

KHAI THÁC DỮ LIỆU & KHAI PHÁ TRI THỨC

Data Mining & Knowledge Discovery

Bài 5. Phân lớp/ Classification

Mã MH: 505043

TS. HOÀNG Anh

Chương 8. Phân loại: Các khái niệm cơ bản

- **Bài toán Classification:** Các khái niệm cơ bản
- **Cây quyết định/ Decision Tree Induction**
- **Phân loại Bayes/ Bayes Classification Methods**
- **Phân loại dựa trên luật/ Rule-Based Classification**
- **Đánh giá và lựa chọn model/ Model Evaluation and Selection**
- **Những kỹ thuật tăng độ chính xác trong bài toán phân loại: Ensemble Methods**
- **Tóm tắt/ Summary**



Học có giám sát vs. Học không giám sát

Supervised vs. Unsupervised Learning

- Học có giám sát/ Supervised learning (Phân loại/ classification)
 - Giám sát/ Supervision: Dữ liệu huấn luyện có nhãn/ labels cho biết lớp/class dữ liệu đó thuộc về
 - Dữ liệu mới được phân lớp dựa trên dữ liệu huấn luyện
- Học không giám sát/ Unsupervised learning (Gom cụm/ clustering)
 - Dữ liệu huấn luyện không bao gồm nhãn
 - Tập các điểm dữ liệu được gom cụm dựa trên sự tương đồng hay khác biệt.

Vấn đề dự đoán:

Dự đoán lớp vs. Dự đoán số

■ Phân lớp/ Classification

- Dự đoán nhãn lớp phân loại (rời rạc hoặc danh nghĩa)
- Phân loại dữ liệu (constructs a model) dựa trên tập huấn luyện và giá trị của nhãn (**class labels**) trong thuộc tính được phân loại, và sử dụng model để phân lớp dữ liệu mới.

■ Dự đoán số/ Numeric Prediction

- Mô hình các hàm có giá trị liên tục, ví dụ dự đoán giá trị bị mất/ missing values

■ Các ứng dụng:

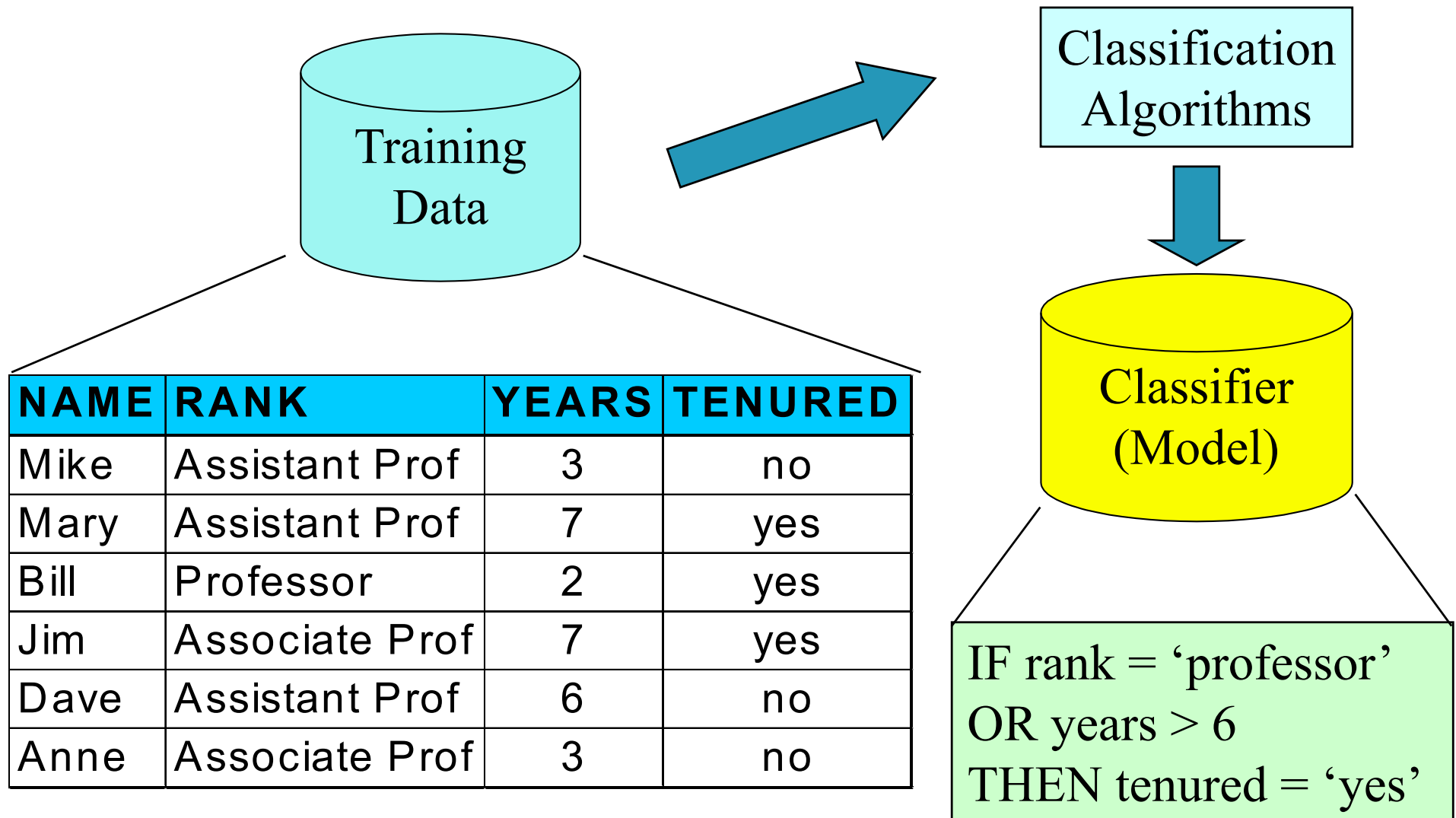
- Phê duyệt tín dụng/ cho vay: Credit/loan approval
- Chẩn đoán y tế/ Medical diagnosis
- Phát hiện gian lận/ Fraud detection
- Phân loại trang/ Web page categorization

Phân lớp/ Classification

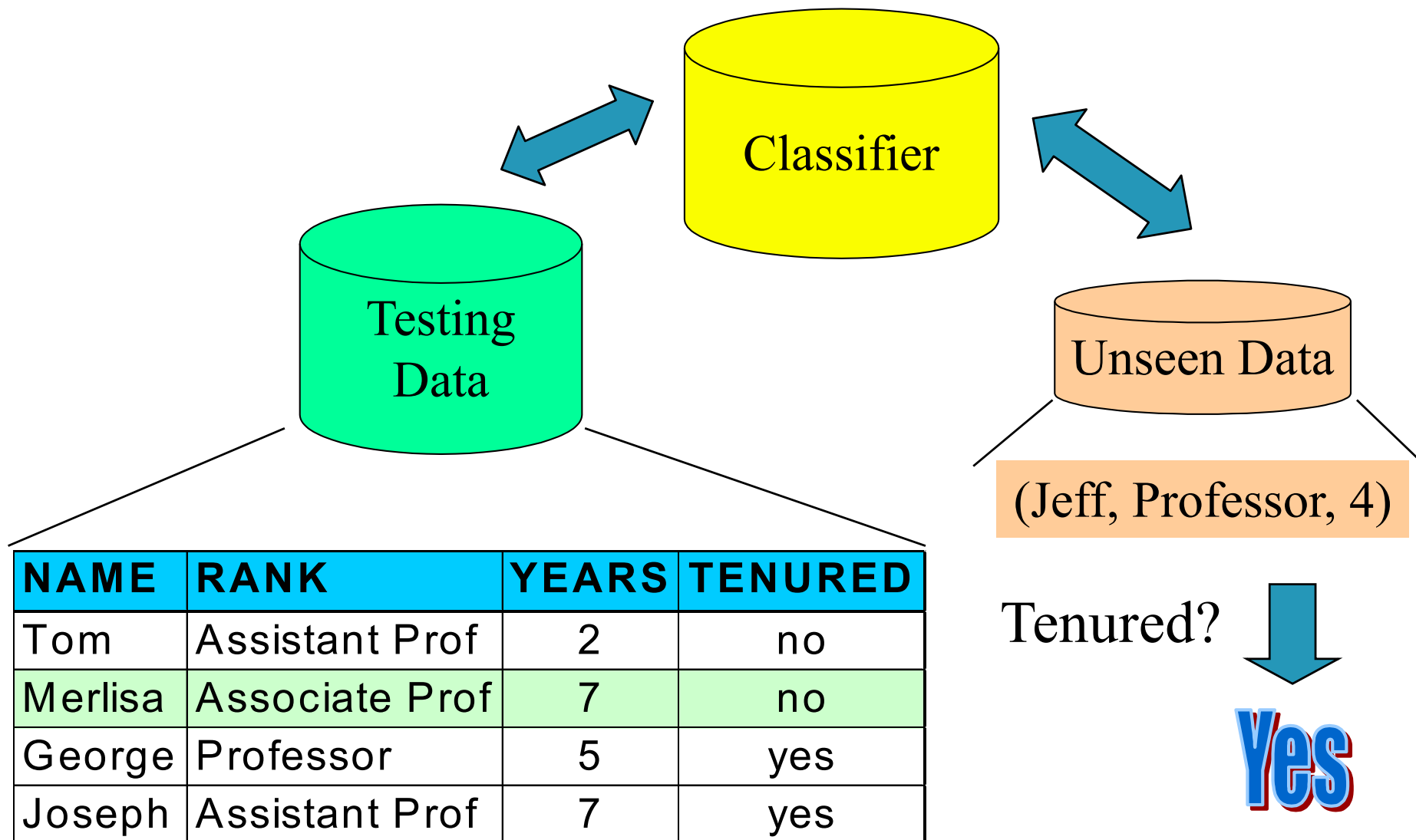
A Two-Step Process

- **Xây dựng mô hình/ Model construction:** mô tả tập các lớp đã xác định
 - Mỗi dữ liệu/ tuple/sample được xem đã thuộc về một lớp, được xác định bởi thuộc tính nhãn lớp/ **class label attribute**
 - Tập dữ liệu được dùng xây dựng mô hình là tập huấn luyện/ **training set**
 - Mô hình được biểu diễn dưới dạng luật phân loại, cây quyết định, hoặc công thức toán học.
- **Sử dụng mô hình/ Model usage:** để phân loại các đối tượng chưa biết
 - **Ước tính độ chính xác của mô hình**
 - Nhãn đã biết của mẫu thử được so sánh với kết quả được phân loại từ mô hình
 - **Tỉ lệ chính xác** là tỉ lệ phần trăm mẫu thử được phân loại chính xác theo mô hình
 - **Tập kiểm tra** độc lập với tập huấn luyện (otherwise overfitting)
 - Khi độ chính xác được chấp nhận, sử dụng mô hình để **classify new data**
- Lưu ý: Nếu *test set* được dùng để chọn mô hình, nó được gọi là **validation (test) set**


