Learning majority rule sorting models from large learning sets

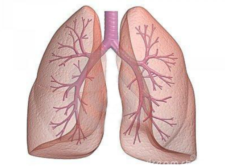
Vincent Mousseau[1] - Marc Pirlot[2] - Olivier Sobrie[1,2]

[1]École Centrale de Paris - Laboratoire de Génie Industriel
[2]University of Mons - Faculty of engineering

April 8, 2014

Introductory example

Application : Lung cancer



Categories :

C_3 : No cancer

C_2 : Curable cancer

C_1 : Incurable cancer

$$C_3 \succ C_2 \succ C_1$$

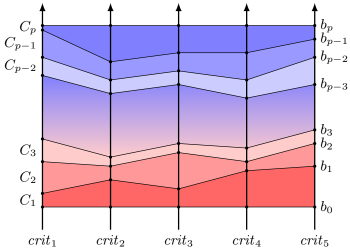
- 9394 patients analyzed
- Monotone attributes (number of cigarettes per day, age, ...)
- Output variable : no cancer, cancer, incurable cancer
- Predict the risk to get a lung cancer for other patients on basis of their attributes

MR-Sort procedure

Main characteristics

- Sorting into ordered categories (ordinal classification)
- based on multicriteria evaluation (monotone attributes)
- Simplified version of ELECTRE TRI procedure [?]
- Axiomatic analysis [?], [?], [?]

Parameters



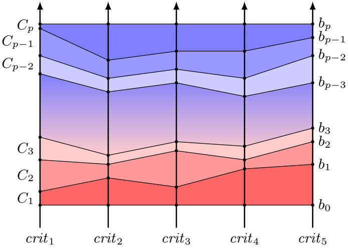
- Profiles' evaluations (b_j^h for $h = 1, ..., p-1; j = 1, ..., n$)
- Criteria weights (w_j for $n = 1, ..., n$)
- Majority threshold (λ)

MR-Sort procedure

Main characteristics

- Sorting into ordered categories (ordinal classification)
- based on multicriteria evaluation (monotone attributes)
- Simplified version of ELECTRE TRI procedure [?]
- Axiomatic analysis [?], [?], [?]

Parameters



Assignment rule

$$a \in C_h$$
$$\Leftrightarrow$$
$$\sum_{j:a_j \geq b_j^{h-1}} w_j \geq \lambda \text{ and } \sum_{j:a_j \geq b_j^{h}} w_j < \lambda$$

Inferring the parameters

Multicriteria Decision Aid perspective

- "Model based learning" (restricted model class)
- Indirect preference information : assignment examples
- Assignment examples are provided by DMs (dataset of limited size)
- Emphasis on interaction (iterative learning process)
- The DM gets insights on her preference from the elicitation procedure
- Emphasis on the interpretability of the model

Preference Learning perspective

- Focus on data and algorithmic efficiency
- Very large learning set
- No (little) interaction (active learning)
- Emphasis on classification accuracy

Inferring the parameters

What already exists to infer MR-Sort parameters ?

- MIP based learning of an MR-Sort model [?]
- Not suitable for large instances : <100 examples (number of binary variables)
- Metaheuristic to learn an ELECTRE TRI model [?]

Our objective

- Learn a MR-Sort model from a large set of assignment examples
- Efficient algorithm (e.g. 1000 alternatives, 10 criteria, 5 categories)

→ one step from MCDA towards preference learning...

Principle of the metaheuristic

Input data

- Examples described by n monotone attributes (criteria)
- Assignment of examples to ordered categories

Objective

- Learn an MR-Sort model restoring a maximum number of examples (classification accuracy),

State of the art

- Learning only the weights and majority threshold : easy (LP)
- Learning only the profiles : Difficult (MIP)

Principle of the metaheuristic

Evolutionary algorithm

- Manage a population of MR-Sort models
- Evolve the population by iteratively
  - optimize weights (profiles fixed)
  - improve profiles (weights fixed)
- ... to get a "good" MR-Sort model in the population

Metaheuristic to learn all the parameters

Algorithm

Generate a population of N MR-Sort models
repeat
  for all MR-Sort model **do**
    Given the current profiles, learn optimal weights w_j and λ using LP,
    Using these w_j and λ, adjust profiles b_h with a heuristic
  end for
  Reinitialize the $\left\lfloor \frac{N}{2} \right\rfloor$ models with the worst CA
until Stopping criterion is met

Stopping criterion

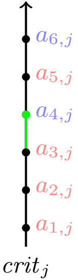
Stopping criterion : one of the models matches all examples or after max_{it} iterations.

Profiles initialization

Principle

- Using a heuristic
- On each criterion j, give to the profile a performance such that CA would be max for the alternatives belonging to h and $h+1$ if $w_j = 1$.
- Take the probability to belong to a category into account

Example : Where should the profile be set on criterion j ?



