CS570: Introduction to Data Mining

Classification Basics

Reading: Chapter 8 Han, Chapters 4 & 5 Tan

Anca Doloc-Mihu, Ph.D.

Slides courtesy of Li Xiong, Ph.D.,

September 17, 2013

Classification and Prediction

- Overview
- Classification algorithms and methods
  - Decision tree induction
  - Bayesian classification
  - kNN classification
  - Support Vector Machines (SVM)
  - Neural Networks
- Regression
- Evaluation and measures
- Ensemble methods

Motivating Example – Fruit Identification

Skin	Color	Size	Flesh	Conclusion
Hairy	Brown	Large	Hard	Safe
Hairy	Green	Large	Hard	Safe
Smooth	Red	Large	Soft	Dangerous
Hairy	Green	Large	Soft	Safe
Smooth	Red	Small	Hard	Dangerous
…				

Classification vs. Prediction

- Classification
  - predicts categorical class labels (discrete, unordered)
  - constructs a model based on the training set and uses it in classifying new data (a classifier is constructed to predict class labels)
- Prediction (Regression)
  - models continuous-valued functions, i.e., predicts numeric values, unknown or missing values
- Typical applications
  - Credit approval
  - Target marketing
  - Medical diagnosis
  - Fraud detection

Example – Credit Approval

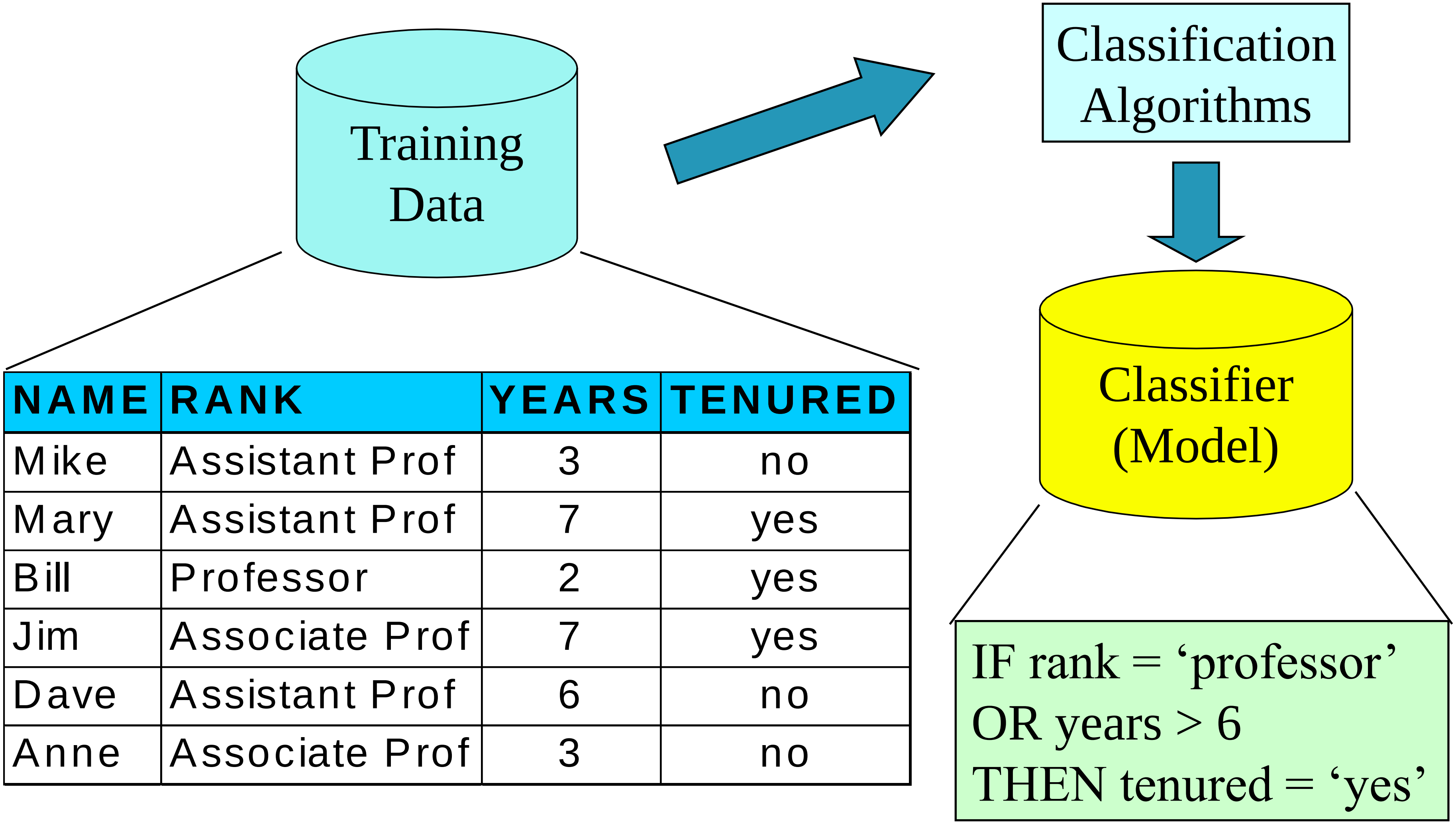
Name	Age	Income	…	Credit
Clark	35	High	…	Excellent
Milton	38	High	…	Excellent
Neo	25	Medium	…	Fair
…	…	…	…	…

- Classification rule (classifier):
    - If age = "31...40" and income = high then credit_rating = excellent
- Future customers
    - Paul: age = 35, income = high ⇒ excellent credit rating
    - John: age = 20, income = medium ⇒ fair credit rating

Classification—A Two-Step Process

- Model construction (learning step): describing a set of predetermined classes
  - Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute (known)
  - The set of tuples used for model construction is training set
  - The model is represented as classification rules, decision trees, or mathematical formulae
- Model usage: for classifying future or unknown objects
  - Estimate accuracy of the model
    - The known label of test sample is compared with the classified result from the model
    - Accuracy rate is the percentage of test set samples that are correctly classified by the model (also error rate)
    - Test set is independent of training set, otherwise over-fitting will occur
  - If the accuracy is acceptable, use the model to classify data tuples whose class labels are not known

