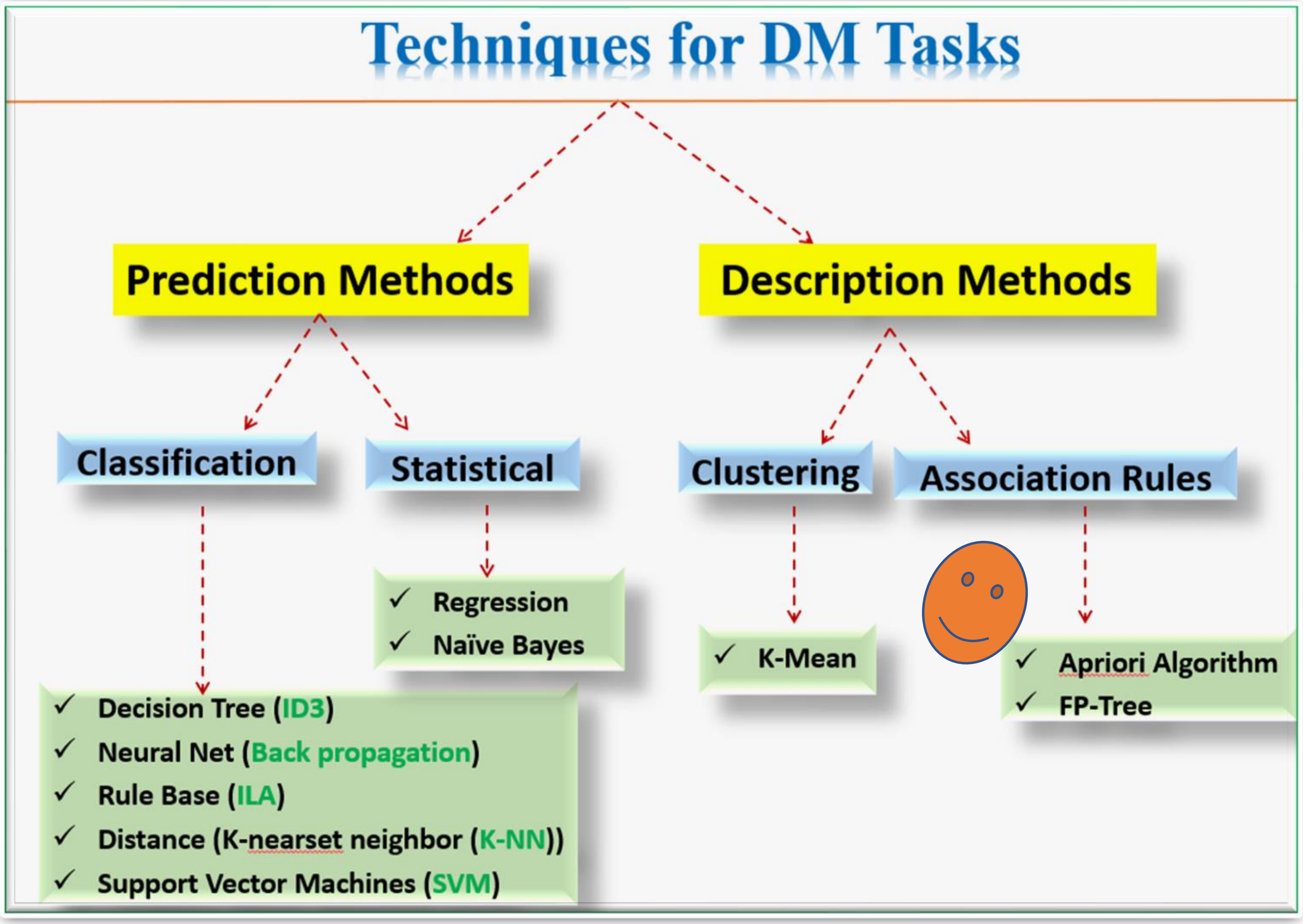
Apriori Algorithm

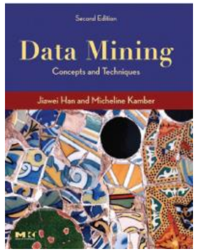
Remember…

Recap of
Previous
Lectures



Techniques for DM Tasks
Prediction Methods
Description Methods
Classification
Statistical
Clustering
Association Rules
✓ Regression
✓ Naïve Bayes
✓ K-Mean
✓ Apriori Algorithm
✓ FP-Tree
✓ Decision Tree (ID3)
✓ Neural Net (Back propagation)
✓ Rule Base (ILA)
✓ Distance (K-nearset neighbor (K-NN))
✓ Support Vector Machines (SVM)

