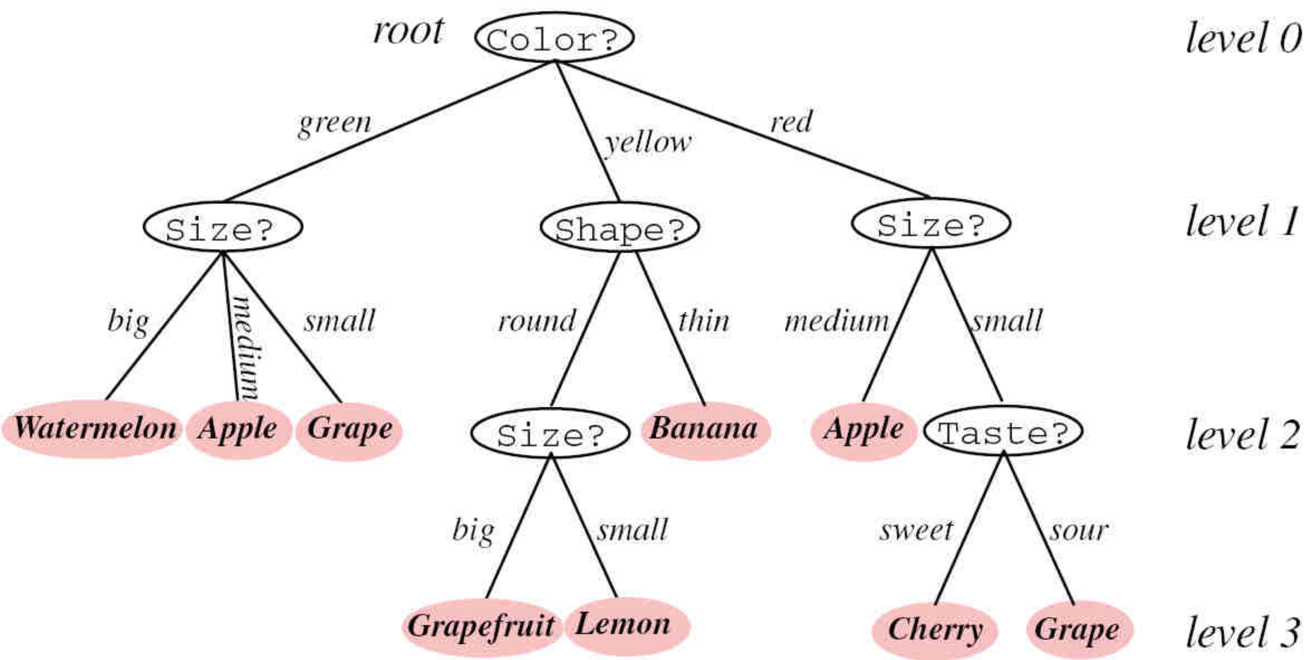
Decision Trees and Random Forests

Pr. Fabien MOUTARDE
Center for Robotics
Mines Paris
PSL Université

`Fabien.Moutarde@minesparis.psl.eu`
`http://people.minesparis.psl.eu/fabien.moutarde`

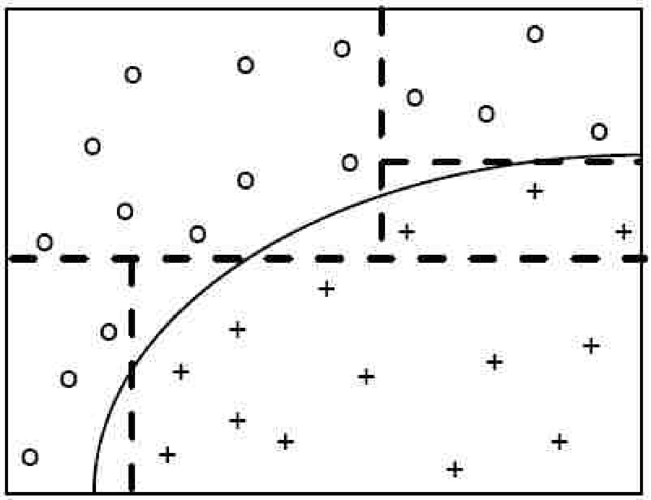
What is a Decision Tree?