Process (1): Model Construction



Training Data

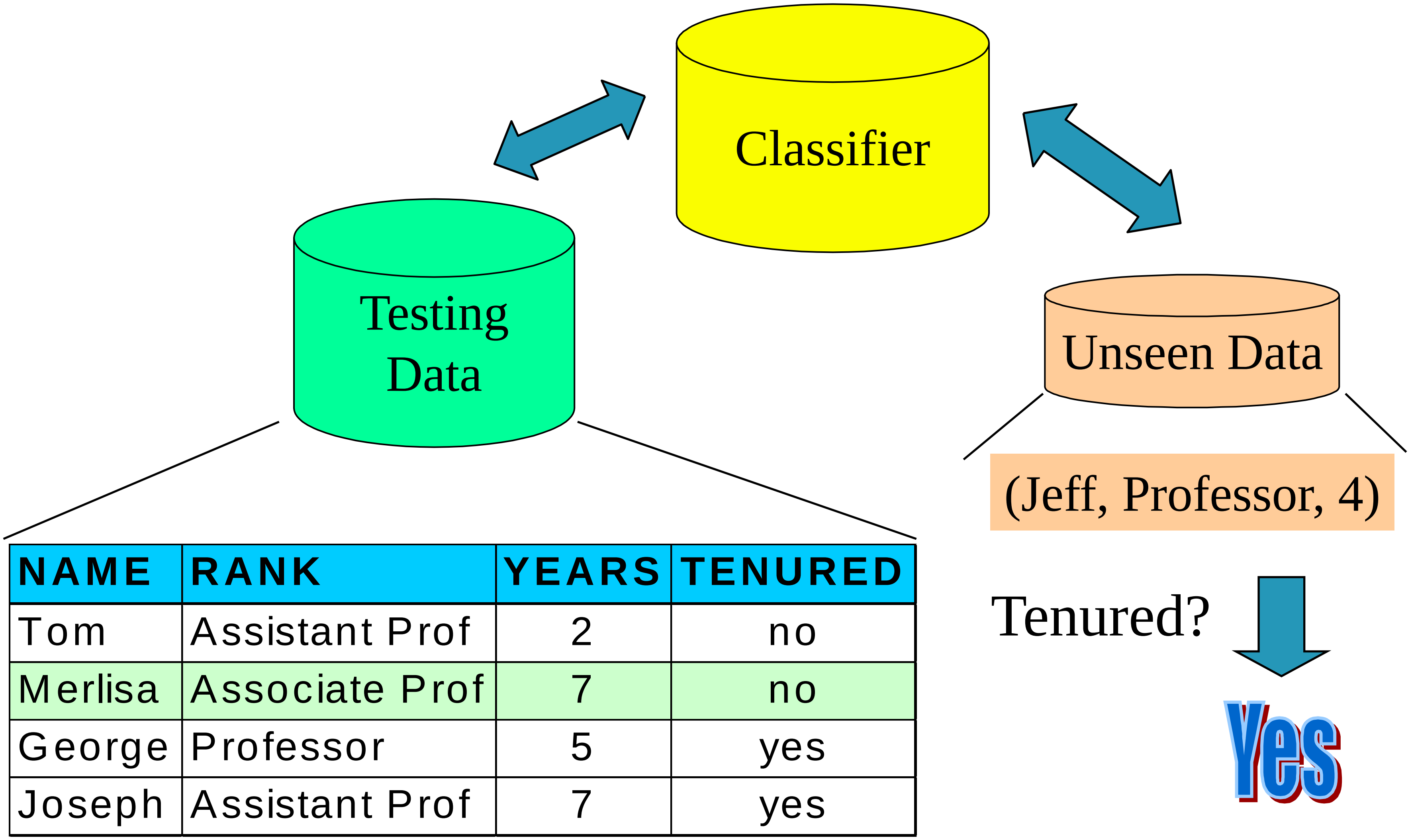
Classification Algorithms

Classifier (Model)

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

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

Process (2): Using the Model in Prediction



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

Supervised vs. Unsupervised Learning

- Supervised learning (classification)
  - Supervision: The training data (observations, measurements, etc.) are accompanied by known labels indicating the class of the observations
  - New data is classified based on the training set
- Unsupervised learning (clustering)
  - The class labels of training data is unknown
  - Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data

Issues: Evaluating Classification Methods

- Accuracy
- 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, decision tree size or compactness of classification rules

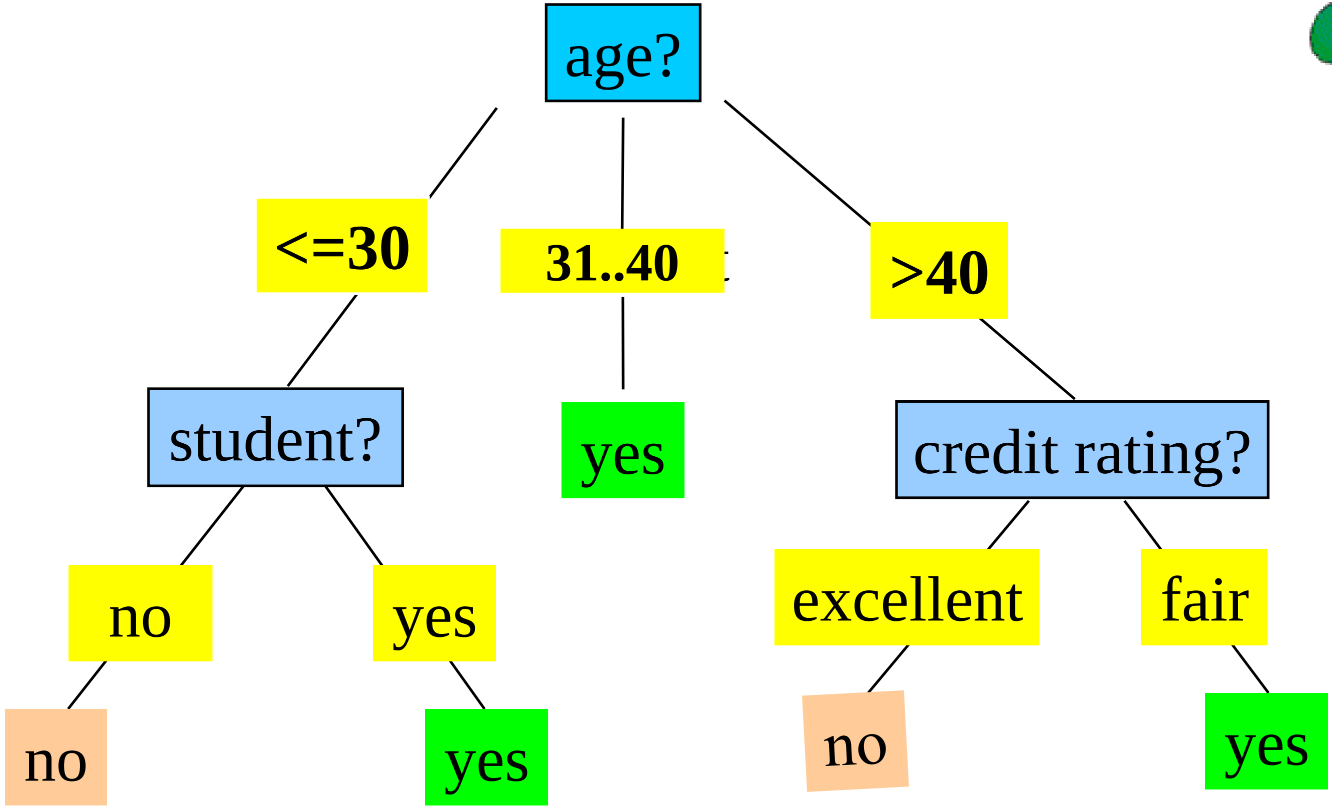
Classification and Prediction

- Overview
- Classification algorithms and methods
  - Decision tree induction
  - Bayesian classification
  - kNN classification
  - Support Vector Machines (SVM)
  - Others
- Evaluation and measures
- Ensemble methods

Training Dataset

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

A Decision Tree for "*buys_computer*"


age?

