Minimum Squared Error

LDF: Minimum Squared-Error Procedures

- Idea: convert to easier and better understood problem

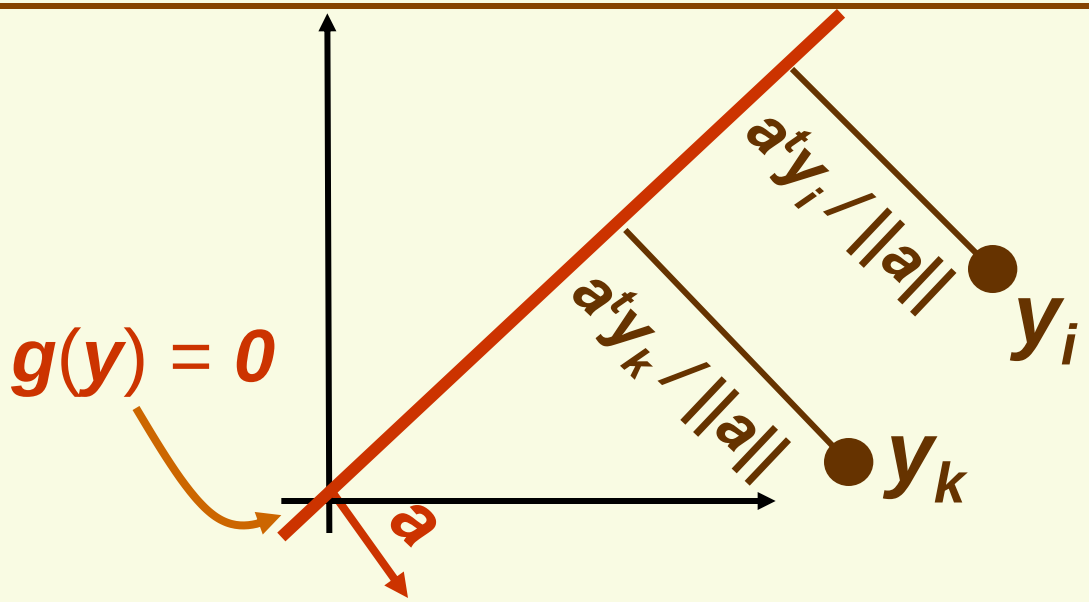
Perceptron → $a^t y_i > 0$ for all samples y_i
solve system of linear inequalities

⇓

$a^t y_i = b_i$ for all samples y_i
solve system of linear equations

- MSE procedure
  - Choose **positive** constants $b_1, b_2, \ldots, b_n$
  - try to find weight vector a s.t. $a^t y_i = b_i$ for all samples y_i
  - If we can find weight vector a such that $a^t y_i = b_i$ for all samples y_i , then a is a solution because b_i's are positive
  - consider all the samples (not just the misclassified ones)

LDF: MSE Margins



- Since we want $\boldsymbol{a}^t\boldsymbol{y}_i = \boldsymbol{b}_i$, we expect sample $\boldsymbol{y}_i$ to be at distance $\boldsymbol{b}_i$ from the separating hyperplane (normalized by $\|\boldsymbol{a}\|$)
- Thus $\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_n$ give relative expected distances or "margins" of samples from the hyperplane
- Should make $\boldsymbol{b}_i$ small if sample $\boldsymbol{i}$ is expected to be near separating hyperplane, and make $\boldsymbol{b}_i$ larger otherwise
- In the absence of any additional information, there are good reasons to set $\boldsymbol{b}_1 = \boldsymbol{b}_2 = \ldots = \boldsymbol{b}_n = \boldsymbol{1}$

