

Lecture 16. Learning with expert advice

COMP90051 Statistical Machine Learning

Lecturer: Christine de Kock



THE UNIVERSITY OF
MELBOURNE

This lecture

- Learning from expert advice / multiplicative weights
 - * Learner listens to some/all experts making predictions
 - * True outcomes are ADVERSARIAL!
 - * Learner updates weights over experts based on their losses
 - * Algorithms all forms of “multiplicative weights”
 - * Nice clean bounds on total mistakes/loss: by “potential function” technique
- Infallible expert (one always perfect)
 - * Majority Vote Algorithm
- Imperfect experts (none guaranteed perfect) – increasingly better
 - * Weighted Majority Vote Algorithm by Halving
 - * Weighted Majority Voting by General Multiplicative Weights
 - * Probabilistic Experts Algorithm

An infallible expert and the Majority Algorithm

Warming up example

Warm-up: Case of the infallible expert

- **Experts** E_1, \dots, E_n predict the stock market daily
 - * Each expert prediction is binary: stocks will go up/down
- Learner's game, daily:
 - * Observe predictions of all experts
 - * Make a prediction of its own
 - * Observe outcome (could be anything!)
 - * Goal: minimise number total mistakes
- **Infallible expert** assumption:
 - * 1 or more experts makes no mistakes



NYSE. CCA: skeeze

Infallible Expert Algorithm: Majority Vote

1. Initialise set of experts who haven't made mistakes $E = \{1, \dots, n\}$
2. Repeat per round
 - a) Observe predictions E_i for all $i \in \{1, \dots, n\}$
 - b) Make **majority prediction**
 $\arg \max_{y \in \{-1, 1\}} \sum_{i \in E} 1[E_i = y]$
 - c) Observe correct outcome
 - d) **Remove mistaken experts** from E



CCA3.0: Krisada, Noun Project

Mistake Bound for Majority Vote

Proposition: Under infallible expert assumption, majority vote makes total mistakes $M \leq \log_2 n$

Intuition: Halving (e.g. tree data structures!)? Expect to see log

Proof

- Loop invariant: If algorithm makes a mistake, then at least $|E|/2$ experts must have been wrong
- I.e. for every incorrect prediction, E reduced by at least half. I.e. after M mistakes, $|E| \leq n/2^M$
- By infallibility, at all times $1 \leq |E|$
- Combine to $1 \leq |E| \leq n/2^M$, then solve for M .

**$|E|$ is the
potential
function**

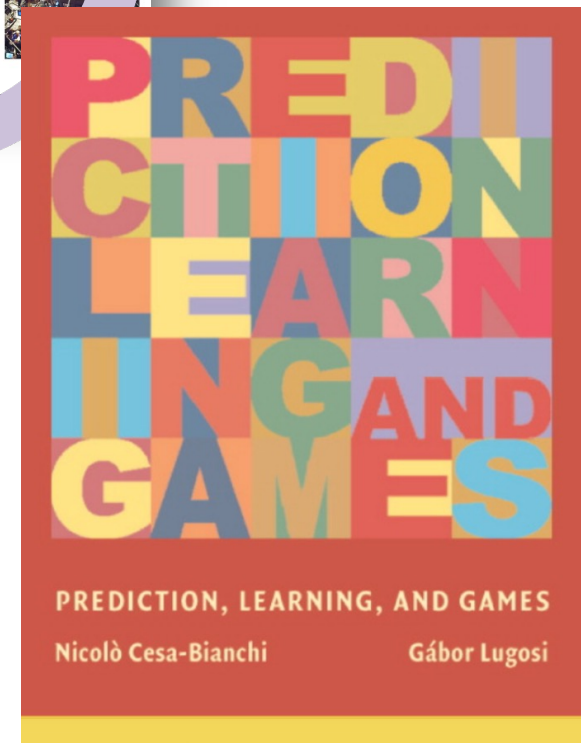
How is this “online learning”?

