A Feature Selection Algorithm Based on Heuristic Decomposition

Luís Cavique[1][0000-0002-5590-1493], Armando B. Mendes[2][0000-0003-3049-5852] and Hugo F.M.C. Martiniano[3][0000-0003-2490-8913]

[1] MAS-BioISI, FCUL and Universidade Aberta, Lisboa, Portugal,
[2] Universidade dos Açores, Ponta Delgada, Portugal,
[3] BioISI, FCUL and Instituto Dr. Ricardo Jorge, Lisboa, Portugal,
luis.cavique@uab.pt, armando.b.mendes@uac.pt, hfmartiniano@fc.ul.pt

Abstract. Feature selection is one of the most important concepts in data mining when dimensionality reduction is needed. The performance measures of feature selection encompass predictive accuracy and result comprehensibility. Consistency based feature selection is a significant category of feature selection research that substantially improves the comprehensibility of the result using the parsimony principle. In this work, the feature selection algorithm LAID, Logical Analysis of Inconsistent Data, is applied to large volumes of data. In order to deal with hundreds of thousands of attributes, a problem de-composition strategy associated with a set covering problem formulation is used. The algorithm is applied to artificial datasets with genome-like characteristics of patients with rare diseases.

Keywords: data mining, feature selection, consistency measure, set covering problem, heuristic decomposition.

1. Introduction

In feature selection two main objectives sustain the performance measures, the predictive accuracy and the result comprehensibility [12]. Some models do not take into account the predictive accuracy, and others, after reduction, tend to destroy the underlying semantics of the features. It would be highly desirable to find a theory that could not only reduce the number of features, but also preserve the data semantics.

In this context, Rough Set theory emerges as the desired tool by discovering the data dependencies and reducing the dimension. Rough Set theory was initially proposed as a tool to reason about vagueness and uncertainty in information systems by Pawlak [13] and later it was also proposed for attribute selection [14]. Rough Sets do not correct or exclude data inconsistencies, but rather for each class they determine a lower and an upper approximation.

In parallel, Peter Hammer's group [8], [2], with works in discrete optimization, developed LAD, Logic Analysis of Data. The key features of LAD are the discovery of the minimum number of attributes that are necessary for explaining all observations and the detection of hidden patterns in a dataset with two classes. An extension of the Boolean approach is needed when nominal non-binary attributes are used.

LAD and Rough Set approaches are a subset of filter models that aim to reduce the number of attributes of datasets using two phases: problem transformation and optimization. Their specificity is to keep the semantics of the data by removing only the redundant data.

By combining the two approaches, we proposed Logic Analysis of Inconsistent Data, LAID [5], which blends the best characteristics of both methods. The LAID method should deal with integer attributes associated with costs as in LAD, and be tolerant to inconsistency as in Rough Sets. The integration of both approaches is so close that LAID can be seen as a Rough Set extension.

One key area for application of the method is high-dimensional omics datasets, such as those generated by High-Throughput Sequencing (HTS) technologies. In the past few years, technological developments in this area have resulted in an astonishing growth of the amount of data produced, with a growth rate which doubles approximately every seven months.

The goal of this work is to solve a feature selection problem with a dataset involving two thousand observations and one million attributes. The performance measure is the comprehensibility of the solution, by achieving the minimum number of attributes with the maximum actionable knowledge that better explains the dataset to the final user. Most disease-related omics datasets are subject to restricted access, so artificial datasets with similar characteristics have been created.

This document extends previous works where LAID was presented [4], [5]. The novelty of this work includes the application of LAID to larger datasets and the development of a heuristic decomposition approach to the set covering problem.

This document is organized as follows. In Section 2, we present the related concepts of feature selection and heuristic decomposition. In Section 3, we present a three phase algorithm which finds the feature selection with minimum number of attributes. In Section 4, the computational results are shown. Finally, in the last section we draw some conclusions.

2. Related work

This work combines the areas of attribute selection and problem decomposition. Consequently, in this section we introduce the related topics: feature selection and heuristic decomposition.

2.1 Feature Selection

Given that the time required to double information in the world is substantially decreasing every year, the motivation for feature selection is to reduce the dimensionality of the feature space. Surveys on feature selection methods can be found in [12], [6].

