CS181 Lecture 7: Decision trees

Overview

• Learning decision trees (ID3)
• Decision tree extensions
• Training set performance
• Pruning

Decision trees

• Classifier model
• Nodes: attributes
• Edges: attribute values
• Leaves: classes

ID3 Pseudocode

ID3(D,X) =
Let T be a new tree
If all instances in D have same class c
Label(T) = c; Return T
If X = φ or no attribute has positive information gain
Label(T) = most common class in D; return T
X ← attribute with highest information gain
Label(T) = X
For each value x of X
D_x ← instances in D with X = x
If D_x is empty
Let T_x be a new tree
Label(T_x) = most common class in D_x
Else
T_x = ID3(D_x, X \{ x \})
Add a branch from T to T_x labeled by x
Return T

Splitting criterion

• Entropy of data
  - measures disorder of data
• Remainder
  - expected disorder of data after splitting
• Information gain
  - decrease in disorder after splitting on attribute

Example

D =

<table>
<thead>
<tr>
<th>X_1</th>
<th>X_2</th>
<th>X_3</th>
<th>X_4</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>P</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>P</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>P</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>N</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>N</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>N</td>
<td></td>
</tr>
</tbody>
</table>

X = \{X_1,X_2,X_3,X_4\}
Example

\[
X = \{X_1, X_2, X_3, X_4\}
\]

Consider \(X_1\)

\[
\begin{array}{cccc}
X_1 & X_2 & X_3 & X_4 & C \\
F & F & F & F & P \\
F & F & T & T & P \\
F & T & F & T & P \\
T & T & T & F & P \\
T & F & F & F & N \\
T & T & T & T & N \\
T & T & T & F & N \\
\end{array}
\]

\[
D = \text{Entropy}(D) = \text{Entropy}(4 : 3) = -\frac{4}{7} \log_2 \frac{4}{7} - \frac{3}{7} \log_2 \frac{3}{7} = 0.98
\]

\[
X = \{X_1, X_2, X_3, X_4\}
\]

Consider \(X_2\)

\[
\begin{array}{cccc}
X_1 & X_2 & X_3 & X_4 & C \\
F & F & F & F & P \\
F & F & T & T & P \\
F & T & F & T & P \\
T & T & T & F & P \\
T & F & F & F & N \\
T & T & T & T & N \\
T & T & T & F & N \\
\end{array}
\]

\[
D = \text{Entropy}(D) = 0.98
\]

\[
X = \{X_1, X_2, X_3, X_4\}
\]

First split

\[
\begin{array}{cccc}
X_1 & X_2 & X_3 & X_4 & C \\
F & F & F & F & P \\
F & F & T & T & P \\
T & T & T & F & P \\
T & F & F & F & N \\
T & T & T & T & N \\
T & T & T & F & N \\
\end{array}
\]

\[
D = X_1
\]

\[
X = \{X_1, X_2, X_3, X_4\}
\]

\[
\text{Gain}(X_1) = 0.52
\]

\[
\text{Gain}(X_2) = 0.01
\]

\[
\text{Gain}(X_3) = 0.01
\]

\[
\text{Gain}(X_4) = 0.01
\]

Consider \(X_1\)

\[
D = \begin{array}{c}
X_1 \\
F \\
T \\
\end{array}
\]

\[
X = \{X_1, X_2, X_3, X_4\}
\]

\[
\text{Entropy}(D) = 0.98
\]

\[
X = \{X_1, X_2, X_3, X_4\}
\]

Left side

\[
\begin{array}{cccc}
X_1 & X_2 & X_3 & X_4 & C \\
F & F & F & F & P \\
F & F & T & T & P \\
F & T & F & T & P \\
\end{array}
\]

\[
D = \text{Entropy}(D) = \text{Entropy}(3 : 0) = -\frac{3}{4} \log_2 \frac{3}{4} - \frac{1}{4} \log_2 \frac{1}{4} = 0.0
\]

\[
X = \{X_1, X_2, X_3, X_4\}
\]

\[
\text{Entropy}(1 : 3) = -\frac{1}{4} \log_2 \frac{1}{4} - \frac{3}{4} \log_2 \frac{3}{4} = 0.81
\]

All instances have the same class.

