Fondements de l'apprentissage machine

Automne 2014

Roland Memisevic

Leçon 6

Plan

- Apprentissage non-supervisé, variables latentes
- K-means clustering
- Modèles de mélange gaussien ("Gaussian mixture models")
- L'algorithme EM

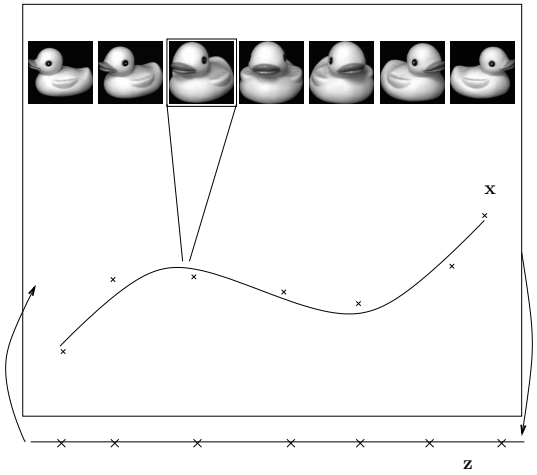
Apprentissage non-supervisé

- Etant donné un ensemble d'observations $\mathbf{x}$, la tâche d'apprentissage non-supervisé est de re-représenter les données, ce qui mène à trouver et comprendre des structures cachées.
- Exemples : **Clustering** - trouvez les groupes, **réduction de la dimensionalité** - trouvez des représentations en basse dimension.
- L'apprentissage non-supervisé peut être formalisé comme l'apprentissage supervisé, où il manque les entrées ou les sorties.
- L'apprentissage non-supervisé suppose que le processus qui a géneré les données contient des paramètres internes qu'on n'a pas observé, mais qui influencent les données tout de même.

Des variables latentes

- Les variables non-observées s'appellent les variables cachées ou les variables latentes.
- Formellement, on postule qu'il y a une variable $\mathbf{z}_n$ associée à chaque exemple d'entraînement $\mathbf{x}_n$.
- Le but de l'apprentissage est d'inférer les valeurs des $\mathbf{z}_n$ en ne sachant que les données $\mathbf{x}_n$.

Des variables latentes

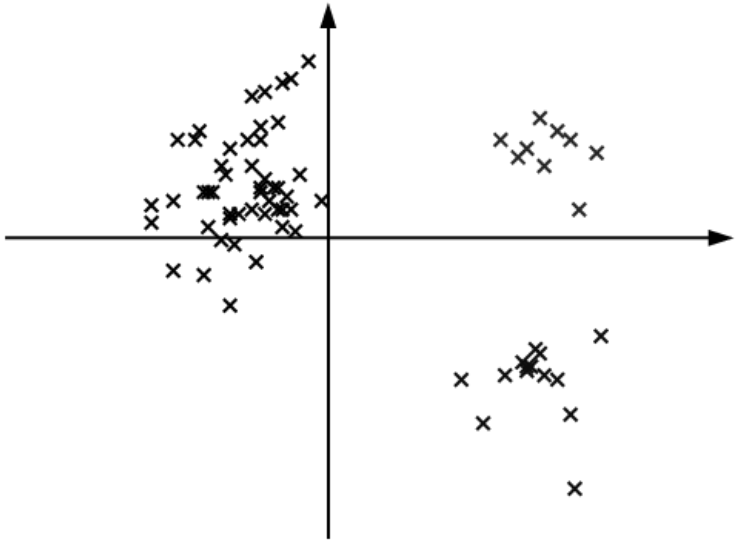


- Apprendre les "causes cachées" qui ont généré les données peut aider à compresser, comprendre ou pré-processer les données.

