**Classifiers**

**Report**

Laboratory #7

DATA CLUSTERING II

**Subsection:**

Ankit Rathi

**Date of the exercise:** 14.06.2019

**Date of the report submission:** 28.06.2019

**I. AIM**

The laboratory aimed at investigation of different clustering procedures, including hierarchical and Kohen clustering, as well as the checking the impact on the clustering performance of the different parameters (distance, linkage and parametrical for Kohonen’s).

**II. HIERARCHICAL CLUSTERING**

The listed below parameters were checked during the procedure of hierarchical clustering:

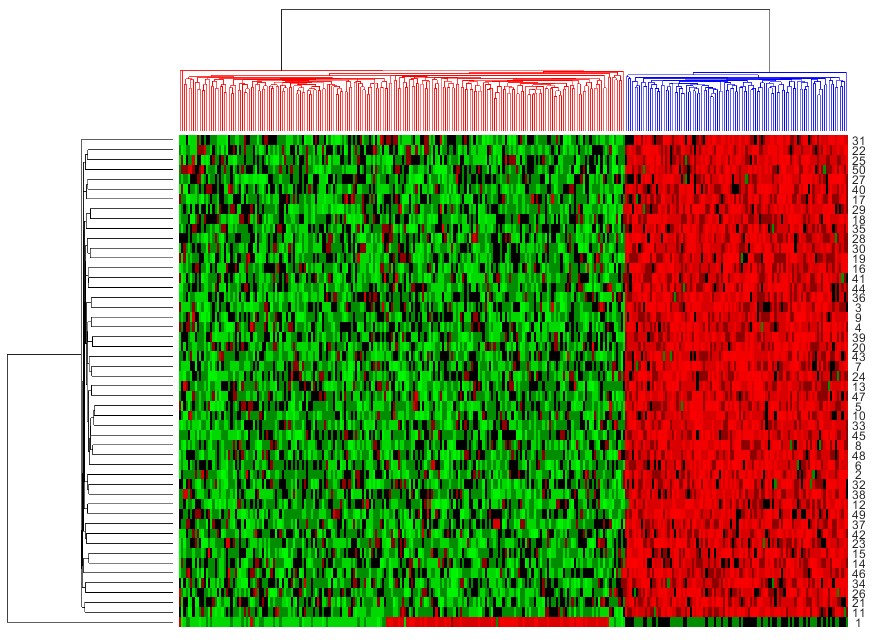
a) agglomeration methods: Average, Complete, Single;

b) measuring distance methods: Euclidean, Spearman.

The analysis also included the comparison of the results, based on the heatmap and the cluster representation. For checking the number of clusters and their sizes, the Cutoff criteria was introduced, where only clusters with at least 30 items (defined in the Matlab function as ‘Distance’ parameter) were taken to the comparison.

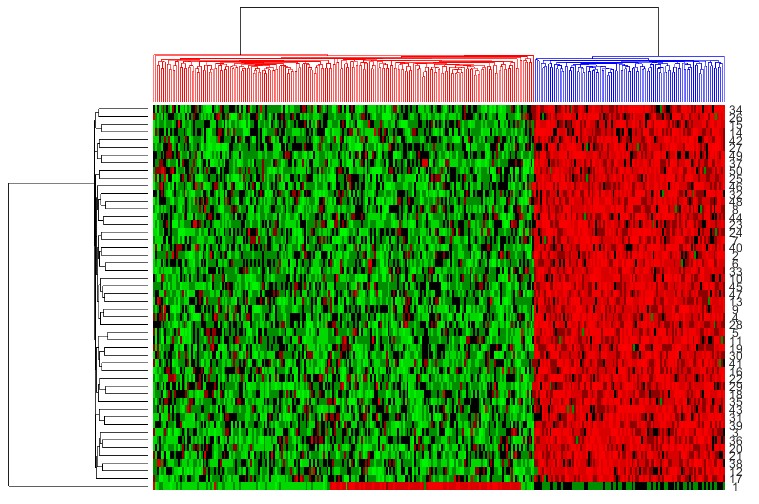
**HEATMAP ANALYSIS**

1. Average agglomeration method; Euclidean distance



The most observable clusters were colored (red – 200 samples and blue – 100 samples). Furthermore, the smaller cluster includes only the dataset coming from the second group, so the results of clustering are satisfying.

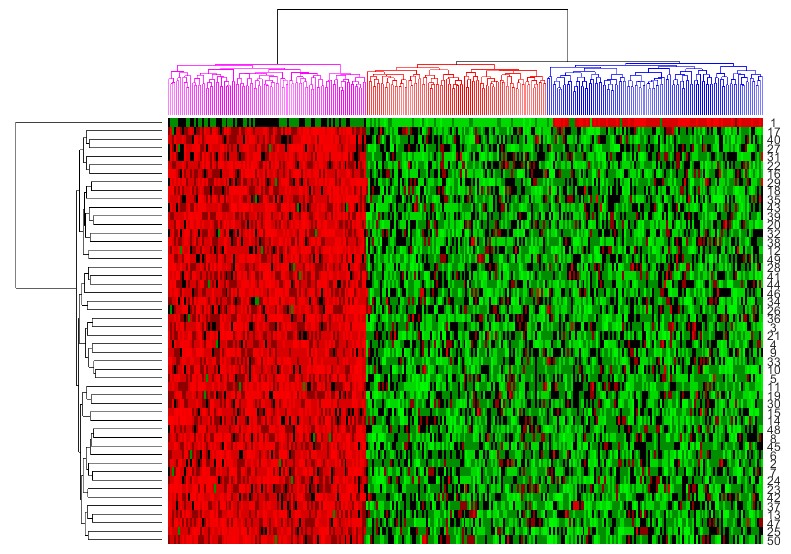
2. Average agglomeration method; Spearman distance



The differences between Spearman and Euclidean distance were not seen during the implementation of average agglomeration method. As well it is dividing the dataset into the 2 clusters (red with the third and first group containing app. 200 samples and the blue with the rest

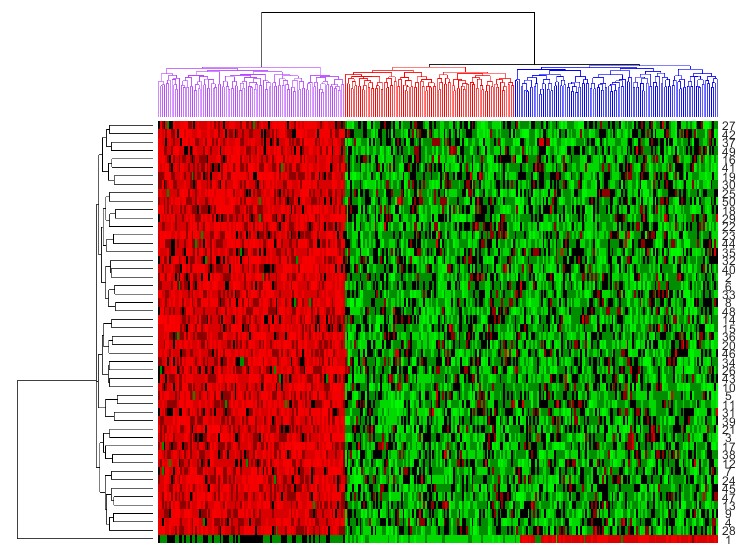
100 samples).

