Association rule mining

Understanding Itemsets

**Introduction**

Welcome to the module on Association Rule Mining.

Have you ever wondered which algorithm powers the "Frequently bought together" list on Amazon/Flipkart, or how you end up buying some items in a general store just because you happen to locate them "at the right place at the right time"? The algorithm that leads to such insights is the Association Rule Mining.

**In this session:**

You will get an introductory idea about the Association Rules. The topics that would be covered are:

* Representing transaction data
* Measure of support
* Measure of confidence
* Generating frequent itemsets
* Apriori principle

# Market Basket Analysis

Association rule mining is an algorithm which is meant to find frequent patterns, correlations or associations from data sets. Given a set of transactions, association rule mining aims to find the rules which enable us to predict the occurrence of a specific item based on the occurrences of the other items in the transaction. Let's learn more about this algorithm.

The main applications of association rule mining:

* **Basket data analysis** - is to analyse the association of purchased items in a single basket or single purchase.
* **Cross marketing**- is to work with other businesses that complement your own, not competitors. For example, vehicle dealerships and manufacturers have cross marketing campaigns with oil and gas companies for obvious reasons.
* **Catalogue design** - the selection of items in a business’ catalogue are often designed to complement each other so that buying one item will lead to buying of another. So these items are often complements or very related.

However, these association rules are only indicative of the existing data pattern and do not imply the causal relationship between the items having the association.

# Support & Confidence

The main purpose of association rule mining is to come up with the association rules, or the rules that signify the relation among the different itemsets. Let's begin our journey towards arriving at such rules from a dataset that has information of all the transactions of a store.

An association rule (X\rightarrow Y ) has two parts, an antecedent (X) and a consequent (Y). An antecedent is any item/itemset found in the data. A consequent is an item/itemset that is found in combination with the antecedent.

Association rules are created by analysing data for frequent if/then patterns and using the support and confidence criteria to identify the most important relationships. Support is an indication of how frequently the items appear in the database. It is measured by the proportion of transactions in which an itemset appears. If you discover that sales of items beyond a certain proportion tend to have a significant impact on your profits, you might consider using that proportion as your support threshold. You may then identify itemsets with support values above this threshold as significant itemsets.

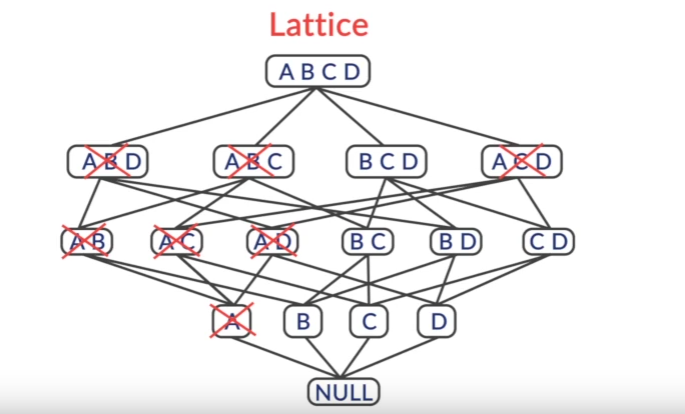
The confidence indicates the number of times the if/then statements have been found to be true. This says how likely item Y is purchased when item X is purchased, expressed as ( X\rightarrow Y ). This is measured by the proportion of transactions with item X, in which item Y also appears. Similar to the support threshold, we also consider a confidence threshold. Only the rules having the confidence above the threshold level are considered important from the business point of view.

# Apriori Principle

So to come up with an association rule, we first need to come up with itemsets that satisfy the minimum threshold level of support. But how do we find such itemsets? Do we need to calculate the support level of each of the possible itemset? Luckily our job is simplified to a large extent by Apriori Principle. Let's learn more about it.

The *apriori principle* can reduce the number of itemsets we need to examine. Put simply, the apriori principle states that if an itemset is infrequent, then all its supersets must also be infrequent. This means that if {X} was found to be infrequent, we can expect {X,Y} to be equally or even more infrequent. So in consolidating the list of popular itemsets, we need not consider {X,Y}, nor any other itemset configuration that contains {X}.

You also learnt that an efficient way to represent the itemsets is the lattice form. Lattice form of representation also helps to visualise the apriori principle in action. For example, just the knowledge that A doesn't have adequate support level, helps you knock out a bunch of other supersets of A out of contention of being in a rule.



