Федеральное государственное бюджетное образовательное учреждение высшего образования   
«Саратовский государственный технический университет имени Ю. А. Гагарина»

Кафедра прикладных информационных технологий

**Практическая работа по курсу**

**Компьютерные методы моделирования искусственных нейронных сетей**

**«Разработка нейроэмулятора сети и карты Кохонена для решения задачи кластеризации»**

Выполнил студент б1-ИФСТ-31,

Кошелева Алина Денисовна

Проверил преподаватель КММИНС,

Кузьмин Алексей Константинович

Саратов, 2024

**Цель работы**

Цель данной практической работы заключается в разработке нейроэмулятора сети и карты Кохонена, который способен эффективно решать задачи кластеризации. В ходе работы необходимо изучить алгоритмы и методы обучения для сети и карты Кохонена, выбрать оптимальную архитектуру для поставленной задачи кластеризации, создать программное обеспечение для обучения нейроэмулятора на обучающей выборке и провести эксперименты для оценки качества его работы. Основной задачей является получение модели нейроэмулятора, способной точно кластеризовать входные данные.

**Описание предметной области**

Для данной практической работы была выбрана задачи кластеризации выбора хорошего вина.

Предметная область кластеризации по выбору вина включает в себя сбор сведений об основных характеристиках необходимых для качественных вин, таких как:

1. Баланс вкусовых характеристик;
2. Качество винограда;
3. Хранение;
4. Процент алкоголя;
5. Рейтинг региона выращивания;
6. Сахар;

Кластеризация позволяет определить качество вина, а также влияние методов хранения и выращивания на его вкусовые характеристики.

**Модель**

На основе описания предметной области определим входные факторы на рисунке 1.



Рисунок 1 - Модель

Составим обучающее множество опираясь на входные параметры:

78 низкое бутылка крепленое низкий да

34 отличное бочка крепленое высокий нет

11 отличное бочка некрепленое высокий нет

23 низкое бутылка некрепленое низкий да

89 низкое бочка крепленое высокий да

90 отличное бочка некрепленое высокий нет

66 отличное бочка некрепленое высокий да

56 отличное бочка крепленое высокий нет

11 низкое бутылка некрепленое низкий да

13 низкое бутылка крепленое низкий да

33 низкое бутылка крепленое низкий нет

44 низкое бутылка некрепленое низкий нет

77 отличное бочка некрепленое высокий нет

76 отличное бочка крепленое высокий да

23 низкое бутылка некрепленое низкий нет

55 низкое бутылка крепленое низкий да

87 отличное бочка некрепленое высокий нет

67 отличное бочка некрепленое высокий да

22 низкое бутылка крепленое низкий нет

88 отличное бочка крепленое высокий нет

23 отличное бочка некрепленое высокий да

34 низкое бутылка крепленое низкий нет

32 низкое бутылка некрепленое высокий да

92 отличное бочка некрепленое высокий нет

53 отличное бочка крепленое высокий да

73 низкое бочка некрепленое низкий нет

31 низкое бутылка некрепленое низкий нет

80 отличное бочка крепленое высокий нет

87 отличное бочка некрепленое высокий да

15 низкое бутылка крепленое низкий да

24 низкое бутылка крепленое низкий нет

66 отличное бочка некрепленое высокий нет

88 отличное бочка некрепленое высокий да

77 отличное бочка крепленое высокий да

22 низкое бутылка крепленое низкий нет

11 низкое бутылка крепленое низкий нет

26 низкое бутылка некрепленое низкий да

69 отличное бочка некрепленое высокий да

88 отличное бочка крепленое высокий да

33 низкое бутылка крепленое низкий нет

27 низкое бутылка некрепленое низкий да

42 низкое бутылка некрепленое низкий да

78 отличное бочка крепленое высокий да

81 отличное бочка некрепленое высокий нет

76 отличное бочка крепленое высокий нет

36 низкое бутылка крепленое низкий да

31 низкое бутылка крепленое низкий нет

87 отличное бочка некрепленое высокий да

89 отличное бочка некрепленое высокий нет

29 низкое бутылка крепленое низкий нет

32 низкое бутылка крепленое низкий да

31 низкое бутылка крепленое низкий нет

89 отличное бочка некрепленое высокий нет

69 отличное бочка крепленое высокий нет

88 отличное бочка некрепленое высокий да

24 низкое бутылка крепленое низкий нет

43 низкое бутылка некрепленое низкий нет

38 низкое бутылка некрепленое низкий да

10 низкое бутылка крепленое низкий нет

57 отличное бочка крепленое высокий да

53 отличное бочка некрепленое высокий нет

85 отличное бочка крепленое высокий да

74 отличное бочка крепленое высокий нет

2 низкое бутылка некрепленое низкий нет

6 низкое бутылка некрепленое низкий да

8 низкое бутылка крепленое низкий нет

9 низкое бутылка крепленое низкий да

79 отличное бочка крепленое высокий нет

61 отличное бочка некрепленое высокий да

89 отличное бочка некрепленое высокий да

58 отличное бочка крепленое высокий да

31 низкое бутылка некрепленое низкий да

1 низкое бутылка крепленое низкий нет

33 низкое бутылка некрепленое низкий да

18 низкое бутылка крепленое низкий нет

89 отличное бочка некрепленое высокий нет

90 отличное бочка крепленое высокий да

56 отличное бочка некрепленое высокий нет

78 отличное бочка крепленое высокий нет

14 низкое бутылка крепленое низкий да

19 низкое бутылка крепленое низкий нет

29 низкое бутылка некрепленое низкий да

31 низкое бутылка крепленое низкий нет

38 низкое бутылка некрепленое низкий нет

45 низкое бутылка некрепленое низкий да

72 отличное бочка крепленое высокий да

90 отличное бочка некрепленое высокий нет

99 отличное бочка некрепленое высокий нет

59 отличное бочка крепленое высокий да

22 низкое бутылка некрепленое низкий нет

11 низкое бутылка крепленое низкий нет

33 низкое бутылка некрепленое низкий да

44 низкое бутылка крепленое низкий нет

77 отличное бочка крепленое высокий да

88 отличное бочка некрепленое высокий да

73 отличное бочка крепленое высокий нет

74 отличное бочка крепленое высокий нет

41 низкое бутылка некрепленое низкий нет

63 низкое бутылка крепленое низкий нет

41 низкое бутылка некрепленое низкий да

13 низкое бутылка крепленое низкий да

67 отличное бочка крепленое низкий нет

78 отличное бочка крепленое низкий нет

89 отличное бочка некрепленое высокий нет

17 низкое бутылка крепленое низкий да

27 низкое бутылка крепленое низкий да

37 низкое бутылка крепленое низкий нет

58 низкое бутылка некрепленое низкий да

89 отличное бочка некрепленое высокий нет

77 отличное бочка крепленое высокий нет

59 отличное бочка крепленое высокий да

33 низкое бутылка некрепленое низкий да

12 низкое бутылка некрепленое высокий нет

78 отличное бочка крепленое высокий нет

58 отличное бочка крепленое высокий да

88 отличное бочка крепленое высокий нет

13 низкое бутылка некрепленое низкий да

20 низкое бутылка крепленое низкий да

23 отличное бочка крепленое высокий нет

34 низкое бутылка некрепленое низкий да

78 отличное бочка крепленое высокий да

12 отличное бочка некрепленое высокий нет

45 низкое бутылка крепленое низкий да

Это обучающее множество поместим в файл dataset3.txt.

**Программная реализация**

Основные компоненты нейроэмулятора представлены классами Neuron, KohonenNet, Cell, KohonenMap.

