Data Grid Models for Preparation and Modeling in Supervised Learning

Marc Boullé marc.boulle@orange-ftgroup.com
France Telecom R&D
2, avenue Pierre Marzin
22300 Lannion, France

Editor: Isabelle Guyon, Gavin Cawley, Gideon Dror and Amir Saffari

Abstract

This paper introduces a new method to automatically, rapidly and reliably evaluate the class conditional probability of any subset of variables in supervised learning. It is based on a partitioning of each input variable into intervals in the numerical case and into groups of values in the categorical case. The cross-product of the univariate partitions forms a multivariate partition of the input representation space into a set of cells. This multivariate partition, called data grid, is a piecewise constant nonparametric estimator of the class conditional probability. The best data grid is searched using a Bayesian model selection approach and an efficient combinatorial algorithm.

We also extend data grids to joint density estimation in unsupervised learning and apply this extension to the problem of coclustering the instances and variables of a sparse binary dataset.

We finally present three classification techniques, exploiting the maximum a posteriori data grid, an ensemble of data grids, or a coclustering data grid, and report results in the Agnostic Learning vs. Prior Knowledge Challenge, where our method achieved the best performance on two of the datasets. These experiments demonstrate the value of using data grid models in machine learning tasks, for conditional density estimation, data preparation, supervised classification, clustering and rule based explanation.

1. Introduction

Univariate partitioning methods have been studied extensively in the past, mainly in the context of decision trees (Kass, 1980; Breiman et al., 1984; Quinlan, 1993; Zighed and Rakotomalala, 2000). Supervised discretization methods split the numerical domain into a set of intervals and supervised value grouping methods partition the input values into groups. Fine grained partitions allow an accurate discrimination of the output values, whereas coarse grain partitions tend to be more reliable. When the size of the partition is a free parameter, the trade-off between information and reliability is an issue. In the MODL approach, supervised discretization (Boullé, 2006) (or value grouping (Boullé, 2005)) is considered as a nonparametric model of dependence between the input and output variables. The best partition is found using a Bayesian model selection approach.

In this paper, we describe an extension of the MODL approach to the supervised bivariate case for pairs of input variables (Boullé, 2007a)[1], and introduce its generalization to any subset of variables of any types, numerical, categorical or mixed types. Each input variable is partitioned, into intervals in the numerical case and into groups of values in the categorical case. The cross-product of the univariate partitions forms a multi-dimensional data grid. The correlation between the cells of this data grid and the output values allows the joint predictive information to be quantified. The trade-off between information and reliability is established using a Bayesian model selection approach. We also extend these models to the unsupervised case, where the data grids are nonparametric models of dependence between all the variables, with a piecewise constant estimation of the joint probability distribution. Sophisticated algorithms are necessary to explore the search space of data grid models. They have to strike a balance between the quality of the optimization and the computation time. Several optimization heuristics, including greedy search, meta-heuristic and post-optimization, are introduced to efficiently search the best possible data grid.

The paper is organized as follows. Section 2 summarizes the MODL method in the univariate supervised discretization and value grouping cases. Section 3 extends the approach to the multivariatiate case and Section 4 describes the generalization of such models to unsupervised learning and coclustering. Section 5 presents the optimization algorithms. Section 6 evaluates the data grid models on artificial datasets. Section 7 reports experiments performed on the agnostic learning vs. prior knowledge challenge datasets (Guyon et al., 2007) and analyzes their interest for classification and explanation. Finally, Section 8 gives a summary and discusses future work.

2. The MODL Supervised Discretization and Value Grouping Methods

For the convenience of the reader, this section summarizes the MODL approach in the univariate case, detailed in (Boullé, 2006) for supervised discretization, and in (Boullé, 2005) for supervised value grouping.

2.1 Discretization

The objective of supervised discretization is to induce a list of intervals which partitions the numerical domain of a continuous input variable, while keeping the information relative to the output variable. A trade-off must be found between information quality (homogeneous intervals in regard to the output variable) and statistical quality (sufficient sample size in every interval to ensure generalization).

In the MODL approach, the discretization is turned into a model selection problem. First, a space of discretization models is defined. The parameters of a specific discretization model are the number of intervals, the bounds of the intervals and the frequencies of the output values in each interval. Then, a prior distribution is proposed on this model space. This prior exploits the hierarchy of the parameters: the number of intervals is first chosen, then the bounds of the intervals and finally the frequencies of the output values. The prior is uniform at each stage of the hierarchy. Finally, we assume that the multinomial distributions of the output values in each interval are independent from each other. A

1. The method is available as a shareware, downloadable at http://perso.rd.francetelecom.fr/boulle/

Bayesian approach is applied to select the best discretization model, which is found by maximizing the probability $p(Model|Data)$ of the model given the data. Using the Bayes rule and since the probability $p(Data)$ is constant under varying the model, this is equivalent to maximizing $p(Model)p(Data|Model)$.

Let N be the number of instances, J the number of output values, I the number of input intervals. N_i denotes the number of instances in the interval i and N_{ij} the number of instances of output value j in the interval i. In the context of supervised classification, the number of instances N and the number of classes J are supposed to be known. A discretization model M is then defined by the parameter set $\left\{I, \{N_i\}_{1\leq i\leq I}, \{N_{ij}\}_{1\leq i\leq I, 1\leq j\leq J}\right\}$.

Using the definition of the model space and its prior distribution, Bayes formula can be used to calculate the exact prior probabilities of the models and the probability of the data given a model. Taking the negative log of the probabilities, this provides the evaluation criterion given in Formula 1.

$$\log N + \log \binom{N+I-1}{I-1} + \sum_{i=1}^{I} \log \binom{N_i+J-1}{J-1} + \sum_{i=1}^{I} \log \frac{N_i!}{N_{i1}!N_{i2}!\ldots N_{iJ}!} \qquad (1)$$

The first term of the criterion stands for the choice of the number of intervals and the second term for the choice of the bounds of the intervals. The third term corresponds to the parameters of the multinomial distribution of the output values in each interval and the last term represents the conditional likelihood of the data given the model, using a multinomial term. Therefore "complex" models with large numbers of intervals are penalized.

Once the evaluation criterion is established, the problem is to design a search algorithm in order to find a discretization model that minimizes the criterion. In (Boullé, 2006), a standard greedy bottom-up heuristic is used to find a good discretization. In order to further improve the quality of the solution, the MODL algorithm performs post-optimizations based on hill-climbing search in the neighborhood of a discretization. The neighbors of a discretization are defined with combinations of interval splits and interval merges. Overall, the time complexity of the algorithm is $O(JN \log N)$.

The MODL discretization method for supervised classification provides the most probable discretization given the data. Extensive comparative experiments report high performance (Boullé, 2006).

2.2 Value Grouping

Categorical variables are analyzed in a way similar to that for numerical variables, using a partitioning model of the input values. In the numerical case, the input values are constrained to be adjacent and the only considered partitions are the partitions into intervals. In the categorical case, there are no such constraints between the values and any partition into groups of values is possible. The problem is to improve the reliability of the estimation of the class conditional probabilities owing to a reduced number of groups of values, while keeping the groups as informative as possible. Producing a good grouping is harder with large numbers of input values since the risk of overfitting the data increases. In the extreme situation where the number of values is the same as the number of instances, overfitting is

obviously so important that efficient grouping methods should produce one single group, leading to the elimination of the variable.

Again, let N be the number of instances, V the number of input values, J the number of output values and I the number of input groups. N_i denotes the number of instances in the group i, and N_{ij} the number of instances of output value j in the group i. The Bayesian model selection approach is applied like in the discretization case and provides the evaluation criterion given in Formula 2. This formula has a similar structure as that of Formula 1. The two first terms correspond to the prior distribution of the partitions of the input values, into groups of values in Formula 2 and into intervals in Formula 1. The two last terms are the same in both formula.

$$\log V + \log B(V,I) + \sum_{i=1}^{I} \log \binom{N_i + J - 1}{J-1} + \sum_{i=1}^{I} \log \frac{N_i!}{N_{i1}!N_{i2}!\ldots N_{iJ}!} \tag{2}$$

$B(V,I)$ is the number of divisions of V values into I groups (with eventually empty groups). When $I = V$, $B(V,I)$ is the Bell number. In the general case, $B(V,I)$ can be written as $B(V,I) = \sum_{i=1}^{I} S(V,i)$, where $S(V,i)$ is the Stirling number of the second kind (see Abramowitz and Stegun, 1970), which stands for the number of ways of partitioning a set of V elements into i nonempty sets. In (Boullé, 2005), a standard greedy bottom-up heuristic is proposed to find a good partition of the input values. Several pre-optimization and post-optimization steps are incorporated, in order to both ensure an algorithmic time complexity of $O(JN\log(N))$ and obtain accurate value groupings.

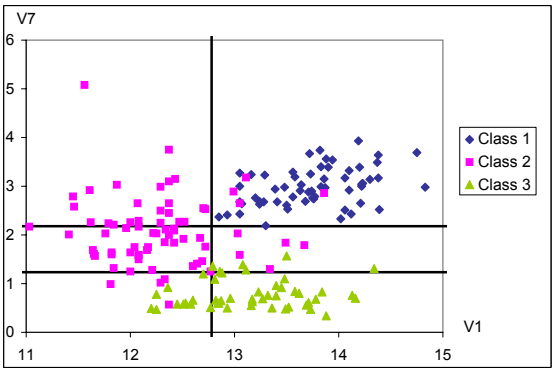
3. Supervised Data Grids Models for any Subset of Variables

In this section, we describe the extension of the MODL approach to pairs of variables introduced in (Boullé, 2007a) and generalize it to any subset of input variables variables for supervised learning, in the numerical, categorical and mixed type case. We first introduce the approach using an illustrative example for the case of supervised bivariate discretization, then summarizes the principles of the extension in the general case, and present the evaluation criterion of such models. Finally, we relate our modeling approach to information theory and discuss the robustness of our method.

3.1 Interest of the joint partitioning of two input variables

Figure 1 gives a multiple scatter plot (per class value) of the input variables V1 and V7 of the wine dataset (Blake and Merz, 1996). This diagram shows the conditional probability of the output values given the pair of input variables. The V1 variable taken alone cannot separate Class 1 from Class 3 for input values greater than 13. Similarly, the V7 variable is a mixture of Class 1 and Class 2 for input values greater than 2. Taken jointly, the two input variables allow a better separation of the class values.

