Ho-Kashyap Procedure (DHS 5.9.1)

Read Introduction in DHS 5.9.1. Check out the differences between the Perceptron and the MSE procedures in the case of linearly separable vs. nonseparable problems.

Task: Find $\underline{w}$ and $\underline{b}$ simultaneously

$Y\underline{w} = \underline{b} > 0$

$J(\underline{w},\underline{b}) = \| Y\underline{w} - \underline{b} \|^2$
Minimize J w.r.t. $\underline{w}$ and $\underline{b}$ with constraint $\underline{b}>\underline{0}$

$\nabla_{\underline{w}} J = 2\, Y^T(Y\underline{w} - \underline{b})$
$\nabla_{\underline{b}} J = -2\, (Y\underline{w} - \underline{b})$
$\underline{w}=\mathbf{Y}^{\dagger}\underline{b} \Rightarrow \nabla_{\underline{w}} J = 0$

Minimize J w.r.t. $\underline{b}$, with $\underline{w}=\mathbf{Y}^{\dagger}\underline{b}$, subject to constraint $\underline{b}>\underline{0}$

Start with $\underline{b}>\underline{0}$
Only add positive elements when updating $\underline{b}$

Gradient descent:
$b(k+1)=b(k)-1/2\alpha[\nabla_{\underline{b}} J - |\nabla_{\underline{b}} J|] \quad \alpha>0$

$1/2[a-|a|] = 0$ if $a\geq 0$
a if $a<0$ to make it sure a positive update
$|\underline{v}|$ means component-wise $|.|$ => $|\underline{v}|=[\ \ldots, |v_i|, \ldots]^T$

<u>Resulting Algorithm</u>

$\underline{b}(0)>\underline{0}$ but otherwise arbitrary
$\underline{w}(k)=Y^{\dagger}\underline{b}(k)$
Let $\underline{e}(k)=Y\underline{w}(k) - \underline{b}(k)$
$\underline{b}(k+1)=\underline{b}(k) + \alpha[\underline{e}(k)+|\underline{e}(k)|]$
$\alpha>0$

This is Ho-Kashyap Pseudoinverse.

Notes on Ho-Kashyap

1. Converges if samples are linearly separable (proved in DHS 5.9.2)
2. Generally required fewer steps to converge than Perceptron. However, each step requires more operations than Perceptron.
3. Update entire b and w, for both classes in each iteration
4. Nonseparability of data is indicated in the course of iterating. If e(k)<=0, not linearly separable.

Appropriate α

Option 1

$0<\alpha<2$

⇨ converges fastest

$\underline{w}(0)=(Y^TY)^{-1}Y^T\underline{b}(0)$

and $\underline{b}(0)=\underline{1}$

⇨ solution $\underline{w}(k)$ is the best linear square fit for a given $\underline{b}(k)$

Option 2

$0<\alpha<\|Y^TY\|^{-1}$

$\|.\|$ can be any of the following

$\|A\|=\sum_{ij}|a_{ij}|$

$\|A\|=\max_i \sum_{j=1}^{N}|a_{ij}|$

$\|A\|=\mathrm{tr}(AA^*)^{\frac{1}{2}}=\left[\sum_{ij}|a_{ij}|^2\right]^{\frac{1}{2}}$

This gives the simplest implementation but converges slower.

<u>Ho-Kashyap Convergence (DHS 5.9.2)</u>

If samples are linearly separable and if $0<\alpha<1$

- ➔ converges to solution in finite no of steps
- ➔ could add a halting condition for when prototypes are correctly classified

can show either

$e(k)=0$ within finite no of steps -> algorithm terminates with a solution vector

or $e(k)\to 0$ as $k\to\infty \Rightarrow Y\underline{w}(k)>\underline{0}$ after finite no of steps

Same convergence properties for linearly separable prototypes

Different options on parameter α

<u>Behavior of Ho-Kashyap Algorithm for Nonseparable Prototypes (DHS 5.9.3)</u>

- If obtain an $\underline{e}(k)$ or converge to an $\underline{e}(k)$ such that $\underline{e}(k)\neq 0$ and no components of $\underline{e}(k)$ are positive, then the prototypes are not linearly separable.
- If the prototypes are not linearly separable, then either the algorithm will yield an $\underline{e}(k)$ such that $\underline{e}(k)\neq 0$ with no positive components, or will asymptotically approach it: $\underline{e}(k)\to\underline{e}(\infty)\neq 0$ with no components of $\underline{e}(\infty)$ being >0

We have covered so far (see Table 5.1)

1. Fixed Increment in Perceptron
2. Variable Increment in Perceptron
3. Relaxation in Perceptron
4. Pseudo-Inverse
5. Windrow-Hoff
6. Ho-Kashyap

* Stochastic Approximation and Linear Programming (i.e., Simplex Algorithm) are not covered here.

