# Chapter 13
# A VNS-Based Heuristic for Feature Selection in Data Mining

A. Mucherino and L. Liberti

**Abstract.** The selection of features that describe samples in sets of data is a typical problem in data mining. A crucial issue is to select a maximal set of pertinent features, because the scarce knowledge of the problem under study often leads to consider features which do not provide a good description of the corresponding samples. The concept of consistent biclustering of a set of data has been introduced to identify such a maximal set. The problem can be modeled as a 0–1 linear fractional program, which is NP-hard. We reformulate this optimization problem as a bilevel program, and we prove that solutions to the original problem can be found by solving the reformulated problem. We also propose a heuristic for the solution of the bilevel program, that is based on the meta-heuristic Variable Neighborhood Search (VNS). Computational experiments show that the proposed heuristic outperforms previously proposed heuristics for feature selection by consistent biclustering.

## 13.1 Introduction

Nowadays technologies are able to produce a large quantity of data which needs to be analyzed. Data mining is a well-established field whose aim is to discover hidden patterns in the data for acquiring novel knowledge. A classic example is given by the huge quantity of data that is contained in DNA molecules of living beings. The relationships among the different genes of a DNA molecule, under different conditions, can provide important information regarding diseases and the functioning of life.

Data can be collected from different resources. *Samples* represent a single measurement of what is under study, and *features* are employed for describing the

A. Mucherino
IRISA, University of Rennes, Rennes, France
e-mail: antonio.mucherino@irisa.fr

L. Liberti
LIX, École Polytechnique, Palaiseau, France
e-mail: liberti@lix.polytechnique.fr

samples. In the example of the DNA molecule, a sample can represent the patients' condition, such as "healthy" and "sick", which is monitored through the expression levels of each gene in his DNA. In other words, each feature represents the expression level of a gene, and a list of feature measurements represents a sample. In many applications, the number of samples is scarce (only a few measurements are available), while the number of features is usually large (many factors are involved in the phenomenon under study).

In this context, *feature selection* is the problem of extracting only important and pertinent features from a set of data. Some considered features, indeed, may not be adequate for describing the samples, and, in such a case, they should be removed from the set of data. This brings two important consequences. First, if only pertinent features are used and all the others are rejected, the memory space necessary for storing this set in databases is optimized. Secondly, a strict relationship between samples and features may be identified, which could be exploited for discovering important information.

If a set of data contains $n$ samples which are described by $m$ features, then the whole set can be represented by a $m \times n$ matrix $A$, where the samples are organized column by column, and the features are organized row by row. In this context, we refer to a *bicluster* of $A$ as a submatrix of $A$, whose elements are a subset of samples and features. Equivalently, a bicluster can be seen as a pair of subsets $(S_r, F_r)$, where $S_r$ is a class (or cluster) of samples, and $F_r$ is a class (or cluster) of features.

**Definition 13.1.** *A biclustering is a partition of A in k biclusters:*

$$\mathbb{B} = \{(S_1,F_1),(S_2,F_2),\ldots,(S_k,F_k)\},$$

*such that the following conditions are satisfied:*

$$\bigcup_{r=1}^{k} S_r = A, \qquad S_\zeta \cap S_\xi = \emptyset \quad 1 \le \zeta \ne \xi \le k,$$

$$\bigcup_{r=1}^{k} F_r = A, \qquad F_\zeta \cap F_\xi = \emptyset \quad 1 \le \zeta \ne \xi \le k,$$

*where $k \le \min(n,m)$ is the number of biclusters [2, 6].*

If a classification for the samples of $A$ is available, as well as a classification for its features, a biclustering $\mathbb{B}$ can be trivially constructed. Inversely, classifications of samples and features can be extracted from $\mathbb{B}$.

In some data mining applications, there exist sets of data for which a classification of its samples is already given. In the example of the DNA molecules, samples may be taken from patients affected by different diseases, so that their classification is already known. In this case, the set $A$ is named *training set*. However, the classification of the features used for describing the samples is not known, or, equivalently, there is no biclustering $\mathbb{B}$ associated to $A$. Therefore, we have no a priori information about possible relationships between samples and features.

A way to obtain a classification for the features from a training set $A$ is to assign each feature to the class where it is “mostly expressed” (see Section 13.2 for more details). Then, once a classification for the features is also available, a biclustering $\mathbb{B}$ for $A$ can be obtained by simply applying Definition 13.1. If the found biclustering is *consistent* (in the sense stated in Section 13.2), then the selected features are most likely the ones that better describe the samples.

The feature selection problem related to consistent biclustering can be formulated as a 0–1 linear fractional optimization problem, which is NP-hard [9]. In this paper, we propose a new heuristic for solving the feature selection problem, that is based on a bilevel reformulation of the 0–1 linear fractional optimization problem. The proposed heuristic is based on the meta-heuristic Variable Neighborhood Search (VNS) [5, 10] and on the idea of solving exactly, at each iteration, the inner problem of the bilevel program, which is linear. Preliminary studies regarding the proposed heuristic for features selection have been previously presented in [12].

