Intro to Intelligent Systems
Assignment 2

Daniël Haitink S2525119
Remco Pronk S2533081

September 21, 2015

1 Assignment 1

We've applied the K-means clustering algorithm in the following way, as seen in listing 1.

```
1  %cluster is the amount of clusters wanted; inputmatrix is a n*2
       matrix
2  function [] = kmeans(clusters, inputmatrix)
3
4  %get length of matrix
5  length_matrix = length(inputmatrix);
6
7  colors = ['m','y','c','r','g','b','k'];
8  markers = ['o', '+', '*', '.', '+', 'x'];
9  %pick random points to become mu
10 randK = randperm(length_matrix,clusters);
11 %Create a vector where the cluster gets saved to which a datapoint
       belongs to
12 clusteredData = zeros(length_matrix,1);
13
14 clusterLook = zeros(clusters, 2);
15 mark = 1;
16 color =1;
17 %Determines appearance of cluster (color and shape)
18 for i = 1:clusters
19     color = color + 1;
20     if color > length(colors)
21         mark = mark+1;
22         color = 1;
23     end
24    clusterLook(i,1) = color;
25    clusterLook(i,2) = mark;
26 end
27
28 %Saves the coordinates of the cluster centers
29 for i = 1:clusters
30     clusterK(i,:) = inputmatrix(randK(i),:);
31 end
32
```

```
33 %Create the plot
34 for i = 1:length_matrix
35       currentK = inputmatrix(i,:);
36       best = inf;
37       bestCluster = -1;
38       for j = 1:clusters
39          %Determine distance using pythagoras
40          dX =abs( currentK(1,1) - clusterK(j,1));
41          dY =abs( currentK(1,2) - clusterK(j,2));
42          currentDist = sqrt(dY*dY+dX*dX);
43          %In case a closer cluster is found, save that
44          if best > currentDist
45               best = currentDist;
46               bestCluster = j;
47          end
48       end
49       clusteredData(i) = bestCluster;
50       scatter(currentK(1,1), currentK(1,2),colors(clusterLook(
             bestCluster,1)), markers(clusterLook(bestCluster,2)));
51       hold on
52 end
53 hold off
```

Listing 1: K-means clustering algorithm

Note that our algorithm works for a high means-size.

The resulting plots can be found in figures 1, 2 and 3.

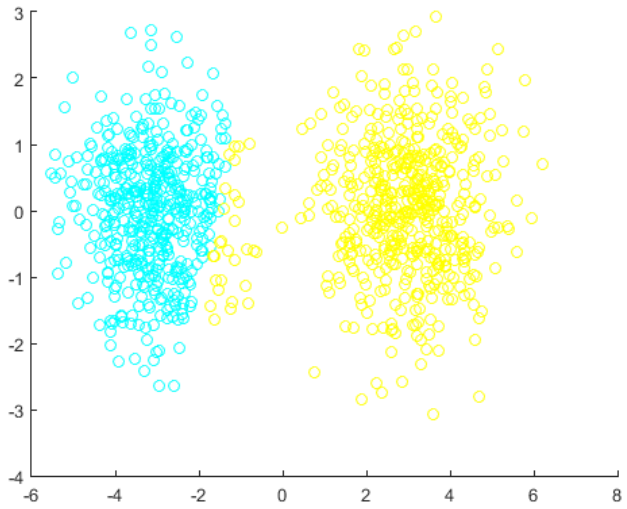

Figure 1: K-means cluster with 2 means.

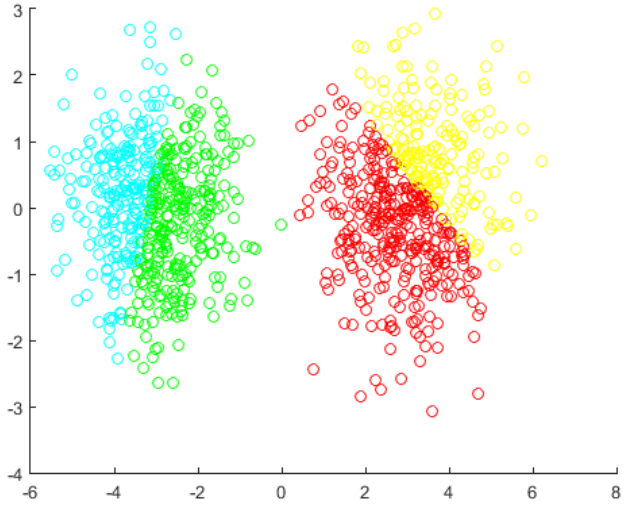

Figure 2: K-means cluster with 4 means.

