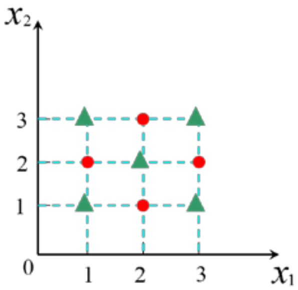
目录

非线性分类与迭代求权向量

1　问题描述

1.1　非线性分类器

求分离●和▲的函数。（不是唯一解）



1.2　迭代修正权向量

data1 = [1 1; 2 0; 2 1; 0 2; 1 3];

data2 = [-1 2; 0 0; -1 0; -1 -1; 0 -2];

求：

迭代修正求权向量的方法，求上述两类的线性识别函数；

迭代修正求权向量的方法，求上述两类的线性识别界面；

画出该识别界面将训练样本的区分结果图示。

2　非线性分类器设计

本次实验中，我设计了如下四种分类器。

2.1　奇偶性判定

从原图点阵中可以明显看出，两类数据点是依次交错排列的，因此根据点的坐标和的奇偶性即可做出分类判定。该分类函数不易画出图像，其表达式是

$$c=(x+y)\bmod 2=\begin{cases}1 \Rightarrow \text{class1}\\0 \Rightarrow \text{class2}\end{cases} \tag{1}$$

2.2　圆形分类界面

图中，红色圆形的点均在一个环内，因此可以考虑用圆作为分类面，但此时需要两个分类函数共同完成分类。分类界面由公式(2)-(4)确定。

$$c_1=(x-2)^2+(y-2)^2-\left(\frac{1}{2}\right)^2 \tag{2}$$

$$c_2=(x-2)^2+(y-2)^2-\left(\frac{6}{5}\right)^2 \tag{3}$$

$$\begin{cases} c_1 \le 0 \vee c_2 \ge 0 \;\Rightarrow\; \text{class1} \\ c_1 \ge 0 \wedge c_2 \le 0 \;\Rightarrow\; \text{class2} \end{cases} \tag{4}$$

分类结果如图 1 所示。

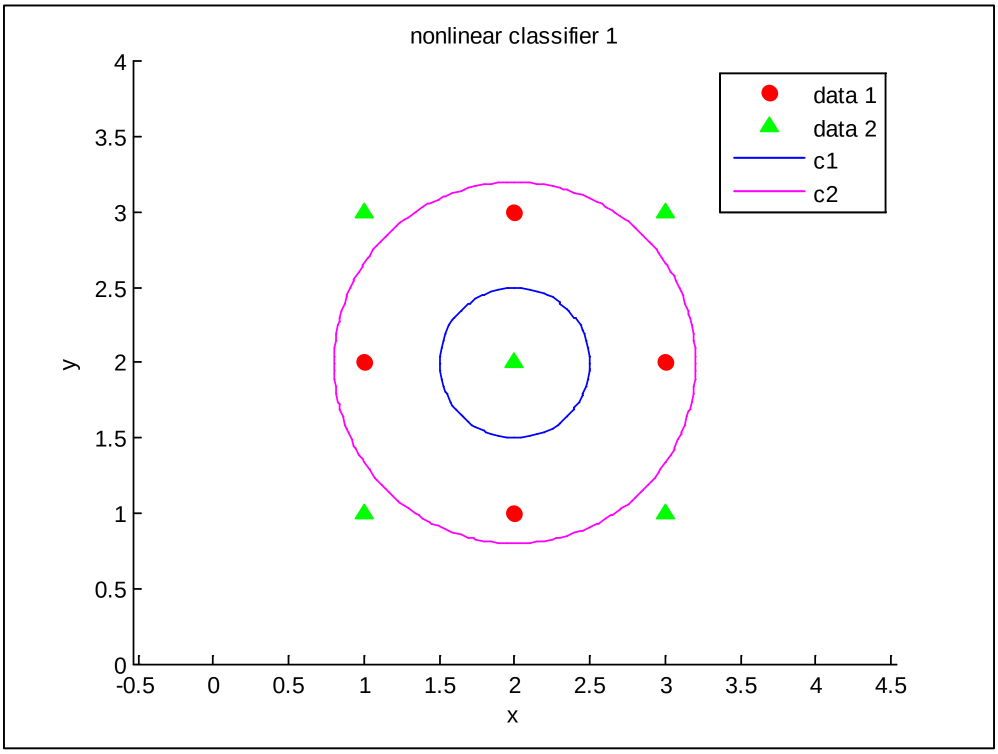


图 1 圆形分类界面

2.3 双曲线分类界面

类似的，可以选取两组双曲线作为分类界面。此时，分类界面由公式(5)-(7)确定。

$$c_1 = \frac{(x-2)^2}{1/3} - \frac{(y-2)^2}{1/6} - 1 \tag{5}$$

$$c_2 = \frac{(y-2)^2}{1/3} - \frac{(x-2)^2}{1/6} - 1 \tag{6}$$

$$\begin{cases} c_1 \ge 0 \vee c_2 \ge 0 \;\Rightarrow\; \text{class1} \\ c_1 \le 0 \wedge c_2 \le 0 \;\Rightarrow\; \text{class2} \end{cases} \tag{7}$$

