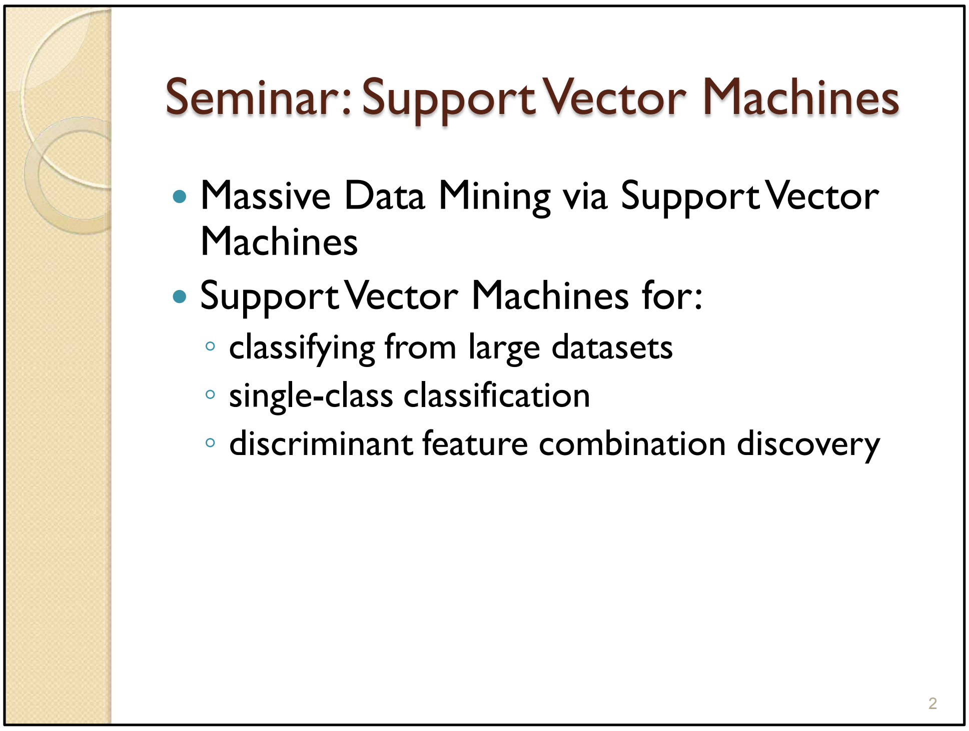
# Midterm: Review

Lecturer: Dr. *Nguyen Thi Ngoc Anh*

*Email: ngocanhnt@ude.edu.vn*

1



## Seminar: Support Vector Machines

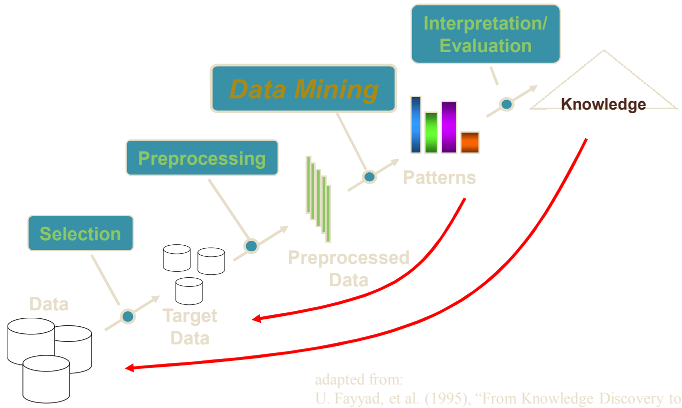
- Massive Data Mining via Support Vector Machines
- Support Vector Machines for:
  - classifying from large datasets
  - single-class classification
  - discriminant feature combination discovery

2

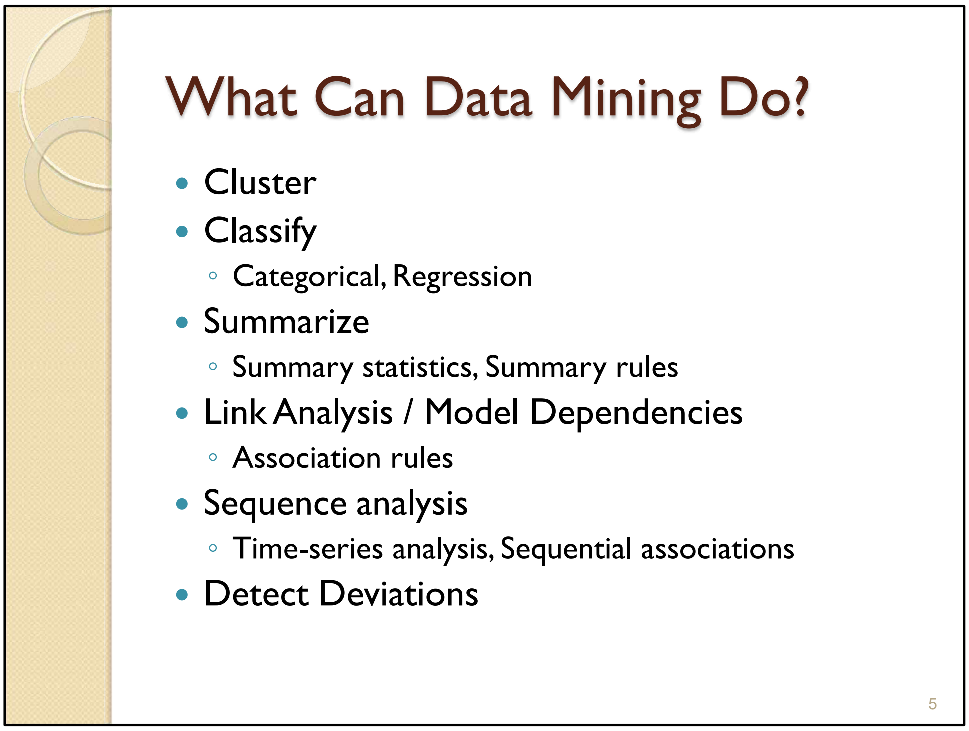
# Data Mining: Classification Schemes

- General functionality
  - Descriptive data mining
  - Predictive data mining
- Different views, different classifications
  - Kinds of data to be mined
  - Kinds of knowledge to be discovered
  - Kinds of techniques utilized
  - Kinds of applications adapted

# Knowledge Discovery in Databases: Process

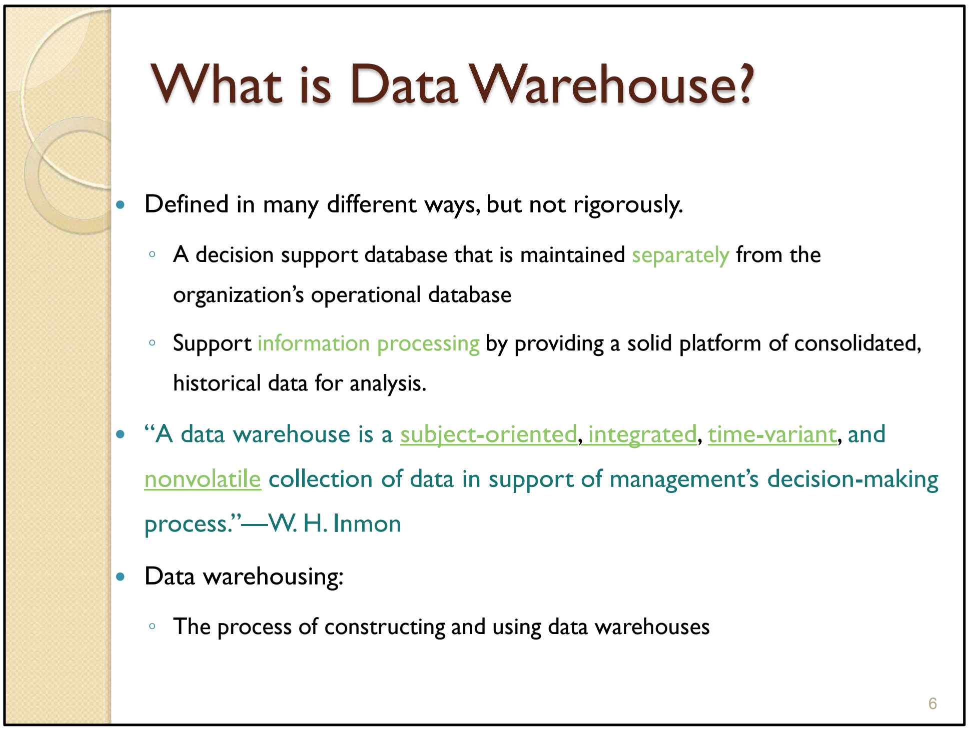


adapted from:
U. Fayyad, et al. (1995), “From Knowledge Discovery to Data Mining: An Overview,” Advances in Knowledge Discovery and Data Mining, U. Fayyad et al. (Eds.), AAAI/MIT Press



# What Can Data Mining Do?

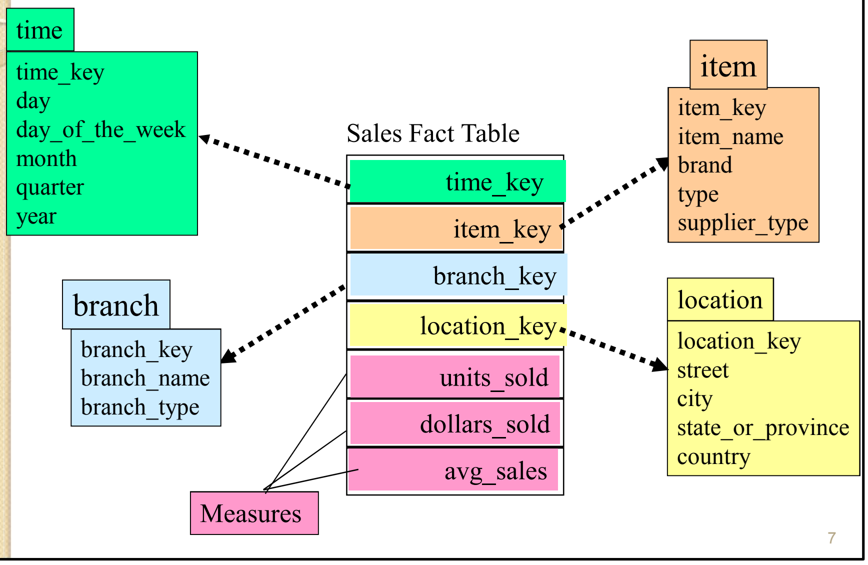
- Cluster
- Classify
  - Categorical, Regression
- Summarize
  - Summary statistics, Summary rules
- Link Analysis / Model Dependencies
  - Association rules
- Sequence analysis
  - Time-series analysis, Sequential associations
- Detect Deviations



# What is Data Warehouse?

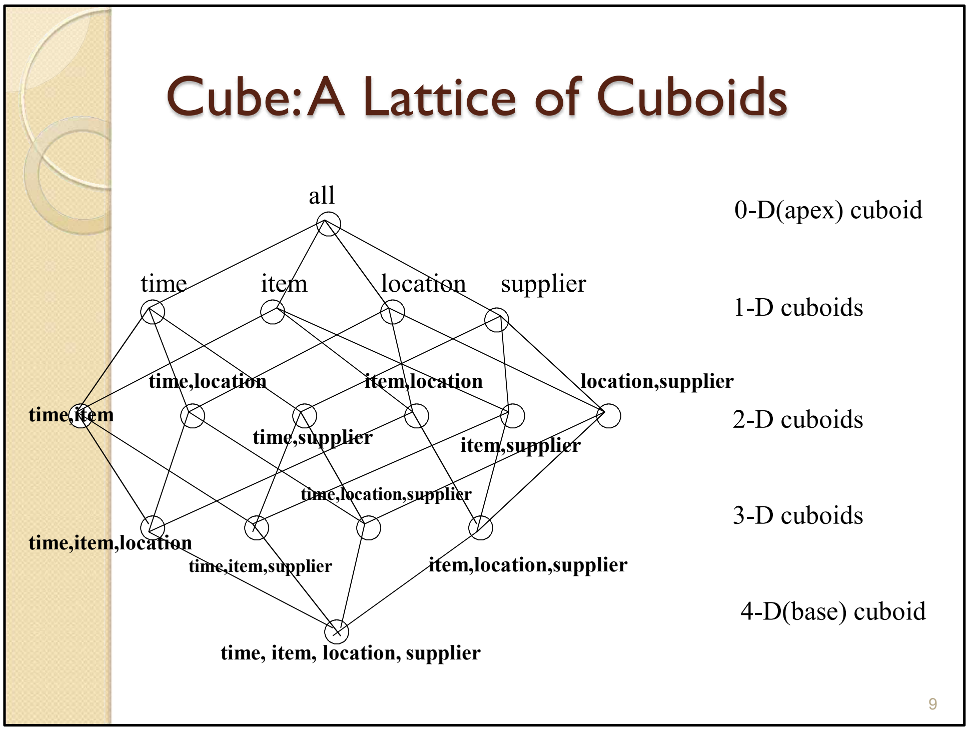
- Defined in many different ways, but not rigorously.
  - A decision support database that is maintained separately from the organization's operational database
  - Support information processing by providing a solid platform of consolidated, historical data for analysis.
- "A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision-making process."—W. H. Inmon
- Data warehousing:
  - The process of constructing and using data warehouses

# Example of Star Schema



# From Tables and Spreadsheets to Data Cubes

- A data warehouse is based on a multidimensional data model which views data in the form of a data cube
- A data cube, such as sales, allows data to be modeled and viewed in multiple dimensions
  - Dimension tables, such as item (item_name, brand, type), or time(day, week, month, quarter, year)
  - Fact table contains measures (such as dollars_sold) and keys to each of the related dimension tables
- In data warehousing literature, an n-D base cube is called a base cuboid. The top most 0-D cuboid, which holds the highest-level of summarization, is called the apex cuboid. The lattice of cuboids forms a data cube.



