Market BasketT Analysis :

Market Basket Analysis

Scenario: customers shopping at a supermarket

Transaction id	Items
1	Bread, Ham, Juice, Cheese, Salami, Lettuce
2	Rice, Dal, Coconut, Curry leaves, Coffee, Milk, Pickle
3	Milk, Biscuit, Bread, Salami, Fruit jam, Egg
4	Tea, Bread, Salami, Bacon, Ham, Sausage, Tomato
5	Rice, Egg, Pickle, Curry leaves, Coconut, Red chilly

- What can we infer from the above data?
- An association rule: {Bread, Salami} → {Ham}, with confidence ≈ 2/3

Support (I) = No. of baskets containing I.

This is the threshold used while designing a recommendation system.

Association Rule :

Say customers who buy (i, j) also tend to buy 'k'

Association Rule: $(i,j) \longrightarrow k$

Confidence :

Confidence of an association rule is given by $\dfrac{n\{(i,j) \cup k\}}{n(i,j)}$

Applications

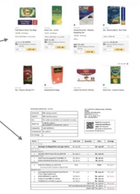
- Information driven marketing
  - Since you viewed *this* product, you may also be interested in *that* product
- Catalog design
- Store layout
  - Make it easy for customers to find products of interest
- Customer segmentation based on buying patterns

- Several papers by Rakesh Agrawal and others in the 1990s
- Rakesh Agrawal and Ramakrishnan Srikant
  Fast Algorithms for Mining Association Rules, VLDB 1994

High confidence asso. rule can sometimes be unimportant because it is just a "common" transaction not a "frequent" transaction.

The Market-Basket Model

- A (large) set of *binary attributes*, called *items*: $I = \{i_1, \ldots, i_n\}$
  e.g. milk, bread, tea: the items sold at the market
- A *transaction* T consists of a (small) subset of I
  e.g. the list of items (bill) bought by one customer at once
- The *database* D is a (large) set of transactions: $D = \{T_1, \ldots, T_n\}$



The Market-Basket Model

- Goal: mining associations between the *items*
  - The transactions or customers also may have associations, but here we are interested in such relations
- Approach: finding subset of items that are present together in transactions frequently (bought together frequently)
- An *itemset*: any subset X of I

Items = Basket
Transactions = Items

Goal: Find frequent itemsets

Support of an Itemset

- Let X be an itemset
- Support count $\sigma(X)$ = # of transactions containing all items of X
- support(X) = fraction of transactions containing all items of X

T.id	Items
1	Bread, Ham, Juice, Cheese, Salami, Lettuce
2	Rice, Dal, Coconut, Curry leaves, Coffee, Milk, Pickle
3	Milk, Biscuit, Bread, Salami, Fruit jam, Egg
4	Tea, Bread, Salami, Bacon, Ham, Sausage, Tomato
5	Rice, Egg, Pickle, Curry leaves, Coconut, Red chilly

support({Bread, Salami}) = 0.6

support({Rice, Pickle, Coconut}) = 0.4

- An association rule would make sense only when support count is at least a few hundreds in a database of several thousand transactions

NOTE : The terminology may vary, but the concept is consistent

Association Rule

- Association rule: an implication of the form $X \to Y$, where $X, Y \subseteq I$, and $X \cap Y = \phi$.
- $\text{support}(X \to Y) = \dfrac{\sigma(X \cup Y)}{|D|}$
  - Fraction of transactions containing all items of *both* X and Y
- $\text{confidence}(X \to Y) = \dfrac{\sigma(X \cup Y)}{\sigma(X)}$
  - For the transactions containing all items of X, the fraction of transactions containing all items of Y (*both* X and Y)

T.id	Items
1	Bread, Ham, Juice, Cheese, Salami, Lettuce
2	Rice, Dal, Coconut, Curry leaves, Coffee, Milk, Pickle
3	Milk, Biscuit, Bread, Salami, Fruit jam, Egg
4	Tea, Bread, Salami, Bacon, Ham, Sausage, Tomato
5	Rice, Egg, Pickle, Curry leaves, Coconut, Red chilly

Example: a rule R: {Bread, Salami} → {Ham}

$\text{support}(R) = \dfrac{2}{5}$

NOTE: Both - High Confidence and Low Confidence Association rules are important in decision making.

minconf ≥ 0.3 or 0.4

minsup ≥ 0.01

Association Rule Mining Task

- Given a set of items I, a set of transactions D, a minimum support threshold *minsup* and a minimum confidence threshold *minconf*

- Find all rules R such that
  support(R) ≥ *minsup*
  confidence(R) ≥ *minconf*

★★

One Approach

- Let $Z = X \cup Y$
- Observe:

$$\text{support}(X \to Y) = \frac{\sigma(X \cup Y)}{|D|} = \frac{\sigma(Z)}{|D|} = \text{support}(Z)$$

