SuSE : Subspace Selection embedded in an EM algorithm

L. Candillier[1,2], I. Tellier[1], F. Torre[1], O. Bousquet[2]

[1] GRAppA, Université Charles de Gaulle, Lille 3
candillier@grappa.univ-lille3.fr
[2] Pertinence, 32 rue des Jeûneurs, 75002 Paris
olivier.bousquet@pertinence.com

Résumé :
Subspace clustering is an extension of traditional *clustering* that seeks to find *clusters* embedded in different subspaces within a dataset. This is a particularly important challenge with high dimensional data where the *curse of dimensionality* occurs. It also has the benefit of providing smaller descriptions of the clusters found.

In this field, we show that using probabilistic models provides many advantages over other existing methods. In particular, we show that the difficult problem of the parameter settings of subspace clustering algorithms can be seen as a *model selection* problem in the framework of probabilistic models. It thus allows us to design a method that does not require any input parameter from the user.

We also point out the interest in allowing the clusters to overlap. And finally, we show that it is well suited for detecting the noise that may exist in the data, and that this helps to provide a more understandable representation of the clusters found.

1 Introduction

Clustering is a powerful exploration tool capable of uncovering previously unknown patterns in data (Berkhin, 2002). *Subspace clustering* is an extension of traditional clustering that is based on the observation that different *clusters* (groups of data points) may exist in different subspaces within a dataset (see figure 1 as an example). Subspace clustering is thus more general than classical *feature selection* for clustering because each subspace may be local to each cluster, instead of global to all of them.

This is a particularly important challenge with high dimensional data where the *curse of dimensionality* can degrad the quality of the results. Besides, it helps to get smaller descriptions of the clusters found since clusters are defined on fewer dimensions than the original number of dimensions.

In this paper, we point out the interest in using *probabilistic models* for subspace clustering. In particular, we show that the difficult problem of the parameter settings

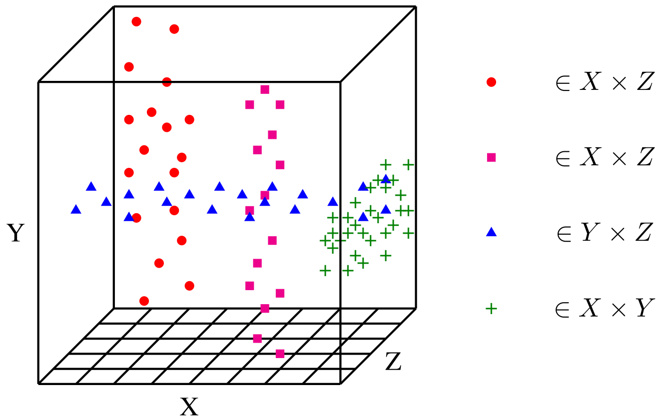


FIG. 1 – Example of four clusters embedded in different subspaces.

of subspace clustering algorithms can be seen as a *model selection* problem in the framework of probabilistic models. It thus allows us to propose a method that does not require any input parameter from the user. We also show that allowing the clusters to overlap may be necessary in that field.

Moreover, the problem of *noise detection* can also naturally be included into the probabilistic framework. Yet another contribution of this work is to tackle the problem of providing interpretable results. And we will see that detecting the noise that may exist in the data can help to provide more understandable results. Finally, another advantage of using probabilistic models is that it allows us to naturally mix different types of attributes, under some specific assumptions.

The rest of the paper is organized as follows : in section 2 we present existing subspace clustering methods and discuss their performances ; we then describe how to adapt probabilistic models for subspace clustering and propose a new algorithm called **SuSE** in section 3 ; the results of our experiments, conducted on artificial as well as real datasets, and where **SuSE** is compared to other existing methods, are then reported in section 4 ; finally, section 5 concludes the paper and suggests topics for future research.

2 Subspace clustering

The subspace clustering issue has been first introduced in (Agrawal *et al.*, 1998). Many other methods emerged then, among which two families can be distinguished according to their subspace search method :

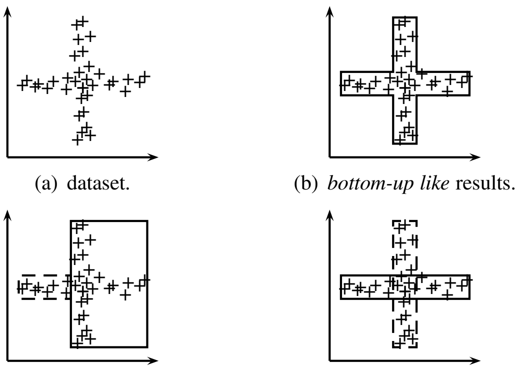
1. *bottom-up* subspace search methods (Agrawal *et al.*, 1998; Cheng *et al.*, 1999; Nagesh *et al.*, 1999; Kailing *et al.*, 2004) seek to find clusters in subspaces of increasing dimensionality, and produce as output a set of clusters that can overlap ;
2. and *top-down* subspace search methods (Aggarwal *et al.*, 1999; Woo & Lee, 2002; Yip *et al.*, 2003; Sarafis *et al.*, 2003; Domeniconi *et al.*, 2004) use *k-means*

like methods with original techniques of local feature selection, and produce as output a partition of the dataset.

