KHAI THÁC DỮ LIỆU & KHAI PHÁ TRI THỨC

Data Mining & Knowledge Discovery

Bài 5. **Phân loại/ Classification**
Advanced methods

Mã MH: 505043

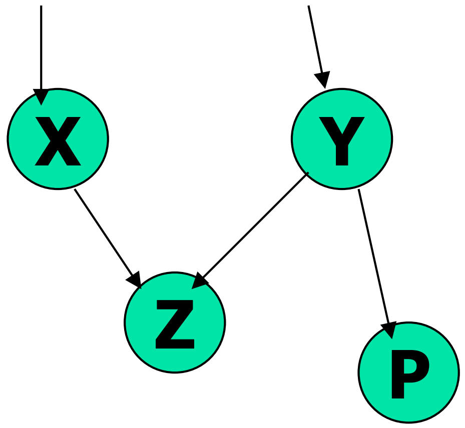
TS. HOÀNG Anh

Chapter 9. Classification: Advanced Methods

- **Bayesian Belief Networks**
- Classification by Backpropagation
- Support Vector Machines
- Classification by Using Frequent Patterns
- **Lazy Learners (or Learning from Your Neighbors)**
- Other Classification Methods
- Additional Topics Regarding Classification
- Summary

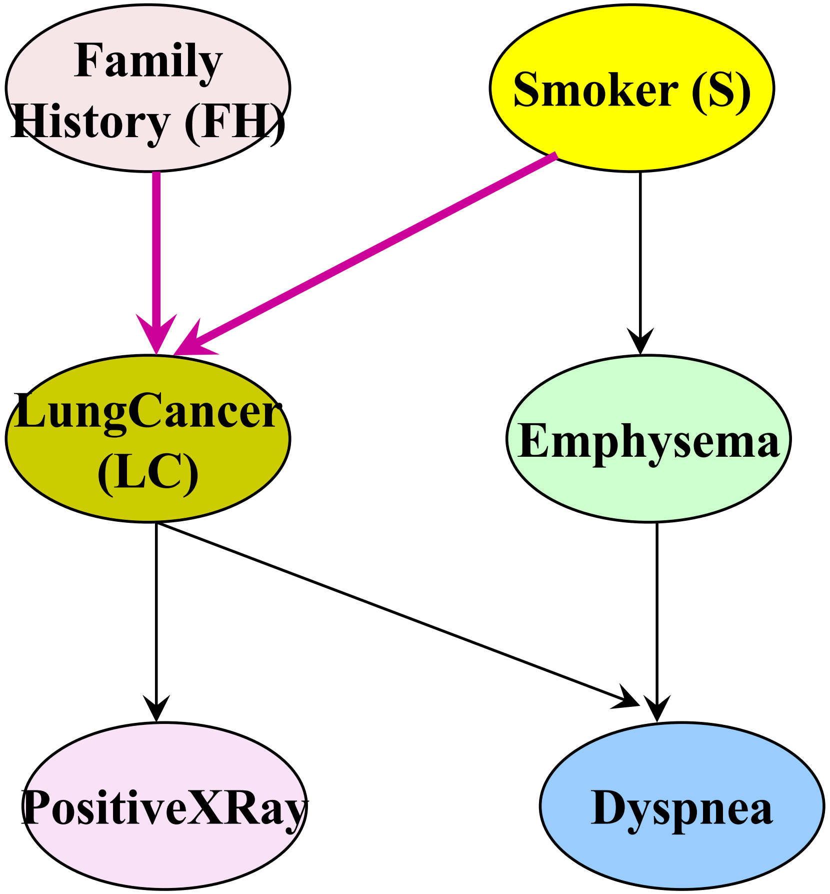
Bayesian Belief Networks

- **Bayesian belief networks** (also known as **Bayesian networks**, **probabilistic networks**): cho phép lớp độc lập có điều kiện giữa các tập con của các biến.
- Một (*directed acyclic*) đồ thị có hướng thể hiện mối quan hệ nhân quả
  - Sự phụ thuộc giữa các biến
  - Đưa ra các đặc trưng của phân phối xác suất liên kết



- ❑ Nodes: các biến ngẫu nhiên
- ❑ Links: sự phụ thuộc giữa các biến
- ❑ X và Y là cha của Z, và Y là cha của P
- ❑ Không có sự phụ thuộc giữa Z và P
- ❑ Không có vòng lặp/ chu kỳ

Bayesian Belief Network: Ví dụ



CPT: Conditional Probability Table for variable LungCancer:

	(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

Thể hiện xác suất có điều kiện đối với mỗi khả năng kết hợp lớp cha mẹ.

Tính toán xác suất mỗi kết hợp của biên **X**, từ bảng CPT:

Bayesian Belief Network

$$P(x_1,\ldots,x_n) = \prod_{i=1}^{n} P(x_i \mid Parents(Y_i))$$

Training Bayesian Networks: Một số kịch bản

- Scenario 1: Biết cấu trúc mạng và tất cả các biến: *tính bảng CPT*
- Scenario 2: Biết cấu trúc mạng, và một số biến ẩn: phương pháp *gradient descent* (leo đồi), i.e., search for a solution along the steepest descent of a criterion function
  - Trọng số được khởi tạo với các giá trị xác suất ngẫu nhiên
  - Tại mỗi vòng lặp, hướng tới giải pháp tốt nhất (bước đi ngắn nhất)
  - Trọng số được cập nhật, và tối ưu về mức cục bộ
- Scenario 3: Không biết cấu trúc mạng, tất cả các biến quan sát được: tìm kiếm thông qua không gian đồ thị/ *reconstruct network topology*
- Scenario 4: Không biết cấu trúc mạng, tất cả các biến ẩn: Không có thuật toán thỏa mãn!
- D. Heckerman. A Tutorial on Learning with Bayesian Networks. In *Learning in Graphical Models,* M. Jordan, ed.. MIT Press, 1999.

Chapter 9. Classification: Advanced Methods

- Bayesian Belief Networks
- **Classification by Backpropagation**
- Support Vector Machines
- Classification by Using Frequent Patterns
- Lazy Learners (or Learning from Your Neighbors)
- Other Classification Methods
- Additional Topics Regarding Classification
- Summary

Phân lớp bằng lan truyền ngược Backpropagation

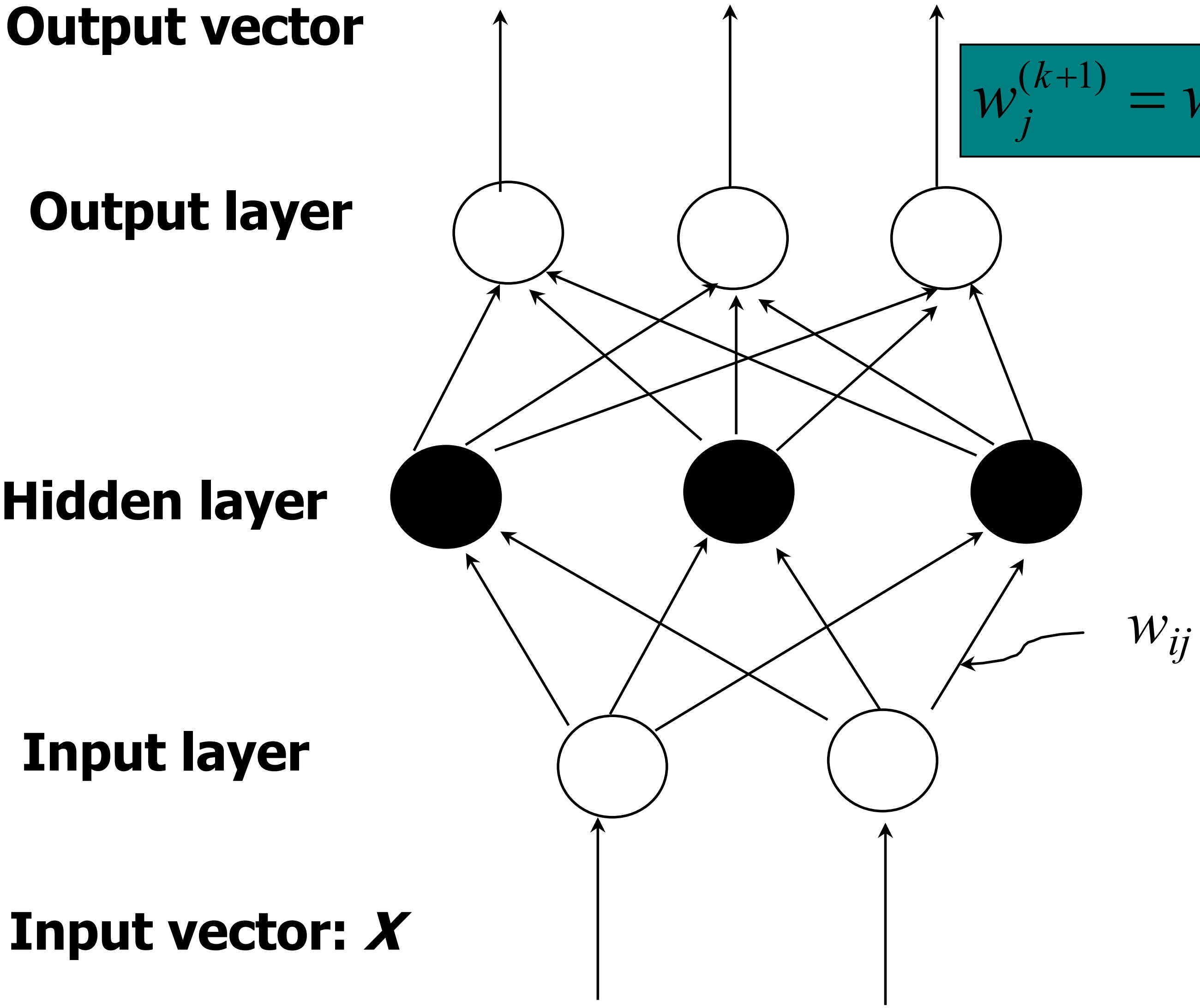
- Lan truyền ngược: Thuật toán nơ-ron, **neural network**
- Các nhà tâm lý học/ psychologists, và thần kinh học/ neurobiologists phát triển và thử nghiệm việc tính toán của mạng nơ-ron
- A neural network: tập các liên kết input/output với mỗi kết nối có trọng số/ **weight**.
- Trong quá trình học, mạng Nơ-ron học bằng cách cập nhật các trọng số/ **network learns by adjusting the weights** nhằm dự đoán đúng nhãn của dữ liệu đầu vào.
- Backpropagation còn được gọi là **connectionist learning** vì sự liên kết giữa các đơn vị.

Neural Network as a Classifier

- Điểm yếu/ Weakness
  - Thời gian huấn luyện dài/ Long training time
  - Nhiều tham số được xác định bằng kinh nghiệm, e.g., the network topology or "structure."
  - Khó giải thích: Difficult to interpret the symbolic meaning behind the learned weights and of "hidden units" in the network
- Điểm mạnh/ Strength
  - Làm việc tốt với dữ liệu nhiễu/ High tolerance to noisy data
  - Khả năng phân loại mẫu cao/ Ability to classify untrained patterns
  - Thích hợp cho giá trị liên tục
  - Thành công với các dữ liệu thực tế, e.g., hand-written letters
  - Thuật toán song song
  - Các kỹ thuật được phát triển để trích xuất các luật từ các mạng thần kinh nhân tạo.

A Multi-Layer Feed-Forward Neural Network

$$w_j^{(k+1)} = w_j^{(k)} + \lambda(y_i - \hat{y}_i^{(k)})x_{ij}$$



Output vector

Output layer

Hidden layer

Input layer

Input vector: ***X***

How A Multi-Layer Neural Network Works Cách thức hoạt động của mạng Nơ-ron đa lớp

- Đầu vào tương ứng với các thuộc tính của từng bộ dữ liệu huấn luyện
- Đầu vào được đưa đồng thời vào các node tạo nên lớp đầu vào
- Sau đó, chúng được gán trọng số và đưa vào một lớp ẩn
- Số lượng các lớp ẩn là tùy ý, thường chỉ có 1 lớp ẩn!
- Đầu ra của lớp ẩn cuối cùng là đầu vào cho các node lớp đầu ra, lớp này quyết định kết quả dự đoán cuối cùng.
- **Mạng feed-forward**: Không có trọng số nào quay trở lại node đầu vào hoặc node đầu ra của lớp trước đó.
- Từ quan điểm thống kê, các mạng thực hiện **nonlinear regression**: Cung cấp đủ các node ẩn và các mẫu huấn luyện, chúng có thể gần đúng với bất kỳ bài toán nào.

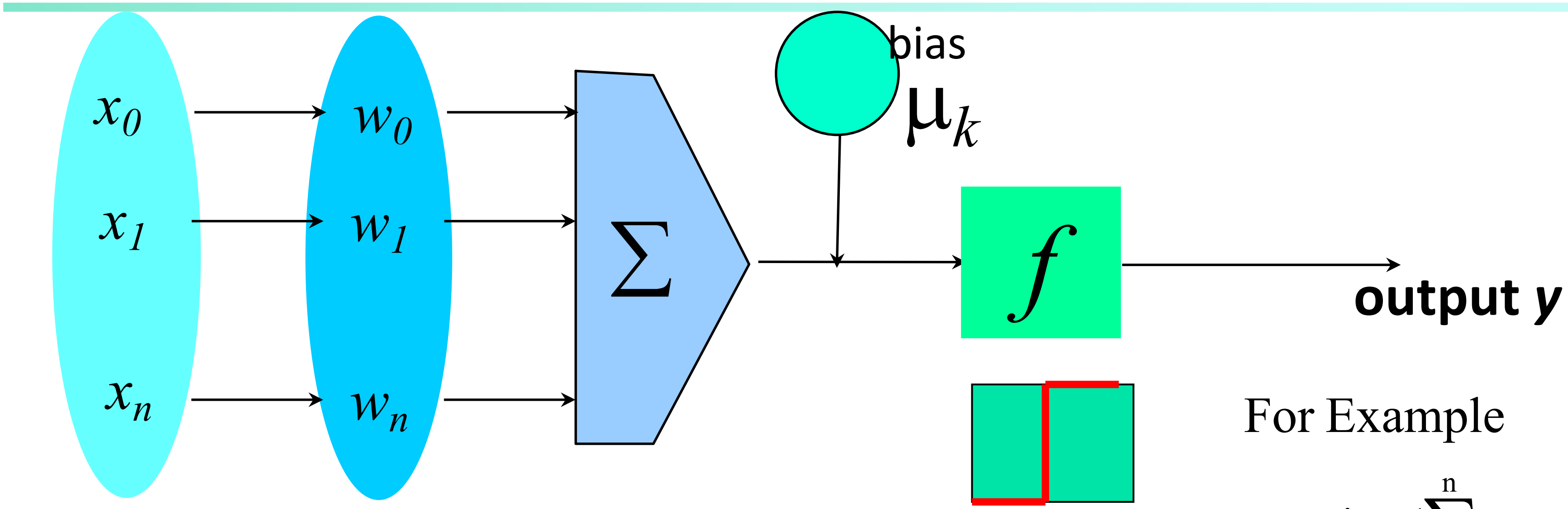
Xác định cấu trúc mạng
Network Topology

- **Network topology:** Specify # of units in the *input layer*, # of *hidden layers* (if > 1), # of units in *each hidden layer*, and # of units in the *output layer*
- Chuẩn hóa giá trị đầu vào của các biến, [0.0—1.0]
- One **input** unit per domain value, each initialized to 0
- **Output**, với bài toán phân loại đa lớp, sử dung mỗi đầu ra tương ứng với mỗi lớp
- Khi mạng được huấn luyện và độ chính xác chưa đạt, **unacceptable**, quá trình huấn luyện được lặp lại với *different network topology* hoặc *different set of initial weights*

Lan truyền ngược/ Backpropagation

- Lặp lại quá trình huấn luyện và so sánh kết quả dự đoán với kết quả thực tế
- Với mỗi dữ liệu huấn luyện, tối thiểu lỗi/ **minimize the mean squared error** giữa kết quả dự đoán và giá trị thật
- Thực hiện cập nhật trong quá trình truyền ngược/ "**backwards**": từ lớp output, thông qua các lớp ẩn đến lớp input đầu tiên, gọi là lan truyền ngược/ "**backpropagation**"
- Các bước thực hiện
  - Khởi tạo trọng số với các giá trị ngẫu nhiên (đủ nhỏ), và biases
  - Tổng hợp các đầu vào (bởi hàm activation function)
  - Lan truyền ngược sai số (cập nhật trọng số/ weights và biases)
  - Điều kiện dừng (khi sai số đủ nhỏ, etc.)

