Generalizing the Notion of Support

Michael Steinbach
Dept. of Comp. Sci. & Eng.
University of Minnesota
steinbac@cs.umn.edu

Pang-Ning Tan
Dept. of Comp. Sci. & Eng.
Michigan State University
ptan@cse.msu.edu

Hui Xiong, Vipin Kumar
Dept. of Comp. Sci. & Eng.
University of Minnesota
{huix,kumar}@cs.umn.edu

ABSTRACT

The goal of this paper is to show that generalizing the notion of support can be useful in extending association analysis to non-traditional types of patterns and non-binary data. To that end, we describe a framework for generalizing support that is based on the simple, but useful observation that support can be viewed as the composition of two functions: a function that evaluates the strength or presence of a pattern in each object (transaction) and a function that summarizes these evaluations with a single number. A key goal of any framework is to allow people to more easily express, explore, and communicate ideas, and hence, we illustrate how our support framework can be used to describe support for a variety of commonly used association patterns, such as frequent itemsets, general Boolean patterns, and error-tolerant itemsets. We also present two examples of the practical usefulness of generalized support. One example shows the usefulness of support functions for continuous data. Another example shows how the hyperclique pattern—an association pattern originally defined for binary data—can be extended to continuous data by generalizing a support function.

Categories and Subject Descriptors: H.2.8 [Database Management]: Database Applications - Data Mining

General Terms: Algorithms, Theory

Keywords: association analysis, support, hyperclique

1. INTRODUCTION

For binary transaction data, the *support* of a set of binary attributes (items) X is the number of objects (transactions) for which all the attributes of X have a value of 1. While simple, this notion of support is central to the definition of frequent and maximal itemsets, association rules, sequential patterns, and other ideas in the area of data mining known as association analysis [1, 2, 6, 7, 14]. Nonetheless, few efforts to extend association analysis to handle non-traditional types of patterns and non-binary data do so by modifying the notion of support, and those efforts that do have been specific to the work at hand. Thus, an overall framework for understanding and extending support is still lacking.

The goal of this paper is to provide such a framework and show its usefulness. Towards that end, this paper makes the following contributions:

We introduce a framework for support based on a view of support as the composition of two functions: a pattern evaluation function that evaluates the strength or presence of a pattern in each object (transaction) and a summarization function that summarizes these evaluations with a single number. Since a key goal of any framework is to allow people to more easily express, explore, and communicate ideas, we illustrate how our framework can be used to describe support for a variety of association patterns. This includes support for traditional frequent itemsets, as well as support for association patterns such as those based on general Boolean formulas [3, 10] and error-tolerant itemsets (ETIs) [13].

We extend traditional support measures to data sets with continuous attributes. Traditional support measures were designed for binary data, and although a continuous attribute can be mapped to binary attributes, this technique has some well known limitations, e.g., information is lost. We illustrate this fact and the usefulness of support functions for continuous data through an example based on Min-Apriori [5]. Also, because the anti-monotone property of support is important for the efficient generation of association patterns, we investigate the conditions under which support measures for continuous data possess this property.

We show how an association pattern defined for binary data, the hyperclique pattern [12], can be extended to continuous data by using a generalized notion of support. The key step is to choose pattern evaluation and summarization functions to construct a version of support that preserves both the anti-monotone and high affinity properties of the hyperclique pattern. The high affinity property guarantees that the attributes in a hyperclique are pairwise similar to one another at some minimum level, e.g., have a pairwise cosine similarity of 0.5.

2. TRADITIONAL SUPPORT

In this section, we review the definitions of support-based concepts used in traditional transaction analysis. An overview of the notation used in this and later sections is provided in Table 1. Also, throughout this document, the terms 'row,' 'transaction,' and 'object' are used interchangeably, as are the terms 'column,' 'item,' 'variable,' and 'attribute.'

Given binary transaction data, the *support* of a set of binary attributes (items) X is the number of objects (transactions) for which all the attributes of X have a value of 1. More formally, for a binary data matrix $\mathcal{D}$, the *support* of an

Table 1: Summary of Notation

Notation	Description
$\mathcal{D}$	Data matrix of M rows and N columns
$\mathcal{T} = \{t_1, t_2, \cdots, t_M\}$	Set of objects (transactions, rows) of $\mathcal{D}$
$\mathcal{I} = \{i_1, i_2, \cdots, i_N\}$	Set of attributes (items, variables, columns) of $\mathcal{D}$
t	An object (transaction, row) or its index
i, j, k	An attribute (item, variable, column) or its index
$\mathbf{i}$, $\mathbf{j}$	An attribute (item, variable, column) considered as a vector
X, Y	A set of attributes (items)

itemset $X \subseteq \mathcal{I}$ is given by $\sigma(X) = |\{t \in \mathcal{T} : \mathcal{D}(t, i) = 1, \forall i \in X\}$ where $|\{\cdot\}|$ denotes the number of elements that belong to a given set. An itemset is *frequent* if $\sigma(X) > minsup$, where *minsup* is a specified minimum support threshold.

