CS434b/641a: Pattern Recognition
Prof. Olga Veksler

Lecture 10

Today

- Continue with Linear Discriminant Functions
    - Last lecture: Perceptron Rule for weight learning
    - This lecture: Minimum Squared Error (MSE) rule
        - Pseudoinverse
        - Gradient descent (Widrow-Hoff Procedure)
   - Ho-Kashyap Procedure

LDF: Perceptron Criterion Function

- The perceptron criterion function
  - try to find weight vector a s.t. $a^t y_i > 0$ for all samples y_i
  - perceptron criterion function $J_p(a) = \sum_{y \in Y_M} (-a^t y)$
  - only look at the misclassified samples
  - will converge in the linearly separable case
- Problem:
  - will not converge in the nonseparable case
  - to ensure convergence can set

    $$\eta^{(k)} = \frac{\eta^{(1)}}{k}$$

  - However we are not guaranteed that we will stop at a good point
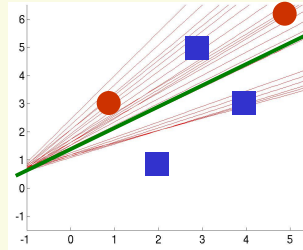
LDF: Minimum Squared-Error Procedures

- Idea: convert to easier and better understood problem

  > $a^t y_i > 0$ 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
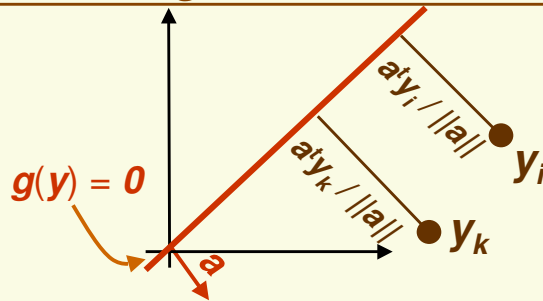
- Since we want $a^t y_i = b_i$, we expect sample y_i to be at distance b_i from the separating hyperplane (normalized by $\|a\|$)
- Thus $b_1, b_2, \dots, b_n$ give relative expected distances or "margins" of samples from the hyperplane
- Should make b_i small if sample i is expected to be near separating hyperplane, and make b_i larger otherwise
- In the absence of any additional information, there are good reasons to set $b_1 = b_2 = \dots = b_n = 1$

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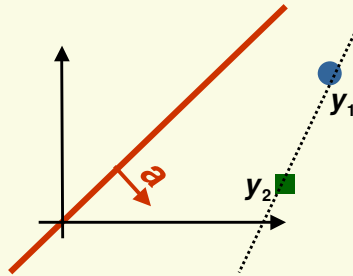
LDF: Approximate Solution

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

$$Y \; a = b$$

- 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 $Ya \approx b$
  - 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 **a** which minimizes the length of the error vector **e**

$$e = Ya - b$$
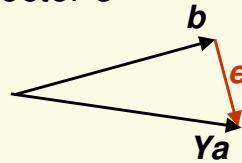
- Thus minimize the *minimum squared error* criterion function:

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

- Unlike the perceptron criterion function, we can optimize the minimum squared error criterion function analytically by setting the gradient to **0**

LDF: Optimizing $J_s(a)$

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

- Let's compute the gradient:

$$\nabla J_s(a) = \begin{bmatrix} \dfrac{\partial J_s}{\partial a_0} \\ \vdots \\ \dfrac{\partial J_s}{\partial a_d} \end{bmatrix} = \frac{dJ_s}{da} = \sum_{i=1}^{n} \frac{d}{da}(a^t y_i - b_i)^2$$

$$= \sum_{i=1}^{n} 2(a^t y_i - b_i)\frac{d}{da}(a^t y_i - b_i)$$

$$= \sum_{i=1}^{n} 2(a^t y_i - b_i)y_i$$

$$= 2Y^t(Ya - b)$$

LDF: Pseudo Inverse Solution

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

- Setting the gradient to 0:

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

- Matrix Y^tY is square (it has $d+1$ rows and columns) and it is often non-singular

- If Y^tY is non-singular, its inverse exists and we can solve for a uniquely:

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

pseudo inverse of Y
$$\left((Y^tY)^{-1}Y^t\right)Y = (Y^tY)^{-1}(Y^tY) = I$$

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 $y_1, \ldots, y_n$

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

$$1_n = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \Big\} n$$
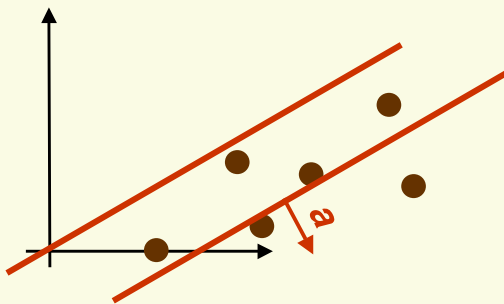
- Then we shift this line to the origin, if this line was a good fit, all samples will be classified correctly

LDF: Minimum Squared-Error Procedures

- Only guaranteed the separating hyperplane if $Ya > 0$

  - that is if all elements of vector $Ya = \begin{bmatrix} a^t y_1 \\ \vdots \\ a^t y_n \end{bmatrix}$ are positive

- We have $Ya \approx b$

- That is $Ya = \begin{bmatrix} b_1 + \varepsilon_1 \\ \vdots \\ b_n + \varepsilon_n \end{bmatrix}$ where ε may be negative

  - If $\varepsilon_1,\ldots, \varepsilon_n$ are small relative to $b_1,\ldots, b_n$, then each element of Ya is positive, and a gives a separating hyperplane
  - If approximation is not good, ε_i may be large and negative, for some i, thus $b_i + \varepsilon_i$ will be negative and a is not a separating hyperplane

- Thus in linearly separable case, least squares solution a does *not necessarily* give separating hyperplane
- But it will give a "reasonable" hyperplane

LDF: Minimum Squared-Error Procedures

- We are free to choose b. May be tempted to make b large as a way to insure $Ya \approx b > 0$

- Does not work
  - Let β be a scalar, let's try βb instead of b
  - if a^* is a least squares solution to $Ya = b$, then for any scalar β, least squares solution to $Ya = \beta b$ is βa^*

  $$\arg\min_a \|Ya - \beta b\|^2 = \arg\min_a \beta^2 \|Y(a/\beta) - b\|^2$$
  $$= \arg\min_a \|Y(a/\beta) - b\|^2 = \beta a^*$$

  - thus if for some ith element of Ya is less than 0, that is $y^t_i a < 0$, then $y^t_i(\beta a) < 0$,

- Relative difference between components of b matters, but not the size of each individual component

LDF: How to choose b in MSE Procedure?

- So far we assumed that constants $b_1, b_2, \ldots, b_n$ are positive but otherwise arbitrary
- Good choice is $b_1 = b_2 = \ldots = b_n = 1$. In this case,

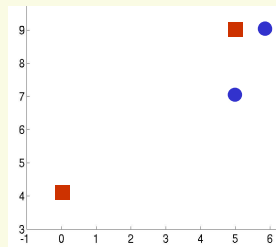1. MSE solution is basically identical to Fischer's linear discriminant solution
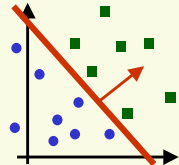
2. MSE solution approaches the Bayes discriminant function as the number of samples goes to infinity

$$g_B(x) = P(c_1 \mid x) - P(c_2 \mid x)$$

LDF: Example

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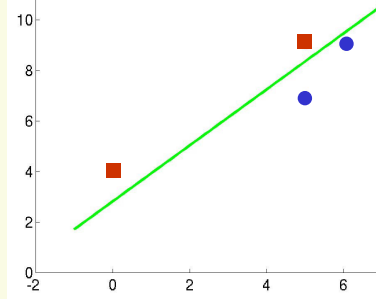
- Set vectors y_1, y_2, y_3, y_4 by adding extra feature and "normalizing"

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

- Matrix Y is then

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

LDF: Example

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

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

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

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

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

- This solution does give a separating hyperplane since $Ya > 0$

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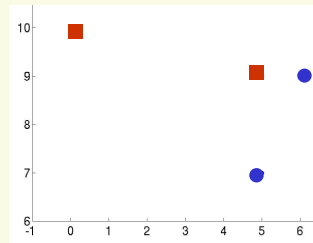
$$y_1 = \begin{bmatrix} 1 \\ 6 \\ 9 \end{bmatrix} \quad y_2 = \begin{bmatrix} 1 \\ 5 \\ 7 \end{bmatrix} \quad y_3 = \begin{bmatrix} -1 \\ -5 \\ -9 \end{bmatrix} \quad y_4 = \begin{bmatrix} -1 \\ 0 \\ -10 \end{bmatrix}$$