Quá trình (1): Xây dựng mô hình



Quá trình (2): Sử dụng mô hình trong dự đoán



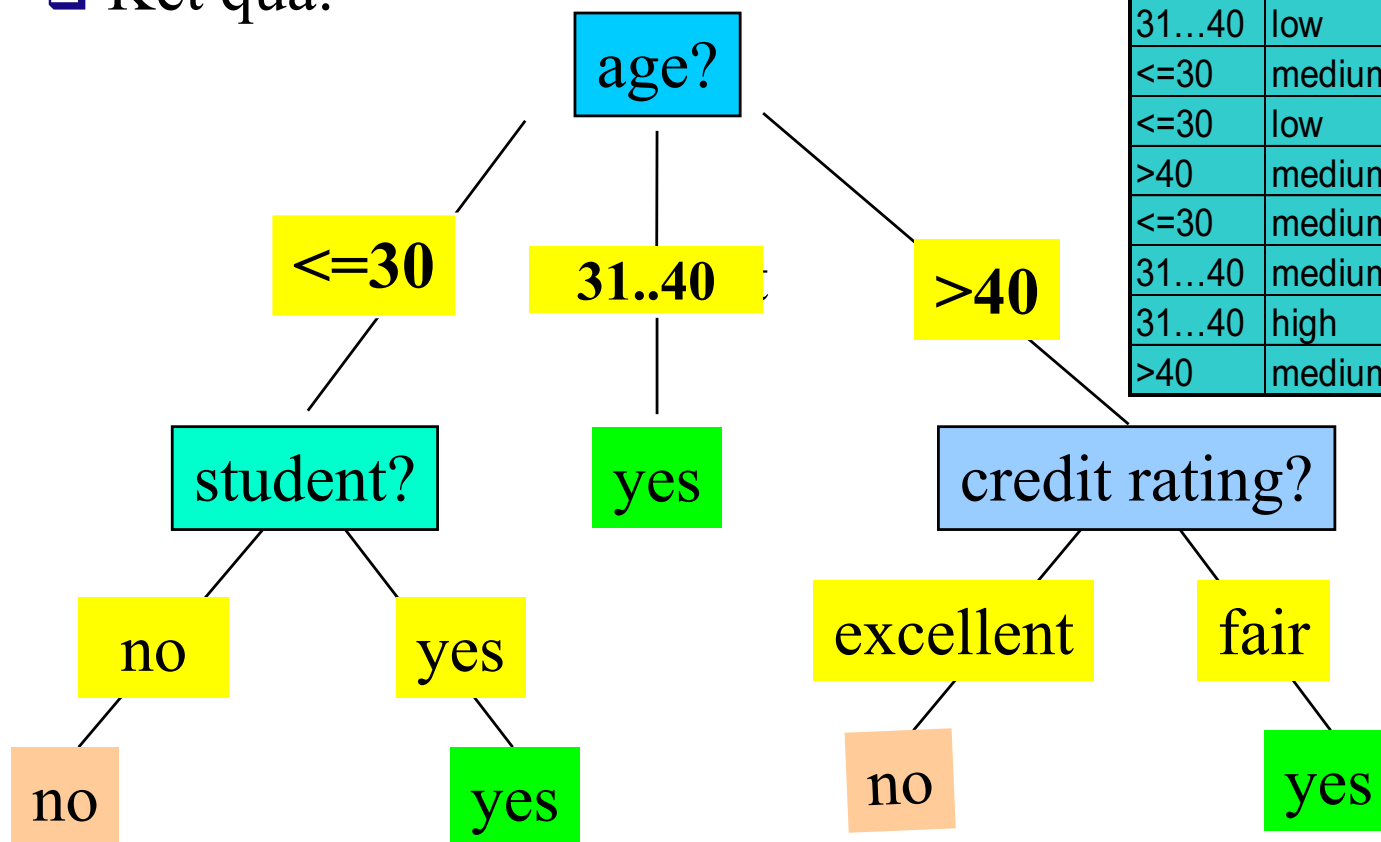
Chương 8. Phân loại: Các khái niệm cơ bản

- **Bài toán Classification:** Các khái niệm cơ bản
- **Cây quyết định/ Decision Tree Induction** 
- **Phân loại Bayes/ Bayes Classification Methods**
- **Phân loại dựa trên luật/ Rule-Based Classification**
- **Đánh giá và lựa chọn model/ Model Evaluation and Selection**
- **Những kỹ thuật tăng độ chính xác trong bài toán phân loại: Ensemble Methods**
- **Tóm tắt/ Summary**

Cây quyết định: ví dụ

- Tập huấn luyện: Mua máy tính
- The data set follows an example of Quinlan's ID3 (Playing Tennis)
- Kết quả:

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



Thuật toán cây quyết định

- Thuật toán cơ bản (a greedy algorithm)
 - Cây được xây dựng theo cách **chia để trị** đệ qui từ trên xuống
 - Khi bắt đầu, tất cả dữ liệu huấn luyện đều ở gốc/ root node
 - Các thuộc tính được phân loại (if continuous-valued, they are discretized in advance)
 - Các điểm dữ liệu được phân vùng đệ qui dựa trên các thuộc tính đã chọn
 - Các thuộc tính kiểm tra được chọn ngẫu nhiên hoặc trên cơ sở đo lường thống kê (e.g., **information gain**)
- Điều kiện dừng phân vùng/ partitioning
 - Tất cả điểm dữ liệu trên 1 node đều thuộc về cùng 1 lớp/ class
 - Không còn thuộc tính nào để phân vùng tiếp – **majority voting** is employed for classifying the leaf
 - Không còn điểm dữ liệu nào!

Entropy

- Entropy (Information Theory)

- A measure of uncertainty associated with a random variable

- Calculation: For a discrete random variable Y taking m distinct values $\{y_1, \dots, y_m\}$,

- $H(Y) = -\sum_{i=1}^m p_i \log(p_i)$, where $p_i = P(Y = y_i)$

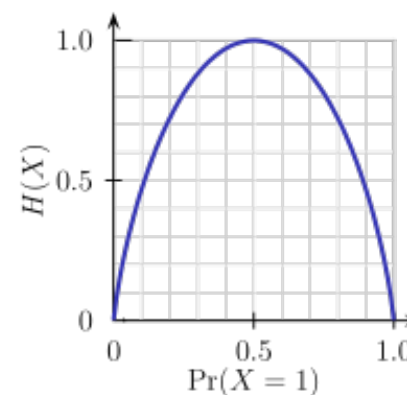
- Interpretation:

- Higher entropy => higher uncertainty

- Lower entropy => lower uncertainty

- Conditional Entropy

- $H(Y|X) = \sum_x p(x)H(Y|X = x)$



m = 2

Chọn thuộc tính: Độ lợi thông tin

Information Gain (ID3/C4.5)

- Lựa chọn đặc trưng với độ lợi thông tin cao nhất
- Đặt p_i là xác suất để một điểm dữ liệu trong D thuộc về lớp C_i , ước tính bằng $|C_{i,D}|/|D|$
- **Thông tin kỳ vọng/ Expected information** (entropy) cần thiết để phân lớp dữ liệu trong D :
$$Info(D) = -\sum_{i=1}^m p_i \log_2(p_i)$$
- **Thông tin cần thiết/ Information** needed (after using A to split D into v partitions) phân lớp D :
$$Info_A(D) = \sum_{j=1}^v \frac{|D_j|}{|D|} \times Info(D_j)$$
- **Độ lợi thông tin/ Information gained** bằng cách phân nhánh trên thuộc tính A
$$Gain(A) = Info(D) - Info_A(D)$$

Attribute Selection: Information Gain

■ Class P: buys_computer = “yes”

■ Class N: buys_computer = “no”

$$Info(D) = I(9,5) = -\frac{9}{14} \log_2\left(\frac{9}{14}\right) - \frac{5}{14} \log_2\left(\frac{5}{14}\right) = 0.940$$

age	p _i	n _i	I(p _i , n _i)
<=30	2	3	0.971
31...40	4	0	0
>40	3	2	0.971

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

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

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

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

Similarly,

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit_rating) = 0.048$$

Tính Information-Gain cho các thuộc tính có giá trị liên tục

- Thuộc tính A có các giá trị liên tục
- Phải xác định điểm phân chia tốt nhất/ *best split point* cho A
 - Sắp xếp giá trị của A tăng dần
 - Thông thường, điểm trung bình/ midpoint mỗi cặp giá trị liên kế được xem là *split point*
 - $(a_i + a_{i+1})/2$ là điểm trung bình giữa 2 giá trị a_i và a_{i+1}
 - Điểm *minimum expected information requirement* cho A được chọn làm điểm split-point cho A
- Phân chia/ Split:
 - D1 là tập dữ liệu con của D thỏa mãn $A \leq \text{split-point}$, và D2 là tập dữ liệu con của D thỏa mãn $A > \text{split-point}$

Gain Ratio cho lựa chọn thuộc tính (C4.5)

- Thước đo độ lợi thông tin thiên về các thuộc tính có số lượng giá trị lớn.
- C4.5 (tiếp theo ID3) sử dụng gain ratio để khắc phục vấn đề trên (normalization to information gain)

$$SplitInfo_A(D) = -\sum_{j=1}^v \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

- $GainRatio(A) = Gain(A)/SplitInfo(A)$

- Ví dụ.

$$SplitInfo_{income}(D) = -\frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) - \frac{4}{14} \times \log_2\left(\frac{4}{14}\right) = 1.557$$

- $gain_ratio(income) = 0.029/1.557 = 0.019$

- Thuộc tính với giá trị cực đại gain ratio sẽ được chọn làm điểm phân chia thuộc tính đó.

Chỉ số Gini

(CART, IBM IntelligentMiner)

- Nếu tập dữ liệu D chứa các mẫu từ n lớp, chỉ số gini, $gini(D)$ được định nghĩa

$$gini(D) = 1 - \sum_{j=1}^n p_j^2$$

trong đó p_j là tần số tương đối của lớp j trong D

- Nếu một tập dữ liệu D được chia trên A thành 2 tập con D_1 và D_2 , chỉ số $gini$

$$gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$$

- Giảm tạp chất:

$$\Delta gini(A) = gini(D) - gini_A(D)$$

- Thuộc tính có $gini_{split}(D)$ nhỏ nhất (or the largest reduction in impurity) được chọn để chia D (*need to enumerate all the possible splitting points for each attribute*)

Tính chỉ số Gini

- Ví dụ. D có 9 mẫu `buys_computer = “yes”` và 5 mẫu “no”

$$gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$$

- Giả sử thuộc tính thu nhập/incomes chia D thành 10 D_1 : {low, medium} và 4 D_2 {high}

$$\begin{aligned} gini_{income \in \{low, medium\}}(D) &= \left(\frac{10}{14}\right) Gini(D_1) + \left(\frac{4}{14}\right) Gini(D_2) \\ &= \frac{10}{14} \left(1 - \left(\frac{7}{10}\right)^2 - \left(\frac{3}{10}\right)^2\right) + \frac{4}{14} \left(1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2\right) \\ &= 0.443 \\ &= Gini_{income \in \{high\}}(D). \end{aligned}$$

$Gini_{\{low, high\}}$ là 0.458; $Gini_{\{medium, high\}}$ là 0.450. Do đó, chia theo {low, medium} (and {high}) vì chỉ số Gini là bé nhất

- Tất cả thuộc tính được mặc định có giá trị là liên tục
- Cần các công cụ khác, như gom cụm, để có các giá trị phân chia
- Có thể được thay đổi với các thuộc tính có giá trị phi số

