Entropy-based Subspace Clustering for Mining Numerical Data

Chun-hung Cheng   Ada Wai-chee Fu   Yi Zhang
Department of Computer Science and Engineering
The Chinese University of Hong Kong
chcheng@cse.cuhk.edu.hk, adafu@cse.cuhk.edu.hk

Abstract

Mining numerical data is a relatively difficult problem in data mining. Clustering is one of the techniques. We consider a database with numerical attributes, in which each transaction is viewed as a multi-dimensional vector. By studying the clusters formed by these vectors, we can discover certain behaviors hidden in the data. Traditional clustering algorithms find clusters in the full space of the data sets. This results in high dimensional clusters, which are poorly comprehensible to human. One important task in this setting is the ability to discover clusters embedded in the subspaces of a high-dimensional data set. This problem is known as subspace clustering. We follow the basic assumptions of previous work CLIQUE. It is found that the number of subspaces with clustering is very large, and a criterion called the coverage is proposed in CLIQUE for the pruning. In addition to coverage, we identify new useful criteria for this problem and propose an entropy-based algorithm called ENCLUS to handle the criteria. Our major contributions are: (1) identify new meaningful criteria of high density and correlation of dimensions for goodness of clustering in subspaces, (2) introduce the use of entropy and provide evidence to support its use, (3) make use of two closure properties based on entropy to prune away uninteresting subspaces efficiently, (4) propose a mechanism to mine non-minimally correlated subspaces which are of interest because of strong clustering, (5) experiments are carried out to show the effectiveness of the proposed method.

1 Introduction

Modern technology provides efficient and low-cost methods for data collection. However, raw data is rarely of direct benefit for higher level management, decision making or more intelligent analysis. Data mining aims at the construction of semi-automatic tools for the analysis of large data sets. The mining of binary association rules has been extensively studied in recent years, but databases in the real world usually have numerical attributes in addition to binary attributes. Unfortunately, mining numerical data is a more difficult problem and relatively few works have been done on this topic. Some previous works include [15, 13, 14]. Here we attempt to mine numerical data using clustering techniques. We consider a database consisting of numerical attributes. We can view each transaction of this database as a multi-dimensional vector. Clustering is to discover homogeneous groups of objects based on the values of these vectors. Hence, we can study the behaviour of the objects by looking at the shapes and locations of clusters. See Figure 1 for an example.

We learn from statistics that it is possible to find correlation among different factors from raw data, but we cannot find the direction of implication and it can be risky to conclude any causal relationship from raw data [16]. Clustering is a method that finds correlations while not infering any causal relationship.

Not all clustering algorithms are suitable for our problem. They must satisfy some special requirements in order to be useful to us. One important requirement is the ability to discover clusters embedded in subspaces of high dimensional data. Given a space X with dimensions formed from a set of attributes S, a space Y with dimensions formed from a subset of S is called a **subspace** of X. Conversely, X will be called a **superspace** of Y. For instance, suppose there are three attributes A, B and C. Clusters may exist inside the subspace formed by A and B, while C is independent of A and B. In such case, C is a noise variable. Since high dimensional information is hard to interpret, it is more desirable if the clustering algorithm can present the cluster in the subspace AB rather than in the full space ABC. Real-life databases usually contain many attributes so that either there is no proper cluster in the full space, or knowing the existence of a cluster in the full space is of little use to the user. Therefore, the ability to discover embedded clusters is important. This problem is called **subspace clustering** in [2].

Data mining by definition deals with huge amount

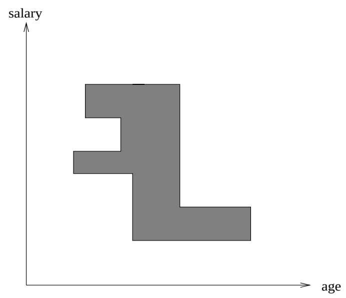

Figure 1: Example of a cluster.

of data, which are often measured in gigabytes or even terabytes. Although some traditional clustering algorithms are elegant and accurate, they involve too many complicated mathematical operations. These methods are shown to handle problem sizes of several hundreds to several thousands transactions, which is far from sufficient for data mining applications [7, 19]. We need an algorithm that gives reasonable performance even on high dimensionality and large data sets.

We prefer clustering algorithms that do not assume some restrictive shapes for the clusters. Some clustering algorithms (e.g. K-means [18, 5], CLARANS [20], BIRCH [25] and ScaleKM [4]) assume that the clusters are concave in shape. We would adopt a definition of cluster that does not have the above limitation. A good algorithm should also not make assumptions about the distribution of the data and not be sensitive to the existence of outliers. It should not require the users to specify some parameters on which the users would have difficulty to decide. For instance, the K-means algorithm requires the user to specify the number of clusters, which is often not known to the user. Finally there should be a meaningful and effective way to convey the resulting clusters to the users for the purpose of data mining.

A solution to the above problem would consist of the following steps: (1) Find the subspaces with good clustering. (2) Identify the clusters in the selected subspaces. (3) Present the result to the users. We shall focus on Step (1). The rest of this paper is organized as follows. In Section 2, we discuss the related work done on similar problems. Section 3 points out some new criteria for good clustering and explains why they are needed. In Section 4, we introduce the entropy-based method for locating subspaces with good clustering. In Section 5 we discuss how the entropy-based method is suitable for the listed criteria. Section 6 describes the proposed algorithm. In Section 7, we would look at some experimental results. Section 8 is a conclusion.

