САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ПЕТРА ВЕЛИКОГО

Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Отчет

Лабораторная 0

Генерация и визуализация исходных данных

Дисциплина
«Нейроинформатика»

выполнил:
Косенков М.А.
группа: 33531/2
преподаватель:
Никитин К.В.

Санкт-Петербург
2018

Оглавление

Задание 1,2

Исходные данные 1 – Крестики-нолики

Задание

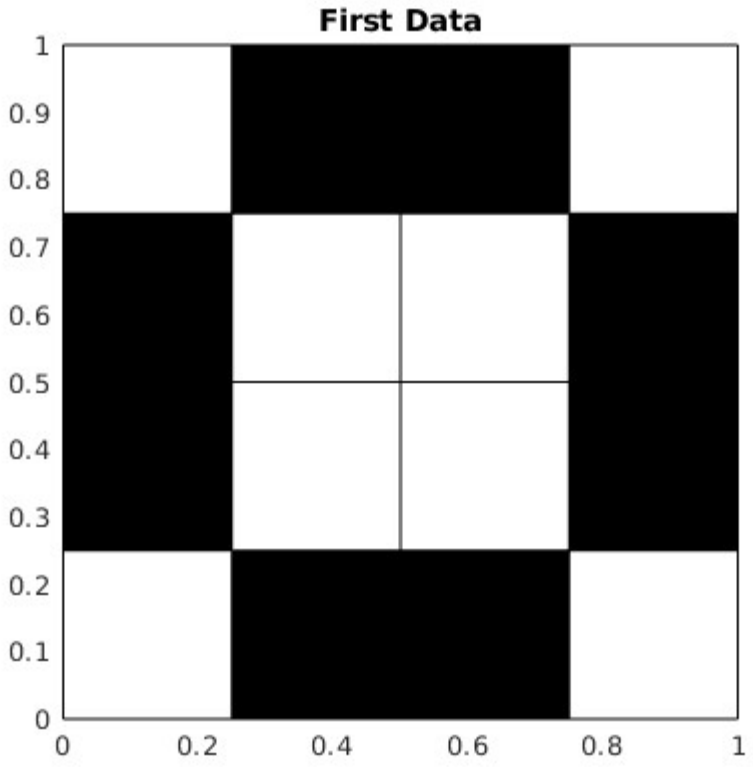
Разделите таблицу 4x4 на крестики ‘X’ и нолики ‘O’ так, чтобы классы ‘X’ и ‘O’ были линейно неразделимы, а количество примеров каждого класса было одинаковым и равным 8.

Решение

0	1	1	0
1	0	0	1
1	0	0	1
0	1	1	0

*Программа **firstData***

```
P = zeros(16, 2);
T = zeros(16, 1);
for i = 1:16
   P(i, 1) = fix((i - 1)/4) * 0.25;
   P(i, 2) = mod((i - 1), 4) * .25;
end

T(2) = 1; T(3) = 1;
T(5) = 1; T(8) = 1;
T(9) = 1; T(12) = 1;
T(14) = 1; T(15) = 1;

% Plot
color1 = [0 0 0];
color2 = [1 1 1];
figure
hold on
for i = 1:length(T)
   r = rectangle('Position', [P(i, 1) P(i, 2) .25 .25], 'FaceColor',color2);
   if T(i) == 1
      r.FaceColor = color1;
   end
end

axis([0 1 0 1])
axis square
title('First Data');
hold off
```



Исходные данные 2 – Логическая функция 5 переменных

Задание

Придумайте логическую функцию (ЛФ) 5 переменных так, чтобы множество ее выходных значений 0 и 1 было линейно неразделимым. Как вариант, придумайте n = 5-6 чисел от 0 до 31. На двоичных представлениях этих чисел ЛФ будет принимать значение 1. На всех остальных значениях ЛФ будет принимать значение 0.

Решение

X	x_1	x_2	x_3	x_4	x_5	Y
0	0	0	0	0	0	**0**
1	0	0	0	0	1	**0**
2	0	0	0	1	0	**1**
3	0	0	0	1	1	**0**
4	0	0	1	0	0	**0**
5	0	0	1	0	1	**0**
6	0	0	1	1	0	**0**
7	0	0	1	1	1	**0**
8	0	1	0	0	0	**0**
9	0	1	0	0	1	**1**
10	0	1	0	1	0	**0**
11	0	1	0	1	1	**0**
12	0	1	1	0	0	**0**
13	0	1	1	0	1	**0**
14	0	1	1	1	0	**0**
15	0	1	1	1	1	**0**
16	1	0	0	0	0	**0**
17	1	0	0	0	1	**1**
18	1	0	0	1	0	**0**
19	1	0	0	1	1	**0**
20	1	0	1	0	0	**0**
21	1	0	1	0	1	**1**
22	1	0	1	1	0	**0**
23	1	0	1	1	1	**0**
24	1	1	0	0	0	**0**
25	1	1	0	0	1	**1**
26	1	1	0	1	0	**0**
27	1	1	0	1	1	**0**
28	1	1	1	0	0	**0**
29	1	1	1	0	1	**0**
30	1	1	1	1	0	**0**
31	1	1	1	1	1	**0**

```
P = zeros(32, 5);
T = zeros(32, 1);

% Fill the matrix P
for i = 1:length(P)
  if mod(i, 2) == 0
     P(i, 5) = 1;
  end
  if mod(fix((i - 1)/ 2), 2) == 1
     P(i, 4) = 1;
  end
  if mod(fix((i - 1)/ 4), 2) == 1
     P(i, 3) = 1;
  end
  if mod(fix((i - 1)/ 8), 2) == 1
     P(i, 2) = 1;
  end
  if mod(fix((i - 1)/ 16), 2) == 1
     P(i, 1) = 1;
  end
end

% Fill the matrix T
T(2) = 1;
T(9) = 1;
T(17) = 1;
T(21) = 1;
T(25) = 1;
```

Исходные данные 3 – Разбиение плоскости на 2 класса

Задание

Прямоугольный участок плоскости с помощью отрезков прямых линий разбейте на два класса так, чтобы

- хотя бы один из классов состоял из нескольких непересекающихся частей;
- классы были линейно неразделимы.

