Classification and Prediction

1. Objectives

- Techniques to classify datasets and provide categorical labels, e.g., sports, technology, kid, etc.

- Example; {credit history, salary}-> credit approval (Yes/No)

- Models to predict certain future behaviors, e.g., who is going to buy PDAs?

2. Classification vs. Prediction

2.1. Definitions

- Classification:
    - Predicts categorical class labels (discrete or nominal)
    - Classifies data (constructs a model) based on the training set and the values (class labels) in a classifying attribute and uses it in classifying new data
- Prediction:
    - Models continuous-valued functions, i.e., predicts unknown or missing values

- Typical Applications
    - Document categorization
    - Credit approval
    - Target marketing
    - Medical diagnosis
    - Treatment effectiveness analysis

2.2. Supervised vs. Unsupervised Learning

- Supervised learning (classification)

    - Supervision: The training data (observations, measurements, etc.) are accompanied by 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

2.3. Classification and Prediction Related Issues

- Data Preparation
  - Data cleaning
    - Preprocess data in order to reduce noise and handle missing values
  - Relevance analysis (**feature selection**)
    - Remove the irrelevant or redundant attributes
  - Data transformation
    - Generalize and/or normalize data

- Performance Analysis
  - Predictive accuracy:
    - Ability to classify new or previously unseen data.
  - Speed and scalability
    - Time to construct the model
    - Time to use the model
  - Robustness
    - Model makes correct predictions: Handling noise and missing values
  - Scalability
    - Efficiency in disk-resident databases
  - Interpretability:
    - Understanding and insight provided by the model
  - Goodness of rules

- Decision tree size
- Compactness of classification rules

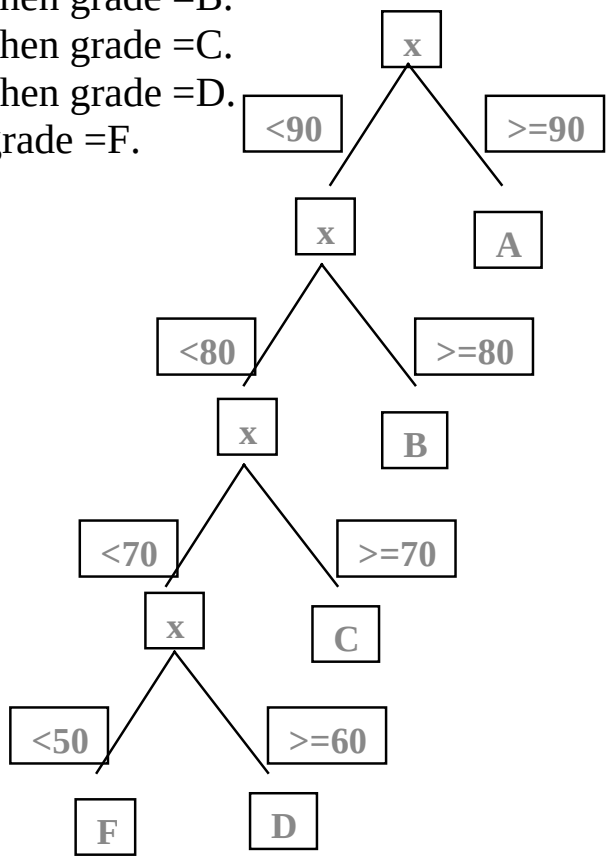
3. Common Test Corpora

- Reuters - Collection of newswire stories from 1987 to 1991, labeled with categories.
- TREC-AP newswire stories from 1988 to 1990, labeled with categories.
- OHSUMED Medline articles from 1987 to 1991, MeSH categories assigned.
- UseNet newsgroups.
- WebKB - Web pages gathered from university CS departments.

4. Classification

- A classical problem extensively studied by statisticians and machine learning researchers
- Predicts categorical class labels
- Example: Typical Applications
  - {credit history, salary} → credit approval (Yes/No)
  - {Temp, Humidity} → Rain (Yes/No)
  - A set of documents → sports, technology, etc.

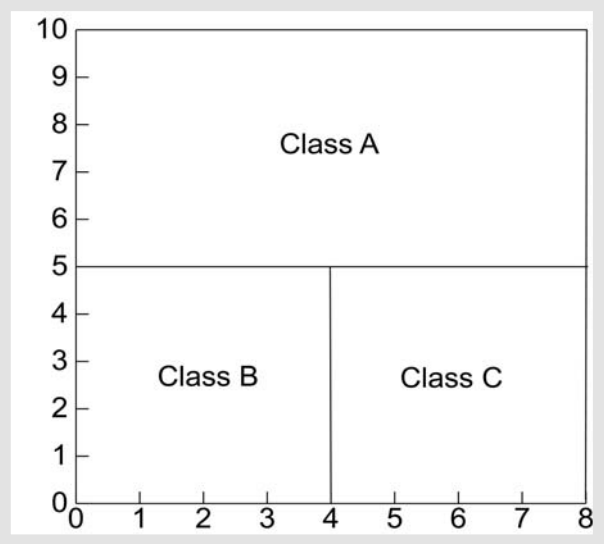
- Another Example:
  - If x >= 90 then grade =A.
  - If 80<=x<90 then grade =B.
  - If 70<=x<80 then grade =C.
  - If 60<=x<70 then grade =D.
  - If x<50 then grade =F.



