Association Rule and its Algorithms

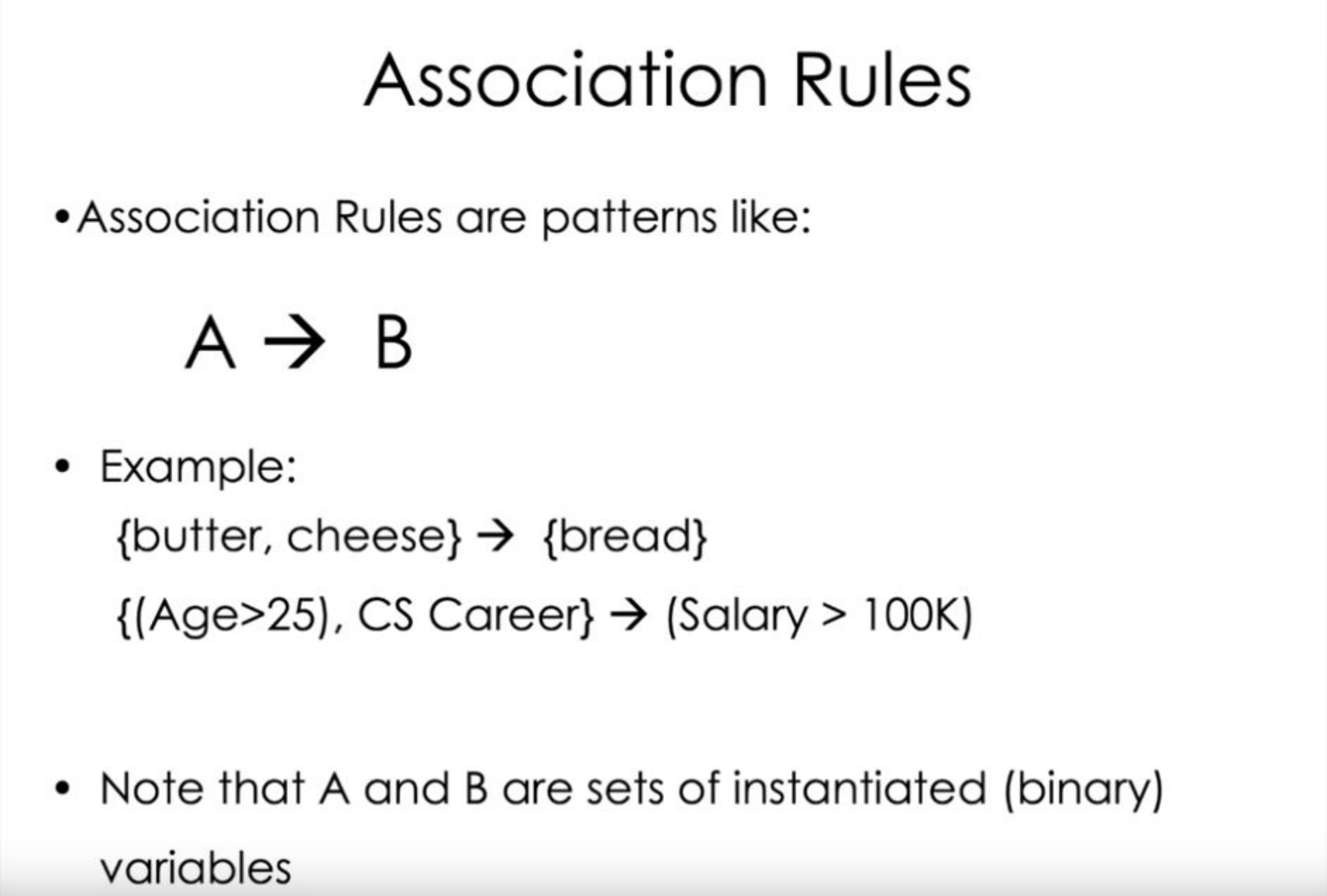
Data Warehousing and Big Data

(DWBI-121)

Masters Of Computer Sciences
Lincoln University College

Feb 16, 2025

Ashesh Shakya
Momik Shrestha



Association Rules

- Association Rules are patterns like:

A → B

- Example:

{butter, cheese} → {bread}

{(Age>25), CS Career} → (Salary > 100K)

- Note that A and B are sets of instantiated (binary) variables

What are association Rules?

