Lecture 8: Boosting (and Compression Schemes)
Boosting the Error

If we have an **efficient** learning algorithm that for any distribution **realizable** by $\mathcal{H}_n$ returns a predictor that is guaranteed to be slightly better than chance, i.e. has error $\epsilon_0 = \frac{1}{2} - \gamma < \frac{1}{2}$ (for some $\gamma > 0$), can we construct a new **efficient** algorithm that for any distribution **realizable** by $\mathcal{H}_n$ achieves arbitrarily low error $\epsilon$?

- Posed (as a theoretical question) by Valiant and Kearns, Harvard 1988
- Solved by MIT student Robert Schapire, 1990
- AdaBoost Algorithm by Schapire and Yoav Fruend, AT&T 1995
AdaBoost

- **Input**: Training set $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$
- **Weak Learner $A$**, which will be applied to distributions $D$ over $S$
  - If thinking of $A(S')$ as accepting a sample $S'$:
    - each $(x, y) \in S'$ is set to $(x_i, y_i)$ w.p. $D_i$ (independently and with replacements)
  - Can often think of $A$ as operating on a weighted sample, with weights $D_i$
- **Output**: hypothesis $h$

```
Initialize $D^{(1)} = \left(\frac{1}{m}, \frac{1}{m}, \ldots, \frac{1}{m}\right)$

For t=1, ..., T:

$h_t = A(D^{(t)})$

$\epsilon_t = L_{D^{(t)}}(h_t) = \sum_i D_i^{(t)} \cdot [h_t(x_i) \neq y_i]]$

$\alpha_t = \frac{1}{2} \log \left(\frac{1}{\epsilon_t} - 1\right)$

$D_i^{(t+1)} = \frac{D_i^{(t)} \exp(-\alpha_t y_i h_t(x_i))}{\sum_j D_j^{(t)} \exp(-\alpha_t y_j h_t(x_j))}$

Output: $\overline{h}_T(x) = \text{sign}\left(\sum_{t=1}^T \alpha_t h_t(x)\right)$
```
AdaBoost: Weight Update

Initialize $D^{(1)} = \left(\frac{1}{m}, \frac{1}{m}, \ldots, \frac{1}{m}\right)$

For $t=1, \ldots, T$:

$ h_t = A(D^{(t)})$

$\epsilon_t = L_{D^{(t)}}(h_t) = \sum_i D^{(t)}_i \cdot [h_t(x_i) \neq y_i]$  

$\alpha_t = \frac{1}{2} \log \left(\frac{1}{\epsilon_t} - 1\right)$

$D^{(t+1)}_i = \frac{D^{(t)}_i \exp(-\alpha_t y_i h_t(x_i))}{\sum_j D^{(t)}_j \exp(-\alpha_t y_j h_t(x_j))}$

Output: $h_s(x) = \text{sign}(\sum_{t=1}^T \alpha_t h_t(x))$

• Increase weight on errors, decrease on correct points:

$$D^{(t+1)}_i \propto \begin{cases} 
D^{(t)}_i \cdot \sqrt{\frac{1-\epsilon_t}{\epsilon_t}} & \text{if } h_t(x_i) \neq y_i \\
D^{(t)}_i \cdot \sqrt{\frac{\epsilon_t}{1-\epsilon_t}} & \text{if } h_t(x_i) = y_i 
\end{cases}$$
AdaBoost: Weight Update

\[ D_i^{(t+1)} = \frac{D_i^{(t)} \exp(-\alpha_t y_i h_t(x_i))}{Z_t} \]

\[ = \frac{1}{Z_t} \begin{cases} 
    D_i^{(t)} \cdot \sqrt{\frac{1 - \epsilon_t}{\epsilon_t}} & \text{if } h_t(x_i) \neq y_i \\
    D_i^{(t)} \cdot \sqrt{\frac{\epsilon_t}{1 - \epsilon_t}} & \text{if } h_t(x_i) = y_i 
\end{cases} \]

\[ = \begin{cases} 
    \frac{D_i^{(t)}}{2\epsilon_t} & \text{if } h_t(x_i) \neq y_i \\
    \frac{D_i^{(t)}}{2(1 - \epsilon_t)} & \text{if } h_t(x_i) = y_i 
\end{cases} \]

