Scalable Frequent Itemset 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
- FP-Growth: A Frequent Pattern-Growth Approach
- Mining Closed Patterns

FP-Growth: Mining Frequent Patterns by Pattern Growth

- Idea: Frequent pattern growth (FP-Growth)
  - Find frequent single items and partition the database based on each such item
  - Recursively grow frequent patterns by doing the above for each partitioned database (also called conditional database)
  - To facilitate efficient processing, an efficient data structure, FP-tree, can be constructed
- Mining becomes
  - Recursively construct and mine (conditional) FP-trees
  - Until the resulting FP-tree is empty, or until it contains only one path—single path will generate all the combinations of its sub-paths, each of which is a frequent pattern
Construct FP-tree from a Transaction Database

<table>
<thead>
<tr>
<th>TID</th>
<th>Items bought (ordered) frequent items</th>
<th>100</th>
<th>{f, a, c, d, g, i, m, p}</th>
<th>{f, c, a, m, p}</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>{a, b, c, f, l, m, o}</td>
<td>{f, c, a, b, m}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>{b, f, h, j, o, w}</td>
<td>{f, b}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>400</td>
<td>{b, c, k, s, p}</td>
<td>{c, b, p}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>500</td>
<td>{a, f, c, e, l, p, m, n}</td>
<td>{f, c, a, m, p}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. Scan DB once, find frequent 1-itemset (single item pattern)
2. Sort frequent items in frequency descending order, f-list
3. Scan DB again, construct FP-tree

F-list = f-c-a-b-m-p

Divide and Conquer Based on Patterns and Data

- Pattern mining can be partitioned according to current patterns
  - Patterns containing p: p’s conditional database: \(fca:2, cb:1\)
  - Patterns having m but no p: m’s conditional database: \(fca:2, fcab:1\)
  - ....... .......
- \(p\)'s conditional pattern base: transformed prefix paths of item \(p\)

<table>
<thead>
<tr>
<th>Item frequency</th>
<th>head</th>
</tr>
</thead>
<tbody>
<tr>
<td>f</td>
<td>4</td>
</tr>
<tr>
<td>c</td>
<td>4</td>
</tr>
<tr>
<td>a</td>
<td>3</td>
</tr>
<tr>
<td>b</td>
<td>3</td>
</tr>
<tr>
<td>m</td>
<td>3</td>
</tr>
<tr>
<td>p</td>
<td>3</td>
</tr>
</tbody>
</table>

Conditional pattern bases

<table>
<thead>
<tr>
<th>item</th>
<th>cond. pattern base</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>f:3</td>
</tr>
<tr>
<td>a</td>
<td>fc:3</td>
</tr>
<tr>
<td>b</td>
<td>fca:1, f:1, c:1</td>
</tr>
<tr>
<td>m</td>
<td>fca:2, fcab:1</td>
</tr>
<tr>
<td>p</td>
<td>fca:2, cb:1</td>
</tr>
</tbody>
</table>
From Conditional Pattern-bases to Conditional FP-trees

- For each conditional pattern-base
  - Accumulate the count for each item in the base
  - Construct the conditional FP-tree for the frequent items of the conditional pattern base

```
Header Table
Item frequency head
f 4
 c 4
 a 3
 b 3
 m 3
 p 3
```

```
m-conditional pattern base: 
  fca:2, fab:1
```

```
All frequent patterns related to m:
  \{\}
  \{m, f, \}
  \{m, f, c, \}
  \{m, f, c, a, \}
```

```
m-conditional FP-tree: min_sup = 3
```

Mine Each Conditional Pattern-Base Recursively

```
Conditional pattern bases

<table>
<thead>
<tr>
<th>item \ cond. pattern base</th>
<th>min_support = 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>f:3</td>
</tr>
<tr>
<td>a</td>
<td>fc:3</td>
</tr>
<tr>
<td>b</td>
<td>fca:1, f:1, c:1</td>
</tr>
<tr>
<td>m</td>
<td>fca:2, fcab:1</td>
</tr>
<tr>
<td>p</td>
<td>fcam:2, cb:1</td>
</tr>
</tbody>
</table>
```

```
For each conditional pattern-base
  - Mine single-item patterns
  - Construct its FP-tree & mine it
```

```
p-conditional PB: fcam:2, cb:1 \rightarrow c: 3
```

```
m-conditional PB: fca:2, fcab:1 \rightarrow fca: 3
```

```
b-conditional PB: fca:1, f:1, c:1 \rightarrow \phi
```

