Boosting
A Framework for Ensemble Learning

CMSC 422
MARINE CARPUAT
marine@cs.umd.edu
Recall: Machine Learning as Function Approximation

Problem setting
- Set of possible instances $X$
- Unknown target function $f: X \rightarrow Y$
- Set of function hypotheses $H = \{ h \mid h: X \rightarrow Y \}$

Input
- Training examples $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})\}$ of unknown target function $f$

Output
- Hypothesis $h \in H$ that best approximates target function $f$
How can we turn a weak learner into a strong learner?

• Weak learner
  – consistently makes better predictions than random guessing (error < 50%)

• Strong learner
  – probably approximately correct (PAC)

Definitions 1. An algorithm $A$ is an $(\epsilon, \delta)$-PAC learning algorithm if, for all distributions $D$: given samples from $D$, the probability that it returns a “bad function” is at most $\delta$; where a “bad” function is one with test error rate more than $\epsilon$ on $D$.

(See CIML 10.3 for more learning theory)
Boosting

• Boosting = process of turning a weak algorithm into a strong (PAC) learner

• AdaBoost
  – first practical boosting algorithm
  – stands for “adaptive boosting”
  – adapts to the training data its given
AdaBoost Intuition

- Train a sequence of weak learners and combine their predictions

- Weak learner $t+1$ focuses on examples that were most challenging for weak learner $t$
  - Give more weight to examples with incorrect predictions at time $t$
  - Give less weights to examples that are correctly classified at time $t$
The AdaBoost Algorithm

Algorithm 31 AdaBoost(\(\mathcal{W}, \mathcal{D}, K\))

1. \(d^{(0)} \leftarrow \langle \frac{1}{N}, \frac{1}{N}, \ldots, \frac{1}{N} \rangle\)  // Initialize uniform importance to each example
2. for \(k = 1 \ldots K\) do
3. \(f^{(k)} \leftarrow \mathcal{W}(\mathcal{D}, d^{(k-1)})\)  // Train \(k\)th classifier on weighted data
4. \(\hat{y}_n \leftarrow f^{(k)}(x_n), \forall n\)  // Make predictions on training data
5. \(\hat{e}^{(k)} \leftarrow \sum_n d_n^{(k-1)} [y_n \neq \hat{y}_n]\)  // Compute weighted training error
6. \(\alpha^{(k)} \leftarrow \frac{1}{2} \log \left( \frac{1-\hat{e}^{(k)}}{\hat{e}^{(k)}} \right)\)  // Compute “adaptive” parameter
7. \(d_n^{(k)} \leftarrow \frac{1}{Z} d_n^{(k-1)} \exp[-\alpha^{(k)} y_n \hat{y}_n], \forall n\)  // Re-weight examples and normalize
8. end for
9. return \(f(\hat{x}) = \text{sgn} \left[ \sum_k \alpha^{(k)} f^{(k)}(\hat{x}) \right]\)  // Return (weighted) voted classifier
Algorithm 31 AdaBoost(\(\mathcal{W}, \mathcal{D}, K\))

1. \(d^{(0)} \leftarrow \left\langle \frac{1}{N}, \frac{1}{N}, \ldots, \frac{1}{N} \right\rangle\)  // Initialize uniform weights

2. for \(k = 1 \ldots K\) do

3. \(f^{(k)} \leftarrow \mathcal{W}(\mathcal{D}, d^{(k-1)})\)  // Train \(k\)

4. \(\hat{y}_n \leftarrow f^{(k)}(x_n), \forall n\)  // Make prediction

5. \(\hat{e}^{(k)} \leftarrow \sum_n d_n^{(k-1)} [y_n \neq \hat{y}_n]\)  // Compute error

6. \(\alpha^{(k)} \leftarrow \frac{1}{2} \log \left( \frac{1 - \hat{e}^{(k)}}{\hat{e}^{(k)}} \right)\)  // Computed weight

7. \(d_n^{(k)} \leftarrow \frac{1}{Z} d_n^{(k-1)} \exp[-\alpha^{(k)} y_n \hat{y}_n], \forall n\)  // Re-weight

8. end for

9. return \(f(\hat{x}) = \text{sgn} \left[ \sum_k \alpha^{(k)} f^{(k)}(\hat{x}) \right]\)  // Return final function
Example

• Let’s boost the majority class weak learner (on board)
Example:
boosting decision stumps
Example:
boosting decision stumps

$D_1$

$h_1$

$\epsilon_1 = 0.30$

$\alpha_1 = 0.42$
Example: boosting decision stumps

$D_1$

$D_2$

$D_3$

$h_1$

$h_2$

$h_3$

$\varepsilon_1 = 0.30$

$\alpha_1 = 0.42$

$\varepsilon_2 = 0.21$

$\alpha_2 = 0.65$

$\varepsilon_3 = 0.14$

$\alpha_3 = 0.92$
例题：
加强决策树

- 最终的决策边界看起来像什么？

- AdaBoost 带有决策树提供了一个学习线性分类器的算法！
AdaBoost

• Very general framework
  – it can use any weak learning algorithm (e.g., decision stumps, Naïve Bayes...)

• Very fast (single pass through data each time)

• Simple to implement

• No parameters to tune

• But sensitive to noise
Many other ensemble learning methods

- General idea: combine predictions of multiple classifiers into a stronger learner
- Boosting is one example
- Another example: Random Forests

**Algorithm 32** \textsc{RandomForestTrain}(\(D,\ depth,\ K\))

1. \textbf{for} \(k = 1 \ldots K\) \textbf{do}
2. \hspace{1em} \(t^{(k)} \leftarrow\) complete binary tree of depth \textit{depth} with random feature splits
3. \hspace{1em} \(f^{(k)} \leftarrow\) the function computed by \(t^{(k)}\), with leaves filled in by \(D\)
4. \textbf{end for}
5. \textbf{return} \(f(\hat{x}) = \text{sgn} \left[ \sum_k f^{(k)}(\hat{x}) \right] \) \hspace{1em} // Return voted classifier
What you should know

- What is ensemble learning
- What is the difference between a weak and a strong learner
- How to boost a weak learner with adaboost
- What is the connection between boosted decision stumps and linear classifiers