**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ**

**федеральное государственное автономное образовательное учреждение высшего образования**

**«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»**

**(РУТ(МИИТ)**

**Институт управления и цифровых технологий**

Кафедра «Вычислительные системы, сети и информационная безопасность»

**Отчет По ПРОЕКТНОй ДЕЯТЕЛЬНОСТИ**

**НА тему:**

**ПроеКТИРОВАНИЕ систем искусственного интеллекта. Разработка системы распознавания образов**

*Направление:* 10.03.01*Информационная безопасность*

*Профиль:**Безопасность компьютерных систем*

Выполнили:  
студенты группы УИБ-112

Глущенко Д.Е

Орлов А.В

Чернышев К.Р

Проверил:

Доцент Голдовский Я.М.

(должность, ФИО)

**МОСКВА 2022**

**АННОТАЦИЯ**

Глущенко Д.Е Орлов А.В Чернышев К.Р Разработка Систем Искусственного Интеллекта. Часть 1. Разработка системы распознавания образов. - Москва: ИУЦТ, 2022, УИБ-112, 47 с., 19 ил., 8 табл., прил~~.~~, 1л.

Объектом исследования является комплексный анализ обучающей выборки.

Целью работы на основе проведенного анализа является разработка алгоритма классификации каждого из 45 объектов.

В работе выявлен способ нахождения двух наиболее информативных признаков (из 7 данных), построено правило распознавания (классификации) признаков в первом узле, отображены (графически) 45 заданных в обучающей выборке объектов в пространстве двух выбранных признаков, в пространстве двух выбранных признаков построено правило распознавания (классификации) объектов во втором узле, отображены (графически) 30 объектов классов, распознаваемых во втором узле, в пространстве двух выбранных признаков, разработана программа на языке C++, распознающая объекты классов А, В и С.

Результаты работы имеют практическую значимость и могут применяться для определения классов объектов

**СОДЕРЖАНИЕ**

[**ВВЕДЕНИЕ** 2](#_Toc105132145)

[**2 ОБЩЕЕ ЗАДАНИЕ** 6](#_Toc105132146)

[**3 РАСЧЕТ ИНФОРМАТИВНЫХ ПРИЗНАКОВ** 7](#_Toc105132147)

[**4 ПОСТРОЕНИЕ КЛАССИФИКАЦИОННЫХ ФУНКЦИЙ** 14](#_Toc105132148)

[**5 РАЗРАБОТКА ИНТЕРАКТИВНОЙ ПРОГРАММЫ НА ЯЗЫКЕ C++** 18](#_Toc105132149)

[**6 ГРАФИЧЕСКИЙ ИНТЕРФЕЙС** 20](#_Toc105132150)

[**7 ВЫВОДЫ** 22](#_Toc105132151)

[**8 СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ** 23](#_Toc105132152)

[**ПРИЛОЖЕНИЯ** 24](#_Toc105132153)

[**ПРИЛОЖЕНИЕ А** 24](#_Toc105132154)

[**ПРИЛОЖЕНИЕ B** 32](#_Toc105132155)

[**ПРИЛОЖЕНИЕ C** 41](#_Toc105132156)

# **ВВЕДЕНИЕ**

Распознавание образов (объектов, сигналов, явлений или процессов) – едва ли не самая распространённая задача, которую человеку приходиться решать практически ежесекундно от первого до последнего дня своего существования. Для решения этой задачи человек использует огромные ресурсы своего мозга, включая одновременно около 7–8 млрд. нейронов. Именно это дает возможность людям мгновенно узнавать друг друга, с большой скоростью читать печатные и рукописные текст, безошибочно водить автомобиль в сложном потоке уличного движения, осуществлять отбраковку деталей на конвейере, дешифровать космические фотоснимки, разглядывать коды, древнюю египетскую клинопись и т. д.

Распознавание представляет собой задачу преобразования входной информации, в качестве которой уместно рассматривать некоторые параметры, признаки распознаваемых образов, в выходную представляющую собой заключение о том, к какому классу относится распознаваемый образ. Поэтому, учитывая, что кибернетика есть наука об общих законах преобразованиях информации в сложных системах, распознавание образов есть один из разделов этой науки.

Потребности в комплексной механизации и автоматизации производства, создания роботов, в широких массах решать задачи технической и медицинской диагностики, метеорологического прогноза, формализованной оценки общественных, экономических и социологических явлений и процессов, определили наиболее вероятных направлений их трансформации и предопределили значительные усилия научной и инженерной мысли, направленные на решение теоретических и прикладных вопросов проблемы распознавания.

Ниже будут обсуждены основные задачи, возникающие при построении систем распознавания, их формальная постановка и методы решения. Однако прежде рассмотрим содержательную интерпретацию задачи распознавания. Выполним это на примере задачи распознавания стороной А самолетов стороны В. Для. построения системы распознавания самолетов стороны В стороне А необходимо провести детальный анализ всей доступной информации об авиации стороны В и, исходя из анализа тактико-технических характеристик своих средств противодействия самолетам стороны В, оценить, какие решения она может принимать в случае налета самолетов стороны В. Такими решениями могут быть, например: 1) применить средство противодействия S1; 2) применить средство противодействия S2; 3) совместно использовать средства S1; и S2, В соответствии с этими решениями самолеты стороны В следует подразделить на три класса, При этом, если появляются самолеты первого класса, следует применить средство S1, если самолеты второго класса - средство S2 и, наконец, если самолеты третьего класса - совместно средства S: и S1 Наличие конкретных технических средств обнаружения самолетов 1 определения их параметров, а также недостаточный объем исходной (априорной) информации о классах самолетов стороны В (положим, первый класс — бомбардировщики, второй класс — истребители, третий класс —штурмовики) может привести к тому, что с точки зрения эффективности стороне А целесообразно ввести в рассмотрение только два класса, так как при этом повышается вероятность правильного распознавания самолетов.

Этот вопрос можно решить только путем математического или физико-математического моделирования разрабатываемой системы распознавания. После проведения классификации самолетов стороны В следует определить, с помощью каких параметров или признаков можно описать выделенные классы самолетов, а затем из полученного перечня исключить те признаки, относительно которых не представляется возможным определить их значения применительно к каждому классу самолетов. Далее, в соответствии с техническими возможностями средств наблюдения за самолетами (радиолокаторы, акустические средства, лазеры, оптические устройства и т. д.) из полученного перечня признаков надо выделить признаки, которые могут быть реально определены (например, крейсерская и максимальная скорости, предельная высота полета, число и тип двигателей, длина фюзеляжа, размах крыльев и др.). И, наконец, на основе априорных данных следует описать на языке выбранных признаков каждый класс самолетов. В данном случае одни признаки имеют качественный характер (тип двигателей), другие — количественный (скорость, высота полета ит. д.). Поэтому в описании, классов должны содержаться сведения как о том, присущи или не присущи каждому классу те или иные признаки качественного характера, так и о возможных диапазонах или законах распределений значений признаков, имеющих количественные выражения, для каждого класса.