Learning

- **Weights on which experts** are worth listening to
- (Infallible case: 0/1 weights)
- Making predictions/
taking actions
- Incurring loss (so far 0/1)
- IID “Distribution” replaced by **adversarial outcomes**

Online

- A **repeated game**



Mini Summary

- Learning with expert advice paradigm
 - * Abstraction of online learning problem
 - * Adversarial feedback
 - * Later: Applications abound
- Bounds on mistakes (later losses) “easy”
 - * Involve “potential function” technique
 - * Later: interested in scaling with best expert performance

Next: Imperfect experts. Down weight don't drop bad experts

Imperfect experts and the Halving Algorithm

*Similar proof technique; similar algorithm;
much more interesting setting*

No one's perfect

- No more guarantee of an infallible expert
- What breaks?
 - * We could end up with $E = \emptyset$, how to predict then?
 - * No sense: “Zero tolerance” dropping experts on a mistake
- Very general setting / very few assumptions
 - * Not assuming anything about expert error rates
 - * Not assuming anything about correlation of expert errors
 - * Not assuming anything about outcome observations. Not even stochastic (could be adversarial!)

Imperfect experts: Halving Algorithm

1. Initialise $w_i = 1$ weight of expert E_i
2. Repeat per round
 - a) Observe predictions E_i
for all $i \in \{1, \dots, n\}$
 - b) Make **weighted majority prediction**
 $\arg \max_{y \in \{-1, 1\}} \sum_{i \in E} w_i 1[E_i = y]$
 - c) Observe correct outcome
 - d) **Downweigh each mistaken expert E_i**
 $w_i \leftarrow w_i / 2$



CCA3.0: Krisada, Noun Project

Mistake Bound for Halving

Proposition: If the best expert makes m mistakes, then weighted majority vote makes $M \leq 2.4(m + \log_2 n)$ mistakes.

Proof

- Invariant: If algorithm makes a mistake, then weight of wrong experts is at least half the total weight $W = \sum_{i=1}^n w_i$
- Weight of wrong experts reduced by $1/2$, therefore total weight reduced by at least $3/4$. I.e. after M mistakes, $W \leq n(3/4)^M$
- Best expert E_i has $w_i = (1/2)^m$
- Combine to $(1/2)^m = w_i \leq W \leq n(3/4)^M$
- Taking logs $-m \leq \log_2 n + M \log_2(3/4)$, solving $M \leq \frac{m + \log_2 n}{\log_2(4/3)}$

Compare, compare: What's going on?

- Price of imperfection (vs. infallibility) is $\mathcal{O}(m)$
 - * Infallible case: $M \in \mathcal{O}(\log n)$
 - * Imperfect case: $M \in \mathcal{O}(m + \log n)$
- Scaling to many experts is no problem
- Online learning vs. PAC frameworks

	Modelling of losses	Ultimate goal
PAC	i.i.d. losses (due to e.g. Hoeffding)	(For ERM; L6) Small estimation error $R[f_m] - R[f^*]$. Bounded in terms of family's VC dimension
Online learning	Adversarial/arbitrary losses	Small $M - m$. Bounded in terms of number of experts.

Mini Summary

- Imperfect expert setting
 - * Don't drop bad experts, just halve their weight
 - * Predict by weighted majority, not simply majority
 - * Mistake bound follows similar “potential function” pattern!
- Learning with expert advice paradigm
 - * Key difference to PAC is adversarial feedback
 - * Similarity: Also concerned with performance relative to “best in class”

Next: Imperfect experts continued. Generalising halving.

From Halving to Multiplying weights by $1 - \varepsilon$

Generalising weighted majority.