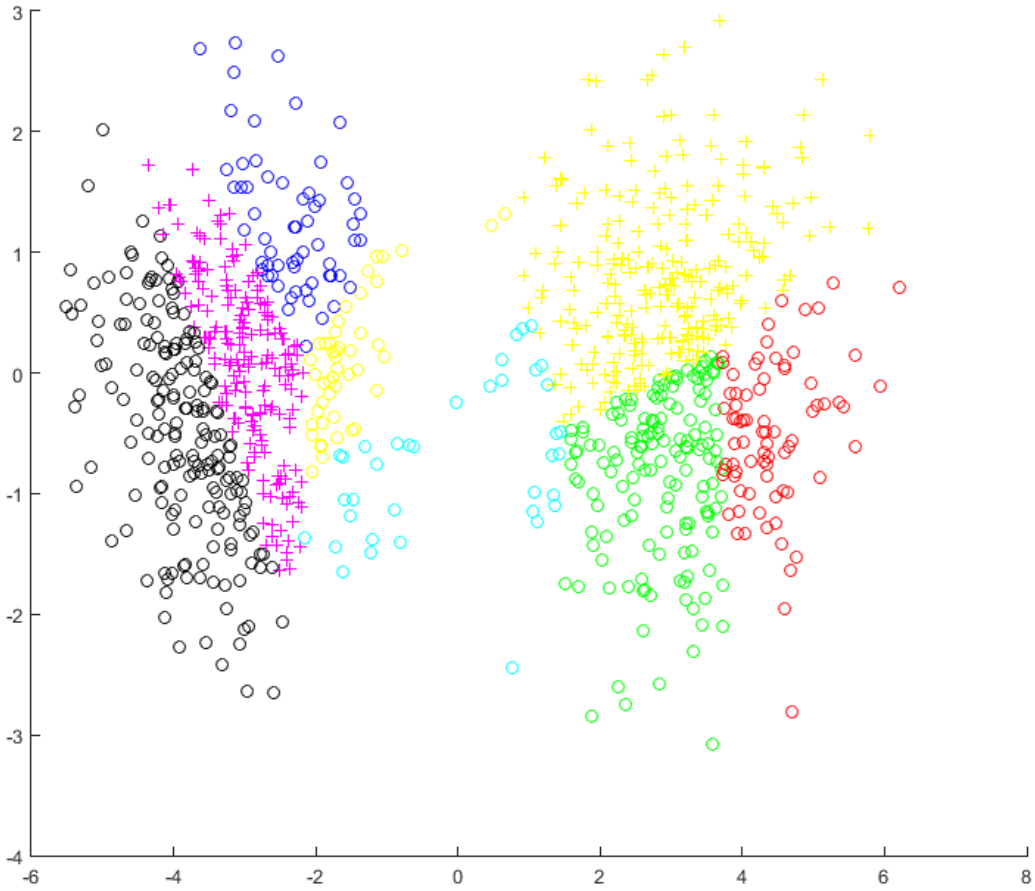

Figure 3: K-means cluster with 8 means.

The clustering with the file w6_1x.mat is different on each execution, because different points will be picked to become the centre point of a cluster each time. It would be stranger to get the same clustering every time when using k-means clustering.

Seven clusters gives mostly the same clustering each iteration with the set w6_1y.mat. The points are very densely distributed around the origin (coordinates 0,0). There also appear clear boarders between the different clusters each time.

When splitting the set w6_1z.mat into only two clusters, it is like drawing a straight line through the points, and putting each point at either side into one

Table 1: Table with characteristics of the whales

Type of whale	Fluke characteristics	Diving characteristics	Has dorsal fin?	Size	Misc. facts
Killer whale	Small, 6-8m	Fluke **not** visible Dorsal fin visible	Yes		Blows water quite often
Beluga whale		Fluke **not** visible	No		
Narwhal whale		Fluke visible	No	<5m	Has a single, long tusk
Bowhead whale		Fluke visible	No	<=20m	
Blue whale	Large, >8m	Fluke visible	Yes	>=30m	

cluster. Because such a line can be drawn at any location and direction, the clustering is not predictable. When splitting this set into four is like drawing an X shape in the plot. This is a far more static devision of the points, thus making it look like the clustering is more predictable This is all the case because all the points are nicely, almost evenly distributed.

We deem the K-means algorithm not really useful, since the means are randomly selected from all available points. It can't give you any information about the meaning your data. The only use for it, according to us, is to see how data is distributed from each other (are the points densily put together at certain points, are some points far away from other points, etc).

2 Assignment 2

We have the following classes in which the input has to be categorised:

- Killer whale
- Beluga whale
- Narhhal whale
- Bowhead whale
- Blue whale

These whales have the following characteristics, as seen in table 1.

We can see that some information is missing for some whales. This is a problem we can circumvent by changing the order of the questions of the decision tree.

