FilterBoost: Regression and Classification on Large Datasets

NIPS’07 paper by Joseph K. Bradley and Robert E. Schapire

* some slides reused from the NIPS ’07 presentation
Typical Boosting Framework

• Batch Framework

Dataset

Sampled i.i.d. from target distribution $D$
Typical Boosting Framework

• Batch Framework

Dataset

Sampled i.i.d. from target distribution \( D \)
Typical Boosting Framework

- Batch Framework

![Diagram of Boosting Framework]

Dataset

- Sampled i.i.d. from target distribution $D$

Booster

Final Hypothesis
Typical Boosting Framework

- Batch Framework

The booster must always have access to the entire dataset.

Dataset

Sampled i.i.d. from target distribution $D$

Final Hypothesis

Booster
Motivation for a New Framework

• In the typical framework, the boosting algorithm must have access to the entire data set.

• This limits the application to scenarios with very large data sets.

• Computationally infeasible because in each round, the distribution information on each point is updated.

• New ideas:
  • use a data stream instead of the entire (fixed) data set.
  • train on new subsets of data in each round.
Filtering Framework

- Boost for 1000 rounds: only store $\sim 1/1000$ of data at a time.

Data Oracle

\[ \sim D \]

Booster

Final Hypothesis

Stores tiny fraction of data!
Paper’s Results

• A new boosting-by-filtering algorithm.
• Provable guarantees.
• Applicable to both classification and conditional probability estimation.
• Good empirical performance.
AdaBoost (batch boosting)

- Given: Fixed data set $S$.
- In each round $t$,
  - choose distribution $D_t$ over $S$.
  - choose hypothesis $h_t$.
  - estimate error $\epsilon_t$ of $h_t$ with $D_t$.
  - give $h_t$ a weight of $\alpha_t = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t}$
- Output: hypothesis:
  $$h(x) := \sum_{i=1}^{T} \alpha_t h_t(x)$$
AdaBoost (batch boosting)

- Given: Fixed data set $S$.
- In each round $t$,
  - choose distribution $D_t$ over $S$.
  - choose hypothesis $h_t$.
  - estimate error $\epsilon_t$ of $h_t$ with $D_t$.
  - give $h_t$ a weight of $\alpha_t = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t}$
- Output: hypothesis:

$$h(x) := \sum_{i=1}^{T} \alpha_t h_t(x)$$

- Higher weight to misclassified examples.

$D_t$ forces the algorithm to correctly classify “harder” examples in later rounds.

- In filtering, no “fixed” data set $S$. What about $D_t$?
Filtering Idea

- Key idea: Simulate $D_t$ using the filter (accept only “hard examples”).
FilterBoost: Main Algorithm

• Given: Oracle to distribution $D$.

• In each round $t$,
  • use Filter$_t$ to obtain $D_t$ (sample $S_t$ really)
  • choose hypothesis $h_t$ that does well on $S_t$
  • estimate error $\epsilon_t$ of $h_t$ by using the oracle again.
  • give $h_t$ a weight of
    \[
    \alpha_t = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t}
    \]

• Output: hypothesis:
  \[
  h(x) := \sum_{i=1}^{T} \alpha_t h_t(x)
  \]
Filtering: Details

- Simulate $D_t$ using rejection sampling.
- Higher weight to misclassified examples.
- If $yH(x) = 1$, then the label and hypothesis agree, low probability of being accepted.
- Otherwise, if $yH(x) = -1$, then misclassified example, high probability of being accepted.
- So, try $D(x, y) = \exp(-yH(x))$.
- Difficulty: too much weight on too few examples.
Filtering: Details

- Truncated exponential weights work for filtering.
  [M-AdaBoost, Domingo & Watanabe, 00]
Filtering: Details

- **FilterBoost**: based on AdaBoost for Logistic Regression. Minimize logistic loss leads to logistic weights.

\[
\Pr[\text{accept}(x, y)] = \frac{1}{1 + e^{yH(x)}}.
\]

\[
\Pr[\text{accept}(x, y)] = \min\{1, e^{-yH(x)}\}.
\]
FilterBoost: Main Algorithm

- Given: Oracle to distribution $D$.
- In each round $t$,
  - use Filter$_t$ to obtain $D_t$ (sample $S_t$ really)
  - choose hypothesis $h_t$ that does well on $S_t$
  - estimate error $\epsilon_t$ of $h_t$ by using the oracle again.
  - give $h_t$ a weight of $\alpha_t = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t}$
- Output: hypothesis:

\[ h(x) := \sum_{i=1}^{T} \alpha_t h_t(x) \]
FilterBoost: Main Algorithm

• Given: Oracle to distribution $D$.

• In each round $t$,
  • use Filter$_t$ to obtain $D_t$ (sample $S_t$ really)
  • choose hypothesis $h_t$ that does well on $S_t$
  • estimate error $\epsilon_t$ of $h_t$ by using the oracle again.
  • give $h_t$ a weight of $\alpha_t = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t}$

• Output: hypothesis:
  $$h(x) := \sum_{i=1}^{T} \alpha_t h_t(x)$$

Q1. How much time does Filter$_t$ take?