Extending the univariate case, we partition the dataset on the cross-product of the input variables to quantify the relationship between the input and output variables. Each input variable is partitioned into a set of *parts* (intervals in the numerical case). The cross-product of the univariate input partitions defines a *data grid*, which partitions the instances into a set of *data cells*. Each data cell is defined by a pair of parts. The connection between the



]2.18, +∞[	(0, 23, 0)	(59, 0, 4)
]1.235, 2.18]	(0, 35, 0)	(0, 5, 6)
]-∞, 1.235]	(0, 4, 11)	(0, 0, 31)
V7xV1	]-∞, 12.78]	]12.78, +∞[

Figure 1: Multiple scatterplot (per class value) of the input variables V1 and V7 of the wine dataset. The optimal MODL supervised bivariate partition of the input variables is drawn on the multiple scatterplot, and the triplet of class frequencies per data grid cell is reported in the right table

input variables and the output variable is evaluated using the distribution of the output values in each cell of the data grid. It is worth noting that the considered partitions can be factorized on the input variables. For instance in Figure 1, the V1 variable is discretized into 2 intervals (one bound at 12.78) and the V7 variable into 3 intervals (two bounds at 1.235 and 2.18). The instances of the dataset are distributed in the resulting bidimensional data grid. In each cell of the data grid, we consider the empirical distribution of the output values. For example, the cell defined by the intervals $]12.78, +\infty[$ on V1 and $]2.18, +\infty[$ on V7 contains 63 instances. These 63 instances are distributed on 59 instances for Class 1 and 4 instances for Class 3. Coarse grain data grids tend to be reliable, whereas fine grain data grids allow a better separation of the output values. In our example, the MODL optimal data grid is drawn on the multiple scatter plot on Figure 1.

3.2 Principles of the Extension to Data Grid Models

The MODL approach has been studied in the case of univariate supervised partitioning for numerical variables (Boullé, 2006) and categorical variables (Boullé, 2005). The extension to the multivariate case applies the same principles as those described in section 3.1. Each input variable is partitioned, into intervals in the numerical case and into groups of values in the categorical case. Taking the cross-product of the univariate partitions, we obtain a data grid of input cells, the content of which characterizes the distribution of the output values. Compared to the bivariate case, we introduce a new level in the hierarchy of the model parameters, related to variable selection. Indeed, a multivariate data grid model implicitly handles variables selection, where the selected variables that bring predictive information are partitioned in at least two parts. The other variables, the partition of which consists of one single part, can be considered as irrelevant and discarded.

The space of multivariate data grid models is very large. Selecting the best model is a difficult task, both from a model selection and optimization point of view. In our approach, we:

1. precisely define the parameters of the data grid models,
2. define a prior on the model parameters,
3. establish an analytic criterion to evaluate the posterior probability of each model
4. design sophisticated optimization algorithm to search the maximum a posteriori (MAP) model.

Our space of models is data dependent: we exploit the input data in order to define the model parameters and restrict their range. Note that our space of models is both non-parametric (the number of parameters increase with the size of the data) and finite (each parameter is discrete with a range bounded according to the input data). To select the best model, we adopt a Bayesian approach and define a prior distribution on the model parameters. Following the principle of parsimony, our prior exploits the hierarchy of the parameters and is uniform at each stage of this hierarchy. We then obtain an analytic formula that evaluates the exact posterior probability of each data grid model. Finally, we exploit the combinatorial algorithm described in section 5 to efficiently search the space of data grid models.

3.3 Evaluation Criterion for Supervised Data Grids

We present in Definition 1 a family of multivariate partitioning models and select the best model owing to a Bayesian model selection approach.

Definition 1 *A data grid classification model is defined by a subset of selected input variables, for each selected variable by a univariate partition, into intervals in the numerical case and into groups of values in the categorical case, and by a multinomial distribution of the output values in each cell of the data grid resulting from the cross-product of the univariate partitions.*

Notation.

- N: number of instances,
- Y: output variable,
- J: number of output values,
- $X_1, \ldots, X_K$: input variables,
- K: number of input variables,
- $\mathcal{K}$: set of input variables ($|\mathcal{K}| = K$),
- $\mathcal{K}_n$: subset of numerical input variables,
- $\mathcal{K}_c$: subset of categorical input variables,
- $V_k, k \in \mathcal{K}_c$: number of values of the categorical input variable X_k,
- K_s: number of selected input variables,
- $\mathcal{K}_s$: subset of selected input variables ($|\mathcal{K}_s| = K_s$),
- I_k: number of parts (intervals or groups of values) in the univariate partition of input variable X_k,
- $N_{i_1 i_2 \ldots i_K}$: number of instances in the input data cell $(i_1, i_2, \ldots, i_K)$,

- $N_{i_1 i_2 \ldots i_K j}$: number of instances of output value j in the input data cell $(i_1, i_2, \ldots, i_K)$.

Like the bivariate case, presented in Section 3.1, any input information is used to define the family of the model. For example, the numbers of instances per cell $N_{i_1 i_2 \ldots i_K}$ do not belong to the parameters of the data grid model: they are derived from the definition of the univariate partitions of the selected input variables and from the dataset. These numbers of instances allow the specification of the multinomial distribution of the output values in each input cell to be constrained.

We now introduce in Definition 2 a prior distribution on the parameters of the data grid models.

Definition 2 *The hierarchical prior of the data grid models is defined as follows:*

- *the number of selected input variables is uniformly distributed between* 1 *and* K*,*
- *for a given number* K_S *of selected input variables, the subsets of* K_S *variables are uniformly distributed (with replacement),*
- *the numbers of input parts, are independent from each other, and uniformly distributed between* 1 *and* N *for numerical variables, between* 1 *and* V_k *for categorical variables,*
- *for each numerical input variable and for a given number of intervals, every partition into intervals is equiprobable,*
- *for each categorical input variable and for a given number of groups, every partition into groups is equiprobable,*
- *for each cell of the input data grid, every distribution of the output values is equiprobable,*
- *the distributions of the output values in each cell are independent from each other.*

Applying the MODL approach, this prior exploits the hierarchy of the parameters and is uniform at each stage of this hierarchy.

For the variable selection parameters, we reuse the prior introduced by Boullé (2007b) in the case of the selective naive Bayes classifier. We first choose the number of variables and second the subset of selected variables. For the number of selected variables K_s, we adopt a uniform prior between 0 and K variables, representing $(K+1)$ equiprobable alternatives. For the choice of the K_s variables, we assign the same probability to every subset of K_s variables. The number of combinations $\binom{K}{K_s}$ seems the natural way to compute this prior, but it has the disadvantage of being symmetric. Beyond $K/2$ variables, adding variables is favored. As we prefer simpler models, we propose to use the number of combinations with replacement $\binom{K+K_s-1}{K_s}$, which leads to a prior probability decreasing with the number of variables.

For the specification of each univariate partition, we reuse the prior presented by Boullé (2006) for supervised discretization of numerical variables and by Boullé (2005) for supervised value grouping of categorical variables (see Section 2). We apply the Bayesian model

selection approach and obtain the evaluation criterion of a data grid model in Formula 3.

$$\begin{aligned}
&\log(K+1) + \log \binom{K+K_s-1}{K_s} \\
&+ \sum_{k \in \mathcal{K}_s \cap \mathcal{K}_n} \left(\log N + \log \binom{N+I_k-1}{I_k-1} \right) + \sum_{k \in \mathcal{K}_s \cap \mathcal{K}_c} (\log V_k + \log B(V_k, I_k)) \\
&+ \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_K=1}^{I_K} \log \binom{N_{i_1 i_2 \ldots i_K} + J - 1}{J-1} \\
&+ \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_K=1}^{I_K} \left(\log N_{i_1 i_2 \ldots i_K}! - \sum_{j=1}^{J} \log N_{i_1 i_2 \ldots i_K j}! \right)
\end{aligned} \tag{3}$$

The first line in Formula 3 corresponds to the prior for variable selection. As in the univariate case, the second line is related to the prior probability of the discretization parameters (like in Formula 1) for the selected numerical input variables and to that of the value grouping parameters (like in Formula 2) for the selected categorical input variables. The binomial terms in the third line represent the choice of the multinomial distribution of the output values in each cell of the input data grid. The multinomial terms in the last line represent the conditional likelihood of the output values given the data grid model.

3.4 Relation with Information Theory

Let us first introduce the null model $M_\emptyset$, where no input variable is selected. The null model is composed of a single cell containing all the instances. Applying Formula 3, the cost $c(M_\emptyset)$ of the null model (its value according to evaluation criterion 3) reduces to

$$c(M_\emptyset) = \log(K+1) + \log \binom{N+J-1}{J-1} + \log \frac{N!}{N_1! N_2! \ldots N_J!},$$

where N_j denotes the frequency of the output value j. This corresponds to the posterior probability of a multinomial model of the output variable, independently of any input variable. To get an asymptotic evaluation of the cost of the null model, we now introduce the Shannon entropy $H(Y)$ (Shannon, 1948) of the output variable, $H(Y) = -\sum_{j=1}^{J} p_j \log p_j$, where p_j if the prior probability of the output value j. Using the approximation $\log N! = N(\log N - 1) + O(\log N)$ based on Stirling's formula, the cost of the null model is asymptotically equivalent to N times the Shannon entropy of the output variable:

$$c(M_\emptyset) = NH(Y) + O(\log N). \tag{4}$$

As the negative log of a probability can be interpreted as a coding length (Shannon, 1948), our model selection technique is closely related to the minimum description length (MDL) approach (Rissanen, 1978; Hansen and Yu, 2001; Grünwald et al., 2005), which aims to approximate the Kolmogorov complexity (Li and Vitanyi, 1997) for the coding length of the output data. The Kolmogorov complexity is the length of the shortest computer program that encodes the output data given the input data.

Overall, our prior approximates the Kolmogorov complexity of the data grid model given the input data and our conditional likelihood encodes the output values given the data grid model. In our approach, the choice of the null model corresponds to the lack of predictive information. The coding length of the null model is asymptotically equivalent to the Shannon entropy the output data (cf. Formula 4), which corresponds to a basic encoding of the output data, with no use of the input data. This is close to the idea of Kolmogorov, who considers data to be random if its algorithmic complexity is high, that is if it cannot be compressed significantly. This makes our approach very robust, since detecting predictive information using data grid models is necessarily related to a coding length better than that of the null model, thus to non random patterns according Kolmogorov's definition of randomness. This robustness has been confirmed using extensive experiments in the univariate case (Boullé, 2006, 2005), and is evaluated in the multivariate case in Section 6.

4. Data Grid Models for Coclustering of Instances and Variables

In (Boullé, 2008b), we have investigated the extension of data grid models to unsupervised learning, in order to evaluate the joint probability distribution of any subset of variables, numerical or categorical. In Section 4.1, we present a detailed description of these models in the case of two categorical variables. In Section 4.2, we show how to apply such bivariate categorical models to the problem of coclustering the instances and variables of a dataset, as a data preparation technique for supervised learning.

4.1 Bivariate Value Grouping of Categorical Variables

In this section, we focus on the case of two categorical variables. We introduce unsupervised bivariate data grid models and their evaluation criterion. We then show how such models can be interpreted as nonparametric models of the correlation between the variables.

4.1.1 Presentation

D	∅	•	∅	•
C	•	∅	•	∅
B	∅	•	∅	•
A	•	∅	•	∅
	a	b	c	d

{B, D}	∅	•
{A, C}	•	∅
	{a, c}	{b, d}

Figure 2: Example of joint density for two categorical variables Y_1 having 4 values a, b, c, d and Y_2 having 4 values A, B, C, D. The initial contingency table on the left contains instances only for one half of the cells (tagged as •), and the remaining cells are empty. After the bivariate value grouping, the preprocessed contingency table on the right provides a synthetic description of the correlation between Y_1 et Y_2.

