

## **Data Mining and Machine Learning**

### **Lecture Notes – Module 2**

**Classification:** Basic Concepts, Decision Tree Induction, Bayes Classification Methods, Model Evaluation and Selection, Techniques to Improve Classification Accuracy.

**Classification and Prediction:** Support Vector Machines – Associative Classification – Lazy Learners – Other Classification Methods – Prediction.

#### **Textbooks:**

1. Jiawei Han and Micheline Kamber: Data Mining Concepts and Techniques, Elsevier, 2nd Edition, 2009.
2. Stephen Marsland, “Machine Learning - An Algorithmic Perspective”, Second Edition, CRC Press - Taylor and Francis Group, 2015.
3. Ethem Alpaydin, “Introduction to Machine Learning”, Second Edition, MIT Press, Prentice Hall of India (PHI) Learning Pvt. Ltd. 2010.
4. Xindong Wu and Vipin Kumar: The top ten Algorithms in Data Mining, Chapman and Hall/CRC press.
5. Pang-Ning Tan, Michael Steinbach and Vipin Kumar, “Introduction to Data Mining”, Pearson Education, 2007.
6. DISCOVERING KNOWLEDGE IN DATA, An Introduction to Data Mining, Second Edition, Daniel T. Larose, Chantal D. Larose.

#### **Reference Books:**

1. K.P. Soman, Shyam Diwakar and V. Aja, “Insight into Data Mining Theory and Practice”, Eastern Economy Edition, Prentice Hall of India, 2006.
2. G. K. Gupta, “Introduction to Data Mining with Case Studies”, Eastern Economy Edition, Prentice Hall of India, 2006.
3. Christopher Bishop, “Pattern Recognition and Machine Learning”, CBS Publishers & Distributors, 2010.
4. Mehryar Mohri, Afshin R. Ameet Talwalkar, "Foundations of Machine Learning", MIT Press, 2012.

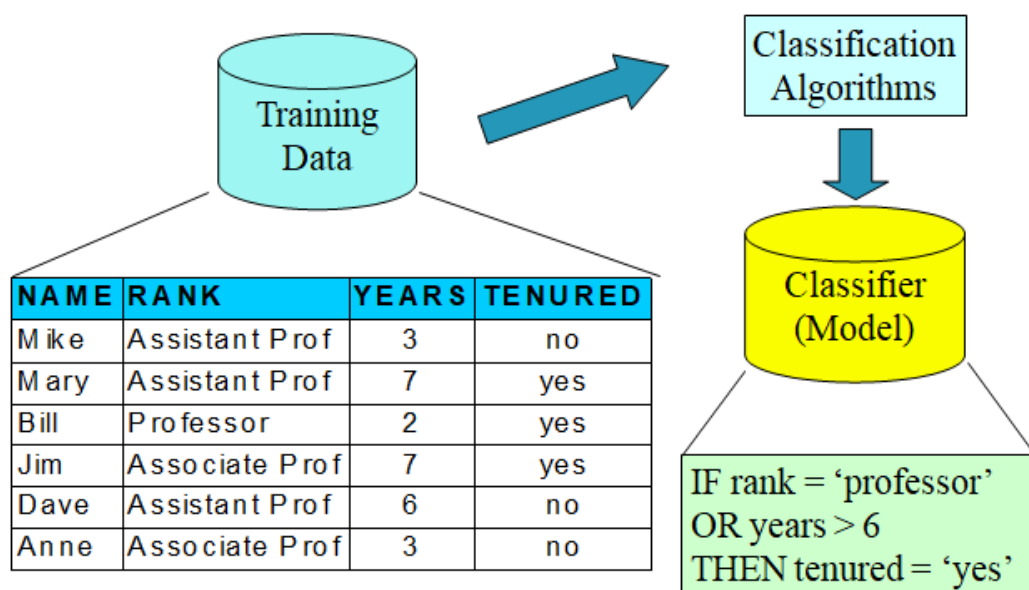
## 1. What is classification? Explain the general approach to classification.

