实验环境

PyCharm-162.1237.1

TextMate , python 2,7

一. 实验题目

考虑将梯度下降（算法一）和牛顿法（算法二）应用到表中的数据上。

(a) 用这两种算法对二维数据给出$\omega1$和$\omega3$的判别。对梯度下降法取η(k) =0.1。画出以迭代次数为准则函数的曲线。

(b) 估计这两种方法的数学运算量。

(c) 画出收敛时间-学习率曲线。求出无法收敛的最小学习率。

二. 实验数据

样本	W1		W3	
	X1	X2	X1	X2
1	0.1	1.1	-3.0	-2.9
2	6.8	7.1	0.5	8.7
3	-3.5	-4.1	2.9	2.1
4	2.0	2.7	-0.1	5.2
5	4.1	2.8	-4.0	2.2
6	3.1	5.0	-1.3	3.7
7	-0.8	-1.3	-3.4	6.2
8	0.9	1.2	-4.1	3.4
9	5.0	6.4	-5.1	1.6
10	3.9	4.0	1.9	5.1

三. 实验过程

【问题一】我们要用到梯度下降法和牛顿法两种方法。

（1） 梯度下降法：

先初始化一个 a，然后通过梯度对 a 不间断的进行修改，一直修改直到到达阈值θ，算法停止，其中η(k)是学习率，是由我们人为设定的

算法 1 （基本梯度下降法）

begin **initialize** **a**，阈值 θ，$\eta(\cdot)$，$k \leftarrow 0$

　　do $k \leftarrow k+1$

　　　$\mathbf{a} \leftarrow \mathbf{a} - \eta(k)\nabla J(\mathbf{a})$

　　until $|\eta(k)\nabla J(\mathbf{a})| < \theta$

return a

end

按照书上公式的方法，我们需要选择一个准则函数，我选择的准则函数如下 （乘上 1/2 方便求导） ：

$$J(\alpha) = \frac{1}{2}\sum_{y\epsilon\Upsilon}\frac{(a^Ty-b)^2}{||y||^2}$$

对上述公式求导，得到梯度公式，如下:

$$\nabla J = \sum_{y\epsilon\Upsilon}\frac{a^Ty-b}{||y||^2}y$$

其中集合γ表示$aTy - b \leq 0$的点，也就是分错了的点。同时我令 b=0。这 样就是当$aTy > 0$表示分对了，当$aTy \leq 0$表示分错了。而我们的目标就 是使得 J(α)最小。

（2） 牛顿法:

牛顿法应该是由梯度下降法演化而来的，它不需要我们人为设定学习率，但是需要我们计算出 a 相应的海森矩阵，同时，它的停止条件是$|H-1\ \nabla J| < \theta$，关键在于海森矩阵。

算法 2 （牛顿下降法）

begin **initialize** a，阈值 θ

　　do

　　　$\mathbf{a} \leftarrow \mathbf{a} - \mathbf{H}^{-1}\nabla J(\mathbf{a})$

　　until $|\mathbf{H}^{-1}\nabla J(\mathbf{a})| < \theta$

　return a

end

海森矩阵（Hessian）是一个多变量实值函数的二阶偏导数组成的方块矩阵，根据定义我们可以简单的求出海森矩阵的表达式如下:

$$H = \sum_{y\epsilon\Upsilon} y * y^T$$

（3） 实验结果

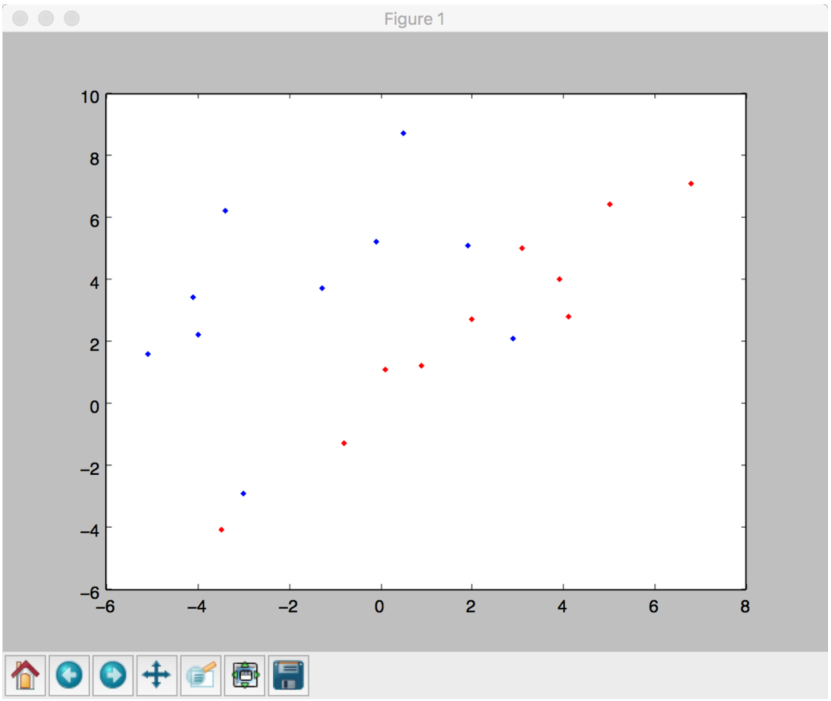


图 1 各点分布图

从图一我们可以看出，这两类点事是线性不可分的

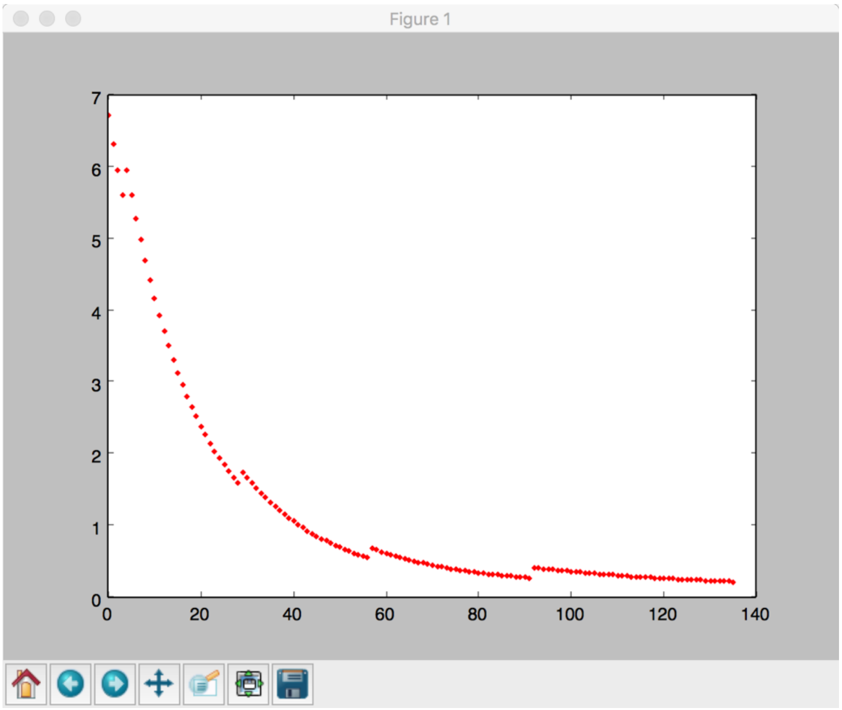


图 2 基本梯度下降法（以迭代次数为准则函数的曲线）

图二时，我发现如果按照题目要求，选择 η (k)=0.1 的话，超过了收敛的最大学习率，会导致梯度下降法不收敛，所以我选择了 η (k)=0.001，同时，经过多次实验，我把阈值设为 0.002。

