Путин Павел Александрович, группа 7-1

Лабораторная работа № 6

Вариант № 1

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

Получить у преподавателя вариант задания и написать код, реализующий соответствующий алгоритм обработки информации. Для ответа на поставленные в задании вопросы провести численный эксперимент или статистическое имитационное моделирование и представить соответствующие графики. Провести анализ полученных результатов и представить его в виде выводов по проделанной работе.

Используя код Вашей лабораторной №2, реализуйте алгоритм распознавания образов, применив оценивание по методу Парзена. Вычислите вероятности ошибок распознавания на основе метода скользящего контроля. Сравните их с вероятностями ошибок (теоретическими или экспериментальными), полученными в ходе выполнения лабораторной №2

Код программы (внесённые изменения в шаблон кода выделены)

Оценивание по методу Парзена. Вычисление ошибок распознавания на основе метода скользящего контроля

```
% Файл pr56_rec_npar_learn. Обучение алгоритмов распознавания на основе оценок
% плотностей распределения методами Парзена и k - ближайших соседей
clear all;
close all;
%% 1. Задание исходных данных
n = 2;M = 2; % размерность признакового пространства и число классов
H = 1; % количество статистических испытаний процесса обучения
K = 1000; % количество статистических испытаний алгоритма распознавания
pw = ones(1, M) / M; % априорные вероятности классов (значение по умолчанию)
% Исходные данные для генерации обучающих выборок
% dm - параметр, определяющий степень рассредоточенности компонентов смесей
% DM - параметр, определяющий сдвиг областей локализации классов по осям
% L - количество компонентов в смеси каждого класса
L = [2, 2];
dm = 2;
DM = 1;
ps = {[0.5; 0.5],[0.5; 0.5]};
mM = {[2 2], [1 -1]};
C = [5 1; 1 5];
kl_kernel = 11;
r = 0.5;
gm = 0.25; % параметры оценок (см.описание функций vkernel, vknn)
%% 2. Генерация обучающих данных в цикле с переменным объемом выборки
Nn = [10, 20, 30, 40, 50, 100, 150, 200, 250];
ln = length(Nn);
Esth1 = zeros(1, ln); % суммарная оценка Парзена методом скользящего контроля
for nn = 1:ln % цикл с изменением объема обучающих выборок
    NN = Nn(nn) * ones(1, M);
    N = Nn(nn);
    h_N = N^(-r / n); % размеры окна Парзена
    k = 2 * round(N^gm) + 1; % k - число ближайших соседей
    for h = 1:H % цикл статистических испытаний процесса обучения
        XN = gen(n, M, NN, L, ps, mM, C, h); % генерация обучающих выборок

        % 2.1. Определение вероятностей ошибок методом скользящего контроля
        Pc1 = zeros(M);
        p1_ = zeros(M, 1);
        for i = 1:M % реализация метода скользящего контроля
             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);
                p1_(i) = vkernel(x, XNi_, h_N, kl_kernel); % оценка Парзена
                for t = 1:M - 1
                     ij = indi(t);
                     p1_(ij) = vkernel(x, XN{ij}, h_N, kl_kernel);
                end
```

```
                [ui1, iai1] = max(p1_);
                Pc1(i, iai1) = Pc1(i, iai1) + 1;
            end
            Pc1(i, :) = Pc1(i, :) / N;
        end
        disp("Матрица ошибок с объёмом выборки " + num2str(N));
        disp(Pc1);
        Esth1(nn) = Esth1(nn) + (1 - sum(pw.* diag(Pc1)')); % суммарная ошибка
    end % по h
end % по nn
Esth1 = Esth1 / H;
%% 4. Визуализация зависимостей вероятностей ошибок
figure;
grid on;
hold on;
ms = max(Esth1);
axis([Nn(1), Nn(ln), 0, ms + 0.001]); % установка границ поля графика по осям
p = plot(Nn, Esth1, '-b');
set(p, 'LineWidth', 1.0);
title('Суммарная вероятность ошибки по методу оценки Парзена с использованием
метода скользящего контроля', 'FontName', 'Courier', 'FontSize', 14);
xlabel('N', 'FontName', 'Courier', 'FontSize', 14);
ylabel('Es', 'FontName', 'Courier', 'FontSize', 14);
hold off;
```

