Learning Decision Trees

Machine Learning

Some slides from Tom Mitchell, Dan Roth and others
This lecture: Learning Decision Trees

1. **Representation**: What are decision trees?

2. **Algorithm**: Learning decision trees
   - The ID3 algorithm: A greedy heuristic

3. Some extensions
This lecture: Learning Decision Trees

1. **Representation**: What are decision trees?

2. **Algorithm**: Learning decision trees
   - The ID3 algorithm: A greedy heuristic

3. Some extensions
History of Decision Tree Research

• Full search decision tree methods to model human concept learning: Hunt et al 60s, psychology

• Quinlan developed the ID3 (*Iterative Dichotomiser 3*) algorithm, with the information gain heuristic to learn expert systems from examples (late 70s)

• Breiman, Freidman and colleagues in statistics developed CART (Classification And Regression Trees)

• A variety of improvements in the 80s: coping with noise, continuous attributes, missing data, non-axis parallel, etc.

• Quinlan’s updated algorithms, C4.5 (1993) and C5 are more commonly used

• Boosting (or Bagging) over decision trees is a very good general purpose algorithm
Will I play tennis today?

• Features
  – Outlook: {Sun, Overcast, Rain}
  – Temperature: {Hot, Mild, Cool}
  – Humidity: {High, Normal, Low}
  – Wind: {Strong, Weak}

• Labels
  – Binary classification task: $Y = \{+, -\}$
Will I play tennis today?

<table>
<thead>
<tr>
<th>O</th>
<th>T</th>
<th>H</th>
<th>W</th>
<th>Play?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>3</td>
<td>O</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>7</td>
<td>O</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>8</td>
<td>S</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>9</td>
<td>S</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>M</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>11</td>
<td>S</td>
<td>M</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>12</td>
<td>O</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>13</td>
<td>O</td>
<td>H</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
</tbody>
</table>

**Outlook:** Sunny, Overcast, Rainy

**Temperature:** Hot, Medium, Cool

**Humidity:** High, Normal, Low

**Wind:** Strong, Weak
Basic Decision Tree Learning Algorithm

- Data is processed in Batch (i.e. all the data available)

<table>
<thead>
<tr>
<th></th>
<th>O</th>
<th>T</th>
<th>H</th>
<th>W</th>
<th>Play?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>W</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>S</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>O</td>
<td>H</td>
<td>H</td>
<td>W</td>
<td>+</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>W</td>
<td>+</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>W</td>
<td>+</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>S</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>O</td>
<td>C</td>
<td>N</td>
<td>S</td>
<td>+</td>
</tr>
<tr>
<td>8</td>
<td>S</td>
<td>M</td>
<td>H</td>
<td>W</td>
<td>-</td>
</tr>
<tr>
<td>9</td>
<td>S</td>
<td>C</td>
<td>N</td>
<td>W</td>
<td>+</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>M</td>
<td>N</td>
<td>W</td>
<td>+</td>
</tr>
<tr>
<td>11</td>
<td>S</td>
<td>M</td>
<td>N</td>
<td>S</td>
<td>+</td>
</tr>
<tr>
<td>12</td>
<td>O</td>
<td>M</td>
<td>H</td>
<td>S</td>
<td>+</td>
</tr>
<tr>
<td>13</td>
<td>O</td>
<td>H</td>
<td>N</td>
<td>W</td>
<td>+</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>S</td>
<td>-</td>
</tr>
</tbody>
</table>
Basic Decision Tree Learning Algorithm

- Data is processed in Batch (i.e. all the data available)
- Recursively build a decision tree top down.

<table>
<thead>
<tr>
<th>O</th>
<th>T</th>
<th>H</th>
<th>W</th>
<th>Play?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>3</td>
<td>O</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>7</td>
<td>O</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>8</td>
<td>S</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>9</td>
<td>S</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>M</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>11</td>
<td>S</td>
<td>M</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>12</td>
<td>O</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>13</td>
<td>O</td>
<td>H</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
</tbody>
</table>
Basic Decision Tree Learning Algorithm

- Data is processed in Batch (i.e. all the data available)
- Recursively build a decision tree top down.

