

---

# Data Science for Geosciences

## Classification

## Classification problem

### Variable terminology

- ▶ observed data  $X \in \mathbb{R}^p$  referred to as *input* vector, *predictors* or *features*
- ▶ data to predict  $Y$  referred to as *output* variables, or *responses*

### Classification task

$Y$  are *categorical* data (discrete qualitative variables) that takes value in a set  $\mathcal{Y}$ , e.g.

- ▶ `email`  $\in \{\text{spam}, \text{ham}\}$
- ▶ `handwritten digits`  $\in \{0, \dots, 9\}$

Given a feature vector  $X \in \mathbb{R}^p$ , build a function  $f(X)$  that takes as input the feature vector  $X$  and predicts its value for  $Y \in \mathcal{Y}$

- ☞ Try to minimize the *misclassification rate*  $\mathcal{E}[f] \equiv \Pr(f(X) \neq Y)$

## Outline

### Model based approaches for classification

Bayes Classifier

Linear/Quadratic Discriminant Analysis (LDA/QDA)

### Black box approaches for classification

K Nearest Neighbors (K-NN)

Support Vector Machine (SVM)

### Unsupervised classification : Clustering

K means

EM Algorithm for Gaussian Mixture Models

## Classification outline

### Model based approaches for classification

Bayes Classifier

Linear/Quadratic Discriminant Analysis (LDA/QDA)

### Black box approaches for classification

K Nearest Neighbors (K-NN)

Support Vector Machine (SVM)

### Unsupervised classification : Clustering

K means

EM Algorithm for Gaussian Mixture Models

## Bayes rule for classification

Classification problem with  $K$  classes :  $Y \in \mathcal{Y} = \{1, \dots, K\}$ ,

Probability of class  $Y = k$  given  $X = x$

Bayes rule :

$$\begin{aligned}\Pr(Y = k|X = x) &= \frac{p(x|Y = k)\Pr(Y = k)}{p(x)} = \frac{p(x|Y = k)\Pr(Y = k)}{\sum_{j=1}^K p(x|Y = j)\Pr(Y = j)}, \\ &= \frac{\pi_k p_k(x)}{\sum_{j=1}^K \pi_j p_j(x)}\end{aligned}$$

- ▶  $p_k(x) \equiv p(x|Y = k)$  is the *density* for  $X$  in class  $k$
- ▶  $\pi_k \equiv \Pr(Y = k)$  is the *weight*, or *prior* probability of class  $k$

## Bayes classifier

### Definition

The Bayes classification rule  $f^*$  is defined as

$$f^*(x) = \arg \max_{k \in \mathcal{Y}} \Pr(Y = k | X = x).$$

### Theorem

The Bayes classification rule  $f^*$  is optimal in the misclassification rate sense where  $\mathcal{E}[f] = \Pr(f(X) \neq Y)$  :

$$\text{for any rule } f, \mathcal{E}[f] \geq \mathcal{E}[f^*],$$

### Remarks

- ▶  $f^*(X) \equiv \text{maximum a posteriori}$  (MAP) estimate
- ▶ In real-word applications, the distribution of  $(X, Y)$  is unknown  $\Rightarrow$  no analytical expression of  $f^*(X)$ . But useful reference on academic examples.

## Generative models

Two kinds of approaches based on a model :

1. Discriminative approaches : direct learning of  $p(Y|X)$ ,  
e.g. Regression, logistic regression
2. Generative models : learning of the joint distribution  $p(X, Y)$

$$p(X, Y) = \underbrace{p(X|Y)}_{\text{likelihood}} \underbrace{\Pr(Y)}_{\text{prior}},$$

e.g. linear/quadratic discriminant analysis, Naïve Bayes

## Generative models : Estimation problem

### Assumptions

- ▶ classification problem with  $K$  classes :  $Y \in \mathcal{Y} = \{1, \dots, K\}$ ,
- ▶ input variables :  $X \in \mathbb{R}^p$

Bayes rule :

$$\Pr(Y = k|X = x) = \frac{p(x|Y = k)\Pr(Y = k)}{p(x)} = \frac{p(x|Y = k)\Pr(Y = k)}{\sum_{j=1}^K p(x|Y = j)\Pr(Y = j)}.$$

In practice, the following quantities are unknown :

- ▶ densities of each class  $p_k(x) \equiv p(x|Y = k)$
- ▶ weights, or prior probabilities, of each class  $\pi_k \equiv \Pr(Y = k)$

### Estimation problem

These quantities must be learned on a training set :

learning problem  $\Leftrightarrow$  estimation problem in a parametric or not way

## Quadratic Discriminant Analysis (QDA)

### Supervised classification assumptions

- ▶  $X \in \mathbb{R}^p$ ,  $Y \in \mathcal{Y} = \{1, \dots, K\}$ ,
- ▶ sized  $n$  training set  $(X_1, Y_1), \dots (X_n, Y_n)$

### QDA Assumptions

The input variables  $X$ , given a class  $Y = k$ , are distributed according to a parametric and Gaussian distribution :

$$X|Y = k \sim \mathcal{N}(\mu_k, \Sigma_k) \Leftrightarrow p_k(x) = \frac{1}{(2\pi)^{p/2} |\Sigma_k|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k)}$$

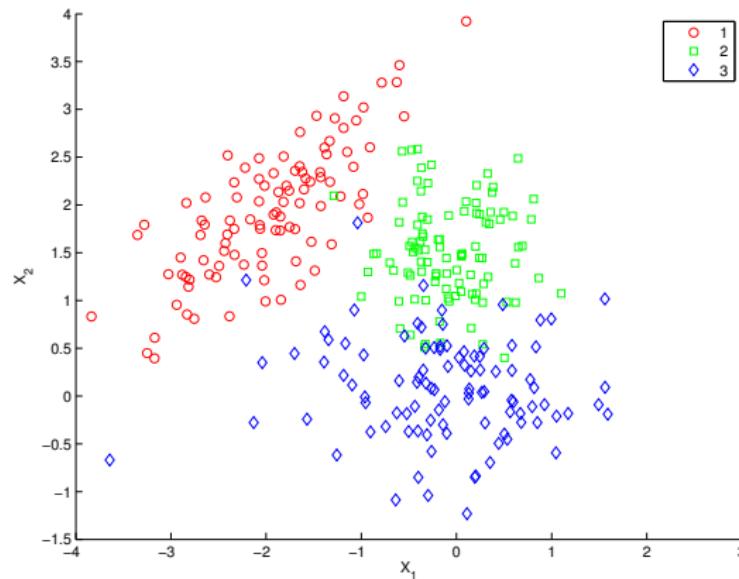
The Gaussian parameters are, for each class  $k = 1, \dots, K$

- ▶ mean vectors  $\mu_k \in \mathbb{R}^p$ ,
- ▶ covariance matrices  $\Sigma_k \in \mathbb{R}^{p \times p}$ ,
- ☞ set of parameters  $\theta_k \equiv \{\mu_k, \Sigma_k\}$ , plus the weights  $\pi_k$ , for  $k = 1, \dots, K$ .

## Example

Mixture of  $K = 3$  Gaussians

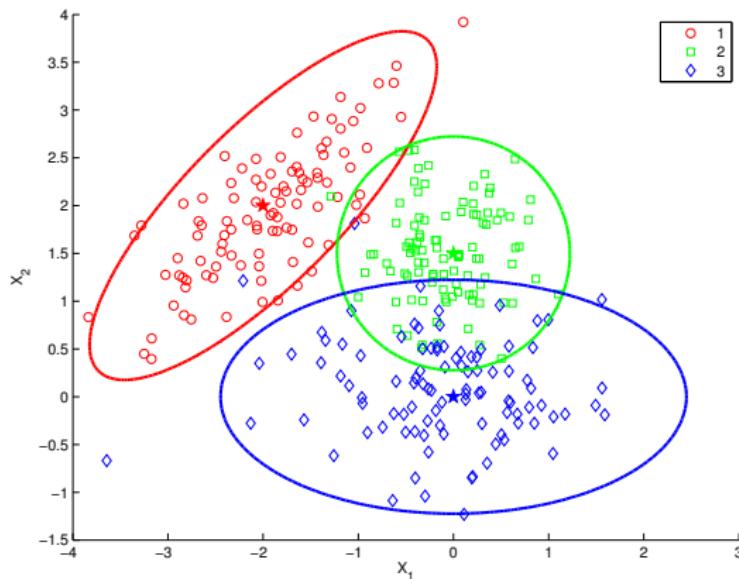
- $Y \in \{1, 2, 3\}$
- $X \in \mathbb{R}^2$



