** **

贝叶斯分类器设计

|  |  |
| --- | --- |
| 院（系）名称 | 自动化科学与电气工程学院 |
| 专业名称 | 模式识别与智能系统 |
| 学生学号 | 15031184 |
| 学生姓名 | 李思奇 |

2018年5月2日

# 1. 实验原理

## 1.1 实验背景

贝叶斯分类器是一种从一定的概率模型出发，根据各类特征的概率模型来估算后验概率并做出决策的分类器。实质是将模式识别的问题转化成了概率模型估计的问题。在许多场合，贝叶斯分类器可以与决策树和神经网络分类算法相媲美，该算法能运用到大型数据库中，而且方法简单、分类准确率高、速度快。

## 1.2 理论推导

根据概率论中的贝叶斯公式，我们知道：

 (1-1)

其中，为先验概率，是x与的联合概率密度函数，是两类所有样本的概率密度，是第i类样本的概率密度（类条件密度）。

对于最小错误率贝叶斯决策，我们希望尽量减少分类中的错误，即目标是追求最小错误率。根据定义，最小错误率决策是求解一种决策规则，使得错误率最小化，即：

 (1-2)

由于对于所有的，有，所以上式等价于对于所有最小化。因此，对于两类问题，得到最小错误率决策如下：

如果，则，反之，则

若引入损失函数，则可得出最小风险贝叶斯决策。设损失函数为样本应属于类而被判为时引起的损失。则对于某个样本，它属于各个状态的后验概率为，对它采取决策的期望损失是：

 (1-3)

则，最小风险贝叶斯决策为：

 (1-4)

## 1.3 决策步骤

①利用贝叶斯公式计算后验概率

 (1-5)

②利用损失函数，计算条件风险

 (1-6)

③决策：在各种决策中选择风险最小的决策，即

 (1-7)

# 2. 基础实验

## 2.1 类条件概率计算

给定两类样本集：



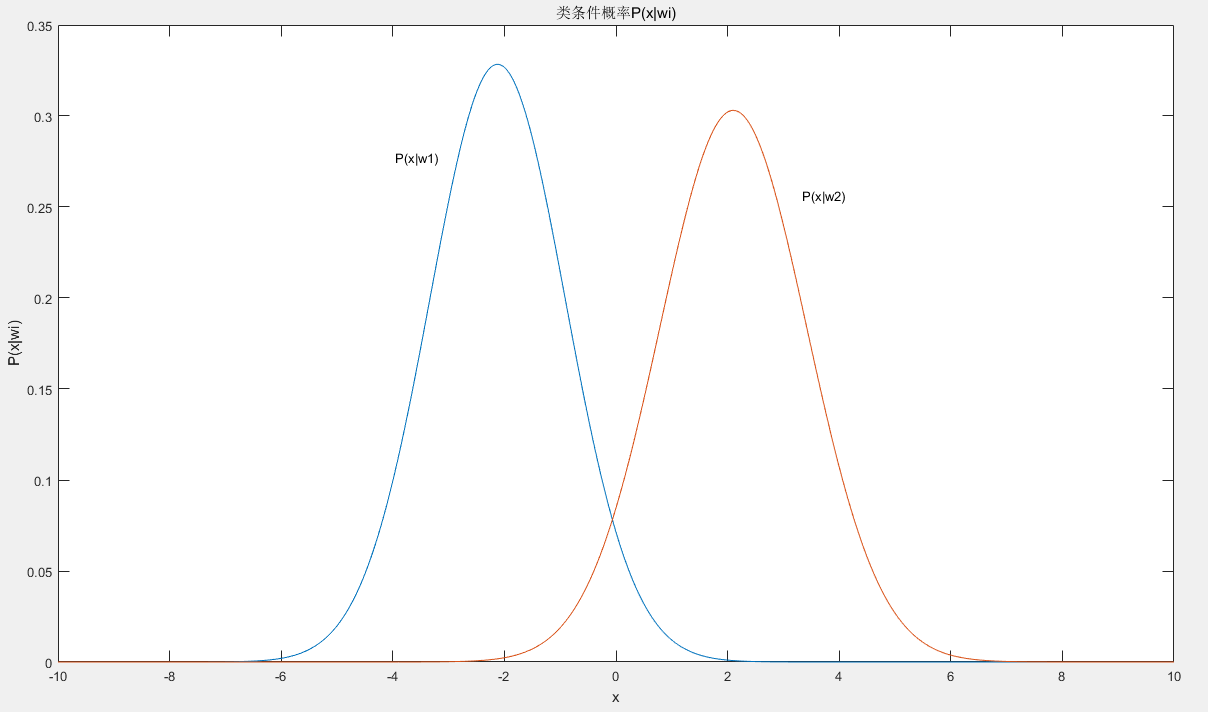


已知两组数据均满足正态分布，根据概率论，应有：

 (2-1)

