**Московский государственный технический университет им. Н.Э.Баумана**

Кафедра «Системы обработки информации и управления»

|  |  |  |  |
| --- | --- | --- | --- |
| Утверждаю: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Согласовано: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | "\_\_\_"\_\_\_\_\_\_\_\_\_\_2018 г. |  | "\_\_\_"\_\_\_\_\_\_\_\_\_\_\_2018 г. |

**Подсистема оценки качества телекоммуникационных услуг на базе алгоритма ANFIS**

Текст программы

(вид документа)

Листы А4

(вид носителя)

23

(количество листов)

Исполнитель:

Студентка группы ИУ5-84

Журавлева У.В. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

"\_\_\_"\_\_\_\_\_\_\_\_\_\_2018 г.

Москва, 2018

Аннотация

В данном документе содержится текст программы «Подсистема оценки качества телекоммуникационных услуг на базе алгоритма ANFIS». Разделы документа соответствуют модулям программы.

Содержание

[Аннотация 2](#_Toc514672231)

[Содержание 3](#_Toc514672232)

[1. Модуль NeuralNetwork 4](#_Toc514672233)

[2. Модуль RuleSet 10](#_Toc514672234)

[3. Модуль RandomNum 14](#_Toc514672235)

[4. Программа ANFIS\_install 14](#_Toc514672236)

[4.1. Модуль Form1 14](#_Toc514672237)

[4.2. Модуль Form2 15](#_Toc514672238)

[5. Программа ANFIS 16](#_Toc514672239)

[5.1. Модуль Form1 16](#_Toc514672240)

[5.2. Модуль Form2 20](#_Toc514672241)

[5.3. Модуль Form3 20](#_Toc514672242)

1. Модуль NeuralNetwork

using System;

using System.Linq;

using System.Globalization;

using System.IO;

using System.Windows.Forms;

namespace ANFIS

{

class NeuralNetwork

{

string draft;

ProgressBar pr;

int step;

const int nperem = 5; //кол-во входных переменных

double \_learningRate; //скорость обучения

int \_M; //кол-во правил

const int \_Dint = 9; //число дискретных значений в интервале для функций x / y в наборе захваченных примеров

const int \_Dint1 = 9;

RuleSet \_rules;

int \_N; //ко-во обучающих примеров

double[,] Dx1;

double[] Dz1;

private double \_learningRateSigm;

public NeuralNetwork(int brojPravila, double learningRate, string filename, ProgressBar pr1) //для обучения

{

pr = pr1;

\_M = brojPravila;

\_learningRate = learningRate;

\_learningRateSigm = learningRate;

Dx1 = new double[nperem, \_Dint1 \* \_Dint1];

Dz1 = new double[\_Dint1 \* \_Dint1];

InitialzieData(filename, ref Dx1, ref Dz1);

\_rules = new RuleSet(brojPravila);

\_rules.InitializeParams();

}

public NeuralNetwork(int brojPravila, double learningRate, string filesettings)

{

\_M = brojPravila;

\_learningRate = learningRate;

\_learningRateSigm = learningRate;

Dx1 = new double[nperem, \_Dint1 \* \_Dint1];

Dz1 = new double[\_Dint1 \* \_Dint1];

\_rules = new RuleSet(brojPravila);

\_rules.InitializeParams();

GetSettingsFromFile(filesettings);

}

private void InitialzieData(string filename, ref double[,] dx, ref double[] dz)

{

try

{

using (StreamReader sr = File.OpenText(Path.Combine(System.AppDomain.CurrentDomain.BaseDirectory.ToString(), filename)))

{

int j = 0;

do

{

string[] redak = sr.ReadLine().Split(' ');

redak = redak.Where(x => !string.IsNullOrEmpty(x)).ToArray();

dx[0, j] = double.Parse(redak[0], CultureInfo.InvariantCulture);

dx[1, j] = double.Parse(redak[1], CultureInfo.InvariantCulture);

dx[2, j] = double.Parse(redak[2], CultureInfo.InvariantCulture);

dx[3, j] = double.Parse(redak[3], CultureInfo.InvariantCulture);

dx[4, j] = double.Parse(redak[4], CultureInfo.InvariantCulture);

dz[j] = double.Parse(redak[5], CultureInfo.InvariantCulture);

j++;

} while (sr.Peek() != -1);

\_N = j;

}

}

catch (FileNotFoundException ex)

{

Console.WriteLine(ex.Message);

}

}

public void EpochTraining1(int numberOfEpochs)

{

step = 100 / (numberOfEpochs / 1000);

for (int i = 0; i < numberOfEpochs; i++)

{

TrainWholeSetStochastic1();

if (numberOfEpochs > 1000)

{

if (i % 1000 == 0 && i > 0)

{

pr.BeginInvoke(new MethodInvoker(delegate

{

pr.Value += step;

}));

//pr.Value += step;

}

}

}

}

public void TrainWholeSetStochastic1()

{

for (int i = 0; i < \_N; i++)

SingleTrainingIterationStoch1(i);

}

private void SingleTrainingIterationStoch1(int exampleIndex)