The rest of the paper is organized as follows. In Section 13.2, we develop the concept of consistent biclustering in more details, and we present the corresponding feature selection problem. In Section 13.3, we reformulate this feature selection problem as a bilevel optimization problem and we formally prove that solutions to the original problem can be found by solving this bilevel program. In Section 13.4, we introduce a new VNS-based heuristic for an efficient solution of the bilevel program. Computational experiments on real-life sets of data are presented in Section 13.5, as well as a comparison to the heuristic presented in [17]. Conclusions are given in Section 13.6.

## 13.2 Consistent Biclustering

Let $A = (a_{ij}) \in \Re^{m\times n}$ be a matrix representing a certain set of data, where samples $a^j$ are organized column by column, and their features $a_i$ are organized row by row. In the following, $k$ is the number of biclusters (known a priori) forming the biclustering, and the index $r \in \{1,2,\dots,k\}$ will refer to the generic class of samples or features.

If the set of data $A$ is a training set, then the classification of its samples in $k$ classes is known:

$$B_S = \{S_1, S_2, \dots, S_k\}.$$

Let $s_{ir}$ be a binary vector which indicates if the $i^{th}$ sample belongs to the class $S_r$ of samples ($s_{ir} = 1$) or not ($s_{ir} = 0$). Since $A$ is a training set, the vector $s_{ir}$ is known a priori. From the classification $B_S$, we can use the following procedure to construct a classification of the features in $k$ classes:

$$B_F = \{F_1, F_2, \dots, F_k\}.$$

The basic idea is to assign each feature to the class $F_{\hat{r}}$ (with $\hat{r} \in \{1,2,\dots,k\}$) such that it is mostly expressed (i.e. *it has higher value*), in average, in the class of

**Algorithm 13.** Procedure for constructing $B_F$ from $B_S$.

```
1: for (each feature $i$,  $i \in \{1,2,\ldots,n\}$) do
2:     let $\hat{r} = \arg\max_r \left( \frac{\sum_{j=1}^m a_{ij} s_{ir}}{\sum_{j=1}^m s_{ir}} \right)$;
3:     for each class $r$,  $r \in \{1,2,\ldots,k\}$ do
4:         let $f_{ir} = 0$;
5:     end for
6:     let $f_{i\hat{r}} = 1$;
7: end for
```

samples $S_{\hat{r}}$. Let $f_{ir}$ be a binary vector which indicates if the $i^{th}$ feature belongs to the class $F_r$ of features ($f_{ir} = 1$) or not ($f_{ir} = 0$), which is not known a priori. In order to define it and hence to give a classification $B_F$ to the features in $A$, we can employ Algorithm 13 [2, 17].

We remark that the same procedure can be used for finding a classification of the samples from a known classification of its features. Let

$$\hat{B}_S = \{\hat{S}_1, \hat{S}_2, \ldots, \hat{S}_k\}$$

be the classification of samples obtained from $B_F$. A biclustering $\mathbb{B}$ for $A$ can be defined by combining the two classifications $B_S$ and $B_F$ (see Definition 13.3). Moreover, if the classifications of samples $B_S$ and $\hat{B}_S$ are equivalent, then the biclustering $\mathbb{B}$ has a particular property that we call *consistency*.

**Definition 13.2.** *Let $A$ be a training set with classification of samples $B_S$. Let $B_F$ be the classification of its features obtained by Algorithm 13 from $B_S$, and let $\hat{B}_S$ the classification of samples obtained by Algorithm 13 from $B_F$. If $B_S = \hat{B}_S$, then the biclustering $\mathbb{B}$ of $A$ obtained by combining $B_S$ and $B_F$ is consistent [2].*

By definition, when a biclustering is consistent, the classification of the samples can be correctly reconstructed from the classification of its features, and vice versa. Therefore, the features are all able to describe accurately the samples of the set of data.

If a consistent biclustering exists for a certain set of data $A$, then $A$ is said to be *biclustering-admitting*. However, sets of data admitting consistent biclusterings are very rare in real-life applications. In other words, the situation $B_S \equiv \hat{B}_S$ is very difficult to be verified in practice, because some of the features used for describing the samples may not be actually pertinent. As a consequence, non-pertinent features should be removed from the set of data with the aim of finding a consistent biclustering for submatrices of $A$ in which some rows have been removed [2]. Note that it is very important to remove the least number of features, in order to preserve the information in the set of data.

Let us suppose that only a subset of features is considered: let

$$x = (x_1, x_2, \ldots, x_m)$$

be a binary vector of variables, where $x_i$ is 1 if the $i^{th}$ feature is selected, and it is 0 otherwise. Let $A[x]$ be the submatrix of $A$ obtained by removing all the rows $a_i$ for which $x_i = 0$. We give the following definition.

**Definition 13.3.** *A biclustering for $A[x]$ is consistent if and only if, $\forall \hat{r}, \xi \in \{1,2,\ldots,k\}$, $\hat{r} \neq \xi, j \in S_{\hat{r}}$, the following inequality is satisfied [2]:*

$$\frac{\sum_{i=1}^{m} a_{ij} f_{i\hat{r}} x_i}{\sum_{i=1}^{m} f_{i\hat{r}} x_i} > \frac{\sum_{i=1}^{m} a_{ij} f_{i\xi} x_i}{\sum_{i=1}^{m} f_{i\xi} x_i}. \tag{13.1}$$

Note that the two fractions in (13.1) are used for computing the *centroids* of the considered biclusters (for each sample in $S_{\hat{r}}$, the average over the features belonging to same class is computed). On the left hand side of (13.1), the $j^{th}$ component of the centroid of the bicluster $(S_{\hat{r}}, F_{\hat{r}})$ is computed. On the right hand side of (13.1), the $j^{th}$ component of the centroid of the bicluster $(S_{\hat{r}}, F_{\xi})$ is computed. In order to have a consistent biclustering for $A[x]$ (i.e. $B_S \equiv \hat{B}_S$), all components of the centroid of $(S_{\hat{r}}, F_{\hat{r}})$ must have a value that is larger than any other. This condition on the classification $B_F$ of features allows Alg. 13 to generate a classification of samples $\hat{B}_S$ that is equivalent to the original classification $B_S$.