In feature selection, given thousands to millions of features, the goal is to select the most relevant ones. The performance measures have two main objectives, the predictive accuracy and the result comprehensibility. In this work we emphasize the comprehensibility of the results, as in Occam's razor principle that aims to obtain the simplest model, as opposed to the valuation of accuracy metrics.

We consider a dataset D={O, X∪C} where O={O_1,O_2,…,O_n} is a non-empty set of observations (instances or cases), X={X_1,X_2,…,X_m} is a non-empty set of features (attributes or columns) and C is the class attribute.

In this short review we use the feature selection taxonomies reported in [12]. There are three basic models in feature selection: Filter, Wrapper and Hybrid model.

In the Filter model the most popular independent criteria are consistency measures, distance measures, information measures and dependency measures. In this sub-section the consistency measure and the distance measure are detailed because they are re-used in the document.

Using the consistency measure criteria, an inconsistency occurs when two or more observations have the same values in all attributes, but belong to different decision classes. This measure is used in algorithms that attempt to find the minimum number of features with the minimum number of inconsistencies. The well-known algorithm FOCUS [1] uses the concept of consistency, where irrelevant features are removed.

Distance measure criteria are applied in observations within the same class and in observations of diverse classes. In the same class, to obtain the disagreement value, the logic operator XOR is applied, where it returns True if (O_x, X_a) ≠ (Oy, X_a) and False otherwise. With diverse classes, the logic operator XOR can also be applied, where if (O_x, X_a) ≠ (O_y, X_a), feature X_a should be chosen because it differentiates the classes. Comparisons of observations within the same class measure the incoherence or noise of the feature. On the other hand, comparisons of observations between different classes measure how strong a feature is in the discrimination or separation of the classes. RELIEF algorithm [11] evaluates a feature subset based on the subtraction between the distance of observations in different classes and the distance of observations within the same class.

In the incremental or decremental process of feature selection, the information measure is given by the information gain from an added or removed feature.

The dependency measure of a feature is related to how important the correlation is between the feature and the class.

The Filter model is divided into two sequential steps. The feature selection step is executed before the learning phase of the prediction model, and there is no interaction between the selection and the prediction model.

The Wrapper model [9] is also divided into two steps, but with strong interaction between the feature selection step and the learning phase, where the results of the prediction are used as a criterion of feature choice.

Filter models are more intuitive and show better performance since they build the solution in a constructive process without iterations. On the other hand, they present the disadvantages of ignoring the modeling process. Consequently, most of the relevant features might not be adequate in the prediction model and the selection criterion is hard to estimate.

In the Wrapper models, the selection criterion is easy to estimate since the features are chosen by the prediction model, and therefore they are classified as model-aware as they incorporate the knowledge of the predictor. They also present the opposite disadvantages of the Filter models, which are computationally expensive and are less intuitive. The main disadvantage of this method is the increasing overfitting risk when the number of observations is insufficient. In other words, the Wrapper models do not

identify statistical dependency, so the features might not be the most explanatory variables and therefore the model can lose its theoretical basis.

Hybrid methods have been proposed to reduce features in classification, where they try to combine the advantages of the two previous methods. The main disadvantage of the method is the significant low scalability when the feature number increases.

To sum up, in Filter methods, the selection criterion is hard to estimate, whereas Wrapper methods tend to destroy the underlying feature semantics.

In this work, a Filter model is implemented, which combines consistency and distance measures.

2.2 Heuristic Decomposition Approach

In order to solve a problem with hundreds of thousands of attributes, a Decomposition Heuristic should be introduced. In Linear Programming optimization some approaches can be mentioned such as Column Generation, Dantzig-Wolfe decomposition and Benders decomposition. A detailed introduction can be found in [3]. Decomposition is the act of breaking a large problem into sub-problems. Decomposition of a problem is possible if the structure of the problem is maintained. If problem A is separable into A^1, A^2 , …, A^k it can also run in a computer parallel environment, it is parallelizable.

