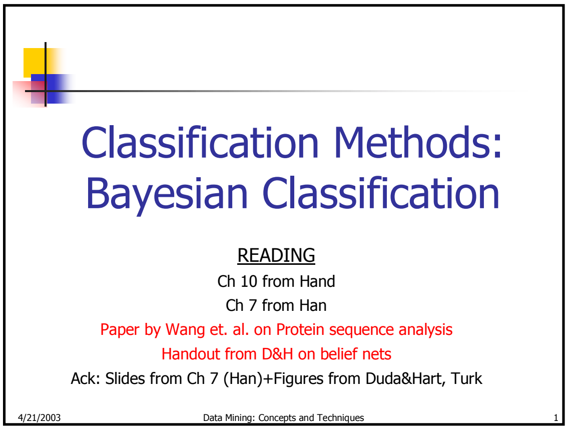


Classification Methods: Bayesian Classification

READING

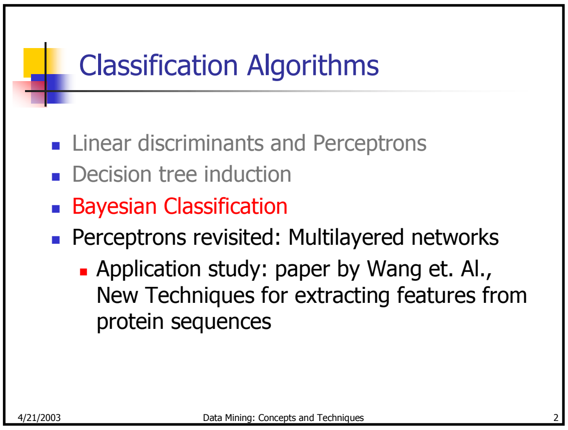
Ch 10 from Hand

Ch 7 from Han

Paper by Wang et. al. on Protein sequence analysis

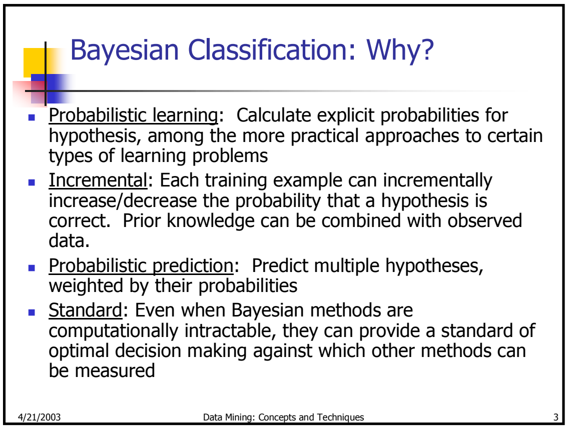
Handout from D&H on belief nets

Ack: Slides from Ch 7 (Han)+Figures from Duda&Hart, Turk



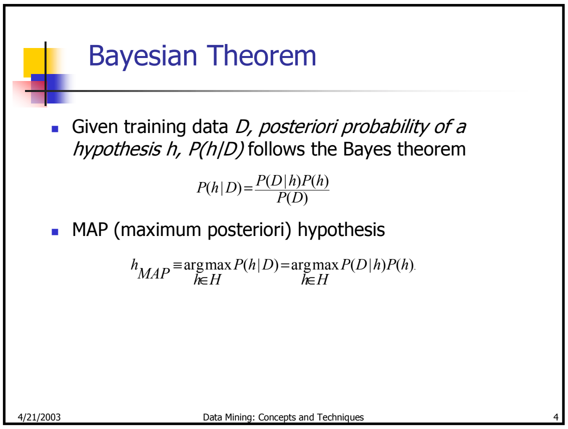
Classification Algorithms

- Linear discriminants and Perceptrons
- Decision tree induction
- Bayesian Classification
- Perceptrons revisited: Multilayered networks
  - Application study: paper by Wang et. Al., New Techniques for extracting features from protein sequences



Bayesian Classification: Why?

