Chapter 10
Pattern Recognition and Cluster Analysis

Let us begin with a terminological remark, which concerns the notion of a pattern. In pattern recognition and cluster analysis various objects, phenomena, processes, structures, etc. can be considered as *patterns*. The notion is not limited to images, which can be perceived by our sight. There are three basic approaches in the area of pattern recognition. In the approach based on a feature space a pattern is represented by a feature vector. If patterns are of a structural nature, then syntactic pattern recognition (introduced in Chap. 8) or the structural approach[1] is used. In the third approach (artificial) neural networks are applied. (This approach is introduced in the next chapter.)

In general, *pattern recognition* consists of classifying an unknown pattern into one of several predefined categories, called *classes*.[2] *Cluster analysis* can be considered a complementary problem to pattern recognition. Grouping a set of patterns into classes (categories) is its main task.[3]

The task of pattern recognition and its basic notions are formulated in the first section. The next five sections concern various methods of pattern recognition. Cluster analysis is introduced in the last section.

[1]In structural pattern recognition patterns are represented by structural representations, similarly to syntactic pattern recognition. However, their recognition is done with the help of *pattern matching* methods, not, as in the syntactic approach, by applying formal grammars and automata.

[2]For example, patients can be considered as patterns and then *pattern recognition* can consist of classifying them into one of several disease entities.

[3]For example in the area of *Business Intelligence* we can try to group customers on the basis of their features such as the date of their last purchase, the total value of their purchases for the last two months, etc. into categories which determine a sales strategy (e.g., cross-selling, additional free services/products).

M. Flasiński, *Introduction to Artificial Intelligence*,
DOI 10.1007/978-3-319-40022-8_10

10.1 Problem of Pattern Recognition

The problem of *pattern recognition* can be described formally in the following way. Let us assume that there are C categories (classes):

$$\omega^1, \omega^2, \ldots, \omega^C, \tag{10.1}$$

into which patterns can be classified. Let us assume also that each pattern is represented by an n-dimensional *feature vector* $\mathbf{X} = (X_1, X_2, \ldots, X_n)$, where $X_i, i = 1, \ldots, n$ is called the ith *component*[4] *of the vector* $\mathbf{X}$.

In order to perform a pattern recognition task we should have a *learning (training) set*, which is defined in the following way:

$$U = (\, (\mathbf{X}^1, u^1), (\mathbf{X}^2, u^2), \ldots, (\mathbf{X}^M, u^M)\,), \tag{10.2}$$

where $\mathbf{X}^j = (X_1^j, X_2^j, \ldots, X_n^j),\ j = 1, \ldots, M$, is the jth vector of the learning set and $u^j = \omega^k, k \in \{1, \ldots, C\}$, is the correct classification of the pattern represented by the vector $\mathbf{X}^j$. (This means that the pattern represented by the vector $\mathbf{X}^j$ belongs to the class ω^k.)

In this chapter we focus on the phase of classification, assuming that the representation of patterns in the form of a learning set has been defined in a correct way. However, before we introduce classification methods in the following sections, we consider a few issues which concern defining a correct representation of patterns. In a pattern recognition system, the following three phases precede classification:

- preprocessing,
- feature extraction,
- feature selection.

During *preprocessing* the following operations are performed: noise removal, smoothing, and normalization. Noise removal is usually done with the help of signal filtering methods.[5] Normalization consists of scaling pattern features so that they belong to comparable ranges.

For the classification phase we require the number of pattern features to be as small as possible, i.e., the dimensionality of feature vectors should be as small as possible. If the dimensionality is big, then it results in a high cost of feature measuring, a (time) inefficiency of classification algorithms, and, interestingly, often more errors in the classification phase.[6] Reduction of this dimensionality is the main task of

[4]A component represents some feature of the pattern.

[5]If patterns are images, then noise filtering, smoothing/sharpening, enhancement, and restoration are typical preprocessing operations. Then, features such as edges, characteristic points, etc. are identified. Finally, image segmentation and object identification are performed.

[6]This interesting phenomenon is discussed, e.g., in [235].

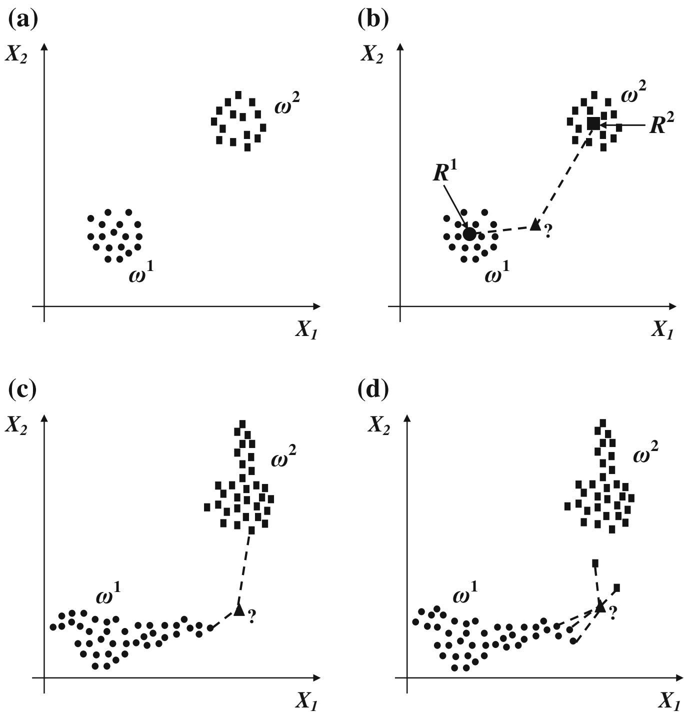

Fig. 10.1 **a** An example of a feature space containing elements of a learning set, and examples of methods: **b** minimum distance classification, **c** nearest neighbor (NN), **d** k-NN

the *feature extraction* phase. The reduction is done by combining and transforming original features into new ones.[7]