Instead of minimizing the evaluation function f(A) the decomposition method minimizes the sub-problems $f(A^1)$, $f(A^2)$, ..., $f(A^k)$ and finally minimizes the master problem using function g( ). Algorithm 1 presents the parallel version and Algorithm 2 the sequential version.

Algorithm 1: General Heuristic Decomposition (parallel version):
Input: sub-problems $A^{1..max\_k}$
1. solve the sub-problems:
sub-problem 1: y_1=minimize $f(A^1)$
sub-problem 2: y_2=minimize $f(A^2)$
...
sub-problem k: y_k=minimize $f(A^k)$
2. solve the master problem:
master = minimize $g(y_1, y_2,\ldots, y_k)$

The sequential version of the decomposition heuristic, Algorithm 2, at each iteration updates the master problem, adding new information from the solved sub-problem.

Algorithm 2: General Heuristic Decomposition (sequential version):
Input: sub-problems $A^{1..max\_k}$
1. k=0
2. while not end condition
2.2. k=k+1
2.1. solve sub-problem: y_k = minimize $f(A^k)$
2.2. solve master problem: master = minimize g(master, y_k)
3. end while

The decomposition principle can also be exploited in exact methods or in metaheuristic methods [10]. Given the large number of attributes, a heuristic decomposition approach [15] is used in this work.

3. Logical Analysis of Inconsistent Data for Large Datasets

As stated before, in this work we propose a feature selection technique to deal with inconsistent data in large datasets. The inconsistent data is processed using the LAID method, to which we added a decomposition approach to manage large sets of attributes.

In Algorithm 3, we describe the LAID Algorithm with a decomposition technique. Firstly, to remove any inconsistency, we add a dummy binary variable named "je ne sais quoi", 'jnsq', since two observations with the same attributes that belong to different classes work against the consistency measure. In a second step, the Disjoint Matrix Generation $[A_{i,j}]$ and Cost Vector $[c_j]$ are created, where both are based on the definition of the distance measure. Finally, a Decomposition Heuristic for the Set Covering problem is applied. The algorithm can be specified as follows:

Algorithm 3: LAID for Large Datasets
Input: dataset D={O, X∪C} with binary variables
Output: a subset of attributes with minimum cost
1. Check data inconsistencies and add dummy variable 'jnsq'
2. Disjoint Matrix Generation $[A_{i,j}]$ and Cost Vector $[c_j]$
3. Heuristic Decomposition for the Minimum Set Covering Problem

To exemplify the different steps of the algorithm a running example with six features and five observations is used.

3.1. Data inconsistencies

Given the inconsistency of two observations O_a and O_b with the same values in all the attributes and belonging to different classes, a dummy binary variable "je ne sais quoi" [5] is added, assigning a value of 1 whenever the class=1, or:

$$jnsq_a = \begin{cases} 1 \text{ if } (O_a = O_b) \text{ and } \big(\text{class}(O_a) \neq \text{class}(O_b)\big) \text{ and } (\text{class}(O_a) = 1) \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{otherwise} \end{cases}$$

Instead of using the complex approximations of the Rough Sets to keep the data inconsistencies, LAID does not exclude the inconsistency by adding a dummy variable that allows the subsequent application of the other steps of the LAD method.

Given the dataset with six features $\{X_1, \ldots, X_6\}$ and five observations $\{O_1, \ldots, O_5\}$, the first step of LAID is verifying the existence of inconsistencies in the dataset. For each inconsistency the dummy variable 'jnsq' is assigned to 1 in class 1.

In Table 1 the dummy variable 'jnsq' is added in the last feature column in order to avoid data inconsistencies between observation O2 and O3, which present the same values for all the features.

Table 1. Variable 'je ne sais quoi' is added

observation\feature	X_1	X_2	X_3	X_4	X_5	X_6	X_7 (jnsq)	Class
O_1	1	1	0	1	1	1	0	0
O_2	1	0	1	1	1	0	0	0
O_3	1	0	1	1	1	0	1	1
O_4	1	0	1	0	0	1	0	1
O_5	0	0	1	0	0	0	0	1

3.2. Disjoint Matrix Generation $[A_{i,j}]$ and Cost Vector $[c_j]$

In the second step the Cost Vector $[c_j]$ and Disjoint Constraints Matrix Generation $[A_{i,j}]$ are created. Both, the Disjoint Matrix Generation $[A_{i,j}]$ and Cost Vector $[c_j]$ use the concept of distance measure for independent criteria in feature selection.

