# The Apriori Algorithm

08 September 2020 06:18 AM

## Frequent Itemsets, Closed Itemsets, and Association Rules

Let  $I = \{I1, I2, ..., Im\}$  be a set of items.

Let D, the task-relevant data, be a set of database transactions where each transaction T is a set of items such that  $T \subseteq I$ . Each transaction is associated with an identifier, called TID

Data base D:

TID	Items			
100	{i1,i2}			
200	{i2}			
300	{i1,i3}			
400	{i1,i2,i3}			

A set of items is referred to as an **itemset** Ex: {i1,i2}. {i1,i3}

An itemset that contains k items is a k-itemset.

The set {computer, antivirus software} is a 2-itemset.

The occurrence frequency of an itemset is the number of transactions that contain the itemset. This is also known, simply, as the **frequency**, **support count**, **or count of the itemset** 

#### **Association Rule:**

An association rule is an implication of the form  $A \Rightarrow B$ , [Support = %, Confidence= %] where  $A \subset I$ ,  $B \subset I$ , and  $A \cap B = \varphi$ 

 $\begin{aligned} support(A \Rightarrow B) &= P(A \cup B) \\ confidence(A \Rightarrow B) &= P(B \mid A) \\ confidence(A \Rightarrow B) &= P(B \mid A) \\ &= support(A \cup B) / support(A) \\ &= support \ count(A \cup B) / support \ count(A \cup B) \\ &= support \ co$ 

## Strong Association Rule:

Association rules that satisfy both a minimum support threshold (min sup) and a minimum confidence threshold (min conf) are called strong

#### Association rule mining can be viewed as a two-step process:

- Find all frequent itemsets: By definition, each of these itemsets will occur at least as frequently as a predetermined minimum support count, min sup.
- Generate strong association rules from the frequent itemsets:
  By definition, these rules must satisfy minimum support and minimum confidence

## **Apriori Algorithm**

The name of the algorithm is based on the fact that the algorithm uses prior knowledge of frequent itemset properties.

Apriori employs an iterative approach known as a level-wise search, where k-itemsets are used to explore (k+1)-itemset

First, the set of frequent 1-itemsets is found by scanning the database to accumulate the count for each item, and collecting those items that satisfy minimum support. The resulting set is denoted L1. Next, L1 is used to find L2, the set of frequent 2-itemsets, which is used to find L3, and so on, until no more frequent k-itemsets can be found.

## A two-step process is followed, consisting of join and prune actions.

1. The join step: To find  $L_k$ , a set of candidate k-itemsets is generated by joining  $L_{k-1}$  with itself. This set of candidates is denoted  $C_k$ . Let  $l_1$  and  $l_2$  be itemsets in  $L_{k-1}$ .

The notation  $l_i[j]$  refers to the *j*th item in  $l_i$  (e.g.,  $l_1[k-2]$  refers to the second to the last item in  $l_1$ ).

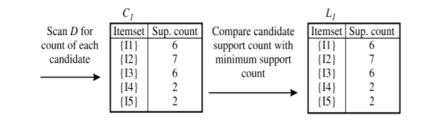
members  $l_1$  and  $l_2$  of  $L_{k-1}$  are joined if  $(l_1[1] = l_2[1]) \land (l_1[2] = l_2[2]) \land \ldots \land (l_1[k-2] = l_2[k-2]) \land (l_1[k-1] < l_2[k-1])$ . The condition  $l_1[k-1] < l_2[k-1]$  simply ensures that no duplicates are generated. The resulting itemset formed by joining  $l_1$  and  $l_2$  is  $l_1[1], l_1[2], \ldots, l_1[k-2], l_1[k-1], l_2[k-1]$ .

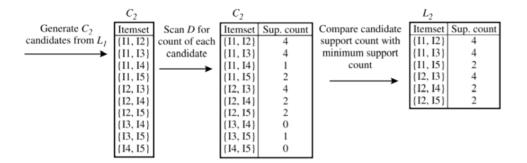
**2.** The prune step:  $C_k$  is a superset of  $L_k$ , that is, its members may or may not be frequent, but all of the frequent *k*-itemsets are included in  $C_k$ .

### Example Problem for Apriori: (V. Imp)

ics branch.			
TID	List of item_IDs		
T100	11, 12, 15		
T200	I2, I4		
T300	I2, I3		
T400	I1, I2, I4		
T500	I1, I3		
T600	I2, I3		
T700	I1, I3		
T800	I1, I2, I3, I5		
T900	11, 12, 13		

Transactional data for an AllElectron-





	$C_3$		$C_3$		Compare candidate	$L_3$	
Generate $C_3$	Itemset	Scan D for	Itemset	Sup. count		Itemset	Sup. count
candidates from	{I1, I2, I3}	count of each	{I1, I2, I3}	2		{I1, I2, I3}	2
$L_2$		candidate			count		
$\longrightarrow$	$\{I1, I2, I5\}$	$\longrightarrow$	{11, 12, 15}	2	$  \longrightarrow$	{I1, I2, I5}	2