

Learning Simple Relations: Theory and Applications

Pavel Berkhin and Jonathan D. Becher

Accrue Software, Inc., 48634 Milmont Drive, Fremont, CA 94538
pavel.berkhin@accrue.com, jonathan.becher@accrue.com

Abstract – In addition to classic clustering algorithms, many different approaches to clustering are emerging for objects of special nature. In this article we deal with the grouping of rows and columns of a matrix with non-negative entries. Two rows (or columns) are considered similar if corresponding cross-distributions are close. This grouping is a dual clustering of two sets of elements, row and column indices. The introduced approach is based on the minimization of reduction of mutual information contained in a matrix that represents the relationship between two sets of elements. Our clustering approach contains many parallels with K-Means clustering due to certain common algebraic properties. The obtained results have many applications, including grouping of Web visit data.

Keywords – Clustering, K-Means, Mutual Information, Web Mining, Data Mining

1. Introduction

There are a broad variety of classic clustering algorithms that are applicable for objects of general structure [15, 23, 25]. For objects of a special nature, new approaches to clustering are emerging [5, 16, 20, 27, 10]. In this article we are concerned with the clustering of rows $\mathbf{x} \in \mathbf{X}$ and columns $\mathbf{y} \in \mathbf{Y}$ of a rectangular matrix with non-negative entries [32, 11]. In the context of a two dimensional OLAP cube, such a matrix represents the simplest relationship between two nominal categorical variables, in which one variable (dimension) is the row and the other is the column. We want to come up with a systematic approach that groups together elements \mathbf{x} (and \mathbf{y}) exhibiting similar behavior with respect to a given relationship. This can be considered as learning the structure of the relationship and ultimately achieves simplification without significantly violating the relationship's patterns.

In set theory relation is the subset of the Cartesian product $\mathbf{X} \times \mathbf{Y}$. Any subset is defined by its indicator function — in our case by a matrix with entries equal to 1 or 0 indicating whether the corresponding element belongs to the relation. A non-negative matrix generalizes this concept.

To illustrate the point, consider the following model: a hall has several entrance doors \mathbf{x} (sources) and several exit doors \mathbf{y} (sinks) that physically may or may not coincide with the entrances. Every time somebody gets in through an entrance door \mathbf{x} , stays and leaves through an exit door \mathbf{y} , a count $N_{\mathbf{x}\mathbf{y}}$ is incremented. After a while the process is stopped. The *frequency count* (or *contingency*) matrix $N_{\mathbf{x}\mathbf{y}}$ represents the averaged pattern of behavior or the relationship between entrances and exits (or sources and sinks). If the behavior is totally random, the row and column distributions will all be uniform. Any asymmetric behavior implies certain associations between \mathbf{x} and \mathbf{y} . The question we are interested in is to group \mathbf{x} (and symmetrically \mathbf{y}) in a way that will represent similarity of their preferences with respect to \mathbf{y} (correspondingly \mathbf{x}). This model can be easily generalized to a situation when each individual (particle) passing through the hall carries

some positive weight and instead of just incrementing the count N_{xy} , the passing weight is added to it. In any case, the resulting matrix has non-negative entries, representing the flow from \mathbf{x} to \mathbf{y} .

Each element \mathbf{x} is characterized by $|\mathbf{Y}|$ numerical attributes. We want to cluster together elements \mathbf{x} with similar (conditional) distributions of \mathbf{y} -outcomes N_{xy}/N_x , where (marginal) entry N_x represents a total flow through the door \mathbf{x} . If some of the distributions are the same, corresponding \mathbf{x} are considered indistinguishable. This simplifies the structure of data. In other words we want to cluster together elements \mathbf{x} with common behavior across another dimension. While general clustering can be applied, all attributes are components of a single probability distribution, which strongly suggests some special approach.

We are motivated by the following interpretation of the above model. Consider two jointly distributed nominal categorical random variables \mathbf{X}, \mathbf{Y} with $|\mathbf{X}|$ and $|\mathbf{Y}|$ outcomes correspondingly and a joint distribution $N_{xy}/N_{..}$. Mutual information $\mathbf{I}(\mathbf{X}, \mathbf{Y})$, as defined below, measures the association of \mathbf{X} and \mathbf{Y} . It reflects a level of asymmetry in a contingency matrix. The exact definition expresses mutual information in terms of matrix N_{xy} alone, and so references to random variables can be omitted. We group together different elements \mathbf{x} into groups $G_a, a \in A$. Corresponding matrix rows are rolled up (summed) into a new matrix N_{ay} with a smaller first dimension. This simplification does not go for free as the mutual information contained in matrix N_{ay} can be smaller, since we deliberately disregarded small differences among elements of each group G_a . We want to achieve grouping that results in minimal reduction of (mutual) information.

The remainder of the paper follows this outline. In section 2 below we present a formal problem definition. The classic K-Means algorithm is well fitted for square of L_2 minimization of within-the-groups errors due to a special algebraic property of L_2 norm [12]. In section 3 we remind the reader of the basic algebra concerning classic K-Means. Parallel algebraic construction is applicable to a minimal information reduction clustering and is presented in section 4. Introduced algebraic property allows designing a feasible iterative algorithm.

So far we have talked about one dimension, keeping the other one fixed. In section 5 we describe an algorithm for *bi-dimensional clustering* (or *co-clustering*), which groups simultaneously elements of \mathbf{X} and \mathbf{Y} . This algorithm actually results in learning relationship patterns contained in matrix N_{xy} . It reduces the matrix size keeping discovered patterns as inviolate as possible.

The introduced model is well fitted for different data mining applications. The one that brought our attention to the problem is about learning a relationship between referrers and important pages in analysis of Web site traffic. This and other applications are explored in section 6.

2. IR-Clustering

We start with basic notations and definitions. Given a non-negative rectangular matrix $\mathbf{N}=(N_{xy})$, *marginal* matrix entries are defined as

$$N_{x.} = \sum_y N_{xy}, N_{.y} = \sum_x N_{xy}, N_{..} = \sum_{xy} N_{xy}.$$

We do not make any sparsity assumptions but we do require that $N_{xy} \geq 0$. Matrix \mathbf{N} defines the joint and two marginal probability distributions:

$$P_{xy} = N_{xy} / N_{..}, P_{x.} = N_{x.} / N_{..}, P_{.y} = N_{.y} / N_{..}.$$

Consider two random variables \mathbf{X}, \mathbf{Y} with joint distribution introduced above. Sometimes, when there is no confusion possible, we will use an informal simplification

$$N_x = N_{x.}, N_y = N_{.y}, P_x = P_{x.}, P_y = P_{.y},$$

Conditional probabilities are defined as

$$P_{y|x} = P_{xy}/P_x = N_{xy}/N_x, P_{x|y} = P_{xy}/P_y = N_{xy}/N_y.$$

Entropy of \mathbf{X} is defined as

$$H(\mathbf{X}) = - \sum_x P_x \log(P_x).$$

The analogous formulae are applicable to \mathbf{Y} . We use continuous interpolation $\mathbf{0} * \log(\mathbf{0}) = \mathbf{0}$, the logarithm base does not matter for future considerations (base two logarithms correspond to *bit* units, while natural logarithms result in *nat* units). The conditional entropy of \mathbf{X} given \mathbf{Y} is defined as the expectation of $H(\mathbf{X}|\mathbf{Y}=\mathbf{y})$ over all \mathbf{Y} :

$$H(\mathbf{X}|\mathbf{Y}) = \sum_y P_y H(\mathbf{X}|\mathbf{Y}=\mathbf{y}) = - \sum_{xy} P_y P_{x|y} \log(P_{x|y}) = - \sum_{xy} P_{xy} \log(P_{x|y})$$

Mutual Information between \mathbf{X} and \mathbf{Y} is defined as

$$I(\mathbf{X}, \mathbf{Y}) = \sum_{xy} P_{xy} \log(P_{xy}/P_x P_y). \quad (2.1)$$

It is a symmetric function, $I(\mathbf{X}, \mathbf{Y}) = H(\mathbf{X}) - H(\mathbf{X}|\mathbf{Y})$. We use the concept of mutual information to define an objective function further utilized in iterative optimization. The use of information theory concepts for different data mining tasks has a long history [30, 33, 3, 7] partially motivated by probability related ideas (as for example, Kullback Leibler distance [8]) and partially by a convenience of the concepts in defining proximity measures for nominal categorical attributes [26]. For further relationships between entropy, conditional entropy, and mutual information see [8].