Первый фрагмент разрабатываемого нейроэмулятора – нейрон. Класс Neuron (рисунок 2):

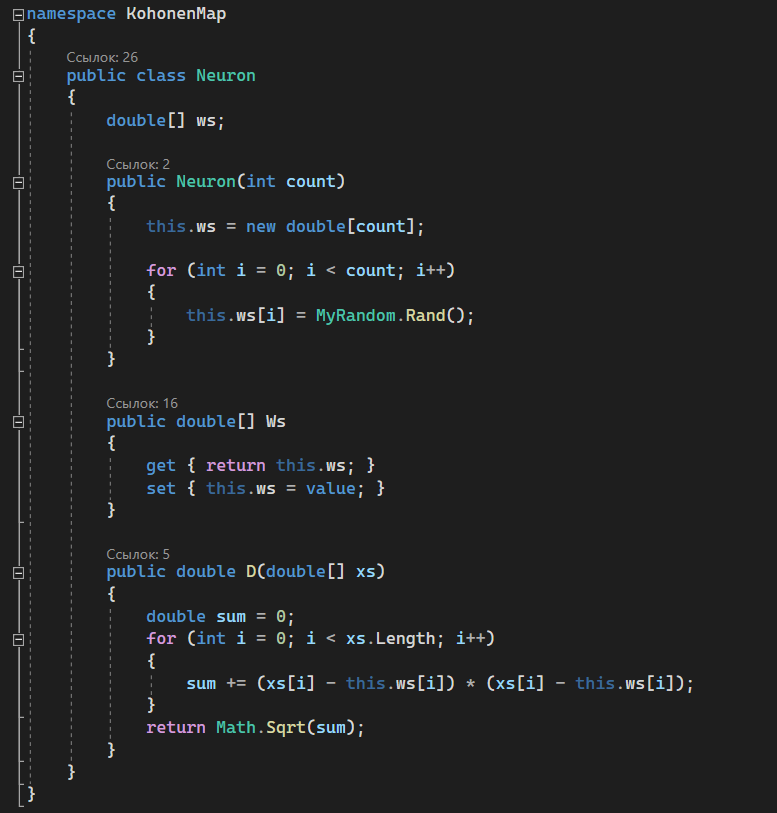


Рисунок 2 – Класс Neuron

Он содержит в себе список double[] ws коэффециентов  и метод D(double[] xs), который позволяет вычислить Евклидово расстояние между вектором ws и вектором xs.

Создание нейрона состоит из заполнения коэффициентов  случайными значениями при помощи класса MyRandom, обладающего статическим методом, возвращающим случайное значение от 0 до 1 (рисунок 3):

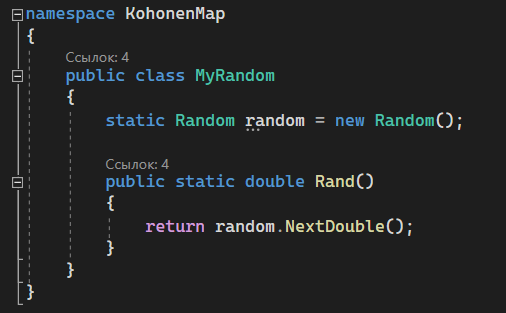
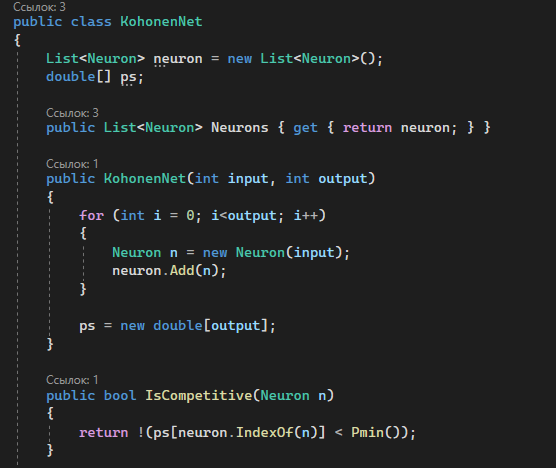
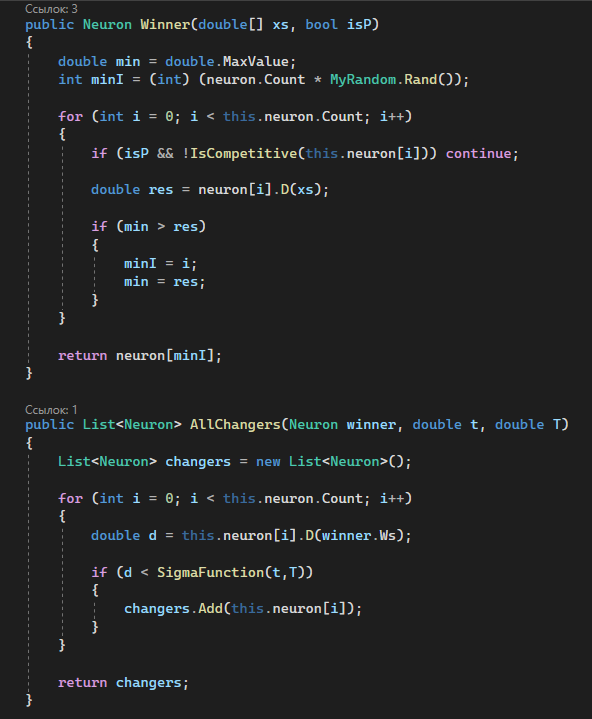
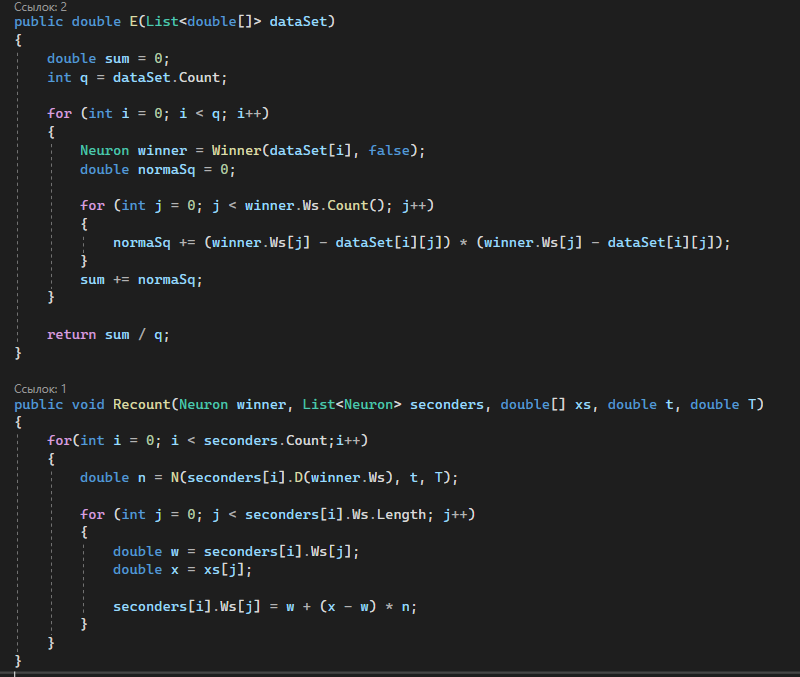


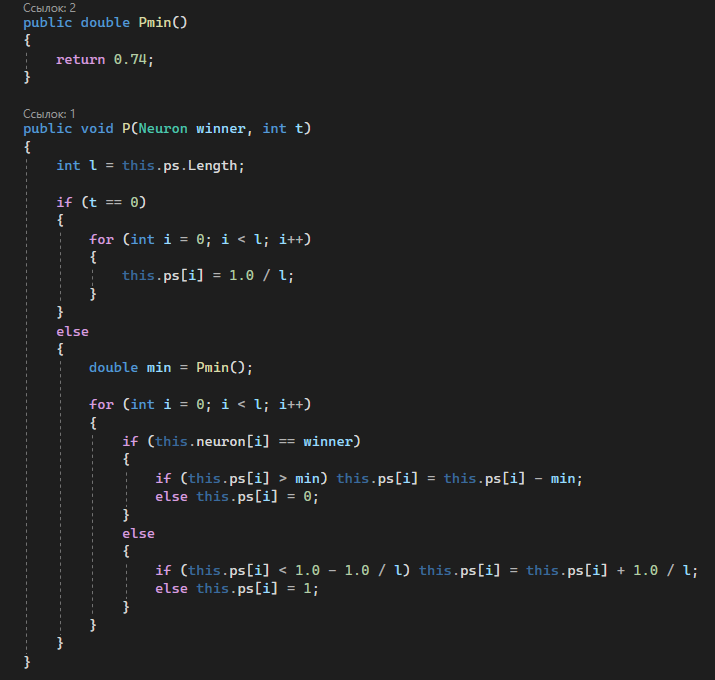
Рисунок 3 - Класс MyRandom

Далее рассмотрим класс KohonenNet представляющий собой сеть Кохонена:









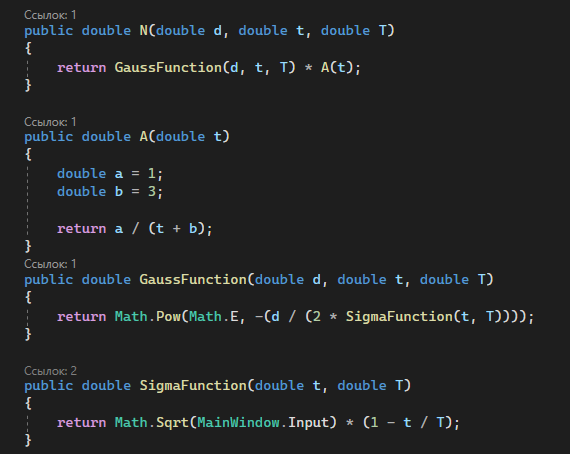


Рисунок 4 - Класс KohonenNet

Содержит поля:

* List<<Neuron> neuron – список нейронов Neuron;
* double[] ps – массив потенциалов нейронов.

И методы:

* bool IsCompetitive(Neuron n) – метод, принимающий нейрон и возвращающий true, если потенциал нейрона достаточно велик, чтобы иметь возможность участвовать в конкуренции;
* Neuron Winner(double[] xs, bool isP) – метод, принимающий входной вектор и параметр, отвечающий за то, учитывать ли потенциалы при определения победителя или нет. Возвращает нейрон победитель;
* List<Neuron> AllChangers(Neuron winner, double t, double T) – метод, возвращающий нейроны, которые попали в радиус обучения помимо нейрона победителя. Принимает нейрон-победитель winner, вокруг которого будет вычисляться радиус по формуле , где A = const, а также принимает текущий номер итерации t и общее число итераций T;
* double E(List<double[]> dataSet) – метод вычисления ошибки по формуле . Принимает нормализованный список обучающего множества;
* void Recount(Neuron winner, List<Neuron> seconders, double[] xs, double t, double T) – метод перерасчета весов нейронов seconders относительно winner, и самого нейрона winner по формуле
* double minP() – метод возвращающий ;
* void P(Neuron winner, int t) – метод перерасчёта потенциалов на t шаге на основе условия , где j - это номер нейрона победителя, n – число нейронов.
* double N(double d, double t, double T) – расчет скорости обучения ;
* double A(double t, double T) – функция скорости обучения ;
* double GaussFunction(double d, double t, double T) - функция Гаусса ;
* double SigmaFunction(double t, double T) – функция радиуса обучения . Стоит отметить, что функция внутри себя задействует статическую переменную MainWindow.Input = 6, которая отвевает за число входных параметров. , так как это максимально возможное расстояние между двумя нейронам. Если вес первого нейрона и вес второго нейрона , то Евклидово расстояние . Таким образом, на первых итерация все нейроны будут попадать в этот радиус.

Перед рассмотрением класса KohonnenMap, рассмотрим класс Cell – ячейка карты Кохонена:

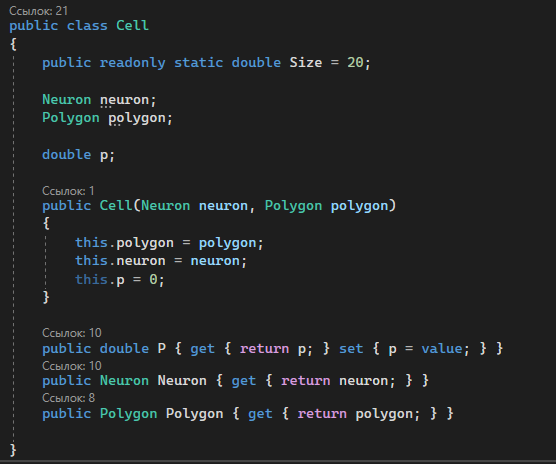
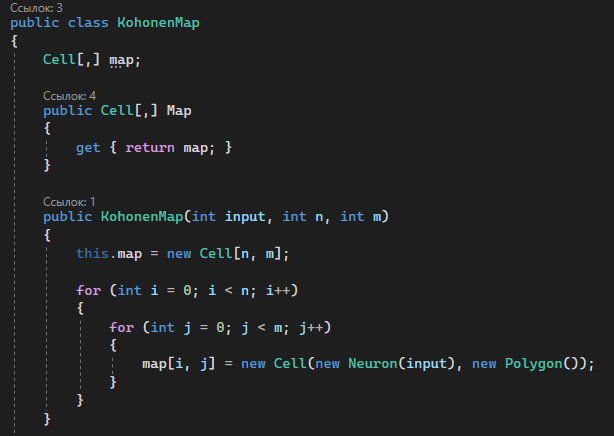


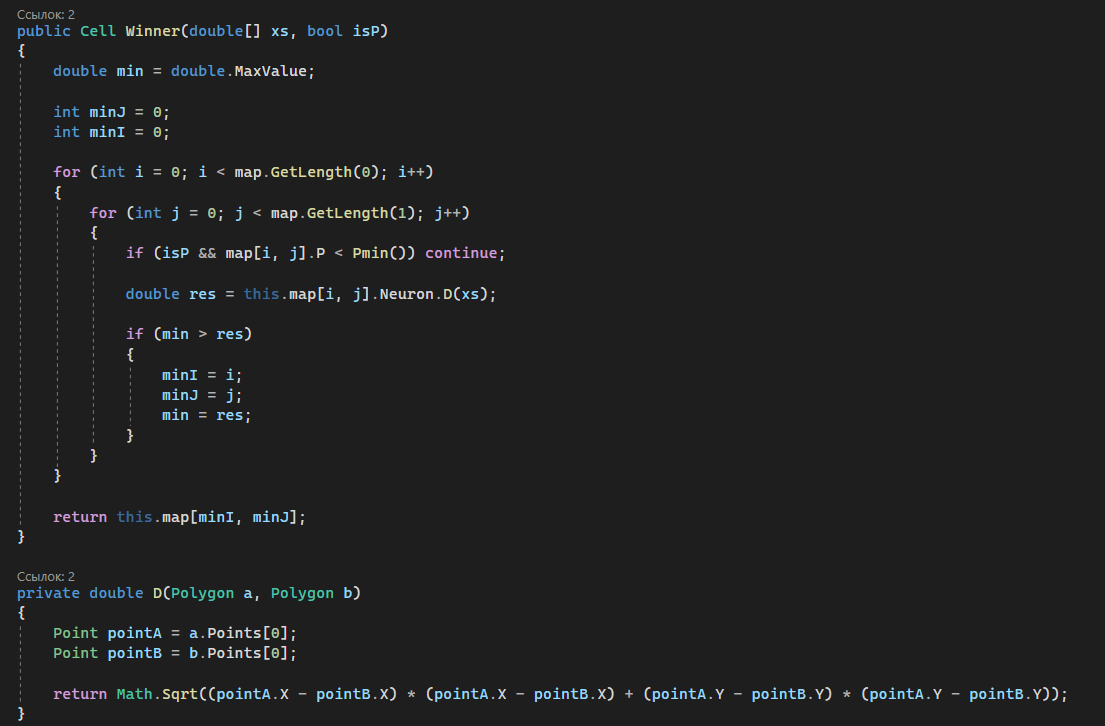
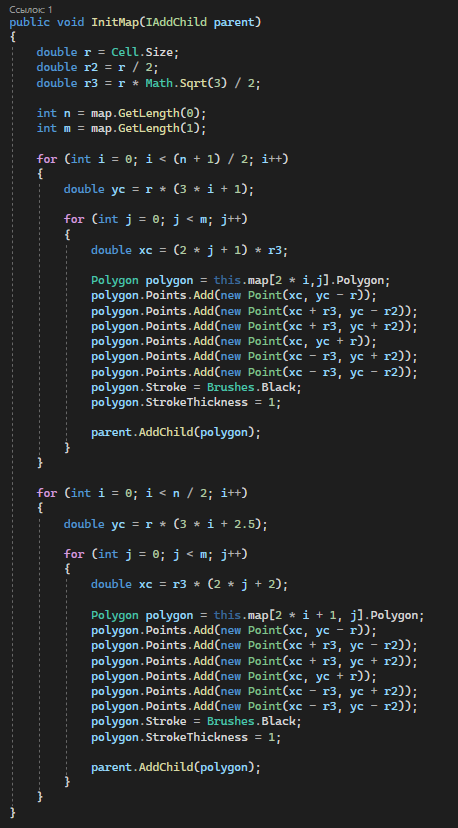
Рисунок 5 - Класс Cell