## Example

Mixture of  $K = 3$  Gaussians

- $Y \in \{1, 2, 3\}$
- $X \in \mathbb{R}^2$



95% theoretical confidence regions

## QDA parameter estimation

### Log-likelihood

For the training set,

$$\begin{aligned}
 \ell(\theta_1, \dots, \theta_K, \pi_1, \dots, \pi_{K-1}) &= \log p((x_1, y_1), \dots, (x_n, y_n)), \\
 &= \sum_{i=1}^n \log p((x_i, y_i)), \quad \leftarrow \text{ i.i.d. training set,} \\
 &= \sum_{i=1}^n \log [p(x_i | y_i) \Pr(y_i)], \\
 &= \sum_{i=1}^n \log [\pi_{y_i} p_{y_i}(x_i; \theta_{y_i})].
 \end{aligned}$$

Rk :  $\pi_K = 1 - \sum_{j=1}^{K-1} \pi_j$  is not a parameter

## QDA parameter estimation (Cont'd)

### Notations

- ▶  $n_k = \#\{y_i = k\}$  is the number of training samples in class  $k$ ,
- ▶  $\sum_{y_i=k}$  is the sum over all the indices  $i$  of the training samples in class  $k$

### (Unbiased) Maximum likelihood estimators (MLE)

- ▶  $\hat{\pi}_k = \frac{n_k}{n}$ ,  $\leftarrow$  sample proportion
- ▶  $\hat{\mu}_k = \frac{\sum_{y_i=k} x_i}{n_k}$ ,  $\leftarrow$  sample mean
- ▶  $\hat{\Sigma}_k = \frac{1}{n_k - 1} \sum_{y_i=k} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T$ ,  $\leftarrow$  sample covariance

Rk :  $\frac{1}{n_k - 1}$  is a bias correction factor for the covariance MLE (otherwise  $\frac{1}{n_k}$ )

## Discriminant functions

For model based approaches, Bayes classifier is defined as

$$f^*(x) = \arg \max_{k \in \mathcal{Y}} \Pr(Y = k | X = x)$$

- ▶ equivalent to consider a set of functions  $\delta_k(x)$ , for  $k \in \mathcal{Y}$ , derived from a monotone transformation of posterior probability  $\Pr(Y = k | X = x)$
- ▶ decision boundary between classes  $k$  and  $l$  is then defined as the set  $\{x \in \mathcal{X} : \delta_k(x) = \delta_l(x)\}$

### Definition

$\delta_k(x)$  are called the **discriminant functions** of each class  $k$

☞  $x$  is predicted in the  $k_0$  class such that  $k_0 = \arg \max_{k \in \mathcal{Y}} \delta_k(x)$

## QDA decision rule

The classification rule becomes

$$\begin{aligned} f(x) &= \arg \max_{k \in \mathcal{Y}} \Pr(Y = k | X = x, \hat{\theta}, \hat{\pi}), \\ &= \arg \max_{k \in \mathcal{Y}} \underbrace{\log \Pr(Y = k | X = x, \hat{\theta}, \hat{\pi})}_{\delta_k(x)}, \end{aligned}$$

where

$$\delta_k(x) = -\frac{1}{2} \log |\hat{\Sigma}_k| - \frac{1}{2} (x - \hat{\mu}_k)^T \hat{\Sigma}_k^{-1} (x - \hat{\mu}_k) + \log \hat{\pi}_k + \text{const},$$

is the **discriminant function**

### Remarks

1. different rule than the Bayes classifier as  $\theta$  replaced by  $\hat{\theta}$  (and  $\pi$  replaced by  $\hat{\pi}$ )
2. when  $n \gg p$ ,  $\hat{\theta} \rightarrow \theta$  (and  $\hat{\pi} \rightarrow \pi$ ) : convergence to the optimal classifier if the Gaussian model is correct...

## QDA decision boundary

The boundary between two classes  $k$  and  $l$  is described by the equation

$$\delta_k(x) = \delta_l(x) \Leftrightarrow C_{k,l} + L_{k,l}^T x + x^T Q_{k,l}^T x = 0, \quad \leftarrow \text{quadratic equation}$$

where

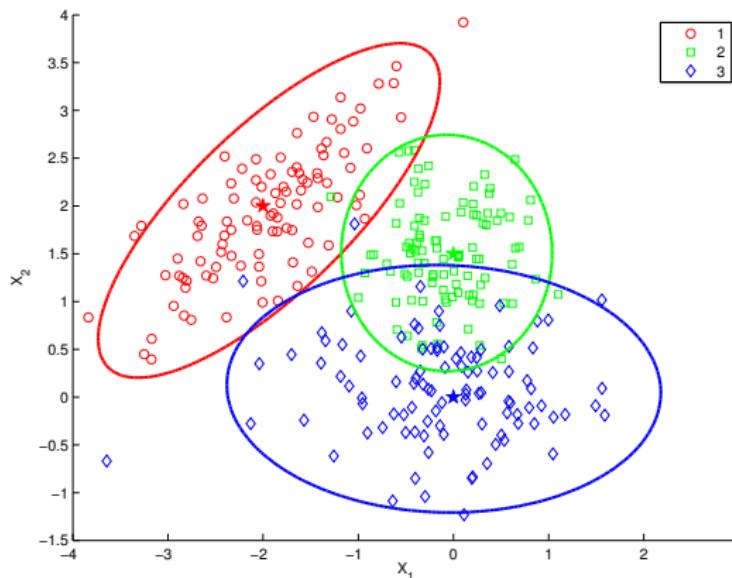
- ▶  $C_{k,l} = -\frac{1}{2} \log \frac{|\hat{\Sigma}_k|}{|\hat{\Sigma}_l|} + \log \frac{\hat{\pi}_k}{\hat{\pi}_l} - \frac{1}{2} \hat{\mu}_k^T \hat{\Sigma}_k^{-1} \hat{\mu}_k + \frac{1}{2} \hat{\mu}_l^T \hat{\Sigma}_l^{-1} \hat{\mu}_l, \quad \leftarrow \text{scalar}$
- ▶  $L_{k,l} = \hat{\Sigma}_k^{-1} \hat{\mu}_k - \hat{\Sigma}_l^{-1} \hat{\mu}_l, \quad \leftarrow \text{vector in } \mathbb{R}^p$
- ▶  $Q_{k,l} = \frac{1}{2} \left( -\hat{\Sigma}_k^{-1} + \hat{\Sigma}_l^{-1} \right), \quad \leftarrow \text{matrix in } \mathbb{R}^{p \times p}$

☞ Quadratic discriminant analysis

## QDA example

Mixture of  $K = 3$  Gaussians

- Estimation of the parameters  $\hat{\mu}_k$ ,  $\hat{\Sigma}_k$  and  $\hat{\pi}_k$ , for  $k = 1, 2, 3$

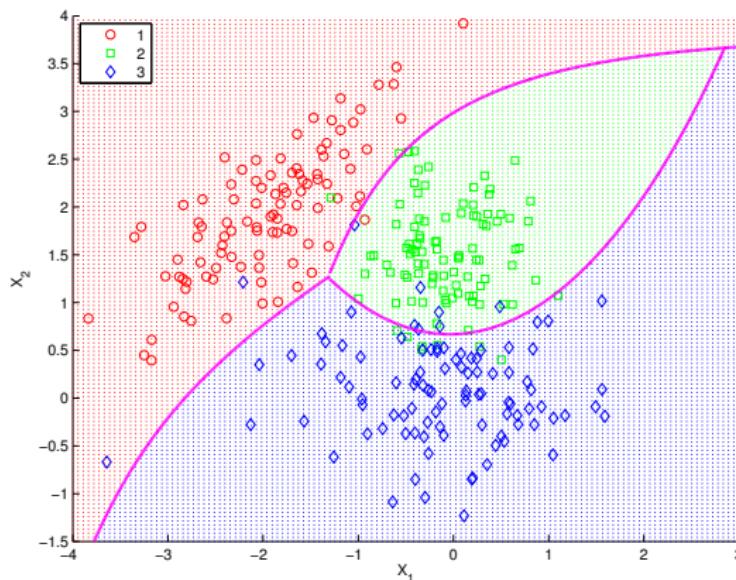


95% estimated confidence regions

## QDA example (Cont'd)

Mixture of  $K = 3$  Gaussians

- ▶ Classification rule :  $\arg \max_{k=1,2,3} \delta_k(x)$
- ▶ Quadratic boundaries  $\{x; \delta_k(x) = \delta_l(x)\}$



## LDA principle

### LDA Assumptions

Additional simplifying assumption w.r.t. QDA : all the class covariance matrices are identical (“homoscedasticity”), i.e.  $\Sigma_k = \Sigma$ , for  $k = 1, \dots, K$

### (Unbiased) Maximum likelihood estimators (MLE)

- ▶  $\hat{\pi}_k$  and  $\hat{\mu}_k$  are unchanged,
- ▶  $\hat{\Sigma} = \frac{1}{n-K} \sum_{k=1}^K \sum_{y_i=k} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T$ ,  $\leftarrow$  pooled covariance

