Lecture 4 - Review

- 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 + variance
- How we calculate bias and variance for a learner:
  - \( T_{\text{train}} \): Training sets drawn randomly from population
- Bias is the expected (mean) error over all training sets
- Variance is the variability of the error.
- Why would a decision tree be biased? Have a high variance?

Ensemble Techniques Reduce Error

- Decision trees are known to have a high variance, particularly when overfitted.
- BMA
  - Expected cost of Bayesian prediction is the noise.
  - Why?
- Bagging
  - Reduces variance but not bias
- Boosting
  - Reduces what?

Boosting – The Idea

- Take weak learners (marginal 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 Psuedo Code

\[ D_j(i) = 1 / I \] \ Initial training instances have same weight
For \( j = 1 \) to \( J \) \( (J \) is the number of rounds (trees)
Build \( H_j \) from \( D_j \)
\[ \alpha_j = 0.5 \log ( \frac{1 - \text{Error}(H_j)}{\text{Error}(H_j)} ) \]
For \( i = 1 \) to \( I \)
  - if instance \( i \) is misclassified then\( D_{j+1}(i) = D_j(i) e^{\alpha_j} \)
  - else\( D_{j+1}(i) = D_j(i) e^{-\alpha_j} \)
  - endif
  - endfor
Elaborate calculation of \( \alpha_j \) is so that \( \sum D_j(i) = 1 \)
Prediction(\( x \)) = \( \sum \alpha_j \cdot H_j(x) \), \( H_j(x) \) produces a 0 or 1
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

Final Hypothesis
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

Ensemble Technique Scorecard

<table>
<thead>
<tr>
<th></th>
<th>BMA</th>
<th>Bagging</th>
<th>Boosting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reducer</td>
<td>Both</td>
<td>Variance</td>
<td>Boost*</td>
</tr>
<tr>
<td>Variance</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sampling</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Voting Scheme</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Degree of Belief in Model</td>
<td>Equal</td>
<td>Depends on Model Error</td>
<td></td>
</tr>
<tr>
<td>Requirement of Learners</td>
<td>Bayesian</td>
<td>Unstable</td>
<td>Weak consistently better than random guessing</td>
</tr>
</tbody>
</table>

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.

Reading for Next Classes

• Additional reading
  – Various papers at http://www.boosting.org
• Lecture 6 – 02/11/02 and onwards
  – Mitchell chapter 4 (Neural Networks)
Lecture 4 - Review

- 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 + variance
- How we calculate bias and variance for a learner?
  - \( T_{\text{train}} \): Training sets drawn randomly from population
- Bias is the expected (mean) error over all training sets
- Variance is the variability of the error.
- Why would a decision tree be biased? Have a high variance?

Ensemble Techniques Reduce Error

- Decision trees are known to have a high variance, particularly when overfitted.
- BMA
  - Expected cost of Bayesian prediction is the noise.
  - Why?
- Bagging
  - Reduces variance but not bias
- Boosting
  - Reduces what?

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 Psuedo Code

\[
D_j(i) = 1 / I \quad \text{Initially training instances have same weight}
\]

For \( j = 1 \) to \( J \) \( \text{is the number of rounds (trees)} \)

Build \( H_j \) from \( D_j \)

\[
\alpha_j = 0.5 \log \left( \frac{1 - \text{Error}(H_j)}{\text{Error}(H_j)} \right)
\]

For \( i = 1 \) to \( I \)

- if instance \( i \) is misclassified then
  - \( D_j,i(i) = D_j(i) e^{\alpha_j} \)
- else
  - \( D_j,i(i) = D_j(i) e^{-\alpha_j} \)
- endif
- endfor
- endfor

Elaborate calculation of \( \alpha_j \) is so that \( \sum D_j(i) = 1 \)

Prediction(\( x \)) = \( \sum \alpha_j H_j(x) \), \( H_j(x) \) produces a 0 or 1
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

Final Hypothesis

Demo at http://www.cs.huji.ac.il/~yossi/adaboost/index.html
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
  \[ \text{Margin}(x) = y \sum \alpha_i h_i(x) \]

<table>
<thead>
<tr>
<th>Ensemble Technique Scorecard</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMA</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>Variance Reducer</td>
</tr>
<tr>
<td>Variance Only</td>
</tr>
<tr>
<td>Degree of Belief in Model</td>
</tr>
<tr>
<td>Requirement of Learners</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

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.