The Cost Vector $[c_j]$ is obtained by adding the pair comparisons within the same class. This calculus is used to measure the general incoherence, or noise, of the feature j.

On the other hand, the Disjoint Matrix $[A_{i,j}]$ is obtained by using the pair comparisons belonging to different classes. The disjoint matrix $[A_{i,j}]$ for each attribute X_j and pair of observations (O_a, O_b), is defined as:

$$A_{i,j}=\begin{cases}1 & \forall i:\ X_j(O_a)\neq X_j(O_b),\ \text{class}(O_a)\neq\text{class}(O_b),\ (O_a,O_b)\in O\times O\\ 0 & \text{otherwise}\end{cases}$$

The dimension of index *i* in matrix $[A_{i,j}]$ depends on the constraint structure of dataset D. The upper bound of line *i* is (n.(n-1))/2 constraints due to the comparison of pairs of observations (O_a, O_b), where *n* is the number of observations. Each constraint results from the comparison of two different arbitrary observations O_a and O_b that belong to distinct classes. If one attribute *j* is different in the observations O_a and O_b the value of $A_{i,j}$ becomes value 1, denoting that at least one column (attribute) *j* must be maintained in order to differentiate the rows (constraint) *i*.

The cost vector $[c_j]$ and the disjoint matrix $[A_{i,j}]$ will be used as input in the Minimum Set Covering problem, where all constraints (or lines *i*) must be covered, at least once by some attributes.

Table 2 presents pairs of observations of the same class. For each observation the features of the same class should be equal in order to avoid disagreement. The sum of the disagreement of each feature is called cost and stored in the Cost Vector $[c_j]$. As stated before, for observations O_i and attributes X_j, the value of disagreement is given by the operator XOR that returns True if $(O_x, X_a) \neq (O_y, X_a)$ and False otherwise.

Table 2. Cost vector [c_j] shows the disagreement within the same class

Pairs same class	X_1	X_2	X_3	X_4	X_5	X_6	X_7 (jnsq)	Class
O_1,O_2		1	1			1		1
O_3,O_4				1	1	1	1	0
O_3,O_5	1			1	1		1	0
O_4,O_5	1					1		0
cost [c_j]	2	1	1	2	2	3	2	

To obtain the disjoint matrix [$A_{i,j}$], for each observation the features of different classes should be different in order to discriminate. The number of lines generated in [$A_{i,j}$] is N.M, where N and M are the number of observations of class 0 and class 1, respectively. The value k_j corresponds to the number of lines covered for each feature j.

Table 3 presents the pairs of observations that belong to different classes, where the logic operator XOR can also be applied, since, if (O_x, X_a) ≠ (O_y, X_a), the feature X_a should be chosen because it distinguishes the two classes.

Table 3. Matrix [$A_{i,j}$] shows the disjoint between different classes

Pairs different classes	X_1	X_2	X_3	X_4	X_5	X_6	X_7 (jnsq)
1 - O_1,O_3		1	1			1	1
2 - O_1,O_4		1	1	1	1		
3 - O_1,O_5	1	1	1	1	1	1	
4 - O_2,O_3							1
5 - O_2,O_4				1	1	1	
6 - O_2,O_5	1			1	1		
k_j	2	3	3	4	4	3	2

3.3 Heuristic Decomposition for the Minimum Set Covering Problem

In the third step of LAID a decomposition heuristic is applied to solve the Set Covering Problem. The optimization problem that finds the minimum number of columns that cover all the rows is the Set Covering problem and can be defined in binary programming as:

$$\text{minimize } f = \sum c_j . y_j$$

$$\text{subject to } \sum A_{i,j} . y_j \geq 1$$

$$\text{and } y_j \in \{0,1\} \qquad j=1,\ldots,m$$

In this sub-section, a greedy heuristic approach is used to solve the Minimum Set Covering problem, which reuses the algorithm proposed in [7].

