深圳大学考试答题纸

(以论文、报告等形式考核专用)

二〇 二〇 ～二〇 二一 学年度第 二 学期

课程编号 1502990001 课程名称 机器学习导论 主讲教师 贾森、徐萌 评分

学 号 2018152051 姓名 刘睿辰 专业年级 计算机科学与技术 2018 级

教师评语：

题目：**线性判别分析原理介绍与代码实现**

一、 实验目标

1. 理解线性判别分析(Linear Discriminant Analysis)，了解其工作背后的数学原理；
2. 通过 MATLAB 来编程实现该算法；
3. 通过数据集测试其可行性。

二、 实验环境

1. 操作系统：Windows 10；
2. 编程平台：MATLAB R2019a。

三、 实验过程及步骤

1. 线性判别分析介绍

线性判别分析(Linear Discriminant Analysis, LDA)是一种典型的线性学习方法，在二分类问题上因为最早由费希尔(Fisher, 1936)提出，所以亦称 Fisher 判别分析。本质上来说，LDA 也属于一种降维的方法。那么，LDA 和主成分分析(Principal Component Analysis, PCA)有什么区别和联系呢？

事实上，LDA 是一种关于监督学习(supervised learning)的降维技术，而 PCA 是针对非监督学习(unsupervised learning)的降维技术。我们以如图 1 所示的例子来做区分。

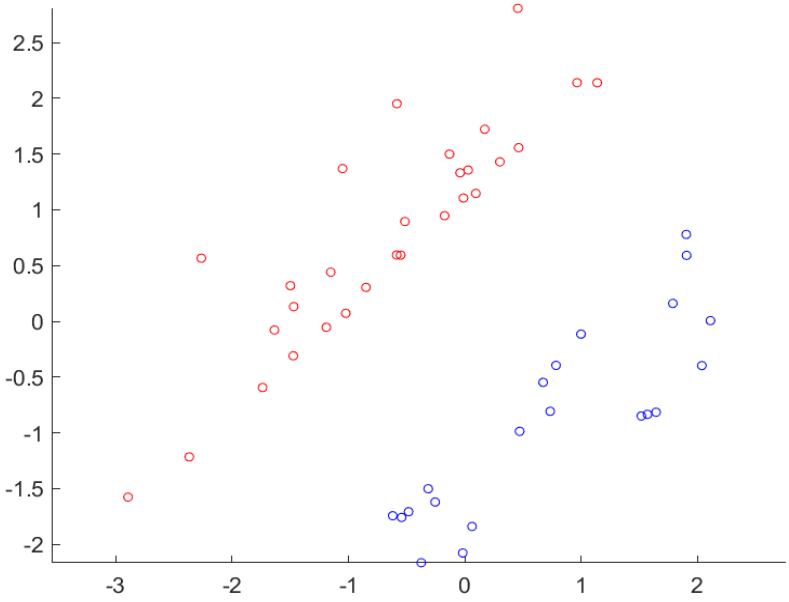

图 1. 监督学习数据集（样例）

对于这个样本集，如果我们想用 PCA 进行降维，结果将会如图 2(a)所示，因为 PCA 保证样本方差达到最大值而协方差达到最小值，降维的超平面使得样本之间距离足够大，也就是使得降维之后的样本足够稀疏。

但是，很明显，PCA 的降维带来一个问题，那就是我们的正负样本已经完全融合了，降维之后的数据集将不再线性可分，甚至是不可分的。如果不可分或者分类很困难的话，这种降维方式就不是我们的选择。

为了强调类与类之间的区别，我们需要用另一种方法来进行降维，也就是说我们既要保留数据的特征，又要在监督学习的条件下保留类与类之间的区别，所以这时候我们想要的降维效果就如图 2(b)所示，这也就引出了线性判别分析的基本概

