GD(梯度下降)和SGD(随机梯度下降)

相同点

在GD和SGD中，都会在每次迭代中更新模型的参数，使得代价函数变小。

不同点

GD

在GD中，每次迭代都要用到**全部**训练数据。 假设线性模型

$$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x,$$

Paste_Image.png

θ是参数

代价函数 ：

$$J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2.$$

Paste_Image.png

那么每次GD的更新算法为：

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta).$$

Paste_Image.png

由此算法可知，在对代价函数求偏导时，是需要用到全部的训练数据的。

SGD

在SGD中，每次迭代可以只用**一个**训练数据来更新参数。 回到GD的更新算法，假设此时我们此时训练数据就只有一条(x,y)，

$$
\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2} \left(h_\theta(x) - y\right)^2 \\
&= 2 \cdot \frac{1}{2} \left(h_\theta(x) - y\right) \cdot \frac{\partial}{\partial \theta_j} \left(h_\theta(x) - y\right) \\
&= \left(h_\theta(x) - y\right) \cdot \frac{\partial}{\partial \theta_j} \left(\sum_{i=0}^{n} \theta_i x_i - y \right) \\
&= \left(h_\theta(x) - y\right) x_j
\end{aligned}
$$

Paste_Image.png

所以此时的更新参数的算法变为：

$$\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)}) \right) x_j^{(i)}.$$

Paste_Image.png

此时更新的算法，只用到了一个样本。 其实具象的理解下，就是来了一条训练数据，算下此时根据模型算出的值和实际值的差距，如果差距大，那么参数更新的幅度大，反之则小。

总结

当训练数据过大时，用GD可能造成内存不够用，那么就可以用SGD了，SGD其实可以算作是一种online-learning。另外SGD收敛会比GD快，但是对于代价函数求最小值还是GD做的比较好，不过SGD也够用了。