PERCEPTRONS

Prof. Dan A. Simovici

UMB

General Framework for Linear Classifiers

- $X \subseteq \mathbb{R}^n$ is the input space and Y is the output domain;
-
  - $Y = \{-1, 1\}$ for binary classification;
  - $Y = \{1, 2, \ldots, m\}$ for m-class classification;
  - $Y \subseteq \mathbb{R}$ for regression;
- A training sequence is a sequence

$$S = \left(\begin{pmatrix} \mathbf{x}_1 \\ y_1 \end{pmatrix}, \ldots, \begin{pmatrix} \mathbf{x}_\ell \\ y_\ell \end{pmatrix} \right),$$

where $\mathbf{x}_i \in X$ are examples or instances, and $y_i \in Y$ are the labels.

Points and Hyperplanes

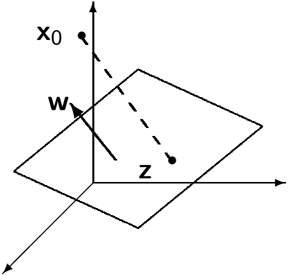
Let $\mathbf{w}'\mathbf{x} + b = 0$ be a hyperplane H in $\mathbb{R}^n$. The vector $\mathbf{w}$ is orthogonal to H, so the line that passes through $\mathbf{x}_0$ and is orthogonal to the hyperplane is

$$\mathbf{x} - \mathbf{x}_0 = a\mathbf{w}.$$

The intersection of this line with the hyperplane is $\mathbf{w}'(\mathbf{x}_0 + a\mathbf{w}) + b = 0$, so

$$a = -\frac{\mathbf{w}'\mathbf{x}_0 + b}{\parallel \mathbf{w} \parallel^2}.$$

Points and Hyperplanes



The projection of $\mathbf{x}_0$ on the hyperplane H given by $\mathbf{w}'\mathbf{x} + b = 0$ is

$$\mathbf{z} = \mathbf{x}_0 - \frac{\mathbf{w}'\mathbf{x}_0 + b}{\parallel \mathbf{w} \parallel^2}\mathbf{w}$$

and the distance from $\mathbf{x}_0$ to H is $\frac{|\mathbf{w}'\mathbf{x}_0 + b|}{\|\mathbf{w}\|}$. When $\parallel \mathbf{w} \parallel = 1$ this distance is $|\mathbf{w}'\mathbf{x}_0 + b|$.
The two half-spaces determined by H are characterized by $\mathbf{w}'\mathbf{x} + b > 0$ and by $\mathbf{w}'\mathbf{x} + b < 0$.

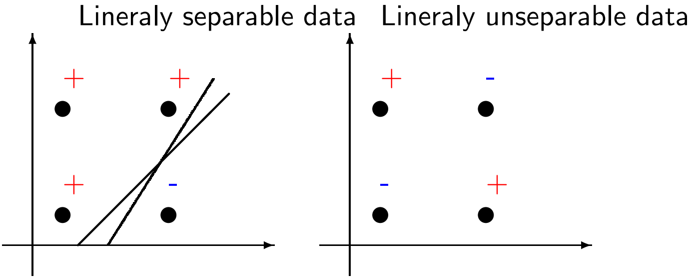
Learning using Perceptrons

Perceptrons were introduced in [4] as models of learning in the brain. A training sequence

$$S = \left(\begin{pmatrix} \mathbf{x}_1 \\ y_1 \end{pmatrix}, \ldots, \begin{pmatrix} \mathbf{x}_\ell \\ y_\ell \end{pmatrix} \right)$$

is linearly separable if there exists a hyperplane $\mathbf{w}'\mathbf{x} + b = 0$ such that $\mathbf{w}'\mathbf{x}_i + b \geq 0$ if $y_i = 1$ and $\mathbf{w}'\mathbf{x}_i + b < 0$ if $y_i = -1$.

Linearly Separable vs. Unseparable Data



Task of Learning algorithm (perceptron): find a hyperplane for a linearly separable data set.

