บทที่ 2
ทฤษฎีพื้นฐานที่ใช้ในการวิจัย และงานวิจัยที่เกี่ยวข้อง

การหากฎความสัมพันธ์เป็นการทำไมน์นิ่งอย่างหนึ่งที่ได้รับความสนใจในการทำงานวิจัย เพื่อค้นหาสารสนเทศที่ซ่อนอยู่ในฐานข้อมูล โดยเฉพาะในเรื่องที่เกี่ยวกับ market basket ทั้งนี้ เพื่อให้สามารถนำข้อมูลหรือความรู้ที่ได้จากการกฎความสัมพันธ์มาใช้ในด้านต่างๆ เช่น การวางแผนกลยุทธ์ทางการตลาด, ช่วยในการตัดสินใจหรือวางแผนเกี่ยวกับผลิตภัณฑ์ เป็นต้น

ในบทนี้จะกล่าวถึงทฤษฎีพื้นฐานต่างๆ ของการหากฎความสัมพันธ์ (Association rules) , การเพิ่มขยายกฎความสัมพันธ์ (Incremental Association rule) และงานวิจัยด้านต่างๆ ที่เกี่ยวข้องกับการหากฎความสัมพันธ์ของข้อมูลที่ปรากฏในฐานข้อมูลซึ่งมีรายละเอียดดังนี้

2.1 การไมน์นิ่งกฎความสัมพันธ์

กระบวนการไมน์นิ่งกฎความสัมพันธ์ (Association rules mining) ได้ถูกนำเสนอในปีค.ศ. 1993 [1] เพื่อใช้ในการค้นหาความสัมพันธ์ระหว่างรายการในแต่ละรายการหรือกลุ่มรายการที่ปรากฏขึ้นในฐานข้อมูลขนาดใหญ่ ความสัมพันธ์ที่ได้สามารถใช้ในการบอกลักษณะของข้อมูล หรือใช้ในการทำนายลักษณะของข้อมูลที่อาจเกิดขึ้นต่อไป โดยจะแสดงอยู่ในรูปของกฎความสัมพันธ์ (Association Rules)

กฎความสัมพันธ์ เป็นเทคนิคที่ใช้หากฎเกณฑ์ความสัมพันธที่ปรากฏขึ้นระหว่างข้อมูลทั้งหมดในฐานข้อมูล โดยกำหนดให้ I เป็นเซตของ item I โดย $I = I_1, I_2, \ldots, I_m$, D เป็นเซตของ transaction ในฐานข้อมูล ซึ่งแต่ละ transaction T เป็นเซตของ items โดย $T \subseteq I$ T แต่ละตัวจะสัมพันธ์กับตัวระบุ (identifier) ที่เรียกว่า TID (Transaction Identifier) ,A เป็นเซตของ items ใน transaction หนึ่ง ซึ่ง $A \subseteq T$ item ต่างๆ ที่ปรากฏในฐานข้อมูลจะถูกนำมาหาความสัมพันธ์ โดย item ที่จะถูกนำมาสร้างเป็นกฎความสัมพันธ์ได้จะต้องมีจำนวนของข้อมูลที่เกิดขึ้นมากกว่าหรือเท่ากับตัววัด 2 ตัวคือ minimum support และ minimum confidence

กฎความสัมพันธ์ที่ได้จะแสดงอยู่ในรูปของกฎ "ถ้า ... แล้ว..." (IF...THEN ...) ซึ่งในกฎหนึ่งๆ ประกอบด้วย 2 ส่วนคือ ส่วนที่ด้านซ้ายกฎหรือส่วนที่อยู่หลัง IF ที่ประกอบด้วยเงื่อนไขที่เป็นจริงหนึ่งหรือมากกว่าหนึ่งเงื่อนไข เรียกส่วนนี้ว่า Rule Body หรือ Antecedent หรือ Left-hand side และส่วนด้านขวาของกฎหรือส่วนที่อยู่หลัง THEN เป็นส่วนที่แสดงผลเมื่อเงื่อนไขที่ระบุในส่วนของ IF เป็นจริง เรียกส่วนนี้ว่า Rule head หรือ Consequent หรือ Right-hand side ตัวอย่างของกฎความสัมพันธ์เช่น IF A THEN B หรือ เขียนแทนด้วยสัญลักษณ์ $A \Rightarrow B$ โดยค่า $A \subset I$,

$B \subset I$ และ $A \cap B = \phi$ กฎ $A \Rightarrow B$ จะถูกจัดให้มีในเซต transaction D ด้วยค่า support s ซึ่งเป็นเปอร์เซ็นต์ของข้อมูล transaction ใน D ที่มีทั้ง A และ B ($A \cup B$) โดยจะเป็นค่าความน่าจะเป็นที่จะเกิด A และ B พร้อมกัน ($P(A \cup B)$) กฎ $A \Rightarrow B$ จะมีค่า confidence c ในเซตของข้อมูล transaction ใน D ถ้า c เป็นเปอร์เซ็นต์ของข้อมูล transaction ใน D ที่มี A แล้วจะมี B ด้วย ซึ่งเป็นเรื่องของความน่าจะเป็นแบบมีเงื่อนไข ($P(B \mid A)$) ดังสามารถสรุปได้ดังนี้

$\text{Support}(A \Rightarrow B) = P(A \cup B)$

$\text{Confidence}(A \Rightarrow B) = P(B \mid A)$

นิยามและความหมายของคำต่าง ๆ ที่ใช้ในการค้นหากฎความสัมพันธ์ได้แก่

1. ไอเทม (Item) คือข้อมูลแต่ละตัวที่ใช้ในการหากฎความสัมพันธ์ เช่น bread, cheese , shampoo, milk เป็นต้น
2. ไอเทมเซต (Itemset) คือ ความสัมพันธ์ของข้อมูลที่หาได้ Itemset ประกอบด้วย Item ที่มีความยาวแตกต่างกัน แทนขนาดความยาวของ itemset ได้ด้วย k (k= 1, 2 ,3,...,n) เรียกว่า k-itemsets เช่น 2-itemsets หมายถึง itemset ที่ประกอบด้วยสมาชิกของ item 2 ตัว เช่น {bread, cheese}, {milk, bread} , { milk , shampoo} เป็นต้น และ 3-itemsets หมายถึง itemset ที่ประกอบด้วยสมาชิกของ item 3 ตัว เช่น {milk, bread , shampoo} , { bread, cheese, milk} เป็นต้น
3. ไอเทมเซตตัวเลือก (Candidate Itemset) คือ Itemset ที่ได้จากการเชื่อมความสัมพันธ์ของ Itemset ก่อนหน้านี้ ซึ่งเป็นไอเทมเซตตัวเลือกสำหรับฟรีเควนท์ไอเทมเซต
4. ฟรีเควนท์ไอเทมเซต (Frequent Itemset) หรือ ลาร์จไอเทมเซต (Large Itemset) คือชุดของ Itemset ที่มีค่าสนับสนุนมากกว่าหรือเท่ากับค่าสนับสนุนน้อยที่สุด
5. ค่าสนับสนุน (Support value) เป็นค่าแสดงความสัมพันธ์ระหว่างจำนวนของไอเท็มที่ปรากฏรายการข้อมูลต่างๆ ทั้งหมด ในฐานข้อมูลสามารถแสดงเป็นสมการได้ดังสมการที่ 2.1