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

LDF: Example

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

- In matlab, $a = Y \backslash b$ solves the least squares problem
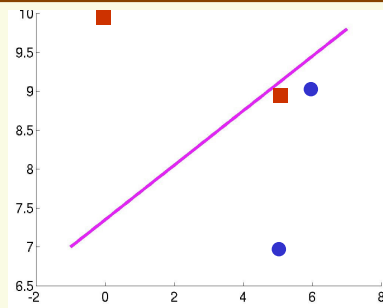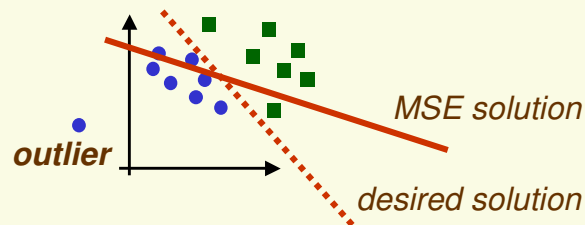$$a = \begin{bmatrix} 3.2 \\ 0.2 \\ -0.4 \end{bmatrix}$$

- Note a is an approximation to $Ya = b$, since no exact solution exists
$$Ya = \begin{bmatrix} 0.2 \\ 0.9 \\ -0.04 \\ 1.16 \end{bmatrix} \neq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

- This solution does not give a separating hyperplane since $a^t y_3 < 0$

LDF: Example

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

outlier

MSE solution

desired solution

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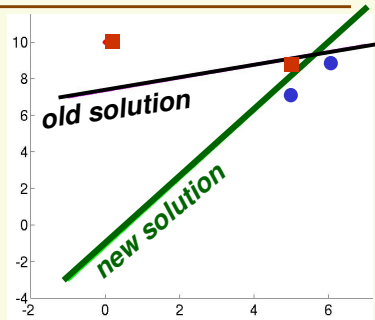
LDF: Example

- we know that 4th point is far far from separating hyperplane
  - In practice we don't know this

- Thus appropriate $b = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 10 \end{bmatrix}$

- In Matlab, solve $a = Y \backslash b$

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

- Note a is an approximation to $Ya = b$, $Ya = \begin{bmatrix} 0.9 \\ 1.0 \\ 0.8 \\ 10.0 \end{bmatrix} \neq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 10 \end{bmatrix}$

- This solution does give the separating hyperplane since $Ya > 0$

LDF: Gradient Descent for MSE solution

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

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

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

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

LDF: Widrow-Hoff Procedure

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

- Thus the update rule for gradient descent:
$$a^{(k+1)} = a^{(k)} - \eta^{(k)}Y^t(Ya^{(k)} - b)$$

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

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

$$a^{(k+1)} = a^{(k)} - \eta^{(k)}y_i(y_i^t a^{(k)} - b_i)$$

LDF: Ho-Kashyap Procedure

- In the MSE procedure, if b is chosen arbitrarily, finding separating hyperplane is not guaranteed

- Suppose training samples are linearly separable. Then there is a^s and positive b^s s.t.
$$Ya^s = b^s > 0$$

- If we knew b^s could apply MSE procedure to find the separating hyperplane
- Idea: find both a^s and b^s
- Minimize the following criterion function, restricting to positive b:
$$J_{HK}(a, b) = \|Ya - b\|^2$$

- $J_{HK}(a^s, b^s) = 0$

LDF: Ho-Kashyap Procedure

$$J_{HK}(a, b) = \|Ya - b\|^2$$

- As usual, take partial derivatives w.r.t. a and b

$$\nabla_a J_{HK} = 2Y^t(Ya - b) = 0$$
$$\nabla_b J_{HK} = -2(Ya - b) = 0$$

- Use modified gradient descent procedure to find a minimum of $J_{HK}(a, b)$

- Alternate the two steps below until convergence:
  1) Fix b and minimize $J_{HK}(a, b)$ with respect to a
  2) Fix a and minimize $J_{HK}(a, b)$ with respect to b

