A Feature Selection Wrapper for Mixtures

Mário A. T. Figueiredo[1], Anil K. Jain[2], and Martin H. Law[2]

[1] Instituto de Telecomunicações, and
Departamento de Engenharia Electrotécnica e de Computadores.
Instituto Superior Técnico, 1049-001 Lisboa, PORTUGAL
E-mail: mtf@lx.it.pt

[2] Department of Computer Science and Engineering
Michigan State University, East Lansing, MI 48824, U.S.A.
E-mail: jain@cse.msu.edu and lawhiu@cse.msu.edu

Abstract

We propose a feature selection approach for clustering which extends Koller and Sahami's mutual-information-based criterion to the unsupervised case. This is achieved with the help of a mixture-based model and the corresponding expectation-maximization algorithm. The result is a backward search scheme, able to sort the features by order of relevance. Finally, an MDL criterion is used to prune the sorted list of features, yielding a feature selection criterion. The proposed approach can be classified as a *wrapper*, since it wraps the mixture estimation algorithm in an outer layer that performs feature selection. Preliminary experimental results show that the proposed method has promising performance.

1 Introduction

A great deal of research has been devoted to the *feature selection* (FS) problem in supervised learning [1–3] (a.k.a. *variable selection* or *subset selection* [4]). FS is important for a variety of reasons: it may improve the performance of classifiers learned from limited amounts of data; it leads to more economical (both in storage and computation) classifiers; in many cases, it leads to interpretable models. However, FS for unsupervised learning has not received much attention.

In mixture-based unsupervised learning (clustering [5]), each group of data is modelled as having been generated according to a probability distribution with known form. Learning then consists of estimating the parameters of these distributions, and is usually done via the *expectation-maximization* (EM) algorithm [6–8]. Although standard EM assumes that the number of components/groups is known, extensions which also estimate are available [9]. Of course, the number of components can also be estimated using more standard model selection criteria such as MDL or BIC (see [9] for references).

Work partially supported by the Foundation for Science and Technology (Portugal), grant POSI/33143/SRI/2000, and the Office of Naval Research (USA), grant 00014-01-1-0266.

Here, we address the FS problem in mixture-based clustering, by extending the mutual-information based criterion proposed in [1] to the unsupervised context. The proposed approach can be classified as a *wrapper* [2], in the sense that the feature selection procedure is *wrapped around* the EM algorithm. This wrapper is able to sort the variables by order of relevance, using backward search. An MDL criterion is used to prune this sorted list leaving a set of *relevant* features.

Finally, let us briefly review previously proposed FS methods in unsupervised learning. In [10], a heuristic to compare the quality of different feature subsets, based on cluster separability, is suggested. A Bayesian approach used in [11] evaluates different feature subsets and numbers of clusters for multinomial mixtures. In [12], the clustering tendency of each feature is assessed by an entropy index. A genetic algorithm was used in [13] for FS in k-means clustering. Finally, [14] uses the notion of "category utility" for FS in a conceptual clustering task.

2 Mixture Based Clustering and the EM Algorithm

Mixture models allow a probabilistic approach to clustering ([6–8]) in which model selection issues (*e.g.*, number of clusters) can be formally addressed. Given n i.i.d. samples $\mathcal{Y}=\{\mathbf{y}_1,...,\mathbf{y}_n\}$, the log-likelihood of a k-component mixture is

$$\log p(\mathcal{Y}|\boldsymbol{\theta}) \;=\; \log\prod_{i=1}^{n} p(\mathbf{y}_i|\boldsymbol{\theta}) = \sum_{i=1}^{n}\log\sum_{m=1}^{k}\alpha_m p(\mathbf{y}_i|\boldsymbol{\theta}_m), \tag{1}$$

where $\alpha_1,...,\alpha_k \geq 0$ are the *mixing probabilities* ($\sum_m \alpha_m = 1$), $\boldsymbol{\theta}_m$ is the set of parameters of the m-th component, and $\boldsymbol{\theta} \equiv \{\boldsymbol{\theta}_1,...,\boldsymbol{\theta}_k,\alpha_1,...,\alpha_k\}$ is the full set of parameters. Each $\mathbf{y}_i$ is a d-dimensional vector of features $[y_{i,1},...,y_{i,d}]^T$, and we assume that all the components have the same form (*e.g.*, Gaussian).

