



Introduction to Association Rule Mining – Market Basket Analysis (MBA)

Association rule mining is a data mining technique used to uncover **if-then relationships** between items in large datasets, such as retail transaction logs. These relationships, known as **association rules**, capture patterns of items that frequently occur together (called **co-occurrences**).

In the context of **market basket analysis**, association rules help answer questions like: *“Which products are often bought together?”* For example, suppose we find that 75% of customers who buy cereal also buy milk. We can express this as the rule:

$$\{\text{cereal}\} \Rightarrow \{\text{milk}\}$$

This rule suggests that customers who purchase cereal often purchase milk as well. Such insights can guide **marketing and retail decisions**, including promotional strategies, product bundling, and shelf placement.

Market Basket Analysis



للاستفسار فُرعي 108 - 06/5814168			
=====			
رقم الفاتة	62464	الرقم الضريبي	16650476
=====			
كاش 6			
التاريخ	28/05/2025	الساعة	3:34:00 pm
ف المادة	الكمية	السعر	المجموع
ارم اند همر الدفاتس	1	1.350	1.350
ارم اند همر معجون اسنان	1	1.350	1.350
معجون اورل بي 123	1	1.390	1.390
المراعي لينة 500 غم	1	2.250	2.250
صابون حمام نابلسي حبة و	1	0.550	0.550
عصير الربيع اناناس 1 لتر	1	1.400	1.400
عصير الربيع كوكتيل 1 لتر	1	1.400	1.400
هاربيك سبترس 495 مل	1	1.200	1.200
هاربيك لافندر 495 مل	1	1.200	1.200
كفوف لاتيكس اسود وسط*	1	2.490	2.490
تشيرورز غسل 375 غم	1	3.750	3.750
رايس كيك طبيعي 100 غم	2	0.950	1.900
رايس كيك سمسم 100 غم	1	0.950	0.950
غاستو مناديل مبللة 96	1	0.990	0.990
ساتينو معجون اسنان الحس	1	0.990	0.990
جينة برمران ايطالي	0.166	12.990	2.156
جينة برمران ايطالي	0.354	12.990	4.598
فلل قرن الغزال	0.552	0.650	0.359
بندورة عناقيد كبيرة	0.7	1.100	0.770
مخلل هلابينو	0.284	1.990	0.565
خيار	0.44	0.790	0.348

المجموع النهائي:			31.956
فيزا :			31.956
المدفوع			0.000
الباقي			0.000
عدد المواد :			22

شكرا لزيارتكم			
لا يتم تبديل البضاعة الا بوجود			
فاتورة شراء خلال 24 ساعة			

للاستفسار فُرعي 108 - 06/5814168			
=====			
رقم الفاتة	62465	الرقم الضريبي	16650476
=====			
كاش 6			
التاريخ	28/05/2025	الساعة	3:35:00 pm
ف المادة	الكمية	السعر	المجموع
نبيل دجاج تندر 900 غم	1	3.750	3.750
نبيل اصابع سمك محمد 40	2	2.390	4.780
خضار مشكلة السنبلة 450	2	1.600	3.200
السنبلة بامية اكسترا 400	2	2.590	5.180
سباتخ السنبلة 400 غم	2	0.900	1.800
جينة فاكيروم روابي جرش 3	1	3.550	3.550
مكدوس بلدي التميمي 1 كغ	1	3.650	3.650
بردايس خبز طابون 4*	1	0.600	0.600
غيث خبز مسخن 5*	1	0.850	0.850
مخلل هلابينو	0.494	1.990	0.983
ديلايت قشقوان	0.3	8.190	2.457
مشمش حموي سوبر	0.796	1.750	1.393
سمسمية مصرية	0.816	1.750	1.428
مفروم عجل + خاروف	0.335	8.490	2.844
مفروم عجل + خاروف	0.355	8.490	3.014
دجاج تندوري مقطع متبل 1.98	0.98	3.250	3.185

المجموع النهائي:			42.664
فيزا :			42.664
المدفوع			0.000
الباقي			0.000
عدد المواد :			20

شكرا لزيارتكم			
لا يتم تبديل البضاعة الا بوجود			
فاتورة شراء خلال 24 ساعة			



✧ Association Rule Use Cases and Domains

Association rule mining can be applied in various domains beyond market basket analysis. Here are some examples:

1. Healthcare

- **Disease Diagnosis:** Identifying associations between symptoms and diseases. For example, a rule like $\{fever, cough\} \rightarrow \{flu\}$ can help predict diseases.
- **Drug Interactions:** Discovering relationships between medications that are frequently prescribed together or identifying combinations that lead to adverse reactions.

