Gradient Descent

梯度下降的目标是使损失函数L最小化，$\theta^* = arg\ min\ L(\theta)$

Suppose that θ has two variables $\{\theta_1, \theta_2\}$

Randomly start at $\theta^0 = \begin{bmatrix}\theta_1^0 \\ \theta_2^0\end{bmatrix}$

$$\nabla L(\theta) = \begin{bmatrix}\partial L(\theta_1)/\partial\theta_1 \\ \partial L(\theta_2)/\partial\theta_2\end{bmatrix}$$

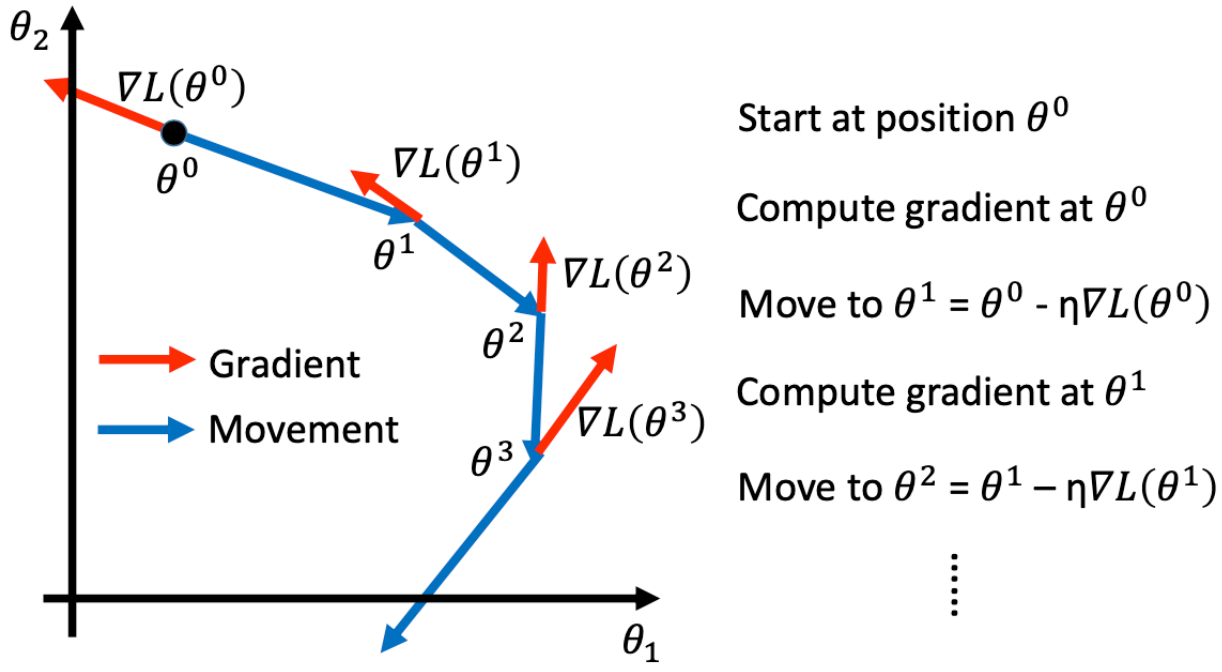
$$\begin{bmatrix}\theta_1^1 \\ \theta_2^1\end{bmatrix} = \begin{bmatrix}\theta_1^0 \\ \theta_2^0\end{bmatrix} - \eta \begin{bmatrix}\partial L(\theta_1^0)/\partial\theta_1 \\ \partial L(\theta_2^0)/\partial\theta_2\end{bmatrix} \Rightarrow \theta^1 = \theta^0 - \eta\nabla L(\theta^0)$$

$$\begin{bmatrix}\theta_1^2 \\ \theta_2^2\end{bmatrix} = \begin{bmatrix}\theta_1^1 \\ \theta_2^1\end{bmatrix} - \eta \begin{bmatrix}\partial L(\theta_1^1)/\partial\theta_1 \\ \partial L(\theta_2^1)/\partial\theta_2\end{bmatrix} \Rightarrow \theta^2 = \theta^1 - \eta\nabla L(\theta^1)$$

