Boosting
Boosting Ideas

- Main idea: use weak learner to create strong learner.
- Ensemble method: combine base classifiers returned by weak learner.
- Finding simple relatively accurate base classifiers often not hard.
- But, how should base classifiers be combined?
AdaBoost

(Freund and Schapire, 1997)

\[ H \subseteq \{-1, +1\}^X. \]

\[
\text{AdaBoost}(S = ((x_1, y_1), \ldots, (x_m, y_m)))
\]

1. \textbf{for} \( i \leftarrow 1 \) \textbf{to} \( m \) \textbf{do}
2. \hspace{1em} \( D_1(i) \leftarrow \frac{1}{m} \)
3. \textbf{for} \( t \leftarrow 1 \) \textbf{to} \( T \) \textbf{do}
4. \hspace{1em} \( h_t \leftarrow \) base classifier in \( H \) with small error \( \epsilon_t = \Pr_{D_t}[h_t(x_i) \neq y_i] \)
5. \hspace{1em} \( \alpha_t \leftarrow \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t} \)
6. \hspace{1em} \( Z_t \leftarrow 2[\epsilon_t(1-\epsilon_t)]^{\frac{1}{2}} \)
   \hspace{1em} \text{normalization factor}
7. \hspace{1em} \textbf{for} \( i \leftarrow 1 \) \textbf{to} \( m \) \textbf{do}
8. \hspace{2em} \( D_{t+1}(i) \leftarrow \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t} \)
9. \hspace{1em} \( f \leftarrow \sum_{t=1}^{T} \alpha_t h_t \)
10. \textbf{return} \( h = \text{sgn}(f) \)
Notes

- Distributions $D_t$ over training sample:
  - originally uniform.
  - at each round, the weight of a misclassified example is increased.
  - observation: $D_{t+1}(i) = \frac{e^{-y_i f_t(x_i)}}{m \prod_{s=1}^{t} Z_s}$, since

$$D_{t+1}(i) = \frac{D_t(i) e^{-\alpha_t y_i h_t(x_i)}}{Z_t} = \frac{D_{t-1}(i) e^{-\alpha_{t-1} y_i h_{t-1}(x_i)} e^{-\alpha_t y_i h_t(x_i)}}{Z_{t-1} Z_t} = \frac{1}{m} \frac{e^{-y_i \sum_{s=1}^{t} \alpha_s h_s(x_i)}}{\prod_{s=1}^{t} Z_s}.$$  

- Weight assigned to base classifier $h_t$: $\alpha_t$ directly depends on the accuracy of $h_t$ at round $t$.  

Illustration
$t = 3$

\[\ldots\]
\[ \alpha_1 + \alpha_2 + \alpha_3 = \]
**Bound on Empirical Error**

(Freund and Schapire, 1997)

- **Theorem:** The empirical error of the classifier output by AdaBoost verifies:

  \[
  \hat{R}(h) \leq \exp \left[ -2 \sum_{t=1}^{T} \left( \frac{1}{2} - \epsilon_t \right)^2 \right].
  \]

- If further for all \( t \in [1, T] \), \( \gamma \leq \left( \frac{1}{2} - \epsilon_t \right) \), then

  \[
  \hat{R}(h) \leq \exp(-2\gamma^2 T).
  \]

- \( \gamma \) does not need to be known in advance: adaptive boosting.
• **Proof:** Since, as we saw, $D_{t+1}(i) = \frac{e^{-y_i f_t(x_i)}}{m \prod_{s=1}^{t} Z_s}$,

$$
\hat{R}(h) = \frac{1}{m} \sum_{i=1}^{m} 1_{y_i f(x_i) \leq 0} \leq \frac{1}{m} \sum_{i=1}^{m} \exp(-y_i f(x_i))
$$

$$
\leq \frac{1}{m} \sum_{i=1}^{m} \left[ m \prod_{t=1}^{T} Z_t \right] D_{T+1}(i) = \prod_{t=1}^{T} Z_t.
$$

• Now, since $Z_t$ is a normalization factor,

$$
Z_t = \sum_{i=1}^{m} D_t(i) e^{-\alpha_t y_i h_t(x_i)}
$$

$$
= \sum_{i:y_i h_t(x_i) \geq 0} D_t(i) e^{-\alpha_t} + \sum_{i:y_i h_t(x_i) < 0} D_t(i) e^{\alpha_t}
$$

$$
= (1 - \epsilon_t) e^{-\alpha_t} + \epsilon_t e^{\alpha_t}
$$

$$
= (1 - \epsilon_t) \sqrt{\frac{\epsilon_t}{1-\epsilon_t}} + \epsilon_t \sqrt{\frac{1-\epsilon_t}{\epsilon_t}} = 2 \sqrt{\epsilon_t (1 - \epsilon_t)}.
$$
• Thus,

\[
\prod_{t=1}^{T} Z_t = \prod_{t=1}^{T} 2 \sqrt{\epsilon_t (1 - \epsilon_t)} = \prod_{t=1}^{T} \sqrt{1 - 4 \left( \frac{1}{2} - \epsilon_t \right)^2 }
\leq \prod_{t=1}^{T} \exp \left[ - 2 \left( \frac{1}{2} - \epsilon_t \right)^2 \right] = \exp \left[ - 2 \sum_{t=1}^{T} \left( \frac{1}{2} - \epsilon_t \right)^2 \right].
\]

