9. Data mining

- definition
- basic concepts
- applications
- challenges

Definition - Data Mining

- extraction of [unknown] patterns from data for actionability
- combines methods from:
  - databases
  - machine learning
  - visualization
- involves <u>large</u> datasets
- consists of:
  - stating the [business] question
  - data collection and (instance) selection
  - preprocessing
  - transformation
  - model building
  - interpretation/evaluation/deployment

Model building

- Supervised
  - (mainly classification)
  - Also ranking, estimation
- Unsupervised
  - Associations
  - Clustering

Associations

Given:

- $I = \{i1, \ldots, im\}$ set of items
- D set of transactions (a database), each transaction is a set of items $T \subset 2^I$

Association rule: $X \Rightarrow Y$, $X \subset I$, $Y \subset I$, $X \cap Y = 0$

confidence c: ratio of # transactions that contain both X and Y to # of *all* transaction that contain X

support s: ratio of # of transactions that contain both X and Y to # of transactions in D

Itemset is *frequent* if its support > θ

An *association rule* $A \Rightarrow B$ is a conditional implication among itemsets A and B, where $A \subset I$, $B \subset I$ and $A \cap B = \varnothing$.

Support of an association rule = $P(A \cup B)$. The *confidence* of an association rule r: $A \Rightarrow B$ is the conditional probability that a transaction contains B, given that it contains A. Confidence = $P(B|A)$

The support of rule r is defined as: $sup(r) = sup(A \cup B)$. The confidence of rule r can be expressed as $conf(r) = sup(A \cup B)/sup(A)$.

Formally, let $A \subset 2^I$; $sup(A) = |\{t: t \in D, A \subset t\}|/|D|$, if $R = A \Rightarrow B$ then $sup(R) = sup(A \cup B)$, $conf(R) = sup(A \cup B)/sup(A)$

Itemsets and association rules

- Itemset = set of items
- k-itemset = set of k items
- Finding association rules in databases:
  - Find all frequent (or large) itemsets (those with support > min_s
  - Generate rules that satisfy minimum confidence

Example

- Computer store
- Customers buying computers and financial software
- What does the rule mean:

computer ➔ financial_mgmt_software

[support = 2%, conf = 60%]

Associations - mining

Given D, generate all assoc rules with c, s > thresholds min_c, min_s

(items are ordered, e.g. by barcode)

Idea:

find all itemsets that have transaction support > min_s : large itemsets

Associations - mining

to do that: start with indiv. items with large support

in ea next step, k,

- use itemsets from step k-1, generate new itemset C_k,
- count support of C_k (by counting the candidates which are contained in any t),
- prune the ones that are not large

Apriori property

- All [non-empty] subsets of a frequent itemset must be frequent

- Based on the fact that an itemset *i* that is NOT frequent has support < min_s
- But inserting an additional item *A* in *i* will not increase the support of $i \cup A$

Associations - mining

```
L1 = {large 1-itemsets};
for ( k = 2; Lk-1 ≠ ∅; k++ ) do begin
    Ck = apriori-gen(Lk-1);  // New candidates
    forall transactions t ∈ D do begin
        Ct = subset(Ck, t);  // Candidates contained in t
        forall candidates c ∈ Ct do
            c.count++;
    end
    Lk = {c ∈ Ck | c.count ≥ minsup}
end
Answer = ∪k Lk;
```

subset(c_k,ℓ) denotes those itemsets that are contained In transaction *ℓ*

Candidate generation

C_k = apriori-gen(L_{k-1})

insert into C_k
select $p.\text{item}_1$, $p.\text{item}_2$, ..., $p.\text{item}_{k-1}$, $q.\text{item}_{k-1}$
from L_{k-1} p, L_{k-1} q
where $p.\text{item}_1 = q.\text{item}_1, \ldots, p.\text{item}_{k-2} = q.\text{item}_{k-2}$,
$p.\text{item}_{k-1} < q.\text{item}_{k-1}$;

Next, in the *prune* step, we delete all itemsets $c \in C_k$ such that some $(k-1)$-subset of c is not in L_{k-1}:

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

Select from *k-1*-frequent itemsets two overlapping subsets, add the differences

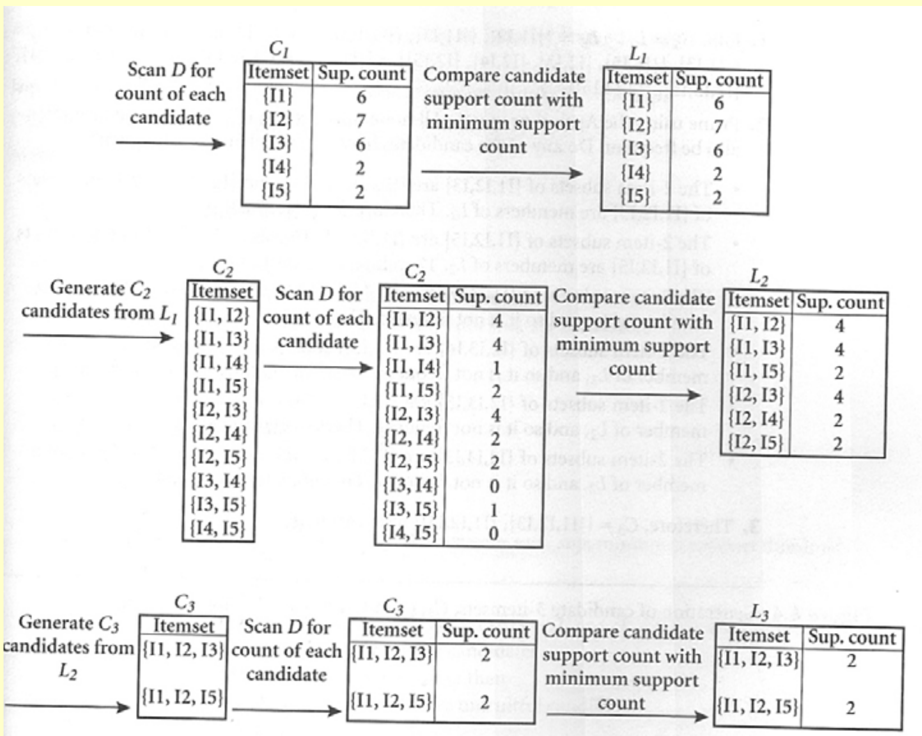
Example

TID	List of item_IDs
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

From Han, Kamber,"Data Mining", p. 232

$I = \{I1,\ldots,I5\}$

$min_s = 2$



C_1

Itemset	Sup. count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

L_1

Itemset	Sup. count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

C_2

Itemset
{I1, I2}
{I1, I3}
{I1, I4}
{I1, I5}
{I2, I3}
{I2, I4}
{I2, I5}
{I3, I4}
{I3, I5}
{I4, I5}

C_2

Itemset	Sup. count
{I1, I2}	4
{I1, I3}	4
{I1, I4}	1
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2
{I3, I4}	0
{I3, I5}	1
{I4, I5}	0

L_2

Itemset	Sup. count
{I1, I2}	4
{I1, I3}	4
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2

C_3

Itemset
{I1, I2, I3}
{I1, I2, I5}

C_3

Itemset	Sup. count
{I1, I2, I3}	2
{I1, I2, I5}	2

L_3

Itemset	Sup. count
{I1, I2, I3}	2
{I1, I2, I5}	2

Firstly, C3 ={{I1,I2,I3},{I1,I2,I5},{I1,I3,I5}{I2,I3,I4},{I2,I3,I5},{I2,I4,I5}}
Only {I1,I2,I3},{I1,I2,I5} are left
C4={I1,I2,I3,I5} is attempted but pruned, C4=Ø terminates the algorithm

From itemsets to association rules

- For ea. frequent itemset *I* generate all the partitions of *I* into *s, I-s*
- Attempt a rule *s* ➔ *I-s* iff *support_count(I)/support_count(s)* > min_c
- e.g. for min_c = 0.5, what rules do we get?

[$conf(r) = sup(A \cup B)/sup(A)$]