The Benefits of Using Multi-objectivization for Mining Pittsburgh Partial Classification Rules in Imbalanced and Discrete Data

Julie Jacques
Société ALICANTE
50 Rue Philippe de Girard
59113 Seclin, France
julie.jacques@alicante.fr

Julien Taillard
Société ALICANTE
50 Rue Philippe de Girard
59113 Seclin, France
julien.taillard@alicante.fr

David Delerue
Société ALICANTE
50 Rue Philippe de Girard
59113 Seclin, France
david.delerue@alicante.fr

Laetitia Jourdan
LIFL, Université Lille 1
Bât. M3
59655 Villeneuve d'Ascq cedex, France
laetitia.jourdan@lifl.fr

Clarisse Dhaenens
LIFL, Université Lille 1
Bât. M3
59655 Villeneuve d'Ascq
cedex, France
clarisse.dhaenens@lifl.fr

ABSTRACT

A large number of rule interestingness measures have been used as objectives in multi-objective classification rule mining algorithms. Aggregation or Pareto dominance are commonly used to deal with these multiple objectives. This paper compares these approaches on a partial classification problem over discrete and imbalanced data. After performing a Principal Component Analysis (PCA) to select candidate objectives and find conflictive ones, the two approaches are evaluated. The Pareto dominance-based approach is implemented as a dominance-based local search (DMLS) algorithm using *confidence* and *sensitivity* as objectives, while the other is implemented as a single-objective hill climbing using *F-Measure* as an objective, which combines *confidence* and *sensitivity*. Results shows that the dominance-based approach obtains statistically better results than the single-objective approach.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; G.2.1 [**Discrete Mathematics**]: Combinatorics; I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Heuristic methods*

General Terms

Algorithms

GECCO'13, July 6-10, 2013, Amsterdam, The Netherlands.

Keywords

partial classification, imbalance data, multiobjective

1. MOTIVATIONS

The classification rule mining problem is a Data mining problem that can be seen as a multi-objective optimization problem. A lot of approaches have been proposed and deal with multiple objectives using Pareto dominance. Most of them are detailed in the review of Srinivasan and Ramakrishnan [26]. *Learning classifier systems* (LCS) are another popular approach, very similar to the previous approaches, at the difference they use a credit assignment module to award good individuals and they deal with multi-objective using an aggregation. As a result, the algorithm acts like a single-objective algorithm, where the objective contains an aggregation of the other objectives, generally a sum. Weights are sometimes introduced to balance between objectives available in the aggregation. However adjusting the weights could be difficult, despite some algorithms such as GAssist – a LCS algorithm – dispose of an auto-weighting feature [3]. Intuitively, aggregation will have to deal with the same pitfalls than the single-objective algorithms: they seem more subject to be stuck in local optima and will have a smaller search space, probably resulting in less interesting performance. Knowles *et al.* made a study in that direction, on the *Traveling Salesman Problem*, showing that decomposing one objective into several objectives could enhance results [20]. This process is called *multi-objectivization*. Since decomposition can be difficult or impossible in some problems, Jensen *et al.* introduced the concept of *helper objective*, to allow *multi-objectivization* on more problems [17]. It consists in adding one objective, preferably conflictive with the initial objective, to guide the search and avoid local optima. They evaluated the performance of the *multi-objectivization* over the *Jobshop Scheduling Problem*, where it improved results. Similar studies have been applied to the *Knapsack problem* [15], the *Vehicle Routing Problem* [28] or to *Protein Structure Prediction* [13]; in all these cases *multi-objectivization* improved the results compared to the use of

an aggregation of objectives. In [5], Deb *et al.* decomposed a 3-objective classification problem into a 2-objective classification problem by aggregating 2 of the objectives. The 3-objective approach was more effective than the 2-objective approach. In this paper we study the differences between a single-objective approach and a multi-objective approach.
This paper studies the effects in terms of performance of the *multi-objectivization* on a partial classification problem over discrete and imbalanced data. This work is a part of OPCYCLIN, an industrial project dedicated to optimizing screening of patients for clinical trials, using hospital data. The data at our disposal is binary, imbalanced and is subject to uncertainty – for each patient information only two values are available: "yes" or "unknown" – which is particularly indicated to partial classification. Section 2 first describes the partial classification problem. Since a lot of candidate objectives exist, we explain how to choose the best set of conflicting objectives thanks to a *Principal Component Analysis*.

Then, the modelization as a multi-objective problem is proposed. Section 3 explains the two local search methods we use to compare the single-objective (SO-LS) and the dominance-based multi-objective (DMLS) approaches of the problem. In Section 4 the two approaches are compared over 10 data sets of the literature and the results are discussed. Finally Section 5 gives the conclusions and perspectives for further work.

2. CONTEXT

2.1 Partial classification rule mining

The aim of classification rule mining is to find rules that predict a given fact – called a class – on unknown observations, using their information. As an illustration it could consists in predicting *flu* on unknown patients, using a combination of tests on their symptoms information (*fever?*, *cough?*,...), called *attributes*. In the case of partial classification, the rules predict only one side of the class, for example the positive class (*flu = yes*). In the context of the OPCYCLIN project we have to deal with hospital data containing a lot of binary information (*'yes'* or *'unknown'*) about patients. Up to 10,000 attributes are available but only a few are actually entered for a same patient. Consequently we have to deal with highly imbalance data; a given *diagnosis* can be available on less than 0.4% of patients to at best 20%. Another complication is *uncertainty*: only diagnoses having a consequence on billing are completed. For example, *diabetes* will not be present in the patient billing file if it does not impact the billing of the medical procedure the patient came for. Partial classification is well indicated to deal with this partial data. Once a rule $C \implies P$ is

Table 1: Confusion matrix.

