An Effective Strategy for Feature Selection in High-Dimensional Datasets

Mariana Tasca[1], Alexandre Plastino[1], Celso Ribeiro[1], Bianca Zadrozny[2]

[1] Universidade Federal Fluminense, Brazil
{mlobo, plastino, celso}@ic.uff.br
[2] IBM Research, Brazil
biancaz@br.ibm.com

Abstract. Feature subset selection (FSS) is an important preprocessing step for the classification task, specially in the case of datasets with high dimensionality, i.e., thousands of potentially predictive attributes. There is an extensive literature on methods for performing FSS, but most of them do not apply to datasets with high dimensionality because of the prohibitive computational cost. This paper proposes a feature subset selection algorithm which is suitable for datasets with high dimensionality. Our proposal is based on the execution of a constructive procedure followed by a local search strategy, in just one iteration. We conducted experiments using a variety of high-dimensional datasets, showing that the proposed method can reach, in most cases, better accuracies – with a much lower computational cost – than some well-known algorithms.

Categories and Subject Descriptors: I.5.2 [**Pattern Recognition**]: Feature evaluation and selection

Keywords: classification, feature selection, high-dimensional datasets

1. INTRODUCTION

One of the most studied and applied tasks in data mining is the classification task, which aims at estimating the class of an instance based on the available set of attributes. One method to improve the performance of the classification process is to perform a feature subset selection (FSS) procedure, an important step in the data mining process, which aims at choosing a subset of attributes that can represent the relevant information within the data, based on some criteria [Liu and Motoda 1998]. The use of this procedure is strongly recommended, especially if the dataset has a huge dimensionality, because most of the data mining algorithms may require a large computational effort if a huge number of attributes is used. The use of an FSS procedure can provide: (a) improvement in classification performance, eliminating useless attributes and those that can deteriorate the results, (b) simpler classification models, reducing the computational cost of executing these models and providing a better understanding of the obtained results, and (c) smaller datasets to be handled.

Because of the exponential (2^n) search space in terms of the number n of attributes, performing FSS through exhaustive search is intractable. For this reason, several approximation strategies were proposed to solve this problem. FSS algorithms are composed of a search method and a strategy to evaluate the candidate solution [Liu and Yu 2005]. There are a number of different search strategies such as ranker, sequential search, incremental search and metaheuristics, which are reviewed in the next section. The evaluation of candidates can be performed in two ways: the *filter* approach, which uses a relevance measure to estimate the quality of attributes or sets of attributes, and the *wrapper* approach, which estimates the merit of candidates by the accuracy values obtained using a classifier.

We present in this work a feature subset selection algorithm – called Local Search Based (LSB) strategy. LSB combines a construction phase followed by a local search, in only one iteration. Because of the reduced number of evaluations of candidate solutions, this strategy is well-suited to high-dimensional datasets, where some of the most popular FSS methods cannot be applied because of prohibitive computational costs. The information gain of individual attributes is used in the first

phase of LSB to produce a ranking of attributes. Based on this information, an initial candidate solution is generated and, in the next step, its neighborhood is explored in order to find better solutions.

This paper is organized as follows: Section 2 presents previous work on FSS. Section 3 describes the proposed algorithm. Section 4 describes some well-known feature subset selection algorithms that we compare experimentally with our proposal in this work. In Section 5, the experiments conducted over nine datasets are showed, and an analysis of the results is presented. In Section 6, we evaluate a variation of the proposed algorithm, by running many times the single iteration of the basic strategy. Finally, the conclusions from this work and some ideas for the future are discussed in Section 7.

2. PREVIOUS WORK ON FEATURE SELECTION

There are some different strategies in the literature which can be applied to the feature subset selection purpose.

Ranker approaches take into account the individual merit of attributes (with respect to their capacity of identifying the class) to create a ranking of attributes [Blum and Langley 1997; Guyon and Elisseeff 2003]. The first k attributes of the resulting ranking are selected to compose the candidate solution. These algorithms are very fast (linear complexity in terms of dataset dimensionality), but because interactions between attributes are not considered, the quality of candidates may be degraded. Moreover, it may be difficult to select an ideal value for k.

Sequential search algorithms are very simple: at each iteration, the inclusion/exclusion of each attribute is evaluated and those that generate the highest improvement in the solution quality are added/removed. Thus, the complexity of worst case is $O(n^2)$. The most common sequential strategies are *Sequential Forward Selection* (SFS) – which starts with an empty solution and adds attributes one by one – and *Sequential Backward Selection* (SBS) – which starts with all attributes and removes one by one [Kittler 1978].

Incremental search strategies also add one attribute per iteration. However, these algorithms use an initial ranking of the attributes, based on their individual merit. Thus, at each iteration, the attribute at the top of the ranking is selected to be added in the candidate solution, and only this new candidate is evaluated ($O(n)$ complexity) [Ruiz et al. 2006; Bermejo et al. 2010].

