Linear Predictors + Perceptron

AW

Lecture Overview

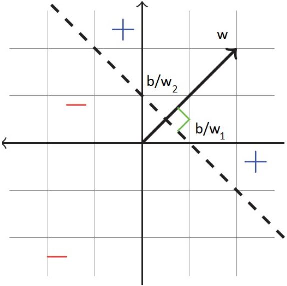
1. Linear Predictors

2. Perceptron

Halfspaces

The class of halfspace classifiers separate instances using a family of hyperplanes $\mathbb{L} = \{L_n\}_{n=1}^{\infty}$ where $L_n = \{H = [f : d] \mid f \equiv \overline{w} \in \mathbb{R}^n,\ d \in R\}$ to separate classes, i.e. hyperplane classifier $H = [f : d]$ where $f \equiv \overline{w} \in \mathbb{R}^n$ computes class y for an instance $\overline{x} \in \mathbb{R}^n$ as $y = \mathbf{sign}(\overline{w} \bullet \overline{x} - d)$.

Solving hyperplane equation $\overline{w} \bullet \overline{x} = d$ gives that hyperplane intercepts with i^{th} axis at $\frac{d}{w_i}$.



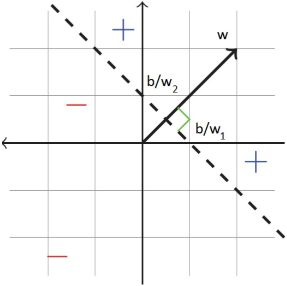
If we are looking for a linear predictor that is ERM predictor w.r.t realizable PAC learning case, then for a given sample set $S = \{(\overline{x}_1, y_1), \ldots, (\overline{x}_m, y_m)\}$ we need to find $\overline{w}$ and $d \in \mathbb{R}$ such that for every $i \in \{1, \ldots, m\}$ we have $\mathbf{sign}(\overline{w} \bullet \overline{x}_i - d) = y_i$. Equivalently $y_i \cdot (\overline{w} \bullet \overline{x}_i - d) > 0, \quad \forall i = 1, \ldots, m$

There must be a solution for the system of m inequalities $y_i \cdot (\overline{w} \bullet \overline{x}_i - d) > 0, \quad \forall i = 1, \ldots, m$ because we assume that it is realizable case. To solve we could use Linear programming, but inequalities are strict so LP is inapplicable.

Halfspaces

The class of halfspace classifiers separate instances using a family of hyperplanes $\mathbb{L} = \{L_n\}_{n=1}^{\infty}$ where $L_n = \{H = [f : d] \mid f \equiv \overline{w} \in \mathbb{R}^n,\ d \in R\}$ to separate classes, i.e. hyperplane classifier $H = [f : d]$ where $f \equiv \overline{w} \in \mathbb{R}^n$ computes class y for an instance $\overline{x} \in \mathbb{R}^n$ as $y = \mathbf{sign}(\overline{w} \bullet \overline{x} - d)$.

Solving hyperplane equation $\overline{w} \bullet \overline{x} = d$ gives that hyperplane intercepts with i^{th} axis at $\frac{d}{w_i}$.



If we are looking for a linear predictor that is ERM predictor w.r.t realizable PAC learning case, then for a given sample set $S = \{(\overline{x}_1, y_1), \ldots, (\overline{x}_m, y_m)\}$ we need to find $\overline{w}$ and $d \in \mathbb{R}$ such that for every $i \in \{1, \ldots, m\}$ we have $\mathbf{sign}(\overline{w} \bullet \overline{x}_i - d) = y_i$. Equivalently $y_i \cdot (\overline{w} \bullet \overline{x}_i - d) > 0, \quad \forall i = 1, \ldots, m$

To find equivalent system with $\geq$ constraints, suppose $\overline{w}^*, d^*$ is a solution. Let then $\gamma = \min_i(y_i(\overline{w}^* \bullet \overline{x}_i - d^*))$ and let $\overline{w}^\dagger = \frac{\overline{w}^*}{\gamma}$ and $d^\dagger = \frac{d^*}{\gamma}$. Then $\frac{1}{\gamma} y_i \cdot (\overline{w}^* \bullet \overline{x}_i - d^*) = y_i \cdot (\overline{w}^\dagger \bullet \overline{x}_i - d^\dagger) \geq 1, \quad \forall i = 1, \ldots, m$. Vector satisfying these conditions can be found by solving LP with dummy objective $(\min 0)$.