Since the concept of probability is only a motivation for this work, we can safely ignore the difference between population and sample probabilities. Instead, the introduced concepts can be expressed in terms of the matrix \mathbf{N} . In particular, for mutual information (2.1) we have:

$$I(\mathbf{N}) = I(\mathbf{X}, \mathbf{Y}) = (1/N_{..}) \sum_{xy} N_{xy} \log(N_{xy} N_{..}/N_x N_y). \quad (2.2)$$

Consider a map $\mathbf{a}: \mathbf{X} \rightarrow \mathbf{A}$. It canonically induces a factorization of $|\mathbf{X}|$ elements into $|\mathbf{A}|$ groups $\mathbf{G}_a = \mathbf{G}[\mathbf{a}]$, $\mathbf{a} \in \mathbf{A}$, where $\mathbf{x} \in \mathbf{G}_a(\mathbf{x})$. We will use similar notation for grouping \mathbf{Y} into $|\mathbf{B}|$ groups \mathbf{G}_b via map $\mathbf{b}: \mathbf{Y} \rightarrow \mathbf{B}$. Maps \mathbf{a} and \mathbf{b} canonically generate a *rolled-up* matrix

$$\bar{\mathbf{N}}_{ab} = \sum_{\mathbf{x} \in \mathbf{G}[\mathbf{a}], \mathbf{y} \in \mathbf{G}[\mathbf{b}]} N_{xy}.$$

In context of data compression, the new rolled-up matrix is a condensed version of the original matrix. This new simplified version is achieved through the grouping of elements in each of the matrix dimensions. In the context of machine learning this process should throw away insignificant data and should preserve specific information. The concept of mutual information allows exactly this. We have already determined that the specific patterns we are interested in are cross-element distributions associated with each row and column. An adverse effect of our clustering is that by combining certain rows and certain columns together we introduce some fuzziness into rolled-up distributions. This by itself suggests that the rows (columns) that have similar distributions are good candidates for grouping. If the original matrix has a mutual information $I(\mathbf{X}, \mathbf{Y})$, a new compressed matrix resulting from clustering has mutual information $I(\mathbf{A}, \mathbf{B})$. Therefore, the damage done can be measured as a mutual information reduction:

$$\mathbf{R} = I(\mathbf{X}, \mathbf{Y}) - I(\mathbf{A}, \mathbf{B}) = I(\mathbf{N}) - I(\bar{\mathbf{N}}). \quad (2.3)$$

Definition. *IR-clustering* is a grouping of rows \mathbf{x} and columns \mathbf{y} of a matrix \mathbf{N} into groups \mathbf{G}_a and \mathbf{G}_b correspondingly, that minimizes Information Reduction \mathbf{R} .

Since both groupings are equivalent to a construction of two maps $\mathbf{a}: \mathbf{X} \rightarrow \mathbf{A}$, $\mathbf{b}: \mathbf{Y} \rightarrow \mathbf{B}$, IR-clustering is a problem of combinatorial optimization. When two dimensions are involved, we

will resort to a heuristic (because the problem is known to be NP-complete) similar to the gradient method in numerical optimization. As described in section 5, it uses as its basic building block an algorithm that clusters one dimension while keeping the other one intact.

3. Algebraic Properties of K-Means Clustering

In this section we describe a special algebraic property of K-Means clustering [12] that closely relates to the K-Means objective function defined in terms of squares of the L_2 norm. This property allows an important cancellation in that it enables a fast implementation of the K-Means optimization process. Our goal is to use this as an analogy and to derive a similar property for IR-clustering.

As a reminder, in classic K-Means algorithm we try to solve the combinatorial optimization problem of minimizing an objective function \mathbf{J} that is equal to the sum of the squares of errors between data points and their corresponding cluster centroids. In the following, the data points $\mathbf{x}=(x_y)$ are vectors in Euclidian space and y index enumerates vector components. The goal is to assign each \mathbf{x} to a group or *cluster* $\mathbf{G}[\mathbf{a}(\mathbf{x})]$, such that the number \mathbf{k} of clusters $\mathbf{G}_a=\mathbf{G}[\mathbf{a}]$ is fixed, $\mathbf{a}=\mathbf{1:k}$. Each cluster is represented by its centroid (a center of mass or *mean*) \mathbf{c}_a equal to a (weighted) arithmetic average of $\mathbf{x}\in\mathbf{G}[\mathbf{a}]$. This assignment is somehow initialized and then iteratively improved. To provide a comprehensive treatment, we derive a weighted version in which each point has a weight w_x . The objective function of the K-Means is equal to:

$$\mathbf{J} = \sum_a \mathbf{J}_a, \quad \mathbf{J}_a = \sum_{\mathbf{x}\in\mathbf{G}[\mathbf{a}]} w_x \|\mathbf{x} - \mathbf{c}_a\|^2. \quad (3.1)$$

Here the L_2 norm squared $\|\mathbf{z}\|^2 = \sum_y |z_y|^2$ is used and each term $\mathbf{J}_a = \mathbf{J}(\mathbf{G}_a)$ is a variance within the cluster \mathbf{G}_a . Different iterative refinement processes for K-Means are known. Classic K-Means optimization process consists of major iterations, each going through every data point. Points are examined one by one and are potentially moved from their current cluster to a new one so as to minimize \mathbf{J} . Deleting a point \mathbf{x} from its current cluster \mathbf{G}_a always decreases \mathbf{J}_a by amount Δ^-_a . Adding a point \mathbf{x} to a new cluster \mathbf{G}_b increases \mathbf{J}_b by another amount Δ^+_b . If the smallest possible increase Δ^+_b (\mathbf{b} can be any index but \mathbf{a}) is less than Δ^-_a , then the move is justified.

Major Iteration: (3.2)

```

for each  $\mathbf{x}$ 
  let  $\mathbf{G}_a$  be a current cluster of  $\mathbf{x}$ 
  find an impact  $\Delta^-_a$  of deleting  $\mathbf{x}$  from  $\mathbf{G}_a$ 
  among all  $\mathbf{b}=\mathbf{1:k}$ ,  $\mathbf{b}\neq\mathbf{a}$ , find the smallest  $\Delta^+_b$ 
  if  $\Delta^+_b < \Delta^-_a$  then
    move  $\mathbf{x}$  from  $\mathbf{G}_a$  to  $\mathbf{G}_b$ 
    update  $\mathbf{J} = \mathbf{J} - \Delta^-_a + \Delta^+_b$ 
    update centroids  $\mathbf{c}_a$  and  $\mathbf{c}_b$ 
  end if
end for

```

The issue with this algorithm is that the computation of the impact of the deletion or addition of a point to a cluster requires an inner loop with respect to all other cluster points. Because the cluster centroid is affected, all the distances to it must be recomputed. Any L_p norm can be used in (3.1). We claim that in the case of L_2 norm all these computations can be performed solely in terms of the point in question and a cluster centroid, making major K-Means iterations computationally feasible. This fact will be generalized to IR-clustering below.

To prove this claim, consider the computation of Δ^+_b . Let $\mathbf{c}=\mathbf{c}_b$ be the original centroid of cluster \mathbf{G}_b , having the weight $\mathbf{v}=\mathbf{v}_b$. Adding the new point \mathbf{x} of weight $\mathbf{w}=\mathbf{w}_x$ to \mathbf{G}_b results in the shift of \mathbf{c}

to a new position \mathbf{c}^+ . A new cluster $\mathbf{G}_b + \{\mathbf{x}\}$ has the total weight $\mathbf{v} + \mathbf{w}$, and, therefore, its centroid satisfies the following equations ($\mathbf{z} \in \mathbf{G}_b$)

$$\begin{aligned} (\mathbf{v} + \mathbf{w})\mathbf{c}^+ &= \sum_z \mathbf{w}_z \mathbf{z} + \mathbf{w}\mathbf{x} = \mathbf{v}\mathbf{c} + \mathbf{w}\mathbf{x}, \\ \mathbf{c}^+ &= (\mathbf{v}/(\mathbf{v} + \mathbf{w})) \mathbf{c} + (\mathbf{w}/(\mathbf{v} + \mathbf{w})) \mathbf{x} = \mathbf{c} + \Delta\mathbf{c}, \quad \Delta\mathbf{c} = (\mathbf{w}/(\mathbf{v} + \mathbf{w})) (\mathbf{x} - \mathbf{c}). \end{aligned} \quad (3.3)$$

The sum (3.1) of squares of errors within a group $\mathbf{G}_b + \{\mathbf{x}\}$ is equal to

$$\begin{aligned} \mathbf{J}(\mathbf{G}_b + \{\mathbf{x}\}) &= \sum_z \mathbf{w}_z \|\mathbf{z} - \mathbf{c}^+\|^2 + \mathbf{w} \|\mathbf{x} - \mathbf{c}^+\|^2 = \sum_z \mathbf{w}_z \|\mathbf{z} - \mathbf{c} - \Delta\mathbf{c}\|^2 + \mathbf{w} \|\mathbf{x} - \mathbf{c} - \Delta\mathbf{c}\|^2 \\ &= \sum_z \mathbf{w}_z \|\mathbf{z} - \mathbf{c}\|^2 + 2(\sum_z \mathbf{w}_z (\mathbf{z} - \mathbf{c}), \Delta\mathbf{c}) + \sum_z \mathbf{w}_z \|\Delta\mathbf{c}\|^2 + \mathbf{w} \|\mathbf{x} - \mathbf{c} - \Delta\mathbf{c}\|^2. \end{aligned}$$

