Approximation des explications probabilistes via la minimisation super-modulaire

Louenas Bounia*, Frédéric Koriche*

Univ. Artois, CNRS, CRIL, F-62300 Lens*
nom@cril.fr,
http://www.cril.univ-artois.fr/

Résumé. Calculer une explication abductive d'une instance vise à expliquer la prédiction faite. Cependant, vu nos limitations cognitives, les explications abductives sont parfois de trop grande taille pour être interprétables. Lorsque cela se produit, nous devons réduire la taille des explications tout en déterminant toujours la classe prédite avec une probabilité élevée. Dans ce travail, nous montrons que calculer de telles « explications probabilistes » est NP-difficile, même pour la classe restreinte des arbres de décision. Afin de contourner le problème, nous étudions dans ce travail l'approximation des explications probabilistes sous l'angle de la super-modularité. Nous examinons à la fois les approches de descente gloutonne (GD) et d'ascension gloutonne (GA) pour la minimisation super-modulaire, dont les garanties d'approximation dépendent de la courbure de la fonction d'erreur « non normalisée » qui évalue la précision de l'explication. Basés sur diverses expériences visant à expliquer les prédictions des arbres de décision, nous montrons que nos algorithmes gloutons offrent une alternative efficace à la méthode exacte basé sur un encodage SAT.

Cet article est un résumé du papier (en anglais) (Bounia et Koriche, 2023).

1 Introduction

Une explication abductive de grande taille est difficilement comprise par les utilisateurs humains. En effet, en psychologie cognitive, il existe une limite supérieure à notre capacité de raisonnement sur des éléments interagissant simultanément. Comme l'a conjecturé (Miller, 1956), cette limite est de $sept$ éléments plus ou moins $deux$, et depuis lors, elle a été confirmée par de nombreuses expériences en sciences cognitives. Ainsi, restreindre la taille des explications apparaît comme nécessaire pour garantir leur intelligibilité. Sur la base de ces considérations, comment pouvons-nous réduire la taille des explications tout en préservant une grande partie leur validité ? C'est là que les *explications probabilistes* (Wäldchen et al., 2021) entrent en jeu. Supposons que nous disposions d'une explication abductive I ainsi que d'une limite de taille $k \leq |I|$. Pour tout sous-ensemble candidat S de I, soit $\epsilon_{h,\boldsymbol{x}}(S)$ la probabilité de commettre une « erreur d'explication » pour déduire $h(\boldsymbol{x})$ en utilisant $\boldsymbol{x}_S$ au lieu de $\boldsymbol{x}$. Sur cette base, le principal problème examiné dans la suite de ce travail est de trouver une explication probabiliste $S \subseteq I$ de taille au plus k de telle sorte que $\epsilon_{h,\boldsymbol{x}}(S)$ soit minimisé. Malheureusement, ce problème est $\mathsf{NP}^{\mathsf{PP}}$-difficile pour les classifieurs d'une manière générale (Wäldchen

et al., 2021), et NP-difficile pour les arbres de décision (Arenas et al., 2022). Comme le montre la présente étude, ce problème reste NP-difficile pour les arbres de décision même dans le cas restreint où S est un sous-ensemble d'une raison suffisante. Afin de surmonter une telle barrière computationnelle, nous étudions *l'approximation* des explications probabilistes à travers le prisme de la super-modularité.