In order to solve a problem with thousands of attributes, a Heuristic Decomposition for the Set Covering problem is applied. The matrix $[A_{i,j}]$ is divided into max_k sub-problems resulting in max_k sub-matrices $[A_{i,j}]^k$. Comparing this approach with meta-heuristics the sub-matrices correspond to different neighborhoods in Variable Neighborhood Search or to new regions in the diversification process of Tabu Search.

Algorithm 4 presents the heuristic decomposition for the minimum set covering problem. We consider the following notation:
k – iteration number
S – current solution
RS - repository of feasible solutions

Algorithm 4: Heuristic Decomposition for Set Covering Problem
Input: sub-matrices $[A_{i,j}]^{1..max\_k}$
Output: a subset of attributes with minimum cost
1. k=0, S = ∅
2. while not end condition
2.1. k=k+1
2.2. solve sub-problem: S = S + setCoverHeuristic ($[A_{i,j}]^k$)
2.3. if S is feasible the
2.3.1. RS=RS+S
2.3.2. S = ∅
2.4. end_if
3. end_while
4. solve master problem: master = setCoverHeuristic ($[A_{i,j}]$, [RS])

The algorithm finds multiple feasible solutions in the iterative process, which are combined in the final step by solving the master problem. In this approach we use the same set covering heuristic for the sub-problems and for the master problem. The advantage is to have feasible solutions from the beginning. An alternative would be to select the best columns from the sub-problem and run the set covering heuristic in the master problem.

In this running example $f(c_j, k_j) = c_j/k_j$ and matrix $[A_{i,j}]$ is divided into two sub-matrices A1 and A2 in order to illustrate the decomposition technique. In Table 4 the sub-matrices of matrix $[A_{i,j}]$ and $f(c_j, k_j)$ are shown.

Table 4. Sub-matrices of matrix $[A_{i,j}]$

Features	**X_1**	**X_2**	**X_3**	**X_4**	**X_5**	**X_6**	**X_7 (jnsq)**
c_j	2	1	1	2	2	3	2
k_j	2	3	3	4	4	3	2
c_j/k_j	1	1/3	1/3	1/2	1/2	1	1
$[A_{ij}]^k$	A^1			A^2			

Since we have two sub-matrices A1 and A2 the sub-problem will run twice. In the first iteration from features $\{X_1, X_2, X_3\}$ solution $\{X_2, X_1\}$ is obtained with cost=3. This solution is not admissible since there are 2 uncovered lines, line 4 and line 5.

In the second iteration from features $\{X_2, X_1\}$ from A_1, plus $\{X_4, X_5, X_6, X_7\}$ from A_2, the solution obtained is $\{X_2, X_4, X_7\}$. The current solution is admissible, since it covers all the lines of matrix $[A_{i,j}]$ and has cost=5.

Given the subset $\{X_2, X_4, X_7\}$, in Table 5 the reduced dataset is presented. The number of features was reduced from 7 to 3 and the number of observations was also reduced from 5 to 4. Nearly all the combinations with 2 features are presented, with the exception of X_2=1 and X_4=0 because no observations occur.

Table 5. Reduced dataset

observation\feature	**X_2**	**X_4**	**X_7 (jnsq)**	**Class**
O_1	1	1	0	0
O_2	0	1	0	0
O_3	0	1	1	1
O_4, O_5	0	0	0	1
no observations	1	0	0	?

4. Computational results

To implement the computational results of this algorithm, some choices such as the computational environment, the performance measures and the datasets must be made.

The computer programs were written in C language and the GCC/Dev-C++ compiler was used. The computational results were obtained from an Intel Core Duo CPU 3.0 GHz processor with 4.0 GB of main memory running under the Windows 10 operating system. The INCD (National Infrastructure for Distributed Computing) services was used to run large datasets.

Two performance measures are used to evaluate results, the computational time and solutions quality. The quality is given by the cost of the set covering heuristic, where the minimum cost corresponds to the minimum number of attributes that better explain the dataset.