Metaheuristics like GRASP [Feo and Resende 1995; Resende and Ribeiro 2014], Tabu Search [Glover and Laguna 1997], Genetic [Goldberg 1989] and Memetic algorithms [Moscato 2003] have been used in the FSS context. For many optimization problems, metaheuristic techniques have proved to be very effective and feasible. However, their computational cost may be extremely high in the context of high-dimensional datasets. Some FSS algorithms that employ metaheuristic approaches can be found in [Yang and Honavar 1998; Inza et al. 2000; Yusta 2009; Esseghir 2010]. Despite the good results achieved, those approaches were applied to low-dimensional datasets (less than 100 attributes).

Some of the most popular FSS methods cannot be applied to high-dimensional datasets because of prohibitive computational costs. For example, for methods that are based on *wrapper* approaches, which require execution of the classifier for each candidate evaluation, it may become infeasible to execute a large number of evaluation steps.

In the last few years, some hybrid algorithms which combine *filter* and *wrapper* approaches have been proposed with the idea of reducing the number of attributes before the *wrapper* evaluation. Some of these approaches can be found in: [Ruiz et al. 2006; Flores et al. 2008], which incrementally explores the attributes by following the ranking obtained by a *filter* measure; [Gutlein et al. 2009], which applies a *wrapper* sequential forward search but only over the first k attributes in the *filter* ranking; [Bermejo et al. 2010; Ruiz et al. 2008], which uses the *filter*-based ranking for a better organization of the search process; [Bermejo et al. 2011], which presents a GRASP with the main

goal of speeding up the FSS process, by reducing the number of *wrapper* evaluations to carry out; [Bermejo et al. 2014], which proposes to embed the classifier into the FSS algorithm instead of using it as a black-box only for evaluating the candidate solutions; and [Moshki et al. 2015], which proposes a GRASP with an extended version of a simulated annealing algorithm for local search. Our strategy also follows a *filter-wrapper* approach in the sense that we use a *filter* in the constructive phase (to rank the attributes and than proceed a pruning on the original list) and we use the *wrapper* strategy to evaluate the candidate solutions (both in construction phase and in the local search).

3. THE PROPOSED ALGORITHM

The proposed heuristic – Local Search Based (LSB) – is a combination of a construction procedure and a local search. As initialization, two steps are performed: (i) the list of attributes E from the dataset is ranked by an individual relevance measure and (ii) the generated ranking is pruned so that only the first k attributes from the ranking (represented by R) are considered in the next phases. The value of k is controlled by a parameter p which defines a percentage of the whole list of attributes. This pruning step is important in the context of high-dimensional datasets, because the evaluation of the whole set of attributes may be impracticable.

The construction phase produces a viable solution S from the pruned ranking R. S is represented by a vector $S[i]$, $1 \leq i \leq |R|$, where if $S[i]$=0, it means that the *i-th* attribute from R does not belong to S; on the other hand, if $S[i]$=1, the *i-th* attribute belongs to the solution S. Then the local search phase tries to improve the quality of S by searching for better neighbors in the $N(S)$ neighborhood. This combination *construction + local search* is executed only once and the final solution is the best neighbor found in the local search procedure. Pseudocode of the proposed algorithm is presented in Figure 1.

In line 01, a ranking E' of the attributes from E is generated. The measure used to evaluate the individual attributes was Information Gain [Quinlan 1993], since it is a well-known measure in the context of feature selection. In line 02, the number of attributes k which will be considered for the algorithm is calculated as $p\%$ of the total number of attributes in the dataset. Line 03 represents the pruning step. R is filled with the first k attributes from the ranked list E'. This step speeds up the algorithm since a reduced number of attributes (k) are considered in the constructive and local search phases.

In line 04, S is initiated with an empty subset. The loop represented in lines 06 to 15 performs the construction of a solution by traversing all the elements of $|R|$.

In line 07, a restricted candidate list (RCL) is generated. The RCL is a list of attributes whose fitness belongs to the range $[max - \alpha * (max - min), max]$, where min and max are the lowest and highest fitness values from R, respectively, and α is a parameter which controls the size of this restricted list. In line 08, one attribute e is randomly selected from RCL to be incorporated, in line 09, in the current solution S. In line 10, the current solution is evaluated by a *wrapper* strategy, using the Naive Bayes classifier, with internal 5-fold cross-validation.

In lines 11 to 13, the fitness of the new solution S' is compared with the fitness of S. If S' outperforms S, it becomes the current solution S. The last step of the iteration is presented in line 14, when the evaluated attribute e is removed from R.

For the local search procedure (lines 16 to 25), the solution S generated by the constructive phase is taken as starting point. A complete iteration of the local search tries to find the best neighbor $S_i \in N(S)$ which outperforms S. A new iteration is performed by taking the best neighbor S_i better than S as the current solution S. The neighborhood $N(S)$ used is made up of all the n subsets $\{S_1, S_2, ...S_n\}$, n=$|R|$, where the *i-th* bit $S_i[i]$, $1 \leq i \leq n$, is inverted. In other words, if $S[i]$=0, then $S_i[i]$=1 and vice versa. This type of neighborhood takes into account insertions (when $S[i]$ is inverted

```
procedure LSB(E, dataset, p, α)

//Initialization
01.  E' ← ranking of attributes from E;
02.  k ← |E|*p/100;
03.  R ← first k attributes from E' //(pruning step);

//Constructive phase
04.  S ← φ;
05.  f(S) ← 0;
06.  while R <> φ do
07.    Generates RCL from R based on α;
08.    e ← randomly selected attribute from RCL;
09.    S' ← {e}∪S;
10.    f(S') ← fitness of solution S';
11.    if f(S') > f(S) then
12.      S ← S';
13.    end if;
14.    R ← R−{e};
15.  end while;

//Local search phase
16.  do
17.  LS-improvement ← false;
18.    for each S_i ∈ N(S) do
19.      if f(S_i) > f(S) then
20.        S ← S_i;
21.        LS-improvement ← true;
22.      end if
23.    end for
24.  while LS-improvement is true;
25.  return S;
end.
```

