CentraleSupélec

Systèmes de Décision
Vincent Mousseau
Wassila Ouerdane
Anaëlle Wilczynski

Introduction: student admission

Consider a situation in which a committee for a higher education program has to decide about the admission of students on the basis of their evaluations in 4 courses: mathematics (M), physics (P), literature (L) and history (H). Evaluations on all courses range in the [0,20] interval. To be accepted ($\mathcal{A}$) in the program, the committee considers that a student should obtain at least 12 on a "*majority*" of courses, otherwise, the student is refused ($\mathcal{R}$). From the committee point of view, all courses (criteria) do not have the same importance. To define the required majority of courses, the committee attaches a weight $w_j \geq 0$ to each course such that they sum to 1; a subset of courses $\mathcal{C} \subseteq \{M, P, L, H\}$ is considered as a majority if $\sum_{j \in C} w_j \geq \lambda$, where $\lambda \in [0, 1]$ is a required majority level.

i. Define the weights w_j, and λ when a student is admitted when (s)he obtains evaluation above 12 on 3 courses out of 4.

Solution:

We can set $w_j = \frac{1}{4}$, $\forall j$ with $\lambda = \frac{3}{4}$.
But $w_L = w_H = 0.49$, $w_M = w_P = 0.02$ with $\lambda = 0.51$ would also be convenient.

ii. If there were only 3 courses, how would you interpret the following values for w_j, and λ.

- $w = (0.49, 0.49, 0.02)$ with $\lambda = 0.5$?
- $w = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ with $\lambda = 0.5$?

What can you conclude from this observation?

Solution:

Both cases correspond to the same situation: majorities are subsets of criteria with at least two out of three criteria. We can conclude that there exists several ways to represent the set of sufficient majorities using a weights vector and majority level λ. As a consequence, we should be cautious, and avoid interpreting the weight value as an indication of the relative importance of criteria.

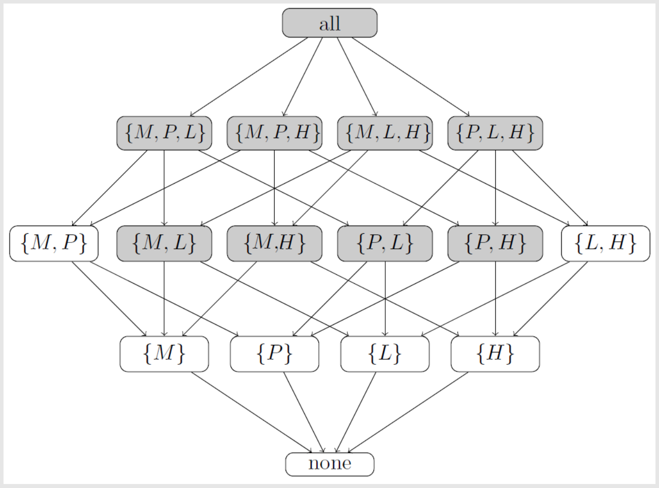
iii. The committee considers that the strength of a coalition of courses varies as a function of the courses belonging to the coalition, and states that in order to be accepted a student should be evaluated above 12/20 on a scientific course (math or physics), and on a non-scientific course (literature or history). Prove that it is not possible to represent such an admission rule with an additive definition of majority. What can you conclude?

Solution:

Suppose that it is possible to represent this set of sufficient coalitions using additive weights. then is should hold:

(i) $w_M + w_L \geq \lambda$,
(ii) $w_M + w_H \geq \lambda$,
(iii) $w_P + w_L \geq \lambda$,
(iv) $w_P + w_H \geq \lambda$,
(v) $w_M + w_P < \lambda$,
(vi) $w_L + w_H < \lambda$,

However, summing (i) and (iv) we get $w_M + w_P + w_L + w_H \geq 2\lambda$, and summing (v) and (vi) we get $w_M + w_P + w_L + w_H < 2\lambda$. Therefore, this set of sufficient majorities is not additively representable.



iv. In order to represent the "sufficient sets" of criteria, we do not consider additive weights anymore. We denote $\mathcal{N} = \{1, 2, ..., n\}$ the set of criteria, and $\mathcal{P}(\mathcal{N})$ the set of subsets of $\mathcal{N}$. We consider (Maj, Min) a bi-partition of $\mathcal{P}(\mathcal{N})$: $\mathcal{P}(\mathcal{N}) = Maj \cup Min$ and $Maj \cap Min = \emptyset$, where Maj (Min, resp.) can be interpreted as the subsets of criteria in $\mathcal{N}$ which forms a *majority* (a *minority*, resp.). For such an interpretation, we pose:

(i) $\mathcal{N} \in Maj$ and $\emptyset \in Min$,
(ii) $\forall A, B \in \mathcal{P}(\mathcal{N})$ such that $A \subset B, A \in Maj \Rightarrow B \in Maj$ and $B \in Min \Rightarrow A \in Min$

How can you check whether a given a bi-partition is additively representable by a weight vector w and a threshold λ ?