• Notes:
  • \(\alpha_t\) minimizer of \(\alpha \mapsto (1 - \epsilon_t) e^{-\alpha} + \epsilon_t e^\alpha\).
  • since \((1 - \epsilon_t) e^{-\alpha_t} = \epsilon_t e^{\alpha_t}\), at each round, AdaBoost assigns the same probability mass to correctly classified and misclassified instances.
  • for base classifiers \(x \mapsto [-1, +1]\), \(\alpha_t\) can be similarly chosen to minimize \(Z_t\).
AdaBoost = Coordinate Descent

- **Objective Function**: convex and differentiable.

\[
F(\alpha) = \sum_{i=1}^{m} e^{-y_i f(x_i)} = \sum_{i=1}^{m} e^{-y_i \sum_{t=1}^{T} \alpha_t h_t(x_i)}.
\]
• **Direction**: unit vector \( e_t \) with

\[
e_t = \text{argmin}_t \left. \frac{dF(\alpha_{t-1} + \eta e_t)}{d\eta} \right|_{\eta=0}.
\]

• Since \( F(\alpha_{t-1} + \eta e_t) = \sum_{i=1}^m e^{-y_i \sum_{s=1}^{t-1} \alpha_s h_s(x_i) e^{-y_i \eta h_t(x_i)}} \),

\[
\left. \frac{dF(\alpha_{t-1} + \eta e_t)}{d\eta} \right|_{\eta=0} = -\sum_{i=1}^m y_i h_t(x_i) \exp \left[ -y_i \sum_{s=1}^{t-1} \alpha_s h_s(x_i) \right]
\]

\[
= -\sum_{i=1}^m y_i h_t(x_i) D_t(i) \left[ m \prod_{s=1}^{t-1} Z_s \right]
\]

\[
= -[(1 - \epsilon_t) - \epsilon_t] \left[ m \prod_{s=1}^{t-1} Z_s \right] = [2\epsilon_t - 1] \left[ m \prod_{s=1}^{t-1} Z_s \right].
\]

Thus, direction corresponding to base classifier with smallest error.
• **Step size:** obtained via

\[
\frac{dF(\alpha_{t-1} + \eta e_t)}{d\eta} = 0 \iff - \sum_{i=1}^{m} y_i h_t(x_i) \exp \left[ - y_i \sum_{s=1}^{t-1} \alpha_s h_s(x_i) \right] e^{-y_i h_t(x_i) \eta} = 0
\]

\[
\iff - \sum_{i=1}^{m} y_i h_t(x_i) D_t(i) \left[ m \prod_{s=1}^{t-1} Z_s \right] e^{-y_i h_t(x_i) \eta} = 0
\]

\[
\iff - \sum_{i=1}^{m} y_i h_t(x_i) D_t(i) e^{-y_i h_t(x_i) \eta} = 0
\]

\[
\iff - \left[ (1 - \epsilon_t) e^{-\eta} - \epsilon_t e^\eta \right] = 0
\]

\[
\iff \eta = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t}.
\]

Thus, step size matches base classifier weight of AdaBoost.
Alternative Loss Functions

- **boosting loss**: $x \mapsto e^{-x}$
- **logistic loss**: $x \mapsto \log_2(1 + e^{-x})$
- **hinge loss**: $x \mapsto \max(1 - x, 0)$
- **zero-one loss**: $x \mapsto 1_{x < 0}$

The square loss is given by $x \mapsto (1 - x)^2 1_{x \leq 1}$.
Standard Use in Practice

- **Base learners**: decision trees, quite often just decision stumps (trees of depth one).

- **Boosting stumps**:
  - data in $\mathbb{R}^N$, e.g., $N = 2$, $(\text{height}(x), \text{weight}(x))$.
  - associate a stump to each component.
  - pre-sort each component: $O(Nm \log m)$.
  - at each round, find best component and threshold.
  - total complexity: $O((m \log m)N + mNT)$.
  - stumps not weak learners: think XOR example!
Overfitting?