$$Support = \frac{n}{N} \tag{2.1}$$

โดยที่ n คือจำนวนไอเทมที่ปรากฏในรายการข้อมูลต่างๆ ของฐานข้อมูล

N คือจำนวนรายการข้อมูลทั้งหมดในฐานข้อมูล

6. ค่าสนับสนุนน้อยที่สุด (Minimum support : min_sup) คือค่าสนับสนุนที่น้อยที่สุดที่ทำให้ความสัมพันธ์ที่ได้นั้นยังมีความน่าสนใจ

$$\text{Support}(A \Rightarrow B) = P(A \cup B) \tag{2.2}$$

โดยที่ $P(A \cup B)$ คือ เป็นค่าความน่าจะเป็นที่จะปรากฏไอเท็ม A และ B พร้อมกันในรายการข้อมูลต่างๆ ของฐานข้อมูล

7. ค่าความเชื่อมั่นน้อยที่สุด (Minimum confidence) คือค่าความเชื่อมั่นที่น้อยที่สุดที่ทำให้กฎความสัมพันธ์ที่ได้นั้นยังมีความน่าสนใจ โดยจะบอกถึงความเข้มแข็งของกฎความสัมพันธ์ที่เกิดขึ้น โดยพิจารณาความน่าจะเป็นของความสัมพันธ์แบบมีเงื่อนไข ซึ่งสามารถคำนวณได้ แสดงดังสมการที่ 2.3

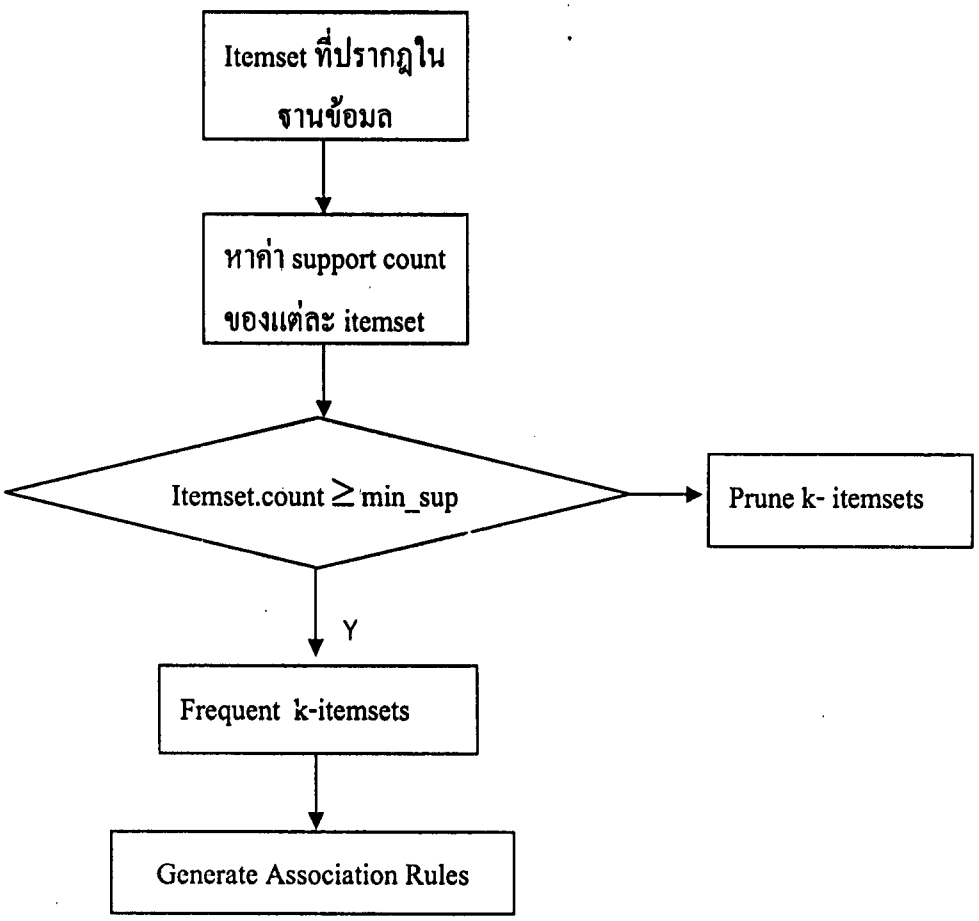
$$\text{Confidence}(A \Rightarrow B) = P(B \mid A) \qquad (2.3)$$

$$P(B|A) = \frac{P(A \cap B)}{P(A)}$$

โดยที่ P(B|A) หมายถึง ความน่าจะเป็นที่ B จะเกิดขึ้นเมื่อ A เกิดขึ้นแล้ว

P(A) คือความน่าจะเป็นของไอเท็ม A

การค้นหากฎความสัมพันธ์มีกระบวนการทำงาน 2 ขั้นตอนคือ หา Frequent Itemset ทั้งหมด และสร้างกฎความสัมพันธ์จาก Frequent Itemset แสดงดังรูปที่ 2.1



รูปที่ 2.1 ขั้นตอนการค้นหากฎความสัมพันธ์

อัลกอริทึมที่ได้รับความนิยมในการหากฎความสัมพันธ์ก็คือ Apriori [2] ซึ่งมีลักษณะดังนี้คือ

2.1.1 Apriori Algorithm

เป็นอัลกอริทึมที่เป็นที่รู้จักและมีการนำมาประยุกต์ใช้กับงานวิจัยต่างๆ อย่างแพร่หลายโดยอัลกอริทึมของ Apriori จะทำการคำนวณหาข้อมูลที่เป็น frequent itemsetsในฐานข้อมูล โดยทำการสแกนข้อมูลแต่ละ transaction ในฐานข้อมูลผ่านการวนซ้ำหลายครั้ง (iterations) และในการวนซ้ำจะคำนวณจาก frequent k-itemsets (itemsets ด้วยสมาชิก ตัว) โดยอาศัยความรู้ที่ได้จากขั้นตอนก่อนหน้าเช่น k-itemset จะได้จากการสร้าง candidate k-1 itemset (C_{k-1}) ในรูปแบบที่เรียกว่า level-wise search จากรูปที่ 2.2 เป็นการแสดงขั้นตอนการหา Frequent item ซึ่งจะประกอบด้วย 2 ขั้นตอนหลักๆ ดังนี้คือ