### ■ Classification

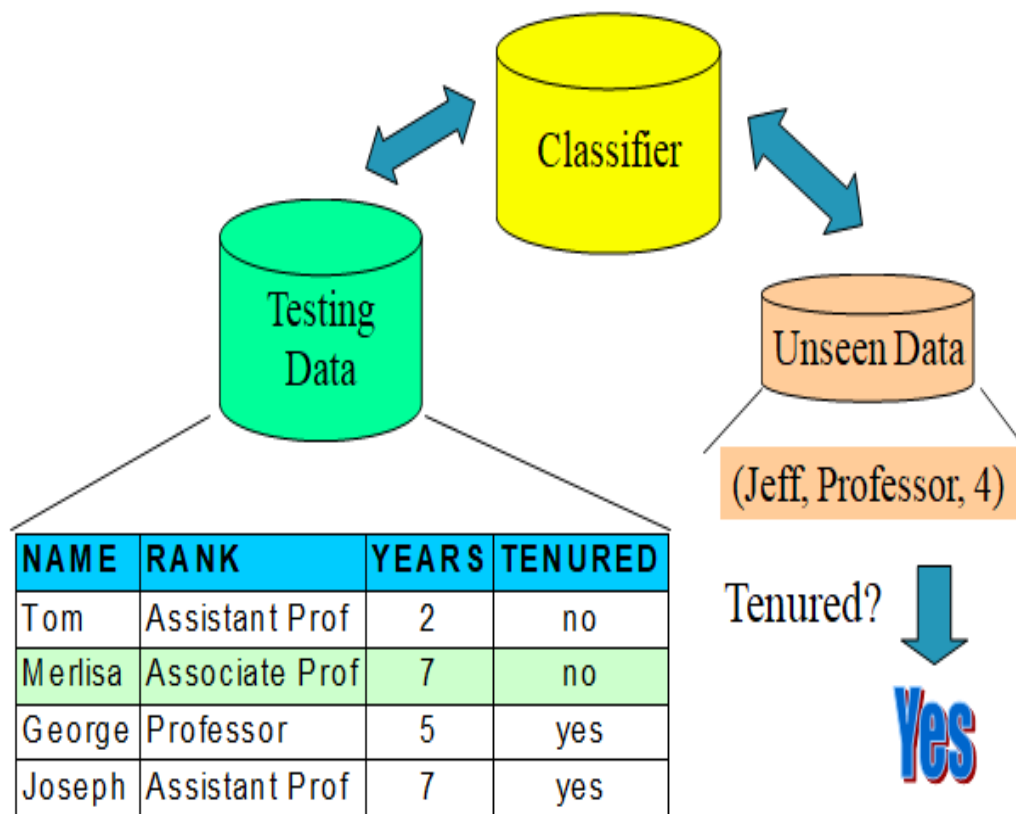
- predicts categorical class labels (discrete or nominal)
- classifies data (constructs a model) based on the training set and the values (class labels) in a classifying attribute and uses it in classifying new data

### 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 overfitting)
  - If the accuracy is acceptable, use the model to classify new data
- Note: If *the test set* is used to select models, it is called validation (test) set



**Figure : Model Construction**



**Figure : Model Prediction**

## 2. Explain Decision Tree Induction in detail.

### ID3 Basic algorithm (a greedy algorithm)

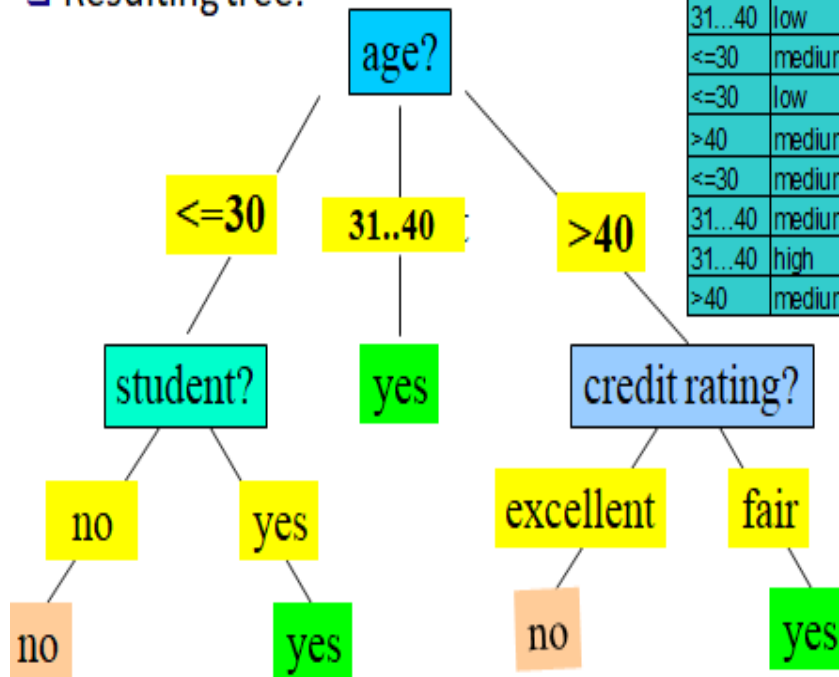
- Tree is constructed in a top-down recursive divide-and-conquer manner
- At start, all the training examples are at the root
- Attributes are categorical (if continuous-valued, they are discretized in advance)
- Examples are partitioned recursively based on selected attributes
- Test attributes are 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

**Example:**

- Training data set: Buys\_computer
- The data set follows an example of Quinlan's ID3 (Playing Tennis)
- Resulting tree:



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

### 3. Explain the Terms Information Gain, Entropy, Gini Index, Gini Ratio.

#### ■ Entropy (Information Theory)

■ A measure of uncertainty associated with a random variable

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

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

■ Interpretation:

- Higher entropy => higher uncertainty
- Lower entropy => lower uncertainty

#### ■ Conditional Entropy

■  $H(Y|X) = \sum_x p(x)H(Y|X = x)$



- 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  $|C_i \cap D| / |D|$

- **Expected information** (entropy) needed to classify a tuple in  $D$ :

$$Info(D) = - \sum_{i=1}^m p_i \log_2(p_i) \quad \text{Eq (5)}$$

- **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 gained** by branching on attribute  $A$

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

- Information gain measure is biased towards attributes with a large number of values
- C4.5 (a successor of ID3) 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.  $SplitInfo_{income}(D) = -\frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) - \frac{4}{14} \times \log_2\left(\frac{4}{14}\right) = 1.557$

- $gain\_ratio(income) = 0.029 / 1.557 = 0.019$

- The attribute with the maximum gain ratio is selected as the splitting attribute

- 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 (**need to enumerate all the possible splitting points for each attribute**)

#### 4. Explain Naïve Bayes Classifier with an Example.

A statistical classifier: performs *probabilistic prediction*, i.e., predicts class membership probabilities

Foundation: Based on Bayes' Theorem.

Performance: A simple Bayesian classifier, *naïve Bayesian classifier*, has comparable performance with decision tree and selected neural network classifiers

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

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

- Total probability Theorem: 
$$P(B) = \sum_{i=1}^M P(B|A_i)P(A_i)$$
- Bayes' Theorem: 
$$P(H|\mathbf{X}) = \frac{P(\mathbf{X}|H)P(H)}{P(\mathbf{X})} = P(\mathbf{X}|H) \times P(H) / P(\mathbf{X})$$
  - Let  $\mathbf{X}$  be a data sample ("evidence"): class label is unknown
  - Let  $H$  be a *hypothesis* that  $\mathbf{X}$  belongs to class  $C$
  - Classification is to determine  $P(H|\mathbf{X})$ , (i.e., *posteriori probability*): the probability that the hypothesis holds given the observed data sample  $\mathbf{X}$
  - $P(H)$  (*prior probability*): the initial probability
    - E.g.,  $\mathbf{X}$  will buy computer, regardless of age, income, ...
  - $P(\mathbf{X})$ : probability that sample data is observed
  - $P(\mathbf{X}|H)$  (*likelihood*): the probability of observing the sample  $\mathbf{X}$ , given that the hypothesis holds
    - E.g., Given that  $\mathbf{X}$  will buy computer, the prob. that  $\mathbf{X}$  is 31..40, medium income
- Given training data  $\mathbf{X}$ , *posteriori probability of a hypothesis*  $H$ ,  $P(H|\mathbf{X})$ , follows the Bayes' theorem
 
$$P(H|\mathbf{X}) = \frac{P(\mathbf{X}|H)P(H)}{P(\mathbf{X})} = P(\mathbf{X}|H) \times P(H) / P(\mathbf{X})$$
- Informally, this can be viewed as *posteriori* = *likelihood* x *prior/evidence*
- Predicts  $\mathbf{X}$  belongs to  $C_i$  iff the probability  $P(C_i|\mathbf{X})$  is the highest among all the  $P(C_k|\mathbf{X})$  for all the  $k$  classes
- Practical difficulty: It requires initial knowledge of many probabilities, involving significant computational cost
- Let  $D$  be a training set of tuples and their associated class labels, and each tuple is

represented by an n-D attribute vector  $\mathbf{X} = (x_1, x_2, \dots, x_n)$

- Suppose there are  $m$  classes  $C_1, C_2, \dots, C_m$ .
- Classification is to derive the maximum posteriori, i.e., the maximal  $P(C_i|\mathbf{X})$
- This can be derived from Bayes' theorem

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

- Since  $P(\mathbf{X})$  is constant for all classes, only needs to be maximized

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

#### ■ 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 Bayes Classifier

### 5. Explain Classifier Evaluation Metrics: Confusion Matrix, Accuracy, Error Rate, Sensitivity and Specificity, Precision and Recall, and F-measures

#### Confusion Matrix:

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

#### Example of Confusion Matrix:

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

- Given  $m$  classes, an entry,  $CM_{i,j}$  in a **confusion matrix** indicates # of tuples in class  $i$  that were labeled by the classifier as class  $j$
- May have extra rows/columns to provide totals
- **Classifier Accuracy**, or recognition rate: percentage of test set tuples that are correctly classified



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

- **Error rate:**  $1 - \text{accuracy}$ , or

$$\text{Error rate} = (FP + FN)/All$$

- **Sensitivity:** True Positive recognition rate

$$\text{Sensitivity} = TP/P$$

- **Specificity:** True Negative recognition rate

$$\text{Specificity} = TN/N$$

- **Precision:** exactness – what % of tuples that the classifier labeled as positive are actually positive

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

- **Recall:** completeness – what % of positive tuples did the classifier label as positive?