\[ Z_t = \sum_{h_t(x_i) \neq y_i} D_i^{(t)} \cdot \sqrt{\frac{1 - \epsilon_t}{\epsilon_t}} + \sum_{h_t(x_i) = y_i} D_i^{(t)} \cdot \sqrt{\frac{\epsilon_t}{1 - \epsilon_t}} \]

\[ = \epsilon_t \sqrt{\frac{1 - \epsilon_t}{\epsilon_t}} + (1 - \epsilon_t) \cdot \sqrt{\frac{\epsilon_t}{1 - \epsilon_t}} = 2\sqrt{\epsilon_t(1 - \epsilon_t)} \]

\[ L_{D^{(t+1)}(h_t)} = \sum_{h_t(x_i) \neq y_i} D_i^{(t+1)} = \sum_{h_t(x_i) \neq y_i} D_i^{(t)} \cdot \frac{1}{2\epsilon_t} = \epsilon_t \cdot \frac{1}{2\epsilon_t} = \frac{1}{2} \]
AdaBoost: Weight Update

Initialize $D^{(1)} = \left( \frac{1}{m}, \frac{1}{m}, \ldots, \frac{1}{m} \right)$

For $t=1, \ldots, T$:

- $h_t = A(D^{(t)})$
- $\epsilon_t = L_{D^{(t)}}(h_t) = \sum_i D^{(t)}_i \cdot \left[ [h_t(x_i) \neq y_i] \right]$
- $\alpha_t = \frac{1}{2} \log \left( \frac{1}{\epsilon_t} - 1 \right)$

$$D^{(t+1)}_i = \frac{D^{(t)}_i \exp(-\alpha_t y_i h_t(x_i))}{\sum_j D^{(t)}_j \exp(-\alpha_t y_j h_t(x_j))}$$

Output: $h_s(x) = \text{sign}(\sum_{t=1}^T \alpha_t h_t(x))$

- Increase weight on errors, decrease on correct points:

\[
D^{(t+1)}_i \propto \begin{cases} 
D^{(t)}_i \cdot \sqrt{\frac{1-\epsilon_t}{\epsilon_t}} & \text{if } h_t(x_i) \neq y_i \\
D^{(t)}_i \cdot \sqrt{\frac{\epsilon_t}{1-\epsilon_t}} & \text{if } h_t(x_i) = y_i 
\end{cases}
\]

$L_{D^{(t+1)}}(h_t) = 0.5$
AdaBoost as Learning a Linear Classifier

• Recall: $\bar{h}_T(x) = \text{sign}(\sum_{t=1}^{T} \alpha_t h_t(x))$

• Let $B = \{ \text{all hypothesis outputed by A} \}$ ("Base Class", e.g. decision stumps)

\[
\begin{align*}
\phi(x)[h] &= h(x) \\
\mathcal{L}(B) &= \{ \text{sign}(\langle w, \phi(x) \rangle) \mid w \in \mathbb{R}^B \} \\
\end{align*}
\]

Class of halfspaces $\mathcal{L}(B)$

\[
L_{S}^{\text{exp}}(w) = \frac{1}{m} \sum \ell_{\text{exp}}(h_w(x_i); y_i) \\
\ell_{\text{exp}}(h_w(x), y) = e^{-y\langle w, \phi(x) \rangle}
\]

• Each step of AdaBoost: Coordinate descent on $L_{S}^{\text{exp}}(w)$
  • Choose coordinate $h$ of $\phi(x)$ s.t. $\frac{\partial}{\partial w[h]} L_{S}^{\text{exp}}(w)$ is very negative
  • Update $w[h] = \arg \min L_{S}^{\text{exp}}(w)$ s.t. $\forall h' \neq h w[h']$ is unchanged
Coordinate Descent on $L_S^{\exp}(w)$

- $\frac{\partial}{\partial w[h]} L_S^{\exp}(w) = \frac{\partial}{\partial w[h]} \frac{1}{m} \sum e^{-y_i h_w(x_i)}$
  $= \frac{1}{m} \sum e^{-y_i h_w(x_i)} \left( -y_i \frac{\partial h_w(x_i)}{\partial w[h]} \right) = \frac{1}{m} \sum e^{-y_i h_w(x_i)} (-y_i h(x_i))$
  $= \frac{1}{m} \sum e^{-y_i \sum_{t=1}^{T-1} \alpha_t h_t(x_i)} (-y_i h(x_i)) \propto 2L_{D(t)}(h) - 1$
  $\prod_{t=1}^{T-1} e^{-y_i \alpha_t h_t(x_i)} \propto D_i^{(T)}$