Rk :  $\frac{1}{n-K}$  is a bias correction factor for the covariance MLE (otherwise  $\frac{1}{n}$ )

### LDA discriminant function

$$\delta_k(x) = -\frac{1}{2} \log |\hat{\Sigma}| - \frac{1}{2}(x - \hat{\mu}_k)^T \hat{\Sigma}^{-1} (x - \hat{\mu}_k) + \log \hat{\pi}_k + \text{Cst},$$

## LDA decision boundary

The boundary between two classes  $k$  and  $l$  reduces to the equation

$$\delta_k(x) = \delta_l(x) \Leftrightarrow C_{k,l} + L_{k,l}^T x = 0, \quad \leftarrow \text{linear equation}$$

where

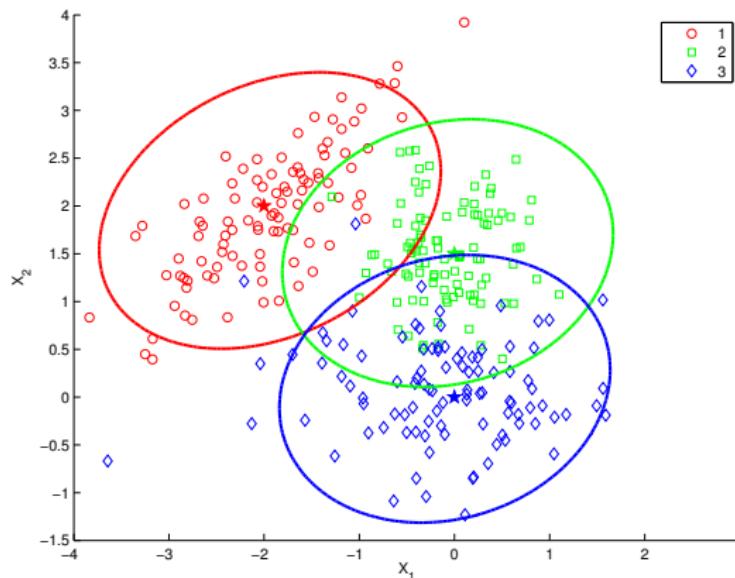
- ▶  $C_{k,l} = \log \frac{\hat{\pi}_k}{\hat{\pi}_l} - \frac{1}{2} \hat{\mu}_k^T \hat{\Sigma}^{-1} \hat{\mu}_k + \frac{1}{2} \hat{\mu}_l^T \hat{\Sigma}^{-1} \hat{\mu}_l, \quad \leftarrow \text{scalar}$
- ▶  $L_{k,l} = \hat{\Sigma}^{-1} (\hat{\mu}_k - \hat{\mu}_l), \quad \leftarrow \text{vector in } \mathbb{R}^p$
- ▶  $Q_{k,l} = 0,$

☞ Linear discriminant analysis

## Linear Discriminant Analysis (LDA)

Mixture of  $K = 3$  Gaussians

- Estimation of the parameters  $\hat{\mu}_k$ ,  $\hat{\pi}_k$ , for  $k = 1, 2, 3$ , and  $\hat{\Sigma}$

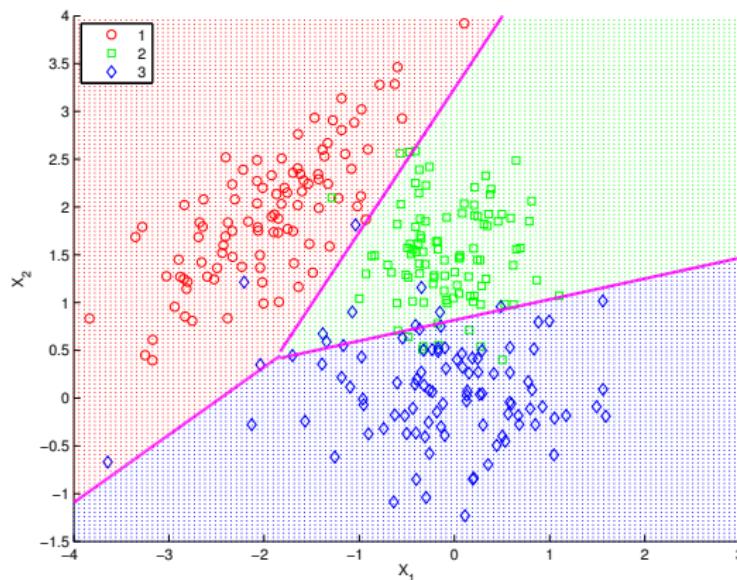


95% estimated confidence regions

## Linear Discriminant Analysis (LDA)

Mixture of  $K = 3$  Gaussians

- ▶ Classification rule :  $\arg \max_{k=1,2,3} \delta_k(x)$
- ▶ linear boundaries  $\{x; \delta_k(x) = \delta_l(x)\}$



## Complexity of discriminant analysis methods

### Effective number of parameters

- ▶ LDA :  $(K - 1) \times (p + 1) = O(Kp)$
- ▶ QDA :  $(K - 1) \times \left( \frac{p(p+3)}{2} + 1 \right) = O(Kp^2)$

### Remarks

- ▶ in high dimension, i.e.  $p \approx n$  or  $p > n$ , LDA is more stable than QDA which is more prone to overfitting,
- ▶ both methods appear however to be robust on a large number of real-word datasets
- ▶ LDA can be viewed in some cases as a least squares regression method
- ▶ LDA performs a dimension reduction to a subspace of dimension  $\leq K - 1$  generated by the vectors  $z_k = \Sigma^{-1}\hat{\mu}_k \leftarrow$  dimension reduction from  $p$  to  $K - 1$ !

## Conclusions on discriminant analysis

### Generative models

- ▶ learning/estimation of  $p(X, Y) = p(X|Y) \Pr(Y)$ ,
- ▶ derivation of  $\Pr(Y|X)$  from Bayes rule,

Different assumptions on the class densities  $p_k(x) = p(X = x|Y = k)$

- ▶ QDA/LDA : Gaussian parametric model
- ☞ performs well on many real-word datasets
- ☞ LDA is especially useful when  $n$  is small

### Perspectives

Black box approaches : direct learning of the prediction rule  $f$

## Classification outline

Model based approaches for classification

Bayes Classifier

Linear/Quadratic Discriminant Analysis (LDA/QDA)

Black box approaches for classification

K Nearest Neighbors (K-NN)

Support Vector Machine (SVM)

Unsupervised classification : Clustering

K means

EM Algorithm for Gaussian Mixture Models

## $k$ Nearest-Neighbors ( $k$ -NN) for regression

For a **regression** problem  $Y \in \mathbb{R}$ , the prediction model is directly defined, for  $X = x$ , as :

$$\hat{Y}(x) = \frac{1}{k} \sum_{X_i \in N_k(x)} Y_i,$$

where  $N_k(x)$  is the neighborhood of  $x$  defined by the  $k$  closest inputs  $X_i$  in the training set  $\{(X_i, Y_i)\}_{i=1\dots n}$

### Properties

$$\hat{Y}(x) = \text{Average } \{Y_i | X_i \in N_k(x)\} \approx E[Y | X = x]$$

But two approximations problematic in high dimension :

- ▶ Expectation  $\approx$  Average,
- ▶ Conditioning at a point  $\approx$  conditioning on a neighborhood

## $k$ Nearest-Neighbors ( $k$ -NN) for classification

### Binary classification problem

For a binary classification problem  $Y \in \{0, 1\}$ , the classification rule can be derived, for  $X = x$ , as

$$f(x) = \begin{cases} 1 & \text{if } \hat{Y}(x) > \frac{1}{2}, \\ 0 & \text{otherwise} \end{cases}$$

where  $\hat{Y}(x) = \frac{1}{k} \sum_{X_i \in N_k(x)} Y_i$  is the average of the binary labels of the  $k$  nearest neighbors of the testing point  $X = x$ .

