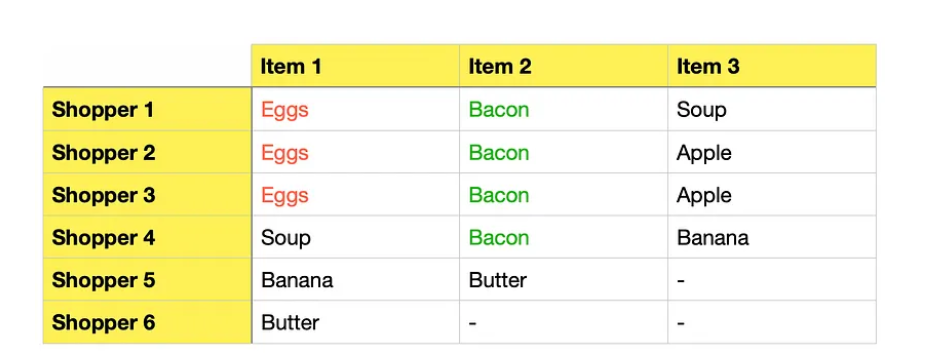
# Apriori Algorithm for Association Rule Learning — How to Find Clear Links Between Transactions

Apriori does not require us to provide a target variable for the model. Instead, the algorithm identifies relationships between data points subject to our specified constraints.

**Note that it is not under sklearn package.**

## **Association Rule Learning**

As briefly mentioned in the introduction, association rule learning is a rule-based machine learning method for discovering interesting relations between variables in large databases. Let’s use a simple supermarket shopping basket analysis to explain how the association rules are found.

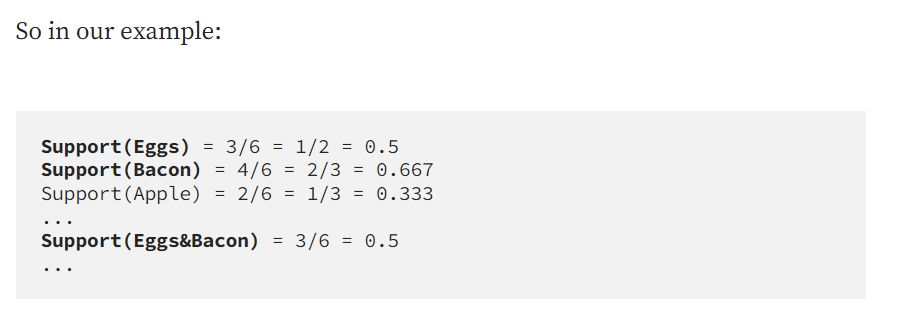


Assume we analyze the above transaction data to find frequently bought items and determine if they are often purchased together. To help us find the answers, we will make use of the following 4 metrics:

* Support
* Confidence
* Lift
* Conviction

## Support

The first step for us and the algorithm is to find frequently bought items. It is a straightforward calculation that is based on frequency:



Here we can set our first constraint by telling the algorithm the minimum support level we want to explore, which is useful when working with large datasets. We typically want to focus computing resources to search for associations between frequently bought items while discounting the infrequent ones.

For the sake of our example, let’s **set minimum support to 0.5**, which leaves us to work with Eggs and Bacon for the rest of this example.

**Important:** while Support(Eggs) and Support(Bacon) individually satisfy our minimum support constraint, it is crucial to understand that we also need the combination of them (Eggs&Bacon) to pass this constraint. Otherwise, we would not have a single item pairing to progress forward to create association rules.

## **Confidence**

Now that we have identified frequently bought items let’s calculate confidence. This will tell us how confident (based on our data) we can be that an item will be purchased, given that another item has been purchased.

**Confidence(A→B)** = Probability(A & B) / Support(A)Note, confidence is the same as what is also known as conditional probability in statistics:  
P(B|A) = P(A & B) / P(A) *Please beware of the notation. The above two equeations are equivalent, although the notations are in different order:* ***(A→B)*** *is the same as* ***(B|A).***

So, let’s calculate confidence for our example:

**Confidence(Eggs→Bacon)** = P(Eggs & Bacon) / Support(Eggs) = (3/6) / (3/6) = 1**Confidence(Bacon→Eggs)** = P(Eggs & Bacon) / Support(Bacon) = (3/6) / (2/3) = 3/4 = 0.75

The above tells us that whenever eggs are bought, bacon is also bought 100% of the time. Also, whenever bacon is bought, eggs are bought 75% of the time.