$a_{i,j}$	Category
$a_{1,j}$	C_1
$a_{2,j}$	C_1
$a_{3,j}$	C_1
$a_{4,j}$	C_2
$a_{5,j}$	C_1
$a_{6,j}$	C_2

Category	$P(a_i \in C_h)$
C_1	$\frac{2}{3}$
C_2	$\frac{1}{3}$

$$C_2 \succ C_1$$

$$a_{3,j} < b_h \leq a_{4,j}$$

Learning the weights and the majority threshold

Principle

- Optimize weights to minimize assignment violation
- Using a linear program without binary variables

Linear program

$$\min \sum_{a \in A} (x'_a + y'_a) \tag{1}$$

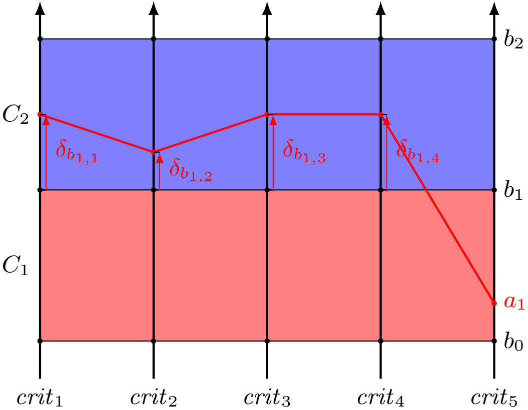
$$\sum_{j : a_j \geq b_j^{h-1}} w_j - x_a + x'_a = \lambda \qquad \forall a \in A_h, h = \{2, ..., p-1\} \tag{2}$$

$$\sum_{j : a_j \geq b_j^{h}} w_j + y_a - y'_a = \lambda - \varepsilon \qquad \forall a \in A_h, h = \{1, ..., p-2\} \tag{3}$$

$$\sum_{j=1}^{n} w_j = 1 \tag{4}$$

Learning the profiles

Case 1 : Alternative a_1 classified in C_2 instead of C_1 ($C_2 \succ C_1$)

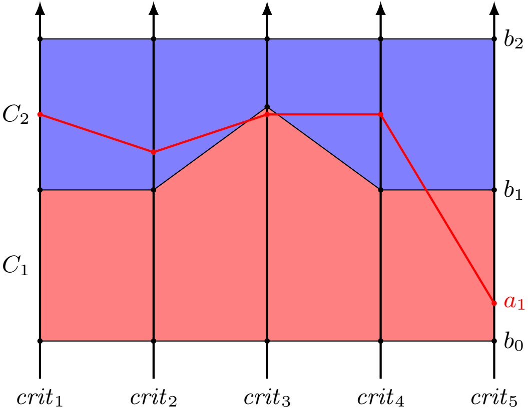


- a_1 is classified by the **DM** into category C_1
- a_1 is classified by the **model** into category C_2
- a_1 outranks b_1
- Profile too low on one or several criteria (in red)

$w_j = 0.2$ for $j = 1, ..., 5$; $\lambda = 0.8$

Learning the profiles

Case 1 : Alternative a_1 classified in C_2 instead of C_1 ($C_2 \succ C_1$)

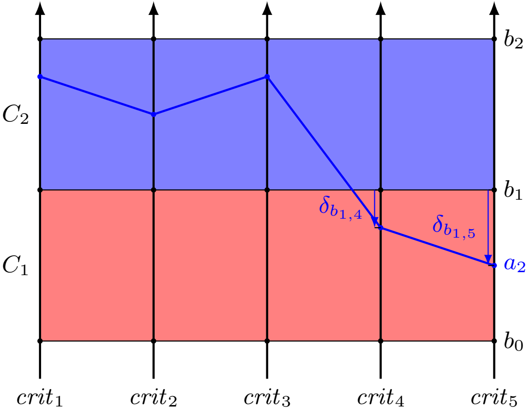


- a_1 is classified by the **DM** into category C_1
- a_1 is classified by the **model** into category C_2
- a_1 outranks b_1
- Profile too low on one or several criteria (in red)

$w_j = 0.2$ for $j = 1, ..., 5$; $\lambda = 0.8$

Learning the profiles

Case 2 : Alternative a_2 classified in C_1 instead of C_2 ($C_2 \succ C_1$)

