梯度下降法

相关数学概念

□ 导数(derivative)

导数表示的是因变量相对于自变量的变化率，如$y = x^2$的导数为$\frac{dy}{dx} = 2x$

□ 偏导数(panrtial derivative)

偏导数表示的是多元函数某个方向（一般是某个轴向）的变化率

$$z = y^2 + x^2 \qquad \frac{\partial z}{\partial x} = 2x \qquad \frac{\partial z}{\partial y} = 2y$$

□ 梯度(gradient)

对多元函数每一个轴向求偏导函数，这些偏导函数组成的向量就称之为梯度

若有一函数：$f(x) = f(x_1, x_2, \dots, x_n)$，则该函数的梯度为$\nabla f = (\frac{\partial f}{\partial x_1}; \frac{\partial f}{\partial x_2}; \dots; \frac{\partial f}{\partial x_n})$

例： 函数 $F(x_1, x_2) = (3x_1 + 4x_2)^2$ ，则对于x_1，x_2的偏导分别为：$\frac{\partial F}{\partial x_1} = 18x_1 + 24x_2 \quad \frac{\partial F}{\partial x_2} = 24x_1 + 32x_2$

则F在(2,3)时的梯度为$(18 \times 2 + 24 \times 3, 24 \times 2 + 32 \times 3) = (108,144)$，记为：$\frac{\partial F}{\partial x|_{x=(2,3)}} = \begin{bmatrix}108\\144\end{bmatrix}$

一元函数梯度下降法推导

若函数$f(x)$在包含x_0的某个闭区间$[a,b]$上具有n阶导数，且在开区间(a,b)上具有$n+1$阶导数，则对闭区间$[a,b]$上任意一点x，泰勒展开式成立

$$f(x)=\frac{f(x_0)}{0!}+\frac{f'(x_0)}{1!}(x-x_0)+\frac{f''(x_0)}{2!}(x-x_0)^2+\cdots+\frac{f^{(n)}(x_0)}{n!}(x-x_0)^n+R_n(x)$$

$f(x)$在x_1处的一阶泰勒展开式：$f_{taylor_1}(x)=f(x_1)+f'(x_1)(x-x_1)$

当x在离x_1很近的距离时：$f(\mathrm{x})\approx f_{taylor_1}(x)$

此时取 $x_2=x_1-\eta f'(x_1)$，则$f(x_2)\approx f_{taylor_1}(x_2)=f(x_1)+f'(x_1)(x_2-x_1)$

根据x_2的取值$[x_2=x_1-\eta f'(x_1)]$，则$f'(x_1)(x_2-x_1)=f'(x_1)(x_1-\eta f'(x_1)-x_1)=-\eta(f'(x_1))^2\leq 0$

因此：$f(x_2)\approx f(x_1)-\eta(f'(x_1))^2$，易得$f(x_2)<f(x_1)$

以此类推：$x_{n+1}=x_n-\eta f'(x_n)$,使得$f(x_{n+1})\leq f(x_n)$

一般情况下迭代停止标准：$f(x_{n+1})-f(x_n)\approx f_{taylor_1}(\mathrm{x}_{n+1})-f(x_n)=-\eta(f'(x_n))^2\leq\varepsilon$

多元函数梯度下降法推导

若函数$f(x)$在包含$x^{(0)}$的某个闭区间$[a,b]$上具有n阶导数，且在开区间(a,b)上具有$n+1$阶导数，则对闭区间$[a,b]$上任意一点x，泰勒展开式成立

$$f(x^1,x^2,\ldots,x^n)=f\left(x_k^1,x_k^2,\ldots,x_k^n\right)+\sum_{i=1}^{n}(x^i-x_k^i)f'_{x^i}\left(x_k^1,x_k^2,\ldots,x_k^n\right)$$

$$+\frac{1}{2!}\sum_{i,j=1}^{n}(x^i-x_k^i)(x^j-x_k^j)\,f''_{x^i}\left(x_k^1,x_k^2,\ldots,x_k^n\right)+\cdots+o^n$$