**Apriori Principle**

# Generating Frequent Itemsets

So till now, you have figured out that an itemset needs to satisfy the minimum threshold support level to be eligible for featuring in a rule. You have also learnt that the Apriori principle helps to curtail the number of candidate itemsets, and reduces our computational time and effort.

Thus, a generalised algorithm for finding all the itemsets of requisite support level can be written as:

* **Step 0**. Start with itemsets containing just a single item, such as {X} and {Y}.
* **Step 1**. Determine the support for itemsets. Keep the itemsets that meet your minimum support threshold, and remove itemsets that do not.
* **Step 2**. Using the itemsets you have kept from Step 1, generate all the possible itemset configurations.
* **Step 3**. Repeat Steps 1 & 2 until there are no more new itemsets.

Let's learn more about this algorithm. But before you begin the lecture, you can visit [this](https://www.mathsisfun.com/combinatorics/combinations-permutations.html) link to refresh your understanding of permutation and combination.

So, once you figure out the frequent itemsets of size k-1, there are 2 ways to generate the frequent itemsets of size k. These are the F_{k-1} X F_{1} method and the F_{k-1} X F_{k-1} method.

Any such algorithm that we may use, should satisfy 3 basic properties. The list of itemsets should be:

* Complete
* Frugal / Parsimonious
* Non-repeating

# Summary

So, got an introductory idea of the association rule mining in this session. Let's recollect all that was covered:

* Market basket analysis
* Measure of support
* Measure of confidence
* Apriori principle
* Generate and prune strategy

Understanding Association Rules

# Introduction

In the last session on Association rule mining, you got an introductory idea of what market basket analysis is. You also learnt about the measure of support and confidence. Before we can come up with association rules, we first need to generate all the frequent itemsets that have the requisite support level. The apriori principle helps us reduce the computational time and effort.

## In this session:

You will build further on your basic understanding, and come up with the final rules. The topics that would be covered are:

* How to structure and store the generated frequent itemsets for further analysis?
* How to formulate rules from the frequent itemsets?
* How to come up with the most important rules from all possible rules?

# Data Structure for Frequent Itemsets

In the last session, you learnt that the two popular ways to generate the frequent itemsets are the F_{k-1} X F_1 and the F_{k-1} X F_{k-1} method. These methods help us reduce the computational time and effort in coming up with the frequent itemsets.

However, there are also other ways to optimise this process. This is through figuring out an efficient way to store and structure the transaction level data, which makes it easy to traverse through the dataset multiple times in order to come up with the support and confidence measures.

**Note: The given information is not part of the association rule algorithm per se, and is just additional information. You would not be assessed on this information.**

Once, you have the set of all frequent itemsets, and you have also figured out an efficient way to store this information in a data structure for easy retrieval, your next step would be to come up with the association rules. You will learn about forming these rules in the next segment.

# Generating Rules from Itemsets

You have already learnt about the association rules and the measure of confidence for each rule. For a rule, X \rightarrow Y, the confidence C is given by C = \frac{s (X\cup Y)}{s(X)}.

Now, just as the apriori principle helped you in pruning out some itemsets in the frequent itemset generation methodology, similarly, the apriori algorithm also helps in reducing the number of rules that can be created or are viable.

So, the anti-monotonic property of the apriori algorithm states that for a rule X \rightarrow Y, if the confidence is less than the minimum confidence threshold level, then for the rule X{}' \rightarrow Y too, the confidence will be below the minimum threshold level, where X{}' \subseteq X.

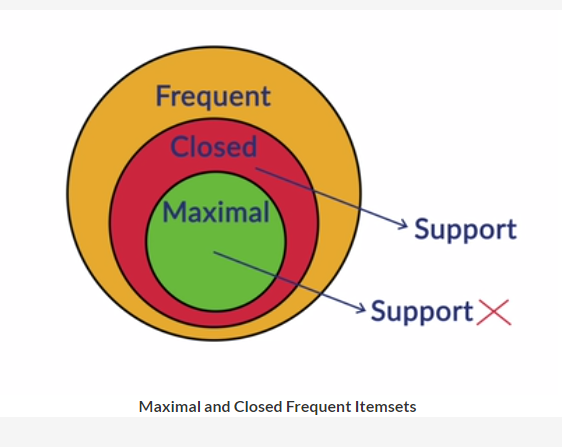
Now before you see a wider application of this anti-monotonic property, let's first see how can you generate rules from a frequent itemset.

