Chapter 6

Linear discriminant functions

In this chapter we look at the simplest possible boundary that can separate two classes - the straight line or equivalently the hyper-plane in higher dimensional space. The equation of any such hyper-plane will be: $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 = 0$ where $\mathbf{w}$ is the slope or weight vector and w_0 is the intercept or threshold. g is called a *discrimination/discriminant function*. Consider two points $\mathbf{x}_1$ and $\mathbf{x}_2$ on the hyper-plane. Both points satisfy $g(\mathbf{x}_1) = 0$ and $g(\mathbf{x}_2) = 0$ giving:

$$\begin{aligned}
&\mathbf{w}^T\mathbf{x}_1 + w_0 = \mathbf{w}^T\mathbf{x}_2 + w_0 \\
\Rightarrow &\mathbf{w}^T(\mathbf{x}_1 - \mathbf{x}_2) = 0 \\
\Rightarrow &\mathbf{w} \text{ is } \perp \text{ to } (\mathbf{x}_1 - \mathbf{x}_2) \\
\Rightarrow &\mathbf{w} \text{ is } \perp \text{ to the hyper-plane since } (\mathbf{x}_1 - \mathbf{x}_2) \text{ is a vector on the hyper-plane.}
\end{aligned}$$

Consider figure 6.1 where the hyper-plane is shown in red. The blue weight vector $\mathbf{w}$ is perpendicular to

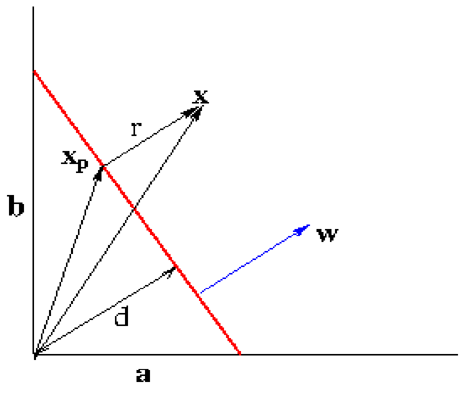

Figure 6.1: Relation between hyper-plane, $\mathbf{w}$, w_0 and an arbitrary vector $\mathbf{x}$.

the hyper-plane. Consider vector $\mathbf{x}$ where $\mathbf{x}_p$ is the point where the perpendicular from $\mathbf{x}$ intersects the hyper-plane. Vectorially, we can write $\mathbf{x} = \mathbf{x}_p + r\frac{\mathbf{w}}{\|\mathbf{w}\|}$. Note that $\frac{\mathbf{w}}{\|\mathbf{w}\|}$ is a unit vector $\perp$ to the hyper-plane

— that is in the direction of $\mathbf{w}$. Now,

$$\begin{aligned}
g(\mathbf{x}) &= g(\mathbf{x}_p + r\frac{\mathbf{w}}{\|\mathbf{w}\|}) \\
&= \mathbf{w}^T(\mathbf{x}_p + r\frac{\mathbf{w}}{\|\mathbf{w}\|}) + w_0 \\
&= \mathbf{w}^T\mathbf{x}_p + w_0 + r\frac{\mathbf{w}^T\mathbf{w}}{\|\mathbf{w}\|} \\
&= r\frac{\mathbf{w}^T\mathbf{w}}{\|\mathbf{w}\|} \text{ since } \mathbf{w}^T\mathbf{x}_p + w_0 = 0 \text{ because } \mathbf{x}_p \text{ is on the hyper-plane.} \\
r &= \frac{|g(\mathbf{x})|}{\|\mathbf{w}\|} \text{ Note that } \mathbf{w}^T\mathbf{w} = \|\mathbf{w}\|^2 \qquad (6.1)
\end{aligned}$$

The $\perp$ distance of any point $\mathbf{x}$ from the hyper-plane is given by equation 6.1. The intercepts a, b (see figure 6.1) on the X and Y axes are given by substituting $\mathbf{x} = (a, 0)$ and $\mathbf{x} = (0, b)$ in the hyper-plane equation. Assuming $\mathbf{w}^T = [w_1, w_2]$ we get:

$$\begin{aligned}
&\mathbf{w}^T[a, 0]^T + w_0 = 0 \\
&\Rightarrow a = -w_0/w_1 \\
&\text{Similarly, } b = -w_0/w_2
\end{aligned}$$

The distance d of the hyper-plane from the origin $\mathbf{x} = (0, 0)$ is: $|g([0, 0]^T)|/\|\mathbf{w}\|$ or $w_0/\|\mathbf{w}\|$.

