Mining Frequent Patterns and Associations: Basic Concepts
(Chapter 6)
Huan Sun, CSE@The Ohio State University
10/17/2017
Mining Frequent Patterns, Association and Correlations: Basic Concepts and Methods

- Basic Concepts
- Efficient Pattern Mining Methods
- Pattern Evaluation
- Summary
Pattern Discovery: Basic Concepts

- What Is Pattern Discovery? Why Is It Important?
- Basic Concepts: Frequent Patterns and Association Rules
- Compressed Representation: Closed Patterns and Max-Patterns
What Is Pattern Discovery?

- Motivation examples:
  - What products were often purchased together?
  - What are the subsequent purchases after buying an iPad?
  - What code segments likely contain copy-and-paste bugs?
  - What word sequences likely form phrases in this corpus?
What Is Pattern Discovery?

- Motivation examples:
  - What products were often purchased together?
  - What are the subsequent purchases after buying an iPad?
  - What code segments likely contain copy-and-paste bugs?
  - What word sequences likely form phrases in this corpus?

- What are patterns?
  - Patterns: A set of items, subsequences, or substructures that occur frequently together (or strongly correlated) in a data set
  - Patterns represent intrinsic and important properties of datasets
What Is Pattern Discovery?

- **Motivation examples:**
  - What products were often purchased together?
  - What are the subsequent purchases after buying an iPad?
  - What code segments likely contain copy-and-paste bugs?
  - What word sequences likely form phrases in this corpus?

- **What are patterns?**
  - **Patterns:** A set of items, subsequences, or substructures that occur frequently together (or strongly correlated) in a data set
  - Patterns represent *intrinsic* and *important properties* of datasets

- **Pattern discovery:** Uncovering patterns from massive data sets
Pattern Discovery: Why Is It Important?

- Finding inherent regularities in a data set
- Foundation for many essential data mining tasks
  - Association, correlation, and causality analysis
  - Mining sequential, structural (e.g., sub-graph) patterns
  - Pattern analysis in spatiotemporal, multimedia, time-series, and stream data
  - Classification: Discriminative pattern-based analysis
  - Cluster analysis: Pattern-based subspace clustering
Pattern Discovery: Why Is It Important?

- Finding inherent regularities in a data set
- Foundation for many essential data mining tasks
  - Association, correlation, and causality analysis
  - Mining sequential, structural (e.g., sub-graph) patterns
  - Pattern analysis in spatiotemporal, multimedia, time-series, and stream data
  - Classification: Discriminative pattern-based analysis
  - Cluster analysis: Pattern-based subspace clustering
- Broad applications
  - Market basket analysis, cross-marketing, catalog design, sale campaign analysis, Web log analysis, biological sequence analysis
Basic Concepts: k-Itemsets and Their Supports

- **Itemset**: A set of one or more items
Basic Concepts: $k$-Itemsets and Their Supports

- **Itemset**: A set of one or more items
- **$k$-itemset**: $X = \{x_1, \ldots, x_k\}$
  - Ex. $\{\text{Beer, Nuts, Diaper}\}$ is a 3-itemset

<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>
Basic Concepts: k-Itemsets and Their Supports

- **Itemset**: A set of one or more items
- **k-itemset**: $X = \{x_1, \ldots, x_k\}$
  - Ex. \{Beer, Nuts, Diaper\} is a 3-itemset

- **(absolute) support (count)** of $X$, $\text{sup}\{X\}$: Frequency or the number of occurrences of an itemset $X$
  - Ex. $\text{sup}\{\text{Beer}\} = 3$
  - Ex. $\text{sup}\{\text{Diaper}\} = 4$
  - Ex. $\text{sup}\{\text{Beer, Diaper}\} = 3$
  - Ex. $\text{sup}\{\text{Beer, Eggs}\} = 1$

<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>
Basic Concepts: k-Itemsets and Their Supports

- **Itemset**: A set of one or more items
- **k-itemset**: \( X = \{x_1, \ldots, x_k\} \)
  - Ex. \( \{\text{Beer, Nuts, Diaper}\} \) is a 3-itemset