LDF: MSE Matrix Notation

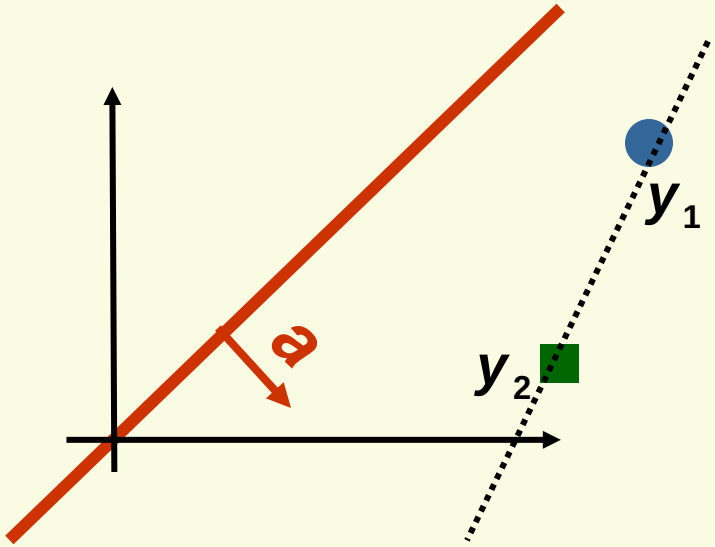
- Need to solve ***n*** equations $\begin{cases} a^t y_1 = b_1 \\ \vdots \\ a^t y_n = b_n \end{cases}$
- Introduce matrix notation:

$$\underbrace{\begin{bmatrix} y_1^{(0)} & y_1^{(1)} & \cdots & y_1^{(d)} \\ y_2^{(0)} & y_2^{(1)} & \cdots & y_2^{(d)} \\ \vdots & & & \vdots \\ y_n^{(0)} & y_n^{(1)} & \cdots & y_n^{(d)} \end{bmatrix}}_{Y} \underbrace{\begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_d \end{bmatrix}}_{a} = \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}}_{b}$$

- Thus need to solve a linear system ***Ya*** = ***b***

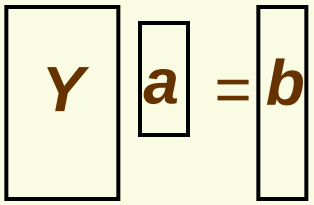
LDF: Exact Solution is Rare

- Thus need to solve a linear system $\boldsymbol{Ya} = \boldsymbol{b}$
  - $\boldsymbol{Y}$ is an $\boldsymbol{n}$ by $(\boldsymbol{d}+\boldsymbol{1})$ matrix
- Exact solution can be found only if $\boldsymbol{Y}$ is nonsingular and square, in which case the inverse $\boldsymbol{Y^{-1}}$ exists
  - $\boldsymbol{a} = \boldsymbol{Y^{-1}b}$
  - (number of samples) = (number of features + 1)
  - almost never happens in practice
  - in this case, guaranteed to find the separating hyperplane



LDF: Approximate Solution

- Typically ***Y*** is overdetermined, that is it has more rows (examples) than columns (features)
  - If it has more features than examples, should reduce dimensionality

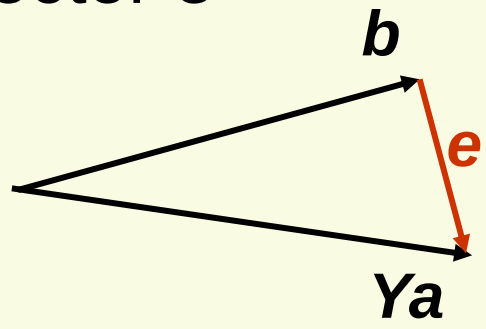


- Need ***Ya*** = ***b***, but no exact solution exists for an overdetermined system of equation
  - More equations than unknowns
- Find an approximate solution ***a***, that is $\boldsymbol{Ya \approx b}$
  - Note that approximate solution ***a*** ***does not*** necessarily give the separating hyperplane in the separable case
  - But hyperplane corresponding to ***a*** may still be a good solution, especially if there is no separating hyperplane

LDF: MSE Criterion Function

- Minimum squared error approach: find $\boldsymbol{a}$ which minimizes the length of the error vector $\boldsymbol{e}$

$$\boldsymbol{e} = \boldsymbol{Ya} - \boldsymbol{b}$$



- Thus minimize the *minimum squared error* criterion function:

$$J_s(\boldsymbol{a}) = \|\boldsymbol{Ya} - \boldsymbol{b}\|^2 = \sum_{i=1}^{n} \left(\boldsymbol{a}^t \boldsymbol{y}_i - b_i\right)^2$$

