



# **Part 15:**

# **Data Mining**

**Database System Concepts, 7<sup>th</sup> Ed.**

©Silberschatz, Korth and Sudarshan

See [www.db-book.com](http://www.db-book.com) for conditions on re-use



# Data Mining

- **Data mining** is the process of analyzing large databases to find useful patterns
  - Similar goals to machine learning, but on very large volumes of data
- Also called **knowledge discovery in databases (KDD)**
- More generally, knowledge is discovered by applying machine learning techniques on past instances of data, to form a **model**
  - Model is then used to make predictions for new instances

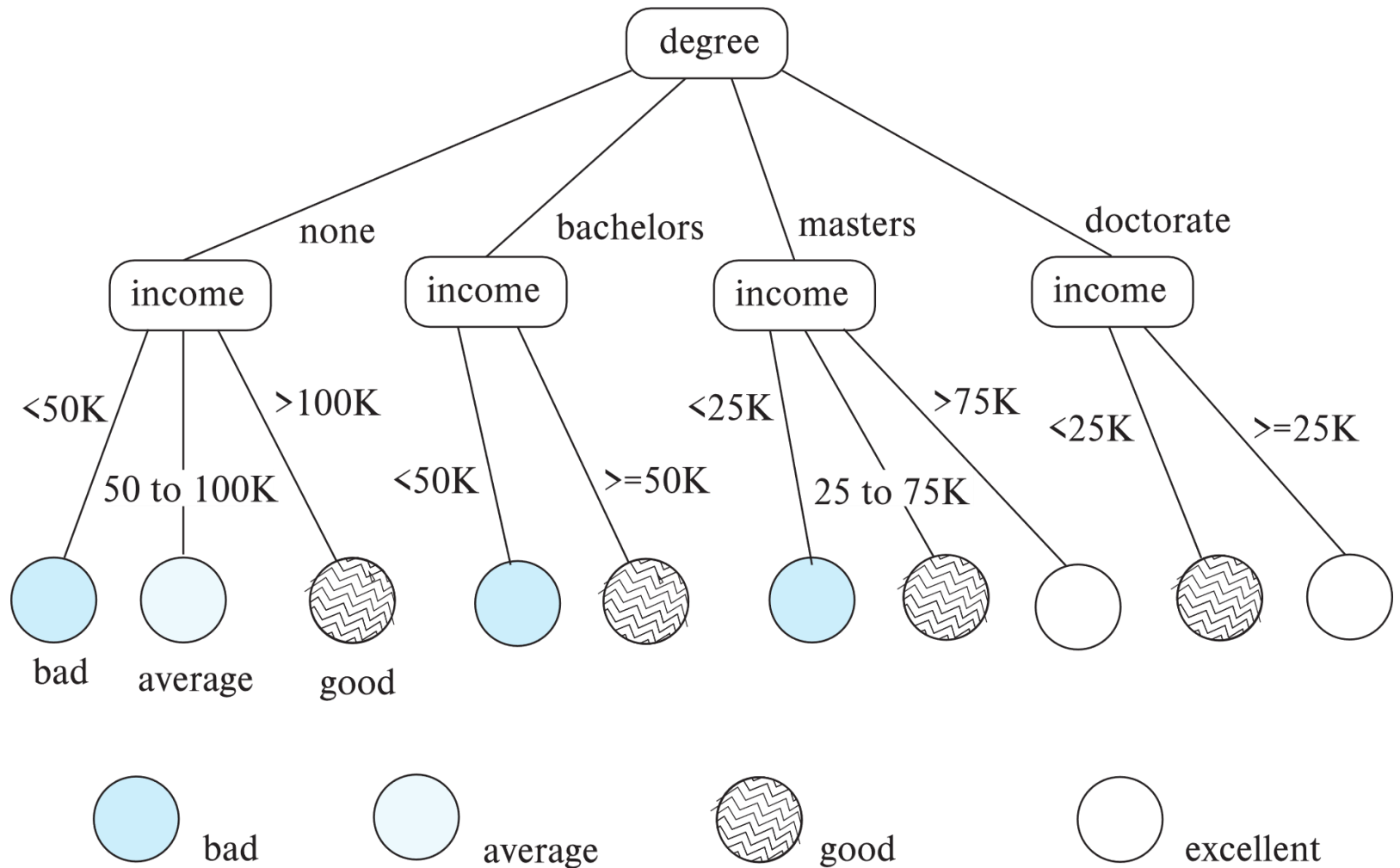


# Types of Data Mining Tasks

- Examples of data mining tasks:
  - **Classification**
    - Items (with associated attributes) belong to one of several classes
    - **Training instances** have attribute values and classes provided
    - Given a new item whose class is unknown, predict to which class it belongs based on its attribute values
  - **Association**
    - Find books that are often bought by “similar” customers. If a new such customer buys one such book, suggest the others too.
    - Associations may be used as a first step in detecting **causation**
      - E.g., association between exposure to chemical X and cancer
  - **Clustering**
    - E.g., typhoid cases were clustered in an area surrounding a contaminated well
    - Detection of clusters remains important in detecting epidemics