 (2-2)

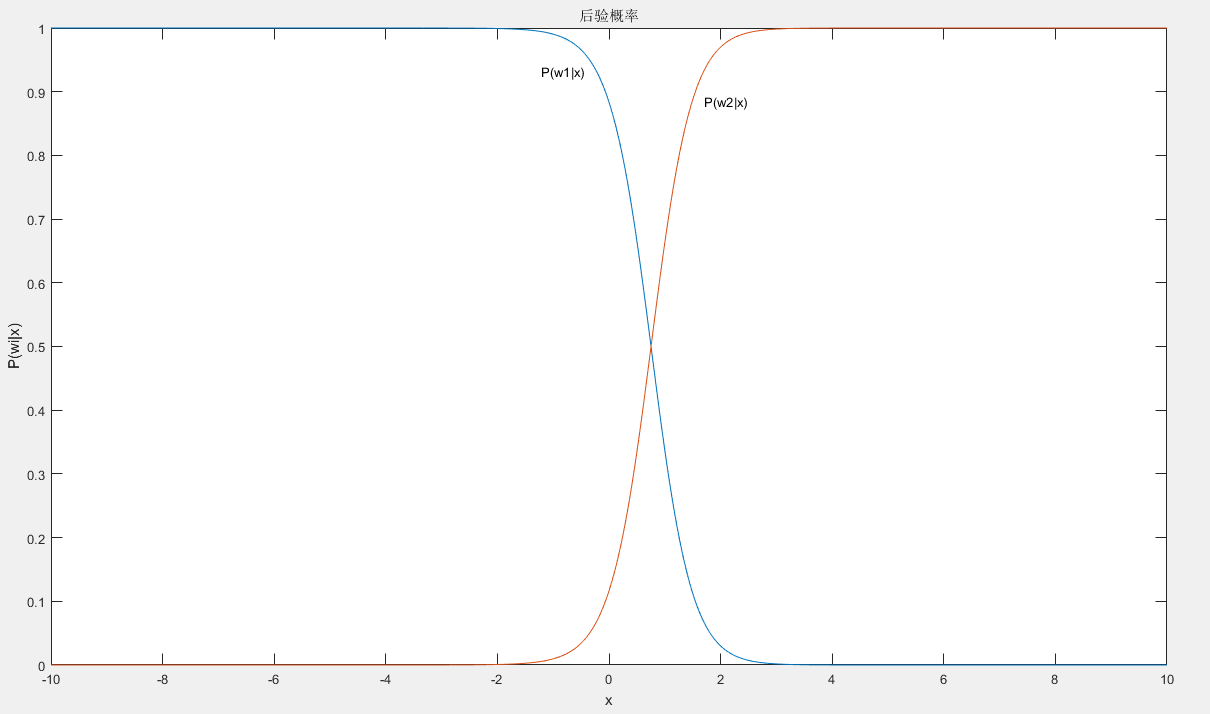
根据式(2-1)和式(2-2)，求出对应的和，并做出类条件概率密度图如图2.1所示。