In order to overcome issues related to sets of data containing noisy data and errors, the concepts of $\alpha$-consistent biclustering and $\beta$-consistent biclustering have been introduced in [17]. The basic idea is to artificially increase the margin between the centroids of the different biclusters in the constraints (13.1). In this way, small variations due to noisy data and errors should not be able to spoil the feature selection.

**Definition 13.4.** *Given a real parameter $\alpha > 0$, a biclustering for $A[x]$ is $\alpha$-consistent if and only if, $\forall \hat{r}, \xi \in \{1,2,\ldots,k\}, \hat{r} \neq \xi, j \in S_{\hat{r}}$, the following inequality is satisfied [17]:*

$$\frac{\sum_{i=1}^{m} a_{ij} f_{i\hat{r}} x_i}{\sum_{i=1}^{m} f_{i\hat{r}} x_i} > \alpha + \frac{\sum_{i=1}^{m} a_{ij} f_{i\xi} x_i}{\sum_{i=1}^{m} f_{i\xi} x_i}. \tag{13.2}$$

The additive parameter $\alpha > 0$ is used to guarantee that the margin between the centroid of $(S_{\hat{r}}, F_{\hat{r}})$ and any other bicluster concerning $S_{\hat{r}}$ is at least greater than $\alpha$, independently from the considered data. Similarly, in the case of $\beta$-consistent biclustering, a multiplicative parameter $\beta$ is employed.

**Definition 13.5.** *Given a real parameter $\beta > 1$, a biclustering for $A[x]$ is $\beta$-consistent if and only if, $\forall \hat{r}, \xi \in \{1,2,\ldots,k\}, \hat{r} \neq \xi, j \in S_{\hat{r}}$, the following condition is satisfied [11]:*

$$\begin{cases} \dfrac{\sum_{i=1}^{m} a_{ij} f_{i\hat{r}} x_i}{\sum_{i=1}^{m} f_{i\hat{r}} x_i} > \beta \, \dfrac{\sum_{i=1}^{m} a_{ij} f_{i\xi} x_i}{\sum_{i=1}^{m} f_{i\xi} x_i} & \text{if} \, c > 0 \\ \dfrac{\sum_{i=1}^{m} a_{ij} f_{i\hat{r}} x_i}{\sum_{i=1}^{m} f_{i\hat{r}} x_i} > (2-\beta) \, \dfrac{\sum_{i=1}^{m} a_{ij} f_{i\xi} x_i}{\sum_{i=1}^{m} f_{i\xi} x_i} & \text{if} \, c < 0 \end{cases} \tag{13.3}$$

*where*

$$c = \frac{\sum_{i=1}^{m} a_{ij} f_{i\xi} x_i}{\sum_{i=1}^{m} f_{i\xi} x_i}.$$

We remark that the concept of $\beta$-consistent biclustering was firstly introduced in [17], but the given definition was only suitable for sets of data containing non-negative entries. In general, different values for the parameters $\alpha$ and $\beta$ could be used for each $j$ in Definitions 13.4 and 13.5. Usually, however, only one value is set up for all components of the centroids.

In real-life applications, there are usually no biclusterings which are consistent, $\alpha$-consistent or $\beta$-consistent if all features are selected (this situation corresponds to a binary vector $x$ with all its components equal to 1). As already mentioned before, this happens because some of the considered features may actually be inadequate. Such features must therefore be removed from the set of data, while the total number of considered features must be maximized in order to preserve as much information as possible. The following combinatorial optimization problem is therefore considered:

$$\max_x \left( f(x) = \sum_{i=1}^{m} x_i \right), \tag{13.4}$$

subject to constraints (13.1), (13.2) or (13.3) depending on the fact that a consistent, $\alpha$-consistent or $\beta$-consistent biclustering, respectively, is searched. The three problems are linear with fractional constraints and binary variables. The solution of this kind of optimization problems could be attempted by general-purpose solvers, such as `Baron` [19, 20] or `Couenne` [1], but the large size of real-life sets of data can make their converge very slow and the computational experiments too expensive. The three optimization problems are in fact all NP-hard [9]. In [2] and [17], two heuristics have been proposed. The heuristic we propose in this paper is able to provide better solutions with respect to the ones previously obtained.

## 13.3 A Bilevel Reformulation

In the following discussion, only the optimization problem (13.1)-(13.4) will be considered, because similar observations can be made for the other two problems. The computational experiments reported in Section 13.5, however, will be related to all three optimization problems.

