ASSOCIATION RULES

Introduction

- Important part of DM
- First introduced in 1993
- Association rule mining means finding interesting associations or correlations
- Usually expressed in the form of rules
- Ex: 90% of customers who buy bread also buy milk

  bread $\Longrightarrow$ milk

  antecedent $\Longrightarrow$ consequent
- Is it supervised or unsupervised learning?
- Initially used for Market Basket Analysis
- Other applications

Definitions

- Set of items: $I = \{i_1, i_2, \dots, i_m\}$
- A transaction: $t \subseteq I$
  For example, $t = \{i_2, i_5, i_{23}\}$ = {milk, bread, cheese}
- Database of transactions: $D = \{t_1, t_2, \dots, t_n\}$, where each $t_i \subseteq I$
- Itemset: $X \subseteq I$, or $\{i_{i1}, i_{i2}, \dots, i_{ik}\} \subseteq I$
  - E.g., X = {milk, bread, cheese} is an itemset
- k-itemset
  - E.g., X = {milk, bread, cheese} is is a 3-itemset.

Definitions (cont.)

- Support of an itemset X: Percentage of transactions in the database that contain that itemset.

  i.e., $\text{support}(X) = \frac{|\{t \in D : X \subseteq t\}|}{|D|}$

- Large (Frequent) itemset: Itemset whose support is at least a threshold, s.
- Notation:
  - L set of large itemsets
  - L_k set of large itemsets of size k

Example

Transaction	Items
t_1	Bread,Jelly,PeanutButter
t_2	Bread,PeanutButter
t_3	Bread,Milk,PeanutButter
t_4	Beer,Bread
t_5	Beer,Milk

- I = { Beer, Bread, Jelly, Milk, PeanutButter}
- What is the support of {Bread, PeanutButter}?
- For $s = 0.6$,
- Is {Bread,PeanutButter} frequent?
- Is {bread, Milk} frequent?

By convention, we list items in alphabetical order within a transaction

Large Itemsets

- Finding the large itemsets in a dataset is not a trivial process because
- 1) the number of transactions in the dataset can be large
- 2) the potential number of large itemsets is exponential to the number of different items
- It is important to have algorithms for discovering association rules that are scalable

The Large Itemest Property

Transaction	Items
t_1	Bread,Jelly,PeanutButter
t_2	Bread,PeanutButter
t_3	Bread,Milk,PeanutButter
t_4	Beer,Bread
t_5	Beer,Milk

- For $s = 0.6$, we notice that {Bread,PeanutButter} is frequent and so are all of its subsets
- We also notice that that {Jelly} is infrequent and so are all of its supersets

The Large Itemest Property

- Any subset of a large/frequent itemset is large/frequent
- Any superset of an infrequent itemset is infrequent
- Large itemsets are said to be downward closed

Association Rule Definitions

- **Association Rule** (AR): implication $X \Rightarrow Y$ where $X,Y \subseteq I$ and $X \cap Y = \emptyset$;
- Example: {Cheese, Milk} $\Rightarrow$ {Bread}
- **Support** of AR (s) $X \Rightarrow Y$: Percentage of transactions that contain $X \cup Y$
- **Confidence** or **strength** of AR (α) $X \Rightarrow Y$: Ratio of number of transactions that contain $X \cup Y$ to the number of transactions that contain X
- Remark: Confidence($X \Rightarrow Y$) equals to support($X \cup Y$)/support(X).
- Large confidence values and small support values are used for discovering Ars
- Aside: support ($X \Rightarrow Y$) = P(X union Y), and confidence($X \Rightarrow Y$) = P(Y|X)

Example

Transaction	Items
t_1	Bread,Jelly,PeanutButter
t_2	Bread,PeanutButter
t_3	Bread,Milk,PeanutButter
t_4	Beer,Bread
t_5	Beer,Milk

$X \Rightarrow Y$	s	α
Bread $\Rightarrow$ PeanutButter	60%	75%
PeanutButter $\Rightarrow$ Bread	60%	100%
Beer $\Rightarrow$ Bread	20%	50%
PeanutButter $\Rightarrow$ Jelly	20%	33.3%
Jelly $\Rightarrow$ PeanutButter	20%	100%
Jelly $\Rightarrow$ Milk	0%	0%

