CentraleSupélec

MASTER DSBI
DECISION MODELING
PROJECT

SOME DECISION MODELS FOR THE NUTRI-SCORE LABEL OF FOODS

1 What is the Nutri-Score ?

The Nutri-Score is a nutrition label that converts the nutritional value of products into a simple code consisting of 5 letters, each with its own colour. Each product is awarded a score based on a scientific algorithm. This formula takes into account the nutrients to avoid (energy value and the amount of sugars, saturated fats and salt) and the positive ones (the amount of fibre, protein, fruit, vegetables and nuts). **You can therefore see at a glance which products are recommended and which should be avoided**[1].

In France, the Nutri-Score logo (see Figure 1) was elaborated by Santé publique France, a department of the Health Ministry, based on the scientific works of Professor Serge Hercberg (University Paris 13) and the experts of ANSES (Agence nationale de sécurité sanitaire de l'alimentation, de l'environnement et du travail), another department of this ministry.

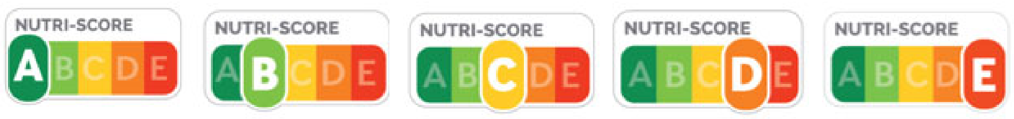


FIGURE 1 – Nutri-score logo

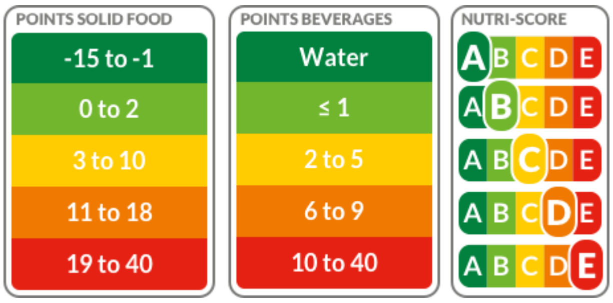
This is how the Nutri-Score is calculated

The algorithm gives points for each element in the nutrition table (per 100 g or ml) - that means bad nutrients (energy, sugars, saturated fatty acids, salt) as well as good nutrients (proteins, fiber, percentage of fruit, vegetables & nuts). We then subtract the positive points from the negative ones and convert the result to the Nutri-Score table (see Figure 2).

Figure 3 below shows how to calculate the Nutri-score of an veggie food with the following information (detailed here `https://fic.colruytgroup.com/productinfo/fr/cogo/2493293` with no fruit/vegetable component) :

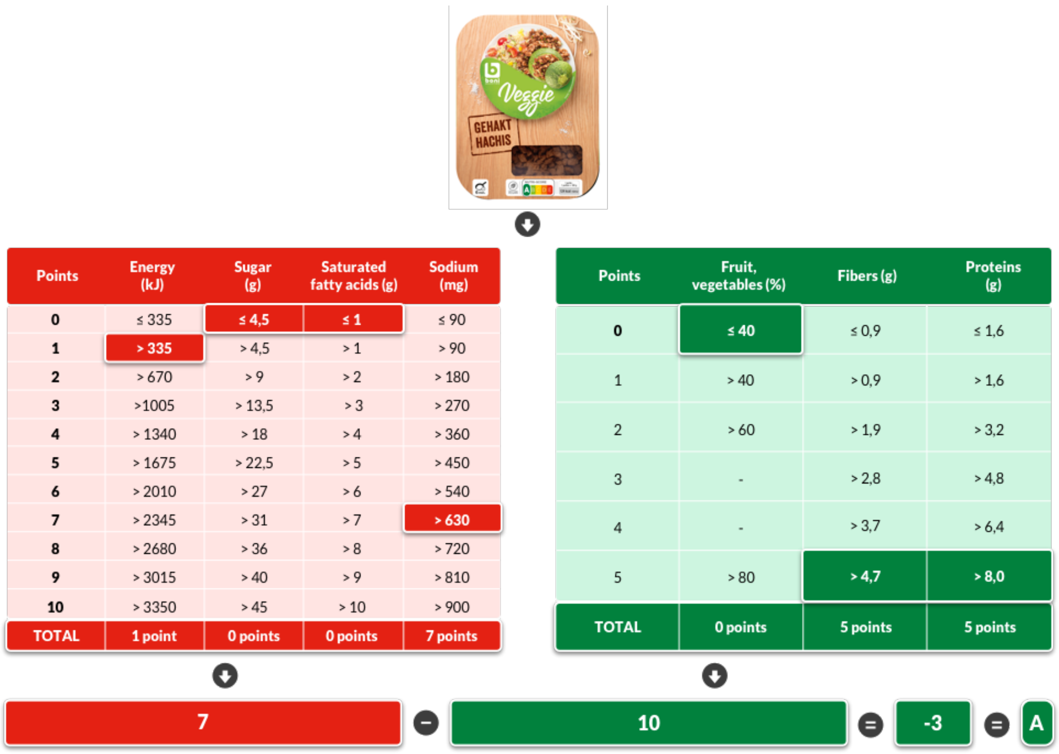
- Energy (KJ) : 485
- Sugars (g) : 0.6
- Saturated fatty acids (g) : 0.1
- Salt (g) : 1.55
- Proteins (g) : 22.4
- Fiber (g) : 6

