

# Arbres de Décision et Forêts Aléatoires

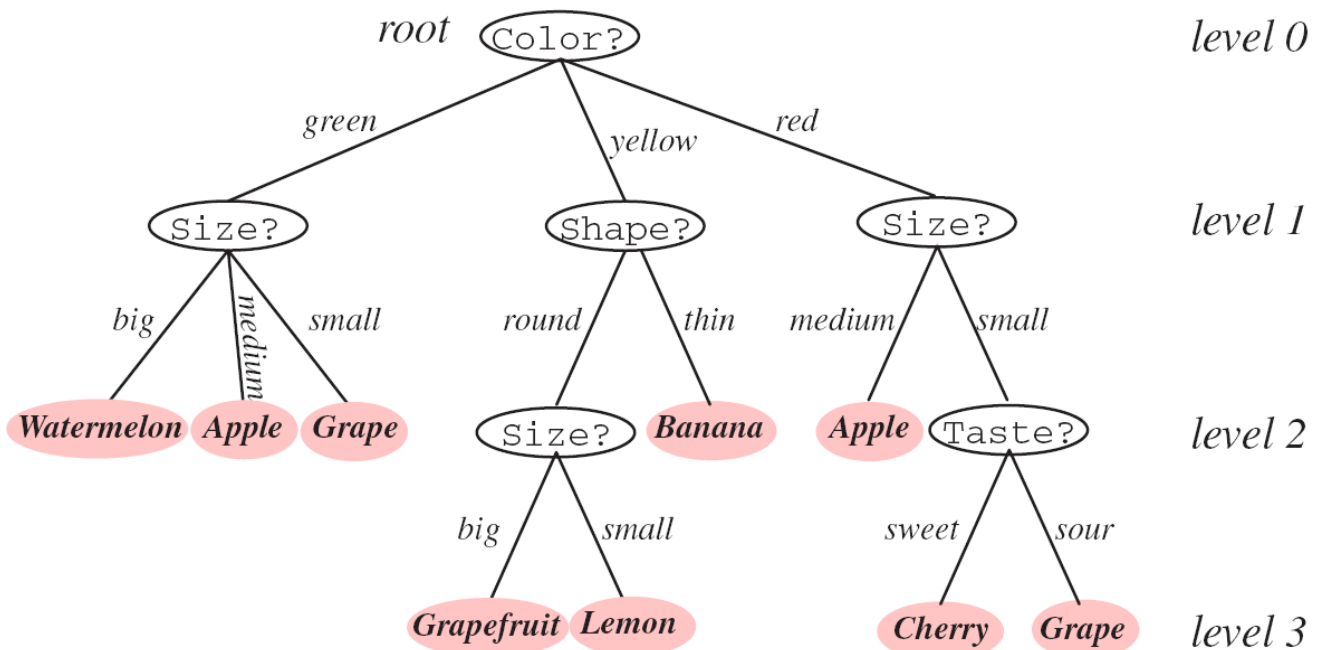
Pr. Fabien Moutarde  
Centre de Robotique (CAOR)  
MINES ParisTech (ENSMP)  
PSL Research University

Fabien.Moutarde@mines-paristech.fr

<http://people.mines-paristech.fr/fabien.moutarde>

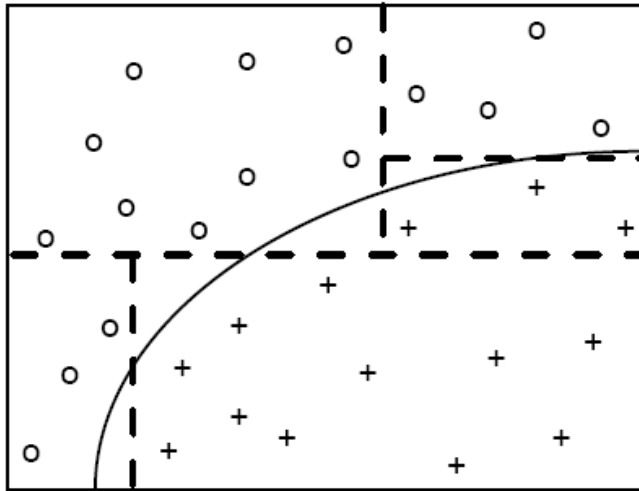
Arbres de décision et Forêts aléatoires Pr. Fabien Moutarde, CAOR, MINES ParisTech, PSL Fév.2017 1