3. Complete agglomeration method; Spearman distance



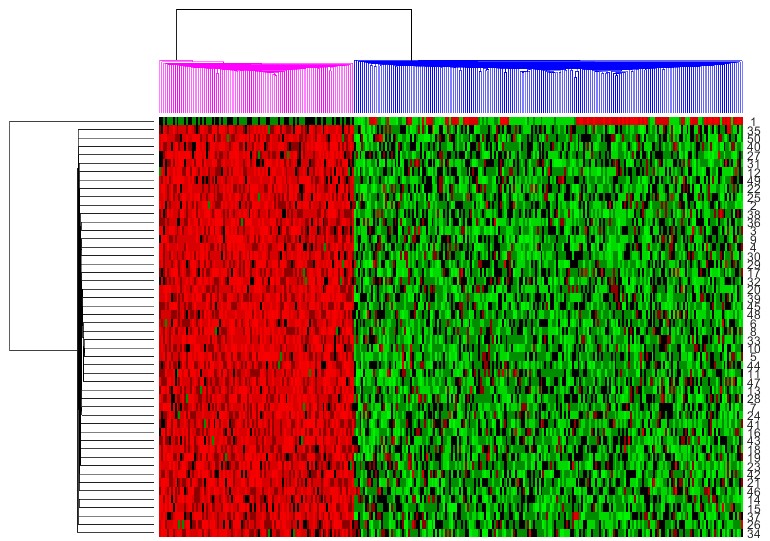
For the complete agglomeration method it was observed that the dataset was divided into 3 clusters. However, they do not correspond with the groups provided during the creating of artificial dataset. Only the whole second group was classified properly, where the first group has 9 samples more that it was expected (91 samples for the first group – colored red and 109 samples for the third group – colored blue). It was seen that the dataset was divided into 3 clusters just during the first steps of hierarchical clustering.

4. Complete agglomeration method; Spearman distance



The differences between Euclidean and Sperman distance calculation were not observed also for the complete agglomeration method. It was divided in the same way as for the complete agglomeration with Euclidean distance scenario. However, the differences in groups are observed, where the clusters are bigger than for Euclidean. This hypothesis could be proved during the comparison implementation

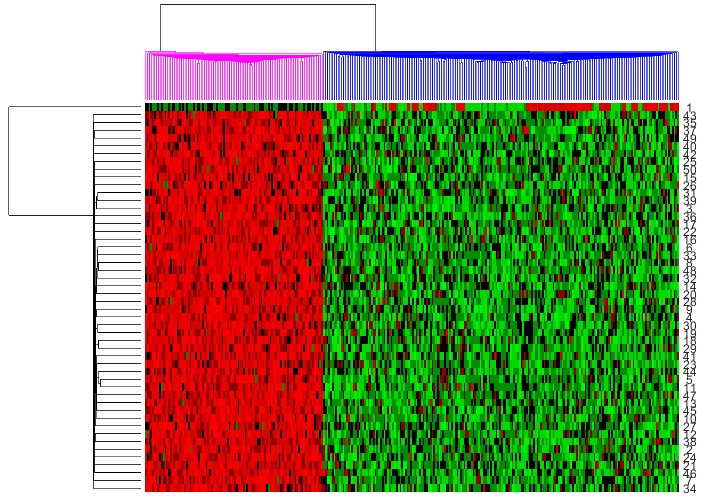
5. Single agglomeration method; Spearman distance



The dataset was divided into 2 observable clusters, as well as for the average agglomeration method. However, it was observed that the clusters are very small – bigger clusters are not

observable for this scenario. Only first appropriate step of clustering was introduced here, which is why it is called single agglomeration method.

6. Single agglomeration method; Spearman distance



The differences between Euclidean and Spearman distance measurement are not observable on the heatmap. However, the next-taken procedures could describe how this method is working during the clustering scenario.

*Tab. 1. Overall results from the cluster analysis (the clusters which were shown by eye and were the most recognized).*

**Agglomeration method**

**Measuring distance method**

**Number of the clusters**

**Representation of the cluster**

Euclidean 2 100

Average

Spearman 2

200

Complete

Euclidean 3 100

91

Spearman 3

109

Euclidean 2 100

Single

Spearman 2

200

**COMPARISON ANALYSIS**

The analysis also included the comparison of the results, based on the heatmap and the cluster representation. For checking the number of clusters and their sizes, the Cutoff criteria was introduced, where only clusters with at least 30 items (defined in the Matlab function as ‘Distance’ parameter) were taken to the comparison.

*Tab. 2. The comparison analysis of the hierarchical clustering results based on the cutoff criteria introduced before – at least 30 items in the cluster.*