An association rule, $X \rightarrow Y$, describes a relationship between two itemsets X and Y such that the items of Y occur in a transaction whenever the items of X occur. We measure the strength of such a relationship by the *support of an association rule*, $\sigma(X \rightarrow Y) = \sigma(X \cup Y)$, which is the number of transactions in which the relationship holds, and by the *confidence of an association rule*, $conf(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)} = \frac{\sigma(X \rightarrow Y)}{\sigma(X)}$, which is the fraction of transactions containing the items of X that also contain the items of Y.

An important property of support is the *anti-monotone property*: If X and Y are two itemsets where $X \subseteq Y$, then $\sigma(Y) \leq \sigma(X)$. The downward closure or anti-monotone property [14] of standard support can be used to efficiently find frequent itemsets and is the foundation of the well-known Apriori algorithm [1]. If a new support measure also possesses the anti-monotone property, then we may also be able to find its associated patterns efficiently, and thus, in what follows, we shall often focus on this issue.

3. A GENERAL SUPPORT FRAMEWORK

3.1 Basics

In the next three subsections, we describe the three concepts that are fundamental to our support framework: pattern evaluation (*eval*) functions, summarization (*norm*) functions, and the support functions that can be created from *eval* and *norm* functions. We then show how this support framework can be used to express support for frequent itemsets, general Boolean patterns, and Error Tolerant Itemsets (ETIs) [13].

3.1.1 Pattern Evaluation Functions

The evaluation of the strength of a pattern can take various forms. Most commonly, and this is the case for traditional association analysis, the pattern is either present, i.e., the pattern strength is 1, or it is absent, i.e., the pattern strength is 0. An example of such a pattern is the elementwise 'and' as defined in Table 2. In other situations, such as continuous or count data, a binary evaluation of pattern strength may not be as interesting. For example, suppose that we are interested in sets of values that are relatively homogenous within an object. Then, for non-binary data, the range of the attribute values might be a useful measure of pattern strength—one that gives a wider variation in strength than 0 and 1. This might be useful for count data, such as that in Table 4, which shows the number of times that a term occurs in a document. However, we may want to combine both of the preceding approaches, by measuring the strength of the pattern using a continuous measure, such as the range, but then evaluating whether this measure meets a specified condition, such as whether the range of the values is less than a specified threshold.

Thus, an evaluation function, *eval*, is a function that takes a set of of attributes $X \subseteq \mathcal{I}$ as an argument, and returns a *pattern evaluation vector*, $\mathbf{v}$, whose i^{th} component is the strength of the target pattern in the i^{th} object. More formally, we can write

$$\mathbf{v} = eval(X), \text{ or} \tag{1}$$

$$\mathbf{v}(t) = eval(t, X), \ \forall t \in \mathcal{T} \tag{2}$$

If there are several sets of attributes under consideration, e.g., X and Y, then we will distinguish between their pattern evaluation vectors by using subscripts, e.g., $\mathbf{v}_X$ and $\mathbf{v}_Y$. Notice that an *eval* function may be applied either to a single object, in which case, it returns a single value, or to a set of objects, in which case, it returns a vector of values. Various *eval* functions are shown in Table 2.

Table 2: ***eval*** **functions.** $X = \{i_1, i_2, \cdots, i_k\} \subseteq \mathcal{I}$.

	eval function	Definition
1	$\wedge$	$eval_{\wedge}(t, X) = \mathcal{D}(t, i_1) \wedge \ldots \wedge \mathcal{D}(t, i_k)$
2	$\prod$	$eval_{\prod}(t, X) = \mathcal{D}(t, i_1) * \ldots * \mathcal{D}(t, i_k)$
3	min	$eval_{\min}(t, X) = \min_{1 \leq j \leq k} \{\mathcal{D}(t, i_j)\}$
4	max	$eval_{\max}(t, X) = \max_{1 \leq j \leq k} \{\mathcal{D}(t, i_j)\}$
5	range	$eval_{\text{range}}(t, X) = eval_{\max}(t, X) - eval_{\min}(t, X)$
6	ETI	$eval_{eti,\epsilon}(t, X) = \frac{\sum_{i \in X} \mathcal{D}(t,i)}{\lvert X \rvert} \geq 1 - \epsilon$