Various Descent Algorithms

Table 5.1: Descent Procedures for Obtaining Linear Discriminant Functions

Name	Criterion	Algorithm	Conditions
Fixed Increment	$J_p = \sum_{\mathbf{a}^t\mathbf{y}\leq 0} (-\mathbf{a}^t\mathbf{y})$	$\mathbf{a}(k+1) = \mathbf{a}(k) + \mathbf{y}^k$ $(\mathbf{a}^t(k)\mathbf{y}^k \leq 0)$	—
Variable Increment	$J'_p = \sum_{\mathbf{a}^t\mathbf{y}\leq 0} -(\mathbf{a}^t\mathbf{y} - b)$	$\mathbf{a}(k+1) = \mathbf{a}(k) + \eta(k)\mathbf{y}^k$ $(\mathbf{a}^t(k)\mathbf{y}^k \leq b)$	$\eta(k) \geq 0$ $\sum \eta(k) \to \infty$ $\frac{\sum \eta^2(k)}{(\sum \eta(k))^2} \to 0$
Relaxation	$J_r = \frac{1}{2}\sum_{\mathbf{a}^t\mathbf{y}\leq b} \frac{(\mathbf{a}^t\mathbf{y}-b)^2}{\|\mathbf{y}\|^2}$	$\mathbf{a}(k+1) = \mathbf{a}(k) + \eta\frac{b-\mathbf{a}^t(k)\mathbf{y}^k}{\|\mathbf{y}^k\|^2}\mathbf{y}^k$ $(\mathbf{a}^t(k)\mathbf{y}^k \leq b)$	$0 < \eta < 2$
Widrow-Hoff (LMS)	$J_s = \sum_i (\mathbf{a}^t\mathbf{y}_i - b_i)^2$	$\mathbf{a}(k+1) =$ $\mathbf{a}(k) + \eta(k)(b_k - \mathbf{a}^t(k)\mathbf{y}^k)\mathbf{y}^k$	$\eta(k) > 0$ $\eta(k) \to 0$
Stochastic Approx.	$J_m = \mathcal{E}\left[(\mathbf{a}^t\mathbf{y} - z)^2\right]$	$\mathbf{a}(k+1) =$ $\mathbf{a}(k) + \eta(k)(z_k - \mathbf{a}^t(k)\mathbf{y}^k)\mathbf{y}^k$	$\sum \eta(k) \to \infty$ $\sum \eta^2(k) \to L < \infty$
		$\mathbf{a}(k+1) =$ $\mathbf{a}(k) + \mathbf{R}(k)(z(k) - \mathbf{a}(k)^t\mathbf{y}^k)\mathbf{y}^k$	$\mathbf{R}^{-1}(k+1) = \mathbf{R}^{-1}(k) + \mathbf{y}_k\mathbf{y}_k^t$
Pseudo-inverse	$J_s = \|\mathbf{Ya} - \mathbf{b}\|^2$	$\mathbf{a} = \mathbf{Y}^\dagger\mathbf{b}$	—
Ho-Kashyap	$J_s = \|\mathbf{Ya} - \mathbf{b}\|^2$	$\mathbf{b}(k+1) = \mathbf{b}(k) + \eta(\mathbf{e}(k) + \|\mathbf{e}(k)\|)$ $\mathbf{e}(k) = \mathbf{Ya}(k) - \mathbf{b}(k)$ $\mathbf{a}(k) = \mathbf{Y}^\dagger\mathbf{b}(k)$	$0 < \eta < 1$ $\mathbf{b}(1) > 0$
		$\mathbf{b}(k+1) = \mathbf{b}(k) + \eta(\mathbf{e}(k) + (\|\mathbf{e}(k)\|)$ $\mathbf{a}(k+1) = \mathbf{a}(k) + \eta\mathbf{RY}^t\|\mathbf{e}(k)\|$	$\eta(k) =$ $\frac{\|\mathbf{e}(k)\|^t\mathbf{YRY}^t\|\mathbf{e}(k)\|}{\|\mathbf{e}(k)\|^t\mathbf{YRY}^t\mathbf{YRY}^t\|\mathbf{e}(k)\|}$ is optimum; $\mathbf{R}$ sym., pos. def.; $\mathbf{b}(1) > 0$
Linear Programming	$\tau = \max_{\mathbf{a}^t\mathbf{y}_i\leq b_i}[-(\mathbf{a}^t\mathbf{y}_i - b_i)]$	Simplex algorithm	$\mathbf{a}^t\mathbf{y}_i + \tau \geq b_i$ $b \geq 0$
	$J'_p = \sum_{i=1}^{n} \tau_i$ $= \sum_{\mathbf{a}^t\mathbf{y}_i\leq b_i} -(\mathbf{a}^t\mathbf{y}_i - b_i)$	Simplex algorithm	$\mathbf{a}^t\mathbf{y}_i + \tau \geq b_i$ $b \geq 0$

