1-1-2010

Localized Feature Selection For Unsupervised Learning

Yuanhong Li
Wayne State University,

LOCALIZED FEATURE SELECTION FOR UNSUPERVISED LEARNING

by

YUANHONG LI

DISSERTATION

Submitted to the Graduate School

of Wayne State University,

Detroit, Michigan

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

2010

MAJOR: COMPUTER SCIENCE

Approved by:

Advisor Date

ACKNOWLEDGMENTS

First and foremost, I want to express my greatest appreciation to my supervisor, Dr. Ming Dong. Under his guidance, I have learned a lot in different aspects of conducting research, including finding a good research topic and writing convincing technical paper. It is his guidance, support and tremendous help that made this dissertation possible. I am also very thankful to the rest of my thesis committee, including Dr. Jing hua, Dr. Hao Ying, and Dr. Chandan Reddy. Their advice and suggestions have been very helpful.

I want to thank Dr. Manjeet Rege, Dr. Changbo Yang, Dr. Xuanwen Luo, Yanhua Chen and Lijun Wang for their enthusiastic help during my Ph.D. study, which I shall never forget.

Let me express my special thanks to my wife, Dr. Hua Gu. Whithout her support and love, I could not complete my doctoral degree.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Advances in computer technology have led to the information age, which some people refer to as “data explosion”. The amount of data available to any person is increased so much that it is more than he or she can handle. This increase in both the volume and variety of data calls for advance methodology of understanding, processing and summarizing the data. In my dissertation, we focus on two important techniques for data analysis in pattern recognition: clustering and feature selection.

1.1 Data Representation

In pattern recognition perspective, data is the description of a set of objects or patterns that can be processed by a computer. The patterns are supposed to have some commonalities, such that the same systematic procedure can be applied to all the objects to generate the description. Data can be represented in many ways. Most often, an object is described by a vector of measurement results of its various properties. A measurement result is called a “feature” in pattern recognition, or a “variable” in statistics. Data matrix of size n by d is formed by arranging the feature vectors of different objects in different rows, where n is the number of patterns and d the number of features. If all the features are numerical, the data can be represented as a point in space $\mathbb{R}^d$, which enables a number of mathematical tools to be used to analyze the objects.

1.2 Categories of Machine Learning Methods

In pattern recognition, most of the analysis concerned with predictive modeling, i.e., predicting the behavior of the unseen data (testing data) based on the existing data (training data). Depending on the feedback one can receive in the learning process, machine learning methods can be categorized into three groups: supervised, unsupervised (clustering), and semi-supervised learning. In supervised learning, labels of the training data are available to verify if the predict is correct or not. In unsupervised learning, such label information is missing. In semi-supervised learning, only some of the data points are labeled. This happens frequently in practice, since data collection and feature extraction can be done automatically, whereas the labeling has to be done manually which is often expensive. In unsupervised learning, no label information is available. The target of machine learning task in this scenario is to discover the natural grouping structure of the data. This is very important in many practical applications, for example, to find different groups of credit card holders and to learn their general behaviors from a huge dataset collected by a credit card provider.

1.3 Dimensionality Reduction

Dimensionality reduction deals with the transformation of high dimensional to low dimensional representation. The underlying assumption is that the data points can be exploited in a certain structure, and the information of the structure can be summarized by a small number of attributes. Intuitively, the more information we have, the better a learning algorithm is expected to perform. This seemingly suggests that we use all the features for the learning task. However, this is not the case in practice. Most learning algorithms perform poorly in high dimensional space with a small number of samples. This difficulty is known as the *curse of dimensionality*. Additionally, datasets often come with noise features which do not contribute to the learning process. Dimensionality reduction yields simple representation of datasets. This can enhance the generalization capability of the output model, reduce the computation time for learning, and shrink the space occupied by the output model. The low dimensional model is also easier for domain experts to interpret, and make it possible to display visually by transforming it into two or three dimensions.

The main drawback of dimensionality reduction is the possibility of information loss. Useful information can be discarded if dimensionality reduction is done poorly.

In general, dimensional reduction methods can be categorized into two groups: feature extraction and feature selection.

1.3.1 Feature Extraction

In feature extraction, a small set of new features is constructed by a general mapping from the high dimensional data. The mapping often isolate the available features. The mapping can be linear, i.e., Principal Component Analysis (PCA) [2], Linear discriminant analysis (LDA), and multiple discriminant analysis (MDA), or non-linear, i.e., Kernel PCA [3], ISOmap [4], and Locally Linear Embedding (LLE) [5].

1.3.2 Feature Selection

Feature selection selects a subset of features that is most appropriate for the task at hand. A feature is either selected or discarded. This constraint can be relax by assigning weights to different features to indicate the saliencies of the individual features. This is also referred to as feature weighting, or feature ranking. The feature selection problem can be formulated as

$$T^{opt} = \arg\max_{T \in S} Q(T) \tag{1.1}$$

where T^{opt} is the optimal feature subset, S is the full set of subsets, and $Q(\cdot)$ is the quality function.

The new features generated by feature extraction algorithms are hard to interpret in practice due to the linear or non-linear transformation. Feature selection, on the other hand, selects a subset of the original features by removing most irrelevant and redundant features from the data and help people to better understand their data by telling them which are the important features and how they are related to each other. The new low-dimensional data set are meaningful and easy to interpret.

1.4 General Procedure of Feature Selection

A typical feature selection algorithm consists of four basic steps as shown in Figure 1.1, namely, subset generation, subset evaluation, stop criterion and result validation.

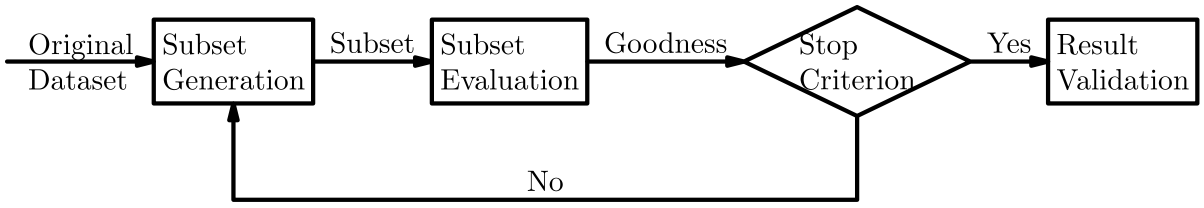


Figure 1.1: General procedure of feature selection.

1.4.1 Subset Generation

Subset generation is the procedure to create the next candidate feature subset for evaluation. The nature of this process is determined by two issues; search starting point and search strategy. The process can start with empty subset; the full set of features; or a random subset to avoid local optimization. For a dataset with D features, there are 2^D possible candidate subsets, which exponentially increases with the number of features. Heuristic search methods are usually applied, such as sequential search, random search, complete search, and integral search.

Sequential Search. This strategy usually employs the greedy hill-climbing method to generate feature subset. For example, sequential forward selection, sequential backward elimination, and bidirectional search [6]. These algorithms add or remove one feature at a time. Another approach is to add or remove p features at a time [7]. Sequential search algorithms avoid navigation over all the subset candidates, thus speed up the feature selection procedure. However, they may risk losing optimal subset.

Complete Search. The complete search guarantees to find the optimal subset. Though its complexity is $O(2^D)$, it does not imply that an exhaustive search is necessary. Typical algorithms include branch and bound search [8], and beam search [7].

Random Search. The random search can be started with a randomly selected subset, then by adding or removing features by sequential search [7]. It also can be selecting another totally random subset for the next evaluation [9]. Simulated annealing [10] and genetic algorithms [11, 12] also belong to this category.

Integrated Search. This strategy does not generate feature subset explicitly. Instead, it introduce quantity of feature importance, namely feature saliency, to achieve the goal of feature subset generation [1, 13].

1.4.2 Subset Evaluation

The candidate feature subsets need to be evaluated by some criteria so that the best feature subset can be determined according to the goodness measure. The evaluation criteria can be roughly categorized into two groups: independent criteria and dependent criteria.

Independent Criteria. An independent criterion is typically used in filter algorithm. It tries to measure the intrinsic characteristics of the dataset without involving any mining algorithm. Some popular criteria are separability measures, information measures, and dependency measures [14–18].

Dependent Criteria. A dependent criterion is used by wrapper models. The criterion is measured with a specific mining algorithm. The performance of the mining algorithm is applied to determine the goodness of the feature subset. Usually, a dependent criterion yields better performance than an independent criterion for the predefined mining algorithm. However, the selected feature subset may not be suitable for other mining algorithms, and the computational cost is often expensive. For classification problems, the predicting accuracy of unseen instances is widely used to select feature subset which yields high testing accuracy [19, 20]. For clustering problems, a wrapper model evaluates the goodness of a feature subset by the quality of the clusters obtained by a specific clustering algorithm. Cluster compactness, scatter separability, and maximum likelihood are some typical cluster goodness measures used for feature selection. Readers can refer to [1, 13, 15, 21–23] for recent development of dependent criterion for unsupervised feature selection.

1.4.3 Stopping Criteria

The feature selection process terminates when a stopping criterion is achieved. Some frequently used stopping criteria are as follows:

- The search is completed.
- Subsequent addition or deletion of any feature does not yield better result.
- A sufficiently good subset is selected.
- Some given bound, i.e. the number of iterations or the number of selected features, is reached.

1.4.4 Result Validation

The prior knowledge of the underlying dataset is often used to directly validate the result of a feature selection process. For a synthetic dataset, the relevant feature subset and irrelevant feature subset is usually known. The former is expected to appear in the resulting feature subset, while the later is not. Thus we can validate the results by comparing the known relevant and/or irrelevant features with the feature subset produced by the feature selection algorithm. However, in real world applications, such a prior knowledge is usually unknown. Validation of results must occur in an indirect way. A frequently used method is to conduct experiments not only on the selected feature subset, but also the whole feature set. The resulted validation is achieved by comparing the performance of these before-and-after feature selection experiments.

1.5 Categories of Feature Selection Algorithms

There are many feature selection algorithms developed in the literature. They can be categorized into different groups according to the subset generation methods, the subset evaluation methods, or data mining tasks. Under subset generation methods, the feature selection algorithms can be categorized into four groups: complete search, sequential search, random search, and integral weighting. Under subset evaluation criteria, they can be categorized into three groups: filters, wrappers, and hybrids. Under data mining task criteria, they can be categorized into two groups: supervised learning and unsupervised learning. Considering the scope of the selected feature subset, they can be categorized into two groups: global feature selection and localized feature selection. We will discuss the three general categories corresponding to the subset evaluation criteria, and the two categories corresponding to the feature scope in this section.

1.5.1 Filter Approach

For a given dataset, a filter algorithm [16, 17, 24] starts from a initial feature subset, and navigate the feature space by a particular search strategy. Each generated subset is evaluated by a measure which is independent to mining algorithm. The search iterations continue until some stopping criteria are reached. The best subset is then returned.

A filter approach does not involving any data mining algorithm; thus it does not inherit any bias of the mining algorithm. Any mining algorithm can be used sequentially to analyze the dataset. However, given a particular mining algorithm, the selected feature subset may not be optimal.

1.5.2 Wrapper Approach

A wrapper approach is similar to the filter approach except that it utilizes a predefined mining algorithm to evaluate the generated feature subset [1, 21, 23, 25]. Since the goodness of the feature subset is controlled by the mining algorithm, the performance of a wrapper method is superior, and different mining algorithms will produce different feature subsets. The computation cost is usually higher than a filter method.

1.5.3 Hybrid Approach

A hybrid approach [26] utilizes a independent measure to preselect a feature subset. A mining algorithm is used to finally decide the output feature subset.

1.6 Localized Feature Selection for Clustering

Feature selection has been extensively studied in supervised learning scenarios [18, 19, 27–30]. In unsupervised learning, feature selection becomes a more complex problem due to

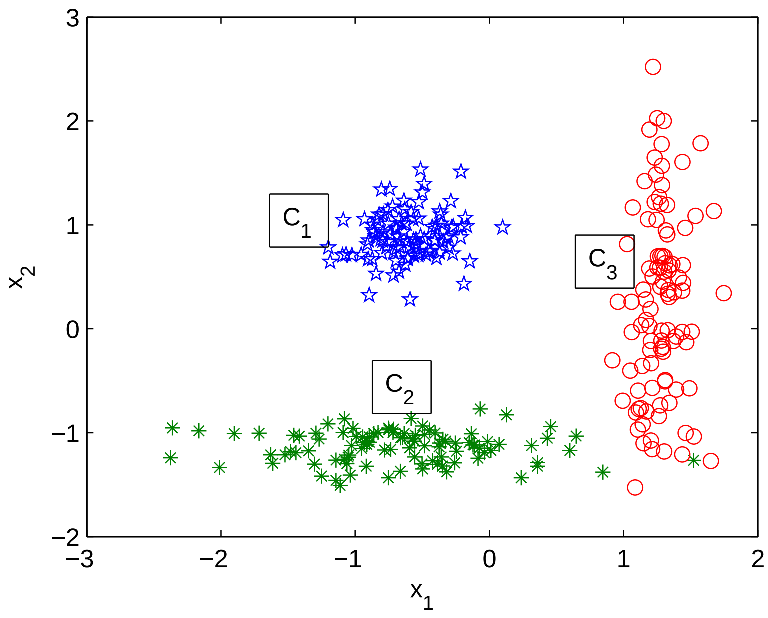


Figure 1.2: A three-cluster dataset with cluster C_1 embedded in feature set $\{x_1, x_2\}$, cluster C_2 embedded in feature subset $\{x_2\}$, and cluster C_3 embedded in feature subset $\{x_1\}$.

the unavailability of class labels. It is received research intention only recent. This research dissertation will focus on feature selection problem for unsupervised learning tasks.

In general, unsupervised feature selection algorithms conduct feature selection in a *global* sense by producing a common feature subset for all the clusters. This, however, can be invalid in clustering practice, where the local intrinsic property of data matters more. In the illustrative example shown in Figure 1.2, the relevant feature subset for cluster C_1 is $\{x_1, x_2\}$, while clusters C_2 and C_3 can be grouped using $\{x_2\}$ and $\{x_1\}$, respectively. A common feature subset, i.e., $\{x_1, x_2\}$, is unable to reflect the inherent structural properties of the three clusters. Apparently, clustering with *local* features is highly desired.

In general, there are two major research trends that select features locally for clusters, namely, co-clustering and subspace clustering.

1.6.1 Co-clustering

In a co-clustering problem, data is stored in contingency or co-occurrence matrix C. The co-clustering process derives sub-matrices from the large data matrix by simultaneously clustering rows and columns of the data matrix. Optimal co-clustering is derived based on the one that leads to the largest mutual information between the clustered random variables [31]. A well studied problem of co-clustering in data mining has been that of documents and words. The goal is to cluster documents based on the common words that appear in them and to cluster words based on the common documents that they appear in [32–36]. Co-clustering algorithms attempt to partition the features exclusively. That means a feature can only belong to a particular cluster. This property limits its application in general feature selection for clustering problems.

1.6.2 Subspace Clustering

Subspace clustering [37] is another extension of traditional clustering that seeks clusters in different subspaces within a dataset. Subspace clustering algorithms localize the search for relevant features such that clusters which exist in multiple, possibly overlapping subspaces are determined. Subspace clustering approaches usually search for possible feature subsets on which density regions may occur, then clusters are discovered in the different subspaces.

1.7 Overview

In this dissertation, we focus on the problem of localized feature selection for unsupervised learning. The rest of the thesis is organized as follows: In Chapter 2, we review related works in the literature. In Chapter 3, we propose an algorithm of localized feature selection for unsupervised learning by cross-projection method. In Chapter 4, a probabilistic model of feature saliency with Gaussian mixture is addressed. The feature selection with model detection is integrated into Maximal Likelihood (ML) learning scenario. We propose another algorithm which performs clustering, feature selection, and cluster number detection simultaneously with Variational Learning in Chapter 5. The conclusions of this thesis and recommendations for future work are addressed in Chapter 6.

CHAPTER 2

RELATED WORK IN UNSUPERVISED FEATURE SELECTION

In Chapter 1, we described the importance of feature selection and presented an overall picture of different approaches for feature selection. This chapter continues the discussion of unsupervised feature selection. We shall survey some of the recent feature selection algorithms. Since we are mostly interested in unsupervised learning, supervised feature selection algorithms are omitted from this survey. We organize the algorithms based on the scope of the feature subset (Global/Local), and the type of evaluation criteria (Filter/Wrapper).

2.1 Global Feature Selection

Feature selection algorithms generally process all clusters in a common subset. In other words, an irrelevant feature f_m is irrelevant to all clusters, and a relevant feature f_n implies that it is relevant to all clusters. The feature selection algorithm does not distinguish the different response of a specified feature on different clusters. The output model is simple and straightforward.

2.1.1 Filters

A filter approach evaluates the quality of a feature subset without involving a particular clustering algorithm. It usually adopts a independent criterion, such as the feature similarity measure, or information measure, and finds the best subset through a search strategy.

The most well-known measure of similarity between two random variables x and y is the correlation coefficient, which is defined as

$$\rho(x,y) = \frac{\texttt{cov}(x,y)}{\sqrt{\texttt{var}(x)\texttt{var}(y)}} \tag{2.1}$$

where $\texttt{var}(\cdot))$ denotes the variance of a variable and $\texttt{cov}(\cdot)$ the covariance between two variables. If x and y are completely correlated, i.e., exact linear dependency exist, $\rho(x,y)$ is 1 or -1. If x and y are totally uncorrelated, $\rho(x,y)$ is 0. Hence, $1-|\rho(x,y)|$ can be used as measure of similarity between two variables. This measure is used as a criterion in [17,38]. The reduced subset is obtained by discarding correlated features with a stepwise clustering scheme.

Correlation coefficient is invariant to scaling and sensitive to rotation, which are not desirable in many feature selection cases. Mitra et al. [17] suggest another linear dependency measure, *Maximal Information Compression Index (MIC)* (λ_2), for feature selection. MIC is defined as follows

$$2\lambda_2(x,y) = \mathtt{var}(x) + \mathtt{var}(y) - \sqrt{\big(\mathtt{var}(x) + \mathtt{var}(y)\big)^2 - 4\mathtt{var}(x)\mathtt{var}(y)\big(1-\rho(x,y)^2\big)} \tag{2.2}$$

The value of λ_2 is zero when the features are linearly dependent and increases as the amount of dependency decreases. Actually, λ_2 is equal to the eigenvalue for the direction normal to the principal component direction of feature pair (x,y). It is also equal to the sum of the squares of the perpendicular distances of the points (x,y) to the best fit line $y = \hat{a} + \hat{b}x$ [39]. Based on the feature similarity measure, the correlated features can be removed by some particular search strategy, such as Branch and Bound Search [40], Sequential Forward Search [40], Sequential Floating Forward Search [41], Stepwise Clustering [38]. In [17], features are partitioned into a number of homogeneous subsets based on the k-nearest-neighbor (KNN) principle using MIC. Among them the features having the most compact subset is selected, and its k neighboring features are discarded. The best feature subset is generated by repeating this process until all of the features are either selected or discarded.

The above feature similarity measures are efficient to detect correlated features. However, they cannot detect irrelevant features. To overcome this issue, Dash et al. [26] proposed a distance-based entropy measure, which is defined as,

$$E = -\sum_{X_i}\sum_{X_j}\big[D_{ij}\log D_{ij} + (1 - D_{ij})\log(1 - D_{ij})\big] \tag{2.3}$$

where D_{ij} is the normalized distance in the range $[0.0,\ 1.0]$. This method assigns a low entropy to intra- and inter-cluster distances, and assigns a higher entropy to noisy distances. This measure suffers from two drawbacks. (a) The mean distance of 0.5, the meeting point (μ) of the left and right side of the entropy plot can be an inter-cluster distance, but still it is assigned the highest entropy. (b) Entropy increases rapidly for very small distances thus assigning very different entropy values for intra-cluster distances. An improved version is proposed in [16] as follows,

$$E = \sum_{X_i}\sum_{X_j} E_{ij} \tag{2.4}$$

$$E_{ij} = \begin{cases} \frac{\exp(\beta * D_{ij})\,\exp(0)}{\exp(\beta * \mu)\,\exp(0)} & 0 \le D_{ij} \le \mu \\ \frac{\exp(\beta * (1.0\, D_{ij})) * \exp(0)}{\exp(\beta * (1.0\, \mu))\,\exp(0)} & \mu \le D_{ij} \le 1.0 \end{cases} \tag{2.5}$$

where E_{ij} is normalized to the range $[0.0,\ 1.0]$. The parameter β, which is set based on the domain knowledge, controls the entropy contribution of between intra- and inter-distances. The parameter μ, which is updated heuristically, shifts the meeting point of the two sides of the entropy-distance plot. The entropy of a particular feature is calculated by removing it from the original feature set and computing the entropy change thereby. Features are ranked based on their entropy. Best feature subset is obtained by selecting the top ranked features.