Useful (but otherwise boring) inequalities

- Lemma 1: For any $\varepsilon \in [0, 0.5]$, we have
$$-\varepsilon - \varepsilon^2 \leq \log_e(1 - \varepsilon) < -\varepsilon$$

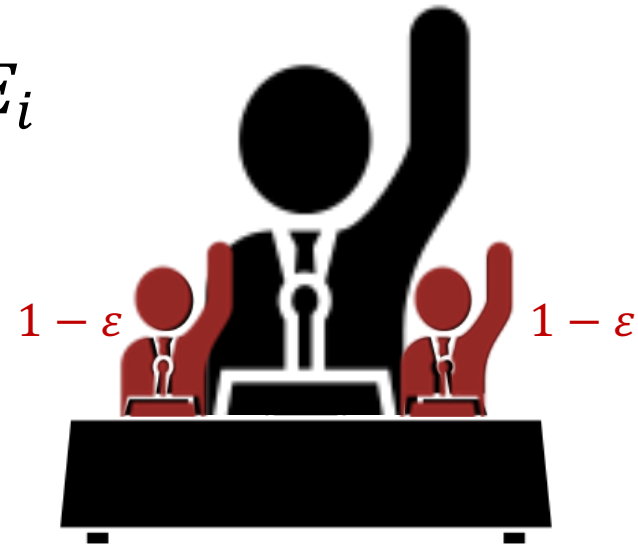
- Proof:
 - * Upper bound by Taylor expansion, dropping all but first term (as they're negative)
 - * Lower bound by convexity of $\exp(-\varepsilon - \varepsilon^2)$

- Lemma 2: For all $\varepsilon \in [0, 1]$ we have,
$$1 - \varepsilon x > \begin{cases} (1 - \varepsilon)^x, & \text{if } x \in [0, 1] \\ (1 + \varepsilon)^{-x}, & \text{if } x \in [-1, 0] \end{cases}$$

- Proof: by convexity of the RHS functions

Weighted Majority Vote Algorithm

1. Initialise $w_i = 1$ weight of expert E_i
2. Repeat per round
 - a) Observe predictions E_i for all $i \in \{1, \dots, n\}$
 - b) Make weighted majority prediction $\arg \max_{y \in \{-1, 1\}} \sum_{i \in E} w_i 1[E_i = y]$
 - c) Observe correct outcome
 - d) Downweigh each mistaken expert E_i
 $w_i \leftarrow (1 - \varepsilon)w_i$



CCA3.0: Krisada, Noun Project

Mistake Bound

Proposition: If the best expert makes m mistakes, then $(1 - \varepsilon)$ -weighted majority makes $M \leq 2(1 + \varepsilon)m + (2\log_e n)/\varepsilon$ mistakes.

Bound improves dependence on m compared to halving. **Why?**

Proof

- Whenever learner mistakes, at least half of total weight reduced by factor of $1 - \varepsilon$. So after M mistakes, $W \leq n(1 - \varepsilon/2)^M$
- Best expert E_i has $w_i = (1 - \varepsilon)^m$
- Combine to $(1 - \varepsilon)^m = w_i \leq W \leq n(1 - \varepsilon/2)^M$
- Taking logs: $m\log_e(1 - \varepsilon) \leq \log_e n + M\log_e(1 - \varepsilon/2)$
- Lemma 1 replaces both $\log_e(1 - \varepsilon)$: $-m(\varepsilon + \varepsilon^2) \leq \log_e n - M\varepsilon/2$
- Solving for M proves the bound.

Dependence in m provably near optimal!

- New to lower bounds? example shows an analysis or even an algorithm can't do better than some limit
- Weighted majority almost achieves $2m$ dependence, with $2(1 + \varepsilon)m$ (considering no. experts fixed)
- Example with $M = 2m$
 - * Consider $n = 2$ with E_1 (E_2) correct on odd (even) days
 - * Then best expert makes mistakes half the time
 - * But after 1st round, for any ε , majority vote is wrong all the time, as incorrect expert gets more than half weight
- Consequence? Can't improve the constant 2 factor in m

Mini Summary

- Imperfect expert setting continued...
- From halving to multiplicative weights!
 - * Mistake bound proved as usual via “potential function” trick
 - * Bound’s dependence on best expert improved to $2 + \varepsilon$ factor
- Lower bound / impossibility result
 - * Factor of 2 is optimal for (deterministic) multiplicative weights!