Once we have the separating hyper-plane, assuming binary classification, the classification rule classifies all vectors on one side of the plane as ω_1 and the other side as ω_2. Since points on the hyper-plane satisfy $g(\mathbf{x}) = 0$ this can be written as:

$$\begin{cases} g(\mathbf{x}) > 0 \Rightarrow \omega_1 \\ g(\mathbf{x}) < 0 \Rightarrow \omega_2 \end{cases}$$

If $g(\mathbf{x}) = 0$ then either class label can be assigned.

6.1 Binary classification

We start by looking at classification tasks where there are only two classes — binary classification. We start with the separable case — that is we assume that the learning set $\mathcal{L}$ can be classified without error by some hyper-plane. The classification task will be solved by converting it into an optimization problem. Different objective functions will give us different algorithms. We can simplify notation by defining extended $\mathbf{w}$ and $\mathbf{x}$ vectors as $\mathbf{w}' = [w_1, \ldots, w_d, w_0]$ and $\mathbf{x}' = [x_1, \ldots, x_d, 1]$ respectively then the hyper-plane is defined by $\mathbf{w}'^T\mathbf{x}' = 0$.

6.1.1 Perceptron

One simple cost function is the number of misclassifications. Let $\mathbf{Y}$ be the set of vectors from $\mathcal{L}$ that are misclassified by the current hyper-plane. Then we can move the hyper-plane such that the size of $\mathbf{Y}$

decreases. We realize this by defining the perceptron cost function J which is a function of the current hyper-plane:

$$J(\mathbf{w}') = \sum_{\mathbf{x}' \in \mathbf{Y}} (\delta_x \mathbf{w}'^T \mathbf{x}') \tag{6.2}$$

where $\mathbf{Y}$ is the set of vectors misclassified by $\mathbf{w}'$ and δ_x is defined by:

$$\delta_x = \begin{cases} -1 & \text{if } \mathbf{x} \in \omega_1 \\ \;\;1 & \text{if } \mathbf{x} \in \omega_2 \end{cases}$$

Note that $J(\mathbf{w}')$ is always positive since the product of δ_x and $\mathbf{w}'^T\mathbf{x}'$ for misclassified vectors is always positive (recall that the classification rule is $g(x) > 0 \Rightarrow \omega_1,\ g(x) < 0 \Rightarrow \omega_2$. So, for a separable learning set $\mathcal{L}$ the minimum value of $J(\mathbf{x}')$ is 0. Notice that $J(\mathbf{x}')$ is a piece wise continuous function so we can use gradient descent to find the local minimum value of $J(\mathbf{x}')$. This gives us the iterator:

$$\begin{aligned} \mathbf{w}'(t+1) &= \mathbf{w}'(t) - \rho(t) \frac{\partial J(\mathbf{w}')}{\partial \mathbf{w}'}\bigg|_{\mathbf{w}'=\mathbf{w}'(t)} \\ &= \mathbf{w}'(t) - \rho(t) \sum_{\mathbf{x}' \in \mathbf{Y}} \delta_x \mathbf{x}' \end{aligned} \tag{6.3}$$

where $\rho(t)$ is a constant that depends on the iteration number. We can write the following simple algorithm to find a separating hyper-plane:

Algorithm 6.1.1: PERCEPTRON($\mathcal{L}$)

```
$\mathbf{w}' \leftarrow$ randomly generated vector
$\mathbf{Y} \leftarrow \{\mathbf{x} \in \mathcal{L} | \ \mathbf{x}'$ misclassified by the hyper-plane $\mathbf{w}'\}$
$\rho \leftarrow$ initial value   comment: Usually between 0 and 1. - $\frac{c}{t+\epsilon}$, $c, \epsilon > 0$ are constants; $t$ is iteration number.
while ($\mathbf{Y} \neq \Phi$)
    do
        $\mathbf{w}' \leftarrow \mathbf{w}' - \rho \sum_{\mathbf{x}' \in \mathbf{Y}} \delta_x \mathbf{x}'$   comment: Update $\mathbf{w}'$
        $\rho \leftarrow$ update $\rho$
        $\mathbf{Y} \leftarrow \{\mathbf{x}' \in \mathcal{L} | \ \mathbf{x}'$ misclassified by the hyper-plane$\}$
```

The major question regarding algorithm 6.1.1 is whether or not it will converge. We argue below that choosing ρ suitably will always lead to convergence.

