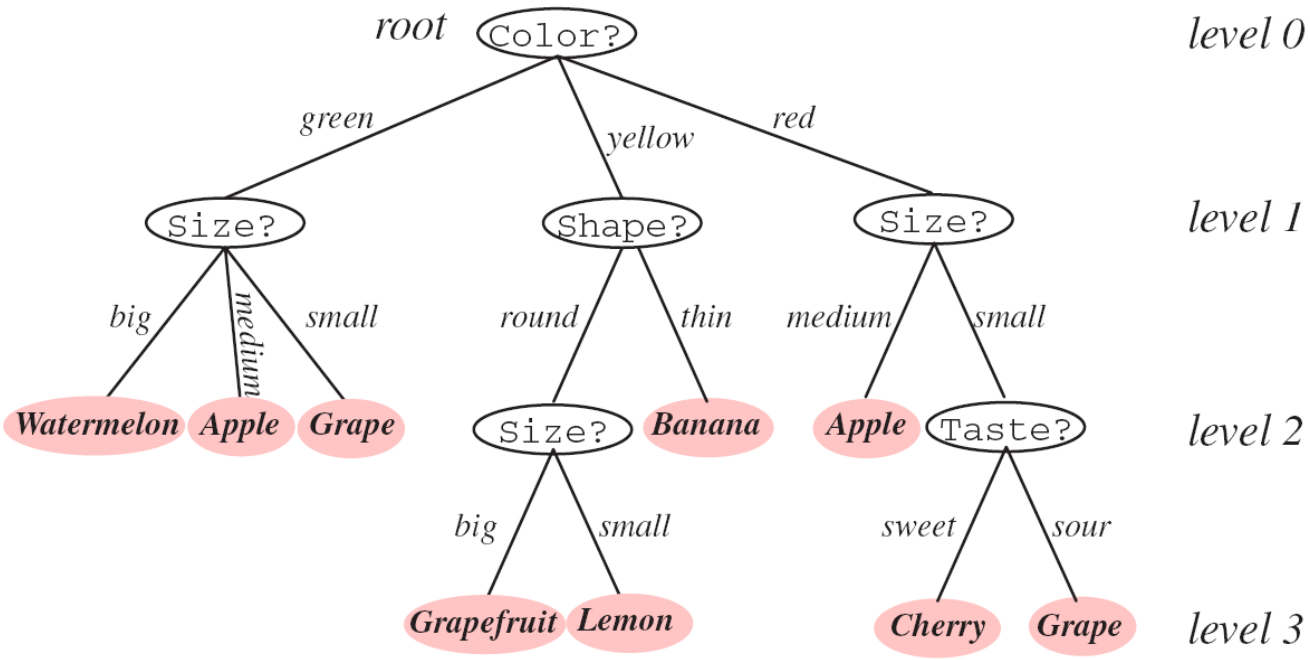
Arbres de Décision et Forêts Aléatoires

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

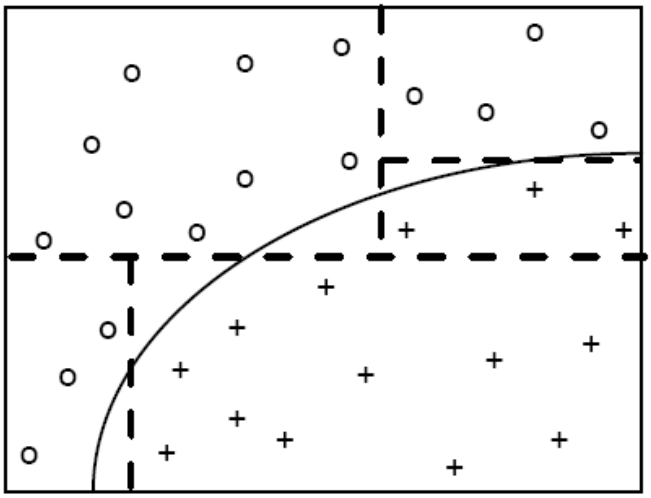
Fabien.Moutarde@mines-paristech.fr
http://people.mines-paristech.fr/fabien.moutarde

Qu'est-ce qu'un arbre de décision ?