Our objective is to provide a joint description of two categorical variables Y_1 et Y_2, as illustrated in Figure 2. In the case of categorical variables with many values, the contin-

gency table between the variables is sparse and does not allow the identification of reliable correlations. Standard statistical tests rely on approximations which are valid only asymptotically. For example, the chi-square test requires an expected frequency of at least 5 in each cell of the contingency table (Cochran, 1954; Connor-Linton, 2003), which does not permit its application in sparse cases. Grouping the values of each variable allows the cell frequencies to be raised (at the expense of potentially mixing interesting patterns), and gives greater confidence in the observed correlation. However, since many grouping models might be considered, there is a risk of overfitting the data. The issue is to find a trade-off between the quality of the density estimation and the generalization ability, on the basis of the granularity of the grid.

4.1.2 Formalization

Our objective is to describe the joint distribution of the data, which turns into describing the value of the instances for each variable. We introduce a family of unsupervised partitioning models, based on groups of values for each variable and on a multinomial distribution of all the instances on the cells of the resulting data grid. This family of models is formalized in Definition 3.

Definition 3 *An unsupervised bivariate value grouping model is defined by:*

- *a number of groups for each variable,*
- *for each variable, the repartition of the values into the groups of values,*
- *the distribution of the instances of the data sample among the cells of the resulting data grid,*
- *for each variable and each group, the distribution of the instances of the group on the values of the group.*

Notation.

- Y_1, Y_2: variables (both considered as output variables)
- V_1, V_2: number of values for each variable (assumed as prior knowledge)
- N: number of training instances
- $D = \{D_1, D_2, \ldots, D_n\}$: training instances
- J_1, J_2: number of groups for each variable
- $G = J_1 J_2$: number of cells in the resulting data grid
- $j_1(v_1), j_2(v_2)$: index of the group containing value v_1 (resp. v_2)
- m_{j_1}, m_{j_2}: number of values in group j_1 (resp. j_2)
- n_{v_1}, n_{v_2}: number of instances for value v_1 (resp. v_2)
- N_{j_1}: number of instances in the group j_1 of variable Y_1
- N_{j_2}: number of instances in the group j_2 of variable Y_2
- $N_{j_1 j_2}$: number of instances in the cell (j_1, j_2) of the data grid

We assume that the numbers of values V_1 and V_2 per categorical variable are known in advance and we aim to model the joint distribution of the finite data sample of size N on these values. The family of models introduced in Definition 3 is completely defined by the parameters describing the partition of the values into groups of values

$$J_1, J_2, \{j_1(v_1)\}_{1 \leq v_1 \leq V_1}, \{j_2(v_2)\}_{1 \leq v_2 \leq V_2},$$

by the parameters of the multinomial distribution of the instances on the data grid cells

$$\{N_{j_1 j_2}\}_{1 \leq j_1 \leq J_1, 1 \leq j_2 \leq J_2},$$

and by the parameters of the multinomial distribution of the instances of each group on the values of the group

$$\{n_{v_1}\}_{1 \leq v_1 \leq V_1}, \{n_{v_2}\}_{1 \leq v_2 \leq V_2}.$$

The numbers of values per groups m_{j_1} and m_{j_2} are derived from the specification of the partitions of the values into groups: they do not belong to the model parameters. Similarly, the number of instances in each group can be deduced by adding the cell frequencies in the rows or columns of the grid, according to $N_{j_1} = \sum_{j_2=1}^{J_2} N_{j_1 j_2}$ and $N_{j_2} = \sum_{j_1=1}^{J_1} N_{j_1 j_2}$.

In order to select the best model, we apply a Bayesian approach, using the prior distribution on the model parameters described in Definition 4.

Definition 4 *The prior for the parameters of an unsupervised bivariate value grouping model are chosen hierarchically and uniformly at each level:*

- *the numbers of groups J_1 and J_2 are independent from each other, and uniformly distributed between 1 and V_1 for Y_1, between 1 and V_2 for Y_2,*
- *for a given number of groups J_1 of Y_1, every partition of the V_1 values into J_1 groups is equiprobable,*
- *for a given number of groups J_2 of Y_2, every partition of the V_2 values into J_2 groups is equiprobable,*
- *for a data grid of given size (J_1, J_2), every distribution of the N instances on the $G = J_1, J_2$ cells of the grid is equiprobable,*
- *for a given group of a given variable, every distribution of the instances of the group on the values of the group is equiprobable.*

Taking the negative log of the probabilities, this provides the evaluation criterion given in Theorem 5.

Theorem 5 *An unsupervised bivariate value grouping model distributed according to a uniform hierarchical prior is Bayes optimal if the value of the following criteria is minimal*

$$\begin{aligned}
&\log V_1 + \log V_2 + \log B(V_1, J_1) + \log B(V_2, J_2) \\
&+ \log \binom{N+G-1}{G-1} + \sum_{j_1=1}^{J_1} \log \binom{N_{j_1}+m_{j_1}-1}{m_{j_1}-1} + \sum_{j_2=1}^{J_2} \log \binom{N_{j_2}+m_{j_2}-1}{m_{j_2}-1} \\
&+ \log N! - \sum_{j_1=1}^{J_1} \sum_{j_2=1}^{J_2} \log N_{j_1 j_2}! \\
&+ \sum_{j_1=1}^{J_1} \log N_{j_1}! + \sum_{j_2=1}^{J_2} \log N_{j_2}! - \sum_{v_1=1}^{V_1} \log n_{v_1}! - \sum_{v_2=1}^{V_2} \log n_{v_2}!
\end{aligned} \tag{5}$$

The first line in Formula 5 relates to the prior distribution of the group numbers J_1 et J_2 and to the specification the partition of the values in groups for each variable. These terms are the same as in the case of the MODL supervised univariate value grouping method (Boullé, 2005), summarized in Section 2.2. The second line in Formula 5 represents the specification of the parameters of the multinomial distribution of the N instances on the G cells of the data grid, followed by the specification of the multinomial distribution of the instances of each group on the values of the group. The third line stands for the likelihood of the distribution of the instances on the data grid cells, by the mean of a multinomial term. The last line corresponds to the likelihood of the distribution of the values locally to each group, for each variable.

4.1.3 Interpretation

The null model $M_\emptyset$ contains one single cell and Formula 5 reduces to

$$\begin{aligned}
c(M_\emptyset) = &\log V_1 + \log V_2 + \log \binom{N+V_1-1}{V_1-1} + \log \binom{N+V_2-1}{V_2-1} \\
&+ \log \frac{N!}{n_{v_1}! n_{v_2}! \ldots n_{V_1}!} + \log \frac{N!}{n_{v_1}! n_{v_2}! \ldots n_{V_2}!}
\end{aligned} \tag{6}$$

which corresponds to the posterior probability of the multinomial model for the distribution of the instances on the values, for each variable. This means that each variable is described independently.

More complex data grid models allow a nonparametric description of the correlation between the variables, by the means of cells where groups of values are correlated. The penalization of the model is balanced by a shorter description of each variable given the model. The best trade-of is searched using a Bayesian model selection approach.

Example with two identical categorical variables. Let us consider two identical categorical variables $Y_1 = Y_2$ and the maximum data grid model M_{Max} with as many groups as values ($J_1 = V_1$), as illustrated in Figure 3. The evaluation criterion of the data grid is equal to

$$c(M_{Max}) = 2 \log V_1 + 2 \log B(V_1, V_1) + \log \binom{N+V_1^2-1}{V_1^2-1} + \log \frac{N!}{n_{v_1}! n_{v_2}! \ldots n_{V_1}!} \tag{7}$$

d	∅	∅	∅	•
c	∅	∅	•	∅
b	∅	•	∅	∅
a	•	∅	∅	∅
	a	b	c	d

Figure 3: Bivariate value grouping data grid with as many groups as values for two identical categorical variables $Y_1 = Y_2$, having four values a, b, c and d.

If we compare $c(M_\emptyset)$ in Formula 6 to $c(M_{Max})$ in Formula 7 in the case of two identical categorical variables, we observe an overhead in the prior terms of the maximum model (specification of the value grouping with Bell numbers and specification of the distribution of the N instances on the ${V_1}^2$ cells of the grid). On the other hand, the likelihood term is divided by a factor two: since the correlation between the variables is perfectly detected using the data grid model, describing the joint distribution of the data given the model reduces to describing the distribution of one single variable.

Let us now compare Formulae (6) and (7) in the asymptotic case. The multinomial term for the distribution of the values of a categorical variable can be approximated with

$$\log \frac{N!}{n_{v_1}! n_{v_2}! \ldots n_{V_1}!} \approx N H(Y_1),$$

where $H(Y_1)$ is the Shannon entropy of variable Y_1 (Shannon, 1948). In the case of the null model having one single cell, we get

$$c(M_\emptyset) \approx 2(V_1 - 1) \log N + 2NH(Y_1).$$

In the case of the maximum model with as many groups as values, we obtain

$$c(M_{Max}) \approx (V_1^2 - 1) \log N + NH(Y_1).$$

The maximum model, which detects the correlation between the variables, will thus be preferred as soon as there are enough instances compared to the number of values. It is worth noting that Formulae (6) and (7) allow us to select the best model in the non-asymptotic case.

4.2 Coclustering of Instances and Variables

In this section, we first introduce the application of unsupervised bivariate data grids to the coclustering problem, and then describe how to build a classifier on the basis of a coclustering model.

4.2.1 Coclustering

A coclustering (Hartigan, 1972) is the simultaneous clustering of the rows and columns of a matrix. In case of sparse binary datasets, coclustering is an appealing data preparation

technique to identify the correlation between clusters of instances and clusters of variables (Bock, 1979).

Let us consider a sparse binary dataset with N instances, K variables and V non-null values. A sparse dataset can be represented in tabular format, with two columns and V rows. This corresponds to a new *dataset* with two *variables* named "Instance ID" and "Variable ID" where each *instance* is a couple of values (Instance ID, Variable ID), like in Figure 4.

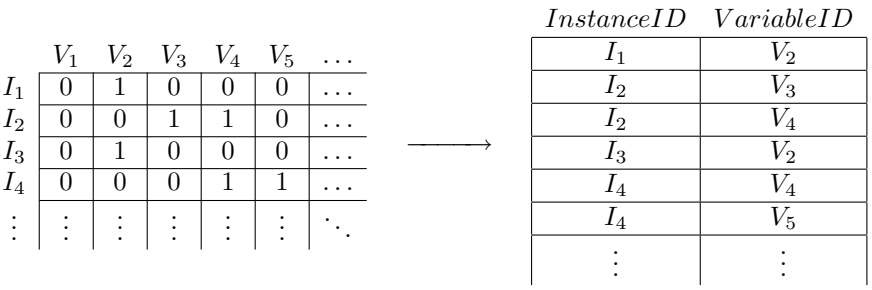


Figure 4: Sparse binary dataset: from the sparse (instances × variables) table to the dense bivariate representation.

The application of bivariate unsupervised data grid models forms groups of instance IDs and groups of variable IDs, so as to maximize the correlation between instances and variables. We expect to find "natural" patterns both in the space of instances and in the space of variables. It is worth noting that the clusters retrieved by data grid models are non-overlapping, since they form a partition of the whole dataset.

4.2.2 Application to Semi-Supervised Learning

We apply a semi-supervised learning approach (Chapelle et al., 2006) to exploit all the data from the train, validation and test datasets. In the first step, all of the instances are processed without any output label to identify the "natural" clusters of instances owing to the data grid coclustering technique. In a second step, the available labeled instances are used to describe the output distribution in each cluster of instances. The label of a test instance is then predicted according to the output distribution of its cluster.