Return class P.
Right side

\[ D = \begin{array}{cccc}
X_1 & X_2 & X_3 & X_4 \\
T & T & T & T \\
T & F & F & F \\
T & T & T & F \\
\end{array} \]

\[ X = \{X_2, X_3, X_4\} \]

All attributes have same information gain.
Break ties arbitrarily.
Choose \( X_2 \)

So far...

\[ X_1 \]

\[ X_2 \]

\[ X_3 \]

\[ X_4 \]

Left side

\[ D = \begin{array}{cccc}
X_1 & X_2 & X_3 & X_4 \\
T & T & F & F \\
T & T & F & F \\
T & T & T & T \\
\end{array} \]

\[ X = \{X_3, X_4\} \]

All instances have the same class.
Return class N.

Right side

\[ D = \begin{array}{cccc}
X_3 & X_4 \\
T & T \\
T & T \\
\end{array} \]

\[ X = \{X_3, X_4\} \]

\( X_3 \) has zero information gain
\( X_4 \) has positive information gain
Choose \( X_4 \)

So far...

\[ X_1 \]

\[ X_2 \]

\[ X_3 \]

\[ X_4 \]

Left side

\[ D = \begin{array}{cccc}
X_3 & X_4 \\
T & T \\
T & T \\
\end{array} \]

\[ X = \{X_3\} \]

\( X_3 \) has zero information gain
No suitable attribute for splitting
Return most common class (break ties arbitrarily)
Note: data is inconsistent!
All instances have the same class. Return N.

Entropy revisited

Entropy as bits

• Let P be some distribution
• Suppose you’re sending bits down a wire to encode samples from P
• Suppose you use an optimal encoding of P
• How many bits on average will you send per message?
• This is the entropy of P

Example

• Suppose P assigns probability 1 to outcome T
• I tell you once that everything is T
• I tell you nothing about individuals
• Entropy is 0

Example

• Supposed P assigns probability 0.5 to T and F
• I can’t tell you anything useful in advance
• For each item, I have to tell you T or F
• Entropy is 1
Information gain intuition

- Maximize the orderliness of the data
- Minimize the number of bits required to encode the data

Observations about entropy

- Derivative is infinite at 0 and 1
- Small changes near 0/1 produce large changes in entropy
  - Entropy(0.14) = 0.59, Entropy(0) = 0
- Small changes near 0.5 produce small changes in entropy
  - Entropy(0.5) = 1, Entropy(0.36) = 0.94

Inductive bias of ID3

Given the choice between

\[
\begin{array}{c|c|c}
X_1 & F & T \\
\hline
2:4 & 1:1
\end{array}
\quad
\begin{array}{c|c|c}
X_2 & F & T \\
\hline
3:3 & 2:0
\end{array}
\]

which will ID3 prefer?

- Remainder = 0.94
- Remainder = 0.75

ID3 strongly prefers extreme partitions.

Inductive bias of ID3

Given the choice between

\[
\begin{array}{c|c|c|c|c}
X_2 & X_3 & F & T & F \\
\hline
3:3 & T & 2:2 & 4:0
\end{array}
\quad
\begin{array}{c|c|c|c|c}
X_3 & F & T & F \\
\hline
2:2 & 4:0 & 1:1
\end{array}
\]

which will ID3 prefer?

- Remainder = 0.75
- Remainder = 0.5

ID3 prefers to classify large subsets well

Inductive bias of ID3

Given the choice between

\[
\begin{array}{c|c|c|c|c}
X_3 & X_4 & F & T & F \\
\hline
2:2 & 4:0 & 1:1
\end{array}
\quad
\begin{array}{c|c|c|c|c}
X_4 & F & T & F \\
\hline
1:1 & 5:1
\end{array}
\]

which will ID3 prefer?

- Remainder = 0.5
- Remainder = 0.74

But preference for extreme partitions is stronger

Inductive bias of ID3

- Will ID3 always find the shortest tree?
- If we want the shortest tree, what could we do?
  - breadth-first search over decision trees
- What is real inductive bias of ID3?
  - trees that have higher information gain attributes nearer the root are preferred over those that do not
- **Heuristic** that tends to prefer shorter trees
An example

• Imagine class is N iff $X_1 = X_2$
• Here’s some example data

<table>
<thead>
<tr>
<th>$X_1$</th>
<th>$X_2$</th>
<th>$X_3$</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>N</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>P</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>P</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>N</td>
</tr>
</tbody>
</table>

• What will ID3 do?