念。

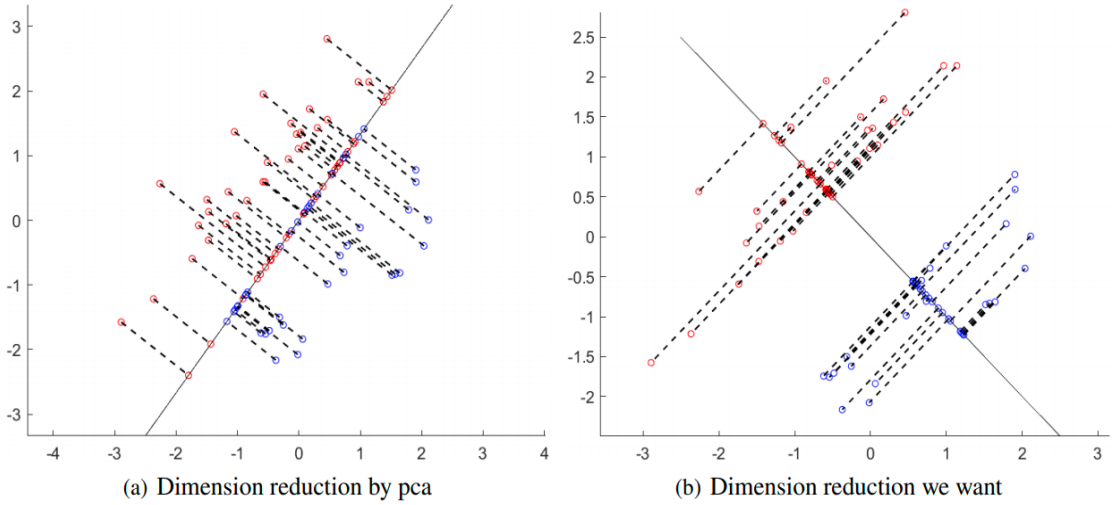

图 2. 两种不同的降维思路

像图 2 这种降维，每一个类中的样本足够紧密，类与类之间也足够清晰，这就是线性判别分析的基本思路，下面我们从数学的角度来了解线性判别分析。

2. 二分类问题中线性判别分析的数学原理

在上一节中我们已经阐明了线性判别分析的基本思想，即：

1） 同一类中样本投影点足够接近；

2） 不同类的投影点尽可能远离。

也就是如图 3 所示。

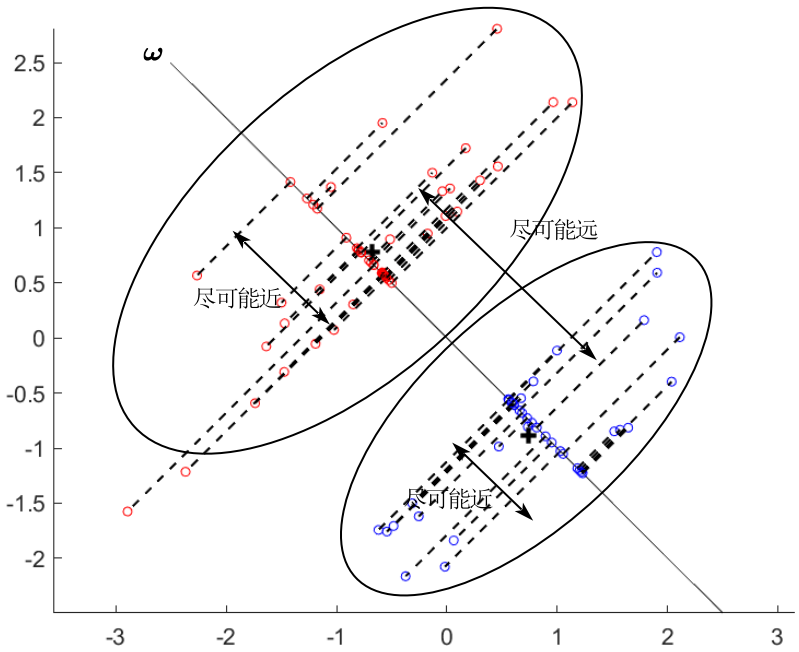

图 3. LDA 算法基本思想图示

了解了 LDA 的基本思想之后，我们将我们的思想进行量化。首先我们先进行符号约定。

$\boldsymbol{\mu}_i$：第i类样本的均值向量；

$\boldsymbol{\omega}$：投影方向；

$\boldsymbol{x}$：样本点向量；

X^i：第i类样本集合。

首先，我们来实现类与类之间的距离的表示。关于向量内积我们有

$$\boldsymbol{a}.\boldsymbol{b}=|\boldsymbol{a}|\,|\boldsymbol{b}|\cos\theta$$

那么，关于向量$\boldsymbol{a}$在向量$\boldsymbol{b}$上的投影可以表示为

$$|\boldsymbol{a}|\cos\theta=\frac{\boldsymbol{a}.\boldsymbol{b}}{|\boldsymbol{b}|}$$

那么第i类样本的均值向量$\boldsymbol{\mu}_i$在投影方向$\boldsymbol{\omega}$上的投影可以表示为

$$l=\frac{\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_i}{|\boldsymbol{\omega}|}$$

所以两个类的均值向量在投影方向$\boldsymbol{\omega}$上的差距就为

$$\Delta=\frac{\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_1-\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_0}{|\boldsymbol{\omega}|}$$

由于$|\boldsymbol{\omega}|$是常数，所以我们首先得到第一个优化目标

$$\max_{\boldsymbol{\omega}}\|\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_1-\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_0\|_2^2$$

此外，我们有

$$\begin{aligned}&\|\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_1-\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_0\|_2^2\\=&(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_1-\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_0)\,(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_1-\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_0)^{\mathrm{T}}\\=&(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_1-\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_0)\,(\boldsymbol{\mu}_1^{\mathrm{T}}\boldsymbol{\omega}-\boldsymbol{\mu}_0^{\mathrm{T}}\boldsymbol{\omega})\\=&\boldsymbol{\omega}^{\mathrm{T}}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_0)\,(\boldsymbol{\mu}_1-\boldsymbol{\mu}_0)^{\mathrm{T}}\boldsymbol{\omega}\end{aligned}$$

我们要做的目标之一就是让上面这个式子达到最大值，也就是“类间距离最大”。

那么接下来我们考虑第二点，也就是类内样本投影点足够接近。我们知道，反应数据离散程度的量就是方差，所以我们尝试用方差来量化这个指标。
在同一个类中，所有样本和该类的均值向量的投影距离之差都可以表示为

$$\|\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}-\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_i\|_2^2$$

那么该类中所有样本与均值投影距离之和应该达到最小，这样才能保证该类中样本足够集中。所以以 0 类为例，我们有

$$\boldsymbol{s}_0^2=\sum_{\boldsymbol{x}\in\mathrm{X}^0}(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}-\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_0)^2$$

$$
\begin{aligned}
&= \sum_{\boldsymbol{x} \in \mathrm{X}^0} \boldsymbol{\omega}^{\mathrm{T}} (\boldsymbol{x} - \boldsymbol{\mu}_0) (\boldsymbol{x} - \boldsymbol{\mu}_0)^{\mathrm{T}} \boldsymbol{\omega} \\
&= \boldsymbol{\omega}^{\mathrm{T}} \sum_{\boldsymbol{x} \in \mathrm{X}^0} (\boldsymbol{x} - \boldsymbol{\mu}_0) (\boldsymbol{x} - \boldsymbol{\mu}_0)^{\mathrm{T}} \boldsymbol{\omega}
\end{aligned}
$$

我们可以发现，$\sum_{\boldsymbol{x} \in \mathrm{X}^0} (\boldsymbol{x} - \boldsymbol{\mu}_0) (\boldsymbol{x} - \boldsymbol{\mu}_0)^{\mathrm{T}}$事实上就是协方差矩阵，表示为$\boldsymbol{\Sigma}_0$，即

$$
\boldsymbol{s}_0^2 = \boldsymbol{\omega}^{\mathrm{T}} \sum_{\boldsymbol{x} \in \mathrm{X}^0} (\boldsymbol{x} - \boldsymbol{\mu}_0) (\boldsymbol{x} - \boldsymbol{\mu}_0)^{\mathrm{T}} \boldsymbol{\omega} = \boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{\Sigma}_0 \boldsymbol{\omega}
$$

针对两个类的情况，我们有

$$
\boldsymbol{s}^2 = \boldsymbol{s}_0^2 + \boldsymbol{s}_1^2 = \boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{\Sigma}_0 \boldsymbol{\omega} + \boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{\Sigma}_1 \boldsymbol{\omega}
$$

我们应该使这个值达到最小，也就是类内方差最小，数据更密集。

综合以上两种优化目标，我们导出我们最终的优化目标

$$
\begin{aligned}
\boldsymbol{J} &= \frac{\| \boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{\mu}_1 - \boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{\mu}_0 \|_2^2}{\boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{\Sigma}_0 \boldsymbol{\omega} + \boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{\Sigma}_1 \boldsymbol{\omega}} \\
&= \frac{\boldsymbol{\omega}^{\mathrm{T}} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^{\mathrm{T}} \boldsymbol{\omega}}{\boldsymbol{\omega}^{\mathrm{T}} (\boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1) \boldsymbol{\omega}}
\end{aligned}
$$

其中，我们将$\boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1$定义为**类内散度矩阵**，表示各类内部数据离散程度，即

$$
\boldsymbol{S}_w = \boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1
$$

此外，我们将$(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^{\mathrm{T}}$定义为**类间散度矩阵**，表示各类间离散程度，即

$$
\boldsymbol{S}_b = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^{\mathrm{T}}
$$

所以此时有

$$
\boldsymbol{J} = \frac{\boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{S}_b \boldsymbol{\omega}}{\boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{S}_w \boldsymbol{\omega}}
$$

在这个优化目标中，优化目标大小仅与$\boldsymbol{\omega}$有关。考虑到不失一般性，我们设

$$
\boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{S}_w \boldsymbol{\omega} = 1
$$

此时优化目标变为

$$
\begin{aligned}
\min_{\boldsymbol{\omega}} \quad & - \boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{S}_b \boldsymbol{\omega} \\
\text{s.t.} \quad & \boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{S}_w \boldsymbol{\omega} = 1
\end{aligned}
$$

