НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Факультет автоматики и вычислительной техники

Кафедра ЭВМ

Отчёт

Лабораторная работа № 2 по дисциплине

«Системы обработки знаний»

Выполнил студент группы ИВТб-4301\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Завьялов Н.В./

Проверил доцента кафедры ЭВМ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Ростовцев В. С./

Киров 2024

1. Цель

Ознакомиться с основными командами создания, обучения и применения нейронных сетей кластеризации и классификации данных в Neural Network Toolbox с помощью сетей Кохонена и LVQ.

1. Задание

1) Выполнить кластеризации с применением нейронных сетей Кохонена в соответствие с заданием (Приложение А). Создать и выполнить моделирование нейронной cети Кохонена, согласно варианту, согласованному с преподавателем, приведенному в приложении А. Результаты кластеризации вывести в виде графика с разными цветами кластеров, а также точки тестовых векторов.

2) Для создания сети LVQ обучающую последовательность использовать сеть Кохонена.

1. Протокол выполнения
   1. Самоорганизующаяся карта Кохонена
      1. Обучение

Для кластеризации данных был найдена датасет, состоящий из количества пропусков студентов и их оценок. Датасет приведен в приложении А.

Для отображения элементов выборки был выполнен следующий код в Matlab:

plot(student1(1,:),student1(2,:),'.g','markersize',20)

hold on

Результат выполнения команд представлен на рисунке 1.

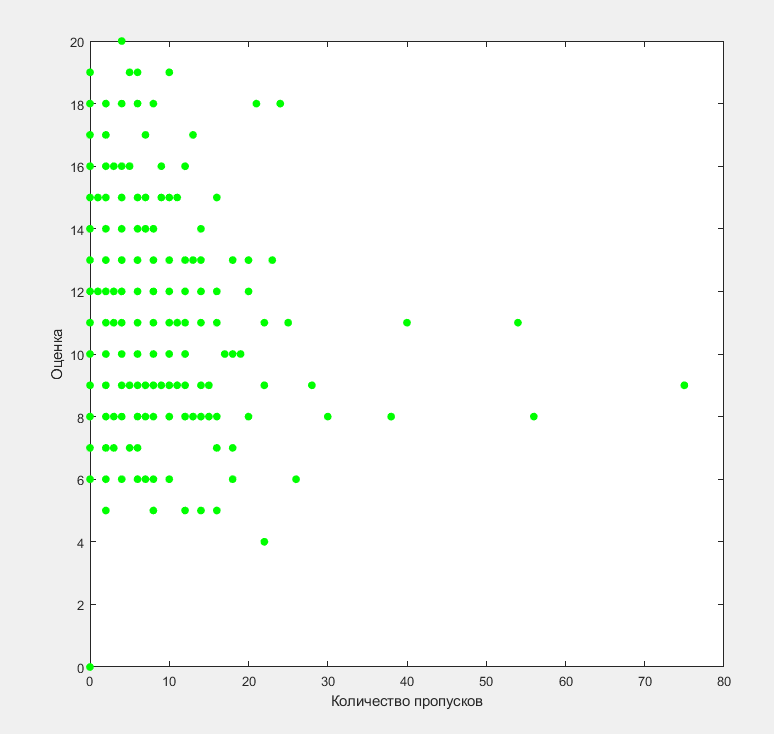


Рисунок 1 – Отображение элементов обучающей выборки

Из-за того, что в обучающей выборке есть похожие данные, то точки накладываются друг на друга.

Для создания самоорганизующейся карты Кохонена выл выполнен следующий код в программе MatLab:

net = selforgmap([6,6]);

Была создана гексагональная сетка размера 6х6, что соответствует 36 нейронам или, условно, кластерам.

Для обучения сети использовалась следующая команда:

[net, tr] = train(net, indata);

Процесс обучения изображен на рисунке 2.

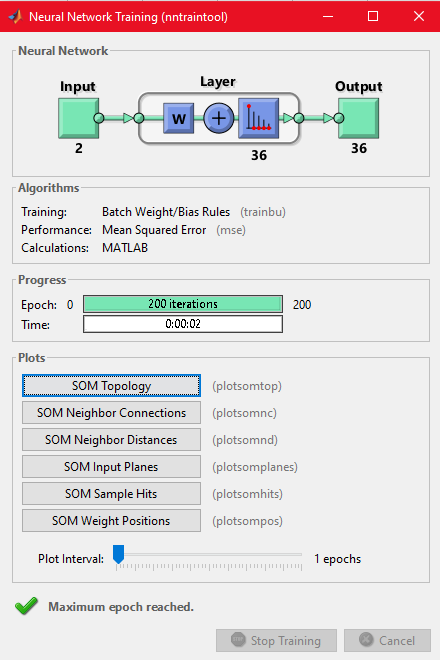


Рисунок 2 – Процесс обучения сети

В результате были получены новые значения весов нейронов. Узлы карты Кохонена были отображена на графике с помощью следующего кода:

W = net.iw{1,1}

for i = 1:length(W)

text(W(i,1),W(i,2),int2str(i),'HorizontalAlignment','center', 'VerticalAlignment', 'bottom')

end

hold on

plotsom(net.iw{1,1},net.layers{1}.distances)

hold off

Результат выполнения кода представлен на рисунке 2.

