Machine Learning

Dr. Mehran Safayani

safayani@iut.ac.ir

safayani.iut.ac.ir

https://www.aparat.com/mehran.safayani

https://github.com/safayani/machine_learning_course

Department of Electrical and computer engineering, Isfahan university of technology, Isfahan, Iran

Stochastic Gradient Descent

$L(\theta) = \frac{1}{m}\sum_{i=1}^{m} L_i(\theta)$ (cost function) $L_i = (\hat{y}_i - y_i)^2$

$L_i(\theta)$ = cost of ith training sample

SGD:

$$\theta^{t+1} = \theta^{t} - \alpha\nabla L_i(\theta^{t})$$

$\nabla L_i(\theta^t)$:

یک تخمین unbiase از $\nabla L_i(\theta^t)$ است و محاسباتش کم هزینه تر است.

می توان نشان داد:

$$E[\nabla L_i(\theta)] = \nabla L(\theta)$$

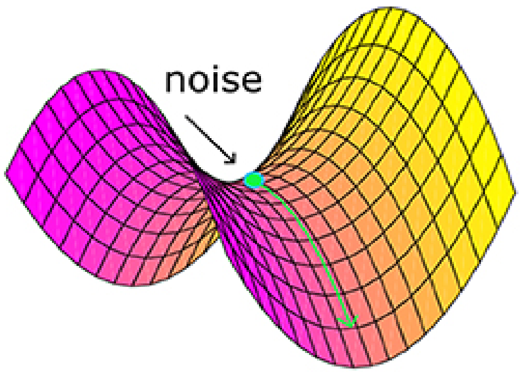
Mini_Batch SGD

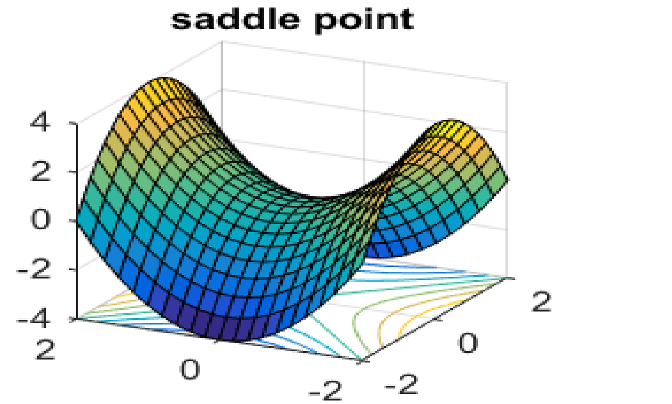
L = $\frac{1}{|B|}\sum_{i \in B} L_i(\theta^t)$ (B: تعداد)

$$\theta^{t+1} = \theta^t - \alpha g \qquad g = \frac{d\mathrm{L}}{d\theta}$$

- یک مجموعه تصادفی به اندازه $|B|$ از داده های آموزشی انتخاب می کنیم.
- امکان موازی سازی با Mini_Batch SGD بیشتر از SGD است.
- حجم محاسبات : O($|B|.n$)

Saddle Points





به کمک **SGD** میتوان از نقطه زین اسبی فرار کرد

GD

Repeat{

$d\theta = 0$

for i = 1 to m:

compute $d\theta^{i}$

$d\theta$ += $d\theta^{i}$

$\theta = \theta - \alpha \frac{1}{m} d\theta$

} until convergence

mini_batch SGD

T = $\frac{m}{B}$; B = batch_size

Repeat{

for j= 1 to T:

$d\theta = 0$

for i = 1 to B:

compute $d\theta^{i}$

$d\theta$ += $d\theta^{i}$

$\theta = \theta - \alpha \frac{1}{B} d\theta$

}until convergence

SGD

Repeat{

for i = 1 to m:

compute $d\theta^{i}$

$\theta = \theta - \alpha\, d\theta^{i}$

} until convergence

Comparison

SGD

از روش های vectorization به خوبی استفاده نمی شود.