Приведите графический эскиз полученного разбиения плоскости.

Решение

Выбраны области, формирующие фигуру. Написана программа, в которой определена матрица точек. Так же в программе выполняется графическое представление выбранных областей, вызов функции определения класса, а также визуализация работы функции определения класса. Функция определения принадлежности к классу определена в отдельном файле *is_in_area(T)*.

*Программа **is_in_area***

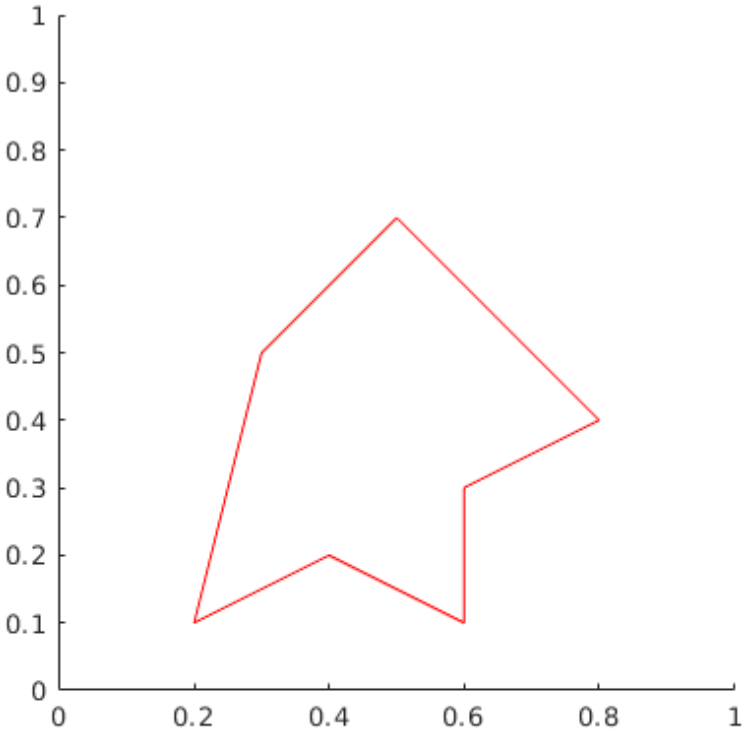
```
function res = is_in_area(T)
M = [  0.2, 0.1; 0.3, 0.5; .4, 0.6; 0.5, 0.7;
   .8, 0.4; 0.6, 0.3; .6, 0.1; .4, 0.2; .2, 0.1];

   n = length(T);
   res = zeros(n, 1);
   for i = 1:n
      res(i) = inpolygon(T(i,1), T(i,2), M(:,1), M(:,2));
   end
   res = res';
end
```

*Программа **thirdData***

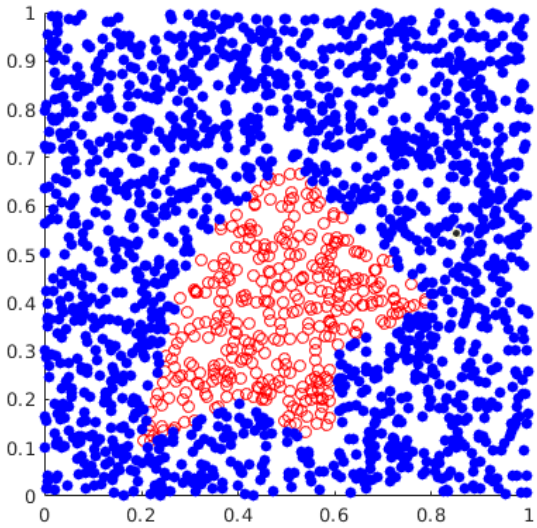
```
Figure = [  0.2, 0.1; 0.3, 0.5; .4, 0.6; 0.5, 0.7;
   .8, 0.4; 0.6, 0.3; .6, 0.1; .4, 0.2; .2, 0.1];

figure
hold on
plot(Figure(:,1), Figure(:,2), 'r');
axis([0 1 0 1])
axis square
hold off

r = rand(2, 2000)';
res = is_in_area(r);
num_of_ones = sum(res(:) == 1);
num_of_zeros = sum(res(:) == 0);

ones_ = zeros(num_of_ones, 2);
zeros_ = zeros(num_of_zeros, 2);

j = 1;
k = 1;
for i = 1:length(res)
   if res(i) == 1
      ones_(j, 1) = r(i, 1);
      ones_(j, 2) = r(i, 2);
      j = j + 1;
   else
      zeros_(k, 1) = r(i, 1);
      zeros_(k, 2) = r(i, 2);
      k = k + 1;
   end
end

figure
hold on
scatter(ones_(:,1), ones_(:,2), 'r');
```

```
scatter(zeros_(:,1), zeros_(:,2), 'b', 'filled');
axis([0 1 0 1])
axis square
hold off
```

Визуализация (разбиение пространства на два класса)



Визуализация (работы функции определения класса)



Исходные данные 4 – Разбиение плоскости на n классов

Задание

Прямоугольный участок плоскости с помощью отрезков прямых линий разбейте на 7-8 классов так, чтобы

- хотя бы два класса состояли из нескольких непересекающихся частей;
- несколько пар классов были линейно неразделимы.