Association Rule Mining Task

- Def: Rules that satisfy both a minimum support threshold and minimum confidence threshold are called **strong**
- An association rule *r* is **strong** if
  - Support(r) ≥ *min_sup*
  - Confidence(r) ≥ *min_conf*
- Given a set of items, I, a transactions database D, *min_sup* , and *min_conf*, the goal of association rule mining is to find all *strong* rules

Association Rule Mining Task

- Two-step approach:
  1. Frequent Itemset Identification
     – find all itemsets whose support $\geq$ *min_sup*
  2. Rule Generation
     – from each frequent itemset, generate all rules whose confidence $\geq$ *min_conf*

– The naïve approach for the 1st step is costly

– Step 2 is straightforward

Algorithm to Generate ARs

Input:

D //**Database of transactions**

I //**Items**

L //**Large itemsets**

s //**Support**

α //**Confidence**

Output:

R //**Association Rules satisfying** s *and* α

ARGen Algorithm:

$R = \emptyset$;

for each $l \in L$ **do**

  for each $x \subset l$ **such that** $x \neq \emptyset$ **and** $x \neq l$ **do**

    if $\frac{support(l)}{support(x)} \geq \alpha$ **then**

     $R = R \cup \{x \Rightarrow (l - x)\}$;

Example

Transaction	Items
t_1	Bread,Jelly,PeanutButter
t_2	Bread,PeanutButter
t_3	Bread,Milk,PeanutButter
t_4	Beer,Bread
t_5	Beer,Milk

Apply Algorithm to the above dataset. Suppose $s = 0.3$, and $\alpha = 0.5$
L = {{Beer}, {Bread}, {Milk}, {PeanutButter} ,{Bread, PeanutButter}}

$$\frac{support(\{\text{Bread, PeanutButter}\})}{support(\{Bread\})} = 3/4 = 0.75$$

So, R = {Bread $\Rightarrow$ PeanutButter}

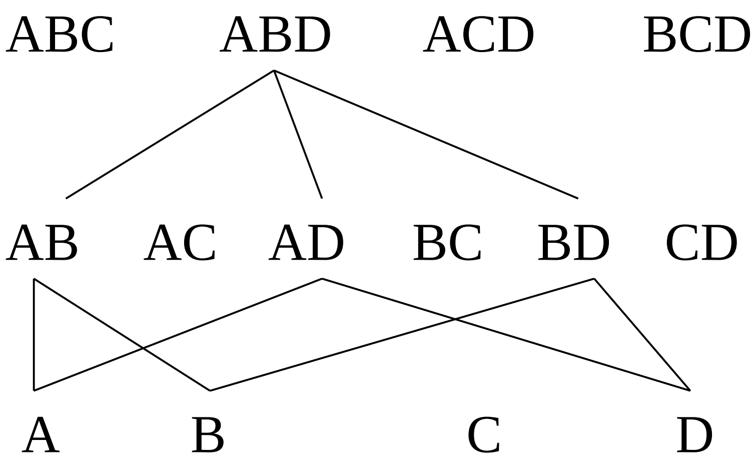
Notice, for $\alpha = 0.8$ the first rule would not be strong.

$$\frac{support(\{\text{Bread, PeanutButter}\})}{support(\{\text{PeanutButter}\})} = 3/3 = 1$$

So, R = {Bread $\Rightarrow$ PeanutButter, PeanutButter $\Rightarrow$ Bread}

The Apriori Algorithm for Generating Frequent Itemsets

- most well known AR mining algorithm
- It uses prior knowledge (L_k) to generate frequent itemset (L_{K+1})
- It uses the large itemset property (downward closure property): any subset of a frequent itemset is also frequent



Write an itemset {A, B, D} as ABD
ABD is frequent, then so are all of its subsets

The Algorithm

- Iterative algorithm (uses level-wise search, where L_k is used to find L_{k+1}):
  - ➢ Find all 1-item frequent itemsets; then all 2-item frequent itemsets, and so on.
  - ➢ In each iteration *k*, only consider itemsets that contain some *k*-1 frequent itemset.