Table 3: ***norm*** **functions.** M **is the length of the vector,** k **is a parameter, and** $\mathbf{w}$ **is a vector of weights.**

	norm function	Definition
1	L_k	$\lvert\lvert\mathbf{v}\rvert\rvert_k = \sqrt[k]{\sum_{t=1}^{M} \lvert\mathbf{v}(t)\rvert^k}$
2	L_1	$\lvert\lvert\mathbf{v}\rvert\rvert_1 = \sum_{t=1}^{M} \lvert\mathbf{v}(t)\rvert$
3	L_2	$\lvert\lvert\mathbf{v}\rvert\rvert_2 = \sqrt{\sum_{t=1}^{M} \lvert\mathbf{v}(t)\rvert^2}$
3	L_2^2	$\lvert\lvert\mathbf{v}\rvert\rvert_2^2 = \sum_{t=1}^{M} \lvert\mathbf{v}(t)\rvert^2$
4	weighted sum	$norm_w(\mathbf{v}, \mathbf{w}) = \sum_{t=1}^{M} \mathbf{w}(t)\mathbf{v}(t)$
5	sum ($norm_{\sum}$)	$norm_{\sum}(\mathbf{v}) = \sum_{t=1}^{M} \mathbf{v}(t)$
6	avg ($norm_{avg}$)	$norm_{avg}(\mathbf{v}) = \frac{1}{M} \sum_{t=1}^{M} \mathbf{v}(t)$
7	weighted avg ($norm_{wavg}$)	$norm_{wavg}(\mathbf{v}, \mathbf{w}) = norm_w(\mathbf{v}, \mathbf{w})$, where $\sum_{t=1}^{M} \mathbf{w}(t) = 1$
8	weighted L_k	$\lvert\lvert\mathbf{v}\rvert\rvert_{k,\mathbf{w}} = \sqrt[k]{\sum_{t=1}^{M} \mathbf{w}(t)\lvert\mathbf{v}(t)\rvert^k}$

3.1.2 Summarization Functions

It is useful to summarize the pattern evaluation vector $\mathbf{v}$ by a single number, e.g., by using a vector norm [4]. The most common vector norm is the L_k norm which is defined in Table 3, along with two of its most useful specific versions, the L_1 and L_2 norms. We also use the squared L_2 norm, L_2^2, which is the sum of the squares of the components of $\mathbf{v}$. We use the notation L_k or $norm_{L_k}$ to refer to these functions.

We can also consider *norm* functions which are weighted sums, where the weights are associated with objects. We identify the following special cases: the weights sum to 1 (the weighted average norm, $norm_{wavg}$); the weights are equal and sum to 1 (the average norm, $norm_{avg}$); (the weights are all 1 (the sum norm, $norm_{\sum}$). It is also possible to define the weighted L_k norm. For completeness, these norm functions are also shown in Table 3, but for simplicity, we restrict our discussion to the L_1, L_2, and L_2^2 norms.

3.1.3 Generalized Support Functions

The support of a pattern among a set of attributes X is a function, $\sigma(X)$, that is the composition of a pattern evaluation function, *eval*, and a summarization function, *norm*, which summarizes these evaluations with a single number.

$$\sigma(X) = (norm \circ eval)(X) = norm(eval(X)) \quad (3)$$

Given a support function, the goal is to use it to find sets of attributes that meet some support criterion. If, our support function has the anti-monotone property, as is typically the case, then we proceed by setting a minimum support threshold *minsup* and using an algorithm such as Apriori. The result is a collection of *strong pattern sets*,[1] i.e., a collection of sets of attributes that have support greater than *minsup*.

3.2 Example: Standard Support

We present different choices of *eval* and *norm* that reproduce the standard definition of support for binary data. Consider the following three support functions from Table 2: the logical *and* of the attribute values, $eval_\wedge$, the product of the values, $eval_\prod$, and the minimum of the values, $eval_{\min}$, and let $X = \{i_1, i_2, \cdots, i_k\}$ be an itemset (set of binary attributes). For a specific binary transaction, any of these pattern evaluation functions will produce a 1 exactly when all the attributes of X have attribute values of 1; if any attribute value is 0, then these functions return a 0.

If we use any of these three *eval* functions to produce the pattern evaluation vector, $\mathbf{v}$, then the L_1, and L_2^2 norms—see Table 3—will yield a value that is the count of the number of transactions that have all the items of X.

We adopt the following notation to refer to the different types of support functions that we have created:

$$\sigma_{eval,\ norm} = norm \ \circ \ eval \quad (4)$$

For example, the support function that is based on the $eval_{\min}$ and $norm_{\mathrm{L}_2^2}$ functions is written as follows:

$$\sigma_{\min,\ \mathrm{L}_2^2} = norm_{\mathrm{L}_2^2} \ \circ \ eval_{\min} = ||eval_{\min}||_2^2 \quad (5)$$