Learning Rate

从θ_0开始，先计算出θ_0的梯度，其中红色箭头表示梯度的方向，蓝色箭头表示移动的方向。

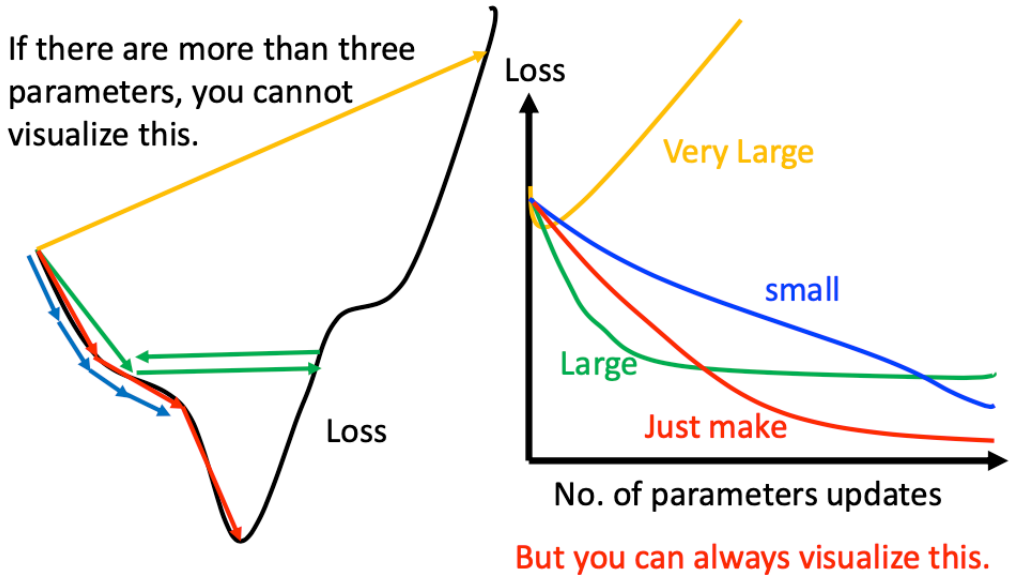
梯度的方向是函数值在这个点增长最快的方向，想要使损失函数L的值达到最小值，就必须要往相反的方向运动。



学习率会出现以下四种不同的情况：

- 学习率太小，即图中蓝色的线，每次跨越的步长很小很小，梯度每次变化的值也小，模型要达到local minima，就必须需要更多的训练时间；
- 学习率太大，即图中绿色的线，每次跨越的步长会很大，很可能形成在山谷之间震荡的现象；
- 学习率特别大，即图中黄色的线，就很可能会直接跳出local minima，loss会越来越大；

- 学习率刚好合适，即图中红色的线，每次跨越的步长非常合适，达到local minima的时间也不需要特别多。



由于手动设置learning rate会导致很多问题，就出现了一些自适应的梯度调整方法。

- 刚开始训练时，我们离local minimum的距离还很远，因此可以使用稍大的learning rate；
- 在经过多次的训练后，离local minimum的距离已经很近了，所以这时可以使用小的learning rate；
- 在经过t次的训练后，learning rate可以衰减为

  $\eta^t = \frac{\eta}{\sqrt{t+1}}$

Adagrad

> Divide the learning rate of each parameter by the **root mean square of its previous derivatives**

理论推导

使用这个公式来更新参数w，

$w^{t+1} \leftarrow w^t - \frac{\eta^t}{\sigma^t} g^t$

其中，t表示第t次的update，$g^t = \frac{\partial L(\theta^t)}{\partial w}$，是损失函数L对参数w的导数，$\sigma^t$表示其先前导数的均方根（root mean square）

计算w的具体例子如下所示，

$$w^1 \leftarrow w^0 - \frac{\eta^0}{\sigma^0} g^0 \qquad \sigma^0 = \sqrt{(g^0)^2}$$

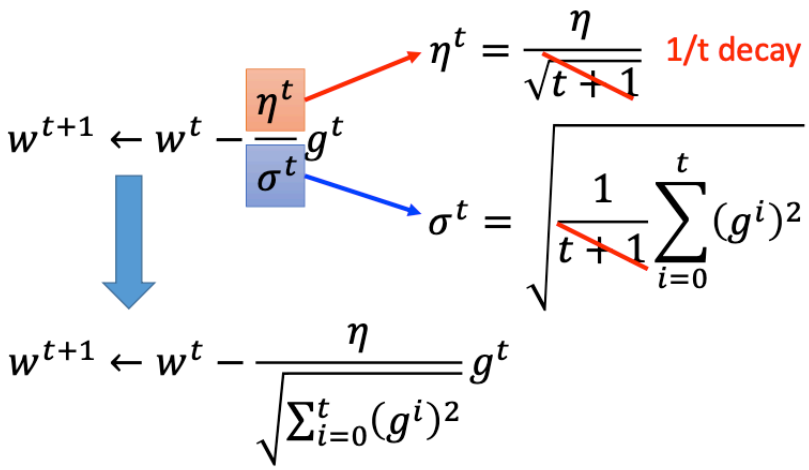
$$w^2 \leftarrow w^1 - \frac{\eta^1}{\sigma^1} g^1 \qquad \sigma^1 = \sqrt{\frac{1}{2}[(g^0)^2 + (g^1)^2]}$$

