Лабораторная работа № 6

Вариант № 4

Распознавание образов на основе непараметрических алгоритмов оценивания плотности распределения случайной величины

Цель работы

Исследовать алгоритмы распознавания образов на основе оценивания плотности распределения случайных величин и случайных векторов при использовании методов Парзена и k ближайших соседей.

Задание

Используя код Вашей лабораторной №3, реализуйте алгоритм распознавания образов, применив оценивание по методу k ближайших соседей. Вычислите экспериментально вероятности ошибок распознавания. Сравните их с вероятностями ошибок (теоретическими или экспериментальными), полученными в ходе выполнения лабораторной №3. Отобразите поверхности плотностей распределения классов, задаваемых теоретически, и полученных в результате оценивания.

Код программы (внесённые изменения в шаблон кода выделены)

```
clear all; close all;
% Построить  график  зависимости  суммарной  экспериментальной  ошибки
% первого рода ( для  третьего  класса)  от числа испытаний  (объема выборки ).
% Сравнить с теоретическим  значением.

% Нужно добавить вот эти строки
% KK = [100 500 1000 2000 5000 10000];    % разные значения объемов выборки
KK = 1000;
err1c3 = zeros(size(KK));                % ошибка первого рода третьего класса
err2c3 = zeros(size(KK));                % ошибка второго рода третьего класса

% Добавляется цикл по объемам выборки
for tt = 1 : numel(KK)  % цикл по объемам выборки
    %1.Задание исходных данных
    n=2;M=2;%%размерность признакового пространства и число классов
    K = KK(tt); % K=1000;%количество статистических испытаний

    %Априорные вероятности, математические ожидания и матрицы ковариации классов
    dm=2.0;%расстояние между математическими ожиданиями классов по координатным осям
    C=zeros(n,n,M); C_=C;%матрица ковариации вектора признаков различных классов
    pw=[0.4 0.6 0.5];
```

```
    pw=pw/sum(pw);
    D=3*eye(2);
    m=[4 1; -1 -2; -4 1]';
    C(:,:,1)=[3 -1; -1 3];
    C(:,:,2)=[3 1; 1 3];
    C(:,:,3)=[3 2; 2 3];
    for k=1:M,
        C_(:,:,k)=C(:,:,k)^-1;
    end;
    np=sum(pw); pw=pw/np; %исключение некорректного задания априорных вероятностей

    % + Этот пункт 1.1. Генерация обучающих выборок классов
    % число образов каждого класса
    Ks = fix(K * pw);
    Ks(end) = K - sum(Ks(1 : end - 1));
    for i=1:M,%цикл по классам
        XN{i} = repmat(m(:,i), [1, Ks(i)]) + randncor(n,Ks(i),C(:,:,i)); %генерация K
образов i-го класса
    end;

    %2.Расчет матриц вероятностей ошибок распознавания
    PIJ=zeros(M); PIJB=zeros(M); mg=zeros(M); Dg=zeros(M); l0_=zeros(M);
    for i=1:M,
        for j=i+1:M,
            dmij=m(:,i)-m(:,j);
            l0_(i,j)=log(pw(j)/pw(i));
            dti=det(C(:,:,i)); dtj=det(C(:,:,j));
            trij=trace(C_(:,:,j)*C(:,:,i)-eye(n)); trji=trace(eye(n)-
C_(:,:,i)*C(:,:,j));
            mg1=0.5*(trij+dmij'*C_(:,:,j)*dmij-log(dti/dtj));
            Dg1=0.5*trij^2+dmij'*C_(:,:,j)*C(:,:,i)*C_(:,:,j)*dmij;
            mg2=0.5*(trji-dmij'*C_(:,:,i)*dmij+log(dtj/dti));
            Dg2=0.5*trji^2+dmij'*C_(:,:,i)*C(:,:,j)*C_(:,:,i)*dmij;
            sD1=sqrt(Dg1); sD2=sqrt(Dg2);
            PIJ(i,j)=normcdf(l0_(i,j),mg1,sD1); PIJ(j,i)=1-normcdf(l0_(i,j),mg2,sD2);
            mu2=(1/8)*dmij'*((C(:,:,i)/2+C(:,:,j)/2)^-1)*dmij...
                +0.5*log((dti+dtj)/(2*sqrt(dti*dtj)));%расстояние Бхатачария
            PIJB(i,j)=sqrt(pw(j)/pw(i))*exp(-mu2);PIJB(j,i)=sqrt(pw(i)/pw(j))*exp(-
mu2);%границы Чернова