# Cube: A Lattice of Cuboids

all

time item location supplier

time,item time,location time,supplier item,location item,supplier location,supplier

time,item,location time,item,supplier time,location,supplier item,location,supplier

time, item, location, supplier

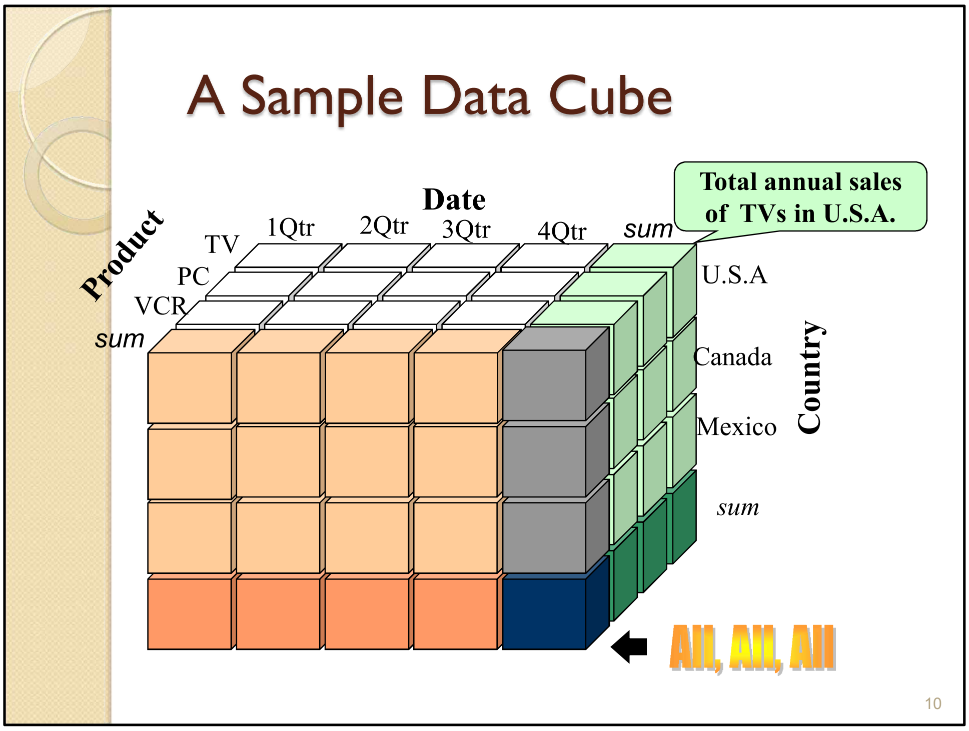
0-D(apex) cuboid

1-D cuboids

2-D cuboids

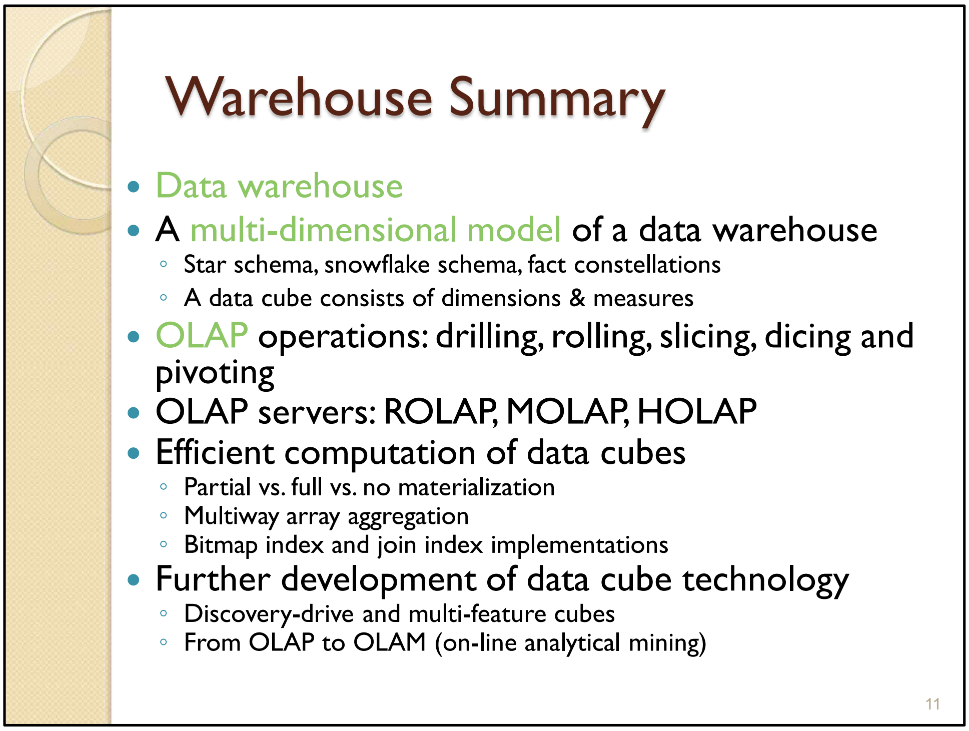
3-D cuboids

4-D(base) cuboid



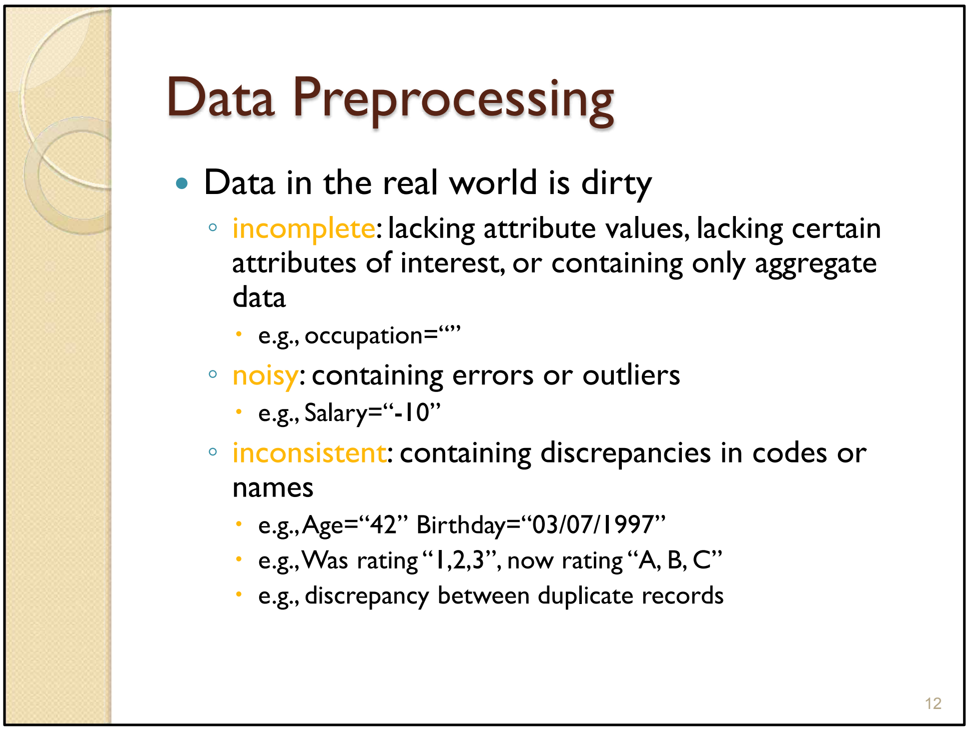
# A Sample Data Cube

Total annual sales of TVs in U.S.A.



## Warehouse Summary

- Data warehouse
- A multi-dimensional model of a data warehouse
  - Star schema, snowflake schema, fact constellations
  - A data cube consists of dimensions & measures
- OLAP operations: drilling, rolling, slicing, dicing and pivoting
- OLAP servers: ROLAP, MOLAP, HOLAP
- Efficient computation of data cubes
  - Partial vs. full vs. no materialization
  - Multiway array aggregation
  - Bitmap index and join index implementations
- Further development of data cube technology
  - Discovery-drive and multi-feature cubes
  - From OLAP to OLAM (on-line analytical mining)



## Data Preprocessing

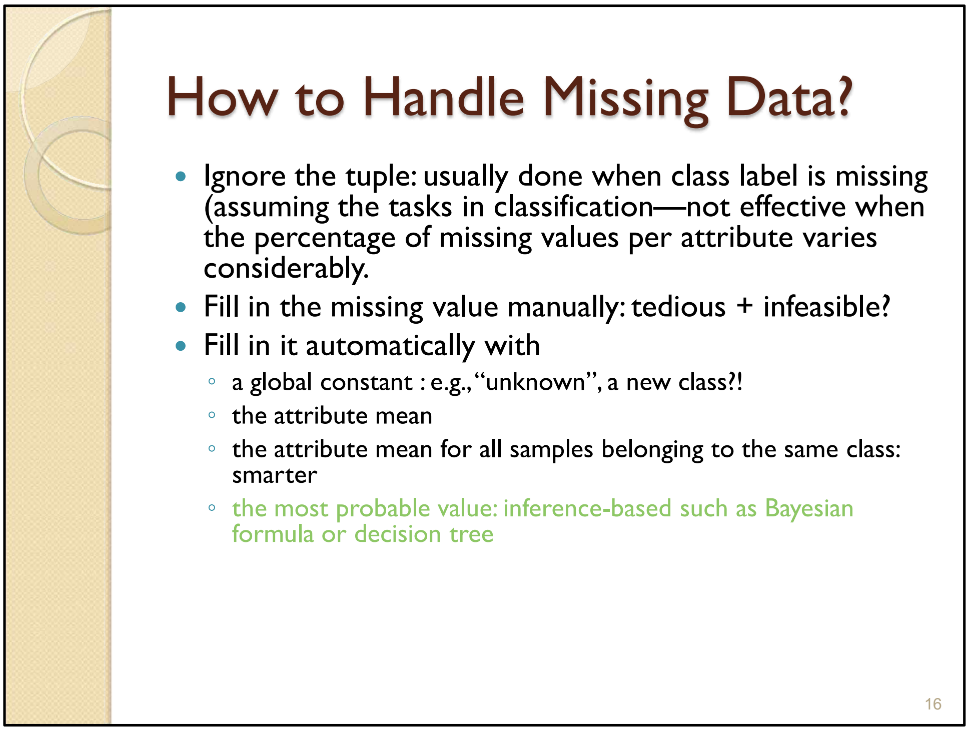
- Data in the real world is dirty
  - incomplete: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data
    - e.g., occupation=“”
  - noisy: containing errors or outliers
    - e.g., Salary=“-10”
  - inconsistent: containing discrepancies in codes or names
    - e.g., Age=“42” Birthday=“03/07/1997”
    - e.g., Was rating “1,2,3”, now rating “A, B, C”
    - e.g., discrepancy between duplicate records

# Multi-Dimensional Measure of Data Quality

- A well-accepted multidimensional view:
  - Accuracy
  - Completeness
  - Consistency
  - Timeliness
  - Believability
  - Value added
  - Interpretability
  - Accessibility
- Broad categories:
  - intrinsic, contextual, representational, and accessibility.

