Decision and Regression Trees

- Decision tree representation
- ID3 learning algorithm
- Entropy, information gain
- Expected mean-squared error
- Overfitting
Decision Tree for PlayTennis

Each internal node tests an attribute

Each branch corresponds to an attribute value node

Each leaf node assigns a classification
Decision Tree for Conjunction

Outlook=Sunny ∧ Wind=Weak
Decision Tree for Disjunction

Outlook=Sunny \lor Wind=Weak

Outlook
  Sunny
    Yes
  Overcast
  Rain
    Wind
      Strong
        No
      Weak
        Yes
        No
    Strong
      Yes
    Weak
      Yes
Decision Tree for XOR

Outlook=Sunny  XOR Wind=Weak
Decision Tree

- decision trees represent disjunctions of conjunctions

\[
\text{Outlook} \quad \text{Sunny} \quad \text{Overcast} \quad \text{Rain} \\
\text{Humidity} \quad \text{Yes} \quad \text{Wind} \\
\text{High} \quad \text{Normal} \quad \text{Strong} \quad \text{Weak} \\
\text{No} \quad \text{Yes} \quad \text{No} \quad \text{Yes}
\]

\[
\text{(Outlook=Sunny} \wedge \text{Humidity=Normal}) \\
\vee \quad \text{(Outlook=Overcast)} \\
\vee \quad \text{(Outlook=Rain} \wedge \text{Wind=Weak)}
\]
Top-Down Induction of Decision Trees
ID3

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
4. Sort training examples to leaf node according to the attribute value of the branch
5. If all training examples are perfectly classified (same value of target attribute) stop, else iterate over new leaf nodes.
Which attribute is best?

\[ [29+, 35-] \quad A_1=? \]

\[ [21+, 5-] \quad G \]
\[ [8+, 30-] \quad H \]

\[ [18+, 33-] \quad L \]
\[ [11+, 2-] \quad M \]
Entropy

- S is a sample of training examples
- \( p_+ \) is the proportion of positive examples
- \( p_- \) is the proportion of negative examples
- Entropy measures the impurity of S

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

- Entropy(S) = expected number of bits needed to encode class (+ or -) of randomly drawn members of S (under the optimal, shortest length-code)

Why?

- Information theory optimal length code assign
  - $\log_2 p$ bits to messages having probability $p$.
- So the expected number of bits to encode
  (+ or -) of random member of S:
  \[-p_+ \log_2 p_+ - p_- \log_2 p_-\]
Information Gain (S=E)

- Gain(S,A): expected reduction in entropy due to sorting S on attribute A

\[
Gain(S, A) \equiv Entropy(S) - \sum_{v \in D_A} \frac{|S_v|}{|S|} Entropy(S_v)
\]

Entropy([29+,35-]) = \(-29/64 \log_2 29/64 - 35/64 \log_2 35/64 = 0.99\)

\[
\text{Entropy}([29+,35-]) = -\frac{29}{64} \log_2 \frac{29}{64} - \frac{35}{64} \log_2 \frac{35}{64} = 0.99
\]
Information Gain

Entropy([21+, 5-]) = 0.71
Entropy([8+, 30-]) = 0.74
Gain(S,A₁) = Entropy(S) - 26/64 * Entropy([21+, 5-])
- 38/64 * Entropy([8+, 30-])
= 0.27

Entropy([18+, 33-]) = 0.94
Entropy([11+, 2-]) = 0.62
Gain(S,A₂) = Entropy(S) - 51/64 * Entropy([18+, 33-])
- 13/64 * Entropy([11+, 2-])
= 0.12
## Training Examples

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temp.</th>
<th>Humidity</th>
<th>Wind</th>
<th>Play Tennis</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>Weak</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>Cold</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>Strong</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>
Selecting the Next Attribute

Humidity provides greater info. gain than Wind, w.r.t target classification.
Selecting the Next Attribute

\[ S = [9+, 5-] \]
\[ E = 0.940 \]

\( \text{Outlook} \)

\( \text{Sunny} \)
\[ [2+, 3-] \]
\[ E = 0.971 \]

