Chapter 10
Solutions to Exercises

In this chapter, all the solutions of the exercises appearing at the ends of the chapters of this book are presented. Each following section contains the solutions related to one chapter.

10.1 Problems of Chapter 2

1 The variability of the components of the points

$$(1, -1), \quad (3, 0), \quad (2, 2)$$

has to be computed. Let us consider the x components. They can assume values 1, 3 and 2, and therefore the range of variability of x is 2. The value 2 comes from the difference between the largest and the smallest values the x component can have. Similarly, the variability of the y component can be computed and it is equal to 3.

2 The following MATLAB® instructions generate a random set of points in a two-dimensional space lying on the line $y = x$. Then, the principal component analysis is applied in order to reduce the dimension of the set of points to 1:

```
>> x = rand(1,20);
>> y = x;
>> A = cov(x,y);
>> [v,d] = eig(A);
>> d

d =

         0         0
         0    0.1619

>> x1 = v(1,2)*x + v(2,2)*y;
>> y1 = v(1,1)*x + v(2,1)*y;
>> var_y1 = max(y1) - min(y1)

var_y1 =

     0
```

A. Mucherino et al., *Data Mining in Agriculture*, Springer Optimization and Its Applications 34, 185
DOI: 10.1007/978-0-387-88615-2_10,

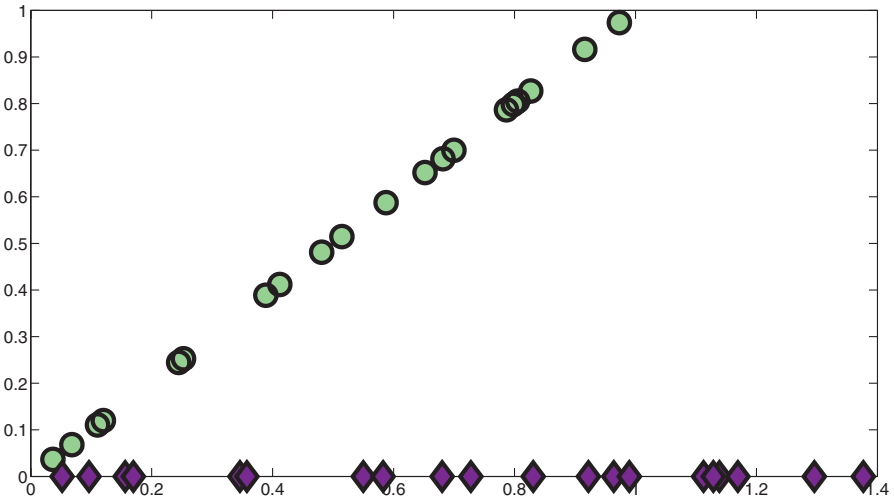

Fig. 10.1 A set of points before and after the application of the principal component analysis.

3 If the variables used in the previous exercise are still in the memory of the MATLAB environment, then the following instructions can be used for creating the Figure 10.1, as required by the exercise:

```
>> plot(x,y,'ko','MarkerSize',10,'MarkerEdgeColor','k','MarkerFaceColor',
       [.49 1 .63])
>> hold on
>> plot(x1,y1,'kd','MarkerSize',10,'MarkerEdgeColor','k','MarkerFaceColor',
       [.49 0 .63])
```

4 The equation of the unique line passing through the two points

$$(x_1, y_1) = (1, 0), \quad (x_2, y_2) = (0, -2)$$

needs to be computed. The general equation of a line l is

$$y = ax + b.$$

In this very easy case, the equation of the l can be easily obtained imposing the passage of the line through the points as follows:

$$(x_1, y_1) \in l \Longrightarrow y_1 = ax_1 + b \Longrightarrow 0 = a + b$$
$$(x_2, y_2) \in l \Longrightarrow y_2 = ax_2 + b \Longrightarrow -2 = 0 + b$$

Then,

$$\begin{cases} a = 2 \\ b = -2 \end{cases}.$$

Let us check if the line l of equation

$$y = 2x - 2$$

passes through the given points. Since

$$x_1 = 1 \Longrightarrow y_1 = ax + b = 2 \cdot 1 - 2 = 0$$
$$x_2 = 0 \Longrightarrow y_2 = ax + b = 2 \cdot 0 - 2 = -2$$

the passage is verified.

5 The following instructions draw the line which is the solution of Exercise 4 (see Figure 10.2):

```
>> x = [1 0];
>> y = [0 -2];
>> plot(x,y)
>> hold on
>> plot(x,y,'ko','MarkerSize',10,'MarkerEdgeColor','k','MarkerFaceColor',
        [.49 1 .63])
```

6 The only parabola passing through the points

$$(x_1, y_1) = (0, 1), \quad (x_2, y_2) = (1, 2), \quad (x_3, y_3) = (-1, 3)$$

has to be computed. The general equation of the Newton polynomial is

$$y = f(x_1) + \sum_{i=2}^{n+1} f[x_1, \ldots, x_i] \prod_{j=1}^{i-1} (x - x_j).$$

In this case, the Newton polynomial can be written as:

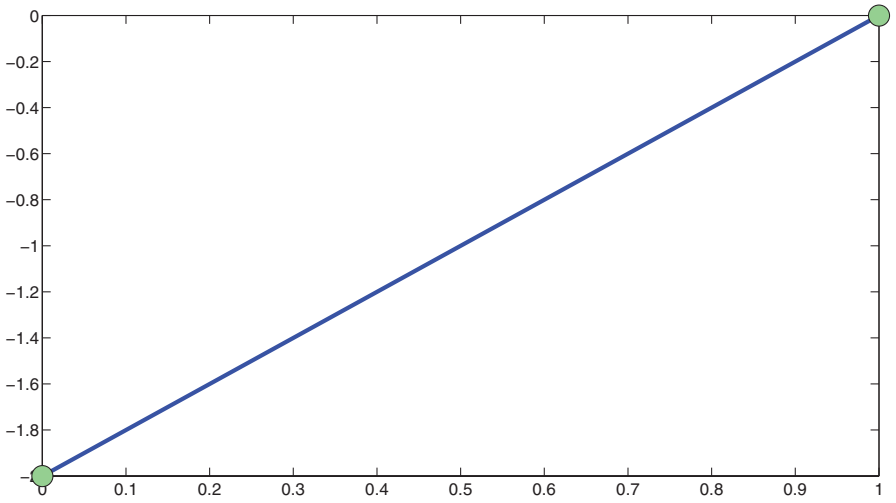

Fig. 10.2 The line which is the solution of Exercise 4.

$$y = f(x_1) + f[x_1, x_2](x - x_1) + f[x_1, x_2, x_3](x - x_1)(x - x_2).$$

Two divided differences have to be computed for finding the equation of the parabola. The first one is

$$f[x_1, x_2] = \frac{y_2 - y_1}{x_2 - x_1} = \frac{2 - 1}{1 - 0} = 1.$$

The second one needs the computation of the divided difference $f[x_2, x_3]$, because

$$f[x_1, x_2, x_3] = \frac{f[x_2, x_3] - f[x_1, x_2]}{x_3 - x_1}.$$

Since

$$f[x_2, x_3] = \frac{y_3 - y_2}{x_3 - x_2} = \frac{3 - 2}{-1 - 1} = -\frac{1}{2},$$

the needed divided difference is

$$f[x_1, x_2, x_3] = \frac{-\frac{1}{2} - 1}{-1 - 0} = \frac{3}{2}.$$

By substituting the divided differences in the Newton polynomial, the following equation is obtained:

$$y = 1 + x + \frac{3}{2}x(x - 1).$$

The passage of the given points is satisfied by the obtained equation:

$$(x_1, y_1) \Longrightarrow 1 = 1 + 0 + \frac{3}{2}0(0 - 1) = 1$$

$$(x_2, y_2) \Longrightarrow 2 = 1 + 1 + \frac{3}{2}1(1 - 1) = 2$$

$$(x_3, y_3) \Longrightarrow 3 = 1 - 1 - \frac{3}{2}1(-1 - 1) = 3.$$

Therefore, the obtained equation is actually a parabola passing from the given points.

7 A figure in which the points

$$(4, 2), (2, 2), (1, 4), (0, 0), (-1, 3)$$

and the join-the-dots function interpolating such points are displayed needs to be created. The MATLAB instructions for performing this exercise are the following ones:

```
>> x = [4 2 1 0 -1];
>> y = [2 2 4 0 3];
>> plot(x,y)
>> hold on
>> plot(x,y,'ko','MarkerSize',10,'MarkerEdgeColor','k','MarkerFaceColor',
       [.49 1 .63])
```

What is obtained is shown in Figure 10.3.

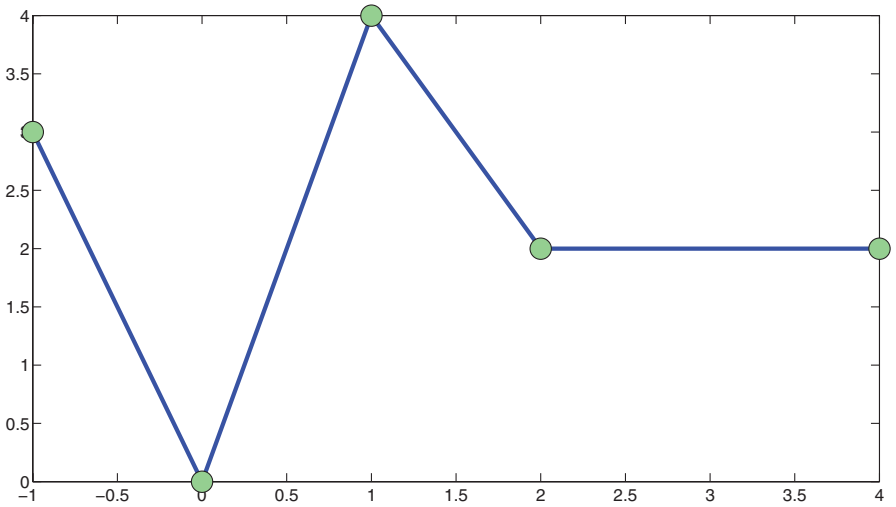

Fig. 10.3 The solution of Exercise 7.

8 Considering the same points given in Exercise 7 and supposing that the join-the-dots function is replaced by a quadratic regression function, then the exercise can be solved by the following MATLAB instructions:

```
>> x = [4 2 1 0 -1];
>> y = [2 2 4 0 3];
>> plot(x,y,'ko','MarkerSize',10,'MarkerEdgeColor','k','MarkerFaceColor',
       [.49 1 .63])
>> hold on
>> c = polyfit(x,y,2);
>> xx = min(x)-1:0.1:max(x)+1;
>> yy = polyval(c,xx);
>> plot(xx,yy)
```

What obtained is shown in Figure 10.4.

