Perceptron and Winnow
This Lecture

On-Line linear classification: two algorithms.

- Perceptron algorithm.
- Winnow algorithm.
Definition: a linear classifier is an algorithm that returns a hypothesis of the form

\[ x \mapsto \text{sgn}(w \cdot x + b), \]

with \( w \in \mathbb{R}^N, b \in \mathbb{R}. \)
Margin Definitions

Definition: the (geometric) margin of a point $x$ with label $y$ for a linear classifier $h: x \mapsto w \cdot x + b$ is its algebraic distance to the hyperplane $w \cdot x + b = 0$,

$$\rho(x) = \frac{y(w \cdot x + b)}{||w||}.$$

Definition: the margin of a linear classifier $h$ for a sample $S = (x_1, \ldots, x_m)$ is the minimum margin of the points in that sample:

$$\rho = \min_{1 \leq i \leq m} \frac{y_i(w \cdot x_i + b)}{||w||}.$$

Perceptron Algorithm

(Rosenblatt, 1958)

\[ \text{PERCEPTRON}(w_0) \]

1. \( w_1 \leftarrow w_0 \) \quad \triangleright \text{typically } w_0 = 0 \\
2. \text{for } t \leftarrow 1 \text{ to } T \text{ do} \\
3. \quad \text{RECEIVE}(x_t) \\
4. \quad \hat{y}_t \leftarrow \text{sgn}(w_t \cdot x_t) \\
5. \quad \text{RECEIVE}(y_t) \\
6. \quad \text{if } (\hat{y}_t \neq y_t) \text{ then} \\
7. \quad \quad w_{t+1} \leftarrow w_t + y_t x_t \quad \triangleright \text{more generally } \eta y_t x_t, \eta > 0 \\
8. \quad \text{else } w_{t+1} \leftarrow w_t \\
9. \text{return } w_{T+1}
Perceptron - Notes

- Update: if \( y_t(w_t \cdot x_t) < 0 \), then

\[
y_t(w_{t+1} \cdot x_t) = y_t(w_t \cdot x_t) + \eta \lVert x_t \rVert^2 \geq 0
\]

change in the desired direction.

- Different modes of applications:
  - repeated passes over sample of size \( m \) drawn according to some distribution \( D \).
  - infinite sample drawn according to \( D \).
  - no distributional assumption.
Separating Hyperplane

- Margin and errors

\[ \mathbf{w} \cdot \mathbf{x} = 0 \]
Perceptron — Stochastic Gradient Descent

- **Objective function**: convex but not differentiable.

\[
F(w) = \frac{1}{T} \sum_{t=1}^{T} \max \left( 0, -y_t (w \cdot x_t) \right) = \mathbb{E}_{x \sim \hat{D}} [f(w, x)]
\]

with \( f(w, x) = \max \left( 0, -y(w \cdot x) \right) \).

- **Stochastic gradient**: for each \( x_t \), the update is

\[
w_{t+1} \leftarrow \begin{cases} 
  w_t - \eta \nabla_w f(w_t, x_t) & \text{if differentiable} \\
  w_t & \text{otherwise},
\end{cases}
\]

where \( \eta > 0 \) is a learning rate parameter.

- **Here**: \( w_{t+1} \leftarrow \begin{cases} 
  w_t + \eta y_t x_t & \text{if } y_t (w_t \cdot x_t) < 0 \\
  w_t & \text{otherwise}.
\end{cases} \)
Perceptron Algorithm - Bound

(Novikoff, 1962)

Theorem: Assume that $\|x_t\| \leq R$ for all $t \in [1, T]$ and that for some $\rho > 0$ and $v \in \mathbb{R}^N$, for all $t \in [1, T]$,

$$\rho \leq \frac{y_t (v \cdot x_t)}{\|v\|}.$$

Then, the number of mistakes made by the perceptron algorithm is bounded by $\frac{R^2}{\rho^2}$.

