Learning MR-Sort rules with coalitional veto

Olivier Sobrie[1,2,3] **, Vincent Mousseau**[2] and **Marc Pirlot**[3]

Abstract. MR-Sort (Majority Rule Sorting) is a multiple criteria sorting method which assigns an alternative a to category C^h when a is better than the lower limit of C^h on a majority of criteria, and this is not true with the upper limit of C^h. We enrich the descriptive ability of MR-Sort by the addition of coalitional vetoes which operate in a symmetric way as compared to the MR-Sort rule w.r.t. to category limits, using specific veto profiles and veto weights. We describe a heuristic algorithm to learn such an MR-Sort model enriched with coalitional veto from a set of assignment examples, and show how it performs on real datasets.

1 Introduction

Multiple Criteria Sorting Problems aim at assigning alternatives to one of the predefined ordered categories $C^1, C^2, ..., C^p$, C^1 and C^p being the worst and the best category, respectively. Many multiple criteria sorting methods have been proposed in the literature (see e.g., [9], [25]). MR-Sort (Majority Rule Sorting, see [10]) is an outranking-based multiple criteria sorting method which corresponds to a simplified version of ELECTRE TRI where the discrimination and veto thresholds are omitted.

In the pessimistic version of ELECTRE TRI, veto effects make it possible to worsen the category to which an alternative is assigned when this alternative has very bad performances on one/several criteria. We consider a variant of MR-Sort which introduces possible veto effects. While in ELECTRE TRI, a veto involves a single criterion, we consider a more general formulation of veto (see [21]) which can involve a coalition of criteria (such a coalition can be reduced to a singleton).

The definition of such a "*coalitional veto*" exhibits a noteworthy symmetry between veto and concordance. To put it simple, in a two-category context (*Bad*/*Good*), an alternative is classified as *Good* when its performances are above the concordance profile on a sufficient majority of criteria, and when its performances are not below the veto profile for a sufficient majority of criteria. Hence, the veto condition can be viewed as the negation of a majority rule using a specific veto profile, and specific veto weights.

Algorithms to learn the parameters of an MR-Sort model without veto (category limits and criteria weights) have been proposed, either using linear programming involving integer variables (see [10]) or using a specific heuristic (see [20, 19]). When the size of the learning set exceeds 100, only heuristic algorithms are able to provide a solution within a limited computing time.

Olteanu and Meyer [14] have developed a simulated annealing based algorithm to learn a MR-Sort model with classical veto (not coalitional ones).

In this paper, we propose a new heuristic algorithm to learn the parameters of a MR-Sort model with coalitional veto (called MR-Sort-CV) which makes use of the symmetry between the concordance and the coalitional veto conditions. We describe preliminary results obtained by experimenting with this algorithm on real data sets.

The paper is organized as follows. In Section 2, we recall MR-Sort and define its extension when considering monocriterion veto and coalitional veto. After a brief reminder of the heuristic algorithm to learn an MR-Sort model, Section 3 is devoted to the presentation of the algorithm to learn an MR-Sort model with coalitional veto. Section 4 presents experimentations of this algorithm and Section 5 groups conclusions and directions for further research.

2 Considering vetoes in MR-Sort

2.1 MR-Sort model

MR-Sort is a method for assigning objects to ordered categories. It is a simplified version of ELECTRE TRI, another MCDA method [23, 16].

The MR-Sort rule works as follows. Formally, let X be a set of objects evaluated on n ordered attributes (or criteria), $F = \{1, ..., n\}$. We assume that X is the Cartesian product of the criteria scales, $X = \prod_{j=1}^{n} X_j$, each scale X_j being completely ordered by the relation $\geq_j$. An object $a \in X$ is a vector $(a_1, \ldots, a_j, \ldots, a_n)$, where $a_j \in X_j$ for all j. The ordered categories which the objects are assigned to by the MR-Sort model are denoted by C^h, with $h = 1, \ldots, p$. Category C^h is delimited by its lower limit profile b^{h-1} and its upper limit profile b^h, which is also the lower limit profile of category C^{h+1} (provided $0 < h < p$). The profile b^h is the vector of criterion values $(b_1^h, \ldots, b_j^h, \ldots, b_n^h)$, with $b_j^h \in X_j$ for all j. We denote by $P = \{1, ..., p\}$ the list of category indices. By convention, the best category, C^p, is delimited by a fictive upper profile, b^p, and the worst one, C^1, by a fictive lower profile, b^0. It is assumed that the profiles dominate one another, i.e.: $b_j^h \geq_j b_j^{h-1}$, for $h = \{1, \ldots, p\}$ and $j = \{1, \ldots, n\}$.