<table>
<thead>
<tr>
<th>O</th>
<th>T</th>
<th>H</th>
<th>W</th>
<th>Play?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>3</td>
<td>O</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>7</td>
<td>O</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>8</td>
<td>S</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>9</td>
<td>S</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>M</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>11</td>
<td>S</td>
<td>M</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>12</td>
<td>O</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>13</td>
<td>O</td>
<td>H</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
</tbody>
</table>

Decide what attribute goes at the top
Decide what to do for each value the root attribute takes
Basic Decision Tree Algorithm: ID3

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label

Input:
S the set of Examples
Label is the target attribute (the prediction)
Attributes is the set of measured attributes
Basic Decision Tree Algorithm: ID3

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label

2. Otherwise

Input:
S the set of Examples
Label is the target attribute (the prediction)
Attributes is the set of measured attributes
Basic Decision Tree Algorithm: ID3

**ID3(S, Attributes, Label):**

1. If all examples are have same label:
   - Return a single node tree with the label

2. Otherwise
   1. Create a **Root** node for tree
      - Decide what attribute goes at the top

4. Return **Root** node

**Input:**
- **S** the set of Examples
- **Label** is the target attribute (the prediction)
- **Attributes** is the set of measured attributes
Basic Decision Tree Algorithm: ID3

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label

2. Otherwise
   1. Create a Root node for tree
   2. A = attribute in Attributes that best classifies S

4. Return Root node

Input:
S the set of Examples
Label is the target attribute (the prediction)
Attributes is the set of measured attributes

Decide what attribute goes at the top
Basic Decision Tree Algorithm: ID3

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label

2. Otherwise
   1. Create a Root node for tree
   2. A = attribute in Attributes that best classifies S
   3. for each possible value v of that A can take:
      1. Add a new tree branch corresponding to A = v
      2. Let S_v be the subset of examples in S with A = v
      3. if S_v is empty: add leaf node with the common value of Label in S
         Else: below this branch add the subtree ID3(S_v, Attributes-{A}, Label)

4. Return Root node
Basic Decision Tree Algorithm: ID3

**ID3(S, Attributes, Label):**

1. **If** all examples are have same label:
   - Return a single node tree with the label

2. **Otherwise**
   1. Create a **Root node** for tree
   2. **A = attribute in Attributes that best** classifies S
   3. **for each possible value** v **of that A can take:**
      1. Add a new tree branch for attribute A taking value v

4. Return **Root node**
Basic Decision Tree Algorithm: ID3

ID3(\(S, \text{Attributes}, \text{Label}\)):

1. If all examples are have same label:
   - Return a single node tree with the label

2. Otherwise
   1. Create a \textbf{Root node} for tree
   2. \(A = \) attribute in Attributes that \textit{best} classifies \(S\)
   3. for each possible value \(v\) of that \(A\) can take:
      1. Add a new tree branch for attribute \(A\) taking value \(v\)
      2. Let \(S_v\) be the subset of examples in \(S\) with \(A=v\)

3. Return \textbf{Root node}
Basic Decision Tree Algorithm: ID3

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label

2. Otherwise
   1. Create a Root node for tree
   2. A = attribute in Attributes that best classifies S
   3. for each possible value v of that A can take:
      1. Add a new tree branch for attribute A taking value v
      2. Let $S_v$ be the subset of examples in S with A=v
      3. if $S_v$ is empty:

4. Return Root node

Input:
S the set of Examples
Label is the target attribute (the prediction)
Attributes is the set of measured attributes

Decide what to do for each value the root attribute takes
Basic Decision Tree Algorithm: ID3

**Input:**
- \( S \) the set of Examples
- Label is the target attribute (the prediction)
- Attributes is the set of measured attributes