Preprocessing the data with semi-supervised coclustering makes sense under the assumption that the "natural" clusters are correlated with the output values (predefined clusters). We expect this assumption to be true for some datasets, especially in the pattern recognition domain.

5. Optimization Algorithm for Multivariate Data Grids

The space of data grid models is so large that straightforward algorithms will almost surely fail to obtain good solutions within a practicable computational time.

Given that the MODL criterion is optimal, the design of sophisticated optimization algorithms is both necessary and meaningful. In this section, we describe such algorithms. They finely exploit the sparseness of the data grids and the additivity of the MODL criterion, and allow a deep search in the space of data grid models with $O(KN)$ memory complexity and $O(N\sqrt{N}\log N \max(K, \log N))$ time complexity.

5.1 Greedy Bottom-Up Heuristic

Let us first focus on the case of numerical input variables. The optimization of a data grid is a combinatorial problem. For each input variable X_k, there are 2^N possible univariate discretizations, which represents $\left(2^N\right)^K$ possible multivariate discretizations. An exhaustive search over the whole space of models is unrealistic.

We describe in Algorithm 1 a greedy bottom up merge heuristic (GBUM) to optimize the data grids. The method starts with the maximum data grid M_{Max}, which corresponds to the finest possible univariate partitions, based on single value parts, intervals or groups. It evaluates all the merges between adjacent parts for any variables (ties are broken randomly), and performs the best merge if the evaluation criterion decreases after the merge. The process is iterated until no further merge can decrease the criterion.

Algorithm 1 Greedy Bottom-Up Merge heuristic (GBUM)

Require: M {Initial data grid solution}
Ensure: $M^*, c(M^*) \leq c(M)$ {Final solution with improved cost}
$M^* \leftarrow M$
while improved solution **do**
  {Evaluate all the merges between adjacent parts}
  $c^* \leftarrow \infty, m^* \leftarrow \emptyset$
  for all Variable $X_k \in \mathcal{K}$ **do**
    for all Merge m between two adjacent parts of variable X_k **do**
      $M' \leftarrow M^* + m$ {Evaluate merge m on data grid M^*}
      if $c(M') < c^*$ **then**
        $c^* \leftarrow c(M'), m^* \leftarrow m$
      end if
    end for
  end for
  {Perform best merge}
  if $c^* < c(M^*)$ **then**
    $M^* \leftarrow M^* + m^*$
  end if
end while

Each evaluation of a data grid requires $O(N^K)$ time, since the initial data grid model M_{Max} contains N^K cells. Each step of the algorithm relies on $O(N)$ evaluations of interval merges times the number K of variables. There are at most $O(KN)$ steps, since the data grid becomes equal to the null model $M_\emptyset$ (one single cell) once all the possible merges have been performed. Overall, the time complexity of the algorithm is $O(K^2N^2N^K)$ using a straightforward implementation of the algorithm. However, the GBUM algorithm can be

optimized in $O(K^2 N \log N)$ time, as shown in next section and demonstrated in (Boullé, 2008a) in the bivariate case.

5.2 Optimized Implementation of the Greedy Heuristic

The optimized algorithm mainly exploits the sparseness of the data and the additivity of the evaluation criterion. Although a data grid may contain $O(N^K)$ cells, at most N cells are non empty. Thus, each evaluation of a data grid can be performed in $O(N)$ time owing to a specific algorithmic data structure.

The additivity of the evaluation criterion means that it can be decomposed according to Definition 6 on the hierarchy of the components of the data grid: grid size, variables, parts and cells.

Definition 6 *An evaluation criterion $c(M)$ of a data grid model M is additive if it can be decomposed as a sum of terms according to*

$$c(M) = c^{(G)}(\mathcal{I}) + \sum_{k=1}^{K} c^{(V)}(X_k, I_k) + \sum_{k=1}^{K} \sum_{i_k=1}^{I_k} c^{(P)}\left(P_{i_k}^{(k)}\right) + \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_k=1}^{I_K} c^{(C)}(C_{i_1 i_2 \ldots i_K})$$

where

- *the grid criterion $c^{(G)}(\mathcal{I})$ relies only on the sizes $\mathcal{I} = \{I_1, I_2, \ldots, I_K\}$ of the univariate partitions of the data grid,*
- *the variable criterion $c^{(V)}(X_k, I_k)$ relies only on features of the input variable X_k and on the number of parts I_k of its partition,*
- *the part criterion $c^{(P)}\left(P_{i_k}^{(k)}\right)$ for each part $P_{i_k}^{(k)}$ of the univariate partition of the input variable X_k relies only on features of the part,*
- *the cell criterion $c^{(C)}(C_{i_1 i_2 \ldots i_K})$ for each cell $C_{i_1 i_2 \ldots i_K}$ of the data grid relies only on features of the cell, and is null for empty cells.*

One can easily check that the evaluation criteria introduced in Formula 3 or Formula 5 are additive. Using this additivity property, all the merges between adjacent parts can be evaluated in $O(N)$ time. Furthermore, when the best merge is performed, the only merges that need to be re-evaluated for the next optimization step are the merges that share instances with the best merge. Since the data grid is sparse, the number of partial re-evaluations of the criterion is limited by the number of instances, not by the number of cells in the data grids. Sophisticated algorithmic data structures and algorithms, detailed in (Boullé, 2008a), are necessary to exploit these optimization principles and guarantee a time complexity of $O(K^2 N \log N)$.

5.3 Post-Optimization

The greedy heuristic is time efficient, but it may fall into a local optimum. First, the greedy heuristic may stop too soon and produce too many parts for each input variable. Second, the boundaries of the intervals may be sub-optimal since the merge decisions of the greedy

heuristic are never reconsidered. We propose to reuse the post-optimization algorithms described in (Boullé, 2006) in the case of univariate discretization.

In a first stage called *exhaustive merge*, the greedy heuristic merge steps are performed without referring to the stopping condition until the data grid consists of one single cell. The best encountered data grid is then memorized. This stage allows escaping local minima with several successive merges and needs $O(K^2 N \log N)$ time.

In a second stage called *greedy post-optimization*, a hill-climbing search is performed in the neighborhood of the best data grid. This search alternates the optimization on each input variable. For each given input X_k, we freeze the partition of all the other input variables and optimize the partition of X_k. Since a multivariate additive criterion turns out to be an univariate additive criterion once all except one univariate partitions are frozen, we reuse the post-optimization algorithms described in (Boullé, 2006) for univariate discretizations. This process is repeated for all variables until no further improvement can be obtained. This algorithm converges very quickly in practice and requires only a few steps.

We summarize the post-optimization of data grids in Algorithm 2.

Algorithm 2 Post-optimization of a Data Grid

Require: M {Initial data grid solution}
Ensure: $M^*; c(M^*) \leq c(M)$ {Final solution with improved cost}
1: $M^* \leftarrow$ call *exhaustive merge* (M)
2: **while** improved solution **do**
3: {Take a random permutation of $\mathcal{K}$}
4: **for all** Variable $X_k \in \mathcal{K}$ **do**
5: Freeze the univariate partition of all the variables except X_k
6: $M^* \leftarrow$ call *univariate post-optimization* (M^*) for variable X_k
7: **end for**
8: **end while**

5.4 Meta-Heuristic

Since the GBUM algorithm is time efficient, it is then natural to apply it several times in order to better explore the search space. This is done according to the *variable neighborhood search* (VNS) meta-heuristic introduced by Hansen and Mladenovic (2001), which consists of applying the primary heuristic to a random neighbor of the solution. If the new solution is not better, a bigger neighborhood is considered. Otherwise, the algorithm restarts with the new best solution and a minimal size neighborhood. The process is controlled by the maximum length of the series of growing neighborhoods to explore.

For the primary heuristic, we choose the greedy bottom-up heuristic followed by the post-optimization heuristic. In order to "purify" the randomly generated solutions given to the primary heuristic, we also incorporate a pre-optimization heuristic, that exploits the same principle as the post-optimization heuristic.

This meta-heuristic is described in Algorithm 3. According to the level of the neighborhood size l, a new solution M' is generated close to the current best solution. We define the structure of neighborhood by exploiting at most $K_{Max} = \log_2 N$ new variables. For each

exploited variable, a random discretization is obtained with the choice of random interval bounds without replacement, with at most $I_{Max} = N^{\frac{1}{K_{Max}}}$ intervals. This heuristic choice for the maximum neighborhood size results from the analysis of Formula 3. In the case of two equidistributed output values, if we have selected K_{Max} variables with I_{Max} intervals per variable and exactly one instance per input cell, the cost of the model is slightly worse than that of the null model with no selected variable. This means that data grids that are too sparse are not likely to be informative according to Formula 3.

The VNS meta-heuristic only requires the number of sizes of neighborhood as a parameter. This can easily be turned into an anytime optimization algorithm, by calling the VNS algorithm iteratively with parameters of increasing size and stopping the optimization only when the allocated time is elapsed. In this paper, all the experiments are performed by calling the VNS algorithm with successive values of $1, 2, 4, \ldots, 2^T$ for the parameter $MaxLevel$.

In order to improve the initial solution, we choose to first optimize the univariate partition of each variable and to build the initial solution from a cross-product of the univariate partitions. Although this cannot help in case of strictly bivariate patterns (such as XOR for example), this might be helpful otherwise.

Algorithm 3 VNS meta-heuristic for data grid optimization

Require: M {Initial data grid solution}
Require: $MaxLevel$ {Optimization level}
Ensure: $M^*, c(M^* \leq c(M)$ {Final solution with improved cost}
1: $Level \leftarrow 1$
2: **while** $Level \leq MaxLevel$ **do**
3: {Generate a random solution in the neighborhood of M^*}
4: $M'' \leftarrow$ random solution with $K_s = \frac{Level}{MaxLevel} \log_2 N$ new selected variables and $\frac{Level}{MaxLevel} N^{\frac{1}{K_s}}$ new intervals per selected variable
5: $M' \leftarrow M^* \cup M''$
6: {Optimize and evaluate the new solution}
7: $M' \leftarrow$ call *Pre-Optimization*(M')
8: $M' \leftarrow$ call *Greedy Bottom-Up Merge*(M')
9: $M' \leftarrow$ call *Post-Optimization*(M')
10: **if** $c(M') < c(M^*)$ **then**
11: $M^* \leftarrow M'$
12: $Level \leftarrow 1$
13: **else**
14: $Level \leftarrow Level + 1$
15: **end if**
16: **end while**

5.5 The Case of Categorical Variables

In the case of categorical variables, the combinatorial problem is worse still for large numbers of values V. The number of possible partitions of the values is equal to the Bell number

$B(V) = \frac{1}{e}\sum_{k=1}^{\infty}\frac{k^V}{k!}$ which is far greater than the $O(2^N)$ possible discretizations. Furthermore, the number of possible merges between parts is $O(V^2)$ for categorical variables instead of $O(N)$ for numerical variables. Specific pre-processing and post-processing heuristics are necessary to efficiently handle the categorical input variables. Mainly, the number of groups of values is bounded by $O(\sqrt{N})$ in the algorithms, and the initial and final groupings are locally improved by the exchange of values between groups. This allows us to keep an $O(N)$ memory complexity per variable and bound the time complexity by $O(N\sqrt{N}\log N)$ per categorical variable, with an overall time complexity of $O(K^2N\sqrt{N}\log N)$ for the complete greedy heuristic.

