Théorie du Machine Learning

L'Algorithme CART : Fondements Mathématiques et Mécanisme d'Élagage

Yanis Remmache[1]

[1]*Ingénieur Chercheur Data Scientist*

Publié le 11 février 2026

Résumé—L'algorithme CART est un pilier de l'apprentissage supervisé. Ce document détaille la construction de l'arbre maximal par maximisation de la variation d'impureté et le processus d'élagage visant à réduire le risque de surapprentissage via une séquence de sous-arbres emboîtés.

Keywords—*CART, Arbres de décision, Indice de Gini, Entropie, Pruning, Scikit-learn*

1. INTRODUCTION

Créer un arbre de décision consiste à partitionner l'espace des observations $\mathcal{X}$ en M régions appelées *feuilles*. La fonction de prédiction associée est alors de la forme :

$$f(\mathbf{x}) = \sum_{m=1}^{M} c_m \mathbb{1}\{\mathbf{x} \in t_m\} \tag{1}$$

CART est un arbre binaire construit via un partitionnement récursif. L'algorithme procède en deux grandes étapes :

1. **Construction de l'arbre maximal** T_{max}.
2. **Élagage** pour aboutir à un compromis taille/information.

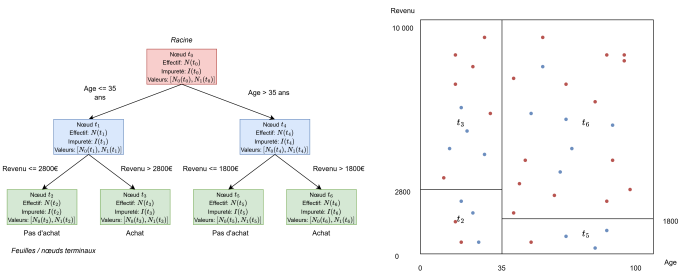

FIGURE 1. Exemple d'arbre CART

La figure 1 présente un arbre CART de profondeur 2, utilisé pour prédire la décision d'achat d'un produit. Les individus ayant au plus 35 ans et gagnant plus de 2800€ par mois, ainsi que ceux qui ont plus de 35 ans et qui gagnent plus de 1800€, sont prédits comme acheteurs par l'algorithme.

2. CONSTRUCTION DE L'ARBRE MAXIMAL

On part d'un noeud initial, la *racine*, que l'on divise en deux noeuds distincts. Ces deux noeuds doivent être plus "purs" que la racine, à l'égard de la variable réponse. Par exemple, si la racine contient l'ensemble des élèves d'une classe et que Y est la note obtenue à un examen, on souhaite que la division sépare correctement ceux qui ont réussi à l'examen et ceux qui ont échoué. Nous formalisons ce principe de deux façons distinctes, selon la nature de la variable cible.

2.1. Le cas de la classification

Soit $\mathcal{Y} = \{1, \dots, K\}$. On définit :

- $p(j|t) = \mathbb{P}(Y = j|\mathbf{X} \in t)$: probabilité conditionnelle au noeud t.
- $N(t)$: nombre d'individus dans t.
- $N_j(t)$: nombre d'individus de classe j dans t.
- $\tilde{T}$: les feuilles / noeuds terminaux de l'arbre T.

Note

Fonction d'impureté ϕ : c'est une fonction $\phi : [0,1]^K \longrightarrow \mathbb{R}_+$ symétrique, maximum au point $(1/K, \dots, 1/K)$ et minimum en cas de pureté parfaite.

L'impureté du noeud t est $I(t) = \phi(p(1|t), \dots, p(K|t))$. Les fonctions usuelles sont :

- **Gini** : $\phi_G = 1 - \sum_k p_k^2 \in [0, 1 - \frac{1}{K}]$. C'est la probabilité de mauvaise étiquette pour deux tirages.
- **Entropie** : $\phi_E = -\sum_k p_k \log_2(p_k) \in [0,1]$. Elle correspond à la quantité d'informations nécessaire pour décrire la classe d'un individu dans le noeud.
- **Erreur de classification** : $\phi_C = 1 - \max p_k$. C'est la part de mauvaises prédictions dans le cadre binaire.