**ID3(\( S, \) Attributes, Label):**

1. If all examples are have same label:
   
   Return a single node tree with the label

2. Otherwise
   
   1. Create a **Root node** for tree
   2. \( A = \) attribute in Attributes that **best** classifies \( S \)
   3. for each possible value \( v \) of that \( A \) can take:
      
      1. Add a new tree branch for attribute \( A \) taking value \( v \)
      2. Let \( S_v \) be the subset of examples in \( S \) with \( A=v \)
      3. if \( S_v \) is empty:
         
         add leaf node with the common value of Label in \( S \)

4. Return **Root node**

**why?**

Decide what to do for each value the root attribute takes
Basic Decision Tree Algorithm: ID3

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label

2. Otherwise
   1. Create a Root node for tree
   2. \( A = \) attribute in Attributes that best classifies \( S \)
   3. for each possible value \( v \) of that \( A \) can take:
      1. Add a new tree branch for attribute \( A \) taking value \( v \)
      2. Let \( S_v \) be the subset of examples in \( S \) with \( A = v \)
      3. if \( S_v \) is empty:
         add leaf node with the common value of Label in \( S \)
   4. Return Root node

Input:
- \( S \) the set of Examples
- Label is the target attribute (the prediction)
- Attributes is the set of measured attributes

Decide what to do for each value the root attribute takes

For generalization at test time
Basic Decision Tree Algorithm: ID3

ID3(S, Attributes, Label):

1. If all examples are have same label:
   Return a single node tree with the label

2. Otherwise
   1. Create a Root node for tree
   2. $A =$ attribute in Attributes that best classifies S
   3. for each possible value $v$ of that $A$ can take:
      1. Add a new tree branch for attribute $A$ taking value $v$
      2. Let $S_v$ be the subset of examples in $S$ with $A=v$
      3. if $S_v$ is empty:
         add leaf node with the common value of Label in $S$
      Else:
         below this branch add the subtree ID3($S_v$, Attributes - \{A\}, Label)
   4. Return Root node
Picking the Root Attribute

• Goal: Have the resulting decision tree as small as possible *(Occam’s Razor)*
  – But, finding the minimal decision tree consistent with data is NP-hard

• The recursive algorithm is a greedy heuristic search for a simple tree, but cannot guarantee optimality

• The main decision in the algorithm is the selection of the next attribute to split on
Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

- < (A=0,B=0), − >: 50 examples
- < (A=0,B=1), − >: 50 examples
- < (A=1,B=0), − >: 0 examples
- < (A=1,B=1), + >: 100 examples
Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

\[
\begin{align*}
&\text{< (A=0,B=0), } \neg \text{ >: 50 examples} \\
&\text{< (A=0,B=1), } \neg \text{ >: 50 examples} \\
&\text{< (A=1,B=0), } \neg \text{ >: 0 examples} \\
&\text{< (A=1,B=1), } + \text{ >: 100 examples}
\end{align*}
\]

What should be the first attribute we select?
Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

- \( (A=0, B=0), - \): 50 examples
- \( (A=0, B=1), - \): 50 examples
- \( (A=1, B=0), - \): 0 examples
- \( (A=1, B=1), + \): 100 examples

What should be the first attribute we select?

Splitting on A: we get purely labeled nodes.
Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

- < (A=0,B=0), − >: 50 examples
- < (A=0,B=1), − >: 50 examples
- < (A=1,B=0), − >: 0 examples
- < (A=1,B=1), + >: 100 examples

What should be the first attribute we select?

Splitting on A: we get purely labeled nodes.

Splitting on B: we don’t get purely labeled nodes.
Picking the Root Attribute

Consider data with two Boolean attributes \((A,B)\).

\[
\begin{align*}
\langle & (A=0,B=0), - \rangle: & 50 \text{ examples} \\
\langle & (A=0,B=1), - \rangle: & 50 \text{ examples} \\
\langle & (A=1,B=0), - \rangle: & 0 \text{ examples} \\
\langle & (A=1,B=1), + \rangle: & 100 \text{ examples}
\end{align*}
\]

What should be the first attribute we select?