- We could expect that AdaBoost would overfit for large values of $T$, and that is in fact observed in some cases, but in various others it is not!
- Several empirical observations (not all): AdaBoost does not seem to overfit, furthermore:

C4.5 decision trees (Schapire et al., 1998). # rounds
L1 Margin Definitions

Definition: the margin of a point \( x \) with label \( y \) is (the \( \| \cdot \|_\infty \) algebraic distance of \( x = [h_1(x), \ldots, h_T(x)]^\top \) to the hyperplane \( \alpha \cdot x = 0 \)):

\[
\rho(x) = \frac{y f(x)}{\sum_{t=1}^{m} \alpha_t} = \frac{y \sum_{t=1}^{T} \alpha_t h_t(x)}{\| \alpha \|_1} = y \frac{\alpha \cdot x}{\| \alpha \|_1}.
\]

Definition: the margin of the classifier for a sample \( S = (x_1, \ldots, x_m) \) is the minimum margin of the points in that sample:

\[
\rho = \min_{i \in [1,m]} \frac{y_i \alpha \cdot x_i}{\| \alpha \|_1}.
\]
• **Note:**

  • **SVM margin:** \[ \rho = \min_{i \in [1,m]} y_i \frac{\mathbf{w} \cdot \Phi(x_i)}{\| \mathbf{w} \|_2} \].

  • **Boosting margin:** \[ \rho = \min_{i \in [1,m]} y_i \frac{\alpha \cdot H(x_i)}{\| \alpha \|_1} \],

  with \( H(x) = \begin{bmatrix} h_1(x) \\ \vdots \\ h_T(x) \end{bmatrix} \).

  • **Distances:** \( \| \cdot \|_q \) distance to hyperplane \( \mathbf{w} \cdot \mathbf{x} + b = 0 \):

    \[ \frac{|\mathbf{w} \cdot \mathbf{x} + b|}{\| \mathbf{w} \|_p} \],

    with \( \frac{1}{p} + \frac{1}{q} = 1 \).
**Convex Hull of Hypothesis Set**

- **Definition:** Let \( H \) be a set of functions mapping from \( X \) to \( \mathbb{R} \). The **convex hull** of \( H \) is defined by

\[
\text{conv}(H) = \left\{ \sum_{k=1}^{p} \mu_k h_k : p \geq 1, \mu_k \geq 0, \sum_{k=1}^{p} \mu_k \leq 1, h_k \in H \right\}.
\]

- ensemble methods are often based on such convex combinations of hypotheses.
Margin Bound - Ensemble Methods

(Koltchinskii and Panchenko, 2002)

**Theorem:** Let $H$ be a set of real-valued functions. Fix $\rho > 0$. For any $\delta > 0$, with probability at least $1 - \delta$, the following holds for all $h \in \text{conv}(H)$:

$$R(h) \leq \widehat{R}_\rho(h) + \frac{2}{\rho} \mathcal{R}_m(H) + \sqrt{\frac{\log \frac{1}{\delta}}{2m}},$$

where $\mathcal{R}_m(H)$ is a measure of the complexity of $H$. 
For AdaBoost, the bound applies to the functions

\[ x \mapsto \frac{f(x)}{\|\alpha\|_1} = \frac{\sum_{t=1}^{T} \alpha_t h_t(x)}{\|\alpha\|_1} \in \text{conv}(H). \]

Note that \( T \) does not appear in the bound.
But, Does AdaBoost Maximize the Margin?

No: AdaBoost may converge to a margin that is significantly below the maximum margin (Rudin et al., 2004) (e.g., 1/3 instead of 3/8)!

Lower bound: AdaBoost can achieve asymptotically a margin that is at least \( \frac{\rho_{\text{max}}}{2} \) if data separable and some conditions on the base learners (Rätsch and Warmuth, 2002).

Several boosting-type margin-maximization algorithms: but, performance in practice not clear or not reported.
Outliers

- AdaBoost assigns larger weights to harder examples.

- **Application:**
  - Detecting mislabeled examples.
  - Dealing with noisy data: regularization based on the average weight assigned to a point (soft margin idea for boosting) \cite{MeirRatsch2003}.
Advantages of AdaBoost

- **Simple**: straightforward implementation.

- **Efficient**: complexity $O(mNT)$ for stumps:
  - when $N$ and $T$ are not too large, the algorithm is quite fast.

- **Theoretical guarantees**: but still many questions.
  - AdaBoost not designed to maximize margin.
  - regularized versions of AdaBoost.
Weaker Aspects

- **Parameters:**
  - need to determine $T$, the number of rounds of boosting: stopping criterion.
  - need to determine base learners: risk of overfitting or low margins.

- **Noise:** severely damages the accuracy of Adaboost (Dietterich, 2000).
  - boosting algorithms based on convex potentials do not tolerate even low levels of random noise, even with L1 regularization or early stopping (Long and Servedio, 2010).
References


References

