Data Science – Apriori Algorithm in Python- Market Basket Analysis

*Data Science Apriori algorithm is a data mining technique that is used for mining frequent itemsets and relevant association rules. This module highlights what association rule mining and Apriori algorithm are, and the use of an Apriori algorithm. Also, we will build one Apriori model with the help of Python programming language in a small business scenario.*



 01st Jul, 2019

 344 Views

**Introduction to Apriori Algorithm in Python**

Apriori algorithm is a classical algorithm in data mining that is used for mining frequent itemsets and association rule mining. Before moving ahead, here’s the table of contents of this module:

* + [**What Is Association Rule Mining?**](https://intellipaat.com/blog/data-science-apriori-algorithm/#What_Is_Association_Rule_Mining)
  + [**What Is an Apriori Algorithm?**](https://intellipaat.com/blog/data-science-apriori-algorithm/#What_Is_an_Apriori_Algorithm)
  + [**How Does the Apriori Algorithm Work?**](https://intellipaat.com/blog/data-science-apriori-algorithm/#How_Does_the_Apriori_Algorithm_Work)
  + [**Support**](https://intellipaat.com/blog/data-science-apriori-algorithm/#Support)
  + [**Confidence**](https://intellipaat.com/blog/data-science-apriori-algorithm/#Confidence)
  + [**Lift**](https://intellipaat.com/blog/data-science-apriori-algorithm/#Lift)
  + [**Conviction**](https://intellipaat.com/blog/data-science-apriori-algorithm/#Conviction)
  + [**Hands-on: Apriori Algorithm in Python- Market Basket Analysis**](https://intellipaat.com/blog/data-science-apriori-algorithm/#Apriori_Algorithm_in_Python)
  + [**Problem Statement**](https://intellipaat.com/blog/data-science-apriori-algorithm/#Problem)
  + [**Dataset**](https://intellipaat.com/blog/data-science-apriori-algorithm/#Dataset)
  + [**Environment Setup**](https://intellipaat.com/blog/data-science-apriori-algorithm/#Environment)
  + [**Implementation**](https://intellipaat.com/blog/data-science-apriori-algorithm/#Implementation)

**Here’s a video from Intellipaat on this topic:**

Without much delay, let’s get started.

**What Is Association Rule Mining?**

As mentioned before, the Apriori algorithm is used for the purpose of association rule mining. Now, what is association rule mining? Association rule mining is a technique to identify frequent patterns and associations among a set of items.

For example, understanding customer buying habits. By finding correlations and associations between different items that customers place in their ‘shopping basket,’ recurring patterns can be derived.

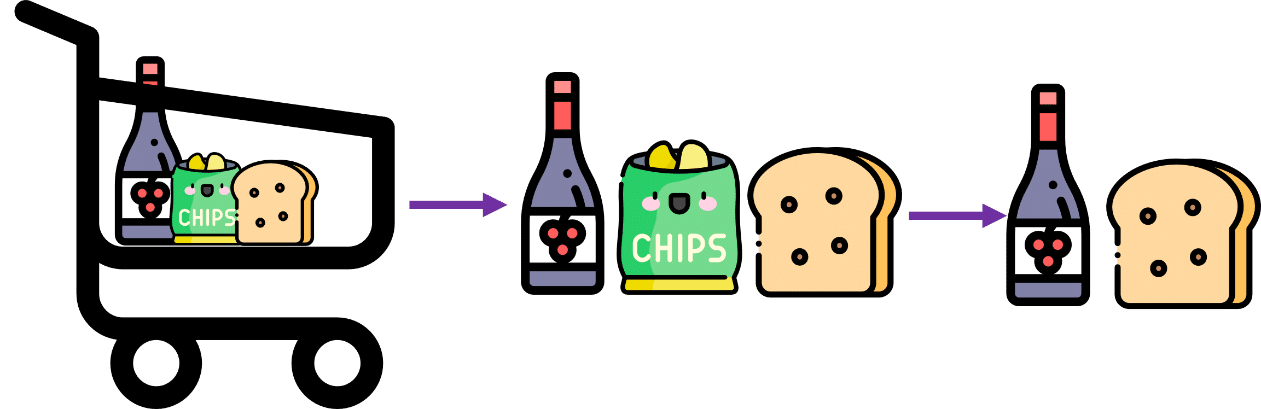


Say, Joshua goes to buy a bottle of wine from the supermarket. He also grabs a couple of chips as well. The manager there analyses that, not only Joshua, people often tend to buy wine and chips together. After finding out the pattern, the manager starts to arrange these items together and notices an increase in sales.

