Online Optimization

Axel Böhm

November 14, 2021

What is online Learning

Consider the game: In each round $t = 1, \dots, T$

- an *adversary* chooses a (secret) number in $y_t \in [0,1]$
- you guess the real number, choosing $x_t \in [0,1]$;
- you pay the squared difference $(x_t - y_t)^2$.

Task: guess a sequence of numbers as precisely as possible.
Question: How to measure success?

Adversary plays i.i.d.

- Adversary numbers are drawn from a fixed distribution (with mean μ and Variance σ^2).
- If we knew the distribution, we could pick the mean and pay in expectation $\sigma^2 T$ (optimal).
- Benchmark against best possible strategy:

$$\mathbb{E}_Y\left[\sum_{t=1}^{T}(x_t - Y)^2\right] - \sigma^2 T,$$

or equivalently considering the average

$$\frac{1}{T}\mathbb{E}_Y\left[\sum_{t=1}^{T}(x_t - Y)^2\right] - \sigma^2 \ .$$

Last quantity should go to zero.

Minimizing Regret

Let's rewrite a bit more general

$$\mathbb{E}\left[\sum_{t=1}^{T}(x_t - Y)^2\right] - \min_{x} \mathbb{E}\left[\sum_{t=1}^{T}(x - Y)^2\right] .$$

($\sigma^2 T$ was just the payoff of the best possible strategy)
Finally:

- remove the assumption on how the data is generated,
- consider any arbitrary sequence of y_t (can remove the expectation).

$$R_T := \sum_{t=1}^{T}(x_t - y_t)^2 - \min_{x\in[0,1]} \sum_{t=1}^{T}(x - y_t)^2$$

Called **Regret**, because it measures how much the algorithm regrets for not sticking on all the rounds to the optimal choice in hindsight.

General loss functions

Online Learning is the study of algorithms to **minimize the regret over a sequence of loss functions** w.r.t. a competitor $u \in \mathbb{R}^d$:

$$R_T(u) := \sum_{t=1}^{T} \ell_t(x_t) - \sum_{t=1}^{T} \ell_t(u) \ .$$

This framework allows to

- reformulate problems in ML and optimization as similar games.
- analyze situations in which the data are not i.i.d. yet want to guarantee that the algorithm is "learning" something.

For example, online learning can be used to analyze

- Click prediction problems;
- spam filter;
- Convergence to equilibrium of *repeated* games.

It can *also* be used to analyze stochastic algorithms, e.g., SGD.

Back to the numbers game

Ideally we pick

$$x_T^\star := \arg\min_x \sum_{t=1}^{T} (x - y_t)^2 = \frac{1}{T} \sum_{t=1}^{T} y_t.$$

- Don't know the future: x_T^* is not an option
- do know the past. in each round: best number **in hindsight**
- not because we expect the future to be like the past (not true)
- optimal guess should not change too much between rounds

Follow-the-Leader (FTL)

Hence, on each round t our strategy is to guess

$$x_t = x_{t-1}^\star = \frac{1}{t-1} \sum_{i=1}^{t-1} y_i.$$

Follow the leader works for the numbers game

Theorem

Let $y_t \in [0,1]$ for $t = 1, \ldots, T$ an arbitrary sequence of numbers. Let the algorithm's output $x_t = x^\star_{t-1} := \frac{1}{t-1}\sum_{i=1}^{t-1} y_i$. Then,

$$R_T = \sum_{t=1}^{T}(x_t - y_t)^2 - \min_{x\in[0,1]} \sum_{t=1}^{T}(x - y_t)^2 \leq 4 + 4\ln T \ .$$

Failure of FTL

Let $V = [-1, 1]$ and consider the sequence of losses $\ell_t(x) = z_t x$, where $z_1 = -0.5$

$$z_t = \begin{cases} 1, & t \text{ even} \\ -1, & t \text{ odd} \end{cases}$$

- Predictions of FTL will be $x_t = 1$ for t even and
- $x_t = -1$ for t odd.
- Cumulative loss of the FTL algorithm will be T, while fixed solution $u = 0$ gives 0.
- regret of FTL is T.