Приведите графический эскиз полученного разбиения плоскости.

Решение

Плоскость была разделена на 5 областей: 4 заданы различными фигурами, а 5-ая область – это все остальное пространство. Была написана функция *is_in_area2(T, type)*, которая определяет принадлежность точки к одному из 5 классов.

*Программа **is_in_area2***

```
function res = is_in_area2(T, type)
A1 = [  0.0, 0.0; 0.6, 0.0; 0.6, 0.25; 0.0, 0.25; 0.0, 0.0 ];
A2 = [ 0.25, 0.3; 0.5, 0.3; 0.5, 0.6; 0.25, 0.6; 0.25, 0.3 ];
A3 = [ 0.15, 0.9; 0.65, 0.8; 0.375, 0.75; 0.15, 0.9];
A4 = [ 0.80, 0.25; 0.9, 0.25; 0.9, 0.55; 0.8, 0.55; 0.8, 0.25 ];
   n = length(T);
if type == 1
   res = zeros(n, 1);
else
   res = zeros(n, 5);
end

   for i = 1:n
      if inpolygon(T(i,1), T(i,2), A1(:,1), A1(:,2))
         if type == 1
            res(i) = 1;
         else
            res(i, 1) = 1;
         end
      elseif inpolygon(T(i,1), T(i,2), A2(:,1), A2(:,2))
         if type == 1
            res(i) = 2;
         else
            res(i, 2) = 1;
         end
      elseif inpolygon(T(i,1), T(i,2), A3(:,1), A3(:,2))
         if type == 1
            res(i) = 3;
         else
            res(i, 3) = 1;
         end
      elseif inpolygon(T(i,1), T(i,2), A4(:,1), A4(:,2))
         if type == 1
            res(i) = 4;
         else
            res(i, 4) = 1;
         end
      else
         if type == 1
            res(i) = 5;
         else
            res(i, 5) = 1;
         end
      end
   end
```

```
  res = res';
end
```

*Программа **fourthData***

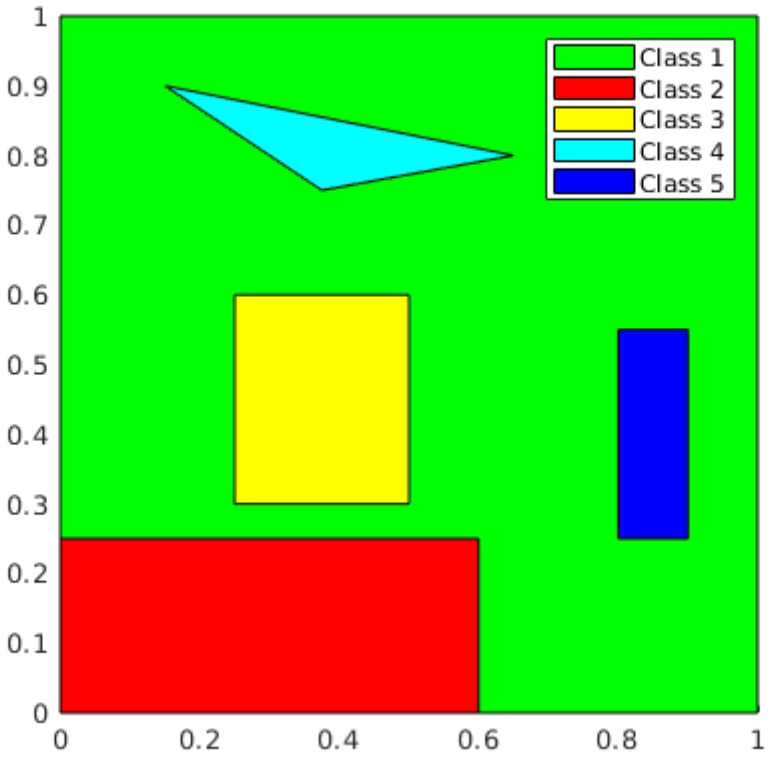
```
A1 = [  0.0, 0.0; 0.6, 0.0; 0.6, 0.25; 0.0, 0.25; 0.0, 0.0 ];
A2 = [ 0.25, 0.3; 0.5, 0.3; 0.5, 0.6; 0.25, 0.6; 0.25, 0.3 ];
A3 = [ 0.15, 0.9; 0.65, 0.8; 0.375, 0.75; 0.15, 0.9];
A4 = [ 0.80, 0.25; 0.9, 0.25; 0.9, 0.55; 0.8, 0.55; 0.8, 0.25 ];

figure
hold on
fill([0 1 1 0], [0 0 1 1], 'g');
fill(A1(:, 1), A1(:, 2), 'r');
fill(A2(:, 1), A2(:, 2), 'y');
fill(A3(:, 1), A3(:, 2), 'c');
fill(A4(:, 1), A4(:, 2), 'b');
legend([ "Class 1" "Class 2" "Class 3" "Class 4" "Class 5"]);
axis([0 1 0 1])
axis square
hold off

r = rand(2, 5000)';

res = is_in_area2(r, 2);
plot5classes(r, res);
```