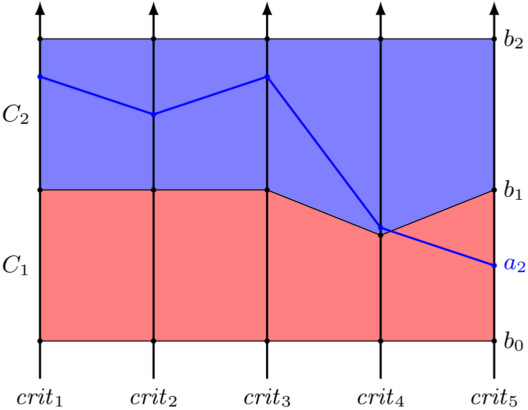


$w_j = 0.2$ for $j = 1, ..., 5$; $\lambda = 0.8$

- a_2 is classified by the **DM** into category C_2
- a_2 is classified by the **model** into category C_1
- a_2 doesn't outrank b_1
- Profile too high on one or several criteria (in blue)
- If profile moved by $\delta_{b_1,2,4}$ on g_4 and/or by $\delta_{b_1,2,5}$ on g_5, the alternative will be rightly classified

Learning the profiles

Case 2 : Alternative a_2 classified in C_1 instead of C_2 ($C_2 \succ C_1$)



$w_j = 0.2$ for $j = 1, ..., 5$; $\lambda = 0.8$

- a_2 is classified by the **DM** into category C_2
- a_2 is classified by the **model** into category C_1
- a_2 doesn't outrank b_1
- Profile too high on one or several criteria (in blue)
- If profile moved by $\delta_{b_1,2,4}$ on g_4 and/or by $\delta_{b_1,2,5}$ on g_5, the alternative will be rightly classified

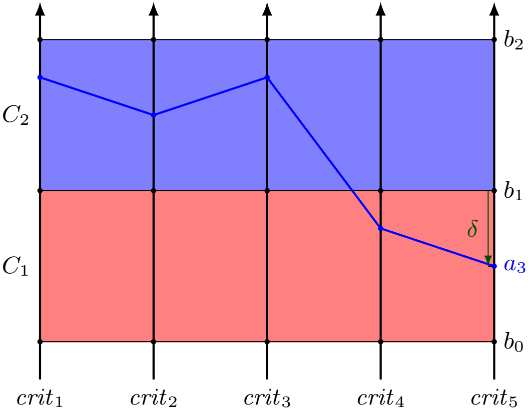
Learning the profiles

For a given δ change for b_j^h, we define subsets in the learning set

- incorrect assignment $\rightarrow$ correct assignment
- incorrect assignment $\rightarrow$ incorrect assignment but "strengthened coalition"
- incorrect assignment $\rightarrow$ incorrect assignment and "weakened coalition"
- correct assignment $\rightarrow$ incorrect assignment
- correct assignment $\rightarrow$ correct assignment but "weakened coalition"
- correct assignment $\rightarrow$ correct assignment and "strengthened coalition"

Learning the profiles

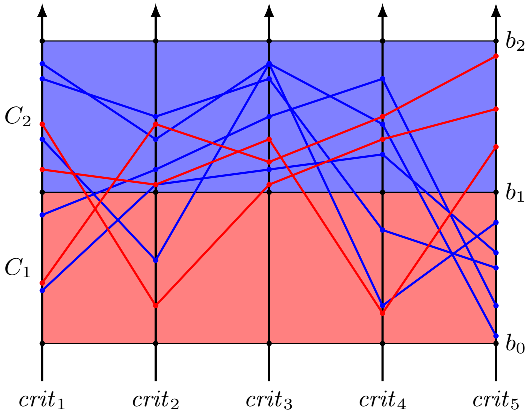
- $V^{+\delta}_{h,j}$ (resp. $V^{-\delta}_{h,j}$) : the sets of alternatives misclassified in C_{h+1} instead of C_h (resp. C_h instead of C_{h+1}), for which moving the profile b_h by $+\delta$ (resp. $-\delta$) on j results in a correct assignment.



- $C_2 \succ C_1$
- $w_j = 0.2$ for $j = 1, ..., 5$
- $\lambda = 0.8$
- $a_3 \in A^{1 \leftarrow \text{Model}}_{2 \leftarrow \text{DM}}$

Learning the profiles

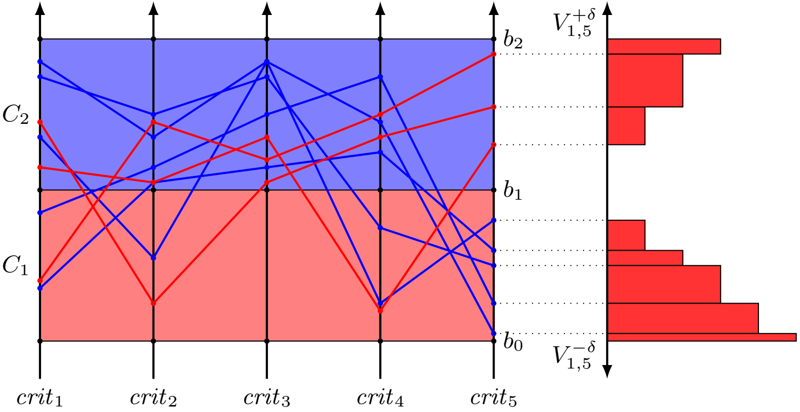
- $V^{+\delta}_{h,j}$ (resp. $V^{-\delta}_{h,j}$) : the sets of alternatives misclassified in C_{h+1} instead of C_h (resp. C_h instead of C_{h+1}), for which moving the profile b_h by $+\delta$ (resp. $-\delta$) on j results in a correct assignment.