This process of identifying an association between products/items is called association rule mining. To implement association rule mining, many algorithms have been developed. Apriori algorithm is one of the most popular and arguably the most efficient algorithms among them. Let us discuss what an Apriori algorithm is.

**What Is an Apriori Algorithm?**

Apriori algorithm assumes that any subset of a frequent itemset must be frequent.



Say, a transaction containing {wine, chips, bread} also contains {wine, bread}. So, according to the principle of Apriori, if {wine, chips, bread} is frequent, then {wine, bread} must also be frequent.

**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.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Transaction ID** | **Wine** | **Chips** | **Bread** | **Milk** |
| 1 | 1 | 1 | 1 | 1 |
| 2 | 1 | 0 | 1 | 1 |
| 3 | 0 | 0 | 1 | 1 |
| 4 | 0 | 1 | 0 | 0 |
| 5 | 1 | 1 | 1 | 1 |
| 6 | 1 | 1 | 0 | 1 |

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**
  + **List**
  + **Conviction**

**Support**

Support of the item *x* is nothing but the ratio of the number of transactions in which the item *x* appears to the total number of transactions.

i.e.,

Support(wine) =

C:\Users\Laptop\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclip\clip_image004.png

Support(wine) = 4/6 = 0.66667

**Confidence**

Confidence (*x* => *y*) signifies the likelihood of the item *y* being purchased when the item *x* is purchased. This method takes into account the popularity of the item *x*.

i.e.,

Conf({wine, chips} => {bread}) =

C:\Users\Laptop\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclip\clip_image005.png

Conf({wine, chips} => {bread})= (2/6)/(3/6) = 0.6667

**Lift**

Lift (*x* => *y*) is nothing but the ‘interestingness’ or the likelihood of the item *y* being purchased when the item *x* is sold. Unlike confidence (*x* => *y*), this method takes into account the popularity of the item *y*.

i.e.,

lift ({wine, chips} => {bread}) =

C:\Users\Laptop\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclip\clip_image005.png

lift ({wine, chips} => {bread}) = (2/6) /(3/6 + 4/6) = 0.2857 (???)

C:\Users\Laptop\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclip\clip_image006.png

=1

* + **Lift (*x*=> *y*)** = 1 means that there is no correlation within the itemset.
  + **Lift (*x* => *y*)** > 1 means that there is a positive correlation within the itemset, i.e., products in the itemset, *x* and *y*, are more likely to be bought together.
  + **Lift (*x* => *y*)** < 1 means that there is a negative correlation within the itemset, i.e., products in itemset, *x* and *y*, are unlikely to be bought together.

**Check this Intellipaat Data Science Full Course video :**

**Conviction**

Conviction of a rule can be defined as follows:

conv(*x* => *y*) =

C:\Users\Laptop\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclip\clip_image007.png

i.e.,

conv({wine, chips} => {bread} ) =

C:\Users\Laptop\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclip\clip_image008.png

= (1- 4/6) / (1 - 2/3) = 1

Its value range is [0, +∞].

* + **Conv(*x* => *y*) = 1** means that *x* has no relation with *y*.
  + Greater the conviction higher the interest in the rule.

Now that we know the methods to find out the interesting rules, let us go back to the example. Before we get started, let us fix the support threshold to 50 percent.