By the definition of \mathbf{c} we have $\sum_z \mathbf{w}_z (\mathbf{z} - \mathbf{c}) = \mathbf{0}$, $\sum_z \mathbf{w}_z = \mathbf{v}$. From (3.3) we get

$$\mathbf{x} - \mathbf{c} - \Delta\mathbf{c} = \mathbf{x} - \mathbf{c} - (\mathbf{w}/(\mathbf{v} + \mathbf{w})) (\mathbf{x} - \mathbf{c}) = (\mathbf{v}/(\mathbf{v} + \mathbf{w})) (\mathbf{x} - \mathbf{c}),$$

Hence

$$\begin{aligned} \mathbf{J}(\mathbf{G}_b + \{\mathbf{x}\}) &= \mathbf{J}(\mathbf{G}_b) + \mathbf{v} \|\Delta\mathbf{c}\|^2 + \mathbf{w} \|(\mathbf{v}/(\mathbf{v} + \mathbf{w})) (\mathbf{x} - \mathbf{c})\|^2 \\ &= \mathbf{J}(\mathbf{G}_b) + \mathbf{v} \|(\mathbf{w}/(\mathbf{v} + \mathbf{w})) (\mathbf{x} - \mathbf{c})\|^2 + \mathbf{w} \|(\mathbf{v}/(\mathbf{v} + \mathbf{w})) (\mathbf{x} - \mathbf{c})\|^2 \\ &= \mathbf{J}(\mathbf{G}_b) + (\mathbf{v}\mathbf{w}/(\mathbf{v} + \mathbf{w})) \|\mathbf{x} - \mathbf{c}\|^2. \end{aligned}$$

We got the desired result. The impact of adding a point \mathbf{x} to group \mathbf{G}_b can be expressed without actual computations involving many members \mathbf{z} of \mathbf{G}_b , but only through \mathbf{x} and $\mathbf{c} = \mathbf{c}_b$:

$$\Delta^+_{\mathbf{b}} = \mathbf{J}(\mathbf{G}_b + \{\mathbf{x}\}) - \mathbf{J}(\mathbf{G}_b) = (\mathbf{v}\mathbf{w}/(\mathbf{v} + \mathbf{w})) \|\mathbf{x} - \mathbf{c}_b\|^2 = (\mathbf{v}\mathbf{w}/(\mathbf{v} + \mathbf{w})) \sum_y |\mathbf{x}_y - \mathbf{c}_{y_b}|^2 \quad (3.4)$$

Similarly to (3.4), it can be shown that impact of moving a point $\mathbf{x} \in \mathbf{G}_a$ from its current cluster with centroid \mathbf{c}_a is equal to

$$\Delta^-_{\mathbf{a}} = \mathbf{J}(\mathbf{G}_a) - \mathbf{J}(\mathbf{G}_a \setminus \{\mathbf{x}\}) = (\mathbf{v}\mathbf{w}/(\mathbf{v} - \mathbf{w})) \|\mathbf{x} - \mathbf{c}_a\|^2. \quad (3.5)$$

Note, that formally (3.5) can be conceived as “adding” a point \mathbf{x} with negative weight $-\mathbf{w}$. The move of \mathbf{x} from \mathbf{G}_a to \mathbf{G}_b in (3.2) is justified if the smallest of impacts (3.4) (among $\mathbf{b} \neq \mathbf{a}$) is less than the impact (3.5).

In both cases (3.4 and 3.5) we started with a definition that involved summation with respect to two indices, \mathbf{z} , \mathbf{y} , and algebraically transformed it into an expression having one \mathbf{y} -summation. All computations can be performed in terms of a centroid and the point \mathbf{x} in question. This is a remarkable simplification that makes classic K-Means algorithm viable.

In contemporary implementations of K-Means, the cluster centroids are not recomputed until the end of a major iteration. In this case, the major iteration is very simple

```

for each  $\mathbf{x}$ 
    find  $\mathbf{a} = \mathbf{argmin} \|\mathbf{x} - \mathbf{c}_a\|$ 
    assign  $\mathbf{x}$  to  $\mathbf{G}_a$ 
end for
compute centroids  $\mathbf{c}_a$ ,  $\mathbf{a} = 1:k$ , for newly assembled  $\mathbf{G}_a$ 

```

It can be shown that the objective function \mathbf{J} decreases after such major iteration. This computational flow has several crucial advantages:

- a) Algorithm is very transparent.
- b) It is not restricted to \mathbf{L}_2 norm.
- c) Organized computation can be parallelized.
- d) Only centroids \mathbf{c}_a are necessary to ignite this implementation, while (3.2) requires initialization of clusters \mathbf{G}_a themselves.

Will the result of this simplified iterative process be the same as for a classic one or does the classic scheme (3.2) have its own advantages? In fact, a counter-example can be provided that shows that the original process (3.2) sometimes achieves smaller values of the objective function. Effectively, at the end of the simplified major iteration, \mathbf{x} are reassigned based on centroids already blurred by their former assignments. This also makes the convergence of the simpler major iterations slower. From the point of view of total operation count, computing (3.4, 3.5) requires almost the same amount ($3|\mathbf{Y}| + 4$) as computing the distance ($3|\mathbf{Y}|$) in the simpler modification! The updating of the centroids \mathbf{c}_a and \mathbf{c}_b in the classic version is compensated by full re-computation of the centroids after reassigning of all \mathbf{x} in the simpler modification. Actually, when major iteration starts to converge, fewer points cause re-computation so that substantial saving is achieved by the classic version. Therefore, the classic schema (3.2) is preferable when initial assignments are known, the implementation is not parallel, and L_2 norm is used.

4. Algebraic Properties of IR-Clustering

Our goal now is to establish parallel results for IR-clustering. In this section we will assume that clustering is performed with respect to elements \mathbf{x} only, so that $\mathbf{B}=\mathbf{Y}$ is left intact. Recall that the objective function we want to minimize is a reduction in mutual information (2.3). We will use probabilistic notations; though as in (2.3) all the formulas below can be trivially expressed in terms of $\mathbf{N}, \bar{\mathbf{N}}$ alone. We do this since, while points in K-Means are \mathbf{n} -dimensional vectors, *points* \mathbf{x} in this section are \mathbf{n} -dimensional distributions associated with elements \mathbf{x} (conditioned by \mathbf{x}).

Our first step is to factorize the total information reduction \mathbf{R} into a sum similarly to the factorization (3.1) of \mathbf{J} into the sum of \mathbf{J}_a . We use the rollup property $\mathbf{P}_{ay} = \sum_{\mathbf{x} \in G[a]} \mathbf{P}_{xy}$:

$$\begin{aligned} \mathbf{R} &= \mathbf{I}(\mathbf{X}, \mathbf{Y}) - \mathbf{I}(\mathbf{A}, \mathbf{Y}) = \sum_{xy} \mathbf{P}_{xy} \log(\mathbf{P}_{xy} / \mathbf{P}_x \mathbf{P}_y) - \sum_{ay} \mathbf{P}_{ay} \log(\mathbf{P}_{ay} / \mathbf{P}_a \mathbf{P}_y) \\ &= \sum_a \sum_{\mathbf{x} \in G[a]} \sum_y \mathbf{P}_{xy} [\log(\mathbf{P}_{xy} / \mathbf{P}_x \mathbf{P}_y) - \log(\mathbf{P}_{ay} / \mathbf{P}_a \mathbf{P}_y)] = \sum_a \mathbf{R}_a, \\ \mathbf{R}_a &= \sum_{\mathbf{x} \in G[a]} \sum_y \mathbf{P}_{xy} \log(\mathbf{P}_{xy} \mathbf{P}_a / \mathbf{P}_{ay} \mathbf{P}_x). \end{aligned} \tag{4.1}$$

It is not immediately obvious that each term (4.1) of this factorization is non-negative. To prove that this is actually true, we obtain a different representation of \mathbf{R} based on a concept of a *Kullback Leibler* (KL) distance [8]. KL-distance is used to compare two probability distributions \mathbf{P}_a and \mathbf{Q}_a over the same space of outcomes \mathbf{y} :

$$\mathbf{D}(\mathbf{P}_y \parallel \mathbf{Q}_y) = \sum_y \mathbf{P}_y \log(\mathbf{P}_y / \mathbf{Q}_y). \tag{4.2}$$

Because it is not symmetric, the KL-distance is not a metric in the general sense. However, it is known to be non-negative. Computing the expectation of KL distance between \mathbf{y} -probability distributions associated with (conditioned by) \mathbf{X} and \mathbf{A} respectively and comparing it with (4.1) yields:

$$\sum_x \mathbf{P}_x \mathbf{D}(\mathbf{P}_{y|x} \parallel \mathbf{P}_{y|a}) = \sum_a \sum_{\mathbf{x} \in G[a]} \mathbf{P}_x \sum_y \mathbf{P}_{y|x} \log(\mathbf{P}_{y|x} / \mathbf{P}_{y|a}) = \sum_a \mathbf{R}_a = \mathbf{R}. \tag{4.3}$$

This result not only proves that each term \mathbf{R}_a is non-negative, but it also gives a new interpretation to them; each one is equal to expectation over the cluster of KL-distance between \mathbf{y} -probability distributions $\mathbf{P}_{y|x}$ corresponding to the point \mathbf{x} and another \mathbf{y} -probability distribution $\mathbf{P}_{y|a}$ corresponding to the cluster centroid. While in K-Means the cluster centroid is the center of mass (mean) with total weight located in it, here the centroid and its total weight are represented by the distribution $\mathbf{P}_{y|a}$, and the probability \mathbf{P}_a , computed by the rollup process.

