- Apriori Algorithm in Data Mining

What Is An Itemset?

- A set of items together is called an itemset. If any itemset has k-items it is called a k-itemset. An itemset consists of two or more items. An itemset that occurs frequently is called a frequent itemset. Thus frequent itemset mining is a data mining technique to identify the items that often occur together.

- For Example, Bread and butter, Laptop and Antivirus software, etc.

What Is A Frequent Itemset?

- A set of items is called frequent if it satisfies a minimum threshold value for support and confidence. Support shows transactions with items purchased together in a single transaction. Confidence shows transactions where the items are purchased one after the other.
- For frequent itemset mining method, we consider only those transactions which meet minimum threshold support and confidence requirements. Insights from these mining algorithms offer a lot of benefits, cost-cutting and improved competitive advantage.
- There is a tradeoff time taken to mine data and the volume of data for frequent mining. The frequent mining algorithm is an efficient algorithm to mine the hidden patterns of itemsets within a short time and less memory consumption.

Frequent Pattern Mining (FPM)

- The frequent pattern mining algorithm is one of the most important techniques of data mining to discover relationships between different items in a dataset. These relationships are represented in the form of association rules. It helps to find the irregularities in data.
- FPM has many applications in the field of data analysis, software bugs, cross-marketing, sale campaign analysis, market basket analysis, etc.
- Frequent itemsets discovered through Apriori have many applications in data mining tasks. Tasks such as finding interesting patterns in the database, finding out sequence and Mining of association rules is the most important of them.
- Association rules apply to supermarket transaction data, that is, to examine the customer behavior in terms of the purchased products. Association rules describe how often the items are purchased together.

Association Rules

- ***Association Rule Mining is defined as:***

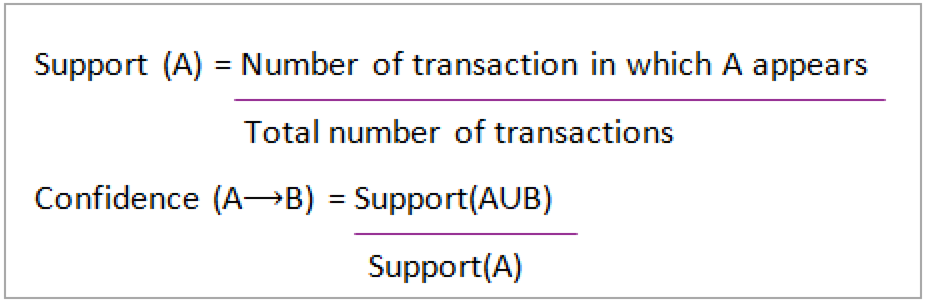
  "Let I= { …} be a set of 'n' binary attributes called items. Let D= { ….} be set of transaction called database. Each transaction in D has a unique transaction ID and contains a subset of the items in I. A rule is defined as an implication of form X->Y where X, Y? I and X?Y=?. The set of items X and Y are called antecedent and consequent of the rule respectively."
- Learning of Association rules is used to find relationships between attributes in large databases. An association rule, A=> B, will be of the form" for a set of transactions, some value of itemset A determines the values of itemset B under the condition in which minimum support and confidence are met".

- Support and Confidence can be represented by the following example:

Bread=> butter [support=2%, confidence-60%]

The above statement is an example of an association rule. This means that there is a 2% transaction that bought bread and butter together and there are 60% of customers who bought bread as well as butter.

- Support and Confidence for Itemset A and B are represented by formulas:



$$\text{Support (A)} = \frac{\text{Number of transaction in which A appears}}{\text{Total number of transactions}}$$

$$\text{Confidence } (A \longrightarrow B) = \frac{\text{Support(AUB)}}{\text{Support(A)}}$$

- Association rule mining consists of 2 steps:

1. Find all the frequent itemsets.
2. Generate association rules from the above frequent itemsets.

Why Frequent Itemset Mining?

- Frequent itemset or pattern mining is broadly used because of its wide applications in mining association rules, correlations and graph patterns constraint that is based on frequent patterns, sequential patterns, and many other data mining tasks.

Apriori Algorithm – **Frequent Pattern Algorithms**

Apriori algorithm was the first algorithm that was proposed for frequent itemset mining. It was later improved by R Agarwal and R Srikant and came to be known as Apriori. This algorithm uses two steps "join" and "prune" to reduce the search space. It is an iterative approach to discover the most frequent itemsets.

Apriori says:

The probability that item I is not frequent is if:

- P(I) < minimum support threshold, then I is not frequent.
- P (I+A) < minimum support threshold, then I+A is not frequent, where A also belongs to itemset.
- If an itemset set has value less than minimum support then all of its supersets will also fall below min support, and thus can be ignored. This property is called the Antimonotone property.