На этом подготовительную работу можно считать завершенной, поскольку накоплена и проанализирована априорная информация о самолетах, произведена их классификация, выбрана система признаков и описаны все классы самолетов на’ языке этих признаков. Положим, что с помощью каких-либо средств наблюдения в результате проведения опытов найдены некоторые признаки неизвестного, подлежащего распознаванию самолета. Сопоставление полученных апостериорных данных 0б этом самолете с данными, заключенными в априорном описании всех классов самолетов на языке признаков, позволяет определить, к какому классу относится неизвестный самолет, т. с. позволяет произвести его распознавание, Рассмотренный пример содержит качественное описание задачи распознавания и не поясняет, как производить разбиение объектов на классы, как накапливать и обрабатывать априорную информацию, из каких соображений осуществлять выбор признаков и как описать на их языке классы, на основе каких методов сопоставлять апостериорную и априорную информацию, т. е. распознавать неизвестный объект.

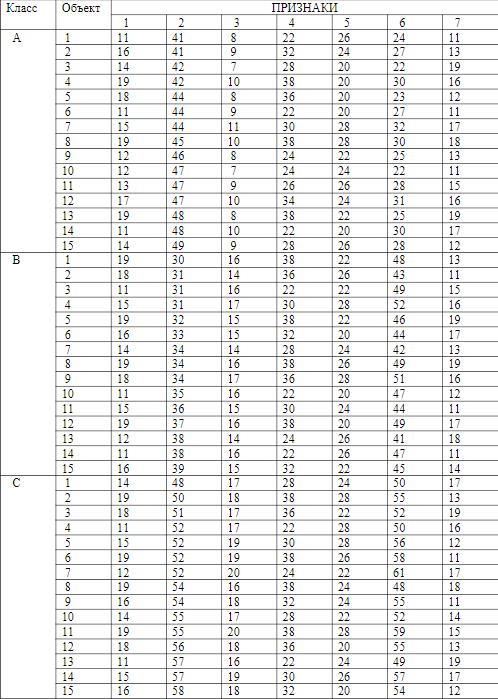
Однако пример дает возможность составить представление о задаче распознавания в следующей трактовке. Имеется некоторая совокупность объектов или явлений. В соответствии с выбранным принципом классификации она подразделена на ряд классов, т. е. составлен алфавит. классов. Разработан словарь признаков, на языке которого описывается каждый класс объектов. Созданы технические средства, обеспечивающие определение признаков, а на вычислительных средствах системы распознавания реализован алгоритм распознавания, позволяющий сопоставлять апостериорные данные о неизвестном объекте с априорной информацией и на основе сопоставления определять, к какому классу он может быть отнесен. Когда появляется объект, подлежащий распознаванию, с помощью технических средств наблюдения проводятся опыты н определяются его признаки. Данные о признаках неизвестного объекта поступают на вход алгоритма распознавания, который, используя априорные описания классов, определяет, к какому классу может быть отнесен этот объект.

# **2 ОБЩЕЕ ЗАДАНИЕ**

Необходимо спроектировать и программно реализовать систему, распознающую объектов классов A, B, C по их описанию (7 признаков)

Обучающая выборка.

Таблица 1 – Обучающая выборка



# **3 РАСЧЕТ ИНФОРМАТИВНЫХ ПРИЗНАКОВ**

Для расчёта информативности признаков мы выбрали метрику Фишера. Её достоинства заключаются в лёгкости использования, и результативности. Изначально мы считаем среднее арифметическое каждого признака для классов, затем – среднее квадратичное отклонение каждого признака для классов. После этого, по формуле (1) вычисляем значение информативности признака между двумя определёнными классами.

*,* (1)

Где - среднее значение признака *X* у объектов класса А;

- среднее значение признака *X* у объектов класса В;

- среднее квадратическое значение признака *X* объектов класса А;

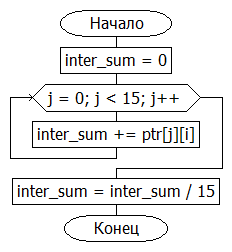
- среднее квадратическое значение признака *X* объектов класса B;

Таким образом, определим самые информативные признаки между классами А и В, В и С, А и С. Реализуем процесс вычисления информативностей с помощью языка С++: Составим функцию, которая высчитывает среднее арифметическое значений (рисунок 1; таблица 2), затем функцию, которая высчитывает среднее квадратичное отклонений значений (рисунок 3; таблица 3), затем составим функцию для определения значения информативностей.

Таблица 2 – таблица имен

|  |  |  |
| --- | --- | --- |
| fun\_Inter\_sum | | |
| Исходные параметры и переменные | | |
| ptr | целочисленный | Указатель на матрицу, в которой хранятся объекты класса |
| i | целочисленный | Номер строки в матрице |
| Рабочие переменные и массивы | | |
| j | целочисленный | Номер итерации (столбца) |
| Результат | | |
| inter\_sum | вещественный | Значение среднего арифметического для всех признаков объекта класса |

Далее составим блок-схему алгоритма (рисунок 1) для функции fun\_inter\_sum:



fun\_Inter\_sum(ptr, i)

возврат

Рисунок 1 - Блок-схема

На основании блок-схемы запишем напишем код (рисунок 2)

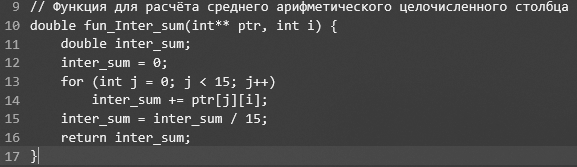


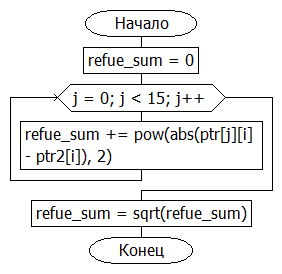
Рисунок 2 – код функции fun\_inter\_sum

Расчет средних значений:  
Таблица 3 – таблицы средних арифметических значений

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 14.7333 | 45 | 8.86667 | 29.4667 | 23,3333 | 26,9333 | 94,6 | A |
| 15.5333 | 34.2 | 15.4667 | 31.0667 | 23,7333 | 46.4667 | 108.467 | B |
| 15.7333 | 53.5333 | 17.9333 | 31.4667 | 24.4 | 54.0667 | 95.0667 | С |

