CMU 15-451/15-651

Lecturer: Avrim Blum 11/16/15

# The multiplicative weights method

## Last time / today

Last time: looked at model where data is coming from some probability distribution.

- Take a sample S, find h with low $err_S(h)$.
- Ask: when can we be confident that $err_D(h)$ is low too? (Or more generally, that the gap $|err_D(h) - err_S(h)|$ is low.)
- Gives us confidence in our predictions.

Today: what if we don't assume the future looks like the past. What can we say then?

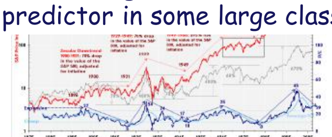
Will be more like online algorithms / competitive analysis, and how we analyzed Perceptron.

## Online learning

- What if we don't want to make assumption that data is coming from some fixed distribution? Or any assumptions on data?
- Can no longer talk about past performance predicting future results.

Idea: regret bounds.

➢Show that our algorithm does nearly as well as best predictor in some large class.



## Using "expert" advice

Say we want to predict the stock market.

- We solicit n "experts" for their advice. (Will the market go up or down?)
- We then want to use their advice somehow to make our prediction. E.g.,

| Expt 1 | Expt 2 | Expt 3 | neighbor's dog | truth |
|---|---|---|---|---|
| down | up | up | up | up |
| down | up | up | down | down |
| … | … | … | … | … |

Basic question: Is there a strategy that allows us to do nearly as well as best of these in hindsight?

["expert" = someone with an opinion. Not necessarily someone who knows anything.]

## Simpler question

- We have n "experts".
- One of these is perfect (never makes a mistake). We just don't know which one.
- Can we find a strategy that makes no more than lg(n) mistakes?

Answer: sure. Just take majority vote over all experts that have been correct so far.

➢Each mistake cuts # available by factor of 2.

➢Note: this means ok for n to be very large.

## What if no expert is perfect?

But what if none is perfect? Can we do nearly as well as the best one in hindsight?

Strategy #1:

- Iterated halving algorithm. Same as before, but once we've crossed off all the experts, restart from the beginning.
- Makes at most $\lg(n)[OPT+1]$ mistakes, where OPT is #mistakes of the best expert in hindsight.

Seems wasteful. Constantly forgetting what we've "learned". Can we do better?

## What if no expert is perfect?

Intuition: Making a mistake doesn't completely disqualify an expert. So, instead of crossing off, just lower its weight.

Weighted Majority / Multiplicative Weights Alg:

- Start with all experts having weight 1.
- Predict based on weighted majority vote.
- Penalize mistakes by cutting weight in half.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Weights: | 1 | 1 | 1 | 1 | | | |
| Predictions: | U | U | U | D | We predict: U | Truth: D | |
| Weights: | $\frac{1}{2}$ | $\frac{1}{2}$ | $\frac{1}{2}$ | 1 | | | |

## Analysis: do nearly as well as best expert in hindsight

- M = # mistakes we've made so far.
- m = # mistakes best expert has made so far.
- W = total weight (starts at n).
- After each mistake, W drops by at least 25%. So, after M mistakes, W is at most $n(3/4)^M$.
- Weight of best expert is $(1/2)^m$. So,

$$(1/2)^m \leq n(3/4)^M$$
$$(4/3)^M \leq n2^m$$
$$M \leq 2.4(m + \lg n)$$

Constant Ratio! So, if m is small, then M is pretty small too.

## Randomized Wtd Majority / Mult Wts

2.4(m + lg n) not so good if the best expert makes a mistake 20% of the time. Can we do better? Yes.

- Instead of taking majority vote, use weights as probabilities. (e.g., if 70% on up, 30% on down, then pick 70:30) Idea: smooth out the worst case.
- Also, multiply by 1- ε instead of ½.

