Lecture 3 - Review

Overfitting the Training Set

- What type of strategy do DT algorithms use?
- Why does over-fitting occur? Two part answer
- Our aim when building a model?

Knowing You’ve Overfitted?

Overcoming Overfitting

- Accuracy_{Training} >> Accuracy_{Test}
- Book definition (* fill in *)
- Model doesn’t make sense.
- Too little data
- Providing some apriori stopping criterion.
- Overfitting then pruning the tree back.
- Last approach tends to obtain better results.

Overfitting: Using Validation Sets

Why use this approach?

- Accuracy_{Training} >> Accuracy_{Test}
- Book definition (* fill in *)
- Model doesn’t make sense.
- Too little data
- Providing some apriori stopping criterion.
- Overfitting then pruning the tree back.
- Last approach tends to obtain better results.
Types of Pruning Using V. Set

• Reduced Error Pruning
  – Prune nodes depending on order of decreasing error
  – May not be sufficient data
• Rule Post Pruning (used in c4.5)
  – Grow, convert to rules, remove pre-conditions, sort rules on accuracy to get application order.
  – Why? Remove exclusiveness, can prune root node, readability.

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
  – \( \Pr(c \mid x, D, H) = \sum_{h \in H} \Pr(c \mid x, h) \cdot \Pr(h \mid D) \)
  – Weight each model’s prediction by how good the model is.
• Boosting (Bootstrap Aggregation), 1996.
  – Improves accuracy
    • Seminal paper says on 19 of 26 data sets improves accuracy by 4%.

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 \( M \) 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>

Example of Bagging

Problem

Single DT Solution

100 DT's

Bagging Solution

Errors

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

$$\text{error}_D(h) = \mathbb{P}_{x \sim D}[f(x) \neq h(x)]$$

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

$$\text{error}_S(h) = \frac{1}{n} \sum_{x \in S} \delta(f(x) \neq h(x))$$

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$_D(h)$?

Bias and Variance

1. Bias: If $S$ is training set, error$_S(h)$ is optimistically biased

$$\text{bias} = E[\text{error}_S(h)] - \text{error}_D(h)$$

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

2. Variance: Even with unbiased $S$, error$_S(h)$ may still vary from error$_D(h)$
Reading for Next Classes

• Additional reading
    http://www.boosting.org/papers/Breiman96.pdf
• Lecture 5 – 02/06/02
  – Mitchell pages (5.3.2 to 5.3.5, pages 135-137)
  – www.boosting.org (tutorial section)
    Conference on Artificial Intelligence, 1999.