<=30

31..40

>40

student?

yes

credit rating?

no

yes

excellent

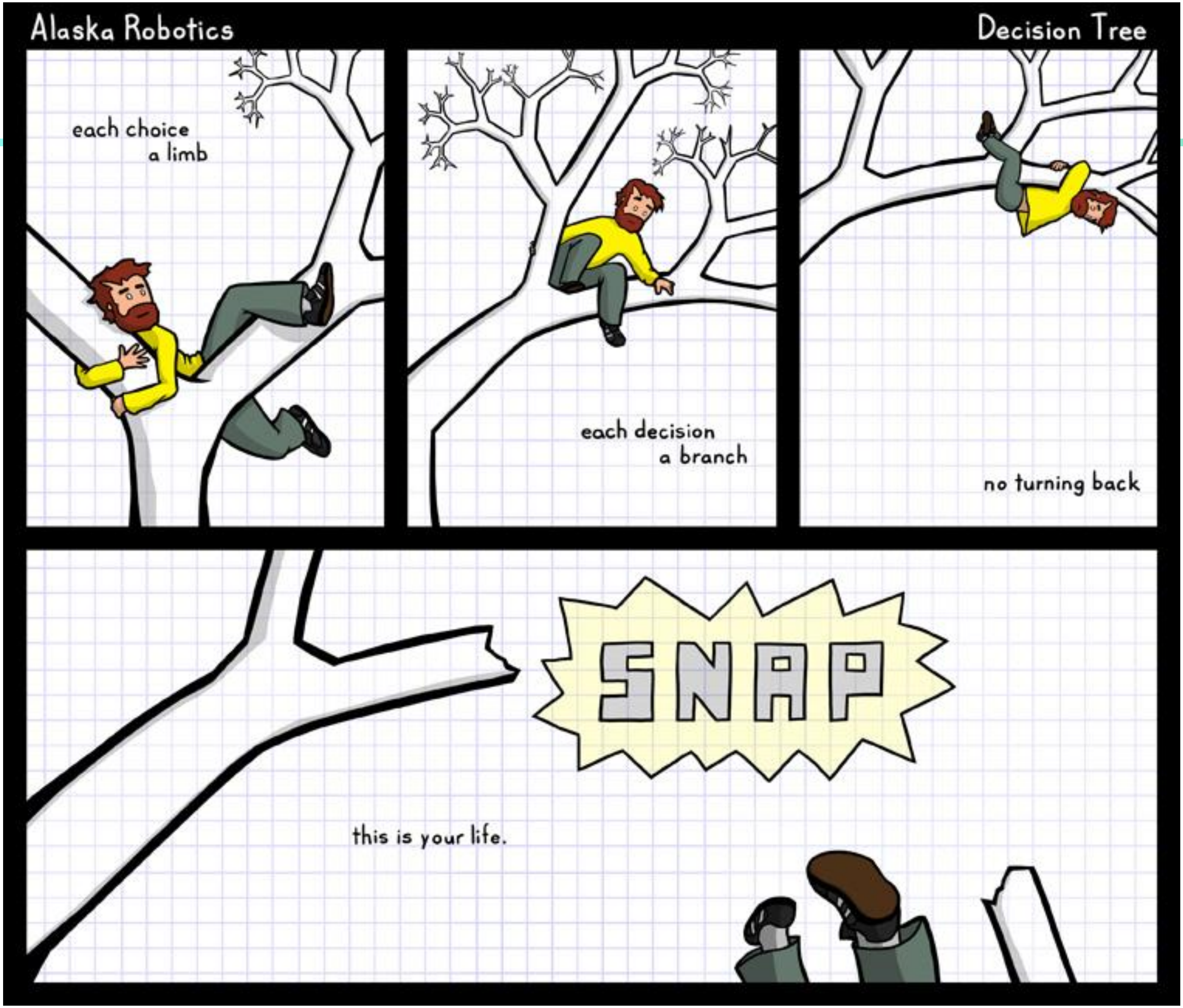
fair

no

yes

no

yes


Alaska Robotics
Decision Tree
each choice a limb
each decision a branch
no turning back
SNAP
this is your life.

Algorithm for Decision Tree Induction

- ID3 (Iterative Dichotomiser), C4.5, by Quinlan – 1970's-1980's
- CART (Classification and Regression Trees) - 1984
- Basic algorithm (a greedy algorithm) - tree is constructed with top-down recursive partitioning
  - At start, all the training examples are at the root
  - A test attribute is selected that "best" separate the data into partitions
  - Samples are partitioned recursively based on selected attributes
- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left
- Cost: n x |D| x log(|D|), n no attributes, |D| no tuples in training set D

Attribute Selection Measures

- Idea: select attribute that partition samples into homogeneous groups
- Measures
  - Information gain (ID3)
  - Gain ratio (C4.5)
  - Gini index (CART)

Attribute Selection Measure: Information Gain (ID3)

- Select the attribute with the **highest information gain**
- Let p_i be the probability that an arbitrary tuple in D belongs to class C_i, estimated by $|C_{i,D}|/|D|$
- Information (entropy) needed to classify a tuple in D (before split):

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

- Information needed (after using A to split D into v partitions) to classify D:

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

- Information gain – difference between before and after splitting on attribute A

$$Gain(A) = Info(D) - Info_A(D)$$

Example: Information Gain

- Class P: buys_computer = “yes”,
- Class N: buys_computer = “no”

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

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

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

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

$$= 0.246$$

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

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit\_rating) = 0.048$$

Information-Gain for Continuous-Value Attributes

- Let attribute A be a continuous-valued attribute
- Must determine the *best split point* for A
  - Sort the value A in increasing order
  - Typically, the midpoint between each pair of adjacent values is considered as a possible *split point*
    - $(a_i+a_{i+1})/2$ is the midpoint between the values of a_i and a_{i+1}
  - The point with the *minimum expected information requirement* for A is selected as the split-point for A
- Split:
  - D1 is the set of tuples in D satisfying A ≤ split-point, and D2 is the set of tuples in D satisfying A > split-point

Attribute Selection Measure: Gain Ratio (C4.5)

