Decision Tree Learning

- Decision tree representation
- ID3 learning algorithm
- Entropy, information gain
- Overfitting
Decision Tree Representation

- A decision tree is a tree
- Each internal node tests an attribute
- Each branch corresponds to an attribute value
- Each leaf node assigns a class
# A Learning Problem: *PlayTennis*

<table>
<thead>
<tr>
<th>No</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Windy</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>false</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>true</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>overcast</td>
<td>hot</td>
<td>high</td>
<td>false</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>rain</td>
<td>mild</td>
<td>high</td>
<td>false</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>rain</td>
<td>cool</td>
<td>normal</td>
<td>false</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>rain</td>
<td>cool</td>
<td>normal</td>
<td>true</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>overcast</td>
<td>cool</td>
<td>normal</td>
<td>true</td>
<td>Yes</td>
</tr>
<tr>
<td>8</td>
<td>sunny</td>
<td>mild</td>
<td>high</td>
<td>false</td>
<td>No</td>
</tr>
<tr>
<td>9</td>
<td>sunny</td>
<td>cool</td>
<td>normal</td>
<td>false</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>rain</td>
<td>mild</td>
<td>normal</td>
<td>false</td>
<td>Yes</td>
</tr>
<tr>
<td>11</td>
<td>sunny</td>
<td>mild</td>
<td>normal</td>
<td>true</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>overcast</td>
<td>mild</td>
<td>high</td>
<td>true</td>
<td>Yes</td>
</tr>
<tr>
<td>13</td>
<td>overcast</td>
<td>hot</td>
<td>normal</td>
<td>false</td>
<td>Yes</td>
</tr>
<tr>
<td>14</td>
<td>rain</td>
<td>mild</td>
<td>high</td>
<td>true</td>
<td>No</td>
</tr>
</tbody>
</table>
Decision Tree for *PlayTennis*

```
Outlook
  Sunny
  Overcast
    Humidity
      High
        No
      Normal
        Yes
    Rain
      Wind
        Strong
          No
        Weak
          Yes
```
Decision Tree Learning

1. Select the “best” decision attribute $A$ for next node
2. For each value $v_i$ ($i = 1, \ldots, n$) of $A$, create a child node $N_i$
3. Partition the training examples in $T$ into $n$ subsets $T_1 \ldots T_n$
   according to $v_1 \ldots v_n$.
4. For each child node $N_i$, if all training examples in $T_i$ are
   from the same class, mark the $N_i$ as the class. Otherwise,
   recursively apply the algorithm to $N_i$ with attribute $A$
   removed.
5. If all nodes are marked as a class, stop.
Attribute Selection

The attribute selected should be maximally distinguish examples of one class from other classes.

Example: 3 attributes x1, x2, and x3
   two examples from class 1:
       (0 1 1) and (0 0 1)
   two examples from class 2:
       (1 1 1) and (1 0 0)
Which attribute, x1, x2, or x3, should be selected?
Entropy

Entropy quantifies the amount of information in a sample of training examples, S. Entropy measures the impurity of S.

\[ E(S) = - \sum_{i=1}^{n} p_i \log_2 p_i \]

\(p_i\) is the probability (proportion) of examples from the ith class in S

n is the number of classes
Information Gain

Expected reduction in entropy due to the selection of an attribute, A.

\[
Gain(S, A) = E(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} E(S_v)
\]

Select the attribute which maximizes the information gain.
Overfitting in Decision Trees

The decision tree generated fits the training examples very well, but it doesn’t perform well on unseen examples.

Avoiding Overfitting:

pre-prune:
  stop growing when data split not statistically significant

post-prune:
  grow full tree, then post-prune
Continuous Valued Attributes

Partition the continuous valued attribute to form a discrete attribute.

Example:
create two values for temperature
temperature < 72 and temperature > 72

There are many different methods to partition a continuous attribute, for example, information gain.
Instance-Based Learning (Nearest Neighbor)

Learning:
store all training examples with their classifications

Classification (Nearest Neighbor):
Given a new example $x$, first find the nearest training example $x_n$, then classify $x$ to the class of $x_n$.

Classification (k-Nearest Neighbor):
Given a new example $x$, take vote among its $k$ nearest neighbors.
Instance-Based Learning

• Advantages:
  – learning is very fast
  – learn complex classes
  – don’t lose information

• Disadvantages:
  – slow at classification time
  – has problem with many irrelevant attributes
Attribute Weights

To solve the problem of irrelevant attributes assign a weigh to each attribute and important attributes get large weights.

attribute weights are used in distance measure

attribute are automatically learned.