COURSE NAME:
DATA WAREHOUSING & DATA MINING

LECTURE 19
TOPICS TO BE COVERED:

- Other Classification Methods
  - Genetic Algorithm
  - Rough Sets
  - Fuzzy techniques
  - Support Vector Machines

OTHER CLASSIFICATION METHODS

- Genetic algorithm
- Rough set approach
- Fuzzy set approaches

GENETIC ALGORITHMS

- In genetic algorithms, populations of rules “evolve” via operations of crossover and mutation until all rules within a population satisfy a specified threshold.
- 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

GENETIC ALGORITHMS

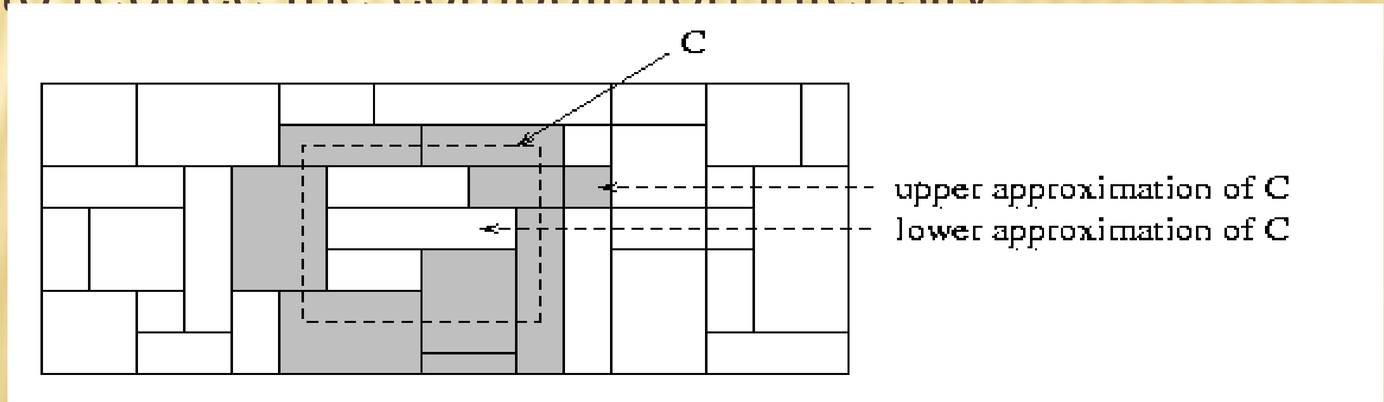
- As a simple example, suppose that samples in a given training set are described by two Boolean attributes, A_1 *and* A_2, and that there are two classes, C_1 *and* C_2 *.The rule "IF* A_1 *AND NOT* A_2 *THEN* C_2*" can be* encoded as the bit string "100," where the two leftmost bits represent attributes A_1 *and* A_2*, respectively, and the rightmost bit represents the class. Similarly, the rule "IF NOT* A_1 *AND NOT* A_2 *THEN* C_1*" can be encoded as "001." If an attribute has k values, where k > 2, then k bits may be used to encode the attribute's values. Classes can be encoded* in a similar fashion.

GENETIC ALGORITHMS

- Based on the notion of survival of the fittest, a new population is formed to consist of the *fittest rules in the current population, as well as offspring of these rules. Typically, the* fitness of a rule is assessed by its classification accuracy on a set of training samples.
- Offspring are created by applying genetic operators such as crossover and mutation.
- In crossover, substrings from pairs of rules are swapped to form new pairs of rules. In mutation, randomly selected bits in a rule's string are inverted.
- The process of generating new populations based on prior populations of rules continues until a population, *P, evolves where each rule in P satisfies a prespecified fitness* threshold.

