Mining Frequent Patterns, Associations, and Correlations
Frequent Pattern

- **Frequent Pattern**: a pattern (a set of items, subsequences, substructures, etc.) that occurs frequently in a data set.

- For example, a set of items, such as milk and bread, that appear frequently together in a transaction data set is a *frequent itemset*.

- A subsequence, such as buying first a PC, then a digital camera, and then a memory card, if it occurs frequently in a shopping history database, is a *(frequent) sequential pattern*.

- A substructure can refer to different structural forms, such as subgraphs, subtrees, or sublattices, which may be combined with itemsets or subsequences. If a substructure occurs frequently, it is called a *(frequent) structured pattern*.
What Is Frequent Pattern Analysis?

- First proposed by Agrawal, Imielinski, and Swami [AIS93] in the context of frequent itemsets and association rule mining.
- Motivation: Finding inherent regularities in data
  - What products were often purchased together?—Beer and diapers?!
  - What are the subsequent purchases after buying a PC?
  - What kinds of DNA are sensitive to this new drug?
  - Can we automatically classify web documents?
- Applications
  - Basket data analysis, cross-marketing, catalog design, sale campaign analysis, Web log (click stream) analysis, and DNA sequence analysis.
Market Basket Analysis

- An example of frequent itemset mining is **market basket analysis**.
  - This process analyzes customer buying habits by finding associations between the different items that customers place in their “shopping baskets”.

- If we think of the universe as the set of items available at the store, then each item has a Boolean variable representing the presence or absence of that item.
- Each basket can then be represented by a Boolean vector of values assigned to these variables.
- The Boolean vectors can be analyzed for buying patterns that reflect items that are frequently associated or purchased together.
- These patterns can be represented in the form of association rules.
Basic Concepts: Frequent Patterns

- **itemset**: A set of one or more items
- **k-itemset** \( X = \{x_1, \ldots, x_k \} \)
- **(absolute) support**, or, **support count of** \( X \): Frequency or occurrence of an itemset \( X \)
- **(relative) support**, \( s \), is the fraction of transactions that contains \( X \) (i.e., the probability that a transaction contains \( X \))
- An itemset \( X \) is **frequent** if \( X \)’s support is no less than a **minsup** threshold.

<table>
<thead>
<tr>
<th>Tid</th>
<th>Items bought</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Beer, Nuts, Diaper</td>
</tr>
<tr>
<td>20</td>
<td>Beer, Coffee, Diaper</td>
</tr>
<tr>
<td>30</td>
<td>Beer, Diaper, Eggs</td>
</tr>
<tr>
<td>40</td>
<td>Nuts, Eggs, Milk</td>
</tr>
<tr>
<td>50</td>
<td>Nuts, Coffee, Diaper, Eggs, Milk</td>
</tr>
</tbody>
</table>

Customer buys both

Customer buys diaper

Customer buys beer
Basic Concepts: Association Rules

Association Rule

- An implication expression of the form $X \rightarrow Y$, where $X$ and $Y$ are itemsets

- Find all the rules $X \rightarrow Y$ with minimum support and confidence
  - **support**, probability that a transaction contains $X \cup Y$: $P(X \cup Y)$
    - Fraction of transactions that contain both $X$ and $Y$
  - **confidence**, conditional probability that a transaction having $X$ also contains $Y$: $P(Y/X) = \text{support}_\text{count}(X \cup Y) / \text{support}_\text{count}(X)$
    - Measures how often items in $Y$ appear in transactions that contain $X$

Let $\text{minsup} = 50\%$, $\text{minconf} = 50\%$

Frequent Patterns: Beer:3, Nuts:3, Diaper:4, Eggs:3, {Beer, Diaper}:3

- **Association rules**: (many more!)
  - { Beer } $\rightarrow$ { Diaper } (60%, 100%)
  - { Diaper } $\rightarrow$ { Beer } (60%, 75%)
Why Use Support and Confidence?

• **Support** is an important measure because a rule that has very low support may occur simply by chance.
  – A low support rule may be uninteresting from a business perspective because it may not be profitable to promote items that customers seldom buy together
  – For these reasons, support is often used to eliminate uninteresting rules

• **Confidence** measures the reliability of the inference made by a rule.
  – For a given rule $X \rightarrow Y$, the higher the confidence, the more likely it is for $Y$ to be present in transactions that contain $X$.

• Association analysis results should be interpreted with caution.
  – The inference made by an association rule does not necessarily imply causality.
  – Instead, it suggests a strong co-occurrence relationship between items in the antecedent and consequent of the rule.
Association Rule Mining Task

- Given a set of transactions $T$, the goal of association rule mining is to find all rules having
  - support $\geq \text{minsup}$ threshold
  - confidence $\geq \text{minconf}$ threshold

- Brute-force approach:
  - List all possible association rules
  - Compute the support and confidence for each rule
  - Prune rules that fail the $\text{minsup}$ and $\text{minconf}$ thresholds

$\Rightarrow$ Computationally not feasible!
Mining Association Rules

Example of Rules:

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bread, Milk</td>
</tr>
<tr>
<td>2</td>
<td>Bread, Diaper, Beer, Eggs</td>
</tr>
<tr>
<td>3</td>
<td>Milk, Diaper, Beer, Coke</td>
</tr>
<tr>
<td>4</td>
<td>Bread, Milk, Diaper, Beer</td>
</tr>
<tr>
<td>5</td>
<td>Bread, Milk, Diaper, Coke</td>
</tr>
</tbody>
</table>

- {Milk,Diaper} → {Beer} (s=0.4, c=0.67)
- {Milk,Beer} → {Diaper} (s=0.4, c=1.0)
- {Diaper,Beer} → {Milk} (s=0.4, c=0.67)
- {Beer} → {Milk,Diaper} (s=0.4, c=0.67)
- {Diaper} → {Milk,Beer} (s=0.4, c=0.5)
- {Milk} → {Diaper,Beer} (s=0.4, c=0.5)

Observations:

- All the above rules are binary partitions of the same itemset: {Milk, Diaper, Beer}
- Rules originating from the same itemset have identical support but can have different confidence
- Thus, we may decouple the support and confidence requirements
Association Rule Mining

• The problem of mining association rules can be reduced to that of mining frequent itemsets.

• In general, association rule mining can be viewed as a two-step process:
  1. **Find all frequent itemsets**: By definition, each of these itemsets will occur at least as frequently as a predetermined minimum support count, \( \text{minsup} \).
     
     – Generate all itemsets whose support \( \geq \text{minsup} \)
  2. **Generate strong association rules from the frequent itemsets**: By definition, these rules must satisfy minimum support and minimum confidence.
     
     – Generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of a frequent itemset

  – Frequent itemset generation is still computationally expensive
Frequent Itemset Generation

Given d items, there are $2^d$ possible candidate itemsets
Frequent Itemset Generation

- Brute-force approach:
  - Each itemset in the lattice is a candidate frequent itemset
  - Count the support of each candidate by scanning the database
    - Match each transaction against every candidate
    - Complexity $\sim O(NMw)$ $\Rightarrow$ Expensive since $M = 2^d$ !!!
Computational Complexity

- Given $d$ unique items:
  - Total number of itemsets $= 2^d$
  - Total number of possible association rules:

$$R = \sum_{k=1}^{d-1} \binom{d}{k} \times \sum_{j=1}^{d-k} \binom{d-k}{j}$$

$$= 3^d - 2^{d+1} + 1$$

If $d=6$, $R = 602$ rules
Frequent Itemset Generation Strategies

• Reduce the number of candidates (M)
  – Complete search: $M=2^d$
  – Use pruning techniques to reduce M

• Reduce the number of transactions (N)
  – Reduce size of N as the size of itemset increases
  – Used by DHP and vertical-based mining algorithms

• Reduce the number of comparisons (NM)
  – Use efficient data structures to store the candidates or transactions
  – No need to match every candidate against every transaction
Reducing Number of Candidates

• **Apriori principle:**
  – If an itemset is frequent, then all of its subsets must also be frequent

• Apriori principle holds due to the following property of the support measure:

\[ \forall X, Y : (X \subseteq Y) \implies s(X) \geq s(Y) \]

  – Support of an itemset never exceeds the support of its subsets
  – This is known as the **anti-monotone** property of support
Illustrating Apriori Principle

Found to be Infrequent

Pruned supersets
### Illustrating Apriori Principle

<table>
<thead>
<tr>
<th>Item</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bread</td>
<td>4</td>
</tr>
<tr>
<td>Coke</td>
<td>2</td>
</tr>
<tr>
<td>Milk</td>
<td>4</td>
</tr>
<tr>
<td>Beer</td>
<td>3</td>
</tr>
<tr>
<td>Diaper</td>
<td>4</td>
</tr>
<tr>
<td>Eggs</td>
<td>1</td>
</tr>
</tbody>
</table>

**Items (1-itemsets)**

**Pairs (2-itemsets)**

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>{Bread,Milk}</td>
<td>3</td>
</tr>
<tr>
<td>{Bread,Beer}</td>
<td>2</td>
</tr>
<tr>
<td>{Bread,Diaper}</td>
<td>3</td>
</tr>
<tr>
<td>{Milk,Beer}</td>
<td>2</td>
</tr>
<tr>
<td>{Milk,Diaper}</td>
<td>3</td>
</tr>
<tr>
<td>{Beer,Diaper}</td>
<td>3</td>
</tr>
</tbody>
</table>

(No need to generate candidates involving Coke or Eggs)

**Triplets (3-itemsets)**

<table>
<thead>
<tr>
<th>Itemset</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>{Bread,Milk,Diaper}</td>
<td>3</td>
</tr>
</tbody>
</table>

**Minimum Support = 3**

If every subset is considered, 
\[ ^6C_1 + ^6C_2 + ^6C_3 = 41 \]

With support-based pruning, 
\[ 6 + 6 + 1 = 13 \]
Closed Patterns and Max-Patterns

- A long pattern contains a combinatorial number of sub-patterns, e.g., 
  \{a_1, \ldots, a_{100}\} contains 
  \[
  \binom{100}{1} + \binom{100}{2} + \cdots + \binom{100}{100} = 2^{100} - 1 \approx 1.27 \times 10^{30}
  \] 
  sub-patterns

- Solution: Mine closed patterns and max-patterns instead

- An itemset X is closed if X is frequent and there exists no super-pattern Y ⊇ X, with the same support as X

- An itemset X is a max-pattern if X is frequent and there exists no frequent super-pattern Y ⊇ X

- Closed pattern is a lossless compression of frequent patterns
  - Reducing the # of patterns and rules
Closed Patterns and Max-Patterns

- Suppose a DB contains only two transactions
  - \(<a_1, \ldots, a_{100}>, <a_1, \ldots, a_{50}>\)
  - Let \(\text{min\_sup} = 1\)

- What is the set of \textit{closed itemset}? 
  - \(\{a_1, \ldots, a_{100}\}: 1\)
  - \(\{a_1, \ldots, a_{50}\}: 2\)

- What is the set of \textit{max-pattern}? 
  - \(\{a_1, \ldots, a_{100}\}: 1\)

- What is the set of all patterns? 
  - \(\{a_1\}: 2, \ldots, \{a_1, a_2\}: 2, \ldots, \{a_1, a_{51}\}: 1, \ldots, \{a_1, a_2, \ldots, a_{100}\}: 1\)
  - A big number: \(2^{100} - 1\)? Why?
Scalable Frequent Itemset Mining Methods

• The **downward closure** property of frequent patterns
  – **Any subset of a frequent itemset must be frequent**
  – If \{beer, diaper, nuts\} is frequent, so is \{beer, diaper\}
  – i.e., every transaction having \{beer, diaper, nuts\} also contains \{beer, diaper\}

• Scalable Frequent Itemset Mining Methods: Three major approaches
  – Apriori
  – Frequent pattern growth
  – Vertical data format approach
The Apriori Algorithm: Finding Frequent Itemsets Using Candidate Generation

- **Apriori pruning principle**: If there is any itemset which is infrequent, its superset should not be generated/tested!

- Method:
  - Initially, scan DB once to get frequent 1-itemset
  - Generate length (k+1) candidate itemsets from length k frequent itemsets
  - Test the candidates against DB
  - Terminate when no frequent or candidate set can be generated
### The Apriori Algorithm - An Example

**Database TDB**

<table>
<thead>
<tr>
<th>Tid</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>A, C, D</td>
</tr>
<tr>
<td>20</td>
<td>B, C, E</td>
</tr>
<tr>
<td>30</td>
<td>A, B, C, E</td>
</tr>
<tr>
<td>40</td>
<td>B, E</td>
</tr>
</tbody>
</table>

**1st scan**

- **$C_1$**
  - $\{A\}$: 2
  - $\{B\}$: 3
  - $\{C\}$: 3
  - $\{D\}$: 1
  - $\{E\}$: 3

**$L_1$**

- $\{A\}$: 2
- $\{B\}$: 3
- $\{C\}$: 3
- $\{E\}$: 3

**2nd scan**

- **$C_2$**
  - $\{A, B\}$: 1
  - $\{A, C\}$: 2
  - $\{A, E\}$: 1
  - $\{B, C\}$: 2
  - $\{B, E\}$: 3
  - $\{C, E\}$: 2

**$L_2$**

- $\{A, B\}$
- $\{A, C\}$
- $\{A, E\}$
- $\{B, C\}$
- $\{B, E\}$
- $\{C, E\}$

**3rd scan**

- **$C_3$**
  - $\{B, C, E\}$

**$L_3$**

- $\{B, C, E\}$: 2

**Sup_{min} = 2**

Data Mining
The Apriori Algorithm

Input:
- $D$, a database of transactions;
- $\text{min}\_\text{sup}$, the minimum support count threshold.

