Construction and Refinement of Preference Ordered Decision Classes

Hoang Nhat Dau[1], Salem Chakhar[2,3], Djamila Ouelhadj[1,3] and Ahmed M. Abubahia[4]

[1] Department of Mathematics, Faculty of Technology, University of Portsmouth, UK
nhat.dau@myport.ac.uk, djamila.ouelhadj@port.ac.uk
[2] Portsmouth Business School, University of Portsmouth, UK
salem.chakhar@port.ac.uk
[3] Centre for Operational Research & Logistics, University of Portsmouth, UK
[4] School of Computing, Faculty of Technology, University of Portsmouth, UK
ahmed.abubahia@port.ac.uk

Abstract. Preference learning methods are commonly used in multicriteria analysis. The working principle of these methods is similar to classical machine learning techniques. A common issue to both machine learning and preference learning methods is the difficulty of the definition of decision classes and the assignment of objects to these classes, especially for large datasets. This paper proposes two procedures permitting to automatize the construction of decision classes. It also proposes two simple refinement procedures, that rely on the 80-20 principle, permitting to map the output of the construction procedures into a manageable set of decision classes. The proposed construction procedures rely on the most elementary preference relation, namely dominance relation, which avoids the need for additional information or distance/(di)similarity functions, as with most of existing clustering methods. Furthermore, the simplicity of the 80-20 principle on which the refinement procedures are based, make them very adequate to large datasets. Proposed procedures are illustrated and validated using real-world datasets.

Keywords: Clustering, Preference learning, Classification, Classes construction

1 Introduction

Preference learning methods [3][12] are commonly used in multicriteria analysis. These methods are often used to build a preference model based on a sample of past decisions for further prescriptive decision purposes. Preference learning methods have been inspired by knowledge discovery techniques and preference modeling methods. The working principle of preference learning methods is similar to classical machine learning methods [10]: they use a subset of data, called learning set, to extract some knowledge permitting to classify unseen objects. In contrary to machine learning approaches, preference learning methods assume that both attributes and decision classes are preference-ordered.

The definition of the learning set is a crucial step in the application of machine and preference learning methods [1][11]. It involves two operations: (i) the selection of a representative subset of objects, and (ii) their assignment into different pre-defined decision classes. A common issue to both to machine learning and preference learning methods is the difficulty of defining the decision classes. The situation is further complicated especially for large datasets.

One possible solution to define the learning set is to use one of existing multicriteria ordered clustering methods e.g. [2][4][5][6][7][14]. However, these methods are very demanding in terms of additional information (such as preference parameters) and require the specification of a distance/(di)similarity functions.

In this paper, we propose two procedures automatizing the construction of decision classes. These procedures permit to reduce the cognitive effort required from the decision maker and then extend the application domain of preference learning methods. We also propose two simple 80-20 principle-based refinement procedures that permit to map the output of the construction procedures into a manageable set of decision classes.

The rest of the paper goes as follows. Section 2 provides the background. Sections 3 and 4 present the construction and refinement procedures, and Section 5 illustrates them using a real-world dataset. Section 6 concludes the paper.

2 Background

2.1 Problem Description

Let U be a non-empty finite set of objects and Q is a non-empty finite set of attributes such that $q : U \to V_q$ for every $q \in Q$. The V_q is the domain of attribute q, $V = \bigcup_{q \in Q} V_q$, and $f : U \times Q \to V$ is the information function defined such that $f(x, q) \in V_q$ for each attribute q and object $x \in U$. The value $f(x, q)$ corresponds to the evaluation of object x on attribute $q \in Q$. The domains of condition attributes are supposed to be ordered according to a decreasing or increasing preference. Such attributes are often called criteria.

The objective is to partition U into a finite number of preference-ordered decision classes $\mathbf{Cl} = \{Cl_t, t \in T\}$, $T = \{1, \cdots, n\}$, such that each $x \in U$ belongs to one and only one class.