$$w^3 \leftarrow w^2 - \frac{\eta^2}{\sigma^2} g^2 \qquad \sigma^2 = \sqrt{\frac{1}{3}[(g^0)^2 + (g^1)^2 + (g^2)^2]}$$

$$\vdots$$

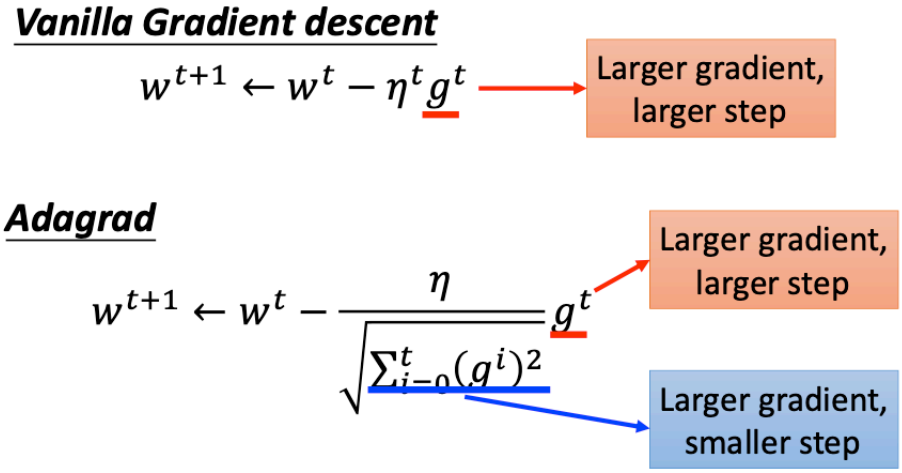
$$w^{t+1} \leftarrow w^t - \frac{\eta^t}{\sigma^t} g^t \qquad \sigma^t = \sqrt{\frac{1}{t+1}\sum_{i=0}^{t} (g^i)^2}$$

得出了σ^t和η^t的表达式后，再带入原式，消除分母上的$\sqrt{t+1}$即可得出下面的公式，

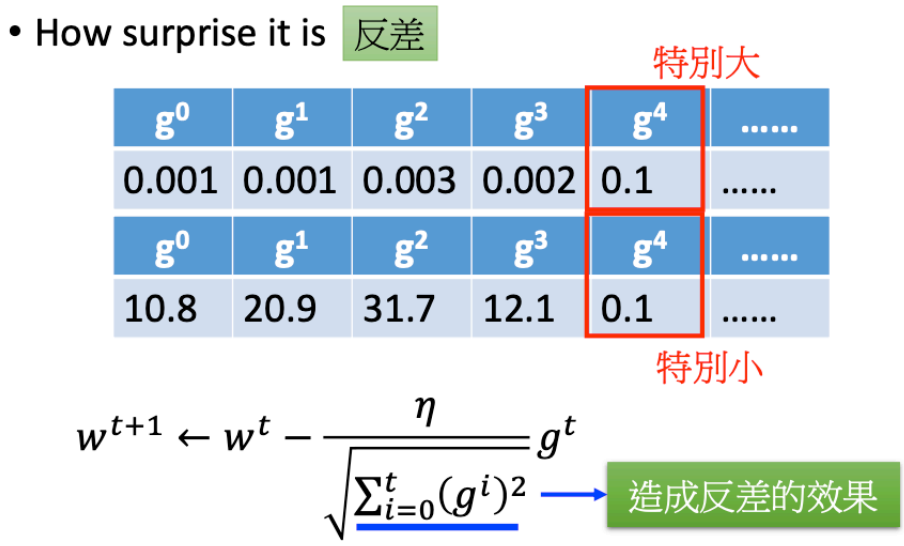


Contradiction

如上图所示，对于一般的梯度下降算法（vanilla gradient descent），当梯度g越大时，步长就越大；对于Adagrad，g^t在分子上，梯度越大步长也越大，$\sum_{i=0}^{t}(g^i)^2$在分母上，数值越大步长也就越小，看似出现了一个矛盾。