So sánh các phương pháp lựa chọn thuộc tính

- Thông thường, có 3 phương pháp cho kết quả tốt, nhưng:
 - **Độ lợi thông tin/ Information gain:**
 - Thiên về các thuộc tính đa giá trị
 - **Gain ratio:**
 - Có xu hướng phân chia mất cân đối, trong đó một phân vùng nhỏ hơn nhiều so với các phân vùng khác
 - **Chỉ số Gini index:**
 - Thiên về các thuộc tính đa giá trị
 - Khó phân vùng khi số lượng lớp tăng lên
 - Có xu hướng các phân vùng có kích thước bằng nhau

Các phương pháp lựa chọn thuộc tính khác

- CHAID: a popular decision tree algorithm, measure based on χ^2 test for independence
- C-SEP: performs better than info. gain and gini index in certain cases
- G-statistic: has a close approximation to χ^2 distribution
- MDL (Minimal Description Length) principle (i.e., the simplest solution is preferred):
 - The best tree as the one that requires the fewest # of bits to both (1) encode the tree, and (2) encode the exceptions to the tree
- Multivariate splits (partition based on multiple variable combinations)
 - CART: finds multivariate splits based on a linear comb. of attrs.
- Which attribute selection measure is the best?
 - Most give good results, none is significantly superior than others

Overfitting và Tree Pruning

- “Quá khớp”/ Overfitting: Cây qui nạp có thể overfit tập dữ liệu huấn luyện
 - Quá nhiều nhánh, một số phản ánh sự bất lượng do nhiễu
 - Độ chính xác thấp đối với điểm dữ liệu mới
- 2 phương pháp tránh overfitting
 - Cắt tỉa trước/ Prepruning: *Halt tree construction early*—dừng tách node nếu việc này làm giảm chỉ số đo, dưới ngưỡng
 - Khó chọn ngưỡng
 - Cắt tỉa sau/ Postpruning: *Remove branches* từ cây hoàn chỉnh—thu được chuỗi các cây được cắt tỉa dần
 - Sử dụng tập dữ liệu khác với tập huấn luyện để thu được “best pruned tree”

Các cải tiến trên Basic Decision Tree Induction

- Cho phép giá trị đặc trưng là liên tục/ **continuous-valued**
 - Dynamically define new discrete-valued attributes that partition the continuous attribute value into a discrete set of intervals
- Xử lý **missing attribute values**
 - Assign the most common value of the attribute
 - Assign probability to each of the possible values
- **Xây dựng thuộc tính/ Attribute construction**
 - Create new attributes based on existing ones that are sparsely represented
 - This reduces fragmentation, repetition, and replication

Phân lớp đối với dữ liệu lớn

- Phân lớp—một vấn đề cổ điển được nghiên cứu rộng rãi bởi các nhà thống kê và các nhà nghiên cứu học máy.
- Khả năng mở rộng: Phân loại tập dữ liệu với hàng triệu mẫu, với hàng trăm thuộc tính, với tốc độ hợp lý
- Tại sao decision tree induction phổ biến?
 - Tốc độ học nhanh (than other classification methods)
 - Chuyển đổi đơn giản và dễ hiểu thành các qui tắc phân loại
 - Có thể sử dụng truy vấn SQL để truy cập CSDL
 - Độ chính xác có thể so sánh được với các pp phân loại khác
- **RainForest** (VLDB'98 — Gehrke, Ramakrishnan & Ganti)
 - Builds an AVC-list (attribute, value, class label)

Khả năng mở rộng với RainForest

- Separates the scalability aspects from the criteria that determine the quality of the tree
- Builds an AVC-list: **AVC (Attribute, Value, Class_label)**
- **AVC-set** (of an attribute X)
 - Projection of training dataset onto the attribute X and class label where counts of individual class label are aggregated
- **AVC-group** (of a node n)
 - Set of AVC-sets of all predictor attributes at the node n

Rainforest: Training Set and Its AVC Sets

Training Examples

age	income	student	credit_rating	com
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no

AVC-set on *Age*

Age	Buy_Computer	
	yes	no
<=30	2	3
31..40	4	0
>40	3	2

AVC-set on *income*

income	Buy_Computer	
	yes	no
high	2	2
medium	4	2
low	3	1

AVC-set on *Student*

student	Buy_Computer	
	yes	no
yes	6	1
no	3	4

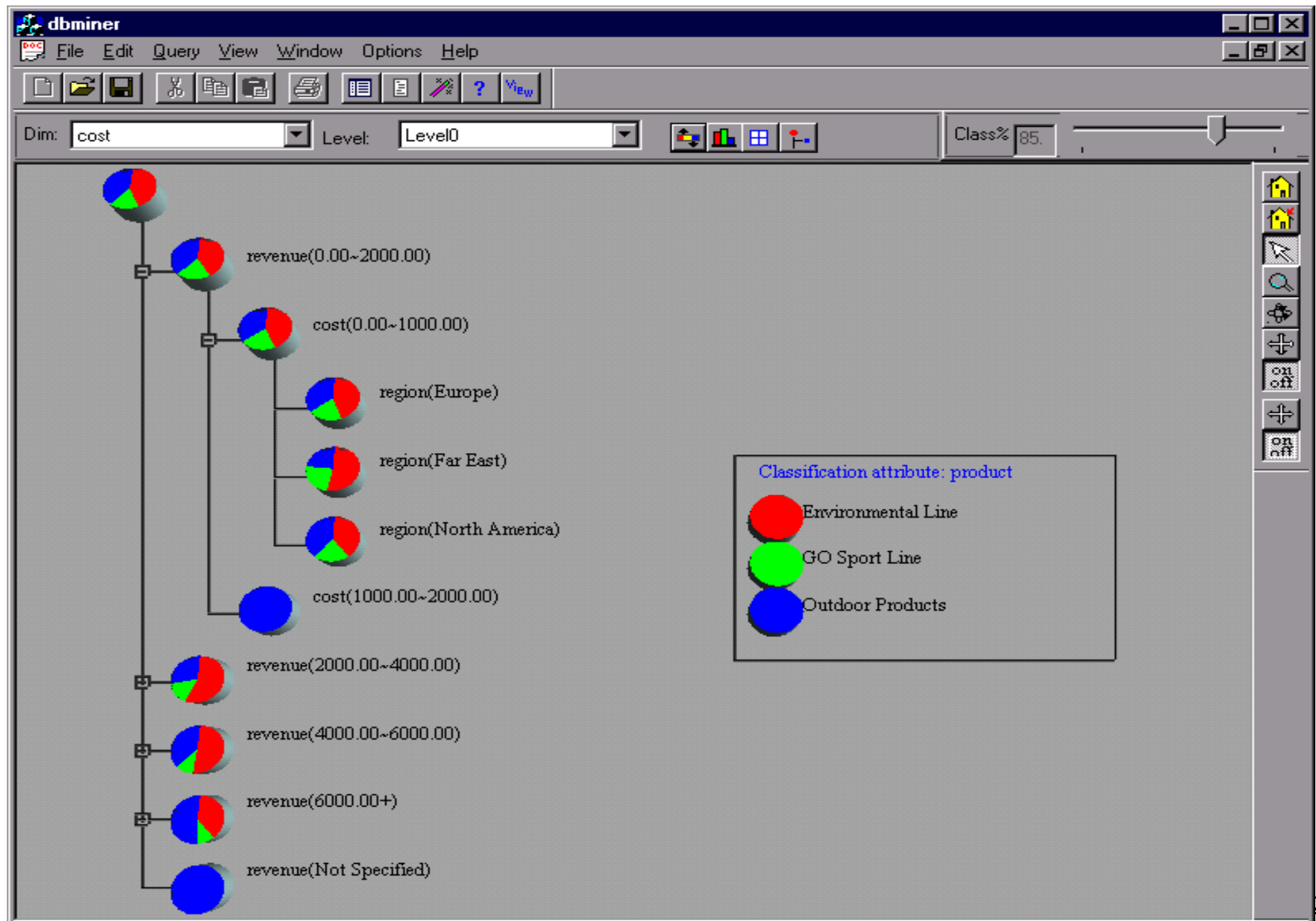
AVC-set on *credit_rating*

Credit rating	Buy_Computer	
	yes	no
fair	6	2
excellent	3	3

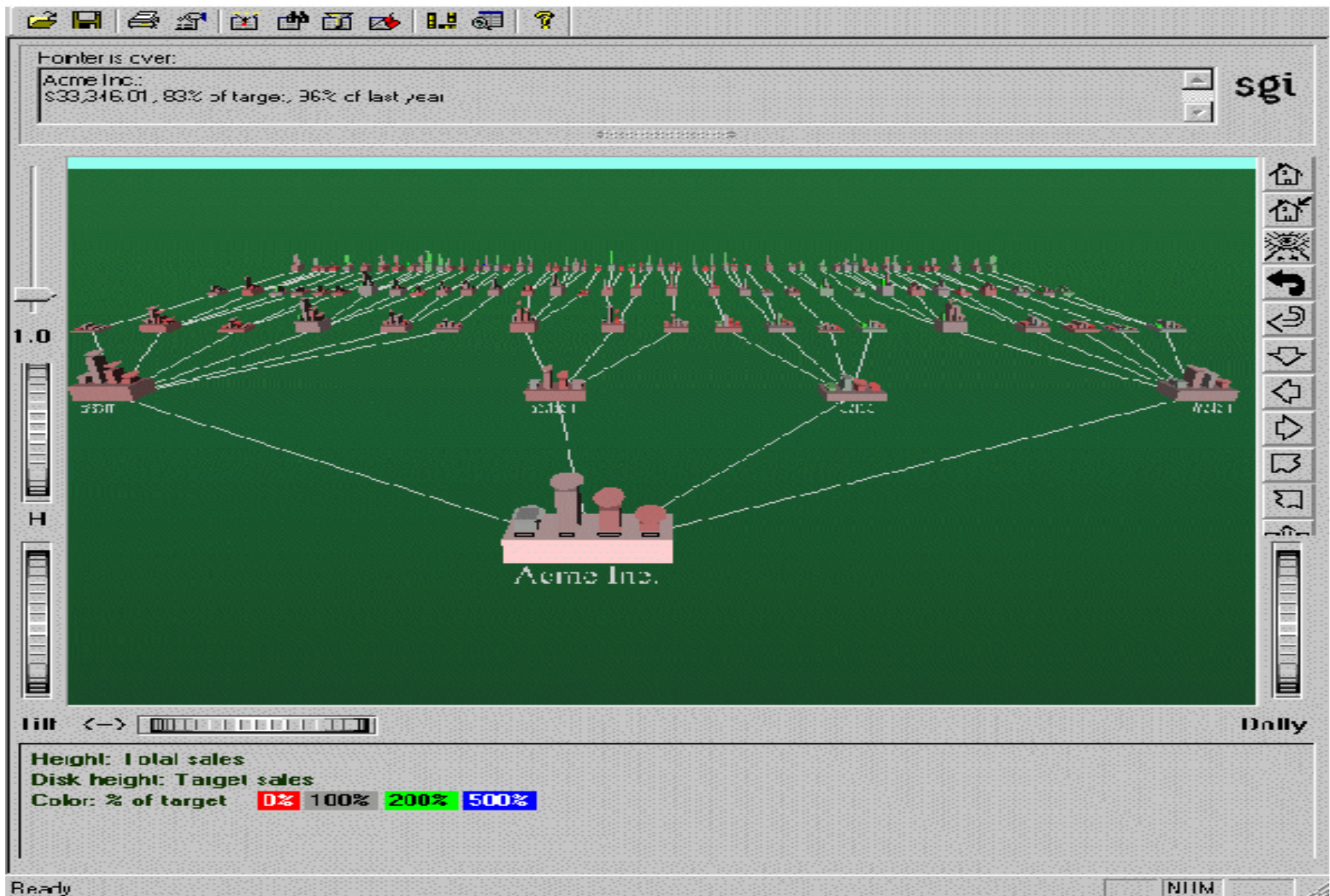
BOAT (Bootstrapped Optimistic Algorithm for Tree Construction)

- Use a statistical technique called *bootstrapping* to create several smaller samples (subsets), each fits in memory
- Each subset is used to create a tree, resulting in several trees
- These trees are examined and used to construct a new tree T'
 - It turns out that T' is very close to the tree that would be generated using the whole data set together
- Adv: requires only two scans of DB, an incremental alg.