- Unlike the perceptron criterion function, we can optimize the minimum squared error criterion function analytically by setting the gradient to $\boldsymbol{0}$

LDF: Optimizing $J_s(a)$

$$J_s(a) = \|Ya - b\|^2 = \sum_{i=1}^{n} \left(a^t y_i - b_i\right)^2$$

- Let's compute the gradient:

$$\nabla J_s(a) = \begin{bmatrix} \frac{\partial J_s}{\partial a_0} \\ \vdots \\ \frac{\partial J_s}{\partial a_d} \end{bmatrix} = 2Y^t(Ya - b)$$

- Setting the gradient to 0:

$$2Y^t(Ya - b) = 0 \Rightarrow Y^tYa = Y^tb$$

LDF: Pseudo Inverse Solution

- Matrix $\boldsymbol{Y^tY}$ is square (it has $\boldsymbol{d+1}$ rows and columns) and it is often non-singular
- If $\boldsymbol{Y^tY}$ is non-singular, its inverse exists and we can solve for $\boldsymbol{a}$ uniquely:

$$\boldsymbol{a} = \boxed{\left(\boldsymbol{Y}^t\boldsymbol{Y}\right)^{-1}\boldsymbol{Y}^t}\,\boldsymbol{b}$$

pseudo inverse of $\boldsymbol{Y}$

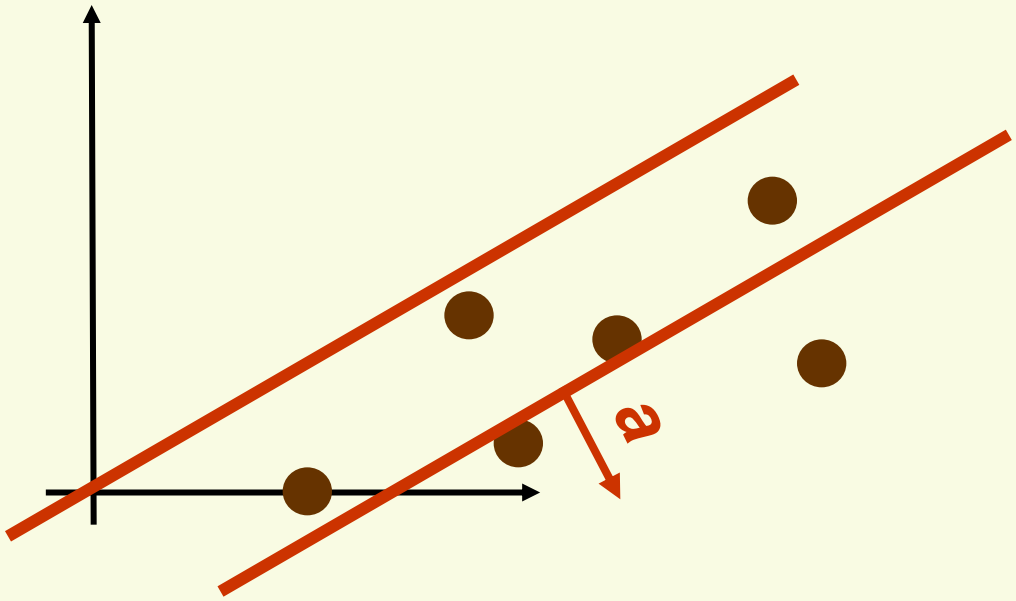
$$\left(\left(\boldsymbol{Y}^t\boldsymbol{Y}\right)^{-1}\boldsymbol{Y}^t\right)\boldsymbol{Y} = \left(\boldsymbol{Y}^t\boldsymbol{Y}\right)^{-1}\left(\boldsymbol{Y}^t\boldsymbol{Y}\right) = \boldsymbol{I}$$

LDF: Minimum Squared-Error Procedures

- If $b_1 = \ldots = b_n = 1$, MSE procedure is equivalent to finding a hyperplane of best fit through the samples $y_1, \ldots, y_n$

$$J_s(a) = \|Ya - 1_n\|^2$$

$$1_n = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \Big\} n$$



- Then we shift this line to the origin, if this line was a good fit, all samples will be classified correctly