{

double o;

double brojnik = 0;

double nazivnik = 0; //знаменатель (сумма)

double brojnik2; //для изменения параметров a и b

double[] x = new double[nperem];

for (int i = 0; i < nperem; i++)

x[i] = Dx1[i, exampleIndex];

for (int i = 0; i < \_M; i++)

{

double alfa = \_rules.Alpha(i, x);

nazivnik += alfa;

brojnik += alfa \* \_rules.Konsekvens(i, x);

}

o = brojnik / nazivnik;

for (int i = 0; i < \_M; i++)

{

brojnik2 = 0;

\_rules.SetW0(i, \_rules.GetW0(i) + \_learningRate \* (Dz1[exampleIndex] - o) \* (\_rules.Alpha(i, x) \* Dx1[0, exampleIndex]) / nazivnik);

\_rules.SetW1(i, \_rules.GetW1(i) + \_learningRate \* (Dz1[exampleIndex] - o) \* (\_rules.Alpha(i, x) \* Dx1[1, exampleIndex]) / nazivnik);

\_rules.SetW2(i, \_rules.GetW2(i) + \_learningRate \* (Dz1[exampleIndex] - o) \* (\_rules.Alpha(i, x) \* Dx1[2, exampleIndex]) / nazivnik);

\_rules.SetW3(i, \_rules.GetW3(i) + \_learningRate \* (Dz1[exampleIndex] - o) \* (\_rules.Alpha(i, x) \* Dx1[3, exampleIndex]) / nazivnik);

\_rules.SetW4(i, \_rules.GetW4(i) + \_learningRate \* (Dz1[exampleIndex] - o) \* (\_rules.Alpha(i, x) \* Dx1[4, exampleIndex]) / nazivnik);

\_rules.SetW5(i, \_rules.GetW5(i) + \_learningRate \* (Dz1[exampleIndex] - o) \* (\_rules.Alpha(i, x)) / nazivnik);

for (int j = 0; j < \_M; j++)

if (i != j)

brojnik2 += \_rules.Alpha(j, x) \* (\_rules.Konsekvens(i, x) - \_rules.Konsekvens(j, x));

\_rules.SetA1(i, \_rules.GetA1(i) + \_learningRateSigm \* (Dz1[exampleIndex] - o) \* (brojnik2) / Math.Pow(nazivnik, 2)\* \_rules.GetB1(i) \* \_rules.Alpha(i, x) \* (1 - \_rules.Antecedent1(i, Dx1[0, exampleIndex])));

\_rules.SetA2(i, \_rules.GetA2(i) + \_learningRateSigm \* (Dz1[exampleIndex] - o) \* (brojnik2) / Math.Pow(nazivnik, 2)\* \_rules.GetB2(i) \* \_rules.Alpha(i, x) \* (1 - \_rules.Antecedent2(i, Dx1[1, exampleIndex])));

\_rules.SetA3(i, \_rules.GetA3(i) + \_learningRateSigm \* (Dz1[exampleIndex] - o) \* (brojnik2) / Math.Pow(nazivnik, 2)\* \_rules.GetB3(i) \* \_rules.Alpha(i, x) \* (1 - \_rules.Antecedent3(i, Dx1[2, exampleIndex])));

\_rules.SetA4(i, \_rules.GetA4(i) + \_learningRateSigm \* (Dz1[exampleIndex] - o) \* (brojnik2) / Math.Pow(nazivnik, 2)\* \_rules.GetB4(i) \* \_rules.Alpha(i, x) \* (1 - \_rules.Antecedent4(i, Dx1[3, exampleIndex])));

\_rules.SetA5(i, \_rules.GetA5(i) + \_learningRateSigm \* (Dz1[exampleIndex] - o) \* (brojnik2) / Math.Pow(nazivnik, 2)\* \_rules.GetB5(i) \* \_rules.Alpha(i, x) \* (1 - \_rules.Antecedent5(i, Dx1[4, exampleIndex])));

\_rules.SetB1(i, \_rules.GetB1(i) - \_learningRateSigm \* (Dz1[exampleIndex] - o) \* (brojnik2) / Math.Pow(nazivnik, 2)\* (Dx1[0, exampleIndex] - \_rules.GetA1(i)) \* \_rules.Alpha(i, x) \* (1 - \_rules.Antecedent1(i, Dx1[0, exampleIndex])));

\_rules.SetB2(i, \_rules.GetB2(i) - \_learningRateSigm \* (Dz1[exampleIndex] - o) \* (brojnik2) / Math.Pow(nazivnik, 2)\* (Dx1[1, exampleIndex] - \_rules.GetA2(i)) \* \_rules.Alpha(i, x) \* (1 - \_rules.Antecedent2(i, Dx1[1, exampleIndex])));

\_rules.SetB3(i, \_rules.GetB3(i) - \_learningRateSigm \* (Dz1[exampleIndex] - o) \* (brojnik2) / Math.Pow(nazivnik, 2)\* (Dx1[2, exampleIndex] - \_rules.GetA3(i)) \* \_rules.Alpha(i, x) \* (1 - \_rules.Antecedent3(i, Dx1[2, exampleIndex])));

