Clustering constrained by dependencies

Satish Tadepalli[†], Naren Ramakrishnan[†], and Layne T. Watson[†*]
Departments of Computer Science[†] and Mathematics[*]
Virginia Polytechnic Institute and State University, VA 24061

Abstract

Clustering is the unsupervised method of grouping data samples to form a partition of a given dataset. Such grouping is typically done based on homogeneity assumptions of clusters over an attribute space and hence the precise definition of the similarity metric affects the clusters inferred. In recent years, new formulations of clustering have emerged that posit indirect constraints on clustering, typically in terms of preserving dependencies between data samples and auxiliary variables. These formulations find applications in bioinformatics, web mining, social network analysis, and many other domains. The purpose of this survey is to provide a gentle introduction to these formulations, their mathematical assumptions, and the contexts under which they are applicable.

1 Introduction

Clustering algorithms [7, 13, 16, 25, 30] are unsupervised methods to group data samples such that the samples assigned to a group (cluster) are highly similar with each other but are dissimilar from those assigned to other clusters. Although the classic survey by Jain, Murty, and Flynn [17] was published nearly a decade back, and retrospective views of classical clustering algorithms have emerged [15], interest in clustering research remains strong – primarily due to the advent of high-throughput data acquisition technologies [1, 8, 14] in practically every field of computational science.

Clustering algorithms grapple with two fundamental issues: (i) what constitutes a cluster? and (ii) how to efficiently infer clusters from data? The former is a matter of problem definition and classical clustering research has addressed it by defining clusters as distributions around cluster prototypes and using distance or similarity measures [9] to determine cluster boundaries. The precise definition of these measures influences the shapes and sizes of clusters found. The efficient inference of clusters is then posed as an optimization over the space of cluster prototypes. Other constraints such as balanced clusters [3] can also be imposed in the optimization.

Recently, new clustering algorithms have been developed that, in addition to/in lieu of a distance measure, also utilize some prior *auxiliary information* about the data to drive the clustering. The intent is that the clustering, when viewed as a compression of the data, must preserve the auxiliary information as much as possible. Such auxiliary information can come in various guises. For instance, it could be some prior class (supervised) information that we seek to mimic in our clustering. It could be relationships between data samples [19] if we are simultaneously clustering two or more datasets. In either case, the objective function is suitably modified to capture the notion of preservation of relevant information.

Our survey here is scoped by clustering algorithms that exhibit one or both of the following traits. First, the given dataset has both attribute-value and relational components and it is desired to use one to influence the other. Second, the objectives of clustering are specified in terms of dependencies or constraints [24, 32]. These constraints can either be direct such as that two given data points should (or should not) belong to a cluster [5, 31], or they could be indirectly specified in terms of properties of induced clusters. Even with these restrictions, the amount of related work is broad and we identify key representative methods to illustrate the diversity of related research. We first begin by reviewing pertinent background in information theory.

2 Review of information theory

We denote discrete random variables by capital letters (X, Y, ...) and their realizations (specific values taken by the random variables) by lowercase letters (x, y, ...). The set of all possible values of a discrete random variable is denoted by calligraphic letters, i.e., $x \in \mathcal{X}$ denotes the possible values of X. Vectors of random variables are denoted by uppercase bold letters, $\mathbf{X}$, $\mathbf{Y}$, ..., and their realizations are denoted by lowercase bold letters, $\mathbf{x}$, $\mathbf{y}$, ... The probability distribution (density function) of X is denoted by $p(x) = P(X = x)$, and the conditional probability distribution is denoted by $p(x|y) = P(X = x|Y = y)$.

Entropy
Entropy is a measure of the uncertainty of a random variable. The entropy of a discrete random variable X with distribution $p(x)$ is defined as

$$H(X) = -\sum_{x\in\mathcal{X}} p(x)\log p(x). \tag{1}$$

For discrete random variables X, Y with joint distribution $f(x, y)$ and marginal distributions $g(x)$, $h(y)$, the conditional entropy is defined as

$$H(X|Y = y) = -\sum_{x\in\mathcal{X}} \frac{f(x, y)}{h(y)} \log \frac{f(x, y)}{h(y)}$$

and

$$H(X|Y) = \sum_{y\in\mathcal{Y}} h(y)H(X|Y = y). \tag{2}$$

KL-divergence
The Kullback Leibler divergence between two probability distributions $p(x)$ and $q(x)$ is defined as

$$D_{KL}[p\|q] = \sum_x p(x)\log\frac{p(x)}{q(x)}, \tag{3}$$

with the limits $\lim_{p(x)\to 0} p(x)\log\frac{p(x)}{q(x)} = 0$, $\lim_{q(x)\to 0} p(x)\log\frac{p(x)}{q(x)} = \infty$ implied. It measures the inefficiency of assuming that the distribution is q when the true distribution is p. KL-divergence is colloquially referred to as a "distance" between two distributions p and q although it doesn't satisfy the requirements of a distance measure. It is nonnegative and is zero if and only if $p = q$. However, it is not symmetric and does not satisfy the triangle inequality.

JS divergence
The Jensen-Shannon divergence between two probability distributions $p(x)$ and $q(x)$ is defined as

$$JS_\Pi(p, q) = \pi_1 D_{KL}[p\|m] + \pi_2 D_{KL}[q\|m], \tag{4}$$

where $\Pi = (\pi_1, \pi_2)$, $0 < \pi_1, \pi_2 < 1$, $\pi_1 + \pi_2 = 1$ and $m = \pi_1 p + \pi_2 q$. It is easy to verify that the JS divergence is a true distance metric.

Mutual Information
The mutual information between two random variables X and Y with a joint distribution $f(x, y)$ and marginal distributions $g(x)$ and $h(y)$ is defined as

$$I(X; Y) = \sum_x \sum_y f(x, y)\log\frac{f(x, y)}{g(x)h(y)}. \tag{5}$$

Mutual information is a measure of dependence between the two random variables. It can be expressed as the KL divergence between $f(x, y)$ and $g(x)h(y)$. In terms of entropy,

$$I(X; Y) = H(X) - H(X|Y) = H(Y) - H(Y|X). \tag{6}$$

Mutual information is the reduction in uncertainty of one random variable due to the knowledge of the other.

3 The single sided information bottleneck

The information bottleneck (IB) method introduced by Tishby et al. [29] is an information theoretic approach to clustering. In the information bottleneck method, each primary data sample x, which is an instance of the random variable X, is associated with some auxiliary data sample y, which is an instance of another discrete random variable Y. A joint probability distribution $p(x, y)$ of this association is assumed to be available. Typically X represents the features of the objects to be clustered and Y represents some relevant information about these objects. For instance, X could denote features of documents and Y could denote some *a priori* classification of documents, such as whether they are news articles or research papers. The goal of IB is to cluster X in such a way that the clustering preserves the distinctions made by Y. Of course, a trivial answer is to just group documents as suggested by Y but such a clustering will not yield a compression of the given dataset. IB aims to find clusters such that the clusters afford compression of X *and* preserve information about Y.

The *IB variational principle* captures the tradeoff between compression (increasing compression reduces I(C;X)) and preservation of auxiliary information (increasing preservation increases I(C;Y)) via

$$\mathcal{L} = I(C; X) - \beta I(C; Y), \tag{7}$$

where C is the desired clustering, $I(C; X)$ is the compression information, and $I(C; Y)$ is the auxiliary information. Lower values of $\beta > 0$ give more importance to compression while higher values give more importance to preserving relevant information.

We seek clusters such that the cluster membership probabilities of the data samples, i.e., $p(c|x)$, minimize $\mathcal{L}$. In order to express $\mathcal{L}$ in terms of $p(c|x)$, we substitute the expressions for mutual information

$$\begin{aligned}\mathcal{L} &= \sum_{x,c} p(c, x) \log \frac{p(c, x)}{p(c)p(x)} - \beta \sum_{c,y} p(c, y) \log \frac{p(c, y)}{p(c)p(y)} \\ &= \sum_{x,c} p(x)p(c|x) \log \frac{p(c|x)}{p(c)} - \beta \sum_{c,y} p(c)p(y|c) \log \frac{p(y|c)}{p(y)}.\end{aligned} \tag{8}$$

Observe that since samples x and y are associated, knowing x and y provides no more information than knowing just x, i.e., $p(c|x, y) = p(c|x)$. Using this independence relation,

$$p(x, y, c) = p(c|x)p(x, y), \tag{9}$$

$$p(c) = \sum_{x,y} p(x, y, c) = \sum_{x} p(x)p(c|x), \tag{10}$$

$$p(y|c) = \frac{1}{p(c)} \sum_{x} p(x, y, c) = \frac{1}{p(c)} \sum_{x} p(x, y)p(c|x). \tag{11}$$

Using the above equations it can be shown that $p(c|x)$ is a stationary point of $\mathcal{L}$ if and only if

$$p(c|x) = \frac{p(c)}{Z(x, \beta)} e^{-\beta D_{KL}[p(y|x) \| p(y|c)]}, \forall c \in \mathcal{C}, \forall x \in \mathcal{X}, \tag{12}$$

where $Z(x, \beta)$ is a normalization function so that $\sum_c p(c|x) = 1$. It is evident that the KL divergence, $D_{KL}[p(y|x) \| p(y|c)]$, is the effective distortion measure for the IB functional. This is a key contribution of the IB framework — without making any assumptions on the distortion measure, IB shows that the KL-divergence between the distributions of auxiliary data as represented by the primary data and the clusters is the exact distortion measure to be minimized, and derives $p(c|x)$ in terms of this measure.

In any clustering algorithm, similar data points are grouped together and cluster prototypes are chosen such that the distance between the data points in a cluster and the prototype of the cluster is minimized. In the IB framework, the distances between the data points are measured in terms of their associations with the auxiliary data, $p(y|x)$. The auxiliary data distribution represented by a cluster $p(y|c)$ is interpreted as the cluster prototype. A data point x is assigned to a cluster c if $p(y|c)$ is similar to $p(y|x)$, and this is captured by the KL-divergence $D_{KL}[p(y|x) \| p(y|c)]$.

