Bias/Variance Tradeoff

Model Loss (Error)

- Squared loss of model on test case i:
  \[
  \left( \text{Learn}(x_i, D) - \text{Truth}(x_i) \right)^2
  \]
- Expected prediction error:
  \[
  \left\langle \left( \text{Learn}(x, D) - \text{Truth}(x) \right)^2 \right\rangle_D
  \]

Bias/Variance Decomposition

\[
\left\langle \left( \text{Learn}(x, D) - \text{Truth}(x) \right)^2 \right\rangle_D = \text{Noise}^2 + \text{Bias}^2 + \text{Variance}
\]

\(\text{Noise}^2\) = lower bound on performance
\(\text{Bias}^2\) = (expected error due to model mismatch)^2
\(\text{Variance}\) = variation due to train sample and randomization

Bias^2

- Low bias
  - linear regression applied to linear data
  - 2nd degree polynomial applied to quadratic data
  - ANN with many hidden units trained to completion
- High bias
  - constant function
  - linear regression applied to non-linear data
  - ANN with few hidden units applied to non-linear data
Variance

- Low variance
  - constant function
  - model independent of training data
  - model depends on stable measures of data
    - mean
    - median
- High variance
  - high degree polynomial
  - ANN with many hidden units trained to completion

Sources of Variance in Supervised Learning

- noise in targets or input attributes
- bias (model mismatch)
- training sample
- randomness in learning algorithm
  - neural net weight initialization
- randomized subsetting of train set:
  - cross validation, train and early stopping set

Bias/Variance Tradeoff

- \((\text{bias}^2 + \text{variance})\) is what counts for prediction

Often:
- low bias => high variance
- low variance => high bias

Tradeoff:
- \(\text{bias}^2\) vs. variance

Bias/Variance Tradeoff

- Hastie, Tibshirani, Friedman “Elements of Statistical Learning” 2001

Reduce Variance Without Increasing Bias

- Averaging reduces variance:
  \[ \text{Var}(\overline{X}) = \frac{\text{Var}(X)}{N} \]

- Average models to reduce model variance
- One problem:
  - only one train set
  - where do multiple models come from?

Bagging: Bootstrap Aggregation

- Leo Breiman (1994)
- Bootstrap Sample:
  - draw sample of size |D| with replacement from D

  \[
  \text{Train } L_i(\text{BootstrapSample}_i(D))
  \]

  Regression: \( L_{\text{bagging}} = \overline{L_i} \)
  
  Classification: \( L_{\text{bagging}} = \text{Plurality}(L_i) \)

Bagging

- Best case:
  \[ \text{Var}(\text{Bagging}(L(x, D))) = \frac{\text{Variance}(L(x, D))}{N} \]

- In practice:
  - models are correlated, so reduction is smaller than 1/N
  - variance of models trained on fewer training cases usually somewhat larger
  - stable learning methods have low variance to begin with, so bagging may not help much
Bagging Results

Table 1 Missclassification Rates (Percent)

<table>
<thead>
<tr>
<th>Data Set</th>
<th>$\hat{e}_S$</th>
<th>$\hat{e}_P$</th>
<th>Decrease</th>
</tr>
</thead>
<tbody>
<tr>
<td>waveform</td>
<td>29.0</td>
<td>19.4</td>
<td>33%</td>
</tr>
<tr>
<td>heart</td>
<td>10.0</td>
<td>5.3</td>
<td>47%</td>
</tr>
<tr>
<td>breast cancer</td>
<td>6.0</td>
<td>4.2</td>
<td>30%</td>
</tr>
<tr>
<td>ionosphere</td>
<td>11.2</td>
<td>8.6</td>
<td>23%</td>
</tr>
<tr>
<td>diabetes</td>
<td>23.4</td>
<td>18.8</td>
<td>20%</td>
</tr>
<tr>
<td>glass</td>
<td>32.0</td>
<td>24.9</td>
<td>22%</td>
</tr>
<tr>
<td>soybean</td>
<td>14.5</td>
<td>10.6</td>
<td>27%</td>
</tr>
</tbody>
</table>

Breiman "Bagging Predictors" Berkeley Statistics Department TR#421, 1994

How Many Bootstrap Samples?

Table 5.1

<table>
<thead>
<tr>
<th>No. Bootstrap Replicates</th>
<th>Missclassification Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>21.8</td>
</tr>
<tr>
<td>25</td>
<td>19.5</td>
</tr>
<tr>
<td>50</td>
<td>19.4</td>
</tr>
<tr>
<td>100</td>
<td>19.4</td>
</tr>
</tbody>
</table>

Breiman "Bagging Predictors" Berkeley Statistics Department TR#421, 1994

More bagging results

More bagging results
Bagging with cross validation

- Train neural networks using 4-fold CV
  - Train on 3 folds earlystop on the fourth
  - At the end you have 4 neural nets
- How to make predictions on new examples?