2 Related Work

There are quite a number of previous works on the clustering problem by the database research community. Some examples are CLARANS [20], DBSCAN [12], DBCLASD [24], Incremental DBSCAN [11], GRIDCLUS [23], CURE [17], BIRCH [25], and ScaleKM [4]. None of the above algorithms satisfies our most important requirement — the ability to identify clusters embedded in subspaces of high-dimensional data. CLIQUE [2] is the only published algorithm we are aware of that satisfies this requirement. Since we follow closely the problem setting of CLIQUE, we shall describe it in more details.

First we introduce the target problem and assumptions of CLIQUE [2]. A set of data points and two parameters, ξ and τ, are given. We discretize the data space $\mathcal{S}$ into non-overlapping rectangular *units*, which is obtained by partitioning every dimension into ξ intervals of equal length. A unit is **dense** if the fraction of total data points contained in the unit is greater than the threshold τ. **Clusters** are unions of connected dense units within a subspace. We need to identify the dense units in different subspaces. The CLIQUE algorithm can be divided into the following three steps: (1) Find dense units and identify subspaces containing clusters. (2) Identify clusters in the selected subspace. (3) Generate minimal description for the clusters in disjunctive normal form.

Although it is theoretically possible to create a histogram in all spaces to identify the dense units, this method would be computationally infeasible when the number of dimensions is large. To reduce the search space, a bottom-up algorithm is used that exploits the monotonicity of the clustering criterion with respect to dimensionality: if a collection of points S is a cluster in a k-dimensional space, then S is also part of a cluster in any $(k-1)$-dimensional projections of the space. The algorithm is iterative: First find 1-dimensional dense units by making a pass over the data. Having determined $(k-1)$-dimensional dense units, D_{k-1}, the candidate k-dimensional units, C_k, are determined using the candidate generation procedures. A pass is made over the data to determine those candidate units that are dense, D_k. The algorithm iterates the above with increasing dimensionality, and terminates if no candidates are found. The candidate generation procedure is similar to the one adopted in the well-known Apriori algorithm [1] for mining association rules.

As the number of dimensions increases, the above method may produce a large amount of dense units in the subspace and the pruning above may not be effective enough. CLIQUE uses a new criteria for the pruning of subspace which is based on the **coverage**. The coverage of a subspace is the fraction of the database that is covered by the dense units. Subspaces with high coverages are selected and those with low coverages are pruned away.

When the subspaces containing clusters are identified, the clusters in each subspace are to be determined. Recall that clusters are connected dense units. We can simply use a depth-first search algorithm [3] to find the connected components. The final step is to generate minimal cluster descriptions. The description is given in form of DNF expression, e.g. $((30 \leq \text{age} < 50) \wedge (4 \leq \text{salary} < 8)) \vee ((40 \leq \text{age} < 60) \wedge (2 \leq \text{salary} < 6))$. This is equivalent to a union of some hyper-rectangular regions. The regions can be found by a greedy growth method. We start with any dense unit and greedily grow a maximal region in each dimension. The process is repeated until the union of all regions cover the whole cluster. Then we need to remove the redundant regions. This is achieved by repeatedly removing the smallest redundant region until no maximal region can be removed.

3 Criteria of Subspace Clustering

There are many factors to be considered for a clustering algorithm in data mining. We mentioned some of these in the introduction: efficiency, shape of clusters, sensitivity to outliers, and the requirements of parameters. A clustering algorithm will assume a certain set of criteria for a cluster, as well as criteria for what is a good clustering given a set of data.

In addition to the clustering problem, we would like to handle the problem of determining subspaces that have "good clustering". We therefore need addition criteria for determining which of two clustering for two different sets of data is better. In the following we introduce a number of such criteria.

3.1 Criterion of High Coverage

The first criterion that we use for goodness of clustering is the **coverage** as defined for CLIQUE. This is a reasonable criterion since a subspace with more distinguished clusters will have high coverage, whereas a subspace with close to random data distribution will have low coverage.

3.2 Criterion of high density

Other than coverage, we believe that other criteria are also needed. The first criterion that we add is the criterion of high density.

Suppose we use only the coverage for measurement of goodness. A problem case is illustrated in Figure 2. It shows the probability density function of a random variable X. The value of coverage can be represented by the area of the shade portion since coverage is the fraction of the database that is covered by the dense units. In this example, both cases (a) and (b) have

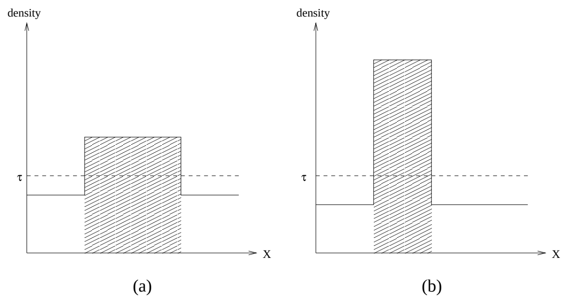

Figure 2: Example of two data sets with equal coverage but different densities. The area of the shaded portion is the value of coverage.

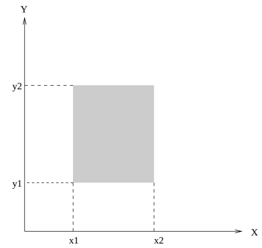

Figure 3: Problem with independent dimensions

the same coverage. However, this contradicts with our intuition, because the points in case (b) is more closely packed and more qualified as a cluster.