Feature selection is the next phase. It consists of selecting those features which have the biggest discriminative power. In other words, we identify those features that lead to the smallest error during the classification phase.[8]

A space containing vectors representing patterns such that their components have been extracted and selected during the phases described above is called a *feature space*. An example feature space is shown in Fig. 10.1a. It is a two-dimensional space, i.e., patterns belonging to it are represented by two features: X_1 and X_2. There are patterns belonging to a class ω^1, marked with circles, in the space. As we can see, these patterns are concentrated, i.e., they are close each to other. We say they create a *cluster*. Similarly, patterns belonging to a class ω^2, marked with rectangles,

[7]The most popular feature extraction methods include *Principal Component Analysis (PCA)*, *Independent Component Analysis*, and *Linear Discriminant Analysis*. The issues related to feature extraction methods are out of the scope of Artificial Intelligence. Therefore, they are not discussed in the book. The reader can find a good introduction to this area in monographs cited at the end of this chapter.

[8]This can be done, for example, with the help of the *search methods* introduced in Chap. 4.

create a second cluster in the feature space. Let us assume that we want to construct a pattern recognition system which distinguishes between *sprats* and *eels*. Thus, there are two classes: ω^1 = *sprats*, ω^2 = *eels*. Then, let us assume that the system classifies fishes on the basis of two features: X_1 = *length of fish*, X_2 = *weight of fish*. Of course, a learning set should be available, i.e., we should have a set of fishes of both species. We should measure the length of the fishes and we should weigh them. Then, we can place patterns of fishes in the feature space. In Fig. 10.1a sprats (marked with circles) are shorter (the feature X_1) and lighter (the feature X_2) than eels. In successive sections we discuss various classification methods, assuming a learning set is placed in a feature space.

10.2 Minimum Distance Classifier

The construction of a *minimum distance classifier* is based on a human mechanism of recognizing objects, phenomena, etc. If we are to assign an unknown object to one of a few categories, we usually assign it to a category containing an object, which is similar to the unknown one. Let us assume that for a set of classes $\omega^1, \omega^2, \dots, \omega^C$ there exists a set of *reference (template) patterns/vectors*[9]:

$$\mathbf{R}^1, \mathbf{R}^2, \dots, \mathbf{R}^C. \tag{10.3}$$

In case clusters corresponding to these classes are *regular*, we can assume that a vector computed as the mean (median, mode) vector of the cluster is the reference pattern (cf. Fig. 10.1b).[10]

Now, we can begin classification. If an unknown pattern $\mathbf{X}$ appears, then we should measure its features and place the corresponding feature vector in the feature space (cf. Fig. 10.1b—an unknown pattern is marked by a triangle with a question mark). Then, a minimum distance classifier computes the distances between the unknown pattern and the reference patterns, i.e.,

$$\rho(\mathbf{X}, \mathbf{R}^1),\ \rho(\mathbf{X}, \mathbf{R}^2),\ \dots\ ,\ \rho(\mathbf{X}, \mathbf{R}^C), \tag{10.4}$$

where $\rho(\mathbf{X}, \mathbf{R}^j)$, $j \in \{1, 2, \dots, C\}$, is the distance between the pattern $\mathbf{X}$ and the reference pattern $\mathbf{R}^j$. Finally, the classifier assigns the pattern $\mathbf{X}$ to the class ω^L containing the reference pattern $\mathbf{R}^L$ which is the nearest to the pattern $\mathbf{X}$, i.e.,

$$\rho(\mathbf{X}, \mathbf{R}^L) = \min\{\rho(\mathbf{X}, \mathbf{R}^1), \rho(\mathbf{X}, \mathbf{R}^2), \dots, \rho(\mathbf{X}, \mathbf{R}^C)\}, \tag{10.5}$$

where the function min selects the smallest element from a set.

[9] Later we equate a pattern with its representation in the form of a feature vector.

[10] In our "fish example" a reference pattern corresponds to a fish of the mean length and the mean weight in a given class.

According to rule (10.5), a classifier assigns the unknown pattern to class ω^1 in Fig. 10.1b, because the distance between this pattern and the reference pattern $\mathbf{R}^1$ is smaller than the distance between this pattern and the reference pattern $\mathbf{R}^2$, which represents the second class.[11] (Distances are marked with a dashed line.)

If we use a minimum distance classifier, or other methods which compute distances, then the choice of an adequate metric is a crucial issue. For various problems the way of computing a distance influences the accuracy of the method. This issue is discussed in Appendix G.2.