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

- Perfect score is 1.0

- Inverse relationship between precision & recall

- **F measure ( $F_1$  or F-score):** harmonic mean of precision and recall,

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

- **$F_\beta$ :** weighted measure of precision and recall

- assigns  $\beta$  times as much weight to recall as to precision

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

## 6. What is Holdout, Cross-Validation Methods & Boosting, ROC Curves ?

### Holdout method

- Given data is randomly partitioned into two independent sets
  - Training set (e.g., 2/3) for model construction
  - Test set (e.g., 1/3) for accuracy estimation
- Random sampling: a variation of holdout
  - Repeat holdout  $k$  times, accuracy = avg. of the accuracies obtained

### Cross-validation ( $k$ -fold, where $k = 10$ is most popular)

- Randomly partition the data into  $k$  *mutually exclusive* subsets, each approximately equal size
- At  $i$ -th iteration, use  $D_i$  as test set and others as training set
- Leave-one-out:  $k$  folds where  $k = \#$  of tuples, for small sized data



- f. **\*Stratified cross-validation\***: folds are stratified so that class dist. in each fold is approx. the same as that in the initial data

### Bootstrap

- Works well with small data sets
- Samples the given training tuples uniformly *with replacement*
- i.e., each time a tuple is selected, it is equally likely to be selected again and re-added to the training set

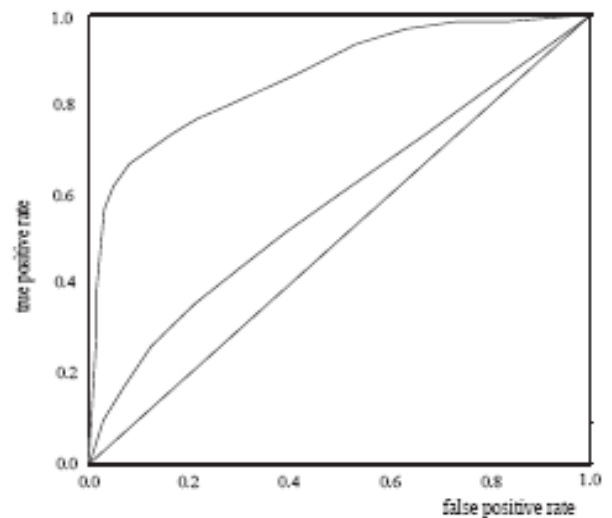
Several bootstrap methods, and a common one is **.632 bootstrap**

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

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

### ROC Curves

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



- A model with perfect accuracy will have an area of 1.0

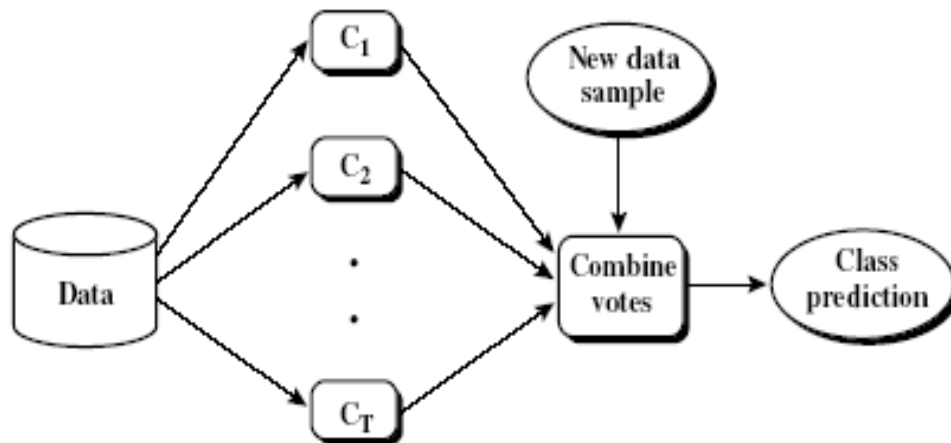
## 7. Explain Ensemble Methods, Bagging: Bootstrap Aggregation, AdaBoost and Random Forest to improve classification accuracy.

