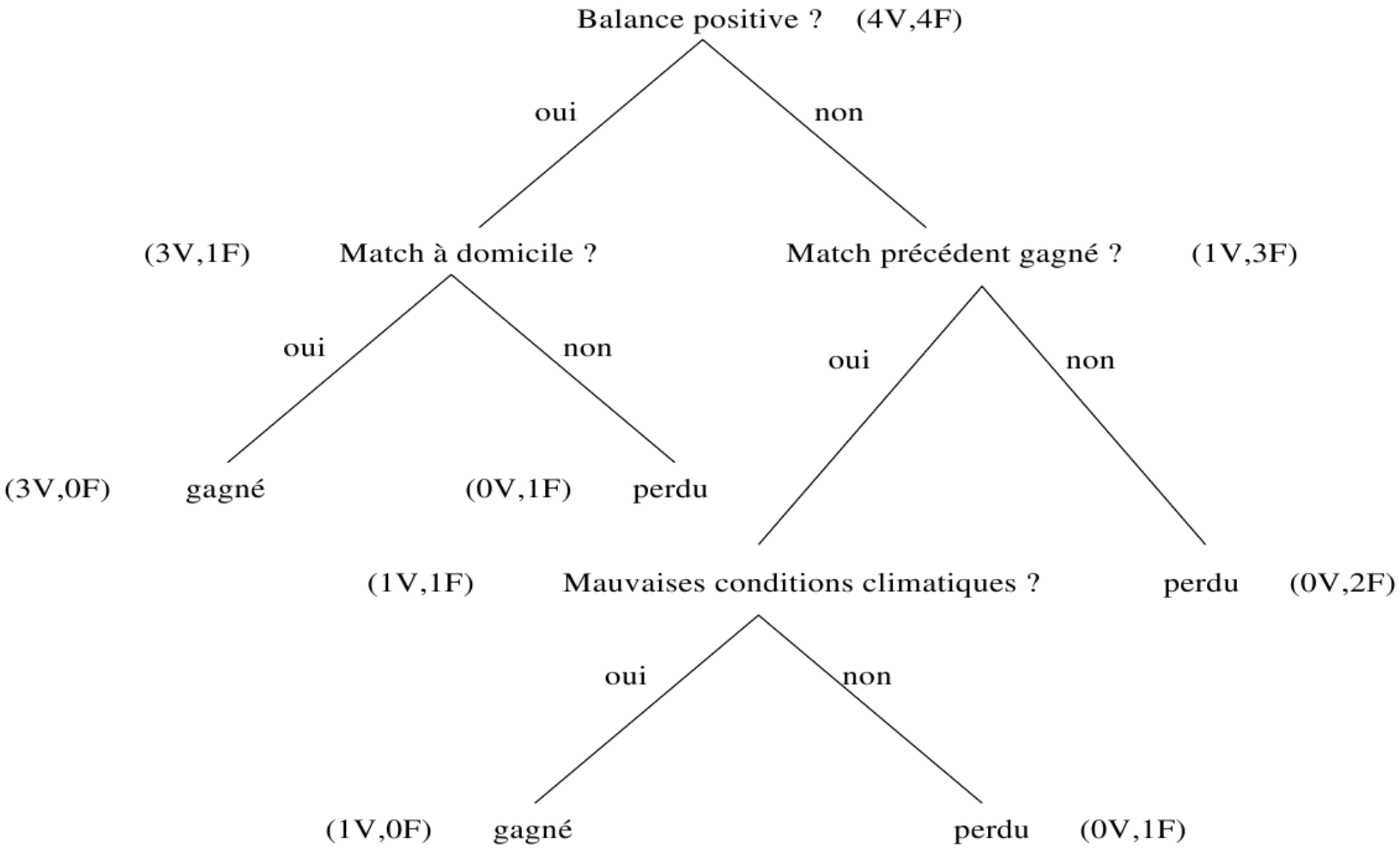
Choix du second test en position S_1 (BAL=V)