2 Explication probabiliste

Les classifieurs considérés dans cette étude sont des fonctionnes booléennes de la forme $h : \{0,1\}^d \to \{0,1\}$. Une instance partielle est un vecteur $\boldsymbol{z} \in \{0,1,*\}^d$, où $z_i = *$ indique que la i-ème attribut de $\boldsymbol{z}$ est non définie. Une instance $\boldsymbol{x}$ est couverte par $\boldsymbol{z}$ si $x_i = z_i$ pour tous les attributs $i \in [d] = \{1, \cdots, d\}$ telles que $z_i \neq *$. La restriction de $\boldsymbol{x}$ à un sous-ensemble $S \subseteq [d]$, notée $\boldsymbol{x}_S$, est l'instance partielle dans $\{0,1,*\}^d$ telle que, pour chaque $i \in [d]$, $(x_S)_i = x_i$ si $i \in S$, et $(x_S)_i = *$ sinon. Clairement, une instance $\boldsymbol{y} \in \{0,1\}^d$ est couverte par $\boldsymbol{x}_S$ si $\boldsymbol{y}_S = \boldsymbol{x}_S$. Nous rappelons également qu'un *littéral* sur X_d est une variable x_i, ou sa négation $\overline{x}_i$. La négation d'un littéral l est donné par $\neg l = x_i$ si $l = \overline{x}_i$, et $\neg l = \overline{x}_i$ si $l = x_i$. Un *terme* est une conjonction de littéraux, et une DNF est une disjonction de termes. Ici, les formules DNF sont considérées comme des ensembles de termes, et les termes comme des ensembles de littéraux. Étant donné un classifieur h, et une instance $\boldsymbol{x}$ pour laquelle la prédiction $h(\boldsymbol{x})$ doit être expliquée, soit $\epsilon_{h,\boldsymbol{x}} : 2^{[d]} \to \mathbb{R}$ la *fonction d'erreur* est définie par

$$\epsilon_{h,\boldsymbol{x}}(S) = \frac{|\{\boldsymbol{y} \in \{0,1\}^d : h(\boldsymbol{y}) \neq h(\boldsymbol{x}), \boldsymbol{y}_S = \boldsymbol{x}_S\}|}{|\{\boldsymbol{y} \in \{0,1\}^d : \boldsymbol{y}_S = \boldsymbol{x}_S\}|} \tag{1}$$

Comme indiqué ci-dessus, $\epsilon_{h,\boldsymbol{x}}(S)$ peut être interprété comme la probabilité de commettre une « erreur d'explication » en utilisant l'instance partielle $\boldsymbol{x}_S$ plutôt que l'instance complète $\boldsymbol{x}$. Étant donné un paramètre de précision $\varepsilon \in [0,1]$, une explication S est appelée $(1-\varepsilon)$-probable si $\epsilon_{h,\boldsymbol{x}}(S) \leq \varepsilon$. S est une raison suffisante pour $\boldsymbol{x}$ étant donné h si $\epsilon_{h,\boldsymbol{x}}(S) = 0$, et $\epsilon_{h,\boldsymbol{x}}(S') > 0$ pour tout sous-ensemble propre S' de S. Notons que (1) peut être réécrit comme

$$\epsilon_{h,\boldsymbol{x}}(S) = \frac{\mu_{h,\boldsymbol{x}}(S)}{2^{d-|S|}} \tag{2}$$

où $\mu_{h,\boldsymbol{x}}(S)$ est le nombre d'erreurs induites par le choix de S, c'est-à-dire,

$$\mu_{h,\boldsymbol{x}}(S) = |\{\boldsymbol{y} \in \{0,1\}^d : h(\boldsymbol{y}) \neq h(\boldsymbol{x}), \boldsymbol{y}_S = \boldsymbol{x}_S\}| \tag{3}$$

Exemple 1. *Considérons le classifieur $h : \{0,1\}^3 \to \{0,1\}$ spécifié par la fonction polynomiale à seuil suivante :*

$$h(\boldsymbol{x}) = 1 \Leftrightarrow x_1x_2x_3 + x_1x_2 - x_1 - x_2 \geq 0 \tag{4}$$

Pour l'instance $\boldsymbol{x} = (1,1,1)$ pour laquelle nous devons expliquer $h(\boldsymbol{x}) = 1$, et en utilisant le diagramme de Hasse dans la figure 1, on observe que le terme associé à l'ensemble $\{1,2,3\}$, à savoir $t_{\{1,2,3\}} = x_1 \wedge x_2 \wedge x_3$ est la seule raison suffisante pour $\boldsymbol{x}$ étant donné h. Cependant, les termes associés aux deux ensembles $\{1,2\}$ et $\{3\}$ sont tous les deux des raisons $\frac{1}{2}$-probables minimales pour l'inclusion ensembliste pour $\boldsymbol{x}$ étant donné h.

