PERCEPTRON ALGORITHM

Perceptron algorithm is used in "linear seperable" problem, meaning problem where we can draw a line (or a plane) to classify the data

→ Pseudocode:

Perceptron (w_0, η)

$w_1 \leftarrow w_0$ (typically $w_0 = 0$)

for $t \leftarrow 1$ to T do:

Reiceive (x_t)

$\hat{y}_t \leftarrow \text{sign}(w_t \cdot x_t)$ → dot product $w_t^T x_t$

- $w_t^T x_t > 0$: upper label
- $w_t^T x_t \leq 0$: lower label

Reiceive (y_t) → the label

if $\hat{y}_t \neq y_t$ then:

$w_{t+1} \leftarrow w_t + \eta \cdot y_t \cdot x_t$

else:

$w_{t+1} \leftarrow w_t$

return w_{T+1}

→ Why updating w leads to better classfier ?

- Assume vector x_t is misclassified, meaning the dot product $y_t(w_t^T x_t) < 0$. Our goal is to make this dot product positive
- By updating current weight w_t with $\eta \cdot y_t \cdot x_t$, we "nudge" the weight so that the misclassified x_t will be correct in the future: $w_{t+1} = w_t + \eta \cdot y_t x_t$
- Mathematically, the new dot product will be:

$$
\begin{aligned}
& y_t (w_{t+1}^T x_t) \\
= \; & y_t (w_t + \eta \cdot y_t x_t)^T x_t \\
= \; & y_t w_t^T x_t + \eta y_t^2 \|x_t\|^2 \\
= \; & y_t w_t^T x_t + \eta \|x_t\|^2
\end{aligned}
$$

< $y_t^2 = 1$ since $y_t = \begin{cases} 1 & \text{if positive label} \\ -1 & \text{otherwise} \end{cases}$ >

→ $\eta \|x_t\|^2$ is alway positive, meaning the dot product will be one step closer to being positive.

the Perceptron algorithm seeks a weight vector w that minimize an objective function F (definition: Objective Function is a function that quantifies total error in the training set, which includes but not equal to empirical error)

Objective function $F(w)$:

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

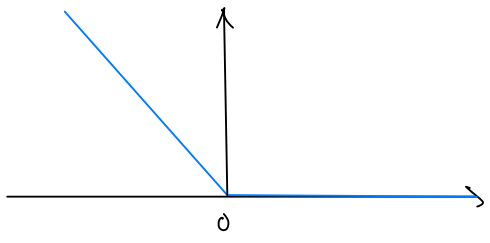
- $-y_t(w^T x_t)$: magnitude of the error at t^{th} example
- $\frac{1}{T}\sum_{t=1}^{T} \max(0, -y_t(w^T x_t))$: average error magnitude over the training set T.
- $\mathop{E}_{x \sim \hat{D}}\left[\tilde{F}(w, x)\right]$: expected value of function $\tilde{F}$ over all possible input x of empirical distribution $\hat{D}$, where: $\tilde{F}(w, x) = \max(0, f(x)(w.x))$

Function $F(w)$ is convex, therefore Perceptron algorithm concides with application of the stochastic subgradient descent technique to F.

Function $F(w)$ is non-differentiable, specifically at points where $(w^T x) = 0$

Visualizing $\tilde{F}(w, x)$ and $F(w)$:

$\tilde{F}(w, x)$:



$F(w)$: multiple lines leads to a flat surface of 0, each line represent component function $\tilde{F}(w, x)$

Stochatic Subgradient Descent

This technique examines one example x_t at a time, and updates the weight w based of the result of function $\tilde{F}(., x_t)$ mentioned above.