Neuron: A Hidden/Output Layer Unit



Input vector x

weight vector w

weighted sum

Activation function

For Example

$$y = \text{sign}\left(\sum_{i=0}^{n} w_i x_i - \mu_k\right)$$

- An *n*-dimensional input vector **x** is mapped into variable y by means of the scalar product and a nonlinear function mapping
- The inputs to unit are outputs from the previous layer. They are multiplied by their corresponding weights to form a weighted sum, which is added to the bias associated with unit. Then a nonlinear activation function is applied to it.

Hiệu suất và khả năng giải thích

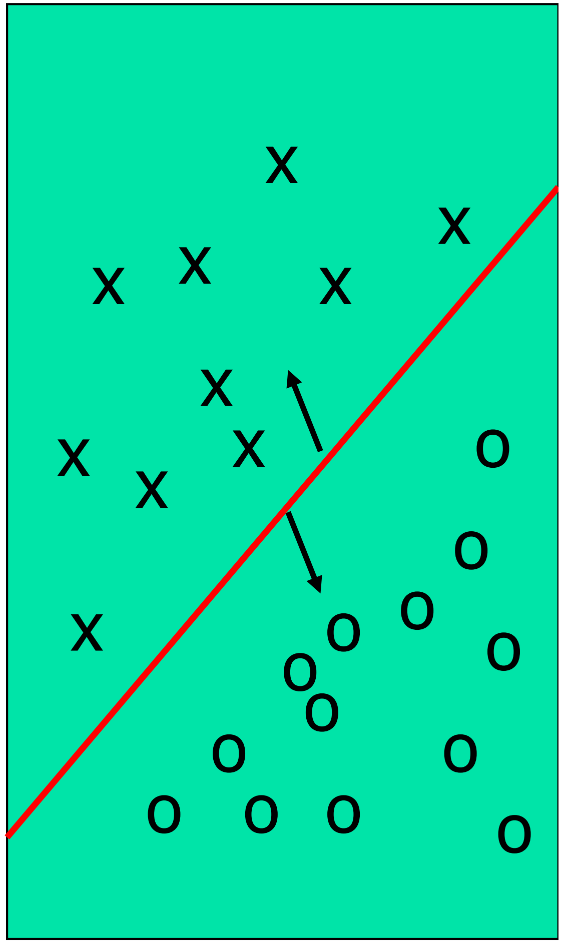
- **Efficiency** of backpropagation: Each epoch (one iteration through the training set) takes O(|D| * *w*), with |D| tuples and *w* weights, but # of epochs can be exponential to n, the number of inputs, in worst case
- For easier comprehension: **Rule extraction** by network pruning
  - Simplify the network structure by removing weighted links that have the least effect on the trained network
  - Then perform link, unit, or activation value clustering
  - The set of input and activation values are studied to derive rules describing the relationship between the input and hidden unit layers
- **Sensitivity analysis**: assess the impact that a given input variable has on a network output. The knowledge gained from this analysis can be represented in rules

Chapter 9. Classification: Advanced Methods

- Bayesian Belief Networks
- Classification by Backpropagation
- **Support Vector Machines**
- Classification by Using Frequent Patterns
- Lazy Learners (or Learning from Your Neighbors)
- Other Classification Methods
- Additional Topics Regarding Classification
- Summary

Phân lớp: A Mathematical Mapping

- Classification: predicts categorical class labels
  - E.g., Personal homepage classification
    - $x_i = (x_1, x_2, x_3, \ldots), y_i = +1 \text{ or } -1$
    - x_1 : # of word "homepage"
    - x_2 : # of word "welcome"
- Định nghĩa: $x \in X = \Re^n, y \in Y = \{+1, -1\}$,
  - Tìm hàm số f: $X \rightarrow Y$
- Phân lớp tuyến tính
  - Phân lớp nhị phân
  - Dữ liệu phía trên đường đỏ, thuộc lớp 'x'
  - Dữ liệu phía dưới đường đỏ, thuộc lớp 'o'
  - Ví dụ: SVM, Perceptron, Probabilistic Classifiers



Discriminative Classifiers

- Advantages
  - Prediction accuracy is generally high
    - As compared to Bayesian methods – in general
  - Robust, works when training examples contain errors
  - Fast evaluation of the learned target function
    - Bayesian networks are normally slow
- Criticism
  - Long training time
  - Difficult to understand the learned function (weights)
    - Bayesian networks can be used easily for pattern discovery
  - Not easy to incorporate domain knowledge
    - Easy in the form of priors on the data or distributions

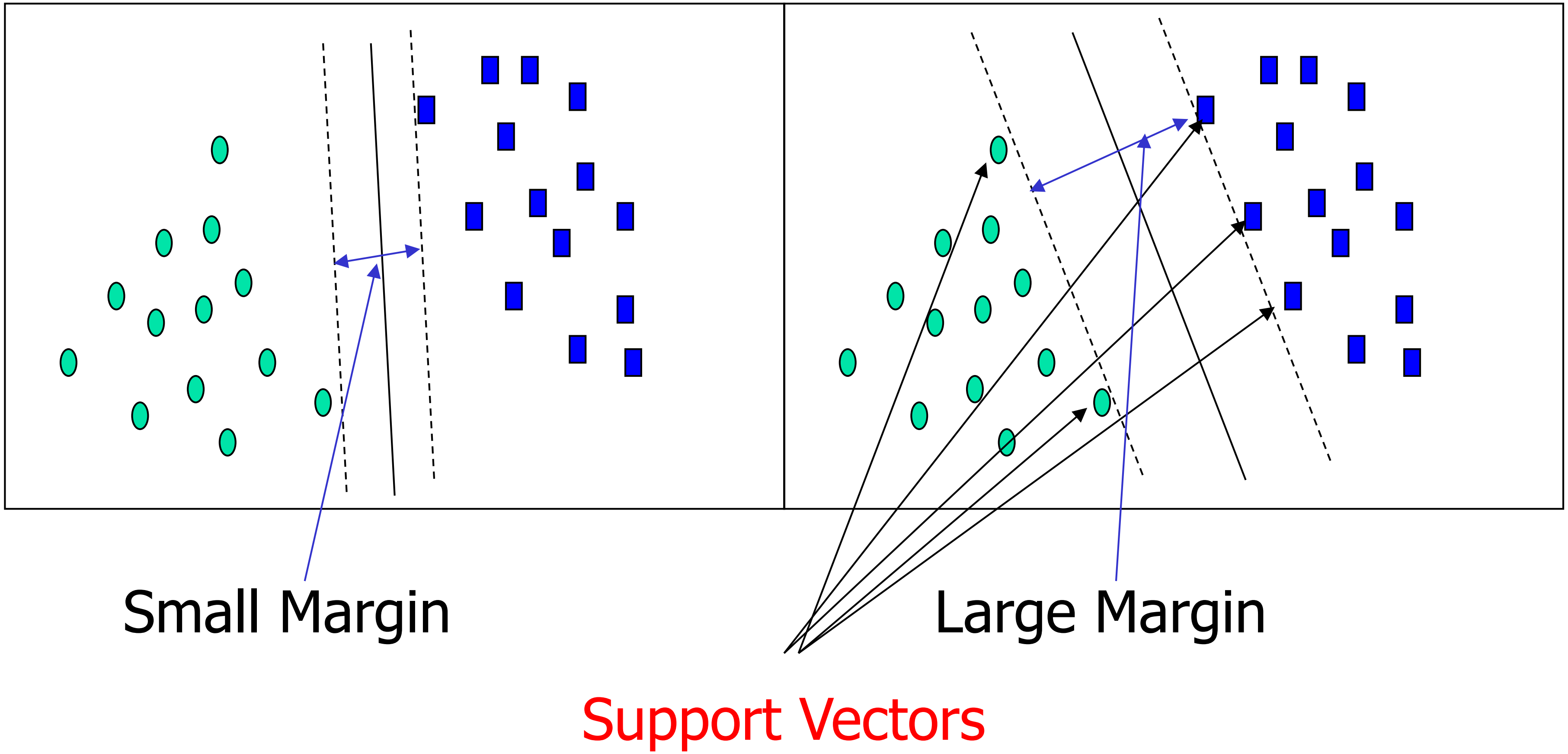
SVM—Support Vector Machines

- Phương pháp phân lớp cho cả dữ liệu tuyến tính/ linear và phi tuyến tính/ nonlinear.
- Sử dụng nonlinear mapping chuyển đổi tập huấn luyện gốc vào không gian nhiều chiều hơn!
- Trong không gian mới, có thể tìm được siêu phẳng tối ưu **hyperplane** (i.e., “decision boundary”)
- With an appropriate nonlinear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane
- SVM xác định hyperplane thông qua **support vectors** (“essential” training tuples) và **margins** (defined by the support vectors)

SVM—History và Applications

- Vapnik and colleagues (1992)—groundwork from Vapnik & Chervonenkis' statistical learning theory in 1960s
- Features: training can be slow but accuracy is high owing to their ability to model complex nonlinear decision boundaries (margin maximization)
- Used for: classification and numeric prediction
- Ứng dụng:
  - handwritten digit recognition, object recognition, speaker identification, benchmarking time-series prediction tests

SVM—General Philosophy

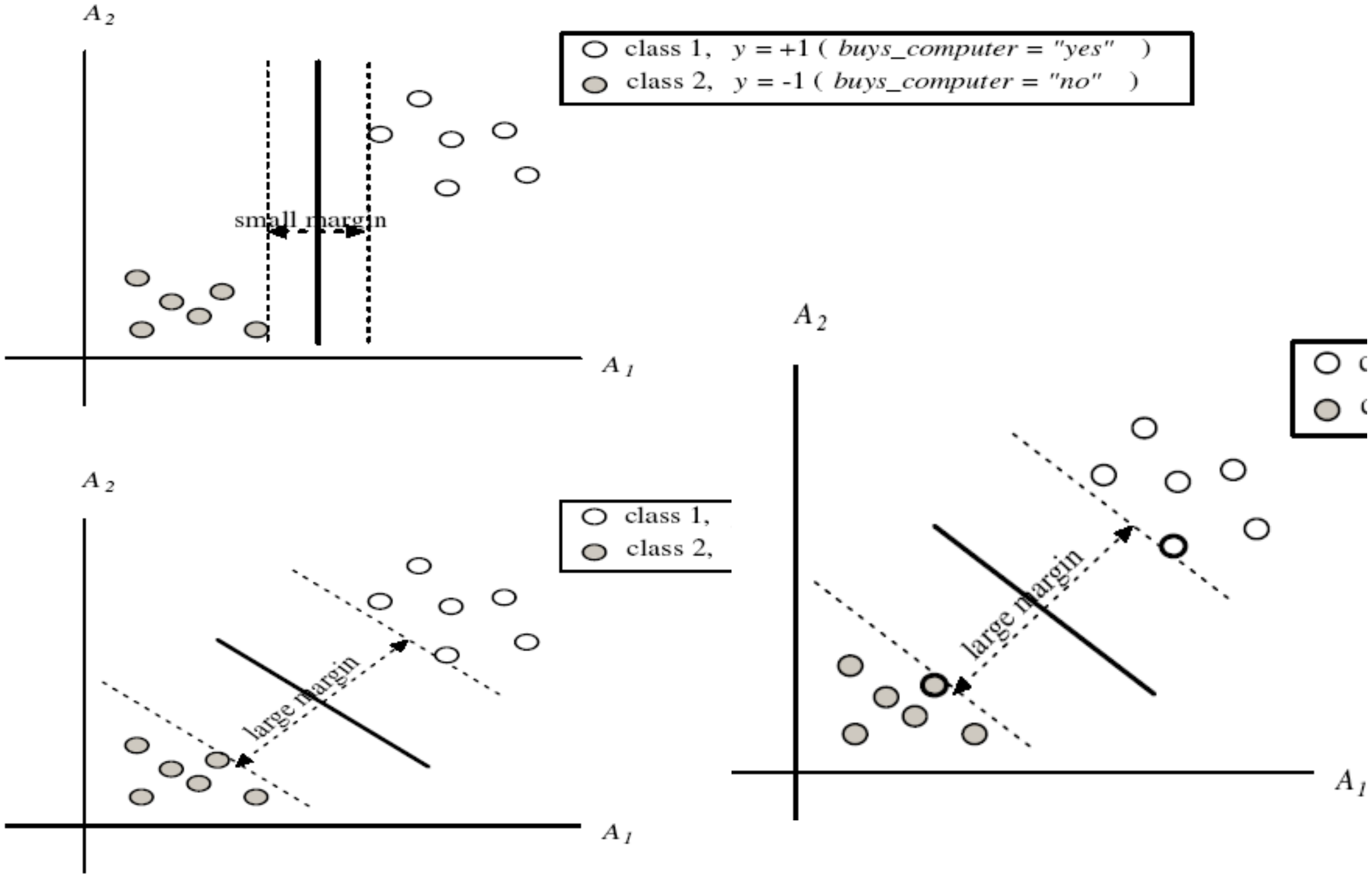


Small Margin

Large Margin

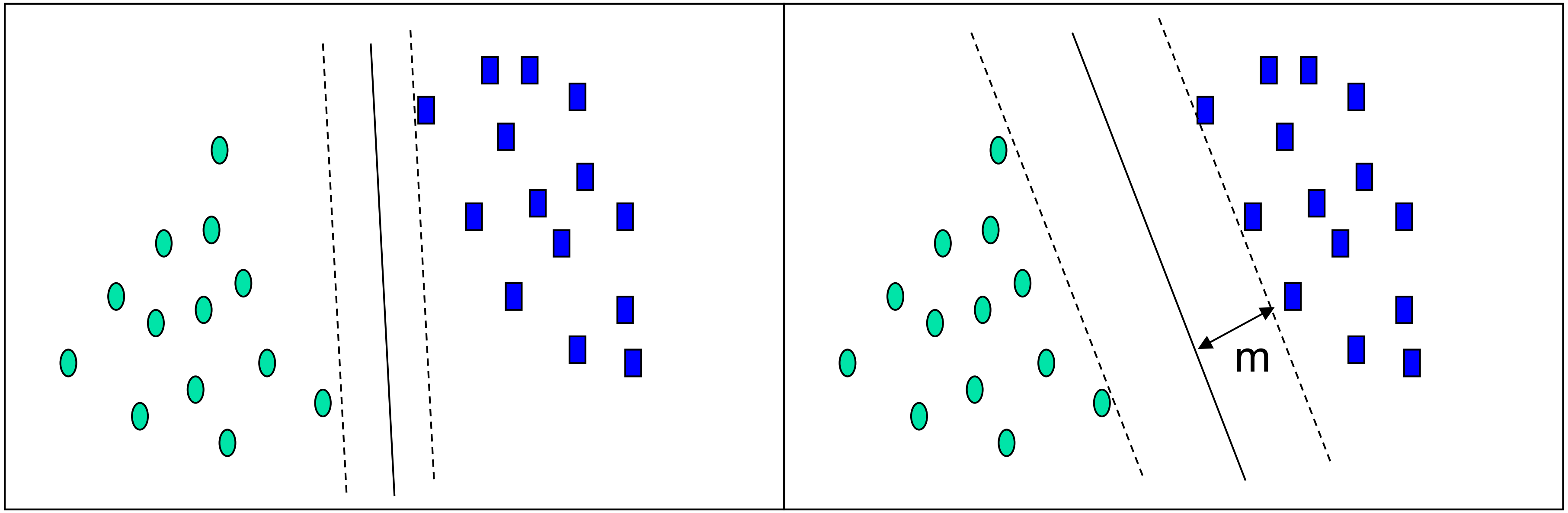
Support Vectors

SVM—Margins và Support Vectors



A 2
A 1
class 1, y = +1 (buys_computer = "yes")
class 2, y = -1 (buys_computer = "no")
small margin
A 2
A 1
class 1,
class 2,
large margin
A 2
A 1
large margin