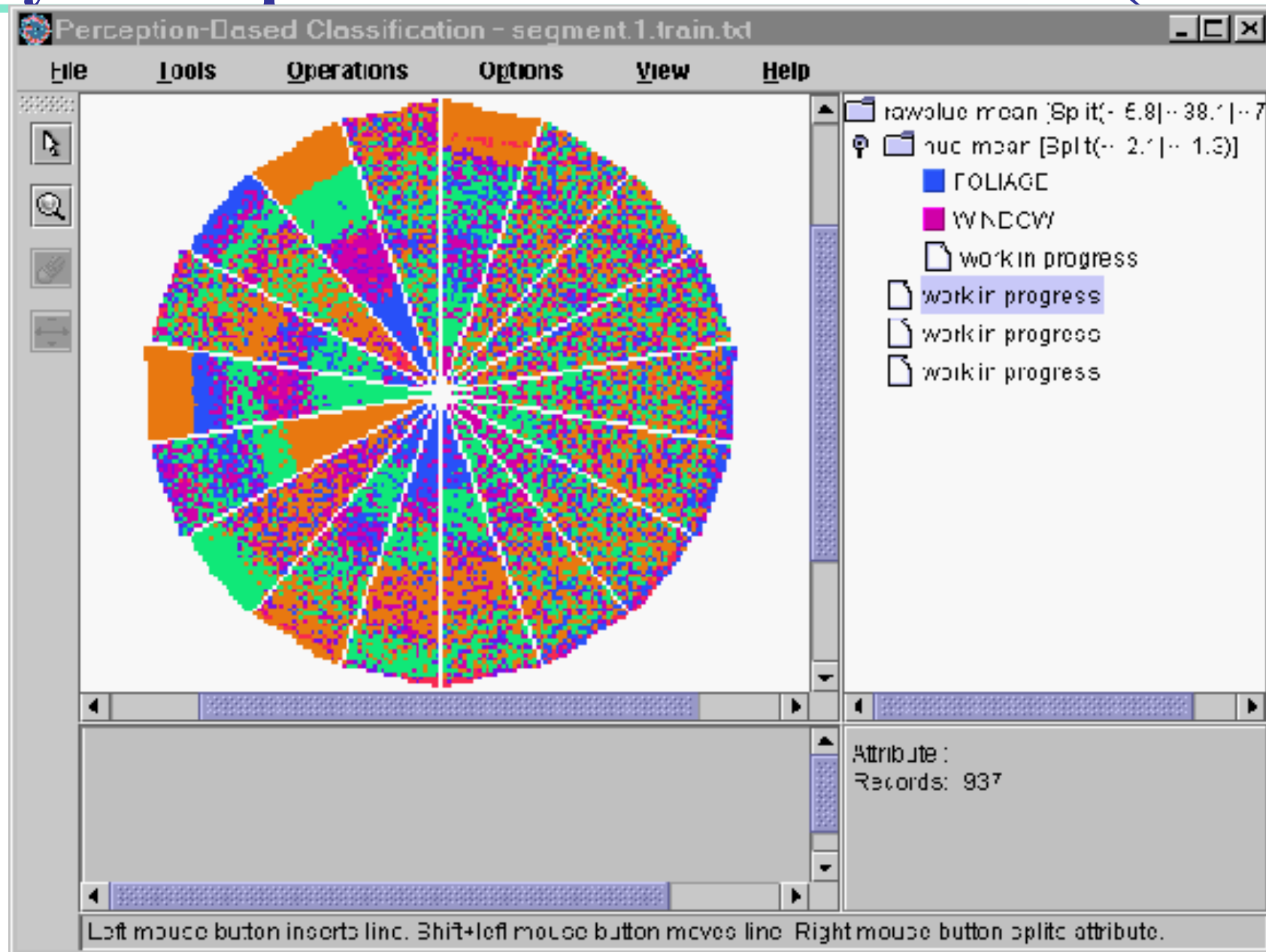
Presentation of Classification Results




Visualization of a Decision Tree in SGI/MineSet 3.0



Interactive Visual Mining by Perception-Based Classification (PBC)



Chương 8. Phân loại: Các khái niệm cơ bản

- **Bài toán Classification:** Các khái niệm cơ bản
- **Cây quyết định/ Decision Tree Induction**
- **Phân loại Bayes/ Bayes Classification Methods** 
- **Phân loại dựa trên luật/ Rule-Based Classification**
- **Đánh giá và lựa chọn model/ Model Evaluation and Selection**
- **Những kỹ thuật tăng độ chính xác trong bài toán phân loại: Ensemble Methods**
- **Tóm tắt/ Summary**

Phân loại Bayesian: Tại sao?

- Phân loại thống kê: thực hiện dự đoán xác suất, *như*, dự đoán xác suất thành viên lớp
- Foundation: dựa trên định lý Bayes'
- Performance: Phân loại Bayes đơn giản, *naïve Bayesian classifier*, có hiệu suất tương đương cây quyết định và mạng thần kinh
- Incremental: Mỗi dữ liệu huấn luyện làm tăng/giảm xác suất một giả thuyết là đúng— kiến thức trước đó có thể kết hợp với dữ liệu quan sát được
- Standard: Ngay cả khi phương pháp Bayes khó tính toán, chúng vẫn có thể cung cấp một tiêu chuẩn cho việc ra quyết định tối ưu mà các phương pháp khác có thể đo lường được.

Định lý Bayes': Cơ bản

- Định lý xác suất tổng:
$$P(B) = \sum_{i=1}^M P(B|A_i)P(A_i)$$
- Định lý Bayes':
$$P(H|\mathbf{X}) = \frac{P(\mathbf{X}|H)P(H)}{P(\mathbf{X})} = P(\mathbf{X}|H) \times P(H) / P(\mathbf{X})$$
 - Với \mathbf{X} là điểm dữ liệu (“evidence”): nhãn lớp không xác định
 - H là giả thuyết/ *hypothesis* rằng \mathbf{X} thuộc lớp C
 - Phân lớp là để xác định $P(H|\mathbf{X})$, (i.e., *posteriori probability*): xác suất hậu nghiệm, là xác suất mà giả thuyết đúng với điều kiện \mathbf{X}
 - $P(H)$ (*prior probability*): xác suất tiên nghiệm
 - E.g., \mathbf{X} will buy computer, regardless of age, income, ...
 - $P(\mathbf{X})$: xác suất dữ liệu mẫu được quan sát
 - $P(\mathbf{X}|H)$ (likelihood): xác suất quan sát mẫu \mathbf{X} , với điều kiện giả thuyết đúng
 - E.g., Given that \mathbf{X} will buy computer, the prob. that \mathbf{X} is 31..40, medium income

Dự đoán dựa trên định lý Bayes

- Cho tập huấn luyện \mathbf{X} , *posteriori probability of a hypothesis* H , $P(H|\mathbf{X})$, định lý Bayes

$$P(H | \mathbf{X}) = \frac{P(\mathbf{X} | H)P(H)}{P(\mathbf{X})} = P(\mathbf{X} | H) \times P(H) / P(\mathbf{X})$$

- Informally, this can be viewed as

posteriori = likelihood x prior/evidence

- Dự đoán \mathbf{X} thuộc lớp C_i nếu xác suất $P(C_i|\mathbf{X})$ là cực đại trong số $P(C_k|\mathbf{X})$ đối với k lớp
- Khó khăn thực tế: đòi hỏi kiến thức ban đầu về nhiều xác suất, liên quan đến chi phí tính toán

Phân loại: cực đại xác suất hậu nghiệm

- Tập huấn luyện D có nhãn, biểu diễn điểm dữ liệu bằng vector

$$\mathbf{X} = (x_1, x_2, \dots, x_n)$$

- Giả sử có m lớp C_1, C_2, \dots, C_m .
- Phân lớp để đạt cực đại xác suất hậu nghiệm, i.e., the maximal $P(C_i|\mathbf{X})$
- Được thực hiện qua định lý Bayes

$$P(C_i|\mathbf{X}) = \frac{P(\mathbf{X}|C_i)P(C_i)}{P(\mathbf{X})}$$

- Vì $P(\mathbf{X})$ như nhau với mọi lớp,

Cần tối đa: $P(C_i|\mathbf{X}) = P(\mathbf{X}|C_i)P(C_i)$

Phân loại Naïve Bayes

- Giả sử: các thuộc tính đều độc lập (i.e., no dependence relation between attributes):

$$P(\mathbf{X} | C_i) = \prod_{k=1}^n P(x_k | C_i) = P(x_1 | C_i) \times P(x_2 | C_i) \times \dots \times P(x_n | C_i)$$

- Điều này giảm được chi phí tính toán: Chỉ tính phân phối lớp
- Nếu A_k là categorical, $P(x_k | C_i)$ là số điểm dữ liệu thuộc lớp C_i mang giá trị x_k trong A_k chia cho $|C_i, D|$ (# of tuples of C_i in D)
- If A_k is continuous-valued, $P(x_k | C_i)$ is usually computed based on Gaussian distribution with a mean μ and standard deviation σ

and $P(x_k | C_i)$ is

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$$P(\mathbf{X} | C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i})$$

Phân loại Naïve Bayes: Tập huấn luyện

Class:

C1:buys_computer =
'yes'

C2:buys_computer = 'no'

Data to be classified:

X = (age <=30,

Income = medium,

Student = yes

Credit_rating = Fair)

age	income	student	credit_rating	comp
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no

Phân loại Naïve Bayes: Ví dụ

age	income	student	credit_rating	comp
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no

- $P(C_i): P(\text{buys_computer} = \text{"yes"}) = 9/14 = 0.643$

$$P(\text{buys_computer} = \text{"no"}) = 5/14 = 0.357$$

- Compute $P(X|C_i)$ for each class

$$P(\text{age} = \text{"<=30"} | \text{buys_computer} = \text{"yes"}) = 2/9 = 0.222$$

$$P(\text{age} = \text{"<= 30"} | \text{buys_computer} = \text{"no"}) = 3/5 = 0.6$$

$$P(\text{income} = \text{"medium"} | \text{buys_computer} = \text{"yes"}) = 4/9 = 0.444$$

$$P(\text{income} = \text{"medium"} | \text{buys_computer} = \text{"no"}) = 2/5 = 0.4$$

$$P(\text{student} = \text{"yes"} | \text{buys_computer} = \text{"yes"}) = 6/9 = 0.667$$

$$P(\text{student} = \text{"yes"} | \text{buys_computer} = \text{"no"}) = 1/5 = 0.2$$

$$P(\text{credit_rating} = \text{"fair"} | \text{buys_computer} = \text{"yes"}) = 6/9 = 0.667$$

$$P(\text{credit_rating} = \text{"fair"} | \text{buys_computer} = \text{"no"}) = 2/5 = 0.4$$