Support Vector Machines (or Maximum Margin Classifier) (DHS 5.11)

Concepts

- Recall linear machines with margins.
- SVMs are very much similar, but rely on preprocessing the data to represent patterns in a high dimension (much higher than original feature space)
- Typically a nonlinear mapping function (or a kernel function) $\phi(.)$ is used. Thus transform a pattern x_k to $y_k = \phi(x_k)$.
- A linear discriminant can be expressed as $g(y_k) = w^T y_k$ in an augmented space.
- The goal of a SVM is to find a separating hyperplane with the largest margin.
- The support vectors are the training samples that define optimal separating hyperplane.
- The support vectors are the most difficult patterns to classify.
- See Fig. 5.19

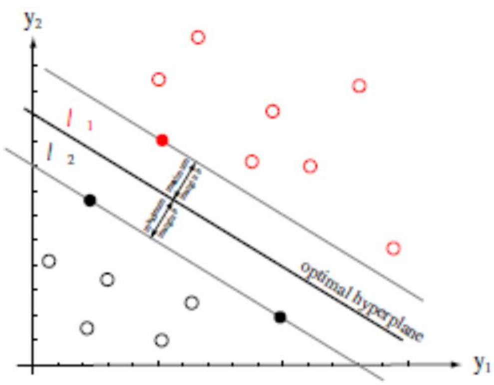


Figure 5.19: Training a Support Vector Machine consists of finding the optimal hyperplane, i.e., the one with the maximum distance from the nearest training patterns. The support vectors are those (nearest) patterns, a distance b from the hyperplane. The three support vectors are shown in solid dots.

Methods

- Modify the familiar Perceptron algorithm: train with the current worst-classified patterns. Of course finding the worst-classified patterns is difficult (computationally expensive)
- Training an SVM
  - Use the method of Lagrange Multipliers (not the focus of this class)
  - The cost function

    $$L(w,\alpha)=\frac{1}{2}\|w\|^2-\sum_{k=1}^{n}\alpha_k[z_k w^T y_k-1] \text{ with } z_k=\pm 1$$

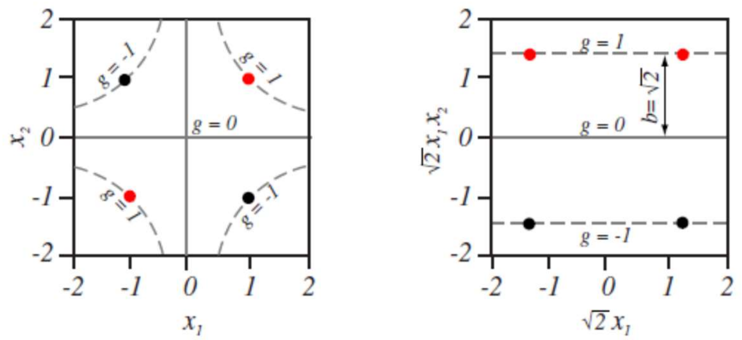
    Minimize L w.r.t. the weight vector w, and maximize it w.r.t. the multipliers $\alpha_k > 0$
  - This problem can be reformulated through the Kuhn-Tucker condition as

    Maximizing $L(\alpha)=\sum_{k=1}^{n}\alpha_i-\frac{1}{2}\sum_{k,j}^{n}\alpha_k\alpha_j z_k z_j y_j^T y_k$ with the constraints

    $\sum_{k=1}^{n} z_k\alpha_k = 0$, $\alpha_k \geq 0$, $k=1,\ldots,n$

Example

- Example 2 (DHS p. 264)



The XOR problem in the original $x_1 - x_2$ feature space is shown at the left; the two red patterns are in category ω_1 and the two black ones in ω_2. These four training patterns x are mapped to a six-dimensional space by 1, $\sqrt{2}x_1$, $\sqrt{2}x_2$, $\sqrt{2}x_1x_2$, x_1^2 and x_2^2. In this space, the optimal hyperplane is found to be $g(x_1, x_2) = x_1x_2 = 0$ and the margin is $b = \sqrt{2}$. A two-dimensional projection of this space is shown at the right. The hyperplanes through the support vectors are $\sqrt{2}x_1x_2 = \pm 1$, and correspond to the hyperbolas $x_1x_2 = \pm 1$ in the original feature space, as shown.

- Try "svmtrain" under Bioinformatics Toolbox of Matlab