The steps followed in the Apriori Algorithm of data mining are:

- **Join Step**: This step generates (K+1) itemset from K-itemsets by joining each item with itself.
- **Prune Step**: This step scans the count of each item in the database. If the candidate item does not meet minimum support, then it is regarded as infrequent and thus it is removed. This step is performed to reduce the size of the candidate itemsets.

Steps In Apriori

Apriori algorithm is a sequence of steps to be followed to find the most frequent itemset in the given database. This data mining technique follows the join and the prune steps iteratively until the most frequent itemset is achieved. A minimum support threshold is given in the problem or it is assumed by the user.

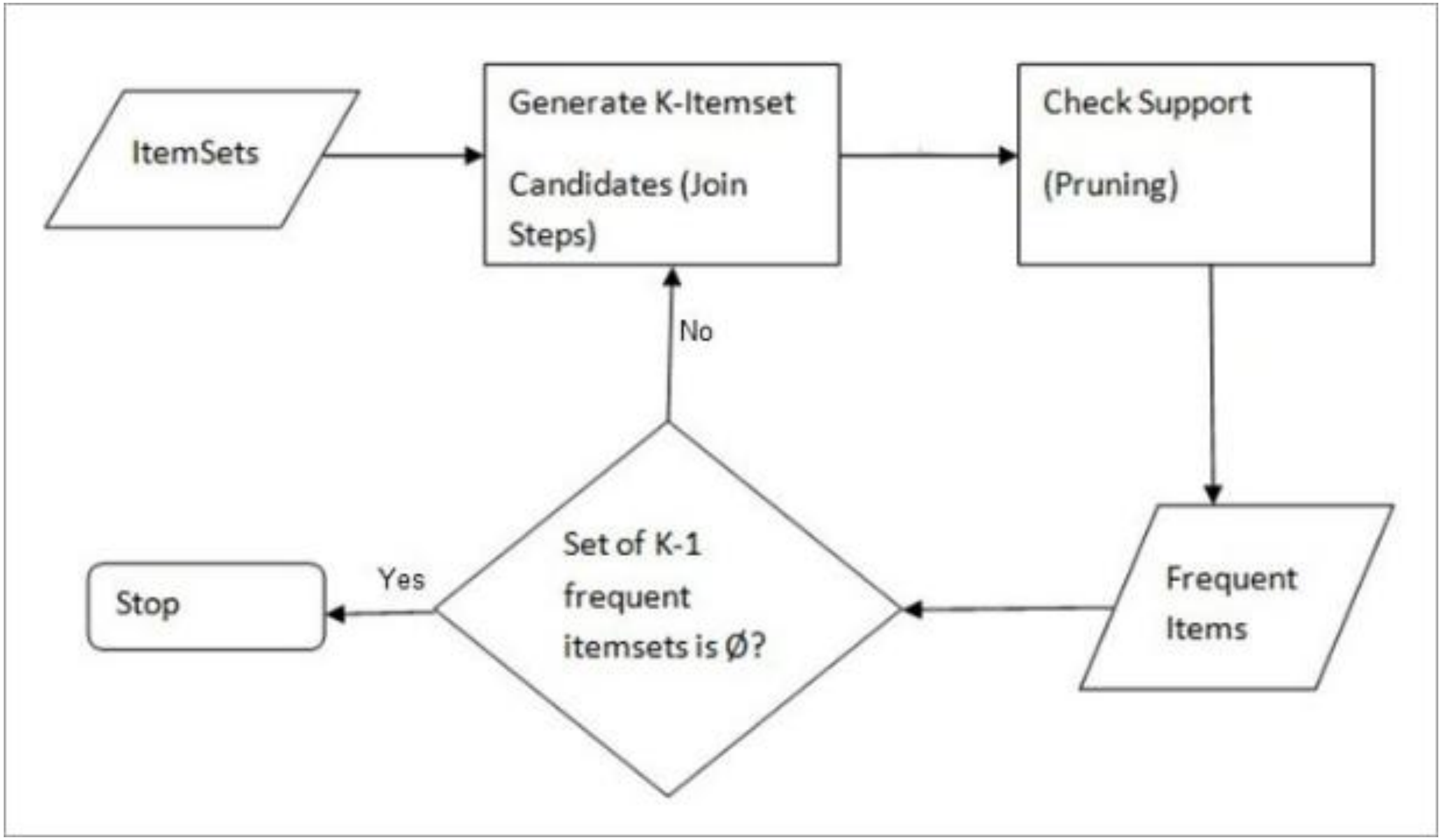
- **#1)** In the first iteration of the algorithm, each item is taken as a 1-itemsets candidate. The algorithm will count the occurrences of each item.
- **#2)** Let there be some minimum support, min_sup (eg 2). The set of 1 – itemsets whose occurrence is satisfying the min sup are determined. Only those candidates which count more than or equal to min_sup, are taken ahead for the next iteration and the others are pruned.