Fig. 1. Pseudocode of the proposed feature subset selection heuristic

from 0 to 1) and removals (when $S[i]$ is inverted from 1 to 0) of attributes in S. When none of the neighbors $S_i \in N(S)$ presents $f(S_i) > f(S)$, the local search ends and returns, in line 25, the best-fitness solution found.

4. SOME WELL-KNOWN FEATURE SUBSET SELECTION ALGORITHMS

WEKA (Waikato Environment for Knowledge Analysis) [Hall et al. 2009] is a powerful open-source Java-based machine learning workbench. Among the techniques available within WEKA, we selected four feature subset selection algorithms to make a comparison with LSB: *GreedyStepwise* (GS), *BestFirst* (BF), *LinearForwardSelection* (LF) and *SubsetSizeForwardSelection* (SS). The aim of the first experiment is to compare LSB with some well-known available algorithms for feature selection.

The known greedy hill climbing search strategy considers local changes to the current feature subset. Often, a local change is simply the addition or deletion of a single feature from the subset. *GreedyStepwise* is an algorithm which performs a greedy forward or backward search through the space of attribute subsets. It may start with no/all attributes or from an arbitrary point in the space. It stops when the addition/deletion of any remaining attributes results in a decrease in evaluation [Hall et al. 2009].

Best-first [Ginsberg 1994; Russell and Norvig 2003] searches the space of attribute subsets by greedy hill-climbing augmented with a backtracking facility. The idea is to select the most promising candidate generated which has not already been expanded. The backtracking level is controlled by a parameter which defines the number of non-improving candidates allowed. *Best-first* may start with the empty set of attributes and search forward, or start with the full set of attributes and search backward. It is also possible start at any point and search in both directions (by considering all possible single attribute additions and deletions at a given point).

In the classical *Sequential Forward Selection* approach, the number of evaluations grows quadratically with the number of attributes: the number of evaluations in each step is equal to the number of remaining attributes that are not in the currently selected subset. This quadratic growth can be problematic for datasets with a large number of attributes. Trying to mitigate this problem, [Gutlein et al. 2009] propose a technique to reduce the number of attribute expansions in each forward selection step: the *LinearForwardSelection*, which is an extension of *BestFirst*. In this proposal, they limit the number of attributes that are considered in each step so that it does not exceed a certain user-specified constant. This drastically reduces the number of evaluations, and therefore improves the runtime performance of the algorithm.

SubsetSizeForwardSelection is an extension of *LinearForwardSelection*. The algorithm determines the subset size to be reached in forward selection to combat overfitting, where the search is forced to stop at a precomputed subset size. The search performs an interior cross-validation (seed and number of folds can be specified). A *LinearForwardSelection* is performed on each fold to determine the optimal subset size m (using the given SubsetSizeEvaluator). Finally, a *LinearForwardSelection* up to m is performed on the whole data. In [Gutlein et al. 2009], they show that this technique reduces subset size while maintaining comparable accuracy with the *LinearForwardSelection* approach.

5. EXPERIMENTS AND RESULTS

Datasets used in the experiments were obtained from public repositories and have hundreds or thousands of attributes. Table I presents these datasets showing their dimensionality and the number of instances. Datasets are split in 10 folds to enable an external 10-fold cross-validation. Thus, accuracy values for each experiment represent the average of 10 executions of the algorithm for the same dataset.

Since LSB uses a random function during the constructive phase to select an attribute from the RCL, it is necessary to define an initial seed value for each execution. We conducted 10 independent executions of each experiment, with 10 different initial seeds. Thus, the presented values in the next section represent the average of 10 independent executions on each dataset, each of them using a 10-fold cross-validation.

We experimented with a range of different values for the α and p parameters, and the best results were produced when α=0.2 and p=5%. The complete results and analysis of these experiments can be found in [Tasca 2015].

We also tested different parameter combinations for each algorithm selected from the WEKA tool. The combination with the best performance (regarding the solution quality) for each one was used in the comparison with LSB. For *GreedyStepwise*, *BestFirst* and *LinearForwardSelection* algorithms, the best accuracy values were obtained with the default parameters from WEKA. For *SubsetSizeForwardSelection*, the best accuracies were obtained when we performed a ranking using the *wrapper* with Naive Bayes and considered 100 attributes from this ranking.

At first, we analyzed the accuracy values obtained with the Naive Bayes (NB) [Duda et al. 2001] classifier, by submitting the selected subset by each evaluated algorithm. NB is a probabilistic classifier based on the assumption of conditional independence among the predictive attributes given the class.

