The **Hedge**(η)Algorithm

Yoav Freund

January 8, 2014

Outline

Outline

Hedge(η)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

Outline

Hedge(η)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 η

Outline

Hedge(η)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 η

Lower Bounds

The hedging problem

- *N* possible actions

The hedging problem

- N possible actions
- At each time step $t = 1, 2, \ldots, T$:

The hedging problem

- N possible actions
- At each time step $t = 1, 2, \ldots, T$:
  - Algorithm chooses a distribution $\mathbf{p}^t$ over actions.

The hedging problem

- N possible actions
- At each time step $t = 1, 2, \ldots, T$:
  - Algorithm chooses a distribution $\mathbf{p}^t$ over actions.
  - Losses $0 \leq \ell_i^t \leq 1$ of all actions $i = 1, \ldots, N$ are revealed.

The hedging problem

- N possible actions
- At each time step $t = 1, 2, \ldots, T$:
  - Algorithm chooses a distribution $\mathbf{p}^t$ over actions.
  - Losses $0 \le \ell_i^t \le 1$ of all actions $i = 1, \ldots, N$ are revealed.
  - Algorithm suffers **expected** loss $\mathbf{p}^t \cdot \ell^t$

The hedging problem

- N possible actions
- At each time step $t = 1, 2, \ldots, T$:
  - Algorithm chooses a distribution $\mathbf{p}^t$ over actions.
  - Losses $0 \leq \ell_i^t \leq 1$ of all actions $i = 1, \ldots, N$ are revealed.
  - Algorithm suffers **expected** loss $\mathbf{p}^t \cdot \ell^t$
- **Goal:** minimize total expected loss

The hedging problem

- N possible actions
- At each time step $t = 1, 2, \ldots, T$:
  - Algorithm chooses a distribution $\mathbf{p}^t$ over actions.
  - Losses $0 \le \ell_i^t \le 1$ of all actions $i = 1, \ldots, N$ are revealed.
  - Algorithm suffers **expected** loss $\mathbf{p}^t \cdot \ell^t$
- **Goal:** minimize total expected loss
- **What is $\mathbf{p}^t$?**

The hedging problem

- N possible actions
- At each time step $t = 1, 2, \ldots, T$:
  - Algorithm chooses a distribution $\mathbf{p}^t$ over actions.
  - Losses $0 \leq \ell_i^t \leq 1$ of all actions $i = 1, \ldots, N$ are revealed.
  - Algorithm suffers **expected** loss $\mathbf{p}^t \cdot \ell^t$
- **Goal:** minimize total expected loss
- **What is $\mathbf{p}^t$?**
  - Distribution wealth in a portfolio.

The hedging problem

- N possible actions
- At each time step $t = 1, 2, \ldots, T$:
  - Algorithm chooses a distribution $\mathbf{p}^t$ over actions.
  - Losses $0 \leq \ell_i^t \leq 1$ of all actions $i = 1, \ldots, N$ are revealed.
  - Algorithm suffers **expected** loss $\mathbf{p}^t \cdot \ell^t$
- **Goal:** minimize total expected loss
- **What is $\mathbf{p}^t$?**
  - Distribution wealth in a portfolio.
  - The algorithm chooses one of the actions at random.

The hedging problem

- N possible actions
- At each time step $t = 1, 2, \ldots, T$:
  - Algorithm chooses a distribution $\mathbf{p}^t$ over actions.
  - Losses $0 \leq \ell_i^t \leq 1$ of all actions $i = 1, \ldots, N$ are revealed.
  - Algorithm suffers **expected** loss $\mathbf{p}^t \cdot \ell^t$
- **Goal:** minimize total expected loss
- **What is $\mathbf{p}^t$?**
  - Distribution wealth in a portfolio.
  - The algorithm chooses one of the actions at random.
- Prediction is a **game** played between algorithm and nature, in which the goal of the algorithm is to minimize **regret**.

Experts vs. actions

- Experts interaction (1 round):

Experts vs. actions

- Experts interaction (1 round):
  1. **Experts** make their predictions.

Experts vs. actions

- Experts interaction (1 round):
  1. **Experts** make their predictions.
  2. **Algorithm** makes it's prediction.

Experts vs. actions

- Experts interaction (1 round):
  1. **Experts** make their predictions.
  2. **Algorithm** makes it's prediction.
  3. **Nature** Chooses label/outcome.