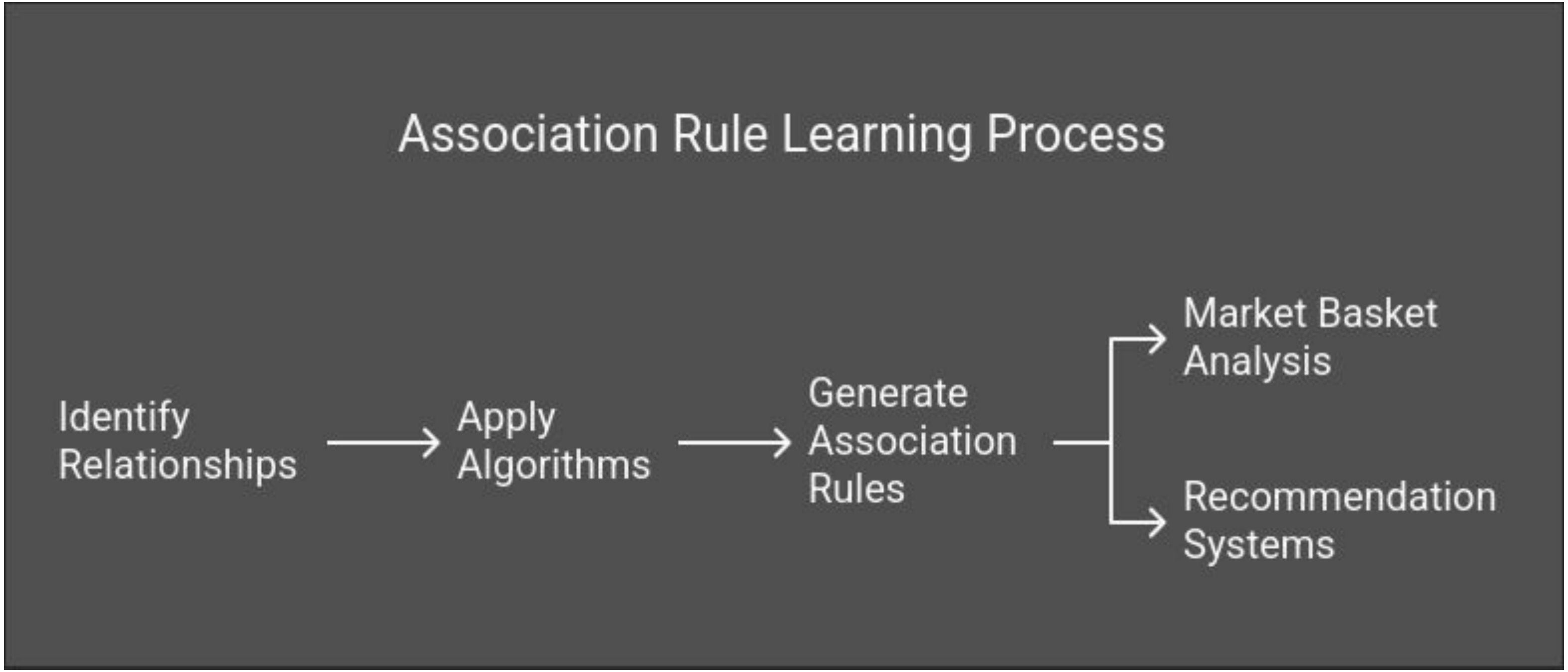
Association rule learning is a fundamental technique in data mining that aims to discover interesting relationships, patterns, or associations among a set of items in large datasets.