2. Web Usage Mining

- **Website Optimization:** Analyzing user navigation patterns to determine common paths or clicks, e.g., $\{homepage \rightarrow product\ page\} \rightarrow \{checkout\}$.
- **Recommendation Systems:** Suggesting content or products based on frequently co-accessed items, e.g., $\{clicked\ 'smartphone'\} \rightarrow \{clicked\ 'smartphone\ accessories'\}$.

3. Education

- **Student Behavior Analysis:** Discovering patterns in course enrollment, such as $\{math101, cs101\} \rightarrow \{stat101\}$.
- **Learning Paths:** Identifying sequences of topics that students study, helping design better curricula.

4. Telecommunications

- **Customer Churn Analysis:** Detecting combinations of usage patterns that are associated with customers leaving the service, e.g., $\{low\ data\ usage, few\ calls\} \rightarrow \{churn\}$.
- **Service Bundling:** Identifying services that are commonly purchased together, like $\{broadband, mobile\} \rightarrow \{TV\ subscription\}$.

5. Banking and Finance

- **Fraud Detection:** Uncovering patterns associated with fraudulent transactions, e.g., $\{high\ transaction\ frequency, odd\ hours\} \rightarrow \{fraud\}$.
- **Loan Approvals:** Identifying attributes of successful loan applications, such as $\{high\ income, good\ credit\ score\} \rightarrow \{loan\ approved\}$.

6. Manufacturing

- **Fault Detection:** Identifying combinations of machine conditions that frequently result in faults, e.g., $\{high\ temperature, low\ pressure\} \rightarrow \{equipment\ failure\}$.
- **Supply Chain Optimization:** Discovering patterns in material usage, e.g., $\{material\ A, material\ B\} \rightarrow \{product\ C\}$.

7. Retail Beyond Market Basket

- **Shelf Placement:** Finding products that are often bought together to optimize store layout.
- **Customer Segmentation:** Identifying customer groups with similar purchasing behaviors, e.g., $\{frequent\ discount\ purchases\} \rightarrow \{low\ brand\ loyalty\}$.

8. Social Media Analysis

- **Trending Topics:** Discovering associations between hashtags, e.g., $\{\#climatechange, \#sustainability\} \rightarrow \{\#renewableenergy\}$.
- **User Behavior Patterns:** Understanding engagement behaviors, such as $\{likes\ post, comments\ on\ post\} \rightarrow \{shares\ post\}$.

9. Energy Sector

- **Usage Patterns:** Identifying associations in energy usage, like $\{high\ A/C\ usage, weekend\} \rightarrow \{peak\ energy\ consumption\}$.
- **Smart Grid Analysis:** Detecting patterns for predictive maintenance, e.g., $\{low\ voltage, high\ demand\} \rightarrow \{power\ outage\}$.

10. Transportation and Logistics

- **Traffic Analysis:** Discovering patterns in traffic conditions, e.g., $\{morning\ rush\ hour, bad\ weather\} \rightarrow \{traffic\ jam\}$.
- **Route Optimization:** Identifying frequently used delivery routes, such as $\{route\ A, route\ B\} \rightarrow \{delivered\ faster\}$.

11. E-commerce

- **User Preferences:** Identifying patterns in user preferences for personalized recommendations.
- **Cross-Selling:** Suggesting related products based on purchase history.

12. Sports Analytics

- **Performance Metrics:** Discovering combinations of player actions that lead to victories, e.g., $\{high\ possession, accurate\ passes\} \rightarrow \{win\}$.
- **Injury Prevention:** Identifying conditions that precede injuries, such as $\{high\ training\ load, lack\ of\ rest\} \rightarrow \{injury\ risk\}$.

