Lecture notes

Knowledge Discovery in Databases
Summer Semester 2012

Lecture 4: Classification

Lecture: Dr. Eirini Ntoutsi
Tutorials: Erich Schubert

http://www.dbs.ifi.lmu.de/cms/Knowledge_Discovery_in_Databases_I_(KDD_I)
Sources

• Previous KDD I lectures on LMU (Johannes Aßfalg, Christian Böhm, Karsten Borgwardt, Martin Ester, Eshref Januzaj, Karin Kailing, Peer Kröger, Jörg Sander, Matthias Schubert, Arthur Zimek)

• Jiawei Han, Micheline Kamber and Jian Pei, *Data Mining: Concepts and Techniques, 3rd ed.*, Morgan Kaufmann, 2011.


• Wikipedia
• Introduction

• The classification process

• Classification (supervised) vs clustering (unsupervised)

• Decision trees

• Evaluation of classifiers

• Things you should know

• Homework/tutorial
Given:

- a dataset $D = \{t_1, t_2, \ldots, t_n\}$ and
- a set of classes $C = \{c_1, \ldots, c_k\}$

the classification problem is to define a mapping $f : D \rightarrow C$ where each $t_i$ is assigned to one class $c_j$.

**Classification**

- predicts categorical (discrete, unordered) class labels
- Constructs a model (classifier) based on a training set
- Uses this model to predict the class label for new unknown-class instances

**Prediction**

- is similar, but may be viewed as having infinite number of classes
- more on prediction in next lectures
A simple classifier:

- **if** Alter > 50
- **if** Alter ≤ 50 **and** Autotyp=LKW
- **if** Alter ≤ 50 **and** Autotyp ≠ LKW

<table>
<thead>
<tr>
<th>ID</th>
<th>Alter</th>
<th>Autotyp</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>23</td>
<td>Familie</td>
<td>high</td>
</tr>
<tr>
<td>2</td>
<td>17</td>
<td>Sport</td>
<td>high</td>
</tr>
<tr>
<td>3</td>
<td>43</td>
<td>Sport</td>
<td>high</td>
</tr>
<tr>
<td>4</td>
<td>68</td>
<td>Familie</td>
<td>low</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>LKW</td>
<td>low</td>
</tr>
</tbody>
</table>
Applications

- Credit approval
  - Classify bank loan applications as e.g. safe or risky.
- Fraud detection
  - e.g., in credit cards
- Churn prediction
  - E.g., in telecommunication companies
- Target marketing
  - Is the customer a potential buyer for a new computer?
- Medical diagnosis
- Character recognition
- ...

Knowledge Discovery in Databases I: Classification
Outline

• Introduction

• The classification process

• Classification (supervised) vs clustering (unsupervised)

• Decision trees

• Evaluation of classifiers

• Things you should know

• Homework/tutorial
Classification techniques

• Typical approach:
  – Create specific model by evaluating training data (or using domain experts’ knowledge).
    • Assess the quality of the model
  – Apply model developed to new data.

• Classes must be predefined!!!

• Many techniques
  – Decision trees
  – Naïve Bayes
  – kNN
  – Neural Networks
  – Support Vector Machines
  – ....
Classification technique (detailed)

- **Model construction**: describing a set of predetermined classes
  - The set of tuples used for model construction is **training set**
  - Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
  - The **model** is represented as classification rules, decision trees, or mathematical formulae

- **Model evaluation**: estimate accuracy of the model
  - The set of tuples used for model evaluation is **test set**
  - The class label of each tuple/sample in the test set is known in advance
  - 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 will occur

- **Model usage**: for classifying future or unknown objects
  - If the accuracy is acceptable, use the model to classify data tuples whose class labels are not known

