Winnow

Goal

Understand the celebrated winnow algorithm for online binary classification.

Alert 1.40: Convention

Gray boxes are not required hence can be omitted for unenthusiastic readers.

This note is likely to be updated again soon.

Alert 1.41: Notation

Recall that $\Delta_{p-1} := \{\mathbf{w} \in \mathbb{R}^p : \mathbf{w} \geq \mathbf{0}, \mathbf{1}^\top \mathbf{w} = 1\}$ is the standard simplex. We denote $\mathrm{int}(\mathbf{A})$ as the interior of a set $\mathbf{A} \subseteq \mathbb{R}^p$, i.e., the largest open set contained in $\mathbf{A}$. In particular, $\mathrm{int}\,\Delta_{p-1} = \{\mathbf{w} \in \mathbb{R}^p : \mathbf{w} > \mathbf{0}, \mathbf{1}^\top \mathbf{w} = 1\}$.

We use $\mathbf{c} = \mathbf{a} \odot \mathbf{b}$ for the Hadamard elementwise product, i.e., $c_i = a_i b_i$ for all i. All familiar algebraic operations, when applied to a vector or matrix, are understood in the elementwise manner (unless mentioned otherwise).

Algorithm 1.42: Winnow

The perceptron algorithm employs the additive update rule $\mathbf{w} \leftarrow \mathbf{w} + \mathbf{a}$. It turns out that there is a multiplicative counterpart $\mathbf{w} \propto \mathbf{w} \odot \exp(\eta \mathbf{a})$, known as winnow:

Algorithm: The Winnow algorithm (Littlestone 1988)

Input: $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_n] \in \mathbb{R}^{\mathsf{p} \times \mathsf{n}}$, threshold $\delta \geq 0$, step size $\eta > 0$, initialize $\mathbf{w} \in \mathrm{int}\Delta_{\mathsf{p}-1}$

Output: approximate solution $\mathbf{w}$

for $k = 1, 2, \ldots$ **do**

  receive training example index $I_k \in \{1, \ldots, \mathsf{n}\}$ // the index I_k can be random

  if $\langle \mathbf{a}_{I_k}, \mathbf{w} \rangle \leq \delta$ **then**

    $\mathbf{w} \leftarrow \mathbf{w} \odot \exp(\eta \mathbf{a}_{I_k})$ // update only when making a mistake

    $\mathbf{w} \leftarrow \mathbf{w} / \|\mathbf{w}\|_1$ // normalize

Littlestone, N. (1988). "Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm". *Machine Learning*, vol. 2, pp. 285–318.

Alert 1.43: What is zero remains zero

Note that if we initialize $w_i = 0$ for some coordinate i, then w_i will remain 0 the entire time in the winnow algorithm. This is typical for *multiplicative* algorithms where we need to initialize $\mathbf{w}$ with strictly positive numbers. The downside is that, if w_i is indeed 0 at a solution, then winnow can only get there in the limit.

Definition 1.44: KL divergence

For two probability vectors $\mathbf{p}, \mathbf{q} \in \Delta_{\mathsf{p}-1}$, we define their KL divergence as:

$$\mathsf{KL}(\mathbf{p} \| \mathbf{q}) := \sum_j p_j \log \frac{p_j}{q_j},$$

where we adopt the convention $0 \log 0/0 = 0$. It can be shown that

- $\mathsf{KL}(\mathbf{p}\|\mathbf{q}) \neq \mathsf{KL}(\mathbf{q}\|\mathbf{p})$ in general, hence KL is not a distance metric;
- $\mathsf{KL}(\mathbf{p}\|\mathbf{q}) \geq 0$ with equality iff $\mathbf{p} = \mathbf{q}$;
- Pinsker's inequality: $\mathsf{KL}(\mathbf{p}\|\mathbf{q}) \geq \frac{1}{2}\|\mathbf{p}-\mathbf{q}\|_1^2$. (Or, using words from convex analysis, the KL divergence is strongly convex w.r.t. the ℓ_1 norm.)

Theorem 1.45: Convergence guarantee of winnow (Littlestone 1988)

Assuming the dataset $\mathbf{A}$ *is (strictly) linearly separable w.r.t. a nonnegative weight vector* $\mathbf{w}^\star$ *and denoting* $\mathbf{w}_t$ *the iterate after the t-th update in the winnow algorithm. Then,* $\mathbf{w}_t \to$ *some* $\mathbf{w}^* > \mathbf{0}$ *in finite time. If each column of A is selected indefinitely, then* $\mathbf{A}^\top \mathbf{w}^* > \delta \mathbf{1}$.

