PURDUE
UNIVERSITY

CS590D: Data Mining

Prof. Chris Clifton

February 1, 2005

Classification

Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Bayesian Classification
- Instance Based Methods
- Classification by decision tree induction
- Classification by Neural Networks
- Classification by Support Vector Machines (SVM)

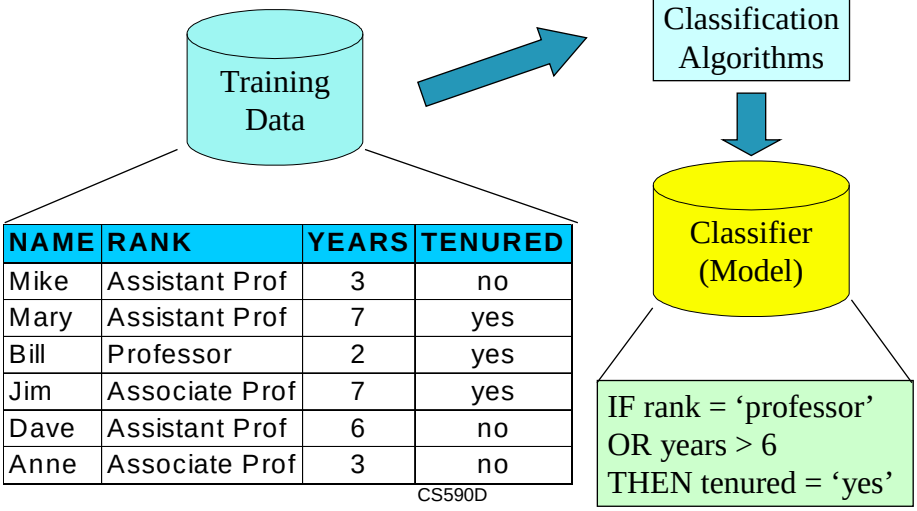
Classification vs. Prediction

- 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
  - credit approval
  - target marketing
  - medical diagnosis
  - treatment effectiveness analysis

Classification—A Two-Step Process

- Model construction: describing a set of predetermined classes
  - Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
  - 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
    - 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

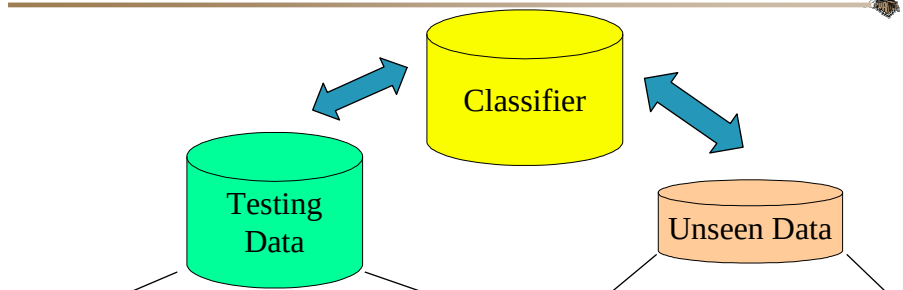
Classification Process (1): Model Construction



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

CS590D

Classification Process (2): Use the Model in Prediction



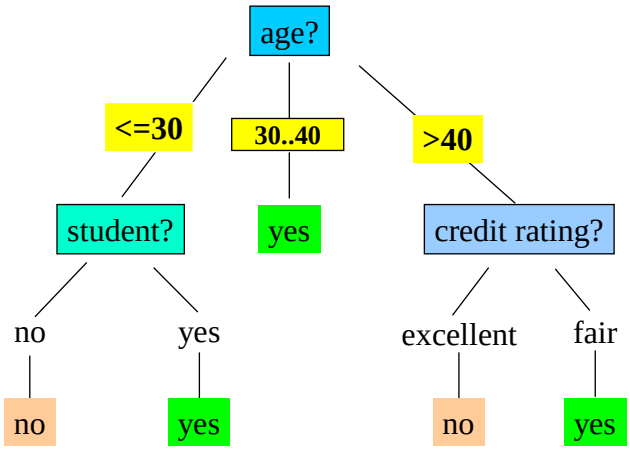
NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

CS590D 6

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	
31…40	high	yes	fair	
>40	medium	no	excellent	

A Decision Tree for *"buys_computer"*



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

Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Bayesian Classification
- Instance Based Methods
- Classification by decision tree induction
- Classification by Neural Networks
- Classification by Support Vector Machines (SVM)

Issues (1): 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

Issues (2): Evaluating Classification Methods

- Predictive accuracy
- Speed and scalability
  - time to construct the model
  - time to use the model
- Robustness
  - 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

Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Bayesian Classification
- Instance Based Methods
- Classification by decision tree induction
- Classification by Neural Networks
- Classification by Support Vector Machines (SVM)

Bayesian Classification: Why?

- 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

Bayes' Theorem: Basics

- Let X be a data sample whose class label is unknown
- Let H be a hypothesis that X belongs to class C
- For classification problems, determine P(H|X): the probability that the hypothesis holds given the observed data sample X
- P(H): prior probability of hypothesis H (i.e. the initial probability before we observe any data, 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 holds

Bayes' 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

  posterior =likelihood x prior / evidence

- MAP (maximum posteriori) hypothesis

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

- Practical difficulty: require initial knowledge of many probabilities, significant computational cost

Naïve Bayes Classifier

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

Training dataset

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

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

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
30…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

- Compute P(X/Ci) 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|Ci) : 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|Ci)*P(Ci) : P(X|buys_computer="yes") * P(buys_computer="yes")=0.028
  P(X|buys_computer="yes") * P(buys_computer="yes")=0.007

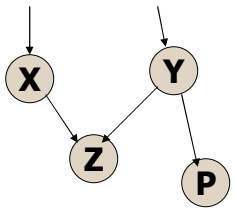
  X belongs to class "buys_computer=yes"

Naïve Bayes Classifier: Comments

- 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

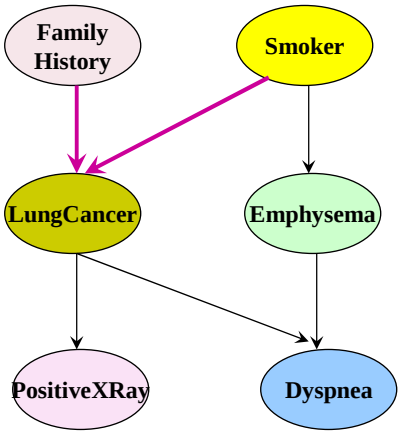
Bayesian Networks

- Bayesian belief network allows a *subset* of the variables conditionally independent
- A graphical model of causal relationships
  - Represents dependency among the variables
  - Gives a specification of joint probability distribution