# Decision Tree Classifiers





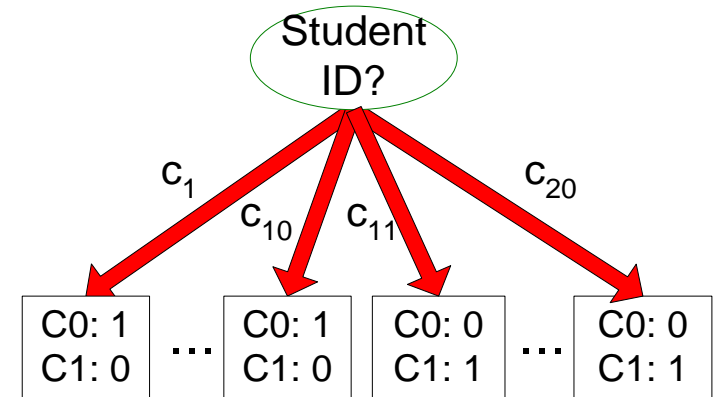
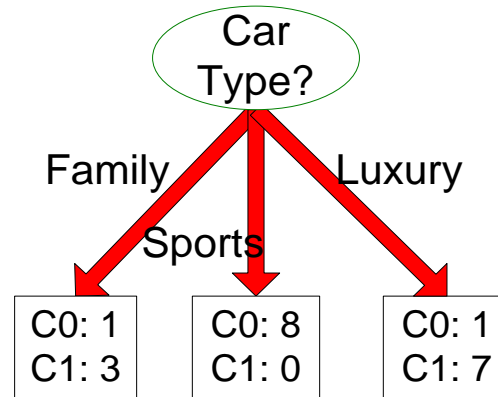
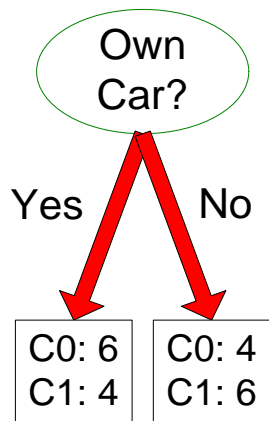
# Decision Trees

- Each internal node of the tree partitions the data into groups based on a **partitioning or splitting attribute**, and a **partitioning condition** for the node
- Node Purity:
  - all (or most) of the items at the node belong to the same class
- Traverse tree from top to make a prediction



# Splitting

Before Splitting: 10 records of class 0 (C0),  
10 records of class 1 (C1)



Which is the best for predicting credit worthiness?



# How to Determine a Good Split

- Greedy approach:
  - Nodes with **homogeneous** or **pure** class distribution are preferred
- Need a measure of node impurity:

C0: 5 C1: 5
----------------

Non-homogeneous  
High degree of impurity

C0: 9 C1: 1
----------------

Homogeneous  
Low degree of impurity



# Measures of Node Impurity

- Gini Index

$$Gini\ Index = 1 - \sum_{i=0}^{c-1} p_i(t)^2$$

- Entropy

$$Entropy = - \sum_{i=0}^{c-1} p_i(t) \log_2 p_i(t)$$

- Misclassification error

$$Misclassification\ error = 1 - \max[p_i(t)]$$

Where  $p_i(t)$  is the relative frequency of class  $i$  at node  $t$ , and  $c$  is the total number of classes.

**We want a low value for these measures**





# Measure of Impurity: GINI

- Gini Index for a given node  $t$ ,

$$\text{Gini Index} = 1 - \sum_{i=0}^{c-1} p_i(t)^2$$

where  $p_i(t)$  is the relative frequency of class  $i$  at node  $t$ , and  $c$  is the total number of classes

- For a 2-class problem  $(p, q)$ , where  $p$  represents the relative frequency of Class 1, and  $q = (1-p)$ 
  - $\text{GINI} = 1 - p^2 - (1 - p)^2 = 2p(1-p) = 2pq$
- Maximum  $(1 - 1/c)$  when records are equally distributed among all classes, implying least beneficial situation for classification
- Minimum (0) when all records belong to one class, implying most beneficial situation for classification

C1	<b>0</b>
C2	<b>6</b>
<b>Gini=0.000</b>	

C1	<b>1</b>
C2	<b>5</b>
<b>Gini=0.278</b>	

C1	<b>2</b>
C2	<b>4</b>
<b>Gini=0.444</b>	

C1	<b>3</b>
C2	<b>3</b>
<b>Gini=0.500</b>	



# Computing Gini Index of a Single Node

$$\text{Gini Index} = 1 - \sum_{i=0}^{c-1} p_i(t)^2$$

C1	<b>0</b>
C2	<b>6</b>

$$P(C1) = 0/6 = 0 \quad P(C2) = 6/6 = 1$$

$$\text{Gini} = 1 - P(C1)^2 - P(C2)^2 = 1 - 0 - 1 = 0$$

C1	<b>1</b>
C2	<b>5</b>

$$P(C1) = 1/6 \quad P(C2) = 5/6$$

$$\text{Gini} = 1 - (1/6)^2 - (5/6)^2 = 0.278$$

C1	<b>2</b>
C2	<b>4</b>

$$P(C1) = 2/6 \quad P(C2) = 4/6$$

$$\text{Gini} = 1 - (2/6)^2 - (4/6)^2 = 0.444$$



# Splitting Based on GINI

- When a node  $p$  is split into  $k$  partitions (children), the quality of split is computed as,

$$GINI_{split} = \sum_{i=1}^k \frac{n_i}{n} GINI(i)$$

where,  $n_i$  = number of records at child  $i$ ,  
 $n$  = number of records at node  $p$ .

