Randomised Algorithms

Lecture 14: Online Learning with Experts

Thomas Sauerwald (`tms41@cam.ac.uk`)

Lent 2022

Outline

Introduction

Deterministic Weighted Majority

Randomised Weighted Majority

Extensions and Conclusions

Landscape of Machine Learning Algorithms

Training Set provided initially

Feedback after Decisions

No Training Set

Supervised Learning

Classification, regression: logistic regr., SVM, decision tree, neural networks, naive Bayes, Perceptron, kNN, Boosting

Online and Reinforcement Learning

Expert Learning: Weighted-Majority, Multiplicative-Update, Markov Decision Processes: Multi-Armed Bandits, Q-Learning

Unsupervised Learning

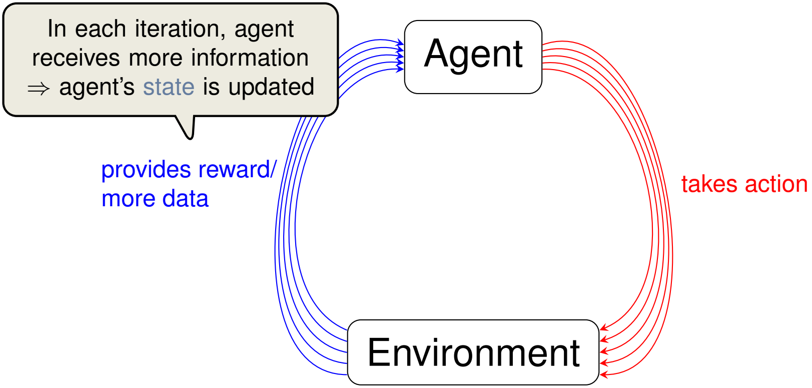
Clustering: spectral, hierarchical, k-means; Dimensionality Reduction, PCA, SVD

Predict unseen data

Maximise Reward

Extract Knowledge

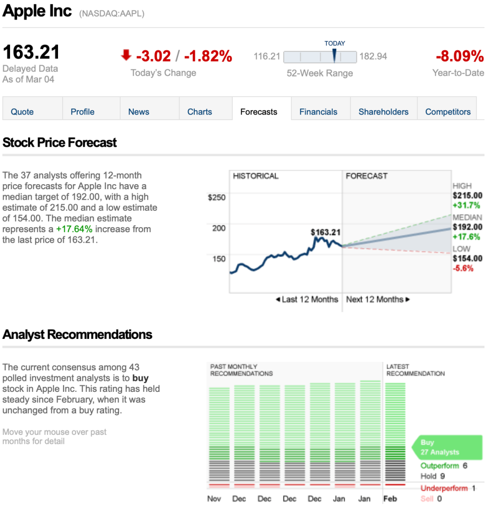
Online Algorithm/Reinforcement Learning Framework



Iteration: 4

Source: Yahoo Finance, 5 March 2022

Disclaimer: This is only given as a high-level motivation for the algorithm. It is not suggested to use any of the following ideas in practice at this or any other point.



Source: CNN Money, 5 March 2022

Other Applications: Spam Filtering, Weather Prediction, . . .

Outline

Online Learning using Expert Advice

Basic Setup

- Assume there is a single stock, and daily price movement is a sequence of binary events (up $= 1$ /down $= 0$)
- The stock movements can be arbitrary (i.e., adversarial)
- We are allowed to watch n experts (these might be arbitrarily bad and correlated)

Weighted Majority Algorithm

Initialization: Fix $\delta \leq 1/2$. For every $i \in [n]$, let $w_i^{(1)} := 1$

Update: For $t = 1, 2, \ldots, T$:

- Make prediction which is the weighted majority of the experts' predictions
- For every expert i who predicts wrongly, decrease his weight by a factor of $(1 - \delta)$:

$$w_i^{(t+1)} = (1 - \delta) w_i^{(t)}$$

Example of an **ensemble method**, combining advice from several other "algorithms".

