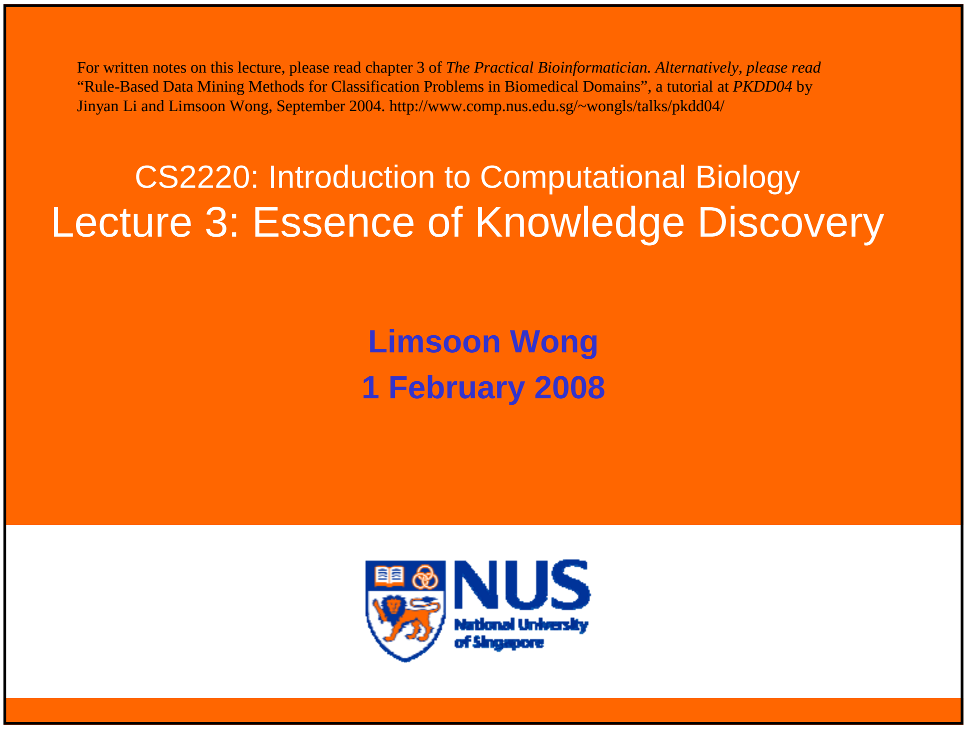
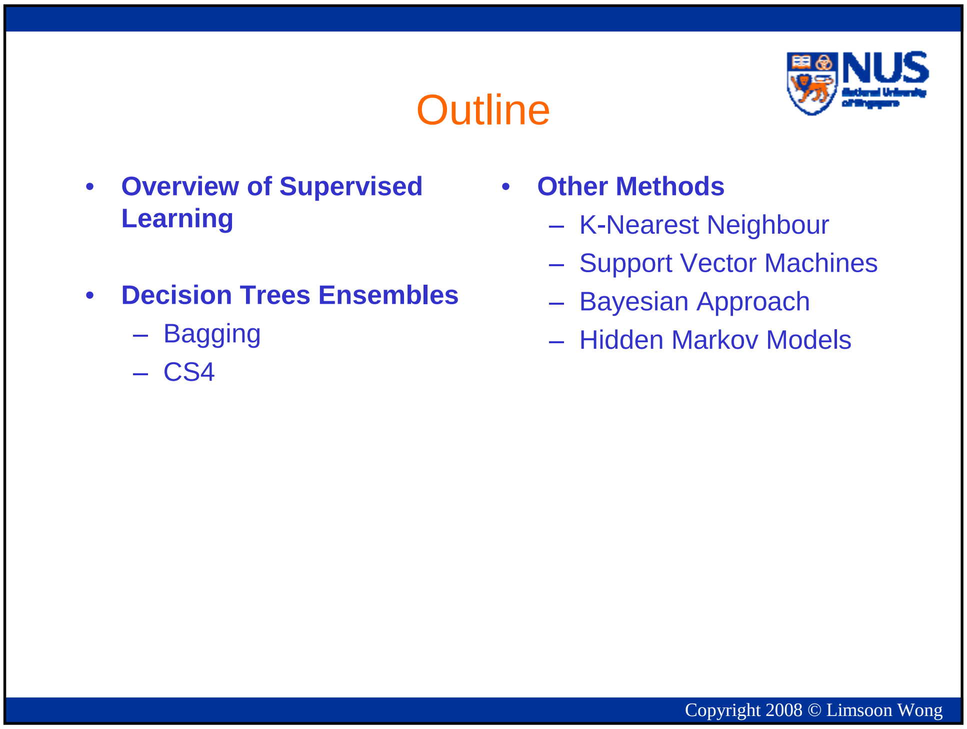


For written notes on this lecture, please read chapter 3 of *The Practical Bioinformatician. Alternatively, please read* "Rule-Based Data Mining Methods for Classification Problems in Biomedical Domains", a tutorial at *PKDD04* by Jinyan Li and Limsoon Wong, September 2004. http://www.comp.nus.edu.sg/~wongls/talks/pkdd04/

CS2220: Introduction to Computational Biology
Lecture 3: Essence of Knowledge Discovery

Limsoon Wong

1 February 2008



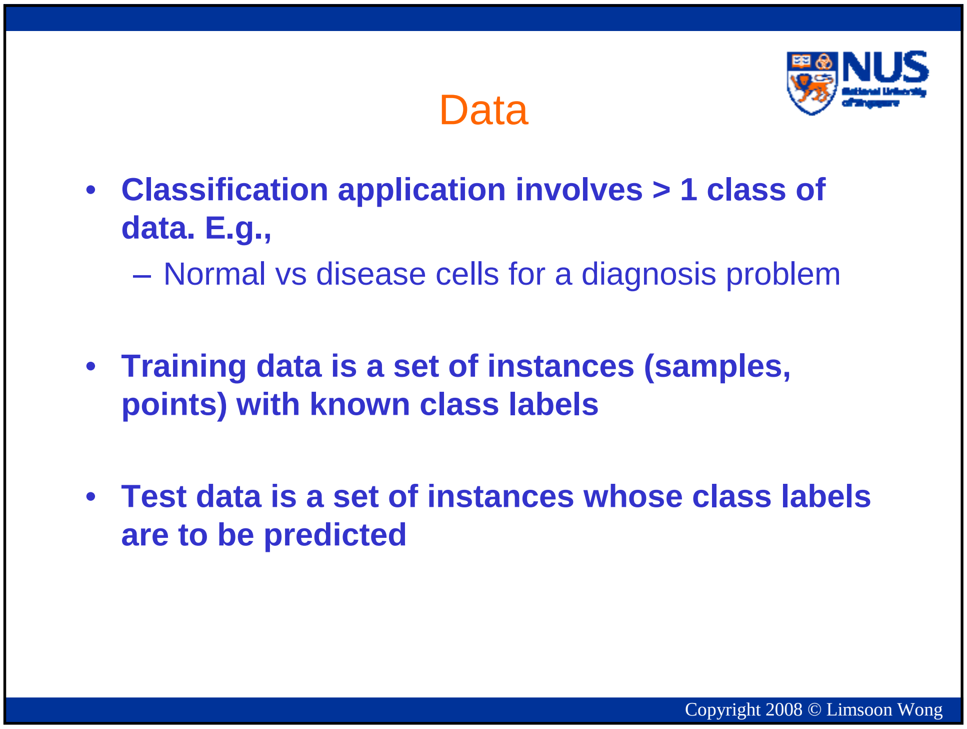
Outline

- **Overview of Supervised Learning**
- **Decision Trees Ensembles**
  - Bagging
  - CS4
- **Other Methods**
  - K-Nearest Neighbour
  - Support Vector Machines
  - Bayesian Approach
  - Hidden Markov Models

Overview of Supervised Learning

Computational Supervised Learning

- **Also called classification**
- **Learn from past experience, and use the learned knowledge to classify new data**
- **Knowledge learned by intelligent algorithms**
- **Examples:**
  - Clinical diagnosis for patients
  - Cell type classification

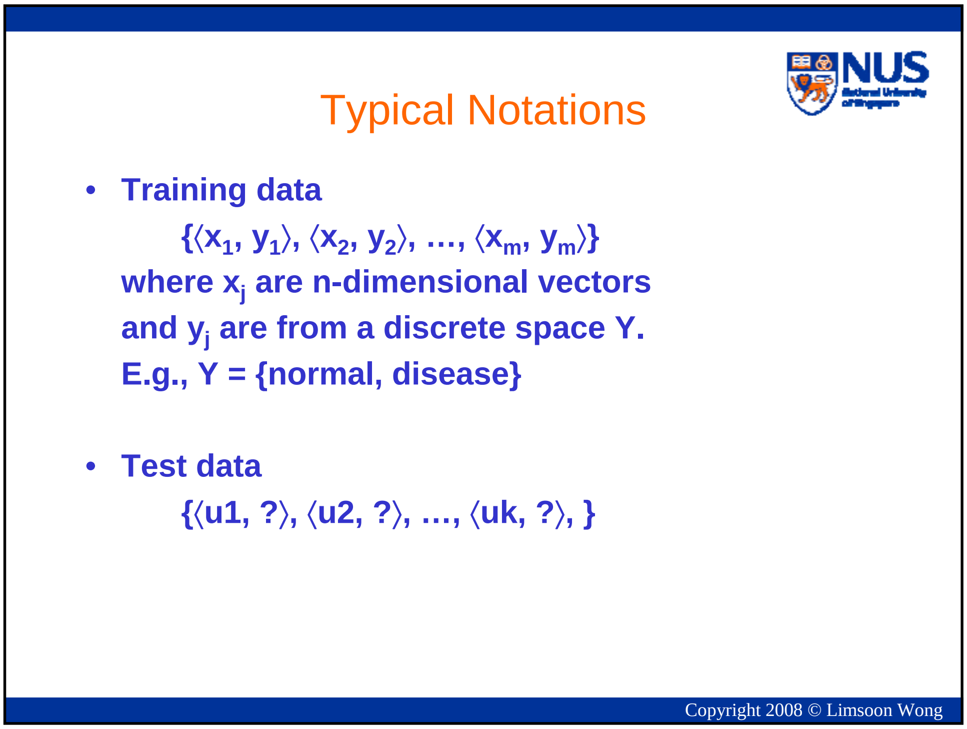


Data

- **Classification application involves > 1 class of data. E.g.,**
  - Normal vs disease cells for a diagnosis problem

- **Training data is a set of instances (samples, points) with known class labels**

- **Test data is a set of instances whose class labels are to be predicted**



Typical Notations

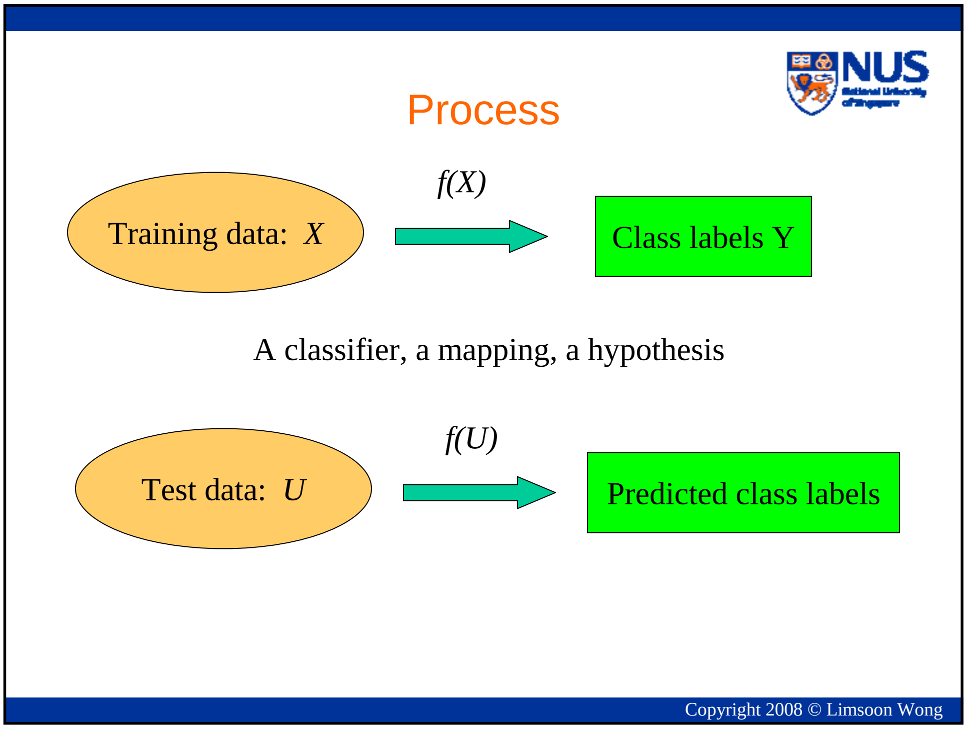
- **Training data**

  $\{\langle x_1, y_1\rangle, \langle x_2, y_2\rangle, \ldots, \langle x_m, y_m\rangle\}$

  where x_j are n-dimensional vectors and y_j are from a discrete space Y. E.g., Y = {normal, disease}

- **Test data**

  $\{\langle u1, ?\rangle, \langle u2, ?\rangle, \ldots, \langle uk, ?\rangle, \}$

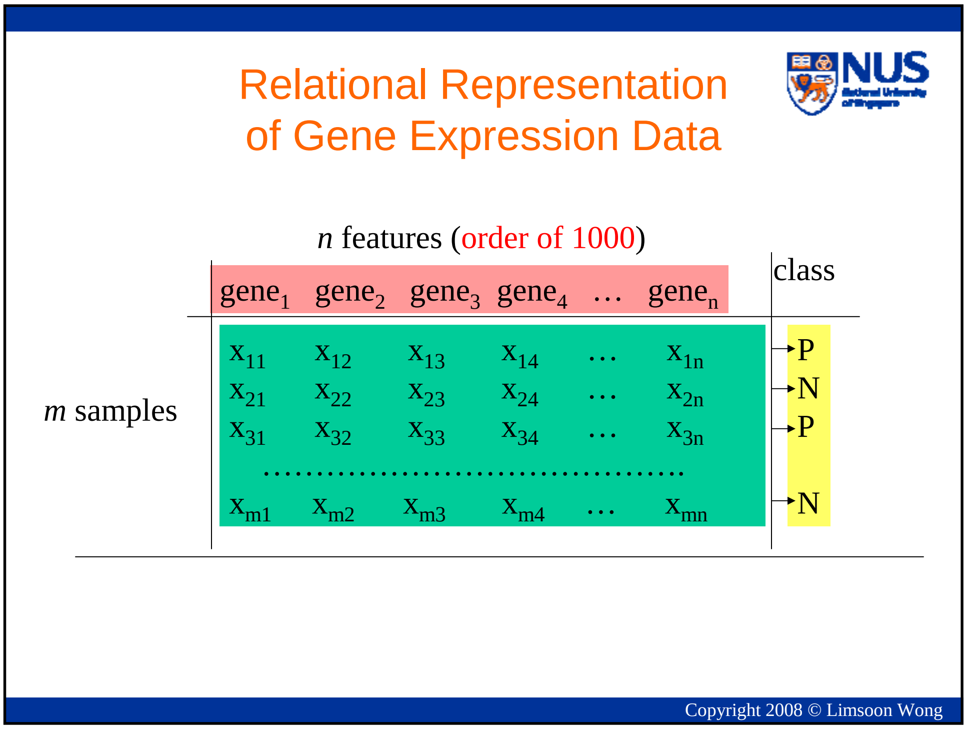


Process

Training data: X $\xrightarrow{f(X)}$ Class labels Y

A classifier, a mapping, a hypothesis

Test data: U $\xrightarrow{f(U)}$ Predicted class labels



Relational Representation of Gene Expression Data

n features (order of 1000)

$$\begin{array}{c|cccccc|c}
 & \text{gene}_1 & \text{gene}_2 & \text{gene}_3 & \text{gene}_4 & \dots & \text{gene}_n & \text{class} \\ \hline
 & x_{11} & x_{12} & x_{13} & x_{14} & \dots & x_{1n} & \to P \\
 & x_{21} & x_{22} & x_{23} & x_{24} & \dots & x_{2n} & \to N \\
m \text{ samples} & x_{31} & x_{32} & x_{33} & x_{34} & \dots & x_{3n} & \to P \\
 & \dots & \dots & \dots & \dots & \dots & \dots & \\
 & x_{m1} & x_{m2} & x_{m3} & x_{m4} & \dots & x_{mn} & \to N \\ \hline
\end{array}$$

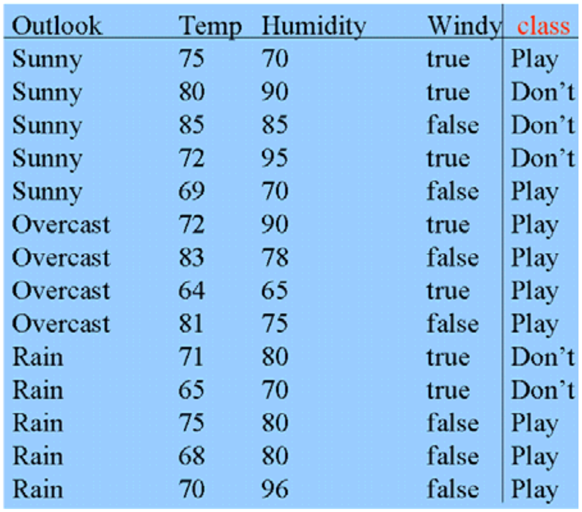
Features (aka Attributes)