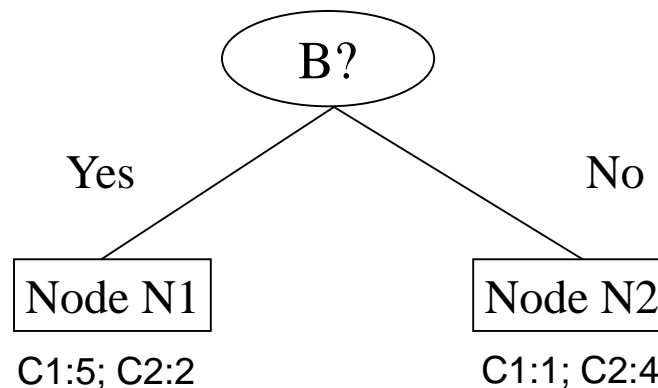


# Computing GINI Index

- Splits into two partitions
- Effect of Weighing partitions:
  - Larger and Purer Partitions are sought for – reduction in GINI

$$\begin{aligned}\text{Gini}(N1) &= 1 - (5/7)^2 - (2/7)^2 \\ &= 1 - 0.51 - 0.082 \\ &= 0.408\end{aligned}$$

$$\begin{aligned}\text{Gini}(N2) &= 1 - (1/5)^2 - (4/5)^2 \\ &= 1 - 0.04 - 0.64 \\ &= 0.32\end{aligned}$$



	N1	N2
C1	5	1
C2	2	4
<b>Gini=0.371</b>		

	Parent
C1	6
C2	6
<b>Gini = 0.500</b>	

$$\begin{aligned}\text{Gini (Children)} &= 7/12 * 0.408 + 5/12 * 0.32 \\ &= 0.238 + 0.133 \\ &= 0.371 \text{ (reduced from 0.5)}\end{aligned}$$



# Splitting Criteria Based on Entropy

- Entropy at a given node  $t$ ,

$$Entropy = - \sum_{i=0}^{c-1} p_i(t) \log_2 p_i(t)$$

where  $p_i(t)$  is the relative frequency of class  $i$  at node  $t$ , and  $c$  is the total number of classes

- Measures lack of homogeneity (or disorderliness) of a node
  - Maximum ( $\log_2 c$ ) when records are equally distributed among all classes implying the least beneficial situation for classification
  - Minimum (0) when all records belong to one class, implying the most beneficial situation for classification
- From  $\log_b a * \log_a x = \log_b x$ , we have  $\log_2 x = \log_2 e * \log_e x = 1.44 \log_e x$
- Entropy based computations are similar to the GINI index computations



# Computing Entropy

$$Entropy = - \sum_{i=0}^{c-1} p_i(t) \log_2 p_i(t)$$

**Note that  $\log_2 x = \log_2 e * \log_e x = 1.44 \log_e x$**

C1	<b>0</b>
C2	<b>6</b>

$$P(C1) = 0/6 = 0 \quad P(C2) = 6/6 = 1$$

$$Entropy = - 0 \log_2 0 - 1 \log_2 1 = - 0 - 0 = 0$$

C1	<b>1</b>
C2	<b>5</b>

$$P(C1) = 1/6 \quad P(C2) = 5/6$$

$$Entropy = - (1/6) \log_2 (1/6) - (5/6) \log_2 (5/6) = 0.65$$

C1	<b>2</b>
C2	<b>4</b>

$$P(C1) = 2/6 \quad P(C2) = 4/6$$

$$Entropy = - (2/6) \log_2 (2/6) - (4/6) \log_2 (4/6) = 0.92$$



# Measure of Impurity: Misclassification Error

- Classification error at a node  $t$

$$Error(t) = 1 - \max_i [p_i(t)]$$

where  $p_i(t)$  is the relative frequency of class  $i$  at node  $t$ , and the maximum is taken over all classes  $c$

- We want  $\max_i [p_i(t)]$  to be large so that  $1 - \max_i [p_i(t)]$  is small
- Maximum of  $(1 - 1/c)$  when records are equally distributed among all classes, implying the least beneficial situation for classification
- Minimum of 0 when all records belong to one class, implying the most beneficial situation for classification



