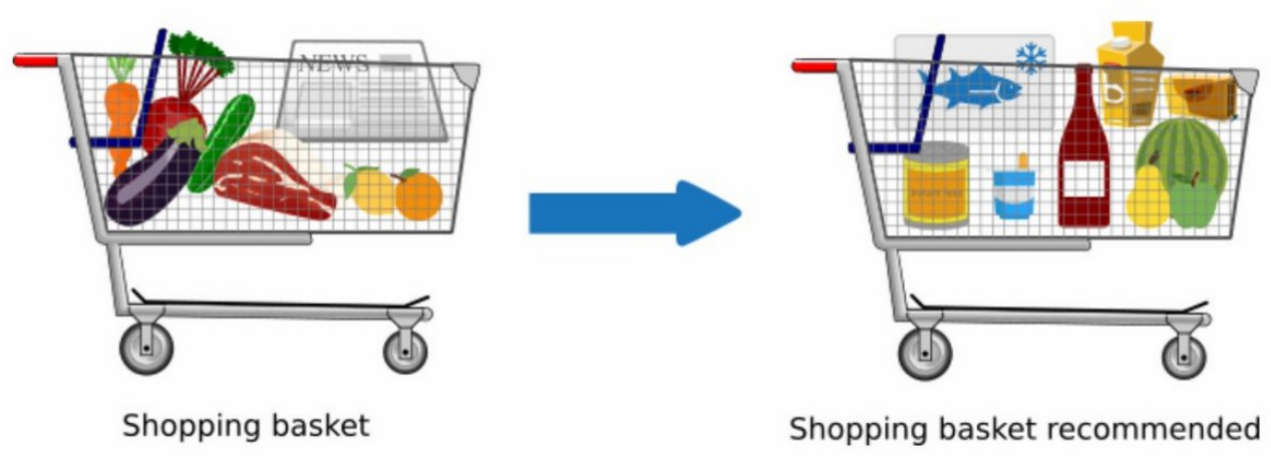
Apriori Algorithm



Apriori Algorithm is a Machine Learning algorithm which is used to gain insight into the structured relationships between different items involved. The most prominent practical application of the algorithm is to recommend products based on the products already present in the user's cart.

Walmart especially has made great use of the algorithm in suggesting products to it's users.

Components of Apriori algorithm

The given three components comprise the apriori algorithm.

1.Support

2.Confidence

3.Lift

Let's take an example to understand this concept.

We have already discussed above; you need a huge database containing a large no of transactions. Suppose you have 4000 customers transactions in a Big Bazar. You have to calculate the Support, Confidence, and Lift for two products, and you may say Biscuits and Chocolate. This is because customers frequently buy these two items together.

Out of 4000 transactions, 400 contain Biscuits, whereas 600 contain Chocolate, and these 600 transactions include a 200 that includes Biscuits and chocolates. Using this data, we will find out the support, confidence, and lift.

Support

Support refers to the default popularity of any product. You find the support as a quotient of the division of the number of transactions comprising that product by the total number of transactions. Hence, we get

Support (Biscuits) = (Transactions relating biscuits) / (Total transactions)

= 400/4000 = 10 percent.

Confidence

Confidence refers to the possibility that the customers bought both biscuits and chocolates together. So, you need to divide the number of transactions that comprise both biscuits and chocolates by the total number of transactions to get the confidence.

Hence,

Confidence = (Transactions relating both biscuits and Chocolate) / (Total transactions involving Biscuits)

= 200/400

= 50 percent.

Lift

Consider the above example; lift refers to the increase in the ratio of the sale of chocolates when you sell biscuits. The mathematical equations of lift are given below.

Lift = (Confidence (Biscuits - chocolates)/ (Support (Biscuits)

= 50/10 = 5

It means that the probability of people buying both biscuits and chocolates together is five times more than that of purchasing the biscuits alone. If the lift value is below one, it requires that the people are unlikely to buy both the items together. Larger the value, the better is the combination.

We will understand this algorithm with the help of an example.

Let’s Code

MLxtend can be installed using pip, so make sure that is done before trying to execute any of the code below. Once it is installed, the code below shows how to get it up and running.

Pandas and MLxtend code imported and read the data:

```
import pandas as pd
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
df = pd.read_excel('https://modcom.co.ke/api/OnlineRetail.csv')
df.head()
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39
2	536365	84406B	CREAM CUPID HEARTS C OAT HANGER	8	2010-12-01 08:26:00	2.75
3	536365	84029G	KNITTED UNION FLAG HO T WATER BOTTLE	6	2010-12-01 08:26:00	3.39
4	536365	84029E	RED WOOLLY HOTTIE WH ITE HEART.	6	2010-12-01 08:26:00	3.39

We drop the rows that don’t have invoice numbers

```
df['Description'] = df['Description'].str.strip()
df.dropna(axis=0, subset=['InvoiceNo'], inplace=True)
df['InvoiceNo'] = df['InvoiceNo'].astype('str')
```

After the cleanup, we need to consolidate the items into 1 transaction per row. For the sake of keeping the data set small, I'm only looking at sales for France. However, in additional code below, I will compare these results to sales from Germany. Further country comparisons would be interesting to investigate.