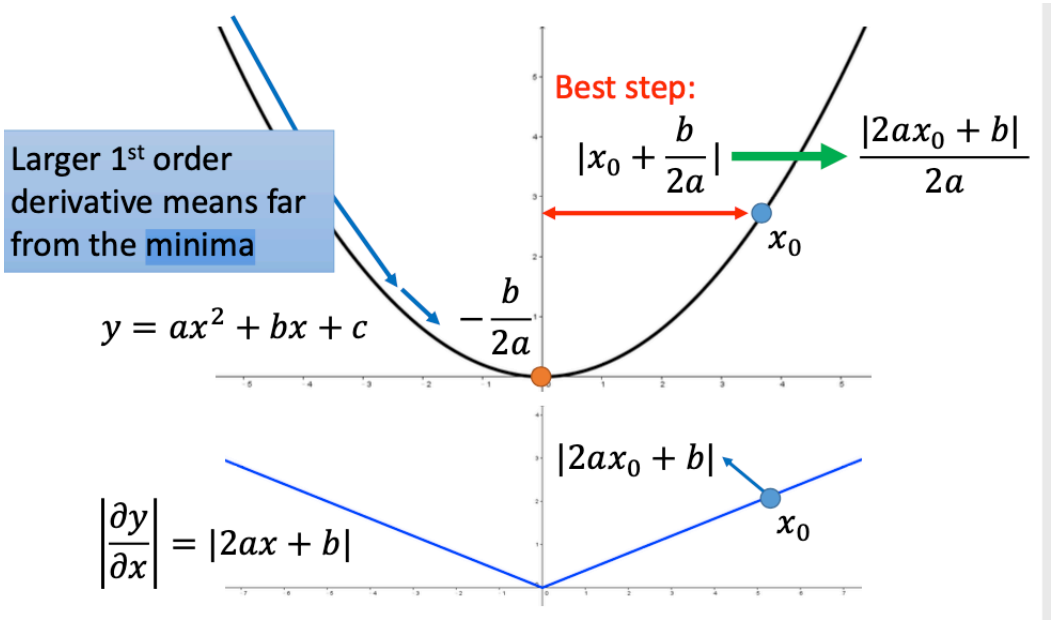
有学者对此也做出了解释，认为Adagrad可以解释g^t和$\sum_{i=0}^{t}(g^i)^2$之间的反差，造成了反差的效果。



g^0	g^1	g^2	g^3	g^4	.....
0.001	0.001	0.003	0.002	0.1	.....
g^0	g^1	g^2	g^3	g^4	.....
10.8	20.9	31.7	12.1	0.1	.....

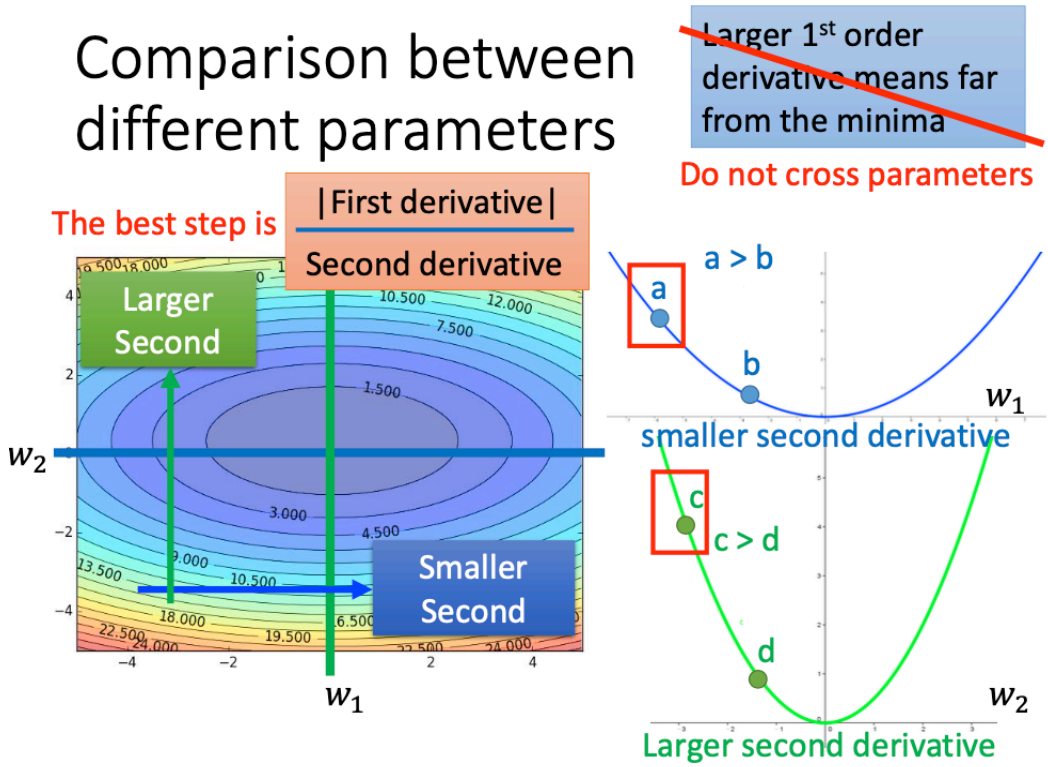
$$w^{t+1} \leftarrow w^t - \frac{\eta}{\sqrt{\sum_{i=0}^{t}(g^i)^2}} g^t$$

