Decision Tree For Playing Tennis

ROOT NODE

BRANCH

INTERNAL NODE

LEAF NODE

Disjunction of conjunctions

Outlook

Sunny

Overcast

Rain

Humidity

High

Normal

Wind

Strong

Weak

No

Yes

No

Yes
Another Perspective of a Decision Tree Model

Age, ... income
Case A. 30, ..., $110K, Default
Case B. 50, ..., $110K, NoDefault
Case C. 45, ..., $90K, NoDefault
Case A. 32, ..., $105K, Default
Case B. 49, ..., $82K, NoDefault
Case C. 29, ..., $50K, NoDefault
Top-Down Tree Induction

Main loop:
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 of node
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

Which attribute is best?

![Diagram](image-url)
Which Column and Split Point?

- Multitude of techniques:
  - Entropy/Information gain
  - Chi square test (CHAID)
    - Test of independence
  - GINI index
Information Gain

\[ \text{Gain}(S, A) = \text{expected reduction in entropy due to sorting on } A \]

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

\[ \text{Entropy}(S) = \text{expected number of bits needed to encode class } (\oplus \text{ or } \ominus) \text{ of randomly drawn member of } S \text{ (under the optimal, shortest-length code)} \]

Why?

Information theory: optimal length code assigns \(- \log_2 p\) bits to message having probability \(p\).

So, expected number of bits to encode \(\oplus\) or \(\ominus\) of random member of \(S\):

\[ p_\oplus(- \log_2 p_\oplus) + p_\ominus(- \log_2 p_\ominus) \]

\[ \text{Entropy}(S) \equiv -p_\oplus \log_2 p_\oplus - p_\ominus \log_2 p_\ominus \]
## Data Set

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis</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>Strong</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>Cool</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>Weak</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>
Choosing the Next Attribute - 1

Which attribute is the best classifier?

\( S: [9+5] \)

\[ E = 0.940 \]

- **Humidity**
  - **High**
    - \([3+4] \)
      \[ E = 0.985 \]
  - **Normal**
    - \([6+1] \)
      \[ E = 0.592 \]

\[ \text{Gain} (S, \text{Humidity}) = 0.940 - (7/14)0.985 - (7/14)0.592 = 0.151 \]

\( S: [9+5] \)

\[ E = 0.940 \]

- **Wind**
  - **Weak**
    - \([6+2] \)
      \[ E = 0.811 \]
  - **Strong**
    - \([3+3] \)
      \[ E = 1.00 \]

\[ \text{Gain} (S, \text{Wind}) = 0.940 - (8/14)0.811 - (6/14)1.0 = 0.048 \]
Choosing the Next Attribute - 2

Which attribute should be tested here?

$S_{\text{Sunny}} = \{D1, D2, D8, D9, D11\}$

$\text{Gain}(S_{\text{Sunny}}, \text{Humidity}) = .970 - (3/5)0.0 - (2/5)0.0 = .970$

$\text{Gain}(S_{\text{Sunny}}, \text{Temperature}) = .970 - (2/5)0.0 - (2/5)1.0 - (1/5)0.0 = .570$

$\text{Gain}(S_{\text{Sunny}}, \text{Wind}) = .970 - (2/5)1.0 - (3/5).918 = .019$
Representational and Search Bias

Note $H$ is the power set of instances $X$

→ Unbiased?

Not really...

- Preference for short trees, and for those with high information gain attributes near the root
- Bias is a preference for some hypotheses, rather than a restriction of hypothesis space $H$
- Occam’s razor: prefer the shortest hypothesis that fits the data
Occam’s Razor

• 14th Century Franciscan friar; William of Occam.
• The principle states that "Entities should not be multiplied unnecessarily."
• People often reinvented Occam's Razor
  – Newton - "We are to admit no more causes of natural things than such as are both true and sufficient to explain their appearances."
• To most scientist the razor is:
  – "when you have two competing theories which make exactly the same predictions, the one that is simpler is the better."
Review of Choosing a Split

Entropy = $\sum -p \cdot \log_2(p)$

Entropy_{Population} = 1

Entropy_{Split on Length} = 0.42
Entropy_{Split on Thread} = 0.85
Stopping Criteria

• What type of tree will perfectly classify the training data (ie. 100% training set accuracy)?
• Is this a bad thing?, Why?
• What does this tell you about the relationship between the dependent and independent attributes?
• Stop growing the tree when:
  – A certain tree depth is reached
  – Number of records at a node goes below some threshold.
  – All potential splits are insignificant
How Do We Know When We’ve Overfitted The Training Data?