سرعت الگوریتم کاهش می یابد.

خیلی نویزی است.

Mini_Batch GD

آموزش سریعتر

Batch GD

هر تکرار خیلی طول می کشد.

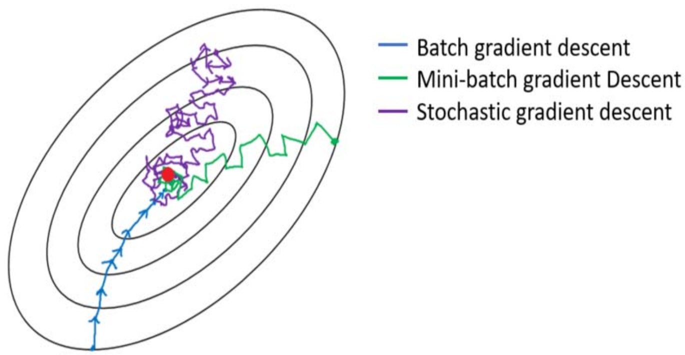
ماتریس ها خیلی بزرگ هستند.

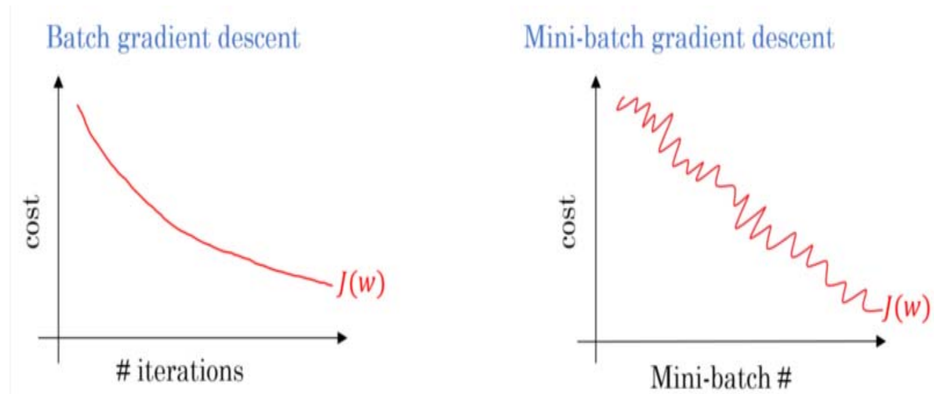
موازی سازی مشکل است.

$m \leq 2000$: Batch

Mini_Batch: 64 , 128, 256 , 512

Comparison





Robbines_Monro Algorithm:

$\sum \alpha^t = \infty$ $\sum_{t=1}^{\infty}(\alpha^t)^2 < \infty$ $\alpha^t = \frac{1}{(t+1)^r}$ $r \in (0.5, 1)$

$$\{x^1, x^2, \dots, x^{1000}\} \quad , \qquad x^1 = x^2 = \dots = x^{1000}$$

$$\frac{1}{1000}\sum_{i=1}^{1000} L(\hat{y}_i, y_i) = \frac{1}{\cancel{1000}} * \cancel{1000}\, L(\hat{y}_i, y_i)$$

$$\left.\begin{matrix} x^1 \\ x^2 \\ \dots \\ x^{1000} \end{matrix}\right\} w^1 \text{ updated}$$

$$\left.\begin{matrix} x^1 \\ x^2 \\ \dots \\ x^{1000} \end{matrix}\right\} w^2 \text{ updated}$$

.....

$$\left.\begin{matrix} x^1 \\ x^2 \\ \dots \\ x^{1000} \end{matrix}\right\} w^{1000} \text{ updated}$$

$$w^1 = w^{*1}$$

$$\equiv$$

SGD:

$$\begin{matrix} 1 \\ \text{epoch} \end{matrix} \left\{\begin{matrix} x^1\text{:}\ \ w^{*1} \text{ updated} \\ x^2\text{:}\ \ w^{*2} \text{ updated} \\ x^3\text{:}\ \ w^{*3} \text{ updated} \\ . \\ . \\ . \\ x^{1000}\text{:}\ \ w^{*1000} \text{ updated} \end{matrix}\right.$$