从图上可以看到，用梯度下降法差不多迭代了 140 次

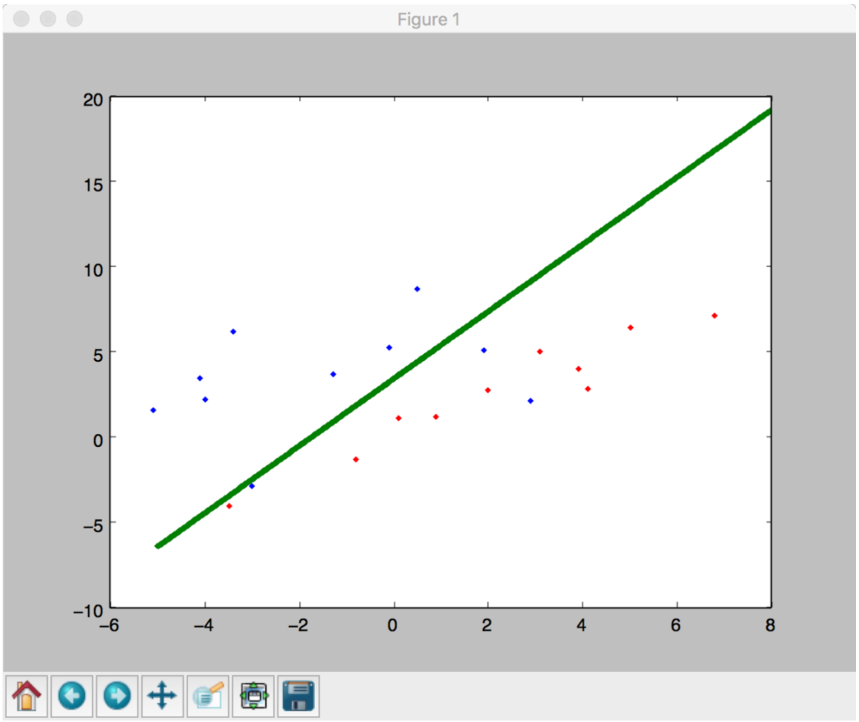


图 3 用基本梯度下降法求出的分界线

通过画出分界线我们看到，通过梯度下降法分错了 3 个点

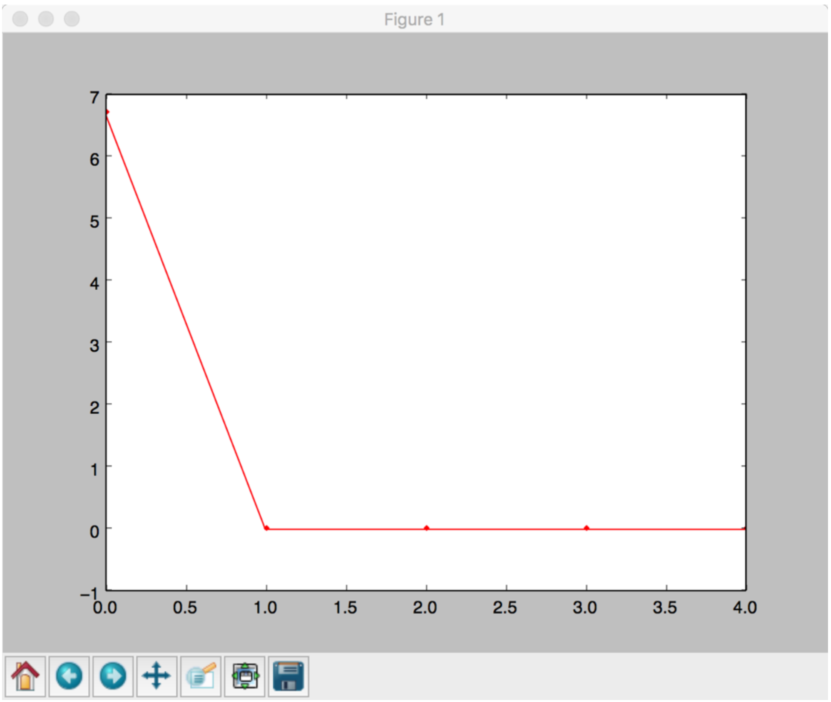


图 4 牛顿法（以迭代次数为准则函数的曲线）

图 4 是牛顿法的曲线，我取了 limit = 1e-56 为阈值。由于最后的几个点都离 0 很近，画出来的点在 x 坐标轴上不容易看清，我在最后加了一个点(4.0，-0.001)，使得 x 轴下移到-1，由此可以看清。