Using the MR-Sort procedure, an object is assigned to a category if its criterion values are at least as good as the category lower profile values on a weighted majority of criteria while this condition is not fulfilled when the object's criterion

[1] email: olivier.sobrie@gmail.com
[2] CentraleSupélec, Université Paris-Saclay, Grande Voie des Vignes, 92295 Châtenay-Malabry, France, email: vincent.mousseau@centralesupelec.fr
[3] Université de Mons, Faculté Polytechnique, 9, rue de Houdain, 7000 Mons, Belgium, email: marc.pirlot@umons.ac.be

values are compared to the category upper profile values. In the former case, we say that the object is *preferred* to the profile, while, in the latter, it is not. Formally, if an object $a \in X$ is *preferred* to a profile b^h, we denote this by $a \succcurlyeq b^h$. Object a is preferred to profile b^h whenever the following condition is met:

$$a \succcurlyeq b^h \Leftrightarrow \sum_{j:a_j \geq_j b_j^h} w_j \geq \lambda, \tag{1}$$

where w_j is the nonnegative weight associated with criterion j, for all j and λ sets a majority level. The weights satisfy the normalization condition $\sum_{j \in F} w_j = 1$; λ is called the *majority threshold*.

The preference relation $\succcurlyeq$ defined by (1) is called an *outranking* relation without veto or a *concordance* relation ([16]; see also [3, 4] for an axiomatic description of such relations). Consequently, the condition for an object $a \in X$ to be assigned to category C^h reads:

$$\sum_{j:a_j \geq_j b_j^{h-1}} w_j \geq \lambda \quad \text{and} \quad \sum_{j:a_j \geq_j b_j^h} w_j < \lambda. \tag{2}$$

The MR-Sort assignment rule described above involves $pn+1$ parameters, i.e. n weights, $(p-1)n$ profiles evaluations and one majority threshold.

A *learning set* A is a subset of objects $A \subseteq X$ for which an assignment is known. For $h \in P$, A_h denotes the subset of objects $a \in A$ which are assigned to category C^h. The subsets A_h are disjoint; some of them may be empty.

2.2 MR-Sort-MV

In this section, we recall the traditional monocriterion veto rule as defined by [1, 2]. In a MR-Sort model with monocriterion veto, an alternative a is "at least as good as" a profile b^h if it has at least equal to or better performances than b^h on a weighted majority of criteria and if it is not strongly worse than the profile on any criterion. In the sequel, we call b^h a *concordance profile* and we define "strongly worse than the profile" b^h by means of a veto profile $v^h = (v_1^h, v_2^h, ..., v_n^h)$, with $v_j^h \leq_j b_j^h$. It represents a vector of performances such that any alternative having a performance worse than or equal to this profile on any criterion would be excluded from category C^{h+1}. Formally, the assignment rule is described by the following condition:

$$a \succcurlyeq b^h \iff \sum_{j:a_j \geq_j b_j^h} w_j \geq \lambda \text{ and not } a\,V b^h,$$

with $a\,V b^h \iff \exists j \in F : a_j \leq_j v_j^h$. Note that non-veto condition is frequently presented in the literature using a veto threshold (see e.g. [15]), i.e. a maximal difference w.r.t. the concordance profile in order to be assigned to the category above the profile. Using veto profiles instead of veto thresholds better suits the context of multicriteria sorting. We recall that a profile b^h delimits the category C^h from C^{h+1}, with $C^{h+1} \succ C^h$; with monocriterion veto, the MR-Sort assignment rule reads as follows:

$$a \in C^h \iff \left[\sum_{j:a_j \geq_j b_j^{h-1}} w_j \geq \lambda \text{ and } \nexists j \in F : a_j < v_j^{h-1} \right]$$
$$\text{and} \left[\sum_{j:a_j \geq_j b_j^h} w_j < \lambda \text{ or } \exists j \in F : a_j \leq v_j^h \right]. \tag{3}$$

We remark that a MR-Sort model with more than 2 categories remains consistent only if veto profiles v_j^h do not overlap, i.e, are chosen such that $v_j^h \geq v_j^{h'}$ for all $\{h, h'\}$ s.t. $h > h'$. Otherwise, an alternative might be on the one hand in veto against a profile b^h, which prevents it to be assigned to C^{h+1} and, on the other hand, not in veto against b^{h+1}, which does not prevent it to be assigned to C^{h+2}.

2.3 MR-Sort-CV

We introduce here a new veto rule considering vetoes w.r.t. coalitions of criteria, which we call "*coalitional veto*". With this rule, the veto applies and forbids an alternative a to be assigned to category C^{h+1} when the performance of an alternative a is not better than v_j^h on a weighted majority of criteria.

