Machine Learning & Pattern Recognition

SONG Xuemeng

songxuemeng@sdu.edu.cn

http://xuemeng.bitcron.com/

Generative Model and Discriminative Model

Imagine your task is to classify a speech to a language (Chinese, English or French).

Imagine your task is to classify a speech to a language (Chinese, English or French).

1. Learning each language, and then classifying it using the knowledge you just gained.

Imagine your task is to classify a speech to a language (Chinese, English or French).

1. Learning each language, and then classifying it using the knowledge you just gained.

2. Determining the difference in the linguistic models without learning the languages, and then classifying the speech.

Imagine your task is to classify a speech to a language (Chinese, English or French).

1. Learning each language, and then classifying it using the knowledge you just gained. **Generative approach**

2. Determining the difference in the linguistic models without learning the languages, and then classifying the speech. **Discriminative approach**

Generative vs. Discriminative

- A **generative algorithm** models how the data was generated in order to categorize a signal.
  - It asks the question: based on my generation assumptions, which category is most likely to generate this signal?
- A **discriminative algorithm** does not care about how the data was generated, it simply categorizes a given signal.

Generative vs. Discriminative

- A **generative algorithm** models how the data was generated in order to categorize a signal.
    - It asks the question: based on my generation assumptions, which category is most likely to generate this signal?
- A **discriminative algorithm** does not care about how the data was generated, it simply categorizes a given signal.

- **Generative models** model the **distribution** of individual classes.
- **Discriminative models** learn the **(hard or soft) boundary** between classes.

Generative vs. Discriminative

$$P(Y|X) = \frac{P(X|Y)\,P(Y)}{P(X)}$$

- Generative Classifiers learn a model of the joint probability $P(X, Y)$, and make their predictions by using Bayes rules.

Generative vs. Discriminative

$$P(Y|X) = \frac{P(X|Y)\,P(Y)}{P(X)}$$

- Generative Classifiers learn a model of the joint probability $P(X, Y)$, and make their predictions by using Bayes rules.
  - Assume some functional form for $P(X|Y), P(X)$.
  - Estimate parameters of $P(X|Y)$, $P(X)$ directly from training data.
  - Use Bayes rule to calculate $P(Y|X = x_i)$.
  - Pick the most likely label y.

Generative vs. Discriminative

$$P(Y|X) = \frac{P(X|Y)\,P(Y)}{P(X)}$$

- Generative Classifiers learn a model of the joint probability $P(X,Y)$, and make their predictions by using Bayes rules.
  - Assume some functional form for $P(X|Y), P(X)$.
  - Estimate parameters of $P(X|Y)$, $P(X)$ directly from training data.
  - Use Bayes rule to calculate $P(Y|X = x_i)$.
  - Pick the most likely label y.

- Discriminative Classifiers model the posterior $P(Y|X)$ directly, or learn a direct map from inputs x to the class labels y.

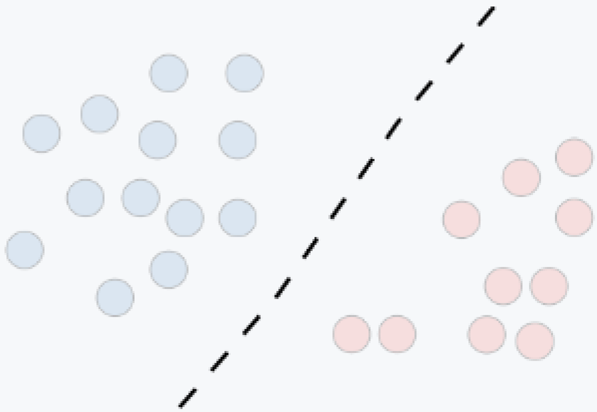
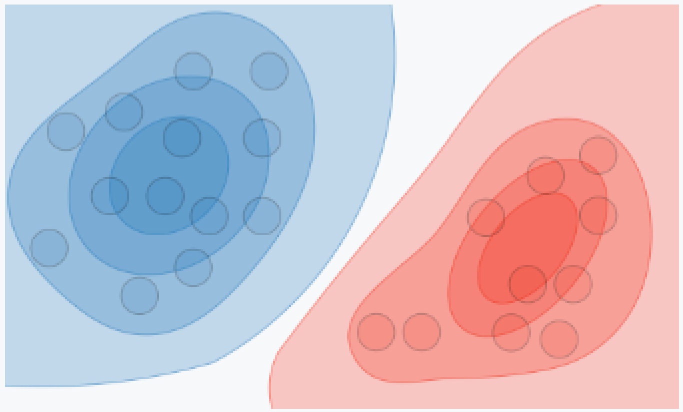
Generative vs. Discriminative

$$P(Y|X) = \frac{P(X|Y)\,P(Y)}{P(X)}$$

- Generative Classifiers learn a model of the joint probability $P(X, Y)$, and make their predictions by using Bayes rules.
  - Assume some functional form for $P(X|Y), P(X)$.
  - Estimate parameters of $P(X|Y)$, $P(X)$ directly from training data.
  - Use Bayes rule to calculate $P(Y|X = x_i)$.
  - Pick the most likely label y.

- Discriminative Classifiers model the posterior $P(Y|X)$ directly, or learn a direct map from inputs x to the class labels y.
  - Assume some function form for $P(Y|X)$.
  - Estimate parameters of $P(Y|X)$ directly from training data.
  - No attempt to model underlying probability distributions.

Generative vs. Discriminative

	Discriminative model	Generative model
Goal	Directly estimate $P(y\|x)$	Estimate $P(x\|y)$ to then deduce $P(y\|x)$
What's learned	Decision boundary	Probability distributions of the data
Illustration		
Examples	Regressions, SVMs	GMM, Naïve Bayes

Generative vs. Discriminative

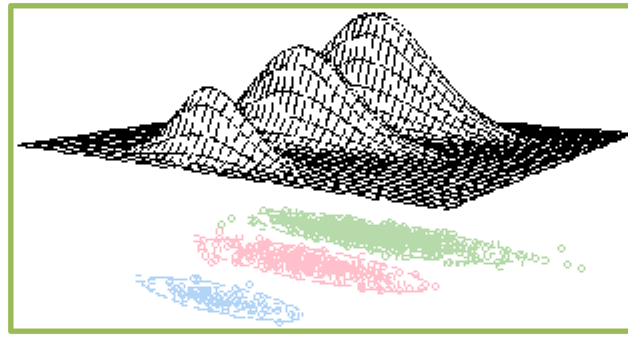
- Generative classifiers learn $P(Y|X)$ indirectly and can get the wrong assumptions of the data distribution.
- Quoting Vapnik from Statistical Learning Theory:

  "one should solve the classification problem directly and never solve a more general problem as an intermediate step, such as modeling $p(x|y)$".

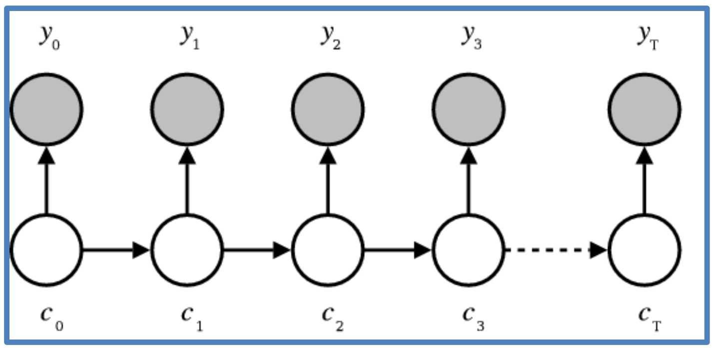
In practice, discriminative classifiers outperform generative classifiers, if you have a lot of data.

Generative Models

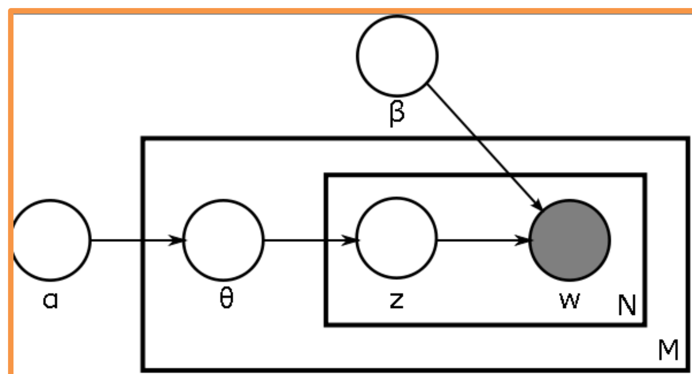
- Gaussian Mixture Model and other mixture model



- Hidden Markov Model



- Latent Dirichlet Allocation (LDA)



Discriminative Models

Nearest Neighbor

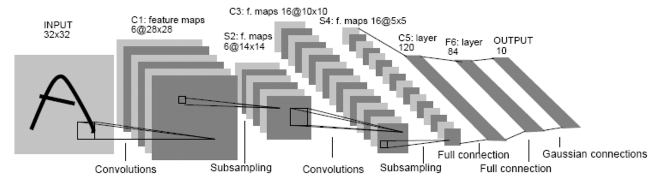
10^6 examples

Shakhnarovich, Viola, Darrell 2003

Berg, Berg, Malik 2005

…

Neural Networks

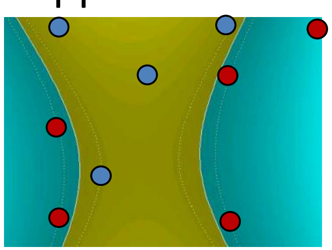


LeCun, Bottou, Bengio, Haffner 1998

Rowley, Baluja, Kanade 1998

…

Support Vector Machines

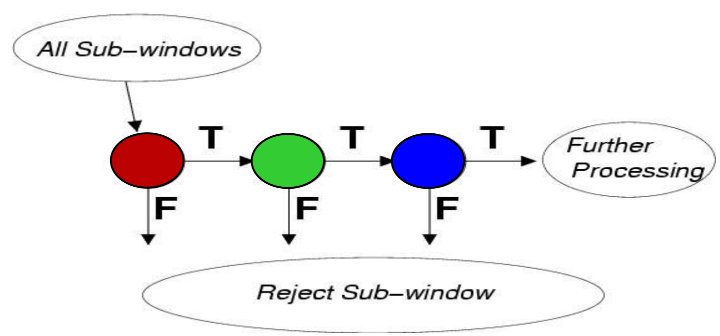


Guyon, Vapnik

Heisele, Serre, Poggio, 2001

…

Boosting



Generative: Gaussian Mixture Model

(GMM)

Review: the Gaussian distribution

The Gaussian Distribution

- If random variable X is Gaussian, it has the following PDF:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2}$$

- **Two parameters:** the mean μ and the variance σ^2 (σ is called the standard deviation).
- We will use "Gaussian" and "Normal" interchangeably.
- To save us some writing, we will write,

$$p(x) = \mathcal{N}(x; \mu, \sigma^2).$$

Parameter Estimation for Gaussians: μ

- Suppose we have i.i.d. observations $X_1, \ldots, X_N$ from a Gaussian distribution with unknown mean μ and known variance σ^2.

- If we want to find the maximum likelihood estimate for μ, then

Parameter Estimation for Gaussians: μ

- Suppose we have i.i.d. observations $X_1, \ldots, X_N$ from a Gaussian distribution with unknown mean μ and known variance σ^2.

- If we want to find the maximum likelihood estimate for μ, then

$$p(x_1, \ldots, x_N) = \prod_{i=1}^{N} \mathcal{N}(x_i; \mu, \sigma^2) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} e^{-(x_i-\mu)^2/2\sigma^2}$$

$$\ln p(x_1, \ldots, x_N) = \sum_{i=1}^{N} \ln \frac{1}{\sqrt{2\pi}\sigma} - \frac{(x_i - \mu)^2}{2\sigma^2}$$

Parameter Estimation for Gaussians: μ

- Suppose we have i.i.d. observations $X_1, \dots, X_N$ from a Gaussian distribution with unknown mean μ and known variance σ^2.

- If we want to find the maximum likelihood estimate for μ, then

$$p(x_1, \dots, x_N) = \prod_{i=1}^{N} \mathcal{N}(x_i; \mu, \sigma^2) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} e^{-(x_i-\mu)^2/2\sigma^2}$$

$$\ln p(x_1, \dots, x_N) = \sum_{i=1}^{N} \ln \frac{1}{\sqrt{2\pi}\sigma} - \frac{(x_i-\mu)^2}{2\sigma^2}$$

$$\frac{d}{d\mu} \ln p(x_1, \dots, x_N) = \sum_{i=1}^{N} \frac{x_i - \mu}{\sigma^2} = 0 \quad \Rightarrow \quad \hat{\mu} = \frac{1}{N} \sum_{i=1}^{N} x_i$$

Does not depend on the σ^2.

