CMP-6002B - Machine Learning

Dr Gavin Cawley

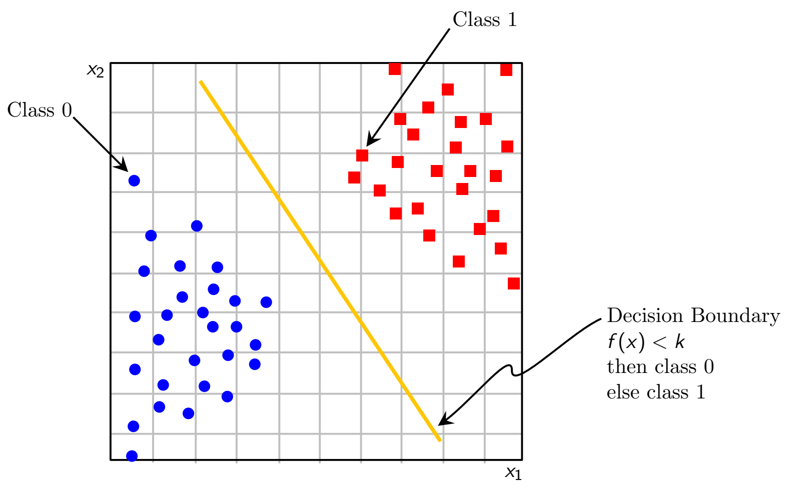
Lecture 4 - Linear Classifiers

Introduction

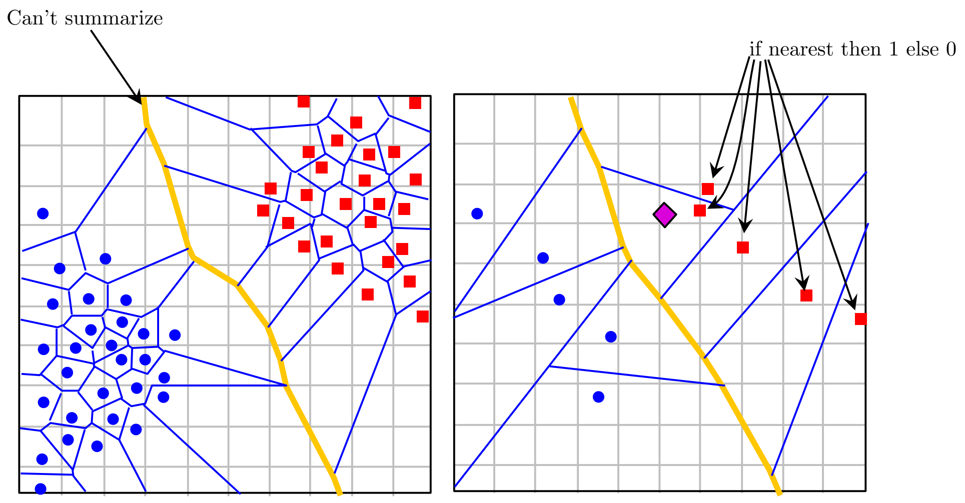
- Objective:
  - To understand the principle and practice of linear classifiers.
- Desired Outcomes:
  - Understand the principles of linear discriminant analysis
  - Understand the principles and practice of the perceptron approach
- Overview of Linear Classifiers
  - Optimal classifiers for Gaussian data
  - Fisher's linear discriminant
  - Linear classifiers as regression
  - Linear classifiers as Perceptrons
- Strengths and weaknesses
- Extensions to non-linear classifiers

Decision Boundaries

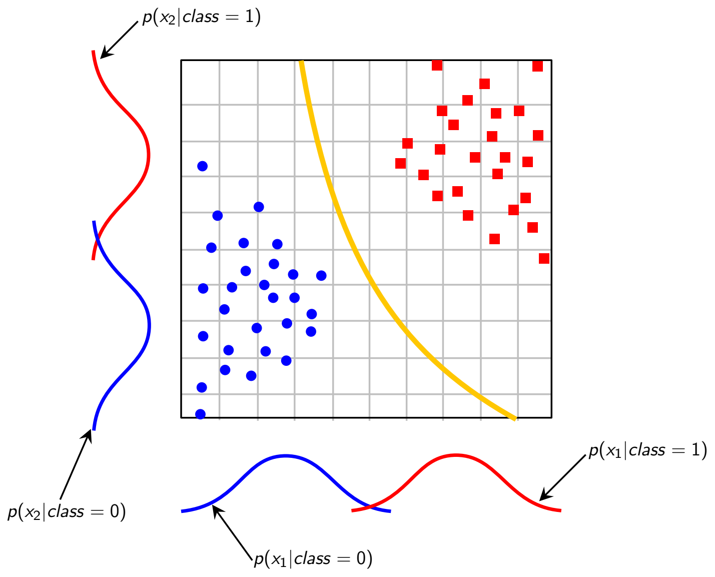
- The general problem of classification is to most effectively separate the classes.



1-Nearest Neighbour Classifier



Naïve Bayes Classifier

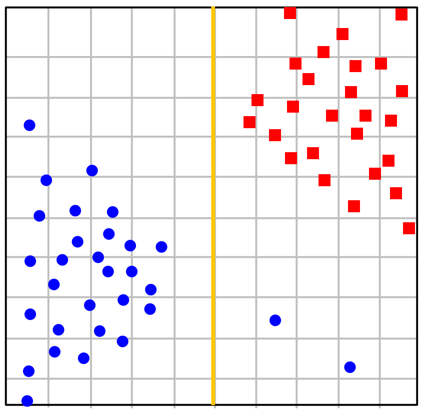


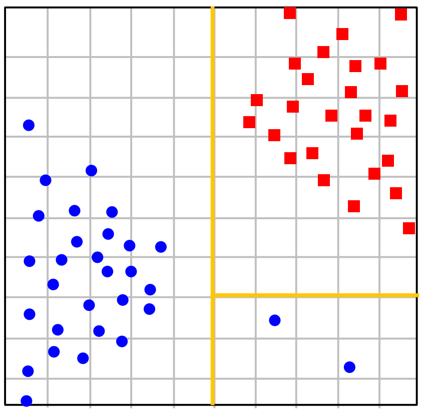
$$p(class = 0|x_1, x_2) = \frac{p(x_1|class = 0) \times p(x_2|class = 0) \times p(class = 0)}{p(x_1) \times p(x_2)}$$

Decision Tree Classifer

```
if ($x_1 < 5$) then
   Circle
else
   Square
```

```
if ($x_1 < 5$) then
   Circle
else
   if ($x_2 > 3$) then
      Square
   else
      Circle
```



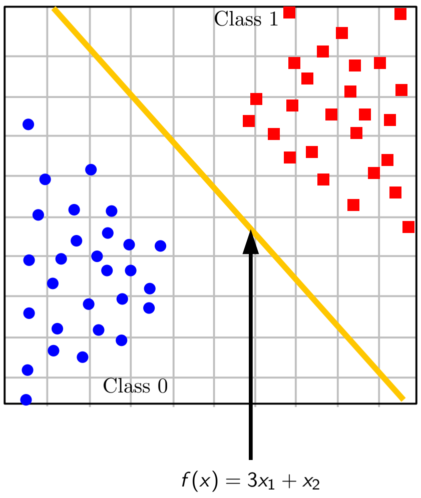


Linear Classifiers

- A linear model takes the form

$$g(\boldsymbol{x}) = f(\boldsymbol{w} \cdot \boldsymbol{x}) = f\left(\sum_{i=1}^{d} w_i x_i\right)$$

where $\boldsymbol{w}$ are the model parameters



Discriminant Functions

- A classifier can be represented using *discriminant* functions

$$g_1(\boldsymbol{x}),\ g_2(\boldsymbol{x}), \dots,\ g_c(\boldsymbol{x}).$$

- Assign to the class with the most positive discriminant,

$$g_i(\boldsymbol{x}) > g_j(\boldsymbol{x}) \quad \forall\ i \neq j$$

- For a probabilistic classifier, the error rate is minimised if

$$g_i(\boldsymbol{x}) = p(\mathcal{C}_i|\boldsymbol{x}) = p(\boldsymbol{x}_i|\mathcal{C}_i)p(\mathcal{C}_i)/p(\boldsymbol{x})$$

