Machine Learning (part II)

Single Layer Neural Network

Angelo Ciaramella

UNIVERSITÀ PARTHENOPE
DiST
Dipartimento di Scienze e Tecnologie

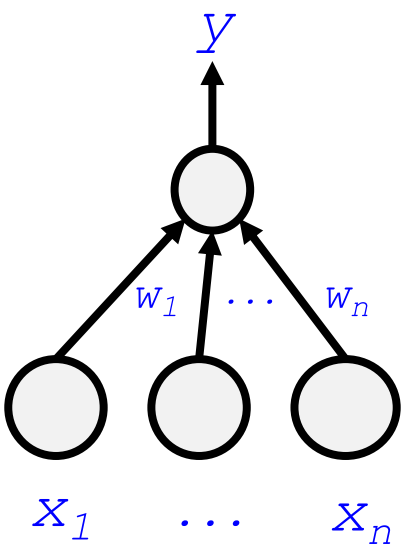
Introduction

- Linear discriminant functions
  - Linear functions of the input variables

- Generalization
  - Consider a *non-linear function*

Artificial neuron

output



weights

[0,1]

inputs

$$\mathbf{w} = \begin{bmatrix} w_1 \\ \dots \\ w_n \end{bmatrix}$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \dots \\ x_n \end{bmatrix}$$

Artificial neuron

- sum

$$z = \sum_{i=1}^{n} w_i\, x_i = \mathbf{w}^{\mathrm{T}}\mathbf{x}$$

- output

$$y = f(\mathbf{w},\mathbf{x}) = \theta(z)$$

- activation functions

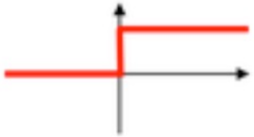
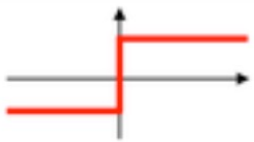
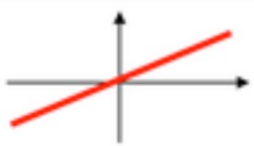
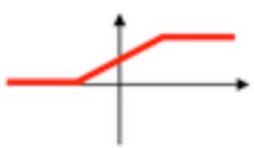
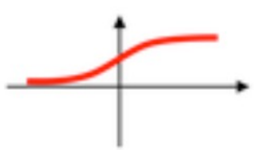
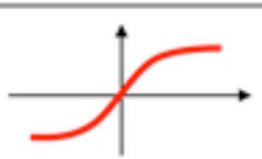
$$\theta = \begin{cases} 1 & if\ z \geq 0 \\ 0 & if\ z < 0 \end{cases}$$

Heaviside

$$\theta = \begin{cases} -1\ if\ z < 0 \\ 0\ if\ z = 0 \\ +1\ if\ z > 0 \end{cases}$$

Signum

Activation functions

Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$	Multi-layer NN	

Linear models for regression

- Linear regression

$$y = f(\mathbf{w}, \mathbf{x}) = w_0 + w_1 x_1 + \cdots + w_n x_n$$

- Basis functions extension

$$y = w_0 + \sum_{j=1}^{n-1} w_j \, \phi_j(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\Phi}(\mathbf{x})$$

$$\Phi = (\phi_0, \ldots, \phi_{M-1})^T$$

Linear models for regression

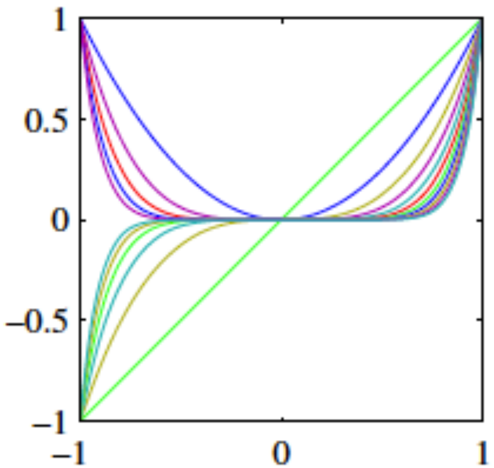
- non-linear regression basis

$$\phi_j(x) = x^j$$

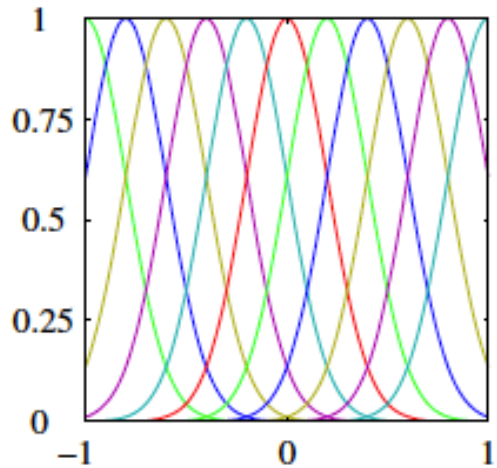
$$\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2\sigma^2}\right\}$$

$$\phi_j(x) = \vartheta\left(\frac{x-\mu_j}{\sigma}\right)$$

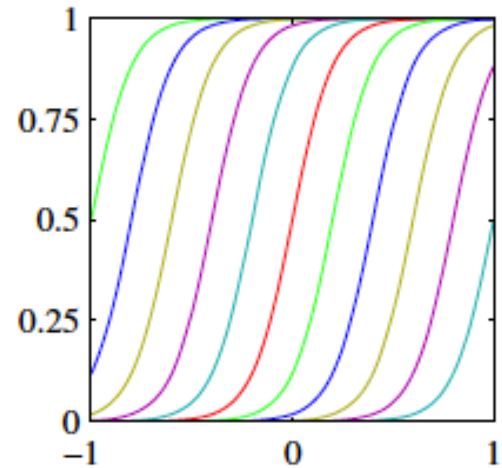
$$\vartheta(a) = \frac{1}{1+\exp(-a)}$$



Polinomial



Gaussians



Sigmoidal

Linear models for classification

- Classification

$$\mathbf{x} \rightarrow C_k \qquad k = 1, .., K$$

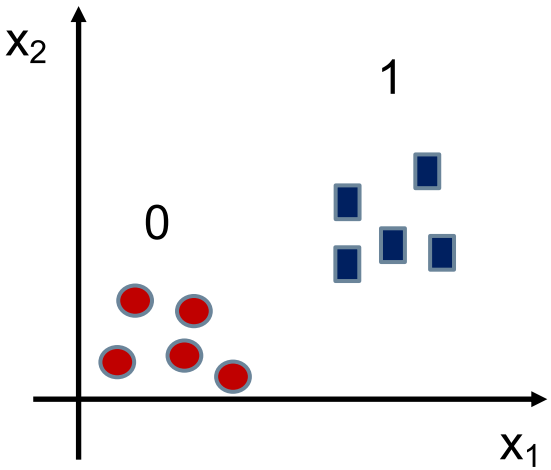
- The classes are taken to be disjoint
  - the input space is divided into decision regions whose boundaries are called decision boundaries
  - for linear models
    - (D-1)-dimensional hyperplanes within the D-dimensional input space

Target values

- K = 2 classes

$$t \in \{0,1\}$$

$$C_1 \to 1$$
$$C_2 \to 0$$

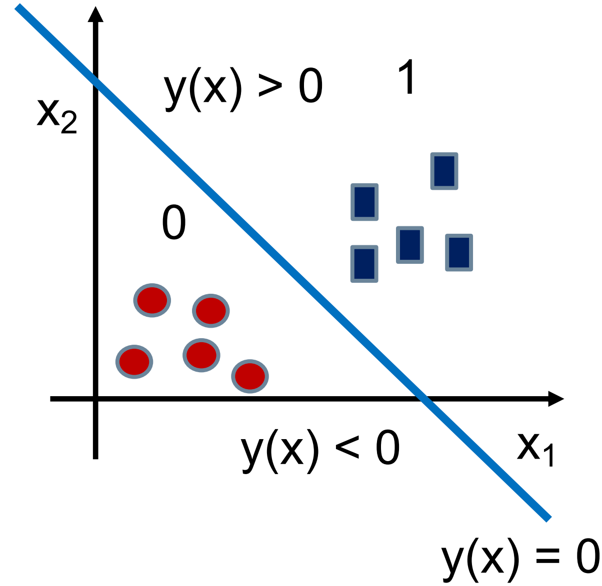


- K > 2 classes

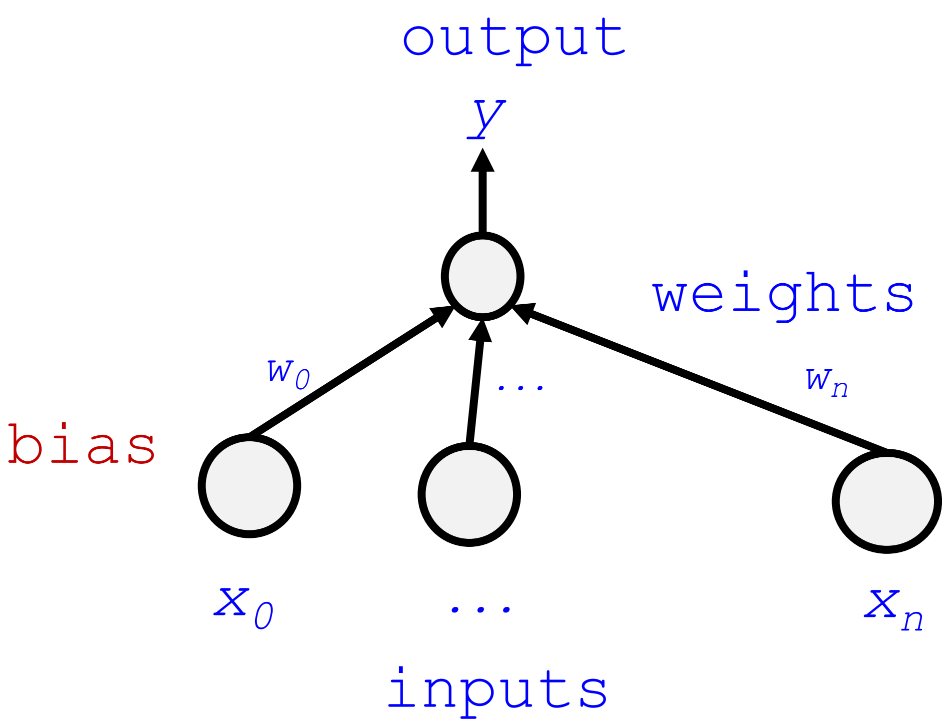
$$\mathbf{t} = (0, 1, 0\ , 0\ , 0)^T$$

1-of *K* coding (*K* = 5)

