Exponential Weights Algorithms for Online Learning

Yoav Freund

May 19, 2011
Outline
The Halving Algorithm
Halving Algorithm
\textbf{Hedge}(\eta) Algorithm
Hedging vs. Halving
Bound on total loss
Upper bound on \( \sum_{i=1}^{N} w_i^{T+1} \)
Lower bound on \( \sum_{i=1}^{N} w_i^{T+1} \)
Combining Upper and Lower bounds
tuning \( \eta \)
Lower Bounds
Some useful loss functions
Combining experts in the log loss framework
The online Bayes Algorithm
The performance bound
Summary table
A very simple online prediction problem

- Binary prediction.
- At time $t$ the algorithm knows receives predictions from $N$ experts.
- We know that one of the experts is \textit{perfect}
- How should the algorithm predict?
Example trace for Halving Algorithm

<table>
<thead>
<tr>
<th></th>
<th>$t = 1$</th>
<th>$t = 2$</th>
<th>$t = 3$</th>
<th>$t = 4$</th>
<th>$t = 5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>expert1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>expert2</td>
<td>1</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>expert3</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>expert4</td>
<td>1</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>expert5</td>
<td>1</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>expert6</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>expert7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>expert8</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>-</td>
</tr>
</tbody>
</table>

alg.  | 1       | 0       | 1       | 1       | 0       |
outcome | 1       | 1       | 1       | 0       | 0       |
Mistake bound for Halving algorithm

- Each time algorithm makes a mistakes, the pool of perfect experts is halved (at least).
- We assume that at least one expert is perfect.
- Number of mistakes is at most $\log_2 N$.
- No stochastic assumptions whatsoever.
- Proof is based on combining a lower and upper bounds on the number of perfect experts.
The hedging problem

- possible outcomes = \{0, 1\}
- At each time step $t = 1, 2, \ldots, T$:
  - each expert predicts with $\phi^t_i \in [0, 1]$.
  - Algorithm predicts with $\phi^t_A \in [0, 1]$.
  - Outcome $o^t \in \{0, 1\}$ is revealed.
  - Experts suffer loss equal to $\ell^t_i = |\phi^t_i - o^t|$
  - Algorithm suffers loss equal to $\ell^t_A = |\phi^t_A - o^t|$
- **Goal**: minimize cumulative loss of algorithm
Hedging vs. Halving

- Like halving - we want to zoom into best action (expert).
- Unlike halving - no action is perfect.
- Basic idea - reduce probability of lossy actions, but not all the way to zero.
- **Modified Goal:** minimize difference between expected total loss and minimal total loss of repeating one action.

\[
\sum_{t=1}^{T} \ell^t_A - \min_i \left( \sum_{t=1}^{T} \ell^t_i \right)
\]
The **Hedge**($\eta$)**Algorithm**

Consider action $i$ at time $t$

- **Total loss:**
  \[
  L_i^t = \sum_{s=1}^{t-1} \ell_i^s
  \]

- **Weight:**
  \[
  w_i^t = w_i^1 e^{-\eta L_i^t}
  \]
  
  Note freedom to choose initial weight ($w_i^1$) $\sum_{i=1}^{n} w_i^1 = 1$.

- $\eta > 0$ is the learning rate parameter. Halving: $\eta \rightarrow \infty$

- algorithm’s prediction:
  \[
  \phi_A^t = \frac{\sum_{i=1}^{N} w_i^t \phi_i^t}{\sum_{i=1}^{N} w_i^t}
  \]
Choosing the initial weights

- Giving an action high initial weight makes alg perform well if that action performs well.
- If good action has low initial weight, our total loss will be larger.
- As $\sum_{i=1}^{n} w_i^1 = 1$ increasing one weight implies decreasing some others.
- Plays a similar role to prior distribution in Bayesian algorithms.
Bound on the loss of Hedge(\(\eta\)) Algorithm

**Theorem (main theorem)**

For any sequence of loss vectors \(\ell^1, \ldots, \ell^T\), and for any \(i \in \{1, \ldots, N\}\), we have

\[
L_{\text{Hedge}(\eta)} \leq \frac{-\ln(w_i^1) + \eta L_i}{1 - e^{-\eta}}.
\]

**Proof**: by combining upper and lower bounds on
\[
\sum_{i=1}^{N} w_i^{T+1}
\]
Lemma (upper bound)

For any sequence of loss vectors $\ell^1, \ldots, \ell^T$ we have

$$\ln \left( \sum_{i=1}^{N} w_i^{T+1} \right) \leq -(1 - e^{-\eta}) L_{\text{Hedge}(\eta)}.$$
Proof of upper bound (slide 1)