Output: $L$, frequent itemsets in $D$.

Method:

$$L_1 = \text{find\_frequent\_1\_itemsets}(D);$$

for $(k = 2; L_{k-1} \neq \emptyset; k++)$

$$C_k = \text{apriori\_gen}(L_{k-1});$$

for each transaction $t \in D$ { // scan $D$ for counts

$$C_t = \text{subset}(C_k, t);$$ // get the subsets of $t$ that are candidates

for each candidate $c \in C_t$

$$c\text{.count}++;$$

}

$$L_k = \{c \in C_k | c\text{.count} \geq \text{min}\_\text{sup}\}$$

} 

return $L = \bigcup_k L_k$;
The Apriori Algorithm

procedure apriori_gen($L_{k-1}$: frequent $(k-1)$-itemsets)
    for each itemset $l_1 \in L_{k-1}$
        for each itemset $l_2 \in L_{k-1}$
            if $(l_1[1] = l_2[1]) \land (l_1[2] = l_2[2]) \land \ldots \land (l_1[k-2] = l_2[k-2]) \land (l_1[k-1] < l_2[k-1])$ then {
                $c = l_1 \times l_2$; // join step: generate candidates
                if has_infrequent_subset($c$, $L_{k-1}$) then
                    delete $c$; // prune step: remove unfruitful candidate
                else add $c$ to $C_k$;
            }
    return $C_k$;

procedure has_infrequent_subset($c$: candidate $k$-itemset; $L_{k-1}$: frequent $(k-1)$-itemsets); // use prior knowledge
    for each $(k-1)$-subset $s$ of $c$
        if $s \notin L_{k-1}$ then
            return TRUE;
    return FALSE;
Implementation of Apriori

• How to generate candidates?
  – Step 1: self-joining $L_k$
  – Step 2: pruning

• Example of Candidate-generation
  – $L_3 = \{abc, abd, acd, ace, bcd\}$
  – Self-joining: $L_3 \times L_3$
    • $abcd$ from $abc$ and $abd$
    • $acde$ from $acd$ and $ace$
  – Pruning:
    • $acde$ is removed because $ade$ is not in $L_3$
  – $C_4 = \{abcd\}$
Generating Association Rules from Frequent Itemsets

Once the frequent itemsets from transactions in a database D have been found, it is straightforward to generate strong association rules from them (where strong association rules satisfy both minimum support and minimum confidence).

Association rules can be generated as follows:

- For each frequent itemset $l$, generate all nonempty subsets of $l$.
- For every nonempty subset $s$ of $l$, output the rule $s \Rightarrow (l - s)$ if $\frac{\text{support}_\text{count}(l)}{\text{support}_\text{count}(s)} \geq \text{min}_\text{conf}$, where $\text{min}_\text{conf}$ is the minimum confidence threshold.

Because the rules are generated from frequent itemsets, each one automatically satisfies minimum support. Frequent itemsets can be stored ahead of time in hash tables along with their counts so that they can be accessed quickly.
Reducing Number of Comparisons

- Candidate counting:
  - Scan the database of transactions to determine the support of each candidate itemset
  - To reduce the number of comparisons, store the candidates in a hash structure
    - Instead of matching each transaction against every candidate, match it against candidates contained in the hashed buckets
How to Count Supports of Candidates?

• Why counting supports of candidates a problem?
  – The total number of candidates can be very huge
  – One transaction may contain many candidates

• Method:
  – Candidate itemsets are stored in a hash-tree
  – Leaf node of hash-tree contains a list of itemsets and counts
  – Interior node contains a hash table
  – Subset function: finds all the candidates contained in a transaction
Generate Hash Tree

- Suppose you have 15 candidate itemsets of length 3:
  - \{1 4 5\}, \{1 2 4\}, \{4 5 7\}, \{1 2 5\}, \{4 5 8\}, \{1 5 9\}, \{1 3 6\}, \{2 3 4\}, \{5 6 7\}, \{3 4 5\}, \{3 5 6\}, \{3 5 7\}, \{6 8 9\}, \{3 6 7\}, \{3 6 8\}

- We need:
  - Hash function
  - Max leaf size: max number of itemsets stored in a leaf node (if number of candidate itemsets exceeds max leaf size, split the node)

- HashFunc: mod 3
Subset Operation

• Given a transaction t, what are the possible subsets of size 3?
Subset Operation Using Hash Tree

Transaction: 1 2 3 5 6

Hash Function:
- 1, 4, 7
- 2, 5, 8
- 3, 6, 9
Subset Operation Using Hash Tree

Hash Function

Data Mining
Subset Operation Using Hash Tree

Match transaction against 11 out of 15 candidates
Factors Affecting Complexity

• **Choice of minimum support threshold**
  – lowering support threshold results in more frequent itemsets
  – this may increase number of candidates and max length of frequent itemsets

• **Dimensionality (number of items) of the data set**
  – more space is needed to store support count of each item
  – if number of frequent items also increases, both computation and I/O costs may also increase

• **Size of database**
  – since Apriori makes multiple passes, run time of algorithm may increase with number of transactions

• **Average transaction width**
  – transaction width increases with denser data sets
  – This may increase max length of frequent itemsets and traversals of hash tree (number of subsets in a transaction increases with its width)
Effect of support threshold

(a) Number of candidate itemsets.

(b) Number of frequent itemsets.
Effect of average transaction width

(a) Number of candidate itemsets.