- **(absolute) support (count)** of \( X \), \( \text{sup}\{X\} \): Frequency or the number of occurrences of an itemset \( X \)
  - Ex. \( \text{sup}\{\text{Beer}\} = 3 \)
  - Ex. \( \text{sup}\{\text{Diaper}\} = 4 \)
  - Ex. \( \text{sup}\{\text{Beer, Diaper}\} = 3 \)
  - Ex. \( \text{sup}\{\text{Beer, Eggs}\} = 1 \)

- **(relative) support**, \( s\{X\} \): The fraction of transactions that contains \( X \) (i.e., the probability that a transaction contains \( X \))
  - Ex. \( s\{\text{Beer}\} = 3/5 = 60\% \)
  - Ex. \( s\{\text{Diaper}\} = 4/5 = 80\% \)
  - Ex. \( s\{\text{Beer, Eggs}\} = 1/5 = 20\% \)

<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>
Basic Concepts: Frequent Itemsets (Patterns)

- An itemset (or a pattern) $X$ is frequent if the support of $X$ is no less than a minsup threshold $\sigma$
An itemset (or a pattern) X is frequent if the support of X is no less than a minsup threshold \( \sigma \).

Let \( \sigma = 50\% \) (\( \sigma \): minsup threshold).

For the given 5-transaction dataset:
- All the frequent 1-itemsets:
  - Beer: 3/5 (60%); Nuts: 3/5 (60%)
  - Diaper: 4/5 (80%); Eggs: 3/5 (60%)
- All the frequent 2-itemsets:
  - \{Beer, Diaper\}: 3/5 (60%)
- All the frequent 3-itemsets?
  - None

<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>
An itemset (or a pattern) \( X \) is frequent if the support of \( X \) is no less than a minsup threshold \( \sigma \).

Let \( \sigma = 50\% \) (\( \sigma \): minsup threshold).

For the given 5-transaction dataset:
- All the frequent 1-itemsets:
  - Beer: 3/5 (60%); Nuts: 3/5 (60%)
  - Diaper: 4/5 (80%); Eggs: 3/5 (60%)
- All the frequent 2-itemsets:
  - \{Beer, Diaper\}: 3/5 (60%)
- All the frequent 3-itemsets?
  - None

Why do these itemsets (shown on the left) form the complete set of frequent \( k \)-itemsets (patterns) for any \( k \)?

**Observation**: We may need an efficient method to mine a complete set of frequent patterns.
From Frequent Itemsets to Association Rules

- Comparing with itemsets, rules can be more telling
  - Ex. *Diaper ➔ Beer*
    - Buying diapers may likely lead to buying beers
From Frequent Itemsets to Association Rules

- Ex. Diaper $\rightarrow$ Beer: Buying diapers may likely lead to buying beers

- How strong is this rule? (support, confidence)
  - Measuring association rules: $X \rightarrow Y (s, c)$
    - Both $X$ and $Y$ are itemsets
Ex. Diaper $\rightarrow$ Beer: Buying diapers may likely lead to buying beers

How strong is this rule? (support, confidence)

Measuring association rules: $X \rightarrow Y (s, c)$

- Both $X$ and $Y$ are itemsets

Support, $s$: The probability that a transaction contains $X \cup Y$

- Ex. $s\{\text{Diaper, Beer}\} = 3/5 = 0.6$ (i.e., 60%)

<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>
From Frequent Itemsets to Association Rules

- Ex. Diaper $\rightarrow$ Beer: Buying diapers may likely lead to buying beers
- How strong is this rule? (support, confidence)
- Measuring association rules: $X \rightarrow Y$ (s, c)
  - Both $X$ and $Y$ are itemsets
- Support, $s$: The probability that a transaction contains $X \cup Y$
  - Ex. $s\{\text{Diaper, Beer}\} = 3/5 = 0.6$ (i.e., 60%)
- Confidence, $c$: The conditional probability that a transaction containing $X$ also contains $Y$
  - Calculation: $c = \text{sup}(X \cup Y) / \text{sup}(X)$
  - Ex. $c = \text{sup}\{\text{Diaper, Beer}\} / \text{sup}\{\text{Diaper}\} = \frac{3}{4} = 0.75$

<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>

Containing both

Containing beer

{Beer} $\cup$ {Diaper}

{Beer, Diaper}

{Beer} $\cup$ {Diaper} = {Beer, Diaper}
Association rule mining
- Given two thresholds: $\text{minsup, minconf}$
- Find all of the rules, $X \rightarrow Y$ ($s$, $c$)
  - such that, $s \geq \text{minsup}$ and $c \geq \text{minconf}$
Mining Frequent Itemsets and Association Rules

