Министерство образования Республики Беларусь
Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей
Кафедра программного обеспечения информационных технологий
Дисциплина: Методы и алгоритмы принятия решений (МиАПР)

ОТЧЕТ
по лабораторной работе №4

по теме:
«Классификация объектов на N классов методом персептрона»

Выполнил
студент: гр. 851006 Верещагин Н.В.

Проверил: Марина И.М.

Минск 2021

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ

1.1 Цель работы

Изучить особенности классификации объектов методом персептрона, а также научиться применять этот метод на практике.

1.2 Исходные данные

– N – Количество классов на которые требуется разделить объекты

– Обучающая выборка, представленная векторами с наборами признаков.

1.3 Результат работы алгоритма

N –решающих функций.

2 АЛГОРИТМ

После того, как получено N решающих функций, предъявляются объекты тестовой выборки, которые необходимо классифицировать, отнеся к одному из классов. Тестовый объект подставляется в каждую из решающих функций и относится к тому из классов, где было получено максимальное значение.

Количество классов, обучающих объектов и их признаков может быть произвольным.

Допустим существование M решающих функций, характеризующихся тем свойством, что при $x \in \omega_i$, где x – объект, ω_i – класс $d_i(x) > d_j(x)$ для всех $i \neq j$.

Рассмотрим M классов $\omega_1, \omega_2, ..., \omega_M$. Пусть на к-м шаге процедуры обучения системе предъявляется образ $x(\ k\ )$, принадлежащий классу ω_i. Вычисляются значения M решающих функций $d_j[x(k)] = w_j(k)x(k), j = 1, 2, .., M$. Затем если выполняются условия $d_i[x(k)] > d_j[x(k)],\ j = 1, 2, ..., M; j \neq i,$ то векторы весов не изменяются, т. е. $w_j(k+1) = w_j(k), j = 1, 2, ..., M$.

С другой стороны, допустим, что для некоторого l $d_i[x(k)] \leq d_l[x(k)]$. В этом случае выполняются следующие коррекции весов:

$$\begin{aligned} &w_i(k+1) = w_i(k) + cx(k), \\ &w_l(k+1) = w_l(k) - cx(k), \\ &w_j(k+1) = w_j(k), j = 1, 2, ..., M; j \neq i, j \neq l, \end{aligned} \tag{1}$$

где c – положительная константа.

Если при рассмотрении случая 3 классы разделимы, то доказано, что этот алгоритм сходится за конечное число итераций при произвольных начальных векторах. Рассмотрим это на примере.

Даны классы, причем каждый из них содержит один образ: $\omega_1 : \{(0, 0)\}$, $\omega_2 : \{(1,1\}$, $\omega_3 : \{(-1, 1)\}$. Дополним заданные образы: $(0, 0, 1), (1, 1, 1), (-1, 1, 1)$.

Выберем в качестве начальных векторов весов $w_1(1) = w_2(1) = w_3(1) = (0, 0, 0)$, положим c = 1 и, предъявляя образы в указанном порядке, получим следующее:

$$\begin{aligned} &d_1[x(1)] = w_1(1)x(1) = 0, \\ &d_2[x(1)] = w_2(1)x(1) = 0, \\ &d_3[x(1)] = w_3(1)x(1) = 0. \end{aligned}$$

Поскольку $x(1) \in \omega_1$ и $d_2[x(1)] = d_3[x(1)] = d_1[x(1)]$, первый весовой вектор увеличивается, а два других уменьшаются в соответствии с соотношениями (1), т. е.

$$w_1(2) = w_1(1) + x(1) = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix},$$

$$w_2(2) = w_2(1) - x(1) = \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix},$$

$$w_3(2) = w_3(1) - x(1) = \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix}.$$

Следующий предъявляемый образ $x(2) = (1, 1, 1)$ принадлежит классу ω_2. Для него получаем

$$w_1(2)x(2) = 1,\ w_2(2)x(2) = -1, w_3(2)x(2) = -1.$$

Поскольку все произведения больше либо равны $w_2(2)x(2)$, вводятся корректировки векторов коэффициентов:

$$w_1(3) = w_1(2) - x(2) = \begin{pmatrix} -1 \\ -1 \\ 0 \end{pmatrix},$$

$$w_2(3) = w_2(2) + x(2) = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix},$$