- Find frequent itemsets of size 1: L_1
- From $k = 2$
  - C_k = candidates of size *k*: those itemsets of size *k* that could be frequent, given L_{k-1}
  - L_k = those itemsets that are actually frequent, $L_k \subseteq C_k$ (need to scan the database once).

Example – Finding frequent itemsets

Dataset T

minsup=0.5

TID	Items
T100	1, 3, 4
T200	2, 3, 5
T300	1, 2, 3, 5
T400	2, 5

itemset:count

1. scan T ➔ C_1: {1}:2, {2}:3, {3}:3, {4}:1, {5}:3

➔ L_1: {1}:2, {2}:3, {3}:3, {5}:3

➔ C_2: {1,2}, {1,3}, {1,5}, {2,3}, {2,5}, {3,5}

2. scan T ➔ C_2: {1,2}:1, {1,3}:2, {1,5}:1, {2,3}:2, {2,5}:3, {3,5}:2

➔ L_2: **{1,3}**:2, **{2,3}**:2, **{2,5}**:3, **{3,5}**:2

➔ C_3: {1, 2, 3}, {1,3,5}, {2, 3,5}

3. scan T ➔ C_3: {1, 2, 3}:1, {1,3,5}:1, **{2, 3, 5}**:2 ➔ L_3: **{2, 3, 5}**

Outline of Apriori Algorithm

1. C_1 = Itemsets of size one in I;
2. Determine all large itemsets of size 1, $L_{1;}$
3. i = 1;
4. Repeat
5. i = i + 1;
6. C_i = Apriori-Gen(L_{i-1});
7. Count C_i to determine $L_{i;}$
8. until no more large itemsets found;

How many DB scans?

Apriori-Gen

- Generate candidates of size i+1 from large itemsets of size i
- Approach used: join large itemsets of size i if they agree on i-1 items
- May also prune candidates who have subsets that are not large

The Apriori-Gen Algorithm – Algorithm

Input:

L_{i-1} //Large itemsets of size $i-1$

Output:

C_i //Candidates of size i

Apriori-gen algorithm:

$C_i = \emptyset$;

for each $I \in L_{i-1}$ **do**

  for each $J \neq I \in L_{i-1}$ **do**

    if $i-2$ of the elements in I and J are equal **then**

      $C_i = C_i \cup \{I \cup J\}$;

$$C_i \ = \ Ci \ \cup (I \cup J)$$

The Apriori Algorithm

```
Input:
    I      //Itemsets
    D      //Database of transactions
    s      //Support
Output:
    L      //Large itemsets
Apriori algorithm:
    k = 0; //k is used as the scan number.
    L = Ø;
```

```
C_1 = I;        //Initial candidates are set to be the items.
repeat
   k = k + 1;
   L_k = ∅;
   for each I_i ∈ C_k do
      c_i = 0;   // Initial counts for each itemset are 0.
   for each t_j ∈ D do
      for each I_i ∈ C_K do
         if I_i ∈ t_j then
            c_i = c_i + 1;
   for each I_i ∈ C_k do
      if c_i ≥ (s× | D |) do
         L_k = L_k ∪ I_i;
   L = L ∪ L_k;
   C_{k+1} = Apriori-Gen(L_k)
until C_{k+1} = ∅;
```

Example – Apriori

- Consider the following dataset

tid	itemset
t1	A, C, D
t2	B, C, E
t3	A, B, C, E
t4	B, E

A	B	C	D	E
X		X	X	
	X	X		X
X	X	X		X
	X			X

- Use Apriori to find all frequent itemsets using minimum support $s = 0.5$
- An itemset must appear in at least $0.5 * 4 = 2$ transactions to be frequent

An itemset must appear in at least 2 transactions to be frequent

Database TDB

tid	items
t1	A, C, D
t2	B, C, E
t3	A, B, C, E
t4	B, E

A	B	C	D	E
X		X	X	
	X	X		X
X	X	X		X
	X			X

1st scan

C_1

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

L_1

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

C_2

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

2nd scan

C_2

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

L_2

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

C_3

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

3rd scan

L_3

Itemset	sup
{B, C, E}	2