Таблица 4 – таблица имен

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| fun\_refue\_sum(ptr, ptr2, i) | | | | |
| Рабочие переменные и массивы | | | | |
| j | целочисленный | | | Номер итерации |
| ptr | вещественный | | | Признаки первого класса |
| ptr2 | вещественный | | | Признаки второго класса |
| Результат | | | | |
| refue\_sum | | вещественный | Сумма среднего квадратичного откл. | |



fun\_refue\_sum(ptr, ptr2, i)

Возврат refue\_sum

Рисунок 3 - Блок-схема

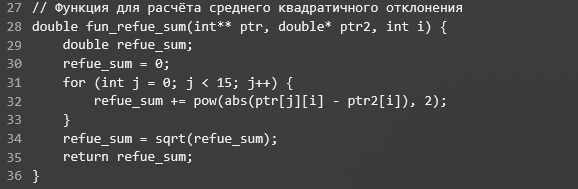


Рисунок 4 – код функции fun\_refue\_sum

Таблица средних квадратических отклонений:  
Таблица 5 – расчёт средних квадратичных отклонений

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 11.5297 | 10 | 4.44222 | 23.0593 | 11.1952 | 12.3666 | 11.563 | A  B  С |
| 11.7360 | 10.8812 | 3.70585 | 23.472 | 10.5325 | 12.2366 | 13.5657 |
| 10.957 | 10.6646 | 4.7888 | 21.8113 | 10.6583 | 14.3852 | 11.5825 |

Расчёт информативностей признаков:

Таблица 6– информативность функции

|  |  |  |
| --- | --- | --- |
| informativnosti | | |
| i | целочисленный | Переменная счётчик |
| Рабочие переменные | | |
| inf\_b\_and\_a[i] | вещественный | информат. между а и б |
| inf\_b\_and\_c[i] | вещественный | информат. между б и с |
| inf\_a\_and\_c[i] | вещественный | информат. между а и с |
| indab | целочисленный | наиб. инф. признаки а и б |
| indbc | целочисленный | наиб. инф. признаки б и с |
| indac | целочисленный | наиб. инф. признаки а и с |

informativnosti

Изображение выглядит как текст

Автоматически созданное описание

1

Рисунок 5 - Блок-схема

Изображение выглядит как текст

Автоматически созданное описание

1

возврат

Рисунок 6 - Блок-схема

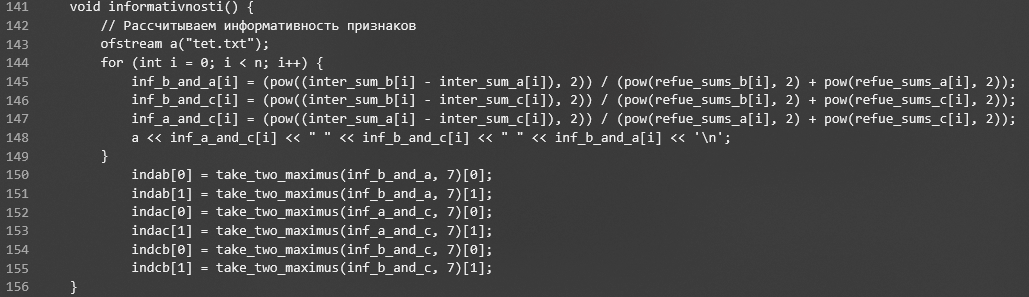


Рисунок 7 – Код функции informativnosti

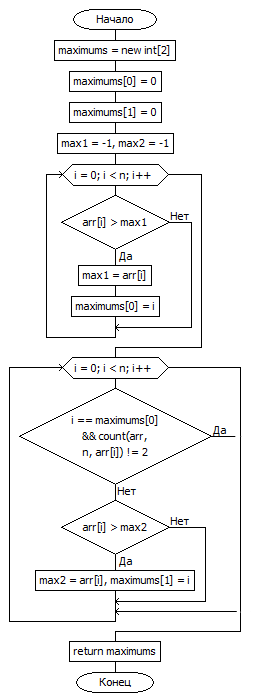
Таким образом, мы получаем таблицу информативностей из семи столбцов, и трёх строк:

Таблица 7- таблица информативностей

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0.00397035 | 0.340695 | 1.92667 | 0.00397 | 0.0047619 | 2.04581 | 0.008135 | AC |
| 0.00015584 | 1.61019 | 0.16593 | 0.00015 | 0.0019791 | 0.161944 | 0.000564 | CB |
| 0.00236453 | 0.534066 | 1.30159 | 0.00236 | 0.0006772 | 1.26063 | 0.000605 | AB |

Для удобства определения двух самых информативных признаков опишем функцию, которая сама забирает два максимальных значения из переданного ей массива вещественных чисел.   
Таблица 8 – таблица имен

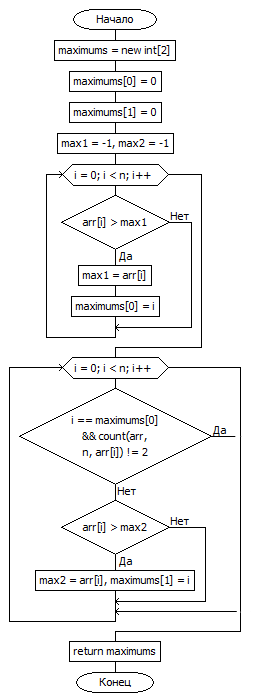
|  |  |  |
| --- | --- | --- |
| take\_two\_maximus(arr, n) | | |
| Исходные данные | | |
| arr | вещественный | Массив для поиска |
| Рабочие переменные | | |
| maximums[0] | вещественный | Первый индекс наиб. знач. |
| maximums[1] | вещественный | Второй признак наиб. знач. |
| max1 | вещественный | Временная переменная для хранения первого найденного максимального значения |
| max2 | вещественный | Временная переменная для хранения второго найденного максимального значения |
| i | Переменная-счётчик в циклах | информат. между б и с |
| Результат | | |
| maximums | Указатель вещ. типа | Массив с индексами двух максимальных элементов |



take\_two\_maximus(arr, n)

