**实验3 分类器性能度量**

**一、实验目的**

（1）理解分类器性能度量的方法。

（2）编程ROC曲线绘制。

**二、实验内容**

1、编程实现ROC曲线绘制的函数roc\_plot。其中，分类概率使用matlab自带的朴素贝叶斯函数得到。

测试代码如下：

load('testingData.mat')

load('trainingData.mat')

nb = NaiveBayes.fit(trAttr, trLabels);

p=posterior (nb,tstAttr);

y1=p(:,2);

[auc1 ,FPR1 ,TPR1 ] = roc\_plot(y1,tstLabels);

**（将执行正确的roc\_plot函数代码粘贴在此处，核心代码要求有注释）**

function [AUC ,FPR ,TPR ] = roc\_plot(deci,tstLabels)

[val,ind] = sort(deci);

roc\_y = tstLabels(ind);

Num = length(roc\_y);

FPR = zeros(Num,1);

TPR = zeros(Num,1);

for i = 1:Num

TP = sum(roc\_y(i:end)==1);

FP = sum(roc\_y(i:end)==0);

P = sum(roc\_y(:)==1);

N = sum(roc\_y(:)==0);

TPR(i) = TP/P;

FPR(i) = FP/N;

end

TPR = [TPR;0];

FPR = [FPR;0];

AUC = 0;

for i = 2 : Num+1;

AUC = AUC + TPR(i)\*(FPR(i-1)-FPR(i));

end

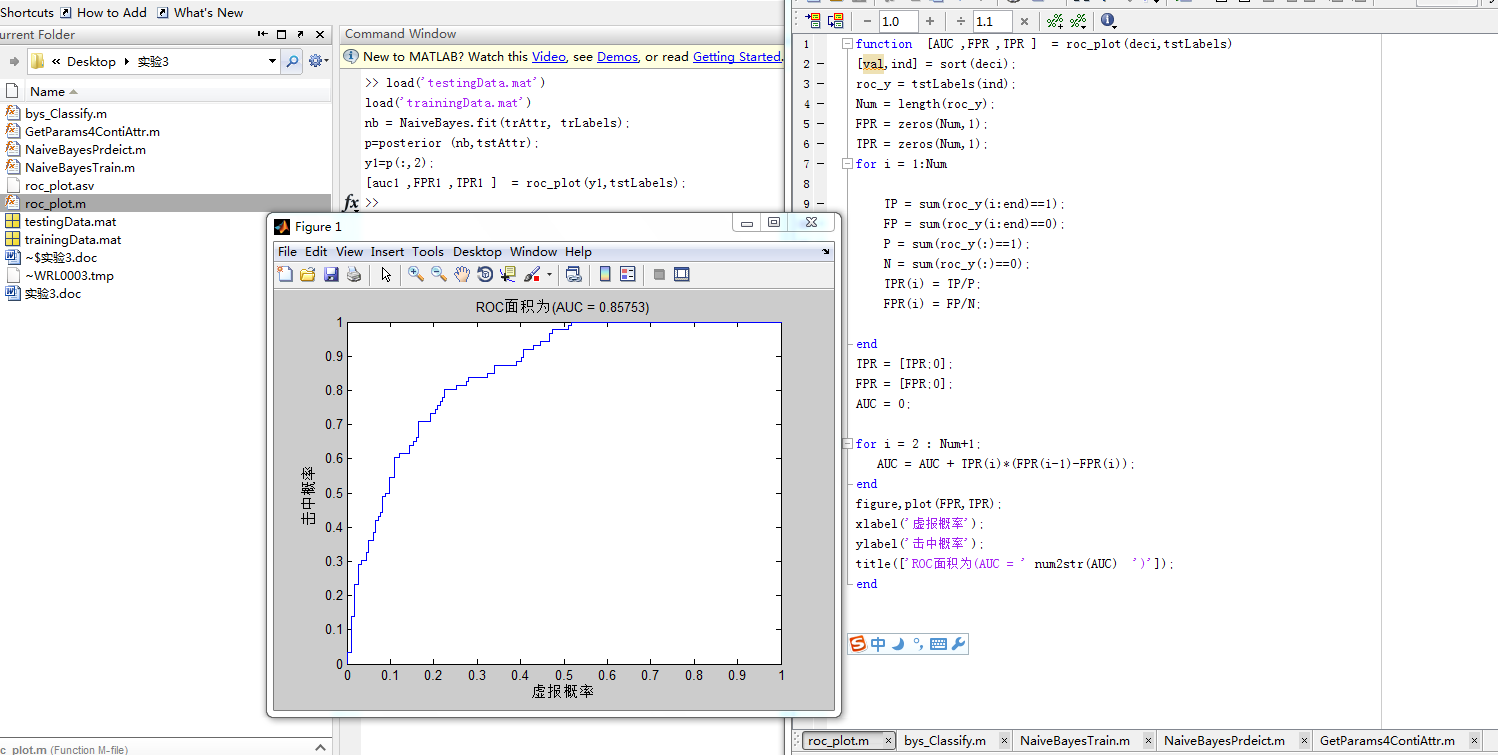
figure,plot(FPR,TPR);