10.3 Nearest Neighbor Method

Sometimes determining reference patterns which represent clusters in an adequate way is troublesome. It is very difficult if clusters are not *regular*, for example if they are *dispersed* and *scattered* in some directions. In such a case we can apply the *Nearest Neighbor, NN*, method. The main idea of the method was introduced by Evelyn Fix[12] and Joseph L. Hodges, Jr[13] in 1951 [91], then it was characterized by Thomas M. Cover[14] and Peter E. Hart[15] in 1967 [60]. In this method we compute the distances between an unknown pattern $\mathbf{X}$ and all the vectors of a learning set U. Then, we select the class containing the pattern, which is the nearest to the pattern $\mathbf{X}$. In the example shown in Fig. 10.1c we assign the unknown pattern $\mathbf{X}$ to the class ω^1, because this class contains the nearest neighbor of $\mathbf{X}$. (In Fig. 10.1c the distance between $\mathbf{X}$ and the nearest pattern belonging to the class ω^2 is also marked.) The NN method has an intuitive interpretation. If we meet an unknown object (event, phenomenon) and we want to classify it, then we can look for a resemblance to a similar object (event, phenomenon) and assign the unknown object to a class including this similar object.

The NN rule can be defined formally in the following way. Let U^k denotes the subset of the learning set including only those patterns that belong to the class ω^k, i.e.,

$$U^k = \{\mathbf{X}^j \ : \ (\mathbf{X}^j, u^j) \in U \text{ and } u^j = \omega^k\}. \tag{10.6}$$

[11]In our fish example this means that an unknown fish corresponding to an unknown pattern in Fig. 10.1b is classified as a sprat (ω^1), because it resembles the "reference sprat" $\mathbf{R}^1$ more than the "reference eel" $\mathbf{R}^2$. (That is, it is nearer to the "reference sprat" in the feature space.).

[12]Evelyn Fix—a professor of statistics of the University of California, Berkeley, a Ph.D. student and then a principal collaborator of the eminent Polish-American mathematician and statistician Jerzy Spława-Neyman, who introduced the notion of a confidence interval (also, the Neyman-Pearson lemma).

[13]Joseph Lawson Hodges, Jr—an eminent statistician (Hodges-Lahmann estimator, Hodges' estimator) at the University of California, Berkeley, a Ph.D. student of Jerzy Spława-Neyman.

[14]Thomas M. Cover—a professor at Stanford University, an author of excellent papers concerning models based on statistics and information theory.

[15]Peter E. Hart—a professor of the Stanford University, a computer scientists (a co-author of a heuristic search method A^* and the model based on the Hough transform).

Then, $\mathbf{X}$ is assigned to the class ω^L, if

$$\rho(\mathbf{X}, \mathbf{X}^r) = \min_{j=1,\dots,M} \{\rho(\mathbf{X}, \mathbf{X}^j)\} \text{ and } \mathbf{X}^r \in U^L. \tag{10.7}$$

In practice a learning set can contain patterns characterized by values of features which have been measured in an erroneous way. Then, the NN method could give an invalid classification if a pattern with erroneous values of features is the nearest neighbor. In order to eliminate such an effect, we apply the *k-Nearest Neighbor, k-NN*, method. In this method we identify not one nearest neighbor, but the k nearest neighbors. (k is a small odd number.) Then, we check which class possesses the biggest number of representatives in the group of the nearest neighbors. An unknown pattern is assigned to this class. Let us consider the example shown in Fig. 10.1d. Two patterns belonging to the class ω^2 are placed near the class ω^1. (Their features have likely been measured in an erroneous way.) If we apply the NN method, then we assign the unknown pattern to the class ω^2 incorrectly. However, if we use the 5-NN method, i.e., $k = 5$, then in the group of the five nearest neighbors three of them belong to the class ω^1, and the unknown pattern is assigned to this class correctly.

10.4 Decision-Boundary-Based Classifiers

Instead of using reference patterns or elements of a learning set, we can try to construct boundaries which divide the feature space into subspaces corresponding to classes. Such an idea was originally introduced by Ronald A. Fisher in 1936 [90]. It is used for defining *decision-boundary-based classifiers*.

Let us begin by considering the case of two classes in a feature space which can be separated in a linear way, i.e. they can be separated by a boundary which is defined with the help of a linear function called a *linear discriminant function*. Let clusters containing elements of a learning set and representing classes ω^1 and ω^2 be placed in a feature space as shown in Fig. 10.2a. These clusters can be separated by a boundary. The points $\mathbf{X} = (X_1, X_2)$ belonging to the boundary fulfill the following equation:

$$d(\mathbf{X}) = 2X_1 - X_2 - 4 = 0. \tag{10.8}$$

Let us notice that all patterns of the learning set $\mathbf{X}$ which belong to the class ω^1 fulfill the following condition:

$$d(\mathbf{X}) > 0, \tag{10.9}$$

and all patterns of the learning set $\mathbf{X}$ which belong to the class ω^2 fulfill the following condition:

$$d(\mathbf{X}) < 0. \tag{10.10}$$

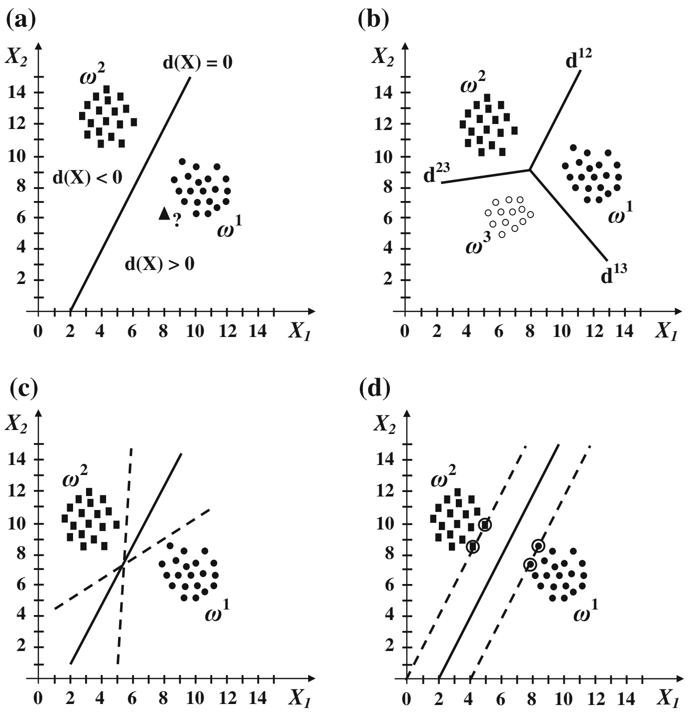

Fig. 10.2 **a** An example of a decision boundary which separates two classes, **b** a separation of three classes by three decision boundaries, **c** possible boundaries between two classes, **d** an example of the Support Vector Machine method

The function d is used for classifying unknown patterns. For example, the unknown pattern marked with a triangle in Fig. 10.2a, which has coordinates $X_1 = 8$, $X_2 = 6$ is assigned to the class ω^1, because the value of the function d for this pattern is greater than zero, according to formula (10.8).

In the general case of an n-dimensional feature space, a linear discriminant function is of the following form:

$$d(\mathbf{X}) = \sum_{i=1}^{n} W_i X_i + W_0, \tag{10.11}$$

where $\mathbf{W} = (W_1, \dots, W_n)$ is called the *weight vector* and W_0 is the *threshold weight*. This function corresponds to a boundary which is a hyperplane.

If there are more than two classes, then we can partition the feature space with the help of many boundaries in such a way that classes are separated *pairwise*. An example of such a dichotomous approach is shown in Fig. 10.2b. The class ω^1 is separated from the class ω^2 with the help of the discriminant function d^{12}, the class ω^1 is separated from the class ω^3 with the help of the discriminant function d^{13}, and the class ω^2 is separated from the class ω^3 with the help of the discriminant function d^{23}.

In the case of the linear discriminant function method we can define a lot of boundaries which separate clusters. Let us look at Fig. 10.2c, in which three boundaries separating clusters which represent classes ω^1 and ω^2 are defined. Although all boundaries are correct, the two boundaries marked by dashed lines seem to be *worse* than the third boundary. Our observation is accurate, because in the cases of the boundaries marked by dashed lines a small shift of some cluster points makes it necessary to modify these boundaries. However, the boundary marked with a solid line runs sufficiently far from both clusters. This means that it is less sensitive to some shifts of patterns in the feature space. This observation inspired Vladimir Vapnik[16] to define *Support Vector Machine, SVM*, in 1979 [308]. The main idea of this method is shown in Fig. 10.2d.

The Support Vector Machine looks for a boundary between two clusters which is placed in such a way that its distance from both clusters is equal and as big as possible. (In Fig. 10.2d it is marked with a solid line.) In order to determine such a boundary, we construct two *support hyper-planes*, which are parallel one to another and have the same distance from the boundary. (They are marked by dashed lines in Fig. 10.2d.) Each hyperplane is *supported* on the elements of its cluster which are protruding the most towards another cluster. These elements are marked by circles. Since these elements are the feature vectors that the hyper-planes are supported on, they are called *support vectors*.[17]

Till now, we have assumed that clusters in a feature space can be separated by linear discriminant functions. If they cannot be separated by linear functions, we have to use non-linear discriminant functions and define *non-linear classifiers*. However, constructing efficient non-linear classifiers is very difficult. In the 1990s some efficient non-linear classifiers were defined [35, 92, 259]. An alternative approach consists of using *splines*, i.e., piecewise-defined functions of smaller order.

10.5 Statistical Pattern Recognition

In *statistical pattern recognition (Bayesian approach)*, presented by Richard O. Duda[18] and Peter E. Hart in [78], the probability[19] of assigning a pattern to a class is taken into consideration. These methods are based on the Bayesian model.[20]

[16]Vladimir Naumovich Vapnik—a professor at the Institute of Control Sciences, Moscow from 1961 to 1990, then at AT&T Bell Labs and NEC Laboratories, Princeton. His work concerns mainly statistics and Artificial Intelligence (Vapnik–Chervonenkis theory).

[17]This is why the method is called *Support Vector Machines*.

[18]Richard O. Duda—a professor of electrical engineering at San Jose State University. His achievements concern pattern recognition. He defined the Hough transform. He is a co-author of the excellent monograph "Pattern Classification and Scene Analysis".

[19]The basic notions of probability theory are introduced in Appendices I.1, B.1 and I.2.