Actually, for single branch FP-tree, all frequent patterns can be generated in one shot

```
m: 3
  fm: 3, cm: 3, am: 3
  fcm: 3, fam:3, cam: 3
  fcam: 3
```
A Special Case: Single Prefix Path in FP-tree

- Suppose a (conditional) FP-tree $T$ has a shared single prefix-path $P$
- Mining can be decomposed into two parts
  - Reduction of the single prefix path into one node
  - Concatenation of the mining results of the two parts

$$r_1 = \begin{array}{c}
\{\} \\
{a_1:n_1} \\
{a_2:n_2} \\
{a_3:n_3}
\end{array} + \begin{array}{c}
{b_1:m_1} \\
{C_1:k_1} \\
{C_2:k_2} \\
{C_3:k_3}
\end{array}$$

The Apriori Algorithm—An Example

- $\text{minsup} = 2$
- 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>

$C_1$ (1st scan)

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

$L_1$

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

$C_2$ (2nd scan)

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

$C_3$ (3rd scan)

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

$L_2$ (2nd scan)

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

$L_3$ (3rd scan)

- $\{B, C, E\}$: 2
Benefits of the FP-tree Structure

- **Completeness**
  - Preserve complete information for frequent pattern mining
  - Never break a long pattern of any transaction

- **Compactness**
  - *Reduce irrelevant info* — infrequent items are gone
  - *Items in frequency descending order*: the more frequently occurring, the more likely to be shared
  - *Never be larger than the original database* (if not counting: node-links and the *count* field)

Scaling FP-growth by Database Projection

- What if FP-tree cannot fit in memory? — DB projection
  - Project the DB based on patterns
  - Construct & mine FP-tree for each projected DB

- **Parallel projection vs. partition projection**
  - **Parallel projection**: Project the DB on each frequent item
    - Space costly, all partitions can be processed in parallel
  - **Partition projection**: Partition the DB in order
    - Passing the unprocessed parts to subsequent partitions

---

### Diagram

**Trans. DB**

- $f_2, f_4, f_5, g, h$
- $f_3, f_4, i, j$
- $f_2, f_4, k$
- $f_1, f_3, h$
- ...

**Parallel projection**

- $f_1, f_2, f_3, f_4, f_5$
- $f_1, f_2, f_3, f_4, f_5$
- $f_1, f_2, f_3, f_4, f_5$
- $f_1, f_2, f_3, f_4, f_5$
- $f_1, f_2, f_3, f_4, f_5$

**Partition projection**

- $f_1, f_2, f_3, f_4, f_5$
- $f_1, f_2, f_3, f_4, f_5$
- $f_1, f_2, f_3, f_4, f_5$
- $f_1, f_2, f_3, f_4, f_5$
- $f_1, f_2, f_3, f_4, f_5$

$g$ will be projected to $f_1, f_2, f_3, f_4, f_5$ only when processing $f_1, f_2, f_3, f_4, f_5$
FP-Growth vs. Apriori: Scalability With the Support Threshold

Data set T25I20D10K

IBM Synth.: Average Transaction length: 25 Itemset length: 10K #Transactions: 20K

Run time (sec.)

Support threshold (%)

D1 FP-growth runtime
D1 Apriori runtime

FP-Growth vs. Tree-Projection: Scalability with the Support Threshold

Data set T25I20D100K

D2 FP-growth
D2 TreeProjection
Advantages of the Pattern Growth Approach

- Divide-and-conquer:
  - Decompose both the mining task and DB according to the frequent patterns obtained so far
  - Lead to focused search of smaller databases

- Other factors
  - No candidate generation, no candidate test
  - Compressed database: FP-tree structure
  - No repeated scan of entire database
  - Basic operations: counting local freq items and building sub FP-tree, no pattern search and matching
  - A good open-source implementation and refinement of FP-Growth
    - FP-Growth+ (Grahne and J. Zhu, FIMI’03)

