哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

2020 年春季学期

计算学部《机器学习》课程

Lab 2 实验报告

logistic 回归

姓名	李卓君
学号	1180300210
班号	1803104
电子邮件	zhuojunlee724@gmail.com
手机号码	18845636557

目录

1 实验目的与实验要求

1.1 实验目的

理解逻辑回归模型，掌握逻辑回归模型的参数估计算法。

1.2 实验要求

实现两种损失函数的参数估计（1，无惩罚项；2.加入对参数的惩罚），可以采用梯度下降、共轭梯度或者牛顿法等。

1.3 实验环境

Windows 10, Visual Studio Code, Python 3.8.4

2 实验背景与原理

2.1 实验背景

实验从分类问题出发，旨在用朴素贝叶斯方法作为理论指导解决二分类问题。以平面上的二维点集为例，对于横纵坐标均符合高斯分布的两类点，我们希望划分出一条边界可以尽可能的区分这两类点使得定义的代价函数最小，且当这两类点对应维度的方差相等时可以证明这条边界恰为低一维度的超平面。

2.2 实验原理

假定 X 是一个 n 维实值列向量$(x_1,x_2,...,x_n)$，Y 是一个满足伯努利分布的布尔值，令 $P(Y=1)=\pi$，目标函数 $f:X\rightarrow Y$。若对于给定的 Y，X 各维度的变量相互独立，$P(X_i|Y=y_k)$ 满足高斯分布 $N(\mu_{ik},\sigma_i)$，根据高斯判别分析方法，我们根据先验概率 $P(Y)$和似然函数 $P(X|Y)$来计算后验分布 $P(Y|X)$。

以 $P(Y=1|X)$的计算为例，若满足朴素贝叶斯假设：

$$P(Y=1|X)=\frac{P(Y=1)P(X|Y=1)}{P(Y=1)P(X|Y=1)+P(Y=0)P(X|Y=0)}$$

$$=\frac{1}{1+\frac{P(Y=0)P(X|Y=0)}{P(Y=1)P(X|Y=1)}}$$

$$= \frac{1}{1+\exp\left(\ln\frac{\mathrm{P(Y=0)P(X|Y=0)}}{\mathrm{P(Y=1)P(X|Y=1)}}\right)}$$

$$= \frac{1}{1+\exp\left(\ln\frac{1-\pi}{\pi}+\sum_i ln\frac{P(X_i|Y=0)}{P(X_i|Y=1)}\right)}$$

假设 $P(X_i|Y=y_k)$满足一维高斯分布 $N(\mu_{ik},\sigma_i)$，所以有其概率分布如下：

$$\mathrm{P(X_i|Y=y_k)} = \frac{1}{\sigma_{ik}\sqrt{2\pi}}\exp\left(\frac{-(\mathrm{x}-\mu_{ik})^2}{2\sigma_{ik}^2}\right)$$

则有

$$\frac{\mathrm{P(X_i|Y=0)}}{\mathrm{P(X_i|Y=1)}} = \frac{\frac{1}{\sigma_{i0}\sqrt{2\pi}}\exp\left(\frac{-(\mathrm{x}-\mu_{i0})^2}{2\sigma_{i0}^2}\right)}{\frac{1}{\sigma_{i1}\sqrt{2\pi}}\exp\left(\frac{-(\mathrm{x}-\mu_{i1})^2}{2\sigma_{i1}^2}\right)}$$

如前所述，令 X 对应维度的方差相等，则有

$$\frac{\mathrm{P(X_i|Y=0)}}{\mathrm{P(X_i|Y=1)}} = \exp\left(\frac{(\mathrm{x}-\mu_{i1})^2-(\mathrm{x}-\mu_{i0})^2}{2\sigma_i^2}\right)$$

$$= \exp\left(\frac{2\mu_{i0}\mathrm{x}-2\mu_{i1}\mathrm{x}-\mu_{i0}^2+\mu_{i1}^2}{2\sigma_i^2}\right)$$

所以有

$$\mathrm{P(Y=1|X)} = \frac{1}{1+\exp\left(\ln\frac{1-\pi}{\pi}+\sum_i\left(\frac{\mu_{i0}-\mu_{i1}}{{\sigma_i}^2}X_i+\frac{{\mu_{i1}}^2-{\mu_{i0}}^2}{2\sigma_i^2}\right)\right)}$$