- Minimize $L_{D(t)}(h) \Rightarrow$ Minimize $\frac{\partial}{\partial w[h]} L_S^{\exp}(w)$ (make it very negative)

- Updating $w[h]$: set $w^{(t)}[h_t] = w^{(t-1)}[h_t] + \alpha$
  $\alpha = \arg \min L_S^{\exp}(w^{(t)})$
  $0 = \frac{\partial}{\partial \alpha} L_S^{\exp}(w^{(t)}) = \frac{\partial}{\partial w[h_t]} L_S^{\exp}(w^{(t)}) \propto 2L_{D(t+1)}(h_t) - 1$
  $\Rightarrow$ choose $\alpha$ s.t. $L_{D(t+1)}(h_t) = \frac{1}{2}$
AdaBoost: Minimizing $L_S(h)$

- Theorem: If $\forall_t \epsilon_t \leq \frac{1}{2} - \gamma$, then $L^0_S(h_T) \leq L^\text{exp}_S(h_T) \leq e^{-2\gamma^2 T}$

Proof: $L^\text{exp}_S(h_T) = \frac{1}{m} \sum_i e^{-y_i \sigma_i T h_i} = \frac{1}{m} \sum_i \left( D_i^{T+1} m \prod_{t=1}^T z_t \right) = \prod_{t=1}^T z_t$

$= \prod_{t=1}^T \left( 2 \sqrt{\epsilon_t} \frac{1 - \epsilon_t}{2} \right) \leq \left( (1 - 2\gamma)(1 + 2\gamma) \right)^{T/2} = (1 - 4\gamma^2)^{T/2} \leq e^{-2\gamma^2 T}$

- If $A(\cdot)$ is a weak learner with $\delta_0$, $\epsilon_0 = \frac{1}{2} - \gamma$, and if $L_D(h) = 0$:
  - $L_S(h) = 0 \Rightarrow L_D(t)(h) = 0 \Rightarrow \text{w.p. } 1 - \delta, L_D(t)(h) \leq \frac{1}{2} - \gamma$
  - $\Rightarrow \text{w.p. } 1 - \delta T, L_S(h_s) \leq e^{-2\gamma^2 T}$

- To get any $\epsilon > 0$, run AdaBoost for $T = \frac{\log(1/\epsilon)}{2\gamma^2}$ rounds

- Setting $\epsilon = \frac{1}{2m}$, after $T = \frac{\log(2m)}{2\gamma^2}$ rounds: $L_S(h_s) = 0$

- What about $L_D(h)$?
Sparse Linear Classifiers

• Recall: \( h_s(x) = \text{sign}(\sum_{t=1}^{T} w_t h_t(x)) \)

• Let \( \mathcal{B} = \{ \text{all hypothesis outputed by } A \} \)
  • “Base Class”, e.g. decision stumps

\[
    h_T \in \left\{ \text{sign}(\langle w, \phi(x) \rangle) \mid w \in \mathbb{R}^B \right\}
\]

Class of halfspaces \( \mathcal{L}(\mathcal{B}) \)
Sparse Linear Classifiers

• Recall: \( h_s(x) = \text{sign}(\sum_{t=1}^{T} w_t h_t(x)) \)

• Let \( \mathcal{B} = \{ \text{all hypothesis output by } A \} \)
  • “Base Class”, e.g. decision stumps

\[
h_T \in \{ \text{sign}(\langle w, \phi(x) \rangle) \mid w \in \mathbb{R}^\mathcal{B}, \|w\|_0 \leq T \}\]

Class of **sparse** halfspaces \( \mathcal{L}(\mathcal{B}, T) \)

• You already saw: \( \text{VCdim}(\mathcal{L}(\mathcal{B}, T)) \leq O(T \log|\mathcal{B}|) \)

• Even if \( \mathcal{B} \) is infinite (e.g. in the case of decision stumps): \( \text{VCdim}(\mathcal{L}(\mathcal{B}, T)) \leq \tilde{O}(T \cdot \text{VCdim}(\mathcal{B})) \)