3.3 Correlation of dimensions

The third criterion that we consider is related to the correlation of dimensions. We note that finding subspaces with good clustering may not always be helpful, we also want the dimensions of the subspace to be correlated. The reason is that although a subspace may contain clusters, this may not be interesting to us if the dimensions are independent to each other. For example, Figure 3 shows such a scenario in 2D. In this example, since all the data points projected on X lies on $[x1, x2)$ and projected on Y lies on $[y1, y2)$, the data objects must be distributed at $[x1, x2) \times [y1, y2)$ in the joint space. If the points are uniformly distributed at $[x1, x2) \times [y1, y2)$, although there is a cluster, looking at the joint space gives us no more knowledge than looking at each of the dimensions independently.

Hence, we also require the dimensions of the subspace to be correlated. Note that when we say correlated here, we mean the dimensions are not completely independent but it need not exist a very strong correlation.

Having identified a number of criteria for clustering, we shall find a metric that can measure all the criteria simultaneously. A subspace which has good clustering by the criteria will have high score in this metric. Then we can set a threshold on this measurement and find

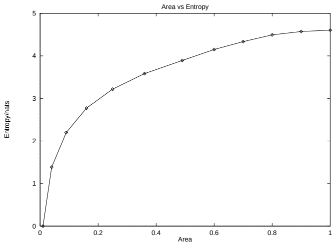

Figure 4: Area of Cluster vs Entropy

subspaces which exceed this threshold. The metric that we use is the entropy, which we shall discuss in the next section.

4 Entropy-based Method

We propose to use an entropy-based method. The method is motivated by the fact that a subspace with clusters typically has lower entropy than a subspace without clusters. Entropy is a measure of uncertainty of a random variable. Let X be a discrete random variable, $\mathcal{X}$ be the set of possible outcomes of X and $p(x)$ be the probability mass function of the random variable X. The entropy $H(X)$ is defined by the following expression [9].

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

If the base of log is 2, the unit for entropy is bit. When there are more than one variable, we can calculate the joint entropy to measure their uncertainty.

$$\begin{aligned} & H(X_1, \ldots, X_n) \\ = & -\sum_{x_1 \in \mathcal{X}_1} \cdots \sum_{x_n \in \mathcal{X}_n} p(x_1, \ldots, x_n) \log p(x_1, \ldots, x_n) \end{aligned}$$

When the probability is uniformly distributed, we are most uncertain about the outcome. Entropy is the highest in this case. On the other hand, when the data points have a highly skewed probability mass function, we know that the variable is likely to fall within a small set of outcomes so the uncertainty and the entropy are low.

4.1 Calculation of Entropy

Similar to CLIQUE, we divide each dimension into intervals of equal length Δ, so the high-dimensional space is partitioned to form a grid. Suppose the data set is scanned once to count the number of points contained in each cell of the grid. The density of each cell can thus be found. Let $\mathcal{X}$ be the set of all cells, and $d(x)$ be the density of a cell x in terms of the percentage of data contained in x. We define the entropy of the data set to be:

$$H(X) = -\sum_{x \in \mathcal{X}} d(x) \log d(x)$$

When the data points are uniformly distributed, we are most uncertain where a particular point would lie on. The entropy is the highest. When the data points are closely packed in a small cluster, we know that a particular point is likely to fall within the small area of the cluster, and so the uncertainty and entropy will be low. Figure 4 shows the result of an experiment studying the relationship between the area of cluster in a two dimensional space $[0,1) \times [0,1)$. The smaller the area of the cluster, the more closely packed the points and the lower the entropy.

The size of interval Δ must be carefully selected. If the interval size is too small, there will be many cells so that the average number of points in each cell can be too small. On the other hand, if the interval size is too large, we may not able to capture the differences in density in different regions of the space. We suggest the use of at least 35 points in each cell on the average since 35 is often considered as the minimum sample size for large sample procedures [10]. The size of interval Δ can be set accordingly.

5 Entropy vs the Clustering Criteria

In Section 3, we propose the use of three criteria for the goodness of clustering: high coverage[1], high density and dimensional correlation. In this section, we discuss how the use of entropy can relate to the criteria we have chosen for the selection of subspaces. First we list the symbols used in the discussion in Table 1.

n	the total number of units
k	the total number of dense units
τ	the threshold on the density of a dense unit (in percentage of the data)
c	coverage (percentage of data covered by all dense units)
$p_1, \ldots, p_k$	the densities of the dense units
$p_{k+1}, \ldots, p_n$	the densities of the non-dense units

Table 1: Notations used

5.1 Entropy and the coverage criterion

To investigate the relationship between entropy and the coverage, we consider the following case. By assuming there are k dense units out of n total units, it follows that

$$\begin{aligned} p_1 + \ldots + p_k &= c \\ p_{k+1} + \ldots + p_n &= 1 - c \end{aligned}$$

[1] Coverage is the percentage of data covered by all dense units in a particular subspace. The original authors of CLIQUE define it as the number of objects covered by all dense units. Our definition is slightly different here.

$$
\begin{aligned}
\frac{dp_1}{dc} + \ldots + \frac{dp_k}{dc} &= 1 \\
\frac{dp_{k+1}}{dc} + \ldots + \frac{dp_n}{dc} &= -1
\end{aligned}
$$

We want to estabish the relationship that, under certain conditions, the entropy decreases as the coverage increases, i.e. $\frac{dH(X)}{dc} \leq 0$.

Theorem 1 $\frac{dH(X)}{dc} \leq 0$ *if and only if* $p_1^{\frac{dp_1}{dc}} \ldots p_n^{\frac{dp_n}{dc}} \geq 1$.