The component \mathbf{R}_a of the total reduction \mathbf{R} increases when the group \mathbf{G}_a contains points \mathbf{x} with distributions $\mathbf{P}_{y|x}$ significantly different from a centroid's distribution $\mathbf{P}_{y|a}$. On the other hand, if all

$\mathbf{P}_{y|x}$ in a group are close to $\mathbf{P}_{y|a}$, then the ratios are close to one, the logarithms are close to zero, and so the term \mathbf{R}_a is small. Thus each (4.3) term $\mathbf{R}_a = \mathbf{R}(\mathbf{G}_a)$ measures the compactness of a cluster, and is similar to the concept of within a group error \mathbf{J}_a in K-Means clustering.

The introduced formulas give rise to the following algorithm similar to K-Means. Perform major iterations of the form (3.2) with \mathbf{R}_a instead of \mathbf{J}_a . As before, Δ_a^- is the impact on term \mathbf{R}_a of the deletion of point \mathbf{x} from the group \mathbf{G}_a , and Δ_b^+ is the impact on term \mathbf{R}_b of addition of point \mathbf{x} to the group \mathbf{G}_b . Here we assume that \mathbf{x} currently belongs to \mathbf{G}_a , that $\mathbf{b} \neq \mathbf{a}$, and that at the start groups are somehow initialized. The new shifted centroid \mathbf{c}^+ (\mathbf{c}^-) in K-Means now corresponds to the shifted probabilities $\mathbf{P}_a^{+/-} = \mathbf{P}_a +/- \mathbf{P}_x$, $\mathbf{P}_{ay}^{+/-} = \mathbf{P}_{ay} +/- \mathbf{P}_{xy}$, where \mathbf{x} is the point that is moved. Starting with the Δ_b^+ term, our goal is to establish that the impact terms can be computed *without* actual involvement of all the \mathbf{P}_{zy} , for $\mathbf{z} \in \mathbf{G}_b$, $\mathbf{z} \neq \mathbf{x}$:

$$\begin{aligned}
\Delta_b^+ &= \mathbf{R}(\mathbf{G}_b + \{\mathbf{x}\}) - \mathbf{R}(\mathbf{G}_b) = \\
&= \sum_z \sum_y \mathbf{P}_{zy} \log(\mathbf{P}_{zy} \mathbf{P}_b^+ / \mathbf{P}_{by}^+ \mathbf{P}_z) + \sum_y \mathbf{P}_{xy} \log(\mathbf{P}_{xy} \mathbf{P}_b^+ / \mathbf{P}_{by}^+ \mathbf{P}_x) \\
&\quad - \sum_z \sum_y \mathbf{P}_{zy} \log(\mathbf{P}_{zy} \mathbf{P}_b / \mathbf{P}_{by} \mathbf{P}_z) \\
&= \sum_z \sum_y \mathbf{P}_{zy} \log(\mathbf{P}_{by} \mathbf{P}_b^+ / \mathbf{P}_{by}^+ \mathbf{P}_b) + \sum_y \mathbf{P}_{xy} \log(\mathbf{P}_{xy} \mathbf{P}_b^+ / \mathbf{P}_{by}^+ \mathbf{P}_x) \\
&= \sum_y \mathbf{P}_{by} \log(\mathbf{P}_{by} \mathbf{P}_b^+ / \mathbf{P}_{by}^+ \mathbf{P}_b) + \sum_y \mathbf{P}_{xy} \log(\mathbf{P}_{xy} \mathbf{P}_b^+ / \mathbf{P}_{by}^+ \mathbf{P}_x). \tag{4.4}
\end{aligned}$$

Similarly

$$\begin{aligned}
\Delta_a^- &= \mathbf{R}(\mathbf{G}_a) - \mathbf{R}(\mathbf{G}_a \setminus \{\mathbf{x}\}) = \\
&= \sum_y \mathbf{P}_{ay} \log(\mathbf{P}_a \mathbf{P}_{ay}^- / \mathbf{P}_a^- \mathbf{P}_{ay}) + \sum_y \mathbf{P}_{xy} \log(\mathbf{P}_{xy} \mathbf{P}_a / \mathbf{P}_{ay} \mathbf{P}_x) \tag{4.5}
\end{aligned}$$

If $\Delta_b^+ < \Delta_a^-$, then a move of \mathbf{x} from \mathbf{G}_a to \mathbf{G}_b is justified and updated $\mathbf{R} = \mathbf{R} - \Delta_a^- + \Delta_b^+$.

The net result achieved is that we started with a definition that involved summation with respect to two indices, \mathbf{z} , \mathbf{y} , and algebraically transformed it into an expression having one index \mathbf{y} summation. From (4.4, 4.5) we see that all computations can be fully performed in terms of the group ‘‘centroid’’ (\mathbf{a} -distribution) and the point \mathbf{x} in question. This is a result fully parallel to the one we demonstrated for K-Means. In both algorithms a linear amount of operations (proportional to $|\mathbf{Y}|$ and \mathbf{k}) is needed to decide whether \mathbf{x} has to be reassigned and, if so, to which cluster.

So far we have been concerned with the common theoretical treatment of two algorithms. Many practical issues are similar between them as well. Different implementations of K-Means clustering algorithms face two questions:

- (1) How to initialize clusters to start the algorithm?
- (2) What is the preferable number \mathbf{k} of clusters to construct?

Usually, K-Means randomly initializes \mathbf{k} -groups and we use the same choice in IR-clustering as well. Although there is a known negative effect that this could result in quite non-optimal local minimum of objective function, there are strategies [4] to compensate for this effect. Regarding the second question, different indicators (such as F-statistic, Marriott index, coefficients of separation, MDL, AWD and BIC criteria [29, 17, 31]) are used to derive the most appropriate \mathbf{k} . In the case of the IR-clustering, a more straightforward criterion can be suggested. Consider the user defined threshold ε , say 0.05, which specifies a percentage (e.g., 5%) of the original information the user is willing to sacrifice for the simplification that results from the clustering. If $\mathbf{R}^{(\mathbf{k})}$ is the information reduction achieved in \mathbf{k} -group clustering, then we stop with the smallest \mathbf{k} such that

$$\mathbf{R}^{(\mathbf{k})} / \mathbf{I}(\mathbf{X}, \mathbf{Y}) \leq \varepsilon. \tag{4.6}$$

5. IR Bi-Dimensional Clustering

So far we have analyzed IR-clustering of \mathbf{X} elements, leaving \mathbf{Y} intact. However, the objective function $\mathbf{R} = \mathbf{I}(\mathbf{X}, \mathbf{Y}) - \mathbf{I}(\mathbf{A}, \mathbf{B}) = \mathbf{I}(\mathbf{N}) - \mathbf{I}(\bar{\mathbf{N}})$ in (2.3) is *symmetric* and is well suited for structural simplification of both \mathbf{X} and \mathbf{Y} . Rather than grouping \mathbf{X} elements having similar \mathbf{y} -distributions or vice versa, simultaneous clustering *learns the relationship* between the two dimensions. Different approaches can be suggested to this bi-dimensional clustering that differ in the organization of the major iterations. This organization must combine the groupings of both \mathbf{x} and \mathbf{y} . We present one such combination below that exploits the duality between \mathbf{X} and \mathbf{Y} .

The suggested algorithm performs a sequence of clustering steps that consecutively reduce the original dimensions $|\mathbf{X}|$, $|\mathbf{Y}|$. We do not try to reduce dimensions drastically, but rather proceed incrementally. This allows switching back and forth between the dimensions, and, as a result, concentrating on the most promising dimension of the moment. If \mathbf{Y} -dimension is reduced, it benefits the following \mathbf{X} -clustering step because the amount of computation per \mathbf{x} -element is proportional to $|\mathbf{Y}|$. Symmetrically, this in turn benefits the consecutive \mathbf{Y} -reduction step. The algorithm uses (4.6) as the stopping criterion.