[20]Thomas Bayes—an eminent English mathematician and a Presbyterian minister. The "father" of statistics.

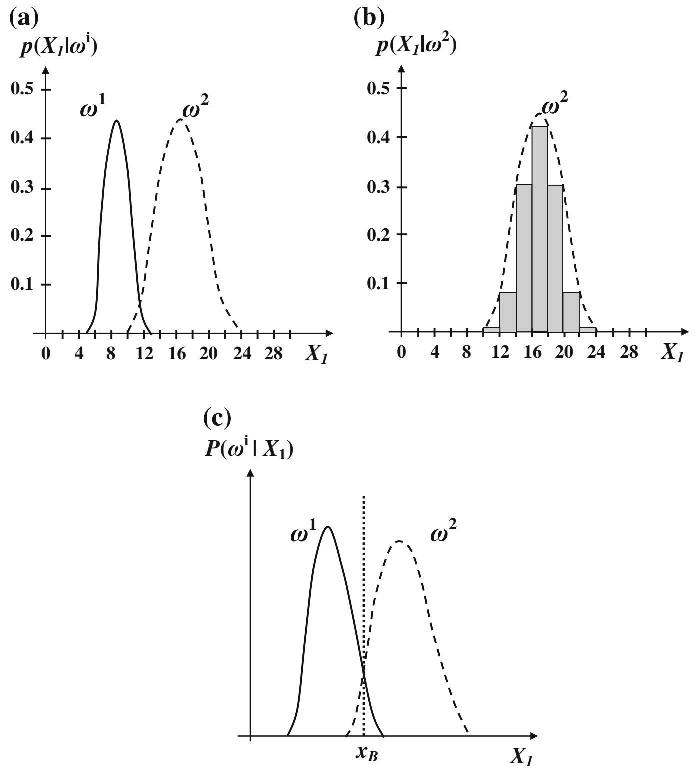

Fig. 10.3 **a** Examples of probability density functions for two classes, **b** a reconstruction of a probability density function on a basis of a histogram, **c** an example of a decision rule for the Bayesian classifier

In order to simplify our considerations, let us assume that all patterns belong to one of two classes in a one-dimensional feature space. Let us assume that for classes ω^1 and ω^2 we know the a priori probabilities $P(\omega^1)$ and $P(\omega^2)$. The a priori probability $P(\omega^1)$ gives the probability that a pattern belongs to the class ω^1 in general.[21]

Then, let us assume that for this (single) feature X_1 characterizing our patterns we know probability density functions $p(X_1|\omega^1)$ and $p(X_1|\omega^2)$ for both classes. The probability density function $p(X_1|\omega^1)$ is a function which for a given value of the variable X_1 assigns the probability of its occurrence, assuming that we say about a pattern which belongs to the class ω^1. Let us return to our "fish example". Now, ω^1 means the class of *sprats*, ω^2 means the class of *anchovy*, and the feature X_1 means the *length of a fish*. Then, $p(9|\textit{sprat}) = 0.42$ means that the probability that a fish is 9 cm long, assuming that it is a *sprat*, is equal to 0.42. For example, probability density functions for two classes are shown in Fig. 10.3a. As we can see, values of

[21]We may know, for example, that there are four times more patterns belonging to the class ω^1 than patterns belonging to the class ω^2 in nature. Then, $P(\omega^1) = 4/5$ and $P(\omega^2) = 1/5$.

the feature X_1 of patterns belonging to the class ω^1 belong to the interval [5, 13]. Values around 9 are the most probable (the probability more than 0.4). Values of the feature X_1 of patterns belonging to the class ω^2 belong to the interval [10, 24]. Values around 17 are the most probable.

If we use the Bayesian approach, the question is how to determine the probability density function for a given class? The simplest technique consists of using a histogram, which shows the empirical distribution of a feature. This is shown in Fig. 10.3b.[22]

If we have defined a priori probabilities for classes ω^1 and ω^2, and probability density functions for these classes, we can define a posteriori probabilities $P(\omega^1|X_1)$ and $P(\omega^2|X_1)$, according to Bayes' rule:

$$P(\omega^j|X_1) = \frac{p(X_1|\omega^j)P(\omega^j)}{\sum_{k=1}^{2} p(X_1|\omega^k)P(\omega^k)}\ , \quad j = 1, 2. \tag{10.12}$$

$P(\omega^1|X_1)$ is the probability that an unknown pattern belongs to the class ω^1, depending on the value of its feature X_1. Thus, for our example $P(\mathit{sprat}|7) = 0.35$ is interpreted in the following way. The probability that a fish of length 7 cm is a *sprat* equals 0.35. Let us notice that we can omit the denominator of formula (10.12), because it is the same for both classes.

Example graphs of a posteriori probability functions for both classes depending on the feature X_1 are shown in Fig. 10.3c. These graphs cross for the value x_B. This means that for values of the feature X_1 greater than x_B the probability that a pattern belongs to the class ω^2 is greater than the probability that the pattern belongs to the class ω^1 (and vice versa). As we can see, there are values of X_1 for which the probabilities of belonging to both classes are non-zero. There are also values of X_1 for which the probability of belonging to a given class is equal to zero.