Calcul des gains pour chaque attribut restant : the winner is DOM

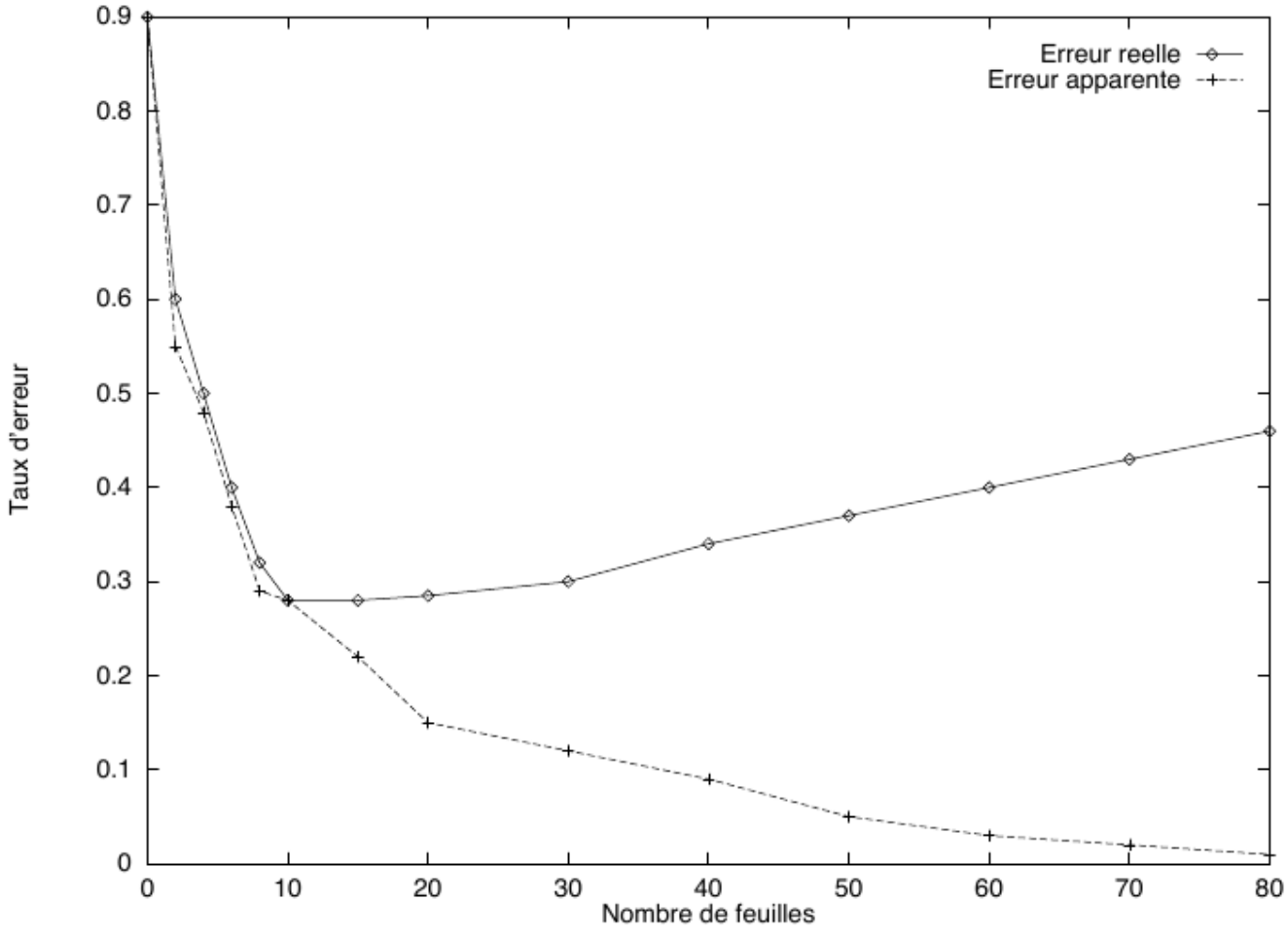
Choix du second test en position S_2 (BAL=F)

The winner si MPG

Arbre final obtenu



Erreur apparente nulle, mais...



Faible pouvoir prédictif (notion de sur-apprentissage – ou apprentissage par coeur)

Nécessité d'obtenir un arbre plus petit : élagage

Phase d'élagage de l'arbre de décision

Deux approches

- Eviter une trop grande croissance de l'arbre en arrêtant sa construction au bon moment (*early stopping* via ensemble de validation).
- Procéder en deux phases : construire l'arbre complètement, puis couper les branches qui dépassent !