Experts vs. actions

- Experts interaction (1 round):
  1. **Experts** make their predictions.
  2. **Algorithm** makes it's prediction.
  3. **Nature** Chooses label/outcome.
  4. **Loss** is associated with each action according to a loss function.

Experts vs. actions

- Experts interaction (1 round):
  1. **Experts** make their predictions.
  2. **Algorithm** makes it's prediction.
  3. **Nature** Chooses label/outcome.
  4. **Loss** is associated with each action according to a loss function.
- Actions interaction (1 round):

Experts vs. actions

- Experts interaction (1 round):
  1. **Experts** make their predictions.
  2. **Algorithm** makes it's prediction.
  3. **Nature** Chooses label/outcome.
  4. **Loss** is associated with each action according to a loss function.
- Actions interaction (1 round):
  1. **Algorithm** a distribution over the actions.

Experts vs. actions

- Experts interaction (1 round):
  1. **Experts** make their predictions.
  2. **Algorithm** makes it's prediction.
  3. **Nature** Chooses label/outcome.
  4. **Loss** is associated with each action according to a loss function.
- Actions interaction (1 round):
  1. **Algorithm** a distribution over the actions.
  2. **Nature** reveals the loss of each action.

Experts vs. actions

- Experts interaction (1 round):
  1. **Experts** make their predictions.
  2. **Algorithm** makes it's prediction.
  3. **Nature** Chooses label/outcome.
  4. **Loss** is associated with each action according to a loss function.
- Actions interaction (1 round):
  1. **Algorithm** a distribution over the actions.
  2. **Nature** reveals the loss of each action.
  3. **Loss** of algorithm is expected loss wrt to chosen distribution.

Hedging vs. Halving / pros and cons

- **Expert framework assumes more structure:** algorithm gets expert predicions before choosing it's own prediction. The set of loss vectors is restricted - better for algorithm. Experts framework achieves better bounds.

Hedging vs. Halving / pros and cons

- **Expert framework assumes more structure:** algorithm gets expert predicions before choosing it's own prediction. The set of loss vectors is restricted - better for algorithm. Experts framework achieves better bounds.
- **Actions framework assumes bounded losses:** In exprets framework, we can deal with some unbounded loss functions.

Hedging vs. Halving / pros and cons

- **Expert framework assumes more structure:** algorithm gets expert predicions before choosing it's own prediction. The set of loss vectors is restricted - better for algorithm. Experts framework achieves better bounds.
- **Actions framework assumes bounded losses:** In exprets framework, we can deal with some unbounded loss functions.
- **Partial visibility:** In experts framework, the outcome defines *all* of the losses. In actions framework you might not know the loss of all of the actions (the *multiple-arm-bandit problem*).

Hedging vs. Halving / pros and cons

- **Expert framework assumes more structure:** algorithm gets expert predicions before choosing it's own prediction. The set of loss vectors is restricted - better for algorithm. Experts framework achieves better bounds.
- **Actions framework assumes bounded losses:** In exprets framework, we can deal with some unbounded loss functions.
- **Partial visibility:** In experts framework, the outcome defines *all* of the losses. In actions framework you might not know the loss of all of the actions (the *multiple-arm-bandit problem*).
- **Actions framework is simple and general**: We can usually get bounds in the experts framework using algorithm designed for the actions framework, but the constants can be worse.

Hedging vs. Halving / pros and cons

- **Expert framework assumes more structure:** algorithm gets expert predicions before choosing it's own prediction. The set of loss vectors is restricted - better for algorithm. Experts framework achieves better bounds.
- **Actions framework assumes bounded losses:** In exprets framework, we can deal with some unbounded loss functions.
- **Partial visibility:** In experts framework, the outcome defines *all* of the losses. In actions framework you might not know the loss of all of the actions (the *multiple-arm-bandit problem*).
- **Actions framework is simple and general**: We can usually get bounds in the experts framework using algorithm designed for the actions framework, but the constants can be worse.

Hedging vs. Halving / pros and cons

- **Expert framework assumes more structure:** algorithm gets expert predicions before choosing it's own prediction. The set of loss vectors is restricted - better for algorithm. Experts framework achieves better bounds.
- **Actions framework assumes bounded losses:** In exprets framework, we can deal with some unbounded loss functions.
- **Partial visibility:** In experts framework, the outcome defines *all* of the losses. In actions framework you might not know the loss of all of the actions (the *multiple-arm-bandit problem*).
- **Actions framework is simple and general**: We can usually get bounds in the experts framework using algorithm designed for the actions framework, but the constants can be worse.