- **Association rule mining**
  - Given two thresholds: \( \text{minsup} \), \( \text{minconf} \)
  - Find all of the rules, \( X \rightarrow Y \) \((s, c)\)
    - such that, \( s \geq \text{minsup} \) and \( c \geq \text{minconf} \)

- Let \( \text{minsup} = 50\% \)
  - Freq. 1-itemsets: Beer: 3, Nuts: 3, Diaper: 4, Eggs: 3
  - Freq. 2-itemsets: \{Beer, Diaper\}: 3

<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>
Mining Frequent Itemsets and Association Rules

- Association rule mining
  - Given two thresholds: \( \text{minsup, minconf} \)
  - Find all of the rules, \( X \rightarrow Y \) (s, c)
    - such that, \( s \geq \text{minsup} \) and \( c \geq \text{minconf} \)

- Let \( \text{minsup} = 50\% \)
  - Freq. 1-itemsets: Beer: 3, Nuts: 3, Diaper: 4, Eggs: 3
  - Freq. 2-itemsets: \{Beer, Diaper\}: 3

- Let \( \text{minconf} = 50\% \)
  - \( \text{Beer} \rightarrow \text{Diaper} \) (60%, 100%)
  - \( \text{Diaper} \rightarrow \text{Beer} \) (60%, 75%)

<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>
Mining Frequent Itemsets and Association Rules

- **Association rule mining**
  - Given two thresholds: minsup, minconf
  - Find all of the rules, \( X \rightarrow Y \) (s, c)
    - such that, \( s \geq \text{minsup} \) and \( c \geq \text{minconf} \)

- Let minsup = 50%
  - Freq. 1-itemsets: Beer: 3, Nuts: 3, Diaper: 4, Eggs: 3
  - Freq. 2-itemsets: \{Beer, Diaper\}: 3

- Let minconf = 50%
  - **Beer \rightarrow Diaper** (60%, 100%)
  - **Diaper \rightarrow Beer** (60%, 75%)

(Q: Are these all rules?)

<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>
Mining Frequent Itemsets and Association Rules

- **Association rule mining**
  - Given two thresholds: minsup, minconf
  - Find all of the rules, $X \rightarrow Y$ ($s$, $c$)
    - such that, $s \geq \text{minsup}$ and $c \geq \text{minconf}$
  - Let $\text{minsup} = 50\%$
    - Freq. 1-itemsets: Beer: 3, Nuts: 3, Diaper: 4, Eggs: 3
    - Freq. 2-itemsets: {Beer, Diaper}: 3
  - Let $\text{minconf} = 50\%$
    - Beer $\rightarrow$ Diaper (60%, 100%)
    - Diaper $\rightarrow$ Beer (60%, 75%)

- **Observations:**
  - Mining association rules and mining frequent patterns are very close problems
  - Scalable methods are needed for mining large datasets
Challenge: There Are Too Many Frequent Patterns!

- A long pattern contains a combinatorial number of sub-patterns
- How many frequent itemsets does the following $TDB_1$ contain?
  - $TDB_1$: $T_1: \{a_1, \ldots, a_{50}\}; T_2: \{a_1, \ldots, a_{100}\}$
  - Assuming (absolute) $\text{minsup} = 1$
  - Let’s have a try
    1-itemsets: $\{a_1\}: 2, \{a_2\}: 2, \ldots, \{a_{50}\}: 2, \{a_{51}\}: 1, \ldots, \{a_{100}\}: 1$
    2-itemsets: $\{a_1, a_2\}: 2, \ldots, \{a_1, a_{50}\}: 2, \{a_1, a_{51}\}: 1 \ldots, \ldots, \{a_{99}, a_{100}\}: 1$
    \ldots, \ldots, \ldots
    99-itemsets: $\{a_1, a_2, \ldots, a_{99}\}: 1, \ldots, \{a_2, a_3, \ldots, a_{100}\}: 1$
    100-itemset: $\{a_1, a_2, \ldots, a_{100}\}: 1$
Challenge: There Are Too Many Frequent Patterns!