Reading for Next Classes

- Additional reading
  - Various papers at http://www.boosting.org
  - Lecture 6 – 02/11/02 and onwards
    - Mitchell chapter 4 (Neural Networks)
Lecture 4 - Review

- 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 + variance
- How we calculate bias and variance for a learner?
  - $T_{i, \omega}$: Training sets drawn randomly from population
- Bias is the expected (mean) error over all training sets
- Variance is the variability of the error.
- Why would a decision tree be biased? Have a high variance?

Ensemble Techniques Reduce Error

- Decision trees are known to have a high variance, particularly when overfitted.
- BMA
  - Expected cost of Bayesian prediction is the noise.
  - Why?
- Bagging
  - Reduces variance but not bias
- Boosting
  - Reduces what?

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

$D_1(i) = 1 / I$ // Initially training instances have same weight
For $j = 1$ to $J$ // $J$ is the number of rounds (trees)
  Build $H_j$ from $D_j$
  $\alpha_j = 0.5 \log (1 - \text{Error}(H_j) / \text{Error}(H_j))$
  For $i = 1$ to $I$
    if instance $i$ is misclassified then
      $D_{j+1}(i) = D_j(i) e^{\alpha_j}$
    else
      $D_{j+1}(i) = D_j(i) e^{-\alpha_j}$
    endif
  endfor
endfor

Elaborate calculation of $\alpha_j$ is so that $\Sigma D_j(i) = 1$
Prediction($x$) = $\Sigma \alpha_j H_j(x)$, $H_j(x)$ produces a 0 or 1
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

Final Hypothesis

Demo at http://www.cs.huji.ac.il/~yossi/adaboost/index.html
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
  \[ \text{Margin}(x) = y \sum \alpha_j h_j(x) / \sum \alpha_j \]

Ensemble Technique Scorecard

<table>
<thead>
<tr>
<th></th>
<th>BMA</th>
<th>Bagging</th>
<th>Boosting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reducer</td>
<td>Both</td>
<td>_variance</td>
<td>Boost*</td>
</tr>
<tr>
<td>Variance</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Or bias</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Voting Scheme</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Degree of</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Belief in Model</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Requirement of</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learners</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bayesian</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unstable</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weak</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>consistently</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>better than</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>random guessing</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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.

Reading for Next Classes

- Additional reading
  - Various papers at http://www.boosting.org
- Lecture 6 – 02/11/02 and onwards
  - Mitchell chapter 4 (Neural Networks)
Lecture 4 - Review

- 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*
  - \(T_{train}\): Training sets drawn randomly from population
- Bias is the expected (mean) error over all training sets
- Variance is the variability of the error.
- Why would a decision tree be biased? Have a high variance?

Ensemble Techniques Reduce Error

- Decision trees are known to have a high variance, particularly when overfitted.
- BMA
  - Expected cost of Bayesian prediction is the noise.
  - Why?
- Bagging
  - Reduces variance but not bias
- Boosting
  - Reduces what?

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 Psuedo Code

\[
D_j(i) = \frac{1}{I} \quad \text{Initially training instances have same weight}
\]

For \(j = 1 \text{ to } J\)

1. \(\alpha_j = 0.5 \log (\frac{1 - \text{Error}(H_j)}{\text{Error}(H_j)})\)

2. For \(i = 1 \text{ to } I\)
   - if instance \(i\) is misclassified then
     \[D_{j+1}(i) = D_j(i) e^{\alpha_j}\]
   - else
     \[D_{j+1}(i) = D_j(i) e^{-\alpha_j}\]

endfor

endfor

Elaborate calculation of \(\alpha_j\) is so that \(\Sigma D_j(i) = 1\)

Prediction(x) = \(\Sigma \alpha_j H_j(x)\), \(H_j(x)\) produces a 0 or 1
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

Final Hypothesis
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.
  \[ \text{Margin}(x) = y \sum \alpha_i h_i(x) / \sum \alpha_i \]