In practice IB algorithms start with a random assignment of the data points to clusters and iteratively improve this solution such that at each step $\mathcal{L}$ decreases and the iterates eventually converge to a local optimal solution. Some of these algorithms are described below.

Iterative Information Bottleneck

This algorithm is a straightforward method that uses the Eqs. (10), (11), and (12). The update steps for the $(n+1)$-st iteration are given by

1. $p^{(n+1)}(c|x) = \frac{p^{(n)}(c)}{Z(x,\beta)} e^{-\beta D_{KL}[p(y|x)\|p^{(n)}(y|c)]}, \forall c \in \mathcal{C}, \forall x \in \mathcal{X}$.
2. $p^{(n+1)}(c) = \sum_x p(x) p^{n+1}(c|x), \forall x \in \mathcal{X}$.
3. $p^{(n+1)}(y|c) = \frac{1}{p^{n+1}(c)} \sum_x p(x,y) p^{n+1}(c|x)$.

These steps are repeated till $p(c|x)$ converges.

Agglomerative Information Bottleneck

The agglomerative IB algorithm [27] is analogous to hierarchical clustering algorithms. A hierarchical clustering algorithm starts with a proximity matrix containing the distance between data points and each data point is treated as a singleton cluster. The most similar pair of clusters is merged in each iteration and the proximity matrix is updated to reflect this merger. The process is repeated until all data points are in the same cluster.

Recall that the IB framework implicitly specifies the distance measure. In order to calculate the cost of merging two clusters, consider the problem of maximizing the IB functional

$$\mathcal{L}_{max} = I(C;Y) - \beta^{-1} I(C;X). \tag{13}$$

Observe that this functional is simply (7) multiplied by $-\beta^{-1}$, and hence maximizing $\mathcal{L}_{max}$ is equivalent to minimizing $\mathcal{L}$. Suppose two clusters c_i and c_j are merged to form a single cluster $\bar{c}$. The membership probabilities with respect to this new cluster $\bar{c}$ are given by

$$p(\bar{c}|x) = p(c_i|x) + p(c_j|x), \forall x \in \mathcal{X}. \tag{14}$$

Observe that such an addition will preserve the requirements of a probability measure. Using this equation we can define

$$p(\bar{c}) = p(c_i) + p(c_j), \tag{15}$$

$$p(y|\bar{c}) = \frac{p(c_i)}{p(\bar{c})} p(y|c_i) + \frac{p(c_j)}{p(\bar{c})} p(y|c_j). \tag{16}$$

$\mathcal{L}_{max}$ in equation 13 has a maximum value when each data point is in its own singleton cluster. Merging any two clusters results in a decrease of $\mathcal{L}_{max}$. This decrease in the value of $\mathcal{L}_{max}$ is

$$\Delta\mathcal{L}_{max}(c_i, c_j) = p(\bar{c}) \cdot d(c_i, c_j), \tag{17}$$

where $\Pi = \big(p(c_i), p(c_j)\big)/p(\bar{c})$ and

$$d(c_i, c_j) = JS_{\Pi}[p(y|c_i), p(y|c_j)] - \beta^{-1} JS_{\Pi}[p(x|c_i), p(x|c_j)]. \tag{18}$$

The agglomerative IB algorithm constructs a hierarchical clustering by merging the pair of clusters that cause minimum information bottleneck loss as given by $\Delta\mathcal{L}_{max}$ at each step.

Sequential Information Bottleneck

The agglomerative IB algorithm constructs a hierarchical clustering representation of the data set. However, for large data sets, this procedure is impractical. The sequential IB algorithm uses the cost function $\Delta\mathcal{L}_{max}$ described above more efficiently.

The algorithm begins with a fixed partition of N clusters. Then in each iteration, the following steps are performed:

1. Choose some data point x_i at random and remove it from its cluster c_k.
2. Assume the data point x_i is in a singleton cluster $\{x_i\}$ and calculate the merger cost $\Delta\mathcal{L}_{\max}(\{x_i\}, c_j)$, $j = 1$, $\ldots$, N and assign x_i to the cluster c_j with minimum cost.
3. Update $p(y|c_j)$ and $p(y|c_k)$.

These steps are repeated till the data points do not change their cluster memberships.

4 Multivariate Information Bottleneck

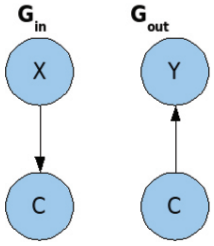


Figure 1: Input and Output networks for single sided IB

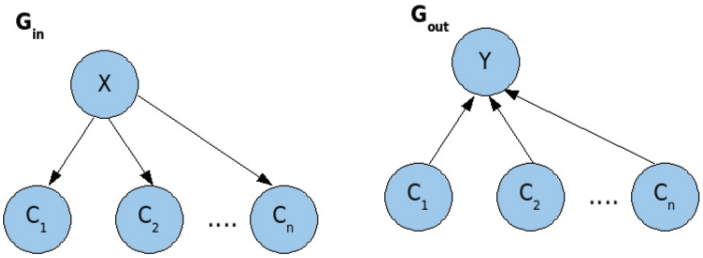


Figure 2: Parallel IB

Multivariate IB [10] generalizes the single sided IB framework to clustering several datasets simultaneously by preserving auxiliary information about several other datasets. The dependencies between the datasets are captured by two Bayesian networks G_{in} and G_{out}. G_{in} represents the "compression relationships"—if C is a clustered representation of X, then a directed edge from X to C exists in G_{in}. The Bayesian network G_{out} captures "information preservation relationships"–if a clustering C preserves information about Y, then a directed edge exists from C to Y in G_{out}. The networks G_{in} and G_{out} for the single sided IB are shown in Figure 1. Parallel IB and symmetric IB shown in Figures 2 and 3 are two multivariate formulations that have interesting applications and interpretations. In parallel IB, we seek several systems of clusters of the same dataset X that preserve information about the same Y independently. In effect, these clustering schemes capture independent aspects of the information X contains about Y. In symmetric IB, the clustering C_X compresses X preserving information about Y and at the same time C_Y compresses Y preserving information about X. In the G_{out} corresponding to symmetric IB, this is captured through a

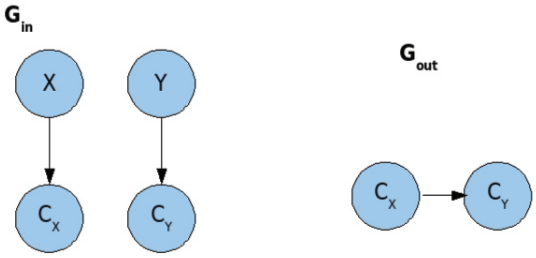


Figure 3: Symmetric IB

single link from C_X to C_Y. C_X preserves information about Y through C_Y. Hence no direct link between C_X and Y is required.

The information represented by a Bayesian network G is defined as

$$I^G = \sum_{i \in S} I(X_i; \mathbf{Pa}_{X_i}^G), \tag{19}$$

where $\mathbf{Pa}_{X_i}^G$ is a vector random variable formed from the parents of X_i in the network G and the sum is over all nodes S of G with indegree ≥ 1. The multivariate IB functional captures the tradeoff between the informations represented by the "compression" network, $I^{G_{in}}$, and the "information preservation" network, $I^{G_{out}}$

$$\mathcal{L}^{MIB} = I^{G_{in}} - \beta I^{G_{out}}. \tag{20}$$

In the following discussion, each X_i represents the random variable corresponding to dataset i and C_i represents its corresponding cluster random variable. Lower case letters correspond to specific values of these random variables. The cluster k in dataset i is represented by c_i^k. For convenience of notation we let $\mathbf{U}_j = \mathbf{Pa}_{C_j}^{G_{in}}$ (the compression variables C_j are always children in G_{in}), $\mathbf{V}_W = \mathbf{Pa}_W^{G_{out}}$ where W can be a X_i or C_l, $\mathbf{V}_{C_l}^{-j} = \mathbf{V}_{C_l} \backslash C_j$, and $\mathbf{V}_{X_i}^{-j} = \mathbf{V}_{X_i} \backslash C_j$ where the notation $\mathbf{V} \backslash C$ means the vector $\mathbf{V}$ with its component C deleted. $E_{(p())}$ denotes the expectation with respect to distribution $p()$. $C_j \vdash \mathbf{V}_W$ indicates that C_j is a component of the vector $\mathbf{V}_W$.

The conditional distributions $\{p(c_j|\mathbf{u}_j)\}$ are a stationary point of the multivariate IB functional $\mathcal{L}^{MIB}$ if and only if

$$p(c_j|\mathbf{u}_j) = \frac{p(c_j)}{Z_{C_j}(\mathbf{u}_j, \beta)} e^{-\beta d(c_j, \mathbf{u}_j)}, \forall c_j \in \mathcal{C}_j, \forall \mathbf{u}_j \in \mathcal{U}_j, \tag{21}$$

where $Z_{C_j}(\mathbf{u}_j, \beta)$ is a normalization function, and

$$\begin{aligned} d(c_j, \mathbf{u}_j) = & \sum_{\{i: C_j \vdash \mathbf{V}_{X_i}\}} E_{p(\mathbf{v}_{X_i}^{-j}|\mathbf{u}_j)}[D_{KL}[p(x_i|\mathbf{v}_{X_i}^{-j}, \mathbf{u}_j) \| p(x_i|\mathbf{v}_{X_i}^{-j}, c_j)]] \\ & + \sum_{\{l: C_j \vdash \mathbf{V}_{C_l}\}} E_{p(\mathbf{v}_{C_l}^{-j}|\mathbf{u}_j)}[D_{KL}[p(c_l|\mathbf{v}_{C_l}^{-j}, \mathbf{u}_j) \| p(c_l|\mathbf{v}_{C_l}^{-j}, c_j)]] \\ & + D_{KL}[p(\mathbf{v}_{C_j}|\mathbf{u}_j) \| p(\mathbf{v}_{C_j}|c_j)]. \end{aligned} \tag{22}$$

The first summation in the distortion measure above is over all the data variables X_i, and the second summation is over all the cluster variables C_l whose information is preserved by clusters in C_j. The last term is used if information about C_j is preserved by $\mathbf{V}_{C_j}$.

