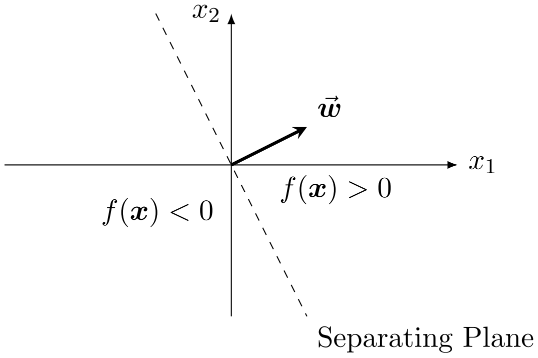
THE PERCEPTRON

The perceptron is a learning algorithm for binary classification of real valued vectors.

1 Introduction

The perceptron binary classifier can be thought of as the following function,

$$f(\boldsymbol{x}) = \begin{cases} 1 \text{ if } \boldsymbol{w}^T\boldsymbol{x} > 0 \\ 0 \text{ otherwise} \end{cases}$$



One severe limitation of this formulation is that the separating hyperplane always passes through origin which might be undesirable in many cases. However, this limitation can be overcome in the following way,

$$f(\boldsymbol{x}) = \begin{cases} 1 \text{ if } \boldsymbol{w}^T\boldsymbol{x} + b > 0 \\ 0 \text{ otherwise} \end{cases}$$

The bias term b frees the separating hyperplane from origin. Additionally, we can employ a trick to simplify the expression as follows.

$$f(\boldsymbol{x}) = \begin{cases} 1 \text{ if } \left(\boldsymbol{w}^T, b\right)\begin{pmatrix}\boldsymbol{x}\\1\end{pmatrix} > 0 \\ 0 \text{ otherwise} \end{cases}$$

Hence, by increasing the dimension of input by one and defaulting the intercept on the new dimension to 1 for every input vector, any separating hyperplane in the old dimensional space becomes a hyperplane passing through origin in the new dimensional space. This simplifies the mathematics without constraining the classifier.

The vector $\vec{\boldsymbol{w}}$ is learned from training data as usual. The algorithm for learning the same is presented in the next section.

2 Learning Algorithm

The training data consists of N pairs of d-dimensional real input vectors $\boldsymbol{x}_i$s and the output binary labels y_is. For mathematical convenience, positive output labels are denoted by +1 and negative output labels by -1. The learning algorithm makes the following assumptions:

- It is assumed that the input vectors have already been extended to account for the bias term.
- The training data is assumed to be linearly separable at origin. So, there exists a unit vector $\boldsymbol{w}^*$ such that $y_i \boldsymbol{w}^{*T} \boldsymbol{x_i} > \gamma \ \forall i$ where $\gamma > 0$.
- It is assumed that all training input vectors are finite.

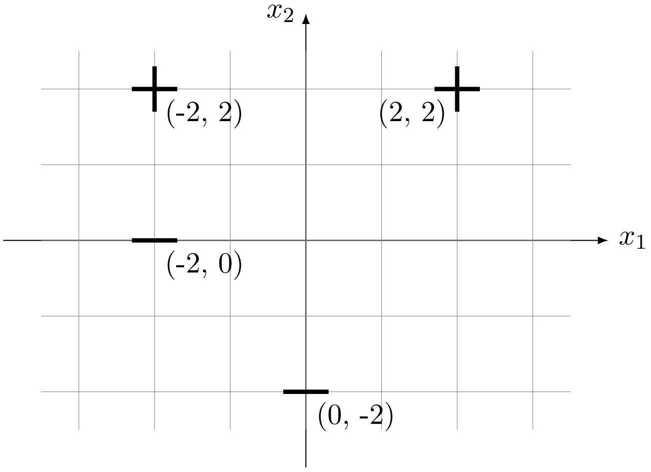
The algorithm is defined as follows.