- **Categorical features**
  - color = {red, blue, green}

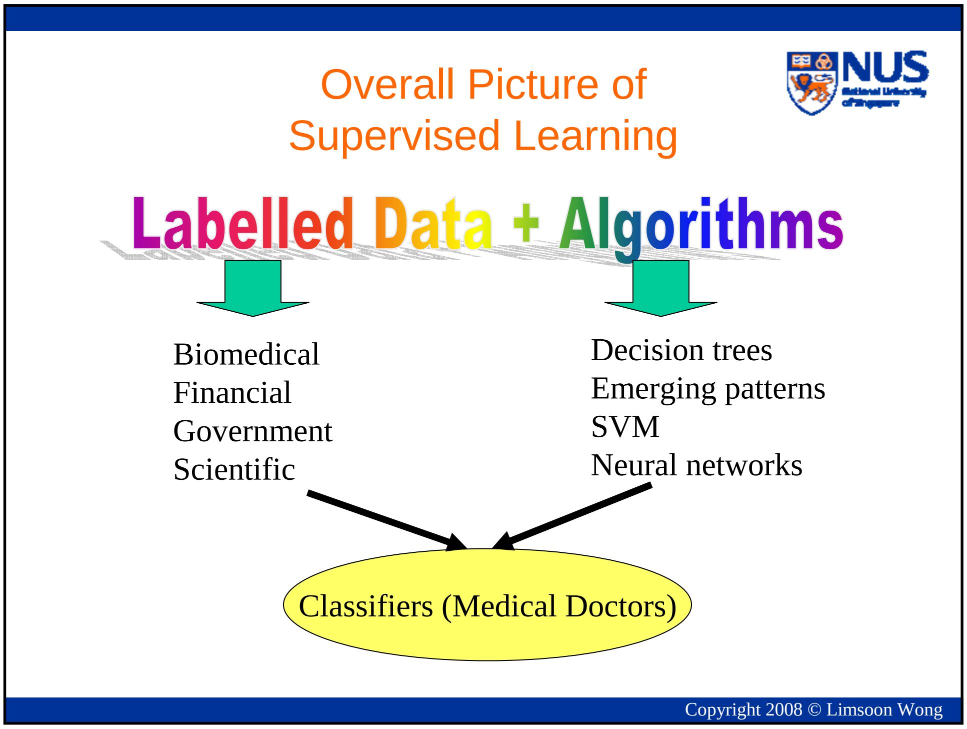
- **Continuous or numerical features**
  - gene expression
  - age
  - blood pressure

- **Discretization**

An Example



Outlook	Temp	Humidity	Windy	class
Sunny	75	70	true	Play
Sunny	80	90	true	Don’t
Sunny	85	85	false	Don’t
Sunny	72	95	true	Don’t
Sunny	69	70	false	Play
Overcast	72	90	true	Play
Overcast	83	78	false	Play
Overcast	64	65	true	Play
Overcast	81	75	false	Play
Rain	71	80	true	Don’t
Rain	65	70	true	Don’t
Rain	75	80	false	Play
Rain	68	80	false	Play
Rain	70	96	false	Play



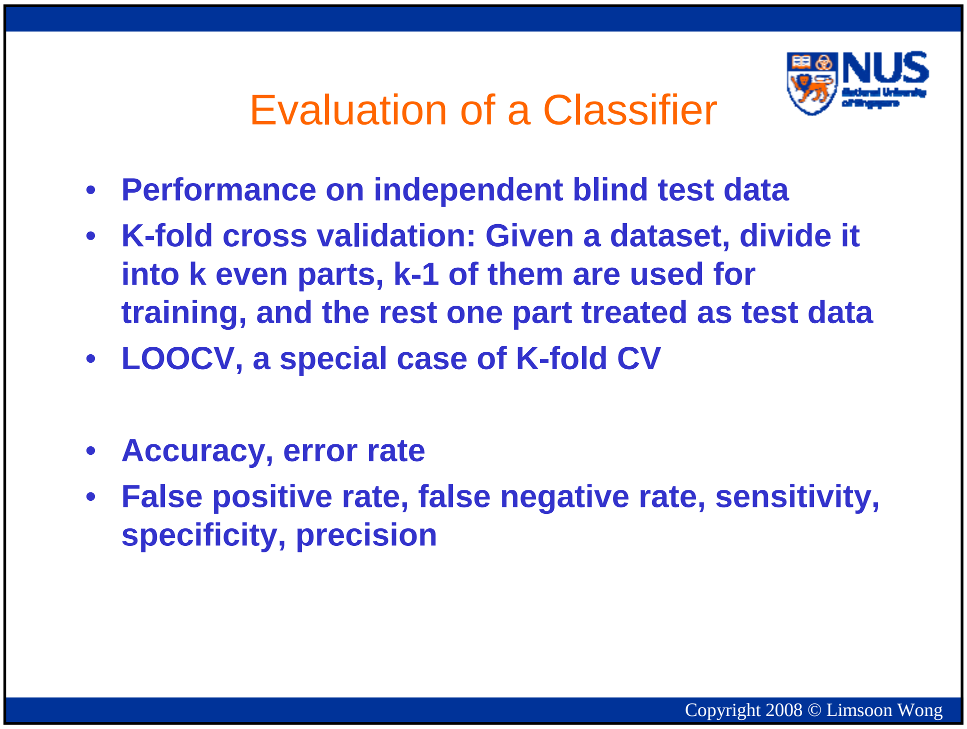
Overall Picture of Supervised Learning

Labelled Data + Algorithms

Biomedical
Financial
Government
Scientific

Decision trees
Emerging patterns
SVM
Neural networks

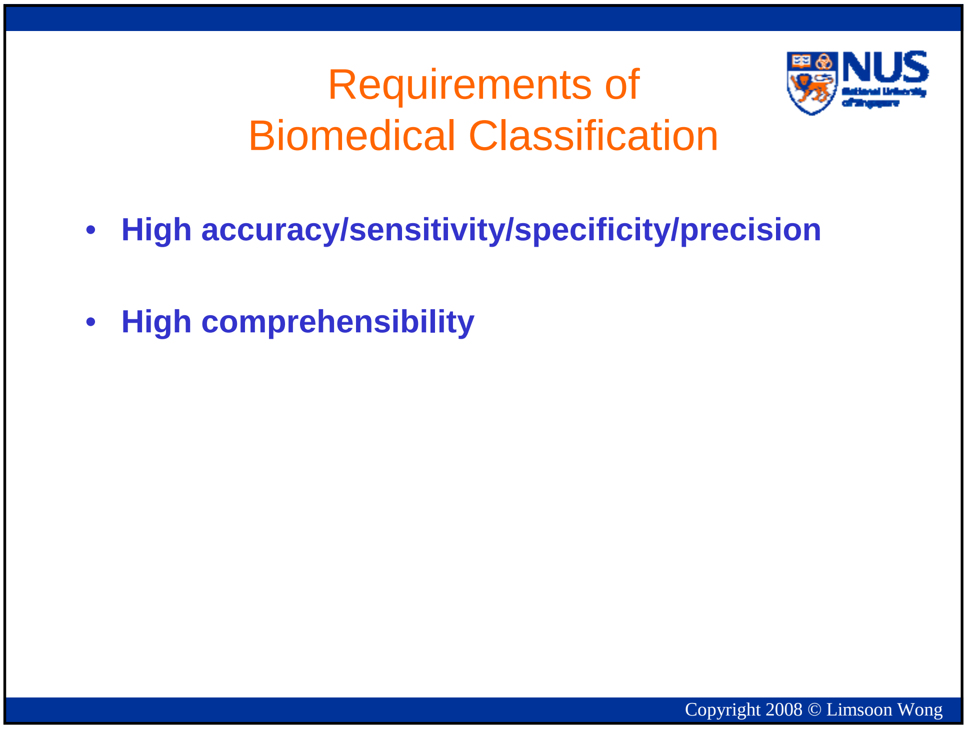
Classifiers (Medical Doctors)



Evaluation of a Classifier

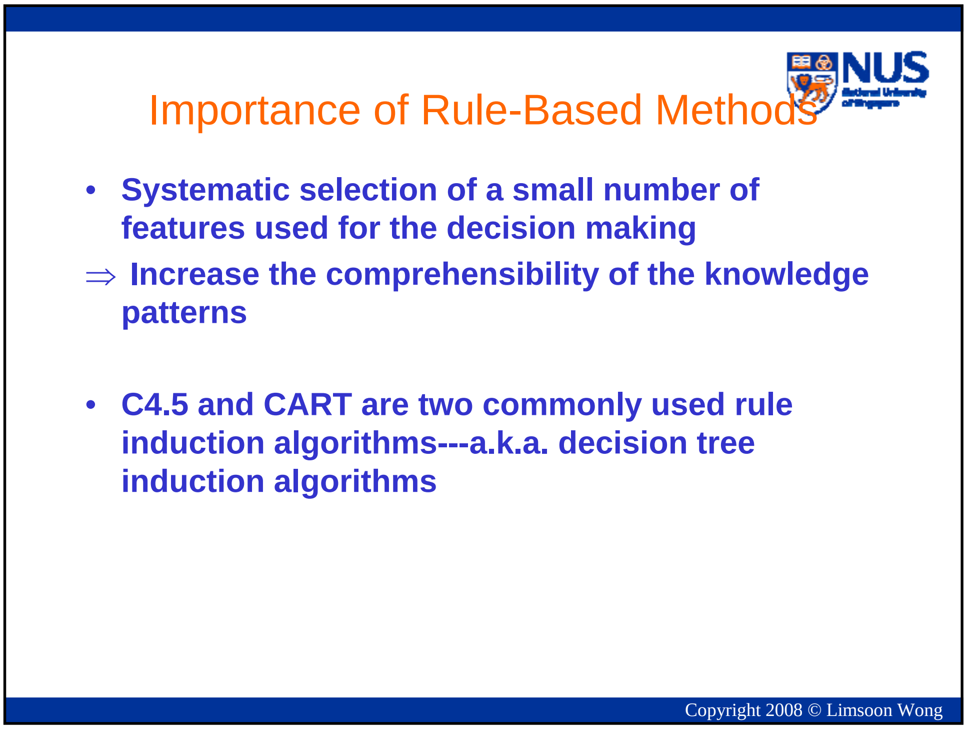
- Performance on independent blind test data
- K-fold cross validation: Given a dataset, divide it into k even parts, k-1 of them are used for training, and the rest one part treated as test data
- LOOCV, a special case of K-fold CV

- Accuracy, error rate
- False positive rate, false negative rate, sensitivity, specificity, precision



Requirements of Biomedical Classification

- High accuracy/sensitivity/specificity/precision
- High comprehensibility

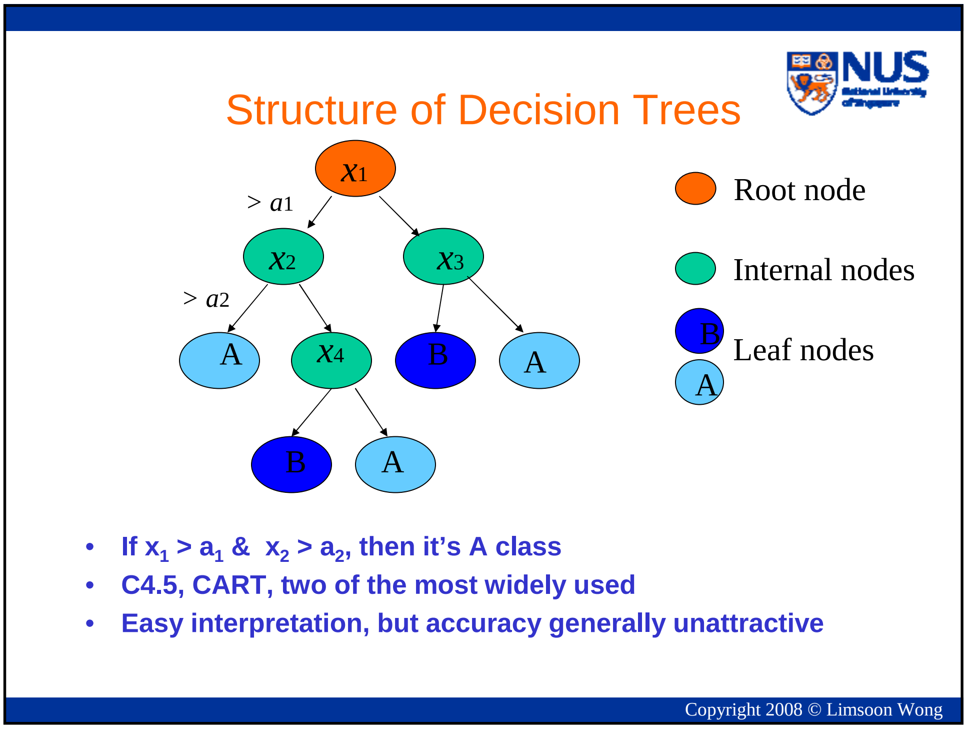


Importance of Rule-Based Methods

- Systematic selection of a small number of features used for the decision making

⇒ Increase the comprehensibility of the knowledge patterns

- C4.5 and CART are two commonly used rule induction algorithms---a.k.a. decision tree induction algorithms



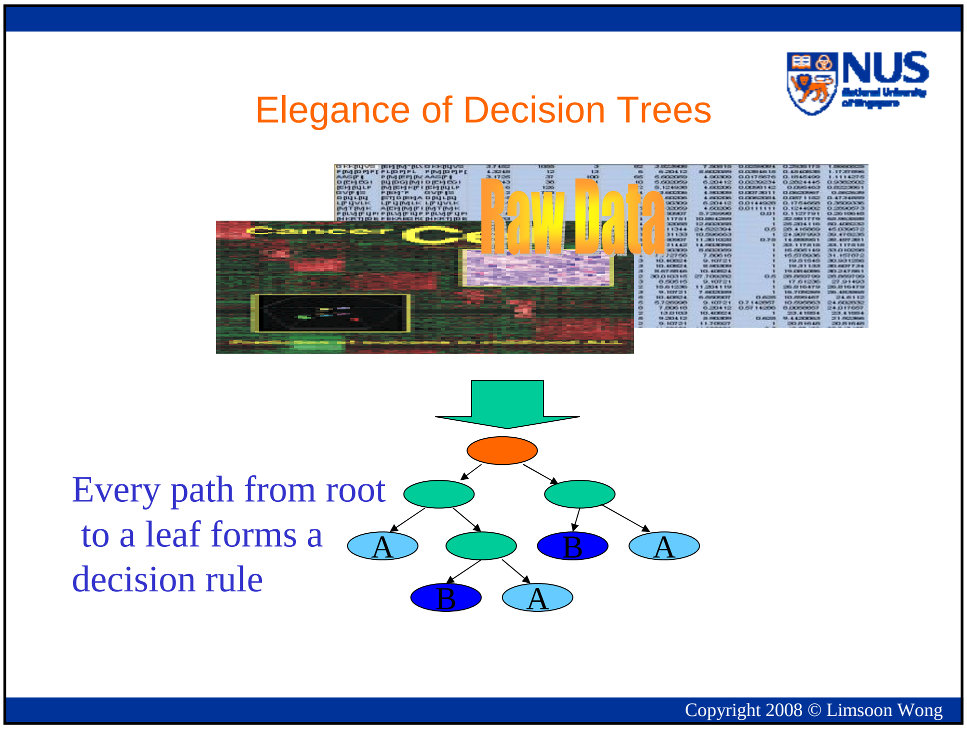
Structure of Decision Trees

● Root node

● Internal nodes

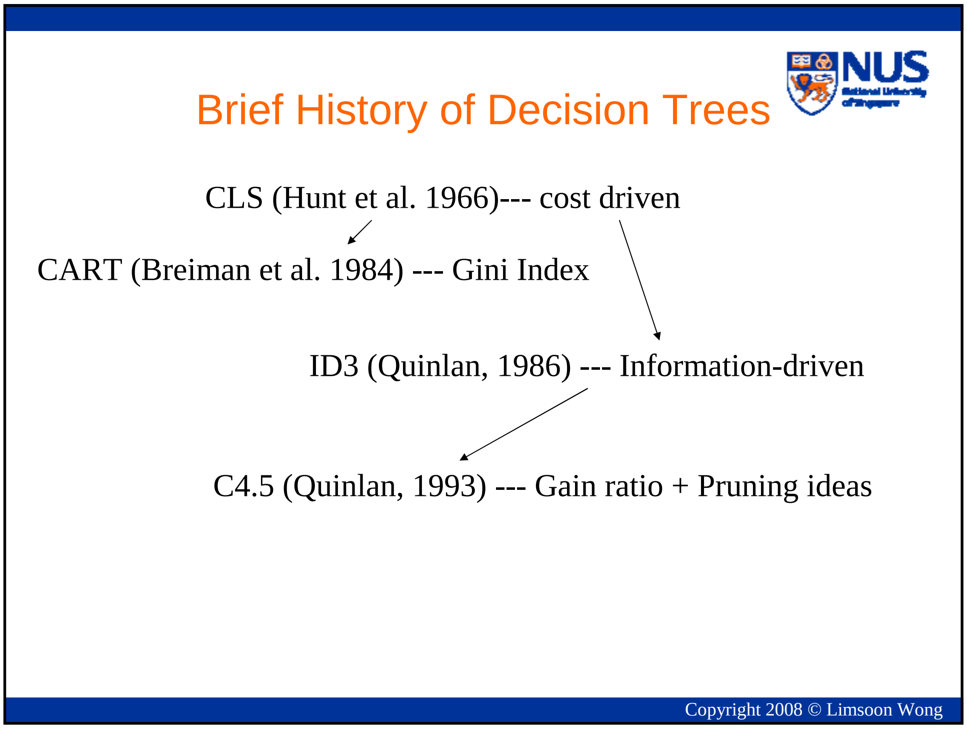
● B ● A Leaf nodes

- If $x_1 > a_1$ & $x_2 > a_2$, then it's A class
- C4.5, CART, two of the most widely used
- Easy interpretation, but accuracy generally unattractive



