 Recap from last time

- Last time: PAC model and Occam’s razor.
  - If data set has \( m \geq \left( \frac{1}{\varepsilon} \right) \left[ s \ln(2) + \ln(1/\delta) \right] \) examples, then whp any consistent hypothesis with size(h) < s has err(h) < \( \varepsilon \).
  - Equivalently, suffices to have \( s \leq (cm \ln(1/\delta))/\ln(2) \)
  - “compression = learning”
- Occam bounds = any class is learnable if computation time is no object.

Online learning

- What if we don’t want to make assumption that data is coming from some fixed distribution? Or any assumptions at all?
- Can no longer talk about past performance predicting future results.
- Can we hope to say anything interesting??

Idea: mistake bounds & regret bounds.

Mistake-bound model

- View learning as a sequence of stages.
- In each stage, algorithm is given \( x \), asked to predict \( f(x) \), and then is told correct value.
- Make no assumptions about order of examples.
- Goal is to bound total number of mistakes.

Simple example: disjunctions

- Suppose features are boolean: \( X = \{0,1\}^n \).
- Target is an OR function, like \( x_3 \lor x_9 \lor x_{12} \).
- Can we find an on-line strategy that makes at most \( n \) mistakes?
- Sure.
  - Start with \( h(x) = x_1 \lor x_2 \lor ... \lor x_n \)
  - Invariant: \( \{\text{vars in } h\} \supseteq \{\text{vars in } f\} \)
  - Mistake on negative: throw out vars in \( h \) set to 1 in \( x \). Maintains invariant and decreases \( |h| \) by 1.
  - No mistakes on positives. So at most \( n \) mistakes total.
**Simple example: disjunctions**

- Algorithm makes at most $n$ mistakes.
- No deterministic alg can do better:
  
  $$
  \begin{align*}
  1000000 &= \text{+ or -} \\
  0100000 &= \text{+ or -} \\
  0010000 &= \text{+ or -} \\
  0001000 &= \text{+ or -} \\
  \ldots
  \end{align*}
  $$

**MB model properties**

An alg $A$ is "conservative" if it only changes its state when it makes a mistake.

Claim: if $C$ is learnable by a deterministic alg with mistake bound $M$, then it is learnable by a conservative alg with mistake bound $M$.

Why?

- Take generic alg $A$. Create new conservative $A'$ by running $A$, but rewinding state if no mistake is made.
- Still $\leq M$ mistakes because $A$ still sees a legal sequence of examples.

**MB learnable $\Rightarrow$ PAC learnable**

Say alg $A$ learns $C$ with mistake-bound $M$.

Transformation 1:

- Run (conservative) $A$ until it produces a hyp $h$ that survives $\geq (1/e)\ln(M/\delta)$ examples.
- $\Pr(\text{fooled by any given bad } h) \leq \delta/M$.
- $\Pr(\text{fooled ever}) \leq \delta$.

  Uses at most $\frac{M}{e} \ln \left( \frac{M}{\delta} \right)$ examples total.

- Fancier method gets $O\left( \frac{1}{e} \left[ M + \ln \left( \frac{1}{\delta} \right) \right] \right)$.

**One more example...**

- Say we view each example as an integer between 0 and $2^n-1$.
- $C = \{[0,a] : a < 2^n\}$. (device fails if gets too hot)
- In PAC model we could just pick any consistent hypothesis. Does this work in MB model?
- What would work?

**What can we do with unbounded computation time?**

- "Halving algorithm": take majority vote over all consistent $h \in C$. Makes at most $\lg(|C|)$ mistakes.

- What if $C$ has functions of different sizes?
  - For any (prefix-free) representation, can make at most 1 mistake per bit of target.
    - Give each $h$ a weight of $\left( \frac{1}{2} \right)^{\text{size}(h)}$
    - Total sum of weights $\leq 1$.
    - Take weighted vote. Each mistake removes at least $\frac{1}{2}$ of total weight left.

**What can we do with unbounded computation time?**

- "Halving algorithm": take majority vote over all consistent $h \in C$. Makes at most $\lg(|C|)$ mistakes.

- What if we had a "prior" $p$ over fns in $C$?
  - Weight the vote according to $p$. Make at most $\lg(1/p)$ mistakes, where $f$ is target fn.

- What if $f$ was really chosen according to $p$?
  - Expected number of mistakes $\leq \Sigma_{f} [p[f] \lg(1/p[f])] = \text{entropy of distribution } p$. 
What if there is no perfect function?

Think of as \( h \in C \) as “experts” giving advice to you. Want to do nearly as well as best of them in hindsight.

These are called “regret bounds”.

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

We’ll look at a strategy whose running time is \( O(|C|) \). So, only computationally efficient when \( C \) is small.

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.,

<table>
<thead>
<tr>
<th>Expert 1</th>
<th>Expert 2</th>
<th>Expert 3</th>
<th>Neighbor's dog</th>
<th>Truth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Down</td>
<td>Up</td>
<td>Up</td>
<td>Up</td>
<td>Down</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Can we do nearly as well as best in hindsight?

[*”expert” : someone with an opinion. Not necessarily someone who knows anything.]