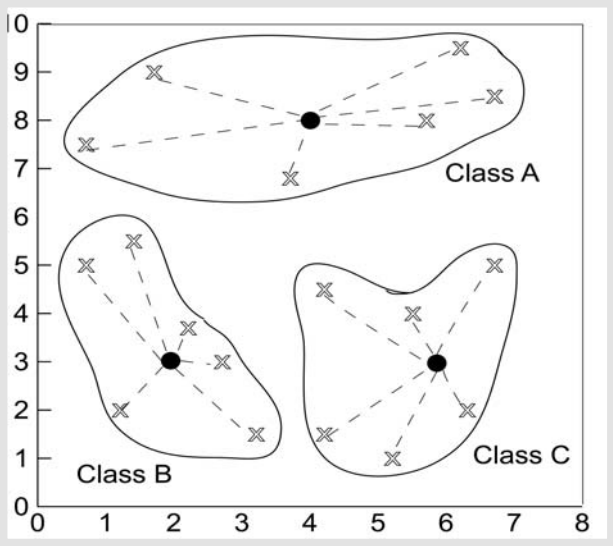
- Classification types:

- Distance based
- Partitioning based

Partitioning Based

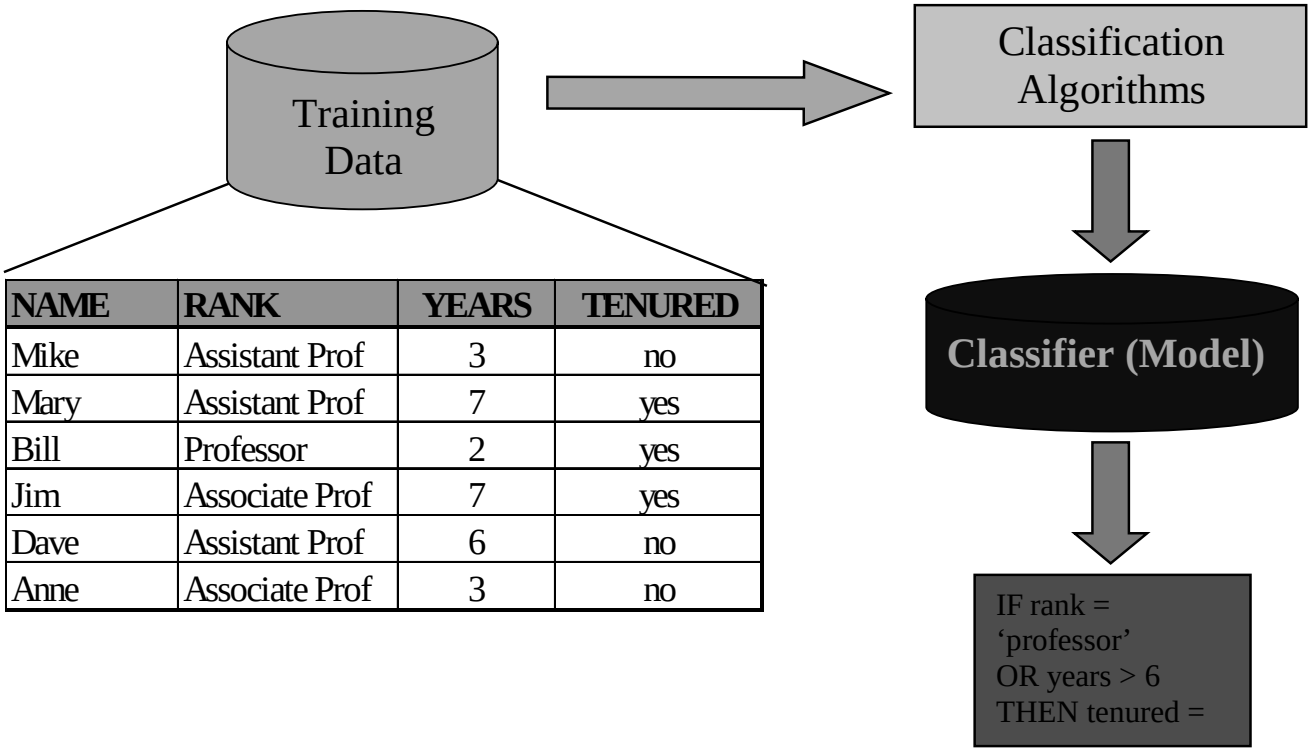


Distance Based



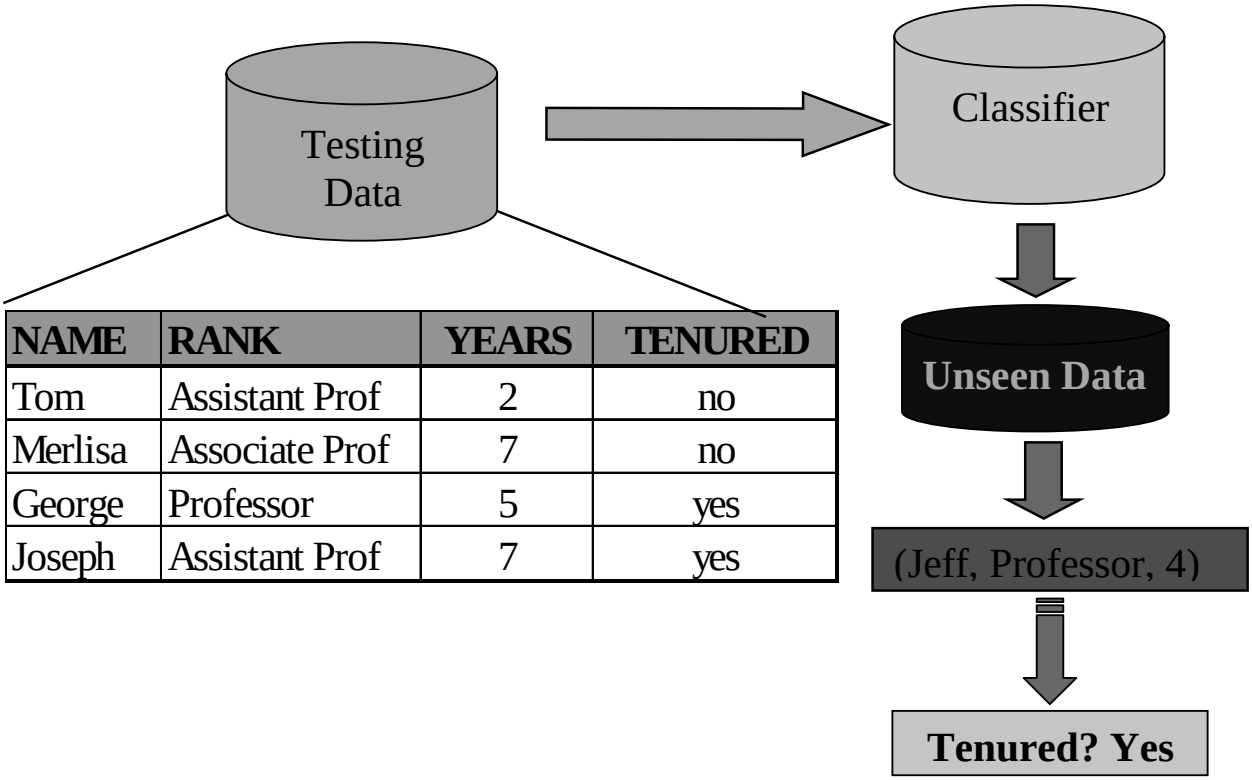
- Classification as a two-step process:
  - **Model construction**: Build a model for pre-determined classes.
  - **Model usage**: Classify unknown data samples
  - If the accuracy is acceptable, use the model to classify data objects whose class labels are not known

- **Model construction**: describing a set of predetermined classes
  - Each data sample is assumed to belong to a predefined class, as determined by the **class label attribute**
  - Use a **training dataset** for model construction.
