Techniques d'apprentissage

IFT 601

Mélange de gaussiennes

Par

Pierre-Marc Jodoin

Mélange de gaussiennes

- Le but est de classifier des données **SANS ENSEMBLE D'APPRENTISSAGE.**
- Pour ce faire, on va regrouper ensemble les données « les plus similaires ».

2

Exemple

Segmentation d'images

3

Quelques définitions

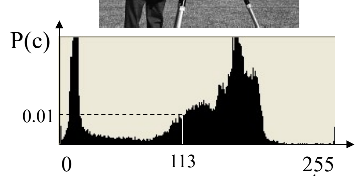
Parfois l'histogramme est normalisé par le nombre de pixels dans l'image:

$$H'(c) = P(c) = \frac{\text{Nb pixels d'intensité c}}{\text{Nb total de pixels dans l'image}}$$

Ainsi défini, *P(c)* donne une idée de la probabilité d'occurrence d'un pixel de niveau de gris "c".

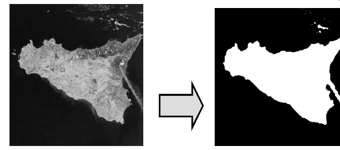
$$\sum_{c=0}^{255} P(c) = 1$$

Si je tire un pixel au hasard dans l'image, j'ai 1% de chance qu'il soit d'intensité 113.



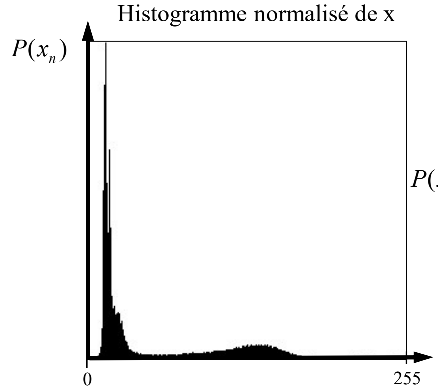
4

Quelques définitions



x → t

Histogramme normalisé de x



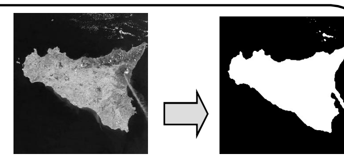
$P(x_n)$ Distribution des niveaux de gris dans l'image *x*.

$P(x_n = a)$ Peut se lire : probabilité d'observer un pixel de niveau de gris « a » dans l'image *t*.

exemple: si $P(x_n = 15) = \mathbf{0.09}$ **alors**

si je tire au hasard un pixel dans l'image x, j'aurai 9 pourcents de chance qu'il soit d'intensité 15.

Quelques définitions



x → t

$P(x_n = a)$ probabilité d'observer un pixel de niveau de gris <u>a</u> dans l'image *x*.

$P(x_n = a, mer)$ est une **probabilité jointe** qui se lit : la probabilité d'observer un pixel de niveau de gris <u>a</u> dans l'image *x* **<u>ET</u>** faisant partie de la classe mer.

$P(mer)$ Probabilité **a priori** d'observer un pixel appartenant à la classe *mer*.

$$\boxed{P(x_n = a, mer) = P(mer)P(x_n = a \mid mer)}$$

$P(x_n = a \mid mer)$ est une **probabilité conditionnelle** qui se lit : la probabilité d'observer un pixel de niveau de gris <u>a</u> dans l'image *x* **<u>ÉTANT DONNÉ</u>** qu'il appartienne à la classe mer.

7

Quelques définitions

$$P(x_n, t_n) = P(t_n) \times P(x_n \mid t_n)$$

terme de vraisemblance
(*likelihood* en anglais)

terme *a priori*

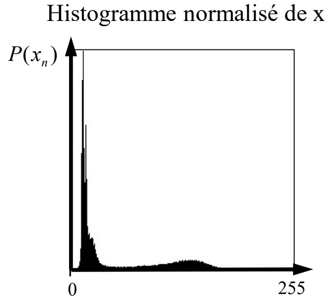
Probabilité jointe

$t_n \in \{mer, terre\}$

8

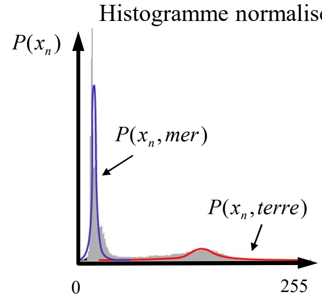
Quelques définitions

Histogramme normalisé de x



$=$

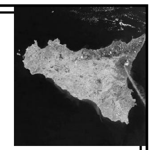
Histogramme normalisé de x



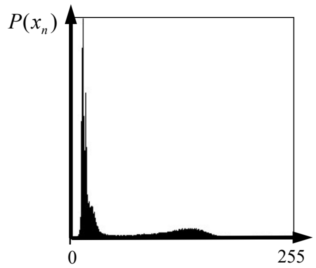
$P(x_n, terre)$	Distribution des niveaux de gris des pixels « terre »
$P(x_n, mer)$	Distribution des niveaux de gris des pixels « mer »

9

Quelques définitions

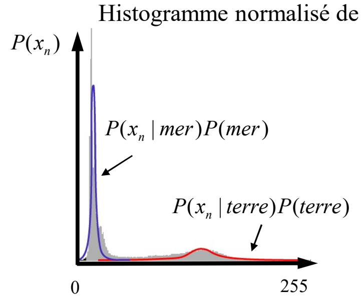


Histogramme normalisé de x



$=$

Histogramme normalisé de x



$P(x_n \mid terre)$	Vraisemblance des niveaux de gris des pixels « terre »
$P(terre)$	A priori (ou proportion) des pixels « terre »

10

Trouver le *meilleur* seuil sans supervision

Une autre définition : **mélange**

De façon plus générale, un mélange s'exprime mathématiquement de la façon suivante:

$$\boxed{P(x_n) = \sum_c P(c)P(x_n \mid c)}$$

Si $P(x_n \mid c)$ est une gaussienne, alors on parlera d'un mélange **de gaussiennes**

11

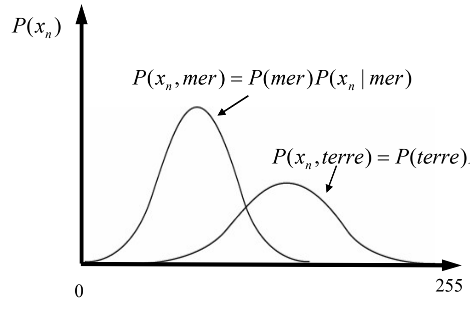
12

Trouver le *meilleur* seuil sans supervision

Étant donné un mélange de gaussiennes dont on connait les paramètres

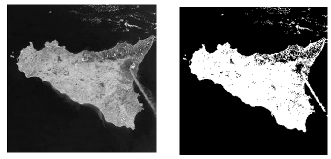
$$\{(P(mer), \mu_{mer}, \sigma_{mer}), (P(terre), \mu_{terre}, \sigma_{terre})\}$$

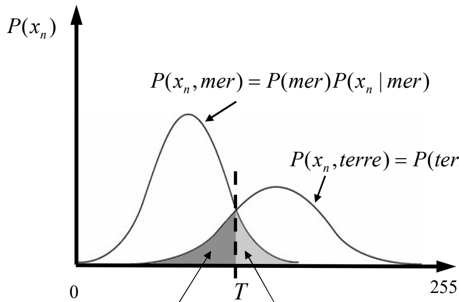
comment trouver le « meilleur » seuil *T?*
Pour y arriver on doit quantifier l'erreur de classification d'un seuil *T*. Le seuil retenu sera celui associé à la plus petite erreur.