•Nodes: random variables
•Links: dependency
•X,Y are the parents of Z, and Y is the parent of P
•No dependency between Z and P
•Has no loops or cycles

Bayesian Belief Network: An Example



Bayesian Belief Networks

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

The conditional probability table for the variable LungCancer: Shows the conditional probability for each possible combination of its parents

$$P(z1,\ldots,zn) = \prod_{i=1}^{n} P(z_i \mid Parents(Z_i))$$

Learning Bayesian Networks

- Several cases
  - Given both the network structure and all variables observable: learn only the CPTs
  - Network structure known, some hidden variables: method of gradient descent, analogous to neural network learning
  - Network structure unknown, all variables observable: search through the model space to reconstruct graph topology
  - Unknown structure, all hidden variables: no good algorithms known for this purpose
- D. Heckerman, Bayesian networks for data mining

Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Bayesian Classification
- Instance Based Methods
- Classification by decision tree induction
- Classification by Neural Networks
- Classification by Support Vector Machines (SVM)

Other Classification Methods

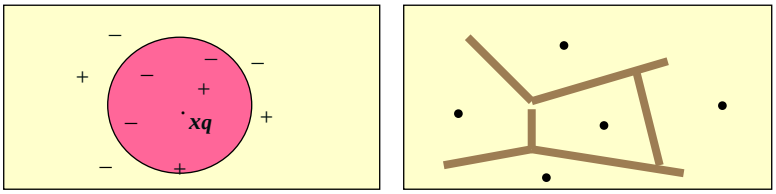
- k-nearest neighbor classifier
- case-based reasoning
- Genetic algorithm
- Rough set approach
- Fuzzy set approaches

Instance-Based Methods

- Instance-based learning:
  - Store training examples and delay the processing ("lazy evaluation") until a new instance must be classified
- Typical approaches
  - *k*-nearest neighbor approach
    - Instances represented as points in a Euclidean space.
  - Locally weighted regression
    - Constructs local approximation
  - Case-based reasoning
    - Uses symbolic representations and knowledge-based inference

The *k*-Nearest Neighbor Algorithm

- All instances correspond to points in the n-D space.
- The nearest neighbor are defined in terms of Euclidean distance.
- The target function could be discrete- or real- valued.
- For discrete-valued, the *k*-NN returns the most common value among the k training examples nearest to x_q.
- Voronoi diagram: the decision surface induced by 1-NN for a typical set of training examples.



Discussion of the *k*-NN Algorithm

- The k-NN algorithm for continuous-valued target functions
  - Calculate the mean values of the *k* nearest neighbors
- Distance-weighted nearest neighbor algorithm
  - Weight the contribution of each of the k neighbors according to their distance to the query point x_q
    - giving greater weight to closer neighbors $w \equiv \frac{1}{d(x_q, x_i)^2}$
  - Similarly, for real-valued target functions
- Robust to noisy data by averaging k-nearest neighbors
- Curse of dimensionality: distance between neighbors could be dominated by irrelevant attributes.
  - To overcome it, axes stretch or elimination of the least relevant attributes.

Classification and Prediction

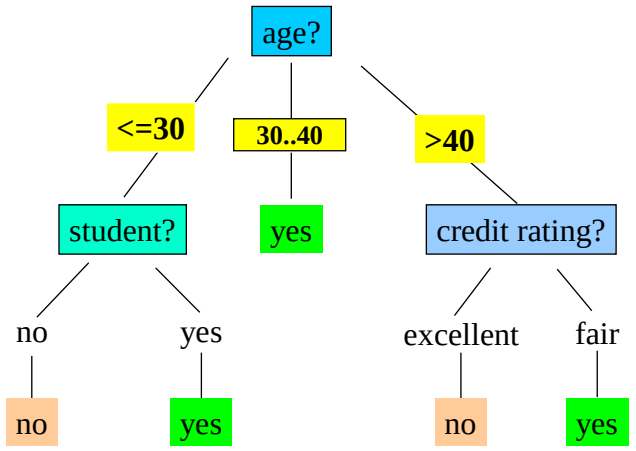
- What is classification? What is prediction?
- Issues regarding classification and prediction
- Bayesian Classification
- Instance Based Methods
- Classification by decision tree induction
- Classification by Neural Networks
- Classification by Support Vector Machines (SVM)

Training Dataset

This follows an example from 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

Output: A Decision Tree for "*buys_computer*"



Algorithm for Decision Tree Induction

- 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)
  - Examples 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 (ID3/C4.5)

- Select the attribute with the highest information gain
- S contains s_i tuples of class C_i for i = {1, …, m}
- information measures info required to classify any arbitrary tuple

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

- entropy of attribute A with values $\{a_1,a_2,\ldots,a_v\}$

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

- information gained by branching on attribute A

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

Attribute Selection by Information Gain Computation

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"
- I(p, n) = I(9, 5) =0.940
- Compute the entropy for *age*:

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
30…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

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

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

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

Similarly,

$$Gain(income) = 0.029$$
$$Gain(student) = 0.151$$
$$Gain(credit\_rating) = 0.048$$

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

Gini Index (IBM IntelligentMiner)

- 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*).

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*"

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"

Approaches to Determine the Final Tree Size

- Separate training (2/3) and testing (1/3) sets
- Use cross validation, e.g., 10-fold cross validation
- Use all the data for training
  - but apply a statistical test (e.g., chi-square) to estimate whether expanding or pruning a node may improve the entire distribution
- Use minimum description length (MDL) principle
  - halting growth of the tree when the encoding is minimized