- Information gain measure is biased towards attributes with a large # of values (# of splits)
- C4.5 uses gain ratio to overcome the problem (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)
- Ex. $SplitInfo_{Income}(D) = -\frac{4}{14} \times \log_2(\frac{4}{14}) - \frac{6}{14} \times \log_2(\frac{6}{14}) - \frac{4}{14} \times \log_2(\frac{4}{14}) = 0.926$
  - GainRatio(income) = 0.029/0.926 = 0.031
- The attribute with the **maximum gain ratio** is selected as the splitting attribute

Attribute Selection Measure: Gini index (CART)

- If a data set *D* contains examples from *n* classes, Gini index, *Gini*(*D*) is defined as

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

  where p_j is the relative frequency of class *j* in *D*

- If a data set *D* is split on A into two subsets D_1 and D_2, the *Gini* index *Gini*(*D*) is defined as

$$Gini_A(D) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2)$$

- Reduction in Impurity:

$$\Delta Gini(A) = Gini(D) - Gini_A(D)$$

- The attribute provides the **smallest** $Gini_{split}(D)$ (or the largest reduction in impurity) is chosen to split the node

Example: Gini index

- Ex. D has 9 tuples in buys_computer = "yes" and 5 in "no"

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

- Suppose the attribute income partitions D into 10 in D_1: {low, medium} and 4 in D_2 : {high}

$$Gini_{income \in \{low,medium\}}(D) = \left(\frac{10}{14}\right)Gini(D_1) + \left(\frac{4}{14}\right)Gini(D_1)$$

$$= \frac{10}{14}(1 - (\frac{6}{10})^2 - (\frac{4}{10})^2) + \frac{4}{14}(1 - (\frac{1}{4})^2 - (\frac{3}{4})^2)$$

$$= 0.450$$

$$= Gini_{income\ \in\ \{high\}}(D)$$

but $\text{Gini}_{\{\text{medium,high}\}}$ is 0.30 and thus the best since it is the lowest

Comparing Attribute Selection Measures

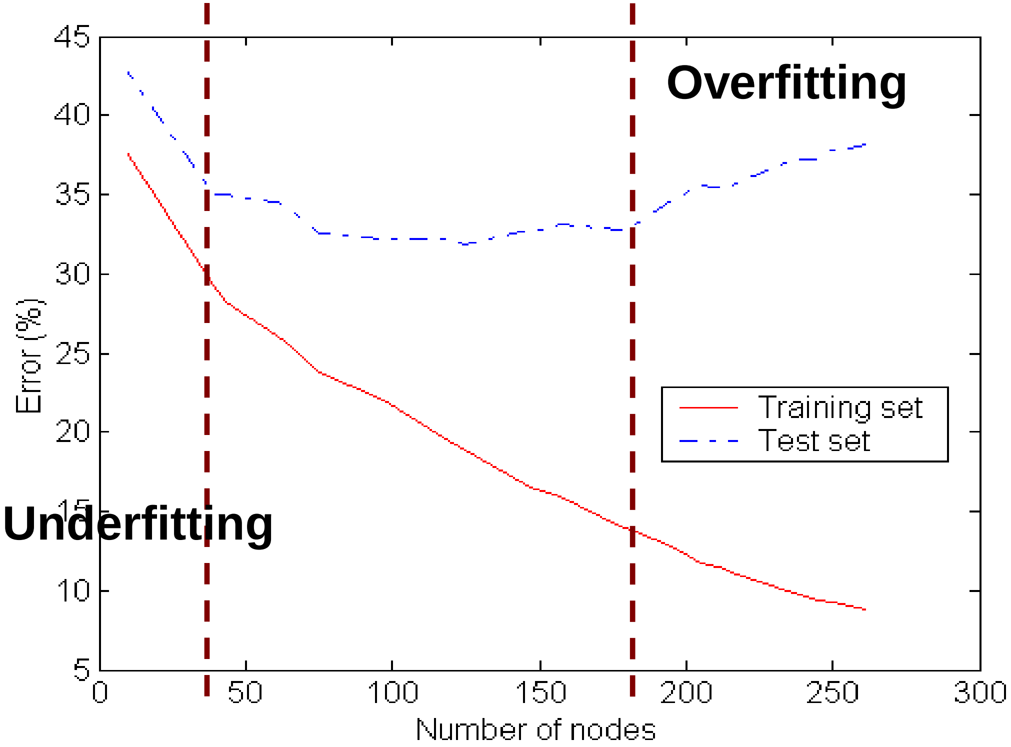
- The three measures, in general, return good results but
  - Information gain:
    - biased towards multivalued attributes
  - Gain ratio:
    - tends to prefer unbalanced splits in which one partition is much smaller than the others
  - Gini index:
    - biased to multivalued attributes
    - tends to favor tests that result in equal-sized partitions and purity in both partitions

Other Attribute Selection Measures

- 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-statistics: 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

- Overfitting: An induced tree may overfit the training data
  - Too many branches, some may reflect anomalies and noises



  - Overfitting (underfitting) are related to the cost complexity and error rate of the tree (Tan ch. 4.4)

Tree Pruning

- Two approaches to avoid overfitting
  - Prepruning: Halt tree construction early—do not split a node if this would result in the goodness measure falling below a threshold
    - Difficult to choose an appropriate threshold
  - Postpruning: Remove branches from a "fully grown" tree
    - Use a set of data different from the training data to decide which is the "best pruned tree"
    - Occam's razor: prefers smaller decision trees (simpler theories) over larger ones
    - CART – based on cost complexity and error rate of tree

Enhancements to Basic Decision Tree Induction

- Allow for continuous-valued attributes
  - Dynamically define new discrete-valued attributes that partition the continuous attribute value into a discrete set of intervals
- Handle missing attribute values
  - Assign the most common value of the attribute
  - Assign probability to each of the possible values
- Attribute construction
  - Create new attributes based on existing ones that are sparsely represented
  - This reduces fragmentation, repetition, and replication

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

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 Illustration

Training Examples

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

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.

Decision Tree: Comments

- Relatively faster learning speed (than other classification methods)
- Convertible to simple and easy to understand classification rules
- Can use SQL queries for accessing databases
- Comparable classification accuracy with other methods

Classification and Prediction

- Overview
- Classification algorithms and methods
  - Decision tree induction
  - Bayesian classification
  - kNN classification
  - Support Vector Machines (SVM)
  - Others
- Evaluation and measures
- Ensemble methods

Bayesian Classification

- A statistical classifier: performs *probabilistic prediction, i.e.,* predicts class membership probabilities
- Foundation: Based on Bayes' Theorem.
- Naïve Bayesian
  - Independence assumption
- Bayesian network
  - Concept
  - Using Bayesian network
  - Training/learning Bayesian network

Bayes’ theorem

- Bayes' theorem/rule/law relates the conditional and marginal probabilities of stochastic events
  - P(*H*) is the prior probability of H. P(*X*) is the prior probability of *X.*
  - P(H|*X*) is the conditional probability (posteriori probability) of H occurring given that *X* occurs.
  - P(*X*|H) is the conditional probability of *X* given H
  - P(*X*|H) is different from P(H|*X*).