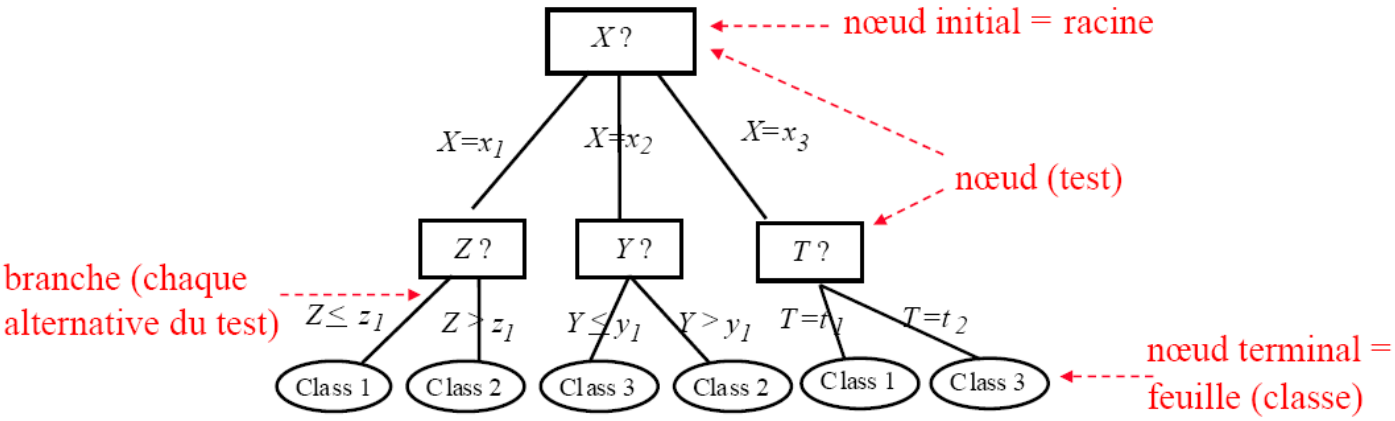
Classification par une série de tests

Principe général des arbres de décision



- **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**

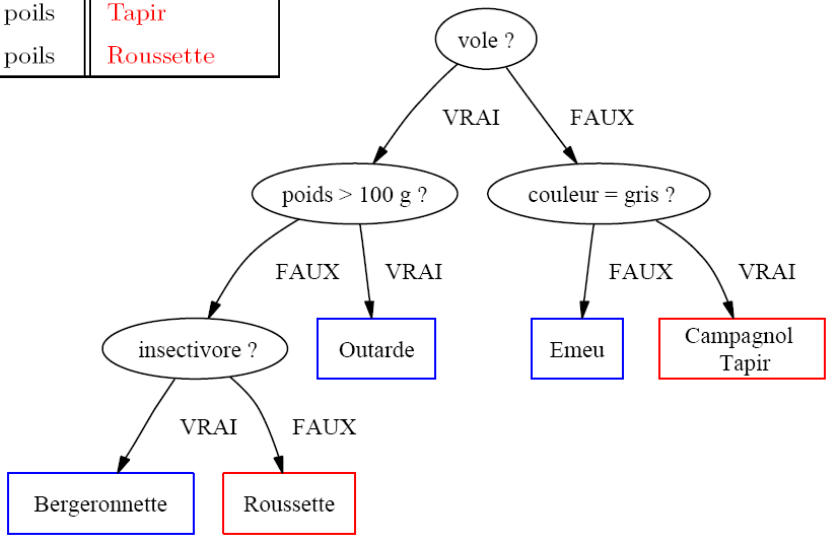
Exemple d'arbre de décision



- **Règle de classification :** aller de la racine à une feuille en effectuant les tests des nœuds
- **Classe d'une feuille :** classe majoritaire parmi les exemples d'apprentissage appartenant à cette feuille