### Training set

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

### Test set

<table>
<thead>
<tr>
<th>NAME</th>
<th>RANK</th>
<th>YEARS</th>
<th>TENURED</th>
<th>PREDICTED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maria</td>
<td>Assistant Prof</td>
<td>3</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>John</td>
<td>Associate Prof</td>
<td>7</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Franz</td>
<td>Professor</td>
<td>3</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NAME</th>
<th>RANK</th>
<th>YEARS</th>
<th>TENURED</th>
<th>PREDICTED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jeff</td>
<td>Professor</td>
<td>4</td>
<td>?</td>
<td>yes</td>
</tr>
<tr>
<td>Patrick</td>
<td>Associate Prof</td>
<td>8</td>
<td>?</td>
<td>yes</td>
</tr>
<tr>
<td>Maria</td>
<td>Associate Prof</td>
<td>2</td>
<td>?</td>
<td>no</td>
</tr>
</tbody>
</table>

Class attribute: tenured={yes, no}
Model construction

<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>

Class attribute

- IF rank = ‘professor’ OR years > 6
  THEN tenured = ‘yes’

- IF (rank! = ‘professor’) AND (years < 6)
  THEN tenured = ‘no’

Attributes

Training Data

Classifier (Model)
Model evaluation

**Database Systems Group**

**Knowledge Discovery in Databases I: Classification**

1. **Classifier (Model)**
   - If rank = 'professor' OR years > 6
     THEN tenured = 'yes'
   - If (rank != 'professor') AND (years < 6)
     THEN tenured = 'no'

2. **Testing Data**

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

**Classifier quality**

Is it acceptable?
Model usage for prediction

**Classification Algorithms**

**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>

**Classifier (Model)**

\[
\text{IF (rank} = \text{‘professor’}) \text{ OR (years} > 6 \text{) THEN tenured} = \text{‘yes’} \\
\text{IF (rank}! = \text{‘professor’}) \text{ AND (years} < 6 \text{) THEN tenured} = \text{‘no’}
\]

**Unseen Data**

<table>
<thead>
<tr>
<th>NAME</th>
<th>RANK</th>
<th>YEARS</th>
<th>TENURED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jeff</td>
<td>Professor</td>
<td>4</td>
<td>?</td>
</tr>
<tr>
<td>Patrick</td>
<td>Assistant Prof</td>
<td>8</td>
<td>?</td>
</tr>
<tr>
<td>Maria</td>
<td>Assistant Prof</td>
<td>2</td>
<td>?</td>
</tr>
</tbody>
</table>

Knowledge Discovery in Databases I: Classification
Outline

• Introduction

• The classification process

• Classification (supervised) vs clustering (unsupervised)

• Decision trees

• Evaluation of classifiers

• Things you should know

• Homework/tutorial
A supervised learning task

• Classification is a \textit{supervised} learning task
  – Supervision: The training data (observations, measurements, etc.) are accompanied by \textit{labels} indicating the \textit{class} of the observations
  – New data is classified based on the training set

• Clustering is an \textit{unsupervised} learning task
  – The class labels of training data is unknown
  – Given a set of measurements, observations, etc., the goal is to group the data into groups of similar data (clusters)
Supervised learning example

**Classification model**

- **Screw**
- **Nails**
- **Paper clips**

**New object (unknown class)**

**Question:**
What is the class of a new object???
Is it a screw, a nail or a paper clip?
Unsupervised learning example

**Clustering**

Cluster 1: paper clips
Cluster 2: nails

**Question:**
Is there any structure in data (based on their characteristics, i.e., width, height)?
Classification techniques

• Statistical methods
  – Bayesian classifiers etc

• Partitioning methods
  – Decision trees etc

• Similarity based methods
  – K-Nearest Neighbors etc
Outline

• Introduction

• The classification process

• Classification (supervised) vs clustering (unsupervised)

• Decision trees

• Evaluation of classifiers

• Things you should know

• Homework/tutorial
Decision trees (DTs)

• One of the most popular classification methods

• DTs are included in many commercial systems nowadays

• Easy to interpret, human readable, intuitive

• Simple and fast methods

• Partition based method: Partitions the space into rectangular regions

• Many algorithms have been proposed
  – ID3 (Quinlan 1986), C4.5 (Quinlan 1993), CART (Breiman et al 1984)....
Decision tree for the “play tennis” problem