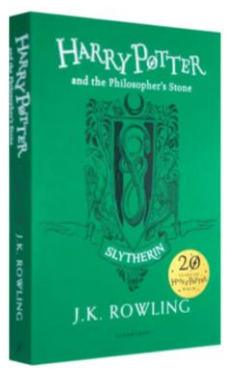
Gaussian Mixture Models

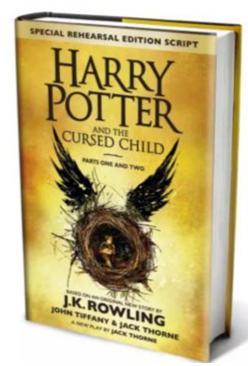
- GMM is useful for modeling data that from one of several groups: data points within the same group can be well-modeled by a Gaussian distribution.

- Example 1

  Suppose the price of a randomly chosen paperback book is normally distributed with mean \$10.00 and standard deviation \$1.00. Similarly, the price of a randomly chosen hardback is normally distributed with mean \$17 and variance \$1.50.

Paperback





Hardback

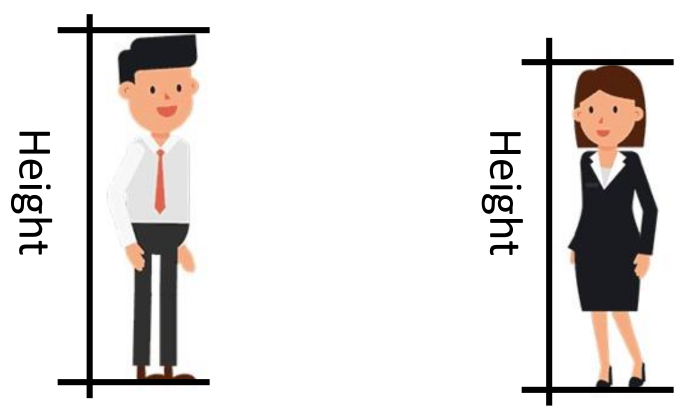
Is the price of randomly chosen book normally distributed?

Gaussian Mixture Models

- GMM is useful for modeling data that from one of several groups: data points within the same group can be well-modeled by a Gaussian distribution.

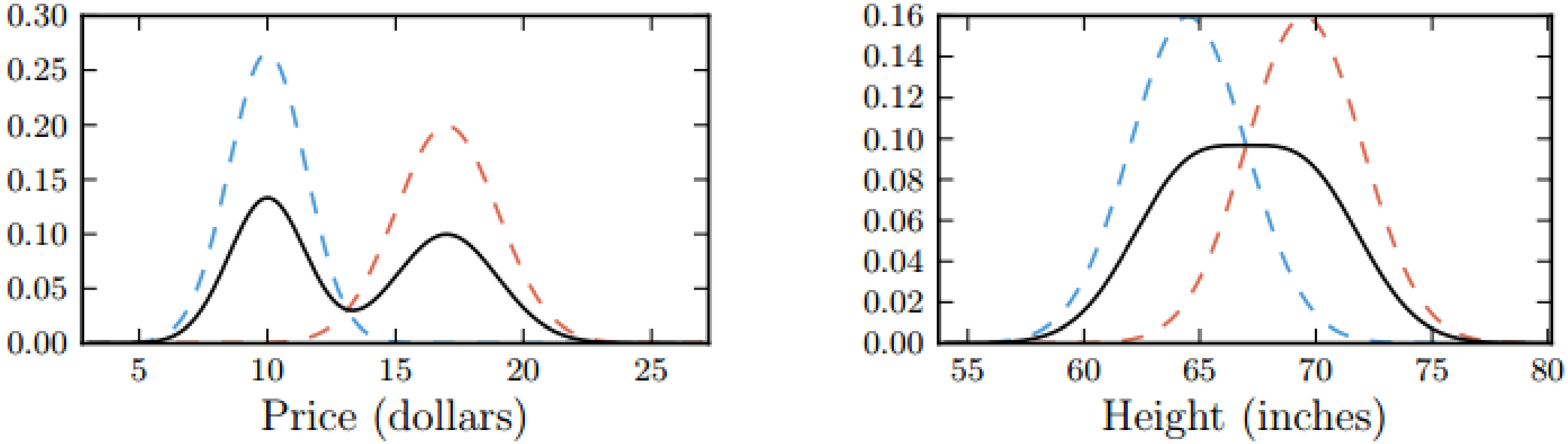
- Example 2

  Suppose the height of a randomly chosen man is normally distributed with a mean around 5’9.5’’ and standard deviation around 2.5’’. Similarly, the height of a randomly chosen woman is normally distributed with a mean around 5’4.5’’ and standard deviation around 2.5’’.



Is the height of a randomly chosen person normally distributed?

Gaussian Mixture Models

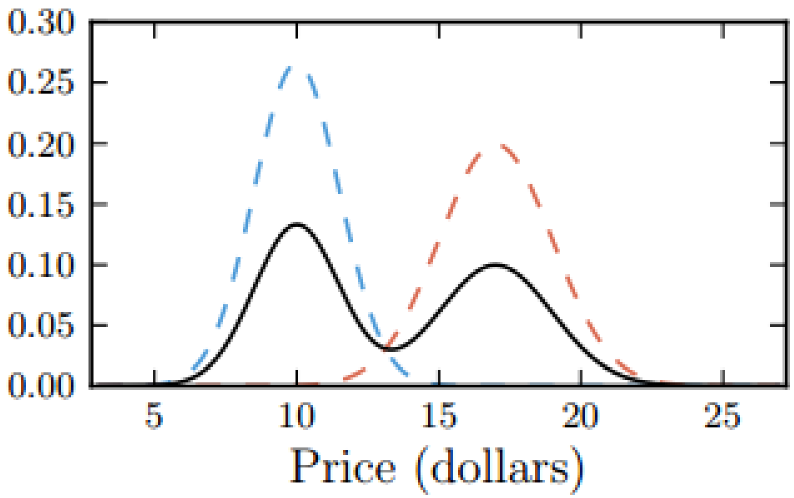


(a) Probability density for paperback books (red), hardback books (blue) and all books (black, solid)

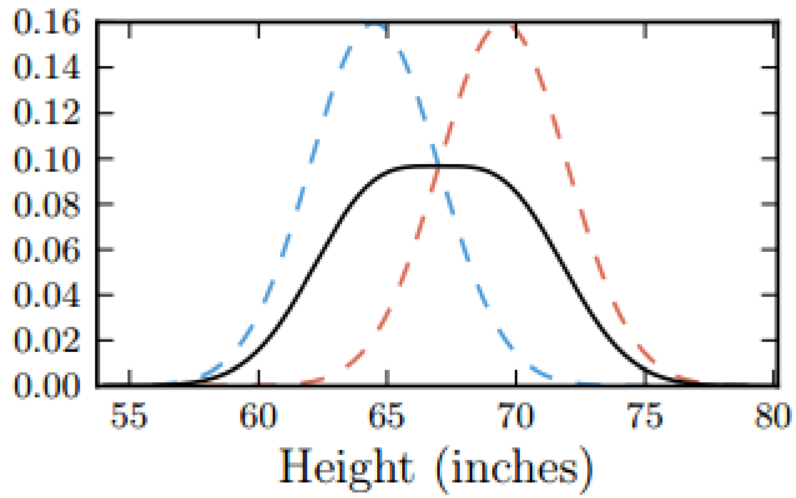
(b) Probability density for heights of women (red), heights of men (blue) and all heights (black, solid)

Figure 1. Two Gaussian mixture models: the component densities are shown in dotted red and blue lines, while the overall density is shown as solid black line.

Gaussian Mixture Models



(a) Probability density for paperback books (red), hardback books (blue) and all books (black, solid)



(b) Probability density for heights of women (red), heights of men (blue) and all heights (black, solid)

Figure 1. Two Gaussian mixture models: the component densities are shown in dotted red and blue lines, while the overall density is shown as solid black line.

Let's look at this a little more formally.

Gaussian Mixture Models

- GMM is able to form smooth approximations to arbitrarily shaped densities (GMM is a universal approximator of densities).
- An arbitrary density $f(\cdot)$, can be approximated by a Gaussian mixture model,

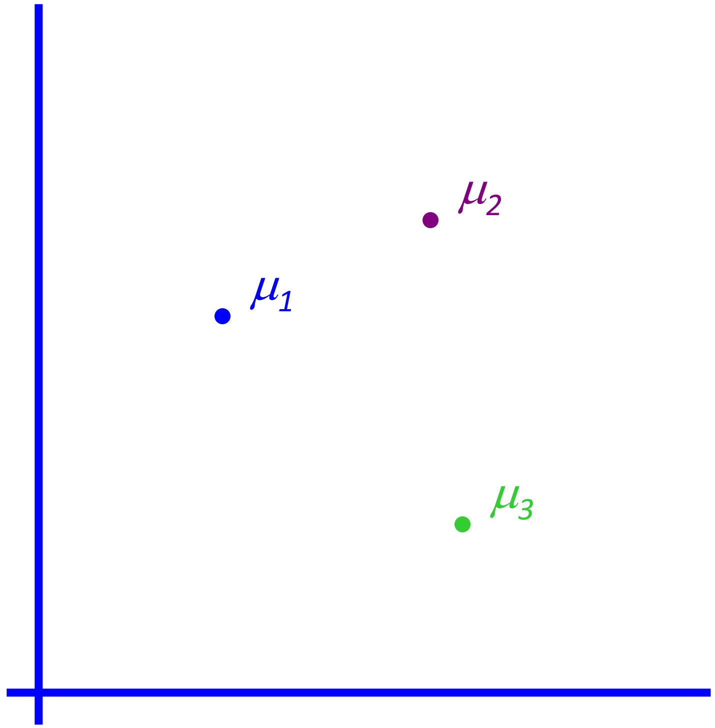
$$g_k(\cdot; \boldsymbol{\omega}, \boldsymbol{\mu}, \boldsymbol{\sigma}) = \sum_{i=1}^{k} \omega_i \varphi(\cdot; \mu_i, \sigma_i)$$

- In the sense that

$$g_k(\cdot; \boldsymbol{\omega}, \boldsymbol{\mu}, \boldsymbol{\sigma}) \overset{k \to \infty}{\longrightarrow} f(\cdot)$$

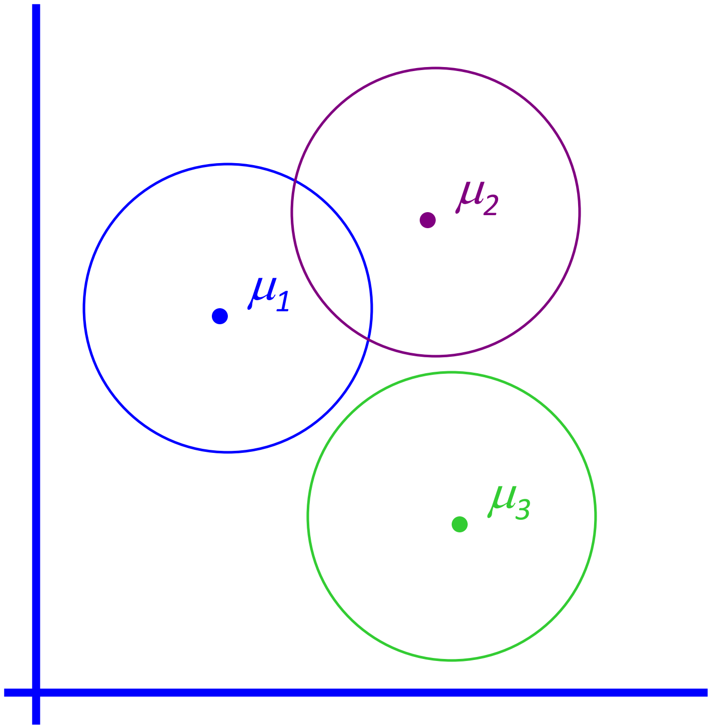
The GMM Assumption

- There are k components. The i-th component is called ω_i
- Component ω_i has an associated mean vector μ_i



The GMM Assumption

- There are k components. The i-th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian model with mean μ_i and covariance matrix $\sigma^2 \boldsymbol{I}$

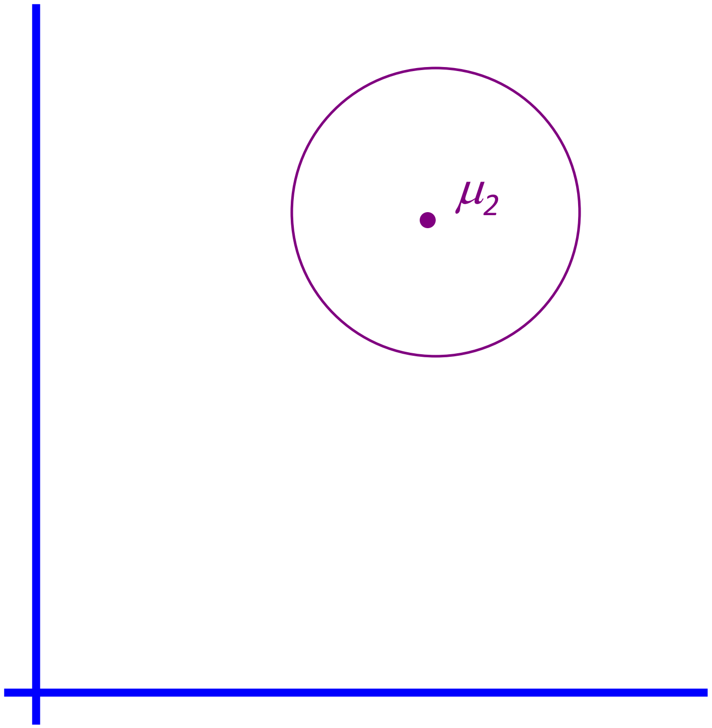


The GMM Assumption

- There are k components. The i-th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian model with mean μ_i and covariance matrix $\sigma^2 \boldsymbol{I}$

