Arbres de décision

Module 5

Objectifs

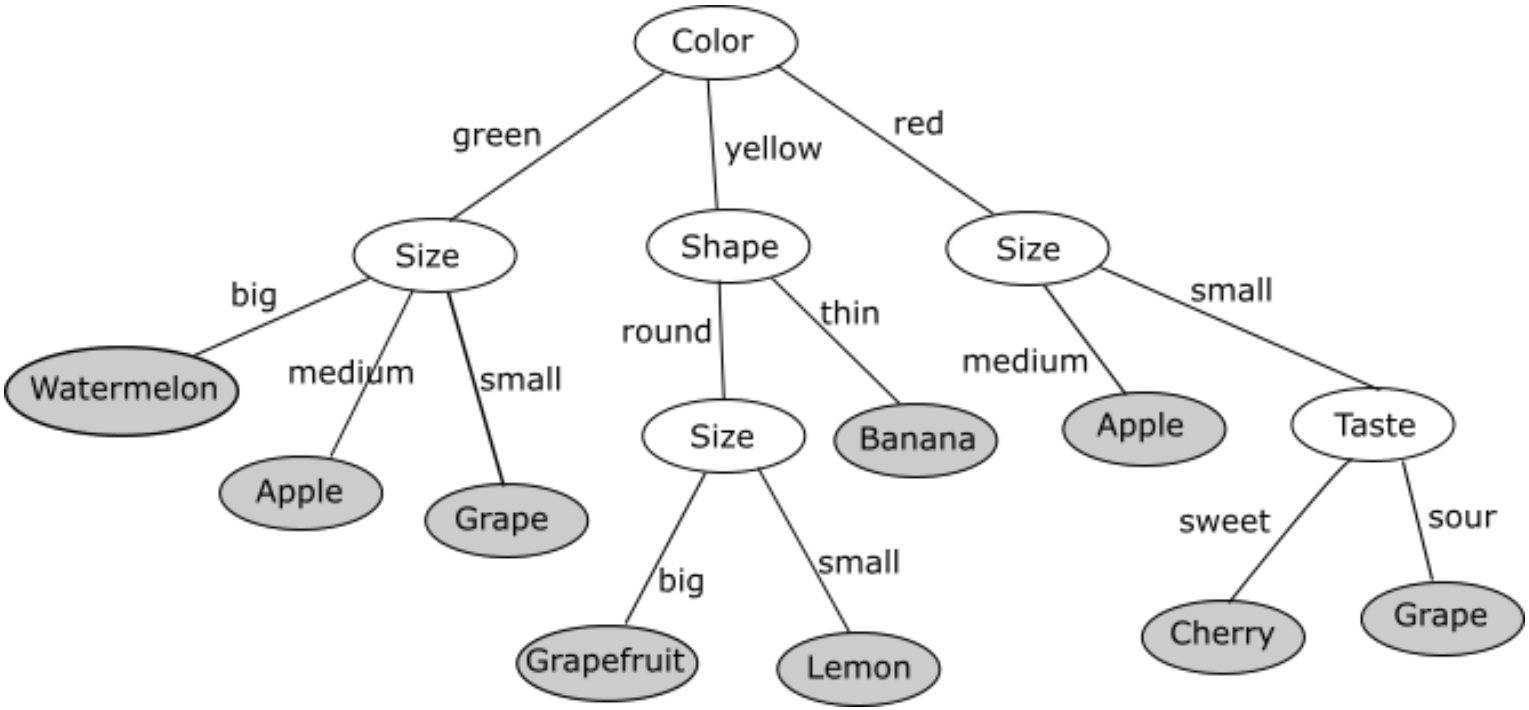
Objectifs

- construire un arbre de décision aussi bien pour la régression que pour la classification
- combiner plusieurs arbres efficacement avec Random Forest

Arbres de décision

Introduction

Modèle de classification ou regréssion qui classe un input dans une de ses feuilles pour rendre sa prédiction :