In (Parsons *et al.*, 2004), the authors have studied and compared these methods. They point out that every method requires input parameters difficult to set for the user, and that influence the results (density threshold, mean number of relevant dimensions of the clusters, minimal distance between clusters, etc.).

Besides, the existing methods do not tackle the problem of handling the noise that may exist in the data. And no proposition was made for producing an interpretable output, although being understandable in the field of clustering is an important challenge. Yet we will see that the noise detection may also help to provide more understandable results. Moreover, although a proposition was made to integrate categorical attributes in *bottom-up* approaches, all experiments were conducted on numerical data only.

Finally, let us present a case where both types of existing methods behave badly compared to what we should expect from subspace clustering algorithms. It is the case for the data in figure 2, where one cluster is defined on one dimension and takes random values on another one, and conversely for the other cluster.



(a) dataset. (b) *bottom-up like* results.

(c) *k-means like* results. (d) *model based* results.

FIG. 2 – A case where existing subspace clustering methods behave badly contrary to methods based on probabilistic models.

In such a case, for *bottom-up* subspace search methods, all points belong to the same cluster because they form a continuous zone. These methods thus tend to describe data like these as a unique 2D-space cluster instead of as a pair of 1D-space clusters. It then becomes worse with many dimensions. Conversely, for *k-means like* methods, as intersections are not allowed, the two clusters may not be retrieved. On the other hand, methods based on probabilistic models and the EM algorithm (Ye & Spetsakis, 2003) are able to identify the two clusters.

However, it is well known that the methods based on probabilistic models and the EM algorithm may be slow to converge. Moreover, it would be interesting to adapt such methods to subspace clustering, by designing a model able to identify the subspaces in which each cluster is embedded. Finally, it would also be interesting to use a model able to handle different types of attributes, and to provide an interpretable output.

In the next section, we present such a new statistical subspace clustering algorithm called **SuSE** . This algorithm belongs to the family of *top-down* subspace search methods. We will show that assuming that the data values follow independent distributions on each dimension helps to resolve many issues we have presented. And we will see the interest in using such statistical approach, in particular in order to resolve the difficult problem of parameter settings.

3 Algorithm SuSE

Let us first introduce some notations. We denote by N the number of data points of the input dataset and M the number of dimensions on which they are defined. These dimensions can be numerical as well as categorical. We suppose values on numerical dimensions are normalized (so that all values belong to the same interval). And we denote by $Categories_d$ the set of all possible categories on a categorical dimension d, and $Frequences_d$ the frequences of all these categories within the dataset.

3.1 Probabilistic model

The basis of our model is the classical mixture of probability distributions $\theta = (\theta_1, ..., \theta_K)$ where each θ_k is the vector of parameters associated with the k^{th} cluster to be found, denoted by C_k (we set to K the total number of clusters).

Besides, we assume that the data values follow independent distributions on each dimension. Thus, our model is less expressive than the classical one that takes into account the possible correlations between dimensions. But it allows us to naturally mix numerical and categorical dimensions. We will see that it also allows us to extract some of the dimensions considered as more relevant to each cluster. Besides, the method is thus faster than with the classical model because our model needs less parameters ($O(M)$ instead of $O(M^2)$) and operations on matrices are avoided. And finally, it is thus adapted to the presentation of the results as a set of rules (hypercubes in subspaces of the original description space are easily understandable by humans) because each dimension of each cluster is characterized independently from one another.

In our model, we suppose that the data follow gaussian distributions on numerical dimensions and multinomial distributions on categorical dimensions. So the model has the following parameters θ_k for each cluster C_k : π_k denotes its weight, μ_{kd} its mean and σ_{kd} its standard deviation on the numerical dimensions d, and $Freqs_{kd}$ the frequences of each category on the categorical dimensions d.

Finally, in order to adapt the model for subspace clustering, we add the parameter R that indicates how many relevant dimensions to consider for the clusters. And we add to the parameters of each cluster C_k the set M_k, of size R, of the dimensions considered as the most relevant to the cluster.