3.3 Example: Boolean Support Functions

A Boolean support function, $\sigma_{b,\ \mathrm{L}_1}$, is any support function that uses a Boolean pattern evaluation function $eval_b$ and the L_1 norm. (A Boolean pattern evaluation function returns either 0 or 1.) An example of a Boolean support function is the traditional support of an itemset X, which is equivalent to measuring the size of the set of transactions for which a conjunction of of the items (binary attributes) in X is true. This approach can be generalized—see for example [3, 10]—to more general Boolean formulas that use the logical connectives $\wedge$ (*and*), $\vee$ (*or*), and $\neg$ (*not*). Even more generally, we can consider a Boolean pattern evaluation function such as $eval_{\mathrm{range}<constant}$ [9], where the *eval* function is not a Boolean formula and where the data may not be binary. To illustrate, consider the data of Table 4. Set $constant = 3$ and let $X = \{term1, term2, term3\}$. Then the pattern evaluation vector is given by $\mathbf{v} = eval_{\mathrm{range}<3}(X) = (1, 0, 0, 0, 0, 1, 1, 0, 0, 0)$, and thus, $\sigma_{\mathrm{range}<constant,\ \mathrm{L}_1} = 3$, i.e., only 3 documents of Table 4 support the pattern. While $\sigma_{\mathrm{range}<constant,\ \mathrm{L}_1}$ has the anti-monotone property, in general, Boolean support functions may not be either monotone or anti-monotone.

[1]The name frequent itemset is not appropriate in the general case.

3.4 Example: Error Tolerant Itemsets

Error tolerant itemsets [13] relax the requirement that every transaction supporting the itemset must contain every item. Instead, it is enough that each transaction contain most of the items in the specified itemset. The definition of a strong ETI given below is taken from [13], but modified to make the notation and terminology consistent with that of this paper. For example, we might specify a strong ETI by requiring that each supporting transaction have at least 4 of the 5 specified items ($\epsilon = 0.2$), and that at least 2% of the transactions ($\kappa = 0.02$) support the strong ETI.

DEFINITION 1. ***Strong Error Tolerant Itemset*** *A strong ETI consists of a set of items $X \subseteq \mathcal{I}$, such that there exists a subset of transactions $R \subseteq \mathcal{T}$ consisting of at least $\kappa * M$ transactions and, for each $t \in R$, the fraction of items in X which are present in t is at least $1 - \epsilon$. M is the number of transactions, κ is the minimum support expressed as fraction of M, and ϵ is the fraction of items that can be missing in a transaction.*

Given a parameter ϵ, we can define a Boolean evaluation function $eval_{eti,\epsilon}$ to detect a strong ETI pattern:

$$eval_{eti,\epsilon}(t, X) = \frac{\sum_{i \in X} \mathcal{D}(t, i)}{|X|} \geq 1 - \epsilon \quad (6)$$

This *eval* function, together with the L_1 norm, can be used to define a support function for strong ETIs.

$$\begin{aligned} \sigma_{(eti,\epsilon),\ \mathrm{L}_1}(X) &= (norm_{\mathrm{L}_1} \ \circ \ eval_{eti,\epsilon})(X) \quad (7) \\ &= ||eval_{eti,\epsilon}(X)||_1 \quad (8) \end{aligned}$$

4. SUPPORT FOR CONTINUOUS DATA

The traditional approach to dealing with continuous data in association analysis is to convert each continuous attribute into a set of binary attributes. This is typically a two step process. First the continuous attribute is discretized, i.e., we find a set of thresholds that can be used to convert the attribute into a categorical variable. Then, each value of the categorical variable is mapped to a binary variable. However, converting continuous data to binary transaction data loses information with respect to both the magnitude of the data and the ordering between values. The motivation for considering continuous support measures is to allow association analysis for continuous data without such a loss of information.

4.1 Example: Min-Apriori

We begin our investigation of continuous support measures with an example based on the Min-Apriori algorithm [5] and the data of Table 4. Min-Apriori corresponds to the use of the support function $\sigma_{\min,\ \mathrm{L}_1}$. However, Min-Apriori first normalizes the data in each column by dividing each column entry by the sum of the column entries. The normalized data is shown in Table 5. One reason for using normalization is to make sure that the resulting support value is a number between 0 and 1. Another, perhaps more important reason is to ensure that all data is on the same scale so that sets of items that vary in the same way have similar support values. For example, suppose we have three items i_1, i_2, and i_3, and that $i_2 = 2i_3$, while $i_3 = 3i_1$. Without normalization, $\sigma_{\min,\ \mathrm{L}_1}(\{i_1, i_2\})$ is not equal to $\sigma_{\min,\ \mathrm{L}_1}(\{i_2, i_3\})$. Thus, normalization is often desirable in many domains, e.g., text documents.

However, a side-effect of normalization is that individual items can no longer be pruned using a support threshold since all items have a support of 1. In Section 6.3, we discuss normalization in the context of the hyperclique pattern.