- Probabilistic learning: Calculate explicit probabilities for hypothesis, among the more practical approaches to certain types of learning problems
- Incremental: Each training example can incrementally increase/decrease the probability that a hypothesis is correct. Prior knowledge can be combined with observed data.
- Probabilistic prediction: Predict multiple hypotheses, weighted by their probabilities
- Standard: Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured



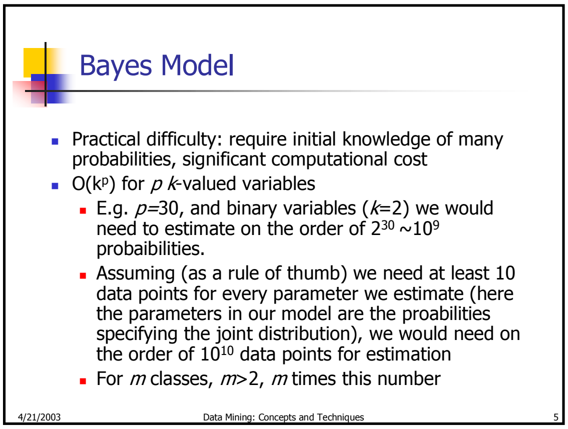
Bayesian Theorem

- Given training data *D, posteriori probability of a hypothesis h, P(h|D)* follows the Bayes theorem

$$P(h|D)=\frac{P(D|h)P(h)}{P(D)}$$

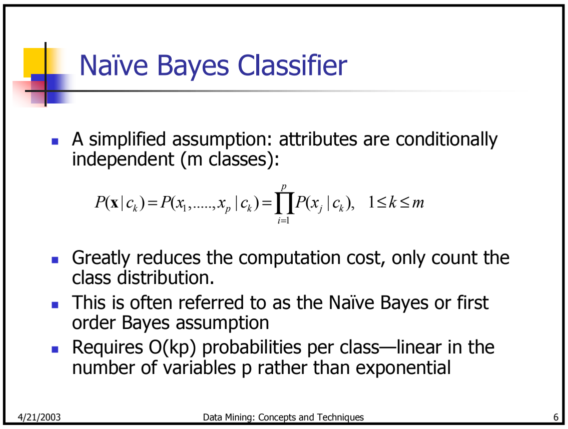
- MAP (maximum posteriori) hypothesis

$$h_{MAP}\equiv\underset{h\in H}{\arg\max}\,P(h|D)=\underset{h\in H}{\arg\max}\,P(D|h)P(h).$$



Bayes Model

- Practical difficulty: require initial knowledge of many probabilities, significant computational cost
- O(k^p) for *p k*-valued variables
  - E.g. *p*=30, and binary variables (*k*=2) we would need to estimate on the order of 2^{30} ~10^9 probabilities.
  - Assuming (as a rule of thumb) we need at least 10 data points for every parameter we estimate (here the parameters in our model are the proabilities specifying the joint distribution), we would need on the order of 10^{10} data points for estimation
  - For *m* classes, *m*>2, *m* times this number



Naïve Bayes Classifier

- A simplified assumption: attributes are conditionally independent (m classes):

$$P(\mathbf{x}|c_k)=P(x_1,.....,x_p|c_k)=\prod_{i=1}^{p}P(x_j|c_k),\quad 1\le k\le m$$

- Greatly reduces the computation cost, only count the class distribution.
- This is often referred to as the Naïve Bayes or first order Bayes assumption
- Requires O(kp) probabilities per class—linear in the number of variables p rather than exponential

Naïve Bayes Assumption (2)

- Note the strong independence assumption
  - May not be realistic
  - However, still permits acurate classification in many practical applications

$$P(c_k \mid \mathbf{x}) \propto p(\mathbf{x} \mid c_k) p(c_k) = p(c_k) \prod_{j=1}^{p} P(x_j | c_k) \quad 1 \le k \le m$$

Example: Playing tennis

- Given a training set, we can compute the probabilities (P=Play, N=no play)

Outlook	P	N		Humidity	P	N
sunny	2/9	3/5		high	3/9	4/5
overcast	4/9	0		normal	6/9	1/5
rain	3/9	2/5				
Tempreature				Windy		
hot	2/9	2/5		true	3/9	3/5
mild	4/9	2/5		false	6/9	2/5
cool	3/9	1/5				

Bayesian classification

- The classification problem may be formalized using a-posteriori probabilities:
- P(C|X) = prob. that the sample tuple $X=<x_1,\ldots,x_p>$ is of class C.
- E.g. P(class=N | outlook=sunny,windy=true,…)
- Idea: assign to sample X the class label C such that P(C|X) is maximal

Estimating a-posteriori probabilities

- Bayes theorem:

  P(C|X) = P(X|C)·P(C) / P(X)
- P(X) is constant for all classes
- P(C) = relative freq of class C samples
- C such that P(C|X) is maximum = C such that P(X|C)·P(C) is maximum
- Problem: computing P(X|C) is not feasible!