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

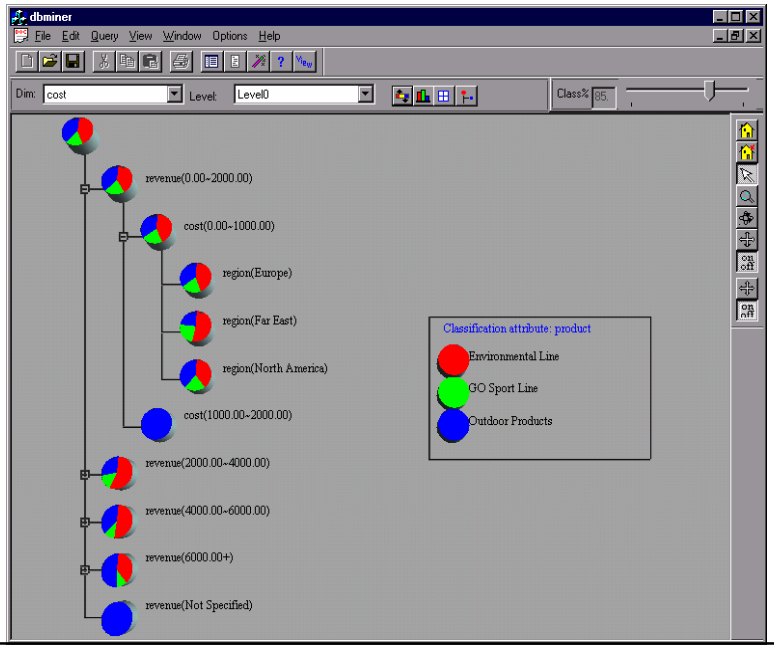
Classification in Large Databases

- 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

Scalable Decision Tree Induction Methods in Data Mining Studies

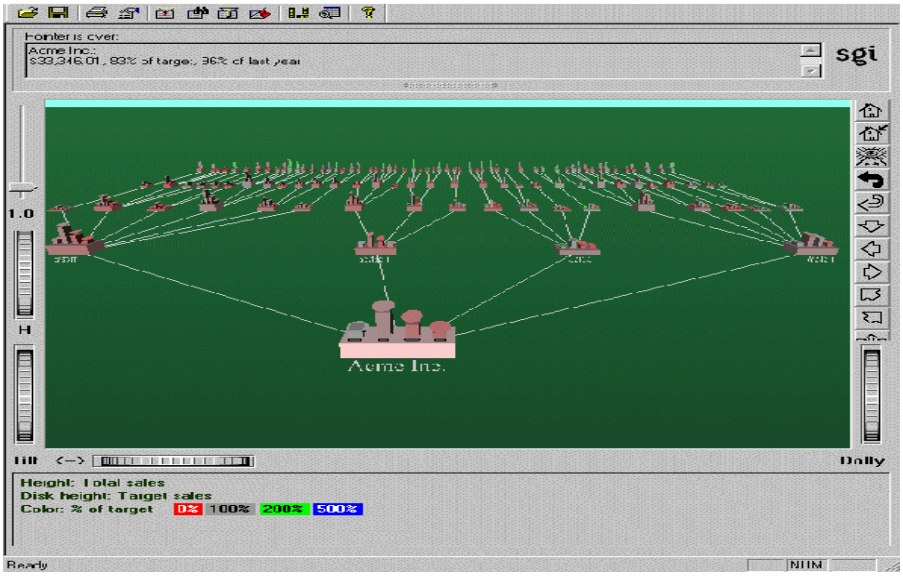
- SLIQ (EDBT'96 — Mehta et al.)
  - builds an index for each attribute and only class list and the current attribute list reside in memory
- SPRINT (VLDB'96 — J. Shafer et al.)
  - constructs an attribute list data structure
- PUBLIC (VLDB'98 — Rastogi & Shim)
  - integrates tree splitting and tree pruning: stop growing the tree earlier
- RainForest (VLDB'98 — Gehrke, Ramakrishnan & Ganti)
  - separates the scalability aspects from the criteria that determine the quality of the tree
  - builds an AVC-list (attribute, value, class label)

Presentation of Classification



47

Visualization of a Decision Tree in SGI/MineSet 3.0



Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Bayesian Classification
- Instance Based methods
- Classification by decision tree induction
- Classification by Neural Networks
- Classification by Support Vector Machines (SVM)

Classification

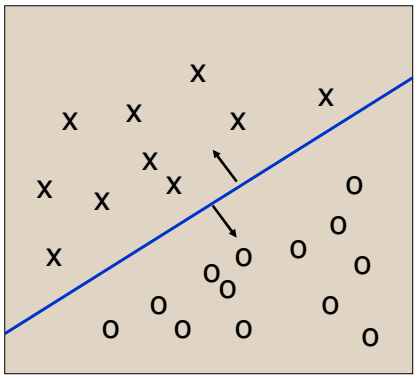
- Classification:
  - predicts categorical class labels
- Typical Applications
  - {credit history, salary}-> credit approval (Yes/No)
  - {Temp, Humidity} --> Rain (Yes/No)

$$x \in X = \{0,1\}^n, y \in Y = \{0,1\}$$

Mathematically $h: X \rightarrow Y$

$$y = h(x)$$

Linear Classification



- Binary Classification problem
- The data above the red line belongs to class 'x'
- The data below red line belongs to class 'o'
- Examples – SVM, Perceptron, Probabilistic Classifiers

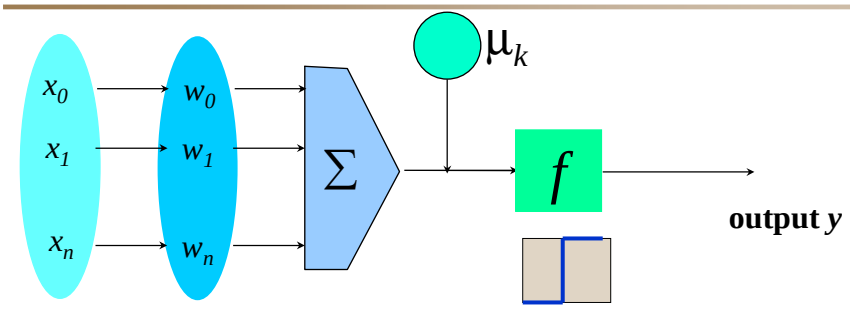
Discriminative Classifiers

- Advantages
  - prediction accuracy is generally high
    - (as compared to Bayesian methods – in general)
  - robust, works when training examples contain errors
  - fast evaluation of the learned target function
    - (Bayesian networks are normally slow)
- Criticism
  - long training time
  - difficult to understand the learned function (weights)
    - (Bayesian networks can be used easily for pattern discovery)
  - not easy to incorporate domain knowledge
    - (easy in the form of priors on the data or distributions)

Neural Networks

- Analogy to Biological Systems (Indeed a great example of a good learning system)
- Massive Parallelism allowing for computational efficiency
- The first learning algorithm came in 1959 (Rosenblatt) who suggested that if a target output value is provided for a single neuron with fixed inputs, one can incrementally change weights to learn to produce these outputs using the perceptron learning rule