Splitting on \(A\): we get purely labeled nodes.

Splitting on \(B\): we don’t get purely labeled nodes.

What if we have: \(\langle(A=1,B=0), - \rangle\): 3 examples
Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

- \( (A=0, B=0) \): 50 examples
- \( (A=0, B=1) \): 50 examples
- \( (A=1, B=0) \): 0 examples
- \( (A=1, B=1) \): 3 examples

Which attribute should we choose?
Picking the Root Attribute

Consider data with two Boolean attributes \((A,B)\).

\[
\begin{align*}
&< (A=0,B=0), - >: 50 \text{ examples} \\
&< (A=0,B=1), - >: 50 \text{ examples} \\
&< (A=1,B=0), - >: 0 \text{ examples } 3 \text{ examples} \\
&< (A=1,B=1), + >: 100 \text{ examples}
\end{align*}
\]

Which attribute should we choose?
Picking the Root Attribute

Consider data with two Boolean attributes \((A, B)\).

- \((A=0, B=0), - >: 50\) examples
- \((A=0, B=1), - >: 50\) examples
- \((A=1, B=0), - >: 0\) examples, 3 examples
- \((A=1, B=1), + >: 100\) examples

Which attribute should we choose?
Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

- (A=0, B=0), - >: 50 examples
- (A=0, B=1), - >: 50 examples
- (A=1, B=0), - >: 0 examples 3 examples
- (A=1, B=1), + >: 100 examples

Which attribute should we choose? Trees looks structurally similar!
Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

- < (A=0,B=0), −, >: 50 examples
- < (A=0,B=1), −, >: 50 examples
- < (A=1,B=0), −, >: 0 examples, 3 examples
- < (A=1,B=1), +, >: 100 examples

Which attribute should we choose? Trees looks structurally similar!
Picking the Root Attribute

Consider data with two Boolean attributes (A,B).

\[
\begin{align*}
&< \ (A=0,B=0), \ - >: \ 50 \text{ examples} \\
&< \ (A=0,B=1), \ - >: \ 50 \text{ examples} \\
&< \ (A=1,B=0), \ - >: \ \underline{0} \text{ examples} \quad 3 \text{ examples} \\
&< \ (A=1,B=1), \ + >: \ 100 \text{ examples}
\end{align*}
\]

Which attribute should we choose? Trees looks structurally similar!

Advantage A. But... Need a way to quantify things.
Picking the Root Attribute

Goal: Have the resulting decision tree as small as possible (Occam’s Razor)

• The main decision in the algorithm is the selection of the next attribute for splitting the data

• We want attributes that split the examples to sets that are relatively pure in one label
  – This way we are closer to a leaf node.

• The most popular heuristic is information gain, originated with the ID3 system of Quinlan
Reminder: Entropy

*Entropy* (impurity, disorder) of a set of examples $S$ with respect to binary classification is

$$Entropy(S) = H(S) = -p_+ \log(p_+) - p_- \log(p_-)$$

- The proportion of positive examples is $p_+$
- The proportion of negative examples is $p_-$
Reminder: Entropy

Entropy (impurity, disorder) of a set of examples $S$ with respect to binary classification is

$$Entropy(S) = H(S) = -p_+ \log(p_+) - p_- \log(p_-)$$

- The proportion of positive examples is $p_+$
- The proportion of negative examples is $p_-$

In general, for a discrete probability distribution with $K$ possible values, with probabilities $\{p_1, p_2, \ldots, p_K\}$ the entropy is given by

$$H(\{p_1, p_2, \ldots, p_K\}) = - \sum_{i=1}^{K} p_i \log(p_i)$$
Reminder: Entropy

*Entropy* (impurity, disorder) of a set of examples S with respect to binary classification is

\[
Entropy(S) = H(S) = -p_+ \log(p_+) - p_- \log(p_-)
\]

