Project report: Student admission

Ducret Romain, Laubé Nicolas and Stutz Antoine

29/01/2022

CentraleSupélec

Academical year 2021-2022

Contents

1 Introduction & context

Throughout this project, we will consider a situation in which a committee for a higher education program has to decide about the admission and merit of students on the basis of their evaluations in different courses. We will study two different student classification models: NCS and MRSort (that will be defined later).

We will implement Inv-MRSort and Inv-NCS methods associated to the previously mentioned models to solve this problem. Both methods rely on a data set (for example the data set of last year's admissions) composed of students grades. Besides, for each student, we should have access to its classification (admitted/rejected or mention). Students were evaluated on n different subjects. In the data set, each student is represented by a n-tuple of evaluations in the different courses. In the following sections, we denote u_j the student j and $\forall i \in N, u_{j,i}$ the grade of student j in course i.

Of course, all methods presented in the following sections can be generalized to other problems.

2 Multicriteria models formulation

2.1 MR-Sort

MR-sort is a sorting model based on a majority rule. Each grade i has a different weight (importance) w_i and we assume that the weights are normalize $\sum_{i \in N} w_i = 1$.

To affect a student to the correct class (merit), we must find the correct class C^h such that

- they have sufficient grades for class C^h
- they don't have sufficient grades for class C^{h+1}

The grades are sufficient for class C^h if the sum of all the weights w_i corresponding to the best grades i of the student (i.e. all the subjects i for which $u_i \geq b_i^h$) is greater or equal to λ.

By the way, we must ensure that a student in class C^h must satisfy the criteria of a class $C^{h'}$ when $h' < h$. To do that, the borders must always be more selective: $\forall i, b_i^h \geq b_i^{h'}$.

MR-Sort

Considering:

- a set of classes: $C^0 ... C^p$
- a set of subjects: $m_1 ... m_n$
- a max grade N (the grades range between 0 and N)
- a set of frontiers (limit grades): $\forall h \in [\![1, p]\!], b^h \in [0, N]^n$
- a set of weights: $w_0 ... w_n \in \mathbb{R}^+$
- a constant: $\lambda \in [0, 1]$

Under constraints:

- $\forall h < h' \in [\![1, p]\!], \forall i \in [\![1, n]\!], b_i^h \leq b_i^{h'}$
- $\sum_{i=1}^{n} w_i = 1$

Is defined:

$u \in [0, N]^n$ is in class C^h if and only if $\sum_{\{i \in [\![0,N]\!] / u_i \geq b_i^h\}} w_i \geq \lambda$ and $\sum_{\{i \in [\![0,N]\!] / u_i \geq b_i^{h+1}\}} w_i < \lambda$

2.2 NCS

NCS is a more general model. Instead of using weights and a lambda value to assess to which class a student must be associated, the set of grades above the frontier is compared to a set of sufficient coalitions T.
Thus, if we have for example two scientific grades (for maths and physics) and two literary grades (for french and english), we can set a student to a superior category if he validate one scientific and one literary grade.

NCS

Considering:

- a set of classes: $C^0 ... C^p$
- a set of subjects: $m_1 ... m_n$
- a max grade N (the grades range between 0 and N)
- a set of frontiers (limit grades): $\forall h \in [\![1, p]\!], b^h \in [0, N]^n$
- a set of sufficient coalitions T

Under constraints:

- $\forall h < h' \in [\![1, p]\!], \forall i \in [\![1, n]\!], b_i^h \leq b_i^{h'}$

Is defined:

$u \in [\![0, N]\!]^n$ is in class C^h if and only if $\{i \in [\![1, n]\!] / u_i \geq b_i^h\} \in T$ and $\{i \in [\![1, n]\!] / u_i \geq b_i^{h+1}\} \notin T$

2.3 NCS with intervals

We can also generalize a bit the NCS model. Indeed, the previous attribution method based on comparing a student grade only to a lower limit may be unfitted for some problems. For example, when doctors want to assess how good the glycemia level of one of their patient is, they must ensure it is above a min level (so they

don't have hypoglycemia) and below a max one (so they don't have hyperglycemia).

We then define categories thanks to both a min and a max border. A student u is then assigned to class C^h based on the grades u_i that are between the borders of class C^h ($[b^h_{min,i}, b^h_{max,i}]$) and outside the borders of class C^{h+A} ($[b^{h+1}_{min,i}, b^{h+1}_{max,i}]$).
We must modify the continuity conditions of the border. To be always more selective, the borders of a class C^h must be inside the ones of a class $C^{h'}$ when $h' < h$ ($[b^h_{min,i}, b^h_{max,i}] \subset [b^{h'}_{min,i}, b^{h'}_{max,i}]$).