- A long pattern contains a combinatorial number of sub-patterns
- How many frequent itemsets does the following TDB₁ contain?
  - TDB₁: \( T₁: \{a₁, …, a_{50}\}; \ T₂: \{a₁, …, a_{100}\} \)
  - Assuming (absolute) \( \text{minsup} = 1 \)
  - Let’s have a try
    - 1-itemsets: \( \{a₁\}: 2, \{a₂\}: 2, …, \{a_{50}\}: 2, \{a_{51}\}: 1, …, \{a_{100}\}: 1, \)
    - 2-itemsets: \( \{a₁, a₂\}: 2, …, \{a₁, a_{50}\}: 2, \{a₁, a_{51}\}: 1, …, …, \{a_{99}, a_{100}\}: 1, \)
      …, …, …, …
    - 99-itemsets: \( \{a₁, a₂, …, a_{99}\}: 1, …, \{a₂, a₃, …, a_{100}\}: 1 \)
    - 100-itemset: \( \{a₁, a₂, …, a_{100}\}: 1 \)
  - The total number of frequent itemsets:
    \[
    \binom{100}{1} + \binom{100}{2} + \binom{100}{3} + \cdots + \binom{100}{100} = 2^{100} - 1
    \]

A too huge set for any one to compute or store!
How to handle such a challenge?

Solution 1: **Closed patterns**: A pattern (itemset) $X$ is closed if $X$ is frequent, and there exists no super-pattern $Y \supset X$, with the same support as $X$. 
How to handle such a challenge?

Solution 1: **Closed patterns:** A pattern (itemset) X is closed if X is frequent, and there exists no super-pattern Y ⊂ X, with the same support as X

Let Transaction DB TDB1: T1: {a1, ..., a50}; T2: {a1, ..., a100}

Suppose minsup = 1. How many closed patterns does TDB1 contain?

- Two: P1: “{a1, ..., a50}: 2”; P2: “{a1, ..., a100}: 1”
Expressing Patterns in Compressed Form: Closed Patterns

- How to handle such a challenge?

- **Solution 1: Closed patterns**: A pattern (itemset) $X$ is **closed** if $X$ is frequent, and there exists no super-pattern $Y \subset X$, with the same support as $X$

  - Let Transaction DB $TDB_1$: $T_1: \{a_1, \ldots, a_{50}\}; \ T_2: \{a_1, \ldots, a_{100}\}$
  
  - Suppose $\text{minsup} = 1$. How many closed patterns does $TDB_1$ contain?
    - Two: $P_1: \{a_1, \ldots, a_{50}\}: 2$; $P_2: \{a_1, \ldots, a_{100}\}: 1$

- Closed pattern is a lossless compression of frequent patterns

  - Reduces the # of patterns but does not lose the support information!
  
  - You will still be able to say: $\{a_2, \ldots, a_{40}\}: 2$, $\{a_5, a_{51}\}: 1$
Solution 2: Max-patterns: A pattern $X$ is a max-pattern if $X$ is frequent and there exists no frequent super-pattern $Y \supseteq X$.
Solution 2: **Max-patterns:** A pattern X is a max-pattern if X is frequent and there exists no frequent super-pattern Y ⊆ X

**Difference from close-patterns?**

- Do not care the real support of the sub-patterns of a max-pattern
- Let Transaction DB TDB₁: T₁: \{a₁, ..., a₅₀\}; T₂: \{a₁, ..., a₁₀₀\}
- Suppose \( \text{minsup} = 1 \). How many max-patterns does TDB₁ contain?
  - One: P: \("\{a₁, ..., a₁₀₀\}: 1"\)
Solution 2: Max-patterns: A pattern X is a **max-pattern** if X is frequent and there exists no frequent super-pattern Y ⊆ X

Difference from close-patterns?
- Do not care the real support of the sub-patterns of a max-pattern
- Let Transaction DB TDB₁: T₁: {a₁, ..., a₅₀}; T₂: {a₁, ..., a₁₀₀}
- Suppose \( \text{minsup} = 1 \). How many max-patterns does TDB₁ contain?
  - One: P: “{a₁, ..., a₁₀₀}: 1”

Max-pattern is a lossy compression!
- We only know \{a₁, ..., a₄₀\} is frequent
- But we do not know the real support of \{a₁, ..., a₄₀\}, ..., any more!
- Thus in many applications, close-patterns are more desirable than max-patterns
Mining Frequent Patterns, Association and Correlations: Basic Concepts and Methods