Example: support({Bread, Ham} → {Salami}) = support({Bread, Ham, Salami})

- If $Z = W \cup V$ for some other itemsets W and V, then support($X \to Y$) = support($W \to V$)
  - Each binary partition of Z represents an association rule
  - With same support
  - However, the confidences may be different

Example: support({Bread} → {Ham, Salami}) = support({Bread, Ham} → {Salami})

- Approach: frequent itemset generation
  1. Find all itemsets Z with support(Z) ≥ *minsup*. Call such itemsets *frequent itemsets.*
  2. From each Z, generate rules with confidence(Z) ≥ *minconf*

Finding Frequent Itemsets

- If $|I| = n$, then number of possible itemsets = 2^n
- Naïve approach: For each itemset, compute the support by scanning the lists of items of each transaction
  - Complexity: $O(N \times w)$, where w is the average length of transactions
- Overall complexity: $O(2^n \times N \times w)$
- Computationally very expensive!!

$|D|$

$D = \{t_1, t_2, t_3 \dots, t_N\}$

Fundamental Concept behind A-Priori algorithm

Anti-monotone Property of Support

- If an itemset is *frequent,* all its subsets are also *frequent*
  - Because if $X \subseteq Y$, then support(X) ≥ support(Y)
  - For all transactions T such that $Y \subseteq T$, we have $X \subseteq T$

Example: if {Bread, Ham, Salami} are bought together frequently, then {Bread, Salami} are also bought together at least those many times.

$3/5$

support({Bread, Salami}) ≥ support({Bread, Ham, Salami})

$\frac{2}{5}$

T-ID	Items
1	Bread, Ham, Juice, Cheese, Salami, Lettuce
2	Rice, Dal, Coconut, Curry leaves, Coffee, Milk, Pickle
3	Milk, Biscuit, Bread, Salami, Fruit jam, Egg
4	Tea, Bread, Salami, Bacon, Ham, Sausage, Tomato
5	Rice, Egg, Pickle, Curry leaves, Coconut, Red chilly

The A-Priori Algorithm: The Approach

(a) Find the set of *frequent itemsets* of size 1 (single items): potentially a lot of them
(b) Having found the frequent itemsets of size $k - 1$, find the frequent itemsets of size k

say 2

- Use anti-monotone property: a frequent itemset of size k must be such that all its proper subsets are also frequent itemsets (of size $< k$, hence they are already found)
- Construct candidate itemsets of size k from known frequent itemsets of size $k - 1$
- Prune invalid ones to preserve anti-monotone property
- Compute support for the candidates and keep the ones which pass the threshold

Presented by: Rakesh Agrawal.
(Paper Title on prev. page.)

The A-Priori Algorithm : Pseudocode

Notation:
L_k = The set of frequent (large) itemsets of size k.
C_k = The candidate set of frequent (large) itemsets of size k.

Algorithm:

```
L_1 = {Frequent 1-itemsets};
for ( k = 2; L_{k−1} ≠ 0; k++ ) do begin
    C_k = apriori_gen( L_{k-1} );   /* Generate new candidates and prune invalid ones */
    for all transactions T in D do begin
        C_T = subset( C_k, T )    /* Find which itemsets are included in T */
        for all candidates c in C_T do
            c.count++;
        end
    L_k = {c in C_k | c.count ≥ minsup}
end
```

Generating set of candidate itemsets C_k from L_{k-1}

- A join of L_{k-1} with itself

  insert into C_k
  select p.item$_1$, p.item$_2$, … , 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$_{k-1}$

this query inserts all items from left table and join it on last entry of right table

- What does it do?

k=4

L_3	L_3
{1, 2, 3}	{1, 2, 3}
{1, 2, 4}	{1, 2, 4}
{1, 3, 4}	{1, 3, 4}
{1, 3, 5}	{1, 3, 5}
{2, 3, 4}	{2, 3, 4}

C_4 = { {1, 2, 3, 4}, {1, 3, 4, 5} }

(1,2,3), (1,2,4)
(1,3,4), (2,3,4)

A prune step to eliminate invalid itemsets:

{1, 3, 4, 5} will be pruned because {1, 4, 5} $\notin L_3$

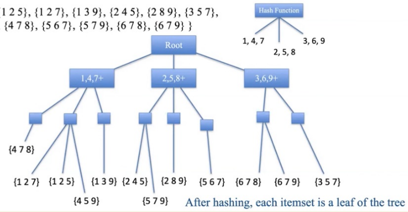
Checking Support for candidates

- Straightforward approach:
  - for each candidate itemset $c \in C_k$
    - for each transactions $T \in D$ do begin
      - check if $c \subseteq T$ — The subset operation
      - if yes, increase support count of c
    - end
  - end
- We want to perform the above much faster

eg.: for c in {(1,2,3,4)}
for t in D
if (1,2,3,4) in t
count (1,2,3,4)++
if final-count ≥ support:
it is freq. itemset

The Hash Tree

C_3 = {{1 2 5}, {1 2 7}, {1 3 9}, {2 4 5}, {2 8 9}, {3 5 7}, {4 5 9}, {4 7 8}, {5 6 7}, {5 7 9}, {6 7 8}, {6 7 9} }



After hashing, each itemset is a leaf of the tree

Where are we now?