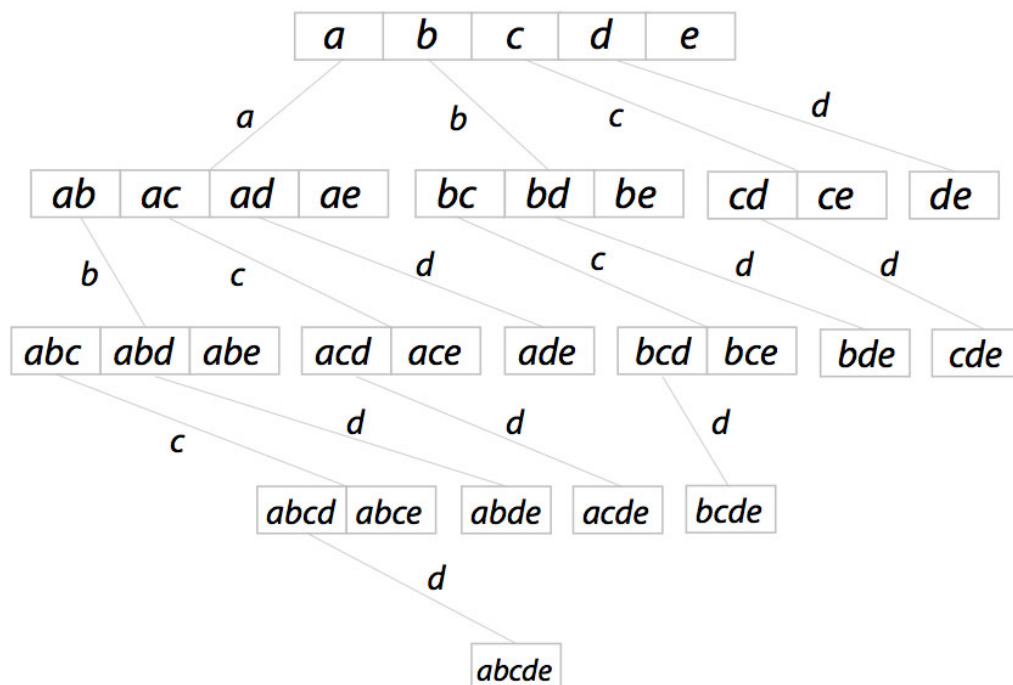
✓ The Apriori Algorithm

Apriori is a classic algorithm for **finding frequent itemsets** and **deriving association rules** in transactional databases (e.g., shopping baskets, web logs). It is based on the

idea that **every subset of a frequent itemset must also be frequent**.

Key points:

- Proposed by **Agrawal & Srikant (1994)**.
- Works on **transaction data** (e.g., sets of items bought together).
- First finds **frequent single items**, then **grows** them into larger itemsets as long as they remain frequent.
- Uses a **bottom-up, level-wise** search with **candidate generation** and testing.
- Stops when **no more frequent extensions** can be found.
- The resulting frequent itemsets are used to build **association rules** (e.g., for **market basket analysis**).