Bagging with cross validation

- Train neural networks using 4-fold CV
  - Train on 3 folds earlystop on the fourth
  - At the end you have 4 neural nets
- How to make predictions on new examples?
  - Train a neural network until the mean earlystopping point
  - Average the predictions from the four neural networks

Can Bagging Hurt?

- Each base classifier is trained on less data
  - Only about 63.2% of the data points are in any bootstrap sample
- However the final model has seen all the data
  - On average a point will be in >50% of the bootstrap samples
Reduce Bias and Decrease Variance?

- Bagging reduces variance by averaging
- Bagging has little effect on bias
- Can we average and reduce bias?
- Yes:

Boosting

Boosting

- Weight all training samples equally
- Train model on train set
- Compute error of model on train set
- Increase weights on train cases model gets wrong!
- Train new model on re-weighted train set
- Re-compute errors on weighted train set
- Increase weights again on cases model gets wrong
- Repeat until tired (100+ iterations)
- Final model: weighted prediction of each model

Boosting

- Freund & Schapire:
  - theory for “weak learners” in late 80’s
- Weak Learner: performance on any train set is slightly better than chance prediction
- intended to answer a theoretical question, not as a practical way to improve learning
- tested in mid 90’s using not-so-weak learners
- works anyway!
Boosting: Initialization

Algorithm AdaBoost.M1

Input: sequence of $m$ examples $(x_1, y_1), \ldots, (x_m, y_m))$
with labels $y_i \in \{1, \ldots, k\}$
weak learning algorithm $WeakLearn$
integer $T$ specifying number of iterations

Initialize $D_t(i) = 1/m$ for all $i$.

Boosting: Iteration

Do for $t = 1, 2, \ldots, T$
1. Call WeakLearn, providing it with the distribution $D_t$.
2. Get back a hypothesis $h_t : X \rightarrow Y$.
3. Calculate the error of $h_t$: $\epsilon_t = \frac{1}{m} \sum_{i, h_t(x_i) \neq y_i} D_t(i)$.
   If $\epsilon_t > 1/2$, then set $T = t - 1$ and abort loop.
4. Set $\beta_t = \epsilon_t / (1 - \epsilon_t)$.
5. Update distribution $D_{t+1}$:
   $D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \left\{ \begin{array}{ll} \beta_t & \text{if } h_t(x_i) = y_i \\ 1 & \text{otherwise} \end{array} \right.$
where $Z_t$ is a normalization constant (chosen so that $D_{t+1}$ will be a distribution).

Boosting: Prediction

Output the final hypothesis:

$$h_{fin}(x) = \arg \max_{y \in Y} \sum_{t : h_t(x) = y} \log \frac{1}{\beta_t}.$$
Reweighting vs Resampling

- Example weights might be harder to deal with
  - Some learning methods can’t use weights on examples
  - Many common packages don’t support weights on the train
- We can resample instead:
  - Draw a bootstrap sample from the data with the probability of drawing each example is proportional to it’s weight
- Reweighting usually works better but resampling is easier to implement

Boosting Performance

Boosting vs. Bagging

- Bagging doesn’t work so well with stable models. Boosting might still help.
- Boosting might hurt performance on noisy datasets. Bagging doesn’t have this problem
- In practice bagging almost always helps.

Boosting vs. Bagging

- On average, boosting helps more than bagging, but it is also more common for boosting to hurt performance.
- The weights grow exponentially. Code must be written carefully (store log of weights, …)
- Bagging is easier to parallelize.
**Bagged Decision Trees**
- Draw 100 bootstrap samples of data
- Train trees on each sample -> 100 trees
- Un-weighted average prediction of trees

**Random Forests (Bagged Trees++)**
- Draw 1000+ bootstrap samples of data
- **Draw sample of available attributes at each split**
- Train trees on each sample/attribute set -> 1000+ trees
- Un-weighted average prediction of trees

**Model Averaging**
- Almost always helps
- Often easy to do
- Models shouldn’t be too similar
- Models should all have pretty good performance (not too many lemons)
- When averaging, favor low bias, high variance
- Models can individually overfit
- Not just in ML

**Out of Bag Samples**
- With bagging, each model trained on about 63% of training sample
- That means each model does not use 37% of data
- Treat these as test points!
  - Backfitting in trees
  - Pseudo cross validation
  - Early stopping sets