13

Trouver le *meilleur* seuil sans supervision





14

Trouver le *meilleur* seuil sans supervision

$$\boxed{\frac{P(x_n \mid mer)}{P(x_n \mid terre)} \underset{\text{terre}}{\overset{\text{mer}}{\gtrless}} \tau \frac{P(terre)}{P(mer)}}$$

Très souvent, on suppose que la vraisemblance de chaque classe se distribue suivant une gaussienne:

$$P(x_n \mid mer) = \frac{1}{\sqrt{2\pi}\sigma_{mer}} \exp\left(-\frac{(x_n - \mu_{mer})^2}{2\sigma_{mer}^2} \right)$$

$$P(x_n \mid terre) = \frac{1}{\sqrt{2\pi}\sigma_{terre}} \exp\left(-\frac{(x_n - \mu_{terre})^2}{2\sigma_{terre}^2} \right)$$

où

μ_{mer} : intensité moyenne des pixels appartenant à la classe *mer*.

σ_{mer} : écart - type de l'intensité des pixels appartenant à la classe *mer*.

15

L'algorithme du seuil « optimal »

Algorithme du seuil « optimal »

1. POUR CHAQUE pixel n de l'image x FAIRE

$$P_m = \frac{P(mer)}{\sqrt{2\pi}\sigma_{mer}} \exp\left(-\frac{(x_n - \mu_{mer})^2}{2\sigma_{mer}^2} \right)$$

$$P_t = \tau \frac{P(terre)}{\sqrt{2\pi}\sigma_{terre}} \exp\left(-\frac{(x_n - \mu_{terre})^2}{2\sigma_{terre}^2} \right)$$

SI $P_t > P_m$ ALORS

$t_n = 1$ /* Étiquette « *terre* » au pixel (i,j) */

SINON

$t_n = 0$ /* Étiquette « *mer* » au pixel (i,j) */

Note : Ceci est un algorithme de segmentation de type **"génératif"**

16

17

Algorithme des nuées dynamiques (*K-means*)

Un des problèmes avec les algorithmes de classification « probabilistes » vus précédemment est qu'ils **exigent de connaître les paramètres (moyenne et écart-type pour une gaussienne)** de chaque classe avant de segmenter l'image d'entrée.

Ce sont des algorithmes dits **supervisés**. Ils ne sont pas autonomes et exigent que l'utilisateur fournisse des données fondamentales pour le traitement.

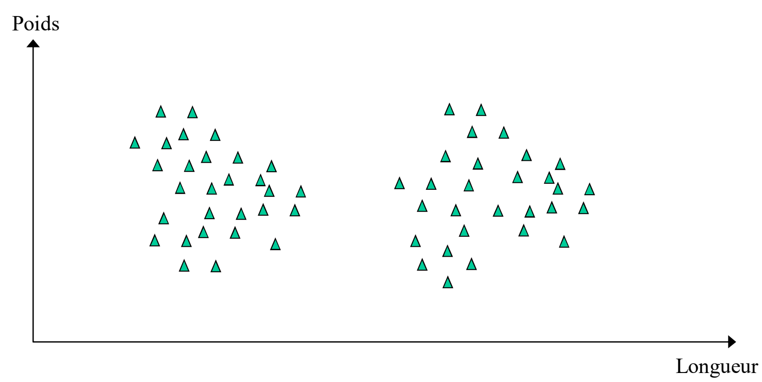
Question: comment segmenter automatiquement une image à l'aide d'une distribution probabiliste (ici un mélange de gaussiennes)? En d'autres mots, comment estimer **SANS SUPERVISION** les paramètres de la mixture de gaussienne:

$$\mu_{t_n}, \sigma_{t_n} \textbf{ et } P(t_n)?$$

18

Algorithme des nuées dynamiques (*K-means*)

Exemple schématique de l'algorithme des nuées dynamiques

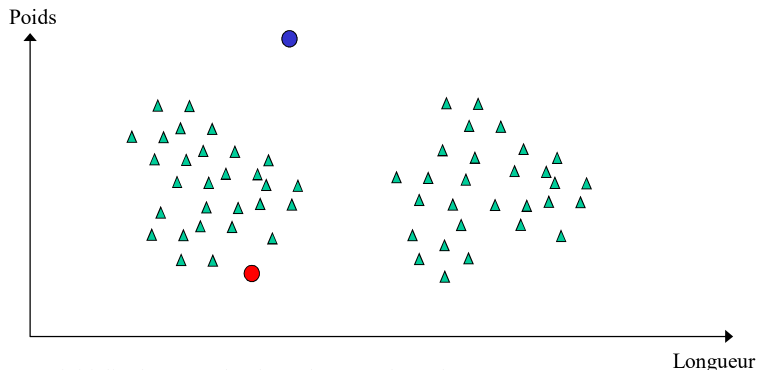


Voici 60 observations. Comment faire pour grouper ces données en deux classes et estimer la moyenne (ou le centre de masse) de ces classes ?

19

Algorithme des nuées dynamiques (*K-means*)

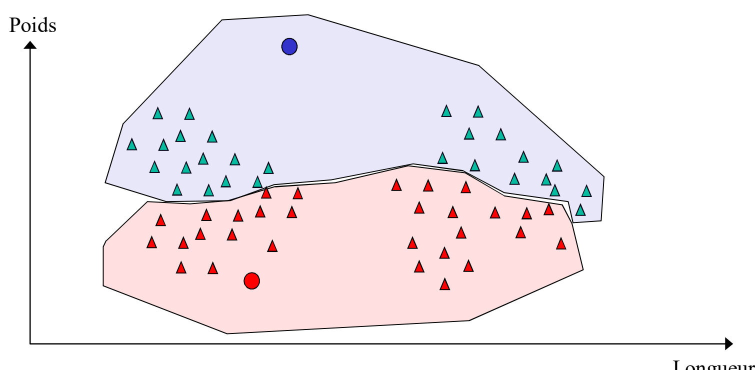
Exemple schématique de l'algorithme des K-moyennes



On initialise le centre des deux classes au hasard

Algorithme des nuées dynamiques (*K-means*)

