Mining Frequent patterns, Associations & Correlations

Shubham kumar
Dept. of CS&IT
MGCU Bihar

Frequent Item set mining

- Item set:- set of items.

  Example- {computer, printer, MS office software} is 3- item set.

  { milk, bread} is 2-item set.

  similarly set of K items is called k-item set.

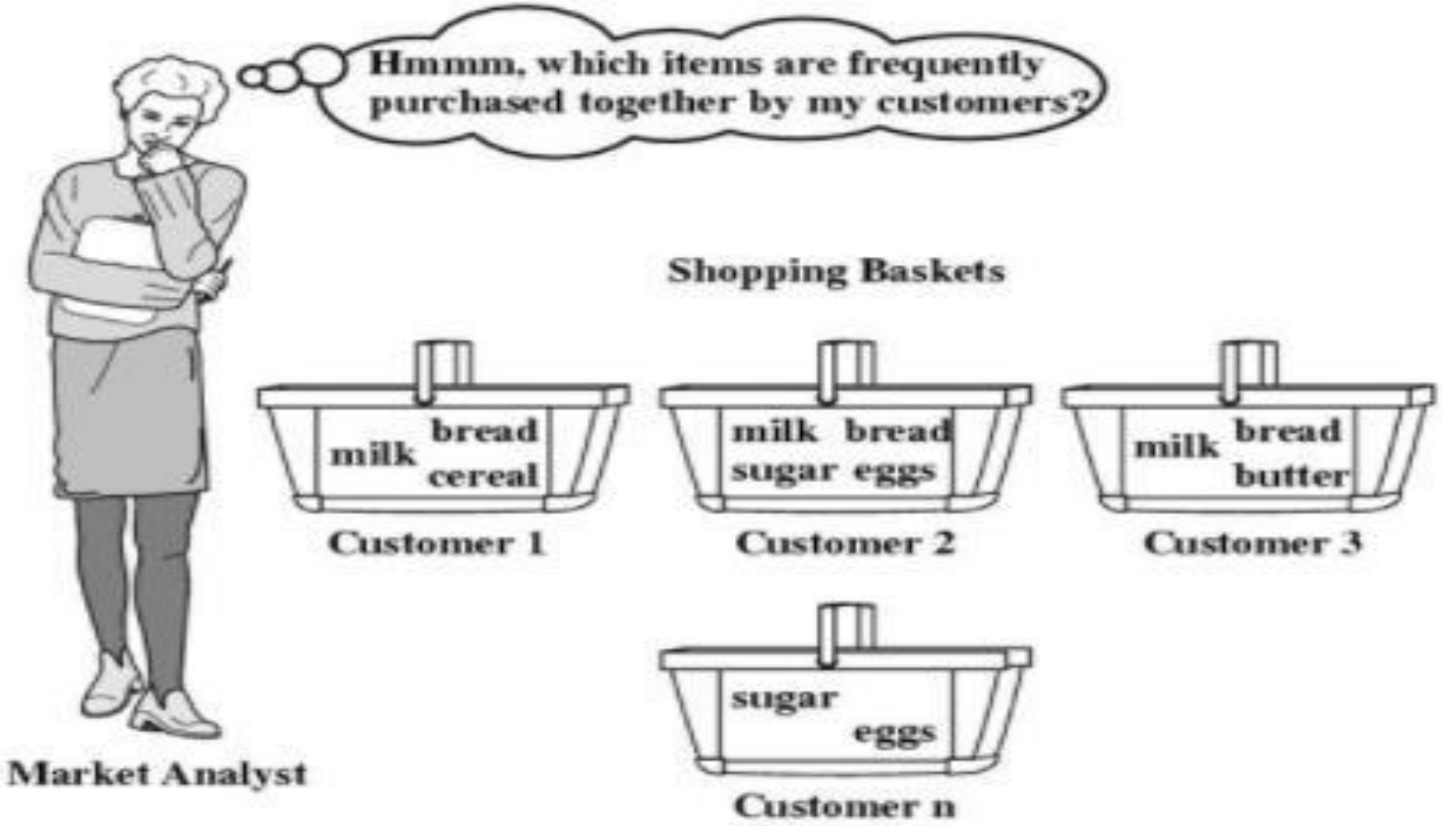
- Frequent patterns are patterns that appear frequently in a data set. Patterns may be itemsets, subsequences or substructures.