Features of the Learning Algorithm

- a hyperplane H defined by $f(\mathbf{x}) = \mathbf{w}'\mathbf{x} + b = 0$;
- $\mathbf{w}$ is the weight vector and b is the bias;
- if $f(\mathbf{x}) \geqslant 0$, $\mathbf{x}$ is a positive example; otherwise, it is a negative example;
- the radius of a ball centered in $\mathbf{0}$ that includes all examples is $R = \max\{\| \mathbf{x}_i \| \mid 1 \leqslant i \leqslant \ell\}$;
- the functional margin of $\begin{pmatrix} \mathbf{x}_i \\ y_i \end{pmatrix}$ relative to the hyperplane $\mathbf{w}'\mathbf{x} + b = 0$ is $\gamma_i = y_i(\mathbf{w}'\mathbf{x}_i + b)$; $\gamma = min\gamma_i$ is the margin of the hyperplane H relative to S;
- if y_i and $\mathbf{w}'\mathbf{x}_i + b$ have the same sign, then $\begin{pmatrix} \mathbf{x}_i \\ y_i \end{pmatrix}$ is classified correctly $(\gamma_i > 0)$; otherwise, is incorrectly classified $(\gamma_i \leqslant 0)$.

Algorithm Perceptron(S, η)

Algorithm 1.1: Learning Algorithm for Perceptron

Data: labelled training sequence S and learning rate η
Result: weight vector $\mathbf{w}$ and parameter b defining classifier

```
initialize w = 0, b_0 = 0, k = 0;
R = max{|| x_i || | 1 ⩽ i ⩽ ℓ};
repeat
    for i = 1 to ℓ do
        if y_i(w'_k x_i + b_k) ⩽ 0 then
            w_{k+1} = w_k + η y_i x_i;
            b_{k+1} = b_k + η y_i R^2;
            k = k + 1;
        end
    end
until no mistakes are made in the for loop ;
return k, (w_k, b_k) where k is the number of mistakes;
```

Rosenblatt-Novikoff Theorem

(variant of Cristianini and Shawe-Taylor [1] of Novikoff's Proof [3])

Theorem

Let $S = \left(\begin{pmatrix} \mathbf{x}_1 \\ y_1 \end{pmatrix}, \ldots, \begin{pmatrix} \mathbf{x}_\ell \\ y_\ell \end{pmatrix} \right)$ *be a non-trivial training sequence that is linearly separable, and let* $R = \max\{\parallel \mathbf{x}_i \parallel \mid 1 \leqslant i \leqslant \ell\}$. *Suppose there exists an optimal weight vector* $\mathbf{w}_{opt}$ *and an optimal bias* b_{opt} *such that*

$$\parallel \mathbf{w}_{opt} \parallel = 1 \text{ and } y_i(\mathbf{w}'_{opt}\mathbf{x}_i + b_{opt}) \geqslant \gamma,$$

for $1 \leqslant i \leqslant \ell$. *The, the number of mistakes made by the algorithm is at most*

$$\left(\frac{2R}{\gamma}\right)^2$$

Proof

Let

- t be the update counter;
- and let

$$\hat{\mathbf{w}} = \begin{pmatrix} \mathbf{w} \\ \frac{b}{R} \end{pmatrix} \text{ and } \hat{\mathbf{x}}_i = \begin{pmatrix} \mathbf{x}_i \\ R \end{pmatrix}$$

for $1 \leqslant i \leqslant \ell$.

The algorithm begins with an augmented vector $\hat{\mathbf{w}}_0 = \mathbf{0}$ and updates it at each mistake.

Let $\hat{\mathbf{w}}_{t-1}$ be the augmented weight vector prior to the t^{th} mistake. The t^{th} update is performed when

$$y_i \hat{\mathbf{w}}'_{t-1} \hat{\mathbf{x}}_i = y_i(\mathbf{w}'_{t-1}\mathbf{x}_i + b_{t-1}) \leqslant 0,$$

where $(\mathbf{x}_i, y_i)$ is the example incorrectly classified by

$$\hat{\mathbf{w}}_{t-1} = \begin{pmatrix} \mathbf{w}_{t-1} \\ \frac{b_{t-1}}{R} \end{pmatrix}.$$

Proof (cont'd)

The update is

$$\begin{aligned}\hat{\mathbf{w}}_t &= \begin{pmatrix}\mathbf{w}_t\\ \frac{b_t}{R}\end{pmatrix} = \begin{pmatrix}\mathbf{w}_{t-1}+\eta y_i \mathbf{x}_i\\ \frac{b_{t-1}+\eta y_i R^2}{R}\end{pmatrix}\\ &= \begin{pmatrix}\mathbf{w}_{t-1}+\eta y_i \mathbf{x}_i\\ \frac{b_{t-1}}{R}+\eta y_i R\end{pmatrix} = \begin{pmatrix}\mathbf{w}_{t-1}\\ \frac{b_{t-1}}{R}\end{pmatrix} + \begin{pmatrix}\eta y_i \mathbf{x}_i\\ \eta y_i R\end{pmatrix}\\ &= \hat{\mathbf{w}}_{t-1}+\eta y_i \hat{\mathbf{x}}_i,\end{aligned}$$