9 In this exercise, the linear and quadratic regression functions approximating the points

$$(1, 2), (2, 3), (1, -1), (-1, 3), (1, -2), (0, -1)$$

have to be computed in MATLAB. Figure 10.5 shows the result obtained by using the following instructions in the MATLAB environment:

```
>> x = [1 2 1 -1 1 0];
>> y = [2 3 -1 3 -2 -1];
>> plot(x,y,'ko','MarkerSize',10,'MarkerEdgeColor','k','MarkerFaceColor',
       [.49 1 .63])
>> hold on
>> c = polyfit(x,y,1);
>> xx = min(x)-1:0.1:max(x)+1;
>> yy = polyval(c,xx);
>> plot(xx,yy)
>> c = polyfit(x,y,2);
>> yy = polyval(c,xx);
>> plot(xx,yy,'m:')
```

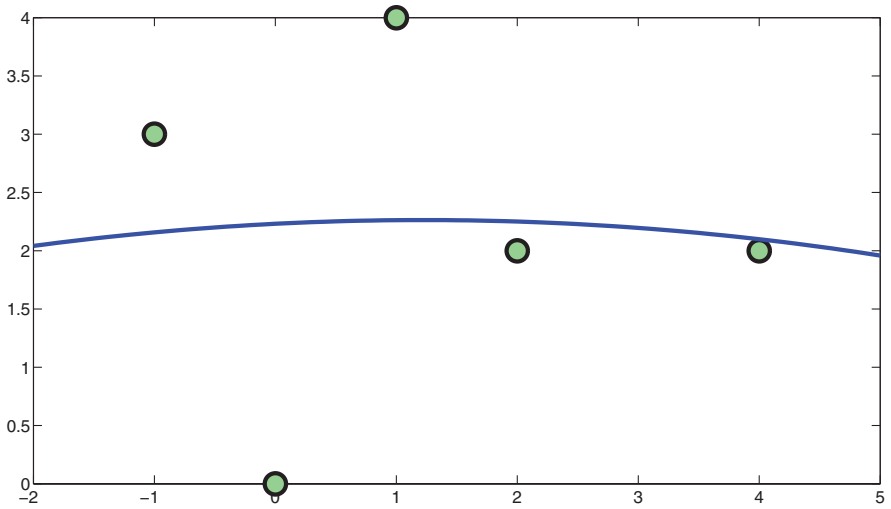

Fig. 10.4 The solution of Exercise 8.

10 In the previous exercise, the linear and quadratic regression functions related to a set of 6 points are computed. If it is supposed that each point (x, y) is approximated with the corresponding point $(x, f(x))$ of the linear regression f, then the mean arithmetic error on these 6 points can be computed by using the following MATLAB code:

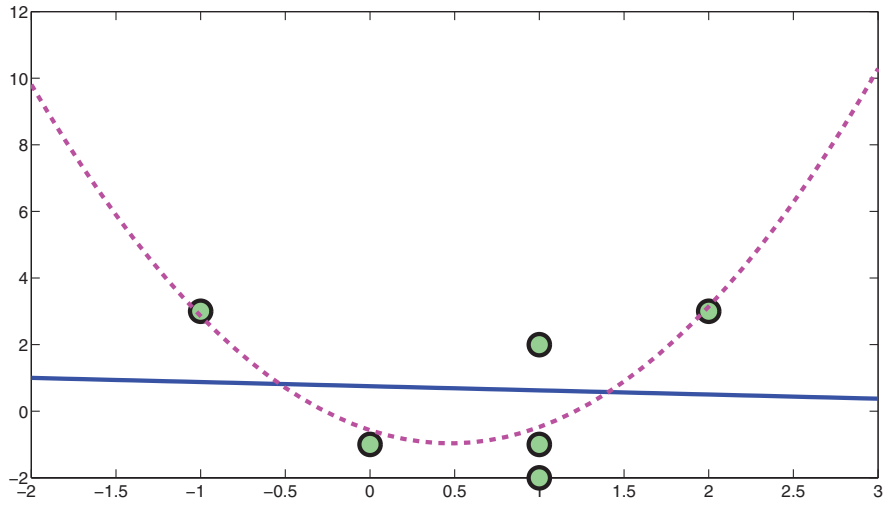

Fig. 10.5 The solution of Exercise 9.

```
>> err = 0; for i = 1:6, err = err + abs(y(i) - polyval(c,x(i))); end
>> err = err/6

err =

     2
```

10.2 Problems of Chapter 3

1 The aim of the exercise is to partition a small set of points by using the standard k-means algorithm. Let us assign a label to each considered point:

$$x_1 = (-1, -1), \quad x_2 = (-1, 1), \quad x_3 = (1, -1),$$
$$x_4 = (1, 1), \quad x_5 = (7, 8), \quad x_6 = (8, 7).$$

As suggested by the exercise, the 1^{st}, 3^{rd} and 5^{th} samples are initially assigned to class 1, and the 2^{nd}, 4^{th} and 6^{th} samples are initially assigned to class 2:

$$x_1 \to 1 \quad x_2 \to 2 \quad x_3 \to 1 \quad x_4 \to 2 \quad x_5 \to 1 \quad x_6 \to 2.$$

Let us compute the centers of these two clusters:

$$c_1 = \frac{x_1 + x_3 + x_5}{3} = \frac{(-1, -1) + (1, -1) + (7, 8)}{3} = \left(\frac{7}{3}, 2\right)$$
$$c_2 = \frac{x_2 + x_4 + x_6}{3} = \frac{(-1, 1) + (1, 1) + (8, 7)}{3} = \left(\frac{8}{3}, 3\right).$$

Following the k-means algorithm, for each point x_i, the distances $d(x_i, c_1)$ and $d(x_i, c_2)$ must be computed and the point has to be assigned to the cluster corresponding to the nearest center. Let us start from the first point x_1:

$$d(x_1, c_1) = 4.48 \quad d(x_1, c_2) = 5.43.$$

Since $d(x_1, c_1) < d(x_1, c_2)$, the point x_1 is closer to the center of the cluster 1, and therefore it is not moved to the other one. Let us consider now the second point:

$$d(x_2, c_1) = 3.48 \quad d(x_2, c_2) = 4.18.$$

The closest center is the one of the cluster 1, whereas x_2 is currently assigned to cluster 2. Then, the point x_2 is moved from cluster 2 to cluster 1. Following the algorithm, the new centers of the two clusters need to be recomputed when there is a change. In fact, the two clusters do not contain the same points anymore, and hence their centers have changed. The new partition is

$$x_1 \to 1 \quad \mathbf{x_2 \to 1} \quad x_3 \to 1 \quad x_4 \to 2 \quad x_5 \to 1 \quad x_6 \to 2$$

and the new centers are

$$c_1 = \frac{x_1 + x_2 + x_3 + x_5}{4} = \frac{(-1,-1) + (-1,1) + (1,-1) + (7,8)}{4} = \left(\frac{3}{2}, \frac{7}{4}\right)$$

$$c_2 = \frac{x_4 + x_6}{2} = \frac{(1,1) + (8,7)}{2} = \left(\frac{9}{2}, 4\right).$$

By considering the centers just computed, let us keep checking the distances starting from the point x_3:

$$d(x_3, c_1) = 2.80 \quad d(x_3, c_2) = 6.10.$$

The point x_3 results to be in the right cluster, hence it is not moved. The next point is x_4:

$$d(x_4, c_1) = 0.90 \quad d(x_4, c_2) = 4.61.$$

In this case, x_4 needs to be moved from cluster 2 to cluster 1:

$$x_1 \to 1 \quad x_2 \to 1 \quad x_3 \to 1 \quad \mathbf{x_4 \to 1} \quad x_5 \to 1 \quad x_6 \to 2,$$

and therefore new centers are computed:

$$c_1 = \frac{x_1 + x_2 + x_3 + x_4 + x_5}{5} = \frac{(-1,-1) + (-1,1) + (1,-1) + (1,1) + (7,8)}{5}$$

$$= \left(\frac{7}{5}, \frac{8}{5}\right)$$

$$c_2 = x_6 = (8,7).$$

The next point to consider is x_5, and its distances from the centers just recomputed are checked:

$$d(x_5, c_1) = 8.50 \quad d(x_5, c_2) = 1.41.$$

Since x_5 is closer to c_2, it is moved to cluster 2:

$$x_1 \to 1 \quad x_2 \to 1 \quad x_3 \to 1 \quad x_4 \to 1 \quad \mathbf{x_5 \to 2} \quad x_6 \to 2$$

and new centers are computed:

$$c_1 = \frac{x_1 + x_2 + x_3 + x_4}{4} = \frac{(-1,-1) + (-1,1) + (1,-1) + (1,1)}{4} = (0,0)$$

$$c_2 = \frac{x_5 + x_6}{2} = \frac{(7,8) + (8,7)}{2} = \left(\frac{15}{2}, \frac{15}{2}\right).$$

The last point of the set that needs to be checked is

$$d(x_6, c_1) = 10.63 \quad d(x_6, c_2) = 0.71,$$

and it is closer to the center of the cluster it is currently assigned to, and hence it is not moved. All the points have been checked at least once, and during this phase the centers changed several times. The centers are therefore not stable yet, and the

algorithm needs to restart checking the points from the first one:

$$d(x_1, c_1) = 1.41 \quad d(x_1, c_2) = 12.02.$$

The point x_1 is not moved. All the other points are not moved as well:

$$\begin{array}{ll} d(x_2, c_1) = 1.41 & d(x_2, c_2) = 10.70 \\ d(x_3, c_1) = 1.41 & d(x_3, c_2) = 10.70 \\ d(x_4, c_1) = 1.41 & d(x_4, c_2) = 9.19 \\ d(x_5, c_1) = 10.63 & d(x_5, c_2) = 0.71 \\ d(x_6, c_1) = 10.63 & d(x_6, c_2) = 0.71. \end{array}$$

Since all the points have been checked and none of them changed cluster, the centers are finally stable and the k-means algorithm can terminate.

