Lecture 15: Ensemble classifiers - Boosting

- Idea of boosting
- AdaBoost algorithm (Freund and Schapire)
- Why does boosting work?
- Margin of a classifier as a measure of true error

Lecture based on material provided by Rob Schapire and Tommi Jaakkola

Recall from last time: Bagging

- Combines the predictions of several classifiers in order to reduce variance
- Repeatedly
  1. Sample with replacement data from the training set
  2. Train a new classifier on the sample data
- The predictions of the classifiers are combined by majority voting

Main idea of boosting

Component classifiers should concentrate more on difficult examples

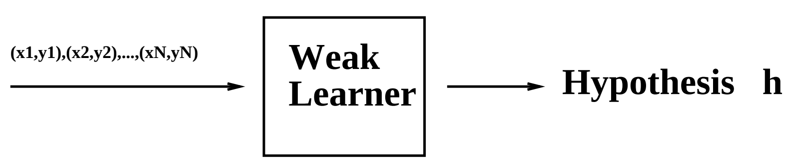
- Examine the training set
- Derive some rough rule of thumb
- *Re-weight* the examples of the training set, concentrating on "hard" cases for the previous rule
- Derive a second rule of thumb
- And so on... (repeat this T times)
- *Combine* the rules of thumb into a single, accurate rule

Questions:

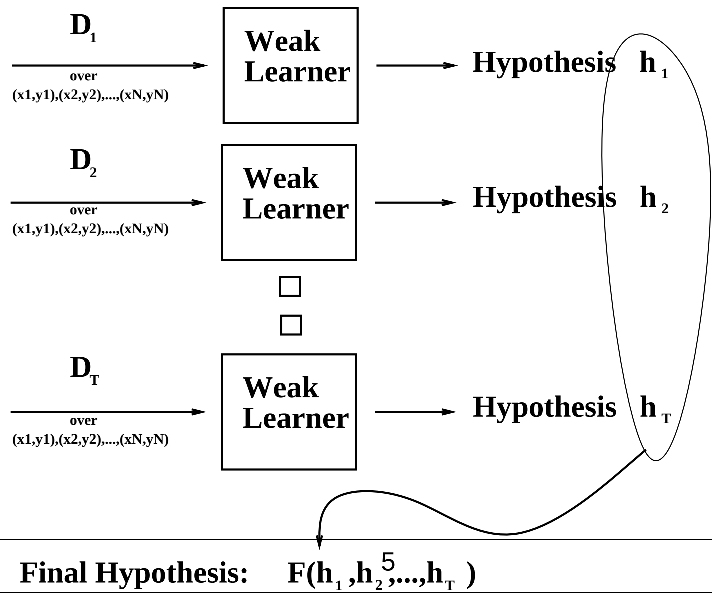
- How do we re-weight the examples?
- How do we combine the rules into a single classifier?

Weak learners

- Assume we have some "weak" binary classifiers (e.g., decision stumps: $x_i > t$)
- "Weak" means $error_{\mathcal{D}}(h) < 1/2 - \gamma$ (i.e., the true error is better than random).



Boosting classifier



AdaBoost (Freund & Schapire, 1995)

1. Input N training examples $\{(x_1, y_1), \ldots (x_N, y_N)\}$, where x_i are the attributes and y_i is the desired class label
2. Let $D_1(x_i) = \frac{1}{N}$ (we start with a uniform distribution)
3. Repeat T times:

   (a) Construct D_{t+1} from D_t as follows:

$$D_{t+1}(x_i) = \frac{1}{Z_t} D_t(x_i) \times \begin{cases} \beta_t, & \text{if } h_t(x_i) = y_i \\ 1, & \text{otherwise} \end{cases} \quad \text{where}$$

$$\beta_t = \frac{error_{D_t}(h_t)}{1 - error_{D_t}(h_t)}$$

   and Z_t is a normalization factor (set such that the probabilities $D_{t+1}(x_i)$ sum to 1).