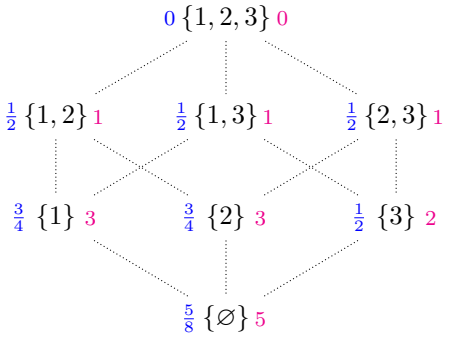

FIG. 1 – L'erreur $\epsilon_{h,\boldsymbol{x}}(S)$ (en bleu) et le nombre d'erreurs $\mu_{h,\boldsymbol{x}}(S)$ (en magenta) pour chaque $S \subseteq [3]$, en utilisant le classifieur h donné par (4) et l'instance $\boldsymbol{x} = (1, 1, 1)$.

3 Minimisation super-modulaire

3.1 Formulation du problème

Le principal problème examiné dans ce travail est le suivant :

Problème 1. *Étant donné un classifieur $h : \{0,1\}^d \to \{0,1\}$, une instance $\boldsymbol{x} \in \{0,1\}^d$, un ensemble $I \subseteq [d]$, une limite de taille $k \leq |I|$, trouver un sous-ensemble $S \subseteq I$ de taille au plus k tel que $\epsilon_{h,\boldsymbol{x}}(S)$ soit minimisé.*

Répondre au problème 1 nécessite l'évaluation de la fonction d'erreur. La fonction d'erreur, notée $\epsilon_{h,\boldsymbol{x}}(S)$ mesure l'erreur de classification du classifieur h pour l'instance $\boldsymbol{x}$ en utilisant le sous-ensemble S. Pour résoudre le problème 1, nous devons essayer différentes combinaisons de sous-ensembles d'attributs, évaluer la fonction d'erreur pour chaque combinaison, et choisir une de celles qui donnent la valeur minimale de la fonction d'erreur.

3.2 Évaluation des erreurs d'explication

Il est facile de voir que le problème d'évaluer $\epsilon_{h,\boldsymbol{x}}(S)$ pour un classifieur arbitraire est en général #P**-difficile** (Wäldchen et al., 2021). Cependant, (Izza et al., 2022) ont montré que $\epsilon_{h,\boldsymbol{x}}(S)$ peut être calculé en temps polynomial lorsque h est un arbre de décision.

Proposition 1. *Soit un classifieur $h : \{0,1\}^d \to \{0,1\}$ représenté par un arbre de décision $\mathcal{T}$, une instance $\boldsymbol{x} \in \{0,1\}^d$, et un sous-ensemble d'attributs $S \subseteq [d]$, évaluer $\epsilon_{h,\boldsymbol{x}}(S)$ peut être réalisé en temps $\mathcal{O}(|S| \cdot |\mathcal{T}|)$.*

Minimisation de l'erreur d'explication. Nous présentons d'abord un résultat qui montre que le problème 1 est NP-difficile lorsque le classifieur h est un arbre de décision. Ce résultat confirme la complexité intrinsèque du problème tel que défini dans la section précédente. Nous avons montré (Bounia et Koriche, 2023) que contrairement à la fonction d'erreur $\epsilon_{h,\boldsymbol{x}}(\cdot)$, qui n'est ni monotone ni super-modulaire, la fonction d'erreur non normalisée $\mu_{h,\boldsymbol{x}}(\cdot)$ présente des propriétés favorables. Elle est super-modulaire, monotone décroissante (Bounia et Koriche, 2023), et toujours positive. Ces propriétés sont cruciales pour l'approximation du problème 1.

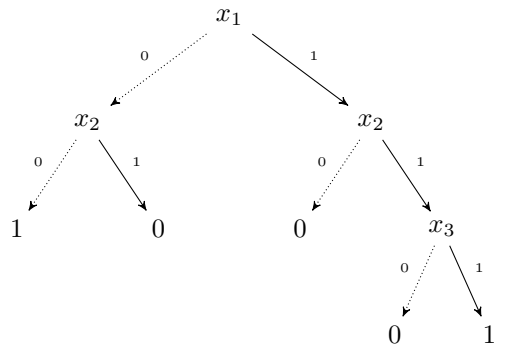

FIG. 2 – Une représentation par arbre de décision de (4).

