1、模型假设

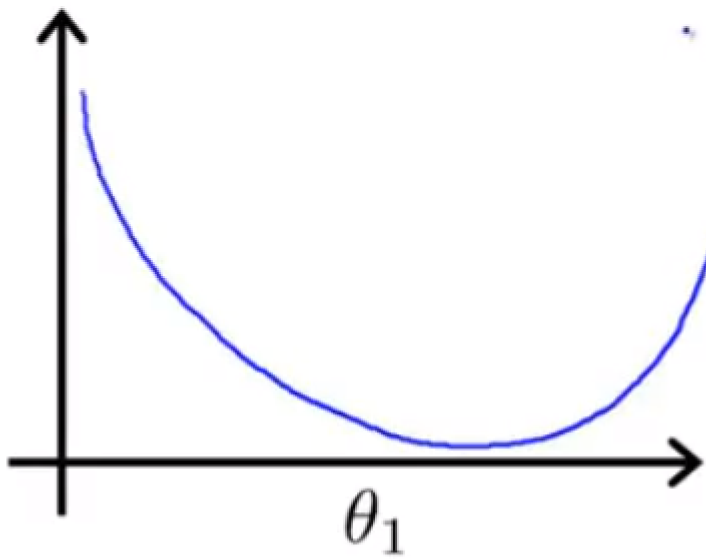
根据房屋面积预测其房价，首先我会给你一些样本，这些样本包含了房屋面积和房价的一个个数据对，我们可以写成（y，x）的形式，其中 y 代表房屋价格，x 代表房屋面积，这样一来，我们便可根据它们之间的某种关系，得到一个方程 y=h(x)，它包含的元数是指结果可能受几种因素影响，而次数以及常数项，是要有编程人员的经验进行调参的。这个由我们假设出来的模型现在似乎可以拿来预测其他数据了，也就是测试数据。

2、代价函数

不过，这个 h(x)可能无法将所有的已知数据拟合得那么完美，总是有误差的，正如模型假设中的函数$h_\theta(x) = \theta_0 + \theta_1 x$我们不能让参数的取值正好使假设经过所有的点，但可以不断精确，确保它已经很接近真实值了，怎样评判假设拟合得好不好，这里给出代价函数的定义$F(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y(i)\right)^2$，求最小值$min_{\theta_0\theta_1}F(\theta_0, \theta_1)$成了我们唯一的目标。

3、梯度下降算法

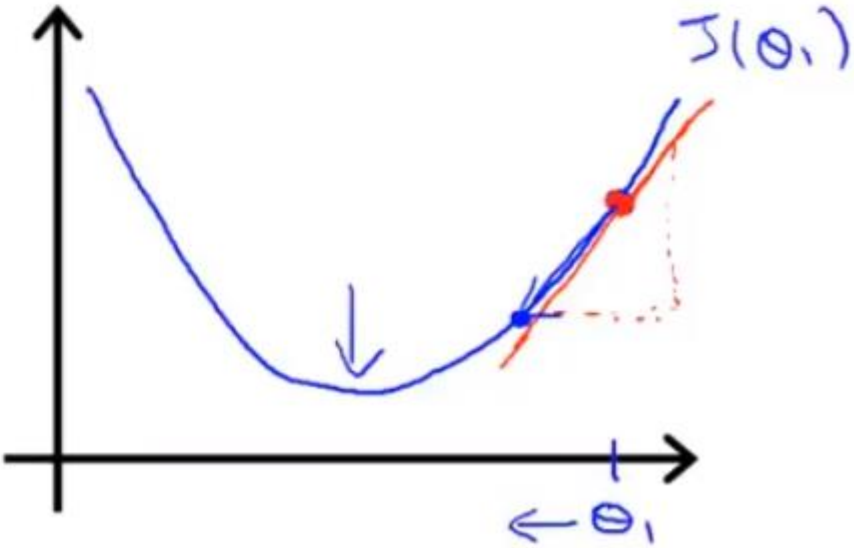
梯度下降算法可以帮助我们找到这样一个局部最优解，确保一定范围内，我们的假设函数是最优的，为简单起见，我们定义$h_\theta(x) = \theta_1 x$，这样一来，我们得到的代价函数图像可能是这样的。