Elegance of Decision Trees

Every path from root to a leaf forms a decision rule



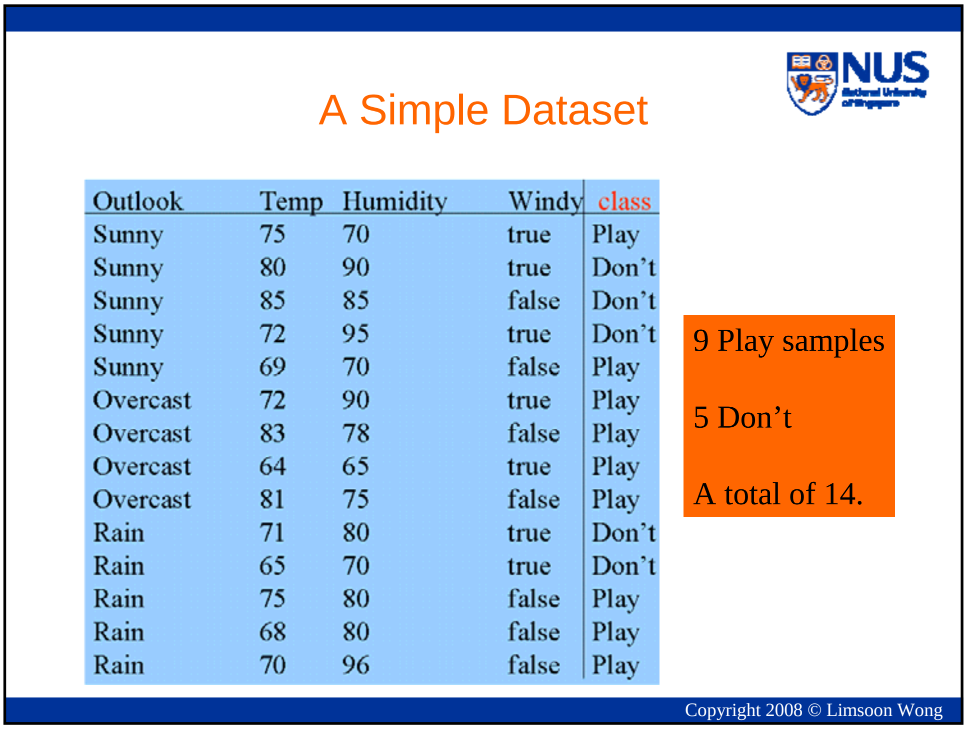
Brief History of Decision Trees

CLS (Hunt et al. 1966)--- cost driven

CART (Breiman et al. 1984) --- Gini Index

ID3 (Quinlan, 1986) --- Information-driven

C4.5 (Quinlan, 1993) --- Gain ratio + Pruning ideas



A Simple Dataset

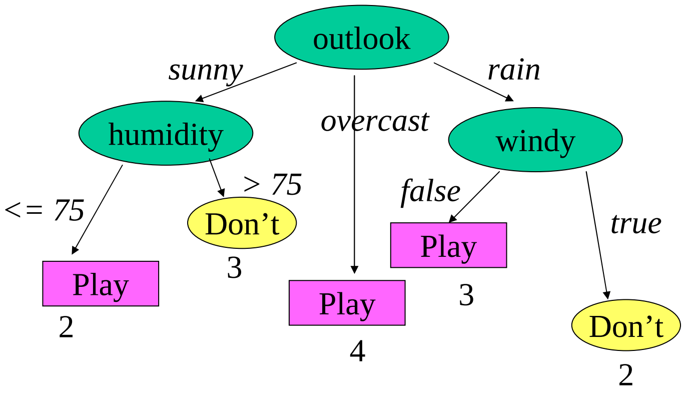
Outlook	Temp	Humidity	Windy	class
Sunny	75	70	true	Play
Sunny	80	90	true	Don't
Sunny	85	85	false	Don't
Sunny	72	95	true	Don't
Sunny	69	70	false	Play
Overcast	72	90	true	Play
Overcast	83	78	false	Play
Overcast	64	65	true	Play
Overcast	81	75	false	Play
Rain	71	80	true	Don't
Rain	65	70	true	Don't
Rain	75	80	false	Play
Rain	68	80	false	Play
Rain	70	96	false	Play

9 Play samples

5 Don't

A total of 14.

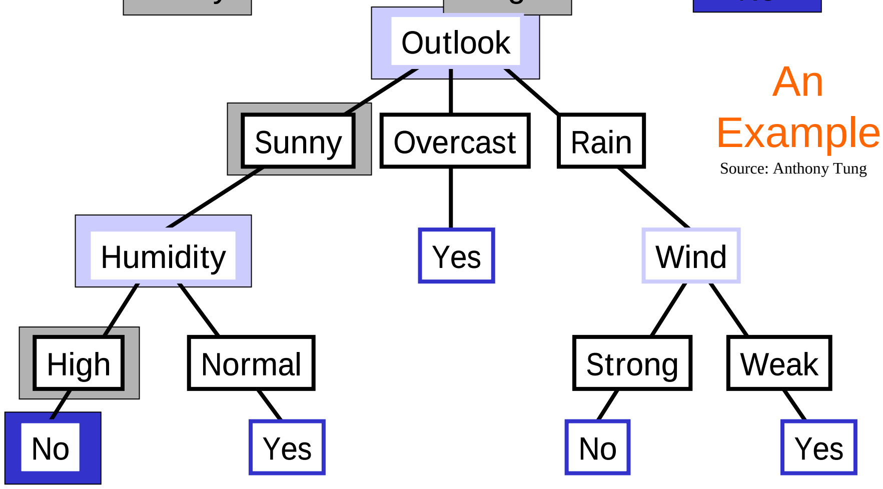
A Decision Tree



- **Construction of a tree is equivalent to determination of the root node of the tree and the root node of its sub-trees**

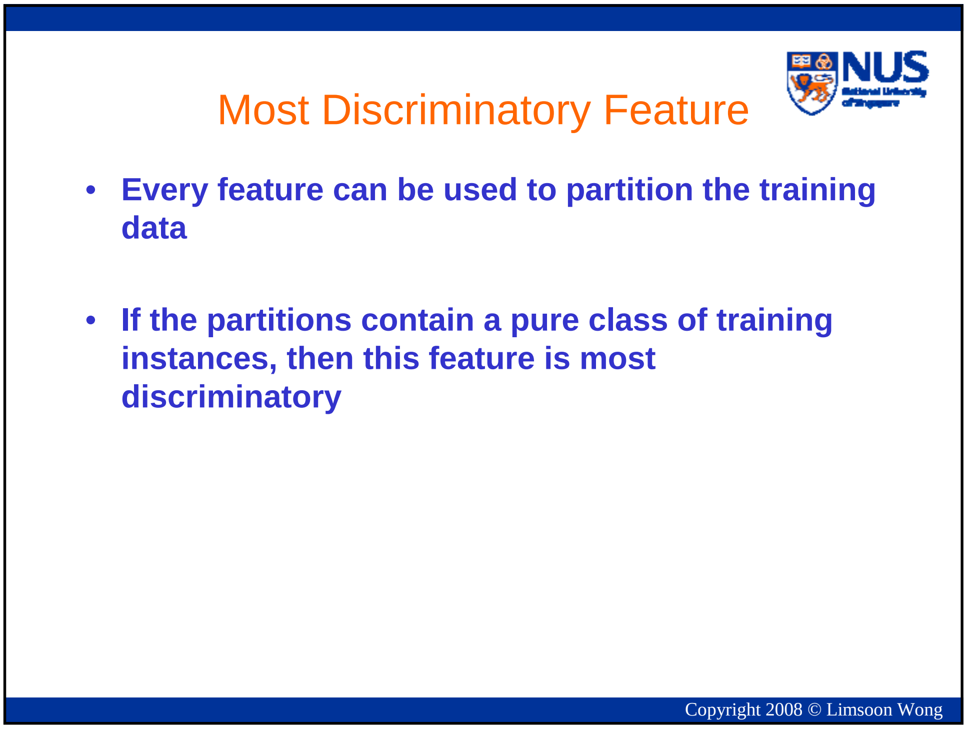
Exercise: What is the accuracy of this tree?

Outlook	Temperature	Humidity	Wind	PlayTennis
Sunny	Hot	High	Weak	?No



An Example

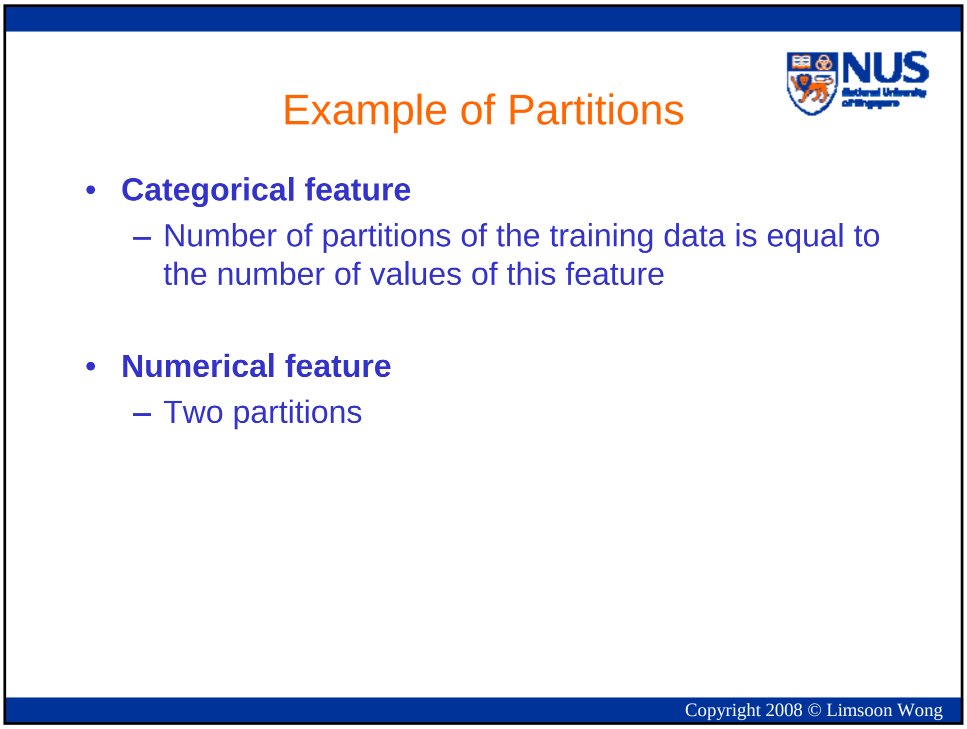
Source: Anthony Tung



Most Discriminatory Feature

- Every feature can be used to partition the training data

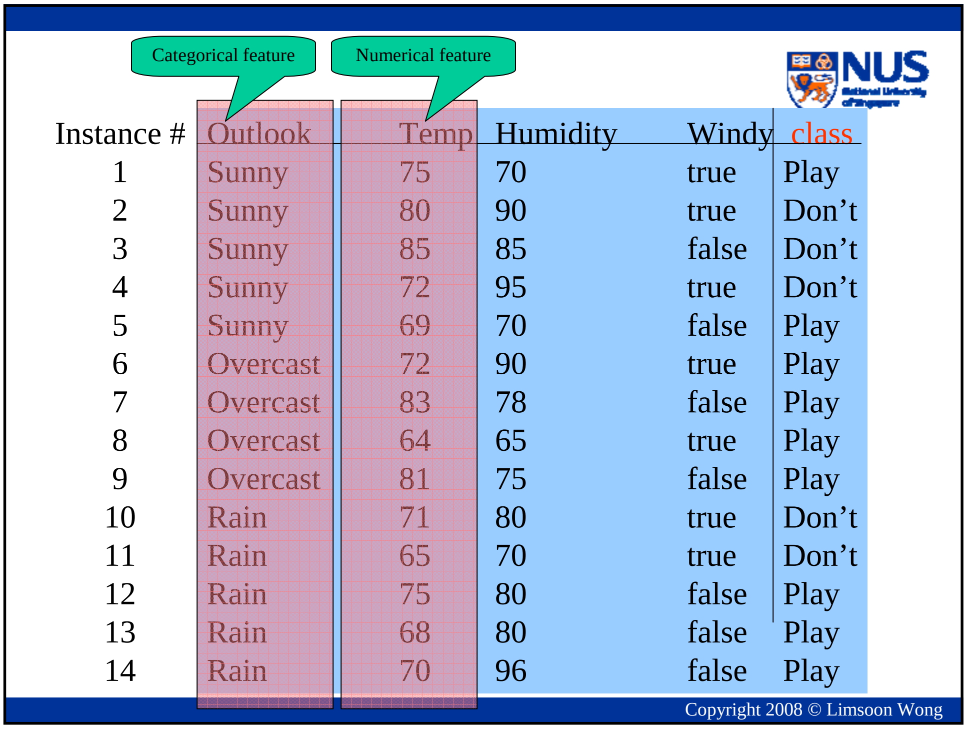
- If the partitions contain a pure class of training instances, then this feature is most discriminatory



Example of Partitions

- **Categorical feature**
  - Number of partitions of the training data is equal to the number of values of this feature

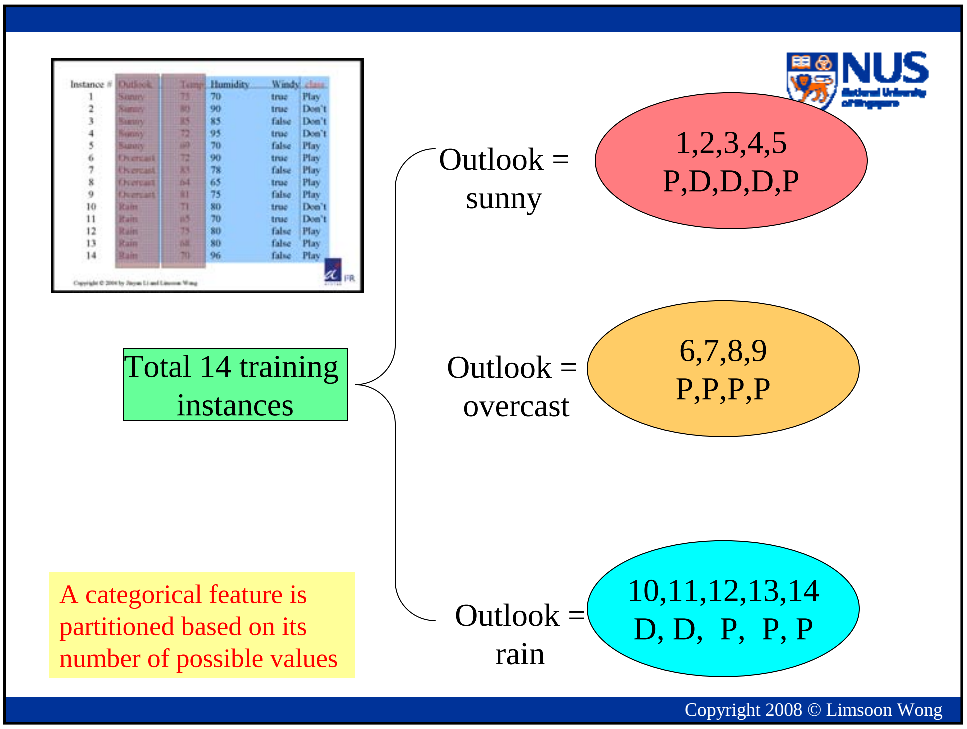
- **Numerical feature**
  - Two partitions



Categorical feature

Numerical feature

Instance #	Outlook	Temp	Humidity	Windy	class
1	Sunny	75	70	true	Play
2	Sunny	80	90	true	Don’t
3	Sunny	85	85	false	Don’t
4	Sunny	72	95	true	Don’t
5	Sunny	69	70	false	Play
6	Overcast	72	90	true	Play
7	Overcast	83	78	false	Play
8	Overcast	64	65	true	Play
9	Overcast	81	75	false	Play
10	Rain	71	80	true	Don’t
11	Rain	65	70	true	Don’t
12	Rain	75	80	false	Play
13	Rain	68	80	false	Play
14	Rain	70	96	false	Play



Instance #	Outlook	Temp	Humidity	Windy	class
1	Sunny	75	70	true	Play
2	Sunny	80	90	true	Don’t
3	Sunny	85	85	false	Don’t
4	Sunny	72	95	true	Don’t
5	Sunny	69	70	false	Play
6	Overcast	72	90	true	Play
7	Overcast	83	78	false	Play
8	Overcast	64	65	true	Play
9	Overcast	81	75	false	Play
10	Rain	71	80	true	Don’t
11	Rain	65	70	true	Don’t
12	Rain	75	80	false	Play
13	Rain	68	80	false	Play
14	Rain	70	96	false	Play