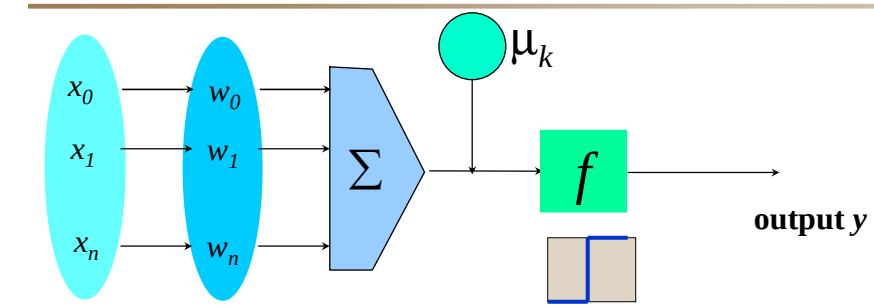
A Neuron



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

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

A Neuron

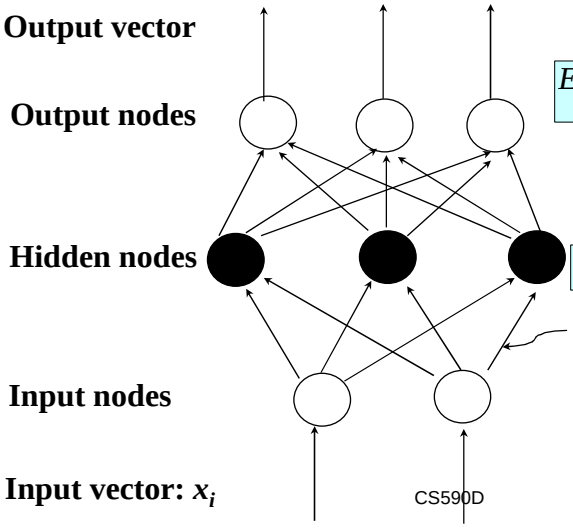


Input vector x **weight vector w** **weighted sum** **Activation function**

For Example

$$y = \text{sign}\left(\sum_{i=0}^{n} w_i x_i + \mu_k\right)$$

Multi-Layer Perceptron



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

$$\theta_j = \theta_j + (l)Err_j$$

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

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

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

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

Network Training

- The ultimate objective of training
  - obtain a set of weights that makes almost all the tuples in the training data classified correctly
- Steps
  - Initialize weights with random values
  - Feed the input tuples into the network one by one
  - For each unit
    - Compute the net input to the unit as a linear combination of all the inputs to the unit
    - Compute the output value using the activation function
    - Compute the error
    - Update the weights and the bias

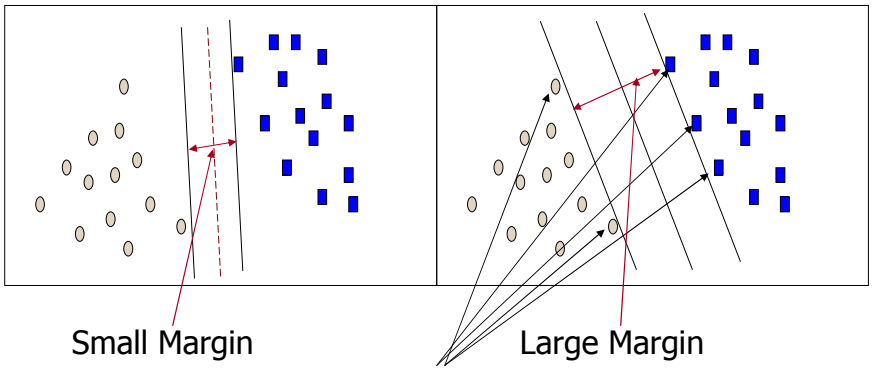
Network Pruning and Rule Extraction

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

Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Bayesian Classification
- Instance Based Methods
- Classification by decision tree induction
- Classification by Neural Networks
- Classification by Support Vector Machines (SVM)

SVM – Support Vector Machines

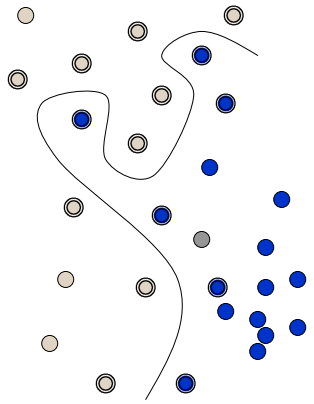


Support vector machine(SVM).

- Classification is essentially finding the best boundary between classes.
- Support vector machine finds the best boundary points called support vectors and build classifier on top of them.
- Linear and Non-linear support vector machine.

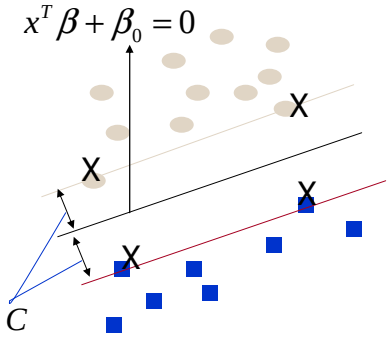
Example of general SVM

The dots with shadow around them are support vectors. Clearly they are the best data points to represent the boundary. The curve is the separating boundary.



Optimal Hyper plane, separable case.

- In this case, class 1 and class 2 are separable.
- The representing points are selected such that the margin between two classes are maximized.
- Crossed points are support vectors.



Analysis of Separable case.

1. Through out our presentation, the training data consists of N pairs:(x1,y1), (x2,y2) ,…, (Xn,Yn).
2. Define a hyper plane:

$$\{x : f(x) = x^T\beta + \beta_0 = 0\}$$

where β is a unit vector. The classification rule is :

$$G(x) = sign[x^T\beta + \beta_0]$$

Analysis Cont.

3. So the problem of finding optimal hyperplane turns to: $\beta, \beta_0, \|\beta\|=1$

   Maximizing C on

   Subject to constrain:

$$y_i(x_i^T\beta + \beta_0) > C, i = 1, \ldots, N.$$

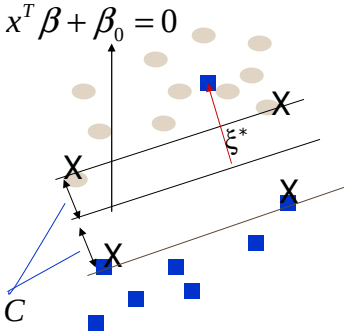
4. It's the same as :

   Minimizing $\|\beta\|$ subject to

$$y_i(x_i^T\beta + \beta_0) > 1, i = 1, \ldots, N.$$

Non-separable case

When the data set is non-separable as shown in the right figure, we will assign weight to each support vector which will be shown in the constraint.



Non-separable Cont.