2.1.2 Wrappers

Filter feature selection approaches can be used by any clustering algorithms. However, the output is often not optimized for a particular clustering algorithm. On the other hand, a wrapper approach utilizes a particular clustering algorithm to evaluate the performance of feature subsets, thus usually produces better feature subset than a filter does. Most unsupervised feature selection algorithms are wrappers. In this section, we review some wrapper approaches proposed very recent.

2.1.2.1 Cross-Projection

The quality of clusters can be measured by the within-cluster scatter matrix (S_w) and the between-cluster scatter matrix (S_b),

$$S_w = \sum_{j=1}^{k} \pi_j E\{(X-\mu_j)(X-\mu_j)^T|\omega_j\} = \sum_{j=1}^{k} \pi_j \Sigma_j \tag{2.6}$$

$$S_b = \sum_{j=1}^{k} \pi_j (\mu_j - M_o)(\mu_j - M_o)^T, \tag{2.7}$$

$$M_o = E\{X\} = \sum_{j=1}^{k} \pi_j \mu_j \tag{2.8}$$

where π_j is the probability that an instance belongs to cluster ω_j, X is a d-dimensional random feature vector representing the data, k the number of clusters, μ_j is the sample mean vector of cluster ω_j, M_o is the total sample mean, Σ_j is the sample covariance matrix of cluster ω_j, and $E\{\cdot\}$ is the expected value operator. Many separability measures can be obtained based on scatter matrix [42]. Among them, $trace(S_w^{-1}S_b)$ is widely used in literature [43]. However, this criterion is biased on dimensionality, which means that the measure monotonically increases with dimension, assuming the clustering assignments remain the same. In order to elevate this bias, Dy et al. [21] proposed a cross-projection method. Given two feature subsets S_1 and S_2, the clustering results are C_1 and C_2, respectively. Let $CRIT(S_i, C_j)$ be the clustering criteria using feature subset S_i to represent the data and C_j as the clustering assignment. The criteria values for (S_1, C_1) and (S_2, C_2) are normalized as,

$$normalizedValue(S_1, C_1) = CRIT(S_1, C_1) \times CRIT(S_2, C_1) \tag{2.9}$$

$$normalizedValue(S_2, C_2) = CRIT(S_2, C_2) \times CRIT(S_1, C_2) \tag{2.10}$$

This cross-projection method ensures that the bias of dimensionality is removed, thus it can be used to compare the clustering quality on two different feature subsets, even though they may have different dimension. In [21], sequential forward search method is used to navigate through possible subset candidates. The number of clusters is estimated by merging clusters one at a time and using a Bayesian Information Criterion (BIC).

2.1.2.2 Law's E-M Approach

Traditional feature selection algorithms have to search through the possible candidate subsets, which demands heavy computational load, even by greedy search methods. Law et al. [1] proposed another approach, which selects salient features and estimates the number of clusters simultaneously by Expectation Maximization (EM) algorithm. Assuming that the features are independent given a mixture component, and following a common distribution up to a probability, the complement of this probability is defined as feature saliency and estimated by the Maximum Likelihood (ML) or Maximum A priori (MAP) with EM algorithm using Gaussian mixture models. The likelihood of such model is defined as follows,

$$p(y|\theta) = \sum_{j=1}^{k} \alpha_j \prod_{l=1}^{d} \big(\rho_l p(y_l|\theta_{jl}) + (1-\rho_l) q(y_l|\lambda_l)\big) \tag{2.11}$$

where $p(\cdot)$ represents a probability distribution of a component, $q(\cdot)$ representing the common distribution, θ_{jl} and λ_l denoting the parameters, ρ_l indicating the saliency of the particular feature, and $\theta = \big\{\{\alpha_j\}, \{\theta_{jl}\}, \{\lambda_l\}, \{\rho_l\}\big\}$. The model selection (estimating the number of clusters) can be accomplished based on minimum message length (MML) criterion [44, 45]. The algorithm tries to minimize the following cost function,

$$-\log p(\mathcal{Y}|\theta) + \frac{k+d}{2}\log n + \frac{r}{2}\sum_{l=1}^{d}\sum_{j=1}^{k}\log(n\alpha_j\rho_j) + \frac{s}{2}\sum \log(n(1-\rho_l)), \tag{2.12}$$

where r and s are the number of parameters in θ_{jl} and λ_l, respectively. This cost function is minimized through EM method. The advantage of this approach is that it accomplishes feature selection with clustering simultaneously, and avoids the navigation over the possible feature subset candidates which is usually very large.

2.1.2.3 Variational Approach

[13] and [46] employ the same Gaussian mixture model as in [1] to describe feature relevance, but integrate model and feature selection under Bayesian framework. The model parameters follow particular distributions instead of fixed values as estimated by EM algorithm. The learning process is to fit the distributions based on the given dataset. [13, 46] utilize variational learning techniques to estimate the underlying model. Since the cluster number also follows a distribution, it can be conducted directly.

2.2 Localized Feature Selection

Feature selection algorithms aforementioned are global, which means that the feature subset selected is common to all the clusters. However, in many applications, the natural grouping structure of a cluster is localized in a particular subspace, which implies that different clusters may have different relevant feature subset. The output format of such an algorithm is $\{C_k, F_k\}$, where C_k and F_k indicate the cluster assignment and feature subset for a specific cluster k. Notice that clustering results is required by those algorithms, thus localized feature selection approaches are wrappers. Co-clustering and subspace clustering are two categories in this research area.

2.2.1 Co-clustering

Co-clustering (also called Biclustering, Bipartite, or two-mode clustering), is simultaneous clustering of both instances and features such that the blocks induced by the row/column partitions are good clusters.

2.2.1.1 Information-Theoretic Co-Clustering

Let X and Y be discrete random variables that take values in the sets $\{x_1, \ldots, x_m\}$ and $\{y_1, \ldots, y_n\}$ respectively. Let $p(X, Y)$ denote the joint probability distribution between X and

Y. Co-clustering tries to find maps C_X and C_Y,

$$C_X : \{x_1, x_2, \ldots, x_m\} \rightarrow \{\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_k\} \tag{2.13}$$

$$C_Y : \{y_1, y_2, \ldots, y_n\} \rightarrow \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_l\} \tag{2.14}$$

which minimizes the following criterion,

$$I(X;Y) - I(\hat{X};\hat{Y}) \tag{2.15}$$

where $I(X;Y)$ is the mutual information between X and Y. Dhillon et al. [31] address that the loss in mutual information can be expressed as,

$$\begin{aligned} I(X;Y) - I(\hat{X};\hat{Y}) &= D\left(p(X,Y,\hat{X},\hat{Y}) \| q(X,Y,\hat{X},\hat{Y})\right) \\ &= \sum_{\hat{X}} \sum_{x: C_X = \hat{x}} p(x) D\left(p(Y|x) \| q(Y|\hat{x})\right) \qquad (2.16) \\ &= \sum_{\hat{Y}} \sum_{y: C_Y = \hat{y}} p(y) D\left(p(X|y) \| q(X|\hat{y})\right) \qquad (2.17) \end{aligned}$$

where $D(\cdot\|\cdot)$ denotes the Kullback-Leibler (KL) divergence, and $q(X,Y,\hat{X},\hat{Y})$ is a distribution of the form:

$$q(x,y,\hat{x},\hat{y}) = p(\hat{x},\hat{y}) p(x|\hat{x}) p(y|\hat{y}). \tag{2.18}$$

Thus the cost function can be minimized by alternatively improving row clusters (Equation (2.16)) and column clusters (Equation (2.17)). Similar models can be found in [47, 48].

2.2.1.2 Graphic Theoretic Co-clustering

Given an undirected bipartite graph $G = (M, R, E)$, where M and R are two sets of vertices, and E a set of edges, let B be the graph weight matrix. An entry B_{ij} in this matrix is the weight of an edge appearing between a vertex $r_i \in R$ and a vertex $m_j \in M$. There are no edges between vertices of the same group. The adjacency matrix of the bipartite graph is expressed as,

$$M = \begin{bmatrix} 0 & B \\ B^T & 0 \end{bmatrix} \tag{2.19}$$

The bipartite Laplacian matrix is defined as,

$$L = \begin{bmatrix} D_R & -B \\ -B^T & D_M \end{bmatrix} \tag{2.20}$$

where $D_R(i,i) = \sum_j B_{ij}$ and $D_M(j,j) = \sum_i B_{ij}$. Co-clustering of the data is achieved by partitioning the bipartite graph into two subset V_1 and V_2. Shi and Malik applied spectral graph partitioning to the problem of image segmentation in [49] by minimizing the objective function,

$$\min \frac{x^T L x}{x^T D x} \tag{2.21}$$

where x is a column vector such that $x_i = c_1$ if $i \in V_1$ and $x_i = -c_2$ if $i \in V_2$. By relaxing x_i from discrete to continuous, it can be shown that the solution to (2.21) is the eigenvector corresponding to the second smallest eigenvalue of the generalized eigenvalue problem [50,51],

$$Lx = \lambda D x \tag{2.22}$$

This eigenvalue problem can be reduced to a much more efficient Singular Value Decomposition (SVD) [51] problem. Dhillon [52] and Zha et al., [53] employed this Spectral-SVD approach to partition a bipartite graph of documents and words. Ding [54] performed document-word co-clustering by extending Hopfield networks [55][58] to partition bipartite graphs and showed that the solution is the principal component analysis (PCA) [2].

Co-clustering has been found to have applications in document and multimedia grouping problems. However, in feature selection prospect, the feature subsets associated to different clusters are disjointed, which implies that a feature cannot be selected for several clusters. This restriction is often inflicted in general feature selection problems.

2.2.2 Subspace Clustering

Subspace clustering algorithms search for relevant features locally to find clusters that exist in multiple, possibly overlapping subspaces. There are two major branches of subspace clustering based on their search strategy. Bottom-up approaches find dense regions in low dimensional spaces and combine them to form clusters. Typical algorithms in this branch are CLIQUE, ENCLUS, MAFIA, Cell Based Clustering (CBF), CLTree, DOC, and SURFAING. Top-down algorithms find an initial clustering in the full set of dimensions and evaluate the subspaces of each cluster, iteratively improving the results. Typical top-down algorithms are COSA, PROCLUS, ORCLUS, and FINDIT.

2.2.2.1 CLIQUE

CLIQUE [56] combines density and grid based clustering to find low dimensional clusters embedded in high dimensional space. Each dimension is divided into bins using a static sized grid. Dense subspaces are sorted by coverage. The subspaces with the greatest coverage are kept and the rest are pruned. Adjacent dense grid units are discovered in each selected subspace using a depth first search. Clusters are formed by combining these units using a greedy growth scheme. The hyper-rectangular clusters are then defined by a Disjunctive Normal Form (DNF) expression. Clusters may be found in the same, overlapping or disjoint subspaces. The clusters may also overlap each other. CLIQUE requires grid size and density threshold as input parameters. Tuning these parameters can be difficult.

2.2.2.2 ENCLUS

ENCLUS [57] is another subspace clustering method based heavily on the CLIQUE algorithm. The algorithm is based on the observation that a subspace with clusters typically has low entropy than a subspace without clusters. Thus ENCLUS computes the entropy measure rather than density and coverage (used in CLIQUE) to determine the clusterability of a subspace. ENCLUS also introduces *interest*, which is defined as the difference between the sum of entropy of measurements for a set of dimensions and the entropy of multi-dimension distribution, to measure the correlation of a subspace. Large values indicate higher correlation between dimensions. ENCLUS search for interesting subspace whose entropy exceeds a threshold ω and interest gain exceeds ϵ'. Clusters in the interesting subspaces can be identified by the same methodology as CLIQUE. Parameters required by ENCLUS are grid interval Δ, entropy threshold ω, and interest threshold ϵ'.

2.2.2.3 MAFIA

CLIQUE and ENCLUS are sensitive to the uniform grid interval. MAFIA [58] introduces an adaptive grid based on the distribution of data to improve efficiency and cluster quality. MAFIA initially computes the histogram to determine the minimum number of bins for each feature. The adjacent cells of similar density are merged to form larger cells. In this manner, the dimension is divided into cells based on the data distribution and the resulting cluster parameters are captured more accurate. Once the bins have been defined, the clusterable subspaces are built up from on dimension as CLIQUE does. MAFIA requires the user to specify the density threshold and the threshold for merging adjacent windows. The running time grows exponentially with the number of dimensions in the clusters.

2.2.2.4 Cell-based Clustering Method (CBF)

The number of bins in many bottom-up algorithms increases dramatically as the number of features increases. To address this scalability issue, CBF [59] introduces a cell creation algorithm by splitting each dimension into a group of sections using a split index. The algorithm creates optimal partitions by repeatedly examining minimum and maximum values on a given dimension which results in the generation of fewer bins. CBF requires two parameters, *section threshold* which determines the bin frequency of a dimension, and *cell threshold* which determines the minimum density of data points in a bin. The results are sensitive to these two parameters.

2.2.2.5 CLTree

CLTree [60] uses a decision tree algorithm to partition each dimension into bins. It evaluates each dimension separately and then uses only those dimensions with areas of high density in further steps. To build CLTree, uniformly distributed noise data is added to the dataset, and the tree tries to split the real data from the noise. The density can be estimated for any given bin under investigation. After the tree is fully constructed, a pruning process is performed to obtain the final hyper-rectangle clusters. CLTree requires two parameters, *min_y* which is the minimum number of points that a region must contain, and *min_rd* which is the minimum relative density between two adjacent regions before the regions are merged to form a larger cluster.

2.2.2.6 DOC

Density-based Optimal projective Clustering (DOC) [61] is a hybrid method which blends the grid based bottom-up approaches and the iterative improvement method of the top-down approaches. DOC attempts to discover projective clusters which are defined as pairs (C, D) where C is a subset of the instances and D is a subset of dimensions of the dataset, such that C exhibits strong clustering tendency in D. The algorithm first selects a small subset X by random sampling. For a given cluster pair (C, D), instance p in C, and instance q in X, the following should hold true: for a dimension i in D, $|q(i) - p(i)| \leq w$, where w is the fixed side length of a subspace cluster or hyper-cube, given by the user. DOC also requires two additional parameters, α that specifies the minimum number of instances in a cluster, and β that specifies the balance between number of points and the number of dimensions in a cluster.

2.2.2.7 SURFING

SURFING (SUbspaces Relevant For clusterING) [62] computes all relevant subspaces and ranks them according to the interestingness of the hierarchical clustering structure they exhibit. The quality of a subspace is measured based on k-nearest-neighbor distance ($k - nn -$

$distance$). The algorithm first introduces a new variance measure that is half of the sum of the difference of all objects to the mean value of k-nn-distance. The quality of the subspace is defined by normalizing this variance to the production of mean value and the number of objects having a smaller k-nn-distance than the mean value. SURFING evaluates subspaces from one-dimension to l dimension. At each iteration, irrelevant subspaces (whose quality decreases w.r.t its $(l-1)$-dimensional subspace below a threshold) are discarded. The remaining l-dimensional subspaces are joined if they share any $(l-1)$ dimensions. SURFING yields a list of interesting subspaces ranked by their quality measure. Clusters existing in each subspace are further discovered by other clustering algorithms such as hierarchical clustering. SURFING requires k as the input parameter. The running time complexity is $O(2^d N^2)$, though [62] shows only a little percentage of subspaces are navigated in practice.

2.2.2.8 PROCLUS

PROCLUS [63] is a top-down subspace clustering algorithm. PROCLUS selects k mediods from a sampled dataset. Those mediods are improved by randomly choosing new medoids and replacing the bad ones. Cluster quality is based on the average distance between instances and the nearest medoid. For each medoid, a set of dimensions is chosen whose average distances are small compared to statistical expectation. Once the subspaces have been selected for each medoid, points are assigned to medoids according to the average Manhattan segmental distance. Clusters with fewer than $(N/k) \times minDeviation$ points, where $minDeviation$ is a input parameter, are thrown out. Finally, the clusters and the associated dimensions are refined based to the points assigned to the medoids. PROCLUS also requires the average dimensionality of subspaces as an input parameter. The algorithm is sensitive to the parameters which are difficult to be determined in advance.

2.2.2.9 COSA

COSA (Clustering On Subsets of Attributes) [64] assigns weights to each dimension for each instance, instead of each cluster. The algorithm starts with equally weighted dimensions.

The weights are updated according to the k-nearest neighbors (knn) of each instance. Higher weights are assigned to those dimensions that have a smaller dispersion within the knn group. New distances are calculated based on the updated weights. This process repeated until weights become stable. The neighborhoods for each instance are increasingly enriched with an instance belonging to its own cluster. The output is a COSA distance matrix based on weighted inverse exponential distance. Clusters are discovered by other distance based clustering algorithms such as hierarchical clustering. After clustering, the weight of each dimension for each cluster is computed based on that of its members. COSA does not need the number of dimensions in clusters to be specified in advance. Instead, it requires a input parameter λ to control the strength of intensive for clustering on more dimensions. Parameter k is also needed but the author claims that the results are stable over a wide range of k values.

2.3 Summary

Clustering is a fundamental technique in data mining and machine learning. Feature selection is essential in many clustering problems, which helps the user focusing on the important attributes of data groups. Feature selection in unsupervised learning is much harder than that for supervised learning, due to the fact that the class labels, which are used to guide feature searching in supervised learning, are unavailable. Feature selection in unsupervised learning arises research intention only very recent. Most related works concentrate on global feature selection which select a common feature subset for all the clusters. Searching subsets for individual clusters is a new research area. Available localized feature selection algorithms can be found in co-clustering and subspace clustering. Co-clustering yields exclusive feature subsets for clusters, which is not suitable in many applications. Subspaces clustering algorithms encountered difficulties such as heavy computational load, overlapping clusters, and/or requiring input parameters which are difficult to be determined in advance.

CHAPTER 3

NORMALIZED PROJECTION

In this chapter, we propose a heuristic localized feature selection algorithm for unsupervised learning. Our approach [65] computes adjusted and normalized scatter separability for individual clusters. A sequential backward search is then applied to find the optimal (perhaps local) feature subsets for individual clusters.

3.1 Introduction

Feature selection involves searching through various feature subsets, followed by the evaluation of each of them using some evaluation criteria [18–20, 30]. The most commonly used search strategies are greedy sequential searches through the feature space, either forward or backward. Different types of heuristics, such as sequential forward or backward searches, floating search, beam search, bidirectional search, and genetic search, have been suggested to navigate the possible feature subsets [11, 20, 41, 66]. In supervised learning, classification accuracy is widely used as evaluation criterion [19, 20, 30, 67, 68].

However in unsupervised learning, feature selection is more challenging since the class labels are unavailable to guide the search. Instead, clustering algorithms use some criteria, such as likelihood, entropy, or cluster separability measure to evaluate clustering quality and the feature subset quality. Regardless what the evaluation criteria are, global feature selection approaches compute them over the entire dataset. Thus, they can only find one relevant feature subset for all clusters. However, it is the local intrinsic properties of data that matter counts during clustering [69]. Such a global approach cannot identify individual clusters that exist in different feature subspaces. An algorithm that performs feature selection for each individual cluster separately is highly preferred.

The problem can best be illustrated using a synthetic dataset. We generate 400 data points with 4 clusters $\{C_1, C_2, C_3, C_4\}$ in 4 dimensional space $\{X_1, X_2, X_3, X_4\}$. Each cluster con-

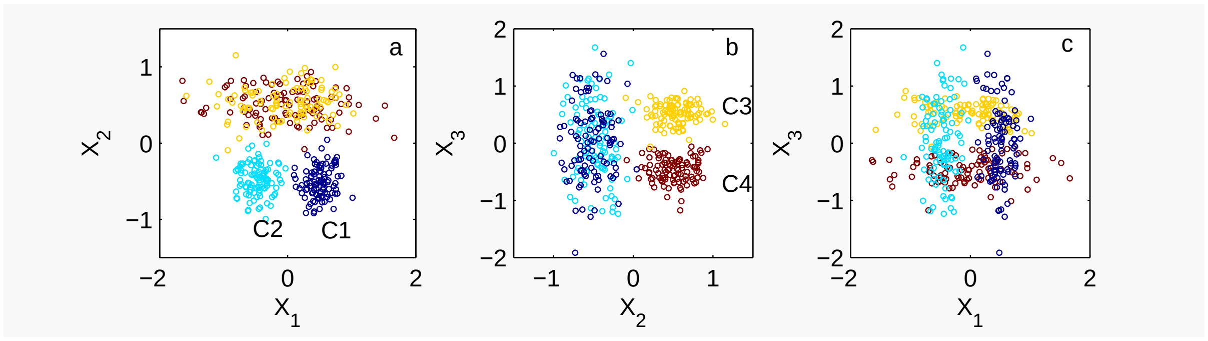


Figure 3.1: Synthetic data plotted in different feature sets. Data from different clusters are marked with different colors. a: in X_1 and X_2. b: in X_2 and X_3. c: in X_1 and X_3.

tains 100 points. Clusters C_1 and C_2 are created in dimensions X_1 and X_2 based on a normal distribution. X_3 and X_4 are white noise features in these two clusters. The means and standard deviations are: $\mu_{C_1} = [0.5, -0.5, 0, 0]$, $\mu_{C_2} = [-0.5, -0.5, 0, 0]$, and $\sigma_{C_1} = \sigma_{C_2} = [0.2, 0.2, 0.6, 0.6]$, respectively. Clusters C_3 and C_4 exist in dimensions X_2 and X_3 with white noise in X_1 and X_4, and are created in the same manner. The means and standard deviations are: $\mu_{C_3} = [0, 0.5, 0.5, 0]$, $\mu_{C_4} = [0, 0.5, -0.5, 0]$, and $\sigma_{C_3} = \sigma_{C_4} = [0.6, 0.2, 0.2, 0.6]$, respectively. Figure 3.1 shows the data in different subspaces. A general clustering algorithm, such as k-means or EM, is unable to obtain satisfactory clustering results for this data either on all features $\{X_1, X_2, X_3, X_4\}$, or on relevant feature subset $\{X_1, X_2, X_3\}$ (may be generated by a global feature selection algorithm) because each cluster still has one irrelevant feature. For data in higher dimensional space, this problem becomes more prominent.