\( \text{Overcast} \)
\[ [4+, 0] \]
\[ E = 0.0 \]

\( \text{Rain} \)
\[ [3+, 2-] \]
\[ E = 0.971 \]

\( \text{Gain}(S, \text{Outlook}) \) 
\[ = 0.940 - \left( \frac{5}{14} \right) \times 0.971 \] 
\[ -(4/14) \times 0.0 - (5/14) \times 0.0971 \] 
\[ = 0.247 \]
Selecting the Next Attribute

The information gain values for the 4 attributes are:
- $\text{Gain}(S, \text{Outlook}) = 0.247$
- $\text{Gain}(S, \text{Humidity}) = 0.151$
- $\text{Gain}(S, \text{Wind}) = 0.048$
- $\text{Gain}(S, \text{Temperature}) = 0.029$

where $S$ denotes the collection of training examples
ID3 Algorithm

\[
[D_1, D_2, \ldots, D_{14}] \\
[9+, 5-]
\]

Outlook

\[
[D_1, D_2, D_8, D_9, D_{11}] \\
[2+, 3-]
\]
Sunny

\[
[D_3, D_7, D_{12}, D_{13}] \\
[4+, 0-]
\]
Overcast

\[
[D_4, D_5, D_6, D_{10}, D_{14}] \\
[3+, 2-]
\]
Rain

\[
S_{\text{sunny}} = [D_1, D_2, D_8, D_9, D_{11}] [D_3, D_7, D_{12}, D_{13}] [D_4, D_5, D_6, D_{10}, D_{14}]
\]

Gain(S_{\text{sunny}}, \text{Humidity}) = 0.970 - (3/5)0.0 - 2/5(0.0) = 0.970

Gain(S_{\text{sunny}}, \text{Temp.}) = 0.970 - (2/5)0.0 - 2/5(1.0) - (1/5)0.0 = 0.570

Gain(S_{\text{sunny}}, \text{Wind}) = 0.970 = -(2/5)1.0 - 3/5(0.918) = 0.019
ID3 Algorithm

- **Outlook**
  - Sunny
  - Overcast
  - Rain

- **Humidity**
  - High
    - No
    - [D1, D2]
  - Normal
    - Yes
    - [D3, D7, D12, D13]

- **Wind**
  - Strong
    - No
    - [D6, D14]
  - Weak
    - Yes
    - [D4, D5, D10]
Continuous Valued Attributes

Create a discrete attribute to test

• Temperature = 24.5°C
• (Temperature > 20.0°C) = {true, false}

Where to set the threshold?

<table>
<thead>
<tr>
<th>Temperature</th>
<th>15°C</th>
<th>18°C</th>
<th>19°C</th>
<th>22°C</th>
<th>24°C</th>
<th>27°C</th>
</tr>
</thead>
<tbody>
<tr>
<td>PlayTennis</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
ID3 for classification and continuous attributes

ID3(x,y)
1. If termination condition applies return leaf with most common class in y
2. Let (a,t) be the attribute and threshold combination with the highest information gain
3. Let (xl, yl) be the training examples for which x(a)<=t
4. Let (xr, yr) be the training examples for which x(a)>t
5. Return decision tree node where attribute = a, threshold = t, leftchild = ID3(xl,yl), rightchild = ID3 (xr,yr)
ID3 for regression and continuous attributes

ID3(x,y)
1. If termination condition applies return leaf with mean value in y
2. Let (a,t) be the attribute and threshold combination with the lowest mean squared error
3. Let (xl, yl) be the training examples for which x(a)≤t
4. Let (xr, yr) be the training examples for which x(a)>t
5. Return decision tree node where attribute = a, threshold = t, leftchild = ID3(xl,yl), rightchild = ID3 (xr,yr)
Regression trees: 
Making a prediction and computing its error

What is the best single prediction we can make (the one with minimum error) for all data?
• The average y value in the dataset:
  \[ p = \frac{\sum y_i}{n} \]
• What is the mean squared error of that prediction?
  \[ \text{MSE} = \frac{\sum (y_i - p_i)^2}{n} \] (it is also the variance of y)
A simple dataset

<table>
<thead>
<tr>
<th></th>
<th>$x[:,0]$</th>
<th>$x[:,1]$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.6</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.9</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.1</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.2</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.8</td>
<td>30</td>
<td></td>
</tr>
</tbody>
</table>
Regression trees: 
Making a prediction and computing its error

What is the best prediction we can make (the one with minimum error)?