1. Constraint changes to the following:

$y_i(x_i^T\beta+\beta_0),>C(1-\xi_i)$, Where

$$\forall i,\xi_i>0,\sum_{i=1}^{N}\xi_i<const.$$

2. Thus the optimization problem changes to:

$$\text{Min } \|\beta\| \text{subject to } \begin{cases} y_i(x_i^T\beta+\beta_0)>1-\xi_i, i=1,\ldots,N. \\ \forall i,\xi_i>0,\sum_{i=1}^{N}\xi_i<const. \end{cases}$$

Compute SVM.

We can rewrite the optimization problem as:

$$\min\{(1/2)\|\beta\|^2+\gamma\sum_{i=1}^{N}\xi_i\}$$

Subject to $\zeta_i>0$,

$$y_i(x_i^T\beta x_i+b)>1-\xi_i,\forall i$$

Which we can solve by Lagrange.

The separable case is when $\gamma=0$.

SVM computing Cont.

The Lagrange function for this problem is:

$$L_p = \frac{1}{2}\| \beta \|^2 + \gamma \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i [y_i (x_i^T \beta + \beta_0) - (1 - \xi_i)] - \sum_{i=1}^{N} \mu_i \xi_i$$

By formal Lagrange procedures, we get a dual problem:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} x_i^T x_{i'}$$

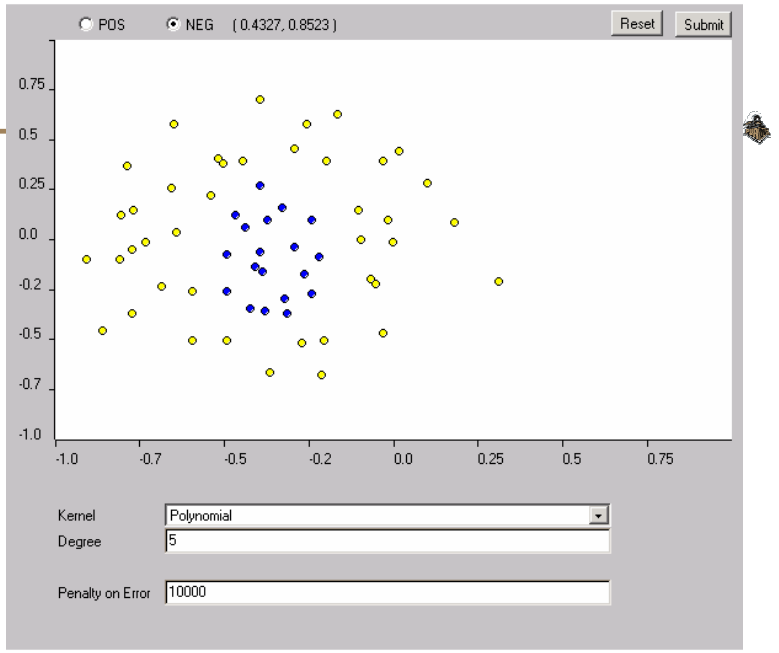
SVM computing Cont.

This dual problem subjects to the original and the K-K-T constraint. Then it turns to a simpler quadratic programming problem

The solution is in the form of:

$$\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i$$

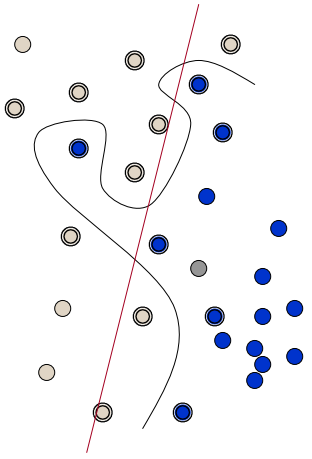
Example of Non-linear SVM



General SVM

This classification problem clearly do not have a good optimal linear classifier.

Can we do better?
A non-linear boundary as shown will do fine.



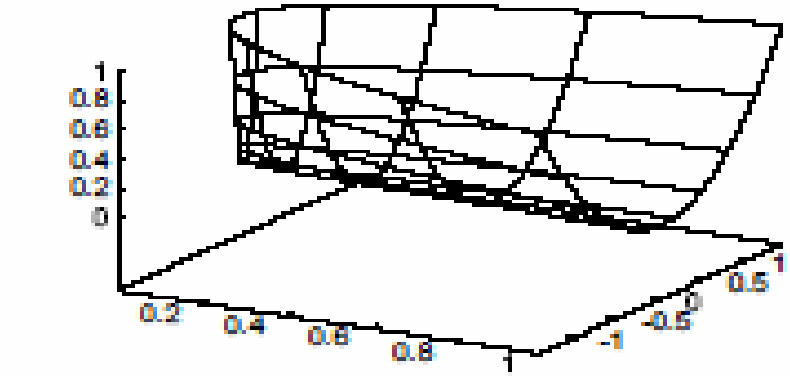
General SVM Cont.

- The idea is to map the feature space into a much bigger space so that the boundary is linear in the new space.
- Generally linear boundaries in the enlarged space achieve better training-class separation, and it translates to non-linear boundaries in the original space.

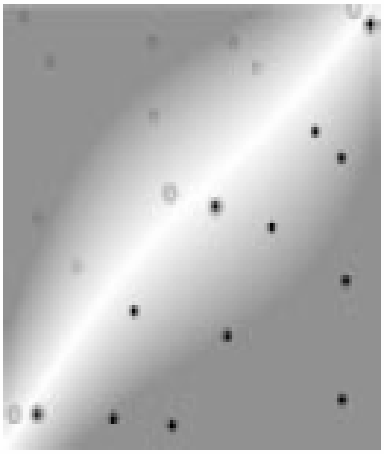
Mapping

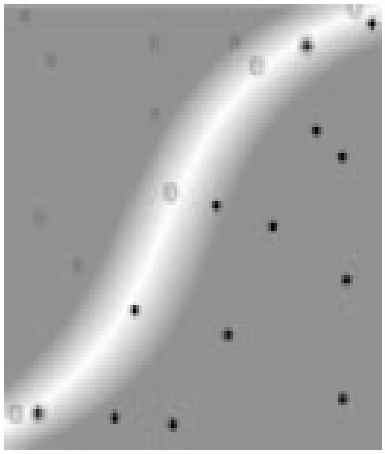
- Mapping $\Phi : \mathbb{R}^d \mapsto H$
  - Need distances in *H*: $\Phi(x_i) \cdot \Phi(x_j)$
- Kernel Function: $K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$
  - Example: $K(x_i, x_j) = e^{-\|x_i - x_j\|^2 / 2\sigma^2}$
- In this example, *H* is infinite-dimensional

Degree 3 Example



Resulting Surfaces





General SVM Cont.