Elagage de CART

Variations d'erreurs à minimiser

- T_0 un arbre de décision à élaguer, $\mathcal{T}$ l'ensemble de tous les arbres de décision obtenus à partir de T_0 en remplaçant certains noeuds internes par des feuilles.
- Pour chaque noeud interne p de T_0 :

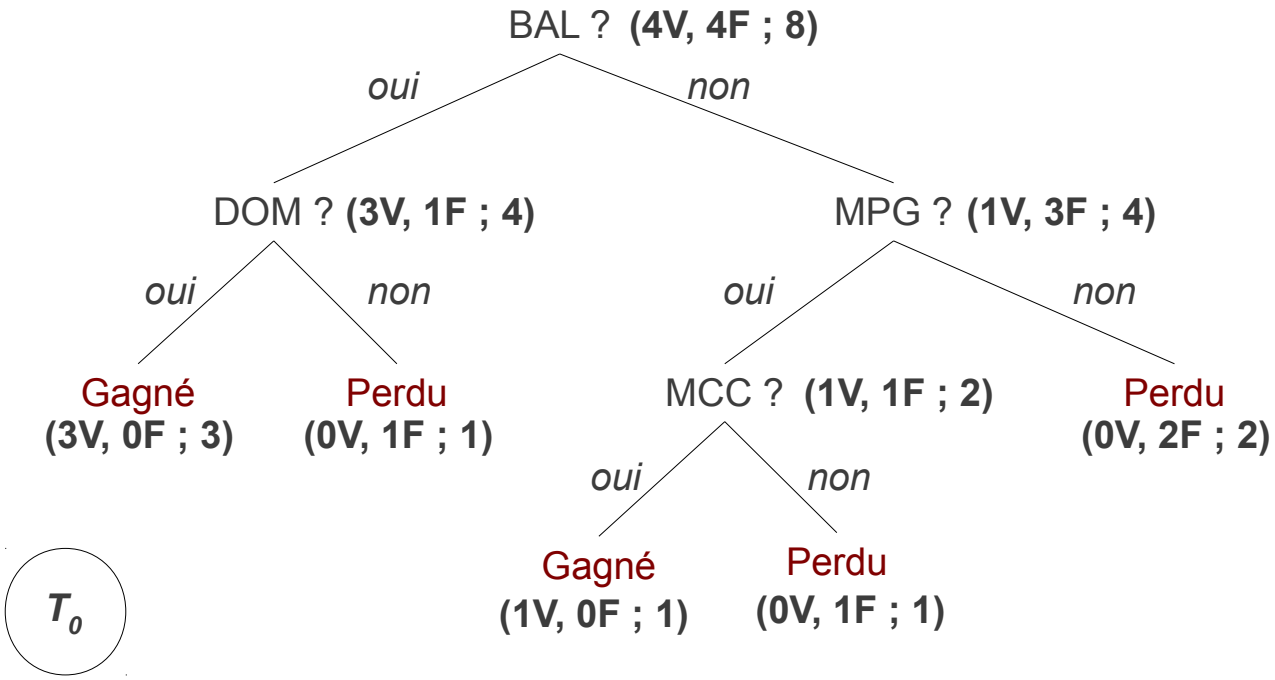
$$\alpha = \frac{\Delta R^S_{\text{emp}}}{|T_p| - 1}$$

où ΔR^S_{emp} est le nombre d'erreurs supplémentaires que commet l'arbre sur S lorsqu'on élague à la position p, et $|T_p| - 1$ mesure le nombre de feuilles supprimées.