1. ขั้นตอนการ join (Join step)

เพื่อหา large itemset (L_k)การสร้างและนับ Candidate itemsets โดยอัลกอริทึมของ Apriori จะต้องมีการเรียงลำดับข้อมูลใน transaction (Lexicographic order) ที่ใช้ในการสร้างเซตของ candidate k-itemsets (C_k) ซึ่งมีค่า support > 0 และ candidate itemsets C_k จะสามารถหาได้จาก L_{k-1} ในกรณีที่ไม่ใช่ 1-itemset จะสามารถหา C_k ได้จากการนำ L_{k-1} มา join กัน ดังแสดงในรูปที่ 2.3 เนื่องจากในการหากฎความสัมพันธ์จะมีการนำข้อมูลมาเรียงลำดับ

การสร้าง Frequent itemset จาก Candidate itemset ที่ L_{k-1} โดยจะตรวจสอบว่าค่าsupport ของ Candidate itemset ใดๆ ว่ามีค่ามากกว่าหรือเท่ากับค่า minimum support ที่กำหนดไว้ item นั้นๆ จะกลายเป็น Large k-itemset (L_k)

2. ขั้นตอนการ Prune (Prune step)

เมื่อ C_k ถูกนำมาพิจารณาว่ามีค่าน้อยกว่า minimum support ก็จะเข้าสู่ขั้นตอนดังแสดงในรูปที่ 2.4 คือการ prune item นั้นๆ ออกไปจาก candidate itemsets เพื่อให้การคำนวณน้อยลงในส่วนของ Apriori ได้มีการกำหนดคุณสมบัติของ item ไว้ดังนี้ "ถ้าซับเซต (k-1) ใด ๆ ของ Candidate k-itemset ไม่ได้เป็นสมาชิก L_{k-1} ดังนั้น candidate ไม่สามารถเป็น frequent ได้ และสามารถลบออกจาก C_k ได้"

Algorithm Apriori. Find frequent itemsets using an iterative level-wise approaches based on candidate generation

Input: Database, D, of transactions; minimum support threshold, min_sup

Output: L,frequent itemsets in D.

Method:

1) L_1 = find_frequent_1-itemsets(D)
2) for (k=2; $L_{k-1} \neq \phi; k ==$){
3) $C_k = apriori\_gen(L_k, \text{min\_sup})$;
4) for each transaction t $\in$ D {// scan D for counts
5) C_t = subset(C_k, t); // get the supsets of t that are candidates
6) for each candidate c $\in$ C_t
7) c.count++;
8) }
9) $L_k = \{c \in C_k \mid c.count \geq \text{min\_sup}\}$
10) }
11) return L=$\bigcup_k L_k$;

รูปที่ 2.2 Apriori algorithm

function apriori_gen

Generate the candidate itemsets in C_k from the frequent itemsets in L_{k-1}

Join L_{k-1} p with L_{k-1}q, as follows:

insert into C_k

select p.item$_1$, q.item$_1$, . . . , p.item$_{k-1}$, q.item$_{k-1}$

from L_{k-1} p, L_{k-1}q

where p.item$_1$ = q.item$_1$, . . . p.item$_{k-2}$ = q.item$_{k-2}$, p.item$_{k-1}$ < q.item$_{k-1}$

รูปที่ 2.3 join step ของ procedure Apriori_gen

```
Prune Step :

for all itemsets c ∈ C_k do
  for all (k-1)-subsets s of c do
    if (s ∉ L_{k-1}) then
      delete c from C_k;
```

รูปที่ 2.4 Prune step ของ Apriori algorithm

กฎความสัมพันธ์ที่เป็นไปตามค่าที่กำหนดไว้ทั้ง minimum support threshold (min_sup) และ minimum confidence threshold (min_conf) จะถูกเรียกว่า กฎที่เข้มแข็ง (Strong) โดยจะเขียนค่า support และ confidenceที่มีในช่วงระหว่าง 0% ถึง 100%

กฎความสัมพันธ์จะหาค่าความสัมพันธ์ระหว่างแอททริบิวต์ที่แตกต่างกัน โดยจะมีการอ้างถึงเซตของ item ที่เรียกว่า itemset โดย itemset ที่ประกอบด้วย k-items (k = 1,2,3,...,n) จะเรียกว่า k-itemsets เช่น {milk, bread} เป็นตัวอย่างของ 2-itemsets และจากข้อมูล transaction จะสามารถทราบถึงความสัมพันธ์ที่เกิดขึ้นระหว่างรายการข้อมูลได้ด้วยการนับจำนวนของ transaction ที่มี itemset ที่เรียกว่า Frequency ของ itemset โดยค่าของ Frequency ของ itemset จะต้องมีค่ามากกว่าหรือเท่ากับค่า minimum support ที่กำหนดไว้ set ของ Frequent k-itemsets จะเขียนแทนด้วยสัญลักษณ์ L_k

การหากฎความสัมพันธ์สามารถทำได้ 2 ขั้นตอนคือ

1. หา frequent itemsets ทั้งหมด โดยแต่ละ itemsets จะต้องเกิดขึ้นอย่างน้อยเท่ากับค่าของ minimum support ที่กำหนดไว้

2. การสร้างกฎความสัมพันธ์จาก frequent itemsets ที่ได้ โดยกฎจะต้องเป็นไปตามค่า minimum support และ minimum confidence ที่กำหนดไว้

ปัญหาของ Apriori

1.เมื่อฐานข้อมูล หรือข้อมูลมีการเปลี่ยนแปลงต้องทำการหากฎความสัมพันธ์ใหม่ทั้งหมด โดยจะต้องสแกนข้อมูลใหม่ทั้งฐานข้อมูล

2. การหากฎความสัมพันธ์จะมีความซับซ้อนในการคำนวณสูงทำให้มีการคิดค้นหาอัลกอริทึมมาช่วยลดการคำนวณและสแกนข้อมูล

3. การหากฎความสัมพันธ์ในรูปแบบ Numeric หรือ Boolean ไม่สามารถทำได้

การหากฎความสัมพันธ์ของข้อมูลที่เป็น crisp set ทำให้ต้องมีการระบุว่าข้อมูลที่นำมาหากฎความสัมพันธ์นั้นอยู่ในกลุ่มใดนั้นอาจทำให้สูญเสียสารสนเทศที่ซ่อนอยู่ในความสัมพันธ์ได้