On the other hand, if we further remove X_3 from the feature subset $\{X_1, X_2, X_3\}$, we can completely separate C_1 and C_2, as shown in Figure $1a$. Similarly, C_3 and C_4 can be well separated by removing X_1 as shown in Figure $1b$. In addition, the clustering results of localized feature selection provides a better understanding of the underlying process that generates the data. For example $C_1 \sim \{X_1, X_2\}$ clearly indicates that cluster C_1 is mainly generated by features X_1 and X_2.

Usually, there are two major components for a feature selection algorithm: evaluation criteria and feature subset search methods. In the following, we first discuss the evaluation criterion for the localized feature selection algorithm, then the search method.

3.2 Evaluation Criteria

In this section, we first provide a brief introduction to scatter separability criterion, one of the well-known clustering criteria [21], and then show how this criterion could be adapted to localized feature selection.

Let S_w and S_b denote within-class scatter matrix and between-class scatter matrix, respectively. We have,

$$S_w = \sum_{i=1}^{k} \pi_i E\{(X - \mu_i)(X - \mu_i)^T | C_i\} = \sum_{i=1}^{k} \pi_i \Sigma_i, \tag{3.1}$$

$$S_b = \sum_{i=1}^{k} \pi_j (\mu_i - \mu_0)(\mu_i - \mu_0)^T, \tag{3.2}$$

$$\mu_0 = E\{X\} = \sum_{i=1}^{k} \pi_i \mu_i, \tag{3.3}$$

where π_i is the probability that an instance belongs to cluster C_i, X the d-dimensional input dataset, k the number of clusters, μ_i the sample mean vector of cluster C_i, μ_0 the total sample mean, Σ_i the sample covariance matrix of cluster C_i and $E\{\cdot\}$ the expected value operator.

Since S_w measures how scattered the samples are from their cluster mean, and S_b measures how scattered the cluster means are from the total mean, the scatter separability is defined as

$$CRIT = \mathbf{tr}(S_w^{-1} S_b) \tag{3.4}$$

Although there are a bunch of other separability criteria available, the measure $CRIT$ enjoys a nice property that it is invariant under any non-singular linear transformation [43]. However, this criteria requires a non-singular within-class scatter matrix S_w. In the case that the S_w is singular, the following separability criteria can be used instead,

$$CRIT = \mathbf{tr}(S_b)/\mathbf{tr}(S_w) \tag{3.5}$$

In the remainder of this paper, we use $\mathbf{tr}(S_w^{-1}S_b)$ in our discussion. However, one should be aware that $\mathbf{tr}(S_b)/\mathbf{tr}(S_w)$ is used for a singular S_w.

Similar to the definition of S_w, we define $S_w^{(i)}$, the within-class matrix of an individual cluster C_i as,

$$S_w^{(i)} = \frac{1}{n_i}E\{(X-\mu_i)(X-\mu_i)^T|C_i\} = \frac{1}{n_i}\Sigma_i \tag{3.6}$$

where n_i is the number of points in cluster C_i. Now we are ready to define the scatter separability of cluster C_i.

Definition 1. *The scatter separability of cluster C_i is defined by,*

$$CRIT(C_i) = \mathbf{tr}(S_w^{(i)-1}S_b) \tag{3.7}$$

Assuming that identical clustering assignments are obtained when more features are added, the scatter separability $CRIT$ prefers higher dimensionality since the criterion value monotonically increases as features are added [43]. The same conclusion could be drawn for the scatter separability for an individual cluster. Specifically, in [43], it is shown that a criterion of the form $X_d^T S_d X_d$, where X_d is d-column vector and S_d is a $d \times d$ positive definite matrix, monotonically increases with dimension. Based on this, we have,

Proposition 1. *$CRIT(C_i)$ monotonically increases with dimensions as long as the clustering assignments remain the same.*

Proof. Since S_b can be expressed as $\sum_{j=1}^{k} Z_j Z_j^T$ where Z_j is a column vector.

$$\begin{aligned} CRIT(C_i) &= \mathbf{tr}(S_w^{(i)-1} S_b) \\ &= \mathbf{tr}(S_w^{(i)-1} \sum_{j=1}^{k} Z_j Z_j^T) \\ &= \sum_{j=1}^{k} \mathbf{tr}(S_w^{(i)-1} Z_j Z_j^T) \\ &= \sum_{j=1}^{k} \mathbf{tr}(Z_j^T S_w^{(i)-1} Z_j) \\ &= \sum_{j=1}^{k} Z_j^T S_w^{(i)-1} Z_j \end{aligned} \tag{3.8}$$

Every term of Equation (3.8) monotonically increases with dimension, thus the criterion for an individual cluster $CRIT(C_i)$ monotonically increases with dimension. □

To alleviate this problem, normalization of the separability criterion with respect to dimensions is necessary for feature selection [21]. Moreover, for localized feature selection strategies, each cluster is associated with a distinct feature subset. It is usually impossible to compute S_b without proper normalization.

In the proposed algorithm, the normalization is performed using cross-projection over individual clusters. Suppose we have a cluster set C,

$$C = \{(C_1, S_1), \ldots (C_i, S_i), \ldots, (C_k, S_k)\} \tag{3.9}$$

where S_i is the feature subset corresponding to cluster C_i. To calculate the scatter separability of (C_i, S_i) in cluster set C, we project all the clusters of C into feature subset S_i, and extend the scatter separability of cluster C_i as follows,

Definition 2. *The scatter separability of cluster C_i in cluster set C on feature subset S_i is given* *by,*

$$CRIT(C_i, S_i)|_C = \mathbf{tr}(S_w^{(i)-1} S_b)|_{C,S_i} \tag{3.10}$$

where $|_{C,S_i}$ *denotes the project of cluster set* C *onto feature subset* S_i.

Assume an iteration of search produces a new cluster set C' on subspace S'_i,

$$C' = \{(C'_1, S'_i), \ldots (C'_i, S'_i), \ldots, (C'_k, S'_i)\} \tag{3.11}$$

Let's also assume that cluster (C'_i, S'_i) corresponds to cluster (C_i, S_i), i.e., (C'_i, S'_i) is the cluster that has the largest overlapping with (C_i, S_i) in set C'. We then generate a new cluster set, C^*, by replacing (C_i, S_i) in C with (C'_i, S'_i),

$$C^* = \{(C_1, S_1), \ldots (C'_i, S'_i), \ldots, (C_k, S_k)\} \tag{3.12}$$

Note that $CRIT(C_i, S_i)|_C$ and $CRIT(C'_i, S'_i)|_{C^*}$ can not be compared directly because of the dimension bias. We have to cross-project them onto each other,

$$NV(C_i, S_i)|_C = CRIT(C_i, S_i)|_C \cdot CRIT(C_i, S'_i)|_C \tag{3.13}$$

$$NV(C'_i, S'_i)|_{C^*} = CRIT(C'_i, S'_i)|_{C^*} \cdot CRIT(C'_i, S_i)|_{C^*} \tag{3.14}$$

After the cross-projection, the bias is eliminated and the normalized value NV can be used to compare two clusters in different feature subspaces. A larger value of NV indicates larger separability, i.e., better cluster structures.

3.2.1 Penalty of Overlapping and Unassigned Points

Localized feature selection implicitly creates overlapping and/or unassigned data points. Overlapping points are the data which belongs to more than one cluster, while unassigned points are the data which belongs to non-cluster. Specifically, the overlapping measure O can be computed as,

$$O = \sum_{i \neq j}^{k} \frac{|C_i \cap C_j|}{mean(|C_i|, |C_j|)} \tag{3.15}$$

where C_i and C_j are two different clusters. And unassigned measure U can be computed as,

$$U = \frac{n_u}{n} \tag{3.16}$$

where n and n_u are the total number of data and the number of unassigned points, respectively. Overlapping and/or unassigned data are allowed in some applications, and may be forbidden by other applications. Depending on the domain knowledge, we could adjust the impact of overlapping and unassigned points by introducing a penalty and obtain the adjusted normalized value ANV.

Definition 3. *The adjusted and normalized scatter separability pair of cluster C_i in cluster set C on feature subset S_i and cluster C'_i in cluster set C^* on feature subset S'_i is given by,*

$$ANV(C_i, S_i)|_C = NV(C_i, S_i)|_C \cdot e^{(-\alpha\Delta O - \beta\Delta U)} \tag{3.17}$$

$$ANV(C'_i, S'_i)|_{C^*} = NV(C'_i, S'_i)|_{C^*} \cdot e^{(\alpha\Delta O + \beta\Delta U)} \tag{3.18}$$

where ΔO and ΔU are the changes on the overlapping and unassigned measure, respectively, if cluster (C_i, S_i) is replaced by cluster (C'_i, S'_i). α and β are two constants.

In Definition 3, α and β are used to control the sensitivity with respect to overlapping points and unassigned points. Large values of α and β discourage the occurrence of overlapping and unassigned data. On the other hand, if α or β is zero, the corresponding effect of overlapping or unassigned data will be ignored when comparing two clusters. The values for α and β depend on the given application and have to be determined empirically. For example, if a large portion of data is unassigned after clustering, β needs to be increased.

When two clusters (C_i, S_i) and (C'_i, S'_i) are compared, if $ANV(C_i, S_i)|_C > ANV(C'_i, S'_i))|_{C^*}$, we choose (C_i, S_i). If $ANV(C_i, S_i)|_C = ANV(C'_i, S'_i))|_{C^*}$, we prefer the cluster in the lower dimensional space. In addition, when two identical clusters are obtained in two different feature subsets, they have equal adjusted normalized value ANV, which is exactly what we want. More formally,

Proposition 2. *Given two identical clusters $C_1 = C_2$, and the corresponding feature subspaces S_1 and S_2, the adjusted normalized value $ANV(C_1, S_1) = ANV(C_2, S_2)$.*

Proof. Since $C_1 = C_2$, we have $C = C^*$. Thus

$$\begin{aligned} NV(C_1, S_1) &= CRIT(C_1, S_1) \cdot CRIT(C_1, S_2) \\ &= CRIT(C_2, S_2) \cdot CRIT(C_2, S_1) \\ &= NV(C_2, S_2) \end{aligned}$$

And $\Delta O = \Delta U = 0$. Thus

$$ANV(C_1, S_1) = ANV(C_2, S_2) \tag{3.19}$$

$\square$

3.2.2 Unassigned/New data

In case some new data is obtained or unassigned data is not allowed by an application, assignments have to be made after clustering for these new/unassigned points. The similarity of an instance and a cluster could be measured by either distance (k-means clustering), or likelihood (EM algorithm). The additional difficulty introduced by localized feature selection algorithm is that clusters are associated with different feature subsets, making the direct comparison among clusters meaningless. For distance based similarity, a straightforward solution is to normalize the distance measure over its variance within each cluster, and assign the instance to a cluster that minimizes the normalized distance,

$$arg \min_{C_j} d = arg \min_{C_j} \left(\frac{\| X_i|_{S_j} - \mu_j \|}{\sigma_j^2} \right) \tag{3.20}$$

where X_i is an unassigned point, μ_j the cluster mean vector of C_j, S_j the feature subset of C_j, $X_i|_{S_j}$ the projection of X_i into S_j, and $\| \cdot \|$ is the norm of a vector. A similar method could be developed for likelihood based similarity measure.

3.3 Search Methods

The cross-projection normalization scheme assumes that the clusters to be compared should be consistent in the structure of the feature space [21]. Consequently, we select sequential backward search instead of the sequential forward search adopted in [21]. The trade off is the slower clustering speed.

Specifically, the data are first clustered based on all available features. Then, for each cluster, the algorithm determines if there exists a redundant or noisy feature based on the adjusted normalized value ANV defined in Equations (3.17) and (3.18). If so, it will be removed. The above process is repeated iteratively on all clusters until no change is made, at which time the clusters with the associated feature subsets will be returned. The sequence of steps shown in Figure 3.2 illustrates our algorithm in detail.

The complexity is $O(ndik)$ for the conventional k-means algorithm, and $O(nd^2ik)$ for the GFS-k-means algorithm, respectively, where n is the number of points, d the number of features, i the number of iteration (usually unknown), and k the number of clusters. The complexity of our approach, in worst case, is $O(nd^3k^2i)$ with backward sequential search. It shows that for data sets with very high dimensions and large number of clusters, the proposed algorithm is slow compared to general k-means and global feature selecting algorithms. However the complexity is in polynomial form, thus is still acceptable in practice.

```
input  : Dataset $X_{n \times d}$
output: Clusters $C = \{(C_i, S_i) | i = 1, \dots, k\}$
initialize $C$ with all features;
repeat
    for $i = 1$ to $k$ do
        Create a new subset $S'_i$ by removing one feature from $S_i$ ;
        Generate a new cluster set $C'$ on $S'_i$ ;
        Compare clusters in $C'$ with corresponding clusters in $C$;
        if Better cluster found then
            Replace the corresponding cluster in $C$
        end
    end
until No change made ;
if Desired then
    Process unassigned data points
end
```

Figure 3.2: The proposed localized feature selection algorithm.

3.4 Experiment and Results

We evaluate the localized feature selection algorithm using both synthetic and real-world datasets. The experiment results are obtained by choosing k-means as the clustering algorithm. However, note that the adjusted normalized value ANV is not restricted to k-means. It can be used together with any general clustering algorithm.

In general, it is difficult to evaluate the performance of a clustering algorithm on high dimensional data. Localized feature selection presents an additional layer of complexity by associating clusters to different feature subsets. Therefore, we take a gradual approach for our evaluation. We first test the proposed algorithm on a small synthetic dataset with known data distribution along each feature dimension. Then, we investigate five real-world datasets downloaded from UCI repository [70]. On all UCI datasets, we perform a semi-supervised learning strategy for evaluation purpose. This makes it possible for us to compute a pseudo-accuracy measure for easy comparison among different algorithms. However, one should be aware that the "true" class labels are not always consistent with the nature grouping of the underlying dataset. Thus, the quality of clusters should be further analyzed in addition to the pseudo-accuracy. For this purpose, we also illustrate our results by visually examining the clusters in the selected feature subspace on synthetic data and Iris data.

On each dataset, we compare our localized feature selection algorithm (with k-means, denoted by LFS-k-means) with global feature selection algorithm (also with k-means, denoted by GFS-k-means), and k-means without feature selection. GFS-k-means is implemented in a similar fashion as [21]. The only difference is that we adopted the backward search strategy due to the reason discussed in Section 2.3.

On the above experiments, the number of clusters k is set to the "true" number of classes. This is not always applicable in real world applications. How to determine the value of k is a common problem in unsupervised learning. It may strongly interact with the predicted cluster structures, as well as the selected feature subset in feature selection algorithms [1]. There are several algorithms available to determine k, i.e., [1, 43, 71]. Another common problem that a clustering algorithm usually faces is how to initialize cluster centroids. Bad initial clusters/centroids might lead to low quality clusters. In traditional clustering algorithms, some techniques, such as randomly picking up k patterns over the dataset, preliminary clustering, or choosing the best from several iterations, are frequently used to alleviate the chance of bad initial clusters. In our approach, bad initial clusters for backward searching may occur more often when many noise features presented, and might affect the final clusters and feature subsets largely. This problem can be alleviated by preliminary clustering with a global feature selection, i.e., [1].

We incorporate another experiment as an example solution for unknown k and preliminary clustering in Section 3.4.4. In this section, we evaluate our algorithm over another three UCI datasets with unknown k. We first employ the algorithm proposed in [1] to estimate the number of clusters, global feature saliencies and cluster centroids. Then we use them as initial parameters and run our algorithm on the particular dataset. Clusters obtained are labeled to its majority portion of true classes. Errors are calculated accordingly.

Table 3.1: Confusion matrix and error rate on the synthetic data. C1 - C4 are the output cluster labels, and T1 - T4 are the true cluster labels

Algo	k-means				k-means w/o X_4				GFS-k-means				LFS-k-means			
Label	C1	C2	C3	C4	C1	C2	C3	C4	C1	C2	C3	C4	C1	C2	C3	C4
T1	77	22	1	0	59	40	1	0	37	17	46	0	99	0	0	1
T2	3	76	0	21	45	49	0	6	33	22	45	0	0	100	0	0
T3	1	7	89	3	0	3	69	28	26	16	58	0	0	1	98	1
T4	23	0	9	68	3	0	45	52	35	14	51	0	2	0	0	99
Error	0.225				0.428				0.708				0.01			

Table 3.2: Feature subset distribution on the synthetic data. C1 - C4 are the output cluster labels.

Algorithm	Feature Subset(s)			
	C1	C2	C3	C4
k-means	$\{1, 2, 3, 4\}$			
GFS-k-means	$\{4\}$			
LFS-k-means	$\{1, 2\}$	$\{1, 2\}$	$\{2, 3\}$	$\{2, 3\}$

3.4.1 Synthetic data

The synthetic data is described in Section 3.1 and illustrated in Figure 3.1. Penalties of overlapping and unassigned points (α and β) are set at 1.

Table 3.1 shows the confusion matrix and error rate of k-means with full feature set, k-means without the totally irrelevant feature X_4, GFS-k-means, and LFS-k-means, and Table 3.2 shows the selected feature subsets. Clearly, by employing all four available features, k-means performs poorly with a error rate of 0.225, which indicates that irrelevant features greatly reduce the clustering performance. Meanwhile, GFS-k-means does a terrible job with an unacceptable error rate of 0.708. The output feature subset contains only the noisy feature X_4! This surprising result could be explained as follows. Since each feature is irrelevant to at least two clusters and each cluster has at least two irrelevant features, NO feature subset are relevant to all clusters. We also evaluated k-means algorithm on the feature subset X_1, X_2, X_3, which are the globally relevant features that could probably be obtained by a *smart* global

Table 3.3: Confusion matrix and error rate on iris data. C1 - C3 are the output cluster labels, and T1 - T3 are the true cluster labels

Algo	k-means			GFS-k-means			LFS-k-means		
Label	C1	C2	C3	C1	C2	C3	C1	C2	C3
T1	50	0	0	50	0	0	50	0	0
T2	0	39	11	0	46	4	0	48	2
T3	0	14	36	0	3	47	0	4	46
Error	0.167			0.0467			0.04		

Table 3.4: Feature subset distribution on iris data. C1-C3 are the output cluster labels.

Algorithm	Feature Subset(s)		
	C1	C2	C3
k-means	$\{1, 2, 3, 4\}$		
GFS-k-means	$\{3\}$		
LFS-k-means	$\{4\}$	$\{3, 4\}$	$\{3, 4\}$

feature selection algorithm, as shown in Table 3.2. The error rate is as high as 0.428, indicating that the group structures can not be recognized with globally relevant feature subset. The reason is that the structures are buried not only by the irrelevant feature X_4, but also by the relevant features X_1 and X_3. On the other hand, the proposed localized feature selection algorithm produces an excellent result with a error rate of 0.01. From Table 3.2, we can see clearly that the relevant features for each cluster are selected correctly, and the clusters are well separated in the corresponding feature subspaces (Figures 1a and 1b). This result confirms that selecting features locally is meaningful and necessary in clustering.

3.4.2 Iris data

Iris data from UCI is a widely used machine learning benchmark dataset for both supervised learning and unsupervised learning. This data has three classes, four features, and 150 instances. In this experiment, we set α and β to be 1 and 6, respectively.