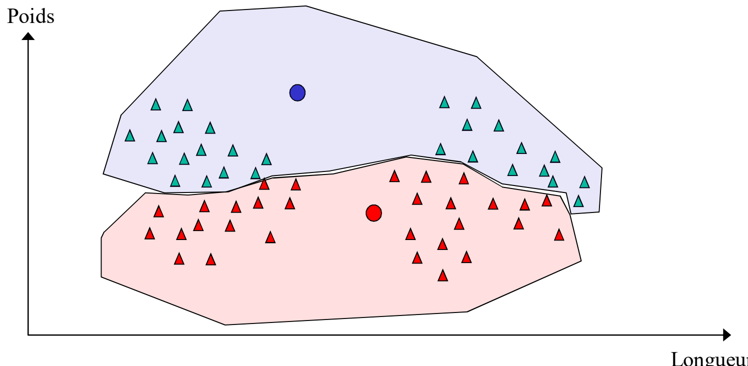
Exemple schématique de l'algorithme des K-moyennes



On associe chaque donnée au centre le plus près

Algorithme des nuées dynamiques (*K-means*)

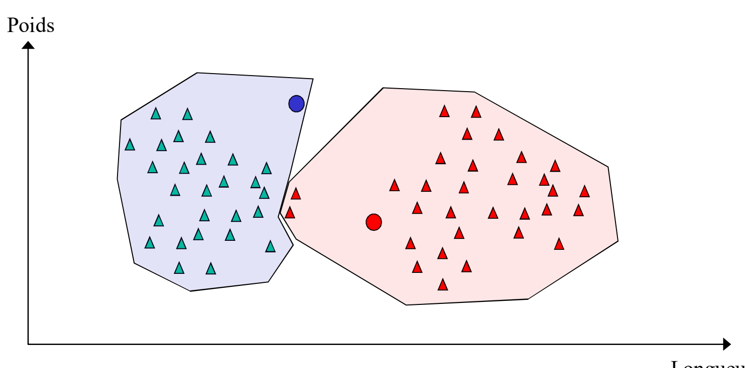
Exemple schématique de l'algorithme des K-moyennes



On déplace les deux centres sur la moyenne des données de chaque classe

Algorithme des nuées dynamiques (*K-means*)

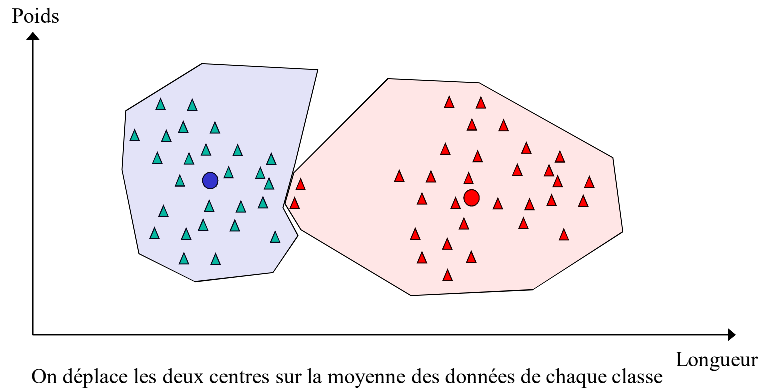
Exemple schématique de l'algorithme des K-moyennes



On associe chaque donnée ▲ au centre le plus près

Algorithme des nuées dynamiques (*K-means*)

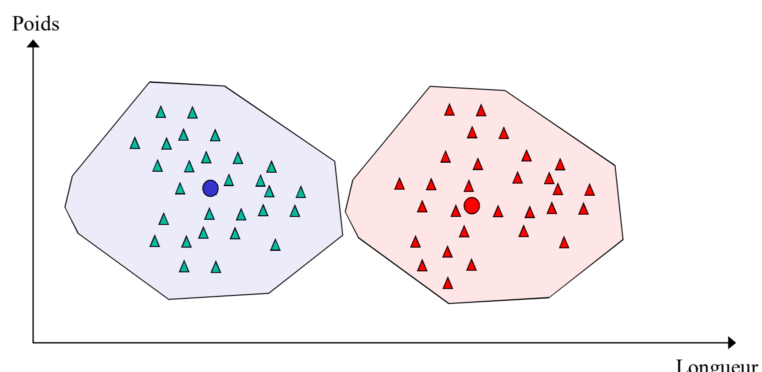
Exemple schématique de l'algorithme des K-moyennes



On déplace les deux centres sur la moyenne des données de chaque classe

Algorithme des nuées dynamiques (*K-means*)

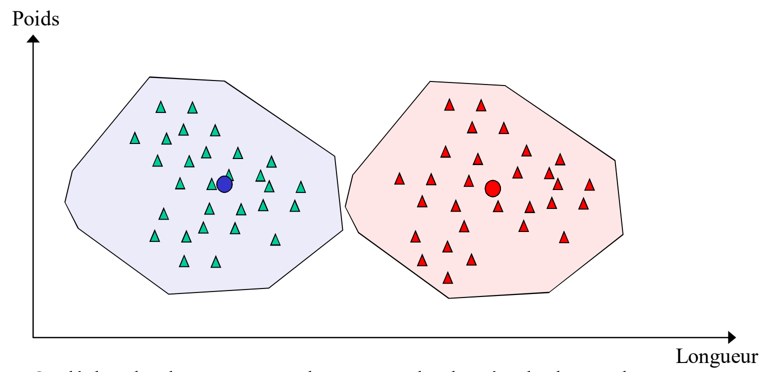
Exemple schématique de l'algorithme des K-moyennes



On associe chaque donnée ▲ au centre le plus près

Algorithme des nuées dynamiques (*K-means*)

Exemple schématique de l'algorithme des K-moyennes



On déplace les deux centres sur la moyenne des données de chaque classe

Lorsque l'algorithme converge, et correspondent à **la moyenne de chaque classe**

26

Algorithme des nuées dynamiques (*K-means*)

L'algorithme des nuées dynamiques est un des algorithmes non supervisés parmi les plus simples. Parce qu'il est simple, cet algorithme fait de nombreuses hypothèses simplificatrices parmi lesquels

1. *P(mer) = P(terre)*
2. Les données de chaque classe se distribuent suivant des gaussiennes ayant **le même écart type**
3. Chaque donnée ne peut appartenir qu'à **une seule classe à la fois**.