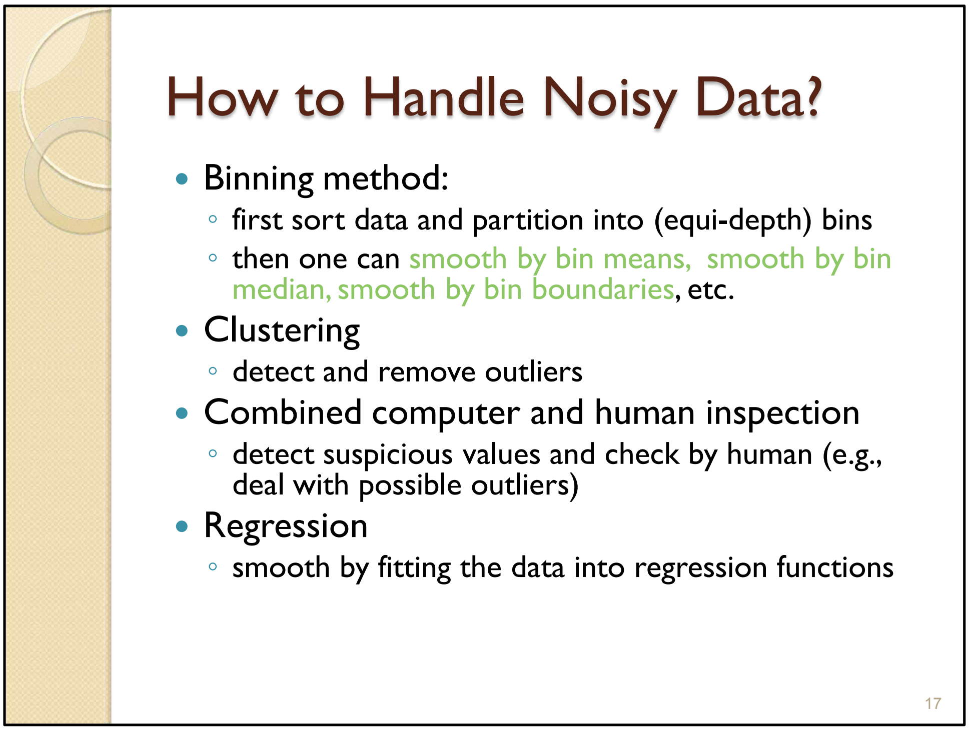
# Major Tasks in Data Preprocessing

- Data cleaning
  - Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies
- Data integration
  - Integration of multiple databases, data cubes, or files
- Data transformation
  - Normalization and aggregation
- Data reduction
  - Obtains reduced representation in volume but produces the same or similar analytical results
- Data discretization
  - Part of data reduction but with particular importance, especially for numerical data



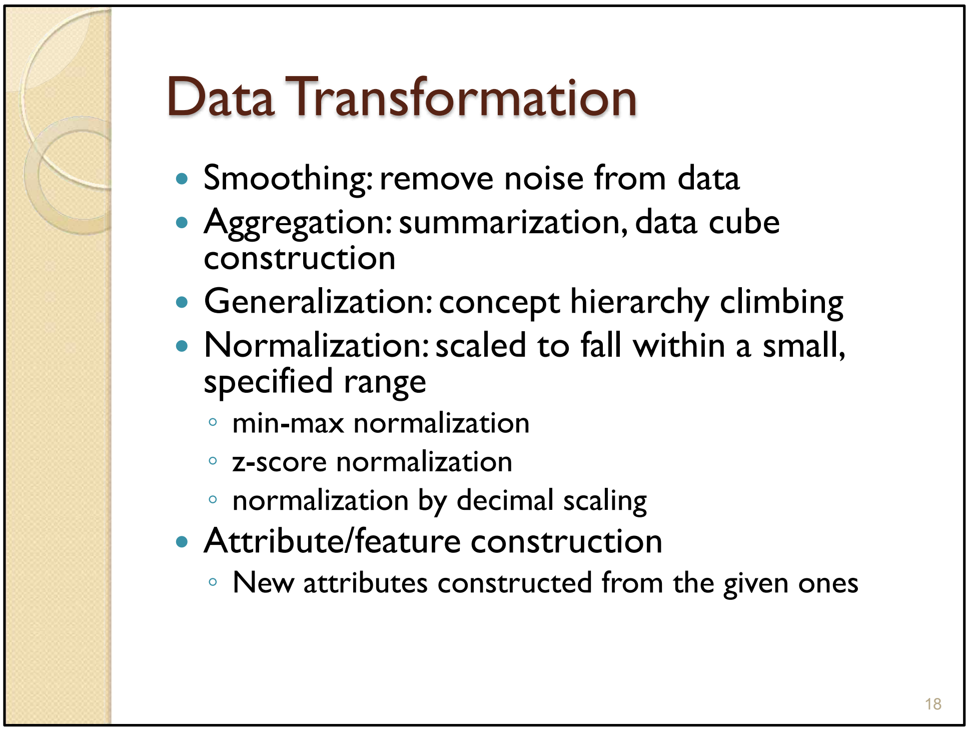
# How to Handle Missing Data?

- Ignore the tuple: usually done when class label is missing (assuming the tasks in classification—not effective when the percentage of missing values per attribute varies considerably.
- Fill in the missing value manually: tedious + infeasible?
- Fill in it automatically with
  - a global constant : e.g., “unknown”, a new class?!
  - the attribute mean
  - the attribute mean for all samples belonging to the same class: smarter
  - the most probable value: inference-based such as Bayesian formula or decision tree



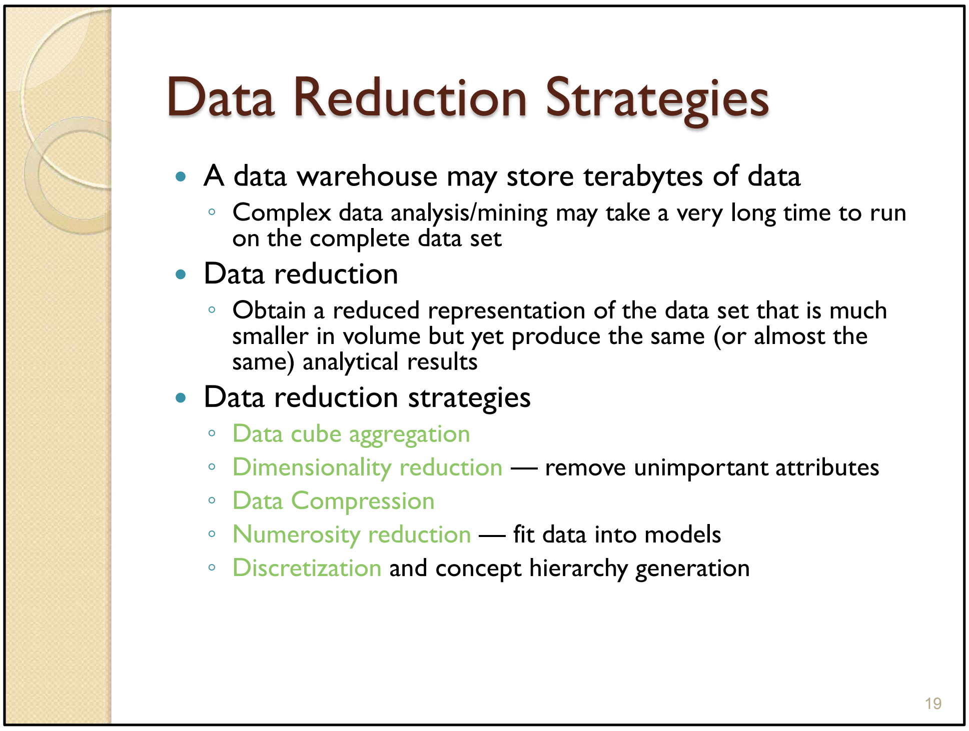
# How to Handle Noisy Data?

- Binning method:
  - first sort data and partition into (equi-depth) bins
  - then one can smooth by bin means, smooth by bin median, smooth by bin boundaries, etc.
- Clustering
  - detect and remove outliers
- Combined computer and human inspection
  - detect suspicious values and check by human (e.g., deal with possible outliers)
- Regression
  - smooth by fitting the data into regression functions



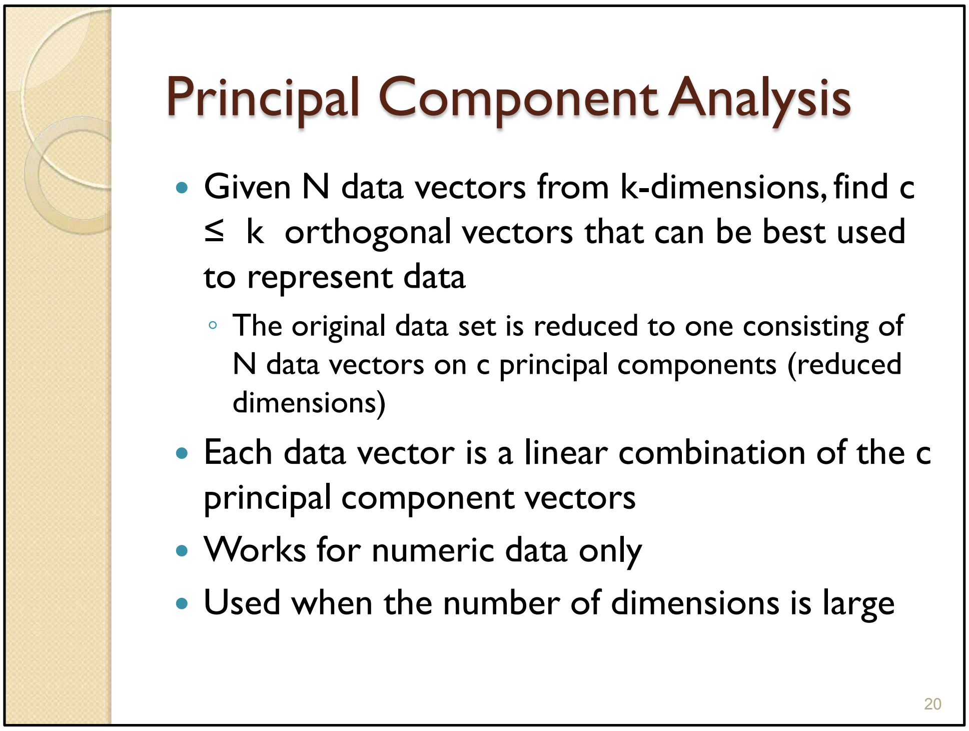
## Data Transformation

- Smoothing: remove noise from data
- Aggregation: summarization, data cube construction
- Generalization: concept hierarchy climbing
- Normalization: scaled to fall within a small, specified range
  - min-max normalization
  - z-score normalization
  - normalization by decimal scaling
- Attribute/feature construction
  - New attributes constructed from the given ones



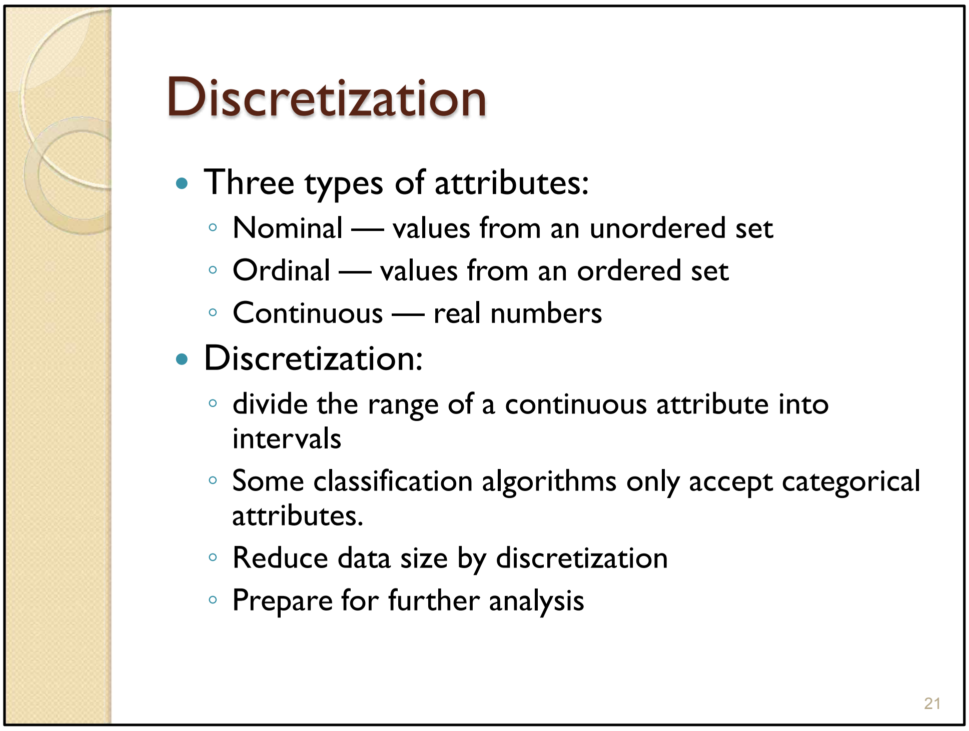
## Data Reduction Strategies

- A data warehouse may store terabytes of data
  - Complex data analysis/mining may take a very long time to run on the complete data set
- Data reduction
  - Obtain a reduced representation of the data set that is much smaller in volume but yet produce the same (or almost the same) analytical results
- Data reduction strategies
  - Data cube aggregation
  - Dimensionality reduction — remove unimportant attributes
  - Data Compression
  - Numerosity reduction — fit data into models
  - Discretization and concept hierarchy generation



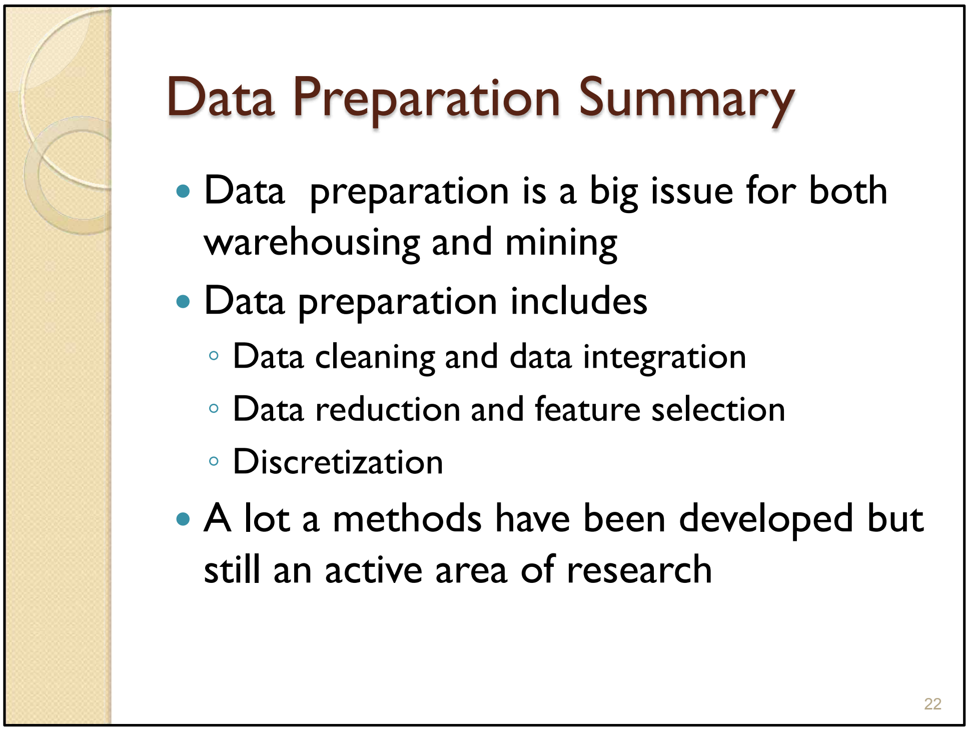
# Principal Component Analysis

- Given N data vectors from k-dimensions, find c ≤ k orthogonal vectors that can be best used to represent data
  - The original data set is reduced to one consisting of N data vectors on c principal components (reduced dimensions)
- Each data vector is a linear combination of the c principal component vectors
- Works for numeric data only
- Used when the number of dimensions is large



