**CS270 Algorithms & Data Structures** **Spring 2008**

# Lecture 16 On-line learning, Multiplicative Weights, and Linear Programs: 3.08.08

*Lecturer: Satish Rao* *Scribe: Rao, for now*

**Disclaimer**: *These are rough notes, with some exercises.*

## 16.1 Multiplicative updates.

Each of $n$ stock advisor gives advice on a stock, whether it is to go up or down, every day. Which one should you trust?

**Question: Can you eventually do as well as the best?**

In the beginning, let's trust them all equally, and make a prediction based on uniformly weighting them,i.e. computing $\sum_i w_i p_i$ for each expert $i$ and for weight initially $1/n$ on each expert.

Let's say up or down based on the sign of the result.

If an expert was wrong we decrease his weight a bit, say by a factor of $(1-\epsilon)$ if he or she was wrong.

**Question: How well do you do?**

**Lemma 16.1** *If $m_i^t$ is the number of mistakes expert $i$, and $m^t$ be the number of mistakes made by the algorithm, then*

$$m^t \le \frac{2\ln n}{\epsilon} + 2(1+\epsilon)m_i^t.$$

**Question: Why? Well, what is $w_i^t$?**

Well,

$$w_i^t = (1-\epsilon)^t.$$

**Question: Well, then?**

Let the potential be the sum of the weights. Every time one makes a mistake the potential decreases, half the weights decrase by a factor of $(1-\epsilon)$ thus the potential decreases by $(1-\epsilon/2)$. Futhermore, $\Phi \ge w_i^t$, for all $i$ and we get that the inequality.

$$(1-\epsilon/2)^{m^t} \ge (1-\epsilon)^{m_i^t}\frac{1}{n}$$

Taking logs, and solving for $m^t$ we get.

$$m^t \leq 2(1-\epsilon)m_i^t + \ln n/\epsilon.$$

## 16.2 Doing better.

Let's choose an expert with probability proportional to $w_i^t/\Phi_i^t$.

Now, the expected cost (or expected number of mistakes) at time $t$ is

$$\sum_{i \text{ makes a mistake at time } t} w_i^t/\Phi^t.$$

Also, the $\Phi^{t+1} - \Phi^t$ is

$$(-\epsilon) \sum_{i \text{ makes a mistake at time } t} w_i^t.$$

Or the fraction that it decreases by is $(1 - \epsilon C^t)$, where $C^t$ is expected cost of time $t$.

Thus

$$\Phi^t < e^{-\epsilon \sum_t C_t} = e^{-\epsilon C},$$

where $C$ is the expected cost of the algorithm. (Assuming $\epsilon < 1$.)

Furthermore,

$$\Phi^t > w_i^t > (1-\epsilon)^m (1/n)$$

where $m$ is the number of mistakes made by the best expert.

Putting them together, you get

$$e^{-\epsilon C} \geq (1-\epsilon)^m (1/n).$$

Taking logs, and solving for $C$ one gets.

$$C \leq -m \ln(1-\epsilon)/\epsilon + \ln n/\epsilon.$$

Here, we note that $-\ln(1-x) \leq x + x^2$ and get

$$C \leq m(1+\epsilon) + \frac{\ln n}{\epsilon}. \tag{16.1}$$

**Question: What if not in 0-1?**

Well, we can scale to [0,1] and multiply bounds by max range. Though, if positive and negative somewhat bad things happen. See the survey by Arora to get a general proof which is easy (though there are a couple of more parameters to understand.)

## 16.3 Applications?

See next lecture notes.