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

- Cookie example:
  - Bowl 1: 10 chocolate + 30 plain; Bowl 2: 20 chocolate + 20 plain
  - Pick a bowl, and then pick a cookie
  - If it’s a plain cookie, what’s the probability the cookie is picked out of bowl A?
  - Event A: pick bowl 1; event B: pick plain cookie => P(A|B) = ?

Naïve Bayesian Classifier

- Naïve Bayesian / simple Bayesian: assumes the effect of an attribute value is independent of other attributes values
- Let D be a training set of tuples and their associated class labels, and each tuple is represented by an n-D attribute vector **X** = (x_1, x_2, …, x_n)
- Suppose there are *m* classes C_1, C_2, …, C_m.
- Classification is to derive the maximum posteriori, i.e., the maximal P(C_i|**X**)

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

- Since P(X) is constant for all classes, then derive maximal $P(\mathbf{X}|C_i)P(C_i)$
- P(C_i) – assumed equal if unknown, then derive maximal

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

Derivation of Naïve Bayes Classifier

- A simplified assumption: attributes are conditionally independent (i.e., no dependence relation between attributes):

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

- If attribute A_k is categorical, $P(x_k|C_i)$ is the # of tuples in C_i having value x_k for A_k divided by $|C_{i,D}|$ (# of tuples of C_i in D)
- If A_k is continous-valued, $P(x_k|C_i)$ is usually computed based on Gaussian distribution with a mean μ and standard deviation σ

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

and $P(x_k|C_i)$ is

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

Naïve Bayesian Classifier: Example

Class:
C1:buys_computer = 'yes'
C2:buys_computer = 'no'

Question: Data sample
X = (age <=30,
Income = medium,
Student = yes
Credit_rating = Fair)
belongs to which class?

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

Naïve Bayesian Classifier: Example

- $P(C_i)$: P(buys_computer = "yes") = 9/14 = 0.643
  P(buys_computer = "no") = 5/14= 0.357

- Compute $P(X|C_i)$ for each class
  P(age = "<=30" | buys_computer = "yes") = 2/9 = 0.222
  P(age = "<= 30" | buys_computer = "no") = 3/5 = 0.6
  P(income = "medium" | buys_computer = "yes") = 4/9 = 0.444
  P(income = "medium" | buys_computer = "no") = 2/5 = 0.4
  P(student = "yes" | buys_computer = "yes) = 6/9 = 0.667
  P(student = "yes" | buys_computer = "no") = 1/5 = 0.2
  P(credit_rating = "fair" | buys_computer = "yes") = 6/9 = 0.667
  P(credit_rating = "fair" | buys_computer = "no") = 2/5 = 0.4

- **X = (age <= 30 , income = medium, student = yes, credit_rating = fair)**

$P(X|C_i)$: P(X|buys_computer = "yes") = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
P(X|buys_computer = "no") = 0.6 x 0.4 x 0.2 x 0.4 = 0.019
$P(X|C_i)*P(C_i)$: P(X|buys_computer = "yes") * P(buys_computer = "yes") = 0.028
P(X|buys_computer = "no") * P(buys_computer = "no") = 0.007

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

Naïve Bayesian Classifier: Comments

- Advantages
  - Fast to train and use
  - Can be highly effective in most of the cases
- Disadvantages
  - Based on a false assumption: class conditional independence - practically, dependencies exist among variables
- Idiot's Bayesian, not so stupid after all? David J. Hand, Keming Yu, International Statistical Review, 2001

- How to deal with dependencies?
  - Bayesian Belief Networks

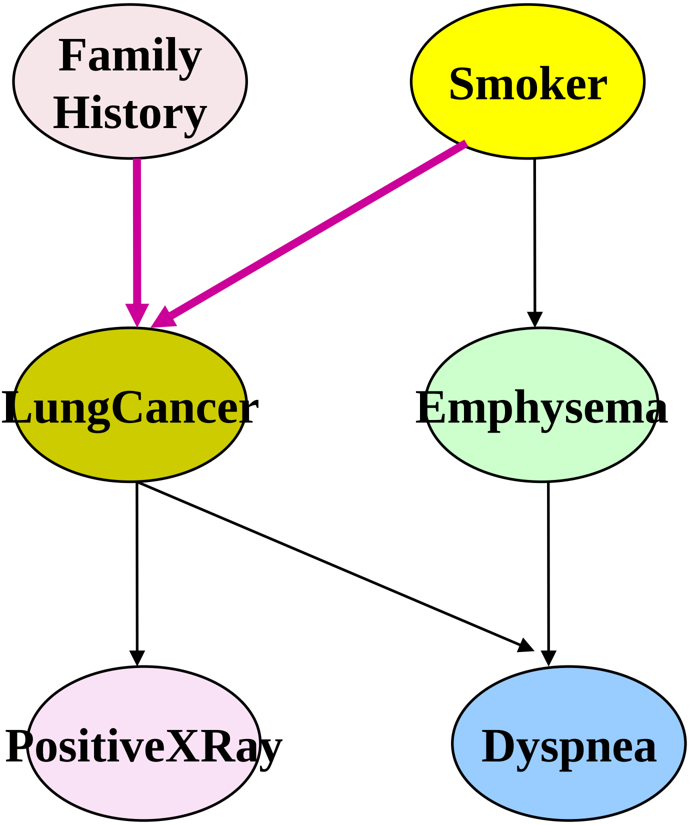
Bayesian Belief Networks – Motivating Example

- Symptoms: difficult to breath
- Patient profile: smoking? age?
- Family history?
- XRay?

Lung Cancer?

