Apriori Principle

When making sure that all of the patterns in a set of data meet the minimum support requirements, we want to find all of the patterns that are supported, and not waste time looking at patterns that aren't. This seems simple, but in much larger data sets, it can become difficult to keep track of which patterns are support and which aren't. The Apriori Principle helps with this.

Q. Find the frequent item set and generate few rules associated with the transaction.

(Take the minimum support is 2 or 66.67%(i.e. 66.67/100 x 3 (No. of transaction))

Transaction ID	Items
10	A, B, C, D
20	B, C, D, E
30	A, B, E

1. **Find support count of each itemset:** We start by looking for single items that meet the support threshold. Support count become after counting number of itemset from table.

Item	Support
A	2
B	3
C	2
D	2
E	2
F	1

2. **Compare minimum support = 2;** In this case, it's simply A, B, C, D, and E, because there is at least 2 of each of these in the table but F has only 1. So, remove itemset F. Now, summarized single item support table is given in right side.

Item	Support
A	2
B	3
C	2
D	2
E	2

single item support table

3. Next, we take all of the items that meet the support requirements, everything so far in this example, a make all of the patterns/ combinations pair we can out of them; AB, AC, AD, AE, BC, BD, BE, CD, CE, DE. When we list all of these combinations in a table, and determine the support for each, we get a table that looks like this.

Item	Support
AB	2
AC	1
AD	1
AE	1
BC	2
BD	2
BE	2
CD	2
CE	1
DE	1

2 items support before filtering

4. Several of these patterns don't meet the support threshold of 2, so we remove them from the list of options.

Item	Support
AB	2
BC	2
BD	2
BE	2
CD	2

2 item support table

5. Generate 3 itemset pair and support count w.r.t. step 4

Item	Support
ABC	1
ABD	1
ABE	1
BCD	2
BCE	1
CDE	1

6. **Compare minimum support = 2.** At this point, we use the surviving items to make other patterns that contain 3 items. If you logically work through all of the options you'll get a list like this: ABC, ABD, ABE, BCD, BCE, BDE

Item	Support
BCD	2

3 item support table

7. The final list of all of the patterns that have support greater than or equal to 2 are summarized here.

Item	Support
A	2
B	3
C	2
D	2
E	2
AB	2
BC	2
BD	2
BE	2
CD	2
BCD	2

Now, Generate association rules for itemset B, C, D

Confidence = Support / No. of times occur in transaction

e.g. B ^ C => D i.e. if customer purchase item B and C then they also purchase D item

No of times B, C occur in Transaction ID 10 = 1

No of times B, C occur in Transaction ID 20 = 1

No of times B, C occur in Transaction ID 30 = 0

No of times B, C occur in Overall Transaction = 1 + 1 + 0 = 2

Association Rule	**Support**	No. of time it occurs	**Confidence** = Support / no of time it occurs	**Confidence %**
B ^ C => D	2	B & C = 2 (T10 and T20)	2 / 2 =1	100
B ^ D => C	2	B & D = 2 (T10 and T20)	2/2 = 1	100
C ^ D => B	2	C & D = 2 (T10 and T20)	2/2 = 1	100
D => B ^ C	2	D = 2 (T10 and T20)	2/2 = 1	100
C => B ^ D	2	C = 2 (T10 and T20)	2/2 =1	100
B => C ^ D	2	B = 3 (T10, T20 and T30)	2/3 = 0.67	67

Suppose, given minimum confidence = 80%

Then final association rules are :-

- B ^ C => D
- B ^ D => C
- C ^ D => B
- D => B ^ C
- C => B ^ D

Takeaways:

1. The Apriori Principle can be used to simplify the pattern generation process when mining patterns in data sets
2. If a simple pattern is not supported, then a more complicated one with that simple pattern in it can not be supported (e.g. if AC isn't supported, there is no way that ABC is supported)
3. You can also look at takeaway 2 in the opposite direction. If a complicated pattern meets the minimum support requirements, all of the simpler patterns that can be created from that complicated pattern must be supported. (e.g. if ABC is supported, then AB, BC, AC, A, B, and C all have to be supported)