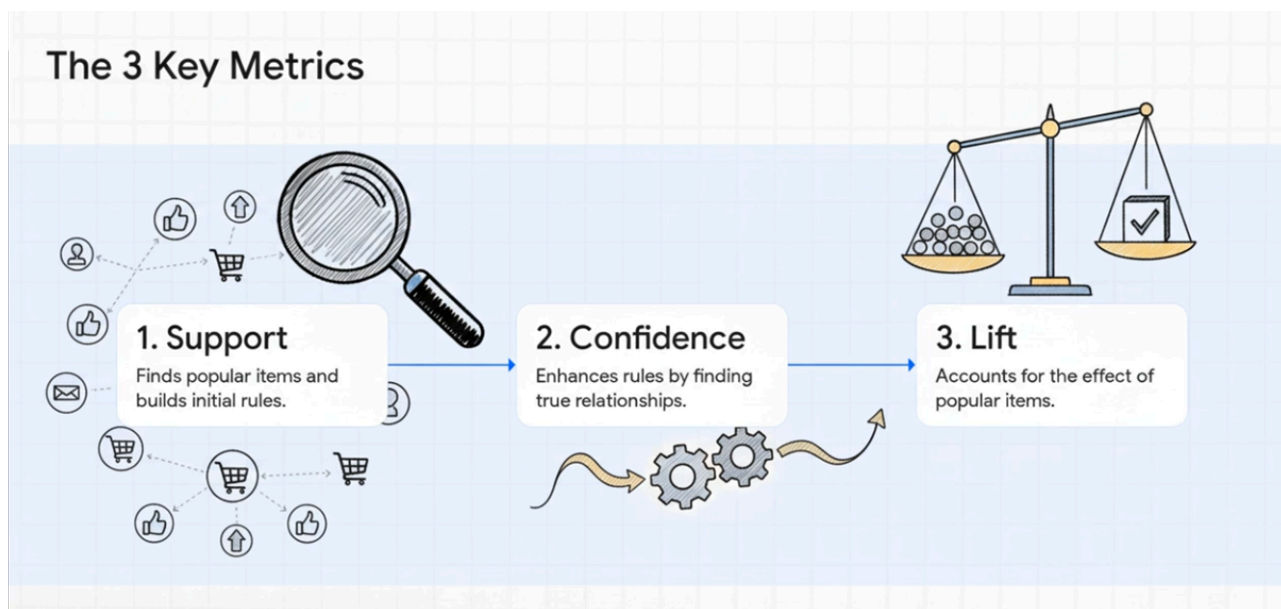


.....

✓ Creating Association Rules

To create association rules, we use 3 metrics:

1. Support: finds the popular items and builds initial rules.
2. Confidence: Enhance initial rules by finding true relationships.
3. Lift: Enhance rules by acomodating for popular items effect (x -> y where x is popular e.g. bread in a bakery)



1. Support

Support measures **how often** an itemset appears in the dataset. It is a probability-like measure between 0 and 1.

Definition

$$\text{Support}(X) = \frac{\text{Number of transactions containing } X}{\text{Total transactions}}$$

Range

- $0 \leq \text{Support}(X) \leq 1$
 - $0 \rightarrow X$ never appears
 - $1 \rightarrow X$ appears in every transaction

Interpretation

- **High support** $\rightarrow X$ is common, good candidate for rules
- **Low support** $\rightarrow X$ is rare, often less useful (unless domain-specific)

Thresholds in practice

- Set a **minimum support** to filter out infrequent itemsets.
- Typical starting points:
 - **5–10%** for large retail datasets
 - **20–30%** for smaller datasets or only very popular combinations
- Too high \rightarrow you miss interesting but less frequent patterns
- Too low \rightarrow you keep many unimportant, noisy patterns

Example

Let's assume that we have a small dataset with 12 transactions:

Transaction ID	Item 1	Item 2	Item 3	Item 4	Item 5
1	Milk	Egg	Bread	Butter	
2	Milk	Butter	Egg	Ketchup	
3	Bread	Butter	Ketchup		
4	Milk	Bread	Butter		
5	Bread	Butter	Cookies		
6	Milk	Bread	Butter	Cookies	
7	Milk	Cookies			
8	Milk	Bread	Butter		
9	Bread	Butter	Egg	Cookies	
10	Milk	Butter	Bread		
11	Milk	Bread	Butter		
12	Milk	Bread	Cookies	Ketchup	

First, we set out the **Minimum Support** value to:

- 50% (focus on items present in at least half of the transactions), this value can be set to 5% or 10% in real-life situations with large datasets.

Step 1: Dataset Transformation Convert the dataset into a binary format:

Transaction	Milk	Egg	Bread	Butter	Ketchup	Cookies
1	1	1	1	1	0	0
2	1	1	0	1	1	0
3	0	0	1	1	1	0
4	1	0	1	1	0	0
5	0	0	1	1	0	1
6	1	0	1	1	0	1
7	1	0	0	0	0	1
8	1	0	1	1	0	0
9	0	1	1	1	0	1
10	1	0	1	1	0	0
11	1	0	1	1	0	0
12	1	0	1	0	1	1

Step 2: Identify Frequent 1-Itemsets

The algorithm starts by calculating the support for each single item.