将常数项合并为$\mathrm{w_0}$，$\mathrm{X_i}$系数设为$\mathrm{w_i}$，于是有

$$\mathrm{P(Y=1|X)} = \frac{1}{1+\exp\left(\mathrm{w_0}+\sum_{i=1}^{n} w_iX_i\right)}$$

X 满足多维高斯分布时可以进行推广，并得到以下结果

$$\mathrm{P(Y=1|X)} = \frac{1}{1+\exp(-\mathrm{a})}$$

$$\mathrm{a} = (\mu_0-\mu_1)^{\mathrm{T}}\Sigma^{-1}\mathrm{x}+\ln\frac{\mathrm{P(Y=0)}}{\mathrm{P(Y=1)}}+\frac{1}{2}\left[{\mu_1}^{\mathrm{T}}\Sigma^{-1}\mu_1-{\mu_0}^{\mathrm{T}}\Sigma^{-1}\mu_0\right]$$

其中Σ是共用的协方差矩阵，μ_0,μ_1分别为两者的期望向量。

a 是关于 x 是线性的，说明了在朴素贝叶斯假设的前提下进行高斯判别分析，对于 n 维的数据进行二分类，我们可以得到一条 n-1 维的超平面作为边界，例如，对于平面上的点集我们可以得到一条直线，对于空间中的点集我们则可以得到一个平面对其进行划分。

接下来考虑关于参数θ的代价函数 $\mathrm{L(\theta)}$，因为 a 是线性的，所以可以用向量的乘积 Xθ 来表示，后验分布函数就可以表示为一个关于矩阵 X 和参数θ的函数：

$$\mathrm{h_\theta(X)} = \frac{1}{1+\exp(-\mathrm{X\theta})}$$

由后验分布的定义知，这个函数实际上表示了满足了 X 的特征下为正例的概率分布，由概率定义知，反例的概率分布为 $1-\mathrm{h_\theta(x)}$，由此，我们将二者统一化，有

$$\mathrm{P(Y|X)} = h_\theta(X)^Y(1-\mathrm{h_\theta(x)}^{1-Y})$$

对于 m 组满足条件独立分布的例子进行最大似然估计有

$$L(\theta)=\prod_{i=1}^{m}P(Y|X)=\prod_{i=1}^{m}h_{\theta_i}(\mathrm{X_i})^{\mathrm{Y_i}}\left(1-h_{\theta_i}(X_i)^{1-Y_i}\right)$$

对 $\mathrm{L}(\theta)$取最小值时的θ即是我们所需的θ，考虑到这样计算不方便，我们对代价函数取一个负对数进行计算：

$$\mathrm{l}(\theta)=-\log\left(\prod_{i=1}^{m}h_{\theta_i}(\mathrm{X_i})^{\mathrm{Y_i}}\left(1-h_{\theta_i}(X_i)^{1-Y_i}\right)\right)$$

$$=-\mathrm{y}\log(\mathrm{h}_\theta(\mathrm{x}))-(1-\mathrm{y})\log(1-\mathrm{h}_\theta(\mathrm{x}))$$

实际计算时往往还要对于样本数量取一个均值，所以有最终的代价函数如下

$$\mathrm{J}(\theta)=\frac{1}{\mathrm{m}}\times(-\mathrm{y}^{\mathrm{T}}\log(\mathrm{h})-(1-\mathrm{y})^{\mathrm{T}}\log(1-\mathrm{h}))$$

其梯度为

$$\mathrm{grad}=\frac{1}{\mathrm{m}}\mathrm{X}^{\mathrm{T}}(\mathrm{h}_\theta(\mathrm{X})-\mathrm{y})$$