这是典型的拉格朗日乘数法优化问题。我们设

$$
L(\boldsymbol{\omega}, \lambda) = - \boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{S}_b \boldsymbol{\omega} + \lambda (\boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{S}_w \boldsymbol{\omega} - 1)
$$

对$\boldsymbol{\omega}$求偏导数则为

$$\frac{\partial L(\boldsymbol{\omega},\lambda)}{\partial \boldsymbol{\omega}} = -2\boldsymbol{S}_b\boldsymbol{\omega} + 2\lambda\boldsymbol{S}_w\boldsymbol{\omega} = 0$$

如果$\boldsymbol{S}_w$是可逆矩阵，则有

$$\boldsymbol{S}_w^{-1}\boldsymbol{S}_b\boldsymbol{\omega} = \lambda\boldsymbol{\omega}$$

所以现在的问题就转化为求$\boldsymbol{S}_w^{-1}\boldsymbol{S}_b$的特征值和特征向量，即

$$\boldsymbol{S}_w^{-1}\boldsymbol{S}_b = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^{\mathrm{T}}$$

求解出来的矩阵$\boldsymbol{U}$为列特征向量组成的方阵。然后按照特征值大小进行降序排序，并取前k列组成投影方向。这一步和主成分分析非常相似，我们所取的都是特征值最大的k个特征向量作为主成分来进行降维。

得到了截取完毕的矩阵$\boldsymbol{U}_{reduced}$，可以在低维度下显示投影的位置，即

$$\boldsymbol{z} = \boldsymbol{U}_{reduced}^{\mathrm{T}}\boldsymbol{x}$$

如果在原维度下显示投影的位置我们就需要将低维空间的投影点还原到原来的高维空间中，原理如图 4 所示。

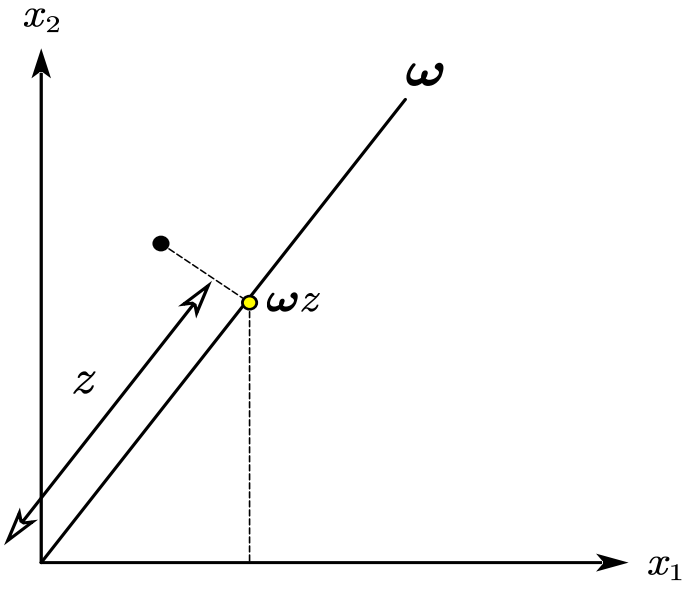

图 4. 还原低维空间中的点到高维空间

图 4 中，z为低维空间中的点，在这里相当于是一个标量，$\boldsymbol{\omega}z$是一个二维空间中的向量，代表了投影点的坐标。

3. 二分类问题线性判别分析的实现
我们以二维空间为例。首先我们将数据集加载进 MATLAB，然后将数据点和样

本标签分别存储，并将所有样本点用散点图的方法绘制出来。代码如图 5 所示。

```
data = load('dataset1.txt');
X = data(:, [1, 2]);
y = data(:, 3);
pos = find(y == 1);
neg = find(y == 0);
scatter(X(pos,1),X(pos,2),15,'r');
hold on;
scatter(X(neg,1),X(neg,2),15,'b');
hold on;
```

图 5. 二分类问题 LDA 算法实现（1）

此时 pos 提取出所有 $y=1$ 的样本，neg 提取出所有 $y=0$ 的样本。

接下来我们来对样本进行标准化，也就是进行

$$\boldsymbol{x}^{*}=\boldsymbol{x}-\boldsymbol{\mu}_{i},\ \boldsymbol{x}\in \mathrm{X}^{i}$$

代码如图 6 所示。

```
function [X_norm mu] = fN(X)
mu = mean(X);
X_norm = bsxfun(@minus, X, mu);
end
```

图 6. 二分类问题 LDA 算法实现（2）

得到了标准化之后的样本矩阵之后，根据上一节的推导过程就可以求得类间散度矩阵和类内散度矩阵了，代码如图 7 所示。

```
% Within-class divergence matrix, (2*60)*(60*2) = (2*2)
Sw = X_pos' * X_pos + X_neg' * X_neg;

% Inter-class divergence matrix, (2*1)*(1*2) = (2*2)
Sb = (mu_pos-mu_neg)' * (mu_pos-mu_neg);
```

图 7. 二分类问题 LDA 算法实现（3）

求得 $\boldsymbol{S}_{w}$ 和 $\boldsymbol{S}_{b}$ 之后，需要对 $\boldsymbol{S}_{w}^{-1}\boldsymbol{S}_{b}$ 进行特征分解，并按照特征值降序序列对特征向量排序，代码如图 8 所示。

```
[V, D] = eig(inv(Sw) * Sb);
[D_sort,index] = sort(diag(D),'descend');
D_sort = D_sort(index);
V_sort = V(:,index);
V = V_sort;
```

图 8. 二分类问题 LDA 算法实现（4）

MATLAB 的 eig 函数是用来做特征分解的，这里返回的 $\boldsymbol{V}$ 是特征向量（列向量），$\boldsymbol{D}$ 是一个对角矩阵，对角元素为特征值。

在这里我们去取第一列特征值来做降维，所以我们得到了我们的目标ω。

下面我们来尝试将投影点在二维空间可视化出来。我们用的方法就是之前介绍的方法，也就是先作一维空间的投影点，然后还原回二维空间。代码如图 9 所示。

```
projection_pos = omega' * X_pos1';
projection_neg = omega' * X_neg1';

recover_pos = omega * projection_pos;
recover_neg = omega * projection_neg;
```

图 9. 投影+还原过程

我们得出的recover_pos 和recover_neg 就是二维空间内的投影点。我们将这些点绘制出来，并绘制原来的样本点，为了突出对应关系，我们将样本和投影相连，突出对应关系来体现投影效果。MATLAB 的drawLine 函数可以方便地连线。代码如图 10 所示。

```
% draw sample points and their projections respectively
for i = 1:size(recover_pos,2)
    scatter(recover_pos(1,i),recover_pos(2,i),15,'r');
    drawLine(recover_pos(:,i), X_pos1(i,:), '--k', 'LineWidth', 1);
end

for i = 1:size(recover_neg,2)
    scatter(recover_neg(1,i),recover_neg(2,i),15,'b');
    drawLine(recover_neg(:,i), X_neg1(i,:), '--k', 'LineWidth', 1);
end
```

