Cours IFT6266,

Maximum de vraisemblance et algorithme EM

Entropie croisée et log-vraisemblance:

La neg-log-vraisemblance avec modèle de densité f est

$$-\frac{1}{n}\sum_i \log f(x_i) = \hat{E}[-\log f(X)] = -\int \hat{p}(x)\log f(x)dx$$

l'entropie croisée de la distribitution empirique $\hat{p}$ et le modèle f. Son espérance (sur D) est

$$-\int p(x)\log f(x)dx$$

l'entropie croisée entre la vraie loi p et le modèle f, qui est minimisée quand $f = p$, ce qui donne l'entropie de p. Note: $E_{x_i}[\delta(x-x_i)] = p(x_i)$. Si on soustrait l'entropie de p on obtient la divergence de Kullback Liebler entre p et f, $KL(p||f)$.

Le défaut du critère de maximum de vraisemblance, c'est qu'il peut donner des résultats pour lesquels la distribution est très pointue autour des exemples et zero ailleurs. En fait, si il n'y avait aucune contrainte sur la famille de fonctions P_θ, on obtiendrait exactement δ/l aux points des exemples et 0 ailleurs (δ est une fonction de masse 1 concentrée en un seul point). Cela signifie que le critère est aussi petit qu'il est possible ($-\infty$) pour les exemples d'apprentissage tout en étant aussi grand qu'il est possible (∞) en tout point qui n'est pas dans l'ensemble d'apprentissage. Des critères alternatifs ont été proposés (voir: *penalized maximum likelihood* et méthodes Bayesiennes): en général le principe est le même qu'avec les algorithmes d'apprentissage, i.e., on pénalise les solutions moins lisses (ou plus "complexes" selon un certain critère, telle la probabilité a priori).

Modèles de mélange

$$p(x) = \sum_{j=1}^{m} p(x|j)P(j)$$

on a une densité $p(x|j)$ pour chaque composante j, donc si $p(x|j)$ normalisé $p(x)$ aussi. Pour un x donné on peut calculer la proba a posteriori $P(j|x) = \frac{p(x|j)P(j)}{p(x)}$.

Si on permet de varier m on obtient qqchose de fondamentalement non-paramétrique (même si pour chaque m fixe on a un modèle paramétrique). On verra la même situation avec les réseaux de neurones.

Mélange de Gaussiennes: $p(x|j)$ est une normale de moyenne μ_j et variance Σ_j. C'est un "approximateur universel" des densités (pour une erreur fixée ϵ on peut trouver $m < \infty$ et les μ_j, Σ_j qui approxime avec une erreur plus petite).

Paramètres: les $P(j)$ et les paramètres de chaque composante.

On peut aussi avoir des mélanges conditionnels:

$$P(Y|X) = \sum_i P(i|X)P(Y|X,i)$$

qu'on appelle aussi des **mélanges d'experts** (*mixtures of experts*). Dans ce cas, chacun des sous-modèles représente une distribution conditionnelle (par exemple on peut utiliser un réseau de neurone pour calculer les paramètres d'une Gaussienne). Rappelons que

$$E[Y|X] = \sum_i P(i|X)E[Y|X,i]$$

Donc si on a un ensemble de modèles de régression, $E[Y|X,i]$, on peut les combiner ainsi. La fonction $P(i|X)$ est aussi une fonction paramétrisée de X, et on l'appelle le **gater** puisqu'elle décide quelle sous-distribution utiliser dans chaque contexte X.

Algorithme EM

Malheureusement on ne peut maximiser analytiquement la vraisemblance pour un modèle de mélange de densités simples (e.g. mélange de Gaussiennes). L'algorithme EM est un algorithme d'optimisation pour modèles probabilistes quand on peut résoudre analytiquement une fois qu'une certaine variable aléatoire est introduite et supposée observée (ici ça sera l'identité de la composante j qui a généré la donnée x).

L'algorithme **EM** est un algorithme d'optimisation (d'estimation de paramètres) pour certaines distributions paramétriques $P_\theta(Y)$ (possiblement conditionnelles $P_\theta(Y|X)$), et qui a été principalement utilisé pour des mixtures, telles que les mixtures de Gaussiennes et les modèles de Markov cachés (HMMs). Il s'agit d'une technique d'optimisation qui permet parfois d'accélerer l'estimation des paramètres (par rapport à un algorithme d'optimisation numérique générique). Il faut pouvoir introduire un nouveau modèle probabiliste $P_\theta(Y,Z)$ avec une *variable cachée* (non-observée) Z (généralement discrète), telle que la maximisation de la vraisemblance de $P_\theta(Y,Z)$ est beaucoup plus facile que celle de $P_\theta(Y)$. C'est par exemple le cas des mélanges de Gaussiennes et des HMMs. Pour les mélanges de Gaussiennes, la

variable cachée est l'identité de la Gaussienne qui est associée à l'exemple Y. Si on connaissait la valeur de cette variable cachée (pour chaque exemple), l'estimation des paramètres deviendrait triviale (c'est comme si on avait plusieurs problèmes indépendents d'estimation des paramètres de plusieurs Gaussiennes). Comme Z n'est pas observée, l'algorithme procède ainsi, de manière itérative:

1. Phase E (estimation):

$$Q(\theta, \theta_{t-1}) = E_Z[\sum_i \log(P_\theta(y_i, Z))|\theta_{t-1}, D]$$

   (où $D = \{y_1 \dots y_n\}$ et la distribution de Z est conditionnée sur la connaissance de D, en utilisant les paramètres θ_{t-1}).

2. Phase M (maximisation):

$$\theta_t \leftarrow \text{argmax}_\theta Q(\theta, \theta_{t-1})$$

La phase M peut se faire analytiquement quand on peut solutionner l'équation $\frac{\partial Q(\theta,\theta_{t-1})}{\partial \theta} = 0$. On peut montrer que cette algorithme converge vers un maximum (possiblement local) ou un point selle (improbable).

D'où vient la fonction auxiliaire Q?

On va utiliser Q pour borner la vraisemblance et on va ensuite optimiser θ par rapport à cette borne. Soit $L(\theta)$ la log-vraisemblance obtenue avec les paramètres θ. Donc

$$\begin{aligned}
L(\theta) - L(\theta_{t-1}) &= \sum_i \log \left\{ \frac{\sum_j P_\theta(Z = j, Y = y_i)}{P_{\theta_{t-1}}(Y = y_i)} \right\} \\
&= \sum_i \log \left\{ \frac{\sum_j P_\theta(Z = j, Y = y_i)}{P_{\theta_{t-1}}(Y = y_i)} \frac{P_{\theta_{t-1}}(Z = j|Y = y_i)}{P_{\theta_{t-1}}(Z = j|Y = y_i)} \right\} \\
&\geq \sum_i \sum_j P_{\theta_{t-1}}(Z = j|Y = y_i) \log \left\{ \frac{P_\theta(Z = j, Y = y_i)}{P_{\theta_{t-1}}(Y = y_i) P_{\theta_{t-1}}(Z = j|Y = y_i)} \right\} \\
&= \sum_i E_Z[\log(P_\theta(Z, y_i))|\theta_{t-1}, D] + \textit{constante en } \theta
\end{aligned}$$

où l'on a utilisé l'inégalité de Jensen pour le log ($\log(E[X]) \geq E[\log X]$).

EM pour mélange de Gaussiennes

Soit un mélange de Gaussiennes

$$P_\theta(Y) = \sum_{i=1}^{K} w_i P_{\theta_i}(Y|i)$$

où $P_{\theta_i}(Y|i)$ est une distribution Gaussienne avec des paramètres μ_i et Σ_i, w_i est le poids de la Gaussiene i, qui peut être interprété comme la probabilité a priori de la Gaussienne i, $w_i = P(Z = i)$. Ceci correspond en fait à introduire une autre variable aléatoire, Z, cachée, qui identifie la Gaussienne associée à un exemple. On a donc une distribution jointe

$$P_\theta(Y = y, Z = i) = P_\theta(Z = i)P_\theta(Y = y|Z = i)$$

Si on applique l'algorithme EM ci-haut, on obtient les formules de réestimation

$$\tilde{\mu_i} \leftarrow \frac{\sum_t P_\theta(Z = i|Y = y_t)y_t}{\sum_t P_\theta(Z = i|Y = y_t)}$$

et

$$\tilde{\Sigma_i} \leftarrow \frac{\sum_t P_\theta(Z = i|Y = y_t)y_t y_t'}{\sum_t P_\theta(Z = i|Y = y_t)} - \tilde{\mu_i}\tilde{\mu_i}'$$

et

$$\tilde{w_i} \leftarrow \frac{\sum_t P_\theta(Z = i|Y = y_t)}{\sum_t 1}$$

où

$$P_\theta(Z = i|Y = y_t) = \frac{P_\theta(Y = y_t|Z = i)w_i}{\sum_j P_\theta(Y = y_t|Z = j)w_j}$$

est appelé le *postérieur* de la Gaussienne i (pour l'exemple t).