Assume that each data point is generated as follows:

1. Pick a component at random. Choose component i with probability $P(\omega_i)$.

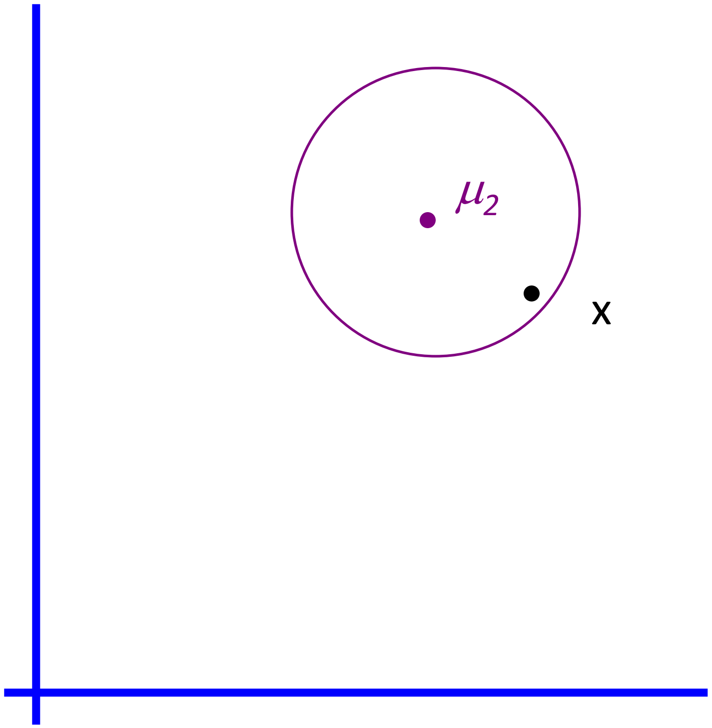


The GMM Assumption

- There are k components. The i-th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian model with mean μ_i and covariance matrix $\sigma^2 \boldsymbol{I}$

Assume that each data point is generated as follows:

1. Pick a component at random. Choose component i with probability $P(\omega_i)$.
2. Data point $\sim N(\mu_i, \sigma^2 \boldsymbol{I})$

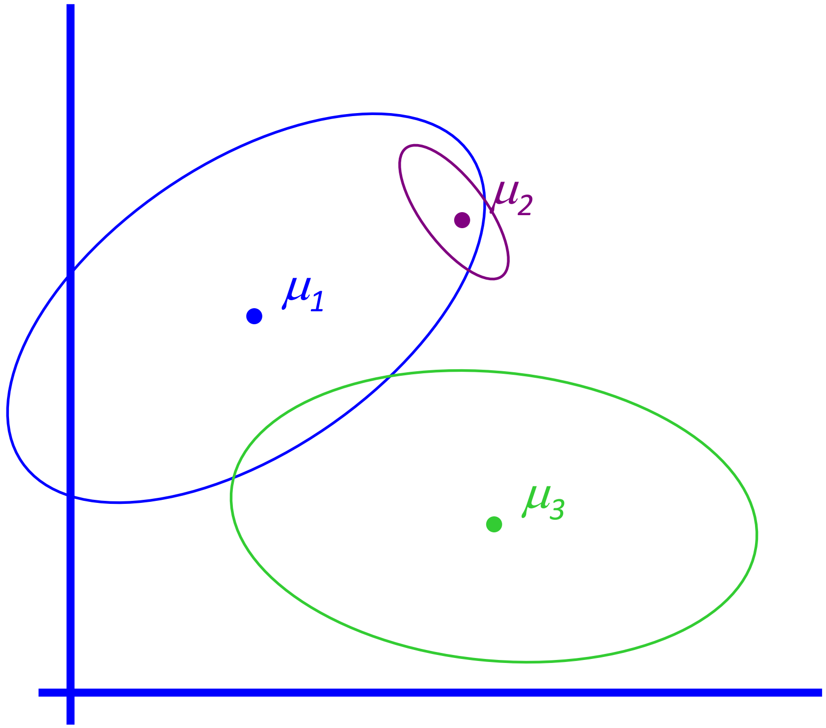


The General GMM Assumption

- There are k components. The i-th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian model with mean μ_i and covariance matrix Σ_i

Assume that each data point is generated as follows:

1. Pick a component at random. Choose component i with probability $P(\omega_i)$.
2. Data point $\sim N(\mu_i, \Sigma_i)$



Computing Likelihood

Supose we have $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N$ [data samples], we know priors for components $P(\omega_1), P(\omega_2), \dots, P(\omega_k)$, and know σ.
We want to find maximum likelihood estimates for $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_k$.

Computing Likelihood

Supose we have $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N$ [data samples], we know priors for components $P(\omega_1), P(\omega_2), \dots, P(\omega_k)$, and know σ.
We want to find maximum likelihood estimates for $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_k$.
This is an *unsupervised learning* problem.
We do not get to observe the label/component for each data.

Computing Likelihood

Supose we have $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N$ [data samples], we know priors for components $P(\omega_1), P(\omega_2), \dots, P(\omega_k)$, and know σ.
We want to find maximum likelihood estimates for $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_k$.
This is an *unsupervised learning* problem.
We do not get to observe the label/component for each data.

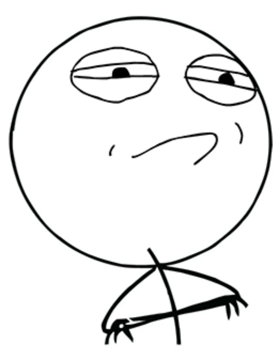
Exercise: Given the above setup, compute the log-likelihood, and then differentiate with respect to $\boldsymbol{\mu}_i$.

Computing Likelihood

Supose we have $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N$ [data samples], we know priors for components $P(\omega_1), P(\omega_2), \dots, P(\omega_k)$, and know σ.
We want to find maximum likelihood estimates for $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_k$.
This is an *unsupervised learning* problem.
We do not get to observe the label/component for each data.

Exercise: Given the above setup, compute the log-likelihood, and then differentiate with respect to $\boldsymbol{\mu}_i$.

CHALLENGING!



Computing Likelihood

Exercise: Given the above setup, compute the log-likelihood, and then differentiate with respect to $\boldsymbol{\mu}_i$.

$$P(x_1, \ldots, x_N | \mu_1, \ldots, \mu_k) = \prod_{i=1}^{N} P(x_i | \mu_1, \ldots, \mu_k)$$

$$= \prod_{i=1}^{N} \sum_{j=1}^{k} P\left(x_i \middle| \omega_j, \mu_1, \ldots, \mu_k\right) P(\omega_j) = \prod_{i=1}^{N} \sum_{j=1}^{k} \mathcal{N}(x_i; \mu_j, \sigma^2) P(\omega_j)$$

$P\left(\boldsymbol{x} \middle| \omega_j, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k\right)$: Probability that an observation from component ω_j would have value $\boldsymbol{x}$, given component means $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$

Computing Likelihood

Exercise: Given the above setup, compote the log-likelihood, and then differentiate with respect to $\boldsymbol{\mu}_i$.

$$P(x_1, \dots, x_N | \mu_1, \dots, \mu_k) = \prod_{i=1}^{N} P(x_i | \mu_1, \dots, \mu_k)$$

$$= \prod_{i=1}^{N} \sum_{j=1}^{k} P\left(x_i \,\middle|\, \omega_j, \mu_1, \dots, \mu_k\right) P(\omega_j) = \prod_{i=1}^{N} \sum_{j=1}^{k} \mathcal{N}(x_i; \mu_j, \sigma^2) P(\omega_j)$$

$$\ln P(x_1, \dots, x_N | \mu_1, \dots, \mu_k) = \sum_{i=1}^{N} \ln\left(\sum_{j=1}^{k} \mathcal{N}(x_i; \mu_j, \sigma^2) P(\omega_j)\right)$$

Computing Likelihood

Exercise: Given the above setup, compote the log-likelihood, and then differentiate with respect to $\boldsymbol{\mu}_i$.

$$P(x_1, \dots, x_N | \mu_1, \dots, \mu_k) = \prod_{i=1}^{N} P(x_i | \mu_1, \dots, \mu_k)$$

$$= \prod_{i=1}^{N} \sum_{j=1}^{k} P\left(x_i \,\middle|\, \omega_j, \mu_1, \dots, \mu_k\right) P(\omega_j) = \prod_{i=1}^{N} \sum_{j=1}^{k} \mathcal{N}(x_i; \mu_j, \sigma^2) P(\omega_j)$$

$$\ln P(x_1, \dots, x_N | \mu_1, \dots, \mu_k) = \sum_{i=1}^{N} \ln\left(\sum_{j=1}^{k} \mathcal{N}(x_i; \mu_j, \sigma^2) P(\omega_j)\right)$$

Without loss of generality, we only consider the mean μ_j.
Before we dive into differentiating, we note that

$$\frac{d}{d\mu} \mathcal{N}(x; \mu, \sigma^2) = \frac{x-\mu}{\sigma^2} \cdot \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} = \mathcal{N}(x; \mu, \sigma^2) \frac{x-\mu}{\sigma^2}$$

Computing Likelihood

Exercise: Given the above setup, compote the log-likelihood, and then differentiate with respect to $\boldsymbol{\mu}_i$.

$$\ln P(x_1, \dots, x_N | \mu_1, \dots, \mu_k) = \sum_{i=1}^{N} \ln\left(\sum_{j=1}^{k} \mathcal{N}(x_i; \mu_j, \sigma^2) P(\omega_j)\right)$$

Without loss of generality, we only consider the mean μ_j.

$$\frac{d}{d\mu_j} \ln P(x_1, \dots, x_N | \mu_1, \dots, \mu_k) = \sum_{i=1}^{N} \frac{d}{d\mu_j} \ln\left(\sum_{j=1}^{k} \mathcal{N}(x_i; \mu_j, \sigma^2) P(\omega_j)\right)$$

Computing Likelihood

Exercise: Given the above setup, compote the log-likelihood, and then differentiate with respect to $\boldsymbol{\mu}_i$.

$$\ln P(x_1, \dots, x_N | \mu_1, \dots, \mu_k) = \sum_{i=1}^{N} \ln\left(\sum_{j=1}^{k} \mathcal{N}(x_i; \mu_j, \sigma^2) P(\omega_j)\right)$$

Without loss of generality, we only consider the mean μ_j.

$$\frac{d}{d\mu_j} \ln P(x_1, \dots, x_N | \mu_1, \dots, \mu_k) = \sum_{i=1}^{N} \frac{d}{d\mu_j} \ln\left(\sum_{j=1}^{k} \mathcal{N}(x_i; \mu_j, \sigma^2) P(\omega_j)\right)$$

$$= \sum_{i=1}^{N} \frac{1}{\sum_{j=1}^{k} \mathcal{N}(x_i; \mu_j, \sigma^2) P(\omega_j)} \cdot P(\omega_j) \mathcal{N}(x_i; \mu_j, \sigma^2) \cdot \frac{x_i - \mu_j}{\sigma^2}$$

$$= 0$$

Computing Likelihood

Exercise: Given the above setup, compote the log-likelihood, and then differentiate with respect to $\boldsymbol{\mu}_i$.

$$\ln P(x_1, \dots, x_N | \mu_1, \dots, \mu_k) = \sum_{i=1}^{N} \ln\Big(\sum_{j=1}^{k} \mathcal{N}(x_i; \mu_j, \sigma^2) P(\omega_j)\Big)$$

Without loss of generality, we only consider the mean μ_j.

$$\frac{d}{d\mu_j} \ln P(x_1, \dots, x_N | \mu_1, \dots, \mu_k) = \sum_{i=1}^{N} \frac{d}{d\mu_j} \ln\Big(\sum_{j=1}^{k} \mathcal{N}(x_i; \mu_j, \sigma^2) P(\omega_j)\Big)$$

$$= \sum_{i=1}^{N} \frac{1}{\sum_{j=1}^{k} \mathcal{N}(x_i; \mu_j, \sigma^2) P(\omega_j)} \cdot P(\omega_j) \mathcal{N}(x_i; \mu_j, \sigma^2) \cdot \frac{x_i - \mu_j}{\sigma^2}$$

$$= 0$$

There's no way we can solve this in closed form to get a clean maximum likelihood expression!

The EM Algorithm

- **Expectation-maximization** (**EM**) is a method for finding maximum likelihood (or maximum a posteriori) estimate of parameter(s) in statistical model, where the model depends on **unobserved** latent variables.