# Computing Error of a Single Node

$$Error(t) = 1 - \max_i [p_i(t)]$$

C1	<b>0</b>
C2	<b>6</b>

$$P(C1) = 0/6 = 0 \quad P(C2) = 6/6 = 1$$

$$Error = 1 - \max(0, 1) = 1 - 1 = 0$$

C1	<b>1</b>
C2	<b>5</b>

$$P(C1) = 1/6 \quad P(C2) = 5/6$$

$$Error = 1 - \max(1/6, 5/6) = 1 - 5/6 = 1/6$$

C1	<b>2</b>
C2	<b>4</b>

$$P(C1) = 2/6 \quad P(C2) = 4/6$$

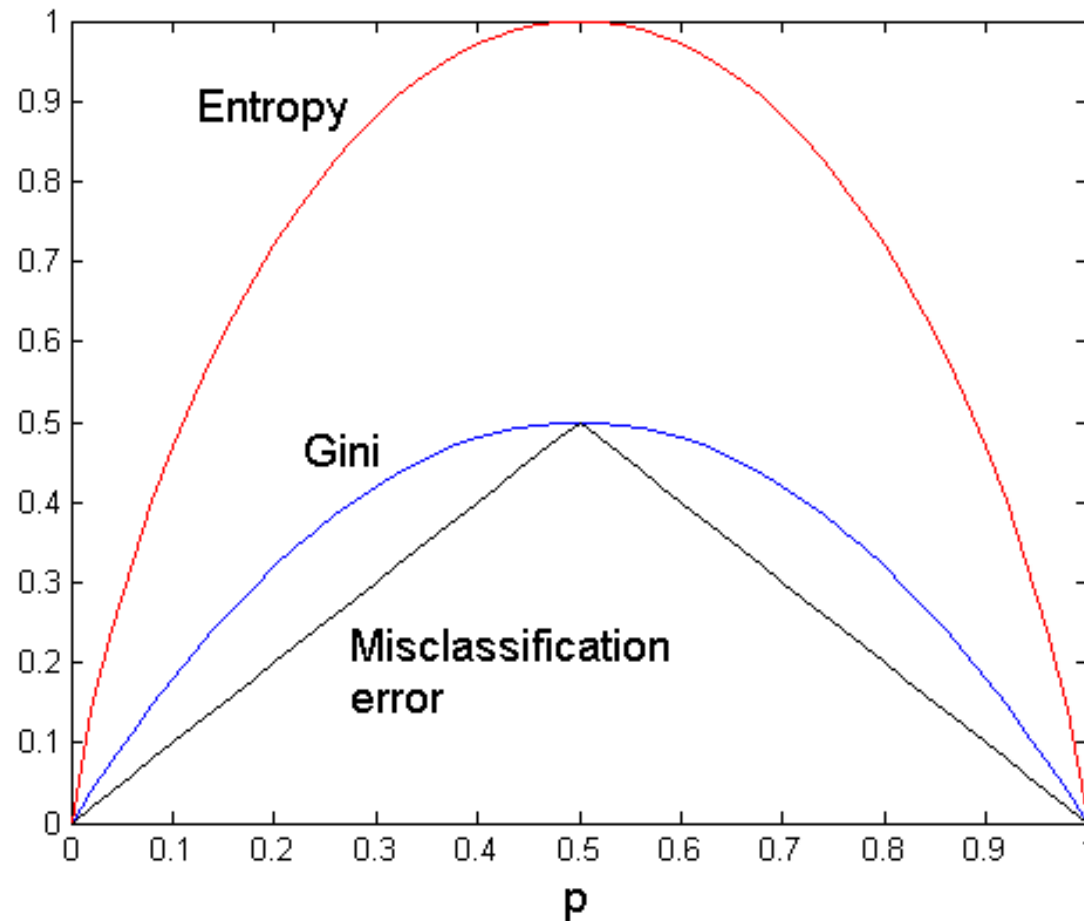
$$Error = 1 - \max(2/6, 4/6) = 1 - 4/6 = 1/3$$





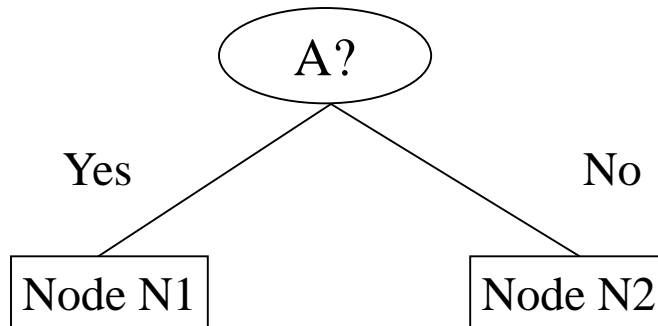
# Comparison among Impurity Measures

For a 2-class problem, with  $p$  giving the relative frequency of Class 1





# Misclassification Error vs Gini Index



	Parent
C1	<b>7</b>
C2	<b>3</b>
<b>Gini = 0.42</b>	

$$\begin{aligned}
 &\text{Gini}(N1) \\
 &= 1 - (3/3)^2 - (0/3)^2 \\
 &= 0
 \end{aligned}$$

	<b>N1</b>	<b>N2</b>
C1	<b>3</b>	<b>4</b>
C2	<b>0</b>	<b>3</b>
<b>Gini=0.342</b>		

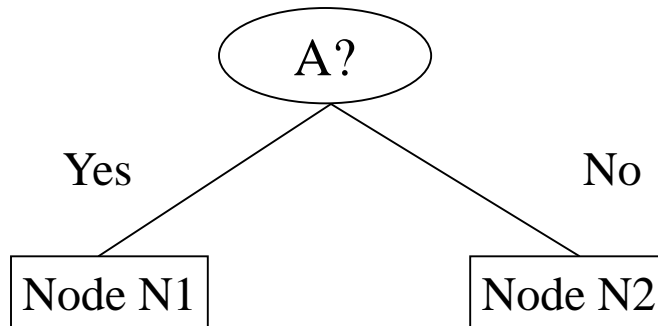
$$\begin{aligned}
 &\text{Gini}(N2) \\
 &= 1 - (4/7)^2 - (3/7)^2 \\
 &= 0.489
 \end{aligned}$$

$$\begin{aligned}
 &\text{Gini(Children)} \\
 &= 3/10 * 0 \\
 &+ 7/10 * 0.489 \\
 &= 0.342
 \end{aligned}$$

Gini improves but  
what about the error?