2.2 Dominance Relation and Graph

The dominance relation Δ is defined for each pair of objects x and y as follows:

$$x \Delta y \Leftrightarrow f(x, q) \geq f(y, q), \forall q \in Q. \tag{1}$$

This definition applies for gain-type (i.e. benefit) criteria. For cost-type criteria, the symbol '$\geq$' should be replaced with '$\leq$'. Equation (1) implements the weak version of the dominance relation since all the inequality are large. Clearly, this version of dominance relation is reflexive (i.e. $x \Delta x$, $x \in U$) and transitive (i.e. if $x \Delta y$ and $y \Delta z$, then $x \Delta z$, $\forall x, y, z \in U$) (but in general not complete).

Consequently, it defines a partial preorder on U. The strict version of the dominance relation Δ_s requires at least one strict inequality in Equation (1). The strict dominance relation is no longer reflexive but still transitive. It is easy to see that $\Delta_s \subseteq \Delta$ (i.e. $x\Delta_s y \Rightarrow x\Delta y$, $\forall x, y \in U$).

The dominance relations on $U \times U$ can be summarized through a dominance matrix $M[x_{ij}]h \times h$ where $h = |U|$, i.e. the number of objects in U and

$$x_{ij} = \begin{cases} 1, \text{ If } x_i \Delta x_j, \\ 0, \text{ Otherwise.} \end{cases} \tag{2}$$

The dominance relation defines a partial preorder on the objects set U. Any preorder can be represented by a directed graph, with elements of the set corresponding to vertices, and the order relation between pairs of elements corresponding to the directed edges between vertices. Therefore, the dominance relation can be represented as directed graph $G = (U, E)$ with elements U corresponding to vertices, and the dominance relation between pairs of elements corresponding to the directed edges E between vertices, defined as $E = \{(x, y) \in U \times U : x\Delta y\}$. The graph G is constructed using a top-to-bottom order. This means that if a node x dominates a node y, x appears above y in the graph.

2.3 Running Example

Table 1 is an extract from dataset used in Section 5. It provides the description of 10 objects with respect to a set $Q = \{A_1, A_2, A_3, A_4\}$ of four criteria, which are introduced later in Section 5. At this level, we just mention that all the criteria are to be maximized and that the first and fourth criteria are ordinal while the second and third are continuous.

Table 1. Information Table

#	A_1	A_2	A_3	A_4
1	2	1312.5	26.25	2
2	2	1365	27.30	2
3	1	347.76	19.32	1
4	1	74.8	7.48	1
5	1	1117.98	62.11	2
6	1	1289.88	71.66	2
7	3	193.55	38.71	1
8	4	1313	13.13	2
9	3	326	3.26	1
10	3	2268	126.00	1

Table 2 is the dominance matrix associated with the dataset in Table 1. The corresponding dominance graph is shown in Figure 1. We note that the self dominance relationships are omitted for simplicity.

3 Construction Procedures

The construction procedures are based on the dominance relation, which is the most elementary preference information. Let $Cl(x)$ be the target class of decision

Table 2. The dominance Matrix

	1	2	3	4	5	6	7	8	9	10
1	1	0	1	1	0	0	0	0	0	0
2	1	1	1	1	0	0	0	0	0	0
3	0	0	1	1	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0	0
5	0	0	1	1	1	0	0	0	0	0
6	0	0	1	1	1	1	0	0	0	0
7	0	0	0	1	0	0	1	0	0	0
8	0	0	0	1	0	0	0	1	1	0
9	0	0	0	0	0	0	0	0	1	0
10	0	0	1	1	0	0	1	0	1	1

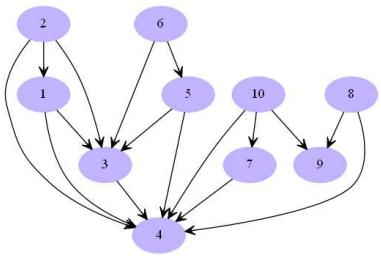

Fig. 1. Dominance Graph

object $x \in U$. Then, the assignment of objects to decision classes relies on the following construction rules:

$$x \Delta y \Leftrightarrow Cl(x) \geq Cl(y), \forall x, y \in U. \tag{3}$$

$$x \Delta_s y \Leftrightarrow Cl(x) > Cl(y), \forall x, y \in U. \tag{4}$$

Based on these rules, we designed two basic procedures: Top-Down and Bottom-Up. These procedures are introduced in the rest of this section.