เนื่องจากข้อมูลที่นำมาหากฎความสัมพันธ์อาจมีความคลุมเครือในการที่จะเป็นสมาชิกในกลุ่มใดกลุ่มหนึ่ง

2.2 แนวคิดของการเพิ่มขยายการค้นหากฎความสัมพันธ์

ฐานข้อมูลเป็นแหล่งที่ใช้ในการจัดเก็บข้อมูลที่ประกอบด้วยข้อมูลต่างๆ มากมาย โดยทั่วไปจะมีการค้นหาความสัมพันธ์ระหว่างข้อมูลในฐานข้อมูล ซึ่งอาจพบความรู้ที่ซ่อนอยู่ภายในฐานข้อมูลจากนั้นสามารถนำความรู้ที่ได้มาสร้างเป็นกฎความสัมพันธ์

ข้อมูลที่ถูกจัดเก็บในฐานข้อมูลสามารถเปลี่ยนแปลงหรืออาจกล่าวได้ว่าข้อมูลในฐานข้อมูลนั้นไม่คงที่ เรียกฐานข้อมูลในลักษณะนี้ว่า dynamic database การเปลี่ยนแปลงฐานข้อมูลอาจเกิดจากการเพิ่มรายการข้อมูลเข้าไปในฐานข้อมูล (insert) , ลบรายการข้อมูลที่มีอยู่ในฐานข้อมูล (delete)หรืออาจมีทั้งการเพิ่มและลบรายการข้อมูลในฐานข้อมูล (modify)โดยจะเรียกส่วนของฐานข้อมูลก่อนทำการเปลี่ยนแปลงว่าฐานข้อมูลเดิม (original database : DB) และเรียกส่วนที่ทำการเปลี่ยนแปลงว่าฐานข้อมูลใหม่ (incremental database : db) และเรียกส่วนของฐานข้อมูลที่ผ่านการปรับปรุงหลังจากมีการเปลี่ยนแปลงข้างต้นแล้วว่า ฐานข้อมูลปรับปรุง (updated database : UP)

เมื่อฐานข้อมูลมีการเปลี่ยนแปลงจะมีผลต่อกฎความสัมพันธ์ที่ได้ไมน์นิ่งไว้ใน original database เนื่องจาก frequent itemset ที่ได้จาก original database อาจไม่สามารถเป็น frequent itemset ในฐานข้อมูลที่ปรับปรุงหรือ itemset ที่ไม่ผ่านค่า min_sup ของฐานข้อมูลเดิม หรือเรียกว่า small itemset อาจกลายมาเป็น frequent itemset ใน updated database ทำให้กฎความสัมพันธ์ที่ได้จาก original database บางกฎไม่สามารถเป็นกฎความสัมพันธ์ใน updated database อีกต่อไป ในขณะเดียวกันอาจพบกฎความสัมพันธ์ใหม่ที่เกิดขึ้นได้

2.3 งานวิจัยเกี่ยวกับ Incremental Association Rule

งานวิจัยด้านนี้เป็นการนำเสนออัลกอริทึมในหาความสัมพันธ์ระหว่างข้อมูลในฐานข้อมูลขนาดใหญ่ เมื่อมีการเพิ่มข้อมูลจาก transaction ใหม่ที่เข้ามา ซึ่งจะมีผลต่อการปรับเปลี่ยนกฎความสัมพันธ์ที่ได้เคยถูกสร้างไว้ โดยจะต้องทำการสแกน original database ที่มีอยู่และใน incremental database ที่มีการเพิ่ม transaction ใหม่เข้าไปเพื่อทำการสร้างกฎความสัมพันธ์ขึ้นมาใหม่ให้สอดคล้องกับข้อมูลใน transaction ที่ถูกเพิ่มเข้ามา [7] ซึ่งการเพิ่มข้อมูลที่เข้ามาจะทำให้เสียเวลาในการสแกนและหากฎความสัมพันธ์ใหม่ๆ งานวิจัย[8] จะวิธีการแบ่งข้อมูลเป็นส่วนๆ (partition) เพื่อลดจำนวนการสแกนฐานข้อมูลจากนั้นจะใช้ threshold ทำการกรองในแต่ละส่วนที่

เกี่ยวข้องกับการสร้าง candidate itemset การทำในรูปแบบนี้จะช่วยลดจำนวนของ candidate itemset และมีผลทำให้ลดซีพียูและหน่วยความจำ

รูปแบบการเพิ่มขยายกฎความสัมพันธ์ (Incremental association rule) จะนำเสนออัลกอริทึมในการหา large itemset เพื่อหากฎความสัมพันธ์ใหม่ที่เกิดขึ้นภายหลังจากมีการเพิ่มหรือปรับปรุงฐานข้อมูลเก่า โดยงานวิจัยที่ส่วนใหญ่จะเป็นการหาวิธีเพื่อลดการสแกน [3][4][6]-[15] เนื่องจากฐานข้อมูลที่เป็น original database มักจะมีขนาดใหญ่กว่า incremental database ดังนั้น ในงานวิจัยจึงมักหลีกเลี่ยงที่จะทำการสแกน original database

จากงานวิจัยที่มีการนำเสนออัลกอริทึมต่างๆ ที่นำมาใช้ในการเพิ่มขยายการค้นหากฎความสัมพันธ์ที่น่าสนใจสามารถสรุปได้ดังนี้คือ

2.3.1 FUP based algorithm

FUP เป็นงานวิจัยแรกที่นำเสนอเกี่ยวกับการ incremental updating technique ที่พัฒนาขึ้นมาเพื่อ maintenance association rules เมื่อมีการเพิ่ม transaction เข้าไปในฐานข้อมูลเดิม โดยลักษณะเด่นของ FUP Algorithm มีดังนี้คือ

1. นำความรู้ที่ได้จากค้นหากฎความสัมพันธ์โดยการไมน์นิ่งในฐานข้อมูลก่อนที่จะมีการเปลี่ยนแปลงมาใช้ เพื่อลดจำนวน transaction ที่จะต้องสแกนในฐานข้อมูลทั้งหมด จากนั้นจึงหาค่า support count ของข้อมูลต่างๆ ที่เกิดขึ้น ซึ่งแตกต่างจากอัลกอริทึม Apriori ที่จะต้องสแกนฐานข้อมูลทั้งหมด (ทั้งข้อมูลเก่าและข้อมูลใหม่ที่เปลี่ยนแปลง) เพื่อหากฎความสัมพันธ์ใหม่โดยไม่นำความรู้ที่ได้จากการหากฎความสัมพันธ์ในฐานข้อมูลเดิมมาใช้ให้เกิดประโยชน์ ดังนั้น FUP จึงเป็นแนวคิดแรกที่มีการนำความรู้ที่ได้จากการค้นหากฎความสัมพันธ์ก่อนหน้ามาใช้เพื่อลดการสแกนข้อมูลทั้งหมด