分类结果如图 2 所示。

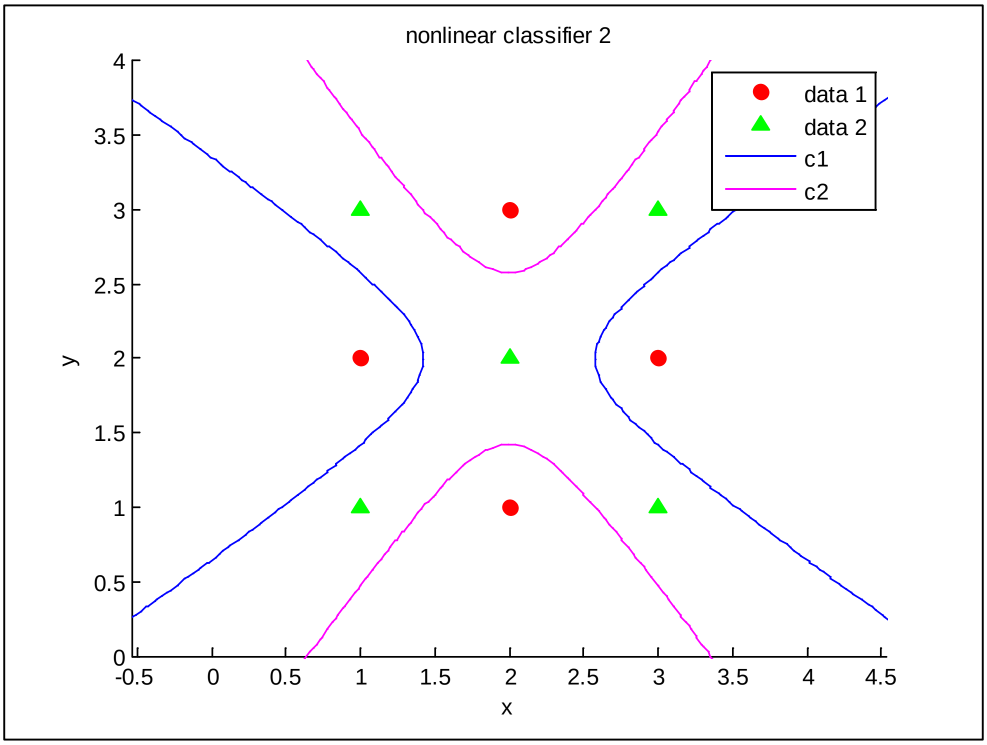


图 2 双曲线分类界面

2.4 正弦曲线分类界面

上述圆形和双曲线分类面均有一个问题，就是需要两个函数共同完成分类。那么，是否可以用一个非线性的函数完成分类呢？

为了更好地在直角坐标系下寻找分类面，我先将图中的数据点旋转 $\frac{\pi}{4}$ 处理，这样得到的新数据点如图 3 所示。

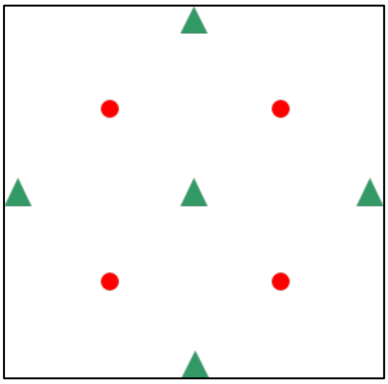


图 3 旋转后的新数据点

可以看出，此时用一个类似于“M”形状的函数即可以将两类数据分开。最初，我尝试了四次函数，试图通过所有点到曲线距离的和最小来得到该四次函数的系数。但是，建模之后才发现这是一个非凸问题，求解难度很大，即使采用凸分析 cvx 工具也无法得到结果。

此外，同时考虑到四次函数过于复杂（实际中即很容易造成过拟合），因此，我最终尝试了采用正弦曲线作为分类界面，这样分类界面的模型很简单，需要求解的参数也明显减少。

整个过程的数学分析如下。

对于数据点(x,y)，将其与旋转矩阵相乘，得到旋转后的新坐标，即

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = R_\theta \begin{pmatrix} x \\ y \end{pmatrix} \tag{8}$$