Copyright © 2004 by Jinyan Li and Limsoon Wong

Total 14 training instances

Outlook = sunny

1,2,3,4,5
P,D,D,D,P

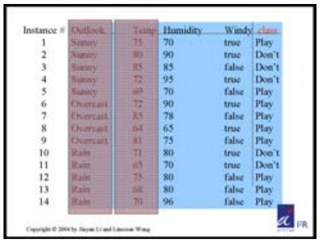
Outlook = overcast

6,7,8,9
P,P,P,P

Outlook = rain

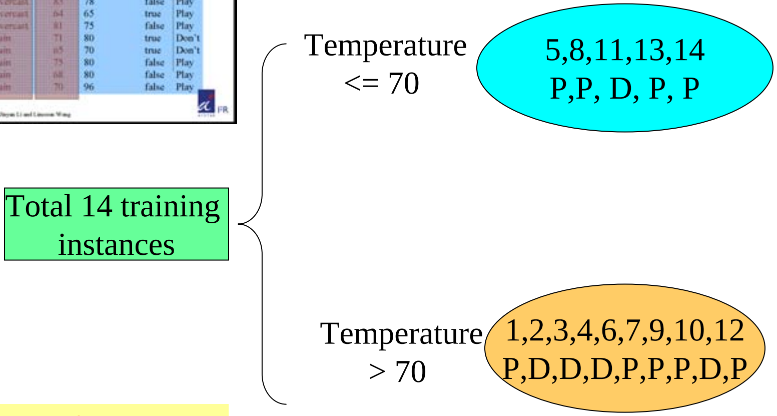
10,11,12,13,14
D, D, P, P, P

A categorical feature is partitioned based on its number of possible values



Instance #	Outlook	Temp	Humidity	Windy	class
1	Sunny	75	70	true	Play
2	Sunny	80	90	true	Don't
3	Sunny	85	85	false	Don't
4	Sunny	72	95	true	Don't
5	Sunny	69	70	false	Play
6	Overcast	72	90	true	Play
7	Overcast	83	78	false	Play
8	Overcast	64	65	true	Play
9	Overcast	81	75	false	Play
10	Rain	71	80	true	Don't
11	Rain	65	70	true	Don't
12	Rain	75	80	false	Play
13	Rain	68	80	false	Play
14	Rain	70	96	false	Play

Copyright © 2004 by Jinyan Li and Limsoon Wong



Total 14 training instances

Temperature <= 70

5,8,11,13,14
P,P, D, P, P

Temperature > 70

1,2,3,4,6,7,9,10,12
P,D,D,D,P,P,P,D,P

A numerical feature is generally partitioned by choosing a “cutting point”

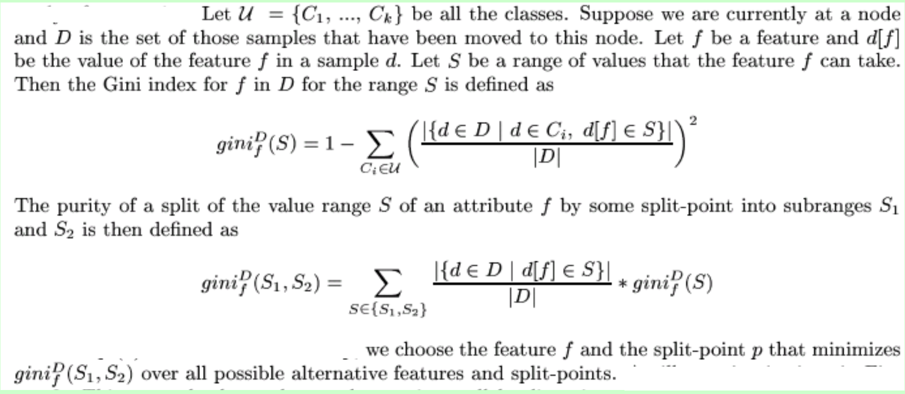
Steps of Decision Tree Construction

- Select the “best” feature as the root node of the whole tree
- Partition the dataset into subsets using this feature so that the subsets are as “pure” as possible
- After partition by this feature, select the best feature (wrt the subset of training data) as the root node of this sub-tree
- Recursively, until the partitions become pure or almost pure

Three Measures to Evaluate Which Feature is Best

- **Gini index**
- **Information gain**
- **Information gain ratio**

Gini Index



Let $\mathcal{U} = \{C_1, ..., C_k\}$ be all the classes. Suppose we are currently at a node and D is the set of those samples that have been moved to this node. Let f be a feature and $d[f]$ be the value of the feature f in a sample d. Let S be a range of values that the feature f can take. Then the Gini index for f in D for the range S is defined as

$$gini_f^D(S) = 1 - \sum_{C_i \in \mathcal{U}} \left(\frac{|\{d \in D \mid d \in C_i,\ d[f] \in S\}|}{|D|} \right)^2$$

The purity of a split of the value range S of an attribute f by some split-point into subranges S_1 and S_2 is then defined as

$$gini_f^D(S_1, S_2) = \sum_{S \in \{S_1, S_2\}} \frac{|\{d \in D \mid d[f] \in S\}|}{|D|} * gini_f^D(S)$$

we choose the feature f and the split-point p that minimizes $gini_f^D(S_1, S_2)$ over all possible alternative features and split-points.

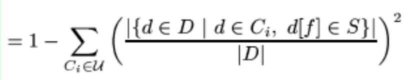
Gini index can be thought of as the expected value of the ratio of the diff of two arbitrary specimens to the mean value of all specimens. Thus the closer it is to 1, the closer you are to the expected "background distribution" of that feature. Conversely, the closer it is to 0, the more "unexpected" the feature is.

$$gini(S) = \frac{diff\ of\ two\ arbitrary\ specimen\ in\ S}{mean\ specimen\ in\ S}$$

$$= \frac{prob(getting\ two\ specimen\ of\ diff\ class\ in\ S)}{prob(getting\ specimen\ of\ some\ class\ in\ S)}$$

$$= \frac{\sum_{i \neq j} prob(getting\ specimen\ of\ class\ i\ in\ S) * prob(getting\ specimen\ of\ class\ j\ in\ S)}{1}$$

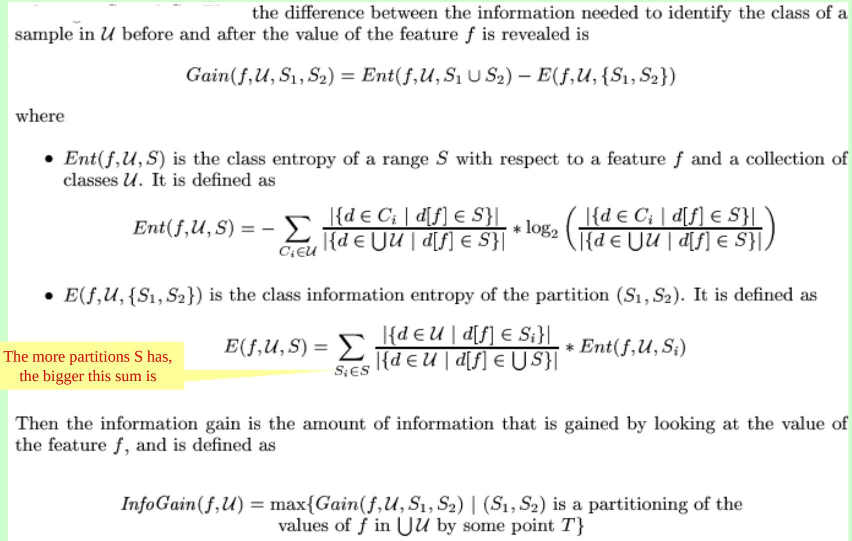
$$= 1 - \sum prob(getting\ specimen\ of\ class\ i\ in\ S)^2$$



$$= 1 - \sum_{C_i \in \mathcal{U}} \left(\frac{|\{d \in D \mid d \in C_i,\ d[f] \in S\}|}{|D|} \right)^2$$

Gini index can be thought of as the expected value of the ratio of the diff of two arbitrary specimens to the mean value of all specimens. Thus the closer it is to 1, the closer you are to the expected "background distribution" of that feature. Conversely, the closer it is to 0, the more "unexpected" the feature is.

Information Gain



the difference between the information needed to identify the class of a sample in $\mathcal{U}$ before and after the value of the feature f is revealed is

$$Gain(f, \mathcal{U}, S_1, S_2) = Ent(f, \mathcal{U}, S_1 \cup S_2) - E(f, \mathcal{U}, \{S_1, S_2\})$$

where

- $Ent(f, \mathcal{U}, S)$ is the class entropy of a range S with respect to a feature f and a collection of classes $\mathcal{U}$. It is defined as

$$Ent(f, \mathcal{U}, S) = -\sum_{C_i \in \mathcal{U}} \frac{|\{d \in C_i \mid d[f] \in S\}|}{|\{d \in \bigcup \mathcal{U} \mid d[f] \in S\}|} * \log_2 \left(\frac{|\{d \in C_i \mid d[f] \in S\}|}{|\{d \in \bigcup \mathcal{U} \mid d[f] \in S\}|} \right)$$

- $E(f, \mathcal{U}, \{S_1, S_2\})$ is the class information entropy of the partition (S_1, S_2). It is defined as

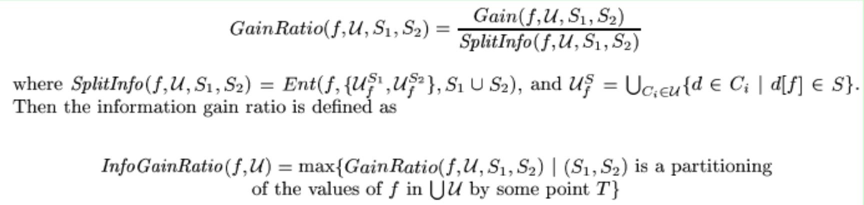
$$E(f, \mathcal{U}, S) = \sum_{S_i \in S} \frac{|\{d \in \mathcal{U} \mid d[f] \in S_i\}|}{|\{d \in \mathcal{U} \mid d[f] \in \bigcup S\}|} * Ent(f, \mathcal{U}, S_i)$$

The more partitions S has, the bigger this sum is

Then the information gain is the amount of information that is gained by looking at the value of the feature f, and is defined as

$$InfoGain(f, \mathcal{U}) = \max\{Gain(f, \mathcal{U}, S_1, S_2) \mid (S_1, S_2) \text{ is a partitioning of the values of } f \text{ in } \bigcup \mathcal{U} \text{ by some point } T\}$$

Information Gain Ratio



$$GainRatio(f,\mathcal{U},S_1,S_2) = \frac{Gain(f,\mathcal{U},S_1,S_2)}{SplitInfo(f,\mathcal{U},S_1,S_2)}$$

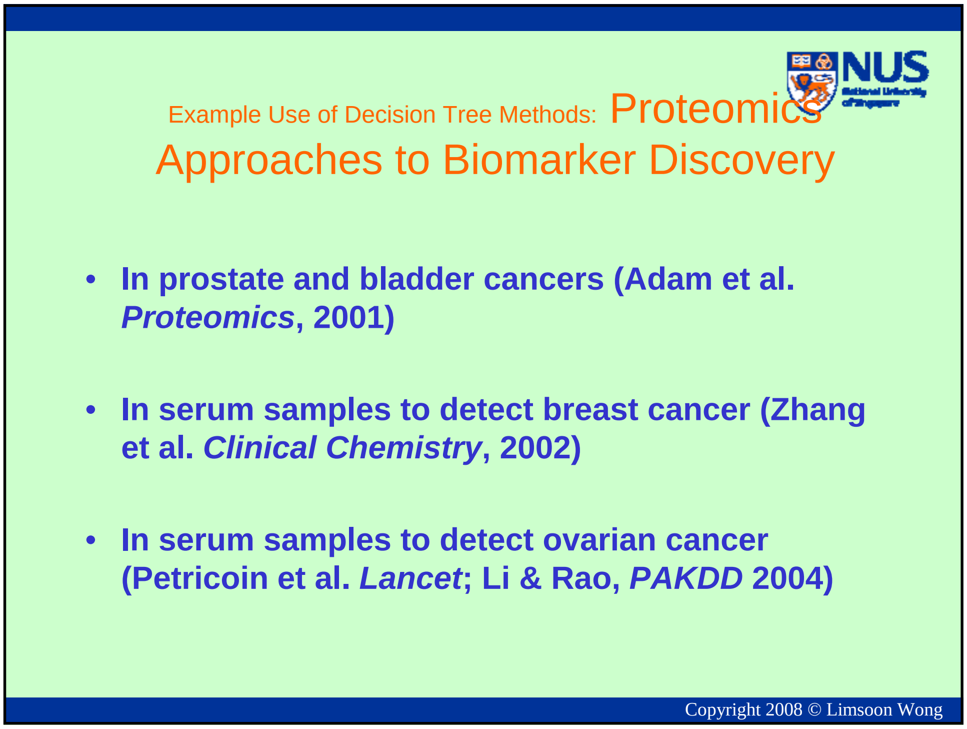
where $SplitInfo(f,\mathcal{U},S_1,S_2) = Ent(f,\{\mathcal{U}_f^{S_1},\mathcal{U}_f^{S_2}\},S_1 \cup S_2)$, and $\mathcal{U}_f^S = \bigcup_{C_i \in \mathcal{U}}\{d \in C_i \mid d[f] \in S\}$.
Then the information gain ratio is defined as

$$InfoGainRatio(f,\mathcal{U}) = \max\{GainRatio(f,\mathcal{U},S_1,S_2) \mid (S_1,S_2) \text{ is a partitioning of the values of } f \text{ in } \bigcup\mathcal{U} \text{ by some point } T\}$$

Characteristics of C4.5 Trees

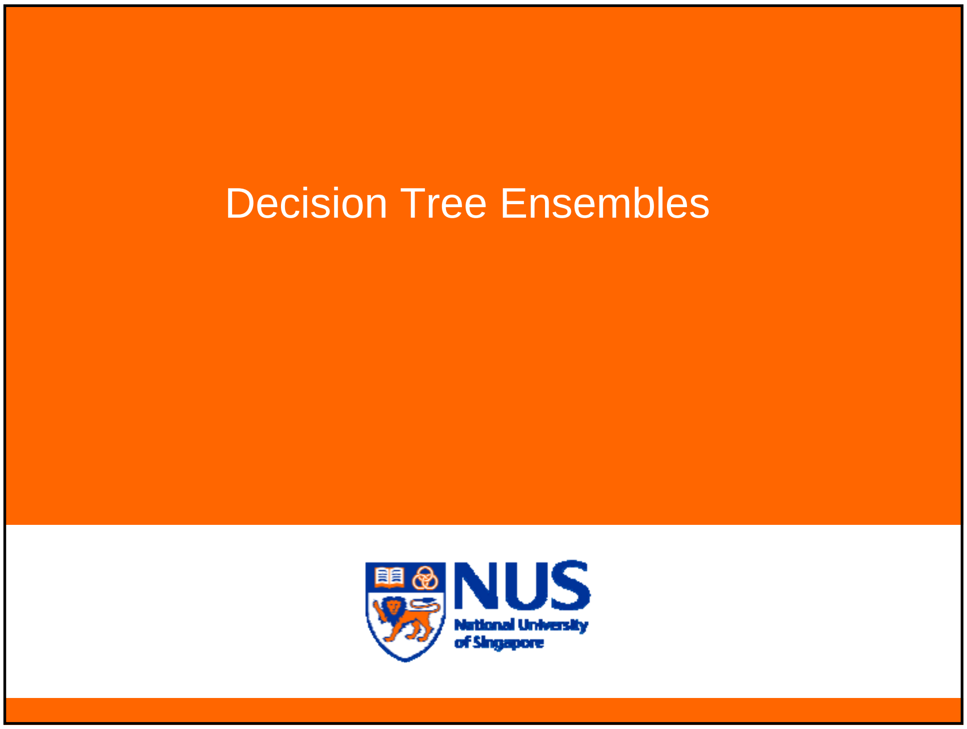
- **Single coverage of training data (elegance)**
- **Divide-and-conquer splitting strategy**
- **Fragmentation problem ⇒ Locally reliable but globally insignificant rules**