Next: Imperfect experts continued. Randomise!!

The probabilistic experts algorithm

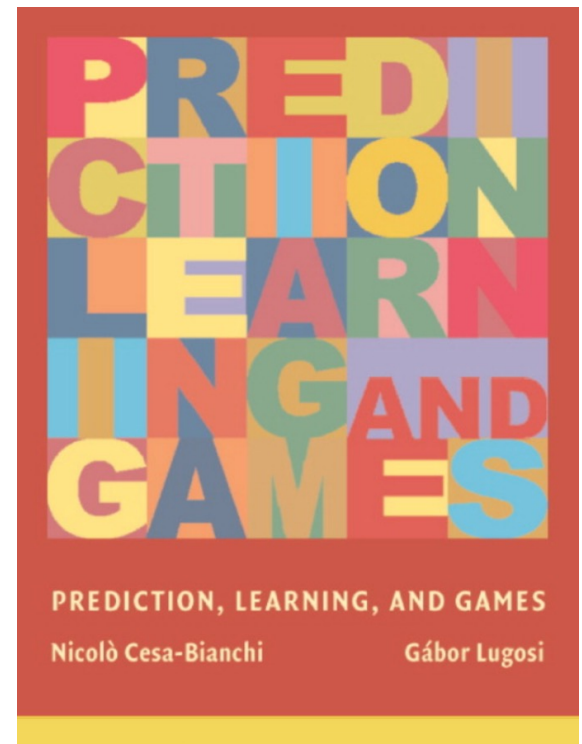
wherein randomisation helps us do better!

Probabilistic experts algorithm

- Change 1 from mistakes: **Loss** $\ell_i^{(t)} \in [0,1]$ of E_i , round t
 - Change 2: Randomised algorithm means, bounding **expected losses** (sound familiar? It should.... a risk!)
1. Initialise $w_i = 1$ weight of expert E_i
 2. Repeat per round
 - a) Observe predictions E_i for all $i \in \{1, \dots, n\}$
 - b) Predict E_i of expert i with probability $\frac{w_i}{W}$ where $W = \sum_{i=1}^n w_i$
 - c) Observe **losses**
 - d) Update each weight $w_i \leftarrow (1 - \varepsilon)^{\ell_i^{(t)}} w_i$

Probabilistic experts: Expected loss bound

- Proposition: Expected loss of the probabilistic experts algorithm is $L \leq \frac{\log e n}{\varepsilon} + (1 + \varepsilon)L^*$ where L^* is the minimum loss over experts.
- Proof: next, follows similar “potential” pattern
- Beats deterministic! Shaves off optimal constant 2
- Generalises in many directions. Active area of research in ML, control, economics, in top labs.



Proof: Upper bounding potential function

- Learner's round t expected loss: $L_t = \frac{\sum_{i=1}^n w_i^{(t)} \ell_i^{(t)}}{W(t)}$
- By Lemma 2, since losses are $[0,1]$:
updated $w_i^{(t+1)} \leftarrow (1 - \varepsilon)^{\ell_i^{(t)}} w_i^{(t)} \leq \left(1 - \varepsilon \ell_i^{(t)}\right) w_i^{(t)}$
- Rearrange to obtain recurrence relation:

$$\begin{aligned} W(t+1) &\leq \sum_{i=1}^n \left(1 - \varepsilon \ell_i^{(t)}\right) w_i^{(t)} = \sum_{i=1}^n w_i^{(t)} \left(1 - \varepsilon \frac{\sum_{i=1}^n w_i^{(t)} \ell_i^{(t)}}{\sum_{i=1}^n w_i^{(t)}}\right) \\ &= W(t) (1 - \varepsilon L_t) \end{aligned}$$

