

FU master's course – May 6th 2011

Machine learning

Julia Lasserre

Max Planck Institute for Molecular Genetics - Berlin

Notations

$\{\mathbf{x}_n, t_n\}$: n^{th} training data point and its target (label or output)

\mathbf{X} : matrix of training data

\mathbf{t} : vector of training targets (labels or outputs)

\mathcal{D} : global data, can be $\{\mathbf{X}, \mathbf{t}\}$ or \mathbf{X}

$\{\hat{\mathbf{x}}, \hat{t}\}$: a new data point and its predicted target

$\{\mathbf{x}, \mathbf{x}'\}$: data points

$\{t, t'\}$: targets (labels or outputs)

Supervised linear models

Plan

- Linear regression
 - for actual regression
 - for classification => linear discriminant functions
 - probabilistic linear regression
- Probabilistic supervised linear models
- Kernels

Supervised linear models

- Reminder: in unsupervised ML
 - one set of observations: the output
 - find a mathematical relationship between this output and a hidden (latent) variable
- Goal: find a mathematical relationship between two sets of observations
 - the relationship has unknown parameters that must be learnt
- Types of variables:
 - feature (or predictor) x : the input
 - target (or response) t : the output

Supervised linear models

- Assumption: this relationship is linear
- Linear model between input \mathbf{x} and output t
 - in one dimension $t = f(w x + w_0)$ with noise
find w and w_0
 - in D dimensions $t = f(w_1 x_1 + \dots + w_D x_D + w_0) = f(\mathbf{w}^T \mathbf{x} + w_0)$ with noise
find \mathbf{w} and w_0
 - the noise depends on the application

Supervised linear models

$t = y(\mathbf{x})$ with noise

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

f : activation function

predictions: $\hat{t} = y(\hat{\mathbf{x}})$

Plan

- *Linear regression*
 - *for actual regression*
 - for classification => linear discriminant functions
 - probabilistic linear regression
- Probabilistic supervised linear models
- Kernels

Supervised linear models

Linear regression...
for regression

Supervised linear models

$t = y(\mathbf{x})$ with noise

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

f : activation function

predictions: $\hat{t} = y(\hat{\mathbf{x}})$

Linear regression

$$t \in \mathbb{R}$$

$t = y(\mathbf{x})$ with noise

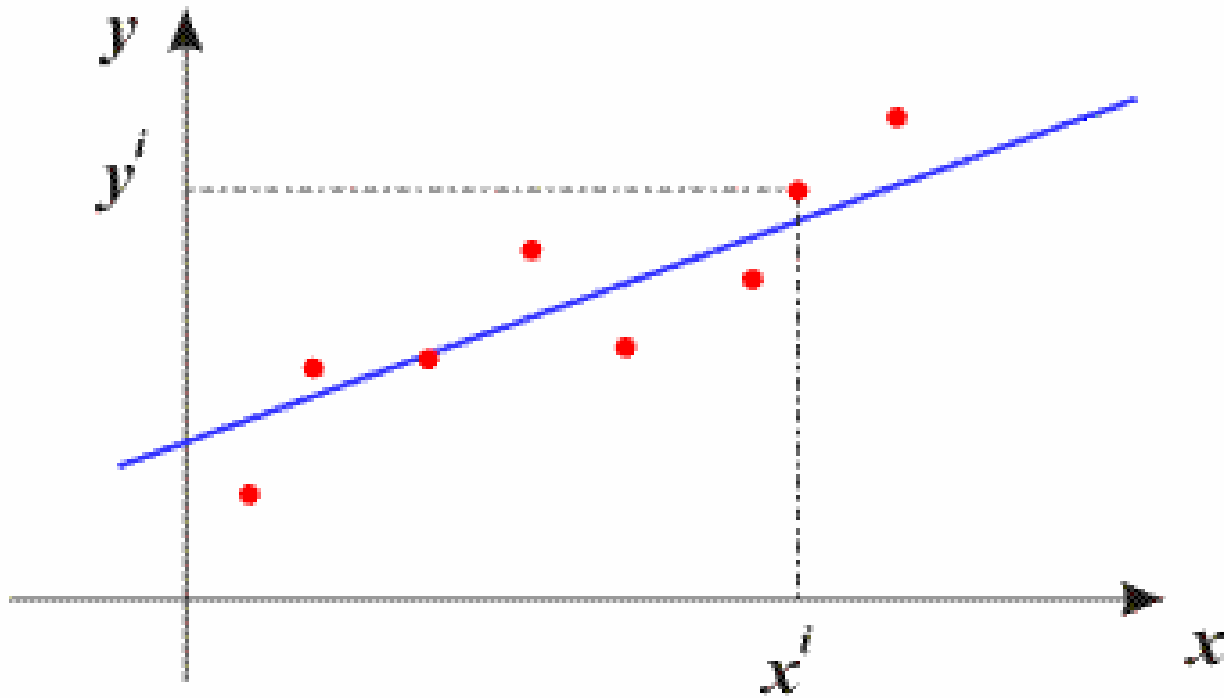
$$y(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0$$

f : identity function

predictions: $\hat{t} = y(\hat{\mathbf{x}})$

Linear regression

AI ACCESS



Least squares

- Target = model function and random noise

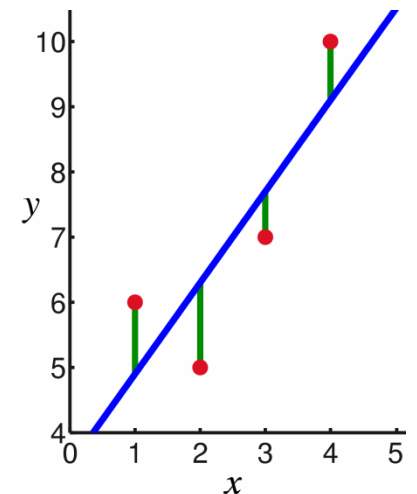
$$t = y(x) \text{ with noise}$$

$$y(x) = w x + w_0$$

- Assume additive noise

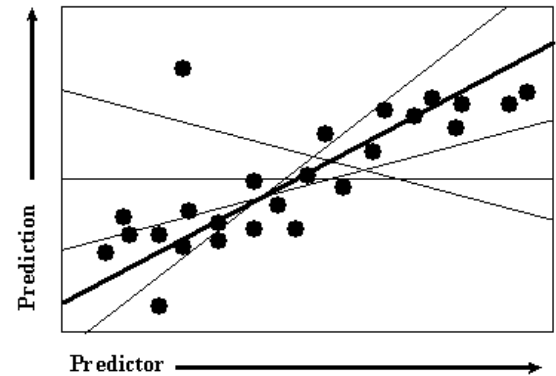
$$t = y(x) + \varepsilon$$

$$y(x) = w x + w_0$$



Least squares

- There may be several w
=> how do we choose?



- Minimise the sum-of-squares error

$$E(\mathbf{w}) = \sum_{n=1}^N (t_n - y(x_n))^2 = \sum_{n=1}^N (t_n - w x_n - w_0)^2$$

Least squares

- Compute the gradient with respect to w_0 and w

$$\frac{\partial E(w)}{\partial w_0} = -2 \sum_{n=1}^N (t_n - w x_n - w_0)$$

$$\frac{\partial E(w)}{\partial w} = -2 \sum_{n=1}^N x_n (t_n - w x_n - w_0)$$

- And set it to 0

$$\sum_{n=1}^N (t_n - w x_n - w_0) = 0$$

$$\sum_{n=1}^N x_n (t_n - w x_n - w_0) = 0$$

Least squares

The solutions are

$$w_0 = \frac{1}{N} \left(\sum_{n=1}^N t_n - w \sum_{n=1}^N x_n \right) = \bar{t} - w \bar{x}$$

$$w = \frac{\sum_{n=1}^N (x_n - \bar{x})(t_n - \bar{t})}{\sum_{n=1}^N (x_n - \bar{x})^2} = \frac{\text{cor}(x, t)}{\text{var}(x)}$$

Least squares

- Least squares in general (several dimensions)

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

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N (t_n - y(\mathbf{x}))^2$$

- Solution

$$E(\mathbf{w}) = \frac{1}{2} (\mathbf{X}\mathbf{w} - \mathbf{t})^T (\mathbf{X}\mathbf{w} - \mathbf{t})$$

$$\frac{d(E(\mathbf{w}))}{d\mathbf{w}} = \mathbf{X}^T (\mathbf{X}\mathbf{w} - \mathbf{t}) = 0 \Leftrightarrow \mathbf{w} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{t}$$

Linear regression

In R

X = your data, with one sample per row

t = your targets

```
model = lm(t~X)
```

```
plot(t,model$fitted.values)
```

Plan

- *Linear regression*
 - *for actual regression*
 - *for classification => linear discriminant functions*
 - probabilistic linear regression
- Probabilistic supervised linear models
- Kernels

Supervised linear models

Linear discriminant functions

Subplan

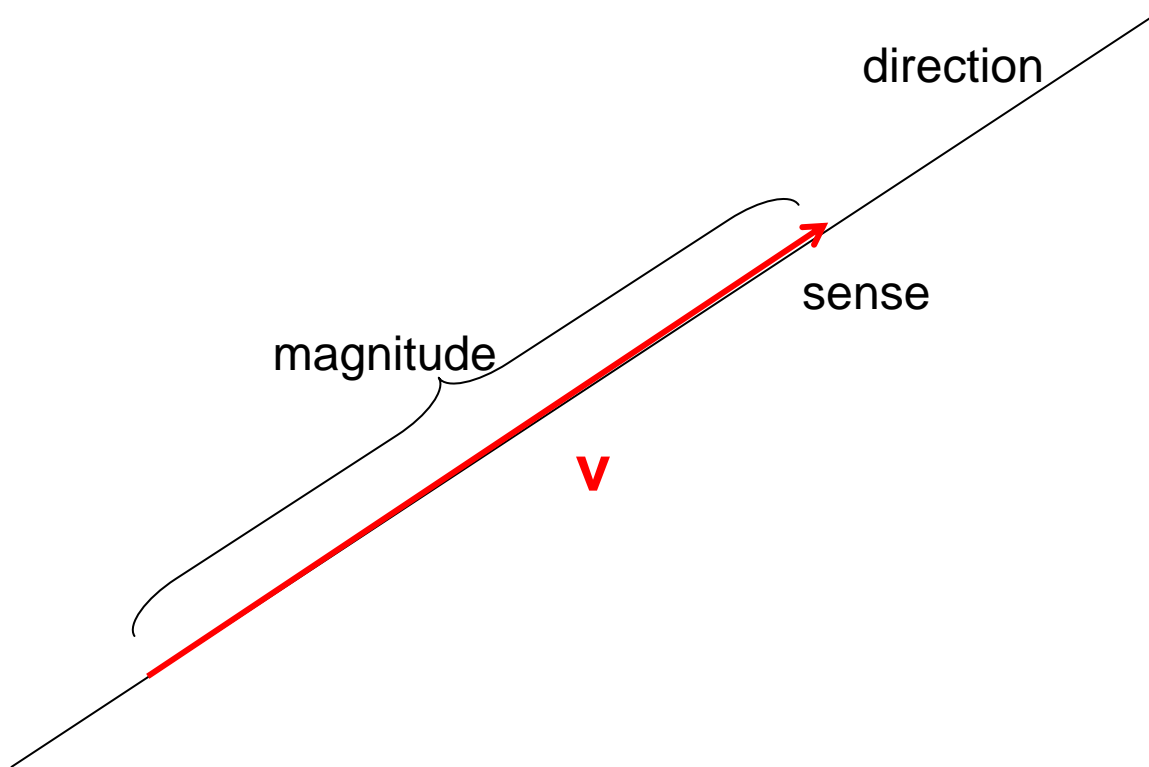
- A bit of geometry
- Linear discriminant functions
 - Regression versus classification
 - Least squares
 - Linear discriminant analysis
 - Support vector machines
- Summary

Subplan

- *A bit of geometry*
- Linear discriminant functions
 - Regression versus classification
 - Least squares
 - Linear discriminant analysis
 - Support vector machines
- Summary

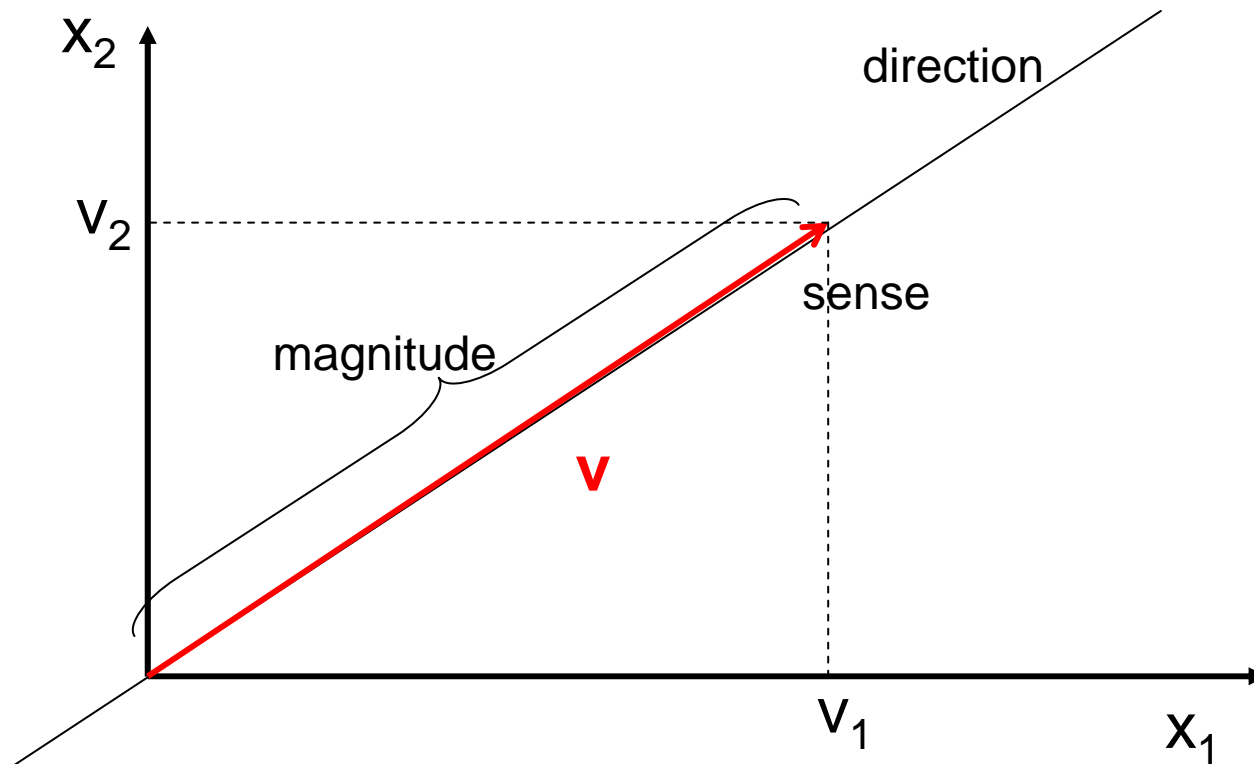
A bit of geometry

What is a vector?



A bit of geometry

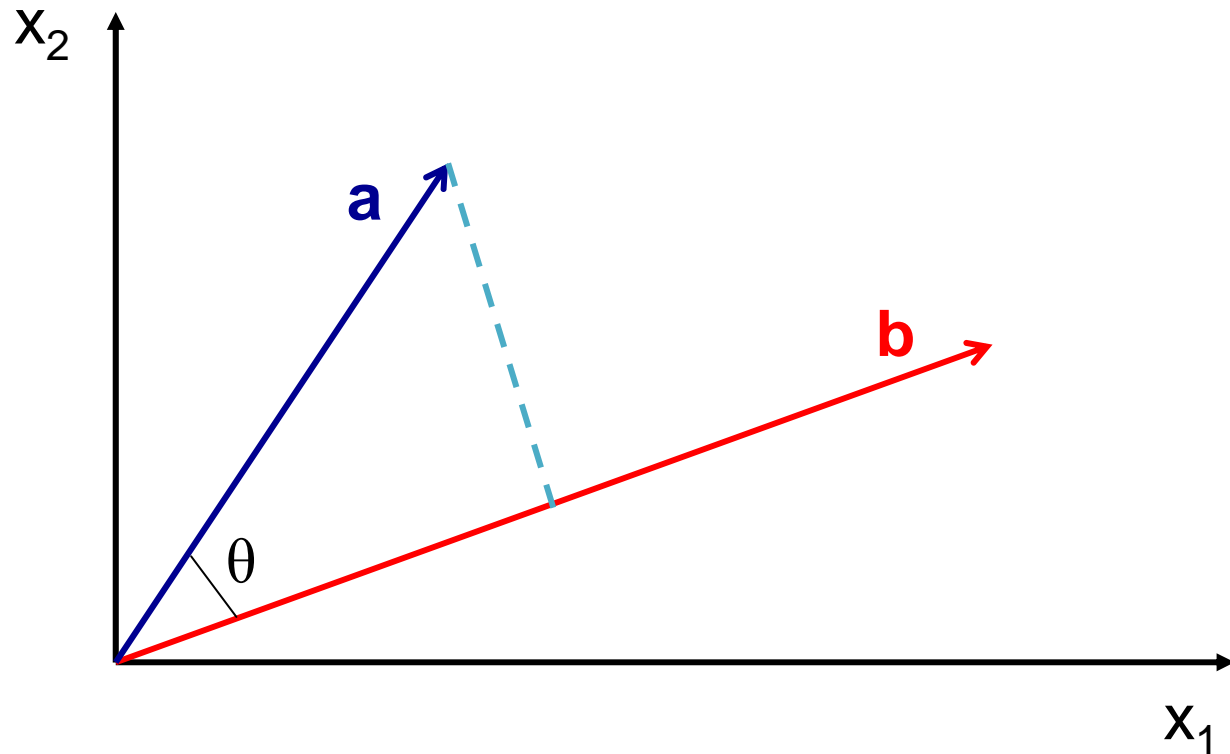
What is a vector?



A bit of geometry

What is a scalar product?

$$s = \mathbf{a}^T \mathbf{b} = |\mathbf{a}| |\mathbf{b}| \cos(\theta)$$



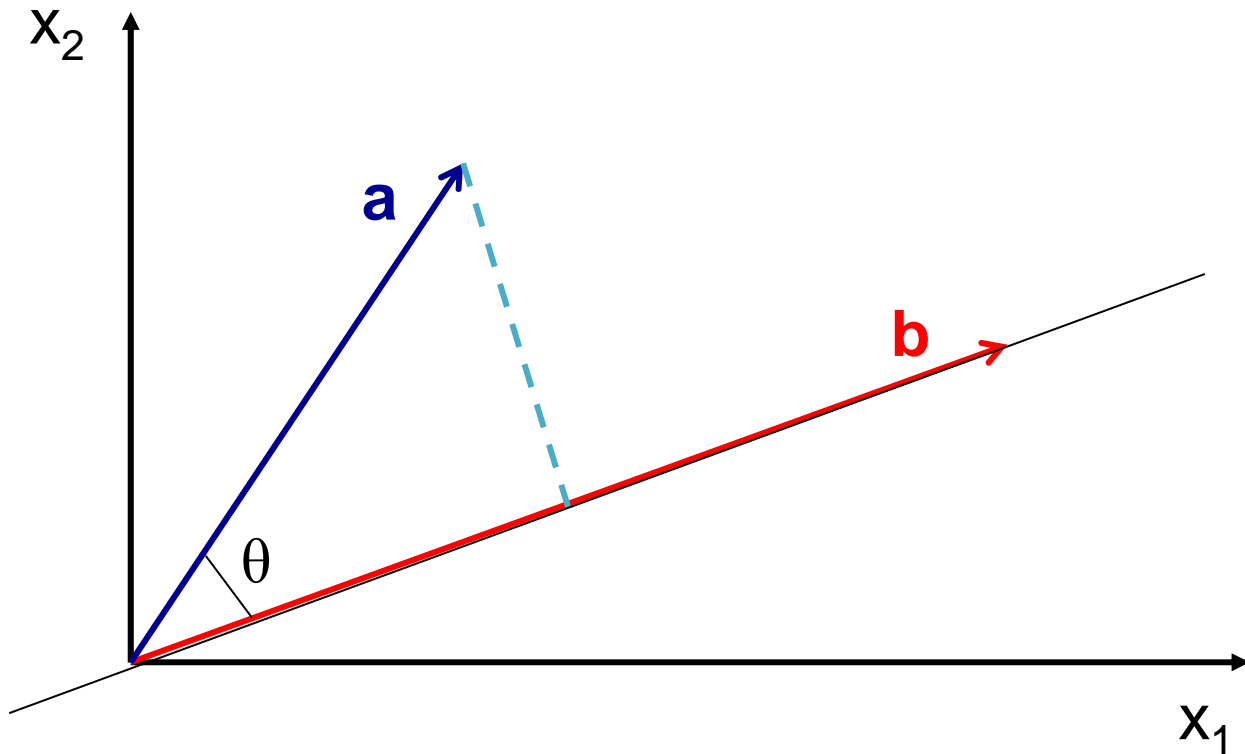
A bit of geometry

What is a scalar product?