Correct tree

N iff $X_1 = X_2$

![Decision Tree Diagram]

Example

• Gain($X_1$)?
  - 0.0
• Gain($X_2$)?
  - 0.0
• Gain($X_3$)?
  - 0.31
• Split on $X_3$

Example

• What if there are many features?
  - ID3 will likely split on all of them: big tree
• Will it ever split on $X_1$ or $X_2$?
  - depends on data, but maybe not
• Combinations of attributes that lead to good classification are **copredictors**
• We have a problem with copredictors...

Model vs. algorithm

• Is this a problem with our model or our algorithm?
  - algorithm (ID3). Our model (decision tree) is fine for representing copredictors, but ID3 won’t make the right splits
• ID3 is greedy: doesn’t look ahead
• Moral: need to think about algorithm, not just model

Overview

• Learning decision trees (ID3)
• Decision tree extensions
• Training set performance
• Pruning
Decision tree extensions

- Multi-class attributes
- Continuous-valued attributes
- Missing information
- Attribute costs
- Disjunctions

Multi-class attributes

\[
\text{ID3}(D, X) = \\
\begin{align*}
&\text{Let } T \text{ be a new tree} \\
&\text{If all instances in } D \text{ have same class } c \\
&\quad \text{Label}(T) = c; \ Return \ T \\
&\text{If } X = \emptyset \text{ or no attribute has positive information gain} \\
&\quad \text{Label}(T) = \text{most common class in } D; \ return \ T \\
&\quad X = \text{attribute with highest information gain} \\
&\quad \text{Label}(T) = X \\
&\text{For each value } x \text{ of } X \\
&\quad D_x \leftarrow \text{instances in } D \text{ with } X = x \\
&\quad \text{If } D_x \text{ is empty} \\
&\quad \quad \text{Let } T_x \text{ be a new tree} \\
&\quad \quad \text{Label}(T_x) = \text{most common class in } D \\
&\quad \text{Else} \\
&\quad \quad T_x = \text{ID3}(D_x, X - \{X\}) \\
&\quad \text{Add a branch from } T \text{ to } T_x \text{ labeled by } x \\
&\text{Return } T
\end{align*}
\]

Continuous-valued attributes

- Can’t have branch for each value
- Need threshold splits
- How do we choose what splits to consider?
  - can look at midpoint between each point
- Can make more than one split at a time

\[
\text{ID3}(D, X) = \\
\begin{align*}
&\text{Let } T \text{ be a new tree} \\
&\text{If all instances in } D \text{ have same class } c \\
&\quad \text{Label}(T) = c; \ Return \ T \\
&\text{If } X = \emptyset \text{ or no attribute has positive information gain} \\
&\quad \text{Label}(T) = \text{most common class in } D; \ return \ T \\
&\quad X = \text{attribute with highest information gain} \\
&\quad \text{Label}(T) = X \\
&\text{For each value } x \text{ of } X \\
&\quad D_x \leftarrow \text{instances in } D \text{ with } X = x \\
&\quad \text{If } D_x \text{ is empty} \\
&\quad \quad \text{Let } T_x \text{ be a new tree} \\
&\quad \quad \text{Label}(T_x) = \text{most common class in } D \\
&\quad \text{Else} \\
&\quad \quad T_x = \text{ID3}(D_x, X - \{X\}) \\
&\quad \text{Add a branch from } T \text{ to } T_x \text{ labeled by } x \\
&\text{Return } T
\end{align*}
\]

Continuous-value attributes

What happens if we split on the Date attribute?

<table>
<thead>
<tr>
<th>Date</th>
<th>Outlook</th>
<th>Temp</th>
<th>Humidity</th>
<th>Wind</th>
<th>Tennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>8/2/06</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>N</td>
</tr>
<tr>
<td>4/14/07</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>P</td>
</tr>
<tr>
<td>5/20/07</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>P</td>
</tr>
<tr>
<td>9/2/07</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>N</td>
</tr>
<tr>
<td>10/16/07</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>P</td>
</tr>
<tr>
<td>11/1/07</td>
<td>Sunny</td>
<td>Low</td>
<td>Low</td>
<td>Weak</td>
<td>P</td>
</tr>
<tr>
<td>3/7/08</td>
<td>Rain</td>
<td>Low</td>
<td>High</td>
<td>Weak</td>
<td>N</td>
</tr>
</tbody>
</table>

What happens if we split on the Date attribute?

