Exercises : Gradient stochastique

Optimisation pour l'apprentissage automatique

M2 IASD Apprentissage, 2024-2025

Dauphine | PSL
UNIVERSITÉ PARIS

Exercice 1: Random reshuffling

On considère un problème de minimisation du risque empirique de la forme

$$\text{minimiser}_{\boldsymbol{w}\in\mathbb{R}^d} f(\boldsymbol{w}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{w}), \tag{1}$$

où chaque f_i est de classe $\mathcal{C}^1$ et dépend uniquement du $i$ème point d'un jeu de données à n éléments. On suppose aussi que la fonction objectif f est convexe, et $\mathcal{C}^{1,1}$. On supposera enfin que le problème (1) possède une solution, et on notera $f^* = \min_{\boldsymbol{w}\in\mathbb{R}^d} f(\boldsymbol{w})$.

On supposera que n est trop large pour que le jeu de données puisse être utilisé en entier lors d'une itération d'un algorithme, et on considère donc l'algorithme du gradient stochastique

$$x_{k+1} = x_k - \alpha_k \nabla f_{i_k}(x_k), \tag{2}$$

avec $\alpha_k > 0$ et $i_k \in \{1,\dots,n\}$. Le but de l'exercice est d'étudier les variantes basées sur le principe de *random reshuffling* (dont le principe peut être traduit par *rebattre les cartes*). Dans ces variantes, les indices $\{i_k\}$ sont tirés selon une perturbation aléatoire de $\{1,\dots,n\}$ qui est modifiée toutes les n itérations. Ainsi, à l'itération 0, une permutation aléatoire de $\{1,\dots,n\}$ définit $\{i_0,\dots,i_{n-1}\}$, puis à l'itération n, une autre permutation aléatoire définit $\{i_n,\dots,i_{2n-1}\}$, et ainsi de suite.

a) Rappeler la définition d'une époque (*epoch*). Avec la stratégie du *random reshuffling*, que peut-on garantir sur les points du jeu de données qui ont été tirés au cours de la première époque ?

b) Soit un indice d'itération k correspondant à la première itération d'une époque (càd $k = \ell n$ avec $\ell \in \mathbb{N}$).

   i) Montrer que
   $$\mathbb{E}_{i_k}\left[\nabla f_{i_k}(x_k)\right] = \nabla f(x_k).$$

   ii) Les indices $i_k,\dots,i_{k+n-1}$ n'étant pas indépendants, justifier que la propriété de la questions b)i) n'est pas vérifiée pour les autres itérations de l'époque.

c) Même sans l'hypothèse de la question b)i), on peut montrer des résultats de convergence pour un bon choix de taille de pas. Pour cela, on regarde la suite des itérés moyens $\{\bar{x}_K\}_K$, avec

$$\bar{x}_K = \frac{1}{K+1}\sum_{k=0}^{K} x_k \quad \forall K \in \mathbb{N}.$$

On peut ainsi montrer qu'après nK itérations, on a

$$\mathbb{E}\left[f\left(\bar{x}_{nK}\right)\right] - f^* \leq \mathcal{O}\left(\frac{1}{\sqrt{nK}}\right),$$

Comparer ce taux avec celui de la descente de gradient pour le même problème.

d) On considère une variantes par fournées du gradient stochastique, pour laquelle les indices sont tirés suivant une approche par *random reshuffling*. Si la taille de fournées est de n, quel algorithme retrouve-t-on ?

e) Comme l'approche par *random reshuffling* ne tire pas le même indice au sein d'une même époque, on peut vouloir utiliser de l'information des itérations précédentes pour améliorer le pas. Quelle technique vue lors des séances proposeriez-vous d'utiliser pour cela ?

