MONASH University
Information Technology

CSE3212 Data Mining

Mining Associations from Large Databases

Outline

- **Terminologies -> revisited**
- **Naïve Algorithm**
- **Apriori algorithm**
  - Basic principles
  - Examples

Terminologies

- **Transactions**
  - An event of an item or items are being purchased.
- **Item sets**
  - A set of items that may be used to generate association rules.
  - Examples:
    - > {Orange, Apple}
    - > {Orange, Apple, Kiwi}
- **Association rules**
  - A rule depicts association among items in an items set. Its 'interestingness' is measure by the prevalence of the rule (support) and the predictability of the rule (confidence).
  - From item set {Orange,Apple}, it is possible to generate the following rules:
    - > Orange -> Apple [support p%, confidence q%]
    - > Apple -> Orange [support m%, confidence n%]

The Story So Far …

- **Given a set of transactions, generate co-occurrence matrix.**
- **Determine an association rule, calculate its support, confidence and lift.**
- **It will be computationally expensive to calculate support, confidence and lift for the exhaustive list of associations.**
  - The total possible rules are huge for a typical 'supermarket' or 'shop' data set.
- **What can we do?**
  - Are we interested to know all of the association? NO
  - We are only interested in association rules that occurs frequently in the transaction.
- **How can we find the associations that occurs frequently from transactions set?**
  - Mining Associations Algorithm

From Transaction Set to Association Rules

- **Main principles**
  - Find only associations that occurs frequently.
- **General steps**
  - Step 1. Discover all frequent items that have support above the minimum support required
  - Step 2. Use the set of frequent items to generate the association rules that have high enough confidence level
- **Step 1 is the most time consuming process. Designing an efficient algorithm to perform step 1 is the main objective of many mining algorithm.**

Transactions - Example

Transaction ID	Items
100	Bread, eggs, cheese
200	Bread, cheese, eggs
300	Bread, yoghurt, eggs
400	Bread, eggs, cheese
500	Cheese

- **4 Items = bread, cheese, eggs, yoghurt**
- **Find all association rules that has support > 50% and confidence > 70%**

Naïve Algorithm (1)

Itemsets	Frequency
{bread}	4
{cheese}	4
{eggs}	4
{yoghurt}	1
{bread,cheese}	3
{bread,eggs}	4
{bread,yoghurt}	1
{cheese,eggs}	3
{cheese, yoghurt}	0
{eggs,yoghurt}	1
{bread,cheese,eggs}	3
{bread,cheese,yoghurt}	0
{cheese,eggs,yoghurt}	0
{bread,cheese,eggs,yoghurt}	0

Naïve-Algorithm (2)

Item sets with support > 50%	
{bread}	4
{cheese}	4
{eggs}	4
{bread,cheese}	3
{bread,eggs}	4
{cheese,eggs}	3
{bread,cheese,eggs}	3

- Frequent Item Set = {{bread,cheese}, {bread,eggs}, {cheese,eggs}, {bread,cheese,eggs}}
- {bread,cheese} item set can be represented using two rules:
  - bread -> cheese [support 60%, confidence 75%]
  - cheese -> bread [support 60%, confidence 100%]
- The confidence of both rules are above the 70% threshold, so both rules will be reported.

Naïve-Algorithm (3)

- **Steps:**
  1. Create all possible items sets.
  2. Scan all transactions to populate the frequency of each items set.
  3. Remove all items sets below the support threshold.
  4. Generate the rules from the reminder items sets.
  5. Calculate the confidence of each rule.
  6. Report association rules that are greater than the confidence threshold.
- **Problems:**
  - The total number of possible items sets is very large
  - 2^n – 1.

Improvement on the Naïve-Algorithm (1)

- **Only generate items sets that have frequency greater than 0.**

Transaction ID	Items	Items set
100	Bread, eggs, cheese	{{bread,eggs},{bread,cheese},{eggs,cheese},{bread,cheese,eggs}}
200	Bread, cheese, eggs	as in transaction 100
300	Bread, yoghurt, eggs	{{bread,eggs},{bread,yoghurt},{eggs,yoghurt},{bread,yoghurt,eggs}}
400	Bread, eggs, cheese	as in transaction 100
500	Cheese	{cheese}

Improvement on the Naïve-Algorithm (2)

Itemsets	Frequency
{bread}	4
{cheese}	4
{eggs}	4
{yoghurt}	1
{bread,cheese}	3
{bread,eggs}	4
{bread,yoghurt}	1
{cheese,eggs}	3
{eggs,yoghurt}	1
{bread,cheese,eggs}	3

Proceed normally as in the naïve algorithm.

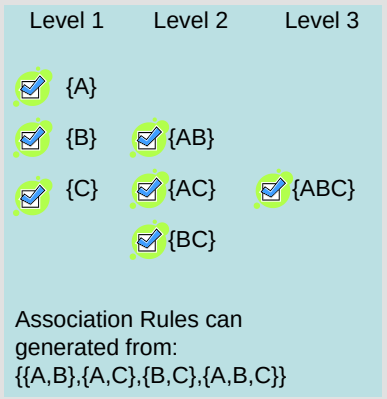
Still need to generate large amount of items sets.

Is it possible to further improve?

