OUTILS D'OPTIMISATION POUR LES SCIENCES DES DONNÉES ET DE LA DÉCISION

27 octobre 2023

Programme du jour :

- Cours gradient stochastique (1/2)
- TD gradient stochastique

GRADIENT STOCHASTIQUE

Contexte : Problème en somme finie

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

Hypothèse : chaque f_i dépend du $i$ème point dans un jeu de données à n éléments

Ex) $\{(x_i, y_i)\}_{i=1}^n$ avec $x_i \in \mathbb{R}^d$ et $y_i \in \mathbb{R}$

But : Trouver un modèle linéaire tel que $x_i^T w \approx y_i$

minimiser $f(w) = \frac{1}{2n} \sum_{i=1}^{n} (x_i^T w - y_i)^2$
$w \in \mathbb{R}^d$

$= \frac{1}{n} \sum_{i=1}^{n} f_i(w)$

avec $f_i(w) = \frac{1}{2}(x_i^T w - y_i)^2$

Forme générale en apprentissage (supervisé)

$\{(x_i, y_i)\}_{i=1}^n$

But : Trouver une fonction h telle que $h(x_i) \approx y_i$

=> En général, on considère que h est paramétrée par un vecteur $w \in \mathbb{R}^d$

=> On quantifie l'écart entre $h(x_i)$ et y_i via une fonction de perte

$$\ell : (h, y) \mapsto \ell(h, y)$$

On obtient le problème d'optimisation

$$\min_{w \in \mathbb{R}^d} f(w) = \frac{1}{n} \sum_{i=1}^{n} f_i(w)$$

avec $f_i(w) = \ell(h(x_i; w), y_i)$

Modèle défini par w

↳ h peut aussi bien être un modèle linéaire qu'un réseau de neurones

$$x_i = x^0 \mapsto x^1 \mapsto x^2 \mapsto \dots \mapsto x^L = h(x_i, w)$$

$$\forall l = 0 \dots L-1, \quad x^{l+1} = \sigma(W^l x^l + b^l)$$

$W^l \in \mathbb{R}^{d_{l+1} \times d_l}$

$b^l \in \mathbb{R}^{d_{l+1}}$

w : concaténation des W^l et b^l

$d = d_0 d_1 + d_1 + d_1 d_2 + d_2 + \dots$

$\sigma(v) = [\sigma(v_i)]$ activation

$\sigma(t) = \max(t, 0)$

$\sigma(t) = \tanh(t)$

BUT : Exploiter la structure en somme finie dans le processus d'optimisation et notamment dans un contexte de données massives ($n \gg 1$) où l'accès aux données peut être coûteux

1) Algorithme du gradient stochastique

↳ On considère un problème en somme finie et on suppose que les fonctions $f_i : \mathbb{R}^d \to \mathbb{R}$ sont de classe C^1.

$\Rightarrow f = \frac{1}{n} \sum f_i$ est aussi de classe C^1

et $\forall w \in \mathbb{R}^d$, $\nabla f(w) = \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(w)$

↳ Si on applique la descente de gradient à ce problème, on obtient l'itération suivante :

$$w_{k+1} = w_k - \alpha_k \nabla f(w_k)$$

avec $\alpha_k > 0$

$$= w_k - \frac{\alpha_k}{n} \sum_{i=1}^{n} \nabla f_i(w_k)$$

1 itération de descente de gradient doit accéder à tout le jeu de données pour calculer le prochain itéré

Itération de gradient stochastique (Robbins & Monro 1954)

$$w_{k+1} = w_k - \alpha_k \nabla f_{i_k}(w_k)$$

i_k tiré aléatoirement dans $\{1, 2, \dots, n\}$

⊕ 1 itération de gradient stochastique n'accède qu'à 1 point du jeu de données

⇒ n fois moins de descente de gradient en revanche coûteux qu'une itération d'accès aux données

? Est-ce que cette itération améliore l'objectif f et converge ?

↳ En général (sans hypothèses sur les f_i), cette méthode ne converge pas

↳ MAIS sur des données réelles (venant d'une distribution), la méthode converge en moyenne mieux que la descente de gradient !

2) Comparaison entre gradient stochastique et descente de gradient

↳ Rappel : La descente de gradient est une méthode déterministe et pour laquelle on peut montrer des vitesses de convergence

Ex) Si f est convexe, alors après $K \geq 1$ itérations de descente de gradient,

$f(w_K) - \min_{w \in \mathbb{R}^d} f(w) \leq O\left(\frac{1}{K}\right)$

↳ On peut prouver un résultat similaire sur le gradient stochastique sous certaines hypothèses, notamment

$$\forall k \in \mathbb{N}, \quad \mathbb{E}_{i_k}\left[\nabla f_{i_k}(w_k)\right] = \nabla f(w_k)$$

- $\mathbb{E}_{i_k}$: espérance moyenne par rapport à i_k
- $\nabla f_{i_k}(w_k)$: gradient stochastique i_k aléatoire dans $\{1, \dots, n\}$
- $\nabla f(w_k)$: "vrai gradient" $= \frac{1}{n}\sum \nabla f_i(w_k)$ = moyenne des ∇f_i

