Matrix Data: Classification: Part 1

- Classification: Basic Concepts
- Decision Tree Induction
- Model Evaluation and Selection
- Summary
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
Prediction Problems: Classification vs. Numeric Prediction

- **Classification**
  - predicts categorical class labels
  - 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

- **Numeric Prediction**
  - models continuous-valued functions, i.e., predicts unknown or missing values

- **Typical applications**
  - Credit/loan approval:
  - Medical diagnosis: if a tumor is cancerous or benign
  - Fraud detection: if a transaction is fraudulent
  - Web page categorization: which category it is
Classification—A Two-Step Process (1)

- **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
    - For data point $i$: $<x_i, y_i>$
    - Features: $x_i$; class label: $y_i$
  - The model is represented as classification rules, decision trees, or mathematical formulae
    - Also called classifier
  - The set of tuples used for model construction is training set
Classification—A Two-Step Process (2)

- **Model usage**: for classifying future or unknown objects
  - **Estimate accuracy of the model**
    - The known label of test sample is compared with the classified result from the model
    - **Test set** is independent of training set (otherwise overfitting)
    - **Accuracy** rate is the percentage of test set samples that are correctly classified by the model
      - Most used for binary classes
    - **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
Process (1): Model Construction

Training Data

<table>
<thead>
<tr>
<th>NAME</th>
<th>RANK</th>
<th>YEARS</th>
<th>TENURED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mike</td>
<td>Assistant Prof</td>
<td>3</td>
<td>no</td>
</tr>
<tr>
<td>Mary</td>
<td>Assistant Prof</td>
<td>7</td>
<td>yes</td>
</tr>
<tr>
<td>Bill</td>
<td>Professor</td>
<td>2</td>
<td>yes</td>
</tr>
<tr>
<td>Jim</td>
<td>Associate Prof</td>
<td>7</td>
<td>yes</td>
</tr>
<tr>
<td>Dave</td>
<td>Assistant Prof</td>
<td>6</td>
<td>no</td>
</tr>
<tr>
<td>Anne</td>
<td>Associate Prof</td>
<td>3</td>
<td>no</td>
</tr>
</tbody>
</table>

Classification Algorithms

IF rank = ‘professor’
OR years > 6
THEN tenured = ‘yes’
Process (2): Using the Model in Prediction

<table>
<thead>
<tr>
<th>NAME</th>
<th>RANK</th>
<th>YEARS</th>
<th>TENURED</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Tom</td>
<td>Assistant Prof</td>
<td>2</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Merlisa</td>
<td>Associate Prof</td>
<td>7</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>George</td>
<td>Professor</td>
<td>5</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td>Joseph</td>
<td>Assistant Prof</td>
<td>7</td>
<td>yes</td>
<td></td>
</tr>
</tbody>
</table>

Testing Data

Classifier

Unseen Data

(Jeff, Professor, 4)

Tenured? Yes
Classification Methods Overview

• Part 1
  • Decision Tree
  • Model Evaluation

• Part 2
  • Bayesian Learning: Naïve Bayes, Bayesian belief network
  • Logistic Regression

• Part 3
  • SVM
  • kNN
  • Other Topics
Matrix Data: Classification: Part 1

• Classification: Basic Concepts
• Decision Tree Induction
• Model Evaluation and Selection
• Summary
Decision Tree Induction: An Example

- Training data set: Buys_computer
- The data set follows an example of Quinlan’s ID3 (Playing Tennis)
- Resulting tree:

<table>
<thead>
<tr>
<th>age</th>
<th>income</th>
<th>student</th>
<th>credit_rating</th>
<th>buys_computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;=30</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>high</td>
<td>no</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>31...40</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&gt;40</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&gt;40</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&gt;40</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>&gt;40</td>
<td>low</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>31...40</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&gt;40</td>
<td>medium</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>medium</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>31...40</td>
<td>medium</td>
<td>no</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>31...40</td>
<td>high</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
</tbody>
</table>
How to choose attributes?

