北京航空航天大學
BEIHANG UNIVERSITY

Pattern Recognition and Machine Learning Experiment Report

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 15031232

学生姓名 刘霜婷

2017 年 5 月 1 日

2 Synthetical Design of Bayesian Classifier

2.1 Introduction

Linear perceptrons allow us to learn a decision boundary that would separate two classes. They are very effective when there are only two classes, and they are well separated. Such classifiers are referred to as discriminative classifiers. In contrast, generative classifiers consider each sample as a random feature vector, and explicitly model each class by their distribution or density functions. To carry out the classification, the likelihood function should be computed for a given sample which belongs to one of candidate classes so as to assign the sample to the class that is most likely. In other words, we need to compute $p(w_i|x)$ for each class w_i. However, the density functions provide only the likelihood of seeing a particular sample, given that the sample belongs to a specific class. i.e., the density functions can be provided as $p(x|w_i)$. The Bayesian rule provides us with an approach to compute the likelihood of the class for a given sample, from the density functions and related information.

2.2 Principle and Theory

The essence of the Bayesian approach is to provide a mathematical rule explaining how you should change your existing beliefs in the light of new evidence. In other words, it allows us to combine new data with their existing knowledge or expertise. The canonical example is to imagine that a precocious newborn observes his first sunset, and wonders whether the sun will rise again or not. He assigns equal prior probabilities to both possible outcomes, and represents this by placing one white and one black marble into a bag. The following day, when the sun rises, the child places another white marble in the bag. The probability that a marble plucked randomly from the bag will be white (i.e., the child's degree of belief in future sunrises) has thus gone from a half to two-thirds. After sunrise the next day, the child adds another white marble, and the probability (and thus the degree of belief) goes from two-thirds to three-quarters. And so on. Gradually, the initial belief that the sun is just as likely as not to rise each morning is modified to become a near-certainty that the sun will always rise.

In terms of classification, the Bayesian theorem allows us to combine prior probabilities, along with observed evidence to arrive at the posterior probability. More or less, conditional probabilities represent the probability of an event occurring given evidence. According to the Bayesian Theorem, if $p(w_i)$, $p(X|w_i)$, i=1, 2, ..., c, and X are known or given, the posterior probability can be derived as follows

$$P(\omega_i|X)=\frac{P(X|\omega_i)P(\omega_i)}{\sum_{j=1}^{c}P(X|\omega_j)P(\omega_j)} \qquad \text{i=1,…, c} \qquad (1)$$

Let the series of decision actions as $\{a_1, a_2, \dots, a_c\}$, the conditional risk of decision action a can be computed by

$$R(a_i|X)=\sum_{j=1,j\neq i}^{c}\lambda(a_i,\omega_j)P(\omega_j|X), \qquad \text{i=1,…, c} \qquad (2)$$

Thus the minimum risk Bayesian decision can be found as

$$a_k^* = \text{Arg min}_i\ R(a_i|X), \qquad \text{i=1,…, c} \qquad (3)$$

2.3 Objective

The goals of the experiment are as follows:

. (1) To understand the computation of likelihood of a class, given a sample.

. (2) To understand the use of density/distribution functions to model a class.

. (3) To understand the effect of prior probabilities in Bayesian classification.