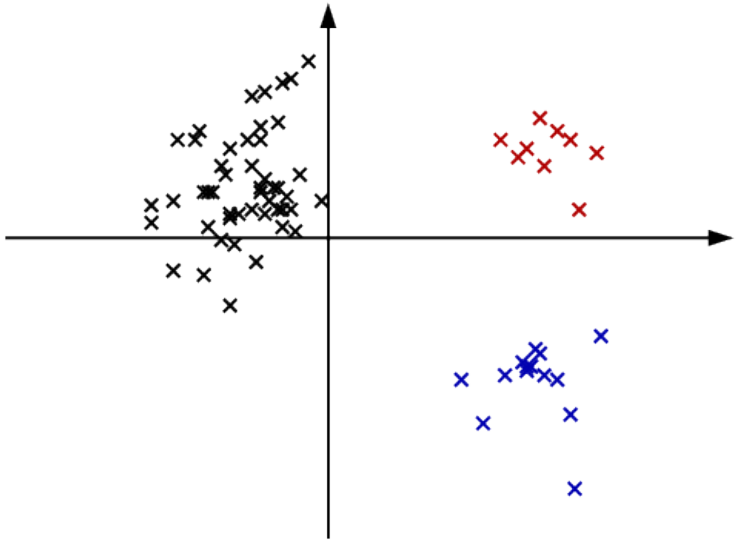
K-means clustering

- Nous considérons les variables latentes discrètes dans cette leçon.
- Pour modéliser des variables latentes discrètes, nous utilisons l'encodage 1-de-K pour $\mathbf{z}_n$.
- L'inférence d'une variable discrète cachée est connue comme clustering parce qu'elle mène à associer chaque point avec un groupe parmi K, où $\mathbf{z}_n$ représente le groupe assigné.
- La méthode la plus connue pour le clustering est K-means.

Exemple



Exemple



K-means clustering

- Notation : Nous empilons des $\mathbf{z_n}$ en codage one-hot dans une matrice $\mathbf{Z}$.
- Nous supposons qu'il existe K *prototypes* $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$ qui représentent les K groupes. La dimensionalité des prototypes est la même que celle des observations $\mathbf{x}$.
- Supposez (pour le moment) que pour chaque point $\mathbf{x_n}$, nous connaissons le groupe $\mathbf{z_n}$ y étant associé.
- La fonction de perte que K-means clustering essaie de minimiser est la distance moyenne entre les points $\mathbf{x}$ et leur représentant $\boldsymbol{\mu}$:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \|\mathbf{x_n} - \boldsymbol{\mu}_k\|^2$$

K-means clustering

- L'apprentissage correspond à trouver les prototypes $\boldsymbol{\mu}_k$ *ainsi que* les associations $\mathbf{z_n}$ qui minimisent J.
- C'est un problème d'optimisation difficile, car les $\mathbf{z_n}$ sont discrets et les $\boldsymbol{\mu}_k$ continus.
- L'apprentissage peut être simplifié si on découple l'optimisation par rapport aux $\mathbf{z_n}$ de l'optimisation par rapport aux $\boldsymbol{\mu}_k$.
- Cela mène à une *descente par blocs de coordonnées* (block coordinate descent), qui est un cas spécial d'une approche d'optimisation dans le domaine d'apprentissage non-supervisé connue sous le nom d'algorithme EM (EM-algorithm).

Trouver les $\mathbf{z_n}$ optimaux

- Notez que sachant les $\boldsymbol{\mu}_k$, nous pouvons optimiser tous les $\mathbf{z_n}$ indépendemment, car l'objectif est la somme sur n.
- L'erreur au carré est minimale si $z_{nk} = 1$ pour *le plus proche* $\boldsymbol{\mu}_k$.
- La solution de l'optimisation est donc :

$$z_{nk} = \begin{cases} 1 & \text{si } k = \arg\min_j \|\mathbf{x_n} - \boldsymbol{\mu}_j\|^2 \\ 0 & \text{sinon.} \end{cases}$$

pour chaque z_{nk}.

Trouver les $\boldsymbol{\mu}_k$ optimaux

- Sachant les $\mathbf{z_n}$, la fonction de perte J est une fonction quadratique de $\boldsymbol{\mu}_k$. Elle peut donc être minimisée en mettant la derivée à zéro :

$$2 \sum_{n=1}^{N} z_{nk}(\mathbf{x_n} - \boldsymbol{\mu}_k) = 0$$

- La solution pour chaque $\boldsymbol{\mu}_k$ est :

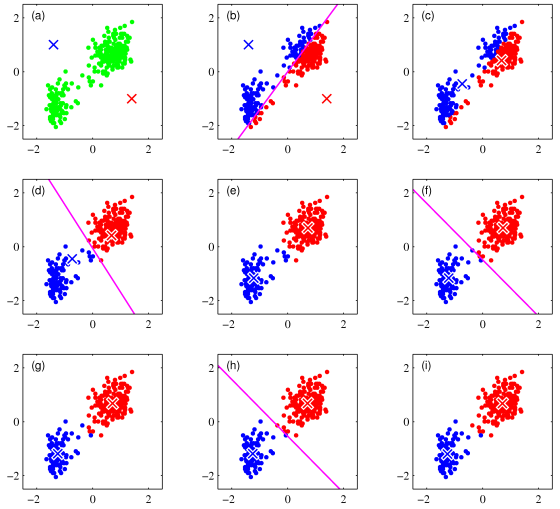
$$\boldsymbol{\mu}_k = \frac{\sum_n z_{nk} \mathbf{x_n}}{\sum_n z_{nk}}$$