- Basic Concepts

- Efficient Pattern Mining Methods
  - The Apriori Algorithm
  - Application in Classification

- Pattern Evaluation

- Summary
Efficient Pattern Mining Methods

- The Downward Closure Property of Frequent Patterns
- The Apriori Algorithm
- Extensions or Improvements of Apriori
- Mining Frequent Patterns by Exploring Vertical Data Format
- FPGrowth: A Frequent Pattern-Growth Approach
- Mining Closed Patterns
Observation: From TDB\(_1\): T\(_1\): \{a\(_1\), ..., a\(_{50}\)\}; T\(_2\): \{a\(_1\), ..., a\(_{100}\)\}

- We get a frequent itemset: \{a\(_1\), ..., a\(_{50}\)\}
- Also, its subsets are all frequent: \{a\(_1\)\}, \{a\(_2\)\}, ..., \{a\(_{50}\)\}, \{a\(_1\), a\(_2\)\}, ..., \{a\(_1\), ..., a\(_{49}\)\}, ...
- There must be some hidden relationships among frequent patterns!
The Downward Closure Property of Frequent Patterns

- Observation: From TDB₁: T₁: \{a₁, ..., a₅₀\}; T₂: \{a₁, ..., a₁₀₀\}
  - We get a frequent itemset: \{a₁, ..., a₅₀\}
  - Also, its subsets are all frequent: \{a₁\}, \{a₂\}, ..., \{a₅₀\}, \{a₁, a₂\}, ..., \{a₁, ..., a₄₉\}, ...
  - There must be some hidden relationships among frequent patterns!

- The downward closure (also called “Apriori”) property of frequent patterns
  - If \{beer, diaper, nuts\} is frequent, so is \{beer, diaper\}
  - Every transaction containing \{beer, diaper, nuts\} also contains \{beer, diaper\}
  - **Apriori**: Any subset of a frequent itemset must be frequent

A sharp knife for pruning!
Observation: From TDB₁: T₁: \{a₁, ..., a₅₀\}; T₂: \{a₁, ..., a₁₀₀\}

- We get a frequent itemset: \{a₁, ..., a₅₀\}
- Also, its subsets are all frequent: \{a₁\}, \{a₂\}, ..., \{a₅₀\}, \{a₁, a₂\}, ..., \{a₁, ..., a₄₉\}, ...
- There must be some hidden relationships among frequent patterns!

The downward closure (also called “Apriori”) property of frequent patterns

- If \{beer, diaper, nuts\} is frequent, so is \{beer, diaper\}
- Every transaction containing \{beer, diaper, nuts\} also contains \{beer, diaper\}
- **Apriori:** Any subset of a frequent itemset must be frequent

Efficient mining methodology

- If any subset of an itemset S is infrequent, then there is no chance for S to be frequent—why do we even have to consider S!? A sharp knife for pruning!
Apriori Pruning and Scalable Mining Methods

- **Apriori pruning principle:** If there is any itemset which is infrequent, its superset should not even be generated!
  - (Agrawal & Srikant @VLDB’94, Mannila, et al. @ KDD’94)

- **Scalable mining Methods:** Three major approaches
  - **Level-wise, join-based approach:**
    - Apriori (Agrawal & Srikant @VLDB’94)
  - **Vertical data format approach:**
    - Eclat (Zaki, Parthasarathy, Ogihara, Li @KDD’97)
  - **Frequent pattern projection and growth:**
    - FPgrowth (Han, Pei, Yin @SIGMOD’00)
Apriori: A Candidate Generation & Test Approach

- Outline of Apriori (level-wise, candidate generation and test)
  - Initially, scan DB once to get frequent 1-itemset
  - Repeat
    - Generate length-(k+1) candidate itemsets from length-k frequent itemsets
    - Test the candidates against DB to find frequent (k+1)-itemsets
    - Set k := k + 1
  - Until no frequent or candidate set can be generated
  - Return all the frequent itemsets derived
The Apriori Algorithm (Pseudo-Code)