$$w_3(3) = w_3(2) - x(2) = \begin{pmatrix} -1 \\ -1 \\ -2 \end{pmatrix}.$$

Следующий предъявленный образ $x(3)$=(-1, 1, 1) принадлежит классу ω_3. Для него получаем $w_1(3)x(3) = 0,\ w_2(3)x(3) = 0, w_3(3)x(3) = -2$. Все эти произведения опять требуют корректировки:

$$w_1(4) = w_1(3) - x(3) = \begin{pmatrix} 0 \\ -2 \\ -1 \end{pmatrix},$$

$$w_2(4) = w_2(3) - x(3) = \begin{pmatrix} 2 \\ 0 \\ -1 \end{pmatrix},$$

$$w_3(4) = w_3(3) + x(3) = \begin{pmatrix} -2 \\ 0 \\ -1 \end{pmatrix}.$$

Поскольку в данном цикле итерации присутствовали ошибки, следует провести новый цикл. Положив $x(4) = x(1)$, $x(5) = x(2)$, $x(6) = x(3)$, получим $w_1(4)x(4) = -1$, $w_2(4)x(4) = -1, w_3(4)x(4) = -1$. Так как образ $x(4)$ принадлежит классу ω_1, то все произведения «неверны». Поэтому

$$w_1(5) = w_1(4) + x(4) = \begin{pmatrix} 0 \\ -2 \\ 0 \end{pmatrix},$$

$$w_2(5) = w_2(4) - x(4) = \begin{pmatrix} 2 \\ 0 \\ -2 \end{pmatrix},$$

$$w_3(5) = w_3(4) - x(4) = \begin{pmatrix} -2 \\ 0 \\ -2 \end{pmatrix}.$$

Следующий предъявленный образ $x(5) = (1, 1, 1)$ принадлежит классу ω_2. Соответствующие скалярные произведения равны $w_1(5)x(5) = -2$,

$w_2(5)x(5) = 0$, $w_3(5)x(5) = -4$. Образ $x(5)$ классифицирован правильно. Поэтому

$$w_1(6) = w_1(5) = \begin{pmatrix} 0 \\ -2 \\ 0 \end{pmatrix},$$

$$w_2(6) = w_2(5) = \begin{pmatrix} 2 \\ 0 \\ -2 \end{pmatrix},$$

$$w_3(6) = w_3(5) = \begin{pmatrix} -2 \\ 0 \\ -2 \end{pmatrix}.$$

Следующий образ $x(6)$=(-1, 1, 1) принадлежит классу ω_3, для него получаем $w_1(6)x(6) = -2, w_2(6)x(6) = -4, w_3(6)x(6) = -0$. Этот образ также классифицирован правильно, так что коррекции не нужны, т е.

$$w_1(7) = w_1(6) = \begin{pmatrix} 0 \\ -2 \\ 0 \end{pmatrix},$$

$$w_2(7) = w_2(6) = \begin{pmatrix} 2 \\ 0 \\ -2 \end{pmatrix},$$

$$w_3(7) = w_3(6) = \begin{pmatrix} -2 \\ 0 \\ -2 \end{pmatrix}.$$

Если продолжить процедуру обучения, рассматривая образы $x(7)$, $x(8)$, $x(9)$, можно убедиться, что в следующем полном цикле никакие коррекции не производятся. Поэтому искомые решающие функции имеют следующий вид:

$$d_1(x) = 0 \cdot x_1 - 2x_2 + 0 = -2x_2,$$
$$d_2(x) = 2x_1 - 0 \cdot x_2 - 2 = 2x_1 - 2,$$
$$d_3(x) = -2x_1 + 0 \cdot x_2 - 2 = -2x_1 - 2.$$