Функция для генерации обучающих выборок образов

```
function XN=gen(n,M,NN,L,ps,mM,C,vis)
%Функция для генерации обучающих и тестовых выборок образов
%на основе смесей гауссовских распределений
%n - размерность признакового пространства
%M - число классов образов
%NN - вектор-строка (1xM), содержащий объемы генерируемых выборок классов
%L - вектор-строка (1xM), содержащий количество компонентов в смесях классов
%ps - массив ячеек, содержащий вероятности (веса) компонентов смесей классов
%mM - массив ячеек, содержащий математические ожидания ГСВ компонентов смесей
%D -  массив ячеек, содержащий значения дисперсии ГСВ
%ro - массив ячеек, содержащий значения коэффициентов корреляции ГСВ в смесях
%vis – ключ для выполнения визуализации областей локализации данных
%XN - массив ячеек, содержащий генерируемые выборки образов различных классов
%Контрольный пример: "заплетенные восьмерки"
% n=2; M=2; NN=[1000,1000]; L=[2,2]; ps={[0.5;0.5],[0.5;0.5]}; D={[1,1],[1,1]};
ro={[0.5;0.5],[-0.5;-0.5]};
% dm=4; mM{1}=[[0;0],[dm;dm]]; mM{2}=[[dm;0],[0;dm]]; DM=0; vis=1;
N=sum(NN);%общее количество генерируемых выборок
XN=cell(1,M);
%Организация цикла генерации выборок по номерам классов
for k=1:M,
    XN{k}=zeros(n,NN(k));
    mMk=mM{k}; psk=ps{k};
    XNk=zeros(n,NN(k));
    for i=1:NN(k),%генерация выборок
         u=rand;%определение индекса принадлежности к компоненту смеси
         a=0; t=L(k);
         for j=1:L(k),
              b=a+psk(j);
```

```
            if (a<=u) && (u<b), t=j; end;
            a=b;
        end;
        XNk(:,i)=randncor(n,1,C)+mMk(:,t);
    end;
    XN{k}=XNk;
end;
M_=M; if M>4, M_=4; end;
%Контрольная визуализация полученных выборок для двумерного вектора признаков и M<5
if (vis==1) && (n>1),
    close
    figure(vis); set(gca,'FontSize',12); grid on; hold on;
    strv=cell(1,4); strv{1}='ro'; strv{2}='k+'; strv{3}='b^'; strv{4}='g*';
    for k=1:M_,
        XNk=XN{k};
        pg2=plot(XNk(1,:),XNk(2,:),strv{k}); set(pg2,'LineWidth',1.25);
    end;
    xlabel('x1','FontName','Courier');ylabel('x2','FontName','Courier');
    title('Пространственная локализация генерируемых
образов','FontName','Courier');hold off;
end;
end
```