Proof

$$
\begin{aligned}
H(X) &= -\sum_{i=1}^{n} p_i \log p_i \\
&= -\sum_{i=1}^{k} p_i \log p_i - \sum_{j=k+1}^{n} p_j \log p_j
\end{aligned}
$$

Let us differentiate the entropy with respect to the coverage.

$$
\begin{aligned}
\frac{dH(X)}{dc} &= -\sum_{i=1}^{k}\Big[\frac{dp_i}{dc}\log p_i + \frac{dp_i}{dc}\Big] - \sum_{j=k+1}^{n}\Big[\frac{dp_j}{dc}\log p_j + \frac{dp_j}{dc}\Big] \\
&= -\sum_{i=1}^{k}\left[\frac{dp_i}{dc}\log p_i\right] - 1 - \sum_{j=k+1}^{n}\left[\frac{dp_j}{dc}\log p_j\right] + 1 \\
&= -\sum_{i=1}^{k}\left[\frac{dp_i}{dc}\log p_i\right] - \sum_{j=k+1}^{n}\left[\frac{dp_j}{dc}\log p_j\right] \\
&= -\log\left[p_1^{\frac{dp_1}{dc}} \ldots p_n^{\frac{dp_n}{dc}}\right]
\end{aligned}
$$

The result follows and the proof is completed. □

Now we have the necessary and sufficient condition for our desirable property to hold. However, the condition is complicated and difficult to understand. Further investigation is needed to make it more comprehensive.

Theorem 2 *Suppose that* $\frac{dp_i}{dc} \geq 0$ *for* $i = 1, \ldots, k$ *and* $\frac{dp_j}{dc} \leq 0$ *for* $j = k+1, \ldots, n$ *and* $\min_{1\leq i\leq k}(p_i) \geq \max_{k+1\leq j\leq n}(p_j)$. *Then we have*

$$
\frac{dH(X)}{dc} \leq 0
$$

Proof

$$
\begin{aligned}
& p_1^{\frac{dp_1}{dc}} \ldots p_n^{\frac{dp_n}{dc}} \\
\geq\ & \left[\min_{1\leq i\leq k}(p_i)\right]^{\frac{dp_1}{dc}+\ldots+\frac{dp_k}{dc}} \cdot \left[\max_{k+1\leq j\leq k}(p_j)\right]^{\frac{dp_{k+1}}{dc}+\ldots+\frac{dp_n}{dc}} \\
=\ & \frac{\min_{1\leq i\leq k}(p_i)}{\max_{k+1\leq j\leq n}(p_j)} \\
\geq\ & 1
\end{aligned}
$$

Then, Theorem 1 applies and the proof is completed. □

The conditions of Theorem 2 hold when the coverage is increased by increasing the densities of some denser units and decreasing the densities of some non-dense units. Although it is not true for all conditions, this is a supportive evidence of the use of entropy to reflect the coverage of clustering for a subspace.

5.2 Entropy and the density criterion

In the example shown in Figure 2, entropy of case (b) is lower than that of case (a), which suggests case (b) is a better cluster. We see that entropy can better capture our intuition of good clustering as compared to the mere use of coverage. To examine the relationship between entropy and density, we consider the following case. Assume that the density of dense units are all equal to α, the density of non-dense units are all equal to β. The total number of dense units is k and thus the total number of non-dense units is $n-k$. Then we have

$$
\begin{aligned}
H(X) &= -\sum_{i=1}^{k} p_i \log p_i \\
&= -\left(\sum_{i=1}^{k} p_i \log p_i + \sum_{j=k+1}^{n} p_j \log p_j\right) \\
&= -[k\alpha\log\alpha + (n-k)\beta\log\beta]
\end{aligned}
$$

By assuming that α and β change continuously, the entropy becomes a differentiable function of density.

Theorem 3 $\frac{dH(X)}{d\alpha} \leq 0$ *if and only if* $\alpha \geq \beta$.

Proof Note that

$$
k\alpha + (n-k)\beta = 1
$$

So

$$
k + (n-k)\frac{d\beta}{d\alpha} = 0
$$

Differential the entropy with respect to the density α, then we have

$$
\begin{aligned}
\frac{dH(X)}{d\alpha} &= -\left[k\log\alpha + k + (n-k)\frac{d\beta}{d\alpha}(\log\beta + 1)\right] \\
&= -k[\log\alpha - \log\beta] \\
&= k\log\frac{\beta}{\alpha}
\end{aligned}
$$

This shows that $\frac{dH(X)}{d\alpha} \leq 0$ if and only if $\alpha \geq \beta$. The proof is completed. □

This says that as the density of the dense units increases, the entropy decreases. Hence entropy can relate to the measurement of density in the clustering of a subspace.

5.3 Entropy and variable correlation

The problem of correlated variables can be easily handled by entropy because the independence and dependence of the variables can be detected using the following relationships in entropy [8].

$$H(X_1, \ldots, X_n) = H(X_1) + \ldots + H(X_n)$$
$$\text{iff } X_1, \ldots, X_n \text{ are independent} \quad (1)$$

$$H(X_1, \ldots, X_n, Y) = H(X_1, \ldots, X_n)$$
$$\text{iff } Y \text{ is a function of } X_1, \ldots, X_n \quad (2)$$

We shall make use of the above property in the following section.