To make these *local feature selections*, we first associate to each dimension d of each cluster C_k a local weight W_{kd} that indicates its relevance to the cluster. These weights are computed according to the shape of the distribution of the cluster on the dimension. For example for numerical dimensions, a high standard deviation will induce a low weight on the dimension whereas a low standard deviation will induce a high weight in determining if a data point belongs to the corresponding cluster.

So these weights are computed as follows. For numerical dimensions, it is the ratio between the local and the global standard deviation according to μ_{kd}. And for categorical dimensions, it is the relative frequence of the most probable category.

$$W_{kd} = \begin{cases} 1 - \frac{\sigma_{kd}^2}{\Sigma_{kd}^2} \text{ , with } \Sigma_{kd}^2 = \frac{1}{N}\sum_i (X_{id} - \mu_{kd})^2 & \text{if } d \text{ numerical} \\ \frac{Freqs_{kd}(cat) - Frequences_d(cat)}{1 - Frequences_d(cat)} & \text{if } d \text{ categorical} \\ \text{with } cat = Argmax_{\{c \in Categories_d\}} Freqs_{kd}(c) & \end{cases}$$

So the weight W_{kd} reflects the capability of the dimension d to discriminate between the data points that belong to the specific cluster C_k and the other ones. And the R dimensions of highest weights, that correspond to the most relevant dimensions of the cluster C_k, can be selected.

By this model, we set that all the clusters have the same number of relevant dimensions, although the dimensions selected for each cluster may be different. If it is not the case, then some irrelevant dimensions may be selected by some clusters. However, the influence of such irrelevant dimensions would be lower than the one of the relevant dimensions.

3.2 EM algorithm

Given a set D of N data points $\overrightarrow{X_i}$, *Maximum Likelihood Estimation* is used to estimate the model parameters that best fit the data. To do this, the *EM algorithm* is an effective two-step process that seeks to optimize the *log-likelihood* of the model θ according to the dataset D, $LL(\theta|D) = \sum_i \log P(\overrightarrow{X_i}|\theta)$.

1. E-step (*Expectation*) : find the class probability of each data point according to the current model parameters.
2. M-step (*Maximization*) : update the model parameters according to the new class probabilities.

These two steps iterate until a stopping criterion is reached. Classicaly, it stops when $LL(\theta|D)$ increases less than a small positive constant δ from one iteration to another.

The E-step consists in computing the membership probability of each data point $\overrightarrow{X_i}$ to each cluster C_k with parameters θ_k. In our case, the dimensions are assumed to be independent, and each cluster has its own set of relevant dimensions M_k. So the membership probability of a data point to a cluster is the product of membership probabilities on each dimension considered as relevant for the cluster. Besides, to avoid that a probability equal to zero on one dimension cancels the global probability, we use a very small positive constant ϵ.

$$P(\overrightarrow{X_i}|\theta_k) = \prod_{d \in M_k} max(P(X_{id}|\theta_{kd}), \epsilon)$$

$$P(X_{id}|\theta_{kd}) = \begin{cases} \frac{1}{\sqrt{2\pi}\sigma_{kd}} e^{-\frac{1}{2}\left(\frac{X_{id}-\mu_{kd}}{\sigma_{kd}}\right)^2} & \text{if } d \text{ numerical} \\ Freqs_{kd}(X_{id}) & \text{if } d \text{ categorical} \end{cases}$$

$$P(\overrightarrow{X_i}|\theta) = \sum_{k=1}^{K} \pi_k \times P(\overrightarrow{X_i}|\theta_k)$$

$$P(\theta_k|\overrightarrow{X_i}) = \frac{\pi_k \times P(\overrightarrow{X_i}|\theta_k)}{P(\overrightarrow{X_i}|\theta)}$$

Then the M-step consists in updating the model parameters according to the new class probabilities as follows :

$$\pi_k = \frac{1}{N} \sum_i P(\theta_k|\overrightarrow{X_i})$$

$$\mu_{kd} = \frac{\sum_i X_{id} \times P(\theta_k|\overrightarrow{X_i})}{\sum_i P(\theta_k|\overrightarrow{X_i})}$$

$$\sigma_{kd} = \sqrt{\frac{\sum_i P(\theta_k|\overrightarrow{X_i}) \times (X_{id} - \mu_{kd})^2}{\sum_i P(\theta_k|\overrightarrow{X_i})}}$$

$$Freqs_{kd}(cat) = \frac{\sum_{\{i|X_{id}=cat\}} P(\theta_k|\overrightarrow{X_i})}{\sum_i P(\theta_k|\overrightarrow{X_i})} \; \forall \, cat \in Categories_d$$

In order to cope with the problem of slow convergence with the classical EM algorithm, the following *k-means like* stopping criterion can be used : stop whenever the membership of each data point to their most probable cluster does not change. To do this, we introduce a new view on each cluster C_k, corresponding to the set of data points that belong to it :

$$S_k = \{\overrightarrow{X_i}|Argmax_{j=1}^{K} P(\overrightarrow{X_i}|\theta_j) = k\}$$

The set of all S_k thus defines a partition on the dataset. However, as we discussed earlier on the example of figure 2, this ability to provide a partition on the object space does not prevent us from considering clusters that may overlap on the description space.