## **Lift**

**Lift(A→B)** = Probability(A & B) / (Support(A) \* Support(B))You should be able to spot that we can simplify this formula by replacing P(A&B)/Sup(A) with Confidence(A→B). Hence, we have:**Lift(A→B)** = Confidence(A→B) / Support(B)

Let’s calculate lift for our associated items:

**Lift(Eggs→Bacon)** = Confidence(Eggs→Bacon) / Support(Bacon) = 1 / (2/3) = 1.5

**Lift(Bacon→Eggs)** = Confidence(Bacon→Eggs) / Support(Eggs) = (3/4) / (1/2) = 1.5

**Lift for the two items is equal to 1.5. Note, lift>1 means that the two items are more likely to be bought together, while lift<1 means that the two items are more likely to be bought separately. Finally, lift=1 means that there is no association between the two items.**

An intuitive way to understand this would be to first think about the probability of eggs being bought: P(Eggs)=Support(Eggs)=0.5 as 3 out of 6 shoppers bought eggs. SUPPORT.

Then think about the probability of eggs being bought whenever bacon was bought: P(Eggs|Bacon)=Confidence(Bacon->Eggs)=0.75 since out of the 4 shoppers that bought bacon, 3 of them also bought eggs. CONFIDENCE.

Now, LIFT is simply a measure that tells us whether the probability of buying eggs increases or decreases given the purchase of bacon. Since the probability of buying eggs in such a scenario goes up from 0.5 to 0.75, we see a positive lift of 1.5 times (0.75/0.5=1.5). This means you are 1.5 times (i.e., 50%) more likely to buy eggs if you have already put bacon into your basket.

## **Conviction**

Conviction is another way of measuring association, although it is a bit harder to get your head around. It compares the probability that A appears without B if they were independent with the actual frequency of the appearance of A without B. Let’s take a look at the general formula first:

**Conviction(A→B)** = (1 - Support(B)) / (1 - Confidence(A→B))

In our example, this would be:

**Conviction(Eggs→Bacon)** = (1 - Sup(Bacon) / (1 - Conf(Eggs→Bacon)) = (1 - 2/3) / (1 - 1) = (1/3) / 0 = infinity

**Conviction(Bacon→Eggs)** = (1 - Sup(Eggs) / (1 - Conf(Bacon→Eggs)) = (1 - 1/2) / (1 - 3/4) = (1/2) / (1/4) = 2

As you can see, we had a division by 0 when calculating conviction for (Eggs→Bacon) and this is because we do not have a single instance of eggs being bought without bacon (confidence=100%).

In general, high confidence for A→B with low support for item B would yield a high conviction.

In contrast to lift, conviction is a directed measure. Hence, while lift is the same for both (Eggs→Bacon) and (Bacon→Eggs), conviction is different between the two, with Conv(Eggs→Bacon) being much higher. **Thus, you can use conviction to evaluate the directional relationship between your items. What items depends more of the other.**

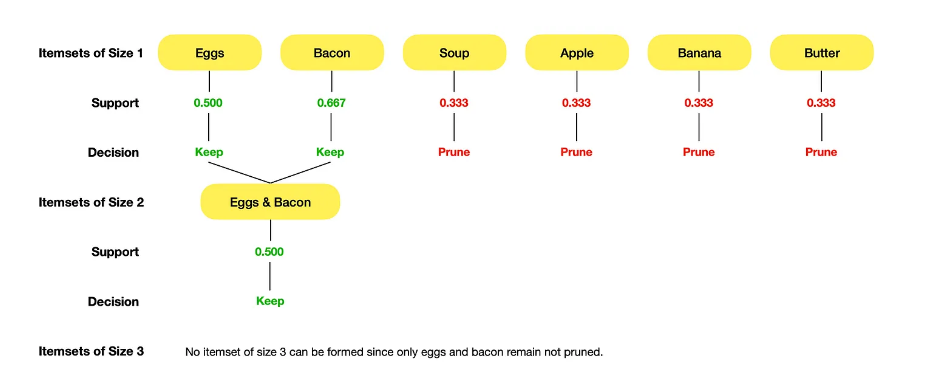
**Finally, similar to lift, conviction=1 means that items are not associated, while conviction>1 indicates the relationship between the items (the higher the value, the stronger the relationship).**

## Apriori algorithm

Apriori is a pretty straightforward algorithm that performs the following sequence of calculations:

1. Calculate support for itemsets of size 1.
2. Apply the minimum support threshold and prune itemsets that do not meet the threshold.
3. Move on to itemsets of size 2 and repeat steps one and two.
4. Continue the same process until no additional itemsets satisfying the minimum threshold can be found.