2 In this exercise, the set of points

$$\begin{array}{c} x_1 = (1, 0), \quad x_2 = (1, 2), \quad x_3 = (2, 0), \\ x_4 = (0, 1), \quad x_5 = (1, -3), \quad x_6 = (2, 3), \quad x_7 = (3, 3) \end{array}$$

has to be partitioned in two clusters using the basic k-means algorithm. The initial partition in clusters is

$$x_1 \to 1 \quad x_2 \to 2 \quad x_3 \to 1 \quad x_4 \to 2 \quad x_5 \to 1 \quad x_6 \to 2 \quad x_7 \to 1.$$

The current centers of the clusters are

$$c_1 = \frac{x_1 + x_3 + x_5 + x_7}{4} = \frac{(1, 0) + (2, 0) + (1, -3) + (3, 3)}{4} = \left(\frac{7}{4}, 0\right)$$

$$c_2 = \frac{x_2 + x_4 + x_6}{3} = \frac{(1, 2) + (0, 1) + (2, 3)}{3} = (1, 2).$$

Following the k-means algorithm, all the points from x_1 to x_7 have to be considered and their distances from the centers of the clusters have to be checked. In this example, all the points from x_1 to x_6 do not need to be moved, because the computed distances are

$$\begin{array}{ll} d(x_1, c_1) = 0.75 & d(x_1, c_2) = 2.00 \\ d(x_2, c_1) = 2.13 & d(x_2, c_2) = 0.00 \\ d(x_3, c_1) = 0.25 & d(x_3, c_2) = 2.23 \\ d(x_4, c_1) = 2.02 & d(x_4, c_2) = 1.41 \\ d(x_5, c_1) = 3.09 & d(x_5, c_2) = 5.00 \\ d(x_6, c_1) = 3.01 & d(x_6, c_2) = 1.41. \end{array}$$

Then, the point x_7 is moved to the cluster 2, because the distances from the centers are

$$d(x_7, c_1) = 3.25 \quad d(x_7, c_2) = 2.24.$$

The new partition is therefore

$$x_1 \to 1 \quad x_2 \to 2 \quad x_3 \to 1 \quad x_4 \to 2 \quad x_5 \to 1 \quad x_6 \to 2 \quad \mathbf{x_7} \to \mathbf{2},$$

and the new centers are

$$c_1 = \frac{x_1 + x_3 + x_5}{3} = \frac{(1,0) + (2,0) + (1,-3)}{4} = \left(\frac{4}{3}, -1\right)$$
$$c_2 = \frac{x_2 + x_4 + x_6 + x_7}{4} = \frac{(1,2) + (0,1) + (2,3) + (3,3)}{4} = \left(\frac{3}{2}, \frac{9}{4}\right).$$

The centers changed, and hence another iteration of the algorithm has to be performed. These are the distances of all the points in the set from the new two centers:

$$\begin{array}{ll} d(x_1, c_1) = 1.05 & d(x_1, c_2) = 2.30 \\ d(x_2, c_1) = 3.02 & d(x_2, c_2) = 0.56 \\ d(x_3, c_1) = 1.20 & d(x_3, c_2) = 2.30 \\ d(x_4, c_1) = 2.40 & d(x_4, c_2) = 1.95 \\ d(x_5, c_1) = 2.03 & d(x_5, c_2) = 5.27 \\ d(x_6, c_1) = 4.06 & d(x_6, c_2) = 0.90 \\ d(x_7, c_1) = 4.33 & d(x_7, c_2) = 1.67. \end{array}$$

Since all the points are closer to the centers of the cluster to which they belong, none of them is moved. The algorithm then stops.

3 In this exercise, the set of points

$$\begin{array}{ccc} x_1 = (-1,-1), & x_2 = (-1,1), & x_3 = (1,-1), \\ x_4 = (1,1), & x_5 = (7,8), & x_6 = (8,7), \end{array}$$

must be partitioned in two clusters using the h-means algorithm. The centers of the initial partition in clusters are (see Exercise 1):

$$c_1 = \left(\frac{7}{3}, 2\right), \quad c_2 = \left(\frac{8}{3}, 3\right).$$

In the h-means algorithm, all the distances from the points and the centers c_1 and c_2 are computed and each point is moved to the cluster with closest center. Even though some point can migrate from a cluster to another, the centers are updated only after all the points have been checked. Let us compute all the distances:

$$\begin{array}{ll} d(x_1, c_1) = 4.48 & d(x_1, c_2) = 5.43 \\ d(x_2, c_1) = 3.48 & d(x_2, c_2) = 4.18 \\ d(x_3, c_1) = 3.28 & d(x_3, c_2) = 4.33 \\ d(x_4, c_1) = 1.67 & d(x_4, c_2) = 2.60 \\ d(x_5, c_1) = 7.60 & d(x_5, c_2) = 6.62 \\ d(x_6, c_1) = 7.76 & d(x_6, c_2) = 6.67. \end{array}$$

According to these distances, the new partition of the points becomes:

$$x_1 \to 1 \quad \mathbf{x_2 \to 1} \quad x_3 \to 1 \quad \mathbf{x_4 \to 1} \quad \mathbf{x_5 \to 2} \quad x_6 \to 2.$$

The new centers are:

$$c_1 = \frac{x_1 + x_2 + x_3 + x_4}{4} = \frac{(-1,-1) + (-1,1) + (1,-1) + (1,1)}{4} = (0,0)$$
$$c_2 = \frac{x_5 + x_6}{2} = \frac{(7,8) + (8,7)}{2} = \left(\frac{15}{2}, \frac{15}{2}\right).$$

This is the same partition obtained at the end of the solution of Exercise 1: this is the optimal partition of the points. Note that the same partition has been obtained by computing the centers only twice by using the h-means algorithm, whereas they have been computed 4 times when the k-means algorithm has been applied.

4 The k-means algorithm can find 4 different partitions in clusters having the same error function value (3.1) if, for instance, the following input is provided:

$$(-1,-1), (-1,1), (1,-1), (1,1).$$

5 An example of 8 points on a Cartesian plane that can be partitioned by k-means in 2 different ways that correspond to the same error function value (3.1) is the following one:

$$(-1,1), (0,1), (1,1),$$
$$(-1,0), (1,0),$$
$$(-1,-1), (0,-1), (1,-1).$$

6 The set of points

$$x_1 = (-1,-1), \quad x_2 = (-1,1), \quad x_3 = (1,-1),$$
$$x_4 = (1,1), \quad x_5 = (7,8), \quad x_6 = (8,7).$$

is initially assigned to the clusters 1, 2 and 3 as follows:

$$x_1 \to 1 \quad x_2 \to 2 \quad x_3 \to 1 \quad x_4 \to 2 \quad x_5 \to 1 \quad x_6 \to 2.$$

Note that the cluster 3 is currently empty. According to the k-means+ algorithm, the cluster 3 can be filled by the point that currently is the farthest from its center. Let us compute the distances from each point to the corresponding center:

$$d(x_1, c_1) = 4.48$$
$$d(x_2, c_2) = 4.18$$
$$d(x_3, c_1) = 3.28$$
$$d(x_4, c_2) = 2.60$$
$$d(x_5, c_1) = 7.60$$
$$d(x_6, c_2) = 6.67.$$

The farthest point is x_5: the new partition of the points is therefore the following one:

$$x_1 \to 1 \quad x_2 \to 2 \quad x_3 \to 1 \quad x_4 \to 2 \quad \mathbf{x_5 \to 3} \quad x_6 \to 2.$$

The current centers are

$$c_1 = (0, 1)\,, \quad c_2 = \left(\frac{8}{7}, 3\right), \quad c_3 = (7, 8)\,.$$

Let us check the distances of the points from these 3 centers:

$$\begin{array}{lll} d(x_1, c_1) = 1.00 & d(x_1, c_2) = 4.54 & d(x_1, c_3) = 12.04 \\ d(x_2, c_1) = 2.24 & d(x_2, c_2) = 2.93 & d(x_2, c_3) = 10.63. \end{array}$$

According to the algorithm, x_2 is moved to the cluster 1, and the updated centers need to be computed before proceeding. The new partition is

$$x_1 \to 1 \quad \mathbf{x_2 \to 1} \quad x_3 \to 1 \quad x_4 \to 2 \quad x_5 \to 3 \quad x_6 \to 2$$

and the new centers are

$$c_1 = \left(-\frac{1}{3}, -\frac{1}{3}\right), \quad c_2 = \left(\frac{9}{2}, 4\right), \quad c_3 = (7, 8)\,.$$

Let us continue checking the other points:

$$\begin{array}{lll} d(x_3, c_1) = 1.49 & d(x_3, c_2) = 6.10 & d(x_3, c_3) = 10.82 \\ d(x_4, c_1) = 1.89 & d(x_4, c_2) = 4.61 & d(x_4, c_3) = 9.22. \end{array}$$

The point x_4 is then moved to cluster 1. The partition is now

$$x_1 \to 1 \quad x_2 \to 1 \quad x_3 \to 1 \quad \mathbf{x_4 \to 1} \quad x_5 \to 3 \quad x_6 \to 2$$

and the centers are

$$c_1 = (0, 0)\,, \quad c_2 = (8, 7)\,, \quad c_3 = (7, 8)\,.$$

Let us continue checking the points until the last one:

$$\begin{array}{lll} d(x_5, c_1) = 10.63 & d(x_5, c_2) = 1.41 & d(x_5, c_3) = 0.00 \\ d(x_6, c_1) = 10.63 & d(x_6, c_2) = 0.00 & d(x_6, c_3) = 1.41. \end{array}$$

x_5 and x_6 are not moved. Another iteration of the algorithm starts:

$$\begin{array}{lll} d(x_1, c_1) = 1.41 & d(x_1, c_2) = 12.04 & d(x_1, c_3) = 12.04 \\ d(x_2, c_1) = 1.41 & d(x_2, c_2) = 10.82 & d(x_2, c_3) = 10.63 \\ d(x_3, c_1) = 1.41 & d(x_3, c_2) = 10.63 & d(x_3, c_3) = 10.82 \\ d(x_4, c_1) = 1.41 & d(x_4, c_2) = 9.22 & d(x_4, c_3) = 9.22 \\ d(x_5, c_1) = 10.63 & d(x_5, c_2) = 1.41 & d(x_5, c_3) = 0.00 \\ d(x_6, c_1) = 10.63 & d(x_6, c_2) = 0.00 & d(x_6, c_3) = 1.41. \end{array}$$

None of the points are moved, none of the clusters are empty, and therefore the k-means+ algorithm can stop.

7 The set of points

$$x_1 = (-1, -1), \quad x_2 = (-1, 1), \quad x_3 = (1, -1),$$
$$x_4 = (1, 1), \quad x_5 = (7, 8), \quad x_6 = (8, 7)$$

are initially assigned to 3 clusters as in the previous exercise. The cluster 3 is empty, and since x_5 is the point which is the farthest from its center (see previous exercise), it is chosen for filling the empty cluster. Then the current partition in clusters is

$$x_1 \to 1 \quad x_2 \to 2 \quad x_3 \to 1 \quad x_4 \to 2 \quad x_5 \to 3 \quad x_6 \to 2$$

and the centers of the clusters are

$$c_1 = (0, 1)\,, \quad c_2 = \left(\frac{8}{7}, 3\right), \quad c_3 = (7, 8)\,.$$

According to the h-means+ algorithm, all the distances from the points and the centers have to be checked and the centers must be updated only when all the points have been checked. The distances are

$$\begin{array}{lll} d(x_1, c_1) = 1.00 & d(x_1, c_2) = 4.54 & d(x_1, c_3) = 12.04 \\ d(x_2, c_1) = 2.24 & d(x_2, c_2) = 2.93 & d(x_2, c_3) = 10.63 \\ d(x_3, c_1) = 1.00 & d(x_3, c_2) = 4.00 & d(x_3, c_3) = 10.82 \\ d(x_4, c_1) = 2.24 & d(x_4, c_2) = 2.01 & d(x_4, c_3) = 9.22 \\ d(x_5, c_1) = 11.40 & d(x_5, c_2) = 7.70 & d(x_5, c_3) = 0.00 \\ d(x_6, c_1) = 11.31 & d(x_6, c_2) = 7.94 & d(x_6, c_3) = 1.41. \end{array}$$

Because of the distances obtained, x_2 is moved to cluster 1, and x_6 is moved to cluster 3. The new partition is then

$$x_1 \to 1 \quad \mathbf{x_2 \to 1} \quad x_3 \to 1 \quad x_4 \to 2 \quad x_5 \to 3 \quad \mathbf{x_6 \to 3}$$

and the corresponding centers are

$$c_1 = \left(-\frac{1}{3}, -\frac{1}{3}\right), \quad c_2 = (1, 1)\,, \quad c_3 = \left(\frac{15}{2}, \frac{15}{2}\right).$$

All the distances are checked another time:

$$\begin{array}{lll} d(x_1, c_1) = 0.94 & d(x_1, c_2) = 2.83 & d(x_1, c_3) = 12.02 \\ d(x_2, c_1) = 1.49 & d(x_2, c_2) = 2.00 & d(x_2, c_3) = 10.70 \\ d(x_3, c_1) = 1.49 & d(x_3, c_2) = 2.00 & d(x_3, c_3) = 10.70 \\ d(x_4, c_1) = 1.89 & d(x_4, c_2) = 0.00 & d(x_4, c_3) = 9.19 \\ d(x_5, c_1) = 11.10 & d(x_5, c_2) = 9.22 & d(x_5, c_3) = 0.71 \\ d(x_6, c_1) = 11.10 & d(x_6, c_2) = 9.22 & d(x_6, c_3) = 0.71. \end{array}$$

None of the points changed cluster, and then the h-means+ can stop.