- $s = \mathbf{a}^T \mathbf{b} = |\mathbf{a}| |\mathbf{b}| \cos(\theta)$
- if \mathbf{b} is a unit vector, $\mathbf{a}^T \mathbf{b}$ is the projection of \mathbf{a} on the line whose direction is governed by \mathbf{b}

A bit of geometry

What is the equation of the line governed by **b**?



A bit of geometry

What is the equation of the line governed by \mathbf{b} ?

Consider the vector $\mathbf{b}(b_1, b_2)$

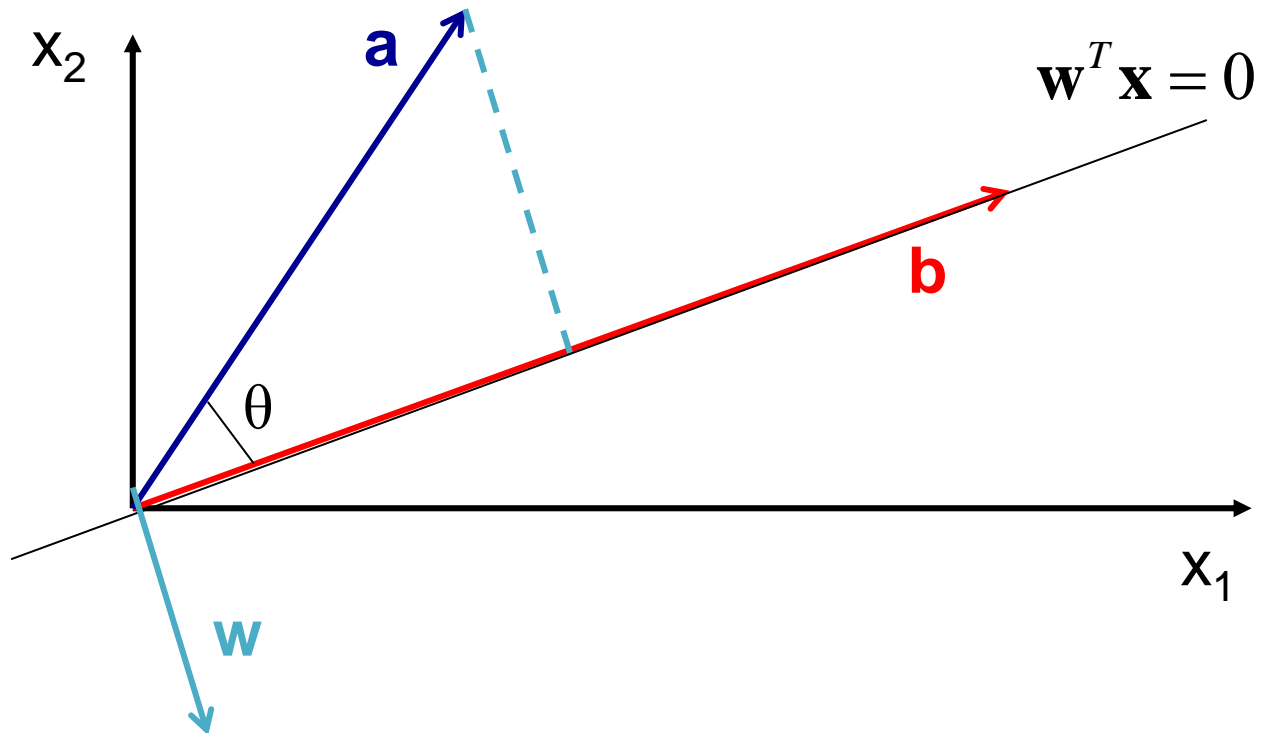
The line governed by \mathbf{b} has direction
$$\begin{cases} x_2 = \frac{b_2}{b_1} x_1 \\ b_2 x_1 - b_1 x_2 = 0 \end{cases}$$

so $\mathbf{w}^T \mathbf{x} = 0$ (with $\mathbf{w}(b_2, -b_1)$) is the equation of the line.

\mathbf{b} belongs to this line so $\mathbf{w}^T \mathbf{b} = 0$. Therefore $\mathbf{w} \perp \mathbf{b}$

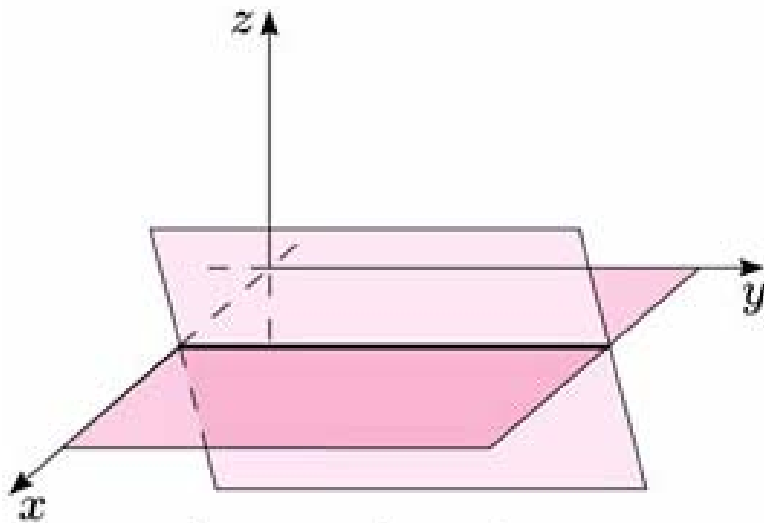
A bit of geometry

What is the equation of the line governed by **b**?

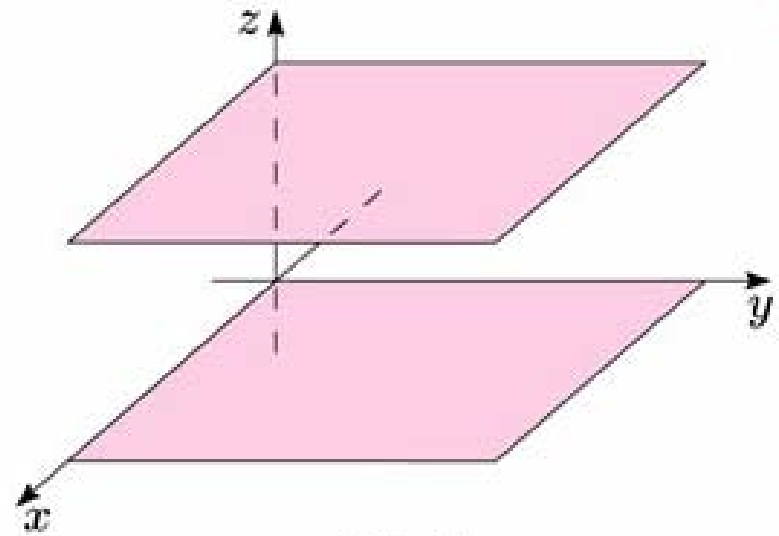


A bit of geometry

What is the intersection of 2 planes?



intersecting planes



parallel planes

Subplan

- *A bit of geometry*
- *Linear discriminant functions*
 - Regression versus classification
 - Least squares
 - Linear discriminant analysis
 - Support vector machines
- Summary

Supervised linear models

$t = y(\mathbf{x})$ with noise

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

f : activation function

predictions: $\hat{t} = y(\hat{\mathbf{x}})$

Linear classification

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

$t = y(\mathbf{x})$ with noise

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

f : activation function

predictions: $\hat{t} = y(\hat{\mathbf{x}})$

Linear classification

Decision boundary

$$y(\mathbf{x}) = \text{cst} \Leftrightarrow f(\mathbf{w}^T \mathbf{x} + w_0) = \text{cst}$$

$$\Leftrightarrow \mathbf{w}^T \mathbf{x} + w_0 = \text{cst}$$

$$\Leftrightarrow \mathbf{w}^T \mathbf{x} = \text{cst}$$

\Leftrightarrow decision surface is linear

Linear discriminant functions

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

$t = y(\mathbf{x})$ with noise

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

f : sign function

predictions: $\hat{t} = y(\hat{\mathbf{x}})$

Linear...

regression

$$t \in \mathbb{R}$$

$t = y(\mathbf{x})$ with noise

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

f : identity function

discriminant functions

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

$t = y(\mathbf{x})$ with noise

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

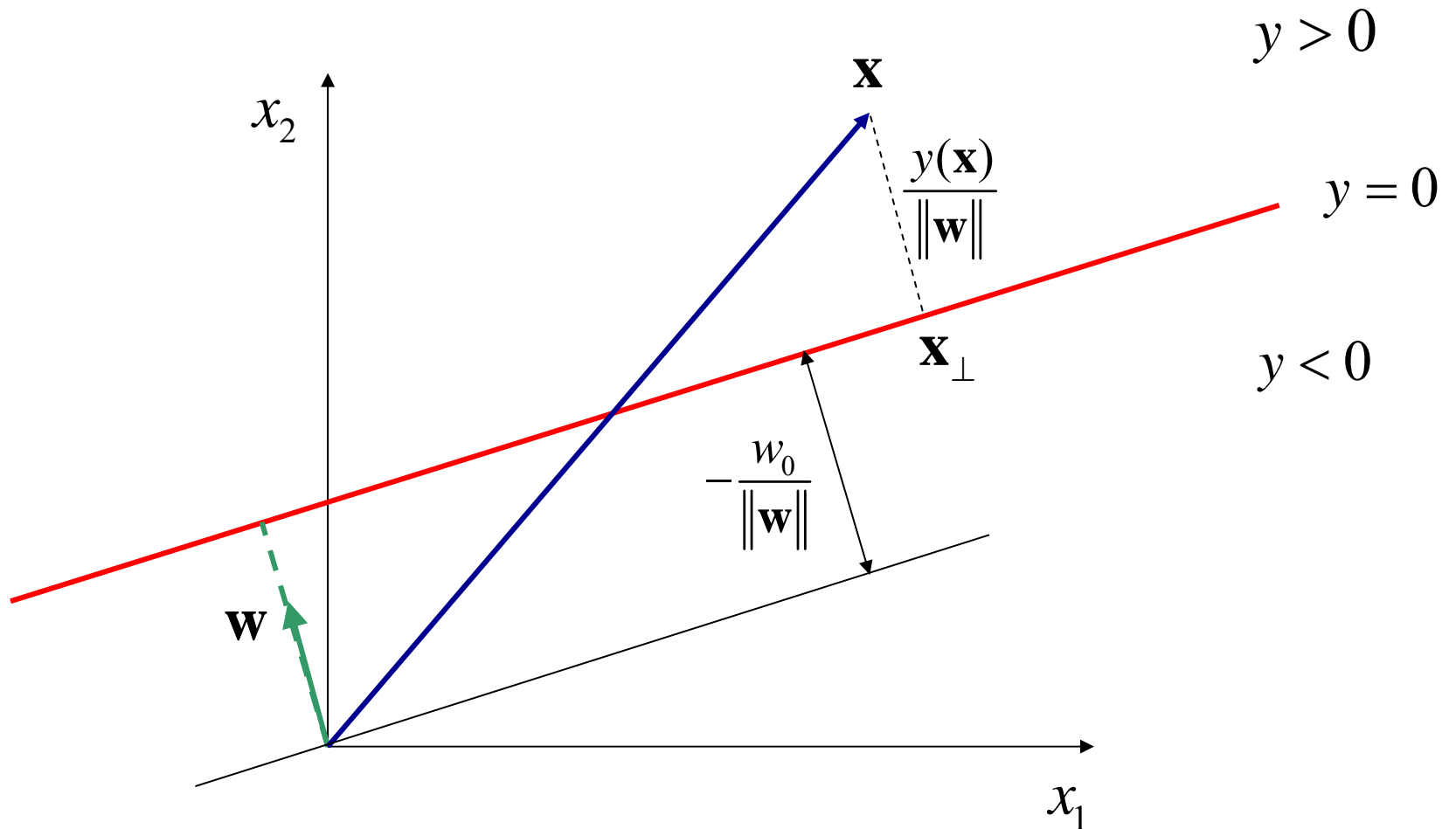
f : sign function

predictions: $\hat{t} = y(\hat{\mathbf{x}})$

Subplan

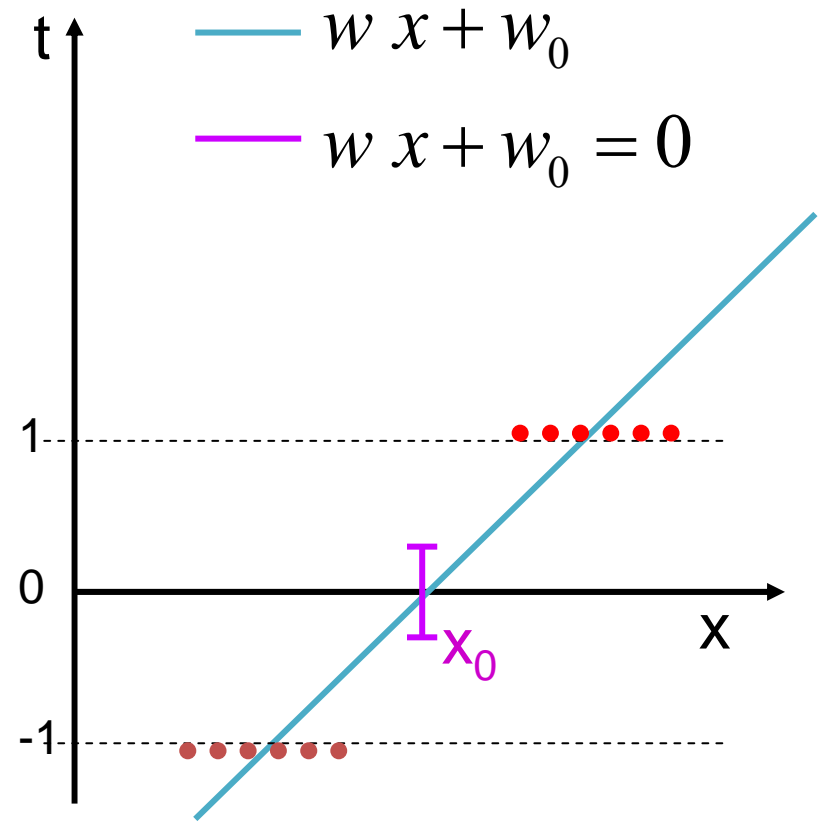
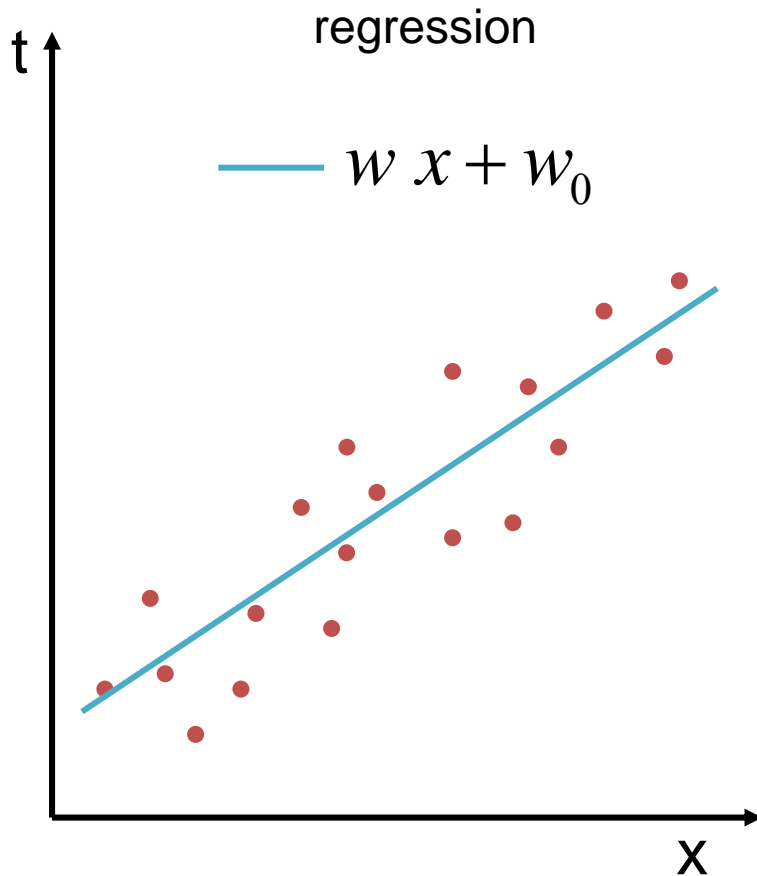
- *A bit of geometry*
- *Linear discriminant functions*
 - *Regression versus classification*
 - Least squares
 - Linear discriminant analysis
 - Support vector machines
- Summary