We can generalize our considerations to the case of more than two classes. For classes $\omega^1, \omega^2, \ldots, \omega^C$ formula (10.12) is of the following form:

$$P(\omega^j|X_1) = \frac{p(X_1|\omega^j)P(\omega^j)}{\sum_{k=1}^{C} p(X_1|\omega^k)P(\omega^k)}\ , \quad j = 1, 2, \ldots, C. \tag{10.13}$$

Now, we can formulate a rule for recognizing an unknown pattern characterized by one feature $\mathbf{X} = (X_1)$ with the help of the *Bayes classifier*. The classifier assigns the pattern to that class for which the a posteriori probability is the biggest. Thus, $\mathbf{X}$ is assigned to the class ω^L, if

$$P(\omega^L|X_1) > P(\omega^j|X_1) \text{ for each } j \in \{1, 2, \ldots, C\}\ , \ j \neq L. \tag{10.14}$$

[22]The height of the bar for an interval $[a, b]$ should be $h = p/w$, where p is the number of elements of the learning set which belong to the given class and are in the interval $[a, b]$, and w is the number of all elements of the learning set which belong to the given class.

In case we would like to recognize patterns in an n-dimensional feature space, i.e., $\mathbf{X} = (X_1, X_2, \ldots, X_n)$, we can use the so-called *naive Bayes classifier*. We make the assumption that all the features are independent. Then, the probability density function for an n-dimensional feature vector is defined in the following way:

$$p(\mathbf{X}|\omega^j) = \prod_{i=1}^{n} p(X_i|\omega^i). \tag{10.15}$$

We can generalize the Bayes classifier even more. We can assume that erroneous decisions concerning the recognition of an unknown pattern can have various costs, i.e., they have various consequences. Then, we introduce a function of the cost (of errors). This function and the a posteriori probability are the basis for defining the risk function for the classifier. In this approach we try to minimize the risk function.

10.6 Decision Tree Classifier

The pattern recognition methods introduced in the previous sections belong to a *one-stage approach*, in which we make the classification decision taking into account all classes and all features in one step. However, the classification process can be decomposed into a sequence of steps. In subsequent steps we can analyze successive features with respect to various subsets of classes. Such an approach is called a *multistage (sequential) approach*. The *decision tree classifier* introduced by J. Ross Quinlan[23] in 1979 [233] is one of the most popular methods belonging to this approach. Let us introduce it with the help of the following example.

Let us assume that we construct a classifier recognizing the creditworthiness of a customer in a bank. We take into account two features of a customer: X_1 = *Income* (yearly) and X_2 = *Debt* (of the customer to the bank). We assume two classes: ω^1 = *creditworthy customers* and ω^2 = *non-creditworthy customers*. Grouping of elements of the learning set into two clusters is shown in Fig. 10.4. After analyzing these clusters, i.e., analyzing the behavior of customers belonging to the corresponding classes, we decide to divide the feature space with the help of a boundary which separates customers with a yearly income more than 50,000 € from those who have a lower income. Defining this boundary corresponds to constructing the part of the decision tree shown in Fig. 10.4a. The condition "*Income* > 50,000" which defines the threshold is written into a node of the tree. If the condition is fulfilled, then an unknown pattern belongs to the class ω^1 (marked with a circle in the decision tree and in the feature space). Further analysis allows us to divide the feature space according to the feature *Debt* and to set a threshold of 100,000 €. Customers whose debt is greater than this threshold belong to the class ω^2 (marked with a rectangle

[23]John Ross Quinlan—an Australian computer scientist, a researcher at the University of Sydney and the RAND Corporation. His research concerns machine learning, decision theory, and data exploration.

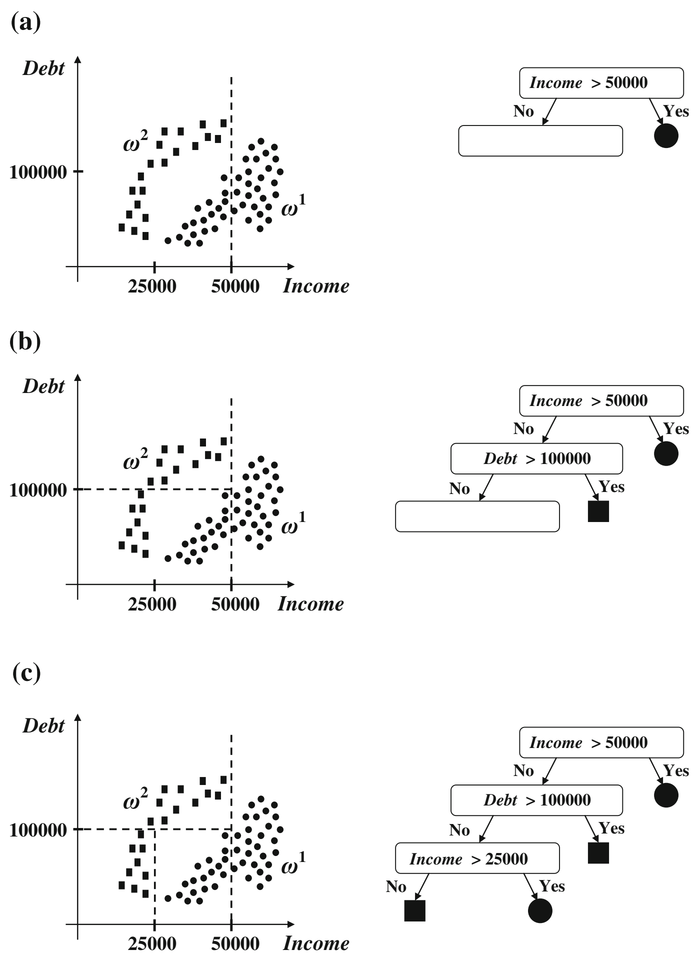

Fig. 10.4 Successive steps constructing a decision tree and partitioning a feature space