<table>
<thead>
<tr>
<th>Date</th>
<th>Outlook</th>
<th>Temp</th>
<th>Humidity</th>
<th>Wind</th>
<th>Tennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>8/2/06</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>N</td>
</tr>
<tr>
<td>4/14/07</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>P</td>
</tr>
<tr>
<td>5/20/07</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>P</td>
</tr>
<tr>
<td>9/2/07</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>N</td>
</tr>
<tr>
<td>10/16/07</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>P</td>
</tr>
<tr>
<td>11/1/07</td>
<td>Sunny</td>
<td>Low</td>
<td>Low</td>
<td>Weak</td>
<td>P</td>
</tr>
<tr>
<td>3/7/08</td>
<td>Rain</td>
<td>Low</td>
<td>High</td>
<td>Weak</td>
<td>N</td>
</tr>
</tbody>
</table>

Date attribute can completely split data.
Split information

• Information gain favors attributes with many values
• But if attribute splits data evenly, it may not be the most informative attribute
• Split information estimate how uniformly the attribute splits the data:

\[ \text{Split Information}(D, X) = -\sum_x \frac{|D_x|}{|D|} \log_2 \frac{|D_x|}{|D|} \]

Split information

• Information gain favors attributes with many values
• But if attribute splits data evenly, it may not be the most informative attribute
• Split information estimate how uniformly the attribute splits the data:

\[ \text{Gain Ratio}(D, X) \equiv \frac{\text{Gain}(D, X)}{\text{Split Information}(D, X)} \]

• If \(|D_x| \approx |D|\), split information goes infinite
• Calculate information gain first, and compute gain ratio only for top attributes

Missing information

• What if we don’t have values for every attribute for every instance?
• Use similar instances to fill in value for attribute
• Similar: instances in node with same label

Missing information example

We have the following mammograph data to split

<table>
<thead>
<tr>
<th>BI-RAD</th>
<th>Age</th>
<th>Shape</th>
<th>Margin</th>
<th>Density</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>48</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>67</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>57</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>60</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>53</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>28</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>70</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>66</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>63</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>78</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Fill in the data according to most common (given class) Split as usual

<table>
<thead>
<tr>
<th>BI-RAD</th>
<th>Age</th>
<th>Shape</th>
<th>Margin</th>
<th>Density</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>48</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>67</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>57</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>60</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>53</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>28</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>70</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>66</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>63</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>78</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
**Missing information example**

Alternatively, divide instances according to proportions

<table>
<thead>
<tr>
<th>Fraction</th>
<th>BI-RAD</th>
<th>Age</th>
<th>Shape</th>
<th>Margin</th>
<th>Density</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.75</td>
<td>4</td>
<td>48</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0.25</td>
<td>4</td>
<td>48</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>5</td>
<td>67</td>
<td>3</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>5</td>
<td>60</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>0.66</td>
<td>4</td>
<td>53</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0.33</td>
<td>4</td>
<td>53</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0.66</td>
<td>4</td>
<td>53</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0.33</td>
<td>4</td>
<td>53</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1.0</td>
<td>4</td>
<td>70</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>0.75</td>
<td>4</td>
<td>70</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0.25</td>
<td>2</td>
<td>66</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>0.75</td>
<td>3</td>
<td>63</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>0.25</td>
<td>3</td>
<td>63</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>1.0</td>
<td>4</td>
<td>78</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

**Attribute costs**

- Attributes may be expensive to fill in
  - e.g. medical tests
- If possible, decide based on cheaper attributes
- Perhaps prefer cheaper attributes at top
- Build cost into splitting criterion
  - e.g. \( \frac{\text{Gain}(D,X)}{\text{Cost}(X)} \)