# Misclassification Error vs Gini Index



	Parent
C1	7
C2	3

**Error = 0.3**

	N1	N2
C1	3	4
C2	0	3

**Error = 0.3**

$$\begin{aligned}
 \text{Error}(N1) &= 1 - \max[(3/3), (0/3)] \\
 &= 0
 \end{aligned}$$

$$\begin{aligned}
 \text{Error}(N2) &= 1 - \max[(4/7), (3/7)] \\
 &= 3/7
 \end{aligned}$$

$$\begin{aligned}
 \text{Error}(\text{Parent}) &= 1 - \max[(7/10), (3/10)] = \\
 &= 0.3
 \end{aligned}$$

$$\begin{aligned}
 \text{Error}(\text{Children}) &= 3/10 * 0 + 7/10 * 3/7 \\
 &= 0.3
 \end{aligned}$$

**Gini improves but error remains the same!!**



# Association Rules

- Given a set of transactions, find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction
- Retail shops are often interested in associations between different items that people buy.
  - Someone who buys bread is quite likely also to buy milk
  - A person who bought the book *Database System Concepts* is quite likely also to buy the book *Operating System Concepts*
- Associations information can be used in several ways.
  - E.g., when a customer buys a particular book, an online shop may suggest associated books
- **Association rules:**
  - bread* → *milk*      *DB-Concepts, OS-Concepts* → *Networks*
  - Left hand side: **antecedent**,    right hand side: **consequent**
  - An association rule must have an associated **population**; the population consists of a set of **instances**
    - E.g., each transaction (sale) at a shop is an instance, and the set of all transactions is the population



# Frequent Item Set

## Itemset

- A collection of one or more items
  - ▶ Example: {Milk}, {Milk, Bread, Diaper}
- **k-itemset**
  - ▶ An itemset that contains  $k$  items

## Support count ( $\sigma$ ) or absolute support

- Frequency of occurrence of an itemset
- E.g.  $\sigma(\{\text{Milk, Bread, Diaper}\}) = 2$

## Support (s) or relative support

- Fraction of transactions that contain an itemset
- $s = \sigma / |T|$ , where  $|T|$  is the number of transactions
- E.g.  $s(\{\text{Bread, Milk, Diaper}\}) = 2/5$

## Frequent (or Large) Itemset

- An itemset whose support is greater than or equal to a **minsup** threshold, where **minsup** is a given minimum support

<i>TID</i>	<i>Items</i>
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

<i>TID</i>	<i>Items</i>
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	<b>Bread, Milk, Diaper, Beer</b>
5	<b>Bread, Milk, Diaper, Coke</b>



# Association Rules

## Association Rule

- An implication expression of the form  $X \rightarrow Y$ , where  $X$  and  $Y$  are itemsets
- Arrow means co-occurrence, not causality
- Example:  
 $\{\text{Milk, Diaper}\} \rightarrow \{\text{Beer}\}$

<i>TID</i>	<i>Items</i>
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

## Rule Evaluation Metrics

- Support ( $s$ ) of an association rule
  - ◆ Fraction of transactions that contain both  $X$  and  $Y$
- Confidence ( $c$ ) of an association rule
  - ◆ Measures how often items in  $Y$  appear in transactions that contain  $X$

Example:

$\{\text{Milk, Diaper}\} \rightarrow \{\text{Beer}\}$

$$s = \frac{\sigma(\text{Milk, Diaper, Beer})}{|T|} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(\text{Milk, Diaper, Beer})}{\sigma(\text{Milk, Diaper})} = \frac{2}{3} = 0.67$$



# Mining Association Rules

<i>TID</i>	<i>Items</i>
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example of Rules:

$\{\text{Milk, Diaper}\} \rightarrow \{\text{Beer}\}$  ( $s=0.4$ ,  $c=0.67$ )

$\{\text{Milk, Beer}\} \rightarrow \{\text{Diaper}\}$  ( $s=0.4$ ,  $c=1.0$ )

$\{\text{Diaper, Beer}\} \rightarrow \{\text{Milk}\}$  ( $s=0.4$ ,  $c=0.67$ )

$\{\text{Beer}\} \rightarrow \{\text{Milk, Diaper}\}$  ( $s=0.4$ ,  $c=0.67$ )

$\{\text{Diaper}\} \rightarrow \{\text{Milk, Beer}\}$  ( $s=0.4$ ,  $c=0.5$ )