- If $a \geq 0$ then $a^r$ is convex.
- For $r \in [0, 1]$, $a^r \leq 1 - (1 - a)r$
Proof of upper bound (slide 2)

Applying \( a^r \leq 1 - (1 - a)^r \) where \( a = e^{-\eta}, r = \ell_i^t \)

\[
\sum_{i=1}^{N} w_{i}^{t+1} = \sum_{i=1}^{N} w_{i}^{t} e^{-\eta \ell_i^t} \\
\leq \sum_{i=1}^{N} w_{i}^{t} (1 - (1 - e^{-\eta}) \ell_i^t) \\
= \left( \sum_{i=1}^{N} w_{i}^{t} \right) \left( 1 - (1 - e^{-\eta}) \frac{w^{t}}{\sum_{i=1}^{N} w_{i}^{t}} \cdot \ell^t \right) \\
= \left( \sum_{i=1}^{N} w_{i}^{t} \right) \left( 1 - (1 - e^{-\eta}) p^{t} \cdot \ell^t \right)
\]
Proof of upper bound (slide 3)

- Combining
\[
\sum_{i=1}^{N} w_{i}^{t+1} \leq \left( \sum_{i=1}^{N} w_{i}^{t} \right) (1 - (1 - e^{-\eta}) p^{t} \cdot \ell^{t})
\]
- for \( t = 1, \ldots, T \)
- yields
\[
\sum_{i=1}^{N} w_{i}^{T+1} \leq \prod_{t=1}^{T} (1 - (1 - e^{-\eta}) p^{t} \cdot \ell^{t})
\]
\[
\leq \exp \left( -(1 - e^{-\eta}) \sum_{t=1}^{T} p^{t} \cdot \ell^{t} \right)
\]
since \( 1 + x \leq e^{x} \) for \( x = -(1 - e^{-\eta}) \).
Lower bound on $\sum_{i=1}^{N} w_i^{T+1}$

For any $j = 1, \ldots, N$:

$$\sum_{i=1}^{N} w_i^{T+1} \geq w_j^{T+1} = w_j^1 e^{-\eta L_j}$$
Combining Upper and Lower bounds

- Combining bounds on \( \ln \left( \sum_{i=1}^{N} w_i^{T+1} \right) \)

\[
\ln w_j^1 - \eta L_j \leq \ln \sum_{i=1}^{N} w_i^{T+1} \leq -(1 - e^{-\eta}) \sum_{t=1}^{T} p^t \cdot \ell^t
\]

- Reversing signs, using \( L_{\text{Hedge}}(\eta) = \sum_{t=1}^{T} p^t \cdot \ell^t \) and reorganizing we get

\[
L_{\text{Hedge}}(\eta) \leq \frac{-\ln(w_i^1) + \eta L_i}{1 - e^{-\eta}}
\]
Tuning $\eta$

How to Use Expert Advice
Tuning $\eta$

- Suppose $\min_i L_i \leq \tilde{L}$
- set

$$\eta = \ln \left( 1 + \sqrt{\frac{2 \ln N}{\tilde{L}}} \right) \approx \sqrt{\frac{2 \ln N}{\tilde{L}}}$$

- use uniform initial weights $w^1 = \langle 1/N, \ldots, 1/N \rangle$
- Then

$$L_{\text{Hedge}(\eta)} \leq \frac{-\ln(w^1_i) + \eta L_i}{1 - e^{-\eta}} \leq \min_i L_i + \sqrt{2\tilde{L} \ln N + \ln N}$$
Tuning $\eta$ as a function of $T$

- trivially $\min_i L_i \leq T$, yielding

$$L_{\text{Hedge}(\eta)} \leq \min_i L_i + \sqrt{2T \ln N} + \ln N$$

- per iteration we get:

$$\frac{L_{\text{Hedge}(\eta)}}{T} \leq \min_i \frac{L_i}{T} + \sqrt{\frac{2 \ln N}{T}} + \frac{\ln N}{T}$$
How good is this bound?

- **Very good!** There is a closely matching lower bound!
- There exists a stochastic adversarial strategy such that with high probability for any hedging strategy $S$ after $T$ trials,

$$L_S - \min_{i} L_i \geq (1 - o(1))\sqrt{2T \ln N}$$

- The adversarial strategy is random, extremely simple, and does not depend on the hedging strategy!
The adversarial strategy

- Adversary sets each loss $\ell_t^i$ independently at random to 0 or 1 with equal probabilities $(1/2, 1/2)$.
- Obviously, nothing to learn!
  $$L_S \approx T/2.$$  
- On the other hand $\min_i L_i \approx T/2 - \sqrt{2T \ln N}$
- The difference $L_S - \min_i L_i$ is due to unlearnable random fluctuations!
- Detailed proof quite involved. See games paper.
Summary

