ECT7110
Classification – Decision Trees

Prof. Wai Lam
Classification and Decision Tree

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Classification by decision tree induction
Classification vs. 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
  - E.g. categorize bank loan applications as either *safe* or *risky*.

- **Prediction:**
  - models *continuous-valued functions*, i.e., predicts unknown or missing values
  - E.g. predict the *expenditures* of potential customers on computer equipment given their income and occupation.

- **Typical Applications**
  - credit approval
  - target marketing
  - medical diagnosis
  - treatment effectiveness analysis
Classification—A Two-Step Process

- **Step1 (Model construction):** describing a predetermined set of data 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: *training set*
  - The individual tuples making up the training set are referred to as *training samples*
  - *Supervised learning*: Learning of the model with a given training set.
  - The learned model is represented as
    - classification rules
    - decision trees, or
    - mathematical formulae.
Classification—A Two-Step Process

- **Step 2 (Model usage):** the model is used for classifying future or unseen 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, the model is used to classify future data tuples with unknown class labels.
Classification Process (1): Model Construction

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>INCOME</th>
<th>CREDIT RATING</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mike</td>
<td>&lt;= 30</td>
<td>low</td>
<td>fair</td>
</tr>
<tr>
<td>Mary</td>
<td>&lt;= 30</td>
<td>low</td>
<td>poor</td>
</tr>
<tr>
<td>Bill</td>
<td>31..40</td>
<td>high</td>
<td>excellent</td>
</tr>
<tr>
<td>Jim</td>
<td>&gt;40</td>
<td>med</td>
<td>fair</td>
</tr>
<tr>
<td>Dave</td>
<td>&gt;40</td>
<td>med</td>
<td>fair</td>
</tr>
<tr>
<td>Anne</td>
<td>31..40</td>
<td>high</td>
<td>excellent</td>
</tr>
</tbody>
</table>

IF age = “31..40”
and income = high
THEN
credit rating = excellent
Classification Process (2): Use the Model in Prediction

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>INCOME</th>
<th>CREDIT RATING</th>
</tr>
</thead>
<tbody>
<tr>
<td>May</td>
<td>&lt;= 30</td>
<td>high</td>
<td>fair</td>
</tr>
<tr>
<td>Ana</td>
<td>31..40</td>
<td>low</td>
<td>poor</td>
</tr>
<tr>
<td>Wayne</td>
<td>&gt;40</td>
<td>high</td>
<td>excellent</td>
</tr>
<tr>
<td>Jack</td>
<td>&lt;=30</td>
<td>med</td>
<td>fair</td>
</tr>
</tbody>
</table>

Credit rating?

fair

Unseen Data

(John, 31..40, med)
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
Issues regarding Classification and Prediction (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
  - E.g. date of a bank loan application is not relevant
  - Improve the efficiency and scalability of data mining

- Data transformation
  - Data can be *generalized* to higher level concepts (concept hierarchy)
  - Data should be *normalized* when methods involving distance measurements are used in the learning step (e.g. neural network)
Issues regarding Classification and Prediction (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 (large amount of data)
- Interpretability:
  - understanding and insight provided by the model
- Goodness of rules
  - decision tree size
  - compactness of classification rules
Classification by Decision Tree Induction

- Decision tree
  - A flow-chart-like tree structure
  - *Internal node* denotes a test on an attribute
  - *Branch* represents an outcome of the test
  - *Leaf nodes* represent class labels or class distribution

- Use of decision tree: Classifying an unknown sample
  - Test the attribute values of the sample against the decision tree
An Example of a Decision Tree For “buys computer”

- **age?**
  - ≤30
    - student?
      - no
      - yes
  - 30..40
    - yes
    - no
  - >40
    - credit rating?
      - excellent
      - fair
        - no
        - yes
How to Obtain a Decision Tree?

- **Manual construction**
- **Decision tree induction:**
  Automatically discover a decision tree from data
  - Tree construction
    - At start, all the training examples are at the root
    - Partition examples recursively based on selected attributes
  - Tree pruning
    - Identify and remove branches that reflect noise or outliers
## Training Dataset

This follows an example from Quinlan’s ID3.

<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>30…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>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>
<tr>
<td>&gt;40</td>
<td>medium</td>
<td>no</td>
<td>excellent</td>
<td>no</td>
</tr>
</tbody>
</table>
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
Basic Algorithm for Decision Tree Induction

- If the samples are all of the same class, then the node becomes a leaf and is labeled with that class.
- Otherwise, it uses a statistical measure (e.g., information gain) for selecting the attribute that will best separate the samples into individual classes. This attribute becomes the “test” or “decision” attribute at the node.
- A branch is created for each known value of the test attribute, and the samples are partitioned accordingly.
- The algorithm uses the same process recursively to form a decision tree for the samples at each partition. Once an attribute has occurred at a node, it need not be considered in any of the node’s descendents.
Basic Algorithm for Decision Tree Induction

- The recursive partitioning stops only when any one of the following conditions is true:
  - All samples for a given node belong to the same class
  - There are no remaining attributes on which the samples may be further partitioned. In this case, *majority voting* is employed. This involves converting the given node into a leaf and labeling it with the class in *majority voting* among samples.
  - There are no samples for the branch *test-attribute=ai*. In this case, a leaf is created with the majority class in samples.
Algorithm: Generate_decision_tree. Generate a decision tree from the given training data.

Input: The training samples, samples, represented by discrete-valued attributes; the set of candidate attributes, attribute-list.

Output: A decision tree.

Method:

1. create a node N;
2. if samples are all of the same class, C then
3. return N as a leaf node labeled with the class C;
4. if attribute-list is empty then
5. return N as a leaf node labeled with the most common class in samples; // majority voting
6. select test-attribute, the attribute among attribute-list with the highest information gain;
7. label node N with test-attribute;
8. for each known value a_i of test-attribute // partition the samples
9. grow a branch from node N for the condition test-attribute = a_i;
10. let s_i be the set of samples in samples for which test-attribute = a_i; // a partition
11. if s_i is empty then
12. attach a leaf labeled with the most common class in samples;
13. else attach the node returned by Generate_decision_tree(s_i, attribute-list−test-attribute);
Attribute Selection by Information Gain Computation

Consider the attribute age:

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

\[ \text{Gain(age)} = 0.246 \]

Consider other attributes in a similar way:

\[ \text{Gain(income)} = 0.029 \]
\[ \text{Gain(student)} = 0.151 \]
\[ \text{Gain(credit\_rating)} = 0.048 \]
Learning (Constructing) a Decision Tree

age?

$\leq 30$  
$30..40$  
$>40$
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”**
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
  - comparable classification accuracy with other methods
Presentation of Classification Results