- **#3)** Next, 2-itemset frequent items with min_sup are discovered. For this in the join step, the 2-itemset is generated by forming a group of 2 by combining items with itself.

- **#4)** The 2-itemset candidates are pruned using min-sup threshold value. Now the table will have 2 –itemsets with min-sup only.

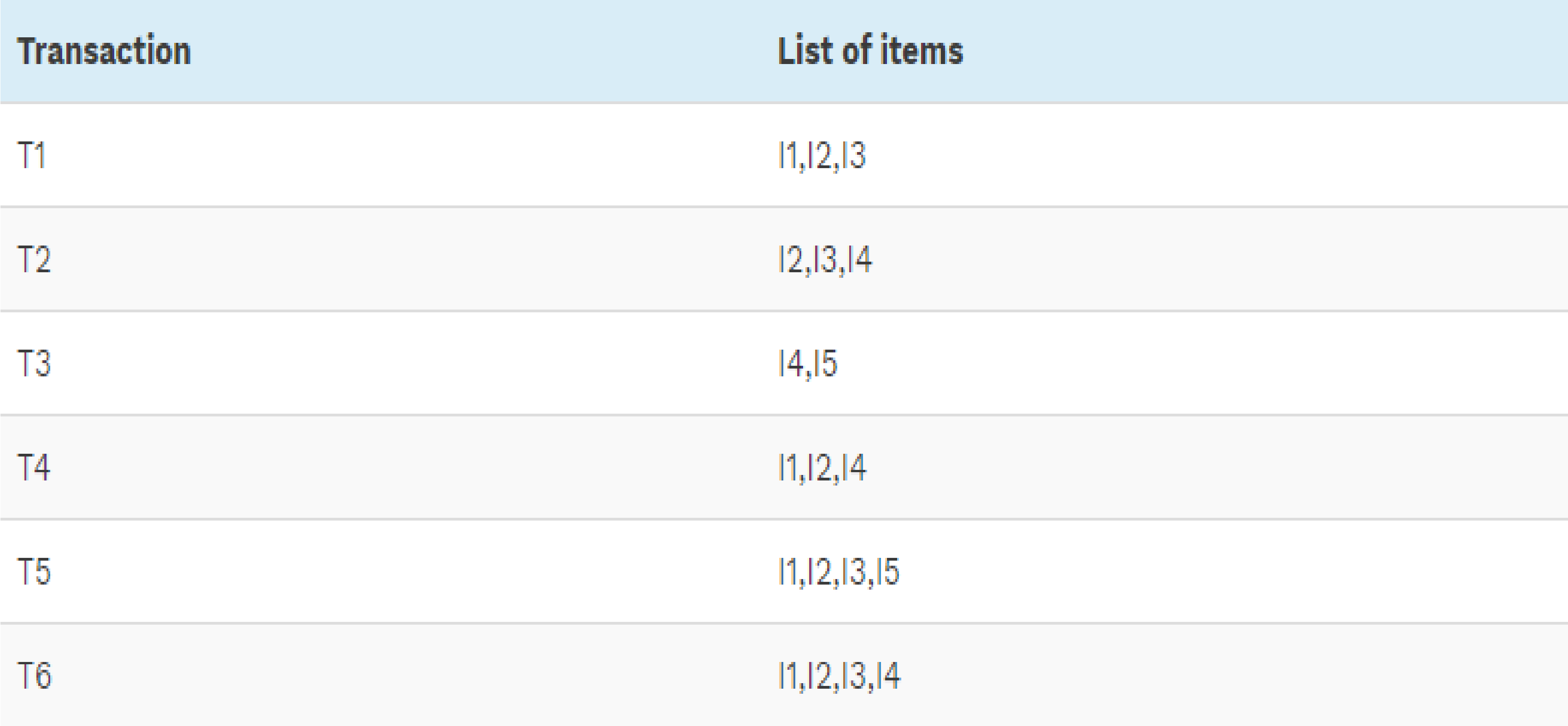
- **#5)** The next iteration will form 3 –itemsets using join and prune step. This iteration will follow antimonotone property where the subsets of 3-itemsets, that is the 2 –itemset subsets of each group fall in min_sup. If all 2-itemset subsets are frequent then the superset will be frequent otherwise it is pruned.

- **#6)** Next step will follow making 4-itemset by joining 3-itemset with itself and pruning if its subset does not meet the min_sup criteria. The algorithm is stopped when the most frequent itemset is achieved.