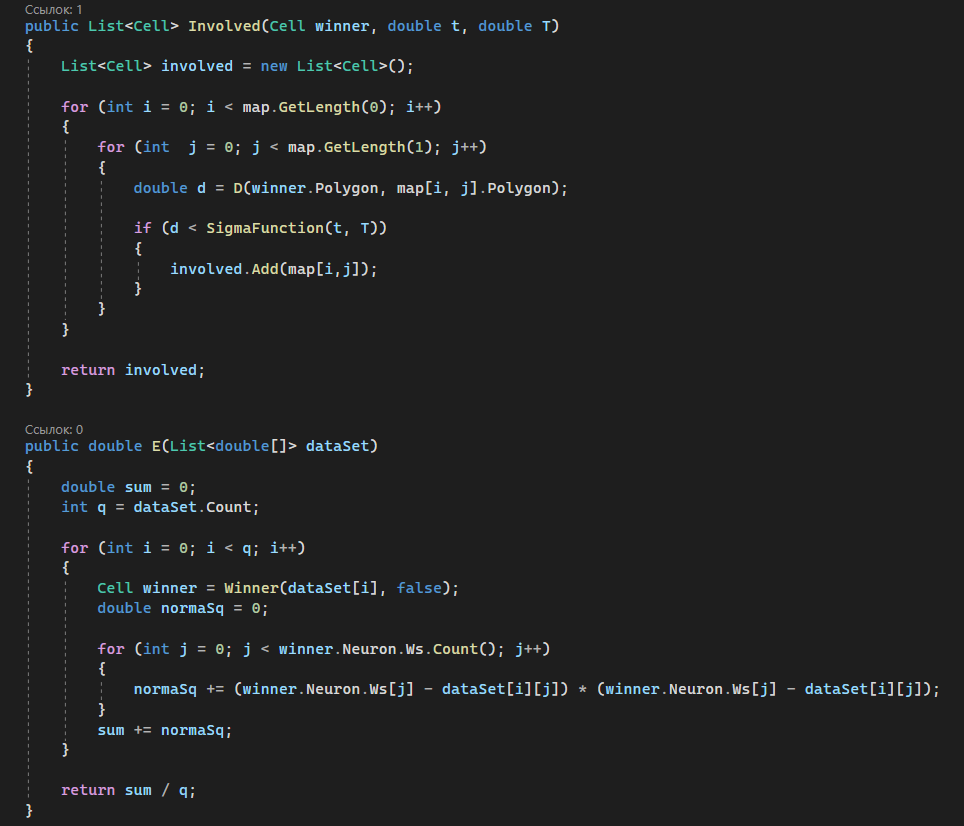
Содержит поля:

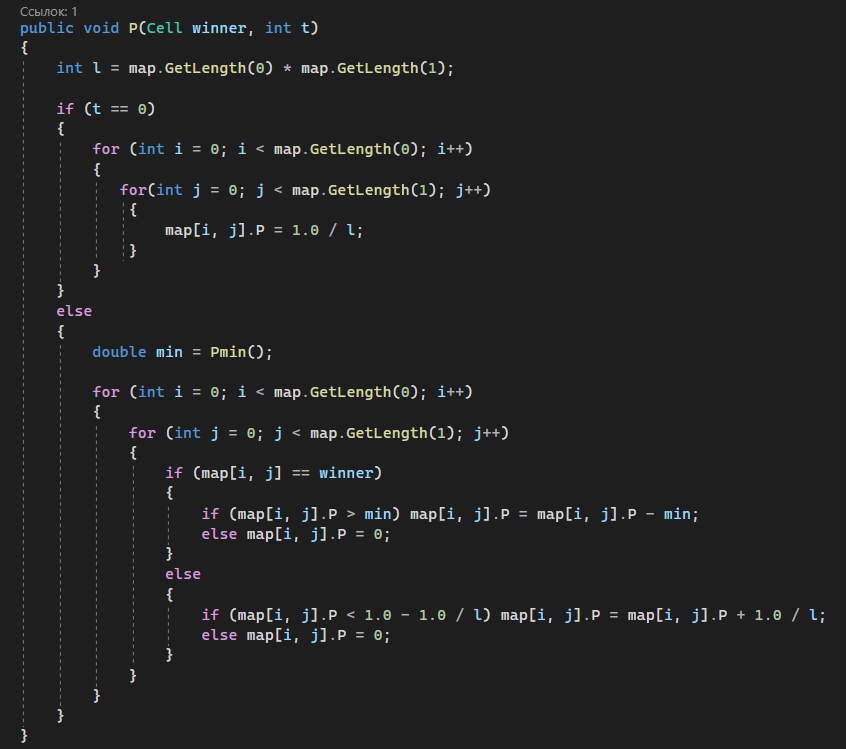
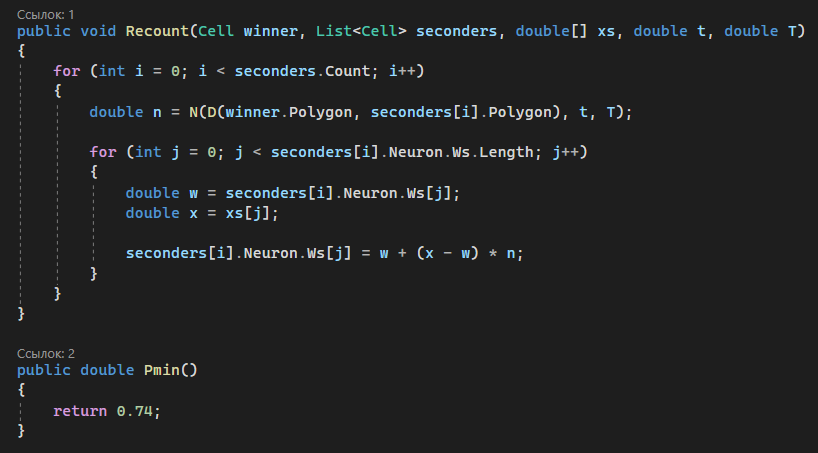
* Neuron neuron – нейрон типа Neuron;
* Polygon polydon – ссылка на ячейку карты;
* double p – потенциал нейрона neuron.