Classification by a tree of tests

MINES PARIS | PSL

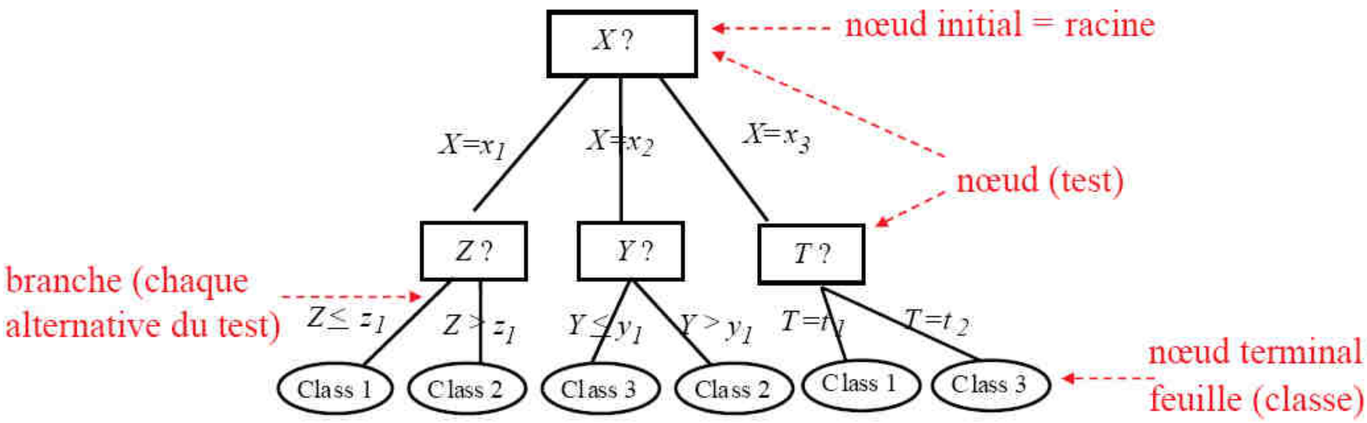
General principle of Decision Trees



Classification by sequences of tests organized in a tree, and corresponding to a *partition of input space into class-homogeneous sub-regions*

MINES PARIS | PSL

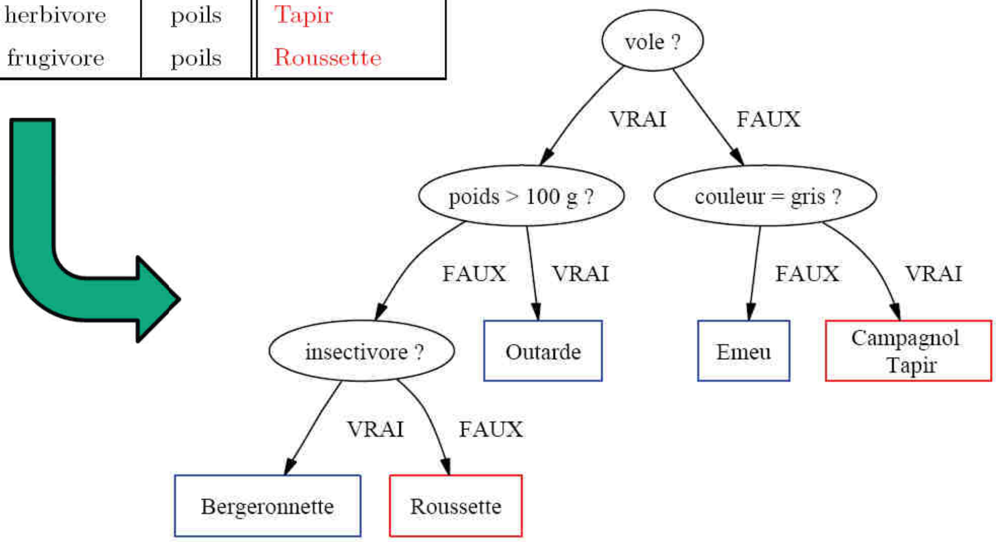
Example of Decision Tree



- **Classification rule:** go from root to a leaf by evaluating the tests in nodes
- **Class of a leaf:** class of the majority of training examples "arriving" to that leaf

"Induction" of the tree?