Exercice 2 : Perte de Huber

On considère un jeu de données $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, où $n \geq 1$, $\boldsymbol{x}_i \in \mathbb{R}^d$ avec $d \geq 1$ et $y_i \in \mathbb{R}$. On cherche un modèle linéaire qui prédise au mieux chaque y_i à partir du $\boldsymbol{x}_i$ correspondant. On définit donc une famille de modèles paramétrée par $\boldsymbol{w} \in \mathbb{R}^d$ comme suit :

$$\begin{array}{rccl} h_{\boldsymbol{w}}: & \mathbb{R}^d & \to & \mathbb{R} \\ & \boldsymbol{x} & \mapsto & \boldsymbol{x}^{\mathrm{T}}\boldsymbol{w} = \sum_{i=1}^d [\boldsymbol{x}]_i [\boldsymbol{w}]_i. \end{array}$$

Pour un modèle $h_{\boldsymbol{w}}$, on considèrera que ce modèle prédit parfaitement y_i à partir de $\boldsymbol{x}_i$ si on a $\ell\left(h_{\boldsymbol{w}}(\boldsymbol{x}_i) - y_i\right) = \ell\left(\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w} - y_i\right) = 0$, où $\ell : \mathbb{R} \to \mathbb{R}$ est la fonction de perte de Huber définie par :

$$\ell(t) = \begin{cases} \frac{1}{2}t^2 & \text{if } |t| < 1 \\ |t| - \frac{1}{2} & \text{otherwise.} \end{cases} \tag{3}$$

Cette fonction se comporte comme $t \mapsto \frac{t^2}{2}$ pour $|t| < 1$ et comme $t \mapsto |t|$ lorsque $|t|$ est très grand. Contrairement à ce que son expression peut suggérer, ℓ est continûment dérivable (ou de classe $\mathcal{C}^1$)

L'expression $\ell\left(h_{\boldsymbol{w}}(\boldsymbol{x}_i) - y_i\right)$ représente l'erreur du modèle en $(\boldsymbol{x}_i, y_i)$, et on cherche un modèle (c'est-à-dire un vecteur $\boldsymbol{w} \in \mathbb{R}^d$) tel que la somme de ces erreurs soit minimale. On considère donc :

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} f(\boldsymbol{w}) := \frac{1}{n}\sum_{i=1}^{n} \ell(\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w} - y_i). \tag{4}$$

a) Justifier que 0 est un minorant de (4). Est-ce sa valeur minimale ?

b) Le gradient de f en $\boldsymbol{w} \in \mathbb{R}^d$ est donné par

$$\nabla f(\boldsymbol{w}) = \frac{1}{n} \sum_{i=1}^{n} \ell'(\boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w} - y_i) \boldsymbol{x}_i, \tag{5}$$

avec

$$\ell'(t) = \begin{cases} 1 & \text{si} \quad t > 1 \\ t & \text{si} \quad |t| \leq 1 \\ -1 & \text{si} \quad t < -1. \end{cases}$$

Écrire (en pseudo-code) l'itération de descente de gradient avec une taille de pas constante α et en utilisant la formule (5). Que devient cette itération si le point courant est un minimum local ?

c) Une constante de Lipschitz pour ∇f est $L = \frac{1}{n} \sum_{i=1}^{n} \|\boldsymbol{x}_i\|^2$. Comment utiliser cette constante pour définir la longueur de pas ? Lorsque L est inconnue, donner deux choix possibles pour la taille de pas.

d) La fonction f s'écrit $f = \frac{1}{n} \sum_{i=1}^{n} f_i$, où $f_i(\boldsymbol{w}) = \ell(\boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w} - y_i)$. Le gradient f_i en $\boldsymbol{w}$ est

$$\nabla f_i(\boldsymbol{w}) = \ell'(\boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w} - y_i) \boldsymbol{x}_i.$$

Écrire (en pseudo-code) l'itération du gradient stochastique pour ce problème sans choix particulier de taille de pas.

e) *On considère ici que notre unité de coût est un accès à un $\boldsymbol{x}_i$.* Quel est le coût d'une itération de descente de gradient, et celui d'une itération de gradient stochastique ?

f) Quand on applique le gradient stochastique avec une longueur de pas fixe, on peut parfois observer que la méthode génère des itérés de norme de plus en plus grande, ce qui conduit à un dépassement de mémoire pour l'algorithme. Fournir une justification à ce phénomène.