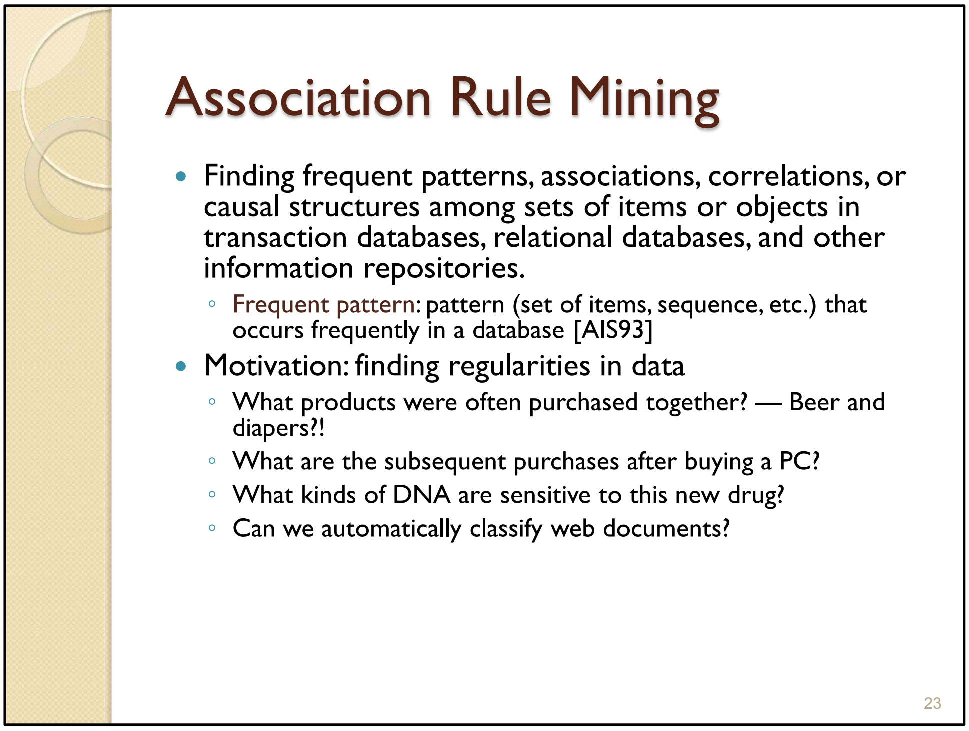
# Discretization

- Three types of attributes:
  - Nominal — values from an unordered set
  - Ordinal — values from an ordered set
  - Continuous — real numbers
- Discretization:
  - divide the range of a continuous attribute into intervals
  - Some classification algorithms only accept categorical attributes.
  - Reduce data size by discretization
  - Prepare for further analysis



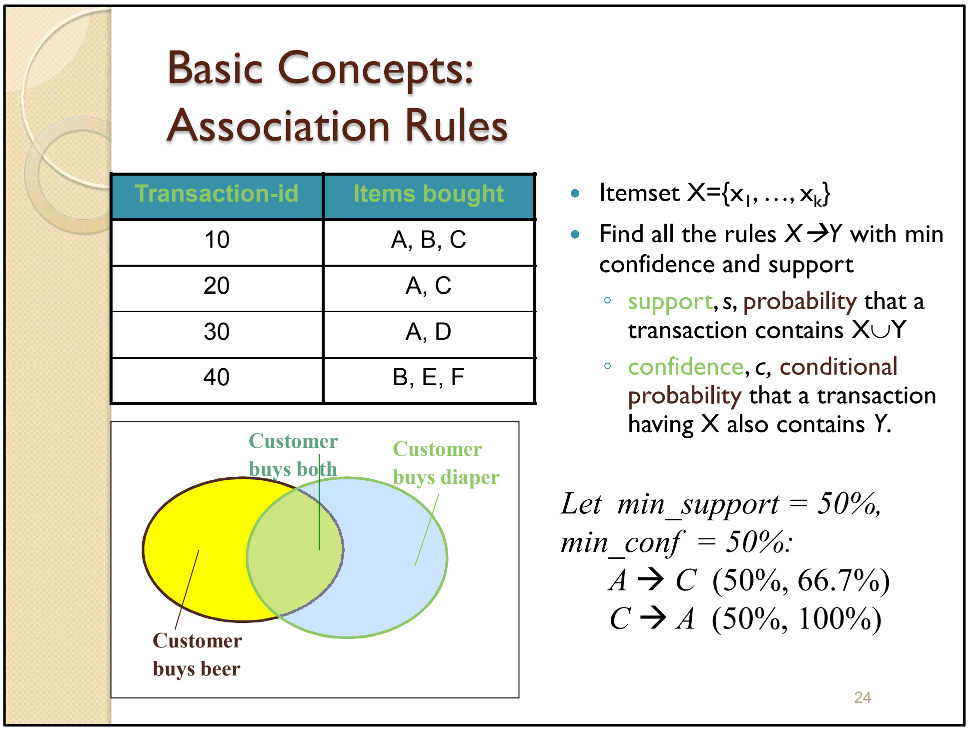
## Data Preparation Summary

- Data  preparation is a big issue for both warehousing and mining
- Data preparation includes
  - Data cleaning and data integration
  - Data reduction and feature selection
  - Discretization
- A lot a methods have been developed but still an active area of research



## Association Rule Mining

- Finding frequent patterns, associations, correlations, or causal structures among sets of items or objects in transaction databases, relational databases, and other information repositories.
  - Frequent pattern: pattern (set of items, sequence, etc.) that occurs frequently in a database [AIS93]
- Motivation: finding regularities in data
  - What products were often purchased together? — Beer and diapers?!
  - What are the subsequent purchases after buying a PC?
  - What kinds of DNA are sensitive to this new drug?
  - Can we automatically classify web documents?



## Basic Concepts: Association Rules

| Transaction-id | Items bought |
|---|---|
| 10 | A, B, C |
| 20 | A, C |
| 30 | A, D |
| 40 | B, E, F |

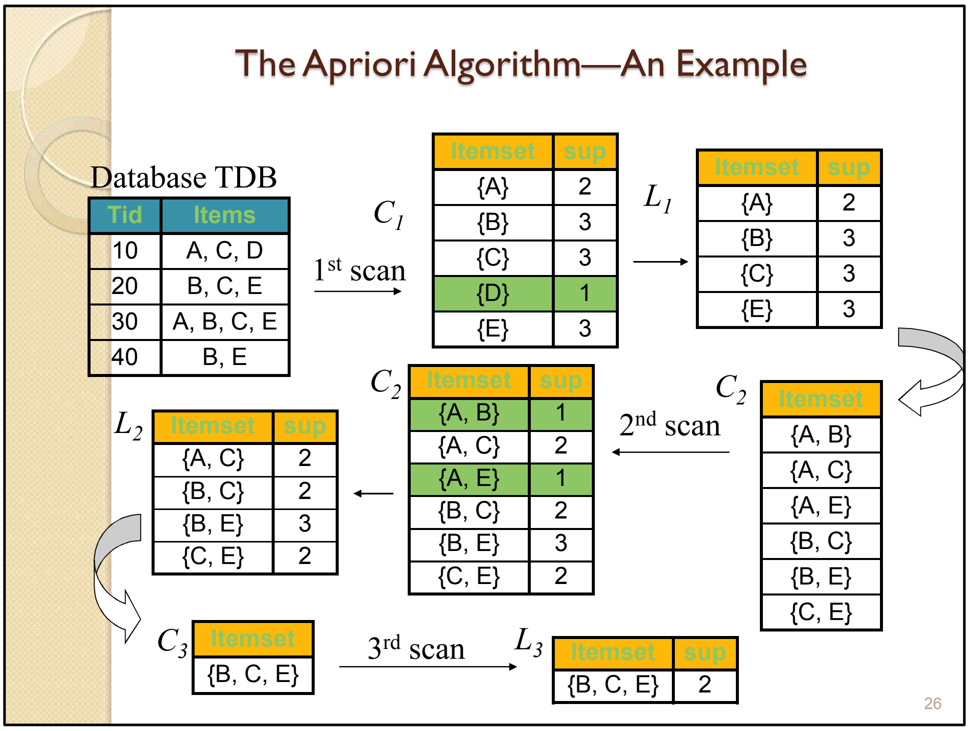
Customer buys both
Customer buys diaper
Customer buys beer

- Itemset X={$x_1$, …, $x_k$}
- Find all the rules X→Y with min confidence and support
  - support, s, probability that a transaction contains X∪Y
  - confidence, c, conditional probability that a transaction having X also contains Y.

*Let min_support = 50%,*
*min_conf = 50%:*

$A \rightarrow C$ (50%, 66.7%)

$C \rightarrow A$ (50%, 100%)



## The Apriori Algorithm—An Example

Database TDB

| Tid | Items |
|---|---|
| 10 | A, C, D |
| 20 | B, C, E |
| 30 | A, B, C, E |
| 40 | B, E |

1st scan →

$C_1$

| Itemset | sup |
|---|---|
| {A} | 2 |
| {B} | 3 |
| {C} | 3 |
| {D} | 1 |
| {E} | 3 |

→ $L_1$

| Itemset | sup |
|---|---|
| {A} | 2 |
| {B} | 3 |
| {C} | 3 |
| {E} | 3 |

$C_2$

| Itemset |
|---|
| {A, B} |
| {A, C} |
| {A, E} |
| {B, C} |
| {B, E} |
| {C, E} |

2nd scan ←

$C_2$

| Itemset | sup |
|---|---|
| {A, B} | 1 |
| {A, C} | 2 |
| {A, E} | 1 |
| {B, C} | 2 |
| {B, E} | 3 |
| {C, E} | 2 |

← $L_2$

| Itemset | sup |
|---|---|
| {A, C} | 2 |
| {B, C} | 2 |
| {B, E} | 3 |
| {C, E} | 2 |

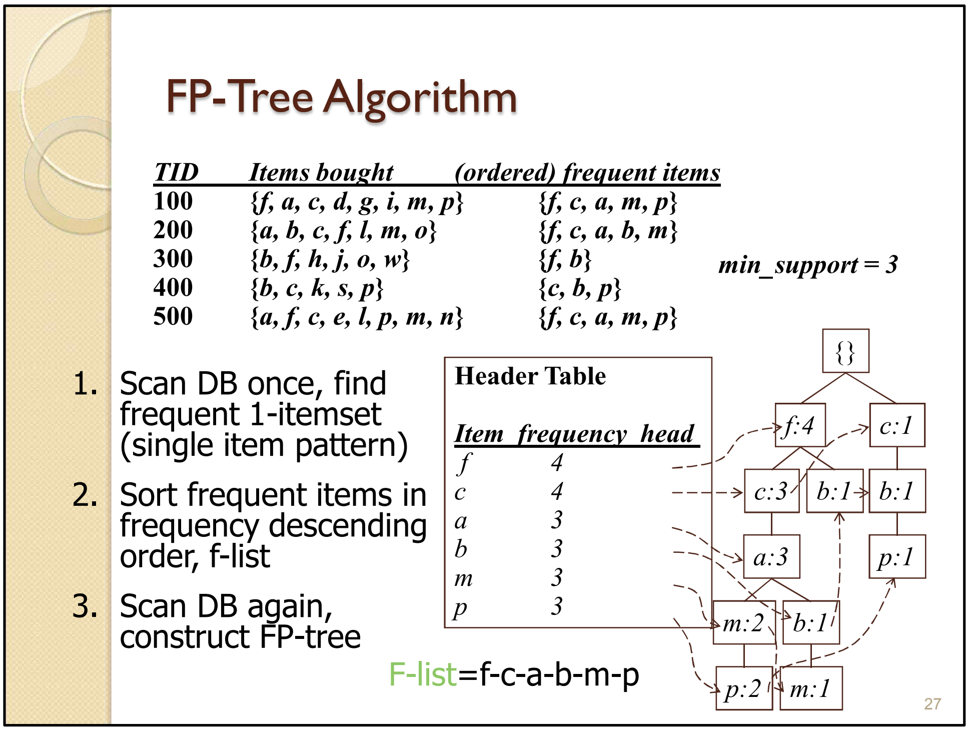
$C_3$

| Itemset |
|---|
| {B, C, E} |

3rd scan →

$L_3$

| Itemset | sup |
|---|---|
| {B, C, E} | 2 |



# FP-Tree Algorithm

| *TID* | *Items bought* | *(ordered) frequent items* |
|---|---|---|
| 100 | {*f, a, c, d, g, i, m, p*} | {*f, c, a, m, p*} |
| 200 | {*a, b, c, f, l, m, o*} | {*f, c, a, b, m*} |
| 300 | {*b, f, h, j, o, w*} | {*f, b*} |
| 400 | {*b, c, k, s, p*} | {*c, b, p*} |
| 500 | {*a, f, c, e, l, p, m, n*} | {*f, c, a, m, p*} |