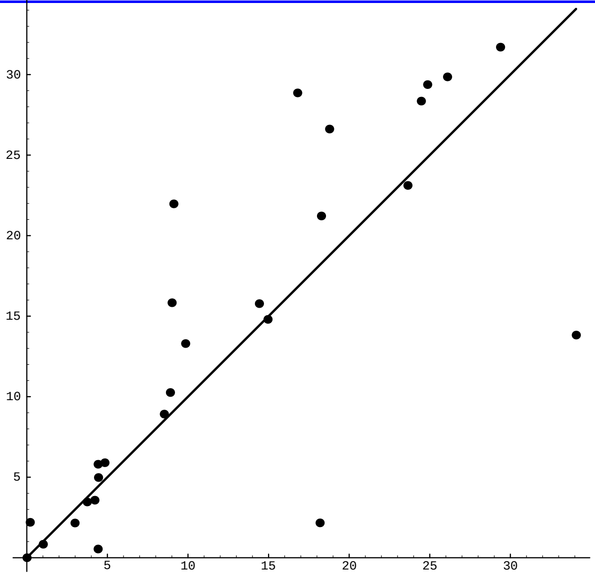
0.001in=0.401920.001in0.1in=0.401920.1in

(b) Train a new hypothesis h_{t+1} on distribution D_{t+1}

4. Construct the final hypothesis:

$$h_f(x) = \text{sign}\left(\sum_t \alpha_t h_t(x)\right), \text{ where } \alpha_t = \frac{\log(1/\beta_t)}{\sum_s \log(1/\beta_s)}$$

Empirical comparison: Boosted stumps vs. C4.5



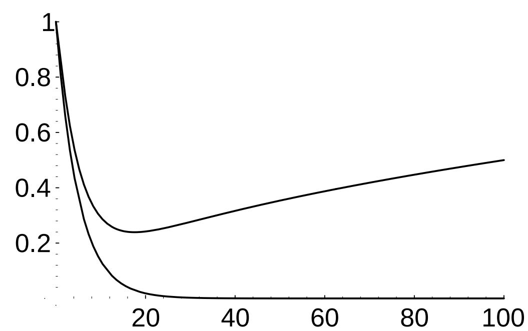
Why does boosting work?

- Weak learners have high bias
- By combining them, we get more expressive classifiers
- Hence, boosting is a *bias-reduction technique*
- What happens as we run boosting longer?

  Intuitively, we get more and more complex hypotheses

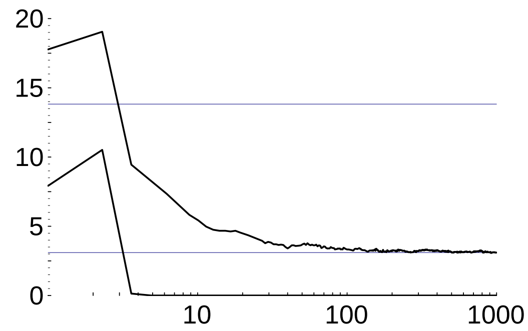
A naive (but reasonable) analysis of generalization error

- Expect the training error to continue to drop (until it reaches 0)
- Expect the test error to *increase* as we get more voters, and h_f becomes too complex.



Actual typical run of AdaBoost

Boosting C4.5 on the letter dataset:

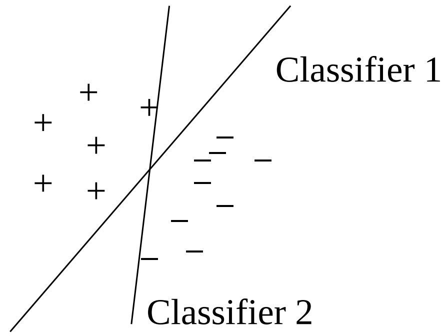


- Test error *does not increase* even after 1000 runs! (more than 2 million decision nodes!)
- Test error *continues to drop* even after training error reaches 0!

These are consistent results through many sets of experiments!