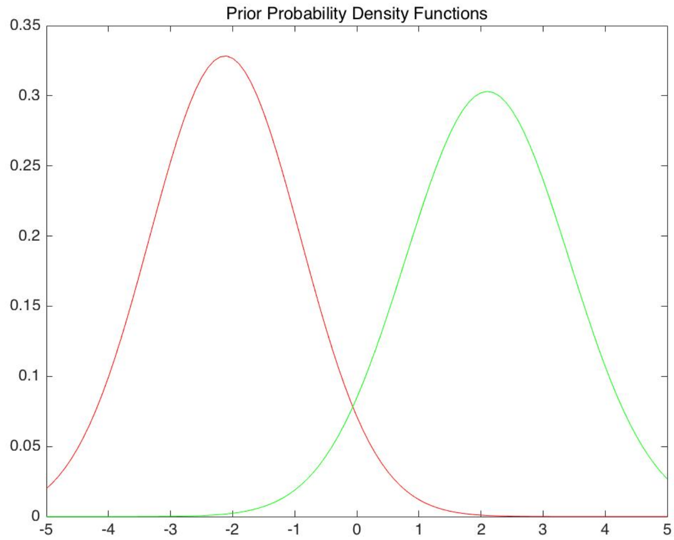
. (4) To understand how two (or more) density functions interact in the feature space to decide a decision boundary between classes.

. (5) To understand how the decision boundary varies based on the nature of density functions.

2.4 Contents and Procedures

Stage1

（1） In stage 1, we first calculate prior and posterior probability density functions and draw the curves. Because the conditional probability distributions are Gaussian, we can calculate the average value and variance of w_1 and w_2, then use the formula of Gaussian Distribution that $f(x)=\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$, we can obtain the prior probability density function $p(X|w_i)$, i=1, 2, and then draw the curve. The result is as follows (the red curve represents w_1 while the green curve represents w_2):



To calculate posterior probability density function, we bring data $P(w_1)$=0.9 and $P(w_2)$＝0.1 into the formula $P(\omega_i|X) = \frac{P(X|\omega_i)P(\omega_i)}{\sum_{j=1}^{c} P(X|\omega_j)P(\omega_j)}$, i=1,2, and get the result of $P(w_1|X)$ and $P(w_2|X)$. The curve is as follows (the red curve represents w_1 while the green curve represents w_2):

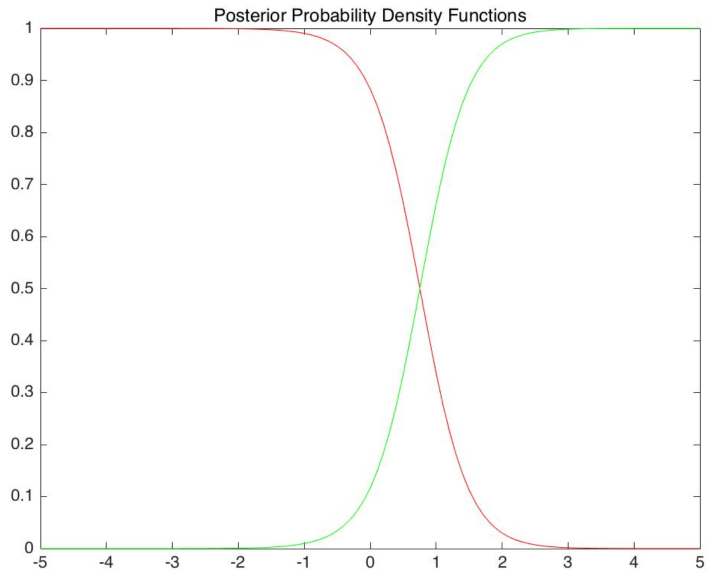
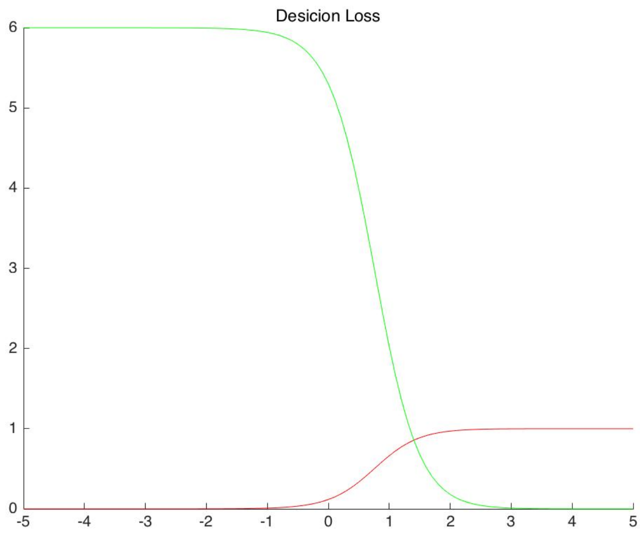