$\{\text{Milk}\} \rightarrow \{\text{Diaper, Beer}\}$  ( $s=0.4$ ,  $c=0.5$ )

- All the above rules are binary partitions of the same itemset:  
 $\{\text{Milk, Diaper, Beer}\}$
- Rules originating from the same itemset have identical support but can have different confidence



# Reducing the Search Space

- The itemsets that have a support that exceeds the threshold minimum are called *large* (or *frequent*) itemsets, where large here means large support
- Discovering all large itemsets together with the value for the support is a major problem if the cardinality of the set of items is very high
- A typical supermarket has thousands of items
  - The number of distinct (non-empty) itemsets is  $2^m$ , where  $m$  is number of items, and counting support for all possible itemsets becomes computation-intensive.
- To reduce the combinatorial search space, algorithms for finding association rules utilize two properties





# Reducing the Search Space

## ■ **Apriori Principle:**

- If an itemset is frequent, then all of its subsets must also be frequent
- Support of an itemset never exceeds the support of its subsets

$$\forall X, Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$$

## ■ *Downward Closure*

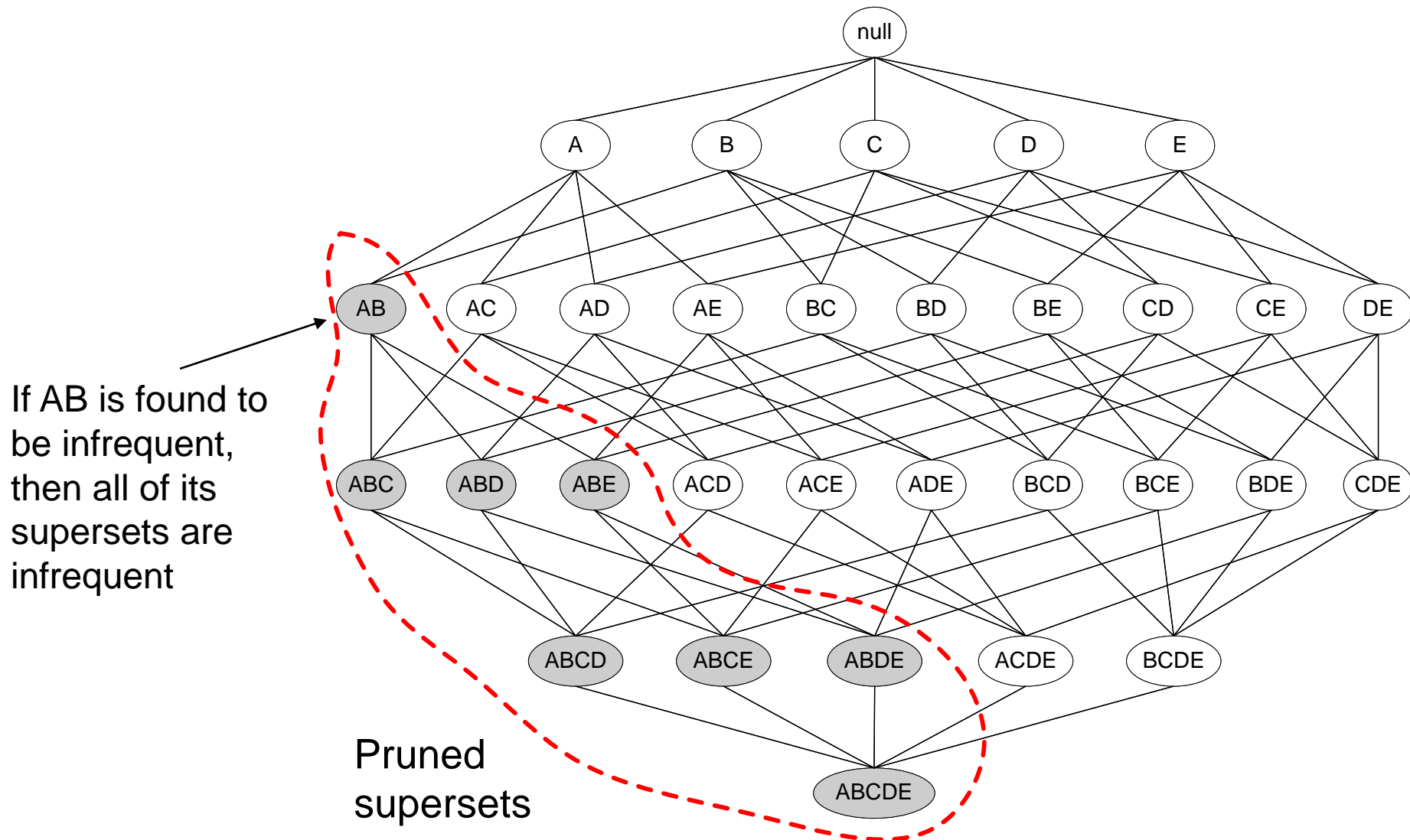
- A subset of a large itemset must also be large (that is, each subset of a large itemset exceeds the minimum required support).