SVM—Khi dữ liệu phân biệt tuyến tính



Let data D be ($\mathbf{X}_1$, y_1), …, ($\mathbf{X}_{|D|}$, $y_{|D|}$), where $\mathbf{X}_i$ is the set of training tuples associated with the class labels y_i

There are infinite lines (hyperplanes) separating the two classes but we want to find the best one (the one that minimizes classification error on unseen data)

SVM searches for the hyperplane with the largest margin, i.e., **maximum marginal hyperplane** (MMH)

SVM—Phân biệt tuyến tính

- Một siêu mặt phẳng/ hyperplane có thể được biểu diễn:

  $\mathbf{W} \bullet \mathbf{X} + b = 0$

  Trong đó, **W**={w_1, w_2, …, w_n} là một vecto trọng số và b là một hằng số (bias)

- Với không gian 2chiều, nó có thể được viết lại:

  $w_0 + w_1 x_1 + w_2 x_2 = 0$

- Siêu mặt phẳng/ hyperplane định nghĩa 2 miền:

  H_1: $w_0 + w_1 x_1 + w_2 x_2 \geq 1$ với $y_i = +1$, và

  H_2: $w_0 + w_1 x_1 + w_2 x_2 \leq -1$ với $y_i = -1$

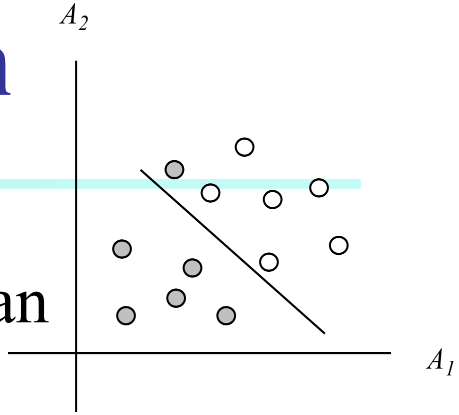
- Any training tuples that fall on hyperplanes H_1 or H_2 (i.e., the sides defining the margin) are **support vectors**

- This becomes a **constrained (convex) quadratic optimization** problem: Quadratic objective function and linear constraints → *Quadratic Programming (QP)* → Lagrangian multipliers

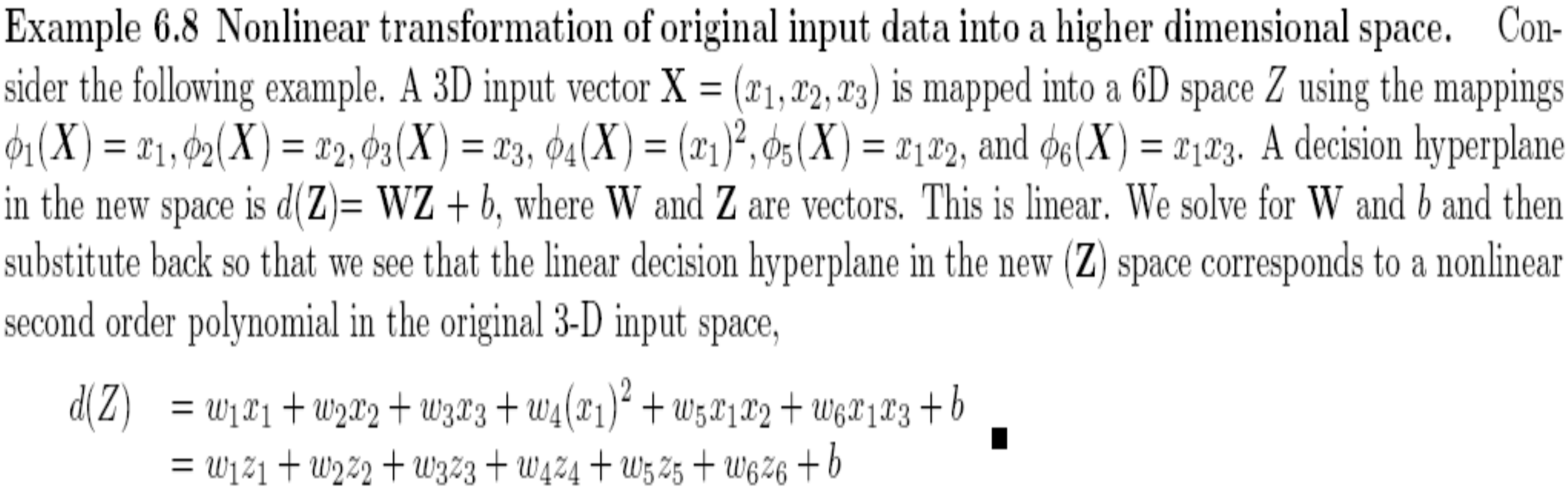
Why Is SVM Effective on High Dimensional Data?
Tại sao SVM hiệu quả với dữ liệu nhiều chiều?

- The **complexity** of trained classifier is characterized by the # of support vectors rather than the dimensionality of the data
- The **support vectors** are the essential or critical training examples —they lie closest to the decision boundary (MMH)
- If all other training examples are removed and the training is repeated, the same separating hyperplane would be found
- The number of support vectors found can be used to compute an (upper) bound on the expected error rate of the SVM classifier, which is independent of the data dimensionality
- Thus, an SVM with a small number of support vectors can have good generalization, even when the dimensionality of the data is high

SVM—Không phân biệt tuyến tính



- Chuyển đổi dữ liệu huấn luyện (gốc) sang không gian nhiều chiều hơn, nơi có phân biệt tuyến tính!



Example 6.8 Nonlinear transformation of original input data into a higher dimensional space. Consider the following example. A 3D input vector $\mathbf{X} = (x_1, x_2, x_3)$ is mapped into a 6D space Z using the mappings $\phi_1(X) = x_1, \phi_2(X) = x_2, \phi_3(X) = x_3$, $\phi_4(X) = (x_1)^2, \phi_5(X) = x_1x_2$, and $\phi_6(X) = x_1x_3$. A decision hyperplane in the new space is $d(\mathbf{Z}) = \mathbf{WZ} + b$, where $\mathbf{W}$ and $\mathbf{Z}$ are vectors. This is linear. We solve for $\mathbf{W}$ and b and then substitute back so that we see that the linear decision hyperplane in the new ($\mathbf{Z}$) space corresponds to a nonlinear second order polynomial in the original 3-D input space,

$$\begin{aligned} d(Z) &= w_1x_1 + w_2x_2 + w_3x_3 + w_4(x_1)^2 + w_5x_1x_2 + w_6x_1x_3 + b \qquad \blacksquare \\ &= w_1z_1 + w_2z_2 + w_3z_3 + w_4z_4 + w_5z_5 + w_6z_6 + b \end{aligned}$$

- Tìm kiếm siêu mặt phẳng phân biệt tuyến tính/ hyperplane trong không gian mới.

SVM: Các hàm Kernel khác nhau

- Thay vì tính toán tích vô hướng trên dữ liệu đã chuyển đổi, tương đương việc áp dung các hàm kernel K($\mathbf{X_i}$, $\mathbf{X_j}$) lên dữ liệu gốc, i.e., K($\mathbf{X_i}$, $\mathbf{X_j}$) = Φ($\mathbf{X_i}$) Φ($\mathbf{X_j}$)
- Typical Kernel Functions

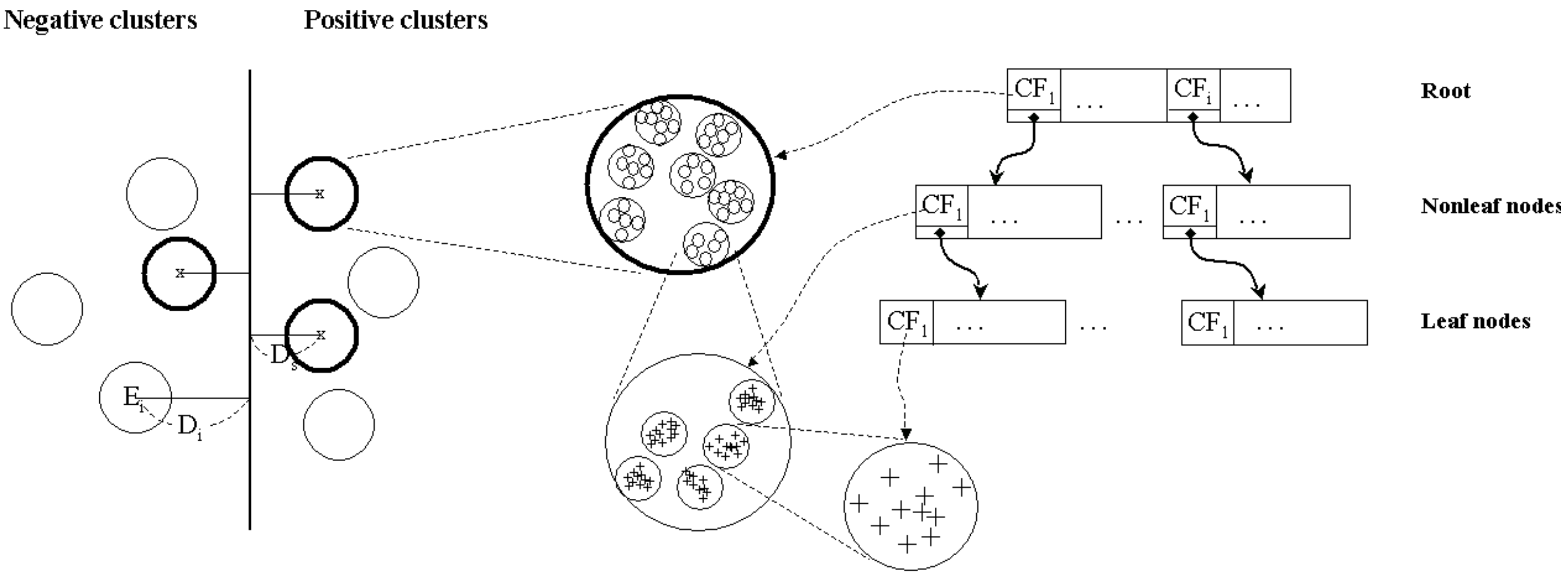
$$\begin{aligned} \textbf{Polynomial kernel of degree } h: \quad & K(\boldsymbol{X_i}, \boldsymbol{X_j}) = (\boldsymbol{X_i} \cdot \boldsymbol{X_j} + 1)^h \\ \textbf{Gaussian radial basis function kernel}: \quad & K(\boldsymbol{X_i}, \boldsymbol{X_j}) = e^{-\|\boldsymbol{X_i} - \boldsymbol{X_j}\|^2/2\sigma^2} \\ \textbf{Sigmoid kernel}: \quad & K(\boldsymbol{X_i}, \boldsymbol{X_j}) = \tanh(\kappa \boldsymbol{X_i} \cdot \boldsymbol{X_j} - \delta) \end{aligned}$$

- SVM có thể được sử dụng với bài toán đa phân lớp (> 2) và phân tích hồi qui (with additional parameters)

Scaling SVM by Hierarchical Micro-Clustering

- SVM is not scalable to the number of data objects in terms of training time and memory usage
- H. Yu, J. Yang, and J. Han, "Classifying Large Data Sets Using SVM with Hierarchical Clusters", KDD'03)
- CB-SVM (Clustering-Based SVM)
  - Given limited amount of system resources (e.g., memory), maximize the SVM performance in terms of accuracy and the training speed
  - Use micro-clustering to effectively reduce the number of points to be considered
  - At deriving support vectors, de-cluster micro-clusters near "candidate vector" to ensure high classification accuracy

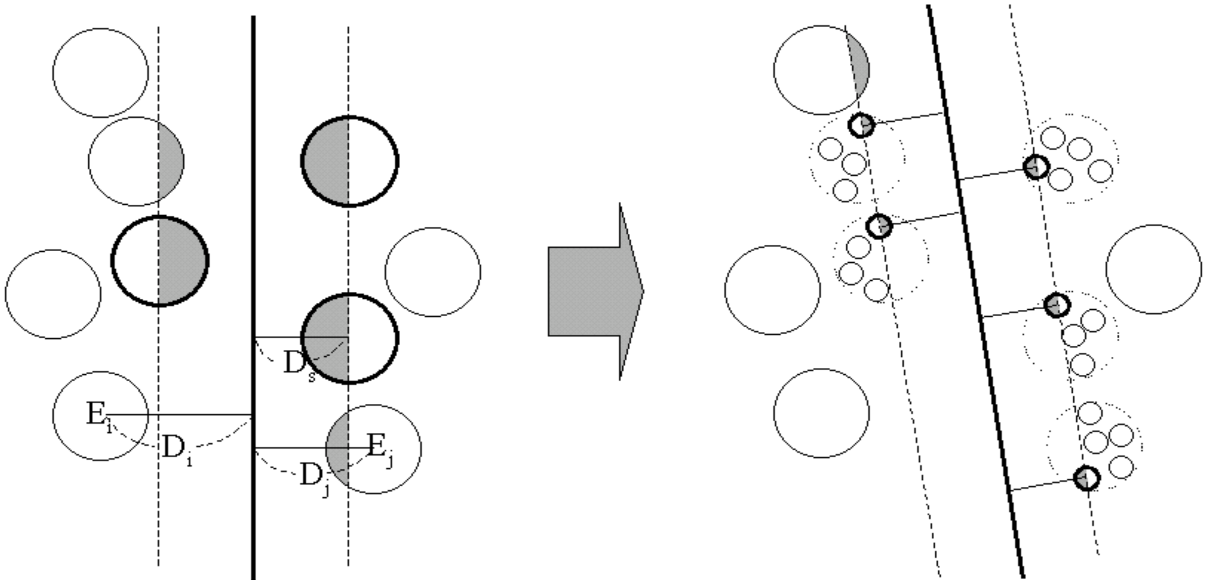
CF-Tree: Hierarchical Micro-cluster



- Read the data set once, construct a statistical summary of the data (i.e., hierarchical clusters) given a limited amount of memory
- Micro-clustering: Hierarchical indexing structure
  - provide finer samples closer to the boundary and coarser samples farther from the boundary