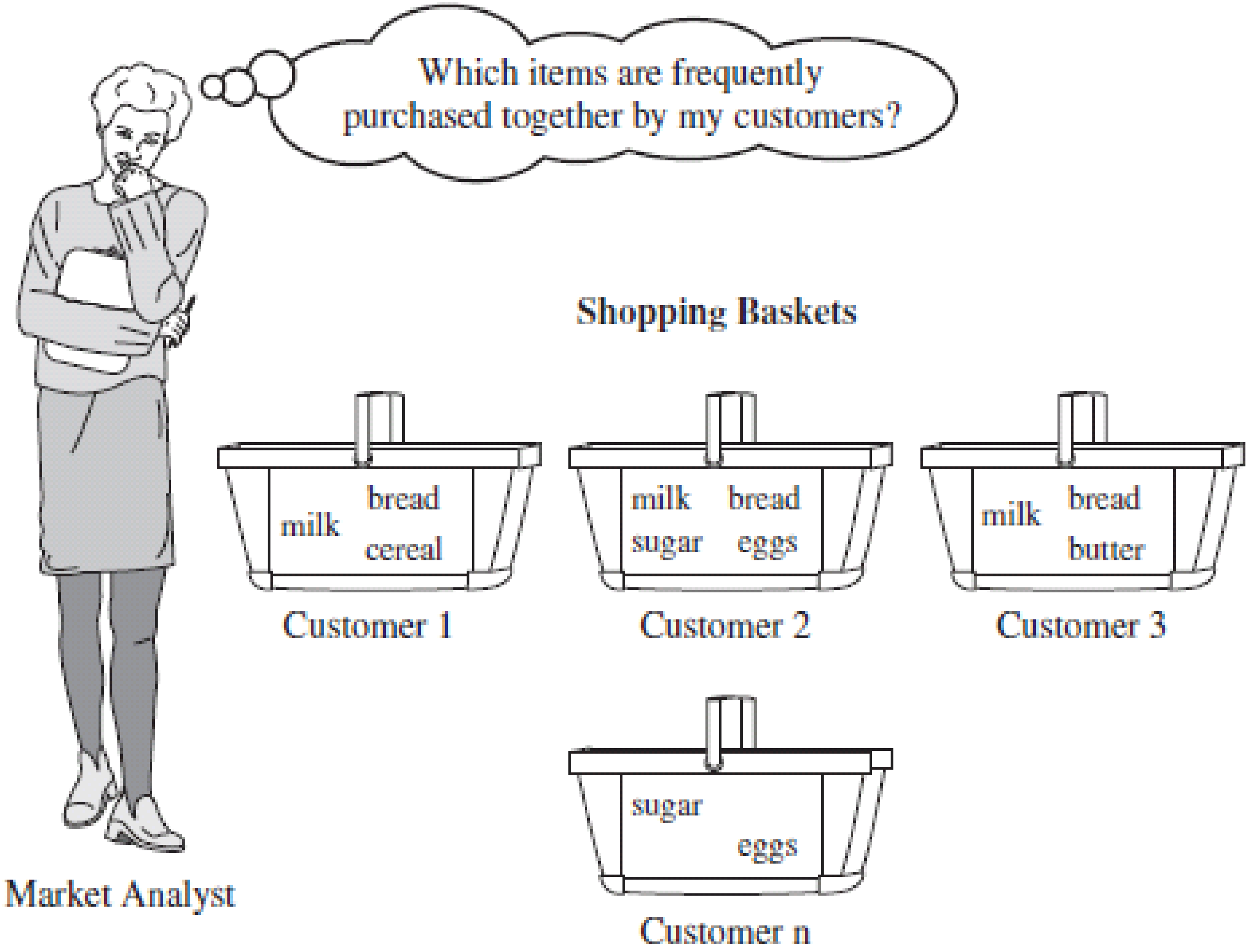
Reference



Other names of Apriori Algorithm

- **Also called:**
  - ✓ **Finding Frequent Item-set Using Candidate Generation**
  - ✓ **Basket Problem (Data)**

Market Basket Analysis: A Motivating Example



Points

$\mathbf{I} = \{i_1, i_2, \ldots., i_m\}$ *set of Items.*

$\mathbf{D} = \{T_1, T_2, \ldots., T_i\}$ *set of Transactions.*

$\mathbf{T} = \{i_k, i_j, \ldots.\}$, *where* $\mathbf{T} \subseteq \mathbf{I}$.

TID : is a unique identifier associated with each **T**

|D| is # of transactions in D

- **Association Rule of the form:** $\mathbf{X} \Longrightarrow \mathbf{Y}$

, *where*

$\mathbf{X} \subset \mathbf{I}$

$\mathbf{Y} \subset \mathbf{I}$

$\mathbf{X} \cap \mathbf{Y} = \emptyset$

Support & Confidence measures

- **Given the Association Rule AR: $X \Longrightarrow Y$**

✓ **Support (S) of the AR :**

$$s = \frac{Support\_count(X \cup Y)}{|D|}$$

✓ **Confidence (C) of the AR :**

$$C = \frac{Support\_count(X \cup Y)}{Support\_count(X)}$$

- **min-conf & min-sup** : are user thresholds (i.e. KB)

Generating the Association Rule (Mining Task)

- The **AR**: $\mathbf{X} \Rightarrow \mathbf{Y}$ is generated if its:

  ✓ Support (S) $\geq$ **min_sup** threshold

  ✓ Confidence (C) $\geq$ **min_conf** threshold

