The Boosting Approach to Machine Learning

Maria-Florina Balcan
10/26/2015

Boosting

- General method for improving the accuracy of any given learning algorithm.
- Works by creating a series of challenge datasets s.t. even modest performance on these can be used to produce an overall high-accuracy predictor.
  - Works amazingly well in practice --- Adaboost and its variations one of the top 10 algorithms.
  - Backed up by solid foundations.

Readings:


Plan for today:

- Motivation.
- A bit of history.
- Adaboost: alg, guarantees, discussion.
- Focus on supervised classification.

An Example: Spam Detection

- E.g., classify which emails are spam and which are important.
  - Not spam
  - spam

Key observation/motivation:

- Easy to find rules of thumb that are often correct,
  - E.g., "If buy now in the message, then predict spam."
  - E.g., "If say good-bye to debt in the message, then predict spam."
- Harder to find single rule that is very highly accurate.
An Example: Spam Detection

- Boosting: meta-procedure that takes in an algo for finding rules of thumb (weak learner). Produces a highly accurate rule, by calling the weak learner repeatedly on cleverly chosen datasets.

  - apply weak learner to a subset of emails; obtain rule of thumb
  - apply to 2nd subset of emails, obtain 2nd rule of thumb
  - apply to 3rd subset of emails, obtain 3rd rule of thumb
  - repeat T times; combine weak rules into a single highly accurate rule.

Boosting: Important Aspects

How to choose examples on each round?

- Typically, concentrate on “hardest” examples (those most often misclassified by previous rules of thumb)

How to combine rules of thumb into single prediction rule?

- take (weighted) majority vote of rules of thumb

Weak Learning vs Strong/PAC Learning

- [Kearns & Valiant ’88]: defined weak learning: being able to predict better than random guessing (error < \(\frac{1}{2} - \frac{\epsilon}{2}\), consistently.

- Posed an open pb: "Does there exist a boosting algo that turns a weak learner into a strong PAC learner" (that can produce arbitrarily accurate hypotheses)?

- Informally, given "weak" learning algo that can consistently find classifiers of error < \(\frac{1}{2} - \frac{\epsilon}{2}\), a boosting algo would provably construct a single classifier with error ≤ \(\epsilon\).
Weak Learning vs Strong/PAC Learning

**Strong (PAC) Learning**
- ∃ algo A
- ∀ ε ∈ H
- ∀ δ > 0
- A produces h s.t.:
  \[ \Pr[\text{err}(h) \geq \varepsilon] \leq \delta \]

**Weak Learning**
- ∃ algo A
- ∃ δ > 0
- ∀ ε > 0
- ∀ δ > 0
- A produces h s.t.:
  \[ \Pr[\text{err}(h) \geq \varepsilon] \leq \delta \]

[Kearns & Valiant '88]: defined weak learning & posed an open pb of finding a boosting algo.

Surprisingly....
**Weak Learning = Strong (PAC) Learning**

**Original Construction [Schapire '89]:**
- poly-time boosting algo, exploits that we can learn a little on every distribution.
  - A modest booster obtained via calling the weak learning algorithm on 3 distributions.
    \[ \text{Error} = \beta < \frac{1}{2} \gamma - \text{error}_{\beta^3 - 2\beta^4} \]
  - Then amplifies the modest boost of accuracy by running this somehow recursively.
  - Cool conceptually and technically, not very practical.

An explosion of subsequent work

**Adaboost (Adaptive Boosting)**

“A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting”
(Freund-Schapire, JCSS'97)

Godel Prize winner 2003
Informal Description Adaboost

- Boosting: turns a weak algo into a strong (PAC) learner.

Input: \( S = \{(x_1, y_1), \ldots, (x_m, y_m)\} \), \( n_i \in [1, m] \)

Weak learning algo \( A \) (e.g., Naïve Bayes, decision stump)

- For \( t = 1, 2, \ldots, T \)
  - Construct \( D_t \) on \( \{x_1, \ldots, x_m\} \)
  - Run \( A \) on \( D_t \) producing \( h_t: X \to \{-1, 1\} \) (weak classifier)
  - \( \epsilon_t = \frac{1}{2} \ln \frac{1}{\epsilon_t} \) error of \( h_t \) over \( D_t \)
- Output \( H_{\text{final}}(x) = \text{sign} \left( \sum_{t=1}^{T} \alpha_t h_t(x) \right) \)

Roughly speaking, \( D_{t+1} \) increases weight on \( x_i \) if \( h_t \) incorrect on \( x_i \); decreases it on \( x_i \) if \( h_t \) is correct.