in the decision trees and in the feature space). Such a division of the feature space corresponds to developing the decision tree as shown in Fig. 10.4b. Finally, in order to separate customers of the two classes in the lower left subspace we should set a threshold equal to 25,000 € for the feature *Income*. As we can see in Fig. 10.4c this threshold separates creditworthy customers from the non-creditworthy ones in this subspace. We obtain the decision tree shown in Fig. 10.4c as a result.

Summing up, a classifier based on a decision tree divides a feature space with the help of boundaries which are parallel to the axes of the coordinate system that define the feature space. These boundaries are constructed by the sequential identification of thresholds for specific features.

10.7 Cluster Analysis

When we have formulated a pattern recognition task in previous sections, we have assumed that we have a learning set, which consists of patterns with their assignments to proper classes. We have introduced various classifiers, which assign an unknown pattern to a proper class. *Cluster analysis* is a problem which can be considered complementary to pattern recognition. We assume here that we have a set of sample patterns, however we do not know their classification. A cluster analysis task consists of grouping these patterns into clusters, which represent classes. Grouping should be done in such a way that patterns belonging to the same cluster are *similar* to one another. At the same time, patterns which belong to distinct clusters should be *different* from one another.

Thus, the notion of *similarity* is crucial in cluster analysis. Since patterns are placed in a feature space, as in pattern recognition, we use the notion of metric[24] also in this case. We compute distances between patterns of a sample set with the help of a given metric. If the distance between two patterns is small, then we treat them as similar (and vice versa).

In general, cluster analysis methods are divided into the following two groups:

- *methods based on partitioning*, where we assume that we know how many clusters should be defined, and
- *hierarchical methods*, where the number of clusters is not predefined.

K-means clustering is one of the most popular methods based on partitioning. The idea of the method was introduced by Hugo Steinhaus[25] in 1956 [287] and the algorithm was defined by James B. MacQueen[26] in 1967 [190]. Firstly, let us introduce the notion of the *centroid of a cluster*. The centroid is the mean of the positions of all patterns which belong to a given cluster. Let us assume that we want to group patterns into k clusters. The method can be defined in the following way.

1. Select k initial centroids of clusters. (The selection can be made by random choice of k patterns as initial centroids or by random choice of k points in the feature space.)
2. Assign each pattern of the sample set to a cluster on the basis of the smallest distance between the pattern and the cluster centroid.
3. For clusters created in Step 2. compute new centroids.
4. Repeat Steps 2. and 3. until clusters are stabilized. (We say that clusters are stabilized, if pattern assignments do not change in a successive step (or changes

[24]Various metrics are introduced in Appendix G.2.

[25]Hugo Steinhaus—a Polish mathematician, a professor of Jan Kazimierz University in Lwów (now Lviv, Ukraine) and Wrocław University, a Ph.D. student of David Hilbert, a co-founder of the Lwów School of Mathematics (together with, among others, Stefan Banach and Stanisław Ulam). His work concerns functional analysis (Banach-Steinhaus theorem), geometry, and mathematical logic.

[26]James B. MacQueen—a psychologist, a professor of statistics at the University of California, Los Angeles. His work concerns statistics, cluster analysis, and Markov processes.

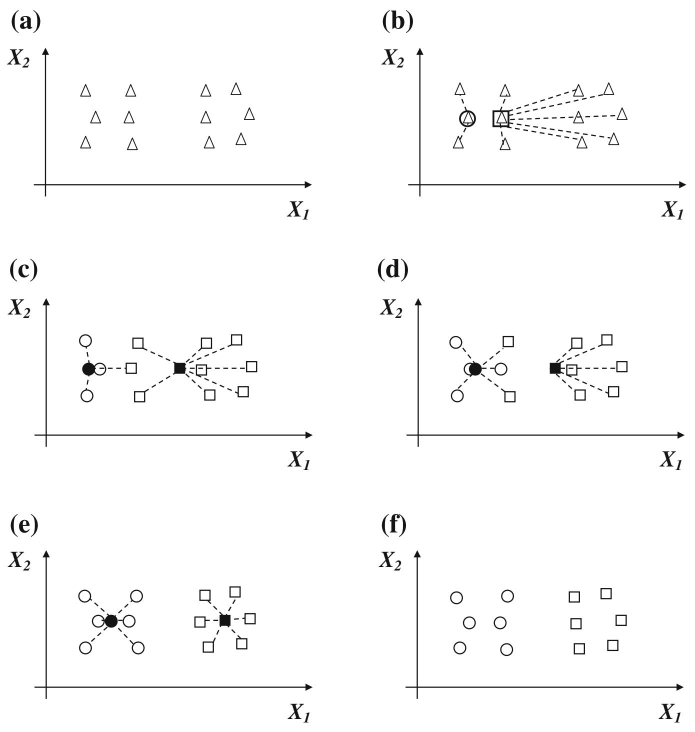

Fig. 10.5 Successive steps of k-means algorithm

are below a certain threshold) or centroids do not change (or changes are below a certain threshold).)

