Association Rules Mining

The Apriori Algorithm

The Apriori Algorithm

Market basket analysis is an association rule method that identifies associations in *transactional data*. It is an **unsupervised machine learning** technique used for **knowledge discovery** (rather than prediction). This analysis results in a set of association rules that identify patterns of relationships among items. A rule can typically be expressed in the form

{peanut butter, jelly} → {bread}

The above rule states that if both peanut butter and jelly are purchased, *then* bread is also likely to be purchased.

Transactional data can be extremely large both in terms of the quantity of transactions and the number of items monitored. Given k items that can either appear or not appear in a set, there are 2^k possible item sets that must be searched for rules.

Thus, even if a retailer only has 100 distinct items, he could have $2^{100} = 1.267651 \times 10^{30}$ item sets to evaluate, which is quite an impossible task. However, a smart rule learner algorithm can take advantage of the fact that in reality, many of the potential item combinations are rarely found in practice.

For example, if a retailer sells both firearms and dairy products, a set of {gun, butter} are extremely unlikely to be common. By ignoring these rare cases, it makes it possible to limit the scope of the search for rules to a much more manageable size.

To resolve this issue *R. Agrawal and R. Srikant* introduced the **apriori algorithm**. The apriori algorithm utilizes a simple prior belief (hence the name a priori) about the properties of *frequent* items. Using this a priori belief, all subsets of **frequent items** must also be frequent. This makes it possible to limit the number of rules to search for.

For example, the set {gun, butter} can only be frequent if {gun} and {butter} both occur frequently. Conversely, if neither {gun} nor {butter} are frequent, then any set containing these two items can be excluded from the search.

Measuring Rules of Interest: Support and Confidence

Let $I = \{i_1, i_2, ..., i_d\}$ be the set of all items in a market basket data and $T = \{t_1, t_2, ..., t_N\}$ be the set of all transactions. Each transaction t_i contains a subset of items chosen from I. In association analysis, a collection of zero or more items is termed an **itemset**. If an itemset contains k items, is called a k-itemset.

A transaction t_j is said to contain an itemset X if X is a subset of t_j. An important property of an itemset is its *support count*, which refers to the number of transactions that contain a particular itemset. Mathematically, the support count, $\sigma(X)$, for an itemset X can be stated as follows:

$$\sigma(X) = |\{t_i \colon X \subseteq t_i, \quad t_i \in T\}|$$

where the symbol $|\cdot|$ denotes the number of elements in a set.

There are two statistical measures that can be used to determine whether or not a rule is deemed *interesting*.

- **Support** : This measures how frequently an item set occurs in the data. It can be calculated as

$$\text{Support}(X) = \frac{\text{Count}(X)}{N} = \frac{\sigma(X)}{N}$$

where X represents an item and N represents the total number of transactions.

An itemset X is called *frequent* if the support is greater than some user-defined threshold (sometimes referred to as *minsup*)

- **Confidence** : This measures the algorithm's predictive power or accuracy. It is calculated as the support of item X and Y divided by the support of item X.

$$\text{Confidence}(X \rightarrow Y) = \frac{\text{Support}(X \cup Y)}{\text{Support}(X)}$$

Confidence measures how frequently items in Y appear in transactions that contain X. That is:

$$\text{Confidence}(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

The important thing to note regarding confidence is that

$$\text{Confidence}(X \rightarrow Y) \neq \text{Confidence}(Y \rightarrow X)$$

- **Lift**: the lift of a rule is defined as

$$\text{Lift}(X \rightarrow Y) = \frac{\text{Support}(X \cup Y)}{\text{Support}(X) \cdot \text{Support}(Y)}$$

and can be interpreted as the deviation of the support of the whole rule from the support expected under independence given the supports of both sides of the rule. Greater lift values ($\gg 1$) indicate stronger associations.

To illustrate, consider the following transactional table.

Transaction	Purchases
1	{flowers, get well card, soda}
2	{toy bear, flowers, balloons, candy}
3	{get well card, candy, flowers}
4	{toy bear, balloons, soda}
5	{flowers, get well card, soda}

In this case we have:

$$\text{Confidence}(\text{get well card} \rightarrow \text{flowers}) = \frac{\text{Support}(\text{get well card} \cup \text{flowers})}{\text{Support}(\text{get well card})} = \frac{0.6}{0.6} = 1.0$$

$$\text{Confidence}(\text{flowers} \rightarrow \text{get well card}) = \frac{\text{Support}(\text{flowers} \cup \text{get well card})}{\text{Support}(\text{flowers})} = \frac{0.6}{0.8} = 0.75$$

This means that a purchase of a get well card results in a purchase of flowers 100% of the time, while a purchase of flowers results in a purchase of a get well card 75% of the time. Rules likes

{get well card} → {flowers}

are considered *strong rules* because they have both high support and confidence.

How the Apriori Algorithm Works

The algorithm has two main steps:

1. *Identify all item sets that meet a minimum support threshold*

This process occurs in multiple iterations. Each successive iteration evaluates the support of storing a set of *increasingly large items.*

The first iteration involves evaluating the set of of 1-item sets. The second iteration involves evaluating the set of 2-item sets, and so on.