Naïve Bayesian Classification

- Naïve assumption: attribute independence

  $P(x_1,\ldots,x_p|C) = P(x_1|C)\cdot\ldots\cdot P(x_p|C)$
- If i-th attribute is categorical: $P(x_i|C)$ is estimated as the relative freq of samples having value x_i as i-th attribute in class C
- If i-th attribute is continuous: $P(x_i|C)$ is estimated thru a Gaussian density function
- Computationally easy in both cases

Play-tennis example: estimating $P(x_i|C)$

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

P(p) = 9/14
P(n) = 5/14

outlook	
P(sunny|p) = 2/9	**P(sunny|n) = 3/5**
P(overcast|p) = 4/9	**P(overcast|n) = 0**
P(rain|p) = 3/9	**P(rain|n) = 2/5**
temperature	
P(hot|p) = 2/9	**P(hot|n) = 2/5**
P(mild|p) = 4/9	**P(mild|n) = 2/5**
P(cool|p) = 3/9	**P(cool|n) = 1/5**
humidity	
P(high|p) = 3/9	**P(high|n) = 4/5**
P(normal|p) = 6/9	**P(normal|n) = 2/5**
windy	
P(true|p) = 3/9	**P(true|n) = 3/5**
P(false|p) = 6/9	**P(false|n) = 2/5**

Play-tennis example: classifying X

- An unseen sample X = <rain, hot, high, false>

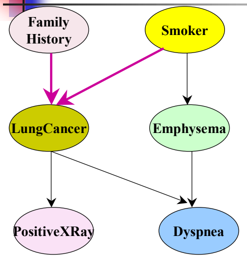
- P(X|p)·P(p) = P(rain|p)·P(hot|p)·P(high|p)·P(false|p)·P(p) = 3/9·2/9·3/9·6/9·9/14 = 0.010582
- P(X|n)·P(n) = P(rain|n)·P(hot|n)·P(high|n)·P(false|n)·P(n) = 2/5·2/5·4/5·2/5·5/14 = 0.018286

- Sample X is classified in class n (don't play)

The independence hypothesis…

- … makes computation possible
- … yields optimal classifiers when satisfied
- … but is seldom satisfied in practice, as attributes (variables) are often correlated.
- Attempts to overcome this limitation:
  - Bayesian networks, that combine Bayesian reasoning with causal relationships between attributes
  - Decision trees, that reason on one attribute at the time, considering most important attributes first

Bayesian Belief Networks



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

Bayesian Belief Networks

- A graphical model of causal relationships
  - Also called *causal networks, belief nets*
  - Toplogical form: directed acyclic graph (DAC)—each link is directional, and no loops (in general)
  - Each node represents one of the system variables, and the links joining the nodes represent conditional probabilities
- Several cases of learning Bayesian belief networks
  - Given both network structure and all the variables: easy
  - When the network structure is not known in advance

Turk

Example

I'm at work and my neighbor John calls to say my home alarm is ringing, but my neighbor Mary doesn't call. The alarm is sometimes triggered by minor earthquakes. Was there a burglar at my house?

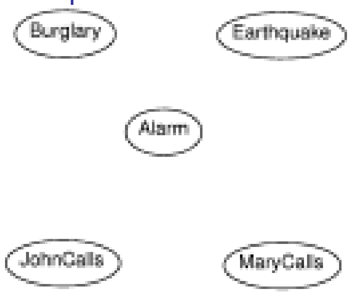
- Random variables:
  - JohnCalls, MaryCalls, Earthquake, Burglar, Alarm (all boolean)

- Draw the belief net, showing the causal links

- This defines the joint probability
  - P(JohnCalls, MaryCalls, Earthquake, Burglar, Alarm)
- For a belief net:
  - $P(var_1, \ldots, var_N) = P(var_1|Parents(var_1)) \ldots P(var_N|Parents(var_N))$

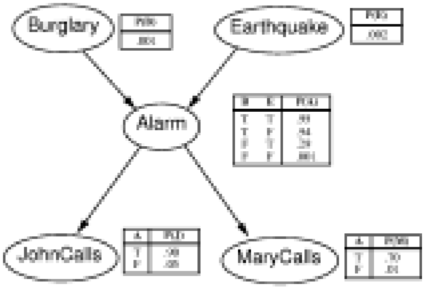
Turk

Example



Links and probabilities?

