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 №3

по теме:
«РАЗДЕЛЕНИЕ ОБЪЕКТОВ НА ДВА КЛАССА ПРИ ВЕРОЯТНОСТНОМ ПОДХОДЕ»

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 при вероятностном подходе и научиться находить ошибку классификации.

1.2 Исходные данные

1. Две случайные величины, распределенные по закону Гаусса.
2. Априорные вероятности отнесения каждой из случайных величин к первому из двух классов, в зависимости от того, для какого из них определяется ошибка классификации.

1.3 Результат работы алгоритма

Вероятность ложной тревоги, вероятность пропуска обнаружения ошибки, вероятность суммарной ошибки классификации. Результаты работы программы должны представляться в графическом виде.

2 АЛГОРИТМ

На основе апостериорных вероятностей можно разработать метод автоматической классификации. Примером апостериорной плотности вероятности является случай одномерного гауссового распределения, выражаемого формулой (1).

$$p(x/j)=\frac{1}{\sigma_j\sqrt{2\pi}}\exp[-\frac{1}{2}(\frac{x-\mu_j}{\sigma_j})^2]. \quad (1)$$

Плотность распределения является функцией двух параметров: μ_j – математическое ожидание и σ_j – среднеквадратичное отклонение. Эти параметры могут быть вычислены по N опытам, в каждом из которых измеряется величина x_k $(k=1,2,...N)$, а затем вычисляются $\hat{\mu}_j=\frac{1}{N}\sum_{k=1}^{N}x_k$; $\hat{\sigma}^2{}_j=\frac{1}{N}\sum_{k=1}^{N}(x_k-\hat{\mu}_j)^2$.

Пусть задано сепарабельное пространство признаков, которое по определению может быть разделено на классы. X – вектор, представляющий k-й класс, сепарабельного пространства. Априорная вероятность того, что X относится к классу с номером k, есть $P(X_k)$. Она считается заданной самой постановкой задачи.

Задача заключается в том, чтобы отнести неизвестный предъявляемый объект X к одному из известных классов C_k с минимальной ошибкой. Для этого выполняют n измерений в соответствии с признаками, выбранными надлежащим образом. В результате получают вектор измерений X_m, для которого можно найти условную вероятность или ее плотность: $p(X_m/C_k)$.

Решение об отнесении неизвестного объекта к классу с номером k можно считать оправданным, если для любого j выполняется условие

$$p(C_k/\vec{X}_m)\geq p(C_j/\vec{X}_m)\ \forall j.$$

Эти вероятности могут быть вычислены согласно теореме Бейеса по тем условным вероятностям $p(\vec{X}_m/C_k)$, которые получаются непосредственно в процессе измерений:

$$P(C_k/\vec{X}_m)=\frac{P(C_k)p(\vec{X}_m/C_k)}{p(X_m)},\ P(C_j/\vec{X}_m)=\frac{P(C_j)p(\vec{X}_m/C_j)}{p(X_m)}.$$

Откуда следует решающее правило:

$$P(C_k)p(\vec{X}_m/C_k)\geq P(C_j)p(\vec{X}_m/C_j).$$

Рассмотрим случай, когда весь набор возможных решений сводится к двум, т. е. предъявленный объект может быть отнесен к одному из двух имеющихся классов. На рис. 1 показаны плотности распределения случайной величины X_m в случае ее отнесения к классам C_1 и C_2. $P(C_1)$ – это вероятность отнесения X_m к классу C_1, а $P(C_2)$ – вероятность отнесения случайной величины к классу C_2. Рассмотрим вероятности ошибок, которые могут возникать при такой процедуре. Очевидно, что на прямой AB неравенство Бейеса выполняется, и можно заключить, что X_m принадлежит классу C_1.