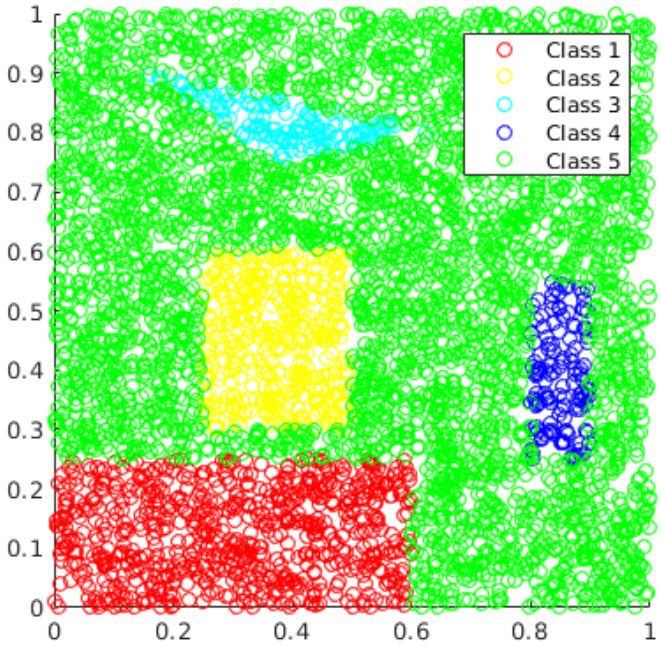
*Программа **plot5classes***

```
function plot5classes(X, res)
num_of_ones = sum(res(1, :) == 1);
num_of_twoes = sum(res(2, :) == 1);
num_of_trees = sum(res(3, :) == 1);
num_of_fours = sum(res(4, :) == 1);
num_of_fives = sum(res(5, :) == 1);

ones_ = zeros(num_of_ones, 2);
twoes_ = zeros(num_of_twoes, 2);
trees_ = zeros(num_of_trees, 2);
fours_ = zeros(num_of_fours, 2);
fives_ = zeros(num_of_fives, 2);

j = 1;
k = 1;
l = 1;
m = 1;
n = 1;
for i = 1:length(res)
  if res(1,i) == 1
    ones_(j,:) = X(i,:);
    j = j + 1;
  elseif res(2,i) == 1
    twoes_(k, :) = X(i, :);
    k = k + 1;
  elseif res(3,i) == 1
    trees_(l, :) = X(i, :);
    l = l + 1;
  elseif res(4,i) == 1
    fours_(m, :) = X(i, :);
```

```
      m = m + 1;
    else
      fives_(n, :) = X(i, :);
      n = n + 1;
    end
end

figure
hold on
scatter(ones_(:,1), ones_(:,2), 'r');
scatter(twoes_(:,1), twoes_(:,2), 'y');
scatter(trees_(:,1), trees_(:,2), 'c');
scatter(fours_(:,1), fours_(:,2), 'b');
scatter(fives_(:,1), fives_(:,2), 'g');
legend(["Class 1" "Class 2" "Class 3" "Class 4" "Class 5"]);
axis([0 1 0 1])
axis square
hold off
end
```

Визуализация (разбиение пространства на n классов)





Исходные данные 5 – Непрерывная функция одной переменной

Задание

Определите функцию одной переменной в некотором интервале входных значений. Функция должна иметь умеренную сложность:

- как минимум 15-20 минимумов и максимумов;
- наличие колебаний различной частоты в различных диапазонах входных значений;
- наличие нескольких изломов.

Приведите графический эскиз полученной функции.

Решение

Была выбрана функция $y = \sin\left(2^{(5-x)}\right)$. Была написана функция *my_fun(T),* которая принимает массив координат по x и возвращает массив значений. Будем использовать значение функции только в диапазоне [-1, 1].

*Программа **my_fun***

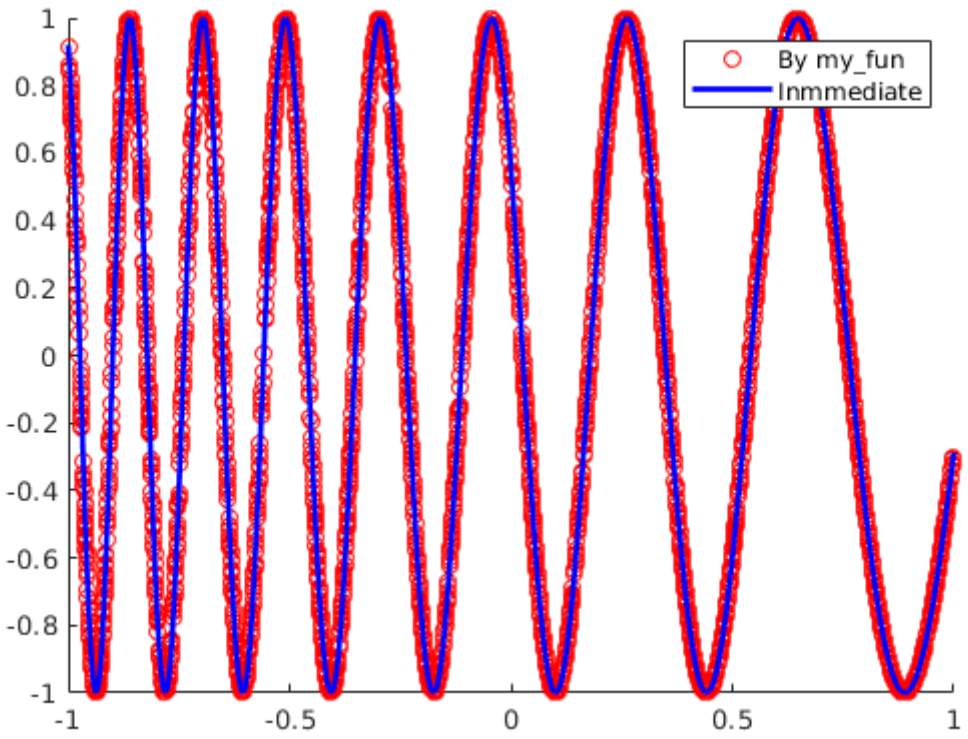
```
function res = my_fun(T)
   n = length(T);
   res = zeros(n, 1)';
   for i = 1:n
      res(i) = sin(2.^(5-T(i)));
   end
end
```