LDF: Minimum Squared-Error Procedures

- Only guaranteed the separating hyperplane if $\boldsymbol{Ya} > \boldsymbol{0}$
  - that is if all elements of vector $\boldsymbol{Ya} = \begin{bmatrix} \boldsymbol{a}^t \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{a}^t \boldsymbol{y}_n \end{bmatrix}$ are positive
- We have $\boldsymbol{Ya} \approx \boldsymbol{b}$
- That is $\boldsymbol{Ya} = \begin{bmatrix} \boldsymbol{b}_1 + \varepsilon_1 \\ \vdots \\ \boldsymbol{b}_n + \varepsilon_n \end{bmatrix}$ where ε may be negative
  - If $\varepsilon_1, \ldots, \varepsilon_n$ are small relative to $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$, then each element of $\boldsymbol{Ya}$ is positive, and $\boldsymbol{a}$ gives a separating hyperplane
  - If approximation is not good, ε_i may be large and negative, for some $\boldsymbol{i}$, thus $\boldsymbol{b}_i + \varepsilon_i$ will be negative and $\boldsymbol{a}$ is not a separating hyperplane
- Thus in linearly separable case, least squares solution $\boldsymbol{a}$ does *not necessarily* gives separating hyperplane
- But it will give a "reasonable" hyperplane

LDF: Minimum Squared-Error Procedures

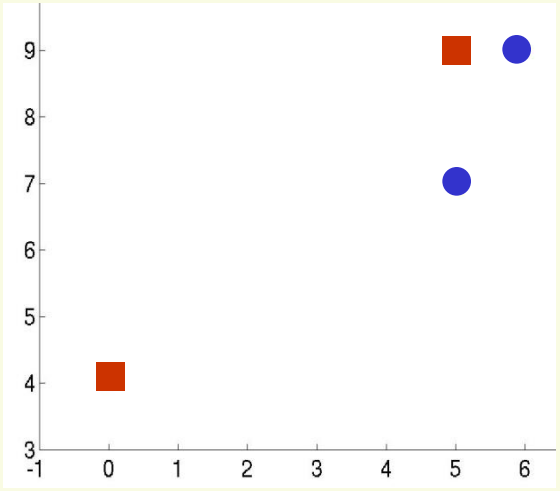
- We are free to choose $\boldsymbol{b}$. May be tempted to make $\boldsymbol{b}$ large as a way to insure $\boldsymbol{Ya} \approx \boldsymbol{b} > \boldsymbol{0}$
- Does not work
  - Let β be a scalar, let's try $\beta\boldsymbol{b}$ instead of $\boldsymbol{b}$
  - if $\boldsymbol{a}^*$ is a least squares solution to $\boldsymbol{Ya} = \boldsymbol{b}$, then for any scalar β, least squares solution to $\boldsymbol{Ya} = \beta\boldsymbol{b}$ is $\beta\boldsymbol{a}^*$

$$\arg\min_{\boldsymbol{a}} \|\boldsymbol{Ya} - \beta\boldsymbol{b}\|^2 = \arg\min_{\boldsymbol{a}} \beta^2 \|\boldsymbol{Y}(\boldsymbol{a}/\beta) - \boldsymbol{b}\|^2$$

$$= \arg\min_{\boldsymbol{a}} \|\boldsymbol{Y}(\boldsymbol{a}/\beta) - \boldsymbol{b}\|^2 = \beta\boldsymbol{a}^*$$

  - thus if for some $\boldsymbol{i}$th element of $\boldsymbol{Ya}$ is less than 0, that is $\boldsymbol{y}^t_i\boldsymbol{a} < \boldsymbol{0}$, then $\boldsymbol{y}^t_i(\beta\boldsymbol{a}) < \boldsymbol{0}$,
- Relative difference between components of $\boldsymbol{b}$ matters, but not the size of each individual component

LDF: Example



- Class 1: (6 9), (5 7)
- Class 2: (5 9), (0 4)
- Set vectors y_1, y_2, y_3, y_4 by adding extra feature and "normalizing"