Linear discriminant functions



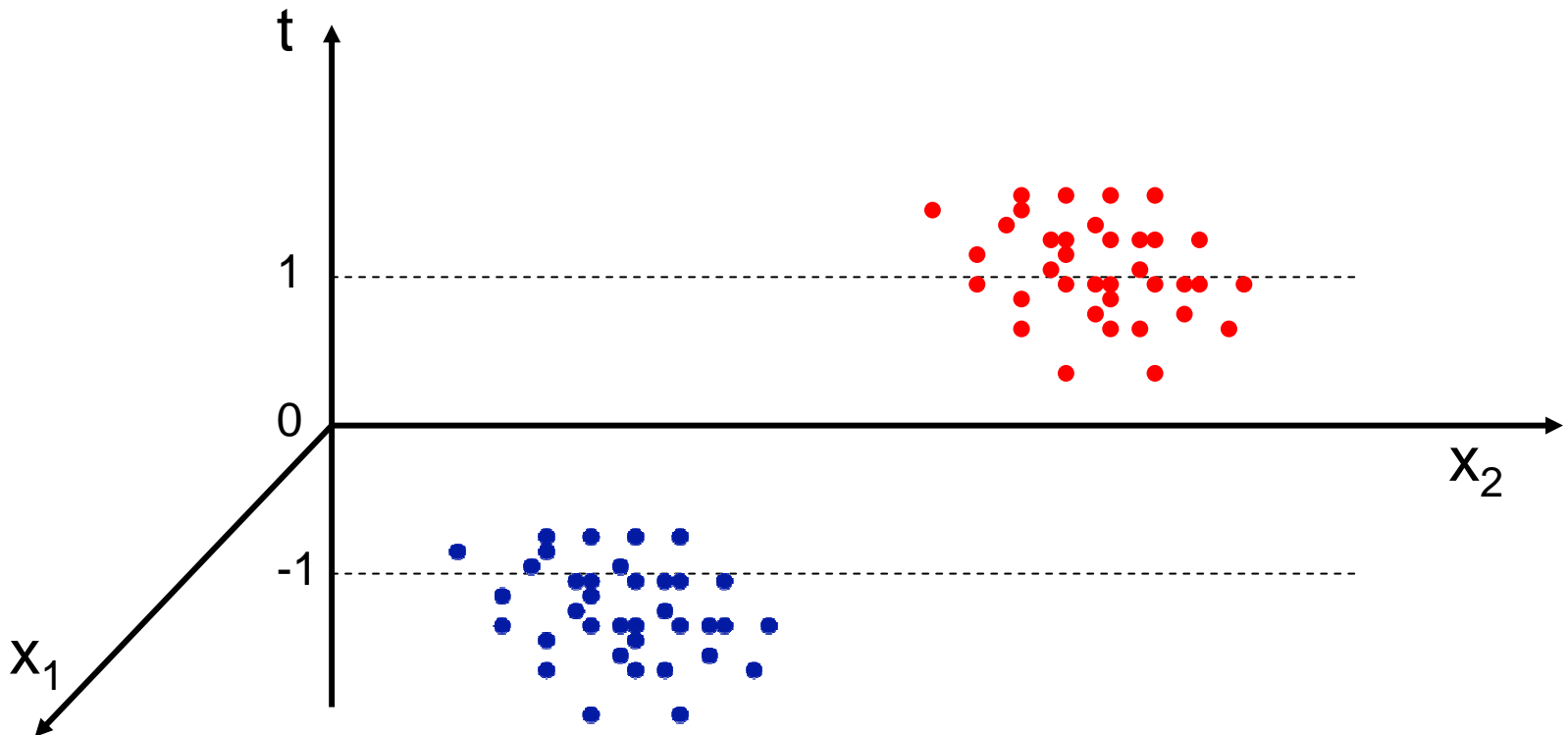
Regression versus classification

1 dimensional data



Regression versus classification

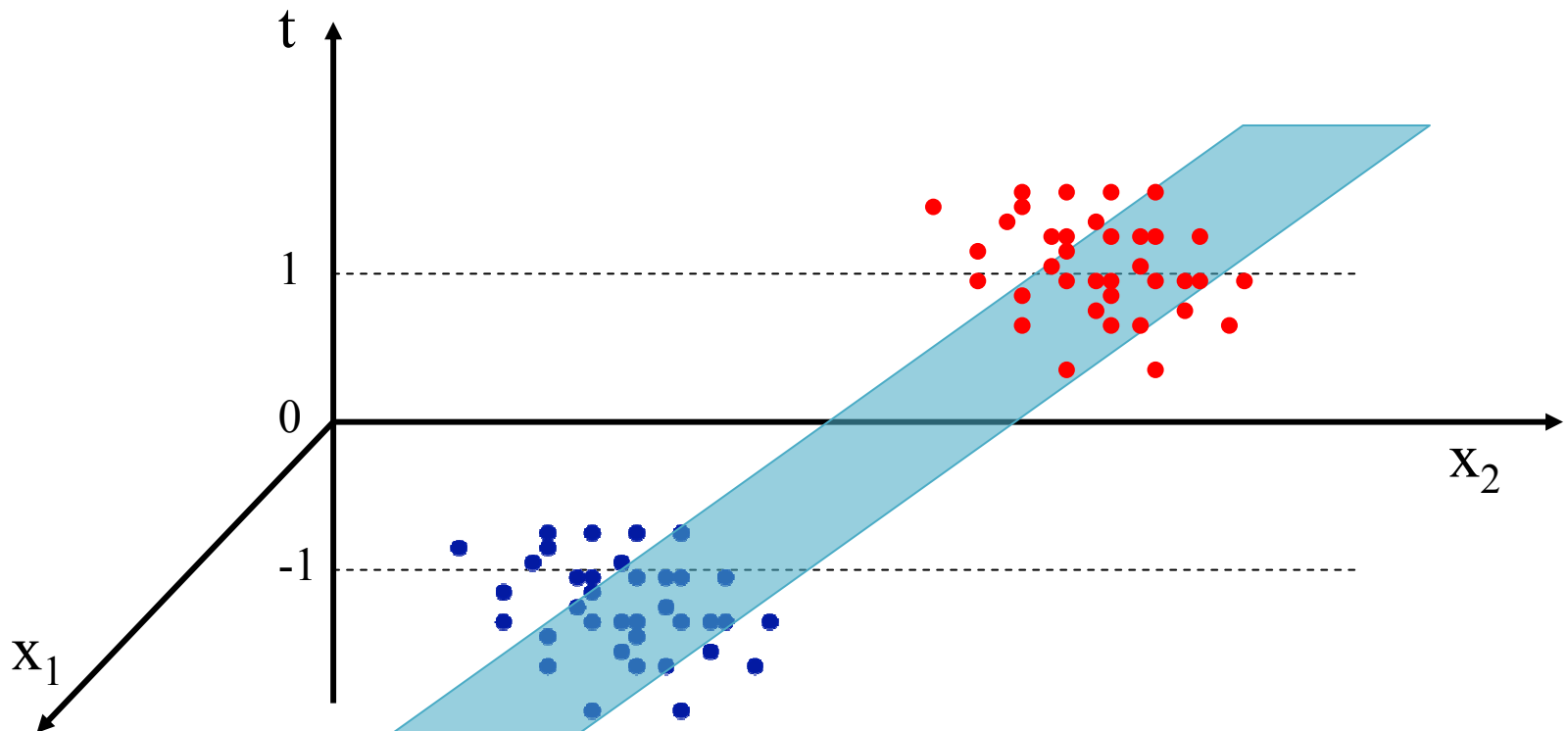
2 dimensional data



Regression versus classification

2 dimensional data

— $w_1 x_1 + w_2 x_2 + w_0$

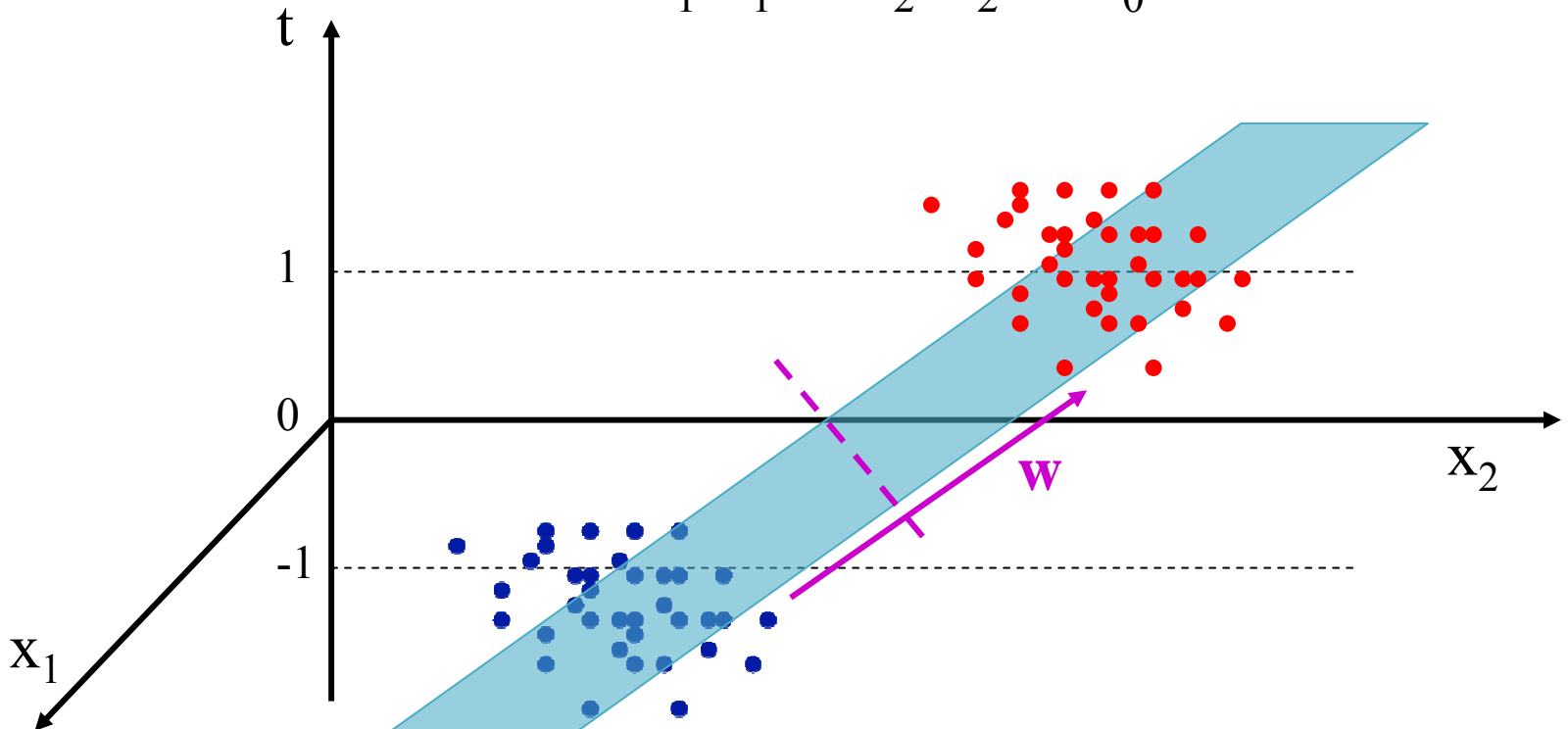


Regression versus classification

2 dimensional data

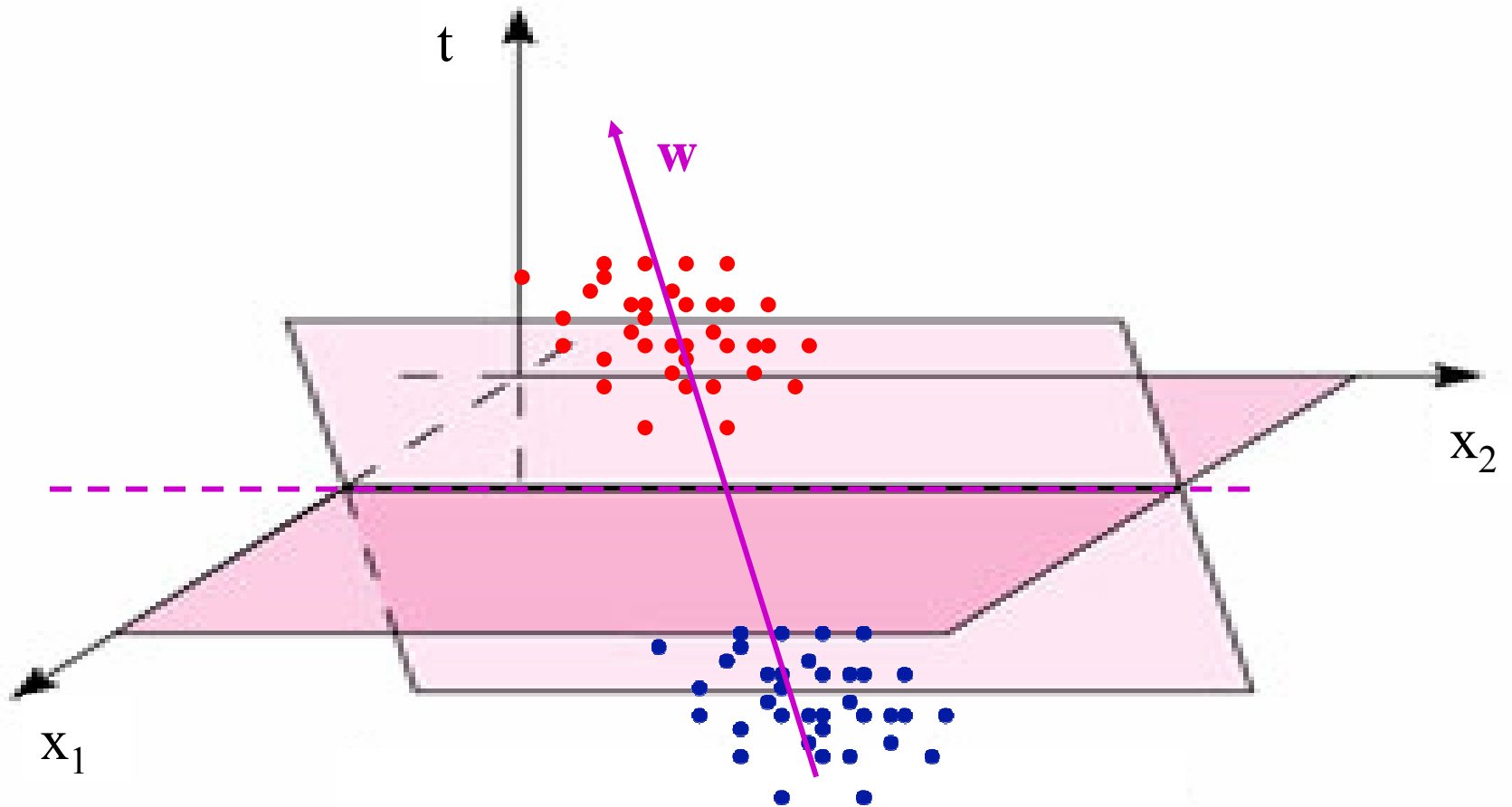
— $w_1 x_1 + w_2 x_2 + w_0$

- - - $w_1 x_1 + w_2 x_2 + w_0 = 0$



Regression versus classification

$$- - - w_1 x_1 + w_2 x_2 + w_0 = 0$$



Subplan

- *A bit of geometry*
- *Linear discriminant functions*
 - *Regression versus classification*
 - *Least squares*
 - Linear discriminant analysis
 - Support vector machines
- Summary

Least squares

Least squares

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

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N (t_n - y(\mathbf{x}))^2$$

Subplan

- *A bit of geometry*
- *Linear discriminant functions*
 - *Regression versus classification*
 - *Least squares*
 - *Linear discriminant analysis*
 - Support vector machines
- Summary

LDA

(linear discriminant analysis)

- $y(\mathbf{x}) = \text{sign}(\mathbf{w}^T \mathbf{x} + w_0)$
- Different point of view from least squares
 - projection on 1 dimension => loss of information
 - idea: find the \mathbf{w} that maximises the class separation

LDA

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

LDA

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

– maximize the class separation:

$$\max_{\mathbf{w}} \left(y(\mathbf{m}_2) - y(\mathbf{m}_1) \right) = \max_{\mathbf{w}} \left(\mathbf{w}^T (\mathbf{m}_2 - \mathbf{m}_1) \right)$$

LDA

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

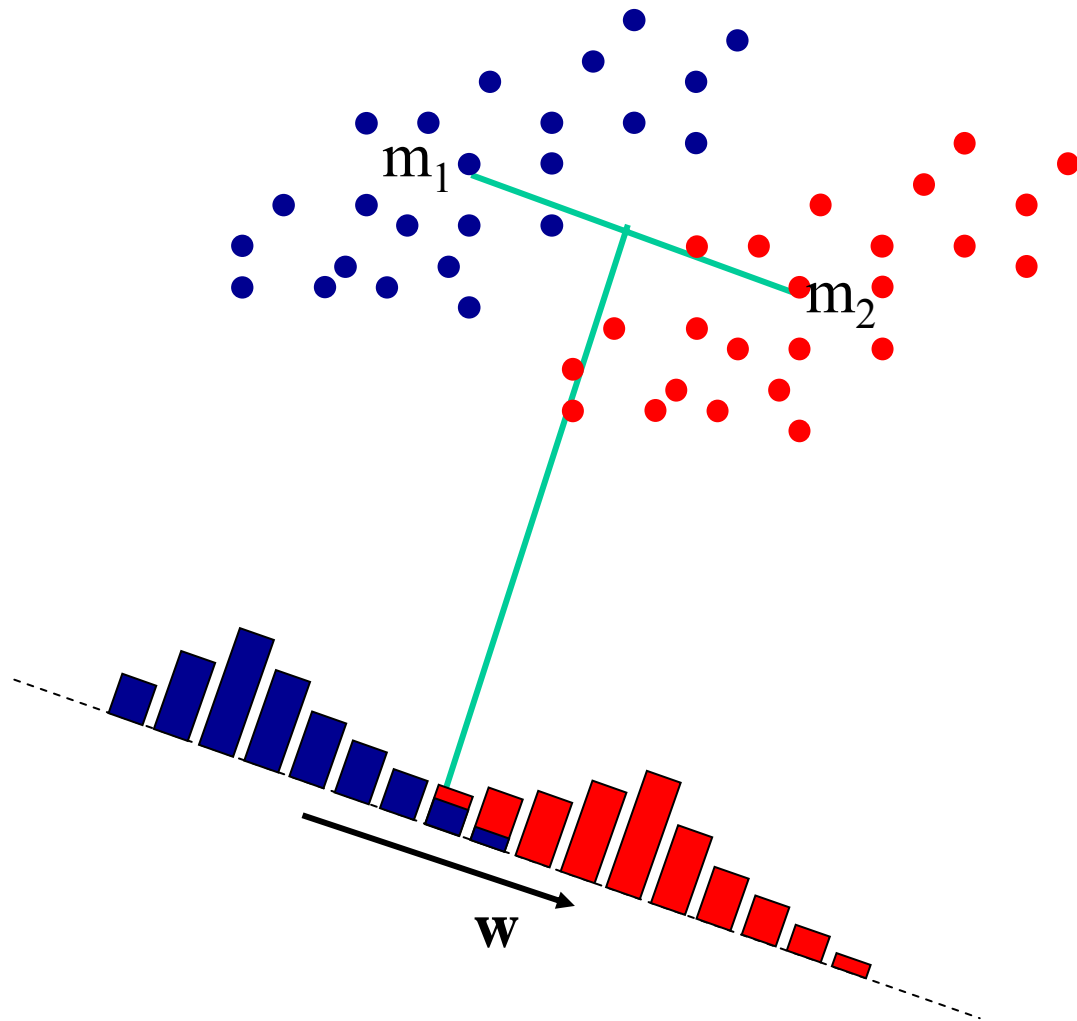
– maximize the class separation:

$$\max_{\mathbf{w}} \left(y(\mathbf{m}_2) - y(\mathbf{m}_1) \right) = \max_{\mathbf{w}} \left(\mathbf{w}^T (\mathbf{m}_2 - \mathbf{m}_1) \right)$$

– but \mathbf{w} can grow arbitrarily large:

$$\max_{\mathbf{w}} \left(\mathbf{w}^T (\mathbf{m}_2 - \mathbf{m}_1) \right) \quad \text{subject to} \quad \|\mathbf{w}\| = 1$$

LDA



LDA

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

– maximize the class separation:

$$\max_{\mathbf{w}} \left(y(\mathbf{m}_2) - y(\mathbf{m}_1) \right) = \max_{\mathbf{w}} \left(\mathbf{w}^T (\mathbf{m}_2 - \mathbf{m}_1) \right)$$

– but \mathbf{w} can grow arbitrarily large:

$$\max_{\mathbf{w}} \left(\mathbf{w}^T (\mathbf{m}_2 - \mathbf{m}_1) \right) \text{ subject to } \|\mathbf{w}\| = 1$$

– problem: does not take the covariance into account

LDA

Fisher's idea

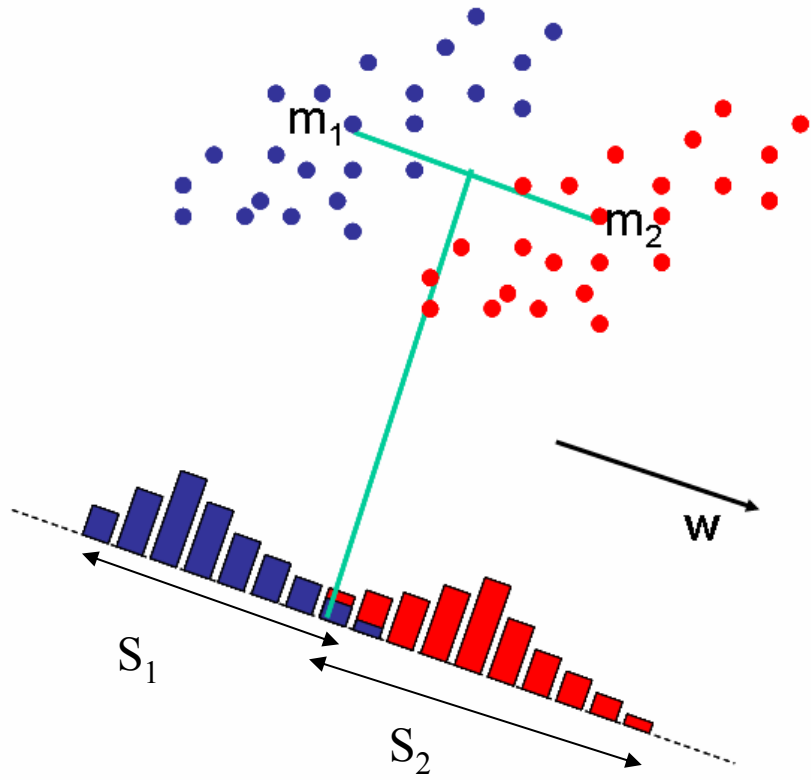
$$s_k^2 = \sum_{n \in \mathcal{C}_k} (y(\mathbf{x}_n) - y(\mathbf{m}_k))^2 \quad \text{within class variance}$$

we now want $\max_{\mathbf{w}} \frac{(\mathbf{w}^T (\mathbf{m}_2 - \mathbf{m}_1))^2}{s_1^2 + s_2^2}$ subject to $\|\mathbf{w}\| = 1$