Linear discriminant function



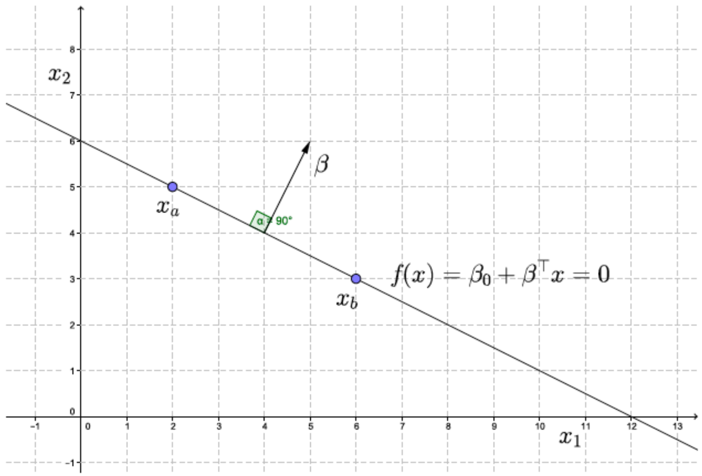
$$y(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T \mathbf{x} + w_0$$



- Learning of the parameters w and w_0

Decision surface orientation

- Decision boundary



$$y(\mathbf{x}, \mathbf{w}) = 0$$

- Two points

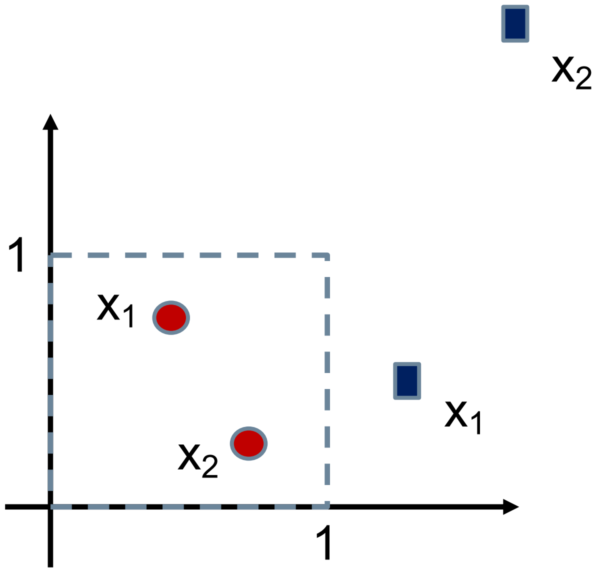
$$y(\mathbf{x_A}, \mathbf{w}) = 0$$
$$y(\mathbf{x_B}, \mathbf{w}) = 0$$

$$\mathbf{w}^T(\mathbf{x_A} - \mathbf{x_B}) = 0$$

w is orthogonal to every vector within the decision surface determining the orientation of the decision surface

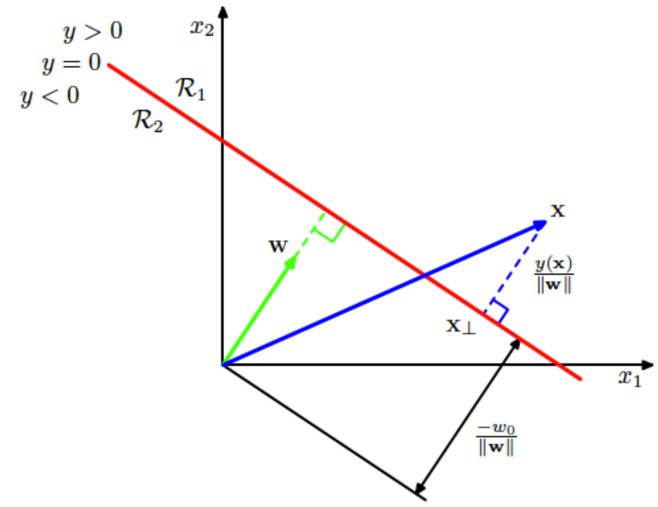
Decision surface distance

- normalization



$$\tilde{x}_i = \frac{x_i}{\|\mathbf{x}\|} = \frac{x_i}{\sqrt{\mathbf{x}\mathbf{x}^T}} = \frac{x_i}{\sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}}$$

Decision surface distance



- point $\mathbf{x}$ on

$$y(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T\mathbf{x} + w_0 = 0$$

$$\frac{\mathbf{w}^T\mathbf{x}}{\|\mathbf{w}\|} = -\frac{w_0}{\|\mathbf{w}\|}$$

normal distance from the origin to decision surface

norm

$$\|\mathbf{w}\| = \sqrt{\mathbf{w}\mathbf{w}^T} = \sqrt{w_1^2 + w_2^2 + \cdots + w_n^2}$$

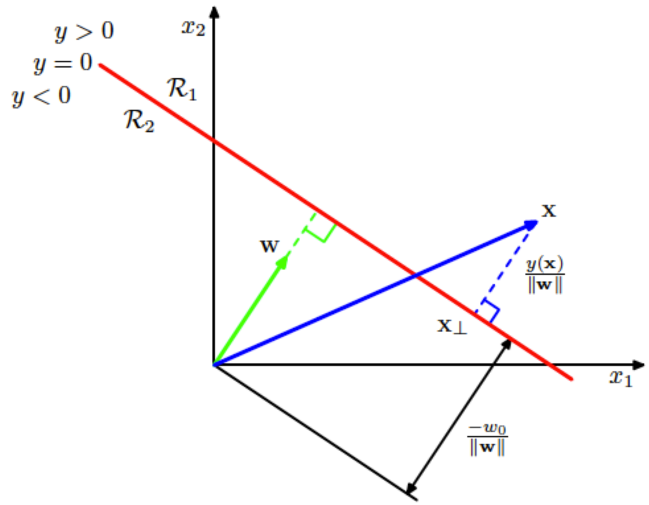
w_0 determines the location of the decision surface

DS point distance

- arbitrary point **x**

orthogonal projection onto the decison surface

$$\mathbf{x} = \mathbf{x}_{\perp} + r\frac{\mathbf{w}}{\|\mathbf{w}\|}$$

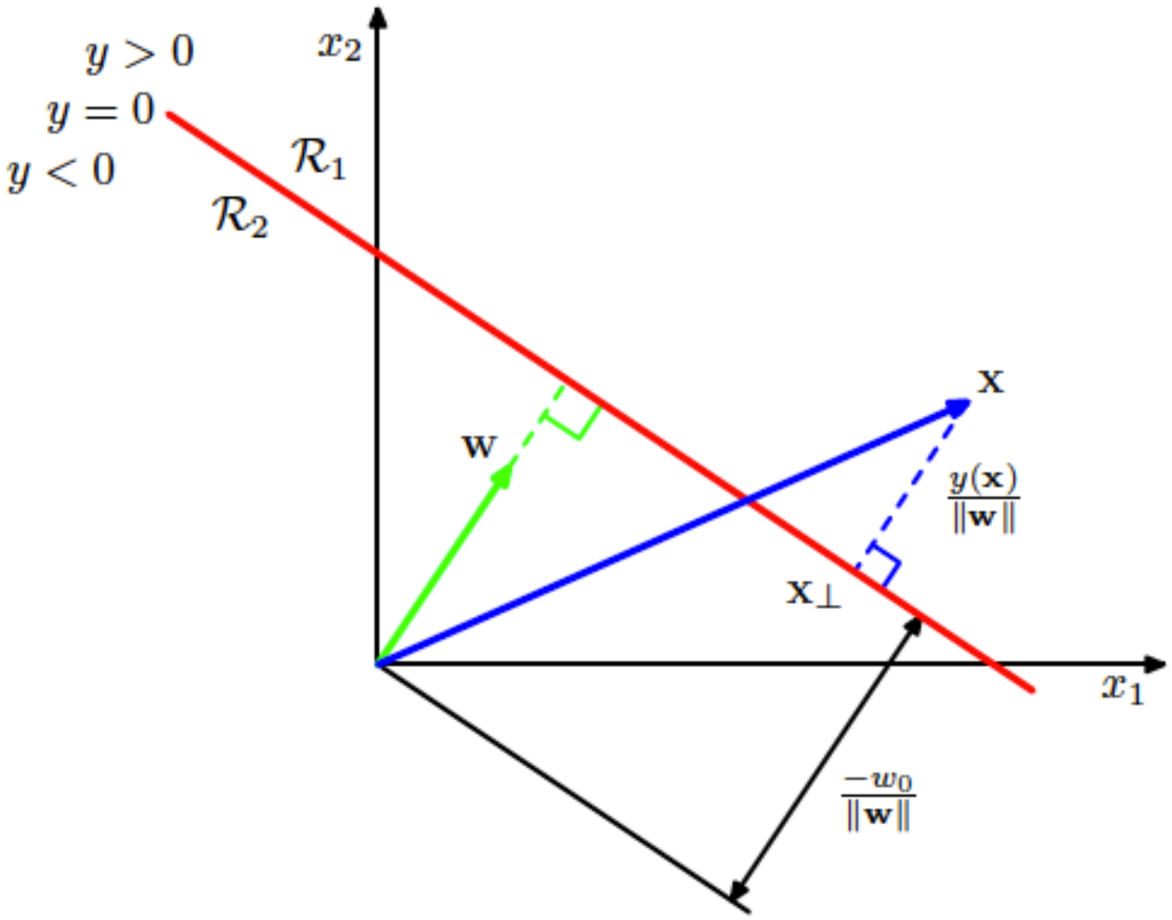


Multiplying both sides by $\mathbf{w}^T$ and adding w_0

$$r = \frac{y(\mathbf{x})}{\|\mathbf{w}\|}$$

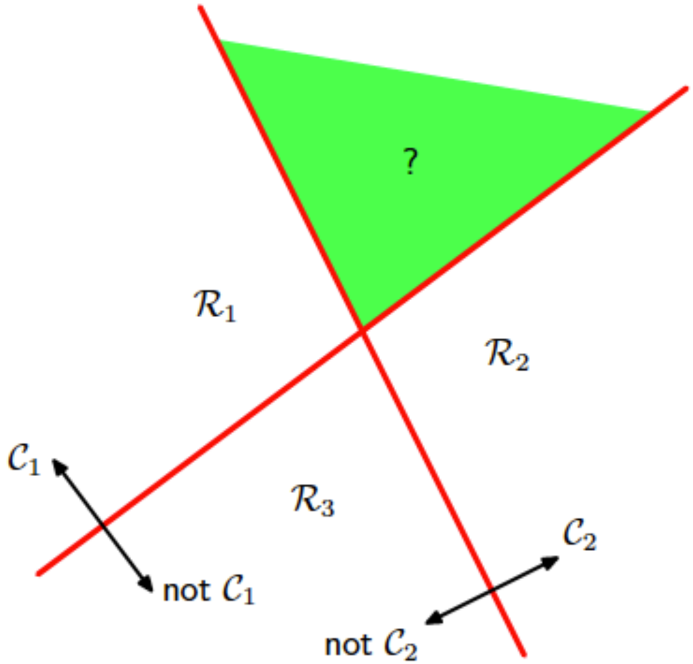
r perpendicular distance of the point **x** from the decision surface