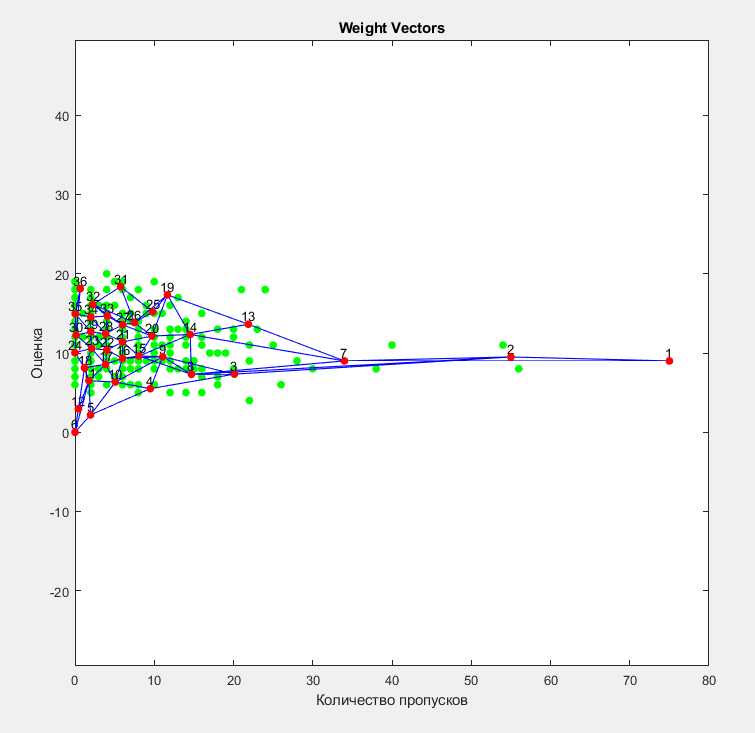


Рисунок 3 – Отображение узлов карты Кохонена после обучения

Как видно из полученного графика, были выделены кластеры, размером с заданной картой. Особо выделаются крайние кластеры, которые соответствуют группам людей с редким весом или ростом.

* + 1. Тестирование сети Кохонена

Для тестирования обучения сети выделим несколько примеров, которые не участвовали в обучении:

* Человек с количеством пропусков 80 и оценкой 20.
* Человек с количеством пропусков 35 и оценкой 20.
* Человек с количеством пропусков 5 и оценкой 5.

Отобразим их на графике синим цветом, рисунок 4.

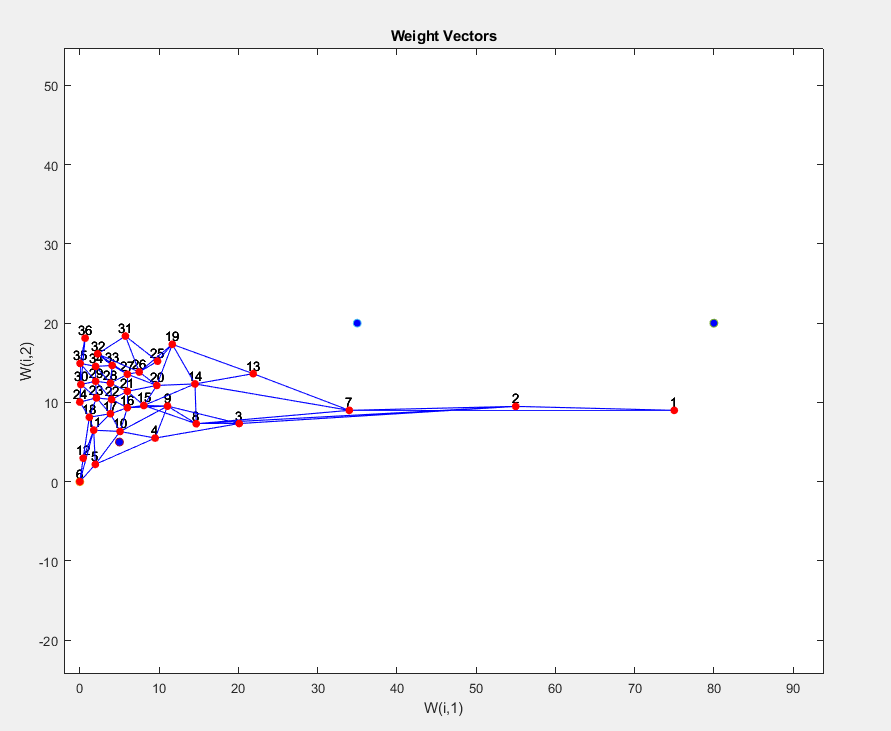


Рисунок 4 – Отображение тестовых примеров на графике синим цветом

Выполним следующий код:

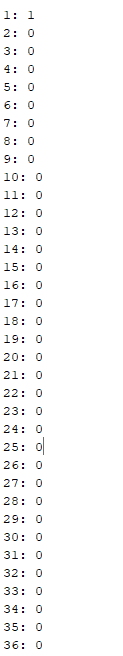
res = sim(net, [100; 195])

for i = 1:length(res)

disp([num2str(i) ': ' num2str(res(i))])

end

Вывод представлен на рисунке 5.