1. `https://nutriscore.colruytgroup.com/colruytgroup/en/about-nutri-score`



POINTS SOLID FOOD	POINTS BEVERAGES	NUTRI-SCORE
-15 to -1	Water	A
0 to 2	≤ 1	B
3 to 10	2 to 5	C
11 to 18	6 to 9	D
19 to 40	10 to 40	E

FIGURE 2 – Assignment of foods to the classes



Points	Energy (kJ)	Sugar (g)	Saturated fatty acids (g)	Sodium (mg)
0	≤ 335	≤ 4,5	≤ 1	≤ 90
1	> 335	> 4,5	> 1	> 90
2	> 670	> 9	> 2	> 180
3	>1005	> 13,5	> 3	> 270
4	> 1340	> 18	> 4	> 360
5	> 1675	> 22,5	> 5	> 450
6	> 2010	> 27	> 6	> 540
7	> 2345	> 31	> 7	> 630
8	> 2680	> 36	> 8	> 720
9	> 3015	> 40	> 9	> 810
10	> 3350	> 45	> 10	> 900
TOTAL	1 point	0 points	0 points	7 points

Points	Fruit, vegetables (%)	Fibers (g)	Proteins (g)
0	≤ 40	≤ 0,9	≤ 1,6
1	> 40	> 0,9	> 1,6
2	> 60	> 1,9	> 3,2
3	-	> 2,8	> 4,8
4	-	> 3,7	> 6,4
5	> 80	> 4,7	> 8,0
TOTAL	0 points	5 points	5 points

7 − 10 = -3 = A

FIGURE 3 – An example of the calculation of the Nutri-Score

2 The Nutri-Score viewed as a Mutlti-Criteria Decision Aiding problem

In this project, we will consider the calculation of the Nutri-Score of a food as a Multi-Criteria Decision Aiding problem where :

- The set of alternatives X corresponds to the foods analyzed.
- The six criteria (the set N) to take into account are :
  1. **Energy** (KJ) (criterion to be minimized)
  2. **Sugars** (g) (criterion to be minimized)
  3. **Saturated fatty acids** (g) (criterion to be minimized)
  4. **Salt** (g) (criterion to be minimized)
  5. **Proteins** (g) (criterion to be maximized)
  6. **Fiber** (g) (criterion to be maximized)

3 Elaboration of a Database of foods

The inputs of your Python's methods should be based on an Excel or csv file containing at least 100 foods (at least 20 foods per category) and their evaluations on the six criteria. This file could be considered as your database.

The Excel file *openfoodfacts_10000_foods.xlsx* contains the information of more than 9000 foods (essentially French products), but only the Nutri-Score of 5925 of them has been calculated. You can work with the simplified version of this file, named *openfoodfacts_simplified_database.xlsx*.

The information about many foods are also available by following this link : `https://fr-en.openfoodfacts.org/`. In this website, you can choose the country of the products you want to evaluate. For instance, the information about foods of Cameroon are available in `https://cm.openfoodfacts.org/`.