Ex) Si i_k est tiré uniformément au hasard dans $\{1, \dots, n\}$ (càd $\mathbb{P}(i_k = 1) = \dots = \mathbb{P}(i_k = n) = \frac{1}{n}$) alors

$$\mathbb{E}_{i_k}\left[\nabla f_{i_k}(w_k)\right] = \sum_{i=1}^{n} \mathbb{P}(i_k = i) \times \nabla f_i(w_k)$$

$$= \sum_{i=1}^{n} \frac{1}{n} \times \nabla f_i(w_k) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(w_k) = \nabla f(w_k)$$

Th) Si f est convexe, alors sous les bonnes hypothèses sur $\{i_k\}_{k\in\mathbb{N}}$, de gradient stochastique, après $K \geq 1$ itérations on a :

$$\mathbb{E}\left[f(w_K) - \min_{w \in \mathbb{R}^d} f(w)\right] \leq \mathcal{O}\left(\frac{1}{\sqrt{K}}\right)$$

- $\mathbb{E}$: espérance par rapport aux $\{i_k\}$

Interprétation : Le gradient stochastique, qui est une méthode aléatoire (ou "randomisée") converge en moyenne

. Dans le cas convexe, la vitesse de convergence est en $\frac{1}{\sqrt{k}}$, ce qui est plus lent que la vitesse en $\frac{1}{k}$ de la descente de gradient

⟹ Ces résultats suggèrent que le gradient Stochastique est moins efficace que la descente de gradient pour un même nombre d'itérations

⇒ Mais le coût d'une itération de gradient stochastique est différent de celui d'une itération de descente de gradient !

GS : 1 accès à un point du jeu de données

DG : n accès (accès à tout le jeu de données)

Métrique pertinente pour comparer les deux méthodes : Époques (epochs en anglais)

Def : 1 époque . Correspond à n accès à un point d'un jeu de données à n éléments

Avec cette définition :

1 itération de DG coûte 1 époque

et 1 itération de GS coûte $\frac{1}{n}$ époque

(⟺ 1 époque est le coût de n itérations de GS)

↳ La notion d'époque permet de comparer les algorithmes pour un budget d'accès aux données fixé, ce qui est une métrique

plus approprié que le nombre d'itérations

Ex: f convexe

	$K \geq 1$ itérations	$N_E \geq 1$ époques
DG	$\frac{1}{K}$	$\frac{1}{N_E}$ ← 1 époque = 1 itération
GS	$\frac{1}{\sqrt{K}}$	$\frac{1}{\sqrt{m N_E}}$ ← 1 époque = m itérations

Lorsque $m >> 1$, on a en général $\frac{1}{\sqrt{m N_E}} \ll \frac{1}{N_E}$

3) Gradient stochastique

↳ Plus général que "avec batch" par fournées

↳ Idée: Considérer plus de données lors d'une itération

le gradient stochastique d'un point du jeu de itération

Itération k

$$w_{k+1} = w_k - \frac{\alpha_k}{|S_k|} \sum_{i \in S_k} \nabla f_i(w_k)$$

↑ S_k est un ensemble d'indices tirés aléatoirement dans $\{1, \dots, m\}$ avec ou sans remise

S_k : "fournée"

$|S_k|$: nombre d'éléments de S_k "Taille de fournée"

Ex). $|S_k| = 1$ $S_k = \{i_k\}$

$w_{k+1} = w_k - \alpha_k \nabla f_{i_k}(w_k)$

⇒ Gradient stochastique

- $|S_k| = m$ et tirage sans remise

⇒ $S_k = \{1, \dots, m\}$

$w_{k+1} = w_k - \alpha_k \frac{1}{m} \sum_{i=1}^{m} \nabla f_i(w_k)$

$= w_k - \alpha_k \nabla f(w_k)$

$\Rightarrow$ Descente de gradient

↳ La méthode par gradient stochastique est la descente classique car particulières lorsque $|S_k| = 1$

↳ Pour les cas restants, on distingue deux grandes classes :

* Grande fournée : $|S_k| \gg 1$

  • Comportement proche de la descente de gradient

  • Coût très supérieur (par itération) à celui du gradient stochastique

  (Plutôt utilisé en combinaison avec d'autres variantes car très coûteux)

* Mini-fournées : $|S_k| > 1$ et $|S_k| \ll n$

  • Coût qui reste très inférieur à celui de la descente de gradient

  • Bénéfice possible par rapport au gradient stochastique (meilleure variance du gradient stochastique)

Remarque : En pratique, la frontière entre les deux classes est difficile à déterminer a priori et le choix de la taille de fournée est un problème complexe

↳ Un choix classique de taille de fournées est celui du nombre de processeurs disponibles pour des calculs en parallèle (ex: $|S_k| = 32$)

⇒ Ce cadre est favorable aux variantes par mini-fournées (souvent $n \gg 32$ donc impossible de paralléliser DG et GS ne se parallélise par construction)