## Qu'est-ce qu'un arbre de décision ?



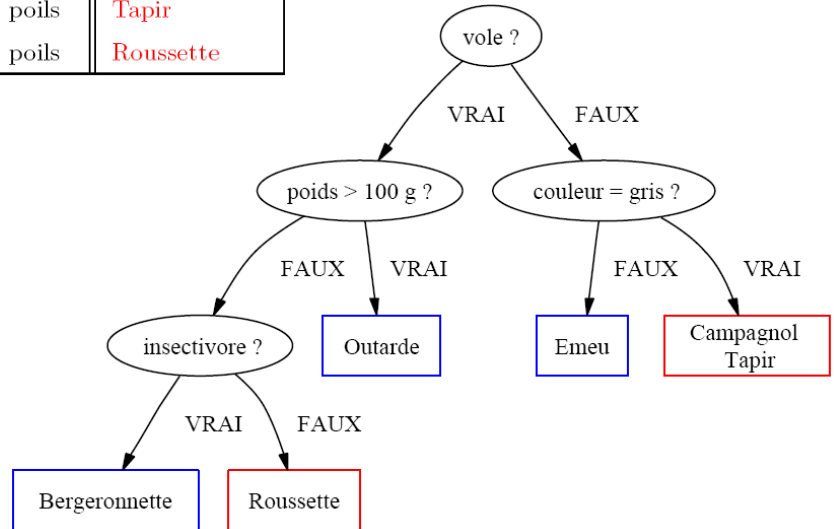
## Classification par une série de tests

Arbres de décision et Forêts aléatoires Pr. Fabien Moutarde, CAOR, MINES ParisTech, PSL Fév.2017 2



- **Décomposition du problème de classification en une suite de tests correspondant à une partition de l'espace des données en sous-régions homogènes en terme de classe**

vole	poids	couleur	alimentation	peau	animal
OUI	1 kg	roux	granivore	plumes	Outarde
OUI	20 g	gris et jaune	insectivore	plumes	Bergeronnette
NON	100 kg	noir et blanc	omnivore	plumes	Emeu
NON	5 g	gris	granivore	poils	Campagnol
NON	40 kg	gris	herbivore	poils	Tapir
OUI	60 g	noir	frugivore	poils	Roussette



**Est-ce le meilleur arbre ?**

## Induction de l'arbre à partir d'une base d'exemples d'apprentissage

- Recherche exhaustive dans l'ensemble de tous les arbres évidemment computationnellement impossible

→ approche *réursive* pour construire l'arbre :

### construire-arbre (X)

SI tous les points de X sont de même classe,  
créer une feuille associée à cette classe

SINON

- choisir (selon critère !) le *meilleur couple (attribut; test)* pour créer un nœud
- ce test sépare X en 2 parties Xg et Xd
- construire-arbre (Xg)
- construire-arbre (Xd)

- **Mesure de l'hétérogénéité du nœud candidat :**
  - entropie (ID3, C4.5)
  - indice Gini (CART)
  - indice d'erreur
- **Entropie :  $H = -\sum_k ( p(w_k) \log_2(p(w_k)) )$  avec  $p(w_k)$  proba de la classe  $w_k$  (estimée par proportion  $N_k/N$ )**
  - minimum (=0) si une seule classe représentée
  - maximum (=log<sub>2</sub>(nb\_classes)) si classes équi-réparties
- **Indice Gini :  $Gini = 1 - \sum_k p^2(w_k)$**
- **Indice d'erreur :  $Er = 1 - \max_k (p(w_k))$**