6 Algorithm

In this section, we introduce the proposed algorithm ENCLUS in more details. There are two variations of ENCLUS, which are discussed at Section 6.2 and 6.3. The overall strategy consists of three main steps:

1. Find out the subspaces with good clustering by an entropy-based method.
2. Identify the clusters in the subspace found.
3. Present the result to the users.

In Step 2 and Step 3, we can adopt the method in CLIQUE or some of the existing clustering algorithms. We examine Step 1. Previously we use the term good clustering to indicate that a subspace contains a good set of clusters in an intuitive sense. Here we shall give the term a more concrete definition by means of entropy. We need to set a threshold ω. A subspace whose entropy is below ω is considered to have **good clustering**. The proposed algorithm uses a bottom-up approach similar to the Apriori algorithm [1] for mining association rule. In Apriori, we start with finding large 1-itemsets. It is used to generate the candidate 2-itemsets, which are checked against the database to determine large 2-itemsets. The process is repeated with increasing itemset sizes until no more large itemset is found.

Similarly, our bottom-up algorithm starts with finding one-dimensional subspaces with good clustering. Then we use them to generate the candidate two-dimensional subspaces and check them against the raw data to determine those that actually have good clustering. The process is repeated with increasing dimensionalities until no more subspaces with good clustering is found. We note a downward closure property for entropy. This is given by the non-negativity of Shannon's information measures[2] [8]. The correctness of the bottom-up approach is based on this property.

[2] The values of entropy, conditional entropy, mutual information and conditional mutual information are always non-negative. This is not true to differential entropy because the value of differential entropy may be either positive or negative.

Lemma 1 *(Downward closure) If a k-dimensional subspace $X_1, \ldots, X_k$ has good clustering, so do all $(k-1)$-dimensional projections of this space.*

Proof Since the subspace $X_1, \ldots, X_k$ has good clustering, $H(X_1, \ldots, X_k) < \omega$.

$$\begin{aligned} & H(X_1, \ldots, X_{k-1}) \\ \leq \; & H(X_1, \ldots, X_{k-1}) + H(X_k | X_1, \ldots, X_{k-1}) \\ = \; & H(X_1, \ldots, X_k) \\ < \; & \omega \end{aligned}$$

Hence, the $(k-1)$-dimensional projection $X_1, \ldots, X_k$ also has good clustering. The above proof can be repeated for other $(k-1)$-dimensional projections. □

6.1 Dimensions Correlation

In Section 3.3 we discuss the criterion of dimensional correlation. In Section 5.3 we examine how entropy can be related to dimensional correlation. Here we show the upward closure property of this criterion. Let $p(x_1, x_2, ..., x_i)$ be the joint probability mass function of variables $X_1, X_2, ..., X_i$. In the following lemma, variables $X_1, X_2, ..., X_n$ are considered not correlation iff $p(x_1, x_2, ..., x_n) = p(x_1)p(x_2)...p(x_n)$.

Lemma 2 *(Upward closure) If a set of dimensions S is correlated, so is every superset of S.*

Proof We proof by contradiction. Suppose X_1 and X_2 are correlated, but X_1, X_2 and X_3 are not.

$$\begin{aligned} p(x_1, x_2) &= \int_{X_3} p(x_1, x_2, x_3) dx_3 \\ &= \int_{X_3} p(x_1)p(x_2)p(x_3) dx_3 \\ &= p(x_1)p(x_2) \int_{X_3} p(x_3) dx_3 \\ &= p(x_1)p(x_2) \end{aligned}$$

Hence X_1 and X_2 are not correlated, which is a contradiction. □

Traditionally, the correlation between two numerical variables can be measured using the correlation coefficient. However, we can also detect correlation by entropy. Since we are already using entropy in the algorithm, using entropy to detect correlation introduces a negligible computational overhead. A set of variables $X_1, \ldots, X_n$ are correlated if Equation 1 of Section 5.3 is not satisfied. To express it more precisely, we define the term *interest*[3] as below.

$$interest(\{X_1, \ldots, X_n\}) = \sum_{i=1}^{n} H(X_i) - H(X_1, \ldots, X_n)$$

[3] The definition of interest is equivalent to the mutual information between all individual dimensions of a subspace $I(X_1; X_2; \ldots; X_n)$. We use the term interest instead of "mutual information between all individual dimensions" to simplify our terminology.

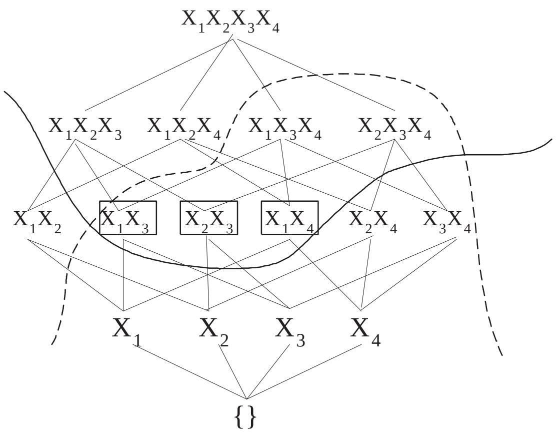

Figure 5: A lattice for 4 variables.

k	Current number of iterations
C_k	Set of k-dimensional candidate subspaces
S_k	Set of k-dimensional significant subspaces
NS_k	Set of k-dimensional subspaces with good clustering but not minimally correlated

Table 2: Notations used in the algorithm

The higher the interest, the stronger the correlation. We consider the variables to be correlated if and only if the interest exceeds a predefined threshold ϵ. The interests of one-dimensional subspaces are always 0.