3.1 Top-Down Procedure

The set of decision classes can be induced from the directed graph $G = (U, E)$ using the following idea. First, we identify a minimal subset $N \subseteq U$ such that: (i) any decision object that is not in N is dominated by at least one decision object from N; and (ii) the objects in set N are incomparable (i.e. they do not dominate each other, except of the self dominance). The set N is called the *kernel* of graph G, the dominant subset or also the external stability. The elements of N are assigned to the most preferred decision class Cl_n. Then, the same procedure is used to identify the kernel N' of the sub-graph $G' = (U \setminus N, E')$ and the elements of set N' are then assigned to the second most preferred decision class Cl_{n-1}. The same procedure is repeated until all the objects are assigned. This idea is formalized in Algorithm 1. The procedure **Kernel** in this algorithm permits to compute the kernel of the graph given as parameter. Different procedures for

computing a Kernel of graph are available in the literature [8][9]. Algorithm 1 runs in $O(\beta|U|)$ where β is the complexity of computing the kernel.

Algorithm 1: TopDown Procedure

Input : $S = \langle U, Q, V, f\rangle$, // information table.
Output: Cl, // equivalence classes.
$n \longleftarrow |U|$;
$Z \longleftarrow \emptyset$;
while $(Z \neq U)$ **do**
  $E \longleftarrow \{(x, y) : x, y \in U \setminus Z$ and $x \Delta y\}$;
  $G \longleftarrow (U \setminus Z, E)$;
  $Cl_n \longleftarrow$ **Kernel**(G);
  $Z \longleftarrow Z \cup Cl_n$;
  $n \longleftarrow n - 1$;
re-label decision classes $Cl_{|U|}, \cdots, Cl_{n+1}$ as $Cl_{|U|-n}, \cdots, Cl_1$;
$Cl \longleftarrow \{Cl_1, \cdots, Cl_n\}$;
return Cl;

Figure 2 illustrates the application of the Top-Down procedure on the dominance graph of Figure 1. The nodes with bold boundary constitute the kernel. The procedure leads to four classes: $Cl_1 = \{4\}$, $Cl_2 = \{3\}$, $Cl_3 = \{1, 5, 7, 9\}$ and $Cl_4 = \{2, 6, 8, 10\}$, where Cl_4 is the most preferred class.

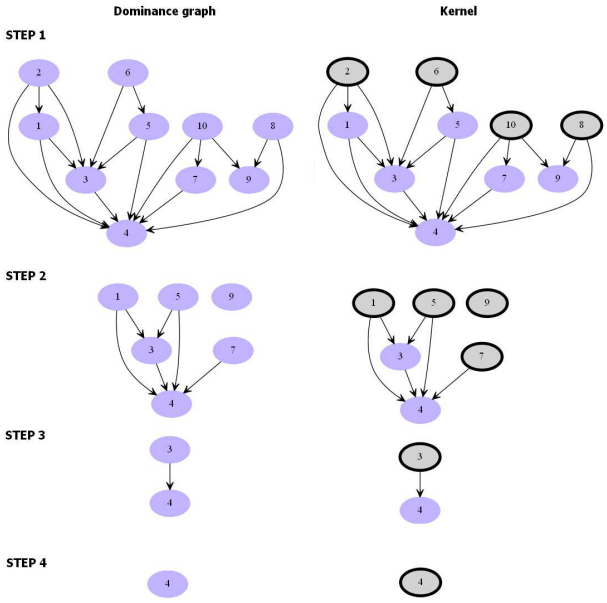

Fig. 2. Illustration of Top-Down Procedure

3.2 Bottom-Up Procedure

The set of decision classes can also be induced by examining the directed graph $G = (U, E)$ from bottom to up. First, we identify a minimal subset $M \subseteq U$ such that: (i) any decision object that is not in M dominates at least one decision object from M; and (ii) the objects in set M are incomparable. We call set M the *anti-kernel* of graph G. The elements of M are assigned to the less preferred decision class Cl_1. Then, the same procedure is used to identify the anti-kernel M' of the sub-graph $G' = (U \setminus M, E')$ and the elements of set M' are then assigned to the second less preferred decision class Cl_2. The same procedure is repeated until all the objects are assigned. This idea is formalized in Algorithm 2. The procedure **AntiKernel** in this algorithm permits to compute the anti-kernel of the graph given as parameter. The procedures for identifying the AntiKernel of a graph can be obtained by adapting those used to compute graph Kernels [8][9]. Algorithm 2 runs in $O(\gamma|U|)$ where γ is the complexity of computing the anti-kernel.