- $C_2 \succ C_1$
- $w_j = 0.2$ for $j = 1, ..., 5$
- $\lambda = 0.8$
- $a_3 \in A^{1 \leftarrow \text{Model}}_{2 \leftarrow \text{DM}}$

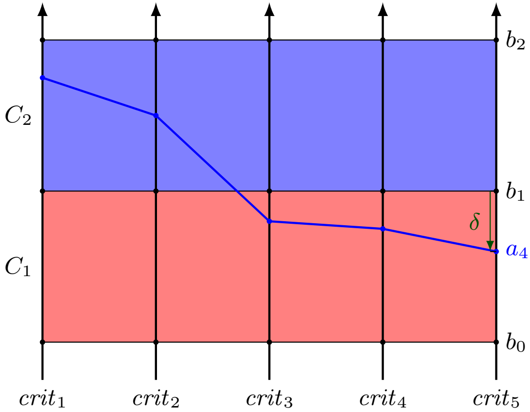
Learning the profiles

- $V_{h,j}^{+\delta}$ (resp. $V_{h,j}^{-\delta}$) : the sets of alternatives misclassified in C_{h+1} instead of C_h (resp. C_h instead of C_{h+1}), for which moving the profile b_h by $+\delta$ (resp. $-\delta$) on j results in a correct assignment.



Learning the profiles

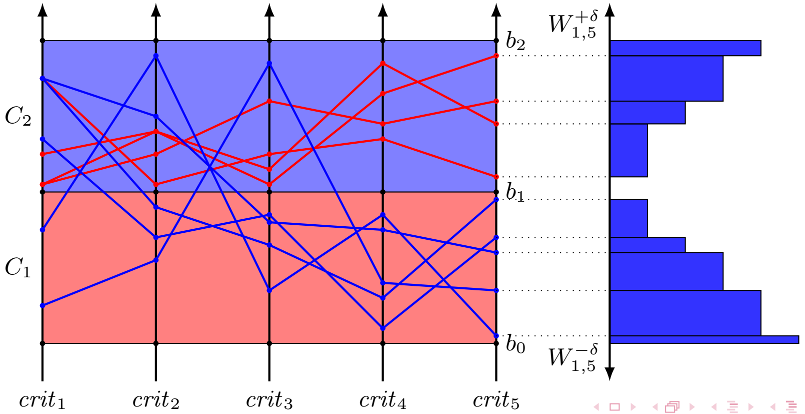
- $W_{h,j}^{+\delta}$ (resp. $W_{h,j}^{-\delta}$) : the sets of alternatives misclassified in C_{h+1} instead of C_h (resp. C_h instead of C_{h+1}), for which moving the profile b_h of $+\delta$ (resp. $-\delta$) on j strengthens the criteria coalition in favor of the correct classification but will not by itself result in a correct assignment.



- $C_2 \succ C_1$
- $w_j = 0.2$ for $j = 1, ..., 5$
- $\lambda = 0.8$
- $a_4 \in A_{2 \leftarrow \mathrm{DM}}^{1 \leftarrow \mathrm{Model}}$

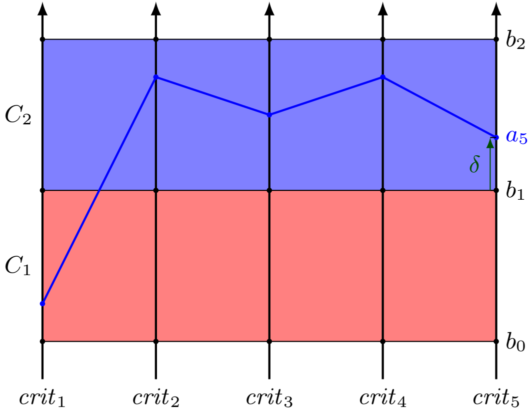
Learning the profiles

- $W_{h,j}^{+\delta}$ (resp. $W_{h,j}^{-\delta}$) : the sets of alternatives misclassified in C_{h+1} instead of C_h (resp. C_h instead of C_{h+1}), for which moving the profile b_h of $+\delta$ (resp. $-\delta$) on j strengthens the criteria coalition in favor of the correct classification but will not by itself result in a correct assignment.