        end;
        PIJB(i,i)=1-sum(PIJB(i,:));
        PIJ(i,i)=1-sum(PIJ(i,:));%нижняя граница вероятности правильного
распознавания
    end;

    %+2.1.Определение вероятностей ошибок методом скользящего контроля
    % Убрать отсюда всё, где есть Pc2 и p2_
    r=0.5; kl_kernel=11;  % параметры оценки Парзена
    Pc1=zeros(M);%матрицы ошибок
    p1_=zeros(M,1);
    for i=1:M,%реализация метода скользящего контроля
        N=Ks(i);
        XNi=XN{i}; XNi_=zeros(n,N-1);
        indi=[1:i-1,i+1:M];
        for j=1:N,
            x=XNi(:,j); indj=[1:j-1,j+1:N];%изъятие тестового образа i-го класса
            XNi_(:,1:j-1)=XNi(:,1:j-1); XNi_(:,j:end)=XNi(:,j+1:end);
            h_N=N^(-r/n);       % + %размеры окна Парзена
            p1_(i)=vkernel(x,XNi_,h_N,11);%оценка Парзена
            for t=1:M-1,
                ij=indi(t);
                h_N=Ks(ij)^(-r/n);       % + %размеры окна Парзена
```

```
                p1_(ij)=vkernel(x,XN{ij},h_N,11);
            end;
            [ui1,iai1]=max(p1_);
            Pc1(i,iai1)=Pc1(i,iai1)+1;
        end;
        Pc1(i,:)=Pc1(i,:)/N;
    end;

    %3.Тестирование алгоритма методом статистических испытаний
    Pcv=zeros(M); p=zeros(M,1); % +

    x=ones(n,1); u=zeros(M,1);
    Pc_=zeros(M);%экспериментальная матрица вероятностей ошибок
    for k=1:K,%цикл по числу испытаний
        for i=1:M,%цикл по классам
            [x,px]=randncor(n,1,C(:,:,i)); x=x+m(:,i);%генерация образа i-го класса
            for j=1:M,%вычисление значения разделяющих функций
                u(j)=-0.5*(x-m(:,j))'*C_(:,:,j)*(x-m(:,j))-
0.5*log(det(C(:,:,j)))+log(pw(j));
                h_N=Ks(j)^(-r/n);       % + %размеры окна Парзена
                p(j)=vkernel(x, XN{j}, h_N, 11);    % +
            end;
            [ui,iai]=max(u);%определение максимума
            Pc_(i,iai)=Pc_(i,iai)+1;%фиксация результата распознавания

            [ui,iai]=max(p);        % + %определение максимума
            Pcv(i,iai)=Pcv(i,iai)+1;% + %фиксация результата распознавания
        end;
    end;
    Pc_=Pc_/K;
    Pcv=Pcv/K;  % +

    % В конце шага цикла фиксируем очередное значение ошибки
    err1c3(tt) = PIJB(2, 1); % первый класс второй род
end % конец цикла по объемам выборки

disp('Теоретическая матрица вероятностей ошибок');disp(PIJ);
disp('Матрица ошибок по методу скользящего контроля');disp(Pc1);
disp('Матрица ошибок на основе границы Чернова');disp(PIJB);
disp('Экспериментальная матрица ошибок (гауссовский классификатор)');disp(Pc_);
disp('Экспериментальная матрица ошибок (с оценками Парзена)');disp(Pcv);

% Вычисление значений вероятностной функции для построения поверхности
[X, Y] = meshgrid(-15:0.5:15, -15:0.5:15);
Z1 = zeros(size(X)); Z2 = zeros(size(X)); Z3 = zeros(size(X));
for i = 1:numel(Z1)
    x = [X(i); Y(i)];
    for j = 1:M
        l0 = log(pw(j)) - 0.5*log(det(C(:,:,j))) - 0.5*(x-m(:,j))'*C_(:,:,j)*(x-
m(:,j));
        Z1(i) = Z1(i) + pw(j)*exp(l0);

        l0 = log(pw(j)) - 0.5*(x-m(:,j))'*C_(:,:,j)*(x-m(:,j));
        Z2(i) = Z2(i) + pw(j)*exp(l0);

        p = vkernel(x, XN{j}, 1.5, 11);
        Z3(i) = Z3(i) + pw(j)*p;
    end
end
% Построение поверхности
figure;
surf(X, Y, Z1);
```

```
hold on;
surf(X, Y, Z2);
surf(X, Y, Z3);
title('Поверхность плотности распределения классов');
xlabel('x_1'); ylabel('x_2'); zlabel('Плотность распределения');
legend('Теоретическая', 'Без учета детерминированных признаков',' С учетом
детерминированных признаков');
```