Consider error of hypothesis $h$ over

- training data: $error_{train}(h)$
- entire distribution $D$ of data: $error_D(h)$

Hypothesis $h \in H$ overfits training data if there is an alternative hypothesis $h' \in H$ such that

$$error_{train}(h) < error_{train}(h')$$

and

$$error_D(h) > error_D(h')$$

Is there any other way?
Training Set Error Should Approximately Equal Test Set Error
Trimming/Pruning Trees

• Stopping criterion can be somewhat arbitrary.

• Automatic pruning of trees
  – Ask the data, “How far should we split the data”.
  – Two general approaches:
    • Use part of the training set as a validation set
    • Use entire training set (usually an MDL approach).
Using Pruning To Prevent Overfitting

How can we avoid overfitting?

- stop growing when data split not statistically significant
- grow full tree, then post-prune

How to select “best” tree:

- Measure performance over training data
- Measure performance over separate validation data set
- MDL: minimize
  \[ \text{size}(\text{tree}) + \text{size}(	ext{misclassifications}(\text{tree})) \]
Reduced Error Pruning

Split data into *training* and *validation* set

Do until further pruning is harmful:

1. Evaluate impact on *validation* set of pruning each possible node (plus those below it)

2. Greedily remove the one that most improves *validation* set accuracy

- produces smallest version of most accurate subtree
- What if data is limited?
Reduced Error Pruning
Consider the use of learning a tree is to make prediction
What is the fundamental assumption that this learning algorithm is making
Rule Post-Pruning

1. Convert tree to equivalent set of rules
2. Prune each rule independently of others
3. Sort final rules into desired sequence for use

Perhaps most frequently used method (e.g., C4.5)
X-Fold Cross Validation

– Used to estimate the accuracy of the learner.
– Feature selection for other supervised learning algorithms.

<table>
<thead>
<tr>
<th>Fold 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fold 2</td>
</tr>
<tr>
<td>Fold 3</td>
</tr>
<tr>
<td>Fold 4</td>
</tr>
<tr>
<td>Fold 5</td>
</tr>
</tbody>
</table>
MDL Base Pruning

- Minimize Overall Message Length
  - \( \text{MessLen(Model, Data)} = \text{MessLen(Model)} + \text{MessLen(Data | Model)} \)
- Encode model using node encoding.
- Encode model in terms of classification error.
- Remove a node if it reduces the cost.
Ensemble of Decision Trees

• Why stop at one decision tree.
• Adopt the committee of experts approach
• Build multiple decision trees, each votes on the classification, highest vote wins.
• What problem will we run up against?
Why Does it Work?

• Brieman
  – Works because decision tree learners are unstable.

• Friedman
  – Reduces the variance of the learner without reducing bias.

• Domingos
  – Underlying learners bias towards simplicity is too great
  – Bagging corrects bias.
C4.5 - Quinlan

- Goto http://www.cse.unsw.edu.au/~quinlan/
- Download C4.5 Release 8
- Need to untar it (use tar –xvf)
- In R8/Src type “make all”, builds c4.5 executable
- May need to remove contents of getopt.c file.
- Use “nroff doc/c4.5.1 | more” to read documentation.
- See me during office hours if you have any problems.
Building a Model Using C4.5

• Options
  • c4.5 - form [ -f filestem] [ -u ] [ -s ] [ -p ] [ -v verb ] [ -t trials ]
    [ -w wsize ] [ -i incr ] [ -g ] [ -m minobjs ] [ -c cf ]

• C4.5 –f golf –m 2
  outlook = overcast: Play (4.0)
  outlook = sunny:
    | humidity <= 75 : Play (2.0)
    | humidity > 75 : Don't Play (3.0)
  outlook = rain:
    | windy = true: Don't Play (2.0)
    | windy = false: Play (3.0)

Size Errors
8 0 (0.0%)
Building and Applying a Model Using C4.5

• Many data sets in the Data directory can are split into .data (training set) and .test (test set).

• Use c4.5 –f <name> -u
  – To build a model and then test it on the training set.
  – (use labor-neg or vote datasets).
Model Uncertainty

• What’s wrong with making predictions from one model?
  – May have two or more equally accurate models that give different predictions.
  – May have two models that are quite fundamentally different
Ensemble of Models Techniques

• Bayesian Modeling Averaging
  – \( \Pr(c,x \mid D, H) = \sum_{h \in H} \Pr(c,x \mid h) \cdot \Pr(h \mid D) \)
  – Weight each model’s prediction by how good the model is.
  – Can this approach be applied to C4.5 Dtrees?