Bayesian Belief Networks

- Bayesian belief networks (belief networks, Bayesian networks, probabilistic networks) is a graphical model that represents a set of variables and their probabilistic independencies
- One of the most significant contribution in AI
- Trained Bayesian networks can be used for classification and reasoning
- Many applications: spam filtering, speech recognition, diagnostic systems

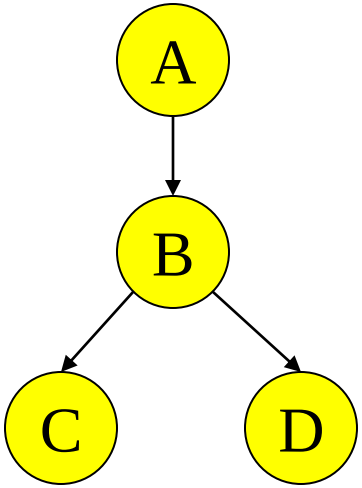


(Chapter 9 @ Han)

Bayesian Network: Definition

A Bayesian network is made up of:

1. A Directed Acyclic Graph



2. A conditional probability table for each node in the graph

A	P(A)
false	0.6
true	0.4

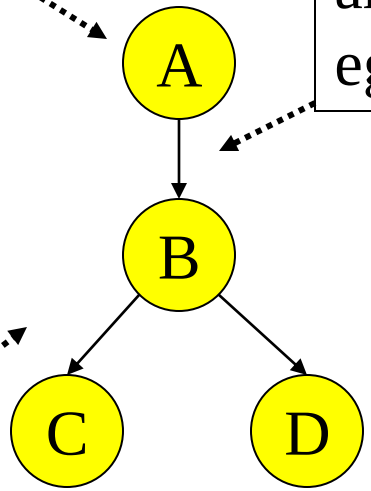
A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1

Directed Acyclic Graph

Each node in the graph is a random variable



A node X is a parent of another node Y if there is an arrow from node X to node Y eg. A is a parent of B

Informally, an arrow from node X to node Y means X has a direct influence on Y

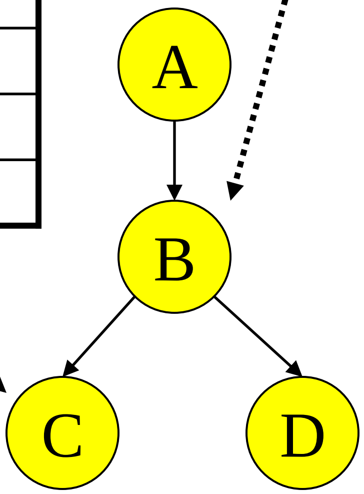
Conditional Probability Table

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

Each node X_i has a conditional probability distribution P(X_i | Parents(X_i)) that quantifies the effect of the parents on the node

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

add up to 1

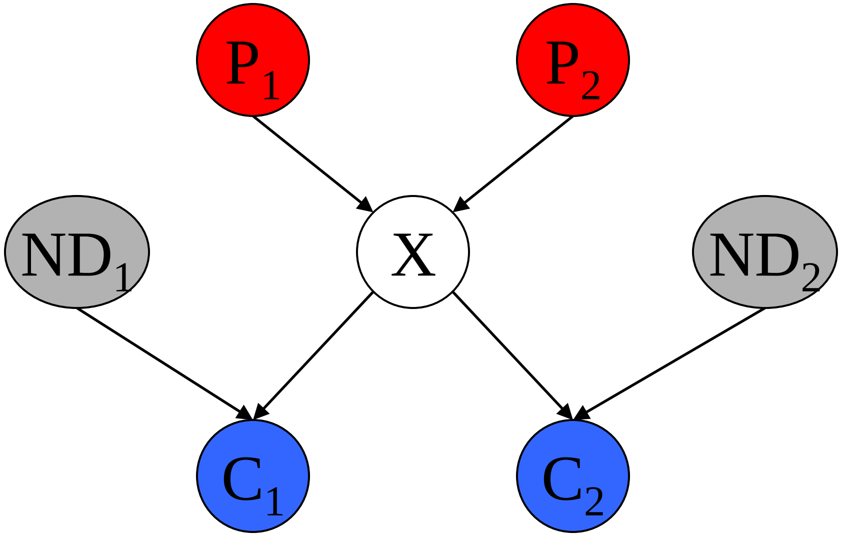
For a Boolean variable with k Boolean parents, how many probabilities need to be stored?

Bayesian Networks: Important Properties

1. Encodes the conditional independence relationships between the variables in the graph structure
2. Is a compact representation of the joint probability distribution over the variables

Conditional Independence

The Markov condition: given its parents (P_1, P_2), a node (X) is conditionally independent of its non-descendants (ND_1, ND_2)

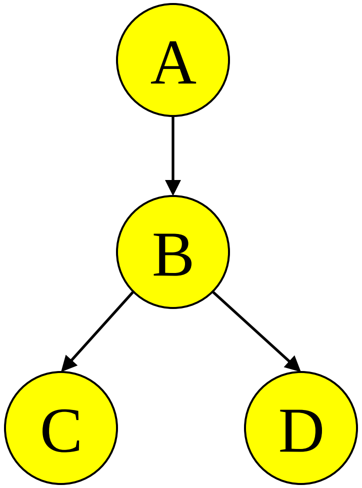


Joint Probability Distribution

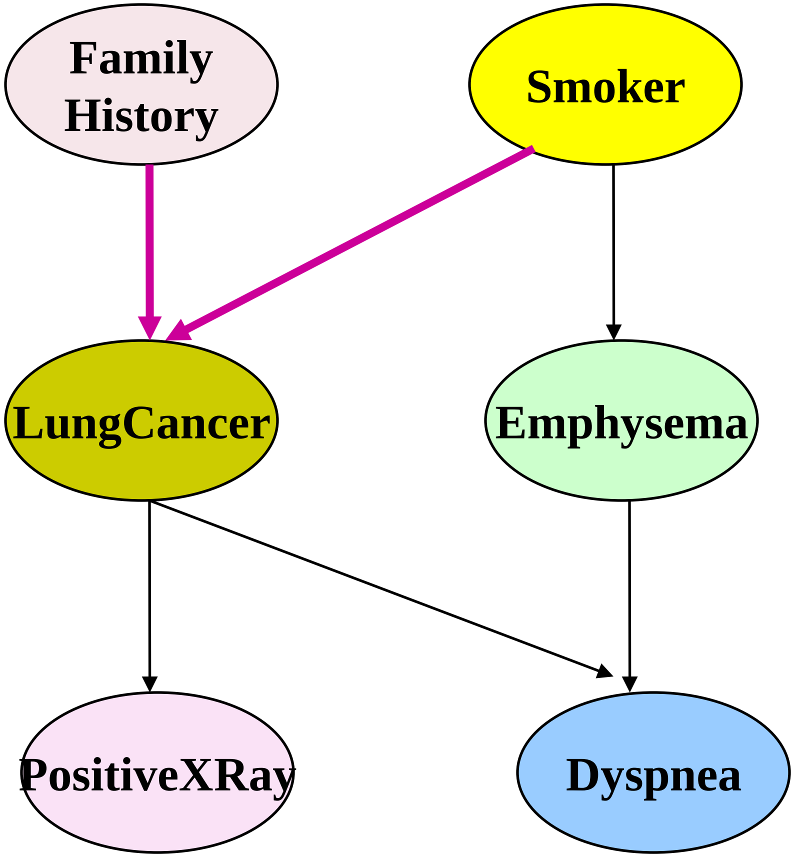
Due to the Markov condition, we can compute the joint probability distribution over all the variables X_1, …, X_n in the Bayesian net using the formula:

$$P(X_1 = x_1, \ldots, X_n = x_n) = \prod_{i=1}^{n} P(X_i = x_i \mid Parents(X_i))$$