As for the monocriterion veto, the veto profiles are vectors of performances $v^h = (v_1^h, v_2^h, ..., v_n^h)$, for all $h = \{1, .., p\}$. Coalitional veto also involves a set of *veto weights* denoted z_j, for all $j \in F$. Without loss of generality, the sum of z_j is set to 1. Furthermore, a veto cutting threshold Λ is also involved and determines whether a coalition of criteria is sufficient to impose a veto. Formally, we express the coalitional veto rule $a\,V b^h$, as follows:

$$a\,V b^h \iff \sum_{j:a_j \leq_j v_j^h} z_j \geq \Lambda. \tag{4}$$

Using coalitional veto, the outranking relation of MR-Sort (2.2) is modified as follows:

$$a \succcurlyeq b^h \iff \sum_{j:a_j \geq_j b_j^h} w_j \geq \lambda \text{ and } \sum_{j:a_j \leq_j v_j^h} z_j < \Lambda. \tag{5}$$

Using coalitional veto with MR-Sort modifies the assignment rule as follows:

$$a \in C^h \iff \left[\sum_{j:a_j \geq_j b_j^{h-1}} w_j \geq \lambda \text{ and } \sum_{j:a_j \leq_j v_j^{h-1}} z_j < \Lambda \right]$$
$$\text{and} \left[\sum_{j:a_j \geq_j b_j^h} w_j < \lambda \text{ or } \sum_{j:a_j \leq_j v_j^h} z_j \geq \Lambda \right] \tag{6}$$

In MR-Sort, the coalitional veto can be interpreted as a combination of performances preventing the assignment of an alternative to a category. We call this new model, MR-Sort-CV.

The coalitional veto rule given in Equation (5) is a generalization of the monocriterion rule. Indeed, if the veto cut threshold Λ is equal to $\frac{1}{n}$ (n being the number of criteria), and each veto weight z_j is set to $\frac{1}{n}$, then the veto rule defined in Equation (4) corresponds to a monocriterion veto for each criterion.

2.4 The Non Compensatory Sorting (NCS) model

In this subsection, we recall the non compensatory sorting (NCS) rule as defined by [1, 2], which will be used in the experimental part (Section 4) for comparison purposes. These rules allow to model criteria interactions. MR-Sort is a particular case of these, in which criteria do not interact.

In order to take criteria interactions into account, it has been proposed to modify the definition of the global outranking relation, $a \succcurlyeq b^h$, given in (1). We introduce the notion of capacity. A *capacity* is a function $\mu : 2^F \to [0,1]$ such that:

- $\mu(B) \geq \mu(A)$, for all $A \subseteq B \subseteq F$ (monotonicity) ;
- $\mu(\emptyset) = 0$ and $\mu(F) = 1$ (normalization).

The Möbius transform allows to express the capacity in another form:

$$\mu(A) = \sum_{B \subseteq A} m(B), \tag{7}$$

for all $A \subseteq F$, with $m(B)$ defined as:

$$m(B) = \sum_{C \subseteq B} (-1)^{|B|-|C|} \mu(C) \tag{8}$$

The value $m(B)$ can be interpreted as the weight that is exclusively allocated to B as a whole. A capacity can be defined directly by its Möbius transform also called "interaction". An interaction m is a set function $m : 2^F \to [-1,1]$ satisfying the following conditions:

$$\sum_{j \in K \subseteq J \cup \{j\}} m(K) \geq 0, \quad \forall j \in F, J \subseteq F \backslash \{i\} \tag{9}$$

and

$$\sum_{K \subseteq F} m(K) = 1.$$

If m is an interaction, the set function defined by $\mu(A) = \sum_{B \subseteq A} m(B)$ is a capacity. Conditions (9) guarantee that μ is monotone [6].

Using a capacity to express the weight of the coalition in favor of an object, we transform the outranking rule as follows:

$$a \succcurlyeq b^h \Leftrightarrow \mu(A) \geq \lambda \text{ with } A = \{j : a_j \geq_j b_j^h\}$$
$$\text{and } \mu(A) = \sum_{B \subseteq A} m(B) \tag{10}$$

Computing the value of $\mu(A)$ with the Möbius transform induces the evaluation of $2^{|A|}$ parameters. In a model composed of n criteria, it implies the elicitation of 2^n parameters, with $\mu(\emptyset) = 0$ and $\mu(F) = 1$. To reduce the number of parameters to elicit, we use a 2-additive capacity in which all the interactions involving more than 2 criteria are equal to zero. Inferring a 2-additive capacity for a model having n criteria requires the determination of $\frac{n(n+1)}{2} - 1$ parameters.

Finally, the condition for an object $a \in X$ to be assigned to category C^h can be expressed as follows:

$$\mu(F_{a,h-1}) \geq \lambda \quad \text{and} \quad \mu(F_{a,h}) < \lambda \tag{11}$$

with $F_{a,h-1} = \{j : a_j \geq_j b_j^{h-1}\}$ and $F_{a,h} = \{j : a_j \geq_j b_j^h\}$.