Table 3.3 shows the confusion matrix and error rate of k-means, GFS-k-means, and LFS-k-means, respectively, and Table 3.4 show the corresponding feature subsets. k-means, with

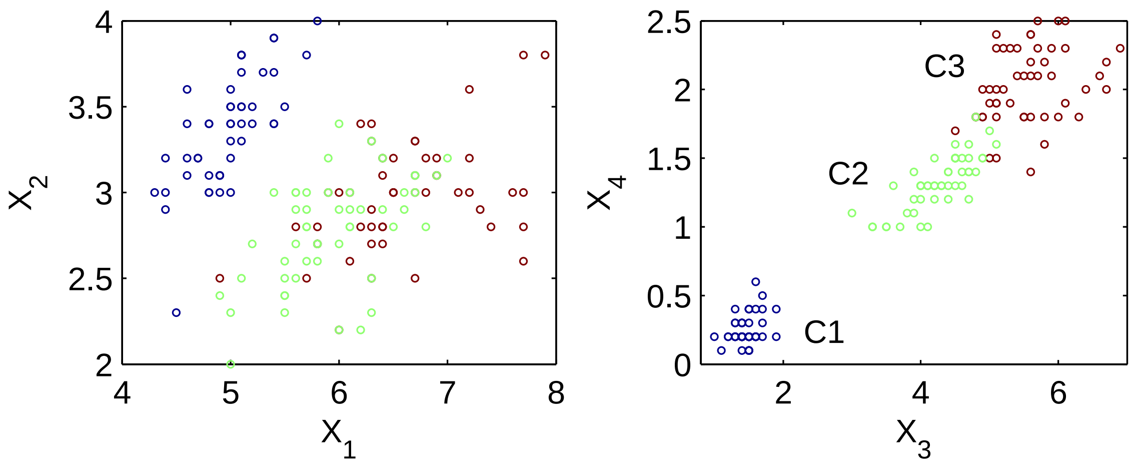


Figure 3.3: Scatterplots on iris data using features 1 and 2 (left panel), and using features 3 and 4 (right panel). Data from different classes are marked with different colors.

all four features, is able to successfully identify cluster 1, “iris-setosa”. However it does not perform well on cluster 2, “iris-versicolor”, with a error rate of 0.22, and cluster 3, “iris-virginica”, with a error of 0.28. The GFS-k-means discards feature 1, 2, and 4, and recognizes the structure of the dataset much better with only feature 3. The proposed LFS-k-means results in the best pseudo-accuracy. The selected feature subsets show that cluster 1 can be separated along feature 4, clusters 2 and 3 can be separated along features 3 and 4. The right panel of Figure 3.3 shows the scatter plot of iris data along features 3 and 4. Clearly, cluster 1 can be separated either by feature 3 or by feature 4. In other words, one of the features is redundant to cluster 1. The proposed algorithm keeps feature 4 and removes feature 3 from the subset. The selected features for clusters 2 and 3 (features 3 and 4) are also consistent with our visual inspection. The left panel of Figure 3.3 clearly shows that features 1 and 2 are not helpful to differentiate these two clusters.

The experiment results on iris dataset show that the proposed algorithm is capable of reducing redundant/noisy features for each individual cluster. It can also provide us a better understanding of the date generation.

Table 3.5: Comparison of k-means, GFS-k-means and LFS-k-means on other UCI data sets

Data Set				Subfeature			Error		
Name	Patt.	Feat.	Clas.	k-means	GFS	LFS	k-means	GFS	LFS

Wine	178	13	3	13	{1 2 3 4 5 6 8 9 10 11 12 13}	C1: {1 3 4 5 7 8 10 11 12 13 }
						C2: {1 2 3 4 5 6 7 8 9 10 11 12 13}
						C3: {3 4 5 9 10 11 12 13}
							0.034	0.039	0.023

Ion	351	32	2	32	{3 7 11 13 15 17 19 29 30 31}	C1: {13}
						C2: {3 7 11 13 15 17 19 29 30 31}
							0.288	0.296	0.296

Glass	214	9	5	9	{2 3 5 6 7 8 9 }	C1: {4 5 7 9}
						C2: {2 3 4 5 7 8 9}
						C3: {3 5 7 9}
						C4: {6 8}
						C5: {5 6}
							0.192	0.201	0.196

Sonar	208	60	2	60	{35 36 37 38 41 42 44 46 47 51 55 56 57 58 59 60}	C1: {9 10 49 50 51 56 58}
						C2: {9 10 49 50 51 56 58}
							0.452	0.466	0.375

3.4.3 Other UCI data

We also evaluated LFS-k-means, and compared the results with k-means and GFS-k-means, on four other UCI datasets, Wine, Ion, Sonar, and Glass, which are more complicated than Iris data set in terms of number of features and number of classes. From Wine to Ion to Sonar, the number of features increases from 13 to 32 to 60 with two or three classes. From Ion to Wine to Glass data set, the number of classes increases from 2 to 3 to 5. Table 3.5 shows the experiment results.

For Wine data set, GFS-k-means kept 12 out 13 features with accuracy of 0.039. On the other hand, LFS-k-means selected 10, 13 and 8 features for different clusters, respectively, with better accuracy of 0.023.

For the Ion data set, GFS-k-means selected 10 features for both clusters. Comparing to GFS-k-means, our proposed algorithm results in 1 feature for cluster C1 and 10 features for the other cluster C2. Notice that these 10 features for C2 are identical to those selected by GFS-k-means. This implies that localized feature selecting algorithm performs at least the same as global feature selecting algorithm. Furthermore, it also shows that it is often unsuitable to only select one feature subset for all the clusters in unsupervised learning.

Experiments on Glass and Sonar data sets give similar results. In summary, LFS-k-means leads to variant feature subsets for different clusters, and provide best (on Wine and Sonar) or similar (on Ion and Glass) pseudo-accuracy comparing to conventional k-means algorithm and GFS-k-means. In addition, the feature subsets selected by LFS-k-means are usually much shorter than GFS-k-means. These results confirm that clusters do exist on localized feature subsets for certain problems.

3.4.4 UCI data with estimation of k and initial clusters

Table 3.6: UCI datasets with estimated number of clusters and initial centroids. GFS: Global feature selection and clustering by algorithm of [1]. $\hat{k}$: Estimated number of clusters by GFS. LFS: Local feature selection by the proposed algorithm with $\hat{k}$ and initial centroids obtained by GFS.

Data Set				GFS		LFS	Error	
Name	Patt.	Feat.	Clas.	$\hat{k}$	Salient Feat.	Feat. Subset	GFS	LFS

WDBC	576	30	2	8	{29 features}	C1: {24 features}
						C2: {25 features}
						C3: {13 14 16 17 23 26 29}
						C4: {26 features}
						C5: {25 features}
						C6: {4 13 14 16 23 26}
						C7: {4 13 14 16 23 26 29}
						C8: {4 14 16 23 26 29}
							0.09	0.10

Image	2310	18	7	18	{17 features}	C1:{7 8 14 17}
						C2: {12 13}
						C3: {2 3 9 11 13 14 15 16 18}
						C4: {3 4 5 9 10 13 16 18 }
						C5: {5 18}
						C6: {18 }
						C7: {18}
						C8: {17 features}
						...
							0.19	0.28

Zernike	2000	47	10	17	{45 features}	C1:{16 features}
						C2: {22 features}
						C3: {13 features}
						C4: {13 features}
						C5: {2 features}
						C6: {44 features}
						C7: {16 features}
						C8: {44 features}
						....
							0.49	0.48

In this section, we evaluate our algorithm on another three UCI datasets, WDBC, Image, and Zernike. WDBC is the Wisconsin diagnostic breast cancer data set with 30 features and 576 patterns from 2 classes (benign or malignant). Image is the image segmentation data set with 2310 patterns and 19 features (18 of them are non-singular) from seven categories (brick-face, sky, foliage, cement, window, path and grass). Zernike contains 47 Zernike moments extracted from 2000 handwriting numerals (0-9), 200 for each digit. Those datasets contain many features and enough numbers of patterns to perform the algorithm proposed in [1].

We suppose that the number of clusters k is unknown for those datasets. We run global feature selection algorithm presented by Law et. cl. [1] with 30 initial clusters, and obtain the estimated value of k, cluster centroids, and global feature saliencies. We run our algorithm over those parameters. Only features with saliency greater than 0.5, called global salient features, are kept. The experiment results are presented in Table 3.6. For Image and Zernike datasets, we only show the first 8 clusters.

On WDBC, the GFS algorithm lead to 29 salient features out of 30. Our approach produces different feature subset for each cluster. The size of feature subsets varies from 6 to 26, with average value of 15.8, which is much less than the size of feature subset obtained by GFS. The same results are observed on both Image and Zernike dataset: On Image dataset, feature subset size varies from 1 to 17 with average value of 6.3, while the size of GFS's is 17. On Zernike dataset, feature subset size varies from 2 to 45 with average value of 22.7, while the size of GFS's is 45.

The error rates of GFS and LFS on WDBC are almost the same (0.09 and 0.10, respectively), as well as the error rate on Zernike (0.49 and 0.48 respectively), which implies that our clustering results are comparative to GFS over those datasets. Note that the error rate on Image is different: 0.19 for GFS and 0.28 for LFS. However one cannot thereby conclude that the clustering quality of LFS are much worse than that of GFS on this dataset, since the cluster structures may be ambiguous between the true classes in this dataset. The benefit of LFS here is relatively small subset of features for individual clusters.

3.5 Summary

In order to identify individual clusters that exist in different feature subspaces, we propose a localized feature selection algorithm. We develop adjusted and normalized scatter separability (ANV) for individual clusters, based on which our algorithm is capable of reducing redundant/noisy features for each cluster separately. The proposed algorithm can also provide us better understanding of the underlying process that generates the data. Our experiment results on both synthetic and real-world datasets show the need for feature selection in clustering and the benefits of selecting features locally.

In this chapter we employ cross-projection method to evaluate the quality of an individual cluster, which makes it impracticable to change the number of clusters during clustering and feature selection process. Thus a fixed k estimated in advance is required to perform localized feature selection with our approach. However in the area of unsupervised learning with feature selection, algorithms with simultaneously computing the number of clusters and the feature subset will be more elegant and desirable. We will try to solve this problem in our future research work.

CHAPTER 4

LOCALIZED FEATURE SELECTION WITH MAXIMAL LIKELIHOOD METHOD

The localized feature selection algorithm we present in Chapter 3 utilizes normalized cross-projection methods to evaluate the quality of clustering and feature subsets, and backward search to find the best subspaces. The draw back of this approach is that the computation cost is high, and the number of clusters k has to be specified in advance. In this chapter, we introduce a probabilistic model based on Gaussian mixture to tackle those issues [72]. Particularly, the feature relevance for an individual cluster is treated as a probability, which is represented by localized feature saliency and estimated through Expectation Maximization (EM) algorithm during the clustering process. In addition, the number of clusters is determined by integrating a Minimum Message Length (MML) criterion.

4.1 Background on EM-based Clustering and Global Feature Selection

From a *model-based* perspective, each cluster can be mathematically represented by a parametric distribution. The entire dataset is therefore modeled by a mixture of these distributions. The most widely used model in practice is the mixture of Gaussians. The clustering process thereby turns to estimating the parameters of the Gaussian mixture, usually by the EM algorithm.

Traditionally, a finite mixture of densities with K components is represented by,

$$p(y) = \sum_{j=1}^{K} \alpha_j p(y|\theta_j), \tag{4.1}$$

where α_j is the *a priori* probability, and θ_j is a set of parameters of component j. The parameters are estimated by maximizing the likelihood as,

$$\hat{\theta}_{ML} = \arg\max_{\theta} \left[\log p(\mathcal{Y}|\theta)\right]. \tag{4.2}$$

Let $\mathcal{Z} = \{z_{ij}\}_{N\times K}$ be a set of missing (latent) cluster labels, where $z_{ij} = 1$ if y_i is a sample of $p(\cdot|\theta_j)$, and $z_{ij} = 0$ otherwise. $\mathcal{Z}$ can be also written as a vector $\mathcal{Z} = (z_1, \ldots, z_N)$ such that $z_i = j$ if y_i is a sample of $p(\cdot|\theta_j)$. The log-likelihood when $\mathcal{Z}$ is observed is,

$$\log p(\mathcal{Y}, \mathcal{Z}|\theta) = \sum_{i=1}^{N}\sum_{j=1}^{K} z_{ij} \log[\alpha_j p(y_i|\theta_j)] \tag{4.3}$$

Let $\mathcal{W} = \mathrm{E}[\mathcal{Z}|\mathcal{Y}, \hat{\theta}(t)]$ represent the expected value of $\mathcal{Z}$, where $\hat{\theta}(t)$ is the estimate of θ at iteration t. The parameters can be estimated by the following updating rule,

$$\hat{\theta}(t+1) = \arg\max_{\theta}\{\log p(\mathcal{Y}, \mathcal{W}|\hat{\theta}(t))\} \tag{4.4}$$

Assuming features are conditionally independent, the mixture of densities can be described as,

$$p(y|\theta) = \sum_{j=1}^{K} \alpha_j p(y|\theta_j) = \sum_{j=1}^{K} \alpha_j \prod_{l=1}^{D} p(y_l|\theta_{jl}) \tag{4.5}$$

where D is the number of features. Define the global feature saliency ρ_l to be the probability that feature l is salient to all the components. Then, $(1-\rho_l)$ is the probability that l is not salient to any of the components. Let $\Phi = (\phi_1, \ldots, \phi_D)$ be the feature relevance vector with $\phi_l = 1$, if feature l is relevant and, $\phi_l = 0$, otherwise. Then, $\rho_l = \Pr(\phi_l = 1)$. Finally, the likelihood function can be rewritten as [1],

$$p(y|\theta) = \sum_{j=1}^{K} \alpha_j \prod_{l=1}^{D} [\rho_l p(y_l|\theta_{jl}) + (1-\rho_l) q(y_l|\lambda_l)] \tag{4.6}$$

where $q(\cdot(\lambda_l))$ is a common density, which defines the distribution of an irrelevant feature l.

If we treat Φ as missing variables, the feature saliency vector ρ can be estimated by the EM algorithm [1].

4.2 Detecting Clusters Embedded in Feature Subspace

In this section, we present a probabilistic model based on Gaussian mixture to detect clusters embedded in feature subspace. First, we define a localized feature saliency and show how it could be integrated into EM clustering. Then, we estimate the number of clusters with the MML criterion.

4.2.1 Localized Feature Saliency

In our approach, the importance of a feature can be different for different clusters, which implies that the feature relevance takes a matrix form, $\Phi = \{\phi_{jl}\}_{K \times D}$, where $\phi_{jl} = 1$ indicates that feature l is associated with component j, otherwise $\phi_{jl} = 0$. Let $\rho_{jl} = \Pr(\phi_{jl} = 1)$ be the probability that feature l is relevant to component j. Then, the likelihood can be obtained based on the following proposition.

Proposition 3. *Let $p(\cdot|\theta_{jl})$ represent the distribution of a salient feature l for a particular component j, and $q(\cdot|\lambda_{jl})$ the distribution if feature l is non-salient to the particular component. Assuming that the features are conditionally independent, the likelihood function can be written as,*

$$p(y|\theta) = \sum_{j=1}^{K} \alpha_j \prod_{l=1}^{D} \big(\rho_{jl} p(y_l|\theta_{jl}) + (1-\rho_{jl}) q(y_l|\lambda_{jl})\big) \tag{4.7}$$

Derivation. Let $\phi_j = (\phi_{j1}, \ldots, \phi_{jD})$. For a particular component j, we have

$$\begin{aligned}
p(y|z=j,\phi_j) &= \prod_{l=1}^{D} \big(p(y_l|\theta_{jl})\big)^{\phi_{jl}} \big(q(y_l|\lambda_{jl})\big)^{1-\phi_{jl}} \\
p(y,\phi_j,z=j) &= p(y|z=j,\phi_j)p(\phi_j|z=j)P(z=j) \\
&= \alpha_j \prod_{l=1}^{D} \big(p(y_l|\theta_{jl})\big)^{\phi_{jl}} \big(q(y_l|\lambda_{jl})\big)^{1-\phi_{jl}} \prod_{l=1}^{D} \rho_{jl}^{\phi_{jl}} (1-\rho_{jl})^{1-\phi_{jl}} \\
&= \alpha_j \prod_{l=1}^{D} \big(\rho_{jl} p(y_l|\theta_{jl})\big)^{\phi_{jl}} \big((1-\rho_{jl}) q(y_l|\lambda_{jl})\big)^{1-\phi_{jl}} \qquad (4.8)
\end{aligned}$$

Marginal density on y gives

$$\begin{aligned}
p(y|\theta) &= \sum_{j,\Phi}^{K} p(y,\phi_j,z=j) \\
&= \sum_{j=1}^{K} \alpha_j \sum_{\phi_{jl}} \prod_{l=1}^{D} (\rho_{jl} p(y_l|\theta_{jl}))^{\phi_{jl}} ((1-\rho_{jl}) q(y_l|\lambda_{jl}))^{1-\phi_{jl}} \\
&= \sum_{j=1}^{K} \alpha_j \prod_{l=1}^{D} (\rho_{jl} p(y_l|\theta_{jl}) + (1-\rho_{jl}) q(y_l|\lambda_{jl})) \qquad (4.9)
\end{aligned}$$

where $\theta = \{\{\alpha_j\}, \{\theta_{jl}\}, \{\rho_{jl}\}, \{\lambda_{jl}\}\}$ is the set of all the parameters. $\square$

Taking $\{z_{ij}\}$ and $\{\phi_{jl}\}$ as latent variables, we derive the E-step and M-step of the EM algorithm to estimate the parameter set.

E-Step: Compute the expectation of the log-likelihood.

From Equation (4.8), the expected complete log-likelihood of the dataset based on $\theta^{(t)}$ is

$$\begin{aligned}
&E_{\theta^{(t)}}[\log P(\mathcal{Y}, z, \Phi)] \\
=&\sum_{i,j,\Phi} P(z_i = j, \Phi|y_i)(\log \alpha_j + \sum_l \phi_{jl} (\log \rho_{jl} + \log p(y_{il}|\theta_{jl})) \\
&\quad + (1 - \phi_{jl}) (\log(1 - \rho_{jl}) + \log q(y_{il}|\lambda_{jl}))) \\
=&\sum_j (\sum_i P(z_i = j|y_i)) \log \alpha_j \\
&\quad + \sum_{jl} \sum_i P(z_i = j, \phi_{jl} = 1|y_i)(\log p(y_{il}|\theta_{jl}) + \log \rho_{jl}) \\
&\quad + \sum_{jl} \sum_i P(z_i = j, \phi_{jl} = 0|y_i) (\log q(y_{il}|\lambda_{jl}) + \log(1 - \rho_{jl}))
\end{aligned} \tag{4.10}$$

The probabilities are computed as follows,

$$P(z_i = j|y_i) = \frac{\alpha_j \prod_l [\rho_{jl} p(y_{jl}|\theta_{jl}) + (1 - \rho_{jl}) q(y_{jl}|\lambda_{jl})]}{\sum_j \alpha_j \prod_l [\rho_{jl} p(y_{jl}|\theta_{jl}) + (1 - \rho_{jl}) q(y_{jl}|\lambda_{jl})]} \tag{4.11}$$

$$P(z_i = j, \phi_{jl} = 1|y_i) = \frac{\rho_{jl} p(y_{jl}|\theta_{jl})}{\rho_{jl} p(y_{jl}|\theta_{jl}) + (1 - \rho_{jl}) q(y_{jl}|\lambda_{jl})} P(z_i = j|y_i) \tag{4.12}$$

$$P(z_i = j, \phi_{jl} = 0|y_i) = \frac{(1 - \rho_{jl}) q(y_{jl}|\lambda_{jl})}{\rho_{jl} p(y_{jl}|\theta_{jl}) + (1 - \rho_{jl}) q(y_{jl}|\lambda_{jl})} P(z_i = j|y_i) \tag{4.13}$$

M-step: Maximize the log-likelihood.

