LINCS: a Python package for learning Non-Compensatory Sorting models

Khaled Belahcène, Laurent Cabaret, Vincent Jacques,
Vincent Mousseau, Wassila Ouerdane

MICS - CentraleSupélec - Université Paris Saclay
{prénom.nom}@centralesupelec.fr,
vincent@jacques.net

Mots-clés : *Multicriteria decision, Sorting, preference learning.*

1 Introduction

Multiple criteria sorting models (see [3]) consider alternatives $(a, b, c, ...)$ evaluated on n criteria. We suppose wlog that evaluations on each criterion, defined on a scale $X_i \subset \mathbb{R}$, are to be maximized. Sorting models aim at classifying an alternative $a \in X = \prod_{i=1}^{n} X_i$ to one of the predefined ordered classes. Non-compensatory Sorting (NCS, [4]) is one of such multicriteria sorting models. Implementing an NCS model requires setting the values for the preference related parameters (class limits, importance of criteria coalitions) so as to fit to the Decision-Maker (DM) preferences.

2 The Non-compensatory Sorting Model

We limit our presentation of NCS to the case with two classes Good and Bad. The parameters that describe a specific NCS model are *(i)* a limiting profile $b = (b_1, ...b_n) \in X$ that defines, for each criterion i, a set $\mathcal{A}_i \subset X_i$ of approved values at least as good as b_i (and, by contrast, a lower set $X \setminus \mathcal{A}_i \subset X_i$ of disapproved values strictly worse than b_i), and *(ii)* a set $\mathcal{T}$ of sufficient coalitions of criteria, which satisfies monotonicity with respect to inclusion. These notions are combined into the following assignment rule: $a \in \textsc{Good} \iff \{i \in \mathcal{N} : x_i \geq b_i\} \in \mathcal{T}$. An interesting particular case of NCS (referred in the literature as MR-Sort, [5]) occurs when the set $\mathcal{T}$ of sufficient coalitions of criteria is additively representable using weights w_i attached to criteria and a threshold λ, i.e., $\mathcal{T} = \{t \in 2^{\mathcal{N}} : \sum_{i \in t} w_i \geq \lambda\}$.

3 Inv-NCS: Infering an NCS Model from a learning set

Implementing an NCS model requires setting the values for the parameters (class limits b, sufficient criteria coalitions $\mathcal{T}$) so as to fit to the Decision-Maker (DM) preferences. Checking whether there exists an NCS model that correctly classifies a learning set is known to be difficult [2]. Several algorithms have however be proposed to solve this Inv-NCS problem: an exact algorithm based on logical formulation [1, 8], and an evolutionary heuristic [7]. Up to now, no library has been implementing these methods in a common framework and on an open source perspective.

4 LINCS: Learning and Infering Non-Compensatory Sorting

This paper aims at presenting a new Python library, Lincs (**L**earning and **I**nferring **N**on-**C**ompensatory **Sorting** models, `pypi.org/project/lincs/`), which implements and makes available the algorithms that have been proposed to solve the Inv-NCS problem. These algorithms have been re-implemented in C++ and made available as a Python library. An important feature of this library results from the fact that it runs both on CPU and GPU architectures.

5 Performance analysis

We performed an extended numerical analysis to have a precise view on the performance of the algorithms implemented in the Lincs library. It appears that these new implementations allow us to cope with datasets of much larger sizes than what is usually tackled in multiple criteria decision making applications, e.g., 5 categories 20 criteria, and 20000 alternatives in the learning set. During the presentation, we will thoroughly analyze the experimental investigations we made.

6 Conclusions and perspectives

This library is intended to be extended in several ways. First, we intend to integrate the possibility of considering single-peaked criteria as proposed by [6]. Another important perspective is related to the fact that the library makes it possible to handle relatively large datasets (up to 20 criteria). With instances of such size, infered NCS models are hardly interpretable, as it is the case for models with a limited number of criteria. Hence, an important field of research consists in introducing parsimony in preference learning algorithms.

References

[1] K. Belahcène, C. Labreuche, N. Maudet, V. Mousseau, and W. Ouerdane. An efficient SAT formulation for learning multiple criteria non-compensatory sorting rules from examples. *Computers & OR*, 97:58–71, 2018.

[2] K. Belahcène, Christophe Labreuche, Nicolas Maudet, V. Mousseau, and W. Ouerdane. A model for accountable ordinal sorting. In *Proceedings of IJCAI*, pages 814–820, 2017.

[3] K. Belahcene, V. Mousseau, M. Pirlot, W. Ouerdane, and O. Sobrie. Multiple criteria sorting models and methods. part I: survey of the literature. *4OR*, 21(1):1–43, 2023.

[4] D. Bouyssou and T. Marchant. An axiomatic approach to noncompensatory sorting methods in MCDM, I: the case of two categories. *EJOR*, 178(1):217–245, 2007.

[5] A. Leroy, V. Mousseau, and M. Pirlot. Learning the parameters of a multiple criteria sorting method. In *Algorithmic Decision Theory*, volume 6992 of *LNAI*, pages 219–233, 2011.

[6] Pegdwende Minoungou. *Learning Majority-Rule models with partially monotone data.* PhD thesis, Université Paris-Saclay, 2022.

[7] O. Sobrie, V. Mousseau, and M. Pirlot. Learning monotone preferences using a majority rule sorting model. *ITOR*, 26(5):1786–1809, 2019.

[8] A. Tlili, K. Belahcène, O. Khaled, V. Mousseau, and W. Ouerdane. Learning non-compensatory sorting models using efficient SAT/MaxSAT formulations. *EJOR*, 298(3):979–1006, 2022.