3 Learning MR-Sort

Learning the parameters of MR-Sort and ELECTRE TRI models has been already studied in several articles [10, 17, 12, 11, 13, 7, 8, 5, 24]. In this section, we recall how to learn the parameters of an MR-Sort model using respectively an exact method [10] and a heuristic algorithm [17]. We then extend the heuristic algorithm to MR-Sort-CV.

3.1 Learning a simple MR-Sort

It is possible to learn a MR-Sort model from a learning set using Mixed Integer Programming (MIP), see [10]. Such a MIP formulation is not suitable for large data sets because of the high computing time required to infer the MR-Sort parameters. In view of learning MR-Sort models in the context of large data sets, a heuristic algorithm has been proposed in [17]. As for the MIP, the heuristic algorithm takes as input a set of assignment examples and their vectors of performances. The algorithm returns the parameters of a MR-Sort model.

The heuristic algorithm proposed in [17] works as follows. First a population of N_{mod} MR-Sort models is initialized. Thereafter, the following two steps are repeated iteratively on each model in the population:

1. A linear program optimizes the weights and the majority threshold on the basis of assignment examples and fixed profiles.
2. Given the inferred weights and the majority threshold, a heuristic adjusts the profiles of the model on the basis of the assignment examples.

After applying these two steps to all the models in the population, the $\left\lfloor \frac{N_{\text{mod}}}{2} \right\rfloor$ models restoring the least numbers of examples are reinitialized. These steps are repeated until the heuristic finds a model that fully restores all the examples or after a number of iterations specified a priori.

The linear program designed to learn the weights and the majority threshold is given by (12). It minimizes a sum of slack variables, x'_a and y'_a, that is equal to 0 when all the objects are correctly assigned, i.e. assigned to the category defined in the input data set. We remark that the objective function of the linear program does not explicitly minimize the 0/1 loss but a sum of slacks. This implies that compensatory effects might appear, with undesirable consequences on the 0/1 loss. However in this heuristic, we consider that these effects are acceptable. The linear program doesn't involve binary variables. Therefore, the computing time remains reasonable when the size of the problem increases.

The objective function of the heuristic varying the profiles maximizes the number of examples compatible with the model. To do so, it iterates over each profile b^h and each criterion j and identifies a set of candidate moves for the profile, which correspond to the performances of the examples on criterion j located between profiles b^{h-1} and b^{h+1}. Each candidate move is evaluated as a function of the probability to improve the classification accuracy of the model. To evaluate if a candidate move is likely or unlikely to improve the classification of one or several objects, the examples which have an evaluation on criterion j located between the current value of the profile, b_j^h, and the candidate move, $b_j^h + \delta$ (resp. $b_j^h - \delta$), are grouped in different subsets:

$$
\begin{array}{rcll}
\multicolumn{4}{l}{\min \sum_{a\in A}(x'_a + y'_a)} \\
\multicolumn{4}{l}{\text{s.t.}} \\
\sum_{j:a_j \geq_j b_j^{h-1}} w_j - x_a + x'_a & = & \lambda & \forall a \in A_h, h = \{2, ..., p\} \\
\sum_{j:a_j \geq_j b_j^{h}} w_j + y_a - y'_a & = & \lambda - \epsilon & \forall a \in A_h, h = \{1, ..., p-1\} \\
\sum_{j=1}^{n} w_j & = & 1 & \\
w_j & \in & [0; 1] & \forall j \in F \\
\lambda & \in & [0; 1] & \\
x_a, y_a, x'_a, y'_a & \in & \mathbb{R}_0^+ & \\
\varepsilon & & \text{a small positive number.} &
\end{array}
\quad (12)
$$

$V_{h,j}^{+\delta}$ **(resp.** $V_{h,j}^{-\delta}$**)** : the sets of objects 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.

$W_{h,j}^{+\delta}$ **(resp.** $W_{h,j}^{-\delta}$**)** : the sets of objects 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 strengthens the criteria coalition in favor of the correct classification but will not by itself result in a correct assignment.

$Q_{h,j}^{+\delta}$ **(resp.** $Q_{h,j}^{-\delta}$**)** : the sets of objects correctly classified in C^{h+1} (resp. C^{h+1}) for which moving the profile b^h by $+\delta$ (resp. $-\delta$) on j results in a misclassification.

$R_{h,j}^{+\delta}$ **(resp.** $R_{h,j}^{-\delta}$**)** : the sets of objects 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 weakens the criteria coalition in favor of the correct classification but does not induce misclassification by itself.