Table I. High-dimensional datasets used in the experiments

Dataset	# of attributes	# of instances
Leukemia	7130	72
DLBCL	4027	47
Lymphoma	4027	96
Madelon	500	2600
Colon	2001	62
Dexter	19999	600
Lung	12534	181
Prostate	12600	136
Gisette	5000	6000

In spite of this hard independence assumption, NB is a competitive classifier, working quite well in many classification tasks [Fang 2013]. We have chosen Naive Bayes because we would like to compare our algorithm with other proposals in the literature, which also used it.

Table II presents the accuracy values obtained in this experiment. Values in brackets represent the position in the ranking which compares the five algorithms, for each dataset. The best accuracy value in each line is marked in bold. The last row in the table presents the sum of the ranking positions (SRP) for each strategy. Considering that position 1.0 represents the best accuracy for the given dataset and position 5.0 represents the worst result, the optimum value for SRP would be 9.0 (when the algorithm is at the top of the ranking for all datasets) and the worst value would be 45.0 (when the algorithm gets the fifth position for all datasets). LSB presented the best behavior among the evaluated algorithms, as it has the lowest sum of ranking positions (SRP).

Table II. Accuracy values obtained from each evaluated algorithm

Datasets	GS	BF	LF	SS	LSB
Leukemia	88.57 (4.5)	88.57 (4.5)	91.61 (2.0)	90.00 (3.0)	**95.94** (1.0)
Dlbcl	77.50 (5.0)	80.00 (4.0)	88.00 (2.5)	88.00 (2.5)	**89.50** (1.0)
Lymphoma	79.11 (2.0)	78.00 (3.0)	74.89 (4.0)	70.67 (5.0)	**80.25** (1.0)
Madelon	61.27 (2.0)	61.19 (4.0)	60.12 (5.0)	61.23 (3.0)	**61.41** (1.0)
Colon	80.48 (4.5)	80.48 (4.5)	**83.81** (1.0)	83.81 (2.0)	82.62 (3.0)
Dexter	81.33 (5.0)	81.67 (4.0)	84.33 (3.0)	85.17 (2.0)	**88.69** (1.0)
Lung	94.53 (4.5)	94.53 (4.5)	97.25 (2.0)	95.58 (3.0)	**99.13** (1.0)
Prostate	70.55 (5.0)	71.92 (3.0)	73.46 (2.0)	70.60 (4.0)	**81.07** (1.0)
Gisette	**93.83** (1.0)	93.80 (2.0)	88.25 (5.0)	89.03 (4.0)	92.33 (3.0)
Sum of Ranking Positions (SRP)	33.50	33.50	26.50	28.50	13.00

To analyze if the results are statistically significant, we applied the non-parametric *Friedman* test [Friedman 1937], which enables a multi-algorithm multi-dataset comparison. The null-hypothesis for the *Friedman* test is that there are no differences between the algorithms. If the null-hypothesis is rejected, we can conclude that at least two of the algorithms are significantly different from each other, and the *Nemenyi* post-hoc test can be applied to identify these differences [Demšar 2006]. According to the *Nemenyi* test, the performances of two algorithms are significantly different if their corresponding average of ranking positions has a difference of at least a determined critical value.

Using $p\text{-}value$=0.0336, the *Friedman* test execution rejected the null-hypothesis with a significance level of 5%, so the *Nemenyi* test was performed (critical value=1.5634) and detected a significant difference between GS and LSB and also between BF and LSB, which shows that LSB outperforms those algorithms with statistical significance.

We also ranked the computational costs of the strategies based on the CPU time. Table III presents the values obtained (in minutes) and the sum of the ranking positions for each evaluated strategy.

LSB also obtained the best result according to this metric. For the nine evaluated datasets, LSB obtained the first position for seven of them, taking the shortest CPU time to perform the FSS.

Table III. CPU time obtained from each evaluated algorithm

Datasets	GS	BF	LF	SS	LSB
Leukemia	1.419 (4.0)	2.722 (5.0)	0.344 (3.0)	0.306 (2.0)	**0.037** (1.0)
Dlbcl	0.615 (4.0)	1.149 (5.0)	0.134 (2.0)	0.140 (3.0)	**0.015** (1.0)
Lymphoma	3.881 (4.0)	7.925 (5.0)	0.298 (2.0)	0.542 (3.0)	**0.249** (1.0)
Madelon	2.085 (4.0)	7.911 (5.0)	1.232 (3.0)	0.939 (2.0)	**0.074** (1.0)
Colon	0.392 (4.0)	1.184 (5.0)	0.075 (2.0)	0.081 (3.0)	**0.017** (1.0)
Dexter	267.951 (4.0)	431.58 (5.0)	**3.260** (1.0)	4.119 (2.0)	4.941 (3.0)
Lung	3.647 (4.0)	8.905 (5.0)	1.149 (3.0)	0.982 (2.0)	**0.146** (1.0)
Prostate	10.726 (4.0)	28.604 (5.0)	1.036 (2.0)	1.044 (3.0)	**0.500** (1.0)
Gisette	504.668 (4.0)	808.300 (5.0)	3.575 (2.0)	**3.243** (1.0)	13.286 (3.0)
Sum of Ranking Positions (SRP)	36.00	45.00	20.00	21.00	13.00

With respect to the size of selected subsets, LSB proved to be very effective in reducing the datasets dimensionality, in the same manner as the other evaluated algorithms. The selected subsets represent, on average, 0.43% of all attributes. GS, BF, LF and SS generated, respectively, solutions with an average size of 0.28%, 0.37%, 0.49% and 0.19% of all attributes. Table IV presents the size of the solution generated by each strategy, as a percentage of the total number of attributes, for each dataset, for each dataset.