**Agglomeration method**

**Measuring distance method**

**Number of the clusters Representation of the cluster**

100

Average

Euclidean 3

100

100

Spearman 1 300

70

30

Complete

Euclidean 5

100

30

70

Single

Spearman 1 300

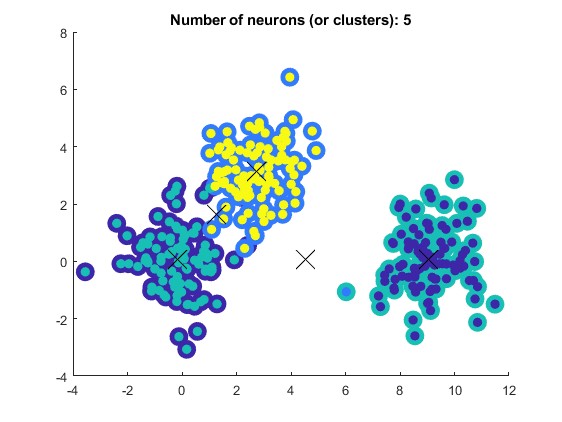
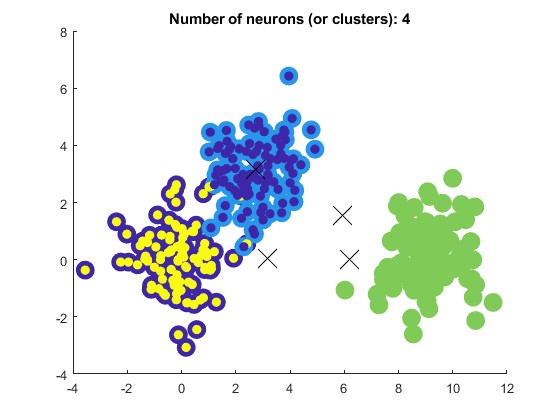
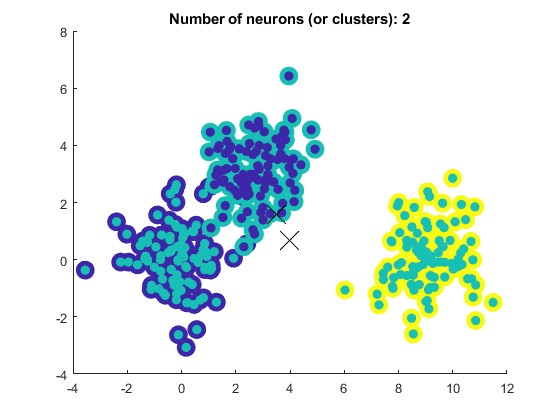
Euclidean 2 100

200

Spearman 1 300

The dataset is divided in more than 1 cluster for Euclidean method. For the Spearman method the criterion introduced during cut-off was too restrictive. This is the prove that during the Spearman method implementation, it is dividing the clusters into much smaller than it was expected previously. So a lot of dividing into clusters was performed for this method. To understand it better, the Spearman formula should be introduced. For the Spearman method (also called nonparametric because of the lack of analysis on dataset) the ranks correlation between observations is calculated. So it is not based on the real numerical dataset, only on ranks treating the dataset as the unstructured sequence of the values. This is the reason why the dataset is divided into smaller and smaller clusters during the hierarchical method application.

**III. KOHONEN’S SELF ORANIZING MAP CLUSTERING**



The listed below parameters were checked during the procedure of Kohonen’s clustering:

a) number of neurons (neurons on the 2nd last default layer): 2, 3, 4, 5

*default number of neurons: 3*

b) topology (hextop, gridtop, randtop)

*default topology: hextop*

c) number of epochs: 100, 200, 500, 1000

*default number of epochs: 100*

d) distance function (Euclidean, Box)

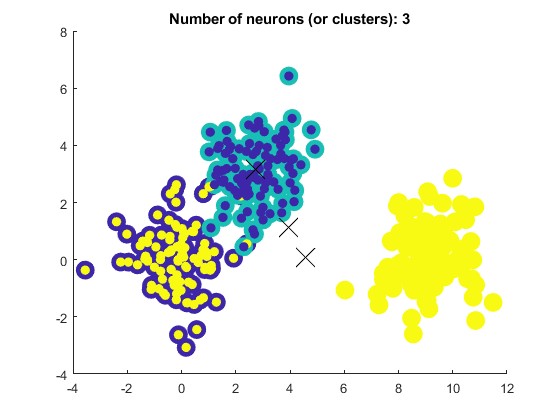
*default distance: Euclidean*

e) layer dimension ([1,3] [1,2,3], [1,2,2,3], [1,2,2,2,3])

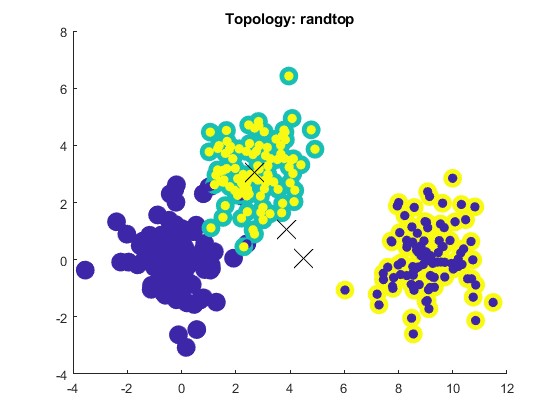
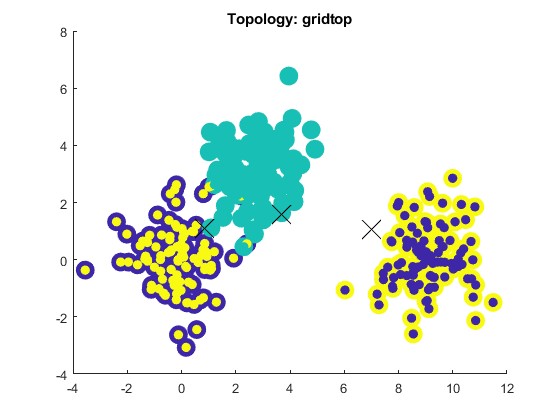
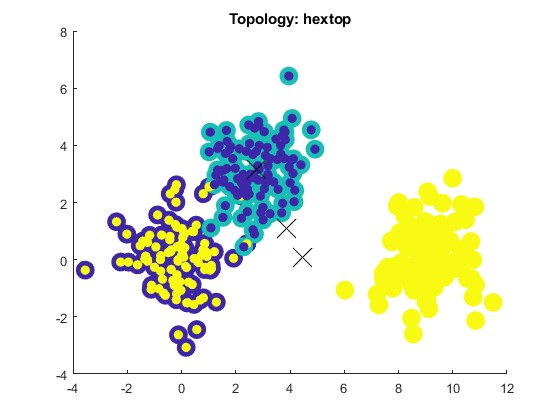
*default layer: [1, 3]*

The results of the Kohonen’s clustering were presented on the scatterplots, including the centroids of newly created clusters (crosses), the information about clustering from prior knowledge (bigger points) and the cluster representation (smaller points). It is important to mention that the colors do not take the current cluster (so the colors could be different than for the prior knowledge decision).