Example

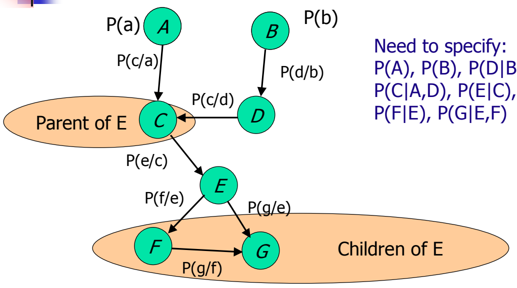


Joint probability? P(J, ¬M, A, B, ¬E)?

Example

- Conditional independence is seen here
  - P(JohnCalls|MaryCalls, Alarm, Earthquake, Burglary) = P(JohnCalls|Alarm)
  - So JohnCalls is independent of MaryCalls, Earthquake, and Burglary, given Alarm
- Does this mean that an earthquake or a burglary do not influence whether or not John calls?
  - No, but the influence is already accounted for in the Alarm variable
  - JohnCalls is conditionally independent of Earthquake, but not absolutely independent of it

Bayesian Belief Networks



Need to specify:
P(A), P(B), P(D|B), P(C|A,D), P(E|C), P(F|E), P(G|E,F)

Bayes' rule

Likelihood of b given a

$$P(B\mid A)=\frac{P(A\mid B)P(B)}{P(A)}$$

Probabilities conditioned on a context c:

$$P(A\mid B,C)P(B\mid C)=P(A,B\mid C)$$

$$P(B\mid A,C)P(A\mid C)=P(A,B\mid C)$$

$$P(B\mid A,C)=\frac{P(A\mid B,C)P(B\mid C)}{P(A\mid C)}$$

Conditional independence

$$P(A\mid B)=P(A\mid B,C)$$

A is independent of C given B

P(**x**|e): belief probability

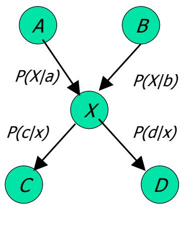
The *belief* of a set of propositions **x** =(x1, x2,..) on node X describes the relative probabilities of the variables given all the evidence **e** in the entire network, i.e., P(**x**|**e**)

$$P(\mathbf{x}\mid \mathbf{e}^{C},\mathbf{e}^{\mathrm{P}})\propto P(\mathbf{e}^{C}\mid \mathbf{x})P(\mathbf{x}\mid \mathbf{e}^{\mathrm{P}})$$

$\mathbf{e}^{\mathrm{P}}$: evidence on the parent nodes

$\mathbf{e}^{C}$: evidence on the children nodes

Evidence from Child nodes



Child nodes are conditionally independent given **x.**

$$P(\mathbf{e}_C,\mathbf{e}_D\mid \mathbf{x})=P(\mathbf{e}_C\mid \mathbf{x})P(\mathbf{e}_D\mid \mathbf{x})$$

$$P(\mathbf{e}^{C}\mid \mathbf{x})=P(\mathbf{e}_{C_1},\mathbf{e}_{C_2},.....,\mathbf{e}_{C_{|C|}}\mid \mathbf{x})$$

$$=P(\mathbf{e}_{C_1}\mid \mathbf{x})P(\mathbf{e}_{C_2}\mid \mathbf{x})....P(\mathbf{e}_{C_{|C|}}\mid \mathbf{x})$$

$$P(\mathbf{e}^{C}\mid \mathbf{x})=\prod_{j=1}^{|C|}P(\mathbf{e}_{C_j}\mid \mathbf{x})$$

Parent nodes

Evidence from the parent nodes:

$$P(\mathbf{x} \mid \mathbf{e}^P) = P(\mathbf{x} \mid \mathbf{e}_{P_1}, \mathbf{e}_{P_2}, \ldots, \mathbf{e}_{P_{|P|}})$$

$$= \sum_{i,j,..,k} P(\mathbf{x} \mid P_{1i}, P_{2j}, \ldots.., P_{|P|k}) P(P_{1i}, P_{2j}, \ldots.., P_{|P|k} \mid \mathbf{e}_{P_1}, \mathbf{e}_{P_2}, \ldots, \mathbf{e}_{P_{|P|}})$$

$$= \sum_{i,j,..,k} P(\mathbf{x} \mid P_{1i}, P_{2j}, \ldots.., P_{|P|k}) P(P_{1i} \mid \mathbf{e}_{P_1}) P(P_{2j} \mid \mathbf{e}_{P_2}) \ldots. P(P_{|P|k} \mid \mathbf{e}_{P_{|P|}})$$

