Optimization Methods in Finance **(EPFL, Fall 2010)**

Lecture 1: Following experts advice

22.09.2010

Lecturer: Friedrich Eisenbrand *Scribe: Friedrich Eisenbrand*

1 Motivation

In many situations, one has to choose repeatedly from a given set of options. In finance, for example, one is regularly confronted with the question on how to invest. There are many *experts* that may tell you how they foresee a future development. These experts are far from perfect and could even give bad advice. Nevertheless, some experts could be qualified. But who the good, the bad and the ugly ones are can be determined only in hindsight. However, you observe that some experts get rewarded along the way and this could help you to bet on the right ones. Our following studies are motivated by the following question: *Are there strategies that are competitive with the ones of the best expert in hindsight?*

We give a somewhat positive answer to this question and quantify exactly the competitiveness. The next lectures circle around these methods and cover in particular

- Weighted majority algorithms
- Zero sum games and their approximate solution
- Construction of a portfolio that is competitive with the best constantly re-balanced portfolio

2 A forecasting problem

Consider the following setting. There are N experts and time-steps $t = 0, \dots, T$. At time t, expert j makes a prediction $y_j^t \in \{0,1\}$, the *forecaster* predicts $\hat{p}_t \in \{0,1\}$ and *nature reveals* $z_t \in \{0,1\}$. The forecaster makes a *prediction mistake* at time t if $z_t \neq \hat{p}_t$ holds.

Some experts are good, others are bad. The question is: *Can we predict nearly as well as the best expert in hindsight?*

Let us assume, for a moment, that at least one expert is perfect and never makes a prediction mistake, i.e., $y_j^t = z_t$ for each $t = 0, \dots, T$. Here is an algorithm that penalizes experts making a mistake and finds the perfect one.

Algorithm 1.

Initialize:	Set $w_j := 1$ for $j = 1, \dots, N$.
At time t:	(Forecast)
	If $\lvert\{j \colon w_j = 1, y_j^t = 1\}\rvert \geq \lvert\{j \colon w_j = 1, y_j^t = 0\}\rvert$
	Then $\hat{p}_j = 1$

Else $\hat{p}_j = 0$

(Observe and Penalize)
For each $j \in 1,\dots,N$ with $y_j^t \neq z_t$
set $w_j := 0$

Theorem 2.1. *If one expert is perfect, then using Algorithm 1, the forecaster makes at most $\lfloor \log_2(N) \rfloor$ prediction mistakes.*

Proof. The forecaster predicts as the majority of active (those with $w_j = 1$) experts does. If the forecaster errs, then at least half of the still active experts become inactive. Let A_m denote the number of active experts after the m-th mistake of the forecaster. One has $A_0 = N$ and $A_{m+1} \leq A_m/2$ and since $A_m \geq 1$ it follows that $1 \leq A_m \leq N/2^m$ holds and thus $m \leq \lfloor \log_2(N) \rfloor$. □

What, if there is no perfect expert? One solution would be to re-activate all experts, once all of them have become inactive. It is quite straightforward to see that the number of mistakes commited by the forecaster is bounded by $(\log_2(N)+1)(m_j+1)$, if m_j is the number of mistakes committed by the j-th expert and thus that by using this scheme, the forecaster's number of mistakes is roughly the logarithm of the number of experts times the number of mistakes committed by the best expert. However, one can do much better with the next algorithm.

Algorithm 2 (Weighted Majority Algorithm)**.**

Initialize: Set $w_j := 1$ for $j = 1,\dots,N$.
At time t: (Forecast)
If $\sum_{j:\, y_j^t=1} w_j \geq \sum_{j:\, y_j^t=0} w_j$
Then $\hat{p}_j = 1$
Else $\hat{p}_j = 0$

(Observe and Re-Weight)
For each $j \in 1,\dots,N$ with $y_j^t \neq z_t$
set $w_j := w_j/2$

Theorem 2.2. *If m^* denotes the number of forecasting mistakes and m_j denotes the number of mistakes committed by expert j, then*

$$m^* \leq \frac{1}{\log_2(4/3)}(m_j + \log_2(N)).$$

Proof. Let $W = \sum_{j=1}^N w_j$ denote the total weight of the experts that is changing from one time-step to another. After each forecasting mistake, the weight drops at least by a factor of $3/4$. Since $W \geq w_j = (1/2)^{m_j}$ one has $(1/2)^{m_j} \leq N \cdot (3/4)^{m^*}$ and thus

$$\begin{aligned} (4/3)^{m^*} &\leq N \cdot 2^{m_j} \\ \iff \quad m^* \log_2(4/3) &\leq \log_2(N) + m_j \end{aligned}$$

which implies the claim. □

We can *re-interpret* this forecasting problem as follows. In each round a *loss vector* $\ell^t \in \{0,1\}^N$ is revealed that here depends on the predictions of the experts and the actual observation of z_t. The number of mistakes of expert j is the *total loss* of expert j

$$\sum_{t=1}^{T} \ell_j^t.$$

Also the forecaster experiences a loss $\hat{\ell}^t$ at time t and the weighted majority algorithm guarantees

$$\sum_{t=0}^{T} \hat{\ell}^t \leq (1/\log_2(4/3)) \cdot \left(\sum_{t=0}^{T} \ell_j^t + \log_2(N) \right).$$

If the loss vector ℓ^t is defined as

$$\ell_j^t = \begin{cases} 1 & \text{if } z_t \neq y_j^t \\ 0 & \text{if } z_t = y_j^t, \end{cases}$$

then one obtains the setting of Algorithm 2.

3 The randomized weighted majority algorithm

We finally come to the strongest algorithm for online-prediction. We generalize the setting along the lines of the re-interpretation given above. Again, we have N experts. In each round, nature provides us with a loss vector $\ell^t \in [0,1]^N$. Notice that in this setting, the experts do not make a prediction anymore. Throughout the time-steps, the expert j experiences a loss $L^j = \sum_{t=0}^{T} \ell_j^t$.

At time t, the forecaster chooses an expert j (before the loss vector ℓ^t is revealed) and experiences the same loss ℓ_j^t as the expert j. In the randomized weighted majority algorithm, this choice of the expert is done at random according to a changing probability distribution on the experts. Here are the details.

Algorithm 3 (Randomized Weighted Majority Algorithm)**.**

Initialize: Set $w_j := 1$ for $j = 1, \ldots, N$.
At time t: (Forecast)
Select expert j with probability $p_j = w_j / \sum_{k=1}^{N} w_k$

(Observe and Re-Weight)
Observe loss vector ℓ^t
Forecaster experiences loss ℓ_j^t
For $j = 1, \ldots, N$
set $w_j := w_j (1-\varepsilon)^{\ell_j^t}$

Further reading to these topics are the papers [2, 1].

References

[1] S. Arora, E. Hazan, and S. Kale. The multiplicative weights update method: a meta algorithm and applications. Technical report, Princeton, 2005.

[2] D. P. Helmbold, R. E. Schapire, Y. Singer, and M. K. Warmuth. On-line portfolio selection using multiplicative updates. *Mathematical Finance*, 8(4):325–347, 1998.