- **$X = (\text{age} \leq 30, \text{income} = \text{medium}, \text{student} = \text{yes}, \text{credit_rating} = \text{fair})$**

$$P(X|C_i) : P(X|\text{buys_computer} = \text{"yes"}) = 0.222 \times 0.444 \times 0.667 \times 0.667 = 0.044$$

$$P(X|\text{buys_computer} = \text{"no"}) = 0.6 \times 0.4 \times 0.2 \times 0.4 = 0.019$$

$$P(X|C_i) * P(C_i) : P(X|\text{buys_computer} = \text{"yes"}) * P(\text{buys_computer} = \text{"yes"}) = 0.028$$

$$P(X|\text{buys_computer} = \text{"no"}) * P(\text{buys_computer} = \text{"no"}) = 0.007$$

Therefore, X belongs to class ("buys_computer = yes")

Tránh xác suất bằng 0

- Naïve Bayesian prediction requires each conditional prob. be **non-zero**. Otherwise, the predicted prob. will be zero

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

- Ex. Suppose a dataset with 1000 tuples, income=low (0), income= medium (990), and income = high (10)
- Use **Laplacian correction** (or Laplacian estimator)

- *Adding 1 to each case*

$$\text{Prob}(\text{income} = \text{low}) = 1/1003$$

$$\text{Prob}(\text{income} = \text{medium}) = 991/1003$$


$$\text{Prob}(\text{income} = \text{high}) = 11/1003$$

- The “corrected” prob. estimates are close to their “uncorrected” counterparts

Phân loại Naïve Bayes: Bình luận

- Ưu điểm
 - Easy to implement
 - Good results obtained in most of the cases
- Hạn chế
 - Assumption: class conditional independence, therefore loss of accuracy
 - Practically, dependencies exist among variables
 - E.g., hospitals: patients: Profile: age, family history, etc.
Symptoms: fever, cough etc., Disease: lung cancer, diabetes, etc.
 - Dependencies among these cannot be modeled by Naïve Bayes Classifier
- How to deal with these dependencies? Bayesian Belief Networks (Chapter 9)

Chương 8. Phân loại: Các khái niệm cơ bản

- Bài toán Classification: Các khái niệm cơ bản
- Cây quyết định/ Decision Tree Induction
- Phân loại Bayes/ Bayes Classification Methods
- Phân loại dựa trên luật/ **Rule-Based Classification** 
- Đánh giá và lựa chọn model/ Model Evaluation and Selection
- Những kỹ thuật tăng độ chính xác trong bài toán phân loại: Ensemble Methods
- Tóm tắt/ Summary

Sử dụng luật Nếu-Thì/ IF-THEN với bài toán phân lớp

- Represent the knowledge in the form of **IF-THEN** rules

R: IF *age* = youth AND *student* = yes THEN *buys_computer* = yes

- Rule antecedent/precondition vs. rule consequent

- Assessment of a rule: *coverage* and *accuracy*

- $n_{\text{covers}} = \#$ of tuples covered by R

- $n_{\text{correct}} = \#$ of tuples correctly classified by R

$\text{coverage}(R) = n_{\text{covers}} / |D|$ /* D: training data set */

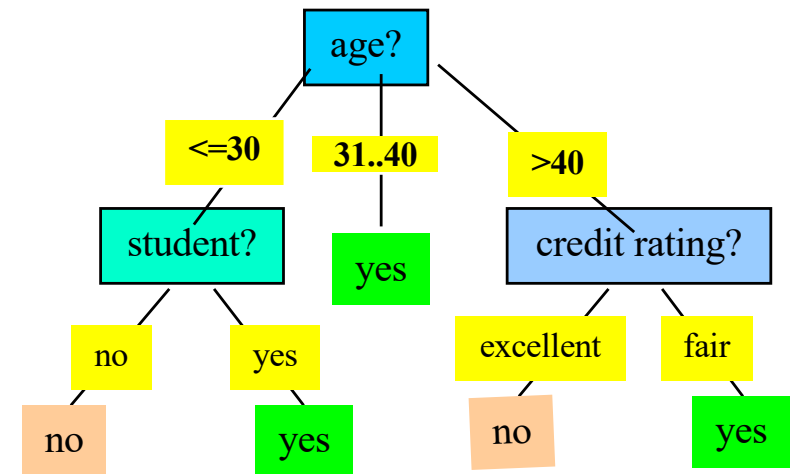
$\text{accuracy}(R) = n_{\text{correct}} / n_{\text{covers}}$

- If more than one rule are triggered, need **conflict resolution**

- Size ordering: assign the highest priority to the triggering rules that has the “toughest” requirement (i.e., with the *most attribute tests*)
- Class-based ordering: decreasing order of *prevalence or misclassification cost per class*
- Rule-based ordering (**decision list**): rules are organized into one long priority list, according to some measure of rule quality or by experts

Trích xuất qui tắc từ cây quyết định

- Rules are *easier to understand* than large trees
- One rule is created *for each path* from the root to a leaf
- Each attribute-value pair along a path forms a conjunction: the leaf holds the class prediction
- Rules are mutually exclusive and exhaustive
- Example: Rule extraction from our *buys_computer* decision-tree



IF *age* = young AND *student* = no

THEN *buys_computer* = no

IF *age* = young AND *student* = yes

THEN *buys_computer* = yes

IF *age* = mid-age

THEN *buys_computer* = yes

IF *age* = old AND *credit_rating* = excellent THEN *buys_computer* = no

IF *age* = old AND *credit_rating* = fair THEN *buys_computer* = yes

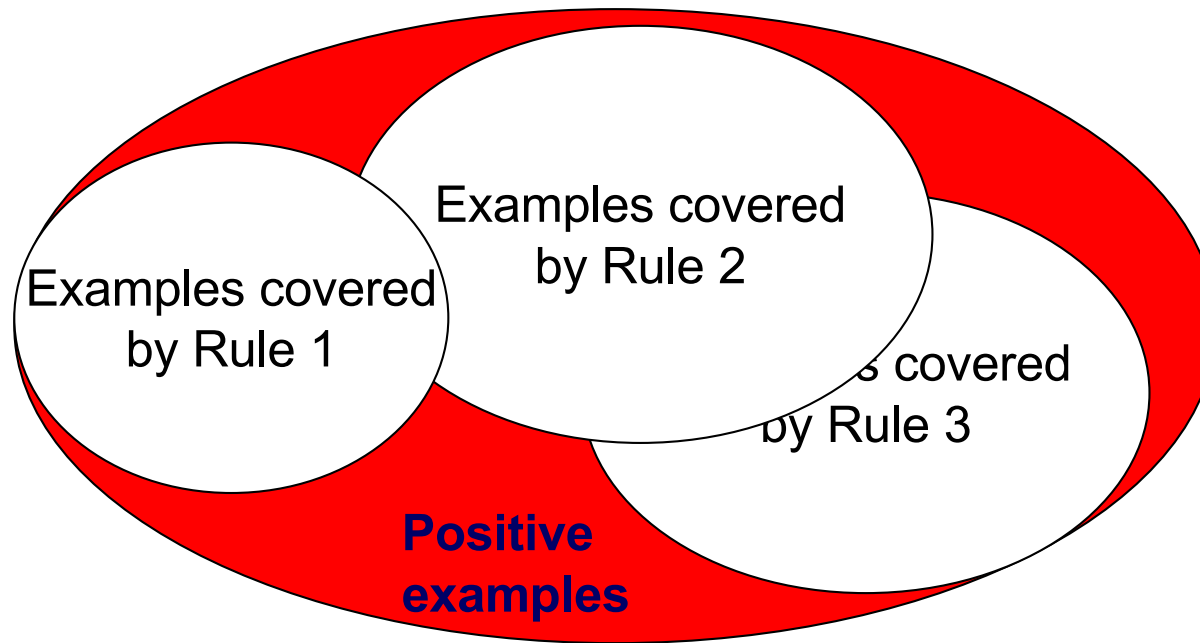
Qui nạp: Phương pháp che phủ tuần tự

- Sequential covering algorithm: Extracts rules directly from training data
- Typical sequential covering algorithms: FOIL, AQ, CN2, RIPPER
- Rules are learned *sequentially*, each for a given class C_i will cover many tuples of C_i but none (or few) of the tuples of other classes
- Steps:
 - Rules are learned one at a time
 - Each time a rule is learned, the tuples covered by the rules are removed
 - Repeat the process on the remaining tuples until *termination condition*, e.g., when no more training examples or when the quality of a rule returned is below a user-specified threshold
- Comp. w. decision-tree induction: learning a set of rules *simultaneously*

Thuật toán che phủ tuần tự

Sequential Covering Algorithm

while (enough target tuples left)
 generate a rule
 remove positive target tuples satisfying this rule



Rule Generation

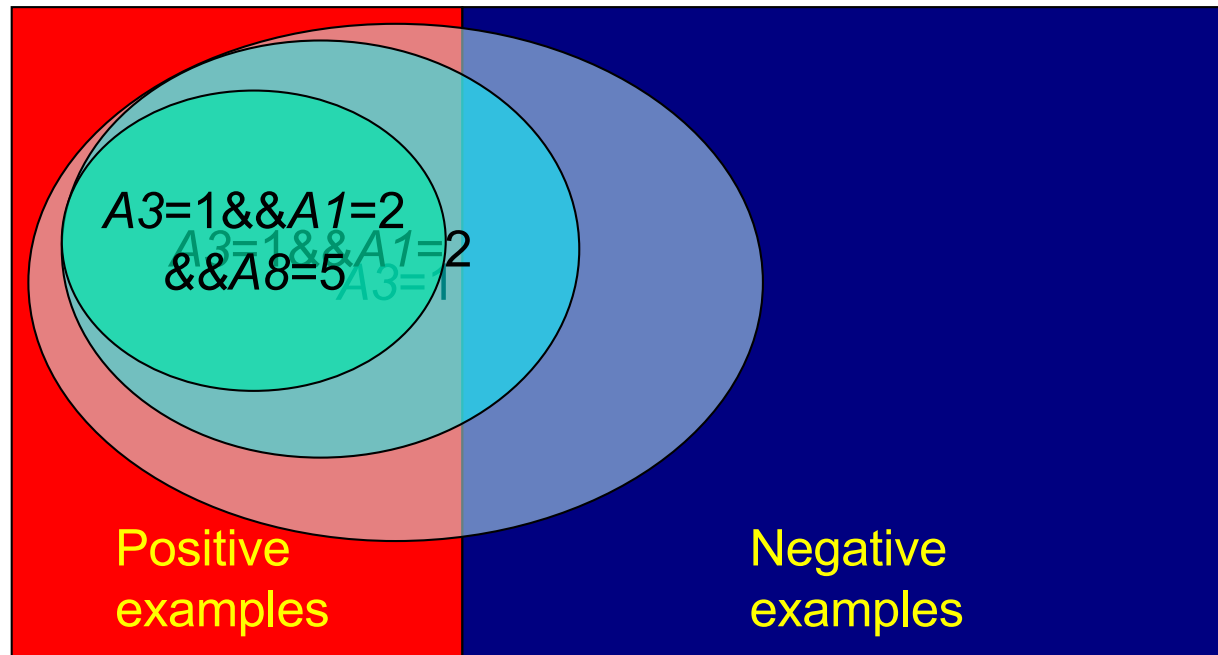
- To generate a rule

while(true)

find the best predicate p

if $\text{foil-gain}(p) > \text{threshold}$ **then** add p to current rule

else break



Học 1 luật như thế nào/ Learn-One-Rule?