Solves to $M \leq \frac{-m\ln(1-\epsilon)+\ln(n)}{\epsilon} \approx \left(1+\frac{\epsilon}{2}\right)m + \frac{1}{\epsilon}\ln(n)$

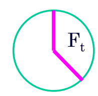
M = expected #mistakes

$M \leq 1.39m + 2\ln n \quad \leftarrow \varepsilon = 1/2$

$M \leq 1.15m + 4\ln n \quad \leftarrow \varepsilon = 1/4$

$M \leq 1.07m + 8\ln n \quad \leftarrow \varepsilon = 1/8$

## Analysis




- Say at time $t$ we have fraction $F_t$ of weight on experts that made mistake.
- So, we have probability $F_t$ of making a mistake, and we remove an $\varepsilon F_t$ fraction of the total weight.
  - $W_{final} = n(1-\varepsilon F_1)(1 - \varepsilon F_2)\ldots$
  - $\ln(W_{final}) = \ln(n) + \sum_t [\ln(1 - \varepsilon F_t)] \leq \ln(n) - \varepsilon \sum_t F_t$ (using $\ln(1-x) < -x$)

    $= \ln(n) - \varepsilon M.$ ($\sum F_t = E[\#\text{ mistakes}]$)
- If best expert makes **m** mistakes, then $\ln(W_{final}) > \ln((1-\varepsilon)^m)$.
- Now solve: $\ln(n) - \varepsilon M > m \ln(1-\varepsilon)$.

Solves to $M \leq \frac{-m\ln(1-\epsilon)+\ln(n)}{\epsilon} \approx \left(1+\frac{\epsilon}{2}\right)m + \frac{1}{\epsilon}\ln(n)$

## What can we use this for?

- Can use for repeated play of matrix game:
  - Consider cost matrix where all entries 0 or 1.
  - Rows are different experts. Start each with weight 1.
    - Notice that the RWM algorithm is equivalent to "pick an expert with prob $p_i = w_i/\sum_j w_j$, and go with it".
    - Can apply when experts are *actions* rather than *predictors*.
    - $F_t$ = fraction of weight on rows that had "1" in adversary's column.
  - Analysis shows do nearly as well as best row in hindsight!

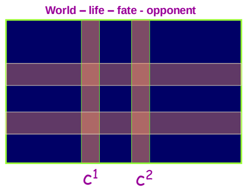
## What can we use this for?

In fact, alg/analysis extends to costs in [0,1], not just {0,1}.

- We assign weights $w_i$, inducing probabilities $p_i = w_i/\sum_j w_j$.
- Adversary chooses column. Gives cost vector $\vec{c}$. We pay (expected cost) $\vec{p} \cdot \vec{c}$.
- Update: $w_i \leftarrow w_i(1 - \epsilon c_i)$.
- A few minor extra calculations in analysis…

## RWM / MW

$(1-\varepsilon c_1^2)(1-\varepsilon c_1^1)1$
$(1-\varepsilon c_2^2)(1-\varepsilon c_2^1)1$
$(1-\varepsilon c_3^2)(1-\varepsilon c_3^1)1$
. . 1
. . 1
$(1-\varepsilon c_n^2)(1-\varepsilon c_n^1)1$




scaling so costs in [0,1]

$\text{E[cost]} \le (1+\epsilon)OPT + \left(\frac{1}{\epsilon}\right)\log(n)$

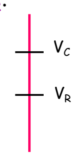
In T steps, $\text{E[cost]} \le OPT + \epsilon T + \left(\frac{1}{\epsilon}\right)\log(n)$

## RWM / WM

In fact, gives a proof of the minimax theorem…

## Nice proof of minimax thm (sketch)

- Suppose for contradiction it was false.
- This means some game G has $V_C > V_R$:
  - If Column player commits first, there exists a row that gets the Row player at least $V_C$.
  - But if Row player has to commit first, the Column player can make him get only $V_R$.
- Scale matrix so payoffs to row are in [-1,0]. Say $V_R = V_C - \delta$.




## Proof sketch, contd

- Now, consider randomized weighted-majority alg, against Col who plays optimally against Row's distrib.
- In T steps,
  - Alg gets $\ge$ [best row in hindsight] $-\epsilon T - \log(n)/\varepsilon$
  - BRiH $\ge TV_C$ [Best against opponent's empirical distribution]
  - Alg $\le TV_R$ [Each time, opponent knows your randomized strategy]
  - Gap is $\delta T$. Contradicts assumption if use $\varepsilon = \delta/2$, once $T > \log(n)/\epsilon^2$.

## [ACFS02]: applying RWM to bandit setting

- What if only get your own cost/benefit as feedback?