### Classification rule associated with $k$ -NN

The binary classification problem can be directly extended for an arbitrary number of class  $K$  :

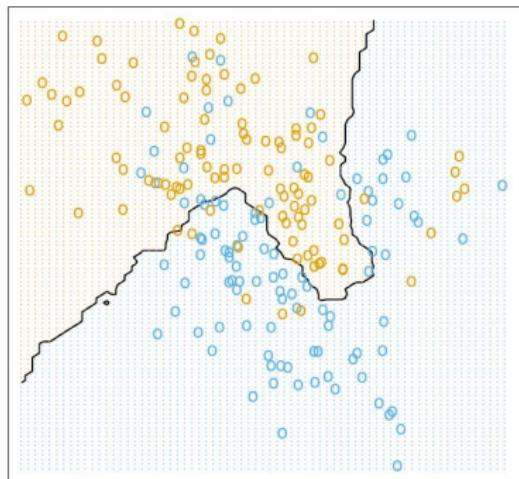
$f(x) \equiv$  majority vote among the  $k$  closest neighbors of the testing point  $x$ ,  
 $\equiv$  assignment to the most common class among the  $k$  nearest neighbors

## K Nearest-Neighbors

*k*-NN : complexity parameter *k*

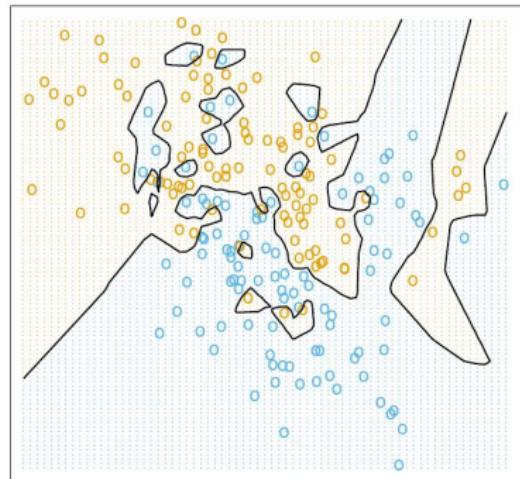
The effective number of parameters expresses as  $N_{\text{eff}} = \frac{n}{k}$ , where  $n$  is the size of the training sample

15-Nearest Neighbor Classifier



$$k = 15, N_{\text{eff}} \approx 13$$

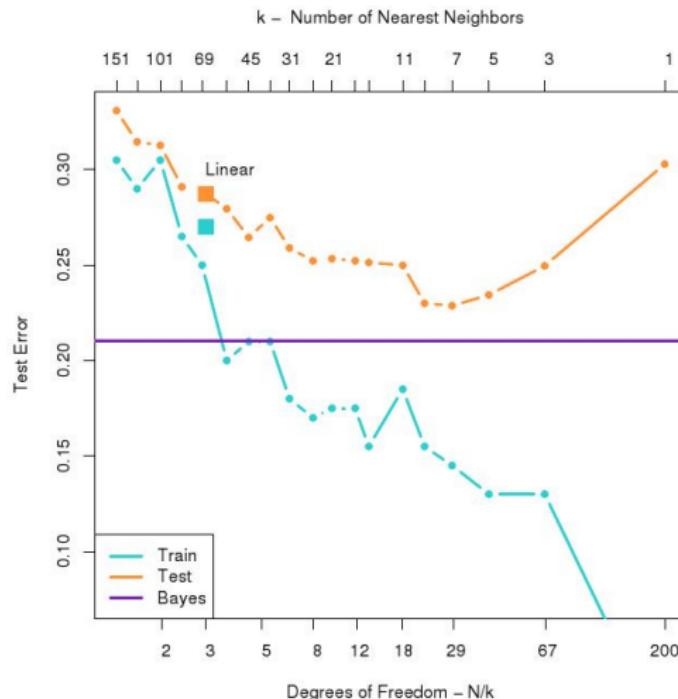
1-Nearest Neighbor Classifier



$$k = 1, N_{\text{eff}} \approx 200$$

- $k = 1 \rightarrow$  training error is always 0!

## Model Selection



## Support Vector Machine (SVM)

Theory elaborated in the early 1990's (Vapnik *et al*) based on the idea of '**maximum margin**'

- ▶ deterministic criterion learned on the training set ← **supervised classification**
- ☒ general, i.e. **model free**, linear classification rule
- ☒ classification rule is linear in a transformed space of higher (possible infinite) dimension than the original input feature/predictor space

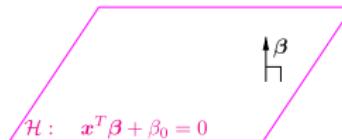
## Linear discrimination and Separating hyperplane

### Binary classification problem

- ▶  $X \in \mathbb{R}^p$
- ▶  $Y \in \{-1, 1\} \leftarrow 2 \text{ classes}$
- ▶ Training set  $(x_i, y_i)$ , for  $i = 1, \dots, n$

Defining a **linear** discriminant function  $h(x) \Leftrightarrow$  defining a separating hyperplane  $\mathcal{H}$  with equation

$$\mathbf{x}^T \boldsymbol{\beta} + \beta_0 = 0,$$

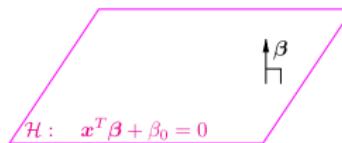


- ▶  $\boldsymbol{\beta} \in \mathbb{R}^p$  is the normal vector (vector normal to the hyperplane  $\mathcal{H}$ ),
- ▶  $\beta_0 \in \mathbb{R}$  is the intercept/offset (regression or geometrical interpretation)
- ☞  $\mathcal{H}$  is an *affine subspace* of dimension  $p - 1$
- ☞  $h(x) \equiv \mathbf{x}^T \boldsymbol{\beta} + \beta_0$  is the associated (linear) discriminant function

## Separating hyperplane and prediction rule

For a given separating hyperplane  $\mathcal{H}$  with equation

$$\mathbf{x}^T \boldsymbol{\beta} + \beta_0 = 0,$$



the **prediction rule** can be expressed as

- ▶  $\hat{y} = +1$ , if  $h(\mathbf{x}) = \mathbf{x}^T \boldsymbol{\beta} + \beta_0 \geq 0$ ,
- ▶  $\hat{y} = -1$ , otherwise,

or in an equivalent way :

$$\hat{y} \equiv G(\mathbf{x}) = \text{sign} [\mathbf{x}^T \boldsymbol{\beta} + \beta_0]$$

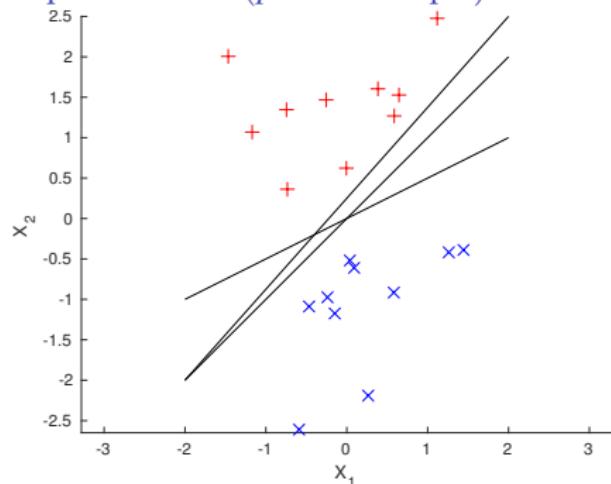
**Rk :**  $\mathbf{x}$  is in class  $y \in \{-1, 1\}$  : prediction  $G(\mathbf{x})$  is correct iff  
 $y (\mathbf{x}^T \boldsymbol{\beta} + \beta_0) \geq 0$

## Separating Hyperplane : separable case

**Linear separability assumption :**  $\exists \boldsymbol{\beta} \in \mathbb{R}^p$  and  $\beta_0 \in \mathbb{R}$  s.t. the hyperplane  $\mathbf{x}^T \boldsymbol{\beta} + \beta_0 = 0$  perfectly separates the two classes on the training set :

$$y_k \left( \mathbf{x}_k^T \boldsymbol{\beta} + \beta_0 \right) \geq 0, \quad \text{for } k = 1, \dots, n,$$

### Separable case ( $p = 2$ example)



Pb : infinitely many possible perfect separating hyperplanes

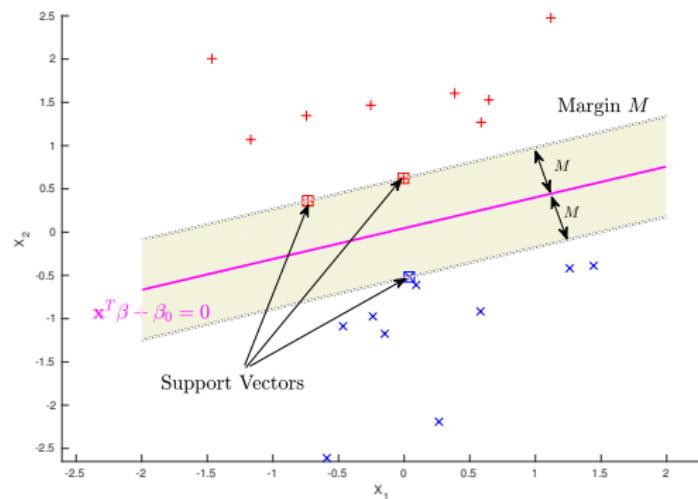
$$\mathbf{x}^T \boldsymbol{\beta} + \beta_0 = 0$$

- ➡ Find the 'optimal' separating hyperplane

## Maximum margin separating hyperplane (separable case)

### Maximum margin principle

We are interested in the 'optimal' perfect separating hyperplane maximizing the distance  $M > 0$ , called the **margin**, between the separating hyperplane and the training data, i.e. with the biggest gap



