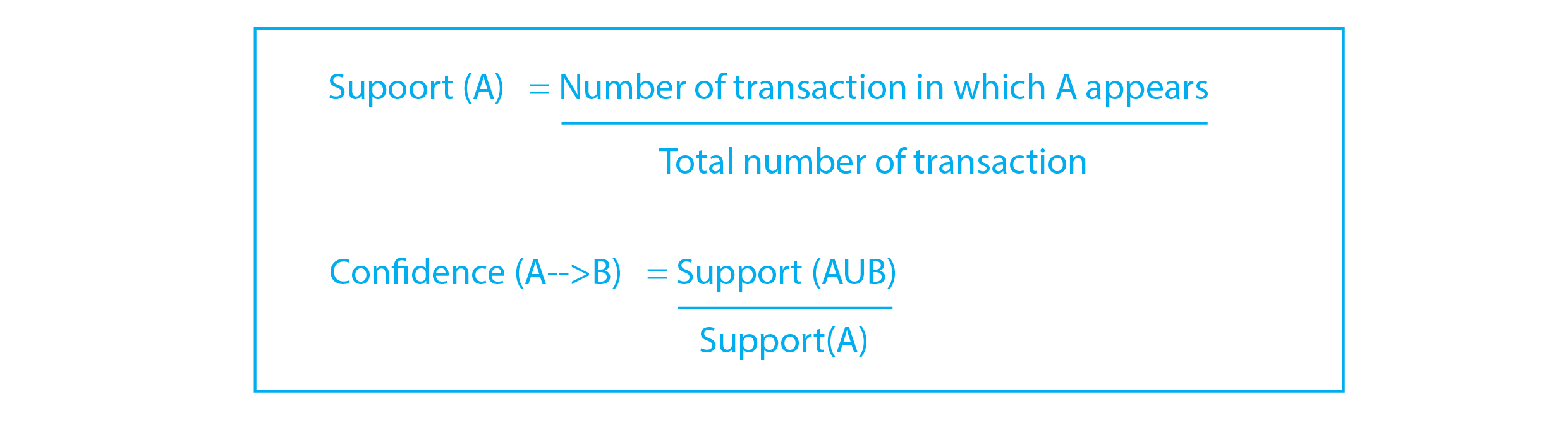
**`Association rules** are a set of rules that indicate the relationship between different items in a dataset. These association rules can be used to discover hidden patterns and relationships between different items in the dataset.

Frequent i[temsets are sets of items](https://botpenguin.com/glossary/data-set) that occur together frequently in a given dataset.

These itemsets can be of any size, and the Apriori Algorithm is used to identify these frequent itemsets. Once these frequent itemsets are identified, association rules can be generated from them.

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

**Support and Confidence**



The first key step while running the Apriori Algorithm is setting the minimum Support and Confidence.

Support measures [the frequency of an item set](https://botpenguin.com/glossary/data-set)in all transactions, while Confidence indicates the likelihood of an item B being purchased when item A is purchased.

**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:**

1. **Join Step**: This step generates (K+1) itemset from K-itemsets by joining each item with itself.
2. **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.

**Example of Apriori: 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%

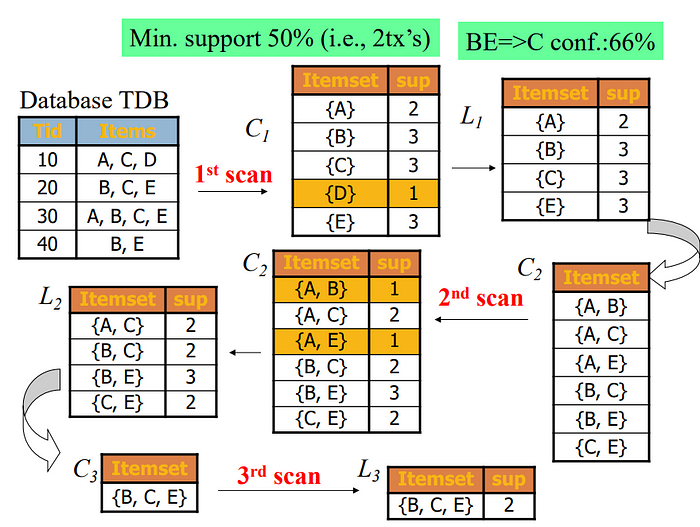
This shows that all the above association rules are strong if minimum confidence threshold is 60%.

**Advantages**

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

**Disadvantages**

1. It requires high computation if the itemsets are very large and the minimum support is kept very low.
2. The entire database needs to be scanned.
3. FPM has many applications in the field of data analysis, software bugs, cross-marketing, sale campaign analysis, market basket analysis, etc.
4. 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.
5. 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.



**Sampling**

It is the practice of selecting an individual group from a population to study the whole population.

Let’s say we want to know the percentage of people who use iPhones in a city, for example. One way to do this is to call up everyone in the city and ask them what type of phone they use. The other way would be to get a smaller subgroup of individuals and ask them the same question, and then use this information as an approximation of the total population.