Weighted majority algorithm

Consider the **learning from experts** scenario. Experts $= 1, \ldots, n$.
Decision: "Yes" or "No".

$$f_t(x_t) = \begin{cases} 1 & \text{if wrong} \\ 0 & \textit{otherwise} \end{cases}$$

(i) weights $w_1(i) = 1$ for all $i = 1, \ldots, n$, pick $\alpha > 0$

(ii) for $t = 1, \ldots, T$

1. compare weights $\sum_{i \in YES} w_t(i)$ vs. $\sum_{i \in NO} w_t(i)$
2. choose Yes or No depending on above comparison
3. observe feedback
4. update weights:

$$w_{t+1}(i) = \begin{cases} w_t(i) & \text{if Expert } i \text{ was right} \\ (1-\alpha) w_t(i) & \text{if Expert } i \text{ made a mistake} \end{cases}$$

Weighted majority algorithm II

Let

- M_t be the number mistakes we make after t attempts and
- $m_t(i)$ the number of mistakes expert i made (until t).

Theorem

Then,

$$M_T \leq 2(1+\alpha)m_T(i) + 2\frac{\log(n)}{\alpha}$$

Regret is

$$M_T - m_T(i^*) = R_T.$$

Approximately: Can bound our mistakes by 2 times number of mistakes of best expert.

Proof of the Theorem

We always have $\|w_{t+1}\|_1 \leq \|w_t\|_1$.
If we made a mistake, then

$$\|w_{t+1}\|_1 \leq \frac{1}{2}\|w_t\|_1 + \frac{1}{2}\|w_t\|_1(1-\alpha)$$
$$= \|w_t\|_1(1-\alpha/2).$$

Therefore

$$\|w_{t+1}\|_1 \leq \|w_1\|_1(1-\alpha/2)^{M_t} = n(1-\alpha/2)^{M_t}.$$

The weight of i-th expert can be bounded by

$$w_{t+1}(i) = (1-\alpha)^{m_t(i)} \leq \|w_{t+1}\|_1$$

Combining the above two yields

$$(1-\alpha)^{m_t(i)} \leq n(1-\alpha/2)^{M_t}$$

and

$$m_t(i)\log(1-\alpha) \leq \log(n) + M_T \log(1-\alpha/2).$$

remainder of the proof

Use the fact that for $x \in (0, \frac{1}{2})$

$$-x - x^2 \leq \log(1 - x) \leq -x$$

to deduce that

$$-m_t(i)(\alpha + \alpha^2) \leq \log(n) - M_T \frac{\alpha}{2}$$
$$-2m_t(i)(1 + \alpha) \leq \frac{2}{\alpha} \log(n) - M_T$$

which yields

$$M_T - \leq 2m_t(i)(1 + \alpha) + \frac{2}{\alpha} \log(n).$$

Randomized Weighted Majority

Instead of picking the optinion of the (weighted) majority, we only do so with a **probability**.

(i) $w_1(i) = 1$ for all $i = 1, \ldots, n$ and $\alpha \in (0, 1)$

(ii) for $t = 1, \ldots, T$

1. compute $p_t(i) = w_t(i)/\|w_t\|_1$
2. choose expert i with probability $p_t(i)$
3. observe feedback
4. update weights:

$$w_{t+1}(i) = \begin{cases} w_t(i) & \text{if expert } i \text{ was right} \\ (1-\alpha)w_t(i) & \text{if expert } i \text{ made a mistake} \end{cases}$$

Comment: Randomizing algorithms typically improves the (worst case) analysis.

Randomized Weighted Majority

Instead of picking the optinion of the (weighted) majority, we only do so with a **probability**.

(i) $w_1(i) = 1$ for all $i = 1, \dots, n$ and $\alpha \in (0, 1)$

(ii) for $t = 1, \dots, T$

1. compute $p_t(i) = w_t(i)/\|w_t\|_1$
2. choose expert i with probability $p_t(i)$
3. observe feedback
4. update weights:

$$w_{t+1}(i) = \begin{cases} w_t(i) & \text{if expert } i \text{ was right} \\ (1-\alpha)w_t(i) & \text{if expert } i \text{ made a mistake} \end{cases}$$

Comment: Randomizing algorithms typically improves the (worst case) analysis.

Randomized Weighted Majority contd.

As before:

- $M_t = \#$ of mistakes we make after t attempts
- $m_t(i) = \#$ of mistakes expert i made (until t)

Theorem

$$\mathbb{E}[M_T] \leq (1+\alpha)m_T(i) + \frac{\log(n)}{\alpha}$$

Improved constants!