- Start with the *most general rule* possible: condition = empty
- *Adding new attributes* by adopting a greedy depth-first strategy
 - Picks the one that most improves the rule quality

- Rule-Quality measures: consider both coverage and accuracy

- Foil-gain (in FOIL & RIPPER): assesses info_gain by extending condition

$$FOIL_Gain = pos' \times (\log_2 \frac{pos'}{pos' + neg'} - \log_2 \frac{pos}{pos + neg})$$


- favors rules that have high accuracy and cover many positive tuples
 - Rule pruning based on an independent set of test tuples

$$FOIL_Prune(R) = \frac{pos - neg}{pos + neg}$$

Pos/neg are # of positive/negative tuples covered by R.

If *FOIL_Prune* is higher for the pruned version of R, prune R

Chương 8. Phân loại: Các khái niệm cơ bản

- Bài toán Classification: Các khái niệm cơ bản
- Cây quyết định/ Decision Tree Induction
- Phân loại Bayes/ Bayes Classification Methods
- Phân loại dựa trên luật/ Rule-Based Classification
- Đánh giá và lựa chọn model/ **Model Evaluation and Selection** 
- Những kỹ thuật tăng độ chính xác trong bài toán phân loại: Ensemble Methods
- Tóm tắt/ Summary

Đánh giá và lựa chọn mô hình

- Các chỉ số đánh giá/ **Evaluation metrics**: Độ chính xác/ Accuracy? Và các chỉ số khác?
- Sử dụng **validation test set** thay vì tập huấn luyện/ training set khi đánh giá độ chính xác.
- Các phương pháp ước tính độ chính xác:
 - Holdout method, random subsampling
 - Cross-validation
 - Bootstrap
- So sánh các thuật toán phân lớp:
 - Khoảng tin cậy/ Confidence intervals
 - Phân tích/ Cost-benefit, và đường cong ROC

Các chỉ số đánh giá phân loại: Confusion Matrix

Confusion Matrix:

Actual class \ Predicted class	C_1	$\neg C_1$
C_1	True Positives (TP)	False Negatives (FN)
$\neg C_1$	False Positives (FP)	True Negatives (TN)

Example of Confusion Matrix:

Actual class \ Predicted class	buy_computer = yes	buy_computer = no	Total
buy_computer = yes	6954	46	7000
buy_computer = no	412	2588	3000
Total	7366	2634	10000

- Cho m lớp, $CM_{i,j}$ là **confusion matrix** với số điểm dữ liệu i được dán nhãn bởi lớp j
- Tổng số được tổng hợp ở hàng/cột mở rộng.

Chỉ số đánh giá: Accuracy, Error Rate, Sensitivity và Specificity

A\P	C	¬C	
C	TP	FN	P
¬C	FP	TN	N
	P'	N'	All

- **Độ chính xác/ Accuracy**, or recognition rate: tỉ lệ bộ dữ liệu kiểm tra/ test, được phân loại chính xác

$$\text{Accuracy} = (TP + TN)/All$$

- **Tỉ lệ lỗi: $1 - accuracy$** , or **Error rate** = $(FP + FN)/All$

- **Vấn đề mất cân bằng lớp:**

- Một lớp có thể có rất ít điểm dữ liệu, ví dụ: fraud, or HIV-positive
- Significant *majority of the negative class* and minority of the positive class
- **Sensitivity:** Tỉ lệ đúng thực sự
 - **Sensitivity** = TP/P
- **Specificity:** Tỉ lệ sai thực sự
 - **Specificity** = TN/N

Các chỉ số đánh giá:

Precision, Recall, và F-measures

- **Precision:** exactness – phần trăm bộ dữ liệu được thuật toán phân loại là dương/ positive thực sự là dương.

$$precision = \frac{TP}{TP + FP}$$

- **Recall:** sự đầy đủ/ Completeness – phần trăm bộ dữ liệu dương/ positive, được thuật toán phân loại là dương

$$recall = \frac{TP}{TP + FN}$$

- Điểm tuyệt đối là 1.0
- Mỗi quan hệ nghịch đảo giữa precision & recall
- **F measure (F_1 or F-score):** trung bình điều hòa của precision và recall,

$$F = \frac{2 \times precision \times recall}{precision + recall}$$

- **F_β :** thước đo trọng số của precision và recall
 - assigns β times as much weight to recall as to precision

$$F_\beta = \frac{(1 + \beta^2) \times precision \times recall}{\beta^2 \times precision + recall}$$

Các chỉ số đánh giá: Ví dụ

Actual Class\Predicted class	cancer = yes	cancer = no	Total	Recognition(%)
cancer = yes	90	210	300	30.00 (<i>sensitivity</i>)
cancer = no	140	9560	9700	98.56 (<i>specificity</i>)
Total	230	9770	10000	96.40 (<i>accuracy</i>)

■ $Precision = 90/230 = 39.13\%$

$$Recall = 90/300 = 30.00\%$$

Đánh giá độ chính xác:

Phương pháp Holdout & Cross-Validation

■ Holdout method

- Dữ liệu đã cho được phân chia ngẫu nhiên thành 2 phần độc lập
 - Tập huấn luyện (e.g., 2/3) để xây dựng mô hình
 - Tập kiểm tra (e.g., 1/3) để ước tính độ chính xác
- Lấy mẫu ngẫu nhiên: một biến thể của pp holdout
 - Lặp lại holdout k lần, độ chính xác = trung bình của các lần holdout.

■ Cross-validation (k -fold, với $k = 10$ là phổ biến)

- Phân vùng ngẫu nhiên dữ liệu thành k *mutually exclusive* tập con, với kích thước xấp xỉ nhau.
- Tại vòng lặp thứ i -th, sử dụng D_i làm tập kiểm tra, và phần còn lại làm tập huấn luyện.
- Leave-one-out: k folds where $k = \#$ of tuples, for small sized data
- *Stratified cross-validation*: folds are stratified so that class dist. in each fold is approx. the same as that in the initial data

Đánh giá độ chính xác: Bootstrap

- **Bootstrap**

- Làm việc tốt với tập dữ liệu nhỏ
- Lấy mẫu tập huấn luyện, có thể lặp/ *replacement*
 - i.e., each time a tuple is selected, it is equally likely to be selected again and re-added to the training set

- Một vài pp Bootstrap, và phổ biến là **.632 bootstrap**

- A data set with d tuples is sampled d times, with replacement, resulting in a training set of d samples. The data tuples that did not make it into the training set end up forming the test set. About 63.2% of the original data end up in the bootstrap, and the remaining 36.8% form the test set (since $(1 - 1/d)^d \approx e^{-1} = 0.368$)
- Repeat the sampling procedure k times, overall accuracy of the model:

$$Acc(M) = \frac{1}{k} \sum_{i=1}^k (0.632 \times Acc(M_i)_{test_set} + 0.368 \times Acc(M_i)_{train_set})$$

Ước tính khoảng tin cậy: Classifier Models M_1 vs. M_2

- Giả sử có 2 thuật toán phân lớp, M_1 và M_2 , cái nào tốt hơn?
- Sử dụng 10-fold cross-validation: $\overline{err}(M_1)$ và $\overline{err}(M_2)$
- These mean error rates are just *estimates* of error on the true population of *future* data cases
- What if the difference between the 2 error rates is just attributed to *chance*?
 - Use a **test of statistical significance**
 - Obtain **confidence limits** for our error estimates

Ước tính khoảng tin cậy: Null Hypothesis

- Thực hiện 10-fold cross-validation
- Giả sử mẫu tuân theo phân bố/ **t distribution** với $k-1$ **degrees of freedom** (here, $k=10$)
- Sử dụng **t-test** (or **Student's t-test**)
- **Null Hypothesis:** M_1 & M_2 là như nhau
- Nếu có thể loại bỏ/ **reject** null hypothesis, thì
 - Kết luận sự khác nhau giữa M_1 và M_2 mang tính chất thống kê/ **statistically significant**
 - Chọn mô hình với tỉ lệ lỗi thấp

Ước tính khoảng tin cậy: t-test

- Nếu chỉ có 1 tập kiểm tra: **pairwise comparison**

- For i^{th} round of 10-fold cross-validation, the same cross partitioning is used to obtain $err(M_1)_i$ and $err(M_2)_i$
- Average over 10 rounds to get $\overline{err}(M_1)$ and $\overline{err}(M_2)$
- **t-test** computes **t-statistic** with $k-1$ **degrees of freedom**:

$$t = \frac{\overline{err}(M_1) - \overline{err}(M_2)}{\sqrt{var(M_1 - M_2)/k}} \quad \text{where}$$

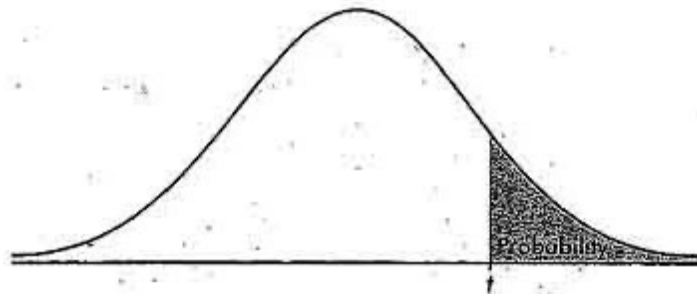
$$var(M_1 - M_2) = \frac{1}{k} \sum_{i=1}^k \left[err(M_1)_i - err(M_2)_i - (\overline{err}(M_1) - \overline{err}(M_2)) \right]^2$$

- Nếu có 2 tập kiểm tra: sử dụng **non-paired t-test**

$$\text{where } var(M_1 - M_2) = \sqrt{\frac{var(M_1)}{k_1} + \frac{var(M_2)}{k_2}},$$

where k_1 & k_2 are # of cross-validation samples used for M_1 & M_2 , resp.