4 Elaboration of a Nutri-Score model based on an additive model

Build a python function `AdditiveNutriScore` returning in an Excel (or csv) file, a score associated to each food (characterized its evaluation on the 6 criteria) given in an Excel or csv file (your database), as a global score coming from an additive model. Your function should be generic, i.e., independent of the nature of foods.

- The inputs of your function contain necessarily the Excel file of the products you have chosen.
- You could set as your reference set, a set of products you have chosen, where the preferences are given by the partial preorder induced by their real Nutri-Score label (see an example given in the file *nutriscore_test_PL.py*).
- The marginal utility function u_i associated to each criterion i will be present in a graphical way.
- You should test your additive model with some other products (more than 100), different with your reference set, and compare the ranking obtained by your additive model with the ranking inferred by the real Nutri-Score.
- Give an analysis of your results. For instance, the real Nutri-Score could be explained by an additive model ?

5 Elaboration of a Nutri-Score model based on a simple sorting (ordered classification) model

Build the functions `PessimisticmajoritySorting` and `OptimisticmajoritySorting`, respectively based on the Pessimistic and Optimistic version of MR-sort rule, which returns an Excel (or csv) file containing the classified foods in the previous five Nutri-Score labels.

- The inputs of your function contain necessarily the Excel file of the products you have chosen.
- You could set manually the profiles used by your model or you could determine them by using an appropriate learning phase that you will detail.
- You could assume that the weights of criteria are equal, i.e., $w_i = 1$ for every criterion i.
- Again, you should test your classification model with some other products (more than 100), different with your reference set, and compare the classification obtained with the one given by the real Nutri-Score (You could use a confusion matrix).
- Give an analysis of your results. For instance, the real Nutri-Score could be explained by a simple sorting model ?

6 Elaboration of a Nutri-Score model based on a machine learning classification model

Build a python function `OtherMethodNutriScore` returning an assignment of each food to a predefined Nutri-score class, by using some machine learning algorithms (decision trees, random forest, ...). Test and compare your results with the real Nutri-Score label of a food.

7 Minimal requirements

1. For this project, each group will be constituted by **one or two students**.
2. Each group will write a report (document in .doc or .pdf) explaining and justifying their results, the parameters chosen, the interpretation of results, etc.
3. Which model you seem comfortable with (among all the models developed in this project) ? Which model is suitable to calculate the Nutri-Score of a food ? Which model is suitable to explain to a consumer(include the original Nutri-Score model) ? Justify all your answers.
4. Send your report and source files by the **19th January 2020, 23h59 (Paris hour)**.

A ELECTRE TRI methods

A.1 Elaboration of the outranking relation $\mathcal{S}_\lambda$

Let A be a set of alternatives evaluated on n real-valued criteria $g_i : A \to \mathbb{R}$, $i \in N = \{1, \ldots, n\}$. We denote by $g_i(a)$ the performance of the alternative a on criterion i. A nonnegative weight w_i is also assigned to each criterion i (w.l.o.g. we suppose $\sum\limits_{i=1}^{n} w_i = 1$).

We associate with each criterion $i \in N$, a nonnegative preference threshold $p_i \geq 0$. If the value $g_i(a) - g_i(b)$ is positive but less than p_i, it is supposed that this difference is not significant, given the way g_i has been built. Hence, on this criterion, the two alternatives should be considered indifferent.

Using this information, we define on each criterion $i \in N$ the partial concordance index $c_i : A \times A \to [0, 1]$ as follows :

$$c_i(a,b) = \begin{cases} 1 \text{ if } g_i(b) - g_i(a) \leq p_i \\ 0 \text{ if } g_i(b) - g_i(a) > p_i \end{cases} \tag{1}$$

The valued relations c_i are aggregated to a single concordance index $c : A \times A \to \mathbb{R}$ by using the following Equation :

$$c(a,b) = \sum_{i=1}^{n} w_i c_i(a,b) \tag{2}$$

The binary relation on A called outranking relation is defined by :

$$a \, \mathcal{S}_\lambda \, b \text{ iff } c(a,b) \geq \lambda \tag{3}$$

where $\lambda \in [0, 1]$ is a cutting level (usually called a threshold and taken above $\frac{1}{2}$).