Multivariate IB clustering algorithms are similar to the single sided IB algorithms. The Iterative MIB algorithm uses updates for distributions based on the stationary equations (21). The agglomerative and sequential MIB algorithms use a cluster merging criterion similar to that in single sided IB. To derive this criterion consider maximizing

$$\mathcal{L}_{max}^{MIB} = I^{G_{out}} - \beta^{-1} I^{G_{in}}. \tag{23}$$

This equation is obtained by multiplying the MIB functional $\mathcal{L}^{MIB}$ in (20) by $-\beta^{-1}$. Suppose clusters c_j^r and c_j^s in dataset j are merged to form $\bar{c}_j$, the cluster probabilities with respect to this new cluster are defined as

$$p(\bar{c}_j|\mathbf{u}_j) = p(c_j^r|\mathbf{u}_j) + p(c_j^s|\mathbf{u}_j). \tag{24}$$

The reduction in the value of $\mathcal{L}_{max}$, which is the cost function for merging c_j^r and c_j^s, is

$$\Delta\mathcal{L}_{max}(c_j^r, c_j^s) = p(\bar{c}_j)\bar{d}(c_j^r, c_j^s), \tag{25}$$

where $\Pi = \big(p(c_j^r|\mathbf{u}_j), p(c_j^s|\mathbf{u}_j)\big)/p(\bar{c}_j|\mathbf{u}_j)$ and

$$\begin{aligned}\bar{d}(c_j^r, c_j^s) = &\sum_{\{i:C_j\vdash\mathbf{V}_{X_i}\}} E_{p(\mathbf{v}_{X_i}^{-j}|\bar{c}_j)}\Big[JS_\Pi\big[p(x_i|\mathbf{v}_{X_i}^{-j}, c_j^r), p(x_i|\mathbf{v}_{X_i}^{-j}, c_j^s)\big]\Big] \\ &+ \sum_{\{l:C_j\vdash\mathbf{V}_{C_l}\}} E_{p(\mathbf{v}_{C_l}^{-j}|\bar{c}_j)}\Big[JS_\Pi\big[p(c_l|\mathbf{v}_{C_l}^{-j}, c_j^r), p(c_l|\mathbf{v}_{C_l}^{-j}, c_j^s)\big]\Big] \\ &+ JS_\Pi\big[p(\mathbf{v}_{C_j}|c_j^r), p(\mathbf{v}_{C_j}|c_j^s)\big] - \beta^{-1}JS_\Pi\big[p(\mathbf{u}_j|c_j^r), p(\mathbf{u}_j|c_j^s)\big].\end{aligned} \tag{26}$$

The agglomerative and sequential multivariate IB algorithms use the cost function in (25) in a similar fashion to their single sided IB counterparts.

5 K-means as IB

The popular k-means clustering algorithm has been shown to be a special case of IB [28]. The regular k-means algorithm can be initialized by randomly chosen cluster prototypes. In each iteration, the distance between the data points and prototypes is measured, and the data points are assigned to the nearest cluster. For each cluster, the mean of the data points in the cluster is chosen as the new prototype. To formulate k-means as IB, given a dataset $\mathcal{Y} = \{\mathbf{y}_i : i \in \mathcal{X}\}$, $\mathcal{X} = \{1, \ldots, N\}$, we interpret the clustering algorithm as compressing the indices x of the data vectors while preserving the information about their location $\mathbf{y}$. The IB functional for k-means is

$$\mathcal{L}_{max}(p(c|i)) = I(C;\mathbf{Y}) + \lambda I(C;X). \tag{27}$$

Note that $\lambda = -\beta^{-1}$ is the penalty parameter. Recall that the optimal solution according to IB is

$$p(c|x) = \frac{p(c)}{Z(x,\lambda)} \exp\left[\frac{1}{\lambda}\sum_{\mathbf{y}} p(\mathbf{y}|x)\log\frac{p(\mathbf{y}|x)}{p(\mathbf{y}|c)}\right].$$

Since the index x is associated with a particular data vector $\mathbf{y}$, $p(\mathbf{y}|x) = 0$ or 1, so the sum inside exp reduces to

$$-\frac{1}{\lambda}\sum_{\mathbf{y}} p(\mathbf{y}|x)\log p(\mathbf{y}|c).$$

Given the index x of a data vector $\mathbf{y}$, we know its cluster membership c, hence, $p(\mathbf{y}|x,c) = p(\mathbf{y}|x)$. This implies

$$p(\mathbf{y}|c) = \frac{1}{p(c)}\sum_x p(\mathbf{y}|x)p(c|x)p(x). \tag{28}$$

Let N_c be the number of clusters. We assume that all N data vectors are equally likely, $p(x) = \frac{1}{N}$. Using the iterative IB algorithm, the n-th iteration is

$$p_n(c|x) = \frac{p_{n-1}(c)}{Z(x,\lambda)} \exp\left[-\frac{1}{\lambda}\sum_{\mathbf{y}} p(\mathbf{y}|x)\log[p_{n-1}(\mathbf{y}|c)]\right], \tag{29}$$

$$p_n(\mathbf{y}|c) = \frac{1}{Np_{n-1}(c)}\sum_x p(\mathbf{y}|x)p_n(c|x), \tag{30}$$

$$p_n(c) = \frac{1}{N}\sum_x p_n(c|x). \tag{31}$$

However, this iteration does not involve cluster prototypes and its limit depends on the initial values for $p(\mathbf{y}|c)$. To make the connection with k-means, we impose a distance measure $d(\mathbf{y}, \mathbf{y}')$ on the data space and initialize $p(\mathbf{y}|c)$ as

$$p_0(\mathbf{y}|c) = \frac{1}{Z_0(c,s)} \exp\left[-\frac{1}{s} d\big(\mathbf{y}, \mathbf{y}_c^{(0)}\big)\right], \tag{32}$$

where $\mathbf{y}_c^{(0)}$ are the initial cluster prototypes chosen at random, $Z_0(c,s)$ is a normalizing factor, and $s > 0$ is a scale factor. After each iteration, the cluster prototypes are chosen to minimize

$$\sum_{\mathbf{y}} p_n(\mathbf{y}|c) d\big(\mathbf{y}, \mathbf{y}_c^{(n)}\big)$$

by solving

$$\sum_{\mathbf{y}} p_n(\mathbf{y}|c) \frac{\partial d\big(\mathbf{y}, \mathbf{y}_c^{(n)}\big)}{\partial \mathbf{y}_c^{(n)}} = 0.$$

For $d(\mathbf{y}, \mathbf{y}')$ being Euclidean distance squared, this becomes

$$\mathbf{y}_c^{(n)} = \sum_{\mathbf{y}} p_n(\mathbf{y}|c)\, \mathbf{y} \Big/ \sum_{\mathbf{y}} p_n(\mathbf{y}|c) = \sum_{\mathbf{y}} p_n(\mathbf{y}|c)\, \mathbf{y}. \tag{33}$$

Now using these cluster prototypes $\mathbf{y}_c^{(n)}$, redefine the conditional probabilities $p_n(\mathbf{y}|c)$ in (30)) by

$$p_n(\mathbf{y}|c) = \frac{1}{Z_n(c,s)} \exp\left[-\frac{1}{s} d\big(\mathbf{y}, \mathbf{y}_c^{(n)}\big)\right]. \tag{34}$$

For $\mathbf{y} = \mathbf{y}_x$, the IB iteration (29) satisfies

$$p_{n+1}(c|x) \propto p_n(\mathbf{y}|c)^{-\frac{1}{\lambda}}.$$

For fixed $\mathbf{y}$, let $d\big(\mathbf{y}, \mathbf{y}_\alpha^{(n)}\big) = \min_c d\big(\mathbf{y}, \mathbf{y}_c^{(n)}\big)$, and observe that $p_n(\mathbf{y}|\alpha) > p_n(\mathbf{y}|c)\ \forall c \neq \alpha$ and $p_n(\mathbf{y}|\alpha)^{-1/\lambda} \gg p_n(\mathbf{y}|c)^{-1/\lambda}$ for $-1 \ll \lambda < 0$ and $\forall c \neq \alpha$. Thus

$$p_n(\mathbf{y}_x|\alpha)^{-1/\lambda} \propto p_{n+1}(\alpha|x) \to 1 \text{ and } p_n(\mathbf{y}_x|c)^{-1/\lambda} \propto p_{n+1}(c|x) \to 0 \tag{35}$$

$\forall c \neq \alpha$ as $\lambda \to 0$. Thus as $\lambda \to 0$, the IB iteration, with the cluster prototype $\mathbf{y}_c^{(n)}$ and distance $d\big(\mathbf{y}, \mathbf{y}_c^{(n)}\big)$ calculations, reduces to k-means, where all the probabilities $p(c|x)$ are 0 or 1, and the $\mathbf{y}_c^{(n)}$ become centroids, since $1/p(\mathbf{y}|c) =$ the number of $\mathbf{y}_i$ assigned to cluster c.

6 Information theoretic co-clustering

Information theoretic co-clustering [6] gives an efficient algorithm for clustering two discrete datasets simultaneously when a joint distribution of the datasets is available. The rows and columns of this joint distribution matrix are grouped simultaneously resulting in homogeneous two-dimensional blocks, each of which is called a co-cluster. The resultant clusters in the datasets are mutually informative of each other. This problem can be formulated as symmetric IB in the multivariate IB framework. However, the co-clustering algorithm gives a more efficient formulation.