where we used the fact that $b_t = b_{t-1} + \eta y_i R^2$.
By hypothesis, we have

$$y_i \hat{\mathbf{w}}'_{opt}\hat{\mathbf{x}}_i = y_i \begin{pmatrix}\mathbf{w}'_{opt} & \frac{b}{R}\end{pmatrix}\begin{pmatrix}\mathbf{x}_i\\ R\end{pmatrix} = y_i(\mathbf{w}'_{opt}\mathbf{x}_i + b) \geqslant \gamma,$$

which implies

$$\hat{\mathbf{w}}'_{opt}\hat{\mathbf{w}}_t = \hat{\mathbf{w}}'_{opt}\hat{\mathbf{w}}'_{t-1} + \eta y_i \hat{\mathbf{w}}'_{opt}\hat{\mathbf{x}}_i \geqslant \hat{\mathbf{w}}'_{opt}\hat{\mathbf{w}}_{t-1} + \eta\gamma.$$

Proof (cont'd)

By repeated application of the inequality $\hat{\mathbf{w}}'_{opt}\hat{\mathbf{w}}_t \geqslant \eta\gamma$ we obtain

$$\hat{\mathbf{w}}'_{opt}\mathbf{w}_t \geqslant t\eta\gamma.$$

Since $\hat{\mathbf{w}}_t = \hat{\mathbf{w}}_{t-1} + \eta y_i \hat{\mathbf{x}}_i$, we have

$$\begin{aligned}
\parallel \hat{\mathbf{w}}_t \parallel^2 &= \hat{\mathbf{w}}'_t\hat{\mathbf{w}}_t = (\hat{\mathbf{w}}'_{t-1} + \eta y_i \hat{\mathbf{x}}'_i)(\hat{\mathbf{w}}_{t-1} + \eta y_i \hat{\mathbf{x}}_i) \\
&= \parallel \hat{\mathbf{w}}_{t-1} \parallel^2 + 2\eta y_i \hat{\mathbf{w}}'_{t-1}\hat{\mathbf{x}}_i + \eta^2 \parallel \hat{\mathbf{x}}_i \parallel^2 \\
& \quad \text{(because } y_i \hat{\mathbf{w}}'_{t-1}\hat{\mathbf{x}}_i \leqslant 0 \text{ when an update occurs)} \\
&\leqslant \parallel \hat{\mathbf{w}}_{t-1} \parallel^2 + \eta^2 \parallel \hat{\mathbf{x}}_i \parallel^2 \\
&\leqslant \parallel \hat{\mathbf{w}}_{t-1} \parallel^2 + \eta^2(\parallel \mathbf{x}_i \parallel^2 + R^2) \\
&\leqslant \parallel \hat{\mathbf{w}}_{t-1} \parallel^2 + 2\eta^2 R^2,
\end{aligned}$$

which implies $\parallel \hat{\mathbf{w}}_t \parallel^2 \leqslant 2t\eta^2 R^2$.

Proof (cont'd)

By combining the inequalities

$$\hat{\mathbf{w}}'_{opt}\mathbf{w}_t \geqslant t\eta\gamma \text{ and } \parallel \hat{\mathbf{w}}_t \parallel^2 \leqslant 2t\eta^2 R^2$$

we have

$$\parallel \hat{\mathbf{w}}_{opt} \parallel \sqrt{2t}\eta R \geqslant \parallel \hat{\mathbf{w}}_{opt} \parallel \parallel \hat{\mathbf{w}}_t \parallel \geqslant \hat{\mathbf{w}}'_{opt}\hat{\mathbf{w}}_t \geqslant t\eta\gamma,$$

which implies

$$t \leqslant 2\left(\frac{R}{\gamma}\right)^2 \parallel \hat{\mathbf{w}}_{opt} \parallel^2 \leqslant \left(\frac{2R}{\gamma}\right)^2$$

because $b_{opt} \leqslant R$ for a non-trivial separation of data and hence

$$\parallel \hat{\mathbf{w}}_{opt} \parallel^2 \leqslant \parallel \mathbf{w}_{opt} \parallel^2 + 1 = 2.$$