The EM Algorithm

- **Expectation-maximization** (**EM**) is a method for finding maximum likelihood (or maximum a posteriori) estimate of parameter(s) in statistical model, where the model depends on **unobserved** latent variables.

Maximum Likelihood from Incomplete Data via the *EM* Algorithm

By A. P. Dempster, N. M. Laird and D. B. Rubin

Harvard University and Educational Testing Service

[Read before the Royal Statistical Society at a meeting organized by the Research Section on Wednesday, December 8th, 1976, Professor S. D. Silvey in the Chair]

TITLE	CITED BY	YEAR
Maximum likelihood from incomplete data via the EM algorithm AP Dempster, NM Laird, DB Rubin Journal of the royal statistical society. Series B (methodological), 1-38	55591	1977

The EM Algorithm

- **Expectation-maximization (EM)** is a method for finding maximum likelihood (or maximum a posteriori) estimate of parameter(s) in statistical model, where the model depends on **unobserved** latent variables.

A. P. Dempster (1929~)
Harvard University
Department of Statistics

N. M. Laird (1943~)
Harvard School of Public Health.
Department of Biostatistics

D. B. Rubin (1943~)
Harvard University (retired)
Department of Statistics

The EM Algorithm

- EM is an iterative method that alternates between performing an expectation (E) step and a maximization (M) step
  - **E-step**
  - **M-step**

The EM Algorithm

- Z: Latent variables; X: data; $\boldsymbol{\Theta}$: model parameters
- $LL(\boldsymbol{\Theta}|\boldsymbol{X},\boldsymbol{Z}) = \ln P(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\Theta})$
- $LL(\boldsymbol{\Theta}|\boldsymbol{X}) = \ln P(\boldsymbol{X}|\boldsymbol{\Theta}) = \sum_{Z} P(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\Theta})$ *marginal likelihood*

E-step computes the expectation of the log-likelihood evaluated using the current estimated distributions for the latent variables based on the parameters inferred from previous step

$$Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t) = E_{Z|X,\boldsymbol{\Theta}^t} LL(\boldsymbol{\Theta}|\boldsymbol{X},\boldsymbol{Z})$$

M-step computes parameters maximizing the expected log-likelihood from the E-step. These parameter-estimates are then used to determine the distribution of the latent variables in the next E-step.

$$\boldsymbol{\Theta}^{t+1} = \arg\max_{\boldsymbol{\Theta}} Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$$

Simple Example

Let events be "grades in a class"

w_1 = Gets an A $\qquad P(A) = \frac{1}{2}$

w_2 = Gets a B $\qquad P(B) = \mu$

w_3 = Gets a C $\qquad P(C) = 2\mu$

w_4 = Gets a D $\qquad P(D) = \frac{1}{2} - 3\mu \quad (0 \leq \mu \leq 1/6)$

Assume we want to estimate μ from data. In a given class, there were

a A's
b B's
c C's
d D's

What's the maximum likelihood estimate of μ given a, b, c, d?

Trivial Statistics

$$P(A) = \frac{1}{2} \quad P(B) = \mu \quad P(C) = 2\mu \quad P(D) = \frac{1}{2} - 3\mu,$$

$$P(\, a, b, c, d \mid \mu) = K(½)^{a}(\mu)^{b}(2\mu)^{c}(½ - 3\mu)^{d},$$

$$log\, P(\, a, b, c, d \mid \mu) = log\, K + a log\, ½ + b log\, \mu + c log\, 2\mu + d log\, (½ - 3\mu)$$

For maximizing the likelihood,

Trivial Statistics

$P(A) = \frac{1}{2}\quad P(B) = \mu \quad P(C) = 2\mu \quad P(D) = \frac{1}{2} - 3\mu,$

$P(a, b, c, d \mid \mu) = K(½)^a(\mu)^b(2\mu)^c(½ - 3\mu)^d,$

$\log P(a, b, c, d \mid \mu) = \log K + a\log ½ + b\log \mu + c\log 2\mu + d\log(½ - 3\mu)$

For maximizing the likelihood, set

$$\frac{\partial \log P}{\partial \mu} = \frac{b}{\mu} + \frac{2c}{2\mu} - \frac{3d}{\frac{1}{2} - 3\mu} = 0 \quad \Rightarrow \quad \mu = \frac{b + c}{6(b + c + d)}$$

Trivial Statistics

$P(A) = \frac{1}{2} \quad P(B) = \mu \quad P(C) = 2\mu \quad P(D) = \frac{1}{2} - 3\mu,$

$P(a, b, c, d \mid \mu) = K(½)^a(\mu)^b(2\mu)^c(½ - 3\mu)^d,$

$\log P(a, b, c, d \mid \mu) = \log K + a \log ½ + b \log \mu + c \log 2\mu + d \log(½ - 3\mu)$

For maximizing the likelihood, set

$$\frac{\partial \log P}{\partial \mu} = \frac{b}{\mu} + \frac{2c}{2\mu} - \frac{3d}{\frac{1}{2} - 3\mu} = 0 \quad \Rightarrow \quad \mu = \frac{b + c}{6(b + c + d)}$$

If class got

A	B	C	D
14	6	9	10

Then

$$\mu = \frac{1}{10}$$

Same Problem with Latent Information

Someone tells us that

Number of High grades (A's + B's) = h

Number of C's = c

Number of D's = d

What is the max likelihood estimate of μ now?

REMEMBER

$$P(A) = \frac{1}{2}, \quad P(B) = \mu, \quad P(C) = 2\mu, \quad P(D) = \frac{1}{2} - 3\mu$$

$$\log(h, c, d|\mu, b)$$
$$= \log K + (h - b)\log\frac{1}{2} + b\log\mu + c\log 2\mu + d\log(\frac{1}{2} - 3\mu)$$

We can answer this question circularly…

Same Problem with Latent Information

$$\log(h, c, d|\mu, b) = \log K + (h - b)\log\frac{1}{2} + b\log\mu + c\log 2\mu + d\log(\frac{1}{2} - 3\mu)$$

We can answer this question circularly:

Expectation Step

If we know the value of μ we could compute the expected value of b:

$$b = \frac{\mu}{\frac{1}{2} + \mu} h$$

Maximization Step

If we know the expected values of b we could compute the maximum likelihood value of μ:

$$\mu = \frac{b + c}{6(b + c + d)}$$

EM for This Problem

We begin with a guess for μ

We iterate between Expectation Step and Maximization Step to improve our estimates of μ and a and b.

Define $\mu(t)$ the estimate of μ on the t-th iteration

$b(t)$ the estimate of b on t-th iteration

- Initial guess $\mu(0)$

E-step $$b(t) = \frac{\mu(t)}{\frac{1}{2} + \mu(t)} h = E[b|\mu(t)]$$

M-step $$\mu(t+1) = \frac{b(t) + c}{6(b(t) + c + d)}$$

Continue iterating until converged.

EM for This Problem

Good news: Converging to local optimum is assured.
Bad news: "local" optimum.

EM Convergence

- Convergence proof based on fact that Prob(data|μ) must increase or remain same between each iteration [NOT STUDY HERE]
- But it can never exceed 1 [OBVIOUS]
- So it must therefore converge [OBVIOUS]

In our example, suppose we had

$h = 20$

$c = 10$

$d = 10$

$\mu(0) = 0$

t	$\mu(t)$	$b(t)$
0	0	0
1	0.0833	2.857
2	0.0937	3.158
3	0.0947	3.185
4	0.0948	3.187
5	0.0948	3.187
6	0.0948	3.187

Back to Learning of GMM

We have unlabeled data $x_1, x_2, \dots, x_N$
We have the prior for k components $P(\omega_1), P(\omega_2), \dots, P(\omega_k), \sigma$
Hidden (Latent) variables, k-dimensional 0-1 vectors $z_1, \dots, z_N$,
indicating which component each point is sampled from.

$$P(x_1, \dots, x_N | \mu_1, \dots, \mu_k, z_1, \dots, z_N) = \prod_{i=1}^{N} P(x_i | \mu_1, \dots, \mu_k, z_1, \dots, z_N)$$

$$= \prod_{i=1}^{N} \sum_{j=1}^{k} z_i^j P(x_i | \mu_1, \dots, \mu_k, \omega_j) P(\omega_j)$$

$$= \prod_{i=1}^{N} \sum_{j=1}^{k} z_i^j C exp\left(-\frac{1}{2\sigma^2}(x_i - \mu_j)^2\right) P(\omega_j)$$

EM for GMMs

Calculate the logarithm of the likelihood,

$$\log P(x_1, \dots, x_N | \mu_1, \dots, \mu_k, z_1, \dots, z_N)$$

$$= \sum_{i=1}^{N} log \sum_{j=1}^{K} z_i^j Cexp\left(-\frac{1}{2\sigma^2}(x_i - \mu_j)^2\right) P(\omega_j)$$

$$= \sum_{i=1}^{N} \sum_{j=1}^{K} z_i^j \log Cexp\left(-\frac{1}{2\sigma^2}(x_i - \mu_j)^2\right) P(\omega_j)$$

Setting $$\frac{\partial \log P(x_1, \dots, x_N | \mu_1, \dots, \mu_k, z_1, \dots, z_N)}{\partial \mu_j} = 0$$

We have $$\mu_j = \frac{\sum_{i=1}^{N} z_i^j x_i}{\sum_{i=1}^{N} z_i^j}$$

EM for GMMs

Iteration. On the t-th iteration let our estimates be

$$\lambda_t = \{\mu_1(t), \mu_2(t), \ldots, \mu_K(t)\}$$

E-step： Computes the expectation of the log-likelihood

$$P\left(z_i^j = 1 \middle| \lambda_t\right) = \frac{P(x_i|\omega_j, \lambda_t)P(\omega_j|\lambda_t)}{P(x_i|\lambda_t)} = \frac{P(x_i|\omega_j, \mu_j(t))P(\omega_j)}{\sum_{l=1}^{k} P(x_i|\omega_l, \mu_l(t))P(\omega_l)}$$

$$E_{P\left(z_i^j \middle| \lambda_t\right)}(\log P(x_1, \ldots, x_N|\mu_1, \ldots, \mu_k, z_1, \ldots, z_N))$$

$$= E_{P\left(z_i^j \middle| \lambda_t\right)} \sum_{i=1}^{N} \sum_{j=1}^{K} z_i^j \log C exp\left(-\frac{1}{2\sigma^2}(x_i - \mu_j)^2\right) P(\omega_j)$$

$$= \sum_{i=1}^{N} \sum_{j=1}^{K} E_{P\left(z_i^j \middle| \lambda_t\right)} z_i^j \log C exp\left(-\frac{1}{2\sigma^2}(x_i - \mu_j)^2\right) P(\omega_j)$$

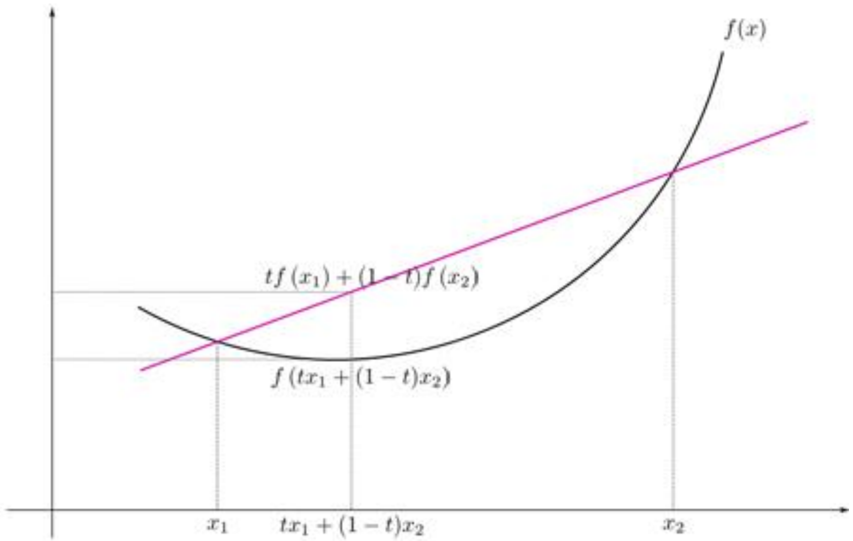
EM for GMMs

Why does $E_{P\left(z_i^j \middle| \lambda_t\right)}(\log P(x_1, \ldots, x_N | \mu_1, \ldots, \mu_k, z_1, \ldots, z_N))$ **work?**

EM for GMMs

[Jensen's inequality] Let f be a convex function. Then

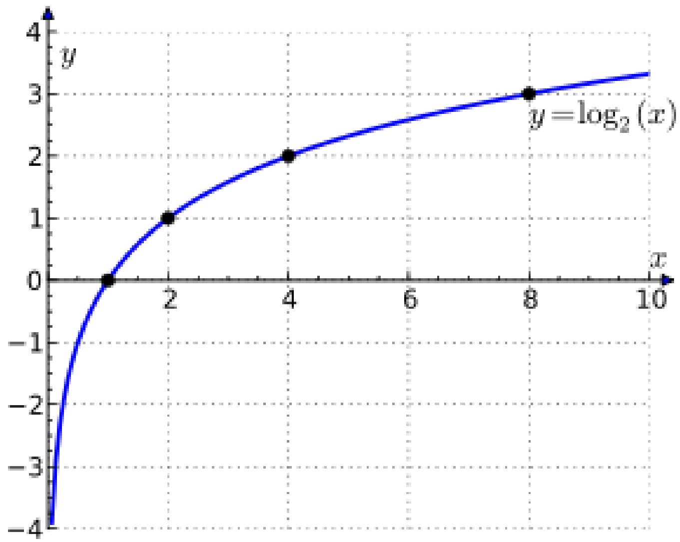
$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda)f(x_2) \quad \text{where} \quad \lambda \in [0,1]$$



- In the context of probability theory, if X is a random variable, f is convex, then, we have $f(E[X]) \leq E[f(X)]$
- If f is strictly convex, then $E[f(X)] = f(E[X])$ holds true if and only if $X = E[X]$ with probability 1 (i.e., if X is constant).