Avantages

Les arbres de décision

- gèrent les inputs numériques comme catégoriels

Avantages

Les arbres de décision

- gèrent les inputs numériques comme catégoriels
- ne nécessitent pas que la variable d'output soit normalement distribuée (regression linéaire)

Avantages

Les arbres de décision

- gèrent les inputs numériques comme catégoriels
- ne nécessitent pas que la variable d'output soit normalement distribuée (regression linéaire)
- sont interprétables

Avantages

Les arbres de décision

- gèrent les inputs numériques comme catégoriels
- ne nécessitent pas que la variable d'output soit normalement distribuée (regression linéaire)
- sont interprétables
- sont très rapides durant l'inférence

Avantages

Les arbres de décision

- gèrent les inputs numériques comme catégoriels
- ne nécessitent pas que la variable d'output soit normalement distribuée (regression linéaire)
- sont interprétables
- sont très rapides durant l'inférence
- ne nécessitent pas de normalisation des données

Avantages

Les arbres de décision

- gèrent les inputs numériques comme catégoriels
- ne nécessitent pas que la variable d'output soit normalement distribuée (regression linéaire)
- sont interprétables
- sont très rapides durant l'inférence
- ne nécessitent pas de normalisation des données
- leur apprentissage est hautement parallèlisable

Avantages

Les arbres de décision

- gèrent les inputs numériques comme catégoriels
- ne nécessitent pas que la variable d'output soit normalement distribuée (regression linéaire)
- sont interprétables
- sont très rapides durant l'inférence
- ne nécessitent pas de normalisation des données
- leur apprentissage est hautement parallèlisable

$\rightarrow$ Couteau-suisse du machine learning tabulaire.

Désavantages

- peuvent overfit les données, mais l'ensembling résoud ce problème

Désavantages

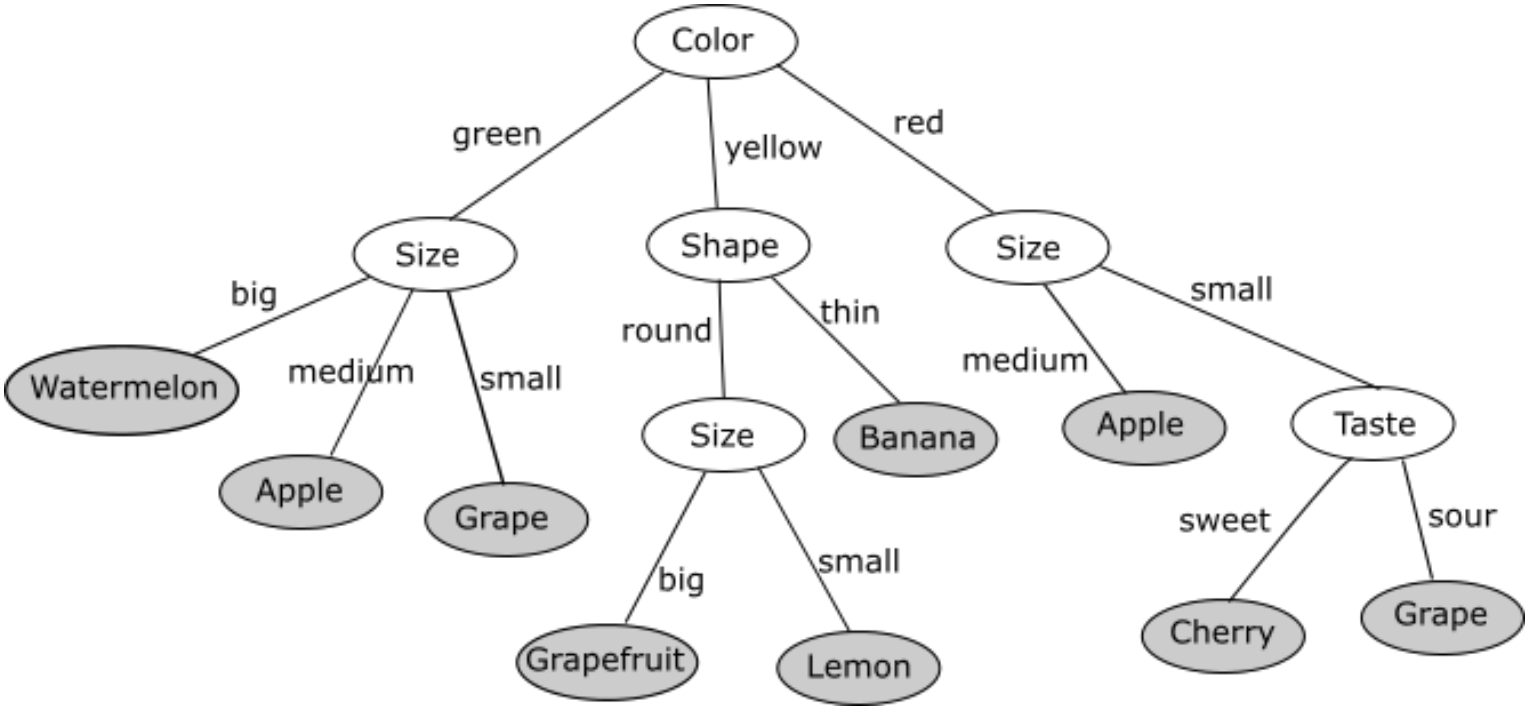
- peuvent overfit les données, mais l'ensembling résoud ce problème
- sont sensibles aux déséquilibres de classe