The data samples across the rows of the joint distribution table $p(x, y)$ are represented by X and those down the columns are represented by Y. The task is to cluster X into C_X and Y into C_Y so that C_X preserves maximum information about Y and C_Y preserves maximum information about X. The loss in mutual information due to clustering the rows and columns is

$$I(X;Y) - I(C_X;C_Y). \tag{36}$$

An optimal co-clustering minimizes this information loss. This information loss is equivalent to the KL-divergence between the probability distributions $p(x, y, c_x, c_y)$ and

$$q(x, y, c_x, c_y) = p(c_x, c_y)p(x|c_x)p(y|c_y), \tag{37}$$

$$I(X;Y) - I(C_X;C_Y) = D_{KL}(p(x, y, c_x, c_y)\|q(x, y, c_x, c_y)). \tag{38}$$

$p(\cdots)$ is used generically here to denote the probability (density) distribution of its arguments. That q as defined is in fact a probability distribution follows from observing that

$$\sum_x \sum_y \sum_{c_x} \sum_{c_y} q(x, y, c_x, c_y) = 1.$$

The distribution $p(x, y, c_x, c_y)$ can be written as

$$p(x, y, c_x, c_y) = p(c_x, c_y)p(x, y|c_x, c_y). \tag{39}$$

Comparing the distributions p and q above, we can see that q tries to approximate the conditional joint distribution $p(x, y|c_x, c_y)$ as a product of $p(x|c_x)$ and $p(y|c_y)$, i.e, the co-clustering tries to capture the joint distribution of X and Y through the individual row and column cluster distributions.

We consider the case of hard clustering, where each data sample belongs to only one cluster. In this case, $p(c_x|x) = 1$ if x is assigned to c_x, else $p(c_x|x) = 0$ (similarly for the column data y and clusters c_y). The distribution q preserves several marginal and conditional distributions of p as shown below, as a result of which it is a good approximation of p. Here $q(\cdots)$ with different arguments is used to denote marginal/conditional probability (density) distributions of $q(x, y, c_x, c_y)$, whereas $p(\cdots)$ with different arguments denotes the true distribution, e.g., $q(x, c_x) \equiv \sum_y \sum_{c_y} q(x, y, c_x, c_y)$ and $q(x|c_x) \equiv q(x, c_x)/q(c_x)$, whereas $p(x|c_x)$ is the probability density of the random variable $X|C_x = c_x$. A list of useful relationships between q and p follows.

$$q(c_x, c_y) = p(c_x, c_y), \quad q(x, c_x) = p(x, c_x), \quad q(y, c_y) = p(y, c_y). \tag{40}$$

$$p(x) = q(x), \quad p(y) = q(y), \quad p(c_x) = q(c_x), \quad p(c_y) = q(c_y). \tag{41}$$

$$p(x|c_x) = q(x|c_x), \quad p(y|c_y) = q(y|c_y). \tag{42}$$

$$p(c_y|c_x) = q(c_y|c_x), \quad p(c_x|c_y) = q(c_x|c_y). \tag{43}$$

$$q(y, c_y|c_x) = q(y|c_y)q(c_y|c_x). \tag{44}$$

$$q(x, y, c_x, c_y) = p(x, c_x)q(y, c_y|c_x). \tag{45}$$

$$q(x, c_x|c_y) = q(x|c_x)q(c_x|c_y). \tag{46}$$

$$q(x, y, c_x, c_y) = p(y, c_y)q(x, c_x|c_y). \tag{47}$$

Let $W(x)$ denote the cluster to which the row data sample x is assigned and $W(y)$ denote the cluster to which the column data sample y is assigned. The following algorithm attempts to minimize the information loss (38).

1. Start with a random partition and initialize $n \leftarrow 0$. Calculate $q^0(c_x, c_y)$, $q^0(c_x)$, $q^0(y|c_y)$ and use these to calculate $q^0(y, c_y|c_x)$ as given by (44) and $q^0(x, y, c_x, c_y)$ using (45).

2. Compute the cluster index for each row data sample x,
   $W^{n+1}(x) = \underset{c_x}{\operatorname{argmin}}\, D_{KL}[p(y, c_y|x)\|q^n(y, c_y|c_x)]$,
   $W^{n+1}(y) = W^n(y)$,
   $W^{n+2}(x) = W^{n+1}(x)$.

3. Using $W^{n+1}(x)$ and $W^{n+1}(y)$, compute $q^{n+1}(c_x, c_y)$, $q^{n+1}(x|c_x)$, $q^{n+1}(c_y)$, and use these to calculate $q^{n+1}(x, c_x|c_y)$ as given by (46).

4. Compute the cluster index for each column data sample y,
$W^{n+2}(y) = \underset{c_y}{\operatorname{argmin}} D_{KL}[p(x, c_x|y)\|q^{n+1}(x, c_x|c_y)]$.

5. Using $W^{n+2}(x)$ and $W^{n+2}(y)$, compute $q^{n+2}(c_x, c_y)$, $q^{n+2}(c_x)$, $q^{n+2}(y|c_y)$, and use these to calculate $q^{n+2}(y, c_y|c_x)$ as given by (44) and $q^{n+2}(x, y, c_x, c_y)$ using (45).

6. If $\left|D_{KL}[p(x, y, c_x, c_y)\|q^n(x, y, c_x, c_y)] - D_{KL}[p(x, y, c_x, c_y)\|q^{n+2}(x, y, c_x, c_y)]\right|$ is negligible, terminate, else $n \leftarrow n + 2$ and repeat Steps 2–6.

7 Multiway clustering using Bregman Divergences

Banerjee et al. [2] describe a generic framework using Bregman divergences to cluster multiple datasets that share a single relationship and extend it to multiple relations between the datasets.

Let $\phi : S \to \mathbb{R}$ be a C^1 strictly convex function defined on the convex set $S \subset \mathbb{R}^n$. The Bregman divergence $d_\phi : S \times \text{int}(S) \to [0, \infty)$ is defined as

$$d_\phi(z_1, z_2) = \phi(z_1) - \phi(z_2) - \langle z_1 - z_2, \nabla\phi(z_2)\rangle, \tag{48}$$

where $\nabla\phi$ is the gradient of ϕ and $\langle\cdot, \cdot\rangle$ is the dot product on $\mathbb{R}^n$. The *Bregman information* of a random variable Z with values $z \in S$ is defined as the expected Bregman divergence

$$I_\phi(Z) = E[d_\phi(Z, E[Z])]. \tag{49}$$

The Bregman information captures the "spread" or "information" in a random variable. Different divergence functions can be modeled by choosing an appropriate ϕ. If $\phi(z) = z \log z$, the divergence d_ϕ is called I-divergence and the Bregman information I_ϕ is proportional to KL-divergence. For squared Euclidean distance $\phi(z) = \|z\|_2^2$, I_ϕ is proportional to the squared Frobenius norm of the diagonal of the covariance matrix of Z.

Multiway clustering is a generalization of the co-clustering approach described in the previous section. In the co-clustering approach only two data sets are simultaneously clustered, while multiway clustering aims to cluster several datasets by preserving the relationships between them. We first consider the case when multiple datasets share a single relationship.

Notation
For $i = 1, \ldots, n$, let X_i be the random variable corresponding to the ith dataset, with range $\mathbf{N}_{m_i} = \{1, 2, \ldots, m_i\}$. Let $f : \mathbf{N}_{m_1} \times \cdots \times \mathbf{N}_{m_n} \to S$ be a function of all the random variables, and let $S \subset \mathbb{R}^k$ be convex. The random variable $Z = f(X_1, \ldots, X_n)$ is an n-dimensional tensor $Z \in S^{m_1 \times \cdots \times m_n}$. Let $X = (X_1, \ldots, X_n)$, $x = (x_1, \ldots, x_n)$, and w denote the measure induced on Z by $p(x)$ so that $w_{x_1,\ldots,x_n} = P(X_1 = x_1, \ldots, X_n = x_n) = P(Z = z_{x_1,\ldots,x_n}) = p(x_1, \ldots, x_n) = p(x)$. Let k_i denote the number of clusters desired for the dataset i. A multiway clustering of the datasets is defined as the n-tuple $\rho = (\rho_1, \ldots, \rho_n)$ where each $\rho_i : \{1, \ldots, m_i\} \to \{1, \ldots, k_i\}$ denotes a mapping from the data samples to their respective clusters. Let $\hat{X}_i = \rho_i(X_i)$ be the cluster random variable for dataset i that takes values in $\{1, \ldots, k_i\}$.

Multiway Clustering Formulation
Using the cluster random variables $\hat{X}_i$, we can construct a new tensor C with $E[C] = E[Z]$ that is an approximation of the original tensor Z. As with any information theoretic clustering algorithm, we seek to minimize the expected distortion between the original representation of the tensor and its approximate representation. In the case of multiway clustering this distortion is measured in the terms of the loss of Bregman information, and thus the objective function is the difference in the Bregman information between the original and the approximate tensors,

$$\mathcal{B} = E\big[d_\phi(Z, C)\big] = I_\phi(Z) - I_\phi(C). \tag{50}$$

An optimal multiway clustering minimizes this distortion.