EM for GMMs

How about the concave function? (E.g., log function)



$$log(E[X]) \geq E[log(X)]$$

EM for GMMs

Log-likelihood

$$\log p_{x_1,\dots x_N}(x_1,\dots x_N;\lambda) = \sum_{i=1}^{N} \log p_{x_i}(x_i;\lambda)$$

Marginalizing over z_i^j and introducing $P\left(z_i^j \middle| \lambda\right) / P\left(z_i^j \middle| \lambda\right)$

$$= \sum_{i=1}^{N} \log \sum_{z_i^j} P\left(z_i^j \middle| \lambda\right) \frac{p_{x_i}\left(x_i, z_i^j \middle| \lambda\right)}{P\left(z_i^j \middle| \lambda\right)}$$

EM for GMMs

Log-likelihood

$$\log p_{x_1,\ldots x_N}(x_1,\ldots x_N;\lambda) = \sum_{i=1}^{N} \log p_{x_i}(x_i;\lambda)$$

Marginalizing over z_i^j and introducing $P\left(z_i^j \middle| \lambda\right) / P\left(z_i^j \middle| \lambda\right)$

$$= \sum_{i=1}^{N} \log \sum_{z_i^j} P\left(z_i^j \middle| \lambda\right) \frac{p_{x_i}\left(x_i, z_i^j \middle| \lambda\right)}{P\left(z_i^j \middle| \lambda\right)}$$

Rewriting as an expectation

EM for GMMs

Log-likelihood

$$\log p_{x_1,\dots x_N}(x_1, \dots x_N; \lambda) = \sum_{i=1}^{N} \log p_{x_i}(x_i; \lambda)$$

Marginalizing over z_i^j and introducing $P\left(z_i^j \middle| \lambda\right) / P\left(z_i^j \middle| \lambda\right)$

$$= \sum_{i=1}^{N} \log \sum_{z_i^j} P\left(z_i^j \middle| \lambda\right) \frac{p_{x_i}\left(x_i, z_i^j \middle| \lambda\right)}{P\left(z_i^j \middle| \lambda\right)}$$

Rewriting as an expectation

$$= \sum_{i=1}^{N} \log E_{P\left(z_i^j \middle| \lambda\right)} \frac{p_{x_i, z_i^j}\left(x_i, z_i^j \middle| \lambda\right)}{P\left(z_i^j \middle| \lambda\right)}$$

EM for GMMs

Log-likelihood

$$\log p_{x_1,\ldots x_N}(x_1, \ldots x_N; \lambda) = \sum_{i=1}^{N} \log p_{x_i}(x_i; \lambda)$$

Marginalizing over z_i^j and introducing $P\left(z_i^j \middle| \lambda\right) / P\left(z_i^j \middle| \lambda\right)$

$$= \sum_{i=1}^{N} \log \sum_{z_i^j} P\left(z_i^j \middle| \lambda\right) \frac{p_{x_i}\left(x_i, z_i^j \middle| \lambda\right)}{P\left(z_i^j \middle| \lambda\right)}$$

Rewriting as an expectation

$$= \sum_{i=1}^{N} \log E_{P\left(z_i^j \middle| \lambda\right)} \frac{p_{x_i, z_i^j}\left(x_i, z_i^j \middle| \lambda\right)}{P\left(z_i^j \middle| \lambda\right)}$$

Using Jensen's inequality

$$\geq \sum_{i=1}^{N} E_{P\left(z_i^j \middle| \lambda\right)} \log \frac{p_{x_i, z_i^j}\left(x_i, z_i^j \middle| \lambda\right)}{P\left(z_i^j \middle| \lambda\right)}$$

EM for GMMs

Log-likelihood

$$\log p_{x_1,\ldots x_N}(x_1,\ldots x_N;\lambda) = \sum_{i=1}^{N} \log p_{x_i}(x_i;\lambda)$$

Marginalizing over z_i^j and introducing $P\left(z_i^j\middle|\lambda\right)/P\left(z_i^j\middle|\lambda\right)$

$$= \sum_{i=1}^{N} \log \sum_{z_i^j} P\left(z_i^j\middle|\lambda\right) \frac{p_{x_i}\left(x_i, z_i^j\middle|\lambda\right)}{P\left(z_i^j\middle|\lambda\right)}$$

Rewriting as an expectation

$$= \sum_{i=1}^{N} \log E_{P\left(z_i^j\middle|\lambda\right)} \frac{p_{x_i,z_i^j}\left(x_i, z_i^j\middle|\lambda\right)}{P\left(z_i^j\middle|\lambda\right)}$$

Using Jensen's inequality

$$\geq \sum_{i=1}^{N} E_{P\left(z_i^j\middle|\lambda\right)} \log \frac{p_{x_i,z_i^j}\left(x_i, z_i^j\middle|\lambda\right)}{P\left(z_i^j\middle|\lambda\right)}$$

$P(X,Y|Z) = P(X|Y,Z)P(Y|Z)$

$$= \sum_{i=1}^{N} E_{P\left(z_i^j\middle|\lambda\right)} \log p_{x_i}\left(x_i\middle|z_i^j,\lambda\right)$$

EM for GMMs

Why does $E_{P\left(z_i^j \middle| \lambda_t\right)}(\log P(x_1, \dots, x_N | \mu_1, \dots, \mu_k, z_1, \dots, z_N))$ **work?**

It is the lower bound on $\log P(x_1, \dots, x_N | \mu_1, \dots, \mu_k)$**.**

EM for GMMs

Iteration. On the t-th iteration let our estimates be

$$\lambda_t = \{\mu_1(t), \mu_2(t), \dots, \mu_K(t)\}$$

E-step: Computes the expectation of the log-likelihood

$$E_{P\left(z_i^j \middle| \lambda_t\right)}(\log P(x_1, \dots, x_N | \mu_1, \dots, \mu_k, z_1, \dots, z_N))$$

$$= \sum_{i=1}^{N} \sum_{j=1}^{K} E_{P\left(z_i^j \middle| \lambda_t\right)}(z_i^j) \log Kexp\left(-\frac{1}{2\sigma^2}(x_i - \mu_j)^2\right) P(\omega_j)$$

M-step: Estimate $\boldsymbol{\mu}$ given the data's component membership distributions

$$\mu_j(t+1) = \frac{\sum_{i=1}^{N} E_{P\left(z_i^j \middle| \lambda_t\right)}(z_i^j) x_i}{\sum_{i=1}^{N} E_{P\left(z_i^j \middle| \lambda_t\right)}(z_i^j)}$$

EM for General GMMs

$P_i(t)$ is shorthand for estimate of $P(\omega_j)$ on t-th iteration

Iteration. On the t-th iteration let our estimates be

$$\lambda_t = \{\mu_1(t), \mu_2(t), \dots, \mu_K(t), \Sigma_1(t), \Sigma_2(t), \dots, \Sigma_K(t), P_1(t), P_2(t), \dots, P_K(t)\}$$

E-step： Computes the expectation of the log-likelihood

$$P\left(z_i^j = 1 \middle| \lambda_t\right) = \frac{P\left(x_i \middle| \omega_j, \lambda_t\right) P\left(\omega_j \middle| \lambda_t\right)}{P\left(x_i \middle| \lambda_t\right)} = \frac{P\left(x_i \middle| \omega_j, \Sigma_j(t), \mu_j(t)\right) P_j(t)}{\sum_{l=1}^{k} P\left(x_i \middle| \omega_l, \Sigma_l(t), \mu_l(t)\right) P_l(t)}$$

$$E_{P\left(z_i^j \middle| \lambda_t\right)}\left(\log P\left(x_1, \dots, x_N \middle| \lambda_t, z_1, \dots, z_N\right)\right)$$

EM for General GMMs

$P_i(t)$ is shorthand for estimate of $P(\omega_j)$ on t-th iteration

Iteration. On the t-th iteration let our estimates be

$$\lambda_t = \{\mu_1(t), \mu_2(t), \ldots, \mu_K(t), \Sigma_1(t), \Sigma_2(t), \ldots, \Sigma_K(t), P_1(t), P_2(t), \ldots, P_K(t)\}$$

E-step： Computes the expectation of the log-likelihood

$$E_{P\left(z_i^j \middle| \lambda_t\right)}(\log P(x_1, \ldots, x_N | \lambda_t, z_1, \ldots, z_N))$$

M-step ： Estimate **parameters** given the data's component membership distributions

$$\frac{\partial\, E_{P\left(z_i^j \middle| \lambda_t\right)}(\log P(x_1, \ldots, x_N | \lambda_t))}{\partial \mu_j} = 0 \qquad \frac{\partial\, E_{P\left(z_i^j \middle| \lambda_t\right)}(\log P(x_1, \ldots, x_N | \lambda_t))}{\partial \Sigma_j} = 0$$

$$\frac{\partial\, E_{P\left(z_i^j \middle| \lambda_t\right)}(\log P(x_1, \ldots, x_N | \lambda_t))}{\partial P_j} = 0$$

EM for General GMMs

$P_i(t)$ is shorthand for estimate of $P(\omega_j)$ on t-th iteration

Iteration. On the t-th iteration let our estimates be

$$\lambda_t = \{\mu_1(t), \mu_2(t), \dots, \mu_K(t), \Sigma_1(t), \Sigma_2(t), \dots, \Sigma_K(t), P_1(t), P_2(t), \dots, P_K(t)\}$$

E-step： Computes the expectation of the log-likelihood

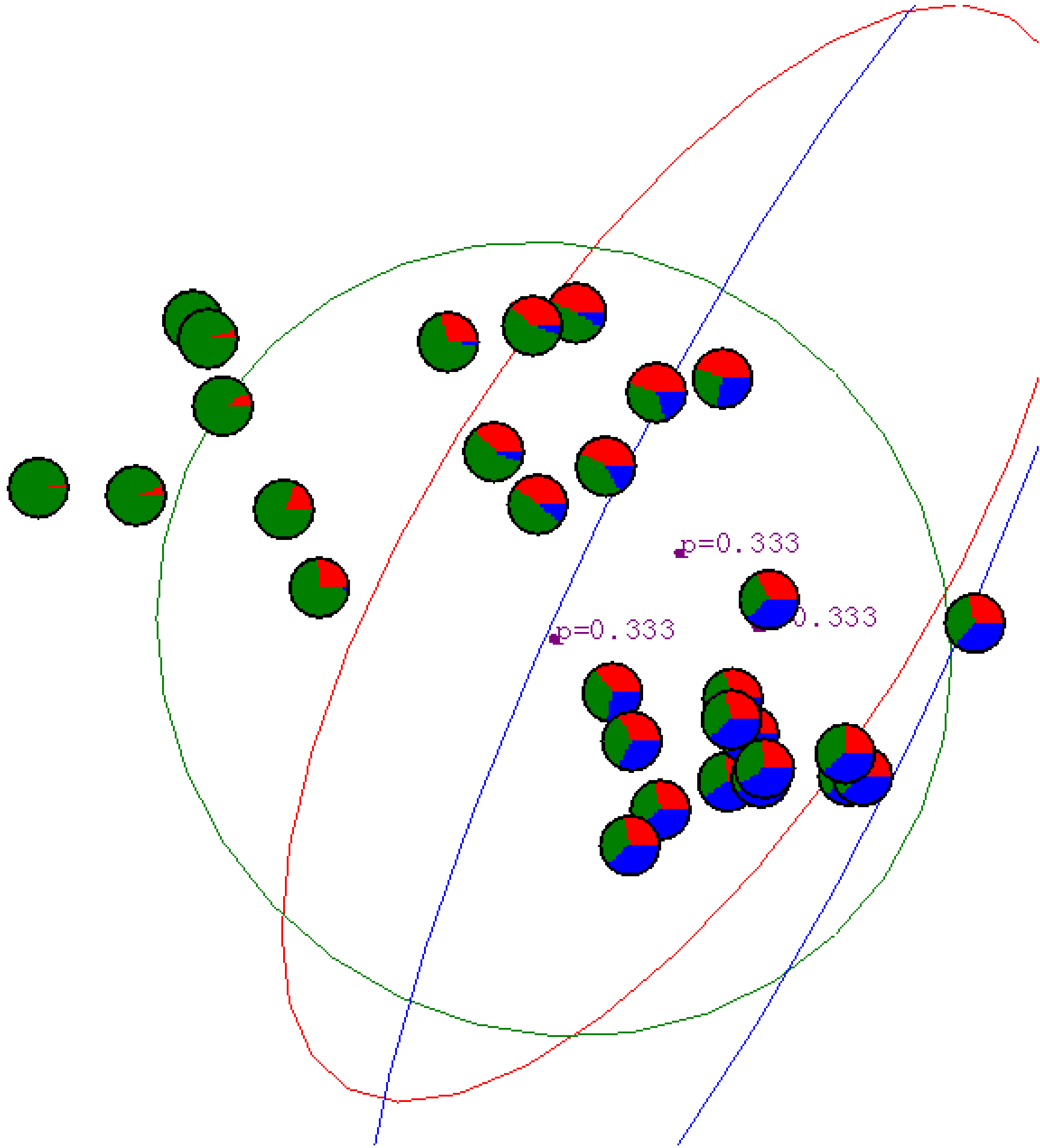
$$E_{P\left(z_i^j \middle| \lambda_t\right)}(\log P(x_1, \dots, x_N | \lambda_t, z_1, \dots, z_N))$$