The three parts of Equation (4.10) can be maximized separately by updating the following quantities,

$$\widehat{\alpha_j} = \frac{\sum_i P(z_i = j|y_i)}{\sum_j \sum_i P(z_i = j|y_i)} \tag{4.14}$$

$$\widehat{\mu_{\theta_{jl}}} = \frac{\sum_i P(z_i = j, \phi_{jl} = 1|y_i) y_{jl}}{\sum_i P(z_i = j, \phi_{jl} = 1|y_i)} \tag{4.15}$$

$$\widehat{\sigma^2_{\theta_{jl}}} = \frac{\sum_i P(z_i = j, \phi_{jl} = 1|y_i)(y_{jl} - \widehat{\mu_{\theta_{jl}}})^2}{\sum_i P(z_i = j, \phi_{jl} = 1|y_i)} \tag{4.16}$$

$$\widehat{\mu_{\lambda_{jl}}} = \frac{\sum_i P(z_i = j, \phi_{jl} = 0|y_i) y_{jl}}{\sum_i P(z_i = j, \phi_{jl} = 0|y_i)} \tag{4.17}$$

$$\widehat{\sigma^2_{\lambda_{jl}}} = \frac{\sum_i P(z_i = j, \phi_{jl} = 0|y_i)(y_{jl} - \widehat{\mu_{\lambda_{jl}}})^2}{\sum_i P(z_i = j, \phi_{jl} = 0|y_i)} \tag{4.18}$$

$$\widehat{\rho_{jl}} = \frac{\sum_i P(z_i = j, \phi_{jl} = 1|y_i)}{\sum_i P(z_i = j, \phi_{jl} = 1|y_i) + \sum_i P(z_i = j, \phi_{jl} = 0|y_i)} \tag{4.19}$$

The EM algorithm alternates between the E-step, which computes an expectation of the likelihood by including the latent variables as if they were observed, and the M-step, which maximizes the expected likelihood found in the E-step. The parameters found in the M-step are then used to begin another iteration of the E-step, and the process is continued until the algorithm converges to a finite mixture model with feature saliency associated with each cluster. Thus, clustering and localized feature saliency detection is achieved simultaneously.

4.2.2 Model Selection Based on Minimum Message Length (MML)

Alternation of E and M steps in the above algorithm eventually results in a maximum likelihood estimate of Gaussian mixtures, which requires the number of clusters K as prior knowledge. To overcome this difficulty, we employ the MML criterion to detect the optimal number of clusters [1]. The MML criterion for our model with respect to θ is as follows,

$$\begin{aligned} J(\theta) = &- \log(\mathcal{Y}|\theta) + \frac{1}{2}(K + DK)\log(N) \\ &+ \frac{R}{2}\sum_{l=1}^{D}\sum_{j=1}^{K}\log(N\alpha_j\rho_{jl}) + \frac{S}{2}\sum_{l=1}^{D}\sum_{j=1}^{K}\log(N\alpha_j(1-\rho_{jl})) \end{aligned} \tag{4.20}$$

In the above equation, R and S are the number of parameters of $p(\cdot)$ and $q(\cdot)$, respectively, which for a Gaussian distribution is 2. Also, $-\log(\mathcal{Y}|\theta)$ corresponds to log-likelihood, and $\frac{1}{2}(K+DK)\log(N)$ represents the code-length of standard Message Description Length (MDL) of parameters α_js and ρ_{jl}s. While $N\alpha_j\rho_{jl}$ indicates the effective number of data for estimating θ_{jl}, $\frac{R}{2}\sum_{l=1}^{D}\sum_{j=1}^{K}\log(N\alpha_j\rho_{jl})$ computes the code-length corresponding to the parameters θ_{jl}. Similarly, $\frac{S}{2}\sum_{l=1}^{D}\sum_{j=1}^{K}\log(N\alpha_j(1-\rho_{jl}))$ represents the code-length for parameters λ_{jl}. The optimal mixture model is the one that minimizes the cost function $J(\theta)$ in Equation (4.20),

$$\hat{\theta} = \arg\min_{\theta}(J(\theta)) \tag{4.21}$$

The algorithm introduced above works well in general cases. However, extreme bad initialization may lead to some clusters with singular covariance matrices, and thus adversely affect the cost function $J(\theta)$. Those clusters can be pruned based on a modification of Equation (4.14) [1],

$$\widehat{\alpha_j} = \frac{\max\left(\sum_i P(z_i = j|y_i) - \frac{RD}{2}, 0\right)}{\sum_j \max\left(\sum_i P(z_i = j|y_i) - \frac{RD}{2}, 0\right)} \tag{4.22}$$

The effect of Equation (4.22) is that some small trivial components are quickly eliminated at an early stage. Similarly, Equation (4.19) is modified to,

$$\widehat{\rho_{jl}} = \frac{\max\left(\sum_i P(z_i = j, \phi_{jl} = 1|y_i) - \frac{R}{2}, 0\right)}{\max\left(\sum_i P(z_i = j, \phi_{jl} = 1|y_i) - \frac{R}{2}, 0\right) + \max\left(\sum_i P(z_i = j, \phi_{jl} = 0|y_i) - \frac{S}{2}, 0\right)} \tag{4.23}$$

The above Equation can prune ρ_{jl} to either 1 or 0.

In summary, the proposed EM clustering with localized feature saliency consists of the following steps,

1. Initialize the algorithm with a large value of K, minimal number of components K_{min},

and the parameter set θ.

2. Alternate between E-step and M-step until the model converges to a local maximum. During this step, components with $\alpha_j = 0$ are pruned.

3. Record the parameter set θ and the message length based on Equation (4.20).

4. Terminate the iterations if K equals K_{min}. Otherwise, reduce K to $K-1$ by removing the smallest component, and repeat steps (2) and (3).

5. Output the model with the smallest message length.

4.2.3 Computational Complexity

The computational load of the proposed algorithm is mainly due to the E and M steps. For every iteration, the complexity of both the steps is $\mathcal{O}(KND)$. The total computational time is dependent on the number of iterations required for converging. Conventional feature selection algorithms usually seek optimal features by trying out large number of combinations. On the other hand, the proposed algorithm computes the localized feature saliency simultaneously with clustering, thus avoiding the navigation over all possible feature subsets. It only needs to search over a small set of possible Ks.

4.3 Experimental Results

In general, the performance of an unsupervised feature selection algorithm is hard to be evaluated. Localized feature selection makes it even more difficult as we have an additional layer of complexity brought by the association of clusters to different feature subsets. In this section, we provide a thorough evaluation of the proposed algorithm by comparing it with the global feature selection approach [1] on both synthetic and real-world datasets. In addition, we show the need for feature selection in clustering and the benefits of selecting features locally through a case-study on Boston housing dataset.

Table 4.1: Summary of the synthetic datasets, where N represents the number of patters, D the number of features, c the number of clusters, D_j the number of relevant feature respecting to the j-th cluster, and N_j the size of the j-th cluster.

Dataset	N	D	c	D_j	N_j
syn_1	600	15	3	3/3/3	200/200/200
syn_2	600	20	3	3/4/5	200/200/200
syn_3	1000	20	5	3/4/5/4/2	200/200/200/200/200
syn_4	900	30	3	3/3/3	200/300/400

4.3.1 Synthetic Data

First, we applied both our method and the global feature selection algorithm to several synthetic datasets. As we know the underlying models from which the patterns were sampled, the performance of an algorithm is interpreted as: can the algorithm find the given model? The synthetic datasets are created by a data generator. It first generates c Gaussian components $\mathcal{N}(\mu_j, \Sigma_j)$, $j = 1, \cdots, c$, separately, where Σ_j is restricted to a diagonal matrix. Components can have different number of features D_j, and different number of patterns N_j. Those Gaussians are then embedded into subsets of a D-dimensional background with Gaussian noise $\mathcal{N}(0, I)$. Finally, a D-dimensional dataset consisting of c Gaussian mixtures, with each component corresponding to an individual relevant feature subset is generated. The total number of patterns is $N = \sum_{j=1}^{c} N_j$. Table 4.1 shows a summary of the four synthetic datasets generated.

In the experiments, we initialized the parameters as follows: number of clusters K is set to 20, the *a priori* probabilities α_j are set equally at $1/20$, the feature saliencies ρ_{jl} are set at 0.5, and the common components are set to cover the entire dataset. We ran the proposed algorithm 10 times independently with stopping threshold of 10^{-7}. The clustering error rates and cluster numbers are computed as the average over the 10 runs, and standard deviations are calculated accordingly. The feature saliency for each cluster at each run is mapped to a Gray-scale image, where each column represents a feature, and each row represents an individual run, as shown in Table 4.2. For all the four datasets, the proposed algorithm successfully detected the number of clusters. Each cluster and its relevant feature subset are also detected correctly. The Rey-

Table 4.2: Results on the synthetic datasets. Saliency in the range [0, 1] is mapped to gray-scale [0, 255] linearly. For the clustering with localized feature saliency, each image is a mapping of feature saliency of one cluster, where rows and columns of pixels represent runs and features, respectively. The separated row pixels above an image represent the true relevant features. The global feature saliency is illustrated in the same way.

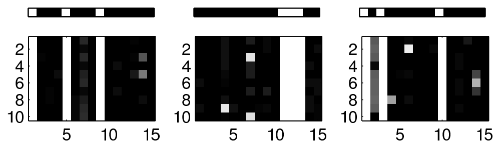
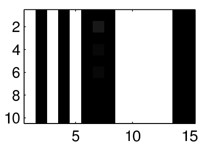
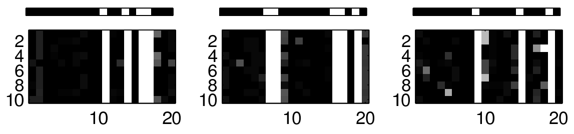
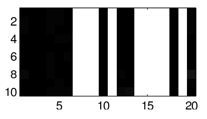
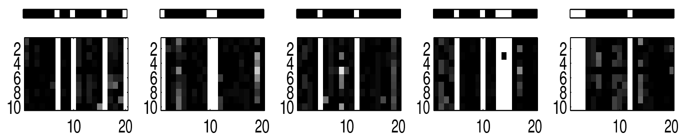
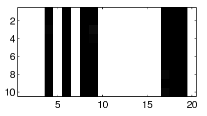
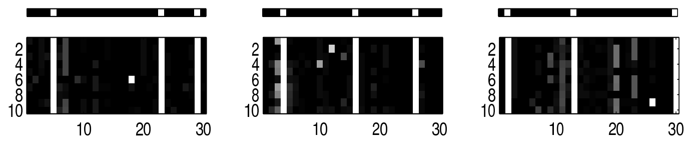
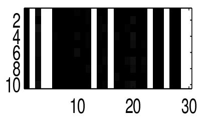
Data	Localized feature selection		Global feature selection	
	$\hat{c}$(std)	Saliency	$\hat{c}$(std)	Saliency
syn_1	3(0)		3(0)	
syn_2	3(0)		3(0)	
syn_3	5(0)		5(0)	
syn_4	3(0)		3(0)	

Table 4.3: Summary of UCI datasets

data	Description	N	D	c
wine	wine recognition	178	13	3
wdbc	Wisconsin diagnostic breast cancer	569	30	2
vehicle	vehicle classification	846	18	4
zernike	Zernike moments of digit images	2000	47	10

scale image is steady vertically, indicating that the algorithm is stable in different runs. In Table 4.2, we also show the performance of the global feature selection algorithm [1] on each of the datasets. We can see that the union of the localized feature subsets is equivalent to the relevant features selected by the global approach. Moreover, while global algorithm is able to detect the number of clusters correctly, it cannot determine if a salient feature really plays a critical role for a particular cluster. On the other hand, our approach yields more informative models, which not only provides information about whether a feature is relevant or not, but also about which cluster the feature is relevant or irrelevant to.

4.3.2 Real-world datasets

For the evaluation on real-world datasets, we utilized four datasets: *wine*, *wdbc*, *vehicle*, and *zernike*, from the UCI machine learning repository [70], having varying number of features, patterns, and categories. The *wine* dataset is used to recognize different wine types by 13 characters of chemical analysis. It consists of 178 patterns and 3 categories. The *wdbc* dataset is used to diagnose if a breast cancer is benign or malignant based on 30 features and contains 576 data points. The *vehicle* dataset contains 846 samples with 18 features extracted from vehicle silhouettes. The purpose is to classify a given silhouette as one of four types of vehicles. The *zernike* dataset records 47 zernike moments extracted from 2000 images of handwriting digits. Summary of these four datasets is shown in Table 4.3. The parameters are initialized in the same way as for the synthetic datasets, except that K is set at 30 for the *zernike* dataset.

The datasets are provided with class labels for supervised learning, which are excluded during the clustering process. We assign a class label to each final cluster afterward so that a

Table 4.4: Cluster numbers and pseudo error rates for UCI datasets.

data	Localized feature selection		Global feature selection	
	error (std)(%)	$\hat{c}$ (std)	error (std)(%)	$\hat{c}$ (std)
wine	2.1 (1.2)	3 (0)	2.4 (1.2)	3.3 (0.5)
wdbc	7.6 (0.6)	7.1 (0.7)	7.5 (1.2)	7.4 (0.8)
vehicle	44.6 (1.3)	9.2 (1.3)	45.4 (2.6)	10.5 (1.3)
zernike	44.9 (2.2)	15.3 (1.9)	47.6 (2.8)	16.7 (1.3)

pseudo error rate can be computed for evaluation purpose. The cluster label is simply selected as the class to which majority of patterns in the cluster belongs. In other words, we assume that each cluster consists of patterns from the same class. Comparing the cluster labels of all the patterns with the true class labels yields the pseudo error rate.

The estimated cluster numbers and pseudo error rates are shown in Table 4.4 for both local and global methods. It is clear that the proposed EM clustering with localized feature saliency generally outperforms the global one with lower error rates and variances. We also compared the feature saliency of the two algorithms as Frey-scale images in Table 4.5. Obviously, different clusters have different relevant feature subsets, which are usually smaller than the globally relevant feature subset. This result indicates that a globally relevant feature can be irrelevant to some clusters. Our experiments also show that a locally relevant feature might be treated as globally irrelevant. For example, the third feature of *wine* dataset is relevant to the first cluster (bright column), but, it has been ignored by the global feature selection algorithm (dark column). Thus, EM clustering with localized feature saliency provides users more accurate knowledge regarding the underlying model from which the cluster component is generated. Moreover, the vertical belt patterns in the Gorey-scale images demonstrate the stability of the proposed algorithm over different runs.

4.3.3 Boston Housing Dataset

In this section, we present a case study of the proposed algorithm on the Boston housing data from UCI [70], which contains 506 neighborhoods in the Boston metropolitan area with

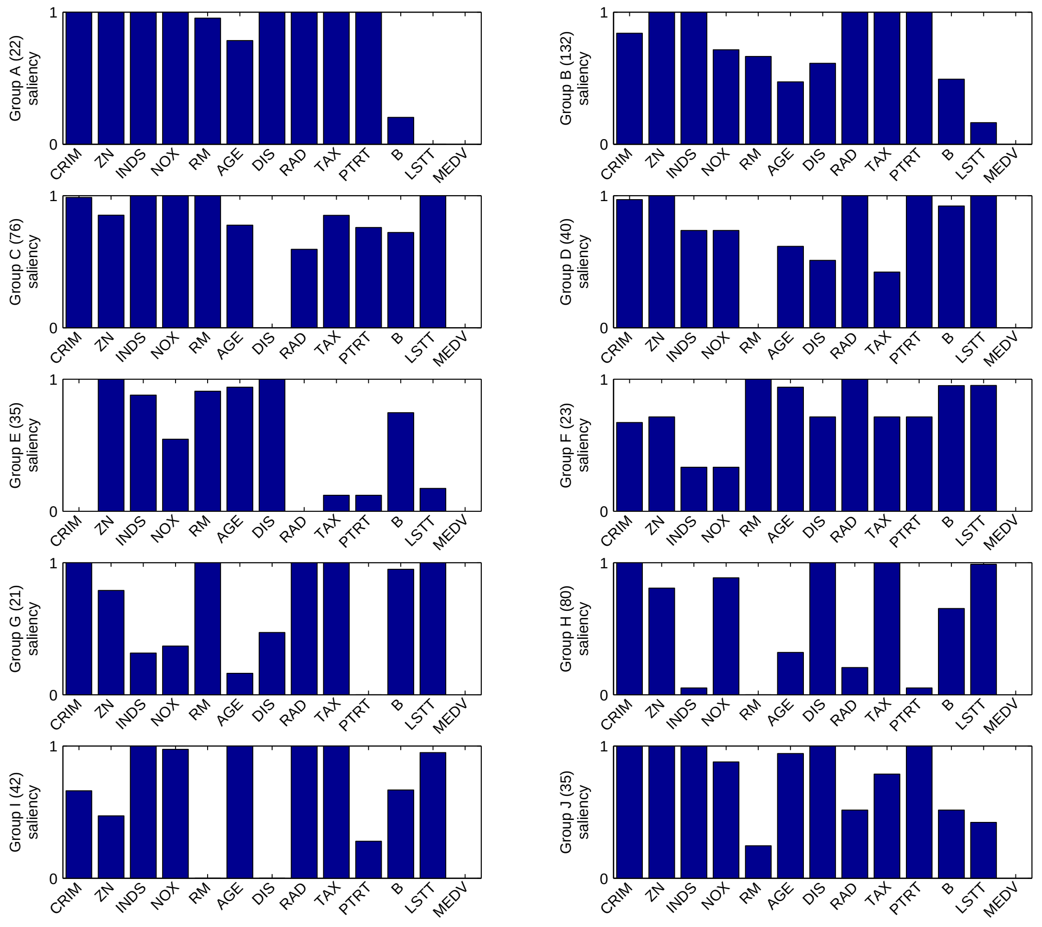


Figure 4.1: Localized feature saliency on the Boston housing dataset. The number of objects grouped together are listed with the group ID.

14 attributes, as described in Table 4.6. This dataset is often used as a test bed to compare the performance of prediction methods by estimating the value of the last attribute MEDV from the other 13 attributes. In our experiment, we remove the binary attribute CHAS, and consider the rest of the 13 attributes on an equal basis. Our goal is to find groups of neighborhoods based on these attributes, and to identify the saliency of attributes for each individual group.

In our experiment, the number of clusters are initialized to 20, and other parameters are initialized in the same way as for the synthetic datasets. As shown in Figure 4.1, 10 clusters are identified. Notice that the attribute saliency varies for each cluster. For example, attributes {CRIM, RAD, TAX, PTRT} are important to Group A but not to Group E, while attribute B is important to Group E but not to Group A. Figure 4.1 clearly shows that the distribution of feature saliency over the 13 attributes is quite different across clusters. Traditional clustering algorithms without feature selection or with global feature selection is not able to reveal these properties of the dataset. Our method, on the other hand, can provide this vital information to users through cluster-wise feature selection.

4.4 Summary

In this chapter, we proposed a EM clustering algorithm with localized feature saliency. In our approach, unsupervised feature selection is performed by estimating feature saliency of individual clusters simultaneously with the EM clustering. The determination of cluster number is also integrated in our method by adopting an MML criterion. Experimental results show that the cluster model produced by the proposed algorithm can provide users more accurate understanding of the underlying process which generates the data.

Table 4.5: Feature saliency. Each image is a mapping of feature saliency for a cluster, with exception that the highlighted one represents the global feature saliency. Saliency values [0,1] are linearly mapped to gray-scale [0,255]. Each row represents a run, and each column represents a feature.