Weighted Majority Algorithm: Example

Let $\delta = 1/2$, $n = 3$

t	Expert Weights	Expert Predictions	Our Pred.	Result	Our Errors
1	$1, 1, 1$	$1, 1, 0$	1 ✓	1	0
2	$1, 1, 1/2$	$0, 1, 0$	0 ✗	1	1
3	$1/2, 1, 1/4$	$1, 0, 1$	0 ✓	0	1
4	$1/4, 1, 1/8$	$0, 1, 1$	1 ✗	0	2
5	$1/4, 1/2, 1/16$	$1, 1, 0$	1 ✓	1	2
6	$1/4, 1/2, 1/32$	$0, 1, 1$	1 ✓	1	2
7	$1/8, 1/2, 1/32$	$0, 1, 0$	1 ✗	0	3
8	$1/8, 1/4, 1/32$	$1, 0, 1$	0 ✗	1	4
9	$1/8, 1/8, 1/32$	$0, 0, 0$	0 ✓	0	4
10	$1/8, 1/8, 1/32$	$1, 0, 1$	1 ✗	0	5
11	$1/16, 1/8, 1/64$	–	–	–	–

$\Rightarrow$ We made 5 mistakes, while the best expert made only 3 mistakes.
This looks quite bad, but the example is **too small** to draw conclusions!

Analysis of the Weighted Majority Algorithm

Notation: Let $m_i^{(t)}$ be the number of mistakes of expert i after t steps.

Analysis

The number of mistakes of our algorithm $M^{(T)}$ satisfies

$$M^{(T)} \leq 2 \cdot (1+\delta) \cdot \min_{i \in [n]} m_i^{(T)} + \frac{2 \ln n}{\delta}.$$

This bound holds for any input, any T and any δ!

Proof Outline:

- Define $\Phi^{(t)} = \sum_{i=1}^{n} w_i^{(t)}$ as the sum of weights
- Update Rule: If we make many mistakes, then $\Phi^{(t)}$ becomes small
- For $\Phi^{(t)}$ to be small, all weights must be small
  ($\Rightarrow$ even the best expert must make many mistakes)

Analysis of the Weighted Majority Algorithm

Notation: Let $m_i^{(t)}$ be the number of mistakes of expert i after t steps.

Analysis

The number of mistakes of our algorithm $M^{(T)}$ satisfies

$$M^{(T)} \leq 2 \cdot (1+\delta) \cdot \min_{i \in [n]} m_i^{(T)} + \frac{2 \ln n}{\delta}.$$

This bound holds for any input, any T and any δ!

Proof:

- Define a potential function $\Phi^{(t)} = \sum_{i=1}^{n} w_i^{(t)}$, so that $\Phi^{(1)} = n$.
- By induction, $w_i^{(t+1)} = (1-\delta)^{m_i^{(t)}}$ (see example!)
  - **Case 1:** Each time we are wrong, the weighted majority of experts is wrong $\Rightarrow$ at least half the total weight decreases by $1-\delta$:

$$\Phi^{(t+1)} \leq \Phi^{(t)} \cdot \left(\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot (1-\delta) \right) = \Phi^{(t)} \cdot (1 - \delta/2).$$

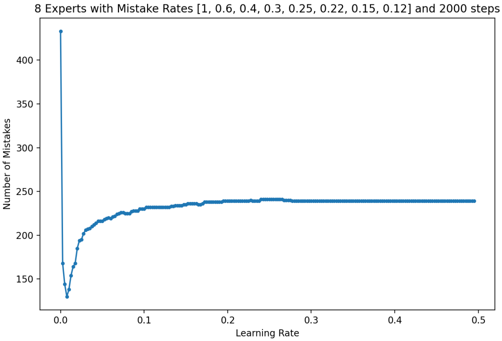
  - **Case 2:** Each time we are correct, $\Phi^{t+1} \leq \Phi^{t}$.
- By induction, $\Phi^{(T+1)} \leq n \cdot (1-\delta/2)^{M^{(T)}}$, but also $\Phi^{(T+1)} \geq w_i^{(T+1)} = (1-\delta)^{m_i^T}$.
- Taking logs:

$$m_i^{(T)} \ln(1-\delta) \leq M^{(T)} \ln(1-\delta/2) + \ln(n).$$

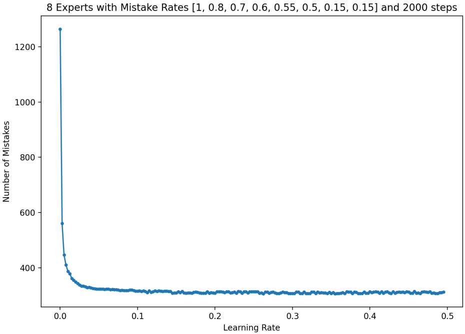
- Using now that $-\delta \geq \ln(1-\delta) \geq -\delta - \delta^2$ completes the proof. □

Simulation of the (Deterministic) WMA (1/2)

- Probabilistic Setting: Each expert i predicts wrongly with some probability $p_i \in [0, 1]$, independently across rounds and experts
- Question: Which learning rate works best?