Algorithm The Perceptron Learning Algorithm

```
$k \leftarrow 1$
$\boldsymbol{w}_k \leftarrow 0$
**while** there exists $j \in \{1, 2, .., N\}$ such that $y_j \boldsymbol{w}^{*T} \boldsymbol{x_j} <= 0$ **do**
  pick $i \in \{1, 2, .., N\}$ such that $y_i \boldsymbol{w}^{*T} \boldsymbol{x_i} <= 0$
  $\boldsymbol{w}_{k+1} \leftarrow \boldsymbol{w}_k + y_i \boldsymbol{x_i}$
  $k \leftarrow k + 1$
**end while**
**return** $\boldsymbol{w}_k$
```

3 Example

The algorithm is illustrated on a very simple example.



3.1 Iteration 1

$$\boldsymbol{w}_1 = \begin{pmatrix}0\\0\end{pmatrix}$$

$\boldsymbol{x}$		
$\begin{pmatrix}-2\\0\end{pmatrix}$	-1	0
$\begin{pmatrix}0\\-2\end{pmatrix}$	-1	0
$\begin{pmatrix}-2\\2\end{pmatrix}$	1	0
$\begin{pmatrix}2\\2\end{pmatrix}$	1	0

3.2 Iteration 2

$$\begin{aligned}\boldsymbol{w}_2 &= \begin{pmatrix}0\\0\end{pmatrix} + (-1)\begin{pmatrix}-2\\0\end{pmatrix}\\ &= \begin{pmatrix}2\\0\end{pmatrix}\end{aligned}$$

$\boldsymbol{x}$		
$\begin{pmatrix}-2\\0\end{pmatrix}$	-1	4
$\begin{pmatrix}0\\-2\end{pmatrix}$	-1	0
$\begin{pmatrix}-2\\2\end{pmatrix}$	1	-4
$\begin{pmatrix}2\\2\end{pmatrix}$	1	4

3.3 Iteration 3

$$\begin{aligned}\boldsymbol{w}_3 &= \begin{pmatrix}2\\0\end{pmatrix} + (-1)\begin{pmatrix}0\\-2\end{pmatrix}\\ &= \begin{pmatrix}2\\2\end{pmatrix}\end{aligned}$$

$\boldsymbol{x}$		
$\begin{pmatrix}-2\\0\end{pmatrix}$	-1	4
$\begin{pmatrix}0\\-2\end{pmatrix}$	-1	4
$\begin{pmatrix}-2\\2\end{pmatrix}$	1	0
$\begin{pmatrix}2\\2\end{pmatrix}$	1	8

3.4 Iteration 4

$$\begin{aligned}\boldsymbol{w}_4 &= \begin{pmatrix}2\\2\end{pmatrix} + (1)\begin{pmatrix}-2\\2\end{pmatrix}\\ &= \begin{pmatrix}0\\4\end{pmatrix}\end{aligned}$$