Теперь рассмотрим класс KohonenMap представляющий собой карту Кохонена:









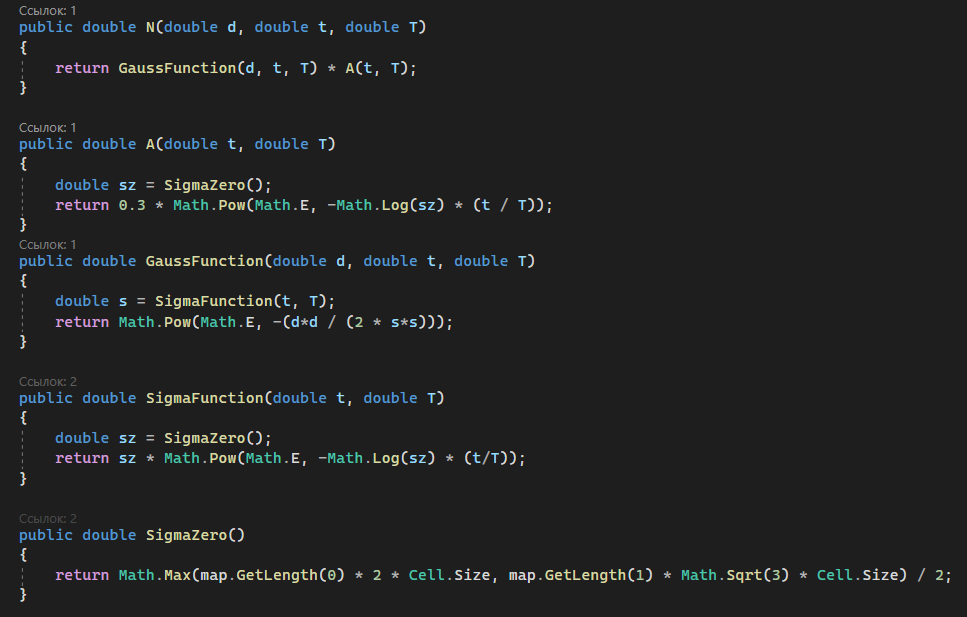


Рисунок 6 - Класс KohonenMap

Содержит полe:

* Cell[,] map – двумерное представление карты в виде матрицы ячеек;

И методы:

* Void InitMap (IAddChild parent) – метод, отрисовывающий карту Кохонена внутри элемента parent;
* Neuron Winner(double[] xs, bool isP) – метод, принимающий входной вектор и параметр, отвечающий за то, учитывать ли потенциалы при определения победителя или нет. Возвращает клетку карты с нейроном победителем;
* double D(Polygon a, Polygon b) – вспомогательный метод вычисления расстояния между шестиугольными ячейками;
* List<Cell> Involved(Neuron winner, double t, double T) – метод, возвращающий клетки карты с нейронами, которые попали в радиус обучения помимо нейрона победителя. Принимает нейрон-победитель winner, вокруг которого будет вычисляться радиус по формуле , где A = const, а также принимает текущий номер итерации t и общее число итераций T;
* double E(List<double[]> dataSet) – метод вычисления ошибки по формуле . Принимает нормализованный список обучающего множества;
* void Recount(Cell winner, List<Cell> seconders, double[] xs, double t, double T) – метод перерасчета весов нейронов в клетках seconders относительно нейрона победителя в клетке winner, и самого нейрона в клетке winner по формуле
* double minP() – метод возвращающий ;
* void P(Neuron winner, int t) – метод перерасчёта потенциалов на t шаге на основе условия , где j - это номер нейрона победителя, n – число нейронов.
* double N(double d, double t, double T) – расчет скорости обучения ;
* double A(double t, double T) – функция скорости обучения ;
* double GaussFunction(double d, double t, double T) – функция Гаусса ;
* double SigmaFunction(double t, double T) – функция радиуса обучения .
* double SigmaZero() – функция расчета константы , где N, M – ширина и высота карты, W,H – ширина и высота нейрона.

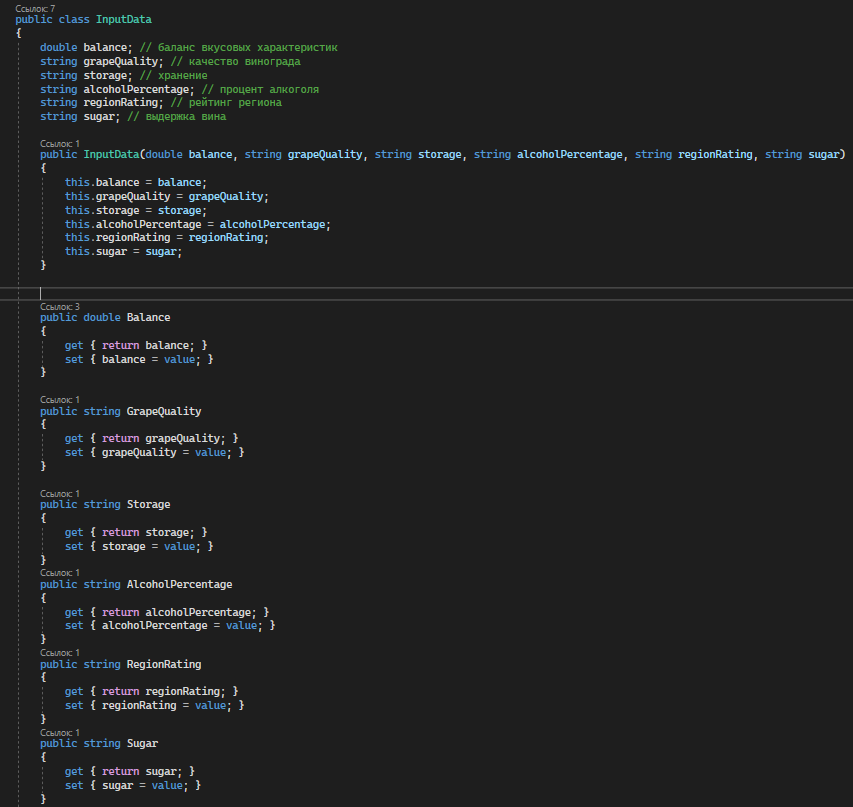
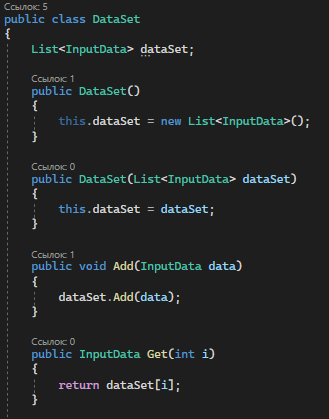
Рассмотрим класс InputData – класс для входных параметров:

Рисунок 7 - Класс InputData

Класс, в котором устанавливаются значения DataSet.



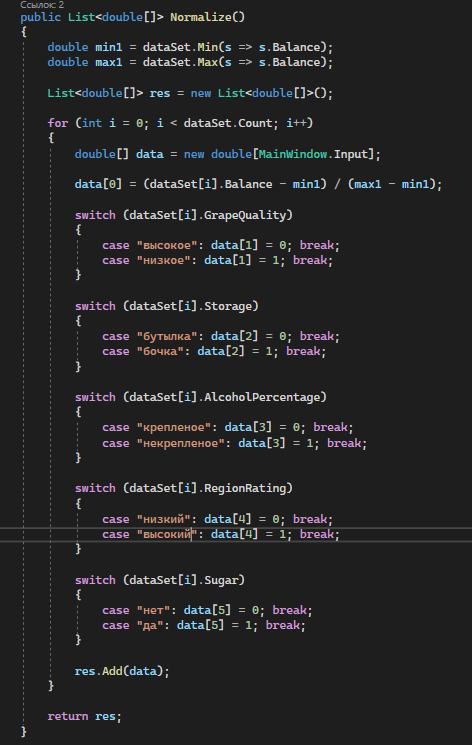


Рисунок 8 - Класс DataSet

Интерфейс взаимодействия представлен на рисунке 1 и состоит из трех блоков: Сеть Кохонена, Карта Кохонена, Визуализация.