**NUMBER OF NEURONS**

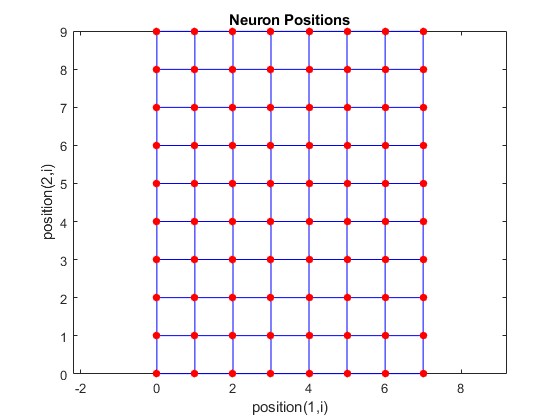
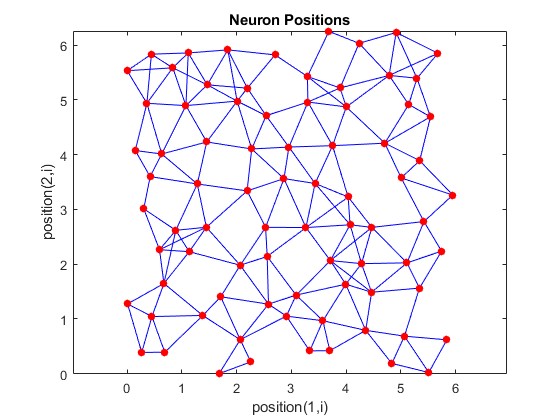
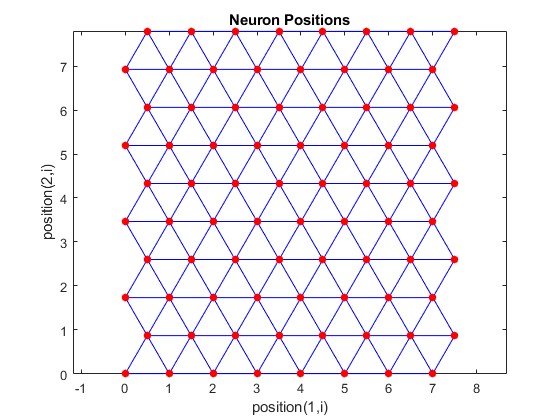


It was observed that number of neurons for the Kohonen’s classifier application sometimes does not include additional neuron weights as consecutive clusters. In the case of computational complexity and performance analysis, the best result should be obtained for using the neurons the same as number of clusters. However, if the number of neurons is smaller than number of clusters, the Kohonen’s classifier has a problem to properly division of the dataset because of the lack of information about the clusters.



**TOPOLOGY**

The topology used during clustering do not affect on the proper clustering (for all of the above-mentioned classes, it was clustered properly). However the position of centroids is different for the given topologies, which is the result of the performance and algorithm of these method, which takes the position of the neurons.

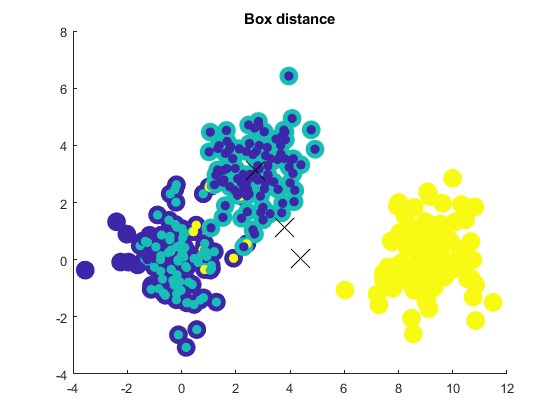
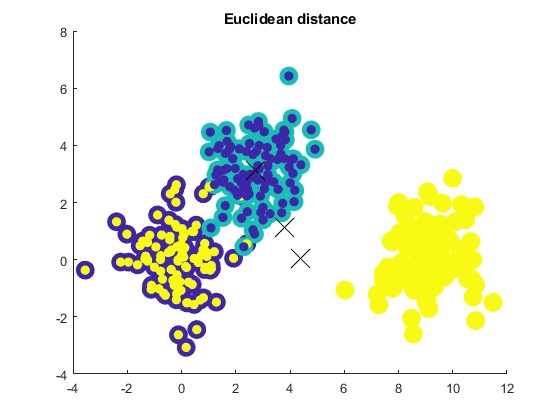
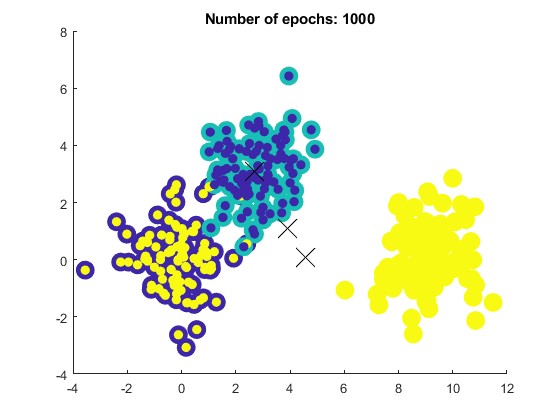
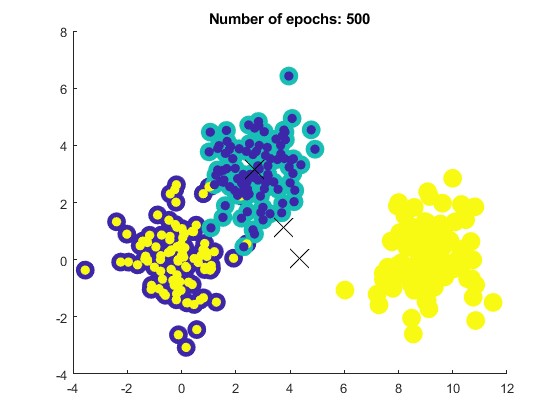
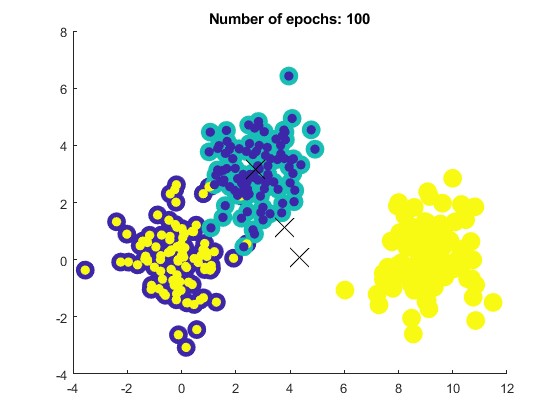


gridtop hextop randtop

*Pictures of the neuron positions for the topologies were taken from*

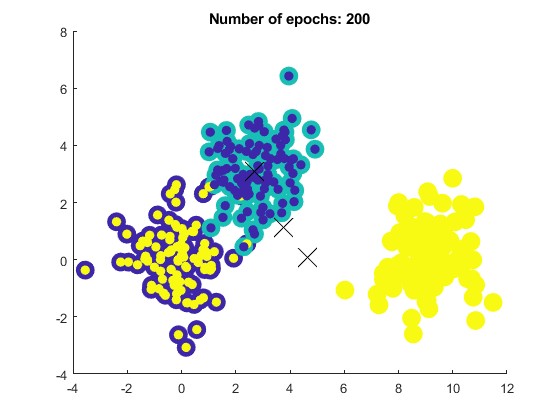
[*https://www.mathworks.com/help/deeplearning/ug/cluster-with-self-organizing-map-neural-network.html#bss4b\_l-2.*](https://www.mathworks.com/help/deeplearning/ug/cluster-with-self-organizing-map-neural-network.html#bss4b_l-2)