Adaboost: A toy example

Weak classifiers: vertical or horizontal half-planes (a.k.a. decision stumps)

Adaboost (Adaptive Boosting)

- Weak learning algorithm \( A \).
- For \( t = 1, 2, \ldots, T \)
  - Construct \( D_t \) on \( \{x_1, \ldots, x_m\} \)
  - Run \( A \) on \( D_t \) producing \( h_t \)
- Output \( H_{\text{final}}(x) = \text{sign} \left( \sum_{t=1}^{T} \alpha_t h_t(x) \right) \)

Adaboost: A toy example
Adaboost: A toy example

Adaboost (Adaptive Boosting)

- Weak learning algorithm A.
- For \( t = 1, 2, \ldots, T \):
  - Construct \( D_t \) on \( (x_1, \ldots, x_n) \)
  - Run A on \( D_t \) producing \( h_t \)

Constructing \( D_t \):

- \( D_1 \) uniform on \( \{x_1, \ldots, x_n\} \) [i.e., \( D_1(i) = \frac{1}{n} \)]
- Given \( D_t \) and \( h_t \) set
  - \( D_{t+1}(i) = \frac{D_t(i) e^{-y_i h_t(x_i)}}{Z_t} \) if \( y_i h_t(x_i) < 0 \)
  - \( D_{t+1}(i) = \frac{D_t(i) e^{y_i h_t(x_i)}}{Z_t} \) if \( y_i h_t(x_i) > 0 \)

Final hyp: \( \text{H}_{\text{final}}(x) = \text{sign} \left( \sum D_t(x) h_t(x) \right) \)

Nice Features of Adaboost

- Very general: a meta-procedure, it can use any weak learning algorithm (e.g., Naive Bayes, decision stumps)
- Very fast (single pass through data each round) & simple to code, no parameters to tune.
- Shift in mindset: goal is now just to find classifiers a bit better than random guessing.
- Grounded in rich theory.
- Relevant for big data age: quickly focuses on “core difficulties”, well-suited to distributed settings, where data must be communicated efficiently (Balcan-Blum-Fine-Mansour COLT 2012).

Analyzing Training Error

Theorem:

\[
\epsilon_t = 1/2 - y_t \text{ (error of } h_t \text{ over } D_t) \\
\text{err}_D(H_{\text{final}}) \leq \exp \left[ -2 \varepsilon T \right]
\]

So, if \( \forall t, \epsilon_t \geq \gamma > 0 \), then \( \text{err}_D(H_{\text{final}}) \leq \exp \left[ -2 \gamma^2 T \right] \)

The training error drops exponentially in T!!

To get \( \text{err}_D(H_{\text{final}}) \leq \epsilon \), need only \( T = O \left( \frac{1}{\epsilon^2 \gamma^2} \right) \) rounds

Adaboost is adaptive

- Does not need to know \( \gamma \) or \( T \) a priori
- Can exploit \( \gamma_t \gg \gamma \)
Understanding the Updates & Normalization

Claim: $D_{t+1}$ puts half of the weight on $x_i$ where $h_t$ was incorrect and half of the weight on $x_i$ where $h_t$ was correct.

Recall $D_{t+1}(i) = \frac{2\xi_t}{r_t} e^{-\xi_t r_t} h_t(x_t)$. Probabilities are equal!

Pr $[Y \neq h_t(x_t)] > \sum_{i \neq h_t(x_t)} \frac{D_t(i)}{Z_t} e^{-\xi_t} = \frac{1}{Z_t} \frac{1 - \xi_t}{\xi_t} \sqrt{\frac{r_t(1 - \xi_t)}{Z_t}}$

Pr $[Y = h_t(x_t)] > \sum_{i = h_t(x_t)} \frac{D_t(i)}{Z_t} e^{-\xi_t} = \frac{1 - \xi_t}{Z_t} \frac{1}{\xi_t} \sqrt{\frac{r_t(1 - \xi_t)}{Z_t}}$

$Z_t = \sum_{i \neq h_t(x_t)} D_t(i) e^{-\xi_t r_t h_t(x_t)} + \sum_{i \neq h_t(x_t)} D_t(i) e^{\xi_t r_t h_t(x_t)} = (1 - \xi_t) e^{-\xi_t r_t} + \xi_t e^{\xi_t r_t} = 2\sqrt{r_t(1 - \xi_t)}$

Analyzing Training Error: Proof Intuition

Theorem: $\epsilon_t = 1/2 - \eta_t$ (error of $h_t$ over $\hat{h}_t$)

$err_t(H_{final}) \leq \exp \left[-2 \sum r_t^2 \right]$

• On round $t$, we increase weight of $x_i$ for which $h_t$ is wrong.
  - Then $x_i$ incorrectly classified by (wtd) majority of $h_t$'s.
  - Which implies final prob. weight of $x_i$ is large.
  - Can show probability $\geq \exp \left[-2 \sum r_t^2 \right]$

• Since sum of prob. $= 1$, can't have too many of high weight.
  - Can show # incorrectly classified $\leq m(1, Z_t)$.
  - And $(\Pi Z_t) = 0$.