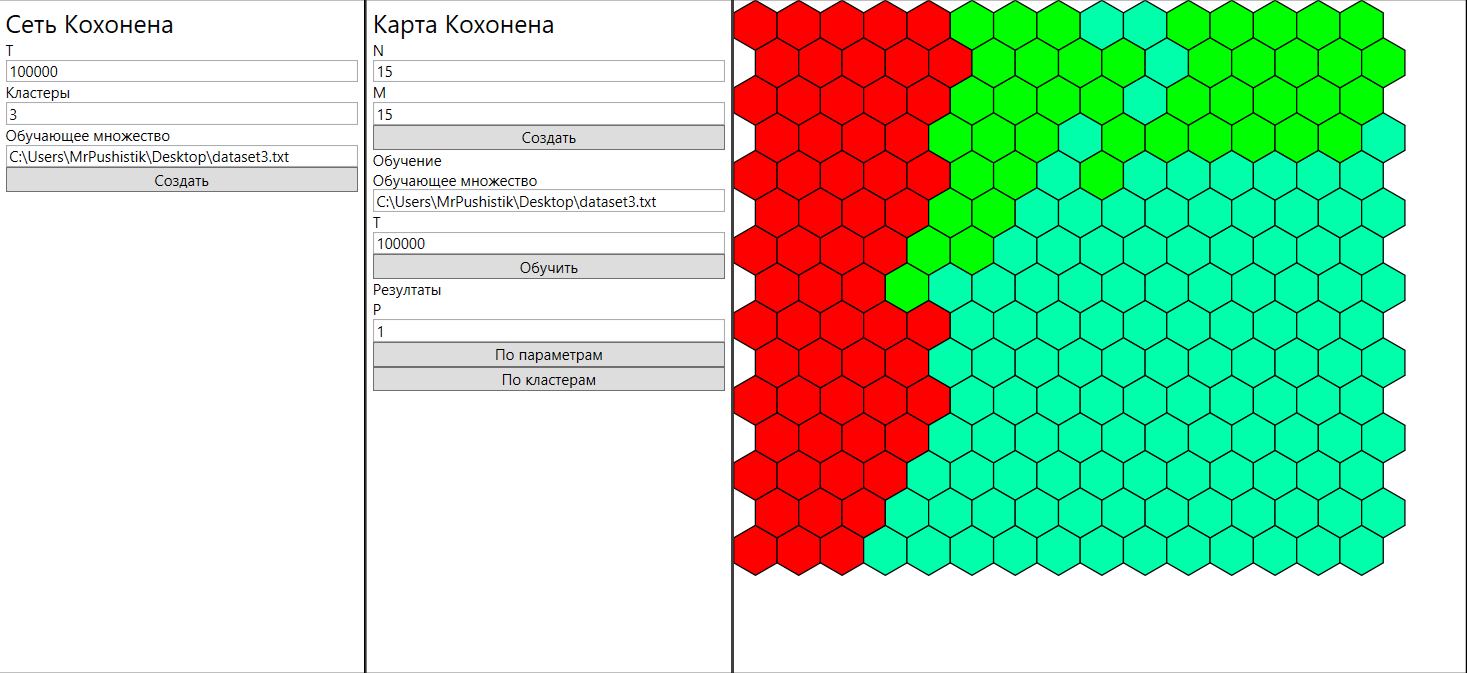


Рисунок 9 – Интерфейс

В блоке “Сеть Кохонена” можно произвести разделение обучающего множества на кластеры при помощи сети Кохонена. Для этого достаточно указать путь к файлу с обучающей выборкой, число желаемых кластеров и количество циклов обучения. Результат разбиения будет представлен в всплывающем окне с информацией о том, как удалось распределить объекты по кластерам, и значение ошибки E, по которой можно оценить удачность такого разбиения (Рисунок 2).

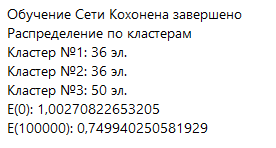


Рисунок 10 - Пример результата разбиения на кластеры

После этого можно перейти во вкладку “Карта Кохонена”. Сначала создадим карту размером N на M. После чего на вкладке визуализации построится пустая карта Кохонена (Рисунок 3).

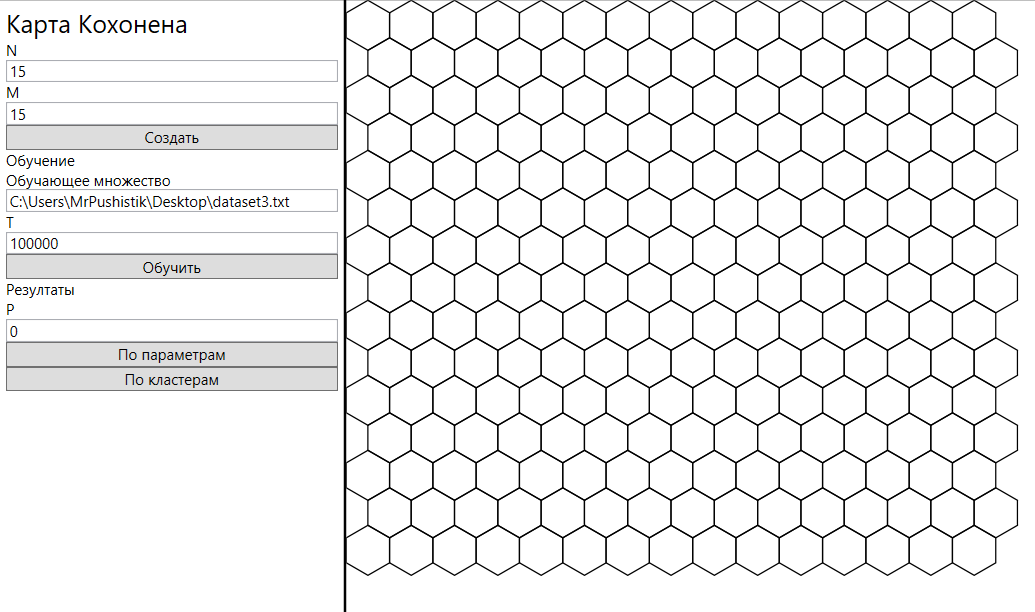


Рисунок 11 - Построение пустой карты Кохонена

После построения карты Кохонена можно произвести ее обучение. Для этого указываем путь до обучающей выборки, затем число циклов обучения. Когда обучение завершиться, появится соответствующее диалоговое окно (Рисунок 4).

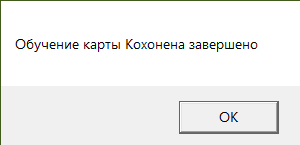
****

Рисунок 12 - Оповещение о завершении обучения карты Кохонена

После этого можно изучать результаты кластеризации (Рисунок 5).

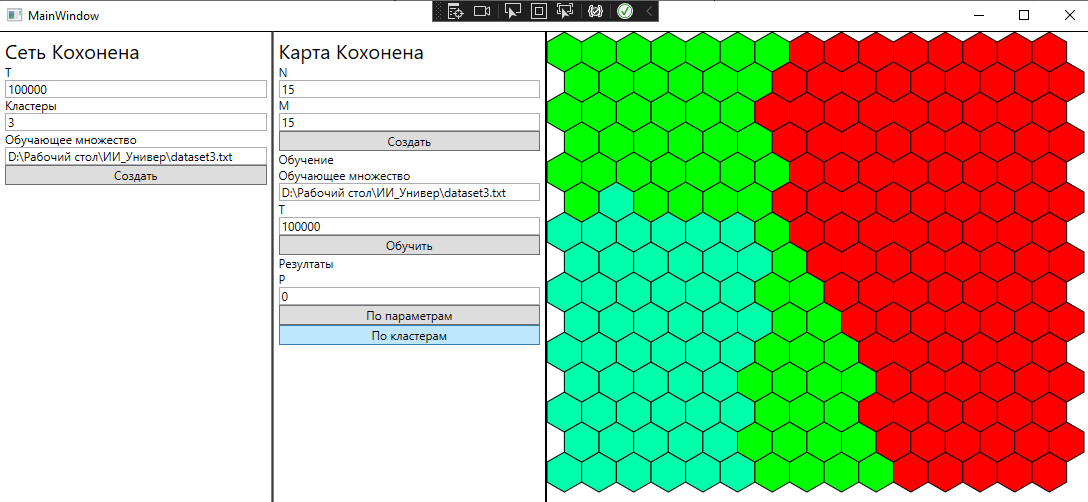


Рисунок 13 - Представление разбиения на кластеры на карте Кохонена

Меняя параметр P, можно визуализировать распределение p-ого признака на карте (Рисунок 6).