**DISTANCE FUNCTION**

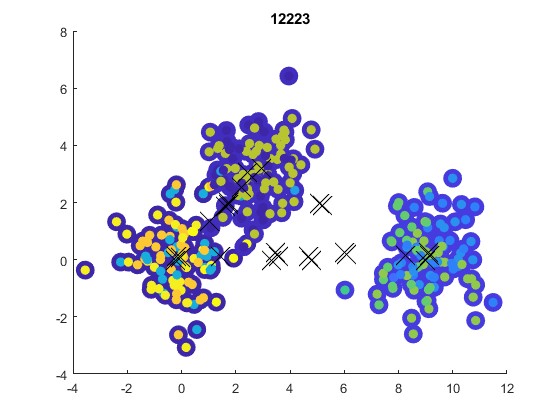
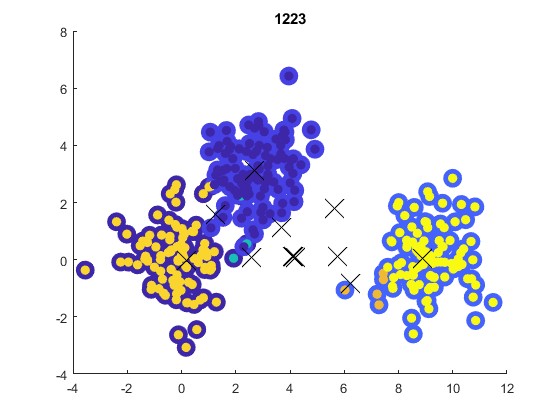
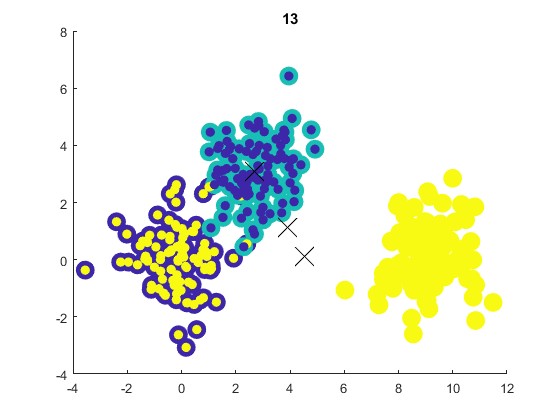


For the distance function used in Kohonen’s algorithm, the big impact on the clustering was observed. For Box distance application, some of the points were not classified properly. It is because of the application procedure of the following distance calculation. Furthermore, it should depend on the topology used during clustering, for example for gridtop, the appropriate method is previously- mentioned boxdist, when for other topologies the Euclidean methods should be introduced to find the distances between the layer’s neurons. This is the reason why some of the points were not clustered in a proper way (default the hextop topology was used, correlated with Euclidean function).

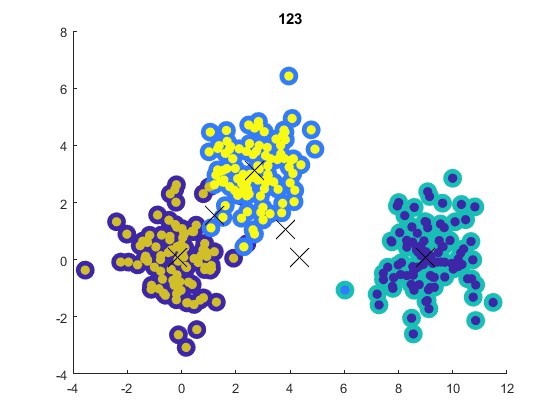
**TRAIN EPOCH NUMBER**



There are no differences between number of epochs used to implement the Kohonen’s algorithm. The dataset was no so complex and the calculations will be defined in the smaller number of epochs than it was expected. The number of epochs has a big influence on the time and complexity of calculations and it should be used in a proper way.



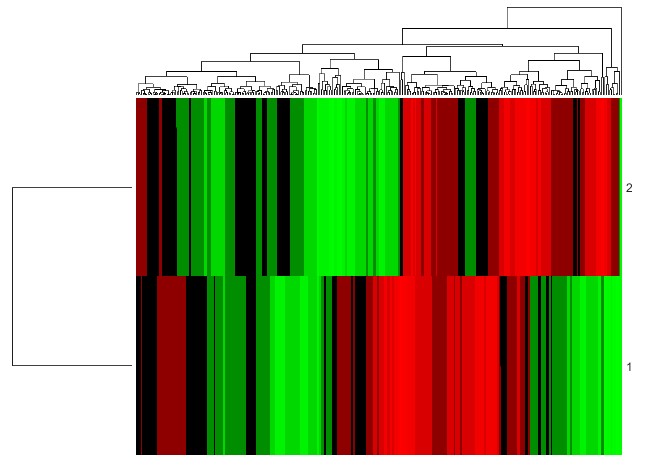
**LAYER DIMENSION**



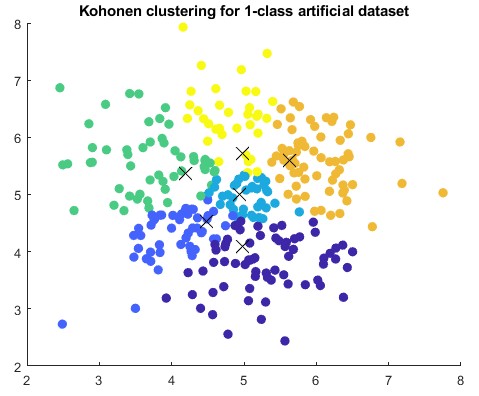
The size of the layer has a strong impact on the number of detected clusters in the dataset. We can meet here with overlearning procedure, which is common problem for machine learning algorithms, whereas the parameters of the algorithm was not properly chosen taking into account the complexity of the dataset and the possibility of the multiple features detection.

**IV. PROOF THE CLUSTERING FUNCTIONALITY FOR ARTIFICAL DATASET FROM LABOLATORY 6**

To proof the clustering functionality of the previously generated artificial datasets with 1, 2 and 3 classes, the hierarchical clustering with Kohonen’s algorithm were used during this scenario. It includes the default parameters for the topology, number of neurons, epochs and layers, as it was followed from III.

