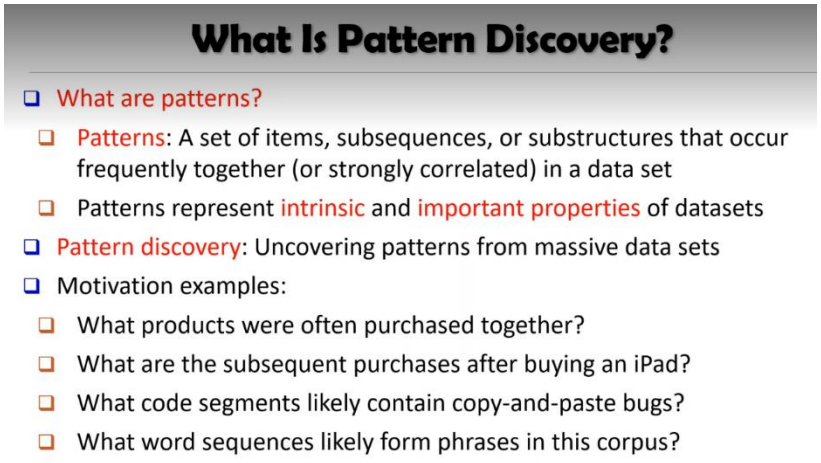
นางสาวอุมาพร คำภิชัย รหัสนักศึกษา 623020547-0
คณะวิทยาศาสตร์ สาขาสถิติ หลักสูตรสารสนเทศสถิติ ชั้นปีที่ 3

สรุป Chapter 6

- หาแพทเทินที่เกิดขึ้นซ้ำ ๆ



- pattern คือ ไอเทมหลาย ๆ อย่างรวมกัน ถูกซื้อร่วมกัน

- เอาไปใช้ประโยชน์ในเรื่องสินค้าไหนที่คนมักซื้อร่วมกัน ซื้อควบคู่กัน

- ลืมเปลี่ยนชื่อ อาจจะบัค ช่วยให้คนทำงานได้ง่ายขึ้น

- เดาคำศัพท์ที่จะค้นหาได้ เช่น แบบค้นหา ใน google

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, sup{X}: Frequency or the number of occurrences of an itemset X
  - Ex. sup{Beer} = 3
  - Ex. sup{Diaper} = 4
  - Ex. sup{Beer, Diaper} = 3
  - Ex. sup{Beer, Eggs} = 1

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

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

- K หมายถึงว่า เราสามารถเติมตัวเลขลงไปได้
- 3 ไอเทมเซ็ตที่มีสมาชิก 3 ตัว

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 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

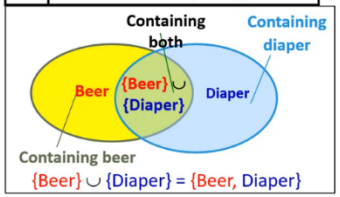
- Why do these itemsets (shown on the left) form the complete set of frequent *k*-itemsets (patterns) for any *k*?
- **Observation**: We may need an efficient method to mine a complete set of frequent patterns

- Thresholds เป็นการแบ่งว่า ถ้ามากกว่า 50 จะถือว่าบ่อย

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*
- How strong is this rule? (support, confidence)
  - Measuring association rules: *X → Y* (s, c)
    - Both *X* and *Y* are itemsets
  - Support, *s*: The probability that a transaction contains X ∪ Y
    - 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 ∪ 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 ∪ Y: the union of two itemsets
■ The set contains both X and Y

- บอกว่าถ้าซื้ออันนึงจะเข้าไปสู่การซื้ออีกอย่างนึง เหนี่ยวนำให้ไปซื้อสินค้าตัวอื่น

Mining Frequent Itemsets and Association Rules

- **Association rule mining**
  - Given two thresholds: *minsup, minconf*
  - Find all of the rules, *X → Y* (s, c)
    - such that, s ≥ *minsup* and *c* ≥ *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 → Diaper* (60%, 100%)
  - *Diaper → Beer* (60%, 75%)

(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

10

- กำหนด 2 thresholds ก็คือ minsup , minconf
- Minsup จะมองข้อมูลของเราทั้งหมดเป็นภาพใหญ่
- Minconf จะพิจารณาแค่ในไอเทมเซ็ตนั้น ว่าไอเทมคู่นั้นเกิดพร้อมกันบ่อยแค่ไหน

The Apriori Algorithm (Pseudo-Code)

C_k: Candidate itemset of size k

F_k : Frequent itemset of size k

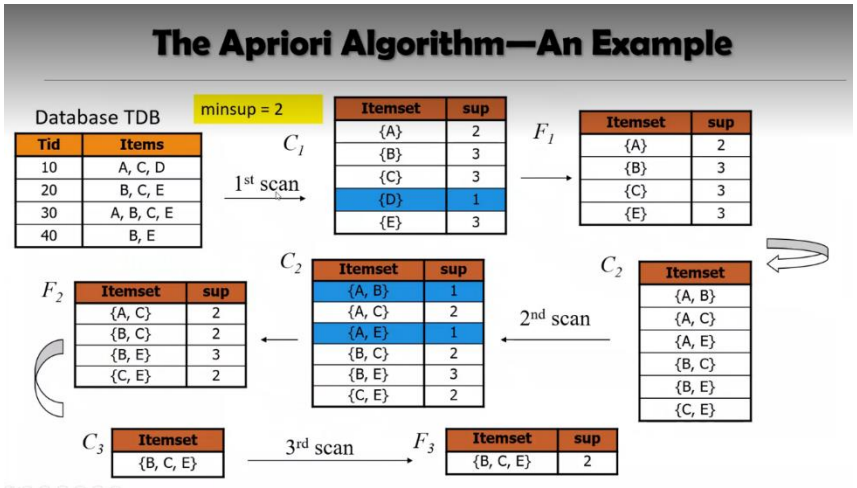
```
K := 1;
Fk := {frequent items};   // frequent 1-itemset
While (Fk != ∅) do {      // when Fk is non-empty
   Ck+1 := candidates generated from Fk;  // candidate generation
   Derive Fk+1 by counting candidates in Ck+1 with respect to TDB at minsup;
   k := k + 1
   }
return ∪k Fk              // return Fk generated at each level
```

- เขียนให้เราแปลงเป็นภาษาที่เราใช้ได้

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

- ขั้นตอนการหาไอเทมเซ็ต



- ขั้นตอนแรกหา 1 ไอเทมเซ็ตก่อน แล้วหา sup
- ตัด sup ไม่ถึง 2 จะถูกตัดทิ้ง