* Рисунок 5 – Тестирование принадлежности человека с количеством пропусков 80 и оценкой 20.

Как видно из рисунка 5, тестирование прошло успешно – человек с количеством пропусков 80 и оценкой 20 был отнесен в группу 1, что соответствует графике, где он близок к узлу под номером 1.

Проделаем аналогичные действия для человека с ростом 168 и весом 60, а также с ростом 172 и весом 77, рисунки 6 и 7.

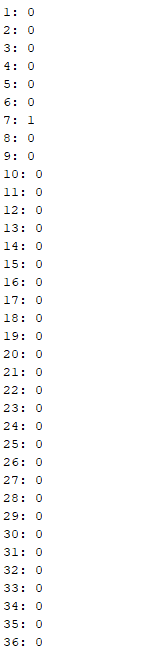


Рисунок 6 – Тестирование принадлежности человека с количеством пропусков 35 и оценкой 20.

Как видно из рисунка 6, тестирование прошло успешно – человек с количеством пропусков 35 и оценкой 20 был отнесен в группу 7, что соответствует графике, где он близок к узлу под номером 7.

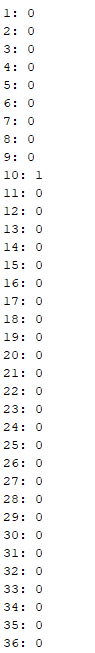


Рисунок 7 – Тестирование принадлежности человека с количеством пропусков 5 и оценкой 5.

Как видно из рисунка 7, тестирование прошло успешно – человек с количеством пропусков 5 и оценкой 5 был отнесен в группу 10, что соответствует графике, где он близок к узлу под номером 10.

* + 1. Выводы по сети Кохонена

В данном разделе была создана сеть Кохонена в программе Matlab. Для неё была найдена обучающая выборка, состоящая из количества пропусков студентов и их оценки. Сеть была обучена, в результате чего экземпляры выборки были отнесены к определенным кластерам, что было показано на графике. Для тестирования использовались 3 примера, которые не участвовали в обучении сети.

* 1. Сеть LVQ
     1. Создание и обучение

Для создания сети LVQ возьмем обучающую последовательность, используя результаты сети Кохонена.

Разобьем людей на 12 классов с помощью сети Кохонена. Результат показан на рисунке 8.