- Donc, le $\boldsymbol{\mu}_k$ optimal est la moyenne de tous les points associés au groupe k.

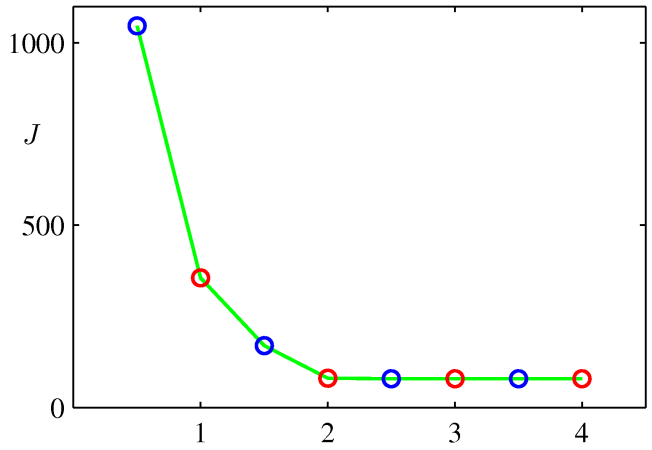
Convergence, minima local

- L'apprentissage consiste à itérer l'inférence des $\mathbf{z_n}$ et d'adapter les paramètres $\boldsymbol{\mu}_k$ jusqu'à convergence.
- Cette procédure est garantie de converger parce que J est positif et à chaque itération, soit J diminue, soit J ne change pas.
- Mais la convergence peut être à un minimum local.
- Une manière d'échapper à de mauvais minima est de faire l'optimisation à plusieurs reprises, en utilisant à chaque fois une initialisation différente des paramètres $\boldsymbol{\mu_k}$.

Exemple



La valeur de J pendant l'apprentissage



Compression

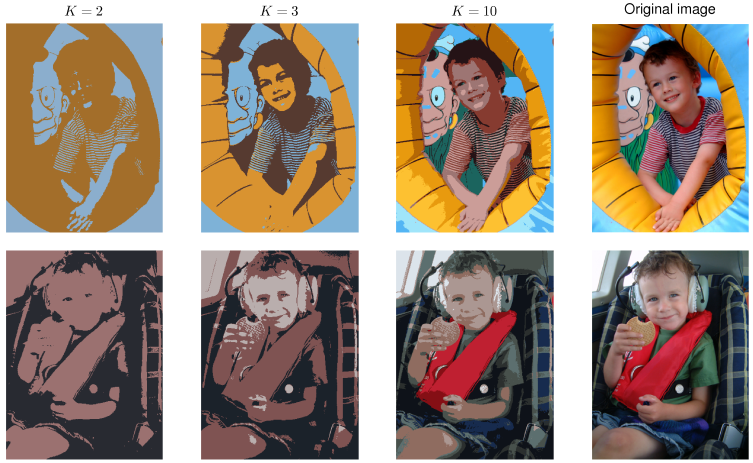
- Étant donné un modèle entraîné, nous pouvons inférer les associations de nouveaux points de test $\mathbf{x}$ aux groupes comme ceci :

$$z_k = \begin{cases} 1 & \text{si } k = \arg\min_j \|\mathbf{x_n} - \boldsymbol{\mu}_j\|^2 \\ 0 & \text{sinon.} \end{cases}$$

- Comme $\mathbf{z}$ représente les vecteurs en haute dimension $\mathbf{x}$ en utilisant un de K entiers, K-means est une manière de faire de la *compression (avec perte)*.
- Dans ce contexte, l'ensemble des K prototypes $\boldsymbol{\mu}_k$ est aussi appellé *codebook*, et K-means *vector quantization*.