Ensemble methods

- Use a combination of models to increase accuracy
- Combine a series of  $k$  learned models,  $M_1, M_2, \dots, M_k$ , with the aim of creating an improved model  $M^*$

Popular ensemble methods

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



### Bagging: Bootstrap Aggregation

- Analogy: Diagnosis based on multiple doctors' majority vote
- Training
  - Given a set  $D$  of  $d$  tuples, at each iteration  $i$ , a training set  $D_i$  of  $d$  tuples is sampled with replacement from  $D$  (i.e., bootstrap)
  - A classifier model  $M_i$  is learned for each training set  $D_i$
- Classification: classify an unknown sample  $X$ 
  - Each classifier  $M_i$  returns its class prediction
  - The bagged classifier  $M^*$  counts the votes and assigns the class with the most votes to  $X$
- Prediction: can be applied to the prediction of continuous values by taking the average value of each prediction for a given test tuple
- Accuracy

- Often significantly better than a single classifier derived from  $D$
- For noise data: not considerably worse, more robust
- Proved improved accuracy in prediction

### Boosting

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

### Adaboost

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

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

- The weight of classifier  $M_i$ 's vote is

$$\log \frac{1 - error(M_i)}{error(M_i)}$$

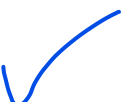
## Random Forest

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

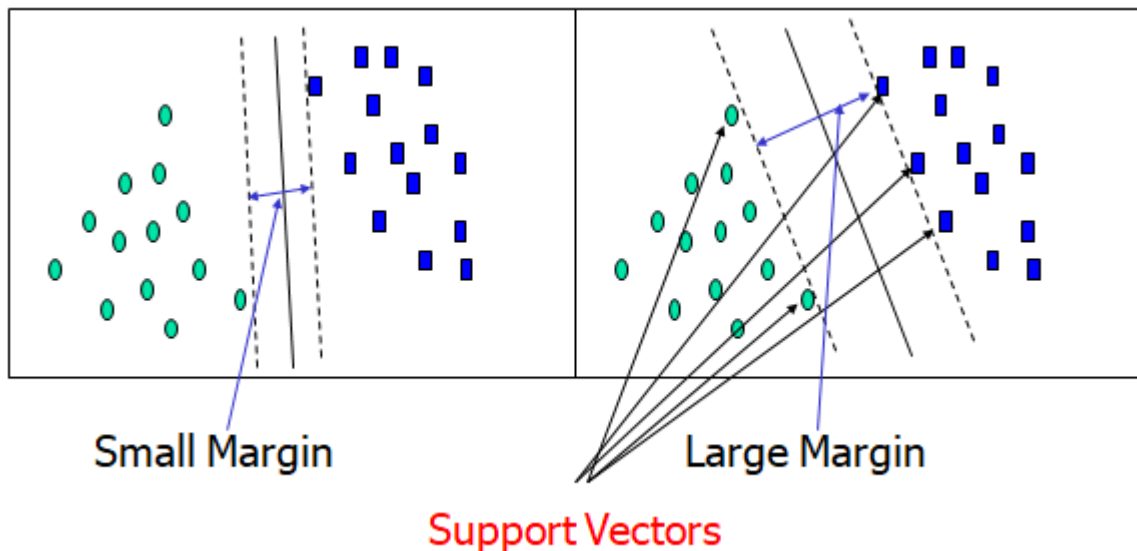
## Classification of Class-Imbalanced Data Sets

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

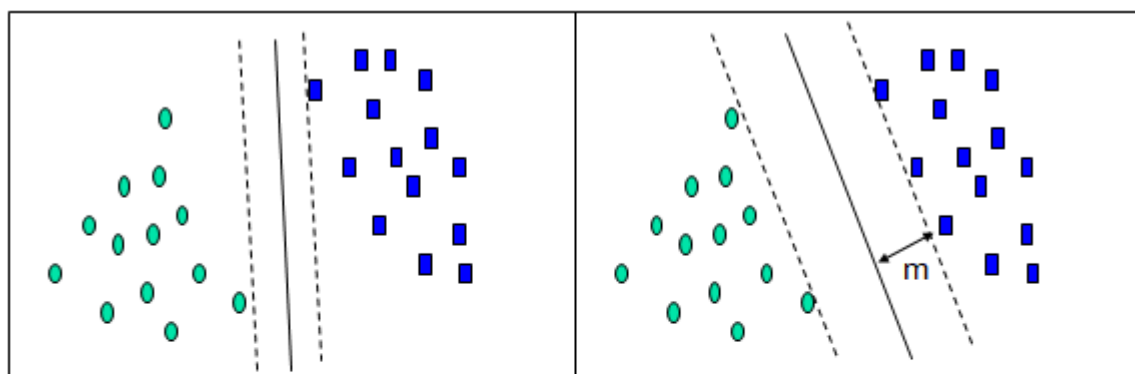
## 8. Explain Support Vector Machines

- 
- A relatively new classification method for both linear and nonlinear data
  - It uses a nonlinear mapping to transform the original training data into a higher dimension
  - With the new dimension, it searches for the linear optimal separating **hyperplane** (i.e., “decision boundary”)

- With an appropriate nonlinear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane
- SVM finds this hyperplane using **support vectors** (“essential” training tuples) and **margins** (defined by the support vectors)