Selective Declustering: Ensure High Accuracy

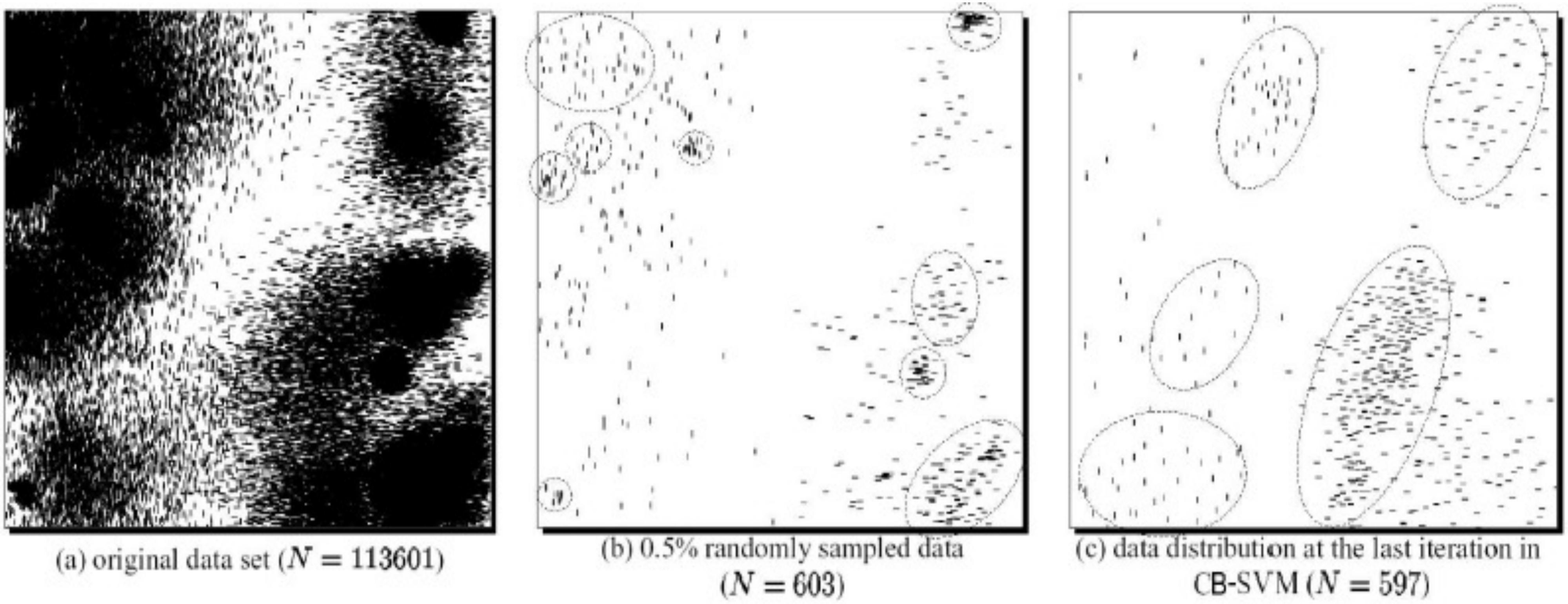
- CF tree is a suitable base structure for selective declustering
- De-cluster only the cluster E_i such that
  - $D_i - R_i < D_s$, where D_i is the distance from the boundary to the center point of E_i and R_i is the radius of E_i
  - Decluster only the cluster whose subclusters have possibilities to be the support cluster of the boundary
    - "Support cluster": The cluster whose centroid is a support vector



CB-SVM Algorithm: Outline

- Construct two CF-trees from positive and negative data sets independently
    - Need one scan of the data set
- Train an SVM from the centroids of the root entries
- De-cluster the entries near the boundary into the next level
    - The children entries de-clustered from the parent entries are accumulated into the training set with the non-declustered parent entries
- Train an SVM again from the centroids of the entries in the training set
- Repeat until nothing is accumulated

Accuracy and Scalability on Synthetic Dataset


(a) original data set ($N = 113601$)

(b) 0.5% randomly sampled data ($N = 603$)

(c) data distribution at the last iteration in CB-SVM ($N = 597$)

Figure 6: Synthetic data set in a two-dimensional space. '|': positive data; '–': negative data

- Experiments on large synthetic data sets shows better accuracy than random sampling approaches and far more scalable than the original SVM algorithm

SVM vs. Neural Network

- **SVM**
  - Deterministic algorithm
  - Nice generalization properties
  - Hard to learn – learned in batch mode using quadratic programming techniques
  - Using kernels can learn very complex functions

- **Neural Network**
  - Nondeterministic algorithm
  - Generalizes well but doesn’t have strong mathematical foundation
  - Can easily be learned in incremental fashion
  - To learn complex functions—use multilayer perceptron (nontrivial)

SVM Related Links

- SVM Website: http://www.kernel-machines.org/
- Representative implementations
  - **LIBSVM**: an efficient implementation of SVM, multi-class classifications, nu-SVM, one-class SVM, including also various interfaces with java, python, etc.
  - **SVM-light**: simpler but performance is not better than LIBSVM, support only binary classification and only in C
  - **SVM-torch**: another recent implementation also written in C

Chapter 9. Classification: Advanced Methods

- Bayesian Belief Networks
- Classification by Backpropagation
- Support Vector Machines
- **Classification by Using Frequent Patterns**
- Lazy Learners (or Learning from Your Neighbors)
- Other Classification Methods
- Additional Topics Regarding Classification
- Summary

Associative Classification

- Associative classification: Major steps
  - Mine data to find strong associations between frequent patterns (conjunctions of attribute-value pairs) and class labels
  - Association rules are generated in the form of

    $P_1 \wedge p_2 \ldots \wedge p_l \rightarrow$ "$A_{class} = C$" (conf, sup)
  - Organize the rules to form a rule-based classifier
- Why effective?
  - It explores highly confident associations among multiple attributes and may overcome some constraints introduced by decision-tree induction, which considers only one attribute at a time
  - Associative classification has been found to be often more accurate than some traditional classification methods, such as C4.5

Typical Associative Classification Methods

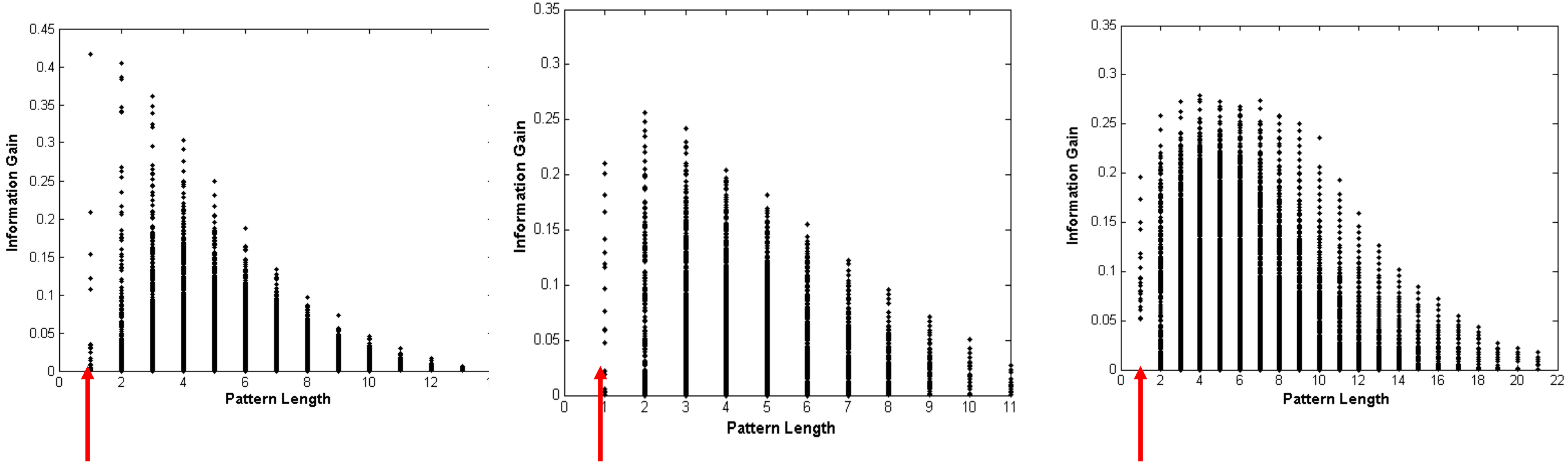
- **CBA** (Classification Based on Associations: Liu, Hsu & Ma, KDD'98)
  - Mine possible association rules in the form of
    - Cond-set (a set of attribute-value pairs) → class label
  - Build classifier: Organize rules according to decreasing precedence based on confidence and then support
- **CMAR** (Classification based on Multiple Association Rules: Li, Han, Pei, ICDM'01)
  - Classification: Statistical analysis on multiple rules
- **CPAR** (Classification based on Predictive Association Rules: Yin & Han, SDM'03)
  - Generation of predictive rules (FOIL-like analysis) but allow covered rules to retain with reduced weight
  - Prediction using best k rules
  - High efficiency, accuracy similar to CMAR

Frequent Pattern-Based Classification

- H. Cheng, X. Yan, J. Han, and C.-W. Hsu, “Discriminative Frequent Pattern Analysis for Effective Classification”, ICDE'07
- Accuracy issue
  - Increase the discriminative power
  - Increase the expressive power of the feature space
- Scalability issue
  - It is computationally infeasible to generate all feature combinations and filter them with an information gain threshold
  - Efficient method (DDPMine: FPtree pruning): H. Cheng, X. Yan, J. Han, and P. S. Yu, "Direct Discriminative Pattern Mining for Effective Classification", ICDE'08

Frequent Pattern vs. Single Feature

The discriminative power of some frequent patterns is higher than that of single features.



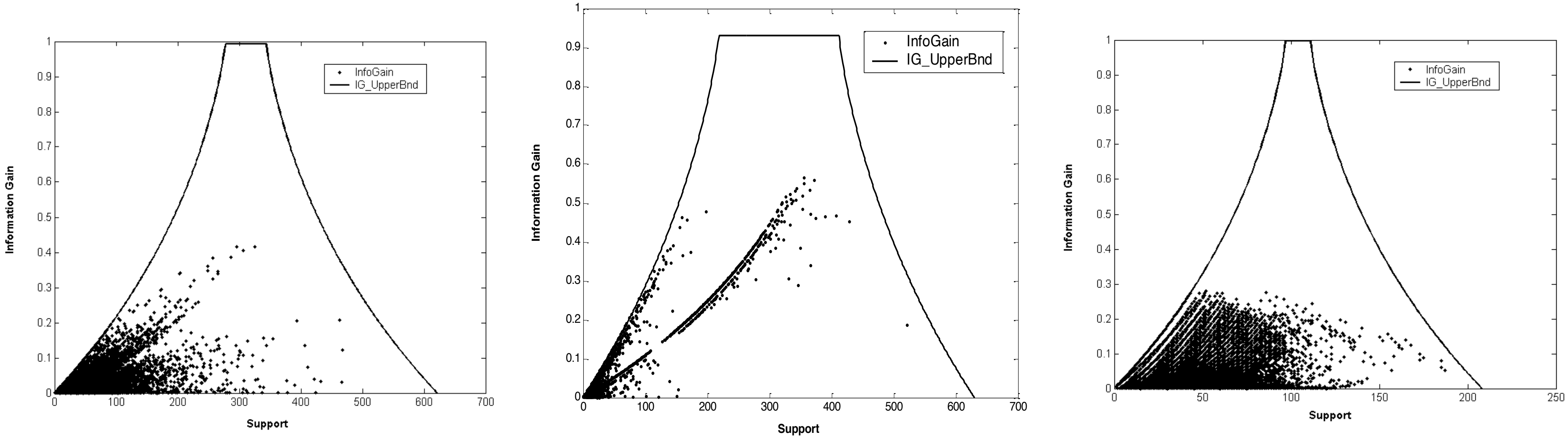
(a) Austral

(b) Cleve

(c) Sonar

Fig. 1. Information Gain vs. Pattern Length

Empirical Results



(a) Austral

(b) Breast

(c) Sonar

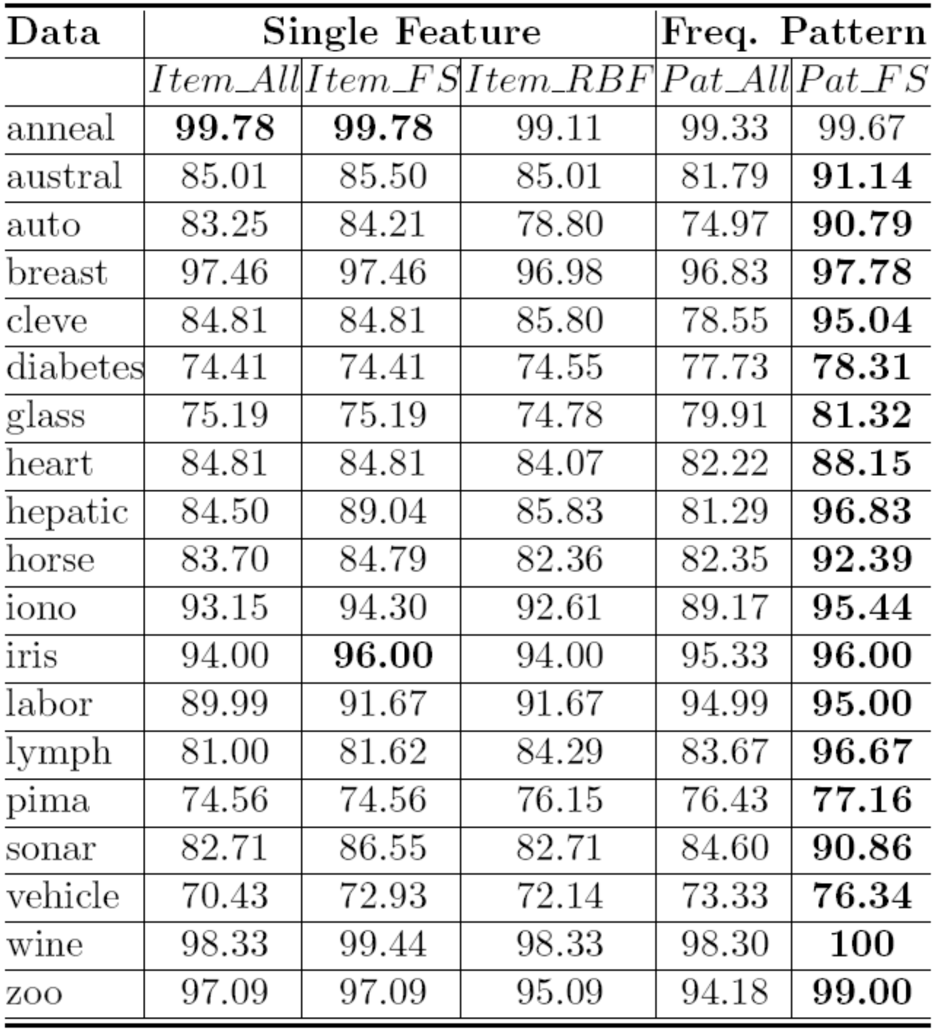
Fig. 2. Information Gain vs. Pattern Frequency