Ước tính khoảng tin cậy: Table for t-distribution



- Symmetric
- Significance level, e.g., $sig = 0.05$ or 5% means M_1 & M_2 are *significantly different* for 95% of population
- Confidence limit, $z = sig/2$

TABLE B: t-DISTRIBUTION CRITICAL VALUES

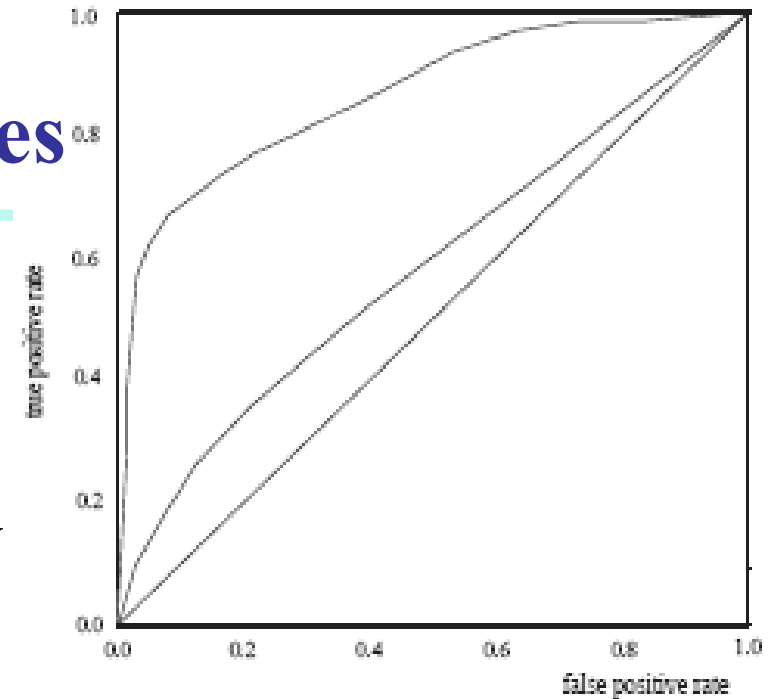
df	Tail probability p											
	.25	.20	.15	.10	.05	.025	.02	.01	.005	.0025	.001	.0005
1	1.000	1.376	1.963	3.078	6.314	12.71	15.89	31.82	63.66	127.3	318.3	636.6
2	.816	1.061	1.386	1.886	2.920	4.303	4.849	6.965	9.925	14.09	22.33	31.60
3	.765	.978	1.250	1.638	2.353	3.182	3.482	4.541	5.841	7.453	10.21	12.92
4	.741	.941	1.190	1.533	2.132	2.776	2.999	3.747	4.604	5.598	7.173	8.610
5	.727	.920	1.156	1.476	2.015	2.571	2.757	3.365	4.032	4.773	5.893	6.869
6	.718	.906	1.134	1.440	1.943	2.447	2.612	3.143	3.707	4.317	5.208	5.959
7	.711	.896	1.119	1.415	1.895	2.365	2.517	2.998	3.499	4.029	4.785	5.408
8	.706	.889	1.108	1.397	1.860	2.306	2.449	2.896	3.355	3.833	4.501	5.041
9	.703	.883	1.100	1.383	1.833	2.262	2.398	2.821	3.250	3.690	4.297	4.781
10	.700	.879	1.093	1.372	1.812	2.228	2.359	2.764	3.169	3.581	4.144	4.587
11	.697	.876	1.088	1.363	1.796	2.201	2.328	2.718	3.106	3.497	4.025	4.437
12	.695	.873	1.083	1.356	1.782	2.179	2.303	2.681	3.055	3.428	3.930	4.318
13	.694	.870	1.079	1.350	1.771	2.160	2.282	2.650	3.012	3.372	3.852	4.221
14	.692	.868	1.076	1.345	1.761	2.145	2.264	2.624	2.977	3.326	3.787	4.140
15	.691	.866	1.074	1.341	1.753	2.131	2.249	2.602	2.947	3.286	3.733	4.073
16	.690	.865	1.071	1.337	1.746	2.120	2.235	2.583	2.921	3.252	3.686	4.015
17	.689	.863	1.069	1.333	1.740	2.110	2.224	2.567	2.898	3.222	3.646	3.965
18	.688	.862	1.067	1.330	1.734	2.101	2.214	2.552	2.878	3.197	3.611	3.922
19	.688	.861	1.066	1.328	1.729	2.093	2.205	2.539	2.861	3.174	3.579	3.883
20	.687	.860	1.064	1.325	1.725	2.086	2.197	2.528	2.845	3.153	3.552	3.850
21	.686	.859	1.063	1.323	1.721	2.080	2.189	2.518	2.831	3.135	3.527	3.819
22	.686	.858	1.061	1.321	1.717	2.074	2.183	2.508	2.819	3.119	3.505	3.792
23	.685	.858	1.060	1.319	1.714	2.069	2.177	2.500	2.807	3.104	3.485	3.768
24	.685	.857	1.059	1.318	1.711	2.064	2.172	2.492	2.797	3.091	3.467	3.745
25	.684	.856	1.058	1.316	1.708	2.060	2.167	2.485	2.787	3.078	3.450	3.725
26	.684	.856	1.058	1.315	1.706	2.056	2.162	2.479	2.779	3.067	3.435	3.707
27	.684	.855	1.057	1.314	1.703	2.052	2.158	2.473	2.771	3.057	3.421	3.690
28	.683	.855	1.056	1.313	1.701	2.048	2.154	2.467	2.763	3.047	3.408	3.674
29	.683	.854	1.055	1.311	1.699	2.045	2.150	2.462	2.756	3.038	3.396	3.659
30	.683	.854	1.055	1.310	1.697	2.042	2.147	2.457	2.750	3.030	3.385	3.646
40	.681	.851	1.050	1.303	1.684	2.021	2.123	2.423	2.704	2.971	3.307	3.551
50	.679	.849	1.047	1.299	1.676	2.009	2.109	2.403	2.678	2.937	3.261	3.496
60	.679	.848	1.045	1.296	1.671	2.000	2.099	2.390	2.660	2.915	3.232	3.460
80	.678	.846	1.043	1.292	1.664	1.990	2.088	2.374	2.639	2.887	3.195	3.416
100	.677	.845	1.042	1.290	1.660	1.984	2.081	2.364	2.626	2.871	3.174	3.390
1000	.675	.842	1.037	1.282	1.646	1.962	2.056	2.330	2.581	2.813	3.098	3.300
∞	.674	.841	1.036	1.282	1.645	1.960	2.054	2.326	2.576	2.807	3.091	3.291
	50%	60%	70%	80%	90%	95%	96%	98%	99%	99.5%	99.8%	99.9%
	Confidence level C											

Ước tính khoảng tin cậy: Statistical Significance

- Are M_1 & M_2 **significantly different**?
 - Compute t . Select *significance level* (e.g. $sig = 5\%$)
 - Consult table for t-distribution: Find t value corresponding to $k-1$ degrees of freedom (here, 9)
 - t-distribution is symmetric: typically upper % points of distribution shown → look up value for **confidence limit** $z=sig/2$ (here, 0.025)
 - **If $t > z$ or $t < -z$** , then t value lies in rejection region:
 - **Reject null hypothesis** that mean error rates of M_1 & M_2 are same
 - Conclude: statistically significant difference between M_1 & M_2
 - **Otherwise**, conclude that any difference is **chance**

Lựa chọn mô hình: ROC Curves

- **ROC** (Receiver Operating Characteristics) curves: for visual comparison of classification models
- Originated from signal detection theory
- Shows the trade-off between the true positive rate and the false positive rate
- The area under the ROC curve is a measure of the accuracy of the model
- Rank the test tuples in decreasing order: the one that is most likely to belong to the positive class appears at the top of the list
- The closer to the diagonal line (i.e., the closer the area is to 0.5), the less accurate is the model




- Vertical axis represents the true positive rate
- Horizontal axis rep. the false positive rate
- The plot also shows a diagonal line
- A model with perfect accuracy will have an area of 1.0

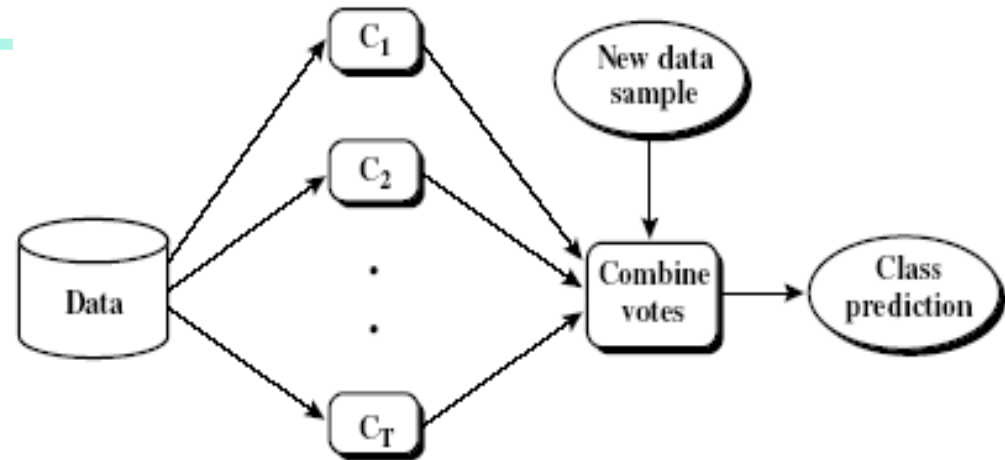
Các vấn đề ảnh hưởng việc chọn mô hình

- **Độ chính xác/ Accuracy**
 - classifier accuracy: predicting class label
- **Tốc độ/ Speed**
 - time to construct the model (training time)
 - time to use the model (classification/prediction time)
- **Robustness**: handling noise and missing values
- **Khả năng mở rộng/ Scalability**: efficiency in disk-resident databases
- **Khả năng giải thích/ Interpretability**
 - understanding and insight provided by the model
- Other measures, e.g., goodness of rules, such as decision tree size or compactness of classification rules

Chương 8. Phân loại: Các khái niệm cơ bản

- Bài toán Classification: Các khái niệm cơ bản
- Cây quyết định/ Decision Tree Induction
- Phân loại Bayes/ Bayes Classification Methods
- Phân loại dựa trên luật/ Rule-Based Classification
- Đánh giá và lựa chọn model/ Model Evaluation and Selection
- **Những kỹ thuật tăng độ chính xác trong bài toán phân loại: Ensemble Methods** 
- Tóm tắt/ Summary

PP tổng hợp: Tăng độ chính xác



■ Ensemble methods

- Sử dụng nhiều mô hình, tăng độ chính xác của thuật toán
- Kết hợp k mô hình học được, M_1, M_2, \dots, M_k , với mục đích đạt được mô hình tốt hơn M^*

■ Các phương pháp tổng hợp phổ biến

- Bagging: averaging the prediction over a collection of classifiers
- Boosting: weighted vote with a collection of classifiers
- Ensemble: combining a set of heterogeneous classifiers

Bagging: Bootstrap Aggregation

- Analogy: Diagnosis based on multiple doctors' majority vote
- Training
 - Given a set D of d tuples, at each iteration i , a training set D_i of d tuples is sampled with replacement from D (i.e., bootstrap)
 - A classifier model M_i is learned for each training set D_i
- Classification: classify an unknown sample X
 - Each classifier M_i returns its class prediction
 - The bagged classifier M^* counts the votes and assigns the class with the most votes to X
- Prediction: can be applied to the prediction of continuous values by taking the average value of each prediction for a given test tuple
- Accuracy
 - Often significantly better than a single classifier derived from D
 - For noise data: not considerably worse, more robust
 - Proved improved accuracy in prediction

Boosting

- Analogy: Consult several doctors, based on a combination of weighted diagnoses—weight assigned based on the previous diagnosis accuracy
- How boosting works?
 - **Weights** are assigned to each training tuple
 - A series of k classifiers is iteratively learned
 - After a classifier M_i is learned, the weights are updated to allow the subsequent classifier, M_{i+1} , to **pay more attention to the training tuples that were misclassified** by M_i
 - The final **M^*** **combines the votes** of each individual classifier, where the weight of each classifier's vote is a function of its accuracy
- Boosting algorithm can be extended for numeric prediction
- Comparing with bagging: Boosting tends to have greater accuracy, but it also risks overfitting the model to misclassified data

Adaboost (Freund and Schapire, 1997)

- Given a set of d class-labeled tuples, $(\mathbf{X}_1, y_1), \dots, (\mathbf{X}_d, y_d)$
- Initially, all the weights of tuples are set the same ($1/d$)
- Generate k classifiers in k rounds. At round i ,
 - Tuples from D are sampled (with replacement) to form a training set D_i of the same size
 - Each tuple's chance of being selected is based on its weight
 - A classification model M_i is derived from D_i
 - Its error rate is calculated using D_i as a test set
 - If a tuple is misclassified, its weight is increased, o.w. it is decreased
- Error rate: $err(\mathbf{X}_j)$ is the misclassification error of tuple \mathbf{X}_j . Classifier M_i error rate is the sum of the weights of the misclassified tuples:

$$error(M_i) = \sum_j^d w_j \times err(\mathbf{X}_j)$$

- The weight of classifier M_i 's vote is $\log \frac{1 - error(M_i)}{error(M_i)}$