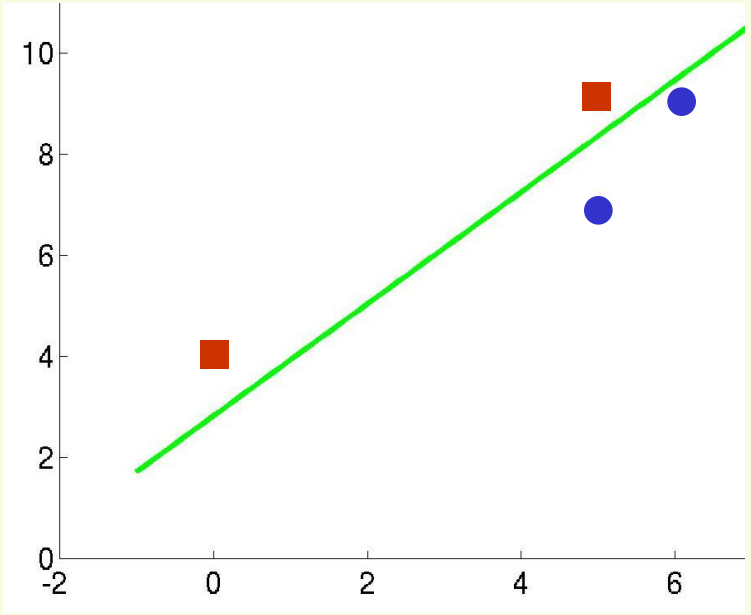
$$y_1 = \begin{bmatrix} 1 \\ 6 \\ 9 \end{bmatrix} \quad y_2 = \begin{bmatrix} 1 \\ 5 \\ 7 \end{bmatrix} \quad y_3 = \begin{bmatrix} -1 \\ -5 \\ -9 \end{bmatrix} \quad y_4 = \begin{bmatrix} -1 \\ 0 \\ -4 \end{bmatrix}$$

- Matrix Y is then

$$Y = \begin{bmatrix} 1 & 6 & 9 \\ 1 & 5 & 7 \\ -1 & -5 & -9 \\ -1 & 0 & -4 \end{bmatrix}$$

LDF: Example

- Choose $\boldsymbol{b} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$



- In matlab, $\boldsymbol{a}=\boldsymbol{Y}\backslash\boldsymbol{b}$ solves the least squares problem

$$\boldsymbol{a} = \begin{bmatrix} 2.7 \\ 1.0 \\ -0.9 \end{bmatrix}$$

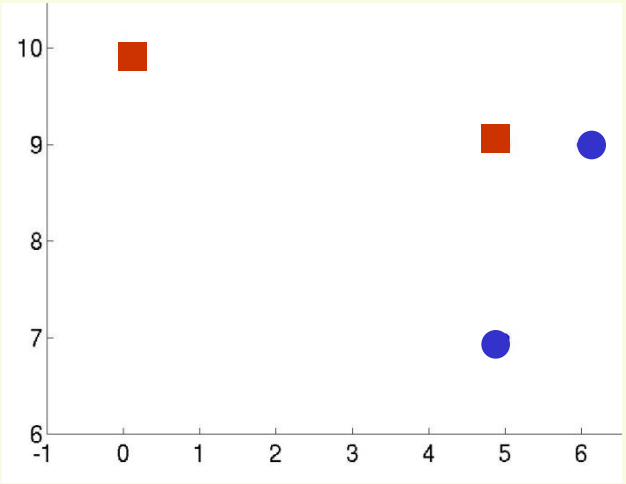
- Note $\boldsymbol{a}$ is an approximation to $\boldsymbol{Ya} = \boldsymbol{b}$, since no exact solution exists

$$\boldsymbol{Ya} = \begin{bmatrix} 0.4 \\ 1.3 \\ 0.6 \\ 1.1 \end{bmatrix} \neq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

- This solution does give a separating hyperplane since $\boldsymbol{Ya} > \boldsymbol{0}$

LDF: Example

- Class 1: (6 9), (5 7)
- Class 2: (5 9), (0 10)
- The last sample is very far compared to others from the separating hyperplane