  - The model is represented as classification rules, decision trees, or mathematical formula



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

- **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
    - Test accuracy rate is the percentage of test set samples that are correctly classified by the model
    - Test set is independent of training set but from the same probability distribution



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

- Common Techniques

  - K-Nearest Neighbor (kNN) - Use k closest training data samples to predict category.
  - Decision Trees (DTree)- Construct classification trees based on training data.

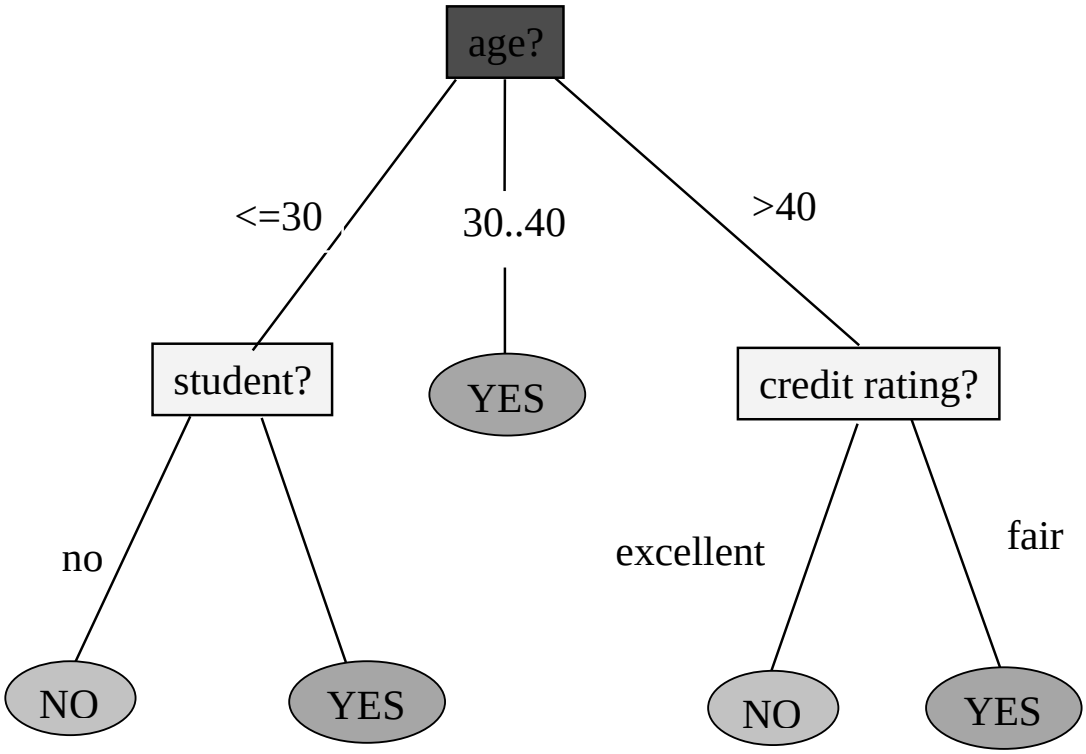
- Neural Networks (NNet) - Learn non-linear mapping from input data samples to categories.
- Support Vector Machines (SVMs).