Ensemble Technique Scorecard

<table>
<thead>
<tr>
<th></th>
<th>BMA</th>
<th>Bagging</th>
<th>Boosting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reducer Variance Or Bias</td>
<td>Both</td>
<td>Variance</td>
<td>Bias*</td>
</tr>
<tr>
<td>Voting Scheme</td>
<td>Degree of Belief in Model</td>
<td>Equal</td>
<td>Depends on Model Error</td>
</tr>
<tr>
<td>Requirement of Learners</td>
<td>Bayesian</td>
<td>Unstable</td>
<td>Weak consistently better than random guessing</td>
</tr>
</tbody>
</table>

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.

Reading for Next Classes

- Additional reading
  - Various papers at http://www.boosting.org
- Lecture 6 – 02/11/02 and onwards
  - Mitchell chapter 4 (Neural Networks)
Lecture 4 - Review

- 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 + variance
- How we calculate bias and variance for a learner
  - $T_i$: Training sets drawn randomly from population
- Bias is the expected (mean) error over all training sets
- Variance is the variability of the error.
- Why would a decision tree be biased? Have a high variance?

Ensemble Techniques Reduce Error

- Decision trees are known to have a high variance, particularly when overfitted.
- BMA
  - Expected cost of Bayesian prediction is the noise.
  - Why?
- Bagging
  - Reduces variance but not bias
- Boosting
  - Reduces what?

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 Psuedo Code

\[
D_j(i) = \frac{1}{I} \quad \text{Initially training instances have same weight}
\]

For $j = 1$ to $J$ is the number of rounds (trees)

Build $H_j$ from $D_j$

$\alpha_j = 0.5 \log \left( \frac{1 \cdot \text{Error}(H_j)}{\text{Error}(H_j)} \right)$

For $i = 1$ to $I$

if instance $i$ is misclassified then

$D_{j+1}(i) = D_j(i) \times e^{\alpha_j}$

else

$D_{j+1}(i) = D_j(i) \times e^{-\alpha_j}$

endif

endfor

Elaborate calculation of $\alpha_j$ is so that $\sum D_j(i) = 1$

Prediction($x$) = $\sum \alpha_j \cdot H_j(x)$, $H_j(x)$ produces a 0 or 1

\[
\]
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

Final Hypothesis

Demo at http://www.cs.huji.ac.il/~yossi/adaBoost/index.html
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
  \[ \text{Margin}(x) = y \sum \alpha_j h_j(x) / \sum \alpha_j \]

Ensemble Technique Scorecard

<table>
<thead>
<tr>
<th></th>
<th>BMA</th>
<th>Bagging</th>
<th>Boosting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reducer Variance</td>
<td>Both</td>
<td>Variance</td>
<td>Bias*</td>
</tr>
<tr>
<td>Voting Scheme</td>
<td>Degree of Belief in Model</td>
<td>Equal</td>
<td>Depends on Model Error</td>
</tr>
<tr>
<td>Requirement of Learners</td>
<td>Bayesian</td>
<td>Unstable</td>
<td>Weak consistently better than random guessing</td>
</tr>
</tbody>
</table>

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.

Reading for Next Classes

- Additional reading
  - Various papers at http://www.boosting.org
- Lecture 6 – 02/11/02 and onwards
  - Mitchell chapter 4 (Neural Networks)
Lecture 4 - Review

- Error of the hypothesis vs error of the learning algorithm?
- Know the training and test set error, good estimate of the learner’s performance?
- Learner Error = noise + bias + variance
- How do we calculate bias and variance for a learner?
  - $T_{i,x}$: Training sets drawn randomly from population
- Bias is the expected (mean) error over all training sets
- Variance is the variability of the error.
- Why would a decision tree be biased? Have a high variance?

Ensemble Techniques Reduce Error

- Decision trees are known to have a high variance, particularly when overfitted.
- BMA
  - Expected cost of Bayesian prediction is the noise.
  - Why?
- Bagging
  - Reduces variance but not bias
- Boosting
  - Reduces what?