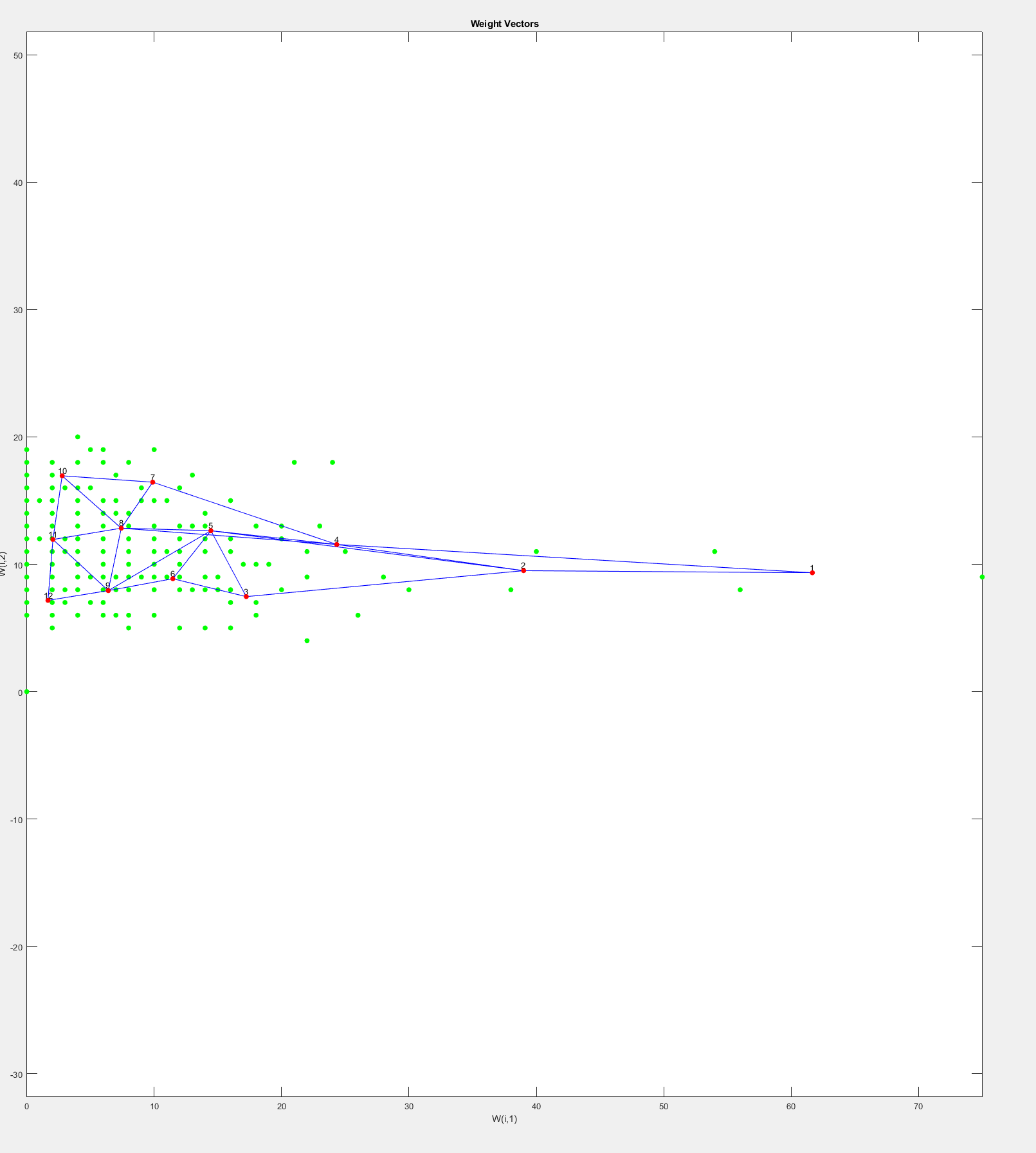


Рисунок 8 – Разбиение людей на 12 классов

Заменим точки числами их классов, также зададим им разный цвет следующим кодом:

T = []

for i = 1:length(student)

y = net([student (1, i); student (2, i)]);

cluster\_index = vec2ind(y);

T = [T cluster\_index];

end

colors = ['r', 'r', 'r', 'r', 'b','b','b','b','g','g','g','g','c','c','c','c']

figure(1), clf, axis([min(student (1,:))-5,max(student (1,:))+5,min(student (2,:))-5,max(student (2,:)+5)]), hold on

for i = 1:12

tmp = find(T==i);

s = num2str(i) + 'r'

% plot(indata(1,tmp),indata(2,tmp), s)

text(student (1,tmp), student (2,tmp), num2str(i),'Color',colors(i))

hold on

end

Результат представлен на рисунке 9.