Learning the profiles

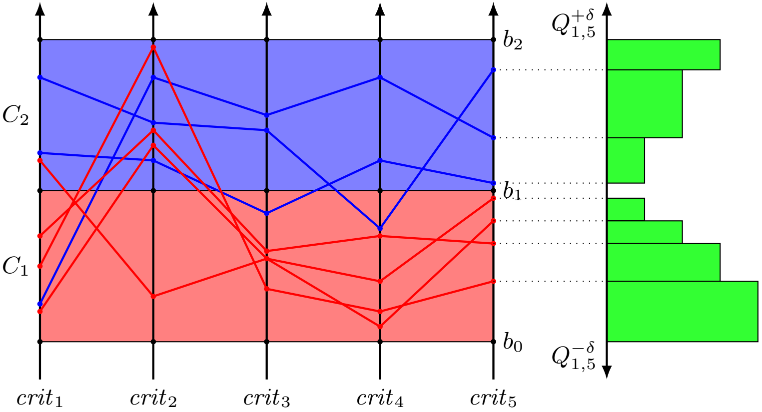
- $Q^{+\delta}_{h,j}$ (resp. $Q^{-\delta}_{h,j}$) : the sets of alternatives correctly classified in C_{h+1} (resp. C_h) for which moving the profile b_h of $+\delta$ (resp. $-\delta$) on j results in a misclassification.



- $C_2 \succ C_1$
- $w_j = 0.2$ for $j = 1, ..., 5$
- $\lambda = 0.8$
- $a_5 \in A^{2 \leftarrow \text{Model}}_{2 \leftarrow \text{DM}}$

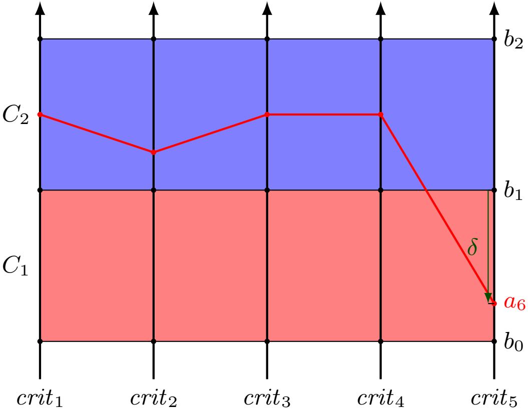
Learning the profiles

- $Q^{+\delta}_{h,j}$ (resp. $Q^{-\delta}_{h,j}$) : the sets of alternatives correctly classified in C_{h+1} (resp. C_h) for which moving the profile b_h of $+\delta$ (resp. $-\delta$) on j results in a misclassification.



Learning the profiles

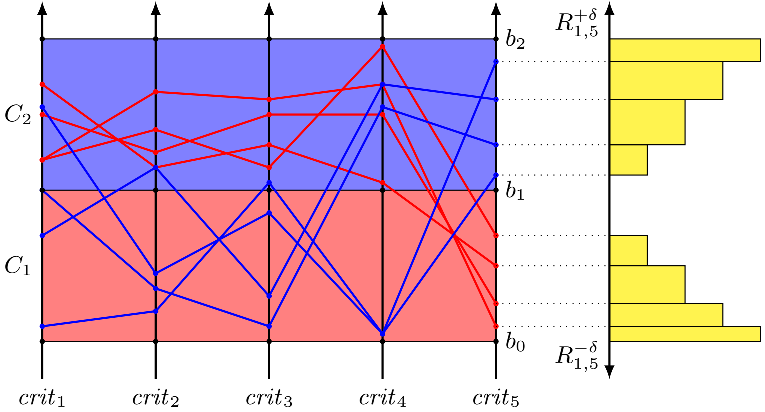
- $R^{+\delta}_{h,j}$ (resp. $R^{-\delta}_{h,j}$) : the sets of alternatives misclassified in C_{h+1} instead of C_h (resp. C_h instead of C_{h+1}), for which moving the profile b_h of $+\delta$ (resp. $-\delta$) on j weakens the criteria coalition in favor of the correct classification but does not induce misclassification by itself.



- $C_2 \succ C_1$
- $w_j = 0.2$ for $j = 1, ..., 5$
- $\lambda = 0.8$
- $a_6 \in A^{1 \leftarrow \text{Model}}_{2 \leftarrow \text{DM}}$

Learning the profiles

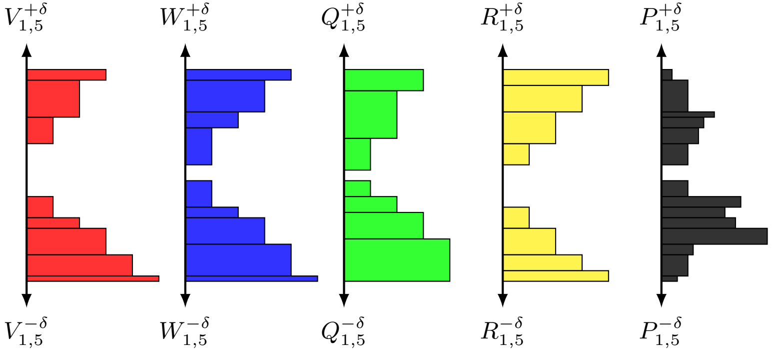
- $R_{h,j}^{+\delta}$ (resp. $R_{h,j}^{-\delta}$) : the sets of alternatives misclassified in C_{h+1} instead of C_h (resp. C_h instead of C_{h+1}), for which moving the profile b_h of $+\delta$ (resp. $-\delta$) on j weakens the criteria coalition in favor of the correct classification but does not induce misclassification by itself.