$T_{h,j}^{+\delta}$ **(resp.** $T_{h,j}^{-\delta}$**)** : the sets of objects misclassified in a category higher than C^h (resp. in a category lower than C^{h+1}) for which the current profile evaluation weakens the criteria coalition in favor of the correct classification.

A formal definition of these sets can be found in [17]. The evaluation of the candidate moves is done by aggregating the number of elements in each subset. Finally, the choice to move or not the profile on the criterion is determined by comparing the candidate move evaluation to a random number drawn uniformly. These operations are repeated multiple times on each profile and each criterion.

3.2 Learning MR-Sort-CV

In (2), the MR-Sort condition $\sum_{j:a_j \geq_j b_j^{h-1}} w_j \geq \lambda$ is a necessary condition for an alternative to be assigned to a category at least as good as C^h. Basically a coalitional veto rule can be viewed as a dual version of the majority rule. It provides a sufficient condition for being assigned to a category worse than C^h. An alternative will be assigned to such a category as soon as $\sum_{j:a_j \leq_j v_j^{h-1}} z_j \geq \Lambda$. This condition has essentially the same form as the MR-Sort rule except that the sum is over the criteria on which the alternative's performance is at most as good as the profile (instead of at least as good, in the MR-Sort rule). Therefore, a straightforward way of implementing an algorithm to learn a MR-Sort-CV model is by using the MR-Sort learning heuristic twice, the second time, looking at each criterion in the reversed order of preference.

In the first step, we learn concordance profiles b^h, a weight vector w and a threshold λ using the MR-Sort learning heuristic [18]. We tune the parameters of this algorithm in order to penalize more the false negative than the false positive assignments. In a second step, we apply essentially the same algorithm to learn veto profiles v^h, a weight vector z and a threshold Λ. The direction of optimization is reversed on each criterion, the veto profiles are constrained to lie below their corresponding concordance profile (i.e. $v_j^h \leq_j b_j^h$, for all j and h), which was determined in the first step. In the second step, the learning algorithm is applied to all assignment examples.

We now give more detail on the way we tuned the parameters of the algorithm used in the first step. Let us call the model obtained in the first step, for category C^h, the concordance rule, and the model in the second step, the veto rule. The main point is that the false positive and the false negative assignments produced by the concordance rule are not treated equally. False positive assignments can be corrected by the veto rule, while false negatives cannot. Moreover, the second step leading to a veto rule will have little impact on classification accuracy in case the proportion of false positives is small in the set of wrongly assigned alternatives. For these reasons, we had to penalize more severely false negatives than false positives in the first step. There are basically three simple actions on the algorithm's parameter that can result in favoring false positive assignments.

1. Model selection process. After having iterated the two steps of the algorithm (weights optimization and profiles adjustment) described in Section 3.1, $[N_{\text{mod}}/2]$ models are discarded and replaced. This is done, in the MR-Sort learning algorithm, by selecting the models that make the more assignment errors. We adapt this selection criterion by adding 0.3 times the number of false positive assignments to the total number of correct assignments. The discarded models are thus those for which the number of true positive plus the number of true negative plus 0.3 times the number of false positive is below the median of that quantity on the models' population.
2. Weights optimization. The concordance profiles being given, the weights are optimized using the linear program (12). The sum of the error variables $x'_a + y'_a$ was the objective to be minimized. In the linear program, x'_a is set to a positive value whenever it is not possible to satisfy the condition which assigns a to a category at least as good as C^h, while a actually belongs to C^h. Impeding the assignment of positive values to x'_a amounts to favor false positive assignments. Hence, positive values of x'_a should be heavily penalized. In contrast, positive values of y'_a correspond to the case in which the conditions for assigning a to the categories above the profile are met while a belongs to the category below the profile. Positive values of y'_a need not be discouraged as much as those of x'_a and therefore we

changed the objective function of the linear program into $\min \sum_{a \in A} 10x'_a + y'_a$.

3. Adjustment of profile. In order to select moves in the profile level on a criterion by a quantity $\pm\delta$, we compute a probability which takes into account the sizes of the sets listed at the end of section 3.1. In all cases, the movements which lower the profile $(-\delta)$ are more favorable to false positive than the opposite movements. Therefore, all other things being equal (i.e. the sizes of the sets), the probability of choosing a downward move $-\delta$ should be larger than that of an upward move $+\delta$. The probability of an upward move is thus computed by the following formula

$$P(b_j^h + \delta) = \frac{2|V_{h,j}^{-\delta}| + 1|W_{h,j}^{-\delta} + 0.1|T_{h,j}^{-\delta}|}{|V_{h,j}^{-\delta}| + |W_{h,j}^{-\delta}| + |T_{h,j}^{-\delta}| + 5|Q_{h,j}^{-\delta}| + |R_{h,j}^{-\delta}|}.$$

while that of a downward move is

$$P(b_j^h - \delta) = \frac{4|V_{h,j}^{+\delta}| + 2|W_{h,j}^{+\delta} + 0.1|T_{h,j}^{+\delta}|}{|V_{h,j}^{+\delta}| + |W_{h,j}^{+\delta}| + |T_{h,j}^{+\delta}| + 5|Q_{h,j}^{+\delta}| + |R_{h,j}^{+\delta}|}$$