Синтез и анализ алгоритмов распознавания ГСВ

```
% Пример вар.4. Вычислить абсолютную ошибку оценивания плотности распределения
% случайного вектора в clear all

close all
%% 1.Задание исходных данных

n = 2;              % размерность признакового пространства
M = 2;              % число классов
K = 1000;           % количество статистических испытаний
m = [2 2; 1 -1]';   % мат. ожидания - координаты центров классов (2,-3) и (1,6)
pw = [0.5, 0.5];    % априорные вероятности классов (доля образов каждого класса в
общей выборке)
C = [5 1; 1 5];     % матрица ковариаций классов
Cinv = C ^ -1;      % обратная ков. матрица
IMS = [];           % общая совокупность образов (общая выборка)

% 1.1. Визуальзация исходной совокупности образов
% Определение числа образов в каждом классе, пропорцианально pw
Ks = fix(pw .* K);
Ks(end) = K - sum(Ks(1 : end - 1));
label = {'bo', 'r+', 'k*', 'gx'}; % маркеры классов для визуализации

figure;
hold on;
title('Исходные метки образов');
for i = 1 : M % цикл по классам
    % генерация Ks(i) образов i-го класса
    ims = repmat(m(:, i), [1, Ks(i)]) + randncor(n, Ks(i), C);
    if n == 2
        plot(ims(1, :), ims(2, :), label{i}, 'MarkerSize', 10, 'LineWidth', 1);
    elseif n == 3
```

```
        plot3(ims(1, :), ims(2, :), ims(3, :), label{i}, 'MarkerSize', 10,
'LineWidth', 1);
    end
    IMS = [IMS, ims];   % добавление в общую совокупность образов
end
%% 2.Расчет разделяющих функций и матрицы вероятностей ошибок распознавания
G = zeros(M, n + 1);    % разделяющие функции ???
PIJ = zeros(M);         % теоретическая матрица ошибок
l0_ = zeros(M);         % порог принятия ошибки

for i = 1 : M % цикл по классам
    G(i, 1 : n) =(Cinv * m(:, i))';
    G(i, n + 1) = -0.5 * m(:, i)' * Cinv * m(:, i);
    for j = i + 1 : M
        l0_(i, j) = log(pw(j) / pw(i));
        h = 0.5 * (m(:, i) - m(:, j))' * Cinv * (m(:, i) - m(:, j));
        sD = sqrt(2 * h);
        PIJ(i, j) = normcdf(l0_(i, j), h, sD);
        PIJ(j, i) = 1 - normcdf(l0_(i, j), -h , sD);
    end
    % нижняя граница вероятности правильного распознавания (на главной диагонали)
    PIJ(i, i) = 1 - sum(PIJ(i, :));
end

% 2.1. Визуальзация результатов распознавания образов
figure;
hold on;
title('Результат классификации образов');
for i = 1 : K               % цикл по всем образам совокупности
    z = [IMS(:, i); 1];     % значение очердного образа из общей совокупности
    u = G * z + log(pw');   % вычисление значения разделяющих функций
    [ui, iai] = max(u);     % определение максимума (iai - индекс класса)
    if n == 2
        plot(IMS(1, i), IMS(2, i), label{iai}, 'MarkerSize', 10, 'LineWidth', 1);
    elseif n == 3
        plot3(IMS(1, i), IMS(2, i), IMS(3, i), label{iai}, 'MarkerSize', 10,