27

Algorithme des nuées dynamiques (*K-means*)

On a vu que le seuil « optimal » T est celui pour lequel

$$P(terre)P(T \mid terre) = P(mer)P(T \mid mer)$$

$$P(terre) \times N(T; \mu_{terre}, \sigma_{terre}) = P(mer) \times N(T; \mu_{mer}, \sigma_{mer})$$

$$N(T; \mu_{terre}, \sigma_{terre}) = N(T; \mu_{mer}, \sigma_{mer}) \qquad \text{(hypothèse 1)}$$

$$\frac{1}{\sqrt{2\pi}\sigma_{terre}} e^{-\frac{(T-\mu_{terre})^2}{2\sigma_{terre}^2}} = \frac{1}{\sqrt{2\pi}\sigma_{mer}} e^{-\frac{(T-\mu_{mer})^2}{2\sigma_{mer}^2}}$$

$$(T - \mu_{terre})^2 = (T - \mu_{mer})^2 \qquad \text{(hypothèse 2)}$$

28

Algorithme des nuées dynamiques (*K-means*)

1. Calculer le champ d'étiquettes *t*

Si μ_{terre} et μ_{mer} sont connus, alors on peut facilement calculer *t*:

POUR CHAQUE pixel *n* de l'image *x* FAIRE
  SI $(x_n - \mu_{terre})^2 < (x_n - \mu_{mer})^2$ ALORS
    $t_n = 1$ /* Étiquette « *terre* » au pixel (i,j) */
  SINON
    $t_n = 0$ /* Étiquette « *mer* » au pixel (i,j) */

29

Algorithme des nuées dynamiques (*K-means*)

2. Calculer μ_{terre} et μ_{mer}

Si t est connu, alors calculer μ_{terre} et μ_{mer} est facile

$\mu_{mer} = \mu_{terre} = 0$
POUR CHAQUE pixel *n* de l'image x FAIRE
    SI $t_n == 1$ ALORS
        $\mu_{terre} = \mu_{terre} + x_n$
    SINON
        $\mu_{mer} = \mu_{mer} + x_n$

$\mu_{mer} = \mu_{mer}$ / Nombre de pixels "Mer"
$\mu_{terre} = \mu_{terre}$ / Nombre de pixels "Terre"

30

Nuées dynamiques pour 2 classes

0. μ_{mer} = un pixel x_n pris au hasard
   μ_{terre} = un autre pixel x_n pris au hasard

1. POUR CHAQUE pixel *n* de l'image x FAIRE
    SI $(x_n - \mu_{terre})^2 < (x_n - \mu_{mer})^2$ ALORS
        $t_n = 1$ /* Étiquette « *terre* » au pixel (i,j) */
    SINON
        $t_n = 0$ /* Étiquette « *mer* » au pixel (i,j) */

Estimer t

2. $\mu_{mer} = \mu_{terre} = 0$
3. POUR CHAQUE pixel *n* de l'image x FAIRE
    SI $t_n == 1$ ALORS
        $\mu_{terre} = \mu_{terre} + x_n$
    SINON
        $\mu_{mer} = \mu_{mer} + x_n$

4. $\mu_{mer} = \mu_{mer}$ / Nombre de pixels "Mer"
   $\mu_{terre} = \mu_{terre}$ / Nombre de pixels "Terre"

5. SI μ_{terre} *et* μ_{mer} ne changent plus ALORS arrêter SINON retour à 1

Recalculer μ_{terre} *et* μ_{mer}

31

Algorithme des nuées dynamiques (*K-means*)

Nuées dynamiques pour un nombre arbitraire de classes

0. Initialiser la moyenne de chaque classe μ_c

1. POUR CHAQUE pixel n de l'image x FAIRE

   t_n = étiquette de la classe dont la moyenne μ_c est la plus proche de x_n.

3. Recalculer la moyenne de chaque classe.

4. SI les moyennes ne changent plus ALORS arrêter SINON retour à 1

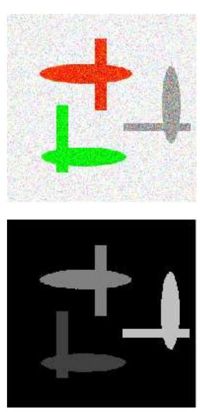
Pour en savoir plus au sujet de K-Means :

- Tutoriel d'andrew Moore : http://www.autonlab.org/tutorials/kmeans.html
- Livre de D.MacKay, « ***Information Theory, Inference, and Learning Algorithms*** », Chapitre 20. (http://www.inference.phy.cam.ac.uk/mackay/itprnn/book.html)

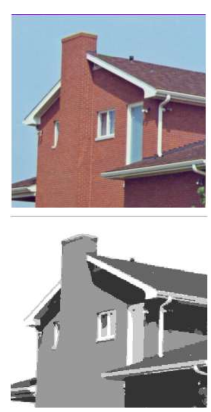
32

Algorithme des nuées dynamiques (*K-means*)

Segmentation 4 classes



Segmentation 5 classes



33

34

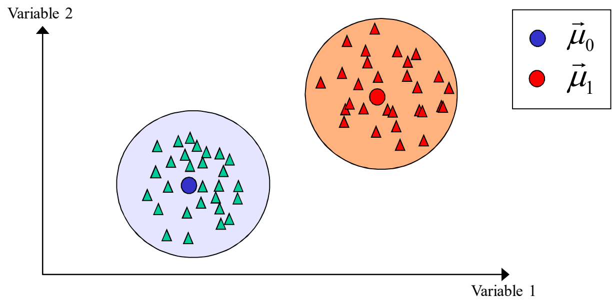
Bref aparté

La quantification vectorielle

(application des nuées dynamiques)

35

Nuées dynamiques et la compression



On peut compresser un ensemble d'observations y (ici ▲ et ▲) en remplaçant chaque observation par l'indice de la classe à laquelle elle appartient (ici 0 et 1). Par exemple, dans le cas présent, pour représenter *N* observations, il faut **N bits + 4 floats** après compression au lieu des **2N floats** requis avant compression.

Dans ce cas, $\{\mu_0, \mu_1\}$ forme un **dictionnaire** (*codebook*).

36

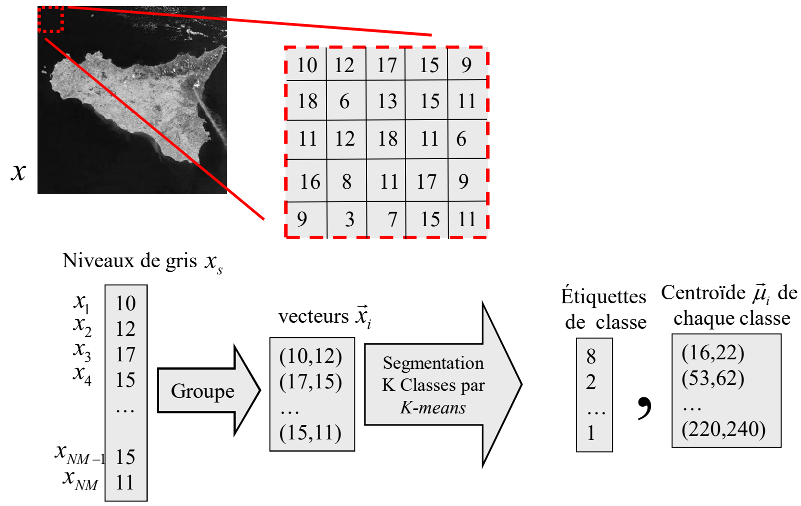
Compression d'images

On peut appliquer la même idée pour compresser une image. Toutefois, dans le but d'optimiser le facteur de compression de ce type d'approche, **on groupe les pixels ensemble avant de segmenter**. C'est ce qu'on appelle la compression par **« quantification vectorielle »**.