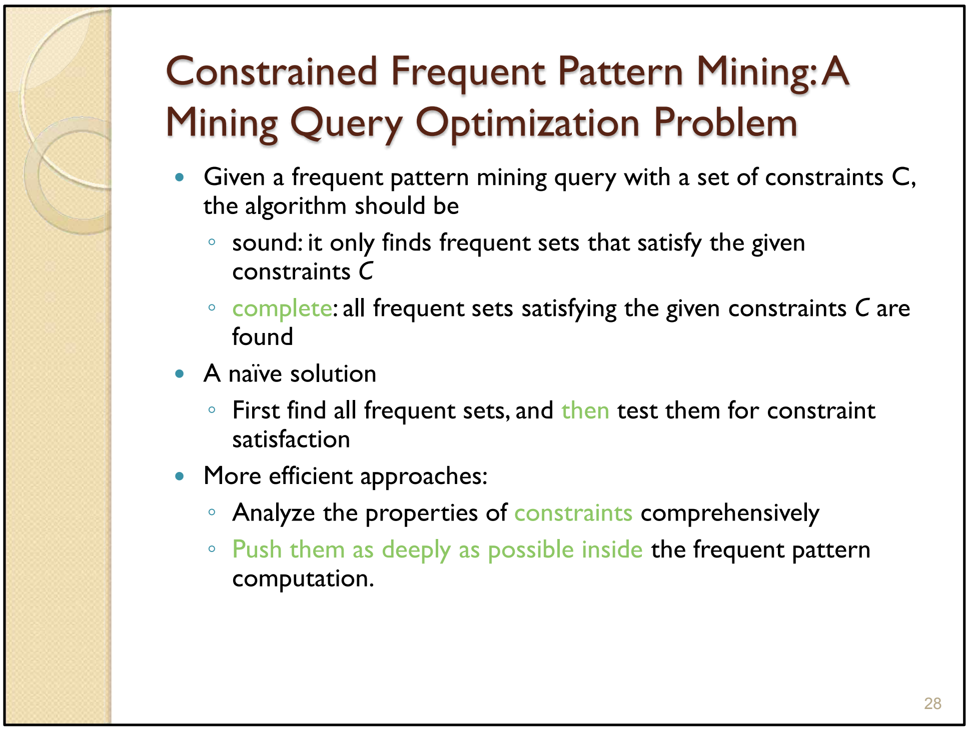
*min_support* = 3

1. Scan DB once, find frequent 1-itemset (single item pattern)
2. Sort frequent items in frequency descending order, f-list
3. Scan DB again, construct FP-tree

**Header Table**

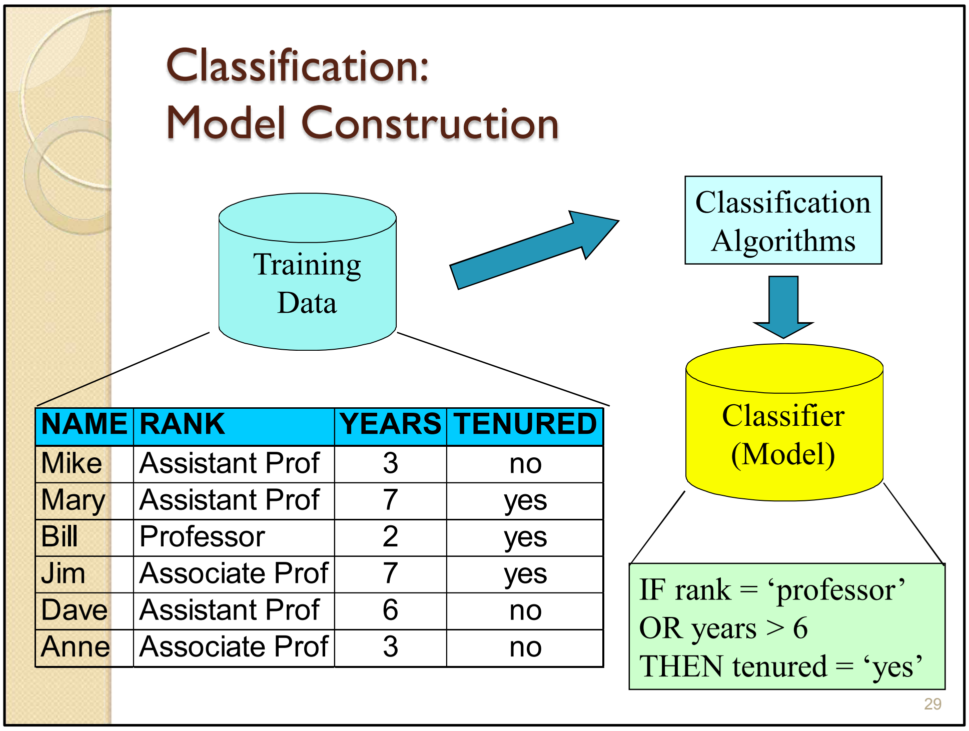
| *Item* | *frequency* | *head* |
|---|---|---|
| *f* | *4* | |
| *c* | *4* | |
| *a* | *3* | |
| *b* | *3* | |
| *m* | *3* | |
| *p* | *3* | |

F-list=f-c-a-b-m-p



# Constrained Frequent Pattern Mining: A Mining Query Optimization Problem

- Given a frequent pattern mining query with a set of constraints C, the algorithm should be
  - sound: it only finds frequent sets that satisfy the given constraints *C*
  - complete: all frequent sets satisfying the given constraints *C* are found
- A naïve solution
  - First find all frequent sets, and then test them for constraint satisfaction
- More efficient approaches:
  - Analyze the properties of constraints comprehensively
  - Push them as deeply as possible inside the frequent pattern computation.



## Classification: Model Construction

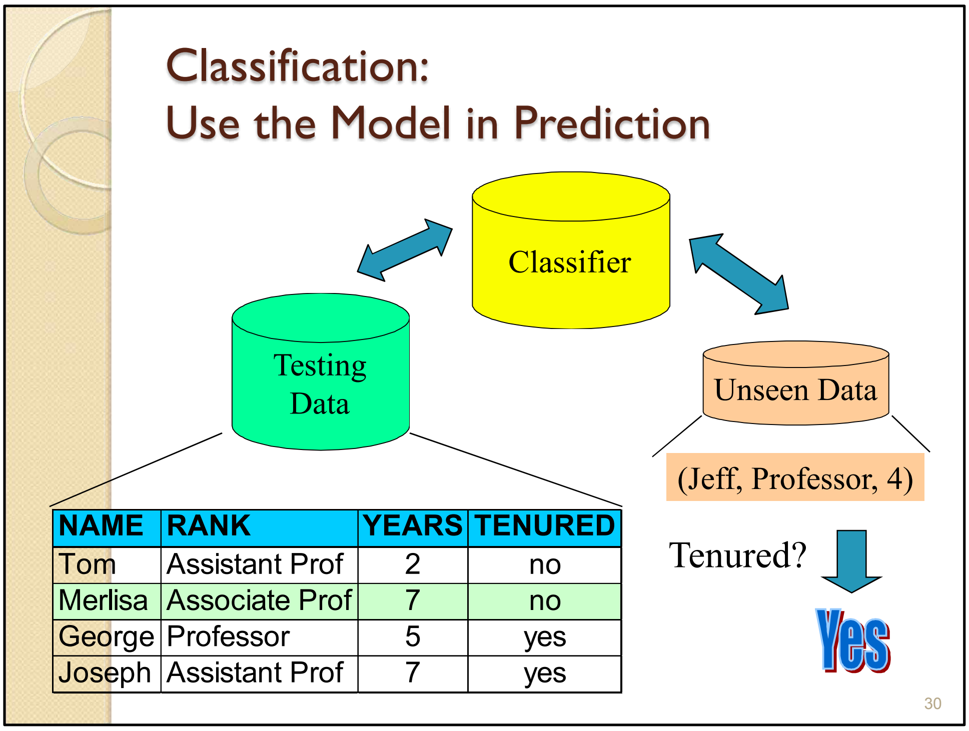
Training Data

Classification Algorithms

Classifier (Model)

| NAME | RANK | YEARS | TENURED |
|---|---|---|---|
| Mike | Assistant Prof | 3 | no |
| Mary | Assistant Prof | 7 | yes |
| Bill | Professor | 2 | yes |
| Jim | Associate Prof | 7 | yes |
| Dave | Assistant Prof | 6 | no |
| Anne | Associate Prof | 3 | no |

IF rank = ‘professor’
OR years > 6
THEN tenured = ‘yes’



## Classification: Use the Model in Prediction

Classifier

Testing Data

Unseen Data

(Jeff, Professor, 4)

Tenured?

Yes

| NAME | RANK | YEARS | TENURED |
|---|---|---|---|
| Tom | Assistant Prof | 2 | no |
| Merlisa | Associate Prof | 7 | no |
| George | Professor | 5 | yes |
| Joseph | Assistant Prof | 7 | yes |

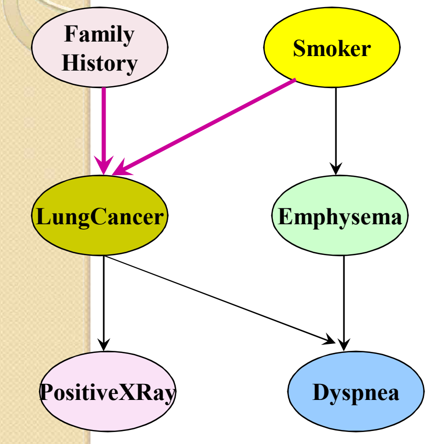
# Naïve Bayes Classifier

- A simplified assumption: attributes are conditionally independent:

$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i)$$

- The product of occurrence of say 2 elements $x_1$ and $x_2$, given the current class is C, is the product of the probabilities of each element taken separately, given the same class $P([y_1,y_2],C) = P(y_1,C) * P(y_2,C)$
- No dependence relation between attributes
- Greatly reduces the computation cost, only count the class distribution.
- Once the probability $P(X|C_i)$ is known, assign X to the class with maximum $P(X|C_i)*P(C_i)$

# Bayesian Belief Network

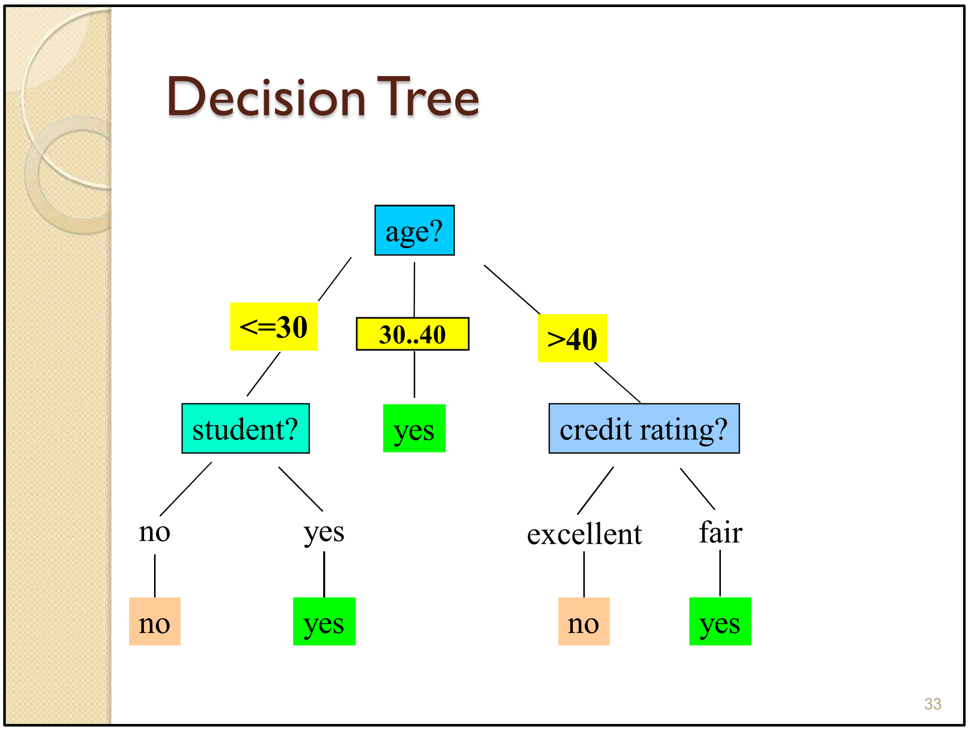


Bayesian Belief Networks

|  | (FH, S) | (FH, ~S) | (~FH, S) | (~FH, ~S) |
|---|---|---|---|---|
| LC | 0.8 | 0.5 | 0.7 | 0.1 |
| ~LC | 0.2 | 0.5 | 0.3 | 0.9 |

The conditional probability table for the variable LungCancer: Shows the conditional probability for each possible combination of its parents

$$P(z1,...,zn) = \prod_{i=1}^{n} P(z_i \mid Parents(Z_i))$$



# Decision Tree

age?