2. การหากฎความสัมพันธ์จะอยู่บนฐานของ Apriori algorithm คือมีการวนหาข้อมูลในลักษณะของ level-wise algorithm และมีการหา large itemset จาก candidate itemset

3. ใช้ minimum support เดียวกันในการหากฎความสัมพันธ์ FUP จะใช้ค่า minimum support เดียวกันในการค้นหา large k-itemset ของฐานข้อมูลเก่าและใหม่

อัลกอริทึมของ FUP (stands for Fast Update) จะอยู่บนฐานของ Apriori โดยเป็นอัลกอริทึมที่กล่าวถึงเทคนิคการทำ incremental updating ในกรณีการเพิ่มข้อมูลรายการเปลี่ยนแปลงเพียงกรณีเดียว ลักษณะการทำงานจะคล้ายกับ Apriori คือจะมีการวนรอบซ้ำ (iteration) เพื่อหาความสัมพันธ์ของข้อมูลเริ่มจาก 1-itemset ไปจนถึง k-itemset (k = 2, 3, …, n) ซึ่ง candidate itemsets ในแต่ละ iteration จะ based on large itemsets ที่พบใน iteration ก่อนหน้า FUP จะใช้ในการแก้ปัญหาด้านประสิทธิภาพในการปรับกฎความสัมพันธ์ (updated association rules) หลังจากที่มีการเพิ่ม transaction ใหม่เข้ามาในฐานข้อมูล จะนำกฎความสัมพันธ์ที่ได้ถูกค้นหาไว้ก่อนในฐานข้อมูลเดิมมาใช้ประโยชน์ เพื่อลดจำนวนข้อมูลที่จะต้องนำมาหากฎ

ความสัมพันธ์ โดยจะทำการสแกนฐานข้อมูลใหม่ซึ่งมีจำนวน transaction ที่เกิดขึ้นน้อยกว่าฐานข้อมูลเดิม แล้วจึงนำค่า support ที่ได้จากการสแกนฐานข้อมูลใหม่มารวมกับฐานข้อมูลเดิม โดยจะทำการพิจารณาหาค่าของlarge itemset ใหม่ (L') ที่ได้จากการรวมกันของฐานข้อมูลเดิมและฐานข้อมูลที่เพิ่มเข้ามา ($DB \cup db$) ด้วยค่า support ของ item ต้องไม่น้อยกว่า $s \times (D+d)$ (โดย s หมายถึง support, D หมายถึง จำนวนของ transaction ใน original database, d หมายถึง จำนวนของ transaction ใน incremental)

ข้อดีของ FUP คือ

1. นำความรู้ที่ได้จากการค้นหาในฐานข้อมูลเดิมมาใช้ร่วมกับรายการข้อมูลที่เพิ่มเข้าไปใหม่ เนื่องจากมีการใช้ minimum support ที่คงที่
2. ลดจำนวนของ candidate โดยการหาค่า candidate ที่ได้จากการสแกนข้อมูลรายการที่เพิ่มเข้าไปในฐานข้อมูลใหม่ (db) ไปหา ค่า support ในฐานข้อมูลเดิม โดยไม่จำเป็นต้องสแกนข้อมูลทั้งหมดในฐานข้อมูลเดิม
3. สามารถลดจำนวน itemset ที่ไม่พบว่าเป็น Frequent k-itemset ในฐานข้อมูลที่ปรับปรุง

ข้อด้อยของ FUP คือ

1. การสแกนฐานข้อมูลเดิมทุก k-itemset ที่พบ frequent itemset ในฐานข้อมูลใหม่

เนื่องจากในการไมน์นิ่งจากฐานข้อมูลเดิมจะเก็บเฉพาะ frequent k-itemset เท่านั้น ดังนั้นการหา new large itemset จะต้องทำการสแกนฐานข้อมูลเดิมทุกรอบของการหา frequent k-itemset ถึงแม้จะเป็นการสแกนเฉพาะค่าของ item ที่เป็นสมาชิกของ candidate ที่เป็น frequent k-itemset ที่ปรากฏในฐานข้อมูลใหม่

2.3.2 Negative border based algorithm

งานวิจัยนี้เป็นการศึกษาการ maintenance association rules เมื่อมีการเพิ่ม transaction ใหม่หรือลบ transaction เก่าออกจากฐานข้อมูล การหากฎความสัมพันธ์ในงานวิจัยนี้จะอยู่ในรูปแบบของ level-wise algorithm เช่นเดียวกับ Apriori แต่จะอาศัยหลักการของ negative border ที่ได้นำเสนอโดย Mannila and Toivonen [5]

สำหรับแนวคิดของ negative border จะเป็นการนำเสนอส่วนของซับเซตของ itemset ที่เป็นสมาชิกของ candidate k-itemset (C_k) โดยส่วนของสมาชิกของ Candidate k-itemset ที่มีค่ามากกว่าหรือเท่ากับค่า minimum support threshold จะจัดเก็บไว้ในส่วนของ Frequent k-itemset แต่ส่วนที่เหลือจะเรียกว่า Border set ทั้งนี้ Candidate k-itemset จะถูกสร้างขึ้นมาด้วยหลักการเดียวกับ APRIORI ที่มีการนำ Frequent k-1 itemset มาสร้าง Candidate k-itemset ดังนั้น Border set ที่ได้จะมีซับเซตที่เป็น frequent k-1 itemset โดยที่สมาชิกของ Border set ตัวนั้นไม่ได้เป็น frequent k-

itemset ความสัมพันธ์ระหว่าง candidate k-itemset(C_k) ,frequent k-itemset (L_k) และ negative border ($NBd(L_k)$) สามารถแสดงได้ดังนี้