ItemSets
Generate K-Itemset Candidates (Join Steps)
Check Support (Pruning)
Frequent Items
Set of K-1 frequent itemsets is Ø?
No
Yes
Stop

- **<u>Example of Apriori:</u> Support threshold=50%, Confidence= 60%**

TABLE-1



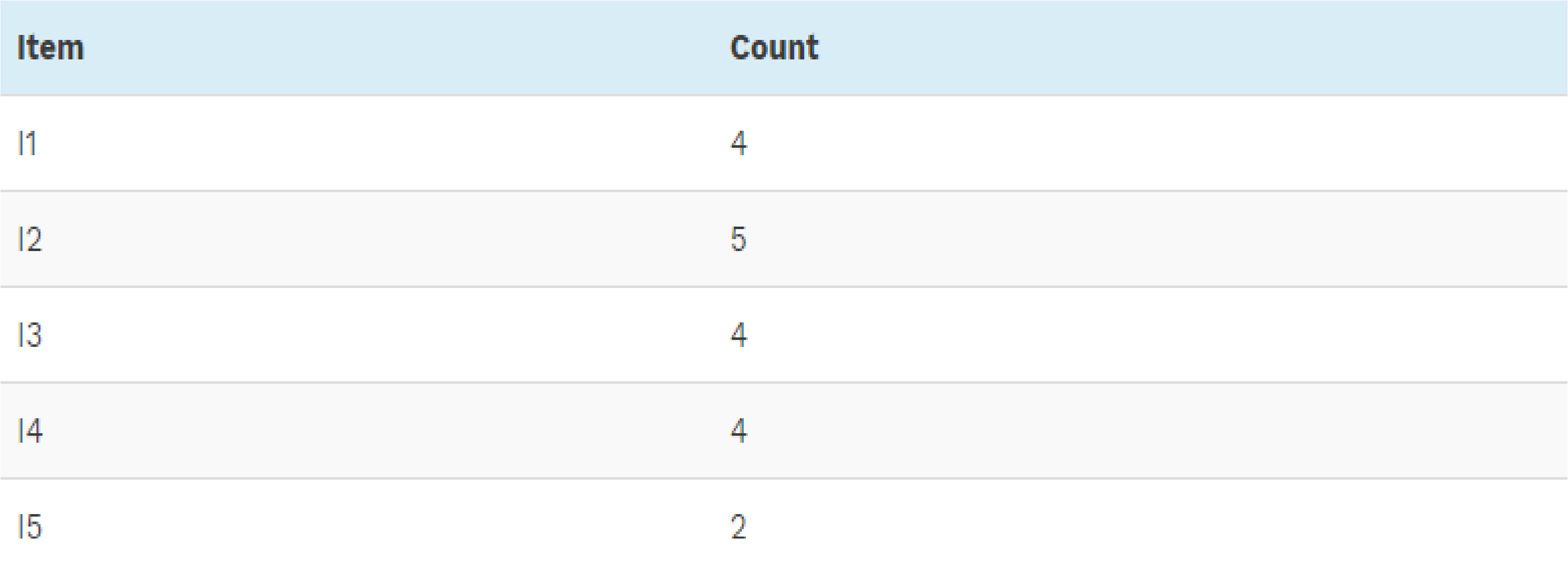
Transaction	List of items
T1	I1,I2,I3
T2	I2,I3,I4
T3	I4,I5
T4	I1,I2,I4
T5	I1,I2,I3,I5
T6	I1,I2,I3,I4

Solution:

Support threshold=50% => 0.5*6= 3 => min_sup=3

1. Count Of Each Item

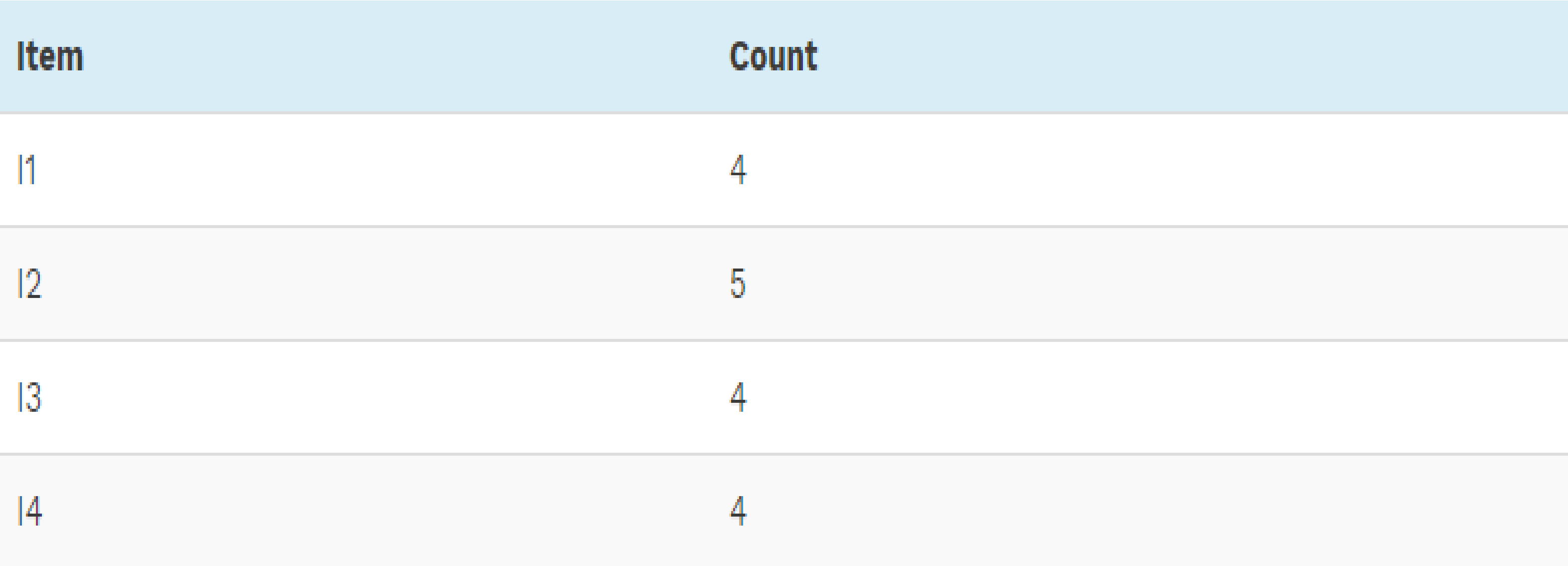
TABLE-2



Item	Count
I1	4
I2	5
I3	4
I4	4
I5	2

2. **Prune Step: TABLE -2** shows that I5 item does not meet min_sup=3, thus it is deleted, only I1, I2, I3, I4 meet min_sup count.

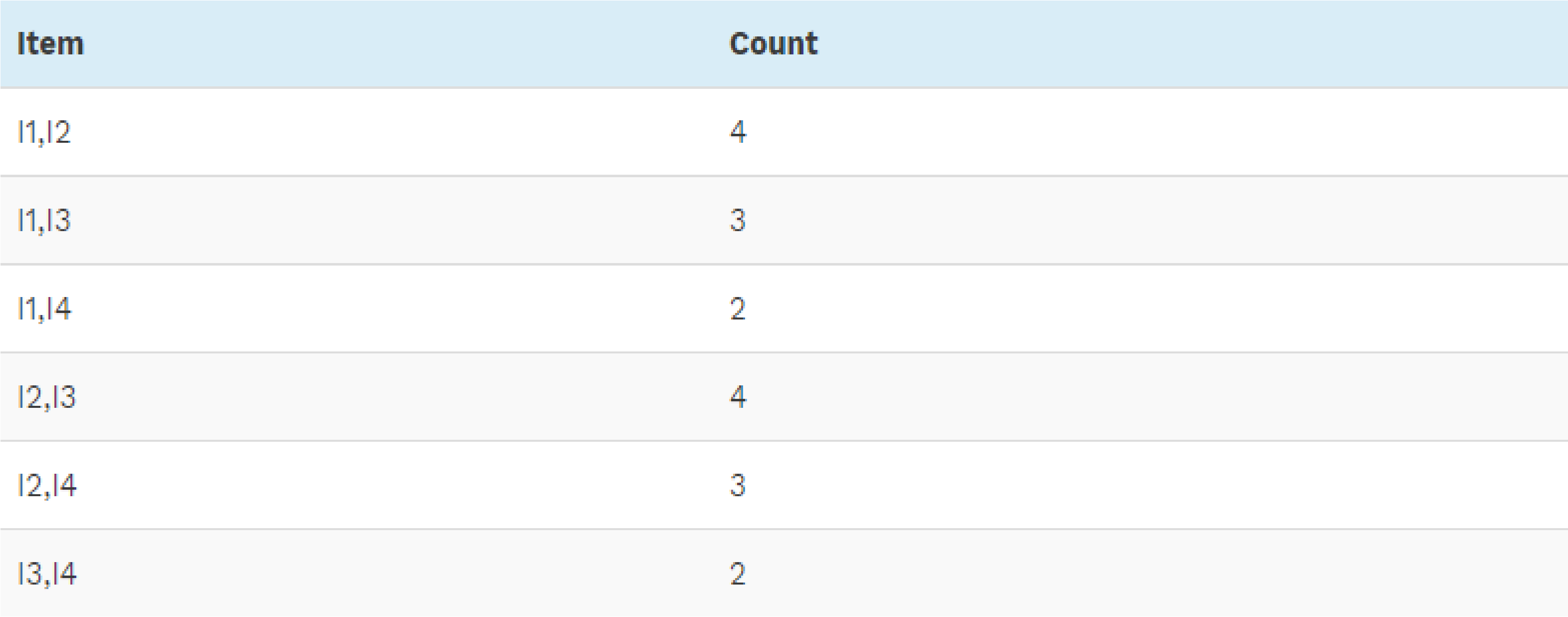
TABLE-3



Item	Count
I1	4
I2	5
I3	4
I4	4

3. **Join Step:** Form 2-itemset. From **TABLE-1** find out the occurrences of 2-itemset.