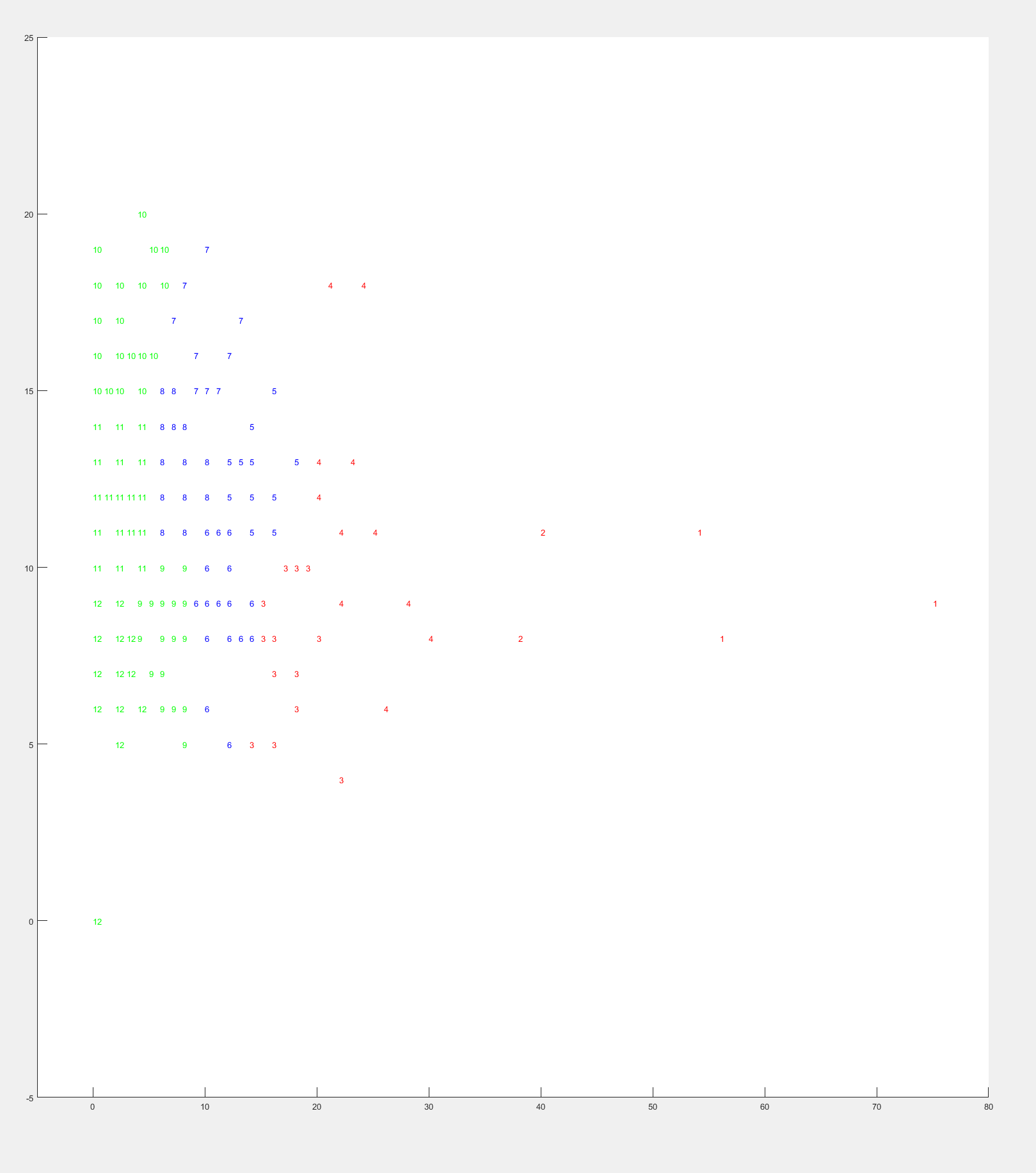


Рисунок 9 – Визуальное представление классов

Создадим сеть LVQ и обучим её:

t = ind2vec(T);

lnet = lvqnet(18,0.2,'learnlv2');

lnet.trainParam.epochs=150;

lnet=train(lnet,indata,t);

Процесс обучения представлен на рисунке 10.

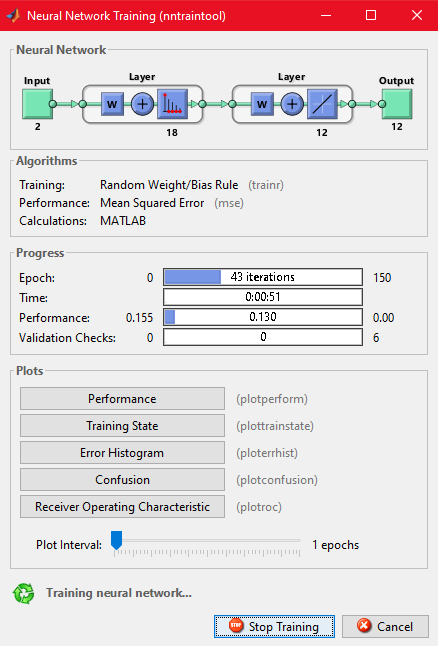
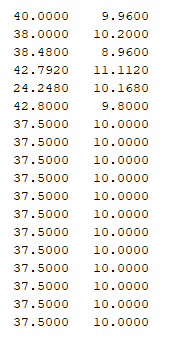


Рисунок 10 – Процесс обучения сети LVQ

После обучения получаем следующие весовые коэффициенты нейронов конкурирующего слоя, которые определяют положения центров кластеризации:



Отобразим их на графике (рисунок 11, цвет точек центров не зависит от цветов изначальных кластеров):

plotvec(lnet.IW{1}',vec2ind(lnet.LW{2}),'o');