Neither the *maximum likelihood* (ML) nor the *maximum a posteriori* (MAP) estimates, $\widehat{\boldsymbol{\theta}}_{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta}}\{\log p(\mathcal{Y}|\boldsymbol{\theta})\}$ and $\widehat{\boldsymbol{\theta}}_{\mathrm{MAP}} = \arg\max_{\boldsymbol{\theta}}\{\log p(\mathcal{Y}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta})\}$, respectively, can be found analytically. The usual alternative is the EM algorithm [7, 8, 15, 16], which finds local maxima of $\log p(\mathcal{Y}|\boldsymbol{\theta})$ or $[\log p(\mathcal{Y}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta})]$.

EM is based on seeing $\mathcal{Y}$ as *incomplete* data, the *missing* part being a set of n labels $\mathcal{Z} = \{\mathbf{z}_1,...,\mathbf{z}_n\}$, flagging which component produced each sample. Each label is a binary vector $\mathbf{z}_i = [z_{i,1},...,z_{i,k}]$, with $z_{i,m}=1$ and $z_{i,p}=0$, for $p \neq m$, meaning that $\mathbf{y}_i$ is a sample of $p(\cdot|\boldsymbol{\theta}_m)$. The complete log-likelihood (*i.e.*, given both $\mathcal{Y}$ and $\mathcal{Z}$) is

$$\log p(\mathcal{Y},\mathcal{Z}|\boldsymbol{\theta}) = \sum_{i=1}^{n}\sum_{m=1}^{k} z_{i,m}\log\left[\alpha_m p(\mathbf{y}_i|\boldsymbol{\theta}_m)\right]. \tag{2}$$

The EM algorithm produces a sequence of estimates $\{\widehat{\boldsymbol{\theta}}(t),\ t=0,1,2,...\}$ by alternatingly applying two steps (until some convergence criterion is met):

• **E-step:** Compute the conditional expectation $\mathcal{W} = E[\mathcal{Z}|\mathcal{Y},\widehat{\boldsymbol{\theta}}(t)]$, and plug it into $\log p(\mathcal{Y},\mathcal{Z}|\boldsymbol{\theta})$, yielding the so-called Q-function: $Q(\boldsymbol{\theta},\widehat{\boldsymbol{\theta}}(t)) = \log p(\mathcal{Y},\mathcal{W}|\boldsymbol{\theta})$.

Since the elements of $\mathcal{Z}$ are binary, their conditional expectations are given by

$$w_{i,m} \equiv E\left[z_{i,m}|\,\mathcal{Y}, \widehat{\boldsymbol{\theta}}(t)\right] = \Pr\left[z_{i,m}=1|\,\mathbf{y}_i, \widehat{\boldsymbol{\theta}}(t)\right] \propto \widehat{\alpha}_m(t)\; p(\mathbf{y}_i|\widehat{\boldsymbol{\theta}}_m(t)) \qquad (3)$$

(normalized such that $\sum_m w_{i,m} = 1$). Notice that α_m is the *a priori* probability that $z_{i,m} = 1$ (*i.e.*, that $\mathbf{y}_i$ belongs to cluster m) while $w_{i,m}$ is the corresponding *a posteriori* probability, after observing $\mathbf{y}_i$.
• **M-step:** Update the parameter estimates, $\widehat{\boldsymbol{\theta}}(t+1) = \arg\max_{\boldsymbol{\theta}} \{Q(\boldsymbol{\theta}, \widehat{\boldsymbol{\theta}}(t)) + \log p(\boldsymbol{\theta})\}$, in the case of MAP estimation, or without $\log p(\boldsymbol{\theta})$ in the ML case.

3 Feature Selection for Mixtures

3.1 Likelihood Formulation

Consider the example in Fig. 1: a 2-component bivariate Gaussian mixture. In this example, y_2 is clearly irrelevant for the "mixture nature" of the data. However, *principal component analysis* (PCA, one of the standard non-supervised feature sorting methods) of this data would declare y_2 as more relevant because it explains more data variance than y_1.