The computation of the support of the set of attributes $X = \{term3, term4\}$ is shown in Table 6, where the first two columns show the normalized values for $term3$ and $term4$, while the third column shows the minimum of these two values for each row (object), i.e., column 3 is the pattern evaluation vector $v = eval_{\min}(\{term3, term4\})$. The support of $\{term3, term4\}$ is computed by taking the sum of column 3. Notice that $term3$ and $term4$ have individual supports of 1, as do all individual terms after normalization. The support of $\{term3, term4\}$ is 0.33, which indicates a moderate relationship. By contrast, the support of $\{term2, term4\}$ is 0 since these two terms do not co-occur in any document.

An alternative would be to convert the original data to a binary matrix[2] and then compute support. If we express support as a fraction, this yields a support of 0.1 for $\{term3, term4\}$. The reason for the discrepancy between the two versions of support is that these two terms do not co-occur much, but both have about a third of their weight in the last document. As an example of a case, where both versions of support are close, the traditional support for $\{term1, term2\}$ is 0.5, which is similar to the value of 0.53 computed using normalized data and $\sigma_{\min,\ L_1}$.

Table 4: Table of document-term frequencies.

	*term*1	*term*2	*term*3	*term*4	*term*5	*term*6
doc1	9	8	8	0	0	0
doc2	5	0	0	1	13	10
doc3	8	3	0	0	1	4
doc4	4	0	0	0	4	10
doc5	0	9	0	0	5	10
doc6	7	5	0	0	11	0
doc7	11	11	12	0	0	0
doc8	9	1	0	0	0	9
doc9	9	0	0	10	0	0
doc10	4	0	10	7	0	0

Table 5: Table of document-term frequencies normalized to have an L_1 norm of 1.

	*term*1	*term*2	*term*3	*term*4	*term*5	*term*6
doc1	0.14	0.22	0.27	0.00	0.00	0.00
doc2	0.08	0.00	0.00	0.06	0.38	0.23
doc3	0.12	0.08	0.00	0.00	0.03	0.09
doc4	0.06	0.00	0.00	0.00	0.12	0.23
doc5	0.00	0.24	0.00	0.00	0.15	0.23
doc6	0.11	0.14	0.00	0.00	0.32	0.00
doc7	0.17	0.30	0.40	0.00	0.00	0.00
doc8	0.14	0.03	0.00	0.00	0.00	0.21
doc9	0.14	0.00	0.00	0.56	0.00	0.00
doc10	0.06	0.00	0.33	0.39	0.00	0.00

Table 6: Computation of support for the set of attributes containing *term*3 and *term*4.

Document/Term	*term*3	*term*4	min($term3, term4$)
doc1	0.27	0.00	0.00
doc2	0.00	0.06	0.00
doc3	0.00	0.00	0.00
doc4	0.00	0.00	0.00
doc5	0.00	0.00	0.00
doc6	0.00	0.00	0.00
doc7	0.40	0.00	0.00
doc8	0.00	0.00	0.00
doc9	0.00	0.56	0.00
doc10	0.33	0.39	0.33
Support	1.00	1.00	**0.33**

[2]We convert entries to a 1 only if they are greater than 0.

4.2 Preserving the Anti-Monotone Property of Support Measures for Continuous Data

The situation with respect to the anti-monotone property of support depends on the *norm* and *eval* functions, as well as the data. We start by defining the concept of an anti-monotone *eval* function and the conditions under which selected *norm* functions are monotonic. We then prove a general theorem that relates the anti-monotone property of an *eval* function and the monotonicity of a *norm* function to the anti-monotone nature of a support function based on them. (This is important, of course, because an anti-monotone support function can yield efficient algorithms for discovering support based patterns.) Using these results and the anti-monotone property of $eval_{\min}$ and $eval_{\prod}$, we can then show that support functions based on $eval_{\prod}$ or $eval_{\min}$ and the L_k and L_2^2 norms, also have the anti-monotone property for continuous data. We will also use this result later.

Simply put, an *eval* function is anti-monotone if its values is guaranteed to be non-increasing as the number of items increases. More formally, we have the following definition:

Property 4.1. ***Anti-monotone Property for Pattern Evaluation Functions***
A pattern evaluation function, eval, is anti-monotone if, for any two sets of attributes X and Y where $X \subseteq Y$, $eval(t, Y) \leq eval(t, X), \forall t \subseteq \mathcal{T}$.

Before proving the main theorem of this section, we need a lemma about *norm* functions.

Lemma 4.1. *For any two vectors $\mathbf{u}$ and $\mathbf{v}$ of length M, if $|\mathbf{u}(t)| \leq |\mathbf{v}(t)|, \forall t\ 1 \leq t \leq M$, then $norm(\mathbf{u}) \leq norm(\mathbf{v})$ for the L_k and L_2^2 norms.*

Proof. The L_k and L_2^2 norms (and their weighted versions with non-negative weights) are monotonic functions of the absolute values of the components of $\mathbf{u}$ and $\mathbf{v}$. $\square$

The following key theorem connects the anti-monotone property of an *eval* function with the anti-monotone property of a support function based on it.