Algorithm 2: BottomUp Procedure

Input : $S = \langle U, Q, V, f \rangle$, // information table.
Output: Cl, // equivalence classes.

$n \longleftarrow 1$;
$Z \longleftarrow \emptyset$;
while $(Z \neq U)$ **do**
  $E \longleftarrow \{(x, y) : x, y \in U \setminus Z \text{ and } y \Delta x\}$;
  $G \longleftarrow (U \setminus Z, E)$;
  $Cl_n \longleftarrow$ **AntiKernel**(G);
  $Z \longleftarrow Z \cup Cl_n$;
  $n \longleftarrow n + 1$;
$Cl \longleftarrow \{Cl_1, \cdots, Cl_{n-1}\}$;
return Cl;

Figure 3 illustrates the application of the Bottom-Up procedure on the dominance graph of Figure 1. The nodes with double circle boundary constitute the anti-kernel. The procedure leads to four classes: $Cl_1 = \{4, 9\}$, $Cl_2 = \{3, 7\}$, $Cl_3 = \{1, 5\}$ and $Cl_4 = \{2, 6, 8, 10\}$, where Cl_4 is the most preferred class.

4 Refinement of Decision Classes

The number of decision classes may be very high. Hence, there is a need to reduce the number of these classes to obtain a manageable set of decision classes. Two simple refinement procedures that rely on the 80-20 rule are proposed in this paper. The 80-20 principle (also known as Pareto principle) relies on the fact that, for many events, roughly 80% of the effects come from 20% of the causes.

Let assume that there are n classes (with Cl_n is best class and Cl_1 is the worst) that should be mapped into $p < n$ classes. The first refinement procedure starts from the top and merge the 20% best classes into class Cl_p. The classes already assigned are removed and then the 20% of best classes of the remaining ones are merged to class Cl_{p-1}. The algorithm continues until the definition of $p - 1$ classes. At the end, the remaining classes are merged into class Cl_1.

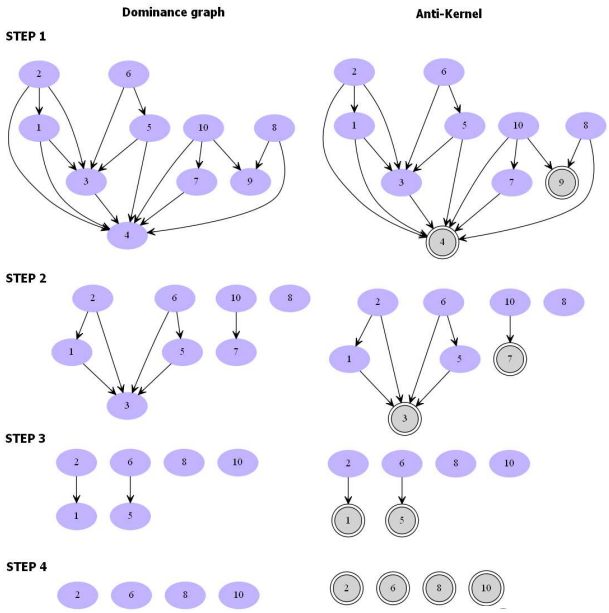

Fig. 3. Illustration of Bottom-Up Procedure

This first solution may lead to large classes. A better solution consists in using the 80-20 rule but by considering objects instead of classes. Let assume that there are n classes (with Cl_n is best class and Cl_1 is the worst) that should be mapped into $p < n$ classes. Then, the working principle of the refinement procedure using 80-20 rule on decision objects is as follows. It starts from the top and merges a set of best classes that contain the best 20% objects into class Cl_p. The classes already assigned are removed and then the classes containing the top 20% of best remaining objects are merged into class Cl_{p-1}. The procedure continues until the definition of $p-1$ classes. At the end, the remaining classes are merged into class Cl_1.

In both versions, a strict application of 80-20 rule may lead to large classes. One possible solution to handle this issue is to use a parameter ϑ to relax the merging conditions.

