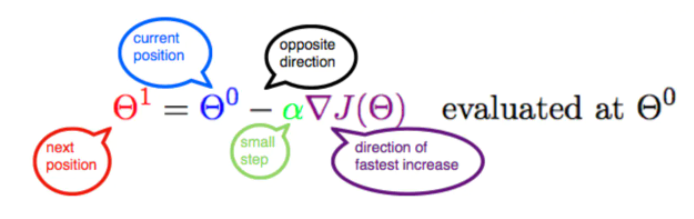
梯度下降法及其实现实验报告

实验人：杜先瑞

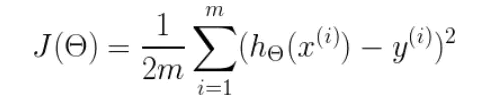
1. **实验原理**

可以将梯度下降与下山的过程做类比，梯度下降就是在下山的过程中不断寻找坡度最陡峭的地方，从而实现找到这座山最底部的地方。在下降过程中如果出现方向偏差需要随时调整方向。为了实现方向的调整，我们可以设置一个步长，每次只走这一个步长的距离，然后重新判断方向，从而实现每次走的方向都是下降速度最快的方向。在数学中，我们把函数比作一个高山，将寻找函数最低点的过程比作走到山的最底部。我们知道，一个函数的梯度方向是函数变化率最大的方向，因此我们可以通过求函数微分，找到梯度方向，也就是找到了函数变化最快的方向。然后通过每次走一个步长，走完一个步长重新求梯度方向，经过多次迭代找到最低点。

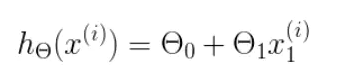
1. **实验过程**
2. 了解梯度下降的基本原理及数学意义，学会如何通过梯度下降求一个函数的局部最小值。



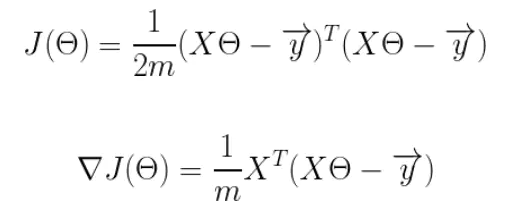
1. 学习代价函数的意义，学习通过均方误差代价函数进行线性拟合的过程。通过梯度下降法求代价函数的最小值，使得代价函数最小值时的参数θ即为目标函数的最合适参数。



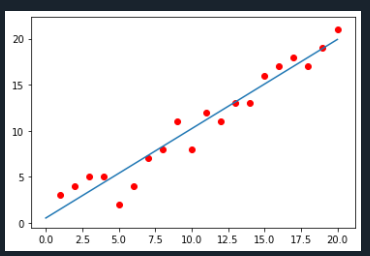
其中h(x)为预测函数



1. 为了方便使用python实现以上线性回归算法，可以将以上公式转化为矩阵形式。具体为给每一个x增加一维，这一维的值全部为1。这样，代价函数和梯度变成如下所示



1. 进行编码。编码实现代价函数及梯度函数，实现梯度下降算法，求出最小值以及参数θ。最后，还可以将结果可视化。



1. **实验结果**

参数=0.01，theta = np.array([1,2]).reshape(2,1) 时，得出结果如下



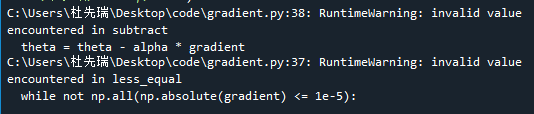
参数=0.005，theta = np.array([1,2]).reshape(2,1) 时，得出结果如下



参数=0.01，theta = np.array([2,2]).reshape(2,1) 时，得出结果如下



参数=0.02时程序报错



从以上结果可以看出，调节参数对于结果影响不大。

1. **实验中遇到的问题**

问题一：为什么可以通过将x增加一维固定值1来使得公式转化成矩阵形式？

给x增加一维值为1的列后与θ相乘并不会改变结果，但是为什么需要这样操作还是不太理解。

问题二：为什么选择均方误差代价函数？

我的理解为均方误差代价函数可以代表预测值与真实值之间的差距，均方误差越小，拟合效果越好。我觉得应该也可以选择其他代价函数。

问题三：为什么=0.02时程序无法运行？

目前还没有找到原因

问题四：为什么后面变成矩阵形式后参数θ只有一个了？为什么传入的参数θ和最后求出的最优θ有两个值？

目前还没有找到原因

1. **总结与收获**

通过这次实验学习到了梯度下降的基本原理，还学习了如何使用梯度下降算法来进行简单的线性拟合。另外，学习了代价函数的意义。虽然学到了一些东西，但是又出现了很多新的问题，比如梯度下降在其他方面有些什么用途？比如代价函数在其他地方代表什么意义？另外，实验过程中也遇到了几个无法解决的问题，这些都需要后面继续学习。