We start from $\mathbf{k}_x = |\mathbf{X}|$, $\mathbf{k}_y = |\mathbf{Y}|$. Given a $\mathbf{k}_x \times \mathbf{k}_y$ matrix \mathbf{N}_{xy} , we can cluster a set \mathbf{X} in \mathbf{m} clusters as described in section 4 by minimizing the information reduction \mathbf{R} . In doing so we assume that \mathbf{Y} is fixed. The result of this operation is a system of \mathbf{m} groups \mathbf{G}_a covering \mathbf{X} . We will use the function notation $\mathbf{G}_a = \mathbf{Cluster}(\mathbf{X}, \mathbf{N}_{xy}, \mathbf{m})$, where \mathbf{G}_a should be understood as $\mathbf{G}_{a=1:m}$ rather than one group, so that in this notation \mathbf{a} is a “free” index. Symmetrically, a set \mathbf{Y} can be clustered into a system of \mathbf{n} groups $\mathbf{G}_b = \mathbf{Cluster}(\mathbf{Y}, \mathbf{N}_{xy}, \mathbf{n})$, keeping \mathbf{X} fixed. These two operations are dual; one reduces rows, the other columns.

Grouping $\mathbf{X} \rightarrow \mathbf{A}$ canonically generates over the rollout $\mathbf{m} \times \mathbf{k}_y$ matrix \mathbf{N}_{ay} by adding together \mathbf{x} -rows of the initial matrix belonging to the same groups \mathbf{G}_a . The resulting rollout distributions are precisely the centroids of clusters \mathbf{G}_a introduced in section 4. We will use the notation $\mathbf{N}_{ay} = \mathbf{Roll}(\mathbf{X}, \mathbf{N}_{xy}, \mathbf{G}_a)$. Correspondingly, $\mathbf{N}_{xb} = \mathbf{Roll}(\mathbf{X}, \mathbf{N}_{xy}, \mathbf{G}_b)$ is a $\mathbf{k}_x \times \mathbf{n}$ matrix obtained by adding the \mathbf{y} -columns in the same groups \mathbf{G}_b . Both operations are related to maps of \mathbf{X} onto \mathbf{A} and \mathbf{Y} onto \mathbf{B} .

In addition to the ε in (4.6), we allow the user to specify minimum \mathbf{x} and \mathbf{y} -dimensions \mathbf{k}_{xmin} , \mathbf{k}_{ymin} . It gives the opportunity to limit potential reductions and, in the extreme case, to disable one of the dimensions. In the following algorithm *SimplifyRelation* we use notation \mathbf{k} for a particular dimension and \mathbf{s} for dimension decrements. This algorithm groups \mathbf{X} and \mathbf{Y} into unspecified numbers \mathbf{k}_x , \mathbf{k}_y of groups \mathbf{G}_a , \mathbf{G}_b respectively.

$[\mathbf{G}_a, \mathbf{G}_b] = \mathbf{SimplifyRelation}(\mathbf{N}_{xy}, \varepsilon, \mathbf{k}_{xmin}, \mathbf{k}_{ymin})$

// Step 0. Initialize:

$\mathbf{k}_x = |\mathbf{X}|$, $\mathbf{k}_y = |\mathbf{Y}|$,

if $\mathbf{k}_{xmin} < |\mathbf{X}|$ then initialize \mathbf{s}_x ; else $\mathbf{s}_x = 0$

if $\mathbf{k}_{ymin} < |\mathbf{Y}|$ then initialize \mathbf{s}_y ; else $\mathbf{s}_y = 0$

$\mathbf{f}_0 = (1 - \varepsilon) \mathbf{I}(\mathbf{N}_{xy})$

$\mathbf{N}_{xb} = \mathbf{N}_{xy}$, $\mathbf{N}_{ay} = \mathbf{N}_{xy}$

while $\mathbf{s}_x > 0$ and $\mathbf{s}_y > 0$

// Step 1. Cluster rows

if $\mathbf{s}_x > 0$ then

$\mathbf{G}_a = \mathbf{Cluster}(\mathbf{X}, \mathbf{N}_{xb}, \mathbf{k}_x - \mathbf{s}_x)$

$\mathbf{N}_{ab} = \mathbf{Roll}(\mathbf{X}, \mathbf{N}_{xb}, \mathbf{G}_a)$

$\mathbf{f}_x = \mathbf{I}(\mathbf{N}_{ab})$

if $\mathbf{f}_x < \mathbf{f}_0$ and $\mathbf{s}_x > 1$ then reduce \mathbf{s}_x and go to step 1

end if

```

// Step 2. Cluster columns
if  $s_y > 0$  then
     $\mathbf{G}_b = \text{Cluster}(\mathbf{Y}, \mathbf{N}_{ay}, \mathbf{k}_y - s_y)$ 
     $\mathbf{N}_{ab} = \text{Roll}(\mathbf{Y}, \mathbf{N}_{ay}, \mathbf{G}_b)$ 
     $\mathbf{f}_y = \mathbf{I}(\mathbf{N}_{ab})$ 
    if  $\mathbf{f}_y < \mathbf{f}_0$  and  $s_y > 1$  then reduce  $s_y$  and go to step 2
end if
// Step 3. Estimate the progress
if  $\mathbf{f}_x < \mathbf{f}_0$  and  $\mathbf{f}_y < \mathbf{f}_0$  then
    exit while loop
else if  $\mathbf{f}_x \geq \mathbf{f}_y$  then
     $\mathbf{k}_x = \mathbf{k}_x - s_x$ 
     $\mathbf{N}_{ay} = \text{Roll}(\mathbf{N}_{xy}, \mathbf{X}, \mathbf{G}_a)$ 
else
     $\mathbf{k}_y = \mathbf{k}_y - s_y$ 
     $\mathbf{N}_{xb} = \text{Roll}(\mathbf{N}_{xy}, \mathbf{X}, \mathbf{G}_b)$ 
end if
if  $\mathbf{k}_x > \mathbf{k}_{x\min}$  then update  $s_x > 0$ ; else  $s_x = 0$ 
if  $\mathbf{k}_y > \mathbf{k}_{y\min}$  then update  $s_y > 0$ ; else  $s_y = 0$ 
end while

```

A few comments are needed regarding the above algorithm:

- The major while-loop proceeds until the relative reduction of mutual information reaches specified threshold ε , or the current dimensions \mathbf{k}_x , \mathbf{k}_y both reach specified minimal values $\mathbf{k}_{x\min}$, $\mathbf{k}_{y\min}$. Under normal circumstances, the two dual attempts to decrease each of the dimensions are done and the one that reduces mutual information less survives.
- Each clustering step is based on the iterative optimization described in section 4. Consider, for example, step 1. Grouping of \mathbf{X} values is performed with respect to cross elements $\mathbf{b} \in \mathbf{B}$, where the set \mathbf{B} enumerates the most currently constructed \mathbf{Y} -clusters. In particular, the time per step monotonically decreases.
- We have not specified how initialization of clusters is done for each clustering step. It can be done randomly, but also the few closest groups of previous steps can be merged together to provide an initial guess. This approach utilizes previous coarser clustering in the same dimension that otherwise are just thrown away.
- We experimented with different strategies of the treatment of decrements s_x , s_y . While the simplest strategy is to keep them always equal to a small number, results show that such a policy is too conservative. A much more liberal heuristic is based on updating decrements depending on the rate of decrease of mutual information \mathbf{f}_x , \mathbf{f}_y .

Algorithm *SimplifyRelation* can itself be used several times to generate hierarchical agglomerative clustering. On each level $v=1,2,\dots$ we allow relative reduction ε_v , $\varepsilon_v < \varepsilon_{v+1}$, where $\prod (1-\varepsilon_v) = 1-\varepsilon$. A few levels (like three or four) usually work well in practice. As a reasonable choice of $\varepsilon_v = 1 - \mathbf{q}_v$, $\mathbf{q}_{v+1} = 2^v \mathbf{q}_1$ can be suggested.

6. Applications

Learning the structure of a relationship presented in a non-negative rectangular matrix by grouping its rows and columns has many applications. With some liberty, we can say that the generic practical problem with a simple two-dimensional OLAP cube is that the presented table is frequently large and thus is not well suited for business action. Some condensation of this

information into a smaller table can be considered as a necessary step preceding any business intelligence handling.