Table IV. Size of generated solutions, in percentage

Datasets	Total # of Attributes	GS (%)	BF (%)	LF (%)	SS (%)	LSB (%)
Leukemia	7129	0.04	0.04	0.05	0.05	0.09
Dlbcl	4026	0.08	0.09	0.12	0.10	0.15
Lymphoma	4026	0.17	0.19	0.21	0.20	0.41
Madelon	500	1.30	1.92	3.32	0.96	1.40
Colon	2000	0.24	0.35	0.34	0.19	0.39
Dexter	19999	0.11	0.16	0.10	0.07	0.18
Lung	12533	0.02	0.02	0.02	0.02	0.04
Prostate	12600	0.05	0.07	0.06	0.05	0.14
Gisette	5000	0.46	0.52	0.18	0.11	1.05
Mean (%)		**0.28**	**0.37**	**0.49**	**0.19**	**0.43**

Besides the algorithms from the WEKA tool, we also compared our proposal with the GRASP presented in [Bermejo et al. 2011] (identified here as HC*). HC* was compared in the referred work with the state-of-the-art feature subset selection algorithms and the results show that this proposal is comparable in accuracy and cardinality of the selected subset to important algorithms, but requires significantly less computational cost.

At first, we analyzed the accuracy values obtained from the Naive Bayes classifier, by submitting the selected subset obtained with each evaluated algorithm (LSB and HC*). Accuracy values of HC* were obtained from [Bermejo et al. 2011], because the source code of HC* was not available and we could not execute it under the same conditions as LSB. The experiment for the LSB algorithm was performed using the same datasets and the same 10 folds than those used in [Bermejo et al. 2011]. Table V shows the results for the two evaluated algorithms over the nine datasets. The best accuracy value in each line is marked in bold.

To analyze if the results are statistically significant, we applied the *Wilcoxon* test [Wilcoxon 1945], which is a non-parametric test used to determine whether two dependent groups of data are different.

Table V. Accuracy values obtained from HC* and LSB

Datasets	HC*	LSB
Leukemia	92.64	**95.94**
Dlbcl	86.17	**89.50**
Lymphoma	74.90	**80.25**
Madelon	60.85	**61.41**
Colon	81.13	**82.62**
Dexter	83.47	**88.69**
Lung	95.69	**99.13**
Prostate	77.87	**81.07**
Gisette	**93.06**	92.33

The null-hypothesis for the *Wilcoxon* test is that the two groups of data are not different. Based on the W statistic, which is calculated from the data, we determine whether to accept or reject the null-hypothesis.

The *Wilcoxon* test execution for the evaluated algorithms rejected the null-hypothesis, because the critical W for 5% level is 6, and the W statistic obtained was 2. This means that LSB outperforms HC* with statistical significance.

The comparison between LSB and HC* in terms of computational cost was made through the number of wrapper evaluations, because we could not run the HC*, for the reason mentioned before, to register the CPU time for both algorithms under the same conditions. We know that the highest cost step in both algorithms is the wrapper evaluation. So algorithms which execute fewer wrapper evaluations have, consequently, lower computational cost.

The comparison between LSB and HC* is showed in Table VI. LSB presents lower computational cost (in terms of number of wrapper evaluations) in seven out of nine datasets. For Dexter and Prostate datasets, HC* was faster. The reason for the different behavior is related to the pruning step. Both of them are based on an initial ranking and only the first k attributes are considered on the construction and local search phases. However, HC* considers a fixed number (100), no matter the dataset dimensionality. But LSB uses a percentage of attributes, set as 5% for these experiments. Thus, since Dexter and Prostate datasets have a very high dimensionality (19999 and 12600 attributes, respectively), the number of candidates to be evaluated for LSB was much higher than for HC* algorithm.

Table VI. Number of wrapper evaluations for HC* and LSB

Datasets	HC*	LSB	% of LSB with respect to HC*
Leukemia	5472.00	1513.60	28%
DLBCL	5206.10	890.30	17%
Lymphoma	5608.60	1467.20	26%
Madelon	5076.80	205.90	4%
Colon	5065.20	509.80	10%
Dexter	5543.20	7063.00	127%
Lung	5940.10	2585.20	44%
Prostate	5252.40	6825.90	130%
Gisette	7206.20	6767.50	94%

With respect to the size of selected subsets, both of the strategies showed to be very effective to reduce the datasets dimensionality. HC* generated solutions with an average size of 0.26% of all attributes while LSB produced solutions with an average size of 0.43%, as we already mentioned before.