NCS with intervals

Considering:

- a set of classes: $C^0 ... C^p$
- a set of subjects: $m_1 ... m_n$
- a max grade N (the grades range between 0 and N)
- a set of min and max frontiers (limit grades): $\forall h \in [\![1, p]\!], b^h_{min} \in [\![0, N]\!]^n$ *and* $b^h_{max} \in [\![0, N]\!]^n$
- a set of sufficient coalitions T

Under constraints:

- $\forall h < h' \in [\![1, p]\!], \forall i \in [\![1, n]\!], [b^{h'}_{i,min}, b^{h'}_{i,max}] \subset [b^h_{i,min}, b^h_{i,max}]$

Is defined:

$$u \in [\![0, N]\!]^n \text{ is in class } C^h \text{ if and only if } \{i \in [\![1, n]\!] / b^h_{i,min} \leq u_i \leq b^h_{i,max}\} \in T \text{ and}$$
$$\{i \in [\![1, n]\!] / b^{h+1}_{i,min} \leq u_i \leq b^{h+1}_{i,max}\} \notin T$$

3 MR-Sort resolution

To solve the Inv-MRSort problem, we used a linear solver (gurobi).

3.1 Binary MR-sort

First, we implemented the binary MR-sort algorithm. Thus, we considered the simple case where there are only two categories, accept (A) and reject (R) with one border between. More precisely, suppose we have a data set of accepted students S_A, and rejected students S_R.

Binary Inv-MRSort

Considering:

- a max grade N (the grades range between 0 and N) and a number of subjects n
- a set of accepted students S_A of length $nb_{stu,A}$: $\{u_a \in [0,N]^n\}_{u_a \in S_A}$
- a set of rejected students S_R of length $nb_{stu,R}$: $\{u_r \in [0,N]^n\}_{u_r \in S_R}$
- M a really big constant, ϵ a very small one, $nb_{stu} = nb_{stu,A} + nb_{stu,R}$

With the following variables:

- $w_i \in [0,1],\ b_i \in [0,N]\ \forall i \in [\![1,n]\!]$
- $c_{i,j} \in [0,1],\ \delta_{i,j} \in \{0,1\}\ \forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!]$
- $\lambda \in [0.5,1],\ \alpha \in \mathbb{R}$
- $x_j, y_j \in \mathbb{R}\ \forall j \in [\![1,nb_{stu}]\!]$

Under constraints:

- $$\sum_i w_i = 1 \tag{1}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!]$,
$$c_{i,j} \leq w_i \tag{2}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!]$,
$$c_{i,j} \leq \delta_{i,j} \tag{3}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!]$,
$$c_{i,j} \geq \delta_{i,j} - 1 + w_i \tag{4}$$
- $\forall j \in [\![1,nb_{stu}]\!]$,
$$\alpha \leq x_j,\ \alpha \leq y_j \tag{5}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu,A}]\!]$,
$$\sum_i c_{i,j} = \lambda + y_j \tag{6}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![nb_{stu,A}+1, nb_{stu,A}+nb_{stu,B}]\!]$,
$$\sum_i c_{i,j} + x_j + \epsilon = \lambda \tag{7}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!]$,
$$M(\delta_{i,j} - 1) \leq u_{i,j} - b_i \tag{8}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!]$,
$$M\delta_{i,j} + \epsilon \geq u_{i,j} - b_i \tag{9}$$

Find:

$$max\ \alpha \tag{10}$$

In the previous resolution we have a few variables:

- $\delta_{i,j}$ variable will be set to 0 if the grade i of student j is considered to be below the border b_i, to 1 otherwise.
- $c_{i,j}$ variables will be set to 0 if the grade i of student j is considered to be below the border b_i, to wi otherwise.
- x_j and y_j are two continuous variables introduced to compute the goal function.

The first four equations represent the structural constraints. Equations 5 to 9 represent the resolution constraints of our problem and equation 10 represents our goal function.

3.2 Relaxed binary MR-sort

A potential problem of the previous method is that our dataset may not always be perfectly separable. We thus implemented a relaxed/flexible to outliers version of the method. A gaussian noise $\mu = 0$ & $\sigma = 0.1$ is used.

Relaxed Binary Inv-MRSort

Considering:

- a max grade N (the grades range between 0 and N) and a number of subjects n
- a set of accepted students S_A of length $nb_{stu,A}$: $\{u_a \in [0,N]^n\}_{u_a \in S_A}$
- a set of rejected students S_R of length $nb_{stu,R}$: $\{u_r \in [0,N]^n\}_{u_r \in S_R}$
- M a really big constant, ϵ a very small one, $nb_{stu} = nb_{stu,A} + nb_{stu,R}$

With the following variables:

- $w_i \in [0,1],\ b_i \in [0,N]\ \forall i \in [\![1,n]\!]$
- $c_{i,j} \in [0,1],\ \delta_{i,j} \in \{0,1\}\ \forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!]$
- $\lambda \in [0.5,1]$
- $\alpha_j \in \{0,1\}\ \forall j \in [\![1,nb_{stu}]\!]$

Under constraints:

- $$\sum_i w_i = 1 \tag{1}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!]$,
$$c_{i,j} \leq w_i \tag{2}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!]$,
$$c_{i,j} \leq \delta_{i,j} \tag{3}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!]$,
$$c_{i,j} \geq \delta_{i,j} - 1 + w_i \tag{4}$$
- $\forall j \in [\![1,nb_{stu,A}]\!]$,
$$\sum_i c_{i,j} \geq \lambda - M(1-\alpha_j) \tag{6'}$$
- $\forall j \in [\![nb_{stu,A}+1, nb_{stu,A}+nb_{stu,B}]\!]$,
$$\sum_i c_{i,j} + \epsilon \leq \lambda + M(1-\alpha_j) \tag{7'}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!]$,
$$M(\delta_{i,j} - 1) \leq u_{i,j} - b_i \tag{8}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!]$,
$$M\delta_{i,j} + \epsilon \geq u_{i,j} - b_i \tag{9}$$

Find:

$$max \sum_j \alpha_j \tag{10'}$$

In the relaxed formulation, we delete the x_j and y_j vars. We replace the continuous α parameter with a binary α_j for each student j, which will be set to 0 if we do not consider student j when computing the model parameters.

3.3 Multiclass MR-sort

Now, we improve our algorithm so that it classifies students in more than two categories. In fact, the committee asked us to attribute to each student an additional merit (e.g., rejected, accepted, satisfactory, good and very good). Therefore, we now have a set of p classes $C_1, ..., C_p$ and $p-1$ frontiers $b^1, ..., b^{p-1}$ between these classes (where $\forall h \in [\![1, p-1]\!],\ b^h = (b_1^h, ..., b_n^h)$).

Relaxed Inv-MRSort

Considering:

- a max grade N (the grades range between 0 and N) and a number of subjects n
- a set of classes: $C^0 ... C^p$
- sets of students in different categories S: $\forall h \in [\![0,p]\!], S_h = \{u_h \in [0,N]^n\}_{u_h \ affected \ in \ C^h}$
- number of students: $\forall h \in [\![0,p]\!], nb_{stu,h} = |S_h|;\ nb_{stu} = \sum_h nb_{stu,h}$
- M a really big constant, ϵ a very small one

With the following variables:

- $w_i \in [0,1]\ \forall i \in [\![1,n]\!]$
- $b_{i,h} \in [0,N]\ \forall i \in [\![1,n]\!], \forall h \in [\![1,p]\!]$
- $c_{i,j,h} \in [0,1],\ \delta_{i,j,h} \in \{0,1\}\ \forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!], \forall h \in [\![0,p]\!]$
- $lambda \in [0.5,1]$
- $\alpha_{j,h} \in \{0,1\}\ \forall j \in [\![1,nb_{stu}]\!], \forall h \in [\![0,p]\!]$

Under constraints:

- $$\sum_i w_i = 1 \tag{1}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!], \forall h \in [\![0,p]\!],$
$$c_{i,j,h} \leq w_i \tag{2"}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!], \forall h \in [\![0,p]\!],$
$$c_{i,j,h} \leq \delta_{i,j,h} \tag{3"}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!], \forall h \in [\![0,p]\!],$
$$c_{i,j,h} \geq \delta_{i,j,h} - 1 + w_i \tag{4"}$$
- $\forall h \leq h' \in [\![0,p]\!], \forall j \in [\![nb_{stu,h'-1}, nb_{stu,h'}]\!],$
$$\sum_i c_{i,j,h} \geq \lambda + M(1 - \alpha_{j,h}) \tag{6"}$$
- $\forall h' > h \in [\![0,p]\!], \forall j \in [\![nb_{stu,h'-1}, nb_{stu,h'}]\!],$
$$\sum_i c_{i,j,h} + M\alpha_{j,h} + \epsilon \leq \lambda \tag{7"}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!], \forall h \in [\![1,p]\!],$
$$M(\delta_{i,j,h} - 1) \leq u_{i,j} - b_{i,h} \tag{8"}$$
- $\forall i \in [\![1,n]\!], \forall j \in [\![1,nb_{stu}]\!], \forall h \in [\![1,p]\!],$
$$M\delta_{i,j,h} \geq \epsilon + u_{i,j} - b_{i,h} \tag{9"}$$

Find:

$$max \sum_{j,h} \alpha_{j,h} \tag{10'}$$

Here, we create c, δ and α variables for each class h. Apart from that, equations remain (almost) the same.

4 NCS

To solve the Inv-NCS problem, we used a SAT solver (gophersat).

4.1 SAT

We first implemented the basic (unweighted) SAT method. With this formulation, we must then take in denoised data as the formula must be valid for the solver to work.