At a basic level, association rule mining involves the use of Unsupervised machine learning models to analyze data for patterns, called co-occurrences, in a database.

Applications of Association rules

Business Use Case: Stores can use these rules to optimize product placement, create promotions, or recommend products (e.g., "Customers who bought butter and cheese also bought bread").

Technical Use Case: It’s a way to uncover hidden patterns in data, which can be applied in recommendation systems, fraud detection, or even healthcare.


Association Rule Learning Process
Identify Relationships
Apply Algorithms
Generate Association Rules
Market Basket Analysis
Recommendation Systems

Key Concepts

1. **Itemset**

   - St of one or more items. For example: {Milk, Bread, Beer}
   - K-itemset
     - An itemset that contains k items
     - For example: {Milk, Bread, Beer, Rice} is a 4-itemset

2. **Support Count of an itemset(σ)**

   - Number of transactions that contain the itemset
   - Example: : σ{Milk, Bread, Beer} = 125
     - Means that there are 125 transactions containing all 3 of those productions.

Key Concepts

3. Support (s)

- A measure of how frequently an itemset appears in the dataset.

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

s({Milk, Bread, Beer}) = 125/15000

- **Frequent Itemset:** An itemset whose support(s) is greater than a defined threshold.

Key Concepts

4. Confidence
   - The fraction of times items in Y appear in transaction that contains x

$$\text{Confidence}(A \rightarrow B) = \frac{\text{Support}(A \cup B)}{\text{Support}(A)}$$

Key Concepts

- **Confidence** measures how often the rule A→B is true.

  It is calculated as: Confidence = $\sigma(A \cup B) / \sigma(A)$.

  In the example, if

  σ\{milk, diapers, beer\} = 100

  σ\{milk, diapers\} = 150,

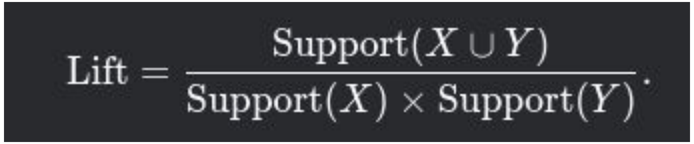
  Then: Confidence = 100/150 = 0.67 = 67%

- This means that 67% of transactions containing milk and diapers also contain beer.

Key Concepts

5. Lift

- How much more likely the outcome is to happen when the condition is met, compared to if it were random
  - Lift = 1 -> independent,
  - Lift > 1 dependent,
  - Lift < 1 substitute



$$\text{Lift} = \frac{\text{Support}(X \cup Y)}{\text{Support}(X) \times \text{Support}(Y)}.$$

- Measures how much the likelihood of buying Y increases after knowing X is also purchased.

Key Concepts

Let’s say:

- s(milk) = 0.4 (40% of transactions contain milk).
- s(bread) = 0.3 (30% of transactions contain bread).
- s(milk, bread) = 0.2 (20% of transactions contain both milk and bread).

Now, plug these values into the lift formula:

Lift(milk→bread)