Interpretation : An alternative $a \in A$ outranks an alternative $b \in A$ if it can be considered at "least as good" as the latter (i.e., a is not worse than b), given the values (performances) of a and b at the n criteria. If a is not worse than b in every criterion, then it is obvious that $a \, \mathcal{S}_\lambda \, b$. However, if there are some criteria where a is worse than b, then a may outrank b or not, depending on the relative importance of those criteria and the differences in the evaluations (small differences might be ignored).

From $\mathcal{S}_\lambda$ we derive the following three binary relations :

☞ "Strictly better than" relation :

$$a \, \mathcal{P}_\lambda \, b \text{ iff } [a \, \mathcal{S}_\lambda \, b \text{ and not}(b \, \mathcal{S}_\lambda \, a)] \tag{4}$$

☞ "Indifferent to" relation :

$$a \, \mathcal{P}_\lambda \, b \text{ iff } [a \, \mathcal{S}_\lambda \, b \text{ and } (b \, \mathcal{S}_\lambda \, a)] \tag{5}$$

☞ "Incomparable to" relation :

$$a \, \mathcal{P}_\lambda \, b \text{ iff } [\text{not}(a \, \mathcal{S}_\lambda \, b) \text{ and not}(b \, \mathcal{S}_\lambda \, a)] \tag{6}$$

A.2 ELECTRE TRI (also called ELECTRE TRI B)

Let us consider r ordered categories $C^1, C^2, \ldots, C^r$, C^1 is the worst one and C^r is the best one. The category C^k is modeled by using limiting profiles. The lower limiting profile of C^k is π^k. The upper limiting profile of C^k is π^{k+1}. We suppose that the limiting profiles are such that π^{k+1} strictly dominates π^k[2]. The profile π^1 (respectively π^{r+1}) is taken low (respectively high). It will be convenient to suppose that $\pi^k \in A$, for each $k = 2, 3, \ldots, r$, while $\pi^1, \pi^{r+1} \notin A$. With this convention we have

2. An alternative a *dominates* an alternative b, we note $a \, \Delta \, b$ iff [for all $i \in N$, $g_i(a) - g_i(b) \geq 0$]. a strictly dominates b if [$a \, \Delta \, b$ and not($b \, \Delta \, a$)]

$$\text{For all } a \in A, a\ \mathcal{P}_\lambda\ \pi^1 \text{ and } \pi^{r+1}\ \mathcal{P}_\lambda\ a. \tag{7}$$

ELECTRE TRI ([9], chap. 6) renamed ELECTRE TRI-B by Almeida-Dias et al. [4] is a MultiCriteria Decision Aid method using limiting profiles. It has two versions called "pessimistic" and "optimistic" in [9]. In [8] the name "pseudo-conjunctive" is used for the "pessimistic" version and "pseudo-disjunctive" for the "optimistic" version. These two versions are defined as follows :

Définition 1 (Pessimistic version : ETRI-B-pc)**.** *Decrease k from $r + 1$ until the first value k such that $a\ \mathcal{S}_\lambda\ \pi^k$. Assign alternative a to C^k.*

ETRI-B-pc assigns an alternative a to the unique category C^k such that a is at least as good as to the lower limiting profile of this category and is not at least as good as its upper limiting profile (the relation "at least as good as" being $\mathcal{S}_\lambda$).

Définition 2 (Optimistic version : ETRI-B-pd)**.** *Increase k from 1 until the first value k such that $\pi^k\ \mathcal{P}_\lambda\ a$. Assign alternative a to C^{k-1}.*

ETRI-B-pd assigns an alternative a to the category C^k such that the upper limiting profile of this category is better than a and the lower limiting profile of this category is not better than a (the relation "better than" being $\mathcal{P}_\lambda$).

Remarque 1. *Roy and Bouyssou ([9],chap.6,pp.393-395) have shown that if $a \in A$ is assigned to the category C^k by the pessimistic version and to the category C^l by the Optimistic version, then $k \leq l$.*

A.3 Majority Rule sorting procedure (MR-Sort)

MR-Sort is a simplified version of the ELECTRE TRI sorting model directly inspired by the work of Bouyssou and Marchant [1, 2] who provide an axiomatic characterization of non-compensatory sorting methods. The general principle of MR-Sort (without veto) is to assign alternatives by comparing their performances to those of profiles delimiting the categories. An alternative is assigned to a category "above" a profile if and only if it is at least as good as the profile on a (weighted) majority of criteria.