Algorithm 3 implements the second refinement procedure. In this algorithm, we assumed that there are n classes with Cl_n is best class and Cl_1 is the worst.

5 Application and Validation

5.1 Dataset

For the purpose of illustration, we consider a real-world dataset (reproduced from [13]) relative to spare parts management corresponding to a Chinese firm. The dataset is composed of 98 spare parts described with respect to four criteria,

Algorithm 3: Refinement Using 80-20 Rule on Decision Objects

Input : $L = \{Cl_1, \cdots, Cl_n\}$, // initial classes.
β, // integer.
Output: $O = \{K_1, \cdots, K_p\}$, // refined classes.

$i \longleftarrow n$;
while $(i \geq 2)$ **do**
  $K_i \longleftarrow$ top classes in L containing the top $20\% \pm \vartheta$ best objects;
  $L \longleftarrow L \setminus K_i$;
  $i \longleftarrow i - 1$;
$K_1 \longleftarrow L$;
$O \longleftarrow \{K_1, \cdots, K_p\}$;
return O;

namely A_1 (Criticality), A_2 (Annual Dollar Usage), A_3 (Average Unit Cost), and A_4 (Lead Time) (see Table 3). The criteria A_2 and A_3 are continuous while A_1 and A_4 criteria are ordinal. The criterion A_1 can take one of four values 1, 2, 3 and 4 where 1 corresponds to the lowest criticality and 4 corresponds to highest critically. The possible values for A_4 are 1, 2 and 3 where 1 means a low lead time and 3 means a high lead time. All the criteria are benefit-type (i.e. the higher their values, the more important spare part is). The evaluation of the spare parts with respect to criteria is given in Table 4.

Table 3. Characteristics of Considered Criteria

Code	Name	Description	Preference	Data type
A_1	Criticality	It represents the influence of spare parts running out on the availability of equipment.	Gain	Ordinal
A_2	Annual Dollar Usage	It is calculated by spare part cost multiply demand volume.	Gain	Continuous
A_3	Average Unit Cost	It refers to spare part cost.	Gain	Continuous
A_4	Lead Time	It refers to the time between the placement of an order and delivery of a new spare part from the firm's supplier.	Gain	Ordinal

5.2 Application and Results

We applied the Top-Down and Bottom-Up procedures using the dataset given in Table 4. The results of these procedures are given in Table 5. As shown in this table, the use of the Top-down procedure leads to 22 classes with class #1 is the most preferred and class #22 is the worst while the application of the Bottom-Up procedure leads to 23 classes with class #23 is most preferred and class #1 is the worst. As shown in Table 5, both the Top-Down and Bottom-Up procedures lead to a high number of classes and should be mapped into a reduced set of classes. Let assume that the obtained decision classes should be mapped into $p = 3$ ordered decision classes, labelled A, B and C, respectively.