And finally, to cope with the problem of sensitivity to the choice of the initial solution, we run the algorithm many times with random initial solutions and keep the model that optimizes the *log-likelihood* $LL(\theta|D)$.

3.3 Model selection

At this stage, our algorithm needs two input parameters : the number K of clusters to be found, and the number R of dimensions to be selected for each cluster. An important advantage of using a probabilistic model over other existing methods is that finding the most appropriate values of these two parameters can be seen as a model selection problem.

So we can for example use the BIC criterion (Ye & Spetsakis, 2003) that consists in adding to the log-likelihood of the model to the data a term that penalizes more complex models. It thus tries to find a compromise between the fit of the model to the data, and the complexity of the model used.

$$BIC(\theta|D) = -2 \times LL(\theta|D) + M_\theta \times \log N$$

M_θ represents the number of independent parameters of the model :

$$M_\theta = \sum_{k=1}^{K} \sum_{d \in M_k} \begin{cases} 2 & \text{if } d \text{ numerical} \\ |Categories_d| & \text{if } d \text{ categorical} \end{cases}$$

BIC criterion must be minimized to optimize the likelihood of the model to the data. So to find the model that best fit the data, we consider different models with different values for the parameters K and R, and the model that minimizes the BIC value is kept.

Contrary to the other existing subspace clustering methods, we thus propose a way to automatically find the most appropriate values for the model parameters. We thus do not need the user to provide any prior knowledge. The relevance of this method will be studied in the next section.

Finally, another advantage of using probabilistic models for subspace clustering is that detecting the noise that may exist in the data can be naturally integrated into the method, by adding a uniform cluster into the model. Moreover, we will see in the next section that handling the noise can also help to get more understandable results.

4 Experiments

Experiments were conducted on artificial as well as real datasets. The first ones are used to observe the evolution of the BIC value according to the number of clusters expected and the number of dimensions selected for each cluster. We also use artificial datasets to observe the robustness of **SuSE** faced with different types of datasets, in particular datasets containing noise. Then experiments on real datasets are conducted according to the methodology proposed in (Candillier *et al.*, 2005), and thus show the effectiveness of our method on real-life data.

In order to compare our method with existing ones, we conduct these experiments on numerical-only datasets. Three other clustering algorithms are used in these experiments :

- K-means, the well-known full-space clustering algorithm based on the evolution of K centroids that represent the K clusters to be found.
- LAC (Domeniconi *et al.*, 2004), an effective *top-down like* subspace clustering method that is based on K-means and associates with each centroid a vector of weights on each dimension. At each step and for each cluster, these weights on each dimension are updated according to the dispersion of the members of the cluster on the dimension (the greater the dispersion, the less the weight).

– And EMI refers to clustering by learning a mixture of gaussians with the EM algorithm under the independence assumption on the dimensions, but without performing local feature selection as is done by **SuSE** .

4.1 Artificial datasets

Artificial datasets are generated according to the following parameters : N the number of data points in the dataset, M the number of (numerical) dimensions on which they are defined, L the number of clusters, m the mean dimensionality of the subspaces on which the clusters are defined, SD_m and SD_M the minimum and maximum standard deviation of the coordinates of the data points that belong to a same cluster, from its centroid and on its relevant dimensions.

L random data points are chosen on the M-dimensional description space and used as seeds of the L clusters $(C_1, ..., C_L)$ to be generated. Let us denote them by $(\overrightarrow{O_1}, ..., \overrightarrow{O_L})$. With each cluster is associated a subset of the N data points and a subset (of size close to m) of the M dimensions that will define its specific subspace. Then the coordinates of the data points that belong to a cluster C_k are generated according to a normal distribution with mean O_{kd} and standard deviation $sd_{kd} \in [SD_m..SD_M]$ on its specific dimensions d. They are generated uniformly between 0 and 100 on the other dimensions.

For all experiments, 100 artificial datasets are generated with N varying between 50 and 300, M between 10 and 50, L between 2 and 5, m between 3 and 10, $SD_m = 3$ and $SD_M = 9$. Then averages on the expected measures over the various trials are computed.