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



Is it the best tree??

Principle of binary Decision Tree induction from training examples

- **Exhaustive search in the set of all possible trees is computationally intractable**

→ *Recursive* approach to build the tree:

```
build-tree(X)
 IF all examples "entering" X are of same class,
    THEN build a leaf (labelled with this class)
 ELSE
  - choose (using some criterion!) the BEST
     (attribute;test) couple to create a new node
  - this test splits X into 2 sub-trees Xl and Xr
  - build-tree(Xl)
  - build-tree(Xr)
```

Criterion for choosing attribute and test

- ***Measure of <u>heterogeneity</u>*** **of candidate node:**
  - entropy (ID3, C4.5)
  - Gini index (CART)

- **<u>Entropy</u>:** $H = -\Sigma_k (p(w_k) \log_2(p(w_k)))$ **with** $p(w_k)$ **probability of class** w_k **(estimated by proportion** N_k/N**)**
  → **minimum (=0) if only one class is present**
  → **maximum (=**$\log_2$**(#_of_classes)) if equi-partition**

- **Gini index:** $Gini = 1 - \Sigma_k p^2(w_k)$

Homogeneity gain by a test

- **Given a test T with m alternatives and therefore orienting from node N into m "sub-nodes"** N_j
- **Let** $I(N_j)$ **be the heterogeneity measures (entropy, Gini, …) of sub-nodes, and** $p(N_j)$ **the proportions of elements directed from N towards** N_j **by test T**

→ **the homogeneity gain brought by test T is** $Gain(N,T) = I(N) - \Sigma_j p(N_j) I(N_j)$

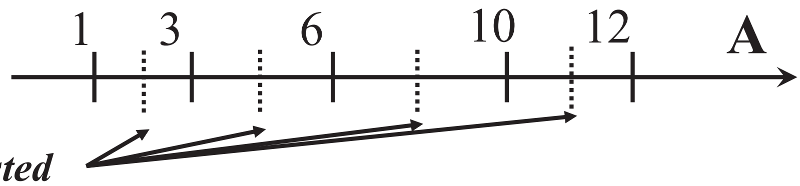
→ **Simple algo = choose the test maximizing this gain**
(or, in the case of C4.5, the "relative" gain G(N,T)/I(N), to avoid bias towards large m)

Tests on *continuous*-valued attributes

- Training set is FINITE → idem for the # of values taken ON TRAINING EXAMPLES by any attribute, even if continuous-valued

➔ In practice, examples are sorted by increasing value of the attribute, and only N-1 potential threshold values need to be compared (typically, the medians between successive increasing values)

For example, if values of attribute A for training examples are 1;3;6;10;12, the following potential tests shall be considered: A>1.5;A>4.5;A>8;A>11)



Threshold values tested

Stopping criteria and pruning

- "Obvious" stopping rules:
  - all examples arriving in a node are of same class
  - all examples arriving in a node have equal values for each attribute
  - node heterogeneity stops decreasing
- Natural stopping rules:
  - # of examples arriving in a node < minimum threshold
  - Control of generalization performance (on independent validation set)
- A posteriori pruning: remove branches that are impeding generalization (bottom-up removal from leaf while generalization error does not decrease)

Criterion for a posteriori pruning of the tree

Let T be the tree, v one of its nodes, and:

- IC(T,v) = # of examples Incorrectly Classified by v in T
- IC_{ela}(T,v) = # of examples Incorrectly Classified by v in T' = T <u>pruned by changing v into a leaf</u>
- n(T) = total # of leaves in T
- nt(T,v) = # of leaves in the sub-tree below node v

THEN the criterion chosen to minimize is:

`w(T,v) = (IC_ela(T,v)-IC(T,v))/(n(T)*(nt(T,v)-1))`

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

→Take simultaneously into account error rate and tree complexity

Pruning algorithm

```
Prune(T_max):
K←0
T_k←T_max
WHILE T_k has more than 1 node, DO
  FOR_EACH node v of T_k DO
   compute w(T_k,v) on train. (or valid.) examples
  END_FOR
  choose node v_m that has minimum w(T_k,v)
  T_k+1: T_k where v_m was replaced by a leaf
  k←k+1
END_WHILE
```