Item	Support Count	Support (%)	Frequent?
Milk	9	75%	Yes

Item	Support Count	Support (%)	Frequent?
Egg	3	25%	No
Bread	10	83%	Yes
Butter	10	83%	Yes
Ketchup	3	25%	No
Cookies	5	42%	No

Frequent 1-itemsets: {Milk}, {Bread}, {Butter}.

Step 3: Generate 2-Itemsets

Now, combine frequent 1-itemsets into 2-itemsets and calculate their support.

Itemset	Support Count	Support (%)	Frequent?
{Milk, Bread}	7	58%	Yes
{Milk, Butter}	7	58%	Yes
{Bread, Butter}	9	75%	Yes

Frequent 2-itemsets: {Milk, Bread}, {Milk, Butter}, {Bread, Butter}.

Step 4: Generate 3-Itemsets

Combine frequent 2-itemsets into 3-itemsets and calculate their support.

Itemset	Support Count	Support (%)	Frequent?
{Milk, Bread, Butter}	6	50%	Yes

Frequent 3-itemset: {Milk, Bread, Butter}.

Step 5: No Larger Itemsets Can Be Created

Since there are no frequent 4-itemsets (support would drop below 50%), we stop here.

Step 6: Generate Initial Association Rules

Now, use the frequent itemsets to generate association rules, we will focus on the largest item set, we can create rules from smaller ones if needed.

Rules from {Milk, Bread, Butter}:

1. Rule 1:

Milk, Bread \rightarrow Butter

$$\text{Support} = P(\text{Milk, Bread, Butter}) = 6/12 = 50\%$$

1. Rule 2:

Milk, Butter \rightarrow Bread

$$\text{Support} = P(\text{Milk, Bread, Butter}) = 6/12 = 50\%$$

2. Rule 3:Bread, Butter \rightarrow Milk

$$\text{Support} = P(\text{Milk, Bread, Butter}) = 6/12 = 50\%$$

.....

✓ 2. Confidence

Why Support Is Not EnoughSupport is useful but **symmetric**:support(milk \cap diapers)

is the same for both rules:

- Milk \rightarrow Diapers
- Diapers \rightarrow Milk

It tells us **how often they appear together**, but **not** which direction is more useful.

Example: in 1,000 transactions, milk and diapers appear together 100 times:

$$\text{support}(\text{milk} \cap \text{diapers}) = \frac{100}{1000} = 10\%.$$

So the pair is common, but support alone cannot tell us which rule to use.

Given a frequent itemset like {Milk, Bread}, we can form two rules:

Milk \rightarrow Bread, Bread \rightarrow Milk.To choose the more useful rule, we use **confidence**.**Definition**

Confidence measures how often (Y) appears when (X) appears (how reliable the rule is):

$$\text{conf}(X \rightarrow Y) = \frac{\text{support}(X \cap Y)}{\text{support}(X)}.$$

- **Range:** ($0 \leq \text{conf}(X \rightarrow Y) \leq 1$)
 - 0 \rightarrow the rule never holds
 - 1 \rightarrow the rule always holds

Interpretation

- **High confidence (~1):** when (X) happens, (Y) almost always happens (strong rule).
- **Low confidence (~0):** when (X) happens, (Y) rarely happens (weak rule).

In practice, we often set a **minimum confidence threshold** (e.g., 70–80%) and keep only rules above that value.

.....

Step 7: Validate the Rules using Confidence

First, let us set the **Minimum Confidence** value to: 70%.

Rules from {Milk, Bread, Butter}:

1. Rule 1:

Milk, Bread \rightarrow Butter

$$\text{Confidence} = P(\text{Butter}|\text{Milk, Bread}) = \frac{P(\text{Milk, Bread, Butter})}{P(\text{Milk, Bread})} = \frac{6}{7} = 85$$

1. Rule 2:

Milk, Butter \rightarrow Bread

$$\text{Confidence} = P(\text{Bread}|\text{Milk, Butter}) = \frac{P(\text{Milk, Bread, Butter})}{P(\text{Milk, Butter})} = \frac{6}{7} = 85$$

1. Rule 3:

Bread, Butter \rightarrow Milk

$$\text{Confidence} = P(\text{Milk}|\text{Bread, Butter}) = \frac{P(\text{Milk, Bread, Butter})}{P(\text{Bread, Butter})} = \frac{6}{9} \approx 66$$

.....