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



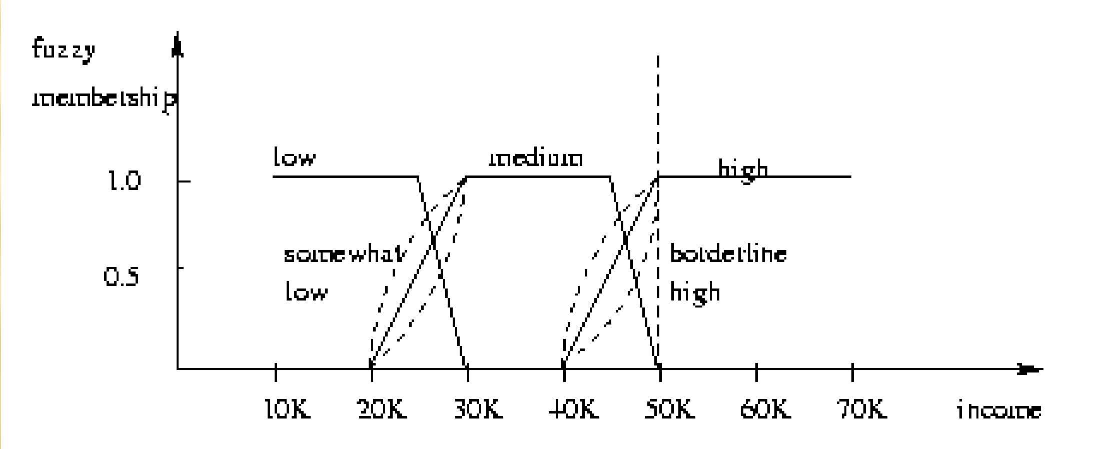
ROUGH SET APPROACH

- The lower and upper approximations for a class *C are shown in Figure* ,where each rectangular region represents an equivalence class. Decision rules can be generated for each class. Typically, a decision table is used to represent the rules.

- Rough set theory can be used to approximately define classes that are not distinguishable based on the available attributes

FUZZY SETS

- 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



FUZZY SETS

- 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

- Fuzzy set approaches replace “brittle” threshold cutoffs for continuous-valued attributes with degree of membership functions.

HISTORY OF SVM (SUPPORT VECTOR MACHINES)

- SVM is related to statistical learning theory
- SVM was first introduced in 1992
- SVM becomes popular because of its success in handwritten digit recognition
  - 1.1% test error rate for SVM. This is the same as the error rates of a carefully constructed neural network, LeNet 4.
- SVM is now regarded as an important example of “kernel methods”, one of the key area in machine learning
  - Note: the meaning of “kernel” is different from the “kernel” function for Parzen windows

SUPPORT VECTOR MACHINE

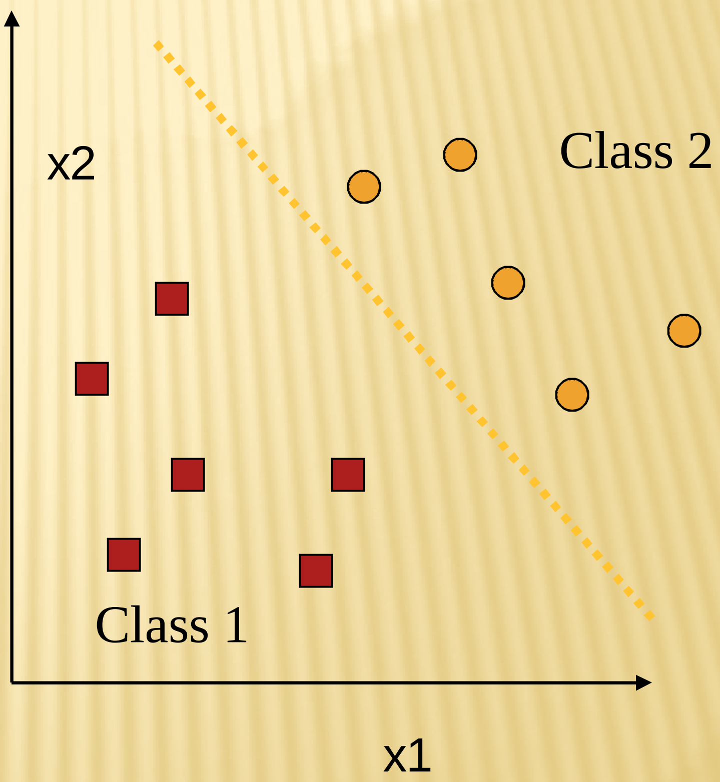
- A Support Vector Machine (SVM) is an algorithm for the classification of both linear and nonlinear data. It transforms the original data in a higher dimension, from where it can find a hyperplane for separation of the data using essential training tuples called support vectors.