3 logistic 回归

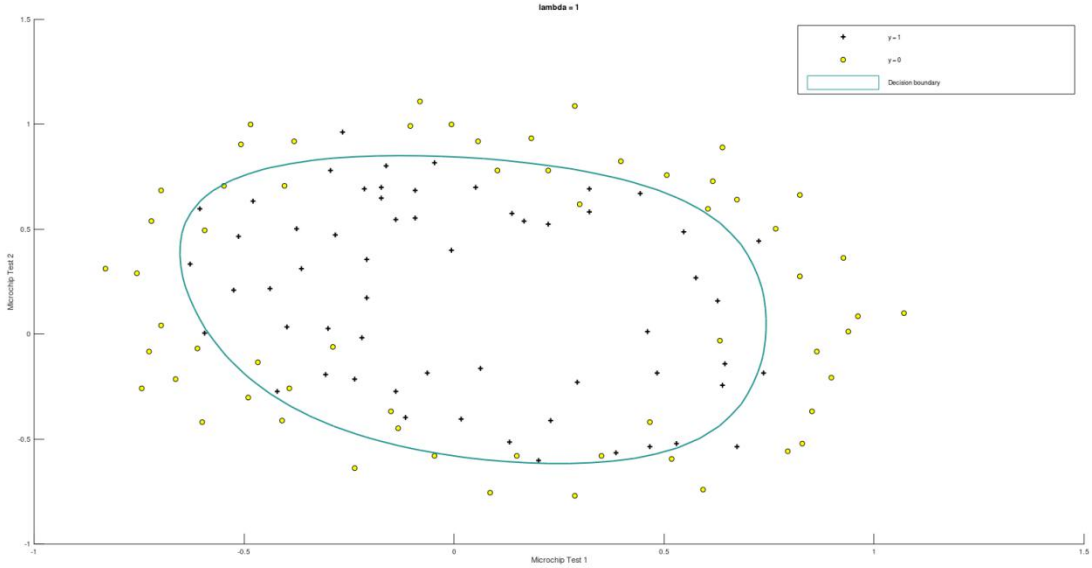
3.1 生成数据集

利用 numpy 库中的 numpy.random.multivariate_normal()函数来生成多维高斯分布的数据，为了便于以图表的形式表现，我在这里采用了二维高斯分布，设定期望向量和协方差矩阵等，进行传参会返回一个 $\mathrm{m}\times 2$ 的向量，其中 m 表示生成样本个数。

生成协方差矩阵时需要注意，根据概率论的知识，当协方差矩阵非主对角线上的数为 0 时各维度满足独立分布条件，也就是满足贝叶斯假设，否则不满足。

3.2 代价函数

在上一小节中已经对于代价函数的方程进行了推导，增加惩罚项的方式与线性回归如出一辙，实际上，由于拟合出来的边界函数是一条直线(因为特征较少)，所以从直觉上判断，出现过拟合的可能几乎没有，而对于满足朴素贝叶斯的高斯分布的样本集，在上一节中也已经证明了边界函数为一条直线，但实际情况中数据不一定满足朴素贝叶斯假设和高斯分布，也就不能直接地用直线去拟合，这时用曲线拟合的情况下，加入惩罚项就十分有必要了，在吴恩达的课程实验中要求选做了这种情况下的数据的 logistic 回归。