In order to characterize C, we specify the information about the summary statistics of Z that are to be preserved by C. Let $\{V_s\}_{s=1}^r$ be a set of r random variables corresponding to the different summary statistics that are to

be preserved. This set is called a *multiway clustering basis*. In the simplest case, the basis is the singleton set $\{V_1\} = \{(\hat{X}_1, \ldots, \hat{X}_n)\}$, where $\hat{X}_i = \rho_i(X_i)\ \forall i$. This is called the block multiway clustering (BMC) basis. Information theoretic co-clustering is the special case of this model where $n = 2$, C is a two-dimensional tensor (matrix), and the clustering basis is $\{V_1\} = \{(C_x, C_y)\}$. Given the summary statistics, the set of potential solutions for C is

$$S_A = \big\{C' \;\big|\; E[Z] = E[C'],\ E[Z|V_s] = E[C'|V_s],\ s = 1, \ldots, r\big\}. \tag{51}$$

All the approximations $C' \in S_A$ preserve the conditional expectations associated with the multiway clustering basis. The best approximation in S_A is chosen according to the *maximum Bregman information principle* (MBI), which states that the best approximation is the one that has maximum Bregman information, i.e,

$$C = \operatorname*{argmax}_{C' \in S_A} I_\phi(C'). \tag{52}$$

A unique solution to the MBI problem always exists in the form

$$C = h_\phi(\Lambda^*, X, \rho(X)), \tag{53}$$

where $\Lambda^* = (\Lambda_1^*, \ldots, \Lambda_r^*)$ is the optimal Lagrange multiplier, $\rho(X) \equiv \big(\rho_1(X_1), \ldots, \rho_n(X_n)\big)$, and h_ϕ is a convex function. Even though the MBI problem has a unique solution, it can be expressed in closed form as a function of the summary statistics only for certain clustering bases such as BMC. The algorithm for multiway clustering follows.

1. Start with a random multiway clustering ρ and estimate for Λ.
2. For $i = 1, \ldots, n$, calculate the new cluster assignments for ρ_i via

$$\begin{aligned}(\rho_i(x_j), \Lambda^*) \quad = \quad & \operatorname*{argmin}_{(\hat{x}_j, \Lambda)} E_{X|X_i = x_j}\Big[\\ & d_\phi\big(Z, h_\phi\big(\Lambda, (X|X_i = x_j), (\rho(X)|\rho_i(X_i) = \hat{x}_j)\big)\big)\Big],\end{aligned}$$

   for $x_j = 1, \ldots, m_i$.
3. Repeat Step (2) until ρ and Λ converge; the solution is then given by (53).

Note that multiple relationships are already modeled by the case $z \in S \subset \mathbb{R}^k$ for $k > 1$. An alternative approach is to use multiple tensors Z_s to model the relationships. The MBI solution for approximating each individual tensor is the same as above. The objective function is the weighted sum of the differences between the Bregman informations of the original and corresponding approximate tensors, and an algorithm similar to the one above optimizes this distortion.

8 Discriminative Clustering

In the algorithms discussed so far, the primary data samples are merely discrete indices and their association with the auxiliary data is available in the form of a joint probability distribution. Typically this joint probability distribution is estimated from the co-occurrence frequencies of the primary and auxiliary data. For example, to cluster words by preserving information about documents, the number of times each word appears across the documents is counted and a joint probability distribution of word-document associations is estimated from these counts. When the primary data are real valued vectors and the auxiliary data is discrete, it is not possible to estimate the joint probability distribution through co-occurrence frequencies. One possible solution is to use kernel density estimation techniques [23] to estimate the joint probability distribution, but these techniques are extremely unreliable for high dimensional vectors. Sinkonnen et al. [26] describe a method to address this problem without any explicit joint density estimation.

Let $\mathbf{X}$ represent the vector valued random variable corresponding to the primary data whose instantiations are $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^n$, Y represent the discrete auxiliary data whose instantiations are $y \in \mathcal{Y} \subset \mathbb{N}$, and C represent the cluster labels whose k instantiations are $c \in \mathcal{C} = \mathbf{N}_k$. For ease of interpretation, we assume that each data vector is labeled with one or more class labels y. These labels y constitute the auxiliary data. Each cluster c has two prototypes—$\mathbf{m}_c$

for primary data, and $p(y|c)$ for auxiliary data. Note that all the algorithms for discrete data described so far use only the auxiliary data prototypes $p(y|c)$, as there is no explicit primary data. Additional prototypes $\mathbf{m}_c$, are required to model primary data in the form of real valued vectors.

In vector quantization [12], the average distortion between the prototypes of the clusters and the data is defined by

$$E = \sum_{c=1}^{k} \int_{\mathcal{X}} W_c(\mathbf{x}) d(\mathbf{x}, \mathbf{m}_c) p(\mathbf{x}) \, d\mathbf{x}, \tag{54}$$

where $W_c(\mathbf{x})$ is a cluster membership function, $p(\mathbf{x})$ is the probability density function of $\mathbf{X}$, and $d(\mathbf{x}, \mathbf{m}_c)$ is a distortion measure that measures the distance between the cluster prototype $\mathbf{m}_c$ and the data $\mathbf{x}$. As with the IB framework, the KL-divergence between the auxiliary data distributions of the primary data and the clusters is chosen as the distortion measure:

$$E_{KL} = \sum_{c=1}^{k} \int_{\mathcal{X}} W_c(\mathbf{x}) D_{KL}[p(y|\mathbf{x}) \| p(y|c)] p(\mathbf{x}) \, d\mathbf{x} \tag{55}$$

$$= -\sum_{c=1}^{k} \sum_{y \in \mathcal{Y}} \int_{\mathcal{X}} [W_c(\mathbf{x}) \log p(y|c)] p(y|\mathbf{x}) \, d\mathbf{x} + \text{constant}. \tag{56}$$

Note that the summation is over the auxiliary data labels y and the cluster labels c. In the case of hard clustering, each data point is assigned to only one cluster, $W_c(\mathbf{x}) = 1$ if $\mathbf{x}$ belongs to cluster c, otherwise $W_c(\mathbf{x}) = 0$. Since $W_c(\mathbf{x})$ is typically constructed to depend on $\mathbf{m}_c$ or $p(y|c)$, using such hard cluster memberships gives rise to a distortion function E_{KL} that is not continuously differentiable with respect to $\mathbf{m}_c$ or $p(y|c)$. To get smooth gradients that are required by many numerical optimization procedures, the cluster membership functions are parametrized as

$$W_c(\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp(f(\mathbf{x}; \mathbf{m}_c)), \tag{57}$$

where $Z(\mathbf{x})$ is a normalization function such that $\sum_c W_c(\mathbf{x}) = 1$. $f(\mathbf{x}; \mathbf{m}_c)$ is chosen based on the shape of clusters desired in the primary data space. The auxiliary data distributions of the clusters are multinomial with $\sum_y p(y|c) = 1$. To enforce this constraint, these probabilities are expressed using the soft max parametrization

$$\log p(y|c) = \gamma_{cy} - \log \left(\sum_{m \in \mathcal{Y}} e^{\gamma_{cm}} \right), \tag{58}$$

where one γ_{cy} is fixed to avoid degeneracy. From equations (57) and (58), E_{KL} is thus parametrized in terms of the prototypes $\mathbf{m}_c$ and constants $\gamma_{cy} \propto \log p(y|c)$. The n-th step of an optimization algorithm that minimizes E_{KL} with respect to $\mathbf{m}_c$ and γ_{cy} has the form:

1. Choose a data sample $\mathbf{x}$ at random and let its class label be y.
2. Choose two clusters c and c' at random.
3. Update the variables by

$$\mathbf{m}_c^{(n+1)} = \mathbf{m}_c^{(n)} - \alpha^{(n)} \left[\frac{\partial f(\mathbf{x}; \mathbf{m}_c^{(n)})}{\partial \mathbf{m}_c} \right] \log \left(\frac{p(y|c')}{p(y|c)} \right),$$

for $s \in \mathcal{Y}$

$$\gamma_{cs}^{(n+1)} = \gamma_{cs}^{(n)} - \alpha^{(n)} [p(s|c) - \delta_{ys}],$$

where δ_{ys} is the Kronecker delta and $\alpha^{(n)}$ is a line search stepsize. Perform a similar update for $\mathbf{m}_{c'}$ and $\gamma_{c's}$. No update is performed if $c = c'$.

An alternative formulation for this clustering problem is given by Kaski et al. [20]. The primary data is a *sequence* of vectors $\bar{\mathbf{x}} = \big(\mathbf{x}^{(i)}\big)_{i=1}^{N}$ for which $\mathbf{x}$ denotes a generic element, and $\bar{y} = \big(y^{(i)}\big)_{i=1}^{N}$ is the corresponding *sequence* of auxiliary class data with generic class $y \in \mathcal{Y} = \big\{y_i\big\}_{i=1}^{M}$. For each cluster $c = 1, \ldots, k$, let ψ_c be the vector with components $\psi_{c,y} = p(y|c)$, $y \in \mathcal{Y}$, let $\psi = \big(\psi_1, \ldots, \psi_k\big)$, and let $\overline{\psi}_c$ be the set of all such possible vectors ψ_c. View $\psi_{c,y}$ as the value of a nonnegative random variable $\Psi_{c,y}$, where the $\Psi_{c,y}$ satisfy $\sum_y \Psi_{c,y} = 1 \; \forall c$. Ψ_c is the vector random variable with components $\Psi_{c,y}$, and Ψ is the vector random variable with components Ψ_c. Given the primary data $\bar{\mathbf{x}}$ and the auxiliary data $\bar{y}$, the posterior of the primary data prototypes $\{\mathbf{m}_c\}_{c=1}^{k}$ is obtained by integrating out all the auxiliary data prototypes ψ_c from the joint conditional distribution,

$$\begin{aligned} A = p(\{\mathbf{m}_c\}|\bar{\mathbf{x}}, \bar{y}) &= \int_{\overline{\psi}_1} \cdots \int_{\overline{\psi}_k} p(\{\mathbf{m}_c\}, \psi|\bar{\mathbf{x}}, \bar{y})\, d\psi_1 \cdots d\psi_k \\ &\propto \int_{\overline{\psi}_1} \cdots \int_{\overline{\psi}_k} p(\bar{y}|\{\mathbf{m}_c\}, \psi, \bar{\mathbf{x}})\, p(\{\mathbf{m}_c\}, \psi, \bar{\mathbf{x}})\, d\psi_1 \cdots d\psi_k \\ &\propto \int_{\overline{\psi}_1} \cdots \int_{\overline{\psi}_k} p(\bar{y}|\psi) \prod_{c=1}^{k} p(\psi_c)\, d\psi_1 \cdots d\psi_k \end{aligned} \tag{59}$$

assuming that $\bar{y}$ is independent of $\{\mathbf{m}_c\}$ and $p(\{\mathbf{m}_c\}, \psi, \bar{\mathbf{x}}) \propto p(\psi) = \prod_{c=1}^{k} p(\psi_c)$.

