Ensemble Methods
(Model Combination)

Duen Horng (Polo) Chau
Assistant Professor
Associate Director, MS Analytics
Georgia Tech

Partly based on materials by Professors Guy Lebanon, Jeffrey Heer, John Stasko, Christos Faloutsos, Parishit Ram (GT PhD alum; SkyTree), Alex Gray
### Numerous Possible Classifiers!

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Training time</th>
<th>Cross validation</th>
<th>Testing time</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>kNN classifier</td>
<td>None</td>
<td>Can be slow</td>
<td>Slow</td>
<td>??</td>
</tr>
<tr>
<td>Decision trees</td>
<td>Slow</td>
<td>Very slow</td>
<td>Very fast</td>
<td>??</td>
</tr>
<tr>
<td>Naive Bayes classifier</td>
<td>Fast</td>
<td>None</td>
<td>Fast</td>
<td>??</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Which Classifier/Model to Choose?

Possible strategies:

• Go from simplest model to more complex model until you obtain desired accuracy
• Discover a new model if the existing ones do not work for you
• Combine all (simple) models
Consider the data set $S = \{(x_i, y_i)\}_{i=1, \ldots, n}$

- Pick a sample $S^*$ with replacement of size $n$ ($S^*$ called a “bootstrap sample”)
- Train on $S^*$ to get a classifier $f^*$
- Repeat above steps $B$ times to get $f_1, f_2, \ldots, f_B$
- Final classifier $f(x) = \text{majority}\{f_b(x)\}_{j=1, \ldots, B}$

http://statistics.about.com/od/Applications/a/What-Is-Bootstrapping.htm
Bagging decision trees

Consider the data set $S$

- Pick a sample $S^*$ with replacement of size $n$
- Grow a decision tree $T_b$
- Repeat $B$ times to get $T_1, \ldots, T_B$
- The final classifier will be

$$f(x) = \text{majority}\{f_{T_b}(x)\}_{b=1,\ldots,B}$$
Random Forests

Almost identical to bagging decision trees, except we introduce some randomness:

- Randomly pick $m$ of the $d$ available attributes, at every split when growing the tree (i.e., $d-m$ attributes ignored)

Bagged random decision trees

= Random forests
Explicit CV not necessary

- Unbiased test error can be estimated using out-of-bag data points (OOB error estimate)
- You can still do CV explicitly, but that's not necessary, since research shows that OOB estimate is as accurate

https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#ooberr
Important points about random forests

Algorithm parameters

- Usual values for $m$: $\sqrt{d}, 1, 10$
- Usual value for $B$: keep adding trees until training error stabilizes
Important points about random forests

Algorithm parameters

- Size/#nodes of each tree
  - as in when building a decision tree
- May randomly pick an attribute, and may even randomly pick the split point!
  - Significantly simplifies implementation and increases training speed
- PERT - Perfect Random Tree Ensembles
- Extremely randomized trees
Advantages

• Efficient and simple training
• Allows you to work with simple classifiers
• Random-forests generally useful and accurate in practice (one of the best classifiers)
  • The other is gradient-boosted tree
    http://fastml.com/what-is-better-gradient-boosted-trees-or-random-forest/
• Embarrassingly parallelizable
Final words

Reading material

- Bagging: ESL Chapter 8.7
- Random forests: ESL Chapter 15