	P	$\overline{P}$	
C	TP	FP	
$\overline{C}$	FN	TN	
			N

obtained, different measures exist to assess the rule effectiveness. Most are based on the confusion matrix given in Table 1. It counts good classifications (*True Positives (TP)* and *True Negatives (TN)*) and wrong classifications (*False Negatives (FN)*) and *False Positives (FP)* made by the rule over a set of known observations. More than 40 measures have been proposed in the literature, Geng and Hamilton indexed most of them [11] while Ohsaki *et al.* focused on measures dedicated to the medical context [22].

2.2 PCA to find conflicting objectives

As a lot of measures exist to evaluate the rule effectiveness, all are potential objectives for multi-objective algorithms. However, Jensen *et al.* showed using too many objectives does not improve results. Moreover, they suggest to choose conflicting objectives: if not, optimizing one objective or more will lead to the same results [17]. Hence, in the following we use a Principal Component Analysis (PCA) to 1) identify a small subset of objectives and 2) make sure these objectives are conflictive.
We selected 12 measures from the literature, based on their ability to deal with imbalanced data. They are presented in Table 2. As regard to F-Measure, we used $\beta = 1$. In addition to these measures, we included three hybrid measures: 1/ *Confidence*$\times$*Sensitivity* since Weiss showed that it is adapted to mine rare rules [29] (CfSe), 2/ a combination of confidence and rule length *Confidence* divided by *RuleLength* (CfRL), to favor simple rules, and 3/ *Sensitivity*$\times$ *Specificity* (SeSp), a measure presented by Carvalho and Freitas [4].

Table 2: Candidate rule quality measures.

Name	Abr.	Formula
Confidence	Cf	$\frac{TP}{TP+FP}$
Conviction	Conv	$\frac{(TP+FP)\times(FP+TN)}{N\times FP}$
Cosine	Cos	$\frac{TP}{\sqrt{(TP+FP)\times(TP+FN)}}$
F-Measure	F1M	$\frac{(1+\beta^2)\times confidence\times sensitivity}{\beta^2\times sensitivity+confidence}$
Laplace	LP	$\frac{TP+1}{TP+FP+2}$
Lift Interest	Lift	$\frac{N\times TP}{(TP+FP)\times(TP+FN)}$
Piatetsky-Shapiro	PS	$\frac{TP}{N} - \frac{TP+FP}{N} \times \frac{TP+FN}{N}$
Sensitivity (True positive rate)	Se	$\frac{TP}{TP+FN}$
Specificity (True negative rate)	Sp	$\frac{TN}{TN+FP}$
Support	S	$\frac{TP}{N}$
Surprise	Sur	$\frac{TP-FP}{TN+FP}$
Uncovered negatives	UN	$\frac{TN}{N}$
Confidence $\times$ Sensitivity	CfSe	$\frac{TP}{TP+FP} \times \frac{TP}{TP+FN}$
Confidence Rule Length	CfRL	$\frac{TP}{(TP+FP)\times ruleLength}$
Sensitivity $\times$ Specificity	SeSp	$\frac{TN}{TN+FP} \times \frac{TP}{TP+FN}$

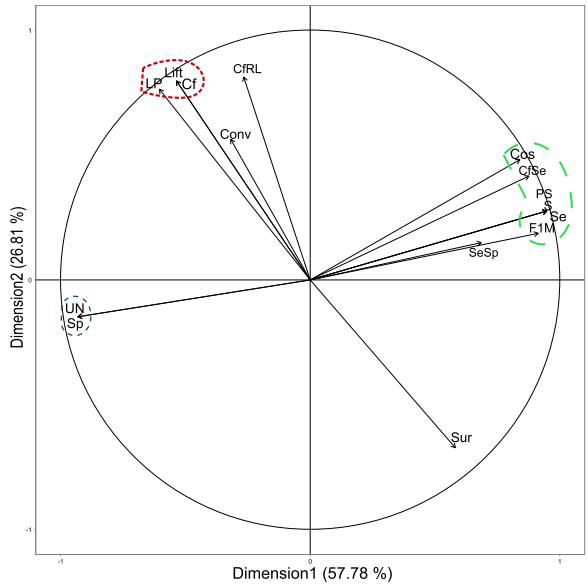

Figure 1: PCA on classification rules with imbalanced data - habermand data set.

2.2.1 Results

This study is realized on three different data sets, to make sure the obtained results are not specific to one data set. These data sets (*habermand*, *yeast3d* and *abalone19d*) have class repartitions between 0.77% and 27.42% and are described further in this paper (see Table 6). The PCA computes correlations between variables by analyzing values of these variables on different observations and then export a correlation map. Therefore we need to generate observations: a population of 1,500 partial classification rules will be generated, predicting the same class, and then the 15 measures under study will be computed for each of them. Since the PCA needs a representative sample group of values for each variable under study, we must ensure there are rules representative of each measure under study. Half of the rules (750) are randomly generated. The other half ($750 = 15 \times 50$) are local optima generated using the single-objective local search algorithm described later in Subsection 3.2. Starting from 50 random solutions, this local search uses successively each of the 15 measures as a single-objective criterion. Thus we obtain for each measure a set of at least 50 rules having a good value. These rules and their 15 associated values are then analyzed using the factoMineR R package [18].