Example:
P(A = true, B = true, C = true, D = true)
= P(A = true) * P(B = true | A = true) *
P(C = true | B = true) P(D = true | B = true)
= (0.4)*(0.3)*(0.1)*(0.95)



Bayesian Networks: Example



The **conditional probability table** (**CPT**) for variable LungCancer:

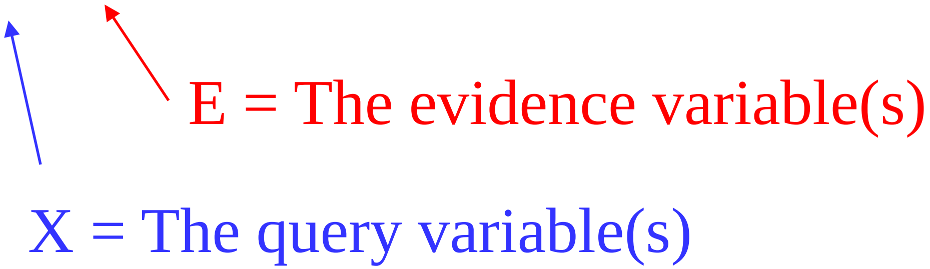
	(FH, S)	(FH, ~S)	(~FH, S)	(~FH, ~S)
LC	0.8	0.5	0.7	0.1
~LC	0.2	0.5	0.3	0.9

Using the Bayesian Network:
P(LungCancer | Smoker, PXRay, Dyspnea)?

Bayesian Belief Networks

Using Bayesian Network for Inference

- Using a Bayesian network to compute probabilities is called inference
- General form: P(X | E)



E = The evidence variable(s)

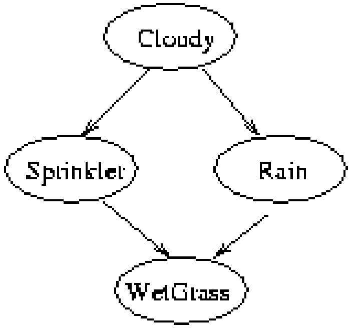
X = The query variable(s)

- Exact inference is feasible in small to medium-sized networks
- Exact inference in large networks takes a very long time
  - Approximate inference techniques which are much faster and give pretty good results

Inference Example

P(C=F)	P(C=T)
0.5	0.5

C	P(S=F)	P(S=T)
F	0.5	0.5
T	0.9	0.1



C	P(R=F)	P(R=T)
F	0.8	0.2
T	0.2	0.8

S	R	P(W=F)	P(W=T)
F	F	1.0	0.0
T	F	0.1	0.9
F	T	0.1	0.9
T	T	0.01	0.99

Joint probability:
P(C, S, R, W) = P(C) * P(S|C) * P(R|C) * P(W|S,R)

Suppose the grass is wet, which is more likely?

$$\Pr(S=1|W=1) = \frac{\Pr(S=1, W=1)}{\Pr(W=1)} = \frac{\sum_{c,r} \Pr(C=c, S=1, R=r, W=1)}{\Pr(W=1)} = 0.2781/0.6471 = 0.430$$

$$\Pr(R=1|W=1) = \frac{\Pr(R=1, W=1)}{\Pr(W=1)} = \frac{\sum_{c,s} \Pr(C=c, S=s, R=1, W=1)}{\Pr(W=1)} = 0.4581/0.6471 = 0.708$$

where

$$\Pr(W=1) = \sum_{c,r,s} \Pr(C=c, S=s, R=r, W=1) = 0.6471$$

Training Bayesian Networks

- Several scenarios:
  - Given both the network structure and all variables observable: *learn only the CPTs*
  - Network structure known, some hidden variables: *gradient descent* (greedy hill-climbing) method, analogous to neural network learning
  - Network structure unknown, all variables observable: search through the model space to *reconstruct network topology*
  - Unknown structure, all hidden variables: No good algorithms known for this purpose
- Ref. D. Heckerman: Bayesian networks for data mining

Related Graphical Models

- Bayesian networks (directed graphical model)
- Markov networks (undirected graphical model)
    - Conditional random field
- Applications:
    - Sequential data
        - Natural language text
        - Protein sequences

Chapter 8&9. Classification and Prediction

- Overview
- Classification algorithms and methods
  - Decision tree induction
  - Bayesian classification
  - Lazy learning and kNN classification
  - Support Vector Machines (SVM)
  - Others
- Evaluation and measures
- Ensemble methods

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.
- 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.
- A. J. Dobson. **An Introduction to Generalized Linear Models**. Chapman and Hall, 1990.
- G. Dong and J. Li. **Efficient mining of emerging patterns: Discovering trends and differences**. KDD'99.

References (2)

- R. O. Duda, P. E. Hart, and D. G. Stork. **Pattern Classification**, 2ed. John Wiley and Sons, 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.
- M. Kamber, L. Winstone, W. Gong, S. Cheng, and J. Han. **Generalization and decision tree induction: Efficient classification in data mining**. RIDE'97.
- B. Liu, W. Hsu, and Y. Ma. **Integrating Classification and Association Rule**. KDD'98.
- W. Li, J. Han, and J. Pei, **CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules**, ICDM'01.