1

Рисунок 8 - Блок-схема



возврат

1

Рисунок 8 – Продолжение блок-схемы

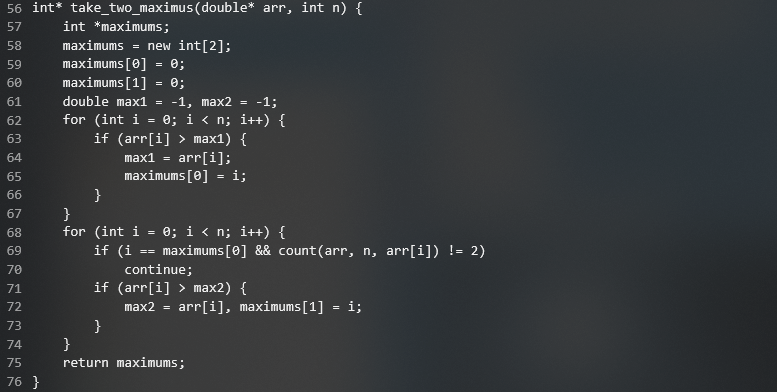
****

Рисунок 9 – Код функции take\_two\_maximus

# **4 ПОСТРОЕНИЕ КЛАССИФИКАЦИОННЫХ ФУНКЦИЙ**

Для построения классификационных функций будем использовать формулу линейной дискриминантной функции, что позволит оценить причины результативности кода, а также наглядно покажет разделение классов в пространстве двух выбранных признаков.

Где - центр класса А;

- центр класса B;

Построим график для каждого пространства двух выбранных признаков с помощью Excel, где поместим объекты классов, а также прямую, их разделяющую. Затем реализуем использование формулы линейной дискриминантной функции в коде программы.

Реализуем подсчёт коэффициентов функции, составим для каждого случая своё уравнение, через которое зададим функцию x1(x2), где x1 – первый информативный признак, х2 – второй информативных признак.

Рассчитываем ЛДФ для А и В

Где - центр класса А;

- центр класса B;

Самые информативные признаки между ними – 3 и 6  
F = (15,46667 – 8,867; 46,467 – 26,934)x(x1;x2)T –  
-0,5(15,46667 – 8,867; 46,467 – 26,934)х(15,46667 + 8,867; 46,467 + 26,934)T  
Делаем из этого уравнение:  
(6, 6; 19,53)(X1;X2)T – ½(6,6;19,53)(24,32;71,401)T = 0  
6,6x1 + 19,53x2 – 1/2(128,898 + 1394,462) = 0  
6,6x1 + 19,53x2 – 761.679 = 0  
x2 = (761.679 – 6,6\*x2)/19,53 – Строим график исходя из данных

Рассчитываем ЛДФ для С и В

Где - центр класса C;

- центр класса B;

Самые информативные признаки между ними – 2 и 3

F = (53.533 – 34.2; 17.933 – 15.466)(x1;x2)T –

½((53.533–34.2; 17.933–15.466)x(53.533 + 34.2; 17.933 + 15.466))

F = (19,33; 2,46)(x1;x2)T – ½(19,33;2,46)(87,73; 33,40)

Делаем из этого уравнение

(19,33; 2,46)(x1;x2)T – ½(19,33;2,46)(87,73; 33,40) = 0

19,33x1 + 2,46x2 – ½(1690,02 + 82,164) = 0

19,33x1 + 2,46x2 – 886,092 = 0

x2 = (886,092 – 19,31x1)/2,46 – строим график исходя из данных

Рассчитываем ЛДФ для А и С

Где - центр класса А;

- центр класса C;

Самые информативные признаки между ними – 3 и 6

F = ( 17,933–8,867; 54,066–26,934)x(x1;x2)T –

-0,5( 17,933–8,867; 54,066–26,934)х(17,933+ 8,867; 54,066 + 26,934)T  
Делаем из этого уравнение:

(9,066; 27,133)(X1;X2)T – ½(9,066; 27,133)(27;82)T = 0

9,066x1 + 27,133x2 – ½(244,782 + 2224,906) = 0

9,066x1 + 27,133x2 – 1234.844 = 0

x2 = (1234.844 – 9.066x1)/27.133 – строим график исходя из данных

Графики представлены на рисунках 10, 11, 12 для каждой из пар классов:

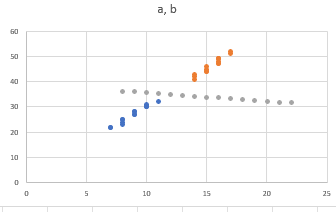


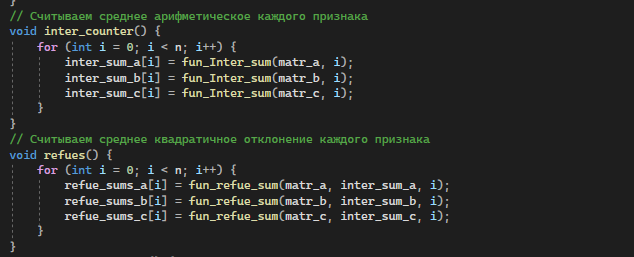
Рисунок 10 – классификационная прямая между а и б

Рисунок 11 - классификационная прямая

Рисунок 12 - классификационная прямая

# **5 РАЗРАБОТКА ИНТЕРАКТИВНОЙ ПРОГРАММЫ НА ЯЗЫКЕ C++**

Таким образом, наша система распознавания образов работает следующим образом:

  
Рисунок 13 – Код функций расчёта величин для формулы

Изначально мы считаем среднее арифметическое значение каждого признака класса, после этого – среднее квадратичное отклонение каждого признака класса, по формуле 1 рассчитываем информативности для каждого из признаков объектов.

Выбираем с каждой комбинации по два максимальных значения, и в пространстве каждой пары информативных признаков строим классификационную функцию.