Proposition 2. *Soit un classifieur h représenté par un arbre de décision $\mathcal{T}$, une instance $\boldsymbol{x} \in \{0,1\}^d$, et une raisons suffisante $I \subseteq [d]$ pour $\boldsymbol{x}$ étant donné h, un entier $k \leq |I|$, et un seuil $\varepsilon \in (0,1)$. Le problème de déterminer un sous-ensemble $S \subseteq I$ d'une taille au plus k qui satisfait $\epsilon_{h,\boldsymbol{x}}(S) \leq \varepsilon$ est* NP*-difficile.*

3.3 Fonctions super-modulaires

Pour une fonction d'ensemble $f : 2^{[d]} \to \mathbb{R}$, $\mathrm{L}_f(i \mid S) = f(S \setminus \{i\}) - f(S)$ capture la *perte marginale* de retirer un élément i d'un ensemble S et $\mathrm{G}_f(i \mid S) = f(S \cup \{i\}) - f(S)$ capture le *gain marginal* d'ajouter un élément i à un ensemble S. Pour tous les $S \subseteq [d]$ et $i \in [d] \setminus S$, f est *décroissante* si $\mathrm{L}_f(i \mid S) \geq 0$ et *croissante* si $\mathrm{G}_f(i \mid S) \geq 0$. f est *super-modulaire* si $\mathrm{L}_f(i \mid S) \geq \mathrm{L}_f(i \mid T)$ pour tous les $S \subseteq T \subseteq [d]$ et $i \in S$, et de manière duale, f est *sous-modulaire* si $\mathrm{G}_f(i \mid S) \geq \mathrm{G}_f(i \mid T)$. f est super-modulaire si et seulement si $-f$ est sous-modulaire et dite *modulaire* si elle est à la fois sous-modulaire et super-modulaire. Et pour un sous-ensemble non vide $I \subseteq [d]$, la *courbure* de f sur 2^I est donnée par :

$$c = 1 - \min_{i \in I} \frac{\mathrm{L}_f(i \mid I)}{\mathrm{L}_f(i \mid \{i\})} = 1 - \min_{i \in I} \frac{\mathrm{G}_f(i \mid I \setminus \{i\})}{\mathrm{G}_f(i \mid \varnothing)} \tag{5}$$

Il est clair que $c \in [0,1]$ chaque fois que f est croissante et sous-modulaire, ou décroissante et super-modulaire. Lorsque $I = [d]$, c est appelée la *courbure totale* de f (Conforti et Cornuéjols, 1984). Dans le cas où f est décroissante et super-modulaire, rappelons que la condition $c < 1$ est suffisante pour garantir que la tâche de minimiser f sous contrainte de cardinalité est approximable à une constante près (Boutsidis et al., 2015).

À la lumière de la figure 1, nous pouvons constater que la fonction d'erreur $\epsilon_{h,\boldsymbol{x}}(\cdot)$ n'est généralement *pas* super-modulaire ou sous-modulaire, et *pas* monotone décroissante ou monotone croissante. Cependant, si nous nous concentrons plutôt sur la version non-normalisée $\mu_{h,\boldsymbol{x}}(\cdot)$ donnée dans (3), alors les propriétés suivantes peuvent être déduites.

Proposition 3. *Soit $h : \{0,1\}^d \to \{0,1\}$ un classifieur, une instance $\boldsymbol{x} \in \{0,1\}^d$, et $I \subseteq [d]$ un ensemble non vide. Donc, $\mu_{h,\boldsymbol{x}}(\cdot)$ est super-modulaire et monotone décroissante. De plus, si I est une raison suffisante pour $\boldsymbol{x}$ étant donné h, alors la courbure c de $\mu_{h,\boldsymbol{x}}(\cdot)$ sur 2^I satisfait $c < 1$.*

4 Algorithmes approchées

Après avoir fourni une vue d'ensemble des explications probabilistes et de la minimisation super-modulaire, nous présentons deux algorithmes pour l'approximation du problème 1.