Apriori algorithm

- Apriori uses a “**bottom-up**” approach, where frequent subsets are extended one item at a time (a step known as **candidate generation**).

- The algorithm **terminates** when no further successful extensions are found.

- Apriori uses **breadth-first search** and a **hash tree structure** to **count candidate item sets efficiently**.

The Main Steps of Apriori Algorithm

- The Apriori algorithm is based on the following two main steps:

1. **Join ⋈**
2. **Prune step**

1. The Join ($\bowtie$) step

- **Given two lists, the operation $L1 \bowtie L2$ will be executed if the following two conditions are satisfied :**
  - **All the elements of the two lists are equal except the last elements**
  - **(Last element of L1) < (Last element of L2)**

✓ **For example :** $L1 = \{1, 2, 3\}$

$L2 = \{1, 2, 4\}$

So, $L1 \bowtie L2 = \{1, 2, 3, 4\}$

Example 1 of Join (⋈) step

- Suppose we have the following set of item Lists

L

{ I1 , I2 }
{ I1, I3 }
{ I2, I3 }

L ⋈ L

	Result
{I1 , I2} ⋈ {I1, I3}	= {I1 , I2, I3}
{I1 , I2} ⋈ {I2, I3}	no join
{I1 , I3} ⋈ {I2, I3}	no join

Example 2 of Join (⋈) step

- **Suspose $L1 = \{I1\}$ and**
  $L2 = \{I2\}$

➢ **In this case we will consider**
$L1 = \{\emptyset , I1\}$ and
$L2 = \{\emptyset , I2\}$