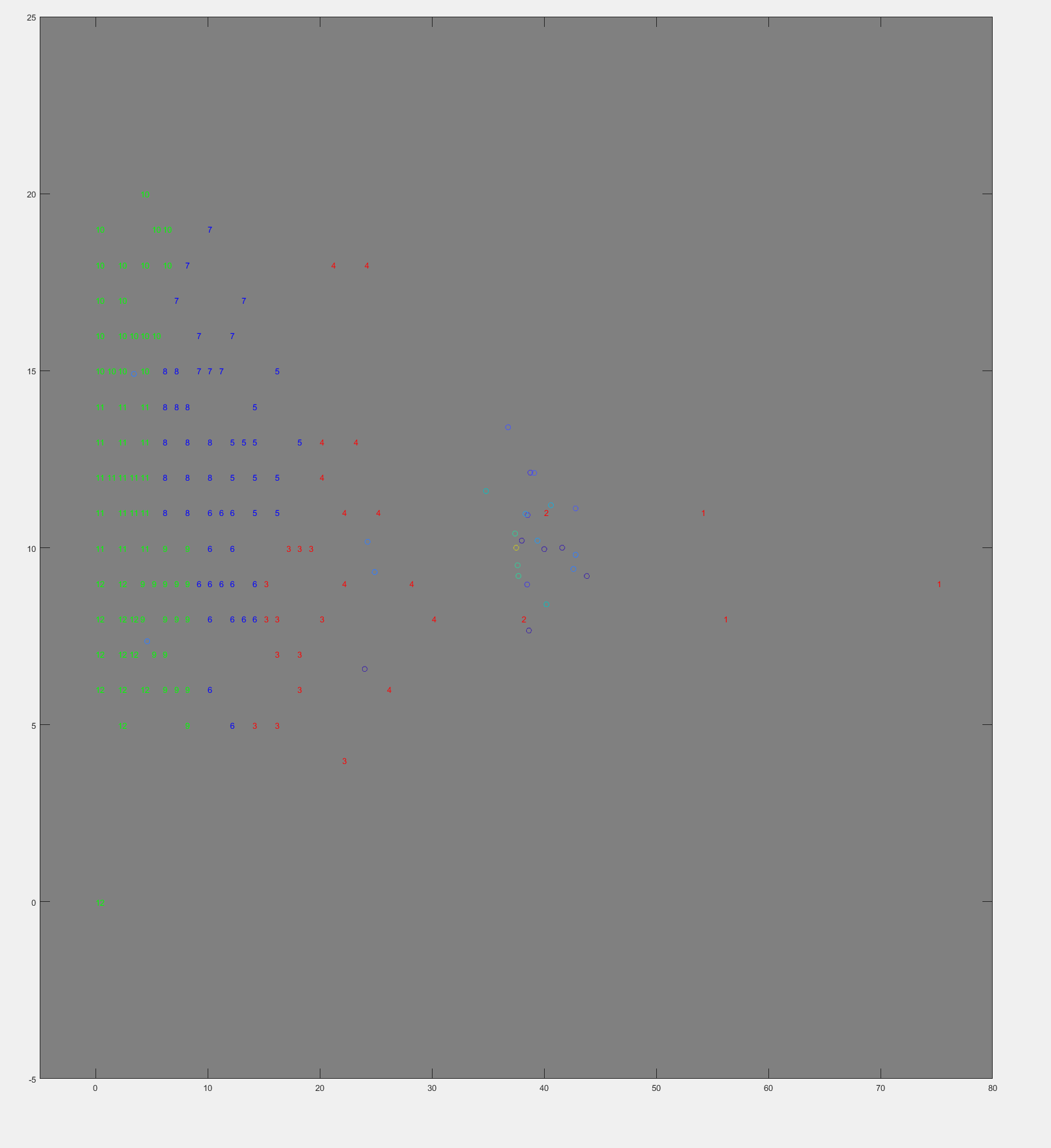


Рисунок 11 – Центры кластеризации

Наибольшее скопление кластеров произошло на стыке 2, 4, и 1 групп.

* + 1. Тестирование сети

Для тестирования сети возьмем 3 примера:

* Человек с количеством пропусков 40 и оценкой 10.
* Человек с количеством пропусков 20 и оценкой 3.
* Человек с количеством пропусков 5 и оценкой 5.

Тестирование показано на рисунке 12.

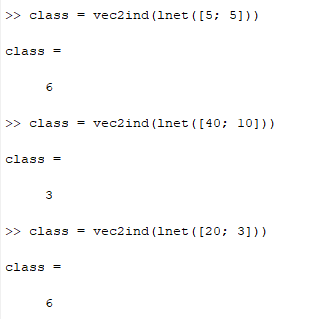


Рисунок 12 – Тестирование сети

Как видно из тестирования, есть смещения центров кластеров относительно изначальных групп, поэтому ответы не совпадают с их классами. Например, человек с количеством пропусков 40 и оценкой 10 по факту принадлежит 2 классу, но тестирование показало 3, но при этом он также расположен не далеко от этого класса.

Человек с количеством пропусков 20 и оценкой 3 принадлежит классу 3 , но тестирование показало 6, но при этом он также расположен не далеко от этого класса.

Человек с количеством пропусков 5 и оценкой 5 принадлежит классу 12 , но тестирование показало 6, но при этом он также расположен не далеко от этого класса.

* + 1. Выводы по сети LVQ

В данном разделе была создана сеть LVQ в программе Matlab. В отличие от сети Кохонена, сеть LVQ требует разбиение входных векторов на классы, поэтому для обучения сети были взяты результаты обучения сети Кохонена. Сеть была обучена, в результате чего экземпляры выборки были отнесены к определенным кластерам, что было показано на графике. Для тестирования использовались 3 примера, которые не участвовали в обучении сети.

1. Выводы

В данной лабораторной работе были изучены основные команды создания, обучения и применения нейронных сетей кластеризации и классификации данных в Neural Network Toolbox с помощью сетей Кохонена и LVQ.

Основная идея кластерного анализа (clustering, cluster analysis) заключается в том, чтобы разбить объекты на группы или кластеры таким образом, чтобы внутри группы эти наблюдения были более похожи друг на друга, чем на объекты другого кластера.

Алгоритм функционирования самообучающихся карт (Self Organizing Maps — SOM) представляет собой один из вариантов кластеризации многомерных векторов. Примером таких алгоритмов может служить алгоритм k-ближайших средних (k-means). Важным отличием алгоритма SOM является то, что в нем все нейроны (узлы, центры классов…) упорядочены в некоторую структуру (обычно двумерную сетку).

При этом в ходе обучения модифицируется не только нейрон-победитель, но и его соседи, но в меньшей степени. За счет этого SOM можно считать одним из методов проецирования многомерного пространства в пространство с более низкой размерностью. При использовании этого алгоритма вектора, схожие в исходном пространстве, оказываются рядом и на полученной карте.

LVQ-сети основаны на обучающемся векторном квантовании (LVQ — Learning Vector Quantization) и представляют собой слой Кохонена, обучающийся с учителем. Для построения LVQ-сети задается количество кластеров (нейронов) n, количество классов m ( n ³ m ) и принадлежность каждого кластера определенному классу. Например, анализы пациентов разделяются на 5 кластеров, 2 из которых соответствуют здоровым людям, а 3 — больным. Разделить кластеры по классам можно в тех же пропорциях, что и распределение примеров соответствующих классов в обучающей выборке. Соответствие между номерами кластеров и номерами классов может быть произвольным. Для простоты интерпретации результатов работы сети целесообразно пронумеровать кластеры последовательно. В процессе обучения LVQ-сети веса нейронов настраиваются с учетом принадлежности обучающих примеров и кластеров одному классу. Обученная LVQ-сеть производит кластеризацию входных векторов с учетом классов.

