CS 412 Intro. to Data Mining

Chapter 6. Mining Frequent Patterns, Association and Correlations: Basic Concepts and Methods

Jiawei Han, Computer Science, Univ. Illinois at Urbana-Champaign, 2017

Chapter 6: Mining Frequent Patterns, Association and Correlations: Basic Concepts and Methods

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

What Is Pattern Discovery?

- 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
- Motivation examples: สามารถ search แล้วเจอเลย
  - 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?

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

สินค้าที่ซื้อร่วมกัน (ชุด)

- Itemset: A set of one or more items
- k-itemset: X = $\{x_1, \ldots, x_k\}$
  - มีสมาชิก 3 ตัว
  - Ex. {Beer, Nuts, Diaper} is a 3-itemset
- แค่นับ
- (*absolute*) *support* (*count*) of X, sup{X}: Frequency or the number of occurrences of an itemset X
  - Ex. sup{Beer} = 3 มีกี่ transaction ที่มี beer อยู่
  - Ex. sup{Diaper} = 4
  - 2 item set
  - Ex. sup{Beer, Diaper} = 3
  - Ex. sup{Beer, Eggs} = 1

จำนวนของ transactions ที่สนับสนุนความคิดเรา

Transaction ID

	Tid	Items bought
1	10	Beer, Nuts, Diaper
2	20	Beer, Coffee, Diaper
3	30	Beer, Diaper, Eggs
4	40	Nuts, Eggs, Milk
5	50	Nuts, Coffee, Diaper, Eggs, Milk

สัดส่วนที่ซัพพอร์ท

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

Basic Concepts: Frequent Itemsets (Patterns)

ย่อย!

- An itemset (or a pattern) X is *frequent* if the support of X is no less than a *minsup* threshold σ เกิน threshold σ
- Let σ = *50%* (σ: *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

Tid	Items bought
10	Beer, Nuts, Diaper
20	Beer, Coffee, Diaper
30	Beer, Diaper, Eggs
40	Nuts, Eggs, Milk
50	Nuts, Coffee, Diaper, Eggs, Milk

- 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

Coffee : 2/5 (40%) ไม่ถึง threshold

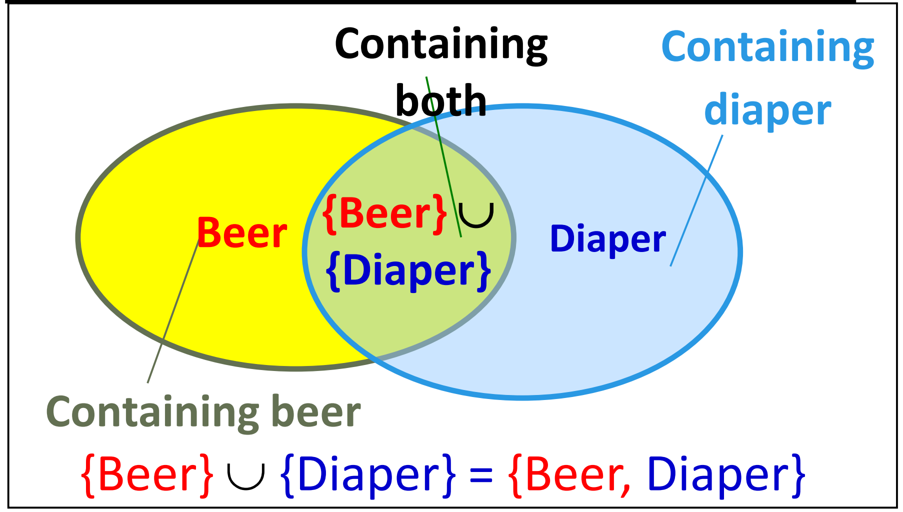
From Frequent Itemsets to Association Rules

- Comparing with itemsets, rules can be more telling
  - Ex. *Diaper* → *Beer* ซื้อ Diaper จะไปซื้อ beer ด้วย
    - *Buying diapers may likely lead to buying beers*

ก็หาในเซตคู่ที่มี support

- 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 $X \cup Y$ → ซื้อของพร้อมกัน
    - Ex. s{Diaper, Beer} = 3/5 = 0.6 (i.e., 60%)
  - Confidence, *c: The conditional probability* that a transaction containing X also contains *Y*
    - Calculation: $c = \sup(X \cup Y) / \sup(X)$
    - Ex. c = sup{Diaper, Beer}/sup{Diaper} = ¾ = 0.75

Tid	Items bought
10	Beer, Nuts, Diaper
20	Beer, Coffee, Diaper
30	Beer, Diaper, Eggs
40	Nuts, Eggs, Milk
50	Nuts, Coffee, Diaper, Eggs, Milk



Note: $X \cup Y$: the union of two itemsets

- The set contains both X and Y

Mining Frequent Itemsets and Association Rules

- **Association rule mining** กำหนด minsup มองเฉพาะ itemset ที่ถึง
  - Given two thresholds: *minsup*, *minconf*
  - Find all of the rules, $X \rightarrow Y$ (s, c)
    - such that, s ≥ *minsup* and *c* ≥ *minconf*

minsup ได้เป็น 30%

item เล็ก ไม่ผ่าน
item ใหญ่ ไม่ต้องดู

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

- Let *minconf = 50%* sub(beer, Diaper) / sub(beer)
  - *Beer* → *Diaper* (60%, 100%) sup confident
  - *Diaper* → *Beer* (60%, 75%) support confident

(Q: Are these all rules?)