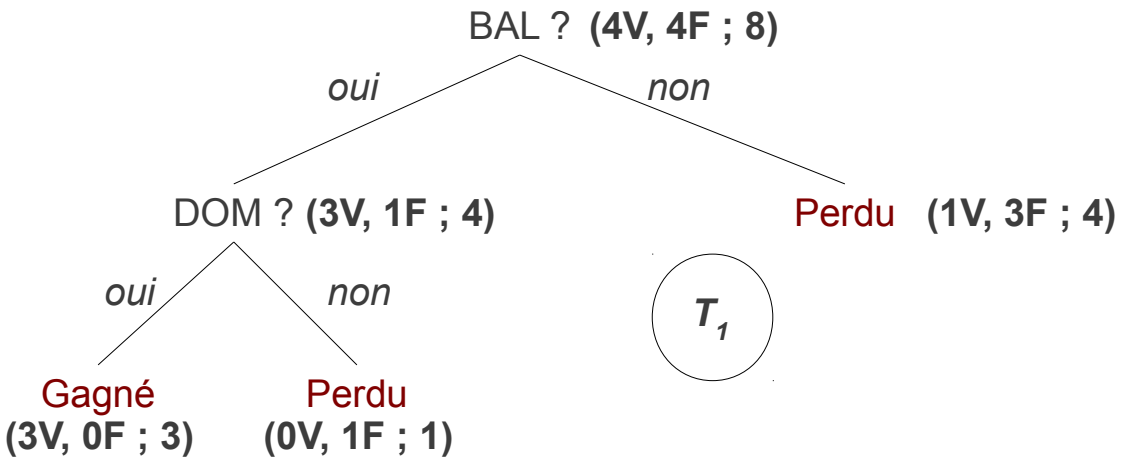
Processus itératif

- T_{i+1} obtenu à partir de T_i, auquel on coupe la branche qui permet un α minimal.
- Soit $T_0, \ldots, T_i, \ldots, T_t$ la suite obtenue, où T_t est réduit à une feuille.
- Sélection de l'arbre T_i dont le nombre d'erreurs calculées sur ensemble

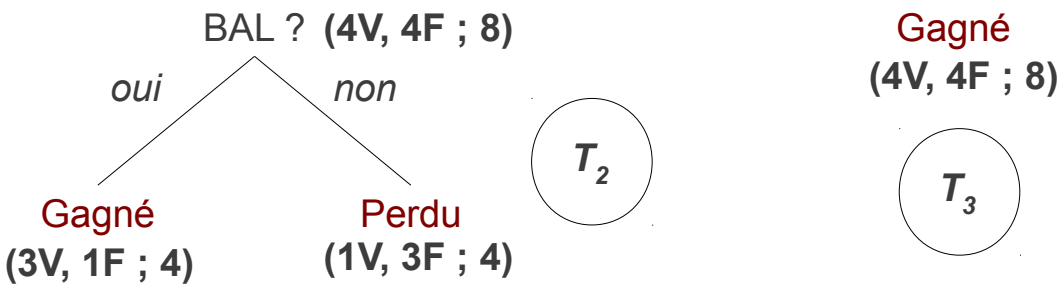
Elagage de CART sur l'exemple



Elagage de CART sur l'exemple



Elagage de CART sur l'exemple



Elagage de CART sur l'exemple (cont'd)

Ensemble de validation

Match dom.	Balance pos.	Mauvais climat	Match préc. gagné	Victoire
V	V	V	F	V
F	V	V	F	V
F	F	F	V	F
V	F	V	F	V

Calculs d'erreurs

- T_0 : 0 en apprentissage, $\frac{1}{2}$ en test
- T_1 : $\frac{1}{4}$ en apprentissage, $\frac{1}{2}$ en test : mêmes erreurs que T_0
- T_2 : $\frac{1}{2}$ en apprentissage, $\frac{1}{4}$ en test
- T_3 : $\frac{1}{2}$ en apprentissage, $\frac{1}{4}$ en test

Compléments

Autres catégories d'attributs

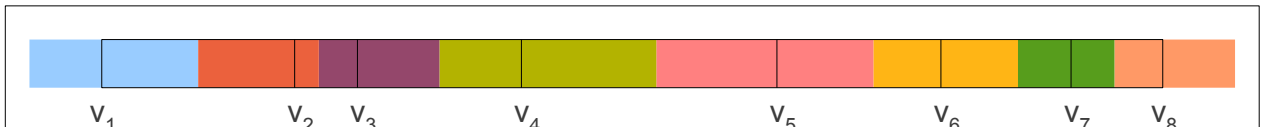
- Traitement en cas d'attributs *n*-aires et continus
- Traitement en cas de multi-classes
- Données manquantes ?