Table 1　the loss parameters for different decision

real class		ω_1	ω_2	
decision action	a_1	0	1	loss parameters
	a_2	6	0	

Considering decision loss, the result is as follows (the red curve represents w_1 while the green curve represents w_2):



（2） Complete the design of Bayesian classifier with and without considering decision loss. Calculating by Matlab, we can conclude that a decision boundary of 0.8 guarantees the minimal probability of error, a decision boundary of 1.4 guarantees minimal decision risk, under the loss parameters listed in the table above.
From the result, we can know that the loss parameters for different solutions will influence the classification result to a large extent. In real situations, we should choose appropriate loss parameters in order to make reasonable classification results.

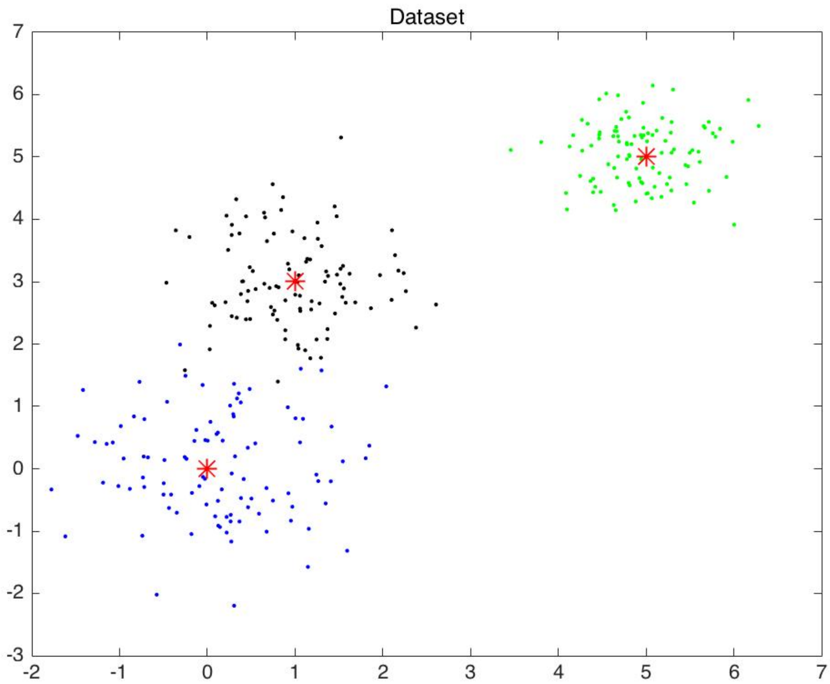
Stage2

In this stage, a dataset with 3 classes in 2-dimentional space is created, and a multi-class and high-dimensional Bayesian classifier is designed.