6.2 Mining Significant Subspaces

A pair of downward and upward closure properties is used in [6], which proposes an algorithm for mining correlation rules. It is pointed out that downward closure is a pruning property. If a subspace does not satisfy this property, we can cross out all its superspaces because we know they cannot satisfy this property either. Upward closure, by contrast, is a constructive property. If a subspace satisfies the property, all its superspaces also satisfy this property. However, upward closure property is also useful for pruning. The trick is that we only find *minimally* correlated subspaces. If we know a subspace is correlated, all its superspaces must not be minimally correlated. Therefore, upward closure becomes a pruning property.

Now we call the subspaces with good clustering and minimally correlated to be *significant subspaces*. Due to the upward closure property, the subspaces we are interested in form a *border*. The border stores all the necessary information. Refer to Figure 5 for an example. In this figure, the subspaces below the dotted lines all have good clustering (downward closed) and the subspaces above the solid lines are all correlated (upward closed). The border $\{X_1X_3, X_2X_3, X_1X_4\}$ stores all the significant subspaces, i.e. minimally correlated subspaces with good clustering.

Algorithm 1 ENCLUS_SIG(ω, ϵ)

1 $k = 1$
2 Let C_k be all one-dimensional subspaces.
3 For each subspace $c \in C_k$ do
4 $\quad f_c(\cdot) = \text{cal\_density}(c)$
5 $\quad H(c) = \text{cal\_entropy}(f_c(\cdot))$
6 $\quad$ If $H(c) < \omega$ then
7 $\quad\quad$ If $interest(c) > \epsilon$ then
8 $\quad\quad\quad S_k = S_k \cup c$.
9 $\quad\quad$ else
10 $\quad\quad\quad NS_k = NS_k \cup c$.
11 End For
12 $C_{k+1} = \text{candidate\_gen}(NS_k)$
13 If $C_{k+1} = \emptyset$, go to step 16.
14 $k = k + 1$
15 Go to step 3.
16 $Result = \bigcup_{\forall k} S_k$

Figure 6: Algorithm for mining significant subspaces

The details of the algorithm, called ENCLUS_SIG, are given in Figure 6. Table 2 lists the notations used. The description of the procedures used in the algorithm is given as follows.

cal_density(c) Build a grid to count number of points that fall in each cell of the grid as described in Section 4.1. The density of each cell can thus be estimated.

cal_entropy($f_c(\cdot)$) Calculate the entropy using the density information obtained from scanning the data set.

candidate_gen(NS_k) Generate the candidate subspaces for $(k+1)$ dimensions using NS_k. There is a join step and a prune step in the candidate generation function. The join step can be expressed by the following pseudo-code. It joins two subspaces having common first $(k-1)$ dimensions.

insert **into** C_{k+1}
select $p.dim_1, p.dim_2, \ldots, p.dim_k, q.dim_k$
from $NS_k\ p$, $NS_k\ q$
where $p.dim_1 = q.dim_1, \ldots, p.dim_{k-1} = q.dim_{k-1}, p.dim_k < q.dim_k$

In the prune step, any subspace having a k-dimensional projection outside NS_k is removed.

6.3 Mining Interesting Subspaces

Since correlation can usually be detected at low dimension, the mining of high dimensional clusters is often avoided. This is good because low dimensional clusters are easier to interpret and the time for mining high

Algorithm 2 ENCLUS_INT(ω, ϵ')

1 $k = 1$
2 Let C_k be all one-dimensional subspaces.
3 For each subspace $c \in C_k$ do
4 $\quad f_c(\cdot) = \text{cal\_density}(c)$
5 $\quad H(c) = \text{cal\_entropy}(f_c(\cdot))$
6 $\quad$ If $H(c) < \omega$ then
7 $\qquad$ If $interest\_gain(c) > \epsilon'$ then
8 $\qquad\quad I_k = I_k \cup c.$
9 $\qquad$ else
10 $\qquad\quad NI_k = NI_k \cup c.$
11 End For
12 $C_{k+1} = \text{candidate\_gen}(I_k \cup NI_k)$
13 If $C_{k+1} = \emptyset$, go to step 16.
14 $k = k + 1$
15 Go to step 3.
16 $Result = \bigcup_{\forall k} I_k$

Figure 7: Algorithm for mining interesting subspaces

dimensional clusters can be saved. However, [6] did not consider that sometimes we are interested in non-minimally correlated subspaces. For instance, A and B are correlated, but we may be interested in the subspace ABC if ABC are more strongly correlated than A and B alone. To measure the increase in correlation, we define the term *interest gain*[4]. The interest gain for subspace $X_1, \ldots, X_n$ is defined as follows.

$$\begin{aligned} & interest\_gain(\{X_1, \ldots, X_n\}) \\ = \; & interest(\{X_1, \ldots, X_n\}) - \\ & \max_i \{interest(\{X_1, \ldots, X_n\} - \{X_i\})\} \end{aligned}$$

The interest gain for one dimensional subspace is defined to be 0. The interest gain of a k-dimensional subspace is the interest of the given subspace minus the maximum interest of its $(k-1)$-dimensional projections. In other words, it is the increase in interest for adding an extra dimension.

Our new goal becomes mining subspaces whose entropy exceeds ω and interest gain exceeds a new threshold ϵ'. We call such subspaces to be *interesting subspaces*. The mining of significant subspace algorithm can be modified slightly to mine interesting subspaces. Figure 7 shows the modified algorithm ENCLUS_INT. Since we relax one of the pruning criteria, we expect more candidates and a longer running time.

