
Algorithm: Apriori

Input:

D: transaction database;

Min_sup: the minimum support threshold

Output: frequent itemsets

Description:

```
1: L1 = find_frequent_1-itemsets(DB);
2: for(k=2; Lk-1 ≠ ∅; k++) {
3:   Ck = Apriori_gen(Lk-1);
4:   for each transaction t ∈ DB { // scan DB for counts
5:     Ct = subset(Ck, t); // get the subsets of t that are candidates
6:     for each candidate c ∈ Ct
7:       c.count++;
8:   }
9:   Lk = {c ∈ Ck | c.count ≥ min_sup}
10: }
11: return L = ∪k Lk;

procedure Apriori_gen(Lk-1:frequent(k-1)-itemsets)
  1: for each itemset l1 ∈ Lk-1
  2:   for each itemset l2 ∈ Lk
  3:     if (l1[1] = l2[1]) ∧ (l1[2] = l2[2]) ∧ ... ∧ (l1[k-2] = l2[k-2]) ∧ (l1[k-1] < l2[k-1]) then {
  4:       c = l1 ⊗ l2; // join step: generate candidates
  5:       if has_infrequent_subset(c, Lk-1) then
  6:         delete c; // prune step: remove unfruitful candidate
  7:       else add c to Ck;
  8:     }
  9: return Ck;

procedure has_infrequent_subset(c: candidate k-itemset;
Lk-1 : frequent (k-1) -itemsets); // use prior knowledge
  1: for each (k-1) -subset s of c
  2:   if s ∉ Lk-1 then
  3:     return TRUE;
  4: return FALSE;
```

Figure 2-3. The pseudo-code of Apriori algorithm (Agrawal et al 1994)