Simulation of the (Deterministic) WMA (2/2)



Observations from these Experiments

- Depending on data set, a high or small learning rate may work best
- **But:** for such a random environment, other Machine Learning techniques (e.g., Naive Bayes or Neural Networks) work much better

The point of WMA is a strong **worst-case** guarantee!

Improving the Weighted Majority Algorithm?

Analysis

The number of mistakes of our algorithm $M^{(T)}$ satisfies

$$M^{(T)} \leq 2 \cdot (1+\delta) \cdot \min_{i \in [n]} m_i^{(T)} + \frac{2 \ln n}{\delta}.$$

Question: Is there a way to avoid the factor of 2?

Exercise: For any **deterministic** algorithm, the factor of 2 cannot be avoided!

Idea: Employ a randomised strategy which selects an expert with probability proportional to its success!

Outline

Randomised Weighted Majority

Randomised Weighted Majority Algorithm

Initialization: Fix $\delta \leq 1/2$. For every $i \in [n]$, let $w_i^{(1)} := 1$

Update: For $t = 1, 2, \ldots, T$:

- Pick expert i with probability proportional to w_i and follow that prediction
- For every expert i who predicts wrongly, decrease his weight by a factor of $(1 - \delta)$:

$$w_i^{(t+1)} = (1 - \delta) w_i^{(t)}$$

Note that the number of mistakes we are making is now a random variable!

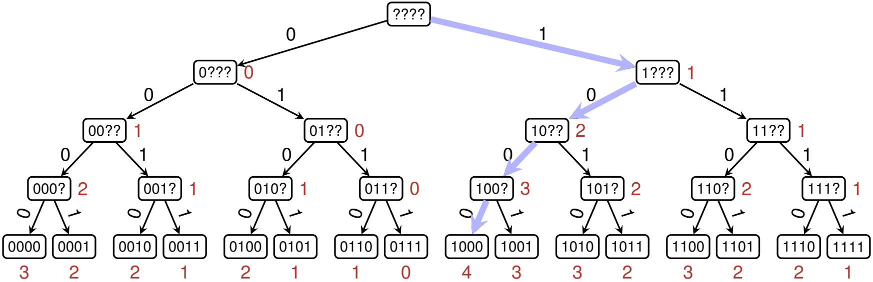
Example: Deterministic vs. Randomised Weighted Majority (1/2)

Consider the following run of the Deterministic Weighted Majority Algorithm:

t	Weights	Predictions	Our Prediction	Actual Result	Our Errors
1	1,1	1,0	1	0	1
2	1/2,1	1,0	0	1	2
3	1/2,1/2	0,1	0	1	3
4	1/4,1/2	1,0	0	1	4
5	1/4,1/4	–	–	–	–

Consider now the Randomised Weighted Majority Algorithm and let us compute the expected number of mistakes, $\mathbf{E}\left[M^{(4)}\right]$

Example: Deterministic vs. Randomised Weighted Majority (2/2)



- Let $x^{(t)}$ be a $0/1$ random variable, indicating if our t-th prediction is wrong.
- Then:

$$\mathbf{E}\left[x^{(1)}\right] = 0 \cdot \frac{1}{2} + 1 \cdot \frac{1}{2} = \frac{1}{2}.$$

- Similarly, $\mathbf{E}\left[x^{(2)}\right] = \frac{2}{3}$, $\mathbf{E}\left[x^{(3)}\right] = \frac{1}{2}$ and $\mathbf{E}\left[x^{(4)}\right] = \frac{2}{3}$.
- Hence,

$$\mathbf{E}\left[M^{(4)}\right] = \mathbf{E}\left[x^{(1)} + x^{(2)} + x^{(3)} + x^{(4)}\right]$$

$$= \mathbf{E}\left[x^{(1)}\right] + \mathbf{E}\left[x^{(2)}\right] + \mathbf{E}\left[x^{(3)}\right] + \mathbf{E}\left[x^{(4)}\right] = \frac{7}{3}$$

Much better than the deterministic algorithm!

Analysis of Randomised Weighted Majority (non-examinable)

Analysis

The expected number of mistakes of our algorithm $M^{(T)}$ satisfies

$$\mathbf{E}\left[M^{(T)}\right] \leq 1 \cdot (1+\delta) \cdot \min_{i \in [n]} m_i^{(T)} + \frac{\ln n}{\delta}.$$

This was a factor of 2 before!