Приложение А

Датасет для самоорганизующейся карты Кохонена. Данные взяты с https://archive.ics.uci.edu/dataset/320/student+performance

|  |  |
| --- | --- |
| Пропуски | Оценка |
| 6 | 6 |
| 4 | 6 |
| 10 | 10 |
| 2 | 15 |
| 4 | 10 |
| 10 | 15 |
| 0 | 11 |
| 6 | 6 |
| 0 | 19 |
| 0 | 15 |
| 0 | 9 |
| 4 | 12 |
| 2 | 14 |
| 2 | 11 |
| 0 | 16 |
| 4 | 14 |
| 6 | 14 |
| 4 | 10 |
| 16 | 5 |
| 4 | 10 |
| 0 | 15 |
| 0 | 15 |
| 2 | 16 |
| 0 | 12 |
| 2 | 8 |
| 14 | 8 |
| 2 | 11 |
| 4 | 15 |
| 4 | 11 |
| 16 | 11 |
| 0 | 12 |
| 0 | 17 |
| 0 | 16 |
| 0 | 12 |
| 0 | 15 |
| 0 | 6 |
| 2 | 18 |
| 7 | 15 |
| 2 | 11 |
| 8 | 13 |
| 25 | 11 |
| 8 | 12 |
| 2 | 18 |
| 0 | 11 |
| 14 | 9 |
| 8 | 6 |
| 12 | 11 |
| 4 | 20 |
| 2 | 14 |
| 2 | 7 |
| 2 | 13 |
| 2 | 13 |
| 6 | 10 |
| 0 | 11 |
| 6 | 13 |
| 8 | 10 |
| 0 | 15 |
| 4 | 15 |
| 2 | 9 |
| 2 | 16 |
| 6 | 11 |
| 6 | 11 |
| 4 | 9 |
| 2 | 9 |
| 0 | 10 |
| 2 | 15 |
| 4 | 12 |
| 4 | 6 |
| 2 | 8 |
| 12 | 16 |
| 0 | 15 |
| 0 | 10 |
| 2 | 5 |
| 2 | 14 |
| 54 | 11 |
| 6 | 10 |
| 8 | 10 |
| 0 | 11 |
| 2 | 10 |
| 12 | 5 |
| 2 | 12 |
| 4 | 11 |
| 10 | 6 |
| 4 | 15 |
| 2 | 10 |
| 6 | 8 |
| 4 | 6 |
| 4 | 14 |
| 12 | 10 |
| 18 | 7 |
| 0 | 8 |
| 4 | 18 |
| 4 | 6 |
| 0 | 10 |
| 6 | 14 |
| 2 | 10 |
| 2 | 15 |
| 2 | 10 |
| 6 | 14 |
| 0 | 8 |
| 14 | 5 |
| 0 | 17 |
| 4 | 14 |
| 26 | 6 |
| 0 | 18 |
| 10 | 11 |
| 8 | 8 |
| 2 | 18 |
| 6 | 13 |
| 4 | 16 |
| 6 | 19 |
| 0 | 10 |
| 6 | 13 |
| 10 | 19 |
| 8 | 9 |
| 2 | 16 |
| 2 | 14 |
| 0 | 13 |
| 20 | 8 |
| 6 | 13 |
| 2 | 15 |
| 6 | 15 |
| 2 | 13 |
| 18 | 13 |
| 0 | 8 |
| 0 | 12 |
| 0 | 11 |
| 2 | 9 |
| 0 | 0 |
| 8 | 18 |
| 0 | 0 |
| 0 | 0 |
| 12 | 12 |
| 16 | 11 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 12 |
| 0 | 15 |
| 0 | 0 |
| 8 | 9 |
| 2 | 11 |
| 2 | 13 |
| 0 | 0 |
| 0 | 11 |
| 0 | 0 |
| 2 | 11 |
| 0 | 0 |
| 0 | 10 |
| 0 | 0 |
| 6 | 14 |
| 8 | 10 |
| 0 | 0 |
| 0 | 12 |
| 2 | 8 |
| 8 | 13 |
| 6 | 10 |
| 2 | 15 |
| 4 | 12 |
| 0 | 0 |
| 6 | 7 |
| 0 | 0 |
| 2 | 10 |
| 0 | 7 |
| 16 | 12 |
| 4 | 10 |
| 0 | 16 |
| 0 | 0 |
| 0 | 14 |
| 0 | 0 |
| 2 | 16 |
| 0 | 10 |
| 0 | 0 |
| 4 | 9 |
| 4 | 9 |
| 2 | 11 |
| 4 | 6 |
| 10 | 9 |
| 4 | 11 |
| 10 | 8 |
| 2 | 12 |
| 0 | 17 |
| 56 | 8 |
| 14 | 12 |
| 12 | 11 |
| 2 | 11 |
| 0 | 15 |
| 6 | 9 |
| 4 | 10 |
| 10 | 13 |