$f(x)$ 在 $x^{(1)}$ 处的一阶泰勒展开式:$f_{taylor_1}(x)=f\left(x^{(1)}\right)+\left(x-x^{(1)}\right)^T\boxed{\nabla f(x^{(1)})}\longrightarrow \nabla f\left(x^{(1)}\right)=[\frac{\partial f}{\partial x_1^{(1)}},\frac{\partial f}{\partial x_2^{(1)}}]$

当x在离$x^{(1)}$很近的距离时: $f(\mathrm{x})\approx f_{taylor_1}(x)$

此时取$x^{(2)}=x^{(1)}-\eta\nabla f\left(x^{(1)}\right)$，则$f\left(x^{(2)}\right)\approx f_{taylor_1}\left(x^{(2)}\right)=f\left(x^{(1)}\right)+\nabla f\left(x^{(1)}\right)(x^{(2)}-x^{(1)})^T$

根据$x^{(2)}$的取值$x^{(2)}=x^{(1)}-\eta\nabla f\left(x^{(1)}\right)$，则$\nabla f\left(x^{(1)}\right)(x^{(2)}-x^{(1)})^T=\nabla f\left(x^{(1)}\right)(-\eta\nabla f\left(x^{(1)}\right))^T=-\eta(||\nabla f\left(x^{(1)}\right)||_2)^2\leq 0$

以此类推: $x^{(n+1)}=x^{(n)}-\eta\nabla f\left(x^{(n)}\right)$,使得$f(x^{(n+1)})\leq f(x^{(n)})$

梯度下降法

梯度下降算法

$J(w)$

$$w = w - \alpha \frac{\partial J(w)}{\partial w}$$

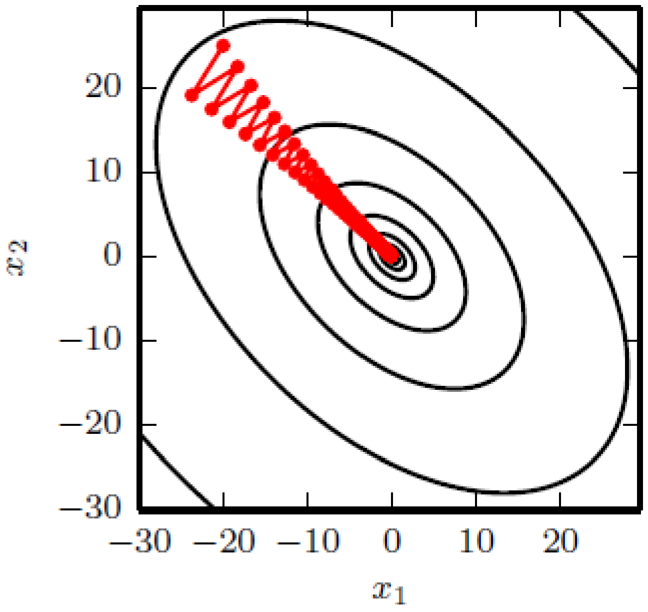
$$\min_{w} J(w)$$

Repeat until convergence{

$$w = w - \alpha \frac{\partial J(w,b)}{\partial w}$$

$$b = b - \alpha \frac{\partial J(w,b)}{\partial b}$$

}



正确方式：同步更新

tmp_w= $w - \alpha \frac{\partial J(w,b)}{\partial w}$

tmp_b= $b - \alpha \frac{\partial J(w,b)}{\partial b}$

w=tmp_w

b=tmp_b

错误方式：单独更新

tmp_w= $w - \alpha \frac{\partial J(w,b)}{\partial w}$

w=tmp_w

tmp_b= $b - \alpha \frac{\partial J(w,b)}{\partial b}$

b=tmp_b

梯度下降法

■ 梯度下降算法

线性回归模型：$f_{w,b}(x) = wx + b$

损失函数：$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})^2$

$$\frac{\partial J(w,b)}{\partial w} = \frac{1}{2m}\sum_{i=1}^{m}(wx^{(i)} + b - y^{(i)})2x^{(i)}$$