'LineWidth', 1);
    end
end

%% 3.Тестирование алгоритма методом статистических испытаний
x = ones(n + 1, 1);     % образы классов ???
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);
        x(1 : n, 1) =m(:, i) + x_;      % генерация образа i-го класса
        u = G * x + log(pw');           % вычисление значения разделяющих функций
        [ui, iai] = max(u);             % определение максимума
        Pc_(i, iai) = Pc_(i, iai) + 1;  % фиксация результата распознавания
    end
end

% матрица ошибок, полученная экспериментально
```

```
% у нее такая же структура как и в PIJ,
% только вычисляется численно, а не по формулам
Pc_ = Pc_ / K;

disp('Теоретическая матрица вероятностей ошибок');
disp(PIJ);
disp('Экспериментальная матрица вероятностей ошибок');
disp(Pc_);
%% 4.Визуализация областей принятия решений для двумерного случая
if n == 2
    D = 1;
    xmin1 = -4 * sqrt(D) + min(m(1,:)); % левая граница графика по оси абсцисс
    xmax1 = 4 * sqrt(D) + max(m(1,:));  % правая граница графика по оси абсцисс
    xmin2 = -4 * sqrt(D) + min(m(2,:)); % нижняя граница графика по оси ординат
    xmax2 = 4 * sqrt(D) + max(m(2,:));  % верхняя граница графика по оси ординат
    x1 = xmin1 : 0.05 : xmax1;          % отсчёты по оси абсцисс
    x2 = xmin2 : 0.05 : xmax2;          % отсчёты по оси ординат
    [X1, X2] = meshgrid(x1, x2);        % матрицы значений координат случайного
вектора
    x12 = [X1(:), X2(:)];

    figure;
    hold on;
    grid on;
    axis([xmin1, xmax1, xmin2, xmax2]); % установка границ поля графика по осям

    for i = 1 : M % цикл по классам
        f2 = mvnpdf(x12, m(:, i)', C); % массив значений плотности распределения
        f3 = reshape(f2, length(x2), length(x1)); % матрица значений плотности
распределения
        [Ch, h] = contour(x1, x2, f3, [0.01, 0.5 * max(f3(:))], 'Color', 'b',
'LineWidth', 0.75);
        clabel(Ch, h);
        for j = i + 1 : M % изображение разделяющих границ
            wij = Cinv * (m(:, i) - m(:, j));
            wij0 = -0.5 * (m(:, i) + m(:, j))' * Cinv * (m(:, i) - m(:, j));
            f4 = wij' * x12' + wij0;
            f5 = reshape(f4, length(x2), length(x1));
            [Ch_, h_] = contour(x1, x2, f5, -l0_(i, j) + 0.0001, 'Color', 'k',
'LineWidth', 1.25);
        end
    end
    set(gca, 'FontSize', 13);
    title('Области локализации классов и разделяющие границы', 'FontName',
'Courier');
    xlabel('x1', 'FontName', 'Courier');
    ylabel('x2', 'FontName', 'Courier');
    strv1 = ' pw=';
    strv2 = num2str(pw, '% G');
    text(xmin1 + 1, xmax2 - 1, [strv1, strv2], 'HorizontalAlignment', 'left',
'BackgroundColor', ...
        [.8 .8 .8], 'FontSize', 12);
    legend('wi', 'gij(x)=0');
    hold off;
end
```