Proof: Let $I$ be the set of $t$s at which there is an update and let $M$ be the total number of updates.
Summing up the assumption inequalities gives:

\[
M \rho \leq \frac{v \cdot \sum_{t \in I} y_t x_t}{\|v\|} \\
= \frac{v \cdot \sum_{t \in I} (w_{t+1} - w_t)}{\|v\|} \quad \text{(definition of updates)} \\
= \frac{v \cdot w_{T+1}}{\|v\|} \\
\leq \|w_{T+1}\| \quad \text{(Cauchy-Schwarz ineq.)} \\
= \|w_{t_m} + y_{t_m} x_{t_m}\| \quad (t_m \text{ largest } t \text{ in } I) \\
= \left[ \|w_{t_m}\|^2 + \|x_{t_m}\|^2 + 2y_{t_m} w_{t_m} \cdot x_{t_m} \right]^{1/2} \\
\leq \left[ \|w_{t_m}\|^2 + R^2 \right]^{1/2} \leq 0 \\
\leq \left[ MR^2 \right]^{1/2} = \sqrt{MR}. \quad \text{(applying the same to previous } t \text{s in } I)}
Notes:

- bound independent of dimension and tight.
- convergence can be slow for small margin, it can be in $\Omega(2^N)$.
- among the many variants: **voted perceptron algorithm**. Predict according to

$$\text{sgn} \left( \left( \sum_{t \in I} c_t w_t \right) \cdot x \right),$$

where $c_t$ is the number of iterations $w_t$ survives.

- $\{x_t : t \in I\}$ are the support vectors for the perceptron algorithm.

- non-separable case: **does not converge**.
Leave-One-Out Error

Definition: let $h_S$ be the hypothesis output by learning algorithm $L$ after receiving sample $S$ of size $m$. Then, the leave-one-out error of $L$ over $S$ is:

$$
\hat{R}_{\text{loo}}(L) = \frac{1}{m} \sum_{i=1}^{m} 1_{h_{S-\{x_i\}}(x_i) \neq f(x_i)}.
$$

Property: unbiased estimate of expected error of hypothesis trained on sample of size $m-1$,

$$
\mathbb{E}_{S \sim D^m} [\hat{R}_{\text{loo}}(L)] = \frac{1}{m} \sum_{i=1}^{m} \mathbb{E}_{S} [1_{h_{S-\{x_i\}}(x_i) \neq f(x_i)}] = \mathbb{E}_{S} [1_{h_{S-\{x\}}(x) \neq f(x)}]
$$

$$
= \mathbb{E}_{S' \sim D^{m-1}} \left[ \mathbb{E}_{x \sim D} [1_{h_{S'}(x) \neq f(x)}] \right] = \mathbb{E}_{S' \sim D^{m-1}} [R(h_{S'})].
$$
Perceptron - Leave-One-Out Analysis

**Theorem:** Assume that the data is separable. Let $h_S$ be the hypothesis returned by the Perceptron algorithm after training on sample $S \sim D^{m+1}$ (repeated passes) and let $M(S)$ be the number of updates made and let $R(h_S)$ be the error of $h_S$. Then,

$$
\mathbb{E}_{S \sim D^m}[R(h_S)] \leq \mathbb{E}_{S \sim D^{m+1}} \left[ \frac{\min(M(S), R_{m+1}^2/\rho_{m+1}^2)}{m+1} \right].
$$

**Proof:** Let $x$ be a point in sample $S$. Then, if $h_{S-\{x\}}$ misclassifies $x$, there must have been an update at $x$ during training to obtain $h_S$. Thus,

$$
\hat{R}_{\text{loo}}(\text{perceptron}) \leq \frac{M(S)}{m+1}.
$$
Dual Perceptron Algorithm

\textbf{Dual-Perceptron}(\(\alpha_0\))

\begin{enumerate}
\item \(\alpha_1 \leftarrow \alpha_0\) \hspace{0.5em} \text{\texttt{\small{\triangleright typically } \alpha_0 = 0}}
\item \text{\texttt{for } } t \leftarrow 1 \text{ \texttt{to } } T \text{ \texttt{do}}
\item \text{\texttt{Receive(} } \mathbf{x}_t \text{\texttt{)}}
\item \(\hat{y}_t \leftarrow \text{sgn}\left( \sum_{s=1}^{T} \alpha_s y_s (\mathbf{x}_s \cdot \mathbf{x}_t) \right)\)
\item \text{\texttt{Receive(} } y_t \text{\texttt{)}}
\item \text{\texttt{if } } (\hat{y}_t \neq y_t) \text{ \texttt{then}}
\item \(\alpha_{t+1} \leftarrow \alpha_t + 1\)
\item \text{\texttt{else } } \alpha_{t+1} \leftarrow \alpha_t
\item \text{\texttt{return } } \alpha
\end{enumerate}
Kernel Perceptron Algorithm

\( K \) PDS kernel.

