STQD6134: Association rule

Scenario: Supermarket Market Basket Analysis

Context:
Imagine you are a data analyst working for a large supermarket chain. The supermarket collects transaction data every day, which contains a list of products purchased by each customer. Your task is to uncover hidden patterns and relationships between products to help the supermarket increase sales and improve product placement.

Business Problem:
The supermarket wants to increase sales by recommending complementary products to customers. For example, if a customer buys bread, the store would like to recommend butter or jam because these items are often bought together. Similarly, if a customer buys milk, they might also be interested in cookies.

Goal:
Using association rule mining, your goal is to identify which items are frequently purchased together, and generate rules that can be used to make product recommendations to customers.

-

What is Association Rule Mining?

Association rule mining is a technique used to uncover hidden patterns or relationships in large datasets, primarily used in **market basket analysis**. It aims to identify associations between different items that frequently co-occur in transactions.

For example, in retail, you might find that customers who buy **bread** often also buy **butter**, which can help stores recommend complementary products.

Key Concepts:

- **Transactions:** A set of items bought together by a customer.
- **Item set:** A collection of items that appear together in a transaction.
- **Association Rule:** A relationship of the form **A → B**, meaning that if item A is bought, item B is likely to be bought as well.

How Association Rules Can Help:

1. Discovering Frequent Itemsets:
You can use association rule mining to find frequent itemsets. For example, you might discover that the following combinations occur frequently:

- Bread and Butter: Customers who buy bread often also buy butter.
- Milk and Cookies: When customers buy milk, they are likely to buy cookies as well.

2. Generating Association Rules:
Once you have the frequent itemsets, you can generate association rules. These rules can help the supermarket recommend products to customers based on what they've already purchased. Here are some examples of possible rules:

- {Bread} → {Butter}: If a customer buys bread, they are likely to buy butter as well. This rule has a high confidence value.
- {Milk} → {Cookies}: If a customer buys milk, they are highly likely to also buy cookies. The supermarket could use this rule to recommend cookies to customers buying milk.

3. Product Recommendations:
With the generated rules, the supermarket can make personalized product recommendations. For instance, while a customer is browsing bread on the store's website or in the physical store, the system can recommend butter or jam, since these items often appear together in transactions.

Similarly, if a customer buys milk, the system could suggest cookies or chocolate based on the rules generated from the transaction data.

Example Rule Interpretation:
- Rule 1: {Bread} → {Butter}
- Support: 0.6 (60% of the transactions contain both bread and butter).
- Confidence: 0.8 (80% of the transactions that contain bread also contain butter).
- Lift: 2.0 (The likelihood of purchasing butter is 2 times higher if bread is bought).

This rule suggests that when a customer buys bread, there's an 80% probability they will also buy butter, and this is twice as likely as if they bought bread and randomly picked other items.

Benefits for the Supermarket:

1. Increased Sales: By recommending complementary products based on association rules, the supermarket can increase the number of items a customer buys per transaction.
2. Improved Product Placement: The supermarket can optimize product placement in stores (e.g., placing butter near the bread aisle) based on the frequent itemsets.
3. Personalized Marketing: The supermarket can send personalized offers or coupons to customers. For instance, if a customer buys bread, they can receive a discount on butter in their next purchase.

Real-World Example:
A popular real-world example of association rule mining is Amazon’s "Customers Who Bought This Item Also Bought" feature. Amazon uses association rule mining on transaction data to recommend products to customers based on their past purchases. For example, if a customer buys a smartphone, Amazon might suggest phone cases or screen protectors, because many other customers who bought a smartphone also bought those items.

Conclusion:
This example highlights how association rule mining can help businesses uncover valuable insights into customer purchasing behavior. By identifying items that are often bought together, businesses can generate rules to guide product recommendations, improve sales strategies, and enhance customer satisfaction.

Important Metrics in Association Rule Mining

Association rule mining involves calculating and using the following metrics to evaluate the quality and strength of the generated rules:

- **Support:**
  - **Definition**: The proportion of transactions in which the itemset appears.
  - **Formula**:

$$\text{Support}(A \rightarrow B) = \frac{\text{Number of transactions containing both A and B}}{\text{Total number of transactions}}$$

  - **Interpretation**: Support indicates the frequency of the itemset in the dataset. High support means the rule is applicable to a large portion of the transactions.

Important Metrics in Association Rule Mining

Confidence:

- **Definition**: The likelihood that item B is bought when item A is bought.
- **Formula**:

$$\text{Confidence}(A \rightarrow B) = \frac{\text{Number of transactions containing both A and B}}{\text{Number of transactions containing A}}$$

- **Interpretation**: Confidence measures the reliability of the rule. A higher confidence value means the rule is more likely to hold.

- **Important Metrics in Association Rule Mining**
  - **Definition**: The ratio of the observed support to the expected support if A and B were independent.
  - **Formula**:

$$\text{Lift}(A \rightarrow B) = \frac{\text{Support}(A \rightarrow B)}{\text{Support}(A) \times \text{Support}(B)}$$

  - **Interpretation**: Lift indicates the strength of the rule. A lift greater than 1 means that items A and B are more likely to occur together than by random chance.

