

# Advanced Computational Econometrics: Machine Learning

## Chapter 6: Gaussian Mixture models

Thierry Denœux

July-August 2019



# Overview

## 1 Introduction

- Gaussian Mixture Model
- Supervised vs. unsupervised learning
- Maximum likelihood estimation
- Reminder on the EM algorithm

## 2 Parameter estimation in GMMs

- Unsupervised learning
- Semi-supervised learning
- Mixture Discriminant Analysis

## 3 Regression models

- Mixture of regressions
- Mixture of experts



# Overview

## 1 Introduction

- Gaussian Mixture Model
- Supervised vs. unsupervised learning
- Maximum likelihood estimation
- Reminder on the EM algorithm

## 2 Parameter estimation in GMMs

- Unsupervised learning
- Semi-supervised learning
- Mixture Discriminant Analysis

## 3 Regression models

- Mixture of regressions
- Mixture of experts



## Return to LDA and QDA

- In LDA and QDA, we assume that the conditional density of  $X$  given  $Y = k$  is **multivariate Gaussian**

$$\phi_k(x; \mu_k, \Sigma_k) = \frac{1}{(2\pi)^{p/2} |\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k)\right).$$

(with  $\Sigma_k = \Sigma$  in the case of LDA)

- The marginal density of  $X$  is then a **mixture of  $c$  Gaussian densities**:

$$p(x) = \sum_{k=1}^c p(x|Y = k)P(Y = k) = \sum_{k=1}^c \pi_k \phi_k(x; \mu_k, \Sigma_k)$$

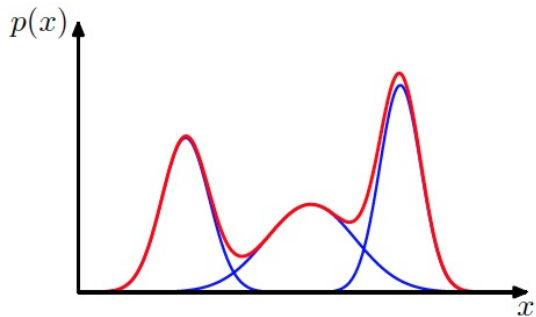
- This is called a **Gaussian Mixture Model (GMM)**.

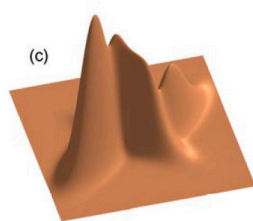
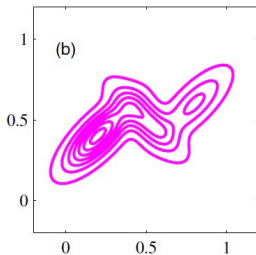
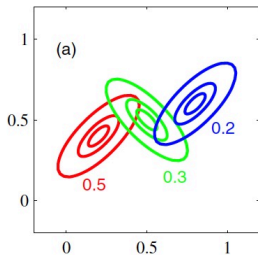


# Gaussian Mixture Models

- GMMs are widely used in Machine Learning for
  - Density estimation
  - Clustering (finding groups in data)
  - Classification (modeling complex-shaped class distributions)
  - Regression (accounting for different linear relations within subgroups of a population)
  - etc.



Example with  $p = 1$ 

Example with  $p = 2$ 

# How to generate data from a mixture?

- Assume  $X \sim \sum_{k=1}^c \pi_k \mathcal{N}(\mu_k, \Sigma_k)$
- How to generate  $X$ ?
  - 1 Generate  $Y \in \{1, \dots, c\}$  with probabilities  $\pi_1, \dots, \pi_c$ .
  - 2 If  $Y = k$ , generate  $X$  from  $p(x|Y = k) = \phi_k(x; \mu_k, \Sigma_k)$ .
- Remark: we can define mixtures of other distributions. In this chapter, we will focus (without loss of generality) on mixtures of normal distributions.





# Overview

## 1 Introduction

- Gaussian Mixture Model
- **Supervised vs. unsupervised learning**
- Maximum likelihood estimation
- Reminder on the EM algorithm

## 2 Parameter estimation in GMMs

- Unsupervised learning
- Semi-supervised learning
- Mixture Discriminant Analysis

## 3 Regression models

- Mixture of regressions
- Mixture of experts



# Supervised learning

- In discriminant analysis, we observe both the input vector  $X$  and the response (class label)  $Y$  for  $n$  individuals taken randomly from the population.
- The learning set has the form  $\mathcal{L}_s = \{(x_i, y_i)\}_{i=1}^n$ .
- Learning a classifier from such data is called **supervised learning**.

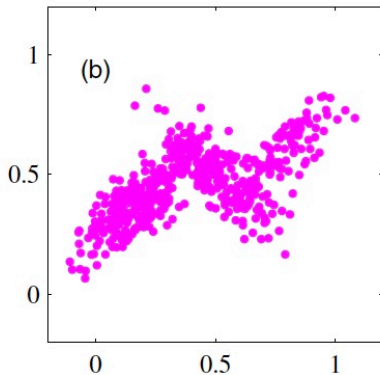
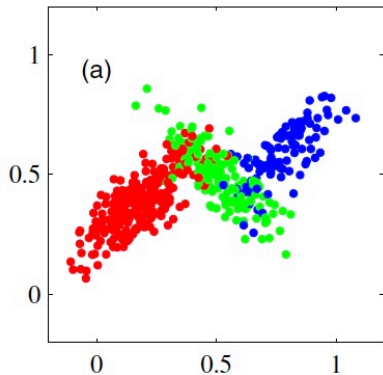


# Unsupervised learning

- In some situations, we observe  $X$ , but  $Y$  is not observed. We say that  $Y$  is a **latent variable**.
- The learning set has the form  $\mathcal{L}_{ns} = \{x_i\}_{i=1}^n$ .
- Estimating the model parameters from such data is called **unsupervised learning**.
- Applications: density estimation, clustering, feature extraction.
- Unsupervised learning is usually more difficult than supervised learning, because we have less information to estimate the parameters.



