11B – ASSOCIATION RULE MINING

CS 1656

Introduction to Data Science

Alexandros Labrinidis – http://labrinidis.cs.pitt.edu
University of Pittsburgh

Association Rule Mining

- One specific type of data mining

- Usually:
  - Try to predict novel and interesting patterns from supermarket data

Famous examples:

- Purchase of Diapers ➔ Purchase of Beer
  - http://www.dssresources.com/newsletters/66.php

- How Target Figured Out A Teen Girl Was Pregnant Before Her Father Did [Forbes, Feb 2012]
  - http://bit.ly/targetpregnant

FREQUENT ITEMSETS

Transactions Example

- Market-Basket Model
  - Multiple items (e.g., milk, bread, etc)
  - Multiple baskets (transactions)
- Assumption:
  - Number of items in basket much smaller than total number of items

TID	Produce
1	MILK, BREAD, EGGS
2	BREAD, SUGAR
3	BREAD, CEREAL
4	MILK, BREAD, SUGAR
5	MILK, CEREAL
6	BREAD, CEREAL
7	MILK, CEREAL
8	MILK, BREAD, CEREAL, EGGS
9	MILK, BREAD, CEREAL

Transactions Example (compressed form)

TID	Produce
1	MILK, BREAD, EGGS
2	BREAD, SUGAR
3	BREAD, CEREAL
4	MILK, BREAD, SUGAR
5	MILK, CEREAL
6	BREAD, CEREAL
7	MILK, CEREAL
8	MILK, BREAD, CEREAL, EGGS
9	MILK, BREAD, CEREAL

TID	Products
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, C

ITEMS:

A = milk
B = bread
C = cereal
D = sugar
E = eggs

Transactions Example (binary form)

TID	Products
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, C

ITEMS:

A = milk
B = bread
C = cereal
D = sugar
E = eggs

Attributes converted to binary flags

TID	A	B	C	D	E
1	1	1	0	0	1
2	0	1	0	1	0
3	0	1	1	0	0
4	1	1	0	1	0
5	1	0	1	0	0
6	0	1	1	0	0
7	1	0	1	0	0
8	1	1	1	0	1
9	1	1	1	0	0

Definitions

- **Item**: *attribute=value* pair or simply *value*
  - usually attributes are converted to binary *flags* for each value, e.g. **product="A"** is written as **"A"**
- **Itemset** *L* : a subset of possible items
  - Example: *L* = {A,B,E} (order unimportant)
- **Transaction**: (TID, itemset)
  - TID is transaction ID

Support and Frequent Itemsets

- **Support count** of an itemset
  - sup(*L*) = number of transactions that support (i.e. contain) *L*
  - Example:
    - sup ({A,B,E}) = 2 and sup ({B,C}) = 4

- **Support percentage** of an itemset
  - supp(*L*) = percentage of transactions that support (i.e. contain) *L*
    - supp(*L*) = sup(*L*) / *total_count*
    - *total_count* is total number of transactions
  - Example:
    - supp ({A,B,E}) = 2/9 and supp ({B,C}) = 4/9

- An itemset *L* is frequent if it has support count at least minsup
  - sup(*I*) >= *minsup*

Q1. Understanding Question

- **Question**:
  - Which of the following doubletons has support count of exactly 5? (based on the transaction data from the handout)

- **Possible Answers**:
  - AB
  - AC
  - BC
  - DE
  - AF

Support counts for doubletons

	F	E	D	C	B
A	AF: 3	AE: 2	AD: 5	AC: 4	AB: 6
B	BF: 4	BE: 2	BD: 7	BC: **5**	
C	CF: 2	CE: 3	CD: 5		
D	DF: 4	DE: 2			
E	EF: 2				

Q2. Understanding Question

- **Question**:
  - What is the combined sum of the support counts of ABC, ABD, and ABE? (based on the transaction data from the handout)

- **Possible Answers**:
  - 9
  - 11
  - 12
  - 13
  - 15

Support count for size-3 itemsets

- ABC: 4
- ABD: 5
- ABE: 2

- Q: Any interesting observations?

- A: Support count of (ABC) is the minimum of support counts of AB (6), BC (5), AC (4)!

SUBSET PROPERTY

SUBSET PROPERTY

- **Every subset of a frequent set is frequent!**

- Why is it so?
- **Example**: Suppose {A,B} is a frequent itemset. Since each occurrence of A, B includes both A and B, then both {A} and {B} must also be frequent.