It is reasonable to assume that Ψ_c has a Dirichlet distribution with the same parameter n_c^0 for each component $\psi_{c,y}$, in which case $p(\psi_c) \propto \prod_{y \in \mathcal{Y}} \psi_{c,y}^{n_c^0 - 1}$ and

$$A \propto \int_{\overline{\psi}_1} \cdots \int_{\overline{\psi}_k} \prod_{c=1}^{k} \prod_{y \in \mathcal{Y}} \psi_{c,y}^{n_c^0 + n_{cy} - 1}\, d\psi_1 \cdots d\psi_k = \prod_{c=1}^{k} \frac{\prod_{y \in \mathcal{Y}} \Gamma\big(n_c^0 + n_{cy}\big)}{\Gamma\Big(\sum_{y \in \mathcal{Y}} \big(n_c^0 + n_{cy}\big)\Big)}, \tag{60}$$

where n_{cy} is the number of data vectors of class y in cluster c, and $\sum_{c,y} n_{cy} = N$.

The function A is the posterior probability of the cluster prototypes in the primary data space given both the primary and auxiliary data. The auxiliary class information is encoded into the objective function by the term n_{cy}. Since the joint distribution $p(\mathbf{x}, y)$ of the primary and auxiliary data is unavailable, interpreting the auxiliary data distribution $\psi_{c,y}$ of a cluster is not meaningful. Instead, the quality of the clusters with respect to the auxiliary classes is analyzed through a two-dimensional contingency table where the rows represent the clusters and the columns represent the classes. The Bayes factor of this contingency table is proportional to the objective function in (60).

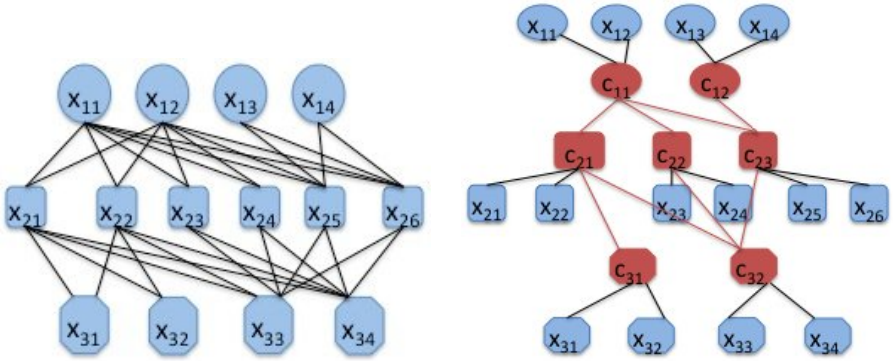
In order to parametrize A as a continuously differentiable function of the prototypes $\mathbf{m}_c$, smooth cluster membership probabilities $W_c(\mathbf{x})$ as defined in (57) are used. Then, the number of samples of class y in cluster c is defined as

$$n_{cy} = \sum_{\mathbf{x}^{(i)}: y^{(i)} = y} W_c(\mathbf{x}^{(i)}).$$

Using these substitutions for $W_c(\mathbf{x})$ and n_{cy} in the expression proportional to A in (60), and then taking the logarithm, gives the objective function that is maximized using standard numerical optimization techniques to find the desired prototypes $\mathbf{m}_c$.

9 Clustering with k-partite graphs

Long et al. [21] propose a framework for clustering multiple datasets simultaneously by modeling the many-to-many relationships between pairs of datasets. Let X_1, X_2, …, X_n be the random variables corresponding to the datasets, and C_1, C_2, …, C_n be their corresponding cluster random variables. Let m_1, m_2, …, m_n be the number of data samples in each dataset and k_1, k_2, k_3, …, k_n be the number of clusters in each dataset. Dataset i is then represented



(a)Input data graph (b) Cluster graph

Figure 4: Clustering with k-partite graphs

as $\mathcal{X}_i = \{x_{ih}\}_{h=1}^{m_i}$ and its clusters are represented as $\mathcal{C}_i = \{c_{ip}\}_{p=1}^{k_i}$. The samples across any pair of datasets i and j have many-to-many relationships. Such datasets can be represented by a k-partite graph as shown in Fig. (4)(a). A k-partite graph is a graph whose vertices can be partitioned into k disjoint sets so that no two vertices within the same set are adjacent. Let $G^X = \big(\mathcal{X}_1, \ldots, \mathcal{X}_n, E^X\big)$ be the k-partite graph corresponding to the data samples. The samples in each dataset constitute the partitions of the graph as shown in Fig. (4)(a). The many-to-many relationships between the datasets are represented by the edges E^X in the graph. For example, in the Fig. (4)(a) the relationships between the datasets $\mathcal{X}_1$ and $\mathcal{X}_2$ are represented by the edges between the sets of vertices $\{x_{1h}\}_{h=1}^{4} = \mathcal{X}_1$ and $\{x_{2l}\}_{l=1}^{6} = \mathcal{X}_2$. Call this graph the input data graph. The clustered representation of the datasets also represents a k-partite graph, $G^C = \big(\mathcal{X}_1, \ldots, \mathcal{X}_n, \mathcal{C}_1, \ldots, \mathcal{C}_n, E^C\big)$ as shown in Fig. (4)(b), and called the cluster graph. This graph has *cluster vertices* labeled c in addition to the *data vertices* labeled x. This graph has the following properties.

1. Each data vertex is adjacent to one and only one cluster vertex corresponding to the dataset. The edge weight between any data vertex and its corresponding cluster vertex is set to unity.

2. The cluster vertices across datasets are adjacent to one another if their corresponding data samples are adjacent in the input data graph. The weights assigned to the edges between the cluster vertices summarize the relationships between the data samples assigned to the clusters across the datasets.

For example, in Fig. (4)(a) the data sample x_{13} in dataset $\mathcal{X}_1$ is related to x_{25} in dataset $\mathcal{X}_2$. The edge weight $e^X(x_{13}, x_{25})$ for the edge $\{x_{13}, x_{25}\}$ in the input graph represents this relationship and is set to the weight of the relationship. In the cluster graph x_{13} is assigned to the cluster c_{12}, and x_{25} is assigned to the cluster c_{23}. As a result the edge weights $e^C(x_{13}, c_{12})$ and $e^C(x_{25}, c_{23})$ are set to one in the cluster graph. The weight $e^C(c_{12}, c_{23})$ for the edge between the clusters c_{12} and c_{23} is determined by the algorithm based on the other samples assigned to the clusters. When the algorithm terminates, this weight represents the average weight of the relationships between the data samples assigned to these clusters. The goal of the clustering algorithm is to assign the data samples in each individual dataset to their clusters while maintaining the many-to-many relationships across the datasets at the cluster level also. The many-to-many relationships between the clusters summarize the many-to-many relationships between the data samples assigned to them and this is captured by the edge weights between the cluster vertices in the cluster graph.

For some Bregman divergence d_ϕ, define the distance between the input data graph G^X and its cluster graph G^C by

$$\mathcal{L} = \sum_{1 \le i < j \le n} \sum_{\substack{x_{ih} \in \mathcal{X}_i, x_{jl} \in \mathcal{X}_j \\ e^C(x_{ih}, c_{ip}) = 1 \\ e^C(x_{jl}, c_{jq}) = 1}} d_\phi\big(e^X(x_{ih}, x_{jl}), e^C(c_{ip}, c_{jq})\big).$$

For ease of computation, the weights of edges between any pair of datasets $\mathcal{X}_i$ and $\mathcal{X}_j$ in the input data graph are represented by the weight matrix $A^{(i,j)} \in \mathbb{R}^{m_i \times m_j}$, where $A^{(i,j)}_{hl} = e^X(x_{ih}, x_{jl})$. Similarly, the weights of the

edges between the cluster vertices across the datasets $\mathcal{X}_i$ and $\mathcal{X}_j$ in the cluster graph G^C are represented by the matrix $B^{(i,j)} \in \mathbb{R}^{k_i \times k_j}$, where $B^{(i,j)}_{pq} = e^C(c_{ip}, c_{jq})$. The assignment of the data samples to the clusters in each individual dataset is represented by the indicator matrices $F^{(i)} \in \{0,1\}^{m_i \times k_i}$, where $F^{(i)}_{hp} = e^C(x_{ih}, c_{ip})$. The distance $\mathcal{L}$ between the data graph and the cluster graph is computed as

$$\mathcal{L} = \sum_{1 \le i < j \le n} \hat{d}_\phi \Big(A^{(i,j)}, F^{(i)} B^{(i,j)} \big(F^{(j)}\big)^T \Big), \tag{61}$$

where $\hat{d}_\phi$ means d_ϕ applied componentwise and summed. An optimal clustering of the datasets satisfies

$$\big(F^{(i)}\big)^T \Big(F^{(i)} B^{(i,j)} \big(F^{(j)}\big)^T - A^{(i,j)} \Big) F^{(j)} = 0, \qquad 1 \le i < j \le n, \tag{62}$$

or (since $m_i \ge k_i$ implies $F^{(i)}$ has full rank)

$$B^{(i,j)} = \Big(\big(F^{(i)}\big)^T F^{(i)} \Big)^{-1} \big(F^{(i)}\big)^T A^{(i,j)} F^{(j)} \Big(\big(F^{(j)}\big)^T F^{(j)} \Big)^{-1}. \tag{63}$$

An exchange algorithm to locally minimize the objective function given in (61) starts with random clusters in each dataset and performs the following steps in each iteration.