Learning the profiles

$$P(b_{1,j}^{+\delta}) = \frac{k_V|V_{1,j}^{+\delta}| + k_W|W_{1,j}^{+\delta}| + k_T|T_{1,j}^{+\delta}|}{d_V|V_{1,j}^{+\delta}| + d_W|W_{1,j}^{+\delta}| + d_T|T_{1,j}^{+\delta}| + d_Q|Q_{1,j}^{+\delta}| + d_R|R_{1,j}^{+\delta}|}$$

with : $k_V = 2, k_W = 1, k_T = 0.1, d_V = d_W = d_T = 1, d_Q = 5, d_R = 1$



Learning the profiles

perform K times

for all profile b_h **do**
  for all criterion j chosen randomly **do**
    Choose, in a randomized manner, a set of positions in the interval $[b_{h-1,j}, b_{h+1,j}]$
    Select the one such that $P(b_{h,j}^{\Delta})$ is maximal
    Draw uniformly a random number r from the interval $[0, 1]$.
    if $r \leq P(b_{h,j}^{\Delta})$ **then**
      Move $b_{h,j}$ to the position corresponding to $b_{h,j} + \Delta$
      Update the alternatives assignment
    end if
  end for
end for

Metaheuristic to learn all the parameters

Algorithm

Generate a population of N MR-Sort models
repeat
  for all MR-Sort model **do**
    Given the current profiles, learn optimal weights w_j and λ using LP,
    Using these w_j and λ, adjust profiles b_h with a heuristic
  end for
  Reinitialize the $\left\lfloor \frac{N}{2} \right\rfloor$ models with the worst CA
until Stopping criterion is met

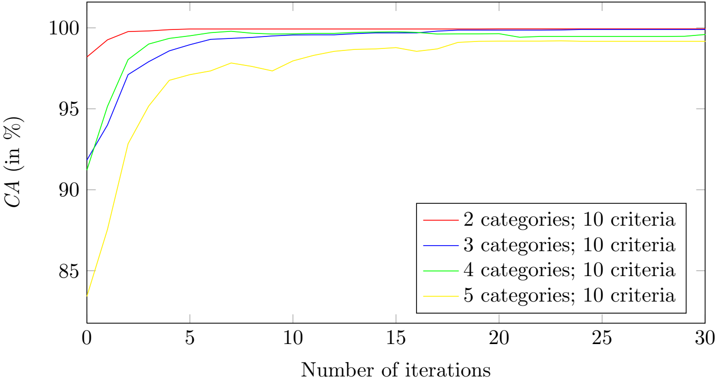
Stopping criterion

Stopping criterion : one of the models matches all examples or after max_{it} iterations.

Experimentations

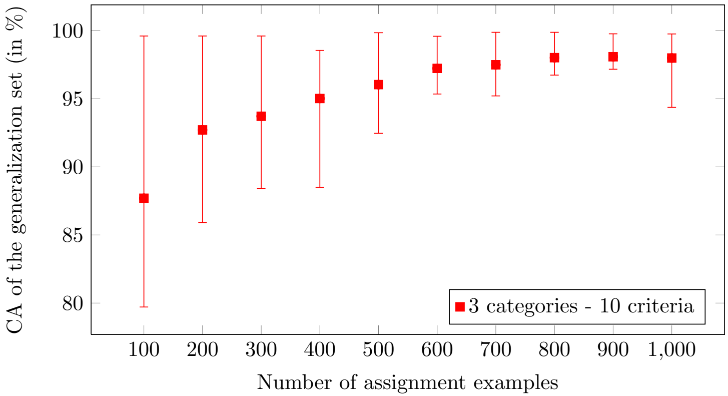
1. What's the efficiency of the algorithm ?
2. How much alternatives are required to learn a good model ?
3. What's the capability of the algorithm to restore assignment when there are errors in the examples ?
4. How the algorithm behaves on real datasets ?