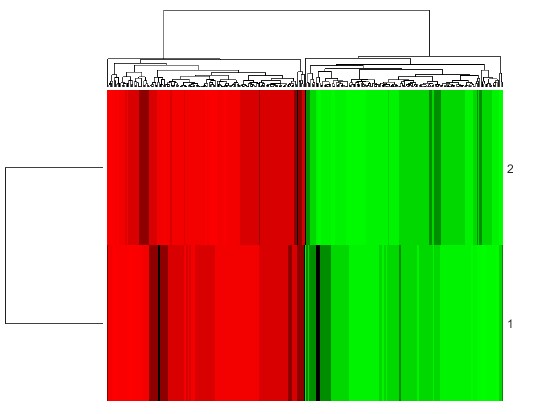


 **Kohonen’s clustering (for [1 2 3] layer)**

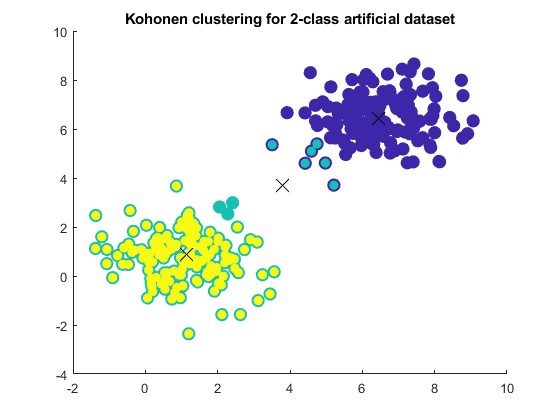


 **Comment**

As it was showed in the last laboratory report, the unstructured data with 1 class could be divided randomly into a lot of clusters, but none of it is related to the actual representation of the dataset. So hierarchical clustering and Kohonen’s clustering procedures are also sensitive to have been overrepresented.

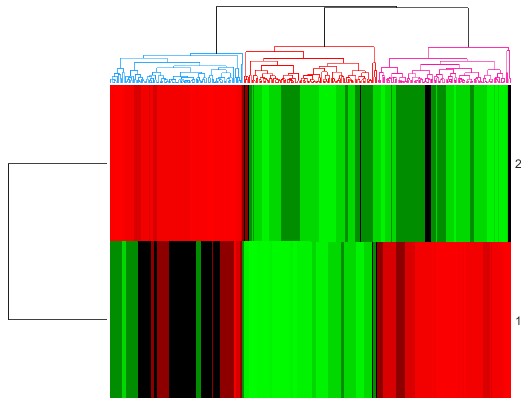


 **Kohonen’s clustering (traditionally for [1 3] layer)**

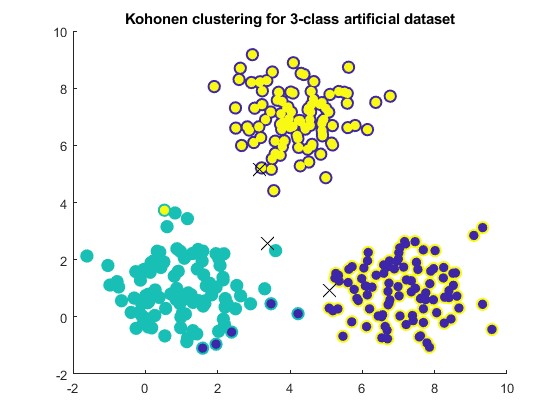


 **Comment**

The hierarchical clustering properly clustered the dataset into two classes, as well as the Kohonen’s algorithm. However, some points which were positioned between two clustered, was described as the third cluster, which is the result of overlearning predefined of the layer (the second layer has 3 neurons, so one of them was additionally activated). Only the combination of hierarchical and Kohonen’s clustering could give the appropriate results.



 **Kohonen’s clustering (traditionally for [1 3] layer)**



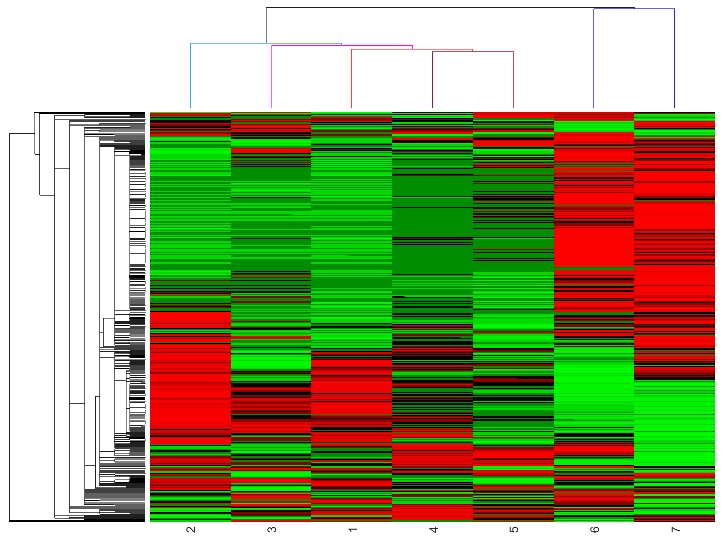
 **Comment**

The hierarchical clustering properly divided the dataset into 3 classes (groups or clusters). However, the Kohonen’s clustering has some difficulties for proper clusterization of the dataset pre-defined classes.

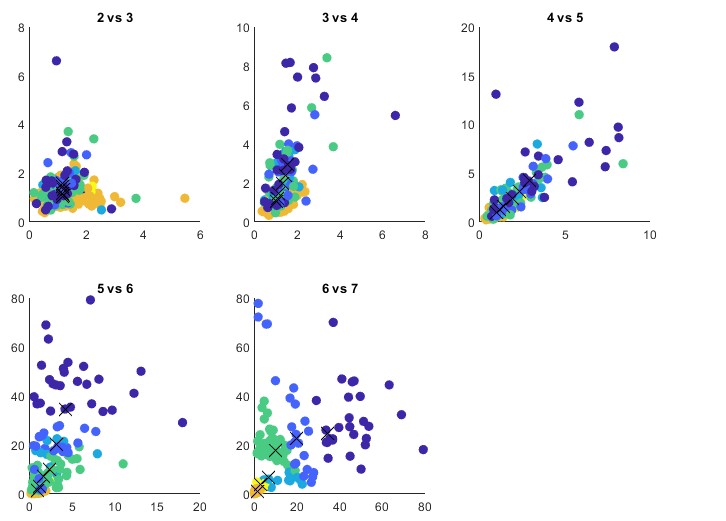
So it was proved that the datasets has the clustering functionality.