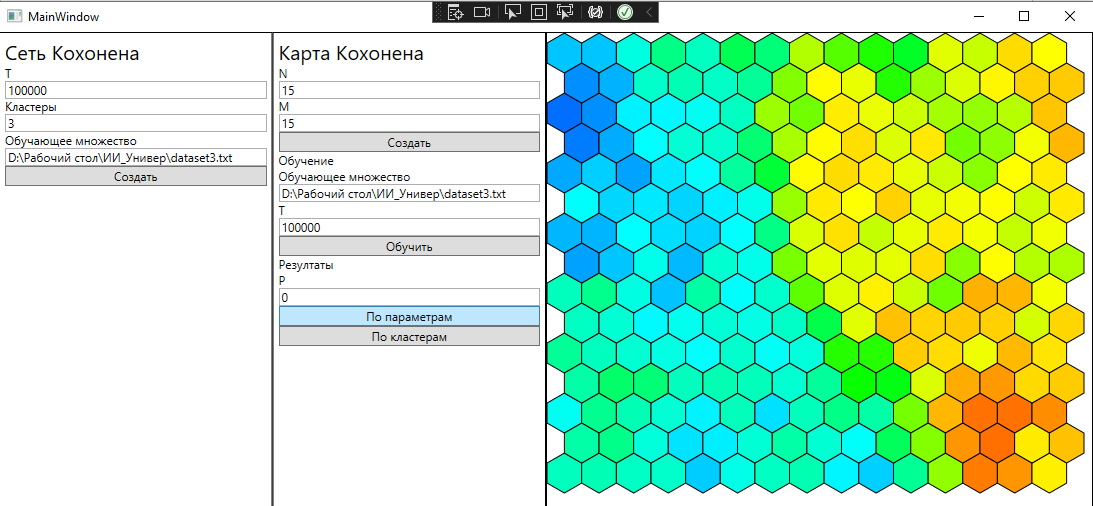


Рисунок 14 - Карта Кохонена на второго признака (P=2)

**Тестирование**

Произведем тестирование разработанного нейроэмулятора.

Первым шагом воспользуемся модулем “Сеть Кохонена”. Определим число циклов обучения T = 100000. Количество кластеров - 3. Результат работы на Рисунке 7.

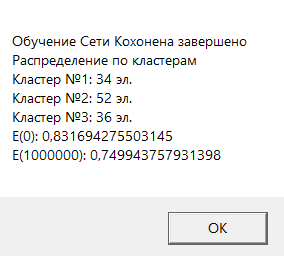


Рисунок 15 - Результат работы сети Кохонена

Создадим карту размером 15 на 15, поставим число циклов обучения T = 100000. Представим результат на рисунке 8. Пусть кластер 1 соответствует красному кластеру, кластер 2 – зеленому, кластер 3 – бирюзовому.

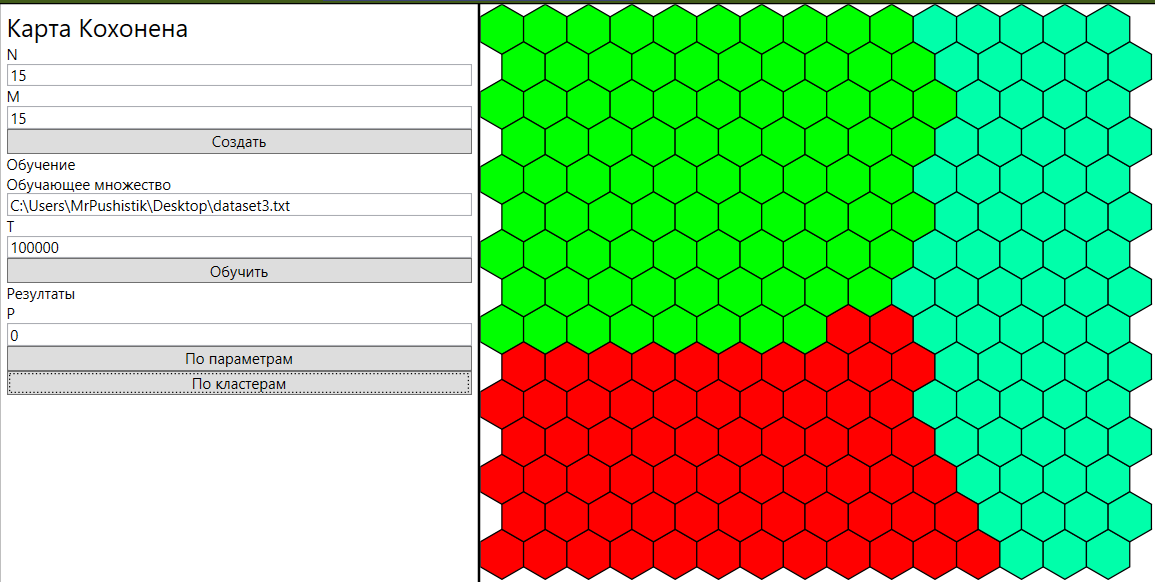


Рисунок 16 - Результат разбиения на кластеры на карте

Проанализируем карты для каждого параметра p. Отметим, что синий цвет соответствует наименьшему значению параметра, зеленый – среднему значению, красный – наибольшему значению.

Параметр p = 0 отвечает за баланс вкусовых характеристик (Рисунок 17). По данной карте видно, что кластеру 1 соответствуют экземпляры с наименьшими вкусовыми характеристиками.

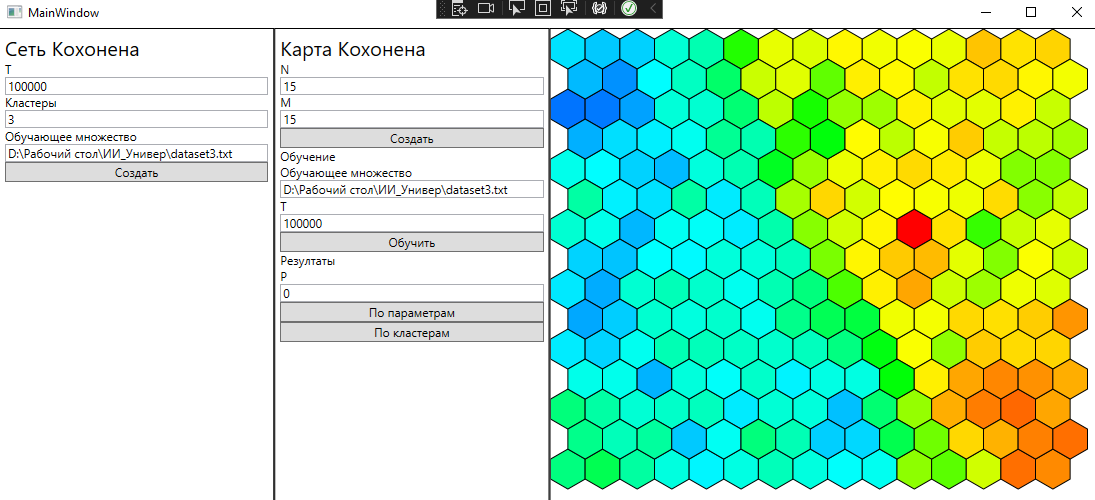


Рисунок 17 - Карта Кохонена при p = 0

Параметр p = 1 отвечает за качество винограда (Рисунок 18). По данной карте видно, что кластеру 1 соответствуют экземпляры с наиболее хорошим виноградом, кластеру 2 – экземпляры с не хорошим виноградом.

