Lecture 2 - Review

- Supervised learning takes +ve and -ve examples and tries to learn how to differentiate between the two. 
  
<table>
<thead>
<tr>
<th>Case A</th>
<th>Case B</th>
<th>Case C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Short</td>
<td>Long</td>
<td>Short</td>
</tr>
<tr>
<td>Read</td>
<td>New</td>
<td>Read</td>
</tr>
</tbody>
</table>

- Decision trees:
  - Recursively partitions the instances based on the independent attributes
  - Tries to choose a split that results in the least amount of disorder (entropy) amongst the labels.
  - Info. gain splitting criterion, biased towards favor smaller trees + those attributes with the highest information gain being towards the tree root.

- But, when do we stop partitioning?
- What if we do too many splits and partition too far?

Review of Choosing a Split

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: \( \text{error}_{\text{train}}(h) \)
- entire distribution \( D \) of data: \( \text{error}_{D}(h) \)

Hypothesis \( h \in H \) overfits training data if there is an alternative hypothesis \( h' \in H \) such that
\[
\text{error}_{\text{train}}(h) < \text{error}_{\text{train}}(h')
\]
and
\[
\text{error}_{D}(h) > \text{error}_{D}(h')
\]

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
  \[ \text{size(tree)} + \text{size(misclassifications(tree))} \]

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?

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.

| Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5 |

MDL Base Pruning
• Minimize Overall Message Length
  • MessLen(Model, Data) = MessLen(Model) + 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?
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.

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

- Go to 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/c-4.5.1 | more” to read documentation.
- See me during office hours if you have any problems.
Building a Model Using C4.5

• Options
  - c4.5 -f filestem [-a] [-p] [-v verb] [-t trials]
  -m msize [-i incr] [-g] [-m minobj] [-c cf]

• C4.5 -f golf -m 2
  outlook = overcast: Play (4.0)
  |   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)

<table>
<thead>
<tr>
<th>Size</th>
<th>Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>0( 0.0%)</td>
</tr>
</tbody>
</table>

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).

Further Reading/Next Week

• Further Reading
  – MDL-based Decision Tree Pruning
  – Finish reading chapter 3
  – First homework

• Ensemble of decision trees
  – More on Bagging
  – Boosting