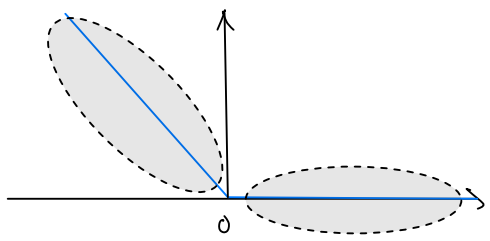
$$w_{t+1} \begin{cases} w_t - \eta \nabla_w \tilde{F}(., x_t) & \text{if } w_t^T x_t \neq 0 \\ w_t + \eta y_t x_t & \text{otherwise} \end{cases}$$

How the "subgradient" is calculated ? And also the proof

The subgradient can generally be understood as "the slope". There are 2 cases:

- <u>When $w_t^T x_t \neq 0$</u> $\Rightarrow$ $\tilde{F}(., x_t) = \max(0, -y_t(w_t^T x_t))$ is differentiable.

Visually, point x_t is on this regions:



So we can calculate "subgradient" by differentiating, $\nabla_w \tilde{F}(w_t, x_t)$, in details:

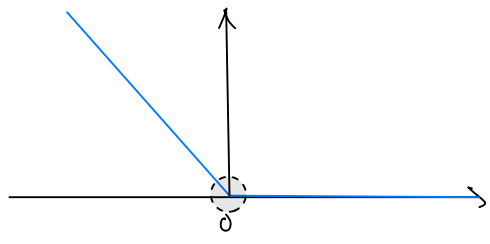
$$\nabla_w \tilde{F}(w_t, x_t) = \nabla_w -y_t(w_t^T x_t) = \begin{cases} -y_t x_t & \text{if } -y_t(w_t^T x_t) > 0 \\ 0 & \text{if } -y_t(w_t^T x_t) < 0 \end{cases}$$

"gradient" means steepest ascent with respect to w

and since we are trying to minimize loss function $\tilde{F}(w_t, x_t)$, we subtract from w: $w_{t+1} = w_t - \eta \nabla_w \tilde{F}(w_t, x_t)$

- When $w_t^T x_t = 0 \Rightarrow \tilde{F}(\cdot, x_t) = \max(0, -y_t(w_t^T x_t))$ is non-differentiable.

Visually, point x_t is this breaking point



In this case, any vector on the line segment from 0 to $-y_t x_t$ is a valid subgradient.

Often, $-y_t x_t$ is chosen in this case because it provides a direct and robust method to adjust w to increase the classification margin.

This analysis of how "subgradient" is calculated leads to Stochastic Gradient Descent to becomes:

$$w_{t+1} \begin{cases} w_t + \eta y_t x_t & \text{if } y_t(w_t^T x_t) \leq 0 \\ w_t & \text{if } y_t(w_t^T x_t) > 0 \end{cases}$$

which is exactly how Perceptron updates its weights

Perceptron algorithm's performance

Now we try to upper-bound the mistakes/updates made by Perceptron algorithm when processing a sequence of T points that are linearly separable by a hyperplane with margin $\rho > 0$

Theorem 8.8 Upper-bound mistakes

Let $x_1, \ldots, x_T \in \mathbb{R}^N$ be sequence of T points with $\|x_t\| \leq r$ for all $t \in [T]$ and some $r > 0$

Assume that there exist $\rho > 0$ and $v \in \mathbb{R}^N$ such that $\rho \leq \frac{y_t(v^T x_t)}{\|v\|}, \forall t \in [T]$

Then, the number of mistakes/updates made by the Perceptron algorithm is bounded by:

$$r^2 / \rho^2$$

Proof:

- Observe that an update happens only when a mistake occurs.
- Let M be number of mistakes.
- Say w_t is our current weight and a mistake happens w.r.t (x_t, y_t), so:

$$w_{t+1} = w_t + y_t x_t \quad \langle \eta = 1 \text{ for simplicity} \rangle$$

Since $\|x_t\| \leq r$, we calculate norm of w_{t+1} to utilize this fact:

$$\|w_{t+1}\|^2 = \|w_t + y_t x_t\|^2$$
$$= (w_t + y_t x_t)^T (w_t + y_t x_t)$$
$$= \|w_t\|^2 + 2\underbrace{y_t (w_t^T x_t)}_{<0} + \underbrace{y_t^2}_{+1}\underbrace{\|x_t\|^2}_{\leq r^2}$$