**图2.1 类条件概率密度**

## 2.2 后验概率

根据式(1-5)，可求出两类的类条件概率密度函数，如图2.2.1所示。



**图2.2.1 后验概率**

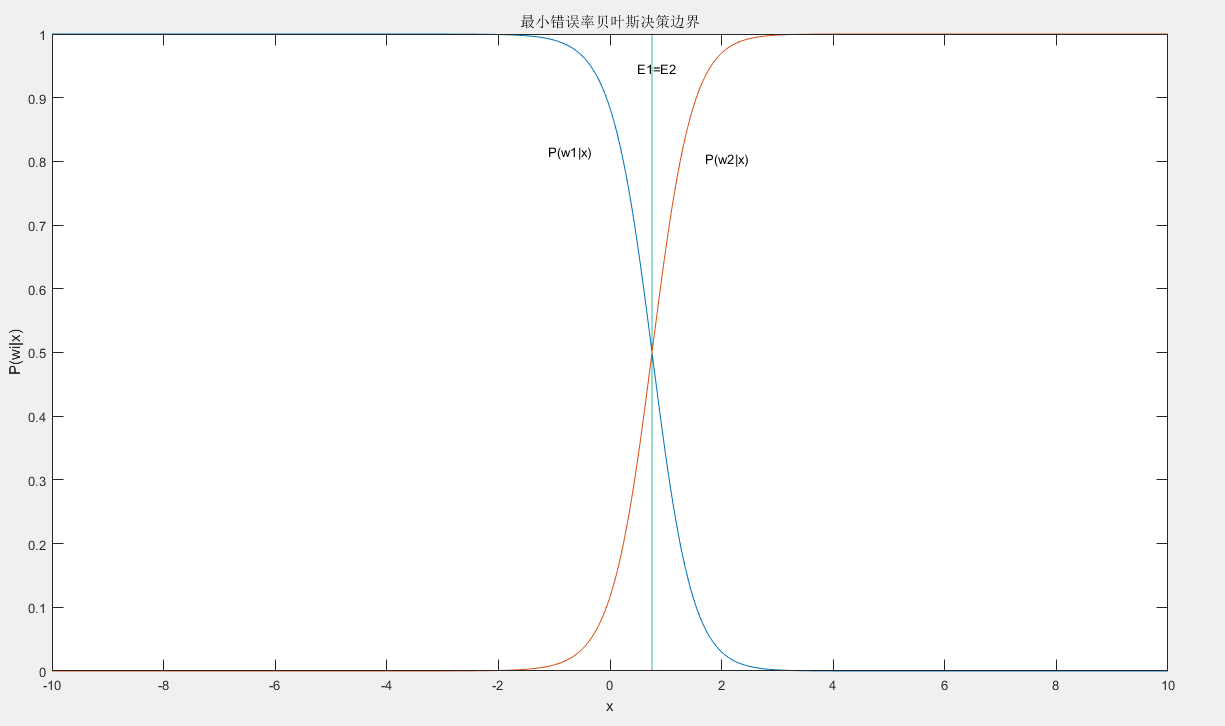
若不考虑损失函数，即采用最小错误率贝叶斯决策，可求出决策边界方程为：



即：



做出最小错误率贝叶斯决策边界如图2.2.2所示。



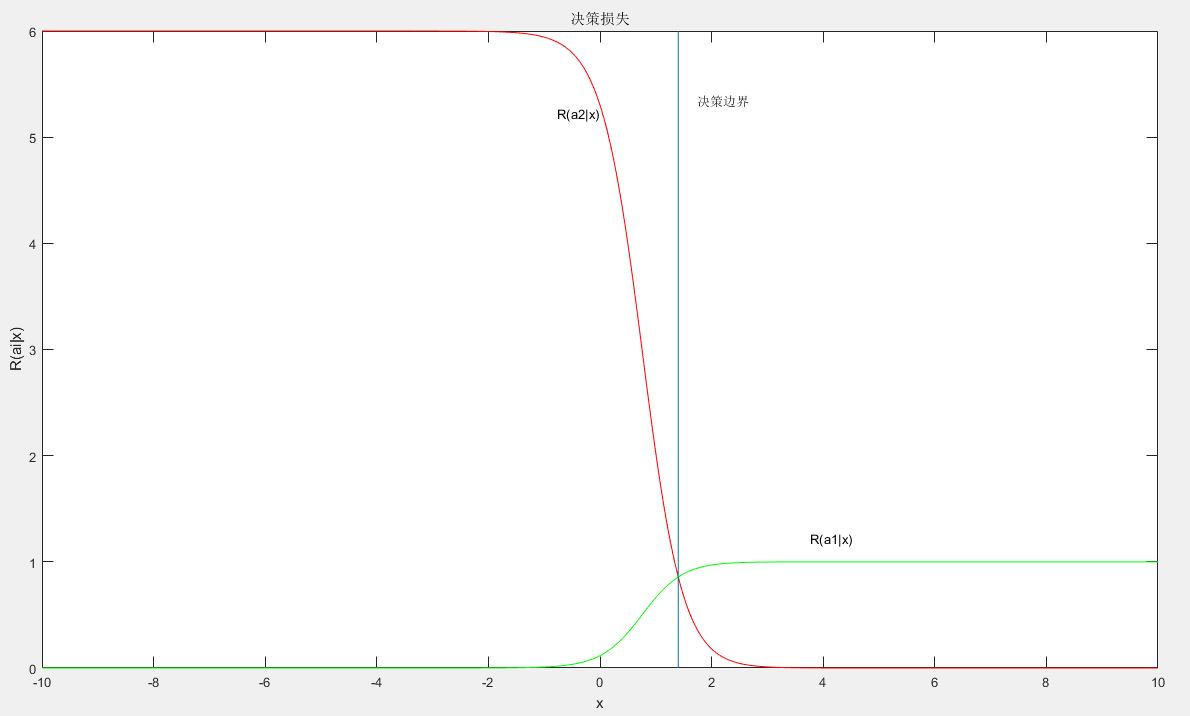
**图2.2.2 最小错误率贝叶斯决策边界**

## 2.3 最小风险贝叶斯决策

由式(1-6)、式(1-7)可求出当时的x取值。决策边界方程为：



做出最小风险贝叶斯决策的决策边界如图2.3所示。