### Training set

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D6</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D7</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D8</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D9</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>
• Representation
  – Each *internal node* specifies a test of some attribute of the instance
  – Each *branch* descending from a node corresponds to one of the possible values for this attribute
  – Each *leaf node* assigns a class label
• Decision trees classify instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance
• Decision trees represent a disjunction of conjunctions of constraints on the attribute values of the instances
• Each path from the root to a leaf node, corresponds to a conjunction of attribute tests
• The whole tree corresponds to a disjunction of these conjunctions
• We can “translate” each path into IF-THEN rules (human readable)

IF \((\text{Outlook} = \text{Sunny}) \land (\text{Humidity} = \text{Normal}))\), THEN (Play tennis=Yes)

IF \((\text{Outlook} = \text{Rain}) \land (\text{Wind} = \text{Weak}))\), THEN (Play tennis=Yes)
Basic algorithm (ID3, Quinlan 1986)

– Tree is constructed in a top-down recursive divide-and-conquer manner
– At start, all the training examples are at the root node
– The question is “which attribute should be tested at the root?”
  • Attributes are evaluated using some statistical measure, which determines how well each attribute alone classifies the training examples
  • The best attribute is selected and used as the test attribute at the root
– For each possible value of the test attribute, a descendant of the root node is created and the instances are mapped to the appropriate descendant node.
– The procedure is repeated for each descendant node, so instances are partitioned recursively.

When do we stop 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
• **Algorithm cont’**

  **Pseudocode**

  Main loop:
  1. $A \leftarrow$ the “best” decision attribute for next node
  2. Assign $A$ as decision attribute for node
  3. For each value of $A$, create new descendant of node
  4. Sort training examples to leaf nodes
  5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

  **But, …. which attribute is the best?**

  • The goal is to select the attribute that is most *useful* for classifying examples.

  • By useful we mean that the resulting partitioning is as *pure* as possible.

  • A partition is *pure* if all its instances belong to the same class.
Attribute selection measure: Information gain

- Used in ID3
- It uses entropy, a measure of pureness of the data
- The information gain $\text{Gain}(S, A)$ of an attribute $A$ relative to a collection of examples $S$ measures the gain reduction in $S$ due to splitting on $A$:

$$
\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)
$$

- Gain measures the expected reduction in entropy due to splitting on $A$
- The attribute with the higher entropy reduction is chosen
Entropy

- Let S be a collection of positive and negative examples for a binary classification problem, C={+, -}.
- \( p_+ \): the percentage of positive examples in S
- \( p_- \): the percentage of negative examples in S
- Entropy measures the impurity of S:

\[
Entropy(S) = -p_+ \log_2(p_+) - p_- \log_2(p_-)
\]

- Examples:
  - Let S: [9+, 5-]  
    \[
    Entropy(S) = -\frac{9}{14} \log_2\left(\frac{9}{14}\right) - \frac{5}{14} \log_2\left(\frac{5}{14}\right) = 0.940
    \]
  - Let S: [7+, 7-]  
    \[
    Entropy(S) = -\frac{7}{14} \log_2\left(\frac{7}{14}\right) - \frac{7}{14} \log_2\left(\frac{7}{14}\right) = 1
    \]
  - Let S: [14+, 0-]  
    \[
    Entropy(S) = -\frac{14}{14} \log_2\left(\frac{14}{14}\right) - 0 \log_2\left(\frac{0}{14}\right) = 0
    \]

- Entropy = 0, when all members belong to the same class
- Entropy = 1, when there is an equal number of positive and negative examples

in the general case (k-classification problem)

\[
Entropy(S) = \sum_{i=1}^{k} -p_i \log_2(p_i)
\]
Information Gain example 1

Which attribute to choose next???