*Программа **fifthData***

```
X = rand(1, 5000);
X = cat(2, -X(1:2500), X(2501:end) );
Y = my_fun(X);

x = linspace(-1, 1, 1000);
y =  sin(2.^(5-x));
figure
hold on
scatter(X, Y, 'r');
plot(x, y, 'b', 'LineWidth', 2);
legend(["By my\_fun" "Inmmediate"]);
hold off
```

Визуализация

Синяя линия – значение, полученное напрямую. Красные точки – значения, полученные с помощью функции *my_fun*.



Исходные данные 8 – Многомерные образы

Задание

Сформируйте по $N = 10\text{-}20$ примеров для $M = 5\text{-}10$ различных классов, представленных в форме изображений невысокой размерности (10x10 или 20x20). Изображения могут быть ЧБ или цветными (буквы, символы, лица, объекты произвольного типа). Можно воспользоваться имеющимися в интернете базами данных.

Решение

Было выбрано 2 класса по несколько примеров на каждый класс. Классы: дома и лошади. Была написана программа для чтения этих изображений с заданной размерностью 20x20 и отображении их в матрицы градаций серого.

*Программа **EightData***

```
addpath('../neural/examples/houses');
addpath('../neural/examples/horses');
houseImages = dir('../neural/examples/houses/*.jpg');
horseImages = dir('../neural/examples/horses/*.jpg');
unix(pwd);

nfiles = length(houseImages);    % Number of files found
houses = {};
for i=1:nfiles
  house = imresize(rgb2gray(imread(houseImages(i).name)), [20 20]);
  houses{end+1} = house;
end


nfiles = length(horseImages);    % Number of files found
horses = {};
for i=1:nfiles
  horse = imresize(rgb2gray(imread(horseImages(i).name)), [20 20]);
  horses{end+1} = horse;
end
```

Пример одной из сформированных матриц (для house)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	196	194	185	187	194	203	204	199	200	200	213	200	167	75	104	56	53	63	103	94
2	185	183	181	182	199	208	209	213	212	212	172	192	171	60	84	67	46	97	134	87
3	168	117	157	156	145	158	169	180	207	190	126	146	199	112	45	50	93	141	163	136
4	116	72	107	115	112	109	103	126	187	147	169	136	172	168	85	65	134	110	125	144
5	68	62	69	89	112	107	118	189	148	145	184	159	127	189	193	132	85	60	86	91
6	63	64	56	76	106	110	181	158	135	143	160	151	134	138	204	149	62	74	60	80
7	51	53	58	62	94	152	176	127	127	126	122	124	125	111	142	135	63	43	50	73
8	64	60	52	59	82	178	198	195	194	191	185	181	177	172	174	186	170	118	52	60
9	69	39	63	61	61	159	181	144	136	146	174	195	177	158	158	177	187	90	36	60
10	42	30	60	58	64	174	133	62	53	52	61	188	92	62	55	86	176	70	45	58
11	36	57	39	81	95	194	159	104	82	66	65	200	125	111	95	75	182	95	30	42
12	31	43	31	106	104	209	173	111	100	77	80	217	138	119	103	93	200	102	23	46
13	27	21	36	96	105	222	175	108	86	53	96	229	137	92	96	107	209	111	27	52
14	23	26	52	84	97	160	140	132	65	30	90	150	131	91	80	76	140	112	44	52
15	80	82	95	96	75	158	110	116	82	42	94	105	122	80	61	61	140	146	102	103
16	92	80	76	104	127	154	110	176	193	180	167	95	173	179	171	172	172	149	114	113
17	91	67	52	67	92	99	120	241	255	255	255	154	146	153	131	151	122	95	98	110
18	92	94	81	70	59	58	96	231	244	243	238	185	81	71	67	78	79	75	79	94
19	78	87	102	96	91	85	100	126	131	156	165	150	118	103	102	104	106	106	106	103
20	67	73	68	70	76	84	100	96	97	101	129	158	159	145	119	99	100	99	99	100

Задание 3

Исходные данные типа 3 – определение качества классификации, 2 класса

Задание

Шаг 1. Сформируйте выборку (*P*,*T*) объемом *N*, как в задании 2.
Шаг 2. Проинвертируйте метки классов для *k* (5, 10, 20) % случайно взятых примеров. Далее интерпретируйте эти данные, как ответ *Y* некоторого распознающего устройства (нейронной сети).
Шаг 3. На основании желаемых *T* и реальных *Y* ответов определите основные показатели качества распознавания:

- матрицу неточностей;
- среднюю вероятность ошибки и среднюю вероятность правильного распознавания;
- ошибки первого и второго рода, чувствительность, специфичность.

Убедитесь, что средняя ошибка совпадает со значением *k*.

Решение

Для выполнения данного задания была написана программа *task3/part1*, а так же вспомогательные программы (файлы-функции): *task3/confmatr* – для вычисления матрицы неточностей, *task3/averrprob* – для вычисления средней вероятноти ошибки, *task3/avrightprob* – для вычисления вероятности правильного распознавания, *task3/alphaerr* – для вычисления ошибки 1-го рода, *task3/falsepos* – для вычисления ошибки 2-го рода. Далее приведены исходные коды программ. Программы с комментариями можно посмотреть в директории *task3*.