**V. PROOF THE CLUSTERING FUNCTIONALITY FOR YEAST DATASET FROM LABOLATORY 6**

 **Hierarchical clustering**



 **Kohonen’s clustering (for layer [1 2 3])**



 **Comment**

The hierarchical clustering proved that some of the features could be combined, as well as the Kohonen’s clustering. The most different features, which are 6 and 7, are clustered properly during the first approach (hierarchical clustering), as well as it was seen for the Kohonen’s clustering in the form of boxplot 1 vs 1 (for sixth vs seventh feature). So the clustering functionality of this dataset was proved.

**VI. FINAL CONCLUSIONS**

 The hierarchical clustering is the best method for defining the number of clusters in the dataset.

 The Spearman and Euclidean distances for hierarchical clustering gives the similar results for the heatmap, however in fact they contribute with different mathematical method (working on real data and working on the ranks of data). However, for the Kohonen’s algorithm the distances have the major role on the performance.

 The clustering functionality was proved for artificial and real dataset by presenting the results of hierarchical clustering and scatterplots coming from Kohonen’s algorithm interpretation.

 The size of the layer has a strong impact on the number of detected clusters in the dataset. We can meet here with overlearning procedure, which is common problem for machine learning algorithms, whereas the parameters of the algorithm was not properly chosen taking into account the complexity of the dataset and the possibility of the multiple features detection.

 There are no differences between number of epochs used to implement the Kohonen’s algorithm. The dataset was no so complex and the calculations will be defined in the smaller number of epochs than it was expected. The number of epochs has a big influence on the time and complexity of calculations and it should be used in a proper way.

 It was observed that number of neurons for the Kohonen’s classifier application sometimes does not include additional neuron weights as consecutive clusters. In the case of computational complexity and performance analysis, the best result should be obtained for using the neurons the same as number of clusters. However, if the number of neurons is smaller than number of clusters, the Kohonen’s classifier has a problem to properly division of the dataset because of the lack of information about the clusters.

**VII. CODE LISTING**

%% GENERATE TEST DATASET (WITH THE SAME PARAMETRES AS IN D.C.I) AND LOAD THE YEAST DATASET ALSO IN D.C.II R1 = mvnrnd([4 7],[1 0; 0 1],100);

R2 = mvnrnd([1 1],[1 0; 0 1],100); R3 = mvnrnd([7 1],[1 0; 0 1],100); test\_dataset\_3 = [R1; R2; R3];

test\_dataset\_3\_labels = [ones(1,100), ones(1,100)\*2, ones(1,100)\*3];

R1 = mvnrnd([6.5 6.5],[1 0; 0 1],150); R2 = mvnrnd([1 1],[1 0; 0 1],150); test\_dataset\_2 = [R1; R2];

test\_dataset\_2\_labels = [ones(1,150), ones(1,150)\*2];

test\_dataset\_1 = mvnrnd([5 5],[1 0; 0 1],300);

yeast\_dataset = load('yeast.mat');

yeast\_dataset = yeast\_dataset.drozdze;

%% GENERATE MUTLICLASS SYNTHETIC DATASET

d1 = randn(50,100);

d2 = randn(50,100)+3;

d3 = randn(50,100); d3(1,:) = d3(1,:) + 9; d = [d1 d2 d3];

cLabel = [ones(1,100), ones(1,100)\*2, ones(1,100)\*3];

%% HIERARCHICAL CLUSTERING:

methods = {'average','complete','single'};

ways = {'euclidean','spearman'};

result = zeros(40,6);

k=1;

for i=1:length(methods)

for j=1:length(ways)

% Present in the form of histogram

gcp = clustergram(d,'Standardize', 'Row');

set(gcp,'Linkage',methods{i},'RowPdist',ways{j}); % the dendogram was setted to 3 as the number of clusters inside

Y = pdist(d', ways{j}); Y = squareform(Y);

Z = linkage(Y,methods{i});

T = cluster(Z,'cutoff',50,'Criterion','distance'); % cutoff criterion: at least 30 items in the cluster

% save the number of items and clusters in the table for z = 1:length(unique(T))

end

result(z,k) = length(find(T==z));

end

end

k = k+1;

%% KOHONEN CLUSTERING:

% default parametres:

% no\_neurons = 3

% topology = hextop

% epochs = 100

% layer = [1 3]

% Euclidean distance

% the proposed parametres checking during the excercise no\_neurons = {[1 2], [1 3], [1 4], [1 5]};

topology = {'hextop','gridtop','randtop'};

epochs = [100, 200, 500, 1000];

layer = {[1 3], [1 2 3], [1 2 2 3], [1 2 2 2 3]};

% NUMBER OF NEURONS CHECKING

for i=1:length(no\_neurons)

layer\_first = no\_neurons{i};

neuron\_first = layer\_first(length(layer\_first)); net = selforgmap(no\_neurons{i},100,3, 'hextop'); net.trainParam.epochs = 100;

net = train(net, d);

distances = dist(d', net.IW{1}'); [min\_dist, cndx] = min(distances, [], 2);

end

figure()

scatter(d(1,:), d(2,:),200, cLabel, 'filled');

hold on

scatter(d(1,:), d(2,:),50, cndx, 'filled');

hold on

scatter(net.IW{1}(:,1), net.IW{1}(:,2),400, 'blackx')

title(sprintf('Number of neurons (or clusters): %d', neuron\_first))

% LAYER SIZE CHECKING

for i=1:length(layer)

net = selforgmap(layer{i},100,3, 'hextop');

net.trainParam.epochs = 100;

net = train(net, d);

distances = dist(d', net.IW{1}'); [min\_dist, cndx] = min(distances, [], 2);

end

figure()

scatter(d(1,:), d(2,:),200, cLabel, 'filled');

hold on

scatter(d(1,:), d(2,:),50, cndx, 'filled');

hold on

scatter(net.IW{1}(:,1), net.IW{1}(:,2),400, 'blackx')

title(sprintf('%i',layer{i}))

% DISTANCE FUNCTION

net = selforgmap([1 3],100,3, 'hextop');

net.trainParam.epochs = 100;

net = train(net, d);

distances = dist(d', net.IW{1}'); [min\_dist, cndx] = min(distances, [], 2);

figure()

scatter(d(1,:), d(2,:),200, cLabel, 'filled');

hold on

scatter(d(1,:), d(2,:),50, cndx, 'filled');

hold on

scatter(net.IW{1}(:,1), net.IW{1}(:,2),400, 'blackx')

title('Euclidean distance')

distancesbox = boxdist(d', net.IW{1}'); [min\_distbox, cndxbox] = min(distancesbox, [], 2);

figure()

scatter(d(1,:), d(2,:),200, cLabel, 'filled');

hold on

scatter(d(1,:), d(2,:),50, cndxbox, 'filled');

hold on

scatter(net.IW{1}(:,1), net.IW{1}(:,2),400, 'blackx')

title('Box distance')