\_rules.SetB4(i, \_rules.GetB4(i) - \_learningRateSigm \* (Dz1[exampleIndex] - o) \* (brojnik2) / Math.Pow(nazivnik, 2)\* (Dx1[3, exampleIndex] - \_rules.GetA4(i)) \* \_rules.Alpha(i, x) \* (1 - \_rules.Antecedent4(i, Dx1[3, exampleIndex])));

\_rules.SetB5(i, \_rules.GetB5(i) - \_learningRateSigm \* (Dz1[exampleIndex] - o) \* (brojnik2) / Math.Pow(nazivnik, 2)\* (Dx1[4, exampleIndex] - \_rules.GetA5(i)) \* \_rules.Alpha(i, x) \* (1 - \_rules.Antecedent5(i, Dx1[4, exampleIndex])));

}

}

public double Loss(double expectedOutput, double output)

{

double sum = 0;

for (int i = 0; i < \_M; i++)

sum += (expectedOutput - output) \* (expectedOutput - output);

return sum / 2;

}

public double Error()

{

double sum = 0;

for (int i = 0; i < \_N; i++)

sum += Loss(Dz1[i], NetworkOutput1(i));

return sum / \_N;

}

public double NetworkOutput1(int exampleIndex)

{

double brojnik = 0;

double nazivnik = 0;

double[] x = new double[nperem];

x[0] = Dx1[0, exampleIndex];

x[1] = Dx1[1, exampleIndex];

x[2] = Dx1[2, exampleIndex];

x[3] = Dx1[3, exampleIndex];

x[4] = Dx1[4, exampleIndex];

for (int i = 0; i < \_M; i++)

{

double alfa = \_rules.Alpha(i, x);

nazivnik += alfa;

brojnik += alfa \* \_rules.Konsekvens(i, x);

}

return brojnik / nazivnik;

}

public int NetworkOutput(double x0, double x1, double x2, double x3, double x4) //Исправили

{

double brojnik = 0;

double nazivnik = 0;

double a;

int answer;

double[] x = new double[nperem];

x[0] = x0;

x[1] = x1;

x[2] = x2;

x[3] = x3;

x[4] = x4;

for (int i = 0; i < \_M; i++)

{

double alfa = \_rules.Alpha(i, x);

nazivnik += alfa;

brojnik += alfa \* \_rules.Konsekvens(i, x);

}

a = Math.Round(brojnik / nazivnik);

if (a <= 0) a = 1;

else if (a >= 5) a = 5;

answer = Convert.ToInt32(a);

return answer;

}

public void WriteDataToFile(string fileName)

{

using (FileStream fs = File.Open(fileName, FileMode.Open))

using (StreamWriter sw = new StreamWriter(fs))

{

for (int i = 0; i < \_M; i++)

{

draft = Math.Round(\_rules.GetA1(i), 5).ToString(CultureInfo.InvariantCulture)+" "+ Math.Round(\_rules.GetA2(i), 5).ToString(CultureInfo.InvariantCulture)+ " " +

Math.Round(\_rules.GetA3(i), 5).ToString(CultureInfo.InvariantCulture)+ " " +

Math.Round(\_rules.GetA4(i), 5).ToString(CultureInfo.InvariantCulture)+ " " +

Math.Round(\_rules.GetA5(i), 5).ToString(CultureInfo.InvariantCulture)+ " " +

Math.Round(\_rules.GetB1(i), 5).ToString(CultureInfo.InvariantCulture)+ " " +

Math.Round(\_rules.GetB2(i), 5).ToString(CultureInfo.InvariantCulture)+ " " +

Math.Round(\_rules.GetB3(i), 5).ToString(CultureInfo.InvariantCulture)+ " " +

Math.Round(\_rules.GetB4(i), 5).ToString(CultureInfo.InvariantCulture)+ " " +

Math.Round(\_rules.GetB5(i), 5).ToString(CultureInfo.InvariantCulture);

sw.WriteLine(draft);

}

}

using (FileStream fs = File.Open("lin\_" + fileName, FileMode.Open))

using (StreamWriter sw = new StreamWriter(fs))

{

for (int i = 0; i < \_M; i++)

{

draft = Math.Round(\_rules.GetW0(i), 5).ToString(CultureInfo.InvariantCulture) + " " +

Math.Round(\_rules.GetW1(i), 5).ToString(CultureInfo.InvariantCulture) + " " +

Math.Round(\_rules.GetW2(i), 5).ToString(CultureInfo.InvariantCulture) + " " +

Math.Round(\_rules.GetW3(i), 5).ToString(CultureInfo.InvariantCulture) + " " +

Math.Round(\_rules.GetW4(i), 5).ToString(CultureInfo.InvariantCulture) + " " +

Math.Round(\_rules.GetW5(i), 5).ToString(CultureInfo.InvariantCulture);

sw.WriteLine(draft);

}

}

}

public void GetSettingsFromFile(string fileName)

{

using (StreamReader sr = File.OpenText(Path.Combine(System.AppDomain.CurrentDomain.BaseDirectory.ToString(), fileName)))

{

int j = 0;

do

{

string[] redak = sr.ReadLine().Split(' ');

redak = redak.Where(x => !string.IsNullOrEmpty(x)).ToArray();

\_rules.SetA1(j, double.Parse(redak[0], CultureInfo.InvariantCulture));