- Given learning rate $\eta$ the $\text{Hedge}(\eta)$ algorithm satisfies

$$L_{\text{Hedge}(\eta)} \leq \frac{\ln N + \eta L_i}{1 - e^{-\eta}}$$

- Setting $\eta \approx \sqrt{\frac{2\ln N}{T}}$ guarantees

$$L_{\text{Hedge}(\eta)} \leq \min_i L_i + \sqrt{2T \ln N} + \ln N$$

- A trivial random data, in which there is nothing to be learned forces any algorithm to suffer this total loss

- Learning rate $\eta$ has to be tuned to achieve optimal performance.
The loss function framework

- **Outcomes:** $\omega^1, \omega_2, \ldots, \omega^t \in \{0, 1\}$
- **Predictions:** $\gamma^1, \gamma^2, \ldots, \gamma^t \in [0, 1]$  
- **Loss function:** $\lambda : (\omega, \gamma) \rightarrow \mathbb{R}$
Absolute loss

\[
\lambda(\omega, \gamma) = |\omega - \gamma|
\]

- Probability of making a mistake if predicting 0 or 1 using a biased coin
- If \( P[\omega^t = 1] = q, \ P[\omega^t = 0] = 1 - q \), then the optimal prediction is

\[
\gamma^t = \begin{cases} 
1 & \text{if } q > 1/2, \\
0 & \text{otherwise}
\end{cases}
\]
Log loss (Entropy loss)

\[ \lambda_{\text{ent}}(\omega, \gamma) = \omega \ln \frac{\omega}{\gamma} + (1 - \omega) \ln \frac{1 - \omega}{1 - \gamma} \]

- Equivalent to minus log likelihood of \( \omega \) with respect to the distribution \((\gamma, 1 - \gamma)\)
- If \( P[\omega_t = 1] = q \), optimal prediction \( \gamma^t = q \)
- Unbounded loss.
- Not symmetric \( \exists p, q \quad \lambda(p, q) \neq \lambda(q, p) \).
- No triangle inequality
  \( \exists p_1, p_2, p_3 \quad \lambda(p_1, p_3) > \lambda(p_1, p_2) + \lambda(p_2, p_3) \)
Some useful loss functions

Square loss (Breier Loss)

$$\lambda_{sq}(\omega, \gamma) = (\omega - \gamma)^2$$

- $P[\omega^t = 1] = q$, $P[\omega^t = 0] = 1 - q$, optimal prediction $\gamma^t = q$
- Bounded loss.
- Defines a metric (symmetric and triangle ineq.)
- Corresponds to regression.
Hellinger Loss

\[ \lambda_{\text{hel}}(\omega, \gamma) = \frac{1}{2} \left( (\sqrt{\omega} + \sqrt{\gamma})^2 + (\sqrt{1 - \omega} + \sqrt{1 - \gamma})^2 \right) \]

- If \( P[\omega^t = 1] = q \), \( P[\omega^t = 0] = 1 - q \), optimal prediction \( \gamma^t = q \)
- Loss is bounded.
- Defines a metric.
- \( \lambda_{\text{hel}}(p, q) \approx \lambda_{\text{ent}}(p, q) \) when \( p \approx q \) and \( p, q \in (0, 1) \)
Hedge(\eta)

Some useful loss functions

Structureless bounded loss

- Prediction is a distribution \( \gamma = \langle p_1, \ldots, p_N \rangle \), \( p_i \geq 0 \), \( \sum_{i=1}^{N} p_i = 1 \)
- Outcome is a loss vector \( \omega = \langle \omega_1, \ldots, \omega_N \rangle \), \( 0 \leq \omega_i \leq 1 \)
- Loss is the dot product: \( \lambda_{\text{dot}}(\omega, \gamma) = \gamma \cdot \omega \)
- Corresponds to the hedging game.
- For hedge loss the regret is \( \Omega(\sqrt{T \log N}) \).
- For the log loss the regret is \( O(\log N) \).
- Which losses behave like entropy loss and which behave like hedge loss?
The log-loss framework

- Algorithm $A$ predicts a sequence $c_1, c_2, \ldots, c_T$ over alphabet $\Sigma = \{1, 2, \ldots, k\}$.
- The prediction for the $c^t$th is a distribution over $\Sigma$:
  $$p^t_A = \langle p^t_A(1), p^t_A(2), \ldots, p^t_A(k) \rangle$$
- When $c^t$ is revealed, the loss we suffer is $- \log p^t_A(c^t)$.
- The cumulative log loss, which we wish to minimize, is
  $$L_A^T = - \sum_{t=1}^{T} \log p^t_A(c^t)$$
- $\lceil L_A^T \rceil$ is the code length if $A$ is combined with arithmetic coding.
The game