Now suppose our mapping from original

Feature space to new space is $h(x_i)$, the dual problem changed to:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} \langle h(x_i), h(x_{i'}) \rangle$$

Note that the transformation only operates on the dot product.

General SVM Cont.

Similar to linear case, the solution can be written as:

$$f(x) = h(x)^T \beta + \beta_0 = \sum_{i=1}^{N} \alpha_i y_i \langle h(x_i), h(x_{i'}) \rangle + \beta_0$$

But function “h” is of very high dimension sometimes infinity, does it mean SVM is impractical?

Reproducing Kernel.

Look at the dual problem, the solution only depends on $\langle h(x_i), h(x_{i'}) \rangle$

Traditional functional analysis tells us we need to only look at their kernel representation: K(X,X')= $\langle h(x_i), h(x_{i'}) \rangle$

Which lies in a much smaller dimension Space than "h".

Restrictions and typical kernels.

- Kernel representation does not exist all the time, Mercer's condition (Courant and Hilbert,1953) tells us the condition for this kind of existence.
- There are a set of kernels proven to be effective, such as polynomial kernels and radial basis kernels.

Example of polynomial kernel.

r degree polynomial:

$K(x,x')=(1+\langle x,x'\rangle)^d$.

For a feature space with two inputs: x1,x2 and

a polynomial kernel of degree 2.

$K(x,x')=(1+\langle x,x'\rangle)^2$

Let $h_1(x)=1, h_2(x)=\sqrt{2}x_1, h_3(x)=\sqrt{2}x_2, h_4(x)=x_1^2, h_5(x)=x_2^2$

and $h_6(x)=\sqrt{2}x_1x_2$, then $K(x,x')=\langle h(x),h(x')\rangle$.

Performance of SVM.

- For optimal hyper planes passing through the origin, we have:

$$E(P(error)) \le \frac{E[D^2/M^2]}{l}$$

- For general support vector machine.

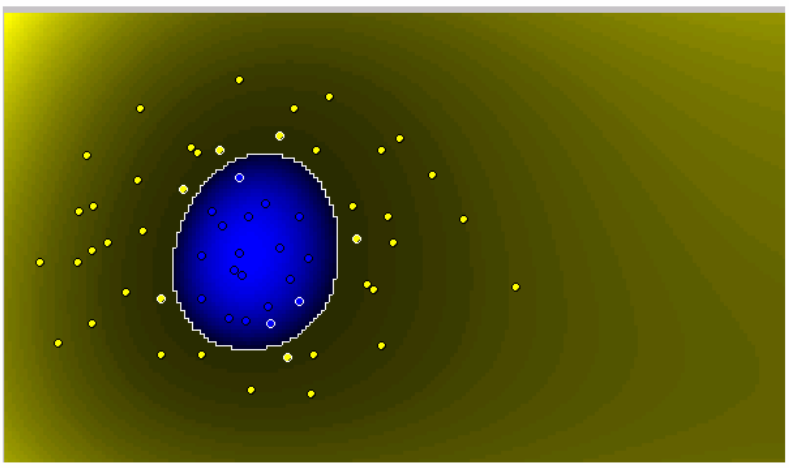
$$E[P(error)] \le$$

E(# of support vectors)/(# training samples)

- SVM has been very successful in lots of applications.

Results

Number of Support Vectors: **9** (-ve: 3, +ve: 6) Total number of points: 60



- Points (+ve, -ve)
- Support Vectors (+ve, -ve)
- Hyperplane (boundary)

SVM vs. Neural Network

- SVM
  - Relatively new concept
  - Nice Generalization properties
  - Hard to learn – learned in batch mode using quadratic programming techniques
  - Using kernels can learn very complex functions
- Neural Network
  - Quiet Old
  - Generalizes well but doesn't have strong mathematical foundation
  - Can easily be learned in incremental fashion
  - To learn complex functions – use multilayer perceptron (not that trivial)

Open problems of SVM.

- How do we choose Kernel function for a specific set of problems. Different Kernel will have different results, although generally the results are better than using hyper planes.
- Comparisons with Bayesian risk for classification problem. Minimum Bayesian risk is proven to be the best. When can SVM achieve the risk.

Open problems of SVM

- For very large training set, support vectors might be of large size. Speed thus becomes a bottleneck.
- A optimal design for multi-class SVM classifier.

SVM Related Links

- http://svm.dcs.rhbnc.ac.uk/
- http://www.kernel-machines.org/
- C. J. C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Knowledge Discovery and Data Mining*, 2(2), 1998.
- SVM^{light} – Software (in C) http://ais.gmd.de/~thorsten/svm_light
- BOOK: An Introduction to Support Vector Machines
  N. Cristianini and J. Shawe-Taylor
  Cambridge University Press

Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Bayesian Classification
- Instance-based methods
- Classification by decision tree induction
- Classification by Neural Networks
- Classification by Support Vector Machines (SVM)
- Other Classification Methods
- Prediction
- Classification accuracy
- Summary

Association-Based Classification

- Several methods for association-based classification
  - ARCS: Quantitative association mining and clustering of association rules (Lent et al'97)
    - It beats C4.5 in (mainly) scalability and also accuracy
  - Associative classification: (Liu et al'98)
    - It mines high support and high confidence rules in the form of "cond_set => y", where y is a class label
  - CAEP (Classification by aggregating emerging patterns) (Dong et al'99)
    - Emerging patterns (EPs): the itemsets whose support increases significantly from one class to another
    - Mine Eps based on minimum support and growth rate

Case-Based Reasoning

- Also uses: lazy evaluation + analyze similar instances
- Difference: Instances are not "points in a Euclidean space"
- Example: Water faucet problem in CADET (Sycara et al'92)
- Methodology
  - Instances represented by rich symbolic descriptions (e.g., function graphs)
  - Multiple retrieved cases may be combined
  - Tight coupling between case retrieval, knowledge-based reasoning, and problem solving
- Research issues
  - Indexing based on syntactic similarity measure, and when failure, backtracking, and adapting to additional cases

Remarks on Lazy vs. Eager Learning

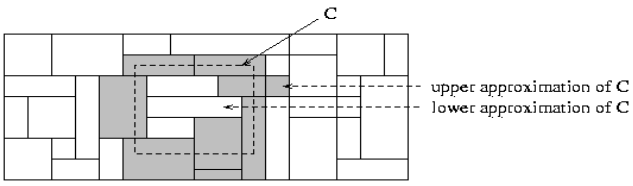
- Instance-based learning: lazy evaluation
- Decision-tree and Bayesian classification: eager evaluation
- Key differences
  - Lazy method may consider query instance *xq* when deciding how to generalize beyond the training data *D*
  - Eager method cannot since they have already chosen global approximation when seeing the query
- Efficiency: Lazy - less time training but more time predicting
- Accuracy
  - Lazy method effectively uses a richer hypothesis space since it uses many local linear functions to form its implicit global approximation to the target function
  - Eager: must commit to a single hypothesis that covers the entire instance space