\_rules.SetA2(j, double.Parse(redak[1], CultureInfo.InvariantCulture));

\_rules.SetA3(j, double.Parse(redak[2], CultureInfo.InvariantCulture));

\_rules.SetA4(j, double.Parse(redak[3], CultureInfo.InvariantCulture));

\_rules.SetA5(j, double.Parse(redak[4], CultureInfo.InvariantCulture));

\_rules.SetB1(j, double.Parse(redak[5], CultureInfo.InvariantCulture));

\_rules.SetB2(j, double.Parse(redak[6], CultureInfo.InvariantCulture));

\_rules.SetB3(j, double.Parse(redak[7], CultureInfo.InvariantCulture));

\_rules.SetB4(j, double.Parse(redak[8], CultureInfo.InvariantCulture));

\_rules.SetB5(j, double.Parse(redak[9], CultureInfo.InvariantCulture));

j++;

} while (j!=5);

}

using (StreamReader sr = File.OpenText(Path.Combine(System.AppDomain.CurrentDomain.BaseDirectory.ToString(), "lin\_" + fileName)))

{

int j = 0;

do

{

string[] redak = sr.ReadLine().Split(' ');

redak = redak.Where(x => !string.IsNullOrEmpty(x)).ToArray();

\_rules.SetW0(j, double.Parse(redak[0], CultureInfo.InvariantCulture));

\_rules.SetW1(j, double.Parse(redak[1], CultureInfo.InvariantCulture));

\_rules.SetW2(j, double.Parse(redak[2], CultureInfo.InvariantCulture));

\_rules.SetW3(j, double.Parse(redak[3], CultureInfo.InvariantCulture));

\_rules.SetW4(j, double.Parse(redak[4], CultureInfo.InvariantCulture));

\_rules.SetW5(j, double.Parse(redak[5], CultureInfo.InvariantCulture));

j++;

} while (j!=5);

}

}

}

}

1. Модуль RuleSet

class RuleSet

{

double[] \_a1, \_b1;

double[] \_a2, \_b2;

double[] \_a3, \_b3;

double[] \_a4, \_b4;

double[] \_a5, \_b5;

double[] \_w0, \_w1, \_w2, \_w3, \_w4, \_w5;

private int \_m;

public RuleSet(int numOfRules)

{

\_m = numOfRules;

\_a1 = new double[numOfRules];

\_b1 = new double[numOfRules];

\_a2 = new double[numOfRules];

\_b2 = new double[numOfRules];

\_a3 = new double[numOfRules];

\_b3 = new double[numOfRules];

\_a4 = new double[numOfRules];

\_b4 = new double[numOfRules];

\_a5 = new double[numOfRules];

\_b5 = new double[numOfRules];

\_w0 = new double[numOfRules];

\_w1 = new double[numOfRules];

\_w2 = new double[numOfRules];

\_w3 = new double[numOfRules];

\_w4 = new double[numOfRules];

\_w5 = new double[numOfRules];

}

public int NumOfRules

{

get

{

return \_m;

}

}

public double Antecedent1(int ruleIndex, double x)

{

return Sigmoid(\_a1[ruleIndex], \_b1[ruleIndex], x);

}

public double Antecedent2(int ruleIndex, double x)

{

return Sigmoid(\_a2[ruleIndex], \_b2[ruleIndex], x);

}

public double Antecedent3(int ruleIndex, double x)

{

return Sigmoid(\_a3[ruleIndex], \_b3[ruleIndex], x);

}

public double Antecedent4(int ruleIndex, double x)

{

return Sigmoid(\_a4[ruleIndex], \_b4[ruleIndex], x);

}

public double Antecedent5(int ruleIndex, double x)

{

return Sigmoid(\_a5[ruleIndex], \_b5[ruleIndex], x);

}

private double Sigmoid(double a, double b, double var)

{

return 1 / (1 + (double)Math.Exp(b \* (var - a)));

}

public double Alpha(int ruleIndex, double[] x)

{

return Antecedent1(ruleIndex, x[0]) \* Antecedent2(ruleIndex, x[1]) \* Antecedent3(ruleIndex, x[2]) \* Antecedent4(ruleIndex, x[3]) \* Antecedent5(ruleIndex, x[4]);

}

public double Konsekvens(int ruleIndex, double[] x)

{

return \_w0[ruleIndex] \* x[0] + \_w1[ruleIndex] \* x[1] + \_w2[ruleIndex] \* x[2] + \_w3[ruleIndex] \* x[3] + \_w4[ruleIndex] \* x[4] + \_w5[ruleIndex];

}

internal void InitializeParams() //TODO CHECK trebaju li svi biti između 0 i 1?