Gain (S, Humidity )
= .940 - (7/14).985 - (7/14).592
= .151

Gain (S, Wind)
= .940 - (8/14).811 - (6/14)1.0
= .048
Information Gain example 2

Training set

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D6</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D7</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D8</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D9</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>

Which attribute should be tested here?

\[
S_{\text{Sunny}} = \{ \text{D1, D2, D8, D9, D11} \}
\]

\[
\text{Gain (} S_{\text{Sunny}}, \text{Humidity} \) = .970 - (3/5) 0.0 - (2/5) 0.0 = .970
\]

\[
\text{Gain (} S_{\text{Sunny}}, \text{Temperature} \) = .970 - (2/5) 0.0 - (2/5) 1.0 - (1/5) 0.0 = .570
\]

\[
\text{Gain (} S_{\text{Sunny}}, \text{Wind} \) = .970 - (2/5) 1.0 - (3/5) .918 = .019
\]
Attribute selection measure: Gain ratio

- Information gain is biased towards attributes with a large number of values
  - Consider the attribute ID (unique identifier)
- C4.5 (a successor of ID3) uses gain ratio to overcome the problem, which normalizes the gain

\[
\text{GainRatio}(S, A) = \frac{\text{Gain}(S, A)}{\text{SplitInformation}(S, A)}
\]

- Example:
  - Humidity=\{High, Low\}
    \[
    \text{SplitInformation}(S, \text{Humidity}) = -\frac{7}{14} \times \log_2\left(\frac{7}{14}\right) - \frac{7}{14} \times \log_2\left(\frac{7}{14}\right) = 1
    \]
  - Wind=\{Weak, Strong\}
    \[
    \text{SplitInformation}(S, \text{Wind}) = -\frac{8}{14} \times \log_2\left(\frac{8}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) = 0.9852
    \]
  - Outlook = \{Sunny, Overcast, Rain\}
    \[
    \text{SplitInformation}(S, \text{Outlook}) = -\frac{5}{14} \times \log_2\left(\frac{5}{14}\right) - \frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{5}{14} \times \log_2\left(\frac{5}{14}\right) = 1.5774
    \]
- The attribute with the maximum gain ratio is selected as the splitting attribute
Attribute selection measure: Gini Index (CART)

- Let a dataset $S$ containing examples from $k$ classes. Let $p_j$ be the probability of class $j$ in $S$. The Gini Index of $S$ is given by:

$$Gini(S) = 1 - \sum_{j=1}^{k} p_j^2$$

- Gini index considers a binary split for each attribute.

- If $S$ is split based on attribute $A$ into two subsets $S_1$ and $S_2$:

$$Gini(S, A) = \frac{|S_1|}{|S|} Gini(S_1) + \frac{|S_2|}{|S|} Gini(S_2)$$

- Reduction in impurity:

$$\Delta Gini(S, A) = Gini(S) - Gini(S, A)$$

- The attribute $A$ that provides the smallest $Gini(S, A)$ (or the largest reduction in impurity) is chosen to split the node.

- How to find the binary splits?
  - For discrete-valued attributes, we consider all possible subsets that can be formed by values of $A$.
  - For numerical attributes, we find the split points (slides 41-42).
Gini index example

Let $S$ has 9 tuples in `buys_computer = "yes"` and 5 in `"no"

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

Suppose the attribute income partitions $S$ into 10 in $S_1$: \{low, medium\} and 4 in $S_2$

\[
gini_{\text{income} \in \{\text{low, medium}\}}(D) = \left( \frac{10}{14} \right) Gini(D_1) + \left( \frac{4}{14} \right) Gini(D_1)
\]
\[
= \frac{10}{14} \left( 1 - \left( \frac{6}{10} \right)^2 - \left( \frac{4}{10} \right)^2 \right) + \frac{4}{14} \left( 1 - \left( \frac{1}{4} \right)^2 - \left( \frac{3}{4} \right)^2 \right)
\]
\[
= 0.450
\]
\[
= Gini_{\text{income} \in \{\text{high}\}}(D)
\]