**Step 1: Create a frequency table of all the items that occur in all transactions**

|  |  |
| --- | --- |
| **Item** | **Frequency** |
| Wine | 4 |
| Chips | 4 |
| Bread | 4 |
| Milk | 5 |

**Step 2**: **Find the significant items based on the support threshold**

Support threshold = 3

|  |  |
| --- | --- |
| **Item** | **Frequency** |
| Wine | 4 |
| Chips | 4 |
| Bread | 4 |
| Milk | 5 |

**Step 3**:**From the significant items, make possible pairs irrespective of the order**

|  |  |
| --- | --- |
| **Item** | **Frequency** |
| Wine, Chips | 3 |
| Wine, Bread | 3 |
| Wine, Milk | 4 |
| Chips, Bread | 2 |
| Chips, Milk | 3 |
| Bread, Milk | 4 |

**Step 4**: **Again, find the significant items based on the support threshold**

|  |  |
| --- | --- |
| **Item** | **Frequency** |
| Wine, Milk | 4 |
| Bread, Milk | 4 |

**Step 5: Now, make a set of three items that are bought together based on the significant items from Step 4**

|  |  |
| --- | --- |
| **Item** | **Frequency** |
| Wine, Bread, Milk | 3 |

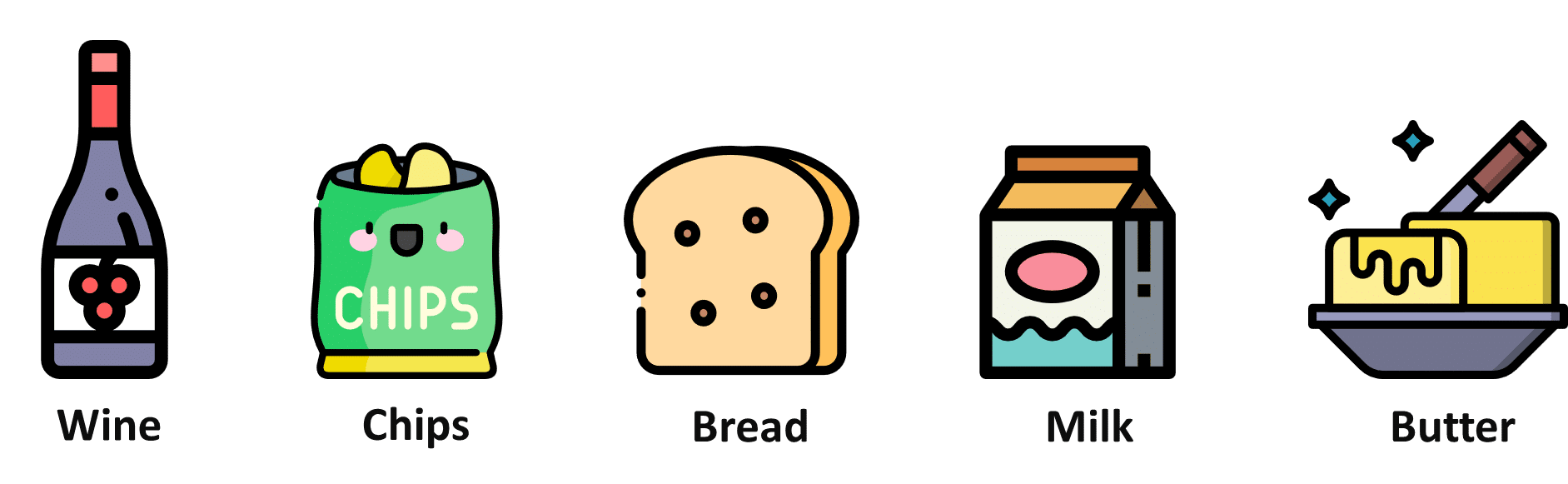
**{Wine, Bread, Milk}** is the only significant itemset we have got from the given data. But in real-world scenarios, we would have dozens of items to build rules from. Then, we might have to make four/five-pair itemsets.

Now here is an Apriori algorithm example to explain how Apriori algorithm works, let us implement this with the help of Python programming language.

**Hands-on: Apriori Algorithm in Python- Market Basket Analysis**

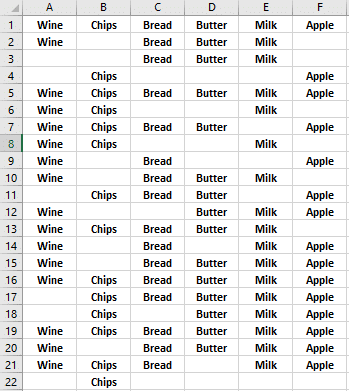
**Problem Statement**

The manager of a retail store is trying to find out an association rule between six items, to figure out which items are more often bought together so that he can keep the items together in order to increase sales.