Proof:

- Define a potential function $\Phi^{(t)} = \sum_{i=1}^n w_i^{(t)}$, so that $\Phi^{(1)} = n$.
- The probability of picking expert i in round t is $p_i^{(t)} := w_i^{(t)} / \sum_{j=1}^n w_j^{(t)} = w_i^{(t)} / \Phi^{(t)}$.
- Let $\lambda_i^{(t)}$ be 1 iff expert i is wrong at time t (and 0 otherwise)
- Then the expected number of mistakes by our algorithm is $\mathbf{E}[M^{(T)}] = \sum_{t=1}^T \lambda^{(t)} \cdot p^{(t)}$.
- The new potential (which is deterministic!) can be upper bounded by:

$$\Phi^{(t+1)} = \sum_{i=1}^n w_i^{(t+1)} = \sum_{i=1}^n (1-\delta\lambda_i^{(t)}) \cdot w_i^{(t+1)} = \Phi^{(t)} \cdot \left(1 - \delta\lambda^{(t)} p^{(t)}\right) \leq \Phi^{(t)} \cdot \exp\left(-\delta\lambda^{(t)} \cdot p^{(t)}\right)$$

- Thus the final potential satisfies

$$\Phi^{(T+1)} \leq \Phi^{(1)} \cdot \exp\left(-\delta \sum_{t=1}^T \lambda^{(t)} \cdot p^{(t)}\right) = n \cdot \exp\left(-\delta \cdot \mathbf{E}\left[M^{(T)}\right]\right),$$

$$\Phi^{(T+1)} \geq w_i^{(T+1)} = \prod_{t=1}^T \left(1 - \delta\lambda_i^{(t)}\right) = (1-\delta)^{m_i^{(T)}}$$

$\ln(1-\delta) \geq -\delta - \delta^2$

$$\Rightarrow \quad \ln n - \delta \cdot \mathbf{E}[M^{(T)}] \geq \ln(1-\delta) \cdot m_i^{(T)} \quad \Rightarrow \quad \mathbf{E}[M^{(T)}] \leq \frac{(\delta+\delta^2)}{\delta} \cdot m_i^{(T)} + \frac{\ln n}{\delta} \qquad \square$$

Optimising the Learning Rate

Analysis

The expected number of mistakes of our algorithm $M^{(T)}$ satisfies

$$\mathbf{E}\left[M^{(T)}\right] \leq 1 \cdot (1+\delta) \cdot \min_{i \in [n]} m_i^{(T)} + \frac{\ln n}{\delta}.$$

Interpretation:

- Suppose that T is known in advance
- Pick learning rate $\delta = \sqrt{\ln(n)/T}$
  (assuming T is large enough so that $\delta \leq 1/2$!)

$$\mathbf{E}\left[M^{(T)}\right] \leq \min_{i \in [n]} m_i^{(T)} + \sqrt{\ln(n)/T} \cdot T + \sqrt{\ln(n) \cdot T}$$

$$= \min_{i \in [n]} m_i^{(T)} + 2 \cdot \sqrt{T \ln(n)}$$

Additive error (“regret”) negligible in most cases compared to $\min_{i \in [n]} m_i^{(T)}$!

Can we do better than that?

A Tight Lower Bound

Corollary

For $\delta = \sqrt{\ln(n)/T}$, the expected number of our mistakes $M^{(T)}$ satisfies

$$\mathbf{E}\left[M^{(T)} \right] \leq \min_{i\in[n]} m_i^{(T)} + 2\cdot\sqrt{T\ln(n)}.$$

- Suppose every expert $i = 1, 2, \ldots, n$ flips an unbiased coin, and the result is also an unbiased coin flip (independent of the experts' predictions)
- $\Rightarrow$ Regardless of our algorithm, the number of our mistakes satisfies

$$\mathbf{E}\left[M^{(T)} \right] = T \cdot \frac{1}{2}$$

- How good is the best expert?
  - Every expert $i \in [n]$ will make $T/2 \pm \Theta(\sqrt{T})$ many mistakes
  - Best expert will make $T/2 - \Theta(\sqrt{T\ln(n)})$ many mistakes (proof omitted, is based on central limit theorem)

- This demonstrates tightness of the error term
- Best expert will be good just by chance!

Outline

Choosing the Learning Rate Dynamically (1/2)

Question: How to adjust learning rate if T is not known in advance?