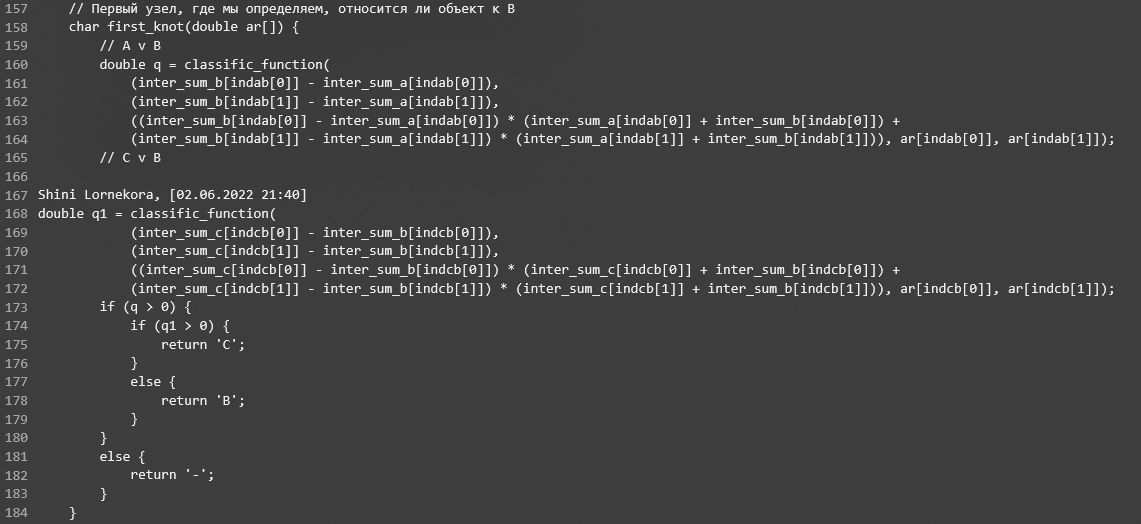


Рисунок 14 – Код первого узла алгоритма

Классификационная функция первого узла представляет собой сравнение объекта Б с другими классами. Иными словами, мы отделяем класс В от А, затем отделяем класс В от С. Если оказывается, что изучаемый объект не относится к классу В, тогда мы переходим ко второму узлу.

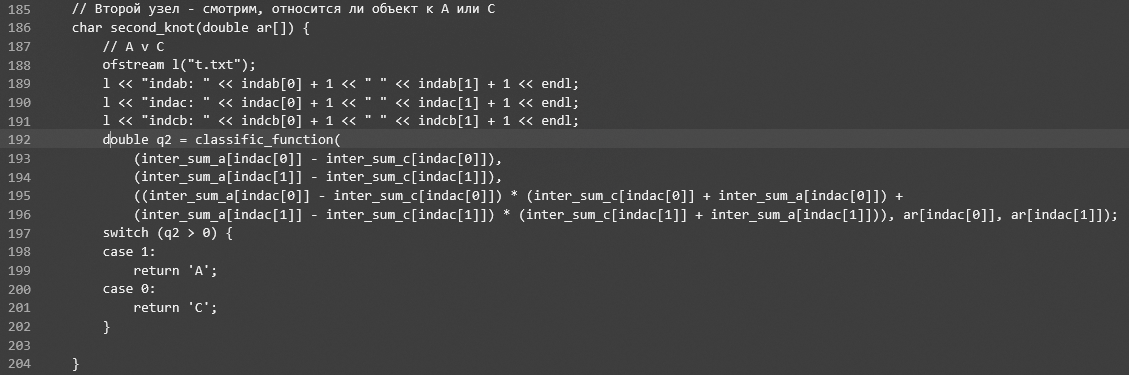


Рисунок 15 – Код первого узла алгоритма

Классификационная функция второго узла заключает в себе отделение класса А от класса С. На основе результатов узлов программа выдаёт результат пользователю.

# **6 ГРАФИЧЕСКИЙ ИНТЕРФЕЙС**

Чтобы облегчить пользователю работу с программой, мы реализовали графический интерфейс с помощью Windows Forms. Расстановка виджетов производилась через панель компонентов.

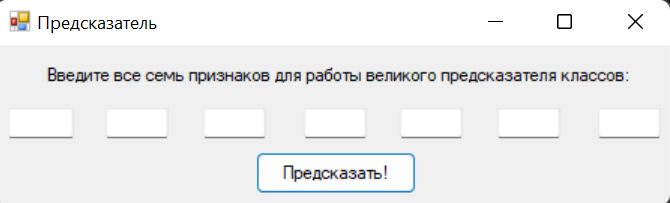


Рисунок 16 –Общий вид окна программы

Основная вычислительная процедура связана с графическим интерфейсом через обработку события нажатии на виджет – кнопку. После того, как пользователь нажал на неё, вся информация в текстовых окошках перейдёт в программу в типе данных System^ string, поэтому мы конвертируем данные в string через библиотеку marchal\_cppstd, после пользуемся функцией stoi(), записываем все числа в целочисленный массив, и отправляем в функцию предсказания.

Функция предсказания – программа, состоящая из двух узлов, на основе выдаваемого значения которой программа принимает решение, к какому классу относится объект. При всех введённых значениях обучающей\проверяющей выборки программа работает корректно. Ниже, на рисунках 17, 18, 19 показан результат работы при введении разных признаков.

Введем признаки объекта A, результат работы алгоритма можно увидеть на рисунке 17

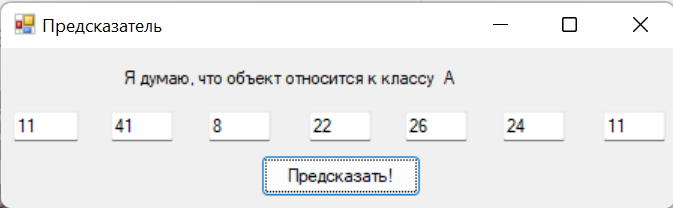


Рисунок 17 – Вид программы при отображении класса А

Введем признаки объекта B, результат работы алгоритма можно увидеть на рисунке 18

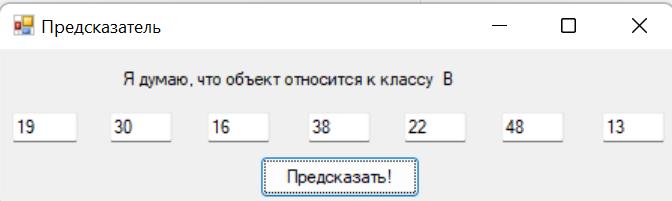


Рисунок 18 – Вид программы при отображении класса В

Введем признаки объекта С, результат работы алгоритма можно увидеть на рисунке 19

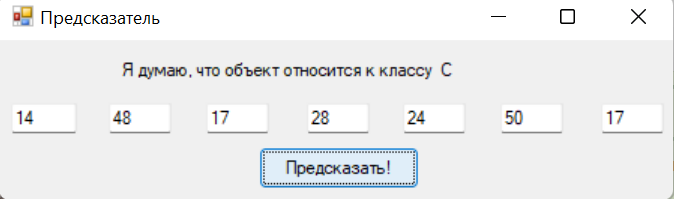


Рисунок 19 – Вид программы при отображении класса С

# **7 ВЫВОДЫ**

В ходе работы мы изучили способы классификации признаков, разработали алгоритм для определения объектов, создали графическую оболочку для алгоритма, научились работать с метрикой Фишера, а также анализировать небольшие объёмы баз данных на примере заданной нам изначально обучающей выборки.