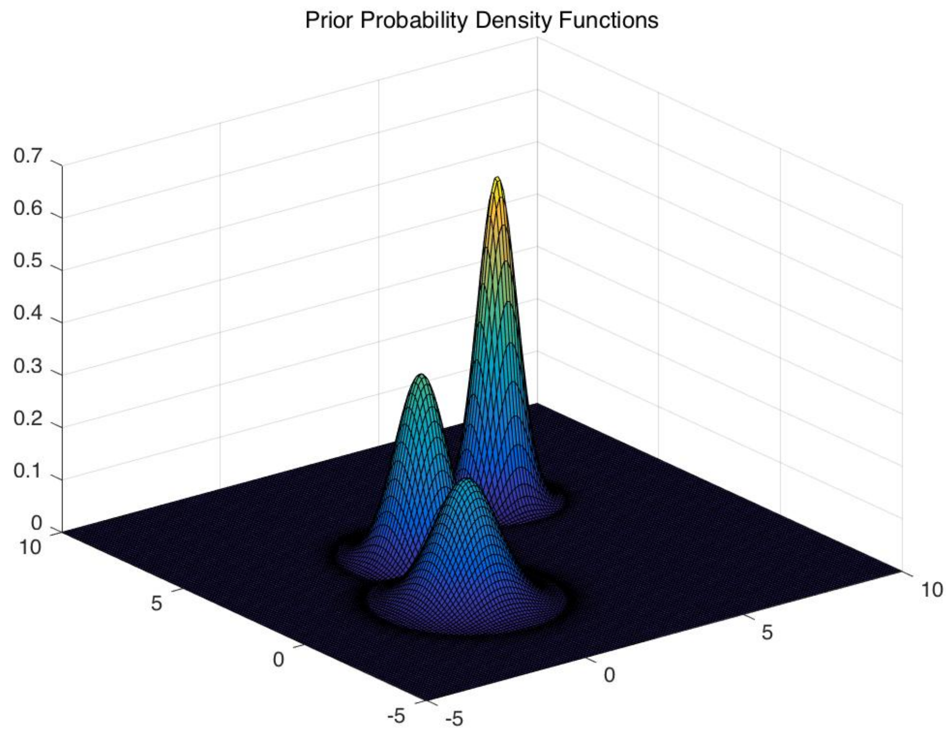
(1) The dataset has 3 classes with 100 samples in each. The conditional probability distribution in each class is 2-dimentional Gaussian distribution. We set the the average value and covariance matrix of data as the table below.

	Average value	covariance matrix	prior probability	
Class1	[0 0]	$\begin{bmatrix} 0.7 & 0 \\ 0 & 0.7 \end{bmatrix}$	0.3	
Class2	[5 5]	$\begin{bmatrix} 0.3 & 0 \\ 0 & 0.3 \end{bmatrix}$	0.1	
Class3	[1 3]	$\begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$	0.6	

The figure of dataset is shown as follows:



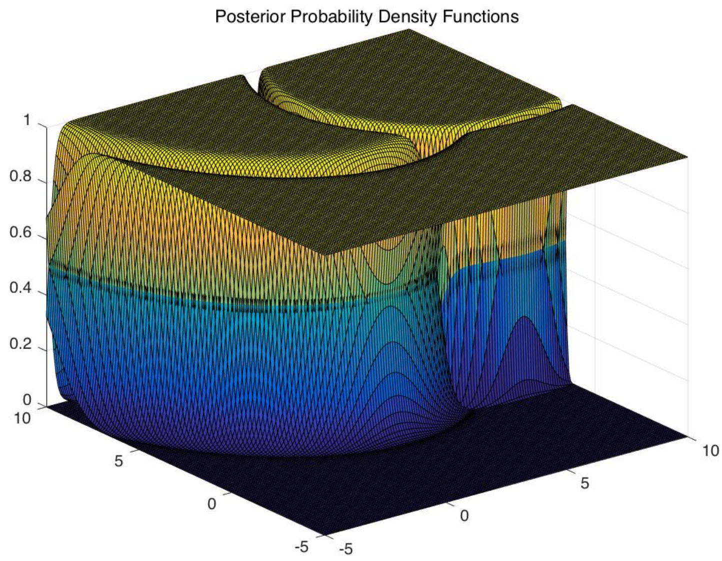
(2) Calculating the mean values and the covariance matrixes of each class of data, then we can obtain two-dimensional normal distribution graph, which represents prior probability density functions.