{

Random rand = new Random();

RandomNum ran = new RandomNum();

for (int i = 0; i < \_m; i++)

{

\_a1[i] = ran.GetDouble(1, -1);

\_b1[i] = ran.GetDouble(1, -1);

\_a2[i] = ran.GetDouble(1, -1);

\_b2[i] = ran.GetDouble(1, -1);

\_a3[i] = ran.GetDouble(1, -1);

\_b3[i] = ran.GetDouble(1, -1);

\_a4[i] = ran.GetDouble(1, -1);

\_b4[i] = ran.GetDouble(1, -1);

\_a5[i] = ran.GetDouble(1, -1);

\_b5[i] = ran.GetDouble(1, -1);

\_w0[i] = ran.GetDouble(1, -1);

\_w1[i] = ran.GetDouble(1, -1);

\_w2[i] = ran.GetDouble(1, -1);

\_w3[i] = ran.GetDouble(1, -1);

\_w4[i] = ran.GetDouble(1, -1);

\_w5[i] = ran.GetDouble(1, -1);

}

}

public double GetA1(int ruleIndex)

{

return \_a1[ruleIndex];

}

public void SetA1(int ruleIndex, double value)

{

\_a1[ruleIndex] = value;

}

public double GetA2(int ruleIndex)

{

return \_a2[ruleIndex];

}

public void SetA2(int ruleIndex, double value)

{

\_a2[ruleIndex] = value;

}

public double GetA3(int ruleIndex)

{

return \_a3[ruleIndex];

}

public void SetA3(int ruleIndex, double value)

{

\_a3[ruleIndex] = value;

}

public double GetA4(int ruleIndex)

{

return \_a4[ruleIndex];

}

public void SetA4(int ruleIndex, double value)

{

\_a4[ruleIndex] = value;

}

public double GetA5(int ruleIndex)

{

return \_a5[ruleIndex];

}

public void SetA5(int ruleIndex, double value)

{

\_a5[ruleIndex] = value;

}

public double GetB1(int ruleIndex)

{

return \_b1[ruleIndex];

}

public void SetB1(int ruleIndex, double value)

{

\_b1[ruleIndex] = value;

}

public double GetB2(int ruleIndex)

{

return \_b2[ruleIndex];

}

public void SetB2(int ruleIndex, double value)

{

\_b2[ruleIndex] = value;

}

public double GetB3(int ruleIndex)

{

return \_b3[ruleIndex];

}

public void SetB3(int ruleIndex, double value)

{

\_b3[ruleIndex] = value;

}

public double GetB4(int ruleIndex)

{

return \_b4[ruleIndex];

}

public void SetB4(int ruleIndex, double value)

{

\_b4[ruleIndex] = value;

}

public double GetB5(int ruleIndex)

{

return \_b5[ruleIndex];

}

public void SetB5(int ruleIndex, double value)

{

\_b5[ruleIndex] = value;

}

public double GetW0(int ruleIndex)

{

return \_w0[ruleIndex];

}

public void SetW0(int ruleIndex, double value)

{

\_w0[ruleIndex] = value;

}

public double GetW1(int ruleIndex)

{

return \_w1[ruleIndex];

}

public void SetW1(int ruleIndex, double value)

{

\_w1[ruleIndex] = value;

}

public double GetW2(int ruleIndex)

{

return \_w2[ruleIndex];

}

public void SetW2(int ruleIndex, double value)

{

\_w2[ruleIndex] = value;

}

public double GetW3(int ruleIndex)

{

return \_w3[ruleIndex];

}

public void SetW3(int ruleIndex, double value)

{

\_w3[ruleIndex] = value;

}

public double GetW4(int ruleIndex)

{

return \_w4[ruleIndex];

}

public void SetW4(int ruleIndex, double value)

{

\_w4[ruleIndex] = value;

}

public double GetW5(int ruleIndex)

{

return \_w5[ruleIndex];

}

public void SetW5(int ruleIndex, double value)

{

\_w5[ruleIndex] = value;

}

}

1. Модуль RandomNum

class RandomNum

{

Random random = new Random();

public double GetDouble(double gornja, double donja)

{

return random.NextDouble() \* (gornja - (donja)) + (donja);

}

}

1. Программа ANFIS\_install
   1. Модуль Form1

public partial class Form1 : Form

{

string learnpath1, settingspath1;

string learnpath2, settingspath2;

string learnpath3, settingspath3;

private void buttonChange2\_Click(object sender, EventArgs e)

{

Process.Start(learnpath2);

}

private void buttonChange3\_Click(object sender, EventArgs e)

{

Process.Start(learnpath3);

}

private void buttonStart\_Click(object sender, EventArgs e)

{

Form2 f2 = new Form2();

f2.ShowDialog();

buttonStart.Enabled = false;

buttonReady.Enabled = true;

}

private void buttonReady\_Click(object sender, EventArgs e)

{

this.Close();

}

private void buttonChange1\_Click(object sender, EventArgs e)

{

Process.Start(learnpath1);

}

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

learnpath1 = "learn\_group1.txt";

settingspath2 = "ANFIS\_settings\_1.txt";

learnpath2 = "learn\_group2.txt";

settingspath2 = "ANFIS\_settings\_2.txt";

learnpath3 = "learn\_group3.txt";

settingspath3 = "ANFIS\_settings\_3.txt";

buttonReady.Enabled = false;

}

}

* 1. Модуль Form2

public partial class Form2 : Form

{

string learnpath1, settingspath1;

string learnpath2, settingspath2;

string learnpath3, settingspath3;

Thread myThread;

public Form2()

{

InitializeComponent();

Shown += new EventHandler(Form2\_Shown);

}

private void buttonReady\_Click(object sender, EventArgs e)

{

this.Close();