**Disjunctions**

Give a decision tree for \( X_1 \land X_2 \)

\[
\begin{array}{lllll}
\text{X}_1 & \text{F} & \text{T} \\
\text{X}_2 & \text{F} & \text{T} \\
\text{N} & \text{N} & \text{P} \\
\end{array}
\]

How about \( (X_1 \land X_2) \lor (X_3 \land X_4) \)?

\[
\begin{array}{lllll}
\text{X}_1 & \text{F} & \text{T} \\
\text{X}_2 & \text{F} & \text{T} \\
\text{X}_3 & \text{N} & \text{P} \\
\text{X}_4 & \text{F} & \text{T} \\
\text{N} & \text{N} & \text{P} \\
\end{array}
\]
Disjunctions

How about \((X_1 \land X_2) \lor (X_3 \land X_4)\)?

Can we avoid this repetition?

Decision graphs

• Disjunctions lead to large trees and data fragmentation
• Decision graphs allow nodes to be joined
• Smaller representation and better use of data
• Harder to construct: more choices (split and join)

Overview

• Learning decision trees (ID3)
• Decision tree extensions
• Training set performance
• Pruning
• Experimental methodology

Training set performance

• Given consistent training set, will ID3 produce classifier with zero error on training set?
• Consider termination conditions:
  - training set empty: zero error
  - all instances have same class: zero error
  - no attributes left to split on: impossible
  - no attribute has positive information gain: possible but unusual
• So, generally yes, occasionally no

Training set performance

• What about with inconsistent training data?
• Consider termination conditions again:
  - training set empty: zero error
  - all instances have same class: zero error
  - no attributes left to split on: inconsistent data. choosing most common class minimizes error
  - no attribute has positive information gain: possible but unusual
• In most cases, minimum training error

Is this good?

• Do we really care about training set performance?
• We really care about generalization
• Conjecture: If hypothesis A has lower training error than hypothesis B, it should generalize better to unseen instances.
• Wrong.
Overfitting

Patterns in data

- True pattern in domain
  - present in large amounts of data
  - generalizes to unseen instances
- Spurious pattern in training set
  - "noise in the data"
  - present in small amounts of data
  - does not generalize

ID3 learning process

- Beginning
  - lots of data
  - discovers true patterns in data
- Later
  - small amount of data
  - likely to learn spurious patterns

Overfitting example

Correct tree:

Training set:

What is the training set error? 0

Possible learned tree:

Training set:

What is the training set error? 2

Overfitting example

Possible learned tree:

Training set:

What is the training set error? 2
Overfitting example

Correct tree:  
\[ \begin{array}{c|c|c|c|c} & X_1 & F & T & P \\ \hline X_2 & P & & & \ \\ \hline X_3 & F & P & T & N \\ \hline X_4 & T & F & & P \\ \hline \ \\ \end{array} \]

Possible trees:  
\[ \begin{array}{c|c|c|c|c} & X_1 & F & T & P \\ \hline X_2 & P & & & \ \\ \hline X_3 & F & P & T & N \\ \hline X_4 & T & F & & P \\ \hline \ \\ \end{array} \]

Classify \langle TTTF \rangle ?

Causes of overfitting

1. Small training set
2. Non-deterministic domain
3. Weak inductive bias
4. Many features

Extreme example

- 1000 features
- True concept is \( X_1 \lor X_2 \)
- 8 training examples
- For a given attribute \( X_i \) (\( i > 2 \)), what is likelihood that \( X_i \) perfectly classifies the training data?
  - 1/128
- Extremely likely that irrelevant feature will perfectly classify the data
- ID3 will return a tree consisting only of one feature

Dealing with overfitting

Examine causes:
- small training set
  - may be expensive to increase
- non-deterministic domain
  - can’t avoid
- many features
  - feature selection can help, but doesn’t fully solve
- weak inductive bias
  - can increase bias
  - restriction or preference bias

Bias tradeoff

- Weaker inductive bias:
  - more influenced by data
  - can learn patterns in data
  - may overfit

- Stronger inductive bias:
  - less influenced by data
  - avoids overfitting
  - may fail to learn true pattern: underfit

Increasing inductive bias

- Restriction bias
  - restrict to smaller hypothesis space
  - e.g. allow only trees of depth \( \leq 3 \)

- Preference bias
  - prefer simpler hypotheses even with higher training error
  - e.g. prefer shorter trees even if they don’t fit the data as well
Overview