A small detour on convergence

Let $e(\mathbf{w})$ be the cumulative error for weight vector $\mathbf{w}$ and learning set $\mathcal{L}$. Let $e_{min}(\mathbf{w}) = \inf_{\mathbf{w}} e(\mathbf{w})$ — it is the minimum possible error on $\mathcal{L}$ (could be non-zero if $\mathcal{L}$ is not separable).

Definition 2. *An algorithm is* **asymptotically optimal** *if* $\lim_{t\to\infty} P(e(\mathbf{w}(t)) - e_{min} < \eta) = 1$ *for every* $\eta > 0$.

Definition 3. *An algorithm is* **finitely optimal** *if* $\exists \hat{t} \ni P(e(\mathbf{w}(t)) = e_{min}, \forall t > \hat{t}) = 1$ *independent of* $\mathcal{L}$.

Proof of convergence for perceptron algorithm

Theorem 2. *The perceptron algorithm is finitely optimal.*

Proof. The basic idea of the proof is to argue that $\rho(t)$ can be chosen such that difference between the current vector $\mathbf{w}'(t)$ and a true separating hyper-plane $\mathbf{w}'^*$ will be consistently reduced in every iteration. So, we will argue that there exists a $\hat{t}$ such that when $t \geq \hat{t}$, $\|\mathbf{w}'(t) - \alpha\mathbf{w}'^*\| \leq 0$ where $\alpha > 0$ is a constant. This implies that $\mathbf{w}' = \mathbf{w}'^*$ since $\|.\|$ cannot be negative.

Let $\mathbf{w}'^*$ be a solution (i.e. separating hyper-plane) and $\alpha \in \mathcal{R},\ \alpha > 0$. Subtract $\alpha\mathbf{w}'^*$ from both sides of equation 6.3

$$\mathbf{w}'(t+1) - \alpha\mathbf{w}'^* = \mathbf{w}'(t) - \alpha\mathbf{w}'^* - \rho(t)\sum_{\mathbf{x}'\in\mathbf{Y}}\delta_x\mathbf{x}'$$

Take norm square on both sides:

$$\|\mathbf{w}'(t+1) - \alpha\mathbf{w}'^*\|^2 = \|\mathbf{w}'(t) - \alpha\mathbf{w}'^*\|^2 + \rho(t)^2\|\sum_{\mathbf{x}'\in\mathbf{Y}}\delta_x\mathbf{x}'\|^2 - 2\rho(t)\sum_{\mathbf{x}\in\mathbf{Y}}\delta_x(\mathbf{w}'(t) - \alpha\mathbf{w}'^*)^T\mathbf{x}'$$

Note that $\delta_x\mathbf{w}'^T\mathbf{x}' > 0$ for all $\mathbf{x} \in \mathbf{Y}$ since $\mathbf{x}$ is misclassified so $-2\rho(t)\sum_{\mathbf{x}\in\mathbf{Y}}\delta_x\mathbf{w}'^T\mathbf{x}' > 0$ and we can write the above equation as the following inequality after dropping the always negative term:

$$\|\mathbf{w}'(t+1) - \alpha\mathbf{w}'^*\|^2 \leq \|\mathbf{w}'(t) - \alpha\mathbf{w}'^*\|^2 + \rho(t)^2\|\sum_{\mathbf{x}\in\mathbf{Y}}\delta_x\mathbf{x}'\|^2 + 2\rho(t)\alpha\sum_{\mathbf{x}\in\mathbf{Y}}\delta_x\mathbf{w}'^{*T}\mathbf{x}' \tag{6.4}$$

Let

$$\beta = \max_{\mathbf{Y}\in\mathcal{P}(\mathcal{L}),\mathbf{Y}\neq\Phi}\|\sum_{\mathbf{x}\in\mathbf{Y}}\delta_x\mathbf{x}'\|$$

and

$$\gamma = \max_{\mathbf{Y}\in\mathcal{P}(\mathcal{L}),\mathbf{Y}\neq\Phi}\sum_{\mathbf{x}\in\mathbf{Y}}\delta_x\mathbf{w}'^{*T}\mathbf{x}'$$

where $\mathcal{P}(\mathcal{L})$ is the power set of $\mathcal{L}$.