Summation is over all possible configurations of values on different parent nodes; P_{mn} denotes a parameter value for state *n* on the parent node P_m. Again the assumption is that (unconnected) parent nodes are statistically independent

Belief Probability

$$P(\mathbf{e}^C \mid \mathbf{x}) = \prod_{j=1}^{|C|} P(\mathbf{e}_{C_j} \mid \mathbf{x})$$

$$P(\mathbf{x} \mid \mathbf{e}^P) = \sum_{all\ P_{mn}} P(\mathbf{x} \mid P_{mn}) \prod_{i=1}^{|P|} P(P_i \mid \mathbf{e}_{P_i})$$

$$\boxed{P(\mathbf{x} \mid \mathbf{e}) \propto \prod_{j=1}^{|C|} P(\mathbf{e}_{C_j} \mid \mathbf{x}) \left[\sum_{all\ P_{mn}} P(\mathbf{x} \mid P_{mn}) \prod_{i=1}^{|P|} P(P_i \mid \mathbf{e}) \right]}$$

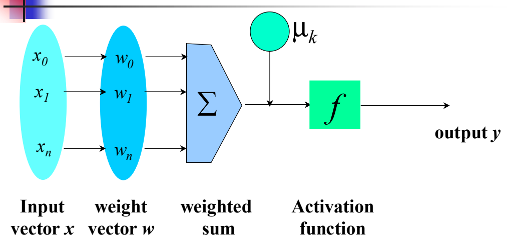
- first term is due to the children (product of their independent likelihoods).
- second is the sum over all possible configurations of states on the prior probabilities of their values and the conditional probabilities of the x variables given those parent values

Example: fish classification

Classification Algorithms

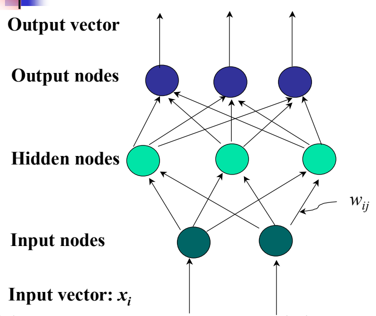
- Linear discriminants and Perceptrons
- Decision tree induction
- Bayesian Classification
- Perceptron revisited: Multilayered networks

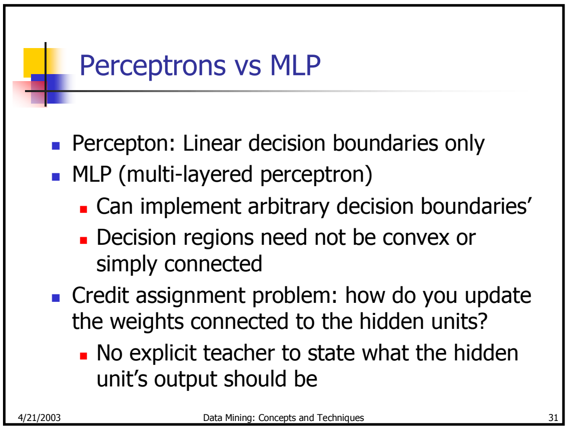
A "Neuron"