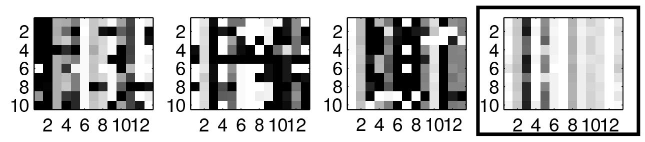
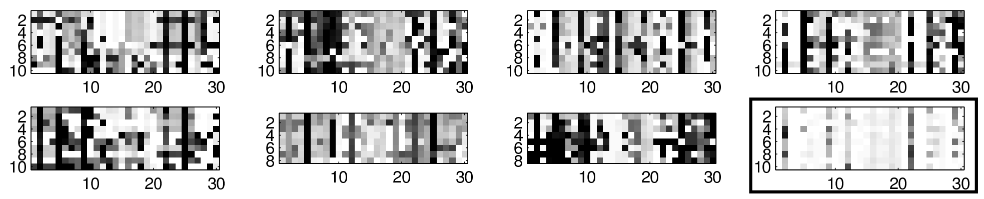
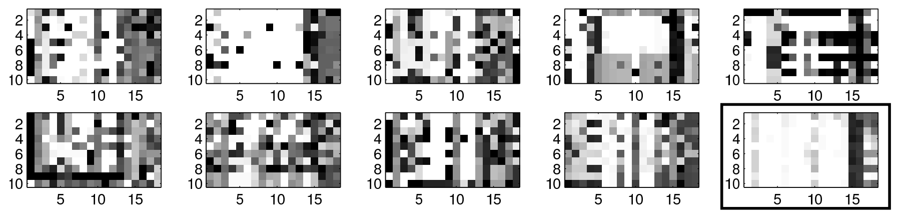
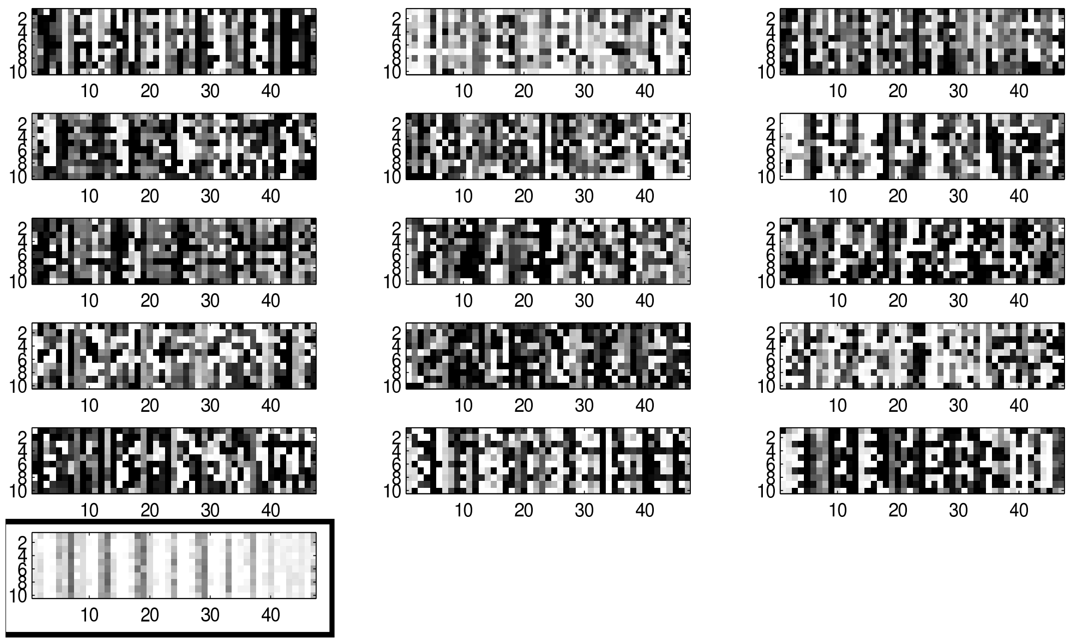
data	Feature saliency
wine	
wdbc	
vehicle	
zernike	

Table 4.6: Attributes for the Boston housing data.

Num.	Var.	Description
1	CRIM	per capita crime rate by town
2	ZN	land zoned for lots over 25,000 sq.ft.
3	INDS	proportion of non-retail business acres per town
4	CHAS	Charles River dummy variable
5	NOX	nitric oxides concentration
6	RM	number of rooms per dwelling
7	AGE	proportion of units built prior to 1940
8	DIS	distances to five Boston employment centers
9	RAD	accessibility to radial highways
10	TAX	full-value property-tax rate
11	PTRT	pupil-teacher ratio by town
12	B	$(Bk - 0.63)^2$ where Bk is the proportion of blacks
13	LSTT	% lower status of the population
14	MEDV	Median value of owner-occupied homes in $1000's

CHAPTER 5

SIMULTANEOUS LOCALIZED FEATURE SELECTION AND MODEL DETECTION FOR GAUSSIAN MIXTURES VIA VARIATIONAL LEARNING

The maximal likelihood (ML) with EM inference approach presented in Chapter 4 assumes that the parameters of Gaussian mixture are unknown but with fixed value. Those unknown parameters are estimated through EM iterations. The cluster number k has to be estimated by introducing some information criteria, for example, minimal message length (MML), or Bayesian information criterion (BIC). In this chapter, we propose a novel Bayesian inference approach [73] for simultaneous localized feature selection, where the parameters of Gaussian mixture are assumed following some particular probability distributions, instead of fixed values in ML approach. The distributions are estimated by variational Bayesian learning (VB) method. With a proper choice of prior probability over mixing coefficients, the cluster number k can be estimated through clustering process. Another problem encountered in ML is that singular components lead to infinite likelihood, which does not happen in VB.

5.1 Variational Approximation

We follow the Gaussian mixture model, with localized feature saliency, used in Chapter 4. Recall that the likelihood function is presented as,

$$p(\mathcal{Y}|\theta) = \prod_{i=1}^{N} \sum_{j=1}^{K} \pi_j \prod_{l=1}^{D} \big(\rho_{jl} p(y_{il}|\theta_{jl}) + (1-\rho_{jl}) q(y_{il}|\lambda_{jl})\big), \tag{5.1}$$

where $\theta = \{\{\pi_j\}, \{\theta_{jl}\}, \{\rho_{jl}\}, \{\lambda_{jl}\}\}$ is the set of all the parameters. In general, to evaluate the likelihood of mixtures, conditioned on the mixing coefficients, we must marginalize the parameters as follows,

$$P(\mathcal{Y}|\theta) = \int P(\mathcal{Y}, \Theta|\theta)\, d\Theta, \tag{5.2}$$

where $\Theta \equiv \{\theta, z, Z, S\}$ denotes all the parameters and latent variables. The integral sign represents the joint integral over θ and the summation over z and S. This integral is analytically intractable. Therefore, we use variational methods to find a lower bound for $P(\mathcal{Y}|\pi)$.

Consider the following transformation applied to the log marginal likelihood,

$$\ln P(\mathcal{Y}|\theta) \geq \int Q(\Theta) \ln \frac{P(\mathcal{Y}, \Theta|\theta)}{Q(\Theta)}\, d\Theta = \mathcal{L}(Q). \tag{5.3}$$

The function $\mathcal{L}(Q)$ forms a rigorous lower bound on the true log marginal likelihood. Through a suitable choice of the Q distribution, the quantity $\mathcal{L}(Q)$ may be tractable to compute. From Equation (5.3), the difference between the true log likelihood $\ln P(\mathcal{Y}|\pi)$ and the bound $\mathcal{L}(Q)$ is given by Kullback-Leibler divergence KL$(Q||P)$. $Q(\Theta)$ is chosen from some family of distributions such that the lower bound $\mathcal{L}(Q)$ is sufficiently simplified for evaluation. Since the true log likelihood is independent of Q, we approximate $P(\Theta)$ with $Q(\Theta)$ by minimizing the KL divergence. Assuming that $Q(\Theta)$ factorizes over subsets $\{\Theta_i\}$ of the variables in Θ,

$$Q(\Theta) = \prod_i Q_i(\Theta_i), \tag{5.4}$$

the KL divergence can then be minimized over all possible factorial distributions by performing free-form minimization over Q_i,

$$Q_i(\Theta_i) = \frac{\exp\langle \ln P(\mathcal{Y}, \Theta)\rangle_{k\neq i}}{\int \exp\langle \ln P(\mathcal{Y}, \Theta)\rangle_{k\neq i}\, d\Theta_i}, \tag{5.5}$$

where $\langle\cdot\rangle_{k\neq i}$ denotes an expectation with respect to the distributions $Q_k(\Theta_k)$ for all $k \neq i$. Equation (5.5) shows that the sufficient statistics of each distribution Q_i depends on the moments of other distributions $Q_{k\neq i}$, which implies an iterative solution for the estimation of the variational variables. In other words, with a sufficient parameter initialization, the statistics can be updated by taking each factor in turn and replacing its sufficient statistics with the revised estimates. In each iteration of the re-estimation process, the KL divergence is reduced, while both the lower bound, $\mathcal{L}(Q)$, and the likelihood are increased. Hence, the convergence is guaranteed.

5.2 Local feature saliency with variational learning

We now apply Bayesian variational approach to the mixture of Gaussians with localized feature saliency. Given the sets of hidden variables $Z = \{z_j^{(i)}\}$ and $S = \{s_{jl}^{(i)}\}$, the distribution of the Gaussian mixture is

$$p(\mathcal{Y}|Z, S, \mu, T, \epsilon, \gamma) = \prod_{i=1}^{N}\prod_{j=1}^{K}\Big[\prod_{l=1}^{D}\big(p(y_{il}|\mu_{jl}, \tau_{jl})\big)^{s_{jl}^{(i)}}\big(q(y_{il}|\epsilon_{jl}, \gamma_{jl})\big)^{1-s_{jl}^{(i)}}\Big]^{z_j^{(i)}}, \tag{5.6}$$

where $\mu = \{\mu_{jl}\}$ and $T = \{\tau_{jl}\}$ denote the means and inverse variances of the “useful” subcomponents, while $\epsilon = \{\epsilon_{jl}\}$ and $\gamma = \{\gamma_{jl}\}$ are the sets of parameters for the “noisy” subcomponents. The distribution of the hidden variable Z (given the mixing probabilities $\pi = \{\pi_j\}$) and the distribution of the hidden variable S (given the mixing probabilities $\rho = \{\rho_{jl}\}$) are governed as,

$$P(Z|\pi) = \prod_{i=1}^{N}\prod_{j=1}^{K}\pi_j^{z_{ij}}, \tag{5.7}$$

$$P(S|\rho) = \prod_{i=1}^{N}\prod_{j=1}^{K}\prod_{l=1}^{D}\rho_{jl}^{s_{jl}^{(i)}}\left(1-\rho_{jl}\right)^{1-s_{jl}^{(i)}}. \tag{5.8}$$

In order to accomplish model selection, the above Bayesian model is augmented with conjugate priors over the means and inverse covariances,

$$P(\mu) = \prod_{j=1}^{K} \prod_{l=1}^{D} \mathcal{N}(\mu_{jl} | m_l, c), \tag{5.9}$$

$$P(T) = \prod_{j=1}^{K} \prod_{l=1}^{D} \Gamma(\tau_{jl} | \alpha, \beta), \tag{5.10}$$

where $\Gamma(\cdot)$ is the gamma distribution, m_l, c, α, and β are hyperparameters that control the prior distributions. The hyperparameters are chosen such that the prior distribution is broad enough to cover the whole dataset. Since the actual model parameters are represented by the means of the corresponding distributions, they are not sensitive to these hyperparameters. For conjugate hierarchical models, the expressions on the right side of Equation (5.5) will have the same functional forms as in the priors.

In the following, we show the derivations of $Q_Z(Z)$, $Q_\mu(\mu)$, $Q_T(T)$, and $Q_S(S)$.

Proposition 4. *Suppose $Q(\Theta)$ can be factorized as $Q(\Theta) = Q_Z(Z) Q_S(S) Q_\mu(\mu) Q_T(T)$, the update functions of $Q_Z(Z)$,$Q_S(S)$ $Q_\mu(\mu)$, and $Q_T(T)$, are given by*

$$Q_Z(Z) = \prod_{i=1}^{N} \prod_{j=1}^{K} r_{ij}^{z_{ij}}, \tag{5.11}$$

$$Q_\mu(\mu) = \prod_{j=1}^{K} \prod_{l=1}^{D} \mathcal{N}(\mu_{jl} | m_{jl}^v, c_{jl}^v), \tag{5.12}$$

$$Q_T(T) = \prod_{j=1}^{K} \prod_{l=1}^{D} \Gamma(\tau_{jl} | \alpha_{jl}^v, \beta_{jl}^v), \tag{5.13}$$

$$Q_S(S) = \prod_{i=1}^{N} \prod_{j=1}^{K} \prod_{l=1}^{D} \omega_{ijl}^{s_{jl}^{(i)}} (1 - \omega_{ijl})^{1 - s_{jl}^{(i)}}, \tag{5.14}$$

where $r_{ij}, m_{jl}^v, c_{jl}^v, \alpha_{jl}^v, \beta_{jl}^v$, and ω_{ijl} are variational parameters for maximization and determin-

ing the density involved in Q, defined by

$$
\begin{aligned}
r_{ij} =& \frac{\pi_j \tilde{r}_{ij}}{\sum_{j=1}^{K} \pi_j \tilde{r}_{ij}}, && (5.15)\\
\tilde{r}_{ij} =& \exp\left\{\frac{1}{2}\sum_{l=1}^{d} \omega_{ijl}\left[\psi(\alpha_{jl}^v) - \log\beta_{jl}^v - \frac{\alpha_{jl}^v}{\beta_{jl}^v}((y_l^i - m_{jl}^v)^2 + \frac{1}{c_{jl}^v})\right]\right\}, && (5.16)\\
m_{jl}^v =& \frac{cm_i + (\alpha_{jl}^v/\beta_{jl}^v)\sum_{i=1}^{n} r_{ij}\omega_{ijl}y_l^i}{c + (\alpha_{jl}^v/\beta_{jl}^v)\sum_{i=1}^{n} r_{ij}\omega_{ijl}}, && (5.17)\\
c_{jl}^v =& c + \frac{\alpha_{jl}^v}{\beta_{jl}^v}\sum_{i=1}^{N} r_{ij}\omega_{ijl}, && (5.18)\\
\alpha_{jl}^v =& \alpha + \frac{1}{2}\sum_{i=1}^{N} r_{ij}\omega_{ijl}, && (5.19)\\
\beta_{jl}^v =& \beta + \frac{1}{2}\sum_{i=1}^{N} r_{ij}\omega_{ijl}[(y_l^i - m_{jl}^v)^2 + \frac{1}{c_{jl}^v}], && (5.20)\\
\omega_{ijl} =& \frac{\rho_{jl}\tilde{\omega}_{ijl}}{\rho_{jl}\tilde{\omega}_{ijl}} + (1-\rho_{jl}), && (5.21)\\
\tilde{\omega}_{ijl} =& \exp\left\{\frac{1}{2}r_{ij}\left[\psi(\alpha_{jl}^v) - \log\beta_{jl}^v - \frac{\alpha_{jl}^v}{\beta_{jl}^v}((y_l^i - m_{jl}^v)^2 + \frac{1}{c_{jl}^v})\right]\right\}, && (5.22)\\
\xi_{ijl} =& \exp\{-\frac{1}{2}\gamma_{jl}(y_l^i - \epsilon_{jl})^2 + \frac{1}{2}\log\gamma_{jl}\}, && (5.23)
\end{aligned}
$$

where $\psi(x)$ is the digamma *function $\psi(x) = d\log\Gamma(x)/dx$.*

Proof. Applying Equation (5.5) on $Q_Z(Z)$ and taking logarithm on both sides yields

$$\ln Q_z(Z) = \langle \ln p(\mathcal{Y}, \mathcal{S}, \theta)\rangle + const., \tag{5.24}$$

where $\langle\cdot\rangle$ represents the expectation of S and θ. Absorbing any terms that do not depend on Z into the additional normalization constant, we have

$$\ln Q_z(Z) = \langle \ln p(Z|\pi)\rangle_\pi + \langle \ln p(\mathcal{Y}|Z, S, \theta)\rangle_\theta + const. \tag{5.25}$$

Substituting the two terms on the right side by Equation (5.6) and (5.7), and absorbing any term that are independent of Z, we get

$$\begin{aligned}\ln Q_Z(Z) =& \left\langle \sum_{i=1}^{N}\sum_{j=1}^{K} z_{ij} \ln \pi_j \right\rangle_{\pi} + \left\langle \sum_{i=1}^{N}\sum_{j=1}^{K} z_{ij} \left[s_{jl}^{(i)} p(y_{il}|\theta_{jl}) + (1 - s_{jl}^{(i)}) q(y_{il}|\lambda_{jl}) \right] \right\rangle_{\theta} + const. \\ =& \sum_{i=1}^{N}\sum_{j=1}^{K} z_{ij} \ln \tilde{r}_{ij} + const., \end{aligned} \tag{5.26}$$

where $\tilde{r}_{ij}$ has the form of

$$\tilde{r}_{ij} = \exp\left\{ \frac{1}{2} \sum_{l=1}^{d} \omega_{ijl} \left[\psi(\alpha_{jl}^{v}) - \log \beta_{jl}^{v} - \frac{\alpha_{jl}^{v}}{\beta_{jl}^{v}} ((y_l^i - m_{jl}^{v})^2 + \frac{1}{c_{jl}^{v}}) \right] \right\}. \tag{5.27}$$

Note that for each value of i, the quantities $\langle z_{ij} \rangle$ are binary and sum to 1. $Q_Z(Z)$ can be normalized to

$$Q_Z(Z) = \prod_{i=1}^{N}\prod_{j=1}^{K} r_{ij}^{z_{ij}}, \tag{5.28}$$

where $r_{ij}^{z_{ij}}$ is computed by

$$r_{ij} = \frac{\pi_j \tilde{r}_{ij}}{\sum_{j=1}^{K} \pi_j \tilde{r}_{ij}} \tag{5.29}$$

Similarly, applying Equation (5.14) to Equation (5.5), we have

$$\begin{aligned}
\ln Q_S(S) =& \langle \ln p(S|\rho)\rangle_\rho + \langle \ln p((Y|Z,S,\theta)\rangle_\theta + const. \\
=& \left\langle \sum_{i=1}^{N}\sum_{j=1}^{K}\sum_{l}^{D}\left[s_{jl}^{(i)}\ln\rho_{jl} + (1-s_{jl}^{(i)})\ln(1-\rho_{jl})\right]\right\rangle_\rho + \\
& \left\langle \sum_{i=1}^{N}\sum_{j=1}^{K} z_{ij}\left[s_{jl}^{(i)} p(y_{il}|\theta_{jl}) + (1-s_{jl}^{(i)}) q(y_{il}|\lambda_{jl})\right]\right\rangle_\theta + const. \\
=& \sum_{i=1}^{N}\sum_{j=1}^{K}\sum_{l=1}^{D}\left\{s_{jl}^{(i)}\ln\rho_{jl}\tilde{\omega}_{ijl} + (1-s_{jl}^{(i)})\ln(1-\rho_{jl})(1-\tilde{\omega}_{ijl})\right\} + const., \qquad (5.30)
\end{aligned}$$

where $\tilde{\omega}_{ijl}$ has the form of

$$\tilde{\omega}_{ijl} = \exp\left\{\frac{1}{2}r_{ij}\left[\psi(\alpha_{jl}^{v}) - \log\beta_{jl}^{v} - \frac{\alpha_{jl}^{v}}{\beta_{jl}^{v}}((y_l^i - m_{jl}^{v})^2 + \frac{1}{c_{jl}^{v}})\right]\right\}$$

Thus, $Q_S(S)$ has the form of

$$Q_S(S) \propto \prod_{i=1}^{N}\prod_{j=1}^{K}\prod_{l=1}^{D}\left[\rho_{jl}\tilde{\omega}_{ijl}\right]^{s_{jl}^{(i)}}\left[(1-\rho_{jl})(1-\tilde{\omega}_{ijl})\right]^{1-s_{jl}^{(i)}}. \qquad (5.31)$$

Normalizing $Q_S(S)$ yields to

$$\begin{aligned}
Q_S(S) =& \prod_{i=1}^{N}\prod_{j=1}^{K}\prod_{l=1}^{D}\omega_{ijl}^{s_{jl}^{(i)}}(1-\omega_{ijl})^{1-s_{jl}^{(i)}} \\
\omega_{ijl} =& \frac{\rho_{jl}\tilde{\omega}_{ijl}}{\rho_{jl}\tilde{\omega}_{ijl}} + (1-\rho_{jl})
\end{aligned}$$

Applying Equation (5.5) to $Q_\mu(\mu)$ yields

$$\ln Q_\mu(\mu) = const. + \left\langle \sum_{j=1}^{K}\sum_{l=1}^{D} \ln p(\mu_{jl}) \right\rangle + \langle \ln p(Z|\pi)\rangle_Z + \left\langle \sum_{i=1}^{N}\sum_{j=1}^{K}\sum_{l=1}^{D} z_j^{(i)} \left[s_{jl}^{(i)} \ln p(y_{ijl}|\mu_{jl},\theta_{jl}) + (1-s_{jl}^{(i)}) \ln p(y_{ijl}|\lambda_{jl}) \right] \right\rangle$$

This leads to a Gaussian distribution

$$Q_\mu(\mu) = \prod_{j=1}^{K}\prod_{l=1}^{D} \mathcal{N}(\mu|m_{jl}^v, c_{jl}^v), \tag{5.32}$$

where m_{jl}^v and c_{jl}^v have the form of

$$m_{jl}^v = \frac{cm_i + (\alpha_{jl}^v/\beta_{jl}^v)\sum_{i=1}^{n} r_{ij}\omega_{ijl}y_l^i}{c + (\alpha_{jl}^v/\beta_{jl}^v)\sum_{i=1}^{n} r_{ij}\omega_{ijl}},$$
$$c_{jl}^v = c + \frac{\alpha_{jl}^v}{\beta_{jl}^v}\sum_{i=1}^{N} r_{ij}\omega_{ijl}$$

For further details on the derivation of variational learning, readers may refer to [74]. $\square$

The model parameters $\pi_j, \rho_{jl}, \epsilon_{jl}$, and γ_{jl} are given by the mean values of corresponding variational factors:

$$\pi_j = \frac{1}{N}\sum_{i=1}^{N} r_{ij}, \tag{5.33}$$

$$\rho_{jl} = \frac{1}{N}\sum_{i=1}^{N} \omega_{ijl}, \tag{5.34}$$

$$\epsilon_{jl} = \frac{\sum_{i=1}^{N} \omega_{ijl}y_l^i}{\sum_{i=1}^{N} \omega_{ijl}}, \tag{5.35}$$

$$\frac{1}{\gamma_{jl}} = \frac{\sum_{i=1}^{N} \omega_{ijl}(y_l^i - \epsilon_{jl})^2}{\sum_{i=1}^{N} \omega_{ijl}}. \tag{5.36}$$

The above steps iterate alternatively until convergence. This model has a property that the components with similar parameters fitting the same Gaussian will compete with each other, yielding a dominant cluster. Thus, we can initialize the model with a large number of clusters, and eliminate the trivial clusters during iteration. Finally, the algorithm will produce a model with localized feature saliency represented by ρ_{jl} and identify the number of clusters simultaneously.