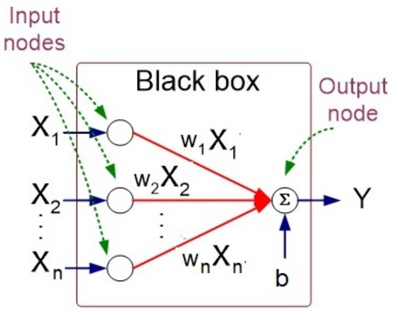
Lecture Overview

1. Linear Predictors

2. Perceptron

Perceptron Device

- The device is an assembly of inter-connected nodes,
- In the simplest variant an input node i does multiplication of an input by a constant w_i and outputs it,
- Output node sums values of its input and compares against threshold b. If the sum is less than b then it outputs 1, otherwise it outputs -1.

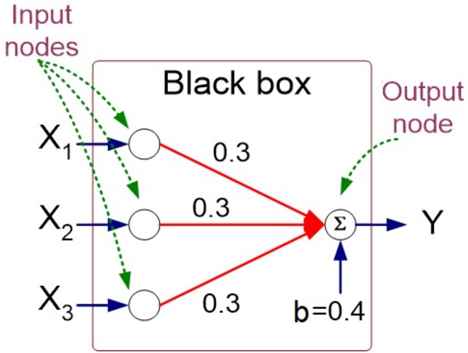


In other words perceptron computes $y = \mathbf{sign}(\overline{w} \bullet \overline{x} - b)$.

Perceptron Device

Example: Majority classifier

X_1	X_2	X_3	Y
1	0	0	-1
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	-1
0	1	0	-1
0	1	1	1
0	0	0	-1



$$y = \mathbf{sign}\left[\begin{pmatrix}0.3\\0.3\\0.3\end{pmatrix}^T\begin{pmatrix}x_1\\x_2\\x_3\end{pmatrix} - 0.4)\right] = \mathbf{sign}\left[0.3x_1 + 0.3x_2 + 0.3x_3 - 0.4\right].$$

Training Perceptron - Desiderata

Find a weight vector $\overline{w}$ and threshold b defining a hyperplane h such that

- all positive examples $\overline{x}_i^+$ from the training set (i.e. the feature vectors from the training set with $y = 1$) are on the positive side of the hyperplane h (i.e. $\overline{w} \bullet \overline{x}_i^+ - b > 0$),
- all negative examples (i.e. feature vectors with $y = -1$) are on the negative side of the hyperplane$h$ (i.e. $\overline{w} \bullet \overline{x}_j^- - b < 0$)

We simplify the task by noticing that $\overline{w} \bullet \overline{x} + b = \overline{w}' \bullet \overline{x}'$ where $\overline{w}' = (\overline{w}^T,\ b)^T$ and $\overline{x}' = (\overline{x}^T,\ 1)^T$.