6. A MULTI-ITERATION OF LSB

Starting from the basic structure of LSB, we tried an extension of this initial strategy. In order to explore the search space more broadly, we created a new approach, called LSB-Nx, which represents the execution of the LSB algorithm for a number x of iterations. The execution of a construction phase followed by a local search a number of times characterizes the GRASP [Feo and Resende 1995; Resende and Ribeiro 2014] metaheuristic.

The idea of this new strategy is to take advantage of the random aspect of LSB. In this way, by running many iterations of the algorithm the strategy has the opportunity to generate different solutions in the constructive phase, which represent different starting points for the local search. In this case, the final solution returned by the algorithm is the best solution found among the x iterations.

For the sake of simplicity, let us call LSBSteps both portions of LSB code named as *Constructive phase* (lines 05 to 15) and *Local search phase* (lines 16 to 25) in Figure 1. Then the pseudocode of LSB-Nx algorithm is presented in Figure 2.

```
procedure LSB-Nx(E, dataset, p, α, x)

//Initialization
01.  E' ← ranking of attributes from E;
02.  k ← |E|*p/100;
03.  R ← first k attributes from E' //(pruning step);

04.  S ← φ;
05.  f(S) ← 0;
06.  for i from 1 to x do
07.     S' ← LSBSteps;
08.     if f(S') > f(S) then
09.        S ← S';
10.     end if;
11.  end for;
12.  return S;
end.
```

Fig. 2. Pseudocode of LSB-Nx

We evaluated different values for the x parameter in the LSB-Nx: 1 (which corresponds the LSB strategy), 2, 5, 10, 30, and 50. Table VII presents the comparison of these executions. In the first column we can see the datasets. The column labeled as LSB-N1 shows the accuracy values obtained with only one iteration. This strategy was defined as the baseline of the experiment. The remaining columns present the accuracy values (Acc%) obtained with different values of x and the percentage difference (Diff%) of the respective strategy regarding the accuracy obtained by the baseline algorithm. For Leukemia dataset, for example, the LSB-N2 represents an accuracy reduction of 1.92% regarding LSB-N1. For Prostate dataset, the LSB-N2 represents an accuracy increasing of 3.57% regarding the baseline.

In contradiction to what would be expected, the execution of LSB a number of times did not improve the quality of solutions for all datasets. When some heuristic is executed several times, better solutions found replace previous ones. Thus, we expect that the quality of the final solution from an experiment with several iterations is at least the same as the solution produced by an experiment with only one iteration. However, in the context of FSS, it is possible to see different behavior in the results of the training and the test datasets. The FSS algorithm searches for the best feature subset which maximizes the accuracy generated by the considered classifier, taking into consideration the training dataset. When the solution subset is defined, the final accuracy reported in the experiment

Table VII. Accuracy values obtained with LSB-Nx algorithm

	LSB-N1	LSB-N2		LSB-N5		LSB-N10		LSB-N30		LSB-N50	
Datasets	Acc%	Acc%	Diff%	Acc%	Diff%	Acc%	Diff%	Acc%	Diff%	Acc%	Diff%
Leukemia	94.89	93.07	-1.92	93.39	-1.58	93.39	-1.58	93.39	-1.58	94.46	-0.45
Dlbcl	88.20	84.60	-4.08	83.80	-4.99	83.40	-5.44	83.40	-5.44	83.33	-5.52
Lymphoma	80.47	80.13	-0.42	80.60	0.17	81.02	0.69	81.24	0.96	80.18	-0.35
Madelon	60.86	59.96	-1.48	60.02	-1.39	60.26	-0.99	60.24	-1.01	60.26	-0.99
Colon	80.52	79.52	-1.24	80.14	-0.47	81.09	0.71	80.43	-0.12	78.09	-3.02
Dexter	87.92	87.84	-0.09	88.17	0.28	88.21	0.33	90.21	2.61	88.31	0.45
Lung	98.57	96.91	-1.68	96.59	-2.01	96.59	-2.01	96.59	-2.01	96.70	-1.90
Prostate	79.85	82.70	3.57	81.63	2.23	82.37	3.17	84.93	6.37	84.94	6.38
Gisette	92.21	92.32	0.12	92.60	0.42	92.65	0.47	92.44	0.25	92.83	0.67
Mean Diff%			-0.80		-0.82		-0.52		0.00		-0.53

is calculated in the test dataset. So, when several iterations are executed in the training dataset, it is possible to observe an improvement in terms of solution quality. But when the solution subset is submitted to the classifier in the test dataset, the generated accuracy may be worse than that generated in the training dataset. For this reason, it is possible that increasing the number of iterations does not improve the quality of solutions. In the training dataset, we could observe this expected quality improvement in all datasets. But since this quality improvement did not occur in the test dataset for all databases, we will analyze each one individually.

Colon, Leukemia, Dlbcl and Lymphoma datasets have a small number of instances (62, 72, 47, and 96, respectively). When these datasets are split in 10 folds to enable cross-validation, each fold in test dataset is very small (less than 10 instances). Thus, when the prediction for one instance is incorrect, this mistake represents a significant percentage of the final accuracy. For this reason, if, by chance, the classifier misses more predictions when considering the solutions generated by the LSB-Nx with $x > 1$, the accuracy values will be worse than the values generated by LSB-N1. This fact makes it seem like increasing x does not contribute to the improvement in solutions quality. In the training datasets this improvement can be observed for the mentioned datasets.