- The proportion of positive examples is \( p_+ \)
- The proportion of negative examples is \( p_- \)
- If all examples belong to the same category, entropy = 0
- If \( p_+ = p_- = 0.5 \), entropy = 1

Entropy can be viewed as the number of bits required, on average, to encode the class labels. If the probability for + is 0.5, a single bit is required for each example; if it is 0.8, we can use less then 1 bit.
**Entropy** (impurity, disorder) of a set of examples $S$ with respect to binary classification is

$$
Entropy(S) = H(S) = -p_+ \log(p_+) - p_- \log(p_-)
$$

- The proportion of positive examples is $p_+$
- The proportion of negative examples is $p_-$

- If all examples belong to the same category, entropy = 0
- If $p_+ = p_- = 0.5$, entropy = 1
Reminder: Entropy

*Entropy* (impurity, disorder) of a set of examples $S$ with respect to binary classification is

The uniform distribution has the highest entropy
Reminder: Entropy

*Entropy* (impurity, disorder) of a set of examples $S$ with respect to binary classification is

The uniform distribution has the highest entropy
**Reminder: Entropy**

*Entropy* (impurity, disorder) of a set of examples $S$ with respect to binary classification is

The uniform distribution has the highest entropy

---

**High Entropy – High level of Uncertainty**

**Low Entropy – Low Uncertainty.**
Picking the Root Attribute

Goal: Have the resulting decision tree **as small as possible** (*Occam’s Razor*)

- The main decision in the algorithm is the selection of the next attribute for splitting the data

- We want attributes that split the examples to sets that are relatively pure in one label
  - This way we are closer to a leaf node.

- The most popular heuristic is **information gain**, originated with the ID3 system of Quinlan

*Intuition: Choose the attribute that reduces the label entropy the most*
Information Gain

The *information gain* of an attribute $A$ is the expected reduction in entropy caused by partitioning on this attribute

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

$S_v$: the subset of examples where the value of attribute $A$ is set to value $v$
Information Gain

The *information gain* of an attribute $A$ is the expected reduction in entropy caused by partitioning on this attribute:

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

$S_v$: the subset of examples where the value of attribute $A$ is set to value $v$

Entropy of partitioning the data is calculated by weighing the entropy of each partition by its size relative to the original set.

- Partitions of low entropy (imbalanced splits) lead to high gain.
Information Gain

The *information gain* of an attribute $A$ is the expected reduction in entropy caused by partitioning on this attribute

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

$S_v$: the subset of examples where the value of attribute $A$ is set to value $v$

Entropy of partitioning the data is calculated by weighing the entropy of each partition by its size relative to the original set

– Partitions of low entropy (imbalanced splits) lead to high gain

*Go back to check which of the A, B splits is better*
Information Gain

The *information gain* of an attribute $A$ is the expected reduction in entropy caused by partitioning on this attribute

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

$S_v$: the subset of examples where the value of attribute $A$ is set to value $v$

Entropy of partitioning the data is calculated by weighing the entropy of each partition by its size relative to the original set

– Partitions of low entropy (imbalanced splits) lead to high gain

*Go back to check which of the $A, B$ splits is better*
Will I play tennis today?

### Outlook
- S(unny), O(vercast), R(ainy)

### Temperature
- H(ot), M(edium), C(cool)

### Humidity
- H(igh), N(ormal), L(ow)

### Wind
- S(trong), W(eak)

<table>
<thead>
<tr>
<th>O</th>
<th>T</th>
<th>H</th>
<th>W</th>
<th>Play?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>3</td>
<td>O</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>7</td>
<td>O</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>8</td>
<td>S</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>9</td>
<td>S</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>M</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>11</td>
<td>S</td>
<td>M</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>12</td>
<td>O</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>13</td>
<td>O</td>
<td>H</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
</tbody>
</table>
Will I play tennis today?