Hedging vs. Halving

- Like halving - we want to zoom into best action.

Hedging vs. Halving

- Like halving - we want to zoom into best action.
- Unlike halving - no action is perfect.

Hedging vs. Halving

- Like halving - we want to zoom into best action.
- Unlike halving - no action is perfect.
- Basic idea - reduce probability of lossy actions, but not all the way to zero.

Hedging vs. Halving

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

$$\sum_{t=1}^{T} \mathbf{p}^t \cdot \ell^t - \min_i \left(\sum_{t=1}^{T} \ell_i^t \right)$$

Using Hedge to generalize halving alg.

- Suppose that there is no perfect action.

Using Hedge to generalize halving alg.

- Suppose that there is no perfect action.
- actions: i for $i \in (1, 2, \ldots, N)$

Using Hedge to generalize halving alg.

- Suppose that there is no perfect action.
- actions: i for $i \in (1, 2, \ldots, N)$
- Now each iteration $t \in \{1, \ldots, T\}$ of game consists of two steps:

Using Hedge to generalize halving alg.

- Suppose that there is no perfect action.
- actions: i for $i \in (1, 2, \ldots, N)$
- Now each iteration $t \in \{1, \ldots, T\}$ of game consists of two steps:
  - Algorithm chooses a distribution $\mathbf{p}^t = (p_1^t, \ldots, p_N^t)$ over the actions.

Using Hedge to generalize halving alg.

- Suppose that there is no perfect action.
- actions: i for $i \in (1, 2, \ldots, N)$
- Now each iteration $t \in \{1, \ldots, T\}$ of game consists of two steps:
  - Algorithm chooses a distribution $\mathbf{p}^t = (p_1^t, \ldots, p_N^t)$ over the actions.
  - Nature chooses the loss of each action: $\ell^t = (\ell_1^t, \ldots, \ell_N^t)$, $\ell_i^t \in [0, 1]$

Using Hedge to generalize halving alg.

- Suppose that there is no perfect action.
- actions: i for $i \in (1, 2, \ldots, N)$
- Now each iteration $t \in \{1, \ldots, T\}$ of game consists of two steps:
  - Algorithm chooses a distribution $\mathbf{p}^t = (p_1^t, \ldots, p_N^t)$ over the actions.
  - Nature chooses the loss of each action: $\ell^t = (\ell_1^t, \ldots, \ell_N^t)$, $\ell_i^t \in [0, 1]$
- Algorithm's cumulative loss is $L_A^T = \sum_{t=1}^T \mathbf{p}^t \cdot \ell^t$

Using Hedge to generalize halving alg.

- Suppose that there is no perfect action.
- actions: i for $i \in (1, 2, \ldots, N)$
- Now each iteration $t \in \{1, \ldots, T\}$ of game consists of two steps:
  - Algorithm chooses a distribution $\mathbf{p}^t = (p_1^t, \ldots, p_N^t)$ over the actions.
  - Nature chooses the loss of each action: $\ell^t = (\ell_1^t, \ldots, \ell_N^t)$, $\ell_i^t \in [0, 1]$
- Algorithm's cumulative loss is $L_A^T = \sum_{t=1}^T \mathbf{p}^t \cdot \ell^t$
- Cumulative loss of action i is $L_i^t = \sum_{t=1}^T \ell_i^t$

Using Hedge to generalize halving alg.

- Suppose that there is no perfect action.
- actions: i for $i \in (1, 2, \ldots, N)$
- Now each iteration $t \in \{1, \ldots, T\}$ of game consists of two steps:
  - Algorithm chooses a distribution $\mathbf{p}^t = (p_1^t, \ldots, p_N^t)$ over the actions.
  - Nature chooses the loss of each action: $\ell^t = (\ell_1^t, \ldots, \ell_N^t)$, $\ell_i^t \in [0, 1]$
- Algorithm's cumulative loss is $L_A^T = \sum_{t=1}^T \mathbf{p}^t \cdot \ell^t$
- Cumulative loss of action i is $L_i^t = \sum_{t=1}^T \ell_i^t$
- Our goal is to minimize the **regret:** $L_A^t - \min_i L_i^t$ for all t and all possible loss sequences.