Inv-NCS

Considering:

- a set of classes: $C^0 ... C^p$
- a max grade N (the grades range between 0 and N)
- a number of subjects n
- sets of students affected to different categories: $\forall h \in [\![0, p]\!], U_h = \{u_h \in [\![0, N]\!]^n\}_{u_h \ affected \ in \ C^h}$

With the following boolean variables:

- $x_{i,h,k}$ $\forall i \in [\![1, n]\!], \forall h \in [\![1, p]\!], k \in [\![0, N]\!]$
- y_B $\forall B \subset [\![1, n]\!]$

Under constraints:

- $\forall i \in [\![1, n]\!], \forall h \in [\![0, p]\!], \forall k < k' \in [\![0, N]\!]$,

$$x_{i,h,k} \vee \neg x_{i,h,k'} \tag{1}$$

- $\forall i \in [\![1, n]\!], \forall h < h' \in [\![0, p]\!], \forall k \in [\![0, N]\!]$,

$$x_{i,h,k} \vee \neg x_{i,h',k} \tag{2}$$

- $\forall B \subset B' \subset [\![1, n]\!]$,

$$y_{B'} \vee \neg y_B \tag{3}$$

- $\forall B \subset [\![1, n]\!], \forall h \in [\![0, p-1]\!], \forall u_{h-1} \in U_{h-1}$,

$$(\bigvee_{i \in B} x_{i,h,u_{h-1}}) \vee \neg y_B \tag{4}$$

- $\forall B \subset [\![1, n]\!], \forall h \in [\![1, p]\!], \forall u_h \in U_h$,

$$(\bigvee_{i \in B} x_{i,h,u_h}) \vee y_{[\![1,N]\!] \setminus B} \tag{5}$$

Find correct values for the x and y variables

We introduce here two kinds of boolean variables:

- $x_{i,h,k}$, which tell us if the grade k for subject i is above border of category h

- y_B, which tell us if B is a sufficient coalition

Hence, the equations 1 to 3 are constraint on the x and y variables:

- The first one means that a student in class C^h is also in class C^{h-1}
- The second one means that a if k in a correct grade, then $k' > k$ is also a correct grade
- The third one means that if B is a correct coalition, any $B' \subset B$ is also a correct one

The last two equations use the given students in each category to determine the borders and the sufficient coalitions:

- The fourth one means that the correct grades of a student in class C^{h-1} for class C^h aren't a sufficient coalition
- The fifth one means that the correct grades of a student in class C^h for class C^h are a sufficient coalition

Once the solver has found a solution, it is easy to find the borders and the sufficient coalitions.

- To find the border of subject i between class C^{h-1} and C^h, you only need to find the minimum k for which $x_{i,h,k}$ is true.
- To find the sufficient coalitions, you only need to find the $B \subset [\![1, n]\!]$ for which y_B is true.

4.2 Max-SAT

With the max-sat formulation, we can ease the model constraints and allow noisy data. Indeed, the solver will now try to define the borders and correct coalitions by taking the largest subset of students able to validate the model. A gaussian noise $\mu = 0$ & $\sigma = 0.1$ is added to the grades for the testing.

Relaxed Inv-NCS

Considering:

- a set of classes: $C^0 ... C^p$
- a max grade N (the grades range between 0 and N)
- a number of subjects n
- sets of students affected to different categories: $\forall h \in [\![0,p]\!], U_h = \{u_h \in [\![0,N]\!]^n\}_{u_h \ affected \ in \ C^h}$

With the following boolean variables:

- $x_{i,h,k}$ $\forall i \in [\![1,n]\!], \forall h \in [\![0,p]\!], k \in [\![0,N]\!]$
- y_B $\forall B \subset [\![1,n]\!]$
- $z_{h,u}$ $\forall h \in [\![0,p]\!], \forall u \in U_h$

Under constraints:

- $\forall i \in [\![1,n]\!], \forall h \in [\![0,p]\!], \forall k < k' \in [\![0,N]\!]$,

$$x_{i,h,k} \vee \neg x_{i,h,k'} \tag{1}$$

- $\forall i \in [\![1,n]\!], \forall h < h' \in [\![0,p]\!], \forall k \in [\![0,N]\!]$,

$$x_{i,h,k} \vee \neg x_{i,h',k} \tag{2}$$

- $\forall B \subset B' \subset [\![1,n]\!]$,

$$y_{B'} \vee \neg y_B \tag{3}$$

- $\forall B \subset [\![1,n]\!], \forall h \in [\![0,p-1]\!], \forall u_{h-1} \in U_{h-1}$,

$$(\bigvee_{i \in B} x_{i,h,u_{h-1}}) \vee \neg y_B \vee \neg z_{h-1,u_{h-1}} \tag{4'}$$

- $\forall B \subset [\![1,n]\!], \forall h \in [\![1,p]\!], \forall u_h \in U_h$,

$$(\bigvee_{i \in B} x_{i,h,u_h}) \vee y_{[\![1,N]\!] \setminus B} \vee \neg z_{h,u_h} \tag{5'}$$

Maximize

$$\bigwedge_{h \in [\![0,p]\!], u \in U_h} z_{h,u} \tag{6}$$

The z constraints added to let the model remove some of the students. Then both the fourth and the fifth equations have been modified, while the first three (representing the constraints on our system) remain the same.