6.1. Retail Assortment Optimization

Retailers with hundreds of stores and thousands of products are faced with the dilemma of how much of each product to place in each store. Because using a distinct product assortment plan for each store causes a logistical nightmare, retailers try to simplify into a smaller number of assortment groups. In this context the different stores are enumerated by elements \mathbf{x} , the different products (or merchandise categories) are enumerated by elements \mathbf{y} , sales per period of time (for example, a month) in a store \mathbf{x} for a category \mathbf{y} is stored as the matrix entry N_{xy} . IR bi-dimensional clustering reduces the original universe of stores \mathbf{x} to a lesser number of store groups \mathbf{G}_a and simultaneously reduces the universe of merchandise categories to super-category groups \mathbf{G}_b . The resulting matrix N_{ab} represents generalized *assortment patterns*. In practice, a retailer can get a significantly simpler assortment plan (reducing each dimension ten-fold results in 100 times simpler assortment) that preserves 90% of the specificity contained in the original assortment matrix!

6.2. Affiliate Visit Analysis

Our second example applies introduced clustering to a Web analysis. The Web can be considered a loose collection of inter-connected sites. Many Web sites attempt to encourage visitors to come to their site by placing special content on third-party sites and paying these so-called affiliates by the number and/or quality of the referrals. This quality is determined by viewing specified important pages such as registration or checkout. Here the external referring affiliates are \mathbf{x} , the specified pages are \mathbf{y} , and the number N_{xy} of visits (or time spent, or any other non-negative characteristic) initiated from a referral \mathbf{x} to a page \mathbf{y} over fixed period of time represents the relationship between referrers and pages. In practice there is a good deal of asymmetry in browsing patterns and affiliates that might otherwise be treated the same (i.e., Yahoo and Excite) do not end up in the same groups \mathbf{G}_a . Learning and simplification of introduced relationships provide important insights into how to reorganize the Web site structure. It also allows to experiment with promotions on low-volume referrer affiliates, and to negotiate contracts based on objective information of visiting activity.

6.3. Online Advertising Effectiveness

In a similar vein, many Web sites use banner ads or outbound e-mails to redirect traffic to a special page that is not the home page. These special pages could be product directories, promotions, registration pages, etc. Here \mathbf{x} are the individual redirects, \mathbf{y} are the potential events (registration, purchase, spurious path, etc.) and matrix N_{xy} represents the frequency counts accumulated over a fixed period of time. The introduced clustering provides convenient means to group sources with similar event-patterns and to group events under consideration in dual groups that respond similarly to advertisement flow. This clustering can compare these ads and e-mails. For example, differences between the same ad published on different external sites or two different ads published on the same site. This information can be crucial for analysis of advertisement effectiveness and for cross selling.

6.4. Text Mining

Co-clustering of documents and words is an important problem in text mining [10]. Given a set of index terms (important or frequent words), document clustering can be performed based on so-called document vector representation that identifies a document with a vector of frequencies (or term frequency – inverse document frequency weights). This approach is standard in text mining. However, the terms themselves can be defined by clustering words based on their distribution among the documents. So this problem has the duality described above, with a word-by-document matrix \mathbf{N} called incidence matrix in this context. For a thorough introduction to co-clustering techniques that are based on graph partitioning, see [10, 11].

6.5. Example Implementation

We will demonstrate how IR-Clustering actually works using example 6.2 above. A set \mathbf{Y} of 203 important pages (columns) on an anonymous Web site and a set \mathbf{X} of 197 different referrers (rows) was subjected to analysis. For brevity, we omit the details of data collection, noise reduction, and pre-processing. Results are presented in Figures 1-3 in section 10.

To give a taste of how dimension reduction depends on ϵ we provide in the table below.

Table 1

ϵ	$ \mathbf{A} $	$ \mathbf{B} $	% of original table size
Original data	197	203	100 %
0.02 (2%)	49	42	5.15 %
0.04 (4%)	33	30	2.48 %
0.06 (6%)	26	22	1.43 %
0.08 (8%)	20	18	0.90 %

7. Conclusions

The introduced concept of IR-clustering demonstrates striking similarity with the K-Means optimization of the sum of squares of \mathbf{L}_2 errors. For both algorithms, we established algebraic formulas that allow efficient iterative optimization. More exactly, the effects of moving a point to or from one cluster does not require computations involving other cluster elements, and can be solely performed in terms of the point itself and a cluster centroid. The objective function of IR-clustering was defined using information theory analogy, which reflects the actual content of information and therefore makes it ideally suited for data mining. As a result, during the clustering process, points that we try to group are not represented by Euclidian vectors, but by probability distributions over the same set of transversal elements (outcomes).

The IR-clustering algorithm has an embedded duality; it can be applied to either of the relationship dimensions. We exploit this symmetry to gradually simplify a matrix using bi-dimensional clustering scheme. Attempts in both directions are performed, and the most promising ones are preserved. This automatically determines the appropriate numbers of clusters for each system, eliminating a troublesome problem in K-Means.

The simplicity of the model that led us to a concept of IR-clustering predisposes it to a wide variety of applications from retail to the Web. In fact, while we started this research with the practical goal of grouping together referrers and pages, we were pleasantly surprised to discover

than it efficiently solved the assortment optimization problem that we had previously attacked with K-Means.

In practical use, IR clustering has several important properties. It addresses the ubiquitous problem of exploring a relatively large and unmanageable table by reducing it to a table of actionable size. Furthermore, IR clustering uses an easily explainable objective function and the process is controlled by a single parameter, relative information reduction. As a result, the granularity is picked automatically. We actively use the mathematical concept of duality. Rather than clustering in a subspace, the approach continually creates derived attributes to cluster on. Finally, the resulting clusters are simple to present and allow rich visualization.

8. Related Work

Clustering is used in a variety of fields and applications, including statistics [25] and machine learning [15]. Several clustering techniques applicable to data with general set of attributes are presented in [23]. In this article we are mostly interested in partitioning clustering, however we also present an extension to a hierarchical clustering [23, 25]. Most partitioning clustering techniques are based on the concepts of closeness, distance, or similarity while other partitioning algorithms look for dense areas [2, 13]. The classic example of the first kind is K-Means algorithm [12, 22]. Different improvements to alleviate effects of initialization on results of K-Means are proposed [4, 34]. While we are not concerned here with scalability issues, we note that classic K-Means has numerous extensions to scalable unsupervised learning in databases CLARANS [14] and BIRCH [35].

IR algorithm is an iterative optimization of objective function (equal to a reduction in mutual information). It can be viewed in a context of general EM framework [9, 28, 30]. Two specific examples of particular algorithms are AutoClass [6] and MCLUST [18].

New approaches are emerging for objects of special nature. Many of them are influenced by Web applications [5, 16, 19], textual analysis [10, 11, 27], and spatial data mining [21]. In [5], for example, probabilistic clustering is employed to cluster visitor sessions. Many practical applications involve data with numerous attributes that make data cells very sparse and distance measures very blurred. Clustering in subspaces is a promising direction for many practical problems [1, 2, 7]. Bi-dimensional clustering or co-clustering that involves duality was used in context of the Web in [32]. An excellent review of graph partitioning methods for co-clustering in text mining is presented in [11].

We started with several application problems. Successful application of clustering requires effective visualization. References to corresponding techniques can be found in [24].

In this article we heavily relied on information theory concepts [8]. Their use in data mining is quite established [30]. In particular, applications in classification learning using decision trees [33], in exploratory data analysis [3], in determining subspaces for clustering [7], and in definition of similarity measures [26].

9. Acknowledgments

All the work described in this paper was performed while the authors were employed by Accrue Software, Inc. Many people helped us in this endeavor. Bob Wyman whose energy, expertise, and enthusiasm helped during the development of the Accrue Affiliate product, Phil Aaronson who first brought our attention to relationships between referrers and pages, Sally Chase, Hamid Kahangi, Michael Pertsov, who developed the software for Affiliate, Dee Jay Randall who tediously reviewed all the results, and Sue Krouscup who helped us with the text preparation.