**Dataset**

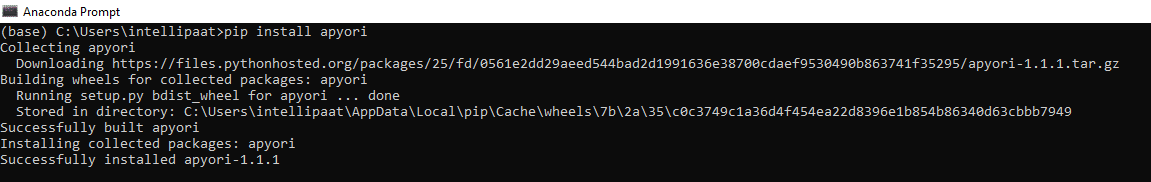
Below is the transaction data from Day 1. This dataset contains 6 items and 22 transaction records.



**Environment Setup:**

Before we move forward, we need to install the ‘apyori’ package first.

pip install apyori

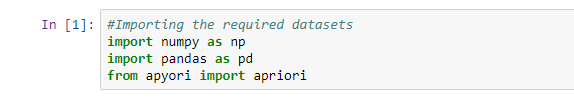


**Market Basket Analysis Implementation with in Python**

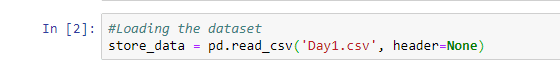
With the help of **apyori** package, we will be implementing the Apriori algorithm in order to help the manager in market basket analysis.



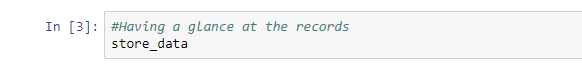
**Step 1: Import the libraries**

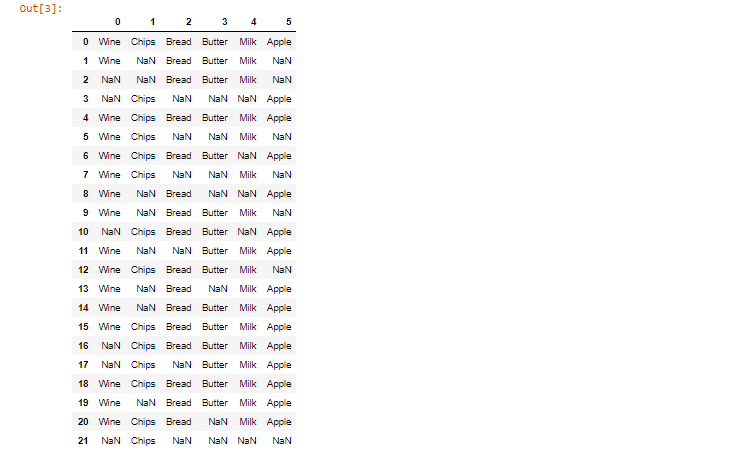


**Step 2: Load the dataset**



**Step 3: Have a glance at the records**



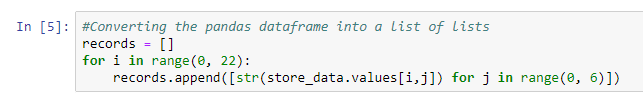


**Step 4: Look at the shape**

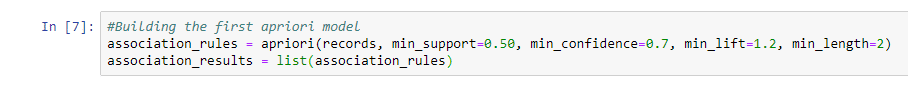
C:\Users\Laptop\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclip\clip_image017.png

C:\Users\Laptop\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclip\clip_image018.png

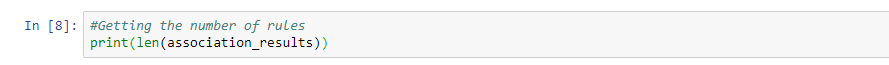
**Step 5:** **Convert Pandas DataFrame into a list of lists**