In spite of the fact that Madelon dataset has many instances (2600), it has 500 attributes in total and, as we perform a 5% pruning, only 25 attributes are considered by the feature subset selection algorithm. This reduced number of attributes makes the search space more restricted, which does not cause improvement quality when several iterations are executed.

Lung and Prostate datasets have 181 and 136 instances, respectively, and about 12500 attributes both. Lung dataset does not present improvement in solutions quality because the quality obtained with only one iteration is already very high (it reaches accuracy of 100% is some folds). Therefore, it does not make sense to perform several iterations of LSB for this dataset. Prostate dataset does not reach very high accuracies with one iteration and, since it has many attributes, we can see an improvement in the solution quality when we increase the number of iterations.

Dexter and Gisette datasets have a high number of attributes (19999 and 5000, respectively) and many instances (600 and 6000). For these datasets we can see a quality solution improvement when more than one iteration are performed.

The computational cost of performing many iterations of LSB is very high, as we can see in Table VIII. Columns labeled as "t CPU" show the CPU time in minutes for executing the referred algorithm. Columns labeled as "Diff%" show the percentage difference of the respective strategy regarding the LSB-N1. For example, LSB-N2 takes 93.74% more time than LSB-N1 to generate a feature subset solution for Leukemia dataset.

As we observed previously, the gain in terms of solutions quality is short for most of datasets. In this way, the decision of performing a more broadly search depends on the dataset properties and how crucial is the matter of computational time for executing the algorithm. In the data mining context,

where FSS is usually a pre-processing step, it may be worth to wait extra time to obtain better quality solutions.

Table VIII. CPU time for different values of parameter x

	LSB-N1	LSB-N2		LSB-N5		LSB-N10		LSB-N30		LSB-N50	
Datasets	t CPU	t CPU	Diff%	t CPU	Diff%	t CPU	Diff%	t CPU	Diff%	t CPU	Diff%
Leukemia	0.0307	0.0594	93.74	0.1280	317.21	0.2386	677.77	0.8527	2679.34	1.5878	5075.47
Dlbcl	0.0126	0.0227	80.60	0.0488	288.24	0.0942	648.81	0.3403	2604.93	0.6316	4920.93
Lymphoma	0.2044	0.3465	69.51	0.8769	328.93	1.7785	769.94	6.8225	3237.16	13.1671	6340.55
Madelon	0.0584	0.1192	104.01	0.2332	299.14	0.4038	591.17	1.1817	1922.70	1.9454	3229.97
Colon	0.0153	0.0256	66.97	0.0515	236.16	0.0973	535.38	0.3205	1991.91	0.5583	3544.26
Dexter	3.3895	6.5777	94.06	18.4554	444.49	35.9899	961.81	136.5803	3929.54	173.0743	5006.23
Lung	0.1036	0.1989	91.88	0.3963	282.36	0.7642	637.38	2.2102	2032.55	5.3244	5037.37
Prostate	0.3588	0.8195	128.42	1.7067	375.72	3.5637	893.33	10.2947	2769.53	24.8796	6834.88
Gisette	13.8982	22.0034	58.32	45.1860	225.12	98.6875	610.07	293.4739	2011.60	479.8762	3352.79
Mean Diff%			87.50		310.82		702.85		2575.47		4815.83

7. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, we proposed a feature subset selection algorithm, based on a simple combination of a construction procedure and a local search phase. We are focused in the context of high-dimensional datasets, since the most popular FSS methods are not applied to this context, specially if the method of evaluating candidates is based on *wrapper* approaches. Our proposal aims at simplicity and efficiency, generating solutions which produces good accuracies and reducing significantly the number of attributes in the dataset, with a low computational cost.

We have compared LSB with some important FSS algorithms available in WEKA over nine high-dimensional datasets. A comparative study between our algorithm and the one proposed in [Bermejo et al. 2011] was also presented. Results showed that LSB is a very competitive proposal. It produces, in most cases, better accuracies with a lower computational cost.

A variation of LSB algorithm was evaluated. The new version, called LSB-Nx, corresponds to the execution of LSB a number x of iterations, which characterizes the GRASP metaheuristic. Results showed that increasing the number of iterations does not necessarily increase the quality of solutions in the context of FSS. It depends on the characteristics of the dataset.

For future work, one idea is to investigate some modifications in the LSB strategy, like changing the relevance measure for generating the initial ranking of attributes and trying different neighborhood strategies for the local search. Another idea is to perform experiments with other classifiers in addition to Naive Bayes. We also consider important to compare LSB with some recent approaches, like the one proposed in [Moshki et al. 2015], which uses the simulated annealing strategy for the local search.

Besides the evaluation of specific variations, new challenges are emerging in the FSS area. The very concept of high-dimensionality itself is also changing with datasets containing trillions of attributes being used nowadays [Bolón-Canedo et al. 2015].