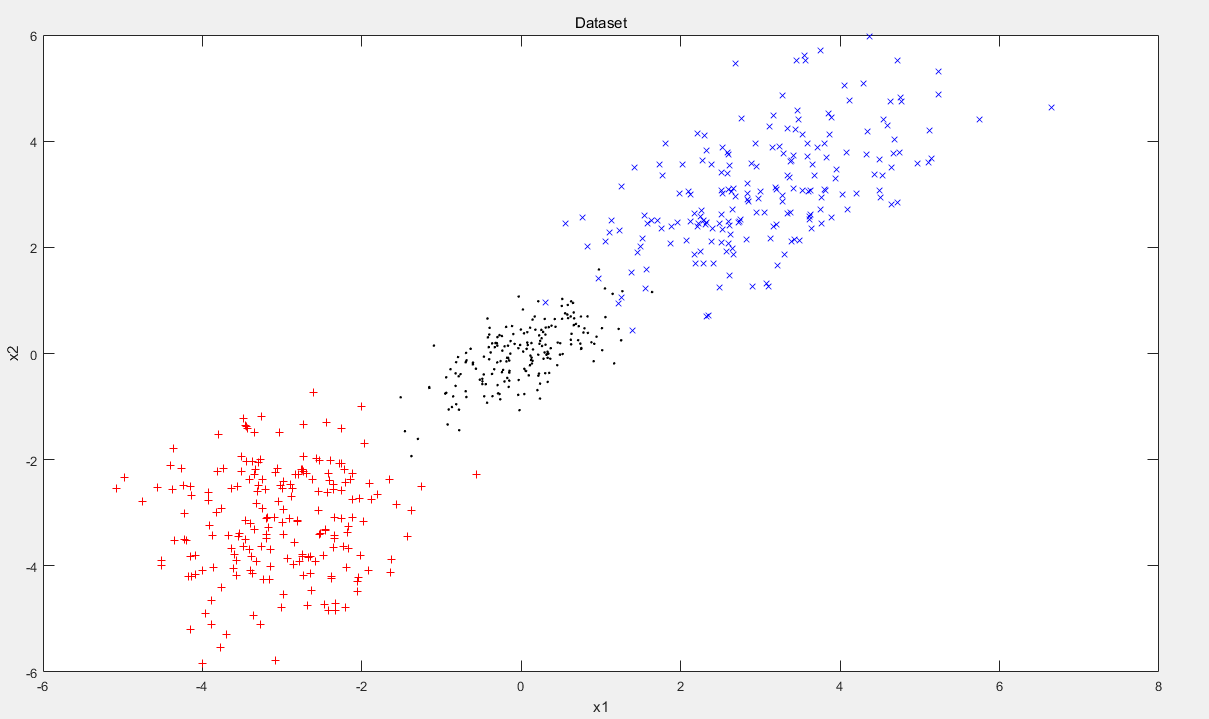
**图2.3最小风险贝叶斯决策边界**

对比图2.2.2与图2.3可以发现，最小风险贝叶斯决策的决策边界比最小错误率的决策边界的x值要更大。这是因为引入损失函数后，将类的样本错分为类时，要付出更大的代价。因此，即使类的后验概率要略大于类的后验概率，也会将样本判定为类。

# 3. 扩大类别数和特征维度

## 3.1 类条件概率

扩大类别数目及特征维度。为便于可视化，构造样本集，类别数为3类、样本数为600例（每类200例样本）、样本特征维度为2维。样本集中样本分布如图3.1.1所示。



**图3.1.1 样本分布**

考虑多元正态分布的数学模型。其概率密度函数为：

 (3-1)