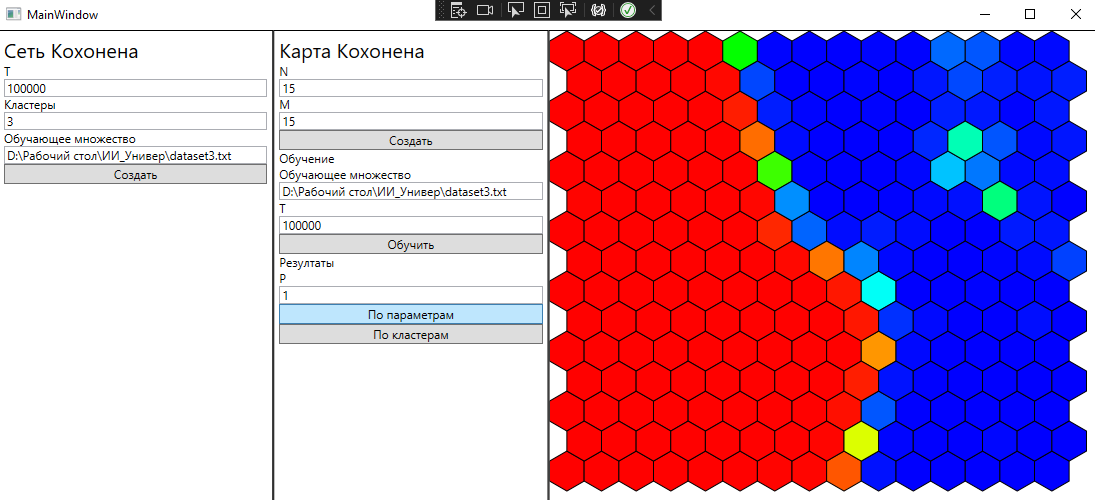


Рисунок 18 - Карта Кохонена при p = 1

Параметр p = 2 отвечает за хранение вина (Рисунок 19). По данной карте можно сказать, что хранение вина является признаком кластера 1.

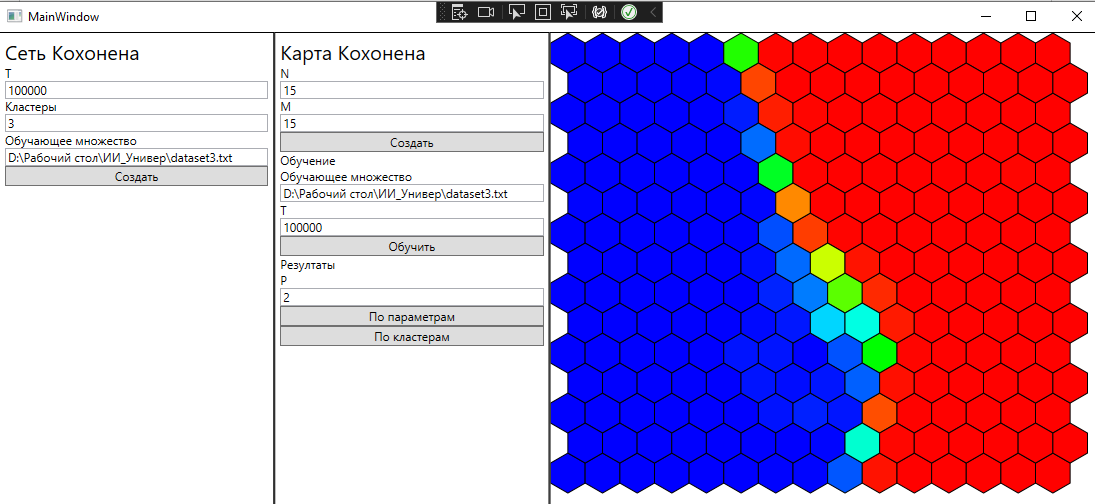


Рисунок 19 - Карта Кохонена при p = 2

Параметр p = 3 отвечает за процент алкоголя (Рисунок 20). По данной карте можно сказать, что процент алкоголя является признаком кластера 2.

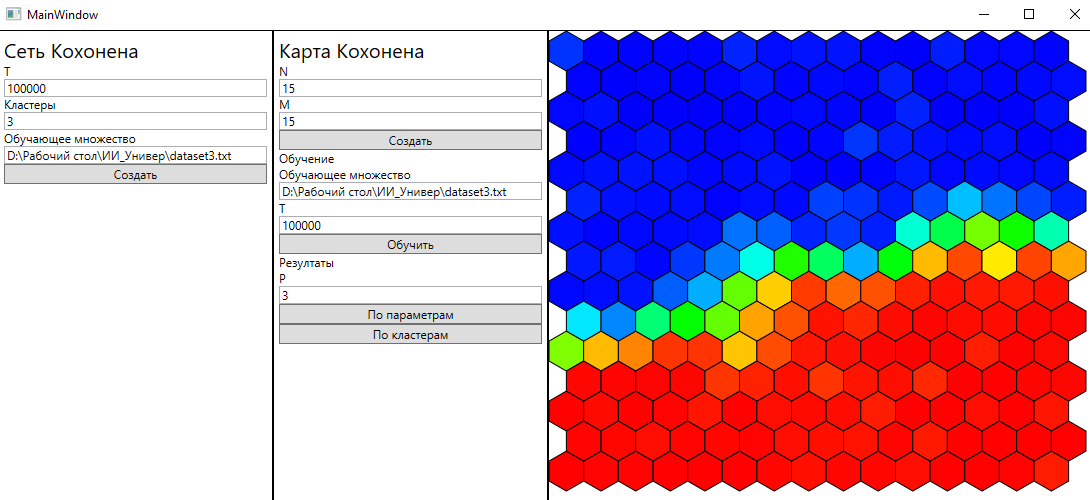


Рисунок 20 - Карта Кохонена при p = 3

Параметр p = 4 отвечает за рейтинг региона (Рисунок 21). По данной карте можно сказать, что рейтинг региона является признаком кластера 2.



Рисунок 21 - Карта Кохонена при p = 4

Параметр p = 5 отвечает за содержание сахара (Рисунок 22). По данной карте можно сказать, что содержание сахара является признаком кластера 1.

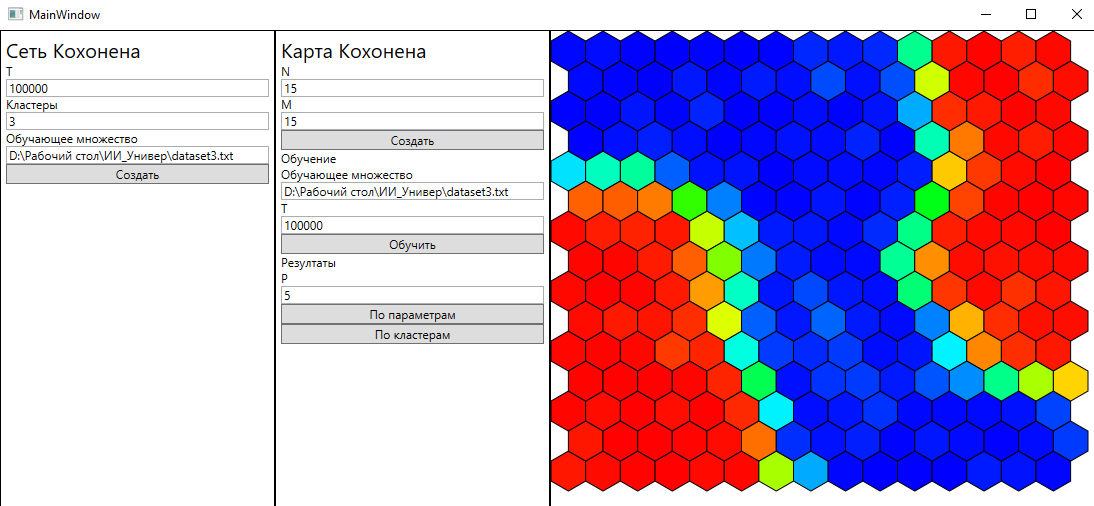


Рисунок 22 - Карта Кохонена при p = 5

**Вывод**

В ходе практической работы был разработан нейроэмулятор сети и карты Кохонена для решения задачи кластеризации.

Сначала разработаны классы Neuron и KohonenNet, которые составили основу нейроэмулятора и обеспечили его возможностью распределения объектов по кластерам. Была решена проблема мертвых нейронов путем учета потенциалов.

Потом реализован класс KohonenMap при помощи, которой стало возможным составлять карты Кохонена произвольного размера для представления распределения признаков на двумерной плоскости.

Далее был составлен интерфейс приложения, в дополнение к которому были добавлены такие возможности, как установление числа циклов обучения, отслеживание значения ошибки, загрузка обучающего множества через текстовый файл.

После этого произведено тестирование. Результаты кластеризации сетью и картой Кохонена в большей части совпали с реальными данными качества вина, что характеризует адекватность разработанной модели.