C_k: Candidate itemset of size k
F_k : Frequent itemset of size k

K := 1;
F_k := {frequent items};       // frequent 1-itemset

While (F_k != ∅) do {
    // when F_k is non-empty
    C_{k+1} := candidates generated from F_k;   // candidate generation
    Derive F_{k+1} by counting candidates in C_{k+1} with respect to TDB at minsup;
    k := k + 1
}

return ∪_{k} F_k                   // return F_k generated at each level
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>

**minsup = 2**

**1st scan**

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>2</td>
</tr>
<tr>
<td>{B}</td>
<td>3</td>
</tr>
<tr>
<td>{C}</td>
<td>3</td>
</tr>
<tr>
<td>{D}</td>
<td>1</td>
</tr>
<tr>
<td>{E}</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>2</td>
</tr>
<tr>
<td>{B}</td>
<td>3</td>
</tr>
<tr>
<td>{C}</td>
<td>3</td>
</tr>
<tr>
<td>{E}</td>
<td>3</td>
</tr>
</tbody>
</table>

**2nd scan**

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B}</td>
<td>1</td>
</tr>
<tr>
<td>{A, C}</td>
<td>2</td>
</tr>
<tr>
<td>{A, E}</td>
<td>1</td>
</tr>
<tr>
<td>{B, C}</td>
<td>2</td>
</tr>
<tr>
<td>{B, E}</td>
<td>3</td>
</tr>
<tr>
<td>{C, E}</td>
<td>2</td>
</tr>
</tbody>
</table>

**3rd scan**

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{B, C, E}</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, C}</td>
<td>2</td>
</tr>
<tr>
<td>{B, C}</td>
<td>2</td>
</tr>
<tr>
<td>{B, E}</td>
<td>3</td>
</tr>
<tr>
<td>{C, E}</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B}</td>
<td>1</td>
</tr>
<tr>
<td>{A, C}</td>
<td>2</td>
</tr>
<tr>
<td>{A, E}</td>
<td>1</td>
</tr>
<tr>
<td>{B, C}</td>
<td>2</td>
</tr>
<tr>
<td>{B, E}</td>
<td>3</td>
</tr>
<tr>
<td>{C, E}</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>2</td>
</tr>
<tr>
<td>{B}</td>
<td>3</td>
</tr>
<tr>
<td>{C}</td>
<td>3</td>
</tr>
<tr>
<td>{E}</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>2</td>
</tr>
<tr>
<td>{B}</td>
<td>3</td>
</tr>
<tr>
<td>{C}</td>
<td>3</td>
</tr>
<tr>
<td>{E}</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B}</td>
<td>1</td>
</tr>
<tr>
<td>{A, C}</td>
<td>2</td>
</tr>
<tr>
<td>{A, E}</td>
<td>1</td>
</tr>
<tr>
<td>{B, C}</td>
<td>2</td>
</tr>
<tr>
<td>{B, E}</td>
<td>3</td>
</tr>
<tr>
<td>{C, E}</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>2</td>
</tr>
<tr>
<td>{B}</td>
<td>3</td>
</tr>
<tr>
<td>{C}</td>
<td>3</td>
</tr>
<tr>
<td>{E}</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B}</td>
<td>1</td>
</tr>
<tr>
<td>{A, C}</td>
<td>2</td>
</tr>
<tr>
<td>{A, E}</td>
<td>1</td>
</tr>
<tr>
<td>{B, C}</td>
<td>2</td>
</tr>
<tr>
<td>{B, E}</td>
<td>3</td>
</tr>
<tr>
<td>{C, E}</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>2</td>
</tr>
<tr>
<td>{B}</td>
<td>3</td>
</tr>
<tr>
<td>{C}</td>
<td>3</td>
</tr>
<tr>
<td>{E}</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B}</td>
<td>1</td>
</tr>
<tr>
<td>{A, C}</td>
<td>2</td>
</tr>
<tr>
<td>{A, E}</td>
<td>1</td>
</tr>
<tr>
<td>{B, C}</td>
<td>2</td>
</tr>
<tr>
<td>{B, E}</td>
<td>3</td>
</tr>
<tr>
<td>{C, E}</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>2</td>
</tr>
<tr>
<td>{B}</td>
<td>3</td>
</tr>
<tr>
<td>{C}</td>
<td>3</td>
</tr>
<tr>
<td>{E}</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B}</td>
<td>1</td>
</tr>
<tr>
<td>{A, C}</td>
<td>2</td>
</tr>
<tr>
<td>{A, E}</td>
<td>1</td>
</tr>
<tr>
<td>{B, C}</td>
<td>2</td>
</tr>
<tr>
<td>{B, E}</td>
<td>3</td>
</tr>
<tr>
<td>{C, E}</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>2</td>
</tr>
<tr>
<td>{B}</td>
<td>3</td>
</tr>
<tr>
<td>{C}</td>
<td>3</td>
</tr>
<tr>
<td>{E}</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B}</td>
<td>1</td>
</tr>
<tr>
<td>{A, C}</td>
<td>2</td>
</tr>
<tr>
<td>{A, E}</td>
<td>1</td>
</tr>
<tr>
<td>{B, C}</td>
<td>2</td>
</tr>
<tr>
<td>{B, E}</td>
<td>3</td>
</tr>
<tr>
<td>{C, E}</td>
<td>2</td>
</tr>
</tbody>
</table>
Apriori: Implementation Tricks