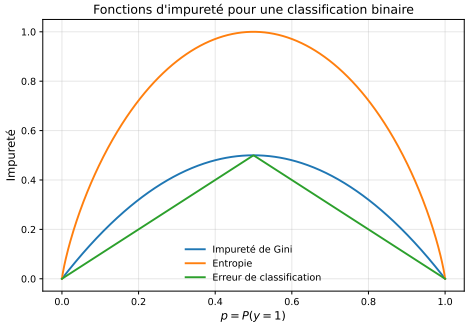

FIGURE 2. Indice de Gini vs Entropie vs Erreur de classification

Note

L'entropie est plus « pessimiste » : pour $p = 0.9$, $\phi_G \approx 0.18$ mais $\phi_E \approx 0.47$. Elle produit souvent des arbres plus profonds.

2.2. Division binaire et optimisation

Une division τ correspond à un seuil $(X \leq a)$ ou un sous-ensemble de modalités. La variation d'impureté engendrée est :

$$\Delta I(t, \tau) = I(t) - \left[\frac{N(t_g)}{N(t)} I(t_g) + \frac{N(t_d)}{N(t)} I(t_d) \right] \tag{2}$$

On cherche $\tau^* = \text{argmax}_{\tau \in \mathcal{S}_t} \Delta I(t, \tau)$. Pour trouver τ^*, on parcourt tous les régresseurs X_j disponibles. Pour chaque variable X_j, on calcule $\Delta I(t, \tau)$ pour chaque partition du support de X_j admissible, dans le noeud t. On sélectionne finalement celle qui maximise la variation d'impureté. Pour une variable continue, on teste les seuils situés aux milieux des valeurs distinctes ordonnées. La prédiction finale, définie dans chaque feuille, est obtenue par un vote à la majorité :

$$\forall t \in \tilde{T}, \hat{Y}_t = \text{argmax}_{j \in \{1,\dots,K\}} \hat{p}(j|t)$$

2.3. Le cas de la régression

Pour $\mathcal{Y} = \mathbb{R}$, la variance du noeud t est définie comme :

$$S^2(t) = \frac{1}{N(t)} \sum_{i:\mathbf{x}_i \in t} (Y_i - \frac{1}{N(t)} \sum_{i:\mathbf{x}_i \in t} Y_i)^2 \quad (3)$$

Le critère à maximiser devient :

$$\Delta S^2(t,\tau) = S^2(t) - \left(S^2(t_g) + S^2(t_d)\right) \quad (4)$$

On cherche alors la division qui provoque la plus forte baisse de variance au sein de chaque noeud enfant. La prédiction finale correspond à la moyenne de la variable cible dans chaque feuille :

$$\forall t \in \tilde{T}, \hat{Y}_t = \frac{1}{N(t)} \sum_{i:\mathbf{x}_i \in \tilde{T}} Y_i$$

2.4. Critères d'arrêt et pré-élagage

Les divisions binaires sont effectuées de façon récursive, tant que le critère d'arrêt spécifié n'est pas atteint. Les critères d'arrêt par défaut sont les suivants :

— Le noeud considéré est totalement pur.
— Le noeud ne contient plus qu'une observation.

Il est aussi possible d'utiliser d'autres critères d'arrêts visant à limiter la complexité de l'arbre. On parle de pré-élagage :

— L'arbre a atteint une profondeur maximale définie a priori (attribut `max_depth` de la classe `DecisionTreeClassifier` de scikit-learn).
— L'arbre a atteint un nombre de noeuds terminaux défini a priori (attribut `max_leaf_nodes`).
— Le noeud contient un nombre d'individus inférieur à celui spécifié pour la division (attribut `min_samples_split`).
— Les noeuds enfants résultant de la division contiennent moins d'individus que le minimum requis (attribut `min_samples_leaf`).
— La division provoque une variation de l'impureté inférieure au minimum requis (attribut `min_impurity_decrease`).

