The Weighted Majority Algorithm

Amrudin Agovic
1/24/2006
Machine Learning
Outline

- The Halving Algorithm/ Motivation for WM
- WM
- WML (shifting target)
- WMI (infinite pool)
- WMG (generalized)
- WMC (continuous)
- WMR (randomized)
- Using Pools of Functions
- Dealing with Anomalies
The Halving Algorithm

Assumptions:
- Suppose we have binary valued labels \( \{0, 1\} \)
  and \( n \) algorithms, which make binary predictions.
- One out of \( n \) algorithms in pool of algorithms is always correct.

Algorithm:
- Choose the label predicted by the majority.
- Each time a mistake is made, throw out the algorithms with the incorrect prediction.
The Halving Algorithm

- The number of mistakes made is bounded above by $\log_2(n)$

**Problem with the algorithm:**
- Not very robust to anomalies in the data.
- If a data point is encountered that cannot be predicted by any algorithm in the pool, all algorithms are thrown out.
Weighted Majority Algorithm

- Associate weight $0 \leq w \leq 1$ with each algorithm in pool

Algorithm:
- Initialize weights to 1
- Take label with largest overall weight
- When a mistake is made, multiply the weights of incorrect algorithms with $\beta$, where $0 \leq \beta \leq 1$
WM Mistake Bounds

- The best algorithm makes $m$ mistakes: WM makes at most $O(\log(n)+m)$ mistakes.
- Each subset of $k$ algorithms makes at most $m$ mistakes: WM makes at most $O(\log(n/k) + m)$ mistakes.
- All algorithms in a subset of $k$ algorithms make together at most $m$ mistakes: WM makes at most $O(\log(n/k) + m/k)$ mistakes.
Suppose that we had a shifting target function. There is no one algorithm that performs best everywhere. Instead we have several algorithms which perform well on certain parts of the data. Each time the subpool of best performing algorithms changes WM would make a lot of mistakes. WML is designed to do a better job in this case.
WML – Shifting Target

- Initialize weights to at least $\beta \gamma / n \cdot W_{\text{total\_initial}}$ with $0 < \gamma \leq 0.5$

- Take prediction with largest overall weight

- If mistake is made, update weight $w_i$ only if it is larger than $\gamma / n \cdot W_{\text{total}}$

Note:

- For $\gamma = 0$ we have the WM algorithm
- The use of $\gamma$ does not allow the overall differences in weight to become too large.
- When the correct subpool of algorithms changes, this results in less mistakes in comparison to WM.
WML – Shifting Algorithm

- The number of mistakes made by WML are bounded above by:

\[
\sum_{i=1}^{k} \min \left\{ \frac{l_i \cdot \frac{\log(n/\beta y) + m_i \log(1/\beta)}{\log(1/((1+\beta/2) + (1-\beta) y))}} {l_i} \right\}
\]

where:
- \(m_i\) is the most number of mistakes made by the best algorithm in the ith subsequence
- \(l_i\) is the number of trails in the ith subsequence
Assume:

- We have an infinite pool of algorithms.
- We always work with a set of $l$ active algorithms.
- $W$ is a computable function that we can use to initialize weights.
- $W'$ is a computable function such that:

$$W'(i) \geq \sum_{j=i}^{\infty} W(j) \quad \text{and} \quad \lim_{i \to \infty} W'(i) = 0$$
1) Until \( W'(l+1) \leq \frac{((1+\beta)/2)^{m+1} W'(1)}{(1-\beta)(m+1)(m+2)} \), increase \( l \)

2) Compute the total weights \( q_0 \) and \( q_1 \)

3) if \( q_0 > q_1 + \frac{((1+\beta)/2)^{m+1} W'(1)}{(1-\beta)(m+1)(m+2)} \) predict 0

   if \( q_1 > q_0 + \frac{((1+\beta)/2)^{m+1} W'(1)}{(1-\beta)(m+1)(m+2)} \) predict 1

   otherwise predict either 1 or 0

4) Multiply weights of incorrect algorithms in active pool by \( \beta \)
WMI - infinite pool

- The mistake bound for WMI is given by:

\[
\inf_{i \geq 1} \left\{ \frac{\log \left( W'(1)/W(i) \right) + m_i \log(1/\beta) + \log(2)}{\log(2/(1+\beta))} \right\}
\]