Bring data $P(w_1)$=0.3 and $P(w_2)$＝0.1, $P(w_2)$＝0.6 into the formula

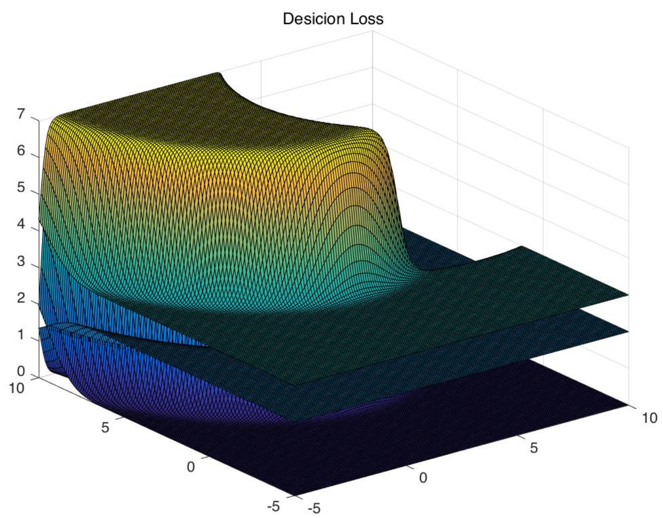
$$P(\omega_i|X)=\frac{P(X|\omega_i)P(\omega_i)}{\sum_{j=1}^{c}P(X|\omega_j)P(\omega_j)}$$

, i=1,2,3. Then posterior probability density function is obtained. The visualization result is shown as follows:

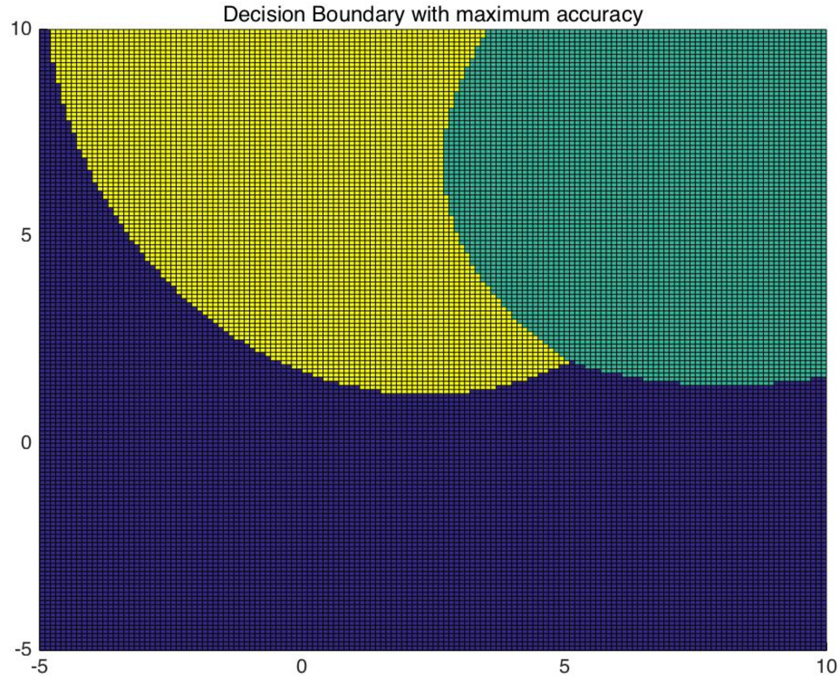


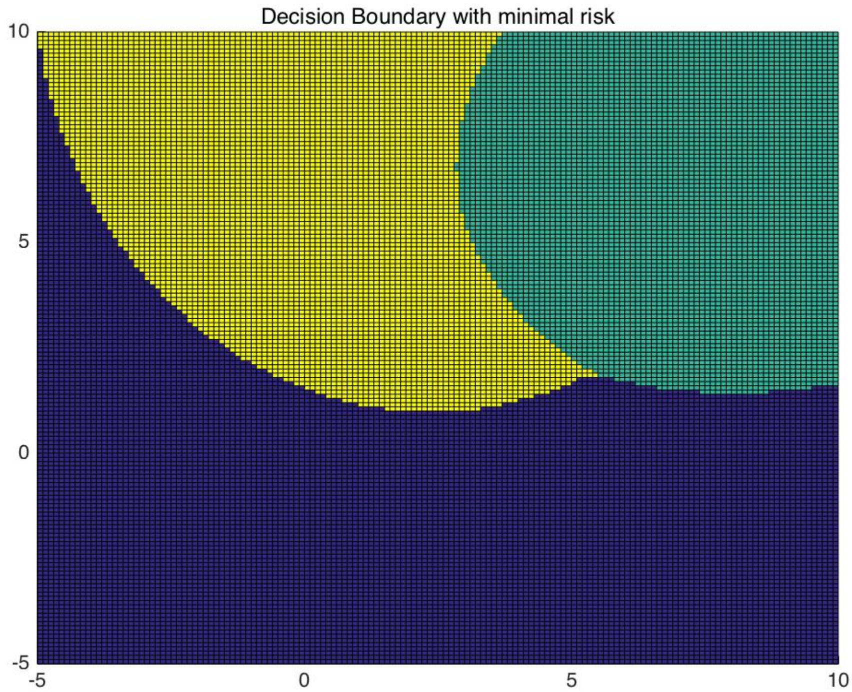
Consider the decision loss as table below, then we can obtain the Bayesian classifier with minimum risk.

Real class		w1	w2	w3	
Decision action	a1	0	2	6	Loss parameters
	a2	3	0	7	
	a3	2	1	0	



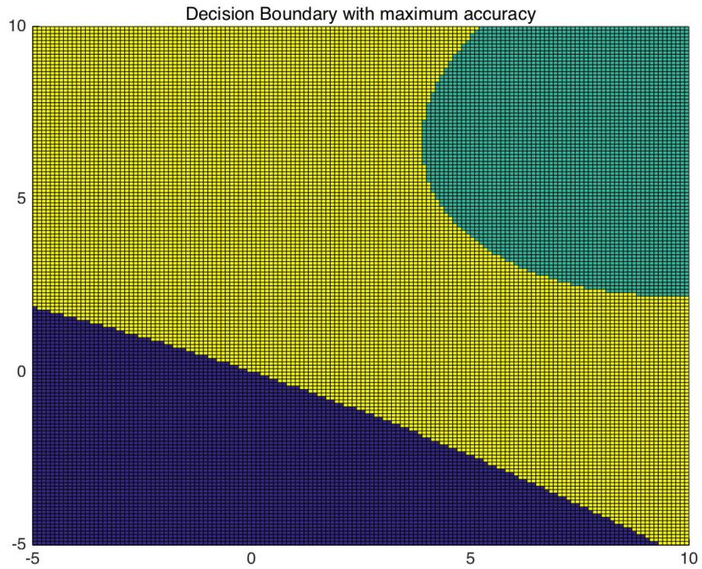
(3) The decision boundary with maximum accuracy and with minimal risk are obtained with Matlab.

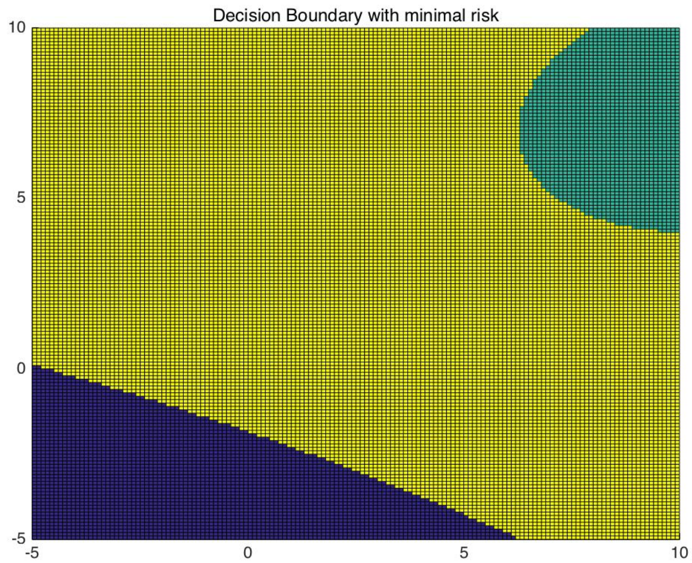




As shown in the figures above, decision boundary just changes a little when considering decision loss. I think this is related to the variance of data. The variance of data is small,