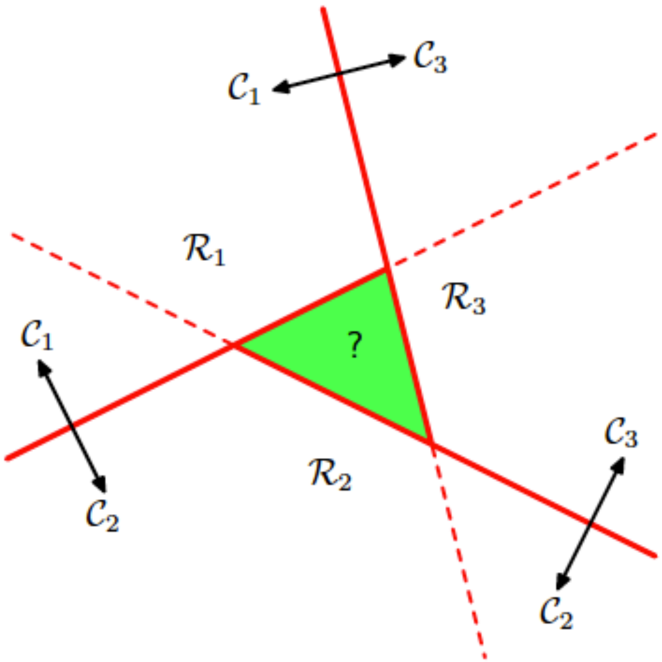
Linear discriminant function



Multiple classes

- Approaches
  - one-versus-the rest
  - one-versus-one



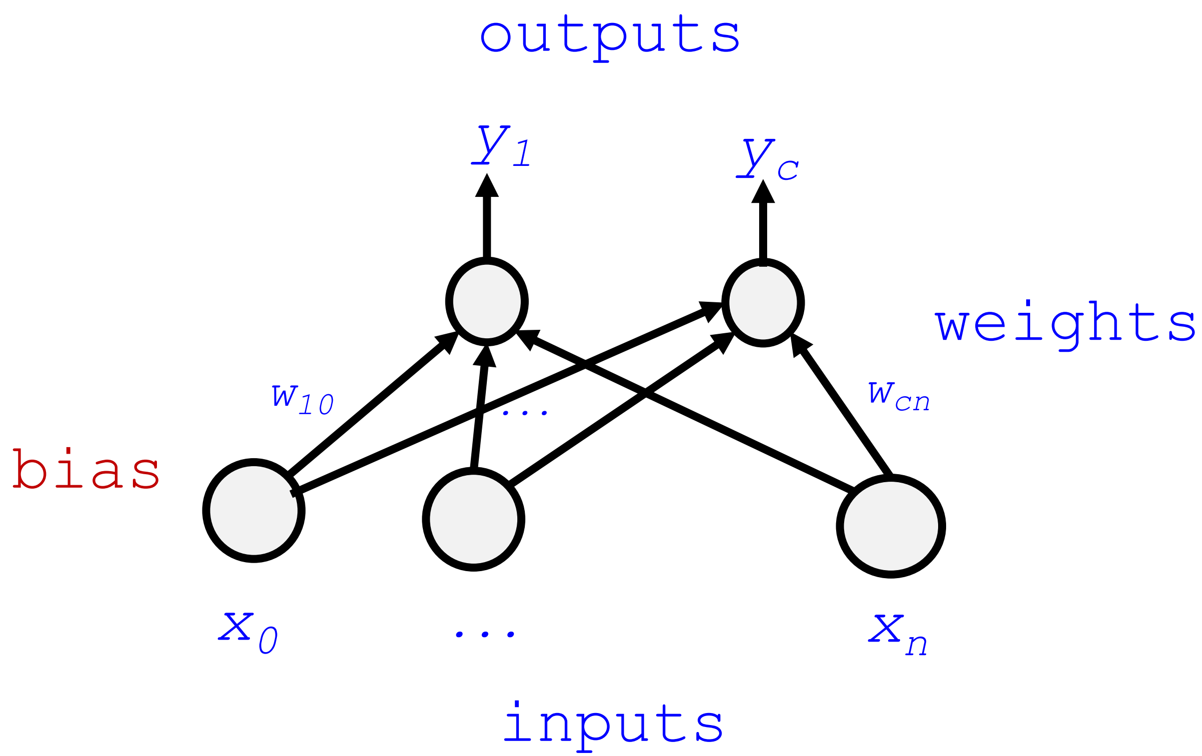


Multiple linear discriminant

- K > 2 classes

$$y_k(\mathbf{x}, \mathbf{w}_k) = \mathbf{w}_k{}^T\mathbf{x} + w_{k0}$$

$$y_k(\mathbf{x}, \mathbf{w}_k) > y_j(\mathbf{x}, \mathbf{w}_j) \quad for\ all\ j \neq k \ \rightarrow C_k$$



- bias is a threshold

Multiple linear discriminant

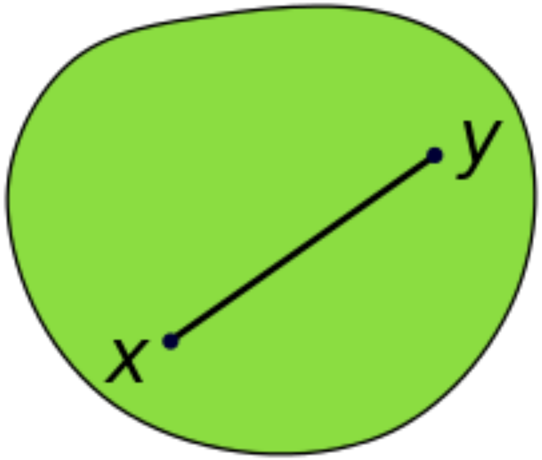
- K > 2 classes

$x_0 = 1$

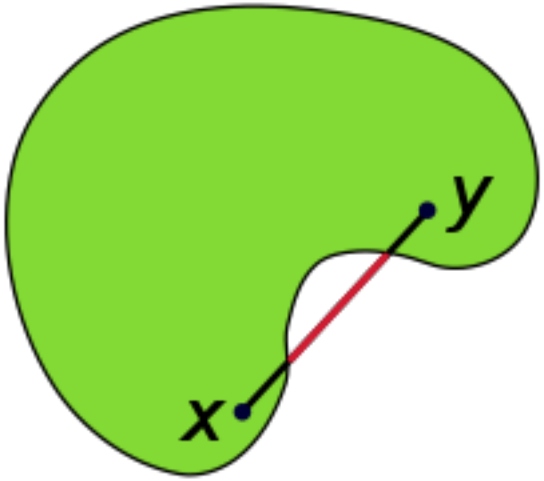
$$y_k(\mathbf{x}, \mathbf{w}_k) = \sum_{i=1}^{n} w_{ki} x_i + w_{k0} = \sum_{i=0}^{n} w_{ki} x_i$$

- Output unit has the largest activation
  - Set of decision regions which are always simply connected and convex

Multiple linear discriminant



$\mathcal{R}_k$ is convex



$\mathcal{R}_k$ is no convex

Multiple linear discriminant

- The decision regions

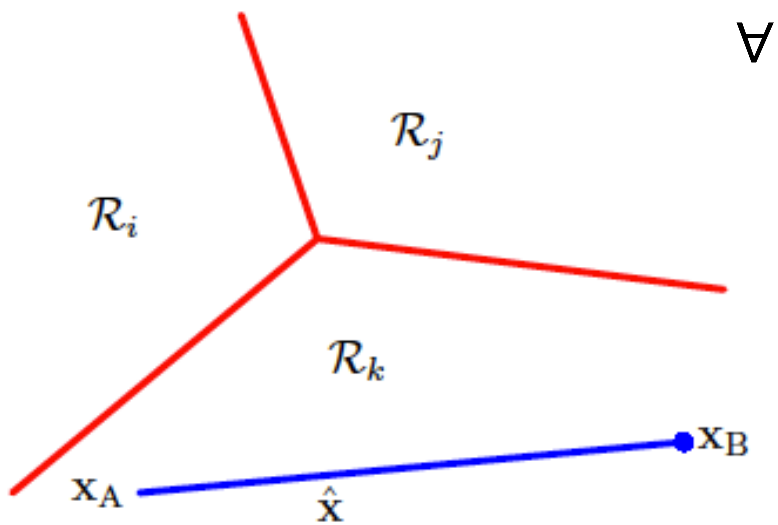
$$\hat{x} = \lambda \mathbf{x}^A + (1-\lambda)\mathbf{x}^B$$

from the linearity

$$y_k(\mathbf{x}^A) > y_j(\mathbf{x}^A)$$
$$y_k(\mathbf{x}^B) > y_j(\mathbf{x}^B)$$
$$\forall\, j \neq k$$

$$y_k(\widehat{x}) = \lambda y_k(\mathbf{x}^A) + (1-\lambda) y_k(\mathbf{x}^B)$$

$$y_k(\hat{x}) > y_j(\hat{x})$$
$$\forall\, j \neq k$$

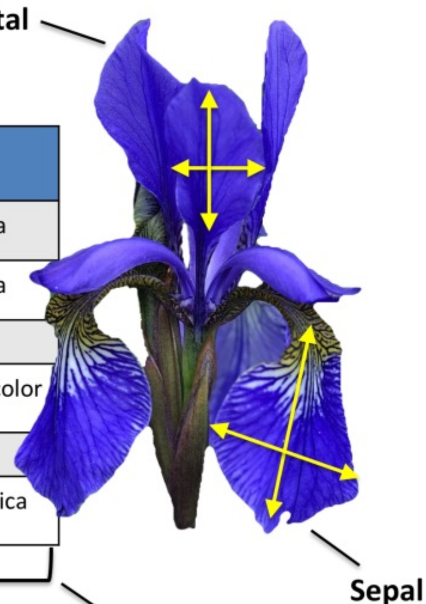


All the points on the line also line in $\mathcal{R}_k$ so the region must be simply connected and convex