SUPPORT VECTOR MACHINE

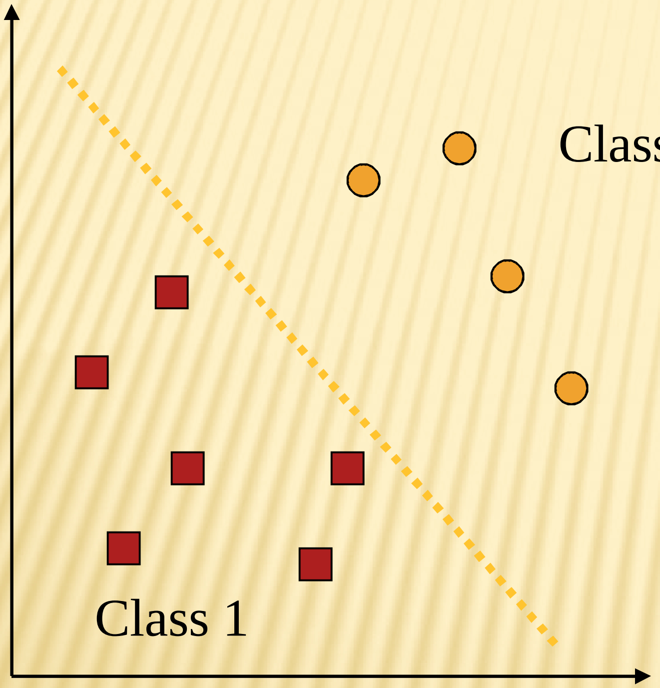
- The training time of even the fastest SVMs can be
- extremely slow, they are highly accurate, owing to their ability to model complex nonlinear decision boundaries.
- They are much less prone to overfitting than other methods.
- The support vectors found also provide a compact description of the learned model.
- SVMs can be used for prediction as well as classification.
- They have been applied to a number of areas, including handwritten digit recognition, object recognition, and speaker identification, as well as benchmark time-series prediction tests.

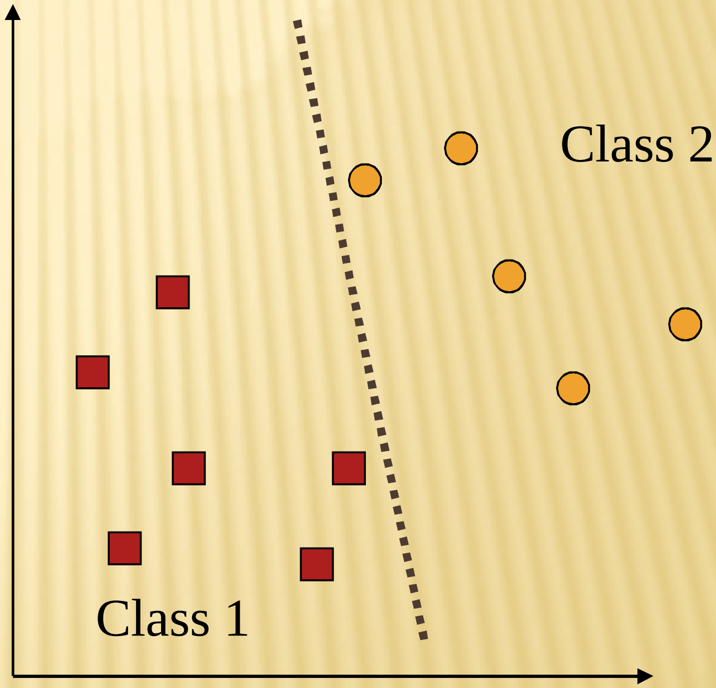
WHAT IS A GOOD DECISION BOUNDARY?

- Consider a two-class, linearly separable classification problem
- Many decision boundaries!
  - The Perceptron algorithm can be used to find such a boundary
  - Different algorithms have been proposed
- Are all decision boundaries equally good?



EXAMPLES OF BAD DECISION BOUNDARIES





THE CASE WHEN THE DATA ARE LINEARLY SEPARABLE

- To explain the mystery of SVMs, let's first look at the simplest case—a two-class problem where the classes are linearly separable.
- Let the data set *D be given as* (X_1, y_1), (X_2, y_2), ..., $(X_{|D|}, y_{|D|})$,***where*** X_i ***is the set of training tuples with associated class labels,*** y_i.
- Each y_i *can take one of two values, either+1 or-1 (i.e.,* $y_i \in \{1,-1\}$*), corresponding* to the classes *buys_computer = yes and buys_computer = no, respectively.*
- *To aid in visualization,* let's consider an example based on two input attributes, *A1 and A2, as shown in* Figure next slide. From the graph, we see that the 2-D data are linearly separable because a straight line can be drawn to separate all of the tuples of class +1 from all of the tuples of class-1.