[4]The definition of interest gain is equivalent to the mutual information between the original subspace $X_{i_1}, \ldots, X_{i_{n-1}}$ and a new dimension X_{i_n}, i.e. $I(X_{i_1}, \ldots, X_{i_{n-1}}; X_{i_n})$. We use the term interest gain instead of "mutual information between the original subspace and the new dimension" to simplify our terminology.

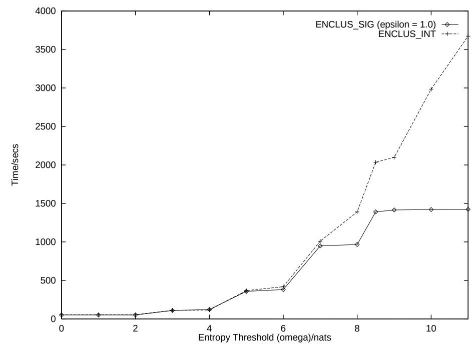

Figure 8: Entropy Threshold vs Running Time

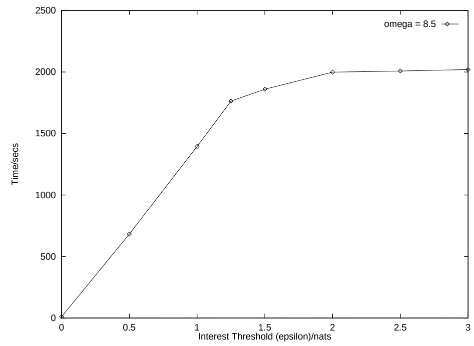

Figure 9: Interest Threshold vs Running Time (ENCLUS_SIG)

7 Experiments

To evaluate the performance of the algorithms, we implemented our algorithms on Sun Ultra 5/270 workstation using GNU C++ compiler. High dimensional synthetic data were generated which contains clusters embedded in the subspaces. Our data generator allows the user to specify the dimensionality of data, the number of subspaces containing clusters, the dimensionality of clusters, the number of clusters in each subspace and the number of transactions supporting each cluster. Unless otherwise specified, we use data of 10 dimensions and 300,000 transactions in the experiments. Some five-dimensional clusters are embedded in the subspaces.

Figure 8 shows the performance of the algorithms under different values of ω. We do not have a smooth curve here, because when ω increases to a certain value, candidates of a higher dimension are introduced which impose a considerable amount of extra computation. From the figure, we can see the running time of the algorithm ENCLUS_SIG ceases to increase when ω is high enough, because after that point, the pruning power of entropy is negligible and most pruning is attributed to the upward closure property which is independent of ω. As for the algorithm ENCLUS_INT, the running time keeps on increasing with ω because only the entropy is utilized for pruning.

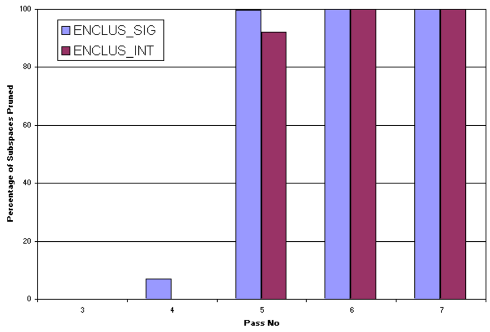

Figure 10: Pass No vs Percentage of Subspaces Pruned

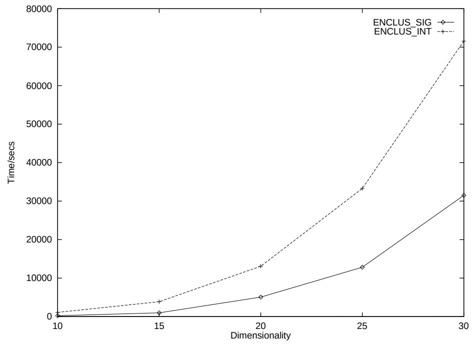

Figure 11: Scalability Test on Dimensionality of Data Set

Figure 9 shows the performance of the ENCLUS_SIG under different values of ϵ. Again, the running time ceases to increase after the a certain point. This is because the pruning power of interest is negligible and most pruning is done by entropy. We have also performed a similar set of experiments for ENCLUS_INT with $\omega = 8.5$. The performance of the ENCLUS_INT under different values of ϵ' is nearly identical, since only the entropy is used for pruning.

The pruning power of the algorithm is illustrated in Figure 10. Our methods are compared to the naive algorithm which examines all possible subspaces. From the result, we can see our methods give significant reduction on number of candidates in later passes. ENCLUS_SIG always prunes more candidates than ENCLUS_INT. This experiment is carried out with a 20-dimensional data set.

The result of the scalability test is shown in Figure 11. As expected, ENCLUS_SIG outperforms ENCLUS_INT because ENCLUS_SIG only finds minimally correlated subspaces while ENCLUS_INT has to pay extra time to mine the non-minimally correlated subspaces. The gap between ENCLUS_SIG and ENCLUS_INT increases with the dimensionality, which suggests the pruning power of the upward closure is more significant there. We have experimented with up to 30 dimensions. For higher dimensions, the computation time would increase further. We suggest that an approach similar to the minimal code length method in CLIQUE can be used which is based on entropy instead of coverage.