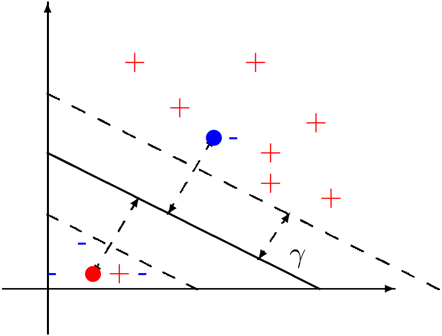
Definition

Let $\gamma > 0$. The margin slack variable of an example $(\mathbf{x}_i, y_i)$ with respect to a hyperplane H given by $\mathbf{w}'\mathbf{x} + b = 0$ and the target margin γ is

$$\xi((\mathbf{x}_i, y), H, \gamma) = \xi_i = \max\{0, \gamma - y_i(\mathbf{w}'\mathbf{x}_i + b)\}$$

- ξ_i measures how much a point fails to have a margin of γ from H; in any case, $\xi_i \geqslant \gamma - y_i(\mathbf{w}'\mathbf{x}_i + b)$, or $y_i(\mathbf{w}'\mathbf{x}_i + b) + \xi_i \geqslant \gamma$;
- if $\xi_i > \gamma$, then $\mathbf{x}_i$ is missclassified by H;
- the norm $\parallel \xi \parallel$ measures the amount by which the training sequence fails to have margin γ;
- points that are correctly classfied have their margin slack variable equal to 0.

The size of the margin slack variables for two missclassified examples for a hyperplane.



Remaining points have their slack variable equal to 0 since they have a margin larger than γ.

Freund-Shapire Theorem

Theorem

Let S be a non-trivial training sequence with no duplicate examples which is included in the ball $B(\mathbf{0}, R)$. If H is a hyperplane $\mathbf{w}'\mathbf{x} + b = 0$ with $\parallel \mathbf{w} \parallel = 1$ and $\gamma > 0$, let

$$D = \parallel \xi \parallel = \sqrt{\sum_{i=1}^{n} \xi_i^2}.$$

The number of mistakes in the first execution of the for loop of the perceptron algorithm is bounded by

$$\left(\frac{2(R+D)}{\gamma}\right)^2$$

Proof

Define a set of extended examples $\tilde{X}$ and an extended weight vector:

$$\tilde{\mathbf{x}}_i = \begin{pmatrix} \mathbf{x}_i \\ \mathbf{e}_i \Delta \end{pmatrix} = \begin{pmatrix} \mathbf{x}_i \\ 0 \\ \vdots \\ \Delta \\ \vdots \\ 0 \end{pmatrix} \text{ and } \tilde{\mathbf{w}} = \begin{pmatrix} \mathbf{w} \\ \frac{y_1 \xi_1}{\Delta} \\ \vdots \\ \frac{y_m \xi_m}{\Delta} \end{pmatrix},$$

where Δ is a parameter. Note that for $\tilde{R} = \max\{\| \tilde{\mathbf{x}}_i \| \mid 1 \leqslant i \leqslant m\}$ we have $\tilde{R}^2 = R^2 + \Delta^2$. Also, $\tilde{\mathbf{w}}'\tilde{\mathbf{x}}_i = \mathbf{w}'\mathbf{x}_i + y_i \xi_i$ and therefore,

$$\begin{aligned} y_i(\tilde{\mathbf{w}}'\tilde{\mathbf{x}}_i + b) &= y_i(\mathbf{w}'\mathbf{x}_i + y_i\xi_i + b) \\ &= y_i(\mathbf{w}'\mathbf{x}_i + b) + y_i^2\xi_i = y_i(\mathbf{w}'\mathbf{x}_i + b) + \xi_i \geqslant \gamma. \end{aligned}$$

Proof (cont'd)

- the inequality $y_i(\tilde{\mathbf{w}}'\tilde{\mathbf{x}}_i + b) \geqslant \gamma$ can be written as

$$y_i\left(\frac{1}{\parallel \tilde{\mathbf{w}} \parallel}\tilde{\mathbf{w}}'\tilde{\mathbf{x}}_i + \frac{1}{\parallel \tilde{\mathbf{w}} \parallel}b\right) \geqslant \frac{\gamma}{\parallel \tilde{\mathbf{w}} \parallel};$$