• The average y value in the dataset:
  
  \[ p = \frac{\sum y_i}{n} \]

  for the example, \( p = \frac{(50+40+20+10+30)}{5} = 30 \)

• What is the mean squared error of that prediction?

  \[ \text{MSE} = \frac{\sum (y_i - p_i)^2}{n} \]

  \[ \text{MSE} = \frac{((50-30)^2 + \ldots + (30-30)^2)}{5} = \frac{(400+100+100+400+0)}{5} = \frac{1000}{5} = 200 \]
Finding the best attribute and threshold

From the dataset we can see that there are 10 possible combinations (2 attributes and 5 examples)

\[ X_0 \leq 1, \ X_0 \leq 2, \ldots, \ X_0 \leq 5 \]

\[ X_1 \leq 0.6, \ X_1 \leq 0.9, \ldots, \ X_1 \leq 0.8 \]

Which one results in the minimum mean squared error?
Finding the best attribute and threshold

\[ x[0] \leq 1 \]

- \[ e_1 \]
  - \[ y = [50] \]
  - \[ p = 50 \]

- \[ e_2, e_3, e_4, e_5 \]
  - \[ y = [40, 20, 10, 30] \]
  - \[ p = 25 \]

\[
\text{MSE} = \frac{((50-50)^2 + (40-25)^2 + (20-25)^2 + (10-25)^2 + (30-25)^2)/5}{= 100}
\]

Thus the mean squared error is reduced from 200 to 100 with this attribute and threshold combination.
Finding the best attribute and threshold – a small improvement

Instead of choosing threshold as values in the training set, we do the following:

Sort attribute by value: $x_0, x_1, x_2, x_n$

Choose as thresholds:

$\frac{x_i + x_{i+1}}{2}$, for $1 \leq i < n$

Thus for $X[0]$, the thresholds would be $[1.5, 2.5, 3.5, 4.5]$

if $(x_i = x_{i+1})$ or $(y_i = y_{i+1})$ it is not necessary to consider $(x_i + x_{i+1})/2$ as threshold

The algorithm doesn’t change, it just returns a (slightly) different threshold – no training points lye on separating hyperplanes

Same performance on training set, usually slight improvement on test set
Overfitting

- One of the biggest problems with decision trees is Overfitting
Occam’s Razor

“If two theories explain the facts equally well, then the simpler theory is to be preferred”

Arguments in favor:
• Fewer short hypotheses than long hypotheses
• A short hypothesis that fits the data is unlikely to be a coincidence
• A long hypothesis that fits the data might be a coincidence

Arguments opposed:
• There are many ways to define small sets of hypotheses
Avoid Overfitting

Stop growing when one of the following is reached:

• Maximum depth
• Minimum number of examples associated to subtree
• Accuracy or MSE threshold

Pruning

• Replace subtrees by a leaf if doing so improves performance on a validation set
• Conver tree to a set of rules and
  • Prune rules (by removing preconditions) if doing so improves performance according to a validation set
  • Sort rules by performance on validation set (better rules first)
  • Apply rules in that order when making predictions for test examples.
Converting a Tree to Rules

R₁: If (Outlook=Sunny) \land (Humidity=High) Then PlayTennis=No
R₂: If (Outlook=Sunny) \land (Humidity=Normal) Then PlayTennis=Yes
R₃: If (Outlook=Overcast) Then PlayTennis=Yes
R₄: If (Outlook=Rain) \land (Wind=Strong) Then PlayTennis=No
R₅: If (Outlook=Rain) \land (Wind=Weak) Then PlayTennis=Yes