We start of by asking if the whale has a dorsal fin. That will give us either 3 or 2 remaining options. If the whale does have a dorsal fin, it is either a killer whale or blue whale. One difference between these two whales is the size of the fluke. If the fluke is larger than 8 metres, it is a blue whale, else it is a killer whale. If the whale doesn't have a dorsal fin, we ask if its fluke is visible when diving. If this the fluke is not visible, the whale is a beluga whale. If the fluke is visible, it is either a narwhal or bowhead whale. We can now either ask if the whale has a single, long tusk or ask about its size. Since there is such a big difference in the size of both whale, and the tusk can be broken of or not easily

visible in the water, we choose to ask about the size as the final question. There is quite the size difference between these two whale. If the whale is smaller than 5 meters, it is a narwhal, else it is a bowhead whale. The resulting tree can be seen in figure 4.

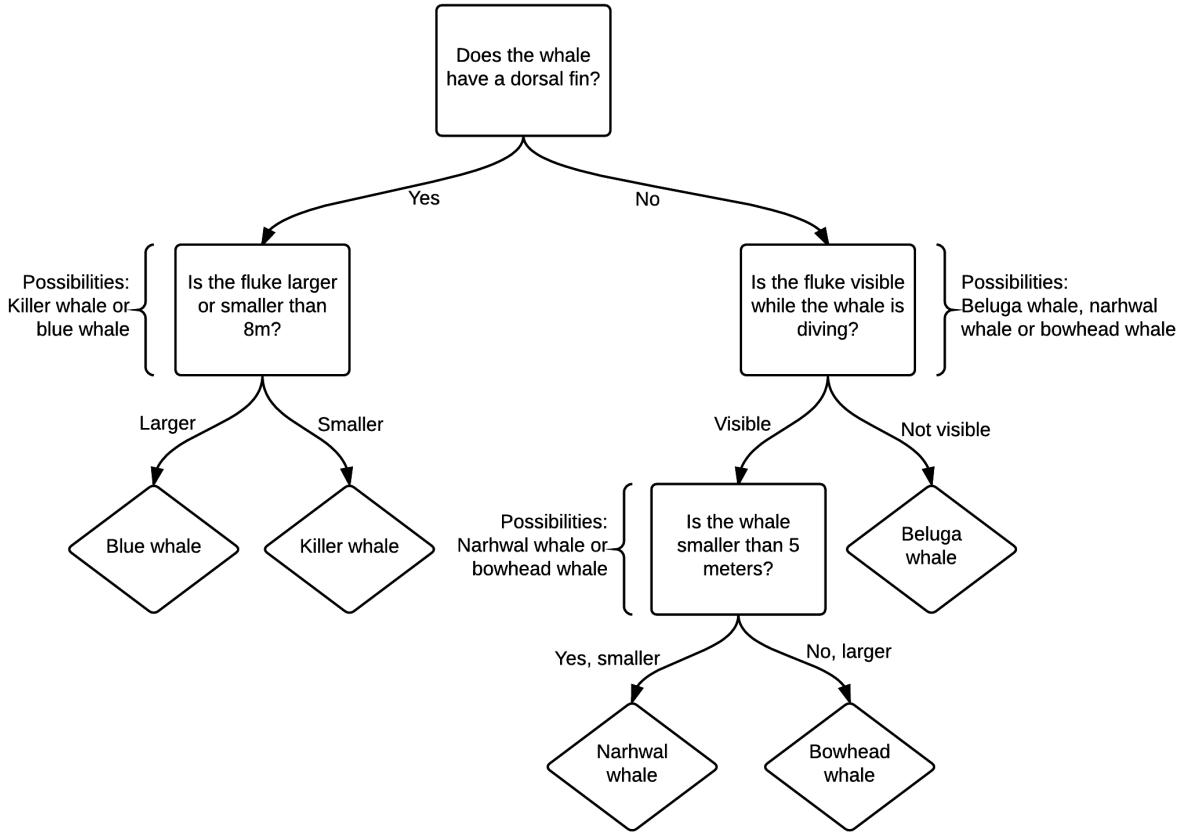

Figure 4: The whale decision tree

We choose this tree because it is not very high, with the deepest node being only on the 3rd layer. This is the minimum size the tree can have. The questions are also simple and straightforward.

3 Assignment 3

You can see the code for this assignment in listing 2. The plots for different values of K are found in figures 5, 6, 7 and 8. The plots for four clusters and

different k values are also plotted below in the figures 9, 10, 11 and 12.

In our initial code, we used the following line of code to determine which class a square on the map belonged to.

```
result(j,i) = mode(NN(:,2));
```

Where *NN* was a matrix with size k*2. Using *mode*, we find the value that occurs most often in this matrix. Because of the way the function *mode* works, in case of a draw, the lowest value of the draw will be selected as the winner.