g) On considère une variante par fournées (*batch*) du gradient stochastique, dans laquelle on choisit un sous-ensemble de n_b composantes dans la somme finie de (4).

i) Écrire l'itération correspondante (en pseudo-code).

ii) Si n_b correspond au nombre de processeurs disponibles pour les calculs, quel peut être l'intérêt de choisir n_b comme taille de fournée ?

iii) Donner un autre intérêt plus général des méthodes par fournées en comparaison avec l'algorithme du gradient stochastique basique.

iv) Supposons que l'on utilise plusieurs tailles de fournées et que l'on observe une amélioration en termes de convergence quand n_b augmente pour $1 \leq n_b \leq \frac{n}{10}$. Supposons que l'on observe aussi qu'augmenter n_b au-delà de $n/10$ conduise à une dégradation de la performance. Comment expliquer ces observations ?

Correction

Correction de l'exercice 1: Random reshuffling

a) Une époque est une unité de coût correspond à n accès à un exemple du jeu de données dans un jeu de données à n éléments. Une époque de gradient avec *random reshuffling* correspond à une passe dans le jeu de données.

b) $k = \ell n$ avec $\ell \in \mathbb{N}$ (première itération d'une époque).

i) Puisque i_k est le premier indice d'une permutation aléatoire, il suit une distribution uniforme dans $\{1, \dots, n\}$, indépendamment de $\boldsymbol{w}_k$. Par conséquent, on a

$$\mathbb{E}_{i_k}\left[\nabla f_{i_k}(\boldsymbol{w}_k)\right] = \sum_{i=1}^{n} \mathbb{P}\left(i_k = i\right) \times \nabla f_i(\boldsymbol{w}_k) = \sum_{i=1}^{n} \frac{1}{n} \times \nabla f_i(\boldsymbol{w}_k) = \nabla f(\boldsymbol{w}_k).$$

ii) Soit l'itération $k+1$. Comme la valeur de i_{k+1} dépend de celle de i_k, on ne peut plus appliquer le raisonnement de la question précédente. Le raisonnement s'applique également aux indices suivants.

c) Après nK itérations de descente de gradient, on obtiendrait un itéré $\boldsymbol{w}_{nK}^G$ tel que

$$f(\boldsymbol{w}_{nK}^G) - f^* \leq \mathcal{O}\left(\frac{1}{nK}\right),$$

Il s'agit d'une meilleure vitesse que l'approche par *random reshuffling*, à la fois parce que la vitesse de convergence est plus rapide quand $K \to \infty$ et parce que la garantie est déterministe.

Dans le même temps, une itération de descente de gradient est n fois plus coûteuse qu'une itération de gradient stochastique en termes d'accès aux données. Pour un nombre fixe d'époques N_E, la méthode de gradient stochastique effectue nN_E itérations, tandis que celle de la descente de gradient n'en effectue que N_E itérations. Lorsque n est suffisamment grand, on a $\sqrt{nN_E} \gg N_E$, et donc la vitesse de convergence sera meilleure pour le gradient stochastique.

d) Avec une taille de fournée de n, on obtient une itération de descente de gradient car les indices seront tirés sans remise.

e) Des techniques de gradient stochastique basées sur du momentum, comme le `gradient stochastique avec momentum` et `Adam` utilisent les informations des itérations précédentes, et seraient donc appropriées.

Correction de l'exercice 2: Perte de Huber

a) La fonction ℓ est positive sur $\mathbb{R}$. Pour tout $\boldsymbol{w} \in \mathbb{R}^d$, on a donc

$$f(\boldsymbol{w}) = \frac{1}{n}\sum_{i=1}^{n} \ell(\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w} - y_i) \geq \frac{1}{n}\sum_{i=1}^{n} 0 = 0.$$

Cela montre que 0 est un minorant du problème (4). Cette valeur est atteinte uniquement lorsqu'il existe un point $\boldsymbol{w}$ tel que $\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w} - y_i = 0$ pour tout i. Cela n'est pas toujours le cas (prendre par exemple $n = 2, d = 1, \boldsymbol{x}_1 = 1, \boldsymbol{x}_2 = -1, y_1 = y_2 = 1$), donc 0 n'est pas nécessairement la valeur minimale du problème.