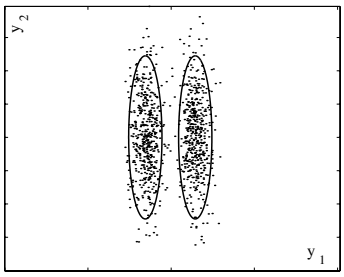

Fig. 1. Feature y_1 is relevant to the mixture nature of the data, while y_2 is not.

To address the FS problem for mixtures, we divide the available feature set $\mathbf{y} = [y_1, ..., y_d]$ into two subsets $\mathbf{y}_U$ and $\mathbf{y}_N$. Here, U and N (standing for "useful" and "non-useful") are two disjoint sub-sets of indices such that $U \cup N = \{1, 2, ..., d\}$. Our key assumption is that the non-useful features are independent of the useful ones, and their distribution is the same for all classes/clusters, *i.e.*,

$$p(\mathbf{y}|U, \boldsymbol{\theta}_U, \boldsymbol{\theta}_N) = p(\mathbf{y}_N|\boldsymbol{\theta}_N) \sum_{m=1}^{k} \alpha_m p(\mathbf{y}_U|\boldsymbol{\theta}_{m,U}), \qquad (4)$$

where $\boldsymbol{\theta}_N$ is the set of parameters characterizing the distribution of the non-useful features, and $\boldsymbol{\theta}_U = [\boldsymbol{\theta}_{1,U}, ..., \boldsymbol{\theta}_{k,U}]$ is the set of parameters characterizing

the mixture distribution of the useful features. Notice that we only need to specify U, because $N = \{1, 2, ..., d\} \backslash U$. The feature selection problem is then to find U and the corresponding parameter $\boldsymbol{\theta} = [\boldsymbol{\theta}_U, \boldsymbol{\theta}_N]$. Let us highlight some aspects of this formulation:

- Consider maximizing the log-likelihood, given observations $\mathcal{Y} = \{\mathbf{y}_1, ..., \mathbf{y}_n\}$,

$$\log p(\mathcal{Y}|U, \boldsymbol{\theta}_U, \boldsymbol{\theta}_N) = \sum_{i=1}^{n} \log p(\mathbf{y}_{i,N}|\boldsymbol{\theta}_N) + \sum_{i=1}^{n} \log \sum_{m=1}^{k} \alpha_m p(\mathbf{y}_{i,U}|\boldsymbol{\theta}_{m,U}), \quad (5)$$

  with respect to U, $\boldsymbol{\theta}_U$ and $\boldsymbol{\theta}_N$. The result would be $U = \{1, ..., d\}$ (as noted in [11]), because a mixture is a more general model and so we can never decrease the likelihood by increasing the number of useful features. This shows that the problem requires some model selection criterion.
- Testing all possible 2^d partitions of $\{1, 2, ..., d\}$ into U and N is prohibitive, even for moderate d. The standard alternative is to use sub-optimal methods, such as sequential forward/backward search (SFS/SBS) schemes [3].

3.2 Connection with Feature Selection for Supervised Learning

Assume the class labels and the full feature vector follow some probability function $p(\mathbf{z}, \mathbf{y})$. A subset of features $\mathbf{y}_N$ is non-useful/irrelevant if it is conditionally independent of the labels, given the useful features $\mathbf{y}_U$ (see [1]), *i.e.*, if

$$p(\mathbf{z}|\mathbf{y}) = p(\mathbf{z}|\mathbf{y}_U, \mathbf{y}_N) = p(\mathbf{z}|\mathbf{y}_U). \quad (6)$$

Observation of the model in (4) reveals that we can look at the m-th mixture component as being $p(\mathbf{y}|\boldsymbol{\theta}_m) = p(\mathbf{y}_U|\boldsymbol{\theta}_{m,U})p(\mathbf{y}_N|\boldsymbol{\theta}_N)$. The outcome of the E-step of the EM algorithm (3), omitting the iteration counter (t) and the sample index i for notational economy, is then

$$w_m = \frac{\alpha_m \, p(\mathbf{y}_U|\boldsymbol{\theta}_{m,U})p(\mathbf{y}_N|\boldsymbol{\theta}_N)}{\sum_{j=1}^{k} \alpha_j \, p(\mathbf{y}_U|\boldsymbol{\theta}_{j,U})p(\mathbf{y}_N|\boldsymbol{\theta}_N)} = \frac{\alpha_m \, p(\mathbf{y}_U|\boldsymbol{\theta}_{m,U})}{\sum_{j=1}^{k} \alpha_j \, p(\mathbf{y}_U|\boldsymbol{\theta}_{j,U})}. \quad (7)$$