$$= \frac{1}{m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})x^{(i)}$$

$$\frac{\partial J(w,b)}{\partial b} = \frac{1}{2m}\sum_{i=1}^{m}(wx^{(i)} + b - y^{(i)})2$$

$$= \frac{1}{m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})$$

Repeat until convergence{

$$w = w - \alpha\frac{\partial J(w,b)}{\partial w}$$

$$b = b - \alpha\frac{\partial J(w,b)}{\partial b}$$

}

Repeat until convergence{

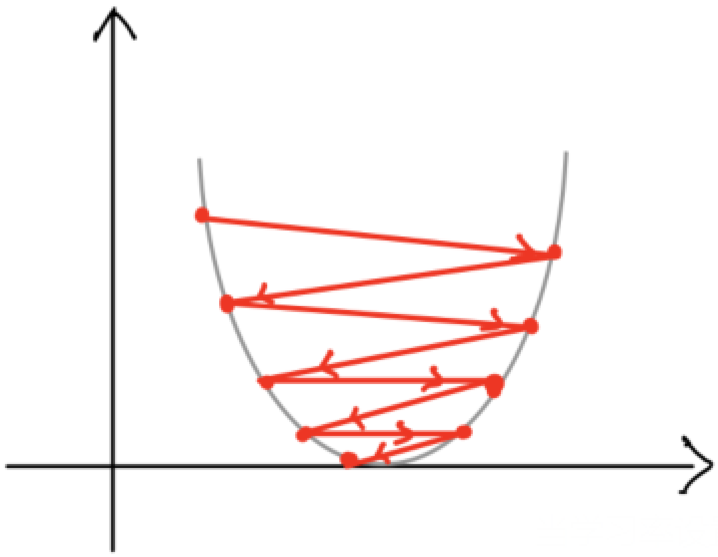
$$w = w - \alpha\frac{1}{m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})x^{(i)}$$

$$b = b - \alpha\frac{1}{m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})$$