This exercise also requires to compare the partition obtained in this exercise to the one obtained in the previous one. The two partitions are different, and this shows that the k-means(+) and h-means(+) algorithms can provide different solutions. In particular, the error function (3.1) has value 5.34 in this partition, and value 5.64 in the one of the previous exercise. Therefore, in this case, the h-means+ algorithm provided a better partition.

8 The following MATLAB code can be used for generating Figure 10.6.

```
x = [-1 -1 1 1 7 8];
y = [-1 1 -1 1 8 7];
class = [1 1 1 1 2 2];
plotp(6,x,y,class)
```

9 The possible code for the MATLAB function `hmeans` implementing the h-means algorithm in the two-dimensional space follows.

```
%
% this function performs a h-means algorithm
% on a two-dimensional set of data
%
% input:
% n - number of samples
% x - x coordinates of the samples
% y - y coordinates of the samples
% k - number of classes
%
% output:
% class - classes to which each sample belongs
%
% [class] = hmeans(n,x,y,k)

function [class] = hmeans(n,x,y,k)
```

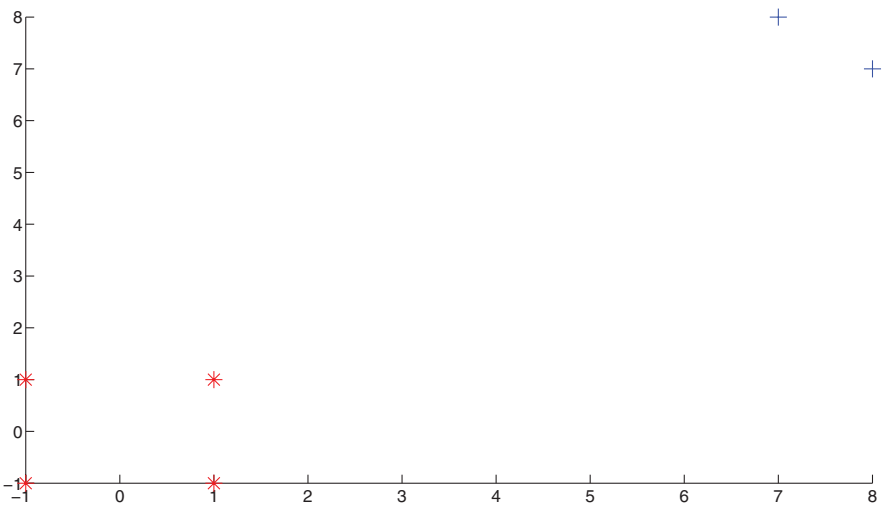

Fig. 10.6 The set of points of Exercise 1 plotted with the MATLAB function `plotp`. Note that 3 of these points lie on the x or y axis of the Cartesian system.

```
% initializing the clusters

for i = 1:n,
  class(i) = int16(k*rand());
  if class(i) == 0,
    class(i) = k;
  end
end

% computing the cluster centers

[cx,cy] = centers(n,x,y,k,class);

stable = 1;  % unstable

while stable == 1,

  % computing the distances between samples (x,y) and centers (cx,cy)
  for i = 1:n,
    mindist = 10.e+100;
    minindex = 0;
    for j = 1:k,
      dist = (x(i) - cx(j))^2 + (y(i) - cy(j))^2;
      dist = sqrt(dist);
      if dist < mindist,
        mindist = dist;
        minindex = j;
      end
    end
    % changing cluster
    class(i) = minindex;
  end

  % checking the cluster centers

  [cxnew,cynew] = centers(n,x,y,k,class);

  stable = 0;
  for j = 1:k,
    if abs(cxnew(j) - cx(j)) > 1.e-6 | abs(cynew(j) - cy(j)) > 1.e-6,
      stable = 1;
    end
  end

  % preparing for the next iteration
  for j = 1:k,
    cx(j) = cxnew(j);  cy(j) = cynew(j);
  end

end % while

end
```

10 The simple proof of the equivalence follows. We have that

$$\begin{aligned} ||x_{j_1} - x_{j_2}||^2 &= ||x_{j_1} - c_i||^2 + ||x_{j_2} - c_i||^2 - 2||x_{j_1} - c_i|| \\ &\quad \cdot ||x_{j_2} - c_i|| \cos(x_{j_1} - c_i, x_{j_2} - c_i) \\ &= ||x_{j_1} - c_i||^2 + ||x_{j_2} - c_i||^2 - 2(x_{j_1} - c_i)(x_{j_2} - c_i). \end{aligned}$$

Then the quantity

$$\sum_{j_1 \in S_i} \sum_{j_2 \in S_i} ||x_{j_1} - x_{j_2}||^2$$

is equal to

$$\sum_{j_1 \in S_i} \sum_{j_2 \in S_i} \left(||x_{j_1} - c_i||^2 + ||x_{j_2} - c_i||^2 \right) - 2 \sum_{j_1 \in S_i} \sum_{j_2 \in S_i} (x_{j_1} - c_i)(x_{j_2} - c_i).$$

The last term is zero, since

$$\sum_{j_1 \in S_i} \sum_{j_2 \in S_i} (x_{j_1} - c_i)(x_{j_2} - c_i) = \sum_{j_1 \in S_i} \left((x_{j_1} - c_i) \sum_{j_2 \in S_i} (x_{j_2} - c_i) \right)$$

and

$$\sum_{j_2 \in S_i} (x_{j_2} - c_i) = \sum_{j_2 \in S_i} x_{j_2} - |S_i| c_i = |S_i| c_i - |S_i| c_i = 0.$$

Thus,

$$\begin{aligned} \sum_{j_1 \in S_i} \sum_{j_2 \in S_i} ||x_{j_1} - x_{j_2}||^2 &= \sum_{j_1 \in S_i} \sum_{j_2 \in S_i} \left(||x_{j_1} - c_i||^2 + ||x_{j_2} - c_i||^2 \right) \\ &= 2|S_i| \sum_{j_1 \in S_i} ||x_j - c_i||^2, \end{aligned}$$

which implies the equality.

10.3 Problems of Chapter 4

1 The 1-NN rule has to be applied for classifying the points $x_1 = (2, 1), x_2 = (-3, 1)$ and $x_3 = (1, 4)$ in the two classes C^+ and C^- by using the training set:

$$\left\{ \{T_1 = (-1, -1), C^-\}, \{T_2 = (-1, 1), C^-\}, \{T_3 = (1, -1), C^+\}, \{T_4 = (1, 1), C^+\} \right\}.$$

Following the 1-NN rule, the points have to be classified in accordance with the classification of their closest point in the training set. Let us consider the first point x_1:

$$d(x_1, T_1) = 3.61,\ d(x_1, T_2) = 3.00,\ d(x_1, T_3) = 2.23,\ d(x_1, T_4) = 1.00.$$

Since the nearest point to x_1 in the training set is T_4, the point is classified in the same way as T_4:

$$x_1 \in C^+.$$

Following the same procedure, the other two points x_2 and x_3 can be classified with the same rule:

$$\begin{aligned} &d(x_2, T_1) = 2.83, \quad d(x_2, T_2) = 2.00, \quad d(x_2, T_3) = 4.47, \\ &d(x_2, T_4) = 4.00 \Longrightarrow x_2 \in C^- \end{aligned}$$

$$d(x_3, T_1) = 5.38, \quad d(x_3, T_2) = 3.61, \quad d(x_3, T_3) = 5.00,$$
$$d(x_3, T_4) = 3.00 \Longrightarrow x_3 \in C^+.$$

2 In this exercise, the points

$$x_1 = (7, 8), \quad x_2 = (0, 0), \quad x_3 = (0, 2), \quad x_4 = (4, -2)$$

have to be classified in the classes C_A and C_B by using as training set the set of points:

$$\{T_1 = (0, 1), T_2 = (-1, -1), T_3 = (1, 1)\} \in C_A,$$
$$\{T_4 = (-2, -2), T_5 = (2, 2)\} \in C_B.$$

The 1-NN rule is applied:

$$\begin{array}{llll} d(x_1, T_1) = 9.90, & d(x_1, T_2) = 12.04, & d(x_1, T_3) = 9.22, \\ d(x_1, T_4) = 13.45, & d(x_1, T_5) = 7.81 \\ d(x_2, T_1) = 1.00, & d(x_2, T_2) = 1.41, & d(x_2, T_3) = 1.41, \\ d(x_2, T_4) = 2.83, & d(x_2, T_5) = 2.83 \\ d(x_3, T_1) = 1.00, & d(x_3, T_2) = 3.16, & d(x_3, T_3) = 1.41, \\ d(x_3, T_4) = 4.47, & d(x_3, T_5) = 2.00 \\ d(x_4, T_1) = 5.00, & d(x_4, T_2) = 5.10, & d(x_4, T_3) = 4.24, \\ d(x_4, T_4) = 6.00, & d(x_4, T_5) = 4.47. \end{array}$$

According to the distance values obtained, the unknown points are classified as follows:

$$x_1 \in C_B \quad x_2 \in C_A \quad x_3 \in C_A \quad x_4 \in C_A.$$

3 In this exercise, the points

$$x_1 = (5, 1), \quad x_2 = (-1, 4),$$

must be classified into the classes C_A and C_B by using the points:

$$\{T_1 = (0, 1), T_2 = (-1, -1), T_3 = (1, 1)\} \in C_A,$$
$$\{T_4 = (-2, -2), T_5 = (2, 2)\} \in C_B.$$

The 3-NN rule is applied:

$$\begin{array}{llll} d(x_1, T_1) = 5.00, & d(x_1, T_2) = 6.32, & d(x_1, T_3) = 4.00, \\ d(x_1, T_4) = 7.62, & d(x_1, T_5) = 3.16 \\ d(x_2, T_1) = 3.16, & d(x_2, T_2) = 5.00, & d(x_2, T_3) = 3.61, \\ d(x_2, T_4) = 6.08, & d(x_2, T_5) = 3.61. \end{array}$$

Both the points x_1 and x_2 are classified as belonging to the class C_A.