# **8 СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

1 Горелкин А.Л., Скрипкин В. А. Методы распознавания (обзор)

# **ПРИЛОЖЕНИЯ**

# **ПРИЛОЖЕНИЕ А**

// Здесь идёт реализация вычислений и предсказание типа объекта класса  
#include <iostream>

#include <fstream>

#include <string>

#include <sstream>

#include <cmath>

#include <iomanip>

using namespace std;

// Функция для расчёта среднего арифметического целочисленного столбца

double fun\_Inter\_sum(int\*\* ptr, int i) {

double inter\_sum;

inter\_sum = 0;

for (int j = 0; j < 15; j++)

inter\_sum += ptr[j][i];

inter\_sum = inter\_sum / 15;

return inter\_sum;

}

// Функция для расчёта среднего арифметического вещественного столбца

double fun\_Inter\_sum\_d(double\*\* ptr, int i) {

double inter\_sum;

inter\_sum = 0;

for (int j = 0; j < 15; j++)

inter\_sum += ptr[j][i];

inter\_sum = inter\_sum / 15;

return inter\_sum;

}

// Функция для расчёта среднего квадратичного отклонения

double fun\_refue\_sum(int\*\* ptr, double\* ptr2, int i) {

double refue\_sum;

refue\_sum = 0;

for (int j = 0; j < 15; j++) {

refue\_sum += pow(abs(ptr[j][i] - ptr2[i]), 2);

}

refue\_sum = sqrt(refue\_sum);

return refue\_sum;

}

// Функция для расчёта квадратичного отклонения у вещественных чисел

double fun\_refue\_sum\_d(double\*\* ptr, double\* ptr2, int i) {

double refue\_sum;

refue\_sum = 0;

for (int j = 0; j < 15; j++) {

refue\_sum += pow(abs(ptr[j][i] - ptr2[i]), 2);

}

refue\_sum = sqrt(refue\_sum/15);

return refue\_sum;

}

int count(double\* arr, int n, double r) {

int cou = 0;

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

if (arr[i] == r)

cou++;

return cou;

}

int\* take\_two\_maximus(double\* arr, int n) {

int \*maximums;

maximums = new int[2];

maximums[0] = 0;

maximums[1] = 0;

double max1 = -1, max2 = -1;

for (int i = 0; i < n; i++) {

if (arr[i] > max1) {

max1 = arr[i];

maximums[0] = i;

}

}

for (int i = 0; i < n; i++) {

if (i == maximums[0] && count(arr, n, arr[i]) != 2)

continue;

if (arr[i] > max2) {

max2 = arr[i], maximums[1] = i;

}

}

return maximums;

}

// Классификационная функция

double classific\_function(double a, double b, double c, double x1, double x2) {

return a \* x1 + b \* x2 - c \* 0.5;

}

class Neuron\_Network {

private:

int m = 15, n = 7;

double

inter\_sum\_a[7], inter\_sum\_b[7], inter\_sum\_c[7],

refue\_sums\_a[7], refue\_sums\_b[7], refue\_sums\_c[7],

inf\_a\_and\_c[7], inf\_b\_and\_a[7], inf\_b\_and\_c[7];

int\* indab = new int[2];

int\* indcb = new int[2];

int\* indac = new int[2];

// Выделяем память под обучающую выборку

int\*\* matr\_a = new int\* [m];

int\*\* matr\_b = new int\* [m];

int\*\* matr\_c = new int\* [m];

double\*\* matr\_ac = new double\* [m];

public:

// Считываем обучающую выборку из файлов

Neuron\_Network() {

ifstream a("thingA.txt");

ifstream b("thingB.txt");

ifstream c("thingC.txt");

string as = "11 41 8 22 26 24 11 16 41 9 32 24 27 13 14 42 7 28 20 22 19 19 42 10 38 20 30 16 18 44 8 36 20 23 12 11 44 9 22 20 27 11 15 44 11 30 28 32 17 19 45 10 38 28 30 18 12 46 8 24 22 25 13 12 47 7 24 24 22 11 13 47 9 26 26 28 15 17 47 10 34 24 31 16 19 48 8 38 22 25 19 11 48 10 22 20 30 17 14 49 9 28 26 28 12";

string bs = "19 30 16 38 22 48 13 18 31 14 36 26 43 11 11 31 16 22 22 49 15 15 31 17 30 28 52 16 19 32 15 38 22 46 19 16 33 15 32 20 44 17 14 34 14 28 24 42 13 19 34 16 38 26 49 19 18 34 17 36 28 51 16 11 35 16 22 20 47 12 15 36 15 30 24 44 11 19 37 16 38 20 49 17 12 38 14 24 26 41 18 11 38 16 22 26 47 11 16 39 15 32 22 45 14";

string cs = "14 48 17 28 24 50 17 19 50 18 38 28 55 13 18 51 17 36 22 52 19 11 52 17 22 28 50 16 15 52 19 30 28 56 12 19 52 19 38 26 58 11 12 52 20 24 22 61 17 19 54 16 38 24 48 18 16 54 18 32 24 55 11 14 55 17 28 22 52 14 19 55 20 38 28 59 15 18 56 18 36 20 55 13 11 57 16 22 24 49 19 15 57 19 30 26 57 17 16 58 18 32 20 54 12";

stringstream helper1, helper2, helper3;

for (int i = 0; i < m; i++) {

matr\_a[i] = new int[n];

matr\_b[i] = new int[n];

matr\_c[i] = new int[n];

}

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

for (int k = 0; k < n; k++) {

helper1 << as;

helper2 << bs;

helper3 << cs;

helper1 >> matr\_a[i][k];

helper2 >> matr\_b[i][k];

helper3 >> matr\_c[i][k];

}

}

// Считываем среднее арифметическое каждого признака

void inter\_counter() {

//ofstream t("text.txt");

for (int i = 0; i < n; i++) {

inter\_sum\_a[i] = fun\_Inter\_sum(matr\_a, i);

inter\_sum\_b[i] = fun\_Inter\_sum(matr\_b, i);

inter\_sum\_c[i] = fun\_Inter\_sum(matr\_c, i);