Désavantages

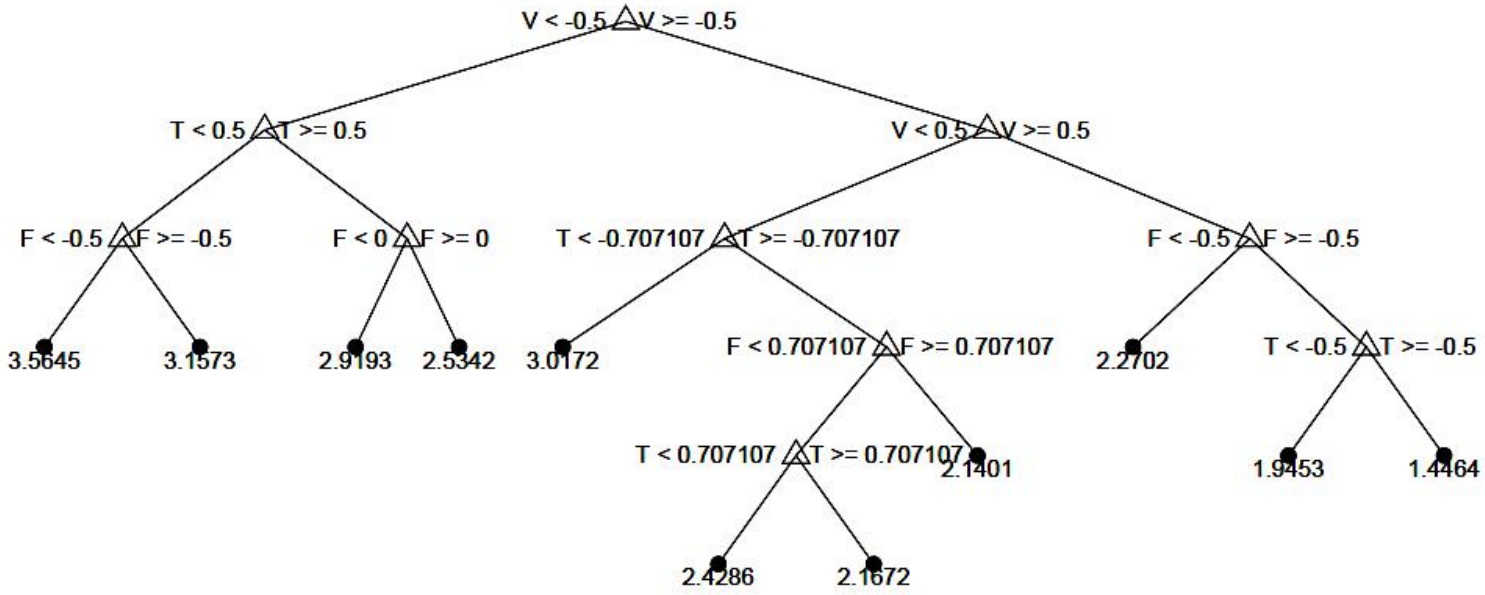
- peuvent overfit les données, mais l'ensembling résoud ce problème
- sont sensibles aux déséquilibres de classe

$\rightarrow$ Si les classes ne sont pas équilibrées, peut-être les resampler.