Problématique de l'induction de l'arbre

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 *récursive* 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
   - contruire-arbre(Xg)
   - construire-arbre(Xd)
```

Critère de choix attribut et test

- **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 (=$\log_2$(nb_classes)) si classes équi-réparties
- **Indice Gini : $Gini = 1 - \sum_k p^2(w_k)$**
- **Indice d'erreur : $Er = 1 - \max_k(p(w_k))$**

Gain d'homogénéité apporté par un test

- **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**

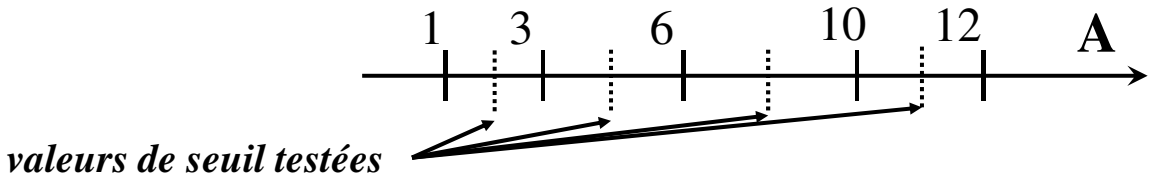
➔ le gain d'homogénéité/information apporté par le test T est $Gain(N,T) = I(N) - \sum_j p(N_j) I(N_j)$

➔ 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)

Tests sur les attributs *continus*

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

➔En pratique, tri des exemples par valeur croissante de l'attribut continu et examen d'au maximum N-1 seuils (typiquement les médianes entre valeurs successives croissantes :par exemple si valeurs de A atteintes sur exemples d'apprentissage sont 1;3;6;10;12, on considérera les tests A>1.5;A>4.5;A>8;A>11)



Critères d'arrêt et élagage

- **Règles « évidentes » :**
  - **tous les exemples du nœud sont de même classe**
  - **tous exemples du nœud ont mêmes valeurs de variables**
  - **l'hétérogénéité des nœuds ne diminue plus**
  - **nb d'exemples dans le nœud < seuil minimal**
- **Contrôle des performances de généralisation (sur base de validation indépendante)**
- **Elagage a posteriori : supprimer des branches peu représentatives et nuisant à généralisation (parcours bottom-up en « remontant » d'un niveau tant que cela diminue erreur en généralisation)**

Critère pour l'élagage a posteriori

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←0