Missing many globally significant rules; mislead the system

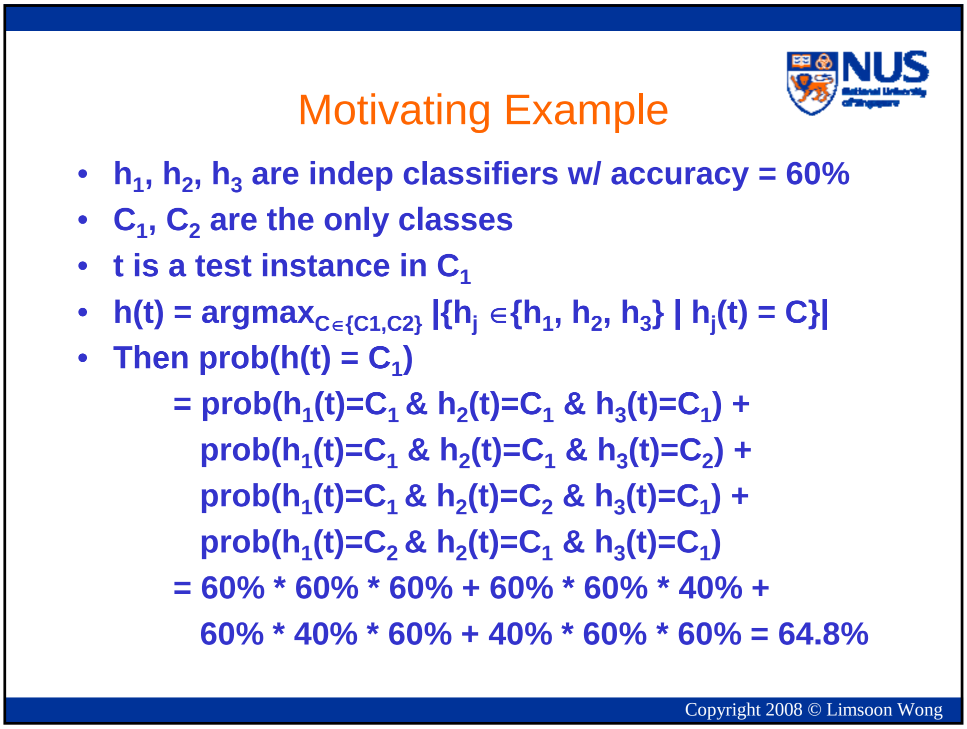


Example Use of Decision Tree Methods: Proteomics Approaches to Biomarker Discovery

- **In prostate and bladder cancers (Adam et al. *Proteomics*, 2001)**
- **In serum samples to detect breast cancer (Zhang et al. *Clinical Chemistry*, 2002)**
- **In serum samples to detect ovarian cancer (Petricoin et al. *Lancet*; Li & Rao, *PAKDD* 2004)**



Decision Tree Ensembles

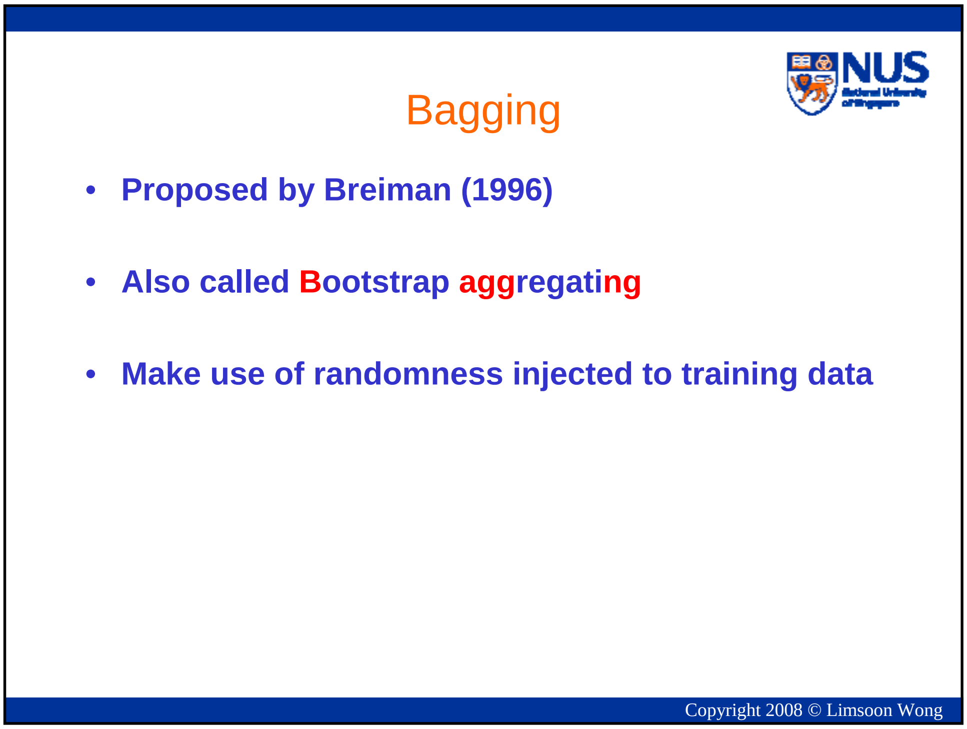


Motivating Example

- h_1, h_2, h_3 are indep classifiers w/ accuracy = 60%
- C_1, C_2 are the only classes
- t is a test instance in C_1
- $h(t) = \text{argmax}_{C \in \{C1,C2\}} |\{h_j \in \{h_1, h_2, h_3\} \mid h_j(t) = C\}|$
- Then $\text{prob}(h(t) = C_1)$

$$= \text{prob}(h_1(t)=C_1 \;\&\; h_2(t)=C_1 \;\&\; h_3(t)=C_1) + \text{prob}(h_1(t)=C_1 \;\&\; h_2(t)=C_1 \;\&\; h_3(t)=C_2) + \text{prob}(h_1(t)=C_1 \;\&\; h_2(t)=C_2 \;\&\; h_3(t)=C_1) + \text{prob}(h_1(t)=C_2 \;\&\; h_2(t)=C_1 \;\&\; h_3(t)=C_1)$$

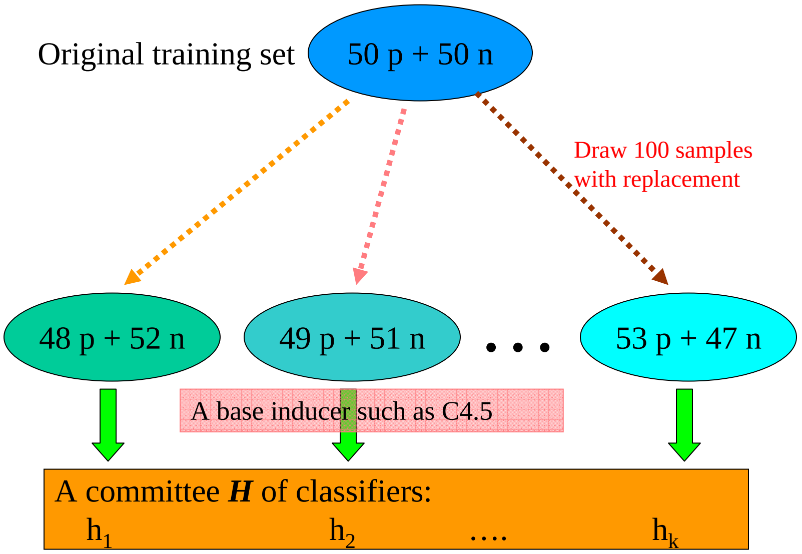
$$= 60\% * 60\% * 60\% + 60\% * 60\% * 40\% + 60\% * 40\% * 60\% + 40\% * 60\% * 60\% = 64.8\%$$



Bagging

- Proposed by Breiman (1996)
- Also called Bootstrap aggregating
- Make use of randomness injected to training data

Main Ideas



Decision Making by Bagging

Given a new test sample T

$$bagged(T) = \operatorname{argmax}_{C_j \in \mathcal{U}} |\{h_i \in \mathcal{H} \mid h_i(T) = C_j\}|$$

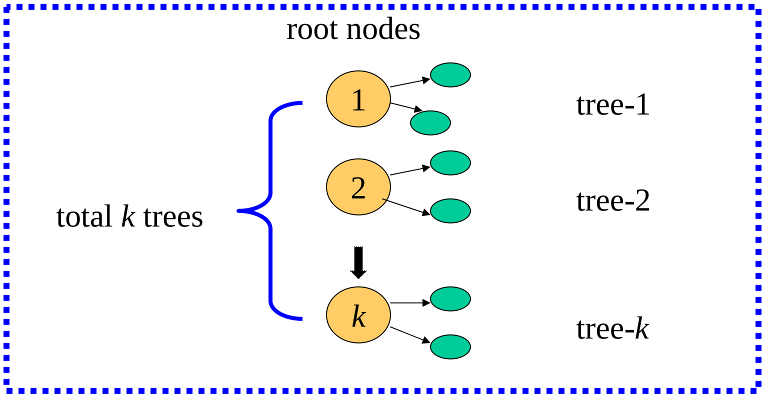
$$\text{where } \mathcal{U} = \{C_1, ..., C_r\}$$

Exercise: What does the above formula mean?

CS4

- Proposed by Li et al (2003)
- CS4: Cascading and Sharing for decision trees
- Doesn’t make use of randomness

Main Ideas



Selection of root nodes is in a cascading manner!

Decision Making by CS4

$$rule_1^{pos}, rule_2^{pos}, \cdots, rule_{k_1}^{pos},$$

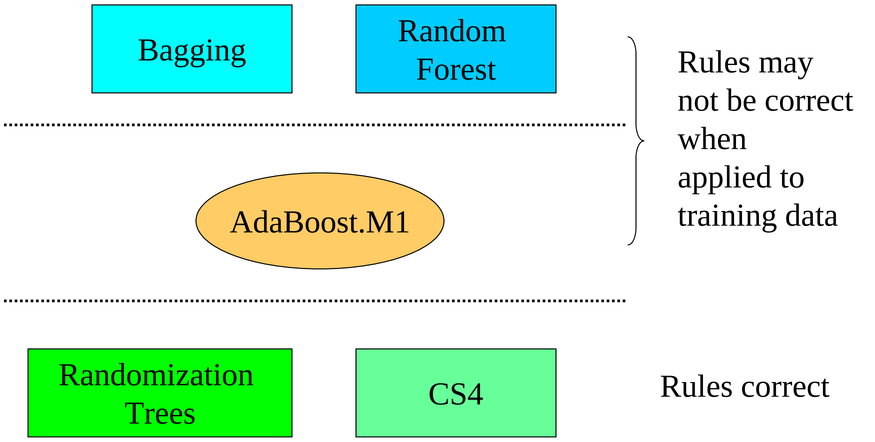
$$rule_1^{neg}, rule_2^{neg}, \cdots, rule_{k_2}^{neg}.$$

$$Score^{pos}(T) = \sum_{i=1}^{k_1} coverage(rule_i^{pos})$$

$$Score^{neg}(T) = \sum_{i=1}^{k_2} coverage(rule_i^{neg})$$

Not equal voting

Summary of Ensemble Classifiers



Bagging

Random Forest

AdaBoost.M1

Rules may not be correct when applied to training data

Randomization Trees

CS4

Rules correct

Exercise: Describe the 3 decision tree ensemble classifiers not explained in this ppt

Other Machine Learning Approaches

Outline

- K-Nearest Neighbour
- Support Vector Machines
- Bayesian Approach
- Hidden Markov Models

Exercise: Name and describe one other commonly used machine learning method

K-Nearest Neighbours

How kNN Works

- Given a new case
- Find k “nearest” neighbours, i.e., k most similar points in the training data set
- Assign new case to the same class to which most of these neighbours belong

- A common “distance” measure betw samples x and y is

$$\sqrt{\Sigma_f (x[f]-y[f])^2}$$

where f ranges over features of the samples

Exercise: What does the formula above mean?

Illustration of kNN (k=8)

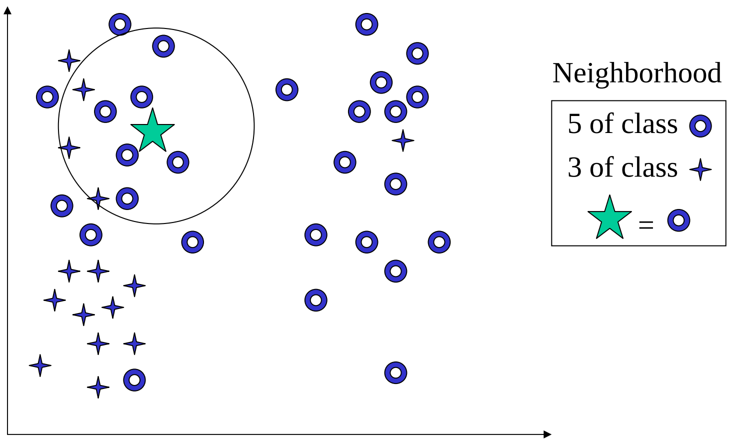


Image credit: Zaki

Some Issues

- Simple to implement
- But need to compare new case against all training cases

⇒ May be slow during prediction

- No need to train
- But need to design distance measure properly

⇒ may need expert for this

- Can’t explain prediction outcome

⇒ Can’t provide a model of the data

Example Use of kNN: Segmentation of White Lesion Matter in MRI

- Anbeek et al, *NeuroImage* 21:1037-1044, 2004
- Use kNN to automated segmentation of white matter lesions in cranial MR images
- Rely on info from T1-weighted, inversion recovery, proton density-weighted, T2-weighted, & fluid attenuation inversion recovery scans

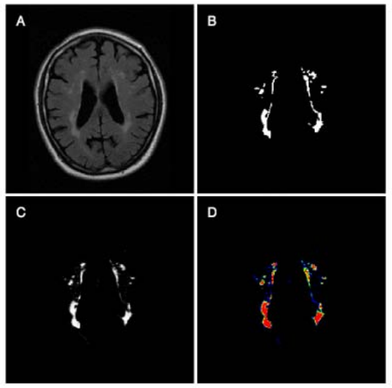


Fig. 3. Classification of a patient with moderate lesion load. (A) FLAIR image, (B) manual segmentation, (C) probability map, (D) segmentations derived from probability map with different thresholds: black: probability $(P) = 0$, blue: $0 < P \leq 0.3$, green: $0.3 < P \leq 0.5$, yellow: $0.5 < P \leq 0.8$, red: $0.8 < P \leq 1$.

Example Use of kNN: Ovarian Cancer Diagnosis Based on SELDI Proteomic Data

- Li et al, *Bioinformatics* 20:1638-1640, 2004
- Use kNN to diagnose ovarian cancers using proteomic spectra
- Data set is from Petricoin et al., *Lancet* 359:572-577, 2002

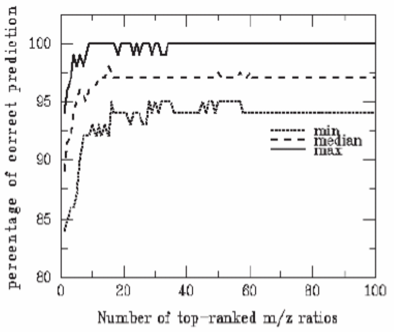
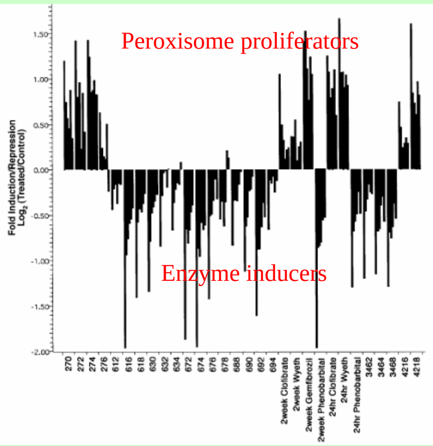


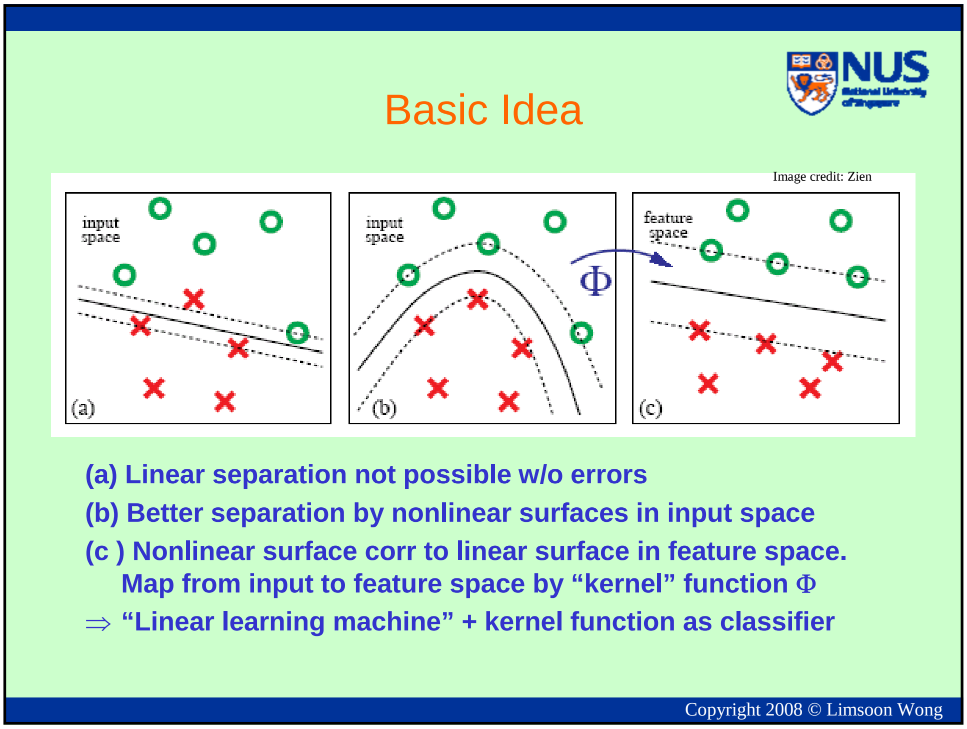
Fig. 1. Minimum, median and maximum of percentages of correct prediction as a function of the number of top-ranked m/z ratios in 50 independent partitions into learning and validation sets.