Since $\mathbf{w}'^*$ is the separating hyper-plane every $\mathbf{x}$ is classified correctly and $\delta_x\mathbf{w}'^{*T}\mathbf{x}$ is always negative, consequently γ is always negative. So, equation 6.4 can be written as:

$$\|\mathbf{w}'(t+1) - \alpha\mathbf{w}'^*\|^2 \leq \|\mathbf{w}'(t) - \alpha\mathbf{w}'^*\|^2 + \rho(t)^2\beta^2 - 2\rho(t)\alpha|\gamma| \tag{6.5}$$

Choose $\alpha = \frac{\beta^2}{2|\gamma|}$, substiute for α on the right hand side of equation 6.5 (to get equation 6.6) and unfold from t to 0.

$$\begin{aligned}
\|\mathbf{w}'(t+1) - \alpha\mathbf{w}'^*\|^2 &\leq \|\mathbf{w}'(t) - \alpha\mathbf{w}'^*\|^2 + \beta^2[\rho(t)^2 - \rho(t)] &(6.6)\\
&\leq \|\mathbf{w}'(t-1) - \alpha\mathbf{w}'^*\|^2 + \beta^2[\rho(t)^2 + \rho(t-1)^2 - (\rho(t) + \rho(t-1))]\\
&\leq \dots\\
&\leq \dots\\
&\leq \|\mathbf{w}'(0) - \alpha\mathbf{w}'^*\|^2 + \beta^2[\sum_{k=0}^{t}\rho(k)^2 - \sum_{k=0}^{t}\rho(k)] &(6.7)
\end{aligned}$$

Now choose $\rho(t)$ such that $\lim_{t\to\infty} \sum_{k=0}^{t} \rho(k) = \infty$ (that is diverges) and $\lim_{t\to\infty} \sum_{k=0}^{t} \rho(k)^2 < \infty$ (that is converges). Then $\exists \hat{t}$ such that the rhs in equation 6.7 becomes ≤ 0 at $\hat{t}$. That is: $0 \leq \|\mathbf{w}'(\hat{t}+1) - \alpha \mathbf{w}'^*\|^2 \leq 0$ (note that a norm by definition must be ≥ 0). This implies that $\mathbf{w}'(\hat{t}) = \alpha \mathbf{w}'^*$ that means the algorithm converges. A possible choice for $\rho(t)$ is $\rho(t) = c/(t+\epsilon)$ where c, ϵ are constants. $\square$

Choosing a proper $\rho(t)$ is important for fast convergence. One can show that even if $\rho(t) = \rho$, ρ a suitably bounded constant the algorithm converges.

The algorithm above is called the batch perceptron algorithm. One can instead make an update to $\mathbf{w}'$ after every $\mathbf{x} \in \mathcal{L}$ is classified using the following update rule:

$$\mathbf{w}'(t+1) = \begin{cases} \mathbf{w}'(t) + \rho(t)\mathbf{x}'(t) & \mathbf{x}(t) \in \omega_1 \wedge \mathbf{w}'(t)^T \mathbf{x}'(t) \leq 0 \\ \mathbf{w}'(t) - \rho(t)\mathbf{x}'(t) & \mathbf{x}(t) \in \omega_2 \wedge \mathbf{w}'(t)^T \mathbf{x}'(t) > 0 \\ \mathbf{w}'(t) \text{ otherwise - do nothing if correctly classified.} \end{cases}$$

The perceptron algorithm with the above update rule also converges.

Perceptron algorithm for non-separable $\mathcal{L}$

A variant of the perceptron algorithm (pocket algorithm) converges, but only asymptotically, to an optimal solution (that is minimum error) even when $\mathcal{L}$ is not separable.