The **Hedge**(η)Algorithm

Consider action i at time t

- Total loss:

$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$

The **Hedge**(η)Algorithm

Consider action i at time t

- Total loss:

$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$

- Weight:

$$w_i^t = e^{-\eta L_i^t}$$

The **Hedge**(η)Algorithm

Consider action i at time t

- Total loss:

$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$

- Weight:

$$w_i^t = e^{-\eta L_i^t}$$

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

The **Hedge**(η)Algorithm

Consider action i at time t

- Total loss:

$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$

- Weight:

$$w_i^t = e^{-\eta L_i^t}$$

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

$$p_i^t = \frac{w_i^t}{\sum_{j=1}^N w_j^t},$$

The **Hedge**(η)Algorithm

Consider action i at time t

- Total loss:

$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$

- Weight:

$$w_i^t = e^{-\eta L_i^t}$$

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

$$p_i^t = \frac{w_i^t}{\sum_{j=1}^N w_j^t},$$

The Hedge(η)Algorithm

Consider action i at time t

- Total loss:

$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$

- Weight:

$$w_i^t = e^{-\eta L_i^t}$$

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

$$p_i^t = \frac{w_i^t}{\sum_{j=1}^N w_j^t}, \quad \mathbf{p}^t = \frac{\mathbf{w}^t}{\sum_{j=1}^N w_i^t}$$

Bound on the loss of **Hedge**(η)Algorithm

Bound on the loss of **Hedge**(η)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*

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

Bound on the loss of **Hedge**(η)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*

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

  - Proof: by combining upper and lower bounds on $\sum_{i=1}^{N} w_i^{T+1}$

Upper bound 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*

$$\ln \left(\sum_{i=1}^{N} w_i^{T+1} \right) \leq -(1 - e^{-\eta}) L_{\textbf{Hedge}(\eta)}.$$

Proof of upper bound (slide 1)

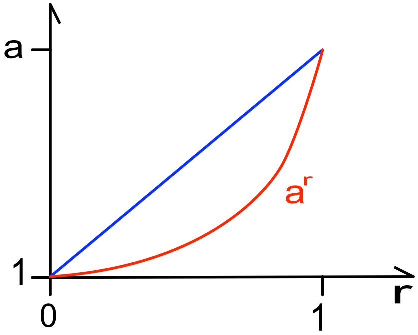
- If $a \geq 0$ then a^r is convex.

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 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 $a = e^{-\eta}, r = \ell_i^t$

$$\sum_{i=1}^{N} w_i^{t+1} = \sum_{i=1}^{N} w_i^t e^{-\eta \ell_i^t}$$

Proof of upper bound (slide 2)

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

$$\begin{aligned}\sum_{i=1}^{N} w_i^{t+1} &= \sum_{i=1}^{N} w_i^t e^{-\eta \ell_i^t} \\ &\leq \sum_{i=1}^{N} w_i^t \left(1-(1-e^{-\eta})\ell_i^t\right)\end{aligned}$$

Proof of upper bound (slide 2)

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

$$\begin{aligned}
\sum_{i=1}^N w_i^{t+1} &= \sum_{i=1}^N w_i^t e^{-\eta \ell_i^t} \\
&\leq \sum_{i=1}^N w_i^t \left(1-(1-e^{-\eta})\ell_i^t\right) \\
&= \left(\sum_{i=1}^N w_i^t\right)\left(1-(1-e^{-\eta})\frac{\mathbf{w}^t}{\sum_{i=1}^N w_i^t}\cdot \ell^t\right)
\end{aligned}$$

Proof of upper bound (slide 2)

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

$$
\begin{aligned}
\sum_{i=1}^N w_i^{t+1} &= \sum_{i=1}^N w_i^t e^{-\eta \ell_i^t} \\
&\leq \sum_{i=1}^N w_i^t \left(1 - (1 - e^{-\eta})\ell_i^t\right) \\
&= \left(\sum_{i=1}^N w_i^t\right) \left(1 - (1 - e^{-\eta}) \frac{\mathbf{w}^t}{\sum_{i=1}^N w_i^t} \cdot \ell^t\right) \\
&= \left(\sum_{i=1}^N w_i^t\right) \left(1 - (1 - e^{-\eta}) \mathbf{p}^t \cdot \ell^t\right)
\end{aligned}
$$