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

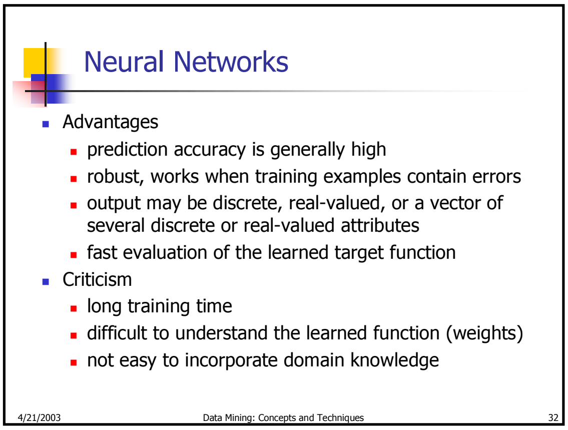
Multi-Layer Perceptron





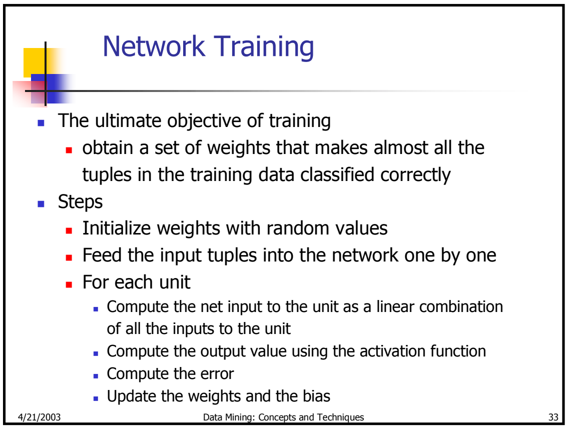
Perceptrons vs MLP

- Percepton: Linear decision boundaries only
- MLP (multi-layered perceptron)
  - Can implement arbitrary decision boundaries'
  - Decision regions need not be convex or simply connected
- Credit assignment problem: how do you update the weights connected to the hidden units?
  - No explicit teacher to state what the hidden unit's output should be



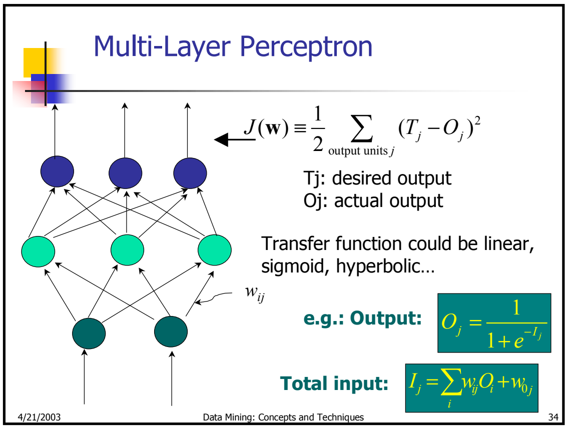
Neural Networks

- Advantages
  - prediction accuracy is generally high
  - robust, works when training examples contain errors
  - output may be discrete, real-valued, or a vector of several discrete or real-valued attributes
  - fast evaluation of the learned target function
- Criticism
  - long training time
  - difficult to understand the learned function (weights)
  - not easy to incorporate domain knowledge



Network 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



Multi-Layer Perceptron

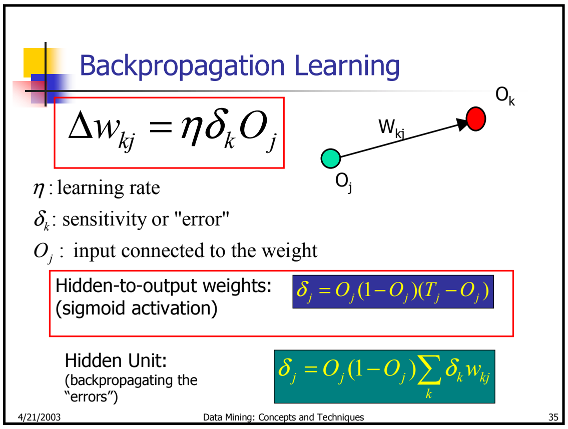
$$J(\mathbf{w}) \equiv \frac{1}{2} \sum_{\text{output units } j} (T_j - O_j)^2$$

Tj: desired output
Oj: actual output

Transfer function could be linear, sigmoid, hyperbolic…

e.g.: Output: $O_j = \dfrac{1}{1+e^{-I_j}}$

Total input: $I_j = \sum_i w_{ij} O_i + w_{0j}$



Backpropagation Learning

$$\boxed{\Delta w_{kj} = \eta \delta_k O_j}$$

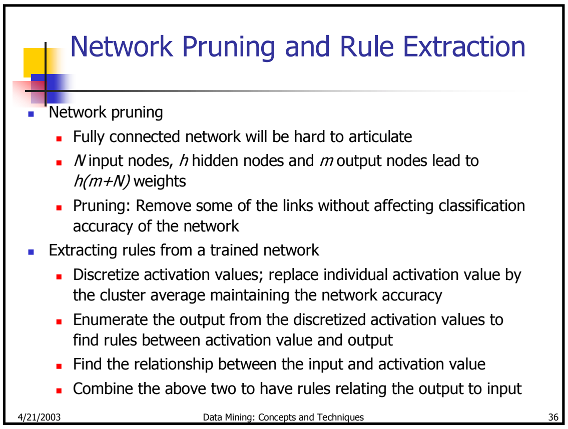
η : learning rate

δ_k: sensitivity or "error"