The result of each iteration k is a set of k-itemsets that meet the minimum threshold. All item sets from iteration k are combined in order to generate candidate item sets for evaluation in iteration $k + 1$.

The apriori principle can eliminate some of the items before the next iteration begins. For example, if {A}, {B}, and {C} are frequent in iteration 1, but {D} is not, then the second iteration will only consider the item sets {A, B}, {A, C}, and {B, C}.

2. *Create rules* from these items that meet a *minimum confidence threshold.*

Example: small transactional dataset

Trace the results of using the Apriori algorithm on the grocery store example below with support threshold $s = 33.34\%$ and confidence threshold $c = 60\%$. Show the candidate and frequent itemsets for each database scan. Enumerate all the final frequent itemsets. Also indicate the association rules that are generated and highlight the strong ones, sort them by confidence.

Transaction ID	Items
T1	Hot-dogs, Buns, Ketchup
T2	Hot-dogs, Buns
T3	Hot-dogs, Coke, Chips
T4	Chips, Coke
T5	Chips, Ketchup
T6	Hot-dogs, Coke, Chips

Given the support threshold is $s = 33.34\%$, this indicates that the threshold is to consider at least 2 transactions

$$s = 0.3334 \leq \text{Support}(X) = \frac{\sigma(X)}{N} = \frac{\sigma(X)}{6}$$

so we need $\sigma(X) \geq 2$.

Pass	Candidate k-itemsets X (with $\sigma(X)$)	Frequent k-itemsets
$k = 1$	{Hot-dogs} (4), {Buns} (2), {Ketchup} (2), {Coke} (3), {Chips} (4)	{Hot-dogs}, {Buns}, {Ketchup}, {Coke}, {Chips}
$k = 2$	{Hot-dogs, Buns}(2),~~{Hot-dogs, Ketchup}~~(1), {Hot-dogs, Coke}(2), {Hot-dogs, Chips}(2), ~~{Buns, Ketchup}~~(1),~~{Buns, Coke}~~(0), ~~{Buns, Chips}~~(0), ~~{Ketchup, Coke}~~(0), ~~{Ketchup, Chips}~~(1),{Coke, Chips}(3)	{Hot-dogs, Buns}, {Hot-dogs, Coke}, {Hot-dogs, Chips}, {Coke, Chips}
$k = 3$	{Hot-dogs, Coke, Chips}(2)	{Hot-dogs, Coke, Chips}
$k = 4$	{}	

Note that {Hot-dogs, Buns, Coke} and {Hot-dogs, Buns, Chips} are not candidates when $k = 3$ because their subsets {Buns, Coke} and {Buns, Chips} are *not frequent*.

Note also that normally, there is no need to go to $k = 4$ since the longest transaction has only 3 items.

All Frequent Itemsets:

- {Hot-dogs}, {Buns}, {Ketchup}, {Coke}, {Chips},
- {Hot-dogs, Buns}, {Hot-dogs, Coke}, {Hot-dogs, Chips}, {Coke, Chips},
- {Hot-dogs, Coke, Chips}

Association Rules: (highlighted rules satisfy the minimum confidence level)

- {Hot-dogs, Buns} would generate:
  - {Hot-dogs} → {Buns} (confidence: 2/4=0.5)
  - **{Buns} → {Hot-dogs} (confidence: 2/2=1)**
- {Hot-dogs, Coke} would generate:
  - {Hot-dogs} → {Coke} (confidence: 0.5)
  - **{Coke} → {Hot-dogs} (confidence: 2/3=0.66)**
- {Hot-dogs, Chips} would generate:
  - {Hot-dogs} → {Chips} (confidence: 0.5)
  - {Chips} → {Hot-dogs} (confidence: 2/4 = 0.5)
- {Coke, Chips} would generate:
  - **{Coke} → {Chips} (confidence: 3/3 = 1)**
  - **{Chips}→ {Coke} (confidence: 3/4 = 0.75)**
- {Hot-dogs, Coke, Chips} would generate:
  - {Hot-dogs} → {Coke, Chips} (confidence: 0.5)
  - **{Coke} → {Hot-dogs, Chips} (confidence: 2/3 = 0.66)**
  - {Chips} → {Hot-dogs, Coke} (confidence: 0.5)
  - **{Hot-dogs, Coke} → {Chips} (confidence: 2/2 = 1)**
  - **{Hot-dogs, Chips} → {Coke} (confidence: 2/2 = 1)**
  - **{Chips, Coke} → {Hot-dogs} (confidence: 2/3 = 0.66)**

With the confidence threshold set to 60%, the strong association rules are (sorted by confidence):

1. {Coke} → {Chips} (confidence: 3/3 = 1)
2. {Buns} → {Hot-dogs} (confidence: 2/2=1)
3. {Hot-dogs, Coke} → {Chips} (confidence: 2/2 = 1)
4. {Hot-dogs, Chips} → {Coke} (confidence: 2/2 = 1)
5. {Chips}→ {Coke} (confidence: 3/4 = 0.75)
6. {Coke} → {Hot-dogs} (confidence: 2/3=0.66)
7. {Coke} → {Hot-dogs, Chips} (confidence: 2/3 = 0.66)
8. {Chips, Coke} → {Hot-dogs} (confidence: 2/3 = 0.66)