# Supervised vs. unsupervised learning



# Semi-supervised learning

- Sometimes, we collect a lot of data, but we can label only a part of them.
- Examples: image data from the web, or from sensors on a robot.
- The data then have the form  $\mathcal{L}_{ss} = \{(x_i, y_i)\}_{i=1}^{n_s} \cup \{x_i\}_{i=n_s+1}^n$ .
- This is called a **semi-supervised learning problem**.
- Semi-supervised learning is intermediate between supervised and unsupervised learning.



# Overview

## 1 Introduction

- Gaussian Mixture Model
- Supervised vs. unsupervised learning
- **Maximum likelihood estimation**
- Reminder on the EM algorithm

## 2 Parameter estimation in GMMs

- Unsupervised learning
- Semi-supervised learning
- Mixture Discriminant Analysis

## 3 Regression models

- Mixture of regressions
- Mixture of experts



## Maximum likelihood: supervised case

- In the case of supervised learning of GMMs, the maximum likelihood estimates (MLE) of  $\mu_k$ ,  $\Sigma_k$  and  $\pi_k$  have simple closed-form expressions.
- The likelihood function is

$$\begin{aligned} L(\theta; \mathcal{L}_S) &= \prod_{i=1}^n p(x_i, y_i) = \prod_{i=1}^n p(x_i | Y_i = y_i) p(Y_i = y_i) \\ &= \prod_{i=1}^n \prod_{k=1}^c \phi(x_i; \mu_k, \Sigma_k)^{y_{ik}} \pi_k^{y_{ik}} \end{aligned}$$

- The log-likelihood function is

$$\ell(\theta; \mathcal{L}_S) = \sum_{k=1}^c \left\{ \sum_{i=1}^n y_{ik} \log \phi(x_i; \mu_k, \Sigma_k) \right\} + \sum_{i=1}^n \sum_{k=1}^c y_{ik} \log \pi_k$$

- The parameters  $\mu_k$ ,  $\Sigma_k$  can be estimated separately using the data from class  $k$ .



# MLE in the supervised case

- We have

$$\sum_{i=1}^n y_{ik} \log \phi(x_i; \mu_k, \Sigma_k) = -\frac{1}{2} \sum_{i=1}^n y_{ik} (x_i - \mu_k)^T \Sigma_k^{-1} (x_i - \mu_k) - \frac{n_k}{2} \log |\Sigma_k| - \frac{n_k p}{2} \log(2\pi)$$

- The derivative wrt to  $\mu_k$  is  $\sum_i y_{ik} \Sigma_k^{-1} (x_i - \mu_k)$ . Hence,

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i=1}^n y_{ik} x_i$$

- It can be shown that

$$\hat{\Sigma}_k = \frac{1}{n_k} \sum_{i=1}^n y_{ik} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T$$





# MLE in the supervised case (continued)

- To find the MLE of the  $\pi_k$ , we maximize

$$\sum_{i=1}^n \sum_{k=1}^c y_{ik} \log \pi_k$$

wrt to  $\pi_k$ , subject to the constraint  $\sum_{k=1}^c \pi_k = 1$ .

- The solution is

$$\hat{\pi}_k = \frac{n_k}{n}, \quad k = 1, \dots, c$$



# Maximum likelihood: unsupervised case

- In the case of unsupervised learning, the log-likelihood function is

$$\begin{aligned}\ell(\theta; \mathcal{L}_{ns}) &= \sum_{i=1}^n \log p(x_i) \\ &= \sum_{i=1}^n \left( \log \sum_{k=1}^c \pi_k \phi_k(x_i; \mu_k, \Sigma_k) \right)\end{aligned}$$

- We can no longer separate the terms corresponding to each class.
- Maximizing the log-likelihood becomes a difficult nonlinear optimization problem, for which no closed-form solution exists.
- A powerful method: the **Expectation-Maximization (EM)** algorithm.



# Overview

## 1 Introduction

- Gaussian Mixture Model
- Supervised vs. unsupervised learning
- Maximum likelihood estimation
- **Reminder on the EM algorithm**

## 2 Parameter estimation in GMMs

- Unsupervised learning
- Semi-supervised learning
- Mixture Discriminant Analysis

## 3 Regression models

- Mixture of regressions
- Mixture of experts



# Notation

$\mathbf{X}$  : Observed variables

$\mathbf{Y}$  : Missing or latent variables

$\mathbf{Z}$  : Complete data  $\mathbf{Z} = (\mathbf{X}, \mathbf{Y})$

$\theta$  : Unknown parameter

$L(\theta)$  : observed-data likelihood, short for  $L(\theta; \mathbf{x}) = p(\mathbf{x}; \theta)$

$L_c(\theta)$  : complete-data likelihood, short for  $L(\theta; \mathbf{z}) = p(\mathbf{z}; \theta)$

$\ell(\theta), \ell_c(\theta)$  : observed and complete-data log-likelihoods



## Notation (continued)

- Suppose we seek to maximize  $L(\theta)$  with respect to  $\theta$ .
- Define  $Q(\theta; \theta^{(t)})$  to be the expectation of the complete-data log-likelihood, conditional on the observed data  $\mathbf{X} = \mathbf{x}$ . Namely

$$\begin{aligned} Q(\theta, \theta^{(t)}) &= \mathbb{E}_{\theta^{(t)}} \{ \ell_c(\theta) \mid \mathbf{x} \} \\ &= \mathbb{E}_{\theta^{(t)}} \{ \log p(\mathbf{Z}; \theta) \mid \mathbf{x} \} \\ &= \int [\log p(\mathbf{z}; \theta)] p(\mathbf{y}|\mathbf{x}; \theta^{(t)}) d\mathbf{y} \end{aligned}$$

where the last equation emphasizes that  $\mathbf{Y}$  is the only random part of  $\mathbf{Z}$  once we are given  $\mathbf{X} = \mathbf{x}$ .