LDF: Ho-Kashyap Procedure

$$\nabla_a J_{HK} = 2Y^t(Ya - b) = 0 \qquad \nabla_b J_{HK} = -2(Ya - b) = 0$$

- Alternate the two steps below until convergence:
  1) Fix b and minimize $J_{HK}(a, b)$ with respect to a
  2) Fix a and minimize $J_{HK}(a, b)$ with respect to b

- Step (1) can be performed with pseudoinverse
  - For fixed b minimum of $J_{HK}(a, b)$ with respect to a is found by solving
  $$2Y^t(Ya - b) = 0$$
  - Thus
  $$a = (Y^tY)^{-1}Y^tb$$

LDF: Ho-Kashyap Procedure

- Step 2: fix a and minimize $J_{HK}(a,b)$ with respect to b

- We can't use $b = Ya$ because b has to be positive

- Solution: use modified gradient descent

- Regular gradient descent rule:
$$b^{(k+1)} = b^{(k)} - \eta^{(k)} \nabla_b J\left(a^{(k)}, b^{(k)}\right)$$

- If any components of $\nabla_b J$ are positive, b will decrease and can possibly become negative

$$b^{(k+1)} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - 2 * \begin{bmatrix} 2 \\ -3 \\ -2 \end{bmatrix} = \begin{bmatrix} -3 \\ 7 \\ 5 \end{bmatrix}$$

LDF: Ho-Kashyap Procedure

- start with positive b, follow negative gradient but refuse to decrease any components of b

- This can be achieved by setting all the positive components of $\nabla_b J$ to 0
$$b^{(k+1)} = b^{(k)} - \eta \frac{1}{2}\left[\nabla_b J\left(a^{(k)}, b^{(k)}\right) - |\nabla_b J\left(a^{(k)}, b^{(k)}\right)|\right]$$

  - here $|v|$ denotes vector we get after applying absolute value to all elements of v
$$b^{(k+1)} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - 2 * \frac{1}{2}\left[\begin{bmatrix} 2 \\ -3 \\ -2 \end{bmatrix} - \begin{bmatrix} 2 \\ 3 \\ 2 \end{bmatrix}\right] = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ -6 \\ -4 \end{bmatrix} = \begin{bmatrix} 1 \\ 7 \\ 5 \end{bmatrix}$$

- Not doing steepest descent anymore, but we are still doing descent **and** ensure that b is positive

LDF: Ho-Kashyap Procedure

$$b^{(k+1)} = b^{(k)} - \eta \frac{1}{2}\left[\nabla_b J\left(a^{(k)}, b^{(k)}\right) - / \nabla_b J\left(a^{(k)}, b^{(k)}\right) /\right]$$

$$\nabla_b J = -2(Ya - b) = 0$$

- Let $\quad e^{(k)} = Ya^{(k)} - b^{(k)} = -\frac{1}{2}\nabla J_b\left(a^{(k)}, b^{(k)}\right)$

- Then
$$b^{(k+1)} = b^{(k)} - \eta \frac{1}{2}\left[-2e^{(k)} - |\, 2e^{(k)}\,|\right]$$

$$= b^{(k)} + \eta\left[e^{(k)} + |\, e^{(k)}\,|\right]$$

LDF: Ho-Kashyap Procedure

- The final Ho-Kashyap procedure:

  0) Start with arbitrary $a^{(1)}$ and $b^{(1)} > 0$, let k = 1

  repeat steps (1) through (4)

      1) $\ e^{(k)} = Ya^{(k)} - b^{(k)}$

      2) Solve for $b^{(k+1)}$ using $a^{(k)}$ and $b^{(k)}$
  $$b^{(k+1)} = b^{(k)} + \eta\left[e^{(k)} + |\, e^{(k)}\,|\right]$$

      3) Solve for $a^{(k+1)}$ using $b^{(k+1)}$
  $$a^{(k+1)} = \left(Y^t Y\right)^{-1} Y^t\, b^{(k+1)}$$