We decided to not use such a arbitrary decider for a draw. We changed the code so the winner of the draw will be the point closest by. This seemed to be a fair way to decide the winner of the draw to us.

```
%function for doing k-nearest neighbour clustering
function [] = knn(K, nrofclasses, data)

colors = ['m','y','c','r','g','b','k'];
markers = ['o', '+', '*', '.', '+', 'x'];
N=64;

Klook = zeros(K+1, 2);
mark = 1;
color =0;
%Determines appearance of cluster (color and shape)
for i = 1:nrofclasses+1
    color = color + 1;
    if color > length(colors)
        mark = mark+1;
        color = 1;
    end
   Klook(i,1) = color;
   Klook(i,2) = mark;
end

for i=1:N
  X=(i-1/2)/N;
   for j=1:N
     Y=(j-1/2)/N;

     % create vertex with infinite numbers and calculate the
         classlength %
     NN = inf(K,2);
     classLength = length(data)/nrofclasses;
     % loop through data %
     for ii = 1:length(data)
          % calculate distance from point i to vector %
          dX =abs( X - data(ii,1));
          dY =abs( Y - data(ii,2));
          currentDist = sqrt(dY*dY+dX*dX);
          % Loop through all K Nearest Neighbours %
         for jj = 1:K
             % If current point is closer to the vector as j,
                 replace it with the current point %
             if  currentDist<NN(jj,1)
                 NN(jj,1) = currentDist;
```

```
41                  % if ii is the length of data (100), make NN(jj,2)
                        the max class number %
42                  if ii == length(data)
43                      NN(jj,2) = nrofclasses;
44                  else
45                      % else, use the calculation below to remove the
                            remainder after the comma, and increase by
                            one %
46                      NN(jj,2) = (ii/classLength)- (mod(ii/
                            classLength,1))+1;
47                  end
48                  break;
49              end
50          end
51          % Find the mode (most occuring class) of the K-nearest
                neighbours and return it %
52          result(j,i) = mode(NN(:,2));
53
54          % check if the result is really the mode, or if it is a
                draw between the clusters %
55          sum = 0;
56          for a = 1:K
57              if result(j,i) == NN(a,2)
58                  sum = sum+1;
59              end
60          end
61          % If there is only one instance of the mode, look for the
                cluster with the smallest distance to the point %
62          if sum == 1
63              for a = 1:K
64                  if min(NN(:,1)) == NN(a,1)
65                      result(j,i) = NN(a,2);
66                  end
67              end
68          end
69      end
70    end
71  end
72  imshow(result,[1 nrofclasses],'InitialMagnification','fit')
73  hold on;
74  data=N*data; % scaling
75
76  % print the point for nrofclasses %
77  for i = 1:nrofclasses
78      startPoint = (i-1)*classLength+1;
79      endPoint = i*classLength;
80     plot(data(startPoint:endPoint,1),data(startPoint:endPoint, 2), [
           colors(Klook(i,1)), markers(Klook(i,2))]);
81     hold on
82  end
83  hold off
```