5. Decision Tree Induction

- Example: Training Dataset [Quinlan's ID3]

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”



5.1. Decision Tree Induction Algorithm

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - Samples are partitioned recursively based on selected attributes
  - **Test attributes are selected** on the basis of a heuristic or statistical measure (e.g., information gain)
  - 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

- Attribute Selection Measure: Information gain

  - Select the attribute with the highest information gain
  - S contains s_i tuples of class C_i for i = {1, …, m}
  - Entropy of the set of tuples:
    - It measures how informative is a node.

$$I(s_1, s_2, ..., s_m) = -\sum_{i=1}^{m} \frac{s_i}{s} \log_2 \frac{s_i}{s}$$

- Entropy after choosing attribute A with values {a1,a2,…,av}

$$E(A) = \sum_{j=1}^{v} \frac{s_{1j} + ... + s_{mj}}{s} I(s_{1j},...,s_{mj})$$

- Information gained by branching on attribute A

$$Gain(A) = I(s_1, s_2,..., s_m) - E(A)$$

- Information Gain Computation
  - Class P:
    - buys_computer = “yes”
    - p: number of samples
  - Class N:
    - buys_computer = “no”
    - n: number of samples
  - The expected information:

    I(p, n) = I(9, 5) =0.940
  - Compute the entropy for *age*:

age	pi	ni	I(pi, ni)
<=30	2	3	0.971
30…40	4	0	0
>40	3	2	0.971

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

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

Hence

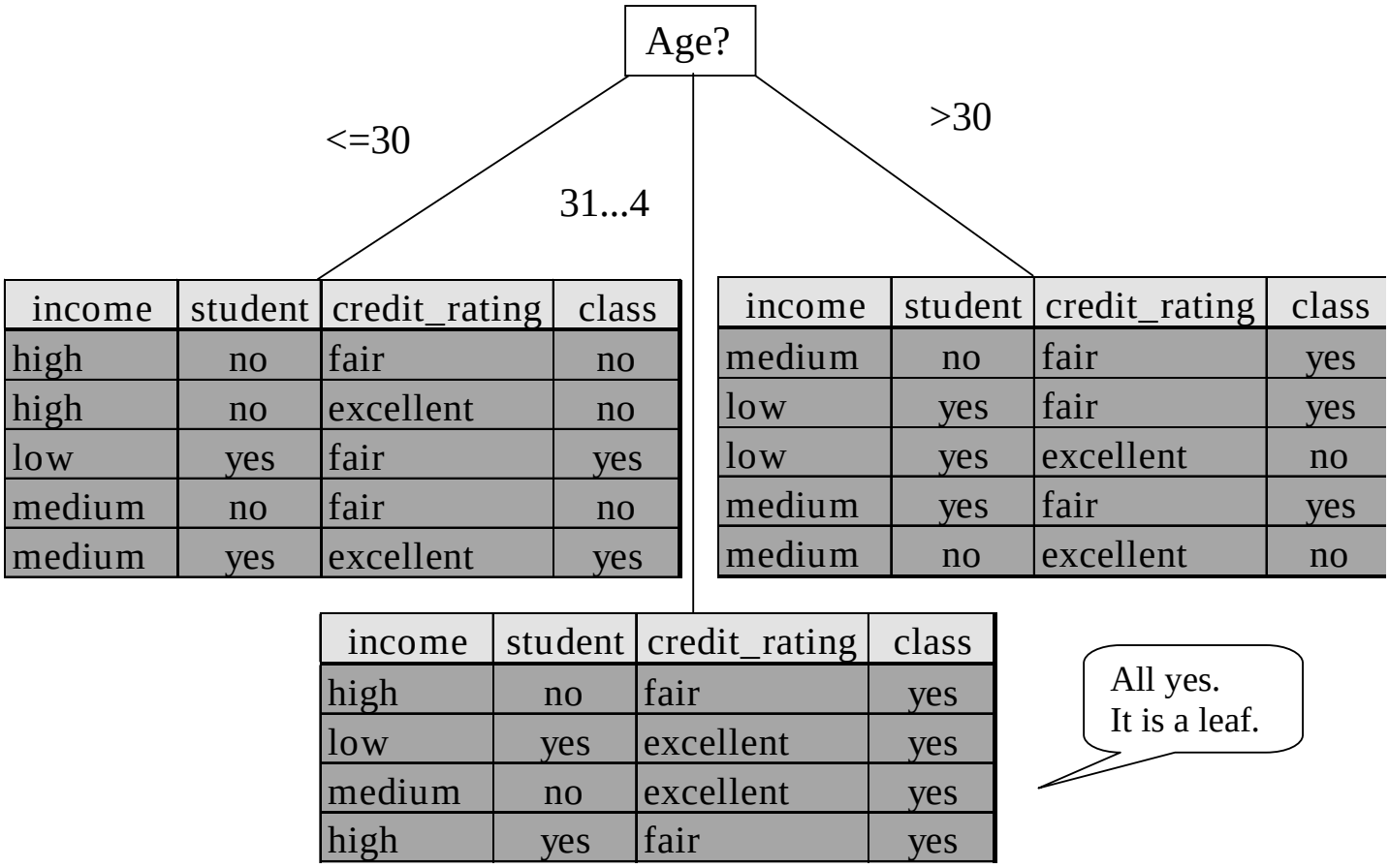
Gain(age) = I(p,n)-E(age) = 0.94- 0.694 = 0.246

Similarly,

Gain(income) = 0.029
Gain(student) = 0.151
Gain(credit_rating) = 0.048

We say that age is ***more informative*** than income, student, and credit_rating.

So age would be chosen as the root of the tree



income	student	credit_rating	class
high	no	fair	no
high	no	excellent	no
low	yes	fair	yes
medium	no	fair	no
medium	yes	excellent	yes

income	student	credit_rating	class
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

income	student	credit_rating	class
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

- Recursively apply the same process to each subset.