✓ 2.3. Lift Metric

Confidence alone can be **misleading** when the consequent (Y) is very frequent: a rule (X \rightarrow Y) may have high confidence just because (Y) is common.

Example (bakery)

- $P(\text{bread}) = 0.80$
- $P(\text{cake}) = 0.20$
- $P(\text{cake} \cap \text{bread}) = 0.16$

Then:

$$\text{conf}(\text{cake} \rightarrow \text{bread}) = \frac{0.16}{0.20} = 0.8$$

Confidence is 80%, but bread is already in 80% of all orders, so cake buyers are **not** more likely than average to buy bread.

To correct for this, we use **lift**:

$$\text{Lift}(X \rightarrow Y) = \frac{\text{Confidence}(X \rightarrow Y)}{\text{Support}(Y)} = \frac{P(Y|X)}{P(Y)}$$

In the example:

$$\text{lift}(\text{cake} \rightarrow \text{bread}) = \frac{0.8}{0.8} = 1$$

Lift = 1 \Rightarrow no real association beyond chance.

- **Lift > 1**: positive association
- **Lift = 1**: independent
- **Lift < 1**: negative association

Interpreting Lift

Lift measures how strongly two items or itemsets are associated:

- (lift > 1): items occur together **more often than expected** (positive association).
- (lift = 1): items occur together **as often as expected** under independence (no association).
- (lift < 1): items occur together **less often than expected** (negative association).

So lift tells us whether items co-occur more (or less) frequently than expected under independence, making it a useful complement to confidence.

Step 8: Validate Rules using the Lift

First, we set the **Lift > 1** for actionable rules

1. Rule 1:

Milk, Bread \rightarrow Butter

$$\text{Lift} = \frac{P(\text{Butter}|\text{Milk, Bread})}{P(\text{Butter})} = \frac{0.85}{0.83} \approx 1.02$$

2. Rule 2:

Milk, Butter \rightarrow Bread

$$\text{Lift} = \frac{P(\text{Bread}|\text{Milk, Butter})}{P(\text{Bread})} = \frac{0.85}{0.83} \approx 1.02$$

3. Rule 3:

Bread, Butter \rightarrow Milk

$$\text{Lift} = \frac{P(\text{Milk}|\text{Bread, Butter})}{P(\text{Milk})} = \frac{0.66}{0.75} \approx .88$$

Final Results

Frequent Itemsets:

- **1-itemsets:** {Milk}, {Bread}, {Butter}
- **2-itemsets:** {Milk, Bread}, {Milk, Butter}, {Bread, Butter}
- **3-itemset:** {Milk, Bread, Butter}

Rules:

Rule	Support	Confidence	Lift	Actionable?
Milk, Bread \rightarrow Butter	50%	85%	1.02	Yes
Milk, Butter \rightarrow Bread	50%	85%	1.02	Yes
Bread, Butter \rightarrow Milk	50%	66%	.88	No

Interpretation and Actionable Insights

Rule	Explanation
Milk, Bread \rightarrow Butter	Customers buying Milk and Bread are highly likely (85%) to also buy Butter. Consider bundl
Milk, Butter \rightarrow Bread	Strong association; placing these items together could increase sales.
Bread, Butter \rightarrow Milk	Suggests that Milk is a complementary product to Bread and Butter, Weak lift thought

Python Example

We will implement the **Apriori algorithm** using the `mlxtend` library for frequent itemset mining and rule generation.

a. Install Required Libraries Make sure you have the required libraries installed:

```
pip install pandas mlxtend
```

b. Load the Dataset We will represent the dataset as a **binary transaction matrix** where each row is a transaction, and each column is an item (1 = purchased, 0 = not purchased).

```

1 import pandas as pd
2
3 # Define the dataset
4 data = {
5     "Milk":    [1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1],
6     "Egg":     [1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
7     "Bread":   [1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1],
8     "Butter":  [1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0],
9     "Ketchup": [0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1],
10    "Cookies": [0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1],
11 }
12
13 # Create a DataFrame
14 df = pd.DataFrame(data)
15 print("Transaction Dataset:")
16 df