10. Screen Shots

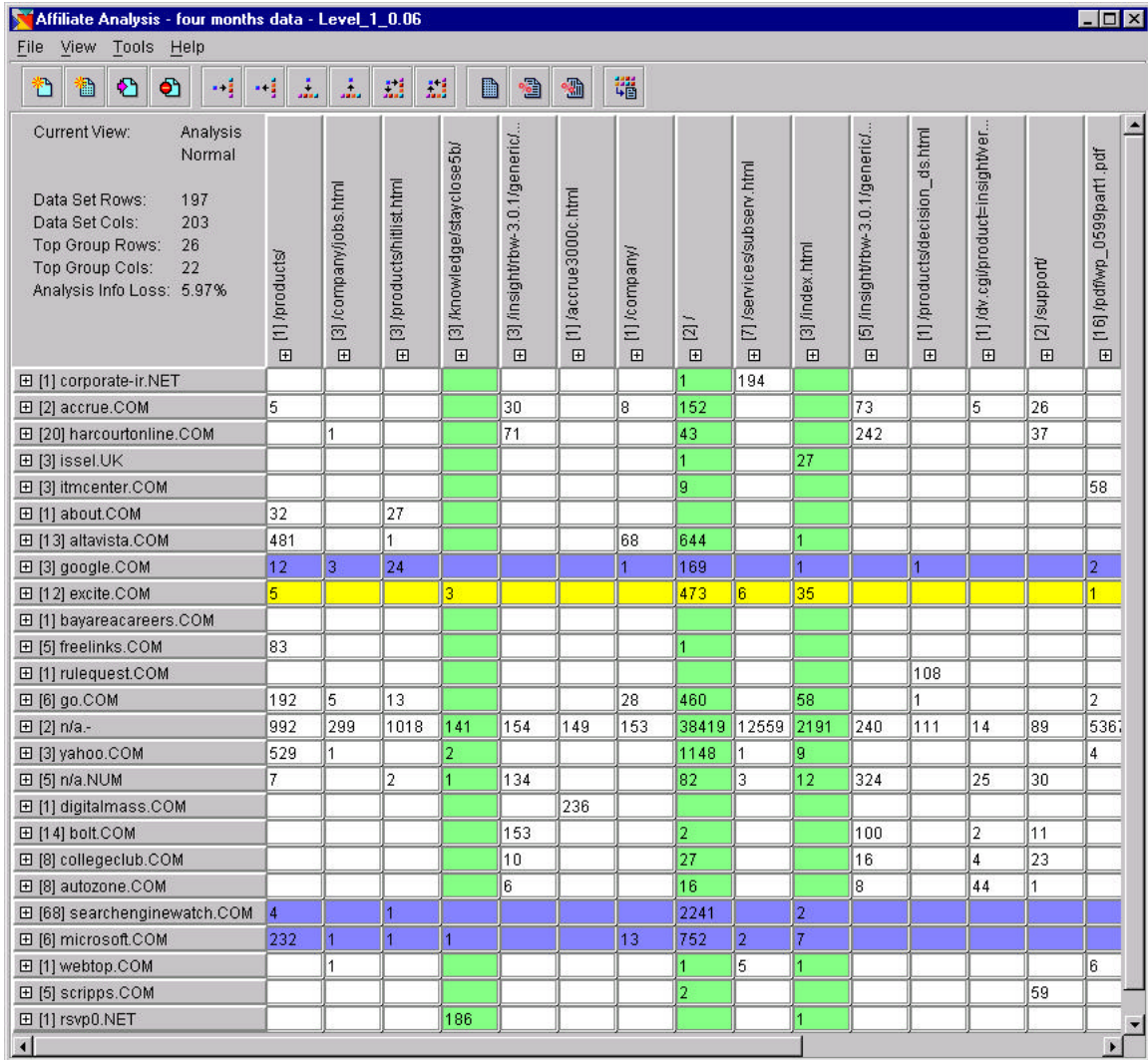


Figure 1: Original 197 rows **X** representing different referrers and 203 columns **Y** representing important pages are clustered into 26 row groups **A** and 22 column groups **B**. Information reduction $\epsilon=6\%$ was used in IR bi-dimensional clustering. Numbers in cells are row percentages (of total referrers volume). Selected light-color row corresponds to a group with 12 elements. It got its name excite.com from its most typical representative. Three darker color rows indicate groups whose behavior is similar to the selected group. Colored columns are cross-groups that reflect the selected group the most.

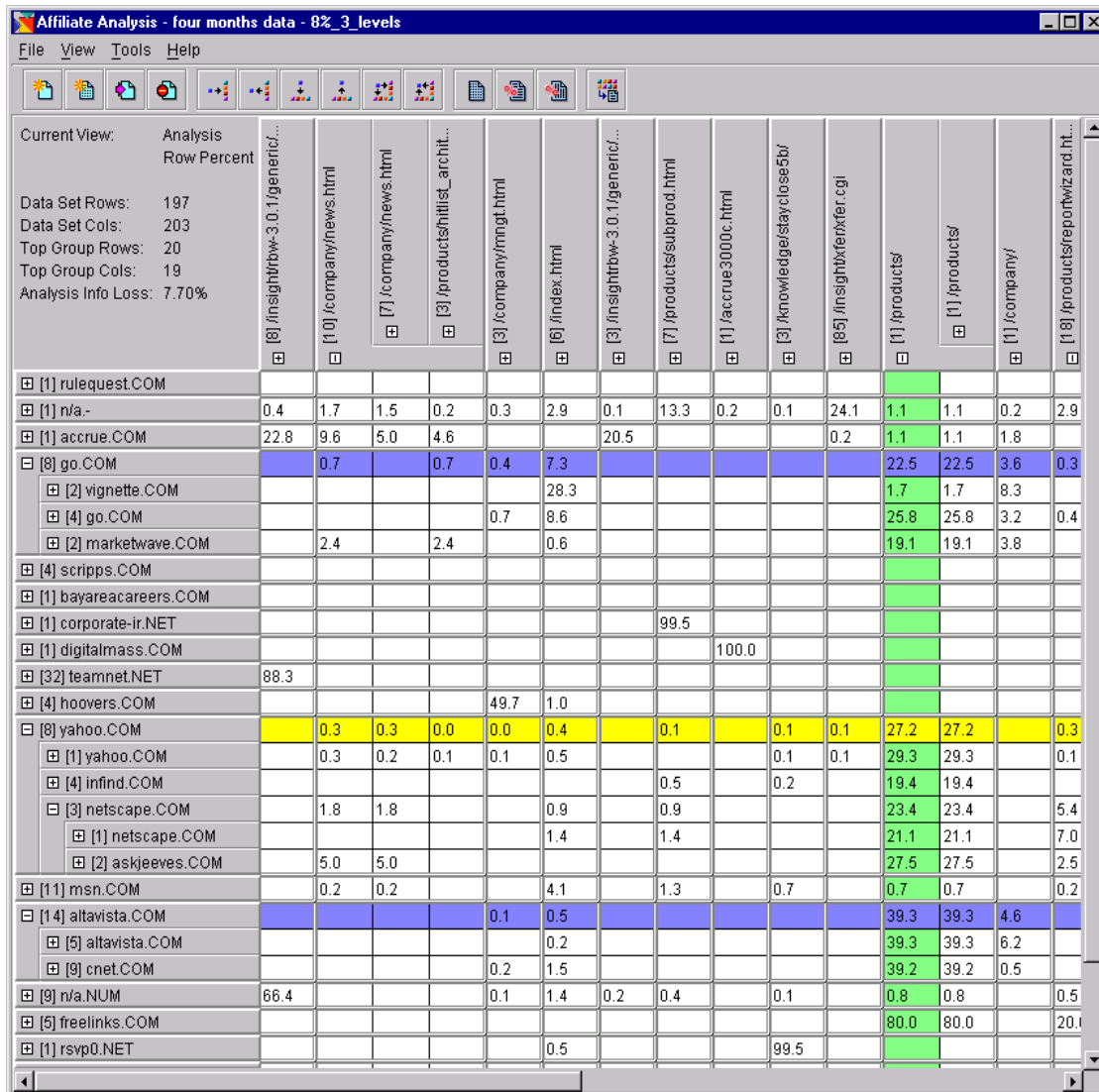


Figure 2: The same data as before is subjected to a hierarchical clustering. Three-level dendrogram is constructed. One top-level row group, yahoo.com, is selected. It contains total 8 rows organized in three sub-groups of level two. For example, the third one, netscape.com, consists in turn of two level-one groups. One of them, askjeeves.com, finally represents two original rows.