**Step 6: Build the Apriori model**



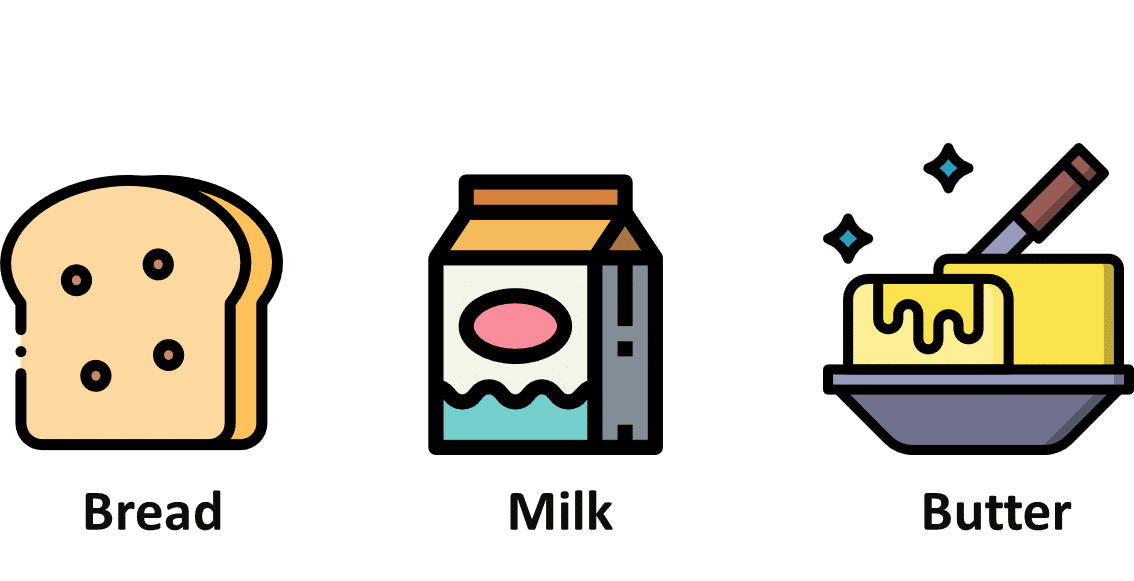
**Step 7: Print out the number of rules**



**Step 8: Have a glance at the rule**



C:\Users\Laptop\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclip\clip_image023.png



The support value for the first rule is 0.5. This number is calculated by dividing the number of transactions containing ‘Milk,’ ‘Bread,’ and ‘Butter’ by the total number of transactions.

The confidence level for the rule is 0.846, which shows that out of all the transactions that contain both “Milk” and “Bread”, 84.6 percent contain ‘Butter’ too.

The lift of 1.241 tells us that ‘Butter’ is 1.241 times more likely to be bought by the customers who buy both ‘Milk’ and ‘Butter’ compared to the default likelihood sale of ‘Butter.’

**What Did We Learn?**

In this tutorial, we have learned what association rule mining is, what the Apriori algorithm is, and with the help of an Apriori algorithm example we learnt how Apriori algorithm works. At the end, we have built an Apriori model in Python programming language on market basket analysis.

**Related Articles**

* + [What is Data Science?](https://intellipaat.com/blog/what-is-data-science/)
  + [Data Science vs Data Analytics vs Big Data](https://intellipaat.com/blog/compare-big-data-data-science-data-analytics/)
  + [Connection Between Data Science and Cloud Computing!](https://intellipaat.com/blog/connection-between-data-science-and-cloud-computing/)

[PREVIOUSWHAT IS AMAZON AWS MANAGEMENT CONSOLE?](https://intellipaat.com/blog/what-is-amazon-aws-management-console/)

[NEXTLINEAR REGRESSION IN PYTHON – SIMPLE AND MULTIPLE LINEAR REGRESSION](https://intellipaat.com/blog/what-is-linear-regression/)

[Data Science](https://intellipaat.com/blog/category/data-science/)

*From <*[*https://intellipaat.com/blog/data-science-apriori-algorithm/*](https://intellipaat.com/blog/data-science-apriori-algorithm/)*>*