- Learning decision trees (ID3)
- Decision tree extensions
- Training set performance
- Pruning
- Experimental methodology

Pruning

- Remove a subtree
- Replace with a leaf
- Class of leaf is most common class of data in subtree

Pruning example

<table>
<thead>
<tr>
<th>X₁</th>
<th>X₂</th>
<th>X₃</th>
<th>X₄</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>P</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>P</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>P</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>P</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>N</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>N</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>N</td>
</tr>
</tbody>
</table>

When to prune

- Pre-pruning
  - prune the tree before you grow it
  - adds a termination condition to ID3
  - saves computation time
- Post-pruning
  - grow complete tree and prune it afterward
  - takes into account more information

Pruning methods

1. Validation set
2. Rule post-pruning
3. Chi-squared pruning
4. Minimum description length

Validation set pruning

- Save a portion of the data for validation

<table>
<thead>
<tr>
<th>Training set</th>
<th>Validation set</th>
<th>Test set</th>
</tr>
</thead>
</table>

- Would pruning achieve better performance on validation set?
Validation set pruning

• Learn complete tree
• Consider each node:
  - \( s \leftarrow \) validation set performance with subtree at node
  - \( t \leftarrow \) validation set performance with subtree replaced with leaf
• If \( s \leq t \), prune subtree

Rule post-pruning

• Convert decision tree into rules
• Prune rules according to validation set
  - can use heuristic pessimistic estimate on training set
• Rank rules according to estimated accuracy
• Use rules to classify new instances

Rule post-pruning example

\[
\begin{align*}
\text{Outlook} & \quad \text{Humidity} \quad \text{Wind} \\
\text{Sunny} & \quad \text{P} \quad \text{P} \\
\text{Overcast} & \quad \text{P} \quad \text{N} \\
\text{Rain} & \quad \text{P} \quad \text{P}
\end{align*}
\]

(Outlook=Sunny ∧ Humidity=High) \(\Rightarrow\) N

All rules

(Outlook = Sunny ∧ Humidity = High) \(\Rightarrow\) N
(Outlook = Sunny ∧ Humidity = Low) \(\Rightarrow\) P
(Outlook = Overcast) \(\Rightarrow\) P
(Outlook = Rain ∧ Wind = Strong) \(\Rightarrow\) N
(Outlook = Rain ∧ Wind = Weak) \(\Rightarrow\) P

Prune preconditions

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temp</th>
<th>Humidity</th>
<th>Wind</th>
<th>Tennis?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rain</td>
<td>Low</td>
<td>High</td>
<td>Weak</td>
<td>N</td>
</tr>
<tr>
<td>Rain</td>
<td>High</td>
<td>High</td>
<td>Strong</td>
<td>N</td>
</tr>
</tbody>
</table>
Prune preconditions

(Outlook = Sunny ∧ Humidity = High) ⇒ N
(Outlook = Sunny ∧ Humidity = Low) ⇒ P
(Outlook = Overcast) ⇒ P
(Outlook = Rain ∧ Wind = Weak) ⇒ P
(Outlook = Rain ∧ Wind = Strong) ⇒ N

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temp</th>
<th>Humidity</th>
<th>Wind</th>
<th>Tennis?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rain</td>
<td>Low</td>
<td>High</td>
<td>Weak</td>
<td>N</td>
</tr>
<tr>
<td>Rain</td>
<td>High</td>
<td>High</td>
<td>Strong</td>
<td>N</td>
</tr>
</tbody>
</table>

Prune preconditions

(Outlook = Sunny ∧ Humidity = High) ⇒ N
(Outlook = Sunny ∧ Humidity = Low) ⇒ P
(Outlook = Overcast) ⇒ P
(Outlook = Rain ∧ Wind = Weak) ⇒ P
(Outlook = Rain) ⇒ N

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temp</th>
<th>Humidity</th>
<th>Wind</th>
<th>Tennis?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rain</td>
<td>Low</td>
<td>High</td>
<td>Weak</td>
<td>N</td>
</tr>
<tr>
<td>Rain</td>
<td>High</td>
<td>High</td>
<td>Strong</td>
<td>N</td>
</tr>
</tbody>
</table>