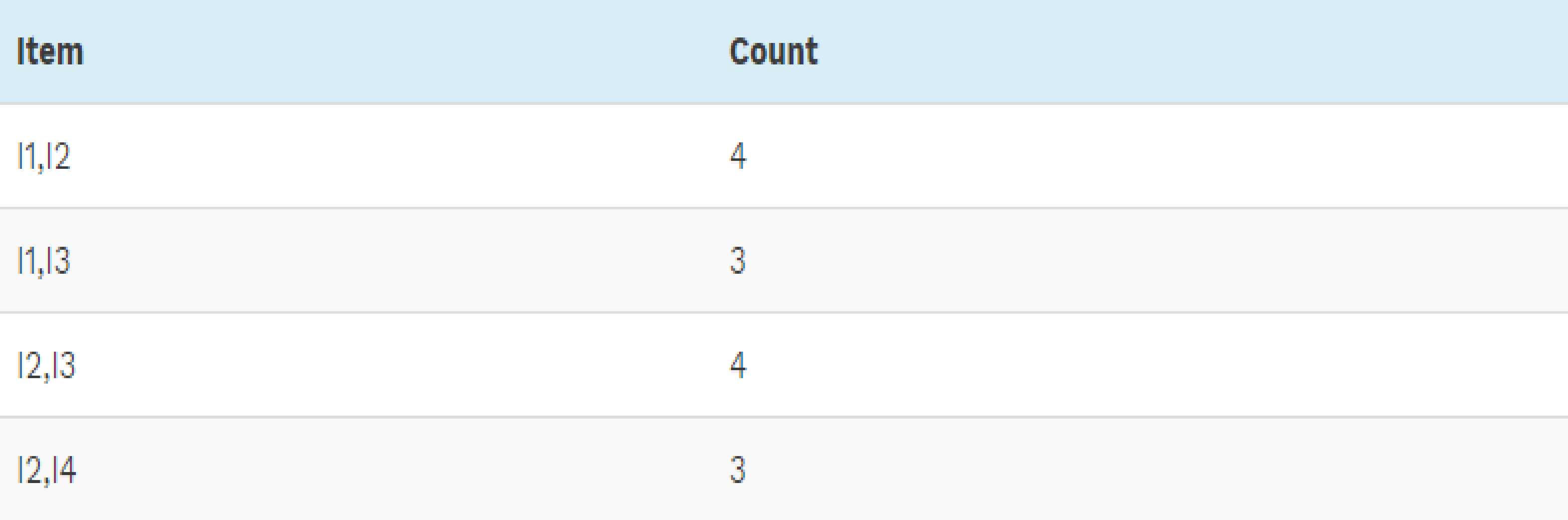
TABLE-4



Item	Count
I1,I2	4
I1,I3	3
I1,I4	2
I2,I3	4
I2,I4	3
I3,I4	2

4. **Prune Step: TABLE -4** shows that item set {I1, I4} and {I3, I4} does not meet min_sup, thus it is deleted.