5.6 Summary of the Optimization Algorithms

The optimization of multivariate data grid models can be summarized as an extension of the univariate discretization and value grouping algorithms to the multivariate case.

The main heuristic is a greedy bottom-up heuristic, which starts from an initial fine grain data grid and iteratively performs the best merges between two adjacent parts of any input variable. Post-optimizations are carried out to improve the best data grid, by exploiting a local neighborhood of the solution. The main optimization heuristic (surrounded by pre-optimization and post-optimization steps) is run from several initial solutions, coming from the exploration of a global neighborhood of the best solution using a meta-heuristic.

These algorithms are efficiently implemented, on the basis of two main properties of the problem: the additivity of the criterion, which consists of a sum of independent terms related to the dimension of the data grid, the variables, the parts and the cells, and the sparseness of the data grids, which contain at most N non empty cells for $O(N^K)$ cells. Furthermore, in the meta-heuristic, we restrict to data grids with at most $K_{Max} = \log_2 N$ variables, which reduces the time complexity of the main greedy heuristic.

Sophisticated algorithms, detailed in (Boullé, 2008a), are necessary to make the most of these problem properties and to reach the following algorithmic performance:

- $O(KN)$ memory complexity for K variables and N instances,
- $O(KN\log N\max(K,\log N))$ if all the input variables are numerical,
- $O(KN\sqrt{N}\log N\max(K,\log N))$ in the general case of numerical variables and categorical variables having large number of input values ($V \geq \sqrt{N}$).

6. Experiments on Artificial Datasets

In the bivariate case, the data grid models have been intensively evaluated on artificial and real datasets in (Boullé, 2007a). In this section, we evaluate the multivariate data grid models on artificial datasets, where the true data distribution is known. Two patterns are considered: noise and multivariate XOR. This enables the evaluation of both the reliability of the method and its rate of convergence for the detection of complex patterns. We also analyze the effect of each part of the algorithm and study the limits of the method.

6.1 The Noise Pattern

The purpose of the noise pattern experiment is to evaluate the noise resistance of the method, under varying sample size and the number of input variables. The noise pattern

consists of an output variable independent from the input variables. The expected data grid contains one single cell, meaning that the output distribution is independent from the input variables. The output variable is equidistributed on two values. The experiment is performed on a set of sample sizes ranging from 2 to 1000 instances, for 1, 2 and 10 numerical input variables uniformly distributed on the [0, 1] numerical domain. The criterion evaluated is the number of cells in the data grid. In order to obtain reliable results, the experiment is performed one million times on randomly generated training datasets for each sample size and number of input variables. In order to study the impact of variable selection in the prior distribution of the models (terms $\log(K+1) + \log\binom{K+K_s-1}{K_s}$ in Formula 3), we perform the experiment with and without the variable selection terms. Figure 5 presents the mean cell number for each sample size and number of input variable, with and without the prior for variable selection.

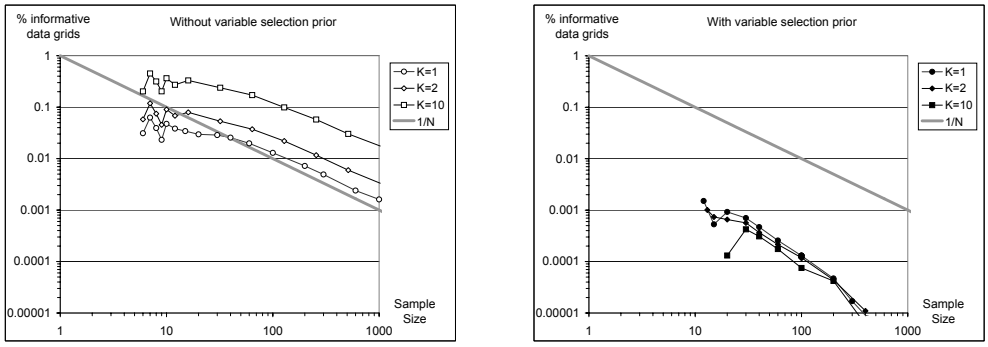


Figure 5: Percentage of informative data grids having more than one cell, for 1, 2 and 10 numerical input variable independent from the target variable, with and without prior for variable selection.

The results demonstrate the robustness of the approach: very few data grids are wrongly detected as informative, and the percentage of false detection rapidly decreases with the sample size. However, without prior for variable selection, the percentage of false detection grows almost linearly with the number of input variables. This makes sense since a set of K variables can be detected as an informative multivariate data grid if at most one of the K variables is detected as an informative univariate discretization.

When the prior for variable selection is accounted for, the percentage of wrongly informative models falls down by two orders of magnitude, and the rates of false detection are rapidly consistent for the different numbers of input variables. The selection prior significantly strengthens the robustness of the method and makes it almost independent from the number of variables in the representation space.

6.2 The Multivariate XOR Pattern

The purpose of the XOR pattern experiment is to evaluate the capacity of the method to detect complex correlations between the input variables. The pattern consists of an output variable which depends upon the input variables according to a XOR schema, as illustrated

in Figure 6. All the input variables are uniformly distributed on the [0, 1] numerical domain. For each input variable, we compute a Boolean index according to whether the input value is below or beyond 0.5, and the output value is assigned a Boolean value related to the parity of the sum of the input indexes, which corresponds to a XOR pattern.

We first present a theoretical threshold of detection for the XOR pattern, then illustrate the behavior of the algorithms for this problem, and finally report experimental results on this complex pattern detection problem.

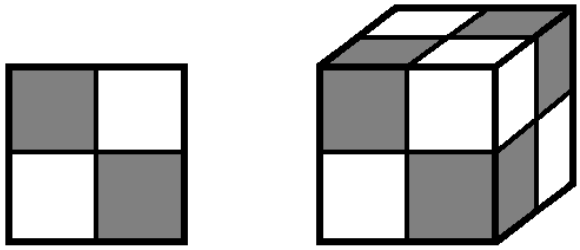


Figure 6: Multivariate XOR pattern in dimension 2 and 3.

6.2.1 Theoretical Detection Threshold

Let us consider K input variables, K_s of which represent a multivariate XOR pattern related to the output variable. The expected multivariate discretization for this pattern consists of a data grid model M_G with K_s selected input variables, each of which is discretized into two intervals. The data grid model M_G contains $G = 2^{K_s}$ cells. In order to obtain a closed formula, let us assume that these cells contain the same number $N_G = N/G$ of instances. Let us evaluate the null model $M_\emptyset$, reduced to one single cell, and the expected XOR data grid model M_G. According to Formula 3, we get

$$c(M_\emptyset) = \log(K+1) + \log(N+1) + \log \frac{N!}{N_1!N_2!}, \tag{8}$$

$$\begin{aligned} c(M_G) = {} & \log(K+1) + \log \binom{K+K_s-1}{K_s-1} + \\ & K_s \log N + K_s \log(N+1) + G \log(N_G+1). \end{aligned} \tag{9}$$

For $N_G = 1$, the null model is always preferred: one instance per cell is not enough to detect the multivariate pattern.

For small values of K and for $K_s = K$, we perform a numerical simulation to compute the minimum cell frequency N_G such that the cost $c(M_G)$ of the multivariate XOR model is lower than that of the null model. The results, reported in Figure 7, indicate that at least ten instances per cell, representing overall forty instances, are necessary to detect the bi-dimensional XOR pattern. This cell frequency threshold decreases with the number of input variables, and falls down to two instances per cell when the number of input variables is beyond ten. Let us notice that in spite of a very small cell frequency threshold, the whole dataset frequency threshold still grows exponentially with the the number of variables.

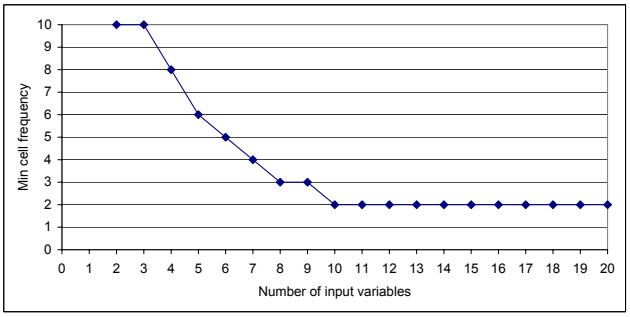


Figure 7: Min cell frequency necessary to detect a multivariate XOR pattern using a data grid model. For example, for a 5-dimensional XOR, 6 instances per cell, or $192 = 2^5 * 6$ instances in the sample, allow to detect the pattern using a data grid of 32 cells.

We now extend these simulation results in the asymptotic case, assuming that each cell contains exactly $N_G = N/G$ instances. From Equations 8 and 9, we get

$$c(M_G) = c(M_\emptyset) + \log \binom{K + K_s - 1}{K_s - 1} + (2K_s - 1) \log N - N(\log 2 - \frac{1}{N_G} \log(N_G + 1)) + O(\log N).$$

This implies that for $N_G \geq 2$, the multivariate XOR model has an asymptotically lower cost than that of the null model, even when the total number K of input variables exceeds the number K_s of informative input variables.

Overall, about 2^{K+1} instances are sufficient to detect K-dimensional informative patterns, which correspond to 2 instances per cell. Since this is close from the theoretical detection threshold, this means that for a dataset consisting of N instances, it might be difficult to detect patterns exploiting more than $\log_2 N$ informative dimensions.

6.2.2 Empirical Analysis of the Algorithms

Let us first analyze the behavior of the greedy bottom-up heuristic presented in Section 5.1. This heuristic starts with the maximum data grid, which contains $O(N^K)$ cells for at most N non-empty cells. During the whole merge process, $O(KN)$ merges are necessary to transform the maximum data grid with N^K elementary cells into the null data grid with one single cell. During the first $(K-1)N$ merges, most of the merges between adjacent intervals concern merges between two empty adjacent cells or merges between one non-empty cell and one empty cell. When the data grid is too sparse, most interval merges do not involve "collisions" between non-empty cells. According to Formula 3, the only cell merges that have an impact on the likelihood of the data grid model are the "colliding" cell merges. This means that at the beginning of the greedy bottom-up heuristic, the earlier merges are guided only by the prior distribution of the models, not by their likelihood. These "blind" merges are thus likely to destroy potentially interesting patterns.

To illustrate this behavior, we perform an experiment with the basic greedy heuristic described in Algorithm 1 on a bi-dimensional XOR pattern. According to Formulas 8 and 9,

about 40 instances are sufficient to detect the pattern. However, the greedy bottom-heuristic fails to discover the XOR pattern when the number of instance is below 1000.