- Soit un test  $T$  à  $m$  alternatives et divisant le nœud  $N$  en  $m$  sous-nœud  $N_j$
  - Soit  $I(N_j)$  les mesures d'hétérogénéité (entropie, Gini, ...) des sous-nœuds, et  $p(N_j)$  les proportions des éléments de  $N$  dirigés vers  $N_j$  par le test  $T$
- le gain d'homogénéité/information apporté par le test  $T$  est  $Gain(N, T) = I(N) - \sum_j p(N_j) I(N_j)$
- À chaque nœud, choix du test maximisant le gain (ou éventuellement le « rapport des gains »  $G(N,T)/H(T)$ , utilisé par C4.5 pour éviter biais vers  $m$  grand)

- Les exemples d'apprentissage sont en Nb FINI  
→ idem pour le nb de valeurs effectivement atteintes par chaque attribut continu

Soit l'arbre  $T$ , et  $v$  un de ses nœuds, et :

- $MC(T,v)$  = nb d'exemples Mal Classés par  $v$  dans  $T$
- $MC_{ela}(T,v)$  = nb d'exemples Mal Classés par  $v$  dans l'arbre  $T$  élagué à  $v$
- $n(T)$  = nb de feuilles de  $T$
- $nt(T,v)$  = nb de feuilles du sous-arbre de  $T$  sous nœud  $v$

ALORS on prend comme critère à minimiser :

$$w(T, v) = (MC_{ela}(T, v) - MC(T, v)) / (n(T) * (nt(T, v) - 1))$$

→ Prise en compte simultanée de l'erreur et de la complexité de l'arbre

## Algorithme d'élagage

Elaguer ( $T_{max}$ ) :

$K \leftarrow 0$

$T_k \leftarrow T_{max}$

TANT QUE  $T_k$  a plus d'un nœud FAIRE

POUR chaque nœud  $v$  de  $T_k$  FAIRE

calculer  $w(T_k, v)$  sur exemples appr. ou valid.