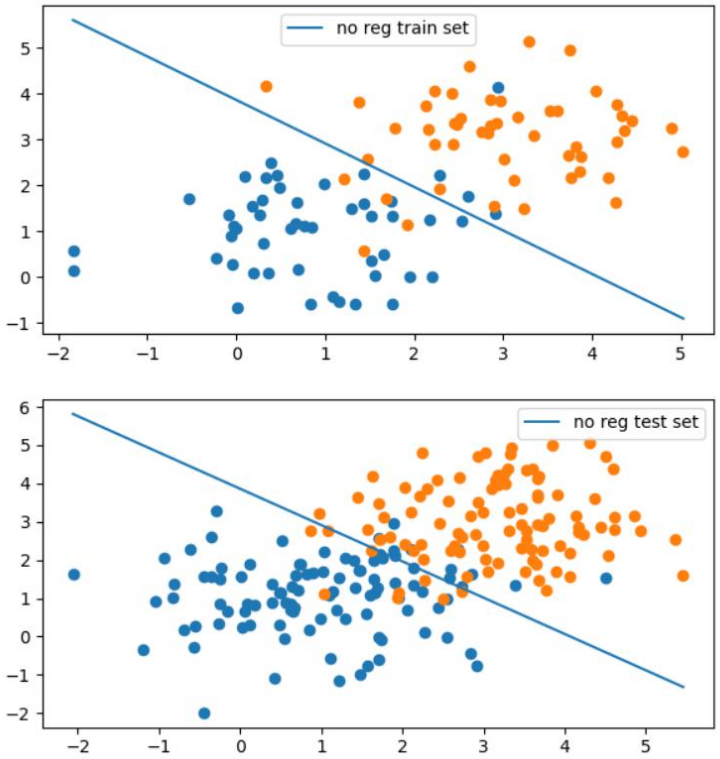


3.3 参数计算

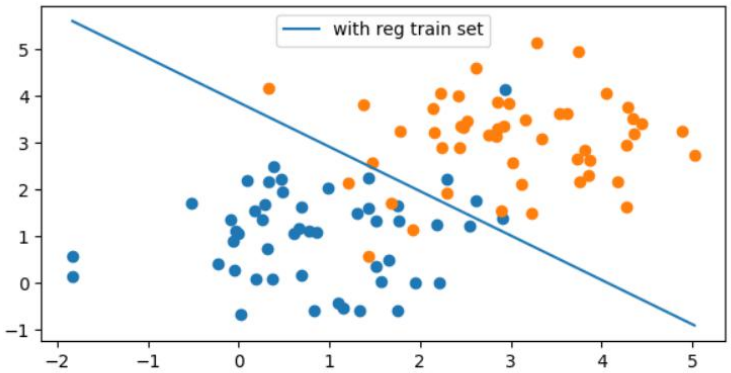
参数计算选用之前较为熟悉的梯度下降法，原理不再赘言。

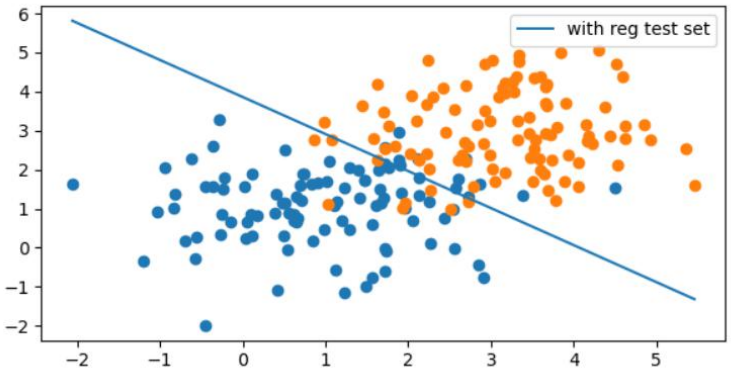
3.4 实验结果

对于满足朴素贝叶斯假设的生成数据，不带正则项的分类结果图示如下



带有正则项的分类结果图示如下





两类点的期望向量分别是(1, 1)和(3, 3)，协方差矩阵都为二维单位阵，其中训练集两类点各为 50 个，测试集各为 100 个，统计结果输出如下

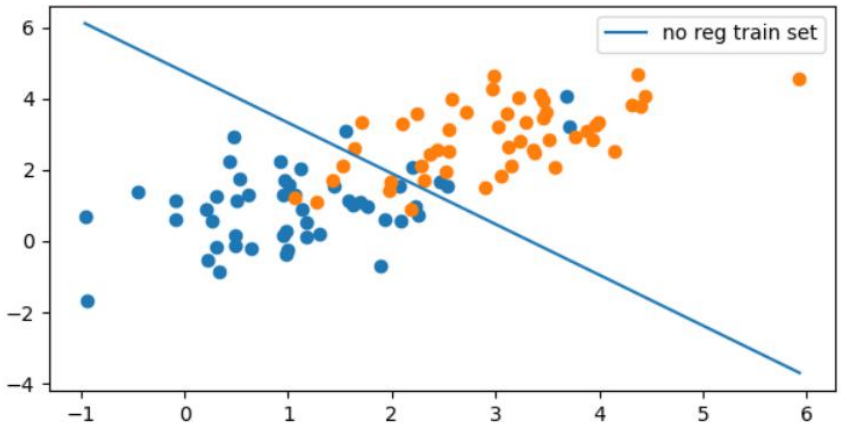
```
梯度下降次数为6160次
无正则条件下的准确率为：92.000000
无正则条件下在测试集的准确率为：89.500000
梯度下降次数为6160次
有正则条件下的准确率为：92.000000
有正则条件下在测试集的准确率为：89.500000
```