To investigate the accuracy of the algorithms, we performed an experiment using a data set containing five 5-dimensional clusters in five subspaces. ENCLUS_INT successfully discovers the five 5-dimensional subspaces that contains our embedded clusters without reporting false alarms of other 5-dimensional subspaces. ENCLUS_SIG expresses the correlated variables using a number of two-dimensional subspaces. It does not examine higher dimensional subspaces because they are not minimally correlated.

8 Conclusion

We propose to tackle the problem of mining numerical data using clustering techniques since each transaction with k attributes can be seen as a data point in a k-dimensional space. However, for large databases, there are typically a large number of attributes and the patterns that occur in subsets of these attributes are important. Mining for clusters in subspaces is therefore an important problem. The proposed solution consists of three steps, namely the identification of subspaces containing clusters, the discovery of clusters in selected subspaces and the presentation to the users. We concentrate on the subproblem of identifying subspaces containing clusters because few works have been done on it, one better known previous method is CLIQUE [2].

We propose using three criteria for the goodness of clustering in subspaces: coverage, density and correlation. Our proposed method is based on the measure of entropy from information theory, which typically gives a lower value for a subspace with good clustering. Although entropy has been used in decision trees for data mining [21, 22], to our knowledge, no previous work has used it for the problem of subspace clustering. We also justify the approach by establishing some relationship between entropy and the three criteria.

Our algorithm incorporates the idea of using a pair of downward and upward closure properties, which are first used by [6] in the problem of mining correlation rules. This approach is shown effective in the reduction of the search space. In our problem, the downward closure property is given by entropy while the upward closure property is given by the dimensional correlation. By the use of the two closure properties, the algorithm is expected to have good pruning power. Experiments have been carried out to show the proposed algorithm can correctly identify the significant/interesting subspaces and the pruning is effective and efficient.

Acknowlegments: We thank the anonymous refer-

ees for helpful suggestions.

References

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proceedings of the 20th VLDB Conference*, pages 487–499, 1994.

[2] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the ACM SIGMOD Conference on Management of Data, Montreal, Canada*, 1998.

[3] A. Aho, J. Hopcroft, and J. Ullman. *The Design and Analysis of Computer Algorithms.* Addison-Welsley, 1974.

[4] P. S. Bradley, Usama Fayyad, and Cory Reina. Scaling clustering algorithms to large databases. In *Proceedings of International Conference on Knowledge Discovery and Data Mining KDD-98, AAAI Press*, 1998.

[5] P. S. Bradley, O. L. Mangasarian, and W. Nick Street. Clustering via concave minimization. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems -9-*, pages 368–374, Cambridge, MA, 1997. MIT Press.

[6] Sergey Brin, Rajeev Motwani, and Craig Silverstein. Beyond market baskets: Generalizing association rules to correlations. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, 1997.

[7] David K. Y. Chiu and Andrew K. C. Wong. Synthesizing knowledge: A cluster analysis approach using event covering. In *IEEE Transactions on Sytems, Man, and Cybernetics, Vol. SMC-16, No. 2, March/April 1986*, pages 251–259, 1986.

[8] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory.* Wiley Series in Telecommunications, 1991.

[9] I. Csiszár and J. Körner. *Information Theory: Coding Theorems for Discrete Memoryless System.* Academic Press, 1981.

[10] Jay L. Devore. *Probability and Statistics for Engineering and the Sciences.* Duxbury Press, 4th edition, 1995.

[11] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Michael Wimmer, and Xiaowei Xu. Incremental clustering for mining in a data warehousing environment. In *Proceedings of the 24th VLDB Conference, New York, USA*, 1998.

[12] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of International Conference on Knowledge Discovery and Data Mining KDD-98, AAAI Press*, pages 226–231, 1996.

[13] Takeshi Fukuda, Yasuhiki Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, 1996.

[14] Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Constructing efficient decision trees by using optimized numeric association rules. In *Proceedings of the 22nd VLDB Conference, Mumbai(Bombay), India*, 1996.

[15] Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Mining optimized association rules for numeric attributes. In *Proceedings of the Fifteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, June 1996.

[16] Clark Glymour, David Madigan, Daryl Pregibon, and Padhraic Smyth. Statistical themes and lessons for data mining. *Data Mining and Knowledge Discovery*, 1:11–28, 1997.

[17] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. CURE: An efficient clustering algorithm for large databases. In *Proceedings of the ACM SIGMOD Conference on Management of Data, Montreal, Canada*, June 1996.

[18] John A. Hartigan. *Clustering algorithms.* Wiley, 1975.

[19] Pierre Michaud. Clustering techniques. In *Future Generation Computer Systems 13*, pages 135–147, 1997.

[20] Raymond T. Ng and Jiawei Han. Efficient and effective clustering methods for spatial data mining. In *Proceedings of the 20th VLDB Conference, Santiago, Chile*, 1994.

[21] J.R. Quinlan. Induction of decision trees. In *Machine Learning*, pages 81–106. Kluwer Academic Publishers, 1986.

[22] J.R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, 1993.

[23] Erich Schikuta. Grid-clustering: An efficient hierarchical clustering method for very large data sets. In *Proceedings of Internation Conference on Pattern Recognition (ICPR)*, pages 101–105, 1996.

[24] Xiaowei Xu, Martin Ester, Hans-Peter Kriegel, and Jörg Sander. A distribution-based clustering algorithm for mining in large spatial databases. In *Proceedings of 14th International Conference on Data Engineering (ICDE'98)*, 1998.

[25] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. BIRCH: An efficient data clustering method for very large databases. In *Proceedings of the ACM SIGMOD Conference on Management of Data, Montreal, Canada*, pages 103–114, June 1996.