- In the example $\overline{w}' = (\overline{w}^T b)^T = (0.3, 03, 0.3, -0.4)^T$ and $\overline{x}' = (\overline{x}^T,\ 1)^T = (x_1, x_2, x_3, 1)^T$ which makes

$$\overline{w}' \bullet \overline{x}' = \begin{pmatrix} 0.3 \\ 0.3 \\ 0.3 \\ -0.4 \end{pmatrix} \bullet \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{pmatrix} = 0.3x_1 + 0.3x_2 + 0.3x_3 - 0.4.$$

Training Perceptron - Take 1

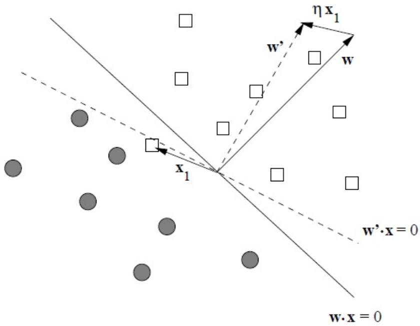
Modified Task: find $\overline{w}'$ such that $\overline{w}' \bullet (\overline{x}_i^+)' > 0$ and $\overline{w}' \bullet (\overline{x}_i^-)' < 0$ for all positive and negative eaxmples.
The following method converges to some hyperplane that separates the positive and negative examples, provided one exists.

1. Initialize the weight vector $\overline{w}$ to all 0's,
2. Pick a learning-rate parameter η, which is a small, positive real number,
3. While $\overline{w}$ keeps changing consider each training example $t = (\overline{x}, y)$ in turn:
   i. Compute $y' = \overline{w} \bullet (\overline{x}^T, 1)^T$;
   ii. If y' and y have different signs or $y' = 0$, then
      replace $\overline{w}$ by $\overline{w} + \eta \cdot y \cdot (\overline{x}^T, 1)^T$. That is, adjust $\overline{w}$ slightly in the direction of $\overline{x}$.

      else do nothing (t is properly classified).

Training Perceptron - What is Going on?

- Moving $\overline{w}$ towards $\overline{x}$ moves the orthogonal hyperplane h in such a direction that it makes it more likely that $\overline{x}$ will be on the correct side of the hyperplane, although it does not guarantee that $\overline{x}$ will then be correctly classified.
- The choice of η affects the convergence of the perceptron. If η is too small, then convergence is slow; if it is too big, then the decision boundary will dance around and again will converge slowly, if at all.



Training Perceptron - Example

from Lescovec, Rajaraman and Ullman. Mining of Massive Datasets

Perceptron to recognize spam email:

- The training set $(\overline{x}, y)$ where $\overline{x}$ is a vector of 0s and 1s, with value of x_i corresponding to the presence or absence of a word x_1 in the eemail,
- Since all values are boolean b must be 0 so to simplify computations we work with $\overline{w}$ instead of $\overline{w}'$ and with $\overline{x}$ instead of $\overline{x}'$,
- Tarining set consistst of 6 emails $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{e}, \mathbf{f}$.
- We use learning rate $\eta = 1/2$.

	and	viagra	the	of	nigeria	y
a	1	1	0	1	1	+1
b	0	0	1	1	0	−1
c	0	1	1	0	0	+1
d	1	0	0	1	0	−1
e	1	0	1	0	1	+1
f	1	0	1	1	0	−1

Step 1: $\overline{w}_1 = (0, 0, 0, 0, 0)^T$ and $y'_{\overline{a}} = \overline{w} \bullet \overline{a} = 0$; case $3.\mathrm{i}$, so $\overline{w} := \overline{w} + \eta \cdot y_{\overline{a}} \cdot \overline{a}$ or $\overline{w}_1 := (0, 0, 0, 0, 0)^T + (1/2)(+1)(1, 1, 0, 1, 1) = (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})^T$.

Training Perceptron - Example

	and	viagra	the	of	nigeria	y
a	1	1	0	1	1	+1
b	0	0	1	1	0	−1
c	0	1	1	0	0	+1
d	1	0	0	1	0	−1
e	1	0	1	0	1	+1
f	1	0	1	1	0	−1

Step 1: $\overline{w}_1 = (0,0,0,0,0)^T$ and $y'_{\overline{a}} = \overline{w} \bullet \overline{a} = 0$; case 3.i, so $\overline{w} := \overline{w} + \eta \cdot y_{\overline{a}} \cdot \overline{a}$ or $\overline{w}_1 := (0,0,0,0,0)^T + (1/2)(+1)(1,1,0,1,1) = (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})^T$.