• Sample complexity: \( m = \tilde{O}\left(\frac{\log(m)}{\gamma^2} \cdot \frac{\text{VCdim}(\mathcal{B})}{\epsilon}\right) = \tilde{O}\left(\frac{\text{VCdim}(\mathcal{B})}{\gamma^2 \epsilon}\right) \)

• But if weak learner is improper and \( \text{VCdim}(\mathcal{B}) = \infty \)?
Compression Bounds

• Focus on realizable case, and learning rules s.t. $L_S(A(S)) = 0$

• Suppose $A(S)$ only dependent on first $r < m$ examples,
  $A((x_1, y_1), ..., (x_m, y_m)) = \tilde{A}((x_1, y_1), ..., (x_r, y_r))$:
  \[ L_S[r+1:m]\left(\tilde{A}(S[1:r])\right) = 0 \Rightarrow \forall S \sim \mathcal{D}^m L_D(A(S)) \leq \frac{\log\left(\frac{1}{\delta}\right)}{m - r} \]

• In fact, same holds for any predetermined $i_1, ..., i_r$, if $A(S)$ only depends on $(x_{i_1} y_{i_1}), ..., (x_{i_r} y_{i_r})$

• Now consider $A(S) = \tilde{A}(S_{I(S)})$ with $I: (\mathcal{X} \times \mathcal{Y})^m \rightarrow \{1..m\}^r$. That is, can represent $A(S)$ using $r$ training points, but need to choose which ones.

• Taking a union bound over $m^r$ choices of indices:
  \[ L_D(A(S)) \leq \frac{r \log m + \log\left(\frac{1}{\delta}\right)}{m - r} \]
Compression Schemes

• $A(S)$ is “$r$-compressing” if $A(S) = \tilde{A}(S_{I(S)})$ for some $I: (X \times Y)^m \to \{1..m\}^r$

• Axis Aligned Rectangles
  • $I(S) = \{\text{leftmost positive, rightmost positive, top positive, bottom positive}\}$
  • $r = 4$

• Halfspaces in $\mathbb{R}^d$
  • A bit trickier, but can be done with $r = d + 1$ (for non-homogenous)

• $A(\cdot)$ is $r$-compressing and $L_S(A(S)) = 0 \Rightarrow$ for $m > 2r$, $\forall S \sim \mathcal{D}^m$

\[ L_D(A(S)) \leq 2 \frac{r \log m + \log(1/\delta)}{m} \]

• By VC lower bound: $FINDCONS_{\mathcal{H}}$ is $r$-compressing $\Rightarrow VCdim(\mathcal{H}) \leq O(r)$

• In fact: $VCdim(\mathcal{H}) \leq r$

• Conjecture: every $\mathcal{H}$ has a $VCdim(\mathcal{H})$-compressing $FINDCONS_{\mathcal{H}}$
Back to Boosting...

• $A(S)$ is an $(\varepsilon_0 = \frac{1}{2} - \gamma, \delta_0)$ weak learner that uses $m_0$ samples.

• Boost the confidence to get a $(\frac{1}{2} - \frac{\gamma}{2}, \delta')$ learner that uses

$$m_1(\delta') = O\left(m_0 \cdot \frac{\log^{1/\delta'}}{\log^{1/\delta_0}} + \frac{\log^{1/\delta'} - \log \log^{1/\delta_0}}{\gamma^2}\right)$$
samples

• Run AdaBoost on $m$ samples for $T = \frac{2 \log m}{\gamma^2}$ iterations, each time using $m' = m_1\left(\frac{\delta}{T}\right)$ samples for the weak learner to get $L_S(\bar{h}_T) = 0$

$$\bar{h}_T = \sum_{t=1}^{T} \alpha_t h_t$$

$h_t = A($subsample of size $m')$

• $(h_1, ..., h_T)$ has a compression scheme with $r = T \cdot m'$ points

• What about $\alpha_t$???
Partial Compression

• Instead of $r$ training points specifying $A(S)$ exactly, suppose they only specify a limited set of hypothesis in which $A(S)$ lies.
  - $I: (X \times Y)^m \rightarrow \{1..m\}^r$
  - $F: (X \times Y)^r \rightarrow$ hypothesis classes, each with $\text{VCdim}(F(S)) \leq D$
  - $A(S) \in F(I(S))$

• Theorem: If $A(S)$ has a compression scheme as above and $L_S(A(S)) = 0$, then for $m \geq 2r + D$, $\forall S \sim \mathcal{D}^m$

$$L_D(A(S)) \leq O\left(\frac{(D + r) \log m + \log^2 \delta}{m}\right)$$

Proof outline: take union bound over choice of indices $I(S)$, of the VC-based uniform convergence bounds, each time using just the points outside $I(S)$. 
Back to Boosting...