Since datasets with rare diseases information are subject to restricted access, we generated artificial datasets with similar characteristics. The dataset under study has 1,700 observations in the test group, with class 1, associated with rare disease patients, and 300 observations in the control group, with class 0, with people without the rare disease. In the original dataset the number of attributes is close to one million. The number of lines in the matrix $[A_{i,j}]$ is 510,000 (1,700 x 300) and the final dimension of the matrix is 510,000 x 1,000,000.

4.1. Computational runtime

To study the performance of large matrices the time complexity should be take into account in order to estimate the total runtime.

For a two-class problem the number of observations are given by the tuple (O_0, O_1); (100,100), (500,500), (700,700) and (1700, 300). The number of columns was tested from 100 to 5000 variables.

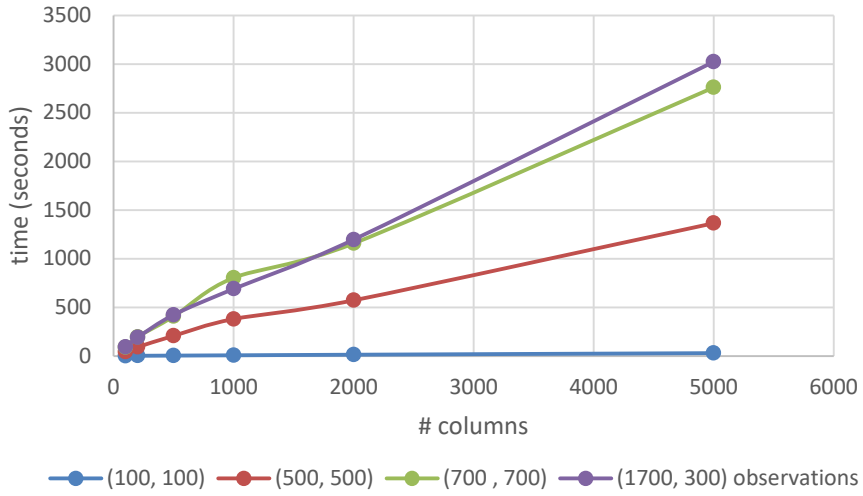

Figure 1. Computational time varying the number of columns and observations

In the Decomposition Heuristic for the Set Covering Problem, for each sub-matrix $[A_{i,j}]^k$, the computational time is shown in Figure 1, where a linear behavior can be found between runtime and the number of columns.

The total runtime for matrix $[A_{i,j}]$ with dimension 510,000 x 1,000,000, and k=200, that is 200 sub-matrices $[A_{i,j}]^k$ of 510,000 x 5,000 each, can be estimated in 570,800 seconds, or approximately seven days. Since it is possible to decompose the problem, it can run in a computer parallel environment.

4.2. Quality of the solutions

In the set covering problem a feasible (or admissible) solution is given when all the lines are covered for at least one chosen column.

The challenge of this work is to solve a feature selection problem with a dataset involving one million attributes. In the computational experiments performed with the artificial dataset with less than 1,000 columns, we verified that the admissibility of the solutions are found in 3 iterations at most, or in 3 sub-matrices $[A_{i,j}]^k$. For datasets with 5,000 columns, feasible solutions are found for each sub-matrix. Running 200 iterations, hundreds of possible good solutions are found and then used in the master step.

In the experiments, we found that each feasible solution includes between 50 to 100 columns, given the low density of the matrix. The minimum cost can be achieved from the set of solved sub-problems. Definitive quality assessment of the solutions must be evaluated by human specialists on rare diseases.

5. Discussion and conclusions

Given the large datasets available in bioinformatics, the aim of this work is to present a feature selection method able to deal with thousands of observations and millions of attributes.

In feature selection two performance measures are considered: the predictive accuracy and result comprehensibility using the parsimony principle. The consistency measure is an important issue in the result comprehensibility. Thus, this document extends the previous work of the LAID algorithm, where the minimum number of attributes that better explain the dataset is found, achieving the maximal actionable knowledge. The novelty of this work includes the application of LAID to larger datasets and the development of a heuristic decomposition for the set covering problem.