图 10. 绘制样本投影并与样本之间进行对应

至此关于二分类问题的降维我们就完成了，我们以一组数据集来进行测试，结果如图 11 所示。

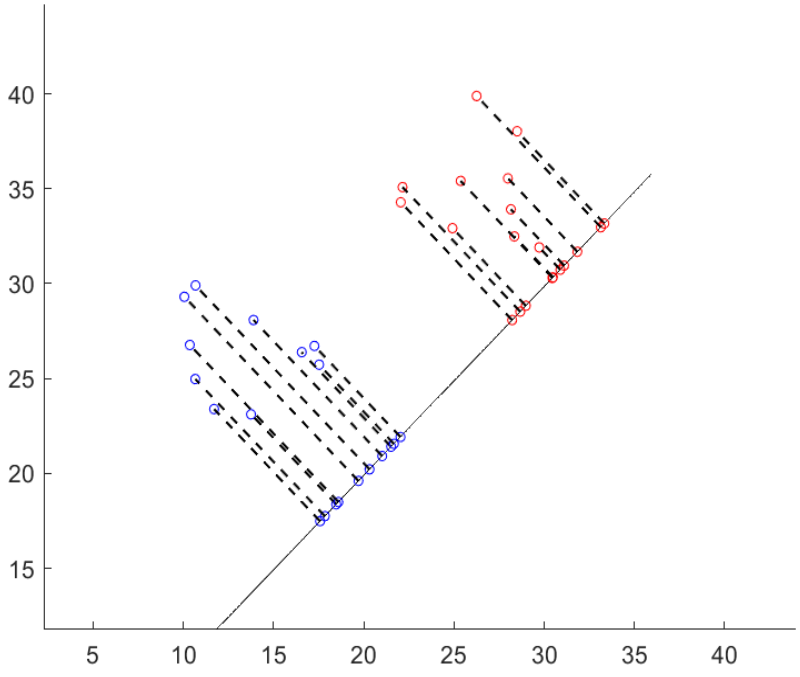

图 11. LDA 投影效果图

从图 11 中我们可以看到红色样本和蓝色样本都被降维到那条直线上，在高维空间我们称之为超平面(hyperplane)，而且正负样本之间的投影点距离足够大，同一个类之间投影点足够逼近。尤其好的性质是，降维之后样本保留了标签而且看起来更加可分，所以 LDA 有时候不仅仅用作降维，也用作分类。
图 11 的两个类的样本看起来离得比较远，体现不出 LDA 的优势。我们以图 12 为例，图 12(a)的正负样本看起来比较密集，边界不明显。但是用 LDA 算法进行投影之后，如图 12(b)所示，在超平面上正负样本之间有一个明显的边界，这就体现出 LDA 的分类功能。

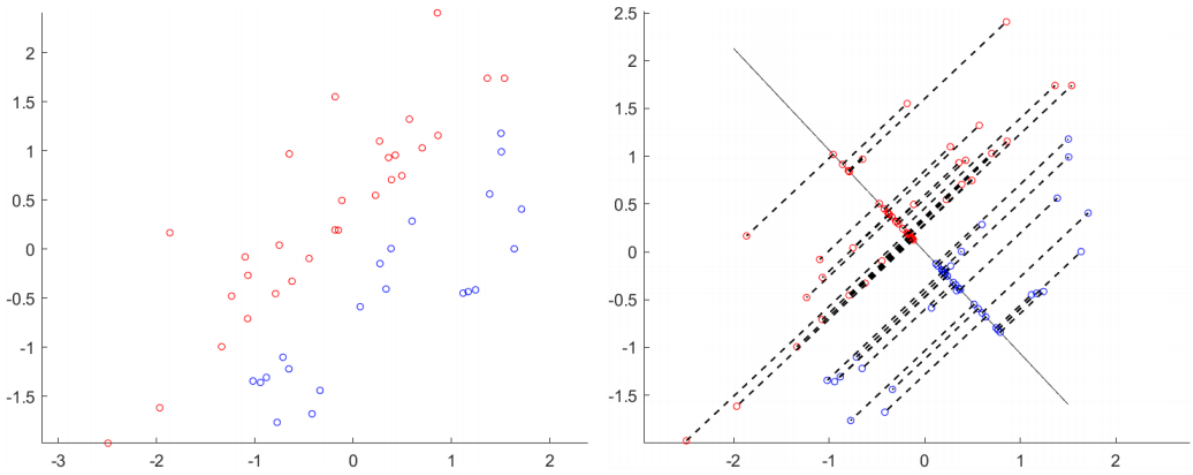

图 12. LDA 的分类功能

4. LDA 解决多分类任务
在这里我们在之前的基础之上再添加几个符号：

N：类的个数；

m_i：每个类中样本的个数；

m：样本容量；

$\boldsymbol{\mu}$：所有样本点的均值向量。

首先我们来规定一个新的变量，叫做**全局散度矩阵**，记作$\boldsymbol{S}_t$，我们有

$$\boldsymbol{S}_t = \boldsymbol{S}_b + \boldsymbol{S}_w$$

关于这个变量，我们的定义方法是

$$\begin{aligned}\boldsymbol{S}_t &= \sum_{i=1}^{m} (\boldsymbol{x}_i - \boldsymbol{\mu})(\boldsymbol{x}_i - \boldsymbol{\mu})^{\mathrm{T}} \\ &= \sum_{i=1}^{N} \sum_{j=1}^{m_i} (\boldsymbol{x}_j - \boldsymbol{\mu})(\boldsymbol{x}_j - \boldsymbol{\mu})^{\mathrm{T}}\end{aligned}$$

全局散度矩阵顾名思义，反映了样本总体偏离均值的程度。
接下来我们定义类内散度矩阵

$$\boldsymbol{S}_w = \sum_{i=1}^{N} \boldsymbol{S}_{w_i}$$

其中$\boldsymbol{S}_{w_i}$代表第i类的类内散度矩阵，根据二分类问题类内散度矩阵的定义我们可以得出

$$\boldsymbol{S}_{w_i}=\sum_{\boldsymbol{x}\in \mathrm{X}^i}\left(\boldsymbol{x}-\boldsymbol{\mu}_i\right)\left(\boldsymbol{x}-\boldsymbol{\mu}_i\right)^{\mathrm{T}}$$

则有

$$\boldsymbol{S}_w=\sum_{i=1}^{N}\sum_{\boldsymbol{x}\in \mathrm{X}^i}\left(\boldsymbol{x}-\boldsymbol{\mu}_i\right)\left(\boldsymbol{x}-\boldsymbol{\mu}_i\right)^{\mathrm{T}}$$