Feature Selection

- Given a set of frequent patterns, both non-discriminative and redundant patterns exist, which can cause overfitting
- We want to single out the discriminative patterns and remove redundant ones
- The notion of Maximal Marginal Relevance (MMR) is borrowed
  - A document has high marginal relevance if it is both relevant to the query and contains minimal marginal similarity to previously selected documents

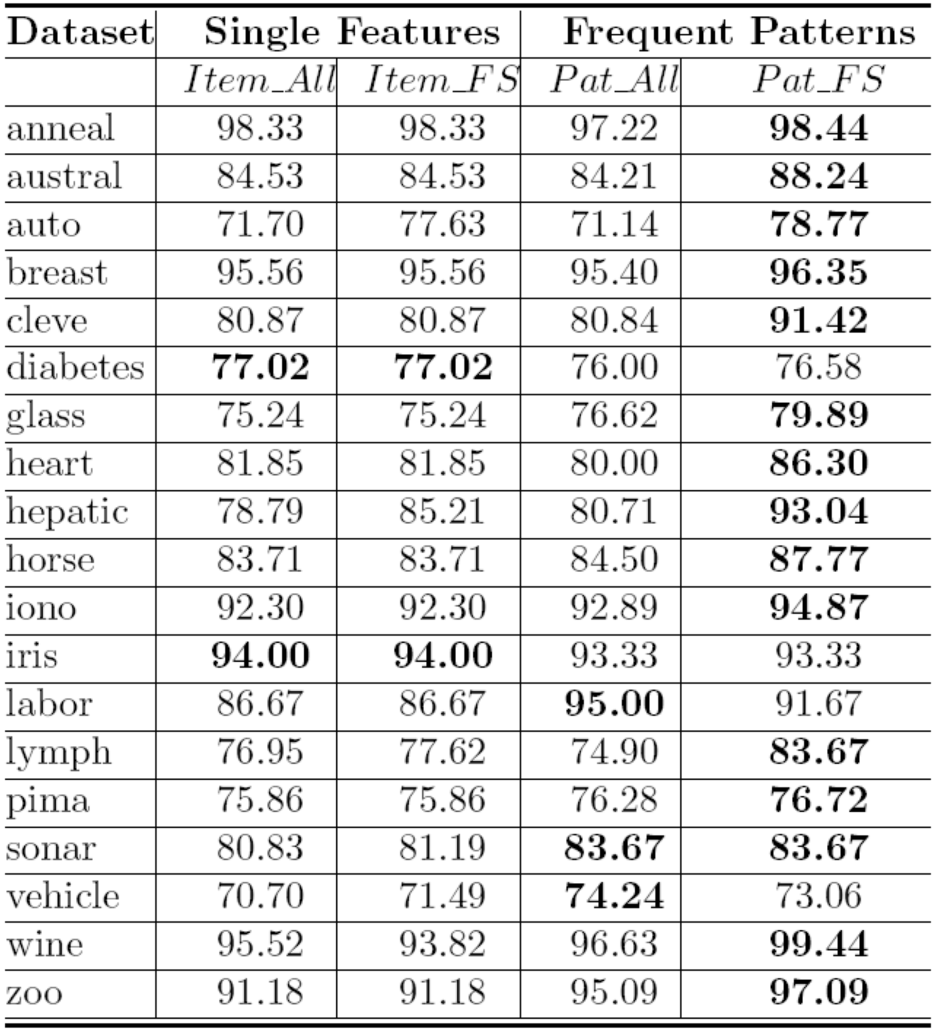
Experimental Results

Table 1. Accuracy by SVM on Frequent Combined Features vs. Single Features



Data	Single Feature			Freq. Pattern	
	Item_All	*Item_FS*	*Item_RBF*	*Pat_All*	*Pat_FS*
anneal	**99.78**	**99.78**	99.11	99.33	99.67
austral	85.01	85.50	85.01	81.79	**91.14**
auto	83.25	84.21	78.80	74.97	**90.79**
breast	97.46	97.46	96.98	96.83	**97.78**
cleve	84.81	84.81	85.80	78.55	**95.04**
diabetes	74.41	74.41	74.55	77.73	**78.31**
glass	75.19	75.19	74.78	79.91	**81.32**
heart	84.81	84.81	84.07	82.22	**88.15**
hepatic	84.50	89.04	85.83	81.29	**96.83**
horse	83.70	84.79	82.36	82.35	**92.39**
iono	93.15	94.30	92.61	89.17	**95.44**
iris	94.00	**96.00**	94.00	95.33	**96.00**
labor	89.99	91.67	91.67	94.99	**95.00**
lymph	81.00	81.62	84.29	83.67	**96.67**
pima	74.56	74.56	76.15	76.43	**77.16**
sonar	82.71	86.55	82.71	84.60	**90.86**
vehicle	70.43	72.93	72.14	73.33	**76.34**
wine	98.33	99.44	98.33	98.30	**100**
zoo	97.09	97.09	95.09	94.18	**99.00**

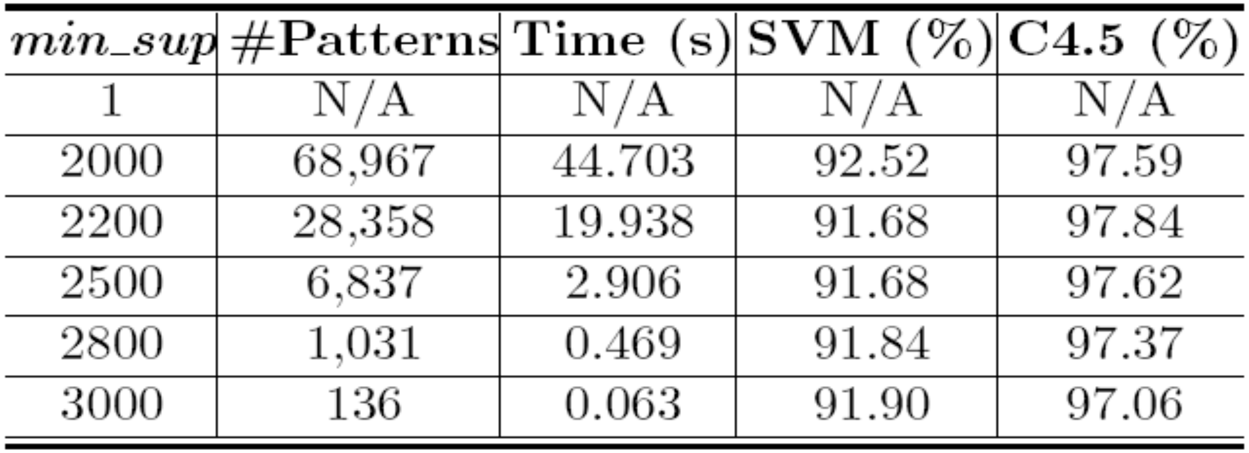
Table 2. Accuracy by C4.5 on Frequent Combined Features vs. Single Features



Dataset	Single Features		Frequent Patterns	
	Item_All	*Item_FS*	*Pat_All*	*Pat_FS*
anneal	98.33	98.33	97.22	**98.44**
austral	84.53	84.53	84.21	**88.24**
auto	71.70	77.63	71.14	**78.77**
breast	95.56	95.56	95.40	**96.35**
cleve	80.87	80.87	80.84	**91.42**
diabetes	**77.02**	**77.02**	76.00	76.58
glass	75.24	75.24	76.62	**79.89**
heart	81.85	81.85	80.00	**86.30**
hepatic	78.79	85.21	80.71	**93.04**
horse	83.71	83.71	84.50	**87.77**
iono	92.30	92.30	92.89	**94.87**
iris	**94.00**	**94.00**	93.33	93.33
labor	86.67	86.67	**95.00**	91.67
lymph	76.95	77.62	74.90	**83.67**
pima	75.86	75.86	76.28	**76.72**
sonar	80.83	81.19	**83.67**	**83.67**
vehicle	70.70	71.49	**74.24**	73.06
wine	95.52	93.82	96.63	**99.44**
zoo	91.18	91.18	95.09	**97.09**

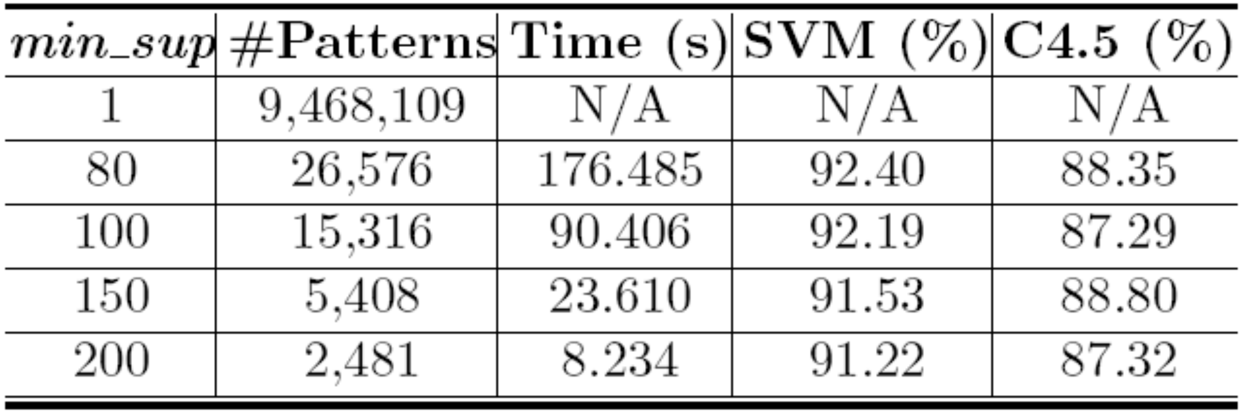
Scalability Tests

Table 3. Accuracy & Time on Chess Data



min_sup	#Patterns	Time (s)	SVM (%)	C4.5 (%)
1	N/A	N/A	N/A	N/A
2000	68,967	44.703	92.52	97.59
2200	28,358	19.938	91.68	97.84
2500	6,837	2.906	91.68	97.62
2800	1,031	0.469	91.84	97.37
3000	136	0.063	91.90	97.06

Table 4. Accuracy & Time on Waveform Data



min_sup	#Patterns	Time (s)	SVM (%)	C4.5 (%)
1	9,468,109	N/A	N/A	N/A
80	26,576	176.485	92.40	88.35
100	15,316	90.406	92.19	87.29
150	5,408	23.610	91.53	88.80
200	2,481	8.234	91.22	87.32