- Not affected by monotonic transformations or constants, so

$$g_i(\boldsymbol{x}) = \log p(\boldsymbol{x}|\mathcal{C}_i) + \log p(\mathcal{C}_i)$$

- Single discriminant for 2-class problems,

$$g(\boldsymbol{x}) = g_1(\boldsymbol{x}) - g_2(\boldsymbol{x}).$$

The Normal or Gaussian Density

- Linear classifiers can be motivated using Gaussian densities.
- Gaussian densities often used for mathematical tractability.
- Univariate Gaussian; parameters: mean μ and variance σ^2

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}$$

- Multivariate Gaussian,

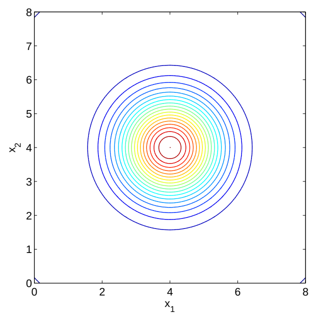
$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right\}$$

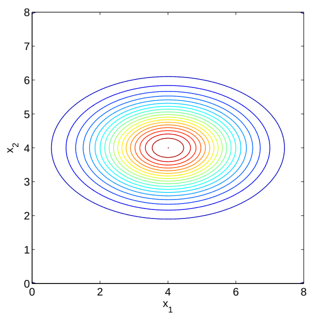
Parameters:

$\boldsymbol{\mu}$	-	Mean vector
$\boldsymbol{\Sigma}$	-	Covariance matrix

Multivariate Gaussian Densities

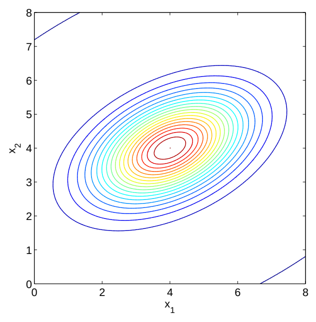
$$\boldsymbol{\mu} = [4 \;\; 4] \qquad \boldsymbol{\Sigma} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$





$$\boldsymbol{\mu} = [4 \;\; 4] \qquad \boldsymbol{\Sigma} = \begin{bmatrix} 2 & 0 \\ 0 & 0.75 \end{bmatrix}$$

$$\boldsymbol{\mu} = [4 \;\; 4] \qquad \boldsymbol{\Sigma} = \begin{bmatrix} 2 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$



Identical Spherical Gaussian Densities

- Assume the class conditional densities are spherical Gaussians with equal variances

$$p(\boldsymbol{x}|\mathcal{C}_1) \sim \mathcal{N}(\boldsymbol{\mu}_1, \sigma^2 \boldsymbol{I}) \quad \text{and} \quad p(\boldsymbol{x}|\mathcal{C}_2) \sim \mathcal{N}(\boldsymbol{\mu}_2, \sigma^2 \boldsymbol{I})$$

- Then the optimal discriminant functions are given by

$$g_i(\boldsymbol{x}) = -\|\boldsymbol{x} - \boldsymbol{\mu}_i\|^2/(2\sigma^2) + \log p(\mathcal{C}_i)$$

- Noting that $\|\boldsymbol{x} - \boldsymbol{\mu}_i\|^2 = (\boldsymbol{x} - \boldsymbol{\mu}_i)^T(\boldsymbol{x} - \boldsymbol{\mu}_i)$,

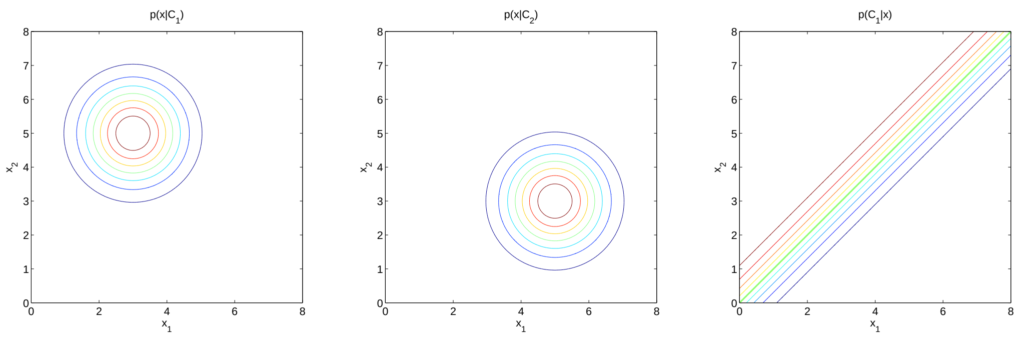
$$g_i(\boldsymbol{x}) = -\left[\boldsymbol{x}^T\boldsymbol{x} - 2\boldsymbol{\mu}_i^T\boldsymbol{x} + \boldsymbol{\mu}_i^T\boldsymbol{\mu}_i\right]/(2\sigma^2) + \log p(\mathcal{C}_i)$$

- So a linear discriminant is optimal, $g_i(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b_i$, where

$$\boldsymbol{w}_i = \boldsymbol{\mu_i}/\sigma^2 \quad \text{and} \quad b_i = -\boldsymbol{\mu}_i^T\boldsymbol{\mu}_i/(2\sigma^2) + \log p(\mathcal{C}_i)$$

Example: Identical Spherical Gaussian Densities

- For this example: $\boldsymbol{\mu}_1 = [3 \;\; 5]$, $\boldsymbol{\mu}_2 = [5 \;\; 3]$ and $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{I}$.
- Equal prior probabilities $p(\mathcal{C}_1) = p(\mathcal{C}_2) = 0.5$.



- The contours of the posterior $p(\mathcal{C}_1|\boldsymbol{x})$ are linear
- The optimal discriminant function is also linear
  - $g(\boldsymbol{x}) = 0$ corresponds to $p(\mathcal{C}_1|\boldsymbol{x}) = 0.5$.

Example: Identical Spherical Gaussian Densities

- Step 1: construct $g_1(\boldsymbol{x}) = \boldsymbol{w}_1^T \boldsymbol{x} + b_1$.

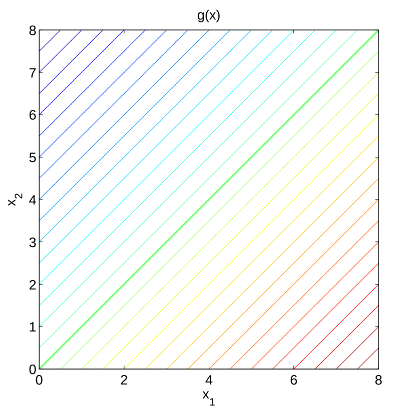
$$\boldsymbol{w}_1 = \boldsymbol{\mu}_i/\sigma^2 = [3 \;\; 5], \qquad b_1 = -\boldsymbol{\mu}_1^T \boldsymbol{\mu}_1/(2\sigma^2) \approx -15.6931$$

- Step 2: construct $g_2(\boldsymbol{x}) = \boldsymbol{w}_2^T \boldsymbol{x} + b_2$.

$$\boldsymbol{w}_2 = \boldsymbol{\mu}_2/\sigma^2 = [5 \;\; 3], \qquad b_2 = -\boldsymbol{\mu}_2^T \boldsymbol{\mu}_2/(2\sigma^2) \approx -15.6931$$

- Step 3: construct single discriminant

$$\begin{aligned} g(x) &= g_1(x) - g_2(x) \\ &= [5 \;\; 3]^T \boldsymbol{x} - [3 \;\; 5]^T \boldsymbol{x} \\ &= 2x_1 - 2x_2 \end{aligned}$$



- c.f. $p(\mathcal{C}_1|\boldsymbol{x}) = 0.5$.