Arbres de classification



Arbres de régression



Apprendre un arbre de décision

Approche « top-down », procédure récursive :

- créer un nœud de départ qui contient toutes les instances du training set

Apprendre un arbre de décision

Approche « top-down », procédure récursive :

- créer un nœud de départ qui contient toutes les instances du training set
- tant qu'il reste des nœuds non-traités :

Apprendre un arbre de décision

Approche « top-down », procédure récursive :

- créer un nœud de départ qui contient toutes les instances du training set
- tant qu'il reste des nœuds non-traités :
  - choisir un nœud non traité

Apprendre un arbre de décision

Approche « top-down », procédure récursive :

- créer un nœud de départ qui contient toutes les instances du training set
- tant qu'il reste des nœuds non-traités :
  - choisir un nœud non traité
  - si le nœud remplit des conditions de feuille finale, ne rien faire

Apprendre un arbre de décision

Approche « top-down », procédure récursive :

- créer un nœud de départ qui contient toutes les instances du training set
- tant qu'il reste des nœuds non-traités :
  - choisir un nœud non traité
  - si le nœud remplit des conditions de feuille finale, ne rien faire
  - sinon, créer deux branches à partir du nœud non traité pour répartir les instances dans deux nouveaux nœuds

Apprendre un arbre de décision

Approche « top-down », procédure récursive :

- créer un nœud de départ qui contient toutes les instances du training set
- tant qu'il reste des nœuds non-traités :
  - choisir un nœud non traité
  - si le nœud remplit des conditions de feuille finale, ne rien faire
  - sinon, créer deux branches à partir du nœud non traité pour répartir les instances dans deux nouveaux nœuds

Conditions de feuilles finales : contient n_{min} éléments, est déjà à profondeur p_{max}, splitterait sans décroître assez l'entropie...

Décision rendue

En fonction de la tâche, une fois arrivé dans la feuille de fin :

Classification classe majoritaire

Régression moyenne des valeurs cibles

Splits possibles

Splits possibles d'une feature donnée :

Catégorielle chaque catégorie vs le reste

Ordinale/Continue milieu de chaque valeur ou quantiles

Évaluation de la qualité d'un split

En fonction de la tâche :

Régression coût si on rendait la moyenne des instances comme résultat

$$Loss = \sum |\hat{y} - y| \approx variance$$

Classification Entropie de Shannon :

$$Loss = -\sum_{x \in X} P_x * \log_2(P_x)$$

$= 0 \Rightarrow$ il n'y a pas d'incertitude
maximale quand on a une distribution uniforme

Exemple — démarrage

ID, jardinage, jeux vidéos, chapeaux, âge

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 13 \\ 2 & 0 & 1 & 0 & 14 \\ 3 & 0 & 1 & 0 & 15 \\ 4 & 1 & 1 & 1 & 25 \\ 5 & 0 & 1 & 1 & 35 \\ 6 & 1 & 0 & 0 & 49 \\ 7 & 1 & 1 & 1 & 68 \\ 8 & 1 & 0 & 0 & 71 \\ 9 & 1 & 0 & 1 & 73 \end{bmatrix}$$