- ID3 Algorithm:

```
function ID3 (R: a set of non-categorical attributes,
              C: the categorical attribute,
              S: a training set) returns a decision tree;
begin
      If S is empty, return a single node with value Failure;
      If S consists of records all with the same value for
         the categorical attribute,
         return a single node with that value;
      If R is empty, then return a single node with as value
         the most frequent of the values of the categorical attribute
         that are found in records of S; [note that then there
         will be errors, that is, records that will be improperly
         classified];
      Let D be the attribute with largest Gain(D,S)
         among attributes in R;
      Let {dj| j=1,2, .., m} be the values of attribute D;
      Let {Sj| j=1,2, .., m} be the subsets of S consisting
         respectively of records with value dj for attribute D;
      Return a tree with root labeled D and arcs labeled
         d1, d2, .., dm going respectively to the trees

           ID3(R-{D}, C, S1), ID3(R-{D}, C, S2), .., ID3(R-{D}, C, Sm);
end ID3;
```

5.2. Other Attribute Selection Measures

- Gini Index (CART, IBM IntelligentMiner)
  - All attributes are assumed continuous-valued
  - Assume there exist several possible split values for each attribute
  - May need other tools, such as clustering, to get the possible split values
  - Can be modified for categorical attributes

- Formal Definition
  - If a data set *T* contains examples from *n* classes, gini index, *gini*(*T*) is defined as

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

    Where p_j is the relative frequency of class *j* in *T*.

  - If a data set *T* is split into two subsets T_1 and T_2 with sizes N_1 and N_2 respectively, the *gini* index of the split data contains examples from *n* classes, the *gini* index *gini*(*T*) is defined as:

$$gini_{split}(T) = \frac{N_1}{N} gini(T_1) + \frac{N_2}{N} gini(T_2)$$

  - The attribute provides the smallest $gini_{split}(T)$ is chosen to split the node (*need to enumerate all possible splitting points for each attribute*).

5.3. Extracting Classification Rules from Trees:

- Represent the knowledge in the form of IF-THEN rules
- 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 node holds the class prediction
- Rules are easier for humans to understand
- Example
  - IF *age* = "<=30" AND *student* = "*no*" THEN *buys_computer* = "*no*"
  - IF *age* = "<=30" AND *student* = "*yes*" THEN *buys_computer* = "*yes*"
  - IF *age* = "31…40" THEN *buys_computer* = "*yes*"
  - IF *age* = ">40" AND *credit_rating* = "*excellent*" THEN *buys_computer* = "*yes*"
  - IF *age* = "<=30" AND *credit_rating* = "*fair*" THEN *buys_computer* = "*no*"

5.4. Avoid Overfitting in Classification

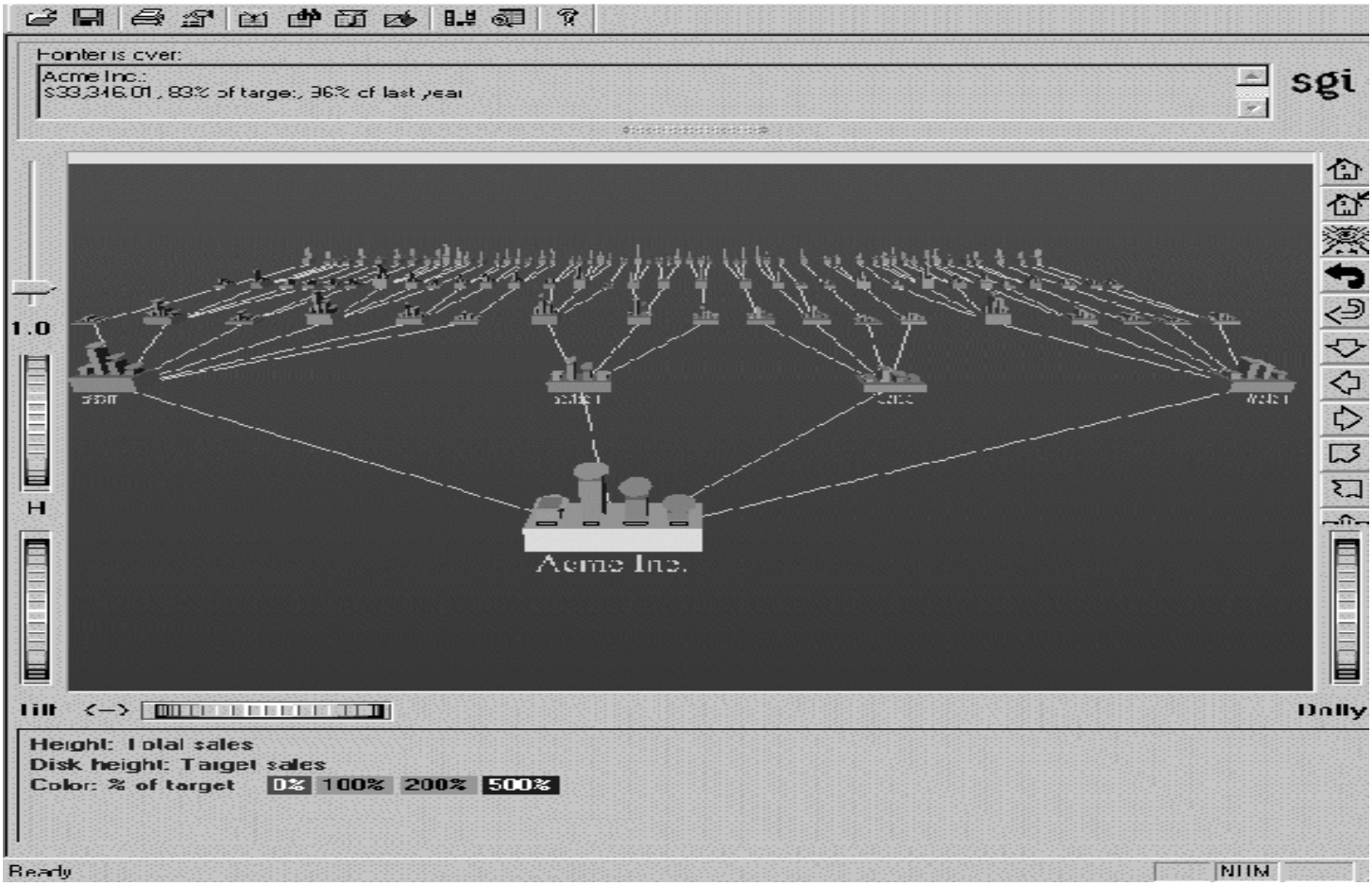
- Overfitting: An induced tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples
- Two approaches to avoid overfitting
  - Prepruning: Halt tree construction early—do not split a node if this would result in the goodness measure falling below a threshold
    - Difficult to choose an appropriate threshold
  - Postpruning: Remove branches from a "fully grown" tree—get a sequence of progressively pruned trees