Step 2: $y'_{\overline{b}} = \overline{w}_1 \bullet \overline{b} = (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})(0,0,1,1,0)^T = \frac{1}{2}$, but $y_{\overline{b}} = -1$, i.e. signs are different; case 3.i, so $\overline{w}_2 := (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})^T + (1/2)(-1)(0,0,1,1,0) = (\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, 0, \frac{1}{2})^T$.

Training Perceptron - Example

	and	viagra	the	of	nigeria	y
a	1	1	0	1	1	+1
b	0	0	1	1	0	−1
c	0	1	1	0	0	+1
d	1	0	0	1	0	−1
e	1	0	1	0	1	+1
f	1	0	1	1	0	−1

Step 1: $\overline{w}_1 = (0,0,0,0,0)^T$ and $y^{'}_{\overline{a}} = \overline{w} \bullet \overline{a} = 0$; case $3.\text{i}$, so $\overline{w} := \overline{w} + \eta \cdot y_{\overline{a}} \cdot \overline{a}$ or $\overline{w}_1 := (0,0,0,0,0)^T + (1/2)(+1)(1,1,0,1,1) = (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})^T$.

Step 2: $y^{'}_{\overline{b}} = \overline{w}_1 \bullet \overline{b} = (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})(0,0,1,1,0)^T = \frac{1}{2}$, but $y_{\overline{b}} = -1$, i.e. signs are different; case $3.\text{i}$, so
$\overline{w}_2 := (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})^T + (1/2)(-1)(0,0,1,1,0) = (\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, 0, \frac{1}{2})^T$.

Step 3: $y^{'}_{\overline{c}} = \overline{w}_2 \bullet \overline{c} = 0$; case $3.\text{i}$, so
$\overline{w}_3 := (\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, 0, \frac{1}{2})^T + (1/2)(1)(0,1,1,0,0) = (\frac{1}{2}, 1, 0, 0, \frac{1}{2})^T$.

Training Perceptron - Example

	and	viagra	the	of	nigeria	y
a	1	1	0	1	1	+1
b	0	0	1	1	0	−1
c	0	1	1	0	0	+1
d	1	0	0	1	0	−1
e	1	0	1	0	1	+1
f	1	0	1	1	0	−1

Step 1: $\overline{w}_1 = (0,0,0,0,0)^T$ and $y'_{\overline{a}} = \overline{w} \bullet \overline{a} = 0$; case 3.i, so $\overline{w} := \overline{w} + \eta \cdot y_{\overline{a}} \cdot \overline{a}$ or $\overline{w}_1 := (0,0,0,0,0)^T + (1/2)(+1)(1,1,0,1,1) = (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})^T$.

Step 2: $y'_{\overline{b}} = \overline{w}_1 \bullet \overline{b} = (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})(0,0,1,1,0)^T = \frac{1}{2}$, but $y_{\overline{b}} = -1$, i.e. signs are different; case 3.i, so
$\overline{w}_2 := (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})^T + (1/2)(-1)(0,0,1,1,0) = (\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, 0, \frac{1}{2})^T$.

Step 3: $y'_{\overline{c}} = \overline{w}_2 \bullet \overline{c} = 0$; case 3.i, so
$\overline{w}_3 := (\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, 0, \frac{1}{2})^T + (1/2)(1)(0,1,1,0,0) = (\frac{1}{2}, 1, 0, 0, \frac{1}{2})^T$.

Step 4: $y'_{\overline{d}} = \overline{w}_3 \bullet \overline{d} = 1$, but $y_{\overline{d}} = -1$; case 3.i, so
$\overline{w}_4 := (\frac{1}{2}, 1, 0, 0, \frac{1}{2})^T + (1/2)(-1)(1,0,0,1,0) = (0, 1, 0, -\frac{1}{2}, \frac{1}{2})^T$.