那么关于类间散度矩阵，我们可以通过全局散度矩阵和类内散度矩阵的差来得出。事实上，类间散度矩阵也可以通过数学推导来得出，下面我们给出一种推导方法。

$$\begin{aligned}
\boldsymbol{S}_b&=\boldsymbol{S}_t-\boldsymbol{S}_w\\
&=\sum_{i=1}^{N}\sum_{j=1}^{m_i}\left(\boldsymbol{x}_j-\boldsymbol{\mu}\right)\left(\boldsymbol{x}_j-\boldsymbol{\mu}\right)^{\mathrm{T}}-\sum_{i=1}^{N}\sum_{\boldsymbol{x}\in \mathrm{X}^i}\left(\boldsymbol{x}-\boldsymbol{\mu}_i\right)\left(\boldsymbol{x}-\boldsymbol{\mu}_i\right)^{\mathrm{T}}\\
&=\sum_{i=1}^{N}\sum_{j=1}^{m_i}\left(\boldsymbol{x}_j-\boldsymbol{\mu}\right)\left(\boldsymbol{x}_j-\boldsymbol{\mu}\right)^{\mathrm{T}}-\sum_{i=1}^{N}\sum_{j=1}^{m_i}\left(\boldsymbol{x}_j-\boldsymbol{\mu}_i\right)\left(\boldsymbol{x}_j-\boldsymbol{\mu}_i\right)^{\mathrm{T}}\\
&=\sum_{i=1}^{N}\sum_{j=1}^{m_i}\left[\left(\boldsymbol{x}_j-\boldsymbol{\mu}\right)\left(\boldsymbol{x}_j-\boldsymbol{\mu}\right)^{\mathrm{T}}-\left(\boldsymbol{x}_j-\boldsymbol{\mu}_i\right)\left(\boldsymbol{x}_j-\boldsymbol{\mu}_i\right)^{\mathrm{T}}\right]\\
&=\sum_{i=1}^{N}\sum_{j=1}^{m_i}\left[\left(\boldsymbol{x}_j-\boldsymbol{\mu}\right)\left(\boldsymbol{x}_j^{\mathrm{T}}-\boldsymbol{\mu}^{\mathrm{T}}\right)-\left(\boldsymbol{x}_j-\boldsymbol{\mu}_i\right)\left(\boldsymbol{x}_j^{\mathrm{T}}-\boldsymbol{\mu}_i^{\mathrm{T}}\right)\right]\\
&=\sum_{i=1}^{N}\sum_{j=1}^{m_i}\left(\boldsymbol{x}_j\boldsymbol{x}_j^{\mathrm{T}}-\boldsymbol{x}_j\boldsymbol{\mu}^{\mathrm{T}}-\boldsymbol{\mu}\boldsymbol{x}_j^{\mathrm{T}}+\boldsymbol{\mu}\boldsymbol{\mu}^{\mathrm{T}}-\boldsymbol{x}_j\boldsymbol{x}_j^{\mathrm{T}}+\boldsymbol{x}_j\boldsymbol{\mu}_i^{\mathrm{T}}+\boldsymbol{\mu}_i\boldsymbol{x}_j^{\mathrm{T}}-\boldsymbol{\mu}_i\boldsymbol{\mu}_i^{\mathrm{T}}\right)\\
&=\sum_{i=1}^{N}\sum_{j=1}^{m_i}\left(-\boldsymbol{x}_j\left(\boldsymbol{\mu}^{\mathrm{T}}-\boldsymbol{\mu}_i^{\mathrm{T}}\right)-\left(\boldsymbol{\mu}-\boldsymbol{\mu}_i\right)\boldsymbol{x}_j^{\mathrm{T}}+\boldsymbol{\mu}\boldsymbol{\mu}^{\mathrm{T}}-\boldsymbol{\mu}_i\boldsymbol{\mu}_i^{\mathrm{T}}\right)\\
&=\sum_{i=1}^{N}\left[\sum_{j=1}^{m_i}\boldsymbol{x}_j\left(\boldsymbol{\mu}_i^{\mathrm{T}}-\boldsymbol{\mu}^{\mathrm{T}}\right)-\left(\boldsymbol{\mu}-\boldsymbol{\mu}_i\right)\sum_{j=1}^{m_i}\boldsymbol{x}_j^{\mathrm{T}}+m_i\left(\boldsymbol{\mu}\boldsymbol{\mu}^{\mathrm{T}}-\boldsymbol{\mu}_i\boldsymbol{\mu}_i^{\mathrm{T}}\right)\right]\\
&=\sum_{i=1}^{N}\left[m_i\boldsymbol{\mu}_i\left(\boldsymbol{\mu}_i^{\mathrm{T}}-\boldsymbol{\mu}^{\mathrm{T}}\right)-\left(\boldsymbol{\mu}-\boldsymbol{\mu}_i\right)m_i\boldsymbol{\mu}_i^{\mathrm{T}}+m_i\left(\boldsymbol{\mu}\boldsymbol{\mu}^{\mathrm{T}}-\boldsymbol{\mu}_i\boldsymbol{\mu}_i^{\mathrm{T}}\right)\right]\\
&=\sum_{i=1}^{N}m_i\left(\boldsymbol{\mu}_i\boldsymbol{\mu}_i^{\mathrm{T}}-\boldsymbol{\mu}_i\boldsymbol{\mu}^{\mathrm{T}}-\boldsymbol{\mu}\boldsymbol{\mu}_i^{\mathrm{T}}+\boldsymbol{\mu}_i\boldsymbol{\mu}_i^{\mathrm{T}}+\boldsymbol{\mu}\boldsymbol{\mu}^{\mathrm{T}}-\boldsymbol{\mu}_i\boldsymbol{\mu}_i^{\mathrm{T}}\right)\\
&=\sum_{i=1}^{N}m_i\left(-\boldsymbol{\mu}_i\boldsymbol{\mu}^{\mathrm{T}}-\boldsymbol{\mu}\boldsymbol{\mu}_i^{\mathrm{T}}+\boldsymbol{\mu}_i\boldsymbol{\mu}_i^{\mathrm{T}}+\boldsymbol{\mu}\boldsymbol{\mu}^{\mathrm{T}}\right)\\
&=\sum_{i=1}^{N}m_i\left(\boldsymbol{\mu}-\boldsymbol{\mu}_i\right)\left(\boldsymbol{\mu}-\boldsymbol{\mu}_i\right)^{\mathrm{T}}
\end{aligned}$$

显然多分类 LDA 有很多方法，常见的一种优化代价函数为

$$\boldsymbol{J}=\frac{\mathrm{tr}\left(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{S}_b\boldsymbol{W}\right)}{\mathrm{tr}\left(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{S}_w\boldsymbol{W}\right)}$$

在这里我们并不能直接用两个矩阵作“除法”，而用矩阵的迹来代替矩阵作除法，本质上其实还是两个矩阵作比。在这里我们设

$$\mathrm{tr}(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{S}_w\boldsymbol{W})=1$$

所以又变成了拉格朗日乘数法优化问题，最终我们依然可以得到

$$\boldsymbol{S}_b\boldsymbol{W}=\lambda\boldsymbol{S}_w\boldsymbol{W}$$

这是一个广义特征值问题。如果$\boldsymbol{S}_w$可逆，我们依然可以将其逆矩阵乘到另一端，来求$\boldsymbol{S}_w^{-1}\boldsymbol{S}_b$的特征值，然后截取前$k$个特征值最大的特征向量即可。