The algorithms presented in Section 5 enhance the basic greedy heuristic using a random initialization, a pre-processing step, the greedy bottom-up merge heuristic and a post-processing step, as illustrated in Figure 8. The random initialization produces a dense enough data grid with at least one instance per cell on average. This is achieved by selecting at most $K_s = \log_2 N$ input variables and N^{1/K_s} parts per variable. The purpose of the pre-processing step is to "purify" the initial solution, since a random solution is likely to be blind to informative patterns. This pre-processing consists in moving the boundaries of the initial data grid, in order to get "cleaner" initial cells, as illustrated in Figure 8. The greedy merge heuristic is then applied on this dense cleaned data grid, and the merges are guided by the data, since the data grid is sufficiently dense. The role of the post-processing step is to improve the final solution, by exploring a local neighborhood of the solution consisting of interval splits, merges and moves of interval boundaries.

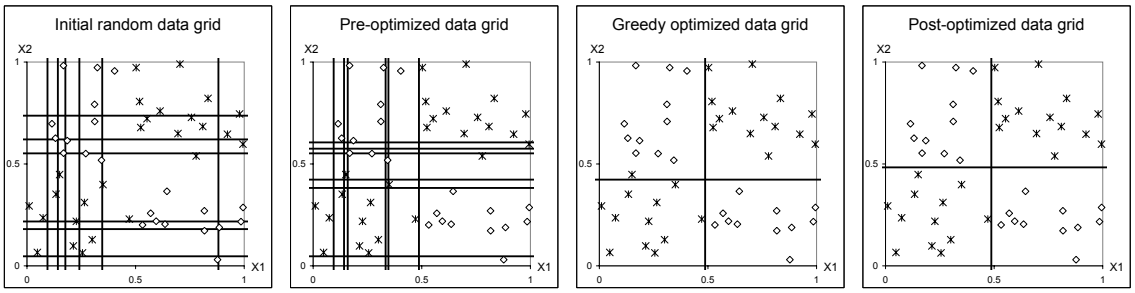


Figure 8: Main steps in the optimization algorithm: a random initial solution is first generated to start with a dense enough data grid, then cleaned during a pre-processing step, optimized with the greedy bottom-up merge heuristic and improved during the post-processing step.

All these steps are repeated several times in the VNS meta-heuristic described in Section 5.4, which generates several random initial data grids of varying size. The only optimization parameter relates to the number of iterations in the meta-heuristic, which controls the intensity of the search.

A quantitative evaluation of the effect of each part of the algorithm is reported in (Boullé, 2008a) in the case of bivariate XOR patterns. The greedy heuristic alone is likely to be misled by the sparsity of the data and needs a very large number of instances to discover the true patterns. The meta-heuristic, which starts multiple times from dense enough random initial solutions, manages to approximate the true patterns with about 100 times fewer instances than the greedy heuristic. However, the random initialization process is not likely to produce candidate data grids with accurate boundaries. This is corrected by the pre-optimization and post-optimization heuristics.

All of the algorithmic components are useful in achieving an effective search of the space of data grids and efficiently detecting informative patterns. Using these algorithms, the empirical threshold for the detection of simple XOR patterns reaches the theoretical

threshold, even with one single iteration in the meta-heuristic. For example, bi-dimensional randomly generated patterns require only 40 instances to be detected, and 5-dimensional XOR pattern only 200 instances. In the following sections, we study the detection of more complex XOR patterns, which require more intensive search.

6.2.3 Detection of a Complex Patterns with Few Instances

In this experiment, we study the detection of a 10-dimensional XOR pattern in a 10-dimensional input space. The experiment is performed on a set of sample sizes ranging from 1000 to 10000 instances, and repeated 100 times for each sample size. We evaluate the empirical detection threshold for the VNS meta-heuristic, with optimization parameters T, where VNS(T) performs around 2^T iterations of the algorithm from a variety of random initial data grids. Figure 9 reports the average computation time for each sample size and for parameters of the VNS meta-heuristic ranging from $T = 1$ to $T = 12$. We also report the threshold related to the sample size and computation time, among which the XOR pattern is detected in 50% of the cases.

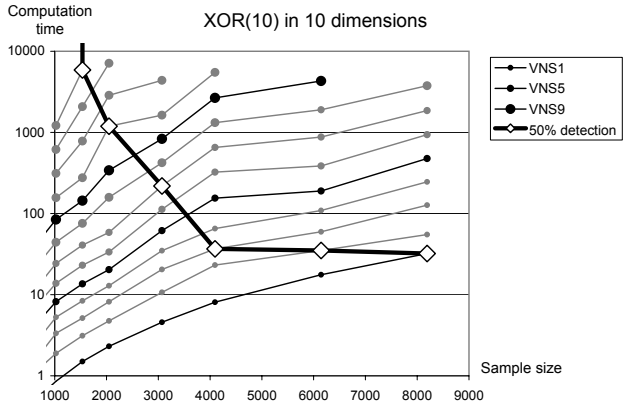


Figure 9: Study of the algorithm for the detection of the 10-dimensional XOR pattern.

The results show that the empirical detection threshold is close to the theoretical threshold: the pattern is never detected with 1000 instances but frequently detected with only 1500 instances, which is less than 2 instances per cell of the 10-dimensional XOR pattern. However, when the instance number is close to the theoretical threshold, the problem of finding the correct 10 variable splits among N^{10} possible XOR patterns and $(2^N)^{10}$ potential multivariate discretizations is very hard. In this case, detecting the pattern requires much more time than when the instance number is large enough or when the pattern is simpler. For example, detecting the pattern with only 1500 instances requires about one hundred times more computation time than with 4000 instances

6.2.4 Finding a Needle in a haystack

In this experiment, we study the detection of a 5-dimensional XOR pattern in a 10-dimensional input space. We use the same protocol as in the previous case, and report the results in Figure 10.

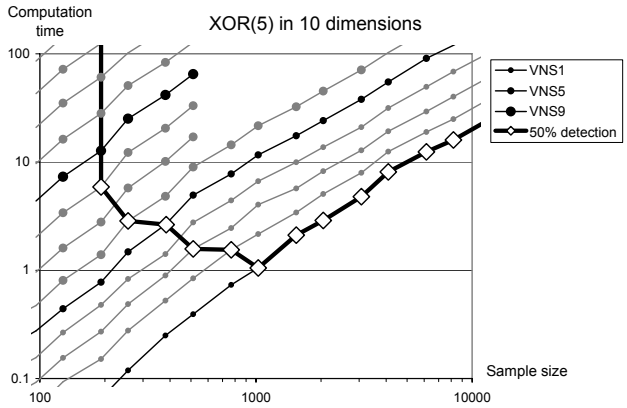


Figure 10: Study of the algorithm for the detection of the 5-dimensional XOR pattern, hidden in a 10-dimensional input space.

The results show that about 200 instances are sufficient to detect this pattern, which is consistent with the theoretical threshold. However, whereas the 5-dimensional XOR pattern is easily detected even within one or two iterations in the VNS meta-heuristic, the search in that 10-dimensional input space requires much more intensive search.

Apart from of the problem of finding the correct XOR boundaries, which is a difficult task, the problem of variable selection complicates the detection of the pattern. The optimization algorithm is restricted to the exploration of dense data grids, which consist of $K_x \leq \max(\log_2 N, K)$ dimensions. Finding the XOR pattern requires us to select a subset of K_x input variables among K, which is a superset of the K_s informative variables. The probability that such a subset contains the informative variable is $\binom{K_x}{K_s}/\binom{K}{K_s}$. For example, for the detection of a 5-dimensional XOR ($K_s = 5$) with 256 instances ($K_x = \log_2 256 = 8$), the probability of finding a potentially good subset is 100% for $K = 5$, 22% for $K = 10$, 0.36% for $K = 20$ and 0.04% for $K = 30$.

We performed an experiment to detect the 5-dimensional XOR in 20 dimensions with samples of size 256. The result confirms that there are enough instances for a reliable detection of the pattern, but the computational time necessary to detect the pattern in 50% of the cases amounts to about one hundred times that in 10 dimensions. This result, consistent with the ratio 22/0.36, illustrates the problem of finding a needle in a haystack.

Overall, the evaluation criterion given in Formula 3 is able to reliably differentiate informative patterns from noise with very few instances. The detection of complex patterns is a combinatorial problem, that is hard to solve when the number of instances is close to the detection threshold or when the informative patterns are hidden in large dimensional

input spaces. Our optimization algorithm succeeds in reliably and efficiently detecting information, with performance close to the theoretical detection threshold.

7. Evaluation on the Agnostic Learning vs. Prior Knowledge Challenge

In this section, we first summarize the evaluation protocol of the challenge, then describe how classifiers are built from data grid models, and finally report the results from a performance and understandability point of view.

7.1 The Agnostic Learning vs. Prior Knowledge Challenge

The purpose of the challenge (Guyon, 2007; Guyon et al., 2007) is to assess the real added value of prior domain knowledge in supervised learning tasks. Five datasets coming from different domains are selected to evaluate the performance of agnostic classifiers vs. prior knowledge classifiers. These datasets come into two formats, as shown in Table 1. In the agnostic format, all the input variables are numerical. In the prior knowledge format, the input variables are both categorical and numerical for three datasets and have a special format in the two other datasets: chemical structure or text. The evaluation criterion is the test balanced error rate (BER).

Name	Domain	Num. ex. train/valid/test	Prior features	Agnostic features
Ada	Marketing	4147/415/41471	14	48
Gina	Handwritting reco.	3153/315/31532	784	970
Hiva	Drug discovery	3845/384/38449	Chem. struct.	1617
Nova	Text classification	1754/175/17537	Text	16969
Sylva	Ecology	13086/1309/130857	108	216

Table 1: Challenge datasets with their prior and agnostic format.

7.2 Building Classifiers from Data Grid Models

In this section, we describe three ways of building classifiers from data grid models.

7.2.1 Data Grid

In this evaluation of data grid models, we consider one individual supervised data grid, the MAP one. We build a classifier from a data grid model by first retrieving the cell related to a test instance, and predicting the output conditional probabilities of the retrieved cell. For empty cells, the conditional probability used for the prediction is that of the entire grid.

Data grid models can be viewed as a feature selection method, since the input variables whose partition reduces to a single part can be ignored. The purpose of this experiment is to focus on understandable models and evaluate the balance between the number of selected variables and the predictive performance.

7.2.2 Data Grid Ensemble

In this evaluation, we focus on the predictive performance rather than on understandability, by means of averaging the prediction of a large number of classifiers. This principle was successfully exploited in Bagging (Breiman, 1996) using multiple classifiers trained from re-sampled datasets. This was generalized in Random Forests (Breiman, 2001), where the subsets of variables are randomized as well. In these approaches, the averaged classifier uses a voting rule to classify new instances. Unlike this approach, where each classifier has the same weight, the Bayesian Model Averaging (BMA) approach (Hoeting et al., 1999) weights the classifiers according to their posterior probability. The BMA approach has stronger theoretical foundations, but it requires both to be able to evaluate the posterior probability of the classifiers and to sample their posterior distribution.

In the case of data grid models, the posterior probability of each model is given by an analytic criterion. Concerning the problem of sampling the posterior distribution of data grid models, we have to strike a balance between the quality of the sampling and the computation time. We adopt a pragmatic choice by just collecting all the data grids evaluated during training, using the optimization algorithms introduced in Section 5. We keep all the local optima encountered in the VNS meta-heuristic and eliminate the duplicates.