Identical Arbitrary Gaussians

- Suppose the densities have different means, but common covariance $\boldsymbol{\Sigma}_i = \boldsymbol{\Sigma}$.
- Again we write the discriminant function,

$$g_1(\boldsymbol{x}) = -(\boldsymbol{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_1)/2 + \log p(\mathcal{C}_1).$$

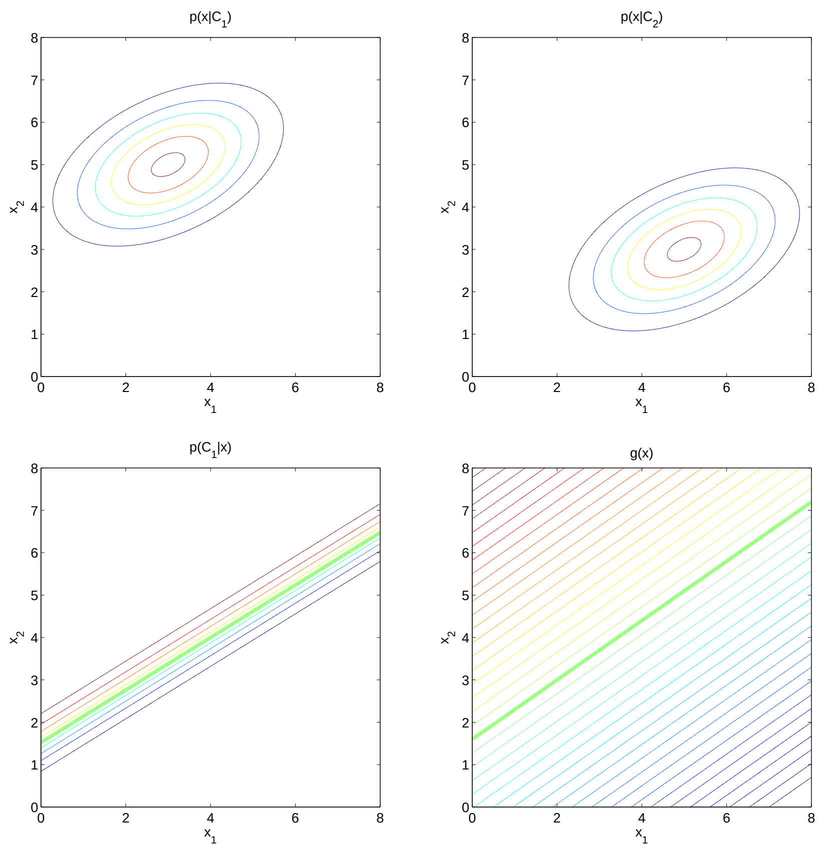
- Multiplying out, we obtain a common term $\boldsymbol{x}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{x}$.
- Again a linear discriminant is optimal,

$$g_1(\boldsymbol{x}) = \boldsymbol{w}_1^T \boldsymbol{x} + b_1,$$

where

$$\boldsymbol{w}_1 = \boldsymbol{\Sigma}^{-1} \mu_1 \quad \text{and} \quad b_1 = -\mu_1 \boldsymbol{\Sigma}^{-1} \mu_1/2 + \log p(\mathcal{C}_1).$$

Example: Identical Arbitrary Gaussian Densities



Arbitrary Gaussian Densities

- Consider the case where $\mathbf{\Sigma}_1$ and $\mathbf{\Sigma}_2$ are arbitrary covariance matrices.
- In this case we have a quadratic classifier

$$g_i(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{W}_i \boldsymbol{x} + \boldsymbol{w}_i^T \boldsymbol{w}_i + b_i$$

where

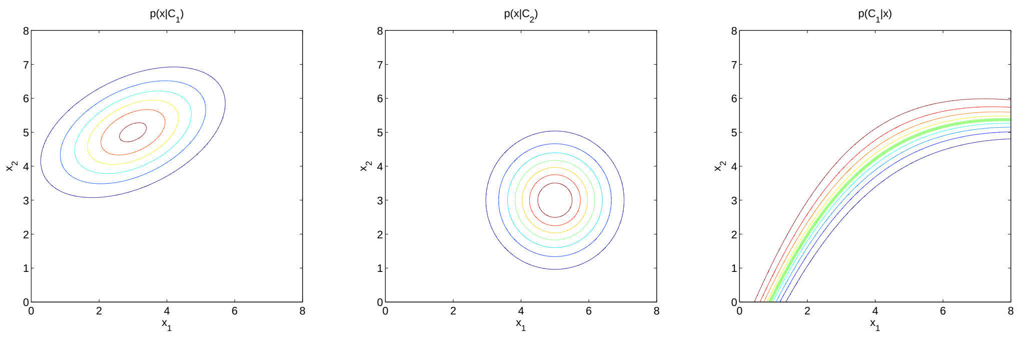
$$\boldsymbol{W}_i = -\frac{1}{2}\mathbf{\Sigma}_i^{-1}, \qquad \boldsymbol{w}_i = \mathbf{\Sigma}_i^{-1}\boldsymbol{\mu}_i,$$

and

$$b_i = -\frac{1}{2}\boldsymbol{\mu}_i \mathbf{\Sigma}_i^{-1}\boldsymbol{\mu}_i - \frac{1}{2}\log|\mathbf{\Sigma}_i| + \log(\mathcal{C}_i).$$

- The optimal discriminant will be a hyper-quadratic.

Example: Arbitrary Gaussian Densities



- In two dimensions, a quadratic discriminant is of the form

$$g(\boldsymbol{x}) = \omega_1 x_1^2 + \omega_2 x_2^2 + \omega_3 x_1 x_2 + \omega_4 x_1 + \omega_5 x_2 + b$$

- Can implement as a linear discriminant in an augmented attribute space.

Linear Discriminant Analysis

- Fisher's Linear Discriminant Analysis[1] (LDA) is *the* classic linear classifier.
- Basic idea: Find the linear projection of the data that maximises the distance between patterns of different classes while minimising the distance between patterns of the same class.
- Consider the projection onto a line joining the class means,

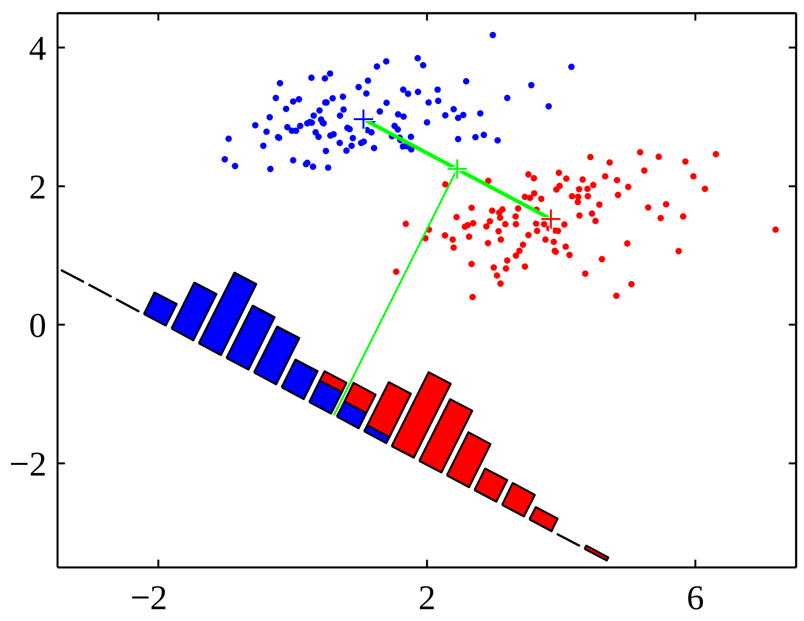
$$g(\boldsymbol{x}) = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T \boldsymbol{x}$$