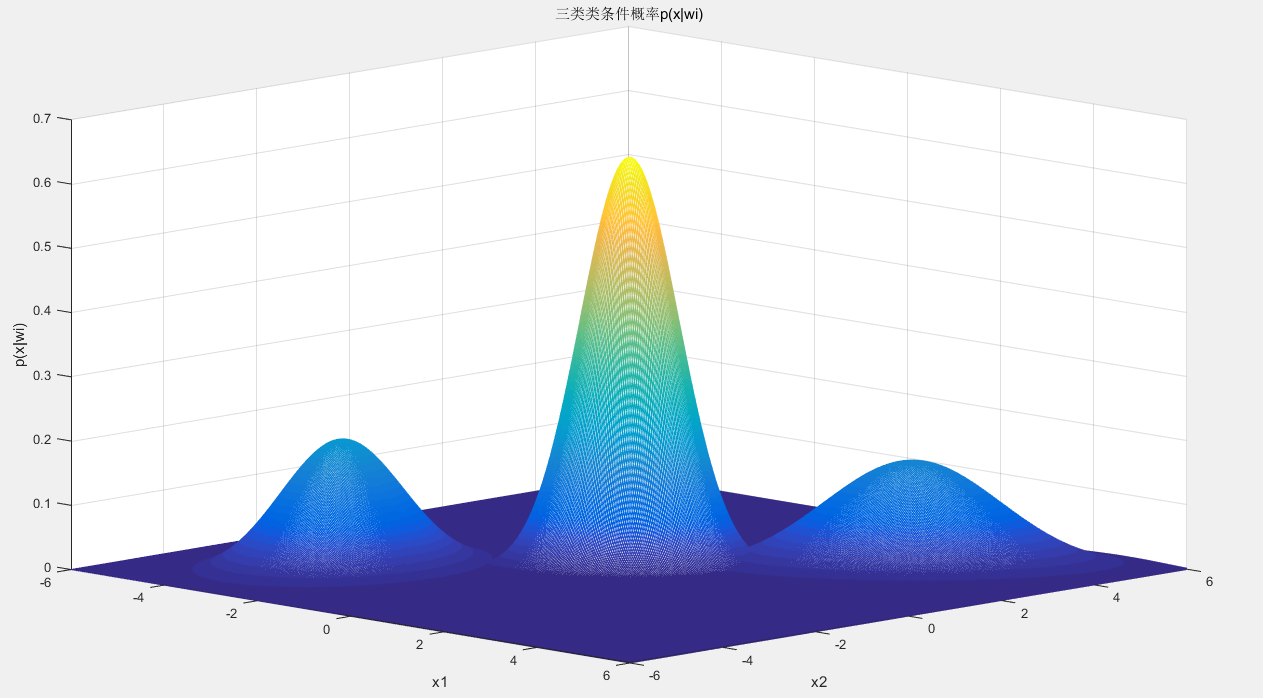
式中：是d维均指向量。是维协方差矩阵。是的逆矩阵，是的行列式。

根据数学推导，有：

 (3-2)

 (3-3)