$$NBd(L_k) = C_k - L_k \quad \text{หรือ} \quad C_k = L_k \cup NBd(L_k)$$

ตัวอย่างเช่น

C_1 = { A, B, C, D, E, F}

L_1 = {A,B, C, F}

$NBd(L_1)$ = {D, E}

C_2 = { {A, B}, {A, C},{A,F},{B,C},{B,F},{C,F}}

L_2 = { {A, B}, {A,C},{A,F},{C,F}}

$NBd(L_2)$ = { {B,C},{B,F}}

แนวคิดนี้ได้มีผู้นำเสนอในส่วนของ incremental association rule 2 งานวิจัยคือ Thomas et al [6] และFeldman et al [12] โดยทั้ง 2 งานวิจัยมีแนวคิดที่จะปรับปรุงการทำงานของอัลกอริทึม FUP ที่มีการนำความรู้ที่ได้จากการไมน์นิ่ง frequent k-itemset ที่ได้จาก original database มาใช้เพื่อลดการไมน์นิ่งฐานข้อมูลทั้งหมด ด้วยการลดจำนวน itemset ที่ต้องทำการไมน์นิ่ง แต่ยังคงต้องสแกนฐานข้อมูลทั้ง original database และ incremental database ในทุกรอบของการค้นหา ด frequent k-itemset ดังนั้นเพื่อเป็นการลดจำนวนการค้นหา frequent k-itemset จึงได้มีการนำส่วนของ border set เข้ามาช่วย ในงานวิจัยทั้ง2 จึงมีการเก็บทั้ง frequent k-itemset และส่วนที่เป็น border k-itemset

หลักการทำงานของ Border set ในส่วนของ incremental association rule นั้นจะประกอบด้วย 2 คุณลักษณะหลัก ๆ คือ

1. ถ้าไม่มี Frequent sets ใหม่ที่เกิดขึ้นจะไม่มีการ access original database ดังนั้นจึงสามารถรับประกันได้ว่าการสแกนฐานข้อมูลทั้งหมดจะเกิดกรณีเดียวเท่านั้นคือเมื่อมี frequent sets ใหม่ โดยอัลกอริทึมที่ได้มีการนำเสนอก่อนหน้านี้จะไม่สามารถรับประกันได้
2. ในกรณีที่ต้องสแกนฐานข้อมูลทั้งหมดใหม่นั้นจำนวนรอบของการสแกนฐานข้อมูลจะมีจำนวนไม่มากเนื่องจากจะทำการสร้าง candidate (k+1)-itemset ให้ครอบคลุมเฉพาะ frequent k-itemset ใหม่เท่านั้น ดังนั้นจึงมี Candidate (k+1)- itemset เท่านั้นที่จะถูกสแกนเพื่อหาค่า support

ข้อดี

1. ลดจำนวนครั้งในการสแกนฐานข้อมูลเดิม

2. สามารถปรับปรุงค่า support ของ frequent k-itemset และ border k-itemset ได้รวดเร็วในกรณีที่ itemset ต่างๆ มีการเปลี่ยนแปลงน้อย หรือไม่มีการเปลี่ยนแปลงจาก border k-itemset มาเป็น frequent k-itemset

ข้อด้อย

1. เมื่อมีการเปลี่ยนแปลงในส่วนของ border itemset มาเป็น Frequent itemset จะต้องทำการ rescan ฐานข้อมูลทั้งหมดใหม่ ทำให้ใช้เวลานานกว่าการทำไมน์นิ่งข้อมูลทั้งหมดอย่าง APRIORI ดังที่พบจากการทดลองในงานวิจัย [13] เนื่องจากต้องทำการปรับปรุงข้อมูลที่เป็น frequent และ border set และหา candidate k-itemset สำหรับ frequent itemset ใหม่ ทำให้ใช้เวลานาน

2. ใช้พื้นที่จำนวนมากในการจัดเก็บ itemset ทั้งหมด เนื่องจากเป็นแนวคิดที่มีการจัดเก็บข้อมูลที่ได้จากการไมน์นิ่งทั้งที่เป็นสมาชิกของ Frequent itemsets และ negative border

3. เมื่อมี item ใหม่เพิ่มขึ้นจากการไมน์นิ่ง ซึ่งอยู่นอกเหนือจาก border และ frequent itemset เดิมที่มีจะไม่สามารถทำการไมน์นิ่ง item ใหม่นี้ได้เนื่องจากอัลกอริทึมไม่มีการรองรับในส่วนนี้

2.3.3 Expected frequent itemset algorithm

Expected frequent itemset เป็นงานวิจัยที่มีการนำเสนอแนวคิดในการลดจำนวนครั้งในการสแกน original database โดยจะมีการใช้ข้อมูลที่ได้จากการไมน์นิ่งในฐานข้อมูลเดิมทั้งส่วนที่เป็น Frequent itemset และในส่วนที่คาดว่าจะกลายมาเป็น Frequent itemset ได้เมื่อมีการเพิ่มฐานข้อมูลใหม่เข้ามาด้วยขนาดของข้อมูลในช่วงที่กำหนดเข้ามา (incremental database) ในฐานข้อมูลเดิม เรียก itemset ที่คาดว่าจะกลายมาเป็น Frequent itemset นี้ว่า Expected frequent itemset

แนวคิดของ Expected frequent itemset ที่ Hong et al [14] และ Amornchewin และ Kreesuradej [15] ได้นำเสนอแนวคิดที่แตกต่างจากการใช้ border set ซึ่งมีการเก็บข้อมูลจำนวนมาก มาใช้เกณฑ์ในการพิจารณาเลือกเก็บเฉพาะ itemset ที่เป็น small itemset หรือ itemset ที่มีค่า support น้อยกว่า minimum support ที่กำหนดไว้ แต่มีโอกาสกลายมาเป็น frequent itemset ได้เมื่อมีการเพิ่มฐานข้อมูลใหม่เข้ามา โดยเรียกส่วนของ itemset นี้ว่า expected frequent itemset หรือ promising frequent itemset

การค้นหา promising frequent itemset นี้จะมีการเกณฑ์ที่ใช้ในการพิจารณา itemset ที่คาดว่าจะกลายเป็นfrequent itemset โดยอาศัยหลักทางสถิติของ itemset ที่เกิดขึ้นในฐานข้อมูลเดิมจะมีความแตกต่างน้อยหรือไม่เปลี่ยนแปลงกับฐานข้อมูลใหม่ที่เกิดขึ้น ด้วยการนำค่า support ที่เกิดขึ้น

สูงสุดจากการทำไมน์นิ่งฐานข้อมูลเดิมมาใช้ในการคำนวณหาเกณฑ์ที่เรียกว่า min_PL ด้วยสมการดังต่อไปนี้

$$min\_sup_{DB} - \left(\frac{maxsupp}{total\ size}) \times inc\_size\right) \le min\_PL < min\_sup_{DB} \quad (1)$$