The PCA produces 9 correlation maps: one per data set and for each available projection over 3 axis. Figure 1 is one of them, concerning *habermand* data set and a projection on axis 1 and 2, with a significant inertia (84.59%). The circled parts are common to all PCA results we obtained. *Laplace*, *Surprise* and *Conviction* are less expressed in some PCA results so we will not focus on them. In some data sets like *abalone19d*, *Cosine* is not grouped with *Sensitivity*. 3 groups are highlighted:

- *Confidence* × *Sensitivity (Cfse)*, *Cosine (Cos)*, *Piatetsky-Shapiro (PS)*, *Sensitivity (Se)*, *Sensitivity* × *Specificity (SeSp)*, *Support (S)* and *F-Measure (F1M)* by axis 1
- *Specificity (Sp)*, *Uncovered Negatives (UN)* and *Surprise (Sur)* by axis 1 and 3
- *Confidence (Cf)*, $\frac{confidence}{ruleLength}$ *(CfRL)*, *Conviction (Conv)*, *Laplace (LP)*, *Lift (Lift)* and *Surprise (Sur)* by axis 2 and 3

In the following we will discuss about the major trends found in all the 3 correlation maps.

2.3 Selected measures

The PCA identifies relations between the different measures and we have now to choose which ones can be used as objectives, both in the single-objective and the multi-objective approaches. Indeed, handling too many measures will add complexity and increase computational time. Since the PCA clusters measures under three groups, we choose one measure from each of them (by giving a priority to measure mostly used in literature): *Confidence*, *Sensitivity* and *Specificity*. First experiments showed that maximizing *Confidence* and *Specificity* most of the time leads to similar rules. According to their formula (Table 2), maximizing them minimizes the number of *False Positives*. The PCA does not group *Confidence* and *Specificity*. Indeed a rule having a good *Specificity* can have a bad *Confidence*. However, to the medical domain point of view, rules having bad *Confidence* are not interesting, therefore we drop *Specificity* to mining only rules having both good *Confidence* and *Sensitivity*.

F-Measure allows to find rules having both a good *Confidence* and a good *Sensitivity* and is indicated to evaluate performance in partial classification as, as compared to other measures like error rate, it enforces a better balance between performance on the minority and the majority class, respectively, and, therefore, it is more suitable in the case of imbalanced data. Thus, this is an interesting objective to use in the studied single-objective approach. Regarding the multi-objective approach, *Confidence* and *Sensitivity* are two interesting objectives but first experiments showed they are subject to bloat when used alone: we obtained rules needlessly complicated or too specific such as $age = 62$ and $cough = yes$ *and* $diabetes = yes$ *and* $metformin = yes$ and $fever = yes$ and $muscle\ pain = yes \implies flu$ matching only one patient, where a simpler rule having the same *Confidence* is sufficient: $cough = yes$ and $fever = yes$ and $muscle\ pain = yes \implies flu$.

A widely used solution to overcome bloat is the *Minimum Description Length* principle introduced by Rissanen [25], which is similar to the *Occam's razor*: when two rules are equivalent, the simpler one must be preferred. In application of this principle, we introduce a third objective to promote simpler rule sets: minimizing each rule set count of terms. Finally, in the multi-objective approach, we choose to find rules optimizing the 3 following objectives:

- maximize *Confidence*
- maximize *Sensitivity*
- minimize *Number of Terms*

3. METHODS UNDER STUDY

We show the partial classification rule mining problem could be seen as a multi-objective problem, either using an aggregation with *F-Measure* or by using the three distinct objectives *Confidence*, *Sensitivity* and *Number of Terms*.

This section explains how to solve the problem using local search, first in a single-objective way to deal with the aggregated objectives, and then in a multi-objective way using a dominance-based local search (DMLS). First the common implementations details of the two approaches are presented, such as encoding and neighborhood. Then the two local search methods under study are presented: the single-objective and the dominance-based local search (DMLS) able to deal with multiple objectives.

3.1 Encoding and Neighborhood

A local search algorithm is a meta-heuristic improving a solution - a rule set, a schedule, a path. . . - by visiting similar solutions, until no more improvement can be done. It needs the definition of a neighborhood function that associates to each solution a set of solutions - called *neighbors* - by applying a small modification on it; and the definition of a fitness function, which assesses if a solution is better than another. In this section we will see which encoding and neighborhood we use in the partial classification rule mining problem, and which algorithms are adapted to deal with this problem in a single-objective way and then in a multi-objective way. We use the encoding and the neighborhood used in MOCA-I algorithm detailed in [16]. Each solution is represented using the Pittsburgh encoding, where each solution is a variable-length set of rules. Because of partial classification, each rule predicts the same outcome, preventing inconsistencies in the rule sets. Each rule is a variable-length conjunction of terms, where a term is a test on an attribute (for example: fever = yes). In this paper we focus on binary attributes and ordered list attributes. The neighborhood consists in generating all rule sets having one more or one less term. It also contains rule sets having a difference on one term, where the value of one attribute can be different, like 'age > 0-10' or 'age > 20-30' instead of 'age > 10-20'...