### SVM—When Data Is Linearly Separable



Let data  $D$  be  $(\mathbf{X}_1, y_1), \dots, (\mathbf{X}_{|D|}, y_{|D|})$ , where  $\mathbf{X}_i$  is the set of training tuples associated with the class labels  $y_i$

There are infinite lines (hyperplanes) separating the two classes but we want to find the best one (the one that minimizes classification error on unseen data)

*SVM searches for the hyperplane with the largest margin, i.e., **maximum marginal hyperplane (MMH)***

- A separating hyperplane can be written as

$$\mathbf{W} \bullet \mathbf{X} + b = 0$$

where  $\mathbf{W} = \{w_1, w_2, \dots, w_n\}$  is a weight vector and  $b$  a scalar (bias)

- For 2-D it can be written as

$$w_0 + w_1 x_1 + w_2 x_2 = 0$$

- The hyperplane defining the sides of the margin:

$$H_1: w_0 + w_1 x_1 + w_2 x_2 \geq 1 \quad \text{for } y_i = +1, \text{ and}$$

$$H_2: w_0 + w_1 x_1 + w_2 x_2 \leq -1 \quad \text{for } y_i = -1$$

- Any training tuples that fall on hyperplanes  $H_1$  or  $H_2$  (i.e., the sides defining the margin) are **support vectors**
- This becomes a **constrained (convex) quadratic optimization** problem: Quadratic objective function and linear constraints  $\rightarrow$  *Quadratic Programming (QP)*  $\rightarrow$  Lagrangian multipliers

### Why Is SVM Effective on High Dimensional Data?

- The **complexity** of trained classifier is characterized by the # of support vectors rather than the dimensionality of the data
- The **support vectors** are the essential or critical training examples —they lie closest to the decision boundary (MMH)
- If all other training examples are removed and the training is repeated, the same separating hyperplane would be found
- The number of support vectors found can be used to compute an (upper) bound on the expected error rate of the SVM classifier, which is independent of the data dimensionality
- Thus, an SVM with a small number of support vectors can have good generalization, even when the dimensionality of the data is high

### SVM—Linearly Inseparable

- Instead of computing the dot product on the transformed data, it is math. equivalent to applying a kernel function  $K(X_i, X_j)$  to the original data, i.e.,  $K(X_i, X_j) = \Phi(X_i) \cdot \Phi(X_j)$
- Typical Kernel Functions


**Polynomial kernel of degree  $h$  :**  $K(X_i, X_j) = (X_i \cdot X_j + 1)^h$

**Gaussian radial basis function kernel :**  $K(X_i, X_j) = e^{-\|X_i - X_j\|^2 / 2\sigma^2}$

**Sigmoid kernel :**  $K(X_i, X_j) = \tanh(\kappa X_i \cdot X_j - \delta)$


- SVM can also be used for classifying multiple ( $> 2$ ) classes and for regression analysis (with additional parameters)

## 9. What is Associative Classification ?

- 
- Associative classification: Major steps
    - Mine data to find strong associations between frequent patterns (conjunctions of attribute-value pairs) and class labels
    - Association rules are generated in the form of
$$P_1 \wedge p_2 \dots \wedge p_l \rightarrow "A_{\text{class}} = C" \text{ (conf, sup)}$$
    - Organize the rules to form a rule-based classifier
  - Why effective?
    - It explores highly confident associations among multiple attributes and may overcome some constraints introduced by decision-tree induction, which considers only one attribute at a time
    - Associative classification has been found to be often more accurate than some traditional classification methods, such as C4.5
  - CBA (Classification Based on Associations)
    - Mine possible association rules in the form of
      - Cond-set (a set of attribute-value pairs)  $\rightarrow$  class label
    - Build classifier: Organize rules according to decreasing precedence based on confidence and then support
  - CMAR (Classification based on Multiple Association Rules: Li, Han, Pei, ICDM'01)
    - Classification: Statistical analysis on multiple rules
  - CPAR (Classification based on Predictive Association Rules: Yin & Han, SDM'03)
    - Generation of predictive rules (FOIL-like analysis) but allow covered rules to retain with reduced weight
    - Prediction using best  $k$  rules

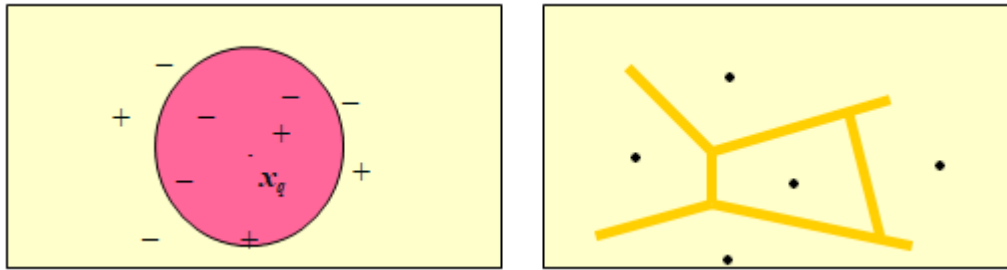
High efficiency, accuracy similar to CMAR