**Kernel-Perceptron** \((\alpha_0)\)

1. \( \alpha_1 \leftarrow \alpha_0 \quad \triangleright \text{typically } \alpha_0 = 0 \)
2. \hspace{1em} \textbf{for } t \leftarrow 1 \textbf{ to } T \textbf{ do}
3. \hspace{2em} \text{RECEIVE}(x_t)
4. \hspace{2em} \hat{y}_t \leftarrow \text{sgn}(\sum_{s=1}^{T} \alpha_s y_s K(x_s, x_t))
5. \hspace{2em} \text{RECEIVE}(y_t)
6. \hspace{2em} \textbf{if } (\hat{y}_t \neq y_t) \textbf{ then}
7. \hspace{3em} \alpha_{t+1} \leftarrow \alpha_t + 1
8. \hspace{2em} \textbf{else } \alpha_{t+1} \leftarrow \alpha_t
9. \hspace{1em} \textbf{return } \alpha
XOR Problem

- Use second-degree polynomial kernel with $c = 1$:

- Linearly non-separable

- Linearly separable by $x_1 x_2 = 0$. 

![Diagram of XOR problem with second-degree polynomial kernel]
Non-Separable Case
(Freund and Schapire, 1998)

Theorem: Let \( v \) be any vector with \( \| v \| = 1 \) and let \( \rho > 0 \). Define the deviation of \( x_t \) by:

\[
d_t = \max\{0, \rho - y_t(v \cdot x_t)\},
\]

and let \( D = \sqrt{\sum_{t=1}^{T} d_t^2} \). Then, the number of perceptron updates after processing \( x_1, \ldots, x_T \) is bounded by

\[
\left[ \frac{R + D}{\rho} \right]^2.
\]
• **Proof:** Reduce problem to separable case in higher dimension.

• **Mapping (similar to trivial mapping):**

$$(N + t)^{th} \text{ component}$$

$$\mathbf{x}_t = \begin{bmatrix} x_{t,1} \\ \vdots \\ x_{t,N} \end{bmatrix} \rightarrow \mathbf{x}_t' = \begin{bmatrix} x_{t,1} \\ \vdots \\ x_{t,N} \\ 0 \\ \Delta \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$\mathbf{v} \rightarrow \mathbf{v}' = \begin{bmatrix} v_1/Z \\ \vdots \\ v_N/Z \\ y_1d_1/(\Delta Z) \\ \vdots \\ y_Td_T/(\Delta Z) \end{bmatrix}$$

$$\|\mathbf{v}'\| = 1 \Rightarrow Z = \sqrt{1 + \frac{D^2}{\Delta^2}}.$$
• Now, \( y_t(v' \cdot x'_t) = y_t \left( \frac{v \cdot x_t}{Z} + \Delta \frac{y_t d_t}{Z \Delta} \right) \)

\[
= \frac{y_t v \cdot x_t}{Z} + \frac{d_t}{Z} \\
\geq \frac{y_t v \cdot x_t}{Z} + \frac{\rho - y_t(v \cdot x_t)}{Z} = \frac{\rho}{Z}.
\]

• Since \( \|x'_t\|^2 \leq R^2 + \Delta^2 \), the bound of the separable case applies:

\[
\frac{(R^2 + \Delta^2)(1 + D^2 / \Delta^2)}{\rho^2}.
\]

• With \( \Delta = \sqrt{RD} \), this bound is minimized and equal to:

\[
\frac{(R+D)^2}{\rho^2}.
\]

• Predictions made by the perceptron in the higher-dimension coincide with those of the perceptron in the original space.
Winnow Algorithm

\textsc{Winnow}(\eta)

1. \( w_1 \leftarrow 1/N \)
2. \textbf{for} \( t \leftarrow 1 \) \textbf{to} \( T \) \textbf{do}
3. \hspace{0.5cm} \textbf{Receive}(x_t)
4. \hspace{0.5cm} \hat{y}_t \leftarrow \text{sgn}(w_t \cdot x_t)
5. \hspace{0.5cm} \textbf{Receive}(y_t)
6. \hspace{0.5cm} \textbf{if} (\hat{y}_t \neq y_t) \textbf{then}
    7. \hspace{1cm} Z_t \leftarrow \sum_{i=1}^{N} w_{t,i} \exp(\eta y_t x_{t,i})
    8. \hspace{1cm} \textbf{for} \ i \leftarrow 1 \ \textbf{to} \ N \ \textbf{do}
        9. \hspace{1.5cm} w_{t+1,i} \leftarrow \frac{w_{t,i} \exp(\eta y_t x_{t,i})}{Z_t}
    10. \ \textbf{else} \ w_{t+1} \leftarrow w_t
11. \textbf{return} \ w_{T+1}