4.1 Greedy descent (GD)

Une approche naturelle pour minimiser une fonction f super-modulaire et décroissante sous contrainte de cardinalité $|S| \leq k$ est de partir de l'ensemble d'entrée I des attributs candidates, et de retirer itérativement de la solution courante S tout attribut i qui minimise la perte marginale $\mathrm{L}_f(i \mid S)$, jusqu'à ce que la taille désirée $|S| = k$ soit atteinte. Comme l'a montré (Il'ev, 2001), cette méthode gloutonne atteint une borne d'approximation de $\frac{e^p-1}{p}$, où $p = \frac{c}{1-c}$, et c est la courbure de f sur 2^I. Dans le cadre de notre étude, la fonction d'erreur $\epsilon_{h,\boldsymbol{x}}(\cdot)$ définie dans (2) est une version normalisé de $\mu_{h,\boldsymbol{x}}(\cdot)$, qui est super-modulaire et décroissante. De plus, le facteur de normalisation $2^{d-|S|}$ est *constant* pour tous les sous-ensembles S de même taille. En se basant sur ces propriétés, nous pouvons combiner l'approche gloutonne décrite ci-dessus pour $f = \mu_{h,\boldsymbol{x}}(\cdot)$ avec une méthode de sélection par niveau qui stocke les ensembles $S_0, S_1, \cdots, S_k$ obtenus pour chaque niveau $j \in \{0, 1, \cdots, k\}$, et qui retourne de cette séquence le meilleur sous-ensemble S_j par rapport à $\epsilon_{h,\boldsymbol{x}}(\cdot)$.

Algorithm 1 Greedy descent (GD)

Input: un classifieur h, une instance $\boldsymbol{x}$, un ensemble d'attributs I, un entier k
$S_n \leftarrow I$, $n \leftarrow |I|$
for $j = n$ ***downto*** 1 **do**
  $i^* \leftarrow \mathrm{argmin}_{i \in S_j} \mu_{h,\boldsymbol{x}}(S_j \setminus \{i\})$
  $S_{j-1} = S_j \setminus \{i^*\}$
$S_{\text{GD}} \leftarrow \mathrm{argmin}_{S \in \{S_0, S_1, \cdots, S_k\}} \epsilon_{h,\boldsymbol{x}}(S)$
return S_{GD}

Proposition 4. *Soit S^* une solution optimale du problème 1, soit c la courbure de $\mu_{h,\boldsymbol{x}}(\cdot)$ sur 2^I, et supposons que I soit une raison suffisante pour $\boldsymbol{x}$ étant donné h. La solution S_{GD} retournée par l'algorithme Greedy Descent (*GD*) satisfait :*

$$\epsilon_{h,\boldsymbol{x}}(S_{\text{GD}}) \leq \left(\frac{e^p - 1}{p}\right) \epsilon_{h,\boldsymbol{x}}(S^*) \text{ où } p = \frac{c}{1-c} < \infty$$

4.2 Greedy ascent (GA)

Une approche alternative consiste à considérer la fonction objective $f = -\mu_{h,\boldsymbol{x}}(\cdot)$, qui est sous-modulaire et croissante. Nous pourrions partir de $S_0 = \varnothing$ et ajouter itérativement à la solution actuelle S_{j-1} tout maximiseur $i \in I \setminus S_{j-1}$ du gain marginal $\mathrm{G}_f(i \mid S_{j-1})$ jusqu'à ce que $|S_j| = k$. Malheureusement, une telle méthode échouerait ici car f n'est pas forcément *positive* (Nemhauser et al., 1978). Cependant, comme l'ont observé les auteurs de (Boutsidis et al., 2015), ce problème peut être atténué en augmentant légèrement la limite de taille k. Plus précisément, étant donné un paramètre $\gamma \in (0, 1)$, la méthode gloutonne

atteint une approximation de $\left(2^{k\lceil \ln \frac{2e}{1-c}\rceil - |S^*|}\right) \cdot \left(\frac{1}{1-\gamma}\right)$, lorsqu'elle est autorisée à améliorer sa solution S_{j-1} jusqu'à ce que $|S_j| = k\lceil \ln (f(\varnothing)/\gamma f(S_{j-1}))\rceil$. En couplant cette idée avec la méthode de sélection par niveau, nous obtenons l'algorithme GA pour minimiser $\epsilon_{h,\boldsymbol{x}}(\cdot)$.