where

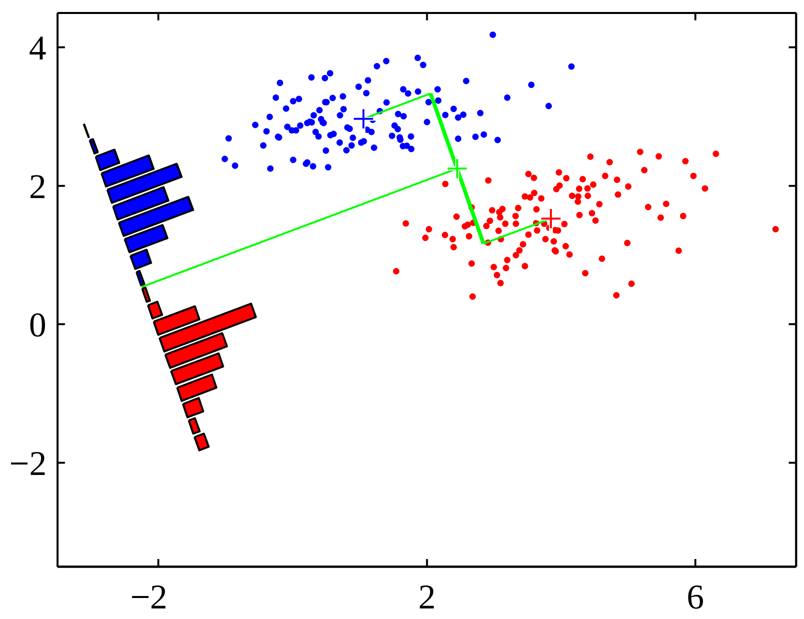
$$\boldsymbol{\mu}_1 = \frac{1}{N_1} \sum_{i \in \mathcal{C}_1} x_i \qquad \text{and} \qquad \boldsymbol{\mu}_2 = \frac{1}{N_2} \sum_{i \in \mathcal{C}_2} x_i$$

[1] Fisher, R.A. The Use of Multiple Measurements in Taxonomic Problems. Annals of Eugenics, 7: 179-188 (1936)

Projection onto Line Joining Class Means



A Better Linear Discriminant



Linear Discriminant Analysis

- The second discriminant takes more account of the *distribution* of positive and negative patterns.
- Linear discriminant $g(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x}$, but how to choose $\boldsymbol{w}$?
- Optimise the Fisher ratio

$$\mathcal{J}(\boldsymbol{w}) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2}$$

- Let $y_n = \boldsymbol{w}^T \boldsymbol{x}_n$, then

$$m_k = \frac{1}{N_k} \sum_{n \in \mathcal{C}_k} y_n \qquad \text{and} \qquad s_k^2 = \frac{1}{N_k} \sum_{n \in \mathcal{C}_k} (y_n - m_k)^2$$

- Distance between class means divided by average distance to class mean.

Fisher's Linear Discriminant Analysis

- We can rewrite the Fisher criterion as

$$\mathcal{J}(\boldsymbol{w}) = \frac{\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w}}$$

- $\boldsymbol{S}_B$ is the *between class scatter matrix*

$$\boldsymbol{S}_B = (\boldsymbol{m}_2 - \boldsymbol{m}_1)(\boldsymbol{m}_2 - \boldsymbol{m}_1)^T$$

- $\boldsymbol{S}_W$ is the *within class scatter matrix*

$$\boldsymbol{S}_W = \sum_{n \in \mathcal{C}_1} (\boldsymbol{x}_n - \boldsymbol{m}_1)(\boldsymbol{x}_n - \boldsymbol{m}_1)^T + \sum_{n \in \mathcal{C}_2} (\boldsymbol{x}_n - \boldsymbol{m}_2)(\boldsymbol{x}_n - \boldsymbol{m}_2)^T$$

- Differentiate Fisher criterion w.r.t. $\boldsymbol{w}$ and set to zero

$$(\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w}) \boldsymbol{S}_W \boldsymbol{w} = (\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w}) \boldsymbol{S}_B \boldsymbol{w} \quad \Longrightarrow \quad \boldsymbol{w} \propto \boldsymbol{S}_W^{-1} (\boldsymbol{m}_2 - \boldsymbol{m}_1)$$

Least Squares Linear Classifiers

- Given m attributes $x_1, x_2, \ldots, x_m$ and observed responses y, estimate parameters $w_0, w_1, w_2, \ldots, w_m$ for the line

$$g(\boldsymbol{x}) = w_0 + w_1 x_1 + \ldots w_m x_m$$

- Fit a line, for example
  - Line 1:

$$g_a(\boldsymbol{x}) = 0.2 + 0.3 \times x_1 - 0.2 \times x_2$$

  - Line 2:

$$g_b(\boldsymbol{x}) = -0.3 + 0.13 \times x_1 + 0.05 \times x_2$$

- How do we choose which is best?

x_1	x_2	y
2.7	5.5	0
0.9	4.7	0
1.1	3.1	0
2.9	1.9	0
0.5	1.0	0
8.0	9.1	1
5.4	8.5	1
6.1	6.6	1
8.3	6.6	1
8.1	4.7	1

Linear Classifiers via Multiple Regression

- Evaluate predicted or fitted values

y	0	0	0	0	0	1	1	1	1	1
$g_a(\boldsymbol{x})$	0.06	-0.08	0.78	1.92	1.65	-0.15	-0.63	0.53	1.19	2.08
$g_b(\boldsymbol{x})$	0.29	0.03	-0.02	0.16	-0.19	1.14	0.78	0.78	1.07	0.96

- From the fitted values we can form a decision boundary by classifying a case as the nearest fitted integer value

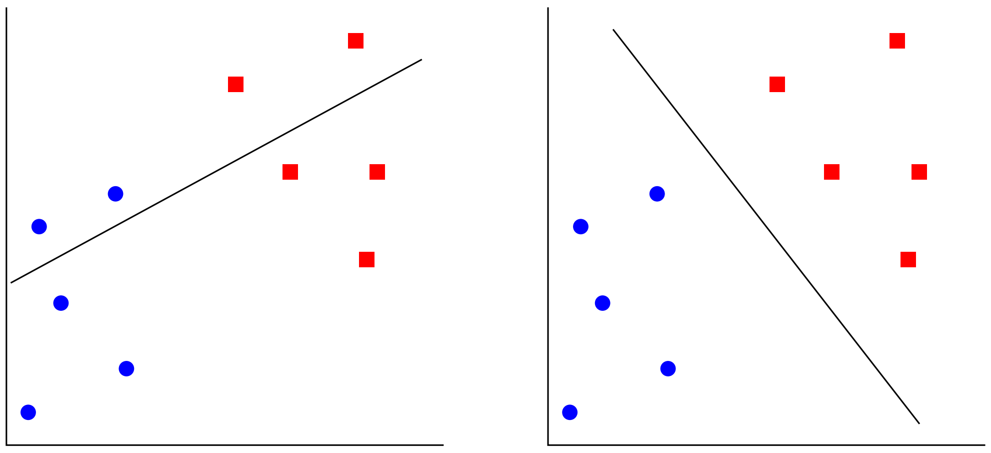
y	0	0	0	0	0	1	1	1	1	1
$g_a(\boldsymbol{x})$	0	0	1	1	1	0	0	1	1	1
$g_b(\boldsymbol{x})$	0	0	0	0	0	1	1	1	1	1

- $g_b(\boldsymbol{x})$ is the better discriminant.