โดย maxsupp หมายถึง ค่า support สูงสุดที่ได้จากการไมน์นิ่งในฐานข้อมูลเดิม

total size หมายถึง ขนาดของฐานข้อมูลเดิม

inc_size หมายถึง ขนาดของฐานข้อมูลใหม่ที่จะเพิ่ม

จากสมการที่ (1) เมื่อทำการไมน์นิ่ง 1-itemset จะทำให้เราทราบค่า support สูงสุดของ 1-itemset ที่เกิดขึ้น และสามารถนำมาใช้ในการหาขนาดของฐานข้อมูลใหม่ที่เพิ่มเข้ามา ได้ดังนี้

$$0 < min\_PL < min\_sup$$

$$0 < min\_PL = \frac{max\_sup\ p}{|DB| + inc\_size} * inc\_size < min\_sup$$

$$0 < inc\_size < \frac{min\_sup * |DB|}{max\_sup\ p - min\_sup} \quad (2)$$

ขนาดของข้อมูลที่คำนวณได้จะช่วยในการประมาณค่าของฐานข้อมูลใหม่ที่เพิ่มเข้ามา ทำให้สามารถ guarantee ได้ว่าด้วยขนาดของ incremental database ที่เพิ่มเข้ามาและจากสมมติฐานของสถิติการเกิดของ items ไม่แตกต่างหรือแตกต่างกันน้อย ทำให้สามารถนำค่า min_PL ที่ได้มาใช้ในการพิจารณาหา itemset ที่คาดว่าจะกลายมาเป็น Frequent k-itemset ได้ และจะช่วยลดหรือหลีกเลี่ยงการสแกน original database ลงไปได้ เนื่องจากนำทั้ง Frequent k-itemset และ Promising Frequent k-itemset ที่ได้มาสร้าง Candidate k+1 itemset เพื่อให้การค้นหา Expected Frequent itemset ครอบคลุมทุก itemset ที่คาดว่าจะเกิดเมื่อมีการเพิ่มฐานข้อมูลใหม่เข้ามา

แนวคิดนี้จะใช้ได้ดีเมื่อ itemset ที่พบใน incremental database เป็นสมาชิกของ frequent k-itemset หรือ expected k-itemset ทั้งนี้ในกรณีที่บาง expected frequent k-itemset กลายมาเป็น frequent k-itemset ใน Updated database ก็ไม่มีความจำเป็นต้องสแกน original database ใหม่

เมื่อมี itemsetใหม่เกิดขึ้น อัลกอริทึมจะทำการตรวจสอบ เพื่อลดการสแกนฐานข้อมูลได้เป็น 2 กรณี

1. กรณีที่ itemset ใหม่ที่เกิดขึ้นนั้น เกิดเฉพาะใน incremental database เนื่องจากในการ update 1-itemset จะสามารถตรวจสอบได้ว่า itemset ต่างๆ ที่เกิดขึ้นในฐานข้อมูลใด ซึ่งถ้าพบว่า itemset ใหม่นั้นปรากฏเพียงใน incremental database อัลกอริทึมนี้จะทำการสแกน candidate k+1 itemset สำหรับ itemset ใหม่ที่เกิดขึ้น ใน incremental database เท่านั้น ทำให้ไม่ต้องเสียเวลาในการสแกน Original database

2. กรณีที่ itemset ใหม่ที่เกิดขึ้นนั้นเป็นสมาชิกของ Candidate 1-itemset แต่ไม่เป็นสมาชิกของ frequent 1-itemset และ expected frequent 1-itemset ซึ่งจาก 1-itemset นี้จะทำให้ทราบถึง itemset ใหม่ที่เกิดขึ้นแตกต่างจาก original database ดังนั้นในกรณีนี้จะมีการสร้าง candidate k+1 itemset โดยทำการ join ระหว่าง itemset ใหม่ และfrequent k-itemset และ expected frequent k-itemset ใน original database เพื่อให้สามารถค้นหา frequent k+1 itemset ที่ครอบคลุม และถ้า candidate k-itemset ใดมีค่ามากกว่าหรือเท่ากับ minimum support *|db| หรือมากกว่าหรือเท่ากับ min_PL ของ updated database หรือ มากกว่าหรือเท่ากับ min_PL ของ original database itemset เหล่านี้จะถูกนำไปสแกนเพื่อหาค่า support count ใน original database

ดังนั้น การสแกน original database จะทำเพียงกรณีเดียวเท่านั้นคือ กรณีที่ 2 ซึ่งจำนวนของ itemset จะมีจำนวนน้อยเนื่องจาก itemset ใหม่ที่จะถูกนำมาสแกนใน original database จะต้องเป็น frequent itemset ที่ปรากฏใน incremental database หรือ expected frequent itemset โดย itemset ที่มีค่า support น้อยกว่า min_PL ของ updated database จะถูก prune ออกไป

ตัวอย่าง original database ซึ่งประกอบด้วย 10 transaction โดยฐานข้อมูลจะประกอบด้วย item 5 ตัว คือ {A, B, C, D, E} ด้วย minimum support = 40% ดังแสดงในรูปที่ 2.5

TID	List of item
1	A, B, E
2	B, D
3	B,C
4	A, B,D
5	A, C
6	B, C
7	A, C
8	A, B, C, E
9	A, B, E
10	A,C

Itemset	support
A	7
B	7
C	6
D	2
E	3

รูปที่ 2.5 Transaction ใน original database และ Candidate 1-itemset

จากรูปที่ 2.5 พบว่า max_supp มีค่าเท่ากับ 7 จากสมการ (1) และ (2) สามารถคำนวณ incremental size ได้คือ

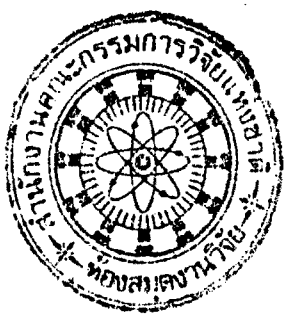
$$min\_sup = 0.4 * 10 = 4$$

$$inc\_size < \frac{4*10}{7-4} \approx 13$$

ถ้าขนาดของ incremental database คือ 3 transactions ดังนั้นค่า min_PL สามารถคำนวณได้ดังนี้

$$min\_PL = 4 - \frac{7}{10} * 3 \approx 2$$

จากรูปที่ 2.5 จะได้ frequent 1-itemset (L_1) คือ {A, B, C} และ Expected Frequent 1-itemset ในที่นี้แทนด้วย PL_1 คือ {D,E}



C2	support
AB	4
AC	4
AD	1
AE	3
BC	3
BD	2
BE	3
CD	0
CE	1
DE	0

L2	Support
AB	4
AC	4
PL2	support
AE	3
BC	3
BD	2
BE	3

รูปที่ 2.6 แสดง Candidate 2-itemset (C2) ,Frequent 2-itemset และ Expected Frequent 2-itemset