Recalling that $w_m = \text{Prob}[\mathbf{y} \in \text{class } m|\mathbf{y}, \boldsymbol{\theta}]$, we can read (7) as: given $\mathbf{y}_U$, the probability that an observation belongs to any class m is independent of $\mathbf{y}_N$. This reveals the link between the likelihood (4) and the irrelevance criterion (6), based on conditional independence.

3.3 A Feature Usefulness Measure for Unsupervised Learning

In practice, there are no strictly non-useful features, but features exhibiting some degree of "non-usefulness". A natural measure of the degree of independence, as suggested in [1], is the expected value of the *Kullback-Leibler divergence* (KLD,

or relative entropy [17]). The KLD between two probability mass functions $p_1(x)$ and $p_2(x)$, over a common (discrete) probability space Ω, is

$$\mathcal{D}_{KL}[p_1(x) \parallel p_2(x)] = \sum_{x \in \Omega} p_1(x) \log \frac{p_1(x)}{p_2(x)},$$

and satisfies $\mathcal{D}_{KL}[p_1(x) \parallel p_2(x)] \geq 0$, and $\mathcal{D}_{KL}[p_1(x) \parallel p_2(x)] = 0$, if and only if $p_1(x) = p_2(x)$, for all $x \in \Omega$. The relationship between conditional independence as stated in (6) and the KLD is given by the following implication

$$p(\mathbf{z}|\mathbf{y}_U, \mathbf{y}_N) = p(\mathbf{z}|\mathbf{y}_U) \quad \Rightarrow \quad \mathcal{D}_{KL}[p(\mathbf{z}|\mathbf{y}_U, \mathbf{y}_N)) \parallel p(\mathbf{z}|\mathbf{y}_U))] = 0, \tag{8}$$

for all values of $\mathbf{y}_U$ and $\mathbf{y}_N$. To obtain a measure of usefulness of a feature set, we have to average this measure over all possible feature values, according to their distribution [1]. In practice, both the KLD and its average over the feature space are approximated by their sample versions on the training samples.

In unsupervised learning we only have the feature samples $\mathcal{Y} = \{\mathbf{y}_1, ..., \mathbf{y}_n\}$, but no labels $\mathcal{Z} = \{\mathbf{z}_1, ..., \mathbf{z}_n\}$. However, after running the EM algorithm we have their expected values $\mathcal{W} = \{w_{i,m},\ m = 1, ..., k,\ i = 1, ..., n\}$. To build a sample-based feature usefulness measure, assume that $\mathcal{W}$ was obtained using the full feature set, and let $\widehat{\boldsymbol{\theta}}$ be the corresponding parameter vector. Now let $\mathcal{V}(N) = \{v_{i,m}(N),\ m = 1, ..., k,\ i = 1, ..., n\}$ be the expected label values obtained using only the features in the corresponding useful subset $U = \{1, ..., d\} \backslash N$, that is,

$$v_{i,m}(N) = \widehat{\alpha}_m\ p(\mathbf{y}_{i,U}|\widehat{\boldsymbol{\theta}}_{m,U}) \left(\sum_{j=1}^{k} \widehat{\alpha}_j\, p(\mathbf{y}_{i,U}|\widehat{\boldsymbol{\theta}}_{j,U}) \right)^{-1}. \tag{9}$$

Then, a natural measure of the "non-usefulness" of the features in N is

$$\Upsilon(N) = \frac{1}{n} \sum_{i=1}^{n} \sum_{m=1}^{k} w_{i,m} \log \frac{w_{i,m}}{v_{i,m}(N)}, \tag{10}$$

which is the sample mean of KLDs between the expected class labels obtained with and without the features in N. A low value of $\Upsilon(N)$ indicates that $\mathbf{y}_N$ is "almost" conditionally independent of the expected class labels, given $\mathbf{y}_U$.