REFERENCES

Bermejo, P., Gámez, J. A., and Puerta, J. M. Improving incremental wrapper-based feature subset selection by using re-ranking. In *Proceedings of the 23rd International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems – Volume Part I*. IEA/AIE'10. pp. 580–589, 2010.

Bermejo, P., Gámez, J. A., and Puerta, J. M. A GRASP algorithm for fast hybrid (filter-wrapper) feature subset selection in high-dimensional datasets. *Pattern Recognition Letters* vol. 32, pp. 701–711, 2011.

Bermejo, P., Gámez, J. A., and Puerta, J. M. Speeding up incremental wrapper feature subset selection with naive bayes classifier. *Knowledge-Based Systems* vol. 55, pp. 140–147, 2014.

Blum, A. L. and Langley, P. Selection of relevant features and examples in machine learning. *Artificial Intelligence* vol. 97, pp. 245–271, 1997.

Bolón-Canedo, V., Sánchez-Maroño, N., and Alonso-Betanzos, A. *Feature Selection for High-Dimensional Data*. Springer International Publishing, 2015.

Demšar, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research* vol. 7, pp. 1–30, 2006.

Duda, R., Hart, P., and Stork, D. *Pattern Classification and Scene Analysis*. John Willey and Sons, USA, 2001.

Esseghir, M. A. Effective wrapper-filter hybridization through GRASP schemata. *Journal of Machine Learning Research – Proceedings Track* vol. 10, pp. 45–54, 2010.

Fang, X. Inference-based naive bayes: Turning naive bayes cost-sensitive. *IEEE Transactions on Knowledge and Data Engineering* vol. 25, pp. 2302–2313, 2013.

Feo, T. and Resende, M. Greedy randomized adaptive search procedures. *Journal of Global Optimization* vol. 6, pp. 109–133, 1995.

Flores, M. J., Gámez, J. A., and Mateo, J. L. Mining the ESROM: A study of breeding value classification in manchego sheep by means of attribute selection and construction. *Computers and Electronics in Agriculture* vol. 60, pp. 167–177, 2008.

Friedman, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association* vol. 32, pp. 675–701, 1937.

Ginsberg, M. *Essentials of Artificial Intelligence*. Morgan Kaufmann Publishers Inc., USA, 1994.

Glover, F. and Laguna, M. *Tabu Search*. Kluwer Academic Publishers, USA, 1997.

Goldberg, D. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, USA, 1989.

Gutlein, M., Frank, E., Hall, M., and Karwath, A. Large-scale attribute selection using wrappers. In *Proceedings of IEEE Symposium on Computational Intelligence and Data Mining*. CIDM'09. pp. 332–339, 2009.

Guyon, I. and Elisseeff, A. An introduction to variable and feature selection. *Journal of Machine Learning Research* vol. 3, pp. 1157–1182, 2003.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. The WEKA data mining software: an update. *SIGKDD Explorations Newsletters* vol. 11, pp. 10–18, 2009.

Inza, I., Larrañaga, P., Etxeberria, R., and Sierra, B. Feature subset selection by bayesian network-based optimization. *Artificial Intelligence* vol. 123, pp. 157–184, 2000.

Kittler, J. Feature set search algorithms. *Pattern Recognition and Signal Processing*, 1978.

Liu, H. and Motoda, H. *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer Academic Publishers, USA, 1998.

Liu, H. and Yu, L. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering* vol. 17, pp. 491–502, 2005.

Moscato, P. A gentle introduction to memetic algorithms. In *Handbook of Metaheuristics*. Kluwer Academic Publishers, pp. 105–144, 2003.

Moshki, M., Kabiri, P., and Mohebalhojeh, A. Scalable feature selection in high-dimensional data based on GRASP. *Applied Artificial Intelligence* vol. 29, pp. 283–296, 2015.

Quinlan, J. R. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., USA, 1993.

Resende, M. and Ribeiro, C. GRASP: Greedy randomized adaptive search procedures. In *Search Methodologies*, E. K. Burke and G. Kendall (Eds.). Springer US, pp. 287–312, 2014.

Ruiz, R., Riquelme, J. C., and Aguilar-Ruiz, J. S. Incremental wrapper-based gene selection from microarray data for cancer classification. *Pattern Recognition* vol. 39, pp. 2383–2392, 2006.

Ruiz, R., Riquelme, J. C., and Aguilar-Ruiz, J. S. Best agglomerative ranked subset for feature selection. *Journal of Machine Learning Research* (4): 148–162, 2008.

Russell, S. J. and Norvig, P. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2003.

Tasca, M. *Contribuições ao Problema de Seleção de Atributos*. Ph.D. thesis, Universidade Federal Fluminense, Niterói, Rio de Janeiro, 2015.

Wilcoxon, F. Individual Comparisons by Ranking Methods. *Biometrics Bulletin* vol. 1, pp. 80–83, 1945.

Yang, J. and Honavar, V. G. Feature subset selection using a genetic algorithm. *IEEE Intelligent Systems* vol. 13, pp. 44–49, 1998.

Yusta, S. C. Different metaheuristic strategies to solve the feature selection problem. *Pattern Recognition Letters* vol. 30, pp. 525–534, 2009.