$T_k \leftarrow T_{max}$

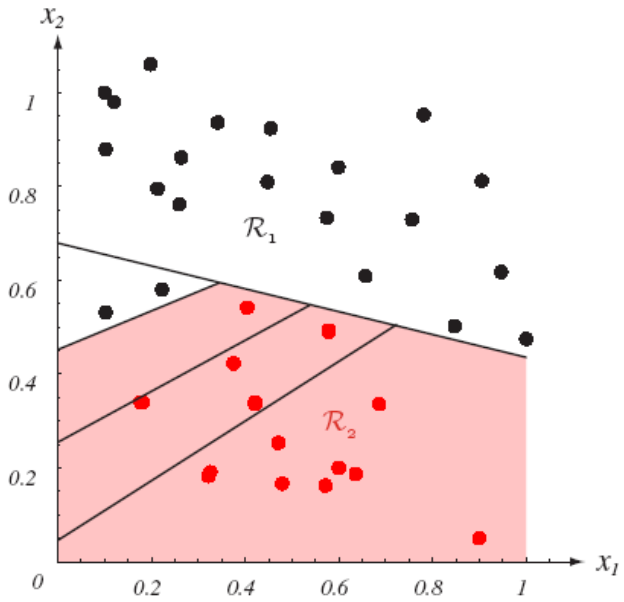
TANT QUE T_k a plus d'un nœud FAIRE
  POUR chaque nœud v de T_k FAIRE
    calculer $w(T_k,v)$ sur exemples appr. ou valid.
  FIN_POUR
  choisir v_m = tel que $w(T_k,v)$ soit minimum
  T_{k+1}: T_k où v_m a été remplacé par une feuille
  k←k+1
FIN_TANT_QUE

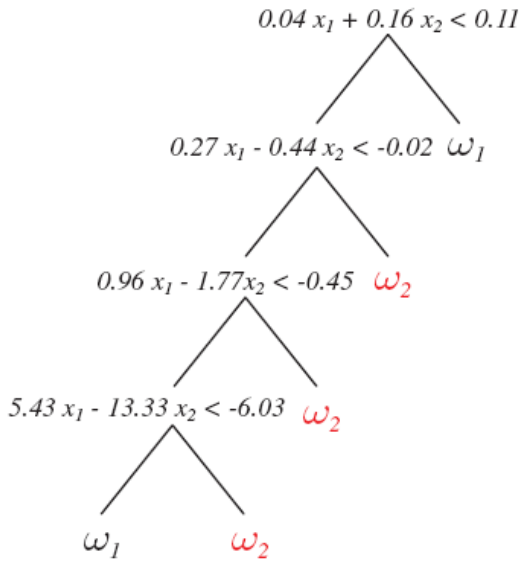
Pour finir, on choisit parmi {Tmax, T1, … Tn} l'arbre qui a la plus petite erreur de classification sur l'ensemble de *validation*

Synthèses des divers algos d'induction

- **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





Avantage/inconvénients des arbres de décision

- **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

Forêts Aléatoires (Random Forests)

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

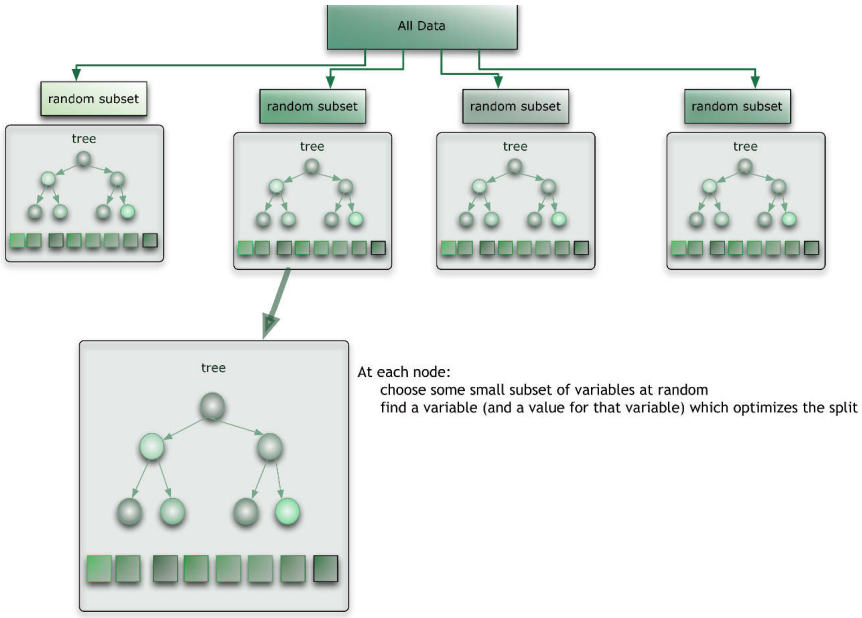
Algo proposé en 2001 par Breiman & Cutter

Apprentissage de Forêt Aléatoire

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<<d la dim des entrées)**



Apprentissage de Forêt Aléatoire (2)

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

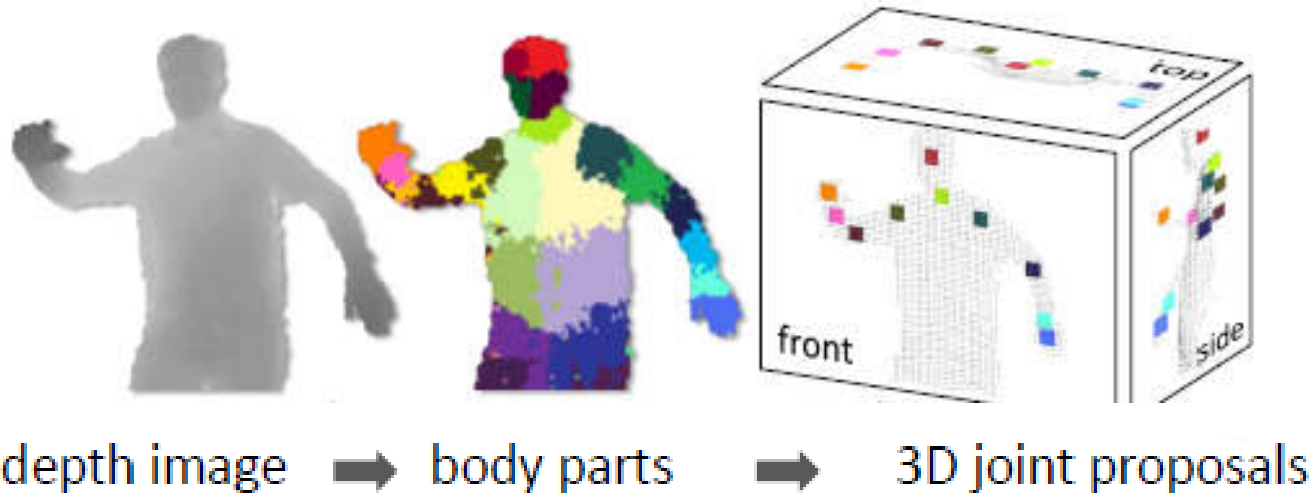
$Z = \{(x_1, y_1). . .(x_n, y_n)\}$ base d'apprentissage, chaque x_i de dimension d

Pour t = 1 . . . T (T = nb d'arbres de la forêt)

- Tirer aléatoirement (avec remise) m exemples dans Z (→ 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<<d, typiquement k~√d)

« Success story »

- **Squelettisation de personne avec Kinect**



Avantages et inconvénient des Forêts Aléatoires

- **<u>Avantages</u>**
  - **Reconnaissance TRES RAPIDE**
  - **Multi-classes par nature**
  - **Efficace sur inputs de grande dimension**
  - **Robustesse aux outliers**

- **<u>Inconvénients</u>**
  - **Apprentissage souvent long**
  - **Valeurs extrêmes souvent mal estimées dans cas de régression**