## ■ *Antimonotonicity*

- Conversely, a superset of a small itemset is also small (implying that it does not have enough support)
- Thus, once an itemset is found to be small (not a large itemset), then any extension to that itemset, formed by adding one or more items to the set, will also yield a small itemset



# Reducing the Search Space





# Candidate Generation: $F_{k-1} \times F_{k-1}$ Method

- Merge two frequent  $(k-1)$ -itemsets if their first  $(k-2)$  items are identical
  
- $F_3 = \{ABC, ABD, ABE, ACD, BCD, BDE, CDE\}$ 
  - Merge(ABC, ABD) = ABCD
  - Merge(ABC, ABE) = ABCE
  - Merge(ABD, ABE) = ABDE
  
  - Do not merge(ABD, ACD) because they share only prefix of length 1 instead of length 2



# Illustrating the Apriori Principle

Item	Count
Bread	4
<b>Coke</b>	<b>2</b>
Milk	4
Beer	3
Diaper	4
<b>Eggs</b>	<b>1</b>

Items (1-itemsets)



Itemset	Count
{Bread, Milk}	3
<b>{Bread, Beer}</b>	<b>2</b>
{Bread, Diaper}	3
<b>{Milk, Beer}</b>	<b>2</b>
{Milk, Diaper}	3
{Beer, Diaper}	3

Pairs (2-itemsets)

(Items are generated systematically from the 1-itemsets table, exhausting all combinations: no need to generate candidates involving Coke or Eggs)



Triplets (3-itemsets)

Itemset	Count
<b>{Bread, Diaper, Milk}</b>	<b>2</b>

Use of  $F_{k-1} \times F_{k-1}$  method for candidate generation results in only one 3-itemset. This is eliminated after the support counting step.

**Note that the ordering of items follows that given in the 1-itemsets table.**

Minimum Support = 3

If every subset is considered,  
 ${}^6C_1 + {}^6C_2 + {}^6C_3$   
 $6 + 15 + 20 = 41$   
 With support-based pruning,  
 $6 + 6 + 1 = 13$



# Illustrating the Apriori Principle

Item	Count
Beer	3
Bread	4
Coke	2
Diaper	4
Eggs	1
Milk	4

Items (1-itemsets)



Itemset	Count
{Beer, Bread}	2
{Beer, Diaper}	3
{Beer, Milk}	2
{Bread, Diaper}	3
{Bread, Milk}	3
{Diaper, Milk}	3

Pairs (2-itemsets)

(Items are generated systematically from the 1-itemsets table, exhausting all combinations: no need to generate candidates involving Coke or Eggs)



Triplets (3-itemsets)

Itemset	Count
{Bread, Diaper, Milk}	2

Minimum Support = 3

If every subset is considered,

$${}^6C_1 + {}^6C_2 + {}^6C_3$$

$$6 + 15 + 20 = 41$$

With support-based pruning,

$$6 + 6 + 1 = 13$$

Here, we follow alphabetical order

TID	Items
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk



# Clustering

- It is often useful to partition data without having training samples
  - this is an instance of *unsupervised learning*
- In business, it may be important to determine groups of customers who have similar buying patterns, or in medicine, it may be important to determine groups of patients who show similar reactions to prescribed drugs.
- The goal of clustering is to place records into groups, such that records in a group are similar to each other and dissimilar to records in other groups, and the groups are usually disjoint.



# Clustering

- An important aspect of clustering is the similarity function used.
- The Euclidean distance is often used to measure similarity:

Consider two  $n$ -dimensional data records as points  $\mathbf{x}$  and  $\mathbf{y}$  in  $n$ -dimensional space. We can consider the value for the  $i$ th dimension as  $x_i$  and  $y_i$  for the two records. The Euclidean distance between points  $\mathbf{x} = (x_1, \dots, x_n)$  and  $\mathbf{y} = (y_1, \dots, y_n)$  in  $n$ -dimensional space is