We propose a reformulation of the problem (13.1)-(13.4) as a bilevel optimization problem. To this aim, we substitute the denominators in the constraints (13.1) with new continuous variables $y_r$, $r = 1,2,\dots,k$, where each $y_r$ is related to the bicluster $(S_r,F_r)$. We can rewrite the constraints (13.1) as follows:

$$\frac{1}{y_{\hat{r}}}\sum_{i=1}^{m} a_{ij} f_{i\hat{r}} x_i > \frac{1}{y_{\xi}}\sum_{i=1}^{m} a_{ij} f_{i\xi} x_i, \tag{13.5}$$

where $y_{\hat{r}}$ and $y_{\xi}$ replace the original fractional parts. The constraints (13.5) must be satisfied $\forall \hat{r},\xi \in \{1,2,\dots,k\}$, $\hat{r} \neq \xi$ and $j \in S_{\hat{r}}$, in order to have a consistent biclustering.

Let us consider $\bar{y}_r = \delta y_r$, where $\delta > 0$. It is easy to see that, given certain values for the variables $x_i$, the constraints (13.5) are satisfied with $\bar{y}_r$ if and only if they are satisfied with $y_r$. As an example, if $k = 3$ and there is a consistent biclustering in which 20, 30 and 50 features are selected in the $k$ biclusters, then the constraints (13.5) are also satisfied if 0.20, 0.30 and 0.50, respectively, replace the actual number of features (in this example, the proportional factor $\delta$ is 0.01). For this reason, the variables $y_r$ can be used for representing the *proportions* among the cardinalities of the classes of features. In the previous example, 20% of the selected features are in the first bicluster, 30% of the features in the second one, and 50% in the last one. As a consequence, the variables $y_r$ can be bound in the real interval $[0,1]$, so that we can consider the following constraint:

$$\sum_{r=1}^{k} y_r \leq 1.$$

A percentage of features is not selected when this sum is smaller than 1.

We introduce the function:

$$c(x,\hat{r},\xi) = \sum_{j\in S_{\hat{r}}} \left| \frac{\sum_{i=1}^{m} a_{ij} f_{i\xi} x_i}{\sum_{i=1}^{m} f_{i\xi} x_i} - \frac{\sum_{i=1}^{m} a_{ij} f_{i\hat{r}} x_i}{\sum_{i=1}^{m} f_{i\hat{r}} x_i} \right|_{+},$$

where $x = (x_1,x_2,\dots,x_m)$ and $\hat{r},\xi \in \{1,2,\dots,k\}$, with $\hat{r} \neq \xi$, and where the symbol $|\cdot|_+$ represents the function which returns its argument if it is positive, and it returns 0 otherwise. As a consequence, the value of $c(x,\hat{r},\xi)$ is positive if and only if at least one constraint (13.1) is not satisfied.

We reformulate the optimization problem (13.1)-(13.4) as the following bilevel optimization problem:

$$\min_y \left( g(x,y) = \sum_{r=1}^{k} \left[ (1-y_r) + \sum_{\xi=1:\xi\neq r}^{k} c(x,r,\xi) \right] \right) \tag{13.6}$$

subject to:

$$\begin{aligned} x = \arg\max_x & \left( f(x) = \sum_{i=1}^{m} x_i \right) \\ \text{subject to} & \begin{cases} \sum_{i=1}^{m} f_{ir} x_i = \lfloor y_r \sum_{i=1}^{m} f_{ir} \rfloor & \forall r \in \{1,\ldots,k\} \\ \text{constraint (5)} \end{cases} \\ \sum_{r=1}^{k} y_r \leq 1. & \end{aligned} \tag{13.7}$$

The objective function $g$ of the outer problem depends on both variables $x_i$, with $i \in \{1,2,\ldots,m\}$, and $y_r$, with $r \in \{1,2,\ldots,k\}$. For each class $S_r$, the generic term of $g$ is the sum of two parts, one depending on the vector $y$ and the other one depending on the vector $x$. The first part is simply the difference $(1-y_r)$, that must be minimized in order to maximize the value for $y_r$, which represents the percentage of selected features in the class $F_r$ (recall that $y_r \leq 1$). The second part is the sum over all the other classes $S_\xi$, with $\xi \neq r$, of the function $c(x,r,\xi)$ (when its value is positive). The minimization of this second part allows to minimize the number of constraints (13.1) that are not satisfied.

The bilevel program is subject to two constraints. The first one is based on the solution of another optimization problem, to which we refer as inner problem. The inner problem can be seen as a linear simplification of the original problem (13.1)-(13.4), where the fractional parts have been substituted by the variables $y_r$, which indicate the percentage of features to be selected in each bicluster. The solution of the inner problem provides a set of values for the variables $x_i$ from the variables $y_r$. Therefore, whatever method is employed for the solution of the outer optimization problem, the search can be reduced to the variables $y_r$ only, because the corresponding values for the variables $x_i$ can be obtained by solving the inner problem. The inner problem is subject to two constraints: the constraints (13.5), as well as another constraint that forces the number of selected features in each bicluster to respect the percentages given by the variables $y_r$. The second constraint of the outer problem requires that the sum of all variables $y_r$ must be smaller or equal to 1 (no more than 100% of features can be selected in total). We formally prove that solutions to the proposed bilevel optimization problem are also solutions to the original problem (13.1)-(13.4).