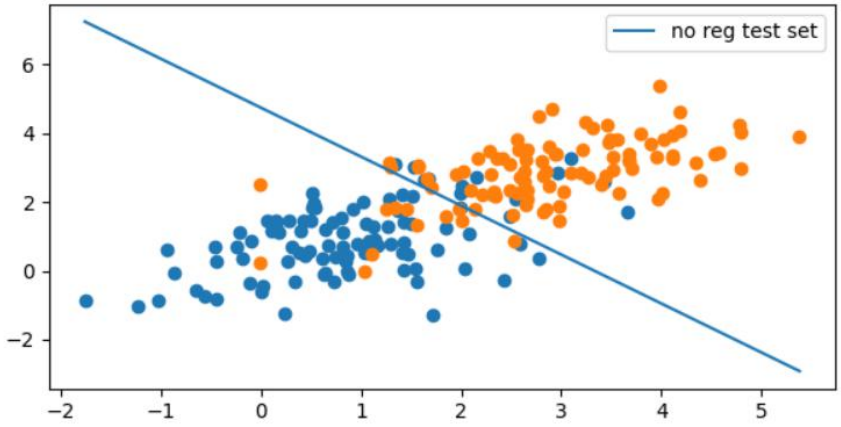
从图表及准确率可以看出：

- logistic 在测试集上的准确率较低；
- 有正则条件下的准确率与无正则条件下的准确率之间无明显区别；
- 分类结果基本是肉眼可见的最佳结果。

在实际的多次实验的情况下，有时也会有测试集的准确率较高的情况出现。

不满足贝叶斯假设意味着两个维度之间不能做到条件独立，反应到协方差矩阵上则是非对角线元素不为 0，首先设置非对角线上的元素为 0.5：

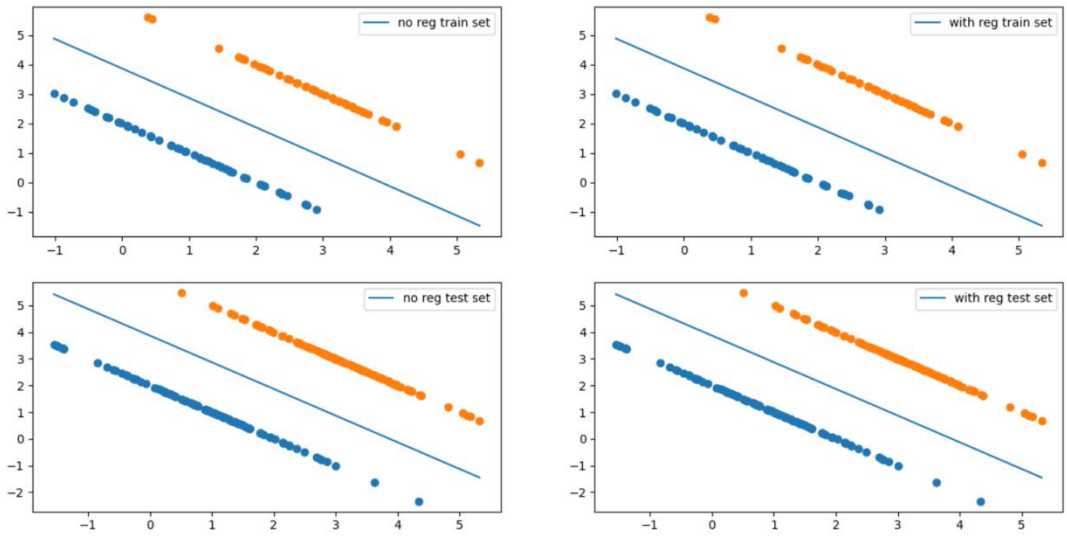




观察对比图像可以看出，散点图的整体趋势更加偏向右上---左下的直线两侧分布，分类结果相对不满足朴素贝叶斯假设的结果稍低，但整体影响不大，如前所述，在节选的这次实验中测试集的准确率相对训练集准确率较高，究其原因，可能是因为数据尽管是随机生成的，但整体上相对于真实数据还会更加符合高斯分布的规律，准确率较高就不足为怪了。

```
梯度下降次数为3823次
无正则条件下的准确率为：87.000000
无正则条件下在测试集的准确率为：87.500000
梯度下降次数为3823次
有正则条件下的准确率为：87.000000
有正则条件下在测试集的准确率为：87.500000
```

非对角线上的元素为正数时，分布偏向右上---左下的直线两侧，考虑如果分布如果能够偏向左上---右下的直线，那么我们的分类直线理应会得到更好的结果，要得到这样的分布可以考虑将非对角线上的元素设置为负数，极端一些，将其设为-1，结果如下：