Let us consider an example of the k-means algorithm, which is shown in Fig. 10.5. A placement of patterns in a feature space is shown in Fig. 10.5a. Let us assume that we would like to group the patterns into two clusters, i.e., $k = 2$. (Elements of these clusters are marked either with circles or rectangles.) We assume that we have selected cluster centroids randomly as shown in Fig. 10.5b. The distance between the two leftmost patterns and the centroid of the "circle" cluster is smaller than the distance between these patterns and the centroid of the "rectangle" cluster. Therefore, they are assigned to this cluster (cf. Fig. 10.5b). The remaining patterns are assigned to the "rectangle" cluster, because they are closer to its centroid than to the centroid of the "circle" cluster. (Assignments of patterns to centroids are marked with dashed lines.) After that we compute new centroids for both clusters. We have marked them with a black circle (the first cluster) and a black rectangle (the second cluster). As we can see in Fig. 10.5c, the centroid of the "rectangle" cluster has moved to the right significantly and the centroid of the "circle" cluster has moved to the left a little bit. After setting the new centroids we assign patterns to clusters anew, according to the closest centroid. This time the "circle" cluster absorbs one pattern which was a "rectangle" previously. (It was the initial centroid of "rectangles".) In Fig. 10.5d we can see the movement of both centroids to the right and the absorption of two "rectangle" patterns by the "circle" cluster. The final placement of the centroids is shown in Fig. 10.5e. After this both clusters are stabilized. The effect of the grouping is shown in Fig. 10.5f.

The idea of *hierarchical cluster analysis* was defined by Stephen C. Johnson[27] in 1967 [150]. In such an approach we do not predefine the number of clusters. Instead of this, we show how clusters defined till now can be merged into bigger clusters (*an agglomerative approach*[28]) or how they can be decomposed into smaller clusters (*a divisive approach*[29]).

Let us assume that a sample set consists of M patterns. The scheme of an agglomerative method can be defined in the following way.

1. Determine M initial clusters, which consist of a single pattern. Compute the distances between pairs of such clusters as the distances between their patterns.
2. Find the nearest pair of clusters. Merge them into one cluster.
3. Compute distances between this newly created cluster and the remaining clusters.
4. Repeat Steps 2. and 3. until one big cluster containing all M patterns is created.

An agglomerative scheme is shown in Fig. 10.6. (Successive steps are shown from left to right.) The feature space is one-dimensional (a feature X_1). Firstly, we merge the first two one-element clusters (counting from the top), because they are the nearest to each other. In the second step we merge the next two one-element clusters. In the third step we merge the second two-element cluster with the last one-element cluster, etc. Let us notice that if clusters contain more than one element, we should define a method to compute a distance between them. The most popular methods include:

- the *single linkage method*—the distance between two clusters A and B is computed as the distance between the two nearest elements E_A and E_B belonging to A and B, respectively,
- the *complete linkage method*—the distance between two clusters A and B is computed as the distance between the two farthest elements E_A and E_B belonging to A and B, respectively,
- the *centroid method*—the distance between two clusters A and B is computed as the distance between their centroids.

Successive steps of a divisive scheme are shown in Fig. 10.6 from right to left.

[27] Stephen Curtis Johnson—a researcher at Bell Labs and AT&T, then the president of USENIX. A mathematician and a computer scientist. He has developed cpp—a C language compiler, YACC—a UNIX generator of parsers, int—a C code analyzer, and a MATLAB compiler.

[28] In this case we begin with clusters containing single patterns and we can end up with one big cluster containing all patterns of a sample set.

[29] In this case we begin with one big cluster containing all patterns of a sample set and we can end up with clusters containing single patterns.

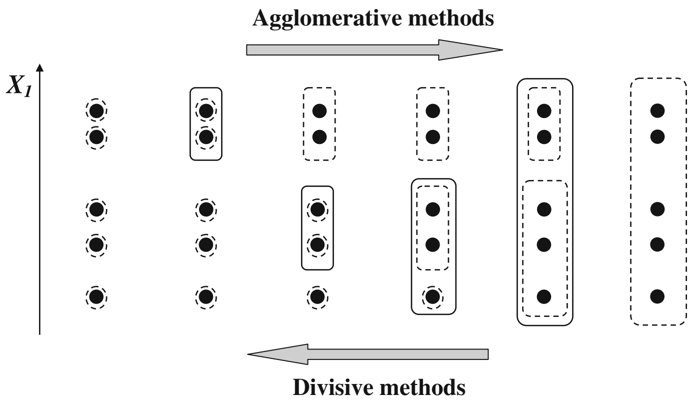

Fig. 10.6 Agglomerative methods and divisive methods in hierarchical cluster analysis

Bibliographical Note

Monographs [28, 78, 79, 106, 171, 309] are good introductions to pattern recognition.
Cluster analysis methods are presented in [4, 85, 127].