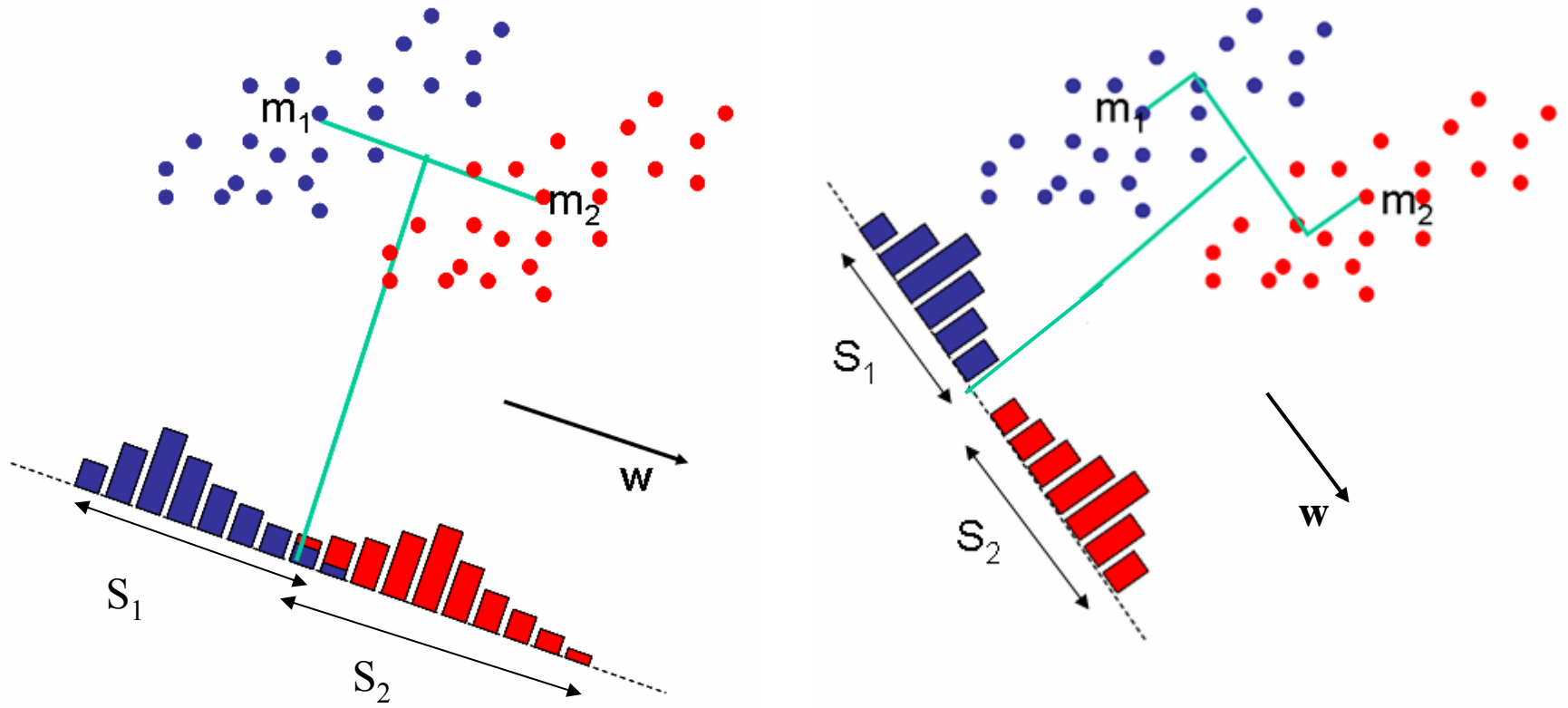
$$\Leftrightarrow \max_{\mathbf{w}} \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}} \quad \text{subject to } \|\mathbf{w}\| = 1 \quad \text{with} \quad \begin{cases} \mathbf{S}_W = \sum_k \sum_{n \in \mathcal{C}_k} (\mathbf{x}_n - \mathbf{m}_k)(\mathbf{x}_n - \mathbf{m}_k)^T \\ \mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T \end{cases}$$

As a result, $\mathbf{w} \propto \mathbf{S}_W^{-1} (\mathbf{m}_2 - \mathbf{m}_1)$

LDA



LDA



LDA

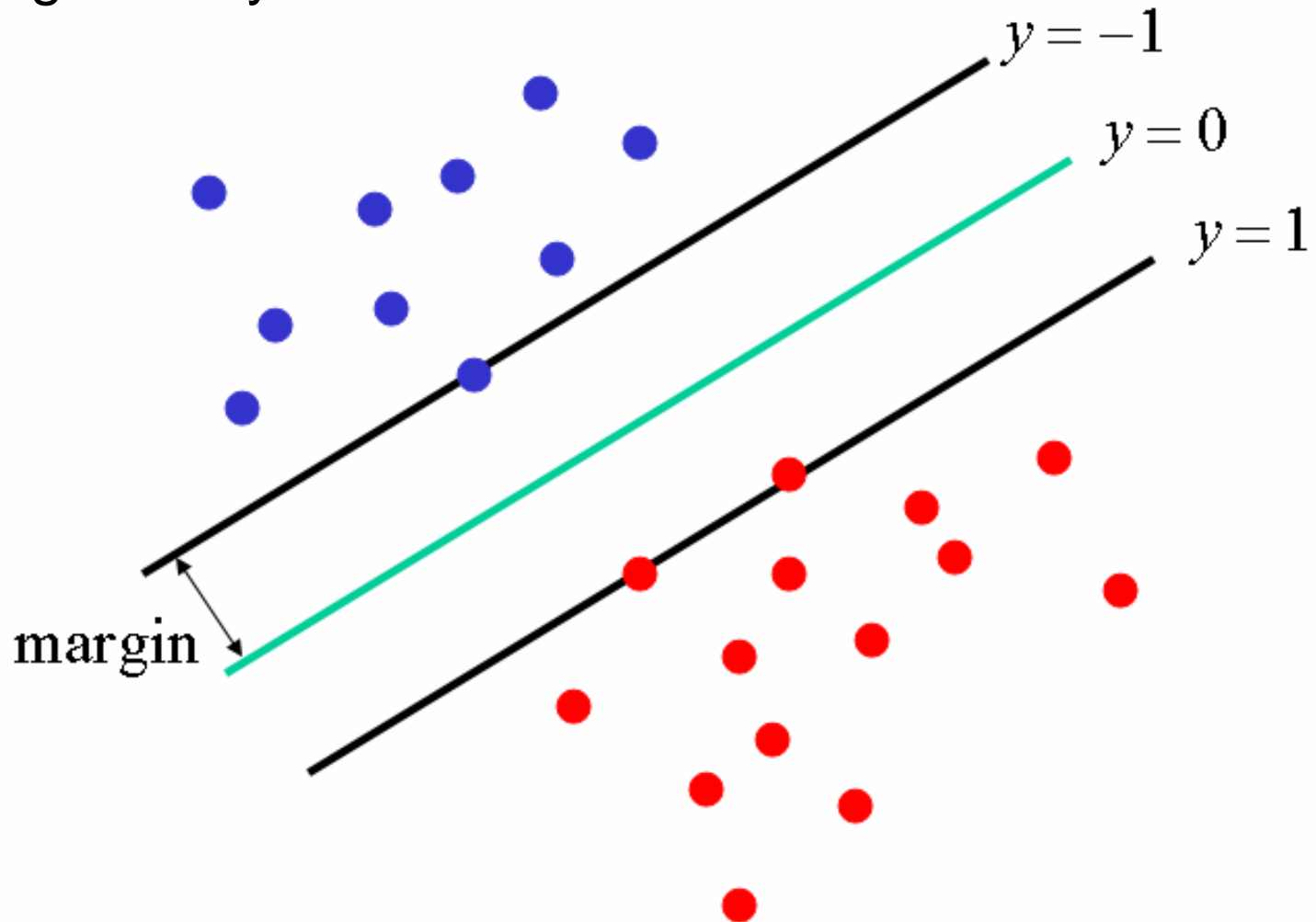
- Once \mathbf{w} is learnt, the projection scores $\mathbf{w}^T \mathbf{x}$ may be obtained for each datapoint
- The 2 classes are separated at b with $\mathbf{w}^T \mathbf{x} > b$ for class C_1 and $\mathbf{w}^T \mathbf{x} < b$ for class C_2
- $w_0 = -b$

Subplan

- *A bit of geometry*
- *Linear discriminant functions*
 - *Regression versus classification*
 - *Least squares*
 - *Linear discriminant analysis*
 - *Support vector machines*
- *Summary*

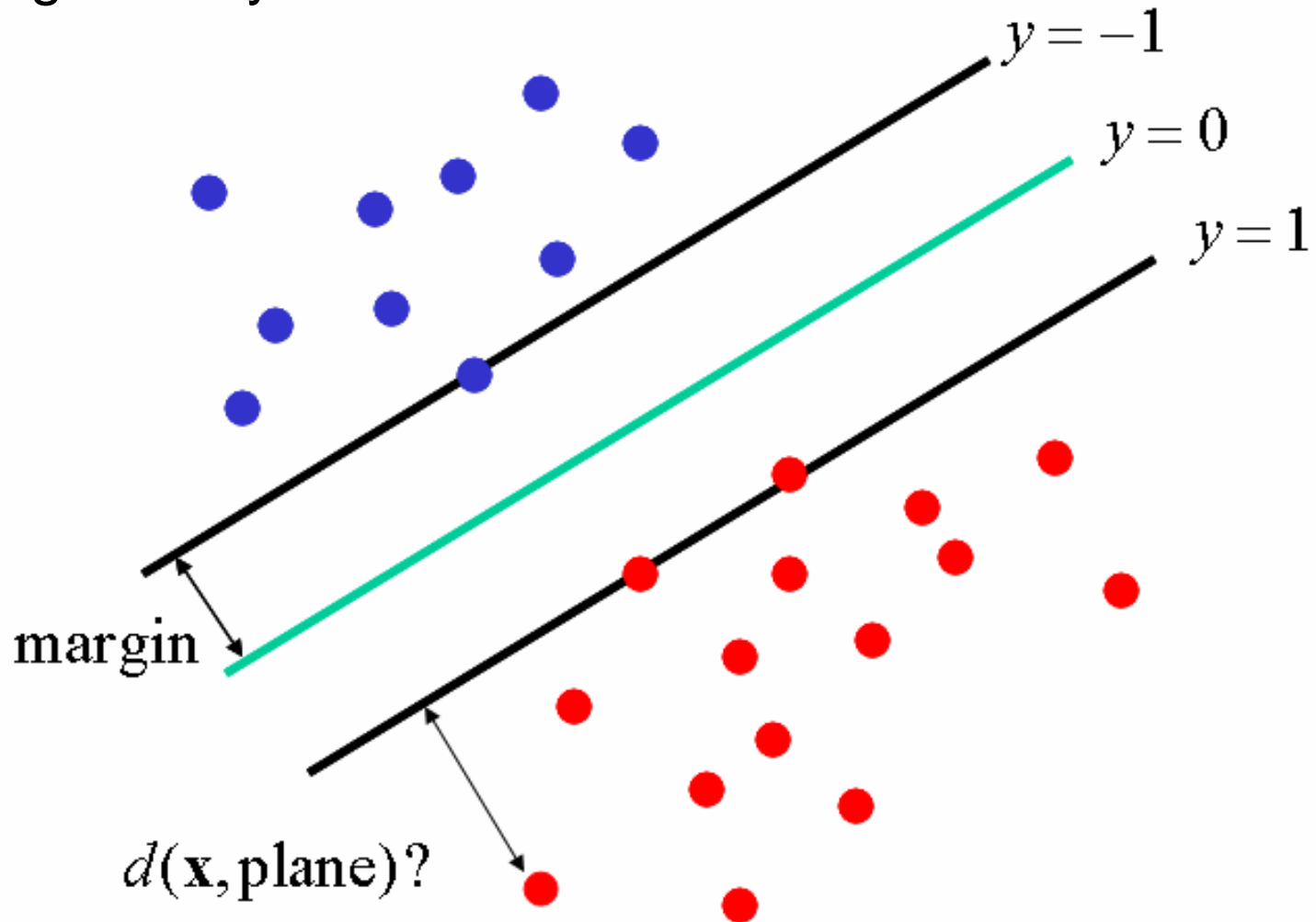
Support vector machines

Some geometry



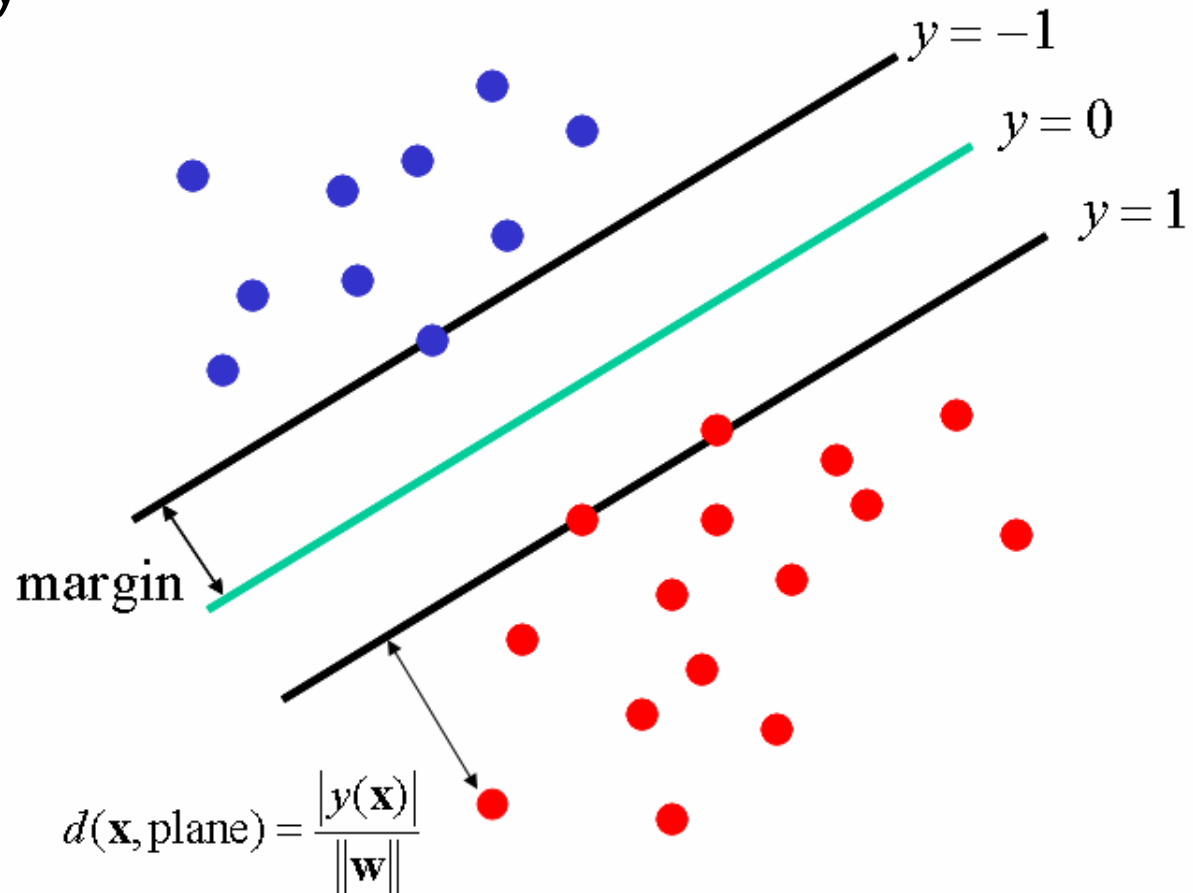
Support vector machines

Some geometry



Support vector machines

Some geometry



$$d(\mathbf{x}, \text{plane}) = \frac{1}{\|\mathbf{w}\|} t(\mathbf{w}^T \mathbf{x} + b) \quad \Rightarrow \quad d(\text{closest point}, \text{plane}) = \frac{1}{\|\mathbf{w}\|} \min_n t_n(\mathbf{w}^T \mathbf{x}_n + b)$$

Support vector machines

Proof

$$\mathbf{x} = \mathbf{x}_\perp + sd(\mathbf{x}, \text{plane}) \frac{\mathbf{w}}{\|\mathbf{w}\|}$$

$$\mathbf{w}^T \mathbf{x} = \mathbf{w}^T \mathbf{x}_\perp + sd(\mathbf{x}, \text{plane}) \frac{\mathbf{w}^T \mathbf{w}}{\|\mathbf{w}\|}$$

$$y(\mathbf{x}) - w_0 = \underbrace{y(\mathbf{x}_\perp)}_0 - w_0 + sd(\mathbf{x}, \text{plane}) \|\mathbf{w}\|$$

$$sd(\mathbf{x}, \text{plane}) = \frac{y(\mathbf{x})}{\|\mathbf{w}\|} \quad \text{so} \quad d(\mathbf{x}, \text{plane}) = \frac{|y(\mathbf{x})|}{\|\mathbf{w}\|}$$

Support vector machines

The error function

- maximise the margin

$$E(\mathbf{w}, w_0) = -\frac{1}{\|\mathbf{w}\|} \left(\min_n t_n (\mathbf{w}^T \mathbf{x}_n + w_0) \right)$$

- equivalent to

$$E(\mathbf{w}, w_0) = \frac{1}{2} \|\mathbf{w}\|^2 \quad \text{subject to} \quad t_n (\mathbf{w}^T \mathbf{x}_n + w_0) \geq 1$$

Subplan

- *A bit of geometry*
- *Linear discriminant functions*
 - *Regression versus classification*
 - *Least squares*
 - *Linear discriminant analysis*
 - *Support vector machines*
- *Summary*

Summary

Analytically

- linear separator $y(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0$
with $y(\mathbf{x}) < 0$ if $t = -1$ and $y(\mathbf{x}) > 0$ if $t = 1$
- what is learnt? \mathbf{w} and w_0
- predictions: $\hat{t} = \text{sign}(y(\hat{\mathbf{x}})) = \text{sign}(\mathbf{w}^T \hat{\mathbf{x}} + w_0)$

Summary

The error function

- Least squares – minimise the sum-of-squares error

$$E(\mathbf{w}, w_0) = \sum_{n=1}^N (t_n - \mathbf{w}^T \mathbf{x}_n - w_0)^2$$

- LDA – maximise the distance between classes and minimise the variance within classes

$$E(\mathbf{w}) = - \frac{(\mathbf{w}^T (\mathbf{m}_2 - \mathbf{m}_1))^2}{\sum_{n \in C_1} (\mathbf{w}^T (\mathbf{x}_n - \mathbf{m}_1))^2 + \sum_{n \in C_2} (\mathbf{w}^T (\mathbf{x}_n - \mathbf{m}_2))^2}$$

Summary

The error function

- Perceptron – minimise the number of misclassifications

$$E(\mathbf{w}, w_0) = -\sum_{n=1}^N t_n (\mathbf{w}^T \mathbf{x}_n + w_0)$$

- Support vector machines – maximise the margin

$$E(\mathbf{w}, w_0) = \frac{1}{2} \|\mathbf{w}\|^2 \quad \text{subject to} \quad t_n (\mathbf{w}^T \mathbf{x}_n + w_0) \geq 1$$

Plan

- *Linear regression*
 - *for actual regression*
 - *for classification => linear discriminant functions*
 - *probabilistic linear regression*
- Probabilistic supervised linear models
- Kernels

Supervised linear models

Linear regression... and
probability

Subplan

- The Gaussian distribution
- Probabilistic linear regression

Subplan

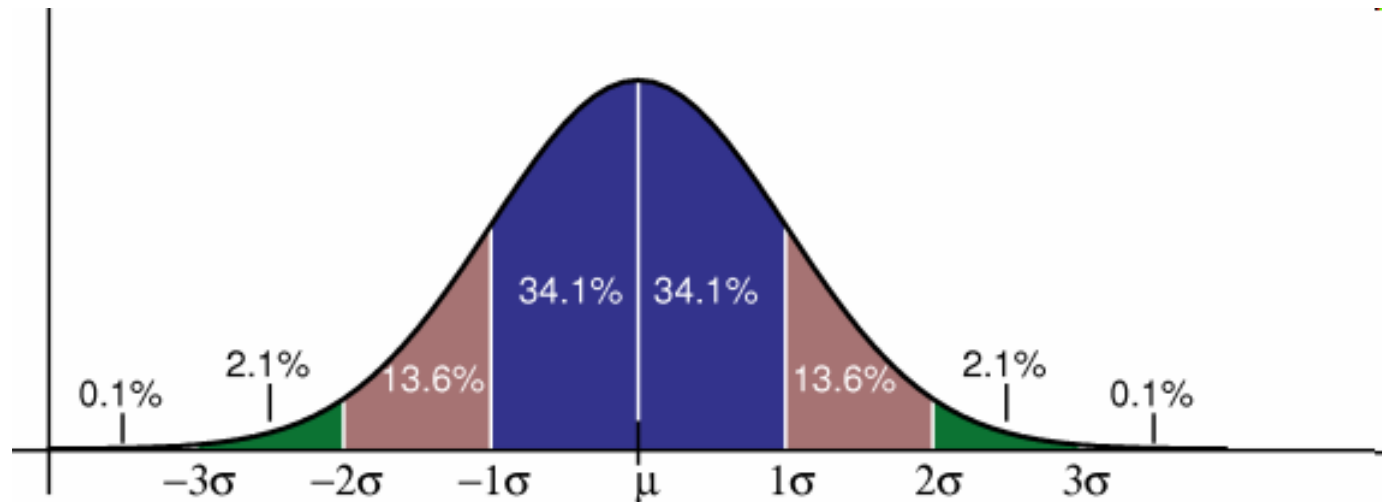
- *The Gaussian distribution*
- Probabilistic linear regression