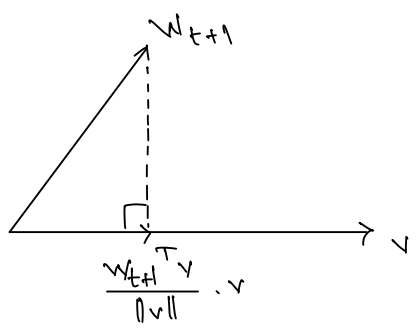
$$\Rightarrow \|w_{t+1}\|^2 \leq \|w_t\|^2 + r^2$$
$$= (\|w_{t-1}\|^2 + r^2) + r^2$$
$$= \underbrace{\|w_0\|^2}_{0} + M \cdot r^2$$
$$= 0 + M \cdot r^2$$

So $\boxed{\|w_{t+1}\|^2 \leq M \cdot r^2}$ ① This step analyze how large the magnitude of w_{t+1} changes

Since $\rho \leq \frac{y_t (v^T x_t)}{\|v\|}$, we dot product $w_{t+1}^T v$ to utilize this fact:

$$w_{t+1}^T v = (w_t + y_t x_t)^T v$$
$$= w_t^T v + \underbrace{y_t (x_t^T v)}_{\geq \rho \cdot \|v\|}$$

$$\Rightarrow w_{t+1}^T v \geq w_t^T v + \rho\|v\|$$
$$= (w_{t-1}^T v + \rho\|v\|) + \rho\|v\|$$
$$= \underbrace{w_0^T v}_{0} + M \cdot \rho \cdot \|v\|$$
$$= M \cdot \rho \|v\|$$



So $\boxed{\frac{w_{t+1}^T v}{\|v\|} \geq M \cdot \rho}$ ② This step analyze how much the direction of w_{t+1} changes

Now we relate ① and ② to find the bound for M:

$$M \cdot \rho \leq \frac{w_{t+1}^T v}{\|v\|} \quad \text{< from ② >}$$

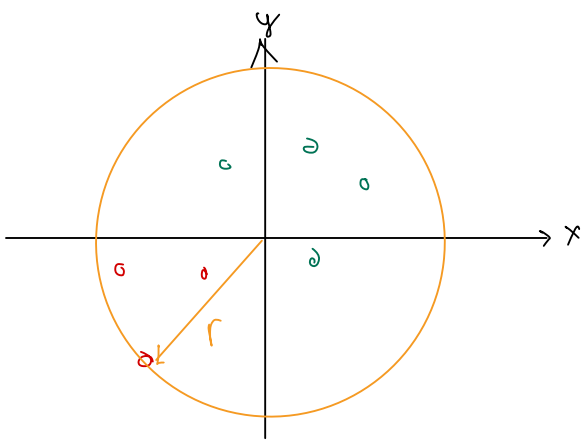
$$\Leftrightarrow M^2 \cdot \rho^2 \leq \frac{(w_{t+1}^T v)^2}{\|v\|^2}$$
$$\leq \frac{\|w_{t+1}\|^2 \cdot \|v\|^2}{\|v\|^2} \quad \text{< Cauchy-Schwartz >}$$
$$= \|w_{t+1}\|^2$$

$$\Rightarrow M^2 \cdot \rho^2 \leq \|w_{t+1}\|^2 \leq M \cdot r^2 \quad \text{< Combine with ① >}$$
$$\Rightarrow M^2 \cdot \rho^2 \leq M r^2$$
$$\Rightarrow \boxed{M \leq r^2/\rho^2}$$

↳ What do r and ρ represent ?