The condition for an alternative $a \in A$ to be assigned to a category C^k is expressed as follows :

$$\sum_{i:g_i(a)\geq g_i(\pi^{k-1})} w_i \geq \lambda \text{ and } \sum_{i:g_i(a)\geq g_i(\pi^{k})} w_i < \lambda \tag{8}$$

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

As demonstrated in [6], the problem of learning the parameters of a MR-Sort model on the basis of assignment examples can be formulated as a mixed integer linear program (MILP) but only instances of modest size can be solved in reasonable computing times. The MILP proposed in [6] contains $m \times (2n + 1)$ binary variables, with n, the number of criteria, and m, the number of alternatives. A problem involving 1000 alternatives, 10 criteria and 5 categories requires 21000 binary variables. For a similar program in [3], it is mentioned that problems with less than 400 binary variables can be solved within 90 minutes.

In [5] a genetic algorithm was proposed to learn the parameters of an ELECTRE TRI model. This algorithm could be transposed for learning the parameters of a MR-Sort model. However, it is well known in [7] that genetic algorithms which take the structure of the problem into account to perform crossovers and mutations give better results. It is not the case of the genetic algorithm proposed in [5] since the authors ? definitions of crossover and mutation operators are standard.

Learning only the weights and the majority threshold of an MR-Sort model on the basis of assignment examples can be done using an ordinary linear program (without binary or integer variables). On the contrary, learning profiles evaluations is not possible by linear programming without binary variables. Taking these observations into account, [10] proposes an algorithm that takes advantage of the ease of learning the weights and the majority threshold by a linear program and adjusts the profiles by means of a dedicated heuristic. This algorithm uses the following components :

1. a heuristic for initializing the profiles ;
2. a linear program learning the weights and the majority threshold, given the profiles ;
3. a dedicated heuristic adjusting the profiles, given weights and a majority threshold.

Références

[1] D. Bouyssou and T. Marchant. An axiomatic approach to noncompensatory scoring methods in MCDM, I : The case of two categories. *Eur. J. of Operational Research*, 178 :217–245, 2007.

[2] D. Bouyssou and Th. Marchant. An axiomatic approach to noncompensatory scoring methods in MCDM, II : More than two categories. *Eur. J. of Operational Research*, 178 :246–276, 2007.

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

[4] J. Almeida Dias, J. Rui Figueira, and B. Roy. Electre tri-c : A multiple criteria sorting method based on characteristic reference actions. *European Journal of Operational Research*, 204(3) :565–580, 2010.

[5] M. Doumpos, Y. Marinakis, M. Marinaki, and C. Zopounidis. 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.

[6] A. Leroy, V. Mousseau, and M. Pirlot. Learning the parameters of a multiple criteria sorting method. In Ronen I. Brafman, Fred S. Roberts, and Alexis Tsoukiàs, editors, *Algorithmic Decision Theory - Second International Conference, ADT 2011, Piscataway, NJ, USA, October 26-28, 2011. Proceedings*, volume 6992 of *Lecture Notes in Computer Science*, pages 219–233. Springer, 2011.

[7] M. Pirlot. General local search methods. *European Journal of Operational Research*, 92(3) :493–511, 1996.

[8] B. Roy. Présentation et interprétation de la méthode ELECTRE TRI pour affecter des zones dans des catégories de risque (p. 25). *Document du LAMSADE*, (124), 2002.

[9] B. Roy and D. Bouyssou. *Aide multicritère à la décision : Méthodes et Cas*. Economica, 1993.

[10] O. Sobrie, V. Mousseau, and M. Pirlot. Learning a majority rule model from large sets of assignment examples. In Patrice Perny, Marc Pirlot, and Alexis Tsoukiàs, editors, *Algorithmic Decision Theory - Third International Conference, ADT 2013, Bruxelles, Belgium, November 12-14, 2013, Proceedings*, volume 8176 of *Lecture Notes in Computer Science*, pages 336–350. Springer, 2013.