- A separating hyperplane can be written as

$$W.X+b = 0;$$

- Where ***W is a weight vector, namely,***

$$W = \{w_1, w_2, \ldots , w_n\};$$

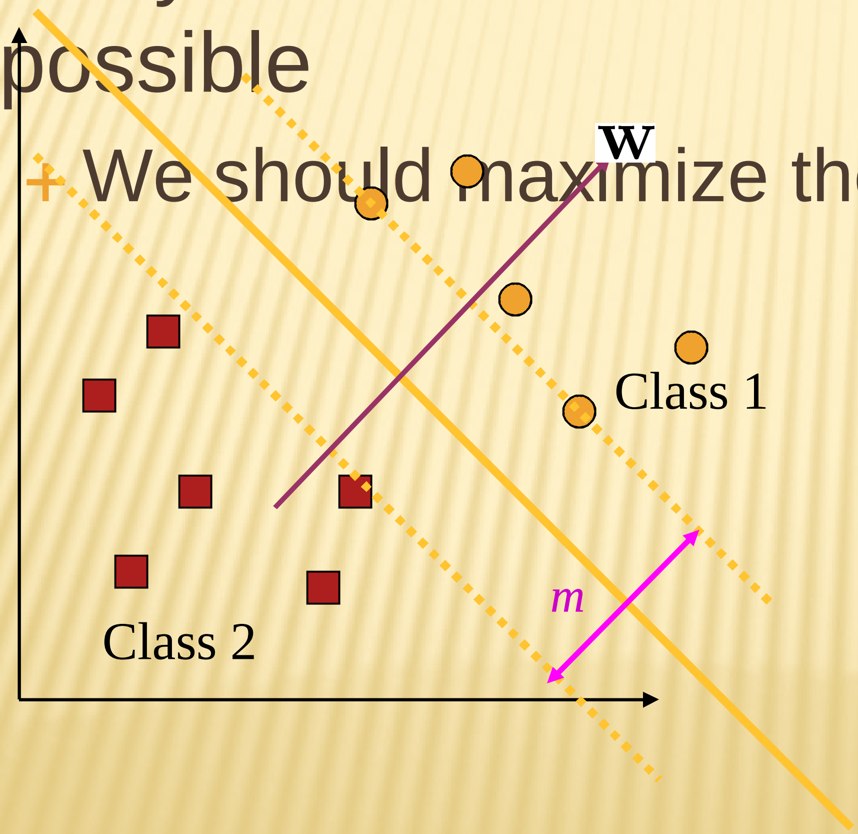
- ***n is the number of attributes;***
- and *b is a scalar, often referred to as a bias. T*

LARGE-MARGIN DECISION BOUNDARY

- The decision boundary should be as far away from the data of both classes as possible
  - We should maximize the m

class 1, $y = +1$ ($buys\_computer = yes$)

class 2, $y = -1$ ($buys\_computer = no$)



- Training tuples are 2-D, e.g., ***X = (x_1, x_2),***
- where x_1 *and* x_2 *are the values of attributes* A_1 *and* A_2*, respectively, for X. If we think of b as an additional weight,* w_0*, we can rewrite the above separating hyperplane as*

$$w_0+w_1x_1+w_2x_2 = 0$$

- Thus, any point that lies above the separating hyperplane satisfies

$$w_0+w_1x_1+w_2x_2 > 0$$

- Similarly, any point that lies below the separating hyperplane satisfies

$$w_0+w_1x_1+w_2x_2 < 0$$

THE CASE WHEN THE DATA ARE LINEARLY INSEPARABLE

- The approach described for linear SVMs can be extended to create *nonlinear SVMs for the classification of linearly inseparable data (also called nonlinearly separable data, or nonlinear data, for short). Such SVMs are capable of finding* nonlinear decision boundaries (i.e., nonlinear hypersurfaces) in input space.
- SVM by extending the approach for linear SVMs as follows. There are two main steps.
- In the first step, we transform the original input data into a higher dimensional space using a nonlinear mapping. Several common nonlinear mappings can be used in this step.
- The second step searches for a linear separating hyperplane in the new space. We again end up with a quadratic optimization problem that can be solved using the linear SVM formulation. The maximal marginal hyperplane found in the new space corresponds to a nonlinear separating hypersurface in the original space.

STRENGTHS AND WEAKNESSES OF SVM

- Strengths
  - Training is relatively easy
    - No local optimal, unlike in neural networks
  - It scales relatively well to high dimensional data
  - Tradeoff between classifier complexity and error can be controlled explicitly
  - Non-traditional data like strings and trees can be used as input to SVM, instead of feature vectors
  - Inherent feature selection capability
- Weaknesses
  - Need to choose a “good” kernel function.