// t << inter\_sum\_a[i] << " " << inter\_sum\_b[i] << " " << inter\_sum\_c[i] << endl;

}

}

// Считываем среднее квадратичное отклонение каждого признака

void refues() {

ofstream t("u.txt");

for (int i = 0; i < n; i++) {

refue\_sums\_a[i] = fun\_refue\_sum(matr\_a, inter\_sum\_a, i);

refue\_sums\_b[i] = fun\_refue\_sum(matr\_b, inter\_sum\_b, i);

refue\_sums\_c[i] = fun\_refue\_sum(matr\_c, inter\_sum\_c, i);

t << refue\_sums\_a[i] << " " << refue\_sums\_b[i] << " " << refue\_sums\_c[i] << endl;

}

}

void informativnosti() {

ofstream a("tet.txt");

// Рассчитываем информативность признаков

for (int i = 0; i < n; i++) {

inf\_b\_and\_a[i] = (pow((inter\_sum\_b[i] - inter\_sum\_a[i]), 2)) / (pow(refue\_sums\_b[i], 2) + pow(refue\_sums\_a[i], 2));

inf\_b\_and\_c[i] = (pow((inter\_sum\_b[i] - inter\_sum\_c[i]), 2)) / (pow(refue\_sums\_b[i], 2) + pow(refue\_sums\_c[i], 2));

inf\_a\_and\_c[i] = (pow((inter\_sum\_a[i] - inter\_sum\_c[i]), 2)) / (pow(refue\_sums\_a[i], 2) + pow(refue\_sums\_c[i], 2));

a << inf\_a\_and\_c[i] << " " << inf\_b\_and\_c[i] << " " << inf\_b\_and\_a[i] << '\n';

}

indab[0] = take\_two\_maximus(inf\_b\_and\_a, 7)[0];

indab[1] = take\_two\_maximus(inf\_b\_and\_a, 7)[1];

indac[0] = take\_two\_maximus(inf\_a\_and\_c, 7)[0];

indac[1] = take\_two\_maximus(inf\_a\_and\_c, 7)[1];

indcb[0] = take\_two\_maximus(inf\_b\_and\_c, 7)[0];

indcb[1] = take\_two\_maximus(inf\_b\_and\_c, 7)[1];

}

// Первый узел, где мы определяем, относится ли объект к B

char first\_knot(double ar[]) {

// A v B

double q = classific\_function(

(inter\_sum\_b[indab[0]] - inter\_sum\_a[indab[0]]),

(inter\_sum\_b[indab[1]] - inter\_sum\_a[indab[1]]),

((inter\_sum\_b[indab[0]] - inter\_sum\_a[indab[0]]) \* (inter\_sum\_a[indab[0]] + inter\_sum\_b[indab[0]]) +

(inter\_sum\_b[indab[1]] - inter\_sum\_a[indab[1]]) \* (inter\_sum\_a[indab[1]] + inter\_sum\_b[indab[1]])), ar[indab[0]], ar[indab[1]]);

// C v B

double q1 = classific\_function(

(inter\_sum\_c[indcb[0]] - inter\_sum\_b[indcb[0]]),

(inter\_sum\_c[indcb[1]] - inter\_sum\_b[indcb[1]]),

((inter\_sum\_c[indcb[0]] - inter\_sum\_b[indcb[0]]) \* (inter\_sum\_c[indcb[0]] + inter\_sum\_b[indcb[0]]) +

(inter\_sum\_c[indcb[1]] - inter\_sum\_b[indcb[1]]) \* (inter\_sum\_c[indcb[1]] + inter\_sum\_b[indcb[1]])), ar[indcb[0]], ar[indcb[1]]);

if (q > 0) {

if (q1 > 0) {

return 'C';

}

else {

return 'B';

}

}

else {

return '-';

}

}

// Второй узел - смотрим, относится ли объект к А или С

char second\_knot(double ar[]) {

// A v C

ofstream l("t.txt");

l << "indab: " << indab[0] + 1 << " " << indab[1] + 1 << endl;

l << "indac: " << indac[0] + 1 << " " << indac[1] + 1 << endl;

l << "indcb: " << indcb[0] + 1 << " " << indcb[1] + 1 << endl;

double q2 = classific\_function(

(inter\_sum\_a[indac[0]] - inter\_sum\_c[indac[0]]),

(inter\_sum\_a[indac[1]] - inter\_sum\_c[indac[1]]),

((inter\_sum\_a[indac[0]] - inter\_sum\_c[indac[0]]) \* (inter\_sum\_c[indac[0]] + inter\_sum\_a[indac[0]]) +

(inter\_sum\_a[indac[1]] - inter\_sum\_c[indac[1]]) \* (inter\_sum\_c[indac[1]] + inter\_sum\_a[indac[1]])), ar[indac[0]], ar[indac[1]]);

switch (q2 > 0) {

case 1:

return 'A';

case 0:

return 'C';

}

}

// Предсказание нейросети на основе математики

char prediction(double f1, double f2, double f3, double f4, double f5, double f6, double f7) {

double all[] = { f1, f2, f3, f4, f5, f6, f7 };

char result = first\_knot(all);

if (!(result == '-'))

return result;

else

return second\_knot(all);

}

};

# **ПРИЛОЖЕНИЕ B**

// Код для графического интерфейса, реализованный через Windows Forms  
#pragma once

#include <Windows.h>

#include "NeuronNetwork.h"

#include <msclr\marshal\_cppstd.h>

namespace Project8 {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Сводка для MyForm

/// </summary>

public ref class MyForm : public System::Windows::Forms::Form

{

public:

MyForm(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~MyForm()

{

if (components)

{

delete components;

}

}

public: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::ColorDialog^ colorDialog1;

private: System::Windows::Forms::ContextMenuStrip^ contextMenuStrip1;

private: System::Windows::Forms::ToolStripTextBox^ toolStripTextBox1;

public: System::Windows::Forms::TextBox^ textBox1;

public: System::Windows::Forms::TextBox^ textBox2;

private: System::Windows::Forms::Label^ label1;

public: System::Windows::Forms::TextBox^ textBox3;

private:

public: System::Windows::Forms::TextBox^ textBox4;

public: System::Windows::Forms::TextBox^ textBox5;

public: System::Windows::Forms::TextBox^ textBox6;

public: System::Windows::Forms::TextBox^ textBox7;

private: System::ComponentModel::IContainer^ components;

protected:

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

this->components = (gcnew System::ComponentModel::Container());

this->button1 = (gcnew System::Windows::Forms::Button());

this->colorDialog1 = (gcnew System::Windows::Forms::ColorDialog());

this->contextMenuStrip1 = (gcnew System::Windows::Forms::ContextMenuStrip(this->components));

this->toolStripTextBox1 = (gcnew System::Windows::Forms::ToolStripTextBox());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->textBox2 = (gcnew System::Windows::Forms::TextBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->textBox3 = (gcnew System::Windows::Forms::TextBox());

this->textBox4 = (gcnew System::Windows::Forms::TextBox());

this->textBox5 = (gcnew System::Windows::Forms::TextBox());

this->textBox6 = (gcnew System::Windows::Forms::TextBox());

this->textBox7 = (gcnew System::Windows::Forms::TextBox());

this->contextMenuStrip1->SuspendLayout();

this->SuspendLayout();

//

// button1

//

this->button1->Location = System::Drawing::Point(260, 109);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(161, 43);

this->button1->TabIndex = 0;

this->button1->Text = L"Предсказать!";

this->button1->UseVisualStyleBackColor = true;

this->button1->Click += gcnew System::EventHandler(this, &MyForm::button1\_Click);

//

// contextMenuStrip1

//

this->contextMenuStrip1->ImageScalingSize = System::Drawing::Size(24, 24);

this->contextMenuStrip1->Items->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(1) { this->toolStripTextBox1 });

this->contextMenuStrip1->Name = L"contextMenuStrip1";

this->contextMenuStrip1->Size = System::Drawing::Size(161, 39);

//

// toolStripTextBox1

//

this->toolStripTextBox1->Font = (gcnew System::Drawing::Font(L"Segoe UI", 9));

this->toolStripTextBox1->Name = L"toolStripTextBox1";

this->toolStripTextBox1->Size = System::Drawing::Size(100, 31);

//

// textBox1

//

this->textBox1->Location = System::Drawing::Point(12, 65);

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(64, 26);

this->textBox1->TabIndex = 2;

//

// textBox2

//

this->textBox2->Location = System::Drawing::Point(110, 65);

this->textBox2->Name = L"textBox2";

this->textBox2->Size = System::Drawing::Size(61, 26);

this->textBox2->TabIndex = 3;

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(46, 20);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(588, 20);

this->label1->TabIndex = 4;

this->label1->Text = L"Введите все семь признаков для работы великого предсказателя классов:";

this->label1->Click += gcnew System::EventHandler(this, &MyForm::label1\_Click);

//

// textBox3

//

this->textBox3->Location = System::Drawing::Point(207, 65);

this->textBox3->Name = L"textBox3";

this->textBox3->Size = System::Drawing::Size(61, 26);

this->textBox3->TabIndex = 5;

//

// textBox4

//

this->textBox4->Location = System::Drawing::Point(307, 65);

this->textBox4->Name = L"textBox4";

this->textBox4->Size = System::Drawing::Size(61, 26);

this->textBox4->TabIndex = 6;

//

// textBox5

//

this->textBox5->Location = System::Drawing::Point(403, 65);

this->textBox5->Name = L"textBox5";

this->textBox5->Size = System::Drawing::Size(61, 26);

this->textBox5->TabIndex = 7;

//

// textBox6

//

this->textBox6->Location = System::Drawing::Point(501, 65);

this->textBox6->Name = L"textBox6";

this->textBox6->Size = System::Drawing::Size(61, 26);

this->textBox6->TabIndex = 8;

//

// textBox7

//

this->textBox7->Location = System::Drawing::Point(601, 65);

this->textBox7->Name = L"textBox7";

this->textBox7->Size = System::Drawing::Size(61, 26);

this->textBox7->TabIndex = 9;

//

// MyForm

//

this->AutoScaleDimensions = System::Drawing::SizeF(9, 20);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(674, 164);

this->Controls->Add(this->textBox7);

this->Controls->Add(this->textBox6);

this->Controls->Add(this->textBox5);

this->Controls->Add(this->textBox4);

this->Controls->Add(this->textBox3);

this->Controls->Add(this->label1);

this->Controls->Add(this->textBox2);

this->Controls->Add(this->textBox1);

this->Controls->Add(this->button1);

this->Name = L"MyForm";

this->Text = L"Предсказатель";

this->contextMenuStrip1->ResumeLayout(false);

this->contextMenuStrip1->PerformLayout();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e) {

Neuron\_Network nn;

System::String^ f\_12 = textBox1->Text;

System::String^ f\_22 = textBox2->Text;

System::String^ f\_32 = textBox3->Text;

System::String^ f\_42 = textBox4->Text;

System::String^ f\_52 = textBox5->Text;

System::String^ f\_62 = textBox6->Text;

System::String^ f\_72 = textBox7->Text;

msclr::interop::marshal\_context context;

std::string standartString1 = context.marshal\_as<std::string>(f\_12);

std::string standartString2 = context.marshal\_as<std::string>(f\_22);

std::string standartString3 = context.marshal\_as<std::string>(f\_32);

std::string standartString4 = context.marshal\_as<std::string>(f\_42);

std::string standartString5 = context.marshal\_as<std::string>(f\_52);

std::string standartString6 = context.marshal\_as<std::string>(f\_62);

std::string standartString7 = context.marshal\_as<std::string>(f\_72);

int f\_1, f\_2, f\_3, f\_4, f\_5, f\_6, f\_7;

f\_1 = stoi(standartString1);

f\_2 = stoi(standartString2);

f\_3 = stoi(standartString3);

f\_4 = stoi(standartString4);

f\_5 = stoi(standartString5);

f\_6 = stoi(standartString6);

f\_7 = stoi(standartString7);

nn.inter\_counter();

nn.refues();

nn.informativnosti();

char result = nn.prediction(f\_1, f\_2, f\_3, f\_4, f\_5, f\_6, f\_7);

switch (result) {

case 67:

label1->Text = L" Я думаю, что объект относится к классу C ";

break;

case 66:

label1->Text = L" Я думаю, что объект относится к классу B ";

break;

case 65:

label1->Text = L" Я думаю, что объект относится к классу A ";

break;

}

}

private: System::Void label1\_Click(System::Object^ sender, System::EventArgs^ e) {

}

};

}

# **ПРИЛОЖЕНИЕ C**

// Код для реализации функции – точки входа в программу

#include "MyForm.h"

using namespace Project8;

int WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int) {

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

Project8::MyForm form;

Application::Run(% form);

return 0;

}