DDPMine: Branch-and-Bound Search

$$\sup(child) \le \sup(parent)$$

$$\sup(b) \le \sup(a)$$

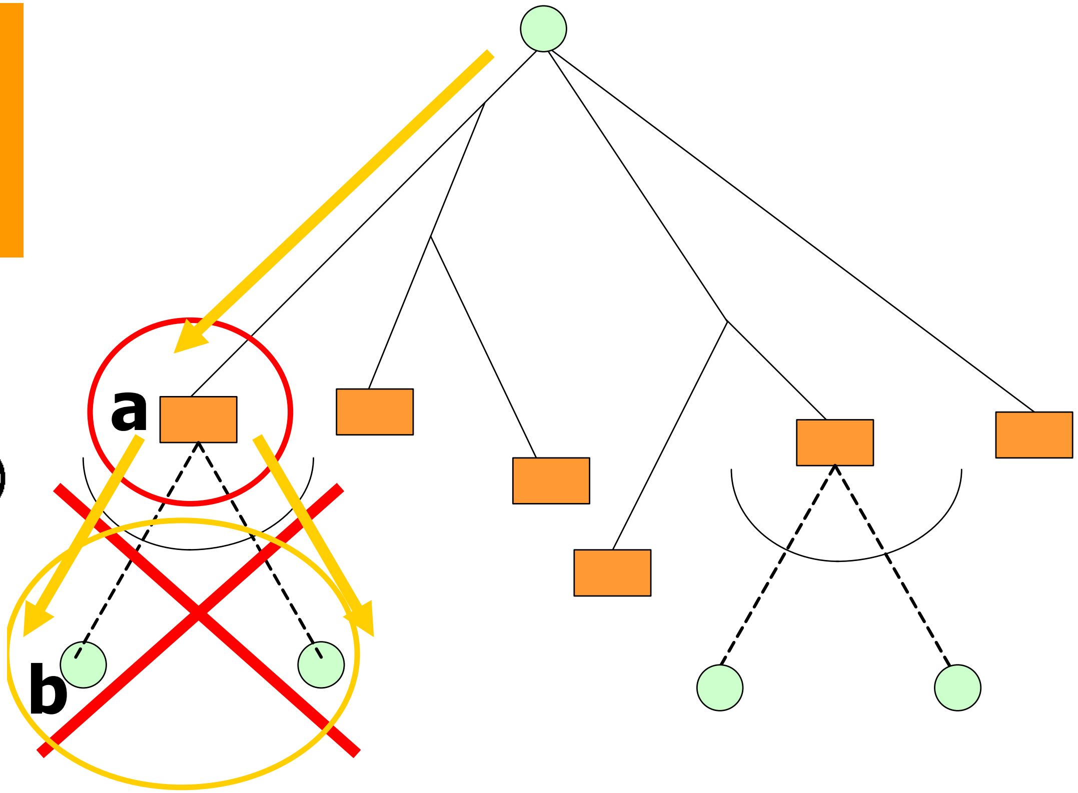
maximize $IG(C|b)$

subject to

$min\_sup \le sup(b) \le sup(a)$

$0 \le sup_{+}(b) \le sup_{+}(a)$

$0 \le sup_{-}(b) \le sup_{-}(a)$

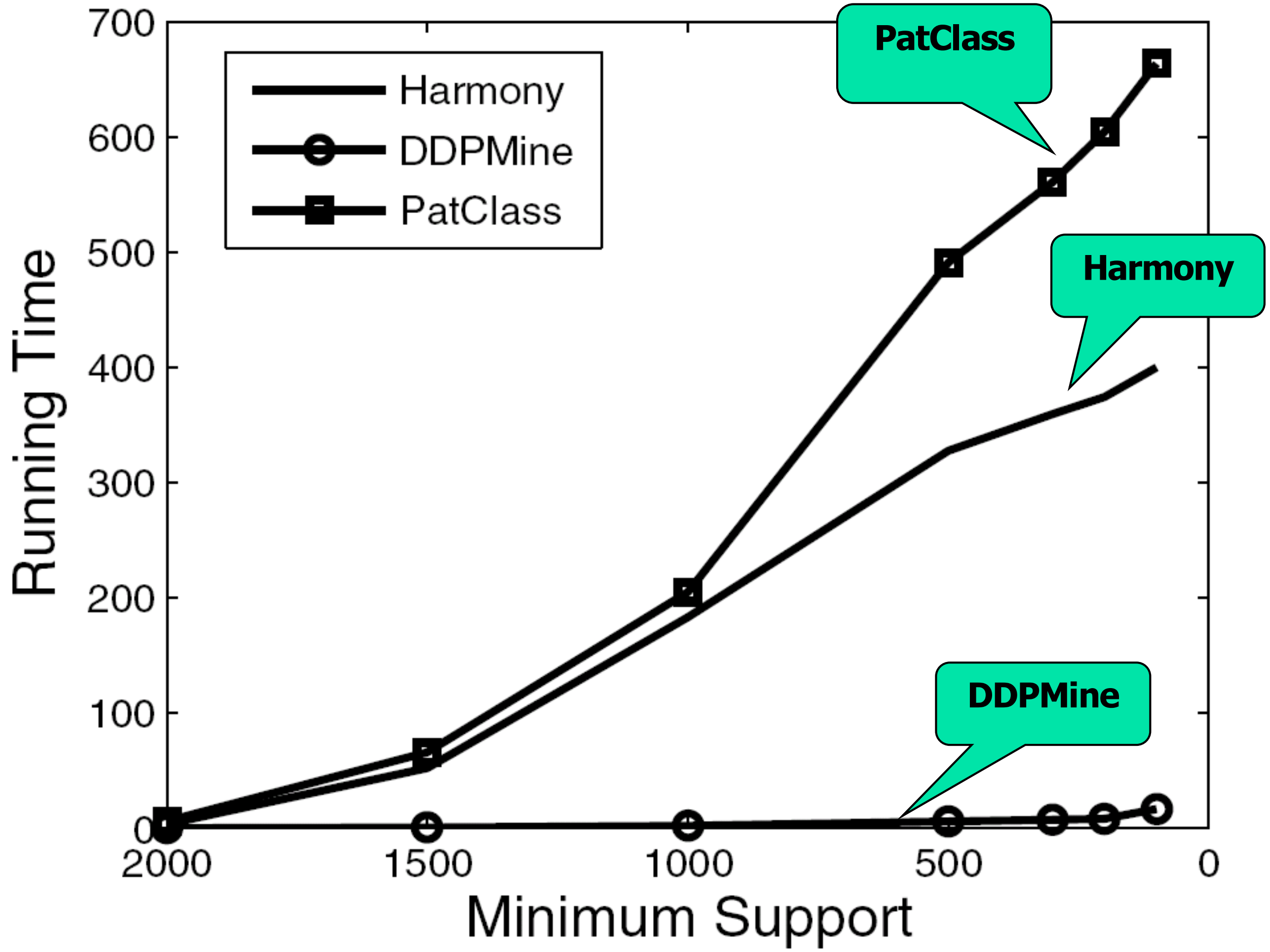


a: constant, a parent node

b: variable, a descendent

Association between information gain and frequency

DDPMine Efficiency: Runtime



PatClass: ICDE'07 Pattern Classification Alg.

Chapter 9. Classification: Advanced Methods

- Bayesian Belief Networks
- Classification by Backpropagation
- Support Vector Machines
- Classification by Using Frequent Patterns
- **Lazy Learners (or Learning from Your Neighbors)**
- Other Classification Methods
- Additional Topics Regarding Classification
- Summary

Lazy vs. Eager Learning

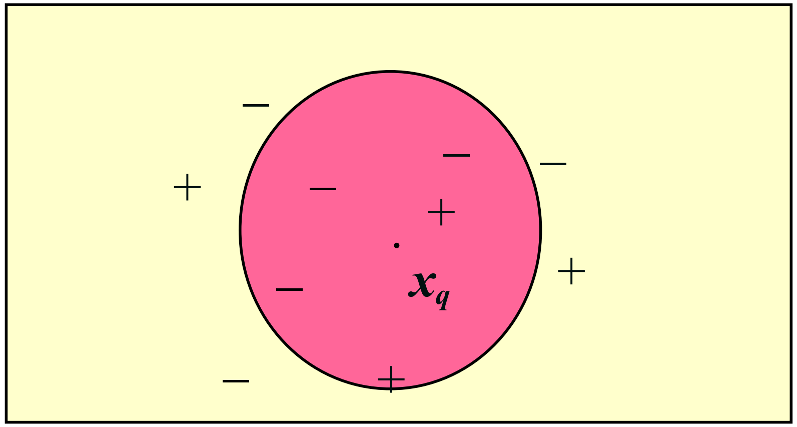
- Lazy vs. eager learning
  - **Lazy learning** (e.g., instance-based learning): Chỉ cần lưu trữ dữ liệu huấn luyện (or only minor processing) và đợi cho đến khi có dữ liệu kiểm tra.
  - **Eager learning** (the above discussed methods): Cho tập dữ liệu huấn luyện, tạo mô hình phân lớp trước khi nhận dữ liệu mới (e.g., test) để phân loại
- Lazy: tốn ít thời gian huấn luyện nhưng mất nhiều thời gian dự đoán
- Độ chính xác
  - Phương pháp học 'lười biếng' sử dụng hiệu quả không gian giả thuyết vì nó sử dụng nhiều hàm tuyến tính cục bộ để tạo thành một xấp xỉ toàn cục cho hàm mục tiêu
  - Eager: Phải cam kết với một giả thuyết duy nhất bao trùm toàn bộ không gian thể hiện

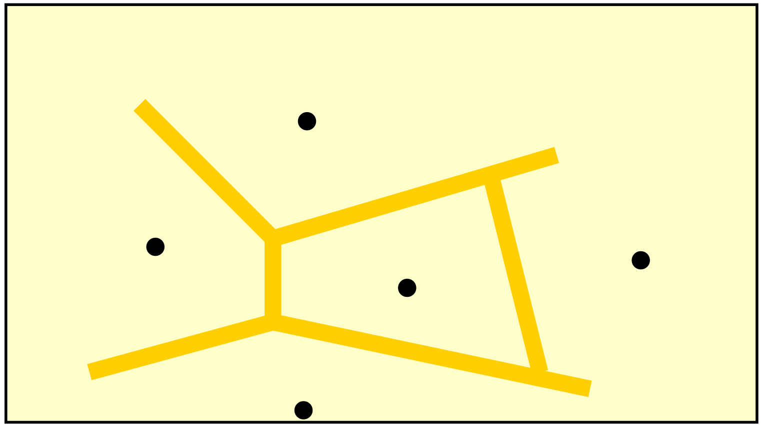
Lazy Learner: Instance-Based Methods

- Học dựa trên điểm dữ liệu/ Instance-based learning:
  - Store training examples and delay the processing ("lazy evaluation") until a new instance must be classified
- Các phương pháp điển hình
  - *k*-nearest neighbor approach
    - Instances represented as points in a Euclidean space.
  - Locally weighted regression
    - Constructs local approximation
  - Case-based reasoning
    - Uses symbolic representations and knowledge-based inference

Thuật toán k-NN

- Tất cả điểm dữ liệu tương ứng với không gian n chiều/ n-D
- Điểm gần nhất được định nghĩa trong không gian Euclidean, dist($\mathbf{X}_1$, $\mathbf{X}_2$)
- Hàm mục tiêu có thể là rời rạc hoặc liên tục
- Với gía trị rời rạc, *k*-NN trả về kết quả là k điểm dữ liệu gần nhất so với x_q
- Biểu đồ Vonoroi: bề mặt quyết định gây ra bởi 1-NN cho một tập mẫu huấn luyện





Thuật toán *k*-NN

- *k*-NN với dự đoán giá trị liên tục
  - Trả về kết quả là trung bình của *k* điểm lân cận gần nhất
- Thuật toán k-NN có trọng số khoảng cách
  - Weight the contribution of each of the *k* neighbors according to their distance to the query x_q
    - Give greater weight to closer neighbors

$$w \equiv \frac{1}{d(x_q, x_i)^2}$$

- Robust to noisy data by averaging *k*-nearest neighbors
- Curse of dimensionality: distance between neighbors could be dominated by irrelevant attributes
  - To overcome it, axes stretch or elimination of the least relevant attributes

Case-Based Reasoning (CBR)

- **CBR**: Uses a database of problem solutions to solve new problems
- Store symbolic description (tuples or cases)—not points in a Euclidean space
- Applications: Customer-service (product-related diagnosis), legal ruling
- Methodology
  - Instances represented by rich symbolic descriptions (e.g., function graphs)
  - Search for similar cases, multiple retrieved cases may be combined
  - Tight coupling between case retrieval, knowledge-based reasoning, and problem solving
- Challenges
  - Find a good similarity metric
  - Indexing based on syntactic similarity measure, and when failure, backtracking, and adapting to additional cases

Chapter 9. Classification: Advanced Methods

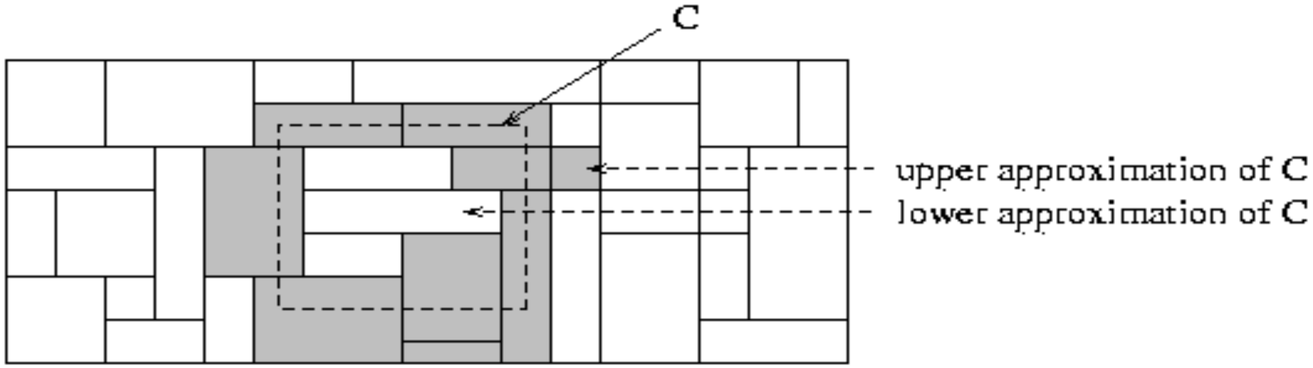
- Bayesian Belief Networks
- Classification by Backpropagation
- Support Vector Machines
- Classification by Using Frequent Patterns
- Lazy Learners (or Learning from Your Neighbors)
- **Other Classification Methods**
- Additional Topics Regarding Classification
- Summary

Genetic Algorithms (GA)

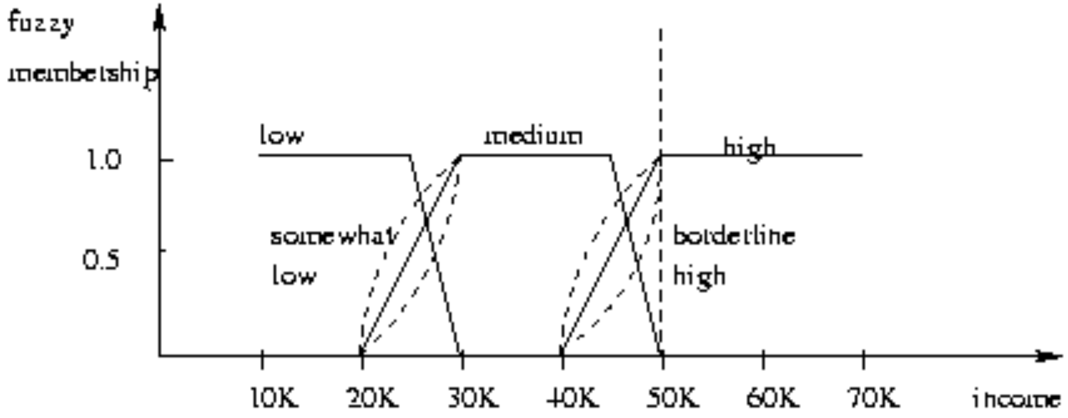
- Genetic Algorithm: based on an analogy to biological evolution
- An initial **population** is created consisting of randomly generated rules
  - Each rule is represented by a string of bits
  - E.g., if A_1 and $\neg A_2$ then C_2 can be encoded as 100
  - If an attribute has $k > 2$ values, k bits can be used
- Based on the notion of survival of the **fittest**, a new population is formed to consist of the fittest rules and their offspring
- The *fitness of a rule* is represented by its classification accuracy on a set of training examples
- Offspring are generated by *crossover* and *mutation*
- The process continues until a population P evolves *when each rule in P satisfies a prespecified threshold*
- Slow but easily parallelizable

Rough Set Approach

- Rough sets are used to **approximately or "roughly" define equivalent classes**
- A rough set for a given class C is approximated by two sets: a lower approximation (certain to be in C) and an upper approximation (cannot be described as not belonging to C)
- Finding the minimal subsets (**reducts**) of attributes for feature reduction is NP-hard but a **discernibility matrix** (which stores the differences between attribute values for each pair of data tuples) is used to reduce the computation intensity



Fuzzy Set Approaches



- Fuzzy logic uses truth values between 0.0 and 1.0 to represent the degree of membership (such as in a *fuzzy membership graph*)
- Attribute values are converted to fuzzy values. Ex.:
    - Income, *x*, is assigned a fuzzy membership value to each of the discrete categories {low, medium, high}, e.g. $49K belongs to "medium income" with fuzzy value 0.15 but belongs to "high income" with fuzzy value 0.96
    - Fuzzy membership values do not have to sum to 1.
- Each applicable rule contributes a vote for membership in the categories
- Typically, the truth values for each predicted category are summed, and these sums are combined

Chapter 9. Classification: Advanced Methods

- Bayesian Belief Networks
- Classification by Backpropagation
- Support Vector Machines
- Classification by Using Frequent Patterns
- Lazy Learners (or Learning from Your Neighbors)
- Other Classification Methods
- **Additional Topics Regarding Classification**
- Summary

Multiclass Classification

- Classification involving more than two classes (i.e., > 2 Classes)
- Method 1. **One-vs.-all** (OVA): Learn a classifier one at a time
  - Given m classes, train m classifiers: one for each class
  - Classifier j: treat tuples in class j as *positive* & all others as *negative*
  - To classify a tuple **X**, the set of classifiers vote as an ensemble
- Method 2. **All-vs.-all** (AVA): Learn a classifier for each pair of classes
  - Given m classes, construct m(m-1)/2 binary classifiers
  - A classifier is trained using tuples of the two classes
  - To classify a tuple **X**, each classifier votes. X is assigned to the class with maximal vote
- Comparison
  - All-vs.-all tends to be superior to one-vs.-all
  - Problem: Binary classifier is sensitive to errors, and errors affect vote count