其中

$$R_\theta = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{9}$$

取旋转角度$\theta = \dfrac{\pi}{4}$，旋转之后，很容易确定一个可行的分界面函数是

$$y' = 2\sin\left(\sqrt{2}\pi x' - \frac{\pi}{2}\right) + 2\sqrt{2} \tag{10}$$

再将公式(8)代入到公式(10)，用$\begin{pmatrix} x \\ y \end{pmatrix}$替代公式(10)中的$\begin{pmatrix} x' \\ y' \end{pmatrix}$，即可得到原始数据的分界面方程。分类结果如图 4 所示。

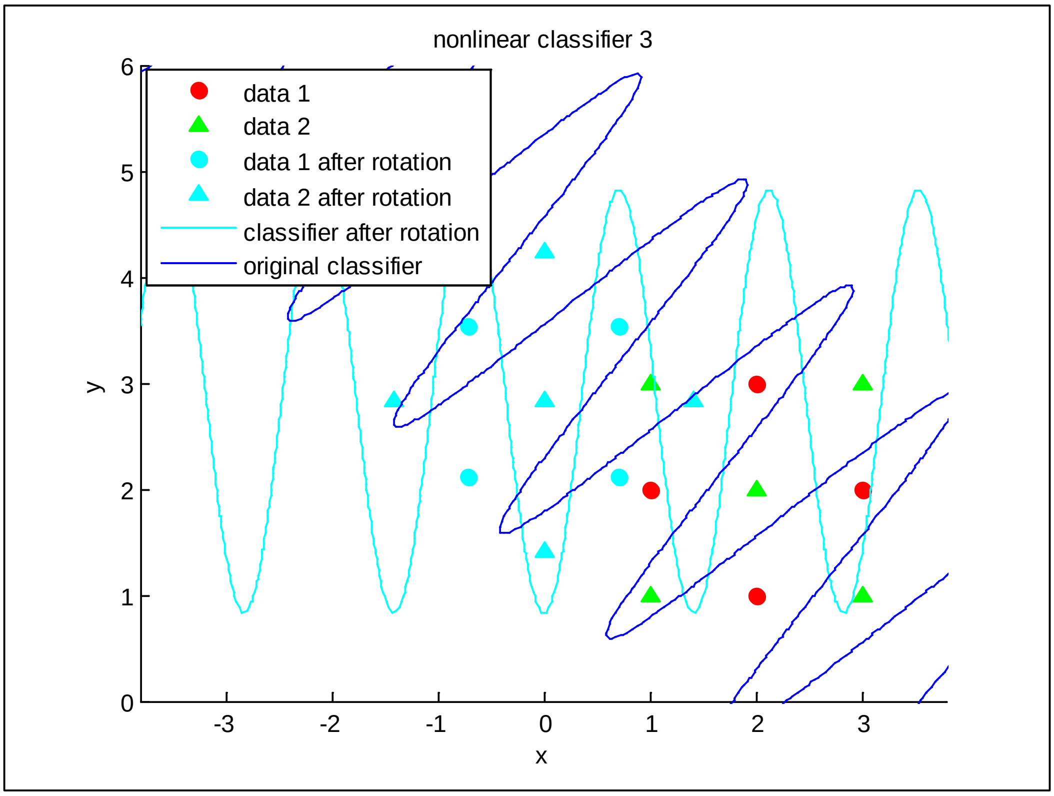


图 4 正弦函数分类界面

图中，所有青色的元素均是旋转之后的，可以看出，在旋转后的空间比较容易求得分类正弦函数，最后再将该正弦函数变换到原空间即可完成对原数据的分类。

3 迭代修正求权向量

3.1 原始迭代方法

(1) 将两类样本合并成数据集

$$S=\left\{X_1^{'},X_2^{'},\cdots,-Y_1^{'},-Y_2^{'},\cdots\right\}=\left\{Z_1^{'},Z_2^{'},\cdots\right\} \tag{11}$$