so that the boundary in 3-D graph will change little when the posterior probability density function multiple the loss parameters.
When set variances of 3 classes as 7,3, and 5 respectively, the results are as follows. Decision boundary changes more.





As for the relationship between Bayesian classifier used for classification of two classes of patterns and the classifiers designed for multiple classes, I think they share similar characteristics. Although classifier for multiple classes is more complex, it is still a classification problem. More classes mean that we need to decide a class that has the minimal risk and maximum accuracy among all classes rather than has more accuracy or has less risk compare to the other class. Multiple classes add the amount of probability density functions we need to compare. For high dimensional dataset, we just used high dimensional space to plot probability density function and obtain decision boundaries.

2.5 Experience

In the classes before, I have already learnt knowledge about Bayesian Classifier. But I haven't programmed to achieve this method and I haven't learnt much about classifiers of multiple classes. By conducting the experiment and visualization the results, I have a directly comprehension about this model. Bayesian Classifier is an effective classification method as it can do classification by both prior probability and new evidence. It is also easy to implement. The principle and the code is not difficult.

2.6 Codes

Bayesian.m

```
% This function is Bayesian classifier used for classification of two
classes of patterns
w1 = [-3.9847 -3.5549 -1.2401 -0.9780 -0.7932 -2.8531 -2.7605 -3.7287 -
3.5414 -2.2692 -3.4549 -3.0752 -3.9934 -0.9780 -1.5799 -1.4885 -0.7431 -
0.4221 -1.1186 -2.3462 -1.0826 -3.4196 -1.3193 -0.8367 -0.6579 -2.9683];
w2 = [2.8792 0.7932 1.1882 3.0682 4.2532 0.3271 0.9846 2.7648 2.6588];
% calculate prior  probability density functions
aver1=mean(w1);
aver2=mean(w2);
s1=std(w1);
s2=std(w2);
x=-5:0.1:5;
f1=(1 / (sqrt(2 * pi) * s1)) * exp(-1 * (x - aver1) .^ 2 / (2 * s1 ^ 2));
f2=(1 / (sqrt(2 * pi) * s2)) * exp(-1 * (x - aver2) .^ 2 / (2 * s2 ^ 2));
plot(x,f1,'r');
hold on
plot(x,f2,'g');
title('Prior Probability Density Functions');
% calculate posterior probability density functions
post_f1=(f1*0.9)./(f1*0.9+f2*0.1);
post_f2=(f2*0.1)./(f1*0.9+f2*0.1);
figure;
plot(x,post_f1,'r');
hold on
plot(x,post_f2,'g');
title('Posterior Probability Density Functions');
for i=1:length(x)
    if(post_f1(i)<0.5)
        disp(['Decision Boundary with maximum accuracy: ', num2str(x(i))])
        break
    end
end
% Consider the decision loss
r21=6;
```

```
r12=1;
l1=post_f2*r12;
l2=post_f1*r21;
figure;
hold on
plot(x,l1,'r');
plot(x,l2,'g');
title('Desicion Loss');
for i=1:length(x)
    if(l1(i)>l2(i))
        disp(['Decision Boundary with minimal risk: ', num2str(x(i))]);
        break
    end
end
```