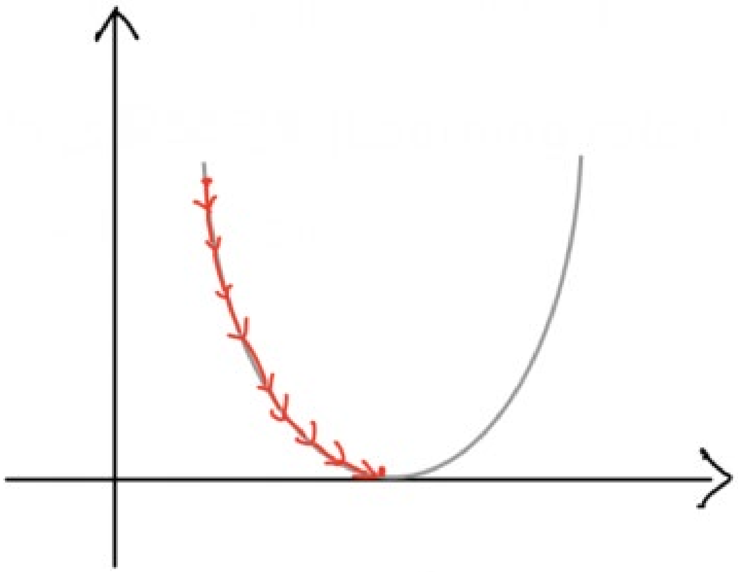
}

■ 学习率选择

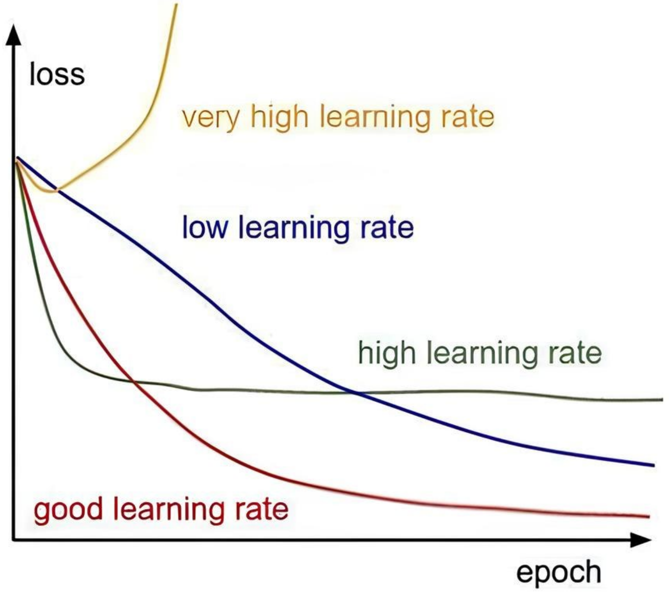
□ 学习率大



□ 学习率小



□ 学习率对训练的影响



梯度下降法

优化梯度下降法

SGD梯度下降法

即随机梯度下降法，在每轮迭代时从数据集中随机挑选一定数量的数据进行计算

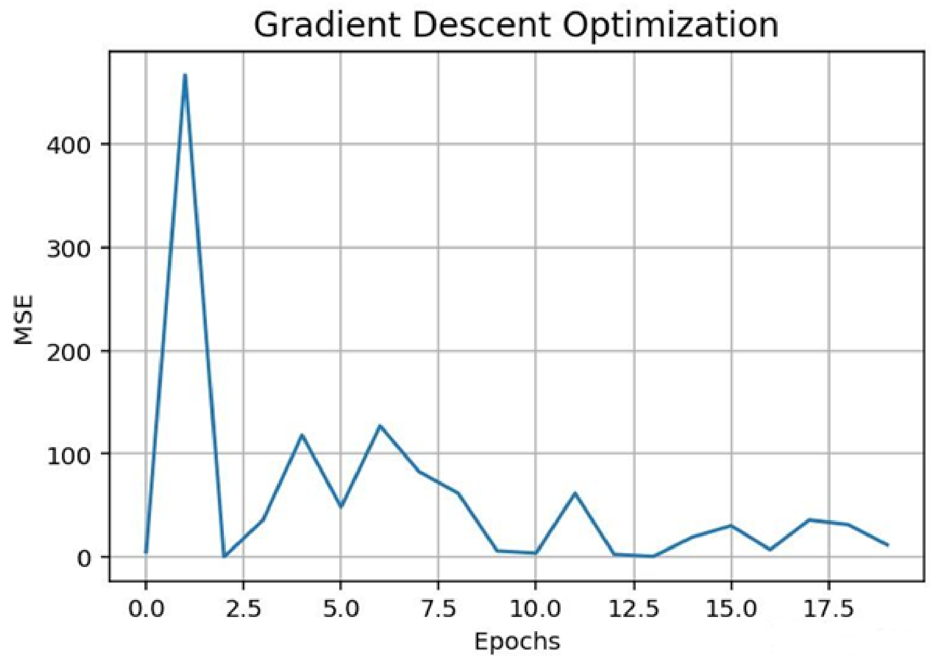
损失函数：$f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x)$