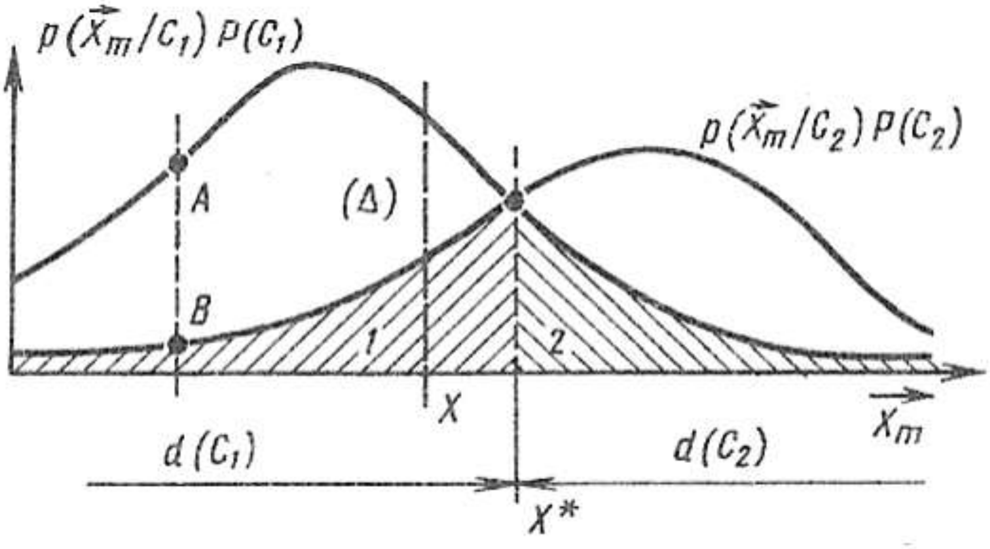

Рис. 1. Плотности распределения случайной величины

Рассмотрим линию раздела, обозначенную Δ. Любая точка, для которой $X_m < X$, считается принадлежащей классу C_1, в то время как все точки, для которых $X_m > X$, относятся к классу C_2. Однако вероятность того, что в первом случае точка может принадлежать классу C_2, отлична от нуля (область 1), так же как и то, что во втором случае точка X принадлежит классу C_1 (область 2). Для класса C_1 зона 1 является зоной ложной тревоги, а зона 2 является зоной пропуска обнаружения. Они определяются соответственно выражениями:

$$P_{л.т} = \int_{-\infty}^{x} P(C_2)\,p(\vec{X}_m / C_2)\,d\vec{X}_m; \qquad P_{п.о} = \int_{x}^{\infty} P(C_1)\,p(\vec{X}_m / C_1)\,d\vec{X}_m.$$

Суммарная ошибка классификации представляется суммой этих двух вероятностей. Если перемещать линию Δ, разделяющую два решения, вдоль оси X, то она должна достичь точки X^*, в которой имеет место равенство $P(C_1)\,p(\vec{X}_m / C_1) = P(C_2)\,p(\vec{X}_m / C_2)$, показывающее, что при бинарных ценах правило максимума правдоподобия обеспечивает оптимальную классификация по отношению к возможности ошибочного решения.