The Gaussian distribution

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

$$\mathbb{E}_x[x] = \mu$$

$$\text{var}[x] = \sigma^2$$

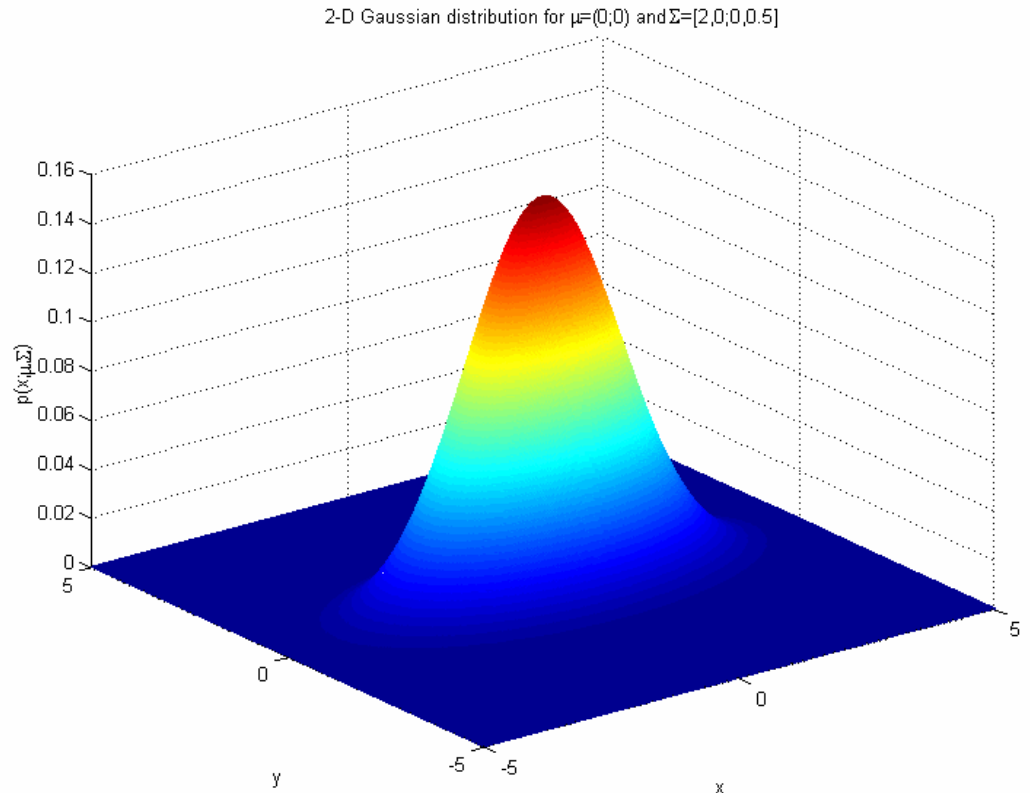


The Gaussian distribution

$$\mathcal{N}(\mathbf{x} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sqrt{\frac{1}{(2\pi)^D |\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right)$$

$$\mathbb{E}_{\mathbf{x}}[\mathbf{x}] = \boldsymbol{\mu}$$

$$\text{var}[\mathbf{x}] = \boldsymbol{\Sigma}$$



The Gaussian distribution

- We suppose our data $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ is independent and identically distributed (i.i.d)

=> then
$$p(\mathbf{X} | \mu, \Sigma) = \prod_{n=1}^N \mathcal{N}(\mathbf{x}_n | \mu, \Sigma)$$

- This is called the likelihood function

The Gaussian distribution

- Find the parameters that maximise the likelihood

$$L(\mu, \Sigma) = \log p(\mathbf{X} | \mu, \Sigma) = \sum_{n=1}^N \log \mathcal{N}(\mathbf{x}_n | \mu, \Sigma)$$

$$\frac{\partial L(\mu, \Sigma)}{\partial \mu} = 0 \quad \Rightarrow \quad \mu = \frac{1}{N} \sum_{n=1}^N \mathbf{x}_n$$

$$\frac{\partial L(\mu, \Sigma)}{\partial \Sigma} = 0 \quad \Rightarrow \quad \Sigma = \frac{1}{N} \sum_{n=1}^N (\mathbf{x}_n - \mu)(\mathbf{x}_n - \mu)^T$$

- Maximum likelihood solution for μ and Σ
=> denoted μ_{ML} and Σ_{ML}

Subplan

- *The Gaussian distribution*
- *Probabilistic linear regression*

Probabilistic linear regression

- Target = model function and random noise

$$t = y(x) \text{ with noise}$$

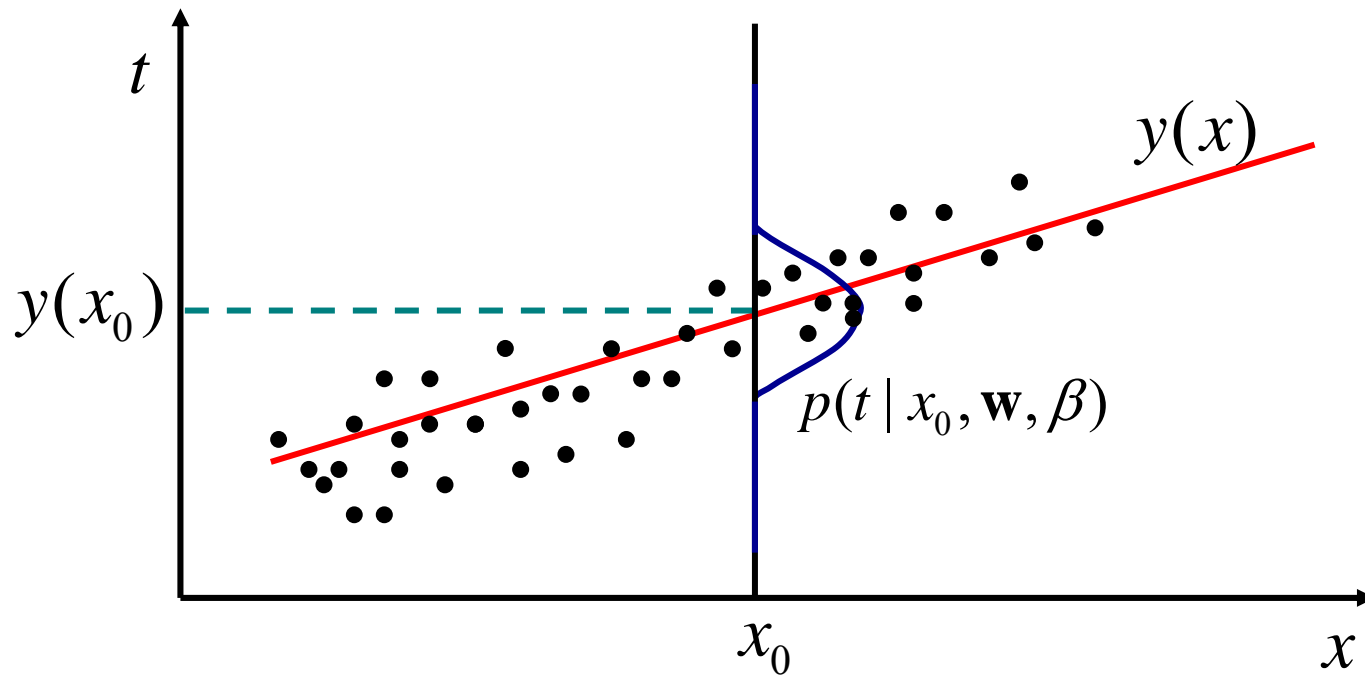
$$y(x) = w x + w_0$$

- Assume Gaussian noise

$$p(t | \mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t | y(\mathbf{x}), \beta^{-1})$$

Probabilistic linear regression

$$p(t | \mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t | y(\mathbf{x}), \beta^{-1})$$



Probabilistic linear regression

- Maximum likelihood training

$$p(\mathbf{t} | \mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^N \mathcal{N}(t_n | y(\mathbf{x}_n), \beta^{-1})$$

$$L(\mathbf{w}, \beta) = -\frac{\beta}{2} \sum_{n=1}^N (t_n - y(\mathbf{x}_n))^2 + \frac{N}{2} \log \beta - \frac{N}{2} \log(2\pi)$$

$$L(\mathbf{w}) \propto -\frac{1}{2} \sum_{n=1}^N (t_n - y(\mathbf{x}_n))^2 + \text{cst} \quad \text{so that } \mathbf{w}_{\text{ML}} = \mathbf{w}_{\text{LS}}$$

- Least squares $E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N (t_n - y(\mathbf{x}_n))^2$
- Maximum likelihood is equivalent to least squares!!!

Probabilistic linear regression

- Maximum likelihood training

$$p(\mathbf{t} | \mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^N \mathcal{N}(t_n | y(\mathbf{x}_n), \beta^{-1})$$

$$L(\mathbf{w}, \beta) = -\frac{\beta}{2} \sum_{n=1}^N (t_n - y(\mathbf{x}_n))^2 + \frac{N}{2} \log \beta - \frac{N}{2} \log(2\pi)$$

$$\beta_{ML}^{-1} = \frac{1}{N} \sum_{n=1}^N (t_n - y(\mathbf{x}_n))^2 \quad \text{so that } \beta_{ML}^{-1} \text{ is the SSE}$$

- The variance is the sum-of-squares-error

Probabilistic linear regression

Consequences

- least squares assumes the targets to be – conditioned on the inputs – normally distributed
- for classification it is not at all the case!
- the distribution of the targets needs to be modelled properly to find the proper error function
- you need to understand what you are dealing with

Plan

- *Linear regression*
 - *for actual regression*
 - *for classification => linear discriminant functions*
 - *probabilistic linear regression*
- *Probabilistic supervised linear models*
- **Kernels**

Probabilistic supervised linear models

Subplan

- Linear regression
- Logistic regression
- Poisson regression

Probabilistic supervised linear models

$$E[t | \mathbf{x}] = y(\mathbf{x})$$

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

f : activation function

predictions: $E[\hat{t} | \hat{\mathbf{x}}] = y(\hat{\mathbf{x}})$

Subplan

- *Linear regression*
- Logistic regression
- Poisson regression

Linear regression

- The targets are assumed to be real
- We suppose that they follow a conditional Gaussian distribution whereby $p(t | \mathbf{x}) = \mathcal{N}(t | \mu, \beta^{-1})$
- $E[t | \mathbf{x}] = \mu$ so in fact we need $p(t | \mathbf{x}) = \mathcal{N}(t | y(\mathbf{x}), \beta^{-1})$
 - => $y(\mathbf{x})$ represents the prediction and must approximate t
 - => the activation function must be thought of carefully
- The identity function is used

Linear regression

$$t \in \mathbb{R}$$

$$E[t \mid \mathbf{x}] = y(\mathbf{x})$$

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

f : identity function

predictions: $E[\hat{t} \mid \hat{\mathbf{x}}] = y(\hat{\mathbf{x}})$

Linear regression

- Maximum likelihood training

$$p(\mathbf{t} | \mathbf{X}, \mathbf{w}) = \prod_{n=1}^N \mathcal{N}(t_n | y(\mathbf{x}_n), \beta^{-1})$$

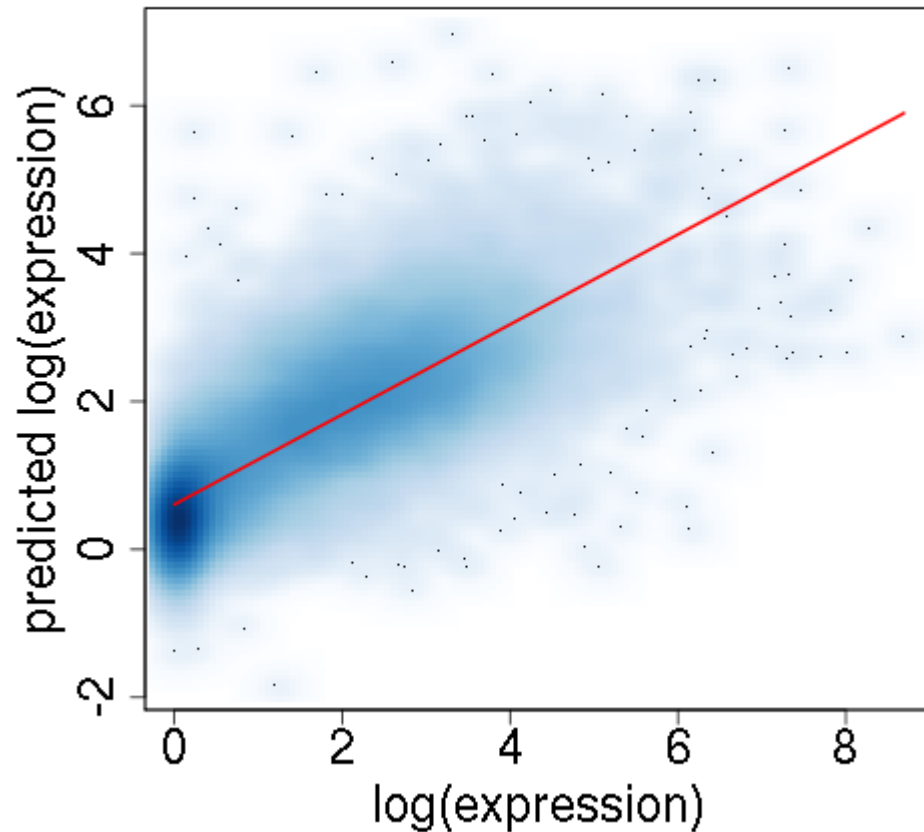
$$L(\mathbf{w}) = -\frac{\beta}{2} \sum_{n=1}^N (t_n - y(\mathbf{x}_n))^2$$

- Corresponds to least squares

Linear regression

Predicting Refseq genes' expression using histone modifications

COR = 0.78



Subplan

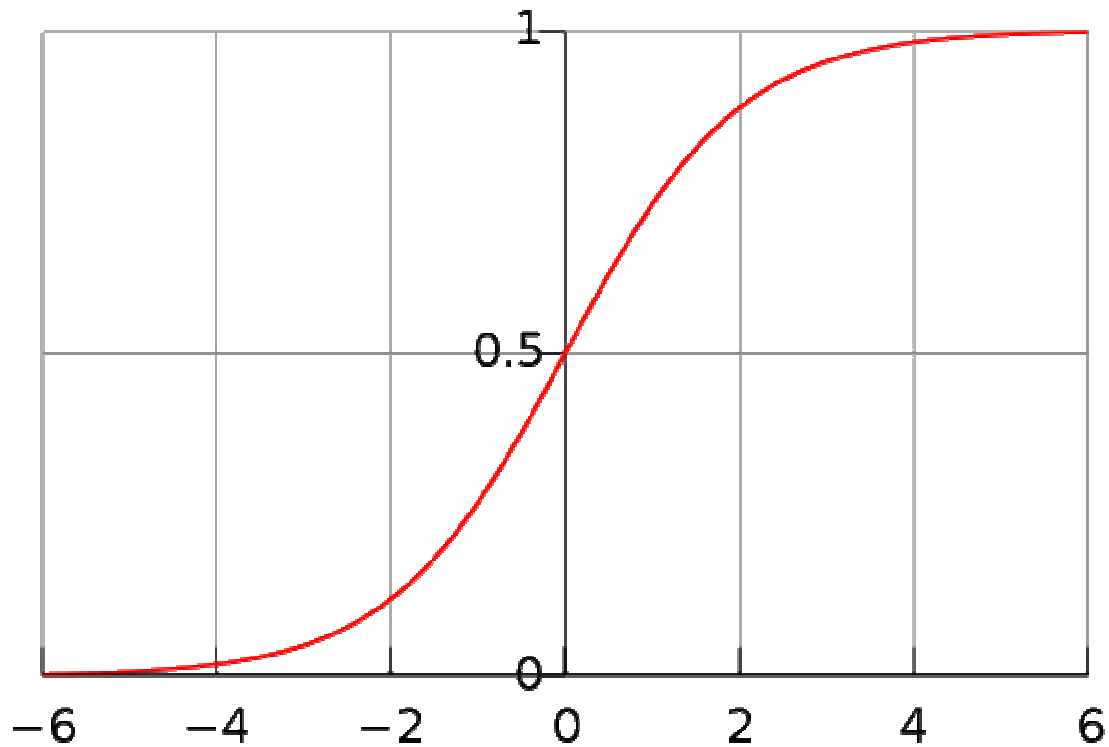
- *Linear regression*
- *Logistic regression*
- Poisson regression

Logistic regression

- The targets are assumed to belong to $\{1,0\}$
- We suppose that they follow a conditional Bernoulli distribution whereby $p(t | \mathbf{x}) = \mathcal{B}(t | q) = q^t (1 - q)^{1-t}$
- $E[t = 1 | \mathbf{x}] = q$ so in fact we need $p(t | \mathbf{x}) = \mathcal{B}(t | y(\mathbf{x}))$
 - => $y(\mathbf{x})$ represents a probability and must belong to $[0,1]$
 - => the activation function must be thought of carefully

Logistic regression

Sigmoid function



Logistic regression

- The targets are assumed to belong to $\{1,0\}$
- We suppose that they follow a conditional Bernoulli distribution whereby $p(t | \mathbf{x}) = \mathcal{B}(t | q) = q^t (1 - q)^{1-t}$
- $E[t = 1 | \mathbf{x}] = q$ so in fact we need $p(t | \mathbf{x}) = \mathcal{B}(t | y(\mathbf{x}))$
 - => $y(\mathbf{x})$ represents a probability and must belong to $[0,1]$
 - => the activation function must be thought of carefully
- The sigmoid function is used

Logistic regression

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

$$E[t \mid \mathbf{x}] = y(\mathbf{x})$$

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

f : sigmoid function

predictions: $E[\hat{t} \mid \hat{\mathbf{x}}] = y(\hat{\mathbf{x}})$

Probabilistic supervised linear models

Gaussian targets

$$t \in \mathbb{R}$$

$$E[t | \mathbf{x}] = y(\mathbf{x})$$

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

f : identity function

Bernoulli targets

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

$$E[t | \mathbf{x}] = y(\mathbf{x})$$

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

f : sigmoid function

predictions: $E[\hat{t} | \hat{\mathbf{x}}] = y(\hat{\mathbf{x}})$

Logistic regression

- Can we use least squares then?
 - the problem is so similar, it seems like a good idea
 - corresponds to maximum likelihood when the targets are – conditioned on the inputs – normally distributed
 - in classification, the targets are 1 or 0, they are definitely not normally distributed!!!
 - no least squares, but maximum likelihood