Theorem 4.1. *Let eval be an anti-monotone, non-negative pattern evaluation function. Then the support functions, $\sigma_{eval,\ L_k}$ and $\sigma_{eval,\ L_2^2}$, have the anti-monotone property.*

Proof. We assume that X and Y are sets of attributes, $X = \{i_1, \ldots, i_k\}$ and $Y = X \cup \{i_{k+1}\}$, where $i_{k+1} \notin X$. Let $eval(X) = \mathbf{v}_X$ and $eval(Y) = \mathbf{v}_Y$. Since *eval* is anti-monotone, $\mathbf{v}_Y(t) \leq \mathbf{v}_X(t)$. Because *eval* is non-negative, $\mathbf{v}_X$ and $\mathbf{v}_Y$ are as well, and Lemma 4.1 can then be applied to yield $norm(\mathbf{v}_Y) \leq norm(\mathbf{v}_X)$ for the L_k and L_2^2 norms. Therefore, $\sigma_{eval,\ L_k}$ and $\sigma_{eval,\ L_2^2}$ have the anti-monotone property for non-negative data. $\square$

The *eval* functions, $eval_{\min}$ and $eval_{\prod}$, have the anti-monotone property, i.e., $eval_{\min}$ is anti-monotone for non-negative data and $eval_{\prod}$ is anti-monotone for non-negative data between 0 and 1. (These proofs are straightforward and are omitted to save space.) Thus, we can prove the following two theorems about the anti-monotone property of support functions based on these two *eval* functions.

Theorem 4.2. *For non-negative data, support functions, $\sigma_{\min,\ L_k}$ and $\sigma_{\min,\ L_2^2}$, have the anti-monotone property.*

Proof. This follows directly from the anti-monotone property of $eval_{\min}$ and Theorem 4.1. $\square$

Theorem 4.3. *For non-negative data between 0 and 1, i.e.,* $0 \leq \mathcal{D}(t,i) \leq 1, t \in \mathcal{T}, i \in \mathcal{I}$, *the support functions,* $\sigma_{\prod,\ \mathrm{L}_k}$ *and* $\sigma_{\prod,\ \mathrm{L}_2^2}$, *have the anti-monotone property.*

Proof. This follows directly from the anti-monotone property of $eval_{\prod}$ and Theorem 4.1 $\square$

5. THE HYPERCLIQUE PATTERN

A hyperclique pattern [12] is a frequent itemset with the additional requirement that every item in the itemset implies the presence of the remaining items with a minimum level of confidence known as the h-confidence (or all-confidence [8]). More formally we have the following definition:

Definition 2. ***Hyperclique*** *A set of attributes,* $X \subseteq \mathcal{I}$, *forms a hyperclique with a particular level of h-confidence, where h-confidence is defined as*

$$\begin{aligned} \mathrm{hconf}(X) &= \min_{i \in X}\{\mathrm{conf}(\{i\} \rightarrow \{X - \{i\}\})\} \quad (9) \\ &= \sigma(X)/\max_{i \in X}\{\sigma(i)\} \quad (10) \end{aligned}$$

5.1 Properties of h-confidence

The following properites of h-confidence are proved in [12]. h-confidence is in the interval $[0,1]$ and has the anti-monotone property. The cross support property, which is useful for efficiently finding hypercliques, states that the only possible attributes that can be in a hyperclique with an attribute i for a given level of h-confidence $\mathrm{h_c}$ are those attributes whose support falls in the interval $[\mathrm{h_c} * \sigma(i), \sigma(i)/\mathrm{h_c}]$. This feature of the hyperclique pattern implies that attributes that are too different in terms of their support cannot belong to the same hyperclique pattern. Finally, hypercliques also have the high affinity property, i.e., items in a hyperclique with a high h-confidence have a high pairwise similarity.

5.2 H-Confidence As Support

In Section 6, we will show that we can extend the hyperclique pattern to continuous data. However, even before that, we can show an important relationship between hypercliques in binary transaction data and the support function $\sigma_{\min,\ \mathrm{L}_2^2}(X)$. In particular, since $\sigma_{\min,\ \mathrm{L}_2^2}(X)$ is equivalent to standard support for binary data, we can substitute $\sigma_{\min,\ \mathrm{L}_2^2}(X)$ for the standard support function $\sigma(X)$ in Equation 10. If we normalize all attributes to have an L_2 norm of 1, then $\sigma_{\min,\ \mathrm{L}_2^2}(\{i\}) = 1$ for all items i, and by Equation 10, $\mathrm{hconf}(X) = \sigma_{\min,\ \mathrm{L}_2^2}(X)$. Hence, our support framework provides a simple interpretation of h-confidence as support for normalized data.