**Proposition 1** *If $(\hat{x},\hat{y})$ is solution for (13.6)-(13.7), then $\hat{x}$ is solution for (13.1)-(13.4).*

*Proof.* By contradiction, let us suppose that there is a solution $\bar{x}$ such that $f(\bar{x}) > f(\hat{x})$ and constraints (13.1) are satisfied. Let

$$\bar{y}_r = \frac{\sum_{i=1}^{n} f_{ir}\bar{x}_i}{\sum_{i=1}^{n} f_{ir}} \qquad \forall r \in \{1,2,\dots,k\}.$$

Since the constraints (13.1) are satisfied,

$$g(\bar{x},\bar{y}) = \sum_{r=1}^{k} (1-\bar{y}_r) = k - \sum_{r=1}^{k} \bar{y}_r.$$

Then,

$$f(\bar{x}) > f(\hat{x}) \Longrightarrow \sum_{r=1}^{k} \bar{y}_r > \sum_{r=1}^{k} \hat{y}_r \Longrightarrow g(\bar{x},\bar{y}) < g(\hat{x},\hat{y}),$$

which brings to a contradiction. □

## 13.4 A VNS-Based Heuristic

The heuristic we propose for the solution of the bilevel program presented in Section 13.3 is based on the meta-heuristic Variable Neighborhood Search (VNS) [5, 10], which is one of the most successful heuristics for global optimization. The VNS is based on the idea of exploring small neighbors of currently known solutions, which are increased in size when no better solutions can be found. At each iteration of the VNS, a local search algorithm is often employed, so that a path of local optima can be defined, that may lead to the global optimum of the considered problem. The local search can however be replaced by another VNS, which is nested in the main one.

The proposed heuristic actually implements a VNS in two main steps with an adaptive value for the percentage of unselected features *unsel*, which is small at the beginning (*unsel* $\simeq 0$), and then it increases when no better solutions can be found in the current neighbor. In this way, the algorithm firstly tries to find solutions where the number of selected features is high. Afterwards, solutions where fewer features are selected are considered. For each neighbor of the first step of VNS, there is a full execution of another step. The neighbors of the second step of VNS are generated so that the set of variables $y_r$ can be slightly perturbed at the beginning ($range = starting\_range$), and larger perturbations can be performed only when no better solutions can be found by considering the current neighbor.

Algorithm 14 is a sketch of our heuristic for feature selection by consistent biclustering. At the beginning, the variables $x_i$ are all set to 1, and the variables $y_r$ are set so that they represent the distribution of all $m$ features among the $k$ classes.

**Algorithm 14.** A VNS-based heuristic for feature selection.

```
1: let $iter = 0$;
2: let $x_i = 1, \forall i \in \{1,2,\ldots,m\}$;
3: let $y_r = \sum_i f_{ir}/m, \forall r \in \{1,2,\ldots,k\}$;
4: let $y_r^{best} = y_r, \forall r \in \{1,2,\ldots,k\}$;
5: let $range = starting\_range$;
6: let $unsel = 0$;
7: while (constraints (13.1) unsatisfied and $unsel \leq max\_unsel$) do
8:    while (constraints (13.1) unsatisfied and $range \leq max\_range$) do
9:       let $iter = iter + 1$;
10:      solve inner optimization problem (linear & cont.);
11:      if (constraints (13.1) unsatisfied) then
12:         increase $range$;
13:         if ($g$ has improved) then
14:            let $y_r^{best} = y_r, \forall r \in \{1,2,\ldots,k\}$;
15:            let $range = starting\_range$;
16:         end if
17:         let $y_r = y_r^{best}, \forall r \in \{1,2,\ldots,k\}$;
18:         let $r'$ = random in $\{1,2,\ldots,k\}$;
19:         choose randomly $y_{r'}$ in $[y_{r'} - range, y_{r'} + range]$;
20:         let $r''$ = random in $\{1,2,\ldots,k\}$ : $r' \neq r''$;
21:         set $y_{r''}$ so that $1 - unsel \leq \sum_r y_r \leq 1$;
22:      end if
23:   end while
24:   if (constraints (13.1) unsatisfied) then
25:      increase $unsel$;
26:   end if
27: end while
```

If the biclustering is already consistent, then all features can be selected, and the algorithm stops.

For each neighbor defined by the second VNS step, the variables $y_r$ are randomly modified. $y_{r'}$ and $y_{r''}$ are chosen randomly so that $r' \neq r''$. Then, $y_{r'}$ is perturbed, and its value is chosen randomly in the interval centered in $y_{r'}^{best}$ and with length $2 \times range$. Then, a random value for $y_{r''}$ is chosen so that $1 - unsel \leq \sum_r y_r \leq 1$. In this way, the new set of values for $y_r$ falls in the two current neighbors defined by the VNS.

The inner optimization problem is solved for each random choice for the variables $y_r$. It is a linear 0–1 optimization problem, and we consider its continuous relaxation, i.e. we allow the variables $x_i$ to take any real value in the interval $[0,1]$. Therefore, after a solution has been obtained, we substitute the fractional values of $x_i$ with 0 if $x_i \leq 1/2$, or with 1 if $x_i > 1/2$. In our experiments, the equality of the first constraint of the inner problem is relaxed to an inequality:

$$\sum_{i=1}^{m} f_{ir} x_i \leq \lfloor y_r \sum_{i=1}^{m} f_{ir} \rfloor \quad \forall r \in \{1,\ldots,k\}.$$

The strict inequality of the constraints (13.5) is also relaxed, so that the domains defined by the constraints are closed domains. Under these hypotheses, the inner problem can be solved by commonly used solvers for mixed integer linear programming (MILP), e.g. CPLEX [7].