Результаты выполнения задания

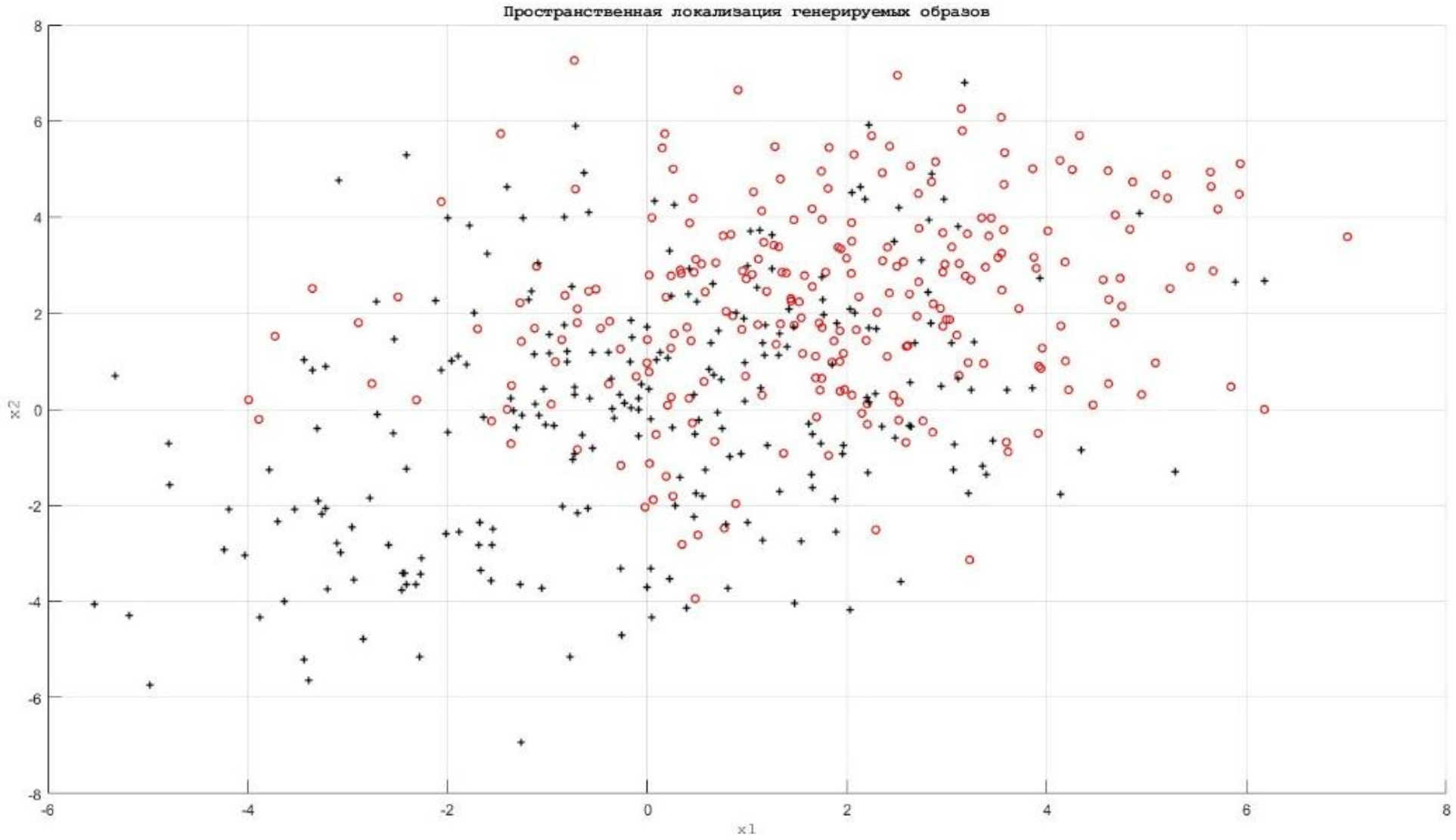

Рисунок 1 – Объекты, сгенерированные при оценке методом Парзена

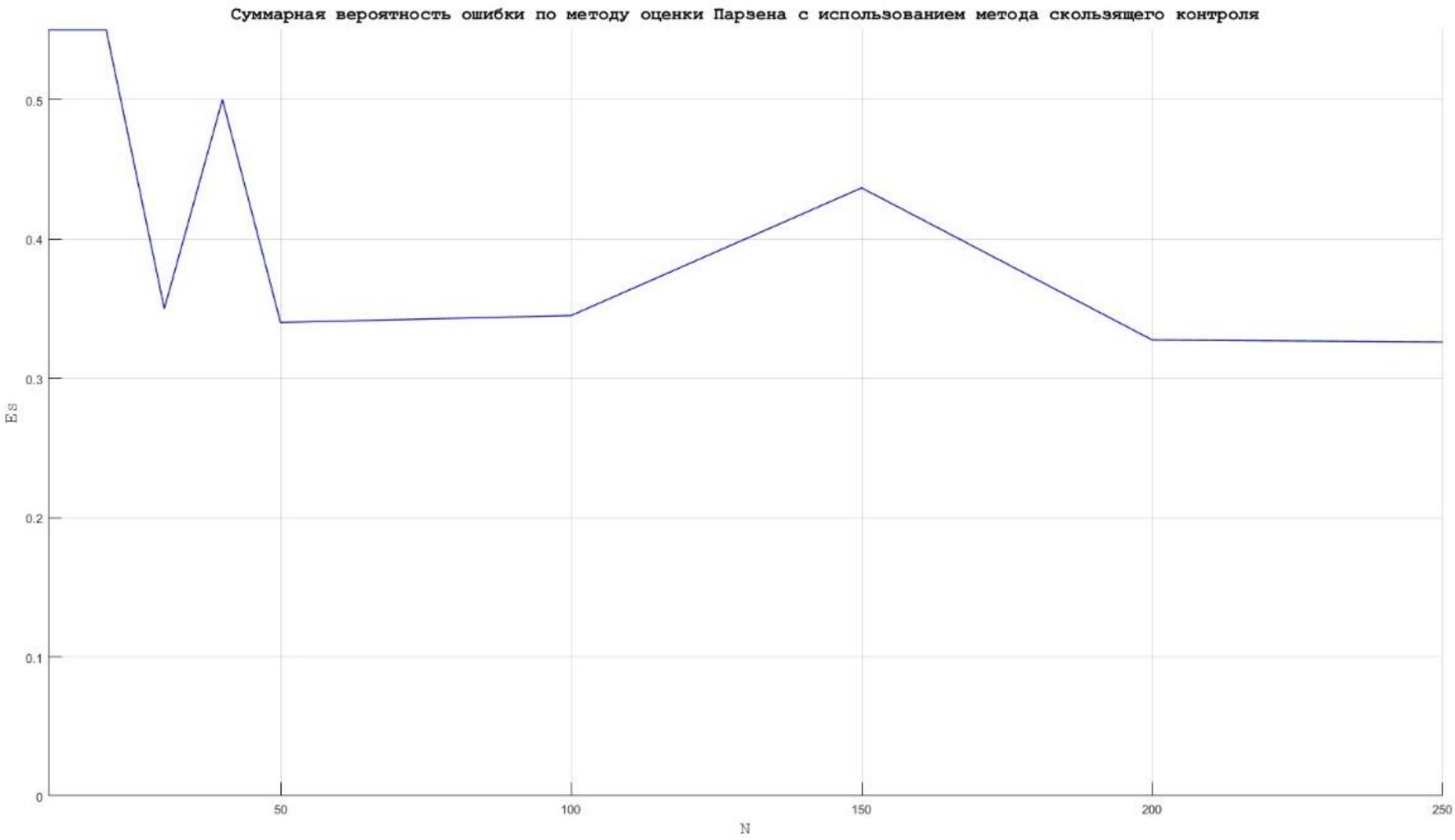

Рисунок 2 – Суммарная ошибка, вычисленная методом скользящего контроля

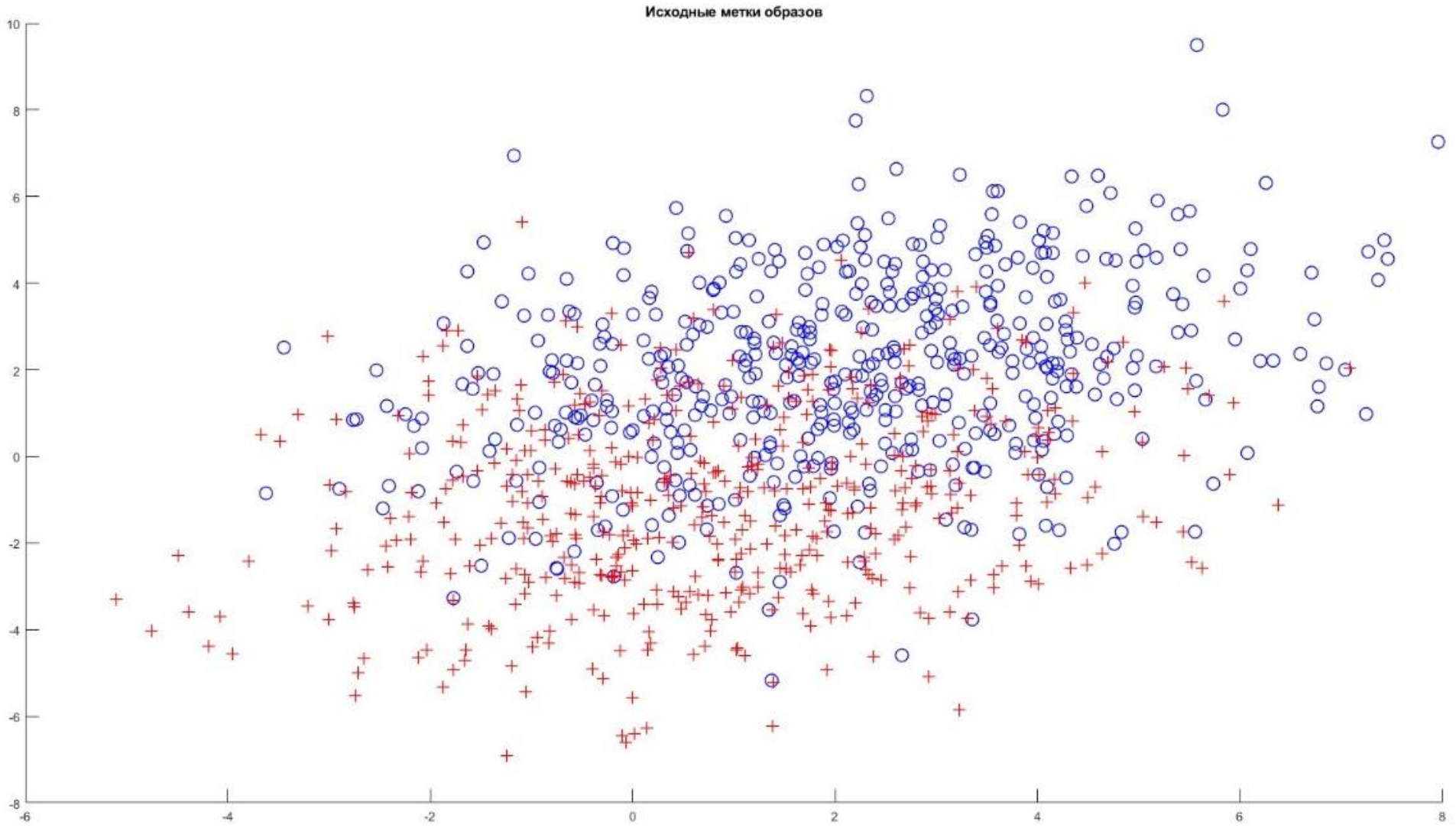

Рисунок 3 – Объекты, сгенерированные в лабораторной 2

Таблица 1 - Матрицы ошибок

Метод скользящего контроля		**Теоретическая**		**Экспериментальная**	
0,6920	0,3080	0,7508	0,2492	0,7400	0,2600
0,3440	0,6560	0,2492	0,7508	0,2260	0,7740

Таблица 2 - Исследование зависимости матрицы ошибок от параметра оконной прямоугольной функции