我们可以看到，相比于基本梯度下降法，牛顿法收敛的更快，只收敛了四次就好了

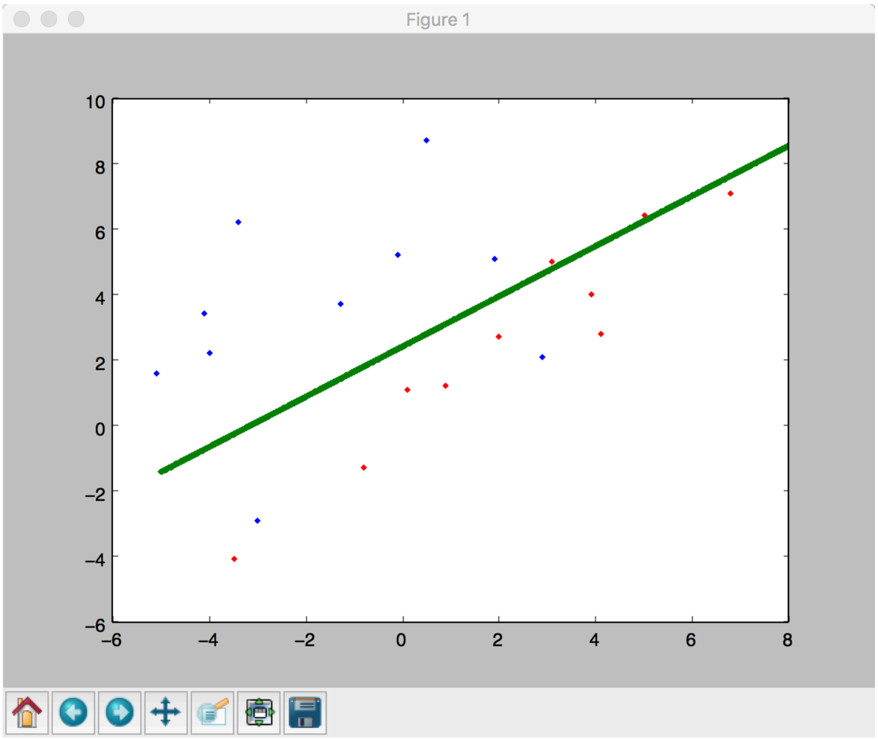


图 5 用牛顿法法求出的分界线

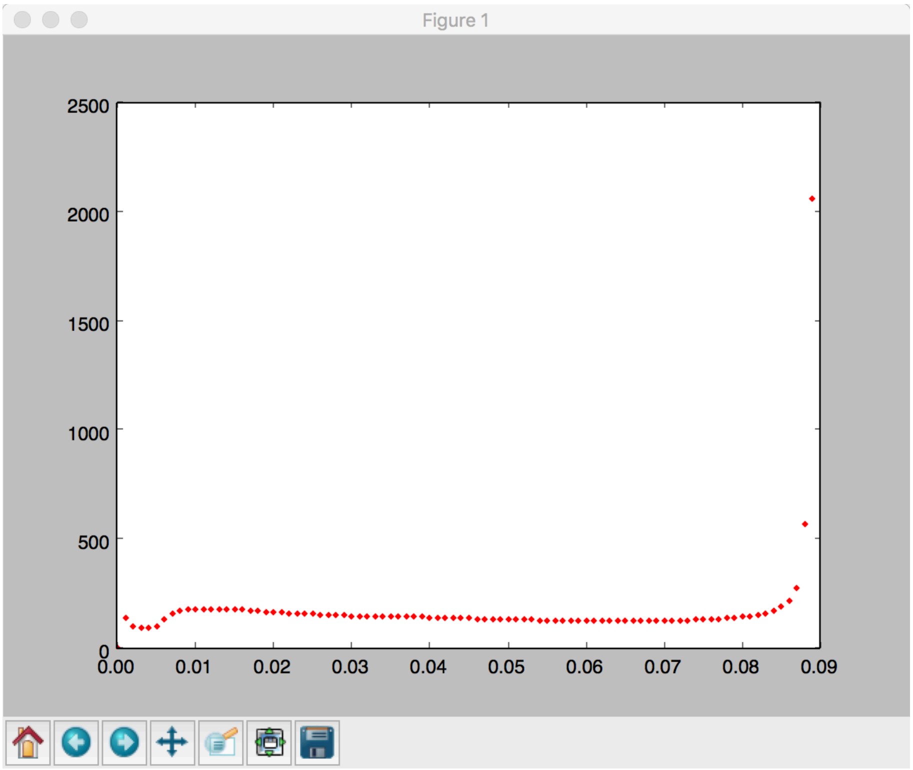
通过画出分界线我们看到，通过梯度下降法分错了 4 个点

【问题二】估计这两种方法的数学运算量

首先对于梯度下降法,我们都是需要计算一个每一步的梯度,而计算梯度的运算量为 $O(d^2)$,但是，由于我们的 η (k)的选择很大，可能会导致，这个永远不收敛的情况,当然也有可能因为我们的 η (k)选 x 取的很小,又会导致收敛的很慢的情况。但是对于我们的牛顿下降法而言，我们每一步计算得到的都是最好的 η (k)的值,但是我们计算何森矩阵的时间复杂度为 $O(d^3)$可能会被抵消，但是总体上看，一定是牛顿法的时间复杂度更低，数学运算量要更小一些。从我们一会的结果也可以看出，牛顿法的收敛更快。但是梯度下降要收敛可能就需要上百次甚至不收敛

【问题三】画出收敛时间-学习率曲线

对于第三问，我们只需要画出一个以迭代次数为 y 轴，然后以学习为 x 轴的曲线，通过观察图像何时 y 的值为无穷大,那么就表示此时已经超过了最大学习率。找到最小的那个 x,也就是无法收敛的最小学习率。可以看到，基本上在 0.085 的地方就已经梯度下降法不收敛了。



图六. 收敛时间-学习率曲线

四. 收获与感悟

a) 经过这次实验首先我更加熟悉了 python 的使用

b) 我了解了梯度下降法的使用

c) 我了解了牛顿法的使用

d) 我对梯度下降法和牛顿法的区别和优劣有了更深的理解

五. 代码

位于同项目路径的文件夹中