The Apriori Algorithm

The **Apriori algorithm** is one of the most widely used algorithms for mining frequent itemsets and generating association rules, particularly in **market basket analysis**. It was proposed by **R. Agrawal** and **R. Srikant** in 1994. The primary goal of the Apriori algorithm is to discover frequent itemsets in transactional data, which can then be used to generate association rules.

The Apriori Algorithm

Key Concepts in Apriori Algorithm

Before diving into the working of the algorithm, let’s break down the key concepts it relies on:

1. **Frequent Itemsets**: These are sets of items that appear together in a transaction more frequently than a specified threshold (called **support**).
2. **Association Rules**: These are rules of the form **{A}** → **{B}**, where:
    - **A** is an item or itemset (e.g., "Bread"),
    - **B** is another item or itemset (e.g., "Butter").
    - The rule suggests that if **A** is bought, **B** is likely to be bought as well.

The Apriori Algorithm

How Apriori Algorithm Works

The Apriori algorithm works by iteratively finding frequent itemsets and then generating association rules from them. The key idea is to use **"bottom-up"** processing, where smaller itemsets (single items) are expanded to larger itemsets. Here's how it works step-by-step:

1. **Generate Candidate Itemsets**:
   - Start with individual items (single-item itemsets). For example, in a supermarket dataset, items could be "Milk," "Bread," "Butter," etc.
2. **Determine Frequent Itemsets**:
   - Calculate the **support** of each itemset (the proportion of transactions in which the itemset appears).
   - **Frequent itemsets** are those whose support is greater than or equal to the **minimum support threshold**.
3. **Generate Candidate Itemsets of Larger Size**:
   - From the frequent itemsets of size k, generate candidate itemsets of size k+1 by combining itemsets that share common items.
   - For example, if {Milk, Bread} and {Bread, Butter} are frequent itemsets, the algorithm will combine them to create {Milk, Bread, Butter}.
4. **Prune Non-Frequent Itemsets**:
   - After generating larger itemsets, check whether they are frequent by calculating their support.
   - If the itemset is not frequent, prune it and do not generate candidate itemsets of larger size from it.
5. **Repeat Until No More Frequent Itemsets Can Be Found**:
   - Continue expanding itemsets and pruning non-frequent ones until no more frequent itemsets are found.
6. **Generate Association Rules**:
   - Once the frequent itemsets are identified, generate association rules based on the frequent itemsets.
   - **Rule Generation**: From the frequent itemset {A, B}, generate rules like **{A}** → **{B}** and **{B}** → **{A}**.
   - Calculate the **confidence** and **lift** of each rule.
   - Only keep rules that satisfy the **minimum confidence threshold**.

The Apriori Algorithm

Main Steps in Association Rule Mining:

1.Transaction Data Preparation: The dataset is typically transformed into a transaction format

2.Frequent Itemsets Generation: Using the apriori() function in R, we generate frequent itemsets based on support and confidence thresholds.

3.Rule Generation: Rules are generated from frequent itemsets with specified thresholds for support, confidence, and lift.

The arules Package in R

The **arules** package in R provides the tools to perform association rule mining. Here's how it works:

1.Loading and Preparing the Data:

- Data must be in **transaction format**, where each transaction is a set of items.
- The arules package provides functions to convert a dataset into a transaction object (as()) and mine frequent itemsets (apriori()).

2.Mining Frequent Itemsets:

- The **apriori()** function is used to mine frequent itemsets and association rules.
- Parameters like support and confidence are defined to filter out weak itemsets and rules.

3.Inspecting and Sorting Rules:

- After mining, you can inspect the rules using the **inspect()** function.
- You can sort rules based on metrics like lift, confidence, or support to find the most interesting ones.

Association rules in R

Step 1: Load the Required Libraries

```r

# Load the arules package
library(arules)
```

Step 2: Prepare the Transaction Data

Assume you have a dataset like the following (you can use real transaction data or simulate it):

```
# Example dataset with transactions (items purchased per transaction)
transactions <- read.csv("transactions.csv")

# Convert data frame to transactions format
transactions <- as(transactions, "transactions")
```

Step 3: Mine Frequent Itemsets using the Apriori Algorithm

Here, we set a minimum support of 5% and minimum confidence of 50%.

```
# Mine frequent itemsets with the Apriori algorithm
rules <- apriori(transactions, parameter = list(support = 0.05, confidence = 0.5))

# Inspect the first few rules
inspect(head(rules))
```

Step 4: Sort and Filter Rules by Lift

You can sort the rules by the lift metric to identify the strongest associations:

```
# Sort rules by lift
sorted_rules <- sort(rules, by = "lift", decreasing = TRUE)

# Inspect the top 5 rules
inspect(sorted_rules[1:5])
```

If you want to visualize the top rules, you can use the `arulesViz` package:

```r

# Load the arulesViz package for visualization
library(arulesViz)

# Plot the top 5 rules
plot(sorted_rules[1:5], method = "graph")
```

Suppose you mined rules for a supermarket, and the output might look like this:

```
  lhs              rhs           support  confidence lift
1 {Bread}          {Butter}       0.12   0.8        2.5
2 {Milk}           {Bread}        0.15   0.6        1.8
3 {Bread, Butter}  {Jam}          0.05   0.5        3.0
```

- **Rule 1**: If a customer buys **bread**, they are likely to buy **butter** (80% confidence and a lift of 2.5, indicating a strong association).