On-Line Learning with Expert Advice
On-Line Learning

- No distributional assumption.
- Worst-case analysis (adversarial).
- Mixed training and test.
- Performance measure: mistake model, regret.
Weather Forecast

Can you come up with your own?

- **objective**: accurate predictions.
- **means**: no meteorological expertise.

wunderground.com weather.com cnn.com bbc.com you
Many Similar Problems

- Route selection (internet, traffic).
- Games (chess, backgammon).
- Stock value prediction.
- Decision making.
Problem

- **Set-up:**
  - $N \geq 1$ experts.
  - prediction set: $\{ \text{\ding{55}}, \text{\ding{54}} \}$.
  - at each time $t \in [1, T]$,
    - receive experts’ predictions.
    - make prediction.

- **Question:** suppose one expert is always correct over $[1, T]$ (in hindsight). Can you design a forecaster making only a small number of mistakes?
Forecasting Algorithm

**Strategy:**
- at each time step predict based on majority vote.
- eliminate wrong experts.
Forecasting Algorithm

Analysis: let $W^m$ be the total number of experts after $m$ mistakes.

- Initially, $W^0 = N$.
- After each mistake: $W^m \leq W^{m-1}/2$.
- Thus, $W^m \leq W^{m-1}/2 \leq (W^{m-2}/2)/2 \leq \cdots \leq W_0/2^m$.
- Since $1 \leq W^m$ (at least one expert is right),
  
  $1 \leq W_0/2^m = N/2^m$
  
  $\iff 2^m \leq N$
  
  $\iff m \log 2 \leq \log N$
  
  $\iff m \leq \log_2 N.$
Halving Algorithm

see (Mitchell, 1997)

$$\text{HALVING}(H)$$

$1 \quad H_1 \leftarrow H$

$2 \quad \textbf{for } t \leftarrow 1 \textbf{ to } T \textbf{ do}$

$3 \quad \text{RECEIVE}(x_t)$

$4 \quad \hat{y}_t \leftarrow \text{MAJORITYVOTE}(H_t, x_t)$

$5 \quad \text{RECEIVE}(y_t)$

$6 \quad \textbf{if } \hat{y}_t \neq y_t \textbf{ then}$

$7 \quad H_{t+1} \leftarrow \{c \in H_t : c(x_t) = y_t\}$

$8 \quad \textbf{return } H_{T+1}$
Application

For $N = 128 = 2^7$,

$$m = |\text{wrong forecasts}| \leq 7.$$

For $N = 1,048,576 = 2^{20}$,

$$m = |\text{wrong forecasts}| \leq 20.$$
Problem

Question:

• suppose now that no expert is exactly correct.
• some expert is the best in hindsight.
• can you design a forecaster making only a small number of mistakes more than that expert?
Forecasting Algorithm

Strategy:
- assign some weight/confidence to each expert.
- predict based on weighted majority.
- shrink weight of wrong experts.

<table>
<thead>
<tr>
<th>experts</th>
<th>prediction</th>
<th>actual</th>
</tr>
</thead>
<tbody>
<tr>
<td>.5</td>
<td>.5</td>
<td>.5</td>
</tr>
<tr>
<td>.5</td>
<td>.25</td>
<td>.25</td>
</tr>
<tr>
<td>.5</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>.5</td>
<td>.25</td>
<td>.25</td>
</tr>
<tr>
<td>.5</td>
<td>1.0</td>
<td>.5</td>
</tr>
<tr>
<td>.5</td>
<td>.125</td>
<td>.125</td>
</tr>
<tr>
<td>.25</td>
<td>1.0</td>
<td>.5</td>
</tr>
</tbody>
</table>
Weighted Majority Algorithm

**Algorithm**: prediction with $N \geq 1$ experts.

- at any time $t$, expert $i$ has weight $w_i^t$.
- originally, $w_i^0 = 1, \forall i \in [1, N]$.
- prediction according to weighted majority.
- weight of each wrong expert updated, $\epsilon \in (0, 1)$, via
  \[
  w_i^{t+1} \leftarrow w_i^t (1 - \epsilon).
  \]
Notation

- Mistakes:
  - $m^t_i$: number of mistakes made by expert $i$ till time $t$.
  - $m^t$: number of mistakes made by algorithm.
Weighted Majority - Analysis

- **Potential:** \( \Phi^t = \sum_{i=1}^{N} w_i^t \).