xlabel('虚报概率');

ylabel('击中概率');

title(['ROC面积为(AUC = ' num2str(AUC) ')']);

end

实验结果截图：  


2、（难度2）编程实现ROC曲线绘制的函数roc\_plot。其中，分类概率使用自己编写的朴素贝叶斯函数bys\_post1得到。

测试代码如下：

load('testingData.mat')

load('trainingData.mat')

[post4all]=bys\_Classify(trAttr, trLabels,tstAttr);

y2=post4all(:,2);

[auc2 ,FPR2 ,TPR2 ] = roc\_plot(y2,tstLabels);

**（将执行正确的bys\_post1函数代码粘贴在此处，核心代码要求有注释）**

朴素贝叶斯函数

function [post4all] = bys\_Classify(trAttr,trLabels,tstAttr)

[Params,prior,AllLabels] = NaiveBayesTrain(trAttr, trLabels);

N = size(tstAttr,1);

Labels = zeros(N,1);

for i=1 : N

tstAttrSample = tstAttr(i,:);

post = NaiveBayesPrdeict(Params,prior,tstAttrSample);

post4all(i,:) = post./sum(post);

end

end

function [Params,prior,AllLabels] = NaiveBayesTrain(trAttr,trLabels)

AllLabels = unique(trLabels);

numClass = length(AllLabels);%取得所有类标号的种类个数

Ind4Class{numClass,1} = [];

for i =1 : numClass

ind = find(trLabels == AllLabels(i));

Ind4Class{i,1} = ind;

prior(i) = length(ind)/length(trLabels);

end

numAttr = size(trAttr,2);

Params{numClass,numAttr} = [];

for AttrInd = 1 : numAttr

AttrVals = trAttr(:,AttrInd);

for i = 1 : numClass

Params{i,AttrInd} = GetParams4ContiAttr(AttrVals,Ind4Class{i,1});

end

end

end

function Paramsl = GetParams4ContiAttr(AttrVals,ind)

Attrvals4ThisClass = AttrVals(ind);

Paramsl(1) = mean(Attrvals4ThisClass);

Paramsl(2) = var(Attrvals4ThisClass);

end

function post = NaiveBayesPrdeict(Params,prior,tstAttr)

numClass = length(prior);

numAttr = length(tstAttr);

post = [];

for i = 1: numClass

p\_AttrCond = 1;

for AttrInd = 1 : numAttr

AttrVals = tstAttr(AttrInd);

mean = Params{i,AttrInd}(1);

sig = sqrt(Params{i,AttrInd}(2));

p = (1/(sqrt(2\*pi)\*sig))\*exp(-(AttrVals-mean)^2/(2\*sig^2));

p\_AttrCond = p\_AttrCond\*p;

end

post(i) = p\_AttrCond\*prior(i);

end

end

实验结果截图：