根据式(3-2)和式(3-3)，可以求出三类样本各自的、：并做出各自的类条件概率如图3.1.2所示。

****

**图3.1.2 三类类条件概率**

其中：





且有第i类类条件概率为：

## 

## 3.2 后验概率

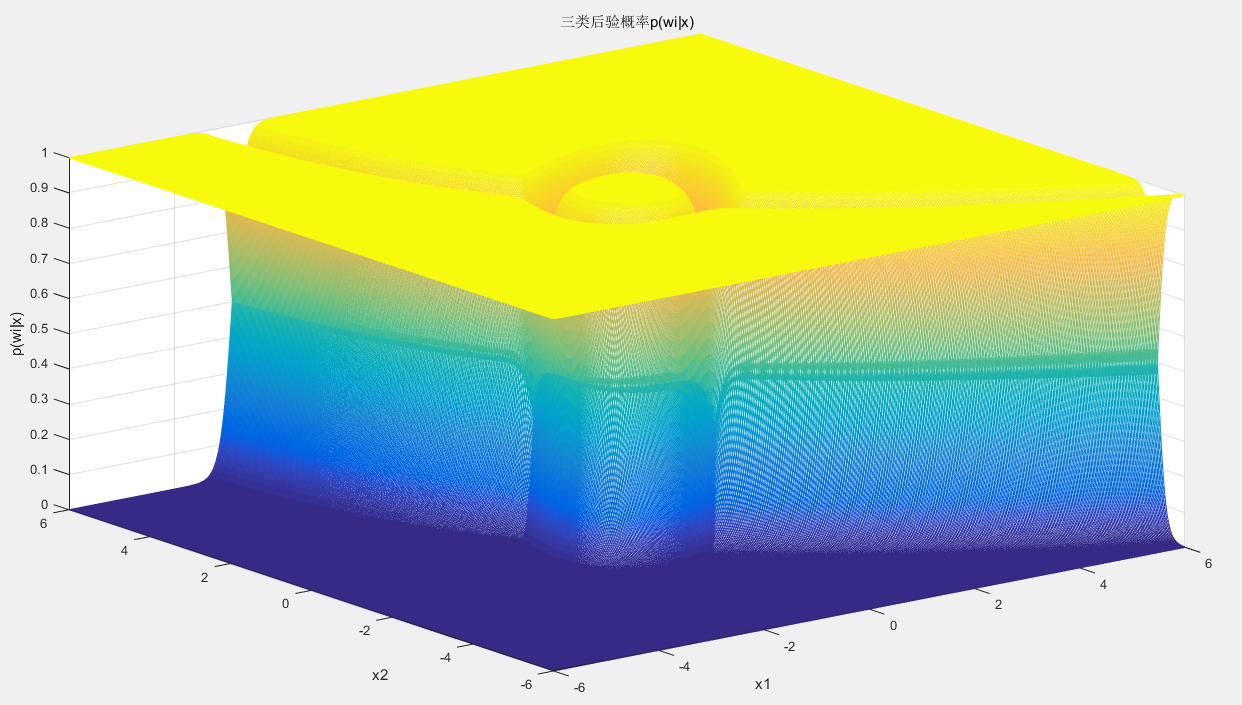
假设三类的先验概率为：



则由式(1-5)可求出第i类的后验概率为：



做出各类后验概率如图3.2.1所示。

****

**图3.2.1三类后验概率**

## 3.3 最小风险贝叶斯决策

设损失函数的决策表如表3.3.1所示。

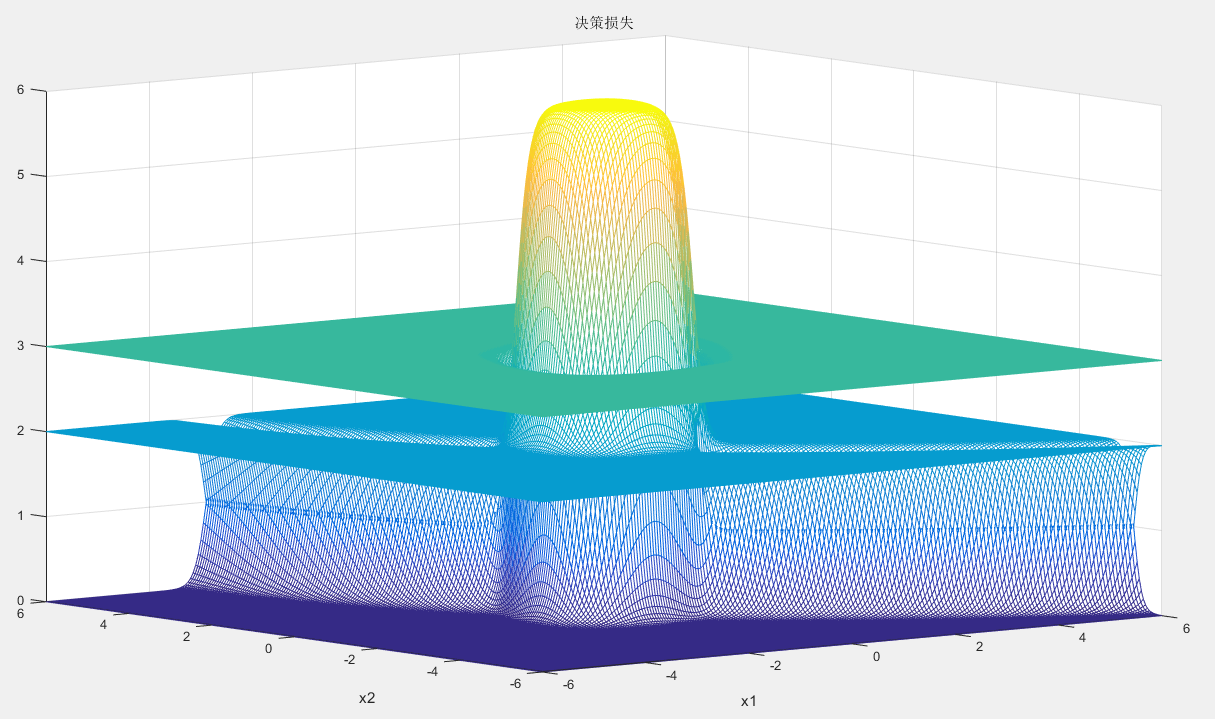
**表3.3.1 损失函数的决策表**

|  |  |  |  |
| --- | --- | --- | --- |
| 决策 | 自然状态 | | |
| ω1 | ω2 | ω3 |
| α1 | 0 | 2 | 18 |
| α2 | 3 | 0 | 3 |
| α3 | 8 | 6 | 0 |

则，对于某个样本，对它采取决策α1，α2，α3的期望损失为：

 (3-4)

根据式(3-4)做出决策损失图，如图3.3.1所示。

****

**图3.3.1 决策损失**

通过比较三种决策的期望损失，可以得出分类决策：

 (3-5)