Une application simple

- Remplacer le valeur RGB (un vecteur 3D) de chaque pixel par un de K prototypes :



Modèles de mélange gaussien ("Mixture of Gaussians")

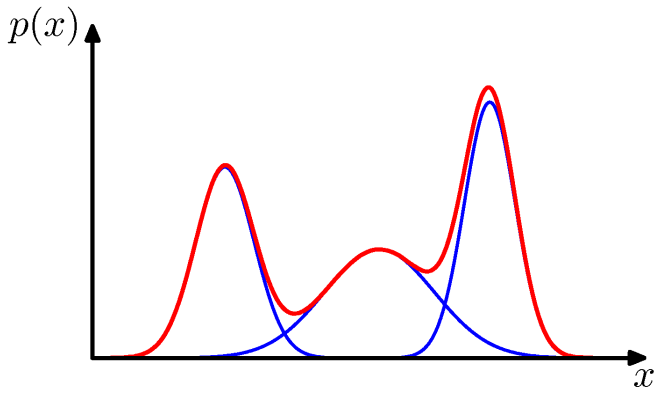
- K-means est lié au

Modèle de mélange gaussien

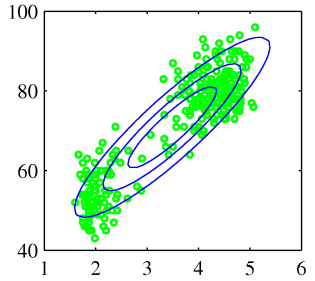
$$p(\mathbf{x}) = \sum_k \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- $\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ sont des paramètres.
- π_k s'appelle *mixing proportion*, et chaque distribution gaussienne s'appelle un *mixture component*.
- Le modèle est simplement une somme ponderée de gaussiennes.
- Il est beaucoup plus puissant qu'une seule gaussienne parce qu'il peut modéliser des distributions *multimodales*.

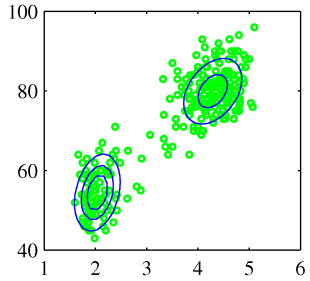
Exemple



Exemple



gaussienne optimale (MV)



MoG pour les mêmes données

Modèles de mélange gaussien

$$p(\mathbf{x}) = \sum_k \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- Pour que $p(\mathbf{x})$ soit une distribution probabiliste, il faut que $\sum_k \pi_k = 1$ est que $\pi_k > 0 \quad \forall k$
- Donc, il faut que les π_k représentent aussi des probabilités.
- Cela suggère d'introduire des variables latentes discrètes $\mathbf{z}$ et de redéfinir le modèle comme suit :

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{z}) p(\mathbf{x}|\mathbf{z})$$

où $p(\mathbf{x}|\mathbf{z})$ est une gaussienne (conditionelle).

Modèles de mélange gaussien

- Cela nous permet de penser au modèle comme le processus génératif suivant pour générer des observations : Nous tirons un composant d'une distribution discrète, puis nous tirons une observation d'une gaussienne dont les paramètres dépendent de ce premier choix.
- Pour inférer le composant sachant l'observation, $\mathbf{x_n}$, nous pouvons utiliser la règle de Bayes :

$$p(\mathbf{z_n}|\mathbf{x_n}) = \frac{p(\mathbf{x_n}|\mathbf{z_n})p(\mathbf{z_n})}{\sum_{\mathbf{z_n}} p(\mathbf{x_n}|\mathbf{z_n})p(\mathbf{z_n})}$$

ce qui répresente la vraisemblance pour le point d'être tiré de chaque gaussienne.
- $p(z_{nk} = 1|\mathbf{x_n})$ s'appelle *responsabilité* du composant k, que nous abbrévions $\gamma(z_{kn})$.
- C'est comme un K-means "doux".