Sort according to accuracy

(Outlook = Sunny ∧ Humidity = High) ⇒ N
(Outlook = Sunny ∧ Humidity = Low) ⇒ P
(Outlook = Overcast) ⇒ P
(Outlook = Rain) ⇒ N
(Outlook = Rain ∧ Wind = Weak) ⇒ P

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temp</th>
<th>Humidity</th>
<th>Wind</th>
<th>Tennis?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rain</td>
<td>Low</td>
<td>High</td>
<td>Weak</td>
<td>N</td>
</tr>
<tr>
<td>Rain</td>
<td>High</td>
<td>High</td>
<td>Strong</td>
<td>N</td>
</tr>
</tbody>
</table>

Classification with rules

(Outlook = Sunny ∧ Humidity = High) ⇒ N
(Outlook = Sunny ∧ Humidity = Low) ⇒ P
(Outlook = Overcast) ⇒ P
(Outlook = Rain) ⇒ N
(Outlook = Rain ∧ Wind = Weak) ⇒ P

Classify new instances:

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temp</th>
<th>Humidity</th>
<th>Wind</th>
<th>Tennis?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sunny</td>
<td>Low</td>
<td>Low</td>
<td>Weak</td>
<td>P</td>
</tr>
<tr>
<td>Rain</td>
<td>High</td>
<td>High</td>
<td>Weak</td>
<td>N</td>
</tr>
</tbody>
</table>

Advantages of rule pruning

• Can remove smaller elements than whole subtrees
• Easier bookkeeping: can remove distinctions from near top of tree
• Improved readability

Chi-squared

• Before splitting, ask:
  - Is the pattern found in the data after splitting statistically significant?
• Pre-pruning method
  - only need distribution of classes at node
Proposed split

- Given instances $D$
- Propose split on $X_i$
- Notation
  - $N_c = \text{number of instances with class } c$
  - $D_x = \text{data set with value } x \text{ for attribute } X_i$
  - $N_x = \text{number of instances in } D_x$
  - $N_{xc} = \text{number of instances in } D_x \text{ with class } c$

Absence of pattern

- Null hypothesis: $X_i$ is irrelevant
- In $D$, proportion with class $c$: $N_c/N$
- If null hypothesis is true, we expect on average the number of instances in $D_x$ with class $c$ to be:
  $$\hat{N}_{xc} = \frac{N_c}{N} |D_x|$$

Deviation

- We don’t expect to see exactly that many, even if null hypothesis is true
- We expect some deviation due to random chance
- Measure the deviation from total absence of pattern:
  $$\text{Dev} = \sum_x \sum_c \left( \frac{N_{xc} - \hat{N}_{xc}}{\hat{N}_{xc}} \right)^2$$

Using deviation

- Dev is the chi-squared statistic
  - larger Dev, more pattern in data
- Basic principle: prune if Dev is small
- How small is small?
- Convert Dev into probability
- With what probability (under null hypothesis) will we see a chi-squared statistic bigger than Dev?

Using the probability

- Dev is chi-squared statistic
  - distributed according to chi-squared with $|X_i|-1$ DOF
- Can compute probability using statistical packages (MATLAB, for example), or table
- If Dev is large, probability will be small
  - amount of pattern is rare for null hypothesis
- Decision rule: accept split if probability < $q$
  - what should $q$ be?

Pros and Cons

- Validation set and rule post-pruning:
  - uses all available information (post-pruning)
  - requires reserving some data
- Chi-squared
  - can use all data for training
  - does not consider all information in subtree
  - statistical test becomes less valid with less data
  - requires specifying cutoff $q$
Minimum description length

- Explicitly trade off complexity and error
  \[ \text{cost}(h, D) = f_1(\text{complexity}(h)) + f_2(\text{error}(h, D)) \]
- Choose hypothesis with lowest cost
- Need complexity and error on same scale

Description length

- \[ \text{cost}(h, D) = f_1(\text{complexity}(h)) + f_2(\text{error}(h, D)) \]
- \( f_1 \) = number of bits needed to encode \( h \)
- \( f_2 \) = number of bits needed to encode data
- Minimum description length principle: choose hypothesis that requires minimum total bits

Next time

- Experimental methodology
- Computational learning theory