The Gini Index measures of the remaining partitions for the income attribute:

\[
Gini_{\{\text{low, high}\} \text{ and } \{\text{medium}\}}(D) = 0.315
\]
\[
Gini_{\{\text{medium, high}\} \text{ and } \{\text{low}\}}(D) = 0.300
\]

So, the best binary split for income is on \{medium, high\} and \{low\}
Comparing Attribute Selection Measures

• The three measures, are commonly used and in general, return good results but
  – Information gain Gain(S,A):
    • biased towards multivalued attributes
  – Gain ratio GainRatio(S,A):
    • tends to prefer unbalanced splits in which one partition is much smaller than the others
  – Gini index:
    • biased to multivalued attributes
    • has difficulty when # of classes is large
    • tends to favor tests that result in equal-sized partitions and purity in both partitions
• Several other measures exist
Hypothesis search space (by ID3)

- Hypothesis space is complete
  - Solution is surely in there
- Greedy approach
- No backtracking
  - Local minima
- Outputs a single hypothesis
Space partitioning

- Decision boundary: The border line between two neighboring regions of different classes

- Decision regions: Axis parallel hyper-rectangles
Comparing DTs/ partitionings

Knowledge Discovery in Databases I: Classification
Consider adding a *noisy* training example $D_{15}$ to the training set
How the earlier tree (built upon $D_1$-$D_{14}$) would be effected?
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

- Overfitting: Consider an hypothesis $h$
  - $\text{error}_{\text{train}}(h)$: the error of $h$ in training set
  - $\text{error}_{\text{D}}(h)$: the error of $h$ in the entire distribution $D$ of data
  - Hypothesis $h$ overfits training data if there is an alternative hypothesis $h'$ in $H$ such that:

\[
\text{error}_{\text{train}}(h) < \text{error}_{\text{train}}(h') \\
\text{and} \\
\text{error}_{\text{D}}(h) > \text{error}_{\text{D}}(h')
\]
Overfitting
Avoiding overfitting

• 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”
      – Test set
Effect of pruning

![Graph showing the effect of pruning on accuracy vs size of tree](image-url)
Dealing with continuous-valued attributes

- Let attribute A be a continuous-valued attribute.
- Must determine the best split point \( t \) for A, \((A \leq t)\)
  - Sort the value A in increasing order.
  - Identify adjacent examples that differ in their target classification.
    - Typically, every such pair suggests a potential split threshold \( t = (a_i + a_{i+1})/2 \)
  - Select threshold \( t \) that yields the best value of the splitting criterion.

| Temperature: 40 48 60 72 80 90 |
| PlayTennis: No No Yes Yes Yes No |

\[
t = (48 + 60)/2 = 54
\]
\[
t = (80 + 90)/2 = 85
\]

2 potential thresholds: \( \text{Temperature} > 54 \), \( \text{Temperature} > 85 \)
Compute the attribute selection measure (e.g. information gain) for both
Choose the best (\( \text{Temperature} > 54 \) here)
Continuous-valued attributes cont’

- Let \( t \) be the threshold chosen from the previous step
- Create a boolean attribute based on \( A \) and threshold \( t \) with two possible outcomes: yes, no
  - \( S_1 \) is the set of tuples in \( S \) satisfying \( (A > t) \), and \( S_2 \) is the set of tuples in \( S \) satisfying \( (A \leq t) \)

How it looks

An example of a tree for the play tennis problem when attributes Humidity and Wind are continuous
When to consider decision trees

• Instances are represented by attribute-value pairs
  – Instances are represented by a fixed number of attributes, e.g. outlook, humidity, wind and their values, e.g. (wind=strong, outlook =rainy, humidity=normal)
  – The easiest situation for a DT is when attributes take a small number of disjoint possible values, e.g. wind={strong, weak}
  – There are extensions for numerical attributes also, e.g. temperature, income.