Our first experiments concern the evolution of the BIC value according to the number R of relevant dimensions selected for each cluster, when the number K of clusters expected is provided. Figure 3 shows such a curve, and thus points out that the BIC value decreases until R reaches m, and then increases. So it experimentally shows that the BIC criterion can be used to automatically determine the most appropriate number of relevant dimensions for each cluster.

Similarly, figure 4 shows the 3D plot of the BIC value according to the number K of clusters to be found, and the number R of relevant dimensions selected for each cluster. For a better visualization of the results, -BIC is reported instead of BIC. It thus experimentally points out that the optimum BIC value is reached when K reaches L and R reaches m.

4.2 Noisy datasets

We also conducted experiments on artificial datasets to observe the robustness of our method to noise. Figure 5 shows the results obtained with or without taking into account the noise that may exist in the data. In this example, we see that detecting the noise leads to more understandable results.

Figure 6 then shows the resistance of **SuSE** to noise, compared to EMI and LAC. The accuracy of the partition is measured by the average purity of the clusters (the purity

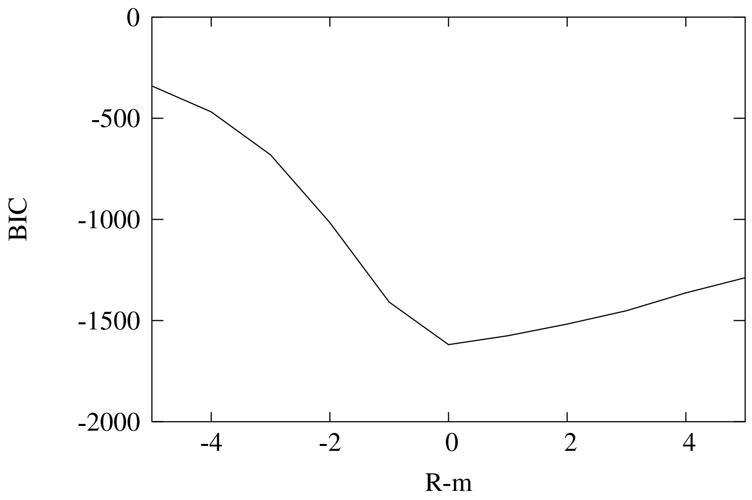


FIG. 3 – Evolution of the BIC value according to the number R of relevant dimensions selected for each cluster.

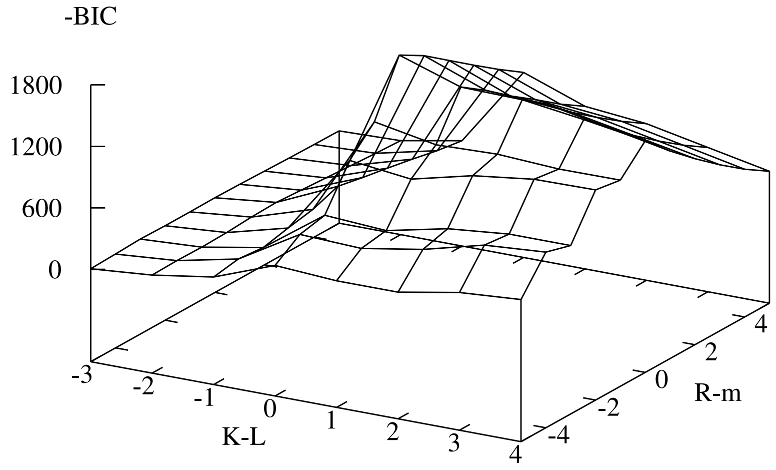
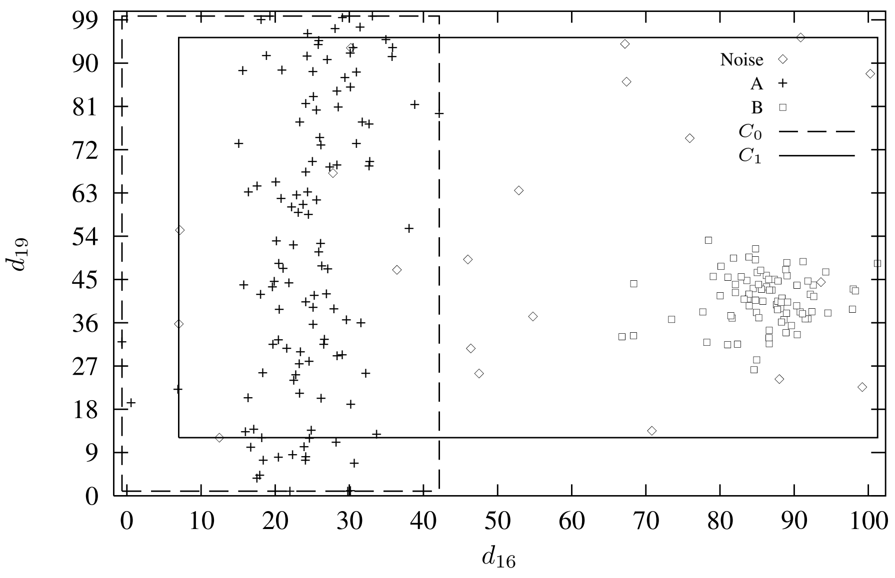
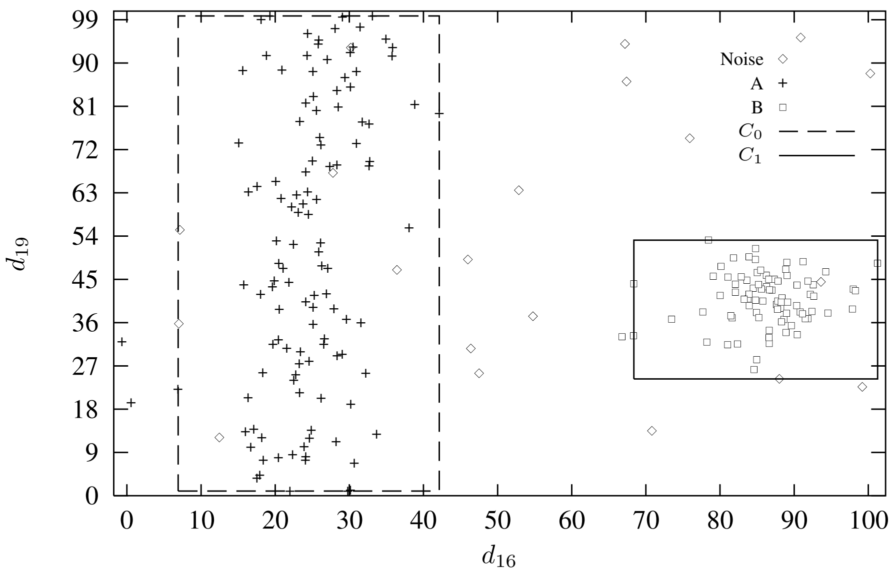