<=30

30..40

>40

student?

yes

credit rating?

no

yes

excellent

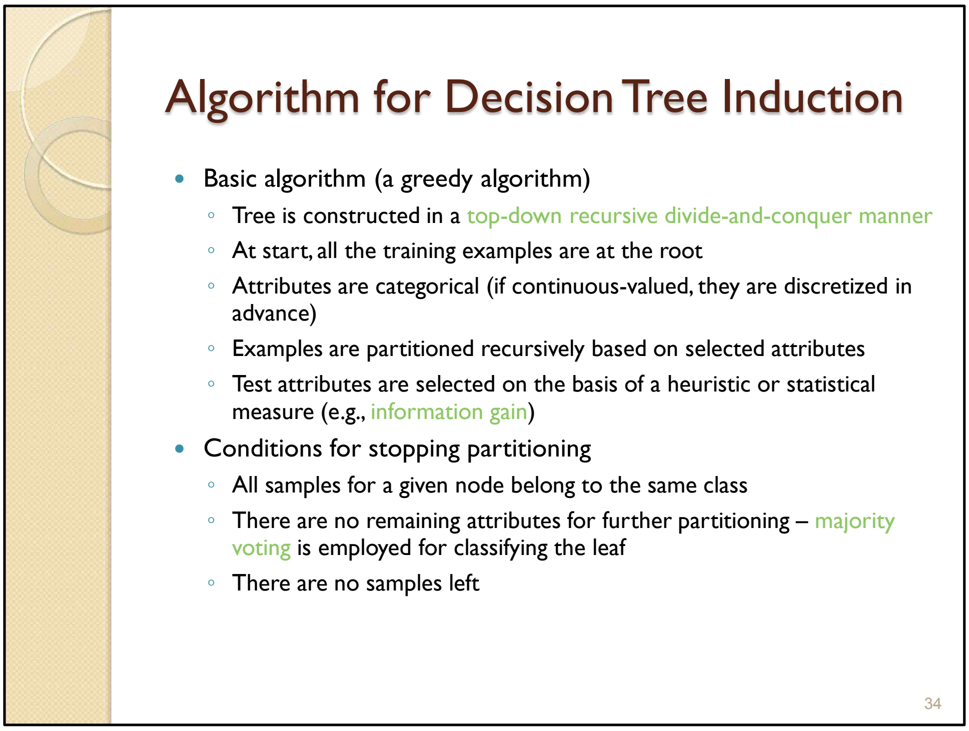
fair

no

yes

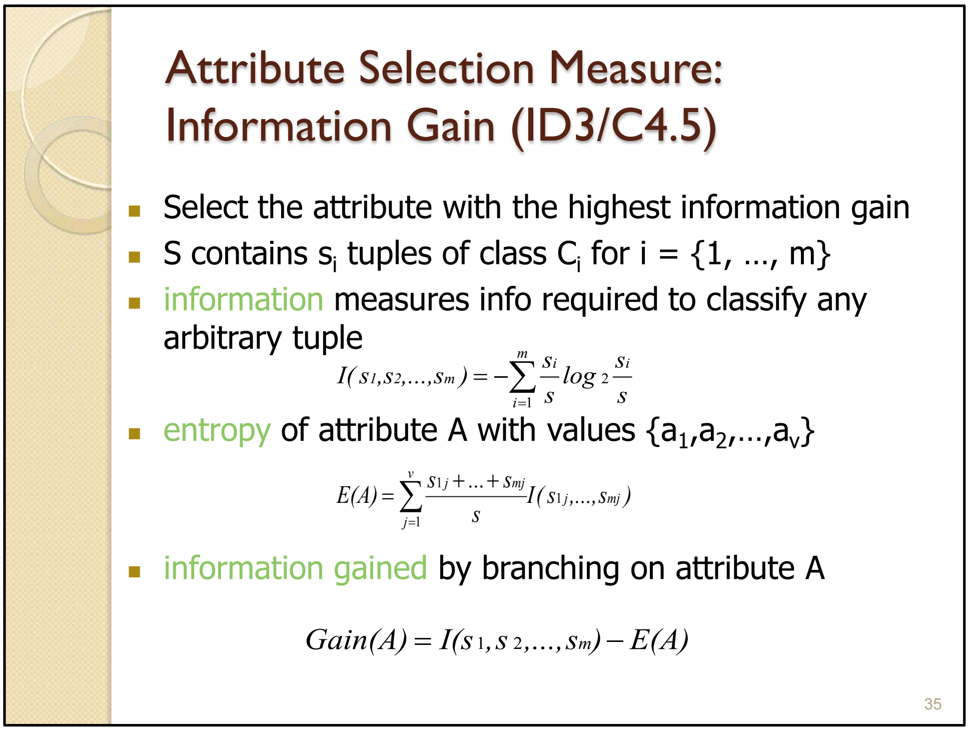
no

yes



# Algorithm for Decision Tree Induction

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - Examples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)
- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left



## Attribute Selection Measure: Information Gain (ID3/C4.5)

- Select the attribute with the highest information gain
- S contains $s_i$ tuples of class $C_i$ for i = {1, …, m}
- information measures info required to classify any arbitrary tuple

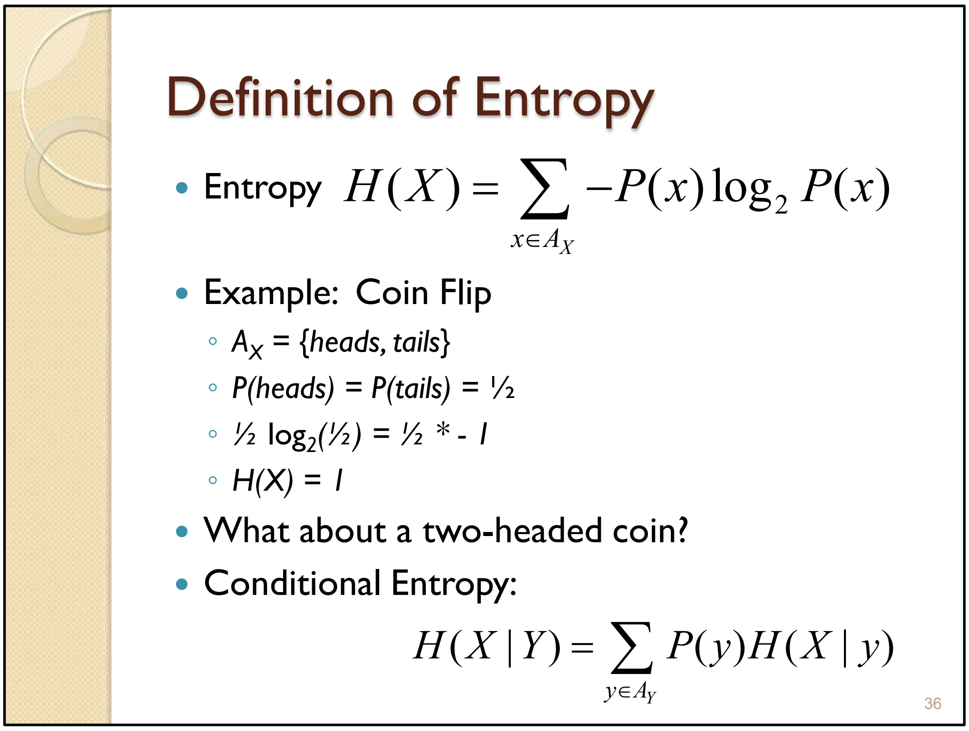
$$I(s_1,s_2,\ldots,s_m) = -\sum_{i=1}^{m} \frac{s_i}{s} \log_2 \frac{s_i}{s}$$

- entropy of attribute A with values $\{a_1,a_2,\ldots,a_v\}$

$$E(A) = \sum_{j=1}^{v} \frac{s_{1j}+\ldots+s_{mj}}{s} I(s_{1j},\ldots,s_{mj})$$

- information gained by branching on attribute A

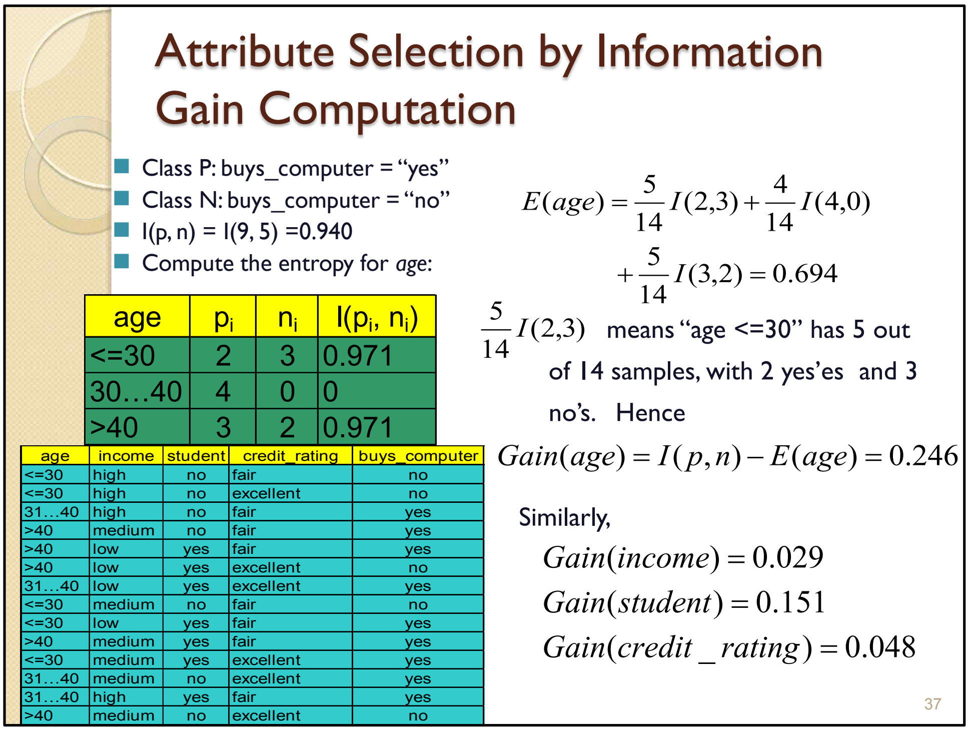
$$Gain(A) = I(s_1, s_2,\ldots, s_m) - E(A)$$



## Definition of Entropy

- Entropy $H(X) = \sum_{x \in A_X} -P(x) \log_2 P(x)$
- Example: Coin Flip
  - $A_X$ = {*heads, tails*}
  - *P(heads)* = *P(tails)* = ½
  - ½ $\log_2$(½) = ½ * - *1*
  - *H(X)* = *1*
- What about a two-headed coin?
- Conditional Entropy:

$$H(X \mid Y) = \sum_{y \in A_Y} P(y) H(X \mid y)$$



## Attribute Selection by Information Gain Computation

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"
- I(p, n) = I(9, 5) =0.940
- Compute the entropy for *age*:

| age | $p_i$ | $n_i$ | $I(p_i, n_i)$ |
|---|---|---|---|
| <=30 | 2 | 3 | 0.971 |
| 30…40 | 4 | 0 | 0 |
| >40 | 3 | 2 | 0.971 |

| age | income | student | credit_rating | buys_computer |
|---|---|---|---|---|
| <=30 | high | no | fair | no |
| <=30 | high | no | excellent | no |
| 31…40 | high | no | fair | yes |
| >40 | medium | no | fair | yes |
| >40 | low | yes | fair | yes |
| >40 | low | yes | excellent | no |
| 31…40 | low | yes | excellent | yes |
| <=30 | medium | no | fair | no |
| <=30 | low | yes | fair | yes |
| >40 | medium | yes | fair | yes |
| <=30 | medium | yes | excellent | yes |
| 31…40 | medium | no | excellent | yes |
| 31…40 | high | yes | fair | yes |
| >40 | medium | no | excellent | no |

$$E(age) = \frac{5}{14} I(2,3) + \frac{4}{14} I(4,0)$$
$$+ \frac{5}{14} I(3,2) = 0.694$$

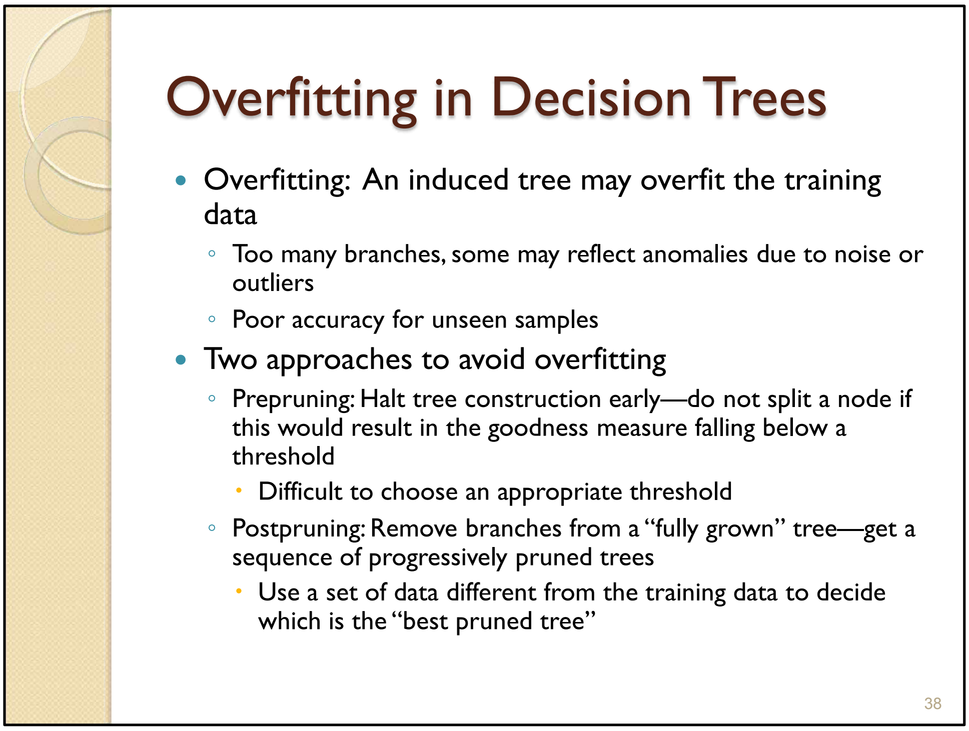
$\frac{5}{14} I(2,3)$ means "age <=30" has 5 out of 14 samples, with 2 yes'es and 3 no's. Hence

$$Gain(age) = I(p,n) - E(age) = 0.246$$

Similarly,

$$Gain(income) = 0.029$$
$$Gain(student) = 0.151$$
$$Gain(credit\_rating) = 0.048$$

