CS787: Advanced Algorithms	**Scribe:** Mikhail Nedbai
Lecture 17: Online Learning	**Date:** 04/02/2019

17.1 Problem statement and discussion

The Online Binary Prediction problem is defined as follows :
Given:

- n exprets indexed by $i \in [n]$
- T days indexed by $t \in [T]$
- Expert i makes prediction $x_{i,t} \in \{0, 1\}$ for every day $t \in [T]$
- $\forall t$,ALG makes prediction x_t on day t
- Outcome $y_t \in \{0, 1\}$ is revealed
- The cost incurred by each expert on day t : $c_{i,t} = \mathbb{1}\{x_{i,t} = y_t\}$
- The cost incurred by algorithm on day t : $c_t = \mathbb{1}\{x_t = y_t\}$

Goal :

- $M = \sum_{t=1}^{T} c_t$
- $m = \min_{i \in [n]} \sum_{t=1}^{T} x_{i,t}$
- Objective : minimize M

For this lecture we focus on minimizing M with relation to m, or in other words we want the predictions of ALG to be no much worse than the predictions of the *best performing expert.* Even though other strategies may be considered, such as comparing ALG to OPT, or comparing ALG to an algorithm capable of following multiple experts, to utilize their performance at different times, both of these metrics would not be meaningful, since both of these algorithms would vastly outperform any ALG. Although we are still focusing on an online problem, there are some differences, namely we know how many predicstions, exactly T will be made.

17.2 Simple Majority Algorithm

We first analyze a variant of this problem, in which $\exists i \in [n] : \sum_{t=1}^{T} x_{i,t} = 0$. In other words, we assume the existence of an *always-correct expert*. A natural approach in this scenario would be to ignore the predictions of experts, who have made mistakes in the past, and use the majority vote among the remaining experts as the current prediction of ALG. This idea is incapsulated in the Simple-Majority algorithm.
Simple Majority

- Let $w_{i,t}$ denote the weight of expert i at time t
- $\forall i, w_{i,t} := 1$
- At step t,
  - $x_t := majority_i(w_{i,t}x_{i,t})$
  - if $x_{i,t} \neq y_t$ then $w_{i,t+1} := 0$

We denote $W_t = \sum_{i=1}^{n} w_{i,t}$ as the total weight of all experts at time t. We can make the following claims about the Simle Majority algorithm :

1. $W_1 = n$
2. $W_T \geq 1$, since at least one expert makes no mistakes
3. If a mistake is made by alg at time t, $W_{t+1} \leq \frac{1}{2}W_t$

From observation 3 we conclude that each time an algorithm makes a mistake, at least half of the experts make a mistake as well, thus each time ALG is mistaken the total weight within the algorithm is at least halved. Therefore , if the algorithm makes a total of M mistakes, we obtain :

$$W_T \leq \frac{1}{2^M} W_1$$

From these observations we derive :

$$1 \leq W_T \leq \frac{1}{2^M} W_1 \leq \frac{n}{2^M} \tag{17.2.1}$$

$$M \leq log_2(n) \tag{17.2.2}$$

17.3 Weighted Majority Algorithm

Now we remove the condition, that there exists an always-correct expert. In this setting we attempt to construct an algorithm that results in M that is not too much larger than m. We introduce the Weighted Majority Algorithm
Weighted Majority$(0 < \epsilon < 1)$

- $\forall i, w_{i,t} := 1$
- At step t,
  - $x_t := \mathbb{1}\{\sum_{i:x_{i,t}=1} w_{i,t} > \sum_{i:x_{i,t}=0} w_{i,t}\}$
  - $\forall i$, if $x_{i,t} \neq y_t$, then $w_{i,t+1} := (1-\epsilon)w_{i,t}$

The nature of this algorithm implies, that if an expert makes a few mistakes, their weight will not decrease by too much. On the other hand when the algorithm makes a mistake, it must be the case that the majority of experts made a mistake as well, so the total weight of experts will decrease substantialy. To asses the performance of this algorithm we make a few observations

1. $W_1 \leq n$
2. $W_t \geq (1-\epsilon)^m$
3. Every time ALG mispredicts, $W_{t+1} = \sum_{i:x_{i,t}=y_t} w_{i,t} - \sum_{i:x_{i,t}\neq y_t}(1-\epsilon)w_{i,t}$