FIG. 4 – Evolution of the BIC value according to the number K of clusters to be found and the number R of relevant dimensions selected for each cluster.



(a) **SuSE** without noise detection.



(b) **SuSE** with noise detection.

FIG. 5 – Example of the interest of the noise detection.

of a cluster is the maximum percentage of data points that belong to the same initial concept).

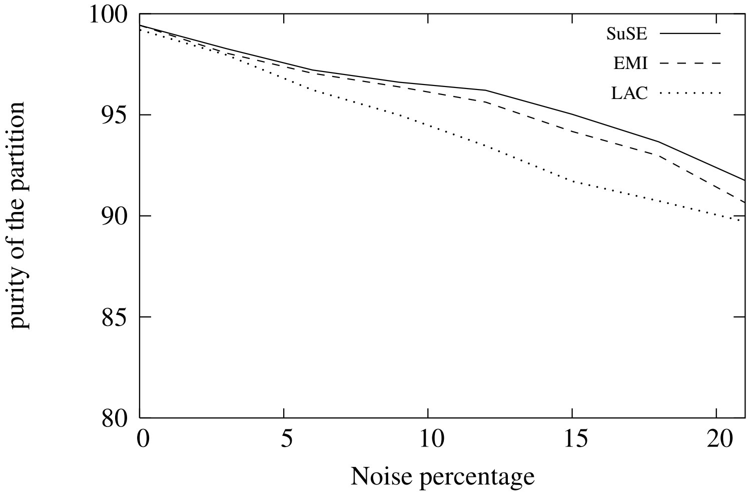


FIG. 6 – Purity of the partition according to the percentage of noise in the dataset.

We can thus observe that **SuSE** is more robust to noise than EMI and LAC. Our method is also robust to missing values. When summing over all the data values on one dimension, the only thing to do is to ignore the missing values.

Let us finally note that our method is still robust even if the data are generated by uniform distributions inside given intervals on the relevant dimensions of the clusters, instead of normal distributions.

4.3 Real datasets

We also conducted various experiments on real datasets, following the methodology proposed in (Candillier *et al.*, 2005) that consists in :

1. performing a supervised learning on a given dataset with classes information
2. performing a supervised learning on the same dataset enriched with the information coming from the clustering algorithm to be evaluated
   – perform a clustering without using the classes information
   – create new attributes from these results
   – add these new attributes to the dataset
   – and perform the supervised learning on the enriched dataset
3. and comparing the classification errors of both methods

Thus, if the results of the supervised learning algorithm are improved when some extra-knowledge is added from the clustering process, then we conjecture that it means that the clustering process managed to capture some new meaningful and useful information. And the decrease of the error rate of the supervised method when it is helped by the information coming from the clustering allows us to quantify the interest of the clustering algorithm.