}

private void Form2\_Load(object sender, EventArgs e)

{

myThread = new Thread(learn);

learnpath1 = "learn\_group1.txt";

settingspath1 = "ANFIS\_settings\_1.txt";

learnpath2 = "learn\_group2.txt";

settingspath2 = "ANFIS\_settings\_2.txt";

learnpath3 = "learn\_group3.txt";

settingspath3 = "ANFIS\_settings\_3.txt";

progressBar1.Maximum = 100;

label1.Text = "Обучение для первой группы...";

buttonReady.Enabled = false;

}

private void Form2\_Shown(object sender, EventArgs e)

{

myThread.Start();

}

public void learn()

{

progressBar1.BeginInvoke(new MethodInvoker(delegate

{

progressBar1.Value =0;

}));

NeuralNetwork nn1 = new NeuralNetwork(5, 0.005, learnpath1, progressBar1);

nn1.EpochTraining1(25001);

nn1.WriteDataToFile(settingspath1);

progressBar1.BeginInvoke(new MethodInvoker(delegate

{

progressBar1.Value = 0;

}));

label1.BeginInvoke(new MethodInvoker(delegate

{

label1.Text = "Обучение для второй группы...";

}));

NeuralNetwork nn2 = new NeuralNetwork(5, 0.005, learnpath2, progressBar1);

nn2.EpochTraining1(25001);

nn2.WriteDataToFile(settingspath2);

progressBar1.BeginInvoke(new MethodInvoker(delegate

{

progressBar1.Value = 0;

}));

label1.BeginInvoke(new MethodInvoker(delegate

{

label1.Text = "Обучение для третьей группы...";

}));

NeuralNetwork nn3 = new NeuralNetwork(5, 0.005, learnpath3, progressBar1);

nn3.EpochTraining1(25001);

nn3.WriteDataToFile(settingspath3);

label1.BeginInvoke(new MethodInvoker(delegate

{

label1.Text = "Готово";

}));

buttonReady.BeginInvoke(new MethodInvoker(delegate

{

buttonReady.Enabled = true;

}));

}

}

1. Программа ANFIS
   1. Модуль Form1

public partial class Form1 : Form

{

NeuralNetwork nn1, nn2, nn3;

string connStr;

string[] id;

public int k, kolvo;

double kRg, kTh, kDy, kEr, kWd;

int group;

private void buttonChange\_Click(object sender, EventArgs e)

{

Process proc = new Process();

proc.StartInfo.FileName = "ANFIS\_install.exe";

proc.Start();

proc.WaitForExit();//ожидания выполнения

nn1 = new NeuralNetwork(5, 0.005, settingspath1);

nn2 = new NeuralNetwork(5, 0.005, settingspath2);

nn3 = new NeuralNetwork(5, 0.005, settingspath3);

}

private void buttonReport\_Click(object sender, EventArgs e)

{

Form3 f3 = new Form3(connStr);

f3.ShowDialog();

}

string settingspath1, settingspath2, settingspath3;

int mark;

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

settingspath1 = "ANFIS\_settings\_1.txt";

settingspath2 = "ANFIS\_settings\_2.txt";

settingspath3 = "ANFIS\_settings\_3.txt";

buttonWork.Enabled = false;

label5.Visible = false;

buttonReport.Enabled = false;

nn1 = new NeuralNetwork(5, 0.005, settingspath1);

nn2 = new NeuralNetwork(5, 0.005, settingspath2);

nn3 = new NeuralNetwork(5, 0.005, settingspath3);

}

private void buttonConnect\_Click(object sender, EventArgs e)

{

try

{

connStr = "server=" + textBoxHost.Text + ";user=" + textBoxLogin.Text + ";port=" + textBoxPort.Text + ";password=" + textBoxPassword.Text + ";";

using (var conn = new MySqlConnection(connStr))

{

conn.Open();

conn.Close();

}

buttonWork.Enabled = true;

buttonConnect.Enabled = false;

label5.Visible = true;

buttonReport.Enabled = true;

}

catch

{

MessageBox.Show("Проверьте правильность ввода параметров");

}

}

private void buttonWork\_Click(object sender, EventArgs e)

{

using (var conn = new MySqlConnection(connStr))

using (var cmd = conn.CreateCommand())

{

conn.Open();

cmd.CommandText = "use QualityInfo;";

cmd.ExecuteNonQuery();

cmd.CommandText = "select count(1) from collection\_file where quality\_mark is null;";

MySqlDataReader reader = cmd.ExecuteReader();

while (reader.Read())

id = new string[Convert.ToInt32(reader[0])];

reader.Close();

cmd.CommandText = "select collection\_id from collection\_file where quality\_mark is null;";

reader = cmd.ExecuteReader();

k = 0;

while (reader.Read())

{

id[k] = reader[0].ToString();

k++;

}

reader.Close();

conn.Close();

}

if (k == 0) MessageBox.Show("Не найдено файлов без оценки");

else

{

Form2 f2 = new Form2(k);

f2.ShowDialog();

}

for(int i=0;i<Data.Value;i++)

{

MessageBox.Show("data.value "+Data.Value.ToString());

mark = GiveMark(connStr, Convert.ToInt32(id[i]));

putMarkToDB(connStr, Convert.ToInt32(id[i]), mark);