3 РЕШЕНИЕ ЗАДАЧИ

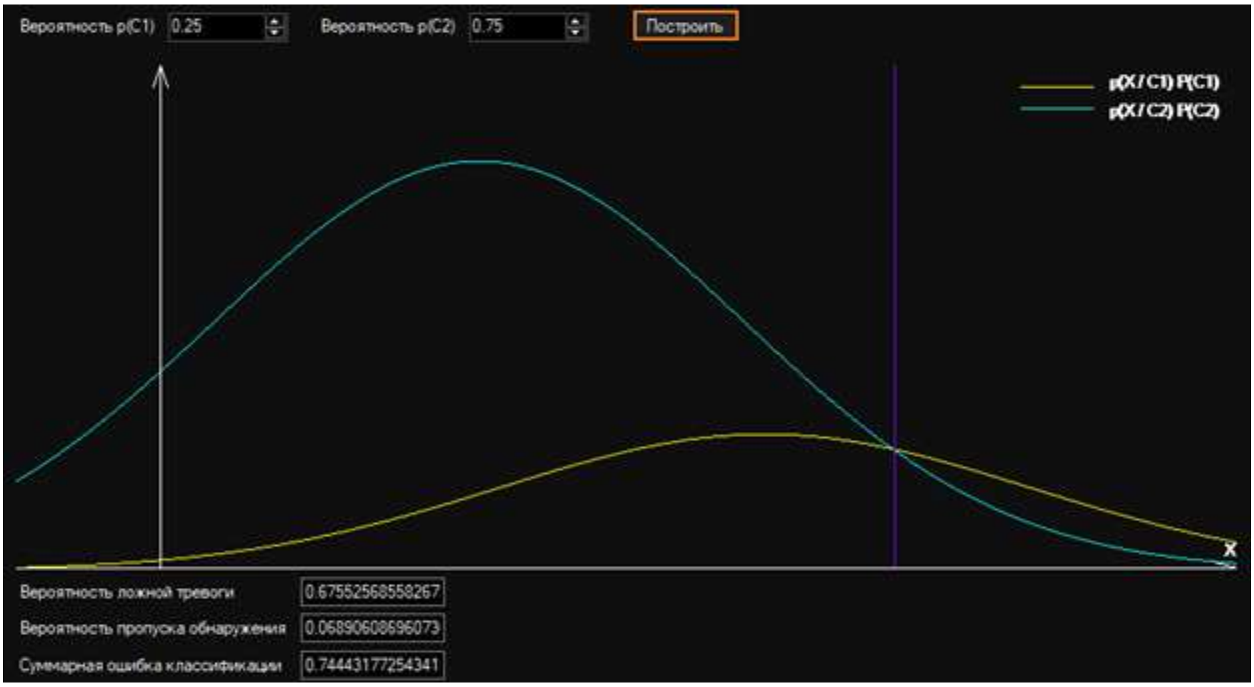

Рисунок 1 – Пример работы программы 1

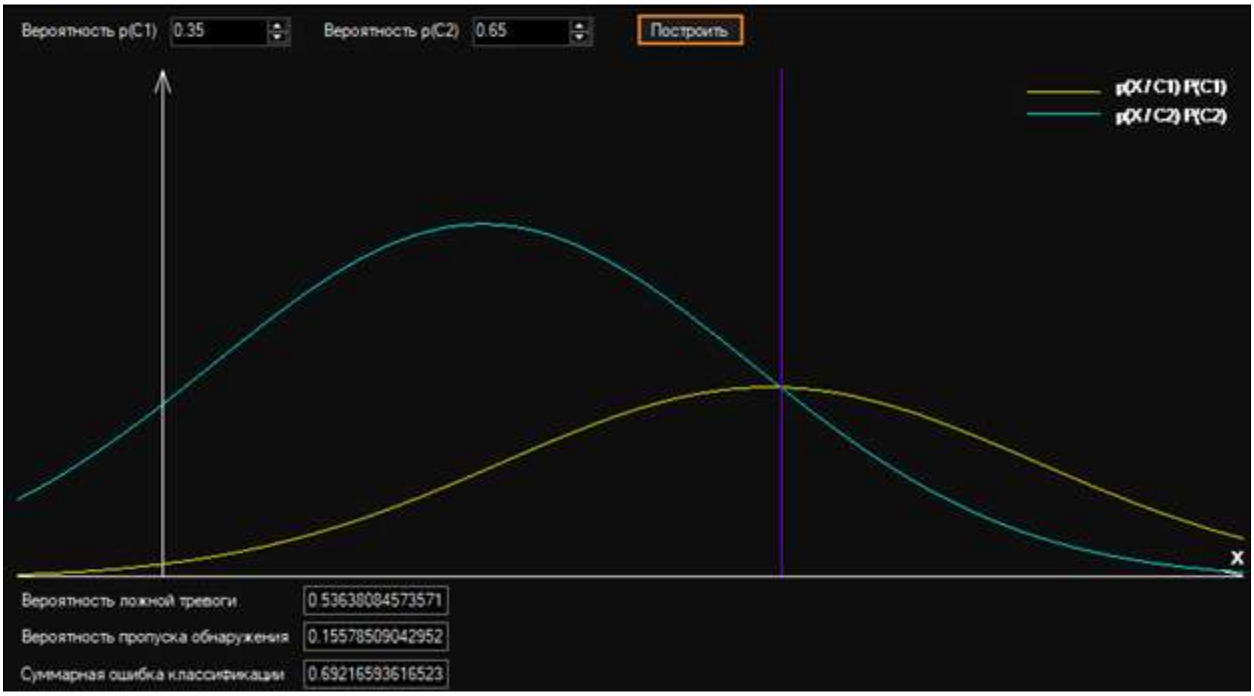

Рисунок 2 – Пример работы программы 2

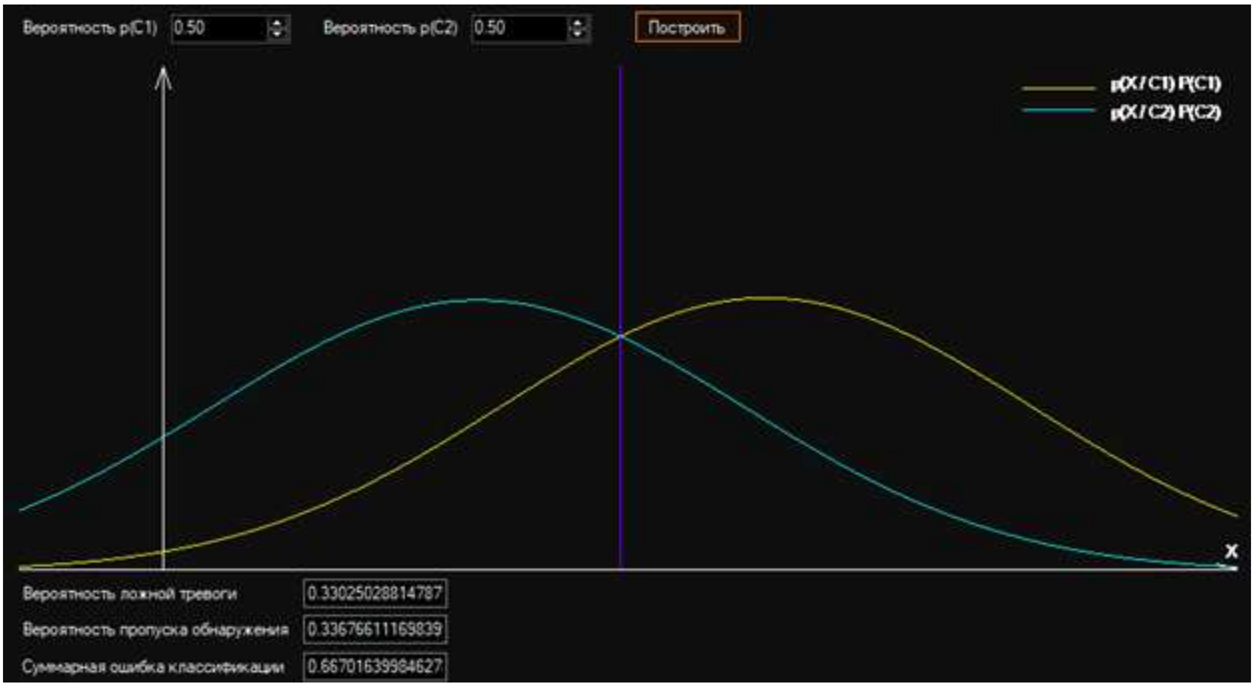

Рисунок 3 – Пример работы программы 3

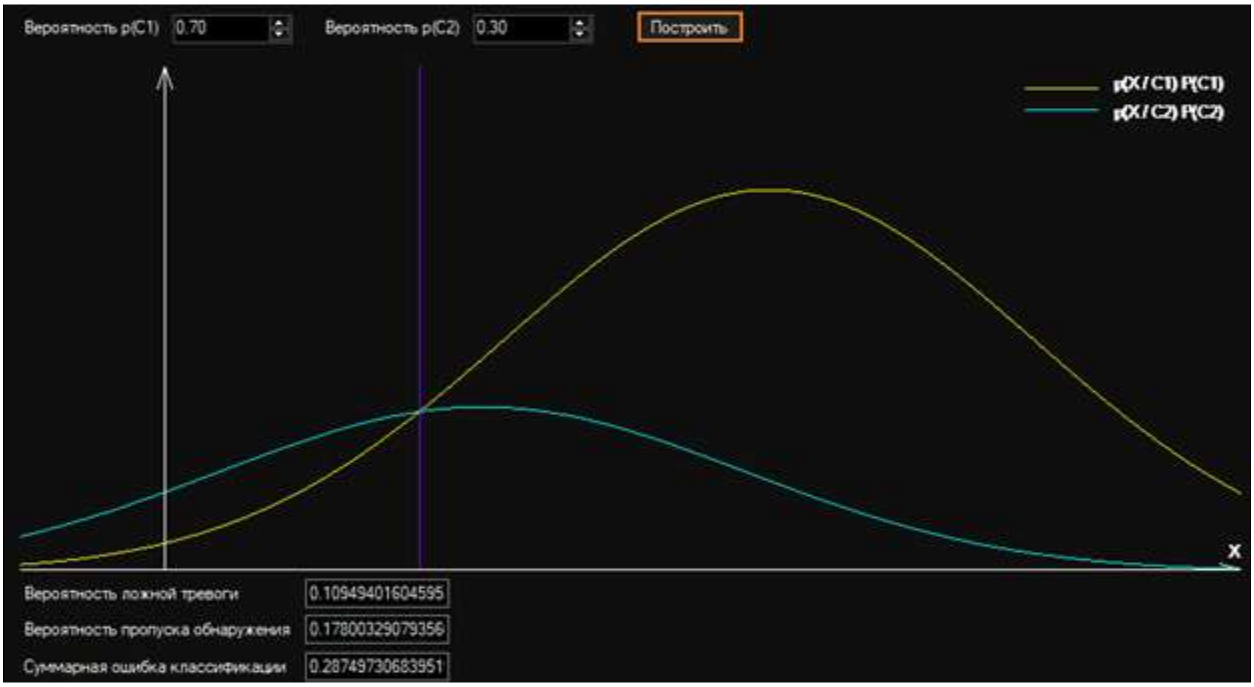

Рисунок 4 – Пример работы программы 4

Код программы:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
```

```csharp
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Lab3
{
    public partial class Form1 : Form
    {
        private const int pointsCount = 10000;
        private double pc1;
        private double pc2;
        private Random random;


        public Form1()
        {
            InitializeComponent();
        }

        private void buttonRun_Click(object sender, EventArgs e)
        {
            pc1 = (double)pc1NumericUpDown.Value;
            pc2 = (double)pc2NumericUpDown.Value;

            random = new Random();
            var bitmap = new Bitmap(pictureBox.Width, pictureBox.Height);
            using (Graphics graphics = Graphics.FromImage(bitmap) )
            {
                Do(graphics);
                pictureBox.Image = bitmap;
            }


        }

        private void Do( Graphics graphics)
        {
            var arrOfPoints1 = new int[pointsCount];
            var arrOfPoints2 = new int[pointsCount];
            double max1 = 0;
            double max2 = 0;

            for (int i = 0; i < pointsCount; i++)
            {
                arrOfPoints1[i] = random.Next(100,740);
                arrOfPoints2[i] = random.Next(-100,540);
                max1 += arrOfPoints1[i];
                max2 += arrOfPoints2[i];
```