To illustrate this point, we provide an example. In tables 7 and table 8 we show, respectively, some sample data and the same data after it has been normalized to have an L_2 norm of 1. Let X be the itemset consisting of all five items. Then, from Table 7, we see that the (standard) support of X is 3, while the maximum support of any item is 5. Thus, the h-confidence of X is $3/5 = 0.6$. Using Table 8, we can compute $\sigma_{\min,\ \mathrm{L}_2^2}(X)$ by taking the min of each row, squaring it, and then summing, i.e., $\sigma_{\min,\ \mathrm{L}_2^2}(X) = 3 * (0.447)^2 = 0.6$.

6. EXTENDING THE HYPERCLIQUE PATTERN TO CONTINUOUS ATTRIBUTES

In this section, we extend the hyperclique pattern to continuous data by using the $\sigma_{\min,\ \mathrm{L}_2^2}$ support function. It is straightforward to show that all the properties of h-confidence

Table 7: Example to illustrate h-confidence as support—original data.

Transaction/Item	1	2	3	4	5
1	1	1	1	1	1
2	1	1	1	1	1
3	1	1	1	1	1
4	1	0	1	1	1
5	1	0	0	0	1

Table 8: Example to illustrate h-confidence as support—normalized data.

Trans/Item	1	2	3	4	5	min
1	0.447	0.577	0.500	0.500	0.447	0.447
2	0.447	0.577	0.500	0.500	0.447	0.447
3	0.447	0.577	0.500	0.500	0.447	0.447
4	0.447	0	0.500	0.500	0.447	0
5	0.447	0	0	0	0.447	0

for binary data that were discussed in section 5.1, also hold for continuous data. However, because of space limitations, we only prove results for the high-affinity property of hypercliques with normalized data. Further details are in [11].

6.1 The High-Affinity Property

As with binary data, the high-affinity property for hypercliques with continuous data guarantees that the attributes are pairwise similar to one another at some minimum level. Specifically, a lower bound for the minimum pairwise cosine similarity is given by the h-confidence of the hyperclique. We formally prove this in Theorem 6.1.

In the following, $\mathbf{i}$ and $\mathbf{j}$ are attributes i and j interpreted as vectors and they have an L_2 norm of 1.

Theorem 6.1. ***Cosine high-affinity property.*** *Assume that the data is non-negative and all attributes have an* L_2 *norm of 1. Let* X *be a set of attributes with an h-confidence of* $\mathrm{h_c}$. *Then, for any two attributes of* X, i *and* j, $cos(i,j) \geq \mathrm{h_c}$, *where* $cos(i,j)$ *is the cosine similarity between* i *and* j.

Proof.

$$\begin{aligned} cos(i,j) &= \mathbf{i} \bullet \mathbf{j} \\ &\geq ||\mathbf{v}||_2^2, \text{ where } \mathbf{v} = eval_{\min}(X) \\ &= \sigma_{\min,\ \mathrm{L}_2^2}(X) \\ &= \mathrm{hconf}(X) \\ &= \mathrm{h_c} \end{aligned}$$

Line 2 follows from line 1 because $\mathbf{i}$ and $\mathbf{j}$ are elementwise $\geq \mathbf{v}$ for $i \in X$ or $j \in X$. Line 3 follows from the definition of $\sigma_{\min,\ \mathrm{L}_2^2}$. Line 4 follows from line 3 because $\mathrm{hconf}(X) = \sigma_{\min,\ \mathrm{L}_2^2}(X)$ when attributes have an L_2 norm of 1. $\square$

6.2 An Example

To illustrate the high-affinity property for continuous hypercliques, we use an example based on the data of Table 4. The computation of the support of the set of attributes $X = \{term1, term2, term3\}$ is shown in Table 9, where the first three columns are normalized versions of $term1$, $term2$, and $term3$ from Table 4. (Here, we use the L_2 norm, not the L_1 norm as in the Min-Apriori example.) The fourth column shows the minimum of these three attributes for a particular row (object). The support of the three terms is computed by taking the sum of squares of column 4, and that value, 0.38, is also the h-confidence. This is indeed a lower bound for the pairwise cosine similarity, since the lowest pairwise similarity of these items is 0.6.

Table 9: Computation of support for the set of attributes containing $term1$, $term2$, and $term3$.

Doc/Term	$term1$	$term2$	$term3$	$\min(1,2,3)$
1	0.39	0.46	0.46	0.39
2	0.22	0	0	0
3	0.35	0.17	0	0
4	0.17	0	0	0
5	0	0.52	0	0
6	0.30	0.29	0	0
7	0.48	0.63	0.68	0.48
8	0.39	0.06	0	0
9	0.39	0	0	0
10	0.17	0	0.57	0
Support	1.0	1.0	1.0	**0.38**

6.3 Normalization

Normalization is not required for extending the hyperclique pattern to continuous data—see [11]. However, as with Min-Apriori, normalization adjusts for attributes with different measurement scales and produces a support value that is between 0 and 1. On the negative side, after normalization, all single items have a support of 1 and thus, cannot be pruned by using a support threshold or the cross support property.