% TRAIN EPOCH NUMBERS

for i=1:length(epochs)

net = selforgmap([1 3],100,3, 'hextop');

net.trainParam.epochs = epochs(i);

net = train(net, d);

distances = dist(d', net.IW{1}');

[min\_dist, cndx] = min(distances, [], 2);

end

figure()

scatter(d(1,:), d(2,:),200, cLabel, 'filled');

hold on

scatter(d(1,:), d(2,:),50, cndx, 'filled');

hold on

scatter(net.IW{1}(:,1), net.IW{1}(:,2),400, 'blackx')

title(sprintf('Number of epochs: %d',epochs(i)))

% TOPOLOGY

for i=1:length(topology)

net = selforgmap([1 3],100,3, topology{i});

net.trainParam.epochs = 100;

net = train(net, d);

distances = dist(d', net.IW{1}'); [min\_dist, cndx] = min(distances, [], 2);

end

figure()

scatter(d(1,:), d(2,:),200, cLabel, 'filled');

hold on

scatter(d(1,:), d(2,:),50, cndx, 'filled');

hold on

scatter(net.IW{1}(:,1), net.IW{1}(:,2),400, 'blackx')

title(sprintf('Topology: %s',topology{i}))

%% PROVE THE CLUSTERITY FUNCTIONALITY FOR ARTIFICAL DATASET

% for 1-class dataset

net\_1class = selforgmap([1 2 3],100, 3, 'hextop');

net\_1class.trainParam.epochs = 100;

net\_1class = train(net\_1class, test\_dataset\_1');

test1\_distances = dist(test\_dataset\_1, net\_1class.IW{1}'); [min\_dist, cndx] = min(test1\_distances, [], 2);

figure()

scatter(test\_dataset\_1(:,1), test\_dataset\_1(:,2),50, cndx, 'filled');

hold on

scatter(net\_1class.IW{1}(:,1), net\_1class.IW{1}(:,2),200, 'blackx')

title('Kohonen clustering for 1-class artificial dataset')

gcp = clustergram(test\_dataset\_1','Standardize', 'Row');

set(gcp,'Linkage','average','RowPdist','euclidean'); % the dendogram was setted to 3 as the number of clusters inside

% for 2-class dataset

net\_1class = selforgmap([1 3],100, 3, 'hextop');

net\_1class.trainParam.epochs = 100;

net\_1class = train(net\_1class, test\_dataset\_2');

test1\_distances = dist(test\_dataset\_2, net\_1class.IW{1}'); [min\_dist, cndx] = min(test1\_distances, [], 2);

figure()

scatter(test\_dataset\_2(:,1), test\_dataset\_2(:,2),100, test\_dataset\_2\_labels, 'filled');

hold on

scatter(test\_dataset\_2(:,1), test\_dataset\_2(:,2),50, cndx, 'filled');

hold on

scatter(net\_1class.IW{1}(:,1), net\_1class.IW{1}(:,2),200, 'blackx')

title('Kohonen clustering for 2-class artificial dataset')

gcp = clustergram(test\_dataset\_2','Standardize', 'Row');

set(gcp,'Linkage','average','RowPdist','euclidean'); % the dendogram was setted to 3 as the number of clusters inside

% for 3-class dataset

net\_1class = selforgmap([1 3],100, 3, 'hextop');

net\_1class.trainParam.epochs = 100;

net\_1class = train(net\_1class, test\_dataset\_3');

test1\_distances = dist(test\_dataset\_3, net\_1class.IW{1}'); [min\_dist, cndx] = min(test1\_distances, [], 2);

figure()

scatter(test\_dataset\_3(:,1), test\_dataset\_3(:,2),100, test\_dataset\_3\_labels, 'filled');

hold on

scatter(test\_dataset\_3(:,1), test\_dataset\_3(:,2),50, cndx, 'filled');

hold on

scatter(net\_1class.IW{1}(:,1), net\_1class.IW{1}(:,2),200, 'blackx')

title('Kohonen clustering for 3-class artificial dataset')

gcp = clustergram(test\_dataset\_3','Standardize', 'Row');

set(gcp,'Linkage','average','RowPdist','euclidean'); % the dendogram was setted to 3 as the number of clusters inside

%% PROVE THE CLUSTERITY FUNCTIONALITY FOR YEAST DATASET net\_yeast = selforgmap([1 2 3],100, 3, 'hextop'); net\_yeast.trainParam.epochs = 100;

net\_yeast = train(net\_yeast, yeast\_dataset');

yeast\_distances = dist(yeast\_dataset, net\_yeast.IW{1}'); [min\_dist, cndx] = min(yeast\_distances, [], 2);

figure()

% 1 vs 1 procedure for Kohonen clustering subplot(2,3,1)

scatter(yeast\_dataset(:,2), yeast\_dataset(:,3),50, cndx, 'filled');

hold on

scatter(net\_yeast.IW{1}(:,2), net\_yeast.IW{1}(:,3),200, 'blackx')

title('2 vs 3')

subplot(2,3,2)

scatter(yeast\_dataset(:,3), yeast\_dataset(:,4),50, cndx, 'filled');

hold on

scatter(net\_yeast.IW{1}(:,3), net\_yeast.IW{1}(:,4),200, 'blackx')

title('3 vs 4')

subplot(2,3,3)

scatter(yeast\_dataset(:,4), yeast\_dataset(:,5),50, cndx, 'filled');

hold on

scatter(net\_yeast.IW{1}(:,4), net\_yeast.IW{1}(:,5),200, 'blackx')

title('4 vs 5')

subplot(2,3,4)

scatter(yeast\_dataset(:,5), yeast\_dataset(:,6),50, cndx, 'filled');

hold on

scatter(net\_yeast.IW{1}(:,5), net\_yeast.IW{1}(:,6),200, 'blackx')

title('5 vs 6')

subplot(2,3,5)

scatter(yeast\_dataset(:,6), yeast\_dataset(:,7),50, cndx, 'filled');

hold on

scatter(net\_yeast.IW{1}(:,6), net\_yeast.IW{1}(:,7),200, 'blackx')

title('6 vs 7')

% Hierarchical clustering

gcp = clustergram(yeast\_dataset','Standardize', 'Row');

set(gcp,'Linkage','average','RowPdist','euclidean'); % the dendogram was setted to 3 as the number of clusters inside