The last equation indicates that the solver has to keep the largest number of students being effectively treated. We can model that with a weighted SAT solver, by putting small weights on the z variables and big enough weights on the other ones so we make sure that all the z variables will be set to false before a x or a y one is modified.

4.3 SAT for NCS with intervals

We can also solve the NCS problem with interval, by modifying a little bit the equations.

Relaxed Inv-NCS with intervals

Considering:

- a set of classes: $C^0 ... C^p$
- a max grade N (the grades range between 0 and N)
- a number of subjects n
- sets of students affected to different categories: $\forall h \in [\![0,p]\!], U_h = \{u_h \in [\![0,N]\!]^n\}_{u_h \ affected \ in \ C^h}$

With the following boolean variables:

- $x_{i,h,k}$ $\forall i \in [\![1,n]\!], \forall h \in [\![0,p]\!], k \in [\![0,N]\!]$
- y_B $\forall B \subset [\![1,n]\!]$
- $z_{h,u}$ $\forall h \in [\![0,p]\!], \forall u \in U_h$

Under constraints:

- $\forall i \in [\![1,n]\!], \forall h \in [\![0,p]\!], \forall k < k' < k'' \in [\![0,N]\!],$

$$x_{i,h,k'} \vee \neg x_{i,h,k} \vee \neg x_{i,h,k''} \tag{1"}$$

- $\forall i \in [\![1,n]\!], \forall h < h' \in [\![0,p]\!], \forall k \in [\![0,N]\!],$

$$x_{i,h,k} \vee \neg x_{i,h',k} \tag{2}$$

- $\forall B \subset B' \subset [\![1,n]\!],$

$$y_{B'} \vee \neg y_B \tag{3}$$

- $\forall B \subset [\![1,n]\!], \forall h \in [\![0,p-1]\!], \forall u_{h-1} \in U_{h-1},$

$$(\bigvee_{i \in B} x_{i,h,u_{h-1}}) \vee \neg y_B \vee \neg z_{h-1,u_{h-1}} \tag{4'}$$

- $\forall B \subset [\![1,n]\!], \forall h \in [\![1,p]\!], \forall u_h \in U_h,$

$$(\bigvee_{i \in B} x_{i,h,u_h}) \vee y_{[\![1,N]\!] \setminus B} \vee \neg z_{h,u_h} \tag{5'}$$

Maximize

$$\bigwedge_{h \in [\![0,p]\!], u \in U_h} z_{h,u} \tag{6}$$

Only the first equation is modified here: we had to change the continuity constraint as indicated in the definition of NCS with intervals.

For NCS with intervals, two borders must be found for each class. To get the min (resp. max) constraint for subject i in class C^h, you can find the minimum (resp. maximum) k for which $x_{i,h,k}$ is true.

5 Implementation

We implemented these resolution methods with Python. We also added some unit tests on all our solvers. You can find our code online on Gitlab.

6 Results evaluation

6.1 Methodology

Results on our different methods are evaluated thanks to randomly generated data sets. A random seed is fixed to get the same generated test data set for model comparisons (section 6.6). We use 200 students for each test. We chose a F1-score with unweighted average to evaluate the performances (to avoid influence of imbalanced data on F1-score).

6.2 Comparison of the different implementations of Inv-MRSort

We notice in table 2 that the relaxed binary solver is slower than the rigid version.

Obviously, the relaxed multiclass Inv-MRSort is slower for a number of categories greater than 2. However, it is as efficient as the binary one for 2 classes.

6.3 Comparison of the different implementations of Inv-NCS

As shown in table 1, NCS relaxed is slower than the rigid version. That may be explained by the the use of the max-SAT solver, which must look for the best solution.

NCS with intervals is even slower, especially as the number of students is increasing. Indeed, we modified the first continuity constraint, adding a new loop on the possible grades in the process.

6.4 Impact of parameters on execution time

For the following tests, if not specified, we consider the following parameters for generation:

- For Inv-MRSort:
  - 50 students
  - 3 categories
  - 10 subjects
  - max grade 20
  - $\forall i \in [\![1, n]\!], \forall h \in [\![0, p]\!], b_i^h = h * \frac{1}{nb_{categories}}$
  - $\forall i \in [\![1, n]\!], w_i = \frac{1}{nb_{subjects}}$
  - $\lambda = 0.6$
- For Inv-NCS:
  - 200 students
  - 3 categories
  - 5 subjects

– max grade 20

– $\forall i \in [\![1, n]\!], \forall h \in [\![0, p]\!], b_i^h = h * \frac{1}{nb_{categories}}$

– T generated from $\{\{1, 3\}, \{2, 4\}, \{1, 2, 5\}\}$

It is interesting to look at the impact of parameters in the execution time to identify our methods limitations. Therefore, we have drawn execution time curves for MR-sort and NCS depending on:

- Number of grades
- Number of categories
- Number of students
- Max grade (for NCS)

See figure 1 for **Inv-MRSort**. Curves show that there is a linear relationship between execution time, number of categories and number of students. The relationship becomes exponential with the number of grades.