L'algorithme EM

- On pourrait utiliser la descente de gradient pour l'optimisation, mais il existe une procédure plus courante semblable à celle utilisée pour K-means, l'algorithme EM.
- Pour un ensemble d'entraînement, $\{\mathbf{x_n}\}$, nous avons :

$$\begin{aligned} L := \sum_n \log p(\mathbf{x_n}) &= \sum_n \log \sum_{\mathbf{z_n}} p(\mathbf{x_n}|\mathbf{z_n})p(\mathbf{z_n}) \\ &= \sum_n \log \sum_{\mathbf{z_n}} q(\mathbf{z_n}) \frac{p(\mathbf{x_n}|\mathbf{z_n})p(\mathbf{z_n})}{q(\mathbf{z_n})} \\ &\geq \sum_n \sum_{\mathbf{z_n}} q(\mathbf{z_n}) \log \frac{p(\mathbf{x_n}|\mathbf{z_n})p(\mathbf{z_n})}{q(\mathbf{z_n})} \\ &:= \mathcal{L} \end{aligned}$$

où nous avons utilisé l'**inégalité de Jensen** :
$\log \sum_i a_i b_i \geq \sum_i a_i \log b_i \quad$ if $\forall i : a_i > 0$ et $\sum_i a_i = 1$

L'algorithme EM

- À la place d'optimiser L, nous allons optimiser une borne inférieure, $\mathcal{L}$, par rapport aux paramètres *ainsi qu'* aux variables auxiliaires $q(\mathbf{z})$.
- Nous pouvons écrire :

$$\mathcal{L} = \sum_{nk} q_{nk} \log p(\mathbf{x_n}|\mathbf{z_n} = k)p(\mathbf{z_n} = k) - \sum_{nk} q_{nk} \log q_{nk}$$

où $q_{nk} := q(\mathbf{z_n} = k)$
- Notez que le premier terme de $\mathcal{L}$ est une espérance de $p(\mathbf{x_n}, \mathbf{z_n})$ par rapport à $q(\mathbf{z_n})$. Il est nommé "expected complete log-likelihood" en anglais.
- C'est le seul terme qui dépend des paramètres du modèle.

Optimisation par rapport aux paramètres

- Pour optimiser la borne par rapport aux paramètres, nous mettons la dérivée à zéro :

$$\frac{\partial \mathcal{L}}{\partial \mu_k} = \sum_n q_{nk} \boldsymbol{\Sigma}_k \left(\mathbf{x_n} - \boldsymbol{\mu}_k \right) = 0$$

$$\Leftrightarrow \boldsymbol{\mu}_k = \frac{\sum_n q_{nk} \mathbf{x_n}}{\sum_n q_{nk}}$$

- Similairement on peut trouver :

$$\boldsymbol{\Sigma}_k = \frac{\sum_n q_{nk} (\mathbf{x_n} - \boldsymbol{\mu}_k)(\mathbf{x_n} - \boldsymbol{\mu}_k)^{\mathrm{T}}}{\sum_n q_{nk}}$$

et

$$\pi_k = p(\mathbf{z}_k) = \frac{\sum_n q_{nk}}{N}$$

Optimisation par rapport aux variables auxiliaires

- Pour optimiser $\mathcal{L}$ par rapport aux variables auxiliaires, nous récrivons $\mathcal{L}$ comme suit :

$$\begin{aligned}
\mathcal{L} &= \sum_n \sum_{\mathbf{z_n}} q(\mathbf{z_n}) \log \frac{p(\mathbf{x_n}|\mathbf{z_n}) p(\mathbf{z_n})}{q(\mathbf{z_n})} \\
&= \sum_n \sum_{\mathbf{z_n}} q(\mathbf{z_n}) \log \frac{p(\mathbf{z_n}|\mathbf{x_n}) p(\mathbf{x_n})}{q(\mathbf{z_n})} \\
&= \sum_n \sum_{\mathbf{z_n}} q(\mathbf{z_n}) \log \frac{p(\mathbf{z_n}|\mathbf{x_n})}{q(\mathbf{z_n})} + \sum_n \sum_{\mathbf{z_n}} q(\mathbf{z_n}) \log p(\mathbf{x_n}) \\
&= -\sum_n \mathrm{KL}\left(q(\mathbf{z_n}) \,||\, p(\mathbf{z_n}|\mathbf{x_n}) \right) + L
\end{aligned}$$

L'algorithme EM

- La

divergence Kullback-Leibler (divergence KL)

$$\mathrm{KL}\left(p_1(\mathbf{z}) \,||\, p_2(\mathbf{z}) \right) = \sum_{\mathbf{z}} p_1(\mathbf{z}) \log \frac{p_1(\mathbf{z})}{p_2(\mathbf{z})}$$

mesure la similarité entre deux distributions probabilistes p_1, p_2.