- Use a set of data different from the training data to decide which is the “best pruned tree”

5.5. Classification in Large Databases

- Classification—a classical problem extensively studied by statisticians and machine learning researchers
- Scalability: Classifying data sets with millions of examples and hundreds of attributes with reasonable speed
- Why decision tree induction in data mining?
  - 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

- Visualization of a Decision Tree in SGI/MineSet 3.0



6. Bayesian Classification

- Why Bayesian?
  - Probabilistic learning: Calculate explicit probabilities for hypothesis, among the most practical approaches to certain types of learning problems

  - Incremental: Each training example can incrementally increase/decrease the probability that a hypothesis is correct. Prior knowledge can be combined with observed data.

  - Probabilistic prediction: Predict multiple hypotheses, weighted by their probabilities

  - Standard: Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured

6.1. Basics

- Let X be a data sample whose class label is unknown
- Let H be a hypothesis that X belongs to class C
- Posterior Probability:
  - For classification problems, determine P(H/X): the probability that the hypothesis H holds given the observed data sample X

- Prior Probability:
  - P(H): prior probability of hypothesis H
  - It is the initial probability before we observe any data

- It reflects the background knowledge

- P(X): probability that sample data is observed
  - P(X|H) : probability of observing the sample X, given that the hypothesis H holds

- Bayesian Theorem:

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

$$P(H \mid X) = \frac{P(X \mid H)P(H)}{P(X)}$$

- Informally, this can be written as

$$\text{posterior} = \frac{likelihood \ \ x \ \ prior}{evidence}$$

- Maximum Posteriori (MAP) Hypothesis:

  Since P(X) is constant of all hypotheses, we have the following:

$$h_{MAP} \equiv \arg\max_{h \in H} P(h \mid D) = \arg\max_{h \in H} P(D \mid h)P(h).$$

  Where D is the data sample, H is the set of all available hypothesis (e.g., all available classes).

- Practical difficulty:
  - Require initial knowledge of many probabilities
  - Significant computational cost

6.2. Naïve Bayesian Classifier

- Algorithm:
  - A simplified assumption: attributes are conditionally independent:

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

  - The product of occurrence of say 2 elements x_1 and x_2, given the current class is C, is the product of the probabilities of each element taken separately, given the same class $P([y_1,y_2],C) = P(y_1,C) * P(y_2,C)$

  - No dependence relation between attributes

  - Greatly reduces the computation cost, only count the class distribution.

  - Once the probability $P(X|C_i)$ is known, assign X to the class with maximum $P(X|C_i)*P(C_i)$

- Example:

  - Given the following data sample from our previous example:

X=(age<=30,Income=medium,Student=yes,Credit_rating=Fair)

  - Compute $P(x_k|C_i)$:

$$P(x_k|C_i)=\frac{s_{ik}}{s_i}$$

Where s_{ik} is the number of training samples of Class C_i having the value x_k for the attribute A_k and s_i is the number of training samples belonging to C_i

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)

- **Compute P(X|Ci) :**

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

P(X|buys_computer="yes")= 0.222 x 0.444 x 0.667 x 0.0.667 =0.044

P(X|buys_computer="no")= 0.6 x 0.4 x 0.2 x 0.4 =0.019

- **$P(X|C_i)*P(C_i)$:**

$P(C_i)$ = the class prior probability

$$= \frac{s_i}{s}$$

Where s_i is the number of training samples in C_i and s is the total number of training samples.

P(buys_computer="yes")= $\frac{9}{14}$ =0.643

P(buys_computer="no")= $\frac{5}{14}$ =0.36

P(X|buys_computer="yes") * P(buys_computer="yes")=0.028
P(X|buys_computer="no") * P(buys_computer="no")=0.007

X belongs to class "buys_computer=yes"

- Advantages:
  - Easy to implement
  - Good results obtained in most of the cases

- Disadvantages
  - 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 Bayesian Classifier

- How to deal with these dependencies?
  - Bayesian Belief Networks

7. Bayesian Belief Networks

- Objectives:
  - The naïve Bayesian classifier assume that attributes are conditionally independents

  - Belief Nets *are PROVEN TECHNOLOGY*
    - Medical Diagnosis
    - DSS for complex machines
    - Forecasting, Modeling, Information Retrieval, etc.

7.1. Definition

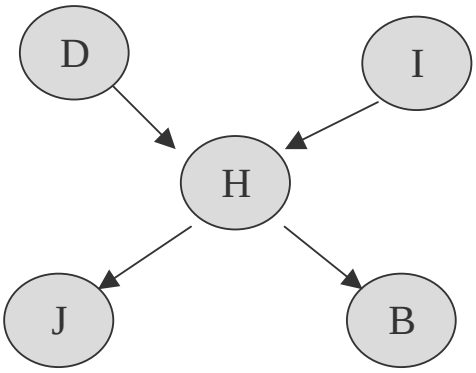
- A bayesian network is a causal directed acyclic graph (DAG), associated with an underlying distribution of probability.
- DAG structure
  - Each node is represented by a variable *v*
  - *v* depends (only) on its parents conditional probability:

$$P(v_i \mid \text{parent}_i = <0,1,\ldots>)$$

    v is INDEPENDENT of non-descendants, given assignments to its parents

- We must specify the conditional probability distribution for each node.