M-step： Estimate **parameters** given the data's component membership distributions

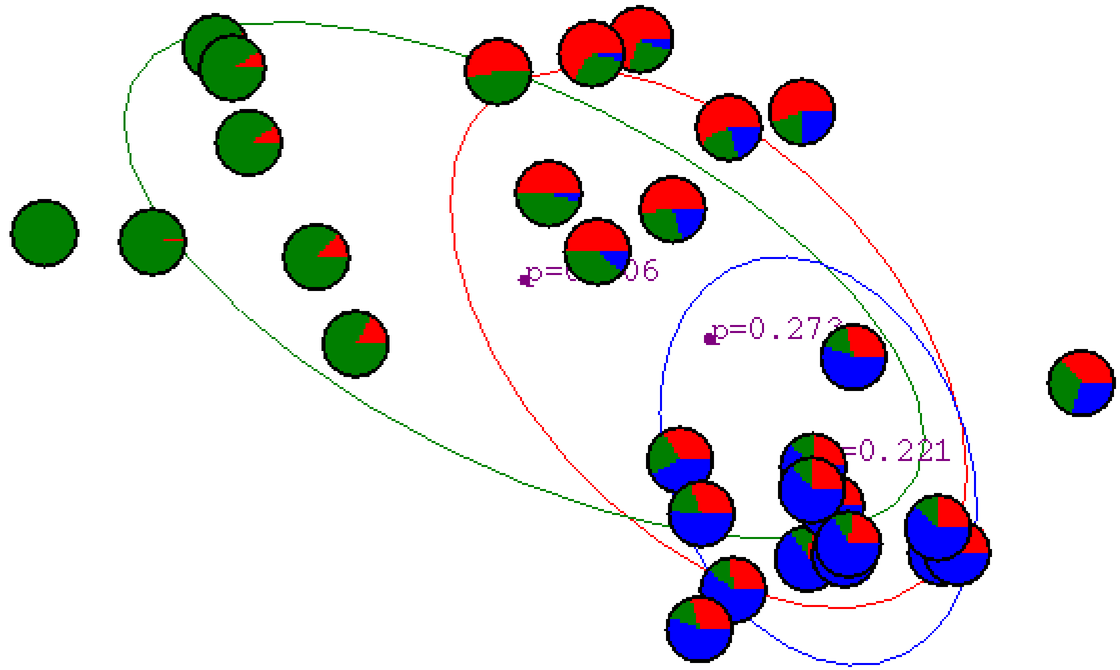
$$\mu_j(t+1) = \frac{\sum_{i=1}^N P\left(z_i^j \middle| \lambda_t\right) x_i}{\sum_{i=1}^N P\left(z_i^j \middle| \lambda_t\right)} \qquad P_j(t+1) = \frac{\sum_{i=1}^N P\left(z_i^j \middle| \lambda_t\right)}{N}$$

$$\Sigma_j(t+1) = \frac{\sum_{i=1}^N P\left(z_i^j \middle| \lambda_t\right) [x_i - \mu_j(t+1)][x_i - \mu_j(t+1)]^T}{\sum_{i=1}^N P\left(z_i^j \middle| \lambda_t\right)}$$