  Example: A set of items, such as Milk & Butter that appear together in a transaction data set. (Also called **Frequent Item set**).

- Frequent item set mining leads to the discovery of associations and correlations among items in large transactional (or) relational data sets.

- This helps in many business decision- making processes like Catalog design, and customer shopping behavior analysis, etc.

- **Market Basket Analysis**: This is the example of frequent item set mining. This process analyzes customer buying habits by finding associations between different items that customer places in their shopping baskets.



- Retailers can use the result by placing the items that are frequently purchased together in proximity to further encourage the combined sale of such items.

In our example(in the figure), Milk and bread is frequent, so it can be kept in proximity.

Other example is, if customers who purchase computers also tend to buy printer at the same time, then placing the hardware display close to the printer may increase the sale of both the items.

<u>Association rules:</u>

Let $I = \{ I_1, I_2, I_{3,} \ldots\ldots\ldots, I_m \}$ be an item set.

$D = \{ T_1, T_2, T_{3,} \ldots\ldots\ldots, T_n \}$ be a set of n transactions where each transaction T_i is non- empty item set such that $T \sqsubseteq I$.

[or]

for each i, $T_i \neq \Phi$ and $T_i \sqsubseteq I$

Let A and B are set of items.

[ex- A= { I_1, I_3, $I_{7,}I_8$ } and B= { I_4, I_5, I_6}]

An Association rule is an implication of the form

$$A \Longrightarrow B$$

where $A \subset I$, $B \subset I$, $A \neq \Phi$ and $B \neq \Phi$ & $A \cap B \neq \Phi$

The rule $A \Longrightarrow B$ holds in the transaction set D with **Support** s and **Confidence** c.

Support: This is the percentage of transaction in D that contain $A \cup B$. Here $A \cup B$ means every item in A and every item in B. Support is also written as $P(A \cup B)$. It is also called **Relative support**.

[Note: $(A \cup B) \neq$ A or B]

- Therefore,

$$\text{Support } (A \Longrightarrow B) = P(A \cup B).$$

Confidence: This is the percentage of transactions in D containing A that also contain B. It is also written as P(B/A).

$$\text{Confidence}(A \Longrightarrow B) = P(B/A).$$

$$= \frac{\text{support}(A \cup B)}{\text{support}(A)}$$

$$= \frac{\text{support count } (A \cup B)}{\text{support count } (A)}$$

Support count or Frequency: Number of transactions that contain the item set. It is also called **Absolute support**.

- Any association rules that satisfy both a minimum support threshold(min_sup) and minimum confidence threshold (min_conf) are called **strong** association.

- We have seen in the previous slide that the confidence can easily be derived from the support counts. i.e. If support counts of A, B and A⋃B are found, then we can derive corresponding association rules $A \Longrightarrow B$ and $B \Longrightarrow A$ and check whether they are strong or not.

- Hence mining association rules can be viewed as a two step process:

1. Finding all frequent item sets and
2. Generate strong association rules from the frequent item sets.

[Note: frequent item set are those item sets that satisfies the min_sup]

- **Closed Frequent item set:** An itemset X is **closed** in a data set D if there exists no proper super-itemset Y such that Y has the same support count as X in D.

An itemset X is a **closed frequent itemset** in data set D if X is both closed and frequent.

Maximal Frequent itemset: An itemset X is a maximal frequent itemset in a data set D if X is frequent and there exist no super-itemset Y such that $X \subset Y$ and Y is frequent in D.

Example: Let $T_1 = (a_1,a_2,a_3,a_4,a_5)$ and $T_2= (a_1,a_2,a_3)$

and minimum support count threshold min_sup=1

Therefore, **Set of closed frequent itemset** C= { $\{a_1,a_2,a_3\}$= 2; $\{a_1,a_2,a_3,a_4,a_5\}$=1}.

and **Set of maximal frequent itemset** M= {$\{a_1,a_2,a_3,a_4,a_5\}$=1}.

Apriori algorithm: (For finding frequent itemsets)

It is an iterative approach where k-itemsets are used to explore (k+1) itemsets.