Attribut continu : partition de l'intervalle

- Attribut a, tests binaires de la forme $a < v$ ou $a \leq v$
- Soient $v_1, v_2, \ldots v_N$ valeurs prises par a dans S
- Exemple de test, pour $i = 1 \ldots N - 1$:

$$a < \frac{v_i + v_{i+1}}{2}$$

- Test sélectionné grâce à la notion de gain



Compléments

Autres catégories d'attributs

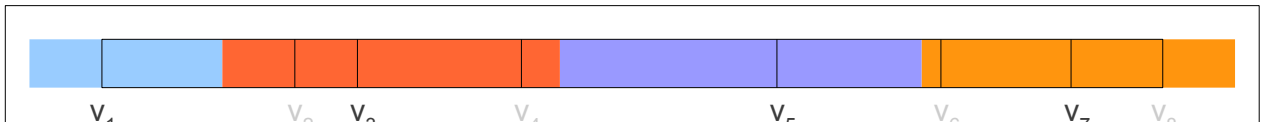
- Traitement en cas d'attributs n-aires et continus
- Traitement en cas de multi-classes
- Données manquantes ?

Attribut continu : partition de l'intervalle

- Attribut a, tests binaires de la forme $a < v$ ou $a \leq v$
- Soient $v_1, v_2, \ldots v_N$ valeurs prises par a dans S
- Exemple de test, pour $i = 1 \ldots N - 1$:

$$a < \frac{v_i + v_{i+1}}{2}$$

- Test sélectionné grâce à la notion de gain



Compléments : attributs n-aires

Généralisation des formules de gain

$$\text{Gain}_f(p, T) = f(S_p) - \sum_{j=1}^{N} P_j \times f(S_{pj})$$

- Privilège des attributs de grande arité
- Cas où attribut = clé identifiant chaque élément de S

Approche de C4.5 : Ratio de gain

Soit a d'arité N, prenant les valeurs $v_1, \ldots, v_N$:

$$\text{GainRatio} = \frac{\text{Gain}}{-\sum_{i=1}^{N} \frac{k_i}{|S|} \log \frac{k_i}{|S|}}$$

où k_i = nombre d'exemples de S pour lesquels $a = v_i$.

Compléments

Les trois grands algorithmes

- ID3 (Iterative Dichotomisor, [Quinlan79]) : sur variables qualitatives (discrimination)
- C4.5 [Quinlan93] : amélioration d'ID3 pour traitement des attributs continus et des valeurs manquantes
- CART [Breiman84] : généralisation à la régression et critère Gini remplace l'entropie. `sklearn.tree.DecisionTreeClassifier`

Instabilité : variance importante

(Mais faible biais !) Sur données réelles :

- choix d'un attribut plutôt qu'un autre : limite serrée !
- influence majeure si proche de la racine

Solutions : agréger sur le hasard

- *Bagging* (Boostrap Aggregating)
- Random Forests

Outline

Les k-plus proches voisins (k-NN) : pas de modèle !

Méthode d'apprentissage supervisé, multi-classes, avec la définition d'une distance entre les points $d : \mathcal{X} \to \mathbb{R}^+$ (ex. distance euclidienne)

Le problème : Apprendre f qui *généralise* au mieux

En entrée L'échantillon iid $S_{\text{train}} = \{(x_i, y_i)\}_{i=1}^n$

En sortie Un *classifieur* $f : \mathcal{X} \to \mathcal{Y}$ tel que $f = \text{argmin}_h R(h)$

Intuition de la méthode

Pour chaque nouvel $x \in \mathcal{X}$, calculer ses k plus proches voisins dans S selon une distance d à définir, et choisir pour x l'étiquette majoritaire parmi ces k voisins.