Extension of Pattern Growth Mining Methodology

- Mining closed frequent itemsets and max-patterns
  - CLOSET (DMKD’00), FPclose, and FPMax (Grahne & Zhu, Fimi’03)

- Mining sequential patterns
  - PrefixSpan (ICDE’01), CloSpan (SDM’03), BIDE (ICDE’04)

- Mining graph patterns
  - gSpan (ICDM’02), CloseGraph (KDD’03)

- Constraint-based mining of frequent patterns
  - Convertible constraints (ICDE’01), gPrune (PAKDD’03)

- Computing iceberg data cubes with complex measures
  - H-tree, H-cubing, and Star-cubing (SIGMOD’01, VLDB’03)

- Pattern-growth-based Clustering
  - MaPie (Pei, et al., ICDM’03)

- Pattern-Growth-Based Classification
  - Mining frequent and discriminative patterns (Cheng, et al, ICDE’07)
Scalable Frequent Itemset 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
- FP-Growth: A Frequent Pattern-Growth Approach
- Mining Closed Patterns

Closed Patterns and Max-Patterns

An itemset $X$ is a **closed pattern**

- if $X$ is frequent and
- there exists no super-pattern $Y \supset X$, with the same support as $X$

An itemset $X$ is a **max-pattern**

- if $X$ is frequent and
- there exists no frequent super-pattern $Y \supset X$

A Closed pattern is a lossless compression offreq. patterns

- Reducing the # of patterns and rules
CLOSET+: Mining Closed Itemsets by Pattern-Growth

- Efficient, direct mining of closed itemsets
- Ex. Itemset merging: If Y appears in every occurrence of X, then Y is merged with X
  - d-proj. db: \{acef, ac\} → acfd-proj. db: \{e\}
  - thus we get: acfd:2
- Many other tricks (but not detailed here), such as
  - Hybrid tree projection
  - Bottom-up physical tree-projection
  - Top-down pseudo tree-projection
  - Sub-itemset pruning
  - Item skipping
  - Efficient subset checking
- For details, see J. Wang, et al., “CLOSET+: ……”, KDD’03

MaxMiner: Mining Max-Patterns

- 1st scan: find frequent items
  - A, B, C, D, E
- 2nd scan: find support for
  - AB, AC, AD, AE, ABCDE
  - BC, BD, BE, BCDE
  - CD, CE, CDE, DE
  - Potential max-patterns
- Since BCDE is a max-pattern, no need to check BCD, BDE, CDE in later scan
- R. Bayardo. Efficiently mining long patterns from databases. SIGMOD’98
Chapter 6: Mining Frequent Patterns, Association and Correlations: Basic Concepts and Methods

- Basic Concepts
- Frequent Itemset Mining Methods
- Which Patterns Are Interesting?
  - Pattern Evaluation Methods
- Summary

How to Judge if a Rule/Pattern Is Interesting?

- Pattern-mining will generate a large set of patterns/rules
  - Not all the generated patterns/rules are interesting
- Interestingness measures: Objective vs. subjective
  - Objective interestingness measures
    - Support, confidence, correlation, ...
  - Subjective interestingness measures: One man’s trash could be another man’s treasure
    - Query-based: Relevant to a user’s particular request
    - Against one’s knowledge-base: unexpected, freshness, timeliness
    - Visualization tools: Multi-dimensional, interactive examination
Limitation of the Support-Confidence Framework

- Are \( s \) and \( c \) interesting in association rules: “\( A \Rightarrow B \) [\( s, c \)]?”
- Example: Suppose one school may have the following statistics on # of students who may play basketball and/or eat cereal:

<table>
<thead>
<tr>
<th></th>
<th>play-basketball</th>
<th>not play-basketball</th>
<th>sum (row)</th>
</tr>
</thead>
<tbody>
<tr>
<td>eat-cereal</td>
<td>400</td>
<td>350</td>
<td>750</td>
</tr>
<tr>
<td>not eat-cereal</td>
<td>200</td>
<td>50</td>
<td>250</td>
</tr>
<tr>
<td>sum(col.)</td>
<td>600</td>
<td>400</td>
<td>1000</td>
</tr>
</tbody>
</table>