Genetic Algorithms

- GA: based on an analogy to biological evolution
- Each rule is represented by a string of bits
- An initial population is created consisting of randomly generated rules
  - e.g., IF A_1 and Not A_2 then C_2 can be encoded as 100
- Based on the notion of survival of the fittest, a new population is formed to consists of the fittest rules and their offsprings
- The fitness of a rule is represented by its classification accuracy on a set of training examples
- Offsprings are generated by crossover and mutation

Rough Set Approach

- Rough sets are used to approximately or “roughly” define equivalent classes
- A rough set for a given class C is approximated by two sets: a lower approximation (certain to be in C) and an upper approximation (cannot be described as not belonging to C)
- Finding the minimal subsets (reducts) of attributes (for feature reduction) is NP-hard but a discernibility matrix is used to reduce the computation intensity



PURDUE UNIVERSITY

CS490D: Introduction to Data Mining

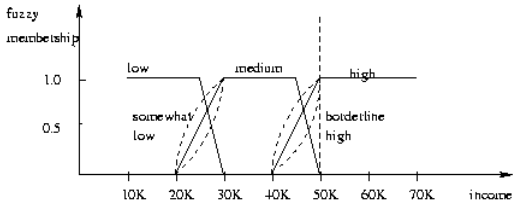
Prof. Chris Clifton

February 20, 2004

Classification

Indiana Center for Database Systems

Fuzzy Set Approaches



- Fuzzy logic uses truth values between 0.0 and 1.0 to represent the degree of membership (such as using fuzzy membership graph)
- Attribute values are converted to fuzzy values
  - e.g., income is mapped into the discrete categories {low, medium, high} with fuzzy values calculated
- For a given new sample, more than one fuzzy value may apply
- Each applicable rule contributes a vote for membership in the categories
- Typically, the truth values for each predicted category are summed

Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Bayesian Classification
- Instance Based Methods
- Classification by decision tree induction
- Classification by Neural Networks
- Classification by Support Vector Machines (SVM)

What Is Prediction?

- Prediction 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

Predictive Modeling in Databases

- Predictive modeling: Predict data values or construct generalized linear models based on the database data.
- One can only predict value ranges or category distributions
- Method outline:
  - Minimal generalization
  - Attribute relevance analysis
  - Generalized linear model construction
  - Prediction
- Determine the major factors which influence the prediction
  - Data relevance analysis: uncertainty measurement, entropy analysis, expert judgement, etc.
- Multi-level prediction: drill-down and roll-up analysis

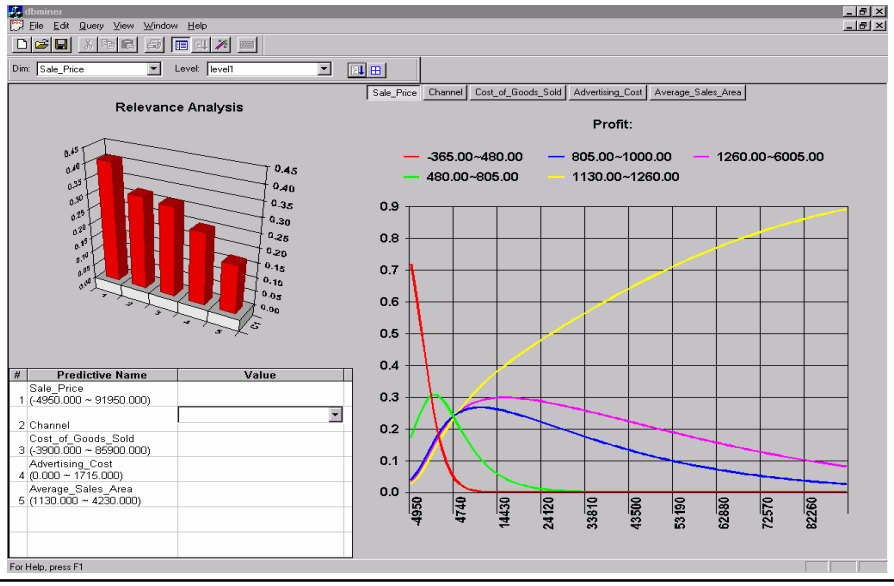
Regress Analysis and Log-Linear Models in Prediction

- Linear regression: $Y = \alpha + \beta X$
  - Two parameters , α and β specify the line and are to be estimated by using the data at hand.
  - using the least squares criterion to the known values of $Y_1, Y_2, \ldots, X_1, X_2, \ldots$.
- Multiple regression: $Y = b0 + b1\ X1 + b2\ X2$.
  - Many nonlinear functions can be transformed into the above.
- Log-linear models:
  - The multi-way table of joint probabilities is approximated by a product of lower-order tables.
  - Probability: $p(a, b, c, d) = \alpha_{ab} \beta_{ac} \chi_{ad} \delta_{bcd}$