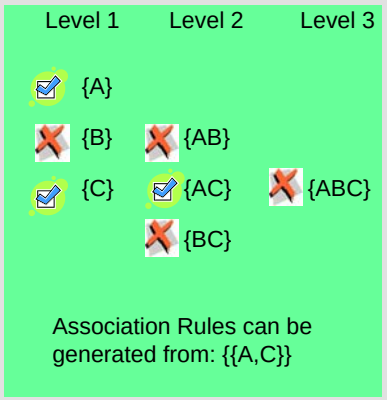
Relation between the values of support for items sets

- **For any given items set, the support will always be greater or equal to the support of its superset.**
- **Example:**
  - Support for {bread} is always greater or equal to the support for {bread,cheese} or any items sets that contain bread.
- **Can we use this observation to design an algorithm?**
  - YES, Apriori algorithm.
  - How?
    - > Use this observation to select candidates for the frequent items sets.

Examples



Example-1



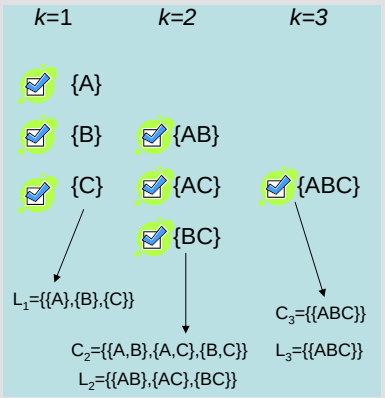
Example-2

Apriori Algorithm

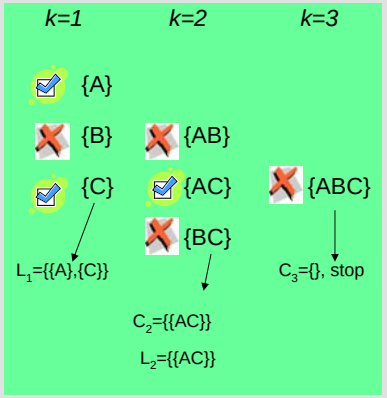
The algorithm works as follows

- Scan all transactions and find all frequent items that have transaction support above *x%*. Let these be L_1.
- Build item pairs from L_1. This is the candidate set C_2. Scan all transactions and find all frequent pairs in C_2. Let this be L_2.
- General Rules:
  - Join step: Build sets of k items from L_{k-1}. This is set C_k.
  - Prune step: Scan all transactions and find all frequent sets in C_k. Let this be L_k.
- The algorithm stop when either $L_{k-1} = \{\}$ *or* $C_k = \{\}$.

Examples- revisited



Example-1



Example-2

Transactions – Example – Revisited (1)

Transaction ID	Items
100	Bread, eggs, cheese
200	Bread, cheese, eggs
300	Bread, yoghurt, eggs
400	Bread, eggs, cheese
500	Cheese

- 4 Items = bread, cheese, eggs, yoghurt
- Find all association rules that has support > 50% and confidence > 70%

Transactions – Example – Revisited (2)

k=1	
{bread}	4
{cheese}	4
{eggs}	4
{yoghurt}	1
L1={{bread},{cheese},{eggs}}	

k=2	
{bread,cheese}	3
{bread,eggs}	4
{cheese,eggs}	3
C2={{bread,cheese},{bread,eggs}, {eggs,cheese}}	
L2={{bread,cheese},{bread,eggs}, {eggs,cheese}}	

Transactions – Example – Revisited (3)

k=3	
{bread,cheese,eggs}	3
C3={bread,cheese,eggs}	
L3={bread,cheese,eggs}	

Generating Rules

- **Rules are generated from all the frequent items set found by the algorithm.**
- **Report the rules that have confidence level higher than the threshold specified by user.**
- **Examples:**
  - Items set {A,B}
    - > Rules: A->B , B->A
  - Items set {A,B,C}
    - > Rules:
      - A->B; B ->A; A->C; C->A; B->C; C->B
      - B,C -> A; A,C ->B; A,B ->C

Generating Rules - Example

k=2	
{bread,cheese}	3
{bread,eggs}	4
{cheese,eggs}	3
C2={{bread,cheese},{bread,eggs}, {eggs,cheese}}	
L2={{bread,cheese},{bread,eggs}, {eggs,cheese}}	

k=3	
{bread,cheese,eggs}	3
C3={bread,cheese,eggs}	
L3={bread,cheese,eggs}	

Frequent items set ={
{bread,cheese},{bread,eggs},{eggs,cheese},{bread,cheese,eggs}}

Rules:

• bread-> cheese; cheese -> bread; bread->eggs; eggs->bread; eggs->cheese; cheese->eggs

• cheese, eggs -> bread ; bread,eggs ->cheese; bread,cheese -> eggs

Example of Rules

Rules	Support	Confidence
bread->cheese	0.6	0.75
cheese -> bread	0.6	0.75
bread -> eggs	0.8	1
eggs -> bread	0.8	1
eggs -> cheese	0.6	0.75
cheese -> eggs	0.6	0.75
cheese, eggs -> bread	0.6	1
bread, eggs -> cheese	0.6	0.75
bread, cheese -> eggs	0.6	1

Finally …

- **Associations with 50% support and 70% confidence.**

Rules	Support	Confidence
bread->cheese	0.6	0.75
cheese -> bread	0.6	0.75
bread -> eggs	0.8	1
eggs -> bread	0.8	1
eggs -> cheese	0.6	0.75
cheese -> eggs	0.6	0.75
cheese, eggs -> bread	0.6	1
bread, eggs -> cheese	0.6	0.75
bread, cheese -> eggs	0.6	1

To be continued …

- **Larger example.**
- **Choosing rules.**
- **Improvement to the Apriori algorithm.**