(b) Number of Frequent Itemsets.
Compact Representation of Frequent Itemsets

- Some itemsets are redundant because they have identical support as their supersets

- Number of frequent itemsets = \(3 \times \sum_{k=1}^{10} \binom{10}{k}\)

- Need a compact representation
  - Maximal and Closed Itemsets
Maximal Frequent Itemset

- An itemset is maximal frequent if none of its immediate supersets is frequent
Closed Itemset

- An itemset is closed if none of its immediate supersets has the same support as the itemset.
Maximal vs Closed Itemsets

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ABC</td>
</tr>
<tr>
<td>2</td>
<td>ABCD</td>
</tr>
<tr>
<td>3</td>
<td>BCE</td>
</tr>
<tr>
<td>4</td>
<td>ACDE</td>
</tr>
<tr>
<td>5</td>
<td>DE</td>
</tr>
</tbody>
</table>

Minsup: %40

Transaction Ids

Not supported by any transactions
Maximal vs Closed Frequent Itemsets

Minimum support = 2

# Closed = 9
# Maximal = 4
Maximal vs Closed Itemsets
FP-growth Algorithm

- **FP-growth algorithm** that takes a radically different approach to discovering frequent itemsets.
  - The algorithm does not subscribe to the generate-and-test paradigm of Apriori
- **FP-growth algorithm** encodes the data set using a compact data structure called an FP-tree and extracts frequent itemsets directly from this structure.
- Use a compressed representation of the database using an FP-tree
- Once an FP-tree has been constructed, it uses a recursive divide-and-conquer approach to mine the frequent itemsets
FP-tree construction

- An FP-tree is a compressed representation of the input data.
- It is constructed by reading the data set one transaction at a time and mapping each transaction onto a path in the FP-tree.
- Different transactions can have several items in common, their paths may overlap.
- The more the paths overlap with one another, the more compression we can achieve using the FP-tree structure.
- Each node in the tree contains the label of an item along with a counter that shows the number of transactions mapped onto the given path.
  - Initially, the FP-tree contains only the root node represented by the null symbol.
  - Every transaction maps onto one of the paths in the FP-tree.
- The size of an FP-tree is typically smaller than the size of the uncompressed data because many transactions in market basket data often share a few items in common.
  - best-case scenario, all transactions have same set of items ➔ FP-tree contains only a single branch.
  - worst-case scenario happens when every transaction has a unique set of items ➔ FP-tree is effectively the same as the size of the original data.
  - physical storage requirement for FP-tree is higher because it requires additional space to store pointers between nodes and counters for each item.
FP-tree construction

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{A,B}</td>
</tr>
<tr>
<td>2</td>
<td>{B,C,D}</td>
</tr>
<tr>
<td>3</td>
<td>{A,C,D,E}</td>
</tr>
<tr>
<td>4</td>
<td>{A,D,E}</td>
</tr>
<tr>
<td>5</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>6</td>
<td>{A,B,C,D}</td>
</tr>
<tr>
<td>7</td>
<td>{B,C}</td>
</tr>
<tr>
<td>8</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>9</td>
<td>{A,B,D}</td>
</tr>
<tr>
<td>10</td>
<td>{B,C,E}</td>
</tr>
</tbody>
</table>

After reading TID=1:

After reading TID=2:

(iii) After reading TID=3

Data Mining
FP-Tree Construction

Transaction Database

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{A,B}</td>
</tr>
<tr>
<td>2</td>
<td>{B,C,D}</td>
</tr>
<tr>
<td>3</td>
<td>{A,C,D,E}</td>
</tr>
<tr>
<td>4</td>
<td>{A,D,E}</td>
</tr>
<tr>
<td>5</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>6</td>
<td>{A,B,C,D}</td>
</tr>
<tr>
<td>7</td>
<td>{B,C}</td>
</tr>
<tr>
<td>8</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>9</td>
<td>{A,B,D}</td>
</tr>
<tr>
<td>10</td>
<td>{B,C,E}</td>
</tr>
</tbody>
</table>

Header table

<table>
<thead>
<tr>
<th>Item</th>
<th>Pointer</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
</tr>
</tbody>
</table>

Pointers are used to assist frequent itemset generation
Frequent Itemset Generation in FP-Growth Algorithm

- FP-growth is an algorithm that generates frequent itemsets from an FP-tree by exploring the tree in a bottom-up fashion.
  - This bottom-up strategy for finding frequent itemsets ending with a particular item is equivalent to the suffix-based approach.
  - Since every transaction is mapped onto a path in the FP-tree, we can derive the frequent itemsets ending with a particular item, say e, by examining only the paths containing node e.
  - The algorithm looks for frequent itemsets ending in e first, followed by d, c, b, and finally, a.

- FP-growth finds all the frequent itemsets ending with a particular suffix by employing a divide-and-conquer strategy to split the problem into smaller subproblems.
  - To find all frequent itemsets ending in e, we must first check whether the itemset \{e\} itself is frequent.
  - If it is frequent, we consider the subproblem of finding frequent itemsets ending in de, followed by ce, be, and ae.
  - In turn, each of these subproblems are further decomposed into smaller subproblems.
  - By merging the solutions obtained from the subproblems, all the frequent itemsets ending in e can be found.
finding frequent itemsets ending with e.

1. The first step is to gather all the paths containing node e. These initial paths are called prefix paths.

2. From the prefix paths, the support count for e is obtained by adding the support counts associated with node e. Assuming that the minimum support count is 2, \{e\} is declared a frequent itemset because its support count is 3.

3. Because \{e\} is frequent, the algorithm has to solve the subproblems of finding frequent itemsets ending in de, ce, be, and ae. Before solving these subproblems, it must first convert the prefix paths into a conditional FP-tree, which is structurally similar to an FP-tree, except it is used to find frequent itemsets ending with a particular suffix.
   – First, the support counts along the prefix paths must be updated because some of the counts include transactions that do not contain item e.
   – The prefix paths are truncated by removing the nodes for e.
   – After updating the support counts along the prefix paths, some of the items may no longer be frequent
     • the node b appears only once and has a support count equal to 1, which means that there is only one transaction that contains both b and e. Item b can be safely ignored from subsequent analysis because all itemsets ending in be must be infrequent.

4. FP-growth uses the conditional FP-tree for e to solve the subproblems of finding frequent itemsets ending in de, ce, and ae.
Prefix Paths and Conditional FP-tree

Prefix paths ending with e

Conditional FP-tree for e

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{A,B}</td>
</tr>
<tr>
<td>2</td>
<td>{B,C,D}</td>
</tr>
<tr>
<td>3</td>
<td>{A,C,D,E}</td>
</tr>
<tr>
<td>4</td>
<td>{A,D,E}</td>
</tr>
<tr>
<td>5</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>6</td>
<td>{A,B,C,D}</td>
</tr>
<tr>
<td>7</td>
<td>{B,C}</td>
</tr>
<tr>
<td>8</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>9</td>
<td>{A,B,D}</td>
</tr>
<tr>
<td>10</td>
<td>{B,C,E}</td>
</tr>
</tbody>
</table>
frequent itemsets ordered by suffixes