➢ **So, $L1 \bowtie L2 = \{I1, I2\}$**

2. The Prune step

- Let L_k is a list of items
- $L_k \bowtie L_k = C_{k+1}$
- If $\{I_i, I_j, \ldots\ldots, I_r\}$ of $L_k \bowtie L_k \notin L_k$ Then this set of elements will be removed

✓ **Example :**

L_2

{ I1 , I2 }
{ I1 , I3 }

$\xrightarrow{L_2 \bowtie L_2}$

C_3

$$C_3 = \{I1, I2, I3\} \left\{ \begin{array}{l} \{I1, I2\} \in L_2 \\ \{I1, I3\} \in L_2 \\ \{I2, I3\} \notin L_2 \end{array} \right.$$

Notation

L_{k+1} : is generated from C_{k+1} after pruning and all itemes are >= min-sup

L_1

{I1}
{I2}
{I3}

$L_1 \bowtie L_1$

C_2

{I1, I2}
{I1, I3}
{I2, I3}

L_2

{I1, I2}
{I1, I3}

Suppose this itemset has sup_count=1 and the min_sup=2, so it will be removed

Low

$$\#\text{ of elements in } C_2 = L_1 \bowtie L_1 = \binom{|L_1|}{2}$$

where,

$$\binom{n}{k} = \frac{n!}{(n-k)!\ k!}$$

- **Example:**

L_1

{I1}
{I2}
{I3}
{I4}

$L_1 \bowtie L_1$

C_2

{I1, I2}
{I1, I3}
{I1, I4}
{I2, I3}
{I2, I4}
{I3, I4}

- # of elements in C_2 $= \binom{4}{2} = \frac{4!}{(2)!\ 2!} = 6$

Example : Mining AR by Apriori

- Consider the following Transaction dataset

  - Let the **min-conf. = 70%**
  - Let the **min-sup. = 2**

TID	Items
T001	{ I1 , I2, I5 }
T002	{ I2, I4 }
T003	{ I2, I3 }
T004	{ I1, I2, I4 }
T005	{ I1, I3 }
T006	{ I2, I3 }
T007	{ I1, I3 }
T008	{ I1, I2, I3, I5 }
T009	{ I1, I2, I3 }

Example : Mining AR by Apriori

TID	Items
T001	{ I1 , I2, I5 }
T002	{ I2, I4 }
T003	{ I2, I3 }
T004	{ I1, I2, I4 }
T005	{ I1, I3 }
T006	{ I2, I3 }
T007	{ I1, I3 }
T008	{ I1, I2, I3, I5 }
T009	{ I1, I2, I3 }

Scan D for count →

ItemSet	Sup-Count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

Compare sup-count with min-sup →

ItemSet	Sup-Count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

Example : Mining AR by Apriori

L_1

ItemSet	Sup-Count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

$L_1 \bowtie L_1$

C_2

Items	count
{I1 , I2}	4
{I1 , I3}	4
{I1 , I4}	1
{I1 , I5}	2
{I2 , I3}	4
{I2 , I4}	2
{I2 , I5}	2
{I3 , I4}	0
{I3 , I5}	1
{I4 , I5}	0

Compare sup-count with min-sup

L_2

Items	count
{I1 , I2}	4
{I1 , I3}	4
{I1 , I5}	2
{I2 , I3}	4
{I2 , I4}	2
{I2 , I5}	2

Example : Mining AR by Apriori

L_2

Items	count
{I1 , I2}	4
{I1 , I3}	4
{I1 , I5}	2
{I2 , I3}	4
{I2 , I4}	2
{I2 , I5}	2