Example Use of kNN: Prediction of Compound Signature Based on Gene Expr Profiles

- **Hamadeh et al, *Toxicological Sciences* 67:232-240, 2002**
- **Store gene expression profiles corr to biological responses to exposures to known compounds whose toxicological and pathological endpoints are well characterized**
- **Use kNN to infer effects of unknown compound based on gene expr profiles induced by it**



Support Vector Machines



Basic Idea

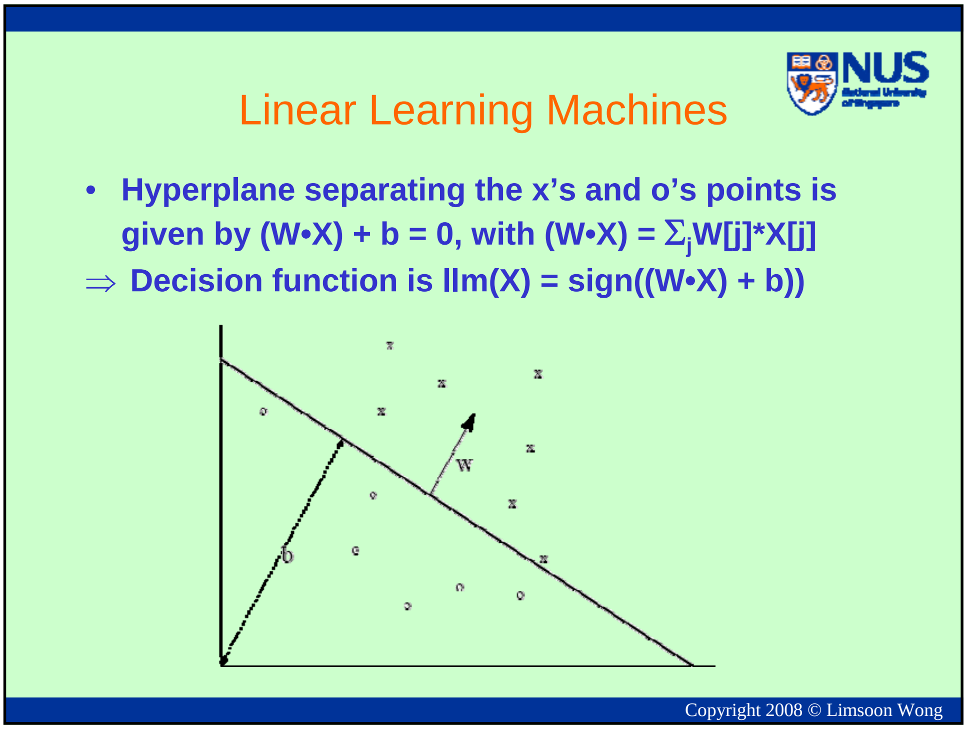
Image credit: Zien

(a) Linear separation not possible w/o errors

(b) Better separation by nonlinear surfaces in input space

(c) Nonlinear surface corr to linear surface in feature space. Map from input to feature space by “kernel” function Φ

⇒ “Linear learning machine” + kernel function as classifier



Linear Learning Machines

- Hyperplane separating the x’s and o’s points is given by (W•X) + b = 0, with (W•X) = Σ_jW[j]*X[j]

⇒ Decision function is llm(X) = sign((W•X) + b))

Linear Learning Machines

- **Solution is a linear combination of training points X_k with labels Y_k**

  $W[j] = \Sigma_k \alpha_k * Y_k * X_k[j]$,

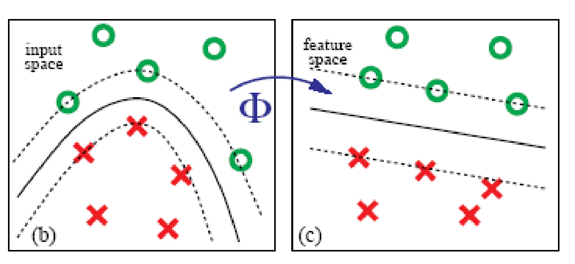
  with $\alpha_k > 0$, and $Y_k = \pm 1$

$\Rightarrow$ llm(X) = $\text{sign}(\Sigma_k \alpha_k * Y_k * (X_k \bullet X) + b)$

“data” appears only in dot product!

Kernel Function

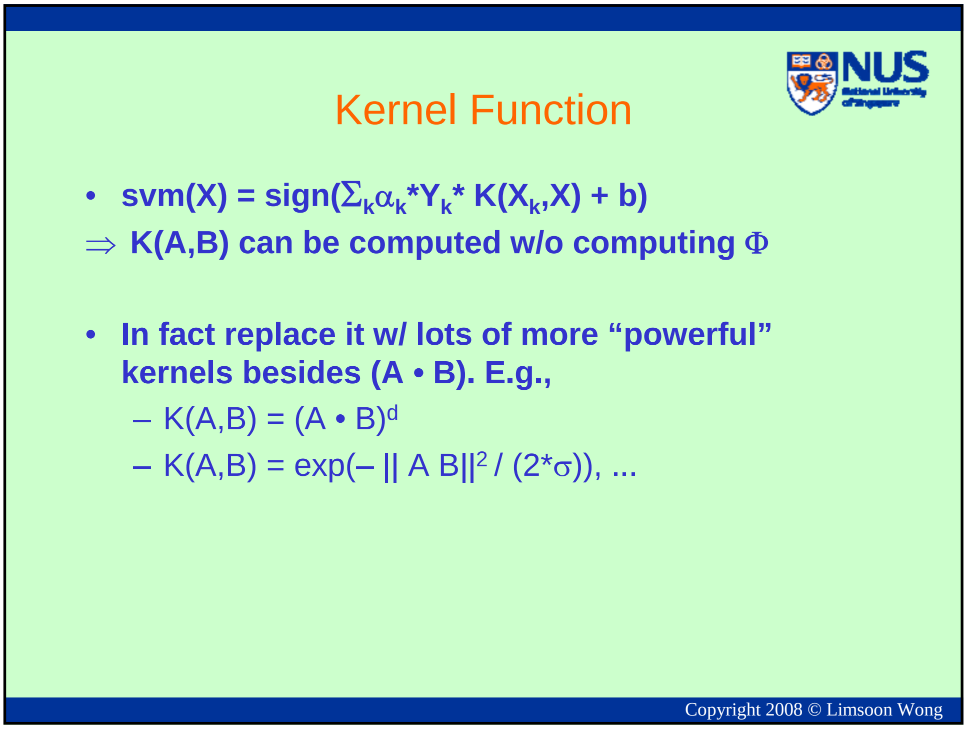
- **llm(X) = $\text{sign}(\Sigma_k \alpha_k * Y_k * (X_k \bullet X) + b)$**



- **svm(X) = $\text{sign}(\Sigma_k \alpha_k * Y_k * (\Phi X_k \bullet \Phi X) + b)$**

$\Rightarrow$ svm(X) = $\text{sign}(\Sigma_k \alpha_k * Y_k * K(X_k, X) + b)$

where $K(X_k, X) = (\Phi X_k \bullet \Phi X)$

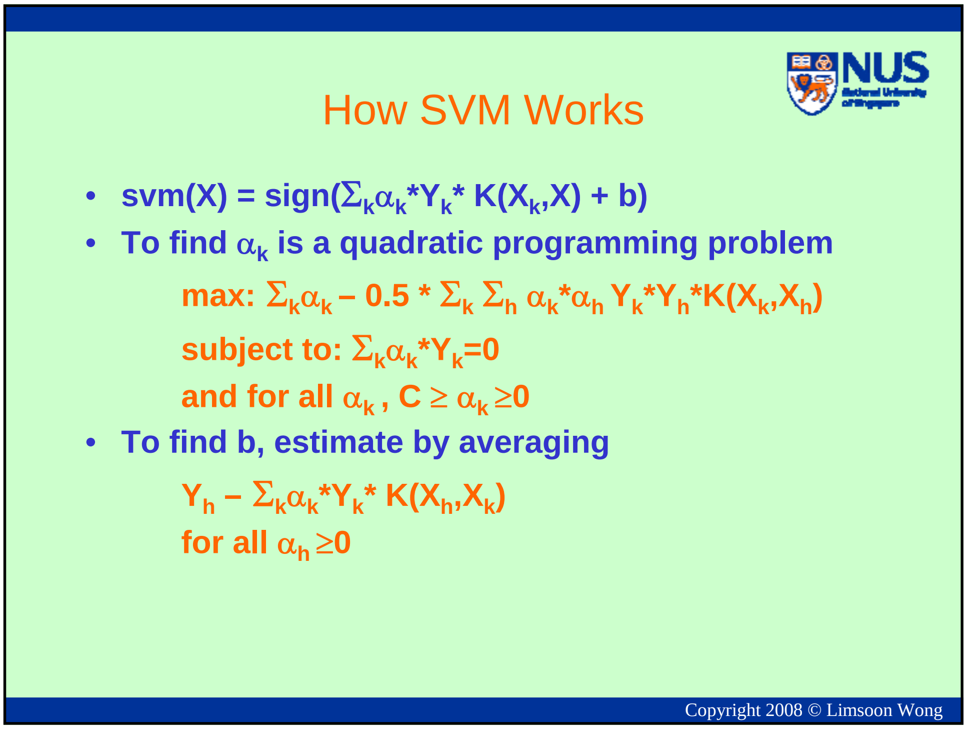


Kernel Function

- **svm(X) = sign($\Sigma_k \alpha_k$*Y_k* K(X_k,X) + b)**

$\Rightarrow$ K(A,B) can be computed w/o computing Φ

- **In fact replace it w/ lots of more “powerful” kernels besides (A • B). E.g.,**
  - K(A,B) = $(A \bullet B)^d$
  - K(A,B) = exp(– || A B$||^2$ / (2*σ)), ...



How SVM Works

- **svm(X) = sign($\Sigma_k \alpha_k$*Y_k* K(X_k,X) + b)**
- **To find α_k is a quadratic programming problem**

max: $\Sigma_k \alpha_k$ – 0.5 * Σ_k Σ_h α_k*α_h Y_k*Y_h*K(X_k,X_h)

subject to: $\Sigma_k \alpha_k$*Y_k=0

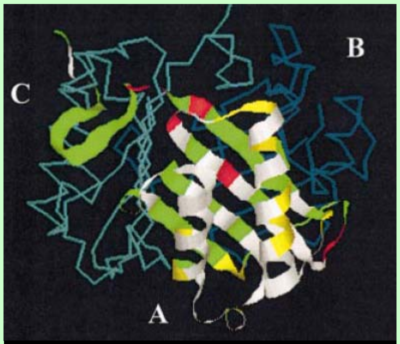
and for all α_k , C $\geq$ α_k $\geq$0

- **To find b, estimate by averaging**

Y_h – $\Sigma_k \alpha_k$*Y_k* K(X_h,X_k)
for all α_h $\geq$0

Example Use of SVM: Prediction of Protein-Protein Interaction Sites From Sequences

- **Koike et al, *Protein Engineering Design & Selection* 17:165-173, 2004**
- **Identification of protein-protein interaction sites is impt for mutant design & prediction of protein-protein networks**
- **Interaction sites were predicted here using SVM & profiles of sequentially/spatially neighbouring residues**



Legend: green=TP, white=TN, yellow=FN, red=FP
A: human macrophage migration inhibitory factor
B & C: the binding proteins

Example Use of SVM: Prediction of Gene Function From Gene Expression

- **Brown et al., *PNAS* 91:262-267, 2000**
- **Use SVM to identify sets of genes w/ a c'mon function based on their expression profiles**
- **Use SVM to predict functional roles of uncharacterized yeast ORFs based on their expression profiles**

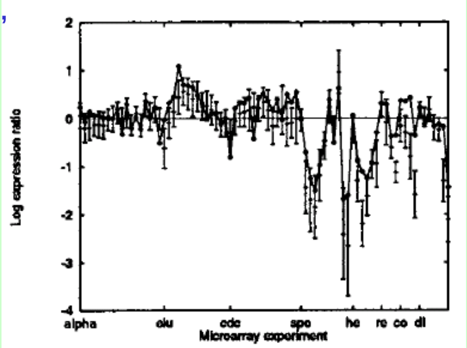
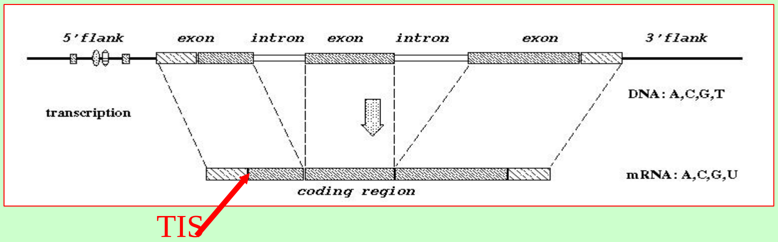


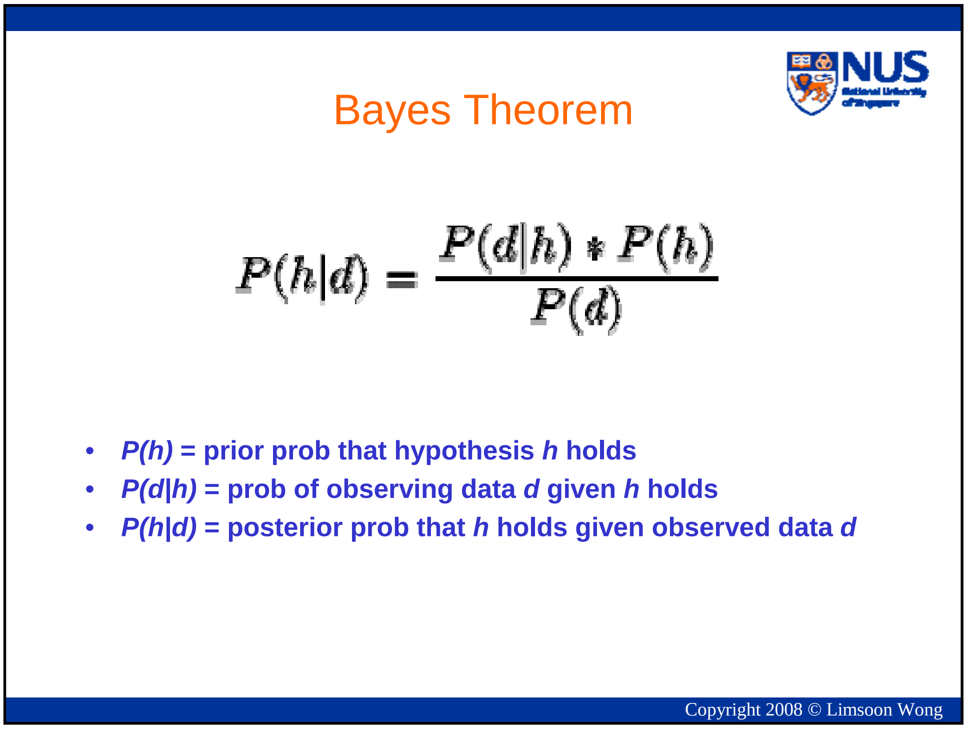
Fig. 1. Expression profile of YPL037C compared with the MYGD class of cytoplasmic ribosomal proteins. YPL037C is classified as a ribosomal protein by the SVMs but is not included in the class by MYGD. The figure shows the expression profile for YPL037C, along with standard deviation bars for the class of cytoplasmic ribosomal proteins. Ticks along the *x* axis represent the beginnings of experimental series.