- Radius assumption: $\forall t \in [T]$, $\|x_t\|_2 \leq r$ for some $r > 0$

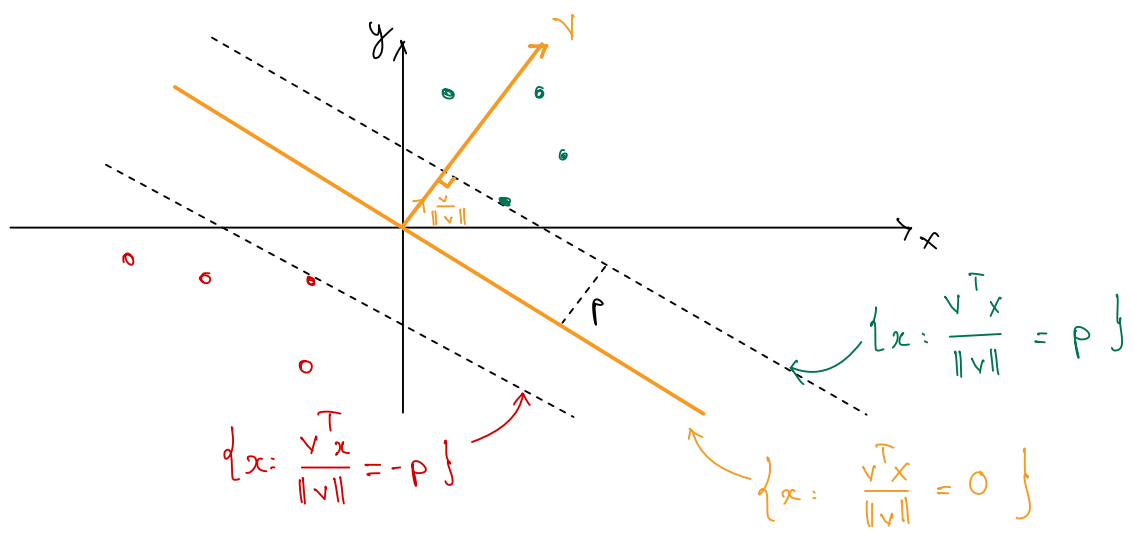
Assume the distance from 0 to any point x_t is lesser or equal to r, the distance to the furthest point.



- Margin assumption:

Assume a dataset $\{(x_1, y_1), \ldots, (x_T, y_T)\}$ is linearly separable by ρ-margin.

if: $\exists\ v \in \mathbb{R}^N$ s.t. $\frac{y_t (v^T x_t)}{\|v\|} \geq \rho$ $\forall\ t \in [T]$, and some $\rho > 0$



Theorem 8.9 Expected error bound

Assume that the data is linearly separable. Let h_S be the hypothesis returned by the Perceptron algorithm after training over a sample S of size m drawn from distribution D.

Then, the expected error of h_S is bounded as follows:

$$E_{S \sim D^m}[R(h_S)] \leq E_{S \sim D^{m+1}}\left[\frac{\min(M(S), r_S^2/\rho_S^2)}{m+1}\right]$$

↳ Proof:

Recall lemma 5.3 said:

" Expected value of leave-one-out error on sample size $m \geq 2$ equals expected value of true error on sample size $m-1$ ". In other words:

$$E_{S \sim D^m}[R_{LOO}(A)] = E_{S' \sim D^{m-1}}[R(h_{S'})]$$

Proof lemma 5.3:

indicator that left-out x_i is misclassified

$$E_{S\sim D^m}\left[R_{LOO}(A)\right] = \frac{1}{m}\sum_{i=1}^{m} E_{S\sim D^m}\left[1_{h_{S-\{x_i\}}(x_i)\neq y_i}\right] \quad \text{<linearity of expectation>}$$

$$= E_{S|_x\sim D^m}\left[1_{h_{S-\{x\}}(x)\neq y}\right]$$

$$R_{LOO}(A) = \frac{1}{m}\sum_{i=1}^{m} 1_{h_{S-\{x_i\}}(x_i)\neq y_i}$$

$$= \underset{S'\sim D^{m-1},\, x\sim D}{E}\left[1_{h_{S'}(x)\neq y}\right]$$

$$= \underset{S'\sim D^{m-1}}{E}\left[\underset{x\sim D}{E}\left[1_{h_{S'}(x)\neq y}\right]\right]$$

$$= \underset{S'\sim D^{m-1}}{E}\left[R(h_{S'})\right]$$

Back to proof for theorem 8.9

Let
- S be a linearly separable sample of size $m+1$ drawn i.i.d. from D.
- x be an example in S
- A be the Perceptron algorithm
- $M(S)$ is the number of mistakes / updates in sample S