Première étape : création du nœud de départ

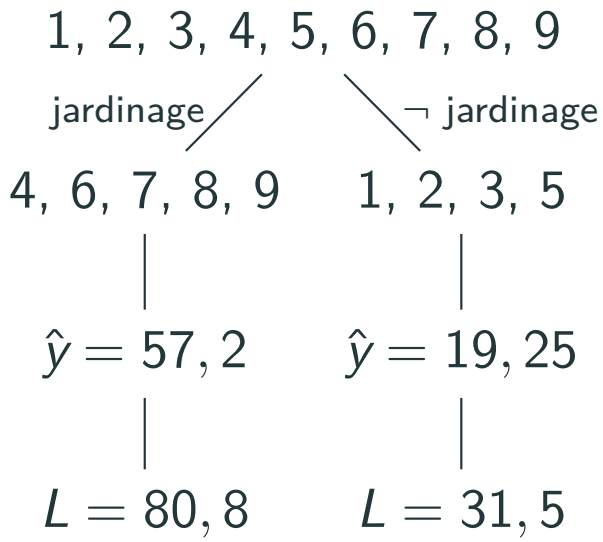
1, 2, 3, 4, 5, 6, 7, 8, 9

Exemple — split

ID, jardinage, jeux vidéos, chapeaux, âge

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 13 \\ 2 & 0 & 1 & 0 & 14 \\ 3 & 0 & 1 & 0 & 15 \\ 4 & 1 & 1 & 1 & 25 \\ 5 & 0 & 1 & 1 & 35 \\ 6 & 1 & 0 & 0 & 49 \\ 7 & 1 & 1 & 1 & 68 \\ 8 & 1 & 0 & 0 & 71 \\ 9 & 1 & 0 & 1 & 73 \end{bmatrix}$$

Split du premier nœud. Il faut tester 3 splits. Split sur jardinage :



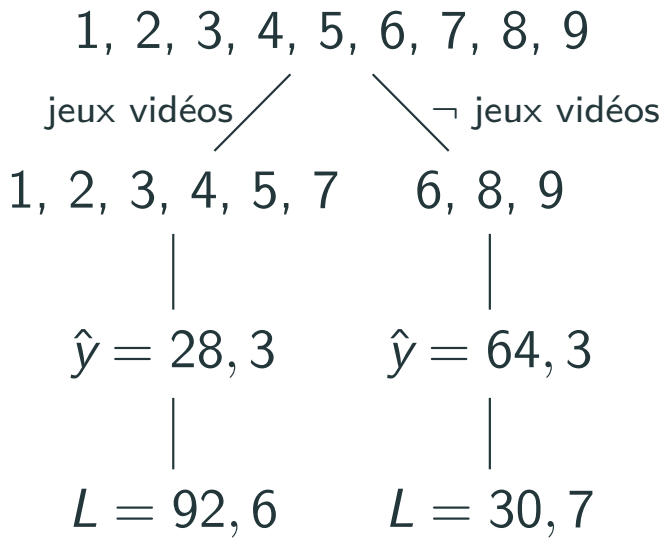
Loss totale : $122,3$

Exemple — split

ID, jardinage, jeux vidéos, chapeaux, âge

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 13 \\ 2 & 0 & 1 & 0 & 14 \\ 3 & 0 & 1 & 0 & 15 \\ 4 & 1 & 1 & 1 & 25 \\ 5 & 0 & 1 & 1 & 35 \\ 6 & 1 & 0 & 0 & 49 \\ 7 & 1 & 1 & 1 & 68 \\ 8 & 1 & 0 & 0 & 71 \\ 9 & 1 & 0 & 1 & 73 \end{bmatrix}$$

Split du premier nœud. Il faut tester 3 splits. Split sur jeux vidéos :