Afin d'illustrer cette méthode, prenons le cas le plus simple, à savoir grouper les pixels par 2.

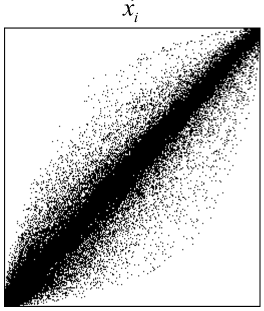
37

Quantification vectorielle



Quantification vectorielle

$\vec{x}_i$



Chaque groupe de 2 niveaux de gris forme un point dans un espace 2D.

$\vec{x}_i$ avec $\vec{\mu}_i$

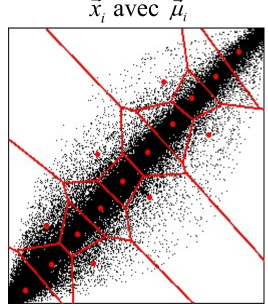
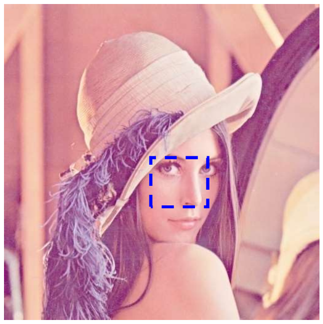


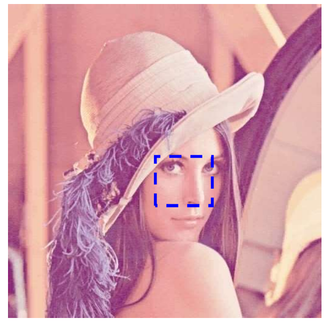
Illustration de K=16 centroïdes de classes

http://www.gamasutra.com/features/20010416/ivanov_01.htm

Exemple



Sans compression, 24 bits par pixel



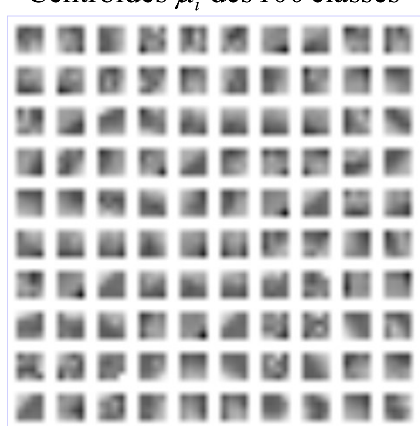
Après compression, K=1024, 2.6 bits par pixel, groupes de 4x4 pixels

http://www.gamasutra.com/features/20010416/ivanov_01.htm

Exemple

Exemple d'un dictionnaire de K=100 classes pour des groupes de 4x4 pixels.

Centroïdes $\vec{\mu}_i$ des 100 classes



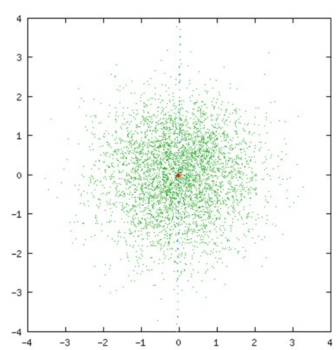
41

Problème avec nuées dynamiques

Malheureusement, les nuées dynamiques souffrent d'un problème fondamental : lorsque le nombre de classes est élevé, il a parfois tendance à **laisser certaines classes vides.** Afin de remédier à ce problème, on utilise généralement **l'approche LBG** (Linde, Buzo et Gray) qui consiste à commencer avec un nombre restreint de classes qu'on subdivise de façon récursive jusqu'à atteindre le nombre de classes désiré.

42

Exemple 2D de l'approche LBG



[Crédit Christophe Charrier]

43

Algorithme LBG

0. Regrouper les pixels en groupes de $n \times n$ pixels.
1. K = 1;   // K indique le nombre de classes
2. $\vec{\mu}_0 =$ initialisation aléatoire.

FAIRE

Générer K nouveaux centroïdes placés à côté des K centroïdes existants
K = K*2;
nuées dynamiques pour calculer les K centroïdes $\{\vec{\mu}_0, \vec{\mu}_1, ..., \vec{\mu}_{K-1}\}$

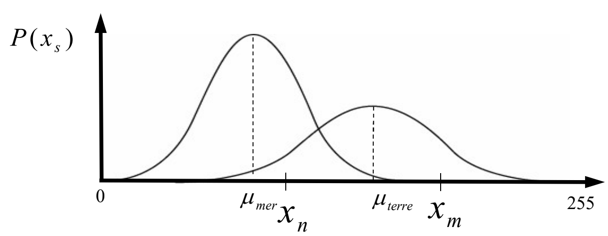
TANT QUE K != nombre de classes désiré

Fin de l'aparté

Algorithme *Soft K-means*

L'algorithme des nuées dynamiques émet l'hypothèse que chaque donnée n'appartient qu'à une seule classe à la fois, ce qui est faux! Pour remédier à cette hypothèse parfois abusive, l'algorithme *« soft » k-means* permet à chaque site d'appartenir à toutes les classes en même temps, mais avec des proportions différentes.

Exemple:



le pixel **n** appartient à la classe mer dans une proportion de 0.7 et à la classe terre dans une proportion de 0.3

le pixel **m** appartient à la classe mer dans une proportion de 0.02 et à la classe terre dans une proportion de 0.98

Algorithme *Soft K-means*

Avec *K-means*, on cherche à minimiser l'erreur quadratique globale

$$J_{km} = \sum_n (x_n - \mu_{t_n})^2$$

Avec *Soft K-means*, on cherche à minimiser l'erreur pondérée globale

$$J_{skm} = \sum_n \sum_{c=1}^{N_c} P(c \mid x_n)(x_n - \mu_c)^2$$

Où c est une étiquette de classe (c ={*mer,terre*})

N_c est le nombre total de classes (2 dans l'exemple terre-mer)

$P(c \mid x_n)$ est la proportion avec laquelle x_n appartient à la classe c.

$$P(c \mid x_n) = \frac{e^{-\beta|x_n - \mu_c|}}{\sum_r e^{-\beta|x_n - \mu_r|}}$$

48

Algorithme *soft k-means* pour deux classes

0. μ_{mer} = un pixel t_n pris au hasard
   μ_{terre} = un autre pixel t_n pris au hasard

1. POUR CHAQUE pixel *n* de l'image x FAIRE

$$d_t = \beta|\mathrm{x_n} - \mu_{terre}|$$
$$d_m = \beta|\mathrm{x_n} - \mu_{mer}|$$
$$P(mer \mid x_n) = \frac{e^{-d_m}}{e^{-d_m} + e^{-d_t}}$$
$$P(terre \mid x_n) = \frac{e^{-d_t}}{e^{-d_m} + e^{-d_t}}$$

Estimer $P(c \mid x_n)$ pour tous les pixels.