From lemma 5.3, we can say that:

$$\underset{S\sim D^m}{E}\left[R(h_S)\right] = \underset{S\sim D^{m+1}}{E}\left[R_{LOO}(A)\right]$$

$$= \underset{S\sim D^{m+1}}{E}\left[\frac{1}{m+1}\sum_{i=1}^{m+1} 1_{h_{S-\{x_i\}}(x_i)\neq y_i}\right] \quad \text{< definition of } R_{LOO}(A)\text{ >}$$

$$\leq \underset{S\sim D^{m+1}}{E}\left[\frac{1}{m+1}\cdot M(S)\right] \quad \text{< \# mistakes in } S-\{x\} \leq \text{\# mistakes in } S\text{ >}$$

$$\leq \underset{S\sim D^{m+1}}{E}\left[\frac{1}{m+1}\cdot\frac{r_S^2}{\rho_S^2}\right] \quad \text{< theorem 8.8 >}$$

So we can conclude that:

$$\underset{S\sim D^m}{E}\left[R(h_S)\right] \leq \underset{S\sim D^{m+1}}{E}\left[\frac{\min\left(M(S),\ r_S^2/\rho_S^2\right)}{m+1}\right]$$

use min to address the potential discrepancy between theoretical upper bound and actual # mistakes

From "Understand machine learning" textbook, chapter 9:

In realizable case, it can be proven that Perceptron algorithm stops with all sample points correctly classified

Theorem 9.1 Upper-bound # of iterations

Assume training set S size m is separable.

let $\begin{cases} B = \min\{ \|w\| : \forall i, [m], \ y_i(w^T x_i) \geq \rho \} \text{ for some } \rho > 0 \\ R = \max_i \|x_i\| \end{cases}$

Then, the Perceptron algorithm stops after at most $\left(\frac{RB}{\rho}\right)^2$ iterations, and when it stops, it holds that $\forall i \in [m], \ y_i(w^{(t)^T} x_i) > 0$

→ Proof:

- This is very similar to proof for theorem 8.8 previously with a little "oops", that is the assumption $B = \min\{ \|w\| : \forall i, [m], \ y_i(w^T x_i) \geq \rho \}$, the vector w in this case is equals $\frac{v}{\|v\|}$ in theorem 8.8
- By definition of stopping criteria, Perceptron stops if all examples are separated.
  Let $\begin{cases} T \text{ be \# of iterations, our goal is to prove } T \leq (RB)^2 \\ w^* \text{ be the weight vector mentioned in assumption B. } \|w^*\| = B = 1 \end{cases}$
- The idea of the proof is to show that cosine angle of w^* and w^{T+1} is at least $\frac{\sqrt{T}}{RB}$, in other words:

$$\cos(w^*, w^{T+1}) = \frac{\langle w^*, w^{T+1} \rangle}{\|w^*\| . \|w^{T+1}\|} \geq \frac{\sqrt{T} . \rho}{RB}$$

$$\Rightarrow \quad 1 \geq \cos(w^*, w^{T+1}) \geq \frac{\sqrt{T}}{RB}$$

- First, we upper-bound $\|w^{T+1}\|$, similar to proof in theorem 8.8, it can be proven that: $\|w^{T+1}\| \leq \sqrt{T} . R$ (1)
- Second, we lower-bound $\langle w^*, w^{T+1} \rangle$, similar to proof in theorem 8.8, it can be proven that: $\langle w^*, w^{T+1} \rangle \geq T . \rho$ (2)
- From (1), (2), combine with the fact that $\|w^*\| = B$, the cosine becomes:

$$1 \geq \frac{\langle w^*, w^{T+1} \rangle}{\|w^*\| . \|w^{T+1}\|} \geq \frac{T . \rho}{B . \sqrt{T} R} = \frac{\sqrt{T} . \rho}{BR}$$

$$y(w^T x_i) \geq 1$$

$$y(v^T x_i) \geq \rho$$

$$\Rightarrow \quad T \leq \left(\frac{RB}{\rho}\right)^2$$