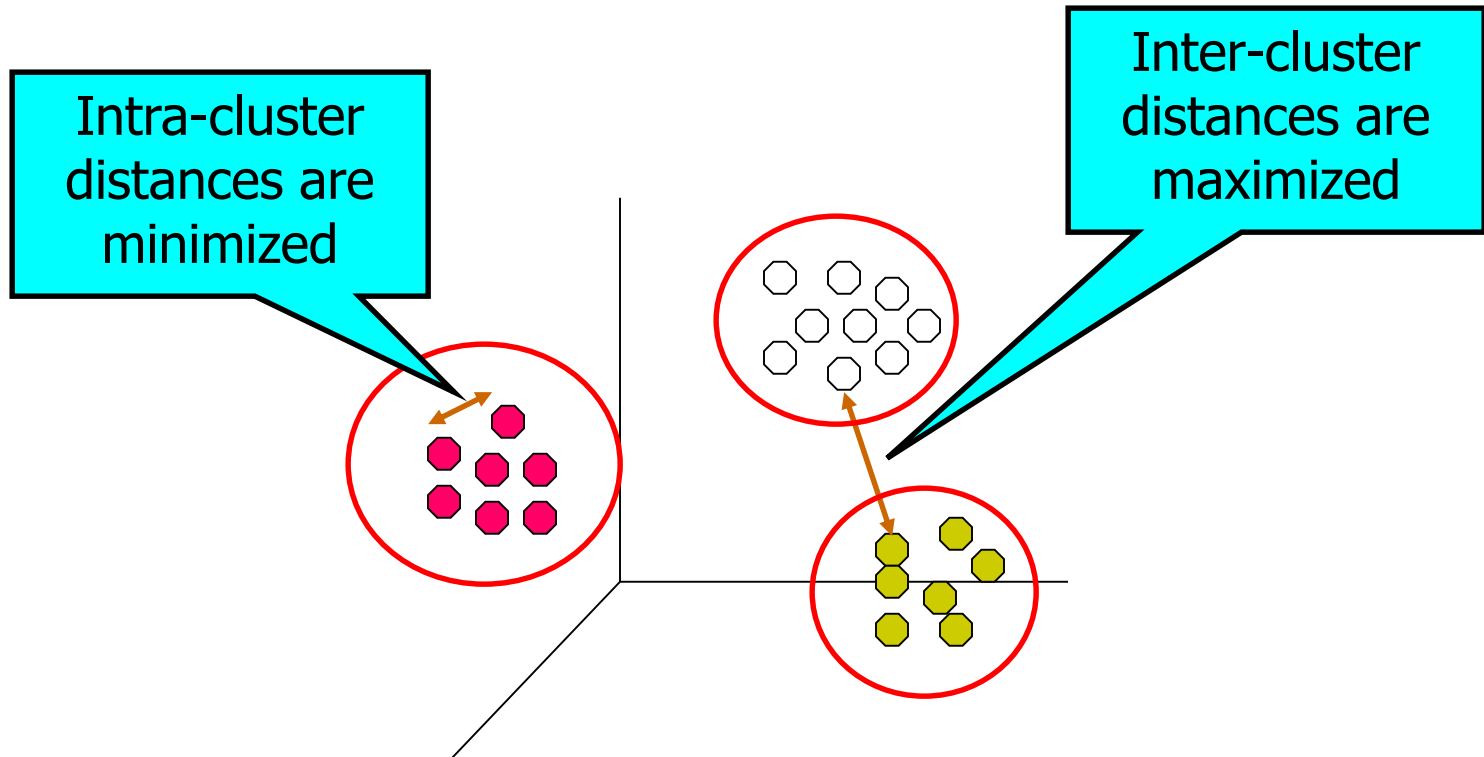
$$\rho(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^n |x_i - y_i|^2}$$

The smaller the distance between two points, the greater is the similarity



# Cluster Analysis

- Finding groups of objects such that the objects in a group will be similar to one another and different from the objects in other groups







# K-Means Clustering

- Each cluster is associated with a **centroid** (center point)
- Each point is assigned to the cluster with the closest centroid
- Number of clusters,  $K$ , must be specified
- The basic algorithm is very simple

- 
- 1: Select  $K$  points as the initial centroids.
  - 2: **repeat**
  - 3:   Form  $K$  clusters by assigning all points to the closest centroid.
  - 4:   Recompute the centroid of each cluster.
  - 5: **until** The centroids don't change
-



# K-Means Clustering

Consider the following table

Record	Age	Years of Service
1	30	5
2	50	25
3	50	15
4	25	5
5	30	10
6	55	25

Assume that the number of desired clusters  $K$  is 2. Let the algorithm choose records with Record 3 for cluster  $C_1$  and Record 6 for cluster  $C_2$  as the initial cluster centroids. The remaining records will be assigned to one of those clusters during the first iteration of the repeat loop.



# K-Means Clustering

Record 1 has a distance from  $C_1$  of  $\sqrt{(20^2 + 10^2)} = 22.4$  and a distance from  $C_2$  of 32.0, so it joins cluster  $C_1$ . Record 2 has a distance from  $C_1$  of 10.0 and a distance from  $C_2$  of 5.0, so it joins cluster  $C_2$  and so on.

Record	Age	Years of Service	Dist from 3	Dist from 6
1	30	5	<u>22.4</u>	32.0
2	50	25	10.0	<u>5.0</u>
<b>3</b>	<b>50</b>	<b>15</b>	0	-
4	25	5	<u>25.5</u>	36.6
5	30	10	<u>20.6</u>	29.2
<b>6</b>	<b>55</b>	<b>25</b>	-	0

Thus we have the clusters

$C_1 = \{\text{Record 1, **Record 3**, Record 4, Record 5}\}$

$C_2 = \{\text{Record 2, **Record 6**}\}.$



# K-Means Clustering

Next, compute the new centroids:

The new centroid for  $C_1$  is  $((30+50+25+30)/4, (5+15+5+10)/4)$   
 $= (33.75, 8.75)$

The new centroid for  $C_2$  is  $((50+55)/2, (25+25)/2) = (52.5, 25)$

In the second iteration the six records are placed into the two clusters as follows:

$C'_1 = \{\text{Record 1, Record 4, Record 5}\}$

$C'_2 = \{\text{Record 2, Record 3, Record 6}\}.$

The mean for  $C'_1$  and  $C'_2$  is re-computed as  $(28.3, 6.7)$  and  $(51.7, 21.7)$ .

In the next iteration, all records stay in their previous clusters and the algorithm terminates.