After the solution of the inner problem, the original set of constraints (13.1) is checked. If the obtained values for the variables $x_i$, along with the used values for the variables $y_r$, define a consistent biclustering, then the algorithm stops. Otherwise, some of the variables $y_r$ are modified and a new iteration of the algorithm is performed.

We point out that heuristics offer no guarantee of optimality. One way to enhance the algorithm is to restart it and to allow only values for the variables $y_r$ corresponding to a larger number of selected features. Moreover, since the algorithm can provide different solutions if it is executed more than once (with different seeds for the generator of random numbers), it can be executed a certain number of times. The best obtained solution is then taken into consideration.

## 13.5 Computational Experiments

We implemented the presented heuristic for feature selection in AMPL [4], from which the ILOG CPLEX11 solver [7] is invoked for the solution of the inner optimization problem. We also implemented in AMPL the heuristic previously proposed in [17] (for more details about this heuristic, the reader is referred to the reference paper). Experiments are carried out on an Intel Core 2 CPU 6400 @ 2.13 GHz with 4GB RAM, running Linux.

The following four subsections are devoted to four different training sets from different real-life applications for which we selected a subset of pertinent features. They are ordered by the increasing number of features originally contained in the training set. We will briefly describe each considered training set and then we will focus our attention on the presented experiments. The interested reader can find more information about these sets of data in the provided references. The comparison of the two algorithms will be carried out by comparing the quality of the found solutions. The heuristic in [17] is in general faster to converge (or to get stuck in non-optimal solutions, see experiments), whereas our heuristic is generally able to find better-quality solutions. CPU times range from a few seconds (wine fermentations) to about half an hour (ovarian cancer).

### *13.5.1 Wine Fermentations*

Problems occurring during the fermentation process of wine can impact the productivity of wine-related industries and also the quality of wine [13, 14]. The fermentation process of wine can be too slow or it can even become stagnant. Predicting

**Table 13.1** Wine fermentations. Total features: 450.

| | VNS-based heuristic | | | | | | | Heuristic in [17] | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\alpha$ | 0 | 0.20 | 0.40 | 0.60 | 0.80 | 1.00 | 1.20 | 0 | 0.20 | 0.40 | 0.60 | 0.80 | 1.00 | 1.20 |
| $f(x)$ | 431 | 430 | 427 | 427 | 424 | 421 | 415 | 425 | 424 | 424 | 420 | stuck | stuck | stuck |
| $\beta$ | 1 | 1.01 | 1.02 | 1.04 | 1.06 | 1.08 | 1.10 | 1 | 1.01 | 1.02 | 1.04 | 1.06 | 1.08 | 1.10 |
| $f(x)$ | 431 | 430 | 429 | 425 | 422 | 411 | 401 | 425 | 424 | 423 | 420 | 415 | 400 | 386 |

how good the fermentation process is going to be may help enologists who can then take suitable steps to make corrections when necessary and to ensure that the fermentation process concludes smoothly and successfully.

We present some analysis performed on a set of data obtained from a winery in Chile's Maipo Valley, which is the result of 24 measurements of industrial vinifications of *Cabernet sauvignon* [22, 23]. The data are related to the harvest of 2002. The level of 30 compounds are analyzed during time: the whole set of data consists of approximately 22000 data points. In this paper, the considered set of data contains 24 fermentations described by $15 \times 30 = 450$ features: the first class contains *normal* fermentations (9 in total), whereas the second class contains *problematic* fermentations (15 in total).

Table 13.1 shows some computational experiments. Note that $\alpha$-consistent biclusterings with $\alpha = 0$ and $\beta$-consistent biclusterings with $\beta = 1$ correspond to consistent biclusterings (see Definition 13.3). We executed Algorithm 14 with different choices for the two parameters $\alpha$ and $\beta$. In the table, the number of selected features $f(x)$ is given in correspondence with each experiment. We can remark that the number of selected features decreases as the values of $\alpha$ or $\beta$ increases. This was expected, because fewer features should be selected when the required margin between the centroids of the biclusters is enlarged. Only the features that better describe the samples in the set of data should be contained in the biclustering we found that contain fewer features (in particular, the $\alpha$-consistent biclustering with $\alpha = 1.20$ and the $\beta$-consistent biclustering with $\beta = 1.10$).

Table 13.1 also shows some results obtained by using the heuristic presented in [17] (the reader is referred to the reference paper for a sketch of the algorithm). The comparison with the VNS-based heuristic proposed in this paper shows that our heuristic was able to find better solutions for all experiments, i.e. it was able to find biclusterings having the desired consistency property where a larger number of features are selected. Moreover, in some experiments, the heuristic in [17] got stuck and was not able to provide any solution. This heuristic is based on the solution of a sequence of linear optimization problems, where some parameters are modified on the basis of partial found solutions. The algorithm got stuck when such parameters stopped changing iteration after iteration.