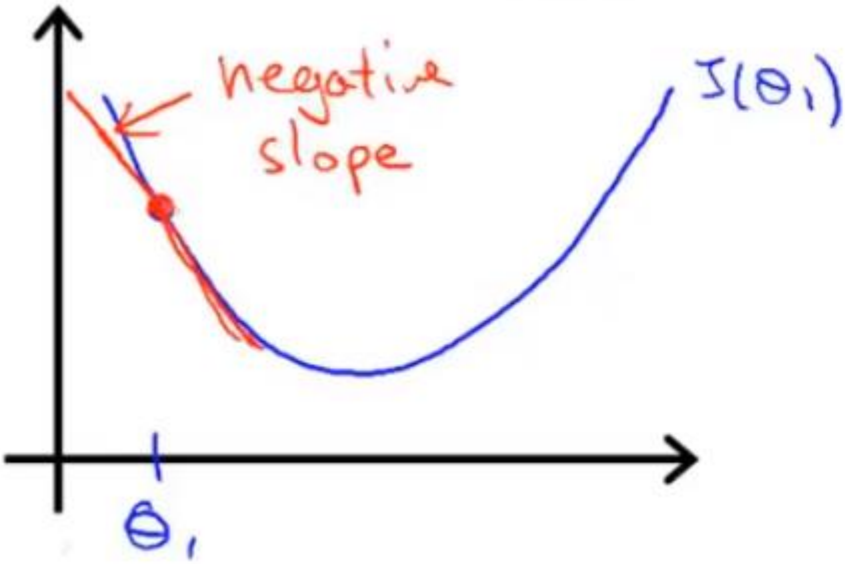
单一参数θ_1对代价函数影响曲线

假如我们事先不知道究竟什么样的值对于θ_1才是合理的，因此我们鲁莽的选择了一个位于图像右侧的点，可以看到，离最小的代价函数点还有些距离，那怎样知道我们应当减小θ_1的值而不是增大呢，是时候介绍我们的梯度下降算法了，给定公式$\theta_1 = \theta_1 - \alpha\frac{d_{J(\theta)}}{d\theta_1}$，当点位于右侧时，我们得到$J_{(\theta)}$关于$\theta_1$的导数为正，因此当我们$\theta_1 -$

$\alpha\frac{d_{J(\theta)}}{d\theta_1}$时，得到的值必然比$\theta_1$小（约定$\alpha$永远为正），可以看到，随着$\theta_1$减小了，我们的代价函数值$J_{(\theta)}$也相应的减小了。



针对另一种情况，假设我们选择的初值θ_1位于图像左侧，像这样，我们同样可以由$\theta_1-\alpha\frac{d_{J(\theta)}}{d\theta_1}$使得新的$\theta_1$值比初值大，$J_{(\theta)}$也减小了。



就目前局势来看，我们的模型假设似乎在朝着正确的方向走，这里还应说明几点值得注意的地方，首先是α学习速率，之前的例子中，如果我们选用过小的α作为学习速率的话，我们的代价函数可能需要很长一段时间才能趋近于正确的值，如果选择过大的α，又会使得步长过大，从一边跳到另一边去，更糟糕的是，出现步长加大的情况会使得代价函数永远无法局部最优。

还需注意的一点是，如果有多个参数需要改变的话，需要做到同步更新，像是这样。

$$\begin{aligned} \text{temp0} &:= \theta_0 - \alpha * \frac{\partial}{\partial \theta_0} \mathrm{F}(\theta_0, \theta_1) \\ \text{temp1} &:= \theta_1 - \alpha * \frac{\partial}{\partial \theta_1} \mathrm{F}(\theta_0, \theta_1) \\ \theta_0 &:= \text{temp0} \\ \theta_1 &:= \text{temp1} \end{aligned}$$

4、数学原理

之前讲的代价函数中只包含θ_1一个参数，所以相对应的偏导数也就降级成了求导数；偏导数是指二元关系下，函数值随两个参数的变化规律，我们关注的对象也仅是切线分别垂直于 y 轴和 x 轴的两条切线；关于方向导数，就应考虑任意方向上的切线了，切线沿着梯度这个方向，下降的是最快的，梯度下降算法也正是利用了这个原理。

5、模型运用

运用梯度下降算法，我们能够得到最小代价函数，第二节我们讲到过，$F(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y(i)\right)^2$是个连续函数，因此得到以下方程组：

$$\frac{\partial}{\partial \theta_j} \mathrm{F}(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_j} \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

$$\begin{aligned} \frac{\partial}{\partial \theta_0} \mathrm{F}(\theta_0, \theta_1) &= \frac{1}{m} \sum_{i=1}^{m} u \frac{\partial u}{\partial \theta_0} \\ &= \frac{1}{m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x^{(i)} - y^{(i)}) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \end{aligned}$$

$$\begin{aligned} \frac{\partial}{\partial \theta_1} \mathrm{F}(\theta_0, \theta_1) &= \frac{1}{m} \sum_{i=1}^{m} u \frac{\partial u}{\partial \theta_1} \\ &= \frac{1}{m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x^{(i)} - y^{(i)}) * x^{(i)} = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) * x^{(i)} \end{aligned}$$

再利用第三节中同步更新的原则，我们可以依次迭代出新的参数值。由于每次迭代，需要计算该点偏导数，也就是需要求所有训练集的平方代价差，因此又称 Batch 算法。

6、前瞻

除了梯度下降算法，还可以利用正规方程组法。但是两者相比较而言，梯度下降适合更大的数据集，后者不需要遍历所有训练集。