Tid	Items bought
10	Beer, Nuts, Diaper
20	Beer, Coffee, Diaper
30	Beer, Diaper, Eggs
40	Nuts, Eggs, Milk
50	Nuts, Coffee, Diaper, Eggs, Milk

- **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: $\{a_1, …, a_{50}\}$; T_2: $\{a_1, …, a_{100}\}$
  - Assuming (absolute) *minsup* = 1
  - Let's have a try

1-itemsets: $\{a_1\}$: 2, $\{a_2\}$: 2, …, $\{a_{50}\}$: 2, $\{a_{51}\}$: 1, …, $\{a_{100}\}$: 1,

2-itemsets: $\{a_1, a_2\}$: 2, …, $\{a_1, a_{50}\}$: 2, $\{a_1, a_{51}\}$: 1 …, …, $\{a_{99}, a_{100}\}$: 1,

…, …, …, …

99-itemsets: $\{a_1, a_2, …, a_{99}\}$: 1, …, $\{a_2, a_3, …, a_{100}\}$: 1

100-itemset: $\{a_1, a_2, …, a_{100}\}$: 1

A too huge set for any one to compute or store!

- The total number of frequent itemsets:

$$\binom{100}{1} + \binom{100}{2} + \binom{100}{3} + \cdots + \binom{100}{100} = 2^{100} - 1$$

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 ⊃ X, *with the same support* as X
  - Let Transaction DB TDB_1: T_1: $\{a_1, …, a_{50}\}$; T_2: $\{a_1, …, a_{100}\}$
  - Suppose *minsup* = 1. How many closed patterns does TDB_1 contain?
    - Two: P_1: "$\{a_1, …, a_{50}\}$: 2"; P_2: "$\{a_1, …, a_{100}\}$: 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: "$\{a_2, …, a_{40}\}$: 2", "$\{a_5, a_{51}\}$: 1"

Expressing Patterns in Compressed Form: Max-Patterns

- Solution 2: **Max-patterns**: A pattern X is a max-pattern if X is frequent and there exists no frequent super-pattern Y ͻ X
- Difference from close-patterns?
  - Do not care the real support of the sub-patterns of a max-pattern
  - Let Transaction DB TDB_1: T_1: $\{a_1, …, a_{50}\}$; T_2: $\{a_1, …, a_{100}\}$
  - Suppose *minsup* = 1. How many max-patterns does TDB_1 contain?
    - One: P: "$\{a_1, …, a_{100}\}$: 1"
- Max-pattern is a lossy compression!
  - We only know $\{a_1, …, a_{40}\}$ is frequent
  - But we do not know the real support of $\{a_1, …, a_{40}\}$, …, any more!
- Thus in many applications, mining close-patterns is more desirable than mining max-patterns

Apriori Pruning and Scalable Mining Methods

ตัดแต่ง

- 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)

ถ้าเกิด itemset ที่ เล็กกว่าไม่ผ่าน ก็จะ ตัด ทิ้งเลย

- 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

เขียนเป็นสัญลักษณ์

- Outline of Apriori (level-wise, candidate generation and test)
  - Initially, scan DB once to get frequent 1-itemset

    Data base
  - Repeat

    สร้างได้ยังไงนะ ?
    - 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 != $\emptyset$) **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 $\cup_k F_k$ // return F_k generated at each level

The Apriori Algorithm—An Example

minsup = 2

Database TDB

Tid	Items
10	A, C, D
20	B, C, E
30	A, B, C, E
40	B, E

1st scan $\rightarrow$ C_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{D}	1
{E}	3

ตัดทิ้ง

F_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{E}	3

จับคู่

2 item set

C_2

Itemset
{A, B}
{A, C}
{A, E}
{B, C}
{B, E}
{C, E}

2nd scan $\rightarrow$ C_2

Itemset	sup
{A, B}	1
{A, C}	2
{A, E}	1
{B, C}	2
{B, E}	3
{C, E}	2

ตัดทิ้ง

F_2

Itemset	sup
{A, C}	2
{B, C}	2
{B, E}	3
{C, E}	2

ดู 3 itemset

C_3

Itemset
{B, C, E}

3rd scan $\rightarrow$ F_3

Itemset	sup
{B, C, E}	2

หยุด! ต้องตอบ

1 item = 4

2 item = 4

3 item = 1