**APRIORI ALGORITHM**

**algorithm refers to the algorithm which is used to calculate the association rules between objects. It means how two or more objects are related to one another. In other words, we can say that the apriori algorithm is an association rule leaning that analyzes that people who bought product A also bought product B**

**The primary objective of the apriori algorithm is to create the association rule between different objects. The association rule describes how two or more objects are related to one another. Apriori algorithm is also called frequent pattern mining. Generally, you operate the Apriori algorithm on a database that consists of a huge number of transactions.**

**We take an example to understand the concept better. You must have noticed that the Pizza shop seller makes a pizza, soft drink, and breadstick combo together. He also offers a discount to their customers who buy these combos**

**. Do you ever think why does he do so?**

**He thinks that customers who buy pizza also buy soft drinks and breadsticks. However, by making combos, he makes it easy for the customers.at the same time he also incraeses his sales perfomance.**

****Components of Apriori algorithm:****

**The given three components comprise the apriori algorithm.**

**1. Support**

**2.Confidence**

**3.Lift**

**Let's take an example to understand this concept.**

**We have already discussed above; you need a huge database containing a large no of transactions. Suppose you have 4000 customers transactions in a Big Bazar. You have to calculate the Support, Confidence, and Lift for two products, and you may say Biscuits and Chocolate. This is because customers frequently buy these two items together. At the same time, he also increases his sales performance**

****Support:****

**Support refers to the default popularity of any product. You find the support as a quotient of the division of the number of transactions comprising that product by the total number of transactions. Hence, we get**

**Support (Biscuits) = (Transactions relating biscuits) / (Total transactions)**

**= 400/4000 = 10 percent.**

****Confidence:****

**Confidence refers to the possibility that the customers bought both biscuits and chocolates together. So, you need to divide the number of transactions that comprise both biscuits and chocolates by the total number of transactions to get the confidence.**

**Hence,**

**Confidence = (Transactions relating both biscuits and Chocolate) / (Total transactions involving Biscuits)**

**Connfidence=200/400=50**

****Lift:****

**Consider the above example; lift refers to the increase in the ratio of the sale of chocolates when you sell biscuits. The mathematical equations of lift are given below.**

**Lift = (Confidence (Biscuits - chocolates)/ (Support (Biscuits)**

**= 50/10 = 5**

**It means that the probability of people buying both biscuits and chocolates together is five times more than that of purchasing the biscuits alone. If the lift value is below one, it requires that the people are unlikely to buy both the items together. Larger the value, the better is the combination.**

**Steps involved in apriori:**

**#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 . 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:

**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%.

------------------------- END ---------------------------

**The Apriori Algorithm: Pseudo Code**

C: Candidate item set of size k

L: Frequent itemset of size k

[](https://www.softwaretestinghelp.com/wp-content/qa/uploads/2019/09/Psudocode.png)

[image [source](https://www.slideshare.net/" \t "https://www.softwaretestinghelp.com/apriori-algorithm/_blank)]

#### 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.

## Methods To Improve Apriori Efficiency

**Many methods are available for improving the efficiency of the algorithm.**

1. **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.
2. **Transaction Reduction:** This method reduces the number of transactions scanning in iterations. The transactions which do not contain frequent items are marked or removed.
3. **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.
4. **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.
5. **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

**Some fields where Apriori is used:**

1. **In Education Field:** Extracting association rules in data mining of admitted students through characteristics and specialties.
2. **In the Medical field:** For example Analysis of the patient’s database.
3. **In Forestry:** Analysis of probability and intensity of forest fire with the forest fire data.
4. Apriori is used by many companies like Amazon in the **Recommender System** and by Google for the auto-complete feature.

## Conclusion

Apriori algorithm is an efficient algorithm that scans the database only once.

It reduces the size of the itemsets in the database considerably providing a good performance. Thus, data mining helps consumers and industries better in the decision-making process.