- It the variables are discrete, each node is described by a table (Conditional Probability Table (CPT)) which lists the probability that the child node takes on each of its different values for each combination of values of its parents.
- Example:



Given H = 1,

- D has no influence on J
- J has no influence on B
- etc.

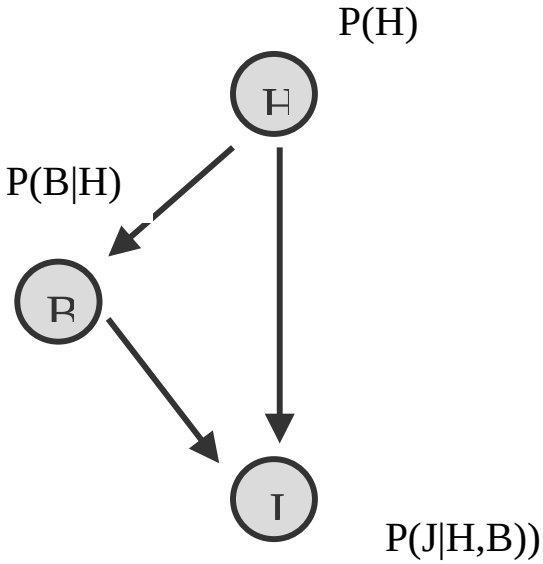
- Example:

Simple Belief Net:

P(H=1)	P(H=0)
0.05	0.95

h	P(B=1 | H=h)	P(B=0 | H=h)
1	0.95	0.05
0	0.03	0.97

h	b	P(J=1|h,b)	P(J=0|h,b)
1	1	0.8	0.2
1	0	0.8	0.2
0	1	0.3	0.7
0	0	0.3	0.7



- Challenges
  - Efficient ways to use BNs
  - Ways to create BNs
  - Ways to maintain BNs
  - Reason about time

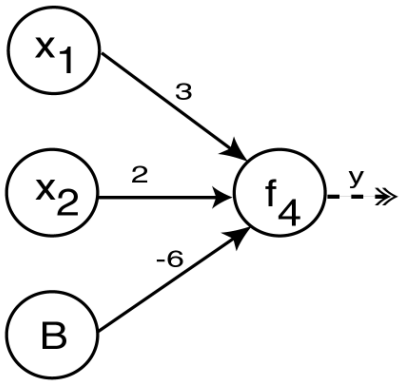
8. Neural Networks: Classification by Backpropagation

- A neural network is a set of connected input/output units where each connection has a weight associated with it.

- During the learning process, the network learns by adjusting the weights to predict the correct class label of the input samples.

- Typical NN structure for classification:
  - One output node per class
  - Output value is class membership function value

- Perceptron
  - It is one of the simplest NN
  - No hidden layers.

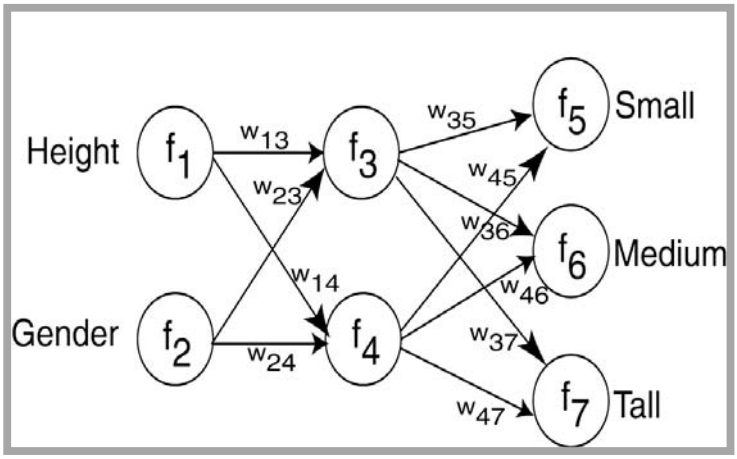


- Supervised learning

- For each tuple in training set, propagate it through NN. Adjust weights on edges to improve future classification.

- Algorithms: Propagation, Backpropagation, Gradient Descent

- Example:



8.1. Neural network Issues

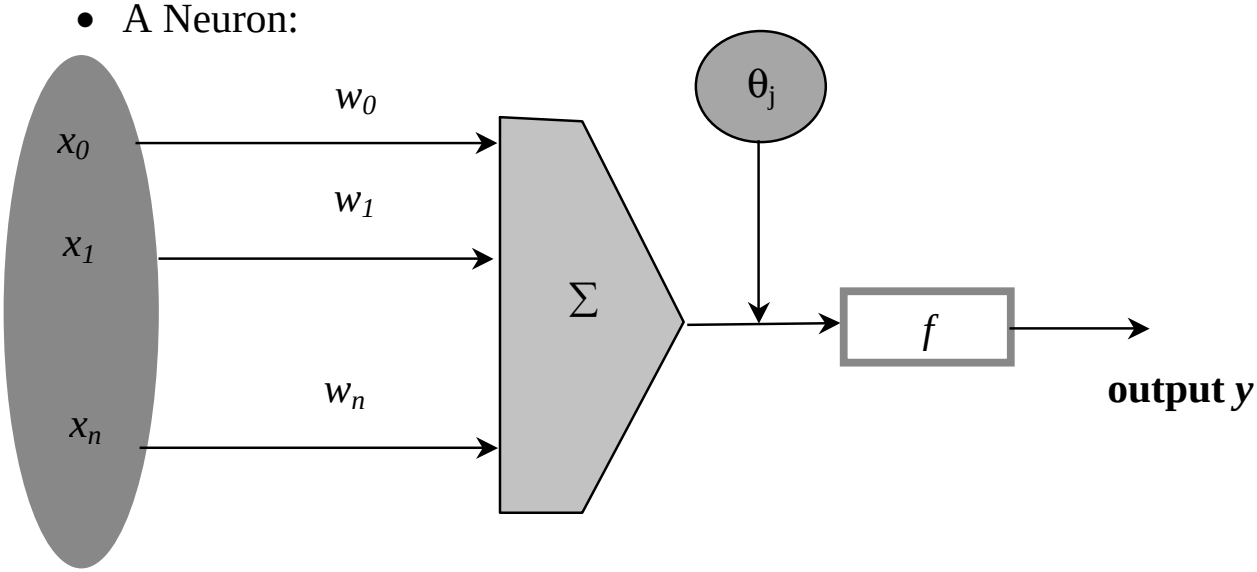
- Number of source nodes
- Number of hidden layers
- Training data
- Number of sinks
- Interconnections
- Weights
- Activation Functions
- Learning Technique
- When to stop learning

8.2. Backpropagation Algorithm

- Backpropagation learns by iteratively processing a set of training samples, comparing the network's prediction for each sample with the actual known class label.

- For each training sample, the weights are modified so as to minimize the mean squared error between the network prediction and the actual class ➔ modifications are made in the "backward" directions.

- A Neuron:



Input vector *x* **weight vector *w*** **weighted sum** **Activation function**

- The *n*-dimensional input vector *x* (A training sample) is mapped into variable *y* by means of the scalar product and a nonlinear function mapping
- For example:

$$y = \sum_{i=0}^{n} w_i x_i + \theta_k$$

- Algorithm:
  - Initialize the weights in the NN and the bias associated to each neuron. The value are generally chosen between –1 and 1 or –5 and +5.
  - For each sample, X, do: Propagate the inputs forward
    - The net input and output of each neuron j in the hidden and output layers are computed:

$$I_j = \sum_i w_{ij} O_i + \theta_j$$

Where

w_{ij} is the weight of the connection from unit ***i*** in the previous layer to unit j

O_i is the output of unit I from the previous layer.

θ_j is the bias of the unit: this is used to vary the activity of the unit.

    - Apply a non-linear logistic or sigmoid function to compute the output of neuron j:

$$O_j = \frac{1}{1 + e^{-I_j}}$$

This is called a squashing function: map Ij to a range of values between 0 and 1.

  - Backpropagate the error: Update the weights and bias to reflect the network's prediction:
    - Output layer:

$$Err_j = O_j(1 - O_j)(T_j - O_j)$$

Where Tj is the true output and Oj is the actual output of neuron j

- Hidden layer: It uses the weighted sum of the errors of the neurons connected neuron j in the next layer. The error of a neuron j is:

$$Err_j = O_j(1-O_j)\sum_k Err_k w_{jk}$$

Where w_{jk} is the weight of the connection of neuron j to a neuron k in the next layer, and Err_k is the error of neuron k

- Adjust the weights and Biases: To reflect the propagated errors

$$\theta_j = \theta_j + (l)Err_j \qquad w_{ij} = w_{ij} + (l)Err_j O_i$$

Where l is a constant learning rate between 0 and 1

- Terminating conditions: Generally after 100s of Ks of iterations or epochs:

  - All Δw_{ij} in the previous epoch (one iteration through the training set) are so small or below a specified threshold.
  - The percentage of samples misclassified in the previous epoch is below some threshold
  - A prespecified number of epochs has expired.

- Do example in the textbook: Page 308

9. Prediction

- Prediction (often called regression) is similar to classification
  - First, construct a model
  - Second, use model to predict unknown value
- Major method for prediction is regression
  - Linear and multiple regression
  - Non-linear regression
- Prediction is different from classification
  - Classification refers to predict categorical class label
  - Prediction models continuous-valued functions
- Many classification methods can also be used for regressions
  - Decision trees can also produce probabilities
  - Bayesian networks: probability
  - Neural networks: continuous output

9.1. **Regress Analysis and Log-Linear Models in Prediction**

- Linear regression: $Y = a + b X$
  - Two parameters, a and b specify the line and are to be estimated by using the data at hand.
  - Using the least squares criterion to the known values of Y1, Y2, …, X1, X2, …. (See example 7.6 on page 320 of your textbook).

- Multiple regression: $Y = b0 + b1 X1 + b2 X2$.
  - Many nonlinear functions can be transformed into the above.

- Non-linear models:
  - Can always be converted to a linear model

10. Classification Accuracy: Estimating Error Rates

- Partition: Training-and-testing
  - Use two independent data sets, e.g., training set (2/3), test set(1/3)
  - Used for data set with large number of samples

- Cross-validation
  - Divide the data set into *k* subsamples
  - Use *k-1* subsamples as training data and one sub-sample as test data—*k*-fold cross-validation
  - For data set with moderate size

- Bootstrapping (leave-one-out)
  - For small size data

- Confusion Matrix:
  - This matrix shows not only how well the classifier predicts different classes
  - It describes information about actual and detected classes:

		Detected	
		Positive	**Negative**
Actual	**Positive**	A: True positive	B: False Negative
	Negative	C: False Positive	D: True Negative

- The recall (or the true positive rate) and the precision (or the positive predictive rate) can be derived from the confusion matrix as follows:
  - Recall = $\frac{A}{A+B}$
  - Precision = $\frac{A}{A+C}$