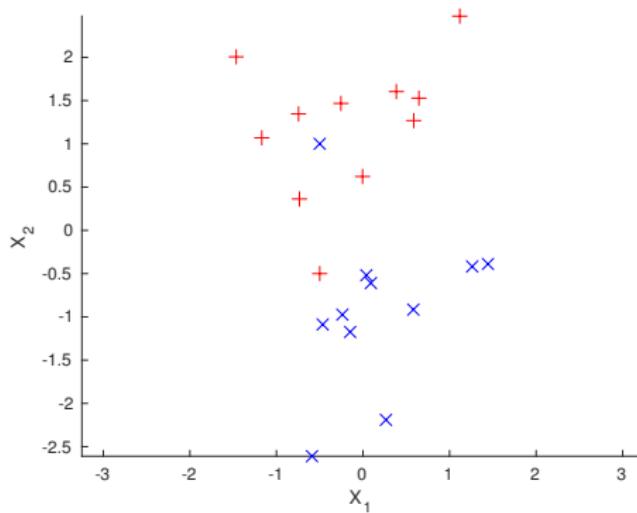
Find  $\beta \in \mathbb{R}^p$  and  $\beta_0 \in \mathbb{R}$  s.t.  
the margin

$$M = \min_{1 \leq k \leq n} \{d(x_k, \mathcal{H})\}$$

is maximized

## Nonseparable case

- ▶ in general, overlap of the 2 classes (unless  $n < p$ )
- ▶ no hyperplane that perfectly separates the training data



- ☞ we can soften what we mean by “separates”

## Maximum margin separating hyperplane (nonseparable case)

### Solution for the nonseparable case

Considering a *soft-margin* that allows wrong classifications

- ▶ introduction of *slack variables*  $\xi_i \geq 0$  s.t.

$$y_i(\mathbf{x}_i^T \boldsymbol{\beta} + \beta_0) \geq (1 - \xi_i)$$

Support vectors include now the wrong classified points, and the points inside the margins ( $\xi_i > 0$ )

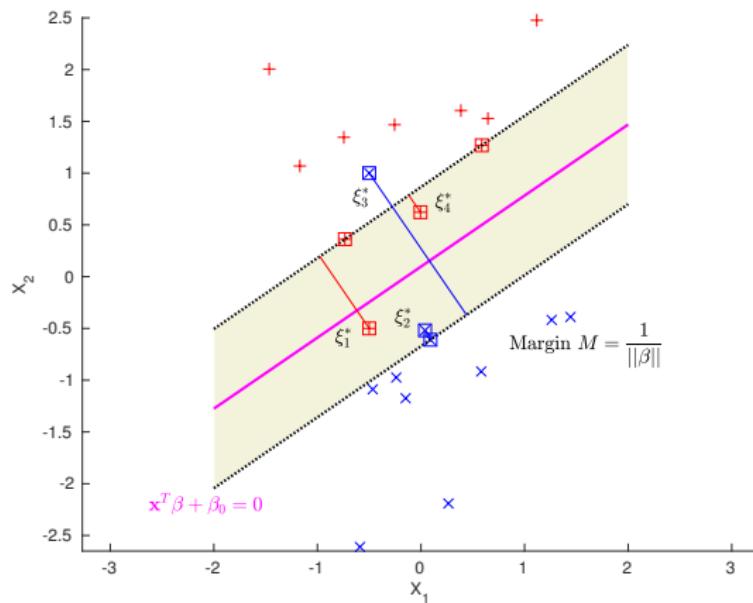
- ▶ Primal problem : adding a constraint on the  $\xi_i$ 's

$$\left\{ \begin{array}{ll} \max_{\boldsymbol{\beta}, \beta_0, \xi} & M, \\ \text{subject to} & y_i(\mathbf{x}_i^T \boldsymbol{\beta} + \beta_0) \geq 1 - \xi_i, \\ & \sum_{i=1}^n \xi_i \leq C. \end{array} \right.$$

where  $C > 0$  is the “cost” parameter

## Optimal separating hyperplane

Example (nonseparable case)

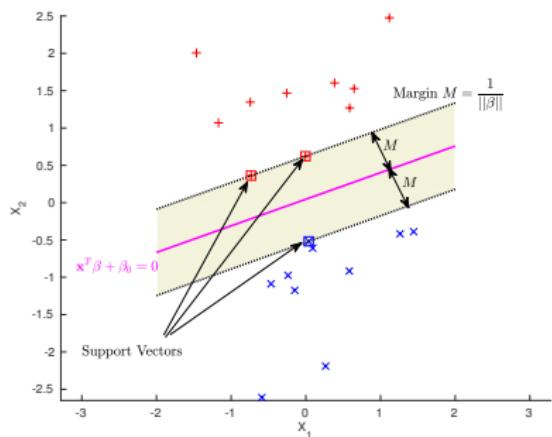


$\xi_i^* \equiv M\xi_i \leftarrow$  distance between a support vector and the margin

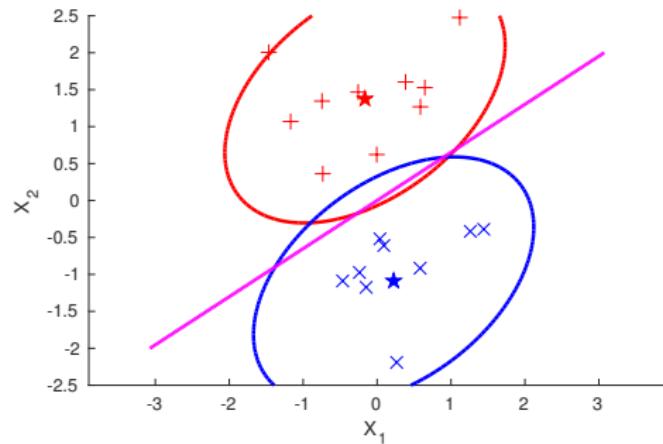
## Linear discrimination : SVM vs LDA

### Linear discrimination

- ▶ Linear Discriminant Analysis (LDA) : Gaussian generative model
- ▶ SVM : criterion optimization (maximizing the margin)



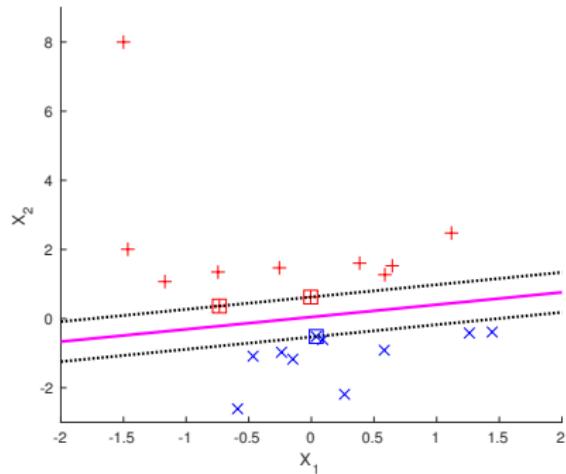
SVM



LDA

## Linear discrimination : SVM vs LDA (Cont'd)

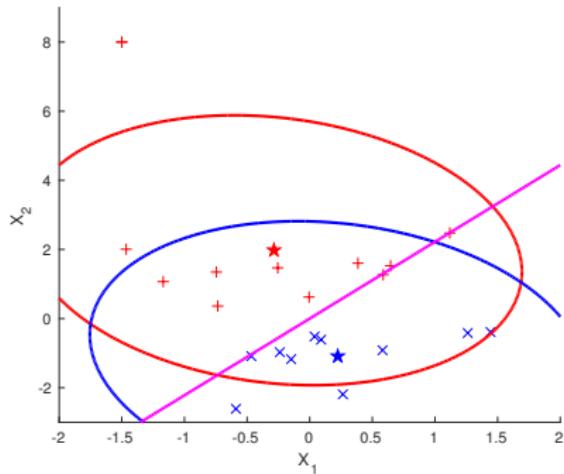
Adding one atypical data



SVM

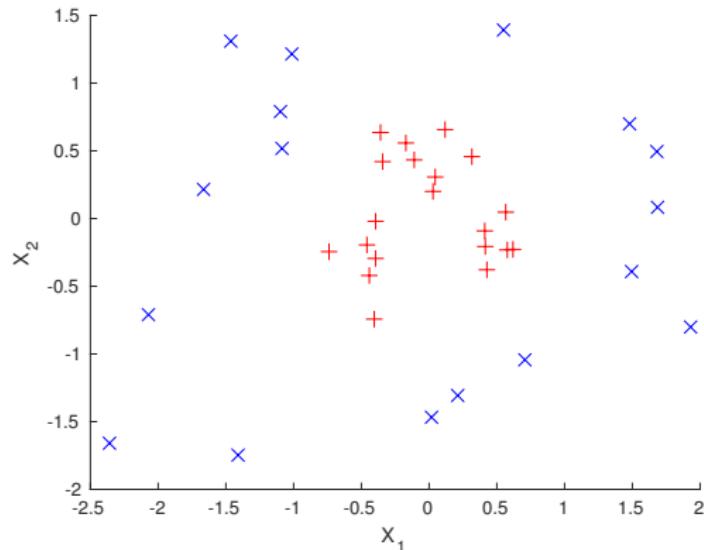
SVM property

- ▶ Nonsensitive to atypical points (outliers) far from the margin
- ☞ sparse method (information  $\equiv$  support vectors)