      4) k = k + 1

  until $e^{(k)} >= 0$ or $k > k_{max}$ or $b^{(k+1)} = b^{(k)}$

- For convergence, learning rate should be fixed between $0 < \eta < 1$

LDF: Ho-Kashyap Procedure

$$b^{(k+1)} = b^{(k)} + \eta \left[e^{(k)} + | e^{(k)} | \right]$$

- What if $e^{(k)}$ is negative for all components?
  - $b^{(k+1)} = b^{(k)}$ and corrections stop

- Write $e^{(k)}$ out:
  $$e^{(k)} = Ya^{(k)} - b^{(k)} = Y \left(Y^t Y \right)^{-1} Y^t b^{(k)} - b^{(k)}$$

- Multiply by Y^t:
  $$Y^t e^{(k)} = Y^t \left(Y \left(Y^t Y \right)^{-1} Y^t b^{(k)} - b^{(k)} \right) = Y^t b^{(k)} - Y^t b^{(k)} = 0$$

- Thus $Y^t e^{(k)} = 0$

LDF: Ho-Kashyap Procedure

- Thus $Y^t e^{(k)} = 0$

- Suppose training samples are linearly separable. Then there is a^s and positive b^s s.t.
  $$Ya^s = b^s > 0$$

- Multiply both sides by $(e^{(k)})^t$
  $$0 = \left(e^{(k)} \right)^t Ya^s = \left(e^{(k)} \right)^t b^s$$

- Either by $e^{(k)} = 0$ or one of its components is positive

LDF: Ho-Kashyap Procedure

- In the linearly separable case,
  - $e^{(k)} = 0$, found solution, stop
  - one of components of $e^{(k)}$ is positive, algorithm continues

- In non separable case,
  - $e^{(k)}$ will have only negative components eventually, thus found proof of nonseparability
  - No bound on how many iteration need for the proof of nonseparability

LDF: Ho-Kashyap Procedure Example

- Class 1: (6 9), (5 7)
- Class 1: (5 9), (0 10)

- Matrix $Y = \begin{bmatrix} 1 & 6 & 9 \\ 1 & 5 & 7 \\ -1 & -5 & -9 \\ -1 & 0 & -10 \end{bmatrix}$
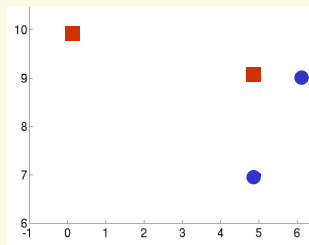
- Start with $a^{(1)} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$ and $b^{(1)} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

- Use fixed learning $\eta = 0.9$

- At the start $Ya^{(1)} = \begin{bmatrix} 16 \\ 13 \\ -15 \\ -11 \end{bmatrix}$

LDF: Ho-Kashyap Procedure Example

- Iteration 1:

  - $e^{(1)} = Ya^{(1)} - b^{(1)} = \begin{bmatrix} 16 \\ 13 \\ -15 \\ -11 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 15 \\ 12 \\ -16 \\ -12 \end{bmatrix}$

  - solve for $b^{(2)}$ using $a^{(1)}$ and $b^{(1)}$

 $b^{(2)} = b^{(1)} + 0.9\left[e^{(1)} + /e^{(1)}/\right] = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} + 0.9\left[\begin{bmatrix} 15 \\ 12 \\ -16 \\ -12 \end{bmatrix} + \begin{bmatrix} 15 \\ 12 \\ 16 \\ 12 \end{bmatrix}\right] = \begin{bmatrix} 28 \\ 22.6 \\ 1 \\ 1 \end{bmatrix}$

  - solve for $a^{(2)}$ using $b^{(2)}$

  $a^{(2)} = \left(Y^t Y\right)^{-1} Y^t b^{(2)} = \begin{bmatrix} -2.6 & 4.7 & 1.6 & -0.5 \\ 0.16 & -0.1 & -0.1 & 0.2 \\ 0.26 & -0.5 & -0.2 & -0.1 \end{bmatrix} * \begin{bmatrix} 28 \\ 22.6 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 34.6 \\ 2.7 \\ -3.8 \end{bmatrix}$

LDF: Ho-Kashyap Procedure Example

- Continue iterations until $Ya > 0$
  - In practice, continue until minimum component of Ya is less then 0.01
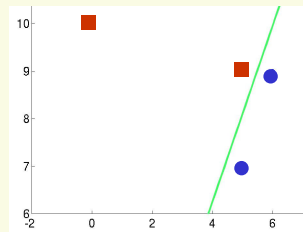
- After 104 iterations converged to solution