See figures 2 and 3 for **Inv-NCS**. Curves show that there is a linear relationship between execution time and number of students, and a squared one between execution time and both the number of categories and the max grade. The latter relationship becomes cubic for NCS with intervals. Finally, it is exponential with the number of grades.

We can understand that by looking at the NCS implementation: indeed, as we have to generate the y_B variables for each subset B of the categories, the number of y variables increases exponentially with the number of different grades.

Besides, experiments show that execution time is much higher with linear method than with SAT (here for rigid cases).

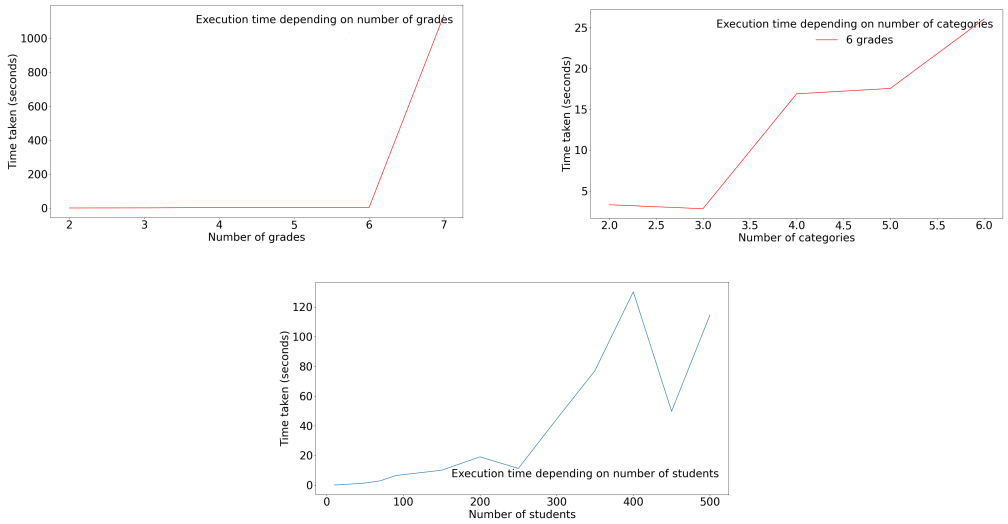


Figure 1: Execution time on linear method with different hyper parameters. We notice that the size of the data set has an important influence on the execution time.

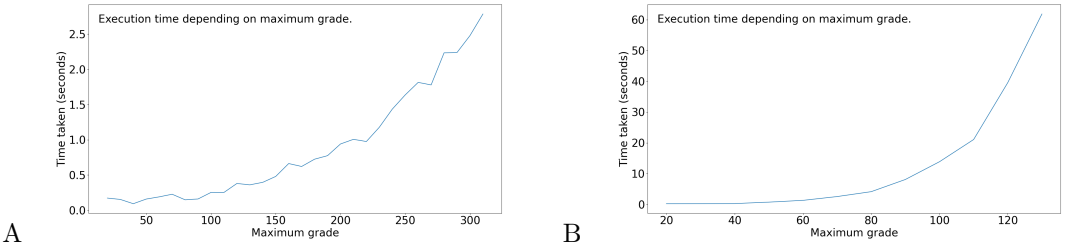


Figure 3: SAT execution time. Figure A represents the execution time depending on the maximum grade used during training while figure B represents the same thing for the NCS problem with the intervals