Multiplicative Weights Algorithm

Before: Loss ℓ_t was 0 or 1.

Now: General loss functions $\boldsymbol{\ell}_t = (\ell_t(1), \ldots, \ell_t(n))$ with $\ell_t(i) \in [-1, 1]$

(i) $w_1(i) = 1$ for all $i = 1, \ldots, n$ and $\alpha \in (0, 1)$

(ii) for $t = 1, \ldots, T$

1. compute $p_t(i) = w_t(i)/\|w_t\|_1$
2. choose expert i with probability $p_t(i)$
3. observe loss $\boldsymbol{\ell}_t = (\ell_t(1), \ldots, \ell_t(n))$
4. update weights:

$$w_{t+1}(i) = (1 - \alpha\ell_t(i))w_t(i)$$

Note that

$$\langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle = p_t(1)\ell_t(1) + \cdots + p_t(n)\ell_t(n) = \mathbb{E}_i[\ell_t(i)]$$

gives expected loss of round t.

Multiplicative Weights Algorithm [contd]

Theorem

if $\ell_t(i) \in [-1, 1]$ *and* $\alpha < \frac{1}{2}$*, then* ***MWA*** *guarantees*

$$\sum_{t=1}^{T} \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle - \sum_{t=1}^{T} \ell_t(i) \leq \alpha \sum_{t=1}^{T} |\ell_t(i)| + \frac{\log(n)}{\alpha} \quad \forall i$$

Hedge Algorithm

(i) $w_1(i) = 1$ for all $i = 1, \ldots, n$ and $\alpha \in (0, 1)$

(ii) for $t = 1, \ldots, T$

1. compute $p_t(i) = w_t(i)/\|w_t\|_1$
2. choose expert i with probability $p_t(i)$
3. observe loss $\boldsymbol{\ell}_t = (\ell_t(1), \ldots, \ell_t(n))$
4. update weights:

$$w_{t+1}(i) = w_t(i) e^{-\alpha \ell_t(i)}$$

Note:

$$e^{-x} \approx 1 - x$$

Hedge Algorithm [contd]

Theorem

If $\ell_t(i) \in [-1, 1]$ *and* $\alpha < \frac{1}{2}$, *then* ***Hedge Alg.*** *guarantees*

$$\sum_{t=1}^{T} \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle - \sum_{t=1}^{T} \ell_t(i) \leq \alpha \sum_{t=1}^{T} \langle p_t, \ell_t^2 \rangle + \frac{\log(n)}{\alpha} \quad \forall i.$$

Further improved constants.

Observe: Iteration t is just

$$w_{t+1}(i) = w_t(i) e^{-\alpha \ell_t(i)}$$

$$p_{t+1}(i) = \frac{w_{t+1}(i)}{\|w_{t+1}\|_1}$$

Online mirror descent! (KL-divergence setting:)

$$h(x) = \sum_i x(i) \log(x(i))$$

Hedge Algorithm [contd]

Theorem

If $\ell_t(i) \in [-1,1]$ and $\alpha < \frac{1}{2}$, then ***Hedge Alg.*** *guarantees*

$$\sum_{t=1}^{T} \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle - \sum_{t=1}^{T} \ell_t(i) \le \alpha \sum_{t=1}^{T} \langle p_t, \ell_t^2 \rangle + \frac{\log(n)}{\alpha} \quad \forall i.$$

Further improved constants.

Observe: Iteration t is just

$$w_{t+1}(i) = w_t(i) e^{-\alpha \ell_t(i)}$$

$$p_{t+1}(i) = \frac{w_{t+1}(i)}{\|w_{t+1}\|_1}$$

Online mirror descent! (KL-divergence setting:)

$$h(x) = \sum_i x(i) \log(x(i))$$

Online mirror descent

$$x_{t+1} = \arg\min_{x \in K} \{ \langle \nabla f_t(x_t), x \rangle + D_h(x, x_k) \}.$$

Gives bound

$$\sum_{t=1}^{T} f_t(x_t) - \sum_{t=1}^{T} f_t(x) \leq \frac{D_h(x, x_0)}{\alpha} + \frac{\alpha T G^2}{2}$$