  $a = \begin{bmatrix} -34.9 \\ 27.3 \\ -11.3 \end{bmatrix} \qquad b = \begin{bmatrix} 28 \\ 23 \\ 1 \\ 147 \end{bmatrix}$

- a does gives a separating hyperplane

  $Ya = \begin{bmatrix} 27.2 \\ 22.5 \\ 0.14 \\ 1.48 \end{bmatrix}$

LDF: MSE for Multiple Classes

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

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

- 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: Many Classes
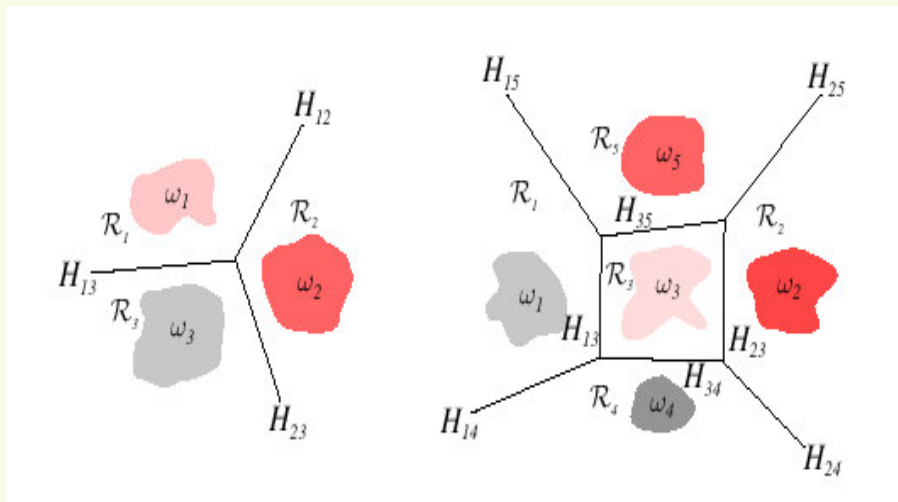
LDF: MSE for Multiple Classes

- We still use augmented feature vectors $y_1,\ldots,y_n$
- Define m linear discriminant functions

$$g_i(y) = a_i^t y \qquad i = 1,\ldots,m$$

- Given y, assign class c_i if

$$a_i^t y \geq a_j^t y \qquad \forall j \neq i$$

- For each class i, makes sense to seek weight vector a_i, s.t.

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

- If we find such $a_1,\ldots,a_m$ the training error will be 0

LDF: MSE for Multiple Classes

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

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

- We can solve for each a_i independently

- Let n_i be the number of samples in class i

- 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:

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

LDF: MSE for Multiple Classes

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

$$b_i = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} \text{\textbf{\textit{rows corresponding to samples from class i}}}$$

- We need to solve: $Ya_i = b_i$

$$\begin{bmatrix} \text{sample from class 1} \\ \text{sample from class 1} \\ \vdots \\ \text{sample from class m} \\ \text{sample from class m} \end{bmatrix} \begin{bmatrix} \text{weights } a_i \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix}$$

LDF: MSE for Multiple Classes

- We need to solve $Ya_i = b_i$

- Usually no exact solution since Y is overdetermined

- Use least squares to minimize norm of the error vector $\| Ya_i - b_i \|$

- LSE solution with pseudoinverse:
$$a_i = \left(Y^t Y\right)^{-1} Y^t b_i$$

- Thus we need to solve m LSE problems, one for each class
- Can write these m LSE problems in one matrix

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 $YA = B$:

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

$$\quad Y \qquad\qquad A \qquad\quad B$$

LDF: MSE for Multiple Classes

- Our objective function is:

$$J(A) = \sum_{i=1}^{m} \left\| Y a_i - b_i \right\|^2$$

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

$$A = (Y^t Y)^{-1} Y B$$

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 Y^tY is not singular and not too large
    - use gradient descent (Widrow-Hoff procedure) otherwise

- **Ho-Kashyap** procedures
    - always converge
    - find separating hyperplane in the linearly separable case
    - more costly