To more fully understand the pluses and minuses of normalization, we consider two additional facts. First, continuous attributes can have widely different support and still be very similar to one another. This is not true for binary attributes.[3] Second, for continuous hypercliques, the cross support property still dictates that two attributes with widely different levels of support cannot be together in a hyperclique with high h-confidence—see [11]. Thus, continuous attributes, which are highly similar, but which have widely different support, can only appear in hypercliques with low h-confidence. However, many attributes in such low h-confidence hypercliques will not be very similar to one another.

To summarize, without normalization, we can effectively find continuous hypercliques with highly similar attributes only if they have similar support. This is exactly the same as with binary attributes. However, to effectively find highly similar continuous attributes with widely differing support, normalization is necessary.

7. RELATED WORK

To save space, we limit our discussion of prior work to that already present in the body of the paper and refer the reader to our technical report [11] for more details.

8. CONCLUSIONS AND FUTURE WORK

We have described a framework for generalizing the notion of support and have shown that this framework can be used to express support for several existing association patterns: frequent itemsets, general Boolean patterns, and error tolerant itemsets. We also showed how this framework can be used to extend binary association patterns, e.g., the hyperclique pattern, to continuous data.

There are many possibilities for future work. On the practical side, we plan to explore applications of the continuous hyperclique pattern. On the theoretical side, we plan to investigate new types of support for non-binary data and non-traditional association patterns, and to explore how confidence should be extended for non-standard support measures. Preliminary work in both areas is presented in [11]. Finally, a key benefit of a framework is that it allows researchers to more easily express, explore, and communicate ideas. We hope that our framework will prove useful and will motivate additional research in this area.

[3]It is straightforward to show that for two binary attributes i and j, with $\sigma(\{i\}) \leq \sigma(\{j\})$, that $cos(i,j) \leq \sqrt{\sigma(\{i\})/\sigma(\{j\})}$, where σ is standard support.

9. ACKNOWLEDGMENTS

This work was partially supported by NASA grant # NCC 2 1231, by DOE/LLNL grant W-7045-ENG-48, by NSF grant IIS-0308264, and by the Army High Performance Computing Research Center under the auspices of the Department of the Army, Army Research Laboratory cooperative agreement number DAAD19-01-2-0014. The content of this work does not necessarily reflect the position or policy of the government and no official endorsement should be inferred. Access to computing facilities was provided by the AHPCRC and Minnesota Supercomputing Institute.

10. REFERENCES

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *VLDB 94*, pages 487–499, 1994.

[2] R. Agrawal and R. Srikant. Mining sequential patterns. In *ICDE 95*, pages 3–14, 1995.

[3] P. Bollmann-Sdorra, A. Hafez, and V. V. Raghavan. A theoretical framework for association mining based on the boolean retrieval model. In *DaWaK 2001, Munich, Germany*, pages 21–30, 2001.

[4] J. W. Demmel. *Applied Numerical Linear Algebra*. SIAM, January 1997.

[5] E.-H. Han, G. Karypis, and V. Kumar. Tr# 97-068: Min-apriori: An algorithm for finding association rules in data with continuous attributes. Technical report, Department of Computer Science, University of Minnesota, Minneapolis, MN, 1997.

[6] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, 2000.

[7] J. Hipp, U. Güntzer, and G. Nakhaeizadeh. Algorithms for association rule mining – a general survey and comparison. *SIGKDD Explorations*, 2(1):58–64, July 2000.

[8] E. R. Omiecinski. Alternative interest measures for mining associations in databases. *IEEE TKDE*, 15(1):57–69, January/February 2003.

[9] J. Pei and J. Han. Can we push more constraints into frequent pattern mining? In *KDD 2000*, pages 350–354, 2000.

[10] R. Srikant, Q. Vu, and R. Agrawal. Mining association rules with item constraints. In *KDD 97*, pages 67–73, 1997.

[11] M. Steinbach, P.-N. Tan, H. Xiong, and V. Kumar. Tr# 2004-114: : Extending the notion of support. Technical report, Army High Performance Computing Research Center, April 2004.

[12] H. Xiong, P. Tan, and V. Kumar. Mining strong affinity association patterns in data sets with skewed support distribution. In *ICDM 2003*, pages 387–394, 2003.

[13] C. Yang, U. M. Fayyad, and P. S. Bradley. Efficient discovery of error-tolerant frequent itemsets in high dimensions. In *KDD 2001*, pages 194–203, 2001.

[14] M. J. Zaki and M. Ogihara. Theoretical foundations of association rules. In *DMKD 98*, pages 7:1–7:8, 1998.