Thus, you start with the largest possible antecedent and the smallest consequent and calculate the confidence for the rule. If the confidence is more than the threshold level, you accept the rule and proceed further. Otherwise, no such rule is possible for the same consequent and a subset of the antecedent. This significantly reduces the number of the rules that would be possible and for which you would have to check the confidence level.

# Maximal & Closed Frequent Itemsets

Now you know, how to generate the frequent itemsets and how to derive association rules. However, as you would have noticed, the whole process is still very time-consuming and computationally intensive. So, let's learn some ways in which we could make the information storage and retrieval more efficient, thereby reducing the computational time and effort.

In conclusion, it is important to point out the relationship between frequent itemsets, closed frequent itemsets and maximal frequent itemsets. As mentioned earlier closed and maximal frequent itemsets are subsets of frequent itemsets but maximal frequent itemsets are a more compact representation because it is a subset of closed frequent itemsets.



 Closed frequent itemsets are more widely used than maximal frequent itemset because when efficiency is more important that space, they provide us with the support of the subsets, so no additional pass is needed to find this information.

# Measure of Lift

You have learnt about confidence as a measure of likeliness of an association rule. However, there are many situations, when the confidence measure may not give the true picture of a rule. Let's hear more about such situation and a possible alternative to the measure of confidence.

While, for a rule,  X \rightarrow Y ,  the confidence C is given by C = \frac{s (X\cup Y)}{s(X)} , the lift is given by \frac{s (X\cup Y)}{s(X) * s(Y)}.

The measure of lift is sometimes more relevant compared to the confidence because the measure of confidence does not take into account the support of the consequent. Thus, in cases where the support of the consequent is too high or low compared to the support of the antecedent, the confidence may give a skewed representation of the complete picture.

If the lift is greater than 1, it suggests that the presence of the antecedent has increased the probability that the consequent will occur on this transaction. If the lift is below 1, it suggests that the presence of the antecedent make the probability that the consequent will be part of the transaction lower. If the lift is 1, it suggests that the presence of antecedent and consequent are independent: knowing that the items on the LHS are present makes no difference to the probability that items will occur on the RHS.

When we perform market basket analysis, then, we are looking for rules with a lift of more than one. Rules with higher confidence are ones where the probability of an item appearing on the RHS is high given the presence of the items on the LHS.

# Association Rule Mining in R

You have learnt about the association rule mining in some detail and are well familiar with the different concepts used in the association rule mining using apriori algorithm. Now is the time to implement this algorithm in R on a real dataset and a business problem.

You can download the dataset and the R-script from the link given below, and code along Prof. Raghavan.

Thus, using the "arules" package simplifies your job to a large extent. You simply need to read the data in the transaction form and then explicitly give the support, confidence and the lift level to come up with requisite rules. The "arulesViz" library offers many tools to graphically visualise the generated rules. You can explore the library further for more insights.

# Summary

This brings us to the end of the module on Association Rule Mining. Let's recapture the topics that we covered in this session:

* Data structure to store frequent itemsets
* Generation of rules from frequent itemsets
* Maximal & closed frequent itemsets
* Measure of lift of a rule
* Executing association rule mining in R

To understand Association rules in more detail, go through the live session by Prof. Raghavan

Apriori is an algorithm for frequent item set mining and association rule learning over transactional databases. It proceeds by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database. The apriori algorithm follows the steps below to generate the set of frequent itemsets at each level:

1. Scan the transaction database and note down the support of each item.
2. Compare the support of the items with the minimum support value and discard the items with support less than the minimum support. Let the set of generated frequent itemsets be F_1.
3. Using the elements of F_1, create all possible sets of itemsets of size two (i.e each set having two items). Similar to the previous case, prune the generated candidates with the minimum support criteria to generate F_2 - set of frequent itemsets of size 2.
4. Using the elements of F_2, use either F_{k-1} X F_1 or F_{k-1} X F_{k-1} method to generate candidates for sets of frequent itemsets of size 3. Use the minimum support criteria to discard some of the non-frequent itemsets. Now, use the Apriori principle to check further if the all the subsets of the generated itemset of size 3 are frequent, otherwise, discard the particular itemset with size 3.
5. Continue the process till you have generated the set of frequent itemsets with maximum possible width.
6. Now, you have the list of all frequent itemsets of varying width. Starting from F2 to Fn (n-maximum width of the frequent itemsets), create all possible rules and compare it's confidence with the minimum confidence. Discard the rules which do not match the minimum confidence criteria. Also keep in mind the anti-monotone property for rules while pruning as it will lessen your work.