Current entropy:
\( p = \frac{9}{14} \)
\( n = \frac{5}{14} \)

\[ H(\text{Play?}) = -\left(\frac{9}{14}\right) \log_2\left(\frac{9}{14}\right) -\left(\frac{5}{14}\right) \log_2\left(\frac{5}{14}\right) \approx 0.94 \]
## Information Gain: Outlook

<table>
<thead>
<tr>
<th></th>
<th>T</th>
<th>H</th>
<th>W</th>
<th>Play?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>3</td>
<td>O</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>7</td>
<td>O</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>8</td>
<td>S</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>9</td>
<td>S</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>M</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>11</td>
<td>S</td>
<td>M</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>12</td>
<td>O</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>13</td>
<td>O</td>
<td>H</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
</tbody>
</table>
### Information Gain: Outlook

**Outlook = sunny:** 5 of 14 examples  
\[ p = \frac{2}{5}, \quad n = \frac{3}{5}, \quad H_S = 0.971 \]

<table>
<thead>
<tr>
<th>O</th>
<th>T</th>
<th>H</th>
<th>W</th>
<th>Play?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>3</td>
<td>O</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>7</td>
<td>O</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>8</td>
<td>S</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>9</td>
<td>S</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>M</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>11</td>
<td>S</td>
<td>M</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>12</td>
<td>O</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>13</td>
<td>O</td>
<td>H</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
</tbody>
</table>
Information Gain: Outlook

Outlook = sunny: 5 of 14 examples
\[ p = \frac{2}{5} \quad n = \frac{3}{5} \quad H_S = 0.971 \]

Outlook = overcast: 4 of 14 examples
\[ p = \frac{4}{4} \quad n = 0 \quad H_o = 0 \]
Information Gain: Outlook

**Outlook = sunny:** 5 of 14 examples

\[ p = \frac{2}{5} \quad n = \frac{3}{5} \quad H_S = 0.971 \]

**Outlook = overcast:** 4 of 14 examples

\[ p = \frac{4}{4} \quad n = 0 \quad H_o = 0 \]

**Outlook = rainy:** 5 of 14 examples

\[ p = \frac{3}{5} \quad n = \frac{2}{5} \quad H_R = 0.971 \]

Expected entropy:

\[
(\frac{5}{14}) \times 0.971 + (\frac{4}{14}) \times 0 + (\frac{5}{14}) \times 0.971 = 0.694
\]

Information gain:

\[ 0.940 - 0.694 = 0.246 \]
### Information Gain: Humidity

<table>
<thead>
<tr>
<th></th>
<th>T</th>
<th>H</th>
<th>W</th>
<th>Play?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S</td>
<td>H</td>
<td>W</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>H</td>
<td>S</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>O</td>
<td>H</td>
<td>W</td>
<td>+</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>7</td>
<td>O</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>8</td>
<td>S</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>9</td>
<td>S</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>M</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>11</td>
<td>S</td>
<td>M</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>12</td>
<td>O</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>13</td>
<td>O</td>
<td>H</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
</tbody>
</table>
### Information Gain: Humidity

**Humidity** = High:

\[ p = \frac{3}{7} \quad n = \frac{4}{7} \quad H_h = 0.985 \]

<table>
<thead>
<tr>
<th>O</th>
<th>T</th>
<th>H</th>
<th>W</th>
<th>Play?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>3</td>
<td>O</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>7</td>
<td>O</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>8</td>
<td>S</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>9</td>
<td>S</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>M</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>11</td>
<td>S</td>
<td>M</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>12</td>
<td>O</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>13</td>
<td>O</td>
<td>H</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
</tbody>
</table>
Information Gain: Humidity

**Humidity** = High:
\[ p = \frac{3}{7} \quad n = \frac{4}{7} \quad H_h = 0.985 \]

**Humidity** = Normal:
\[ p = \frac{6}{7} \quad n = \frac{1}{7} \quad H_o = 0.592 \]

**Expected entropy:**
\[ (\frac{7}{14}) \times 0.985 + (\frac{7}{14}) \times 0.592 = 0.7885 \]
**Information Gain: Humidity**