4 The following training set allows different classification for the point $\hat{x} = (1, 1)$ if the k-NN rule is applied with k equal to 1 or 3. The set of points contains:

$$x_{A1} = (1, 0), \quad x_{A2} = (3, 0),$$
$$x_{B1} = (0, 0), \quad x_{B2} = (-1, 0), \quad x_{B3} = (0, 2),$$

and they are classified in the classes C_A and C_B according to their subscripts. The point $\hat{x}$ is classified as belonging to class C_A if k is 1 and it is classified as belonging to class C_B if k is 3. Let us compute the distances between $\hat{x}$ and all the points in the training set:

$$d(\hat{x}, x_{A1}) = 1.00, \quad d(\hat{x}, x_{A2}) = 2.24,$$
$$d(\hat{x}, x_{B1}) = 1.41, \quad d(\hat{x}, x_{B2}) = 2.24, \quad d(\hat{x}, x_{B3}) = 1.41.$$

The nearest point to $\hat{x}$ is x_{A1}. If the 1-NN rule is then applied, $\hat{x}$ is classified as x_{A1}, i.e., it is assigned to the class C_A. If the 3-NN rule is instead used, the three nearest neighbors of $\hat{x}$ are x_{A1}, x_{B1} and x_{B3}. Since two of them belong to the class C_B and only one to the class C_A, the unknown point $\hat{x}$ is classified as the majority of its neighbors. In this case, then, $\hat{x}$ is assigned to the class C_B.

5 The training set and the unknown sample that satisfies the requirements of Exercise 3 can be plotted by the MATLAB function `plotp`. Figure 10.7 shows the training set and the point given as solution of Exercise 4.

6 The classification problem proposed in Exercise 1 can be easily solved by using the MATLAB environment and the function `knn`. A list of instructions in MATLAB follows:

```
>> ntrain = 4;
>> xtrain = [-1 -1 1 1];
>> ytrain = [-1 1 -1 1];
>> ctrain = [1 1 2 2];
>> x = [2 -3 1];
```

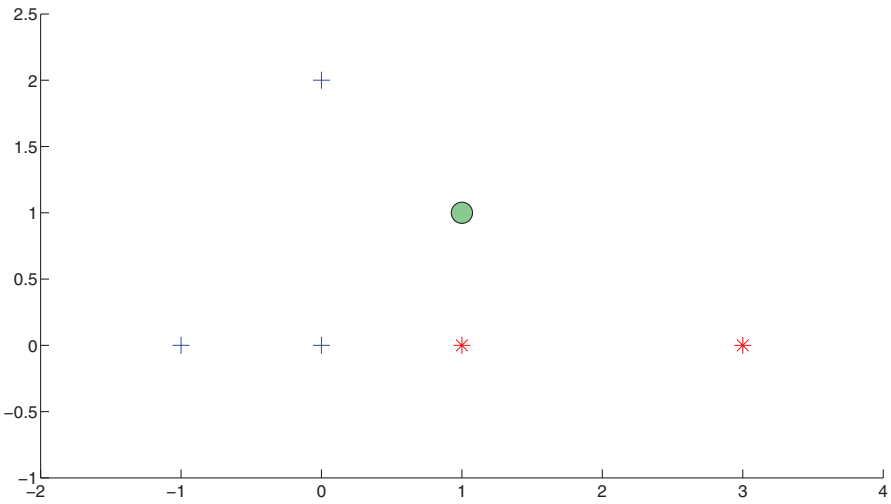

Fig. 10.7 The training set and the unknown point that represents a possible solution to Exercise 4.

```
>> y = [1 1 4];
>> class = knn(3,x,y,2,ntrain,xtrain,ytrain,ctrain)

class =

     2     1     1
```

7 In this exercise, a training set has to be randomly created and the corresponding condensed and reduced set have to be computed. In MATLAB, the following instructions can be used for this purpose:

```
>> [x,y] = generate(200,0.1);
>> [class] = hmeans(200,x,y,2);
>> [ntcnn,xtcnn,ytcnn,ctcnn] = condense(200,x,y,class,2);
>> ntcnn

ntcnn =

    11

>> [ntrnn,xtrnn,ytrnn,ctrnn] = reduce(200,x,y,class,2);
>> ntrnn

ntrnn =

     9
```

As shown, the condensed training set has only 11 points, and the reduced training set has only 9 points. The original training set was created with 200 points.

8 The figures required by the exercise can be generated using the function `plotp`. If the variables used in the previous exercise in MATLAB are still in memory, then the following instructions can be used:

```
>> plotp(200,x,y,class)
>> plotp(ntcnn,xtcnn,ytcnn,ctcnn)
>> axis([-1.5 1.5 -1 1])
>> plotp(ntrnn,xtrnn,ytrnn,ctrnn)
>> axis([-1.5 1.5 -1 1])
```

The first call to the function `plotp` generates Figure 10.8. The other two calls create Figures 10.9(a) and 10.9(b).

9 The solution of the exercise can be found by using the following instructions in MATLAB. It is supposed that the variables `x`, `y` and `class` used in the Exercise 7 are still in memory.

```
>> ntrain = 200;
>> xtrain = x;
>> ytrain = y;
>> ctrain = class;
>> [x,y] = generate(500,0);
>> [class] = knn(500,x,y,2,ntrain,xtrain,ytrain,ctrain);
>> plotp(500,x,y,class)
```

The call to the function `plotp` generates Figure 10.10.

10 If it is supposed that all the variables used in Exercise 7 are still in memory, such as the condensed and reduced subsets, then the following code can be used:

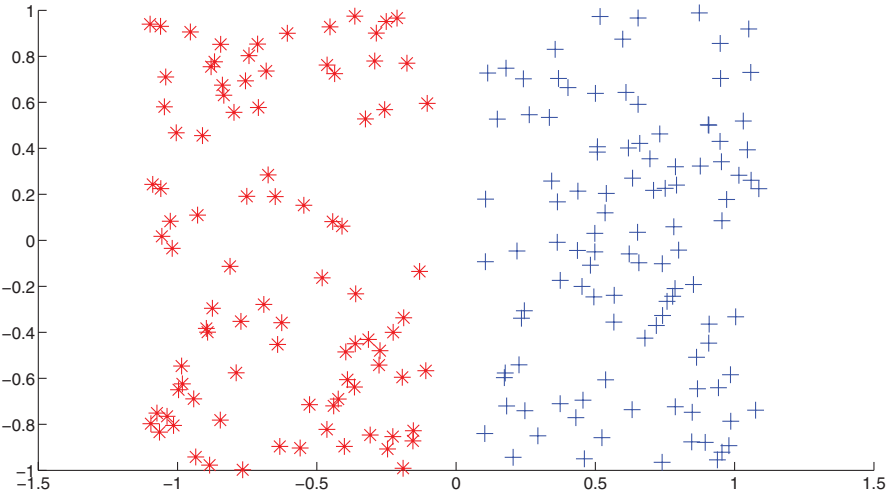

Fig. 10.8 A random set of 200 points partitioned in two clusters.

```
>> [class] = knn(500,x,y,2,ntcnn,xtcnn,ytcnn,ctcnn);
>> plotp(500,x,y,class)
>> [class] = knn(500,x,y,2,ntrnn,xtrnn,ytrnn,ctrnn);
>> plotp(500,x,y,class)
```

The two calls to the function `plotp` generate Figure 10.11.

10.4 Problems of Chapter 5

1 A multilayer perceptron having one input neuron, two hidden neurons on only one hidden layer and one output neuron has the structure shown in Figure 10.12. For the labels assigned to each neuron and weight, refer to the figure. The network has to be trained so that it is able to model the equation

$$y = 2x.$$

For simplicity, the function O_j assigned to each active neuron is the identity function, which can be expressed by the equation $y = x$. For training the network, let us consider a subset of couples of independent variables x and dependent variables y satisfying the equation $y = 2x$. For instance, the points

$$(1, 2), \quad (-1, -2), \quad (2, 4)$$

satisfy the equation. Let us start considering the first point: (1, 2). A network trained as required should be able to provide 2 when 1 is fed. When $x = 1$ is fed, this signal is sent from the input neuron A to both the neurons of the hidden layer, B and C.

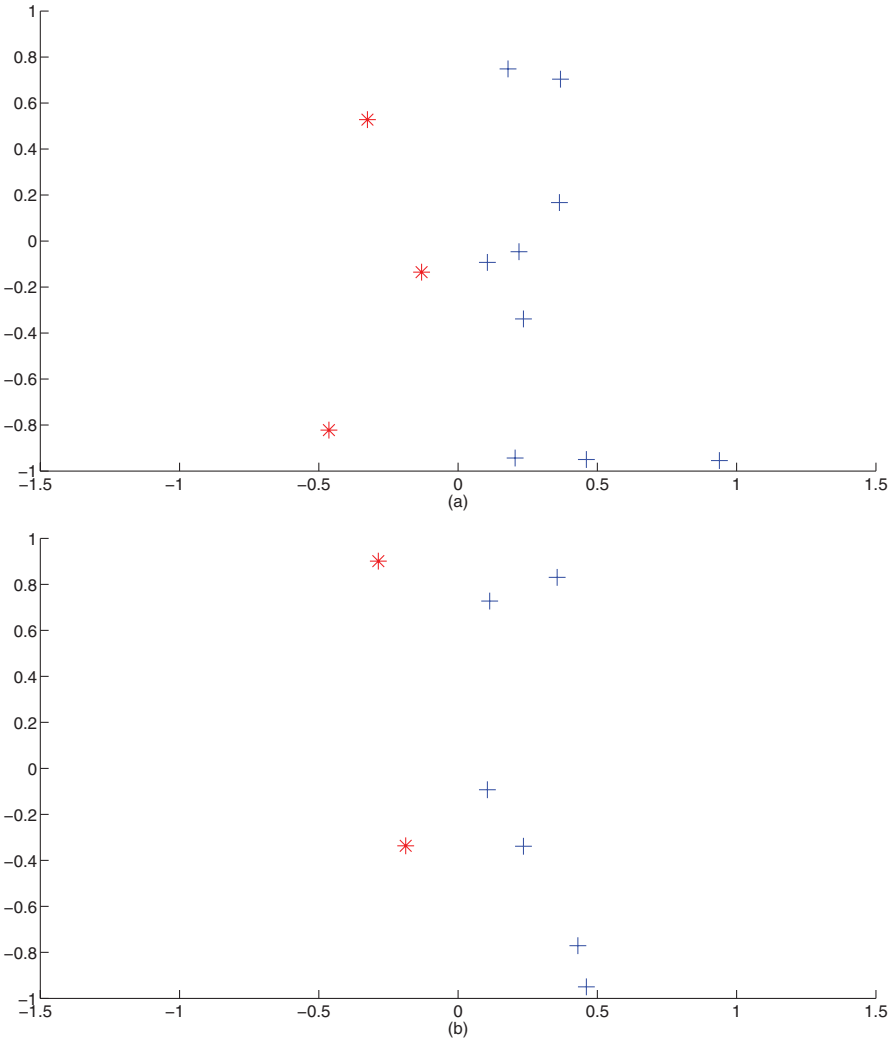

Fig. 10.9 The condensed and reduced set obtained in Exercise 7: (a) the condensed set corresponding to the set in Figure 10.8; (b) the reduced set corresponding to the set in Figure 10.8.

