Getting trivial error bounds:

Assume that all negative borders are at u.

This gives the maximum error from the error theorem.

Example:

Telecom Alarm Database:

3.3 million windows of an alarm sequence. Locate frequent sets with support 0.0001

There are 128 frequent sets, 713 sets on the negative border. Only 5 have more than 2 elements.

Generate 100 random disjunctions of size 4. Each disjunction is a frequent alarm.

The maximum error bound for the frequency of the disjunction was:

0.00041

In 53 cases, it was < 0.0001 and in 28 cases, it was < 0.00005

On an average, error bound is 6.2% of the estimated size of disjunctions.

Example:

Course Enrollment database:

|Bd-(r, 0.05)| = 336

From the error theorem, we cand erive the following:

For the set W of courses, Data structures and Database systems I, all three terms are known:
o'(W) = o(W) = 0.25, e(W) = 0

For the set W of three C or Unix related courses, 4 of 7 terms are known and 2 are on negative border. o'(W) = 0.24, e(w) <= 0.017

For a set W of three different intermediate courses TOC, AI, and CG, we have 3 of 7 terms, 3 are in Bd-. o'(W) = 0.21, e(w) <= 0.079

For W consisting of 6 programming projects, o'(W) = 0.31, e(W) <= 0.012
7 of 63 terms are known, 3 are in Br-.

Example:

Consider the previous example:

To compute o({a, b, c, b}), the theorem states that the bound is

a({a, b, c}) + a({d})

Note: Algorithms that compute collections of frequent sets also compute frequencies of the negative border.

Example:

R = {a, b, c, d}

Fr(r,u) = {{a,b}, {b,c}, {a,c}, {a}, {b}, {c}, phi}

Bd-(r,u) = {{a,b,c}, {d}}

Theorem: For all W in R, denote K = {Y in Bd-{r} | Y is in W}

if K is empty, then e(W) = 0

else, let

c = max{$2^{|K| - 2}$ / |K|, 1}

Then

e(W) <= c x sum over all Y in W and Y in Bd-(r) (a(Y)).

Error in Approximation

If we compute the sum of the truncated series, we can get an approximation of o(X), o'(X).

The error in this approximation is given by

$e(W) = |o(W) - o'(W)|$

There are many studies that show that knowing the terms

$(-1)^{|X| + 1} a(X)$ with $|X| <= \text{sqrt}(|W|)$ is sufficient.

However, this rule does not apply for us.

Given r over R and threshold u, the negative border BD-(r,u) of Fr(r,u) consists of sets W in R and not in Fr(r,u) such that all proper subsets of W are in Fr(r,u).

Negative border defines which sets are frequent and which are not.

Approximate inclusion - Exclusion

Consider the following approximation:

Given X in R, we want to know how many tuples have 1 in atleast one of the columns of W; i.e. o(W).

This is trivial to do by reading the transactions and keeping count.

Can we do better?

The inclusion-exclusion formula says that

q(w) = sum over all non-empty subsets X of W ($(-1)^{|X| + 1}$ a(x))

However, we do not know all the terms of this sequence. What we know is the truncated sum (i.e. only those a(X)'s that are frequent).

Example:

To find the confidence of x => y OR z

we need to know

a(x), a(xUy) + a(xUz) - a(xUyUz)

What if one of these quantities is under the threshold?

To get bounds on them if a(xUy) or a(xUz) to zero.

Similarly, if a(xUyUz) is unknown, set it to the threshold u.

This will give you bounds on confidence!

Some more notation:

Given a relation r with attributes R, a threshold u in [0, 1], the u-frequent sets of r are subsets X of R such that a(X, r) >= u

The collection is denoted by

Fr(r,u) or simply Fr(r).

The complexity of this is bounded by O(|Fr(r,u)| . |r|)

The rule confidences can be bound by frequent sets.

The idea:

To get a lower bound on the confidence, replace every term a(W) such that W is not in Fr(r) in the formula by either 0 or u depending on whether it is in the numerator or denominator and whether it is being added or subtracted.

Example:

Database of Student names and courses.

Total courses = 127

First determine all subsets with support >= 0.05

No. of frequent sets = 489

Now compute association rules, rules with disjunctions on left or right and rules with negations:

We find that

X => C programming in UNIX OR C programming

has much higher confidence than

X => C programming

and

X => C programming in UNIX.

Examples (continued)

Similarly, for the rule x OR y => z

confidence = ((a(xUz) + a(yUz) - a(xUyUz)) / (a(x) + a(y) - a(xUy))

x AND NOT y => z

confidence = (a(xUz) - a(xUyUz)) / (z(x) - a(xUy))

x => y AND NOT z

confidence = (a(xUy) - a(xUyUz)) / a(x)

Proposition:

For all boolean formulae p over R, there exist sets W1, W2, ... Wm in R and coefficients e1, e2, ... em in {-1, +1} such that for all relations r,

s (q, r) = sum from i = 1 to m (ei * a(wi,r))

Example

ASsociation has the form x => y

x and y in R.

confidence = a(xUy) / a(x)

A disjunctive rule has the form x => y OR z

confidence = (a(xUy) + a(xUz) - a(xUyUz)) / a(x)

General Rules and Frequent Sets:

Given a set R of 0/1 valued attributes, a boolean formula over R is a formula built from the atomic formulae a = 0 and a = 1 (a in R) using AND, OR, NOT.

Given a 0/1 relation r with attributes R, the frequency s(p,r) of p in r is

| {t in r such that p is true of t}| / |r|

i.e. the fraction of rows where p holds.

Notation:

for X = {a1, a2, ..., ak},

a(X,r) = s(a1 = 1 and a2 = 1 and ... and ak = 1}

o(X,r) = s(a1 = 1 or a2 = 1 or ... or ak = 1}

A boolean rule over R is of the form p => q where p and q are boolean formulae.

cofidence of the rule is s (p and q) / s (p)

Using Frequent Sets:

Reference:

Heikki Mannila and Hannu Toivonen: Multiple uses of frequent sets and condensed representations. In Second International Conference on Knowledge Discovery and Data Mining, Portland, Oregon, August 2-4, 1996.

http://www.cs.helsinki.fi/~mannila/data-mining-publications.html

Associations proceed by forming frequent sets.

From the frequent sets, we compute rules and confidence factors.

What else can we do with frequent sets?

Collection of frequent sets can serve as a condensed representation of input data. They can be used for more sophisticated rules than implications (eg. disjunctions and negations).