37

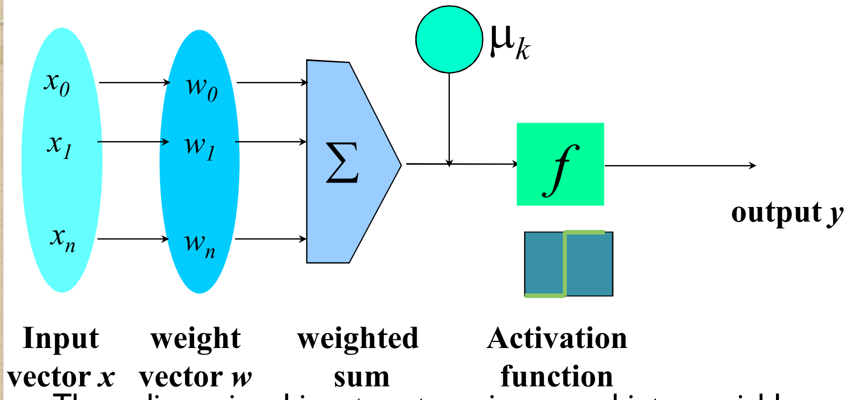


## Overfitting in Decision Trees

- Overfitting: An induced tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples
- Two approaches to avoid overfitting
  - Prepruning: Halt tree construction early—do not split a node if this would result in the goodness measure falling below a threshold
    - Difficult to choose an appropriate threshold
  - Postpruning: Remove branches from a "fully grown" tree—get a sequence of progressively pruned trees
    - Use a set of data different from the training data to decide which is the "best pruned tree"

38

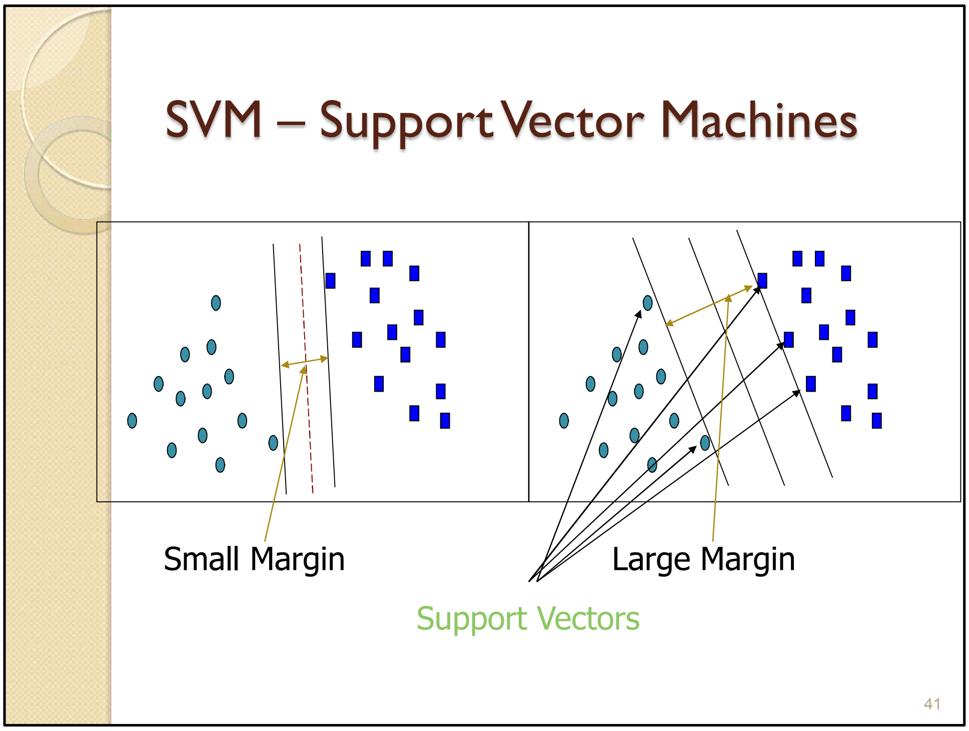
# Artificial Neural Networks: A Neuron



- The *n*-dimensional input vector *x* is mapped into variable *y* by means of the scalar product and a nonlinear function mapping

# Artificial Neural Networks: Training

- The ultimate objective of training
  - obtain a set of weights that makes almost all the tuples in the training data classified correctly
- Steps
  - Initialize weights with random values
  - Feed the input tuples into the network one by one
  - For each unit
    - Compute the net input to the unit as a linear combination of all the inputs to the unit
    - Compute the output value using the activation function
    - Compute the error
    - Update the weights and the bias

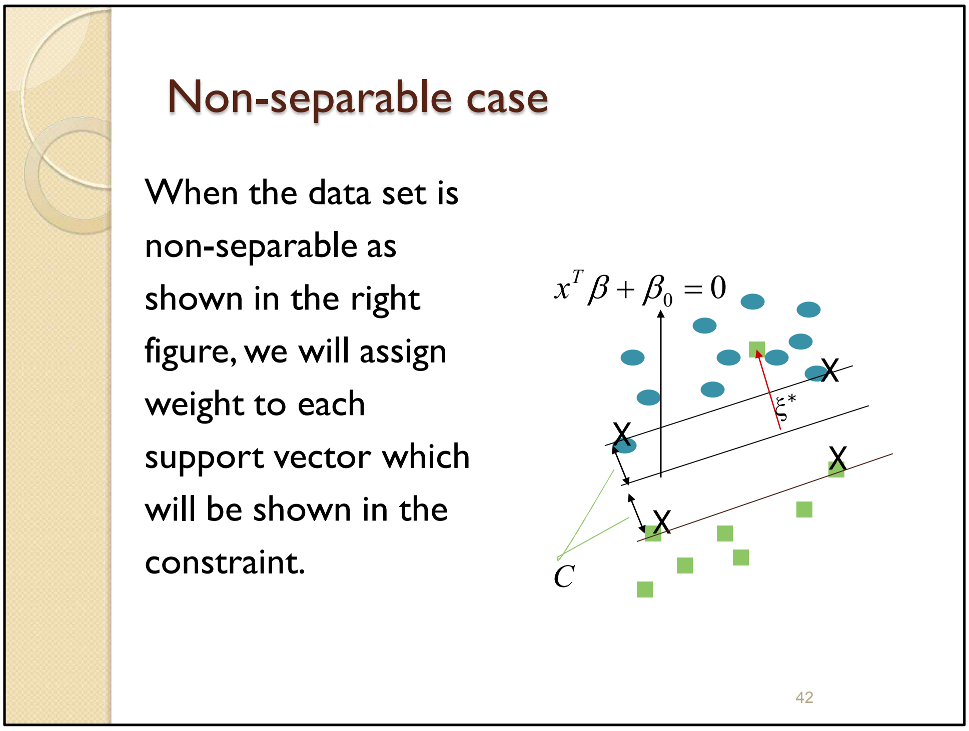


## SVM – Support Vector Machines

Small Margin

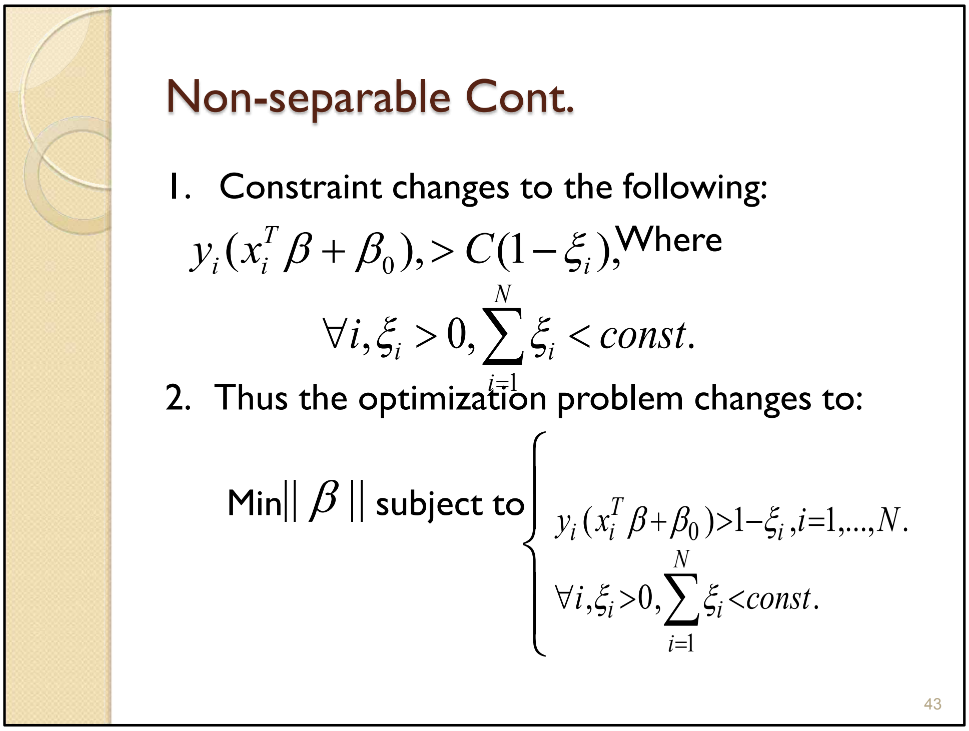
Large Margin

Support Vectors



## Non-separable case

When the data set is non-separable as shown in the right figure, we will assign weight to each support vector which will be shown in the constraint.



## Non-separable Cont.

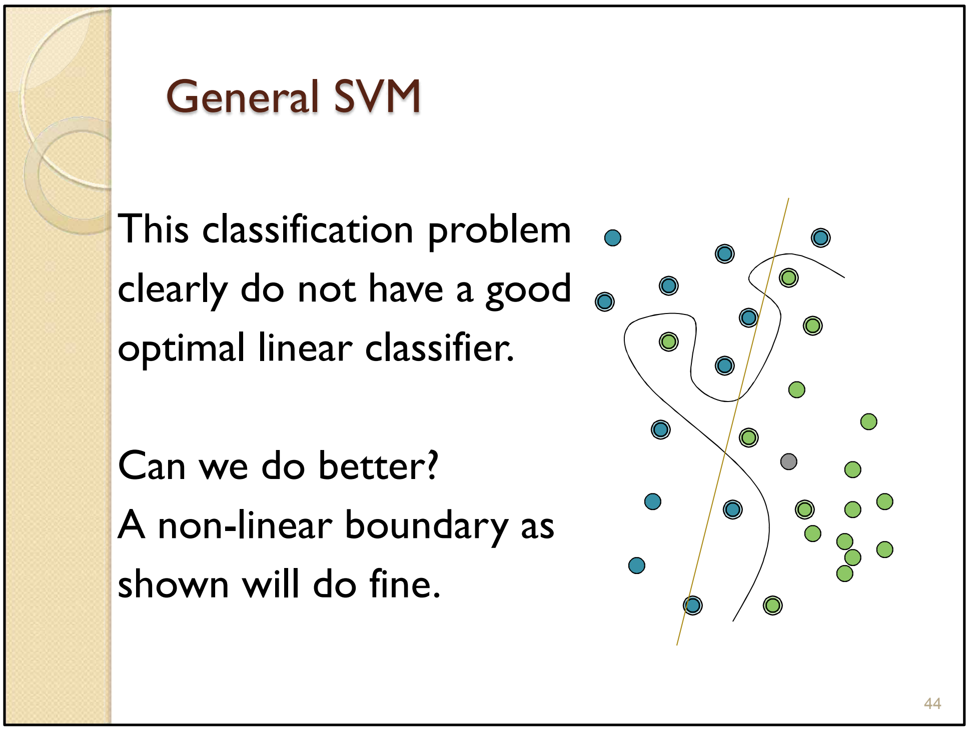
1. Constraint changes to the following:

$$y_i(x_i^T\beta + \beta_0), > C(1-\xi_i),\text{Where}$$

$$\forall i, \xi_i > 0, \sum_{i=1}^{N} \xi_i < const.$$

2. Thus the optimization problem changes to:

$$\text{Min}\|\beta\| \text{ subject to } \begin{cases} y_i(x_i^T\beta+\beta_0)>1-\xi_i, i=1,\ldots,N. \\ \forall i, \xi_i>0, \sum_{i=1}^{N}\xi_i<const. \end{cases}$$

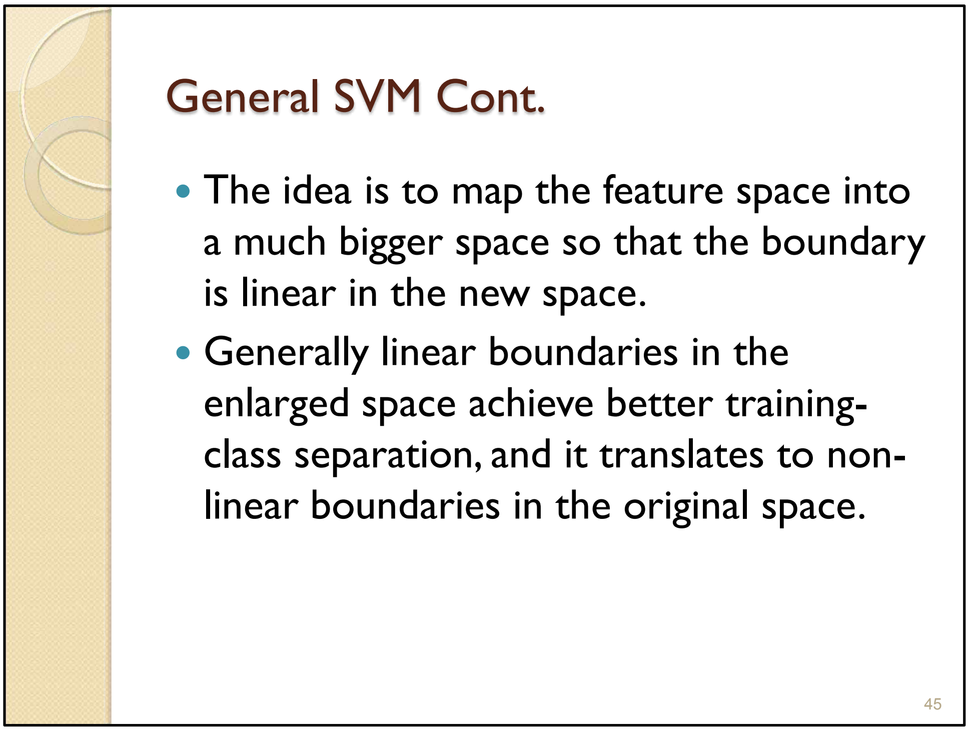


## General SVM

This classification problem clearly do not have a good optimal linear classifier.