| 0 | 9 |
| 12 | 8 |
| 8 | 10 |
| 0 | 14 |
| 0 | 15 |
| 4 | 16 |
| 8 | 10 |
| 24 | 18 |
| 0 | 10 |
| 2 | 16 |
| 6 | 10 |
| 4 | 10 |
| 18 | 6 |
| 6 | 11 |
| 28 | 9 |
| 5 | 7 |
| 10 | 13 |
| 6 | 10 |
| 6 | 7 |
| 10 | 8 |
| 13 | 13 |
| 0 | 14 |
| 15 | 8 |
| 12 | 10 |
| 2 | 15 |
| 22 | 4 |
| 13 | 8 |
| 3 | 8 |
| 4 | 10 |
| 2 | 6 |
| 0 | 0 |
| 2 | 17 |
| 0 | 13 |
| 0 | 14 |
| 16 | 7 |
| 10 | 15 |
| 2 | 12 |
| 14 | 9 |
| 10 | 12 |
| 14 | 14 |
| 4 | 11 |
| 14 | 9 |
| 2 | 13 |
| 18 | 6 |
| 10 | 10 |
| 4 | 13 |
| 20 | 12 |
| 2 | 11 |
| 0 | 0 |
| 14 | 12 |
| 2 | 12 |
| 0 | 0 |
| 0 | 12 |
| 0 | 0 |
| 6 | 18 |
| 4 | 13 |
| 16 | 8 |
| 8 | 5 |
| 0 | 15 |
| 0 | 8 |
| 6 | 10 |
| 4 | 8 |
| 0 | 8 |
| 0 | 12 |
| 2 | 8 |
| 6 | 13 |
| 12 | 11 |
| 8 | 14 |
| 0 | 0 |
| 21 | 18 |
| 2 | 8 |
| 1 | 12 |
| 4 | 9 |
| 0 | 0 |
| 13 | 17 |
| 2 | 10 |
| 8 | 11 |
| 10 | 10 |
| 0 | 0 |
| 15 | 9 |
| 4 | 14 |
| 2 | 11 |
| 2 | 14 |
| 2 | 10 |
| 6 | 12 |
| 75 | 9 |
| 22 | 9 |
| 15 | 8 |
| 8 | 10 |
| 30 | 8 |
| 19 | 10 |
| 1 | 12 |
| 4 | 10 |
| 4 | 11 |
| 2 | 11 |
| 5 | 19 |
| 6 | 12 |
| 6 | 14 |
| 9 | 15 |
| 11 | 11 |
| 0 | 15 |
| 12 | 13 |
| 6 | 18 |
| 8 | 14 |
| 4 | 11 |
| 0 | 0 |
| 10 | 8 |
| 0 | 14 |
| 5 | 16 |
| 14 | 11 |
| 0 | 10 |
| 0 | 14 |
| 0 | 18 |
| 20 | 13 |
| 8 | 12 |
| 0 | 18 |
| 38 | 8 |
| 0 | 12 |
| 18 | 10 |
| 0 | 0 |
| 20 | 13 |
| 3 | 11 |
| 22 | 11 |
| 14 | 13 |
| 40 | 11 |
| 0 | 0 |
| 9 | 9 |
| 0 | 10 |
| 2 | 11 |
| 23 | 13 |
| 12 | 9 |
| 3 | 11 |
| 1 | 15 |
| 0 | 15 |
| 3 | 11 |
| 3 | 16 |
| 8 | 10 |
| 7 | 9 |
| 4 | 14 |
| 2 | 8 |
| 7 | 14 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 16 | 15 |
| 12 | 13 |
| 0 | 0 |
| 7 | 17 |
| 4 | 10 |
| 4 | 11 |
| 0 | 0 |
| 11 | 15 |
| 0 | 0 |
| 4 | 10 |
| 7 | 14 |
| 9 | 16 |
| 0 | 9 |
| 0 | 15 |
| 10 | 13 |
| 8 | 8 |
| 2 | 13 |
| 7 | 8 |
| 4 | 8 |
| 4 | 11 |
| 0 | 9 |
| 4 | 13 |
| 2 | 11 |
| 4 | 10 |
| 0 | 16 |
| 0 | 13 |
| 2 | 12 |
| 0 | 10 |
| 0 | 15 |
| 0 | 12 |
| 4 | 10 |
| 0 | 13 |
| 0 | 0 |
| 0 | 10 |
| 10 | 11 |
| 4 | 9 |
| 3 | 12 |
| 8 | 11 |
| 14 | 5 |
| 0 | 19 |
| 2 | 10 |
| 4 | 15 |
| 4 | 10 |
| 0 | 15 |
| 17 | 10 |
| 4 | 14 |
| 5 | 7 |
| 2 | 10 |
| 0 | 0 |
| 14 | 5 |
| 2 | 10 |
| 7 | 6 |
| 0 | 0 |
| 0 | 8 |
| 0 | 0 |
| 11 | 9 |
| 3 | 16 |
| 3 | 7 |
| 0 | 10 |
| 5 | 9 |