3.2 Single-objective local search (SO-LS)

As single objective local search we chose to use the well-known *Hill Climbing* algorithm. It starts from an initial solution - here a rule set - and visits its neighborhood. When a neighbor with a better fitness than the current solution is found, it continues the search from this neighbor. This algorithm naturally stops when it reaches a local optimum: a solution whose neighborhood does not contain any improving neighbor. As defined previously, we use the maximization of the *F-Measure* as a fitness function. *F-measure* is the harmonic mean of *Confidence* and *Sensitivity* which are the objectives used in the multi-objective local search. It allows obtaining solutions with both high *Confidence* and high *Sensitivity*.

3.3 Dominance-based local search (DMLS)

Dominance-based multi-objective local search (DMLS) algorithms are an adaptation of single-objective local search algorithms to multi-objective problems [21]. They use a dominance relation, like Pareto dominance, to handle each objective separately. Thus, the main difference with single-objective approaches is that they have to cope with a population of compromise solutions, instead of one single solution. Diverse algorithms have been proposed like *Pareto Local Search* [23] or *Pareto Archived Evolution Strategy* [19]. Liefooghe *et al.* proposed a model to unify them [21], which is quickly introduced in the following.

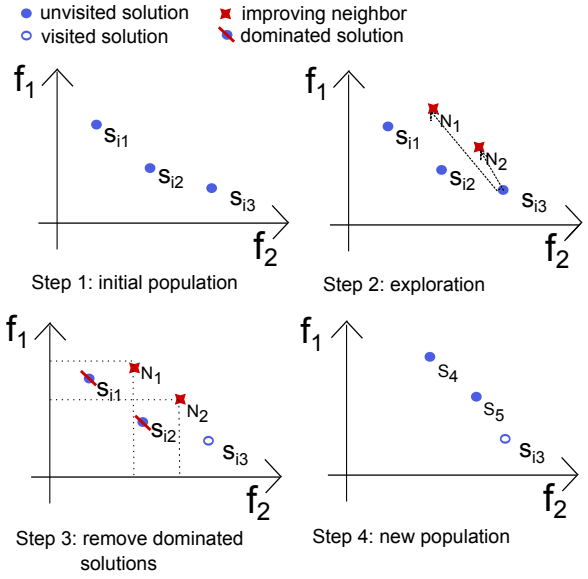

Figure 2: An iteration of DMLS algorithm.

Figure 2 illustrates an iteration of DMLS, where objective functions f_1 and f_2 have to be maximized. DMLS starts with an initial population of non-dominated solutions, in our case a population of random rule sets. The current set of solutions is called *archive* and contains only non-dominated solutions. Before the neighborhood exploration, solutions to explore must be selected. Diverse strategies exist; a simple one consists in selecting one random unvisited solution. In Figure 2, S_{i3} is selected. During the neighborhood exploration, all improving neighbors – better than the solution under exploration – are kept. As in single-objective local searches, several neighborhood exploration strategies may be implemented. As an example, visiting exhaustively all the neighbors and keeping all improving neighbor encountered is one solution. Since the archive contains a lot of solutions to explore, after a complete neighborhood exploration the solution will be labeled as *visited* to avoid multiple explorations. In Figure 2, the exploration of the neighborhood of S_{i3} brought improving neighbors N_1 and N_2. After neighborhood exploration, improving neighbors are added to the archive. Dominated solutions are removed and a new DMLS iteration can be done. In Figure 2, S_{i1} and S_{i2} are removed because they are dominated by new solutions N_1 and N_2. DMLS naturally stops when all solutions of the archive are *visited*. As diverse strategies are available regarding the selection of solutions to explore, and the way to explore the neighborhood, this paper compares each of them with the single-objective approach. Therefore, we consider the following strategies:

Current solution(s) selection (*1* or *)** A simple strategy consists in randomly selecting one unvisited solution to explore from the archive. It will be referred as *1* in the following of this paper. The other strategy is an *exhaustive exploration*, referred as ***, and consists in exploring all unvisited solutions from the archive. Strategy *** needs much more computational time than strategy *1*

Neighborhood exploration ($1_\succ$ or *) Neighborhood exploration can be partial or exhaustive. *First improving* strategy ($1_\succ$) explores neighbors until a neighbor dominating the current solution is found, in opposition to *Exhaustive* strategy (*) where the entire neighborhood is explored. When a dominating neighbor is found with *First improving* strategy, the solution is not labeled as visited and can be further explored in another iteration.

In the next of this paper, each variant of DMLS will be denoted as DMLS (*current set selection* · *neighborhood exploration*). Thus, DMLS ($1 \cdot 1_\succ$) refers to DMLS using *1-random selection* for current set selection and *first improving* as neighborhood exploration strategy. Table 3 summarizes implementation details. First experiments showed performance issues with some data sets, due to an archive containing more than 3,000 solutions. In order to solve this problem, the archive size is bounded to 500 individuals with a simple strategy: if the archive is full, only solutions dominating at least one solution of the archive are accepted. In order to compare with SO-LS we need to obtain one single solution. However, the result of DMLS is an archive of solutions. Therefore we will select the solution of the archive having the best F-measure, that allows us to compare with the solution obtained by SO-LS.

Table 3: Implementation details under study.

Component	Strategy	Abr.
current set selection	1-random selection	1
	exhaustive selection	*
neighborhood exploration	1st improving	$1_\succ$
	exhaustive	*

4. EXPERIMENTATIONS

This section compares the performance of the single-objective approach with those of the multi-objective approaches, using the proper statistical tests. As a reference, we also provide the results of C4.5-CS - a cost-sensitive version of C4.5 which is adapted to imbalance data [27]. The different approaches will be compared on several different data sets.