Linear Classifiers via Multiple Regression

So, for any line, the decision boundary for a binary response variable is simply

$$w_0 + w_1 x_1 + \cdots + w_m x_m < 0.5$$



Linear Classifiers via Multiple Regression

- Which of the infinite possible lines should we choose?
- The obvious candidate is the one that *minimises the sum of the squared error*

$$SSE = \sum_{i=1}^{n} (y_i - g(\boldsymbol{x}_i))^2$$

- Compute SSE for each discriminant:

y	0	0	0	0	0	1	1	1	1	1	
$g_a(\boldsymbol{x})$	0.06	-0.08	0.78	1.92	1.65	-0.15	-0.63	0.53	1.19	2.08	
$g_b(\boldsymbol{x})$	0.29	0.03	-0.02	0.16	-0.19	1.14	0.78	0.78	1.07	0.96	
Error g_a	0.00	0.01	0.61	3.69	2.72	1.32	2.66	0.22	0.04	1.17	**12.44**
Error g_b	0.09	0.00	0.00	0.03	0.04	0.02	0.05	0.05	0.00	0.00	**0.28**

- Clearly $g_b(\boldsymbol{x})$ has a lower SSE.

Linear Classifiers via Multiple Regression

- There is always a single line that minimizes the SSE
- With a little matrix algebra, it is simple to derive the formula for this line.
- To fit model

$$\hat{y} = w_0 + w_1 x_1 + \cdots + w_m x_m$$

- We can re-write this in matrix form as $\hat{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{w}$, where

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad \boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_m \end{bmatrix} \quad \boldsymbol{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1m} \\ 1 & x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}$$

Least Squares Linear Classifier

- The model errors are defined by $\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{y}}$
- The sum of squared errors is given by

$$SSE = \sum (y_i - \hat{y}_i)^2 = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w})^2 = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w})$$

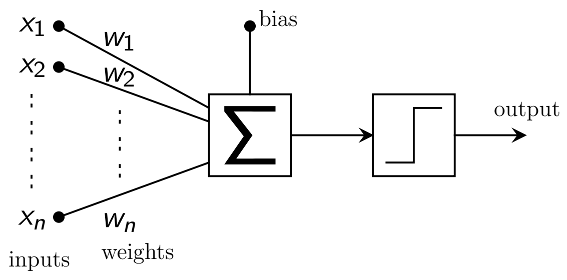
- Differentiate and set to zero

$$\begin{aligned} \frac{\partial(SSE)}{\partial \boldsymbol{w}} &= -2\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}) = 0 \\ \boldsymbol{X}^T\boldsymbol{y} - \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} &= 0 \\ \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} &= \boldsymbol{X}^T\boldsymbol{y} \\ \boldsymbol{w} &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \end{aligned}$$

  Assuming the inverse exists.
- Equivalent to Fisher's linear discriminant!

Linear Classifiers: Perceptron Approach

One of the first attempts to make an artificial neural network[2].



A *Perceptron* computes the sum of its weighted inputs and passes the result to a hard-limit threshold function.

Given input attributes $x_1, x_2, \ldots, x_m$ and response variable t that can take values -1 or +1, let

$$y = \psi\{w_0 + w_1 x_1 + \cdots + w_m x_m\} \quad \text{where} \quad \psi\{z\} = \begin{cases} +1 & \text{if } z \geq 0 \\ -1 & \text{if } z < 0 \end{cases}$$

[2]Rosenblatt, Frank (1958), The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain, Cornell Aeronautical Laboratory, Psychological Review, v65, No. 6, pp. 386-408.

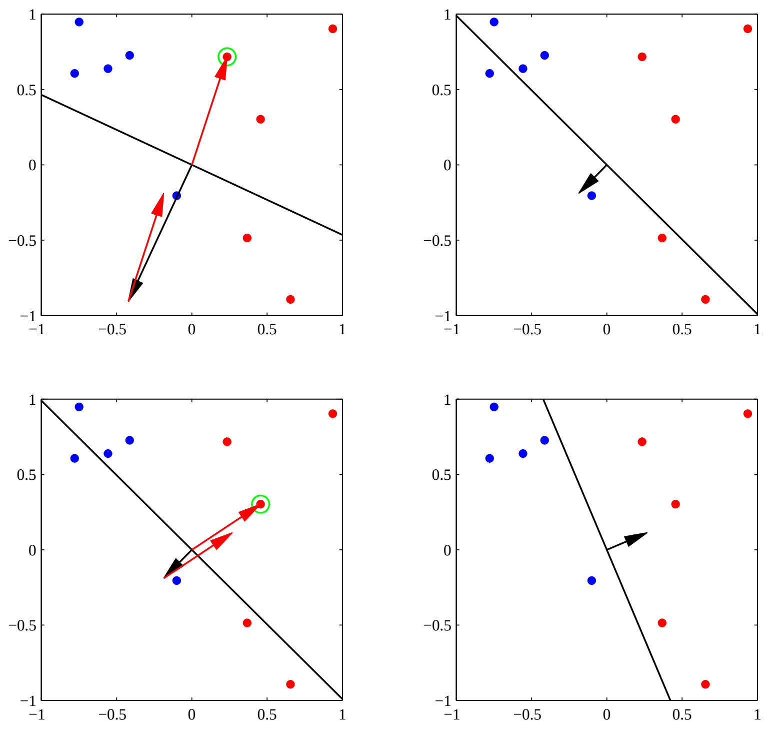
Perceptron Training Rule

```
**Algorithm PerceptronTraining(DataSet X, t)**
**Returns LinearModel $\boldsymbol{w}$**

initialise $\boldsymbol{w}$ to random values
initialise learning rate $\eta$
do
    for i=1 to n
        $y_i = \psi(\boldsymbol{w}, \boldsymbol{x}_i)$
        for j=1 to m
            $\Delta w_j \leftarrow 0.5\eta(t_i - y_i)x_{ij}$
            $w_j \leftarrow w_j + \Delta w_j$
while (Stopping($\boldsymbol{t}, \boldsymbol{y}$) == false)

return $\boldsymbol{w}$
```

The Perceptron Rule

- Basic idea, cycle through training patterns, if a pattern is currently misclassified add/subtract the input vector to the weights.
  - This shifts the discriminant to make it more likely to classify that pattern correctly next time.
- η is the learning rate which moderates how much the weights are changed on each iteration.
  - The discriminant is unaffected by the magnitude of $\boldsymbol{w}^T\boldsymbol{x}$, so let $\eta = 1$.
- Stopping conditions vary, simplest is to stop when there is no change in $\boldsymbol{y}$ or error is zero.
  - A complete pass through the dataset is made without modifying the weights.