A1. If filter takes too long, then hypothesis is accurate enough.

Q2. How many boosting rounds are needed?

A2. If weak hypothesis’ error bounded away from $1/2$, we make good progress.

Q3. How can we estimate hypothesis errors?

A3. Adaptive Sampling
[Watanabe, 00]
• Interpreted as an additive logistic regression model. Suppose

$$\log \frac{\Pr[y = 1|x]}{\Pr[y = -1|x]} = \sum_t f_t(x) = F(x)$$

• Which implies $$\Pr[y = 1|x] = \frac{1}{1 + e^{-F(x)}}.$$  

• In the case of FilterBoost, $$f_t(x) = \alpha_t h_t(x).$$

• Expected Negative log-likelihood of an example is:

$$\pi(F) = \mathbb{E} \left[ -\ln \frac{1}{1 + e^{-yF(x)}} \right]$$

• FilterBoost minimizes this function. Like AdaBoost, gradient descent is used to determine weak learner and step size.

$$\pi(F + \alpha_t h_t).$$

\[ \text{weak learner} \]
\[ \text{step-size} \]
FilterBoost: Details

• Second order expansion of $\pi(F + \alpha h)|_{h=0}$:

\[
\pi(F + \alpha h) = \pi(F) + \alpha h \pi'(F) + \frac{\alpha^2 h^2}{2} \pi''(F)
\]

\[
= \mathbb{E} \left[ \ln(1 + e^{-yF(x)}) - \frac{y\alpha h}{1 + eyF(x)} + \frac{1}{2} \frac{y^2 \alpha^2 h^2 e^yF(x)}{(1 + e^yF(x))^2} \right]
\]

• For positive $\alpha$, this expression is minimized when we maximize:

\[
\mathbb{E} \left[ \frac{yh(x)}{1 + eyF(x)} \right] \equiv \mathbb{E}_q[yh(x)]
\]

• This is maximized for $f(x) = \text{sign}(\mathbb{E}_q[y|x])$.

• Once $h(x)$ is fixed, $\alpha$ is determined by to minimize the upper-bound $\pi(F + \alpha h) \leq \mathbb{E}[e^{-y(F(x)+\alpha h(x))}]$

\[
\alpha = \frac{1}{2} \log \left( \frac{1/2 + \gamma}{1/2 - \gamma} \right).
\]
FilterBoost: Details (1)

Define $F_t(x) \equiv \sum_{t'=1}^{t-1} \alpha_{t'} h_{t'}(x)$

Algorithm FilterBoost accepts Oracle(), $\varepsilon$, $\delta$, $\tau$:

For $t = 1, 2, 3, \ldots$

\[
\delta_t \leftarrow \frac{\delta}{3t(t+1)}
\]

Call Filter($t$, $\delta_t$, $\varepsilon$) to get

$m_t$ examples to train WL; get $h_t$

\[
\hat{\gamma}_t' \leftarrow \text{getEdge}(t, \tau, \delta_t, \varepsilon)
\]

\[
\alpha_t \leftarrow \frac{1}{2} \ln \left( \frac{1/2 + \hat{\gamma}_t'}{1/2 - \hat{\gamma}_t'} \right)
\]

Define $H_t(x) = \text{sign}\left(F_{t+1}(x)\right)$

(Algorithm exits from Filter() function.)
FilterBoost: Details (2)

Function $Filter(t, \delta_t, \varepsilon)$ returns $(x, y)$

Define $r = \#$ calls to Filter so far on round $t$

\[ \delta_t' \leftarrow \frac{\delta_t}{r(r+1)} \]

For $(i = 0; i < \frac{2}{\varepsilon} \ln(\frac{1}{\delta_t'}); i = i + 1)$:

\[ (x, y) \leftarrow Oracle() \]

\[ q_t(x, y) \leftarrow \frac{1}{1 + e^{yF_t(x)}} \]

Return $(x, y)$ with probability $q_t(x, y)$

End algorithm; return $H_{t-1}$
Function $getEdge(t, \tau, \delta_t, \varepsilon)$ returns $\hat{\gamma}_t$

Let $m \leftarrow 0$, $n \leftarrow 0$, $u \leftarrow 0$, $\alpha \leftarrow \infty$

While ($|u| < \alpha(1 + 1/\tau)$):

$(x, y) \leftarrow Filter(t, \delta_t, \varepsilon)$
$n \leftarrow n + 1$
$m \leftarrow m + I(h_t(x) = y)$
$u \leftarrow m/n - 1/2$
$\alpha \leftarrow \sqrt{(1/2n) \ln(n(n + 1)/\delta_t)}$

Return $u/(1 + \tau)$
FilterBoost: Theory

• Theorem:
  Assume that the weak hypotheses have edge at least $\gamma$. Let $\epsilon$ be the target error rate. FilterBoost produces a final hypothesis with error less than $\epsilon$ in $T$ rounds, where

  $$T = \tilde{O}\left(\frac{1}{\epsilon\gamma^2}\right)$$

• The real bound is given by: $T > \frac{2 \ln(2)}{\epsilon(1 - 2\sqrt{1/4 - \gamma^2})}$