- **Upper bound:** after each mistake,
  
  - more than half of the weight, \( \Phi^t / 2 \), was on experts that turned out to be wrong.

\[
\Phi^{t+1} \leq \Phi^t / 2 + \Phi^t / 2 \times (1 - \epsilon) \\
= \Phi^t - \epsilon / 2 \times \Phi^t \\
= (1 - \epsilon / 2) \Phi^t.
\]

Thus, \( \Phi^t \leq (1 - \epsilon / 2)^m N \).
Weighted Majority - Analysis

- **Lower bound:** for any expert $i$, 
  \[ \Phi^t \geq w^t_i = (1 - \epsilon)^{m^t_i}. \]

- **Comparison:**

  \[
  (1 - \epsilon)^{m^t_i} \leq (1 - \epsilon/2)^{m^t N}
  \Rightarrow m^t_i \log(1 - \epsilon) \leq \log N + m^t \log(1 - \epsilon/2).
  \]
Using the identities:

\[-(x + x^2) \leq \log(1 - x) \leq -x,\]

\[m_i^t \log(1 - \epsilon) \leq \log N + m^t \log(1 - \epsilon/2)\]

\[\Rightarrow -m_i^t(\epsilon + \epsilon^2) \leq \log N - m^t \epsilon/2\]

\[\Rightarrow m^t \epsilon/2 \leq \log N + m_i^t(\epsilon + \epsilon^2)\]

\[\Rightarrow m^t \leq 2 \frac{\log N}{\epsilon} + 2(1 + \epsilon)m_i^t.\]
Theorem (mistake bound): let $m^t_i$ be the number of mistakes made by expert $i$ till time $t$ and $m^t$ the total number of mistakes. Then, for all $t$ and for any expert $i$ (in particular best expert),

$$m^t \leq \frac{2 \log N}{\epsilon} + 2(1 + \epsilon)m^t_i.$$

- Thus, $m^t \leq O(\log N) + \text{constant} \times \text{best expert}$.

- Realizable case: $m^t \leq O(\log N)$. 
Weighted Majority Algorithm

(Littlestone and Warmuth, 1988)

Weighted-Majority($N$ experts) \implies y_t, y_{t,i} \in \{0, 1\}.
\epsilon \in [0, 1).

\begin{verbatim}
1 for i ← 1 to N do
2    w_{1,i} ← 1
3 for t ← 1 to T do
4    RECEIVE($x_t$)
5    \hat{y}_t ← 1 \sum_{i=1}^{N} w_t y_{t,i} \geq \frac{1}{2} \implies \text{weighted majority vote}
6    RECEIVE($y_t$)
7    if \hat{y}_t \neq y_t then
8        for i ← 1 to N do
9            if (y_{t,i} \neq y_t) then
10               w_{t+1,i} ← (1 - \epsilon) w_t,i
11            else w_{t+1,i} ← w_t,i
12 return w_{T+1}
\end{verbatim}
Regret

Definition: the regret at time $T$ is the difference between the loss incurred up to $T$ by the algorithm and that of the best expert in hindsight:

$$R_T = L_T - L_T^{\text{min}}.$$ 

• for best regret minimization algorithms:

$$R_T \leq O(\sqrt{T \log N}).$$
Weighted Majority - Regret

- Observe that:
  \[ m^T \leq \frac{2 \log N}{\epsilon} + 2(1 + \epsilon)m^*_T \leq \frac{2 \log N}{\epsilon} + 2\epsilon T + 2m^*_T. \]

- If \( T \) known in advance, best value of \( \epsilon = \min\{\sqrt{(\log N)/T}, 1/2\} \).
  
  Thus, \( m^T \leq 4\sqrt{T \log N} + 2m^*_T \).

- Poor regret guarantee:
  \[ R_T \leq 4\sqrt{T \log N} + m^*_T. \]
Zero-One Loss

No deterministic algorithm can achieve $R_T = o(T)$:

- for any algorithm, choose $y_t$ adversarially, then
  \[ L_T = T. \]
- let $N = 2$ with constant experts 0 and 1. Then,
  \[ L_T^{\min} \leq T/2 \]
- Thus, $R_T = L_T - L_T^{\min} \geq T/2$.

randomization.
Convex Losses

- **Loss property**: $L$ convex in its first argument and taking values in $[0, 1]$.

- **Algorithm**: extension of Weighted Majority.
  - **weight update**: $w_{t+1,i} \leftarrow w_{t,i} e^{-\eta L(\hat{y}_{t,i}, y_t)} = e^{-\eta L_{t,i}}$.
  - **prediction**: $\hat{y}_t = \frac{\sum_{i=1}^N w_{t,i} y_{t,i}}{\sum_{i=1}^N w_{t,i}}$.

- **Guarantee**: for any $\eta > 0$, $R_T \leq \frac{\log N}{\eta} + \frac{\eta T}{8}$.
  
  For $\eta = \sqrt{8 \log N/T}$,
  
  $\text{Regret}(T) \leq \sqrt{(T/2) \log N}$.
Conclusion

On-line learning, regret minimization:
• rich branch of machine learning.
• connections with game theory.
• simple and minimal assumptions.
• algorithms easy to implement.
• scale to very large data sets.