*Программа **confmatr***

```
function C = confmatr(T, Y, k)
  l = length(k);
  C = zeros(l, l);

  for i = 1:length(T)
    indC = find(k==T(i));
    indR = find(k==Y(i));
    C(indR, indC) = C(indR, indC) + 1;
  end
end
```

*Программа **averrprob***

```
function r = averrprob(T)
  [m, ~] = size(T);
  nondiag = 0;
  all = 0;
  for i = 1:m
    sumi = sum(T(i,:));
    all = all + sumi;
    nondiag = nondiag + (sumi - T(i, i));
  end
  r = nondiag / all;
end
```

*Программа **avrightprob***

```
function r = avrightprob(T)
  [m, ~] = size(T);
  diag = 0;
  all = 0;
  for i = 1:m
    all = all + sum(T(i,:));
    diag = diag + T(i, i);
  end
  r = diag / all;
end
```

*Программа **alphaerr***

```
function r = alphaerr(T)
  [m, n] = size(T);
  r = zeros(1, n);
  for i = 1:n
    err = 0;
    all = 0;
    for j = 1:m
      if i ~= j; err = err + T(j, i); end
      all = all + T(j, i);
    end
    r(i) = err / all;
  end
end
```

*Программа **falsepos***

```
function r = falsepos(T)
  [m, n] = size(T);
  r = zeros(1, n);
  for i = 1:n
    err = 0;
    all = 0;
    for j = 1:m
      if i ~= j; err = err + T(i, j); end
      all = all + T(i, j);
    end
    r(i) = err / all;
  end
end
```

```
invert = @(e) (e ~= 1);
rng(777);
N = 1000;
r = rand(2, N)';
res = is_in_area(r);

inverted5 = res;
inverted10 = res;
inverted20 = res;

rand5indicies = randperm(N, round(N * 0.05));
rand10indicies = randperm(N, round(N * 0.1));
rand20indicies = randperm(N, round(N * 0.2));

for index = rand5indicies
   inverted5(index) = invert(res(index));
end
for index = rand10indicies
   inverted10(index) = invert(res(index));
end
for index = rand20indicies
   inverted20(index) = invert(res(index));
end

C5 = confmatr(res, inverted5, [0 1]);
C10 = confmatr(res, inverted10, [0 1]);
C20 = confmatr(res, inverted20, [0 1]);

AERR5 = averrprob(C5);
AERR10 = averrprob(C10);
AERR20 = averrprob(C20);

ARIG5 = avrightprob(C5);
ARIG10 = avrightprob(C10);
ARIG20 = avrightprob(C20);

ALPERR5 = alphaerr(C5);
ALPERR10 = alphaerr(C10);
ALPERR20 = alphaerr(C20);

FALPOS5 = falsepos(C5);
FALPOS10 = falsepos(C10);
FALPOS20 = falsepos(C20);

SPECIFILITY5 = C5(1,1) / sum(sum(C5));
SPECIFILITY10 = C10(1,1) / sum(sum(C10));
SPECIFILITY20 = C20(1,1) / sum(sum(C20));

SENSITIVITY5 = C5(2,2) / sum(sum(C5));
SENSITIVITY10 = C10(2,2) / sum(sum(C10));
SENSITIVITY20 = C20(2,2) / sum(sum(C20));
```

Результаты

k	5%	10%	20%
Матрица неточностей	724 13 37 226	692 31 69 208	611 50 150 189
Средняя вероятность ошибки	0.05	0.1	0.2
Средняя вероятность правильного распознавания	0.95	0.9	0.8
Ошибка 1-го рода	0,0486 0,0544	0,0907 0,1297	0,1971 0,2092
Ошибка 2-го рода	0,0176 0,1407	0,0429 0,2491	0,0756 0,4425
Специфичность	0.7240	0.6920	0.6110
Чувствительность	0.2260	0.2080	0.1890

Исходные данные типа 4 – определение качества классификации, N классов

Задание

Шаг 1. Сформируйте выборку (*P*,*T*) объемом *N*, как в задании 2.
Шаг 2. Измените метки классов на случайные другие (равномерно распределенные) для *k* (5, 10, 20) % случайно взятых примеров. Далее интерпретируйте эти данные, как ответ *Y* некоторого распознающего устройства (нейронной сети).
Шаг 3. На основании желаемых *T* и реальных *Y* ответов определите основные показатели качества распознавания:

- матрицу неточностей;
- среднюю вероятность ошибки и среднюю вероятность правильного распознавания;
- ошибки первого и второго рода для каждого класса.
  Убедитесь, что средняя ошибка совпадает со значением *k*.

Решение

Для решения данного задания была написана программа *task3/part2*, а также вспомогательная программа *task3/randfrmnoteq* – которая выдает случайное число, не равное заданному, из массива. В данном пункте используются программы из пердыдущего пункта.

*Программа **randfrmnoteq***

```
function res = randfrmnoteq(x, c)
  x(x==c) = [];
  res = x(randperm(length(x), 1));
end
```

*Программа **part2***

```
rng(777);
N = 1000;
r = rand(2, N)';
res = is_in_area2(r, 1);

inverted5 = res;
inverted10 = res;
inverted20 = res;

rand5indicies = randperm(N, round(N * 0.05));
rand10indicies = randperm(N, round(N * 0.1));
rand20indicies = randperm(N, round(N * 0.2));

for index = rand5indicies
   inverted5(index) = randfrmnoteq(res, res(index));
end

for index = rand10indicies
   inverted10(index) = randfrmnoteq(res, res(index));
end

for index = rand20indicies
   inverted20(index) = randfrmnoteq(res, res(index));
end

k = [1 2 3 4 5];
C5 = confmatr(res, inverted5, k);
C10 = confmatr(res, inverted10, k);
C20 = confmatr(res, inverted20, k);

AERR5 = averrprob(C5);
AERR10 = averrprob(C10);
AERR20 = averrprob(C20);

ARIG5 = avrightprob(C5);
ARIG10 = avrightprob(C10);
ARIG20 = avrightprob(C20);

ALPERR5 = alphaerr(C5);
ALPERR10 = alphaerr(C10);
ALPERR20 = alphaerr(C20);

FALPOS5 = falsepos(C5);
FALPOS10 = falsepos(C10);
FALPOS20 = falsepos(C20);
```