```
            }
            max1 /= pointsCount;
            max2 /= pointsCount;

            double sigma1 = 0;
            double sigma2 = 0;
            for (int i = 0; i < pointsCount; i++)
            {
                sigma1 += Math.Pow(arrOfPoints1[i] - max1,2);
                sigma2 += Math.Pow(arrOfPoints2[i] - max2,2);
            }
            sigma1 = Math.Sqrt(sigma1/pointsCount);
            sigma2 = Math.Sqrt(sigma2/pointsCount);

            var result1 = new double[pictureBox.Width];
            var result2 = new double[pictureBox.Width];
            result1[0] = (Math.Exp(-0.5 * Math.Pow((-100 - max1) / sigma1, 2)) /
                    (sigma1 * Math.Sqrt(2 * Math.PI)) * pc1);
            result2[0] =
                    (Math.Exp(-0.5 * Math.Pow((-100 - max2) / sigma2, 2)) /
                    (sigma2 * Math.Sqrt(2 * Math.PI)) * pc2);

            int D = 0;
            for (int x = 1; x < pictureBox.Width; x++)
            {
                result1[x] =
                    (Math.Exp(-0.5*Math.Pow((x-100 - max1)/sigma1, 2))/
                    (sigma1*Math.Sqrt(2*Math.PI))*pc1);

                result2[x] =
                    (Math.Exp(-0.5*Math.Pow((x-100 - max2)/sigma2, 2))/
                    (sigma2*Math.Sqrt(2*Math.PI))*pc2);

                if (Math.Abs(result1[x]*500 - result2[x]*500) < 0.002) D = x;

                graphics.DrawLine(Pens.Blue,
                     new Point(x - 1, (pictureBox.Height - (int)(result1[x-
1]*pictureBox.Height*500))),
                    new Point(x, (pictureBox.Height - (int)(result1[x] * picture-
Box.Height * 500))));
                graphics.DrawLine(Pens.Red,
                     new Point(x - 1, (pictureBox.Height - (int)(re-
sult2[x - 1] * pictureBox.Height * 500))),
                    new Point(x, (pictureBox.Height - (int)(result2[x] * picture-
Box.Height * 500))));

            }
            double error1 = result2.Take((int)D).Sum();
            double error2;
            if (pc1 > pc2)
```

```
            {
                error2 = result2.Skip((int) D).Sum();
            }
            else
            {
                error2 = result1.Skip((int) D).Sum();
            }

            using (var textBrush = new SolidBrush(Color.Black))
            {


                graphics.DrawLine(Pens.Chartreuse, D, 0, D, pictureBox.Height);
                graphics.DrawLine(Pens.Black, 0, pictureBox.Height - 1,
                    pictureBox.Width, pictureBox.Height - 1);
                graphics.DrawLine(Pens.Black, pictureBox.Width,
                    pictureBox.Height - 1, pictureBox.Width - 15,
                    pictureBox.Height - 5);
                graphics.DrawLine(Pens.Black, 100, picture-
Box.Height - 1, 100, 0);
                graphics.DrawLine(Pens.Black, 100, 0, 95, 15);
                graphics.DrawLine(Pens.Black, 100, 0, 105, 15);
                graphics.DrawString("X", this.Font, Brushes.Black,
                    pictureBox.Width - 10, pictureBox.Height - 20);
                graphics.DrawLine(Pens.Blue, pictureBox.Width - 150, 15,
                    pictureBox.Width - 100, 15);
                graphics.DrawString("p(X / C1) P(C1)", this.Font, textBrush,
                    pictureBox.Width - 90, 5);

                graphics.DrawLine(Pens.Red, pictureBox.Width - 150, 30,
                    pictureBox.Width - 100, 30);
                graphics.DrawString("p(X / C2) P(C2)", this.Font, textBrush,
                    pictureBox.Width - 90, 25);

            }

            error1TextBox.Text = error1.ToString();
            error2TextBox.Text = error2.ToString();
            sumErrorTextBox.Text = (error1 + error2).ToString();


        }
    }
}
```