Approach 1: "The Doubling Trick"

- Algorithm:
  1. For $m = 1, 2, \ldots$
  2. Run a new instance of algorithm on the 2^m rounds $t = 2^m, \ldots, 2^{m+1} - 1$ with "optimal" learning rate (for an algorithm that runs for 2^m steps)
- Analysis before shows that in phase m, number of additional mistakes compared to best expert (regret) is at most $\alpha \cdot \sqrt{2^m}$

$\Rightarrow$ total regret after T steps is at most

$$\frac{\sqrt{2}}{\sqrt{2}-1}\alpha\sqrt{T},$$

where $\alpha = 2 \cdot \sqrt{\ln(n)}$.

Choosing the Learning Rate Dynamically (2/2)

Approach 2

- Algorithm:
  1. Run the randomised WMA with learning rate $\delta_t \approx 1/\sqrt{t}$ in round t
- A modification of our analysis proves for any time-interval $[T/2, T]$:

$$\sum_{t=T/2}^{T} \delta_t \cdot \lambda^{(t)} \cdot p^{(t)} \leq \log(\Phi^{(T/2)}) + (1-\delta_T)^{m_i^{[T/2,T]}}$$

$$\Rightarrow \qquad \mathbf{E}\left[M^{[T/2,T]} \right] \leq \frac{m_i^{[T/2,T]} \cdot \log(1-\delta_T)}{\delta_{T/2}} + \frac{\log(\Phi^{(T/2)})}{\delta_{T/2}}$$

Approach 3: "Self-Confident Algorithm"

- Algorithm:
  1. Run the randomised WMA with learning rate $\delta_t \approx 1/\sqrt{\min_{i\in[n]} m_i^{(t)}}$ (or $1/\sqrt{M^{(t)}}$) in round t

A More General Setting

New Setup

- At each step, we pick one expert i randomly out of n experts
- That expert i and our algorithm incur a cost $m_i^{(t)}$, but we also observe the costs of all experts (a vector $(m_j^{(t)})_{i=1}^n$)
- costs $m_j^{(t)}$ can be arbitrary in the range $[-1, 1]$

Coming back to our example of **stock prediction**:

- could define cost $m_j^{(t)} = 0$ if expert j is neutral (HOLD)
- cost $m_j^{(t)} > 0$ if expert j makes the wrong prediction (closer to 1 the stronger prediction and stronger the price change)
- cost $m_j^{(t)} < 0$ if expert j makes the correct prediction

Idea of the "Multiplicative Weights-Algorithm"

- In the first iteration, simply pick an expert uniformly at random
- Every expert will be penalised or rewarded through a multiplicative weight-update

The Multiplicative Weights Algorithm

The Multiplicative Weights Algorithm

Initialization: Fix $\delta \leq 1/2$. For every $i \in [n]$, let $w_i^{(1)} := 1$

Update: For $t = 1, 2, \ldots, T$:

- Choose expert i with prop. proportional to $w_i^{(t)}$.
- Observe the costs of all n experts in round t, $m^{(t)}$
- For every expert i, update its weight by:

$$w_i^{(t+1)} = (1 - \delta m_i^{(t)}) w_i^{(t)}$$

Analysis

For any expert i, the expected cost of this algorithm is at most

$$\sum_{t=1}^{T} m_i^{(t)} + \delta \cdot \sum_{t=1}^{T} \left| m_i^{(t)} \right| + \frac{\log n}{\delta}.$$

Derivation is very similar to the ones shown before.

Conclusions

Summary

- Weighted Majority Algorithm
  - natural, simple (and deterministic) algorithm
  - good performance, but could be a factor of 2 worse than the best expert
- Randomised Weighted Majority Algorithm
  - Randomised extension
  - almost optimal performance thanks to randomisation which guards against tailored worst-case instances (cmp. Quick-Sort!)
  - impact of the learning rate: small learning rate gives very good performance guarantees. However, actual performance may depend on the specific data set at hand (cf. simulations!)
- Multiplicative Weight-Update Algorithm
  - further generalisation of the (randomised) weighted majority algorithm

Outlook

- These algorithms are examples of the Ensemble-Method: Framework combining weak predictions into a strong learner
- A closely related algorithmic approach: Follow the Perturbed Leader
- Similar update schemes are Perceptron and AdaBoost

References

- S. Arora, E. Hazan and S. Kale
  The Multiplicative Weights Update Method: A Meta-Algorithm and Applications
  Theory of Computing, Volume 8 (2012).
- N. Littlestone and M.K. Warmuth
  The Weighted Majority Algorithm
  Information and Computation, Volume 108, Issue 2, 1994.
- S. Shalev-Shwartz and S. Ben-David
  Understanding Machine Learning: From Theory to Algorithms
  Cambridge University Press, 2014.
  https://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/understanding-machine-learning-theory-algorithms.pdf