Training Perceptron - Example

	and	viagra	the	of	nigeria	y
a	1	1	0	1	1	+1
b	0	0	1	1	0	−1
c	0	1	1	0	0	+1
d	1	0	0	1	0	−1
e	1	0	1	0	1	+1
f	1	0	1	1	0	−1

Step 1: $\overline{w}_1 = (0,0,0,0,0)^T$ and $y'_{\overline{a}} = \overline{w} \bullet \overline{a} = 0$; case $3.\text{i}$, so $\overline{w} := \overline{w} + \eta \cdot y_{\overline{a}} \cdot \overline{a}$ or $\overline{w}_1 := (0,0,0,0,0)^T + (1/2)(+1)(1,1,0,1,1) = (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})^T$.

Step 2: $y'_{\overline{b}} = \overline{w}_1 \bullet \overline{b} = (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})(0,0,1,1,0)^T = \frac{1}{2}$, but $y_{\overline{b}} = -1$, i.e. signs are different; case $3.\text{i}$, so
$\overline{w}_2 := (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})^T + (1/2)(-1)(0,0,1,1,0) = (\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, 0, \frac{1}{2})^T$.

Step 3: $y'_{\overline{c}} = \overline{w}_2 \bullet \overline{c} = 0$; case $3.\text{i}$, so
$\overline{w}_3 := (\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, 0, \frac{1}{2})^T + (1/2)(1)(0,1,1,0,0) = (\frac{1}{2}, 1, 0, 0, \frac{1}{2})^T$.

Step 4: $y'_{\overline{d}} = \overline{w}_3 \bullet \overline{d} = 1$, but $y_{\overline{d}} = -1$; case $3.\text{i}$, so
$\overline{w}_4 := (\frac{1}{2}, 1, 0, 0, \frac{1}{2})^T + (1/2)(-1)(1,0,0,1,0) = (0, 1, 0, -\frac{1}{2}, \frac{1}{2})^T$.

Step 5: $y'_{\overline{e}} = \overline{w}_4 \bullet \overline{e} = \frac{1}{2}$, and $y_{\overline{e}} = 1$; case $3.\text{ii}$, do nothing.

Training Perceptron - Example

	and	viagra	the	of	nigeria	y
a	1	1	0	1	1	+1
b	0	0	1	1	0	−1
c	0	1	1	0	0	+1
d	1	0	0	1	0	−1
e	1	0	1	0	1	+1
f	1	0	1	1	0	−1

Step 1: $\overline{w}_1 = (0,0,0,0,0)^T$ and $y'_{\overline{a}} = \overline{w} \bullet \overline{a} = 0$; case $3.\mathrm{i}$, so $\overline{w} := \overline{w} + \eta \cdot y_{\overline{a}} \cdot \overline{a}$ or $\overline{w}_1 := (0,0,0,0,0)^T + (1/2)(+1)(1,1,0,1,1) = (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})^T$.

Step 2: $y'_{\overline{b}} = \overline{w}_1 \bullet \overline{b} = (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})(0,0,1,1,0)^T = \frac{1}{2}$, but $y_{\overline{b}} = -1$, i.e. signs are different; case $3.\mathrm{i}$, so
$\overline{w}_2 := (\frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2})^T + (1/2)(-1)(0,0,1,1,0) = (\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, 0, \frac{1}{2})^T$.

Step 3: $y'_{\overline{c}} = \overline{w}_2 \bullet \overline{c} = 0$; case $3.\mathrm{i}$, so
$\overline{w}_3 := (\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, 0, \frac{1}{2})^T + (1/2)(1)(0,1,1,0,0) = (\frac{1}{2}, 1, 0, 0, \frac{1}{2})^T$.

Step 4: $y'_{\overline{d}} = \overline{w}_3 \bullet \overline{d} = 1$, but $y_{\overline{d}} = -1$; case $3.\mathrm{i}$, so
$\overline{w}_4 := (\frac{1}{2}, 1, 0, 0, \frac{1}{2})^T + (1/2)(-1)(1,0,0,1,0) = (0, 1, 0, -\frac{1}{2}, \frac{1}{2})^T$.