Proof: The proof is similar to that of the perceptron algorithm, but we use the KL divergence (instead of the squared Euclidean distance) to measure the progress of the winnow algorithm. Under the linearly separable assumption, there exists some $\mathbf{w}^\star \in \Delta_{\mathsf{p}-1}$ such that $\mathbf{A}^\top \mathbf{w}^\star \geq s\mathbf{1} > \mathbf{0}$. Slightly perturb $\mathbf{w}^\star$ we can assume w.l.o.g. that $\mathbf{w}^\star > \mathbf{0}$, i.e. $\mathbf{w}^\star \in \mathrm{int}(\Delta_{\mathsf{p}-1})$. Then, upon making an update from $\mathbf{w}_t$ to $\mathbf{w}_{t+1}$ (using the data instance denoted as $\mathbf{a}$):

$$\begin{aligned}
\mathsf{KL}(\mathbf{w}^\star\|\mathbf{w}_{t+1}) - \mathsf{KL}(\mathbf{w}^\star\|\mathbf{w}_t) &= \sum_j \mathsf{w}_j^\star \log \frac{\mathsf{w}_{jt}}{\mathsf{w}_{j,t+1}} \\
&= \sum_j \mathsf{w}_j^\star \log \frac{\|\mathbf{w}_t \odot \exp(\eta \mathbf{a})\|_1}{\exp(\eta \mathsf{a}_j)} \\
&= \log \|\mathbf{w}_t \odot \exp(\eta \mathbf{a})\|_1 - \eta \langle \mathbf{a}, \mathbf{w}^\star \rangle
\end{aligned}$$

Let $\|\mathbf{A}\|_{\infty,\infty} := \max_{j,i} |\mathsf{a}_{ji}|$. Using Jensen's inequality for the convex function exp:

$$\begin{aligned}
\|\mathbf{w}_t \odot \exp(\eta \mathbf{a})\|_1 &= \sum_j \mathsf{w}_{jt} \exp(\eta \mathsf{a}_j) \\
&\leq \sum_j \mathsf{w}_{jt} \Big[\frac{1 + \mathsf{a}_j/\|\mathbf{A}\|_{\infty,\infty}}{2} \exp(\eta \|\mathbf{A}\|_{\infty,\infty}) + \frac{1 - \mathsf{a}_j/\|\mathbf{A}\|_{\infty,\infty}}{2} \exp(-\eta \|\mathbf{A}\|_{\infty,\infty}) \Big] \\
&= \frac{\exp(\eta\|\mathbf{A}\|_{\infty,\infty}) + \exp(-\eta\|\mathbf{A}\|_{\infty,\infty})}{2} + \frac{\langle \mathbf{a}, \mathbf{w}_t \rangle (\exp(\eta\|\mathbf{A}\|_{\infty,\infty}) - \exp(-\eta\|\mathbf{A}\|_{\infty,\infty}))}{2\|\mathbf{A}\|_{\infty,\infty}} \\
&\leq \frac{\exp(\eta\|\mathbf{A}\|_{\infty,\infty}) + \exp(-\eta\|\mathbf{A}\|_{\infty,\infty})}{2} + \frac{\delta (\exp(\eta\|\mathbf{A}\|_{\infty,\infty}) - \exp(-\eta\|\mathbf{A}\|_{\infty,\infty}))}{2\|\mathbf{A}\|_{\infty,\infty}} \\
&= \beta \exp(\eta\|\mathbf{A}\|_{\infty,\infty}) + (1-\beta) \exp(-\eta\|\mathbf{A}\|_{\infty,\infty}),
\end{aligned}$$

where $\beta = \frac{\|\mathbf{A}\|_{\infty,\infty} + \delta}{2\|\mathbf{A}\|_{\infty,\infty}}$.

Thus, $0 \leq \mathsf{KL}(\mathbf{w}^\star\|\mathbf{w}_t) \leq \mathsf{KL}(\mathbf{w}^\star\|\mathbf{w}_0) + t\Big[\log\Big(\beta \exp(\eta\|\mathbf{A}\|_{\infty,\infty}) + (1-\beta)\exp(-\eta\|\mathbf{A}\|_{\infty,\infty})\Big) - \eta s\Big]$, i.e.

$$t \leq \frac{\mathsf{KL}(\mathbf{w}^\star\|\mathbf{w}_0)}{\eta s - \log[\beta \exp(\eta\|\mathbf{A}\|_{\infty,\infty}) + (1-\beta)\exp(-\eta\|\mathbf{A}\|_{\infty,\infty})]}$$