Analyzing Training Error: Proof Math

**Step 1:** unwrapping recurrence: $D_{t+1}(i) = \frac{1}{m} \left( \frac{\exp(-r_t f(i))}{Z_t} \right)$

where $f(x_t) = \sum_t \alpha_t h_t(x_t)$. [Unthresholded weighted vote of $h_t$ on $x_t$]

**Step 2:** $err_t(H_{final}) \leq \Pi_{t} Z_{t}$

**Step 3:** $\Pi_{t} Z_{t} = \Pi_{t} 2\sqrt{r_t(1 - \xi_t)} = \Pi_{t} \sqrt{1 - 4\eta_t^2} \leq e^{-2 \sum r_t^2}$
Analyzing Training Error: Proof Math

**Step 1:** unwrapping recurrence: $D_{T+1}(l) = \frac{1}{m} \left( \exp(-y_l f(x_l)) \right)$
where $f(x_l) = \sum_i a_i b_i(x_i)$.

**Step 2:** $\text{err}_{\text{approx}}(H_{\text{final}}) \leq \prod_i Z_i$.

- $\text{err}_{\text{approx}}(H_{\text{final}}) = \frac{1}{m} \sum_{l=1}^{m} 1_{y_l \neq h_{\text{final}}(x_l)}$
- $\leq \frac{1}{m} \sum_{l=1}^{m} \exp(-y_l f(x_l))$
- $= \sum_{l=1}^{m} D_{T+1}(l) \prod_i Z_i = \prod_i Z_i$.

**Step 3:** $\prod_i Z_i = \prod_i 2 \sqrt{\epsilon_l (1 - \epsilon_l)} = \prod_i \sqrt{1 - 4\epsilon_l^2} \leq e^{-2 \sum_i \epsilon^2 l}$

Note: recall $Z_i = (1 - \epsilon_l)e^{-\epsilon_l} + \epsilon_l e^{\epsilon_l} = 2\sqrt{\epsilon_l (1 - \epsilon_l)}$
a_i minimizer of $a \rightarrow (1 - \epsilon_l)e^{-a} + \epsilon_l e^a$.

---

Analyzing Training Error: Proof Math

**Step 1:** unwrapping recurrence: $D_{T+1}(l) = \frac{1}{m} \left( \exp(-y_l f(x_l)) \right)$
where $f(x_l) = \sum_i a_i b_i(x_i)$.

**Step 2:** $\text{err}_{\text{approx}}(H_{\text{final}}) \leq \prod_i Z_i$.

- $\text{err}_{\text{approx}}(H_{\text{final}}) = \frac{1}{m} \sum_{l=1}^{m} 1_{y_l \neq h_{\text{final}}(x_l)}$
- $\leq \frac{1}{m} \sum_{l=1}^{m} \exp(-y_l f(x_l))$
- $= \sum_{l=1}^{m} D_{T+1}(l) \prod_i Z_i = \prod_i Z_i$.

**Step 3:** $\prod_i Z_i = \prod_i 2 \sqrt{\epsilon_l (1 - \epsilon_l)} = \prod_i \sqrt{1 - 4\epsilon_l^2} \leq e^{-2 \sum_i \epsilon^2 l}$

Note: recall $Z_i = (1 - \epsilon_l)e^{-\epsilon_l} + \epsilon_l e^{\epsilon_l} = 2\sqrt{\epsilon_l (1 - \epsilon_l)}$
a_i minimizer of $a \rightarrow (1 - \epsilon_l)e^{-a} + \epsilon_l e^a$.

---

Analyzing Training Error: Proof Math

**Step 1:** unwrapping recurrence: $D_{T+1}(l) = \frac{1}{m} \left( \exp(-y_l f(x_l)) \right)$
where $f(x_l) = \sum_i a_i b_i(x_i)$.

**Step 2:** #mistakes($H_{\text{final}}$) $= \sum_{l=1}^{m} 1_{y_l \neq H_{\text{final}}(x_l)}$

- $= \sum_{l=1}^{m} 1_{y_l f(x_l) < 0}$
- $\leq \frac{1}{m} \sum_{l=1}^{m} \exp(-y_l f(x_l))$
- $= \sum_{l=1}^{m} D_{T+1}(l) \prod_i Z_i = \prod_i Z_i$.