<table>
<thead>
<tr>
<th>O</th>
<th>T</th>
<th>H</th>
<th>W</th>
<th>Play?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>3</td>
<td>O</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>7</td>
<td>O</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>8</td>
<td>S</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>9</td>
<td>S</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>M</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>11</td>
<td>S</td>
<td>M</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>12</td>
<td>O</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>13</td>
<td>O</td>
<td>H</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
</tbody>
</table>

**Humidity = High:**

\[
p = \frac{3}{7} \quad n = \frac{4}{7} \quad H_h = 0.985
\]

**Humidity = Normal:**

\[
p = \frac{6}{7} \quad n = \frac{1}{7} \quad H_o = 0.592
\]

**Expected entropy:**

\[
(\frac{7}{14}) \times 0.985 + (\frac{7}{14}) \times 0.592 = 0.7885
\]

**Information gain:**

\[
0.940 - 0.7885 = 0.1515
\]
## Which feature to split on?

### Information gain:
- Outlook: 0.246
- Humidity: 0.151
- Wind: 0.048
- Temperature: 0.029

<table>
<thead>
<tr>
<th>O</th>
<th>T</th>
<th>H</th>
<th>W</th>
<th>Play?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>3</td>
<td>O</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>7</td>
<td>O</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>8</td>
<td>S</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>9</td>
<td>S</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>M</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>11</td>
<td>S</td>
<td>M</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>12</td>
<td>O</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>13</td>
<td>O</td>
<td>H</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
</tbody>
</table>
Which feature to split on?

<table>
<thead>
<tr>
<th></th>
<th>T</th>
<th>H</th>
<th>W</th>
<th>Play?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>3</td>
<td>O</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>7</td>
<td>O</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>8</td>
<td>S</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>9</td>
<td>S</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>M</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>11</td>
<td>S</td>
<td>M</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>12</td>
<td>O</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>13</td>
<td>O</td>
<td>H</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
</tbody>
</table>

Information gain:
- Outlook: 0.246
- Humidity: 0.151
- Wind: 0.048
- Temperature: 0.029

→ Split on Outlook
An Illustrative Example

Gain(S, Humidity) = 0.151
Gain(S, Wind) = 0.048
Gain(S, Temperature) = 0.029
Gain(S, Outlook) = 0.246
An Illustrative Example

<table>
<thead>
<tr>
<th>Outlook</th>
<th>T</th>
<th>H</th>
<th>W</th>
<th>Play?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sunny</td>
<td>H</td>
<td>H</td>
<td>H</td>
<td>-</td>
</tr>
<tr>
<td>Overcast</td>
<td>H</td>
<td>H</td>
<td>S</td>
<td>-</td>
</tr>
<tr>
<td>Rain</td>
<td>C</td>
<td>N</td>
<td>W</td>
<td>+</td>
</tr>
<tr>
<td>Sunny</td>
<td>M</td>
<td>H</td>
<td>W</td>
<td>-</td>
</tr>
<tr>
<td>Overcast</td>
<td>C</td>
<td>N</td>
<td>S</td>
<td>-</td>
</tr>
<tr>
<td>Rain</td>
<td>O</td>
<td>C</td>
<td>N</td>
<td>+</td>
</tr>
<tr>
<td>Sunny</td>
<td>S</td>
<td>M</td>
<td>N</td>
<td>+</td>
</tr>
<tr>
<td>Overcast</td>
<td>R</td>
<td>M</td>
<td>N</td>
<td>+</td>
</tr>
<tr>
<td>Rain</td>
<td>S</td>
<td>M</td>
<td>N</td>
<td>+</td>
</tr>
<tr>
<td>Sunny</td>
<td>O</td>
<td>M</td>
<td>H</td>
<td>+</td>
</tr>
<tr>
<td>Overcast</td>
<td>O</td>
<td>H</td>
<td>N</td>
<td>+</td>
</tr>
<tr>
<td>Rain</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>-</td>
</tr>
</tbody>
</table>
An Illustrative Example