Table 4. Information Table

#	A_1	A_2	A_3	A_4	#	A_1	A_2	A_3	A_4	#	A_1	A_2	A_3	A_4
1	2	1312.5	26.25	2	34	4	1260	12.6	2	67	3	2126.25	47.25	1
2	2	1365	27.3	2	35	4	2100	21	2	68	3	623.7	34.65	2
3	1	347.76	19.32	1	36	4	1050	10.5	2	69	3	420	4.2	2
4	1	74.8	7.48	1	37	4	1575	15.75	2	70	4	840	8.4	2
5	1	1117.98	62.11	2	38	4	578	5.78	2	71	4	3150	31.5	3
6	1	1289.88	71.66	2	39	3	2936	29.36	1	72	4	2625	26.25	3
7	3	193.55	38.71	1	40	4	19682.3	3936.46	2	73	4	13925.3	1392.53	3
8	4	1313	13.13	2	41	4	1444.2	24.07	1	74	3	199.5	5.25	2
9	3	326	3.26	1	42	3	463.05	13.23	1	75	3	472.5	9.45	2
10	3	2268	126	1	43	3	132.3	7.35	1	76	3	336	16.8	2
11	3	4134.6	91.88	1	44	1	2734.2	97.65	1	77	1	57.8	5.78	2
12	3	1587.6	88.2	1	45	1	3071.25	87.75	1	78	1	161.84	5.78	2
13	3	2063.4	54.3	1	46	4	785.7	17.46	1	79	3	840	8.4	2
14	3	1786.4	44.66	1	47	4	955.2	11.94	1	80	4	840	8.4	3
15	3	10365.75	121.95	1	48	1	28.44	1.58	1	81	4	2625	26.25	3
16	3	770.26	20.27	1	49	4	851	8.51	2	82	4	2100	21	3
17	3	2646	52.92	1	50	1	352.8	8.82	1	83	4	25725	257.25	3
18	3	113.4	5.67	1	51	3	105.84	2.94	1	84	4	40056	400.56	3
19	1	650	65	1	52	1	1304.48	21.04	1	85	4	3780	126	2
20	1	418.88	14.96	1	53	4	3580.5	65.1	2	86	3	882	29.4	2
21	1	948.3	31.61	1	54	2	1325.52	73.64	1	87	1	1470	36.75	2
22	3	410.7	13.69	1	55	4	18375	1837.5	3	88	3	126	12.6	2
23	3	26995.6	2699.56	2	56	2	236.7	2.63	1	89	4	1071	17.85	2
24	4	746	7.46	2	57	2	862	8.62	1	90	3	121.1	3.46	2
25	4	3150	31.5	2	58	3	735	7.35	1	91	3	43.56	2.42	2
26	4	3675	36.75	2	59	1	2315.34	128.63	1	92	1	823.2	29.4	2
27	3	27562.5	1837.5	2	60	1	1984.5	110.25	1	93	1	1029	5.78	2
28	4	840	8.4	3	61	1	157.5	15.75	1	94	4	1025.55	22.79	2
29	4	1670	16.7	2	62	1	340.2	18.9	2	95	4	2688	33.6	2
30	4	1754	17.54	2	63	1	642.6	35.7	2	96	3	1470	29.4	2
31	4	437	4.37	2	64	4	346.5	34.65	2	97	2	264.6	14.7	2
32	4	2625	26.25	2	65	3	1890	189	1	98	4	11025	5512.5	3
33	4	462	4.62	2	66	3	567	31.5	1					

Table 5. Results of Top-Down and Bottom-Up Procedures

Preference order	Top-Down		Bottom-Up	
	Class ID	Content	Class ID	Content
Best	1	23, 27, 40, 55, 84, 98	1	55, 84
	2	73, 83,	2	40, 73, 83, 98
	3	15, 59, 65, 71, 85	3	85
	4	6, 10, 11, 26, 44, 53, 72, 81	4	26, 53, 71
	5	5, 12, 13, 17, 25, 39, 45, 60, 64, 68, 82, 87, 95	5	25, 72, 81, 95
	6	32, 54, 63, 67, 96, 28, 80	6	32, 82
	7	2, 14, 19, 35, 41, 86, 94	7	35
	8	1, 7, 16, 21, 30, 66, 89, 92	8	30
	9	29, 46, 52, 62, 76	9	29
	10	3, 37	10	37
	11	8, 20, 22, 42, 61, 97	11	8
	12	34, 49, 88	12	23, 27, 34, 89
	13	36, 47,	13	15, 36, 94
	14	57, 70, 75, 93	14	11, 28, 49, 80
	15	24, 79	15	10, 17, 70
	16	38, 50, 58	16	13, 24, 65, 67, 96
	17	4, 33, 43, 74, 78	17	2, 6, 12, 14, 38, 39, 41, 59, 68, 86
	18	18, 31, 77	18	1, 5, 16, 33, 44, 45, 46, 47, 54, 60, 64, 66, 79, 87
	19	69,	19	7, 19, 21, 22, 31, 42, 52, 58, 63, 75, 76, 92
	20	9, 90	20	3, 20, 43, 62, 69, 74, 88, 93, 97
	21	51, 56, 91	21	9, 18, 50, 57, 61, 78, 90
	22	48	22	4, 51, 56, 77, 91
Worst			23	48

In this paper, we used the second refinement procedure. The application of this procedure on the output of Top-Down and Bottom-Up procedures is summarized in Table 6. In both cases, we used a relaxation parameter of $\vartheta = 5$. Table 6 shows that the refinement of the Top-Down procedure's output leads to the assignment of 21 spare parts to class A, 13 to class B and 64 to class C, which makes a percentage of 21.43%, 13.27% and 65.30%, respectively. It also shows that the refinement of the Bottom-Up procedure's output leads to relatively the same rate with the assignment of 19 spare parts to class A, 16

to class B and 63 to class C, which makes a percentage of 19.39%, 16.33% and 64.28%, respectively.