Boosting – The Idea

- Take weak learners (marginally better than random guessing) make them stronger.
- Freund and Schapire, 95 – AdaBoost
- AdaBoost premise
  - Each training instance 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 Psuedo Code

$D_j(i) = 1/I$ // Initially training instances have same weight
For $j = 1$ to $J$ // $J$ is the number of rounds (trees)
Build $H_j$ from $D_j$

$\alpha_j = 0.5 \log \left( \frac{1}{\text{Error}(H_j)} / \text{Error}(H_j) \right)$

For $i = 1$ to $I$
if instance $i$ is misclassified then
  $D_{j+1}(i) = D_j(i) \cdot e^{\alpha_j}$
else
  $D_{j+1}(i) = D_j(i) \cdot e^{-\alpha_j}$
endif
endfor

Elaborate calculation of $\alpha_j$ is so that $\Sigma D_j(i) = 1$
Prediction$(x) = \Sigma \alpha_j \cdot H_j(x)$, $H_j(x)$ produces a 0 or 1
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
- Final Hypothesis

Demo at http://www.cs.huji.ac.il/~yoadf/adaboost/index.html
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$(x) = y \sum \alpha_j h_j(x) / \sum \alpha_j$

Ensemble Technique Scorecard

<table>
<thead>
<tr>
<th></th>
<th>BMA</th>
<th>Bagging</th>
<th>Boosting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reducer</td>
<td>Both</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Variance</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Or bias</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Voting Scheme</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Degree of</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Belief in Model</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Depends on</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model Error</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Requirement of</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learners</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bayesian</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unstable</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weak</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>consistently</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>better than</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>random guessing</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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.

Reading for Next Classes

• Additional reading
  – Various papers at http://www.boosting.org
• Lecture 6 – 02/11/02 and onwards
  – Mitchell chapter 4 (Neural Networks)
Lecture 4 - Review

- 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 + variance
- How we calculate bias and variance for a learner:
  - $T_{i,n}$: Training sets drawn randomly from population
- Bias is the expected (mean) error over all training sets
- Variance is the variability of the error.
- Why would a decision tree be biased? Have a high variance?

Ensemble Techniques Reduce Error

- Decision trees are known to have a high variance, particularly when overfitted.
- BMA
  - Expected cost of Bayesian prediction is the noise.
  - Why?
- Bagging
  - Reduces variance but not bias
- Boosting
  - Reduces what?

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 Psuedo Code

$$D_j(i) = 1 / I$$ Initial training instances have same weight
For $j$ = 1 to J is the number of rounds (trees)
Build $H_j$ from $D_j$
$$\alpha_i = 0.5 \log_2 \left( \frac{1 - \text{Error}(H_j)}{\text{Error}(H_j)} \right)$$
For $i = 1$ to $I$
if instance $i$ is misclassified then
$$D_{j+1}(i) = D_j(i) \cdot e^{\alpha_i}$$
else $$D_{j+1}(i) = D_j(i) \cdot e^{-\alpha_i}$$
endif
endfor
Elaborate calculation of $\alpha_i$ is so that $\Sigma D_j(i) = 1$
Prediction$(x) = \Sigma \alpha_i \cdot H_j(x)$, $H_j(x)$ produces a 0 or 1
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

Final Hypothesis

Demo at http://www.cs.huji.ac.il/~yosef/adaboost/index.html
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
  \[ \text{Margin}(x) = y \sum \alpha_j h_j(x) / \sum \alpha_j \]

Ensemble Technique Scorecard

<table>
<thead>
<tr>
<th></th>
<th>BMA</th>
<th>Bagging</th>
<th>Boosting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reducer</td>
<td>Both</td>
<td>Variance</td>
<td>Bias*</td>
</tr>
<tr>
<td>Variance</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Or bias</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Voting Scheme</td>
<td>Degree of Belief in Model</td>
<td>Equal</td>
<td>Depends on Model Error</td>
</tr>
<tr>
<td>Requirement of Learners</td>
<td>Bayesian</td>
<td>Unstable</td>
<td>Weak consistently better than random guessing</td>
</tr>
</tbody>
</table>

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.

Reading for Next Classes

- Additional reading
  - Various papers at http://www.boosting.org
- Lecture 6 – 02/11/02 and onwards
  - Mitchell chapter 4 (Neural Networks)