Introduction to Data Mining

Apriori Principles for Frequent Itemset Mining

Apriori is a classic algorithm used for learning association rules.

Let I be a set of n binary attributes called items. Let D be the set of transactions, called the database. Each transaction in D has a unique ID and contains a subset of the items in I. A rule is defined in the form X->Y where X,Y belong are a subset of I and are mutually exclusive with no common elements. X->Y says that if X exists in a transaction, so does Y.

To select rules, we use constraints on minimum thresholds of support and confidence.

Support(X) is defined as the proportion of transactions that contain the itemset X.

Confidence(X->Y) is the support(X union Y)/support(X).

Lift(X->Y) is the support(X union Y)/(support(X)*support(Y)).

Conviction(X->Y) is (1-support(Y))/(1-confidence(X->Y))

The Apriori Algorithm

Association rule generation is usually split into 2 steps

1. Minimum support is applied to find all frequent itemsets in a database
2. These frequent itemsets and the minimum confidence are used to form rules

The main algorithm is

1. Establish a minimum support level as frequent.
2. Start with one item as the new itemset, and eliminate all infrequent items
3. Out of the new set, take two items as a combination and eliminate all infrequent items
4. Apply the same until no itemset is able to cross the confidence level.

The apriori algorithm uses BFS to count candidate sets efficiently. Then it prunes the candidates with infrequent sub-patterns.

The main inefficiency of this algorithm is that it generates a large number of candidate subsets, causing exponential searching.

FP-Growth

The main idea is to compress a large database into a frequent pattern tree structure. The benefits of doing this are

1. Highly compact and complete
2. Avoid repeated database scans
3. Avoid candidate generation

The construction of the FP tree has 2 main steps

1. Scan the DB for the first time and find frequent patterns and order them in a list L in the frequency descending order with their support
2. For each transaction, order its frequent items according to L, and scan the DB the second time, constructing the FP tree by putting each frequency ordered transaction in it.

The FP tree is a frequent pattern tree.

1. One root is labelled as null, for the set of item prefix subtrees as children and a frequent-item header table
2. Each node in the item prefix subtrees has 3 fields
    a. Item name
    b. Count
    c. Node-link
3. Each entry in the frequent-item header table has 2 fields
    a. Item name
    b. Head of node link

Mining Frequent Patterns using FP Trees

The general idea is to use divide and conquer, where we recursively grow frequent patterns by looking at the short ones and then concatenating the suffix.

For each frequent item, a conditional pattern base is constructed, with its conditional FP tree.

Repeat the process on each newly created conditional FP tree until the resulting tree is empty or contains only one path, which generates all combinations of frequent patterns.

Properties of the FP tree

1. Node link
    a. For any frequent item a, all the possible frequent patterns that contain a can be obtained by following a's links.
2. Prefix Path
    a. To calculate the frequent patterns for a node a in path P, only the prefix sub path of a in P needs to be accumulated, and its frequency count should carry the same count as node a.

Rules from Frequent Itemsets

Given a frequent itemset L, all non empty subsets of L satisfy the minimum confidence requirement are found as f -> L - f. If |L| is k, then there are 2k-2 candidate association rules.

If a rule has low confidence, all its subset rules also have low confidence.