- Called the "multi-armed bandit problem"
- Will do a somewhat weaker version of their analysis (same algorithm but not as tight a bound).
- For fun, talk about it in the context of online pricing…

## Online pricing

- Say you are selling lemonade (or a cool new software tool, or bottles of water at the world expo).
- Protocol #1: for t=1,2,…T
  - Seller sets price $p^t$
  - Buyer arrives with valuation $v^t$
  - If $v^t \ge p^t$, buyer purchases and pays $p^t$, else doesn't.
  - $v^t$ revealed to algorithm.
  - repeat
- Protocol #2: same as protocol #1 but without $v^t$ revealed.
- Assume all valuations in [1,h]
- Goal: do nearly as well as best fixed price in hindsight.

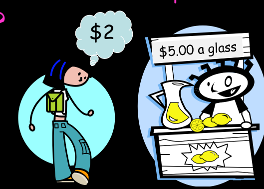
## Online pricing

- Say you are selling lemonade (or a cool new software tool, or bottles of water at the world expo).
- Protocol #1: for t=1,2,…T
  - Seller sets price $p^t$
  - Buyer arrives with valuation $v^t$
  - If $v^t \geq p^t$, buyer purchases and pays $p^t$, else doesn't.
  - $v^t$ revealed to algorithm.
- Good algorithm: RWM / MW!
  - Define one expert for each price $p \in [1,h]$. (#experts = h)
  - Best price of this form gives profit OPT.
  - Run RWM algorithm. Get expected gain at least: $OPT(1-\epsilon) - O(\epsilon^{-1} h \log h)$

[extra factor of h coming from range of gains]

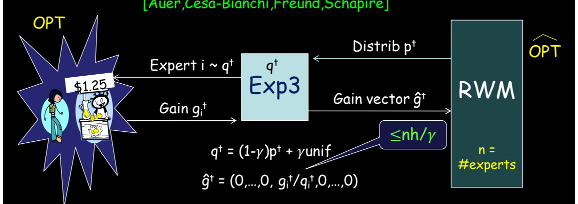
## Online pricing

- Say you are selling lemonade (or a cool new software tool, or bottles of water at the world expo).
- What about Protocol #2? [just see accept/reject decision]
  - Now we can't run RWM directly since we don't know how to penalize the experts!
  - Called the "adversarial multiarmed bandit problem"
  - How can we solve that?




## Multi-armed bandit problem

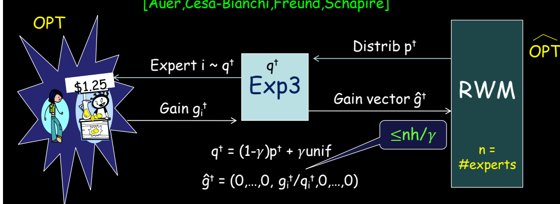
Exponential Weights for Exploration and Exploitation (exp$^3$)
[Auer,Cesa-Bianchi,Freund,Schapire]




1. RWM believes gain is: $p^t \cdot \hat{g}^t = p_i^t(g_i^t/q_i^t) \equiv g^t_{RWM}$
2. $\sum_t g^t_{RWM} \geq \widehat{OPT}(1-\epsilon) - O(\epsilon^{-1}\, nh/\gamma \log n)$
3. Our actual gain is: $g_i^t = g^t_{RWM}(q_i^t/p_i^t) \geq g^t_{RWM}(1-\gamma)$
4. $E[\widehat{OPT}] \geq OPT$. Because $E[\hat{g}_j^t] = (1-q_j^t)0 + q_j^t(g_j^t/q_j^t) = g_j^t$, so $E[\max_j[\sum_t \hat{g}_j^t]] \geq \max_j[E[\sum_t \hat{g}_j^t]] = OPT$.

## Multi-armed bandit problem

Exponential Weights for Exploration and Exploitation (exp$^3$)
[Auer,Cesa-Bianchi,Freund,Schapire]




Conclusion ($\gamma = \epsilon$):
$E[Exp3] \geq OPT(1-\epsilon)^2 - O(\epsilon^{-2}\, nh \log(n))$

Can reduce $1/\epsilon^2$ term to $1/\epsilon$ with more care in analysis.

## Summary

Algorithms for online decision-making with strong guarantees on performance compared to best fixed choice.

- Application: play repeated game against adversary. Perform nearly as well as fixed strategy in hindsight.

Can apply even with very limited feedback.

- Application: online pricing, even if only have buy/no buy feedback.