Remarks The learning algorithm described above is a preliminary version. Many different strategies for learning appropriate concordance and veto profiles as well as their associated weight vectors and thresholds are possible, even with the present option consisting of using successively two variants of the MR-Sort heuristic. In order to assess the latter idea, we report the results of both the first step (concordance part) and the second step (adding the veto) in the experiments done in Section 4. The results of the first part are reported as MR-Sort-FP.

4 Experiments

4.1 Datasets

In view of assessing the performance of the heuristic algorithm designed for learning the parameters of a MR-Sort-CV model, we use it to learn MR-Sort-CV models from several real data sets available at `http://www.uni-marburg.de/fb12/kebi/research/repository/monodata`, which serve as benchmarks to assess monotone classification algorithms [22]. They involve from 120 to 1728 instances, from 4 to 8 monotone attributes and from 2 to 36 categories. In our experiments, categories have been binarized by thresholding at the median. We split the datasets in a twofold 50/50 partition: a learning set and a test set. Models are learned on the first set and evaluated on the test set; this is done 100 times on learning sets drawn at random.

Data set	#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

Table 1: Data sets

4.2 Results obtained with MR-Sort and NCS

A similar experimental study [20] compares the results obtained with MR-Sort and NCS. The classification accuracy of both methods are provided in Table 2. No significant improvement in classification accuracy was observed when comparing NCS to MR-Sort.

Data set	Heuristic MR-Sort	Heuristic NCS
DBS	0.8377 ± 0.0469	0.8312 ± 0.0502
CPU	0.9325 ± 0.0237	0.9313 ± 0.0272
BCC	0.7250 ± 0.0379	0.7328 ± 0.0345
MPG	0.8219 ± 0.0237	0.8180 ± 0.0247
ESL	0.8996 ± 0.0185	0.8970 ± 0.0173
MMG	0.8268 ± 0.0151	0.8335 ± 0.0138
ERA	0.7944 ± 0.0173	0.7944 ± 0.0156
LEV	0.8408 ± 0.0122	0.8508 ± 0.0188
CEV	0.8516 ± 0.0091	0.8662 ± 0.0095

Table 2: Average and standard deviation of the classification accuracy on the datasets

4.3 Comparing MR-Sort-CV to MR-Sort

In this section, we investigate empirically the benefit obtained by adding "coalitional veto" to MR-Sort, i.e, we compare MR-Sort-CV to MR-Sort. Table 3 provides, for each binarized dataset, the confusion matrices; values provided are mean values of the proportion of alternatives. C^1 and C^2 are the true classes in the dataset, and $\hat{C}^1$ and $\hat{C}^2$ are the computed classifications. The first confusion table contains the proportions obtained with the MR-Sort heuristic. The second confusion table contains the proportions obtained with the MR-Sort-FP heuristic which favors false positives. Finally, the last confusion table contains the proportions obtained with MR-Sort-CV, i.e, when coalitional veto is added to MR-Sort-FP.

These first results show that MR-Sort and MR-Sort-CV provide similar results in terms of classification accuracy and that no benefit is induced from the introduction of coalitional veto. However, it should be noted that MR-Sort-FP obtains only a limited proportion of false positives ($C^2 - \hat{C}^1$). As coalitional veto is able to "correct" alternatives which are overclassified by MR-Sort-FP, we performed a second set of experiments in order to increase the proportion of false positives obtained by MR-Sort-FP (to do so we modify the model selection criterion so that the discarded models are those for which the number of true positive plus 0.9 times the number of false positives plus 0.1 times the number of true negatives is below the median of that quantity on the models' population). The results are provided in Table 4.

These results show that MR-Sort-FP provided higher proportions of false positives, even if this results in a lower overall classification accuracy. MR-Sort-CV is able to significantly improve these results, but the values of classification accuracy for MR-Sort-CV are still, after these changes, similar to the ones of MR-Sort.

These results tend to show that there is no significative improvement in classification accuracy when comparing the results of the standard MR-Sort to the results obtained with MR-Sort-CV. Although MR-Sort-CV is formally a generalization of MR-Sort which brings additional descriptive ability, the experiments fail to show an improvement on the ability of MR-Sort-CV to classify the benchmark datasets better than MR-Sort.