Proof of upper bound (slide 3)

- Combining

$$\sum_{i=1}^{N} w_i^{t+1} \leq \left(\sum_{i=1}^{N} w_i^{t}\right)\left(1-(1-e^{-\eta})\mathbf{p}^t \cdot \ell^t\right)$$

Proof of upper bound (slide 3)

- Combining

$$\sum_{i=1}^{N} w_i^{t+1} \leq \left(\sum_{i=1}^{N} w_i^{t}\right) \left(1 - (1 - e^{-\eta})\mathbf{p}^t \cdot \ell^t\right)$$

- for $t = 1, \ldots, T$

Proof of upper bound (slide 3)

- Combining

$$\sum_{i=1}^{N} w_i^{t+1} \leq \left(\sum_{i=1}^{N} w_i^t\right)\left(1-(1-e^{-\eta})\mathbf{p}^t\cdot\ell^t\right)$$

- for $t = 1, \ldots, T$
- yields

$$\sum_{i=1}^{N} w_i^{T+1} \quad \leq \quad N\prod_{t=1}^{T}\left(1-(1-e^{-\eta})\mathbf{p}^t\cdot\ell^t\right)$$

Proof of upper bound (slide 3)

- Combining

$$\sum_{i=1}^{N} w_i^{t+1} \leq \left(\sum_{i=1}^{N} w_i^t\right) (1 - (1 - e^{-\eta})\mathbf{p}^t \cdot \ell^t)$$

- for $t = 1, \ldots, T$
- yields

$$\sum_{i=1}^{N} w_i^{T+1} \quad \leq \quad N \prod_{t=1}^{T} (1 - (1 - e^{-\eta})\mathbf{p}^t \cdot \ell^t)$$

Proof of upper bound (slide 3)

- Combining

$$\sum_{i=1}^N w_i^{t+1} \leq \left(\sum_{i=1}^N w_i^t\right)(1-(1-e^{-\eta})\mathbf{p}^t\cdot\ell^t)$$

- for $t = 1, \ldots, T$
- yields

$$\begin{aligned}\sum_{i=1}^N w_i^{T+1} &\leq N\prod_{t=1}^T(1-(1-e^{-\eta})\mathbf{p}^t\cdot\ell^t) \\ &\leq \exp\left(\ln N-(1-e^{-\eta})\sum_{t=1}^T\mathbf{p}^t\cdot\ell^t\right)\end{aligned}$$

since $1+x\leq e^x$ for $x = -(1-e^{-\eta})$.

Lower bound on $\sum_{i=1}^{N} w_i^{T+1}$

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

$$\sum_{i=1}^{N} w_i^{T+1} \geq w_j^{T+1} = e^{-\eta L_j}$$

Combining Upper and Lower bounds

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

$$-\eta L_j \le \ln \sum_{i=1}^{N} w_i^{T+1} \le \ln N - (1 - e^{-\eta}) \sum_{t=1}^{T} \mathbf{p}^t \cdot \ell^t$$

Combining Upper and Lower bounds

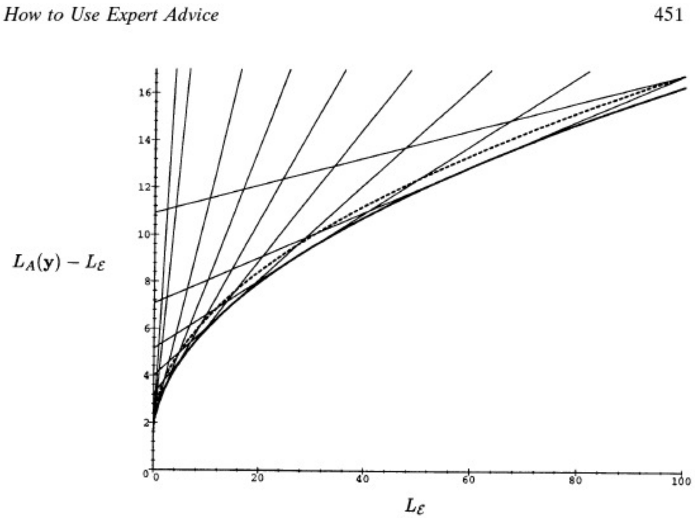
- Combining bounds on $\ln\left(\sum_{i=1}^{N} w_i^{T+1}\right)$