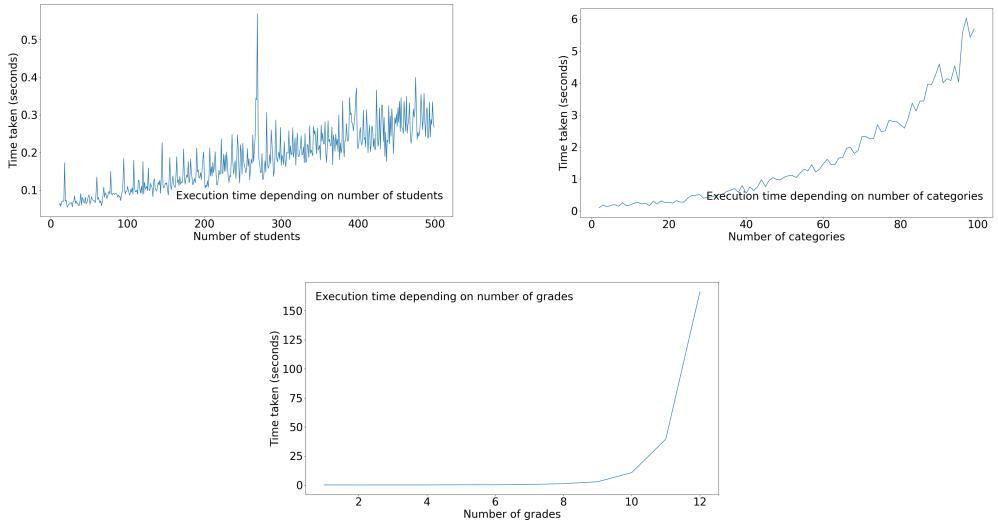


Figure 2: SAT execution time. Figure A represents the execution time depending on the number of students used during training while figure B. represents the execution depending on the maximum grade and C. represents the execution time depending on the number of grades/courses. We can notice that execution time has a linear dependence with the data set size, a square one with the maximum grade and an exponential one with number of grades.

6.5 Impact of parameters on F1-score

For the following tests, if not specified, we consider the following parameters for generation:

- For Inv-MRSort:
  - 3 categories
  - 10 subjects
  - max grade 20
  - $\forall i \in [\![1, n]\!], \forall h \in [\![0, p]\!], b_i^h = h * \frac{1}{nb_{categories}}$
  - $\forall i \in [\![1, n]\!], w_i = \frac{1}{nb_{subjects}}$
  - $\lambda = 0.6$
- For Inv-NCS:
  - 3 categories

- 5 subjects
- max grade 20
- $\forall i \in [\![1, n]\!], \forall h \in [\![0, p]\!], b_i^h = h * \frac{1}{nb_{categories}}$
- T generated from $\{\{1, 3\}, \{2, 4\}, \{1, 2, 5\}\}$

As we saw in the previous section, hyper-parameters have an impact on execution time. Here we are looking at their impact on F1-score. As in every supervised method, a larger quantity of data increases results. Intuitively, the quantity of data needed to train the methods should increase with the number of categories and number of grades/courses. Figures 4 and 5 confirm it.

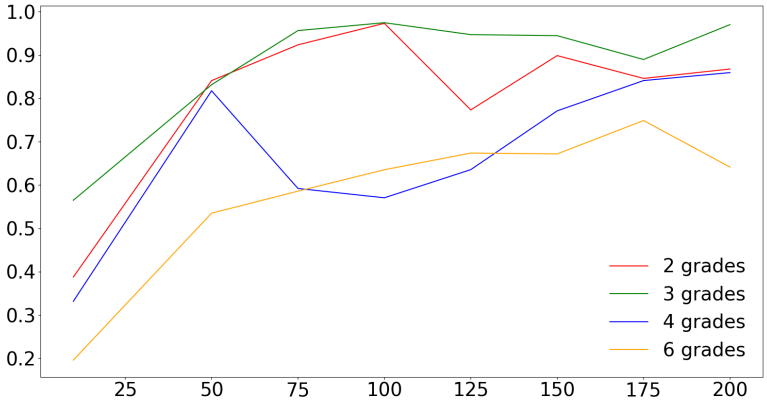
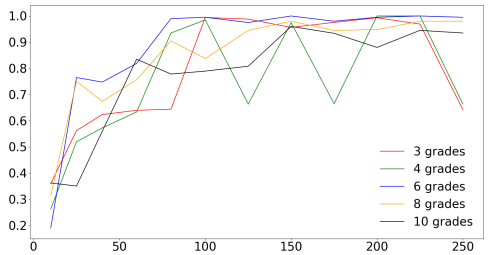


Figure 4: Averaged F1-score with MRSort depending on the number students used for training and number of grades evaluated (different colored lines). First, we can notice that more training data improves F1-score. Also, the quantity training data needed to get good results increases when there are more classes to evaluate. The fluctuations which might be noticed on the graph are probably due to the random data set generations.

A.



B.

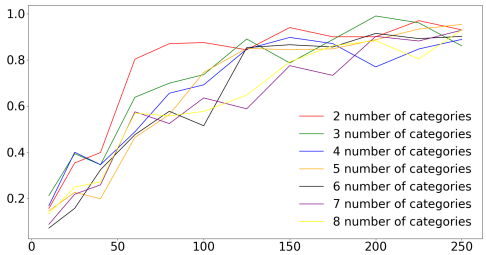


Figure 5: A. Averaged F1-score with NCS depending on the number students used for training and number of grades (different colored lines). B. Averaged F1-score with NCS depending on the number students used for training and number of categories (different colored lines).

6.6 Comparison

We are now going to compare MR-sort with NCS. Both methods have their advantages and drawbacks. As we explained during their formulation, MRSort deals with continuous data while NCS only handles discrete

values. However, as the results in the previous section showed, execution time of NCS is much lower than that of MR-sort. Also, F1-score results on NCS are better than on MR-sort (see 1).

It is also interesting to visualize the confusion matrices (see 6 and 7). One can then note that results are consistent since, even when predicting the wrong category, the methods almost always predict a neighboring category. This means that a student who is wrongly be classified should still be in a category "near the one he deserved".