# The EM Algorithm (reminder)

Start with  $\theta^{(0)}$  and set  $t = 0$ . Then

- 1 **E step:** Compute  $Q(\theta, \theta^{(t)})$ .
- 2 **M step:** Maximize  $Q(\theta, \theta^{(t)})$  with respect to  $\theta$ . Set  $\theta^{(t+1)}$  equal to the maximizer of  $Q$ .
- 3 Return to the E step and increment  $t$  unless a stopping criterion has been met, e.g.,

$$|\ell(\theta^{(t+1)}) - \ell(\theta^{(t)})| \leq \epsilon$$



# Overview

- 1 Introduction
  - Gaussian Mixture Model
  - Supervised vs. unsupervised learning
  - Maximum likelihood estimation
  - Reminder on the EM algorithm
- 2 Parameter estimation in GMMs
  - Unsupervised learning
  - Semi-supervised learning
  - Mixture Discriminant Analysis
- 3 Regression models
  - Mixture of regressions
  - Mixture of experts



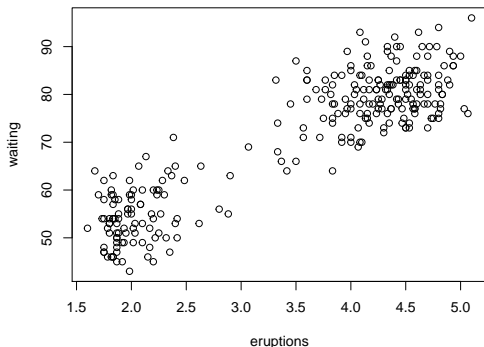
# Overview

- 1 Introduction
  - Gaussian Mixture Model
  - Supervised vs. unsupervised learning
  - Maximum likelihood estimation
  - Reminder on the EM algorithm
- 2 Parameter estimation in GMMs
  - Unsupervised learning
  - Semi-supervised learning
  - Mixture Discriminant Analysis
- 3 Regression models
  - Mixture of regressions
  - Mixture of experts





# Old Faithful geyser data



Waiting time between eruptions and duration of the eruption (in min) for the Old Faithful geyser in Yellowstone National Park, Wyoming, USA (272 observations).



# General GMM

- Let  $\mathbf{X} = (X_1, \dots, X_n)$  be an i.i.d. sample from a mixture of  $K$  multivariate normal distributions  $\mathcal{N}(\mu_k, \Sigma_k)$  with proportions  $\pi_k$ . The pdf of  $X_i$  is

$$p(x_i; \theta) = \sum_{k=1}^c \pi_k \phi(x_i; \mu_k, \Sigma_k),$$

where  $\theta$  is the vector of parameters.

- We introduce latent variables  $\mathbf{Y} = (Y_1, \dots, Y_n)$ , such that
  - $Y_i \sim \mathcal{M}(1, \pi_1, \dots, \pi_c)$ ,
  - $p(x_i | Y_i = k) = \phi(x_i; \mu_k, \Sigma_k)$ ,  $k = 1 \dots, c$



# Observed and complete-data likelihoods

- Observed-data likelihood:

$$L(\theta) = \prod_{i=1}^n p(x_i; \theta) = \prod_{i=1}^n \sum_{k=1}^c \pi_k \phi(x_i; \mu_k, \Sigma_k)$$

- Complete-data likelihood:

$$\begin{aligned} L_c(\theta) &= \prod_{i=1}^n p(x_i, y_i; \theta) = \prod_{i=1}^n p(x_i | y_i; \theta) p(y_i | \pi) \\ &= \prod_{i=1}^n \prod_{k=1}^c \phi(x_i; \mu_k, \Sigma_k)^{y_{ik}} \pi_k^{y_{ik}}, \end{aligned}$$

with  $y_{ik} = I(y_i = k)$ .



# Derivation of function $Q$

- Complete-data log-likelihood:

$$\ell_c(\theta) = \sum_{i=1}^n \sum_{k=1}^c y_{ik} \log \phi(x_i; \mu_k, \Sigma_k) + \sum_{i=1}^n \sum_{k=1}^c y_{ik} \log \pi_k$$

- It is linear in the  $y_{ik}$ . Consequently, the  $Q$  function is simply

$$Q(\theta, \theta^{(t)}) = \sum_{i=1}^n \sum_{k=1}^c y_{ik}^{(t)} \log \phi(x_i; \mu_k, \Sigma_k) + \sum_{i=1}^n \sum_{k=1}^c y_{ik}^{(t)} \log \pi_k$$

with  $y_{ik}^{(t)} = \mathbb{E}_{\theta^{(t)}}[Y_{ik} | x_i] = \mathbb{P}_{\theta^{(t)}}[Y_i = k | x_i]$ .



## EM algorithm

- E-step: compute

$$\begin{aligned} y_{ik}^{(t)} &= \mathbb{P}_{\theta^{(t)}}[Y_i = k | x_i] \\ &= \frac{\phi(x_i; \mu_k^{(t)}, \Sigma_k^{(t)}) \pi_k^{(t)}}{\sum_{\ell=1}^c \phi(x_i; \mu_\ell^{(t)}, \Sigma_\ell^{(t)}) \pi_\ell^{(t)}} \end{aligned}$$

- M-step: Maximize  $Q(\theta, \theta^{(t)})$ . We get

$$\pi_k^{(t+1)} = \frac{n_k^{(t)}}{n}, \quad \mu_k^{(t+1)} = \frac{1}{n_k^{(t)}} \sum_{i=1}^n y_{ik}^{(t)} x_i$$

$$\Sigma_k^{(t+1)} = \frac{1}{n_k^{(t)}} \sum_{i=1}^n y_{ik}^{(t)} (x_i - \mu_k^{(t+1)})(x_i - \mu_k^{(t+1)})^T$$

with  $n_k^{(t)} = \sum_{i=1}^n y_{ik}^{(t)}$ .