(2) 适当选取W初始值W_0

(3) 令$k=0$，依次从集合S中取出样本$Z_i^{'}$，作如下操作：

a) 如果$W_k^T Z_i^{'}>0$，那么$i=i+1$（即取下一个样本）

b) 否则进行一次迭代$W_{k+1}=W_k+cZ_i^{'}$，可取$c=1$

(4) 对S中的全部样本按顺序周而复始地重复进行上一步骤操作

(5) 如果数据集确实线性可分，上述迭代一定可以收敛。即，可以找到某个W，使得S中的全部样本都满足$W_k^T Z_i^{'}>0$。

3.2 改进迭代过程

为了尽快收敛，可以采用绝对值法对迭代过程加以改进。即，在S中的全部样本中，挑选出不满足条件$W_k^T Z_i^{'}>0$的全部样本，求出其平均值$\overline{Z^{'}}$，然后取

$$c>-\frac{W_k^T\overline{Z^{'}}}{\left\|\overline{Z^{'}}\right\|^2} \tag{12}$$

其他处理过程与原始迭代方程相同。

3.3 结果与分析

用上述两种方法迭代求解权向量，得到的结果如图 5 所示。

仿真中，迭代的次数与最终得到的权向量分别如下：

```
w1 = [6 3 -1]
iter count = 12
w2 = [5.38589324618736 2.56627450980392 -0.132549019607843]
iter count = 8
>>
```

即，原始迭代方法迭代 12 次后收敛，改进后的方法迭代 8 次后收敛，改进的方法确实加快了收敛速度。

对迭代求取权向量方法的分析如下：

(1) 由于c,W_0以及取样本的顺序等不同，可能收敛到不同的状态，但都满足线性可分条件（只要原始数据集线性可分）。

(2) 该题目中的数据确实线性可分，因此算法较快地收敛，但往往我们事先难以知道是否收敛（高维空间），实际操作时要采取措施。

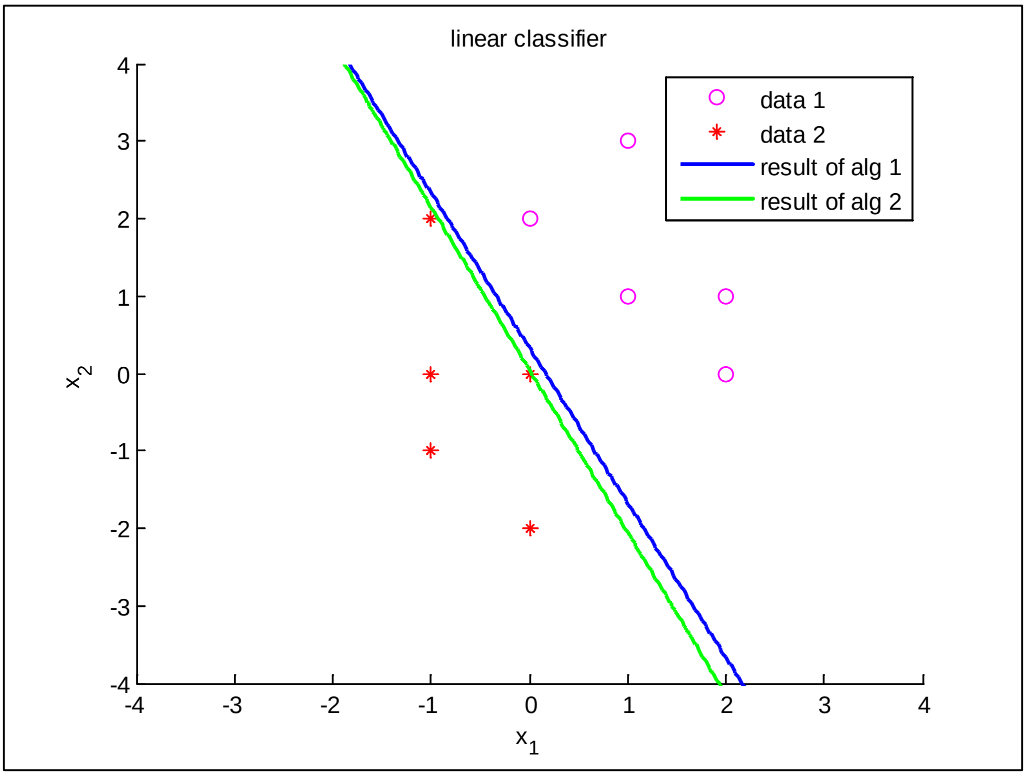


图 5 迭代求解权向量的结果

(3) 从图 5 中可以看出，最终得到的线性分类面与其中一类数据点距离非常小，这意味着，若将分类器应用于新数据，分界面附近的数据点很容易被误判。之所以如此，是因为算法在迭代过程中，一旦找到了一个能正确分类的分界面，迭代修正过程即终止，最终导致迭代收敛时分类面往往在某一类数据点附近。

(4) 为了避免上述情况，类似于神经网络更新权值的做法，我们可以在每次迭代时给更新权重添加一个冲量项。如果采用 SVM 算法，上述问题可以很好地避免，因为 SVM 总是最大化决策边界的边缘。