LDA

## Nonlinear discrimination in the input space

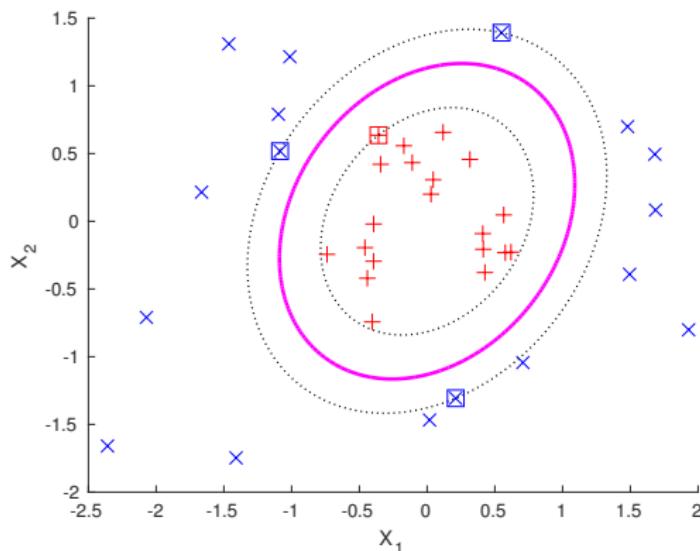


## Transformed space $\mathcal{F}$

- ▶ Choice of a transformed space  $\mathcal{F}$  (expansion space) where the linear separation assumption is more relevant
- ▶ Nonlinear expansion map  $\phi : \mathbb{R}^p \rightarrow \mathcal{F}, \mathbf{x} \mapsto \phi(\mathbf{x}) \leftarrow$  enlarged features

## Nonlinear discrimination in the input space

- $X \in \mathbb{R}^2$ ,  $\phi(x) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)^T$



Linear separation in the feature space  $\mathcal{F} \Rightarrow$  Nonlinear separation in the input space

## Kernel trick

The SVM solution depends only on the **inner product** between the input features  $\phi(\mathbf{x})$  and the support vectors  $\phi(\mathbf{x}_{\text{margin}})$

### Kernel trick

Use of a kernel function  $k$  associated with an expansion/feature map  $\phi$ :

$$\begin{aligned} k : \quad \mathbb{R}^p \times \mathbb{R}^p &\rightarrow \mathbb{R} \\ (\mathbf{x}, \mathbf{x}') &\mapsto k(\mathbf{x}, \mathbf{x}') \equiv \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle \end{aligned}$$

## Advantages

- ▶ computations are performed in the original input space : less expansive than in a high dimensional transformed space  $\mathcal{F}$
- ▶ explicit representations of the feature map  $\phi$  and enlarged feature space  $\mathcal{F}$  are not necessary, the only expression of  $k$  is required !
- ☞ possibility of complex transformations in possible infinite space  $\mathcal{F}$
- ☞ **standard trick** in machine learning not limited to SVM (kernel-PCA, gaussian process, kernel ridge regression, spectral clustering ...)

## Choosing the Kernel function

### Usual kernel functions

- ▶ Linear kernel ( $\mathcal{F} \equiv \mathbb{R}^p$ ) :  $k(x, x') = x^T x'$
- ▶ Polynomial kernel (dimension of  $\mathcal{F}$  increases with the order  $d$ )

$$k(x, x') = (x^T x')^d \quad \text{or} \quad (x^T x' + 1)^d$$

- ▶ Gaussian radial function ( $\mathcal{F}$  with infinite dimension)

$$k(x, x') = \exp(-\gamma \|x - x'\|^2)$$

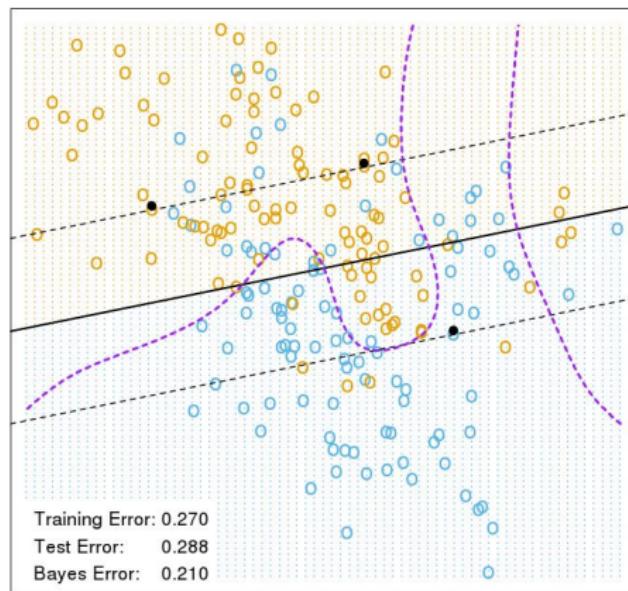
- ▶ Neural net kernel ( $\mathcal{F}$  with infinite dimension)

$$k(x, x') = \tanh(\kappa_1 x^T x' + \kappa_2)$$

- ☞ standard practice is to estimate optimal values of kernel parameters by cross validation

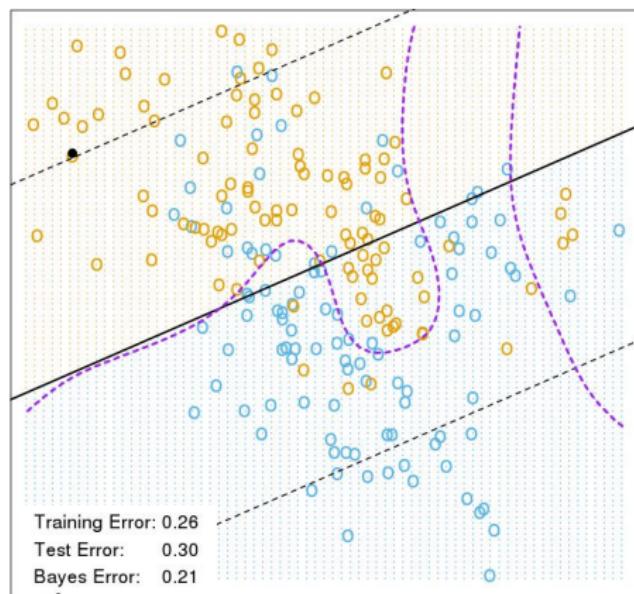
## Application : binary data (cf course 01)

Linear kernel



## Application : binary data (cf course 01)

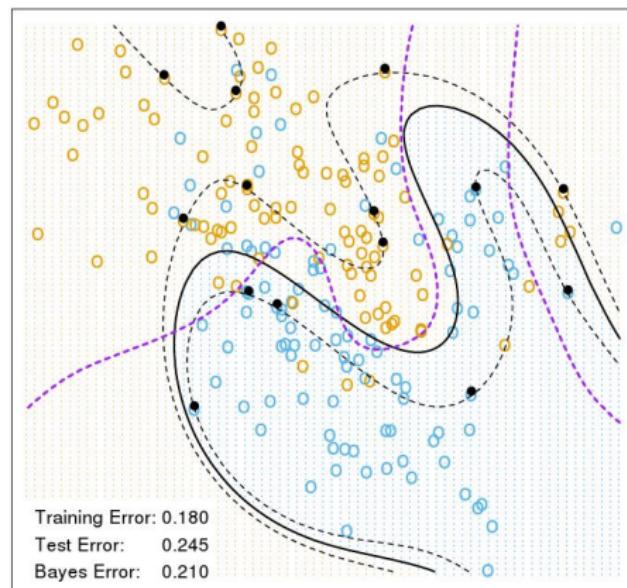
Linear kernel



$$C = 0.01$$

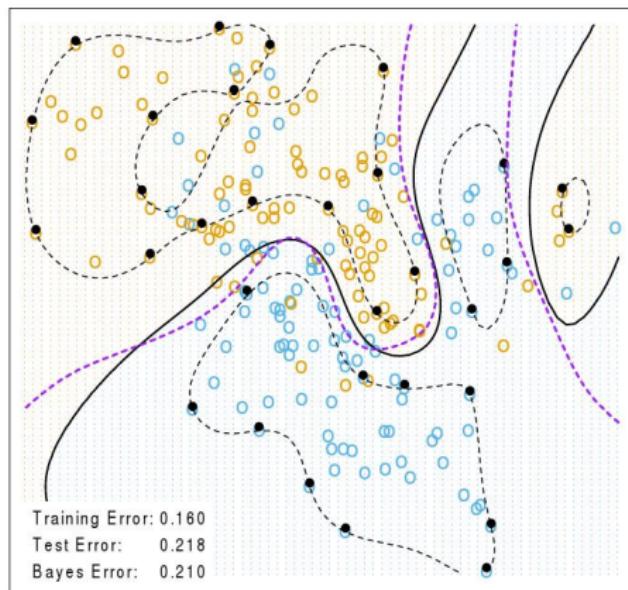
## Application : binary data (cf introduction course)