1. Calculate the current objective function value $\mathcal{L}^{\text{current}}$ given in (61).
2. Change the cluster assignments in each dataset. For each $i = 1, \ldots, n$, and $h = 1, \ldots, m_i$:
   - for each $p = 1, \ldots, k_i$, calculate the objective function value $\mathcal{L}_p$ obtained by removing the data sample x_{ih} from its current cluster c_{iq} and assigning it to the cluster c_{ip};
   - assign x_{ih} to the cluster c_{ip*}, $p* = \operatorname{argmin}_p \mathcal{L}_p$, giving the minimum value for the objective function $\mathcal{L}_p$, and update the indicator matrix via $F^{(i)}_{hq} := 0$, $F^{(i)}_{hp*} := 1$.
3. Update the weights for the edges between the cluster vertices in G^C according to (63).
4. Calculate the new objective function $\mathcal{L}^{\text{new}}$. Terminate if $\mathcal{L}^{\text{new}} = \mathcal{L}^{\text{current}}$, else repeat the steps with the new cluster assignments.

10 Clustering binary matrices — cross associations

	y_1	y_2	y_3	y_4	y_5	y_6
x_1	1	0	0	1	0	1
x_2	0	1	1	0	1	0
x_3	1	0	0	1	0	1
x_4	1	1	1	1	1	1
x_5	0	1	1	0	1	0
x_6	1	0	1	1	1	1

(a) Input binary matrix

	y_1	y_4	y_6	y_2	y_3	y_5
x_1	1	1	1	0	0	0
x_3	1	1	1	0	0	0
x_2	0	0	0	1	1	1
x_5	0	0	0	1	1	1
x_4	1	1	1	1	1	1
x_6	1	1	1	0	1	1

(b) Discovered cross associations

Figure 5: Cross associations. Dense regions of ones or zeros obtained by clustering rows and columns of a binary matrix simultaneously, shown by the rectangles in (b).

Chakrabarti et al. [4] present an algorithm to discover hidden structure in binary matrices. This algorithm is similar to co-clustering described in Section 6, but is targeted towards clustering rows and columns of large sparse

binary matrices. The row and column clusters that the algorithm discovers divide a row and column permutation of a binary matrix into homogeneous rectangular regions (submatrices) called cross associations. Ideally the cross associations are dense regions of either ones or zeros as shown in Figure (5)(b).

Suppose the rows of the $m \times n$ binary matrix are grouped into k clusters and the columns are grouped into l clusters giving rise to $k \cdot l$ cross associations. Given a cross association D_{ij}, let $n_0(D_{ij})$, $n_1(D_{ij})$ be the number of zeros, ones in D_{ij}, respectively, and the total number of entries be $n(D_{ij}) = n_0(D_{ij}) + n_1(D_{ij})$. Let $X_{D_{ij}}$ be the Bernoulli (binary) random variable taking values 0 or 1 associated with D_{ij}. The probability of zeros in D_{ij} is $P(X_{D_{ij}} = 0) = n_0(D_{ij})/n(D_{ij})$ and that of ones is $P(X_{D_{ij}} = 1) = n_1(D_{ij})/n(D_{ij})$. Using arithmetic coding [22], the cost of compressing D_{ij} is given by

$$\begin{aligned} C(D_{ij}) &\equiv -n_0(D_{ij}) \log_2\big(P(X_{D_{ij}} = 0)\big) - n_1(D_{ij}) \log_2\big(P(X_{D_{ij}} = 1)\big) && (64) \\ &\equiv n(D_{ij}) H(X_{D_{ij}}), && (65) \\ & && (66) \end{aligned}$$

and the total compression cost

$$\mathcal{F} = \sum_{i=1}^{k} \sum_{j=1}^{l} C(D_{ij}) \tag{67}$$

of the model is obtained by summing up the individual cross association costs. Minimizing $\mathcal{F}$ leads to optimal row and column clusters and in turn optimal cross associations. The cross association algorithm starts with a random set of row and column clusters. In each iteration, the following steps are performed.

1. Hold the column clusters constant and adjust the row clusters. Remove each row from its current cluster and assign it to the cluster that results in the maximum decrease in the current objective function value $\mathcal{F}$ from (67), updating the row clusters and the current objective function value.
2. Hold the row clusters constant and adjust the column clusters similarly to Step 1.
3. Terminate if there is no change in the objective function value, otherwise continue.

This is a greedy algorithm, similar to k-means, that will terminate in a finite number of iterations because there are finitely many row and column permutations, and will converge to only a local minimum point (row and column cluster assignments) of $\mathcal{F}$. Precisely, the number of possible cross association combinations (ignoring row and column permutations within a cross association) is $\left\{ {m \atop k} \right\} \left\{ {n \atop l} \right\}$, the product of Stirling numbers of the second kind [11].

11 Clustering with constraints

Davidson et al. [5] present a modified k-means algorithm that takes into account certain constraints on the data samples being grouped together while clustering. The constraints that this algorithm models are of two kinds:

1. **Must-link:** Each must-link constraint specifies a pair of distinct samples that must be in the same cluster.
2. **Cannot-link:** Each cannot-link constraint specifies a pair of distinct samples that cannot be in the same cluster.

Let $\mathbf{x}_1, \ldots, \mathbf{x}_n$ be the n data sample vectors to be clustered into the k clusters $c_1, \ldots, c_k$ whose prototypes are $\mathbf{m}_1, \ldots, \mathbf{m}_k$. The clustering has to satisfy r constraints represented by $\phi_1, \ldots, \phi_r$. Accompanying each constraint ϕ_l is an indicator variable b_l that indicates whether ϕ_l is of type must-link ($b_l = 1$) or cannot-link ($b_l = 0$). Suppose the constraint ϕ_l involves the data samples $\mathbf{x}_i$ and $\mathbf{x}_j$ and $b_l = 1$, then $\mathbf{x}_i$ and $\mathbf{x}_j$ must be in the same cluster. If $b_l = 0$, $\mathbf{x}_i$ and $\mathbf{x}_j$ cannot be clustered together. The function $g_1(l)$ accepts the constraint index l as input and returns the cluster index of the first data sample in the constraint. Similarly $g_2(l)$ returns the cluster index of the second data sample in the constraint ϕ_l.

The objective function in traditional k-means is given by

$$\mathcal{L} = \sum_{j=1}^{k} E_j, \tag{68}$$

$$E_j = \frac{1}{2} \sum_{\{i:\mathbf{x}_i \in c_j\}} \|\mathbf{m}_j - \mathbf{x}_i\|^2. \tag{69}$$

E_j represents the sum of squared distances between the cluster j prototype and the samples assigned to cluster j, and $\mathcal{L}$ represents the sum over all the clusters. The k-means algorithm minimizes $\mathcal{L}$. Given the constraints $\phi_1, \ldots, \phi_r$, the authors propose a modified objective function

$$\begin{aligned} \mathcal{L}' &= \sum_{j=1}^{k} {E'}_j, \\ {E'}_j &= \frac{1}{2}\left[\sum_{\{i:\mathbf{x}_i \in c_j\}} T^1_{ji} + \sum_{\{l:g_1(l)=j\}} (T^2_{jl} \times T^3_{jl}) \right], \end{aligned} \tag{70}$$

where, taking $0^0 \equiv 1$,

$$\begin{aligned} T^1_{ji} &= \|\mathbf{m}_j - \mathbf{x}_i\|^2, \\ T^2_{jl} &= \left[\|\mathbf{m}_j - \mathbf{m}_{g_2(l)}\|^2 \, (1 - \delta(g_1(l), g_2(l)))\right]^{b_l}, \\ T^3_{jl} &= \left[\|\mathbf{m}_j - \mathbf{m}_{h(g_2(l))}\|^2 \, \delta(g_1(l), g_2(l))\right]^{1-b_l}. \end{aligned}$$

Here $\delta(a_1, a_2)$ is the Kronecker delta function, defined as $\delta(a_1, a_2) = 1$ if $a_1 = a_2$, and 0 otherwise. The function $h(j)$ returns the index of the cluster (other than j) whose prototype is closest to the prototype of cluster j.

For each cluster c_j , the term T^1_{ji} represents the sum of squared distances of the samples from the cluster prototype, which is the same as in the original k-means objective function. The second term T^2_{jl} is the penalty due to the must-link constraints violated in c_j while the third term T^3_{jl} is the penalty due to the cannot-link constraints violated in c_j. Suppose $\mathbf{x}_p$ and $\mathbf{x}_q$ are to be clustered together (by must-link constraint l), but in the current iteration $\mathbf{x}_p \in c_u$ and $\mathbf{x}_q \in c_v$. The penalty term T^2_{ul} is then set to the squared distance between the prototypes of c_u and c_v. Similarly, suppose $\mathbf{x}_p$ and $\mathbf{x}_q$ cannot be clustered together, but in the current iteration $\mathbf{x}_p, \mathbf{x}_q \in c_u$. Let c_w be the cluster whose prototype is closest to that of c_u, then the penalty term T^3_{ul} is set to the squared distance between the prototypes of c_u and c_w.

