Département d'informatique et de recherche opérationnelle Automne 2006

IFT 6266
Algorithmes d'apprentissage

Yoshua Bengio

Bureau : PAA #3339, courriel: pift6266@iro

Devoir #3

Donné le 5 septembre 2006, Dû le 19 octobre 2006

1. Considérez un problème de régression avec plusieurs variables à prédire (donc Y est un vecteur) dans lequel on suppose que la loi conditionnelle de Y, étant donné l'intrant X (aussi un vecteur), (i.e. $Y|X$) implique des corrélations entre les Y_i. On veut modéliser ces dépendances en supposant que $Y|X = x$ suit une normale d'espérance $f(x)$ et de matrice de covariance Σ. On va entraîner notre modèle selon le critère de maximum de vraisemblance conditionnelle avec des données $D = \{(x_t, y_t)\}_{t=1}^n$, c'est à dire la minimisation de $-\sum_t \log p(y_t|x_t)$.

   (a) Si on applique le principe de maximum de vraisemblance à ce modèle de loi conditionnelle, on obtient un critère d'apprentissage qui est différent de celui de la régression ordinaire. Dérivez une formule pour ce critère, en supposant d'abord Σ fixe et donné.

   (b) Supposez ensuite que Σ est un paramètre. Proposez une manière pour estimer conjointement Σ et les paramètres de $f(x)$.

   (c) (**BONUS**) Supposez finalement que la corrélation entre les Y_i peut varier selon X. On va donc avoir Σ non plus comme un paramètre libre mais comme une fonction paramétrisée de x, $\Sigma(x)$. Comment pourrait-on utiliser un réseau de neurones pour prédire non seulement l'espérance $f(x)$ mais aussi la covariance $\Sigma(x)$? La réponse à cette question tourne essentiellement autour d'une manière de convertir une série de nombres (certaines sorties d'un réseau de neurones) qui sont sans contraintes (des réels quelquonques) en une série de nombres qui forment une matrice de covariance (qui a donc la contrainte d'être positive définie).

   N.B. En **finance** et en particulier en *gestion du risque* et en *gestion de portefeuille* on s'intéresse beaucoup à prédire non seulement l'espérance de rendement de plusieurs actifs (car on voudrait investir plus sur ceux qui sont plus rentables) mais aussi à prédire la covariance future entre les rendements de ces actifs (car on voudrait réduire nos risques en investissant plus dans des actifs peu corrélés entre eux). Dans ce cas X représente l'information disponible au présent pour prendre certaines décisions de placement ou bien évaluer le risque d'un portefeuille, et Y représente le rendement futur des actifs de notre portefeuille. Prédire $P(Y|X)$ en tenant compte des dépendences entre les Y_i est donc très important.

2. Considérez un problème de classification dans lequel on sait que les observations $Y \in \{0, 1\}$ peuvent nous être fournies avec une erreur, et que la probabilité d'erreur est ϵ. Avec l'hypothèse i.i.d. habituelle, comment est-ce que le critère de maximum de vraisemblance pour la classification probabiliste est transformé par cette supposition que les étiquettes sont bruitées? Le cas $\epsilon = 0$ devrait redonner la formule habituelle, qui est le coût $-\sum_t \log f_{y_t}(x_t)$ quand $f_i(x)$ représente un estimateur de $P(Y = i|X = x)$.

3. Montrez le détail du calcul de la dérivée de la softmax par rapport à son argument. Appliquez cela pour obtenir le gradient de la log-vraisemblance négative pour la classification probabiliste $Q = -\log f_y(x)$ par rapport à l'argument a de la softmax, quand $f(x) = \mathrm{softmax}(a)$. i.e. calculez $\frac{\partial Q}{\partial a_i}$.

4. On sait que l'erreur d'apprentissage d'un algorithme d'apprentissage (après minimisation de cette erreur) est généralement biaisée (trop optimiste, donc plus petite que l'erreur de généralisation). Si on utilise un ensemble de validation pour optimiser la valeur d'un hyper-paramètre, l'erreur de validation, ayant servi à "entrainer" notre hyper-paramètre, souffrira aussi d'un "optimisme": on observe que la valeur minimale de l'erreur de validation (correspondant à la valeur choisie "optimale" pour l'hyper-paramètre) est un estimateur biaisé (aussi optimiste) de l'erreur de généralisation. Une fois que l'on a sélectionné un hyper-paramètre, on a donc plus d'estimateur non-biaisé de l'erreur de généralisation. On peut cependant obtenir un estimateur non-biaisé en utilisant un TROISIÈME ensemble de données, que l'on appelle parfois l'ensemble de test. Vous allez vérifier ce phénomène empiriquement en réutilisant les données et votre code pour la dernière question du devoir 2. Vous allez diviser l'ensemble de validation V en deux parties. La première, avec 10 éléments de V, servira à choisir σ, donc comme ensemble de validation. Les autres 90 exemples serviront comme ensemble de test, pour estimer l'erreur de généralisation. Comme 10 éléments sont très peu, les résultats varieront beaucoup selon quels 10 éléments de V ont été utilisés comme ensemble de validation. Vous allez donc répéter l'expérience avec 50 partitions aléatoires de V (en un groupe de 10 et un groupe de 90). La quantité que nous voulons calculer est la différence entre l'erreur de validation minimale (celle choisie après optimisation de σ) et l'erreur de test (avec ce même σ). Pour chacune des 50 partitions vous allez donc optimiser σ. Pour ne pas faire cette optimisation "à la main" (ce qui serait très long), nous vous demandons de sélectionner une valeur de σ qui optimise l'erreur de validation parmi la liste de valeurs suivantes: $(0.005, 0.015, 0.020, 0.025, 0.030, 0.035, 0.1)$. Pour vérifier que la différence est statistiquement significative, calculez la moyenne et l'erreur-type de ces 50 différences. L'erreur-type au carré est la variance de la moyenne, qui est obtenue en divisant par 50 la variance empirique de ces 50 différences. Si la différence moyenne est plus grande que deux erreurs-type, on peu considérer cette différence comme statistiquement significative. Est-ce le cas dans votre expérience?