Classification margin

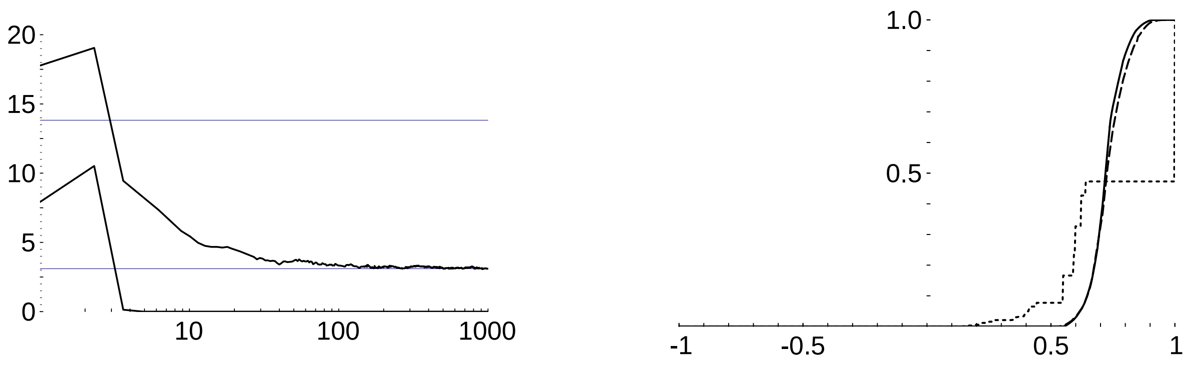
- The training error does not tell the whole story. We also need to think about the classification confidence
- Consider the following two classifiers, each of which have 0 error. Which one would you prefer?



Definition of margin

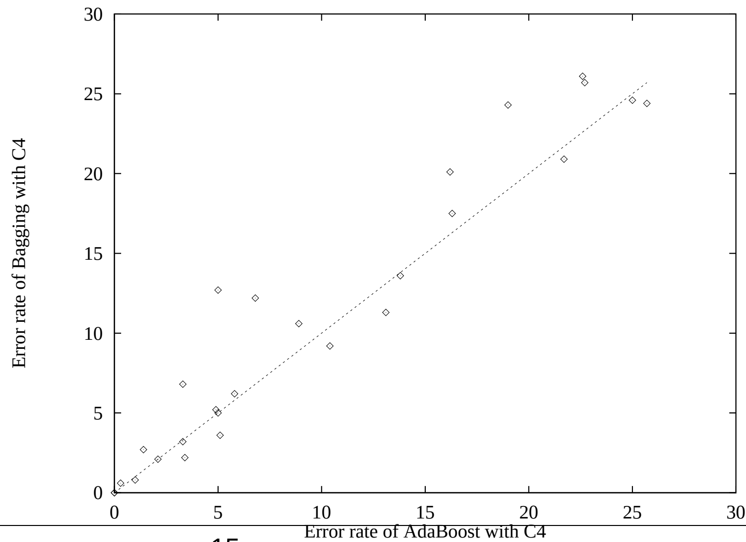
- Boosting constructs hypotheses of the form
  $h_f(x) = \text{sign}(f(x))$
- The classification of an example is correct if $\text{sign}(f(x)) = y$
- The **margin** is defined as: $\text{margin}_f(x, y) = y \cdot f(x)$
- The margin tells us how close the decision boundary is to the data points on each side.
- A higher margin on the training set should yield a lower generalization error
- Intuitively, increasing the margin is similar to lowering the variance

Effect of boosting on the margin



- Between rounds 5 and 10 there is no training error reduction
- But there is a **significant shift** in margin distribution!
- There is a proof that boosting increases the margin

Bagging vs. Boosting



Parallel of bagging and boosting

- Bagging is typically faster, but may get a less error reduction (not by much)
- Bagging works well with “reasonable” classifiers
- Boosting works with very simple classifiers

  E.g., Boostexter - text classification using decision stumps based on single words

Summary

- Errors in classification are either systematic (bias) or due to the particular data set (variance)
- Different algorithms make different trade-offs.
- Ensemble methods work by reducing either bias or variance (or both)
- Bagging is a variance-reduction technique
- Main idea is to sample the data repeatedly, train several classifiers and average their predictions.
- Boosting works by focusing on harder examples, and giving a weighted vote to the hypotheses.
- Boosting works by reducing bias and increasing classification margin.