Example Use of SVM: Recognition of Protein Translation Initiation Sites



- Zien et al., *Bioinformatics* 16:799-807, 2000
- Use SVM to recognize protein translation initiation sites from genomic sequences
- Raw data set is same as Liu & Wong, *JBCB* 1:139-168, 2003

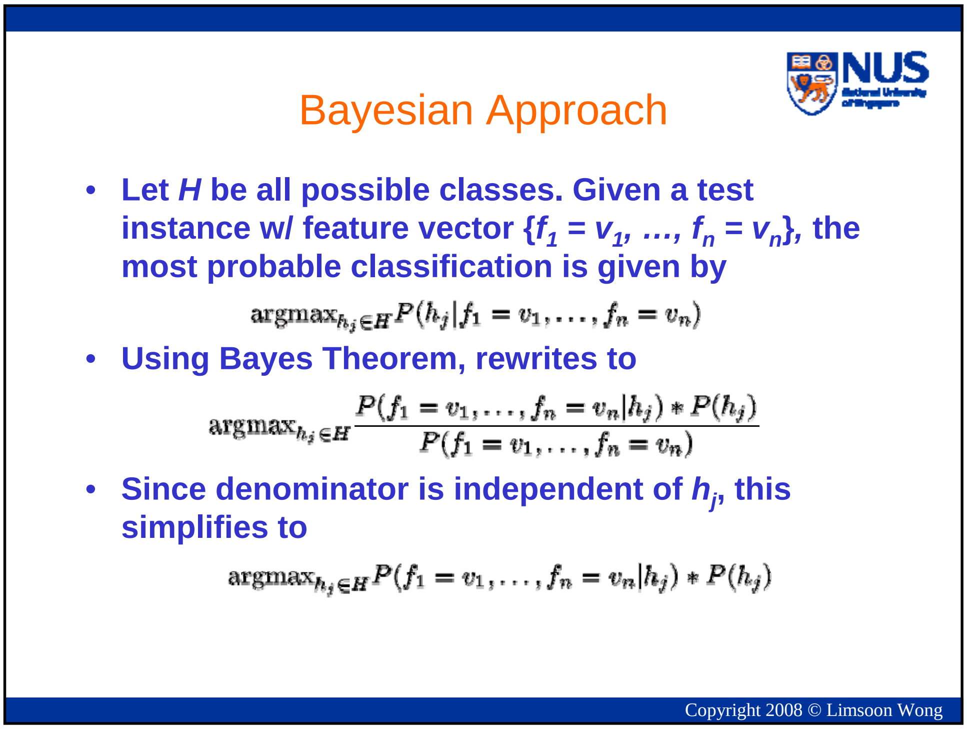
Bayesian Approach



Bayes Theorem

$$P(h|d) = \frac{P(d|h) * P(h)}{P(d)}$$

- *P(h)* = prior prob that hypothesis *h* holds
- *P(d|h)* = prob of observing data *d* given *h* holds
- *P(h|d)* = posterior prob that *h* holds given observed data *d*



Bayesian Approach

- Let *H* be all possible classes. Given a test instance w/ feature vector {f_1 = v_1, …, f_n = v_n}, the most probable classification is given by

$$\text{argmax}_{h_j \in H} P(h_j | f_1 = v_1, \ldots, f_n = v_n)$$

- Using Bayes Theorem, rewrites to

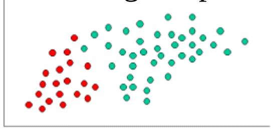
$$\text{argmax}_{h_j \in H} \frac{P(f_1 = v_1, \ldots, f_n = v_n | h_j) * P(h_j)}{P(f_1 = v_1, \ldots, f_n = v_n)}$$

- Since denominator is independent of h_j, this simplifies to

$$\text{argmax}_{h_j \in H} P(f_1 = v_1, \ldots, f_n = v_n | h_j) * P(h_j)$$

An Example

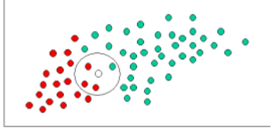
Training samples



$$\text{Prior probability for GREEN} \propto \frac{\text{Number of GREEN objects}}{\text{Total number of objects}} = 40/60$$

$$\text{Prior probability for RED} \propto \frac{\text{Number of RED objects}}{\text{Total number of objects}} = 20/60$$

A testing instance X



$$\text{Likelihood of X given GREEN} \propto \frac{\text{Number of GREEN in the vicinity of X}}{\text{Total number of GREEN cases}} = 1/40$$

$$\text{Likelihood of X given RED} \propto \frac{\text{Number of RED in the vicinity of X}}{\text{Total number of RED cases}} = 3/20$$

$$\text{Posterior probability of } X \text{ being GREEN} \propto$$
$$\text{Prior probability of GREEN} \times \text{Likelihood of } X \text{ given GREEN}$$
$$= \frac{4}{6} \times \frac{1}{40} = \frac{1}{60}$$

$$\text{Posterior probability of } X \text{ being RED} \propto$$
$$\text{Prior probability of RED} \times \text{Likelihood of } X \text{ given RED}$$
$$= \frac{2}{6} \times \frac{3}{20} = \frac{1}{20}$$

we classify X as RED since its class membership achieves the largest posterior probability

Source: http://www.statsoft.com/textbook/stnaiveb.html

Copyright 2008 © Limsoon Wong

Naïve Bayes

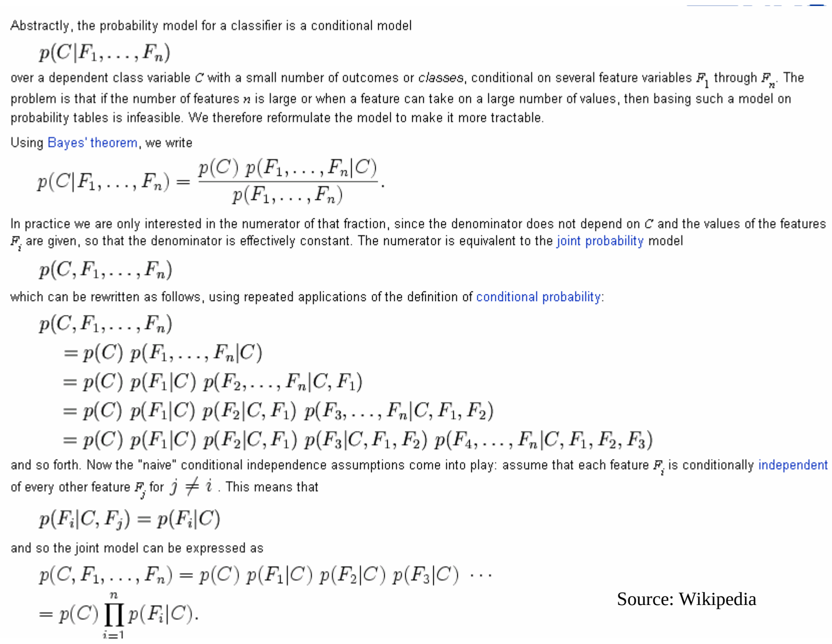
- But estimating $P(f_1=v_1, \ldots, f_n=v_n|h_j)$ accurately may not be feasible unless training data set is sufficiently large
- "Solved" by assuming $f_1, \ldots, f_n$ are conditionally independent of each other
- Then $\text{argmax}_{h_j \in H} P(f_1 = v_1, \ldots, f_n = v_n|h_j) * P(h_j)$

$$= \text{argmax}_{h_j \in H} \prod_i P(f_i = v_i|h_j) * P(h_j)$$

- where $P(h_j)$ and $P(f_i=v_i|h_j)$ can often be estimated reliably from typical training data set

Exercise: How do you estimate $P(h_j)$ and $P(f_j=v_j|h_j)$?

Copyright 2008 © Limsoon Wong



Abstractly, the probability model for a classifier is a conditional model

$$p(C|F_1,\dots,F_n)$$

over a dependent class variable C with a small number of outcomes or *classes*, conditional on several feature variables F_1 through F_n. The problem is that if the number of features n is large or when a feature can take on a large number of values, then basing such a model on probability tables is infeasible. We therefore reformulate the model to make it more tractable.

Using Bayes' theorem, we write

$$p(C|F_1,\dots,F_n) = \frac{p(C)\ p(F_1,\dots,F_n|C)}{p(F_1,\dots,F_n)}.$$

In practice we are only interested in the numerator of that fraction, since the denominator does not depend on C and the values of the features F_i are given, so that the denominator is effectively constant. The numerator is equivalent to the joint probability model

$$p(C,F_1,\dots,F_n)$$

which can be rewritten as follows, using repeated applications of the definition of conditional probability:

$$\begin{aligned}
&p(C,F_1,\dots,F_n)\\
&\quad= p(C)\ p(F_1,\dots,F_n|C)\\
&\quad= p(C)\ p(F_1|C)\ p(F_2,\dots,F_n|C,F_1)\\
&\quad= p(C)\ p(F_1|C)\ p(F_2|C,F_1)\ p(F_3,\dots,F_n|C,F_1,F_2)\\
&\quad= p(C)\ p(F_1|C)\ p(F_2|C,F_1)\ p(F_3|C,F_1,F_2)\ p(F_4,\dots,F_n|C,F_1,F_2,F_3)
\end{aligned}$$

and so forth. Now the "naive" conditional independence assumptions come into play: assume that each feature F_i is conditionally independent of every other feature F_j for $j \neq i$. This means that

$$p(F_i|C,F_j) = p(F_i|C)$$

and so the joint model can be expressed as

$$\begin{aligned}
&p(C,F_1,\dots,F_n) = p(C)\ p(F_1|C)\ p(F_2|C)\ p(F_3|C)\ \cdots\\
&= p(C)\prod_{i=1}^{n} p(F_i|C).
\end{aligned}$$

Source: Wikipedia

Independence vs Conditional Independence

- **Independence: P(A,B) = P(A) * P(B)**
- **Conditional Independence: P(A,B|C) = P(A|C) * P(B|C)**
- **Indep does not imply conditional indep**
  - Consider tossing a fair coin twice
    - **A is event of getting head in 1st toss**
    - **B is event of getting head in 2nd toss**
    - **C is event of getting exactly one head**
  - Then A={HT, HH}, B={HH, TH} and C={HT, TH}
  - P(A,B|C) =P({HH}|C)=0
  - P(A|C) = P(A,C)/P(C) =P({HT})/P(C)=(1/4)/(1/2) =1/2
  - Similarly, P(B|C) =1/2

Example Use of Bayesian: Design of Screens for Macromolecular Crystallization

- **Hennessy et al., *Acta Cryst* D56:817-827, 2000**
- **Xtallization of proteins requires search of expt settings to find right conditions for diffraction-quality xtals**
- **BMCD is a db of known xtallization conditions**
- **Use Bayes to determine prob of success of a set of expt conditions based on BMCD**

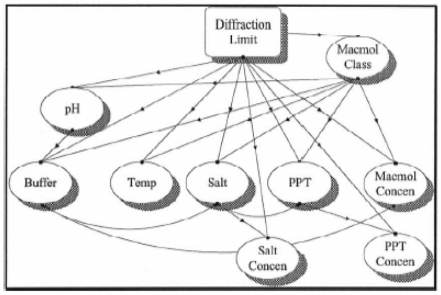
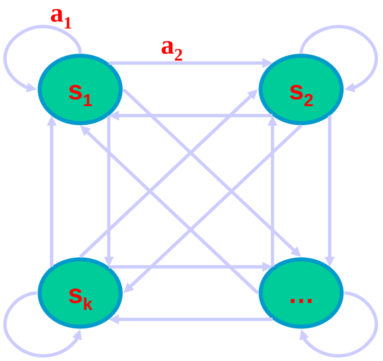


Figure 1
Crystallization parameter dependency graph. The graph represents the parameters included in the calculation of the estimated probability of success and their dependencies. A connecting arc from pH to buffer indicates that the probability distribution for the buffer may depend on the value of the pH. The lack of a connecting arc between two parameters reflects conditional independence (the probability distribution for a parameter is independent of the value of the other parameter).

Hidden Markov Models

What is a HMM

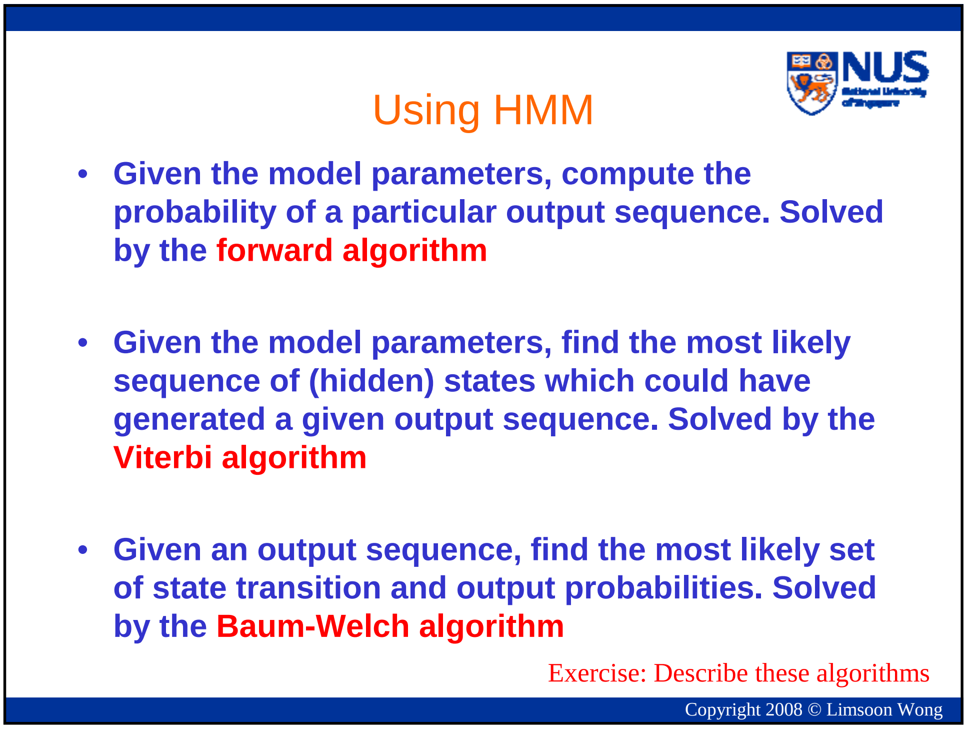
- **HMM is a stochastic generative model for sequences**
- **Defined by model parameters**
  - finite set of states *S*
  - finite alphabet *A*
  - transition prob matrix *T*
  - emission prob matrix *E*
- **Move from state to state according to *T* while emitting symbols according to *E***



The Order of a HMM

- **In *n*th order HMM, *T* & *E* depend on all *n* previous states**
- **E.g., for 1st order HMM, given emissions $X = x_1, x_2, \ldots$, & states $S = s_1, s_2, \ldots$, the prob of this seq is**

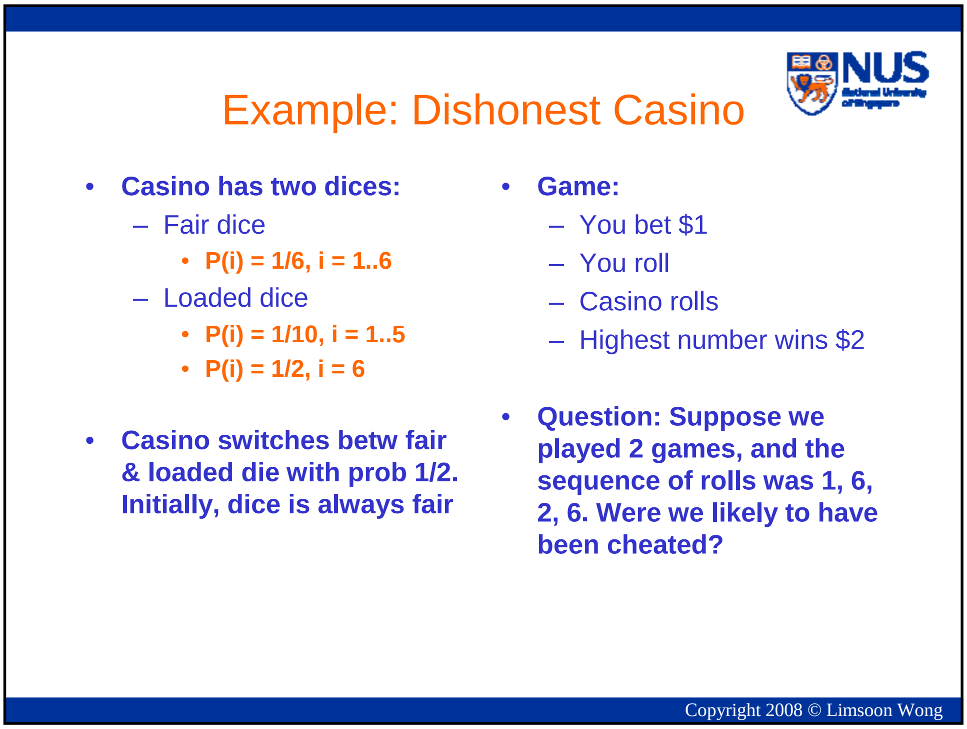
$$Prob(X,S) = \prod_i Prob(x_i|s_i) = \prod_i E(x_i|s_i) * T(s_{i-1}, s_i)$$



Using HMM

- **Given the model parameters, compute the probability of a particular output sequence. Solved by the forward algorithm**
- **Given the model parameters, find the most likely sequence of (hidden) states which could have generated a given output sequence. Solved by the Viterbi algorithm**
- **Given an output sequence, find the most likely set of state transition and output probabilities. Solved by the Baum-Welch algorithm**

Exercise: Describe these algorithms

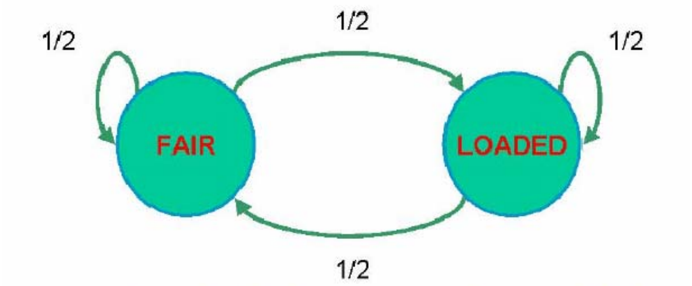


Example: Dishonest Casino

- **Casino has two dices:**
  - Fair dice
    - **P(i) = 1/6, i = 1..6**
  - Loaded dice
    - **P(i) = 1/10, i = 1..5**
    - **P(i) = 1/2, i = 6**
- **Casino switches betw fair & loaded die with prob 1/2. Initially, dice is always fair**

- **Game:**
  - You bet $1
  - You roll
  - Casino rolls
  - Highest number wins $2
- **Question: Suppose we played 2 games, and the sequence of rolls was 1, 6, 2, 6. Were we likely to have been cheated?**