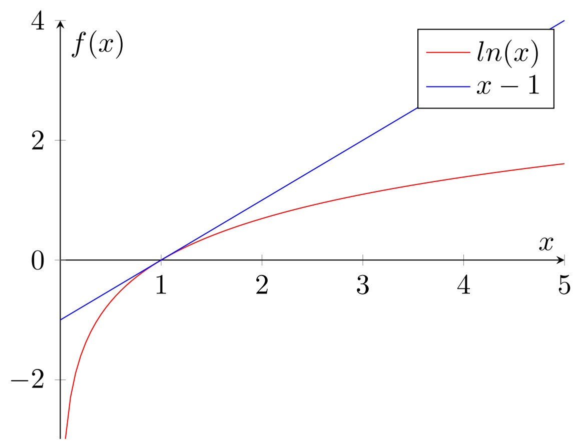
Using these observations we bound performance of Weighted Majority.

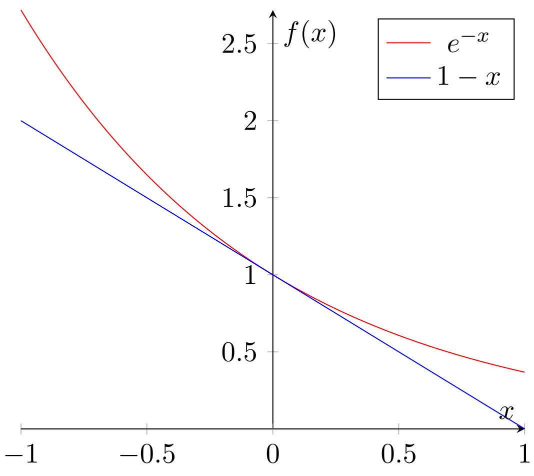
$$W_{t+1} = \sum_{i:x_{i,t}=y_t} w_{i,t} - \sum_{i:x_{i,t}\neq y_t}(1-\epsilon)w_{i,t} \tag{17.3.3}$$
$$= \sum_{i=1}^{n} w_{i,t} - \epsilon \sum_{i:x_{i,t}\neq y_t} w_{i,t} \tag{17.3.4}$$
$$\leq W_t - \epsilon W_t \frac{1}{2} \tag{17.3.5}$$
$$= W_t\Big(1 - \frac{\epsilon}{2}\Big) \tag{17.3.6}$$

Using this information we obtain

$$W_T \leq W_1\Big(1-\frac{\epsilon}{2}\Big)^M \tag{17.3.7}$$
$$(1-\epsilon)^m \leq n\Big(1-\frac{\epsilon}{2}\Big)^M \tag{17.3.8}$$
$$m \cdot log(1-\epsilon) \leq log(n) + M \cdot log\Big(1-\frac{\epsilon}{2}\Big) \tag{17.3.9}$$

We will proceed by observing, that $x - 1$ can be used as an upper bound for $ln(x)$, and e^{-x} can upper bound $1 - x$.





$$\begin{aligned}
log(n) + M \cdot log\Big(1 - \frac{\epsilon}{2}\Big) &\leq log(n) + M\Big(-\frac{\epsilon}{2}\Big) && (17.3.10)\\
m \cdot log(1 - \epsilon) &\leq log(n) + M\Big(-\frac{\epsilon}{2}\Big) && (17.3.11)\\
M\frac{\epsilon}{2} &\leq log(n) + m \cdot log\Big(\frac{1}{1-\epsilon}\Big) && (17.3.12)\\
&\leq log(n) + m\Big(\frac{1}{1-\epsilon} - 1\Big) && (17.3.13)\\
&= log(n) + m\frac{\epsilon}{1-\epsilon} && (17.3.14)\\
M &\leq \frac{2}{1-\epsilon} + \frac{2}{\epsilon}log(n) && (17.3.15)
\end{aligned}$$

So far the two results we have observed are

- Simple Majority : $M \leq log_2(n)$, if $m = 0$
- Weighted Majority : $M \leq \frac{2}{1-\epsilon} + \frac{2}{\epsilon}log(n)$

In the next section we will attempt to come up with an algorithm that achieves the likes of $M \leq m + O(log(n))$.