## 10. Explain 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,  $\text{dist}(\mathbf{X}_1, \mathbf{X}_2)$
  - Target function could be discrete- or real- valued
  - For discrete-valued,  $k$ -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



- k-NN for real-valued prediction for a given unknown tuple
  - Returns 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  $x_q$ 
    - Give greater weight to closer neighbors
- 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

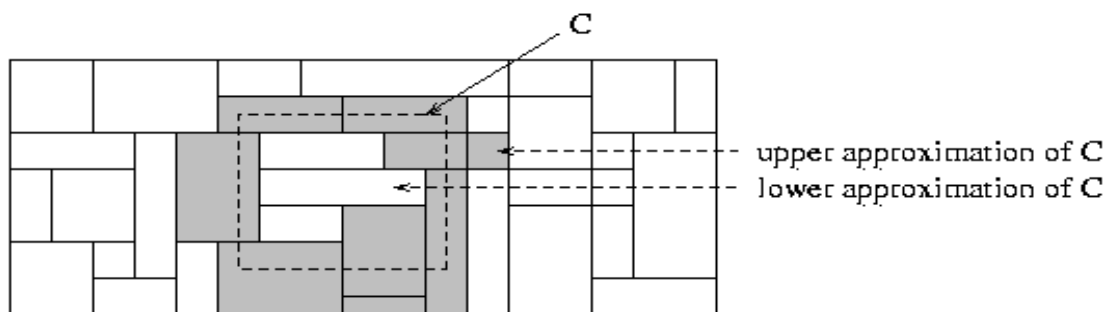
## 11. Explain Genetic Algorithms (GA), Rough Set Approach, Fuzzy Set Approaches

**Genetic Algorithm:** based on an analogy to biological evolution

- An initial **population** is created consisting of randomly generated rules
  - Each rule is represented by a string of bits
  - E.g., if  $A_1$  and  $\neg A_2$  then  $C_2$  can be encoded as 100
  - If an attribute has  $k > 2$  values,  $k$  bits can be used
- Based on the notion of survival of the **fittest**, a new population is formed to consist of the fittest rules and their offspring
- The *fitness of a rule* is represented by its classification accuracy on a set of training examples
- Offspring are generated by *crossover* and *mutation*
- The process continues until a population  $P$  evolves *when each rule in  $P$  satisfies a prespecified threshold*, Slow but easily parallelizable

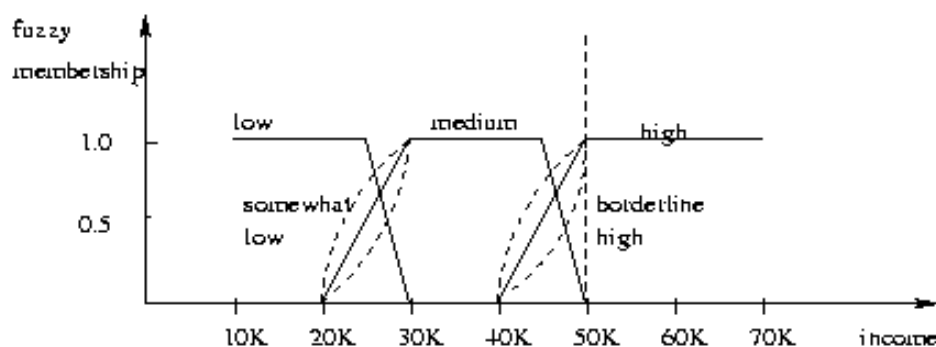
## Rough Set Approach

- 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** (which stores the differences between attribute values for each pair of data tuples) is used to reduce the computation intensity



## Fuzzy Set Approaches

- Fuzzy logic uses truth values between 0.0 and 1.0 to represent the degree of membership (such as in a *fuzzy membership graph*)
- Attribute values are converted to fuzzy values. Ex.:
  - Income,  $x$ , is assigned a fuzzy membership value to each of the discrete categories {low, medium, high}, e.g. \$49K belongs to “medium income” with fuzzy value 0.15 but belongs to “high income” with fuzzy value 0.96
  - Fuzzy membership values do not have to sum to 1.
- Each applicable rule contributes a vote for membership in the categories
- Typically, the truth values for each predicted category are summed, and these sums are combined



### Possible Questions:

1.
  - a) Illustrate Bayesian Classification with a neat diagram.
  - b) Discuss the relationship between the association mining and correlation analysis with examples.
2.
  - a) Outline the steps to select the best attribute for the root of a Decision tree" to play tennis "using ID3 algorithm with the given dataset.