To make the process more visual, here is a diagram that illustrates what the algorithm does:

****

It is important to realize that by setting a lower minimum support threshold we would produce many more itemsets of size 2. To be precise, with a minimum support threshold of 0.3, none of the itemsets of size 1 would get pruned giving us a total of 15 itemsets of size 2 (5+4+3+2+1=15).

This is not an issue when we have a small dataset, but it could become a bottleneck if you are working with a large dataset. E.g., 1,000 items can create as many as 499,500 item pairs. Hence, choose your minimum support threshold carefully.

Here we can set our first constraint by telling the algorithm the minimum support level we want to explore, which is useful when working with large datasets. We typically want to focus computing resources to search for associations between frequently bought items while discounting the infrequent ones.

This basically gives us three metrics to interpret:

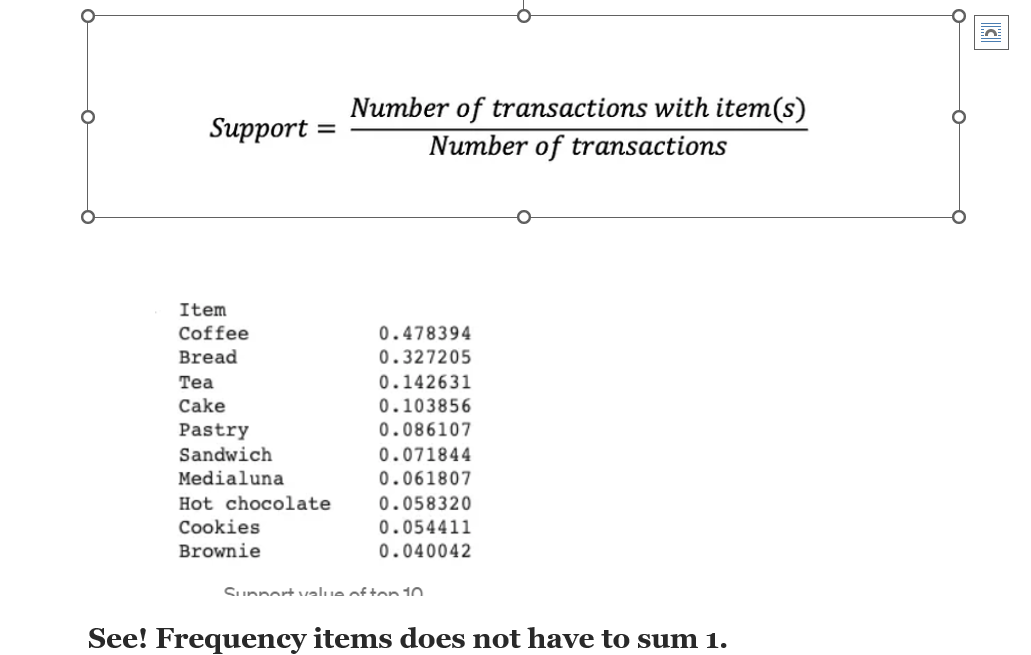
* support (the number of times, or percentage, that the products co-occur)
* confidence (the number of times that a rule occurs, also the conditional probability of the right-hand side given the left-hand side)
* lift (**the strength of association**)
* Those three metrics all have their own validity. It is therefore hard to choose between them. For example, if you have a rule that has a higher lift but lower confidence than another rule, it would be difficult to state that one rule is ‘better than another. At this point, you may just want to keep both rules or try to find a reason to prefer one metric over the other in your specific use case.