4.1 Protocol

This work follows the recommendations proposed by Demsar to compare multiple learning algorithms over multiple data sets [6]. These recommendations present a way to evaluate the general performance of an algorithm over several independent data sets or problems, instead of evaluating the performance on a single problem. We will use *Friedman* [9] and *Iman-Davenport* [14] statistical tests to detect differences between multiple algorithms over several problems. These tests are based on ranks obtained by the algorithms over the different problems. Additionally we will use the average ranks to graphically draw the results. Then post-hoc all pairwise comparison of algorithms will be performed using *Wilcoxon* statistical test and *Bergmann and Hommel's* [2] procedure as multiple testing correction, as recommended by García and Herrera [10]. We avoided using parametric statistical tests, since their conditions of use are seldom met in machine learning.

Each algorithm – the 4 DMLS, the single-objective local search (SO-LS) and C4.5-CS – will be evaluated using *F-Measure*. To assess the capacity of the algorithms to deal with unknown data, experiments are realized in 5-fold cross validation: each data set is split in 5 fold and 5 successive runs are executed, each using 4 folds as training data to discover the rules and the last fold as test data to evaluate the quality of the obtained rules. In this paper we provide *F-Measure* obtained both on training and test data to allow detecting *overfitting*; statistical tests are realized on test data only because we want to assess the capacity of the algorithms to deal with unknown data.

Each algorithm will be run 5*5 times per data set: 5 times for each fold, leading to 25 measures for each algorithm and each data set. To make sure to give each DMLS algorithm a chance to reach local optima, each DMLS is given as much run time as DMLS ($*\cdot*$), which is the algorithm needing the highest execution time to reach local optimum. Using time as a stopping criterion allows penalizing DMLS algorithms that spend too much time in archiving instead of exploring new solutions. Since the single-objective local search does not handle an archive, we use the number of neighbor evaluations of DMLS ($* \cdot *$) as a stopping criteria. An algorithm reaching local optima before the allowed time can start again from another initial population until the stopping criteria is reached. Hence, Table 4 shows the average number of restarts performed by each algorithm (DMLS or SO-LS), on each data set. As we can see, DMLS ($*\cdot*$) is not authorized to restart, while using the same number of evaluations the other versions of DMLS are quicker and have the time to reach local optima and restart. Since SO-LS has to deal with only one single solution, it disposes of much more time and can restart more. Table 5 shows the average time spent by each algorithm. We can observe that each DMLS algorithm spent in average the same time on each data set. Since SO-LS has a number of evaluations as a stopping criterion, the average execution time can be different from the DMLS version. Tests are carried out on a computer with a Xeon 3500 quad core and 8 GB ram, under Ubuntu 12, using gcc 4.6.1.

Table 4: Average number of restarts over 25 runs.

	DMLS				SO-LS
	$1 \cdot 1_\succ$	$1 \cdot *$	$* \cdot 1_\succ$	$* \cdot *$	
hab	0.00	0.00	0.88	0.00	258.32
ec1	1.52	1.32	1.16	0.00	128.08
ec2	2.04	1.92	1.28	0.00	89.60
ye3	0.72	0.88	0.92	0.00	272.40
ab9	0.48	0.52	1.00	0.00	690.68
ye2	1.88	1.64	1.20	0.00	140.04
ab1	0.88	0.76	0.96	0.00	765.52
a1a	0.68	0.60	1.16	0.00	329.56
luc	0.76	0.68	1.24	0.00	245.20
w1a	1.04	1.00	1.12	0.00	293.56

4.2 Data sets

To follow the recommendations of Demsar, the tests will be run on a sample of 10 data sets. Most of the data sets we selected come from the UCI repository[1]. In order to obtain imbalanced data sets, multiclass data sets are modified into binary-class data sets, as proposed by

[1]http://archive.ics.uci.edu/ml

Table 5: Average time (seconds) over 25 runs.

	DMLS				SO-LS
	$1 \cdot 1_{\succ}$	$1 \cdot *$	$* \cdot 1_{\succ}$	$* \cdot *$	
hab	23.51	23.54	23.77	23.22	12.59
ec1	12.27	12.27	12.40	12.22	12.11
ec2	5.59	5.60	5.72	5.57	4.99
ye3	130.57	130.37	131.65	130.02	176.21
ab9	125.12	124.95	125.68	124.50	73.23
ye2	5.90	5.90	6.07	5.86	5.10
ab1	532.81	532.86	537.69	531.90	501.90
a1a	319.78	320.16	333.06	318.69	278.25
luc	474.37	474.55	489.64	473.01	671.42
w1a	233.54	233.61	242.66	233.19	199.76

Table 6: Details of the data sets.

name	abr.	#obs.	#att.	I.R.	
habermand	hab	306	3 (3)	0.274	[8]
ecoli1d	ec1	336	7 (7)	0.229	[8]
ecoli2d	ec2	336	7 (7)	0.155	[8]
yeast3d	ye3	1 484	8 (8)	0.104	[8]
abalone9vs18d	ab9	731	8 (7)	0.056	[8]
yeast2vs8d	ye2	482	8 (8)	0.041	[8]
abalone19d	ab1	4 174	8 (7)	0.008	[8]
w1a	w1a	2 477	300 (0)	0.03	[24]
lucap0	luc	2 000	144 (0)	0.278	[12]
a1a	a1a	1 605	123 (0)	0.246	[24]

Fernandez *et al.* [7]. Moreover, data set containing numerical attributes are discretized using Weka (*weka.filters.unsupervised.attribute.Discretize ; bins=10, findNumBins=true*) to allow our algorithm to handle these attributes. Since these data sets contain a small number of attributes, we selected three additional binary imbalanced data sets from literature, having a higher number of attributes: *w1a*, *a1a* [24] and *lucap0* [12]. Table 6 shows the main characteristics of these data sets such as the number of observations (#obs), the number of attributes (#att.) (Including the number of discretized numerical attributes) or *imbalance ratio* (I.R.). An I.R. of 0.77% indicates the positive class is available on 0.77% of the observations; the remaining 99.23% observations match the negative class.