By using the found biclusterings, we were able to identify a subset of compounds that are most likely the cause of problematic wine fermentations [15]. For example, among the organic acids, the features related to lactic, malic, succinic, and tartaric acids are always preserved during the feature selection. Moreover, all the features

**Table 13.2** Colon cancer, set I. Total features: 2000.

| | VNS-based heuristic | | | | | | Heuristic in [17] | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\alpha$ | 0 | 1 | 2 | 3 | 5 | 8 | 0 | 1 | 2 | 3 | 5 | 8 |
| $f(x)$ | 1700 | 1698 | 1617 | 1596 | 1583 | 1352 | 1700 | 1531 | 1590 | 1590 | 1530 | 1211 |
| $\beta$ | 1 | 1.02 | 1.03 | 1.05 | 1.08 | 1.10 | 1 | 1.02 | 1.03 | 1.05 | 1.08 | 1.10 |
| $f(x)$ | 1700 | 1593 | 1577 | 1566 | 1432 | 1269 | 1700 | 1578 | 1509 | 1108 | 1082 | stuck |

related to each of these organic acids are assigned to only one bicluster, showing that they can play a very important role for the classification of the fermentations. Features related to the same compound can also be always discarded, or they can show some regular patterns. The study of all these features in the biclusterings can give some insights on fermentation process of wine. Moreover, the found biclusterings can also be exploited for performing supervised predictions of new fermentations from which the selected compounds have been monitored [16].

### *13.5.2 Colon Cancer – Set I*

This set of data contains 62 samples collected from colon-cancer patients [21]. Among them, 40 *tumor biopsies* are from tumors and 22 *normal biopsies* are from healthy parts of the colons of the same patients. 2000 out of around 6500 genes were selected based on the confidence in the measured expression levels. This set of data, along with the known classification of its samples, is available on the Kent Ridge Database [8].

Table 13.2 shows the results of some experiments performed with the aim of finding consistent, $\alpha$-consistent and $\beta$-consistent biclusterings of this set of data. As in the previous experiments, the two algorithms selected a smaller number of features when the values for the parameters $\alpha$ or $\beta$ were larger. In these experiments, the two heuristics found two consistent biclusterings with the same number of features only once (1700 out of 2000 features), when $\alpha = 0$ and $\beta = 1$. In all other cases, our VNS-based heuristic was able to provide better solutions. The heuristic in [17] got stuck when $\beta$ was set to 1.10. Moreover, in the experiments regarding $\alpha$-consistent biclustering, $f(x)$ does not decrease regurarly with larger $\alpha$ values, showing that the heuristic in [17] was not able to find the optimal solution.

### *13.5.3 Colon Cancer – Set II*

The third set of data that we consider is a set of gene expressions related to human tissues from sick patients (affected by colon cancer) and healthy patients [18]. This set of data is available on the web site of the Princeton University (see the reference

**Table 13.3** Colon cancer, set II. Total features: 7457.

| | VNS-based heuristic | | | | | Heuristic in [17] | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $\alpha$ | 0 | 1 | 2 | 5 | 10 | 0 | 1 | 2 | 5 | 10 |
| $f(x)$ | 7450 | 7448 | 7444 | 7413 | 7261 | 7450 | 7430 | 7291 | stuck | stuck |
| $\beta$ | 1 | 1.10 | 1.50 | 2.00 | 3.00 | 1 | 1.10 | 1.50 | 2.00 | 3.00 |
| $f(x)$ | 7450 | 7420 | 7107 | 6267 | 5365 | 7450 | 7349 | 7099 | 6054 | 5252 |

**Table 13.4** Ovarian cancer. Total features: 15154.

| | VNS-based heuristic | | | |
|---|---|---|---|---|
| $\alpha$ | 0 | 0.001 | 0.005 | 0.009 |
| $f(x)$ | 12701 | 12471 | 12198 | 11027 |
| $\beta$ | 1 | 1.001 | 1.005 | 1.009 |
| $f(x)$ | 12701 | 12519 | 12392 | 12233 |

for the web link). It contains 36 samples classified as *normal* or *cancer*, and each sample is described through 7457 features.

Table 13.3 shows some computational experiments. Even in these experiments, there is the tendency to select a smaller number of features when $\alpha$ or $\beta$ are increased in value. The number of features that are selected by the heuristic in [17] is always smaller than the number of features selected by the VNS-based heuristic.

### *13.5.4 Ovarian Cancer*

This set contains data collected from experiments performed with the aim of identifying gene patterns that can distinguish ovarian cancer from non-cancer [3]. As the authors of the reference paper remark, this study is significant to women who have a high risk of ovarian cancer due to family or personal history of cancer. The set of data includes 91 samples classified as *normal* and 162 samples classified as *ovarian cancer*. The total number of considered features is 15154. After the experiments, the intensity values of the raw data were normalized so that each intensity value can fall within the interval $[0,1]$. More details on these experiments can be found in [3]. The set of data can be downloaded from [8]: it is the largest set of data ever considered for feature selection by consistent biclustering. Our VNS-based heuristic was able to identify some consistent, $\alpha$-consistent and $\beta$-consistent biclusterings by selecting a subset of pertinent features. Table 13.4 shows some computational experiments.