Algorithme en $\mathcal{O}(n)$

En parcourant chaque $x_i \in \mathcal{X}$, extraire $V_k(x) = \{r_1, \cdots, r_k\}$ = les indices i dans S des k plus proches voisins de x

$$\text{Calculer } f(x) = \underset{y \in \mathcal{Y}}{\text{argmax}} \sum_{j=1}^{k} \mathbb{I}(y = y_{r_j})$$

Illustration des k-NN (1)

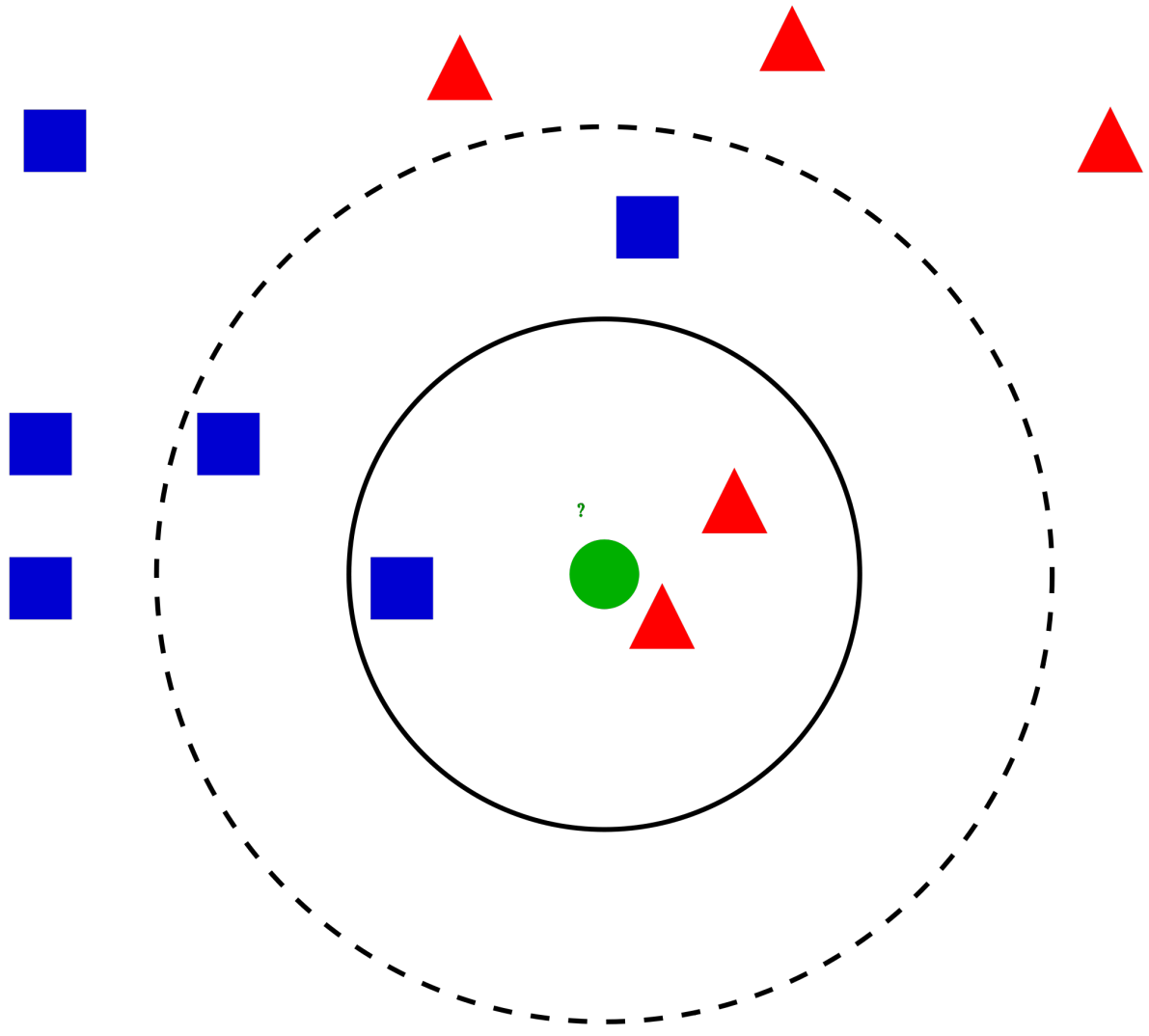
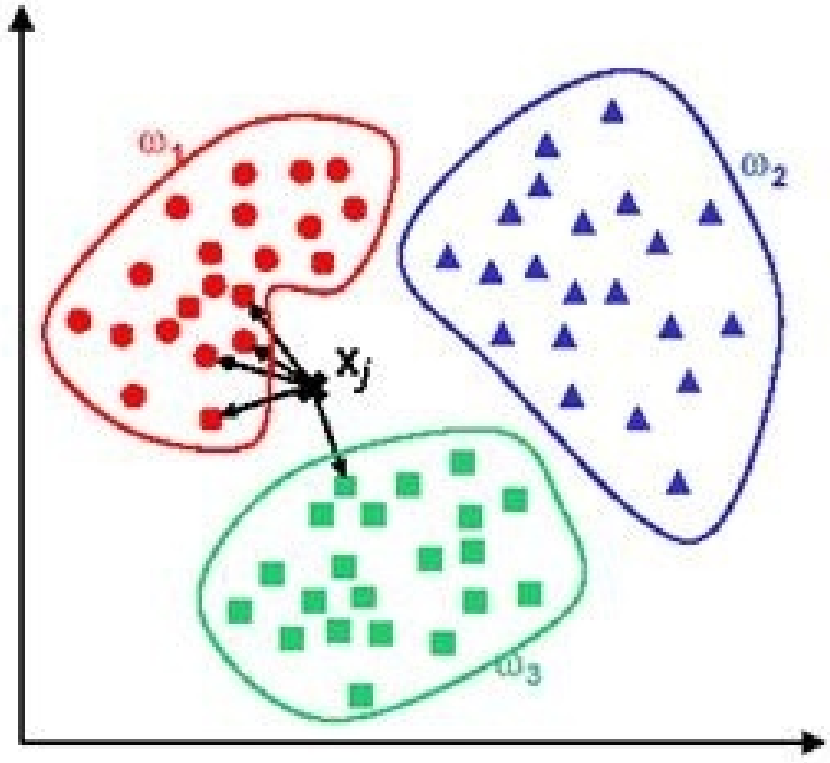