# 3.4 最小错误率决策边界与最小风险决策边界

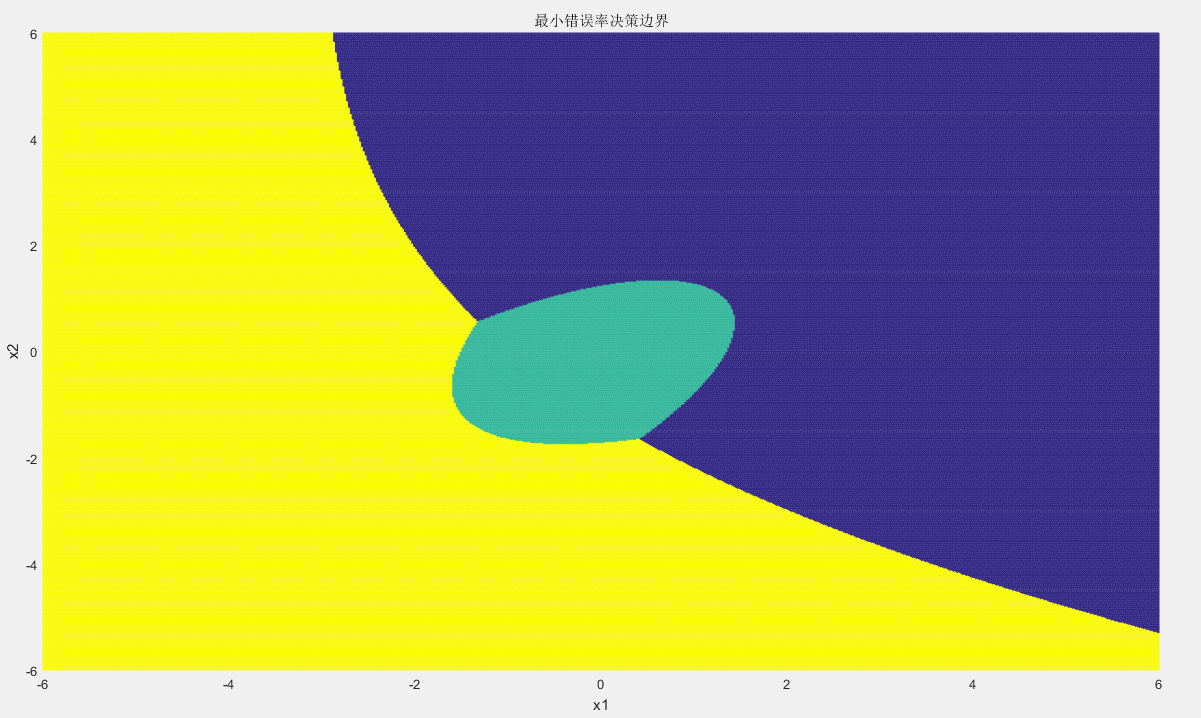
由图3.2.2以及最小错误率决策：



可以得出，最小错误率决策的决策边界方程为：



做出决策边界曲线，即是图3.2.2中各类后验概率相交的曲线，如图3.4.1所示。

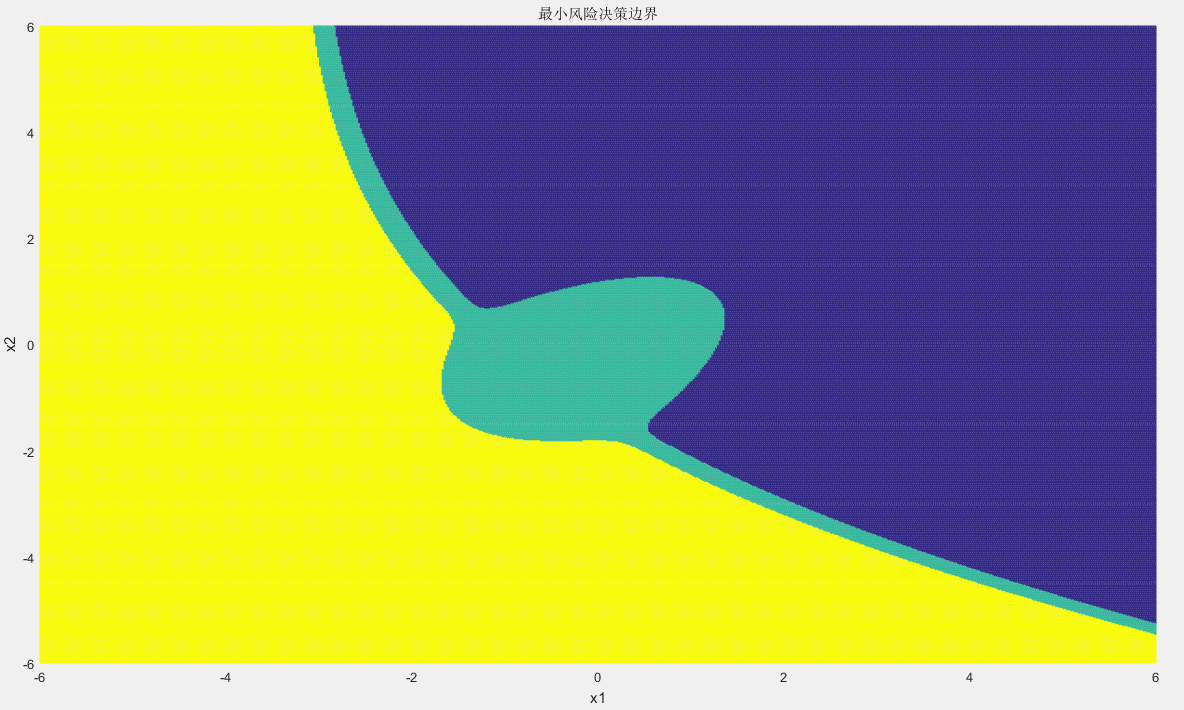
****

**图3.4.1最小错误率决策边界**

由式(3-4)和式(3-5)可以得出最小风险贝叶斯决策的决策边界方程为：



做出决策边界曲线，如图3.4.3所示。



**图3.4.3 最小风险决策边界**

可以看出，加入损失函数****之后，利用最小风险贝叶斯决策做出的决策边界略有变化。

# 4. 结论分析

由以上实验可以看出，一维与多维、一类与多类的贝叶斯分类器在本质原理上并没有太大差别。当维度很高时，可能设计贝叶斯分类器会变得极为麻烦。

相比于线性分类器，贝叶斯分类器在处理多类问题的时候会显得更加有用。但是当样本的概率模型不确定或不是很清楚的时候，应当谨慎使用或避免使用贝叶斯分类器。

# 附录：MATLAB程序代码

clc;clear;n=200;

X1 = load('data1.txt');

X2 = load('data2.txt');

X3 = load('data3.txt');

figure(1);plot(X1(:,1),X1(:,2),'bx',X2(:,1),X2(:,2),'k.',X3(:,1),X3(:,2),'r+');

xlabel('x1');ylabel('x2');title('Dataset');

X1=X1';X2=X2';X3=X3';

[mu1,sigma1]=get\_mu\_sigma(X1,n);

[mu2,sigma2]=get\_mu\_sigma(X2,n);

[mu3,sigma3]=get\_mu\_sigma(X3,n);

p1=0.35;p2=0.15;p3=0.5;

[x,y]=meshgrid(-6:0.02:6,-6:0.02:6);

P1 = pxy(x,y,mu1,sigma1);

P2 = pxy(x,y,mu2,sigma2);

P3 = pxy(x,y,mu3,sigma3);

figure(2);mesh(x,y,P1);hold on;mesh(x,y,P2);hold on;mesh(x,y,P3);

xlabel('x1');ylabel('x2');zlabel('p(x|wi)');

title('三类类条件概率p(x|wi)');

P1=P1\*p1;P2=P2\*p2;P3=P3\*p3;Pall=P1+P2+P3;

P1=P1./Pall; P2=P2./Pall; P3=P3./Pall;

figure(3);mesh(x,y,P1);hold on;mesh(x,y,P2);hold on;mesh(x,y,P3);hold on;