17.4 Hedge Algorithm

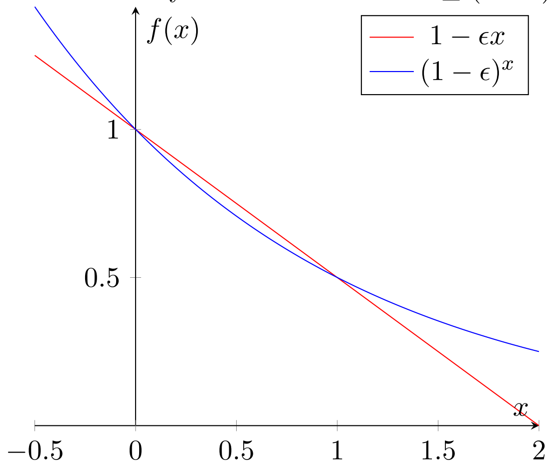
For the hedge algorithm we analyze a more general case of the problem, where the cost incurred by each expert per guess is $c_{i,t} \in [0,1]$.
Hedge$(0 < \epsilon < 1)$

- $\forall i, w_{i,t} := 1$
- At step t,
  - $x_t := x_{i,t}$ with probability $p_{i,t} = \frac{w_{i,t}}{W_t}$
  - $\forall i$, if $x_{i,t} \neq y_t$, then $w_{i,t+1} := (1-\epsilon)^{c_{i,t}} w_{i,t}$

We make several observations to analyze Hedge

1. $W_1 \leq n$
2. $W_T \geq \max_i w_{i,t} \geq (1-\epsilon)^m$
3. $W_{t+1} = \sum_{i=1}^{n} (1-\epsilon)^{c_{i,t}} w_{i,t}$

We additionaly observe that $1 - \epsilon x \geq (1-\epsilon)^x, x \in [0,1]$.



With that

$$E[c_t] = \sum_{i=1}^{n} c_{i,t} p_{i,t} = \frac{1}{W_t} \sum_{i=1}^{n} c_{i,t} w_{i,t} \tag{17.4.16}$$

$$W_{t+1} = \sum_{i=1}^{n} (1-\epsilon)^{c_{i,t}} w_{i,t} \leq \sum_{i=1}^{n} (1 - c_{i,t}\epsilon) w_{i,t} \tag{17.4.17}$$

$$\leq W_t - \epsilon \sum_{i=1}^{n} c_{i,t} w_{i,t} \leq W_t - \epsilon c_t W_t \tag{17.4.18}$$

$$= W_t(1 - c_t\epsilon) \leq W_t e^{-\epsilon c_t} \tag{17.4.19}$$

$$W_T \leq W_1 e^{-\epsilon \sum_{t=1}^{T} c_t} \tag{17.4.20}$$

$$M = E\Big[\sum_{t=1}^{T} c_t\Big] \tag{17.4.21}$$

$$W_T \leq W_1 e^{-\epsilon M} \tag{17.4.22}$$

$$(1-\epsilon)^m \leq n e^{-\epsilon M} \tag{17.4.23}$$

$$m \cdot log(1-\epsilon) \leq log_e(n) - \epsilon M \tag{17.4.24}$$

$$\epsilon M \leq log_e(n) + m \cdot log\Big(\frac{1}{1-\epsilon}\Big) \tag{17.4.25}$$

$$\epsilon M \leq log_e(n) + m \frac{\epsilon}{1-\epsilon} \tag{17.4.26}$$

Therefore we conclude that Hedge incurrs a cost M

$$M \leq m \frac{1}{1-\epsilon} + \frac{1}{\epsilon} log(n)$$

We observe that $\frac{1}{1-\epsilon} < 1 + 2\epsilon$, and that $m < T$, thus

$$M \leq m + 2\epsilon m + \frac{1}{\epsilon} log(n)$$

Thus, if we set $\epsilon = \sqrt{\frac{log(n)}{2T}}$, we can obtain

$$M \leq m + 2\sqrt{2T log(n)}$$

The result obtained by the Hedge algorithm is tight, in respect to its regret. Regret of an algorithm is defined to be $M - m$, and in the case of Hedge we obtain

$$M - m \leq O(\sqrt{T log(n)}) = o(T)$$

Hedge belongs to a larger family of exponential weight update, or multiplicative weight update algorithms.