Day	Outlook	Temp	Humidity	Wind	Tennis?
<b>D1</b>	<b>Sunny</b>	<b>Hot</b>	<b>High</b>	<b>Weak</b>	<b>No</b>
<b>D2</b>	<b>Sunny</b>	<b>Hot</b>	<b>High</b>	<b>Strong</b>	<b>No</b>
<b>D3</b>	<b>Overcast</b>	<b>Hot</b>	<b>High</b>	<b>Weak</b>	<b>Yes</b>
<b>D4</b>	<b>Rain</b>	<b>Mild</b>	<b>High</b>	<b>Weak</b>	<b>Yes</b>
<b>D5</b>	<b>Rain</b>	<b>Cool</b>	<b>Normal</b>	<b>Weak</b>	<b>Yes</b>
<b>D6</b>	<b>Rain</b>	<b>Cool</b>	<b>Normal</b>	<b>Strong</b>	<b>No</b>
<b>D7</b>	<b>Overcast</b>	<b>Cool</b>	<b>Normal</b>	<b>Strong</b>	<b>Yes</b>
<b>D8</b>	<b>Sunny</b>	<b>Mild</b>	<b>High</b>	<b>Weak</b>	<b>No</b>
<b>D9</b>	<b>Sunny</b>	<b>Cool</b>	<b>Normal</b>	<b>Weak</b>	<b>Yes</b>
<b>D10</b>	<b>Rain</b>	<b>Mild</b>	<b>Normal</b>	<b>Weak</b>	<b>Yes</b>
<b>D11</b>	<b>Sunny</b>	<b>Mild</b>	<b>Normal</b>	<b>Strong</b>	<b>Yes</b>
<b>D12</b>	<b>Overcast</b>	<b>Mild</b>	<b>High</b>	<b>Strong</b>	<b>Yes</b>
<b>D13</b>	<b>Overcast</b>	<b>Hot</b>	<b>Normal</b>	<b>Weak</b>	<b>Yes</b>
<b>D14</b>	<b>Rain</b>	<b>Mild</b>	<b>High</b>	<b>Strong</b>	<b>No</b>

- b) Using Naïve Bayes classifier classify a "Red Domestic SUV "as stolen or not. Use the given below data set.

Example No.	Color	Type	Origin	Stolen?
1	Red	Sports	Domestic	Yes
2	Red	Sports	Domestic	No
3	Red	Sports	Domestic	Yes
4	Yellow	Sports	Domestic	No
5	Yellow	Sports	Imported	Yes
6	Yellow	SUV	Imported	No
7	Yellow	SUV	Imported	Yes
8	Yellow	SUV	Domestic	No
9	Red	SUV	Imported	No
10	Red	Sports	Imported	Yes

3.
  - A) Explain two step procedures for Classification.
  - B) Define classification. Explain different types of attributes.
  - C) Explain Naïve Bayesian Classification with an example.

3. a) For the data set given below, calculate Information gain, Gain ratio and  $\Delta$ Gini for the attribute “Humidity”.

Outlook	Temperature	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

- b) Explain Naïve Bayesian classification. What is the Naïve Bayesian classifier prediction of buys\_computer for tuple X where age= “senior” and income= “low” in the dataset given below.

Age	Income	Student	Credit rating	Buys Computer
Youth	Low	No	Fair	No
Youth	High	Yes	Excellent	Yes
Middle_aged	Low	Yes	Fair	No
Middle_aged	High	Yes	Excellent	Yes
Senior	Low	No	Excellent	No
Senior	high	No	Fair	Yes

- c) Use KNN classifier to predict whether the new data point (x1,y1) = (57kg, 170cm) belongs to the class **Under Weight** or **Normal**. Also find appropriate K value for classification.

Weight (x2)	Height(y2)	Class
51	167	Under Weight
62	182	Normal
69	176	Normal
64	173	Normal
65	172	Normal
56	174	Under Weight
58	169	Normal
57	173	Normal
55	170	Normal

4.  
a) The following contingency table summarizes supermarket transaction data, where hot dogs refers to the transactions containing hot dogs,

hot dogs

refers to the transactions that do not contain hot dogs

hamburgers

refers to the transactions containing hamburgers, and

hamburgers

refers to the transactions that do not contain hamburgers.

	<i>hot dogs</i>	<i>hot dogs</i>	$\Sigma_{row}$
<i>hamburgers</i>	2000	500	2500
<i>hamburgers</i>	1000	1500	2500
$\Sigma_{col}$	3000	2000	5000

- i) Suppose that the association rule “hot dogs  $\Rightarrow$  hamburgers” is mined. Given a minimum support threshold of 25% and a minimum confidence threshold of 50%, is this association rule strong?
  - ii) Based on the given data, is the purchase of hot dogs independent of the purchase of hamburgers? If not, what kind of correlation relationship exists between the two?
- b) Explain decision tree induction with an example.