Algorithm efficiency



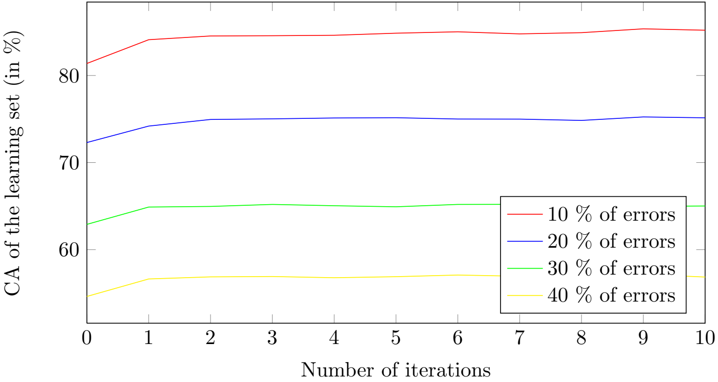
- Random model M generated
- Learning set : random alternatives assigned through the model M
- Model M' learned with the metaheuristic from the learning set

Model retrieval



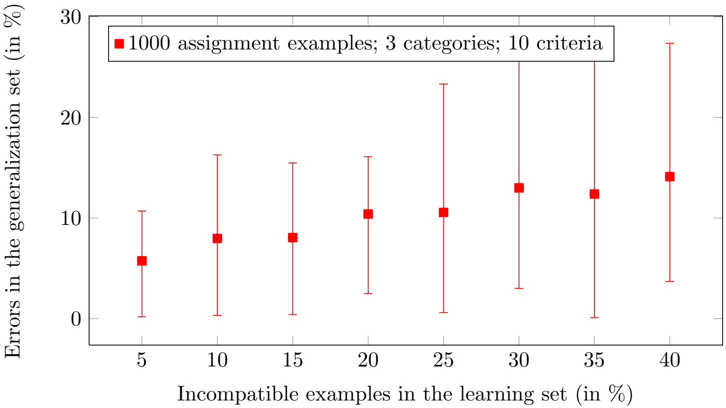
- Random model M generated
- Learning set : random alternatives assigned through model M
- Model M' learned with the metaheuristic from the learning set
- Generalization set : random alternatives assigned through M and M'

Tolerance for errors



- Random model M generated
- Learning set : random alternatives assigned through model M + errors
- Model M' learned with the metaheuristic from the learning set

Tolerance for errors



- Random model M generated
- Learning set : random alternatives assigned through model M + errors
- Model M' learned with the metaheuristic from the learning set
- Generalization set : random alternatives assigned through M and M'

Application on real datasets

Dataset	#instances	#attributes	#categories
DBS	120	8	2
CPU	209	6	4
BCC	286	7	2
MPG	392	7	36
ESL	488	4	9
MMG	961	5	2
ERA	1000	4	4
LEV	1000	4	5
CEV	1728	6	4

- Instances split in two parts : learning and generalization sets
- Binarization of the categories

Source : [?]

Application on real datasets - Binarized categories

Learning set	Dataset	MIP MR-SORT	META MR-SORT	LP UTADIS	CR
20 %	DBS	0.8023 ± 0.0481	0.8012 ± 0.0469	0.7992 ± 0.0533	0.8287 ± 0.0424
	CPU	0.9100 ± 0.0345	0.8960 ± 0.0433	0.9348 ± 0.0362	0.9189 ± 0.0103
	BCC	0.7322 ± 0.0276	0.7196 ± 0.0302	0.7085 ± 0.0307	0.7225 ± 0.0335
	MPG	0.7920 ± 0.0326	0.7855 ± 0.0383	0.7775 ± 0.0318	0.9291 ± 0.0193
	ESL	0.8925 ± 0.0158	0.8932 ± 0.0159	0.9111 ± 0.0160	0.9318 ± 0.0129
	MMG	0.8284 ± 0.0140	0.8235 ± 0.0135	0.8160 ± 0.0184	0.8275 ± 0.012
	ERA	0.7907 ± 0.0174	0.7915 ± 0.0146	0.7632 ± 0.0187	0.7111 ± 0.0273
	LEV	0.8386 ± 0.0151	0.8327 ± 0.0221	0.8346 ± 0.0160	0.8501 ± 0.0122
	CEV	-	0.9214 ± 0.0045	0.9206 ± 0.0059	0.9552 ± 0.0089
50 %	DBS	0.8373 ± 0.0426	0.8398 ± 0.0487	0.8520 ± 0.0421	0.8428 ± 0.0416
	CPU	0.9360 ± 0.0239	0.9269 ± 0.0311	0.9770 ± 0.0238	0.9536 ± 0.0281
	BCC	-	0.7246 ± 0.0446	0.7146 ± 0.0246	0.7313 ± 0.0282
	MPG	-	0.8170 ± 0.0295	0.7910 ± 0.0236	0.9423 ± 0.0251
	ESL	0.8982 ± 0.0155	0.8982 ± 0.0203	0.9217 ± 0.0163	0.9399 ± 0.0126
	MMG	-	0.8290 ± 0.0153	0.8242 ± 0.0152	0.8333 ± 0.0144
	ERA	0.8042 ± 0.0137	0.7951 ± 0.0191	0.7658 ± 0.0171	0.7156 ± 0.0306
	LEV	0.8554 ± 0.0151	0.8460 ± 0.0221	0.8444 ± 0.0132	0.8628 ± 0.0125
	CEV	-	0.9216 ± 0.0067	0.9201 ± 0.0091	0.9624 ± 0.0059
80 %	DBS	0.8520 ± 0.0811	0.8712 ± 0.0692	0.8720 ± 0.0501	0.8584 ± 0.0681
	CPU	0.9402 ± 0.0315	0.9476 ± 0.0363	0.9848 ± 0.0214	0.9788 ± 0.0301
	BCC	-	0.7486 ± 0.0640	0.7087 ± 0.0510	0.7504 ± 0.0485
	MPG	-	0.8152 ± 0.0540	0.7920 ± 0.0388	0.9449 ± 0.016
	ESL	0.8992 ± 0.0247	0.9017 ± 0.0276	0.9256 ± 0.0235	0.9458 ± 0.0218
	MMG	-	0.8313 ± 0.0271	0.8266 ± 0.0265	0.8416 ± 0.0251
	ERA	0.8144 ± 0.0260	0.7970 ± 0.0272	0.7644 ± 0.0292	0.7187 ± 0.028
	LEV	0.8628 ± 0.0232	0.8401 ± 0.0321	0.8428 ± 0.0222	0.8686 ± 0.0176
	CEV	-	0.9204 ± 0.0130	0.9201 ± 0.0132	0.9727 ± 0.1713