Random Forest (Breiman 2001)

- Random Forest:
 - Each classifier in the ensemble is a *decision tree* classifier and is generated using a random selection of attributes at each node to determine the split
 - During classification, each tree votes and the most popular class is returned
- Two Methods to construct Random Forest:
 - Forest-RI (*random input selection*): Randomly select, at each node, F attributes as candidates for the split at the node. The CART methodology is used to grow the trees to maximum size
 - Forest-RC (*random linear combinations*): Creates new attributes (or features) that are a linear combination of the existing attributes (reduces the correlation between individual classifiers)
- Comparable in accuracy to Adaboost, but more robust to errors and outliers
- Insensitive to the number of attributes selected for consideration at each split, and faster than bagging or boosting

Classification of Class-Imbalanced Data Sets

- Class-imbalance problem: Rare positive example but numerous negative ones, e.g., medical diagnosis, fraud, oil-spill, fault, etc.
- Traditional methods assume a balanced distribution of classes and equal error costs: not suitable for class-imbalanced data
- Typical methods for imbalance data in 2-class classification:
 - **Oversampling**: re-sampling of data from positive class
 - **Under-sampling**: randomly eliminate tuples from negative class
 - **Threshold-moving**: moves the decision threshold, t , so that the rare class tuples are easier to classify, and hence, less chance of costly false negative errors
 - **Ensemble techniques**: Ensemble multiple classifiers introduced above
- Still difficult for class imbalance problem on multiclass tasks

Chương 8. Phân loại: Các khái niệm cơ bản

- Bài toán Classification: Các khái niệm cơ bản
- Cây quyết định/ Decision Tree Induction
- Phân loại Bayes/ Bayes Classification Methods
- Phân loại dựa trên luật/ Rule-Based Classification
- Đánh giá và lựa chọn model/ Model Evaluation and Selection
- Những kỹ thuật tăng độ chính xác trong bài toán phân loại: Ensemble Methods
- **Tóm tắt/ Summary** 

Tóm tắt (I)

- **Phân lớp/ Classification** là một dạng phân tích dữ liệu rút ra mô hình/ **models** mô tả các lớp dữ liệu quan trọng.
- Các phương pháp hiệu quả, có khả năng mở rộng đã được phát triển cho **decision tree induction**, **Naive Bayesian classification**, **rule-based classification**, và nhiều phương pháp phân lớp khác.
- **Các chỉ số đánh giá/ Evaluation metrics**: accuracy, sensitivity, specificity, precision, recall, F measure, và F_β measure.
- **Stratified k-fold cross-validation** được đề xuất để tính độ chính xác. **Bagging** và **boosting** có thể được dùng tăng độ chính xác bằng việc học và kết hợp các mô hình đơn lẻ.

Tóm tắt (II)

- **Significance tests** và **ROC curves** là hữu ích cho việc lựa chọn mô hình.
- Chủ đề nghiên cứu: Phương pháp so sánh các mô hình phân lớp khác nhau/ **comparisons of the different classification**
- Không có một mô hình nào là tốt nhất đối với tất cả các kiểu/ loại dữ liệu.
- Các vấn đề như độ chính xác, thời gian huấn luyện, sự mạnh mẽ, khả năng mở rộng, và khả năng giải thích phải được nhắc đến và cân đối để tìm ra mô hình phân lớp tốt nhất đối với từng bài toán cụ thể.

References (1)

- C. Apte and S. Weiss. **Data mining with decision trees and decision rules**. Future Generation Computer Systems, 13, 1997
- C. M. Bishop, **Neural Networks for Pattern Recognition**. Oxford University Press, 1995
- L. Breiman, J. Friedman, R. Olshen, and C. Stone. **Classification and Regression Trees**. Wadsworth International Group, 1984
- C. J. C. Burges. **A Tutorial on Support Vector Machines for Pattern Recognition**. *Data Mining and Knowledge Discovery*, 2(2): 121-168, 1998
- P. K. Chan and S. J. Stolfo. **Learning arbiter and combiner trees from partitioned data for scaling machine learning**. KDD'95
- H. Cheng, X. Yan, J. Han, and C.-W. Hsu, **Discriminative Frequent Pattern Analysis for Effective Classification**, ICDE'07
- H. Cheng, X. Yan, J. Han, and P. S. Yu, **Direct Discriminative Pattern Mining for Effective Classification**, ICDE'08
- W. Cohen. **Fast effective rule induction**. ICML'95
- G. Cong, K.-L. Tan, A. K. H. Tung, and X. Xu. **Mining top-k covering rule groups for gene expression data**. SIGMOD'05

References (2)

- A. J. Dobson. **An Introduction to Generalized Linear Models**. Chapman & Hall, 1990.
- G. Dong and J. Li. **Efficient mining of emerging patterns: Discovering trends and differences**. KDD'99.
- R. O. Duda, P. E. Hart, and D. G. Stork. **Pattern Classification**, 2ed. John Wiley, 2001
- U. M. Fayyad. **Branching on attribute values in decision tree generation**. AAAI'94.
- Y. Freund and R. E. Schapire. **A decision-theoretic generalization of on-line learning and an application to boosting**. J. Computer and System Sciences, 1997.
- J. Gehrke, R. Ramakrishnan, and V. Ganti. **Rainforest: A framework for fast decision tree construction of large datasets**. VLDB'98.
- J. Gehrke, V. Gant, R. Ramakrishnan, and W.-Y. Loh, **BOAT -- Optimistic Decision Tree Construction**. SIGMOD'99.
- T. Hastie, R. Tibshirani, and J. Friedman. **The Elements of Statistical Learning: Data Mining, Inference, and Prediction**. Springer-Verlag, 2001.
- D. Heckerman, D. Geiger, and D. M. Chickering. **Learning Bayesian networks: The combination of knowledge and statistical data**. Machine Learning, 1995.
- W. Li, J. Han, and J. Pei, **CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules**, ICDM'01.

References (3)

- T.-S. Lim, W.-Y. Loh, and Y.-S. Shih. **A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms.** Machine Learning, 2000.
- J. Magidson. **The Chaid approach to segmentation modeling: Chi-squared automatic interaction detection.** In R. P. Bagozzi, editor, Advanced Methods of Marketing Research, Blackwell Business, 1994.
- M. Mehta, R. Agrawal, and J. Rissanen. **SLIQ : A fast scalable classifier for data mining.** EDBT'96.
- T. M. Mitchell. **Machine Learning.** McGraw Hill, 1997.
- S. K. Murthy, **Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey,** Data Mining and Knowledge Discovery 2(4): 345-389, 1998
- J. R. Quinlan. **Induction of decision trees.** *Machine Learning*, 1:81-106, 1986.
- J. R. Quinlan and R. M. Cameron-Jones. **FOIL: A midterm report.** ECML'93.
- J. R. Quinlan. **C4.5: Programs for Machine Learning.** Morgan Kaufmann, 1993.
- J. R. Quinlan. **Bagging, boosting, and c4.5.** AAAI'96.

References (4)

- R. Rastogi and K. Shim. **Public: A decision tree classifier that integrates building and pruning.** VLDB'98.
- J. Shafer, R. Agrawal, and M. Mehta. **SPRINT : A scalable parallel classifier for data mining.** VLDB'96.
- J. W. Shavlik and T. G. Dietterich. **Readings in Machine Learning.** Morgan Kaufmann, 1990.
- P. Tan, M. Steinbach, and V. Kumar. **Introduction to Data Mining.** Addison Wesley, 2005.
- 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.
- S. M. Weiss and N. Indurkha. **Predictive Data Mining.** Morgan Kaufmann, 1997.
- I. H. Witten and E. Frank. **Data Mining: Practical Machine Learning Tools and Techniques**, 2ed. Morgan Kaufmann, 2005.
- X. Yin and J. Han. **CPAR: Classification based on predictive association rules.** SDM'03
- H. Yu, J. Yang, and J. Han. **Classifying large data sets using SVM with hierarchical clusters.** KDD'03.



CS412 Midterm Exam Statistics

- Opinion Question Answering:
 - Like the style: 70.83%, dislike: 29.16%
 - Exam is hard: 55.75%, easy: 0.6%, just right: 43.63%
 - Time: plenty: 3.03%, enough: 36.96%, not: 60%
- Score distribution: # of students (Total: 180)

■ ≥ 90 : 24	■ 60-69: 37	■ < 40 : 2
■ 80-89: 54	■ 50-59: 15	
■ 70-79: 46	■ 40-49: 2	
- Final grading are based on overall score accumulation and relative class distributions

Issues: Evaluating Classification Methods

- Accuracy
 - classifier accuracy: predicting class label
 - predictor accuracy: guessing value of predicted attributes
- Speed
 - time to construct the model (training time)
 - time to use the model (classification/prediction time)
- Robustness: handling noise and missing values
- Scalability: efficiency in disk-resident databases
- Interpretability
 - understanding and insight provided by the model
- Other measures, e.g., goodness of rules, such as decision tree size or compactness of classification rules

Predictor Error Measures

- Measure predictor accuracy: measure how far off the predicted value is from the actual known value

- **Loss function:** measures the error betw. y_i and the predicted value y_i'

- Absolute error: $|y_i - y_i'|$

- Squared error: $(y_i - y_i')^2$

- Test error (generalization error): the average loss over the test set

- Mean absolute error: $\frac{\sum_{i=1}^d |y_i - y_i'|}{d}$ Mean squared error: $\frac{\sum_{i=1}^d (y_i - y_i')^2}{d}$

- Relative absolute error: $\frac{\sum_{i=1}^d |y_i - y_i'|}{\sum_{i=1}^d |y_i - \bar{y}|}$ Relative squared error: $\frac{\sum_{i=1}^d (y_i - y_i')^2}{\sum_{i=1}^d (y_i - \bar{y})^2}$

The mean squared-error exaggerates the presence of outliers

Popularly use (square) root mean-square error, similarly, root relative squared error

Scalable Decision Tree Induction Methods

- **SLIQ** (EDBT'96 — Mehta et al.)
 - Builds an index for each attribute and only class list and the current attribute list reside in memory
- **SPRINT** (VLDB'96 — J. Shafer et al.)
 - Constructs an attribute list data structure
- **PUBLIC** (VLDB'98 — Rastogi & Shim)
 - Integrates tree splitting and tree pruning: stop growing the tree earlier
- **RainForest** (VLDB'98 — Gehrke, Ramakrishnan & Ganti)
 - Builds an AVC-list (attribute, value, class label)
- **BOAT** (PODS'99 — Gehrke, Ganti, Ramakrishnan & Loh)
 - Uses bootstrapping to create several small samples

Data Cube-Based Decision-Tree Induction

- Integration of generalization with decision-tree induction (Kamber et al.'97)
- Classification at primitive concept levels
 - E.g., precise temperature, humidity, outlook, etc.
 - Low-level concepts, scattered classes, bushy classification-trees
 - Semantic interpretation problems
- Cube-based multi-level classification
 - Relevance analysis at multi-levels
 - Information-gain analysis with dimension + level