4 A Sequential Backward Feature Sorting Algorithm

4.1 The Algorithm

Of course, evaluating $\Upsilon(N)$ for all 2^d possible subsets is unfeasible, even for moderate values of d. Instead, we propose a sequential backward search (SBS) scheme (Fig. 2) which starts with the full set of features set and removes them one by one in the order of irrelevance (according to the criterion (10)). This algorithm will produce an ordered set $I = \{i_1, ..., i_d\}$, which is a permutation of $\{1, 2, ..., d\}$ corresponding to a sorting of the features by increasing usefulness.

Input: Training data $\mathcal{Y} = \{\mathbf{y}^{(1)}, ..., \mathbf{y}^{(n)}\}$
Output: Set I of sorted feature indices.
  Initialization:
  $I \leftarrow \{\ \}$
  $U \leftarrow \{1, 2, ..., d\}$
  Run EM with all the features to get $\mathcal{W} = \{w_{i,m},\ m = 1, ..., k,\ i = 1, ..., n\}$
  while $|I| < d$ **do**
    $\Upsilon_{\min} \leftarrow +\infty$
    for $i \in U$ **do**
      $I'_i \leftarrow \{i\} \cup I$
      Compute $\Upsilon(I'_i)$ according to (10)
      if $\Upsilon(I'_i) < \Upsilon_{\min}$ **then**
        $\Upsilon_{\min} \leftarrow \Upsilon(I'_i)$
        $i_{\min} \leftarrow i$
      end if
    end for
    $I \leftarrow \{i_{\min}\} \cup I$
    $U \leftarrow U \backslash \{i_{\min}\}$
    Update $\mathcal{W}$ by running EM using only the features in U.
  end while

Fig. 2. Feature sorting algorithm. Notice that the sets used in the algorithm are ordered sets and the set union preserves that ordering (*e.g.*, $\{c\} \cup \{b, a\} = \{c, b, a\} \neq \{a, b, c\}$).

4.2 An Illustrative Example: Trunk's Data

To illustrate the algorithm, we use the problem suggested by Trunk [18]: two equiprobable d-variate Gaussian classes, with identity covariance and means $\boldsymbol{\mu}_1 = [1, 1/\sqrt{2}, 1/\sqrt{3}, ..., 1/\sqrt{d}]^T$ and $\boldsymbol{\mu}_2 = -\boldsymbol{\mu}_1$. Clearly, these features are already sorted in order of usefulness, and so any feature sorting scheme can be evaluated by how much it agrees with this ordering. In [3] (for supervised learning) a measure of the quality of the sorted set $I = \{i_1, ..., i_d\}$ was defined as

$$Q(I) = \frac{1}{d-1} \sum_{i=1}^{d-1} \frac{|I_1^i \cap \{1, ..., i\}| + |I_{i+1}^d \cap \{i+1, ..., d\}|}{d},$$

where $I_a^b = \{i_a, i_{a+1}, ..., i_b\}$. Note that $Q(I)$ is a measure of agreement between I and the optimal feature ordering $\{1, 2, ..., d\}$, with $Q(I) = 1$ meaning perfect agreement. Fig. 3 plots $Q(I)$ versus the sample set size, for $d = 20$, averaged over 5 data sets for each sample size. Remarkably, these values are extremely similar to those reported in [3], although here we are in an unsupervised learning scenario. Finally, the $\Upsilon_{\min}$ values are a measure of the relevance of each feature; in Fig. 3 we plot these values for the case of 500 samples per class.

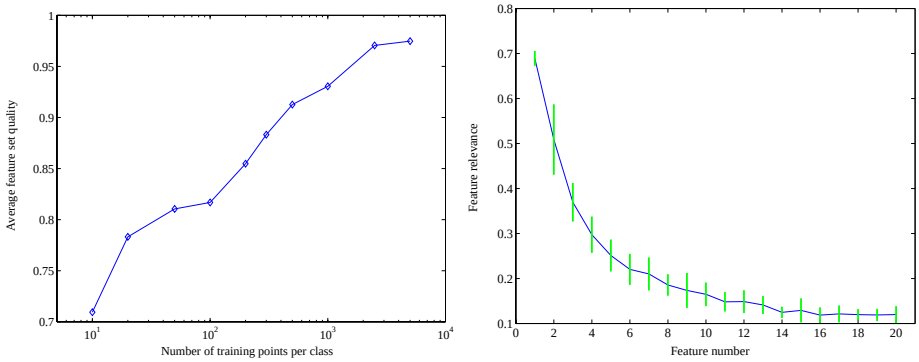

Fig. 3. Trunk data example. Left: feature sorting quality versus training set size. Right: feature relevance averaged over 20 data sets (vertical bars are ± 1 standard dev.).