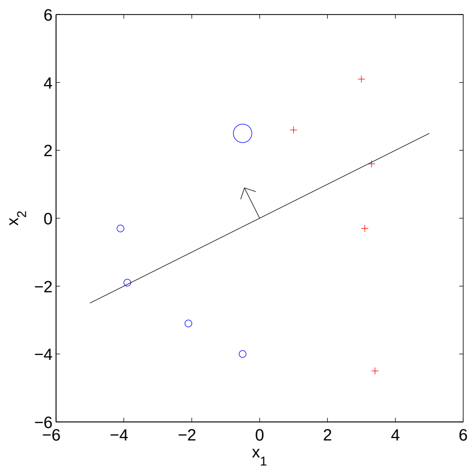
Example: The Percepton Rule



Perceptron Example - Step #1

x_1	x_2	t
-0.5	+2.5	-1
+3.0	+4.1	+1
-4.1	-0.3	-1
+3.4	-4.5	+1
-3.9	-1.9	-1
+1.0	+2.6	+1
-2.1	-3.1	-1
+3.3	+1.6	+1
-0.5	-4.0	-1
+3.1	-0.3	+1

$$\boldsymbol{w} = \begin{bmatrix} -1.0 \\ +2.0 \end{bmatrix}$$

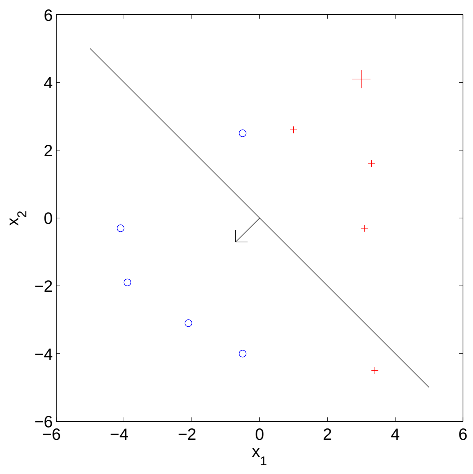


$$\begin{aligned} y_1 &= \psi(-1.0 \times -0.5 + 2.0 \times 2.5) = \psi(5.5) \quad \text{– WRONG!} \\ \boldsymbol{w} &\leftarrow \boldsymbol{w} + 0.5\eta(t_i - y_i)\boldsymbol{x}_i \\ &= [-1 \;\; 2] + 0.5 \times (-1 - +1)[-0.5 \;\; 2.5] = [-0.5 \quad -0.5] \end{aligned}$$

Perceptron Example - Step #2

x_1	x_2	t
-0.5	+2.5	-1
+3.0	+4.1	+1
-4.1	-0.3	-1
+3.4	-4.5	+1
-3.9	-1.9	-1
+1.0	+2.6	+1
-2.1	-3.1	-1
+3.3	+1.6	+1
-0.5	-4.0	-1
+3.1	-0.3	+1

$$\boldsymbol{w} = \begin{bmatrix} -0.5 \\ -0.5 \end{bmatrix}$$

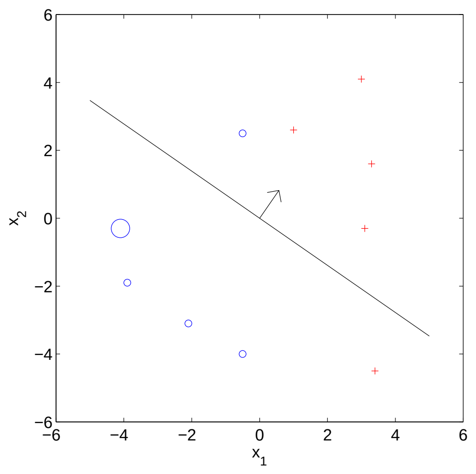


$$\begin{aligned} y_2 &= \psi(-0.5 \times 3.0 + -0.5 \times 4.1) = \psi(-3.55) \quad \text{– WRONG!} \\ \boldsymbol{w} &\leftarrow \boldsymbol{w} + 0.5\eta(t_i - y_i)\boldsymbol{x}_i \\ &= [-0.5 \quad -0.5] + 0.5 \times (+1 - -1)[3.0 \;\; 4.1] = [2.5 \;\; 3.6] \end{aligned}$$

Perceptron Example - Step #3

x_1	x_2	t
-0.5	+2.5	-1
+3.0	+4.1	+1
-4.1	-0.3	-1
+3.4	-4.5	+1
-3.9	-1.9	-1
+1.0	+2.6	+1
-2.1	-3.1	-1
+3.3	+1.6	+1
-0.5	-4.0	-1
+3.1	-0.3	+1

$$\boldsymbol{w} = \begin{bmatrix} +2.5 \\ +3.6 \end{bmatrix}$$



$$y_3 = \psi(2.5 \times -4.1 + 3.6 \times -0.3) = \psi(-11.330000) \quad \text{– RIGHT!}$$

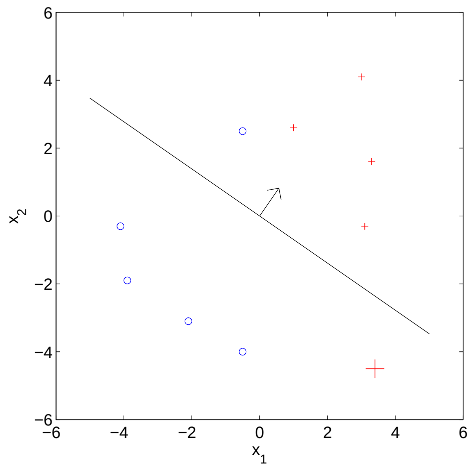
$$\boldsymbol{w} \leftarrow \boldsymbol{w} + 0.5\eta(t_i - y_i)\boldsymbol{x}_i$$

$$= [2.5 \quad 3.6] + 0.5 \times (-1 - -1)[-4.1 \quad -0.3] = [2.5 \quad 3.6]$$

Perceptron Example - Step #4

x_1	x_2	t
-0.5	+2.5	-1
+3.0	+4.1	+1
-4.1	-0.3	-1
+3.4	-4.5	+1
-3.9	-1.9	-1
+1.0	+2.6	+1
-2.1	-3.1	-1
+3.3	+1.6	+1
-0.5	-4.0	-1
+3.1	-0.3	+1

$$\boldsymbol{w} = \begin{bmatrix} +2.5 \\ +3.6 \end{bmatrix}$$



$$y_4 = \psi(2.5 \times 3.4 + 3.6 \times -4.5) = \psi(-7.7) \quad \text{– WRONG!}$$

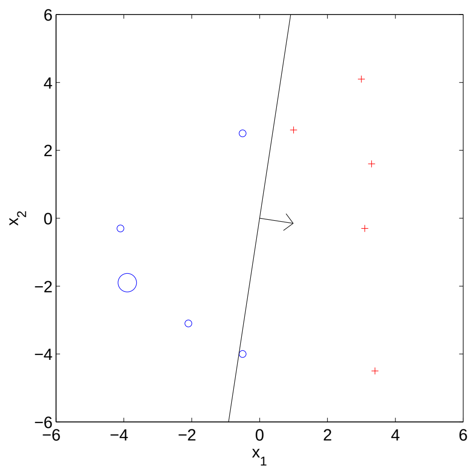
$$\boldsymbol{w} \leftarrow \boldsymbol{w} + 0.5\eta(t_i - y_i)\boldsymbol{x}_i$$

$$= [2.5 \quad 3.6] + 0.5 \times (+1 - -1)[3.4 \quad -4.5] = [5.9 \quad -0.9]$$

Perceptron Example - Step #5

x_1	x_2	t
-0.5	+2.5	-1
+3.0	+4.1	+1
-4.1	-0.3	-1
+3.4	-4.5	+1
-3.9	-1.9	-1
+1.0	+2.6	+1
-2.1	-3.1	-1
+3.3	+1.6	+1
-0.5	-4.0	-1
+3.1	-0.3	+1

$$\boldsymbol{w} = \begin{bmatrix} +5.9 \\ -0.9 \end{bmatrix}$$



$$y_5 = \psi(5.9 \times -3.9 + -0.9 \times -1.9) = \psi(-21.3) \quad \text{– RIGHT!}$$

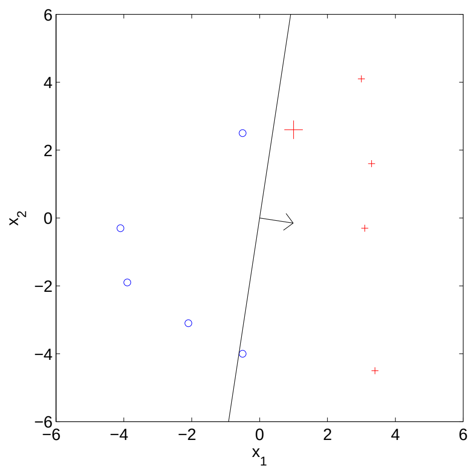
$$\boldsymbol{w} \leftarrow \boldsymbol{w} + 0.5\eta(t_i - y_i)\boldsymbol{x}_i$$

$$= [5.9 \quad -0.9] + 0.5 \times (-1 - -1)[-3.9 \quad -1.9] = [5.9 \quad -0.9]$$