梯度下降法：$\nabla f(x) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(x)$

随机梯度下降法：$\nabla f(x) = \frac{1}{m}\sum_{i=1}^{m} \nabla f_i(x)$

$f(x)$ 是目标损失函数

$f_i(x)$是第i个样本所对应的损失函数



优化梯度下降法

Momentum梯度下降法

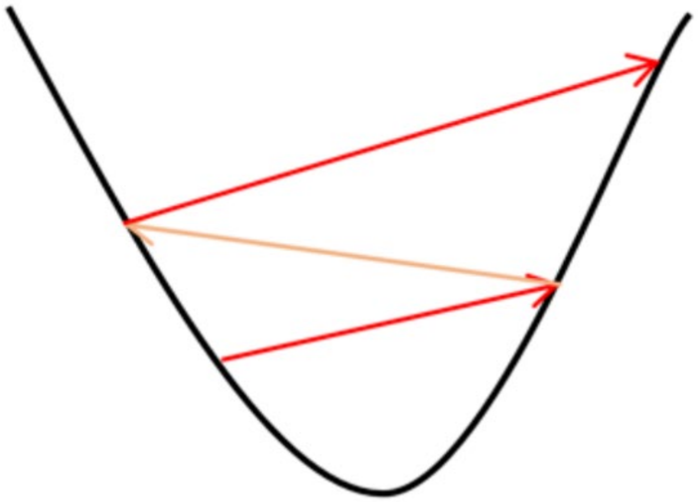
引入动量，使得因学习率过大而来回摆动的参数，梯度能前后抵消，阻止发散

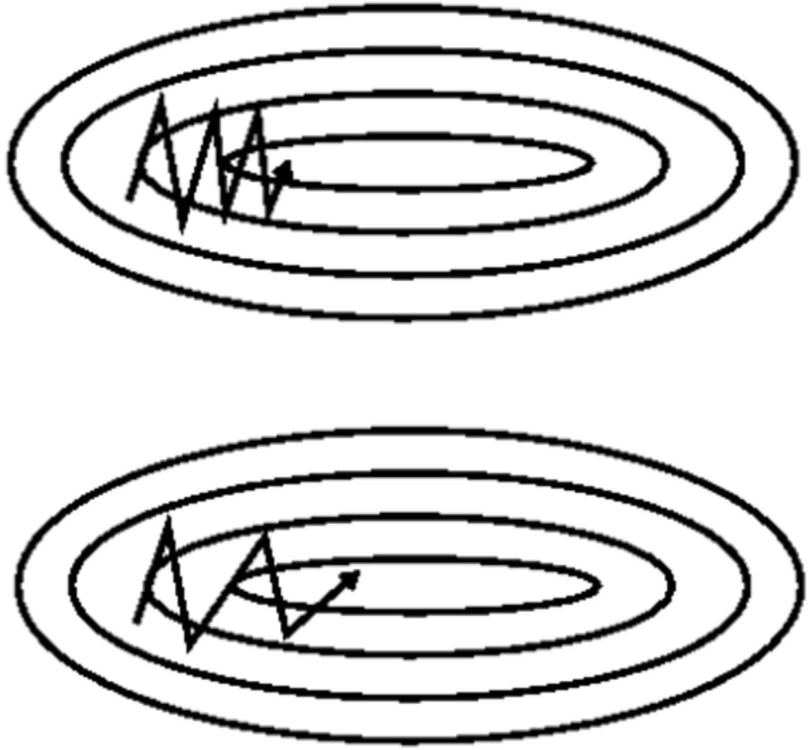
$$v_t = \beta v_{t-1} + \eta \nabla J(w)$$