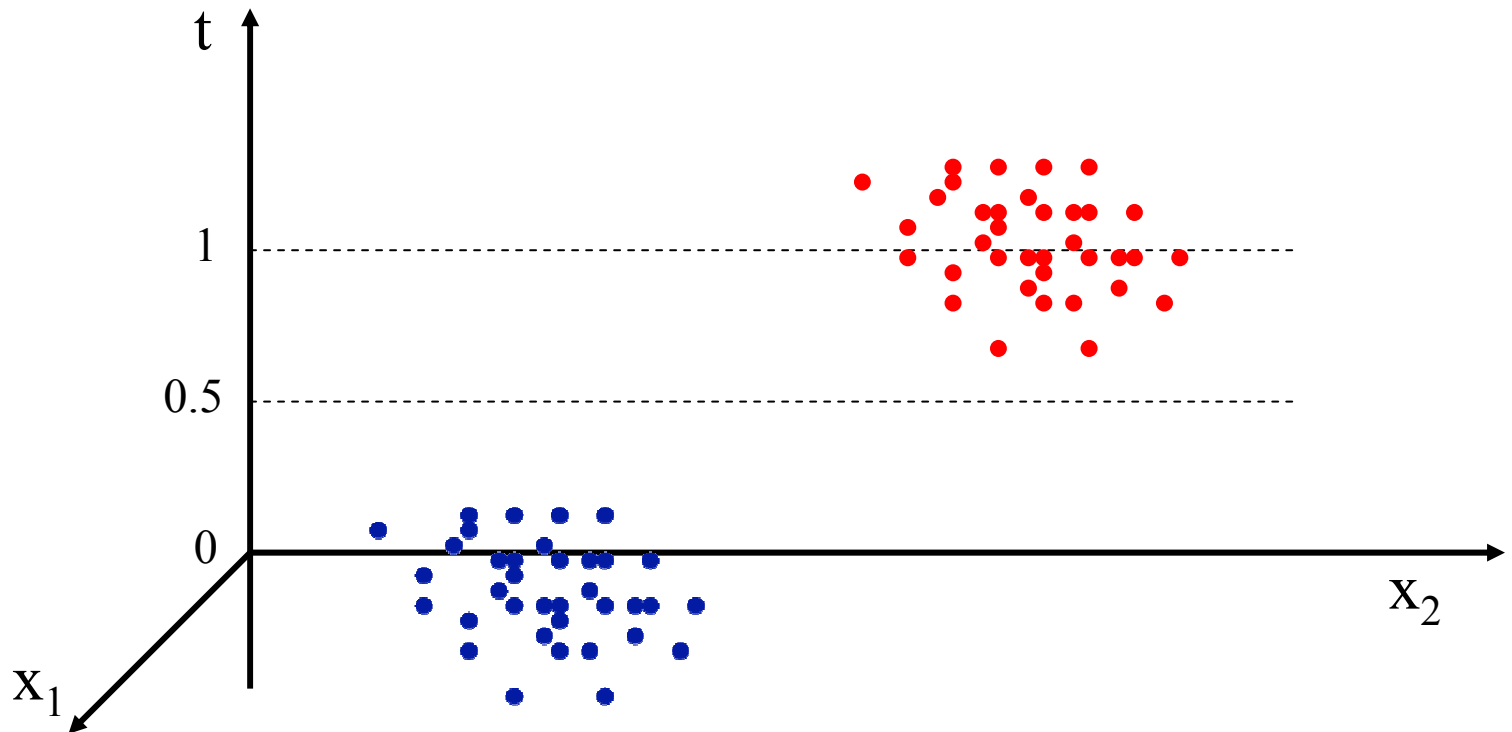
Logistic regression

Maximum likelihood training

$$p(\mathbf{t} | \mathbf{X}, \mathbf{w}) = \prod_{n=1}^N \mathcal{B}(t_n | y(\mathbf{x}_n))$$

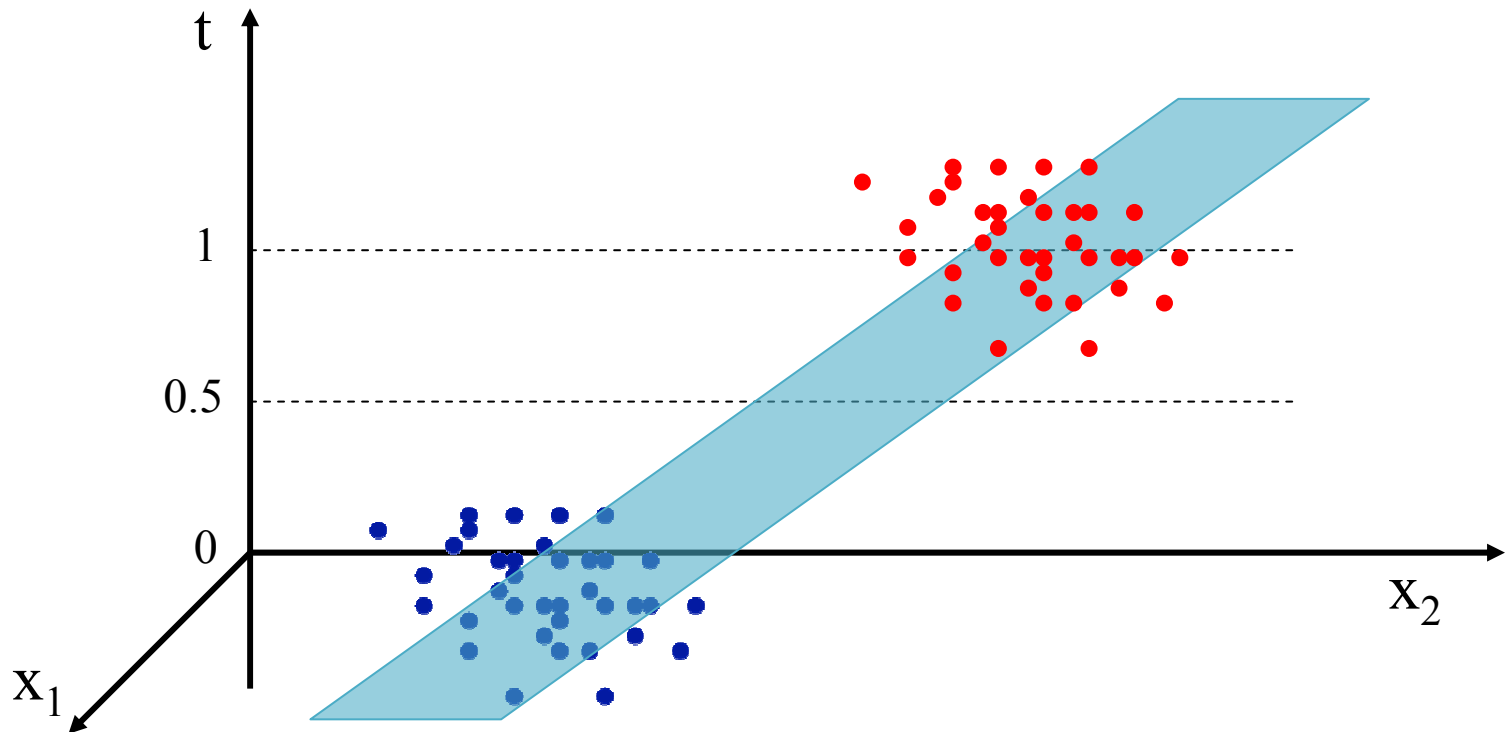
$$L(\mathbf{w}) = \sum_{n=1}^N [t_n \log y(\mathbf{x}_n) + (1 - t_n) \log(1 - y(\mathbf{x}_n))]$$

Logistic regression



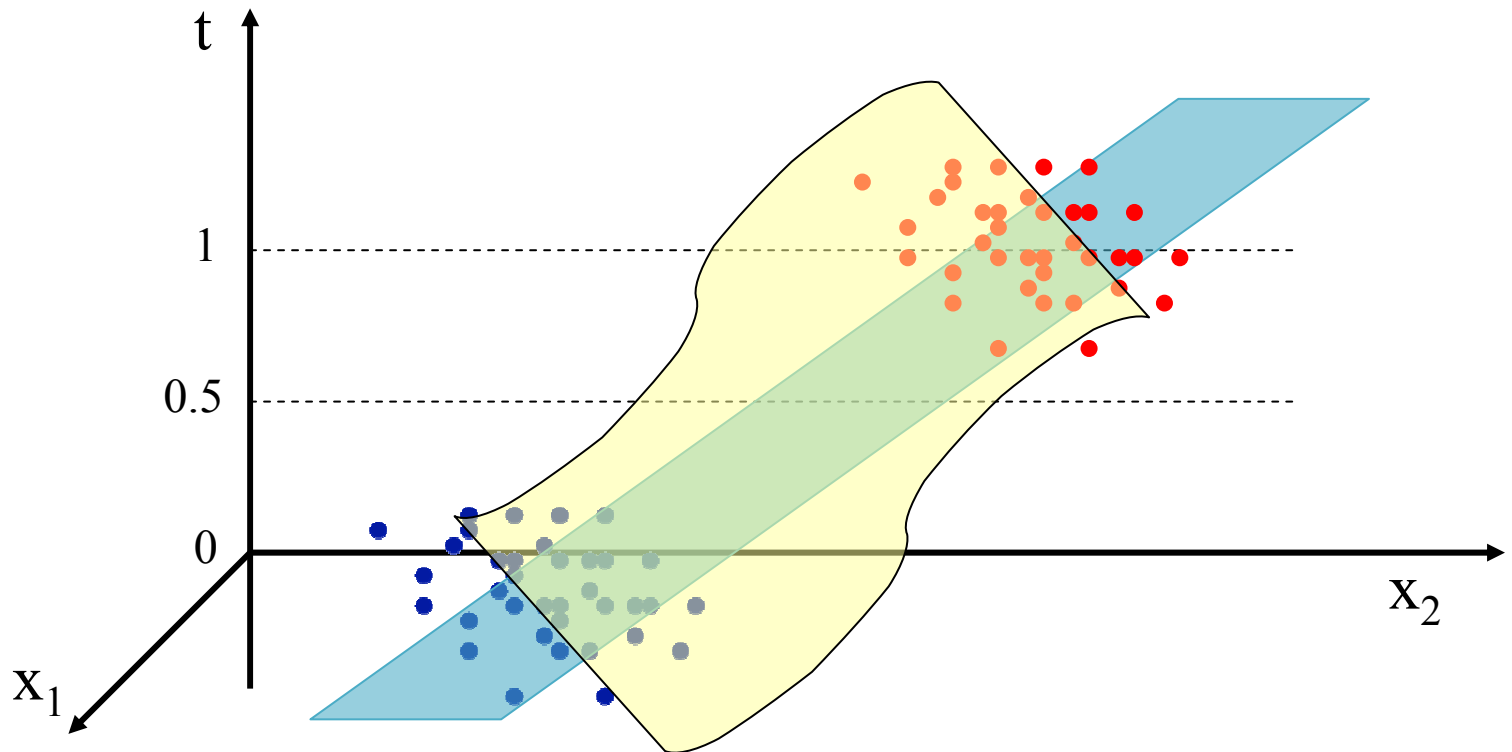
Logistic regression

— $w_1 x_1 + w_2 x_2 + w_0$



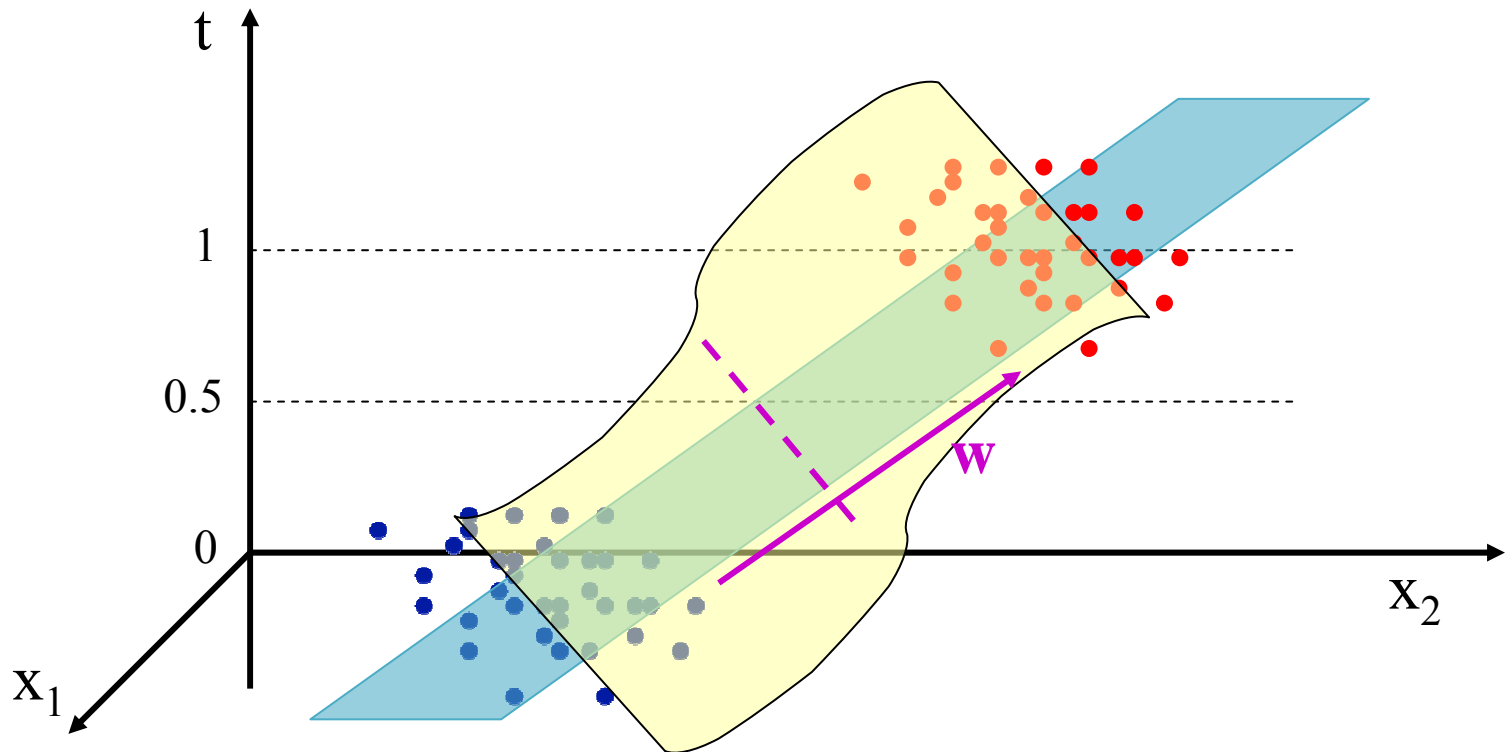
Logistic regression

— $w_1 x_1 + w_2 x_2 + w_0$
■ $y(\mathbf{x}) = \sigma(w_1 x_1 + w_2 x_2 + w_0)$



Logistic regression

- $w_1 x_1 + w_2 x_2 + w_0$
- $y(\mathbf{x}) = \sigma(w_1 x_1 + w_2 x_2 + w_0)$
- - - $w_1 x_1 + w_2 x_2 + w_0 = 0.5$



Subplan

- *Linear regression*
- *Logistic regression*
- *Poisson regression*

Poisson regression

- Count data: the targets are assumed to belong to $[0, +\infty[$
- We suppose that they follow a conditional Poisson distribution whereby $p(t | \mathbf{x}) = \mathcal{P}(t | \lambda) = \frac{\lambda^t}{t!} \exp(-\lambda)$
- $E[t | \mathbf{x}] = \lambda$ so in fact we need $p(t | \mathbf{x}) = \mathcal{P}(t | y(\mathbf{x}))$
 - => $y(\mathbf{x})$ represents the prediction, and must approximate t
 - => t is always positive
 - => the activation function must be thought of carefully
- The exponential function is used

Poisson regression

$$t \in \mathbb{N}^+$$

$$E[t \mid \mathbf{x}] = y(\mathbf{x})$$

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

f : exponential function

predictions: $E[\hat{t} \mid \hat{\mathbf{x}}] = y(\hat{\mathbf{x}})$

Probabilistic supervised linear models

Gaussian targets

$$t \in \mathbb{R}$$

$$E[t | \mathbf{x}] = y(\mathbf{x})$$

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

f : identity function

Bernoulli targets

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

$$E[t | \mathbf{x}] = y(\mathbf{x})$$

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

f : sigmoid function

Poisson targets

$$t \in \mathbb{N}^+$$

$$E[t | \mathbf{x}] = y(\mathbf{x})$$

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

f : exponential function

$$\text{predictions: } E[\hat{t} | \hat{\mathbf{x}}] = y(\hat{\mathbf{x}})$$

Poisson regression

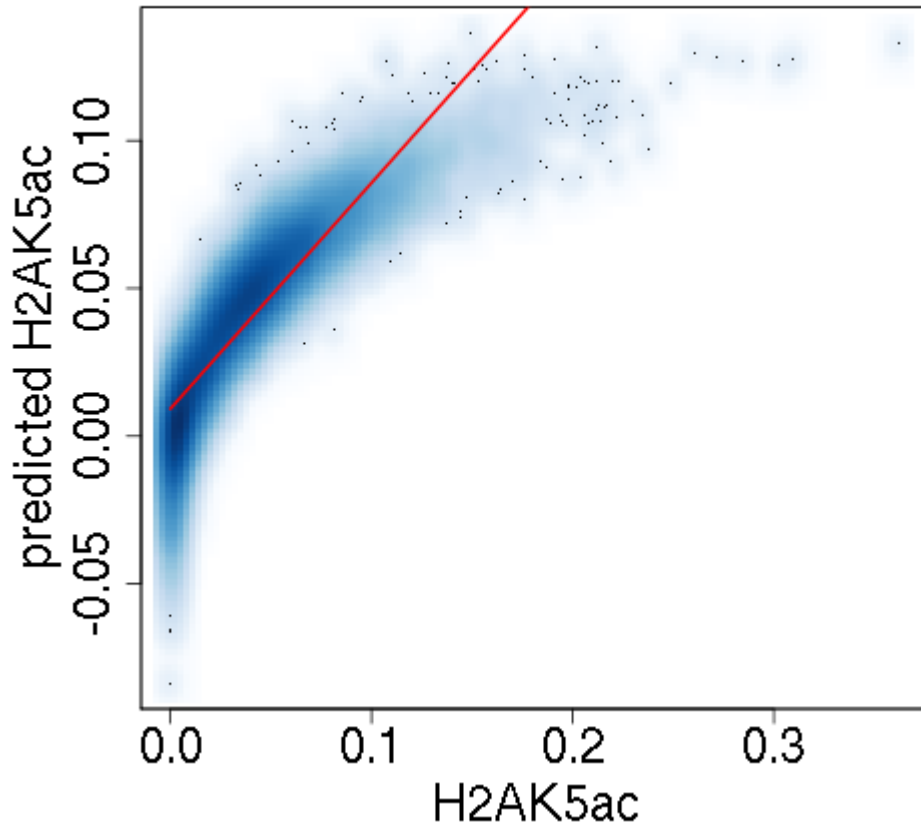
Maximum likelihood training

$$p(\mathbf{t} | \mathbf{X}, \mathbf{w}) = \prod_{n=1}^N \mathcal{P}(t_n | y(\mathbf{x}_n))$$

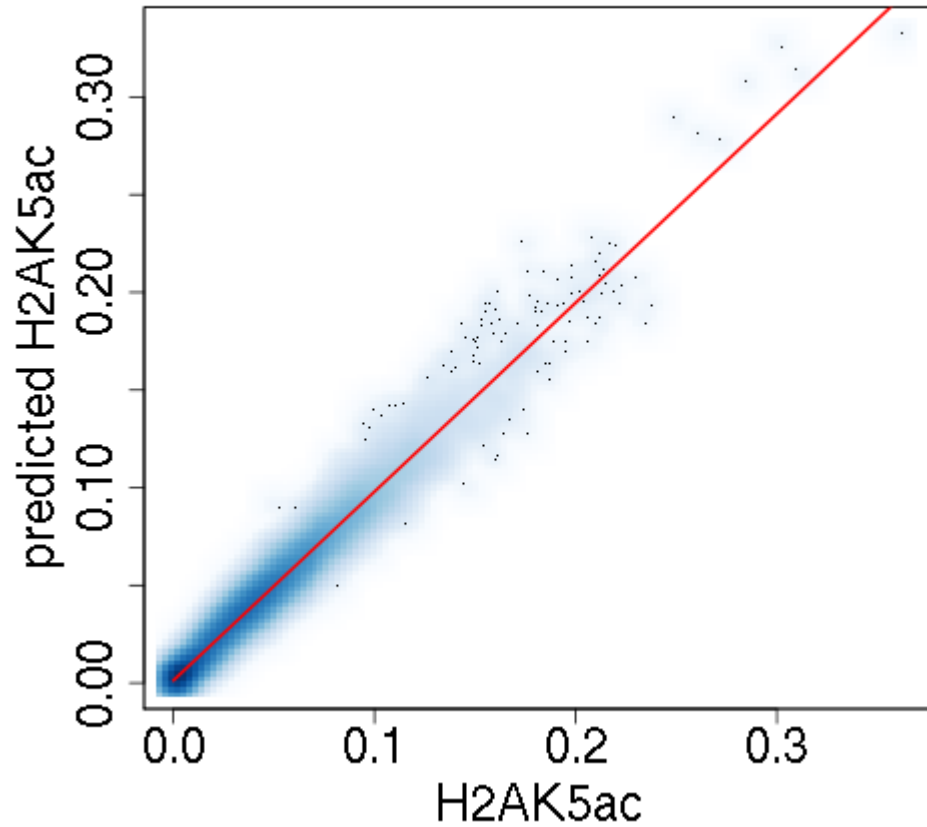
$$L(\mathbf{w}) = \sum_{n=1}^N [t_n \log y(\mathbf{x}_n) - y(\mathbf{x}_n)]$$