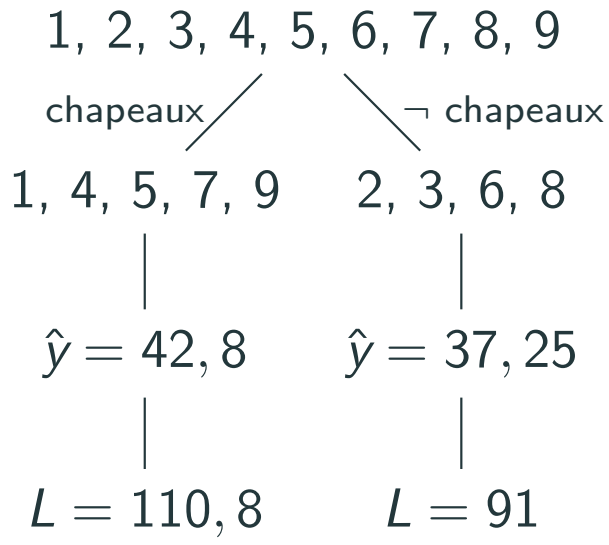
Loss totale : $123, 3$

Exemple — split

ID, jardinage, jeux vidéos, chapeaux, âge

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 13 \\ 2 & 0 & 1 & 0 & 14 \\ 3 & 0 & 1 & 0 & 15 \\ 4 & 1 & 1 & 1 & 25 \\ 5 & 0 & 1 & 1 & 35 \\ 6 & 1 & 0 & 0 & 49 \\ 7 & 1 & 1 & 1 & 68 \\ 8 & 1 & 0 & 0 & 71 \\ 9 & 1 & 0 & 1 & 73 \end{bmatrix}$$

Split du premier nœud. Il faut tester 3 splits. Split sur chapeaux :



Loss totale : $201,8$

Exemple — split

ID, jardinage, jeux vidéos, chapeaux, âge

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 13 \\ 2 & 0 & 1 & 0 & 14 \\ 3 & 0 & 1 & 0 & 15 \\ 4 & 1 & 1 & 1 & 25 \\ 5 & 0 & 1 & 1 & 35 \\ 6 & 1 & 0 & 0 & 49 \\ 7 & 1 & 1 & 1 & 68 \\ 8 & 1 & 0 & 0 & 71 \\ 9 & 1 & 0 & 1 & 73 \end{bmatrix}$$

122,3 jardinage

123,3 jeux vidéos

201,8 chapeaux

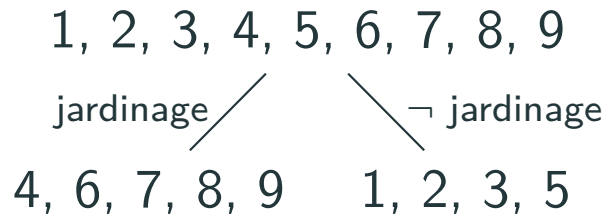
$\rightarrow$ On split donc sur jardinage

Exemple — split

ID, jardinage, jeux vidéos, chapeaux, âge

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 13 \\ 2 & 0 & 1 & 0 & 14 \\ 3 & 0 & 1 & 0 & 15 \\ 4 & 1 & 1 & 1 & 25 \\ 5 & 0 & 1 & 1 & 35 \\ 6 & 1 & 0 & 0 & 49 \\ 7 & 1 & 1 & 1 & 68 \\ 8 & 1 & 0 & 0 & 71 \\ 9 & 1 & 0 & 1 & 73 \end{bmatrix}$$

Résultat après le premier split :



À vous de jouer !

Limiter l'overfit

Fait par :

- la profondeur maximum
- le nombre minimum d'instances dans chaque feuille
- une baisse d'entropie maximale à chaque split
- le nombre minimum d'instances pour split
- le pruning

Random Forest

Introduction

- les arbres de décision overfit facilement
- ils sont rapides à apprendre
- en combiner beaucoup est faisable et réduit la variance

$\rightarrow$ création d'une forêt (ensemble d'arbres) aléatoire

But

Produire des arbres décorrélés et moyenner leurs prédictions pour réduire la variance.