还原样本点的方法和之前是一样的，下面我们通过实例来用 LDA 进行多分类。

5. LDA 解决多分类任务代码实现
我们使用一组数据集，该数据集中有三个类，将样本点可视化之后如图 13 所示，不同的颜色代表不同的类。

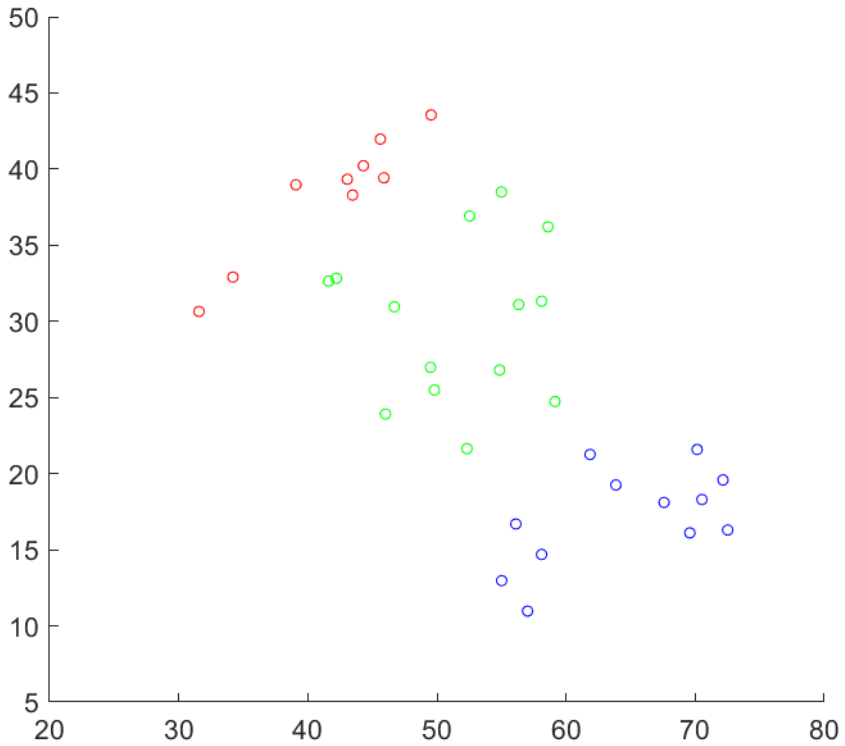

图 13. 样本可视化

按照多分类问题的步骤，我们先来计算全局散度矩阵，这部分比较简单，代码如图 14 所示。

```
[X_norm, mu] = fN(X);
St = X_norm' * X_norm;   %(2*31) * (31*2) = (2*2)
```

图 14. 计算全局散度矩阵

接下来我们需要计算类内散度矩阵。假设类的个数为K，我们用Mu来记录各类的均值向量，m数组记录每一个类中样本的个数以备后用。代码如图 15 所示。

```
Mu = zeros(K,size(X,2));
m = zeros(1,K);
Sw = zeros(size(X,2),size(X,2));  % 2*2
for i = 1:K
    % column vector which records the row index of ith type
    t = find(y == i-1);
    m(i) = size(t,1);  % sample size of the ith type
    X_type = X(t,:);  % extract the samples in ith type
    [X_type_norm, Mu(i,:)] = fN(X_type);
    Sw = Sw + X_type_norm' * X_type_norm;
end
```

图 15. 计算类内散度矩阵

得到了类内散度矩阵和前面计算的全局散度矩阵之后，我们可以直接用二者的差来表示类间散度矩阵，也可以用前面导出的结论，即

$$\boldsymbol{S}_b = \sum_{i=1}^{N} m_i (\boldsymbol{\mu} - \boldsymbol{\mu}_i) (\boldsymbol{\mu} - \boldsymbol{\mu}_i)^{\mathrm{T}}$$

来表示。

在后续的过程中，我们对$\boldsymbol{S}_w^{-1}\boldsymbol{S}_b$进行特征分解得到投影方向$\boldsymbol{\omega}$，然后降维得到低维空间的投影，再还原回原来的维度的过程和前面是一样的，这里就不再赘述。

我们来观察一下图 12 所示的样本点投影后的结果，如图 16 所示。

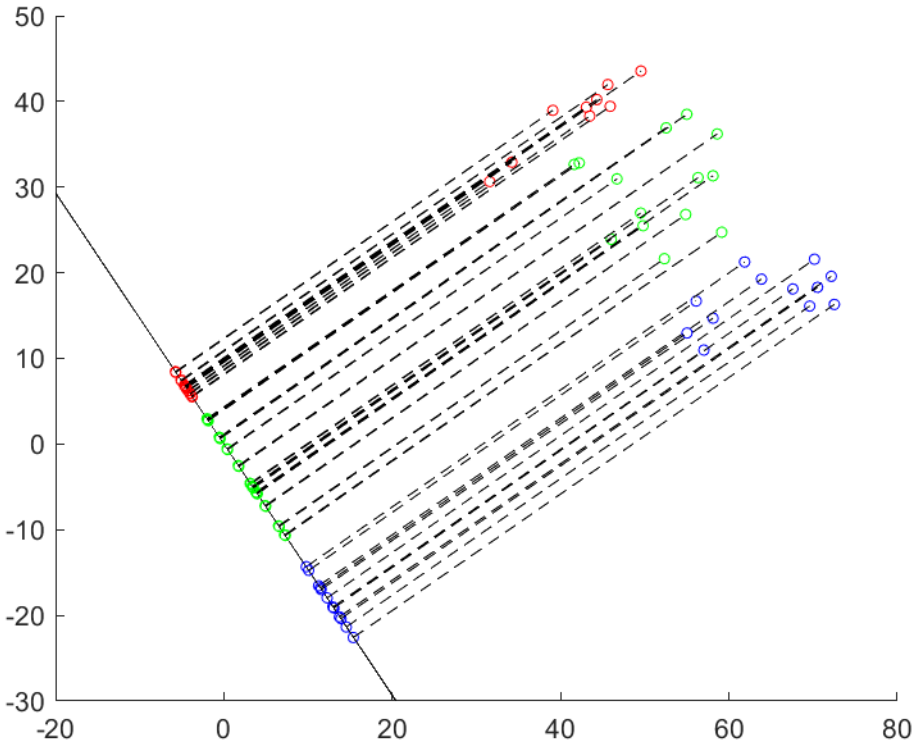

图 16. 投影后的样本分布

从图 16 中我们可以看到，红、绿、蓝三种样本在超平面上的投影之间都有一段距离，也就是有明显的分界。在每一个类中，样本投影尽量做到紧密排列，尤其是红色样本，在图 16 中可以看出排列非常紧密，而蓝色样本和绿色样本尽管看

起来比较相对稀疏，但是很明显可以看到类与类之间的间隔。在实际的分类问题中我们可以用这些间隔来设置分类标准。

6. LDA 分类的实际应用例程

6.1 LDA 对于西瓜数据集的分类：LDA 的弊端

我们采用西瓜数据集3.0α作为测试集，内容如表 1 所示。

Index	Density	Sugar content	Class
1	0.697	0.460	1
2	0.774	0.376	1
3	0.634	0.264	1
4	0.608	0.318	1
5	0.556	0.215	1
6	0.403	0.237	1
7	0.481	0.149	1
8	0.437	0.211	1
9	0.666	0.091	0
10	0.243	0.267	0
11	0.245	0.057	0
12	0.343	0.099	0
13	0.639	0.161	0
14	0.657	0.198	0
15	0.360	0.370	0
16	0.539	0.042	0
17	0.719	0.103	0