Listing 2: K-nearest-neighbor classification algorithm

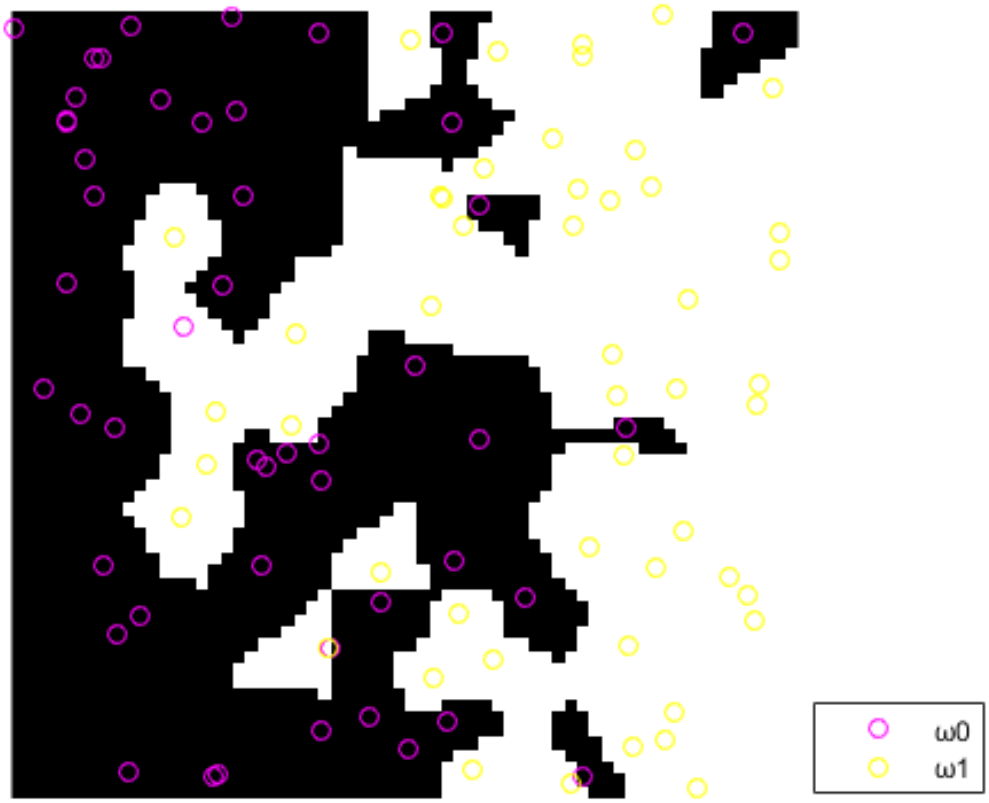

Figure 5: Plot of KNN, where k = 1

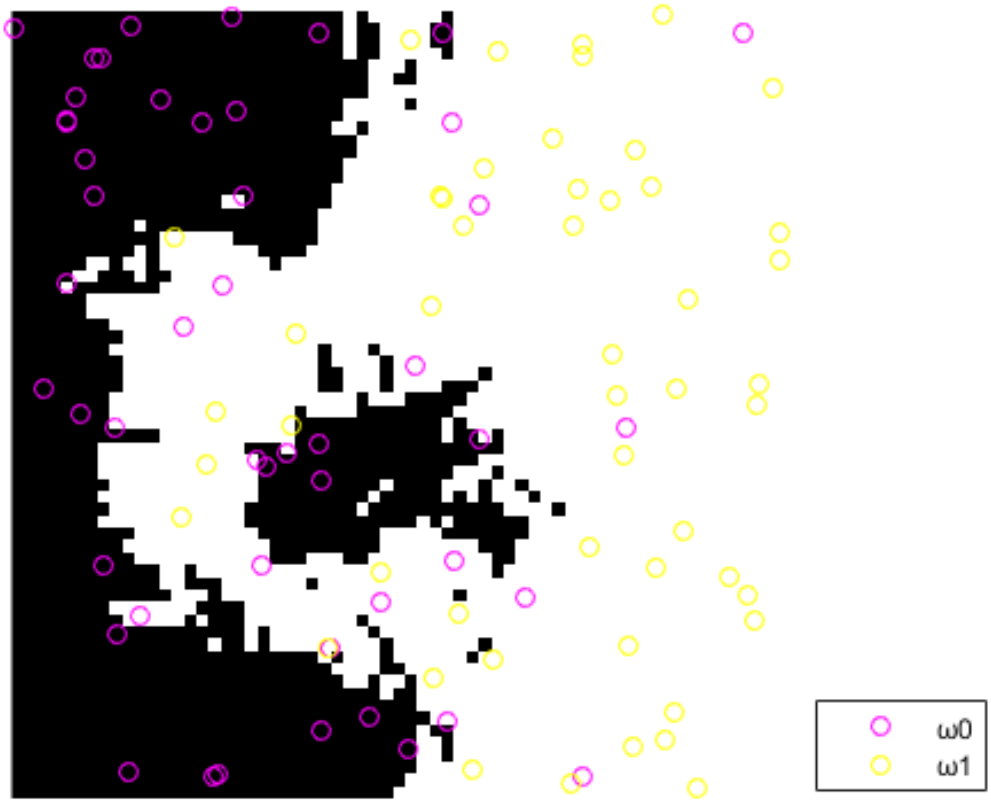

Figure 6: Plot of KNN, where k = 3

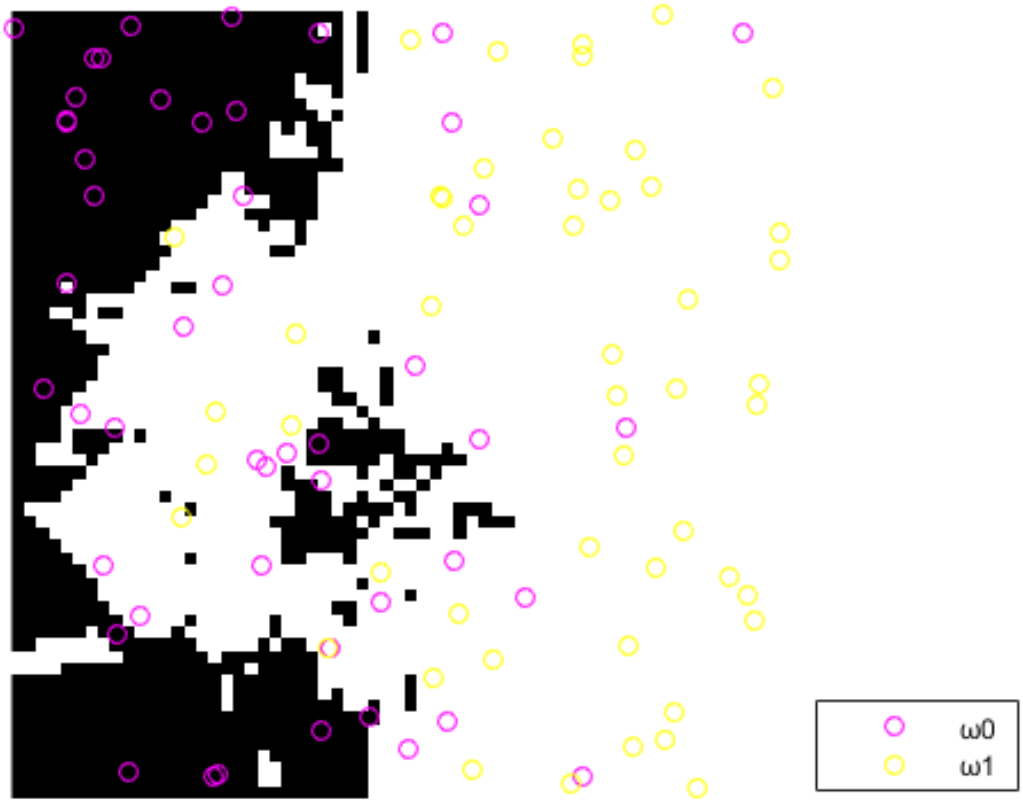