One should notice that seeking the feature saliency for individual clusters introduces more parameters than global feature selection approaches. Consequently, this increases the potential risks posed by local extrema. To this end, variational learning is a better choice for the optimization than EM. Unlike EM, which assumes an unknown but fixed value for a parameter, variational learning formulates the model parameters as distributions. Chandan et al. [75] showed that EM might yield poor clusters with improperly initialized parameters. In our proposed algorithm, the variational parameters are initialized based on broad distributions. In addition, the estimated model parameters are represented by the means of the corresponding approximation functions. Therefore, variational learning can provide robust and stable optimization results, and can also alleviate the overfitting problem, often suffered by EM.

5.3 Computational Complexity

The computational complexity of the proposed algorithm is $\mathcal{O}(NDK)$ in each iteration. The total computational time depends on the number of iterations required for converging. Specifically, in each iteration, we have to compute measures in Equations (5.15)-(5.36). Computing ξ_{ijl} is $\mathcal{O}(1)$. There are (NDK) ξs so that it requires $\mathcal{O}(NDK)$. Similarly, ω and $\tilde{\omega}$ require $\mathcal{O}(NDK)$. Computing α_{jl}^{v} requires to navigate through all the samples, resulting in the complexity $\mathcal{O}(NDK)$. Similar results can be obtained for β^{v}, c^{v}, m^{v}, and $\tilde{r}$. For r, the complexity is $\mathcal{O}(NK)$, since the summation of Equation (5.15) can be re-used. The complexity for ρ, ϵ, and γ is $\mathcal{O}(NDK)$. For p, it is $\mathcal{O}(NK)$. In summary, the overall computational complexity for one iteration is $\mathcal{O}(NDK)$.

5.4 Advantages of the proposed approach

The proposed method integrates localized feature selection, model detection and clustering into a unified framework. Its major advantages are summarized as follows,

1. Compared with global methods, our method can reveal cluster-wise feature relevance, hence providing users more accurate information about the underlying model which generates the data.

2. Compared with subspace clustering methods, our method does not require users to provide values of the parameters that are critical but almost impossible to be set in advance, for example, the number of clusters, the density threshold, or the desired dimensionality.

3. Our method avoids heuristic navigation through the large pool of possible feature subsets. The computational cost for each iteration of the proposed algorithm is $\mathcal{O}(NDK)$. It does not grow exponentially with D or N. Therefore, our method is scalable to large datasets.

5.5 Experimental Results

In general, the performance of an unsupervised feature selection algorithm is difficult to be evaluated. Localized feature selection makes it even more difficult as we have an additional layer of complexity brought by the association of clusters with different feature subsets. To thoroughly evaluate the proposed Localized Feature Selection with Variational Bayesian (LFSVB) algorithm, we have compared it with the leading unsupervised feature selection methods on both synthetic and real-world datasets. Specifically, in the comparison, we choose a global method proposed in [13], which is also based on the Bayesian framework with variational learning (GFSVB). In addition, we have selected a recently published and well-accepted subspace clustering method, viz., COSA [64]. Unlike other subspace clustering approaches that usually yield only hard-decisions (either accept or reject a feature), COSA can produce soft feature saliency (feature importance), similar to our approach, and thus make the comparison more meaningful. Note that COSA software is publicly available [1]. Finally, we also compare our approach with the parsimonious model with Gaussian mixtures (PMGM) [76].

5.5.1 Synthetic Data

5.5.1.1 Synthetic datasets with hard feature saliency

First, we have applied the four algorithms (LFSVB, GFSVB, COSA and PMGM) to 100 synthetic datasets with 0-1 (hard) feature saliency (a feature is either relevant or irrelevant). As we know the underlying model from which the patterns were sampled, the performance of an algorithm is assessed through whether the algorithm can find the given model. The synthetic datasets are created by a data generator. It first generates c Gaussian components $\mathcal{N}(\mu_{\mathbf{j}}, \mathbf{\Sigma_j})$, $j = 1, \cdots, c$, separately, where $\mathbf{\Sigma_j}$ is restricted to a diagonal matrix. The values of $\mu_{\mathbf{j}}$ are chosen randomly from -4 to 4 and from 0.1 to 0.3 for Σ_j. Components can have different numbers of features D_j, and different numbers of patterns N_j. Those Gaussians are then embedded into subsets of a D-dimensional background with Gaussian noise $\mathcal{N}(\mathbf{0}, \mathbf{I})$. Specifically, we randomly select D_1 features from the background data, and replace the first N_1 positions with component 1. This embeds the first component into the background. Similarly, we can embed all the rest clusters into the background. Finally, a D-dimensional dataset consisting of c Gaussian mixtures, with each component corresponding to an individual relevant feature subset, is generated. The total number of patterns is $N = \sum_{j=1}^{c} N_j$. In our experiment, one hundred datasets are generated with dimensionality (D) varying from 10 to 200, the number of salient features (D_j) from 1 to 8, the cluster size (N_j) from 100 to 500, and the number of clusters from 3 to 7.

We initialize LFSVB with $k = 20$. The global feature selection approach is initialized in the same manner. COSA is initialized with default settings. COSA-distance matrix is computed, and then processed by hierarchical clustering. Clusters are manually selected based on the visual inspection of the dendrogram. Feature importance is normalized so that the value of the

[1]`http://www-stat.stanford.edu/~jhf/COSA.html`

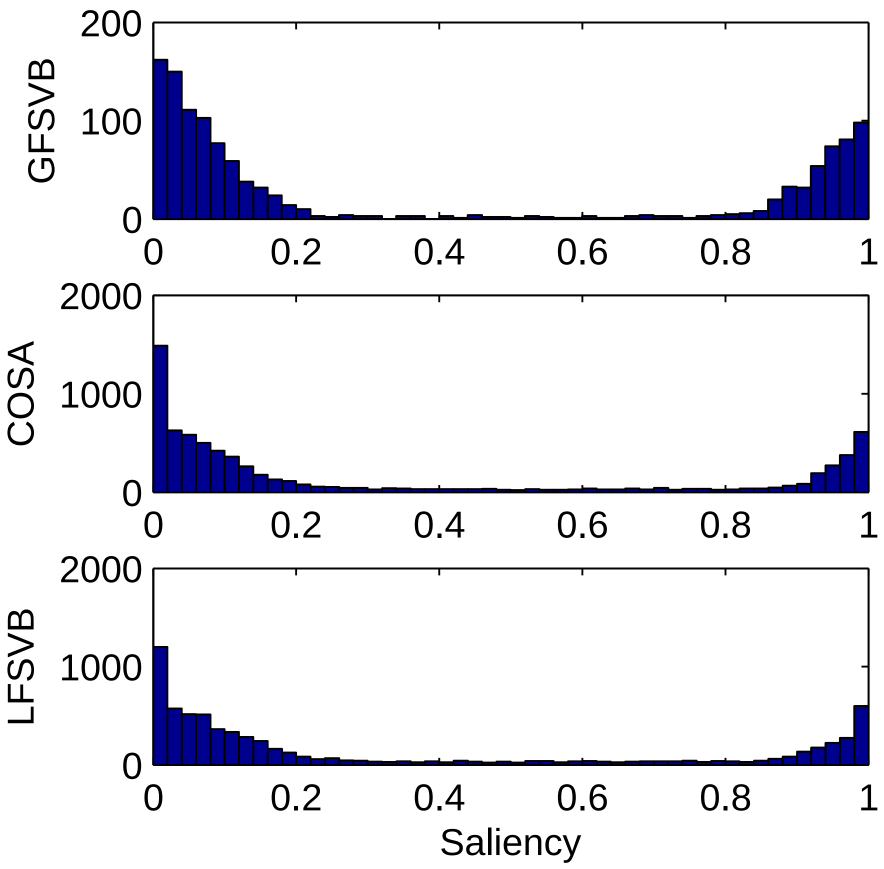


Figure 5.1: Histograms of feature saliency on 100 synthetic datasets for GFSVB (upper panel), COSA (middle panel), and LFSVB (lower panel), respectively.

top-rank-feature is 1.

Note that, PMGM produces binary feature weights (either 0 or 1), while the other three algorithms yield feature weight in the range of [0,1]. To evaluate the performance of the algorithms for feature selection, we need to find a cut-off threshold of feature saliency for LFSVB, GFSVB, and COSA. Figure 5.1 shows the histograms of the feature saliency obtained by GFSVB, COSA, and LFSVB, respectively, for all the clusters in the 100 datasets. Clearly, the saliency values are mainly distributed in the range of [0,0.2] and [0.8,1]. In the following experiments, we simply choose 0.5 as the cut-off threshold for the three algorithms.

We compute four quantities to evaluate the performance of the algorithms: (1) accuracy

Table 5.1: Statistical summary on 100 synthetic datasets, where $\overline{ACN}$ is the average accuracy of cluster number, $\overline{CA}$ is the average clustering accuracy, $\overline{FP_j}$ is the average feature precision, and $\overline{FR_j}$ is the average feature recall. For COSA, the number of clusters $(\hat{c})$ is set manually with visual inspection of the dendrogram (denoted by *).

algorithm	$\overline{ACN}$	$\overline{CA}$	$\overline{FP_j}$	$\overline{FR_j}$
GFSVB	0.952 (0.015)	0.922 (0.020)	0.384 (0.086)	0.941 (0.022)
COSA	0.992* (0.03)	0.933 (0.011)	0.892 (0.015)	0.897 (0.021)
LFSVB	0.980 (0.017)	0.910 (0.023)	0.925 (0.017)	0.950 (0.025)
PMGM	0.983 (0.017)	0.914 (0.026)	0.920 (0.022)	0.945 (0.018)

Table 5.2: Experimental results on synthetic dataset (syn_0) with hard feature saliency. For COSA, the number of clusters $(\hat{c})$ is set manually with visual inspection of the dendrogram (denoted by *). *Truth* indicates the actual model which generates the dataset. C1,C2, and C3 represent different clusters.

Data	Algo.	$\hat{c}$	accuracy	Feature subset
syn_0 $D = 30$	Truth	3	-	C1:[8, 19, 30],C2:[5, 23, 24], C3:[7, 16, 26]
	LFSVB	3	99.2%	C1:[8, 19, 30], C2:[5, 23, 24], C3:[7, 16, 26]
	COSA	3*	98.5%	C1:[8, 19, 30], C2:[5, 23, 24], C3:[7, 16]
	GFSVB	3	98.3%	[5, 8, 16, 23, 24, 26, 30]
	PMGM	3	99.0%	C1:[8, 19, 30], C2:[5, 23, 24], C3:[7, 16, 26]

of cluster number $ACN = \frac{|\hat{c}-c|}{c}$, where $\hat{c}$ is the estimated number of clusters and c is the true value; (2) clustering accuracy $CA = 1 - \frac{\tilde{N}}{N}$, where $\tilde{N}$ is the number of mis-clustered samples; (3) feature precision $FP_j = \frac{|\hat{D}_j \bigcap D_j|}{|\hat{D}_j \bigcup D_j|}$, where $\hat{D}_j$ and D_j are the estimated and true feature subset for cluster j, respectively, and $|\cdot|$ represents the set length; and (4) feature recall $FR_j = \frac{|\hat{D}_j \bigcap D_j|}{|D_j|}$. The statistical summary over the 100 synthetic datasets are reported in Table 5.1, while an example is provided in Table 5.2, showing the results for the synthetic dataset (syn_0) with 30 features and 3 clusters.

Compare to global feature selection. From the example in Table 5.2, we can see clearly that the proposed algorithm correctly detects the underlying clusters as well as the feature subsets corresponding to each cluster. On the other hand, GFSVB yields a feature subset which is close to the union of feature subsets identified by LFSVB, except that feature 19 is missing. Table 5.1 shows that, over the 100 synthetic datasets, LFSVB yields higher accuracy than GFSVB on cluster number estimation. The cluster accuracy of GFSVB is slightly higher than that of LFSVB. However, both the algorithms can discover the clusters very well. The feature recall measure of GFSVB is high, but the feature precision measure of GFSVB is low, while both values of LFSVB are high. This indicates that the global feature selection algorithm can detect if a feature is relevant to the dataset, however, it can not determine if a salient feature really plays a critical role on a particular cluster. On the other hand, the proposed model not only provides information on whether a feature is relevant or not, but also shows which cluster the feature is relevant or irrelevant to.

Compare to subspace clustering. As an example, Table 5.2 shows that localized feature subsets for $C1$ and $C2$ are correctly identified by COSA. It misses a salient feature (feature 26) for cluster 3, while LFSVB can recognize all three clusters with the corresponding feature subsets. The overall cluster accuracy of COSA is slightly better than that of LFSVB, while LFSVB outperforms COSA on feature precision and feature recall, as shown in Table 5.1. Moreover, COSA only produces a COSA-distance matrix and requires other clustering algorithms for subsequent processing. The number of clusters is also required as an input. On the other hand, our method provides a fully-automated solution by integrating localized feature selection, model detection, and clustering into a unified framework.

Compare to parsimonious model with Gaussian mixture. The example results shown in Table 5.2 and the statistical measures shown in Table 5.1 indicate that the proposed algorithm performs equivalently to PMGM. Notice that PMGM yields binary feature weight (either 0 or 1), while our algorithm produces feature saliency as a probability measure in the range of [0,1]. Subsequently, the proposed method can be applied for both feature selection and feature evaluation.

5.5.1.2 Synthetic dataset with soft feature saliency

The feature saliency in real-world datasets is usually soft, which means that it can be any value between 0 or 1. To approximate this situation, we generate a 20-dimensional dataset where the feature saliency is distributed in the range of $[0, 1]$. This dataset contains 2 Gaussian components $\mathcal{N}(\mu_{\mathbf{1}}, \mathbf{\Sigma_1})$ and $\mathcal{N}(\mu_{\mathbf{2}}, \mathbf{\Sigma_2})$, where $\mu_{\mathbf{1}} = (0.5, \ldots, 0.5)$, $\mu_{\mathbf{2}} = (-0.5, \ldots, -0.5)$, $\mathbf{\Sigma_1}$ and $\mathbf{\Sigma_2}$ are both diagonal, having $(0.2, \ldots, 0.2)$ on the diagonal terms. The feature saliency of clusters 1 and 2 are $(0.05, 0.10, \ldots, 1)$ and $(1, 0.95, \ldots, 0.05)$, respectively. Each component contains 500 points. We generate the data based on the procedure described in Section III with a common distribution of $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

We run the four algorithms on this dataset 10 times. Both LFSVB and PMGM detect two clusters successfully, while GFSVB yields 3 clusters. For COSA, we manually select the clusters. Table 5.3 shows the feature saliency obtained by LFSVB, COSA, PMGM, and GFSVB, respectively. We can see that GFSVB determines that all feature saliency is greater than 0.5. PMGM can discover that the two clusters have different relevant feature subsets. However, it does not obtain the true feature saliency due to its binary coding scheme. On the other hand, LFSVB and COSA correctly discover that feature relevance associated to cluster 1 is different from that of cluster 2. Specifically, the relevance of features increases with feature index for cluster 1, but decreases for cluster 2. This provides additional and more accurate information than GFSVB and PMGM.

5.5.2 Real-world datasets

For the evaluation on real-world datasets, we utilized six datasets: *Heart*, *Ion*, *Vehicle*, *Wine*, *WDBC*, and *Yeast*, from the UCI machine learning repository [70], with varying number of features, patterns, and categories, as summarized in Table 5.4. Class labels are provided in the datasets for supervised learning, which are excluded during the clustering process. A confusion matrix is computed according to the true class labels and the cluster index. Based on confusion matrix, mutual information is calculated as

$$I(\mathcal{X}; \mathcal{Y}) = -\sum_{\mathcal{X}} \sum_{\mathcal{Y}} p(x, y) \log \frac{p(x, y)}{p(x)p(y)}, \tag{5.37}$$

Table 5.3: Average feature saliency on the synthetic dataset with soft feature saliency. The feature saliency is in a decreasing order for cluster 1, and in a increasing order for cluster 2.

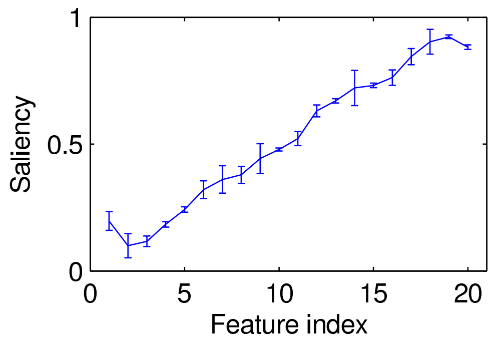
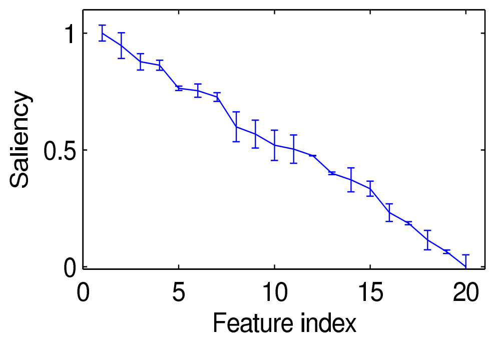
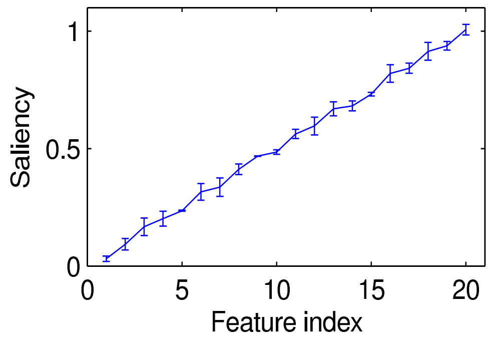
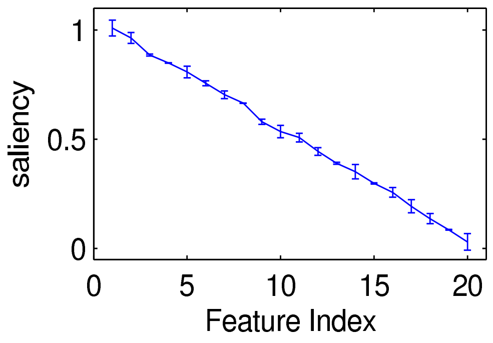
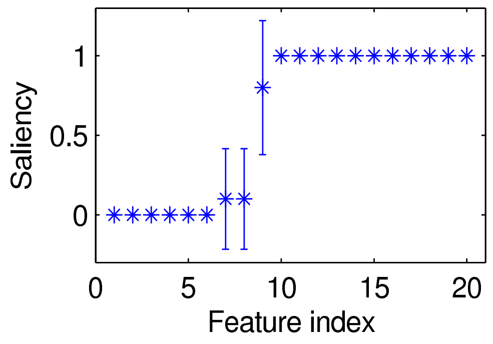
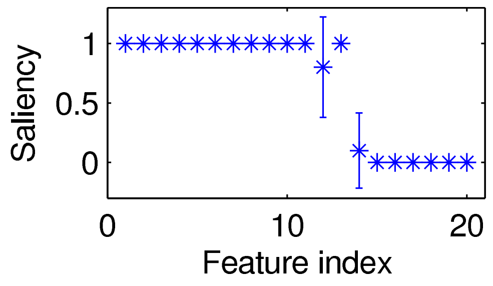
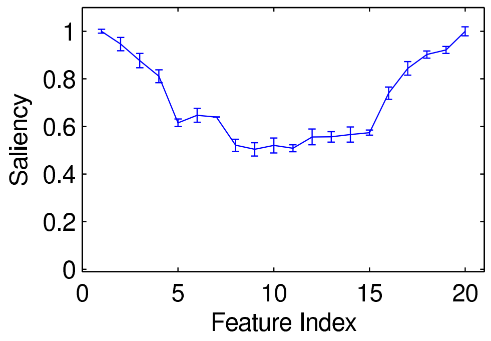
Algo.	Cluster 1	Cluster 2
LFSVB		
COSA		
PMGM		
GFSVB		

Table 5.4: Summary of the UCI datasets, where N is the number of samples, D the number of attributes, and c the number of classes.

data	Description	N	D	c
Heart	Heart Disease of Statlog	270	13	2
Ion	Ionosphere Database	351	34	2
Vehicle	vehicle classification	846	18	4
Wine	wine recognition	178	13	3
WDBC	Diagnostic breast cancer	569	30	2
Yeast	Protein Localization Sites	1484	8	10

where x and y are true labels and cluster index respectively, $p(x, y)$ is the joint probability, and $p(x)$ and $p(y)$ are the marginal probability distribution of $\mathcal{X}$ and $\mathcal{Y}$, respectively. The mutual information measures the dependence between $\mathcal{X}$ and $\mathcal{Y}$. Thus, a higher value of I indicates that the clustering results are closer to the true class group.