- Prediction algorithm $A$ has access to $N$ experts.
- The following is repeated for $t = 1, \ldots, T$
  - Experts generate predictive distributions: $p^t_1, \ldots, p^t_N$
  - Algorithm generates its own prediction $p^t_A$
  - $c^t$ is revealed.
- **Goal:** minimize regret:

\[
- \sum_{t=1}^{T} \log p^t_A(c^t) + \min_{i=1,\ldots,N} \left( - \sum_{t=1}^{T} \log p^t_i(c^t) \right)
\]
The online Bayes Algorithm

- Total loss of expert $i$ $L_i^t = -\sum_{s=1}^{t} \log p_i^s(c^s)$; $L_i^0 = 0$
- Weight of expert $i$ $w_i^t = w_i^1 e^{-L_i^{t-1}} = w_i^1 \prod_{s=1}^{t-1} p_i^s(c^s)$
  - $w_i^t$ is the un-normalized posterior probability of model $i$ at time $t$.
- Freedom to choose initial weights.
  - $w_i^1 \geq 0$, $\sum_{i=1}^{n} w_i^1 = 1$
- Prediction of algorithm $A$ $p_A^t = \frac{\sum_{i=1}^{N} w_i^t p_i^t}{\sum_{i=1}^{N} w_i^t}$
The **Hedge**$(\eta)$**Algorithm**

Consider action $i$ at time $t$

- Total loss:
  \[
  L_i^t = \sum_{s=1}^{t-1} \ell_s^i
  \]

- Weight:
  \[
  w_i^t = w_i^1 e^{-\eta L_i^t}
  \]

  Note freedom to choose initial weight ($w_i^1$) $\sum_{i=1}^{n} w_i^1 = 1$.

- $\eta > 0$ is the learning rate parameter. Halving: $\eta \to \infty$

- Probability:
  \[
  p_i^t = \frac{w_i^t}{\sum_{j=1}^{N} w_j^t}, \quad p^t = \frac{w^t}{\sum_{j=1}^{N} w_j^t}
  \]
Cumulative loss vs. Final total weight

Total weight: \( W^t = \sum_{i=1}^{N} w_i^t \)

\[
\frac{W^{t+1}}{W^t} = \frac{\sum_{i=1}^{N} w_i^t e^{\log p_i^t(c^t)}}{\sum_{i=1}^{N} w_i^t} = \frac{\sum_{i=1}^{N} w_i^t p_i^t(c^t)}{\sum_{i=1}^{N} w_i^t} = p_A^t(c^t)
\]

\[-\log \frac{W^{t+1}}{W^t} = -\log p_A^t(c^t)\]

\[-\log W^{T+1} = -\log \frac{W^{T+1}}{W^1} = -\sum_{t=1}^{T} \log p_A^t(c^t) = L_A^T\]

**EQUALITY** not bound!
Simple Bound

- Use uniform initial weights $w_i^1 = 1/N$
- Total Weight is at least the weight of the best expert.

\[
L_T^A = -\log W^{T+1} = -\log \sum_{i=1}^{N} w_i^{T+1}
\]

\[
= -\log \sum_{i=1}^{N} \frac{1}{N} e^{-L_i^T} = \log N - \log \sum_{i=1}^{N} e^{-L_i^T}
\]

\[
\leq \log N - \log \max_i e^{-L_i^T} = \log N + \min_i L_i^T
\]

- Dividing by $T$ we get $\frac{L_T^A}{T} = \min_i \frac{L_i^T}{T} + \frac{\log N}{T}$
Upper bound on $\sum_{i=1}^{N} w_{i}^{T+1}$ for $\text{Hedge} (\eta)$

Lemma (upper bound)
For any sequence of loss vectors $\ell^1, \ldots, \ell^T$ we have

$$\ln \left( \sum_{i=1}^{N} w_{i}^{T+1} \right) \leq -(1 - e^{-\eta}) L_{\text{Hedge} (\eta)}.$$
Summary of bounds for mixable losses

<table>
<thead>
<tr>
<th>Loss Functions:</th>
<th>$c$ values: $(\eta = 1/c)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L_{sq}(p, q)$</td>
<td>$\text{pred}_{w\text{mean}}(v, x)$: 2</td>
</tr>
<tr>
<td>$L_{\text{ent}}(p, q)$</td>
<td>1</td>
</tr>
<tr>
<td>$L_{\text{hel}}(p, q)$</td>
<td>1</td>
</tr>
</tbody>
</table>

*Figure 2. $(c, 1/c)$-realizability: $c$ values for loss and prediction function pairing*