- La divergence KL est toujours non-négative et est zéro seulement si $p_1 = p_2$!
- Pour cette raison, $\mathcal{L} = L$ si nous mettons $q(\mathbf{z_n}) = p(\mathbf{z_n}|\mathbf{x_n})$!
- En d'autres termes, le choix optimal pour $q(\mathbf{z_n})$ est $q(\mathbf{z_n}) = p(\mathbf{z_n}|\mathbf{x_n})$, ce qui rendra la borne $\mathcal{L}$ pour L exacte !

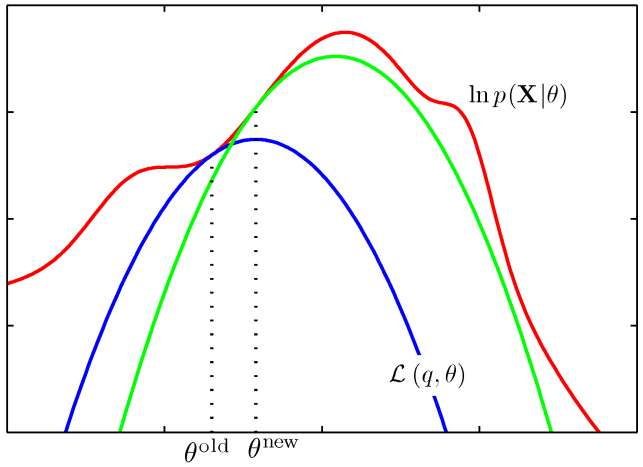
L'algorithme EM

L'algorithme EM

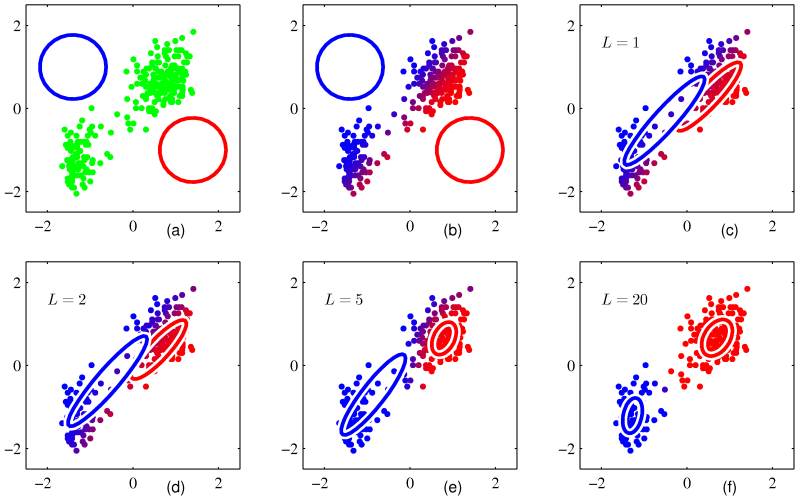
1. E-step : Évaluez les distributions postérieures $p(\mathbf{z_n}|\mathbf{x_n})$
2. M-step : Optimisez $\mathcal{L}$ par rapport aux paramètres, gardant $q(\mathbf{z_n}) = p(\mathbf{z_n}|\mathbf{x_n})$.

- Le E-step calcule la "expected complete log-likelihood". Il correspond à évaluer les responsabilités $\gamma(z_{nk})$ pour chaque point.
- Le M-step maximise la "expected complete log-likelihood". Dans un modèle de mélange gaussien, cela correspond à mettre les paramètres à des sommes ponderées par les responsabilités.

EM comme optimisation d'une séquence de bornes



Exemple



Des commentaires

- L'algorithme EM peut être appliqué à beaucoup de modèles à variables latentes (pas seulement les mélanges de gaussiennes).
- Le E-step et le M-step doivent être dérivés individuellement pour chaque modèle, mais la perspective de la borne inférieure $\mathcal{L}$ de la log-vraisemblance est toujours la même.
- Un des premiers modèles optimisés par l'algorithme EM était le Hidden Markov Model (HMM).
- Il existe des modèles pour lequels calculer $p(\mathbf{z}|\mathbf{x})$ n'est pas traitable ("intractable"). Dans ces cas, on peut utiliser une variation de EM où on améliore la KL-divergence dans le E-step à la place de trouver la distribution a posteriori exacte.

Modèle de mélange gaussien et K-means

- Le modèle de mélange gaussien est comme K-means, où nous utilisons une association "douce" aux groupes.
- On peut dériver K-means formellement comme limite d'un modèle de mélange gaussien où les variances vont à zéro.