These two neurons compute their activation levels using the weights assigned to the links connecting them to the input neuron. In general, the activation level in B is $w_{11}x$ and the activation level in C is $w_{12}x$. Hence, in this case, the activation in B is w_{11} and the activation in C is w_{12}. The function O_j is the identity function, and therefore the neurons B and C do not modify the activation values, which are sent as they are to the output neuron. The activation level in D is $w_{11}w_{21} + w_{12}w_{22}$. As before, O_j is the identity function, and hence this is the final output provided by the network. Since the network has to mimic the equation $y = 2x$, the following

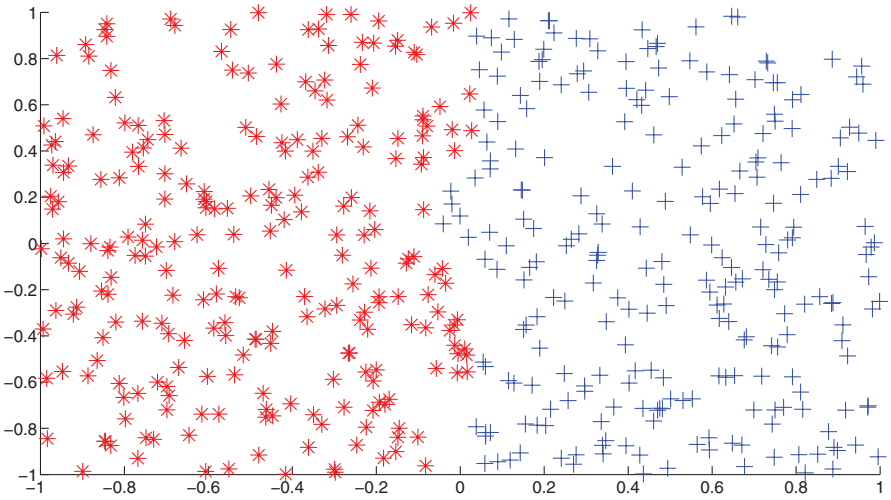

Fig. 10.10 The classification of a random set of points by using a training set of 200 points.

condition has to be satisfied:

$$w_{11}w_{21} + w_{12}w_{22} = 2. \tag{10.1}$$

If the point $(-1, -2)$ is considered, and -1 is fed to the network, the output from the network is -2 if the condition

$$-(w_{11}w_{21} + w_{12}w_{22}) = -2$$

is satisfied. Similarly, if $(2, 4)$ is considered, the condition

$$2(w_{11}w_{21} + w_{12}w_{22}) = 4$$

is obtained. Note that all these conditions depend on each other, and hence only one of them can be considered and the others discarded. If other points are considered, and other conditions obtained, they would be dependent on these ones. Let us take in account the condition (10.1). There are 4 unknown weights in only one condition, and therefore there is an infinite number of combinations of the 4 weights that satisfy such condition. For instance the weights

$$w_{11} = 1, \quad w_{21} = 1, \quad w_{12} = 2, \quad w_{22} = 1$$

satisfy the condition (10.1). The network with these weights works as the equation $y = 2x$.

2 It is needed to prove that a multilayer perceptron having one input neuron, two hidden neurons on only one hidden layer and one output neuron having the structure in Figure 10.12 cannot model the equation $y = 2x + 1$ exactly. In the previous exercise,

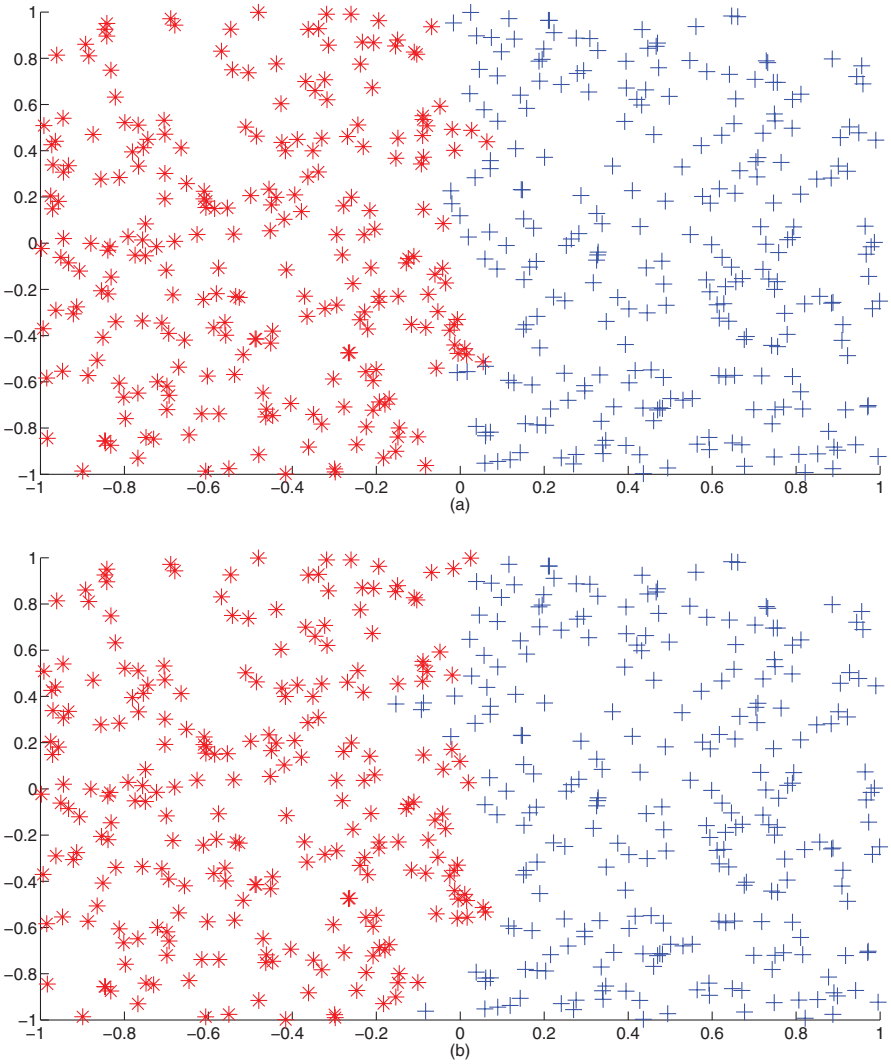

Fig. 10.11 The classification of a random set of points by using (a) the condensed set of the set in Figure 10.8; (b) the reduced set of the set in Figure 10.8.

the network has been fed with different points satisfying the equation $y = 2x$. Let us consider now the generic point satisfying the equation $y = 2x + 1$:

$$(x, 2x + 1).$$

Let us feed x to the network. The activation level in B is $w_{11}x$ and the activation level in C is $w_{12}x$. The function O_j is the identity function, and then these two activation levels are sent as they are to the output neuron. In D, the activation level

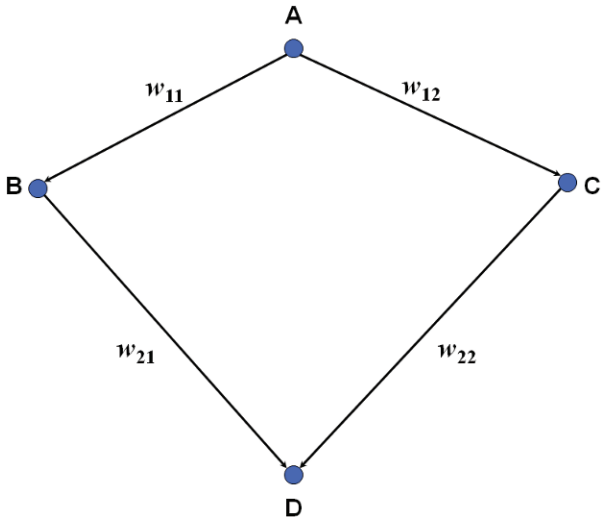

Fig. 10.12 The structure of the network considered in Exercise 1.

is $w_{11}w_{21}x + w_{12}w_{22}x$. Therefore, the following condition has to be satisfied if the network has to approximate the equation $y = 2x + 1$:

$$(w_{11}w_{21} + w_{12}w_{22})\, x = 2x + 1.$$

It follows that:

$$(w_{11}w_{21} + w_{12}w_{22} - 2)\, x = 1,$$

and this implies that the weights must depend on x for satisfying the equation. There are no possible choices for the weights that satisfy the condition for all the x, and for this reason this network cannot model the equation $y = 2x + 1$ exactly.

3 A multilayer perceptron having one hidden layer with 2 neurons has to be trained for the AND classification problem. Given two logical variables, X and Y, X AND Y must be the answer of the classification rule. As known, the AND logical operator works in accordance with the following table.

X	Y	X AND Y
True	True	True
True	False	False
False	True	False
False	False	False

In the exercise, the logical value 'true' is indicated by 0, and the logical value 'false' is indicated by 1. In this way, the previous table can be written in terms of 0 and 1.

X	Y	X AND Y
0	0	0
0	1	1
1	0	1
1	1	1

The network is trained so that, when X and Y are fed, the corresponding X AND Y value is given as output. The network has two input neurons, one corresponding to X and the other corresponding to Y, and it has only one output value, where X AND Y is provided. The hidden neurons on one hidden layer are 2. The structure of this network is in Figure 10.13: refer to the figure for the labels given to the neurons and the weights.

Let us feed the network with a generic couple (X,Y). The signal containing X starts from the neuron A and reaches the neuron C. The activation level of the neuron C is then w_{11}X. Similarly, the signal containing Y starts from the neuron B and reaches the neuron D. The activation level of the neuron D is then w_{12}Y. Successively, both neurons C and D send their signal to the input neuron E. The activation level on E is

$$w_{11}w_{21}\mathrm{X} + w_{12}w_{22}\mathrm{Y}.$$

Therefore, the network is able to provide the following results:

X	Y	Network output
0	0	0
0	1	$w_{12}w_{22}$
1	0	$w_{11}w_{21}$
1	1	$w_{11}w_{21} + w_{12}w_{22}$

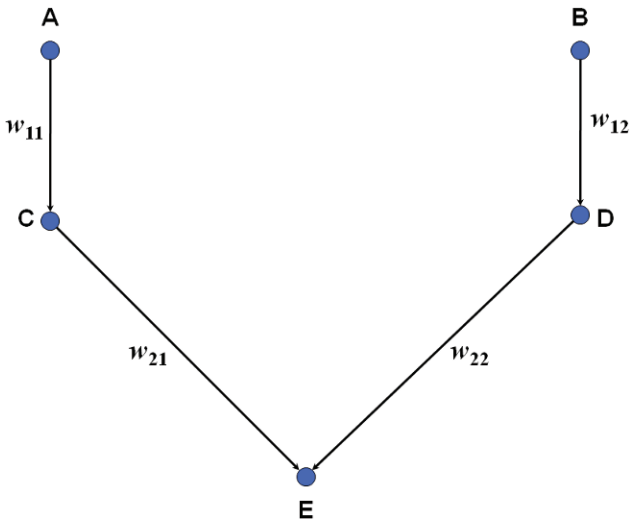

Fig. 10.13 The structure of the network considered in Exercise 3.