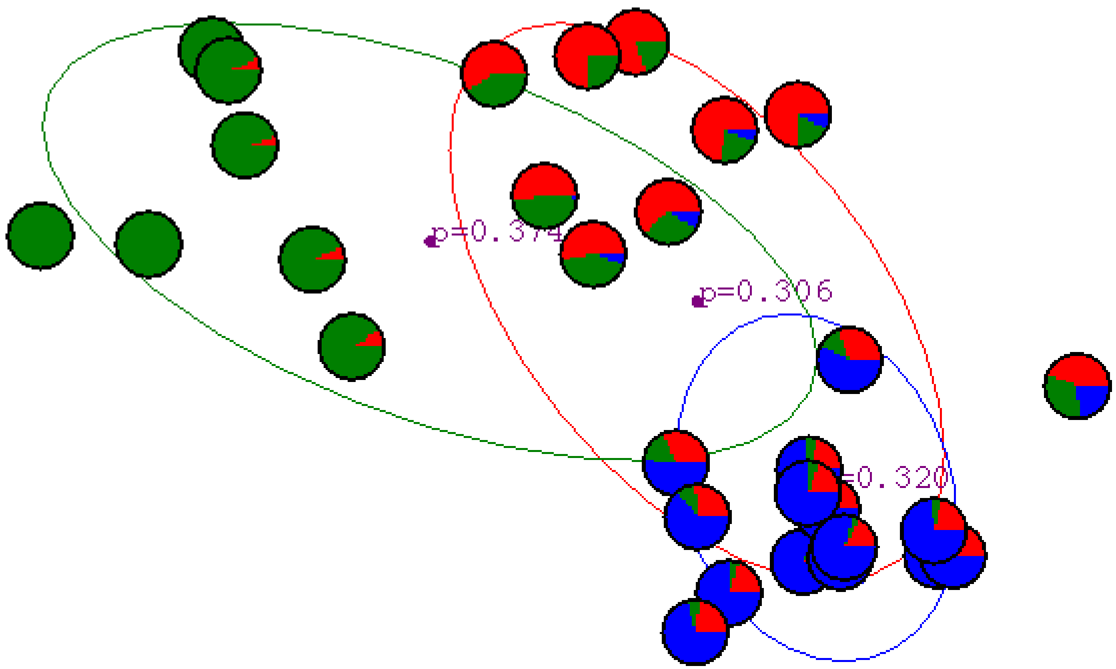
Gaussian Mixture Model Example: Start



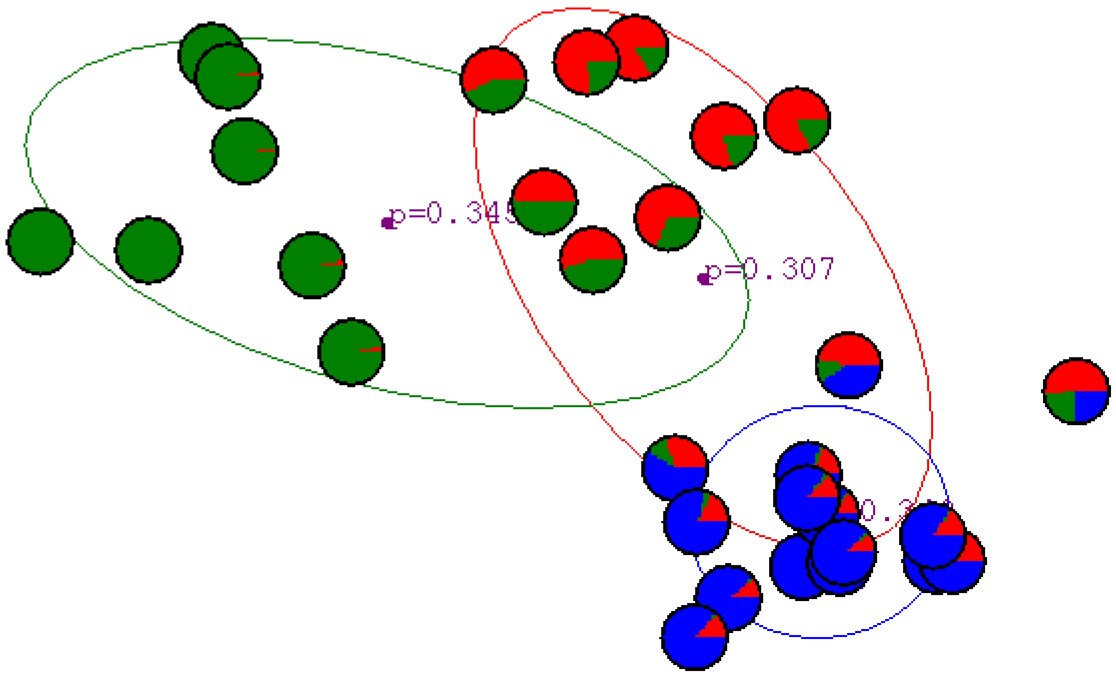
After 1st iteration



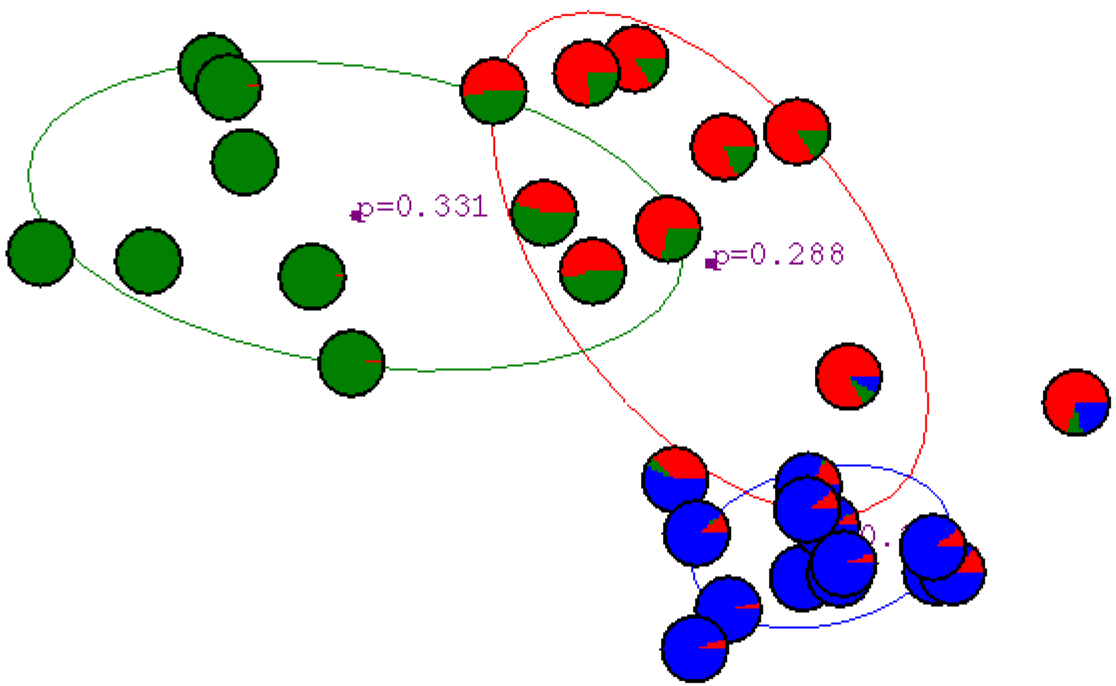
After 2nd iteration



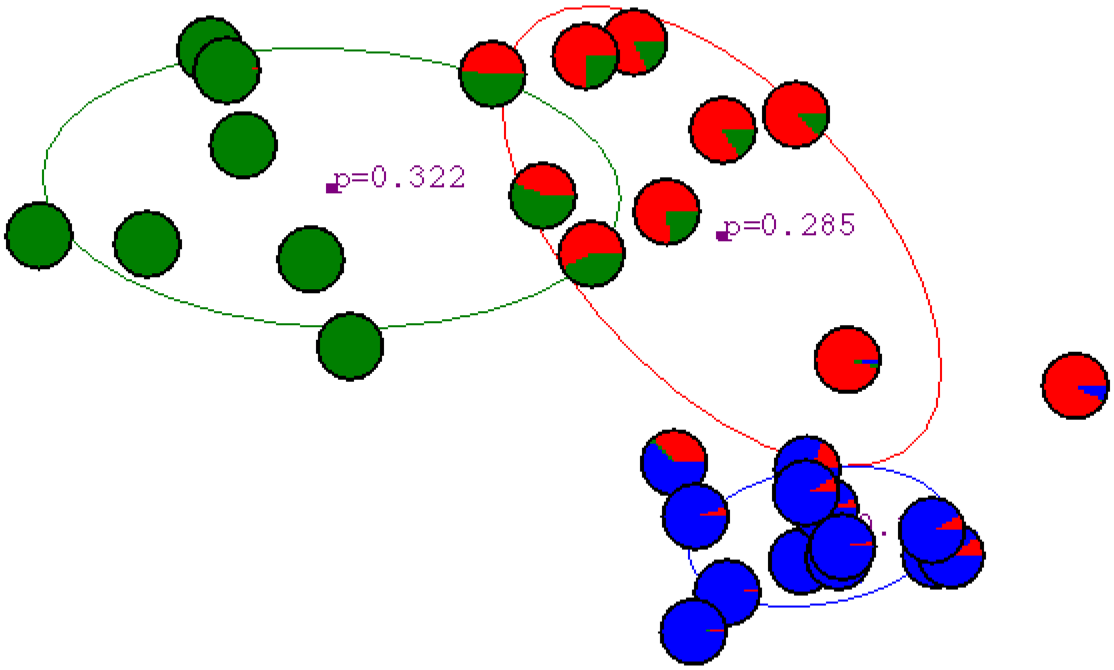
After 3rd iteration



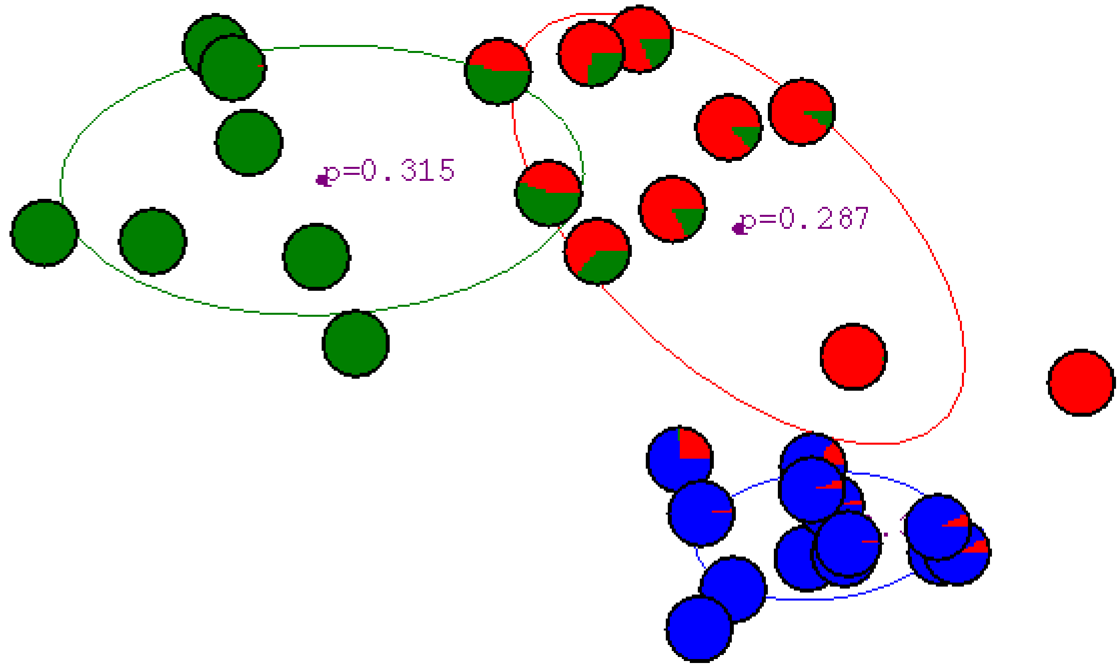
After 4th iteration



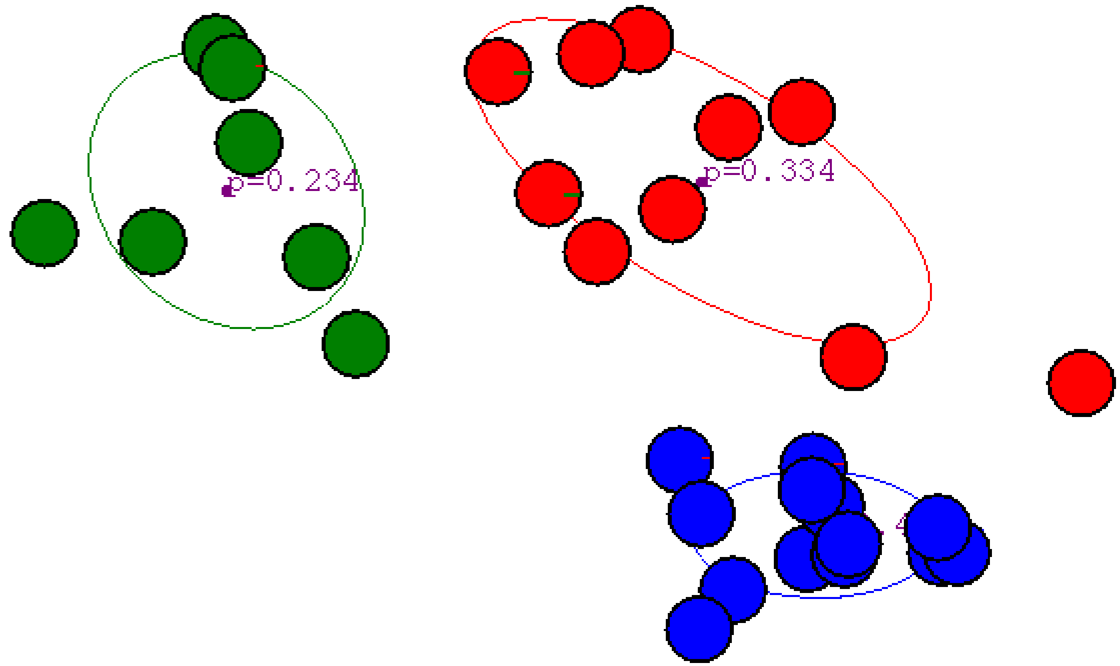
After 5th iteration



After 6th iteration

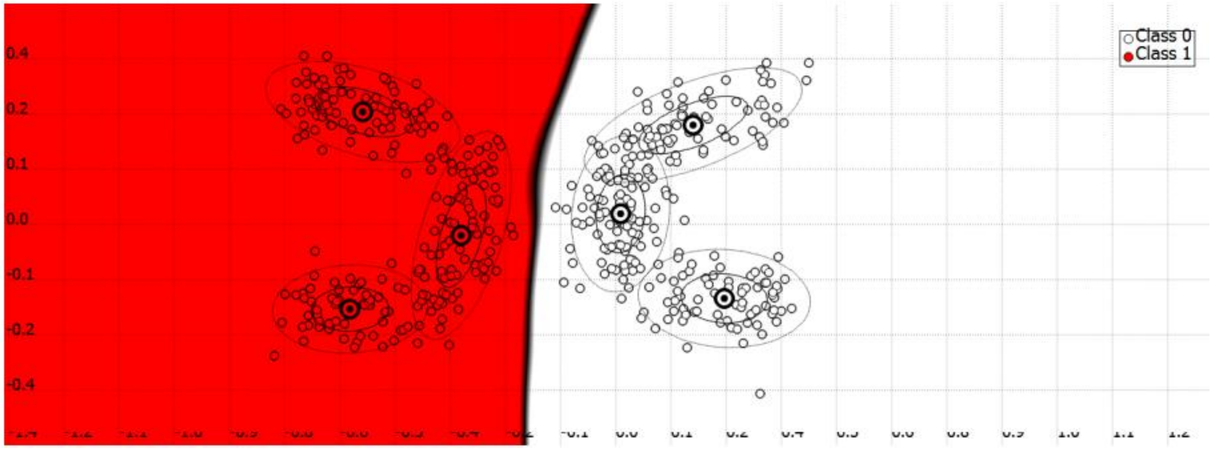


After 20th iteration



How to Use GMM for Classification

- Train GMM for each class , and then use Bayesian Rule for classification



Example of binary classification using two GMMs

Train each GMM separately, using data set of Class 1 for GMM1 and dataset of class 2 for GMM2 (3 Gaussians for each GMM)

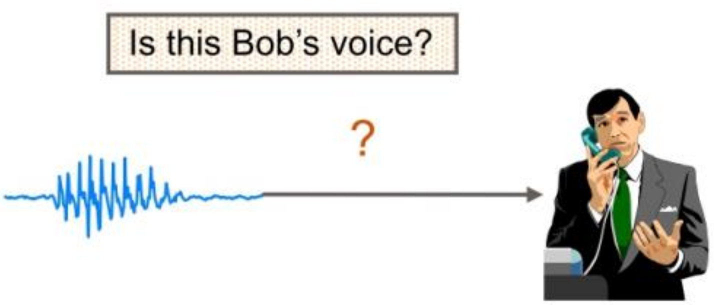
How to Use GMM for Classification

- Train GMM for each class , and then use Bayesian Rule for classification

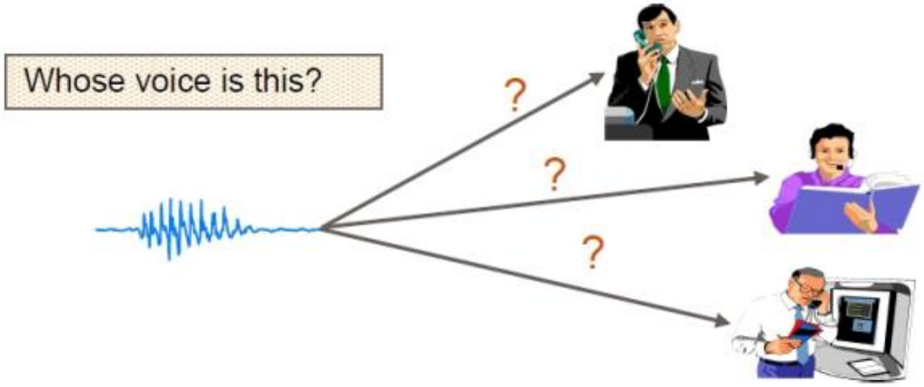
- Train universal GMM, and then adapt it for individual class, and finally do classification
  - Widely used in speaker verification

Speaker Verification and Identification

- **Speaker Verification:** Determine whether unknown speaker matches a specific speaker
  - One-to-one mapping
- **Speaker Identification:** Determine whether unknown speaker matches one of a set known speakers
  - One-to-many mapping



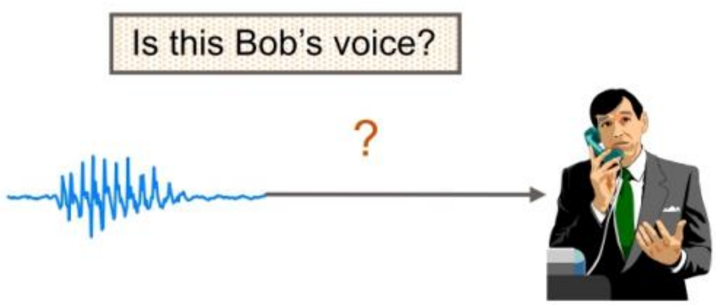
Speaker Verification



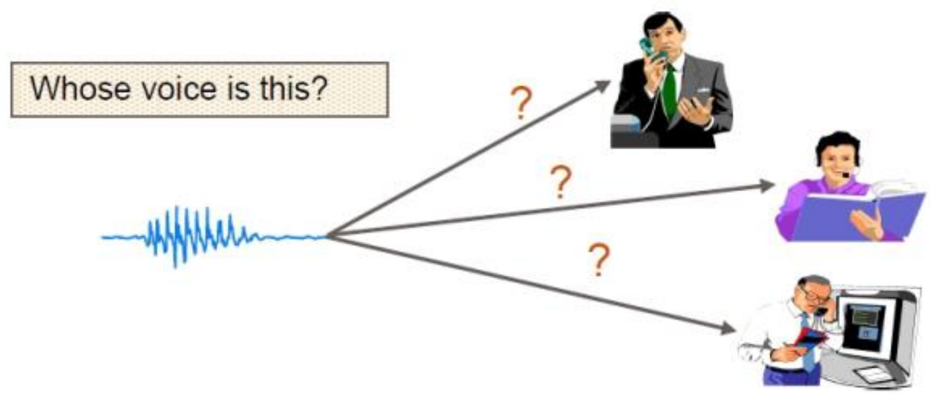
Speaker Identification

Speaker Verification and Identification

- **Speaker Verification:** Determine whether unknown speaker matches a specific speaker
  - One-to-one mapping
- **Speaker Identification:** Determine whether unknown speaker matches one of a set known speakers
  - One-to-many mapping