• The class attribute has discrete output values
  – Usually binary classification, e.g. {yes, no}, but also for more class values, e.g. {pos, neg, neutral}

• The training data might contain errors
  – DTs are robust to errors: both errors in the class values of the training examples and in the attribute values of these examples

• The training data might contain missing values
  • DTs can be used even when some training examples have some unknown attribute values
Outline

• Introduction

• The classification process

• Classification (supervised) vs clustering (unsupervised)

• Decision trees

• Evaluation of classifiers

• Things you should know

• Homework/tutorial
Classifier evaluation

• The quality of a classifier is evaluated over a test set, different from the training set
• For each instance in the test set, we know its true class label
• Compare the predicted class (by some classifier) with the true class of the test instances
• Terminology
  – Positive tuples: tuples of the main class of interest
  – Negative tuples: all other tuples

• A useful tool for analyzing how well a classifier performs is the confusion matrix
• For an m-class problem, the matrix is of size m x m
• An example of a matrix for a 2-class problem:

<table>
<thead>
<tr>
<th>Actual class</th>
<th>Predicted class</th>
</tr>
</thead>
<tbody>
<tr>
<td>C₁</td>
<td>C₂</td>
</tr>
<tr>
<td>C₁</td>
<td>TP (true positive)</td>
</tr>
<tr>
<td>C₂</td>
<td>FP (false positive)</td>
</tr>
<tr>
<td>Totals</td>
<td>P’</td>
</tr>
</tbody>
</table>

Knowledge Discovery in Databases I: Classification
Classifier evaluation measures

- **Accuracy/ Recognition rate:**
  \[
  \text{accuracy}(M) = \frac{TP + TN}{P + N}
  \]
  \%
  of test set instances correctly classified

- **Error rate/ Missclassification rate:**
  \[
  \text{error_rate}(M) = 1 - \text{accuracy}(M)
  \]

  More effective when the class distribution is relatively balanced

<table>
<thead>
<tr>
<th>classes</th>
<th>buy_computer = yes</th>
<th>buy_computer = no</th>
<th>total</th>
<th>recognition(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>buy_computer = yes</td>
<td>6954</td>
<td>46</td>
<td>7000</td>
<td>95.42</td>
</tr>
<tr>
<td>buy_computer = no</td>
<td>412</td>
<td>2588</td>
<td>3000</td>
<td></td>
</tr>
<tr>
<td>total</td>
<td>7366</td>
<td>2634</td>
<td>10000</td>
<td></td>
</tr>
</tbody>
</table>
If classes are imbalanced:

• **Sensitivity/ True positive rate/ recall:**
  % of positive tuples that are correctly recognized

\[
\text{sensitivity}(M) = \frac{TP}{P}
\]

• **Specificity/ True negative rate:** % of negative tuples that are correctly recognized

\[
\text{specificity}(M) = \frac{TN}{N}
\]

---

<table>
<thead>
<tr>
<th>classes</th>
<th>buy_computer = yes</th>
<th>buy_computer = no</th>
<th>total</th>
<th>recognition(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>buy_computer = yes</td>
<td>6954</td>
<td>46</td>
<td>7000</td>
<td>99.34</td>
</tr>
<tr>
<td>buy_computer = no</td>
<td>412</td>
<td>2588</td>
<td>3000</td>
<td>86.27</td>
</tr>
<tr>
<td>total</td>
<td>7366</td>
<td>2634</td>
<td>10000</td>
<td>95.42</td>
</tr>
</tbody>
</table>
Classifier evaluation measures cont’

- **Precision**: % of tuples labeled as positive which are actually positive

\[
\text{precision}(M) = \frac{TP}{TP + FP}
\]

- **Recall**: % of positive tuples labeled as positive

\[
\text{recall}(M) = \frac{TP}{TP + FN} = \frac{TP}{P}
\]

- Precision does not say anything about misclassified instances
- Recall does not say anything about possible instances from other classes labeled as positive