gradient越大，函数值离minima的距离就越远这个说法不一定在所有情况下都是成立的。



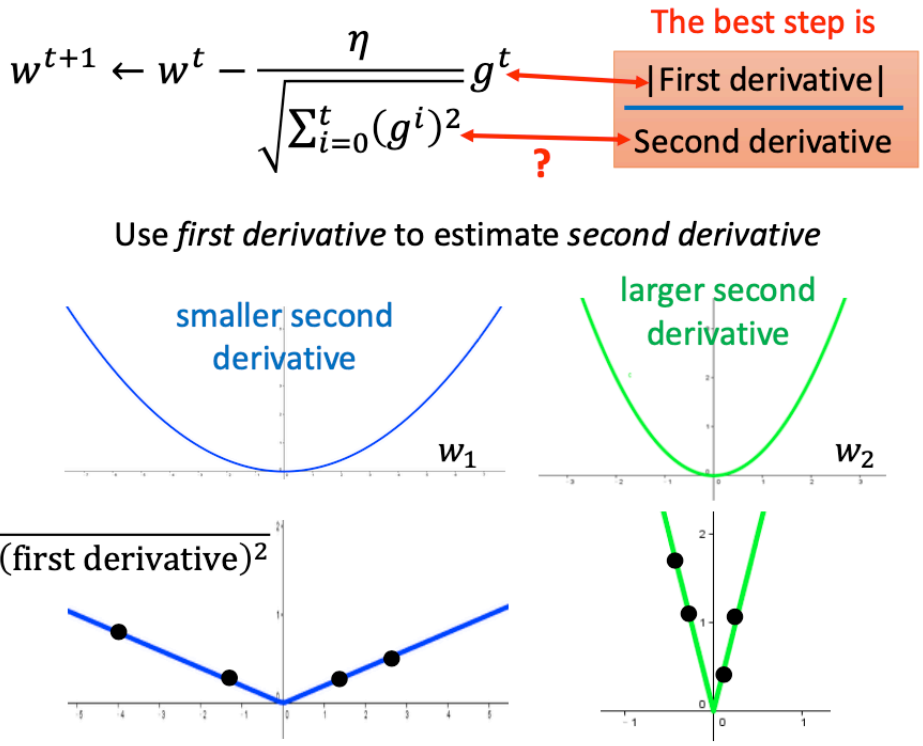
对于左图中的两个参数w1和w2，画两条直线，保持其中一个变量不变，得出另一个变量的变化曲线，分别对应右图中的曲线。在右图中，对于w1中的a点和w2中的c点，c点距离minimum的距离最近，但梯度却更大。因此在分析梯度和步长时，我们不能只考虑一阶导数的大小，还必须要要考虑二阶导数的大小，即$y'' = 2a$。

右图中的w1曲线，曲率半径比w2的曲线更大，一阶导数变化得更平缓，因此二阶导数的变化就比w2大



再来回顾下Adagrad中每次更新w的表达式，$w^{t+1} \leftarrow w^t - \frac{\eta}{\sqrt{\sum_{t=0}^{t}(g^i)^2}} g^t$

一阶导数用g^t表示，二阶导数的值则用分母中的$\sum_{i=0}^{t}(g^i)^2$来进行评估，即使用一阶导数的值来表示二阶导数的值。



Stochastic Gradient Descent

对于传统的梯度下降算法，损失函数L的计算包含了所有的样本;

随机梯度下降算法，损失函数L^n则只使用其中一个样本，计算效率可以提高很多

$$L = \sum_n \left(\hat{y}^n - \left(b + \sum w_i x_i^n \right) \right)^2$$

Loss is the summation over all training examples

◆***Gradient Descent*** $\theta^i = \theta^{i-1} - \eta \nabla L\left(\theta^{i-1}\right)$

◆***Stochastic Gradient Descent*** Faster!