4.3 Benefits of multi-objectivization

Table 7 gives the average of the 25 *F-Measure* values obtained for each algorithm per data set. Additionally, we provide the results of C4.5-CS to give an indication of the results that can be obtained by state-of-the-art algorithms. We experimentally selected C4.5-CS among 10 other state-of-the-art algorithms available in KEEL framework [1], be-

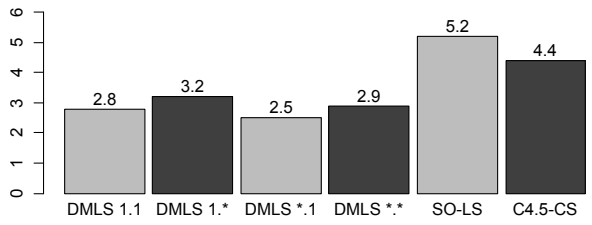

Figure 3: Average ranks on test data.

Table 7: Average F-Measure (training and test) over 25 runs.

	DMLS				SO-LS	C4.5
	$1 \cdot 1_{\succ}$	$1 \cdot *$	$* \cdot 1_{\succ}$	$* \cdot *$		CS
hab	0.621	0.620	0.621	**0.626**	0.592	0.425
	0.411	0.410	0.404	0.382	0.385	0.405
ec1	**0.909**	0.906	0.902	0.902	0.766	0.759
	0.760	0.766	0.773	**0.776**	0.724	0.769
ec2	0.938	**0.939**	0.933	0.934	0.830	0.526
	0.818	0.811	**0.823**	0.807	0.779	0.467
ye3	0.812	0.812	**0.812**	0.812	0.517	0.370
	0.739	0.730	**0.740**	0.734	0.618	0.344
ab9	0.683	0.675	**0.698**	0.693	0.576	0.587
	0.312	0.312	**0.336**	0.333	0.306	0.260
ye2	0.893	**0.900**	0.880	0.886	0.475	0.636
	0.511	0.486	**0.516**	0.508	0.426	0.348
ab1	0.318	**0.321**	0.312	0.304	0.146	0.256
	0.024	**0.034**	0.021	0.026	0.012	0.031
a1a	0.661	0.662	0.653	0.653	0.650	**0.771**
	0.623	0.621	0.626	**0.630**	0.605	0.617
luc	0.840	0.840	0.836	0.836	0.834	**0.945**
	0.816	0.812	0.814	0.817	0.815	**0.825**
w1a	0.691	**0.693**	0.692	0.693	0.604	0.197
	0.492	**0.494**	0.481	0.474	0.443	0.129

cause it gives the best results over the proposed data sets. If we come back to Table 7, we notice that *SO-LS* algorithm is less subject to *overfitting* than *DMLS*. *Overfitting* happens when the algorithm fits too much to the training data set, showing a gap between results on training and test data and problems to deal with data different from the training data set. Despite being more subject to *overfitting* than SO-LS, DMLS gives more interesting results on test *F-Measure*, showing its ability to deal with unknown data. However, none of DMLS algorithms seems to outperform all others. Table 8 completes these results; it gives more details about the *F-Measure* on the test data, by including the standard deviation. On the first data sets, SO-LS has an higher standard deviation of *F-Measure* than the other algorithms, meaning that it is less robust between several runs. Both DMLS and SO-LS approaches seem to have high differences between runs over *ab9*, *ye2* and *w1a* data sets, because they show a high standard deviation. C4.5-CS has a lower standard deviation than DMLS and SO-LS but it is not constraint by local optima that can gives different results. This Table also highlights the results obtained over the test data, since we want to measure the ability of the studied algorithms to deal with new data. We notice that DMLS ($* \cdot 1_{\succ}$) obtains the best results on most of the data sets, but does not outperform all algorithms. SO-LS is always outperformed by at least 1 DMLS implementation. This is confirmed in Figure 3, where the average rank of each algorithm over the 10 data sets is represented. As an example, algorithm DMLS ($* \cdot 1_{\succ}$) has an average rank of: (4+2+1+1+1+1+5+2+5+3)/10=2.5. The algorithm SO-LS has the highest bar with average rank of 5.2, indicating it is outperformed by the others algorithms, both DMLS and C4.5-CS. The different DMLS algorithms seem to have similar performance, since their average ranks are similar.

Table 8: Average and Standard deviation of F-Measure on test data, over 25 runs.

