ID3 Decision Tree Induction Algorithm

- ID3 attempts to induce concepts from examples
- It does this by building decision trees that will correctly classify examples
- What makes ID3 interesting is the way it uses information theory to build the decision tree
- Information theory itself predates modern computers; a seminal work was published by Shannon in 1948
- Shannon formally measures the information content of message; this theory has many applications in telecommunications

Data for Credit History

<table>
<thead>
<tr>
<th>NO.</th>
<th>RISK</th>
<th>CREDIT HISTORY</th>
<th>DEBT</th>
<th>COLLATERAL</th>
<th>INCOME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>high</td>
<td>bad</td>
<td>high</td>
<td>none</td>
<td>$0 to $15k</td>
</tr>
<tr>
<td>2.</td>
<td>high</td>
<td>unknown</td>
<td>high</td>
<td>none</td>
<td>$15 to $35k</td>
</tr>
<tr>
<td>3.</td>
<td>moderate</td>
<td>unknown</td>
<td>low</td>
<td>none</td>
<td>$15 to $35k</td>
</tr>
<tr>
<td>4.</td>
<td>high</td>
<td>unknown</td>
<td>low</td>
<td>none</td>
<td>$0 to $15k</td>
</tr>
<tr>
<td>5.</td>
<td>low</td>
<td>unknown</td>
<td>low</td>
<td>none</td>
<td>over $35k</td>
</tr>
<tr>
<td>6.</td>
<td>low</td>
<td>unknown</td>
<td>low</td>
<td>adequate</td>
<td>over $35k</td>
</tr>
<tr>
<td>7.</td>
<td>high</td>
<td>bad</td>
<td>low</td>
<td>none</td>
<td>$0 to $15k</td>
</tr>
<tr>
<td>8.</td>
<td>moderate</td>
<td>bad</td>
<td>low</td>
<td>adequate</td>
<td>over $35k</td>
</tr>
<tr>
<td>9.</td>
<td>low</td>
<td>good</td>
<td>low</td>
<td>none</td>
<td>over $35k</td>
</tr>
<tr>
<td>10.</td>
<td>low</td>
<td>good</td>
<td>high</td>
<td>adequate</td>
<td>over $35k</td>
</tr>
<tr>
<td>11.</td>
<td>high</td>
<td>good</td>
<td>high</td>
<td>none</td>
<td>$0 to $15k</td>
</tr>
<tr>
<td>12.</td>
<td>moderate</td>
<td>good</td>
<td>high</td>
<td>none</td>
<td>$15 to $35k</td>
</tr>
<tr>
<td>13.</td>
<td>low</td>
<td>good</td>
<td>high</td>
<td>none</td>
<td>over $35k</td>
</tr>
<tr>
<td>14.</td>
<td>high</td>
<td>bad</td>
<td>high</td>
<td>none</td>
<td>$15 to $35k</td>
</tr>
</tbody>
</table>
A Decision Tree for Credit Risk

Comments on the Decision Tree

• Nodes in the tree
  – Internal nodes are decisions where the subtrees represent different branches based on values
  – The leaf nodes represent the classifications
  – The tree may not include all of the properties in the table, some properties may not be required to make a classification
  – Even if some properties are skipped, every example data must be correctly classified

• Tree size can vary depending on the order of tests
  – ID3 uses information theoretic test selection
  – The next slide shows a simpler tree by reordering tests
Building the Tree

- The tree is built in a top down fashion
  - Once a test is determine, the subtrees represent the disjoint subsets of values for that test
  - This continues until all members of the subset are in the same class; this becomes the leaf node
  - The algorithm on the next page creates the tree, assuming the order of decisions is known

- Determining a proper order for the decision is integral to ID3 and a difficult part of ID3
- This is done on an information theoretic basis, as will be described shortly
Inductive, top-down algorithm

function induce_tree (example_set, Properties)
begin
if all entries in example_set are in the same class
then return a leaf node labeled with that class
else if Properties is empty
then return leaf node labeled with disjunction of all classes in example_set
else begin
select a property, P, and make it the root of the current tree;
delete P from Properties;
for each value, V, of P,
begin
create a branch of the tree labeled with V;
let partition, be elements of example_set with values V for property P;
call induce_tree(partition, Properties), attach result to branch V
end
end
end

First Stages of the Construction

```
Income?
/    \    /    \\
$0 to $15k $15 to $35k over $35k
/     /     /     \
examples {1, 4, 7, 11} examples {2, 3, 12, 14} examples {5, 6, 8, 9, 10, 13}
```
Information Content of Messages - 1