表 1. 西瓜数据集3.0α

将西瓜数据集3.0α作为样本空间，绘制散点图如图 17 所示。

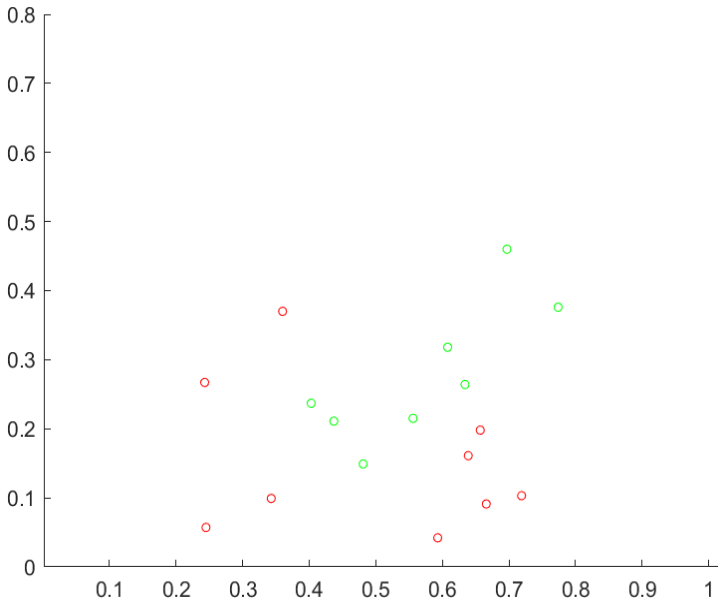

图 17. 西瓜数据集3.0α可视化

这是一个二分类问题。我们调用之前的程序，得到如图 18 所示的投影结果。

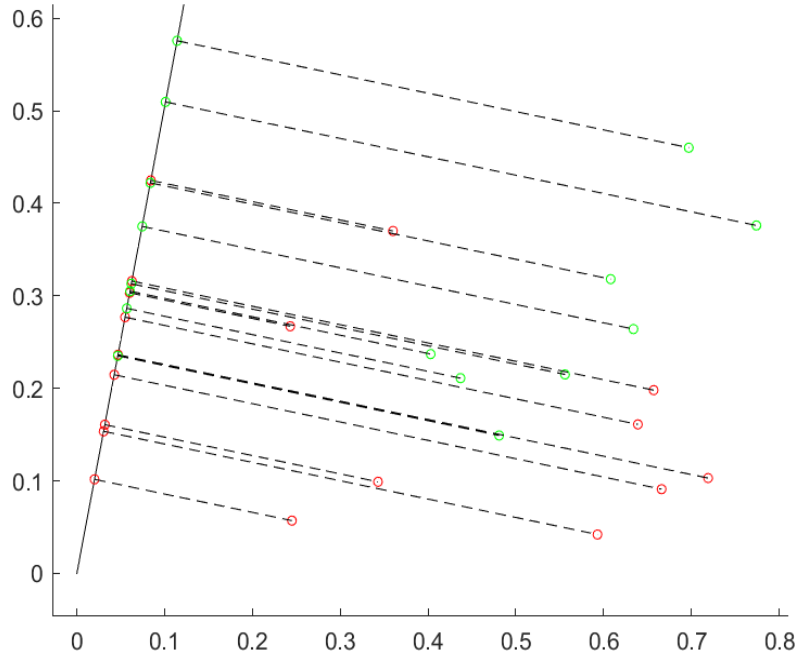

图 18. 西瓜数据集3.0α的投影结果

从图 16 中我，我们可以明显地看到投影已经出现了混杂的情况。事实上，**线性判别分析仅在线性可分（即存在线性超平面能将不同的样本点分开）数据集上能取得一个理想的结果**。在非线性问题中，一般情况下使用带有核函数的线性判别分析来解决非线性判别分析。