Dataset - iris

Samples
(instances, observations)

	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
			...		
50	6.4	3.5	4.5	1.2	Versicolor
			...		
150	5.9	3.0	5.0	1.8	Virginica



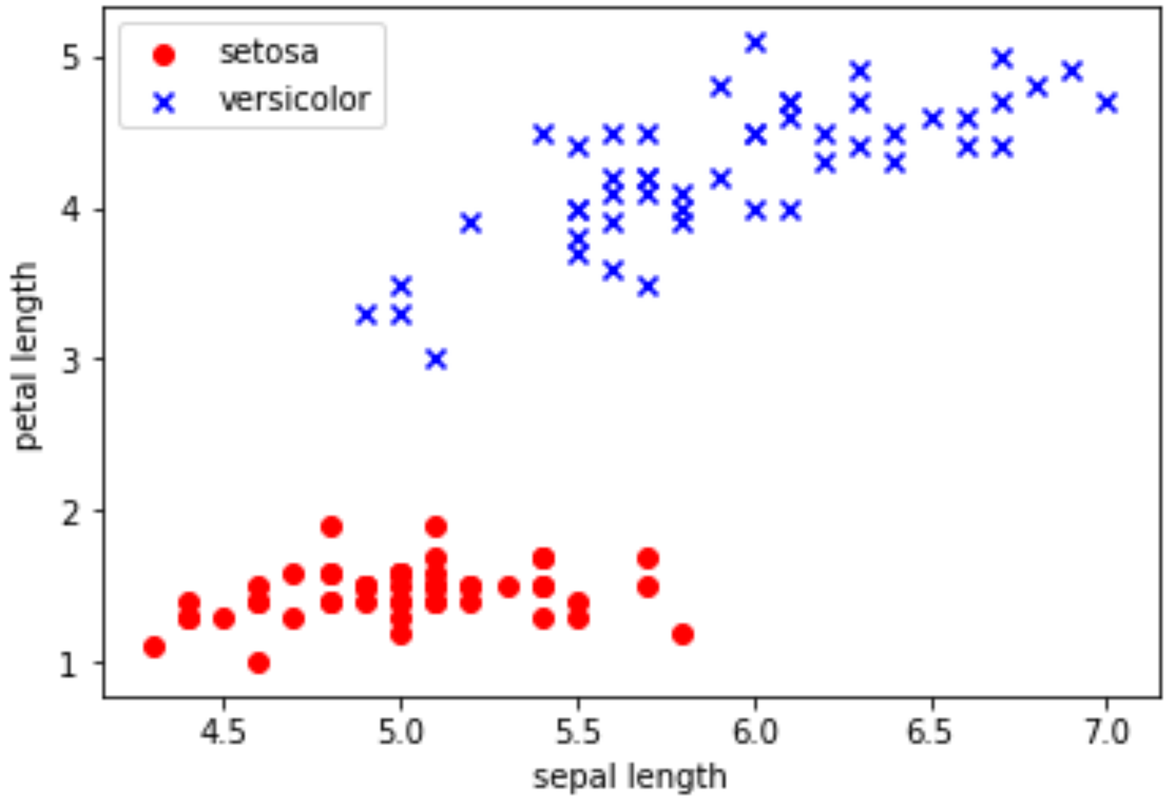
Features
(attributes, measurements, dimensions)

x_i

Class labels
(targets)