- Given the universe of messages, \( M = \{m_1, m_2, \ldots, m_n\} \), where each \( m_i \) has probability \( p(m_i) \), then the information content is defined as:

\[
I(M) = \sum_{i=1}^{n} -p(m_i) \log_2(p(m_i))
\]

- For example, consider the toss of a “fair” coin where heads and tails have equal probability

\[
I(\text{Coin toss}) = -p(\text{heads})\log_2(p(\text{heads})) - p(\text{tails})\log_2(p(\text{tails}))
\]

\[
= -\frac{1}{2} \log_2\left(\frac{1}{2}\right) - \frac{1}{2} \log_2\left(\frac{1}{2}\right)
\]

\[
= 1 \text{ bit}
\]

- The information content is expressed in a number of bits

Information Content of Messages - 2

- If a coin comes up heads 75% of the time, then

\[
I(\text{Coin toss}) = -\frac{3}{4} \log_2\left(\frac{3}{4}\right) - \frac{1}{4} \log_2\left(\frac{1}{4}\right)
\]

\[
= -\frac{3}{4} \times (-0.415) - \frac{1}{4} \times (-2)
\]

\[
= 0.811 \text{ bits}
\]

- Applying this approach to Table 13.1 we have

\[
p(\text{risk is high}) = \frac{6}{14}, \ p(\text{risk is moderate}) = \frac{3}{14}, \ p(\text{risk is low}) = \frac{5}{14}
\]

\[
I(\text{Table13.1}) = -\frac{6}{14} \log_2\left(\frac{6}{14}\right) - \frac{3}{14} \log_2\left(\frac{3}{14}\right) - \frac{5}{14} \log_2\left(\frac{5}{14}\right)
\]

\[
= -\frac{6}{14} \times (-1.222) - \frac{3}{14} \times (-2.222) - \frac{5}{14} \times (-1.485)
\]

\[
= 1.531 \text{ bits}
\]
Information Gain - 1

• Assume
  – A set of training instances C
  – If P is the property with n values we select for the root of the tree, it will partition C into subsets \{C_1, C_2, \ldots, C_n\}
  – The expected information content needed to complete the tree after making P the root is:

\[
E(P) = \sum_{i=1}^{n} \frac{|C_i|}{|C|} I(C_i)
\]

• The information gain after choosing property P is

\[
gain(P) = I(C) - E(P)
\]

Information Gain - 2

• If income is used in Table 13.1, then C_1 = \{1, 4, 7, 11\}, C_2 = \{2, 3, 12, 14\}, and C_3 = \{5, 6, 8, 9, 10, 13\}; therefore

\[
E(\text{income}) = \frac{4}{14} \cdot I(C_1) + \frac{4}{14} \cdot I(C_2) + \frac{6}{14} \cdot I(C_3)
\]

\[
= \frac{4}{14} \cdot 0.0 + \frac{4}{14} \cdot 1.0 + \frac{6}{14} \cdot 0.650
\]

\[
= 0.564 \text{ bits}
\]

The information gain is:

\[
gain(\text{income}) = I(\text{table 12.1}) - E(\text{income})
\]

\[
= 1.531 - 0.564
\]

\[
= 0.967 \text{ bits}
\]

• The gain for the other choices is given below, so clearly it is best to select income as the root property

\[
gain(\text{credit history}) = 0.266
\]

\[
gain(\text{debt}) = 0.581
\]

\[
gain(\text{collateral}) = 0.756
\]
ID3 Performance

- Evaluating end game positions in chess
  - Recognize board positions that would lose in 3 moves
  - Tested based on 23 attributes, such as “an inability to move the king safely”
  - After symmetry, there were 1.4 million positions with 474,000 with a lose in 3 moves

- Here are the results based on the size of the training set

<table>
<thead>
<tr>
<th>Size of Training Set</th>
<th>Percentage of Whole Universe</th>
<th>Errors in 10,000 Trials</th>
<th>Predicted Maximum Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>0.01</td>
<td>199</td>
<td>728</td>
</tr>
<tr>
<td>1,000</td>
<td>0.07</td>
<td>33</td>
<td>146</td>
</tr>
<tr>
<td>5,000</td>
<td>0.36</td>
<td>8</td>
<td>29</td>
</tr>
<tr>
<td>25,000</td>
<td>1.79</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>125,000</td>
<td>8.93</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>