xlabel('x1');ylabel('x2');zlabel('p(wi|x)');title('三类后验概率p(wi|x)');

region = cregion(x, P1, P2, P3, '1');

figure(4);mesh(x, y, region);

xlabel('x1');ylabel('x2');title('最小错误率决策边界');

r1 = 2\*P2 + 8\*P3; r2 = 3\*P1 + 3\*P3; r3 = 6\*P2 + 18\*P1;

figure(5);mesh(x, y, r1);hold on;mesh(x, y, r2);mesh(x, y, r3);

xlabel('x1');ylabel('x2');title('决策损失');

region = cregion(x, r1, r2, r3, '2');

figure(6);mesh(x, y, region);

title('最小风险决策边界');xlabel('x1');ylabel('x2');

function [mu,sigma]=get\_mu\_sigma(X,n)

mu=0;

for i=1:n

mu = mu + X(:,i);

end

mu = mu ./ n;

sigma = [0 0;0 0];

for i=1:n

sigma = sigma + (X(:,i)-mu)\*(X(:,i)-mu)' ;

end

sigma = sigma ./ n;

end

function Z = pxy(X,Y,mu,sigma)

DX=sigma(1,1); dx=sqrt(DX);

DY=sigma(2,2); dy=sqrt(DY);

COV=sigma(1,2); r=COV/(dx\*dy);

part1=1/(2\*pi\*dx\*dy\*sqrt(1-r^2));

p1=-1/(2\*(1-r^2));

px=(X-mu(1)).^2./DX;

py=(Y-mu(2)).^2./DY;

pxy=2\*r.\*(X-mu(1)).\*(Y-mu(2))./(dx\*dy);

Z=part1\*exp(p1\*(px-pxy+py));

end

function region = cregion(x, p1, p2, p3, op)

assert(nargin == 5);

region = zeros(size(p1));

for i = 1 : size(x, 1)

for j = 1 : size(x, 2)

if(strcmp(op, '1'))

[~, loc] = max([p1(i, j), p2(i, j), p3(i, j)]);

elseif(strcmp(op, '2'))

[~, loc] = min([p1(i, j), p2(i, j), p3(i, j)]);

end

region(i, j) = loc - 1;

end

end

end