Solution:

To test whether a bi-partition (Maj, Min) is additively representable, let us proceed as follows:
Consider weights $w_j \geq 0$ such that $\sum_{j\in\mathcal{N}} w_j = 1$, and a majority level *lambda*. For each coalition $\mathcal{C} \in Maj$, it should hold $\sum_{j\in\mathcal{C}} w_j \geq \lambda$. For each coalition $\mathcal{C} \in Min$, it should hold $\sum_{j\in\mathcal{C}} w_j < \lambda$. Therefore, checking whether the bi-partition (Maj, Min) is additively representable, boils down to checking whether the polyedron

$$\begin{cases} \sum_{j\in\mathcal{C}} w_j \geq \lambda, \forall \mathcal{C} \in Max \\ \sum_{j\in\mathcal{C}} w_j < \lambda, \forall \mathcal{C} \in Min \\ w_j \geq 0,\ \sum_{j\in\mathcal{N}} w_j = 1 \end{cases}$$

is empty or not (if it is non-empty, then the bi-partition (Maj, Min) is additively representable, otherwise it is not).

v. A capacity is a set function $\mu : 2^{\mathcal{N}} \to [0,1]$ such that:

(i) $\mu(A) \leq \mu(B)$, for all $A \subseteq B \subseteq \mathcal{N}$

(ii) $\mu(\emptyset) = 0$ and $\mu(\mathcal{N}) = 1$.

define a capacity μ from which you can derive the bi-partition implementing the admission rule defined in iii.

Solution:

$$\begin{cases} \mu(\mathcal{N}) = 1 \\ \mu(\{M,P,L\}) = \mu(\{M,P,H\}) = \mu(\{M,L,H\}) = \mu(\{P,L,H\}) = \frac{2}{3} \\ \mu(\{M,L\}) = \mu(\{M,H\}) = \mu(\{P,L\}) = \mu(\{P,H\}) = \frac{1}{2} \\ \mu(\{M,P\}) = \mu(\{L,H\}) = \mu(\{M\}) = \mu(\{P\}) = \mu(\{L\}) = \mu(\{H\}) = \frac{1}{3} \\ \mu(\emptyset) = 0 \end{cases} \quad \text{with } \lambda = \frac{1}{2}$$

vi. The following table shows:

- the number $n(|\mathcal{N}|$ of bi-partitions of $\mathcal{P}(\mathcal{N})$ verifying (i) $\mathcal{N} \in Maj$ and $\emptyset \in Min$, and (ii) $\forall A, B \in \mathcal{P}(\mathcal{N})$ such that $A \subset B, A \in Maj \Rightarrow B \in Maj$ and $B \in Min \Rightarrow A \in Min$,
- the number n^{add} of additively representable such bi-partitions, and
- the proportion $p^{add}(|\mathcal{N}|)$ of additively representable such bi-partitions,

as a function of $|\mathcal{N}|$. What can be learned from this table?

$\lvert\mathcal{N}\rvert$	3	4	5	6
$n(\lvert\mathcal{N}\rvert)$	20	168	7581	7828354
$n^{add}(\lvert\mathcal{N}\rvert)$	20	150	3287	244158
$p^{add}(\lvert\mathcal{N}\rvert)$	100%	89.1%	43.4%	3.1%

Solution:

- The number of bi-partitions of $\mathcal{P}(\mathcal{N})$ compatible with inclusion increases very quickly with $|\mathcal{N}|$. Indeed $n(7) = 2414682040998$ and $n(8) = 56130437228687557907788$ ($n(9)$ is not even known). It corresponds to the so-called Dedekind numbers (`https://oeis.org/A000372`).
- As $|\mathcal{N}|$ increases, the $p^{add}(|\mathcal{N}|)$ tends to be very small.
- There is a huge gap between bi-partitions of $\mathcal{P}(\mathcal{N})$ and additively representable bi-partitions of $\mathcal{P}(\mathcal{N})$.

vii. We are now interested in learning such a sorting model from a dataset. We consider the simple case with two categories accept ($\mathcal{A}$) and reject ($\mathcal{R}$). More precisely, suppose we have a dataset of accepted objects A^*, and rejected objects R^*. We suppose that the coalitions of criteria are represented using additive weights w_j and a majority threshold λ (as in i. and ii.), but unknown. Formulate a linear program which learn w_j and λ that best match the dataset (provided that the minimum acceptable evaluation on each criterion is known, 12 for each criterion in our example).