The network works as the AND classifier if all the weights are set to 1 and the function

$$O_j = \begin{cases} 0 \longrightarrow 0 \\ 1 \longrightarrow 1 \\ 2 \longrightarrow 1 \end{cases}$$

is associated to the neuron E.

4 The network considered in this exercise has the same structure as the one in Exercise 3. Its structure is provided in Figure 10.13. All the weights are set to 1, and the sigmoid function

$$O_j = \text{sigmoid}(x) = \frac{1}{1+e^{-x}}$$

is associated to the output neuron. Let us feed the network with (6, 1). The signal containing 6 starts from the neuron A and arrives at the neuron C unaltered. Similarly, the signal containing 1 starts from the neuron B and arrives at the neuron D unaltered. These signals start from the neurons C and D and arrive at E. The activation level in E is the weighted sum of the received signals, and therefore it is $6+1=7$. Associated to E is the sigmoid function, and hence the output value of the network is

$$\text{sigmoid}(7) = \frac{1}{1+e^{-7}}.$$

If instead $(-1, -1)$ is fed to the network, the output value of the network is

$$\text{sigmoid}(-2) = \frac{1}{1+e^{2}}.$$

5 In this exercise, the considered network has the same structure as the one in Figure 10.13. All the weights are equal to 2 and the logistic function is associated to the neuron E. When the signal propagates from one neuron to another it is doubled in value. Since there is only one hidden layer, the original signal is sent from the input layer to the hidden layer, and then from the hidden layer to the output neuron. In total, therefore, the original signal is amplified four times when it passes the network. When the neuron E receives its inputs, it sums them and applies the logistic function to the result. Thus, if (1, 1) is fed to the network, then the output provided by the network is

$$\text{logistic}(4+4) = \frac{1}{1+e^{-\frac{8}{2}}}.$$

The same result can be obtained when (0, 2) is fed:

$$\text{logistic}(0+8) = \frac{1}{1+e^{-\frac{8}{2}}}.$$

6 Two networks having the same structure as shown in Figure 10.13 are considered. The first network has all the weights equal to 1 and the sigmoid function associated to the output neuron. The second one has all the weights equal to 2 and the logistic

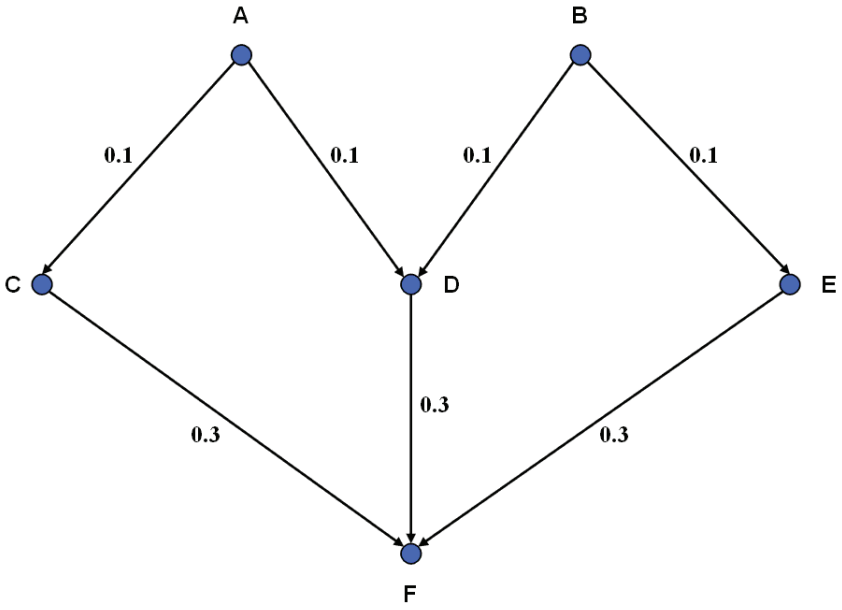

Fig. 10.14 The structure of the network considered in Exercise 7.

function associated to the output neuron. The first network can have the hidden layer removed without changing the its outputs, because the weights related to the hidden neurons are equal to 1 and no functions are associated to them. Such neurons actually do not have any effect.

7 The structure of the network considered in this exercise is shown in Figure 10.14. The weights on the links are assigned as specified in the figure. Let us feed the network with an arbitrary input (1, 2). The signal containing 1 propagates from A and the signal containing 2 propagates from B. In C the signal is 0.1, in D it is 0.2, and it is 0.1 in E. The signal in the output neuron is 0.12. It is easy to verify that, if the link between A and C is removed, then the neuron C remains inactive. Similarly, E remains inactive if the link between B and E is removed. If only one of the other links is removed, no neurons remain inactive.

8 A network having the features required by the exercise is given in Figure 10.15.

10.5 Problems of Chapter 6

1 The set of points

$$(A, B, C)$$

whose components can have 0 or 1 as value are separated in the two classes

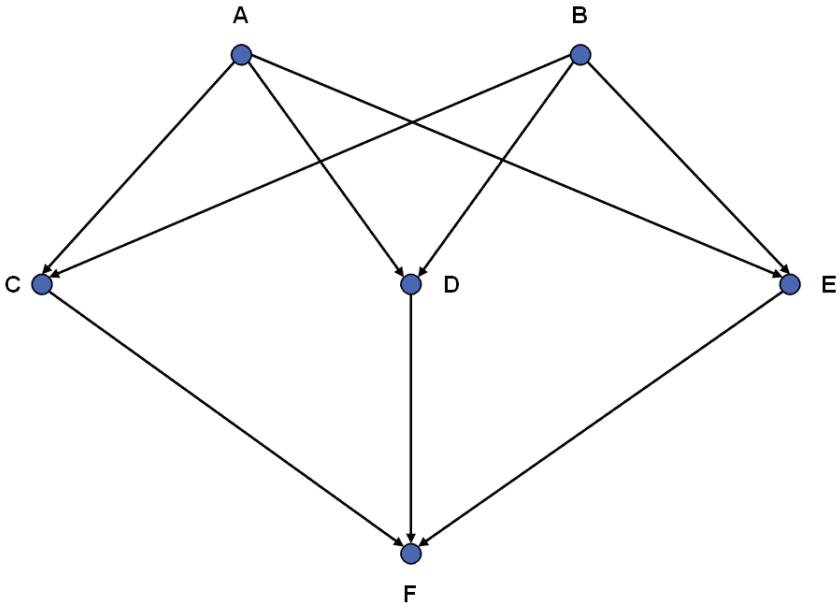

Fig. 10.15 The structure of the network required in Exercise 8.

$$C^0 = \{(A, B, C): \quad A \quad \text{AND} \quad B \quad \text{AND} \quad C \quad = \quad 0\},$$

and

$$C^1 = \{(A, B, C): \quad A \quad \text{AND} \quad B \quad \text{AND} \quad C \quad = \quad 1\}.$$

The aim of the exercise is to check if the two classes are linearly separable or not. Note that the points (A, B, C) lie on the vertices of a three-dimensional cube. Suppose that 0 stands for 'true' and that 1 stands for 'false.' From the definition of the AND operator it follows that only the point $(0, 0, 0)$ belongs to the class C^0 and all the others belong to the class C^1. Therefore, the two classes are linearly separable.

2 The classes

$$\begin{aligned} C^0 &= \{(A, B, C): \quad \text{NOT} \quad A \quad \text{AND} \quad B \quad = \quad 0\} \\ C^1 &= \{(A, B, C): \quad \text{NOT} \quad A \quad \text{AND} \quad B \quad = \quad 1\} \end{aligned}$$

are linearly separable since they can be separated by the place having equation $B - A \geq 1$. The classes

$$\begin{aligned} C^0 &= \{(A, B, C): \quad (A \quad \text{OR} \quad B) \quad \text{AND} \quad (A \quad \text{AND} \quad C) = 0\} \\ C^1 &= \{(A, B, C): \quad (A \quad \text{OR} \quad B) \quad \text{AND} \quad (A \quad \text{AND} \quad C) = 1\} \end{aligned}$$

are linearly separable as well. The plane $2A + B + C \geq 2$ separates the two classes.

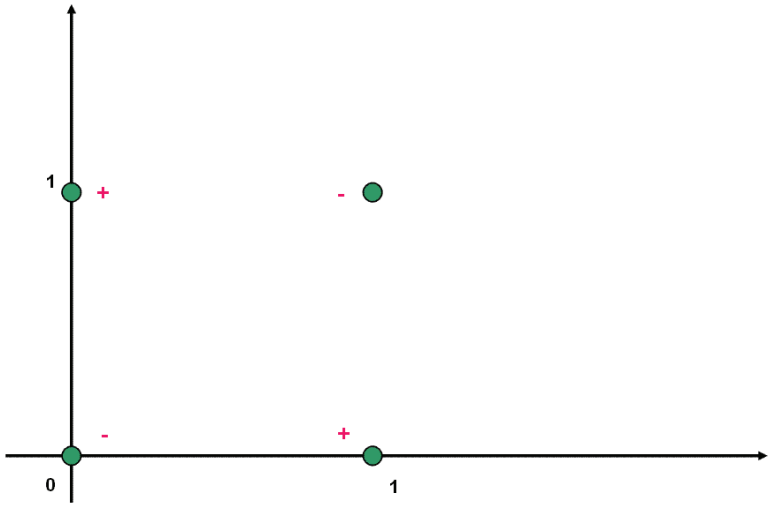

Fig. 10.16 The classes C^+ and C^- in Exercise 3.

3 A set of points and their classifications in two classes C^+ and C^- are specified as follows:

$$\left((0,0), C^-\right), \quad \left((0,1), C^+\right), \quad \left((1,0), C^+\right), \quad \left((1,1), C^-\right).$$

As it is possible to see from Figure 10.16, the classes C^+ and C^- are not linearly separable.

4 The same set of points and the same classification described in Exercise 3 are considered in this exercise. Figure 10.16 gives a geometric representation of these points. In this exercise, the transformation

$$\Phi(x_1, x_2) = \begin{pmatrix} 1 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \\ x_1^2 \\ x_2^2 \\ \sqrt{2}x_1x_2 \end{pmatrix},$$

has to be applied in order to get the two classes C^+ and C^- linearly separable. The transformation is applied point by point:

$$\begin{aligned} (0,0) &\Longrightarrow (1,0,0,0,0,0) \\ (0,1) &\Longrightarrow (1,0,\sqrt{2},0,1,0) \\ (1,0) &\Longrightarrow (1,\sqrt{2},0,1,0,0) \\ (1,1) &\Longrightarrow (1,\sqrt{2},\sqrt{2},1,1,\sqrt{2}). \end{aligned}$$

Note that the first component of the transformed points is always 1, and therefore it can be discarded. Moreover, the components 2 and 3 and the components 4 and 5 of the transformed points satisfy a particular symmetry property. Indeed, the components 2 and 3 are

$$(0, 0), \quad (0, \sqrt{2}), \quad (\sqrt{2}, 0), \quad (\sqrt{2}, \sqrt{2}),$$

and the components 4 and 5 are

$$(0, 0), \quad (0, 1), \quad (1, 0), \quad (1, 1).$$

Thus, these two couples of components have the same coefficients in the separating hyperplane equation because of the symmetry. Simplifying, the given points are transformed in:

$$\left((0, 0, 0), C^-\right), \quad \left((\sqrt{2}, 1, 0), C^+\right), \quad \left((\sqrt{2}, 1, 0), C^+\right), \quad \left((2\sqrt{2}, 2, \sqrt{2}), C^-\right).$$

The second and the third point are identical, and therefore only three points are considered. There is always a plane in the three-dimensional space that can separate a point by other two different points, and therefore the obtained points belong to classes that are linearly separable.

