Construction d'arbres de décision optimaux

Valentine Huré, Zacharie Alès, Amélie Lambert

Construction d'arbres de décision optimaux

Zacharie Ales[1], Valentine Huré[2], Amélie Lambert[2]

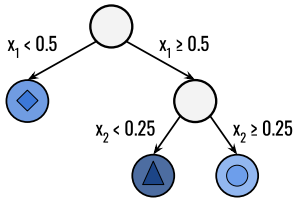
[1] UMA, ENSTA Paris, Institut Polytechnique de Paris, Palaiseau, France.
zacharie.ales@ensta-paris.fr
[2] CNAM, Paris, France
valentine.hure@lecnam.net,amelie.lambert@cnam.fr

Mots-clés : *classification supervisée, arbre de décision, PLNE, QP*

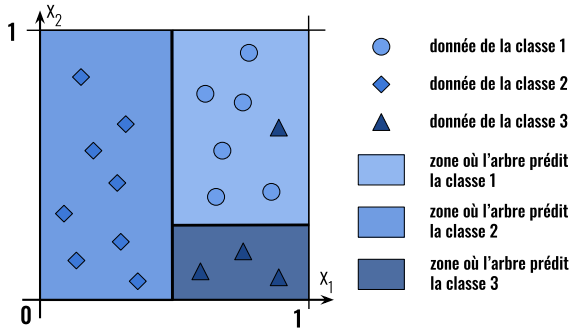
1 Les arbres de décision

On considère un jeu de données $(X_i, y_i)_{i\in\mathcal{I}}$ où $X_i \in \mathbb{R}^{|\mathcal{J}|}$ est le vecteur d'attributs de la donnée i et $y_i \in \mathcal{K}$ est sa classe. La classification supervisée consiste à construire une fonction de classification $f : \mathbb{R}^{|\mathcal{J}|} \mapsto \mathcal{K}$ (ou classifieur) à partir d'un jeu données.

Les arbres de décision constituent une famille de classifieurs qui attribuent une classe aux données en fonction de leur parcours dans l'arbre. Nous présentons en Figure (1) un exemple d'arbre de décision pour un jeu de données où $|\mathcal{J}| = 2$ et $|\mathcal{K}| = 3$. Les nœuds de branchement, en blanc dans la Figure (1a), appliquent des règles de la forme $aX < b$ qui orientent les données vers leurs deux nœuds fils. Les règles sont dites univariées quand a est unitaire et multivariées sinon. Les feuilles, représentées en bleu dans la Figure (1a), attribuent une classe aux données qui les atteignent. Une erreur de classification correspond au fait qu'une donnée arrive dans une feuille de classe différente de la sienne (e.g. un triangle dans la Figure (1b)).



(a) Arbre de décision



(b) Partition de l'espace en fonction de la classe prédite

FIG. 1 – Exemple d'arbre de décision

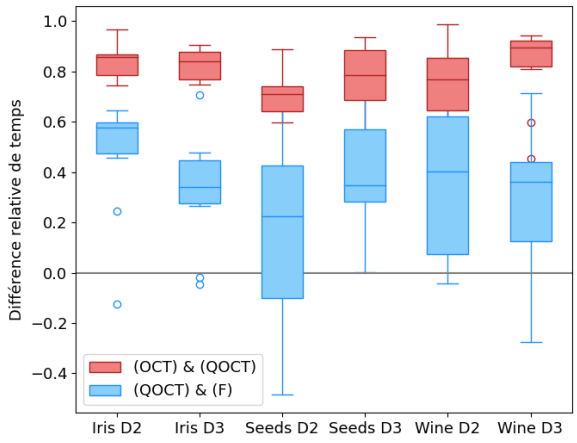
Déterminer un arbre de décision minimisant le nombre d'erreurs de classification est un problème NP-complet [6]. L'heuristique CART [4] est l'algorithme de la littérature le plus utilisé, mais son approche gloutonne fournit généralement des solutions sous-optimales. Récemment, plusieurs modèles ont été introduits pour résoudre à l'optimalité ce problème notamment via la PLNE [2, 3] ou la programmation dynamique [5]. Certains modèles se cantonnent à la construction d'arbres de décision pour des données binaires (*i.e.* $X_i \in \{0,1\}^{|\mathcal{J}|}$) [2, 5]. Le modèle ($OCT$) [3] permet de créer des arbres de décisions à partir de données continues (*i.e.* $X_i \in \mathbb{R}^{|\mathcal{J}|}$) pour des règles de branchement univariées et multivariées. Nos travaux portent sur une quadratisation de (OCT) et sur une extension du modèle de [2] aux données continues.

2 Nouvelles modélisations

Quadratisation de la fonction objectif d'OCT [3]
Le modèle (OCT) utilise trois familles de variables pour dénombrer les erreurs de classification. Nous proposons une reformulation $(QOCT)$ de la fonction objectif qui permet de supprimer ces variables. De plus, nous constatons expérimentalement que cette quadratisation améliore la valeur de la relaxation linéaire et réduit significativement le temps de résolution.

Généralisation du modèle de Aghaei et al. [2] aux données continues
Ce modèle associe à chaque donnée correctement classée un flot unitaire représentant son parcours dans l'arbre. Nous introduisons une extension (F) de ce modèle permettant de classifier des données continues. Dans le cas multivarié, nous démontrons que notre formulation permet des séparations non autorisées par (OCT). Enfin, nous observons expérimentalement que notre formulation est significativement plus rapide que (OCT) et même que $(QOCT)$.

Quelques résultats expérimentaux
Nous avons comparé les modèles sur plusieurs jeux de données et nous présentons des résultats pour trois d'entre eux : Iris, Seed et Wine [1]. Chaque modèle a été utilisé pour résoudre 10 instances issues de chaque jeu de données pour des arbres de profondeur 2 et 3. Sur la Figure (2) sont représentées par des diagrammes en boîte les différences relatives de temps de résolution. Nous observons que $(QOCT)$ est significativement plus rapide que (OCT), avec une différence relative entre 70% et 90%, et que (F) est plus rapide que $(QOCT)$ mais la différence est moins prononcée. Pour certaines instances, la différence relative est négative (*i.e.* (F) est plus lent que $(QOCT)$).



En notant $T_{(P)}$ le temps de résolution pour le modèle (P), la différence relative est :
- en rouge : $\frac{T_{(OCT)}-T_{(QOCT)}}{T_{(OCT)}}$;
- en bleu : $\frac{T_{(QOCT)}-T_{(F)}}{T_{(QOCT)}}$.

FIG. 2 – Différences relatives des temps de résolution

Références

[1] UCI Machine Learning Repository. http://archive.ics.uci.edu/ml.

[2] S. Aghaei, A. Gomez, and P. Vayanos. Learning Optimal Classification Trees : Strong Max-Flow Formulations. *arXiv*, Feb 2020.

[3] D. Bertsimas and J. Dunn. Optimal classification trees. *Mach. Learn.*, 106(7) :1039–1082, Jul 2017.

[4] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. Classification and regression trees. *Cole Statistics/Probability Series*, 1984.

[5] E. Demirović, A. Lukina, E. Hebrard, J. Chan, J. Bailey, C. Leckie, K. Ramamohanarao, and P.J. Stuckey. MurTree : Optimal Classification Trees via Dynamic Programming and Search. *arXiv*, Jul 2020.

[6] L. Hyafil and R.L. Rivest. Constructing optimal binary decision trees is NP-complete. *Inf. Process. Lett.*, 5(1) :15–17, May 1976.