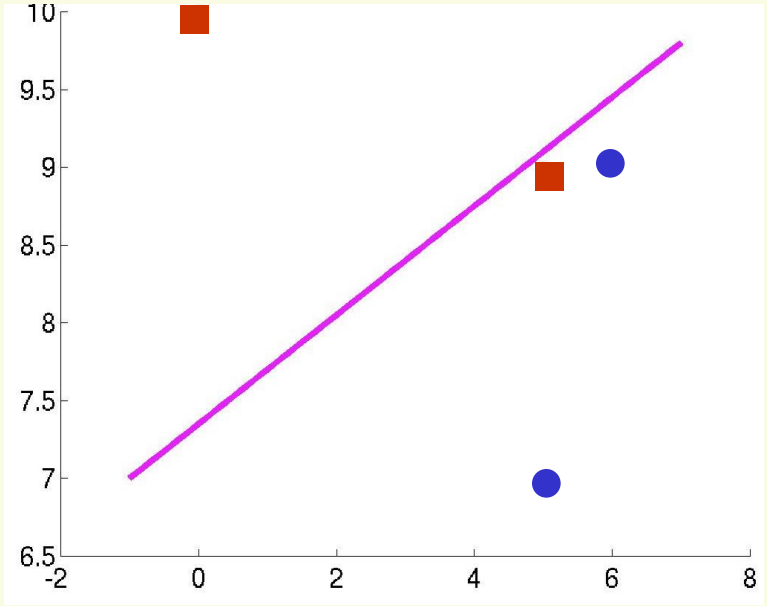


$$y_1 = \begin{bmatrix} 1 \\ 6 \\ 9 \end{bmatrix} \quad y_2 = \begin{bmatrix} 1 \\ 5 \\ 7 \end{bmatrix} \quad y_3 = \begin{bmatrix} -1 \\ -5 \\ -9 \end{bmatrix} \quad y_4 = \begin{bmatrix} -1 \\ 0 \\ -10 \end{bmatrix}$$

- Matrix $Y = \begin{bmatrix} 1 & 6 & 9 \\ 1 & 5 & 7 \\ -1 & -5 & -9 \\ -1 & 0 & -10 \end{bmatrix}$

LDF: Example

- Choose $\boldsymbol{b} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$



- In matlab, $\boldsymbol{a}=\boldsymbol{Y}\backslash\boldsymbol{b}$ solves the least squares problem

$$\boldsymbol{a} = \begin{bmatrix} 3.2 \\ 0.2 \\ -0.4 \end{bmatrix}$$

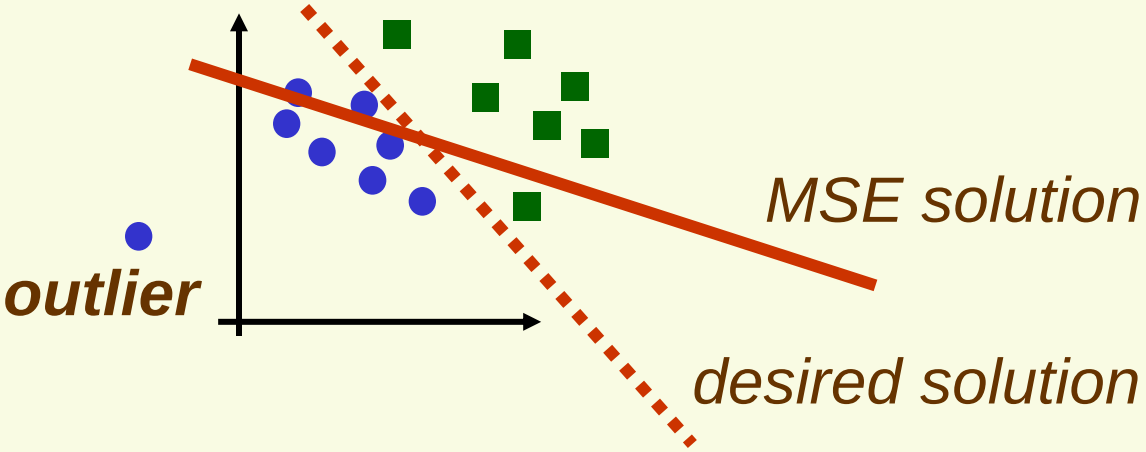
- Note $\boldsymbol{a}$ is an approximation to $\boldsymbol{Ya} = \boldsymbol{b}$, since no exact solution exists

$$\boldsymbol{Ya} = \begin{bmatrix} 0.2 \\ 0.9 \\ -0.04 \\ 1.16 \end{bmatrix} \neq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