So winnow performs only a finite number of updates, and the theorem follows. ■

If $\delta = 0, \mathbf{w}_0 = \frac{1}{\mathsf{p}}\mathbf{1}$, and set $\eta = \frac{1}{2\|\mathbf{A}\|_{\infty,\infty}} \log \frac{\|\mathbf{A}\|_{\infty,\infty} + s}{\|\mathbf{A}\|_{\infty,\infty} - s}$, then we can simplify the bound as:

$$\begin{aligned}
t &\leq \frac{\log \mathsf{p}}{\frac{1+s/\|\mathbf{A}\|_{\infty,\infty}}{2} \log \frac{1+s/\|\mathbf{A}\|_{\infty,\infty}}{2} + \frac{1-s/\|\mathbf{A}\|_{\infty,\infty}}{2} \log \frac{1-s/\|\mathbf{A}\|_{\infty,\infty}}{2} - \log \frac{1}{2}} \\
&= \frac{\log \mathsf{p}}{\mathsf{KL}\left(\binom{q}{1-q} \| \binom{1/2}{1/2}\right)}, \qquad \text{where} \quad q = \frac{1+s/\|\mathbf{A}\|_{\infty,\infty}}{2}
\end{aligned}$$

$$\leq \frac{2\|\mathbf{A}\|_{\infty,\infty}^2 \log \mathsf{p}}{s^2},$$

where the last inequality follows from Pinsker's inequality.

Again, we can optimize the "fictional" parameter s:

$$\max_{(\mathbf{w},s):\mathbf{A}^\top\mathbf{w}\geq s\mathbf{1},\mathbf{w}\in\Delta_{\mathsf{p}-1}} s = \underbrace{\max_{\mathbf{w}\in\Delta_{\mathsf{p}-1}} \min_i \langle \mathbf{a}_i, \mathbf{w}\rangle}_{\ell_1 \text{ margin } \gamma_1} \leq \min_i \max_{\mathbf{w}\in\Delta_{\mathsf{p}-1}} \langle \mathbf{a}_i, \mathbf{w}\rangle \leq \|\mathbf{A}\|_{\infty,\infty}. \tag{1.7}$$

By setting $\eta = \frac{1}{2\|\mathbf{A}\|_{\infty,\infty}} \log \frac{\|\mathbf{A}\|_{\infty,\infty}+\gamma_1}{\|\mathbf{A}\|_{\infty,\infty}-\gamma_1}$ we obtain

$$t \leq T_1 := T_1(\mathbf{A}) = \frac{2\|\mathbf{A}\|_{\infty,\infty}^2 \log \mathsf{p}}{\gamma_1^2}.$$

Littlestone, N. (1988). "Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm". *Machine Learning*, vol. 2, pp. 285–318.

Remark 1.46: Sparse SVM

As before, we can in fact try to find a weight vector $\mathbf{w}$ that attains the margin bound in (1.7):

$$\max_{\mathbf{w}\in\Delta_{\mathsf{p}-1}} \min_i \langle \mathbf{a}_i, \mathbf{w}\rangle \quad\equiv\quad \min_{\mathbf{A}^\top\mathbf{w}\geq\mathbf{1},\mathbf{w}\geq\mathbf{0}} \|\mathbf{w}\|_1,$$

which is essentially the (hard-margin) sparse SVM (if we drop the nonnegative constraint on $\mathbf{w}$).

Remark 1.47: The duplication trick

The linear separable condition in Theorem 1.45 appears to be stronger than the one in Theorem 1.28, due to the nonnegative constraint. This slight restriction can be easily remedied by the duplication trick:

replace each $\mathbf{a} \in \mathbf{A}$ with $[\mathbf{a}; -\mathbf{a}]$.

Because, $\langle \mathbf{a}, \mathbf{w}\rangle = \langle \mathbf{a}, \mathbf{w}^+ - \mathbf{w}^-\rangle = \langle \mathbf{a}, \mathbf{w}^+\rangle + \langle -\mathbf{a}, \mathbf{w}^-\rangle = \langle [\mathbf{a}; -\mathbf{a}], [\mathbf{w}^+; \mathbf{w}^-]\rangle$. Thus, any margin γ continues to hold under the nonnegative constraint, if we double the dimension of our data (which really is a mild overhead).

Remark 1.48: Winnowing irrelevant features

Comparing the bounds of perceptron and winnow, we see that the latter incurs an additional mild $2\log\mathsf{p}$ factor and replaces the constant $\|\mathbf{A}\|_{2,\infty}$ with $\|\mathbf{A}\|_{\infty,\infty}$ (the margin parameter γ is also different). In high dimensions, $\|\mathbf{A}\|_{2,\infty} \gg \|\mathbf{A}\|_{\infty,\infty}$ hence the winnow algorithm is more suitable when there are lots of irrelevant features (which do not affect the margin or $\|\mathbf{A}\|_{\infty,\infty}$ much but may affect $\|\mathbf{A}\|_{2,\infty}$ significantly).