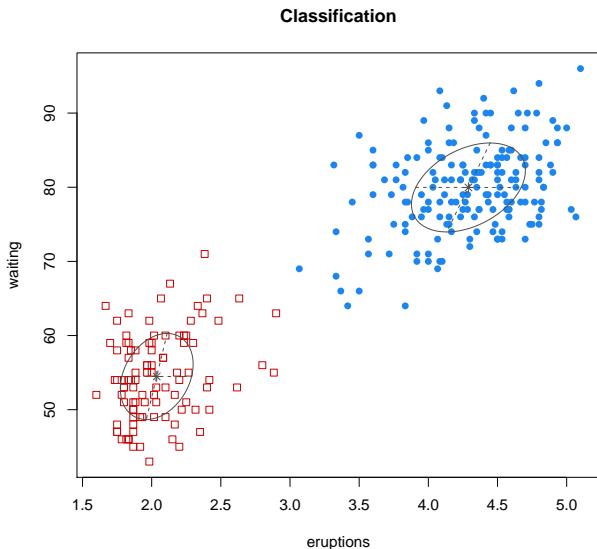
# GMM with the package mclust

```
library(mclust)
data(faithful)

faithfulMclust <- Mclust(faithful,G=2,modelNames="VVV")
plot(faithfulMclust)
```



## Result



## Choosing the number of clusters

- In clustering, selecting the number of clusters is often a difficult problem.
- This is a model selection problem. We can use the BIC criterion. (Reminder:  $BIC = -2\ell + d \log(N)$ )

```
> faithfulMclust <- Mclust(faithful,modelNames="VVV")
> summary(faithfulMclust)
```

```
-----
Gaussian finite mixture model fitted by EM algorithm
-----
```

Mclust VV (ellipsoidal, varying volume, shape, and orientation) model with 2 components:

log.likelihood	n	df	BIC	ICL
-1130.264	272	11	-2322.192	-2322.695

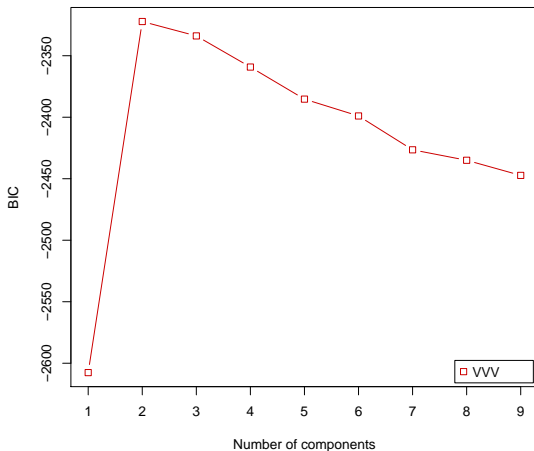
Clustering table:

1	2
175	97



# Choosing the number of clusters

```
plot(faithfulMclust)
```



# Reducing the number of parameters

- The general model has  $c[p + p(p + 1)/2 + 1] - 1$  parameters.
- When  $n$  is small and/or  $p$  is large: we need more **parsimonious** models (i.e., models with fewer parameters).
- Simple approaches:
  - Assume equal covariance matrix (homoscedasticity)
  - Assume the covariance matrices to be diagonal, or scalar
- More flexible approach: use the **eigendecomposition** of matrix  $\Sigma_k$



# Eigendecomposition $\Sigma_k$

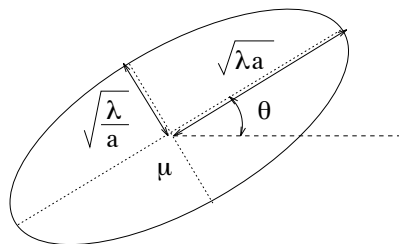
- As matrix  $\Sigma_k$  is symmetric, we can write

$$\Sigma_k = D_k \Lambda_k D_k^T = \lambda_k D_k A_k D_k^T,$$

where

- $\Lambda = \text{diag}(\lambda_{k1}, \dots, \lambda_{kp})$  is a diagonal matrix whose components are the **decreasing eigenvalues** of  $\Sigma_k$
- $D_k$  is an orthogonal matrix ( $D_k D_k^T = I$ ) whose columns are the **normalized eigenvectors** of  $\Sigma_k$
- $A_k$  is a diagonal matrix such that  $|A| = 1$ , with decreasing diagonal values proportional to the eigenvalues of  $\Sigma_k$
- $\lambda_k = \left( \prod_{j=1}^p \lambda_{kj} \right)^{1/p} = |\Sigma_k|^{1/p}$
- Interpretation:
  - $A_k$  describes the **shape** of the cluster
  - $D_k$  (a rotation matrix) describes its **orientation**
  - $\lambda_k$  describes its **volume**



Example in  $\mathbb{R}^2$ 

$$A = \begin{bmatrix} a & 0 \\ 0 & 1/a \end{bmatrix} \quad D = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix}$$

- $D$ : rotation matrix, angle  $\theta$
- $A$ : diagonal matrix with diagonal terms  $a$  and  $1/a$

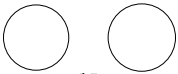
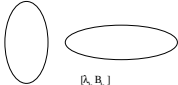







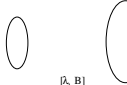


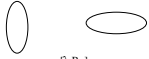



# Parsimonious models

- With this parametrization, the parameters of the GMM are: the centers, volumes, shapes, orientations and proportions.
- 28 different models
  - Spherical, diagonal, arbitrary
  - Volumes equal or not
  - Shapes equal or not
  - Directions equal or not
  - Proportions equal or not



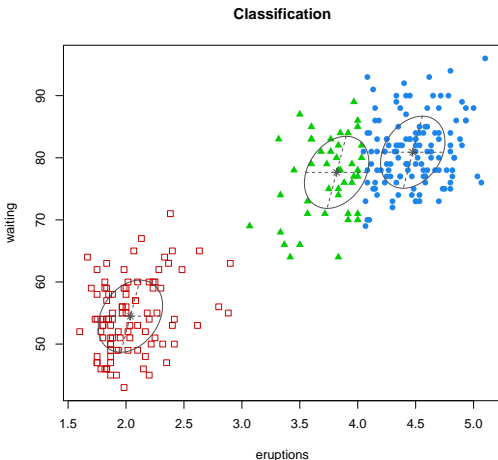
# The 14 models based on assumptions on variance matrices

 $[\lambda I]$	 $[\lambda_x B_x I]$	 $[\lambda D_x A D_x']$
 $[\lambda_x I]$	 $[\lambda D A D']$	 $[\lambda_x D_x A D_x']$
 $[\lambda B]$	 $[\lambda_x D A D']$	 $[\lambda D_x A_x D_x']$
 $[\lambda_x B]$	 $[\lambda D A_x D']$	 $[\lambda_x D_x A_x D_x']$
 $[\lambda B_x]$	 $[\lambda_x D A_x D']$	