- Initialisation gave $W(0) = n$, so telescoping we get:

$$W(T) \leq n \prod_{t=1}^T (1 - \varepsilon L_t)$$

Proof: Lower bounding potential, Wrap up

- Proved upper bound: $W(T) \leq n \prod_{t=1}^T (1 - \varepsilon L_t)$

- Lower bound from best expert total loss L^* :

$$W(T) \geq (1 - \varepsilon)^{L^*}$$

- Combining bounds and taking log's:

$$L^* \log_e(1 - \varepsilon) \leq \log_e n + \sum_{t=1}^T \log_e(1 - \varepsilon L_t)$$

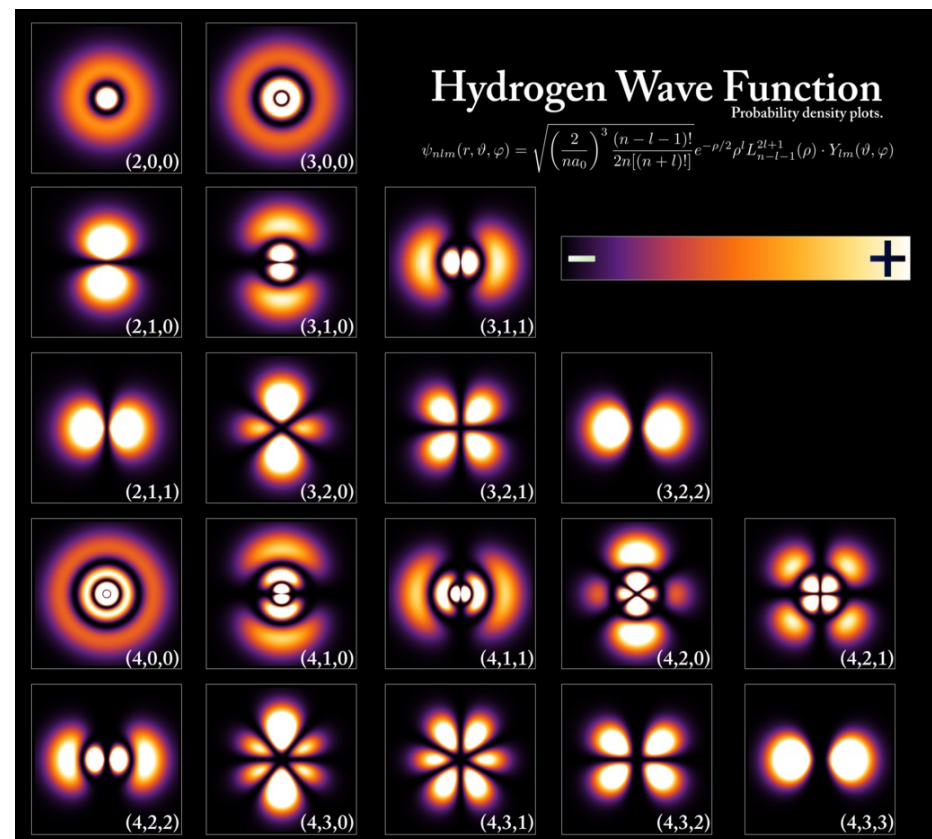
- By Lemma 1: $-L^*(\varepsilon + \varepsilon^2) \leq \log_e n - \varepsilon \sum_{t=1}^T L_t$

- Linearity of expectation $L = \sum_{t=1}^T L_t$, rearranging:

$$L \leq \frac{\log_e n}{\varepsilon} + (1 + \varepsilon)L^*$$

Applications of multiplicative weights [Kale thesis 2007]

- Learning quantum states from noisy measurements
- Derandomising algorithms
- Solving certain zero-sum games
- Fast graph partitioning
- Fast solving of semidefinite programming problems
- Portfolio optimisation
- A basis for boosting
- Sparse vector technique in differential privacy



Public domain

Mini Summary

- Introducing randomisation to learning with experts
 - * Algorithm choosing a random expert to follow
 - * Weights become probabilities
 - * Mistakes generalise to losses
- Loss bound
 - * Have to bound expected loss (hey, risk!!)
 - * Shaves off that 2 factor. Proves that randomisation really helps!

Next: Only observe reward of chosen expert → bandits!