These results seem to us preliminary as it is not straightforward to state whether the inability to improve the result by the addition of coalitional veto comes from an insufficient performance of the algorithm, or from the limited additional descriptive ability induced by the introduction of coalitional veto to MR-Sort. Further analysis should be conducted.

Dataset		MR-Sort		MR-Sort-FP		MR-Sort-CV	
		$\hat{C}^1$	$\hat{C}^2$	$\hat{C}^1$	$\hat{C}^2$	$\hat{C}^1$	$\hat{C}^2$
DBS	C^1	42.1	8.5	40.5	10.2	41.8	8.8
	C^2	7.7	41.7	6.0	43.3	7.2	42.2
CPU	C^1	46.7	2.6	46.0	3.2	46.6	2.7
	C^2	4.2	46.6	2.9	47.9	3.3	47.4
BCC	C^1	60.5	10.3	58.1	12.7	63.2	7.6
	C^2	17.2	12.0	17.2	11.9	19.9	9.4
MPG	C^1	44.3	9.2	41.9	11.5	44.5	8.9
	C^2	8.6	37.9	7.9	38.7	9.1	37.4
ESL	C^1	48.6	6.0	46.9	7.6	49.3	5.2
	C^2	4.1	41.4	2.5	42.9	3.8	41.6
MMG	C^1	43.8	7.7	42.4	9.2	44.0	7.5
	C^2	9.6	38.8	8.4	40.0	9.8	38.7
ERA	C^1	69.3	5.1	68.0	6.4	71.8	2.5
	C^2	15.5	10.2	16.9	8.7	18.4	7.3
LEV	C^1	71.1	6.6	69.4	8.4	72.4	5.3
	C^2	9.3	12.9	9.2	13.1	10.9	11.4
CEV	C^1	59.2	10.9	59.2	10.8	61.5	8.5
	C^2	4.1	25.9	4.0	26.0	5.1	24.9

Table 3: Confusion matrices on the datasets for MR-Sort, MR-Sort-FP, and MR-Sort-CV

Dataset		MR-Sort		MR-Sort-FP		MR-Sort-CV	
		$\hat{C}^1$	$\hat{C}^2$	$\hat{C}^1$	$\hat{C}^2$	$\hat{C}^1$	$\hat{C}^2$
DBS	C^1	42.1	8.5	35.4	15.2	39.5	11.2
	C^2	7.7	41.7	2.8	46.6	5.3	44.0
CPU	C^1	46.7	2.6	37.7	11.5	43.9	5.3
	C^2	4.2	46.6	1.4	49.4	4.8	46.0
BCC	C^1	60.5	10.3	6.6	64.3	59.5	11.3
	C^2	17.2	12.0	1.1	28.1	18.8	11.2
MPG	C^1	44.3	9.2	7.7	45.8	44.7	8.8
	C^2	8.6	37.9	0.8	45.8	13.2	33.3
ESL	C^1	48.6	6.0	31.1	23.4	38.7	15.9
	C^2	4.1	41.4	0.7	44.7	3.2	42.3
MMG	C^1	43.8	7.7	8.9	42.6	38.2	13.3
	C^2	9.6	38.8	1.0	47.5	7.8	40.7
ERA	C^1	69.3	5.1	22.2	52.1	56.8	17.5
	C^2	15.5	10.2	3.2	22.5	13.1	12.6
LEV	C^1	71.1	6.6	40.6	37.1	68.2	9.5
	C^2	9.3	12.9	2.3	20.0	13.0	9.3
CEV	C^1	59.2	10.9	56.6	13.5	59.9	10.1
	C^2	4.1	25.9	3.0	27.0	3.4	26.6

Table 4: Confusion matrices for MR-Sort, MR-SortFP, and MR-Sort-CV when strengthening the bias in favor of false positives

5 Conclusion

We have presented MR-Sort-CV a new original extension of the MR-Sort ordered classification model. This model introduces a new and more general form of veto condition which applies on coalitions of criteria rather than a single criterion. This coalitional veto condition can be expressed as a reversed MR-Sort rule. Such a symmetry enables us to design a heuristic model to learn an MR-Sort-CV model, based on the use of an algorithm to learn MR-Sort. Preliminary results are interesting, but further investigations are needed to take benefit of this new ordered classification model.