- \( A(S) \) is an \((\epsilon_0 = \frac{1}{2} - \gamma, \delta_0)\) weak learner that uses \( m_0 \) samples.

- Boost the confidence to get a \((\frac{1}{2} - \frac{\gamma}{2}, \delta')\) learner that uses

\[
m_1(\delta') = O\left(m_0 \cdot \frac{\log^{1/\delta'}}{\log^{1/\delta_0}} + \frac{\log^{1/\delta'} - \log \log^{1/\delta_0}}{\gamma^2}\right) \text{ samples}
\]

- Run AdaBoost on \( m \) samples for \( T = \frac{2 \log m}{\gamma^2} \) iterations, each time using \( m' = m_1\left(\frac{\delta}{T}\right) \) samples for the weak learner to get \( L_S(\overline{h_T}) = 0 \)

\[
\overline{h_T} = \sum_{t=1}^{T} \alpha_t h_t \in \mathcal{L}(\{h_1, ..., h_T\}) = F(I(S))
\]

- Conclusion:

\[
L_D(\overline{h_T}) \leq O\left(\frac{(T + Tm') \log m + \log \frac{1}{\delta}}{m}\right) = O\left(\frac{m_0 \cdot \log^2 m \cdot \log \frac{1}{\delta}}{m}\right)
\]

\[\Rightarrow m(\epsilon, \delta) = O\left(\frac{m_0 \log^{2\frac{1}{\epsilon}} \log^{1/\delta}}{\epsilon} \cdot \frac{1}{\gamma^2 \log\frac{1}{\delta_0}}\right)\]

For fixed \( \epsilon_0, \delta_0 \)
AdaBoost In Practice

- Complexity control is in terms of sparsity (#iterations) \( T \)
- Realizable case (MDL): use first \( T \) s.t. \( L_S(\overline{h}_T) = 0 \)
- More realistically (SRM): Use validation/cross-validation to select \( T \)

- Even after \( L_S(\overline{h}_T) = 0 \), AdaBoost keeps improving the \( \ell_1 \) margin
Interpretations of AdaBoost

• “Boosting” weak learning to get arbitrary small error
  • Theory is for realizable case
  • Shows efficient weak and strong learning equivalent

• Ensemble method for combining many simpler predictors
  • E.g. combining decision stumps or decision trees
  • Other ensemble methods: bagging, averaging, gating networks

• Method for learning using \textit{sparse} linear predictors with large (infinite?) dimensional feature space
  • Sparsity controls complexity
  • Number of iterations controls sparsity

• Coordinate-wise optimization of $L_{S}^{\exp}(w)$
  • We’ll get back to this when we talk about real-valued loss

• Learning (in high dimensions) with large $\ell_1$ margin
  • Learning guarantee in terms of $\ell_1$ margin
  • We’ll get back to this when we talk about $\ell_1$ margin
Just one more thing...
Back to Hardness of Agnostic Learning

\[ \mathcal{H} = \{ x \mapsto [\langle w, x \rangle > 0] | w \in \mathbb{R}^n \} \]

\[ \mathcal{H}_{k(n)} = \{ h_1 \land h_2 \land \cdots \land h_k | h_i \in \mathcal{H} \} \]

- Lemma: \( \exists h \in \mathcal{H}_k L_D(h) = 0 \Rightarrow \exists h \in \mathcal{H} L_D(h) < \frac{1}{2} - \frac{1}{2k^2} \)

\( \mathcal{H} \) is efficiently agnostically learnable

\downarrow

Efficient weak learner for \( \mathcal{H}_{k(n)} \) with \( \gamma = \frac{1}{2k^2} \)

\downarrow

\( \mathcal{H}_{k(n)} \) is efficiently learnable (in realizable case) for, e.g. \( k(n) = n \)

- Conclusion: subject to lattice-based crypto hardness, halfspaces are not efficiently agnostically learnable (even improperly)