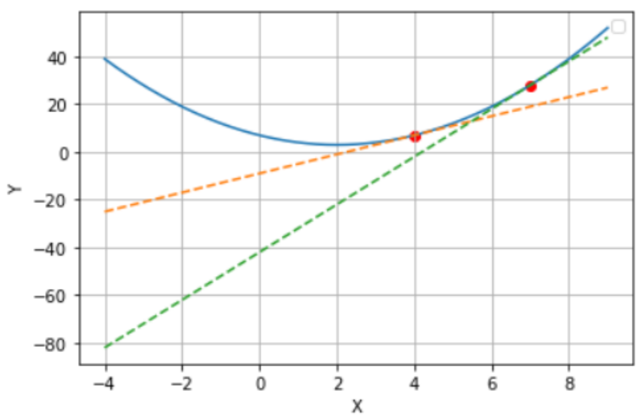
Subgradient Method

این روش برای توابعی که در برخی نقاط مشتق پذیر نیستند بکار می رود.

برای توابع محدب مشتق پذیر داریم:

$$\mathcal{L}(\mathbf{u}) \geq \mathcal{L}(\mathbf{w}) + \nabla \mathcal{L}(\mathbf{w})^{\top}(\mathbf{u} - \mathbf{w}) \quad \forall \mathbf{u}, \mathbf{w}$$

بدین معنی که تابع همیشه بزرگتر از تخمین خطی اش است.



Subgradient Method

Subgradient:

A vector $\mathbf{g} \in \mathbb{R}^D$ such that

$$\mathcal{L}(\mathbf{u}) \geq \mathcal{L}(\mathbf{w}) + \mathbf{g}^\top(\mathbf{u} - \mathbf{w}) \quad \forall \mathbf{u}$$

is called a subgradient to the function $\mathcal{L}$ at $\mathbf{w}$.

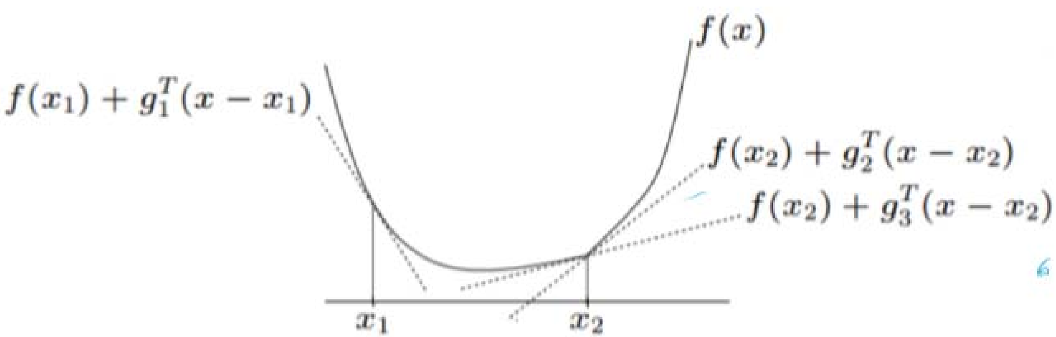
اگر تابع L(w) مشتق پذیر باشد $g = \Delta L(w)$ است.

Subgradient Descent:

$$\mathbf{w}^{(t+1)} := \mathbf{w}^{(t)} - \gamma\, \mathbf{g}$$

Subgradient Method

$$|x| \longrightarrow \quad g = \begin{cases} 1 & x > 0 \\ [-1\,,1] & x = 0 \\ -1 & x < 0 \end{cases}$$



$$\boxed{\mathbf{g}(\boldsymbol{x_i}) = [\boldsymbol{g_3}\,, \boldsymbol{g_2}]}$$