Advanced Topics in Machine Learning

Kartik Srinivas

September 13, 2023

Important Remarks

- The Notes often use *Einstein Notation* with the iterating variable usually denoted as i

---

1 Lecture - 1

Algorithm 1 OF Framework

```
1: for $t = 1 \to T$ do
2:     Algorithm $A$ chooses $p_t \in D_t$
3:     $y_t \leftarrow f(p_t)$
4:     $l_t \leftarrow l(y_t, a_t) \quad l \in \mathcal{L}$        ▷ I must do well on every single $l$
5: end for
```

Applications may include weather filetring, spam filtering and stoick prediction. In spam filtering the objective function

In stock prediction, the reward can be considered to be some r_t (decided by the adversary) and the weight vector is that of the distribution of money in the stocks The loss will be negative of the sum of inner products.

$$l = -\sum_t \langle p_t, r_t \rangle \tag{1}$$

Algorithm 2 Learning with Expert advice

```
Require: $N$ experts
1: for $t = 1 \to T$ do
2:     Create expert predictions $D_t = \{f_{i,t}\}_{i=1}^{i=n}$
3:     Algorithm $A$ chooses $p_t \in D_t$
4:     Adversary chooses $y_t$ and $l_t$
5:     $l_t \leftarrow l(y_t, a_t) \quad l \in \mathcal{L}$        ▷ I must do well on every single $l$
6: end for
```

Regret

The correct expression for regret of an algorithmic $\mathcal{A}$ that takes in the prediction history $p_{1:t-1}$ and outputs a **specific** $p_t \in D_t$ is given by

$$R_T(\mathcal{A}) = \sup_{l_i, y_i \in \mathcal{L}, \mathcal{Y}} \{\sum_T l_t(y_t, p_t = \mathcal{A}(p_1, p_2, \dots p_{t-1})) - \min_{p_t^* \in D_t} \sum_T l_t(y_t, p_t^*)\} \quad (2)$$

Say that the problem is Binary output i.e the p_t and y_t are both binary output (space $\mathcal{Y} = 0, 1$) However note that this has nothing to do with the D_t, which is a set whose every term belongs to the output space $\mathcal{Y}$. The loss function is also fixed at 0,1 loss, the only thing the adversary can pick now is the output y_t. Let us also assume that there is a **perfect** expert. Which means the RHT in regret is now zero.

This implies that the algorithm $\mathcal{A}$ should ideally mimic the perfect expert's predictions as much as possible in-order to make the regret value become 0. The question, then is to devise an algorithm from the history that can find the perfect expert **as quick as possible**, by cutting down other options in D_t.

Algorithm 3 Majority voting

```
Require: $N$ experts
 1: for $t = 1 \to T$ do
 2:     Create expert predictions $D$
 3:     Algorithm $A$ chooses The Majority $p_t \in D$
 4:     Adversary chooses $y_t$ and $l_t$
 5:     if $l_t(p_t, y_t) = 1$ then
 6:         Throw out the majority
 7:     else
 8:         Keep the majority, no loss incurred
 9:     end if
10: end for
```

The number of experts that are thrown out every mistake are atleast half the total number of experts (because the majority is thrown) But this is an idealization. In reality, there will be no 'best expert'. Instead we need to find the 'okay-ish' expert.

Potential Function

Algorithm 4 Weighted Majority Algorithm

```
  $w_1 \leftarrow 1$
  for $t = 1 \rightarrow T$ do
      $p_t \leftarrow 1\{\sum_i 1_{f_{t,i}=1} w_{t,i} \geq \sum_i 1_{f_{t,i}=0} w_{t,i}\}$
      Adversary chooses $y_t$ and $l_t$
      if $l_t(f_{i,t}, y_t) = 0$ then
          $w_{t+1,i} \leftarrow w_{t,i}$
      else
          $w_{t+1,i} \leftarrow \beta w_{t,i}$                    ▷ $\beta \leq 1$
      end if
      Normalize $w_{t+1}$
  end for
```