Step 5: $y'_{\overline{e}} = \overline{w}_4 \bullet \overline{e} = \frac{1}{2}$, and $y_{\overline{e}} = 1$; case $3.\mathrm{ii}$, do nothing.

Step 6: $y'_{\overline{f}} = \overline{w}_4 \bullet \overline{f} = -\frac{1}{2}$, and $y_{\overline{f}} = -1$; case $3.\mathrm{ii}$, do nothing.

If we now check $\overline{a}$ through $\overline{d}$, we find that this current $\overline{w}_4$ correctly classifies them all.

Perceptron Convergence

What means that for a given training dataset D there is a separating hyperplane h that passes through the origin? This means that there is $\delta > 0$ such that marginal distance of any data point in D from h is at least $\delta > 0$. Let $\overline{w}^*$ be a unit vector orthogonal to h. Then marginal distance of $\overline{x}$ from h is the length of its projection onto $\overline{w}^*$. Since $\overline{w}^*$ is a unit vector the projection is $(\overline{x} \bullet \overline{w}^*)\overline{w}^*$. So existence of separating plane for D means that $\|(\overline{x} \bullet \overline{w}^*)\overline{w}^*\| = |\overline{x} \bullet \overline{w}^*| \geq \delta$.

Theorem (Convergence)

If for a given training dataset D there is a separating hyperplane h that passes through the origin then the perceptron training converges in at most $\left(\frac{\max_{\overline{x} \in D} \|\overline{x}\|}{\delta}\right)^2$ iterations of step 3.

Perceptron Convergence

Theorem (Convergence)

If for a given training dataset D there is a separating hyperplane h that passes through the origin then the perceptron training converges in at most $\left(\frac{\max_{\overline{x}\in D}\|\overline{x}\|}{\delta}\right)^2$ iterations of step 3.

Proof.

Let $\overline{w}_{n+1}$ denote the weight vector after $n+1$ iterations, let $\overline{w}^*$ be a unit vector orthogonal to h and let δ be separating margin. If we obtained $\overline{w}^{n+1}$ then $\overline{w}_n$ did not classify D correctly and $\overline{w}_{n+1} = \overline{w}_n + y\eta\overline{x}$ where $\overline{x}$ was last misclassified feature vector. Then

$$\begin{array}{rcl} \overline{w}_{n+1} \bullet \overline{w}^* & = & (\overline{w}_n + y\eta\overline{x}) \bullet \overline{w}^* \\ & = & \overline{w}_n \bullet \overline{w}^* + y\eta\overline{x} \bullet \overline{w}^* \\ & = & \overline{w}_n \bullet \overline{w}^* + \eta|\overline{x} \bullet \overline{w}^*| \\ & \geq & \overline{w}_n \bullet \overline{w}^* + \eta\delta \end{array}.$$

By Cauchy-Schwartz we have $\overline{w}_{n+1} \bullet \overline{w}^* \leq \|\overline{w}_{n+1}\|\,\|\overline{w}^*\|$, so since $\|\overline{w}\|^* = 1$ holds $\|\overline{w}_{n+1}\| \geq \overline{w}_n \bullet \overline{w}^* + \eta\delta$. □

Perceptron Convergence

Theorem (Convergence)

If for a given training dataset D there is a separating hyperplane h that passes through the origin then the perceptron training converges in at most $\left(\frac{\max_{\overline{x}\in D}\|\overline{x}\|}{\delta}\right)^2$ iterations of step 3.

Proof.