$L_2 \bowtie L_2$

C_3

Items
{I1, I2, I3}
{I1, I2, I5}
{I1, I3, I5}
{I2, I3, I4}
{I2, I3, I5}
{I2, I4, I5}

Pruning step

Removed bcz {I3, I5} $\notin L_2$

Removed bcz {I3, I4} $\notin L_2$

Removed bcz {I3, I5} $\notin L_2$

Removed bcz {I4, I5} $\notin L_2$

C_3

Items	Sup.count
{I1, I2, I3}	2
{I1, I2, I5}	2

Compare sup-count with min-sup

L_3

Items	Sup.count
{I1, I2, I3}	2
{I1, I2, I5}	2

Example : Mining AR by Apriori

L_3

Items	Sup.count
{I1, I2, I3}	2
{I1, I2, I5}	2

$L_3 \bowtie L_3$

C_4

Items
{I1, I2, I3, I5}

Pruning step

{I2, I3, I5} $\notin L_3$

So, $C_4 = \emptyset$

So, the Frequent item sets are in L_3

Items	Sup.count
{I1, I2, I3}	2
{I1, I2, I5}	2

- **Discovering ARs From L_3** **(√) means Interesting, (X) means Rejected**

Items
{I1, I2, I3}
{I1, I2, I5}

From {I1, I2, I3}:

Rule1: I1 ^ I2 ⟹ I3 confidence = 2/4 = 50% (X)

Rule2: I1 ^ I3 ⟹ I2 confidence = 2/4 = 50% (X)

Rule3: I2 ^ I3 ⟹ I1 confidence = 2/4 = 50% (X)

Rule4: I1 ⟹ I2 ^ I3 confidence = 2/6 = 33% (X)

Rule5: I2 ⟹ I1 ^ I3 confidence = 2/7 = 29% (X)

Rule6: I3 ⟹ I1 ^ I2 confidence = 2/6 = 33% (X)

From {I1, I2, I5}:

Rule7: I1 ^ I2 ⟹ I5 confid. = 2/4 = 50% (X)

Rule8: I1 ^ I5 ⟹ I2 confid. = 2/2 = 100% (√)

Rule9: I2 ^ I5 ⟹ I1 confid. = 2/2 = 100% (√)

Rule10: I1 ⟹ I2 ^ I5 confid. = 2/6 = 33% (X)

Rule11: I2 ⟹ I1 ^ I5 confid. = 2/7 = 29% (X)

Rule12: I5 ⟹ I1 ^ I2 confid. = 2/2 = 100% (√)

Page: 239

Apriori Algorithm

Algorithm: Apriori. Find frequent itemsets using an iterative level-wise approach based on candidate generation.

Input:

- D, a database of transactions;
- min_sup, the minimum support count threshold.

Output: L, frequent itemsets in D.

Method:

L_1 = find_frequent_1-itemsets(D);
for ($k = 2; L_{k-1} \neq \phi; k{+}{+}$) {
    C_k = apriori_gen(L_{k-1});
    for each transaction $t \in D$ { // scan D for counts
        C_t = subset(C_k, t); // get the subsets of t that are candidates
        for each candidate $c \in C_t$
            c.count++;
    }
    $L_k = \{c \in C_k \mid c.count \geq min\_sup\}$
}
return $L = \cup_k L_k$;

Page: 239

Apriori Algorithm

```
procedure apriori_gen(L_{k-1}:frequent (k - 1)-itemsets)
    for each itemset l_1 ∈ L_{k-1}
        for each itemset l_2 ∈ L_{k-1}
            if (l_1[1] = l_2[1]) ∧ (l_1[2] = l_2[2]) ∧ ... ∧ (l_1[k-2] = l_2[k-2]) ∧ (l_1[k-1] < l_2[k-1]) then {
                c = l_1 ⋈ l_2; // join step: generate candidates
                if has_infrequent_subset(c, L_{k-1}) then
                    delete c; // prune step: remove unfruitful candidate
                else add c to C_k;
            }
    return C_k;
```

```
procedure has_infrequent_subset(c: candidate k-itemset;
            L_{k-1}: frequent (k - 1)-itemsets); // use prior knowledge
    for each (k - 1)-subset s of c
        if s ∉ L_{k-1} then
            return TRUE;
    return FALSE;
```

The End…

$$\sum^{Prof.} \sqrt{\frac{Fadl}{Ba - Alwi}}$$