O_j : input connected to the weight

Hidden-to-output weights: (sigmoid activation) $\delta_j = O_j(1-O_j)(T_j - O_j)$

Hidden Unit: (backpropagating the "errors") $\delta_j = O_j(1-O_j)\sum_k \delta_k w_{kj}$



Network Pruning and Rule Extraction

- Network pruning
  - Fully connected network will be hard to articulate
  - *N* input nodes, *h* hidden nodes and *m* output nodes lead to *h(m+N)* weights
  - Pruning: Remove some of the links without affecting classification accuracy of the network
- Extracting rules from a trained network
  - Discretize activation values; replace individual activation value by the cluster average maintaining the network accuracy
  - Enumerate the output from the discretized activation values to find rules between activation value and output
  - Find the relationship between the input and activation value
  - Combine the above two to have rules relating the output to input

Summary

- Classification is an extensively studied problem (mainly in statistics, machine learning & neural networks)
- Classification is probably one of the most widely used data mining techniques with a lot of extensions
- Scalability is still an important issue for database applications: thus combining classification with database techniques should be a promising topic
- Research directions: classification of non-relational data, e.g., text, spatial, multimedia, etc..

References (I)

- C. Apte and S. Weiss. Data mining with decision trees and decision rules. Future Generation Computer Systems, 13, 1997.
- L. Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and Regression Trees. Wadsworth International Group, 1984.
- P. K. Chan and S. J. Stolfo. Learning arbiter and combiner trees from partitioned data for scaling machine learning. In Proc. 1st Int. Conf. Knowledge Discovery and Data Mining (KDD'95), pages 39-44, Montreal, Canada, August 1995.
- U. M. Fayyad. Branching on attribute values in decision tree generation. In Proc. 1994 AAAI Conf., pages 601-606, AAAI Press, 1994.
- J. Gehrke, R. Ramakrishnan, and V. Ganti. Rainforest: A framework for fast decision tree construction of large datasets. In Proc. 1998 Int. Conf. Very Large Data Bases, pages 416-427, New York, NY, August 1998.
- M. Kamber, L. Winstone, W. Gong, S. Cheng, and J. Han. Generalization and decision tree induction: Efficient classification in data mining. In Proc. 1997 Int. Workshop Research Issues on Data Engineering (RIDE'97), pages 111-120, Birmingham, England, April 1997.

References (II)

- J. Magidson. The Chaid approach to segmentation modeling: Chi-squared automatic interaction detection. In R. P. Bagozzi, editor, Advanced Methods of Marketing Research, pages 118-159. Blackwell Business, Cambridge Massechusetts, 1994.
- M. Mehta, R. Agrawal, and J. Rissanen. SLIQ : A fast scalable classifier for data mining. In Proc. 1996 Int. Conf. Extending Database Technology (EDBT'96), Avignon, France, March 1996.
- S. K. Murthy, Automatic Construction of Decision Trees from Data: A Multi-Diciplinary Survey, Data Mining and Knowledge Discovery 2(4): 345-389, 1998
- J. R. Quinlan. Bagging, boosting, and c4.5. In Proc. 13th Natl. Conf. on Artificial Intelligence (AAAI'96), 725-730, Portland, OR, Aug. 1996.
- R. Rastogi and K. Shim. Public: A decision tree classifer that integrates building and pruning. In Proc. 1998 Int. Conf. Very Large Data Bases, 404-415, New York, NY, August 1998.
- J. Shafer, R. Agrawal, and M. Mehta. SPRINT : A scalable parallel classifier for data mining. In Proc. 1996 Int. Conf. Very Large Data Bases, 544-555, Bombay, India, Sept. 1996.
- S. M. Weiss and C. A. Kulikowski. Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems. Morgan Kaufman, 1991.

Course outline

- Introduction* (1)
- Data Warehouse (1)
- Data Preprocessing (2)
- Classification Methods (4)
- Clustering Methods
- Pattern finding (2)

- Applications (1-2)
- Multimedia Mining (2)
- Survey of recent research (2)
  - Presentations
- Course project (2)
  - Presentations

Clustering methods

- Chapter 8 (Han)
- Chapter 9 (Hand)
- Topics
  - Distance based methods
  - Hierarchical methods
  - Probabilistic model based clustering, mixture models, EM algorithm
  - Application examples