Ages

- <=30
  - Yes
  - Yes
  - No
  - No
- 31...40
  - Yes
  - Yes
  - No
  - No
- >40
  - Yes
  - Yes
  - No
  - No

Credit_Rating

- Excellent
  - Yes
  - Yes
  - Yes
  - Yes
  - No
  - No
- Fair
  - Yes
  - Yes
  - Yes
  - Yes
  - No
  - No

VS.
Brief Review of Entropy

• Entropy (Information Theory)
  • A measure of uncertainty (impurity) associated with a random variable
  • Calculation: For a discrete random variable $Y$ taking $m$ distinct values \{${y_1, \ldots, y_m}$\},
    \[ H(Y) = -\sum_{i=1}^{m} p_i \log(p_i) , \text{ where } p_i = P(Y = y_i) \]
  • Interpretation:
    • Higher entropy => higher uncertainty
    • Lower entropy => lower uncertainty

• Conditional Entropy
  • $H(Y|X) = \sum_{x} p(x)H(Y|X = x)$
Attribute Selection Measure:
Information Gain (ID3/C4.5)

- Select the attribute with the highest information gain
- Let $p_i$ be the probability that an arbitrary tuple in $D$ belongs to class $C_i$, estimated by $|C_{i,D}|/|D|$
- Expected information (entropy) needed to classify a tuple in $D$: $\text{Info}(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$
- Information needed (after using $A$ to split $D$ into $v$ partitions) to classify $D$ (conditional entropy): $\text{Info}_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \text{Info}(D_j)$
- Information gained by branching on attribute $A$ $\text{Gain}(A) = \text{Info}(D) - \text{Info}_A(D)$
Attribute Selection: Information Gain

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

\[
\text{Info}(D) = I(9,5) = -\frac{9}{14} \log_2 \left( \frac{9}{14} \right) - \frac{5}{14} \log_2 \left( \frac{5}{14} \right) = 0.940
\]

<table>
<thead>
<tr>
<th>age</th>
<th>( p_i )</th>
<th>( n_i )</th>
<th>( I(p_i, n_i) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;=30</td>
<td>2</td>
<td>3</td>
<td>0.971</td>
</tr>
<tr>
<td>31…40</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>&gt;40</td>
<td>3</td>
<td>2</td>
<td>0.971</td>
</tr>
</tbody>
</table>

\[
I(2,3) = \frac{5}{14} I(2,3) + \frac{4}{14} I(4,0)
\]

\[
\frac{5}{14} I(2,3) = 0.694
\]

\[
I(2,3) \text{ means “age } \leq 30 \text{” has 5 out of 14 samples, with 2 yes’es and 3 no’s. Hence}
\]

\[
\text{Gain}(\text{age}) = \text{Info}(D) - \text{Info}_{\text{age}}(D) = 0.246
\]

Similarly,

\[
\text{Gain}(\text{income}) = 0.029
\]

\[
\text{Gain}(\text{student}) = 0.151
\]

\[
\text{Gain}(\text{credit_rating}) = 0.048
\]
Attribute Selection for a Branch

Which attribute next?

<table>
<thead>
<tr>
<th>age</th>
<th>income</th>
<th>student</th>
<th>credit_rating</th>
<th>buys_computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;=30</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>high</td>
<td>no</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>medium</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
</tbody>
</table>

\[ D_{age\leq30} \]

- \( \text{Gain}_{age\leq30}(income) = \text{Gain}_{age\leq30}(student) = 0.971 \)
- \( \text{Gain}_{age\leq30}(credit\_rating) = 0.02 \)

\( \text{Gain}_{age\leq30}(income) = - \frac{2}{5} \log_2 \frac{2}{5} - \frac{3}{5} \log_2 \frac{3}{5} = 0.971 \)
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 — use majority voting in the parent partition
Computing Information-Gain for Continuous-Valued Attributes

• Let attribute A be a continuous-valued attribute

• Must determine the best split point for A
  
  • Sort the value A in increasing order
  
  • Typically, the midpoint between each pair of adjacent values is considered as a possible split point
    
    • \((a_i + a_{i+1})/2\) is the midpoint between the values of \(a_i\) and \(a_{i+1}\)

  • The point with the minimum expected information requirement for A is selected as the split-point for A

• Split:
  
  • \(D_1\) is the set of tuples in D satisfying \(A \leq \text{split-point}\), and \(D_2\) is the set of tuples in D satisfying \(A > \text{split-point}\)
Gain Ratio for Attribute Selection (C4.5)

- 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.

\[
SplitInfo_{\text{income}}(D) = -\frac{4}{14} \times \log_2 \left( \frac{4}{14} \right) - \frac{6}{14} \times \log_2 \left( \frac{6}{14} \right) - \frac{4}{14} \times \log_2 \left( \frac{4}{14} \right) = 1.557
\]

- \(\text{gain\_ratio\{income\}} = 0.029/1.557 = 0.019\)
- The attribute with the maximum gain ratio is selected as the splitting attribute.
Gini Index (CART, IBM IntelligentMiner)

• 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)
Computation of Gini Index

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

\[
gini(D) = 1 - \left( \frac{9}{14} \right)^2 - \left( \frac{5}{14} \right)^2 = 0.459
\]