MessageBox.Show("Файлу поставлена оценка "+mark.ToString());

}

}

public int GiveMark(string connStr, int id)

{

int paramid = 0, userid=0;

using (var conn = new MySqlConnection(connStr))

using (var cmd = conn.CreateCommand())

{

conn.Open();

cmd.CommandText = "use QualityInfo;";

cmd.ExecuteNonQuery();

cmd.CommandText = "select user\_id from collection\_file where collection\_id = "+id+";";

MySqlDataReader reader = cmd.ExecuteReader();

while (reader.Read())

userid = Convert.ToInt32(reader[0]);

reader.Close();

cmd.CommandText = "select user\_group from users where user\_id = "+userid+";";

reader = cmd.ExecuteReader();

while (reader.Read())

group = Convert.ToInt32(reader[0]);

reader.Close();

cmd.CommandText = "select parameter\_id from parameter where collection\_id = "+id+";";

reader = cmd.ExecuteReader();

while (reader.Read())

paramid = Convert.ToInt32(reader[0]);

reader.Close();

cmd.CommandText = "select kRg from parameter where collection\_id = "+paramid+";";

reader = cmd.ExecuteReader();

while (reader.Read())

kRg = Convert.ToDouble(reader[0]);

reader.Close();

cmd.CommandText = "select kTh from parameter where collection\_id = " + paramid + ";";

reader = cmd.ExecuteReader();

while (reader.Read())

kTh = Convert.ToDouble(reader[0]);

reader.Close();

cmd.CommandText = "select kDy from parameter where collection\_id = " + paramid + ";";

reader = cmd.ExecuteReader();

while (reader.Read())

kDy = Convert.ToDouble(reader[0]);

reader.Close();

cmd.CommandText = "select kEr from parameter where collection\_id = " + paramid + ";";

reader = cmd.ExecuteReader();

while (reader.Read())

kEr = Convert.ToDouble(reader[0]);

reader.Close();

reader.Close();

cmd.CommandText = "select kWd from parameter where collection\_id = " + paramid + ";";

reader = cmd.ExecuteReader();

while (reader.Read())

kWd = Convert.ToDouble(reader[0]);

reader.Close();

conn.Close();

}

if (group == 1) return nn1.NetworkOutput(kRg, kTh, kDy, kEr, kWd);

else if (group == 2) return nn2.NetworkOutput(kRg, kTh, kDy, kEr, kWd);

else return nn3.NetworkOutput(kRg, kTh, kDy, kEr, kWd);

}

public void putMarkToDB(string connStr, int id, int mark)

{

using (var conn = new MySqlConnection(connStr))

using (var cmd = conn.CreateCommand())

{

conn.Open();

cmd.CommandText = "use QualityInfo;";

cmd.ExecuteNonQuery();

MessageBox.Show("id " + id.ToString());

cmd.CommandText = "update collection\_file set quality\_mark = "+mark+" where collection\_id = "+id+";";

cmd.ExecuteNonQuery();

conn.Close();

}

}

}

* 1. Модуль Form2

public partial class Form2 : Form

{

int kolvo;

public Form2(int text)

{

InitializeComponent();

kolvo = text;

}

private void Form2\_Load(object sender, EventArgs e)

{

if (kolvo%10==1) label1.Text = "Найден " + kolvo.ToString() + " файл без оценки.";

else if (kolvo%10==2 || kolvo % 10 == 3 || kolvo % 10 == 4)

label1.Text = "Найдено " + kolvo.ToString() + " файла без оценки.";

else

label1.Text = "Найдено " + kolvo.ToString() + " файлов без оценки.";

}

private void button1\_Click(object sender, EventArgs e)

{

Data.Value = 1;

this.Close();

}

private void button2\_Click(object sender, EventArgs e)

{

Data.Value = kolvo;

this.Close();

}

}

* 1. Модуль Form3

public partial class Form3 : Form

{

string report\_start;

string report\_end;

string connStr;

int k;

string[] kRg, kTh, kDy, kEr, kWd, group, UID, date, mark;

string[] user\_id, coll\_id;

int count;

string path;

int columns = 9;

private Excel.Application m\_objExcel = null;

private Excel.Workbooks m\_objBooks = null;

private Excel.\_Workbook m\_objBook = null;

private Excel.Sheets m\_objSheets = null;

private Excel.\_Worksheet m\_objSheet = null;

private Excel.Range m\_objRange = null;

private Excel.Font m\_objFont = null;

private object m\_objOpt = System.Reflection.Missing.Value;

private object m\_strSampleFolder = "D:\\ExcelData\\";

Excel.Application excel = new Excel.Application();

DateTime now;

string filename;

public Form3(string conn)

{

InitializeComponent();

connStr = conn;

path = "report.xlsx";

}

private void Form3\_Load(object sender, EventArgs e)

{

dateTimePicker1.CustomFormat = "dd.MM.yyyy";

dateTimePicker1.Format = DateTimePickerFormat.Custom;

dateTimePicker2.CustomFormat = "dd.MM.yyyy";

dateTimePicker2.Format = DateTimePickerFormat.Custom;