# Best model

Best model: EEE or  $\lambda DAD^T$  (ellipsoidal, equal volume, shape and orientation) model with 3 components





# Overview

- 1 Introduction
  - Gaussian Mixture Model
  - Supervised vs. unsupervised learning
  - Maximum likelihood estimation
  - Reminder on the EM algorithm
- 2 Parameter estimation in GMMs
  - Unsupervised learning
  - **Semi-supervised learning**
  - Mixture Discriminant Analysis
- 3 Regression models
  - Mixture of regressions
  - Mixture of experts



# Semi-supervised learning

- In **semi-supervised learning**, the data have the form

$$\mathcal{L}_{SS} = \{(x_i, y_i)\}_{i=1}^{n_s} \cup \{x_i\}_{i=n_s+1}^n.$$

- Observed-data likelihood:

$$\begin{aligned} L(\theta) &= \prod_{i=1}^{n_s} p(x_i, y_i; \theta) \prod_{i=n_s+1}^n p(x_i; \theta) \\ &= \left( \prod_{i=1}^{n_s} \prod_{k=1}^c \phi(x_i; \mu_k, \Sigma_k)^{y_{ik}} \pi_k^{y_{ik}} \right) \left( \prod_{i=n_s+1}^n \sum_{k=1}^c \pi_k \phi(x_i; \mu_k, \Sigma_k) \right) \end{aligned}$$

- Complete-data likelihood: same as in the unsupervised case,

$$L_c(\theta) = \prod_{i=1}^n \prod_{k=1}^c \phi(x_i; \mu_k, \Sigma_k)^{y_{ik}} \pi_k^{y_{ik}},$$

with  $y_{ik} = 1$  if  $y_i = k$  and  $y_{ik} = 0$  otherwise.



# EM algorithm

- E-step: compute

$$y_{ik}^{(t)} = \begin{cases} y_{ik} & i = 1, \dots, n_s \text{ (fixed)} \\ \frac{\phi(x_i; \mu_k^{(t)}, \Sigma_k^{(t)}) \pi_k^{(t)}}{\sum_{\ell=1}^c \phi(x_i; \mu_\ell^{(t)}, \Sigma_\ell^{(t)}) \pi_\ell^{(t)}}, & i = n_s + 1, \dots, n \end{cases}$$

- M-step: same as in the unsupervised case.

$$\pi_k^{(t+1)} = \frac{n_k^{(t)}}{n}, \quad \mu_k^{(t+1)} = \frac{1}{n_k^{(t)}} \sum_{i=1}^n y_{ik}^{(t)} x_i$$

$$\Sigma_k^{(t+1)} = \frac{1}{n_k^{(t)}} \sum_{i=1}^n y_{ik}^{(t)} (x_i - \mu_k^{(t+1)})(x_i - \mu_k^{(t+1)})^T$$

with  $n_k^{(t)} = \sum_{i=1}^n y_{ik}^{(t)}$



# Package upclass

```
library(upclass)

data(iris)
X <- as.matrix(iris[,-5])
cl <- as.matrix(iris[,5])
indtrain <- sort(sample(1:150,110))
Xtrain <- X[indtrain,]
cltrain <- cl[indtrain]
indtest <- setdiff(1:150, indtrain)
Xtest <- X[indtest,]
models <- c("EII", "VII", "VEI","EVI")

fitupmodels <- upclassify(Xtrain,cltrain,Xtest,modelscope=models)
fitupmodels$Best$modelName # What is the best model?
```



# Overview

- 1 Introduction
  - Gaussian Mixture Model
  - Supervised vs. unsupervised learning
  - Maximum likelihood estimation
  - Reminder on the EM algorithm
- 2 Parameter estimation in GMMs
  - Unsupervised learning
  - Semi-supervised learning
  - **Mixture Discriminant Analysis**
- 3 Regression models
  - Mixture of regressions
  - Mixture of experts



# Mixture Discriminant Analysis

- GMM can also be useful in supervised classification.
- Here, we model the distribution of  $X$  in each class by a GMM:

$$p(x|Y = k) = \sum_{r=1}^{R_k} \pi_{kr} \phi(x; \mu_{kr}, \Sigma_{kr})$$

with  $\sum_{r=1}^{R_k} \pi_{kr} = \pi_k$ .

- This method is called **Mixture Discriminant Analysis (MDA)**. It extends LDA.
- By varying the number of components in each mixture, we can handle classes of any shape, and obtain arbitrarily complex nonlinear decision boundaries.
- We may impose  $\Sigma_{kr} = \Sigma$ , or any other parsimonious model, to control the complexity of the model.



# Observed-data likelihood

- Observed-data likelihood:

$$\begin{aligned}
 L(\theta) &= \prod_{i=1}^n p(x_i, y_i; \theta) = \prod_{i=1}^n p(x_i | y_i; \theta) p(y_i; \theta) \\
 &= \prod_{i=1}^n \prod_{k=1}^c \left( \sum_{r=1}^{R_k} \pi_{kr} \phi(x; \mu_{kr}, \Sigma_{kr}) \right)^{y_{ik}} \pi_k^{y_{ik}}
 \end{aligned}$$

- Again, the EM algorithm can be used to estimate the model parameters, separately in each class (see ESL page 449 for details).



# MDA using package mclust

```
odd <- seq(from = 1, to = nrow(iris), by = 2)
even <- odd + 1
X.train <- iris[odd,-5]
Class.train <- iris[odd,5]
X.test <- iris[even,-5]
Class.test <- iris[even,5]

# general covariance structure selected by BIC
irisMclustDA <- MclustDA(X.train, Class.train)
summary(irisMclustDA, newdata = X.test, newclass = Class.test)

plot(irisMclustDA)
```





## Result

```
> summary(irisMclustDA, newdata = X.test, newclass = Class.test)
```

```
-----
Gaussian finite mixture model for classification
-----
```

MclustDA model summary:

```
log.likelihood n df      BIC
-63.55015 75 53 -355.9272
```

```
Classes      n Model G
setosa      25 VEI 2
versicolor 25 EEV 2
virginica   25 XXX 1
```

Training classification summary:

Class	Predicted		
	setosa	versicolor	virginica
setosa	25	0	0
versicolor	0	25	0
virginica	0	0	25

Training error = 0

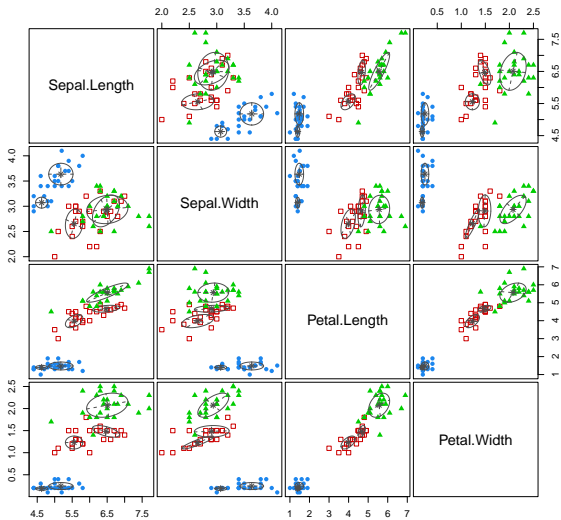
Test classification summary:

Class	Predicted		
	setosa	versicolor	virginica
setosa	25	0	0
versicolor	0	24	1
virginica	0	0	25

Test error = 0.01333333



## Result



# Overview

## 1 Introduction

- Gaussian Mixture Model
- Supervised vs. unsupervised learning
- Maximum likelihood estimation
- Reminder on the EM algorithm

## 2 Parameter estimation in GMMs

- Unsupervised learning
- Semi-supervised learning
- Mixture Discriminant Analysis

## 3 Regression models

- Mixture of regressions
- Mixture of experts

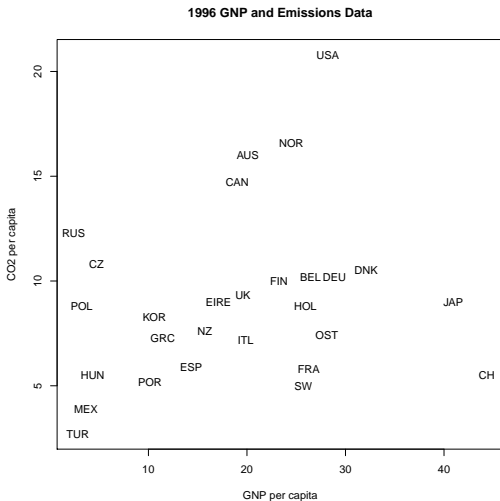


# Overview

- 1 Introduction
  - Gaussian Mixture Model
  - Supervised vs. unsupervised learning
  - Maximum likelihood estimation
  - Reminder on the EM algorithm
- 2 Parameter estimation in GMMs
  - Unsupervised learning
  - Semi-supervised learning
  - Mixture Discriminant Analysis
- 3 Regression models
  - Mixture of regressions
  - Mixture of experts



# Introductory example



# Introductory example (continued)

- The data in the previous slide do not show any clear linear trend.
- However, there seem to be several groups for which a linear model would be a reasonable approximation.
- How to identify those groups and the corresponding linear models?



# Formalization

- We assume that the response variable  $Y$  depends on the input variable  $X$  in different ways, depending on a latent variable  $Z$ . (Beware: we have switched back to the classical notation for regression models!)
- This model is called **mixture of regressions** or **switching regressions**. It has been widely studied in the econometrics literature.



# Model

- Model:

$$Y = \begin{cases} \beta_1^T X + \epsilon_1, \epsilon_1 \sim \mathcal{N}(0, \sigma_1) & \text{if } Z = 1, \\ \vdots \\ \beta_K^T X + \epsilon_K, \epsilon_K \sim \mathcal{N}(0, \sigma_K) & \text{if } Z = c, \end{cases}$$

with  $X = (1, X_1, \dots, X_p)$ , and

$$\mathbb{P}(Z = k) = \pi_k, \quad k = 1, \dots, c.$$

- So,

$$p(y|X = x) = \sum_{k=1}^c \pi_k \phi(y; \beta_k^T x, \sigma_k)$$





# Observed and complete-data likelihoods

- Observed-data likelihood:

$$L(\theta) = \prod_{i=1}^n p(y_i; \theta) = \prod_{i=1}^n \sum_{k=1}^c \pi_k \phi(y_i; \beta_k^T x_i, \sigma_k)$$

- Complete-data likelihood:

$$\begin{aligned} L_c(\theta) &= \prod_{i=1}^n p(y_i, z_i; \theta) = \prod_{i=1}^n p(y_i | z_i; \theta) p(z_i | \pi) \\ &= \prod_{i=1}^n \prod_{k=1}^c \phi(y_i; \beta_k^T x_i, \sigma_k)^{z_{ik}} \pi_k^{z_{ik}}, \end{aligned}$$

with  $z_{ik} = 1$  if  $z_i = k$  and  $z_{ik} = 0$  otherwise.



# Derivation of function $Q$

- Complete-data log-likelihood:

$$\ell_c(\theta) = \sum_{i=1}^n \sum_{k=1}^c z_{ik} \log \phi(y_i; \beta_k^T x_i, \sigma_k) + \sum_{i=1}^n \sum_{k=1}^c z_{ik} \log \pi_k$$

- It is linear in the  $z_{ik}$ . Consequently, the  $Q$  function is simply

$$Q(\theta, \theta^{(t)}) = \sum_{i=1}^n \sum_{k=1}^c z_{ik}^{(t)} \log \phi(y_i; \beta_k^T x_i, \sigma_k) + \sum_{i=1}^n \sum_{k=1}^c z_{ik}^{(t)} \log \pi_k$$

with  $z_{ik}^{(t)} = \mathbb{E}_{\theta^{(t)}}[Z_{ik}|y_i] = \mathbb{P}_{\theta^{(t)}}[Z_i = k|y_i]$ .