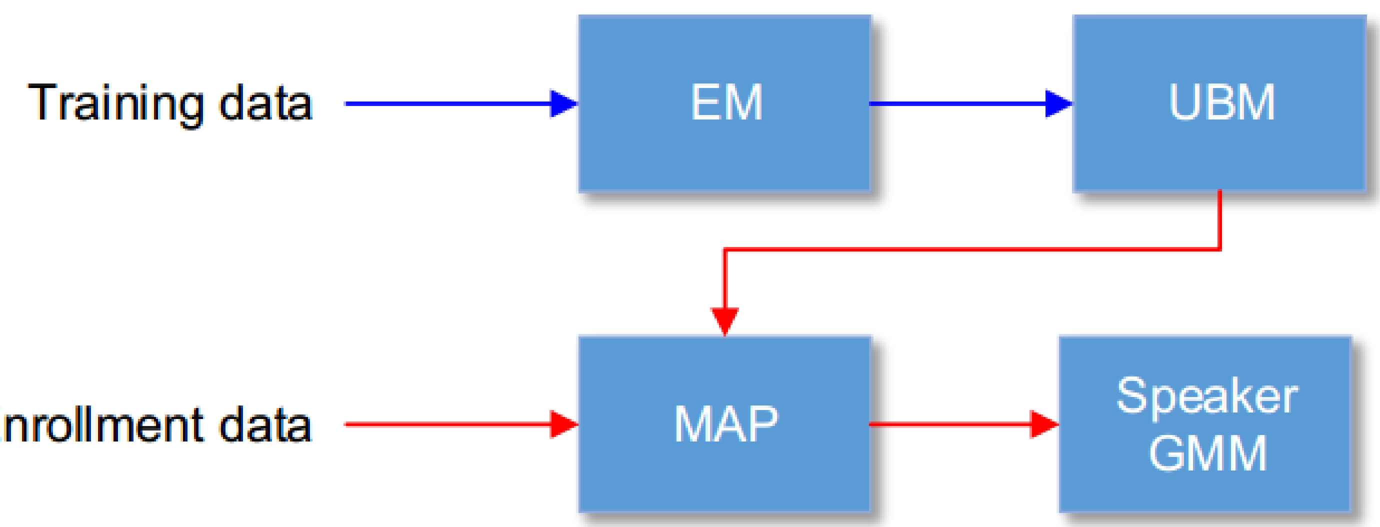
Speaker Verification



Speaker Identification

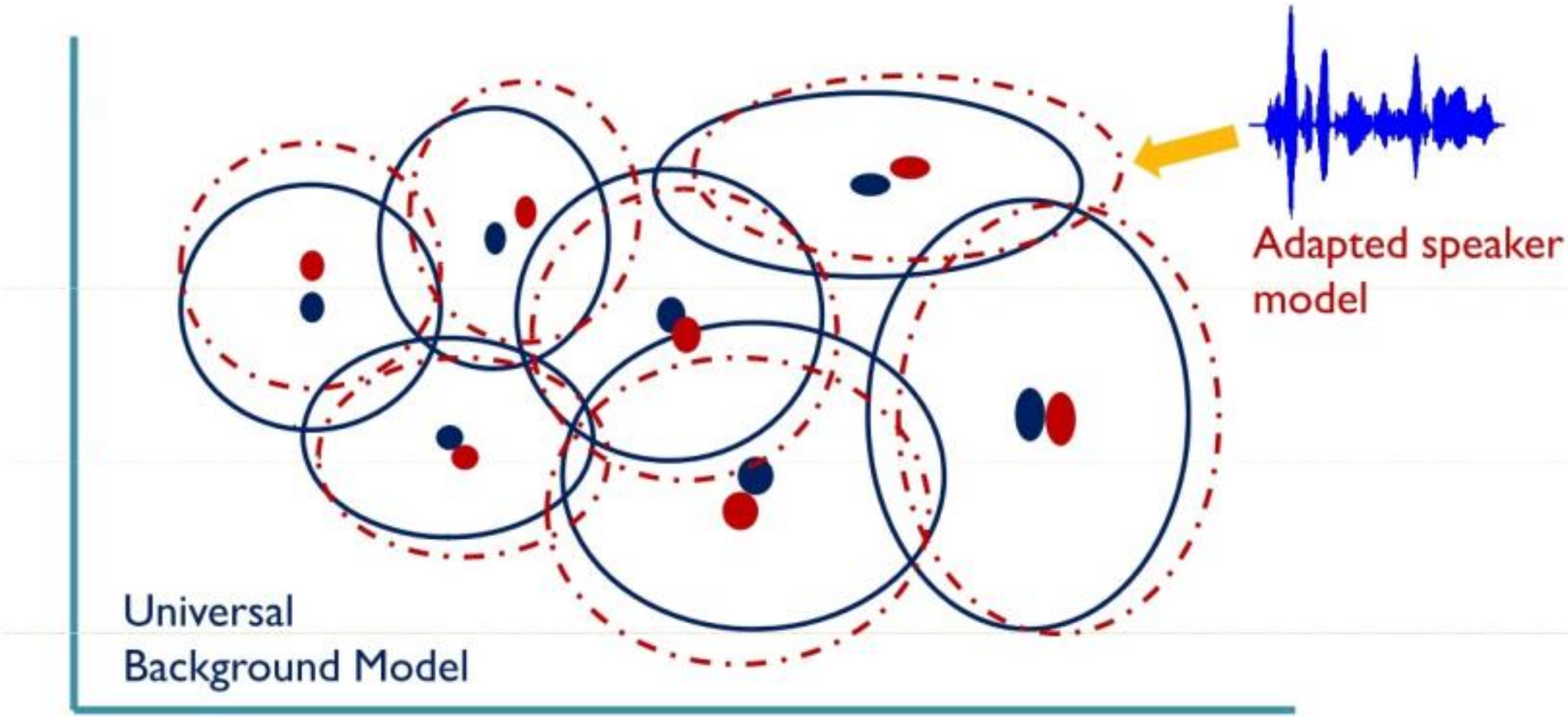
GMM-UBM Speaker Verification

- A GMM, namely the universal background model (UBM), is trained to represent the speech of the general population.
- Speaker GMM (Target model) is established by adjusting UBM by using MAP adaptation.



MAP Adaption

- In practice, only the mean vectors will be adapted:



Maximum a Posteriori (MAP)

- The MAP algorithm finds the parameters of target-speaker's GMM given UBM parameters
- First step is the same as EM. Given T_s acoustic vectors $\mathcal{X}^{(s)} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{T_s}\}$ from speaker s, we compute the statistics:

Probability estimated by the UBM

$$n_c = \sum_{t=1}^{T_s} \gamma_t(c) \quad \text{and} \quad E_c(\mathcal{X}^{(s)}) = \frac{1}{n_c} \sum_{t=1}^{T_s} \gamma_t(c) \mathbf{x}_t$$

- Adapt UBM parameters by

$$\boldsymbol{\mu}_c^{(s)} = \alpha_c E_c(\mathcal{X}^{(s)}) + (1 - \alpha_c) \boldsymbol{\mu}_c^{\text{ubm}}$$

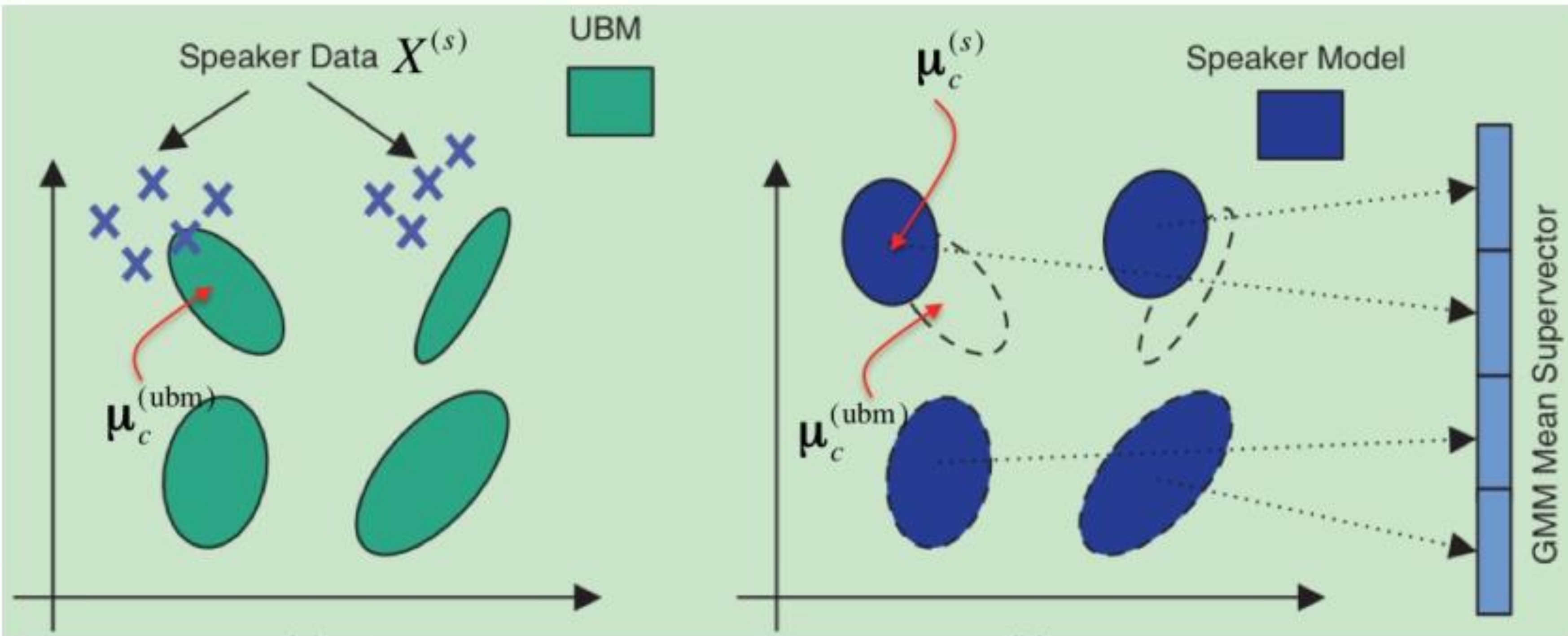
where

$$\alpha_c = \frac{n_c}{n_c + r}$$

and r is called the relevance factor. Typically, $r = 16$.

MAP Adaption

- Adapt the UBM model to each speaker using the MAP algorithm:[1]



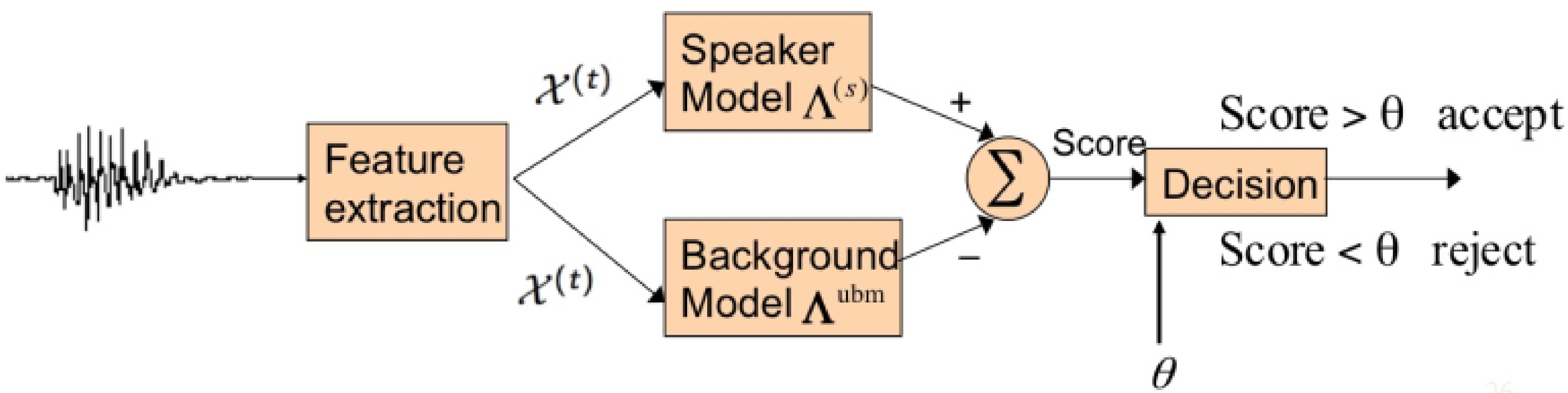
$$\boldsymbol{\mu}_c^{(s)} = \alpha_c E_c(\mathcal{X}^{(s)}) + (1 - \alpha_c)\boldsymbol{\mu}_c^{\text{ubm}}$$

- $\alpha_c \to 1$ when $\mathcal{X}^{(s)}$ comprises lots of data and $\alpha_c \to 0$ otherwise.

GMM-UBM Scoring

- Given the acoustic vectors $\mathcal{X}^{(t)}$ from a test speaker and a claimed identity s, speaker verification can be formulated as a 2-class hypothesis problem:
  - H_0: $\mathcal{X}^{(t)}$ comes from the true speaker s
  - H_1: $\mathcal{X}^{(t)}$ comes from an impostor
- Verification score is a log-likelihood ratio:

$$S_{\mathrm{LR}}(\mathcal{X}^{(t)}|\Lambda^{(s)}, \Lambda^{\mathrm{ubm}}) = \log p(\mathcal{X}^{(t)}|\Lambda^{(s)}) - \log p(\mathcal{X}^{(t)}|\Lambda^{\mathrm{ubm}})$$



Conclusion

- **Generative Models**
- **Discrimination Models**
- **GMM**
- **EM Algorithm**