2. $\mu_{mer} = \mu_{terre} = T_{mer} = T_{terre} = 0$
3. POUR CHAQUE pixel *n* de l'image x FAIRE

$$\mu_{mer} = \mu_{mer} + P(mer \mid x_n) \times x_n ; T_{mer} = T_{mer} + P(mer \mid x_n)$$
$$\mu_{terre} = \mu_{terre} + P(terre \mid x_n) \times x_n ; T_{terre} = T_{terre} + P(terre \mid x_n)$$

4. $\mu_{mer} = \mu_{mer} / T_{mer}$
   $\mu_{terre} = \mu_{terre} / T_{terre}$

Recalculer μ_{terre} *et* μ_{mer}

5. SI μ_{terre} *et* μ_{mer} ne changent plus ALORS arrêter SINON retour à 1

49

Algorithme *Soft K-means*

Algorithme de soft k-means pour un nombre arbitraire de classes

0. Initialiser la moyenne de chaque classe μ_c

1. POUR CHAQUE pixel *n* de l'image *x* FAIRE

   Calculer pour chaque classe "c": $P(c \mid x_n) = \dfrac{e^{-d_c}}{\sum_r e^{-d_r}}$

3. Calculer la moyenne de chaque classe « c »

$$\mu_c = \frac{\sum_n P(c \mid x_n) x_n}{\sum_n P(c \mid x_n)}$$

4. SI les moyennes μ_c ne changent plus ALORS arrêter SINON retour à 1

Après convergence, on peut estimer le champ d'étiquettes *t* à l'aide d'un algorithme génératif

50

Algorithme *Soft K-means*

$$P(c \mid x_n) = \frac{e^{-\beta |x_n - \mu_c|}}{\sum_r e^{-\beta |x_n - \mu_r|}}$$

Il est intéressant de noter que *K-means* est un cas particulier de « soft » K-means.

En effet, lorsque $\beta \to \infty$, $P(c \mid x_n) = \{0,1\}$ et donc $J_{skm} = J_{km}$

Pour de plus amples détails sur les algorithmes *K-means* et *Soft K-means*, voir :

Duda,Hart,Stork « ***Pattern Classification, 2nd ed.*** », Chapitre 10.4.

MacKay, « ***Information Theory, Inference, and Learning Algorithms*** », Chapitre 20. (http://www.inference.phy.cam.ac.uk/mackay/itprnn/book.html)

51

52

Expectation-Maximization

K-means et soft *k-means* sont deux algorithmes qui ne font qu’estimer la **moyenne** de chaque classe.

Un algorithme plus puissant du nom de «E-M» permet d’estimer TOUS les paramètres de la mixture de gaussiennes, c’est-à-dire:

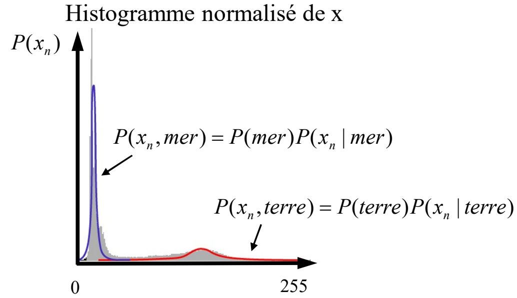
$$\mu_i, \sigma_i \text{ et } P(t_n = i)$$

pour chaque gaussienne.

53

Expectation-Maximization

Histogramme normalisé de x



$$P(x_n) = \sum_c P(c)P(x_n \mid c)$$

Puisqu'on a un mélange de **gaussiennes**

$$P(x_n \mid \theta) = \sum_c P(c)P(x_n \mid c, \theta_c)$$

où $\theta_c = \{\mu_c, \sigma_c\}$

54

Expectation-Maximization

$P(x_n \mid \theta) = \sum_c P(c)P(x_n \mid c, \theta_c)$ probabilité d'observer un pixel x_n

$P(x \mid \theta) = \prod_n P(x_n \mid \theta)$ probabilité d'observer tous les pixels de l'image x

Vraisemblance des données observées

Les meilleurs paramètres θ sont ceux qui **maximisent la vraisemblance**

$$\begin{aligned}\theta &= \arg\max_{\theta'} \prod_n P(x_n \mid \theta') \\ &= \arg\max_{\theta'} \sum_n \ln P(x_n \mid \theta')\end{aligned}$$

55

Expectation-Maximization

$\mu_c = \dfrac{\sum_n P(c \mid x_n, \theta) x_n}{\sum_n P(c \mid x_n, \theta)}$ $\sigma_c^2 = \dfrac{\sum_n P(c \mid x_n, \theta)(x_n - \mu_c)^2}{\sum_n P(c \mid x_n, \theta)}$ $P(c) = \dfrac{1}{N_n} \sum_n P(c \mid x_n, \theta)$	$P(c \mid x_n, \theta) = \dfrac{P(x_n \mid c, \theta_c) P(c)}{\sum_i P(x_n \mid i, \theta_i) P(i)}$ $P(x_n \mid c, \theta_c) = \dfrac{1}{\sqrt{2\pi}\sigma_c} e^{-\frac{(x_n - \mu_c)^2}{2\sigma_c^2}}$

60

Algorithme des E-M pour ***2* classes**

0. $P(mer), P(terre), \mu_{mer}, \sigma_{mer}, \mu_{terre}, \sigma_{terre} \leftarrow Init$

1. FAIRE

E:

Calculer $P(mer \mid x_n, \theta)$ et $P(terre \mid x_n, \theta)$ pour tous les pixels "n":

$P(mer), P(terre), \mu_{mer}, \sigma_{mer}, \mu_{terre}, \sigma_{terre} \leftarrow 0$

M:

2. POUR CHAQUE pixel *n* de l'image *x* FAIRE

$\mu_{mer} = \mu_{mer} + P(mer \mid x_n, \theta) x_n;$ $\qquad \mu_{terre} = \mu_{terre} + P(terre \mid x_n, \theta) x_n$

$\sigma^2_{mer} = \sigma^2_{mer} + P(mer \mid x_n, \theta)(x_n - \mu_{mer})^2;$ $\qquad \sigma^2_{terre} = \sigma^2_{terre} + P(terre \mid x_n, \theta)(x_n - \mu_{terre})^2$

$P(mer) = P(mer) + P(mer \mid x_n, \theta);$ $\qquad P(terre) = P(terre) + P(terre \mid x_n, \theta)$

$\mu_{mer} = \mu_{mer} / P(mer);$ $\qquad \mu_{terre} = \mu_{terre} / P(terre)$

$\sigma^2_{mer} = \sigma^2_{mer} / P(mer);$ $\qquad \sigma^2_{terre} = \sigma^2_{terre} / P(terre)$