3 РЕШЕНИЕ ЗАДАЧИ

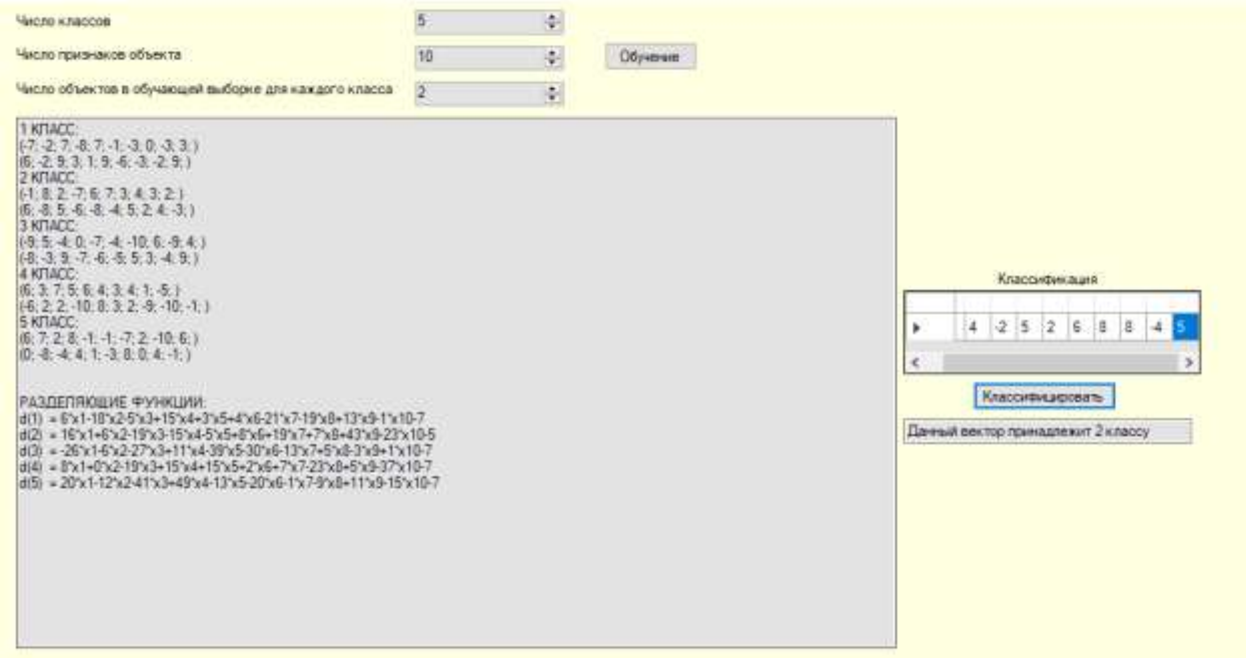

Рисунок 1 – Пример работы программы 1

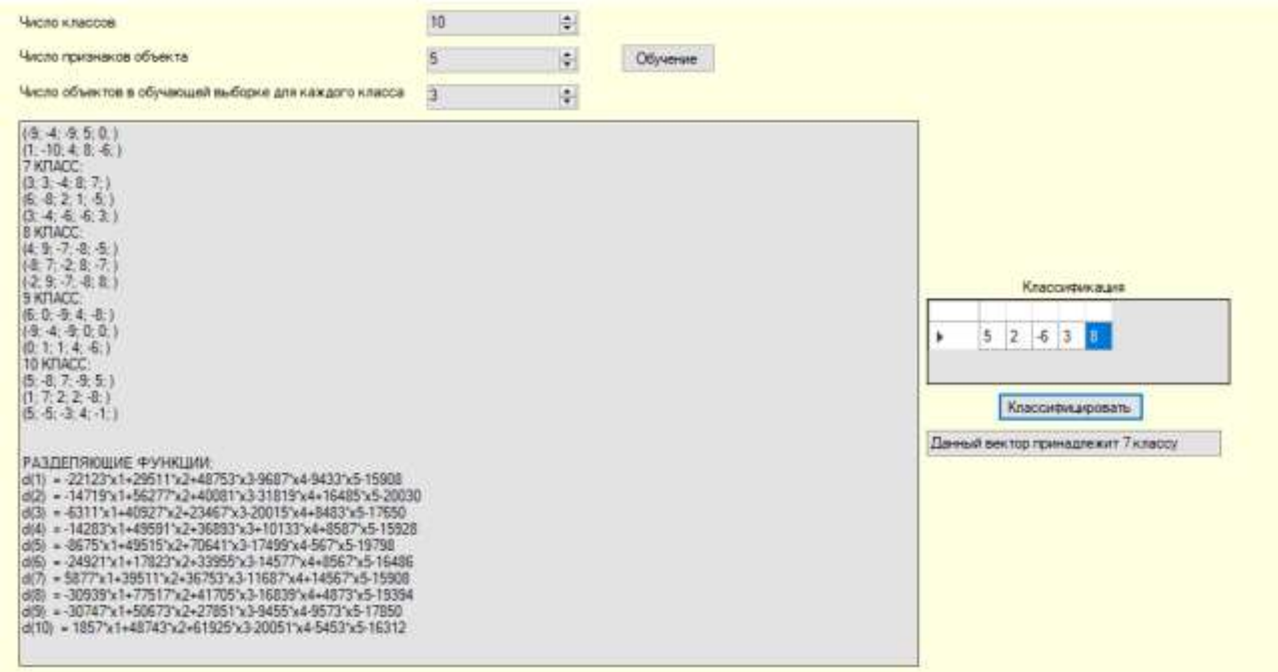

Рисунок 2 – Пример работы программы 2

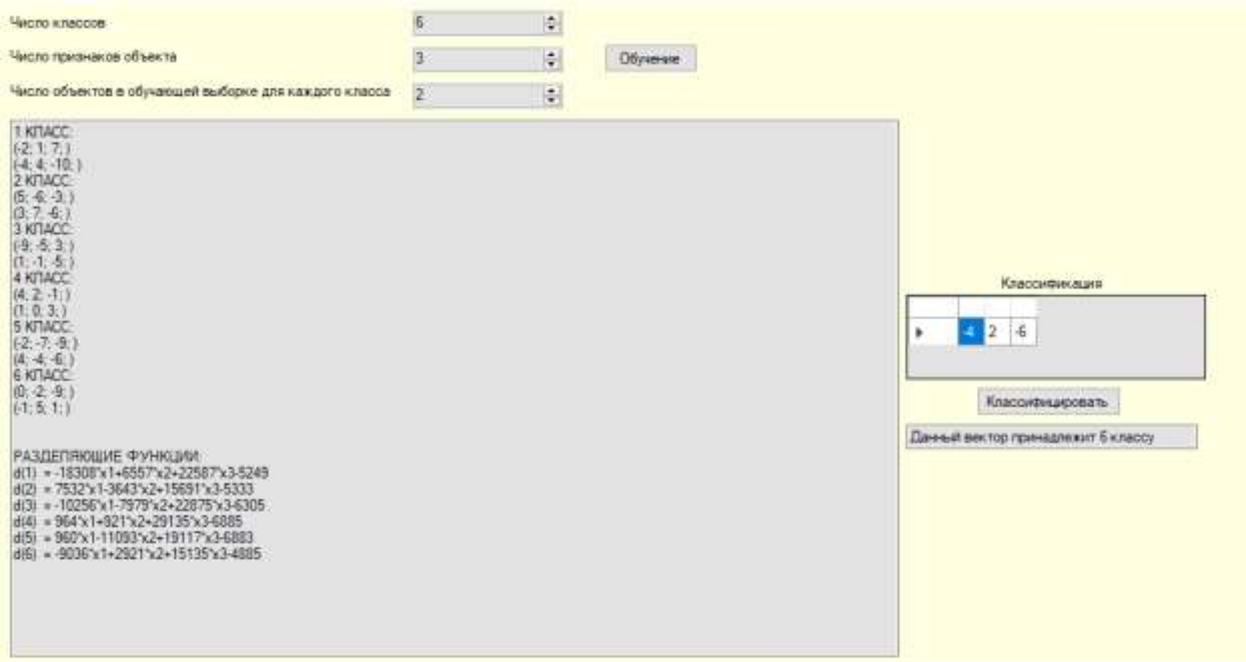

Рисунок 3 – Пример работы программы 3

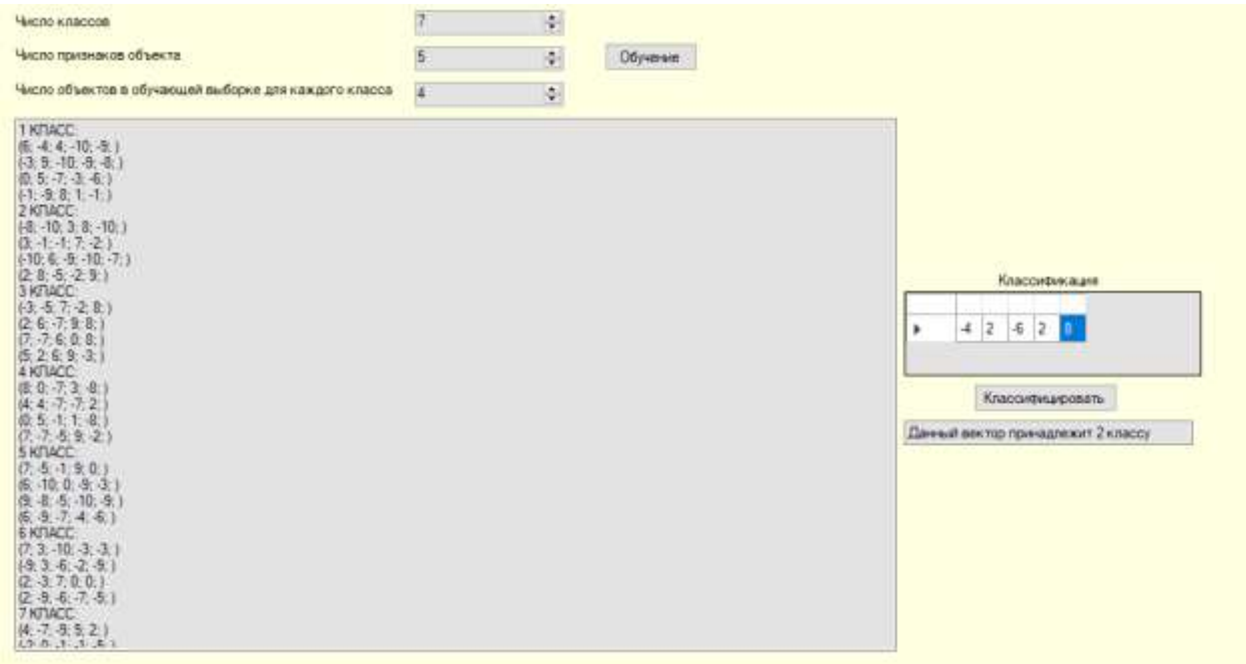

Рисунок 4 – Пример работы программы 4

Код программы:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
namespace Lab4
{
    public class Persiptron
    {
        private readonly int classCount;

        private readonly int vectorsSize;

        public bool warningFlag;

        public Persiptron(int classCount, int vectorsSize)
        {
            this.classCount = classCount;
            this.vectorsSize = vectorsSize;
        }

        public Function[] GetSepareteFunctions(Vector[][] teachingVectors)
        {
            var result = EmptyFunctions();
            warningFlag = false;
            bool nextIteration = true;
            int iterationNumber = 0;
            while (nextIteration && iterationNumber < 1000)
            {
                iterationNumber++;
                nextIteration = DoOneIteration(teachingVectors, result);
            }
            if (iterationNumber == 1000) warningFlag = true;