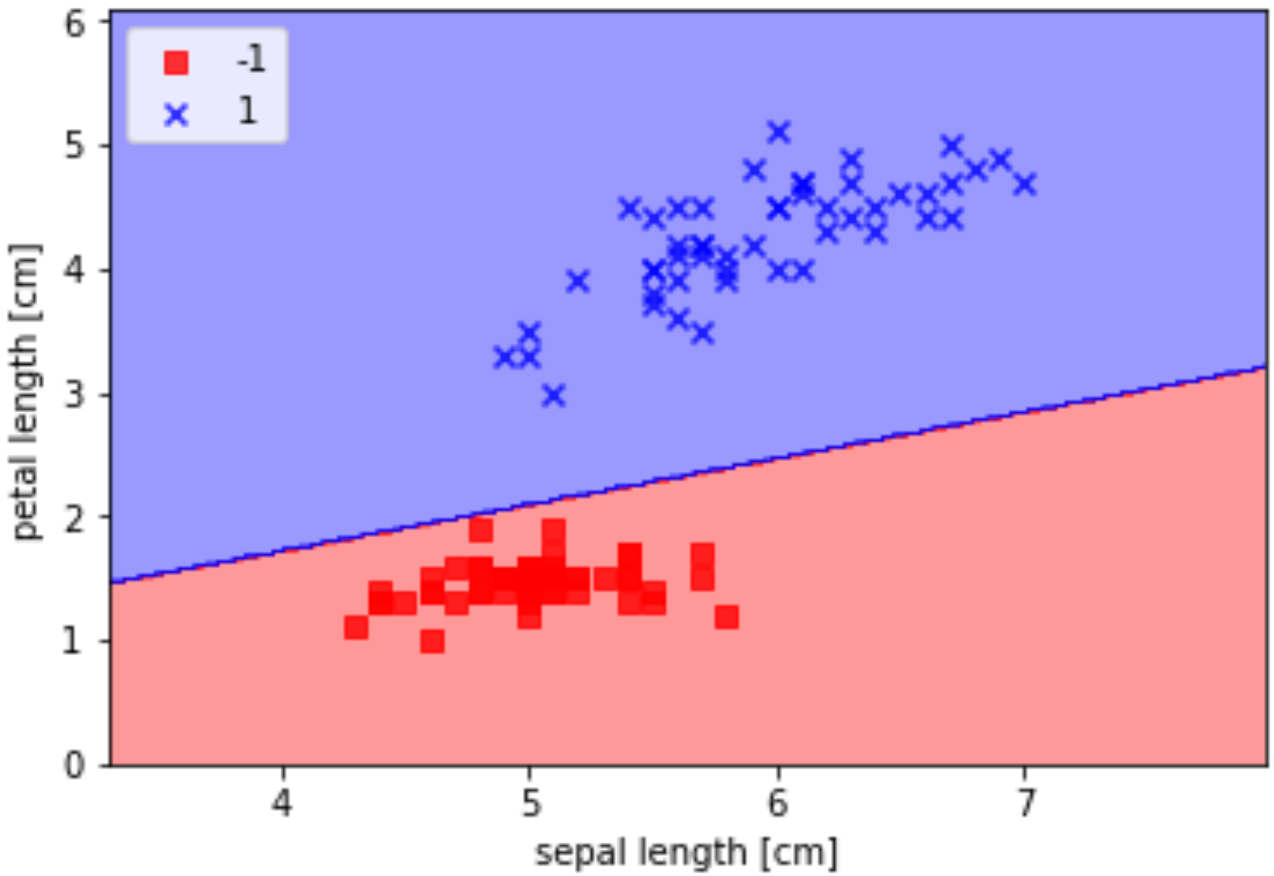
t_i

Iris visualization



2D Plot of IRIS data

Decision boudary

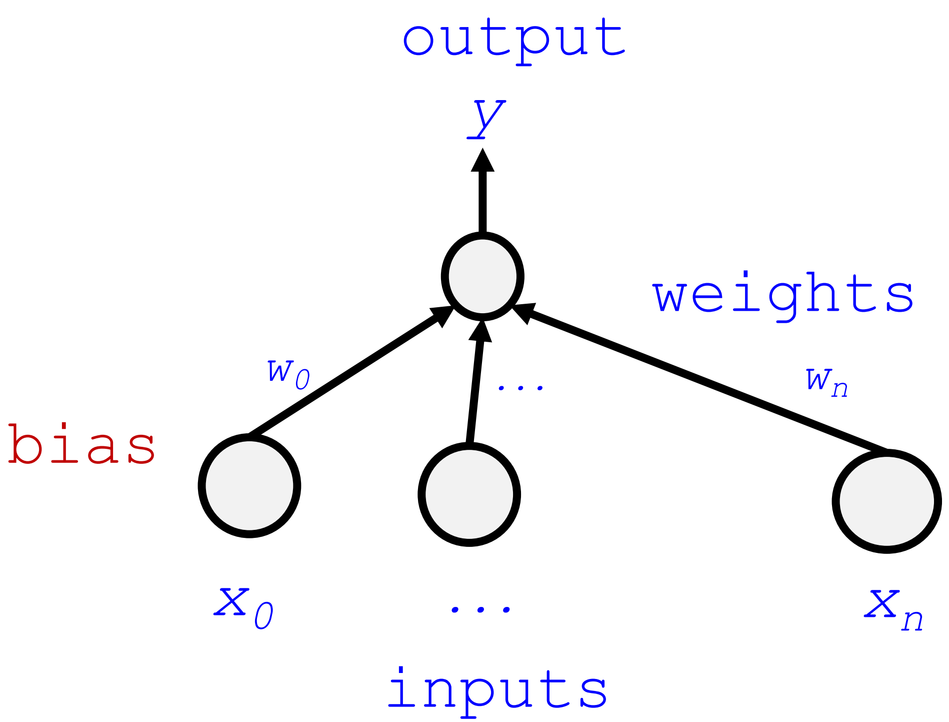


Decision boundary after learning

Linear discriminant function

activation function

$$y(\mathbf{x}) = \boldsymbol{\sigma}(\mathbf{w}^T\mathbf{x} + w_0)$$

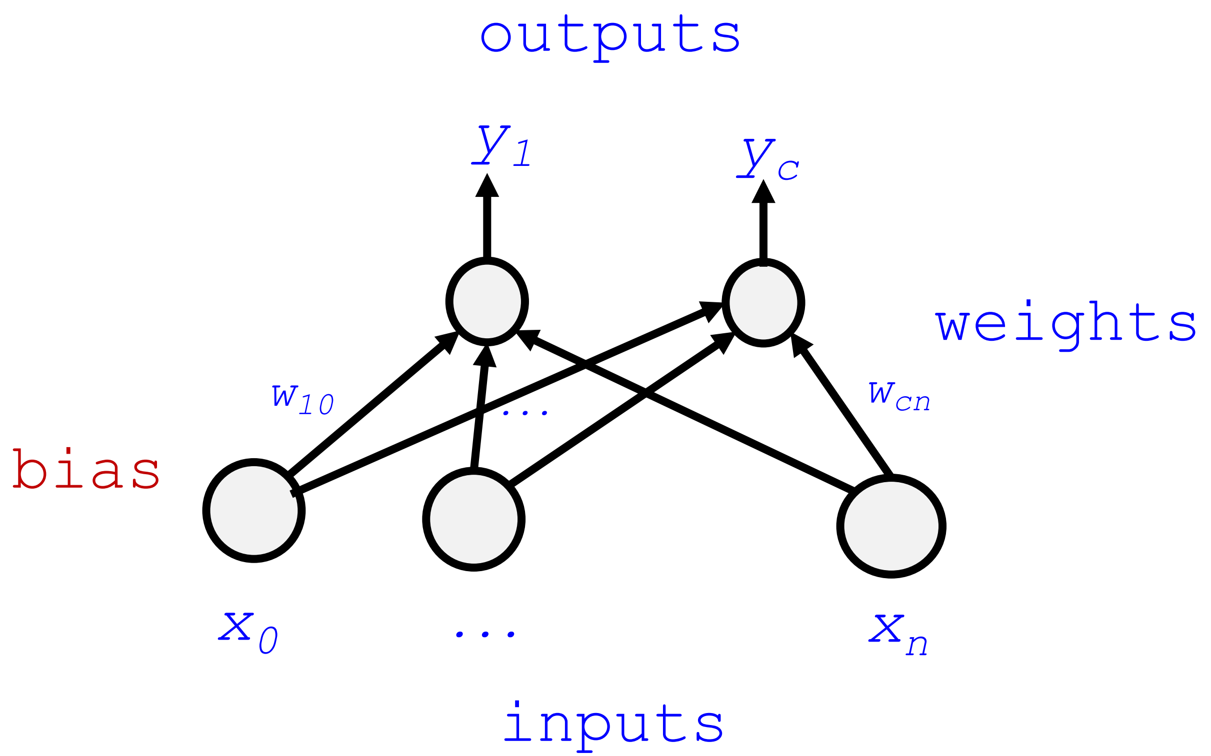


- bias is a threshold

Multiple linear discriminant

- K > 2 classes

$$y_k(\mathbf{x}) = \boldsymbol{\sigma}(\mathbf{w}_k{}^T\mathbf{x} + w_{k0})$$



- bias is a threshold

Probabilistic view

- Classification
  - probabilistic view of classification
  - models with linear decision boundaries arise from simple assumptions about the distribution of the data
- Posterior probability of class C_1

a posterior probability

likelihood

a prior probability

$$p(\mathcal{C}_1|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1) + p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)}$$

normalization factor

$$= \frac{1}{1 + \exp(-a)} = \sigma(a)$$

Probabilistic view

- Classification
  - probabilistic view of classification
  - models with linear decision boundaries arise from simple assumptions about the distribution of the data
- Posterior probability of class C_1

a posterior probability

likelihood

a prior probability

$$p(\mathcal{C}_1|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1) + p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)}$$

normalization factor

$$= \frac{1}{1+\exp(-a)} = \sigma(a)$$

Probabilistic view

- Making decision
  - X-ray image x
  - $P(C_1)$ probability that a person has cancer
  - $P(C_1|x)$ probablity that a person has cancer after oberving information of X-ray

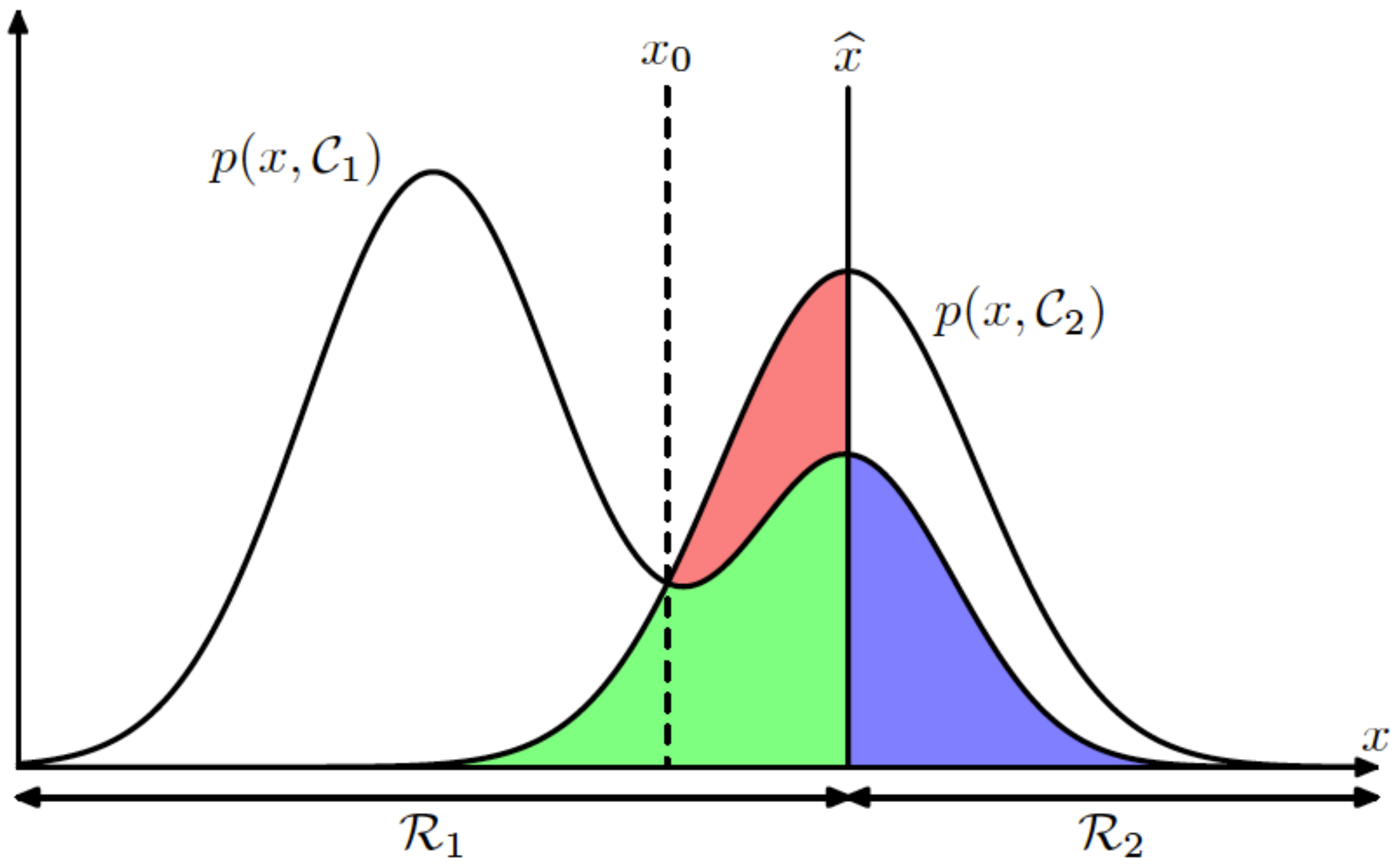
Minimizing the misclassification rate

- Decision boundaries

$$
\begin{aligned}
p(\text{mistake}) &= p(\mathbf{x} \in \mathcal{R}_1, \mathcal{C}_2) + p(\mathbf{x} \in \mathcal{R}_2, \mathcal{C}_1) \\
&= \int_{\mathcal{R}_1} p(\mathbf{x}, \mathcal{C}_2)\,\mathrm{d}\mathbf{x} + \int_{\mathcal{R}_2} p(\mathbf{x}, \mathcal{C}_1)\,\mathrm{d}\mathbf{x}
\end{aligned}
$$

$$
\begin{aligned}
p(\text{correct}) &= \sum_{k=1}^{K} p(\mathbf{x} \in \mathcal{R}_k, \mathcal{C}_k) \\
&= \sum_{k=1}^{K} \int_{\mathcal{R}_k} p(\mathbf{x}, \mathcal{C}_k)\,\mathrm{d}\mathbf{x}
\end{aligned}
$$