One way of creating new attributes from the results of a clustering is for example to add to each data point an identifier of the cluster it belongs to. We could also add to each data point a set of attributes referring to the center of the cluster it belongs to. In these experiments, we use C4.5 (Quinlan, 1993) as the supervised algorithm, but it has been experimented in (Candillier *et al.*, 2005) that the results do not depend on the supervised algorithm used.

To evaluate the improvement in the results of C4.5 with or without the new information coming from the clustering process, we test both methods on various independent datasets coming from the UCI Machine Learning Repository (Blake & Merz, 1998). On each dataset, we perform five 2-fold cross-validations, as proposed in (Dietterich, 1998). For each 2-fold cross-validation, we compute the balanced error rates of both methods.

Table 1 reports the error rates of C4.5 on the initial datasets, and on the datasets enriched with the corresponding clustering algorithms. Rand is a random clustering that is used as a reference.

	C4.5 alone	C4.5 + Rand	C4.5 + K-means	C4.5 + LAC	C4.5 + EMI	C4.5 + **SuSE**
ecoli	48.5	48.3	42.8	**40.3**	42	43
glass	**32.6**	40.8	35.7	37	40.4	35.8
image	4.8	6	4.8	4.6	4.6	**4.2**
iono	14.1	15.8	14.2	13.1	**9.8**	10.9
iris	7.3	7.9	6.7	**3.7**	5.1	4.8
pima	31	35	32.1	32.1	30.8	**30.5**
sonar	31	35.2	30	28.8	28.8	**27.4**
vowel	29.5	38.5	25	26.4	24.1	**23.7**
wdbc	5.9	6.8	4.6	3.9	5.1	**3.8**
wine	8.7	8.8	10.4	9.6	**2.7**	4.1

TAB. 1 – Balanced error rates (in %) of C4.5 enriched by clustering algorithms. The bold values correspond to the minimum error rates obtained on each dataset.

From this table, we can already observe that most of the time, the results of C4.5 are improved when some information coming from real clustering algorithms are added, whereas adding information from a random clustering degrads the results. Besides, we see that the results of the methods based on the use of probabilistic models are often better than those of K-means based methods.

Then four measures are used to compare the results of C4.5 alone with those of C4.5 helped with the corresponding clustering algorithms :

– *nb wins* : the number of wins of each method
– *sign wins* : the number of significant wins, using the *5×2cv F-test* (Alpaydin, 1999) to check if the results are significantly different

- *wilcoxon* : the wilcoxon signed rank test, that indicates if a method is significantly better than another one on a set of independent problems (if its value is above 1.96)
- and *av perf* : the mean balanced error rate (in %)

Table 2 shows the results of such an evaluation. The first column concerns the measures obtained using C4.5 on the initial dataset, the second column using C4.5 on the dataset enriched with information coming from the random clustering, and the next ones using C4.5 on the dataset enriched with information coming from the corresponding clustering algorithm.

	C4.5 alone	C4.5 + Rand	C4.5 + K-means	C4.5 + LAC	C4.5 + EMI	C4.5 + **SuSE**
nb wins	-	1/9	5/4	7/3	9/1	**9/1**
sign wins	-	0/1	0/0	1/0	2/0	**3/0**
wilcoxon	-	-2.67	-0.05	1.31	1.83	**2.36**
av perf	21.3	24.3	20.6	20	19.3	**18.8**

TAB. 2 – Comparison of C4.5 alone with C4.5 enriched by clustering algorithms.

It thus shows that **SuSE** is the only clustering algorithm among the ones tested here that significantly helps C4.5 improve its results, according to the wilcoxon signed rank test. It is significantly better on 3 datasets according to the *5×2cv F-test*. But as **SuSE** , EMI improves the results of C4.5 nine times over ten, contrary to K-means and LAC. All algorithms improve the results of C4.5 on average, except the random clustering.

5 Conclusion

We have shown in this paper the interest in using probabilistic models for subspace clustering. Indeed, we have seen that it allows us to transform the difficult problem of the parameter settings into a model selection problem, so that the most appropriate number of relevant dimensions to consider for each cluster can be determined automatically, instead of requiring the user to specify it, as is done by the other subspace clustering methods. Besides, we have also shown the interest in allowing the clusters to overlap in that field.