b) En un point $\boldsymbol{w}_k \in \mathbb{R}^d$, l'itération de la méthode de descent de gradient avec un pas constant α s'écrit: :

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \frac{\alpha}{n}\sum_{i=1}^{n} \ell'(\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w}_k - y_i)\boldsymbol{x}_i.$$

Si $\boldsymbol{w}_k$ est un minimum local, on a $\nabla f(\boldsymbol{w}_k) = 0$, et l'itération devient $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k$.

c) Si on connaît une constante de Lipschitz L pour le gradient, on peut alors choisir un pas constant égal à $\alpha = \frac{1}{L}$.
Si on ne connaît pas cette valeur, il est possible d'utiliser un pas décroissant (par exemple $\alpha_k = \frac{1}{k+1}$) ou d'effectuer une recherche linéaire pour calculer un pas approprié à l'itération.

d) En $\boldsymbol{w}_k \in \mathbb{R}^d$, l'itération du gradient stochastique (avec pas α_k) se décompose en deux parties. On tire tout d'abord un indice i_k au hasard dans $\{1, \ldots, n\}$; on calcule ensuite le nouvel itéré $\boldsymbol{w}_{k+1}$ via

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \alpha_k \nabla f_{i_k}(\boldsymbol{w}_k) = \boldsymbol{w}_k - \alpha_k \ell'(\boldsymbol{x}_{i_k}^{\mathrm{T}}\boldsymbol{w}_k - y_{i_k})\boldsymbol{x}_{i_k}.$$

e) Chaque itération de descente de gradient doit accéder à toute la donnée pour calculer le gradient : comme notre unité de coût correspond à un accès à un point $\boldsymbol{x}_i$, le coût d'une itération de descente de gradient est de n. Quant à l'itération du gradient stochastique, elle a un coût de 1, puisqu'elle n'accède qu'à un point du jeu de données ($\boldsymbol{x}_{i_k}$, avec i_k tiré aléatoirement).

f) L'algorithme du gradient stochastique est aléatoire par nature, car son exécution dépend d'un tirage aléatoire d'une suite d'indices : il se peut donc qu'il ne converge pas (même s'il converge en moyenne), et c'est ce qui se produit ici.

i) L'itération de gradient stochastique par fournées (*batch*) de taille n_b en $\boldsymbol{w}_k \in \mathbb{R}^d$ consiste d'abord à tirer aléatoirement un ensemble d'indices $S_k \subset \{1, \ldots, n\}$ tel que $|S_k| = n_b$, puis à effectuer l'itération :

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \frac{\alpha_k}{|S_k|}\sum_{i \in S_k} \nabla f_i(\boldsymbol{w}_k),$$

avec $\alpha_k > 0$ la longueur de pas.

ii) Si n_b processeurs sont disponibles et que les gradients des f_i peuvent être calculés en parallèle, alors le coût de la fournée peut être réparti sur ces n_b processeurs.

iii) Ces méthodes utilisent un estimateur du gradient ($\frac{1}{|S_k|}\sum_{i \in S_k} \nabla f_i(\boldsymbol{w}_k)$) dont la variance est plus faible que celle de l'estimateur utilisé par le gradient stochastique ($\nabla f_{i_k}(\boldsymbol{w}_k)$).

iv) Utiliser plus d'un point à chaque itération permet de réduire la variance des itérations tout en bénéficiant de plus de données, ce qui peut expliquer pourquoi une taille de fournée $n_b = n/10$ conduit à une meilleure performance que $n_b = 1$ (qui correspond au gradient stochastique classique). En revanche, lorsque n_b se rapproche de n, le coût de la méthode se rapproche de celui d'une itération de descente de gradient, et dans le même temps la méthode est plus sensible aux redondances dans le jeu de données. Cela explique que la convergence de la méthode se dégrade, ici lorsque $n_b > n/10$.