- Computed *frequent itemsets*, i.e. the itemsets with required support *minsup*
- Each frequent k-itemset X gives rise to several association rules
- How many?
- Ignoring $X \to \phi$ and $\phi \to X$, $\boxed{2^k - 2 \text{ rules}}$
- **Next step:**
  - Generate rules from the frequent itemsets
  - The rules need to be checked for minimum confidence
  - (All these rules already satisfy the support condition because the itemsets do so)

Rules Generated from the Same Itemset

- Let $X \subset Y$, for non empty itemsets X, and Y
- Then $X \to Y - X$ is an association rule
- *Theorem*: If $X' \subset X \subset Y$, then $c(X \to Y - X) \geq c(X' \to Y - X')$

$Y = \{1,2,3,4,5\}$
$X = \{1,2,3\}$

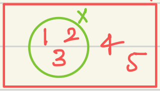
eg.: $c(\{1,2,3\} \to \{4,5\}) \geq c(\{1,2\} \to \{3,4,5\})$

$$c(X \to Y - X) = \frac{\sigma(X) \cup \sigma(Y-X)}{\sigma(X)} = \frac{\sigma(Y)}{\sigma(X)}$$

$$X' = \{1,2\} \quad c(X' \to Y - X') = \frac{\sigma(Y)}{\sigma(X')}$$

Clearly $c(X \to Y - X) \leq c(X' \to Y - X')$

given $X' \subset X$



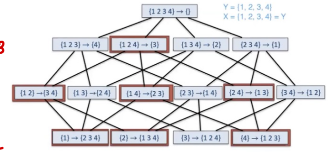
Level-wise Approach for Rule Generation

Frequent itemset: {1 2 3 4}

$4C_3$

$4C_2$

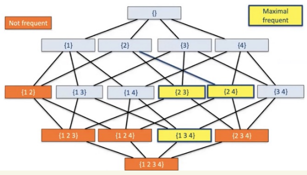
$4C_1$



- Suppose {1 2 4} → {3} fails the confidence bar
- Then all rules in the subtree under {1 2 4} → {3} can be discarded

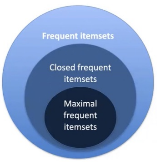
Maximal Frequent itemsets

All frequent itemsets are subsets of one of the maximal frequent itemsets.



Closed Frequent Itemsets

- *Closed itemset*: an itemset X for which none of its *immediate supersets* has exactly the *same support count* as X
  - If X is not closed, at least one of its immediate supersets have the same support as the support of X
- *Closed frequent itemset*: an itemset which is *closed* and *frequent* (support ≥ *minsup*)
- Support for non-closed frequent itemsets can be determined from the support information of the closed frequent itemsets



Subjective Measure of Interestingness

- The rule {Salami} → {Bread} is *not so interesting* because it is *obvious!*
- Rules such as {Salami} → {Dish washer detergent}, {Salami} → {Diper}, etc are less obvious
- Subjectively more interesting for marketing experts
  - Non-trivial cross sell
- Methods for subjective measurement
  - Visualization aided: human in the loop
  - Template-based: constrains are provided for rules

Contingency Table

	Coffee	~~Coffee~~	
Tea	150	50	200
~~Tea~~	650	150	800
	800	200	1000

	B	B'	
A	f_{11}	f_{10}	f_{1+}
A'	f_{01}	f_{00}	f_{0+}
	f_{+1}	f_{+0}	

- Frequency tabulated for a pair of binary variables
- Used as a useful evaluation and illustration tool
- Generally:
  - A' (or B') denotes the transactions in which A (or B) is absent
  - f_{1+} = support count of A
  - f_{+1} = support count of B

Lift :

$$\text{Lift}(X \rightarrow Y) = \frac{\sup(X \rightarrow Y)}{\sup(Y)}$$

Interest Factor :

$$I(X,Y) = \frac{s(X \cup Y)}{s(X) \cdot s(Y)} = \frac{N f_{11}}{f_{1+} \cdot f_{+1}}$$

More Measures

- Correlation coefficient for binary variables:

$$\boxed{\phi = \frac{f_{11}f_{00} - f_{01}f_{10}}{\sqrt{f_{1+}f_{+1}f_{0+}f_{+0}}}}$$

- IS Measure: I and S measures combined

$$\boxed{IS(A,B) = \sqrt{I(A,B) \times s(A,B)} = \frac{s(A,B)}{\sqrt{s(A)s(B)}}}$$

- Mathematically equivalent to cosine measure of binary variables