Figure 7: Plot of KNN, where k = 5

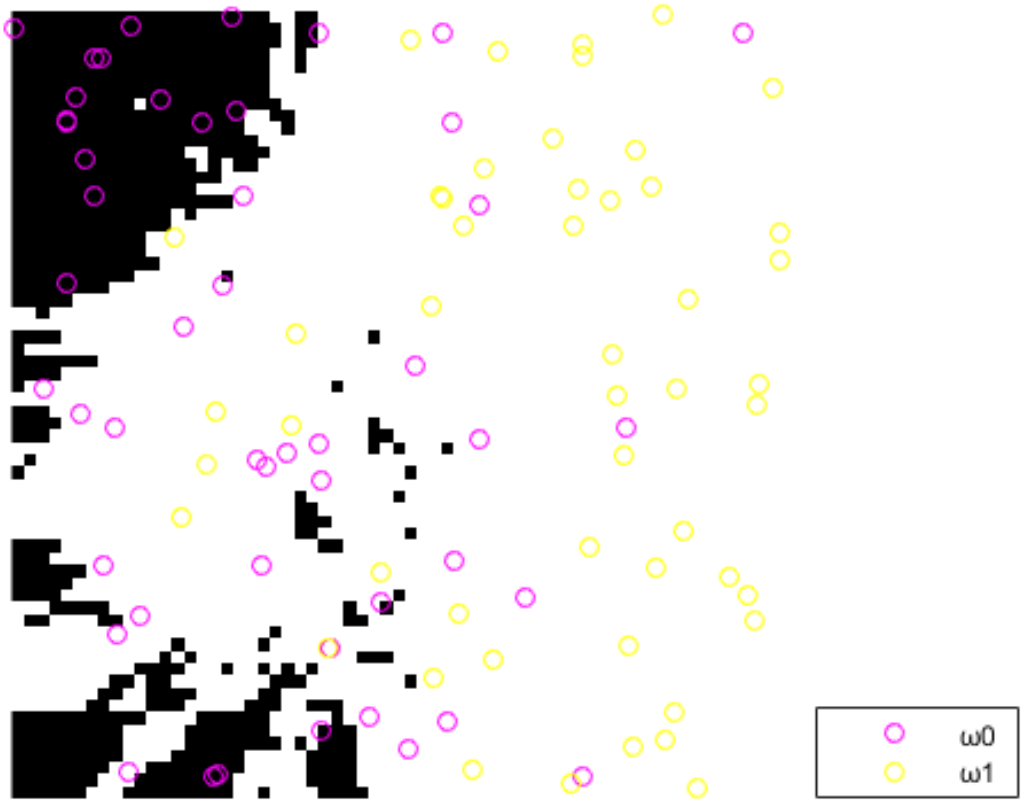

Figure 8: Plot of KNN, where k = 7

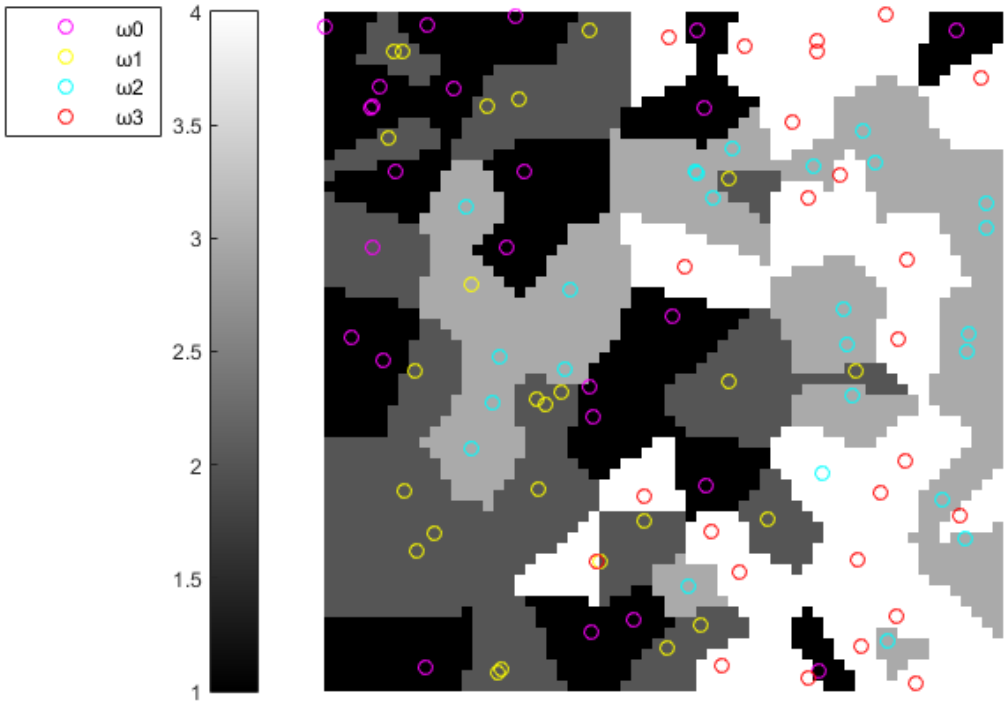

Figure 9: Plot of KNN, where k = 1 and clusters = 4

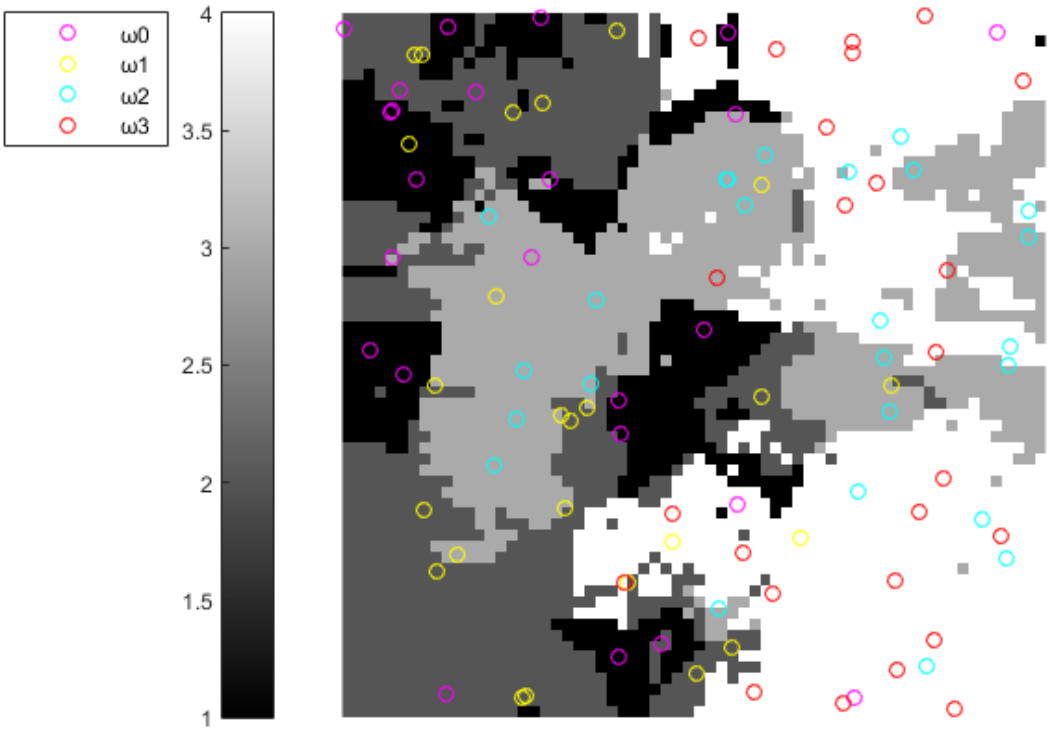