However, this process is not as simple as it sounds. Whenever you follow this method, your sample size has to be ideal - it should not be too large or too small. Then once you have decided on the size of your sample, you must use the right type of sampling techniques to collect a sample from the population.

**Types Of Sampling Techniques**

1. Simple Random Sampling

In simple random sampling, the researcher selects the participants randomly. There are a number of [data analytics tools](https://www.simplilearn.com/top-data-analysis-tools-article) like random number generators and random number tables used that are based entirely on chance.

Example: The researcher assigns every member in a company database a number from 1 to 1000 (depending on the size of your company) and then use a random number generator to select 100 members.

2. Systematic Sampling

In systematic sampling, every population is given a number as well like in simple random sampling. However, instead of randomly generating numbers, the samples are chosen at regular intervals.

Example: The researcher assigns every member in the company database a number. Instead of randomly generating numbers, a random starting point (say 5) is selected. From that number onwards, the researcher selects every, say, 10th person on the list (5, 15, 25, and so on) until the sample is obtained.

3. Stratified Sampling

In stratified sampling, the population is subdivided into subgroups, called strata, based on some characteristics (age, gender, income, etc.). After forming a subgroup, you can then use random or systematic sampling to select a sample for each subgroup. This method allows you to draw more precise conclusions because it ensures that every subgroup is properly represented.

Example: If a company has 500 male employees and 100 female employees, the researcher wants to ensure that the sample reflects the gender as well. So the population is divided into two subgroups based on gender.

4. Cluster Sampling

In cluster sampling, the population is divided into subgroups, but each subgroup has similar characteristics to the whole sample. Instead of selecting a sample from each subgroup, you randomly select an entire subgroup. This method is helpful when dealing with large and diverse populations.

Example: A company has over a hundred offices in ten cities across the world which has roughly the same number of employees in similar job roles. The researcher randomly selects 2 to 3 offices and uses them as the sample.

**Data Parallelisms and Task Parallelisms**

The key differences between Data Parallelisms and Task Parallelisms are −

| **Data Parallelisms** | **Task Parallelisms** |
| --- | --- |
| 1. Same task are performed on different subsets of same data. | 1. Different task are performed on the same or different data. |
| 2. Synchronous computation is performed. | 2. Asynchronous computation is performed. |
| 3. As there is only one execution thread operating on all sets of data, so the speedup is more. | 3. As each processor will execute a different thread or process on the same or different set of data, so speedup is less. |
| 4. Amount of parallelization is proportional to the input size. | 4. Amount of parallelization is proportional to the number of independent tasks is performed. |
| 5. It is designed for optimum load balance on multiprocessor system. | 5. Here, load balancing depends upon on the e availability of the hardware and scheduling algorithms like static and dynamic scheduling. |

**INCREMENTAL ASSOCIATION RULE MINING**

Applying data mining techniques to real-world applications is a challenging task because the databases are dynamic i.e., changes are continuously taking place due to addition, deletion, modification etc., of the contained data. Generally if the dataset is incremental in nature, the frequent item sets discovering problem consumes more time. Once in a while, the new records are added in an incremental dataset. Generally when compared to the entire data set, the size of the increments or the number of records added to the dataset is very small. But the assumption of the rules in the updated dataset may get distorted due to the addition of these new records. Hence a few new association rules may be created and a few old ones may become obsolete. When new transactions are inserted into the original databases, traditional batch-mining algorithms resolve this problem by reprocessing the entire new databases. But they require much computational time and ignore the available mined knowledge

In recent times, developing approaches for incremental mining of association rules has gained huge importance in real life applications. Recently many researchers have investigated incremental mining algorithms for mining frequent patterns that use information collected during earlier mining process to cut down the cost of finding new patterns in the whole database. Since mining afresh every time the database grows, it becomes inefficient and hence the algorithm for incremental mining has to be investigated. The primary aim is to avoid or minimize scans of the older databases and to avoid re-learning of rules for the old data and utilize the knowledge that has been discovered. Recent important applications have called for the need of incremental mining. This is due to the increasing use of the record-based databases whose data is being continuously added. Examples of such applications include Web log records, stock market data, grocery sales data, transactions in electronic commerce and daily weather/traffic records. In many applications it is likely to mine the transaction database for a fixed amount of most recent data (say, data in the last 12 months). That is, in the incremental mining, one has to not only include new data (i.e., data in the new month) into, but also remove the old data (i.e., data in the most obsolete month) from the mining process. A naive approach to solve the incremental mining problem is to re-run the mining algorithm on the updated database. However, it obviously lacks efficiency since previous results are not utilized for discovering new results while the updated portion is usually small compared to the whole dataset.

Fast update (FUP) proposed by Cheung et al [2] is one approach to association rules that can handle incremental update to reduce the size of the candidate to be searched in original large data bases.. The FUP algorithm uses information from old frequent itemsets to improve its performance.