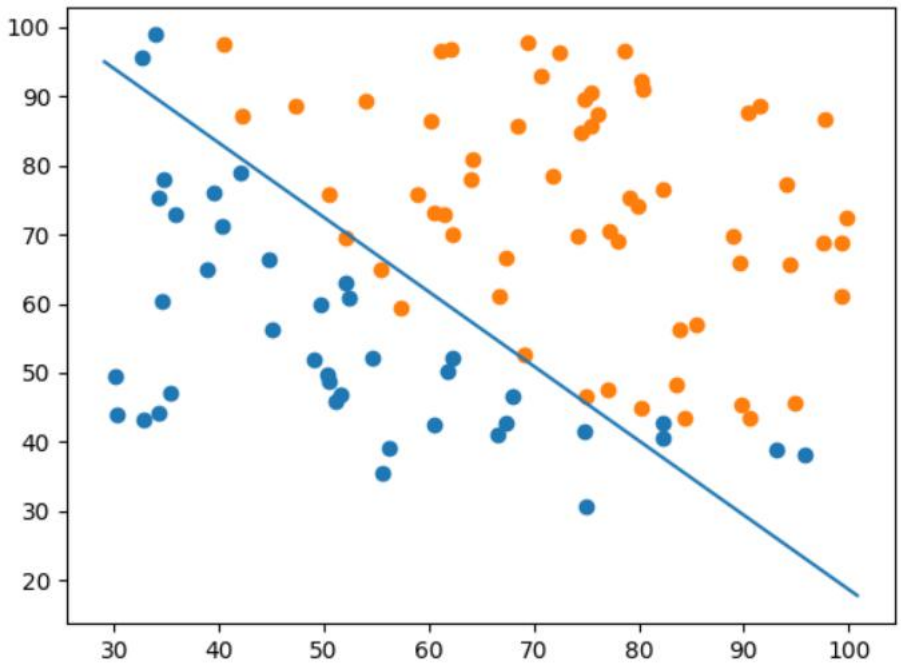

```
梯度下降次数为67366次
无正则条件下的准确率为：100.000000
无正则条件下在测试集的准确率为：100.000000
梯度下降次数为67169次
有正则条件下的准确率为：100.000000
有正则条件下在测试集的准确率为：100.000000
```

可以看到，准确率达到了 100%。

由以上可知，哪怕在不满足朴素贝叶斯的假设下，我们的 logistic 回归仍能取到较好的结果，类似于在马尔可夫链中我们假设当前状态只与前一个状态有关仍能得到不错的结果。

3.5 UCI 数据集

为了便于体现分类结果，选择使用吴恩达在 Coursera 中给出的一个有两个特征的二分类数据集。分类结果如下：



结果的准确率较高，达到了 91%，但是梯度下降的次数较多，下降较慢，达到了 617272 次。

```
梯度下降次数为617292次
准确率为：91.000000
```

实际上，下降次数过慢的原因主要是学习率过小，我的初始学习率为 0.01，而在之前我一般设置为 0.1，在迭代的过程中再根据实际情况进行减半处理，但是由于这个数据集的特征相对较大，学习率设置为 0.1 时的初始几次迭代会因为 $X\theta$各维度值都较大，代入到 sigmoid 函数时计算$e^{-X\theta}$整体值极小，因为计算机的舍入误差，会出现后验分布函数结果在各维度上均为 1 的情况，而在计算代价函数时需要这样一步：

```
J = (-Y.T @ np.log(h(X, Theta)) - (I - Y).T @ np.log(I - h(X, Theta)) + _lambda * sum(Theta[1:n]**2) / 2) / m
```

其中 np.log(I – h(X, Theta))这一部分会出现 log(0)，导致结果错误。

思考一下，想到了三种解决方案：

- 修改参数θ初始值，避免各维度值过大的情况，但无法保证每次迭代都可以满足将维度限制在一定范围；
- 采用牛顿法等其他方法进行计算；
- 进行 feature scaling，将特征值减去平均数再除以极差，缩小特征的范围。

最终我采用了第三种方案，结果如下：

```
梯度下降次数为28420次
准确率为：89.000000
```

可以看出，因为初始学习率调高到 0.1，因此梯度下降次数大幅减小到 28420 次，准确率略有下降，但可以接受。