Bayesian_2.m

```
% This function is Bayesian classifier used for classification of mutiple
classes of patterns
% Parameters
prob1 = 0.3;
prob2 = 0.1;
prob3 = 0.6;
risk = [0 2 6; 3 0 7; 2 1 0];
%set the average value of data
u1 = [0 0]';
u2 = [5 5]';
u3 = [1 3]';
%covariance matrix
Sigma1 = eye(2)*0.7;
Sigma2 = eye(2)*0.3;
Sigma3 = eye(2)*0.5;
%generate points
p1 = mvnrnd(u1,Sigma1,100);
p2 = mvnrnd(u2,Sigma2,100);
p3 = mvnrnd(u3,Sigma3,100);
figure;
plot(p1(:,1),p1(:,2),'b.',u1(1),u1(2),'r*','MarkerSize',10);
hold on
plot(p2(:,1),p2(:,2),'g.',u2(1),u2(2),'r*','MarkerSize',10);
plot(p3(:,1),p3(:,2),'k.',u3(1),u3(2),'r*','MarkerSize',10);
title('Dataset');
% calculate prior  probability density functions
aver1 = mean(p1);
aver2 = mean(p2);
aver3 = mean(p3);
cov1 = cov(p1(:, 1), p1(:, 2));
cov2 = cov(p2(:, 1), p2(:, 2));
cov3 = cov(p3(:, 1), p3(:, 2));
[x, y] = meshgrid(-5 : 0.1 : 10);
f1=reshape(mvnpdf([x(:), y(:)], aver1, cov1), size(x));
f2=reshape(mvnpdf([x(:), y(:)], aver2, cov2), size(x));
f3=reshape(mvnpdf([x(:), y(:)], aver3, cov3), size(x));
figure;
surf(x, y, f1);
hold on
surf(x, y, f2);
surf(x, y, f3);
title('Prior Probability Density Functions');
% calculate posterior probability density functions
post_f1 = (f1 * prob1)./(f1 * prob1 + f2 * prob2 + f3 * prob3);
```

```
post_f2 = (f2 * prob2)./(f1 * prob1 + f2 * prob2 + f3 * prob3);
post_f3 = (f3 * prob3)./(f1 * prob1 + f2 * prob2 + f3 * prob3);
figure;
surf(x, y, post_f1);
hold on
surf(x, y, post_f2);
surf(x, y, post_f3);
title('Posterior Probability Density Functions');
% Decision Boundary with maximum accuracy:
Max=zeros(length(x),length(y));
for i=1:length(x)
    for j=1:length(y)
        if(post_f1(i,j)>post_f2(i,j))
            if(post_f1(i,j)>post_f3(i,j))
                Max(i,j)=0;
            elseif(post_f3(i,j)>=post_f1(i,j))
                Max(i,j)=2;
            end
        else
            if(post_f2(i,j)>post_f3(i,j))
                Max(i,j)=1;
            elseif(post_f3(i,j)>=post_f2(i,j))
                Max(i,j)=2;
            end
        end
    end
end
figure;
hold on
surf(x, y, Max);
title('Decision Boundary with maximum accuracy');

% Consider the decision loss
l1=post_f2*risk(1,2)+post_f3*risk(1,3);
l2=post_f1*risk(2,1)+post_f3*risk(2,3);
l3=post_f1*risk(3,1)+post_f2*risk(3,2);
figure;
surf(x, y, l1);
hold on
surf(x, y, l2);
surf(x, y, l3);
title('Desicion Loss');


% Decision Boundary with with minimal risk:
Min=zeros(length(x),length(y));
for i=1:length(x)
    for j=1:length(y)
        if(l1(i,j)<l2(i,j))
            if(l1(i,j)<l3(i,j))
                Min(i,j)=0;
            elseif(l3(i,j)<=l1(i,j))
                Min(i,j)=2;
            end
        else
            if(l2(i,j)<l3(i,j))
                Min(i,j)=1;
            elseif(l3(i,j)<=l2(i,j))
                Min(i,j)=2;
            end
        end
    end
```

```
end
figure;
hold on
surf(x, y, Min);
title('Decision Boundary with minimal risk');
```