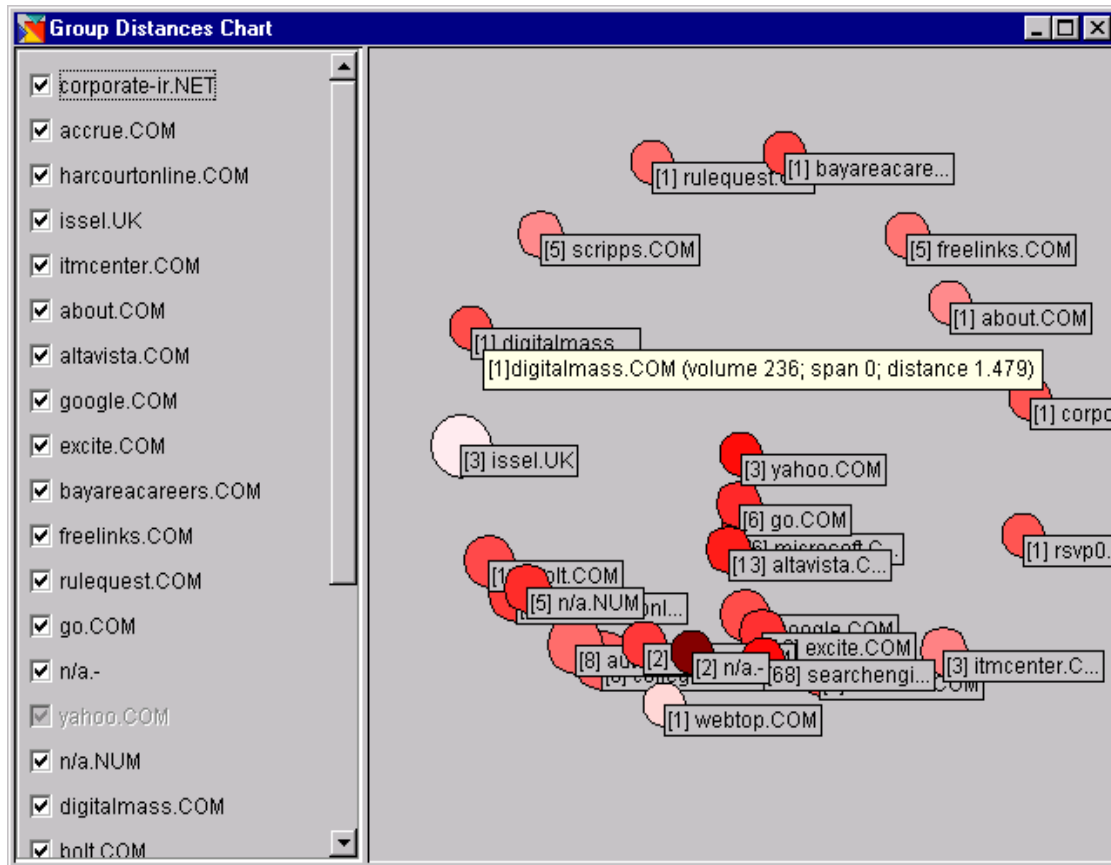


Figure 3: IR clustering allows rich visualization. Again consider the same data and clustering that are presented on Figure 1. Instead of the row group excite.com, the group yahoo.com (containing actually yahoo.com and two other referrers) is selected. In the above visualization, the selected group is located in the center. The radial distance from this central group is (logarithmically) proportional to the informational distance between the groups. Therefore, most similar groups are go.com (6 elements), microsoft.com (6 elements), and altavista.com (13 elements). The angle reflects metric relationships between non-central groups. The intensity of color corresponds to the group click volume, the number in square brackets is a number of affiliates in the group, and the size of a circle reflects a group's compactness.

References

- [1] Aggrawal, C., Yu, P.S., *Finding Generalized Projected Clusters in High Dimensional Spaces*, Sigmoid Record v.29, No. 2, 2000, Intl. Conf. On Management of Data, Dallas, TX, 2000.
- [2] Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P., *Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications*, Proceedings of the ACM SIGMOD Conference on Management of Data, Montreal, Canada, 1998.
- [3] Becher, J.D., Berkhin, P., Freeman, E., *Automating Exploratory Data Analysis for Efficient Data Mining*, KDD-2000, 424-429, ACM, New York, NY, 2000.
- [4] Bradley, P.S., Fayyad, U.M., *Refining Initial Points for K-Means Clustering*, *Machine Learning*, ICML'98, 91-99, Morgan Kaufmann, San Francisco, CA, 1998.
- [5] Cadez, I.V., Gaffney, S., Smyth, P., *A General Probabilistic Framework for Clustering Individuals and Objects*, KDD-2000, 140-149, ACM, New York, NY, 2000.
- [6] Cheeseman, P. and Stutz, J., *Bayesian Classification (AutoClass): Theory and Results*. In Advances in Knowledge Discovery and Data Mining, U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, & R. Uthurusamy, eds. The AAAI Press, Menlo Park, 1995
- [7] Chung-hung Cheng, Fu, A.W., Zhang, Y., *Entropy-based Subspace Clustering for Mining Numerical Data*, KDD-99, 84-93, AAAI Press, Menlo Park, CA, 1999.
- [8] Cover, T.M., Thomas, J.A., *Elements of Information Theory*, John Wiley & Sons, New York, NY, 1990.
- [9] Dempster, A., Laird, N. and Rubin, D., *Maximum likelihood from incomplete data via the EM algorithm*, Journal of the Royal Statistical Society, Series B, 39(1):1-38, 1977.
- [10] Dhillon, I.S., Fan, J., Guan, Y., *Efficient Clustering of Very Large Document Collections*, In R. Grossman, G. Kamath, and R. Naburu, editors, *Data Mining for Scientific and Engineering Applications*, Kluwer Academic Publ., 2001.
- [11] Dhillon I.S., *Co-clustering documents and words using Bipartite Spectral Graph Partitioning*, KDD-2001, 269-274, ACM, San Francisco, CA, 2001.
- [12] Duda, R.O., Hart, P.E., *Pattern Classification and Scene Analysis*, John Wiley & Sons, New York, NY, 1973.
- [13] Ester, M., Kriegel, H.-P., Sander, J., Xu, X., *A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*, Proceedings of the 2nd Intl. Conf. KDD, Portland, Oregon, 1996.
- [14] Ester, M., Kriegel, H.-P., Xu, X., *A Database Interface for Clustering in Large Spatial Databases*, Proceedings Of the 1st Intl. Conf. KDD, Montreal, Canada, 1995.
- [15] Everitt, B.S., *Cluster Analysis*, Arnold, London, 1993.
- [16] Foss, A., Wang, W., Zaiane, O.R., *A Non-parametric Approach to Web Log Analysis*, *Workshop on Web Mining*, First SIAM Intl. Conf. on Data Mining, Chicago, 41-50, 2001.
- [17] Fraley, C., Raftery, A.E., *How Many Clusters? Which Clustering Method? Answers via Model-based Cluster Analysis*, *Computer Journal*, 4, 578-588, 1998.
- [18] Fraley, C., Raftery, A., *MCLUST: Software for model-based cluster and discriminant analysis*, Tech Report 342, Dept. Statistics, Univ. of Washington, 1999.

- [19] Fu, Y., Sandhu K., Shih, M., *Clustering of Web Users Based on Access Patterns*, WEBKDD-99, San Diego, 1999.
- [20] Gaffney, S., Smyth, P., *Trajectory Clustering with Mixtures of Regression Models*, KDD-99, 63-72, AAAI Press, Menlo Park, CA, 1999.
- [21] Harel, D., Koren, Y., *Clustering Spatial Data Using Random Walks*, KDD-2001, 281-286, ACM, San Francisco, CA, 2001.
- [22] Hartigan, J.A., *Clustering Algorithms*, Wiley, 1975.
- [23] Jain, A., Dubes, R., *Algorithms for Clustering Data*, Prentice-Hall, Englewood Cliffs, NJ, 1998.
- [24] Kandogan, E., *Visualizing Multi-dimensional Clusters, Trends, and Outliers Using Star Coordinates*, KDD-2001, 107-116, ACM, San Francisco, CA, 2001.
- [25] Kaufman, L., Rousseeuw, P.J., *Finding Groups in Data: An Introduction to Cluster Analysis*, John Wiley and Sons, 1990.
- [26] Lin, D., *An Information-Theoretic Definition of Similarity*, ICML '98, 296-304, Morgan Kaufmann, San Francisco, CA, 1998.
- [27] McCallum, A.K., Nigam, K., *Employing EM and Pool-based Active Learning for Text Classification*, Machine Learning: ICML'98, 350-367, Morgan Kaufmann, San Francisco, CA, 1998.
- [28] McLachlan, G. and Krishnan, T. *The EM Algorithm and Extensions*. John Wiley & Sons, New York, NY, 1997.
- [29] Milligan, G.W., Cooper, M.C., *An Examination of Procedures for Determining the Number of Clusters in Data Set*, Psychometrica, 50:159-179, 1985.
- [30] Mitchel, T.M., *Machine Learning*. McGraw-Hill, Boston, MA, 1997.
- [31] Oliver, J.O., Baxter, R.A., Wallace, C.S., *Unsupervised Learning Using MML*, Machine Learning, ICML '96, 1996.
- [32] Oyanagi, S., Kubota, K., Nakase, A., *Application of Matrix Clustering to Web Log Analysis and Access Prediction*, WEBKDD-2001, 13-21, San Francisco, CA, 2001.
- [33] Quinlan, J.R., *C4.5: Programs For Machine Learning*, Morgan Kaufmann, San Mateo, 1993.
- [34] Zhang, B., *Generalized K-Harmonic Means – Dynamic Weighting of Data in Unsupervised Learning*, First SIAM Intl. Conf. On Data Mining, Chicago, 2001.
- [35] Zhang, T., Ramakrishnan, R., Livny, M, *BIRCH: A New Data Clustering Algorithm and its Applications*, Data Mining and Knowledge Discovery, v. 1, no. 2, 1997.