Finally, select among {Tmax, T1, … Tn} the pruned tree that has the smallest classification error on the *validation set*

Names of variants of Decision Tree variants

- **ID3 (Inductive Decision Tree, Quinlan 1979):**
  - only "discrimination" trees (i.e. for data with all attributes being <u>qualitative</u> variables)
  - heterogeneity criterion = entropy
- **C4.5 (Quinlan 1993):**
  - Improvement of ID3, allowing "regression" trees (ie continuous-valued attribute), and handling missing values
- **CART (Classification And Regression Tree, Breiman et al. 1984):**
  - heterogeneity criterion = Gini

Hyper-parameters for Decision Trees

- **Homogeneity criterion (entropy or Gini)**
- **Recursion stop criteria:**
  - **Maximum depth of tree**
  - **Minimum # of examples associated to each leaf**
- **Pruning parameters**

Pros and cons of Decision Trees

- <u>Advantages</u>
  - Easily manipulate "symbolic"/discrete-valued data
  - OK even with variables of totally ≠ amplitudes (no need for explicit normalization)
  - Multi-class BY NATURE
  - INTERPRETABILITY of the tree!
  - Identification of "important" inputs
  - Very efficient classification (especially for very-high dimension inputs)
- <u>Drawbacks</u>
  - High sensitivity to noise and "erroneous outliers"
  - Pruning strategy rather delicate

Random (decision) Forests *[Forêts Aléatoires]*

<u>Principle:</u> "*Strength lies in numbers*"

[en français, "L'union fait la force"]

- A forest = a set of trees
- <u>Random Forest:</u>
  - Train a large number T (~ few 10s or 100s) of *simple* Decision Trees
  - Use a *vote of the trees* (majority class, or even estimates of class probabilities by % of votes) if classification, or an *average of the trees* if regression

Algorithm proposed in 2001 by Breiman & Cutter

Theoretical background of "*ensembling*" methods

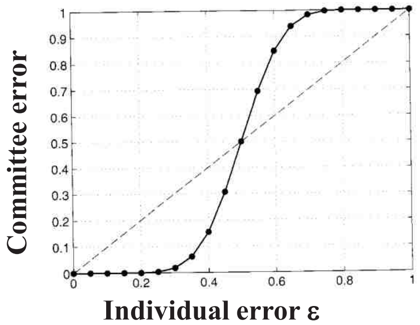
Set-up a "committee of experts"

each one can be wrong, but combining opinions increases the chance to obtain correct prediction!

Theoretical justification:

– suppose N ***independent*** classifiers, each with same error rate $E_{gen} = \varepsilon$

– decision by a "majority" vote is wrong if and only if more than half of the committee is wrong ➔ $$Error_{committee} = \sum_{k=N/2}^{N} C_k^N \varepsilon^k (1-\varepsilon)^{N-k}$$



Spectacular improvement of decision (under condition that ε<0.5!!)… …and the larger N (# of experts), the bigger the improvement

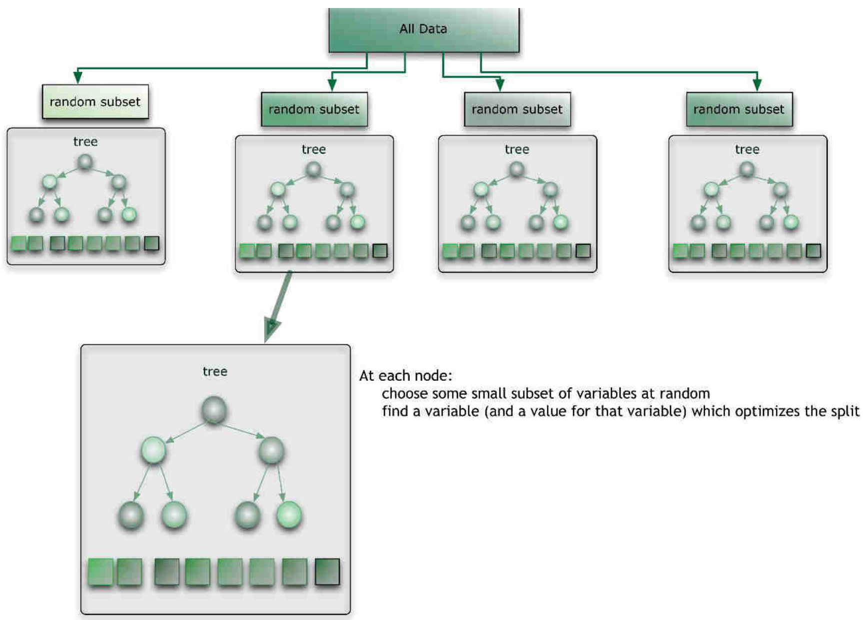
"wisdom of the crowd" (?)