## **Preparing the data for efficient-apriori algorithm:**

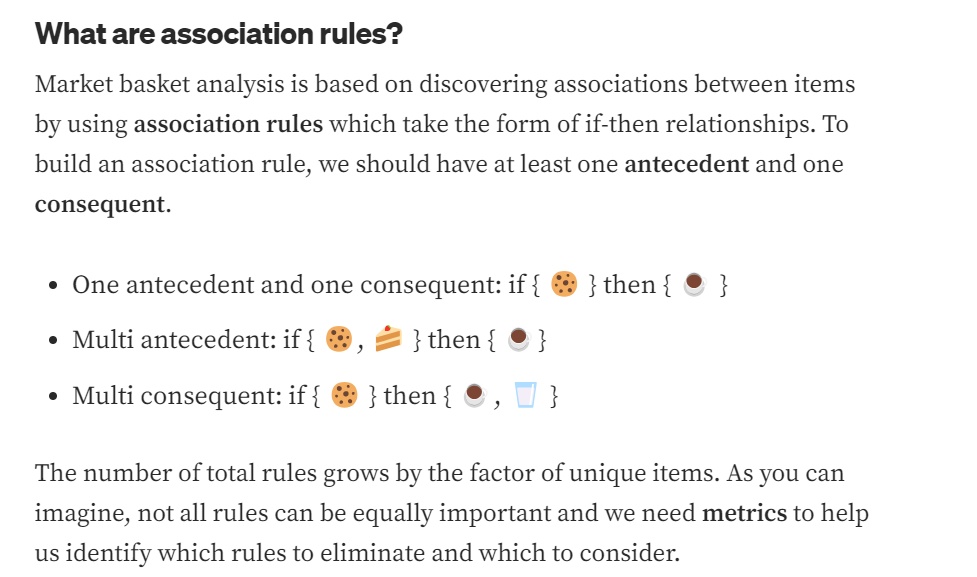
You cannot use data frames in the efficient-apriori algorithm. You need to use a **list of transactions**. In this list, **each transaction is represented as a tuple of the products in that transaction**.

The Apriori library we are going to use requires our dataset to be in the form of a list of lists, where the whole dataset is a big list and each transaction in the dataset is an inner list within the outer big list.

**Baskets must be identified with a unique ID.**

****

Association rule mining is a rule-based machine learning technique used to find frequent patterns in a data set. Frequent patterns may include frequent itemsets that are usually bought together or subsequences that are bought in sequence. For example, cookies and coffee can be frequent itemset for a cafe, and a laptop and external monitor can be a subsequence for an electronics store. Finding frequent patterns in a transactional database and detecting associations between items is an extremely popular data science use case.

****

**Recap:**

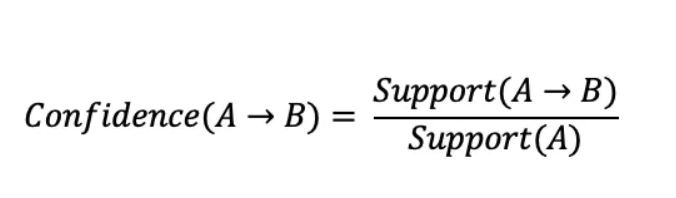
## **Support:**

Support is the main metric to measure how interesting and important a rule is. It can be applied to a single item or pair of antecedents and consequents. It is calculated by dividing the number of transactions including certain item(s) by the number of total transactions. Support value ranges from 0 to 1.

**Confidence:**

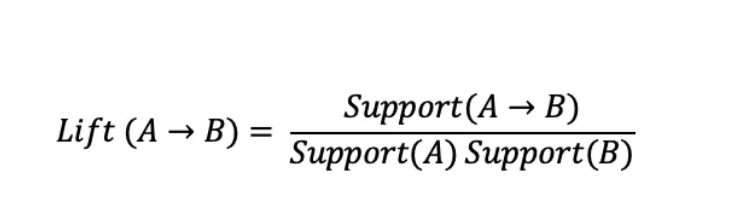
It is important to mention that, confidence metric is not symmetric and Confidence(A→B) is different than Confidence(B→A).

We can interpret these results as 18% confidence means that 18% of the transactions that include Coffee also include Bread. It is important the order.

****

**Lift:**

 It is calculated by dividing the proportion of transactions that contain items A and B by the proportion of item A and item B that takes place independently. The lift value ranges from 0 to infinity.