Association rule mining may generate the following:
- \( \text{play-basketball} \Rightarrow \text{eat-cereal} \) [40%, 66.7%] (higher \( s \) & \( c \))
- But this strong association rule is misleading: The overall % of students eating cereal is 75% > 66.7%, a more telling rule:
- \( \neg \text{play-basketball} \Rightarrow \text{eat-cereal} \) [35%, 87.5%] (high \( s \) & \( c \))

Interestingness Measure: Lift

- Measure of dependent/correlated events: \( \text{lift} \)

\[
\text{lift} (B, C) = \frac{c(B \Rightarrow C)}{s(C)} = \frac{s(B \cup C)}{s(B) \times s(C)}
\]
- \( \text{lift}(B, C) \) may tell how \( B \) and \( C \) are correlated
- \( \text{lift}(B, C) = 1 \): \( B \) and \( C \) are independent
- \( > 1 \): positively correlated
- \( < 1 \): negatively correlated
- For our example,

\[
\text{lift} (B, C) = \frac{\frac{400}{1000}}{\frac{600}{1000} \times \frac{750}{1000}} = 0.89
\]

\[
\text{lift} (B, \neg C) = \frac{\frac{200}{1000}}{\frac{600}{1000} \times \frac{250}{1000}} = 1.33
\]
- Thus, \( B \) and \( C \) are negatively correlated since \( \text{lift}(B, C) < 1 \);
- \( B \) and \( \neg C \) are positively correlated since \( \text{lift}(B, \neg C) > 1 \)
Interestingness Measure: $\chi^2$

- Another measure to test correlated events: $\chi^2$

$$\chi^2 = \sum \frac{(Observed - Expected)^2}{Expected}$$

- General rules
  - $\chi^2 = 0$: independent
  - $\chi^2 > 0$: correlated, either positive or negative, so it needs additional test to determine which correlation

- Now,

$$\chi^2 = \frac{(400 - 450)^2}{450} + \frac{(350 - 300)^2}{300} + \frac{(200 - 150)^2}{150} + \frac{(50 - 100)^2}{100} = 55.56$$

- $\chi^2$ shows B and C are negatively correlated since the expected value is 450 ($= 600 \times 750/1000$) but the observed is lower, only 400
- $\chi^2$ is also more telling than the support-confidence framework

Lift and $\chi^2$: Are They Always Good Measures?

- Null transactions:
  - Transactions that contain neither B nor C

- Let’s examine the dataset $D$

  - BC (100) is much rarer than B~C (1000) and ~BC (1000), but there are many ~B~C (100000)
  - In these transactions it is unlikely that B & C will happen together!

- But, Lift(B, C) = 8.44 >> 1
  (Lift shows B and C are strongly positively correlated!)

- $\chi^2 = 670$: Observed(BC) >> expected value (11.85)
  - Too many null transactions may “spoil the soup”!
Interestingness Measures & Null-Invariance

- **Null invariance**: Value does not change with the # of null-transactions
- A few interesting measures: Some are null invariant

<table>
<thead>
<tr>
<th>Measure</th>
<th>Definition</th>
<th>Range</th>
<th>Null-Invariant</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\chi^2(A, B)$</td>
<td>$\sum_{i,j=0,1} \frac{(c(i,j) - e(i,j))^2}{e(i,j)}$</td>
<td>$[0, \infty]$</td>
<td>No</td>
</tr>
<tr>
<td>Lift(A, B)</td>
<td>$\frac{s(A \times B)}{\sum (c(A), c(B))}$</td>
<td>$[0, \infty]$</td>
<td>No</td>
</tr>
<tr>
<td>AllConf(A, B)</td>
<td>$\frac{s(A \times B)}{\max (c(A), c(B))}$</td>
<td>$[0, 1]$</td>
<td>Yes</td>
</tr>
<tr>
<td>Jaccard(A, B)</td>
<td>$\frac{s(A \times B)}{\sqrt{c(A) + c(B) - s(A \times B)}}$</td>
<td>$[0, 1]$</td>
<td>Yes</td>
</tr>
<tr>
<td>Cosine(A, B)</td>
<td>$\frac{s(A \times B)}{\sqrt{c(A) \times c(B)}}$</td>
<td>$[0, 1]$</td>
<td>Yes</td>
</tr>
<tr>
<td>Kulczynski(A, B)</td>
<td>$\frac{1}{2} \left( \frac{s(A \times B)}{c(A)} + \frac{s(A \times B)}{c(B)} \right)$</td>
<td>$[0, 1]$</td>
<td>Yes</td>
</tr>
<tr>
<td>MaxConf(A, B)</td>
<td>$\max \left{ \frac{s(A \times B)}{c(A) \times c(B)} \right}$</td>
<td>$[0, 1]$</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Null Invariance: An Important Property