	DMLS								SO-LS		C4.5- CS	
	$1 \cdot 1_{\succ}$		$1 \cdot *$		$* \cdot 1_{\succ}$		$* \cdot *$					
hab	**0.411**	± 0.107	0.410	± 0.118	0.404	± 0.085	0.382	± 0.106	0.385	± 0.137	0.405	± 0.028
ec1	0.760	± 0.065	0.766	± 0.067	0.773	± 0.045	**0.776**	± 0.054	0.724	± 0.145	0.769	± 0.081
ec2	0.818	± 0.071	0.811	± 0.060	**0.823**	± 0.065	0.807	± 0.087	0.779	± 0.176	0.467	± 0.058
ye3	0.739	± 0.047	0.730	± 0.050	**0.740**	± 0.053	0.734	± 0.058	0.618	± 0.267	0.344	± 0.025
ab9	0.312	± 0.209	0.312	± 0.215	**0.336**	± 0.222	0.333	± 0.199	0.306	± 0.175	0.260	± 0.239
ye2	0.511	± 0.181	0.486	± 0.150	**0.516**	± 0.175	0.508	± 0.172	0.426	± 0.285	0.348	± 0.084
ab1	0.024	± 0.049	**0.034**	± 0.067	0.021	± 0.051	0.026	± 0.054	0.012	± 0.041	0.031	± 0.032
a1a	0.623	± 0.019	0.621	± 0.015	0.626	± 0.025	**0.630**	± 0.023	0.605	± 0.024	0.617	± 0.006
luc	0.816	± 0.023	0.812	± 0.023	0.814	± 0.023	0.817	± 0.025	0.815	± 0.025	**0.825**	± 0.024
w1a	0.492	± 0.200	**0.494**	± 0.199	0.481	± 0.194	0.474	± 0.193	0.443	± 0.187	0.129	± 0.031

Table 10: Post-hoc N x N comparison using Bergmann-Hommel's procedure.

		DMLS				SO-LS
		$1 \cdot 1_{\succ}$	$1 \cdot *$	$* \cdot 1_{\succ}$	$* \cdot *$	
DMLS	$1 \cdot 1_{\succ}$	×	≡(0.4795)	≡(0.6714)	≡(0.8875)	≺(0.0019)
	$1 \cdot *$	≡(0.4795)	×	≡(0.2579)	≡(0.5716)	≺(0.0162)
	$* \cdot 1_{\succ}$	≡(0.6714)	≡(0.2579)	×	≡(0.5716)	≺(0.0004)
	$* \cdot *$	≡(0.8875)	≡(0.5716)	≡(0.5716)	×	≺(0.0030)
SO-LS		≻(0.0019)	≻(0.0162)	≻(0.0004)	≻(0.0030)	×

Table 9: Friedman and Iman-Davenport tests with α=0.05.

Test	Crit. Value	Value	Hypothesis
Friedman	11.0704978	16.1142857	REJECTED
Iman-Davenport	2.42208546	4.27993255	REJECTED

In order to determine if one algorithm outperforms the others over the 10 data sets, we applied *Friedman* and *Iman-Davenport* statistical tests. These tests are realized using the average *F-Measure* over the test data sets (Table 8). The *H0* hypothesis is that all algorithms are equivalent over the 10 data sets, regarding their measured average ranks. We obtained the results given in Table 9. The critical values are the values under which ones we can consider H0 is accepted with $\alpha = 0.05$. Since the values obtained by *Friedman* and *Iman-Davenport* tests are higher than their respective critical values, we can reject H0. In other words, it detects that at least one of the algorithms under study outperforms the others. As the null hypothesis is rejected, we can now proceed with a post-hoc test to determine which algorithm outperforms the others. We use *Bergmann and Hommel's* procedure to run the multiple comparisons, it avoids multiple test issues but is more efficient than other methods such as *Bonferroni correction* [10]. Results are available in Table 10 and show for each pair of algorithms the result of the statistical comparison, with the associated p-values. We can observe that SO-LS is outperformed by each version of DMLS. However, the experimental data is not sufficient to reach any conclusion towards which of the DMLS version is better than the others. The same happened with C4.5-CS, this is why we do not display its results. Since SO-LS is outperformed by each DMLS algorithm, it shows that multi-objectivization is more efficient than single-objective search in the context of partial classification rule mining.

5. CONCLUSION AND FURTHER RESEARCH

This paper first described the partial classification rule mining problem, in the context of imbalanced and discrete data. This problem can be solved as a multi-objective problem, either using a multi-objective approach based on Pareto dominance or as a single-objective approach using an aggregation of objectives. In this paper we compared these two approaches within local search algorithm, using a single-objective local search algorithm (SO-LS) and a dominance-based multi-objective local search algorithm (DMLS) respectively for the aggregation approach and the Pareto dominance approach. Since a lot of candidate objectives are available in the literature we first performed a statistical study using a PCA to determine which objectives are the more appropriate. That study highlighted *Confidence* and *Sensitivity* as candidate objectives. Thus, they are used for the DMLS approach. *F-Measure* – combining *Confidence* and *Sensitivity* – is used as an objective in the SO-LS approach. Our statistical comparison over 10 data sets showed that the DMLS approach – multi-objective approach using Pareto Dominance – is more efficient than SO-LS, the single-objective approach using an aggregation. These results show that the multi-objectivization is efficient in the context of partial classification rule mining on imbalanced and discrete data sets. This is consistent with those obtained on other problems, as the *Traveling Salesman Problem*, *Knapsack Problem* or *Vehicle Routing Problem* [20, 15, 28]. Further research could focus on improving the results of some state-of-the-art classification algorithms in Data mining: in a recent study Fernandez *et al.* showed that GAssist - a LCS classifier based on an aggregation - gives the better results than the others state-of-the-art algorithms over 40 data sets [7]. We believe that multi-objectivization could improve these results.