Poisson regression

linear regression, COR=0.87



Poisson regression, COR=0.98



Probabilistic supervised linear models

In R

X = your data, with one sample per row

t = your targets

```
model = glm(t~X,family="gaussian")
```

```
model = glm(t~X,family="binomial")
```

```
model = glm(t~X,family="poisson")
```

```
plot(t,model$fitted.values)
```

Plan

- *Linear regression*
 - *for actual regression*
 - *for classification => linear discriminant functions*
 - *probabilistic linear regression*
- *Probabilistic supervised linear models*
- *Kernels*

Kernels

Subplan

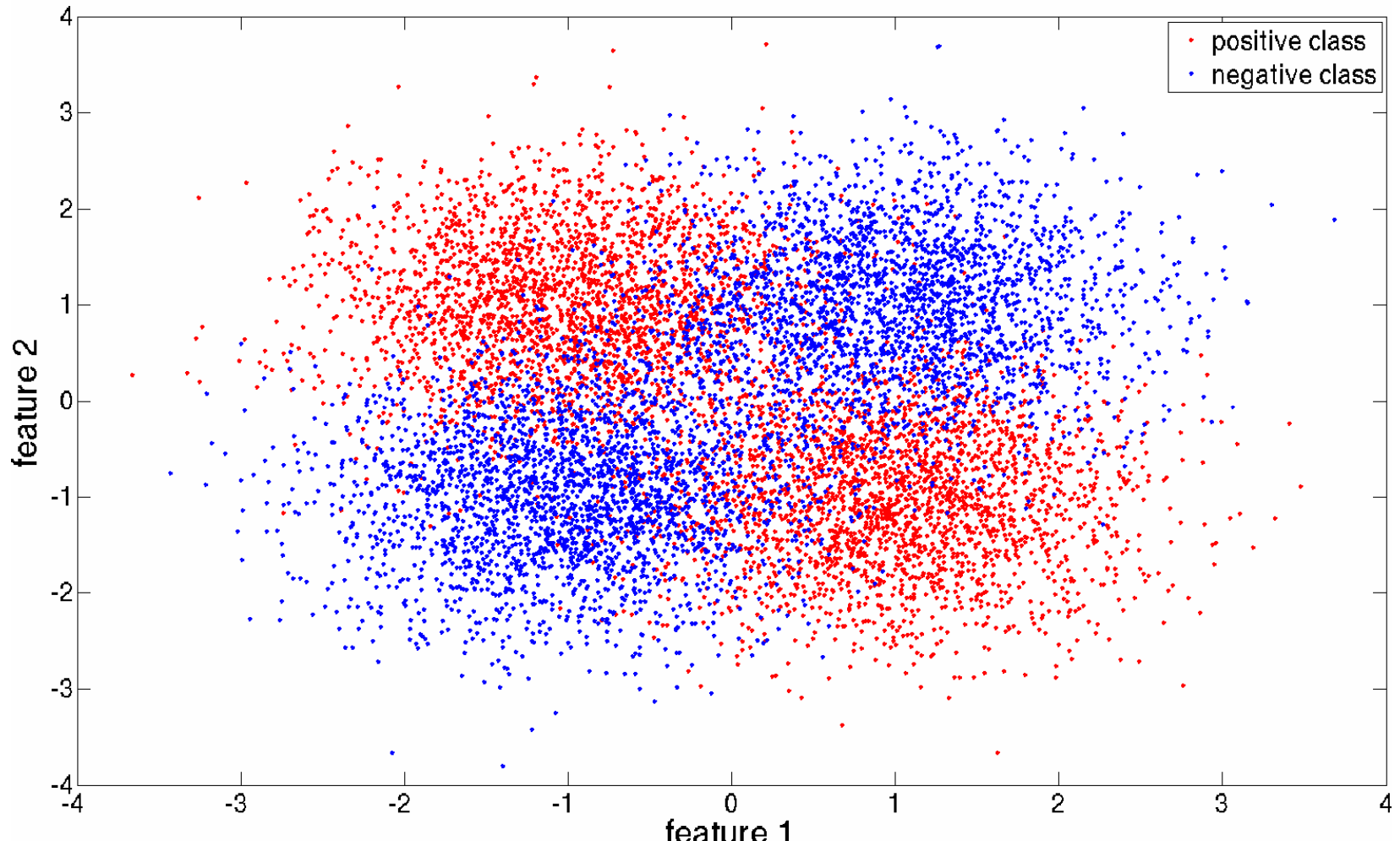
- Adding features
- Dual formulation
- Kernels
- Kernel regression

Subplan

- *Adding features*
- Dual formulation
- Kernels
- Kernel regression

Adding features

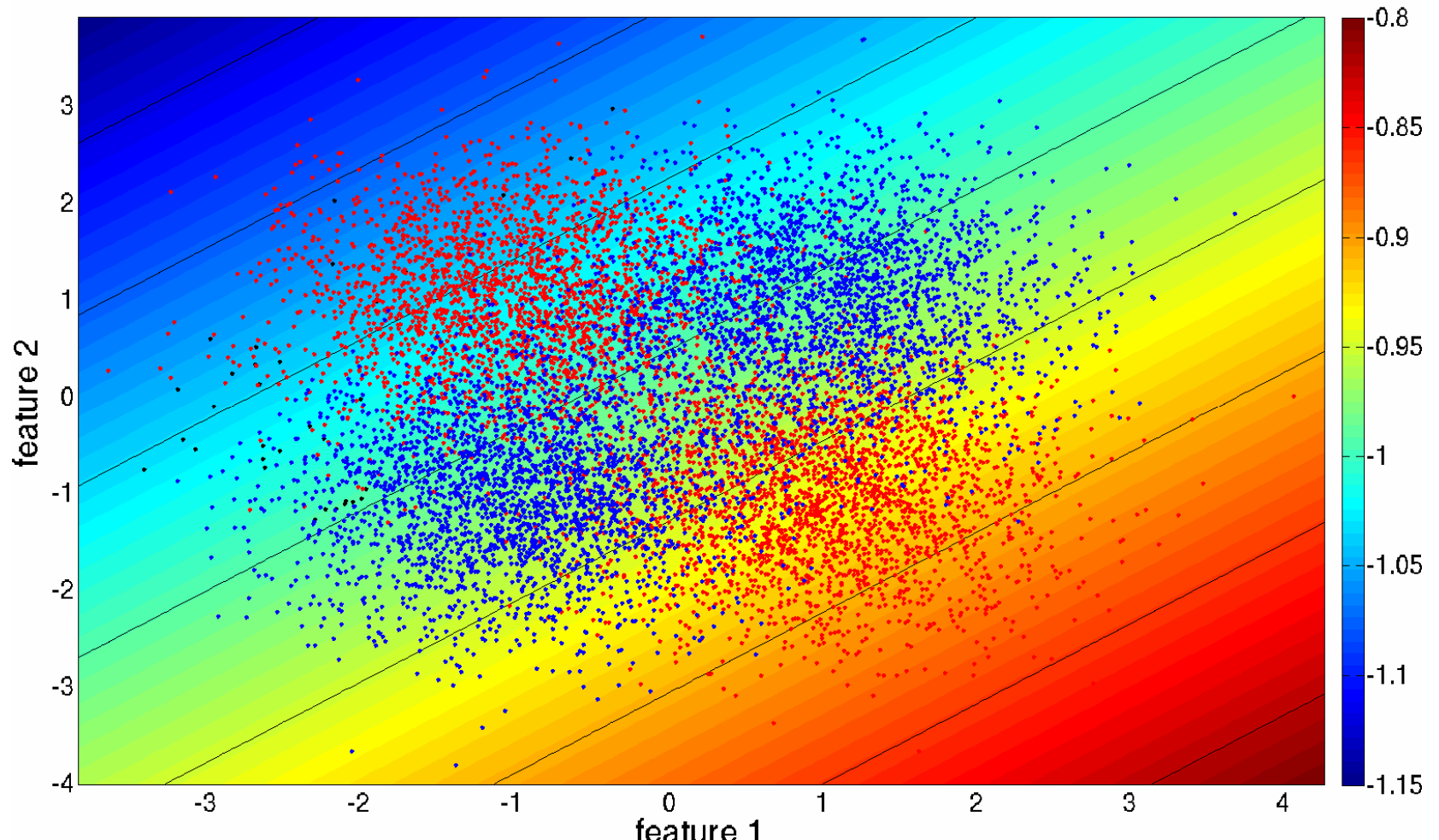
Works for linearly separable problems only
data



Adding features

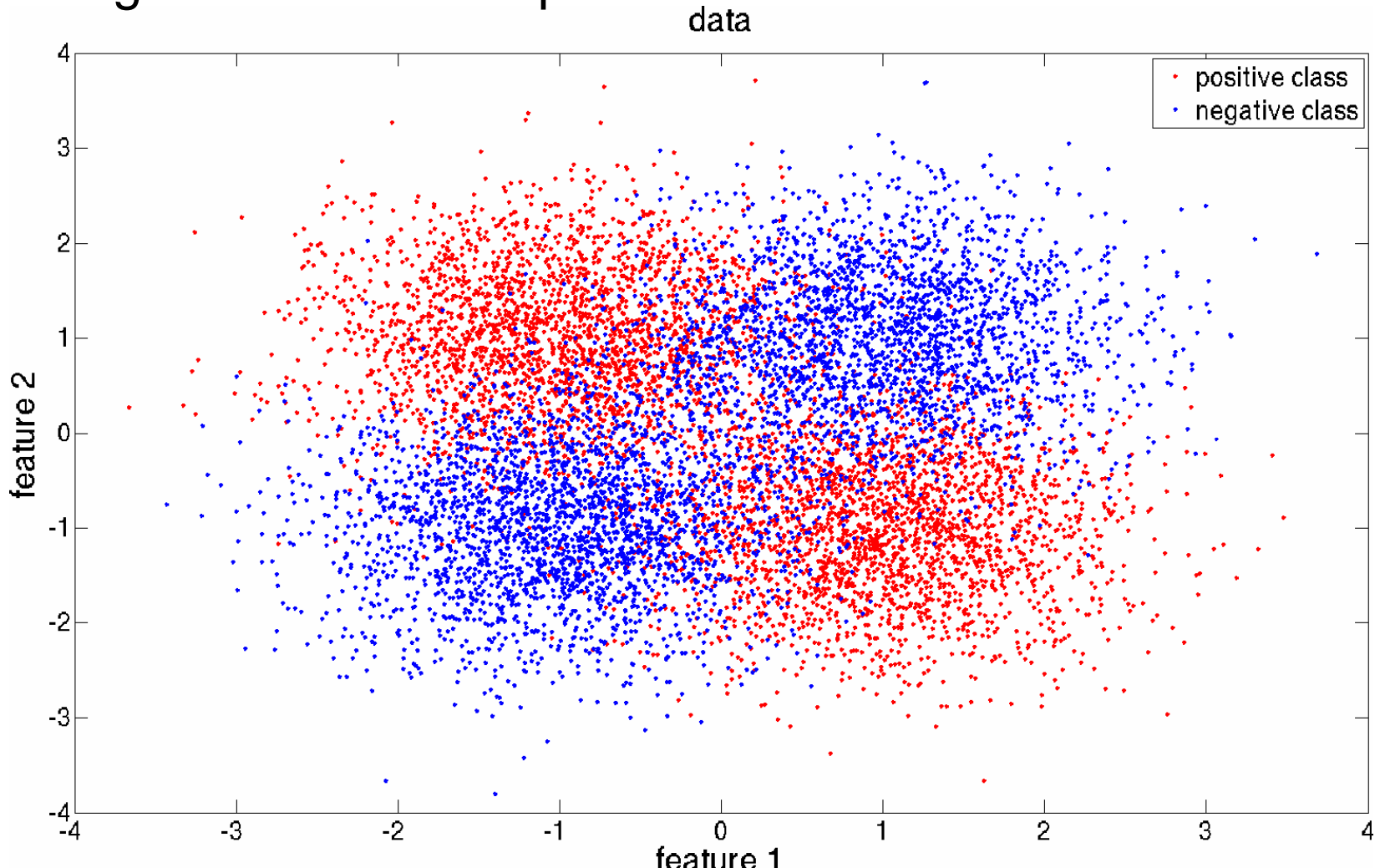
Works for linearly separable problems only