**Step 3:** Each mistake of $H_{\text{final}}$ has $D_{T+1}(l) \geq \frac{1}{m} \frac{1}{\prod_i Z_i}$, so total number of mistakes $\leq m(\prod_i Z_i)$.

**Step 4:** $\prod_i Z_i = \prod_i 2 \sqrt{\epsilon_l (1 - \epsilon_l)} = \prod_i \sqrt{1 - 4\epsilon_l^2} \leq e^{-2 \sum_i \epsilon^2 l}$

- Why does $(\prod_i Z_i) \rightarrow 0$?
- On round $t$, we have $1 - \epsilon$, probability mass that $h_t$ gets correct and $\epsilon$ that $h_t$ gets incorrect.
- Our reweighting replaces these with their geometric mean $\sqrt{\epsilon_l (1 - \epsilon_l)}$, which is **less than $\frac{1}{2}$**.
- So we normalize by $Z_t = 2 \sqrt{\epsilon_l (1 - \epsilon_l)}$, which is less than 1.
- If $\epsilon = \frac{1}{2}$, then geometric mean would be $\frac{1}{2}$, would normalize by 1, and get nowhere, but that makes sense since $h_t$ is just guessing!
Analyzing Training Error: Proof Intuition

**Theorem** \( \varepsilon_t = 1/2 - \gamma_t \) (error of \( h_t \) over \( \mathcal{D}_t \))

\[
\text{err}_{\varepsilon_t}(H_{\text{final}}) \leq \exp \left[ -2 \sum_t \gamma_t^2 \right]
\]

- On round \( t \), we increase weight of \( x_t \) for which \( h_t \) is wrong.
- If \( H_{\text{final}} \) incorrectly classifies \( x_t \),
  - Then \( x_t \) incorrectly classified by (wtd) majority of \( h_t \)'s.
  - Which implies final prob. weight of \( x_t \) is large.
  
  Can show probability \( \geq \frac{1}{\left( \sum_i y_i \right)} \)

- Since sum of prob. = 1, can’t have too many of high weight.
  
  Can show # incorrectly classified \( \leq \left( \prod_i y_i \right) \).
  
  And \( \left( \prod_i y_i \right) \to 0 \).

Generalization Guarantees

**Theorem** \( \text{err}_{\varepsilon_t}(H_{\text{final}}) \leq \exp \left[ -2 \sum_t \gamma_t^2 \right] \) where \( \varepsilon_t = 1/2 - \gamma_t \)

How about generalization guarantees?

Original analysis [Freund&Schapire’97]

- \( H \) space of weak hypotheses: \( d: \text{VCdim}(H) \)
  - \( H_{\text{final}} \) is a weighted vote, so the hypothesis class is:
  - \( \mathcal{G} \{ \text{all fns of the form } \text{sign}(\sum_{t=1}^T \alpha_t h_t(x)) \} \)

**Theorem** [Freund&Schapire’97]

\[
\forall \; g \in \mathcal{G}, \text{err}(g) \leq \text{err}_{\varepsilon_t}(g) + \Theta \left( \frac{T \alpha}{\sqrt{m}} \right) \; T = \# \text{ of rounds}
\]

Key reason: \( \text{VCdim}(\mathcal{G}) = O(dT) \) plus typical VC bounds.

Generalization Guarantees

- Experiments with boosting showed that the test error of the generated classifier usually does not increase as its size becomes very large.

- Experiments showed that continuing to add new weak learners after correct classification of the training set had been achieved could further improve test set performance!!
Generalization Guarantees

• Experiments with boosting showed that the test error of the generated classifier usually does not increase as its size becomes very large.
• Experiments showed that continuing to add new weak learners after correct classification of the training set had been achieved could further improve test set performance.
• These results seem to contradict FS’87 bound and Occam’s razor (in order to achieve good test error the classifier should be as simple as possible).

How can we explain the experiments?

R. Schapire, Y. Freund, P. Bartlett, W. S. Lee. present in “Boosting the margin: A new explanation for the effectiveness of voting methods” a nice theoretical explanation.

Key Idea:
Training error does not tell the whole story.
We need also to consider the classification confidence.

Classification Margin

• $H$ space of weak hypotheses. Define the convex hull of $H$ to be $conv(H) = \{f = \sum_{i=1}^{\infty} a_i h_i, a_i \geq 0, \sum a_i = 1, h_i \in H\}$
• Let $f \in conv(H), f = \sum_{i=1}^{\infty} a_i h_i, a_i \geq 0, \sum a_i = 1$.