Результаты

Матрица неточностей для $k = 5\,\%$

136	1	0	0	14
1	72	0	0	11
0	0	16	0	1
0	0	0	29	5
12	1	3	1	697

Матрица неточностей для $k = 10\%$

133	1	1	0	38
0	63	0	0	16
0	0	17	0	2
0	1	0	27	12
16	9	1	3	660

Матрица неточностей для $k = 20\%$

112	5	0	0	81
9	59	0	0	41
1	0	18	0	8
5	1	0	25	12
22	9	1	5	586

k	5%	10%	20%
Средняя вероятность ошибки	0.05	0.1	0.2
Средняя вероятность правильного распознавания	0.95	0.9	0.8
Ошибка 1-го рода (для столбцов)	0,0993 0,1429 0,0588 0,1471 0,0238	0,2312 0,2025 0,1053 0,3250 0,0421	0,4343 0,4587 0,3333 0,4186 0,0594
Ошибка 2-го рода (для столбцов)	0,0872 0,0270 0,1579 0,0333 0,0426	0,1074 0,1486 0,1053 0,1000 0,0934	0,2483 0,2027 0,0526 0,1667 0,1951

Исходные данные типа 5 – определение качества аппроксимации

Задание

Шаг 1. Сформируйте выборку (*P*,*T*) объемом *N*, как в задании 2.
Шаг 2. Добавьте к значениям *T* равномерный шум различной амплитуды (5, 10, 20 % от максимального значения). Далее интерпретируйте полученный сигнал, как ответ *Y* некоторого распознающего устройства (нейронной сети).
Шаг 3. На основании желаемых *T* и реальных *Y* ответов определите основные показатели качества распознавания:

- среднюю абсолютную ошибку;
- среднюю относительную ошибку;
- максимальную по модулю ошибку.

Сравните полученные значения ошибок и убедитесь, что они соответствуют исходному уровню шума.

Решение

Для решения данного задания была написана программа *task3/part3*, а также вспомогательная программа *task3/makesomenoise* – для генерации белого шума.

*Программа **makesomenoise***

```
function res = makesomenoise(x, maxval)
   len = size(x);
   std_noise = random('Uniform', -maxval, maxval, len);
   res = x + std_noise;
end
```

*Программа **part3***

```
rng(777);
N = 10000;
X = rand(1, N);
X = cat(2, -X(1:N/2), X(N/2 + 1:end));
Y = my_fun(X);
maxv = max(Y);
percofmax5 = 0.05 * maxv;
percofmax10 = 0.1 * maxv;
percofmax20 = 0.2 * maxv;
noised5 = makesomenoise(Y, percofmax5);
noised10 = makesomenoise(Y, percofmax10);
noised20 = makesomenoise(Y, percofmax20);
avabserr5 = sum(abs(Y - noised5)) / N;
avabserr10 = sum(abs(Y - noised10)) / N;
avabserr20 = sum(abs(Y - noised20)) / N;
avrelerr5 = sum(abs(Y - noised5) ./ maxv) / N;
avrelerr10 = sum(abs(Y - noised10) ./ maxv) / N;
avrelerr20 = sum(abs(Y - noised20) ./ maxv) / N;
maxerr5 = max(abs(Y - noised5));
maxerr10 = max(abs(Y - noised10));
maxerr20 = max(abs(Y - noised20));
```

k	5%	10%	20%
Средняя абсолютная ошибка	0.0249	0.0499	0.1003
Средняя относительная ошибка	0.0249	0.0499	0.1003
Максимальная по модулю ошибка	0.0500	0.1000	0.2000

Полученные средние относительные ошибки равны половине исходного шума, это связано с тем, что накладываемый шум равномерно распределен от 0 до $maxv * k$.

Задание 4
Исходные данные типа 3 – кросс-валидация

Задание

Шаг 1. Сформируйте выборку (P,T) объемом N, как в задании 2.
Шаг 2. Разделите выборку на обучающую и тестовую, выбрав случайно k% примеров как тестовые, а остальные – как обучающие.
Шаг 3. Выполните визуализацию, как в задании 2, при этом отобразив тестовые и обучающие примеры разными символами.
Шаг 4. Примените K-fold кроссвалидацию (k=4,8) к исходной выборке. Для этого перемешайте выборку, разделите ее на k частей и далее сформируйте разбиения всей выборки на подвыборки так, чтобы одна часть была тестовой, а все остальные – обучающие. Визуализируйте полученные разбиения подходящим способом.

Решение

Для выполнения задания были написана программа *part4* и вспомогательная программа для построения графиков – *myplot*.