- $\parallel \tilde{\mathbf{w}} \parallel = \sqrt{\sum_{i=1}^{n} w_i^2 + \frac{D^2}{\Delta^2}} = \sqrt{1 + \frac{D^2}{\Delta^2}};$
- Rosenblatt's theorem can be applied if the norm of the optimal weight vector is 1 and this is case for $\parallel \tilde{\mathbf{w}} \parallel \tilde{\mathbf{w}}$; therefore, we need to replace the margin by

$$\tilde{\gamma} = \frac{\gamma}{\parallel \tilde{\mathbf{w}} \parallel} = \frac{\gamma}{\sqrt{1 + \frac{D^2}{\Delta^2}}}.$$

Proof (cont'd)

Since he training examples have non-zero entries in different coordinates, running the perceptron algorithm for the first **for** loop on $\tilde{X}$ has the same effect as running it on X, so the number of mistakes is bounded by

$$\begin{aligned}\left(\frac{2\tilde{R}}{\tilde{\gamma}}\right)^2 &= \frac{4(R^2+\Delta^2)(1+\frac{D^2}{\Delta^2})}{\gamma^2}\\ &= \frac{4}{\gamma}\left(R^2+D^2+\Delta^2+\frac{R^2D^2}{\Delta^2}\right).\end{aligned}$$

The optimal value is obtained when $\Delta = \sqrt{RD}$, which equals

$$\left(\frac{2(R+D)}{\gamma}\right)^2.$$

Remark

- since D can be defined for every hyperplane, the Freund-Shapire bound does not assume that the data is linearly separable;
- the perceptron algorithm works by adding missclassified positive examples and by subtracting missclassified negative examples to an initially arbitrary weight vector;
- if the initial weight vector is $\mathbf{0}$ the final weight vector is a linear combination of the examples

$$\mathbf{w} = \sum_{i=1}^{\ell} a_i y_i \mathbf{x}_i;$$

- a_i are positive values proportional to the number of times missclassifiation of $\mathbf{x}_i$ triggered updates; a_i is the embedding strength of $\mathbf{x}_i$.

The decision function is

$$\begin{aligned} h(\mathbf{x}) &= sign(\mathbf{w}'\mathbf{x} + b) \\ &= sign\left(\left(\sum_{i=1}^{\ell} a_i y_i \mathbf{x}_i\right)' \mathbf{x} + b\right) \\ &= sign\left(\sum_{i=1}^{\ell} a_i y_i (\mathbf{x}_i'\mathbf{x}) + b\right) \end{aligned}$$

which allows the expression of the perceptron algorithm in the dual form. Note that the learning rate does not appear in the dual form.

The Dual Perceptron Algorithm

Algorithm 1.2: Dual Learning Algorithm for Perceptron

Data: labelled training sequence S

Result: vector $\mathbf{a}$ and parameter b

```
initialize a = 0, b = 0;
R = max{‖ x_i ‖ | 1 ⩽ i ⩽ ℓ};
repeat
    for i = 1 to ℓ do
        if y_i ( Σ_{j=1}^{ℓ} a_j y_j x'_j x_i + b ) ⩽ 0 then
            a_i = a_i + 1;
            b = b + y_i R^2;
        end
    end
until no mistakes are made in the for loop ;
return a, b to define h;
```

$$\text{if } y_i\left(\sum_{j=1}^{\ell} a_j y_j \mathbf{x}_j' \mathbf{x}_i + b\right) \leqslant 0 \text{ then}$$

- the fact that points that are harder to learn have larger α_is can be used to rank the data according to their information content;
- since the number of updates equals the number of mistakes and each update causes 1 to be added to exactly one of its components, the 1-norm of α satisfies the inequality

$$\parallel \alpha \parallel_1 \leqslant \left(\frac{2R}{\gamma} \right)^2 ;$$

  this norm can be viewed as a measure of complexity of the target concept;
- the training data enter the algorithm through the matrix $G = (\mathbf{x}_i' \mathbf{x}_j)$, known as the Gram matrix.

Basic papers for perceptrons are [4] and [2].
Recommended references are [1] and [5].

N. Cristianini and J. Shawe-Taylor.
Support Vector Machines.
Cambridge, Cambridge, UK, 2000.

Y. Freund and R. E. Shapire.
Large margin classification using the perceptron algorithm.
Machine Learning, 37:277–296, 1999.

A. B. J. Novikoff.
On convergence proofs on perceptrons.
In *Proceedings of the Symposium on Mathematical Theory of Automata*.

F. Rosenblatt.
The perceptron: A probabilistic model for information storage and organization in the brain.
Psychological Review, 65:386–407, 1958.

J. Shawe-Taylor and N. Cristianini.
Kernel Methods for Pattern Analysis.
Cambridge, Cambridge, UK, 2004.