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

\[
gini_{\text{income} \in \{\text{low, medium}\}}(D) = \left( \frac{10}{14} \right) Gini(D_1) + \left( \frac{4}{14} \right) Gini(D_2)
\]

\[
= \frac{10}{14} \left( 1 - \left( \frac{7}{10} \right)^2 - \left( \frac{3}{10} \right)^2 \right) + \frac{4}{14} \left( 1 - \left( \frac{2}{4} \right)^2 - \left( \frac{2}{4} \right)^2 \right)
\]

\[
= 0.443
\]

\[
= Gini_{\text{income} \in \{\text{high}\}}(D).
\]

\(Gini_{\{\text{low,high}\}}\) is 0.458; \(Gini_{\{\text{medium,high}\}}\) is 0.450. Thus, split on the \{low,medium\} (and \{high\}) since it has the lowest Gini index
Comparing Attribute Selection Measures

• The three measures, in general, return good results but
  
  • **Information gain:**
    • biased towards multivalued attributes
  
  • **Gain ratio:**
    • tends to prefer unbalanced splits in which one partition is much smaller than the others (why?)
  
  • **Gini index:**
    • biased to multivalued attributes
**Other Attribute Selection Measures**

- **CHAID**: a popular decision tree algorithm, measure based on $\chi^2$ test for independence
- **C-SEP**: performs better than info. gain and gini index in certain cases
- **G-statistic**: has a close approximation to $\chi^2$ distribution
- **MDL (Minimal Description Length) principle** (i.e., the simplest solution is preferred):
  - The best tree as the one that requires the fewest # of bits to both (1) encode the tree, and (2) encode the exceptions to the tree
- Multivariate splits (partition based on multiple variable combinations)
  - **CART**: finds multivariate splits based on a linear comb. of attrs.
- Which attribute selection measure is the best?
  - Most give good results, none is significantly superior than others
Overfitting and Tree Pruning

- **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”
Matrix Data: Classification: Part 1

- Classification: Basic Concepts
- Decision Tree Induction
- Model Evaluation and Selection
- Summary
Model Evaluation and Selection

• Evaluation metrics: How can we measure accuracy? Other metrics to consider?

• Use validation test set of class-labeled tuples instead of training set when assessing accuracy

• Methods for estimating a classifier’s accuracy:
  • Holdout method, random subsampling
  • Cross-validation
Classifier Evaluation Metrics: Confusion Matrix

Confusion Matrix:

<table>
<thead>
<tr>
<th>Actual class \ Predicted class</th>
<th>$C_1$</th>
<th>$\neg C_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_1$</td>
<td>True Positives (TP)</td>
<td>False Negatives (FN)</td>
</tr>
<tr>
<td>$\neg C_1$</td>
<td>False Positives (FP)</td>
<td>True Negatives (TN)</td>
</tr>
</tbody>
</table>

Example of Confusion Matrix:

<table>
<thead>
<tr>
<th>Actual class \ Predicted class</th>
<th>$buy_{\text{computer}}$ = yes</th>
<th>$buy_{\text{computer}}$ = no</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>$buy_{\text{computer}}$ = yes</td>
<td>6954</td>
<td>46</td>
<td>7000</td>
</tr>
<tr>
<td>$buy_{\text{computer}}$ = no</td>
<td>412</td>
<td>2588</td>
<td>3000</td>
</tr>
<tr>
<td>Total</td>
<td>7366</td>
<td>2634</td>
<td>10000</td>
</tr>
</tbody>
</table>

- 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 Evaluation Metrics: Accuracy, Error Rate, Sensitivity and Specificity

- **Classifier Accuracy**, or recognition rate: percentage of test set tuples that are correctly classified
  \[
  \text{Accuracy} = \frac{(TP + TN)}{\text{All}}
  \]

- **Error rate**: \( 1 - \text{accuracy} \), or
  \[
  \text{Error rate} = \frac{(FP + FN)}{\text{All}}
  \]

- **Class Imbalance Problem**:
  - One class may be *rare*, e.g. fraud, or HIV-positive
  - Significant *majority of the negative class* and minority of the positive class

- **Sensitivity**: True Positive recognition rate
  - \( \text{Sensitivity} = \frac{TP}{P} \)

- **Specificity**: True Negative recognition rate
  - \( \text{Specificity} = \frac{TN}{N} \)
Classifier Evaluation Metrics: Precision and Recall, and F-measures

