Information Gain

Which test is more informative?

**Split over whether Balance exceeds 50K**

- Less or equal 50K
- Over 50K

**Split over whether applicant is employed**

- Unemployed
- Employed
Information Gain

Impurity/Entropy (informal)

– Measures the level of impurity in a group of examples
Impurity

Very impure group

Less impure

Minimum impurity
Entropy: a common way to measure impurity

- Entropy = \( \sum_i -p_i \log_2 p_i \)

  \( p_i \) is the probability of class i
  Compute it as the proportion of class i in the set.

  16/30 are green circles; 14/30 are pink crosses
  \( \log_2(16/30) = -0.9; \quad \log_2(14/30) = -1.1 \)
  Entropy = -(16/30)(-0.9) –(14/30)(-1.1) = 0.99

- Entropy comes from information theory. The higher the entropy the more the information content.

  What does that mean for learning from examples?
2-Class Cases:

- What is the entropy of a group in which all examples belong to the same class?
  - entropy = \(-1 \log_2 1 = 0\)
  
  not a good training set for learning

- What is the entropy of a group with 50% in either class?
  - entropy = \(-0.5 \log_2 0.5 – 0.5 \log_2 0.5 = 1\)
  
  good training set for learning
Information Gain

• We want to determine which attribute in a given set of training feature vectors is most useful for discriminating between the classes to be learned.

• Information gain tells us how important a given attribute of the feature vectors is.

• We will use it to decide the ordering of attributes in the nodes of a decision tree.
Calculating Information Gain

Information Gain = entropy(parent) – [average entropy(children)]

```
Information Gain = 0.996 - 0.615 = 0.38  for this split
```

(Weighted) Average Entropy of Children = \[
\left( \frac{17}{30} \cdot 0.787 \right) + \left( \frac{13}{30} \cdot 0.391 \right) = 0.615
\]
Quinlan suggested information gain in his ID3 system and later the gain ratio, both based on entropy.
Using Information Gain to Construct a Decision Tree

1. Choose the attribute A with highest information gain for the full training set at the root of the tree.

2. Construct child nodes for each value of A. Each has an associated subset of vectors in which A has a particular value.

3. Repeat recursively till when?

```plaintext
Full Training Set S

Attribute A

v1 v2 vk

Set S' = \{s \in S | \text{value}(A) = v1\}
```
How would you distinguish class I from class II?

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>I</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>I</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>II</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>II</td>
</tr>
</tbody>
</table>
Split on attribute X

If X is the best attribute, this node would be further split.

\[ E_{\text{child}1} = -(1/3)\log_2(1/3) - (2/3)\log_2(2/3) \]
\[ = .5284 + .39 \]
\[ = .9184 \]

\[ E_{\text{child}2} = 0 \]

\[ E_{\text{parent}} = 1 \]

\[ \text{GAIN} = 1 - (3/4)(.9184) - (1/4)(0) = .3112 \]
<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>I</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>I</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>II</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>II</td>
</tr>
</tbody>
</table>

Split on attribute Y

\[
\begin{align*}
E_{\text{parent}} &= 1 \\
GAIN &= 1 - \left(\frac{1}{2}\right) 0 - \left(\frac{1}{2}\right) 0 = 1; \text{ BEST ONE}
\end{align*}
\]
<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>I</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>I</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>II</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>II</td>
</tr>
</tbody>
</table>

Split on attribute Z

\[ E_{\text{parent}} = 1 \]

\[ GAIN = 1 - \left( \frac{1}{2}\right)(1) - \left( \frac{1}{2}\right)(1) = 0 \]

ie. NO GAIN; WORST