Essentially the weights of the mistake-makers are reduced and the weights of the correct ones remain constant The bound you can prove using induction is as follows

Provided the step t is a mistake, then the following holds

Motivation for usage of Φ_t is unclear

$$\Phi_t = \sum_{i=1}^{N} w_{t,i} \qquad \Phi_{t+1} \leq \frac{1+\beta}{2}\Phi_t \tag{3}$$

WLOG, assume that $y_t = 0$ and $p_t = 1$ (works for mistake cases)

$$\begin{aligned}
\Phi_{t+1} &= 1_{f_{i,t}=1} w_{i,t+1} + 1_{f_{i,t}=0} w_{i,t+1} &(4)\\
&= 1_{f_{i,t}=1} \beta w_{i,t} + 1_{f_{i,t}=0} w_{i,t} &(5)\\
&= 1_{f_{i,t}=1} \beta w_{i,t} + \beta 1_{f_{i,t}=0} w_{i,t} + (1-\beta) 1_{f_{i,t}=0} w_{i,t} &(6)\\
&\leq \beta\Phi_t + (1-\beta) 1_{f_{i,t}=0} \Phi_t/2 &(7)\\
&\leq \frac{1+\beta}{2}\Phi_t &(8)
\end{aligned}$$

For a more rigorous proof with general y_t:-

$$\begin{aligned}
\Phi_{t+1} &= 1_{f_{i,t}\neq y_t} w_{i,t+1} + 1_{f_{i,t}=y_t} w_{i,t+1} &(9)\\
&= 1_{f_{i,t}\neq y_t} \beta w_{i,t} + 1_{f_{i,t}=y_t} w_{i,t} &(10)\\
&= \beta\Phi_t + (1-\beta) 1_{f_{i,t}=y_t} w_{i,t} &(11)\\
&\leq \beta\Phi_t + (1-\beta)\phi_t/2 \quad \leq \frac{1+\beta}{2}\Phi_t &(12)
\end{aligned}$$

Let the number of mistakes of the i'th expert be called $M_{i,t}$. Let the number of mistakes for the generic algorithm be called M_T. We need to find a bound on the regret

Let

$$i^* = \text{argmin} \quad M_{i,t}$$

.

The appropriate format for regret $= M_T - M_{i^*,T}$. Which was the expert that I should have listened to (essentially).

You can note that M_T is a generic algorithm, that does not listen to any of the experts. Essentially it tries to solve the problem with any possible adversary and gives a 'worst' case bound

It is possible that our strategy for choosing p_t can also be purely randomized, in this case the regret should relate to the average number of errors made by the algorithm, as compared to the **best average expert**.

Algorithm 5 Randomized Weighted Majority Algorithm

```
$w_1 \leftarrow 1$
for $t = 1 \rightarrow T$ do

    $p_t = \mathrm{Ber}(\frac{1_{f_{i,t} \neq y_t} w_{i,t-1}}{w_{t-1,i}})$ ▷ The probability of choosing any class $y$ as given
by the previous step

    Adversary chooses $y_t$ and $l_t$
    if $l_t(f_{i,t}, y_t) = 0$ then
        $w_{t+1,i} \leftarrow w_{t,i}$
    else
        $w_{t+1,i} \leftarrow \beta w_{t,i}$                                ▷ $\beta \leq 1$
    end if
    Normalize $w_{t+1}$
end for
```

$$\mathbb{E}[\Phi_{t+1}] \leq (1 - (1 - \beta)b_t)\mathbb{E}[\Phi_t] \tag{13}$$

The term $\sum b_t$ is roughly like the number of "mistakes" made by the algorithm, somewhat like the average weight that is being put into the wrong class. In the case where there was no randomization, atleast half the weight was put into the wrong class whenever there was a mistake.