Minimizing the misclassification rate



Minimizing the misclassification rate

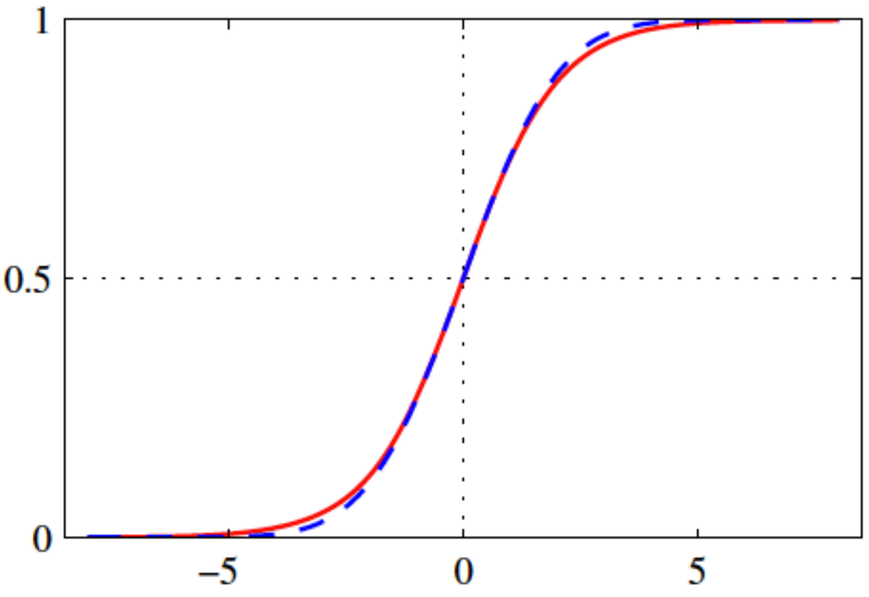
Probabilistic view

- where

$$a = \ln \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)}$$

- and logistic sigmoid function

$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$



Logistic sigmoid

- Symmetry property

$$\sigma(-a) = 1 - \sigma(a)$$

- Inverse (logit)

$$a = \ln\left(\frac{\sigma}{1-\sigma}\right)$$

Probabilistic view

- K > 2

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{\sum_j p(\mathbf{x}|\mathcal{C}_j)p(\mathcal{C}_j)}$$

$$= \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$

normalized exponential or softmax function

$$a_k = \ln p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k).$$

$$a_k \gg a_j \qquad p(\mathcal{C}_k|\mathbf{x}) \simeq 1 \qquad p(\mathcal{C}_j|\mathbf{x}) \simeq 0$$

Probabilistic view

- Gaussian class-conditional density

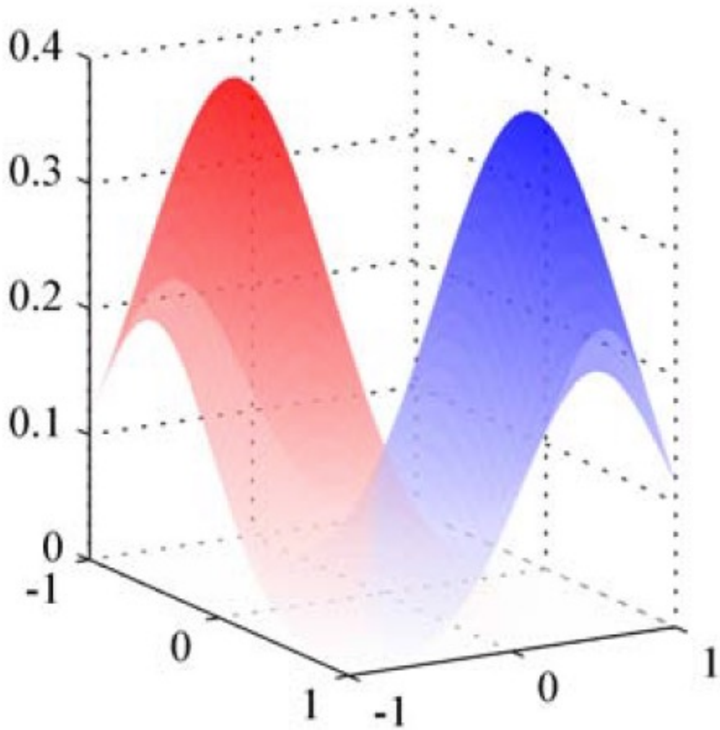
$$p(\mathbf{x}|\mathcal{C}_k) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)\right\}$$

- Posterior probability

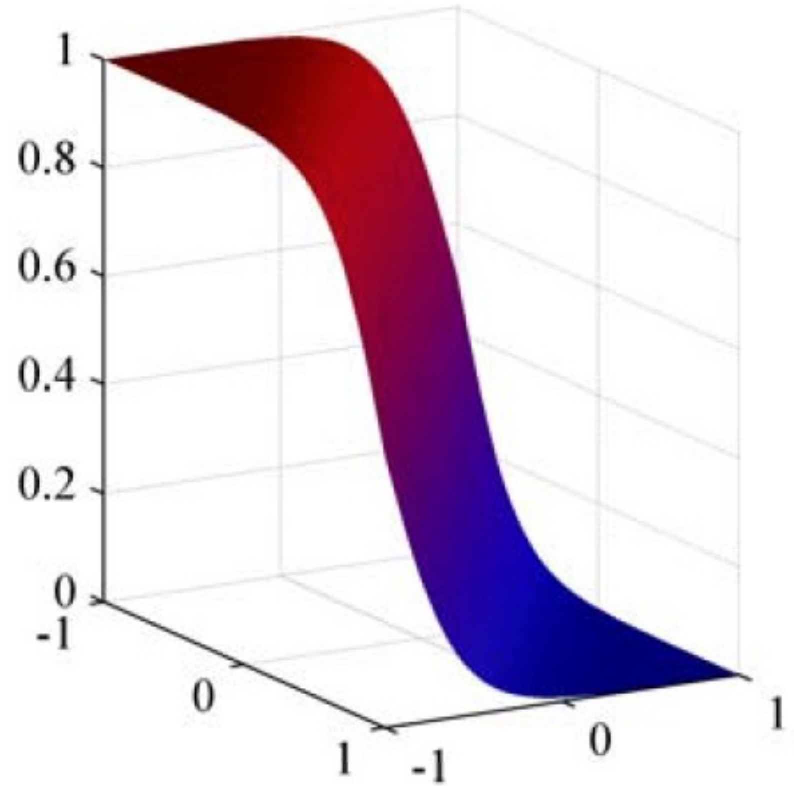
$$p(\mathcal{C}_1|\mathbf{x}) = \sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0)$$

$$\begin{aligned} \mathbf{w} &= \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \\ w_0 &= -\frac{1}{2}\boldsymbol{\mu}_1^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_2 + \ln\frac{p(\mathcal{C}_1)}{p(\mathcal{C}_2)} \end{aligned}$$

Probabilistic view



Class-conditional densities
for two classes



$p(C_1|\mathbf{x})$

Probabilistic view

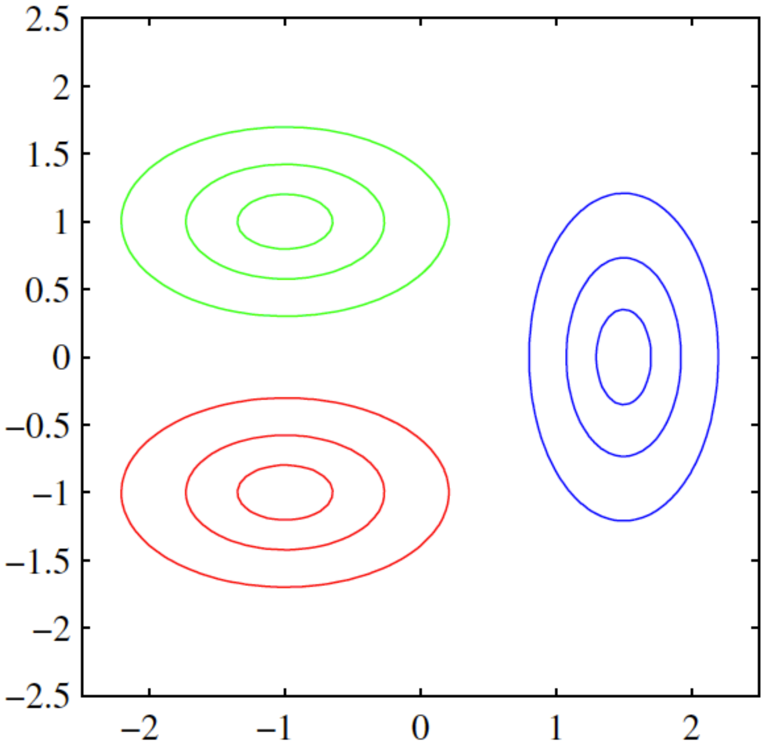
- K > 2

$$a_k(\mathbf{x}) = \mathbf{w}_k^{\mathrm{T}}\mathbf{x} + w_{k0}$$

- where

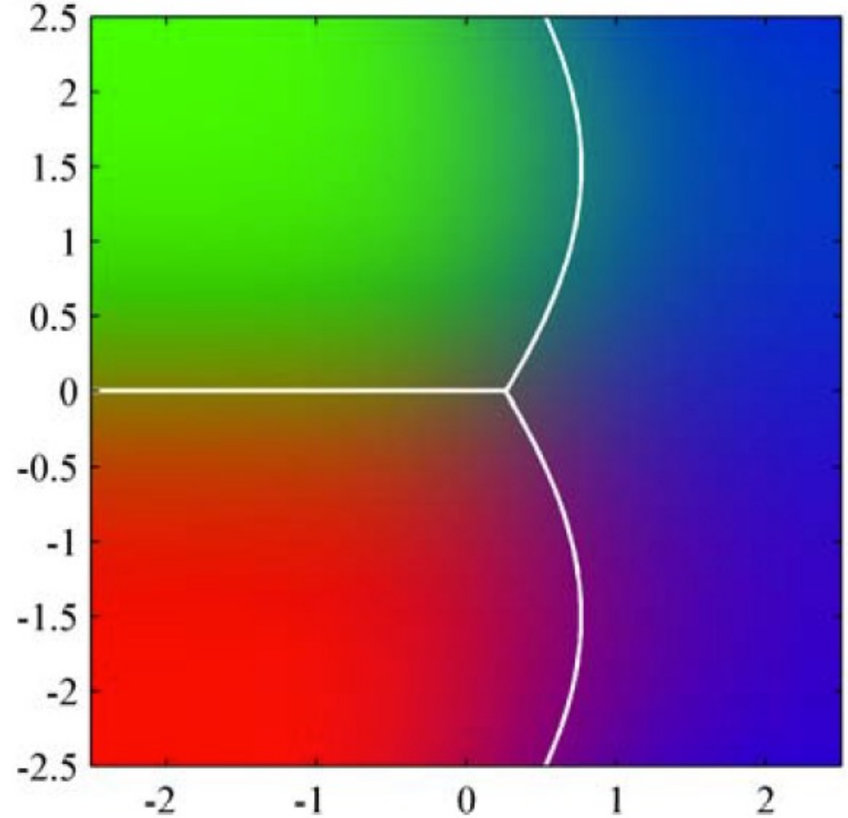
$$\begin{aligned} \mathbf{w}_k &= \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k \\ w_{k0} &= -\frac{1}{2}\boldsymbol{\mu}_k^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k + \ln p(\mathcal{C}_k) \end{aligned}$$