- This solution does not give a separating hyperplane since $\boldsymbol{a}^t\boldsymbol{y}_3 < 0$

LDF: Example

- MSE pays to much attention to isolated “noisy” examples (such examples are called outliers)

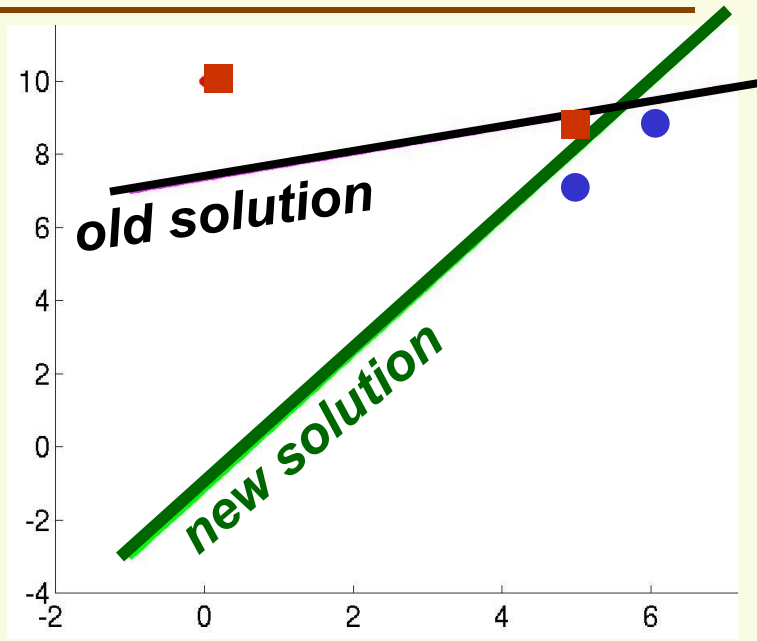


- No problems with convergence though, and solution it gives ranges from reasonable to good

LDF: Example

- we know that 4th point is far far from separating hyperplane
  - In practice we don't know this
- Thus appropriate $\boldsymbol{b} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 10 \end{bmatrix}$
- In Matlab, solve $\boldsymbol{a}=\boldsymbol{Y}\backslash\boldsymbol{b}$

$$\boldsymbol{a} = \begin{bmatrix} -1.1 \\ 1.7 \\ -0.9 \end{bmatrix}$$



- Note $\boldsymbol{a}$ is an approximation to $\boldsymbol{Ya} = \boldsymbol{b}$, $\boldsymbol{Ya} = \begin{bmatrix} 0.9 \\ 1.0 \\ 0.8 \\ 10.0 \end{bmatrix} \neq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 10 \end{bmatrix}$
- This solution does give the separating hyperplane since $\boldsymbol{Ya} > \boldsymbol{0}$

LDF: Gradient Descent for MSE solution

$$J_s(\boldsymbol{a}) = \|\boldsymbol{Ya} - \boldsymbol{b}\|^2$$

- May wish to find MSE solution by gradient descent:
  1. Computing the inverse of $\boldsymbol{Y}^t\boldsymbol{Y}$ may be too costly
  2. $\boldsymbol{Y}^t\boldsymbol{Y}$ may be close to singular if samples are highly correlated (rows of $\boldsymbol{Y}$ are almost linear combinations of each other)
     - computing the inverse of $\boldsymbol{Y}^t\boldsymbol{Y}$ is not numerically stable

- In the beginning of the lecture, computed the gradient:

$$\nabla J_s(\boldsymbol{a}) = 2\boldsymbol{Y}^t(\boldsymbol{Ya} - \boldsymbol{b})$$

LDF: Widrow-Hoff Procedure

$$\nabla J_s(\boldsymbol{a}) = 2\boldsymbol{Y}^t(\boldsymbol{Ya} - \boldsymbol{b})$$