3. ÉLAGAGE (PRUNING)

Lors de la phase de croissance, un arbre CART continue de se diviser jusqu'à ce que chaque feuille soit "pure" (ne contienne qu'une seule classe) ou atteigne une limite minimale d'observations. Par conséquent, l'arbre capture non seulement la structure des données, mais aussi le bruit statistique. L'élagage intervient alors pour obtenir un modèle généralisable sur de nouvelles données. L'idée est de construire une séquence de sous-arbres emboîtés puis de choisir l'arbre avec le meilleur pouvoir prédictif en validation croisée, parmi ceux de la séquence. On définit la fonction de coût-complexité :

$$R_\alpha(T) = R(T) + \alpha|\tilde{T}| \quad (5)$$

$R(T)$ est la mesure d'erreur considérée (taux de mauvaises classifications ou MSE en régression), $|\tilde{T}|$ est le nombre de feuilles, α est un paramètre de complexité strictement positif. Ainsi, $R_\alpha(T)$ pénalise la complexité de l'arbre. En général, pour un noeud t donnée, $R(T_t) < R(t)$. Cependant, on peut avoir $R_\alpha(T_t) = R_\alpha(t)$, selon la valeur de α.

3.1. Création de la séquence de sous-arbres emboîtés

Soient $T_{\max}$, l'arbre complet, et $t \in T_{\max} \backslash \tilde{T}_{\max}$, un noeud intermédiaire. On compare deux scénarios : garder le sous-arbre issu de t, T_t, ou l'élaguer. Les deux scénarios sont équivalents en termes de coût-complexité si et seulement si $R_\alpha(T_t) = R_\alpha(t) \iff \alpha = \frac{R(t)-R(T_t)}{|\tilde{T}_t|-1}$. On définit alors la valeur critique $\alpha_t = \frac{R(t)-R(T_t)}{|\tilde{T}_t|-1}$. α_t représente la résistance du sous-arbre à l'élagage : plus α_t est grand, plus le sous-arbre est efficace pour réduire l'erreur par rapport à sa complexité. À chaque étape, on cherche le noeud t^* qui minimise cette valeur. On détermine ensuite le plus petit α_t parmi tous les noeuds testés :

$$\alpha^* = \min_{t \in T_{\max} \backslash \tilde{T}_{\max}} \alpha_t \quad (6)$$

On supprime la branche associée à α^* de sorte à créer le sous-arbre $T_1 = T_{\max} \backslash T_{t^*}$. Si α^* est plus grand que l'hyperparamètre $\alpha^{(ccp)}$ alors, on réitère la procédure sur le nouvel arbre élagué T_1, jusqu'à ce qu'il ne reste plus que la racine. Sinon, on stoppe l'élagage. On obtient ainsi une séquence d'arbres emboîtés $\{t_0\} \subset \cdots \subset T_2 \subset T_1 \subset T_{\max}$.

3.2. Choix de l'arbre optimal

Finalement, on choisit l'arbre associé à la plus petite erreur, en validation croisée, parmi ceux de la sous-séquence créée.

4. CONCLUSIONS ET PERSPECTIVES

L'algorithme CART demeure un standard en raison de sa grande interprétabilité et de sa capacité à capturer des relations non-linéaires sans hypothèse forte sur la distribution des données. Sa structure binaire permet une lecture directe des règles de décision, facilitant la communication des résultats aux métiers.

Cependant, les arbres simples souffrent d'une forte instabilité : une légère modification du jeu de données peut radicalement changer la structure de l'arbre. Pour pallier cette limite, les perspectives modernes s'orientent vers :

— **Le Bagging et les Forêts Aléatoires** : pour réduire la variance par agrégation d'arbres indépendants.
— **Le Boosting (XGBoost, LightGBM)** : pour réduire le biais par construction séquentielle.

CONTACT

linkedin.com/in/yanis-remmache-60613a151
github.com/yanisrem
yanisrem.github.io