We have also pointed out the interest in assuming that the data values follow independent distributions on each dimension. Indeed, it allows us to speed up the algorithm, to naturally mix different types of attributes, and to provide an understandable result as a set of rules. However, in the case where correlations between dimensions exist, our method does not provide irrelevant results. Instead, in many cases, it points out the correlations by generating clusters along the axes defined by the correlations. An example of such results is presented in figure 7.

The experiments we conducted on artificial datasets pointed out the robustness of our method to noise. Moreover, we have seen that detecting the noise may help to get more understandable results. Then experiments on real datasets pointed out the relevance of using probabilistic models for subspace clustering. In particular, methods based on the

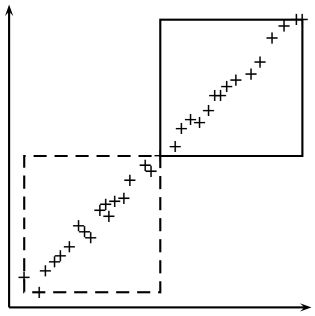


FIG. 7 – Results of **SuSE** when a correlation between dimensions exists.

use of probabilistic models have been shown to outperform K-means based methods. And more specifically, **SuSE** has been shown to outperform EMI, thus pointing out the relevance of our proposed method for selecting the most appropriate number of relevant dimensions.

To continue our investigations in that field, we could now conduct more experiments and compare our method with many others, on many other artificial and real datasets. Finally, we could also study more in detail how to design an efficient way to reach the optimum BIC value, referring to the most appropriate number of clusters and number of relevant dimensions to be considered.

Références

AGGARWAL C. C., WOLF J. L., YU P. S., PROCOPIUC C. & PARK J. S. (1999). Fast algorithms for projected clustering. In *ACM SIGMOD Int. Conf. on Management of Data*, p. 61–72.

AGRAWAL R., GEHRKE J., GUNOPULOS D. & RAGHAVAN P. (1998). Automatic subspace clustering of high dimensional data for data mining applications. In *ACM SIGMOD Int. Conf. on Management of Data*, p. 94–105, Seattle, Washington.

ALPAYDIN E. (1999). Combined 5x2cv F-test for comparing supervised classification learning algorithms. *Neural Computation*, **11**(8), 1885–1892.

BERKHIN P. (2002). *Survey Of Clustering Data Mining Techniques*. Rapport interne, Accrue Software, San Jose, California.

BLAKE C. & MERZ C. (1998). UCI repository of machine learning databases [http ://www.ics.uci.edu/~mlearn/MLRepository.html].

CANDILLIER L., TELLIER I., TORRE F. & BOUSQUET O. (2005). Cascade evaluation. NIPS 2005 Workshop on Theoretical Foundations of Clustering.

CHENG C. H., FU A. W.-C. & ZHANG Y. (1999). Entropy-based subspace clustering for mining numerical data. In *Knowledge Discovery and Data Mining*, p. 84–93.

DIETTERICH T. G. (1998). Approximate statistical test for comparing supervised classification learning algorithms. *Neural Computation*, **10**(7), 1895–1923.

DOMENICONI C., PAPADOPOULOS D., GUNOPOLOS D. & MA S. (2004). Subspace clustering of high dimensional data. In *SIAM Int. Conf. on Data Mining*.

KAILING K., KRIEGEL H.-P. & KRÖGER P. (2004). Density-connected subspace clustering for high-dimensional data. In *SIAM Int. Conf. on Data Mining*, p. 246–257.

NAGESH H., GOIL S. & CHOUDHARY A. (1999). *MAFIA : Efficient and scalable subspace clustering for very large data sets*. Rapport interne, Northwestern University.

PARSONS L., HAQUE E. & LIU H. (2004). Evaluating subspace clustering algorithms. In *Workshop on Clustering High Dimensional Data and its Applications, SIAM Int. Conf. on Data Mining*, p. 48–56.

QUINLAN J. R. (1993). *C4.5 : Programs for Machine Learning*. KAUFM.

SARAFIS I. A., TRINDER P. W. & ZALZALA A. M. S. (2003). Towards effective subspace clustering with an evolutionary algorithm. In *IEEE Congress on Evolutionary Computation*, Canberra, Australia.

WOO K.-G. & LEE J.-H. (2002). *FINDIT : a fast and intelligent subspace clustering algorithm using dimension voting*. PhD thesis, Korea Advanced Institute of Science and Technology, Department of Electrical Engineering and Computer Science.

YE L. & SPETSAKIS M. (2003). *Clustering on Unobserved Data using Mixture of Gaussians*. Rapport interne, York University, Toronto, Canada.

YIP K. Y., CHEUNG D. W. & NG M. K. (2003). A highly-usable projected clustering algorithm for gene expression profiles. In *3rd ACM SIGKDD Workshop on Data Mining in Bioinformatics*, p. 41–48.