```

Transaction Dataset:

	Milk	Egg	Bread	Butter	Ketchup	Cookies
0	1	1	1	1	0	0
1	1	1	0	1	1	0
2	0	0	1	1	1	0
3	1	0	1	1	0	0
4	0	0	1	1	0	1
5	1	0	1	1	0	1
6	1	0	0	0	0	1
7	1	0	1	1	0	0
8	0	1	1	1	0	1
9	1	0	1	1	0	0
10	1	0	1	1	0	0
11	1	0	1	0	1	1

c. Apply the Apriori Algorithm We will use the `mlxtend` library to find frequent itemsets and generate association rules.

```
1 ! pip install mlxtend
```

Collecting mlxtend

Downloading mlxtend-0.23.4-py3-none-any.whl.metadata (7.3 kB)

Requirement already satisfied: scipy>=1.2.1 in /home/me/myenv/lib/python3.12/site-packages (1.11.0)

```
Requirement already satisfied: numpy>=1.16.2 in /home/me/myenv/lib/python3.12/si
Requirement already satisfied: pandas>=0.24.2 in /home/me/myenv/lib/python3.12/s
Requirement already satisfied: scikit-learn>=1.3.1 in /home/me/myenv/lib/python3
Requirement already satisfied: matplotlib>=3.0.0 in /home/me/myenv/lib/python3.1
Requirement already satisfied: joblib>=0.13.2 in /home/me/myenv/lib/python3.12/s
Requirement already satisfied: contourpy>=1.0.1 in /home/me/myenv/lib/python3.12
Requirement already satisfied: cycler>=0.10 in /home/me/myenv/lib/python3.12/sit
Requirement already satisfied: fonttools>=4.22.0 in /home/me/myenv/lib/python3.1
Requirement already satisfied: kiwisolver>=1.3.1 in /home/me/myenv/lib/python3.1
Requirement already satisfied: packaging>=20.0 in /home/me/myenv/lib/python3.12/
Requirement already satisfied: pillow>=8 in /home/me/myenv/lib/python3.12/site-p
Requirement already satisfied: pyparsing>=2.3.1 in /home/me/myenv/lib/python3.12
Requirement already satisfied: python-dateutil>=2.7 in /home/me/myenv/lib/python
Requirement already satisfied: pytz>=2020.1 in /home/me/myenv/lib/python3.12/sit
Requirement already satisfied: tzdata>=2022.7 in /home/me/myenv/lib/python3.12/s
Requirement already satisfied: threadpoolctl>=3.1.0 in /home/me/myenv/lib/python
Requirement already satisfied: six>=1.5 in /home/me/myenv/lib/python3.12/site-pa
Downloading mlxtend-0.23.4-py3-none-any.whl (1.4 MB)
1.4/1.4 MB 5.5 MB/s eta 0:00:00[36m
```

Installing collected packages: mlxtend
Successfully installed mlxtend-0.23.4

```
1 from mlxtend.frequent_patterns import apriori, association_rules
2
3 # Step 1: Generate frequent itemsets with Apriori
4 frequent_itemsets = apriori(df, min_support=0.5, use_colnames=True)
5
6 # Display frequent itemsets
7 print("\nFrequent Itemsets:")
8 print(frequent_itemsets)
```

```
Frequent Itemsets:
  support  itemsets
0  0.750000      (Milk)
1  0.833333      (Bread)
2  0.833333      (Butter)
3  0.583333  (Bread, Milk)
4  0.583333  (Milk, Butter)
5  0.750000  (Bread, Butter)
6  0.500000 (Bread, Milk, Butter)
/home/me/myenv/lib/python3.12/site-packages/mlxtend/frequent_patterns/fpcommon.p
warnings.warn(
```

d. Generate Association Rules We generate rules based on **confidence** and calculate **lift**.