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

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

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

Tuning η


How to Use Expert Advice
451
$L_A(\mathbf{y}) - L_\mathcal{E}$
16
14
12
10
8
6
4
2
0
0
20
40
60
80
100
$L_\mathcal{E}$

Tuning η

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

Tuning η

- Suppose $\min_i L_i \leq \tilde{L}$
- set
$$\eta = \ln\left(1 + \sqrt{\frac{2\ln N}{\tilde{L}}}\right) \approx \sqrt{\frac{2\ln N}{\tilde{L}}}$$

Tuning η

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

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

- Then

$$L_{\textbf{Hedge}(\eta)} \leq \frac{-\ln(w_i^1) + \eta L_i}{1 - e^{-\eta}} \leq \min_i L_i + \sqrt{2\tilde{L}\ln N} + \ln N$$

Tuning η as a function of T

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

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

Tuning η as a function of T

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

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

- per iteration we get:

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

How good is this bound?

- Very good! There is a closely matching lower bound!

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

$$L_{\mathbf{S}} - \min_i L_i \geq (1 - o(1))\sqrt{2T \ln N}$$

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

$$L_{\mathbf{S}} - \min_i L_i \geq (1 - o(1))\sqrt{2T \ln N}$$

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

The adversarial strategy

- Adversary sets each loss ℓ_i^t indepedently at random to 0 or 1 with equal probabilities $(1/2, 1/2)$.

The adversarial strategy

- Adversary sets each loss ℓ_i^t indepedently at random to 0 or 1 with equal probabilities $(1/2, 1/2)$.
- Obviously, nothing to learn !
  $L_{\mathbf{S}} \approx T/2$.

The adversarial strategy

- Adversary sets each loss ℓ_i^t indepedently at random to 0 or 1 with equal probabilities $(1/2, 1/2)$.
- Obviously, nothing to learn !
  $L_{\mathbf{S}} \approx T/2$.
- On the other hand $\min_i L_i \approx T/2 - \sqrt{2T \ln N}$

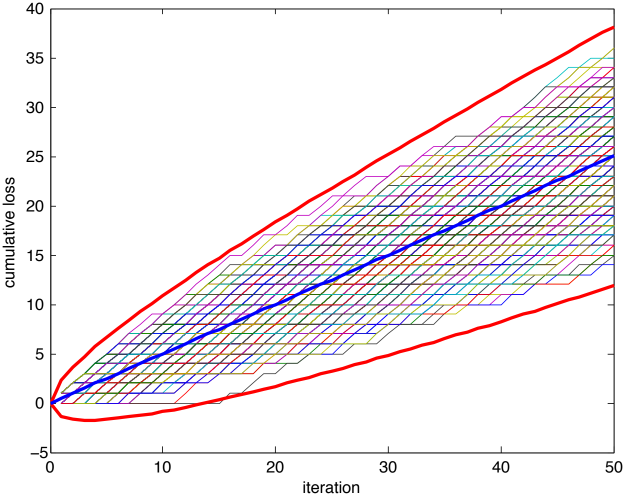
The adversarial strategy

- Adversary sets each loss ℓ_i^t indepedently at random to 0 or 1 with equal probabilities $(1/2, 1/2)$.
- Obviously, nothing to learn !
  $L_{\mathbf{S}} \approx T/2$.
- On the other hand $\min_i L_i \approx T/2 - \sqrt{2T \ln N}$
- The difference $L_{\mathbf{S}} - \min_i L_i$ is due to unlearnable random fluctuations!

The adversarial strategy

- Adversary sets each loss ℓ_i^t indepedently at random to 0 or 1 with equal probabilities $(1/2, 1/2)$.
- Obviously, nothing to learn !
  $L_{\mathbf{S}} \approx T/2$.
- On the other hand $\min_i L_i \approx T/2 - \sqrt{2T \ln N}$
- The difference $L_{\mathbf{S}} - \min_i L_i$ is due to unlearnable random fluctuations!
- Detailed proof in "Adaptive Game playing using multiplicative weights" Freund and Schapire.

The adversarial construction



Summary

- Given learning rate η the **Hedge**(η)algorithm satisfies

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

Summary

- Given learning rate η the **Hedge**(η)algorithm satisfies

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

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

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

Summary

- Given learning rate η the **Hedge**(η)algorithm satisfies

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

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

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

- A trivial random data, for which there is nothing to be learned forces any algorithm to suffer this total regret.