The majority vote rule $\hat{y}$ given by $f$ (given by $\hat{y} = sign(f(x))$) predicts wrongly on example $(x,y)$ iff $y \hat{f}(x) \leq 0$.

Definition: margin of $\hat{y}$ (or of $f$) on example $(x,y)$ to be $y \hat{f}(x)$.

\[
y \hat{f}(x) = \sum_{i=1}^{\infty} [a_i h_i(x)] - \sum_{i=1}^{\infty} [a_i h_i(x)] = \sum_{x \in H_{\hat{y}(x)}} a_i - \sum_{x \notin H_{\hat{y}(x)}} a_i
\]

The margin is positive iff $y = \hat{y}(x)$.
See $|y \hat{f}(x)|$ as the strength or the confidence of the vote.
Gen. error as a function of margin
Distributions

Assume that the examples are generated i.i.d. according to some distr. $D$ over $X \times \{-1,1\}$; denote by $P_r$ the probability when $(x,y)$ is chosen from $D$.

If $S$ is a training set (a sample of size $m$, $S = \{(x_i,y_i), \cdots (x_m,y_m)\}$), then we denote by $P_r$ the probability when $(x,y)$ is chosen uniformly at random from $S$.

Theorem 2: if $H$ is finite, then with prob $\geq 1 - \delta$, $\forall f \in {\mathcal{C}}(H)$, $\forall \theta > 0$,
$$P_r[y(f) \leq 0] \leq P_r[y(f) \leq \theta] + O \left( \frac{\ln \frac{m \ln (1/\delta)}{\delta}}{\ln 2} \right)$$

Theorem 3: if $H$ has VC-dim $d$, then with prob $\geq 1 - \delta$, $\forall f \in {\mathcal{C}}(H)$, $\forall \theta > 0$,
$$P_r[y(f) \leq 0] \leq P_r[y(f) \leq \theta] + O \left( \frac{\ln \frac{m \ln (1/\delta)}{\delta}}{\sqrt{d}} \right)$$

Note: bound does not depend on the # of rounds of boosting, depends only on the complexity of the weak hyp space and the margin.

Boosting and Margins

Theorem 2: If $H$ is finite, then with prob $\geq 1 - \delta$, $\forall f \in {\mathcal{C}}(H)$, $\forall \theta > 0$,
$$P_r[y(f) \leq 0] \leq P_r[y(f) \leq \theta] + O \left( \frac{1}{\ln 2} \sqrt{\frac{m \ln (1/\delta)}{\delta}} \right)$$

Theorem 3: $\text{VCdim}(H) = d$, then with prob $\geq 1 - \delta$, $\forall f \in {\mathcal{C}}(H)$, $\forall \theta > 0$,
$$P_r[y(f) \leq 0] \leq P_r[y(f) \leq \theta] + O \left( \frac{1}{\sqrt{d}} \ln \frac{m \ln (1/\delta)}{\delta} \right)$$

Note: bound does not depend on the # of rounds of boosting, depends only on the complexity of the weak hyp space and the margin.

Boosting and Margins

Theorem 2: if $H$ is finite, then with prob $\geq 1 - \delta$, $\forall f \in {\mathcal{C}}(H)$, $\forall \theta > 0$,
$$P_r[y(f) \leq 0] \leq P_r[y(f) \leq \theta] + O \left( \frac{1}{\ln 2} \sqrt{\frac{m \ln (1/\delta)}{\delta}} \right)$$

Theorem 3: $\text{VCdim}(H) = d$, then with prob $\geq 1 - \delta$, $\forall f \in {\mathcal{C}}(H)$, $\forall \theta > 0$,
$$P_r[y(f) \leq 0] \leq P_r[y(f) \leq \theta] + O \left( \frac{1}{\sqrt{d}} \ln \frac{m \ln (1/\delta)}{\delta} \right)$$

Note: bound does not depend on the # of rounds of boosting, depends only on the complexity of the weak hyp space and the margin.

Boosting summary

• Shift in mindset: goal is now just to find classifiers a bit better than random guessing.
• Backed up by solid foundations.
• AdaBoost work and its variations well in practice with many kinds of data (one of the top 10 algorithms).
• Very general: can use any given weak learning algorithm.
• AdaBoost is very fast (single pass through data each round) & simple to code, no parameters to tune.
• Relevant for big data age: quickly focuses on “core difficulties”, so well-suited to distributed settings, where data must be communicated efficiently [Balcan-Bun-Fine-Mansour COLT12].