Generate rules based on confidence

```

1 # Step 2: Generate association rules based on confidence
2 rules = association_rules(frequent_itemsets, num_itemsets=len(df), metric='
3
4 # Display the rules
5 print("\nAssociation Rules:")
6 print(rules[['antecedents', 'consequents', 'support', 'confidence', 'lift']]

```

Association Rules:

	antecedents	consequents	support	confidence	lift
0	(Bread)	(Milk)	0.583333	0.700000	0.933333
1	(Milk)	(Bread)	0.583333	0.777778	0.933333
2	(Butter)	(Milk)	0.583333	0.700000	0.933333
3	(Milk)	(Butter)	0.583333	0.777778	0.933333
4	(Butter)	(Bread)	0.750000	0.900000	1.080000
5	(Bread)	(Butter)	0.750000	0.900000	1.080000
6	(Milk, Bread)	(Butter)	0.500000	0.857143	1.028571
7	(Butter, Milk)	(Bread)	0.500000	0.857143	1.028571

We can also generate rules based on lift

```

1 # Step 2: Generate association rules based on lift
2 rules = association_rules(frequent_itemsets, num_itemsets=len(df), metric='
3
4 # Display the rules
5 print("\nAssociation Rules:")
6 print(rules[['antecedents', 'consequents', 'support', 'confidence', 'lift']]

```

Association Rules:

	antecedents	consequents	support	confidence	lift
0	(Butter)	(Bread)	0.75	0.900000	1.080000
1	(Bread)	(Butter)	0.75	0.900000	1.080000
2	(Milk, Bread)	(Butter)	0.50	0.857143	1.028571
3	(Butter, Milk)	(Bread)	0.50	0.857143	1.028571
4	(Bread)	(Butter, Milk)	0.50	0.600000	1.028571
5	(Butter)	(Milk, Bread)	0.50	0.600000	1.028571

e. Filter and Interpret Rules You can filter rules for high lift or specific itemsets.

```

1 # Filter rules with Lift > 1
2 filtered_rules = rules[rules['lift'] > 1]
3 print("\nFiltered Rules with Lift > 1:")
4 print(filtered_rules[['antecedents', 'consequents', 'support', 'confidence']]

```

Filtered Rules with Lift > 1:

	antecedents	consequents	support	confidence	lift
0	(Butter)	(Bread)	0.75	0.900000	1.080000
1	(Bread)	(Butter)	0.75	0.900000	1.080000
2	(Milk, Bread)	(Butter)	0.50	0.857143	1.028571

3	(Butter, Milk)	(Bread)	0.50	0.857143	1.028571
4	(Bread)	(Butter, Milk)	0.50	0.600000	1.028571
5	(Butter)	(Milk, Bread)	0.50	0.600000	1.028571

Interpretation

1. Milk, Bread \rightarrow Butter

- Support = 58%, Confidence = 87.5%, Lift = 1.05.
- Suggest bundling these items for promotions.

2. Milk, Butter \rightarrow Bread

- Similar metrics as above, showing strong relationships.

3. Bread, Butter \rightarrow Milk

- High confidence but slightly weaker lift, still actionable.

Two-Item Rules

If desired, we can focus on association rules where the antecedent (X) contains just **one item** (i.e., 1-item antecedent, 2 items in the rule overall).

Consider the rule:

Milk \rightarrow Bread

Suppose we have **12 transactions**, and we observe:

- **Milk** appears in **9** transactions.
- **Milk and Bread together** appear in **7** transactions.

Then:

- **Support** of the rule is the fraction of all transactions that contain both Milk and Bread: $\text{supp}(\text{Milk} \rightarrow \text{Bread}) = \frac{7}{12} \approx 58\%$
- **Confidence** of the rule is the fraction of Milk-transactions that also contain Bread: $\text{conf}(\text{Milk} \rightarrow \text{Bread}) = \frac{7}{9} \approx 78\%$

So we can write the rule as:

Milk \rightarrow Bread {S = 58%, C = 78%}.

2-Item Rules vs 3-Item Rules

2-Item Rules (e.g., Milk \rightarrow Bread)

- **Simplicity:** Easy to read, explain, and act on.
- **Higher support and confidence (typically):** Fewer items need to co-occur, so these rules tend to appear more often in the data.
- **Broad applicability:** Capture general trends such as Milk \rightarrow Bread, which can inform store layout (placing items nearby) or simple promotions.

3-Item Rules (e.g., Milk, Bread \rightarrow Butter)