Polynomial kernel ( $d = 4$ )



## Application : binary data (cf course 01)

Gaussian radial kernel ( $\gamma = 1$ )



## Multiclass SVM

- ▶  $Y \in \{1, \dots, K\} \leftarrow K$  classes

Standard approach : direct generalization by using multiple binary SVMs

OVA : one-versus-all strategy

- ▶  $K$  classifiers between one class (+1 label) versus all the other classes (-1 label)
- ☞ classifier with the highest confidence value (e.g. the maximum distance to the separator hyperplane) assigns the class

OVO : one-versus-one strategy

- ▶  $\binom{K}{2} = K(K - 1)/2$  classifiers between every pair of classes
- ☞ majority vote rule : the class with the most votes determines the instance classification

Which to choose ? if  $K$  is not too large, choose OVO

## Conclusions on 'Black Box' approaches

### k-NN

- ▶ non-parametric method which does not rely on a fixed model
- ▶ algorithm which is conceptually among the simplest of all machine learning algorithms
- ▶ badly behaved procedure in high dimension : dimension reduction, e.g. PCA, is usually performed prior to k-NN algorithm in order to avoid curse of dimensionality and to reduce computational complexity of the classification rule

### SVM

- ▶ maximum margin learning criterion ← model free
- ▶ classification algorithm nonlinear in the original input space by performing an implicit linear classification in a higher dimensional space
- ▶ sparse solutions characterized by the support vectors
- ▶ popular algorithms, with a large literature

## Classification outline

Model based approaches for classification

Bayes Classifier

Linear/Quadratic Discriminant Analysis (LDA/QDA)

Black box approaches for classification

K Nearest Neighbors (K-NN)

Support Vector Machine (SVM)

Unsupervised classification : Clustering

K means

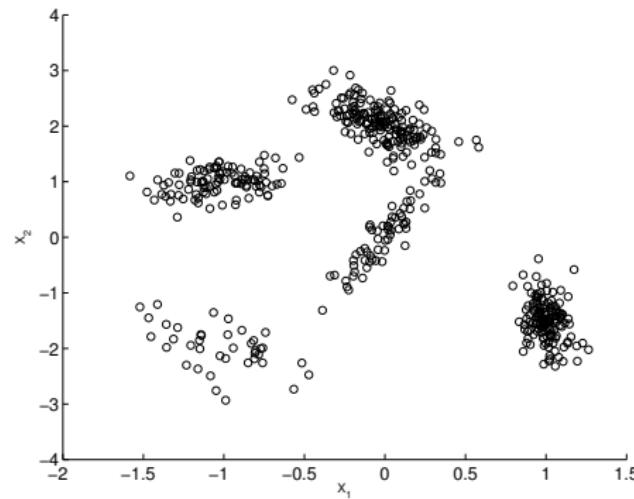
EM Algorithm for Gaussian Mixture Models

## Unsupervised classification

### Assumptions

- $X \in \mathbb{R}^p, Y \in \{1, \dots, K\} \leftarrow K$  classes
- Training set  $(x_1, \dots, x_n) \leftarrow$  unknown outputs  $y_i$

Example ( $p = 2$ )

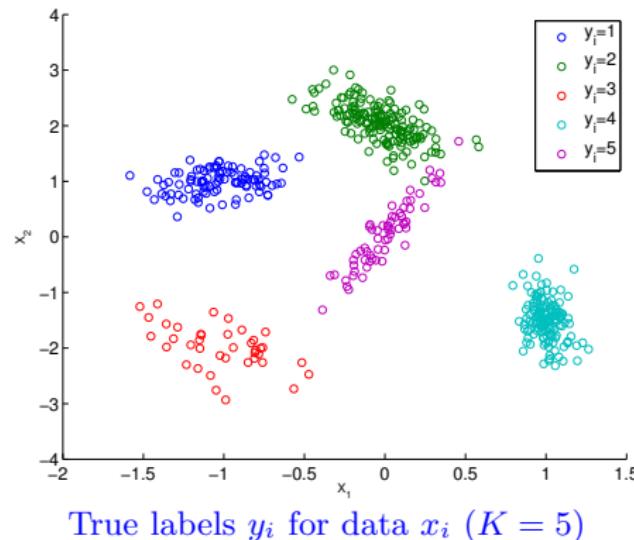


# Unsupervised classification : Clustering

## Objectives

- ▶ grouping similar data in the same cluster ← clustering
- ▣ For each  $x_i$ ,  $1 \leq i \leq n$ , predict the class variable  $Y_i \in \{1, \dots, K\}$

Example ( $p = 2$ )



## Clustering limitations

### Combinatorics problem

- ▶ Number of partitions into  $K$  classes for a sized  $n$  dataset : *Stirling number* of the 2nd kind  $S(n, K)$
- ▶ Number of partitions for a sized  $n$  dataset : *Bell number*  

$$B_n = \sum_{k=1}^n S(n, k)$$

dataset size $n$	2	5	10	100	200
$S(n, 2)$ ( $K = 2$ classes)	1	15	511	$6.3 \times 10^{29}$	$8.0 \times 10^{59}$
$S(n, 4)$ ( $K = 4$ classes)	0	10	34105	$6.7 \times 10^{58}$	$1.1 \times 10^{119}$
$B_n$	2	52	115975	$4.8 \times 10^{115}$	$6.2 \times 10^{275}$

- ▶ Remember  $\simeq 10^{80}$  atoms in the Universe...

Pb : Exhaustive search (brute-force) not possible in practice

☞ local search around initial solutions/values → sub-optimal

## Cost criterion : K-means clustering

### Notations

For a given clustering  $Y$ , let

- ▶  $n_k = \#\{i \mid Y_i = k\}$  is the size of the  $k$ th cluster,
- ▶  $\hat{\mu}_k = \frac{1}{n_k} \sum_{i|Y_i=k} x_i$  is the sample mean of the points assigned in the  $k$ th cluster

'Optimal' clustering  $Y$  defined as the solution of the following optimization problem :

### K-means cost criterion

$$\text{Minimize } J(Y) = \sum_{k=1}^K \sum_{i|Y_i=k} \|x_i - \hat{\mu}_k\|^2,$$

- ☞  $J(Y)$  is the sum of **within-cluster** dispersions

## K-means algorithm

- ▶ **Require :**  $K$  the number of clusters,
- ▶ **Initialization :** Set the centroid  $\mu_k$ ,  $1 \leq k \leq K$ , to a starting value  $\mu_k^{(0)}$ ,
- ▶ **For**  $t = 1 \rightarrow \dots$  **until convergence** (i.e.  $\mu_k^{(t)} = \mu_k^{(t-1)}$ )
  1. **Assignment step :** assign  $x_i$  to the class of the closest center

$$Y_i^{(t)} = \arg \min_{k=1, \dots, K} \|x_i - \mu_k^{(t-1)}\|^2, \quad \text{for } i = 1, \dots, n$$

2. **Update step :** update the centroids  $\mu_k$ , for  $k = 1, \dots, K$

$$\mu_k^{(t)} = \arg \min_{\mu_k} \sum_{i|Y_i^{(t)}=k} \|x_i - \mu_k\|^2 = \frac{1}{n_k^{(t)}} \sum_{i|Y_i^{(t)}=k} x_i,$$

i.e.  $\mu_k^{(t)}$  is the sample mean of the  $k$ th cluster

☞ hard-thresholding clustering

## Convergence of $K$ -means algorithm

### Convergence

- ▶ each step decreases the criterion,
- ▶ there is a (huge) finite number of partitions,
- ☞ the algorithm **converges** to a solution (in a finite number of steps)

**But** no guaranty of the solution optimality (depend on the initialization)...

### Stopping criterion

$K$ -means usually very fast for a small/moderate number of clusters  $K$ , but

- ▶ running time increases with the number of clusters  $K$
- ▶ in the worst case, can be very slow to converge even for  $K = 2$ ,

Thus, to shorten the computational time, the algorithm can be stopped when the cost criterion does not decrease significantly

## Variants/Improvements of $K$ -means algorithm

### Initialization heuristics

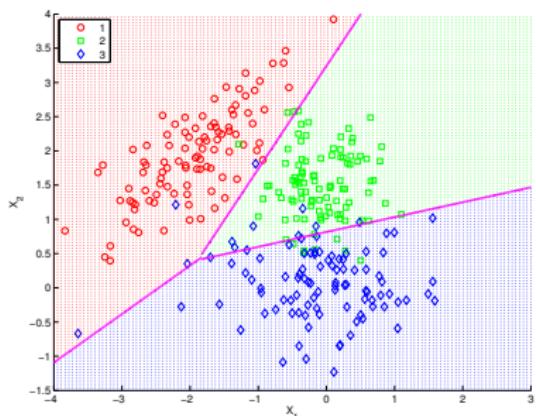
- ▶ Forgy method
  - ▶ pick randomly  $K$  observations from the dataset as initial centers,
  - ▶ run  $K$ -means algorithm with these starting values
  - ▶ repeat these 2 steps several times and retain the best (cost sense) clustering
- ▶ lot of variants : `Random partitions`, `k-means++`, `power init.`
  - ☞ may lower the computation time of one run,
  - ☞ can give some guarantees that the solution is competitive w.r.t. to the optimal one.