****

*Lift(A → B) > 1 means that items are****positively****correlated and occurrence of one positively affects the occurrence of other*

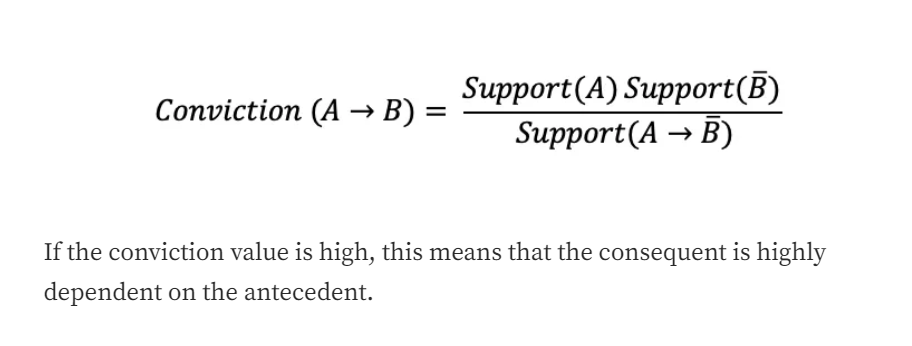
*Lift(A → B) =1 means that there is no correlation*

*Lift(A → B) < 1 means that items are****negatively****correlated and occurrence of one negatively affects the occurrence of other*

**The three more popular metrics are support, confidence and lift.**

## **Conviction:**

Conviction metric is used to measure how much a consequent depends on an antecedent.

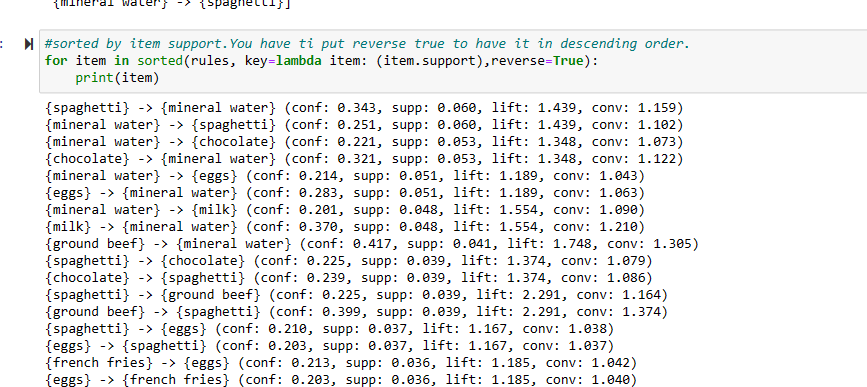
****

**There are two ways of doing association:**

One is: from efficient\_apriori import apriori # for association analysis

You need to convert the dataset to a list of lists.

itemsets, rules = apriori(txns2, min\_support=0.03, min\_confidence=0.2, verbosity=1), does not accept min\_lift.



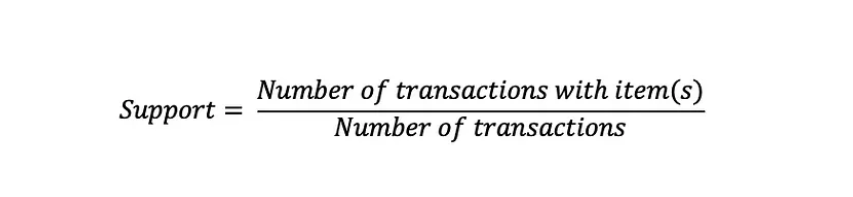
from mlxtend.frequent\_patterns import apriori, association\_rules

apriori module requires a dataframe that has either 0 and 1 or True and False as data. The data we have is all string (name of items), we need to One Hot Encode the data.

you can choose only minimum support threshold at the beginning:

freq\_items = apriori(ohe\_df2, min\_support=0.05, use\_colnames=True, verbose=1)

support goes from 0 to 1 is the number of times an item appears in the whole transcations.



**then for the association:**

# Metric can be set to confidence, lift, support, leverage and conviction.

rules = association\_rules(freq\_items, metric="confidence", min\_threshold=0.6)

rules.head()

**Using multiple metrics one code:**

itemsets,rules=apriori(transactions\_movies,min\_support=0.0053,min\_confidence=0.20)