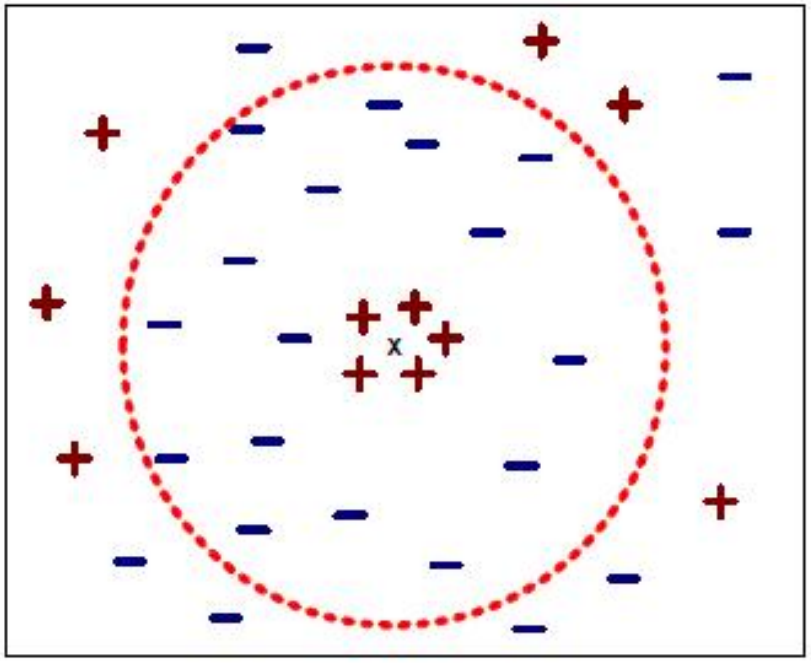


Illustration des k-NN (2)





Pro's and con's des k-NN

Avantages

- Simple, non-paramétrique
- Choix de k facile, par validation croisée
- Borne sur l'erreur en généralisation par rapport à celle de Bayes
  $e^* < e^k < e^{k-1} < \cdots < 2e^*$

Inconvénients

- Overfitting si k est grand, underfitting si k petit
- Forte dépendance à la distance choisie
- Pas un vrai modèle