\textbf{(Littlestone, 1988)}
Winnow - Notes

- Winnow = weighted majority:
  - for $y_{t,i} = x_{t,i} \in \{-1, +1\}$, $\text{sgn}(w_t \cdot x_t)$ coincides with the majority vote.
  - multiplying by $e^\eta$ or $e^{-\eta}$ the weight of correct or incorrect experts, is equivalent to multiplying by $\beta = e^{-2\eta}$ the weight of incorrect ones.

- Relationships with other algorithms: e.g., boosting and Perceptron (Winnow and Perceptron can be viewed as special instances of a general family).

- Motivation: large number of irrelevant features.
Winnow Algorithm - Bound

**Theorem:** Assume that $\|x_t\|_\infty \leq R_\infty$ for all $t \in [1, T]$ and that for some $\rho_\infty > 0$ and $v \in \mathbb{R}^N, v \geq 0$ for all $t \in [1, T],$

$$\rho_\infty \leq \frac{y_t (v \cdot x_t)}{\|v\|_1}.$$

Then, the number of mistakes made by the Winnow algorithm is bounded by $2 \left( \frac{R_\infty^2}{\rho_\infty^2} \right) \log N.$

**Proof:** Let $I$ be the set of $t$s at which there is an update and let $M$ be the total number of updates.
Winnow Algorithm - Bound

- **Potential:** \( \Phi_t = \sum_{i=1}^{N} \frac{v_i}{\| v \|} \log \frac{v_i}{\| v \| \cdot w_{t,i}}. \) (relative entropy)

- **Upper bound:** for each \( t \) in \( I \),

\[
\Phi_{t+1} - \Phi_t = \sum_{i=1}^{N} \frac{v_i}{\| v \|} \log \frac{w_{t,i}}{w_{t+1,i}} \\
= \sum_{i=1}^{N} \frac{v_i}{\| v \|} \log \frac{Z_t}{\exp(\eta y_t x_{t,i})} \\
= \log Z_t - \eta \sum_{i=1}^{N} \frac{v_i}{\| v \|} y_t x_{t,i} \\
\leq \log \left[ \sum_{i=1}^{N} w_{t,i} \exp(\eta y_t x_{t,i}) \right] - \eta \rho_\infty \\
= \log \mathbb{E}_{w_t} \left[ \exp(\eta y_t x_t) \right] - \eta \rho_\infty \\
\text{(Hoeffding)} \leq \log \left[ \exp(\eta^2 (2R_\infty)^2 / 8) \right] + \eta y_t w_t \cdot x_t - \eta \rho_\infty \\
\leq \eta^2 R_\infty^2 / 2 - \eta \rho_\infty.
\]
Winnow Algorithm - Bound

- **Upper bound**: summing up the inequalities yields
  \[ \Phi_{T+1} - \Phi_1 \leq M \left( \eta^2 R^2_{\infty} / 2 - \eta \rho_{\infty} \right). \]

- **Lower bound**: note that
  \[
  \Phi_1 = \sum_{i=1}^{N} \frac{v_i}{\|v\|_1} \log \frac{v_i/\|v\|_1}{1/N} = \log N + \sum_{i=1}^{N} \frac{v_i}{\|v\|_1} \log \frac{v_i}{\|v\|_1} \leq \log N
  \]
  and for all \( t \), \( \Phi_t \geq 0 \) (property or relative entropy).

Thus, \( \Phi_{T+1} - \Phi_1 \geq 0 - \log N = -\log N. \)

- **Comparison**: \( -\log N \leq M \left( \eta^2 R^2_{\infty} / 2 - \eta \rho_{\infty} \right). \) For \( \eta = \frac{\rho_{\infty}}{R^2_{\infty}} \)
  we obtain
  \[ M \leq 2 \log N \frac{R^2_{\infty}}{\rho^2_{\infty}}. \]
Comparison with perceptron bound:

- dual norms: norms for $x_t$ and $v$.
- similar bounds with different norms.
- each advantageous in different cases:
  - Winnow bound favorable when a sparse set of experts can predict well. For example, if $v = e_1$ and $x_t \in \{\pm 1\}^N$, $\log N \text{ vs } N$.
  - Perceptron favorable in opposite situation.