Pick an example x^n

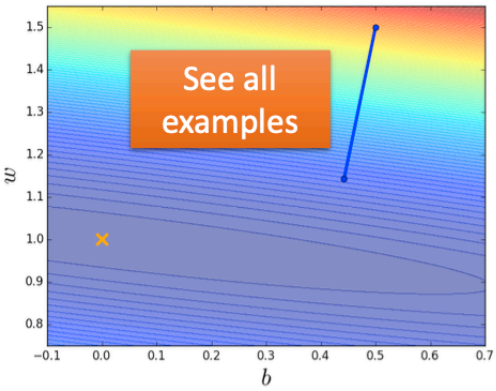
$$L^n = \left(\hat{y}^n - \left(b + \sum w_i x_i^n \right) \right)^2 \qquad \theta^i = \theta^{i-1} - \eta \nabla L^n\left(\theta^{i-1}\right)$$

Loss for only one example

对比示意图如下，

Gradient Descent

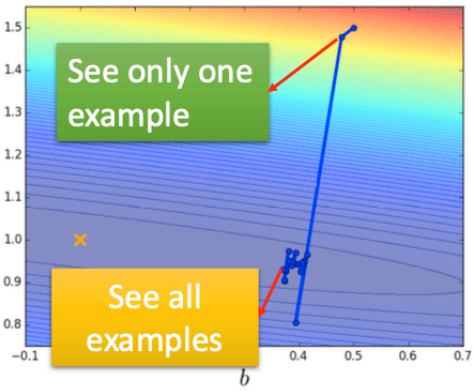
Update after seeing all examples



Stochastic Gradient Descent

Update for each example

If there are 20 examples, 20 times faster.