	Execution time (s)	F1-Score
Multiclass MR-sort	78.9	0.71
NCS (no noise)	0.2	1.0
NCS relaxed	416.9	0.81
NCS intervals	33.2	0.99

Table 1: Results for parameters 5 grades; 200 students; 4 categories; max grade 20; noise 1

	Execution time (s)	F1-Score
Binary MRSort	5.1	0.99
Multiclass MRSort	4.5	0.99
Binary relaxed	13.0	0.83

Table 2: Results for parameters 5 grades; 200 students; 2 categories; max grade 20; noise 1

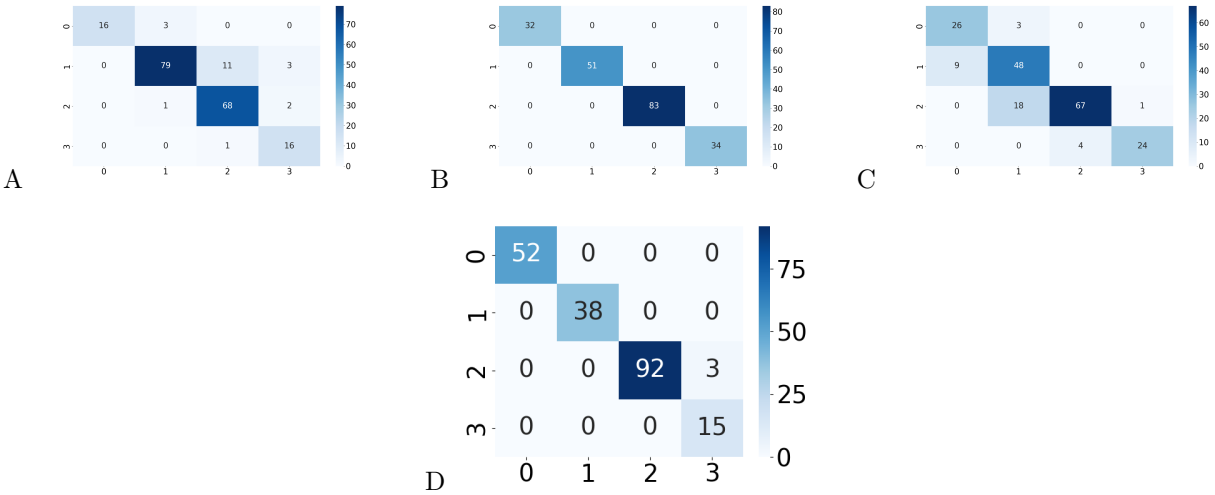


Figure 6: Confusion matrices of MRSort (A), NCS (B), NCS relaxed (C) and NCS relaxed with intervals (D) on 200 test samples trained on 200 train examples, 5 grades and 4 categories.

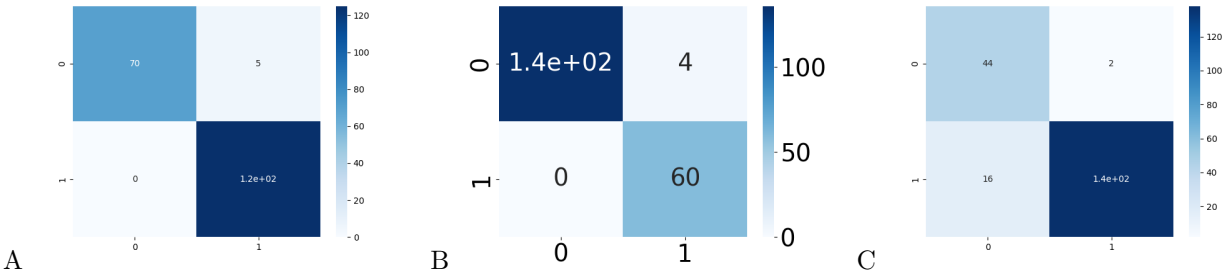


Figure 7: Confusion matrices of MR-sort (A), Binary MR (B), Multiclass MR-sort on 200 test samples trained on 200 train examples, 5 grades and 2 categories.

7 Conclusion

In this project, we defined and implemented inverse methods for two multicriteria models. We used a linear method for one, and a SAT method for the other. We then analyzed their performances.

While the SAT solver if quicker than the linear one and is able to solve more general problems, the linear solver can process continuous values. We must then understand what kind of problem we want to solve before choosing one method or the other.

References

[1] Learning the Parameters of a Multiple Criteria Sorting Method Based on a Majority Rule, *Agnès Leroy and Vincent Mousseau and Marc Pirlot*, **Ecole Centrale Paris, Université de Mons**. 2011.

[2] An efficient SAT formulation for learning multiple criteria non-compensatory sorting rules from examples, *K. Belahcène, C. Labreuche, N. Maudet, V. Mousseau, and W. Ouerdane* , **Ecole CentraleSupélec, Thales Research Technology, Sorbonne Université**. 2018.