# EM algorithm

- E-step: compute

$$\begin{aligned} z_{ik}^{(t)} &= \mathbb{P}_{\theta^{(t)}}[Z_i = k | y_i] \\ &= \frac{\phi(y_i; \beta_k^{(t)T} x_i, \sigma_k^{(t)}) \pi_k^{(t)}}{\sum_{\ell=1}^c \phi(y_i; \beta_\ell^{(t)T} x_i, \sigma_\ell^{(t)}) \pi_\ell^{(t)}} \end{aligned}$$

- M-step: Maximize  $Q(\theta, \theta^{(t)})$ . As before, we get

$$\pi_k^{(t+1)} = \frac{n_k^{(t)}}{n},$$

with  $n_k^{(t)} = \sum_{i=1}^n z_{ik}^{(t)}$ .



## M-step: update of the $\beta_k$ and $\sigma_k$

- In  $Q(\theta, \theta^{(t)})$ , the term depending on  $\beta_k$  is

$$SS_k = \sum_{i=1}^n z_{ik}^{(t)} (y_i - \beta_k^T x_i)^2.$$

- Minimizing  $SS_k$  w.r.t.  $\beta_k$  is a weighted least-squares (WLS) problem. In matrix form,

$$SS_k = (\mathbf{y} - \mathbf{X}\beta_k)^T \mathbf{W}_k (\mathbf{y} - \mathbf{X}\beta_k),$$

where  $\mathbf{W}_k = \text{diag}(z_{1k}^{(t)}, \dots, z_{nk}^{(t)})$  is a diagonal matrix of size  $n$ .



# M-step: update of the $\beta_k$ and $\sigma_k$ (continued)

- The solution is the WLS estimate of  $\beta_k$ :

$$\beta_k^{(t+1)} = (\mathbf{X}^T \mathbf{W}_k \mathbf{X})^{-1} \mathbf{X}^T \mathbf{W}_k \mathbf{y}$$

- The value of  $\sigma_k$  minimizing  $Q(\theta, \theta^{(t)})$  is the weighted average of the residuals,

$$\begin{aligned} \sigma_k^{2(t+1)} &= \frac{1}{n_k^{(t)}} \sum_{i=1}^n z_{ik}^{(t)} (y_i - \beta_k^{(t+1)T} \mathbf{x}_i)^2 \\ &= \frac{1}{n_k^{(t)}} (\mathbf{y} - \mathbf{X} \beta_k^{(t+1)})^T \mathbf{W}_k (\mathbf{y} - \mathbf{X} \beta_k^{(t+1)}) \end{aligned}$$



# Mixture of regressions using mixtools

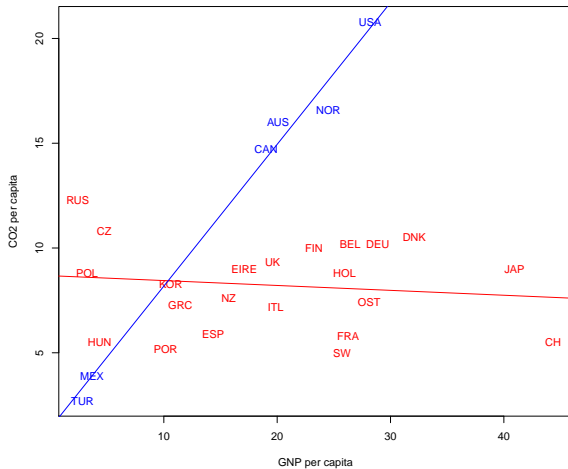
```
library(mixtools)
data(CO2data)
attach(CO2data)

CO2reg <- regmixEM(CO2, GNP)
summary(CO2reg)

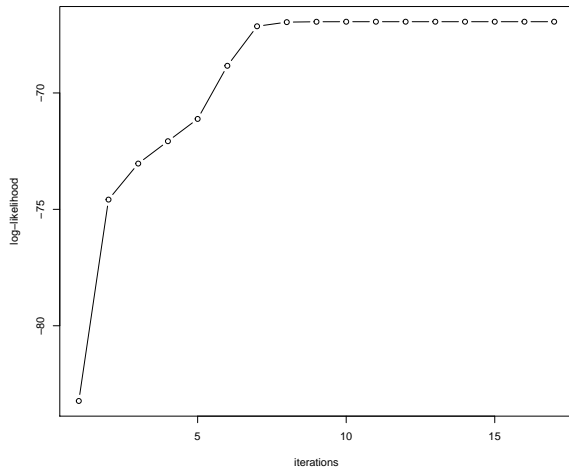
ii1<-CO2reg$posterior>0.5
ii2<-CO2reg$posterior<=0.5
text(GNP[ii1],CO2[ii1],country[ii1],col='red')
text(GNP[Cii2],CO2[ii2],country[ii2],col='blue')
abline(CO2reg$beta[,1],col='red')
abline(CO2reg$beta[,2],col='blue')
```



## Best solution in 10 runs

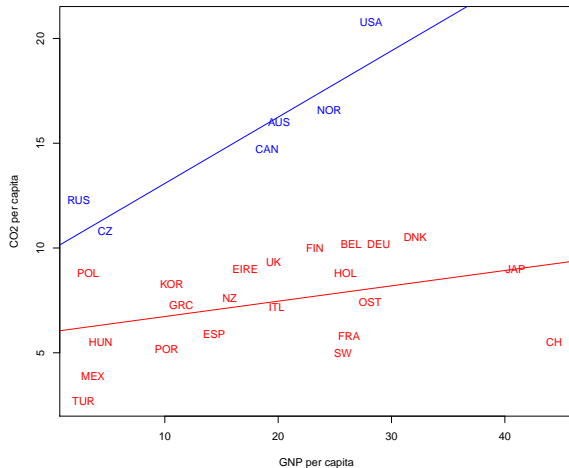


# Increase of log-likelihood

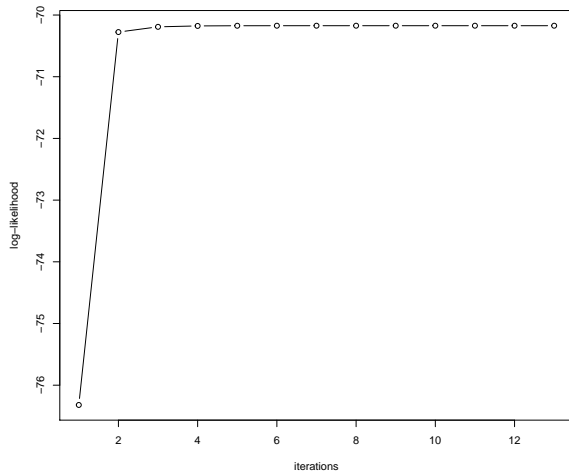




# Another solution (with lower log-likelihood)



# Increase of log-likelihood



# Overview

- 1 Introduction
  - Gaussian Mixture Model
  - Supervised vs. unsupervised learning
  - Maximum likelihood estimation
  - Reminder on the EM algorithm
- 2 Parameter estimation in GMMs
  - Unsupervised learning
  - Semi-supervised learning
  - Mixture Discriminant Analysis
- 3 Regression models
  - Mixture of regressions
  - Mixture of experts