}

private void button1\_Click(object sender, EventArgs e)

{

ReportCreate();

if (count == 0) MessageBox.Show("Не найдено данных за выбранный период времени.");

else WriteReportToFile();

}

public void CreateObj(int k)

{

kRg = new string[k];

kTh = new string[k];

kDy = new string[k];

kEr = new string[k];

kWd = new string[k];

group = new string[k];

UID = new string[k];

date = new string[k];

mark = new string[k];

coll\_id = new string[k]; ;

user\_id = new string[k];

}

public void ReportCreate()

{

report\_start = dateTimePicker1.Value.ToShortDateString() + " 00:00:00";

report\_end = dateTimePicker2.Value.ToShortDateString() + " 00:00:00";

using (var conn = new MySqlConnection(connStr))

using (var cmd = conn.CreateCommand())

{

conn.Open();

cmd.CommandText = "use QualityInfo;";

cmd.ExecuteNonQuery();

cmd.CommandText = "select count(1) from collection\_file WHERE collection\_start BETWEEN " +

"STR\_TO\_DATE('" + report\_start + "', '%d.%m.%Y %H:%i:%s') AND " +

"STR\_TO\_DATE('" + report\_end + "', '%d.%m.%Y %H:%i:%s');";

MySqlDataReader reader = cmd.ExecuteReader();

while (reader.Read())

count = Convert.ToInt32(reader[0].ToString());

reader.Close();

CreateObj(count);

cmd.CommandText = "select collection\_id, collection\_end, quality\_mark, user\_id from collection\_file WHERE collection\_start BETWEEN " +

"STR\_TO\_DATE('" + report\_start + "', '%d.%m.%Y %H:%i:%s') AND " +

"STR\_TO\_DATE('" + report\_end + "', '%d.%m.%Y %H:%i:%s');";

reader = cmd.ExecuteReader();

k = 0;

while (reader.Read())

{

coll\_id[k] = reader[0].ToString();

date[k] = reader[1].ToString();

mark[k] = reader[2].ToString();

user\_id[k] = reader[3].ToString();

k++;

}

reader.Close();

for (int i = 0; i < count; i++)

{

cmd.CommandText = "select kRg, kTh, kDy, kEr, kWd from parameter WHERE collection\_id = '" + coll\_id[i] + "';";

reader = cmd.ExecuteReader();

while (reader.Read())

{

kRg[i] = reader[0].ToString();

kTh[i] = reader[1].ToString();

kDy[i] = reader[2].ToString();

kEr[i] = reader[3].ToString();

kWd[i] = reader[4].ToString();

}

reader.Close();

cmd.CommandText = "select user\_uid, user\_group from users WHERE user\_id = '" + user\_id[i] + "';";

reader = cmd.ExecuteReader();

while (reader.Read())

{

UID[i] = reader[0].ToString();

group[i] = reader[1].ToString();

}

reader.Close();

}

conn.Close();

}

}

public void WriteReportToFile()

{

// Start a new workbook in Excel.

m\_objExcel = new Excel.Application();

m\_objBooks = (Excel.Workbooks)m\_objExcel.Workbooks;

m\_objBook = (Excel.\_Workbook)(m\_objBooks.Add(m\_objOpt));

m\_objSheets = (Excel.Sheets)m\_objBook.Worksheets;

m\_objSheet = (Excel.\_Worksheet)(m\_objSheets.get\_Item(1));

// Create an array for the headers and add it to cells A1:C1.

object[] objHeaders = { "UID", "Группа", "Дата", "Уровень использования", "Скорость", "Задержка", "Ошибки", "Временное окно", "Оценка" };

m\_objRange = m\_objSheet.get\_Range("A1", "I1");

m\_objRange.Value = objHeaders;

m\_objFont = m\_objRange.Font;

m\_objFont.Bold = true;

// Create an array and add it to the worksheet starting at cell A2.

object[,] objData = new Object[count, columns];

for (int r = 0; r < count; r++)

{

objData[r, 0] = UID[r];

objData[r, 1] = group[r];

objData[r, 2] = date[r];

objData[r, 3] = kRg[r];

objData[r, 4] = kTh[r];

objData[r, 5] = kDy[r];

objData[r, 6] = kEr[r];

objData[r, 7] = kWd[r];

objData[r, 8] = mark[r];

}

m\_objRange = m\_objSheet.get\_Range("A2", m\_objOpt);

m\_objRange = m\_objRange.get\_Resize(count, columns);

m\_objRange.Value = objData;

m\_objExcel.DisplayAlerts = false;

now = DateTime.Now;

filename = m\_strSampleFolder + "report\_" + now.ToString("dd/MM/yyyy\_hh-mm-ss") + ".xlsx";

File.Create(filename).Close();

m\_objBook.SaveAs(filename, m\_objOpt, m\_objOpt,

m\_objOpt, m\_objOpt, m\_objOpt, Excel.XlSaveAsAccessMode.xlNoChange,

m\_objOpt, m\_objOpt, m\_objOpt, m\_objOpt);

m\_objExcel.DisplayAlerts = true;

m\_objBook.Close(false, m\_objOpt, m\_objOpt);

m\_objExcel.Quit();

Process.Start(filename);

}

}