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{A,B}</td>
</tr>
<tr>
<td>2</td>
<td>{B,C,D}</td>
</tr>
<tr>
<td>3</td>
<td>{A,C,D,E}</td>
</tr>
<tr>
<td>4</td>
<td>{A,D,E}</td>
</tr>
<tr>
<td>5</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>6</td>
<td>{A,B,C,D}</td>
</tr>
<tr>
<td>7</td>
<td>{B,C}</td>
</tr>
<tr>
<td>8</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>9</td>
<td>{A,B,D}</td>
</tr>
<tr>
<td>10</td>
<td>{B,C,E}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Suffix</th>
<th>Frequent Itemsets</th>
</tr>
</thead>
<tbody>
<tr>
<td>e</td>
<td>{e}, {d,e}, {a,d,e}, {c,e}, {a,e}</td>
</tr>
<tr>
<td>d</td>
<td>{d}, {c,d}, {b,c,d}, {a,c,d}, {b,d}, {a,b,d}, {a,d}</td>
</tr>
<tr>
<td>c</td>
<td>{c}, {b,c}, {a,b,c}, {a,c}</td>
</tr>
<tr>
<td>b</td>
<td>{b}, {a,b}</td>
</tr>
<tr>
<td>a</td>
<td>{a}</td>
</tr>
</tbody>
</table>

null

A:7

B:5

C:3

null

Header table

<table>
<thead>
<tr>
<th>Item</th>
<th>Pointer</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
</tr>
</tbody>
</table>
Rule Generation

- Given a frequent itemset L, find all non-empty subsets $f \subset L$ such that $f \rightarrow L - f$ satisfies the minimum confidence requirement
  - If \{A,B,C,D\} is a frequent itemset, candidate rules:
    
    \begin{align*}
    ABC & \rightarrow D, & ABD & \rightarrow C, & ACD & \rightarrow B, & BCD & \rightarrow A, \\
    A & \rightarrow BCD, & B & \rightarrow ACD, & C & \rightarrow ABD, & D & \rightarrow ABC \\
    AB & \rightarrow CD, & AC & \rightarrow BD, & AD & \rightarrow BC, & BC & \rightarrow AD, \\
    BD & \rightarrow AC, & CD & \rightarrow AB, & & & \\
    \end{align*}

- If $|L| = k$, then there are $2^k - 2$ candidate association rules (ignoring $L \rightarrow \emptyset$ and $\emptyset \rightarrow L$)
Rule Generation

• How to efficiently generate rules from frequent itemsets?
  – In general, confidence does not have an anti-monotone property
    \[ c(ABC \rightarrow D) \text{ can be larger or smaller than } c(AB \rightarrow D) \]
  – But confidence of rules generated from the same itemset has an anti-monotone property
  – e.g., \( L = \{A, B, C, D\} \):
    \[ c(ABC \rightarrow D) \geq c(AB \rightarrow CD) \geq c(A \rightarrow BCD) \]

  • Confidence is anti-monotone w.r.t. number of items on the RHS of the rule
Rule Generation for Apriori Algorithm

Lattice of rules

- Low Confidence Rule

Pruned Rules

ABC = { }

BCD => A
ACD => B
ABD => C
ABC => D
BC => AD
BD => AC
CD => AB
AD => BC
AC => BD
AB => CD
D => ABC
C => ABD
B => ACD
A => BCD

Data Mining
**Rule Generation for Apriori Algorithm**

- Candidate rule is generated by merging two rules that share the same prefix in the rule consequent.

- \( \text{join}(CD \Rightarrow AB, BD \Rightarrow AC) \) would produce the candidate rule \( D \Rightarrow ABC \).

- Prune rule \( D \Rightarrow ABC \) if its subset \( AD \Rightarrow BC \) does not have high confidence.
Effect of Support Distribution

- Many real data sets have skewed support distribution

Support distribution of a retail data set
Effect of Support Distribution

• How to set the appropriate \( \text{minsup} \) threshold?
  – If \( \text{minsup} \) is set too high, we could miss itemsets involving interesting rare items (e.g., expensive products)
  – If \( \text{minsup} \) is set too low, it is computationally expensive and the number of itemsets is very large

• Using a single minimum support threshold may not be effective
Multiple Minimum Support

• How to apply multiple minimum supports?
  – MS(i): minimum support for item i
  – e.g.: MS(Milk)=5%, MS(Coke) = 3%, MS(Broccoli)=0.1%, MS(Salmon)=0.5%
  – MS({Milk, Broccoli}) = min (MS(Milk), MS(Broccoli))
    = 0.1%

  – Challenge: Support is no longer anti-monotone
    • Suppose: Support(Milk, Coke) = 1.5% and Support(Milk, Coke, Broccoli) = 0.5%

    • {Milk,Coke} is infrequent but {Milk,Coke,Broccoli} is frequent
Multiple Minimum Support

<table>
<thead>
<tr>
<th>Item</th>
<th>MS(I)</th>
<th>Sup(I)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.10%</td>
<td>0.25%</td>
</tr>
<tr>
<td>B</td>
<td>0.20%</td>
<td>0.26%</td>
</tr>
<tr>
<td>C</td>
<td>0.30%</td>
<td>0.29%</td>
</tr>
<tr>
<td>D</td>
<td>0.50%</td>
<td>0.05%</td>
</tr>
<tr>
<td>E</td>
<td>3%</td>
<td>4.20%</td>
</tr>
</tbody>
</table>
Multiple Minimum Support

<table>
<thead>
<tr>
<th>Item</th>
<th>MS(I)</th>
<th>Sup(I)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.10%</td>
<td>0.25%</td>
</tr>
<tr>
<td>B</td>
<td>0.20%</td>
<td>0.26%</td>
</tr>
<tr>
<td>C</td>
<td>0.30%</td>
<td>0.29%</td>
</tr>
<tr>
<td>D</td>
<td>0.50%</td>
<td>0.05%</td>
</tr>
<tr>
<td>E</td>
<td>3%</td>
<td>4.20%</td>
</tr>
</tbody>
</table>
Multiple Minimum Support

- Order the items according to their minimum support (in ascending order)
  - e.g.: MS(Milk) = 5%, MS(Coke) = 3%, MS(Broccoli) = 0.1%, MS(Salmon) = 0.5%
  - Ordering: Broccoli, Salmon, Coke, Milk

- Need to modify Apriori such that:
  - $L_1$: set of frequent items
  - $F_1$: set of items whose support is $\geq$ MS(1) where MS(1) is $\min_i$ (MS(i))
  - $C_2$: candidate itemsets of size 2 is generated from $F_1$ instead of $L_1$
Multiple Minimum Support