Параметр оконной функции	Матрица ошибок	
0,75	0,6600	0,3400
	0,3800	0,6200
0,50	0,6680	0,3320
	0,3520	0,6480
0,25	0,7600	0,2400
	0,3600	0,6360
0,125	0,7360	0,2640
	0,3520	0,6480

Таблица 3 – Исследование зависимости матрицы ошибок от вида оконной функции (значение параметра 0,25)

Вид оконной функции	**Матрица ошибок**	
Гауссовская функция с использованием диагональной матриц	0,7240	0,2760
	0,3240	0,6760
Гауссовская функция с использованием матрицы ковариации	0,7800	0,2200
	0,3760	0,6240
Оконная прямоугольная функция	0,6960	0,3040
	0,4400	0,5600
Оконная треугольная функция	0,6480	0,3520
	0,4440	0,5560
Показательная функция	0,7600	0,2400
	0,3600	0,6400

Выводы

1. На основе данных из таблицы
2. можно сделать вывод, что наиболее точное определение ошибки методом скользящего контроля достигается при значении параметра окна равном 0,125. Но стоит учитывать, что результаты являются случайными, и наилучшее значение достигается не во всех запусках.
3. На основе данных из таблицы 3 можно сделать вывод, что наиболее точное определение ошибки методом скользящего контроля достигается при использовании гауссовской функции с использованием диагональной матрицы. Но стоит учитывать, что результаты являются случайными, и наилучшее значение достигается не во всех запусках