- Similar argument for larger itemsets

- Almost all association rule algorithms are based on this subset property

Q3. Understanding Question

- **Question**:
  - If minsup =4 can ABF be frequent itemset?
    (based on the transaction data from the handout)

- **Possible Answers**:
  - Yes
  - No

ASSOCIATION RULES

Association Rules

- Association rule *R* : *Itemset1* => *Itemset2*
  - *Itemset1*, *Itemset2* are disjoint and
  - *Itemset2* is non-empty
  - Simplified definition: *Itemset2* has only one item
- Meaning:
  - if transaction includes *Itemset1* then it also has *Itemset2*
- Examples
  - A,B => E
  - A => B,C

From Frequent Itemsets to Association Rules

- *Q: Given frequent set {A,B,E}, what are possible association rules?*
  - A, B => E
  - A, E => B
  - B, E => A

  - A => B, E
  - B => A, E
  - E => A, B

  - __ => A,B,E (empty rule), or true => A,B,E
    - We will ignore empty rules from this point on

Definition of **Support** for Association Rules

- Association Rule R: I => J
  - Example: {A, B} => {C}

- Support count for R:

  sup(R) = sup (I => J) = sup (I U J)
  - **Example**:
    sup({A,B}=>{C}) = sup ({A,B} U {C}) = sup ({A,B,C}) = 2

- Support percentage for R:

  supp(R) = supp (I => J) = supp (I U J)
  - **Meaning**:
    fraction of transactions that involve both left-hand side (LHS) and right-hand side (RHS) itemsets

Definition of **Confidence** for Association Rules

- Association Rule R: I => J
  - Example: {A, B} => {C}

- Confidence for R:
  conf (R) = conf (I=>J) = **sup (I U J) / sup(I)**
  - **Example**:
    conf ({A,B}=>{C}) = sup ({A,B,C}) / sup ({A,B})
    = 2 / 4 = 50%
  - **Meaning**:
    probability that RHS will appear given that LHS appears

Associate Rules Example

- ***Q: Given frequent set {A,B,E}, what association rules have at least minsup = 2 and minconf = 50% ?***

  A, B => E : conf=2/4 = 50%

  A, E => B : conf=2/2 = 100%

  B, E => A : conf=2/2 = 100%

  E => A, B : conf=2/2 = 100%

Do not qualify:

A =>B, E : conf=2/6 =33%< 50%

B => A, E : conf=2/7 = 28% < 50%

__ => A,B,E : conf: 2/9 = 22% < 50%

TID	List of items
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, C

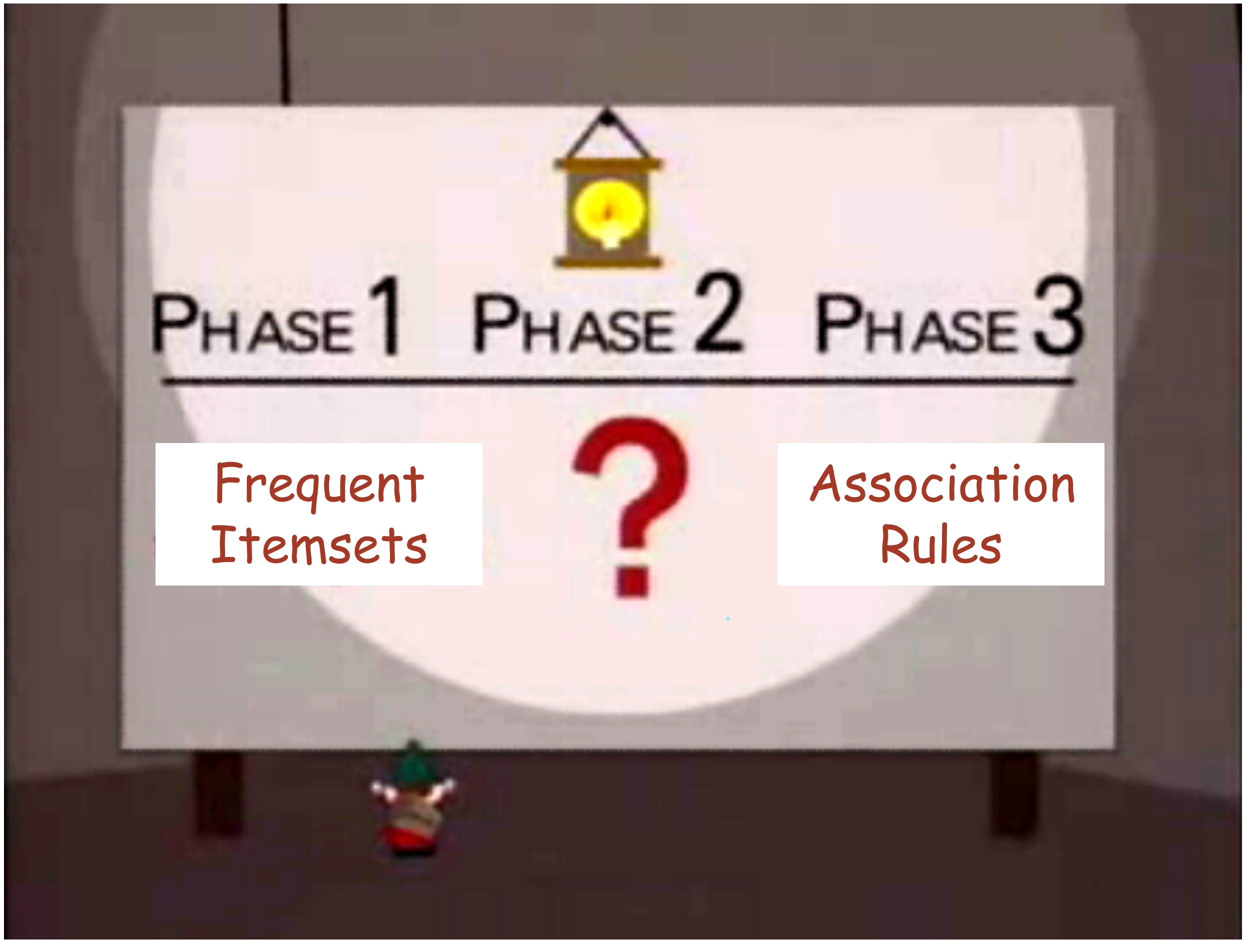
Q4. Understanding Question

- **Question**:
  - What is the confidence of association rule **A B => C**? (based on the transaction data from the handout)

- **Possible Answers**:
  - 4
  - 4 / 6
  - 5
  - 6 / 4
  - 6

A-PRIORI ALGORITHM

How to generate association rules?



Find Strong Association Rules

- An association rule has parameters *minsup* and *minconf*:
  - sup(R) >= *minsup* and conf (R) >= *minconf*
- **Problem Statement**:
  - Find all association rules with given *minsup* and *minconf*
- First, find all frequent itemsets
  - Start by finding one-item sets (easy)
  - *Q: How?*
  - A: Simply count the frequencies of all items

Finding itemsets: next level

- Apriori Algorithm (Agrawal & Srikant, 1993)
- **Idea**: use one-item sets to generate two-item sets, two-item sets to generate three-item sets, …
  - If {A, B} is a frequent item set, then {A} and {B} have to be frequent item sets as well! (subset property)
  - In general: if X is frequent *k*-item set, then all (*k*-1)-item subsets of X are also frequent

  $\Rightarrow$ Compute *k*-item set by merging (*k*-1)-item sets

An example

- Given: five frequent three-item sets

`(A B C), (A B D), (A C D), (A C E), (B C D)`

- Lexicographic order improves efficiency
- Candidate four-item sets:

`(A B C D)` **Q: OK?**

A: Yes, because all 3-item subsets are frequent

`(A C D E)` **Q: OK?**

A: No, because (C D E) is not frequent
Also: (A D E) is not frequent

Implementation Issues

- How to store support counts?
  - **First step**: convert strings to integers (using hash function)

  - Naïve method:
    - a[i,j] stores count for pair {i,j} (assume i<j)

  - Triangular Matrix Method:
    - a[k] stores count for pair {i,j} (assume i<j)
    - $k = (i – 1) (n – i/2) + j – i$
    - Stores data as: {1,2}, {1,3}, …, {1,n}, {2,3}, {2,4}, …, {2,n}, …, {n-1,n}

  - Triples Method:
    - Store triple [i,j,c] where c is count for pair {i,j} and i<j
    - Use hash table with i,j as key

[Source: http://www.mmds.org]

Beyond Binary Data

- Hierarchies
  - drink → milk → low-fat milk → Stop&Shop low-fat milk …
  - find associations on any level
- Sequences over time
- …