# Making the mixing proportions predictor-dependent

- An interesting extension of the previous model is to assume the proportions  $\pi_k$  to be partially explained by a vector of **concomitant variables**  $W$ .
- If  $W = X$ , we can approximate the regression function by different linear functions in different regions of the predictor space.
- In ML, this method is referred to as the **mixture of experts** method.
- A useful parametric form for  $\pi_k$  that ensures  $\pi_k \geq 0$  and  $\sum_{k=1}^c \pi_k = 1$  is the multinomial logit (softmax) model

$$\pi_k(w, \alpha) = \frac{\exp(\alpha_k^T w)}{\sum_{l=1}^c \exp(\alpha_l^T w)}$$

with  $\alpha = (\alpha_1, \dots, \alpha_c)$  and  $\alpha_1 = 0$ .



# EM algorithm

- The  $Q$  function is the same as before, except that the  $\pi_k$  now depend on the  $w_i$  and parameter  $\alpha$ :

$$Q(\theta, \theta^{(t)}) = \sum_{i=1}^n \sum_{k=1}^c z_{ik}^{(t)} \log \phi(y_i; \beta_k^T x_i, \sigma_k) + \sum_{i=1}^n \sum_{k=1}^c z_{ik}^{(t)} \log \pi_k(w_i, \alpha)$$

- In the M-step, the update formula for  $\beta_k$  and  $\sigma_k$  are unchanged.
- The last term of  $Q(\theta, \theta^{(t)})$  can be maximized w.r.t.  $\alpha$  using an iterative algorithm, such as the Newton-Raphson procedure. (See remark on next slide)



# Generalized EM algorithm

- To ensure convergence of EM, we only need to increase (but not necessarily maximize)  $Q(\theta, \theta^{(t)})$  at each step.
- Any algorithm that chooses  $\theta^{(t+1)}$  at each iteration to guarantee the above condition (without maximizing  $Q(\theta, \theta^{(t)})$ ) is called a **Generalized EM (GEM) algorithm**.
- Here, we can perform a single step of the Newton-Raphson algorithm to maximize

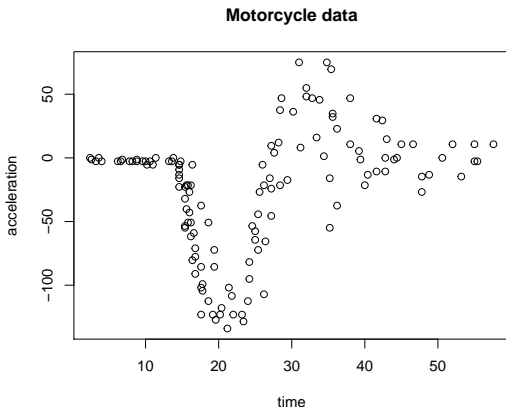
$$\sum_{i=1}^n \sum_{k=1}^c z_{ik}^{(t)} \log \pi_k(w_i, \alpha)$$

with respect to  $\alpha$ .

- Backtracking can be used to ensure ascent.



# Example: motorcycle data



```
library('MASS')  
x<-mcycle$times  
y<-mcycle$accel  
plot(x,y)
```



# Mixture of experts using flexmix

```
library(flexmix)

K<-5
res<-flexmix(y ~ x,k=K,model=FLXMRglm(family="gaussian"),
concomitant=FLXPmultinom(formula=~x))

beta<- parameters(res)[1:2,]
alpha<-res@concomitant@coef
```





# Plotting the posterior probabilities

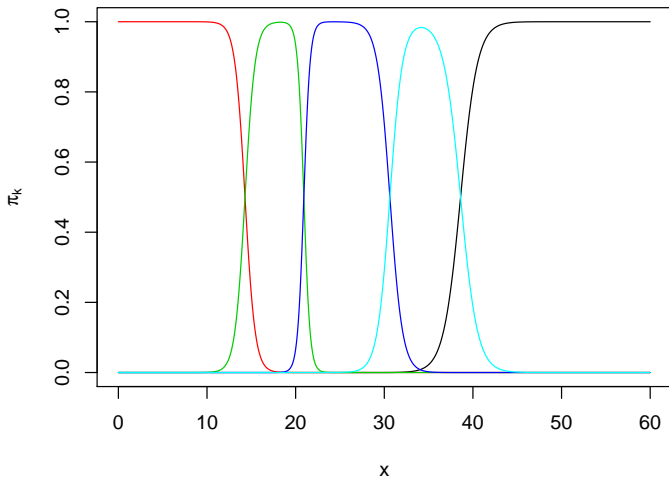
```
xt<-seq(0,60,0.1)
Nt<-length(xt)
plot(x,y)
pit=matrix(0,Nt,K)
for(k in 1:K) pit[,k]<-exp(alpha[1,k]+alpha[2,k]*xt)
pit<-pit/rowSums(pit)

plot(xt,pit[,1],type="l",col=1)
for(k in 2:K) lines(xt,pit[,k],col=k)
```



# Posterior probabilities

## Motorcycle data – posterior probabilities



# Plotting the predictions

```
yhat<-rep(0,Nt)
for(k in 1:K) yhat<-yhat+pit[,k]*(beta[1,k]+beta[2,k]*xt)

plot(x,y,main="Motorcycle data",xlab="time",ylab="acceleration")
for(k in 1:K) abline(beta[1:2,k],lty=2)
lines(xt,yhat,col='red',lwd=2)
```



# Regression lines and predictions