Algorithm 6.1.2: POCKET($\mathcal{L}$)

```
$\mathbf{w}'(0) \leftarrow$ random initialization
$\mathbf{w}'_s \leftarrow \mathbf{w}'(0)$   comment: stored or pocket weight vector
$h_s \leftarrow 0$   comment: history counter
$t \leftarrow 0$
while (not converged)
    do
        $\mathbf{w}'(t+1) \leftarrow$ update with perceptron rule
        $h \leftarrow$ number of vectors correctly classified by $\mathbf{w}'$(t+1)
        if ($h > h_s$)
            then
                $h_s \leftarrow h$
                $\mathbf{w}'_s \leftarrow \mathbf{w}'(t+1)$
        $t \leftarrow t+1$
return ($\mathbf{w}'_s$)
```

The convergence condition has to be based on the number of iterations for which $\mathbf{w}'_s$ has not changed. One can prove[1,2] that the pocket algorithm is asymptotically optimal — so it is not possible to put an apriori upper bound on the number of iterations of the while loop.

[1]SI Gallant, Perceptron based learning algorithm, IEEE Tran. Neural networks, 1(2), 179-191 (1990)

[2]M Muselli, On convergence properties of pocket algorithm, IEEE Tran. Neural networks, 8(3), 623-629, (1997)

6.1.2 Least mean square algorithm

Here, we try to minimize the expected square of the error. That is:

$$J(\mathbf{w}') = E[|y - \mathbf{w}'^T\mathbf{x}'|^2]$$
$$\hat{\mathbf{w}'} = \underset{\mathbf{w}'}{\text{argmin}}\ J(\mathbf{w}') \tag{6.8}$$

Minimizing $J(\mathbf{w}')$ gives

$$\frac{\partial J(\mathbf{w}')}{\partial \mathbf{w}'} = 2E[\mathbf{x}'(y - \mathbf{w}'^T\mathbf{x})] = 0 \tag{6.9}$$

This immediately gives $\hat{\mathbf{w}'} = R_{\mathbf{x}'}^{-1}E[\mathbf{x}'y]$, where $R_{\mathbf{x}'} \equiv E[\mathbf{x}'\mathbf{x}'^T]$. R_x is called the correlation or auto-correlation matrix:

$$\begin{bmatrix} E[\mathbf{x}'_1\mathbf{x}'_1] & \dots & E[\mathbf{x}'_1\mathbf{x}'_{d+1}] \\ \dots & \dots & \dots \\ \dots & \dots & \dots \\ E[\mathbf{x}'_{d+1}\mathbf{x}'_1] & \dots & E[\mathbf{x}'d+1\mathbf{x}'d+1] \end{bmatrix}$$

$E[\mathbf{x}'y]$ is the cross-correlation matrix:

$$E[\mathbf{x}'y] = E\left[\begin{bmatrix} \mathbf{x}'_1 y \\ \cdot \\ \cdot \\ \mathbf{x}'_{d+1}y \end{bmatrix}\right]$$

$\hat{\mathbf{w}'}$ can be obtained by solving the above set of linear equations provided $R_{\mathbf{x}'}^{-1}$ exists. The real problem with the above method is that it requires the calculation of the expectation which requires knowledge of the class conditional pdfs and these are not usually known.

Instead this can be calculated by a stochastic approximation scheme.

Stochastic approximation

Given $E[f(\mathbf{x}_k, \mathbf{w})] = 0,\ k = 1, 2, \dots$ where the $\mathbf{x}_k$s are a sequence of random vectors drawn iid from the distribution then one can approximate the unknown $\hat{\mathbf{w}}$ using the following iterative scheme[3]:

$$\hat{\mathbf{w}}(k+1) = \hat{\mathbf{w}}(k) + \rho(k) f(\mathbf{x}_k, \hat{\mathbf{w}}(k)) \tag{6.10}$$