## 13.6 Conclusions

We considered a problem of great interest in data mining, that is the one of the identification of consistent biclusterings of sets of data, which are used to identify pertinent features describing the samples of a set of data. We presented a reformulation of the problem, originally modeled as a 0–1 linear fractional optimization problem, as a bilevel program and we proposed a new heuristic for its solution. This heuristic is based on the meta-heuristic Variable Neighborhood Search. Computational experiments on various sets of data available in the literature show that the proposed heuristic outperforms previously proposed ones and is promising for the solution of large instances.

Data are nowadays obtained from many resources and they need to be efficiently analyzed. The VNS-based heuristic we proposed represents a good step forward a satisfactory solution of feature selection problems. However, a wider test analysis of the algorithm on other training sets is needed in order to study possible improvements. To this aim, we plan to implement the heuristic in C/C++, so that the CPLEX solver can be invoked more efficiently and the overall execution can be optimized.

**Acknowledgements.** The authors are thankful to Sonia Cafieri for the fruitful discussions on the bilevel reformulation presented in the paper.

## References

1. Belotti, P.: Couenne: a user's manual. Technical report, Lehigh University (2009)
2. Busygin, S., Prokopyev, O.A., Pardalos, P.M.: Feature selection for consistent biclustering via fractional 0–1 programming. Journal of Combinatorial Optimization 10, 7–21 (2005)
3. Hitt, B.A., Levine, P.J., Fusaro, V.A., Steinberg, S.M., Mills, G.B., Simone, C., Fishman, D.A., Kohn, E.C., Liotta, L.A., Petricoin III, E.F., Ardekani, A.M.: Use of proteomic patterns in serum to identify ovarian cancer. The Lancet 359, 572–577 (2002)
4. Fourer, R., Gay, D.M., Kernighan, B.W.: AMPL: A Modeling Language for Mathematical Programming. Brooks/Cole Publishing Company, Cengage Learning (2002)
5. Hansen, P., Mladenovic, N.: Variable neighborhood search: Principles and applications. European Journal of Operational Research 130(3), 449–467 (2001)
6. Hartigan, J.: Clustering Algorithms. John Wiles & Sons, New York (1975)
7. Ilog cplex solver, `http://www.ilog.com/products/cplex/`
8. Kent ridge database, `http://datam.i2r.a-star.edu.sg/datasets/krbd/`
9. Kundakcioglu, O.E., Pardalos, P.M.: The complexity of feature selection for consistent biclustering. In: Butenko, S., Pardalos, P.M., Chaovalitwongse, W.A. (eds.) Clustering Challenges in Biological Networks. World Scientific Publishing (2009)
10. Mladenovic, M., Hansen, P.: Variable neighborhood search. Computers and Operations Research 24, 1097–1100 (1997)
11. Mucherino, A.: Extending the definition of $\beta$-consistent biclustering for feature selection. In: Proceedings of the Federated Conference on Computer Science and Information Systems, FedCSIS 2011. IEEE (2011)
12. Mucherino, A., Cafieri, S.: A new heuristic for feature selection by consistent biclustering. Technical Report arXiv:1003.3279v1 (March 2010)
13. Mucherino, A., Papajorgji, P., Pardalos, P.M.: Data Mining in Agriculture. Springer (2009)
14. Mucherino, A., Papajorgji, P., Pardalos, P.M.: A survey of data mining techniques applied to agriculture. Operational Research: An International Journal 9(2), 121–140 (2009)

15. Mucherino, A., Urtubia, A.: Consistent biclustering and applications to agriculture. In: Proceedings of the Industrial Conference on Data Mining, ICDM 2010, Workshop on Data Mining and Agriculture DMA 2010, IbaI Conference Proceedings, pp. 105–113. Springer, Berlin (2010)
16. Mucherino, A., Urtubia, A.: Feature selection for datasets of wine fermentations. In: Proceedings of the 10th International Conference on Modeling and Applied Simulation, MAS 2011. I3A (2011)
17. Nahapatyan, A., Busygin, S., Pardalos, P.M.: An improved heuristic for consistent biclustering problems, vol. 102, pp. 185–198. Springer
18. Notterman, D.A., Alon, U., Sierk, A.J., Levine, A.J.: Transcriptional gene expression profiles of colorectal adenoma, adenocarcinoma, and normal tissue examined by oligonucleotide arrays. Cancer Research 61, 3124–3130 (2001)
19. Sahinidis, N.V., Tawarmalani, M.: BARON 9.0.4: Global Optimization of Mixed-Integer Nonlinear Programs. User's Manual (2010)
20. Tawarmalani, M., Sahinidis, N.V.: A polyhedral branch-and-cut approach to global optimization. Mathematical Programming 103, 225–249 (2005)
21. Notterman, D.A., Gish, K., Ybarra, S., Mack, D., Levine, A.J., Alon, U., Barkai, N.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. PNAS 96, 6745–6750 (1999)
22. Urtubia, A., Perez-Correa, J.R., Meurens, M., Agosin, E.: Monitoring large scale wine fermentations with infrared spectroscopy. Talanta 64(3), 778–784 (2004)
23. Urtubia, A., Perez-Correa, J.R., Soto, A., Pszczolkowski, P.: Using data mining techniques to predict industrial wine problem fermentations. Food Control 18, 1512–1517 (2007)