• Boosting (Bootstrap Aggregation), 1996.
  – Improves accuracy
    • Seminal paper says on 19 of 26 data sets improves accuracy by 4%.
Bagging

• Take a number of bootstrap samples of the training set.
• Build a decision tree from each
• When predicting the category for a test set instance:
  – Each tree gets to vote on the decision
  – Ties are resolved by choosing the most populous class
• Empirical evidence shows that you get consistently better results on most data set.
The Bagging Algorithm

• Building the Models
  For $i = 1$ to $k$ // $k$ is the number of bags
    $T_i = \text{BootStrap}(D)$ // $D$ is the training set
    Build Model $M_i$ from $T_i$ (ie. Induce the tree)
  End

• Applying the Models To Make a Prediction
  For a test set example, $x$
  For $i = 1$ to $k$ // $k$ is the number of bags
    $C_i = M_i(x)$
  End
  Prediction is the class with the most vote.
Take A Bootstrap Sample

Sample with replacement
Bootstrapping and model building can be easily parallelized

<table>
<thead>
<tr>
<th>Original</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training set 1</td>
<td>2</td>
<td>7</td>
<td>8</td>
<td>3</td>
<td>7</td>
<td>6</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Training set 2</td>
<td>7</td>
<td>8</td>
<td>5</td>
<td>6</td>
<td>4</td>
<td>2</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>Training set 3</td>
<td>3</td>
<td>6</td>
<td>2</td>
<td>7</td>
<td>5</td>
<td>6</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Training set 4</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>8</td>
</tr>
</tbody>
</table>
### Bagging - Results

<table>
<thead>
<tr>
<th></th>
<th>C4.5</th>
<th>Bagged C4.5 vs C4.5</th>
<th>Boosted C4.5 vs C4.5</th>
<th>Boosting vs Bagging</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>err (%)</td>
<td>err (%)</td>
<td>w-l</td>
<td>ratio</td>
</tr>
<tr>
<td>anneal</td>
<td>7.67</td>
<td>6.25</td>
<td>10-0</td>
<td>.814</td>
</tr>
<tr>
<td>audiology</td>
<td>22.12</td>
<td>19.29</td>
<td>9-0</td>
<td>.872</td>
</tr>
<tr>
<td>auto</td>
<td>17.66</td>
<td>19.66</td>
<td>2-8</td>
<td>1.113</td>
</tr>
<tr>
<td>breast-w</td>
<td>5.28</td>
<td>4.23</td>
<td>9-0</td>
<td>.802</td>
</tr>
<tr>
<td>chess</td>
<td>8.55</td>
<td>8.33</td>
<td>6-2</td>
<td>.975</td>
</tr>
<tr>
<td>colic</td>
<td>14.92</td>
<td>15.19</td>
<td>0-6</td>
<td>1.018</td>
</tr>
<tr>
<td>credit-a</td>
<td>14.70</td>
<td>14.13</td>
<td>8-2</td>
<td>.962</td>
</tr>
<tr>
<td>credit-g</td>
<td>28.44</td>
<td>25.81</td>
<td>10-0</td>
<td>.908</td>
</tr>
<tr>
<td>diabetes</td>
<td>25.39</td>
<td>23.63</td>
<td>9-1</td>
<td>.931</td>
</tr>
<tr>
<td>glass</td>
<td>32.48</td>
<td>27.01</td>
<td>10-0</td>
<td>.852</td>
</tr>
<tr>
<td>heart-c</td>
<td>22.94</td>
<td>21.52</td>
<td>7-2</td>
<td>.938</td>
</tr>
<tr>
<td>heart-h</td>
<td>21.53</td>
<td>20.31</td>
<td>8-1</td>
<td>.943</td>
</tr>
<tr>
<td>hepatitis</td>
<td>20.39</td>
<td>18.52</td>
<td>9-0</td>
<td>.908</td>
</tr>
<tr>
<td>hypo</td>
<td>.48</td>
<td>.45</td>
<td>7-2</td>
<td>.928</td>
</tr>
<tr>
<td>iris</td>
<td>4.80</td>
<td>5.13</td>
<td>2-6</td>
<td>1.069</td>
</tr>
<tr>
<td>labor</td>
<td>19.12</td>
<td>14.39</td>
<td>10-0</td>
<td>.752</td>
</tr>
<tr>
<td>letter</td>
<td>11.99</td>
<td>7.51</td>
<td>10-0</td>
<td>.626</td>
</tr>
<tr>
<td>lymphography</td>
<td>21.69</td>
<td>20.41</td>
<td>8-2</td>
<td>.941</td>
</tr>
<tr>
<td>phoneme</td>
<td>19.44</td>
<td>18.73</td>
<td>10-0</td>
<td>.964</td>
</tr>
<tr>
<td>segment</td>
<td>3.21</td>
<td>2.74</td>
<td>9-1</td>
<td>.893</td>
</tr>
<tr>
<td>sick</td>
<td>1.34</td>
<td>1.22</td>
<td>7-1</td>
<td>.907</td>
</tr>
<tr>
<td>sonar</td>
<td>25.62</td>
<td>23.80</td>
<td>7-1</td>
<td>.929</td>
</tr>
<tr>
<td>soybean</td>
<td>7.73</td>
<td>7.58</td>
<td>6-3</td>
<td>.981</td>
</tr>
<tr>
<td>splice</td>
<td>5.91</td>
<td>5.58</td>
<td>9-1</td>
<td>.943</td>
</tr>
<tr>
<td>vehicle</td>
<td>27.09</td>
<td>25.54</td>
<td>10-0</td>
<td>.943</td>
</tr>
<tr>
<td>vote</td>
<td>5.06</td>
<td>4.37</td>
<td>9-0</td>
<td>.864</td>
</tr>
<tr>
<td>waveform</td>
<td>27.33</td>
<td>19.77</td>
<td>10-0</td>
<td>.723</td>
</tr>
<tr>
<td><strong>average</strong></td>
<td><strong>15.66</strong></td>
<td><strong>14.11</strong></td>
<td><strong>.905</strong></td>
<td><strong>13.36</strong></td>
</tr>
</tbody>
</table>
Example of Bagging