An inspection of the data grids collected reveals that their posterior distribution is so sharply peaked that averaging them according to the BMA approach almost reduces to the MAP model. In this situation, averaging is useless. The same problem has been noticed by Boullé (2007b) in the case of averaging Selective Naive Bayes models. To find a trade-off between equal weights as in bagging and extremely unbalanced weights as in the BMA approach, we exploit a logarithmic smoothing of the posterior distribution called compression-based model averaging (CMA), like that introduced in (Boullé, 2007b).

To summarize, we collect the data grid models encountered during the data grid optimization algorithm and weight them according to a logarithmic smoothing of their posterior probability to build a data grid ensemble classifier.

7.2.3 Coclustering

The coclustering method introduced in Section 4 applies on binary sparse datasets. Whereas the supervised data grids are limited in practice to a small number of selected input variables (see Section 6), the coclustering data grids are able to account for all the input variables.

The coclustering data grid is trained on all the available input data (train, validation and test), then the available labeled instances are used to predict the output distribution in each cluster of instances. In the case where a test instance belongs to a cluster with no labeled instance, we iteratively merge this unlabeled cluster so as to keep the coclustering evaluation criterion as low as possible, until at least one labeled cluster is encountered.

7.3 Evaluation of Supervised Data Grids

We first analyze the classification performance of supervised data grids, then focus on their benefit for understandability.

7.3.1 Classification Results

To evaluate the supervised data grid models, we use all the datasets in their agnostic format and only three of them in their prior format (the ones that come in a tabular format). In the case of the Sylva dataset in its prior format, we replace each subset (per record) of 40 binary SoilType variables by one single categorical variable with 40 values. The resulting dataset has only 30 variables instead of 108.

The data grid techniques are able to predict the output conditional probabilities for each test instance. When the evaluation criterion is the classification accuracy, predicting the class with the highest conditional probability is optimal. This is not the case for the BER criterion used in the challenge. We post-process each trained classifier by optimizing the probability threshold in order to maximize the BER. This optimization is performed directly on the train dataset.

Our four submissions related to supervised data grid models are named *Data Grid (MAP)* and *Data Grid (CMA)* in the prior or agnostic track and dated from February 27, 2007 for the challenge March 1st, 2007 milestone. The classifiers are trained with the any time optimization algorithm described in Section 5 using VNS(12) parameter. About 4000 data grids are evaluated, needing around one hour optimization time per dataset. Table 2 and Table 3 report our results in the agnostic and prior track.

Dataset	Winner	Best BER	Data Grid (CMA)	Data Grid (MAP)
Ada	Roman Lutz	0.166	0.1761	0.2068
Gina	Roman Lutz	0.0339	0.1436	0.1719
Hiva	Vojtech Franc	0.2827	0.3242	0.3661
Nova	Mehreen Saeed	0.0456	0.1229	0.2397
Sylva	Roman Lutz	0.0062	0.0158	0.0211

Table 2: Best challenge results versus our supervised data grid methods results for the datasets in the agnostic track.

Dataset	Winner	Best BER	Data Grid (CMA)	Data Grid (MAP)
Ada	Marc Boullé	0.1756	0.1756	0.2058
Gina	Vladimir Nikulin	0.0226	0.1254	0.1721
Sylva	Roman Lutz	0.0043	0.0228	0.0099

Table 3: Best challenge results versus our supervised data grid methods results for the Gina, Hiva and Nova datasets in the prior track.

The data grid classifiers obtain good results on the Ada and Sylva datasets, especially on the prior track, with a winning submission for the Ada dataset. The other datasets contain very large numbers of variables, which explains the poor performance of the data grid models. Since individual data grid models are essentially restricted to about $\log_2 N$ selected variable, they cannot exploit much of the information contained in the representation space. This is analyzed in Section 7.3.2.

The data grid ensemble classifiers confirm the benefits of compression-based model averaging. They obtain a very significant improvement of the BER criterion compared to the individual data grid classifiers. This focus on predictive performance is realized at the expense of understandability, since each trained data grid ensemble averages several hundreds of elementary data grid models.

However, even data grid ensembles fail to achieve competitive performance for datasets with large numbers of variables. A close inspection reveals that although about 4000 data grids are evaluated for each dataset, only a few hundreds (≈ 500) of different solutions are retrieved. Removing the duplicates significantly improves the performances, but there is still too much redundancy between data grids to produce an efficient ensemble classifier. Furthermore, a few hundred redundant classifiers, each with only $\approx \log_2 N$ variables, is not enough to exploit all the variables (think of Nova with 17000 variables for example). In future work, we plan to improve our meta-heuristic in order to better explore the search space and to collect a set of data grid solutions with better diversity.

7.3.2 Understandability

Let us now focus on understandability and inspect the number of selected variables in each trained data grid model. In the agnostic track, the MAP data grid exploits only 5 variables for Ada, 5 for Gina, 4 for Hiva, 8 for Nova and 8 for Sylva. In the prior track, the MAP data grid exploits 6 variables for Ada, 7 for Gina and 4 for Sylva. These numbers of variables are remarkably small w.r.t the BER performance of the models.

ID	relationship	occupation	education number	age	capital gain	capital loss	frequency	% class 1
1	Married	Low	≤ 12	> 27	≤ 4668	≤ 1805	736	22.1%
2	Not married	Low	≤ 12	> 27	≤ 4668	≤ 1805	577	3.1%
3	Not married	High	≤ 12	> 27	≤ 4668	≤ 1805	531	5.8%
4	Married	High	≤ 12	> 27	≤ 4668	≤ 1805	489	41.3%
5	Married	High	> 12	> 27	≤ 4668	≤ 1805	445	68.5%
6	Not married	Low	≤ 12	≤ 27	≤ 4668	≤ 1805	425	0.2%
7	Not married	High	≤ 12	≤ 27	≤ 4668	≤ 1805	316	0.6%
8	Not married	High	> 12	> 27	≤ 4668	≤ 1805	268	20.5%
9	Not married	High	> 12	≤ 27	≤ 4668	≤ 1805	112	0.9%
10	Married	Low	≤ 12	≤ 27	≤ 4668	≤ 1805	96	5.2%
11	Married	High	> 12	> 27	> 5095	≤ 1805	93	100.0%
12	Married	Low	> 12	> 27	≤ 4668	≤ 1805	50	24.0%

Table 4: Most frequent cells in the best individual data grid model for the Ada dataset in the prior track.

In Table 4, we summarize the MAP data grid trained using the 4562 train+valid instances of the Ada dataset in the prior track. This data grid selects six variables among 14 and obtains a 0.2068 test BER. The selected variables are relationship, occupation, education number, age, capital gain and capital loss, which are partitioned into 2, 2, 2, 2, 3 and 3 groups or intervals. The relationship variable is grouped into Married = {Husband, Wife}

vs. Not Married = {Not-in-family, Own-child, Unmarried, Other-relative}, and the occupation into Low = {Craft-repair, Other-service, Machine-op-inspct, Transport-moving, Handlers-cleaners, Farming-fishing, Priv-house-serv} vs. High = {Prof-specialty, Exec-managerial, Sales, Adm-clerical, Tech-support, Protective-serv, Armed-Forces}. Overall, the data grid contains $144 = 2 * 2 * 2 * 2 * 3 * 3$ cells, but 57 of them are non empty and the twelve most frequent cells reported in Table 4 contains 90% of the instances.

Each cell of the data grid can directly be interpreted as a decision rule. For example, the most frequent cell is described by Rule 1, with a support of 736 instances.

Rule 1:	IF	relationship $\in$ Married = {Husband, Wife}
		occupation $\in$ Low = {Craft-repair, Other-service, Machine-op-inspct,...}
		education number ≤ 12
		age > 27
		capital gain ≤ 4668
		capital loss ≤ 1805
	THEN	P(class=1) = 22.1%

The whole data grid forms a set of rules (Mitchell, 1997) which corresponds to a partition (not a coverage) of the training set. Since all rules exploit the same variables with the same univariate partitions, interpretation is much easier. For example, rule 5 (ID cell=5 in Table 4) has a large support of 445 instances with 68.5% of class 1. Rule 4 with 41.3% of class 1 only differs in the education number variable (≤ 12 vs. > 12), and rule 8 with 20.5% of class 1 in the relationship variable (Not married vs. Married).

7.4 Evaluation of Coclustering Data Grids

We first inspect the dimension of the data grids resulting from the coclustering method introduced in Section 4.2, then analyze its performance results and finally present its interest for understandability in the case of the Nova text corpus.

To evaluate the coclustering data grid models, we consider three datasets (Gina, Hiva and Nova) as sparse binary datasets. For the Gina dataset, the binary representation is obtained from the prior format by replacing each non zero value by 1. The Hiva dataset is used directly in its agnostic binary format. For the Nova dataset, we exploit the prior format in order to get insights on the understandability of the models. We preprocess the Nova text format by keeping all words of at least three characters, converting them to lowercase, truncating them to at most seven characters, and keeping the most frequent resulting words (≥ 8) so as to get a manageable bag-of-words representation (we keep the most frequent 19616 words using this frequency threshold). This preprocessing is very similar to that for the agnostic track, except that we do not exclude the 2000 most frequent words.

7.4.1 Dimensionality Reduction

The coclustering method exploits all the available unlabeled data to represent the initial binary matrix (instances $\times$ variables) which is potentially sparse into a denser matrix with clusters of instances related to clusters of variables. It is worth noting that the space of coclustering models is very large. For example, in the case of the Nova dataset, the number of ways of partitioning both the text and the words, based on the Bell number, is greater

than 10^{120000}. To obtain the best possible coclustering according to our MAP approach, we allocated several days of computation time to our anytime optimization heuristic.

Dataset	Initial representation				Coclustering representation			
	Inst.	Var.	Size	Sparseness	Inst. cl.	Var. cl.	Size	Sparseness
Gina	35000	784	$2.74\ 10^7$	19.2%	480	125	$6.00\ 10^4$	79.1%
Hiva	42673	1617	$6.90\ 10^7$	9.1%	1230	210	$2.58\ 10^5$	52.2%
Nova	17537	19616	$3.44\ 10^8$	0.6%	207	1058	$2.19\ 10^5$	84.3%

Table 5: Properties of the (instances × variables) matrix for the Gina, Hiva and Nova datasets, in their initial and coclustering representation.

In Table 5, we recall the properties of each dataset in its initial representation and present its pre-processed representation after the coclustering. The datasets are initially represented using very larges matrices, with up to hundreds of millions of cells. Their sparseness vary from less than 1% to about 20%. The number of non-zero elements (one variable activated for one instance) is about five million for Gina, six million for Hiva and two million for Nova. Once the coclustering is performed, we get dense representations with numbers of cells reduced by a factor of one hundred to one thousand.

7.4.2 Classification Results

In order to evaluate the quality of the representation, we train classifiers using the train and validation labeled instances to learn the distribution of the labels in each cluster of instances.

Dataset	Prior track		Agnostic track		Coclustering
	Winner	Best BER	Winner	Best BER	BER
Gina	Vladimir Nikulin	0.0226	Roman Lutz	0.0339	0.0516
Hiva	Chloé Azencott	0.2693	Vojtech Franc	0.2827	0.3127
Nova	Jorge Sueiras	0.0659	Mehreen Saeed	0.0456	0.0370

Table 6: Best challenge results vs. our coclustering method results for the Gina, Hiva and Nova datasets.