Using “expert” advice

If one expert is perfect, can get \( \leq \log(n) \) mistakes with halving alg.

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 \( \log(n)[\text{OPT}+1] \) mistakes, where \( \text{OPT} \) is #mistakes of the best expert in hindsight.

Seems wasteful. Constantly forgetting what we’ve “learned”. Can we do better?

Weighted Majority Algorithm

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

Weighted Majority Alg:

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

Weights: \( \frac{1}{2} \) \( \frac{1}{2} \) \( \frac{1}{2} \) \( 1 \)

Predictions: \( U \) \( U \) \( U \) \( D \)

We predict: \( U \)

Weights: \( \frac{1}{4} \) \( \frac{1}{4} \) \( \frac{1}{4} \) \( 1 \)

Truth: \( D \)

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,

\[
(1/2)^m \leq n(3/4)^M \\
(4/3)^M \leq n^{2^m} \\
M \leq 2.4(m + \log n)
\]

Randomized Weighted Majority

\( 2.4(m + \log n) \) 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, generalize \( \frac{1}{2} \) to \( 1 - \epsilon \).

Solves to:

\[
M \leq \frac{-m \ln(1 - \epsilon) + \ln(n)}{\epsilon} \approx (1 + \epsilon/2)m + \frac{1}{\epsilon} \ln(n)
\]

\( M \approx \text{expected} \) # mistakes

\( M \leq 1.35n + 2 \ln n \quad \epsilon = 1/2 \)

\( M \leq 1.14n + 4 \ln n \quad \epsilon = 1/4 \)

\( M \leq 1.07n + 8 \ln n \quad \epsilon = 1/8 \)

Unlike most worst-case bounds, numbers are pretty good.
**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.
- $W_{\text{final}} = n(1-\varepsilon F_t)(1 - \varepsilon F_t)$.
- $\ln(W_{\text{final}}) = \ln(n) + \sum [\ln(1 - \varepsilon F_t)] \leq \ln(n) - \varepsilon \sum F_t$, (using $\ln(1-x) \leq -x$)
- If best expert makes $m$ mistakes, then $\ln(W_{\text{final}}) > \ln(1-\varepsilon)^m$.
- Now solve: $\ln(n) - \varepsilon M > m \ln(1-\varepsilon)$.  
  $$M \leq \frac{-m \ln(1-\varepsilon) + \ln(n)}{\varepsilon} \approx (1 + \varepsilon/2)m + \frac{1}{\varepsilon} \log(n)$$

**Summarizing**

- $E[\# mistakes] \leq (1+\varepsilon)OPT + \varepsilon^3 \log(n)$  
  $= OPT + (\varepsilon OPT + \varepsilon^3 \log(n))$
- If set $\varepsilon = (\log(n)/OPT)^{1/2}$ to balance the two terms out (or use guess-and-double), get bound of $M \leq OPT + 2(OPT \cdot \log n)^{1/2}$.  
  $$\frac{M}{T} \leq \frac{OPT}{T} + O\left(\frac{\log(n)}{\sqrt{T}}\right)$$
  Regret term goes to 0 or better as $T \to \infty$ = "no-regret" algorithm.

**Extensions**

- What if experts are actions? (rows in a matrix game, ways to drive to work, …)
- At each time $t$, each has a loss (cost) in [0,1).
- Can still run the algorithm
  - Rather than viewing as "pick a prediction with prob proportional to its weight",  
  - View as "pick an expert with probability proportional to its weight"  
  - Alg pays expected cost $\bar{p}_t \cdot \bar{c}_t = F_t$.
- Same analysis applies.  
  Do nearly as well as best action in hindsight!

**RWM (multiplicative weights alg)**

- (1-εc^2)(1-εc^2)  
  (1-εc^2)(1-εc^2)  
  (1-εc^2)(1-εc^2)  
  .  
  .  
  .  
  (1-εc^2)(1-εc^2)  

<table>
<thead>
<tr>
<th>World</th>
<th>Life</th>
<th>Opponent</th>
</tr>
</thead>
<tbody>
<tr>
<td>c^2</td>
<td>c^2</td>
<td>scaling so costs in [0,1]</td>
</tr>
</tbody>
</table>

Guarantee: do nearly as well as fixed row in hindsight

Which implies doing nearly as well (or better) than minimax optimal

**Connections to minimax optimality**

- (1-εc^2)(1-εc^2)  
  (1-εc^2)(1-εc^2)  
  (1-εc^2)(1-εc^2)  
  .  
  .  
  .  
  (1-εc^2)(1-εc^2)

<table>
<thead>
<tr>
<th>World</th>
<th>Life</th>
<th>Opponent</th>
</tr>
</thead>
<tbody>
<tr>
<td>c^2</td>
<td>c^2</td>
<td>scaling so costs in [0,1]</td>
</tr>
</tbody>
</table>

If play RWM against a best-response oracle, $\tilde{p}$ will approach minimax optimality.

(If if didn’t, wouldn’t be getting promised guarantee)
Connections to minimax optimality

If play two RWM against each other, then empirical distributions must be near-minimax-optimal.

(Else, one or the other could & would take advantage)