Probabilistic view



Class-conditional densities for three classes

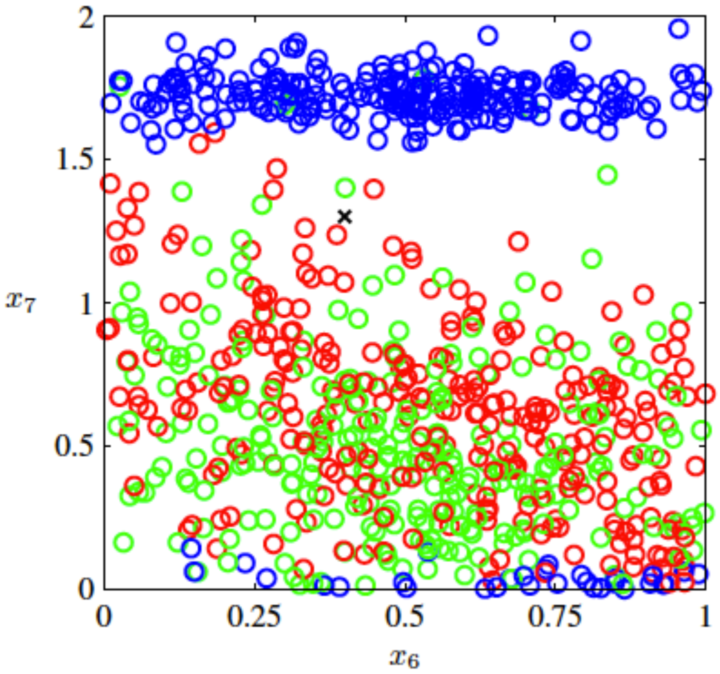
Posterior probabilities



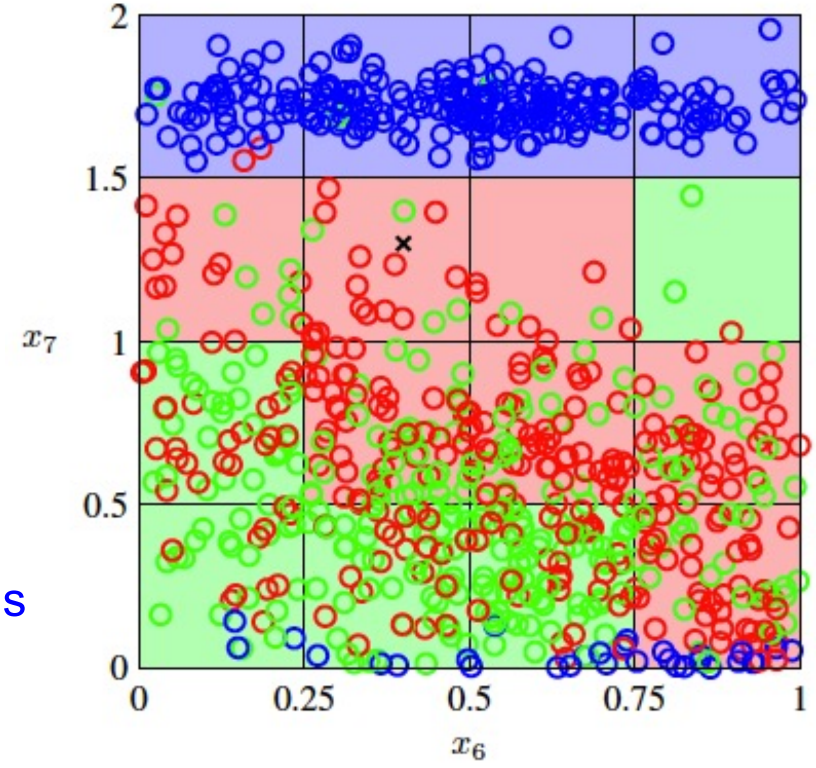
Fisher's linear discriminant

- Curse of dimensionality
  - The design of a good classifier becomes rapidly more difficult as the dimensionality of the input space increases
  - Pre-processing
    - To reduce its dimensionality
    - Fisher discriminant aims to achieve an optimal linear dimensionality reduction

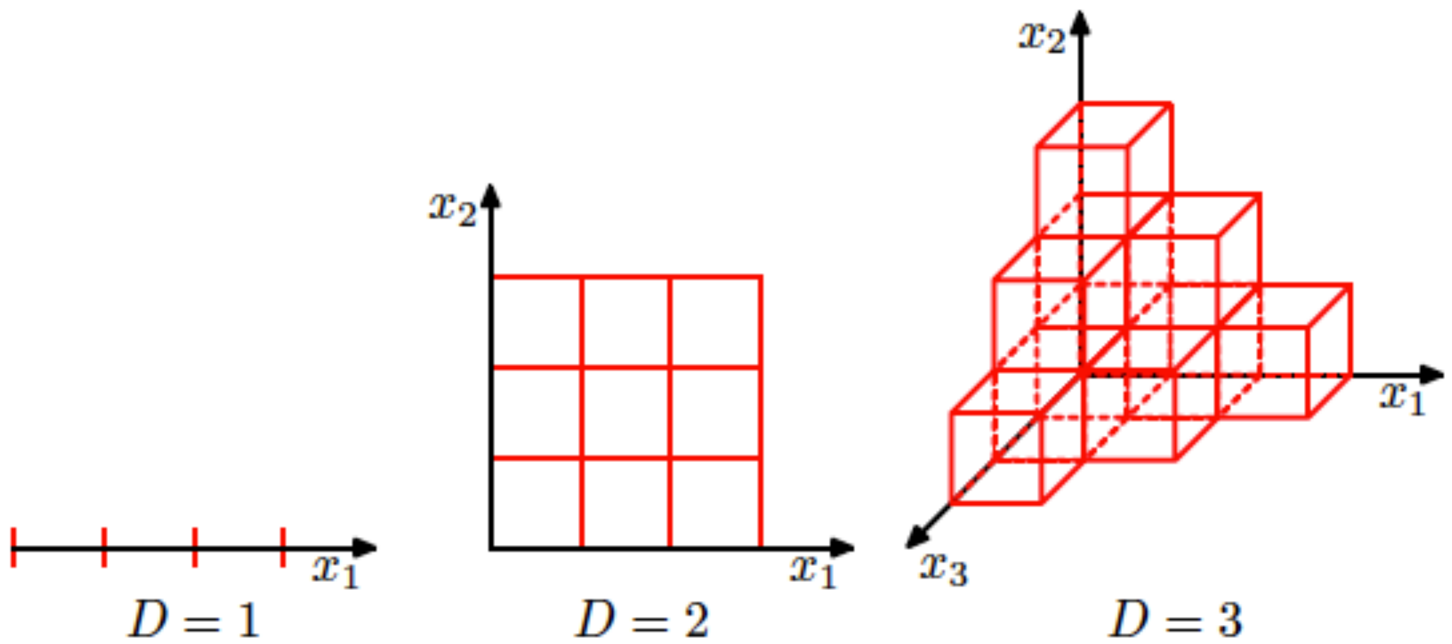
Fisher's linear discriminant



Classification by voting in the cells



Fisher's linear discriminant



The number of regions of a regular grid grows exponentially with the dimensionality of D

Fisher's linear discriminant

- Projection

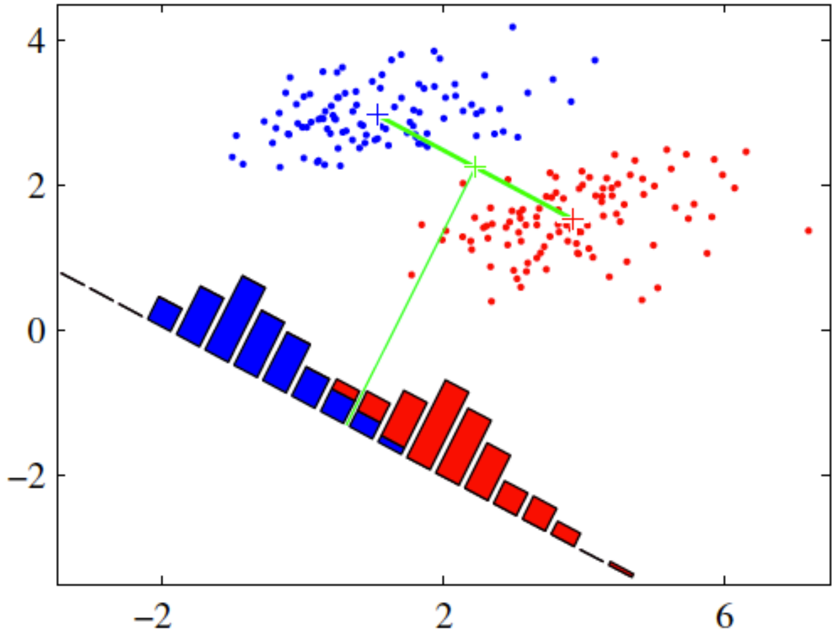
$$y = \mathbf{w}^{\mathrm{T}}\mathbf{x}. \qquad y \geqslant -w_0 \text{ as class } \mathcal{C}_1$$