Notice that

$$\begin{aligned}
\|\overline{w}_{n+1}\|^2 &= \|\overline{w}_n + y\eta\overline{x}\|^2 \\
&= \|\overline{w}_n\|^2 + 2y\eta(\overline{x} \bullet \overline{w}_n) + \eta^2 \|\overline{x}\| \\
&\leq \|\overline{w}_n\|^2 + \eta^2(\max_{\overline{x}\in D} \|\overline{x}\|)^2
\end{aligned}$$

The last line is justified because $2y\eta(\overline{x} \bullet \overline{w}_n) \leq 0$ which holds because otherwise $\operatorname{sgn} y \neq \operatorname{sgn} \overline{x} \bullet \overline{w}_n$ because otherwise no update is done. $\square$

Perceptron Convergence

Theorem (Convergence)

If for a given training dataset D there is a separating hyperplane h that passes through the origin then the perceptron training converges in at most $\left(\frac{\max_{\overline{x}\in D}\|\overline{x}\|}{\delta}\right)^2$ iterations of step 3.

Proof.

We proved $\|\overline{w}_{n+1}\| \geq \overline{w}_{n+1} \bullet \overline{w}^* \geq \overline{w}_n \bullet \overline{w}^* + \eta\delta$ and $\|\overline{w}_{n+1}\|^2 \leq \|\overline{w}_n\|^2 + \eta^2(\max_{\overline{x}\in D}\|\overline{x}\|)^2$. So since we started with $\overline{w}_1 = \overline{0}$ by induction after N actual updates $\|\overline{w}_N\|^2 \leq N\eta^2(\max_{\overline{x}\in D}\|\overline{x}\|)^2$ and $\|\overline{w}_N\| \geq \overline{w}_N \bullet \overline{w}^* \geq N\eta\delta$. Combining we get

$$(N\eta\delta)^2 \leq (\overline{w}_N \bullet \overline{w}^*)^2 \leq \|\overline{w}_N\|^2 \leq N\eta^2(\max_{\overline{x}\in D}\|\overline{x}\|)^2$$

thus training converges in $N \leq \left(\frac{\max_{\overline{x}\in D}\|\overline{x}\|}{\delta}\right)^2$ updates. $\square$

Dealing with Perceptron Convergence

- If the data points are linearly separable, then the perceptron will eventually find a separator hyperplane. But if margin δ is very small it'll take very long time
- If the data is not linearly separable, then the perecptron will eventually repeat a weight vector and loop infinitely

Yet if convergence rate is very slow then the two cases are indistinguishable. Need a termination strategy. Possible solutions:

- Termination strategy:
    1. Terminate after a fixed number of rounds,
    2. Terminate after normalized distance between true classification vector $\overline{y}$ and computed classification vector $\overline{y}'$ becomes small, i.e. $\frac{\|\overline{y}-\overline{y}'\|}{n} \leq \theta$ where θ is some small constant,
    3. Terminate when the number of misclassified training points stops changing.

Speedup Improvement Ideas

- Lower the training rate as the number of rounds increases. For example, start with the training rate η_0 and lower it to $\frac{\eta_0}{1+ct}$ after the t^{th} round, where c is some small constant. Works well inmost cases in practice but no convergence guarantee.
- Instead of η use flexible rate increases for different components of $\overline{w}$:
  - For a training data vector $\overline{x}$ that is in the class $y_{\overline{x}} = +1$ if current $\overline{w}$ is such that $\overline{w} \bullet \overline{x}$ is small/negative then we have to raise the weights w_i in those components x_i where value of x is positive and relatively large. Thus multiplier for w_i for these components must be $c > 1$. Usually this is implemented by taking $c = 2$
  - For a training data vector $\overline{x}$ that is in the class $y_{\overline{x}} = -1$ if current $\overline{w}$ is such that $\overline{w} \bullet \overline{x}$ is large/positive then we have to lower the weights w_i in those components x_i where value of x is positive and relatively large. Thus multiplier for w_i for these components must be < 1. Usually this is implemented by taking $c = \frac{1}{2}$

  Implementation of this idea with appropriate values of multipliers guarantees convergence and in most cases works faster than basic perceptron.