<table>
<thead>
<tr>
<th>O</th>
<th>T</th>
<th>H</th>
<th>W</th>
<th>Play?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>H</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>3</td>
<td>O</td>
<td>H</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>7</td>
<td>O</td>
<td>C</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>8</td>
<td>S</td>
<td>M</td>
<td>H</td>
<td>W</td>
</tr>
<tr>
<td>9</td>
<td>S</td>
<td>C</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>M</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>11</td>
<td>S</td>
<td>M</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>12</td>
<td>O</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>13</td>
<td>O</td>
<td>H</td>
<td>N</td>
<td>W</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
</tbody>
</table>

Continue until:
- Every attribute is included in path, or,
- All examples in the leaf have same label

Outlook

- Sunny: 1, 2, 8, 9, 11
  - 2+, 3-
  - ?
- Overcast: 3, 7, 12, 13
  - 4+, 0-
  - Yes
- Rain: 4, 5, 6, 10, 14
  - 3+, 2-
  - ?

Play?
An Illustrative Example

Gain_{sunny, Humidity} = .97 - (3/5) 0 - (2/5) 0 = .97
Gain_{sunny, Temp} = .97 - 0 - (2/5) 1 = .57
Gain_{sunny, wind} = .97 - (2/5) 1 - (3/5) .92 = .02

Day | Outlook | Temperature | Humidity | Wind | PlayTennis
--- | --- | --- | --- | --- | ---
1 | Sunny | Hot | High | Weak | No
2 | Sunny | Hot | High | Strong | No
8 | Sunny | Mild | High | Weak | No
9 | Sunny | Cool | Normal | Weak | Yes
11 | Sunny | Mild | Normal | Strong | Yes
An Illustrative Example

Outlook

- Sunny
  - 1, 2, 8, 9, 11
  - 2+, 3-
  - ?

- Overcast
  - 3, 7, 12, 13
  - 4+, 0-
  - Yes

- Rain
  - 4, 5, 6, 10, 14
  - 3+, 2-
  - ?
An Illustrative Example

Outlook

- Sunny: 1, 2, 8, 9, 11
  - Humidity: 2+, 3-
- Overcast: 3, 7, 12, 13
  - Humidity: 4+, 0-
- Rain: 4, 5, 6, 10, 14
  - Humidity: 3+, 2-

- Humidity: Normal
  - Yes
- Humidity: High
  - No
An Illustrative Example

Outlook

- Sunny
  - 1,2,8,9,11
  - 2+,3-
  - High
    - No
  - Normal
    - Yes

- Overcast
  - 3,7,12,13
  - 4+,0-
  - Yes

- Rain
  - 4,5,6,10,14
  - 3+,2-
  - Strong
    - No
  - Weak
    - Yes
induceDecisionTree(S)

1. Does S uniquely define a class?
   
   if all \( s \in S \) have the same label \( y \): return \( S \);

2. Find the feature with the most information gain:
   
   \( i = \text{argmax}_i \text{Gain}(S, X_i) \)

3. Add children to \( S \):
   
   for \( k \) in \( \text{Values}(X_i) \):
     
     \( S_k = \{ s \in S \mid x_i = k \} \)
     
     addChild(S, \( S_k \))
     
     induceDecisionTree(\( S_k \))

   return \( S \);
Hypothesis Space in Decision Tree Induction

• Search over decision trees, which can represent all possible discrete functions (has pros and cons)

• Goal: to find the best decision tree

• Finding a minimal decision tree consistent with a set of data is NP-hard.

• ID3 performs a greedy heuristic search: hill climbing without backtracking

• Makes statistically based decisions using all data
Summary: Learning Decision Trees

1. **Representation**: What are decision trees?
   - A hierarchical data structure that represents data

2. **Algorithm**: Learning decision trees

   The ID3 algorithm: A greedy heuristic
   - If all the examples have the same label, create a leaf with that label
   - Otherwise, find the “most informative” attribute and split the data for different values of that attributes
   - Recurse on the splits