5 The optimization problem to be solved for training a support vector machine related to the set of points in the transformed space considered in the previous exercise is

$$\min_w \frac{1}{2}\left(w_1^2 + w_2^2 + w_3^2\right)$$

subject to

$$\begin{aligned} b &\leq -1 \\ \sqrt{2}w_1 + w_2 + b &\geq 1 \\ 2\sqrt{2}w_1 + 2w_2 + \sqrt{2}w_3 + b &\leq 1. \end{aligned}$$

6 The experiments discussed in Section 6.6 regard the use of the freeware software LIBSVM. In the quoted section, a training test and a testing set have been generated randomly by using the MATLAB function `generate4libsvm`. In the experiments, a support vector machine has been trained by using a sigmoidal kernel. In the following, two different support vector machines are trained by using the same training set but two different kernel functions.

```
LIBSVM>svmtrain -t 1 trainset.txt
*
optimization finished, #iter = 35
nu = 0.584346
obj = -47.033541, rho = -0.249598
nSV = 61, nBSV = 58
Total nSV = 61

LIBSVM>svmpredict testset.txt trainset.txt.model
                  testresult-polynomial-kernel.txt
Accuracy = 82.6% (826/1000) (classification)

LIBSVM>svmtrain -t 2 trainset.txt
*
```

```
optimization finished, #iter = 17
nu = 0.175650
obj = -11.319766, rho = 0.030302
nSV = 20, nBSV = 16
Total nSV = 20

LIBSVM>svmpredict testset.txt trainset.txt.model
                  testresult-polynomial-kernel.txt
Accuracy = 98.6% (986/1000) (classification)
```

These experiments show that the kernel that performs better on the considered problem is the radial basis kernel, which is specified by '2' when the option '`-t`' of the procedure `svmtrain` is used.

7 This exercise uses the same notations introduced in Section 6.1. For instance, w and b are the parameters of the general equation of the hyperplane:

$$w^T x + b = 0.$$

As known, the two parameters w and b can be normalized so that $w^T x + b = +1$ is the hyperplane that goes through the support vectors of the class C^+, and $w^T x + b = -1$ is the hyperplane that goes through the support vectors of the class C^-. If x^+ is a sample on the hyperplane C^+ and x^- is the sample closest to x^+ on the hyperplane C^-, then the margin between the two hyperplanes can be written as:

$$M = |x^+ - x^-|.$$

The aim of this exercise is to prove that the margin M between the two classes can be also written as:

$$M = \frac{2}{\sqrt{w^T w}}.$$

Since w is orthogonal to both C^+ and C^-, then

$$x^+ = x^- + \lambda w$$

for some real λ. The following system of conditions

$$\begin{cases} w^T x^+ + b = +1 \\ w^T x^- + b = -1 \\ x^+ = x^- + \lambda w \\ M = |x^+ - x^-| \end{cases}$$

implies that

$$\begin{aligned} & w^T (x^- + \lambda w) = 1 \\ \Longrightarrow\ & w^T x^- + b + \lambda w^T w = 1 \\ \Longrightarrow\ & -1 + \lambda w^T w = 1 \\ \Longrightarrow\ & \lambda = \frac{2}{w^T w}. \end{aligned}$$

Therefore,

$$M = |x^+ - x^-| = |\lambda w| = \lambda |w| = \lambda \sqrt{w^T w},$$

and thus

$$M = \frac{2}{\sqrt{w^T w}},$$

and hence the proof is completed.

10.6 Problems of Chapter 7

1 The matrix

$$A = \begin{pmatrix} 1 & 2 & 3 & -4 & 5 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 2 & 2 & 0 \\ -1 & 3 & 1 & 0 & 2 \\ 3 & -1 & 1 & 2 & 1 \end{pmatrix}$$

represents a set of samples and features that can be partitioned in biclusters. Each column of the matrix represents a sample, each row of A represents instead a feature. A possible bicluster with constant row values is

$$C_A = \begin{pmatrix} 0 & 0 \\ 2 & 2 \end{pmatrix},$$

where C_A can be obtained by A by extracting its second and third rows and its third and fourth column.

2 The set of points:

$$\begin{array}{lll} x_1 = (7, 0, 0), & x_2 = (5, 0, 0), & x_3 = (0, 1, 0), \\ x_4 = (0, 3, 0), & x_5 = (0, 0, 1), & x_6 = (0, 0, 5) \end{array}$$

is given and their partition is assigned as follows:

$$x_1 \in S_1, \quad x_2 \in S_1, \quad x_3 \in S_2, \quad x_4 \in S_2, \quad x_5 \in S_3, \quad x_6 \in S_3.$$

The matrix A associated to this set of data is

$$A = \begin{pmatrix} 7 & 5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 5 \end{pmatrix}$$

and then the features are represented by the three 6-dimensional points:

$$\begin{array}{l} f_1 = (7, 5, 0, 0, 0, 0) \\ f_2 = (0, 0, 1, 3, 0, 0) \\ f_3 = (0, 0, 0, 0, 1, 5). \end{array}$$

Let us compute the centers of the three clusters S_1, S_2 and S_3:

$$c_1^S = \frac{x_1 + x_2}{2} = \frac{(7,0,0) + (5,0,0)}{2} = (6,0,0) = (c_{11}^S, c_{21}^S, c_{31}^S)$$
$$c_2^S = \frac{x_3 + x_4}{2} = \frac{(0,1,0) + (0,3,0)}{2} = (0,2,0) = (c_{12}^S, c_{22}^S, c_{32}^S)$$
$$c_3^S = \frac{x_5 + x_6}{2} = \frac{(0,0,1) + (0,0,5)}{2} = (0,0,3) = (c_{13}^S, c_{23}^S, c_{33}^S).$$

By applying the rule (7.2), it follows that

$$\begin{aligned} c_{11}^S > c_{12}^S \quad \text{and} \quad c_{11}^S > c_{13}^S \quad &\Longrightarrow \quad f_1 \in F_1 \\ c_{22}^S > c_{21}^S \quad \text{and} \quad c_{22}^S > c_{23}^S \quad &\Longrightarrow \quad f_2 \in F_2 \\ c_{33}^S > c_{31}^S \quad \text{and} \quad c_{33}^S > c_{32}^S \quad &\Longrightarrow \quad f_3 \in F_3. \end{aligned}$$

Thus, the partition in biclusters is

$$B = \{(x_1, x_2, f_1), (x_3, x_4, f_2), (x_5, x_6, f_3)\}.$$

3 In this exercise, the partition in biclusters obtained in the previous exercise must be checked for consistency. In such a partition, each feature is contained in a different bicluster, and therefore each center c_r^F equals the r^{th} feature f_r:

$$c_1^F = f_1, \quad c_2^F = f_2, \quad c_3^F = f_3.$$

The rule (7.3) can be applied:

$$\begin{aligned} c_{11}^F > c_{12}^F \quad \text{and} \quad c_{11}^F > c_{13}^F \quad &\Longrightarrow \quad x_1 \in \hat{S}_1 \\ c_{21}^F > c_{22}^F \quad \text{and} \quad c_{21}^F > c_{23}^F \quad &\Longrightarrow \quad x_2 \in \hat{S}_1 \\ c_{32}^F > c_{31}^F \quad \text{and} \quad c_{32}^F > c_{33}^F \quad &\Longrightarrow \quad x_3 \in \hat{S}_2 \\ c_{42}^F > c_{41}^F \quad \text{and} \quad c_{42}^F > c_{43}^F \quad &\Longrightarrow \quad x_4 \in \hat{S}_2 \\ c_{53}^F > c_{51}^F \quad \text{and} \quad c_{53}^F > c_{52}^F \quad &\Longrightarrow \quad x_5 \in \hat{S}_3 \\ c_{63}^F > c_{61}^F \quad \text{and} \quad c_{63}^F > c_{62}^F \quad &\Longrightarrow \quad x_6 \in \hat{S}_3. \end{aligned}$$

The partition found in clusters $\hat{S}_r$ is equal to the partition in clusters S_r. Thus, the partition in biclusters is consistent.

4 The samples x_i and the features f_i related to this exercise can be summarized in the matrix

$$A = \begin{pmatrix} 1\,2\,3\,4 \\ 2\,3\,4\,5 \\ 3\,4\,2\,1 \end{pmatrix}.$$

The columns of the matrix represent the 4 points in the three-dimensional space to which a partition in cluster is already assigned: the first two columns belong to the cluster S_1, whereas the last two columns belong to the cluster S_2. Let us compute the centers of these two clusters:

$$c_1^S = \frac{x_1 + x_2}{2} = \frac{(1,2,3) + (2,3,4)}{2} = \left(\frac{3}{2}, \frac{5}{2}, \frac{7}{2}\right)$$
$$c_2^S = \frac{x_3 + x_4}{2} = \frac{(3,4,2) + (4,5,1)}{2} = \left(\frac{7}{2}, \frac{9}{2}, \frac{3}{2}\right).$$

Let us apply the rule (7.2):

$$\begin{aligned} c_{11}^S > c_{12}^S \quad &\Longrightarrow \quad f_1 = (1,2,3,4) \in F_1 \\ c_{21}^S > c_{22}^S \quad &\Longrightarrow \quad f_2 = (2,3,4,5) \in F_1 \\ c_{31}^S < c_{32}^S \quad &\Longrightarrow \quad f_3 = (3,4,2,1) \in F_2. \end{aligned}$$

Then, the partition in biclusters is

$$B = \{(x_1, x_2, f_1, f_2), (x_3, x_4, f_3)\}.$$

Let us now check if the obtained partition B is consistent. The centers of the clusters F_r are

$$c_1^F = \frac{f_1 + f_2}{2} = \frac{(1,2,3,4) + (2,3,4,5)}{2} = \left(\frac{3}{2}, \frac{5}{2}, \frac{7}{2}, \frac{11}{2}\right)$$
$$c_2^F = f_3 = (3,4,2,1).$$

The rule (7.3) is applied:

$$\begin{aligned} c_{11}^F < c_{12}^F \quad &\Longrightarrow \quad x_1 = (1,2,3) \in \hat{S}_2 \\ c_{21}^F < c_{22}^F \quad &\Longrightarrow \quad x_2 = (2,3,4) \in \hat{S}_2 \\ c_{31}^F > c_{32}^F \quad &\Longrightarrow \quad x_3 = (3,4,2) \in \hat{S}_1 \\ c_{41}^F > c_{42}^F \quad &\Longrightarrow \quad x_4 = (4,5,1) \in \hat{S}_1. \end{aligned}$$

The partitions in biclusters S_r and $\hat{S}_r$ are different, and therefore the obtained biclustering B is not consistent.

5 Impossible. Every α-consistent biclustering, for any α, is also consistent.