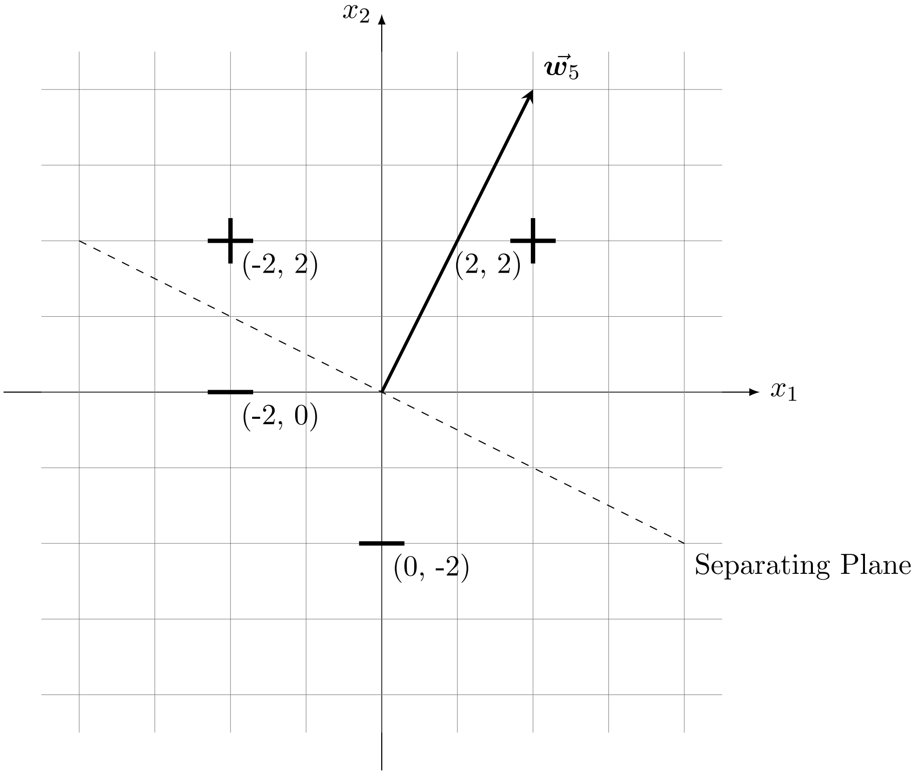
$\boldsymbol{x}$		
$\begin{pmatrix}-2\\0\end{pmatrix}$	-1	0
$\begin{pmatrix}0\\-2\end{pmatrix}$	-1	8
$\begin{pmatrix}-2\\2\end{pmatrix}$	1	8
$\begin{pmatrix}2\\2\end{pmatrix}$	1	8

3.5 Iteration 5

$$\begin{aligned}\boldsymbol{w}_5 &= \begin{pmatrix}0\\4\end{pmatrix} + (-1)\begin{pmatrix}-2\\0\end{pmatrix}\\ &= \begin{pmatrix}2\\4\end{pmatrix}\end{aligned}$$

$\boldsymbol{x}$		
$\begin{pmatrix}-2\\0\end{pmatrix}$	-1	4
$\begin{pmatrix}0\\-2\end{pmatrix}$	-1	8
$\begin{pmatrix}-2\\2\end{pmatrix}$	1	4
$\begin{pmatrix}2\\2\end{pmatrix}$	1	12

All the points are correctly classified, hence the algorithm terminates. The algorithm converged in 5 iterations. In fact, as will be proved later, the algorithm is always guaranteed to converge in finite iterations.The resulting separating hyperplanes is shown below.



4 Proof of Convergence

The intention is to show that the perceptron learning algorithm always converges to a perfect classifier in finite iterations given that all the assumptions that the algorithm makes are met.

Assume that after k+1 iterations, $\vec{\boldsymbol{w}_{k+1}}$ makes an angle θ_{k+1} with $\vec{\boldsymbol{w}}^*$.

$$cos(\theta_{k+1}) = \frac{\vec{\boldsymbol{w}_{k+1}}.\vec{\boldsymbol{w}}^*}{|\vec{\boldsymbol{w}_{k+1}}|\ |\vec{\boldsymbol{w}}^*|}$$

Since $\vec{\boldsymbol{w}}^*$ is a unit vector,

$$cos(\theta_{k+1}) = \frac{\vec{\boldsymbol{w}_{k+1}}.\vec{\boldsymbol{w}}^*}{|\vec{\boldsymbol{w}_{k+1}}|}$$

Now, bounds for numerator and denominator are established to proceed ahead.