One can prove that the iteration converges in probability ($\lim_{k\to\infty} P[\hat{\mathbf{w}}(k) - \mathbf{w}] = 1$) to the original equation provided: $\sum_{k=1}^{\infty} \rho(k) \to \infty$ and $\sum_{k=1}^{\infty} \rho(k)^2 < \infty$. This implies $\rho(k) \to 0$. It is also true that: $\lim_{k\to\infty} E[\|\hat{\mathbf{w}}(k) - \mathbf{w}\|^2] = 0$. We can use equation 6.10 to solve: $\frac{\partial J(\mathbf{w}')}{\partial \mathbf{w}'} = 2E[\mathbf{x}'(y - \mathbf{w}'^T\mathbf{x})] = 0$ giving $\hat{\mathbf{w}}(k+1) = \hat{\mathbf{w}}(k) + \rho(k)\mathbf{x}(y_k - \mathbf{x}^T\hat{\mathbf{w}}(k))$ where $(\mathbf{x}_k, y_k)$ are successive elements of $\mathcal{L}$. This if often called the Widrow-Hoff algorithm[4]

There are variants of LMS like using a fixed ρ instead of $\rho(k)$. However, this does not converge to the MSE solution. If $0 < \rho < 2/trace(R_{\mathbf{x}})$ then $E[\hat{\mathbf{w}}(k)] \to \mathbf{w}_{MSE}$ and $E[\|\hat{\mathbf{w}}(k) - \mathbf{w}_{MSE}\|^2] \to$ const, $\mathbf{w}_{MSE}$ is the MSE optimal estimate. The smaller ρ the smaller the variation about $\mathbf{w}_{MSE}$ but also slower the convergence.

[3]H Robbins, S Monroe, A stochastic approximation method, Ann. of Math. Stat. 22, 400-407, 1951.

[4]See - B Widrow, MA Lehr, 30 years of adaptive neural networks: Perceptron, madaline, and backpropagation, Proc. of the IEEE,78(9), 1415-1442, 1990.

6.1.3 Sum of squares algorithm

A closely related algorithm is the sum of error squared on $\mathcal{L}$.

$$J(\mathbf{w}') = \sum_{i=1}^{n}(y_i - \mathbf{w}'^T\mathbf{x}')^2 = \sum_{i=1}^{n} e_i^2 \tag{6.11}$$

In this case we do not need any knowledge of the pdfs. Minimizing with respect to $\mathbf{w}'$ gives:

$$\sum_{i=1}^{n}\mathbf{x}'_i(y_i - \mathbf{x}'^T\hat{\mathbf{w}'}) = 0 \text{ (Note: } \mathbf{w}'^T\mathbf{x}' = \mathbf{x}'^T\mathbf{w}') \tag{6.12}$$

$$(\sum_{i=1}^{n}\mathbf{x}'_i\mathbf{x}'^T_i)\hat{\mathbf{w}'} = \sum_{i=1}^{n}(\mathbf{x}'_i y_i)$$

Let $X = [\mathbf{x}'_1, \ldots, \mathbf{x}'_n]$ and $\mathbf{y} = [y_1, \ldots, y_n]^T$ where X is a $n \times (d+1)$ matrix. Then we have $\sum_{i=1}^{n}\mathbf{x}'_i\mathbf{x}'^T_i = X^TX$, $\sum_{i=1}^{n}\mathbf{x}'_i y_i = X^T\mathbf{y}$. So, equation 6.12 can be written as:

$$(X^TX)\hat{\mathbf{w}'} = X^T\mathbf{y}$$
$$\hat{\mathbf{w}'} = (X^TX)^{-1}X^T\mathbf{y}$$

where $\tilde{X} = (X^TX)^{-1}X^T$ is called the *pseudo inverse*. If $n = d+1$ then the pseudo inverse is a square matrix and $\hat{\mathbf{w}'} = \tilde{X}\mathbf{y}$ is a set of linear equations that can be solved to obtain $\hat{\mathbf{w}'}$. More generally, $\tilde{X}$ is not square as $n > d+1$ and there is no single solution. Then one calculates the limit: $\tilde{X} = \lim_{\epsilon\to 0}(X^TX + \epsilon I)^{-1}X^T$. One can prove that the limit always exists and $\hat{\mathbf{w}'}^T = \tilde{X}\mathbf{y}$ is an MSE solution to $\mathbf{y} = X\hat{\mathbf{w}'}$.