Solution:

...

viii. In what follows, the limit profile defining the frontier between the two classes $b = (b_1, b_2, ..., b_n)$ is unknown and yet to be learned. We define Boolean variables $\delta_i(s) \in \{0, 1\}$, for each criterion i and each student s, such that $\delta_i(s) = 1 \Leftrightarrow s_i \geq b_i$ and $\delta_i(s) = 0 \Leftrightarrow s_i < b_i$.

Show that if constraints $M(\delta_i(s) - 1) \leq s_i - b_i < M.\delta_i(s)$ hold, then variables $\delta_i(s)$ behave as desired (M being an arbitrarily large positive constant).

Solution:

- If $s_i \geq b_i$, then $s_i - b_i \geq 0$. If $s_i - p_i < M.\delta_i(s)$ hold, then $M.\delta_i(s)$ should be strictly positive, which enforces $\delta_i(s) = 1$.
- If $s_i < b_i$, then $s_i - b_i < 0$. If $M(\delta_i(s) - 1) \leq s_i - b_i$ hold, then $M(\delta_i(s) - 1)$ should be strictly negative, which enforces $\delta_i(s) = 0$.

ix. From these Boolean variables $\delta_i(s)$, we define continuous variables $w_i(s)$, for each criterion i and each student s, such that:

$$w_i(s) = \begin{cases} w_i & \text{, if } s_i \geq b_i \\ 0 & \text{, otherwise.} \end{cases}$$

where w_i represents the weight of criterion i. Prove that if the constraints defined bellow hold, then variables $\delta_i(s)$ and $w_i(s)$ behave as desired. What does the expression $\sum_i w_i(s)$ represent? Deduce a condition on variables $w_i(s)$ that ensures that student s is preferred to b.

$$\begin{cases} w_i \geq w_i(s) \geq 0 \\ \delta_i(s) \geq w_i(s) \geq \delta_i(s) + w_j - 1 \end{cases}$$

Solution:

The expression $\sum_i w_i(s)$ represents the sum of the weights of criteria for which $s_i \geq p_i$.

x. Write the mathematical program that makes it possible to learn simultaneously the limit profile defining the frontier between the two classes $b = (b_1, b_2, ..., b_n)$, and the winning criteria coalitions (w_j and λ) from a learning set.

Solution:

...

xi. Implement a program that learn the frontiers between consecutive classes (with more than two classes), and the winning criteria coalitions (w_j and λ) from a learning set provided as input.

Solution:

...

xii. Test your program on a learning set generated from a compatible ground truth.

Solution:

...

xiii. We want now to elaborate a logical formulation to check for the existence of a NCS model compatible with a learning set. Formally, this amounts at formulating a set of clauses (disjunction of logical variables) that ensure that the learning set can be restored by a NCS model. We will first restrict to the case where there are two classes only. With two classes, the NCS rule can be expressed as follows :

$$x = (x_1, ...x_n) \in \mathcal{A} \quad \Leftrightarrow \quad \{i : x_i \geq b_i\} \in \mathcal{S}$$

where $\mathcal{S}$ is the set of sufficient criteria coalitions. We will consider the following binary variables:

- $\alpha_{ki} = true$ iff the evaluation k on criterion i if above the frontier ($k \geq b_i$), $\forall k$ a performance on criterion i in the learning set.
- $\beta_C = true$ iff the coalition of criterion is a majority, $\forall C \subseteq \mathcal{N}$.

We consider a learning set $L = A^* \cup R^*$, where A^* (R^*, resp.) are accepted (rejected, resp.) students. We will suppose that the evaluations on all criteria are to be maximized (the greater the better). Express clauses that ensure this on the learning set L.

Solution:

...

xiv. The set of "sufficient" criteria coalitions should be monotone wrt inclusion, i.e., if $C \subseteq C'$ and C is sufficient, then C' is sufficient. Express clauses that ensure this condition.

Solution:

...

xv. Express clauses that ensure that all students in A^* are accepted. A student is accepted ($s \in \mathcal{A}$) iff the set of criteria i for which s has an evaluation above the frontier ($s_i \geq b_i$) is sufficient.