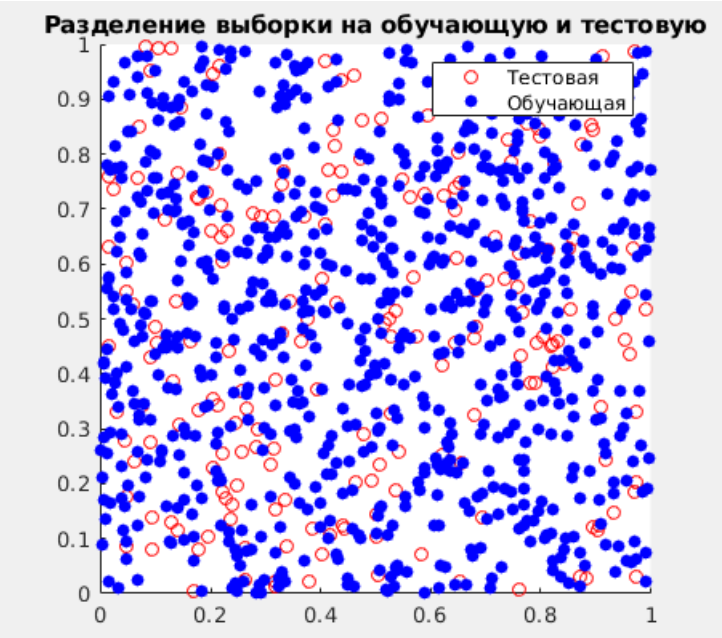
*Программа **myplot***

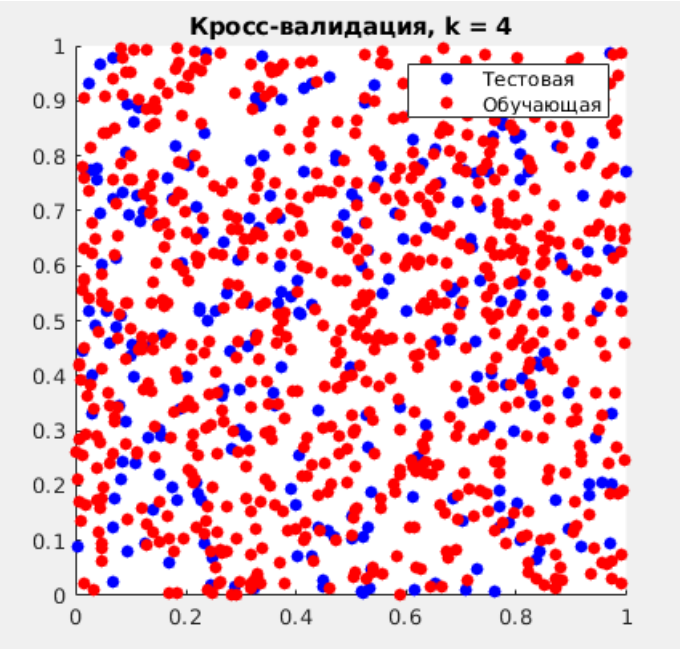
```
function myplot(X, testing, title_)
   figure
   hold on
   title(title_);
   [~,~,l] = size(X);
   scatter(X(1,:,testing), X(2,:,testing), 'b', 'filled');
   for i = 1:l
      if i ~= testing
         scatter(X(1,:,i), X(2,:,i), 'r', 'filled');
      end
   end
   axis([0 1 0 1])
   axis square
   legend(["Тестовая" "Обучающая"]);
   hold off
end
```

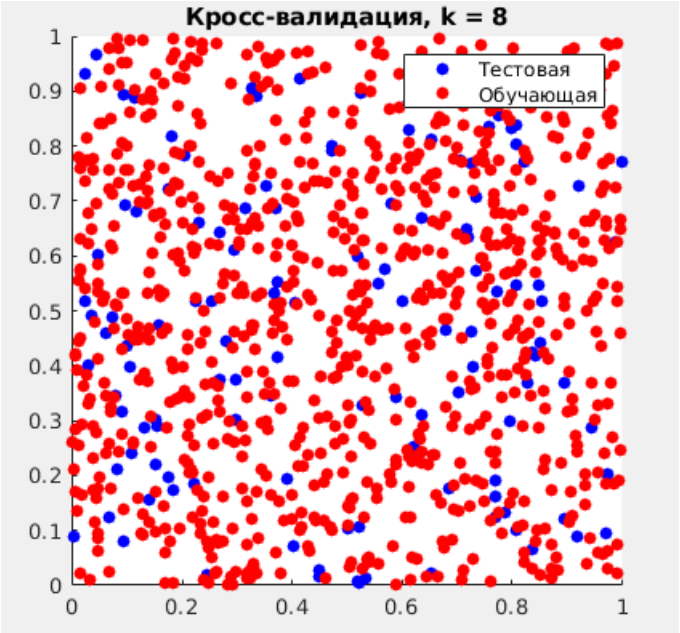
*Программа **part4***

```
rng(777);
N = 1000;
r = rand(2, N)';
K = 0.2;
testindecies = randperm(N, round(N * K));
baseindecies = 1:N;
baseindecies(testindecies) = [];
test = r(testindecies, :);
base = r(baseindecies, :);

figure
hold on
title('Разделение выборки на обучающую и тестовую');
scatter(test(:,1), test(:,2), 'r');
scatter(base(:,1), base(:,2), 'b', 'filled');
axis([0 1 0 1])
axis square
legend(["Тестовая" "Обучающая"]);
hold off

k = 4;
reshaped4 = reshape(r', [2 N/k k]);
k = 8;
reshaped8 = reshape(r', [2 N/k k]);

myplot(reshaped4, 1, 'Кросс-валидация, k = 4');
myplot(reshaped8, 1, 'Кросс-валидация, k = 8');
```

Визуализация




Кросс-валидация, k = 4
Тестовая
Обучающая
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
0.2
0.4
0.6
0.8
1


Кросс-валидация, k = 8
Тестовая
Обучающая
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
0.2
0.4
0.6
0.8
1