- How to generate candidates?
  - Step 1: self-joining $F_k$
  - Step 2: pruning
Apriori: Implementation Tricks

- How to generate candidates?
  - Step 1: self-joining \( F_k \)
  - Step 2: pruning

- Example of candidate-generation
  - \( F_3 = \{abc, abd, acd, ace, bcd\} \)
  - Self-joining: \( F_3 * F_3 \)
    - \( abcd \) from \( abc \) and \( abd \)
    - \( acde \) from \( acd \) and \( ace \)
Apriori: Implementation Tricks

- How to generate candidates?
  - Step 1: self-joining $F_k$
  - Step 2: pruning
- Example of candidate-generation
  - $F_3 = \{abc, abd, acd, ace, bcd\}$
  - Self-joining: $F_3 \times F_3$
    - $abcd$ from $abc$ and $abd$
    - $acde$ from $acd$ and $ace$
  - Pruning:
    - $acde$ is removed because $ade$ is not in $F_3$
  - $C_4 = \{abcd\}$
Suppose the items in $F_{k-1}$ are listed in an order.

**Step 1: self-joining $F_{k-1}$**

Insert into $C_k$

```
select p.item_1, p.item_2, ..., p.item_{k-1}, q.item_{k-1}
from $F_{k-1}$ as p, $F_{k-1}$ as q
where p.item_1 = q.item_1, ..., p.item_{k-2} = q.item_{k-2}, p.item_{k-1} < q.item_{k-1}
```

**Step 2: pruning**

For all itemsets $c$ in $C_k$ do

For all $(k-1)$-subsets $s$ of $c$ do

- If $(s$ is not in $F_{k-1}$) then delete $c$ from $C_k$
Apriori Adv/Disadv

- **Advantages:**
  - Uses large itemset property
  - Easily parallelized
  - Easy to implement

- **Disadvantages:**
  - Assumes transaction database is memory resident
  - Requires up to m database scans
Classification based on Association Rules (CBA)

- **Why?**
  - Can effectively uncover the correlation structure in data
  - AR are typically quite scalable in practice
  - Rules are often very intuitive
    - Hence classifier built on intuitive rules is easier to interpret

- **When to use?**
  - On large dynamic datasets where class labels are available and the correlation structure is unknown.
  - Multi-class categorization problems
  - E.g. Web/Text Categorization, Network Intrusion Detection
Example: Text categorization

- **Input**
  - `<feature vector> <class label(s)>`
  - `<feature vector> = w1,…,wN`
  - `<class label(s)> = c1,…,cM`

- **Run AR with minsup and minconf**
  - Prune rules of form
    - $w_1 \rightarrow w_2, [w_1,c_2] \rightarrow c_3$ etc.
  - Keep only rules satisfying the constraining
    - $W \rightarrow C$ (LHS only composed of $w_1,…,w_N$ and RHS only composed of $c_1,…,c_M$)
- Order remaining rules
  - By confidence
    - 100%
      - R1: W1 → C1 (support 40%)
      - R2: W4 → C2 (support 60%)
    - 95%
      - R3: W3 → C2 (support 30%)
      - R4: W5 → C4 (support 70%)
  - And within each confidence level by support
    - Ordering R2, R1, R4, R3
CBA: Text Categorization (cont.)