Steps:

[1] The set of frequent 1-itemset is found by scanning the data base and selecting those whose support count satisfy the minimum support. And denote this set as $L_{1.}$

[2] L_1 is used to find set of frequent 2-itemset say L_2 .

[3] Further L_2 is used to find L_3 and so on until no more frequent k-itemset can be found.

[Note: the finding of each L_k requires one full scan of the database.]

Finding L_k (k>=2) :

[1] Join step:

{ Assumption: 1. itemsets are sorted in lexicographic order.

2. l_i [j] means j^{th} item in l_i .

}

The join ($L_{k-1} \bowtie L_{k-1}$) (say it C_k) is performed where members of L_{k-1} are joinable if their first (k-2) items are in common.

i.e. Members l_1 and l_2 of L_{k-1} are joined **if** ($l_1[1] = l_2[1]$ ^ $l_1[2]= l_2[2]$ ^ $l_1[3]= l_2[3]$ ^ ……………. ^ $l_1[k-2]= l_2[k-2]$ ^ $l_1[k-1] < l_2[k-1]$)

[condition $l_1[k-1] < l_2[k-1]$ ensures no duplicity]

Therefore, resulting itemset formed by joining l_1 and l_2 is {$l_1[1]$, $l_1[2]$, $l_1[3]$, $l_1[k-2]$, $l_2[k-1]$}

Example:

Let L_2 = [{I_1, I_2}, {I_1, I_3}, {I_1, I_5}]

then,

$L_2 \bowtie L_2$ (i.e. C_3) = [{I_1, I_2, I_3}, {I_1, I_2, I_5},{I_1, I_3, I_5}]

[2] prune step:

The support count of each itemset in C_k is calculated and determine L_k by putting all those itemsets which satisfy the min_sup in C_k .

[Note: To determine the support count of each candidate in C_k a complete database scan is needed. Therefore to reduce the size of C_k the **Apriori property** is used.

Apriori property: if an itemset I does not satisfy the minimum support threshold then (I ∪ A) also will not satisfy the min_sup.]

Therefore if any (k-1) subset of a candidate k-itemset is not in L_{k-1} , then the candidate can't be frequent (i.e. does not satisfy min_sup) hence can be removed from C_k .

Example:

Consider the following dataset and for this we have to find frequent itemsets and also have to generate association rules for them

TID	List of items_IDs
T1	I1,I2,I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

Let min_sup = 2

Transactional Dataset D

Step 1: create a table C1 that contain support count of each item present in the dataset D.

C1

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

Now, Compare candidate support count with minimum support count. This gives itemset L1.

L1

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

Step 2: Generate C2 candidates from L1 (join step), and scan D for count of each candidate.

C2

Itemset	Support 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

Compare candidate support count with minimum support count. This gives itemset L2.

L2

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

Step 3: Generate candidate set C3 using L2 (join step). And scan D for count of each candidate.

$L_2 \bowtie L_2$

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

But, using Apriori property we can remove {I1, I3, I5},{I2, I3, I4},{I2, I4, I5} and {I2, I3, I5} because every subsets of these sets are not frequent.

Example- for itemset {I1,I3,I5} subset {I3,I5} is not frequent. And for {I2, I3, I4} subset {I3, I4} is not frequent.

Therefore,

C3

Itemset	Support count
{I1,I2,I3}	2
{I1,I2,I5}	2

Compare candidate support count with minimum support count. This gives itemset L3.

L3

Itemset	Support count
{I1,I2,I3}	2
{I1,I2,I5}	2

- Step 4: Generate candidate set C4 using L3 (join step). And scan D for count each candidate.

$L_3 \bowtie L_3$ **{I1,I2,I3,I5}**

Therefore C4 = Φ

Because the subset {I1, I3, I5} of itemset {I1, I2, I3, I5} is not frequent so there is no itemset in C4.

Hence algorithm terminated .

we have discovered all the frequent item-sets.

In next lecture we will see the generation of strong association rules and pseudocode for Apriori algorithm.

Reference

- Jiawei Han, Micheline kamber and Jian pei. “DATA MINING concepts and Techniques” 3/e, Elsevier, 2012