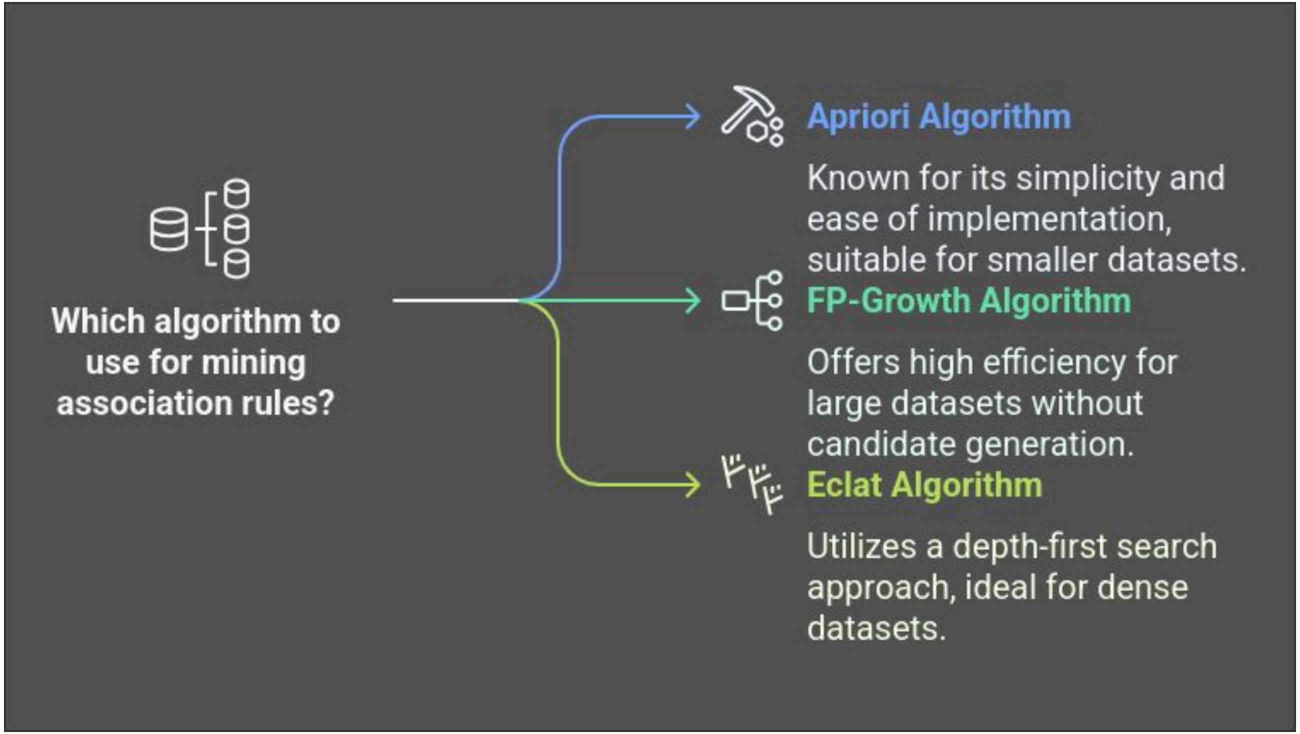
= s(milk,bread)/(s(milk)s(bread))

= 0.2/(0.4×0.3)

=1.67

Lift(milk→bread)=1.67

Algorithms for Association Rule Mining



Which algorithm to use for mining association rules?

Apriori Algorithm

Known for its simplicity and ease of implementation, suitable for smaller datasets.

FP-Growth Algorithm

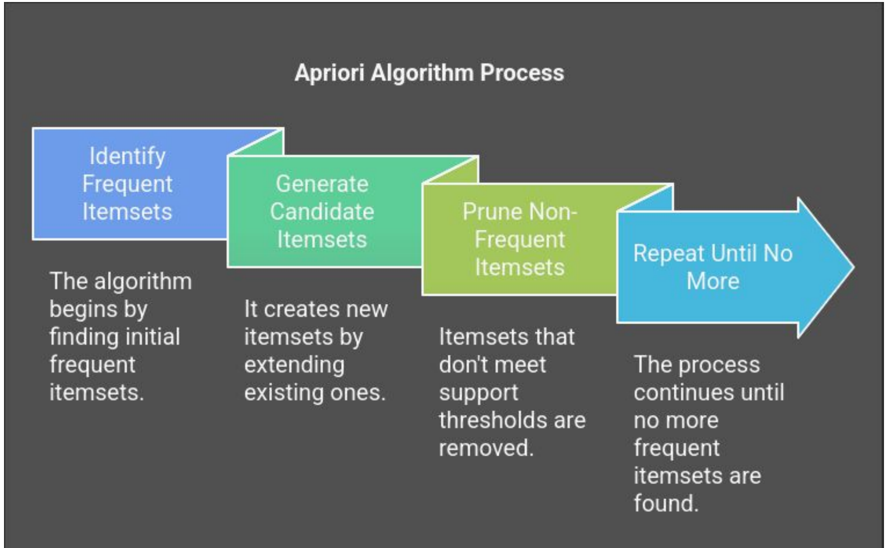
Offers high efficiency for large datasets without candidate generation.