Результаты выполнения задания

В итоге получаем:

```
Теоретическая матрица вероятностей ошибок
    0.9604    0.0396
    0.0588    0.9412

Матрица ошибок по методу скользящего контроля
    0.9737    0.0263
    0.0551    0.9449

Матрица ошибок на основе границы Чернова
    0.7030    0.2970
    0.1980    0.8020

Экспериментальная матрица ошибок (гауссовский классификатор)
    0.9490    0.0510
    0.0470    0.9530

Экспериментальная матрица ошибок (с оценками Парзена)
    0.9720    0.0280
    0.0700    0.9300
```

Рисунок 1 – Матрицы вероятностей ошибок

1.0000	0.0000
0	1.0000

Рисунок 2 – Матрица из 3 лабораторной

Отображение поверхности плотностей распределения классов:

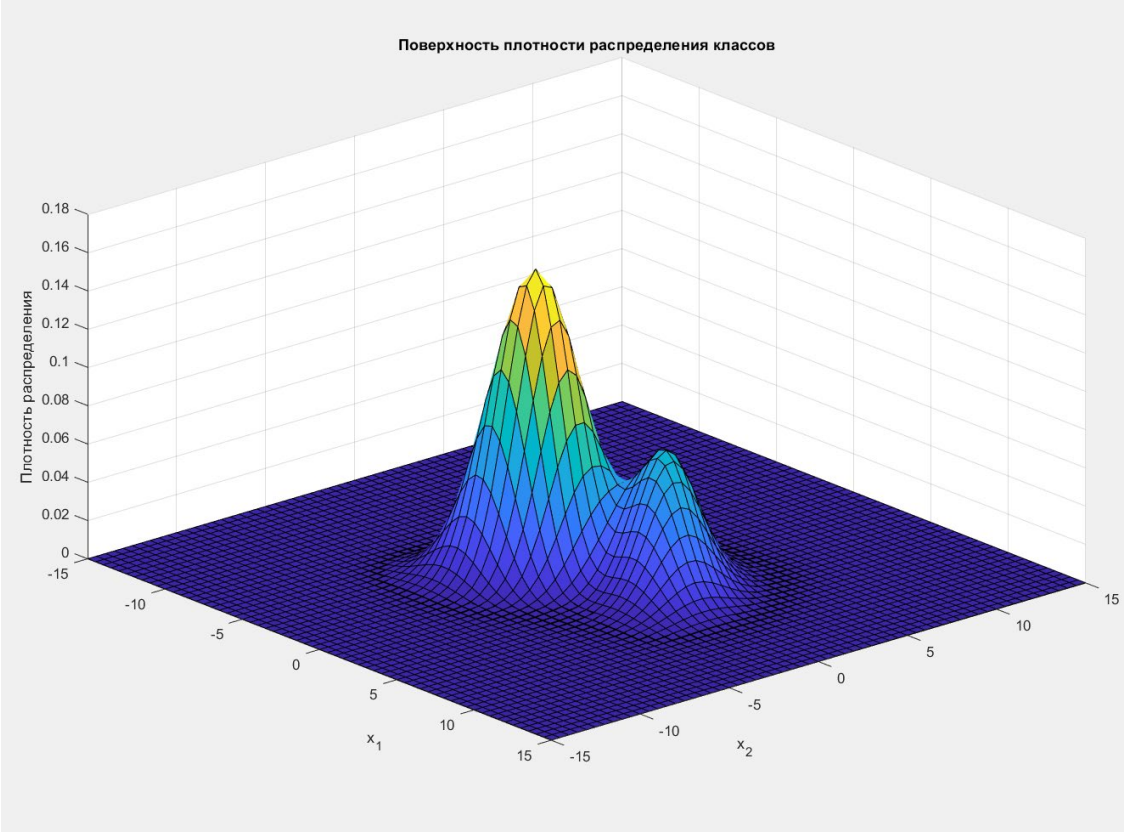

Рисунок 3 – Поверхность плотности распределения классов

Выводы

1. Добиться улучшения качества оценивания можно манипулируя параметром r оконной функции. Например, при r=0.37 получается значение куда лучше:

```
Экспериментальная матрица ошибок (с оценками Парзена)
    0.9550    0.0450
    0.0480    0.9520
```

Рисунок 4 – Экспериментальная матрица ошибок (с оценками Парзена при r = 0.37)

2. Наилучшую точность распознавания, исходя из матриц ошибок, обеспечивает гауссовская оконная функция

```
Матрица ошибок на основе границы Чернова
    0.7030    0.2970
    0.1980    0.8020
```

Рисунок 5 – Матрица ошибок на основе границы Чернова