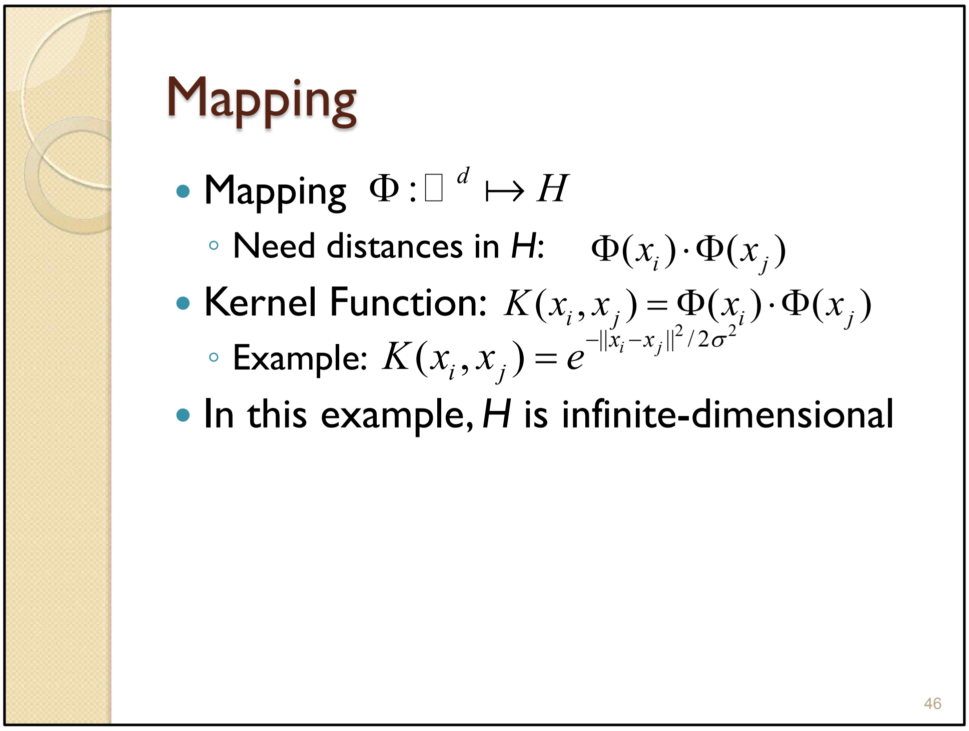
Can we do better?

A non-linear boundary as shown will do fine.



## General SVM Cont.

- The idea is to map the feature space into a much bigger space so that the boundary is linear in the new space.
- Generally linear boundaries in the enlarged space achieve better training-class separation, and it translates to non-linear boundaries in the original space.

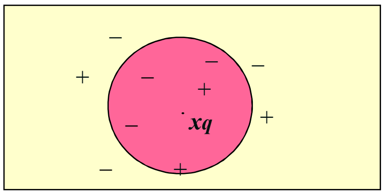


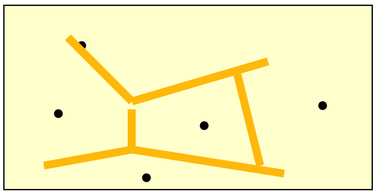
## Mapping

- Mapping $\Phi : \mathbb{R}^{d} \mapsto H$
  - Need distances in *H*: $\Phi(x_i)\cdot\Phi(x_j)$
- Kernel Function: $K(x_i, x_j) = \Phi(x_i)\cdot\Phi(x_j)$
  - Example: $K(x_i, x_j) = e^{-\|x_i - x_j\|^2/2\sigma^2}$
- In this example, *H* is infinite-dimensional

## The *k*-Nearest Neighbor Algorithm

- All instances correspond to points in the n-D space.
- The nearest neighbor are defined in terms of Euclidean distance.
- The target function could be discrete- or real- valued.
- For discrete-valued, the *k*-NN returns the most common value among the k training examples nearest to *xq*.
- Voronoi diagram: the decision surface induced by 1-NN for a typical set of training examples.





## Case-Based Reasoning

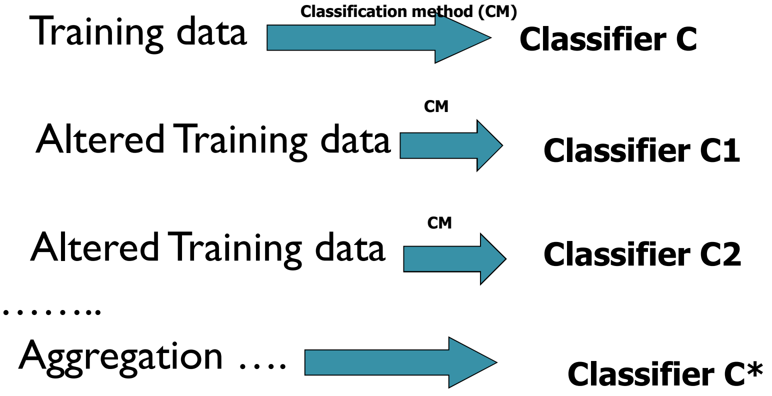
- Also uses: lazy evaluation + analyze similar instances
- Difference: Instances are not "points in a Euclidean space"
- Example: Water faucet problem in CADET (Sycara et al'92)
- Methodology
  - Instances represented by rich symbolic descriptions (e.g., function graphs)
  - Multiple retrieved cases may be combined
  - Tight coupling between case retrieval, knowledge-based reasoning, and problem solving
- Research issues
  - Indexing based on syntactic similarity measure, and when failure, backtracking, and adapting to additional cases

## Regress Analysis and Log-Linear Models in Prediction

- Linear regression: $Y = \alpha + \beta X$
  - Two parameters , $\alpha$ and $\beta$ specify the line and are to be estimated by using the data at hand.
  - using the least squares criterion to the known values of $Y_1, Y_2, \ldots, X_1, X_2, \ldots$.
- Multiple regression: $Y = b0 + b1\ X1 + b2\ X2$.
  - Many nonlinear functions can be transformed into the above.
- Log-linear models:
  - The multi-way table of joint probabilities is approximated by a product of lower-order tables.
  - Probability: $p(a, b, c, d) = \alpha_{ab}\ \beta_{ac}\chi_{ad}\ \delta_{bcd}$

## Bagging and Boosting

- General idea



Training data

Classification method (CM)

Classifier C

Altered Training data

CM

Classifier C1

Altered Training data

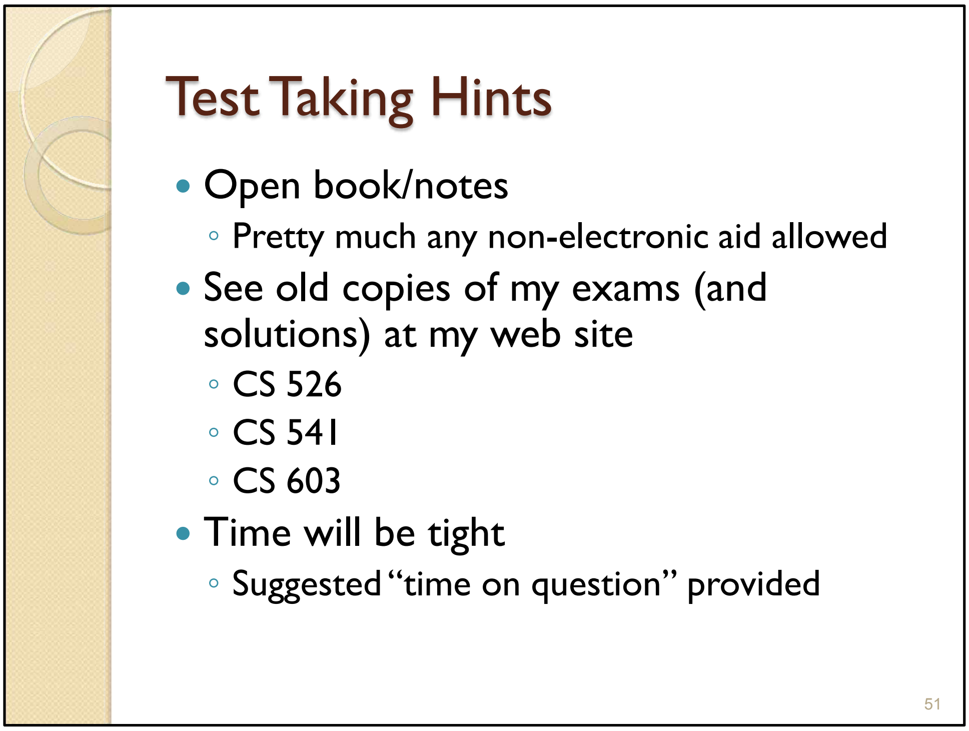
CM

Classifier C2

……..

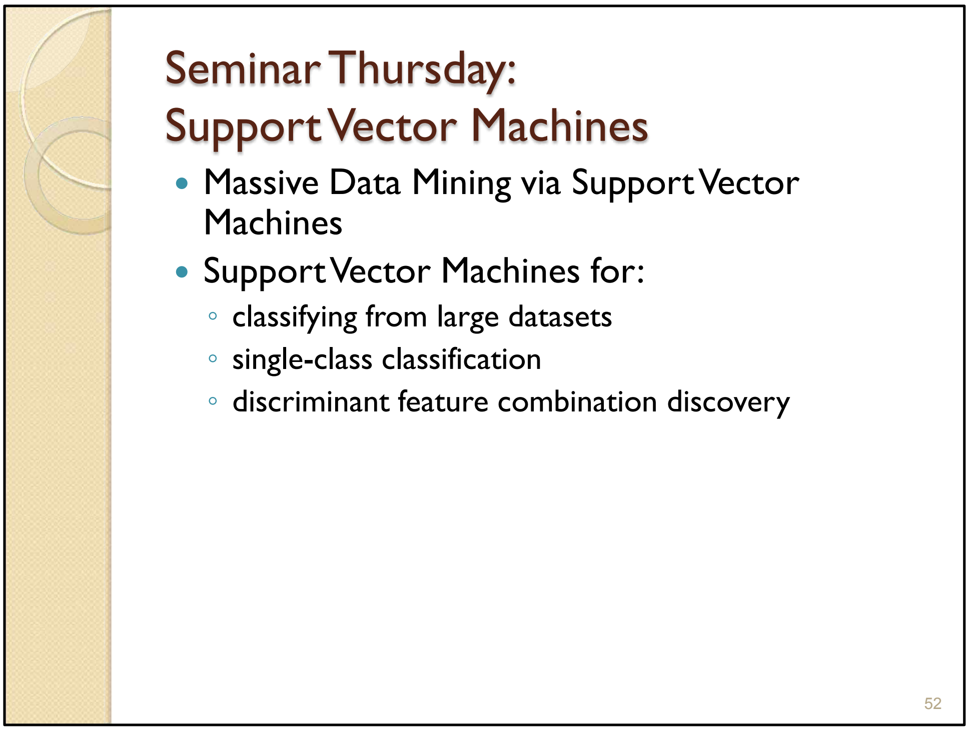
Aggregation ….

Classifier C*



# Test Taking Hints

- Open book/notes
  - Pretty much any non-electronic aid allowed
- See old copies of my exams (and solutions) at my web site
  - CS 526
  - CS 541
  - CS 603
- Time will be tight
  - Suggested "time on question" provided



# Seminar Thursday: Support Vector Machines

- Massive Data Mining via Support Vector Machines
- Support Vector Machines for:
  - classifying from large datasets
  - single-class classification
  - discriminant feature combination discovery