• Modifications to Apriori:
  – In traditional Apriori,
    • A candidate \((k+1)\)-itemset is generated by merging two frequent itemsets of size \(k\)
    • The candidate is pruned if it contains any infrequent subsets of size \(k\)
  – Pruning step has to be modified:
    • Prune only if subset contains the first item
    • e.g.: Candidate=\{Broccoli, Coke, Milk\} (ordered according to minimum support)
    • \{Broccoli, Coke\} and \{Broccoli, Milk\} are frequent but \{Coke, Milk\} is infrequent
      – Candidate is not pruned because \{Coke, Milk\} does not contain the first item, i.e., Broccoli.
Evaluation of Association Patterns

• Association rule algorithms tend to produce too many rules
  – many of them are *uninteresting* or *redundant*
  – *Redundant* if \{A,B,C\} \rightarrow \{D\} and \{A,B\} \rightarrow \{D\}
    have same support & confidence

• *Interestingness measures* can be used to prune/rank the derived patterns

• In the original formulation of association rules, support & confidence are the only measures used
Application of Interestingness Measure

Interestingness Measures

Preprocessed Data

Selected Data

Data

Patterns

Preprocessing

Mining

Postprocessing

Knowledge
Computing Interestingness Measure

- Given a rule $X \rightarrow Y$, information needed to compute rule interestingness can be obtained from a contingency table.

**Contingency table** for $X \rightarrow Y$

<table>
<thead>
<tr>
<th></th>
<th>$Y$</th>
<th>$\neg Y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>$f_{11}$</td>
<td>$f_{10}$</td>
</tr>
<tr>
<td>$\neg X$</td>
<td>$f_{01}$</td>
<td>$f_{00}$</td>
</tr>
<tr>
<td>$f_{+1}$</td>
<td>$f_{+0}$</td>
<td>$</td>
</tr>
</tbody>
</table>

- $f_{11}$: support of $X$ and $Y$
- $f_{10}$: support of $X$ and $\neg Y$
- $f_{01}$: support of $\neg X$ and $Y$
- $f_{00}$: support of $\neg X$ and $\neg Y$
Drawback of Confidence

<table>
<thead>
<tr>
<th></th>
<th>Coffee</th>
<th></th>
<th>Coffee</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tea</td>
<td>15</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>Tea</td>
<td>75</td>
<td>5</td>
<td>80</td>
</tr>
<tr>
<td></td>
<td>90</td>
<td>10</td>
<td>100</td>
</tr>
</tbody>
</table>

Association Rule: Tea → Coffee

Confidence = \( P(\text{Coffee}|\text{Tea}) = 0.75 \)

but \( P(\text{Coffee}) = 0.9 \)

⇒ Although confidence is high, rule is misleading

⇒ \( P(\text{Coffee}|\text{Tea}) = 0.9375 \)
Statistical Independence

- Population of 1000 students
  - 600 students know how to swim (S)
  - 700 students know how to bike (B)
  - 420 students know how to swim and bike (S,B)

  - $P(S \land B) = \frac{420}{1000} = 0.42$
  - $P(S) \times P(B) = 0.6 \times 0.7 = 0.42$

  - $P(S \land B) = P(S) \times P(B) \Rightarrow$ Statistical independence
  - $P(S \land B) > P(S) \times P(B) \Rightarrow$ Positively correlated
  - $P(S \land B) < P(S) \times P(B) \Rightarrow$ Negatively correlated
Statistical-based Measures

• Measures that take into account statistical dependence

\[
Lift = \frac{P(Y \mid X)}{P(Y)}
\]

\[
Interest = \frac{P(X,Y)}{P(X)P(Y)}
\]

\[
PS = P(X,Y) - P(X)P(Y)
\]

\[
\phi - coefficien t = \frac{P(X,Y) - P(X)P(Y)}{\sqrt{P(X)[1 - P(X)]P(Y)[1 - P(Y)]}}
\]
Lift and Interest Factors

- Lift

\[ Lift = \frac{c(A \rightarrow B)}{s(B)} \]

- Interest Factor

\[ I(A, B) = \frac{s(A, B)}{s(A) \times s(B)} = \frac{N f_{11}}{f_{1+} f_{+1}} \]

<table>
<thead>
<tr>
<th></th>
<th>Y</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>f_{11}</td>
<td>f_{10}</td>
</tr>
<tr>
<td>X</td>
<td>f_{01}</td>
<td>f_{00}</td>
</tr>
<tr>
<td></td>
<td>f_{+1}</td>
<td>f_{+0}</td>
</tr>
</tbody>
</table>

- Lift and Interest Factors are equivalent.

\[ \text{Lift} = \frac{P(Y \mid X)}{P(Y)} \]

\[ \text{Interest} = \frac{P(X, Y)}{P(X) P(Y)} \]

\[ I(A, B) \begin{cases} = 1, & \text{if } A \text{ and } B \text{ are independent;} \\ > 1, & \text{if } A \text{ and } B \text{ are positively correlated;} \\ < 1, & \text{if } A \text{ and } B \text{ are negatively correlated.} \end{cases} \]
Example: Lift/Interest

<table>
<thead>
<tr>
<th></th>
<th>Coffee</th>
<th>Coffee</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tea</strong></td>
<td>15</td>
<td>5</td>
</tr>
<tr>
<td><strong>Tea</strong></td>
<td>75</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>90</td>
<td>10</td>
</tr>
</tbody>
</table>

Association Rule: Tea $\rightarrow$ Coffee

Confidence = $P(\text{Coffee}|\text{Tea}) = 0.75$

but $P(\text{Coffee}) = 0.9$

$\Rightarrow$ Lift = $0.75/0.9 = 0.8333$ ($< 1$, therefore is negatively associated)

$\Rightarrow$ a slight negative correlation between tea drinkers and coffee drinkers
Limitations of Interest Factor

• we expect the words data and mining to appear together more frequently than the words compiler and mining in a collection of computer science articles.

<table>
<thead>
<tr>
<th></th>
<th>p</th>
<th>(\overline{p})</th>
<th>q</th>
<th>(\overline{q})</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>880</td>
<td>50</td>
<td>930</td>
<td>70</td>
</tr>
<tr>
<td>(\overline{q})</td>
<td>50</td>
<td>20</td>
<td>70</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td>930</td>
<td>70</td>
<td>1000</td>
<td>1000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>r</th>
<th>(\overline{r})</th>
<th>s</th>
<th>(\overline{s})</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>20</td>
<td>50</td>
<td>70</td>
<td>930</td>
</tr>
<tr>
<td>(\overline{s})</td>
<td>50</td>
<td>880</td>
<td>930</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>930</td>
<td>1000</td>
<td></td>
</tr>
</tbody>
</table>

Contingency tables for word pairs \{p,q\} and \{r,s\}.