Perceptron Example - Step #6

x_1	x_2	t
-0.5	+2.5	-1
+3.0	+4.1	+1
-4.1	-0.3	-1
+3.4	-4.5	+1
-3.9	-1.9	-1
+1.0	+2.6	+1
-2.1	-3.1	-1
+3.3	+1.6	+1
-0.5	-4.0	-1
+3.1	-0.3	+1

$$\boldsymbol{w} = \begin{bmatrix} +5.9 \\ -0.9 \end{bmatrix}$$



$$y_6 = \psi(5.9 \times 1.0 + -0.9 \times 2.6) = \psi(3.56) \quad \text{– RIGHT!}$$

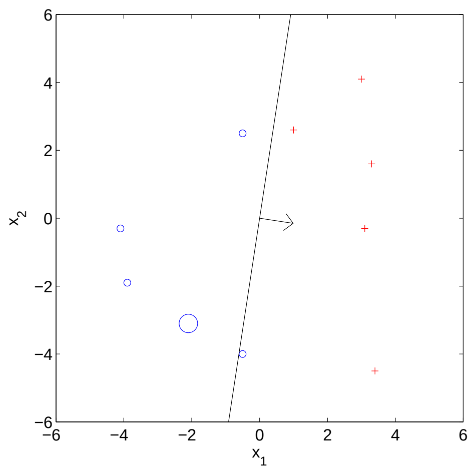
$$\boldsymbol{w} \leftarrow \boldsymbol{w} + 0.5\eta(t_i - y_i)\boldsymbol{x}_i$$

$$= [5.9 \quad -0.9] + 0.5 \times (+1 - +1)[1.0 \quad 2.6] = [5.9 \quad -0.9]$$

Perceptron Example - Step #7

x_1	x_2	t
-0.5	+2.5	-1
+3.0	+4.1	+1
-4.1	-0.3	-1
+3.4	-4.5	+1
-3.9	-1.9	-1
+1.0	+2.6	+1
-2.1	-3.1	-1
+3.3	+1.6	+1
-0.5	-4.0	-1
+3.1	-0.3	+1

$$\boldsymbol{w} = \begin{bmatrix} +5.9 \\ -0.9 \end{bmatrix}$$



$$y_7 = \psi(5.9 \times -2.1 + -0.9 \times -3.1) = \psi(-9.6) \quad \text{– RIGHT!}$$

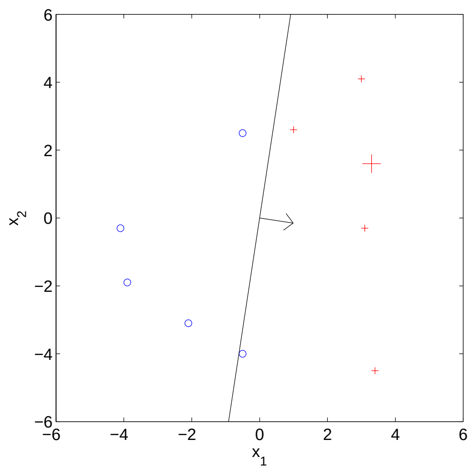
$$\boldsymbol{w} \leftarrow \boldsymbol{w} + 0.5\eta(t_i - y_i)\boldsymbol{x}_i$$

$$= [5.9 \quad -0.9] + 0.5 \times (-1 - -1)[-2.1 \quad -3.1] = [5.9 \quad -0.9]$$

Perceptron Example - Step #8

x_1	x_2	t
-0.5	+2.5	-1
+3.0	+4.1	+1
-4.1	-0.3	-1
+3.4	-4.5	+1
-3.9	-1.9	-1
+1.0	+2.6	+1
-2.1	-3.1	-1
+3.3	+1.6	+1
-0.5	-4.0	-1
+3.1	-0.3	+1

$$\boldsymbol{w} = \begin{bmatrix} +5.9 \\ -0.9 \end{bmatrix}$$



$$y_8 = \psi(5.9 \times 3.3 + -0.9 \times 1.6) = \psi(18.03) \quad \text{– RIGHT!}$$

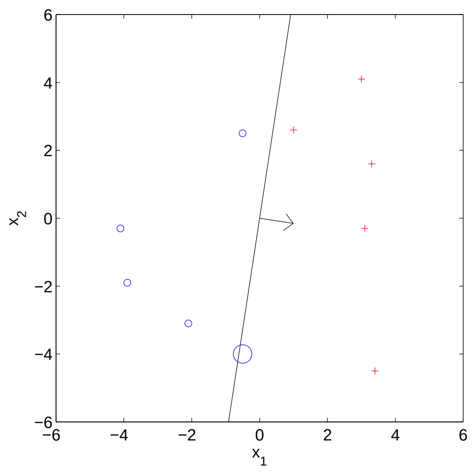
$$\boldsymbol{w} \leftarrow \boldsymbol{w} + 0.5\eta(t_i - y_i)\boldsymbol{x}_i$$

$$= [5.9 \quad -0.9] + 0.5 \times (+1 - +1)[3.3 \quad 1.6] = [5.9 \quad -0.9]$$

Perceptron Example - Step #9

x_1	x_2	t
-0.5	+2.5	-1
+3.0	+4.1	+1
-4.1	-0.3	-1
+3.4	-4.5	+1
-3.9	-1.9	-1
+1.0	+2.6	+1
-2.1	-3.1	-1
+3.3	+1.6	+1
-0.5	-4.0	-1
+3.1	-0.3	+1

$$\boldsymbol{w} = \begin{bmatrix} +5.9 \\ -0.9 \end{bmatrix}$$

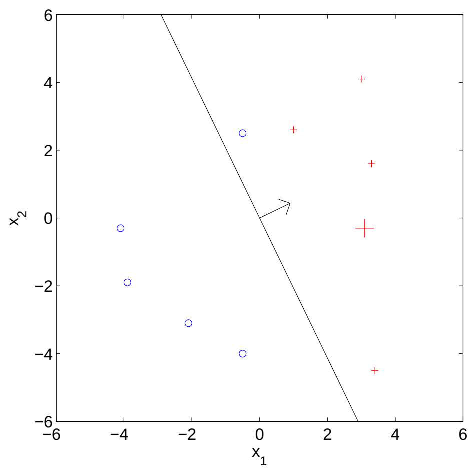


$$\begin{aligned} y_9 &= \psi(5.9 \times -0.5 + -0.9 \times -4.0) = \psi(0.65) \quad \text{– WRONG!} \\ \boldsymbol{w} &\leftarrow \boldsymbol{w} + 0.5\eta(t_i - y_i)\boldsymbol{x}_i \\ &= [5.9 \quad -0.9] + 0.5 \times (-1 - +1)[-0.5 \quad -4.0] = [6.4 \quad 3.1] \end{aligned}$$