```
basket = (df[df['Country'] =="France"]
          .groupby(['InvoiceNo', 'Description'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('InvoiceNo'))
```

Here's what the first few columns look like (note, I added some numbers to the columns to illustrate the concept - the actual data in this example is all 0's):

Description	10 COLOUR SPACEBOY PEN	12 COLOURED PARTY BALLOONS	12 EGG HOUSE PAINTED WOOD
InvoiceNo			
536370	11.0	0.0	0.0
536852	0.0	0.0	0.0
536974	0.0	0.0	0.0
537065	0.0	0.0	0.0
537463	0.0	0.0	9.0

There are a lot of zeros in the data but we also need to make sure any positive values are converted to a 1 and anything less the 0 is set to 0. This step will complete the one hot encoding of the data and remove the postage column (since that charge is not one we wish to explore):

```
def encode_units(x):
    if x <= 0:
        return 0
    if x >= 1:
        return 1


basket_sets = basket.applymap(encode_units)
basket_sets.drop('POSTAGE', inplace=True, axis=1)
```

Now that the data is structured properly, we can generate frequent item sets that have a support of at least 7% (this number was chosen so that I could get enough useful examples):

```
frequent_itemsets = apriori(basket_sets, min_support=0.07,
use_colnames=True)
```

The final step is to generate the rules with their corresponding support, confidence and lift:

```
rules = association_rules(frequent_itemsets, metric="lift",
min_threshold=1)

rules.head()
```

	antecedants	consequents	support	confidence	lift
0	(PLASTERS IN TIN WOODLAND ANIMALS)	(PLASTERS IN TIN CIRCUS PARADE)	0.170918	0.597015	3.545907
1	(PLASTERS IN TIN CIRCUS PARADE)	(PLASTERS IN TIN WOODLAND ANIMALS)	0.168367	0.606061	3.545907
2	(PLASTERS IN TIN CIRCUS PARADE)	(PLASTERS IN TIN SPACEBOY)	0.168367	0.530303	3.849607
3	(PLASTERS IN TIN SPACEBOY)	(PLASTERS IN TIN CIRCUS PARADE)	0.137755	0.648148	3.849607
4	(PLASTERS IN TIN WOODLAND ANIMALS)	(PLASTERS IN TIN SPACEBOY)	0.170918	0.611940	4.442233

That’s all there is to it! Build the frequent items using `apriori` then build the rules with `association_rules` .

For instance, we can see that there are quite a few rules with a high lift value which means that it occurs more frequently than would be expected given the number of transaction and product combinations. We can also see several where the confidence is high as well.

We can filter the dataframe using standard pandas code. In this case, look for a large lift (6) and high confidence (.8):

```
rules[ (rules['lift'] >= 6) &

       (rules['confidence'] >= 0.8) ]
```

	antecedants	consequents	support	confidence	lift
8	(SET/6 RED SPOTTY PAPER CUPS)	(SET/6 RED SPOTTY PAPER PLATES)	0.137755	0.888889	6.968889
9	(SET/6 RED SPOTTY PAPER PLATES)	(SET/6 RED SPOTTY PAPER CUPS)	0.127551	0.960000	6.968889
10	(ALARM CLOCK BAKELIKE GREEN)	(ALARM CLOCK BAKELIKE RED)	0.096939	0.815789	8.642959
11	(ALARM CLOCK BAKELIKE RED)	(ALARM CLOCK BAKELIKE GREEN)	0.094388	0.837838	8.642959
16	(SET/6 RED SPOTTY PAPER CUPS, SET/6 RED SPOTTY...	(SET/20 RED RETROSPOT PAPER NAPKINS)	0.122449	0.812500	6.125000
17	(SET/6 RED SPOTTY PAPER CUPS, SET/20 RED RETRO...	(SET/6 RED SPOTTY PAPER PLATES)	0.102041	0.975000	7.644000
18	(SET/6 RED SPOTTY PAPER PLATES, SET/20 RED RET...	(SET/6 RED SPOTTY PAPER CUPS)	0.102041	0.975000	7.077778
22	(SET/6 RED SPOTTY PAPER PLATES)	(SET/20 RED RETROSPOT PAPER NAPKINS)	0.127551	0.800000	6.030769

In looking at the rules, it seems that the green and red alarm clocks are purchased together and the red paper cups, napkins and plates are purchased together in a manner that is higher than the overall probability would suggest.

At this point, you may want to look at how much opportunity there is to use the popularity of one product to drive sales of another. For instance, we can see that we sell 340 Green Alarm clocks but only 316 Red Alarm Clocks so maybe we can drive more Red Alarm Clock sales through recommendations?

```
basket['ALARM CLOCK BAKELIKE GREEN'].sum()

340.0

basket['ALARM CLOCK BAKELIKE RED'].sum()


316.0
```

What is also interesting is to see how the combinations vary by country of purchase. Let’s check out what some popular combinations might be in Germany:

```
basket2 = (df[df['Country'] =="Germany"]

          .groupby(['InvoiceNo', 'Description'])['Quantity']

          .sum().unstack().reset_index().fillna(0)
```

```
          .set_index('InvoiceNo'))


basket_sets2 = basket2.applymap(encode_units)

basket_sets2.drop('POSTAGE', inplace=True, axis=1)

frequent_itemsets2 = apriori(basket_sets2, min_support=0.05,
use_colnames=True)

rules2 = association_rules(frequent_itemsets2, metric="lift",
min_threshold=1)


rules2[ (rules2['lift'] >= 4) &

        (rules2['confidence'] >= 0.5)]
```

	antecedants	consequents	support	confidence	lift
7	(PLASTERS IN TIN SPACEBOY)	(PLASTERS IN TIN WOODLAND ANIMALS)	0.107221	0.571429	4.145125
9	(PLASTERS IN TIN CIRCUS PARADE)	(PLASTERS IN TIN WOODLAND ANIMALS)	0.115974	0.584906	4.242887
10	(RED RETROSPOT CHARLOTTE BAG)	(WOODLAND CHARLOTTE BAG)	0.070022	0.843750	6.648168

It seems that in addition to David Hasselhoff, Germans love Plasters in Tin Spaceboy and Woodland Animals.

End

Useful Links

https://www.geeksforgeeks.org/apriori-algorithm/

https://en.wikipedia.org/wiki/Apriori_algorithm