6. REFERENCES

[1] J. Alcalá-Fdez, L. Sánchez, S. García, M. del Jesus, S. Ventura, J. Garrell, J. Otero, C. Romero, J. Bacardit, V. Rivas, J. Fernández, and F. Herrera. Keel: a software tool to assess evolutionary algorithms for data mining problems. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 13:307–318, 2009.

[2] G. H. B. Bergmann. Improvements of general multiple test procedures for redundant systems of hypotheses. *Proc. Symp. on Multiple Hypotheses Testing, Springer, Berlin*, pages 100–115, 1987.

[3] J. Bacardit. *Pittsburgh Genetic-Based Machine Learning in the Data Mining Era: Representations, generalization, and run-time*. PhD thesis, Universitat Ramon Llull Barcelona, 2004.

[4] D. R. Carvalho and A. A. Freitas. A genetic algorithm-based solution for the problem of small disjuncts. In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, PKDD '00, pages 345–352, London, UK, UK, 2000.

[5] K. Deb and A. Raji Reddy. Reliable classification of two-class cancer data using evolutionary algorithms. *Biosystems*, 72(1-2):111–129, 2003.

[6] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, 2006.

[7] A. Fernández, S. Garciá, J. Luengo, E. Bernadó-Mansilla, and F. Herrera. Genetics-based machine learning for rule induction: State of the art, taxonomy, and comparative study. *Evolutionary Computation, IEEE Transactions on*, 14(6):913–941, dec. 2010.

[8] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[9] M. Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200):pp. 675–701, 1937.

[10] S. García, F. Herrera, and J. Shawe-taylor. An extension on 'statistical comparisons of classifiers over multiple data sets' for all pairwise comparisons. *Journal of Machine Learning Research*, pages 2677–2694, 2008.

[11] L. Geng and H. J. Hamilton. Interestingness measures for data mining: A survey. *ACM Computing Surveys (CSUR)*, Volume 38 Issue 3, 2006.

[12] I. Guyon, C. F. Aliferis, G. F. Cooper, A. Elisseeff, J.-P. Pellet, P. Spirtes, and A. R. Statnikov. Design and analysis of the causation and prediction challenge. *Journal of Machine Learning Research - Proceedings Track*, 3:1–33, 2008.

[13] J. Handl, S. C. Lovell, and J. Knowles. Investigations into the effect of multiobjectivization in protein structure prediction. In *Proceedings of the 10th international conference on Parallel Problem Solving from Nature: PPSN X*, pages 702–711, 2008.

[14] R. L. Iman and J. M. Davenport. Approximations of the critical region of the friedman statistic. *Communications in Statistics*, pages 571–595, 1980.

[15] H. Ishibuchi, Y. Nojima, and T. Doi. Comparison between single-objective and multi-objective genetic algorithms: Performance comparison and performance measures. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 1143–1150, 0-0 2006.

[16] J. Jacques, J. Taillard, D. Delerue, L. Jourdan, and C. Dhaenens. MOCA-I: discovering rules and guiding decision maker in the context of partial classification in large and imbalanced datasets. *Learning and Intelligent OptimizatioN, LNCS*, page (in press), 2013.

[17] M. Jensen. Guiding single-objective optimization using multi-objective methods. In *Applications of Evolutionary Computing*, volume 2611 of *LNCS*, pages 268–279. Springer Berlin Heidelberg, 2003.

[18] J. Josse. Factominer : An R package for multivariate analysis. *Journal of Statistical Software*, 25(1):1–18, 2008.

[19] J. Knowles and D. Corne. Approximating the non-dominated front using the pareto archived evolution strategy. *Evolutionary Computation*, 8:149–172, 1999.

[20] J. Knowles, R. Watson, and D. Corne. Reducing local optima in single-objective problems by multi-objectivization. In *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization*, EMO '01, pages 269–283. Springer-Verlag, 2001.

[21] A. Liefooghe, J. Humeau, S. Mesmoudi, L. Jourdan, and E.-G. Talbi. On dominance-based multiobjective local search: design, implementation and experimental analysis on scheduling and traveling salesman problems. *J. of Heuristics*, 18:317–352, 2012.

[22] M. Ohsaki, H. Abe, S. Tsumoto, H. Yokoi, and T. Yamaguchi. Evaluation of rule interestingness measures in medical knowledge discovery in databases. *Artificial Intelligence in Medicine*, 41:177–196, 2007.

[23] L. Paquete, M. Chiarandini, and T. Stützle. Pareto local optimum sets in the biobjective traveling salesman problem: An experimental study. In *Metaheuristics For Multiobjective Optimization, Lecture*, pages 177–200, 2004.

[24] J. C. Platt. Advances in kernel methods. chapter Fast training of support vector machines using sequential minimal optimization, pages 185–208. MIT Press, 1999.

[25] J. Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

[26] S. Srinivasan and S. Ramakrishnan. Evolutionary multi objective optimization for rule mining: a review. *Artificial Intelligence Review*, 36(3):205–248, 2011.

[27] K. Ting. An instance-weighting method to induce cost-sensitive trees. *IEEE Transactions on Knowledge and Data Engineering*, 14(3):659–665, 2002.

[28] S. Watanabe and K. Sakakibara. A multiobjectivization approach for vehicle routing problems. In *Proceedings of the 4th international conference on Evolutionary multi-criterion optimization*, EMO'07, pages 660–672. Springer-Verlag, 2007.

[29] G. M. Weiss. *Timeweaver : a Genetic Algorithm for Identifying Predictive Patterns in Sequences of Events*, volume 1, pages 718–725. Morgan Kaufmann, 1999.