$P(mer) = P(mer) / N_s;$ $\qquad P(terre) = P(terre) / N_s$

3. TANT QUE μ_c, σ_c et $P(c)$ stabilisent

Algorithme des E-M pour ***K>2* classes**

0. $P(c), \mu_c, \sigma_c \leftarrow$ initialiser les paramètres de chaque classe "c"

1. FAIRE

Calculer $P(c \mid x_n, \theta)$ pour tous les pixels "n" et toutes les classes "c"

$$P(c \mid x_n, \theta) = \frac{P(x_n \mid c, \theta_c) P(c)}{\sum_i P(x_n \mid i, \theta_i) P(i)}$$

E

Calculer les paramètres de chaque gaussienne :

$$\mu_c = \frac{\sum_n P(c \mid x_n, \theta) x_n}{\sum_n P(c \mid x_n, \theta)}$$

$$\sigma_c^2 = \frac{\sum_n P(c \mid x_n, \theta)(x_n - \mu_c)^2}{\sum_n P(c \mid x_n, \theta)}$$

$$P(c) = \frac{1}{N_n} \sum_n P(c \mid x_n, \theta)$$

M

3. TANT QUE μ_c, σ_c et $P(c)$ ne sont pas stabilisés

Expectation-Maximization

Pour de plus amples détails sur l'algorithmes *E-M*, voir :

Duda,Hart,Stork « ***Pattern Classification, 2nd ed.*** », Chapitre *10.4*.

MacKay, « ***Information Theory, Inference, and Learning Algorithms*** », Chapitre *20*. (http://www.inference.phy.cam.ac.uk/mackay/itprnn/book.html)

Bishop, « ***Neural Networks for Pattern Recognition*** », Chapitre *2.6*

Wikipedia

En résumé...

Algorithme du seuil	Simple, mais le seuil doit être donné par l'usager (**approche supervisée**).
Algorithme probabiliste du seuil	Si on connaît la distribution probabiliste de chaque classe, le seuil optimal peut être calculé.
Algorithme des nuées dynamiques	Algorithme permettant d'estimer automatiquement la **moyenne** de chaque classe (gaussiennes). $\longrightarrow$ **Quantification vectorielle**
Algorithme Soft K-Means	Algorithme permettant d'estimer automatiquement la **moyenne** de chaque classe (gaussiennes)..
Algorithme EM	Algorithme permettant d'estimer automatiquement la **moyenne, la variance et la proportion** de chaque classe (gaussiennes).

64

65

Estimation du nombre de classes

Nous avons vu qu'**EM** permet d'estimer les paramètres d'un mélange de gaussiennes à savoir : μ_c, σ_c et $P(c)$

Toutefois, le nombre de classes « ***K*** » doit toujours être spécifié par un utilisateur.

Nous verrons donc 2 méthodes pour estimer ***K*** automatiquement

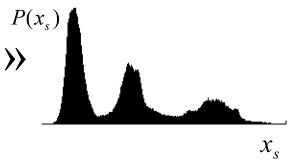
1. **Test de « gaussianité »**
2. **MDL (*Minimum Description Lenght*)**

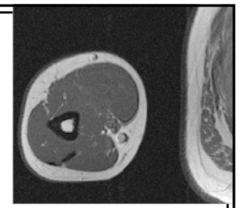
Test de « gaussianité »

Soient des données $\{x_s \mid s \in S\}$ se distribuant suivant une mixture de Gaussiennes dont les paramètres doivent être estimés (incluant ***K*** le nombre de classes).

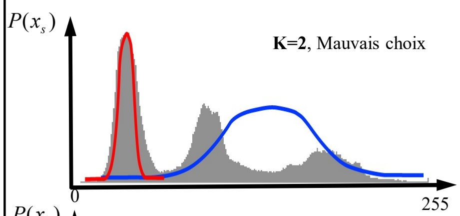
L'idée derrière le test Gaussien est de sélectionner un certain K pour lequel les données appartenant à une classe « i » se **distribuent vraiment selon la i-ème gaussienne**.

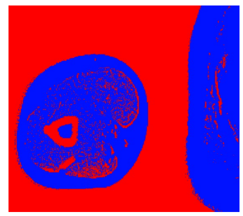
Test de « gaussianité »



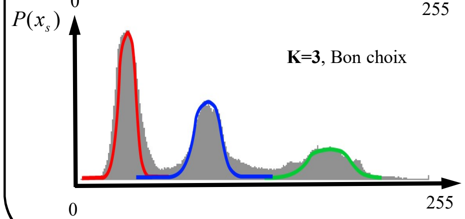


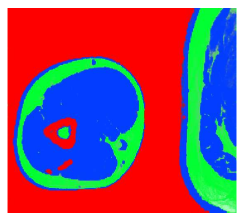
Exemple





- Dist. Gaussienne
- **Dist. Non Gaussienne**





- Dist. Gaussienne
- Dist. Gaussienne
- Dist. Gaussienne

68

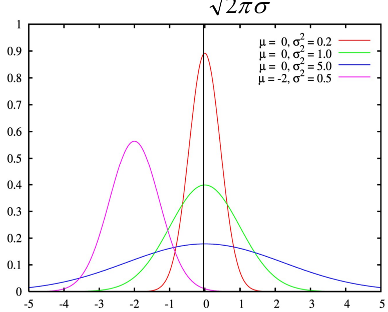
Test de « gaussianité »

Comment savoir si les données d'une classe suivent une distribution gaussienne ?

Test de normalité ou test de « **goodness of fit** ».

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

±σ	68.26894921371%
±2σ	95.44997361036%
±3σ	99.73002039367%



Wikipedia

69

Test de « gaussianité »

1- Étant donné une gaussienne $N(\mu_c, \sigma_c^2)$ et un ensemble de N valeurs $L = \{x_s\}$

2- TabCmpt[3] = {0,0,0};

3- POUR CHAQUE valeur x_s dans L FAIRE

$dist = |x_s - \mu_c|$

SI $dist < \sigma$ ALORS

TabCmpt[1]++;

SINON SI $dist < 2\sigma$ ALORS

TabCmpt[2]++;

SINON

TabCmpt[3]++;

4- TabCmpt = TabCmpt/N;

5- SI abs(TabCmpt[1]-0.68)>Seuil OU abs(TabCmpt[1]-0.27)>Seuil OU abs(TabCmpt[1]-0.05)>Seuil
RETOUR FAUX;
SINON
RETOUR VRAI;

Estimer le nombre de classes **K** à l'aide d'un test de « gaussianité »

1. POUR ***K*** = 2 à NB_CLASSES_MAX

$\{\mu_c, \sigma_c^2, P(c)\}_{c=1\ldots K} \leftarrow$ Segmentation en K classes avec EM

$x \leftarrow$ Seuil optimal$(\mu_c, \sigma_c^2, P(c))$

fin = VRAI;