• The interest factor for \{p,q\} is 1.02 and for \{r, s\} is 4.08.
  – Although p and q appear together in 88% of the documents, their interest factor is close to 1, which is the value when p and q are statistically independent.
  – On the other hand, the interest factor for \{r, s\} is higher than \{p, q\} even though r and s seldom appear together in the same document.
  – Confidence is perhaps the better choice in this situation because it considers the association between p and q (94.6%) to be much stronger than that between r and s (28.6%).
Different Measures

- There are lots of measures proposed in the literature.
- Some measures are good for certain applications, but not for others.
- What criteria should we use to determine whether a measure is good or bad?

<table>
<thead>
<tr>
<th>#</th>
<th>Measure</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\phi$-coefficient</td>
<td>$\frac{P(A\mid B) - P(A)P(B)}{\sqrt{P(A)P(B)(1 - P(A))(1 - P(B))}}$</td>
</tr>
<tr>
<td>2</td>
<td>Goodman-Kruskal's $\lambda$</td>
<td>$\frac{\sum \max P(A_j, B_k) + \sum \max P(A_j, B_k) - \max P(A_j) - \max P(B_k)}{2\max P(A_j) - \max P(B_k)}$</td>
</tr>
<tr>
<td>3</td>
<td>Odds ratio ($\alpha$)</td>
<td>$\frac{P(A\mid B)P(B)}{P(A\mid \overline{B})P(\overline{B})}$</td>
</tr>
<tr>
<td>4</td>
<td>Yule's $Q$</td>
<td>$\frac{P(A, B)P(\overline{A}, \overline{B}) - P(A, \overline{B})P(\overline{A}, B)}{P(A, B)P(\overline{A}, B) + P(A, \overline{B})P(\overline{A}, \overline{B})} = \frac{a - 1}{a + 1}$</td>
</tr>
<tr>
<td>5</td>
<td>Yule's $Y$</td>
<td>$\frac{P(A, B)P(\overline{A}, \overline{B}) + P(A, \overline{B})P(\overline{A}, B)}{P(A, B) + P(A, \overline{B})} = \frac{\sqrt{a + 1}}{\sqrt{a + 1}}$</td>
</tr>
<tr>
<td>6</td>
<td>Kappa ($\kappa$)</td>
<td>$\frac{1 - P(A)P(B) - P(A)\overline{P}(B) - P(A)\overline{P}(B)}{\sum \sum P(A_i, B_j) \log \frac{P(A_i)P(B_j)}{P(A)P(B)}}$</td>
</tr>
<tr>
<td>7</td>
<td>Mutual Information ($M$)</td>
<td>$\min(- \sum \log P(A_i) - \sum \log P(B_j) \log \frac{P(A_i)P(B_j)}{P(A)P(B)})$</td>
</tr>
<tr>
<td>8</td>
<td>J-Measure ($J$)</td>
<td>$\max \left( P(A, B) \log \frac{P(A \mid B)}{P(A)} + P(\overline{A}, B) \log \frac{P(\overline{A} \mid B)}{P(\overline{A})} \right)$</td>
</tr>
<tr>
<td>9</td>
<td>Gini index ($G$)</td>
<td>$\max \left( P(A) [P(B \mid A)^2 + P(B \mid \overline{A})^2] + P(\overline{A}) [P(B \mid A)^2 + P(B \mid \overline{A})^2] \right)$</td>
</tr>
<tr>
<td>10</td>
<td>Support ($s$)</td>
<td>$\max {P(B \mid A), P(A \mid B)}$</td>
</tr>
<tr>
<td>11</td>
<td>Confidence ($c$)</td>
<td>$\max \left( \frac{NP(A \mid B) + 1}{NP(A)} - \frac{NP(A, B)}{NP(A) - 1} \right)$</td>
</tr>
<tr>
<td>12</td>
<td>Laplace ($L$)</td>
<td>$\max \left( \frac{P(A, B)P(B)}{P(A)P(B)} \right)$</td>
</tr>
<tr>
<td>13</td>
<td>Conviction ($V$)</td>
<td>$\max \left( \frac{P(A)P(B)}{P(\overline{A})P(B)} \right)$</td>
</tr>
<tr>
<td>14</td>
<td>Interest ($I$)</td>
<td>$\frac{P(A)P(B)}{P(B)}$</td>
</tr>
<tr>
<td>15</td>
<td>cosine ($S$)</td>
<td>$\frac{P(A, B)}{\sqrt{P(A)P(B)}}$</td>
</tr>
<tr>
<td>16</td>
<td>Piatesky-Shapiro's $PS$</td>
<td>$P(A, B) - P(A)P(B)$</td>
</tr>
<tr>
<td>17</td>
<td>Certainty factor ($F$)</td>
<td>$\max \left( \frac{P(B \mid A) - P(B)}{1 - P(B)}, \frac{P(A \mid B) - P(A)}{1 - P(A)} \right)$</td>
</tr>
<tr>
<td>18</td>
<td>Added Value ($AV$)</td>
<td>$\max {P(B \mid A), P(A \mid B)} - P(A)$</td>
</tr>
<tr>
<td>19</td>
<td>Collective strength ($S$)</td>
<td>$\frac{P(A, B) - P(B)}{P(B)} \times \frac{1 - P(A)P(B) - P(A \mid B)}{1 - P(A)P(B)}$</td>
</tr>
<tr>
<td>20</td>
<td>Jaccard ($\zeta$)</td>
<td>$\frac{P(A \cap B)}{P(A) + P(B) - P(A \cap B)}$</td>
</tr>
<tr>
<td>21</td>
<td>Klosgen ($K$)</td>
<td>$\sqrt{P(A \mid B) \max {P(B \mid A) - P(B), P(A \mid B) - P(A)}}$</td>
</tr>
</tbody>
</table>
Properties of A Good Measure

3 properties a good measure $M$ must satisfy:

- $M(A,B) = 0$ if $A$ and $B$ are statistically independent

- $M(A,B)$ increase monotonically with $P(A,B)$ when $P(A)$ and $P(B)$ remain unchanged

- $M(A,B)$ decreases monotonically with $P(A)$ [or $P(B)$] when $P(A,B)$ and $P(B)$ [or $P(A)$] remain unchanged
Comparing Different Measures

10 examples of contingency tables:

<table>
<thead>
<tr>
<th>Example</th>
<th>f_{11}</th>
<th>f_{10}</th>
<th>f_{01}</th>
<th>f_{00}</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
<td>8123</td>
<td>83</td>
<td>424</td>
<td>1370</td>
</tr>
<tr>
<td>E2</td>
<td>8330</td>
<td>2</td>
<td>622</td>
<td>1046</td>
</tr>
<tr>
<td>E3</td>
<td>9481</td>
<td>94</td>
<td>127</td>
<td>298</td>
</tr>
<tr>
<td>E4</td>
<td>3954</td>
<td>3080</td>
<td>5</td>
<td>2961</td>
</tr>
<tr>
<td>E5</td>
<td>2886</td>
<td>1363</td>
<td>1320</td>
<td>4431</td>
</tr>
<tr>
<td>E6</td>
<td>1500</td>
<td>2000</td>
<td>500</td>
<td>6000</td>
</tr>
<tr>
<td>E7</td>
<td>4000</td>
<td>2000</td>
<td>1000</td>
<td>3000</td>
</tr>
<tr>
<td>E8</td>
<td>4000</td>
<td>2000</td>
<td>2000</td>
<td>2000</td>
</tr>
<tr>
<td>E9</td>
<td>1720</td>
<td>7121</td>
<td>5</td>
<td>1154</td>
</tr>
<tr>
<td>E10</td>
<td>61</td>
<td>2483</td>
<td>4</td>
<td>7452</td>
</tr>
</tbody>
</table>

Rankings of contingency tables using various measures:

<table>
<thead>
<tr>
<th>#</th>
<th>φ</th>
<th>λ</th>
<th>α</th>
<th>Q</th>
<th>Y</th>
<th>κ</th>
<th>M</th>
<th>J</th>
<th>G</th>
<th>s</th>
<th>c</th>
<th>L</th>
<th>V</th>
<th>I</th>
<th>IS</th>
<th>PS</th>
<th>F</th>
<th>AV</th>
<th>S</th>
<th>ζ</th>
<th>K</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>E2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>8</td>
<td>3</td>
<td>5</td>
<td>1</td>
<td>8</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>E3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>7</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>10</td>
<td>1</td>
<td>8</td>
<td>6</td>
<td>10</td>
<td>3</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>E4</td>
<td>4</td>
<td>7</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>E5</td>
<td>5</td>
<td>4</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>5</td>
<td>4</td>
<td>7</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>9</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>E6</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>9</td>
<td>8</td>
<td>8</td>
<td>7</td>
<td>2</td>
<td>8</td>
<td>6</td>
<td>7</td>
<td>2</td>
<td>7</td>
<td>8</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>E7</td>
<td>7</td>
<td>5</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>6</td>
<td>8</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>7</td>
<td>7</td>
<td>8</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>8</td>
<td>5</td>
<td>6</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>E8</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>8</td>
<td>10</td>
<td>10</td>
<td>8</td>
<td>4</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>9</td>
<td>7</td>
<td>7</td>
<td>10</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>E9</td>
<td>9</td>
<td>9</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>9</td>
<td>9</td>
<td>7</td>
<td>8</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>7</td>
<td>9</td>
<td>9</td>
<td>3</td>
<td>7</td>
<td>9</td>
<td>9</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>E10</td>
<td>10</td>
<td>8</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>10</td>
<td>5</td>
<td>9</td>
<td>10</td>
<td>10</td>
<td>6</td>
<td>6</td>
<td>5</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td>7</td>
</tr>
</tbody>
</table>
Inversion Property

- If a measure is **invariant** under the inversion operation, then its value for the vector pair (C, D) should be identical to its value for (A, B).
- An objective measure M is invariant under the inversion operation if its value remains the same when exchanging the frequency counts $f_{11}$ with $f_{00}$ and $f_{10}$ with $f_{01}$.

Inversion Operation

**Symmetric measures:**

- support, lift, correlation, collective strength, cosine, Jaccard, etc

**Asymmetric measures:**

- confidence, conviction, Laplace, J-measure, etc

Symmetric measures may not be suitable for analyzing asymmetric binary data.
Null Addition Property

• Suppose we are interested in analyzing the relationship between a pair of words, such as data and mining, in a set of documents.
  – If a collection of articles about ice fishing is added to the data set, should the association between data and mining be affected?
  – This process of adding unrelated data (in this case, documents) to a given data set is known as the null addition operation.

• Null Addition Property: An objective measure $M$ is invariant under the null addition operation if it is not affected by increasing $f_{00}$, while all other frequencies in the contingency table stay the same.

• Cosine, Jaccard have Null Addition Property

• Interest Factor, Correlation Coefficient, PS do not have Null Addition Property
Scaling Property

Grade-Gender Example:

<table>
<thead>
<tr>
<th></th>
<th>Male</th>
<th>Female</th>
<th></th>
<th>Male</th>
<th>Female</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>30</td>
</tr>
<tr>
<td>Low</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>7</td>
<td>10</td>
<td>6</td>
<td>70</td>
</tr>
</tbody>
</table>

Underlying association should be independent of the relative number of male and female students in the samples

**Definition 6.8 (Scaling Invariance Property).** An objective measure $M$ is invariant under the row/column scaling operation if $M(T) = M(T')$, where $T$ is a contingency table with frequency counts $[f_{11}; f_{10}; f_{01}; f_{00}]$, $T'$ is a contingency table with scaled frequency counts $[k_1 k_3 f_{11}; k_2 k_3 f_{10}; k_1 k_4 f_{01}; k_2 k_4 f_{00}]$, and $k_1$, $k_2$, $k_3$, $k_4$ are positive constants.
## Properties of Symmetric Measures

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Measure</th>
<th>Inversion</th>
<th>Null Addition</th>
<th>Scaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\phi$</td>
<td>$\phi$-coefficient</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>odds ratio</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>$\kappa$</td>
<td>Cohen's</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>$I$</td>
<td>Interest</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>$IS$</td>
<td>Cosine</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>$PS$</td>
<td>Piatetsky-Shapiro's</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>$S$</td>
<td>Collective strength</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>$\zeta$</td>
<td>Jaccard</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>$h$</td>
<td>All-confidence</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>$s$</td>
<td>Support</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
Support-based Pruning

• Most of the association rule mining algorithms use support measure to prune rules and itemsets

• Study effect of support pruning on correlation of itemsets
  – Generate 10000 random contingency tables
  – Compute support and pairwise correlation for each table
  – Apply support-based pruning and examine the tables that are removed
Support-based Pruning

All Itempairs

Correlation
Effect of Support-based Pruning

Support-based pruning eliminates mostly negatively correlated itemsets
Subjective Interestingness Measure

- Objective measure:
  - Rank patterns based on statistics computed from data
  - e.g., 21 measures of association (support, confidence, Laplace, Gini, mutual information, Jaccard, etc).

- Subjective measure:
  - Rank patterns according to user’s interpretation
    - A pattern is subjectively interesting if it contradicts the expectation of a user
    - A pattern is subjectively interesting if it is actionable