5 Feature Selection by MDL

5.1 The Criterion

Having features sorted by order of relevance, we may now look for the best place to cut this sorted list, for a given data set. To this end, we return to the likelihood formulation (4), and to a comment made above: maximizing the likelihood leads to the selection of a full feature set. To avoid this over-fitting, we resort to the *minimum description length* (MDL) principle [19], criterion:

$$\widehat{U} = \arg\min_{U} \left\{ \min_{\boldsymbol{\theta}_U, \boldsymbol{\theta}_N} \{-\log p(\mathcal{Y}|U, \boldsymbol{\theta}_U, \boldsymbol{\theta}_N)\} + \frac{|\boldsymbol{\theta}_U| + |\boldsymbol{\theta}_N|}{2} \log(n) \right\}, \qquad (11)$$

where $\log p(\mathcal{Y}|U, \boldsymbol{\theta}_U, \boldsymbol{\theta}_N)$ is given in (5) and $|\boldsymbol{\theta}_U|$ and $|\boldsymbol{\theta}_N|$ are the total numbers of parameters in $\boldsymbol{\theta}_U$ and $\boldsymbol{\theta}_N$, respectively. Notice that the inner minimization simply corresponds to the ML estimate of $\boldsymbol{\theta}_U$ and $\boldsymbol{\theta}_N$ for a given U, obtained by the EM algorithm for $\boldsymbol{\theta}_U$ and by simple maximum likelihood estimates in the case of $\boldsymbol{\theta}_N$. The numbers of parameters $|\boldsymbol{\theta}_U|$ and $|\boldsymbol{\theta}_U|$ depend on the particular form of $p(\mathbf{y}_N|\boldsymbol{\theta}_N)$ and $p(\mathbf{y}_U|\boldsymbol{\theta}_{m,U})$. For example, with Gaussian mixtures with arbitrary mean and covariance, $|\boldsymbol{\theta}_U| = k(3u + u^2)/2$. With $p(\mathbf{y}_N|\boldsymbol{\theta}_N)$ also a Gaussian density with arbitrary mean and covariance, $|\boldsymbol{\theta}_N| = (3(d-u) + (d-u)^2)/2$.

This MDL criterion is used to select which features to keep, by searching for the solution of (11) among the following set of candidate subsets, produced by the feature sorting algorithm of Fig. 2: $\{I_1^q = \{i_1, ..., i_q\};\ q = 1, ..., d\}$.

5.2 Illustrative Example

We illustrate the behavior of the feature selection algorithm with a simple synthetic example. Consider a three-component mixture in 8 dimensions with component means $\boldsymbol{\mu}_1 = [3, 0, 0, 0, ..., 0]^T$,$\boldsymbol{\mu}_2 = [0, 3, 0, 0, ..., 0]^T$, $\boldsymbol{\mu}_3 = [0, 0, 3, 0, ..., 0]^T$,

and identity covariance matrices. Clearly, only the first three features are relevant to the mixture, features 4 to 8 are simply "noise". We have applied the feature sorting algorithm described above to 40 sets of 450 samples of this mixture (150 per class) and features 1, 2, and 3 were always placed ahead of the others in the sorted feature list. Next, we used the criterion in (11) to select the "optimal" number of features, and three features were always selected; the left plot in Fig. 4 shows the mean description length curve for the 40 tests, with ± one standard deviation bars. Since we have the true class labels for this data, we have computed error rates, which are plotted on the right side of Fig. 4 (again, mean over 40 test, ± one standard deviation bars); notice that the minimum error rate is achieved for the true number of relevant features; observe also that with too few or too many features, the obtained classifier becomes more instable (larger error bars).