- Take training data and evaluate the predictive ability of each rule, prune away rules that are subsumed by superior rules
  - T1: W1 W5 C1,C4
  - T2: W2 W4 C2 Note: only subset
  - T3: W3 W4 C2 of transactions
  - T4: W5 W8 C4 in training data
  - T5: W9 C2
    - Rule R3 would be pruned in this example if it is always subsumed by Rule R2

- For remaining transactions pick most dominant class as default
  - T5 is not covered, so C2 is picked in this example
Formal Concepts of Model

- Given two rules \( r_i \) and \( r_j \), define: \( r_i \succ r_j \) if
  - The confidence of \( r_i \) is greater than that of \( r_j \), or
  - Their confidences are the same, but the support of \( r_i \) is greater than that of \( r_j \), or
  - Both the confidences and supports are the same, but \( r_i \) is generated earlier than \( r_j \).

- Our classifier model is of the following format:
  \(<r_1, r_2, ..., r_n, \text{default\_class}>,\>
  where \( r_i \in R, r_a \succ r_b \) if \( b > a \)

- Other models possible
  - Sort by length of antecedent
Using the CBA model to classify

- For a new transaction
  - W1, W3, W5

- Pick the k-most confident rules that apply (using the precedence ordering established in the baseline model)

- The resulting classes are the predictions for this transaction
  - If $k = 1$ you would pick C1
  - If $k = 2$ you would pick C1, C2 (multi-class)

- Similarly if W9, W10 you would pick C2 (default)

- Accuracy measurements as before (Classification Error)
CBA: Procedural Steps

- Preprocessing, Training and Testing data split

- Compute AR on Training data
  - Keep only rules of form X → C
    - C is class label itemset and X is feature itemset

- Order AR
  - According to confidence
  - According to support (at each confidence level)

- Prune away rules that lack sufficient predictive ability on Training data (starting top-down)
  - Rule subsumption

- For data that is not predictable pick most dominant class as default class

- Test on testing data and report accuracy
Backup Slides
Mining Frequent Patterns, Association and Correlations: Basic Concepts and Methods

- Basic Concepts
- Efficient Pattern Mining Methods
- Pattern Evaluation
- Summary
Summary

- **Basic Concepts**
  - What Is Pattern Discovery? Why Is It Important?
  - Basic Concepts: Frequent Patterns and Association Rules
  - Compressed Representation: Closed Patterns and Max-Patterns

- **Efficient Pattern Mining Methods**
  - The Downward Closure Property of Frequent Patterns
  - The Apriori Algorithm
  - Extensions or Improvements of Apriori
  - Mining Frequent Patterns by Exploring Vertical Data Format
  - FPGrowth: A Frequent Pattern-Growth Approach
  - Mining Closed Patterns

- **Pattern Evaluation**
  - Interestingness Measures in Pattern Mining
  - Interestingness Measures: Lift and $\chi^2$
  - Null-Invariant Measures
  - Comparison of Interestingness Measures
Recommended Readings (Basic Concepts)

- R. Agrawal, T. Imielinski, and A. Swami, “Mining association rules between sets of items in large databases”, in Proc. of SIGMOD'93

- R. J. Bayardo, “Efficiently mining long patterns from databases”, in Proc. of SIGMOD'98


Recommended Readings
(Efficient Pattern Mining Methods)

- J. Han, J. Pei, and Y. Yin, “Mining frequent patterns without candidate generation”, SIGMOD’00
- M. J. Zaki and Hsiao, “CHARM: An Efficient Algorithm for Closed Itemset Mining”, SDM’02
- J. Wang, J. Han, and J. Pei, “CLOSEST+: Searching for the Best Strategies for Mining Frequent Closed Itemsets”, KDD’03
- C. C. Aggarwal, M.A., Bhuiyan, M. A. Hasan, “Frequent Pattern Mining Algorithms: A Survey”, in Aggarwal and Han (eds.): Frequent Pattern Mining, Springer, 2014
Recommended Readings (Pattern Evaluation)

- E. Omiecinski. Alternative Interest Measures for Mining Associations. TKDE’03
- P.-N. Tan, V. Kumar, and J. Srivastava. Selecting the Right Interestingness Measure for Association Patterns. KDD'02
- T. Wu, Y. Chen and J. Han, Re-Examination of Interestingness Measures in Pattern Mining: A Unified Framework, Data Mining and Knowledge Discovery, 21(3):371-397, 2010