Perceptron Example - Step #10

x_1	x_2	t
-0.5	+2.5	-1
+3.0	+4.1	+1
-4.1	-0.3	-1
+3.4	-4.5	+1
-3.9	-1.9	-1
+1.0	+2.6	+1
-2.1	-3.1	-1
+3.3	+1.6	+1
-0.5	-4.0	-1
+3.1	-0.3	+1

$$\boldsymbol{w} = \begin{bmatrix} +6.4 \\ +3.1 \end{bmatrix}$$

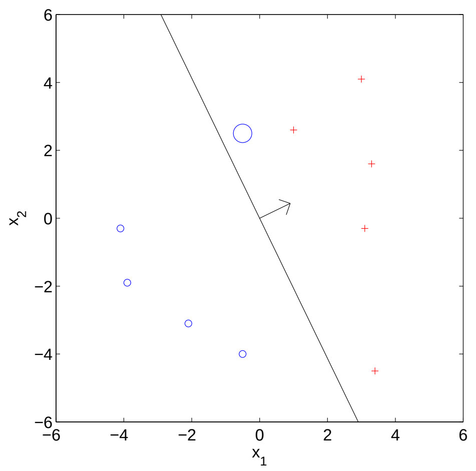


$$\begin{aligned} y_10 &= \psi(6.4 \times 3.1 + 3.1 \times -0.3) = \psi(18.91) \quad \text{– RIGHT!} \\ \boldsymbol{w} &\leftarrow \boldsymbol{w} + 0.5\eta(t_i - y_i)\boldsymbol{x}_i \\ &= [6.4 \quad -3.1] + 0.5 \times (+1 - +1)[3.1 \quad -0.3] = [6.4 \quad 3.1] \end{aligned}$$

Perceptron Example - Step #11

x_1	x_2	t
-0.5	+2.5	-1
+3.0	+4.1	+1
-4.1	-0.3	-1
+3.4	-4.5	+1
-3.9	-1.9	-1
+1.0	+2.6	+1
-2.1	-3.1	-1
+3.3	+1.6	+1
-0.5	-4.0	-1
+3.1	-0.3	+1

$$\boldsymbol{w} = \begin{bmatrix} +6.4 \\ +3.1 \end{bmatrix}$$



$$y_1 = \psi(6.4 \times -0.5 + 3.1 \times 2.5) = \psi(4.55) \quad \text{- WRONG!}$$

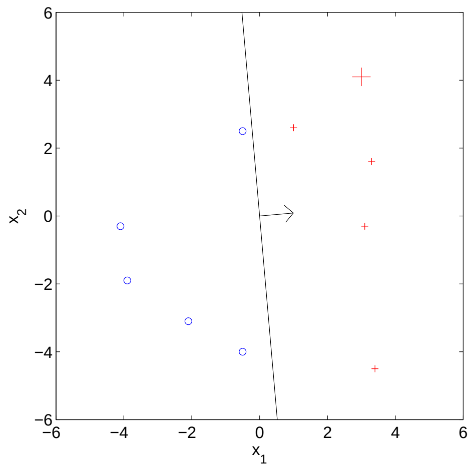
$$\boldsymbol{w} \leftarrow \boldsymbol{w} + 0.5\eta(t_i - y_i)\boldsymbol{x}_i$$

$$= [6.4 \;\; 3.1] + 0.5 \times (-1 - +1)[-0.5 \;\; 2.5] = [6.9 \;\; 0.6]$$

Perceptron Example - Step #12

x_1	x_2	t
-0.5	+2.5	-1
+3.0	+4.1	+1
-4.1	-0.3	-1
+3.4	-4.5	+1
-3.9	-1.9	-1
+1.0	+2.6	+1
-2.1	-3.1	-1
+3.3	+1.6	+1
-0.5	-4.0	-1
+3.1	-0.3	+1

$$\boldsymbol{w} = \begin{bmatrix} +6.9 \\ +0.6 \end{bmatrix}$$



$$y_2 = \psi(6.9 \times 3.0 + 0.6 \times 4.1) = \psi(23.16) \quad \text{- RIGHT!}$$

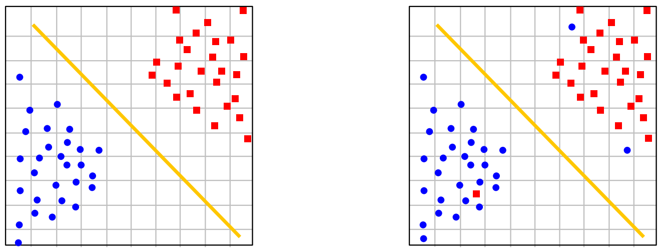
$$\boldsymbol{w} \leftarrow \boldsymbol{w} + 0.5\eta(t_i - y_i)\boldsymbol{x}_i$$

$$= [6.9 \quad -0.6] + 0.5 \times (+1 - +1)[3.0 \;\; 4.1] = [6.9 \;\; 0.6]$$

Linearly Separable Problems

- A problem is linearly separable if there exists a linear discriminant that gives perfect classification

Linearly Separable Not Linearly Separable



- The perceptron rule will converge to a correct solution for a linearly separable problem in a finite number of steps[3]
- It may **fail to converge** on linearly inseparable problems.

[3] Novikoff, A. B. (1962). On convergence proofs on perceptrons. Symposium on the Mathematical Theory of Automata, 12, 615-622. Polytechnic Institute of Brooklyn.

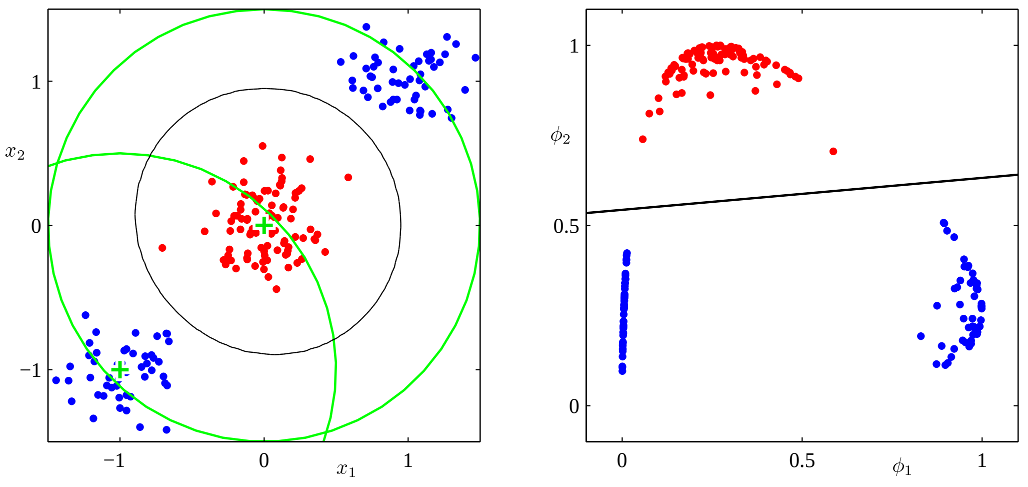
Off-Line Perceptron Algorithm

```
Algorithm GradientDescentTraining(DataSet X, t)
Returns Classifications R

initialise $\boldsymbol{w}$ to random values
initialise learning rate $\eta$
do
    initialise $\boldsymbol{\Delta w}$ to zeros
    for i=1 to n
        $y_i = \psi(\boldsymbol{w}, \boldsymbol{x}_i)$
        for j=1 to m
            $\Delta w_j \leftarrow \Delta w_j + \frac{1}{2}\eta(t_i - y_i)x_{ij}$
    for j=1 to m
        $w_j \leftarrow w_j + \Delta w_j$
while (Stopping($\boldsymbol{r}, \boldsymbol{y}$)==false)
return $\boldsymbol{w}$
```

Non-Linear Transformation

- Construct features by placing a Gaussian basis function on two patterns

$$\phi(\boldsymbol{x}) = \exp\left\{-\gamma \|\boldsymbol{x} - \boldsymbol{x}_a\|^2\right\}$$



Summary

- Linear models can be justified in many ways:
  - Optimal classifiers for Gaussian data.
  - Fisher's linear discriminant.
  - Least-squares regression.
  - The Perceptron.
- Advantages:
  - Provably optimal for data from [some] normal distributions
  - Practically effective for a wide range of problems
  - Not particularly sensitive to redundant features
- Disadvantages:
  - Cannot solve even simple non-linear problems
  - Sensitive to correlated features
  - Sensitive to ill conditioned problems (matrix inverse)