Algorithm 2 Greedy ascent (GA)

Input: un classifieur h, une instance $\boldsymbol{x}$, un ensemble d'attributs I, un entier k
$j \leftarrow 0$, $S_0 \leftarrow \varnothing$, $\gamma \leftarrow \max\{\frac{1}{e}, c\}$
repeat
  $i^* \leftarrow \operatorname{argmin}_{i \in I \setminus S_{j-1}} \mu_{h,\boldsymbol{x}}(S_{j-1} \cup \{i\})$
  $S_j = S_{j-1} \cup \{i^*\}$
until $j = k\left\lceil \ln\left(\frac{\mu_{h,\boldsymbol{x}}(\varnothing)}{\gamma \cdot \mu_{h,\boldsymbol{x}}(S_j)}\right)\right\rceil$;
$S_{\text{GA}} \leftarrow \operatorname{argmin}_{S \in \{S_0, S_1, \cdots, S_j\}} \epsilon_{h,\boldsymbol{x}}(S)$
return S_{GA}

Proposition 5. *Sous les mêmes conditions que celles données à la proposition 4, la solution* S_{GA} *retournée par (*GA*) satisfait :* $|S_{\text{GA}}| \leq k\left(1 + \left\lceil \ln \frac{\mu_{h,\boldsymbol{x}}(\varnothing)}{\mu_{h,\boldsymbol{x}}(S^*)}\right\rceil\right) \leq k\left\lceil \ln \frac{2e}{1-c}\right\rceil$ *et*

$$\epsilon_{h,\boldsymbol{x}}(S_{\text{GA}}) \leq \left(2^{k\lceil \ln \frac{2e}{1-c}\rceil - |S^*|}\right) \cdot \left(\frac{1}{1-\gamma}\right) \cdot \epsilon_{h,\boldsymbol{x}}(S^*), \text{ où } \gamma = \max\{\frac{1}{e}, c\}$$

5 Expérimentations

Pour valider l'efficacité de nos algorithmes, nous avons considéré diverses instances du problème 1, où le classifieur d'entrée est décrit par un arbre de décision $\mathcal{T}$. Le code a été écrit en utilisant le langage Python. Toutes les expériences ont été conduites sur un ordinateur équipé d'un processeur Intel(R) Core $i9-9900$ cadencé à $3,1$ GHz et avec 64 GiB de RAM.

Protocole expérimental. Nous avons considéré $B = 18$ benchmarks, issus des référentiels standards *Kaggle*, *OpenML* et *UCI*. *mnist38* et *mnist49* sont des sous-ensembles de *mnist*. Pour chaque benchmark $b \in [B]$, une tâche d'explication consiste en un tuple $(\mathcal{T}, \boldsymbol{x}, I, k)$. $\mathcal{T}$ est un arbre de décision, qui a été appris à partir de l'ensemble d'entraînement de b en utilisant l'algorithme CART de `Scikit-Learn`. La précision de $\mathcal{T}$ est mesurée sur l'ensemble de test de b. L'ensemble I est donné par la raison directe pour $\boldsymbol{x}$ étant donné $\mathcal{T}$ (Audemard et al., 2022). La performance de nos algorithmes sur un benchmark b est mesurée en calculant l'erreur moyenne de $\epsilon_{\mathcal{T},\boldsymbol{x}}(S)$ et de la taille moyenne $|S|$ de la sortie $S \subseteq I$ sur $\min\{s, 150\}$ instances, où s est la taille de l'ensemble de test de b. Pour comparer les performances de GD et GA avec une méthode exacte, nous avons choisi l'approche basée sur SAT présentée dans (Arenas et al., 2022). Dans le cadre de notre protocole expérimental, S est un sous-ensemble de la raison directe I pour $\boldsymbol{x}$. L'encodage SAT sus-mentionné a été étendu à la version de décision du problème 1, en ajoutant la clause $\bigvee\{x_i : (x_I)_i = 1\} \vee \{\overline{x}_i : (x_I)_i = 0\}$. Pour la version originale du problème 1, une recherche binaire (dichotomique) sur l'intervalle $]0, 1]$ a été effectuée afin de trouver un ensemble S qui minimise $\epsilon_{\mathcal{T},\boldsymbol{x}}(\cdot)$ avec une précision de 10^{-3}, ce qui nécessite au plus 10 appels au solveur SAT, un temps limite (TO) de 30 minutes a été défini par instance.