Eclat Algorithm

Utilizes a depth-first search approach, ideal for dense datasets.

1. Apriori Algorithm

- Generates frequent itemsets using a bottom-up approach.
- Works by finding combinations of items that frequently appear together in transactions.
- Key feature: Prunes non-frequent itemsets. (itemsets that have feq < minimum support)



The algorithm begins by finding initial frequent itemsets.

It creates new itemsets by extending existing ones.

Itemsets that don't meet support thresholds are removed.

The process continues until no more frequent itemsets are found.

Step 1: Initial Transactions

- Transaction 1: {Bread, Milk, Eggs}
- Transaction 2: {Bread, Diapers, Beer}
- Transaction 3: {Milk, Diapers, Beer}
- Transaction 4: {Bread, Milk, Diapers, Beer}
- Transaction 5: {Bread, Milk, Cola}

Step 2: Count Item Frequencies

- Bread appears in 4/5 transactions
- Milk appears in 4/5 transactions
- Diapers appears in 3/5 transactions
- Beer appears in 3/5 transactions

Step 3: Set Minimum Support (e.g., 60%)

Qualifying Frequent Itemsets:

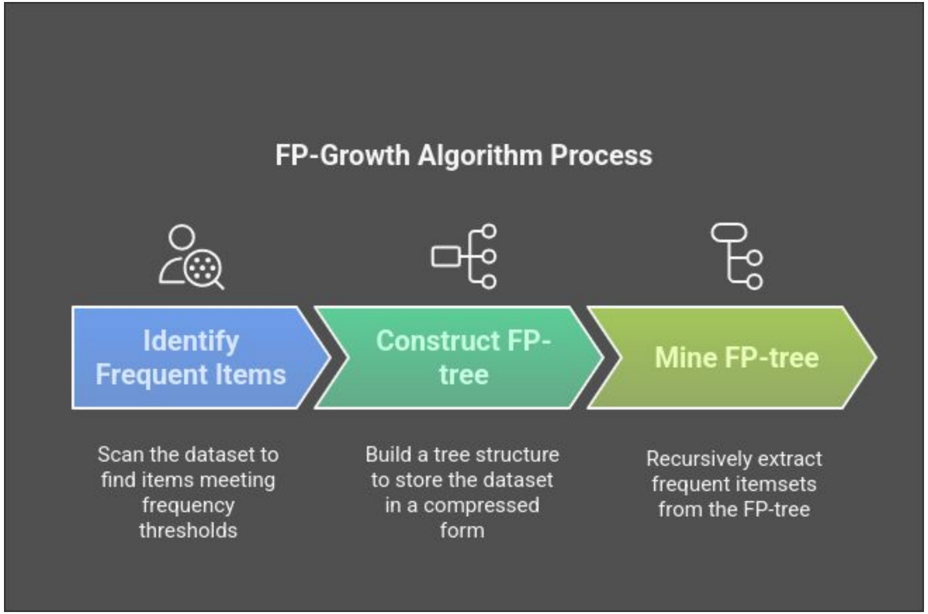
- Bread (80%)
- Milk (80%)
- Diapers (60%)
- Beer (60%)

Step 4: Generate Rules

- If Bread, then likely Milk
- If Diapers, then likely Beer
- If Milk, then likely Bread

2. FP-Growth Algorithm

- Uses an FP-tree (Frequent Pattern Tree) to compress the dataset.
- Key feature: Efficiently mines patterns by recursively exploring the FP-tree.