REFERENCES

[1] Bouyssou, D., Marchant, T.: 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] Bouyssou, D., Marchant, T.: 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] Bouyssou, D., Pirlot, M.: A characterization of concordance relations. European Journal of Operational Research 167(2), 427–443 (2005)

[4] Bouyssou, D., Pirlot, M.: Further results on concordance relations. European Journal of Operational Research 181, 505–514 (2007)

[5] Cailloux, O., Meyer, P., Mousseau, V.: Eliciting ELECTRE TRI category limits for a group of decision makers. European Journal of Operational Research 223(1), 133–140 (2012)

[6] Chateauneuf, A., Jaffray, J.Y.: Derivation of some results on monotone capacities by Möbius inversion. In: Bouchon-Meunier, B., Yager, R.R. (eds.) Uncertainty in Knowledge-Based Systems, International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU '86, Paris, France, June 30 - July 4, 1986, Selected and Extended Contributions. Lecture Notes in Computer Science, vol. 286, pp. 95–102. Springer (1986), http://dx.doi.org/10.1007/3-540-18579-8_8

[7] Dias, L., Mousseau, V., Figueira, J.R., Clímaco, J.: An aggregation/disaggregation approach to obtain robust conclusions with ELECTRE TRI. European Journal of Operational Research 138(1), 332–348 (2002)

[8] Doumpos, M., Marinakis, Y., Marinaki, M., Zopounidis, C.: 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 (2009)

[9] Doumpos, M., Zopounidis, C.: Multicriteria Decision Aid Classification Methods. Kluwer Academic Publishers (2002)

[10] Leroy, A., Mousseau, V., Pirlot, M.: Learning the parameters of a multiple criteria sorting method. In: Brafman, R., Roberts, F., Tsoukiàs, A. (eds.) Algorithmic Decision Theory, Lecture Notes in Artificial Intelligence, vol. 6992, pp. 219–233. Springer (2011)

[11] Mousseau, V., Figueira, J.R., Naux, J.P.: Using assignment examples to infer weights for ELECTRE TRI method: Some experimental results. European Journal of Operational Research 130(1), 263–275 (2001)

[12] Mousseau, V., Słowiński, R.: Inferring an ELECTRE TRI model from assignment examples. Journal of Global Optimization 12(1), 157–174 (1998)

[13] Ngo The, A., Mousseau, V.: Using assignment examples to infer category limits for the ELECTRE TRI method. Journal of Multi-criteria Decision Analysis 11(1), 29–43 (2002)

[14] Olteanu, A.L., Meyer, P.: Inferring the parameters of a majority rule sorting model with vetoes on large datasets. In: DA2PL 2014 : From Multicriteria Decision Aid to Preference Learning. pp. 87–94 (2014)

[15] Roy, B.: The outranking approach and the foundations of ELECTRE methods. Theory and Decision 31, 49–73 (1991)

[16] Roy, B., Bouyssou, D.: Aide multicritère à la décision: méthodes et cas. Economica Paris (1993)

[17] Sobrie, O., Mousseau, V., Pirlot, M.: Learning a majority rule model from large sets of assignment examples. In: Perny, P., Pirlot, M., Tsoukiás, A. (eds.) Algorithmic Decision Theory. Lecture Notes in Artificial Intelligence, vol. 8176, pp. 336–350. Springer, Brussels, Belgium (2013)

[18] Sobrie, O., Mousseau, V., Pirlot, M.: Learning the parameters of a multiple criteria sorting method from large sets of assignment examples. In: 77th meeting of the EWG on MCDA. Rouen, France (April 2013)

[19] Sobrie, O., Mousseau, V., Pirlot, M.: Learning the parameters of a majority rule sorting model taking attribute interactions into account. In: DA2PL 2014 Workshop From Multiple Criteria Decision Aid to Preference Learning. pp. 22–30 (2014), `http://www.lgi.ecp.fr/~mousseau/DA2PL-2014`, paris, France

[20] Sobrie, O., Mousseau, V., Pirlot, M.: Learning the parameters of a non compensatory sorting model. In: Walsh, T. (ed.) Algorithmic Decision Theory. Lecture Notes in Artificial Intelligence, vol. 9346, pp. 153–170. Springer, Lexington, KY, USA (2015)

[21] Sobrie, O., Pirlot, M., Mousseau, V.: New veto relations for sorting models. Tech. rep., Laboratoire Génie Industriel, Ecole Centrale Paris (October 2014), `http://www.lgi.ecp.fr/Biblio/PDF/CR-LGI-2014-04.pdf`, cahiers de recherche 2014-04

[22] Tehrani, A.F., Cheng, W., Dembczyński, K., Hüllermeier, E.: Learning monotone nonlinear models using the Choquet integral. Machine Learning 89(1–2), 183–211 (2012)

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

[24] Zheng, J., Metchebon, S.A., Mousseau, V., Pirlot, M.: Learning criteria weights of an optimistic Electre Tri sorting rule. Computers & OR 49(0), 28–40 (2014), `http://www.sciencedirect.com/science/article/pii/S0305054814000677`

[25] Zopounidis, C., Doumpos, M.: Multicriteria classification and sorting methods: A literature review. European Journal of Operational Research 138(2), 229–246 (April 2002)