In the set covering problem, the disjoint matrix $[A_{i,j}]$ and cost vector $[c_j]$ reuse the distance measure concept for the independent criteria in feature selection. Therefore, the algorithm combines consistency measure and the distance measure approaches. In the Decomposition Heuristic for the Set Covering Problem, the disjoint matrix is separable into sub-matrices $[A_{i,j}]^k$.

The algorithm finds multiple feasible solutions that are gathered in the master phase. The computational runtime for the feature selection with dimension 2,000 observations and 1,000,000 features are satisfactory.

The main conclusion of this study regarding the large volumes of data can be summarized considering two aspects: the algorithm and the data storage. The algorithm requires polynomial time complexity and separation of the problem to be parallelizable in a cloud environment. With respect to data storage, the in-memory access should be replaced by on-disk access to deal with large datasets.

In future work we plan to use data storage in HDF5 format. The Hierarchical Data Format, originally developed at the National Center for Supercomputing Applications, is designed to store and organize large amounts of data that can be accessed on-disk in multiple ways.

Acknowledgements
The first author would like to thank the FCT support UID/Multi/04046/2013. This work used the EGI infrastructure with the support of NCG-INGRID-PT (Portugal) and BIFI (Spain).

References

1. Almuallim, H., Dietterich, T.G.: Learning with many irrelevant features. In: Proceedings of the 9th National Conference on Artificial Intelligence, MIT Press, pp. 547-552 (1991).
2. Boros, E., Hammer, P.L., Ibaraki, T., Kogan, A., Mayoraz, E., Muchnik, I.: An Implementation of Logical Analysis of Data. IEEE Transactions on Knowledge and Data Engineering, vol. 12(2), pp. 292-306 (2000).
3. Boyd, S., Xiao, L., Mutapcic, A., Mattingley, J.: Notes on decomposition methods. Notes for EE364B, Stanford University, pp. 1-36 (2008).

4. Cavique, L., Mendes, A.B., Funk, M.: Logical analysis of inconsistent data (LAID) for a paremiologic study. In: Processing 15th Portuguese Conference on Artificial Inteligence, EPIA (2011).
5. Cavique, L., Mendes, A.B., Funk, M., Santos, J.M.A., A feature selection approach in the study of azorean proverbs. In: Exploring Innovative and Successful Applications of Soft Computing, Advances in Computational Intelligence and Robotics (ACIR) Book Series, IGI Global, pp. 38-58 (2013).
6. Chandrashekar, G., Sahin, F.: A survey on feature selection methods. Computers and Electrical Engineering, vol. 40(1), pp. 16-28 (2014).
7. Chvatal, V.: A greedy heuristic for the set-covering problem. Mathematics of Operations Research, vol. 4, pp. 233-235 (1979).
8. Crama, Y., Hammer, P. L., Ibaraki, T.: Cause-effect relationships and partially defined Boolean functions. Annals of Operations Research, vol. 16, pp. 299-326 (1988).
9. John, G.H., Kohavi, R., Pfleger. K.: Irrelevant Features and the Subset Selection Problem. In: Proceedings of the 11th International Conference on Machine Learning, ICML 94, pp. 121-129 (1994).
10. Joncour, C., Michel. S., Sadykov, R., Sverdlov, D., Vanderbeck, F.: Column generation based primal heuristics. Electronic Notes in Discrete Mathematics, Elsevier, vol. 36, pp. 695–702 (2010).
11. Kira, K., Rendell, L.A.: The feature selection problem: traditional methods and a new algorithm. In: Proceedings of 9th National Conference on Artificial Intelligence, pp. 129-134 (1992).
12. Liu, H., Yu, L.: Toward integrating feature selection algorithms for classification and clustering. IEEE Transactions on Knowledge and Data Engineering, vol.17, vol. 4, pp. 491-502 (2005).
13. Pawlak, Z.: Rough Sets. International Journal of Computer and Information Science, vol. 1, pp. 341-356 (1982).
14. Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishers, Boston (1991).
15. Smet P., Ernst, A., Vanden Berghe, G.: Heuristic decomposition approaches for an integrated task scheduling and personnel rostering problem. Computers and Operations Research, vol. 76, pp. 60-72 (2016).