Figure 10: Plot of KNN, where k = 3 and clusters = 4

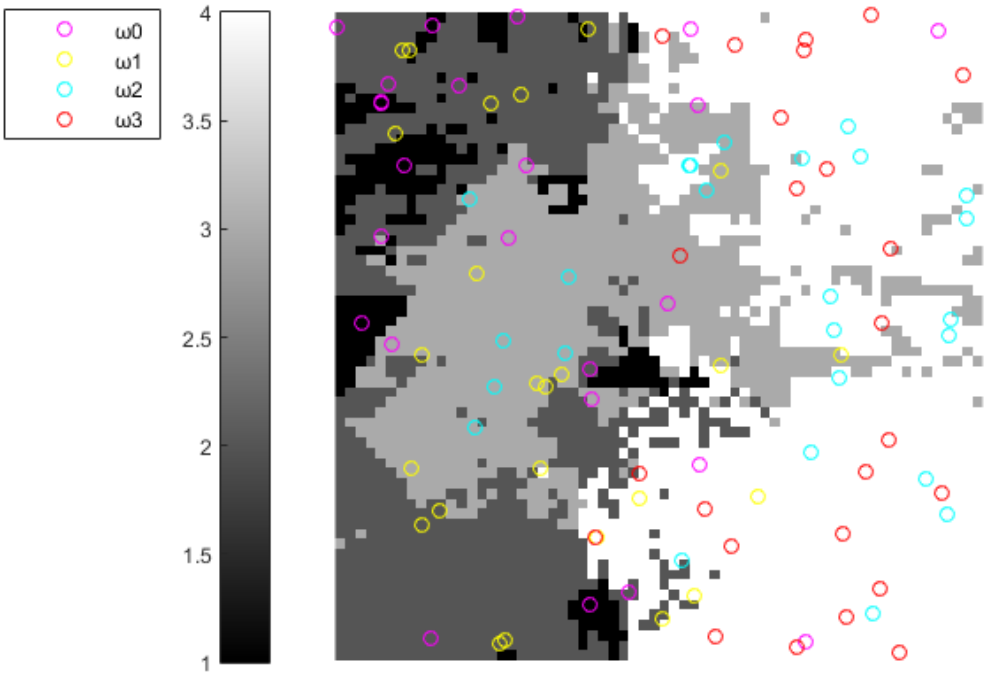

Figure 11: Plot of KNN, where k = 5 and clusters = 4

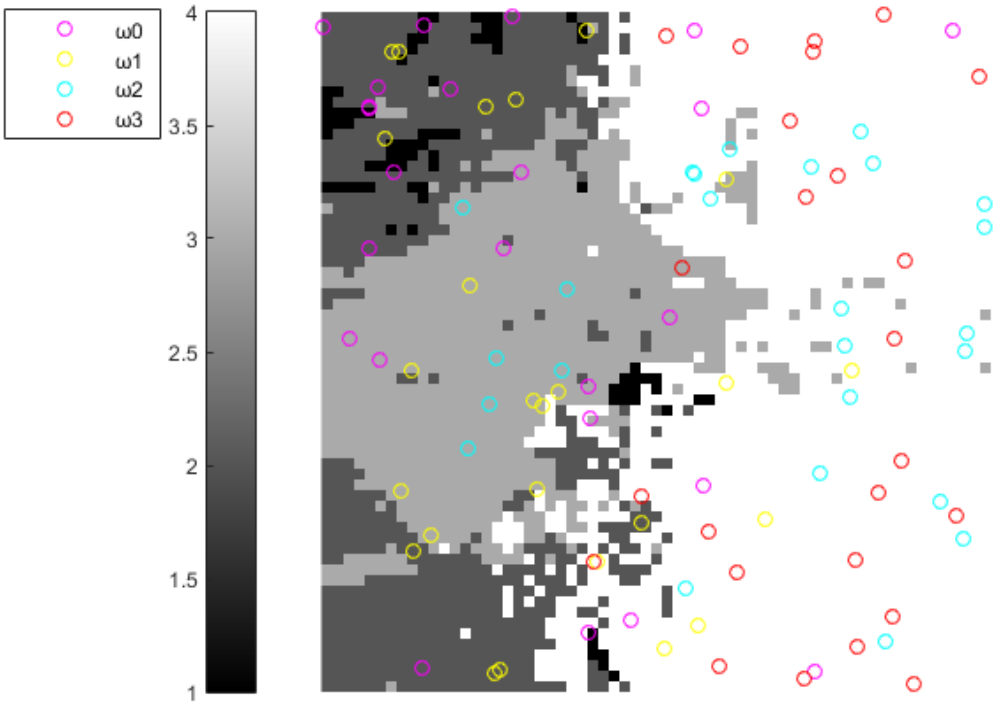

Figure 12: Plot of KNN, where k = 7 and clusters = 4

4 Division of labour

We worked together on assignment 1, where Daniël did most of the coding and Remco did most of the questions. Remco did assignment 2. The code of assignment 3 was done by Daniël, and we cooperated on the questions.