Table 5.5 shows the mean and standard deviation of the cluster numbers and mutual information over 10 runs of the four algorithms. Again, cluster numbers for COSA are set manually based on the dendrogram. On the average mutual information, LFSVB outperforms GFSVB on five (out of six) datasets (Heart, Ion, Vehicle, Wine, and Yeast). On WDBC, it is as good as GFSVB. LFSVB also outperforms COSA on five (out of six) datasets (Ion, Vehicle, Wine, WDBC and Yeast). The proposed algorithm outperforms PMGM on two datasets (Heart, Vehicle). On the other datasets, those two algorithms have similar performance.

LFSVB shows that different relevant feature subsets are associated with different clusters, whose sizes are usually smaller than the global relevant feature subset. PMGM also selects a feature subset for each component. The difference between LFSVB and PMGM is that LFSVB evaluates the relevance of a feature with a saliency value in the range of [0, 1] while PMGM uses a binary one. In addition, model detection is fully integrated in LFSVB through variational learning. A separate measure such as BIC is not required.

Table 5.5: Mutual information I and the estimated cluster number $\hat{c}$, represented by mean and standard deviation over 10 different runs, on UCI datasets. For COSA, the number of clusters is determined manually (denoted by *).

Data	Algo	$\hat{c}$(std)	I(std)
Heart	LFSVB	2.8(0.8)	0.15(0.07)
	COSA	2*	**0.21**(0.01)
	GFSVB	3.0(0.7)	0.09(0.06)
	PMGM	3.1(0.6)	0.11 (0.05)
Ion	LFSVB	3.8(1.1)	**0.33**(0.1)
	COSA	4*	0.30(0.01)
	GFSVB	3.4(0.9)	0.21(0.05)
	PMGM	3.3(0.8)	0.31 (0.05)
Vehicle	LFSVB	9.9(1.7)	**0.63**(0.05)
	COSA	9*	0.48(0.01)
	GFSVB	10.5(1.5)	0.58(0.09)
	PMGM	9.5(1.6)	0.60 (0.04)
Wine	LFSVB	3.1(0.3)	**1.44**(0.07)
	COSA	3*	1.26(0.01)
	GFSVB	3.4(0.7)	1.42(0.06)
	PMGM	3.2(0.6)	1.42 (0.07)
WDBC	LFSVB	6.3(0.8)	**0.68**(0.02)
	COSA	10*	0.59(0.01)
	GFSVB	7.6 (0.9)	0.67(0.02)
	PMGM	8.1(0.6)	**0.68** (0.03)
Yeast	LFSVB	11.4(2.1)	**0.40**(0.06)
	COSA	13*	0.15(0.02)
	GFSVB	6.8(0.8)	0.36(0.01)
	PMGM	8.2(1.5)	0.39(0.05)

5.6 Summary

In this chapter, we developed a novel approach of simultaneous localized feature selection and model detection for unsupervised learning. Our approach provides a fully-automated solution to identify useful patterns embedded in feature subspaces by integrating local feature selection, model detection, and clustering into a unified Bayesian framework. We have demonstrated the advantages of our algorithm over global feature selection and subspace clustering methods on both synthetic and real-world datasets.

CHAPTER 6

CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE WORK

6.1 Conclusions

The objective of this dissertation is to identify individual clusters that exist in different feature subspaces. We designed three algorithms to achieve this goal. The contributions of this dissertation are summarized as follows:

- We developed an adjusted and normalized scatter separability measure (ANV) for individual clusters. Based on this measure, A normalized-cross-projection (NCP) method is designed to perform unsupervised localized feature selection. The algorithm identifies the feature subsets for each individual cluster by sequential backward search. Our experimental results on both benchmark datasets and image datasets show the need for feature selection in clustering and the benefits of selecting features locally.

- We formulated the problem of unsupervised localized feature selection as a probability problem based on Gaussian mixture. the feature relevance for an individual cluster is treated as a probability, which is represented by localized feature saliency and estimated through Expectation Maximization (EM) algorithm during clustering process. In addition, the number of clusters is determined by integrating a Minimum Message Length (MML) criterion.

- In the most recent work of this dissertation, we address the problem of simultaneous localized feature selection and model detection for unsupervised learning. We proposed a novel localized Bayesian inference approach of Gaussian mixtures, which computes the local feature saliency, the number of clusters, and other parameters of a mixture model through variational learning.

6.2 Future work

There are many research interests in localized feature selection for data mining. We list some, but not all, among them for future research guidance:

- *Feature dependence in localized feature selection in unsupervised learning.*

  The algorithms in Chapters 3, 4, and 5 assume that features are conditionally independent when cluster labels are known. However, this assumption is not true in general practice. The algorithms need to be designed to cope with this situation.

- *Constraints in localized feature selection for unsupervised learning.*

  Unsupervised feature selection encounters difficulty when clustering is in ill-posed nature. To this end, user defined constrains can improve the clustering accuracy. In many applications, limited supervision is available to resolve this issue. This leads to a research field as known as semi-supervised learning. One could apply similar idea to localized feature selection. Remark that the optimal number of clusters can be inferred by minimizing the constraint violations. A desirable algorithm should also detect the number of clusters automatically.

REFERENCES

[1] M. H. Law, M. A. T. Figueiredo, A. K. Jain, Simultaneous feature selection and clustering using mixture models, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (9) (2004) 1154–1166.

[2] I. T. Jolliffe, Principal Compoent Analysis, 2nd Edition, Springer Series in Statistics, Springer, NY, 2002.

[3] B. Scholkopf, A. Smola, K. R. Muller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Comput 10 (5) (1998) 1299–1319.

[4] H. S. Seung, D. D. Lee., The manifold ways of perception, Science 290 (5500) (2000) 2268–2269.

[5] S. Roweis, L. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (5500) (2000) 2323–2326.

[6] H. Liu, H. Motoda, Instance Selection and Construction for Data Mining, Kluwer Academic Publishers, Boston, 2001.

[7] J. Doak, An evaluation of feature selection methods and their application to computer security, Tech. rep., Univ. of California at Davis, Dept. Computer Science (1992).

[8] P. M. Narendra, K. Fukunaga, A branch and bound algorithm for feature subset selection, IEEE Transactions on Computers 26 (9) (1977) 917–922.

[9] G. Brassard, P. Bratley, Fundamentals of Algorithms, Prentice Hall, New Jersey, 1996.

[10] J. C. W. Debuse, V. J. Rayward-Smith, Feature subset selection within a simulated annealing datamining algorithm, Journal of Intelligent Information Systems 9 (1) (1997) 57–81.

[11] J. Yang, V. Honavar, Feature subset selection using a genetic algorithm, IEEE Intelligent Systems 13 (1998) 44–49.

[12] H. Vafaie, K. D. Jong, Genetic algorithms as a tool for feature selection in machine learning, in: Proceedings of the 4th International Conference on Tools with Artificial Intelligence, Arlington, VA, 1992, pp. 200–204.

[13] C. Constantinopoulos, M. K. Titsias, A. Likas, Bayesian feature and model selection for gaussian mixture models, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (6) (2006) 1013–1018.

[14] M. A. Hall, Correlation based feature selection for discrete and numeric class machine learning, in: Proceedings of the 17th International Conference on Machine Learning, 2000, pp. 395–366.

[15] H. Liu, L. Yu, Toward integrating feature selection algorithms for classification and clustering, IEEE Transactions on Knowledge and Data Engineering 17 (4) (2005) 491–502.

[16] M. Dash, K. Choi, P. Scheuermann, H. Liu, Feature selection for clustering - a filter solution, in: IEEE International Conference on Data Mining, 2002, pp. 115–122.

[17] P. Mitra, C. Murthy, S. Pal, Unsupervised feature selection using feature similarity, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (3) (2002) 301–312.

[18] M. Dong, R. Kothari, Feature subset selection using a new definition of classifiability, Pattern Recognition Letters 23 (2003) 1215–1225.

[19] A. Blum, P. Langley, Selection of relevant features and examples in machine learning, Artificial Intelligence 97 (1-2) (1997) 245–271.

[20] R. Kohavi, G. H. John, Wrappers for feature subset selection, Artificial Intelligence 97 (1997) 273–324.

[21] J. G. Dy, C. E. Brodley, Feature selection for unsupervised learning., Journal of Machine Learning Research 5 (2004) 845–889.

[22] D. Modha, S. Spangler, Feature weighting in k-means clustering, Machine Learning 52 (2003) 217–237.

[23] Y. S. Kim, W. N. Street, F. Menczer, Feature selection in unsupervised learning via evolutionary search, in: Proceedings of the 6th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), 2000, pp. 365–369.

[24] M. Dash, H. Liu, J. Yao, Dimensionality reduction of unsupervised data, in: Proceedings of 9th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'97), 1997, pp. 532–539.

[25] M. Devaney, A. Ram, Efficient feature selection in conceptual clustering, in: Proceedings of the 14th International Conference on Machine Learning, 1997, pp. 92–97.

[26] M. Dash, H. Liu, Feature selection for clustering, in: Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2000, pp. 110–121.

[27] A. Jain, D. Zongker, Feature selection: Evaluation, application, and small sample performance, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (2) (1997) 153–158.

[28] S. Avidan, Joint feature-basis subset selection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'04), Vol. 1, 2004, pp. 283–290.

[29] Y. Wu, A. Zhang, Feature selection for classifying high-dimensional numerical data, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'04), Vol. 2, 2004, pp. 251–258.

[30] L. Yu, H. Liu, Efficient feature selection via analysis of relevance and redundancy, Journal of Machine Learning Research 5 (2004) 1205–1224.

[31] I. S. Dhillon, S. Mallela, D. D. Modha, Information-theoretic co-clustering, in: Proceedings of the 9th ACM International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD'03), 2003, pp. 89–98.

[32] T. George, S. Merugu, A scalable collaborative filtering framework based on co-clustering, in: Proceedings of the 5th IEEE International Conference on Data Mining (ICDM 05), 2005, pp. 625–628.

[33] N. Slonim, N. Tishby, Document clustering using word clusters via the information bottleneck method, in: Proceedings of the 23th ACM International Conference on Information Retrieval (SIGIR'00), 2000, pp. 208–215.

[34] B. Mandhani, S. Joshi, K. Kummamuru, A matrix density based algorithm to hierarchically co-cluster documents and words, in: Proceedings of the 12th International Conference on World Wide Web (WWW'03), 2003, pp. 511–518.

[35] C. H. Oh, K. Honda, H. Ichihashi, Fuzzy clustering for categorical multivariate data, in: Proceedings of the Joint 9th IFSA World Congress and 20th NAFIPS International Conference, 2001, pp. 2154–2159.

[36] K. Kummamuru, A. Dhawale, R. Krishnapuram, Fuzzy co-clustering of documents and keywords, in: Proceedings of the 12th IEEE International Conference on Fuzzy Systems (FUZZ'03), 2003, pp. 772–777.

[37] L. Parsons, E. Haque, H. Liu, Subspace clustering for high dimensional data: a review, ACM Knowledge Discovery and Data Mining Explorations Newsletter 6 (1) (2004) 90–105.

[38] B. King, Step-wise clustering procedures, Journal of American Statistical Association (1967) 86–101.

[39] C. R. Rao, Linear Statistical Inference and Its Applications, John Wiley, 1973.

[40] P. A. Devijver, J. Kittler, Pattern Recognition: A Statistical Approach, Prentice-Hall, New Jersey, 1982.

[41] P. Pudil, J. Novovicova, J. Kittler, Floating search methods in feature selection, Pattern Recognition Letters 15 (1994) 1119–1125.

[42] R. O. Duda, P. E. Hart, D. G. Stork, Pattern Classification, 2nd Edition, John Willey & Sons, Inc, 2000.

[43] K. Fukunaga, Introduction to Statistical Pattern Recognition, Academic Press, Boston, 1990.

[44] C. Wallace, D. Dowe, MML clustering of multi-state, Poisson, von Mises circular and Gaussian distributions, Statistics and Computing 10 (2000) 73–83.

[45] C. Wallace, P. Freeman, Estimation and inference via compact coding, Journal of the Royal Statistical Society, Series B 49 (3) (1987) 241–252.

[46] S. Chang, N. Dasgupta, L. Carin, A bayesian approach to unsupervised feature selection and density estimation using expectation propagation, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05), Vol. 2, 2005, pp. 1043–1050.

[47] T. Hofmann, Probabilistic latent semantic indexing, in: Proceedings of the 22nd International Conference on Research and Development in Information Retrieval (SIGIR'99), 1999, pp. 50–57.

[48] T. Hofmann, J. Puzicha, Statistical models for co-occurrence and histogram data, in: Proceedings of International Conference on Pattern Recognition, 1998, pp. 192–194.

[49] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (8) (2000) 888–905.

[50] F. R. K. Chung, Spectral Graph Theory, American Mathematical Society, 1997.

[51] G. H. Golub, G. F. Van-Loan, Matrix Computations, John Hopkins Press, 1989.

[52] I. S. Dhillon, Co-clustering documents and words using bipartite spectral graph partitioning, in: Proceedings of the 7th International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD'01), 2001, pp. 269–274.

[53] H. Zha, X. He, C. Ding, M. Gu, H. Simon, Bipartite graph partitioning and data clustering, in: Proceedings of the Conference on Information and Knowledge Management (ACM CIKM'01), 2001, pp. 25–32.

[54] C. H. Q. Ding, Document retrieval and clustering: from principal component analysis to self-aggregation networks, in: Proceedings of the 9th International Workshop on Artificial Intelligence and Statistics, 2003.

[55] J. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities, in: Proceedings of the National Academy of Sciences of the United States of America, Vol. 79, 1982, pp. 2554–2558.

[56] R. Agrawal, J. Gehrke, D. Gunopulos, P. Raghavan, Automatic subspace clustering of high dimensional data for data mining applications, in: Proceedings of the ACM International Conference on Management of Data (ACM-SIGMOD), Seattle, Washington, 1998, pp. 94–105.

[57] C. H. Cheng, A. W.-C. Fu, Y. Zhang, Entropy-based subspace clustering for mining numerical data, in: Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD'99), ACM Press, 1999, pp. 84–93.

[58] S. Goil, H. Nagesh, A. Choudhary, MAFIA: Efficient and scalable subspace clustering for very large data sets, Tech. rep., Department of Electrical and Computer Engineering, Northwestern University, 2145 Sheridan Rd, Evanston, IL 60208 (1999).

[59] C. H. Chang, D. S. Jin, A new cell-based clustering method for large, high-dimensional data in data mining, in: Proceedings of the 2002 ACM Symposium on Applied Computing, 2002, pp. 503–507.

[60] B. Liu, Y. Xia, P. S. Yu, Clustering through decision tree construction, in: Proceedings of the 9th International Conference on Information and Knowledge Management (ACM SIGIKM'00), ACM Press, 2000, pp. 20–29.

[61] C. M. Procopiuc, M. Jones, P. K. Agarwal, T. M. Murali, A Monte Carlo algorithm for fast projective clustering, in: Proceedings of the International Conference on Management of Data (ACM SIGMOD), ACM Press, 2002, pp. 418–427.

[62] C. Baumgartner, C. Plant, K. Kailing, H.-P. Kriegel, P. Kröger, Subspace selection for clustering high-dimensional data, in: Proceedings of the 4th IEEE International Conference on Data Ming (ICDM'04), 2004, pp. 11–18.

[63] C. C. Aggarwal, C. M. Procopiuc, J. L. Wolf, P. S. Yu, J. S. Park, Fast algorithms for projected clustering, in: Proceedings of the International Conference on Management of Data (ACM-SIGMOD'99), 1999, pp. 61–72.

[64] J. H. Friedman, J. J. Meulman, Clustering objects on subsets of attributes, Journal of the Royal Statistical Society: Series B 66 (4) (2004) 815–849.

[65] Y. Li, M. Dong, J. Hua, Localized feature selection for clustering, Pattern Recognition Letters 29 (2008) 10–18.

[66] R. Caruana, D. Freitag, Greedy attribute selection, in: Proceedings of the International Conference on Machine Learning, 1994, pp. 28–36.

[67] H. Motoda, H. Liu, Data reduction: feature selection, Oxford University Press, Inc., 2002, pp. 208–213.

[68] J. Yang, V. G. Honavar, Feature subset selection using a genetic algorithm, IEEE Intelligent Systems 13 (2) (1998) 44–49.

[69] Q. Ke, T. Kanade, Robust subspace clustering by combined use of kNND metric and SVD algorithm, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'04), IEEE, 2004, pp. 592–599.

[70] A. Asuncion, D. Newman, UCI machine learning repository (2007).
URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`

[71] M. Figueiredo, A. K. Jain., Unsupervised learning of finite mixture models, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (2002) 381–396.

[72] Y. Li, M. Dong, J. Hua, A gaussian mixture model to detect clusters embedded in feature subspace, Journal of Communications in Information and Systems 7 (4) (2007) 337–352.

[73] Y. Li, M. Dong, J. Hua, Simultaneous localized feature selection and model detection for gaussian mixtures, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (5) (2009) 953–960.

[74] C. M. Bishop, Pattern Recognition and Machine Learning, 1st Edition, Springer, 2006, Ch. 10.

[75] C. K. Reddy, H. Dong Chiang, B. Rajaratnam, TRUST-TECH-based expectation maximization for learning finite mixture models, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (7) (2008) 1–12.

[76] M. W. Graham, D. J. Miller, Unsupervised learning of parsimonious mixtures on large spaces with integrated feature and component selection, IEEE Transactions on Signal Processing 54 (4) (2006) 1289–1303.

ABSTRACT

LOCALIZED FEATURE SELECTION FOR UNSUPERVISED LEARNING

by

YUANHONG LI

August 2010

Advisor: Dr. Ming Dong

Major: Computer Science

Degree: Doctor of Philosophy

Clustering is the unsupervised classification of data objects into different groups (clusters) such that objects in one group are similar together and dissimilar from another group. *Feature selection* for unsupervised learning is a technique that chooses the *best* feature subset for clustering. In general, unsupervised feature selection algorithms conduct feature selection in a *global* sense by producing a common feature subset for all the clusters. This, however, can be invalid in clustering practice, where the local intrinsic property of data matters more, which implies that *localized feature selection* is more desirable.

In this dissertation, we focus on cluster-wise feature selection for unsupervised learning. We first propose a Cross-Projection method to achieve localized feature selection. The proposed algorithm computes adjusted and normalized scatter separability for individual clusters. A sequential backward search is then applied to find the optimal (perhaps local) feature subsets for each cluster. Our experimental results show the need for feature selection in clustering and the benefits of selecting features locally.

We also present another approach based on Maximal Likelihood with Gaussian mixture. We introduce a probabilistic model based on Gaussian mixture. The feature relevance for an individual cluster is treated as a probability, which is represented by localized feature saliency and estimated through Expectation Maximization (EM) algorithm during the clustering process. In addition, the number of clusters is determined by integrating a Minimum Message

Length (MML) criterion. Experiments carried out on both synthetic and real-world datasets illustrate the performance of the approach in finding embedded clusters.

Another novel approach based on Bayesian framework is successfully implemented. We place prior distributions over the parameters of the Gaussian mixture model, and maximize the marginal log-likelihood given mixing co-efficient and feature saliency. The parameters are estimated by Bayesian Variational Learning. This approach computes the feature saliency for each cluster, and detects the number of clusters simultaneously.

AUTOBIOGRAPHICAL STATEMENT

YUANHONG LI

Yuanhong Li received a B.S. degree and a Ph.D. degree in Materials Science from the University of Science and Technology of China in 1994 and 1999, respectively. He joined the Wayne State University in 2001, in pursuit of a master degree in computer science. In December 2003, he received his M.S. degree in Computer Science. Currently, he is a Ph.D. candidate in the Machine Vision and Pattern Recognition Laboratory in the Department of Computer Science, Wayne State University, Detroit, Michigan. He also worked as a contract scientist at General Motors Corp., Detroit, since 2008. His research interests are in the areas of Pattern Recognition, Machine Learning, Data Mining, Signal Processing, Sensor Fusion, and Information Retrieval.