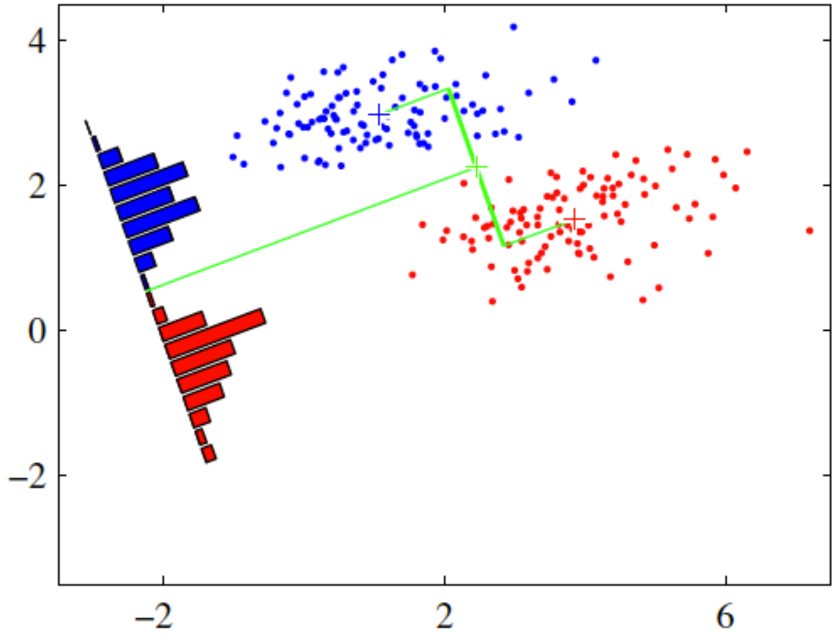
- Projection that maximizes the class separation

$$\mathbf{m}_1 = \frac{1}{N_1}\sum_{n \in \mathcal{C}_1} \mathbf{x}_n, \qquad \mathbf{m}_2 = \frac{1}{N_2}\sum_{n \in \mathcal{C}_2} \mathbf{x}_n.$$

$$m_2 - m_1 = \mathbf{w}^{\mathrm{T}}(\mathbf{m}_2 - \mathbf{m}_1)$$

Fisher's linear discriminant





Fisher directions

Fisher's linear discriminant

- within-class variance

$$s_k^2 = \sum_{n \in \mathcal{C}_k} (y_n - m_k)^2$$

- Fisher criterion

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

between-class variance

within-class variance

Fisher's linear discriminant

- Fisher criterion

$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w}}{\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w}}$$

- Between-class covariance

$$\mathbf{S}_{\mathrm{B}} = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^{\mathrm{T}}$$

- within-class covariance

$$\mathbf{S}_{\mathrm{W}} = \sum_{n \in \mathcal{C}_1} (\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^{\mathrm{T}} + \sum_{n \in \mathcal{C}_2} (\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^{\mathrm{T}}$$

Fisher's linear discriminant

- Differentianting with respect to weights

direction of $(m_2 - m_1)$

$$(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w})\mathbf{S}_{\mathrm{W}}\mathbf{w} = (\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w})\mathbf{S}_{\mathrm{B}}\mathbf{w}$$

scalar factor   scalar factor

$$\mathbf{w} \propto \mathbf{S}_{\mathrm{W}}^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$$

Fisher linear discriminant

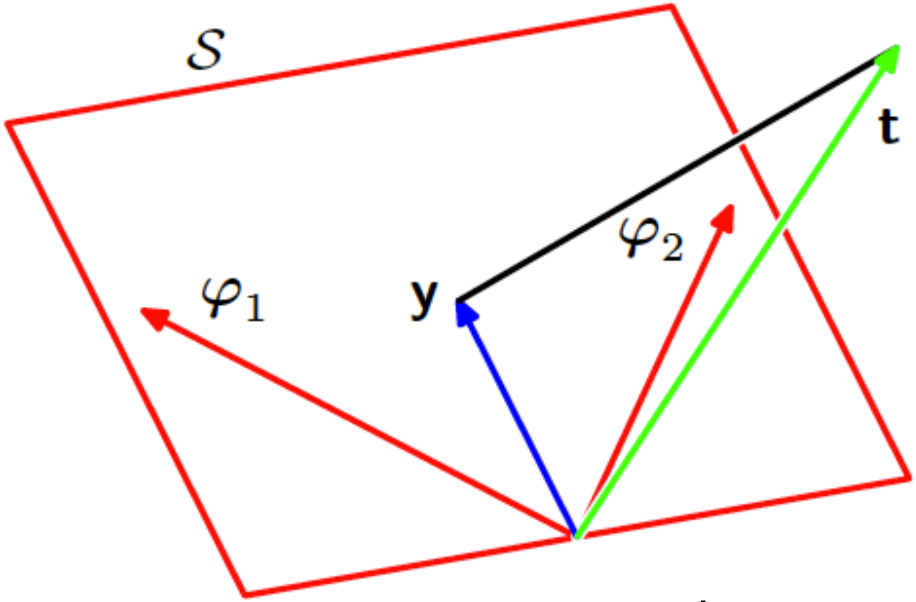
Generalization for more classes

Sum-of squares error

- Quadratic error function

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{c}\{y_k(\mathrm{x}^n;\mathrm{w}) - t_k^n\}^2$$

Geometrical representation



$$y = \sum_{j=0}^{M} w_j \phi_j^n$$

$$\frac{\partial E}{\partial w_j} = 0 = \phi_j^T(y - t)$$

$y = t_{||}$ Solution

Gradient descent

- Error function

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} E^n(\mathrm{w})$$

- Stocchastic gradient descent algorithm

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E^n$$

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \frac{\partial E^n}{\partial \mathbf{w}}$$

The perceptron algorithm

- Output

$$y(\mathbf{x}) = f\left(\mathbf{w}^{\mathrm{T}}\phi(\mathbf{x})\right)$$

- Activation function

$$f(a) = \begin{cases} +1, & a \geqslant 0 \\ -1, & a < 0 \end{cases}$$

The perceptron algorithm

- Classification

$$\mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_n) > 0 \qquad \mathcal{C}_1$$

$$\mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_n) < 0 \qquad \mathcal{C}_2$$

$$t \in \{-1, +1\}$$

- Perceptron criterion

$$E_{\mathrm{P}}(\mathbf{w}) = -\sum_{n \in \mathcal{M}} \mathbf{w}^{\mathrm{T}}\phi_n t_n$$

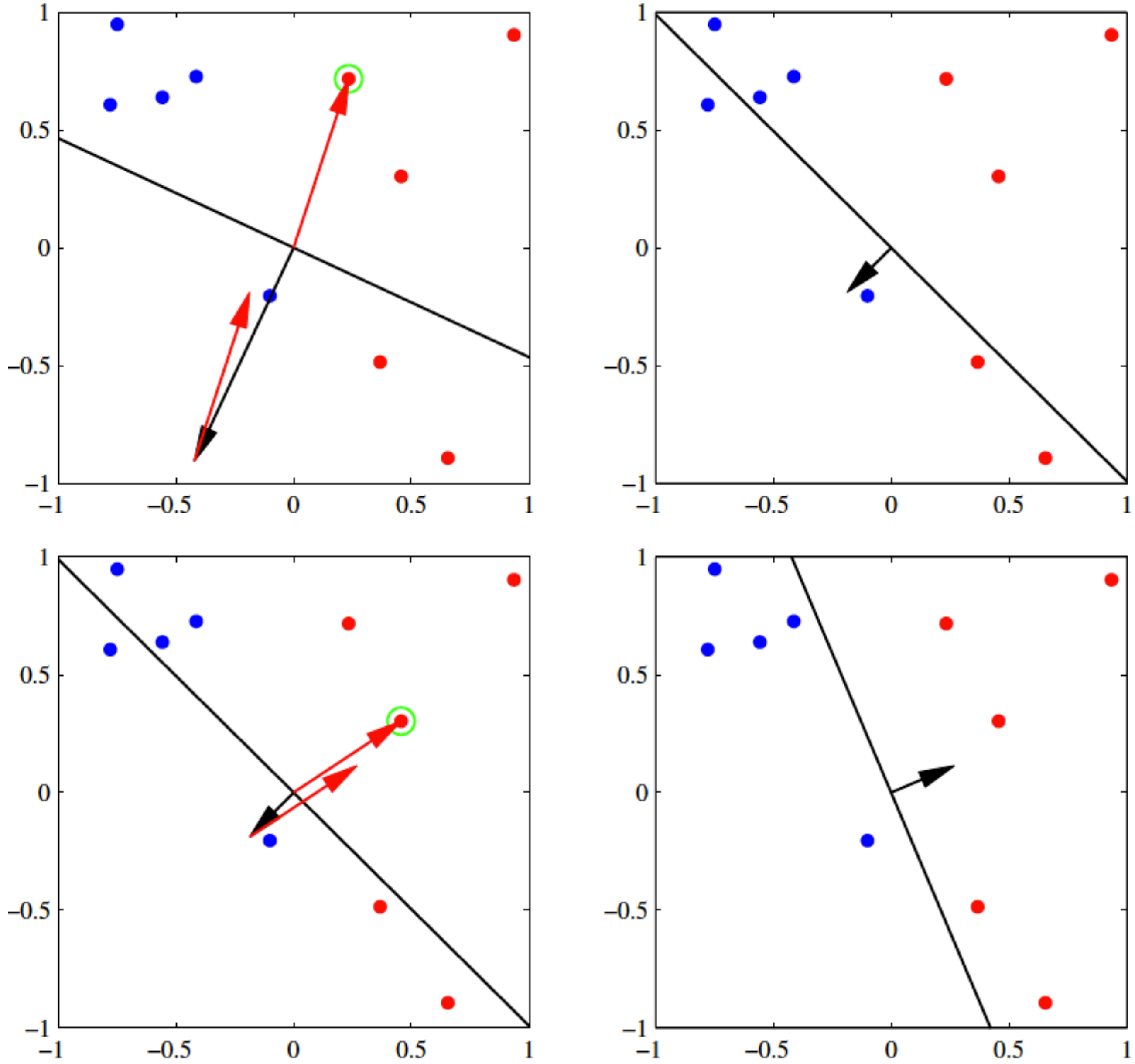
The perceptron algorithm

- Gradient descent algorithm

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_{\mathrm{P}}(\mathbf{w}) = \mathbf{w}^{(\tau)} + \eta \boldsymbol{\phi}_n t_n$$

- Perceptron convergence theorem
  - if there exists an exact solution (if the training data set is linearly separable) then the perceptron learning algorithm is guaranteed to find an exact soultion in a finite number of steps

The perceptron algorithm



Linear separability

- Linearly separble
  - The points can be classified correclty by a linear decision boundary
- No linearly separable
  - exclusive-OR (XOR) problem