“Visualization” of Dishonest Casino



Emission Matrix		Transition Matrix
E(1|Fair) = 1/6	E(1|Loaded) = 1/10	T(Loaded,Loaded) = 1/2
E(2|Fair) = 1/6	E(2|Loaded) = 1/10	T(Loaded,Fair) = 1/2
E(3|Fair) = 1/6	E(3|Loaded) = 1/10	T(Fair,Fair) = 1/2
E(4|Fair) = 1/6	E(4|Loaded) = 1/10	T(Fair,Loaded) = 1/2
E(5|Fair) = 1/6	E(5|Loaded) = 1/10	T(?,Fair) = 1.0
E(6|Fair) = 1/6	E(6|Loaded) = 1/2	T(?,Loaded) = 0.0

1, 6, 2, 6?
We were probably cheated...

$$\begin{aligned} Prob(X, S = Fair, Fair, Fair, Fair) &= E(1|Fair) * T(?, Fair) * \\ &\quad E(6|Fair) * T(Fair, Fair) * \\ &\quad E(2|Fair) * T(Fair, Fair) * \\ &\quad E(6|Fair) * T(Fair, Fair) \\ &= \frac{1}{6} * 1 * \frac{1}{6} * \frac{1}{2} * \frac{1}{6} * \frac{1}{2} * \frac{1}{6} * \frac{1}{2} \\ &= 9.6451 * 10^{-5} \end{aligned}$$

$$\begin{aligned} Prob(X, S = Fair, Loaded, Fair, Loaded) &= E(1|Fair) * T(?, Fair) * \\ &\quad E(6|Loaded) * T(Fair, Loaded) * \\ &\quad E(2|Loaded) * T(Loaded, Fair) * \\ &\quad E(6|Loaded) * T(Fair, Loaded) \\ &= \frac{1}{6} * 1 * \frac{1}{2} * \frac{1}{2} * \frac{1}{6} * \frac{1}{2} * \frac{1}{2} * \frac{1}{2} \\ &= 8.6806 * 10^{-4} \end{aligned}$$

Example Use of HMM: Protein Families Modelling

- Baldi et al., *PNAS* 91:1059-1063, 1994
- HMM is used to model families of biological sequences, such as kinases, globins, & immunoglobulins

- Bateman et al., *NAR* 32:D138-D141, 2004
- HMM is used to model 6190 families of protein domains in Pfam

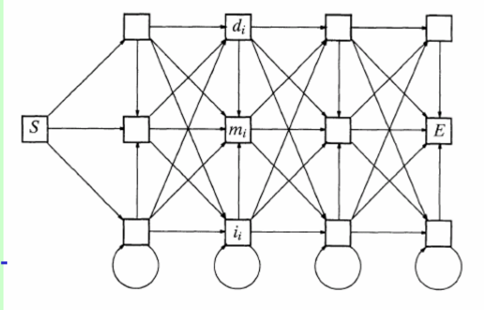


FIG. 1. HMM architecture. S and E are the start and end states. Sequence of main states m_i is the backbone. Side states d_i (resp. i_i) correspond to deletions (resp. insertions).

Example Use of HMM: Gene Finding in Bacterial Genomes

- Borodovsky et al., *NAR* 23:3554-3562, 1995

- Investigated statistical features of 3 classes (wrt level of codon usage bias) of E. coli genes

- HMM for nucleotide sequences of each class was developed

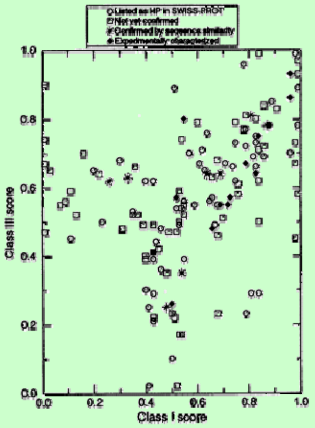
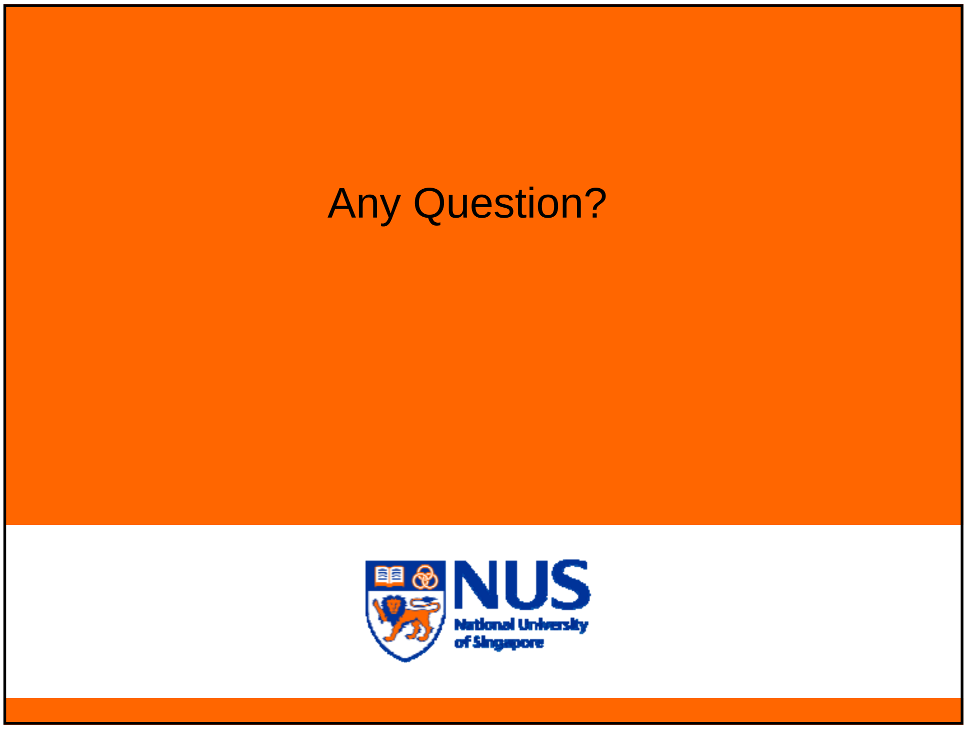


Figure 4. Distribution of GeneMark scores for 126 new genes. The x axis represents the score computed by GM5_ECO1 program, y axis represents the score computed by GM4_ECO3 program. The quadrant x < 0.4, y < 0.4 is empty since a threshold of 0.4 was applied.

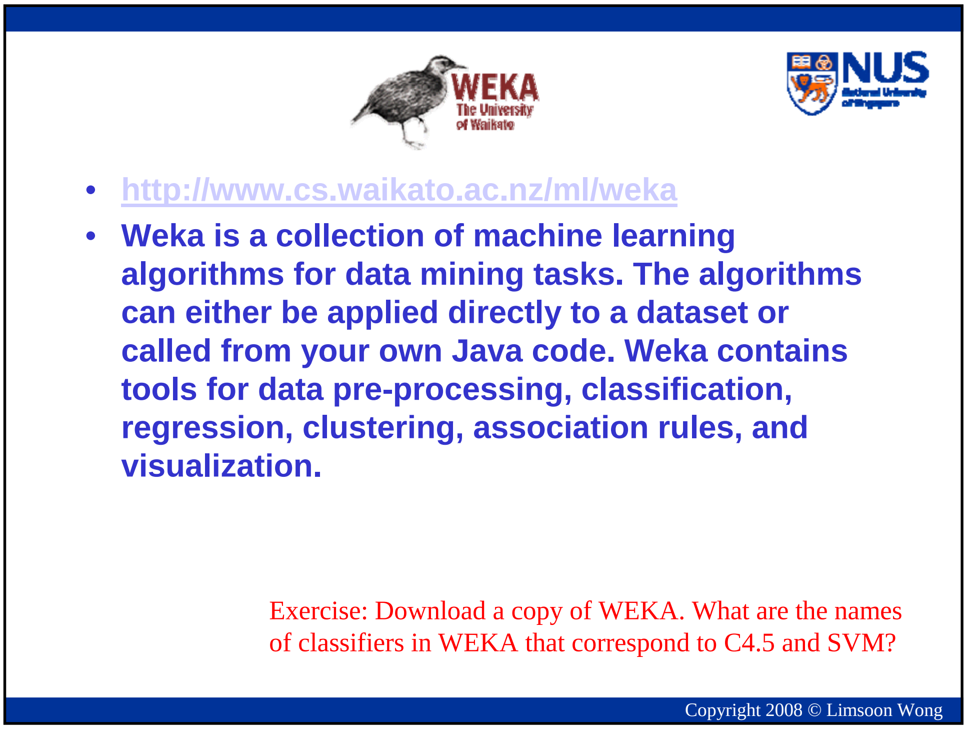
Concluding Remarks…

What have we learned?

- **Decision Trees**
- **Decision Trees Ensembles**
  - Bagging
  - CS4
- **Other Methods**
  - K-Nearest Neighbour
  - Support Vector Machines
  - Bayesian Approach
  - Hidden Markov Models

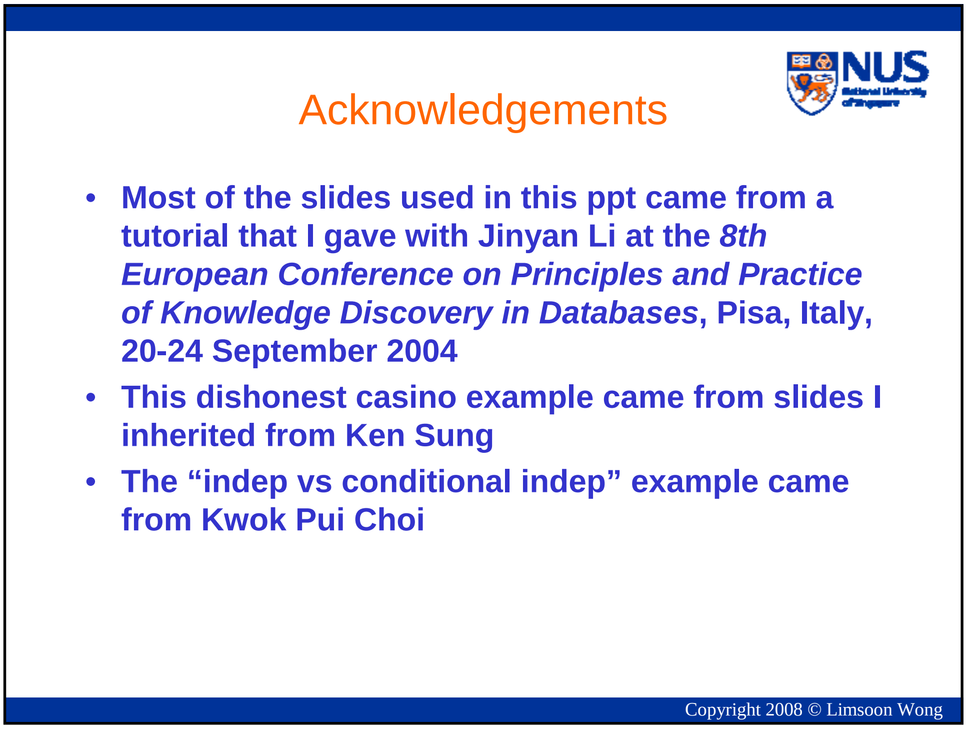


Any Question?



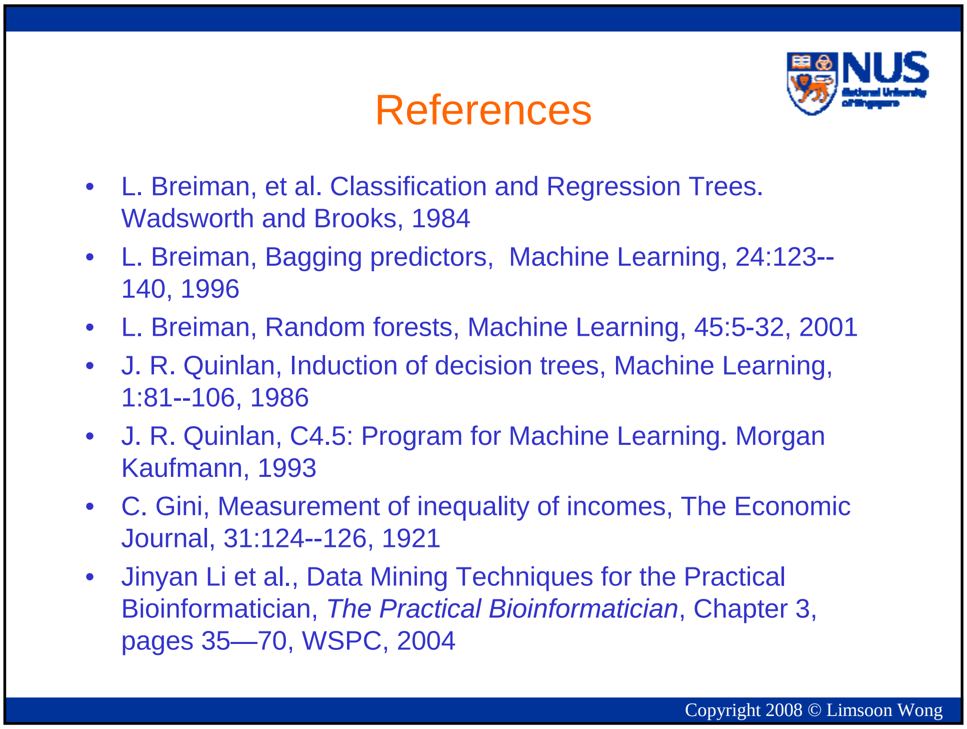
- http://www.cs.waikato.ac.nz/ml/weka
- Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization.

Exercise: Download a copy of WEKA. What are the names of classifiers in WEKA that correspond to C4.5 and SVM?



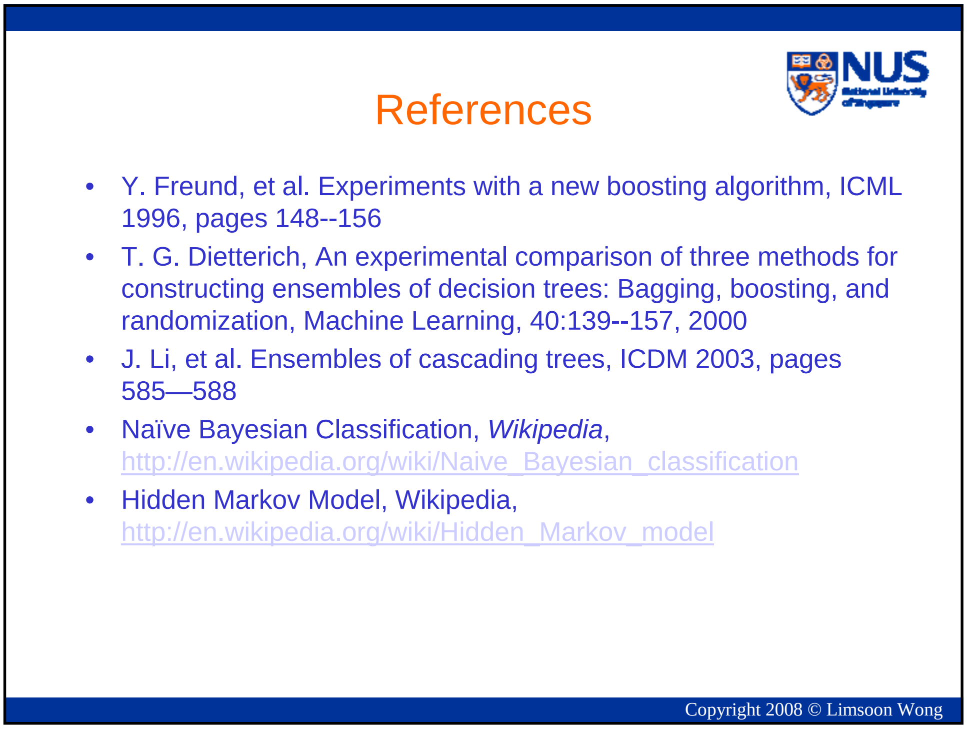
Acknowledgements

- **Most of the slides used in this ppt came from a tutorial that I gave with Jinyan Li at the *8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, Pisa, Italy, 20-24 September 2004**
- **This dishonest casino example came from slides I inherited from Ken Sung**
- **The "indep vs conditional indep" example came from Kwok Pui Choi**



References

- L. Breiman, et al. Classification and Regression Trees. Wadsworth and Brooks, 1984
- L. Breiman, Bagging predictors, Machine Learning, 24:123--140, 1996
- L. Breiman, Random forests, Machine Learning, 45:5-32, 2001
- J. R. Quinlan, Induction of decision trees, Machine Learning, 1:81--106, 1986
- J. R. Quinlan, C4.5: Program for Machine Learning. Morgan Kaufmann, 1993
- C. Gini, Measurement of inequality of incomes, The Economic Journal, 31:124--126, 1921
- Jinyan Li et al., Data Mining Techniques for the Practical Bioinformatician, *The Practical Bioinformatician*, Chapter 3, pages 35—70, WSPC, 2004



References

- Y. Freund, et al. Experiments with a new boosting algorithm, ICML 1996, pages 148--156
- T. G. Dietterich, An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization, Machine Learning, 40:139--157, 2000
- J. Li, et al. Ensembles of cascading trees, ICDM 2003, pages 585—588
- Naïve Bayesian Classification, *Wikipedia*, http://en.wikipedia.org/wiki/Naive_Bayesian_classification
- Hidden Markov Model, Wikipedia, http://en.wikipedia.org/wiki/Hidden_Markov_model