- Why is null invariance crucial for the analysis of massive transaction data?
  - Many transactions may contain neither milk nor coffee!

- Null transactions w.r.t. m and c

<table>
<thead>
<tr>
<th>milk vs. coffee contingency table</th>
</tr>
</thead>
<tbody>
<tr>
<td>coffee</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>~coffee</td>
</tr>
</tbody>
</table>

Assignments: Check the interestingness measures in the table.

- Lift and $\chi^2$ are not null-invariant: not good to evaluate data that contain too many or too few null transactions!
- Many measures are not null-invariant!
Comparison of Null-Invariant Measures

- Not all null-invariant measures are created equal
- Which one is better?
  - $D_4 - D_6$ differentiate the null-invariant measures
  - Kulc (Kulczynski 1927) holds firm and is in balance of both directional implications

<table>
<thead>
<tr>
<th>Data set</th>
<th>$m_e$</th>
<th>$\sim m_e$</th>
<th>$m_{\sim e}$</th>
<th>$\sim m_{\sim e}$</th>
<th>AllConf</th>
<th>Jaccard</th>
<th>Cosine</th>
<th>Kulc</th>
<th>MaxConf</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_1$</td>
<td>10,000</td>
<td>1,000</td>
<td>100,000</td>
<td>0.91</td>
<td>0.83</td>
<td>0.91</td>
<td>0.91</td>
<td>0.91</td>
<td></td>
</tr>
<tr>
<td>$D_2$</td>
<td>10,000</td>
<td>1,000</td>
<td>100,000</td>
<td>0.91</td>
<td>0.83</td>
<td>0.91</td>
<td>0.91</td>
<td>0.91</td>
<td></td>
</tr>
<tr>
<td>$D_3$</td>
<td>100</td>
<td>1,000</td>
<td>100,000</td>
<td>0.09</td>
<td>0.09</td>
<td>0.09</td>
<td>0.09</td>
<td>0.09</td>
<td></td>
</tr>
<tr>
<td>$D_4$</td>
<td>1,000</td>
<td>100</td>
<td>100,000</td>
<td>0.09</td>
<td>0.09</td>
<td>0.29</td>
<td>0.5</td>
<td>0.91</td>
<td></td>
</tr>
<tr>
<td>$D_5$</td>
<td>1,000</td>
<td>100</td>
<td>100,000</td>
<td>0.09</td>
<td>0.09</td>
<td>0.10</td>
<td>0.5</td>
<td>0.99</td>
<td></td>
</tr>
</tbody>
</table>

All 5 are null-invariant

Subtle: They disagree on those cases

Analysis of DBLP Coauthor Relationships

Recent DB conferences, removing balanced associations, low sup, etc.