6.2 LDA 对三维空间线性可分情况下的分类

我们选取一组三维空间的样本，将其可视化之后如图 19 所示。

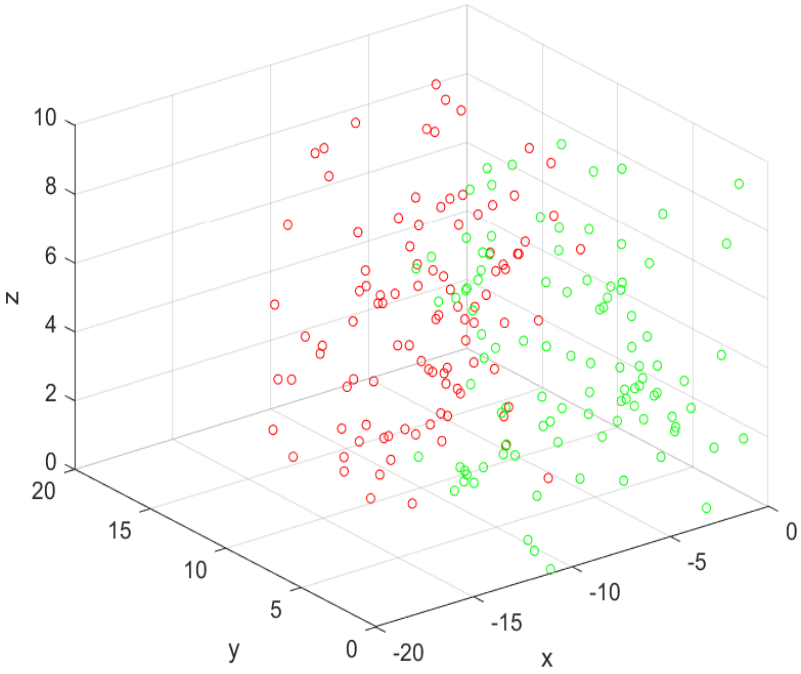

图 19. 三维空间样本点

我们用 LDA 的方法将其投影到二维空间，此时$\boldsymbol{\omega}$应该是一个3×2的矩阵，因为

我们要将三维空间的样本点降维到二维。此时二维空间投影点分布如图 20 所示。

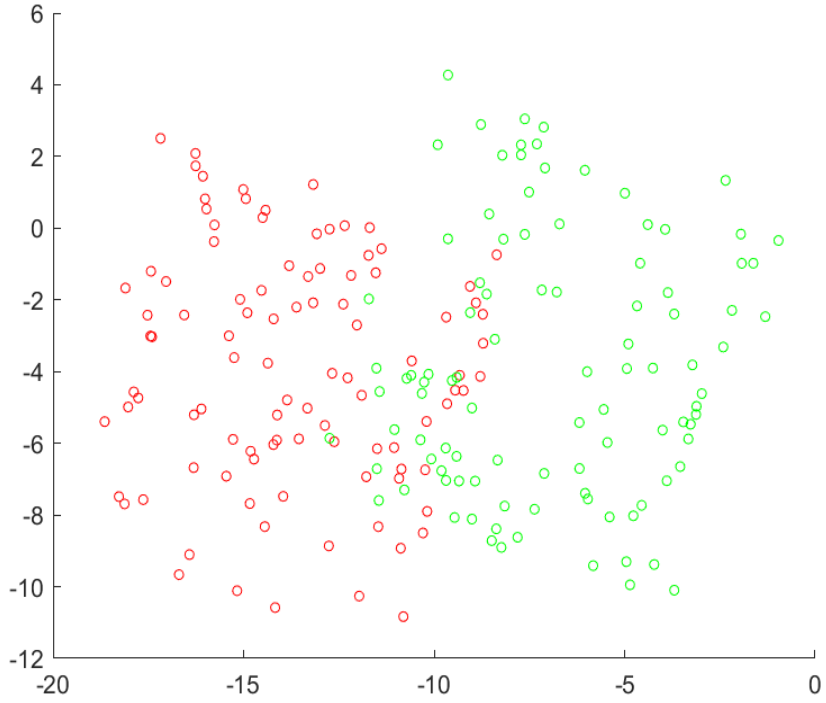

图 20. 二维空间的投影点

从图 20 中可以看到，红色和绿色的样本点之间的边界较为明显，所以这次 LDA 分类是比较理想的。那么在降维之后，我们可以进行分类然后还原到原来的维度，从而实现样本的分类。我们采取 Logistic 回归来进行分类，得到了分类边界如图 21 所示。

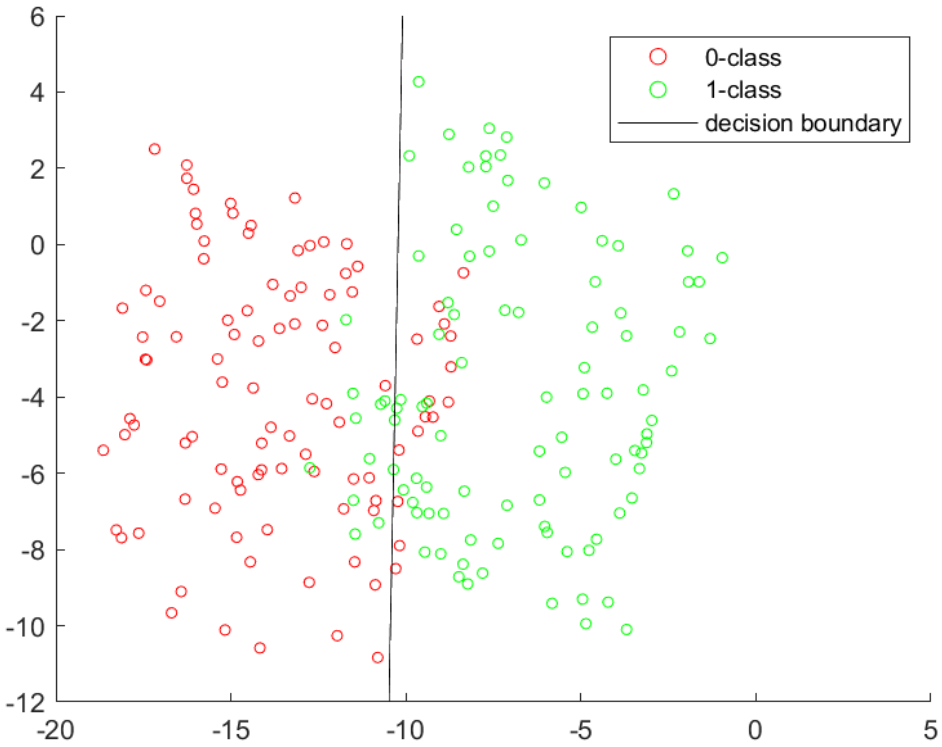

图 21. 二维空间投影点的逻辑回归结果

现在我们有了决策边界$\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}$，那么如果新来一个测试样本$\boldsymbol{x}^{(i)}$，我们先将其降维

到二维空间，投影为$\boldsymbol{z}^{(i)}$，然后我们只需计算$g(\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{z}^{(i)})$，按照逻辑回归的思路，$g(\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{z}^{(i)})>0.5$则为正类，$g(\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{z}^{(i)})<0.5$则为负类，这也体现了 LDA 的优势之一，即完全可以一一对应。代码如图 22 所示。

```
% test sample, sample_x
sample_x = [x;y;z];

% reduce the dimension first
X_projection = omega' * sample_x;

% adopt Logistic Regression to give a prediction
X_projection = [1;X_projection];
res = theta' * X_projection;
if res > 0
    fprintf('This is a sample belongs to class 1.\n');
else
    fprintf('This is a sample belongs to class 0.\n');
end
```

图 22. 结合 Logistic 回归和 LDA 来进行预测

接下来我们用新的正例和负例各 10 组来进行验证，准确率为 85%。这个例程的意义在于说明了投影之后分类的过程。

7. 总结

回顾本次实验，我们可以得出以下结论：

1) LDA 作为一种降维方法，应用于监督学习的使用。它和 PCA 的区别在于，PCA 是基于无监督学习的，它所作的只是将整组数据整体映射到最方便表示这组数据的坐标轴上，映射时没有利用任何数据内部的分类信息；而 LDA 算法在监督学习的基础之上，尽量使得同一类的样本尽可能地靠近而非同一类的样本尽量远离，从而在降维的前提下达到最好的分类效果。
2) LDA 在降维之后依然可以保留数据标签方便在降维之后来做分类。
   例如，以语音识别为例，如果使用 PCA 进行降维的话，如果单纯用 PCA 降维，则可能功能仅仅是过滤掉了噪声，还是无法很好的区别人声；但是如果用 LDA 进行降维的话，则降维后的数据会使得每个人的声音都具有可分性。
3) LDA 有其弊端。例如，在线性不可分的样本集合中，例如图 17 所示的情况，LDA 的工作便不再理想。在这种情况下，核方法 LDA 可以将其映射到高维空间，即使用特征映射矩阵来进行映射。

   例如，关于样本点$\boldsymbol{x}=[x_1,x_2]^{\mathrm{T}}$，如果映射到二次方，映射之后变为

$$\boldsymbol{x}=[x_1,x_2]\rightarrow\boldsymbol{x}^{\phi}=[x_1,x_2,x_1^2,x_1x_2,x_2^2]^{\mathrm{T}}$$

   这样提高了矩阵的维度，也就可以实现非线性超平面来划分数据。

四、 实验总结与体会

本次实验推导了 LDA 算法的数学原理，并通过 MATLAB 语言实现了 LDA 算法。LDA 算法是应用非常广泛的非监督学习降维技术，因此非常值得学习。本次实验还总结了 LDA 的优势与弊端。关于 LDA 算法还有很多新的拓展，像改善分类效果的核方法 LDA 等等。

五、 实验附件

1. 线性判别分析二维情况运行lda.m；
2. 线性判别分析二维下多分类问题运行ldamulti.m；
3. 线性判别分析实践一：ldapractice.m；
4. 线性判别分析实践二：ldathreedimensions.m 。