FIN\_POUR

choisir  $v_m$  = tel que  $w(T_k, v)$  soit minimum

$T_{k+1}$ :  $T_k$  où  $v_m$  a été remplacé par une feuille

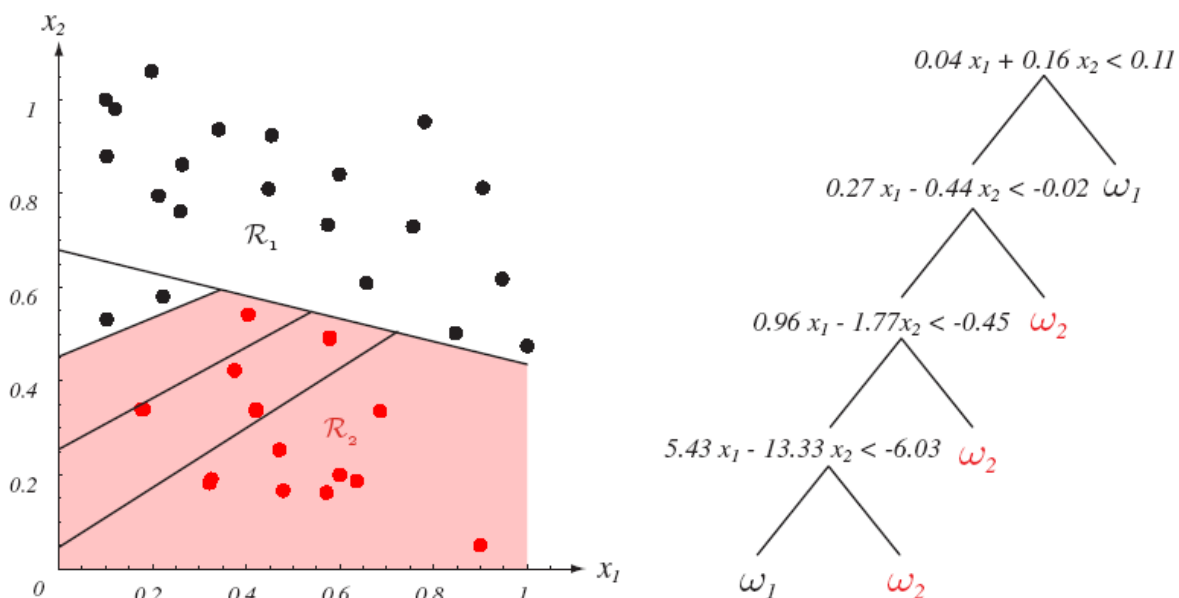
$k \leftarrow k+1$

FIN\_TANT\_QUE

Pour finir, on choisit parmi  $\{T_{max}, T_1, \dots, T_n\}$  l'arbre qui a la plus petite erreur de classification sur l'ensemble de validation

- **ID3 (Inductive Decision Tree, Quinlan 1979) :**
  - arbres « de discrimination » (ie variables uniquement qualitatives)
  - critère d'hétérogénéité = entropie
- **C4.5 (Quinlan 1993) :**
  - Amélioration ID3, permettant notamment arbres « de régression » (gestion des variables continues) et valeurs manquantes
- **CART (Classification And Regression Tree, Breiman et al. 1984) :**
  - critère d'hétérogénéité = Gini

## Variante : arbres multivariés



- **Avantages**
  - Facilité à manipuler des données « symboliques »
  - OK avec variables d'amplitudes très différentes
  - Multi-classe par nature
  - Interprétabilité de l'arbre !
  - Identification des inputs « importants »
  - Classification très efficace (en particulier sur inputs de grande dimension)
- **Inconvénients**
  - Sensibilité au bruit et points aberrants
  - Stratégie d'élagage délicate

### **Principe : l'union fait la force**

- Une « forêt » = un ensemble d'arbres
- Forêt Aléatoire :
  - apprendre grand nombre  $T$  (~ qlq 10aines ou 100aines) d'arbres *simples*
  - utilisation par *vote des arbres* (classe majoritaire, voire probabilités des classes par % des votes) si classification, ou *moyenne des arbres* si régression

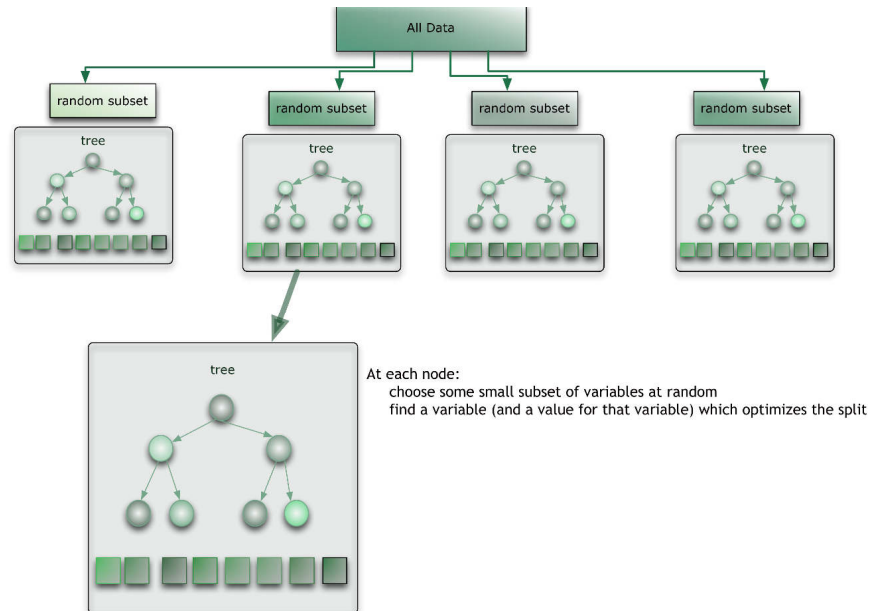
**Algo proposé en 2001 par Breiman & Cutter**