TABLE-5



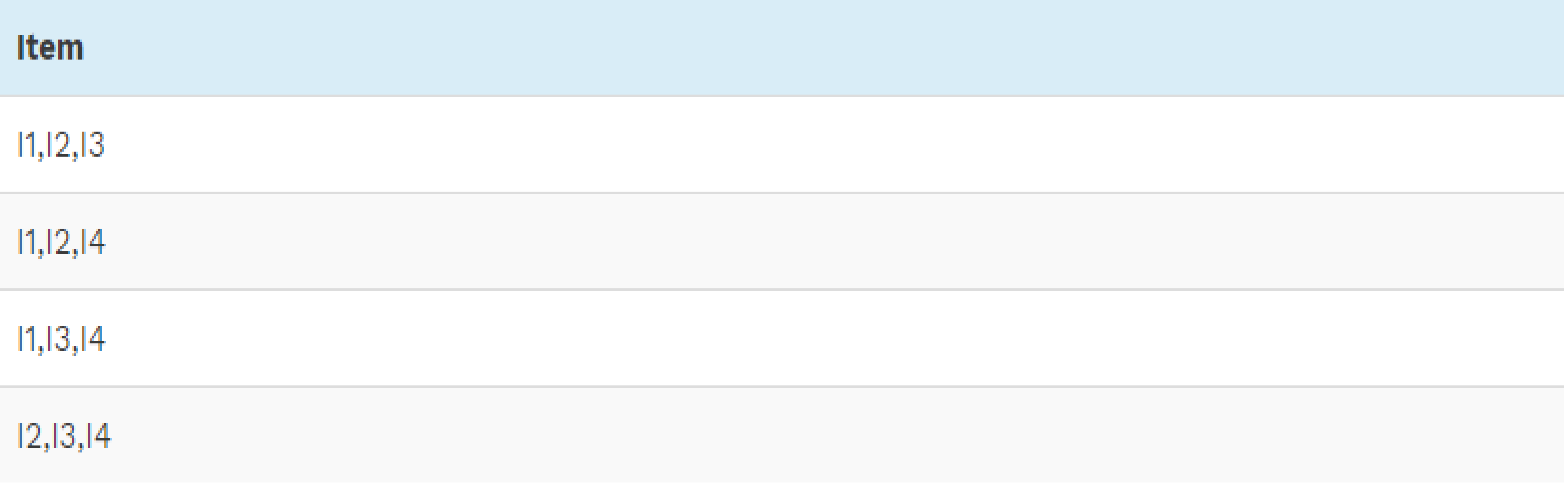
Item	Count
I1,I2	4
I1,I3	3
I2,I3	4
I2,I4	3

- **5. Join and Prune Step:** Form 3-itemset. From the **TABLE- 1** find out occurrences of 3-itemset. From **TABLE-5**, find out the 2-itemset subsets which support min_sup.

  We can see for itemset {I1, I2, I3} subsets, {I1, I2}, {I1, I3}, {I2, I3} are occurring in **TABLE-5** thus {I1, I2, I3} is frequent.

  We can see for itemset {I1, I2, I4} subsets, {I1, I2}, {I1, I4}, {I2, I4}, {I1, I4} is not frequent, as it is not occurring in **TABLE-5** thus {I1, I2, I4} is not frequent, hence it is deleted.

TABLE-6



Item
I1,I2,I3
I1,I2,I4
I1,I3,I4
I2,I3,I4

Only {I1, I2, I3} is frequent.

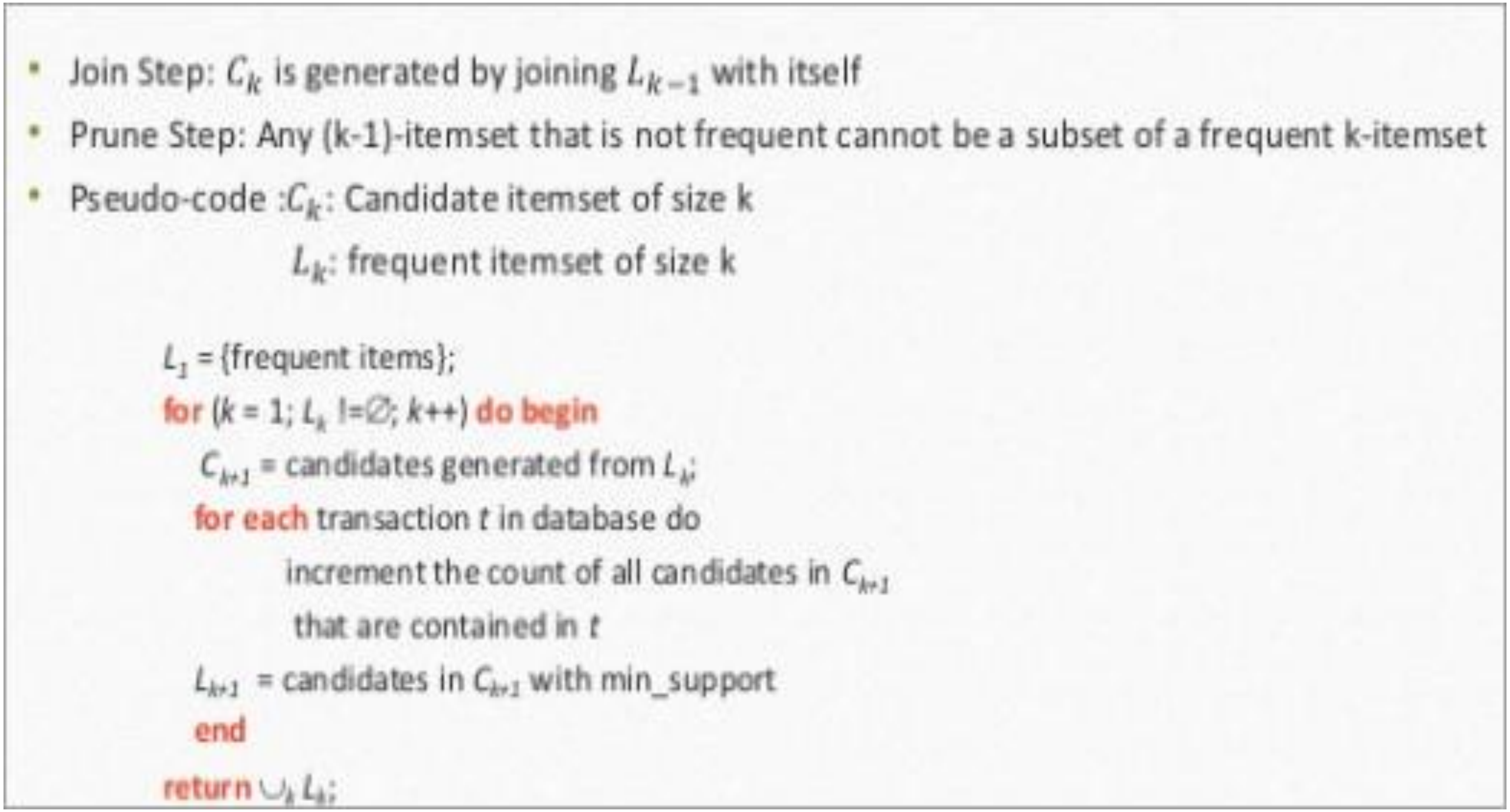
6. Generate Association Rules: From the frequent itemset discovered above the association could be:

- {I1, I2} => {I3}
- Confidence = support {I1, I2, I3} / support {I1, I2} = (3/ 4)* 100 = 75%
- {I1, I3} => {I2}
- Confidence = support {I1, I2, I3} / support {I1, I3} = (3/ 3)* 100 = 100%
- {I2, I3} => {I1}
- Confidence = support {I1, I2, I3} / support {I2, I3} = (3/ 4)* 100 = 75%
- {I1} => {I2, I3}
- Confidence = support {I1, I2, I3} / support {I1} = (3/ 4)* 100 = 75%
- {I2} => {I1, I3}
- Confidence = support {I1, I2, I3} / support {I2 = (3/ 5)* 100 = 60%
- {I3} => {I1, I2}
- Confidence = support {I1, I2, I3} / support {I3} = (3/ 4)* 100 = 75%
- This shows that all the above association rules are strong if minimum confidence threshold is 60%.

- **The Apriori Algorithm: Pseudo Code**

  C: Candidate item set of size k

  L: Frequent itemset of size k



- Join Step: C_k is generated by joining L_{k-1} with itself
- Prune Step: Any (k-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset
- Pseudo-code :C_k: Candidate itemset of size k

  L_k: frequent itemset of size k

```
$L_1$ = {frequent items};
for ($k$ = 1; $L_k$ !=$\varnothing$; $k$++) do begin
    $C_{k+1}$ = candidates generated from $L_k$;
    for each transaction $t$ in database do
            increment the count of all candidates in $C_{k+1}$
            that are contained in $t$
    $L_{k+1}$ = candidates in $C_{k+1}$ with min_support
    end
return $\cup_k L_k$;
```

Advantages

- Easy to understand algorithm
- Join and Prune steps are easy to implement on large itemsets in large databases

Disadvantages

- It requires high computation if the itemsets are very large and the minimum support is kept very low.
- The entire database needs to be scanned.

Methods To Improve Apriori Efficiency

- **Hash-Based Technique:** This method uses a hash-based structure called a hash table for generating the k-itemsets and its corresponding count. It uses a hash function for generating the table.
- **Transaction Reduction:** This method reduces the number of transactions scanning in iterations. The transactions which do not contain frequent items are marked or removed.
- **Partitioning:** This method requires only two database scans to mine the frequent itemsets. It says that for any itemset to be potentially frequent in the database, it should be frequent in at least one of the partitions of the database.
- **Sampling:** This method picks a random sample S from Database D and then searches for frequent itemset in S. It may be possible to lose a global frequent itemset. This can be reduced by lowering the min_sup.
- **Dynamic Itemset Counting:** This technique can add new candidate itemsets at any marked start point of the database during the scanning of the database.

Applications Of Apriori Algorithm

- **In Education Field:** Extracting association rules in data mining of admitted students through characteristics and specialties.

- **In the Medical field:** For example Analysis of the patient's database.

- **In Forestry:** Analysis of probability and intensity of forest fire with the forest fire data.

- Apriori is used by many companies like Amazon in the **Recommender System** and by Google for the auto-complete feature.