Benchmark				$\epsilon_{h,\boldsymbol{x}}(S)$			$\lvert S\rvert$			Time (s)
name	acc	d	$\lvert I\rvert$	GA	GD	SAT	GA	GD	SAT	SAT
glass	78.46	31	5.38	0.26 (±0.11)	0.26 (±0.11)	0.26 (±0.11)	2.14	2.14	2.14	2.36
student perf.	91.79	30	5.41	0.26 (±0.11)	0.26 (±0.11)	0.26 (±0.11)	2.00	2.00	2.00	2.16
primary tumor	84.31	23	6.23	0.09 (±0.09)	0.09 (±0.09)	0.09 (±0.08)	4.22	4.22	4.22	3.58
schizophrenia	80.39	33	6.39	0.37 (±0.24)	0.37 (±0.24)	0.37 (±0.24)	1.27	1.27	1.27	4.79
hungarian	62.92	13	6.65	0.12 (±0.12)	0.12 (±0.12)	0.11 (±0.10)	3.58	3.56	3.56	1.68
horse colic	75.68	40	6.73	0.14 (±0.07)	0.13 (±0.07)	0.13 (±0.07)	4.03	4.06	4.06	11.56
indian liver	64.57	84	8.21	0.10 (±0.09)	0.10 (±0.09)	0.16 (±0.12)	5.08	4.89	6.12	176.28
patient treat.	66.01	10	8.92	0.05 (±0.09)	0.03 (±0.06)	0.03 (±0.08)	5.63	5.94	5.94	24.08
wine	69.58	11	9.03	0.09 (±0.10)	0.09 (±0.09)	0.09 (±0.12)	5.59	5.64	5.62	36.32
employee attr.	82.45	63	10.56	0.06 (±0.09)	0.06 (±0.09)	0.20 (±0.11)	6.41	6.39	6.98	1017.24
contraceptive	51.36	90	10.84	0.06 (±0.08)	0.06 (±0.08)	0.39 (±0.17)	4.27	4.26	5.95	1096.07
compas	67.60	40	10.95	0.03 (±0.07)	0.04 (±0.08)	0.05 (±0.09)	5.68	5.83	6.78	1082.32
dorothea	91.88	10^5	12.90	0.25 (±0.10)	0.25 (±0.10)	–	6.70	6.70	–	–
mnist49	95.99	784	15.57	0.37 (±0.14)	0.37 (±0.14)	–	6.97	6.89	–	–
spambase	92.11	236	16.09	0.24 (±0.11)	0.23 (±0.09)	–	6.87	6.87	–	–
mnist38	96.42	784	17.89	0.37 (±0.13)	0.38 (±0.14)	–	6.93	6.93	–	–
gisette	94.10	5000	21.42	0.32 (±0.11)	0.32 (±0.11)	–	6.88	6.88	–	–
farm ads	80.78	54877	23.15	0.13 (±0.17)	0.13 (±0.17)	–	6.31	6.31	–	–

TAB. 1 – Résultats expérimentaux sur 18 benchmarks pour les explications de $\mathcal{T}$, pour $k = 7$.

Résultats. Dans le tableau 1, nous présentons nos résultats sur 18 benchmarks, pour $k = 7$. Les colonnes acc, d et I donnent respectivement la précision et le nombre d'attributs binaires et la taille moyenne de la raison directe. Les cinquième, sixième et septième colonnes présentent l'erreur moyenne $\epsilon_{\mathcal{T},\boldsymbol{x}}(S)$ de l'explication S renvoyée par GA, GD, et l'approche SAT, respectivement. Les trois colonnes suivantes rapportent la taille moyenne de S pour ces algorithmes. La dernière colonne donne les temps d'exécution moyens (en secondes) de l'approche SAT. Il est important de noter que, pour les benchemarks en *bleu*, le solveur SAT atteint le temps limite de 30 minutes avant même la fin de la recherche dichotomique, ce qui entraîne une dégradation de la précision. Pour les benchmarks en *magenta*, le solveur n'arrive pas à donner des résultats même pour seule instance avant d'atteindre le temps limite de 30 minutes.