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

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

- Perfect score is 1.0

- Inverse relationship between precision & recall

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

- **$F_\beta$**: weighted measure of precision and recall
  - assigns $\beta$ times as much weight to recall as to precision
**Classifier Evaluation Metrics: Example**

<table>
<thead>
<tr>
<th>Actual Class \ Predicted class</th>
<th>cancer = yes</th>
<th>cancer = no</th>
<th>Total</th>
<th>Recognition(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>cancer = yes</td>
<td>90</td>
<td>210</td>
<td>300</td>
<td>30.00 (sensitivity)</td>
</tr>
<tr>
<td>cancer = no</td>
<td>140</td>
<td>9560</td>
<td>9700</td>
<td>98.56 (specificity)</td>
</tr>
<tr>
<td>Total</td>
<td>230</td>
<td>9770</td>
<td>10000</td>
<td>96.50 (accuracy)</td>
</tr>
</tbody>
</table>

- **Precision** = \( \frac{90}{230} = 39.13\% \)
- **Recall** = \( \frac{90}{300} = 30.00\% \)
Evaluating Classifier Accuracy: Holdout & Cross-Validation Methods

• 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
  • *Stratified cross-validation*: folds are stratified so that class dist. in each fold is approx. the same as that in the initial data
Model Selection: 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
- Area under the curve: the closer to the diagonal line (i.e., the closer the area is to 0.5), the less accurate is the model
Plotting an ROC Curve

- True positive rate: $TPR = TP/P$ (sensitivity)
- False positive rate: $FPR = FP/N$ (1-specificity)

- Rank tuples according to how likely they will be a positive tuple
  - Idea: when we include more tuples in, we are more likely to make mistakes, that is the trade-off!
  - Nice property: not threshold (cut-off) need to be specified, only rank matters
<table>
<thead>
<tr>
<th>Tuple #</th>
<th>Class</th>
<th>Prob.</th>
<th>TP</th>
<th>FP</th>
<th>TN</th>
<th>FN</th>
<th>TPR</th>
<th>FPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>p</td>
<td>0.9</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td>4</td>
<td>0.2</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>p</td>
<td>0.8</td>
<td>2</td>
<td>0</td>
<td>5</td>
<td>3</td>
<td>0.4</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>n</td>
<td>0.7</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>0.4</td>
<td>0.2</td>
</tr>
<tr>
<td>4</td>
<td>p</td>
<td>0.6</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>0.6</td>
<td>0.2</td>
</tr>
<tr>
<td>5</td>
<td>p</td>
<td>0.55</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>0.8</td>
<td>0.2</td>
</tr>
<tr>
<td>6</td>
<td>n</td>
<td>0.54</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0.8</td>
<td>0.4</td>
</tr>
<tr>
<td>7</td>
<td>n</td>
<td>0.53</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0.8</td>
<td>0.6</td>
</tr>
<tr>
<td>8</td>
<td>n</td>
<td>0.51</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>9</td>
<td>p</td>
<td>0.50</td>
<td>5</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>1.0</td>
<td>0.8</td>
</tr>
<tr>
<td>10</td>
<td>n</td>
<td>0.4</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

**Example**

![ROC curve diagram](image)
Matrix Data: Classification: Part 1

- Classification: Basic Concepts
- Decision Tree Induction
- Model Evaluation and Selection
- Summary
Summary

• **Classification** is a form of data analysis that extracts **models** describing important data classes.

• **decision tree induction**

• **Evaluation**
  
  • **Evaluation metrics** include: accuracy, sensitivity, specificity, precision, recall, $F$ measure, and $F_β$ measure.

  • **k-fold cross-validation** is recommended for accuracy estimation.

  • **ROC curves** are useful for model selection.

C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995


H. Cheng, X. Yan, J. Han, and C.-W. Hsu, *Discriminative Frequent Pattern Analysis for Effective Classification*, ICDE'07

H. Cheng, X. Yan, J. Han, and P. S. Yu, *Direct Discriminative Pattern Mining for Effective Classification*, ICDE'08

W. Cohen. *Fast effective rule induction*. ICML'95

G. Cong, K.-L. Tan, A. K. H. Tung, and X. Xu. *Mining top-k covering rule groups for gene expression data*. SIGMOD'05
References (2)

• U. M. Fayyad. *Branching on attribute values in decision tree generation*. AAAI’94.
• W. Li, J. Han, and J. Pei, *CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules*, ICDM’01.
References (3)


References (4)

- X. Yin and J. Han. CPAR: Classification based on predictive association rules. SDM'03.
- H. Yu, J. Yang, and J. Han. Classifying large data sets using SVM with hierarchical clusters. KDD'03.