Solution:

...

xvi. Express clauses that ensure that all students in R^* are rejected. A student is rejected ($s \in \mathcal{R}$) iff the set of criteria i for which s has an evaluation above the frontier ($s_i \geq b_i$) is not sufficient.

Solution:

...

xvii. Implement, using a SAT solver a program that check whether a learning set can be represented by an NCS model. Formulate it at first with 2 classes, then extend it to more than 2 classes.

Solution:

...

References

[1] D. Bouyssou, T. Marchant, An axiomatic approach to noncompensatory sorting methods in MCDM, I: The case of two categories, *European Journal of Operational Research*, 178(1):217–245,(2007).

[2] D. Bouyssou, T. Marchant, An axiomatic approach to noncompensatory sorting methods in MCDM, II: More than two categories, *European Journal of Operational Research*, 178(1):246–276, (2007).

[3] Eda Ersek Uyanik, Vincent Mousseau, Marc Pirlot, and Olivier Sobrie. Enumerating and categorizing positive boolean functions separable by a k-additive capacity. *Discrete Applied Mathematics*, 229:17-30, (2017).

[4] Agnes Leroy, Vincent Mousseau, and Marc Pirlot. Learning the parameters of a multiple criteria sorting method. *Algorithmic Decision Theory*, 219-233, (2011).

[5] Belahcene K., Labreuche C., Maudet N., Mousseau V., and Ouerdane, W. An efficient SAT formulation for learning multiple criteria non-compensatory sorting rules from examples, Computers & Operations Research, 97, 58–71, (2018).

Cahier des charges du projet

Les formulations de problèmes d'apprentisage de préférences en terme d'optimisation et de Satisfiabilité booléennes (SAT/MaxSAT) et les solveurs associés constituent des outils puissants pour l'intelligence artificielle. On considère dans ce projet les problèmes Inv-MR-Sort et Inv-NCS qui visent à apprendre les modèles de classification ordonnée (MR-Sort et NCS). L'objet du projet est d'implémenter Inv-MR-Sort à l'aide d'un solveur d'optimisation, et Inv-NCS à l'aide d'un solveur SAT/MaxSAT.

Votre travail devra permettre de comparer l'efficacité de ces deux formulations sur des jeux de données. Les comparaisons devront s'appuyer sur des jeux de données que vous préciserez et concernera: i) le temps de calcul, ii) la capacité à apprendre un jeux de données et l'aptitude à généraliser et iii) s'adapter à des données bruitées.

- Le projet est à réaliser en groupe de 3 (trinômes), les groupes doivent être formés le 17/12/2021, et inscrit sur la "feuille" dont le lien se trouve sur Edunao.
- Deux séances de cours sont consacrées au projet (17/12/2021 et 25/01/2022).
- Deux livrables du projets sont demandés :
  - un livrable intermédiaire pour le 14/01/2022 qui implémente l'apprentissage d'un modèle MR-Sort à partir d'un learning set (par programmation linéaire / Gurobi) et l'apprentissage d'un modèle NCS à partir d'un learning set (SAT/MaxSAT),
  - puis le livrable final pour le 01/02/2022 dont les spécifications seront précisées ultérieurement.

  Pour chaque livrable : préparer un gitlab (celui de l'école) qui contiendra l'ensemble des modules implémentés et les Dataset. En faisant attention à:

  - mettre des commentaires[1] à chaque fonction pour dire ce qu'elle fait. Il existe aussi des conventions d'écriture de code (PEP 8[2] par exemple). Vous pouvez les lire et essayez de les respecter,
  - construire un fichier README[3] : fichier qui contient un texte explicatif sur le projet. Il faudrait avoir à minima :
    * Le nom de chacun des élèves du groupe,
    * Un rappel du sujet,
    * Une petite explication de votre structure de fichiers : où est le code à exécuter/le main ? Avez-vous un (des) fichier(s) de tests ? Comment avez-vous organisé le code ?
    * Les instructions à taper pour faire tourner le code (pour mettre vos correcteurs dans une bonne disposition, et leur épargner 5-10 minutes de recherche dans votre code pour trouver tous seuls),
    * Les modules à importer si besoin
  - rajouter Vincent Mousseau, Wassila Ouerdane et Anaelle Wilczynski comme maintainers à vos gits.

[1] voir: `https://realpython.com/documenting-python-code/`
[2] `https://realpython.com/python-pep8/`
[3] `https://www.makeareadme.com/`