Table 6. Results of Refinement

Preference order	Top-Down				Bottom-Up			
	Class ID	Content	Nb of objects	%	Class ID	Content	Nb of objects	%
Best	A	23, 27, 40, 55, 84, 98, 73, 83, 15, 59, 65, 71, 85, 6, 10, 11, 26, 44, 53, 72, 81	21	21.43	A	55, 84, 40, 73, 83, 98, 85, 26, 53, 71, 25, 72, 81, 95, 32, 82, 35, 30, 29	19	19.39
	B	5, 12, 13, 17, 25, 39, 45, 60, 64, 68, 82, 87, 95	13	13.27	B	37, 8, 23, 27, 34, 89, 15, 36, 94, 11, 28, 49, 80, 10, 17, 70	16	16.33
	C	32, 54, 63, 67, 96, 28, 80, 2, 14, 19, 35, 41, 86, 94, 1, 7, 16, 21, 30, 66, 89, 92, 29, 46, 52, 62, 76, 3, 37, 8, 20, 22, 42, 61, 97, 34, 49, 88, 36, 47, 57, 70, 75, 93, 24, 79, 38, 50, 58, 4, 33, 43, 74, 78, 18, 31, 77, 69, 9, 90, 51, 56, 91, 48	64	65.30	C	13, 24, 65, 67, 96, 2, 6, 12, 14, 38, 39, 41, 59, 68, 86, 1, 5, 16, 33, 44, 45, 46, 47, 54, 60, 64, 66, 79, 87, 7, 19, 21, 22, 31, 42, 52, 58, 63, 75, 76, 92, 3, 20, 43, 62, 69, 74, 88, 93, 97, 9, 18, 50, 57, 61, 78, 90, 4, 51, 56, 77, 91, 48	63	64.28

5.3 Validation

To validate the results, we applied the preference learning method Dominance-based Rough Set Approach (DRSA) [12] on the output of refinement procedures. The input of DRSA is a learning dataset representing the description of a set of objects with respect to a set of criteria. The main output of DRSA is a collection of decision rules. A decision rule is a consequence relation $E \rightarrow H$ (read as If E, then H) where E is a condition (evidence or premise) and H is a conclusion (decision). Each elementary condition is built upon a single criterion while a consequence is defined based on a decision class. The obtained decision rules can then be applied to classify unseen objects.

We applied the DRSA using the outputs (after refinement) of the Top-Down and Bottom-Up procedures as learning sets. This led to a collection of decision rules, which are then validated through the reclassification of the spare parts in the learning sets using the inferred decision rules. The results of reclassification are summarized in the confusion matrices given in Table 7. As shown in this table, the reclassification using the results obtained from the Bottom-Up procedure shows a perfect match since all the spare parts have been re-classified to the same class. In turn, the reclassification using the results obtained from the Top-Down procedure shows a rate of 92% of correct classifications and a rate of 8% of misclassifications. This holds because some spare parts have been assigned to more than one class.

To further evaluate the results of the DRSA, we used a series of well-known non-parametric statistics to measure the correlation between the assignments provided by the Top-Down and Bottom-Up procedures and those generated

Table 7. Confusion Matrices

	Possible		
Original	C	B	A
C	64/64	0/1	0/1
B	0/5	13/13	0/2
A	0/0	0/1	21/21

Top-Down Procedure

	Possible		
Original	C	B	A
C	63/63	0/0	0/0
B	0/0	16/16	0/0
A	0/0	0/0	19/19

Bottom-Up Procedure

based on the corresponding rules. The result of the statistical analysis is summarized in Table 8. Let us first note that for the Top-Down procedure, we distinguished two cases concerning the 9 ambiguous assignments: (i) case of best choice in which the assignment intervals have been reduced into a single assignment equal to the one generated by Top-Down procedure, and (ii) worst choice in which the assignment intervals have been reduced into a single assignment different from the one generated by Top-Down procedure. Based on Table 8, we can conclude that all the statistics show a perfect agreement for the case of Bottom-Up procedure and Top-Down with best choices procedure. For the Top-Down procedure with worst choices, all the statistics show a relatively high to very high agreement.