- Which pairs of authors are strongly related?
  - Use Kulc to find Advisor-advisee, close collaborators
Imbalance Ratio with Kulczynski Measure

- IR (Imbalance Ratio): measure the imbalance of two itemsets A and B in rule implications:
  \[
  IR(A, B) = \frac{|s(A) - s(B)|}{s(A) + s(B) - s(A \cup B)}
  \]

- Kulczynski and Imbalance Ratio (IR) together present a clear picture for all the three datasets D_4 through D_6:
  - D_4 is neutral & balanced; D_5 is neutral but imbalanced
  - D_6 is neutral but very imbalanced

<table>
<thead>
<tr>
<th>Data set</th>
<th>me</th>
<th>me-</th>
<th>m-c</th>
<th>m-c-</th>
<th>Jaccard</th>
<th>Cosine</th>
<th>Kulc</th>
<th>IR</th>
</tr>
</thead>
<tbody>
<tr>
<td>D_1</td>
<td>100</td>
<td>1</td>
<td>1</td>
<td>100</td>
<td>0.83</td>
<td>0.91</td>
<td>0.91</td>
<td>0</td>
</tr>
<tr>
<td>D_2</td>
<td>100</td>
<td>1</td>
<td>1</td>
<td>100</td>
<td>0.81</td>
<td>0.91</td>
<td>0.91</td>
<td>0</td>
</tr>
<tr>
<td>D_3</td>
<td>100</td>
<td>1</td>
<td>1</td>
<td>100</td>
<td>0.05</td>
<td>0.09</td>
<td>0.09</td>
<td>0</td>
</tr>
<tr>
<td>D_4</td>
<td>1</td>
<td>100</td>
<td>1</td>
<td>100</td>
<td>0.33</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
</tr>
<tr>
<td>D_5</td>
<td>1</td>
<td>100</td>
<td>1</td>
<td>100</td>
<td>0.06</td>
<td>0.29</td>
<td>0.29</td>
<td>0.9</td>
</tr>
<tr>
<td>D_6</td>
<td>1</td>
<td>10</td>
<td>100</td>
<td>100</td>
<td>0.01</td>
<td>0.10</td>
<td>0.10</td>
<td>0.9</td>
</tr>
</tbody>
</table>

What Measures to Choose for Effective Pattern Evaluation?

- Null value cases are predominant in many large datasets
  - Neither milk nor coffee is in most of the baskets; neither Mike nor Jim is an author in most of the papers; ......

- Null-invariance is an important property

- Lift, \( \chi^2 \) and cosine are good measures if null transactions are not predominant
  - Otherwise, Kulczynski + Imbalance Ratio should be used to judge the interestingness of a pattern

Optional Exercise: Mining research collaborations from research bibliographic data

- Find a group of frequent collaborators from research bibliographic data (e.g., DBLP)
- Can you find the likely advisor-advisee relationship and during which years such a relationship happened?
- Ref.: C. Wang, J. Han, Y. Jia, J. Tang, D. Zhang, Y. Yu, and J. Guo, "Mining Advisor-Advisee Relationships from Research Publication Networks", KDD’10
Chapter 5: Mining Frequent Patterns, Association and Correlations: Basic Concepts and Methods

- Basic Concepts
- Frequent Itemset Mining Methods
- Which Patterns Are Interesting?—Pattern Evaluation Methods
- Summary

Summary: Mining Frequent Patterns, Association and Correlations

- Basic Concepts:
  - Frequent Patterns, Association Rules, Closed Patterns and Max-Patterns
- Frequent Itemset Mining Methods
  - The Downward Closure Property and 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
- Which Patterns Are Interesting?—Pattern Evaluation Methods
  - Interestingness Measures: Lift and $\chi^2$
  - Null-Invariant Measures
  - Comparison of Interestingness Measures
Ref: Basic Concepts of Frequent Pattern Mining

- **(Sequential pattern)** R. Agrawal and R. Srikant. Mining sequential patterns. ICDE’95
- J. Han, H. Cheng, D. Xin, and X. Yan, "Frequent Pattern Mining: Current Status and Future Directions", Data Mining and Knowledge Discovery, 15(1): 55-86, 2007

Ref: Apriori and Its Improvements

References (II) Efficient Pattern Mining Methods

- J. S. Park, M. S. Chen, and P. S. Yu, "An effective hash-based algorithm for mining association rules", SIGMOD'95
- S. Sarawagi, S. Thomas, and R. Agrawal, "Integrating association rule mining with relational database systems: Alternatives and implications", SIGMOD'98
- 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, "CLOSET+: 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

References (III) Pattern Evaluation

- C. C. Aggarwal and P. S. Yu. A New Framework for Itemset Generation. PODS'98
- 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