**But = obtenir des arbres les + décorrelés possible**

→ chaque arbre appris sur un sous-ensemble (~2/3) aléatoire différent de  $s$  exemples d'apprentissage

→ chaque nœud de chaque arbre choisi comme « split » optimal parmi  $k$  variables tirées aléatoirement dans les entrées (avec  $k \ll d$  la dim des entrées)



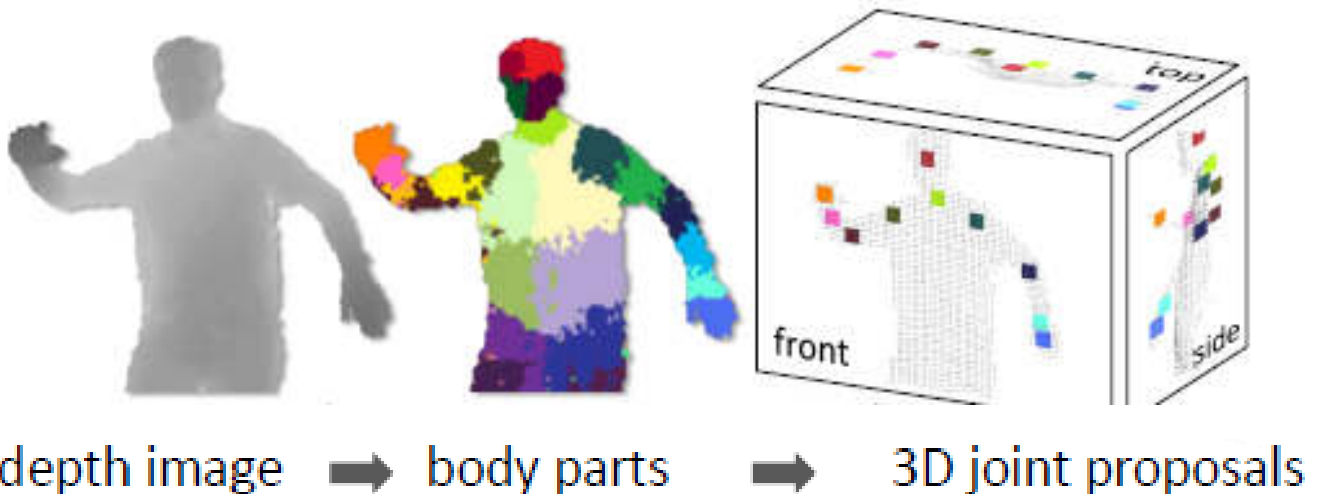
- Chaque arbre appris avec algo **CART sans élagage**
- On limite fortement la profondeur  $p$  des arbres (~ 2 à 5)

$Z = \{(x_1, y_1) \dots (x_n, y_n)\}$  base d'apprentissage, chaque  $x_i$  de dimension  $d$

Pour  $t = 1 \dots T$  ( $T = \text{nb d'arbres de la forêt}$ )

- Tirer aléatoirement (avec remise)  $m$  exemples dans  $Z$  ( $\rightarrow Z_t$ )
- Apprendre un arbre sur  $Z_t$ , avec **CART modifié** pour randomiser choix de variables : chaque nœud cherché uniquement parmi  $k$  variables tirées aléatoirement parmi les  $d$  dimensions ( $k \ll d$ , typiquement  $k \sim \sqrt{d}$ )

- **Squelettisation de personne avec Kinect**



## Avantages et inconvénient des Forêts Aléatoires

- **Avantages**

- **Reconnaissance TRES RAPIDE**
- **Multi-classes par nature**
- **Efficace sur inputs de grande dimension**
- **Robustesse aux outliers**

- **Inconvénients**

- **Apprentissage souvent long**
- **Valeurs extrêmes souvent mal estimées dans cas de régression**