We recall in Table 6 the BER results of the challenge winner in the agnostic and prior track (see Guyon et al., 2007), and present our results obtained with the semi-supervised coclustering method (submission named "Data Grid(Coclustering)", dated 2007-02-27 for Gina and Hiva and 2007-09-19 for Nova). It is noteworthy that for datasets with large numbers of variables, the coclustering technique obtains far better performance than the supervised technique, with BER results about three times better on the Gina and Nova datasets. This comes from the ability of the coclustering to exploit all the variables, whereas each supervised data grid is restricted to a subset of about $\log_2 N$ variables.

Overall, the supervised coclustering method obtains good predictive performance, competitive with that of most of the challenge participants. On the Gina dataset, which is not

very sparse, the BER is over twice as high as that of the leader. In the case of the Nova dataset, which is very sparse, the predictive performance significantly outperforms that of the winners and reaches the best reference result of the organizers, which is remarkable since our clusters were learned without using any class label.

7.4.3 Understandability

The assumption that the "natural" patterns identified by the coclustering are correlated with the classes looks true in the challenge datasets. Since we obtain many more patterns than classes, it is interesting to provide an interpretation of our coclusters.

The Gina dataset comes from the MNIST dataset (LeCun and Cortes, 1998). The task, which is handwritten digit recognition, consists of predicting the value of a digit from an image representation of $28*28$ pixels. The coclustering method identifies about one hundred clusters of pixels (regions) and five hundred clusters of images ("natural" shapes), each of them distributed similarly on the regions. Although the classification BER is only 0.0516 (about twice as high as that of the winner), it is interesting to notice that each digit (among the ten possible output digits) comes under a large variety of shapes. This is discovered without any domain knowledge and could be the foundation for adequate preprocessing.

In the case of the Hiva, further investigation with a domain specialist would be necessary to understand the meaning of the clusters of instances and variables.

The Nova dataset comes from the 20-Newsgroup dataset (Mitchell, 1999). The original task is to classify the texts into 20 classes (atheism, graphics, forsale, autos, motorcycles, baseball, hockey, crypt, electronics, med, space, religion.christian, politics.guns, politics.mideast, politics.misc, religion.misc). In the challenge, the classification task was a binary one, with two groups of classes (politics or religion vs. others). The coclustering method identifies about one thousand clusters of words (vocabulary themes) and two hundred clusters of texts ("natural" topics), each of them distributed similarly on the themes.

The distribution of the 17537 texts in the 207 clusters of texts (topics) is reasonably balanced. On the other hand, the repartition of the 19616 words in the 1058 clusters of words (themes) is not balanced at all. About 150 themes are singletons, like for example *the*, *and*, *for*, *that*, *this*, *have*, *you*. These are frequent words with low semantic, and even slightly different distribution of the topics on these singleton themes are significant and might be helpful for classification. For example, observing one of the singleton themes *say*, *why* or *who* approximately doubles the conditional probability of being in the challenge positive class (politics or religion).

A correlation study between the themes and the 20 original labels available on the train dataset reveals that the most informative themes are:

- *hockey, playoff, nhl, penguin, devils, pens, leafs, bruins, islande, goalie, mario, puck,...*
- *team, season, league, fans, teams, rangers, detroit, montrea, wins, scored, coach,...*
- *clipper, encrypt, nsa, escrow, pgp, crypto, wiretap, privacy, cryptog, denning,...*
- *dod, bike, motorcy, ride, riding, bikes, ama, rider, helmet, yamaha, harley, moto,...*
- *basebal, sox, jays, giants, mets, phillie, indians, cubs, yankees, stadium, cardina,...*
- *bible, scriptu, teachin, biblica, passage, theolog, prophet, spiritu, testame, revelat,...*
- *christi, beliefs, loving, rejecti, obedien, desires, buddhis, deity, strive, healed,...*
- *windows, dos, apps, exe, novell, ini, borland, ver, lan, desqvie, tsr, workgro, sdk,...*
- *pitcher, braves, pitch, pitchin, hitter, inning, pitched, pitches, innings, catcher,...*
- *car, cars, engine, auto, automob, mileage, autos, cactus, pickup, alarm, sunroof,...*

About one third of the theme are detected as informative with respect to the original labels. The partition of the words is very fine grained, so that many themes are hard to interpret, whereas other ones clearly capture semantics, such as:

- *book, books, learnin, deals, booksto, encyclo, titled, songs, helper*
- *cause, caused, causes, occur, occurs, causing, persist, excessi, occurin*
- *importa, extreme, careful, essenti, somewha, adequat*
- *morning, yesterd, sunday, friday, tuesday, saturda, monday, wednesd, thursda,...*
- *receive, sent, placed, returne, receivi, sends, resume*

Overall, our coclustering preprocessing method is able to produce a precise and reliable summary of the corpus of texts, which is demonstrated by the very good classification performance reported in Table 6.

8. Conclusion

The supervised data grid models introduced in this paper are based on a partitioning model of each input variable, into intervals for numerical variables and into groups of values for categorical variables. The cross-product of the univariate partitions, called a data grid, allows the quantification of the conditional information relative to the output variable. We have detailed this technique in the multivariate case, with a Bayesian approach for model selection and sophisticated combinatorial algorithms to efficiently search the model space.

We have also presented the principles of the extension of data grid models to unsupervised learning to evaluate the joint probability distribution of the variables. We have detailed the case of two categorical variables and applied it to the problem of coclustering the instances and variables of a sparse binary dataset.

In extensive artificial experiments, we have shown that our technique is able to reliably detect complex patterns. Our experiments quantify the limits of the approach, with data grid models limited to about $\log_2$ variables, and provides insights into the relation between the complexity of the patterns and the required computation time necessary to detect them.

We have introduced three ways of building classifiers from data grids and experimented them on the Agnostic Learning vs. Prior Knowledge challenge. This preliminary evaluation looks promising since our method was first on two of the datasets, one within the challenge deadline and the other one using a later submission. The analysis of the results demonstrates that the data grid models are of considerable interest for data understandability and data preparation.

Overall, the supervised data grids obtain good performance on datasets with small numbers of variables, while the coclustering data grids perform well on sparse binary datasets with very large numbers of variables. In future research, we plan to investigate how to better exploit the potential of these models to build more powerful classifiers. Apart from improving the optimization algorithms and building ensemble classifiers based on a wider diversity of data grid models, we intend to further explore the problem of conditional or joint density estimation.

Whereas the naive Bayes strategy (Langley et al., 1992) factorizes the multivariate density estimation on univariate estimations, our strategy with the data grid models directly estimates the multivariate joint density, which encounters a limit in the number of variables considered. Between these two opposite strategies, other approaches have been considered,

based on a relaxation of the naive Bayes assumption. This is the case for example in semi-naive Bayesian classifiers (Kononenko, 1991) or in Bayesian network classifiers (Friedman et al., 1997). In this context, we expect data grid models to be promising building blocks of future better multivariate density estimators.

References

M. Abramowitz and I. Stegun. *Handbook of mathematical functions*. Dover Publications Inc., New York, 1970.

C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1996. http://www.ics.uci.edu/mlearn/MLRepository.html.

H. Bock. Simultaneous clustering of objects and variables. In E. Diday, editor, *Analyse des Données et Informatique*, pages 187–203. INRIA, 1979.

M. Boullé. Optimal bivariate evaluation for supervised learning using data grid models. *Advances in Data Analysis and Classification*, 2007a. submitted.

M. Boullé. A Bayes optimal approach for partitioning the values of categorical attributes. *Journal of Machine Learning Research*, 6:1431–1452, 2005.

M. Boullé. Compression-based averaging of selective naive Bayes classifiers. *Journal of Machine Learning Research*, 8:1659–1685, 2007b.

M. Boullé. MODL: a Bayes optimal discretization method for continuous attributes. *Machine Learning*, 65(1):131–165, 2006.

M. Boullé. Bivariate data grid models for supervised learning. Technical Report NSM/R&D/TECH/EASY/TSI/4/MB, France Telecom R&D, 2008a. http://perso.rd.francetelecom.fr/boulle/publications/BoulleNTTSI4MB08.pdf.

M. Boullé. Multivariate data grid models for supervised and unsupervised learning. Technical Report NSM/R&D/TECH/EASY/TSI/5/MB, France Telecom R&D, 2008b. http://perso.rd.francetelecom.fr/boulle/publications/BoulleNTTSI5MB08.pdf.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees*. California: Wadsworth International, 1984.

O. Chapelle, B. Schölkopf, and A. Zien. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

W.G. Cochran. Some methods for strengthening the common chi-squared tests. *Biometrics*, 10(4):417–451, 1954.

J. Connor-Linton. Chi square tutorial, 2003. http://www.georgetown.edu/faculty/ballc/-webtools/web_chi_tut.html.

N. Friedman, D. Geiger, and M. Goldsmidt. Bayesian network classifiers. *Machine Learning*, 29:131–163, 1997.

P.D. Grünwald, I.J. Myung, and M.A. Pitt. *Advances in minimum description length : theory and applications*. MIT Press, 2005.

I. Guyon. Agnostic learning vs. prior knowledge challenge, 2007. http://clopinet.com/-isabelle/Projects/agnostic/.

I. Guyon, A.R. Saffari, G. Dror, and G. Cawley. Agnostic learning vs. prior knowledge challenge. In *International Joint Conference on Neural Networks*, pages 829–834, 2007.

M.H. Hansen and B. Yu. Model selection and the principle of minimum description length. *J. American Statistical Association*, 96:746–774, 2001.

P. Hansen and N. Mladenovic. Variable neighborhood search: principles and applications. *European Journal of Operational Research*, 130:449–467, 2001.

J.A. Hartigan. Direct clustering of a data matrix. *Journal of the American Statistical Association*, 67(337):123–129, 1972.

J.A. Hoeting, D. Madigan, A.E. Raftery, and C.T. Volinsky. Bayesian model averaging: A tutorial. *Statistical Science*, 14(4):382–417, 1999.

G.V. Kass. An exploratory technique for investigating large quantities of categorical data. *Applied Statistics*, 29(2):119–127, 1980.

I. Kononenko. Semi-naive Bayesian classifier. In Y. Kodrato, editor, *Sixth European Working Session on Learning (EWSL91)*, volume 482 of *LNAI*, pages 206–219. Springer, 1991.

P. Langley, W. Iba, and K. Thompson. An analysis of Bayesian classifiers. In *10th national conference on Artificial Intelligence*, pages 223–228. AAAI Press, 1992.

Y. LeCun and C. Cortes. The MNIST database of handwritten digits, 1998. http://yann.lecun.com/exdb/mnist/.

M. Li and P.M.B. Vitanyi. *An Introduction to Kolmogorov Complexity and Its Applications*. Springer-Verlag, Berlin, 1997.

T.M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

T.M. Mitchell. The 20 newsgroup dataset, 1999. http://kdd.ics.uci.edu/databases/-20newsgroups/20newsgroups.html.

J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

C.E. Shannon. A mathematical theory of communication. Technical Report 27, Bell systems technical journal, 1948.

D.A. Zighed and R. Rakotomalala. *Graphes d'induction*. Hermes, France, 2000.