Error-Correcting Codes for Multiclass Classification

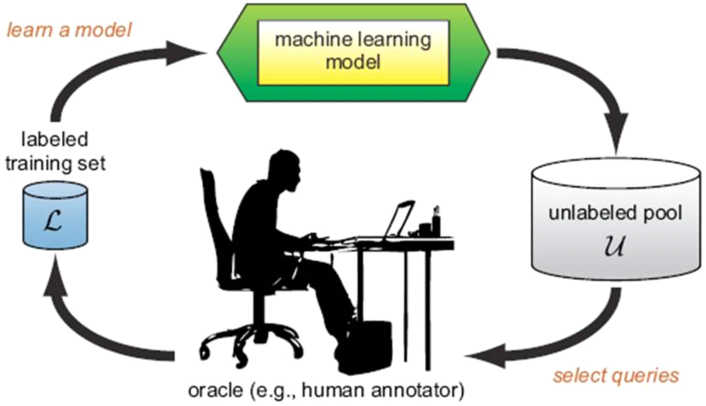
Class	Error-Corr. Codeword						
C_1	1	1	1	1	1	1	1
C_2	0	0	0	0	1	1	1
C_3	0	0	1	1	0	0	1
C_4	0	1	0	1	0	1	0

- Originally designed to correct errors during data transmission for communication tasks by exploring data redundancy
- Example
  - A 7-bit codeword associated with classes 1-4
  - Given a unknown tuple **X**, the 7-trained classifiers output: 0001010
  - Hamming distance: # of different bits between two codewords
  - H(**X**, C_1) = 5, by checking # of bits between [1111111] & [0001010]
  - H(**X**, C_2) = 3, H(**X**, C_3) = 3, H(**X**, C_4) = 1, thus C_4 as the label for **X**
- Error-correcting codes can correct up to (h-1)/h 1-bit error, where h is the minimum Hamming distance between any two codewords
- If we use 1-bit per class, it is equiv. to one-vs.-all approach, the code are insufficient to self-correct
- When selecting error-correcting codes, there should be good row-wise and col.-wise separation between the codewords

Semi-Supervised Classification

- Semi-supervised: Uses labeled and unlabeled data to build a classifier
- Self-training:
  - Build a classifier using the labeled data
  - Use it to label the unlabeled data, and those with the most confident label prediction are added to the set of labeled data
  - Repeat the above process
  - Adv: easy to understand; disadv: may reinforce errors
- Co-training: Use two or more classifiers to teach each other
  - Each learner uses a mutually independent set of features of each tuple to train a good classifier, say f_1
  - Then f_1 and f_2 are used to predict the class label for unlabeled data X
  - Teach each other: The tuple having the most confident prediction from f_1 is added to the set of labeled data for f_2, & vice versa
- Other methods, e.g., joint probability distribution of features and labels

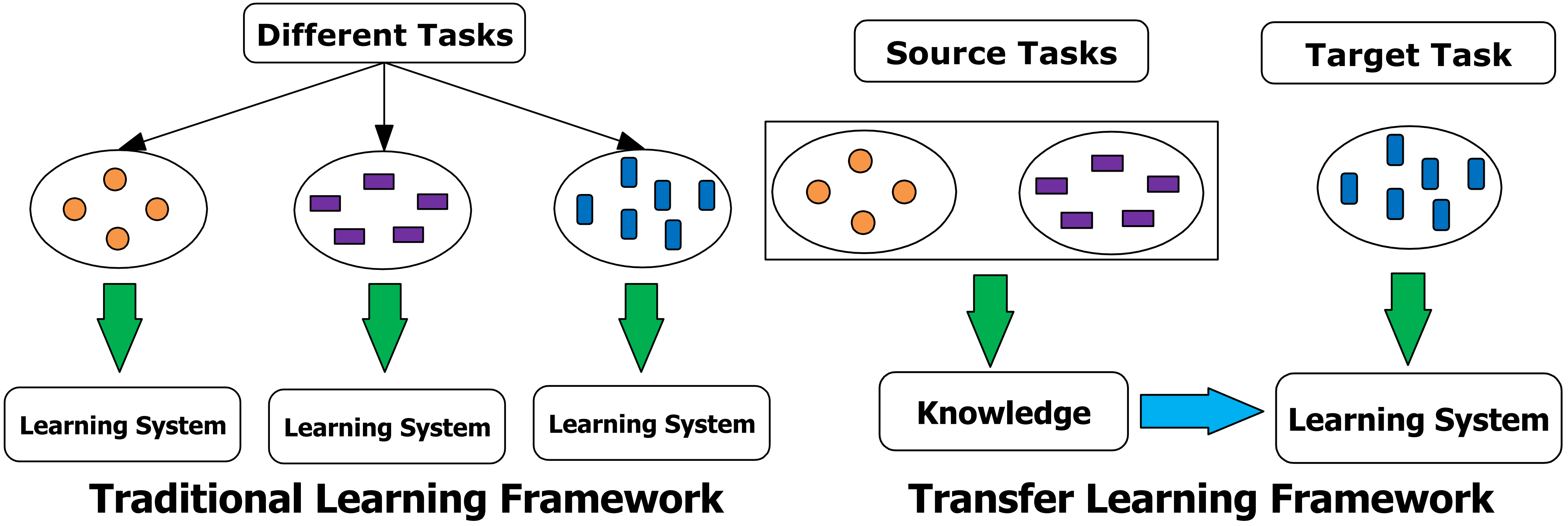
Active Learning



- Class labels are expensive to obtain
- Active learner: query human (oracle) for labels
- Pool-based approach: Uses a pool of unlabeled data
  - L: a small subset of D is labeled, U: a pool of unlabeled data in D
  - Use a query function to carefully select one or more tuples from U and request labels from an oracle (a human annotator)
  - The newly labeled samples are added to L, and learn a model
  - Goal: Achieve high accuracy using as few labeled data as possible
- Evaluated using *learning curves*: Accuracy as a function of the number of instances queried (# of tuples to be queried should be small)
- Research issue: How to choose the data tuples to be queried?
  - Uncertainty sampling: choose the least certain ones
  - Reduce *version space*, the subset of hypotheses consistent w. the training data
  - Reduce expected entropy over U: Find the greatest reduction in the total number of incorrect predictions

Transfer Learning: Conceptual Framework

- Transfer learning: Extract knowledge from one or more source tasks and apply the knowledge to a target task
- Traditional learning: Build a new classifier for each new task
- Transfer learning: Build new classifier by applying existing knowledge learned from source tasks



Transfer Learning: Methods and Applications

- Applications: Especially useful when data is outdated or distribution changes, e.g., Web document classification, e-mail spam filtering
- *Instance-based transfer learning*: Reweight some of the data from source tasks and use it to learn the target task
- TrAdaBoost (Transfer AdaBoost)
  - Assume source and target data each described by the same set of attributes (features) & class labels, but rather diff. distributions
  - Require only labeling a small amount of target data
  - Use source data in training: When a source tuple is misclassified, reduce the weight of such tupels so that they will have less effect on the subsequent classifier
- Research issues
  - Negative transfer: When it performs worse than no transfer at all
  - Heterogeneous transfer learning: Transfer knowledge from different feature space or multiple source domains
  - Large-scale transfer learning

Chapter 9. Classification: Advanced Methods

- Bayesian Belief Networks
- Classification by Backpropagation
- Support Vector Machines
- Classification by Using Frequent Patterns
- Lazy Learners (or Learning from Your Neighbors)
- Other Classification Methods
- Additional Topics Regarding Classification
- **Summary**

Summary

- Effective and advanced classification methods
  - Bayesian belief network (probabilistic networks)
  - Backpropagation (Neural networks)
  - Support Vector Machine (SVM)
  - Pattern-based classification
  - Other classification methods: lazy learners (KNN, case-based reasoning), genetic algorithms, rough set and fuzzy set approaches
- Additional Topics on Classification
  - Multiclass classification
  - Semi-supervised classification
  - Active learning
  - Transfer learning

References

- Please see the references of Chapter 8

Surplus Slides

What Is Prediction?

- (Numerical) prediction is similar to classification
  - construct a model
  - use model to predict continuous or ordered value for a given input
- Prediction is different from classification
  - Classification refers to predict categorical class label
  - Prediction models continuous-valued functions
- Major method for prediction: regression
  - model the relationship between one or more *independent* or **predictor** variables and a *dependent* or **response** variable
- Regression analysis
  - Linear and multiple regression
  - Non-linear regression
  - Other regression methods: generalized linear model, Poisson regression, log-linear models, regression trees

Linear Regression

- Linear regression: involves a response variable y and a single predictor variable x

  $$y = w_0 + w_1 x$$

  where w_0 (y-intercept) and w_1 (slope) are regression coefficients

- Method of least squares: estimates the best-fitting straight line

$$w_1 = \frac{\sum_{i=1}^{|D|}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{|D|}(x_i - \bar{x})^2} \qquad w_0 = \bar{y} - w_1 \bar{x}$$

- Multiple linear regression: involves more than one predictor variable
  - Training data is of the form ($\mathbf{X_1}$, y_1), ($\mathbf{X_2}$, y_2),…, ($\mathbf{X_{|D|}}$, $y_{|D|}$)
  - Ex. For 2-D data, we may have: $y = w_0 + w_1 x_1 + w_2 x_2$
  - Solvable by extension of least square method or using SAS, S-Plus
  - Many nonlinear functions can be transformed into the above

Nonlinear Regression

- Some nonlinear models can be modeled by a polynomial function
- A polynomial regression model can be transformed into linear regression model. For example,

  $$y = w_0 + w_1 x + w_2 x^2 + w_3 x^3$$

  convertible to linear with new variables: $x_2 = x^2$, $x_3 = x^3$

  $$y = w_0 + w_1 x + w_2 x_2 + w_3 x_3$$

- Other functions, such as power function, can also be transformed to linear model
- Some models are intractable nonlinear (e.g., sum of exponential terms)
  - possible to obtain least square estimates through extensive calculation on more complex formulae

Other Regression-Based Models

- Generalized linear model:
  - Foundation on which linear regression can be applied to modeling categorical response variables
  - Variance of y is a function of the mean value of y, not a constant
  - Logistic regression: models the prob. of some event occurring as a linear function of a set of predictor variables
  - Poisson regression: models the data that exhibit a Poisson distribution
- Log-linear models: (for categorical data)
  - Approximate discrete multidimensional prob. distributions
  - Also useful for data compression and smoothing
- Regression trees and model trees
  - Trees to predict continuous values rather than class labels

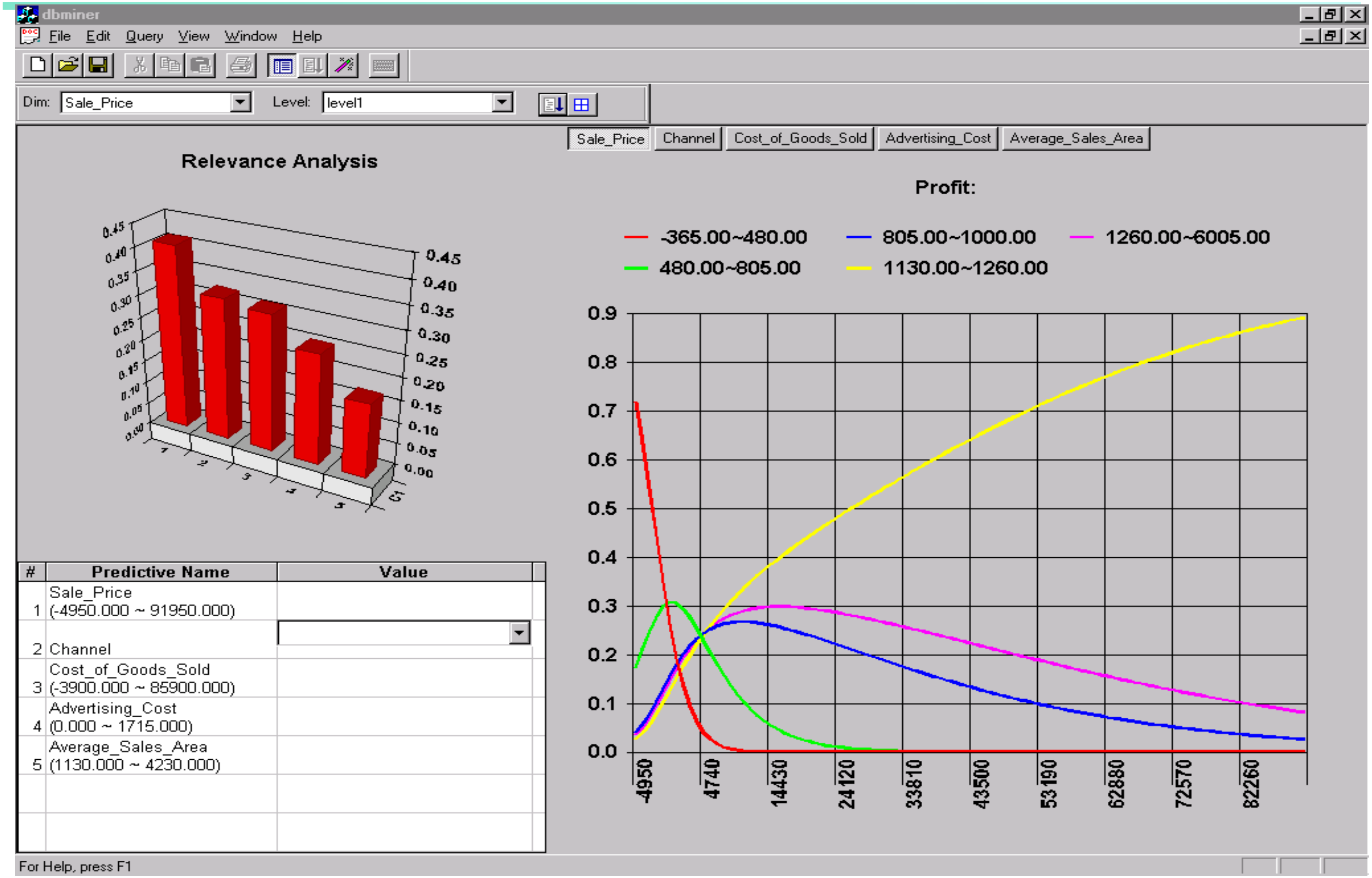
Regression Trees and Model Trees

- Regression tree: proposed in CART system (Breiman et al. 1984)
  - CART: Classification And Regression Trees
  - Each leaf stores a *continuous-valued prediction*
  - It is the *average value of the predicted attribute* for the training tuples that reach the leaf
- Model tree: proposed by Quinlan (1992)
  - Each leaf holds a regression model—a multivariate linear equation for the predicted attribute
  - A more general case than regression tree
- Regression and model trees tend to be more accurate than linear regression when the data are not represented well by a simple linear model