Outil 1 — bagging (row sampling)

Boostrap **agg**regat**ing** (Bagging) :

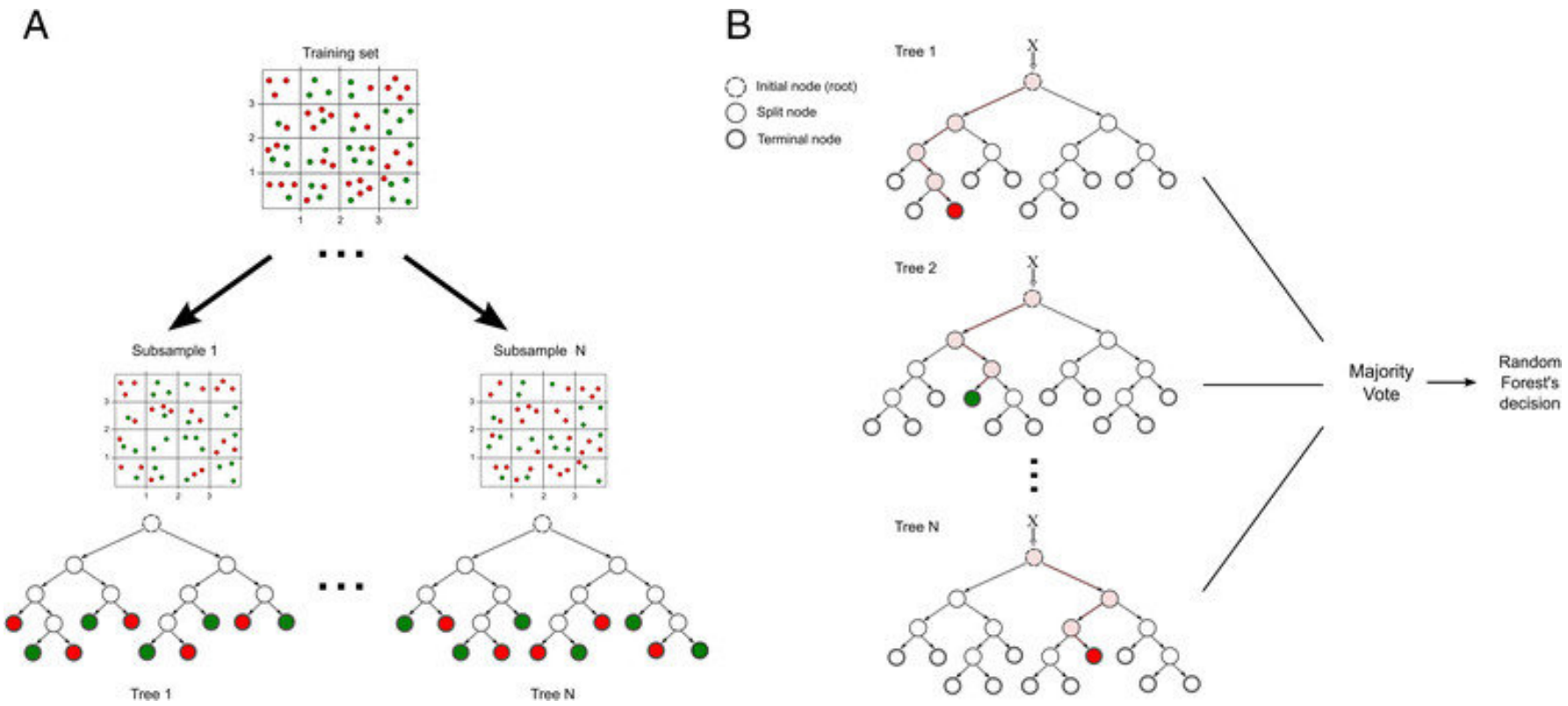
- tirer un échantillon du dataset avec replacement
- entraîner un arbre sur cet échantillon
- répéter B fois

Le bagging s'appelle aussi row sampling.

Outil 2 — random subspace method (column sampling)

- à chaque split, considérer seulement un sous-ensemble des features
- valeurs conseillées :
  - classification : $|\sqrt{m}|$ features par split
  - regréssion : $|\frac{m}{3}|$ features par split, 5 exemples par node minimum

Random Forest



Random Forest

- Pas de sur-apprentissage en augmentant le nombre d'arbres
- Une fois appris, le modèle est très rapide

Conclusion

Conclusion

- les arbres sont interprétables, rapides à entraîner, combinables.
- random forest combine des arbres faibles en un prédicteur versatile