C3	Support
ABC	1
ABE	3
ACE	1
BCD	0
BCE	0
BDE	0

PL3	support
ABE	3

รูปที่ 2.7 แสดง Candidate 3-itemset (C_3) ,Frequent 3-itemset (L_3)และ Expected Frequent 3-itemset (PL_3)

เมื่อมีการเพิ่ม incremental database จะทำการ update ค่า support count ของ frequent k-itemset และ Expected Frequent k-itemset โดยจะทำการคำนวณค่า min_PL ของ updated database ใหม่ ดังสมการที่ 3

$$min\_PL_{DB\cup db} = min\_sup_{DB\cup db} - \left(\frac{max\,supp}{total\ size} \times inc\_size\right) \qquad (3)$$

ข้อดี

1. ลดจำนวนการสแกนฐานข้อมูลเดิม เนื่องจากข้อมูลบางตัวที่เป็นสมาชิกของ Expected Frequent itemset อาจกลายเป็น Frequent itemset ได้ ด้วยข้อมูลที่ได้จากฐานข้อมูลใหม่เพียงเล็กน้อย

2. ลดเวลาในการหา itemset ใหม่โดยใช้ expected value ในที่นี้คือ min_PL มาช่วยในการพิจารณา itemset ที่คาดว่าจะกลายเป็น Frequent itemset

3. เมื่อมีการเปลี่ยนแปลงจาก expected frequent itemset มาเป็น frequent itemset ไม่จำเป็นต้องเข้าไปทำการไมน์นิ่งใหม่เนื่องจากหา combination ของ frequent itemset และ expected frequent itemset รองรับไว้แล้ว

ข้อเสีย

1. การเพิ่มขนาดของรายการข้อมูลเข้าไปในฐานข้อมูลเดิมทำได้จำกัด เนื่องจากขนาดของข้อมูลต้องอยู่ในช่วงที่ได้จากการคำนวณซึ่งมีช่วงขนาดของข้อมูลที่ไม่มาก

2. การปรับปรุงข้อมูลที่ได้จากการไมน์นิ่งมีจำนวนมาก เมื่อมี incremental database เข้ามาจะต้องทำการปรับปรุง itemset ที่เป็น Frequent k-itemset และ Expected Frequent k-itemset และในกรณีที่ค่าของ expected value ที่คำนวณได้มีค่า = 1 จะเป็นกรณีที่ต้องเก็บข้อมูลทุก itemset ที่เกิดขึ้นในการไมน์นิ่ง ซึ่งอาจใช้เวลานานในการปรับปรุงทุก itemset

2.4 ทฤษฎีเบอร์นูลลี (Bernoulli Theorem)

ทฤษฎีทางสถิติที่ใช้ในการพิจารณาการทดลอง ซึ่งผลการทดลอง 1 ครั้งมีผลอย่างใดอย่างหนึ่งใน 2 แบบเท่านั้น คือ ความสำเร็จ (success) หรือ ความไม่สำเร็จ (failure) เช่น โยนเหรียญ 1 อัน จะปรากฏผลเป็นหัวหรือก้อย ซึ่งอาจถือว่าการขึ้นหัวเป็นผลสำเร็จ ส่วนการขึ้นก้อยเป็นความไม่สำเร็จ สำหรับตัวอย่างของการโยนเหรียญนี้ ความน่าจะเป็นที่จะขึ้นหัวเท่ากับ $\frac{1}{2}$ และความน่าจะเป็นที่จะขึ้นก้อยเท่ากับ $\frac{1}{2}$ โดยความน่าจะเป็นที่จะเกิดความสำเร็จเท่ากับ p และ ความน่าจะเป็นที่จะเกิดผลความไม่สำเร็จเท่ากับ q ซึ่ง p+q = 1

การค้นหากฎความสัมพันธ์ได้มีการนำเบอร์นูลลีมาประยุกต์ใช้ในการพิจารณา itemset ที่มีโอกาสจะกลายมาเป็น Frequent itemset เมื่อทำการเพิ่มรายการข้อมูลใหม่เข้ามาในฐานข้อมูลเดิม ด้วยสูตรการคำนวณดังต่อไปนี้

$$P(x)=\binom{n+m}{x}\cdot p^{x}\cdot(1-p)^{n+m-x}$$

โดย P หมายถึงค่าความน่าจะเป็น

(n+m) หมายถึง ขนาดของข้อมูลที่จะทำการทดลอง

p หมายถึง ค่าความน่าจะเป็นที่เกิดความสำเร็จจากการทดลอง

(1-p) หมายถึง ค่าความน่าจะเป็นที่การทดลองไม่มีความสำเร็จ

x หมายถึง จำนวนครั้งที่การทดลองจะเกิดความสำเร็จ

ในส่วนของ association rule mining ได้นำ Bernoulli distribution มาใช้ในการประมาณค่า itemset ที่จะเป็น Frequent itemset หรือ small itemset โดย itemset ที่เป็น Frequent จะเป็นเหตุการณ์ที่สำเร็จ (success) ส่วนitemset ใดที่ไม่เป็น Frequent itemset จะเรียกว่า ความไม่สำเร็จ (Failure) โดย Zhang et al [16] ได้นำ Bernoulli distribution มาใช้ในการประมาณค่าการเกิด frequent itemset จากข้อมูลสุ่มของ original database ด้วย method ของ Central limit theorem มาช่วยเพิ่มประสิทธิภาพ แนวคิดนี้พัฒนาขึ้นมาภายใต้ข้อจำกัดของเวลา ซึ่งเวลาเป็นสิ่งที่สำคัญ

มากกว่าความถูกต้องที่สูง เช่น การลงทุนในตลาดหุ้น ถ้านักลงทุนสามารถ ที่จะประมาณค่าของ frequent itemset จากข้อมูลใน stock database ได้อย่างรวดเร็ว จะช่วยให้การตัดสินใจที่จะลงทุนได้รวดเร็วและถูกต้อง แทนที่จะต้องทำการค้นหา frequent itemset จากฐานข้อมูลทั้งหมด

การเพิ่มขยายการค้นหากฎความสัมพันธ์ของงานวิจัยนี้ สามารถค้นหากฎความสัมพันธ์ของฐานข้อมูลได้อย่างมีประสิทธิภาพในกรณีของการเพิ่ม incremental database เข้าไป โดยจะสามารถลดจำนวน itemset ที่จะต้องนำไปสแกนใน original database และสามารถหา frequent itemsetใหม่ได้อย่างมีประสิทธิภาพโดยนำ Bernoulli Theorem มาประยุกต์ใช้ดังจะกล่าวรายละเอียดต่างๆ ในบทต่อไป