4.1 Bound for numerator

The update rule for $\vec{\boldsymbol{w}_{k+1}}$ is,

$$\vec{\boldsymbol{w}_{k+1}} = \vec{\boldsymbol{w}}_k + y_i\vec{\boldsymbol{x_i}}$$

Taking dot product with $\vec{\boldsymbol{w}}^*$ on both sides.

$$\vec{\boldsymbol{w}}^*.\vec{\boldsymbol{w}_{k+1}} = \vec{\boldsymbol{w}}^*.\vec{\boldsymbol{w}}_k + y_i\vec{\boldsymbol{w}}^*.\vec{\boldsymbol{x_i}}$$

Since $y_i\vec{\boldsymbol{w}}^*.\vec{\boldsymbol{x_i}} > \gamma$ as per the algorithm assumption.

$$\begin{aligned}\vec{\boldsymbol{w}}^*.\vec{\boldsymbol{w}_{k+1}} &> \vec{\boldsymbol{w}}^*.\vec{\boldsymbol{w}}_k + \gamma\\ \vec{\boldsymbol{w}}^*.\vec{\boldsymbol{w}_{k+1}} &> \vec{\boldsymbol{w}}^*.\vec{\boldsymbol{w}_{k-1}} + 2\gamma\\ \vec{\boldsymbol{w}}^*.\vec{\boldsymbol{w}_{k+1}} &> \vec{\boldsymbol{w}}^*.\vec{\boldsymbol{w}}_1 + k\gamma\end{aligned}$$

since $\vec{\boldsymbol{w}}^*.\vec{\boldsymbol{w}}_1 = 0$.

$$\vec{\boldsymbol{w}}^*.\vec{\boldsymbol{w}_{k+1}} > k\gamma$$

4.2 Bound for denominator

Again starting with the update rule for $\vec{\boldsymbol{w}_{k+1}}$,

$$\vec{\boldsymbol{w}_{k+1}} = \vec{\boldsymbol{w}}_k + y_i\vec{\boldsymbol{x_i}}$$

Taking modulus on both sides,

$$|\vec{\boldsymbol{w}_{k+1}}|^2 = |\vec{\boldsymbol{w}}_k|^2 + |y_i\vec{\boldsymbol{x_i}}|^2 + 2y_i\vec{\boldsymbol{w}}_k.\vec{\boldsymbol{x_i}}$$

Since $y_i\vec{\boldsymbol{w}}_k.\vec{\boldsymbol{x_i}} <= 0$ according to the algorithm specification.

$$|\vec{\boldsymbol{w}_{k+1}}|^2 \leq |\vec{\boldsymbol{w}}_k|^2 + |y_i\vec{\boldsymbol{x_i}}|^2$$

Assuming that the largest input vector has length R.

$$\begin{aligned}|\vec{\boldsymbol{w}_{k+1}}|^2 &\leq |\vec{\boldsymbol{w}}_k|^2 + R^2\\ |\vec{\boldsymbol{w}_{k+1}}|^2 &\leq |\vec{\boldsymbol{w}_{k-1}}|^2 + 2R^2\\ |\vec{\boldsymbol{w}_{k+1}}|^2 &\leq kR^2\\ |\vec{\boldsymbol{w}_{k+1}}| &\leq \sqrt{k}R\end{aligned}$$

4.3 Bound for Number of Iterations

Using both the numerator and denominator bounds,

$$
\begin{aligned}
cos(\theta_{k+1}) &> \frac{k\gamma}{\sqrt{k}R} \\
cos(\theta_{k+1}) &> \sqrt{k}\frac{\gamma}{R}
\end{aligned}
$$

This shows that after each iteration, the lower bound of the angle between $\vec{\boldsymbol{w}}$ and $\vec{\boldsymbol{w}}^*$ increases. And since $cos(\theta_{k+1})$ can never be greater than one, $\sqrt{k}\frac{\gamma}{R} \leq 1$ must hold true yielding $k \leq \frac{R^2}{\gamma^2}$.

5 Conclusion

The perceptron was one of the first algorithms indicating that computers could learn from data. And today, after half a century, it continues to be the basis of Artificial Neural Networks which are at the forefront of Artificial Intelligence.