# Market Basket Analysis using Association Rule-Mining

Unsupervised learning & knowledge discovery in database using Python



Recently, I got an opportunity to work on a project based on Market Basket Analysis, and here I am sharing my experience.

**What is Market basket analysis?**

Market Basket Analysis is one of the fundamental techniques used by large retailers to uncover the association between items. In other words, it allows retailers to identify the relationship between items which are more frequently bought together.

**Association Rules :**



**An example of Association Rules**

* Assume there are 100 customers
* 10 of them bought milk, 8 bought butter and 6 bought both of them.
* bought milk => bought butter
* support = P(Milk & Butter) = 6/100 = 0.06
* confidence = support/P(Butter) = 0.06/0.08 = 0.75
* lift = confidence/P(Milk) = 0.75/0.10 = 7.5

**Usage**:

1. How likely is one to buy bread if (s)he bought milk & eggs?
2. Product placement optimization
3. Product recomendations

Advantages:

1. Fast
2. Works with relatively small amounts of data
3. Few if any feature engineering

Feature engineering:

Association rule is the process of engineering data into a predictive feature that fits the requirements (and / or improves the performance) of a machine learning model.

Apriory:

Apriory is the algorithm implementing association rule mining over structured data.

How Does the Apriori Algorithm Work?

The key concept in the Apriori algorithm is that it assumes all subsets of a frequent itemset to be frequent. Similarly, for any infrequent itemset, all its supersets must also be infrequent.

Let us try and understand the working of an Apriori algorithm with the help of a very famous business scenario, market basket analysis.

Here is a dataset consisting of six transactions in an hour. Each transaction is a combination of 0s and 1s, where 0 represents the absence of an item and 1 represents the presence of it.

We can find multiple rules from this scenario. For example, in a transaction of wine, chips, and bread, if wine and chips are bought, then customers also buy bread.

{wine, chips} => {bread}

In order to select the interesting rules out of multiple possible rules from this small business scenario, we will be using the following measures:

* **Support**
* **Confidence**
* **Lift**
* **Conviction**

Remember I told y’all that we’ll get back to the three most popular criteria evaluating the quality or the strength of an association rule. There are **support, confidence**and**lift**:  
1. Support is the percentage of transactions containing a particular combination of items relative to the total number of transactions in the database. The support for the combination A and B would be,

P(AB) or P(A) for Individual A

2. Confidence measures how much the consequent (item) is dependent on the  
antecedent (item). In other words, confidence is the conditional probability of the consequent given the antecedent,

P(B|A)

where P(B|A) = P(AB)/P(A)

3. Lift (also called improvement or impact) is a measure to overcome the  
problems with support and confidence. Lift is said to measure the difference — measured in ratio — between the confidence of a rule and the expected confidence. Consider an association rule “if A then B.” The lift for the rule is defined as

P(B|A)/P(B) or P(AB)/[P(A)P(B)].

As shown in the formula, lift is symmetric in that the lift for “if A then B” is the same as the lift for “if B then A.”  
4. Each criterion has its advantages and disadvantages but in general we would like association rules that have high confidence, high support, and high lift.

As a summary,

Confidence = P(B|A)

Support = P(AB)

Lift = P(B|A)/P(B)

**Input Dataset :**

<https://github.com/stedy/Machine-Learning-with-R-datasets/blob/master/groceries.csv>

**Let’s look at the code of market basket analysis using Python:**

**Import the Library**

#import packages  
#mlxtend for calculation of support,confidence and lift  
import sys  
import pandas as pd  
from mlxtend.preprocessing import TransactionEncoder  
from mlxtend.frequent\_patterns import apriori

**Read data and Display**

dataframe = pd.read\_csv("groceries.csv", sep='delimiter', header=None, engine='python')  
display(dataframe.head(20))  
print(dataframe.shape)

Preprocessing on Data

* Here we need a data in form of list for Apriori Algorithm.