Nous observons que les performances de GA et GD sont remarquables, en particulier en comparaison avec l'approche SAT. Pour les benchmarks où le solveur SAT pouvait renvoyer une solution optimale S^*, les différences $\epsilon_{\mathcal{T},\boldsymbol{x}}(S_{\text{GD}}) - \epsilon_{\mathcal{T},\boldsymbol{x}}(S^*)$ et $\epsilon_{\mathcal{T},\boldsymbol{x}}(S_{\text{GA}}) - \epsilon_{\mathcal{T},\boldsymbol{x}}(S^*)$ sont le plus souvent négligeables. De plus, pour les benchmarks de grande dimension tels que *dorothea*, *gisette* et *farm ads*, GA et GD restent stables en fournissant des explications avec des erreurs comparables en quelques dixièmes de milliseconde. Nous observons également que $|S_{\text{GD}}|$ et $|S_{\text{GA}}|$ sont en moyenne plus petit que $|S^*|$. Enfin, GA et GD ont pu réduire les raisons directe I considérées. Autrement, les deux algorithmes sont suffisamment robustes pour traiter les tâches d'explication pour lesquelles la courbure c de $\mu_{h,\boldsymbol{x}}$ est proche ou égale à 1.

6 Conclusion

Dans notre étude, nous avons examiné les explications probabilistes sous l'angle de la minimisation super-modulaire. Nous avons proposé deux algorithmes d'approximation gloutonne pour minimiser les erreurs d'explication sous une contrainte de cardinalité. Les performances de ces algorithmes dépendent principalement de la courbure c de la fonction d'erreur non normalisée $\mu_{h,\boldsymbol{x}}(\cdot)$. Nos expérimentations démontrent que nos algorithmes gloutons offrent une alternative trés efficace à la méthode exacte basée sur un encodage SAT.

Références

Arenas, M., P. Barceló, M. Romero Orth, et B. Subercaseaux (2022). On computing probabilistic explanations for decision trees. *Advances in Neural Information Processing Systems 35*, 28695–28707.

Audemard, G., S. Bellart, L. Bounia, F. Koriche, J.-M. Lagniez, et P. Marquis (2022). On the explanatory power of boolean decision trees. *Data Knowledge Engineering 142*, 102088.

Bounia, L. et F. M. Koriche (2023). Approximating probabilistic explanations via supermodular minimization. In *The 39th Conference on Uncertainty in Artificial Intelligence*.

Boutsidis, C., E. Liberty, et M. Sviridenko (2015). Greedy minimization of weakly supermodular set functions. *arXiv preprint arXiv :1502.06528*.

Conforti, M. et G. Cornuéjols (1984). Submodular set functions, matroids and the greedy algorithm : Tight worst-case bounds and some generalizations of the rado-edmonds theorem. *Discret. Appl. Math. 7*, 251–274.

Il'ev, V. P. (2001). An approximation guarantee of the greedy descent algorithm for minimizing a supermodular set function. *Discrete Applied Mathematics 114*(1), 131–146. discrete analysis operations research.

Izza, Y., X. Huang, A. Ignatiev, N. Narodytska, M. C. Cooper, et J. Marques-Silva (2022). On computing probabilistic abductive explanations. *Int. J. Approx. Reason. 159*, 108939.

Miller, G. A. (1956). The magical number seven, plus or minus two : Some limits on our capacity for processing information. *The Psychological Review 63*(2), 81–97.

Nemhauser, G., L. Wolsey, et M. Fisher (1978). An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming 14*, 265–294.

Wäldchen, S., J. Macdonald, S. Hauch, et G. Kutyniok (2021). The computational complexity of understanding binary classifier decisions. *J. Artif. Intell. Res. 70*, 351–387.

Summary

An *abductive explanation* for the predicted label of some data instance is a subset-minimal collection of features such that the restriction of the instance to these features is sufficient to determine the prediction. However, due to cognitive limitations, abductive explanations are often too large to be interpretable. In those cases, we need to reduce the size of abductive explanations, while still determining the predicted label with high probability. In this paper, we show that finding such *probabilistic explanations* is NP-hard, even for decision trees. In order to circumvent this issue, we investigate the *approximability* of probabilistic explanations through the lens of supermodularity. We examine both greedy descent and greedy ascent approaches for supermodular minimization, whose approximation guarantees depend on the curvature of the "unnormalized" error function that evaluates the precision of the explanation. Based on various experiments for explaining decision tree predictions, we show that our greedy algorithms provide an efficient alternative to the state-of-the-art constraint optimization method.