Problem

100 DT’s

Single DT Solution

Bagging Solution
Boosting – The Idea

• Take weak learners (marginally better than random guessing) make them stronger.
• Freund and Schapire, 95 – AdaBoost
• AdaBoost premise
  – Each training instances has equal weight
  – Build first Model from training instances
  – Training instances that are classified incorrectly given more weight
  – Build another model with re-weighted instances and so on and so on.
Boosting Pseudo Code

- Initialize distribution over the training set $D_1(i) = 1/m$
- For $t = 1, \ldots, T$:
  1. Train Weak Learner using distribution $D_t$.
  2. Choose a weight (or confidence value) $\alpha_t \in \mathbb{R}$.
  3. Update the distribution over the training set:
     \[
     D_{t+1}(i) = \frac{D_t(i) e^{-\alpha_t y_i h_t(x_i)}}{Z_t} \quad (2)
     \]
     Where $Z_t$ is a normalization factor chosen so that $D_{t+1}$ will be a distribution.
- Final vote $H(x)$ is a weighted sum:
  \[
  H(x) = \text{sign}(f(x)) = \text{sign} \left( \sum_{t=1}^{T} \alpha_t h_t(x) \right) \quad (3)
  \]
Some Implementation Comments

• Difficult to parallelize
• Factoring instance weights into decision tree induction.
• Tree vote is weighted inversely to error.
• Adaptive Boosting (AdaBoosting) according to the tree error
• Free scaled down version of C5.0 incorporates boosting available at http://www.rulequest.com/download.html
Toy Example (Freund COLT 99)
Round 1
Round 2 + 3

$\epsilon_2 = 0.21$
$\alpha_2 = 0.65$

$\epsilon_3 = 0.14$
$\alpha_3 = 0.92$
Final Hypothesis

\[ H_{\text{final}} = \text{sign}(0.42 + 0.65 + 0.92) \]

Some Insights into Boosting

• Final aggregate model will have no training error (given some conditions).
• Seems to over-fit but reduces test set error
• Larger margins on training set correspond to better generalization error
  – Margin(x) = y Σ α_j h_j(x) / Σ α_j
The Performance of Models and Learners

• Error of the hypothesis vs error of the learning algorithm?
• Know the training and test set error, good estimate of the learner’s performance?
• Learners Error = noise + bias^2 + variance
• How we calculate bias and variance for a learner*
  – T_{1...n}: Training sets drawn randomly from population
• Bias is the difference in error over all training sets – true error.
• Variance is the variability of the error.
• Why would a decision tree be biased? Have a high variance?
Errors

The true error of hypothesis $h$ with respect to target function $f$ and distribution $\mathcal{D}$ is the probability that $h$ will misclassify an instance drawn at random according to $\mathcal{D}$.

$$error_{\mathcal{D}}(h) \equiv \Pr_{x \in \mathcal{D}}[f(x) \neq h(x)]$$

The sample error of $h$ with respect to target function $f$ and data sample $S$ is the proportion of examples $h$ misclassifies

$$error_{S}(h) \equiv \frac{1}{n} \sum_{x \in S} \delta(f(x) \neq h(x))$$

Where $\delta(f(x) \neq h(x))$ is 1 if $f(x) \neq h(x)$, and 0 otherwise.

How well does $error_{S}(h)$ estimate $error_{\mathcal{D}}(h)$?
Bias and Variance

1. \textit{Bias}: If $S$ is training set, $error_S(h)$ is optimistically biased

\[
\text{bias} \equiv E[error_S(h)] - error_D(h)
\]

For unbiased estimate, $h$ and $S$ must be chosen independently

2. \textit{Variance}: Even with unbiased $S$, $error_S(h)$ may still vary from $error_D(h)$
Retrospective on Decision Trees

• Representation and search
• Does Bagging and Boosting change model representation space?
• Do they change search preference?
• Order of data presented does not count.