• Proof elements:
  - Step 1: $err_t \leq 2p_t$. probability of accepting an example in round $t$
  - Step 2: $\pi_t - \pi_{t+1} \geq p_t \left(1 - 2\sqrt{1/4 - \gamma_t^2}\right)$.
  - Assume that for all $t \in \{1, \ldots, T\}$, $err_t \geq \epsilon$. Contradiction.
FilterBoost: Theory

• Step 1:

  • recall that \( \Pr[\text{accept}(x, y)] = \frac{1}{1 + e^{y H(x)}} \). Call it \( q_t(x, y) \).

\[
\begin{align*}
err_t &= \Pr_D[H_t(x) \neq y] = \Pr_D[y F_{t-1}(x) \leq 0] \\
&= \Pr_D[q_t(x, y) \geq 1/2] \leq 2 \cdot E_D[q_t(x, y)] \\
&= 2p_t \text{ (using Markov’s inequality above)}
\end{align*}
\]
FilterBoost: Theory

• Step II:
  - recall that \( \pi(F) = \mathbb{E} \left[ -\ln \frac{1}{1 + e^{-yF(x)}} \right] \).
  - expanding the expectation, \( \pi_t = \sum_{(x,y)} D(x,y) \ln(1 - q_t(x,y)) \)

\[
\pi_t - \pi_{t+1} = \sum_{(x,y)} D(x,y) \ln \left( \frac{1 - q_t(x,y)}{1 - q_{t+1}(x,y)} \right)
\]

\[
q_t(x,y) = \frac{1}{1 + e^{yF_t(x)}}, \quad F_t(x) = \sum_{t'=1}^{t-1} \alpha_{t'} h_{t'}(x),
\]

\[
eyF_t(x) = \frac{1}{q_t(x,y)} - 1 \text{ and}
\]

\[
q_{t+1}(x,y) = \frac{1}{1 + e^{yF_t(x) + \alpha_t y h_t(x)}}
\]

\[
q_{t+1}(x,y) = \frac{1}{1 + (\frac{1}{q_t(x,y)} - 1) e^{v_t(x,y)}} = \frac{q_t(x,y)}{q_t(x,y) + (1 - q_t(x,y)) e^{v_t(x,y)}}
\]
FilterBoost: Theory

\[
\pi_t - \pi_{t+1} = -\sum_{(x,y)} D(x, y) \ln(q_t(x, y)e^{-v_t(x,y)} + 1 - q_t(x, y)) \\
\geq -\sum_{(x,y)} D(x, y)(-q_t(x, y) + q_t(x, y)e^{-v_t(x,y)}) \\
= \sum_{(x,y)} D(x, y)q_t(x, y) - \sum_{(x,y)} D(x, y)q_t(x, y)e^{-v_t(x,y)}
\]

- Let \( D_t(x, y) = \frac{D(x, y)q_t(x, y)}{p_t} \).

\[
\pi_t - \pi_{t+1} \geq p_t - p_t \sum_{(x,y)} D_t(x, y)e^{-\alpha_t y h_t(x)}
\]

- Recall that \( \alpha_t = \frac{1}{2} \ln\left(\frac{1/2+\gamma_t}{1/2-\gamma_t}\right) \) & \( \epsilon_t \equiv \Pr_{D_t}[\text{sign}(h_t(x)) \neq y] \)

\[
\sum_{(x,y)} D_t(x, y)e^{-\alpha_t y h_t(x)} = e^{-\alpha_t}(1 - \epsilon_t) + e^{\alpha_t} \epsilon_t = 2\sqrt{\frac{1}{4} - \gamma_t^2}
\]
FilterBoost vs. Rest

- Comparison (as reported in the paper):

<table>
<thead>
<tr>
<th>Method</th>
<th>Need edges decreasing?</th>
<th>Need bound on min edge?</th>
<th>Inf. weak learner space</th>
<th># rounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>M-AdaBoost [Domingo &amp; Watanabe,00]</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>$1/\varepsilon$</td>
</tr>
<tr>
<td>AdaFlat [Gavinsky,02]</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>$1/\varepsilon^2$</td>
</tr>
<tr>
<td>GiniBoost [Hatano,06]</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>$1/\varepsilon$</td>
</tr>
<tr>
<td>FilterBoost</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>$1/\varepsilon$</td>
</tr>
</tbody>
</table>
FilterBoost: Details

• In the previous analysis, overlooked the probability of failure introduced by the three steps:
  • training the weak learner
  • deciding when to stop boosting
  • estimating the edges
Experiments

• Tested FilterBoost against M-AdaBoost, AdaBoost, Logistic AdaBoost.

• Synthetic and real data sets.

• Tested: classification and conditional probability estimation.
Experiments: Classification

- Noisy Majority Vote (synthetic).
- Decision stumps are the weak learners.
- 500,000 examples.

FilterBoost achieves high accuracy fast.
Experiments: CPE

• Conditional Probability Estimation
Conclusion

• FilterBoost good for boosting over large data sets.
• Fewer assumptions, better guarantees.
• Validated empirically, in classification and conditional probability estimation.