Learning of a Random Forest

Goal= obtain trees *as decorrelated as possible*

➦ each tree is learnt on a *random* different subset (~2/3) of the whole training set

➦ each node of each tree is chosen as an optimal "split" among only k variables *randomly* chosen from all d inputs (and k<<d)



Training algorithm for Random Forest

- **Each tree is learnt using CART *without pruning***
- **The maximum depth p of the trees is usually strongly limited (~ 2 à 5)**

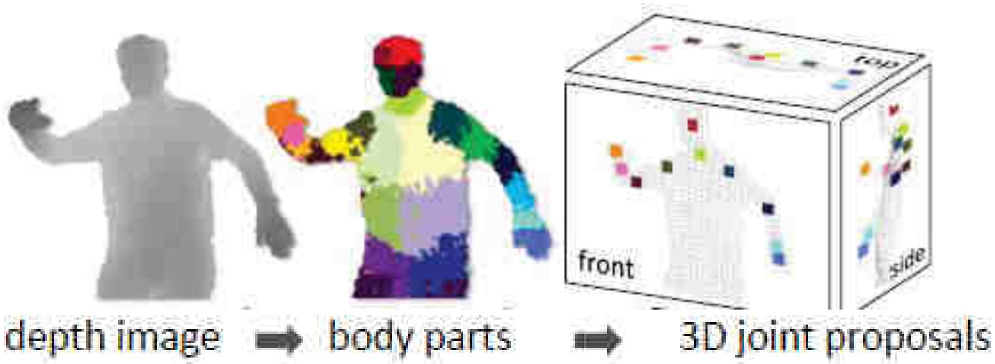
Z ={ (x_1,y_1),…,(x_n,y_n)} training set, each x_i of dimension d

FOR t = 1,…,T (T = # of trees in the forest)

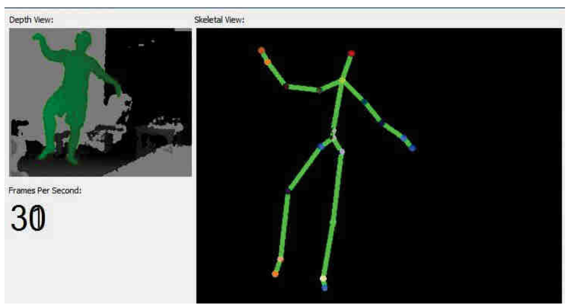
- **Randomly choose m examples in Z (→ Z_t)**
- **Learn a tree on Z_t, with *<u>CART modified for randomizing variables choice</u>*: each node is searched as a test on one of <u>ONLY k variables randomly chosen</u> among all d input dimensions (k<<d, typically k~√d)**

RdF "Success story"

"Skeletonization" of persons (and movement tracking) with Microsoft Kinect™ depth camera



Algo of Shotton et al. <u>using RDF</u> for labelling body parts



Hyper-parameters for Random Forests

- The number of trees
- Maximum depth of trees
- The size of randomized subset of training examples
- The proportion K/D of attributes considered for inference of each tree

Pros and Cons of Random Forests

- <u>Advantages</u>
  - VERY FAST recognition
  - Multi-class by nature
  - Efficient on large-dimension inputs
  - Robustness to outliers
- <u>Drawbacks</u>
  - Training often rather long
  - Extreme values often incorrectly estimated in case of regression