### Choice of the distance

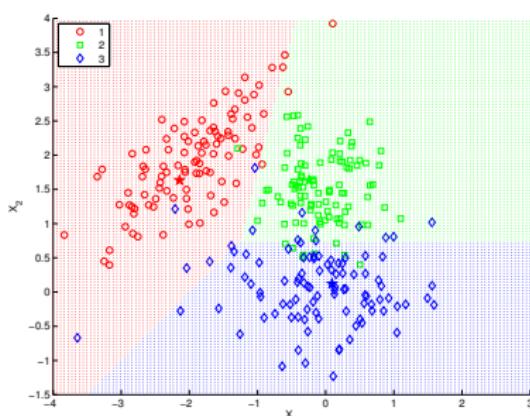
- ▶ Standard  $K$ -means based on the squared  $\ell_2$  (euclidean) distance.
- ▶ Other distance can be considered : e.g. using  $\ell_1$  distance yields the  $K$ -medians algorithm where the cluster centroid becomes the median

## K-means

## Prediction vs Clustering



LDA (supervised approach)

K-means with  $K = 3$  classes

- ▶ the points  $x_1, \dots, x_n$  are grouped according to the color of the regions
- ▶ Prediction : performance on *new* data is what matters
- ▶ Clustering : performance on *current* data is what matters

## Mixture of distributions

- ▶ Data  $X_1, \dots, X_n$  assumed to be i.i.d. with pdf  $f$
- ▶  $f$  is modeled as a *mixture of distributions*

$$f(x) = \sum_{k=1}^K \pi_k \phi(x; \theta_k)$$

- ▶  $\pi_1, \dots, \pi_k$  are the relative sizes ( $\sum_{k=1}^K \pi_k = 1$ ) of the classes :

$$\Pr(Y_i = k) = \pi_k$$

- ▶ density  $\phi$  is the parametric shape of a class,
- ▶ parameters  $\theta_1, \dots, \theta_K$  are the *centroids* of the classes/clusters

### Latent variable

$Y \in \{1, \dots, K\}$  indicating the class of the r.v.  $X$

- ▶  $Y \sim$  discrete distribution s.t.  $\Pr(Y_i = k) = \pi_k, \quad k = 1, \dots, K$
- ▶  $X|Y = k \sim$  distribution with pdf  $\phi(\cdot | \theta_k)$

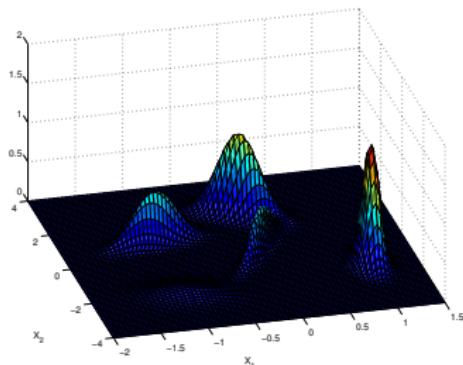
## Gaussian mixture model

- ▶ Class centroid :  $\theta = (\mu \leftarrow \text{mean}, \Sigma \leftarrow \text{covariance matrix})$
- ▶ Density  $\phi$  of a class : multivariate normal distribution  $\mathcal{N}(\mu, \Sigma)$  pdf

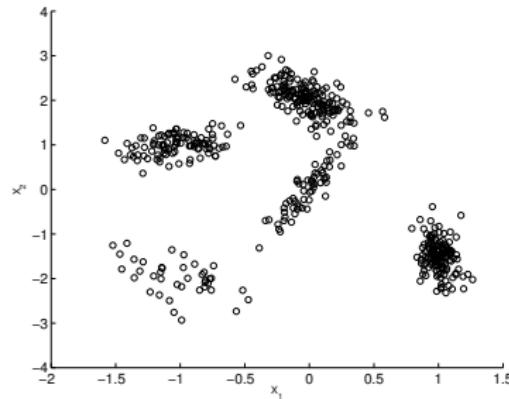
$$\phi(x; \mu, \Sigma) = (\det(2\pi\Sigma))^{-1/2} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right)$$

- ▶ Mixture density  $f(x) = \sum_{k=1}^K \pi_k \phi(x; \mu_k, \Sigma_k)$

Example ( $p = 2$ ,  $K = 5$ )



Mixture density  $f$



$n = 500$  realizations

## EM (Expectation-Maximization) algorithm

EM method is a general and important tool of statistical analysis :

- ▶ method for finding maximum likelihood (ML) estimates of parameters by maximizing **iteratively** the model likelihood
- ▶ **introduction** of unobserved **latent variables  $Z$**  to decompose the optimization problem in simpler sub-problems in an iterative way
- ▶ EM iteration **alternates** between performing an **expectation (E) step**, and a **maximization (M) step**

### Convergence

- ▶ EM : convergence toward a local maximum of the log-likelihood
- ☒ no guaranty of convergence toward the optimal solution (depend on the initial values)..

## EM algorithm for Gaussian mixture models

### EM clustering

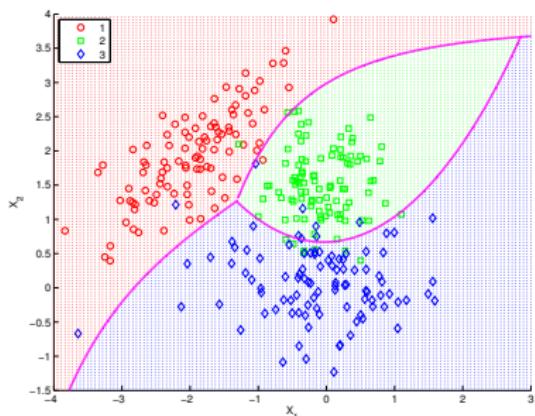
- ▶ Initialize  $\pi_k^{(0)}, \mu_k^{(0)}, \Sigma_k^{(0)}$ , for  $k = 1, \dots, K$
- ▶ For  $t = 1, \dots$  until convergence
  - (E) compute  $t_{ik}^{(t-1)} \equiv \Pr(Y_i = k | x_i, \theta^{(t-1)})$ , for  $i = 1, \dots, n, k = 1, \dots, K$
  - (M) compute  $\pi_k^{(t)}, \mu_k^{(t)}, \Sigma_k^{(t)}$ , for  $k = 1, \dots, K$

### Prediction/Correction structure

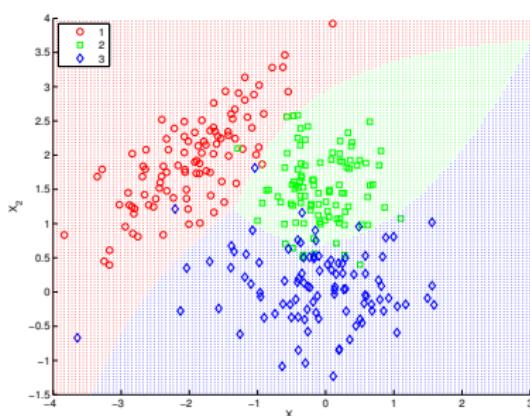
- ▶ E step  $\Leftrightarrow$  prediction step : computation of the posterior probability  $t_{ik}^{(t-1)}$  in  $\theta^{(t-1)}$
- ▶ M step  $\Leftrightarrow$  update/correction step : empirical averages weighted by the posterior probability  $t_{ik}^{(t-1)}$
- ☞ soft-thresholding algorithm

## Gaussian mixture model and EM algorithm

### Prediction vs Clustering



QDA (supervised approach)



EM with  $K = 3$  classes

- ▶ the points  $x_1, \dots, x_n$  are grouped according to the color of the regions
- ▶ Prediction : performance on *new* data is what matters
- ▶ Clustering : performance on *current* data is what matters