In order to minimize the objective function $\mathcal{L}'$, we start with a random set of clusters and iteratively update the cluster prototypes so that the new clusters further minimize $\mathcal{L}'$. The gradient of (70) with respect to the prototypes is set to zero, and the solution of this equation provides the updated prototypes that further minimize $\mathcal{L}'$ in each iteration. The updated prototypes based on current cluster assignments are given by

$$\mathbf{m}_j = \frac{\mathbf{y}_j}{z_j}, \tag{71}$$

where

$$\begin{aligned} \mathbf{y}_j &= \sum_{\{i:\mathbf{x}_i \in c_j\}} \mathbf{x}_i \\ &\quad + \sum_{\{l:g_1(l)=j\}} \left[\mathbf{m}_{g_2(l)} \, b_l(1 - \delta(j, g_2(l))) + \mathbf{m}_{h(g_2(l))} \, (1 - b_l)\delta(j, g_2(l))\right], \\ z_j &= |c_j| + \sum_{\{l:g_1(l)=j\}} \left[b_l(1 - \delta(j, g_2(l))) + (1 - b_l)\delta(j, g_2(l))\right]. \end{aligned}$$

12 Disparate clusterings

Jain et al. [18] present an approach to partition a given data set into several alternate or disparate clusterings that are different from each other. Each clustering provides an alternate view of the data and such clusterings are useful to uncover the different groups inherent in the data.

Given a data set $\mathcal{G} = \{\mathbf{g}_1, \ldots, \mathbf{g}_n\}$, $\mathbf{g}_i \in \mathbb{R}^l$, we wish to uncover two clusterings $C^{(x)} = \{c_1^{(x)}, \ldots, c_r^{(x)}\}$ and $C^{(y)} = \{c_1^{(y)}, \ldots, c_s^{(y)}\}$ with r and s clusters, respectively. Let $\mathbf{m}_1^{(x)}, \ldots, \mathbf{m}_r^{(x)}$ be the prototypes for the clusters in $C^{(x)}$ and $\mathbf{m}_1^{(y)}, \ldots, \mathbf{m}_s^{(y)}$ for those in $C^{(y)}$.

The objective function for disparate clustering that the authors propose is

$$\begin{aligned}\mathcal{F} &= \sum_{i=1}^{r} \sum_{\{k:\mathbf{g}_k \in c_i^{(x)}\}} \|\mathbf{g}_k - \mathbf{m}_i^{(x)}\|^2 + \sum_{j=1}^{s} \sum_{\{k:\mathbf{g}_k \in c_j^{(y)}\}} \|\mathbf{g}_k - \mathbf{m}_j^{(y)}\|^2 \\ &\quad + \lambda \sum_{i=1}^{r} \sum_{j=1}^{s} \left({\mathbf{m}_i^{(x)}}^T \mathbf{m}_j^{(y)} \right)^2.\end{aligned}$$

The intuition behind this objective function is that if the prototype vectors of the clusterings are orthogonal to one another, the cluster assignments (based on nearest distance to prototype) generated are independent. A k-means style iterative algorithm is used to minimize $\mathcal{F}$, and the prototypes are updated in each iteration using the equations

$$\begin{aligned}\mathbf{m}_i^{(x)} &= \left(I + \frac{\lambda}{|c_i^{(x)}|} \sum_{j=1}^{s} \mathbf{m}_j^{(y)} {\mathbf{m}_j^{(y)}}^T \right)^{-1} \mathbf{m}_i^{(x)}, \\ \mathbf{m}_j^{(y)} &= \left(I + \frac{\lambda}{|c_j^{(y)}|} \sum_{i=1}^{r} \mathbf{m}_i^{(x)} {\mathbf{m}_i^{(x)}}^T \right)^{-1} \mathbf{m}_j^{(y)}.\end{aligned}$$

These solutions are obtained by setting the gradient of $\mathcal{F}$ with respect to the prototypes to zero. In each iteration, we first update the prototypes using the equations above. Then the data samples are reassigned to the nearest clusters in the two clusterings and the procedure is repeated till convergence.

13 Discussion

A comparative survey of the above methods yields the following key characteristics:

1. clustering multiple datasets with relationships between them
   (IB, multi-variate IB, multi-way clustering, discriminative clustering, disparate clustering),

2. clustering relational data
   (co-clustering, clustering k-partite graphs, cross-associations),

3. explicit constraints
   (must-link and cannot-link constraints).

The variety of formulations presented here reveals that constrained clustering (either explicitly or indirectly specified) is a key emerging theme in the last decade for knowledge discovery and data mining. Relational data is only going to become more prominent with the advent of social networks, web mining, and other domains where relationships are key elements of data being modeled. The nature of constraints are both prescriptive (e.g., to preserve information about an auxiliary variable, to capture overlaps in induced similarity) as well as prohibitive (e.g., to disallow data items from being clustered together and to infer disparate clusterings). We anticipate that, as the scope of clustering applications grows, newer formulations will be invented and lead to a unified theory of modeling, exploiting, and enforcing constraints over data/clusters.

References

[1] A. A. Alizadeh, M. B. Eisen, R. E. Davis, C. Ma, I. S. Lossos, A. Rosenwald, J. C. Boldrick, H. Sabet, T. Tran, X. Yu, J. I. Powell, L. Yang, G. E. Marti, T. Moore, J. Hudson, L. Lu, D. B. Lewis, R. Tibshirani, G. Sherlock, W. C. Chang, T. C. Greiner, D. D. Weisenburger, J. O. Armitage, R. Warnke, R. Levy, W. Wilson, M. R. Grever, J. C. Byrd, D. Botstein, P. O. Brown, and L. M Staudt. Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, Vol. 403(6769):503–11, 2000.

[2] A. Banerjee, S. Basu, and S. Merugu. Multi-way clustering on relation graphs. In *Proceedings of the 7th SIAM International Conference on Data Mining*, pages 69–79, 2007.

[3] A. Banerjee and J. Ghosh. Scalable clustering algorithms with balancing constraints. *Data Mining and Knowledge Discovery*, 13(3):365–395, 2006.

[4] D. Chakrabarti, S. Papadimitriou, D. S. Modha, and C. Faloutsos. Fully automatic cross-associations. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 79–88, 2004.

[5] I. Davidson and S. S. Ravi. Clustering with constraints: Feasibility issues and the k-means algorithm. In *Proceedings of the 7th SIAM International Conference on Data Mining*, pages 25–35, 2005.

[6] I. S. Dhillon, S. Mallela, and D. S. Modha. Information theoretic co-clustering. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 89–98, 2003.

[7] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience Publication, 2000.

[8] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *PNAS*, 95(25):14863–14868, 1998.

[9] B. J. Frey and D. Dueck. Clustering by Passing Messages Between Data Points. *Science*, 315(5814):972–976, 2007.

[10] N. Friedman, O. Mosenzon, N. Slonim, and N. Tishby. Multivariate information bottleneck. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, pages 152–161, 2001.

[11] R. L. Graham, D. E. Knuth, and O. Patashnik. *Concrete Mathematics*. Addison-Wesley, 1994.

[12] R. M. Gray. Vector quantization. *IEEE ASSP Magazine*, pages 4–29, April 1984.

[13] J. A. Hartigan. *Clustering Algorithms*. Wiley, 1975.

[14] V. R. Iyer, M. B. Eisen, D. T. Ross, G. Schuler, T. Moore, J. C. F. Lee, J. M. Trent, L. M. Staudt, J. Hudson Jr., M. S. Boguski, D. Lashkari, D. Shalon, D. Botstein, and P. O. Brown. The transcriptional program in the response of human fibroblasts to serum. *Science*, 283(5398):83–87, 1999.

[15] A. K. Jain. Data Clustering: 50 years beyond K-means. *Pattern Recognition*, 2009. to appear.

[16] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice hall, 1988.

[17] A. K. Jain, M. N. Murty, and P. J. Flynn. Data Clustering: A Review. *ACM Computing Surveys*, Vol. 31(3):264–323, Sep 1999.

[18] P. Jain, R. Meka, and I. S. Dhillon. Simultaneous unsupervised learning of disparate clusterings. In *Proceedings of the 8th SIAM International Conference on Data Mining*, pages 89–98, 2008.

[19] S. Kaski, J. Nikkilä, J. Sinkkonen, L. Lahti, J. E. A. Knuuttila, and C. Roos. Associative clustering for exploring dependencies between functional genomics data sets. *IEEE/ACM TCBB*, 2(3):203–216, 2005.

[20] S. Kaski, J. Sinkkonen, and A. Klami. Discriminative clustering. *Neurocomputing*, 69(1–3):18–41, 2005.

[21] B. Long, X. Wu, Z. Zhang, and P. S. Yu. Unsupervised learning on k-partite graphs. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 317–326, 2006.

[22] J. Rissanen and G. G. Langdon Jr. Arithmetic coding. *IBM journal of research and development*, 23:149–162, 1979.

[23] D. W. Scott. *Multivariate Density Estimation : Theory, Practice, and Visualization*. Wiley-Interscience, 1992.

[24] J. Sese, Y. Kurokawa, M. Monden, K. Kato, and S. Morishita. Constrained clusters of gene expression profiles with pathological features. *Bioinformatics*, 20(17):3137–3145, 2004.

[25] B. W. Silverman. *Density Estimation*. London: Chapman and Hall, 1986.

[26] J. Sinkkonen and S. Kaski. Clustering based on conditional distributions in an auxiliary space. *Neural Computation*, 14(1):217–239, 2002.

[27] N. Slonim and N. Tishby. Agglomerative information bottleneck. In *Advances in Neural Information Processing Systems 12*, pages 617–623, 1999.

[28] S. Still, W., and L. Bottou. Geometric clustering using the information bottleneck method. In *Advances in Neural Information Processing Systems 16*, pages 27–37, 2003.

[29] N. Tishby, F. C. Pereira, and W. Bialek. The Information Bottleneck Method. In *the 37th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

[30] J.W. Tukey. *Exploratory Data Analysis*. Addison-Wesley, 1977.

[31] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl. Constrained k-means clustering with background knowledge. In *Proceedings of the 18th International Conference on Machine Learning*, pages 577–584, 2001.

[32] Y. Xu, V. Olman, and D. Xu. Clustering gene expression data using a graph-theoretic approach: an application of minimum spanning trees. *Bioinformatics*, 18(4):536–545, 2002.