- There is a trade off in choosing initial weights: mistake bound vs. number of active pool members
WMG, WMC, WMR

- **WMG** makes binary predictions. Pool algorithms output is in [0,1]. Updates weights in every trial, or only when mistakes happen.
- **WMC** makes predictions in [0,1]. Output of pool algorithms is in [0,1]. It updates weights in every trial.
- **WMR** is a randomized version. It makes binary predictions. Output of pool algorithms is in [0,1].
Let

- $x_i$ be the prediction of the $i^{th}$ algorithm
- $W_{\text{total}}$ be the sum over all weights
- Let $\gamma$ be defined as:

$$\gamma = \frac{\sum_{i=1}^{n} w_i x_i}{W_{\text{total}}}$$

**WMG - prediction**

- If $\gamma > 1/2$ predict 1
- If $\gamma < 1/2$ predict 0
- Otherwise predict either 0 or 1
WMG, WMC, WMR - prediction

WMC prediction
• predict $\gamma$

WMR prediction
• Choose member of pool at random
• Make same prediction as chosen pool member with probability $\gamma$
WMG, WMC, WMR - update

- Let $\rho^{(j)}$ be the actual label in trial $j$
- Let $x_i^{(j)}$ be the output of the $i$th algorithm in trial $j$
- The weight is updated as follows:
  \[ w_i^{(j+1)} = F w_i^{(j)} \]
- where:
  \[ \beta |x_i^{(j)} - \rho^{(j)}| \leq F \leq 1 - (1 - \beta) |x_i^{(j)} - \rho^{(j)}| \]
Mistake Bounds

- Let $m_i$ be the total loss of the $i^{th}$ algorithm.
- Let $\lambda^{(i)}$ be the prediction of the master algorithm.

WMG:

\[
\log(n) + m_i \log \left( \frac{1}{\beta} \right)
\]

\[
\log \left( \frac{2}{1 + \beta} \right)
\]

WMC:

\[
\ln(n) + m_i \ln \left( \frac{1}{\beta} \right)
\]

\[
\frac{1}{1 - \beta}
\]

WMR:

\[
\log(n) + E(m_i) \log \left( \frac{1}{\beta} \right)
\]

\[
\frac{1}{1 - \beta}
\]

where: $E(\lambda^{(i)}|(x^{(1)},\rho^{(1)}), \ldots,(x^{(i)},\rho^{(j)})) = \gamma^{(i)}$ with probability 1.
Consider now pools of functions rather than pools of algorithms.

For shattered functions the Weighted Majority Algorithm can be shown to be optimal, provided that the weights are initialized properly.

A set of \{0,1\}-valued functions \{f_1(x),..,f_n(x)\} with domain X is shattered by X, if \( f_1(x),..,f_n(x) \) range over all of \{0,1\}^n, as x ranges over X.
Using Pool Of Functions

- Let $f_1(x), \ldots, f_n(x)$ be a pool of functions with range $\{0,1\}$
- Let $M_1, \ldots, M_2$ be non-negative integers such that
  \[ \sum_{i=1}^{n} 2^{-M_i} \]
- Initialize weights to $w_j = 2^{-M_j}$ and $\beta = 0$
- Trials consistent with a function $f_i$

- Assuming the above the algorithm makes at most $M_i$ mistakes.
Anomalies

- For both WM and WMR the rate at which the mistake bound grows with the number of anomalies can be kept arbitrarily close to the best possible rate by choosing the appropriate $\beta$
  - For WM the rate is greater than two
  - For WMR the rate is slightly bigger than one
Conclusions (positive)

- The authors have shown that the weighted majority algorithm can be applied to a wide range of settings.
- Each algorithm is presented with a proven bound on prediction accuracy.
- Almost all bounds are derived using a similar approach. This makes it easier to develop new versions of the algorithm and find the bounds.
Conclusions (negative)

- Algorithms are presented in paragraph format, which is not very readable.
- Some of the proofs are missing quite a bit of detail and not straightforward to follow.
- There are a lot of proofs in the paper but no empirical illustration of the results. It would have been nice if the authors had some sample data where one could compare the algorithms in terms of actual mistakes made.
- Some issues not addressed: for example in the infinite pool case, how does one decide which algorithm to add next to the pool.