Application on real datasets

	Dataset	MIP MR-SORT	META MR-SORT	LP UTADIS
20 %	CPU	0.7542 ± 0.0506	0.7443 ± 0.0559	0.8679 ± 0.0488
	ERA	-	0.5104 ± 0.0198	0.4856 ± 0.0169
	LEV	-	0.5528 ± 0.0274	0.5775 ± 0.0175
	CEV	-	0.7761 ± 0.0183	0.7719 ± 0.0153
50 %	CPU	-	0.8052 ± 0.0361	0.9340 ± 0.0266
	ERA	-	0.5216 ± 0.0180	0.4833 ± 0.0171
	LEV	-	0.5751 ± 0.0230	0.5889 ± 0.0158
	CEV	-	0.7833 ± 0.0180	0.7714 ± 0.0158
80 %	CPU	-	0.8055 ± 0.0560	0.9512 ± 0.0351
	ERA	-	0.5230 ± 0.0335	0.4824 ± 0.0332
	LEV	-	0.5750 ± 0.0344	0.5933 ± 0.0305
	CEV	-	0.7895 ± 0.0203	0.7717 ± 0.0259

Conclusions and further research

- attempt at bringing MCDA methods into the context of preference learning
- Algorithm able to handle large datasets
- Web service available to test (Decision Deck) `www.decision-deck.org`

- Integrate veotoes into MR-Sort models
- Learning reference based ranking model [Rolland 2013]
- Test the algorithm on other real datasets


That's all Folks!

References I

- Bouyssou, D. and Marchant, T. (2007a).
  An axiomatic approach to noncompensatory sorting methods in MCDM, I : The case of two categories.
  European Journal of Operational Research, 178(1) :217–245.
- Bouyssou, D. and Marchant, T. (2007b).
  An axiomatic approach to noncompensatory sorting methods in MCDM, II : More than two categories.
  European Journal of Operational Research, 178(1) :246–276.
- Doumpos, M., Marinakis, Y., Marinaki, M., and Zopounidis, C. (2009).
  An evolutionary approach to construction of outranking models for multicriteria classification : The case of the ELECTRE TRI method.
  European Journal of Operational Research, 199(2) :496–505.

References II

Leroy, A., Mousseau, V., and Pirlot, M. (2011).
Learning the parameters of a multiple criteria sorting method.
In Brafman, R., Roberts, F., and Tsoukiàs, A., editors, *Algorithmic Decision Theory*, volume 6992 of *Lecture Notes in Computer Science*, pages 219–233. Springer Berlin / Heidelberg.

Slowínski, R., Greco, S., and Matarazzo, B. (2002).
Axiomatization of utility, outranking and decision-rule preference models for multiple-criteria classification problems under partial inconsistency with the dominance principle.
Control and Cybernetics, 31(4) :1005–1035.

Tehrani, A. F., Cheng, W., Dembczynski, K., and Hüllermeier, E. (2012).
Learning monotone nonlinear models using the choquet integral.
Machine Learning, 89(1-2) :183–211.

References III

Yu, W. (1992).
Aide multicritère à la décision dans le cadre de la problématique du tri : méthodes et applications.
PhD thesis, LAMSADE, Université Paris Dauphine, Paris.