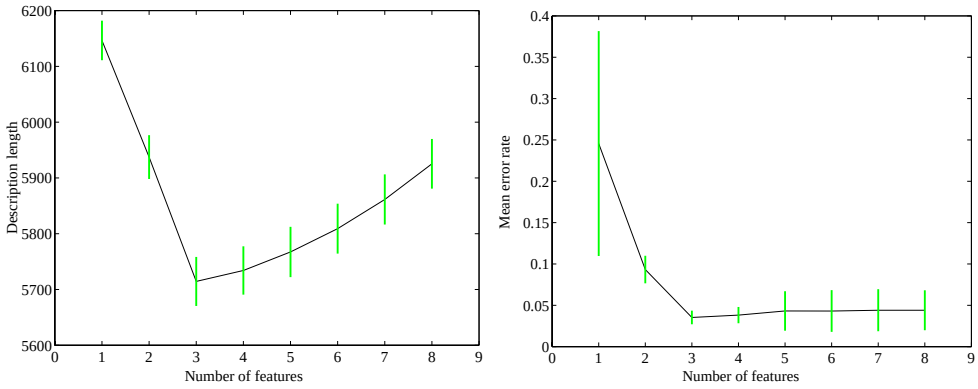

Fig. 4. MDL-based feature selection example. Left: description length, (11), as a function of the number of "useful" features (mean curve for 40 tests, ± one standard deviation). Right plot: mean error rate (also for 40 tests, ± one standard deviation).

6 Concluding Remarks

We have presented an approach to feature sorting and selection for mixture-based clustering. Tests on synthetic data show that the method is promising. Of course the method has yet to be extensively tested on real data, but assessing the quality of a feature selection method for unsupervised learning with real data is not an obvious task. Future developments will include extending the method to also estimate the number of clusters, by wrapping the feature selection procedure around a mixture-fitting algorithm that estimates the number of components (such as the one in [9]). Also, searching strategies other than backward search (*e.g.*, floating search [3]) will be considered in future work.

References

1. D. Koller and M. Sahami, "Toward optimal feature selection," in *Proceedings of the International Conference on Machine Learning*, (Bari, Italy), pp. 284–292, 1996.
2. R. Kohavi and G. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997.
3. A. K. Jain and D. Zongker, "Feature selection: Evaluation, application, and small sample performance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 2, pp. 153–158, 1997.
4. T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. New York: Springer Verlag, 2001.
5. A. K. Jain and R. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, N. J.: Prentice Hall, 1988.
6. C. Fraley and A. Raftery, "Model based clustering, discriminant analysis, and density estimation," *Journal of the American Statist. Assoc.*, vol. 97, pp. 611–631, 2002.
7. G. McLachlan and K. Basford, *Mixture Models: Inference and Application to Clustering*. New York: Marcel Dekker, 1988.
8. G. McLachlan and D. Peel, *Finite Mixture Models*. New York: John Wiley & Sons, 2000.
9. M. Figueiredo and A. K. Jain, "Unsupervised learning of finite mixture models," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 381–396, 2002.
10. J. Dy and C. Brodley, "Feature subset selection and order identification for unsupervised learning," in *Proc. 17th International Conf. on Machine Learning*, pp. 247–254, Morgan Kaufmann, San Francisco, CA, 2000.
11. S. Vaithyanathan and B. Dom, "Generalized model selection for unsupervised learning in high dimensions," in *Advances in Neural Information Processing Systems 12* (S. Solla, T. Leen, and K.-R. Müller, eds.), MIT Press, 2000.
12. M. Dash and H. Liu, "Feature selection for clustering," in *Proc. of Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2000.
13. Y. Kim, W. Street, and F. Menczer, "Feature Selection in Unsupervised Learning via Evolutionary Search," in *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000.
14. M. Devaney and A. Ram, "Efficient feature selection in conceptual clustering," in *International Conference on Machine Learning*, pp. 92–97, 1997.
15. A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood estimation from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society B*, vol. 39, pp. 1–38, 1977.
16. G. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*. New York: John Wiley & Sons, 1997.
17. T. Cover and J. Thomas, *Elements of Information Theory*. New York: John Wiley & Sons, 1991.
18. G. Trunk, "A problem of dimensionality: A simple example," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 1, no. 3, pp. 306–307, 1979.
19. J. Rissanen, *Stochastic Complexity in Stastistical Inquiry*. Singapore: World Scientific, 1989.