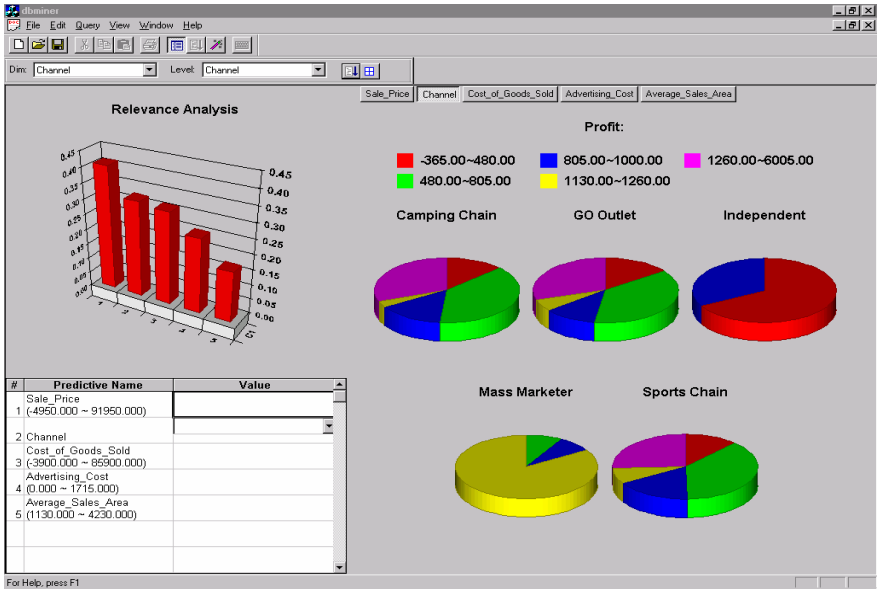
Locally Weighted Regression

- Construct an explicit approximation to *f* over a local region surrounding query instance x_q.
- Locally weighted linear regression:
  - The target function *f* is approximated near x_q using the linear function:
    $$\hat{f}(x) = w_0 + w_1 a_1(x) + \cdots + w_n a_n(x)$$
  - minimize the squared error: distance-decreasing weight *K*
    $$E(x_q) \equiv \frac{1}{2} \sum_{x \in k\_nearest\_neighbors\_of\_x_q} (f(x) - \hat{f}(x))^2 K(d(x_q, x))$$
  - the gradient descent training rule:
    $$\Delta w_j \equiv \eta \sum_{x \in k\_nearest\_neighbors\_of\_x_q} K(d(x_q, x))((f(x) - \hat{f}(x)) a_j(x)$$
- In most cases, the target function is approximated by a constant, linear, or quadratic function.

Prediction: Numerical Data



Prediction: Categorical Data



Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Bayesian Classification
- Instance Based Methods
- Classification by decision tree induction
- Classification by Neural Networks
- Classification by Support Vector Machines (SVM)

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

Bagging and Boosting

- General idea

Training data —Classification method (CM)→ **Classifier C**

Altered Training data —CM→ **Classifier C1**

Altered Training data —CM→ **Classifier C2**

……..

Aggregation …. → **Classifier C***

Bagging

- Given a set S of s samples
- Generate a bootstrap sample T from S. Cases in S may not appear in T or may appear more than once.
- Repeat this sampling procedure, getting a sequence of k independent training sets
- A corresponding sequence of classifiers C1,C2,…,Ck is constructed for each of these training sets, by using the same classification algorithm
- To classify an unknown sample X,let each classifier predict or vote
- The Bagged Classifier C* counts the votes and assigns X to the class with the “most” votes

Boosting Technique — Algorithm

- Assign every example an equal weight *1/N*
- *For t = 1, 2, …, T Do*
  - Obtain a hypothesis (classifier) $h^{(t)}$ under $w^{(t)}$
  - Calculate the error of *h(t)* and re-weight the examples based on the error . Each classifier is dependent on the previous ones. Samples that are incorrectly predicted are weighted more heavily
  - Normalize $w^{(t+1)}$ to sum to 1 (weights assigned to different classifiers sum to 1)
- Output a weighted sum of all the hypothesis, with each hypothesis weighted according to its accuracy on the training set

Bagging and Boosting

- Experiments with a new boosting algorithm, freund et al (AdaBoost)
- Bagging Predictors, Brieman
- Boosting Naïve Bayesian Learning on large subset of MEDLINE, W. Wilbur

Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Classification by decision tree induction
- Bayesian Classification
- Classification by Neural Networks
- Classification by Support Vector Machines (SVM)
- Instance Based Methods
- Prediction
- Classification accuracy
- Summary

Summary

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

References (1)

- C. Apte and S. Weiss. Data mining with decision trees and decision rules. Future Generation Computer Systems, 13, 1997.
- L. Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and Regression Trees. Wadsworth International Group, 1984.
- 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. In Proc. 1st Int. Conf. Knowledge Discovery and Data Mining (KDD'95), pages 39-44, Montreal, Canada, August 1995.
- U. M. Fayyad. Branching on attribute values in decision tree generation. In Proc. 1994 AAAI Conf., pages 601-606, AAAI Press, 1994.
- J. Gehrke, R. Ramakrishnan, and V. Ganti. Rainforest: A framework for fast decision tree construction of large datasets. In Proc. 1998 Int. Conf. Very Large Data Bases, pages 416-427, New York, NY, August 1998.
- J. Gehrke, V. Gant, R. Ramakrishnan, and W.-Y. Loh, BOAT -- Optimistic Decision Tree Construction . In *SIGMOD'99* , Philadelphia, Pennsylvania, 1999

References (2)

- M. Kamber, L. Winstone, W. Gong, S. Cheng, and J. Han. Generalization and decision tree induction: Efficient classification in data mining. In Proc. 1997 Int. Workshop Research Issues on Data Engineering (RIDE'97), Birmingham, England, April 1997.
- B. Liu, W. Hsu, and Y. Ma. Integrating Classification and Association Rule Mining. *Proc. 1998 Int. Conf. Knowledge Discovery and Data Mining (KDD'98)* New York, NY, Aug. 1998.
- W. Li, J. Han, and J. Pei, *CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules,* , Proc. 2001 Int. Conf. on Data Mining (ICDM'01), San Jose, CA, Nov. 2001.
- J. Magidson. The Chaid approach to segmentation modeling: Chi-squared automatic interaction detection. In R. P. Bagozzi, editor, Advanced Methods of Marketing Research, pages 118-159. Blackwell Business, Cambridge Massechusetts, 1994.
- M. Mehta, R. Agrawal, and J. Rissanen. SLIQ : A fast scalable classifier for data mining. (EDBT'96), Avignon, France, March 1996.

References (3)

- T. M. Mitchell. *Machine Learning*. McGraw Hill, 1997.
- S. K. Murthy, Automatic Construction of Decision Trees from Data: A Multi-Diciplinary Survey, Data Mining and Knowledge Discovery 2(4): 345-389, 1998
- J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81-106, 1986.
- J. R. Quinlan. Bagging, boosting, and c4.5. In Proc. 13th Natl. Conf. on Artificial Intelligence (AAAI'96), 725-730, Portland, OR, Aug. 1996.
- R. Rastogi and K. Shim. Public: A decision tree classifer that integrates building and pruning. In Proc. 1998 Int. Conf. Very Large Data Bases, 404-415, New York, NY, August 1998.
- J. Shafer, R. Agrawal, and M. Mehta. SPRINT : A scalable parallel classifier for data mining. In Proc. 1996 Int. Conf. Very Large Data Bases, 544-555, Bombay, India, Sept. 1996.
- S. M. Weiss and C. A. Kulikowski. Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems. Morgan Kaufman, 1991.
- S. M. Weiss and N. Indurkhya. *Predictive Data Mining*. Morgan Kaufmann, 1997.