Predictive Modeling in Multidimensional Databases

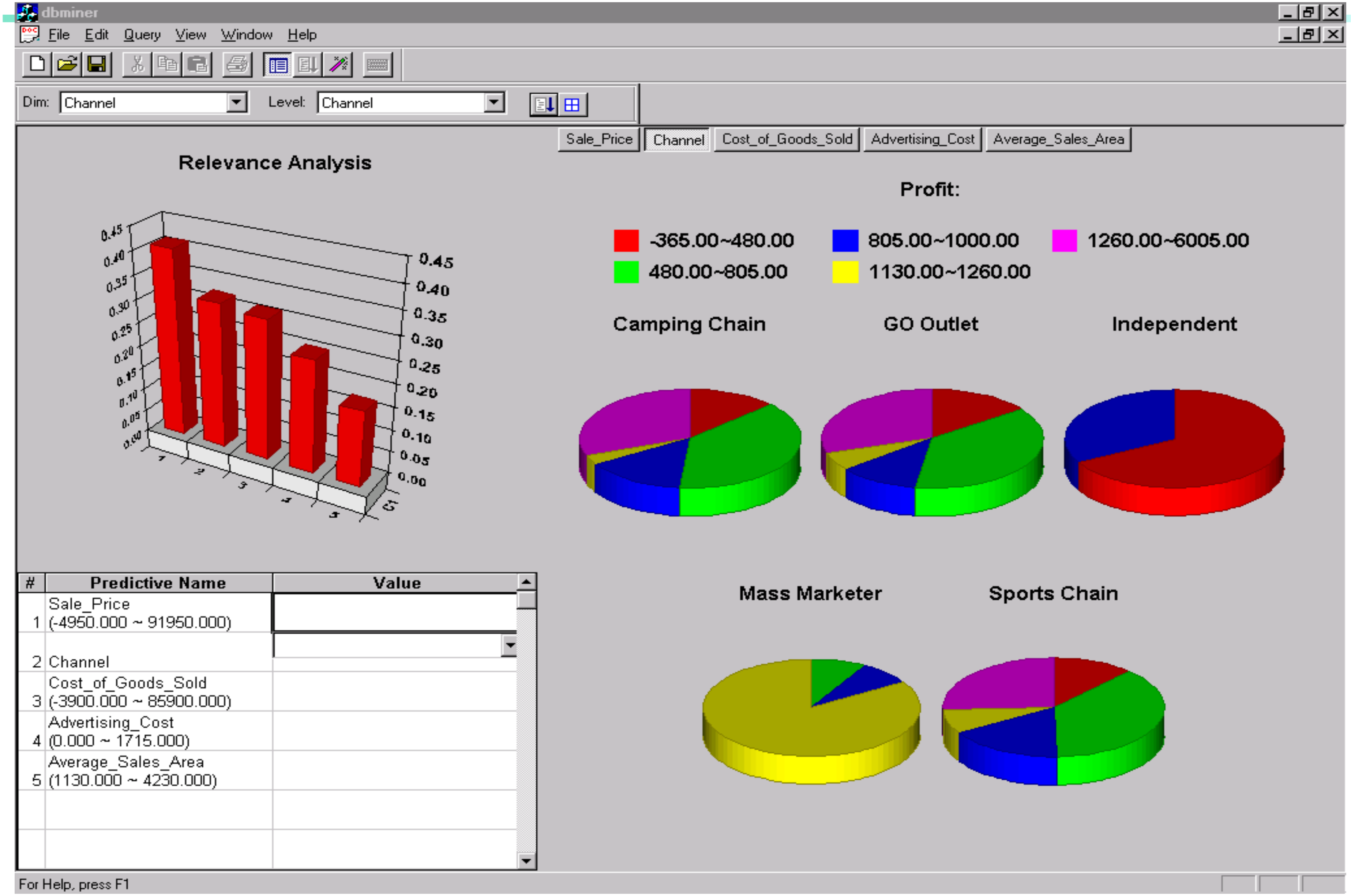
- Predictive modeling: Predict data values or construct generalized linear models based on the database data
- One can only predict value ranges or category distributions
- Method outline:
    - Minimal generalization
    - Attribute relevance analysis
    - Generalized linear model construction
    - Prediction
- Determine the major factors which influence the prediction
    - Data relevance analysis: uncertainty measurement, entropy analysis, expert judgement, etc.
- Multi-level prediction: drill-down and roll-up analysis

Prediction: Numerical Data



#	Predictive Name	Value
1	Sale_Price (-4950.000 ~ 91950.000)	
2	Channel	
3	Cost_of_Goods_Sold (-3900.000 ~ 85900.000)	
4	Advertising_Cost (0.000 ~ 1715.000)	
5	Average_Sales_Area (1130.000 ~ 4230.000)	

Prediction: Categorical Data



Dim: Channel

Level: Channel

#	Predictive Name	Value
1	Sale_Price (-4950.000 ~ 91950.000)	
2	Channel	
3	Cost_of_Goods_Sold (-3900.000 ~ 85900.000)	
4	Advertising_Cost (0.000 ~ 1715.000)	
5	Average_Sales_Area (1130.000 ~ 4230.000)	

For Help, press F1

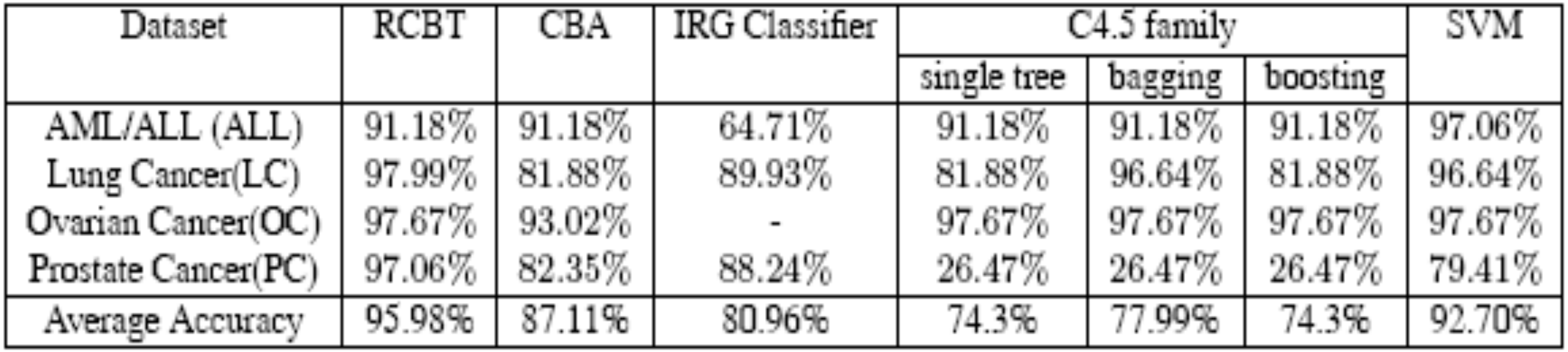
SVM—Introductory Literature

- “Statistical Learning Theory” by Vapnik: extremely hard to understand, containing many errors too.
- C. J. C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Knowledge Discovery and Data Mining*, 2(2), 1998.
  - Better than the Vapnik’s book, but still written too hard for introduction, and the examples are so not-intuitive
- The book “An Introduction to Support Vector Machines” by N. Cristianini and J. Shawe-Taylor
  - Also written hard for introduction, but the explanation about the mercer’s theorem is better than above literatures
- The neural network book by Haykins
  - Contains one nice chapter of SVM introduction

Notes about SVM— Introductory Literature

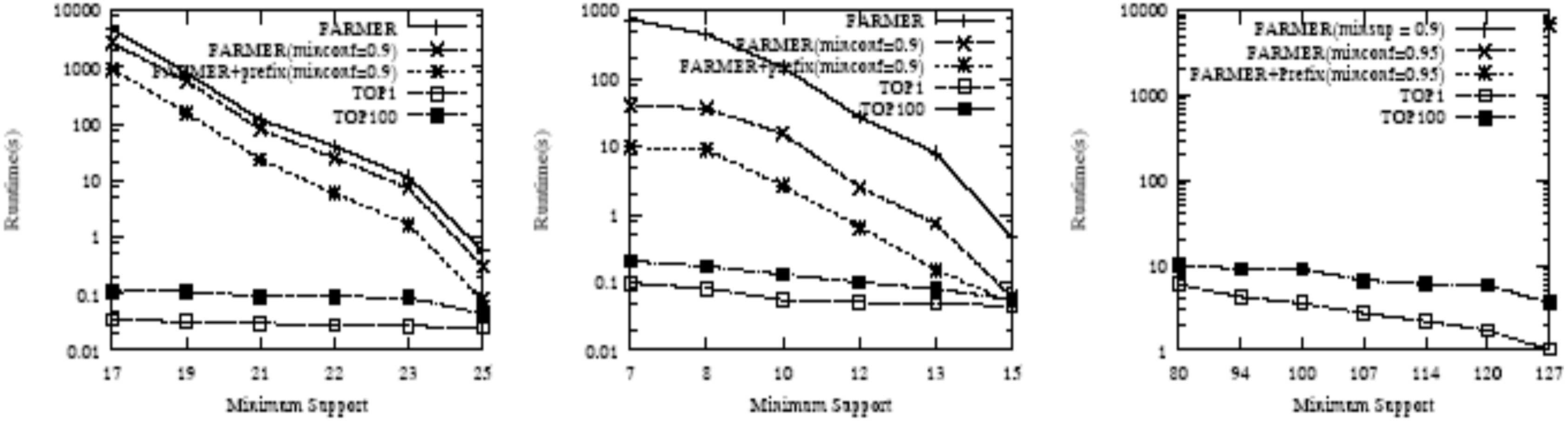
- "Statistical Learning Theory" by **Vapnik**: difficult to understand, containing many errors.
- C. J. C. **Burges**. A Tutorial on Support Vector Machines for Pattern Recognition. *Knowledge Discovery and Data Mining*, 2(2), 1998.
  - Easier than Vapnik's book, but still not introductory level; the examples are not so intuitive
- The book An Introduction to Support Vector Machines by **Cristianini and Shawe-Taylor**
  - Not introductory level, but the explanation about Mercer's Theorem is better than above literatures
- Neural Networks and Learning Machines by **Haykin**
  - Contains a nice chapter on SVM introduction

Associative Classification Can Achieve High Accuracy and Efficiency (Cong et al. SIGMOD05)



Dataset	RCBT	CBA	IRG Classifier	C4.5 family			SVM
				single tree	bagging	boosting	
AML/ALL (ALL)	91.18%	91.18%	64.71%	91.18%	91.18%	91.18%	97.06%
Lung Cancer(LC)	97.99%	81.88%	89.93%	81.88%	96.64%	81.88%	96.64%
Ovarian Cancer(OC)	97.67%	93.02%	-	97.67%	97.67%	97.67%	97.67%
Prostate Cancer(PC)	97.06%	82.35%	88.24%	26.47%	26.47%	26.47%	79.41%
Average Accuracy	95.98%	87.11%	80.96%	74.3%	77.99%	74.3%	92.70%

Table 2: Classification Results



(a) ALL-AML leukemia

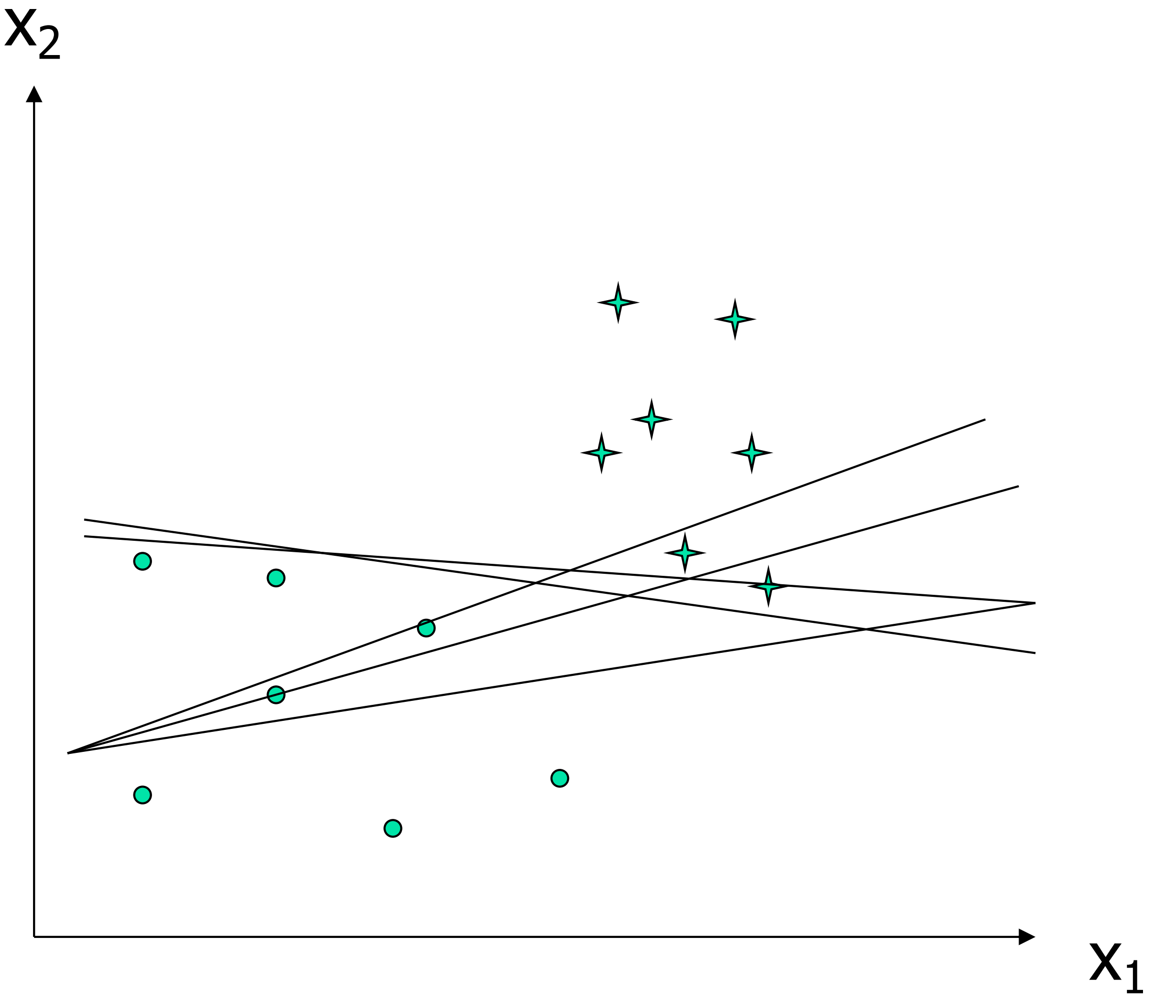
(b) Lung Cancer

(c) Ovarian Cancer

A Closer Look at CMAR

- **CMAR** (Classification based on Multiple Association Rules: Li, Han, Pei, ICDM'01)
- Efficiency: Uses an enhanced FP-tree that maintains the distribution of class labels among tuples satisfying each frequent itemset
- Rule pruning whenever a rule is inserted into the tree
  - Given two rules, R_1 and R_2, if the antecedent of R_1 is more general than that of R_2 and $\text{conf}(R_1) \geq \text{conf}(R_2)$, then prune R_2
  - Prunes rules for which the rule antecedent and class are not positively correlated, based on a χ^2 test of statistical significance
- Classification based on generated/pruned rules
  - If only *one rule* satisfies tuple X, assign the class label of the rule
  - If a *rule set* S satisfies X, CMAR
    - divides S into groups according to class labels
    - uses a weighted χ^2 measure to find the strongest group of rules, based on the statistical correlation of rules within a group
    - assigns X the class label of the strongest group

Perceptron & Winnow



- Vector: **x**, **w**
- Scalar: x, y, w

Input: $\{(\mathbf{x}_1, y_1), \ldots\}$

Output: classification function $f(\mathbf{x})$

$f(\mathbf{x}_i) > 0$ for $y_i = +1$

$f(\mathbf{x}_i) < 0$ for $y_i = -1$

$f(\mathbf{x}) \Rightarrow \mathbf{w}\mathbf{x} + b = 0$

or $w_1x_1+w_2x_2+b = 0$

- Perceptron: update W additively
- Winnow: update W multiplicatively