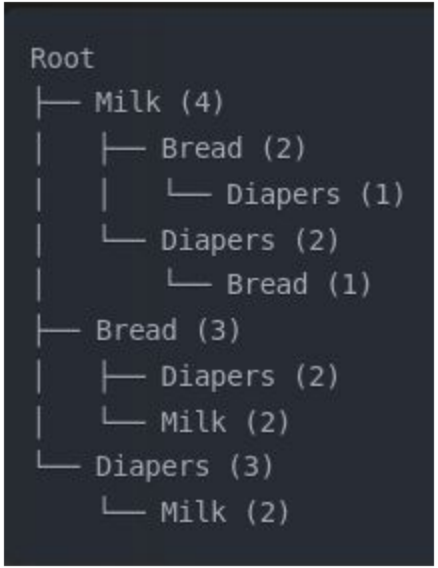


Transactions:

1. {Bread, Milk, Eggs}
2. {Bread, Diapers, Beer}
3. {Milk, Diapers, Beer}
4. {Bread, Milk, Diapers}
5. {Milk, Eggs, Cola}

Item Frequency (Sorted):

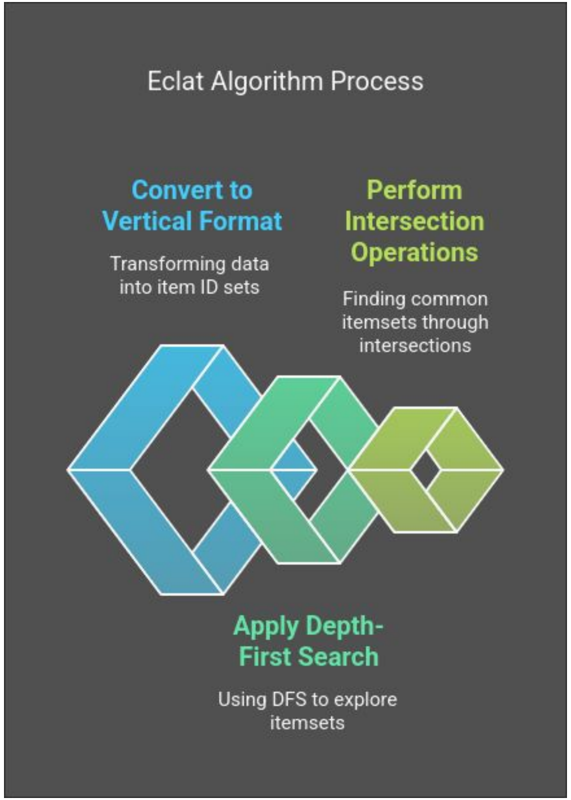
- Milk: 4 times
- Bread: 3 times
- Diapers: 3 times
- Beer: 2 times
- Eggs: 2 times



```
Root
├── Milk (4)
│   ├── Bread (2)
│   │   └── Diapers (1)
│   └── Diapers (2)
│       └── Bread (1)
├── Bread (3)
│   ├── Diapers (2)
│   └── Milk (2)
└── Diapers (3)
    └── Milk (2)
```

3. Eclat Algorithm

- Generates frequent itemsets using a depth-first search approach.
- Key feature: Computes intersections of TID sets to find frequent itemsets.
- Requires only a single scan of the dataset to convert transactions into vertical format.



Eclat Algorithm Process

Convert to Vertical Format

Transforming data into item ID sets

Perform Intersection Operations

Finding common itemsets through intersections

Apply Depth-First Search

Using DFS to explore itemsets

Transactions:

1. {Milk, Bread, Eggs}
2. {Bread, Diapers, Beer}
3. {Milk, Diapers, Beer}
4. {Bread, Milk, Diapers}
5. {Milk, Eggs, Cola}

Vertical Data Representation:

- Milk: [1, 3, 4, 5]
- Bread: [1, 2, 4]
- Diapers: [2, 3, 4]
- Beer: [2, 3]
- Eggs: [1, 5]
- Cola: [5]

Intersections:

- {Milk, Bread}: [1, 4]
- {Milk, Diapers}: [3, 4]
- {Bread, Diapers}: [2, 4]

Thank You