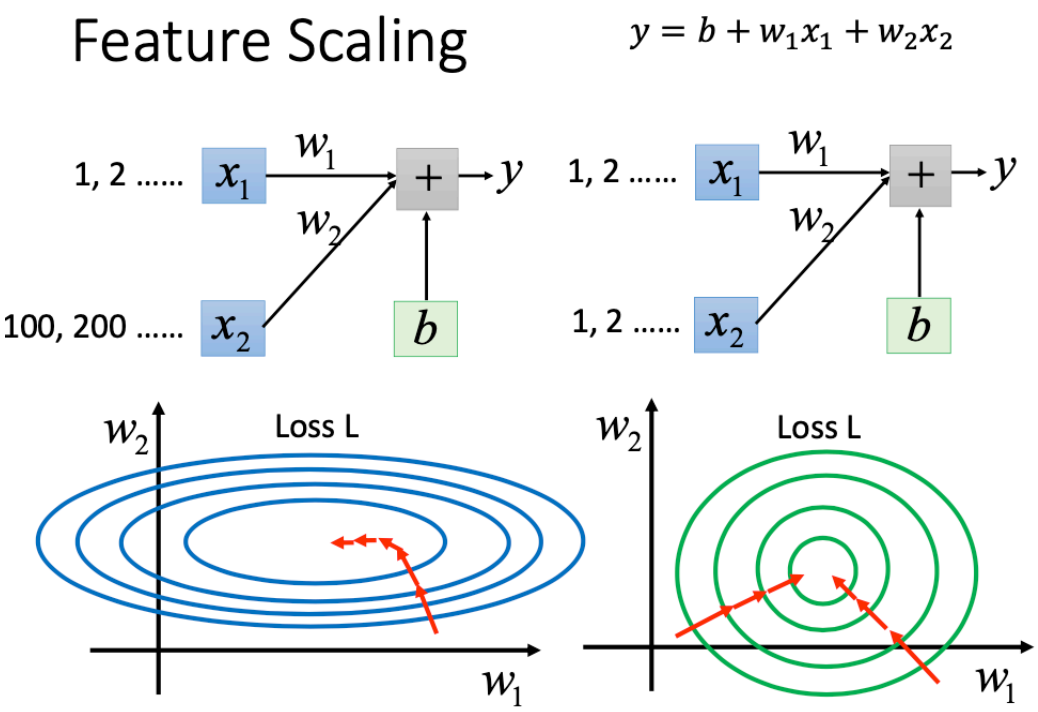


Feature scaling

原理介绍

> Make different features have the same scaling

使不同量级的数据集都具有相同的规模，比如x2的都是大于100的值，经过feature scaling，就可以使x2的数值范围和x1相接近。

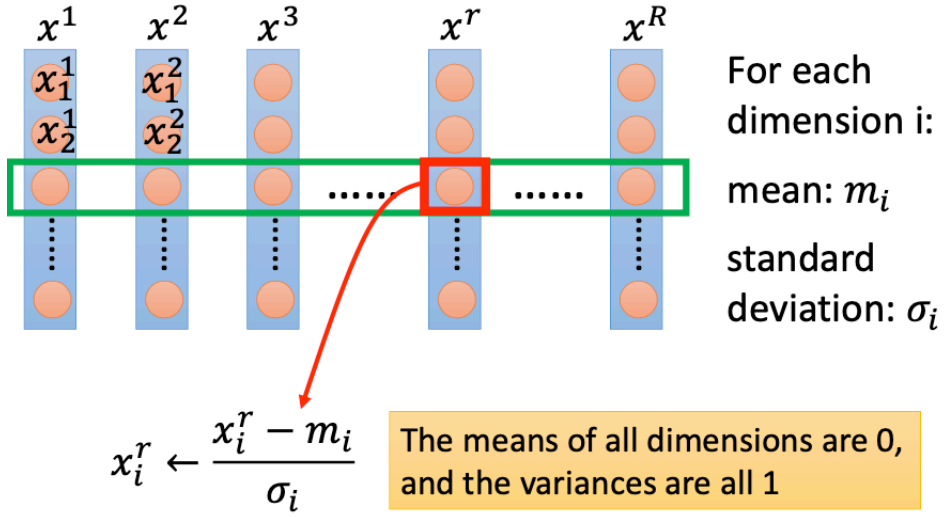


x2对应w2，x1对应w1，对于同一个w1、w2，但x2的数值不同

在左图中，由于x1的数值相对于x2来说都很小，x1的变化对于y来说影响很小，w1对y的影响也很小，对loss的影响也小，因此梯度$\frac{\partial L}{\partial w_1}$在w1方向的变换也比较平缓；x2的数值较大，对loss的影响也大，因此梯度$\frac{\partial L}{\partial w_2}$在w2方向的变换就比较sharp

在右图中，x1和x2的规模（scale）是接近的，对y的影响不相上下，对loss的影响也差不多

计算



计算方式：用m_i表示当前样本的平均值，σ_i为当前样本的标准差，i表示维度，x_i^r表示第r个example，使用公式$x_i^r \leftarrow \frac{x_i^r - m_i}{\sigma_i}$来进行归一化计算

feature scaling其实就是将每一个example都进行归一化，使之服从标准正态分布$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$

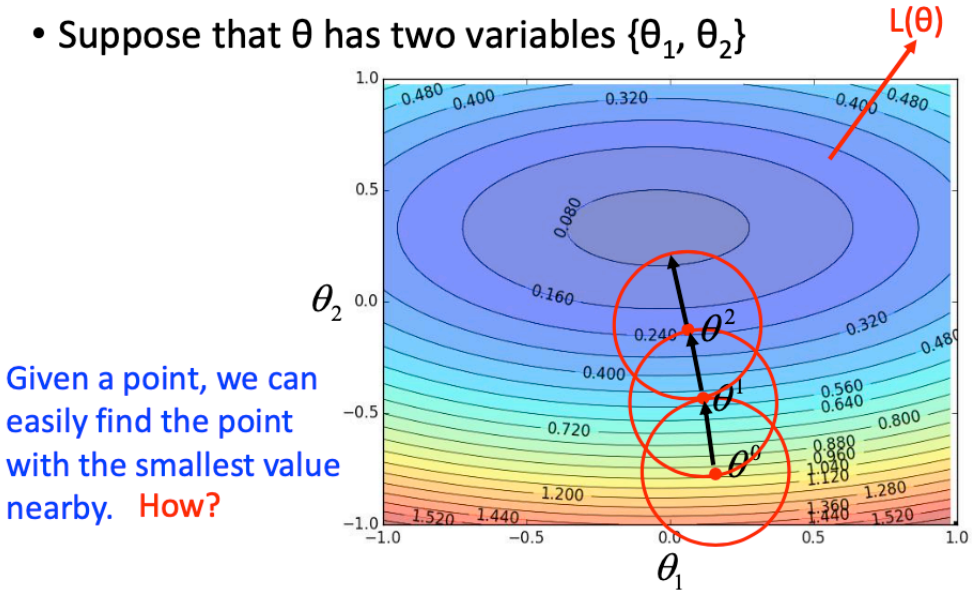
$$\frac{X - \bar{X}}{\sqrt{D(x)}} \sim N(0,1)$$

Gradient Descent Theory

Question

在求解最小化问题时，$\theta^* = arg\ min\ L(\theta)$，每次更新$\theta$的值，并不一定能使$L(\theta^0) > L(\theta^1) > L(\theta^2) > \cdots$成立

对于给出的θ^1, θ^2，我们要如何根据这些值来找出最小的loss？这也是我们接下来会研究的问题



Talor Series

泰勒公式定义如下，将函数h(x)在x=x0处展开

$$h(x) = \sum_{k=0}^{\infty} \frac{h^{(k)}(x_0)}{k!}(x - x_0)^k$$

当$x \to x_0$时，表达式可写为$h(x) \approx h(x_0) + h'(x_0)(x - x_0)$

对于二元函数，当$x \to x_0, y \to y_0$时，相应的表达式可以简化为

$$h(x, y) \approx h(x_0, y_0) + \frac{\partial h(x_0, y_0)}{\partial x}(x - x_0) + \frac{\partial h(x_0, y_0)}{\partial y}(y - y_0)$$