linear kernel



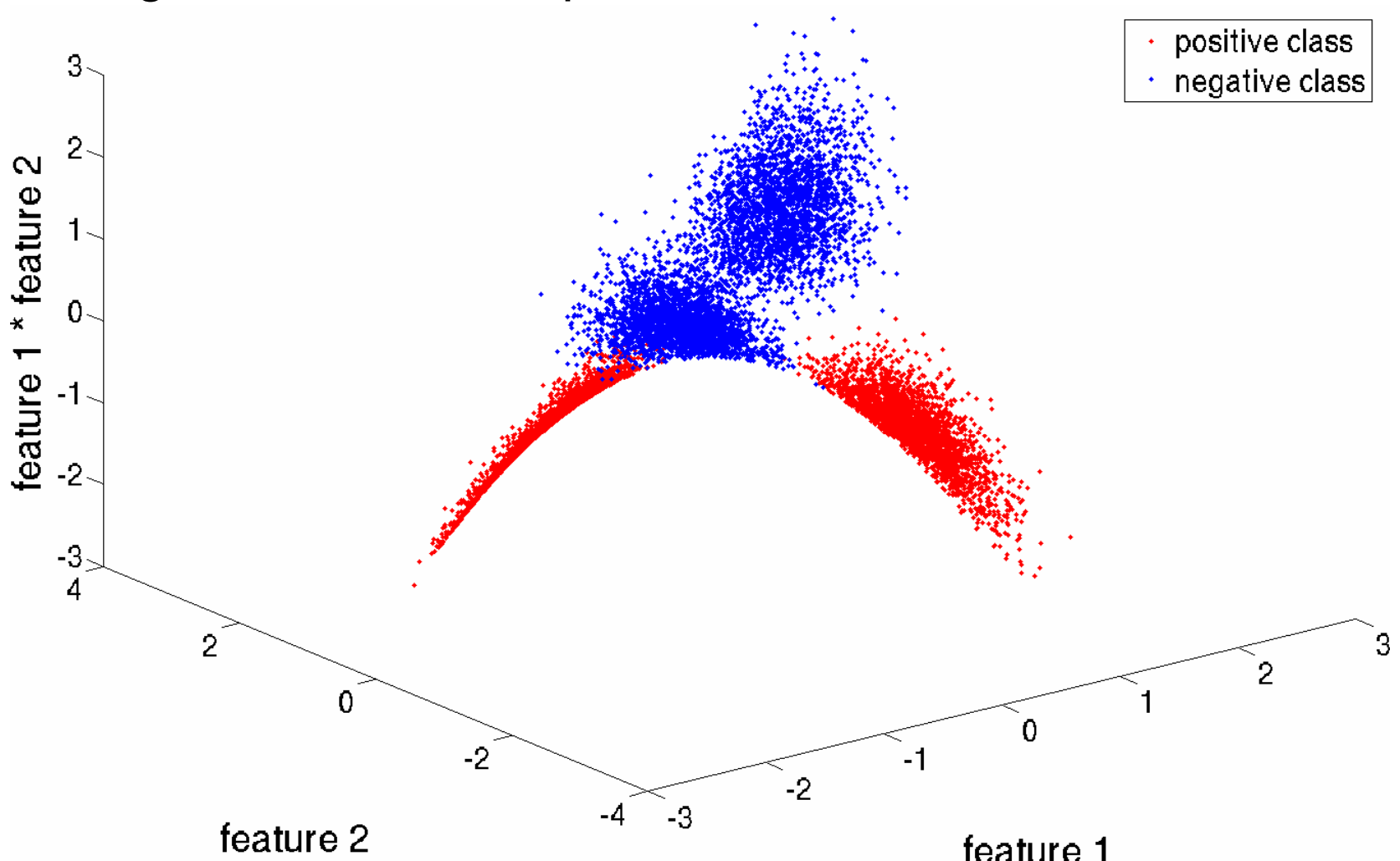
Adding features

Adding features can help



Adding features

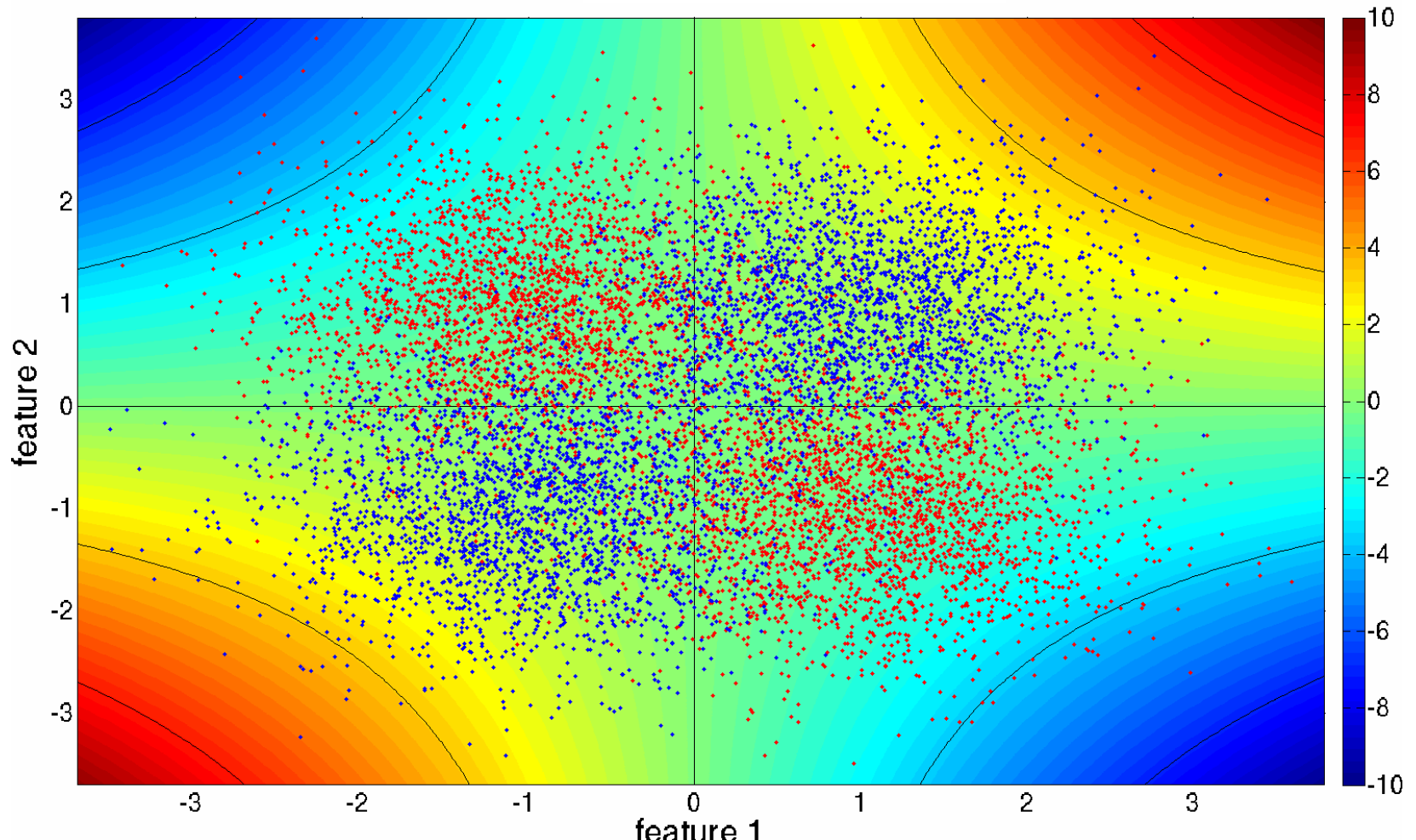
Adding features can help



Adding features

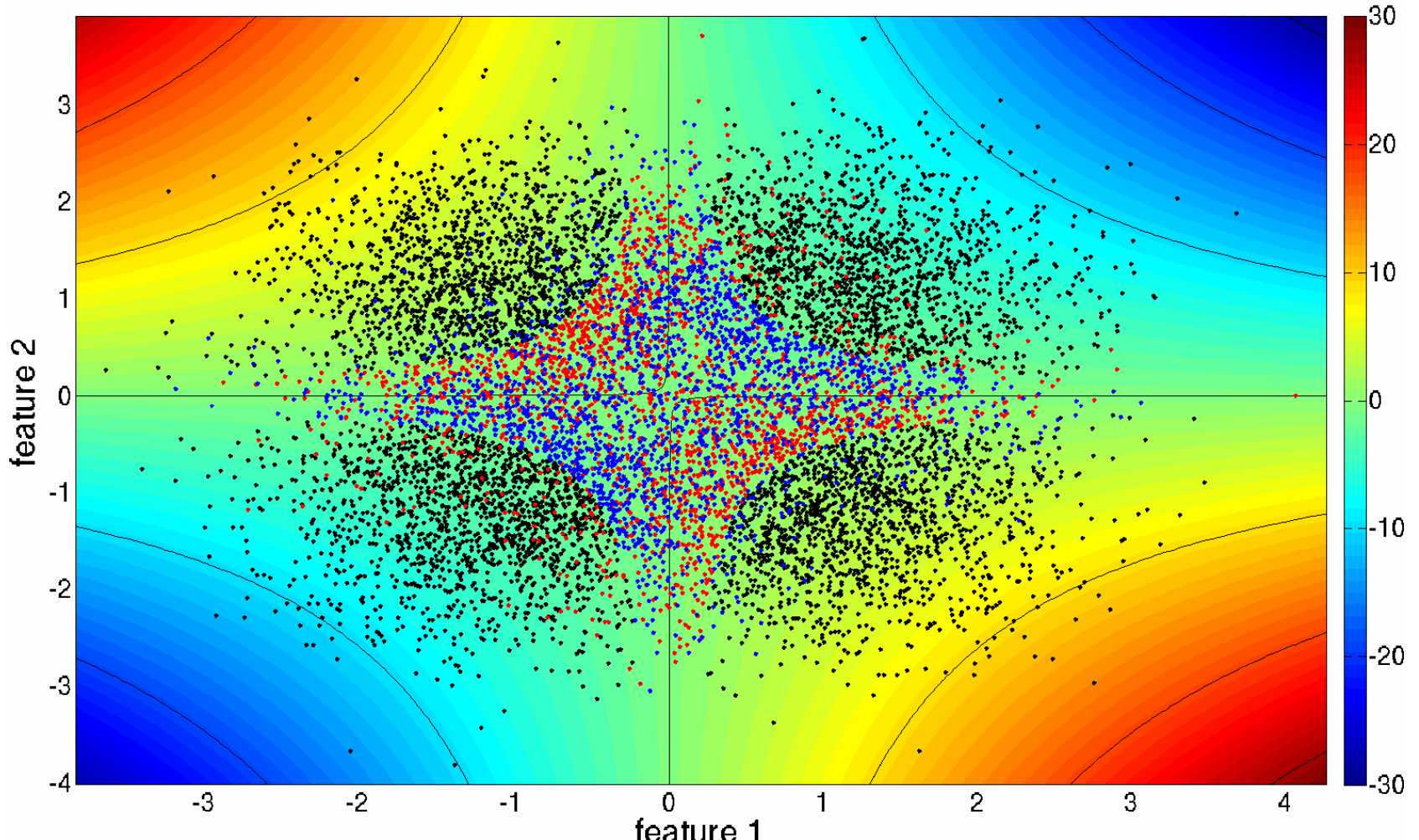
Adding features can help

Feature 1 * feature 2



Adding features

Adding features can help linear kernel



Adding features

- Adding features can help
- To defeat linearity... one solution is to change space!!
- Principle of kernel methods

Subplan

- *Adding features*
- *Dual formulation*
- Kernels
- Kernel regression

Dual formulation

- Reminder: $y(\mathbf{x}) = f(\mathbf{w}^T \mathbf{x} + w_0)$
- Rewrite \mathbf{w} as a linear combination of the training points

$$\mathbf{w} = \sum_{n=1}^N a_n \mathbf{x}_n = \mathbf{X}^T \mathbf{a}$$

- \mathbf{w} lives in the space of the datapoints so it is perfectly possible
- \mathbf{a} must be learnt instead of \mathbf{w}

Dual formulation

- Then we have

$$\begin{aligned}y(\mathbf{x}) &= f\left(\mathbf{w}^T \mathbf{x} + w_0\right) = f\left(\left(\sum_{n=1}^N a_n \mathbf{x}_n\right)^T \mathbf{x} + w_0\right) \\ &= f\left(\sum_{n=1}^N a_n \mathbf{x}_n^T \mathbf{x} + w_0\right) \\ &= f\left(\sum_{n=1}^N a_n \left(\mathbf{x}_n^T \mathbf{x}\right) + w_0\right)\end{aligned}$$

- Predictions are now made using

$$y(\hat{\mathbf{x}}) = f\left(\sum_{n=1}^N a_n \left(\mathbf{x}_n^T \hat{\mathbf{x}}\right) + w_0\right)$$

Dual formulation

- Back to the dual formulation of the error function

$$y(\mathbf{x}) = f\left(\sum_{n=1}^N a_n (\mathbf{x}_n^T \mathbf{x}) + w_0\right)$$

- Now \mathbf{x}_n never appears alone but always in a scalar product
- The scalar product is simply a measure of similarity, it can be replaced by any such measure

Kernel formulation

- The measure is called a kernel function and is denoted K
- Kernel formulation

$$y(\mathbf{x}) = f\left(\sum_{n=1}^N a_n K(\mathbf{x}_n, \mathbf{x}) + w_0\right)$$

- Kernel matrix \mathbf{K} such that $\mathbf{K}_{mn} = K(\mathbf{x}_m, \mathbf{x}_n)$
- For now $\mathbf{K} = \mathbf{X}\mathbf{X}^t$

Kernel formulation

- $y(\mathbf{x}) = f\left(\sum_{n=1}^N a_n K(\mathbf{x}_n, \mathbf{x}) + w_0\right)$
- For training points, this can be rewritten using \mathbf{K}
$$\begin{aligned}\mathbf{t} &= f(\mathbf{X}\mathbf{w} + w_0) = f(\mathbf{X}(\mathbf{X}^T \mathbf{a}) + w_0) = f((\mathbf{X}\mathbf{X}^T) \mathbf{a} + w_0) \\ &= f(\mathbf{K}\mathbf{a} + w_0)\end{aligned}$$
- Suppose we do Gaussian regression and $w_0=0$
$$\mathbf{t} = \mathbf{K}\mathbf{a} \Leftrightarrow \mathbf{a} = \mathbf{K}^{-1}\mathbf{t}$$

Subplan

- *Adding features*
- *Dual formulation*
- *Kernels*
- Kernel regression

Kernels

What is a kernel then?

- a function K such that $K(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T \phi(\mathbf{x}')$
 - ϕ : transformation from the original feature space \mathbf{x} to the induced feature space $\phi(\mathbf{x})$
 - note that sometimes ϕ is of infinite dimension
- a similarity measure

Kernels

Why is it such a big deal?

- we can choose any similarity measure we want!!
- the classifier is now linear in $\phi(\mathbf{x})$, not in \mathbf{x} , which can be useful if we know a good transformation
- we don't have to write $\phi(\mathbf{x})$ anymore, which means that very large or even infinite feature spaces can be used

Kernels

Examples of popular kernels

- Linear

$$K(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$$

- Gaussian (σ)

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2} \frac{\|\mathbf{x} - \mathbf{x}'\|^2}{\sigma^2}\right)$$

- Polynomial (d)

$$K(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T \mathbf{x}')^d$$

- Weighted spectrum (D)

$$K(\mathbf{x}, \mathbf{x}') = \sum_{d=1}^D w_d \phi_d(\mathbf{x})^T \phi_d(\mathbf{x}')$$

with $\phi_d(\mathbf{x})$ = number of occurrences of the d-mers in \mathbf{x}

Kernels

Combining kernels

- Linear combination $K(\mathbf{x}, \mathbf{x}') = \sum_i \alpha_i K_i(\mathbf{x}, \mathbf{x}')$
- Multiplication $K(\mathbf{x}, \mathbf{x}') = \prod_i K_i(\mathbf{x}, \mathbf{x}')$
- Other combinations are less popular

Subplan

- *Adding features*
- *Dual formulation*
- *Kernels*
- *Kernel regression*

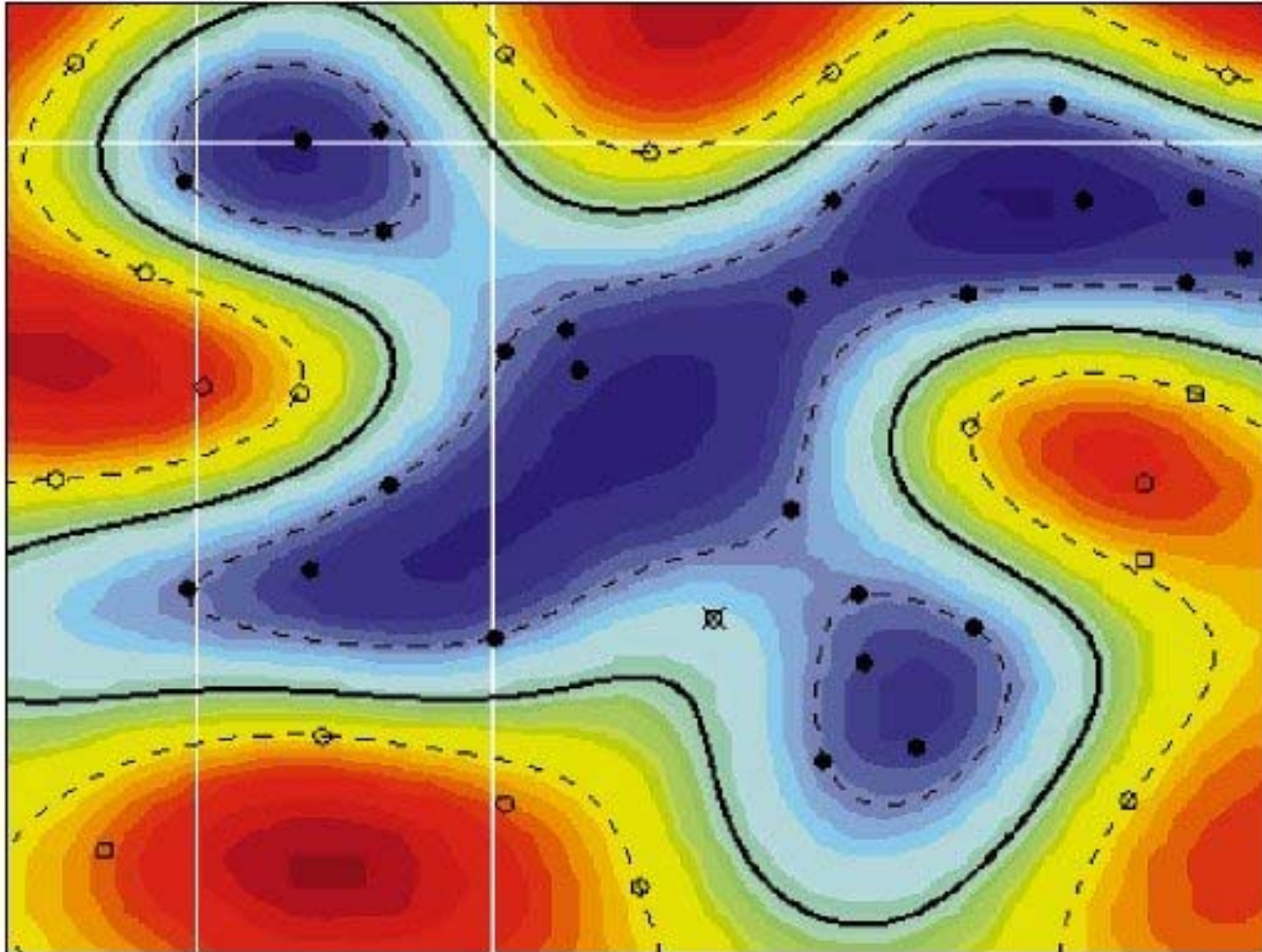
Kernel regression

Consequences of using a kernel

- no longer linear

Kernel regression

Consequences of using a kernel



Kernel regression

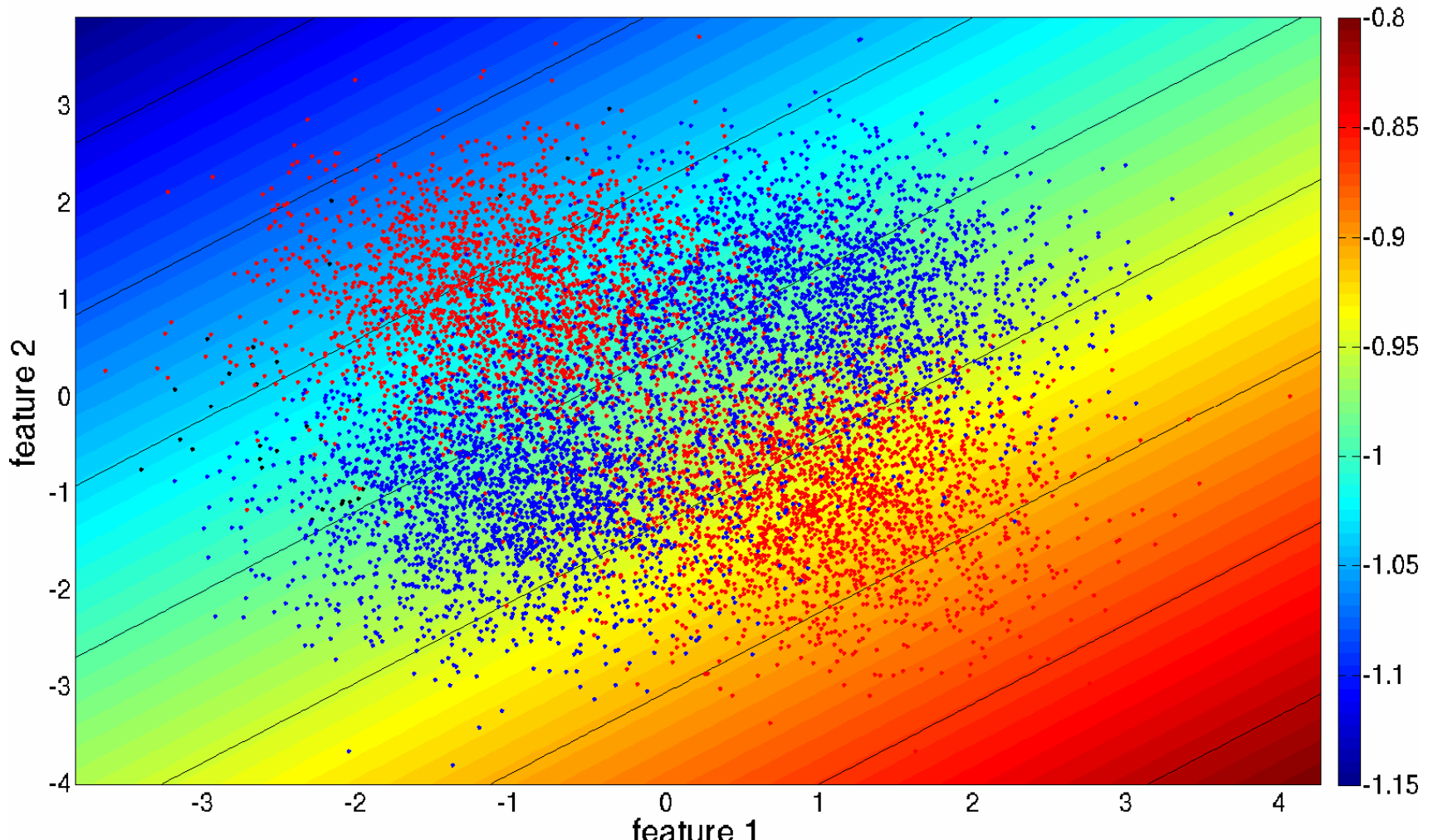
Consequences of using a kernel

- no longer linear
- more problems can be solved

Kernel regression

Consequences of using a kernel

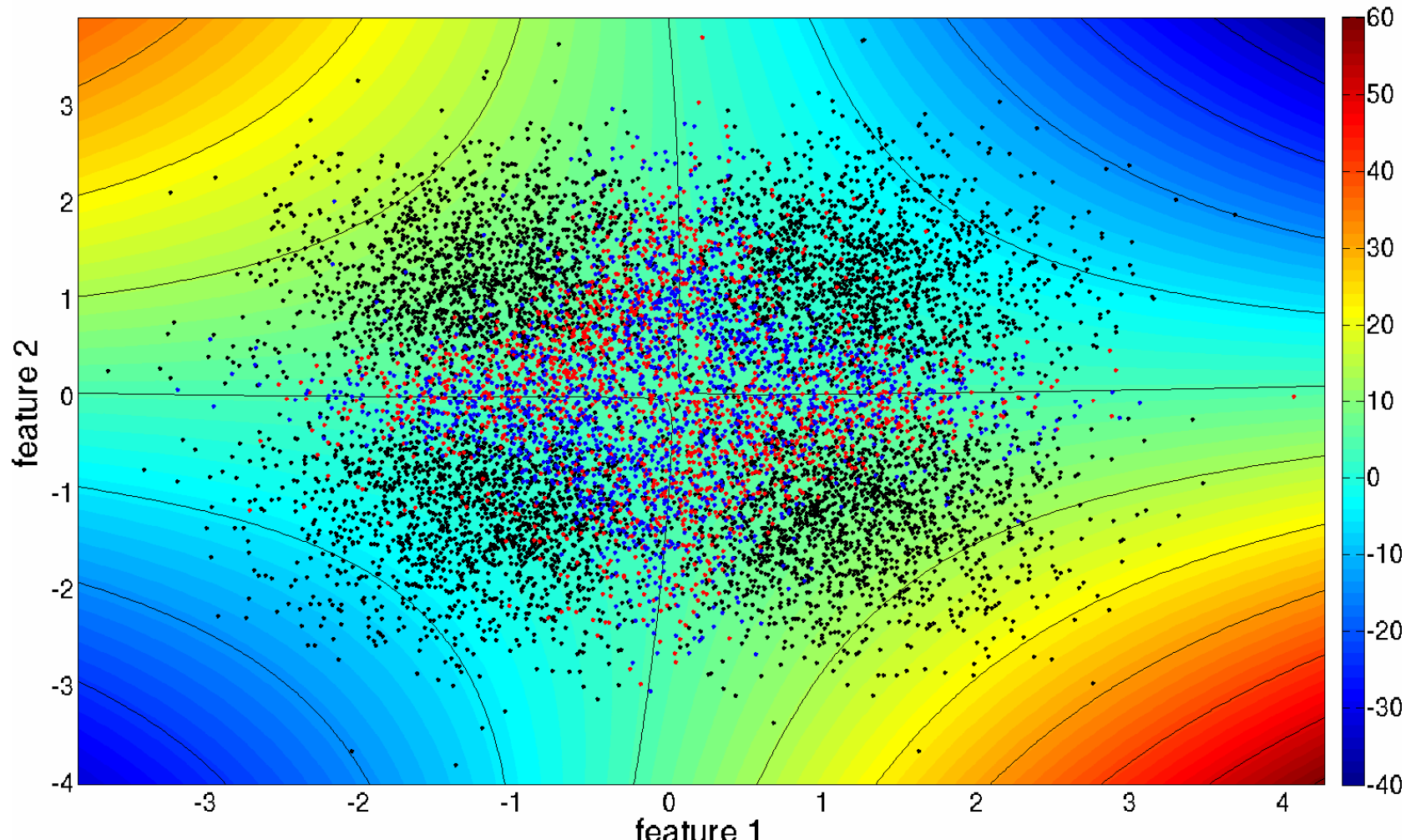
linear kernel



Kernel regression

Consequences of using a kernel

gaussian kernel - width = w_0



Kernel regression

Consequences of using a kernel

- no longer linear
- more problems can be solved
- the kernel induces a different (usually highly dimensional) feature space where linear classification is easier – same trick we applied at the beginning

Kernels

Only for linear regression?

- no, pretty much most linear models can be recast into a dual form to see kernels appear
- kernel regression, kernel discriminant analysis (kernel LDA), kernel SVMs, kernel logistic regression, kernel PCA, etc etc