            return result;

        }

        private Function[] EmptyFunctions()
        {
            var result = new Function[classCount];
            for (int i = 0; i < classCount; i++)
            {
                result[i] = new Function(vectorsSize);
            }
            return result;
        }

        private bool DoOneIteration(Vector[][] teachingVectors, Function[] re-
sult)
        {
            bool nextIteration = false;
            if (teachingVectors.Length != classCount) throw new Exception();
            for (int classNumber = 0; classNumber < classCount; classNumber++)
            {
```

```
                for (int i = 0; i < teachingVectors[classNumber].Length; i++)
                {
                    if (WorkWithVector(teachingVectors[classNumber][i], re-
sult, classNumber))
                    {
                        nextIteration =true;
                    }
                }
            }
            return nextIteration;
        }

        private bool WorkWithVector(Vector currentVector, Function[] re-
sult, int vectorsClass)
        {
            var maxClass = GetMaxVectorClass(result, currentVector);
            if (maxClass != vectorsClass)
            {
                Panish(currentVector, result, vectorsClass);
                return true;
            }
            return false;
        }

        private void Panish(Vector currentVector, Function[] result, int vec-
torsClass)
        {
            for (int i = 0; i < classCount; i++)
            {
                if (i == vectorsClass)
                {
                    result[i] += currentVector;
                }
                else
                {
                    result[i] += -1*currentVector;
                }
            }
        }

        public int GetMaxVectorClass(Function[] result, Vector currentVector)
        {
            int max = result[0].GetValue(currentVector);
            int maxClass = 0;
            int maxCount = 1;
            for (int j = 1; j < classCount; j++)
            {
                int currentValue = result[j].GetValue(currentVector);
                if (currentValue > max)
                {
```

```
                    maxCount = 0;
                    max = currentValue;
                    maxClass = j;
                }
                if (currentValue == max)
                {
                    maxCount++;
                }
            }
            return maxCount == 1 ?  maxClass : -1;
        }
    }
}
```