- **F-measure/ F$_1$ score/F-score** combines both

\[
F(M) = \frac{2 \cdot \text{precision}(M) \cdot \text{recall}(M)}{\text{precision}(M) + \text{recall}(M)}
\]

*It is the harmonic mean of precision and recall*

- **F$_\beta$-measure** is a weighted measure of precision and recall

\[
F_\beta(M) = \frac{(1 + \beta^2) \cdot \text{precision}(M) \cdot \text{recall}(M)}{\beta^2 \cdot \text{precision}(M) + \text{recall}(M)}
\]

*Common values for $\beta$:
$\beta=2$
$\beta=0.5$
Classifier evaluation 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
  – It takes no longer to compute (+)
  – It depends on how data are divided (-)
  – Random sampling: a variation of holdout
    • Repeat holdout k times, accuracy is the avg accuracy obtained
Classifier evaluation methods cont’

• **Cross-validation** \((k\)-fold cross validation, \(k = 10\) usually)
  - Randomly partition the data into \(k\) mutually exclusive subsets \(D_1, \ldots, D_k\) each approximately equal size
  - Training and testing is performed \(k\) times
    • At the \(i\)-th iteration, use \(D_i\) as test set and others as training set
  - Accuracy is the avg accuracy over all iterations
  - Does not rely so much on how data are divided (+)
  - The algorithm should re-run from scratch \(k\) times (-)

  – **Leave-one-out**: \(k\) folds where \(k = \#\)of tuples, so only one sample is used as a test set at a time; for small sized data

  – **Stratified cross-validation**: folds are stratified so that class distribution in each fold is approximately the same as that in the initial data
    • Stratified 10 fold cross-validation is recommended
Classifier evaluation methods cont’

- **Bootstrap**: Samples the given training data 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
  - Works well with small data sets
- **Several bootstrap methods, and a common one is .632 bootstrap**
  - Suppose we are given a data set of \(d\) tuples.
  - The data set is sampled \(d\) times, with replacement, resulting in a training set of \(d\) samples (also known as bootstrap sample):
    - It is very likely that some of the original tuples will occur more than once in this set
    - The data tuples that did not make it into the training set end up forming the test set.
    - On average, 36.8 of the tuples will not be selected for training and thereby end up in the test set; the remaining 63.2 will form the train test
      - Each sample has a probability \(1/d\) of being selected and \((1-1/d)\) of not being chosen. We repeat \(d\) times, so the probability for a tuple to not be chosen during the whole period is \((1-1/d)^d\).
      - For large \(d\):
        \[
        \left(1 - \frac{1}{n}\right)^n \approx e^{-1} \approx 0.368
        \]
    - Repeat the sampling procedure \(k\) times, report the overall accuracy of the model:

\[
acc(M) = \sum_{i=1}^{k} (0.632 \times acc(M_i)_{test\_set} + 0.368 \times acc(M_i)_{train\_set})
\]

Accuracy of the model obtained by bootstrap sample \(i\) when it is applied on test set \(i\).
Accuracy of the model obtained by bootstrap sample \(i\) when it is applied over all cases.
Classifier evaluation summary

- Accuracy measures
  - accuracy, error rate, sensitivity, specificity, precision, F-score, $F_\beta$

- Other parameters
  - Speed (construction time, usage time)
  - Robustness to noise, outliers and missing values
  - Scalability for large data sets
  - Interpretability from humans
Things you should know

- What is classification
- Class attribute, attributes
- Train set, test set, new unknown instances
- Supervised vs unsupervised
- Decision tree induction algorithm
- Choosing the best attribute for splitting
- Overfitting
- Dealing with continuous attributes
- Evaluation of classifiers
**Tutorial:** No tutorial this Thursday (Christi Himmelfahrt)
- Repeat exercises from the previous tutorials
- Get familiar with Weka/ Elki/ R/ SciPy.

**Homework:**
- Run decision tree classification in Weka
- Implement a decision tree classifier 😊

**Suggested reading:**