- Thus the update rule for gradient descent:

$$\boldsymbol{a}^{(k+1)} = \boldsymbol{a}^{(k)} - \eta^{(k)}\boldsymbol{Y}^t\left(\boldsymbol{Ya}^{(k)} - \boldsymbol{b}\right)$$

  - If $\eta^{(k)} = \eta^{(1)}/k$ weight vector $\boldsymbol{a}^{(k)}$ converges to the MSE solution $\boldsymbol{a}$, that is $\boldsymbol{Y}^t(\boldsymbol{Ya}-\boldsymbol{b})=0$

- *Widrow-Hoff procedure* reduces storage requirements by considering single samples sequentially:

$$\boldsymbol{a}^{(k+1)} = \boldsymbol{a}^{(k)} - \eta^{(k)}\boldsymbol{y}_i\left(\boldsymbol{y}_i^t\boldsymbol{a}^{(k)} - \boldsymbol{b}_i\right)$$

LDF: MSE for Multiple Classes

- Suppose we have m classes
- Define m linear discriminant functions

$$g_i(x) = w_i^t x + w_{i0} \qquad i = 1,\ldots,m$$

- Given x, assign class c_i if

$$g_i(x) \geq g_j(x) \qquad \forall j \neq i$$

- Such classifier is called a ***linear machine***

- A linear machine divides the feature space into c decision regions, with $g_i(x)$ being the largest discriminant if x is in the region R_i

LDF: MSE for Multiple Classes

- For each class i, find weight vector a_i, s.t.

$$\begin{cases} a_i^t y = 1 & \forall y \in \text{class i} \\ a_i^t y = 0 & \forall y \notin \text{class i} \end{cases}$$

- Let Y_i be matrix whose rows are samples from class i, so it has $d+1$ columns and n_i rows

- Let's pile all samples in n by $d+1$ matrix Y:

$$Y = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_m \end{bmatrix} = \begin{bmatrix} \text{sample from class 1} \\ \text{sample from class 1} \\ \vdots \\ \text{sample from class } m \\ \text{sample from class } m \end{bmatrix}$$

LDF: MSE for Multiple Classes

- Let $\boldsymbol{b_i}$ be a column vector of length $\boldsymbol{n}$ which is $\boldsymbol{0}$ everywhere except rows corresponding to samples from class $\boldsymbol{i}$, where it is $\boldsymbol{1}$:

$$\boldsymbol{b_i} = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix}$$

rows corresponding to samples from class i

LDF: MSE for Multiple Classes

- Let's pile all b_i as columns in n by c matrix B

$$B = [b_1 \ \cdots \ b_n]$$

- Let's pile all a_i as columns in $d+1$ by m matrix A

$$A = [a_1 \ \cdots \ a_m] = \begin{bmatrix} \text{weights } a_1 & \text{weights } a_2 & & \text{weights } a_m \end{bmatrix}$$

- m LSE problems can be represented in $YA = B$:

$$\begin{bmatrix} \text{sample from class 1} \\ \text{sample from class 1} \\ \text{sample from class 2} \\ \text{sample from class 3} \\ \text{sample from class 3} \\ \text{sample from class 3} \end{bmatrix} \begin{bmatrix} \text{weights for c1} & \text{weights for c2} & \text{weights for c3} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

Y A B

LDF: MSE for Multiple Classes

- Our objective function is:

$$J(A) = \sum_{i=1}^{m} \|Ya_i - b_i\|^2$$

- $J(A)$ is minimized with the use of pseudoinverse

$$A = \left(Y^t Y\right)^{-1} YB$$

LDF: Summary

- ***Perceptron*** procedures
  - find a separating hyperplane in the linearly separable case,
  - do not converge in the non-separable case
  - can force convergence by using a decreasing learning rate, but are not guaranteed a reasonable stopping point

- ***MSE*** procedures
  - converge in separable and not separable case
  - may not find separating hyperplane if classes are linearly separable
  - use pseudoinverse if $\boldsymbol{Y^tY}$ is not singular and not too large
  - use gradient descent (Widrow-Hoff procedure) otherwise