Back to Formal Derivation

损失函数loss可以用以下公式表示，

$$L(\theta) = L(a, b) + \frac{\partial L(a, b)}{\partial \theta_1}(\theta_1 - a) + \frac{\partial L(a, b)}{\partial \theta_2}(\theta_2 - b)$$

简化表达形式，令$s = L(a, b)$, $u = \frac{\partial L(a,b)}{\partial \theta_1}$, $v = \frac{\partial L(a,b)}{\partial \theta_2}$

则$L(\theta) \approx s + u(\theta_1 - a) + v(\theta_2 - b)$

如下图所示，在图中red circle的范围内，找到θ_1, θ_2，使得loss最小化，设red circle的半径为d，圆心坐标为(a,b)，就新增了一个限制条件$(\theta_1 - a)^2 + (\theta_2 - b)^2 \leq d^2$

Based on Taylor Series:

If the red circle is ***small enough***, in the red circle

constant

$$s = L(a,b)$$

$$u = \frac{\partial L(a,b)}{\partial \theta_1}, v = \frac{\partial L(a,b)}{\partial \theta_2}$$

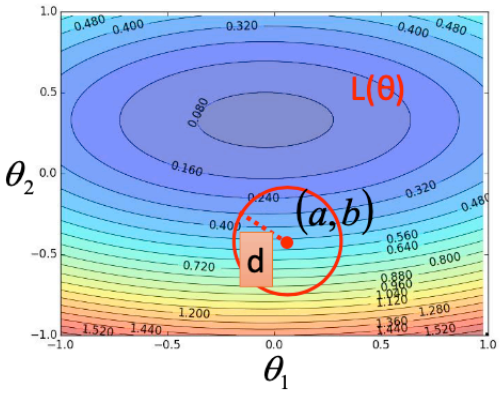
$$L(\theta) \approx s + u(\theta_1 - a) + v(\theta_2 - b)$$

Find θ_1 and θ_2 in the red circle

minimizing L(θ)

$$(\theta_1 - a)^2 + (\theta_2 - b)^2 \le d^2$$

Simple, right?



由于s与θ_1, θ_2不相关，这里把s项去掉，$L(\theta)$可以转为内积

$$\begin{aligned} L(\theta)_{min} &\approx u(\theta_1 - a) + v(\theta_2 - b) \\ &= (u, v) \cdot (\theta_1 - a, \theta_2 - b) \\ &= (u, v) \cdot (\Delta\theta_1, \Delta\theta_2) \end{aligned}$$

当$(\Delta\theta_1, \Delta\theta_2)$与$(u, v)$方向相反时，两者的内积为最小值，由于两者的模长不同，用参数η来表示两者之间的关系。

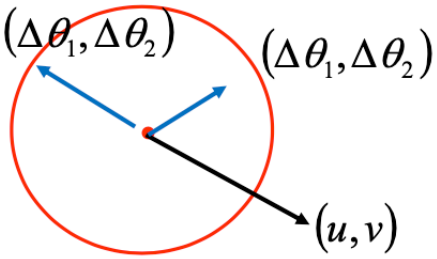
Red Circle: (If the radius is small)

$$L(\theta) \approx \cancel{s} + u\underbrace{(\theta_1 - a)}_{\Delta\theta_1} + v\underbrace{(\theta_2 - b)}_{\Delta\theta_2}$$

Find θ_1 and θ_2 in the red circle

minimizing L(θ)

$$\underbrace{(\theta_1 - a)}_{\Delta\theta_1}^2 + \underbrace{(\theta_2 - b)}_{\Delta\theta_2}^2 \le d^2$$



To minimize L(θ)

$$\begin{bmatrix} \Delta\theta_1 \\ \Delta\theta_2 \end{bmatrix} = -\eta \begin{bmatrix} u \\ v \end{bmatrix} \Rightarrow \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} a \\ b \end{bmatrix} - \eta \begin{bmatrix} u \\ v \end{bmatrix}$$

由于$\Delta\theta_1 = \theta_1 - a, \Delta\theta_2 = \theta_2 - b$，则$\theta_1 = a + \Delta\theta_1, \theta_2 = b + \Delta\theta_2$，可得出

$$\begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} a \\ b \end{bmatrix} - \eta \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} a \\ b \end{bmatrix} - \eta \begin{bmatrix} \frac{\partial L(a,b)}{\partial \theta_1} \\ \frac{\partial L(a,b)}{\partial \theta_2} \end{bmatrix}$$

Limitation

在真实的实验环境中，我们往往会设置一个临界值（比如10^{-4}），当该点的梯度小于该值（即≈ 0）时，就停止训练。

因此，gradient descent的限制是，gradient为0的点并不一定是local minimum，还有可能是saddle point，也有可能是接近于0的点