2. POUR CHAQUE classes c FAIRE

$L = \{x_s\}_{y_s = c}$ // Mettre dans L tous les pixels associés à la classe "c"

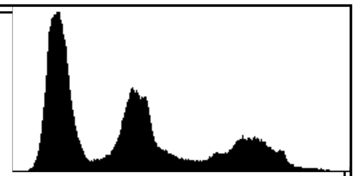
SI les pixels dans "L" **NE SUIVENT PAS** la distribution gaussienne $N(\mu_c, \sigma_c^2)$ ALORS
fin = FAUX;

SI fin == TRUE ALORS
RETOUR K;

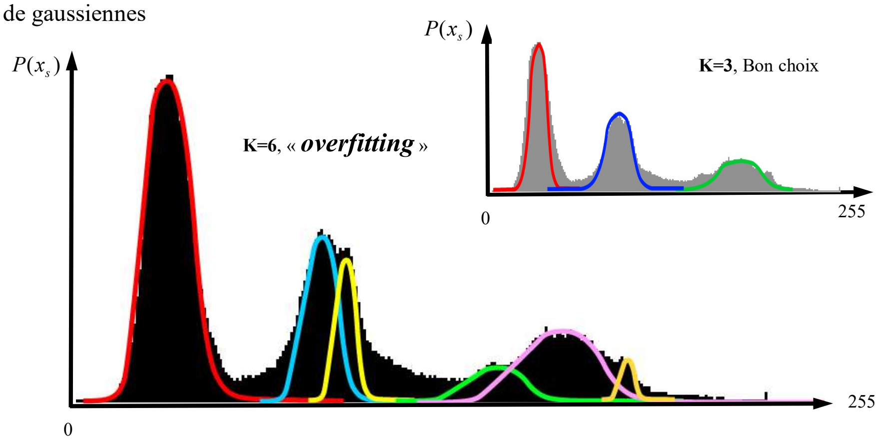
71

MDL

Minimum Description Length



Malheureusement, les tests de normalité ont parfois tendance à choisir un grand nombre de classes « K » lorsque les données ne suivent pas exactement une mixture de gaussiennes



MDL

Minimum Description Length

On cherche un mélange de gaussiennes dont les paramètres :

$$\theta = \{(\mu_1, \sigma_1, P(1)), \ldots, (\mu_K, \sigma_K, P(K))\}$$

et le nombre de classes K vont satisfaire le critère suivant

$$\boxed{K, \theta = \arg\min_{\theta, K} \left[\underbrace{L(x \mid K, \theta)}_{\text{Mesure le « goodness of fit »}} + \underbrace{L(K, \theta)}_{\text{Pénalise les solutions avec un nombre de paramètres élevé.}} \right]}$$

MDL

Minimum Description Length

$$MDL(\theta, K) = L(x \mid \theta, K) + L(\theta, K)$$

Suivant le critère de Rissanen

$$MDL(\theta, K) = -\log P(x \mid \theta) + \frac{1}{2} R \log(ND)$$
$$= -\log \prod_s P(x_s \mid \theta) + \frac{1}{2} R \log(ND)$$

où

N = nombre total de pixels

$$R = K\left(\frac{(D+1)D}{2} + D + 1\right) - 1,\ K = \text{Nombre de classes}$$

D=1 pour images en niveaux de gris
D=3 pour images couleur RGB.

MDL

Minimum Description Length

$$MDL(\theta, K) = -\log \prod_s P(x_s \mid \theta) + \frac{1}{2} R \log(ND)$$

Sachant que les données suivent un mélange de K gaussiennes

$$P(x_s \mid \theta, K) = \sum_{c=1}^{K} P(c) N(x_s \mid \mu_c, \sigma_c)$$

On peut dire que

$$MDL(\theta, K) = -\log \prod_s \left(\sum_{c=1}^{K} P(c) N(x_s \mid \mu_c, \sigma_c) \right) + \frac{1}{2} R \log(ND)$$

$$\boxed{MDL(\theta, K) = -\sum_{\mathrm{s}} \log \left(\sum_{c=1}^{K} P(c) N(x_s \mid \mu_c, \sigma_c) \right) + \frac{1}{2} R \log(ND)}$$

A13

Estimer le nombre de classes **K** à l'aide de MDL

1. POUR $\boldsymbol{K}$ = 2 à NB_CLASSES_MAX FAIRE

2. $\theta = \{\mu_c, \sigma_c^2, P(c)\}_{c=1\ldots K} \leftarrow$ Segmentation en K classes avec EM

3. $MDL[K] = -\sum_s \log\left(\sum_{c=1}^{K} P(c) N(x_s \mid \mu_c, \sigma_c)\right) + \frac{1}{2} R \log(ND)$

4. $k_{optimal} = \arg\min_k MDL[k]$

5. RETOUR $k_{optimal}$

J. Rissanen, « Universal Prior for Integers and Estimation by Minimum DescriptionLength » Annals of Statistics, vol. 11, no. 2, pp. 417-431, 1983.
C. Bouman, « CLUSTER: An Unsupervised Algorithm for Modeling Gaussian Mixtures », 2005 Appendice A

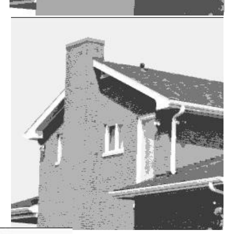
76

Exemple MDL

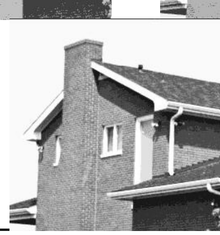
K= 2

K= 3

K= 4



K= 5



K= 12

K: 12	MDL: 2068.578461
K: 11	MDL: 2049.159599
K: 10	MDL: 2029.588927
K: 9	MDL: 2004.600938
K: 8	MDL: 1985.956678
K: 7	MDL: 1970.612420
K: 6	MDL: 1953.424010
K: 5	**MDL: 1909.510934**
K: 4	MDL: 1917.610814
K: 3	MDL: 1933.961104
K: 2	MDL: 1940.469298
K: 1	MDL: 2151.018760

En résumé...

Algorithme du seuil	Simple, mais le seuil doit être donné par l'usager (**<u>approche supervisée</u>**).
Algorithme probabiliste du seuil	Si on connaît la distribution probabiliste de chaque classe, le seuil optimal peut être calculé.
Algorithme des nuées dynamiques	Algorithme permettant d'estimer automatiquement la **<u>moyenne</u>** de chaque classe (gaussiennes).
Algorithme Soft K-Means	Algorithme permettant d'estimer automatiquement la **<u>moyenne</u>** de chaque classe (gaussiennes)..
Algorithme EM	Algorithme permettant d'estimer automatiquement la **<u>moyenne, la variance et la proportion</u>** de chaque classe (gaussiennes).
Test Gaussien + MDL	Méthodes permettant d'estimer automatiquement le nombre de classes.