Table 8. Statistical analysis

Statistics Statistics	Kendall's tau	Spearman's rho	Unweighted kappa	Weighted kappa
Top-Down with best choices vs rules	1	1	1	1
Top-Down with worst choices vs rules	0.8698	0.8893	0.8257	0.8714
Bottom-Up vs rules	1	1	1	1

6 Conclusion

The paper addresses the problem of ordered decision classes construction and refinement. It has several theoretical and practical contributions. Firstly, it proposes two procedures permitting to 'automatise' the construction of decision classes. These procedures rely on the most elementary preference relation, namely dominance relation. Thus and in contrary to most of existing clustering methods, the proposed procedures avoid the need for additional information or distance/(di)similarity functions. Secondly, it extends the application domain of preference learning methods, especially the DRSA, to decision problems involving large datasets. Thirdly, it introduces two refinement procedures that rely on the 80-20 principle. The refinement procedures permit to map the output of the construction procedures into a manageable set of decision classes. The simplicity of the 80-20 principle on which the refinement procedures are based, make them very adequate to large datasets.

Several topics need to be investigated in the future. The first topic concerns the use of a mixed construction procedure by combining the results of Top-Down

and Bottom-Up procedures. The second topic is related to the application and performance evaluation of the procedures with very large data sets. The last topic concerns the use of other refinement techniques.

References

1. Albatineh, A., M.Niewiadomska-Bugaj: MCS: A method for finding the number of clusters. Journal of Classification **28**(2) (2011) 184–209
2. Baroudi, R., Bahloul, S.: A multicriteria clustering approach based on similarity indices and clustering ensemble techniques. International Journal of Information Technology and Decision Making **13**(04) (2014) 811–837
3. Bregar, A., Györkös, J., Jurič, M.: Interactive aggregation/disaggregation dichotomic sorting procedure for group decision analysis based on the threshold model. Informatica **19**(2) (2008) 161–190
4. de la Paz-Marín, M., Gutiérrez, P., Hervás-Martínez, C.: Classification of countries' progress toward a knowledge economy based on machine learning classification techniques. Expert Systems with Applications **42**(1) (2015) 562–572
5. De Smet, Y.: P2CLUST: An extension of PROMETHEE II for multicriteria ordered clustering. In: Industrial Engineering and Engineering Management (IEEM), 2013 IEEE International Conference on. (Dec 2013) 848–851
6. De Smet, Y.: An extension of PROMETHEE to divisive hierarchical multicriteria clustering. In: Industrial Engineering and Engineering Management (IEEM), 2014 IEEE International Conference on. (Dec 2014) 555–558
7. De Smet, Y., Nemery, P., Selvaraj, R.: An exact algorithm for the multicriteria ordered clustering problem. Omega **40**(6) (2012) 861–869
8. Emmert-Streib, F., Dehmer, M., Shi, Y.: Fifty years of graph matching, network alignment and network comparison. Information Sciences **346-347** (2016) 180–197
9. Ghosh, S., Das, N., calves, T.G., Quaresma, P., Kundu, M.: The journey of graph kernels through two decades. Computer Science Review **27** (2018) 88–111
10. Gilboa, I., Schmeidler, D.: Case-based knowledge and induction. IEEE Transactions on Systems, Man, and Cybernetics: Part A **30**(2) (2000) 85–95
11. Gionis, A., Mannila, H., Tsaparas, P.: Clustering aggregation. ACM Transactions on Knowledge Discovery from Data **1**(1) (2007) Article 4.
12. Greco, S., Matarazzo, B., Słowiński, R.: Rough sets theory for multicriteria decision analysis. European Journal of Operational Research **129**(1) (2001) 1–47
13. Hu, Q., Chakhar, S., Siraj, S., Labib, A.: Spare parts classification in industrial manufacturing using the dominance-based rough set approach. European Journal of Operational Research **262**(3) (2017) 1136–1163
14. Rocha, C., Dias, L.: MPOC: an agglomerative algorithm for multicriteria partially ordered clustering. 4OR **11**(3) (2013) 253–273