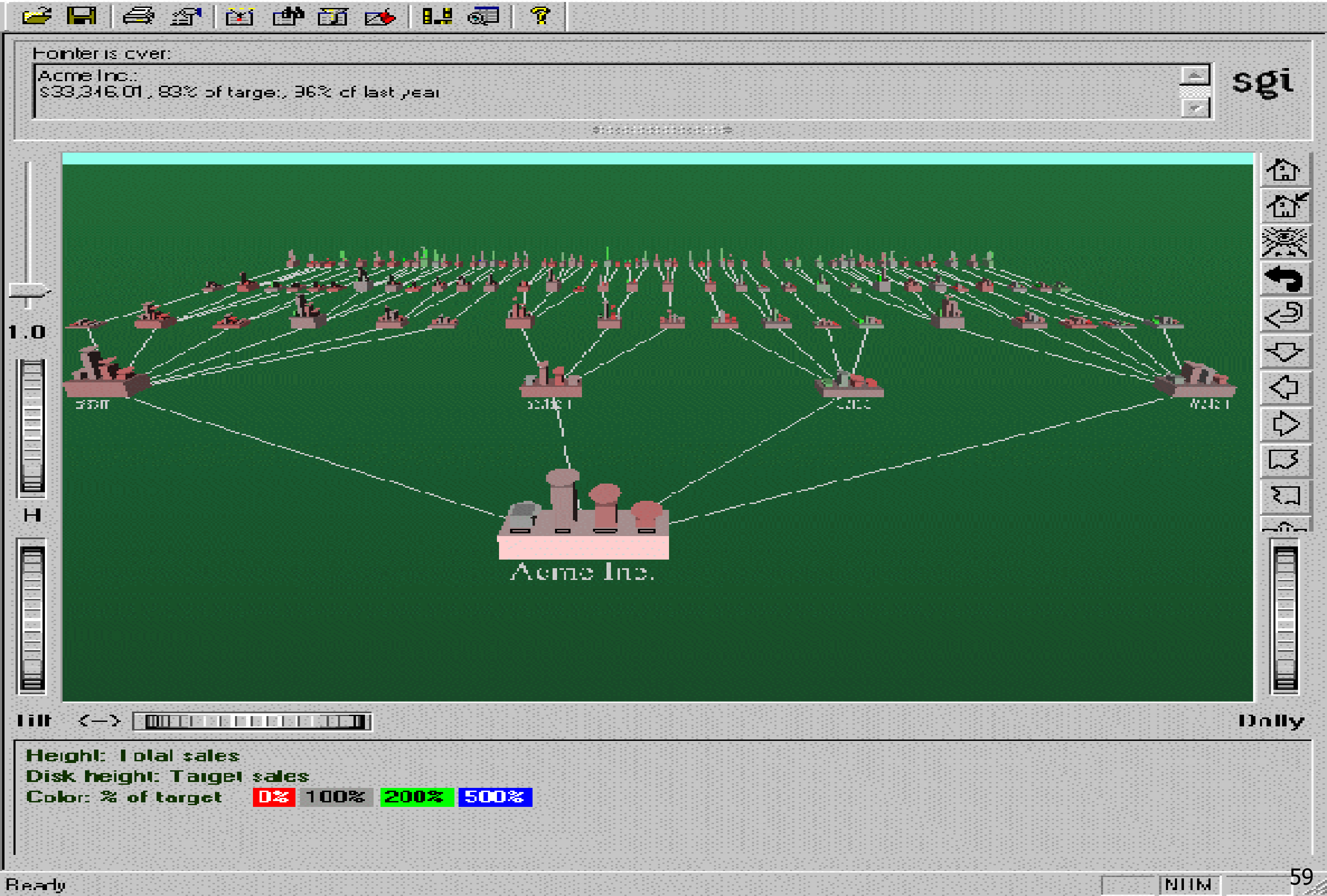
References (3)

- 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. Indurkhya. **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.

Visualization of a Decision Tree in SGI/MineSet 3.0

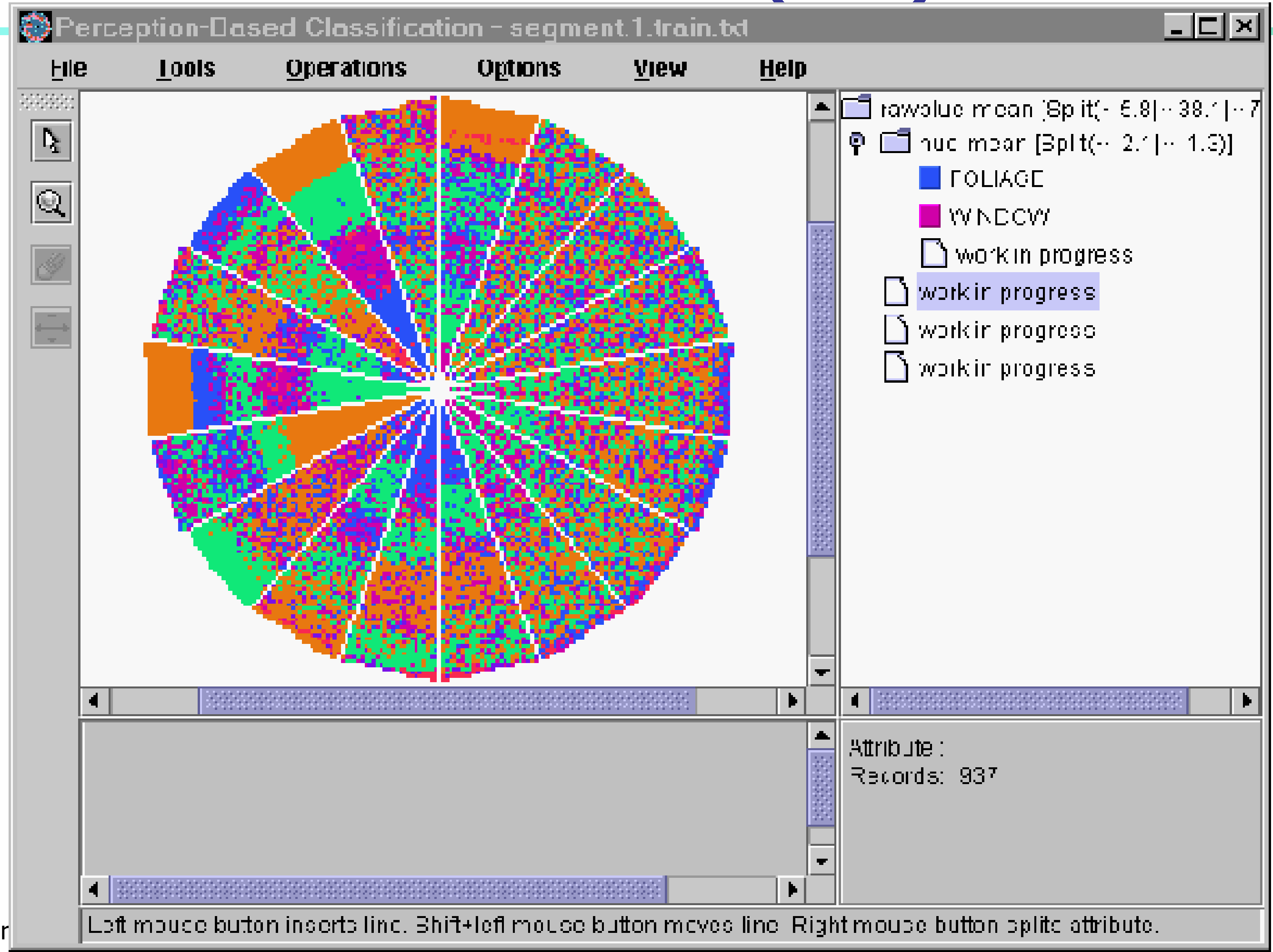


Pointer is over:

Acme Inc.:
$33,346.01, 83% of target, 36% of last year

Height: Total sales
Disk height: Target sales
Color: % of target 🟥 0% ▨ 100% 🟩 200% 🟦 500%

Interactive Visual Mining by Perception-Based Classification (PBC)



rawvalue mean [Split(-- 5.8|-- 38.1|-- 7

hue mean [Split(-- 2.1|-- 1.3)]

🟦 FOLIAGE

🟪 WINDOW

work in progress

work in progress

work in progress

work in progress

Attribute :

Records: 937

Left mouse button inserts line. Shift+left mouse button moves line. Right mouse button splits attribute.