#converting the dataframe to a list  
data = dataset.values.tolist()  
data#convert the single string in each list to multiple strings separated by commas  
table = []for x in data:  
 new\_list = []  
 for y in x:  
 for z in y.split(','):  
 new\_list.append(z)  
 table.append(new\_list)  
   
table#encode the datasette = TransactionEncoder()  
te\_ary = te.fit(table).transform(table)  
df = pd.DataFrame(te\_ary, columns=te.columns\_)  
df

Using Apriori algorithm:

#generate frequent itemsets using Apriori algorithmfrequent\_itemsets = apriori(df,min\_support=0.2,use\_colnames=True)  
frequent\_itemsets#generating association rules  
from mlxtend.frequent\_patterns import association\_rules  
rules =  
association\_rules(frequent\_itemsets,metric='support',min\_threshold=0.1)ruleses\_specific = rules[['antecedents', 'consequents', 'support']]  
res\_specific.head(20)

Code:

**CODE :**import numpy as np  
import matplotlib.pyplot as plt  
import pandas as pd# Data Preprocessing  
dataset = pd.read\_csv(‘groceries.csv’)transactions = []for i in range(0, 9835):  
 transactions.append([str(dataset.values[i,j]) for j in range(0, 32)])# Training Apriori on the dataset  
from apyori import apriori  
rules = apriori(transactions, min\_support = 0.007, min\_confidence = 0.5, min\_lift = 3, min\_length = 2)# Visualising the results  
results = list(rules)dataset.head()  
dataset.shape  
**OUTPUT**:  
(9835, 32)  
**CODE :**for a in results:  
 print("------------------------------------------------------")  
 print(a)  
**OUTPUT :** RelationRecord(items=frozenset({'citrus fruit', 'other vegetables', 'root vegetables'}), support=0.010371123538383325, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'citrus fruit', 'root vegetables'}), items\_add=frozenset({'other vegetables'}), confidence=0.5862068965517241, lift=3.0296084222733612)])  
-------------------------------------------------------------------------------------------------------------  
RelationRecord(items=frozenset({'tropical fruit', 'other vegetables', 'root vegetables'}), support=0.012302999491611592, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'tropical fruit', 'root vegetables'}), items\_add=frozenset({'other vegetables'}), confidence=0.5845410628019324, lift=3.020999134344196)])  
-------------------------------------------------------------------------------------------------------------  
RelationRecord(items=frozenset({'nan', 'citrus fruit', 'other vegetables', 'root vegetables'}), support=0.010269445856634469, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'nan', 'citrus fruit', 'root vegetables'}), items\_add=frozenset({'other vegetables'}), confidence=0.5838150289017341, lift=3.0172468782178425)])  
-------------------------------------------------------------------------------------------------------------  
RelationRecord(items=frozenset({'tropical fruit', 'nan', 'other vegetables', 'root vegetables'}), support=0.012201321809862735, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'tropical fruit', 'nan', 'root vegetables'}), items\_add=frozenset({'other vegetables'}), confidence=0.5825242718446603, lift=3.010576044977527)])  
-------------------------------------------------------------------------------------------------------------  
RelationRecord(items=frozenset({'tropical fruit', 'whole milk', 'other vegetables', 'root vegetables'}), support=0.007015760040671073, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'tropical fruit', 'whole milk', 'root vegetables'}), items\_add=frozenset({'other vegetables'}), confidence=0.5847457627118644, lift=3.0220570553185424)])

Conclusion

* More algorithms
* More parameter tuning
* More data complexities

This article is a walkthrough for a basic example of implementation of association rule learning for market basket analysis. We focused on theory and application of the most common algorithms.

From the output above, we see that the top associations are not surprising, with one flavor of an item being purchased with another flavor from the same item family . As mentioned, one common application of association rules mining is in the domain of recommending systems. Once item pairs have been identified as having positive relationship, recommendations can be made to customers in order to increase sales. And hopefully, along the way, also introduce customers to items they never would have tried before or even imagined existed!