$$w = w - v_t$$

其中：v_t表示当前动量

$\nabla J(w)$为目标函数的当前梯度





■ 优化梯度下降法

☐ NAG梯度下降法

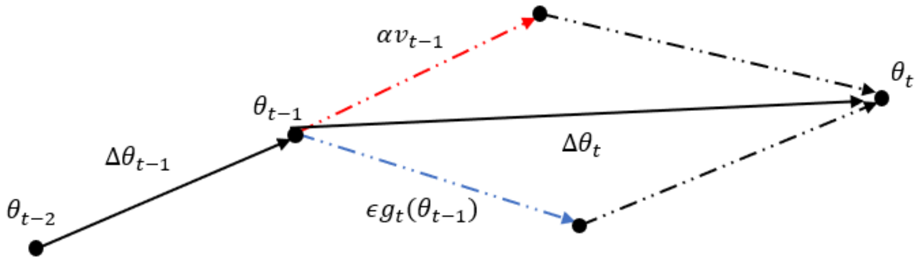
Nesterov加速梯度下降法，查看当前累积动量所在位置，再去计算该位置的梯度

$$\mathrm{v_t} = \beta v_{t-1} + \eta \nabla J(w - \beta v_{t-1})$$

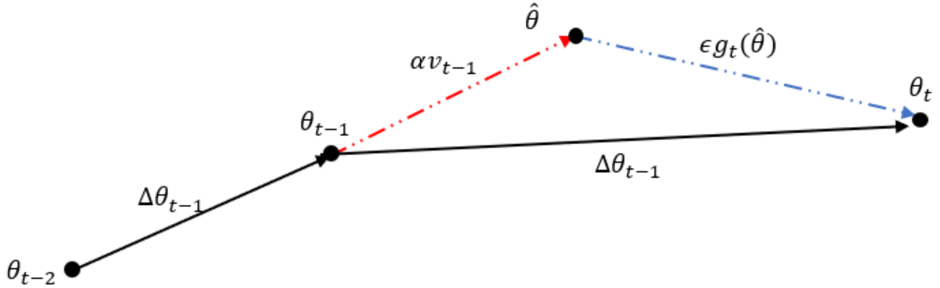
$$w = w - v_t$$

其中：$\mathrm{v_t}$表示当前动量

$\nabla J(w - \beta v_{t-1})$为在累计动量方向的梯度



动量法



Nesterov 加速梯度法

梯度下降法

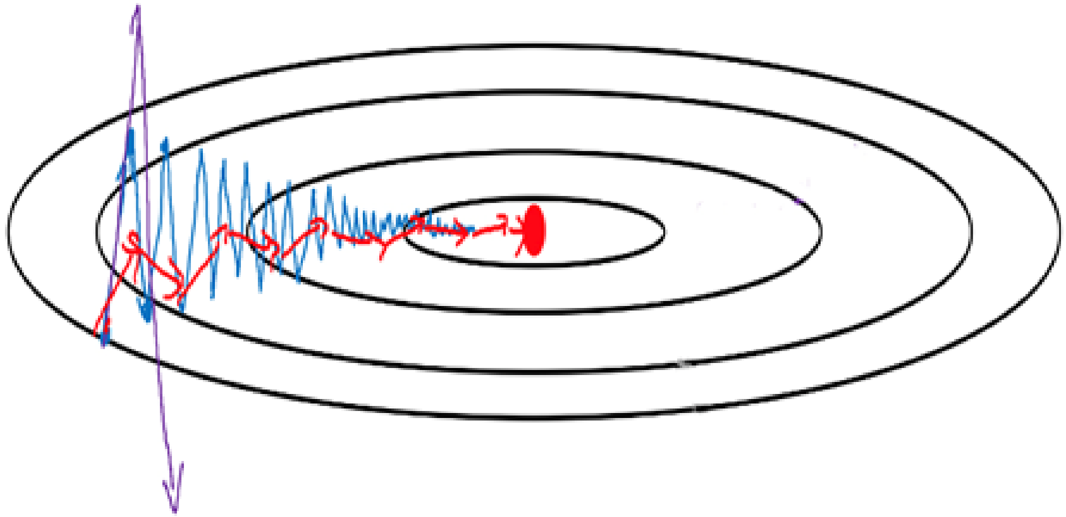
优化梯度下降法

RMSprop梯度下降法

当目标函数有很多自变量时，收敛速度较快的自变量方向使用较大步长，收敛速度慢的地方使用较小步长

$$g^{(n)} = \rho g^{(n-1)} + (1-\rho)\left(\frac{\partial f}{\partial x}_{|x=x^{(n)}}\right)^2$$

$$x^{(n+1)} = x^{(n)} - \frac{\eta}{\sqrt{g^n + \varepsilon}} \frac{\partial f}{\partial x}_{|x=x^n}$$



优化梯度下降法

Adagrad梯度下降法

针对不同的参数自适应的调节对应的学习率，若其梯度累计值较大则实际学习率小一些，若梯度累计值较小则实际学习率大一些

$$w_{(t)i} = w_{(t-1)i} - \frac{\eta}{\sqrt{S_{(t)} + \varepsilon}} \Delta w_{(t)i}$$

其中：$S_{(t)} = S_{(t-1)} + \Delta w_{(t)i}{}^{2}$

$$\Delta w_{(t)i} = \frac{\partial J(w_{(t-1)i})}{\partial w_j}$$

- η为初始学习率
- ϵ是为了数值稳定性而加上的，通常 ϵ 取10的负10次方
- 不同的参数由于梯度不同，他们对应的 s 大小也就不同，学习率也就不同

■ 优化梯度下降法

□ 梯度下降法比较