Chapter 3
Machine Learning

Abstract Machine learning relates to the study, design and development of algorithms that give computers the capability to learn without being explicitly programmed. Machine learning techniques are fairly generic and can be applied in various settings. To utilize such kinds of algorithms, one has to translate the problem to the domain of machine learning, which usually expects a set of features and a desirable output or grouping criterion. In this chapter, we introduce the three machine learning paradigms often employed by the literature: supervised, unsupervised, and semi-supervised. We show that supervised algorithms exclusively utilize external information to induce or to train their hypotheses. In contrast, unsupervised learning methods are guided exclusively by the intrinsic structure of the data items throughout the learning process, i.e., without any sort of external knowledge. In-between these two learning paradigms lies the semi-supervised learning, which employs both the labeled and unlabeled data in the learning process. Here, we focus on supplying the shortcomings and potentialities of traditional and representative techniques that are well-known by the machine learning community. We will not go into technical details of traditional machine learning techniques in this chapter, because these are not the focus of this book.

3.1 Overview of Machine Learning

Machine learning aims at developing computational methods that are capable of "learning" with accumulated experiences [9, 19, 36, 43–45]. Traditionally, there are two fundamental types of learning in machine learning. The first is entitled *unsupervised learning*, whose main task consists in revealing intrinsic structures that are embedded within the data relationships. The learning process, in this case, is solely guided by the provided data, for no prior knowledge about the data is supplied [38, 44, 46]. Essentially, a typical problem in unsupervised learning consists in estimating the underlying density function that generated the data distribution under analysis [10]. Among the main tasks of unsupervised learning, one can highlight: clustering [23, 33, 48, 49], outlier detection [40, 41], dimensionality reduction [39], and association [53]. In a clustering task, we expect to find groups in which data items in the same group are very similar to each other, while data items that correspond to different groups are expected to be dissimilar. In this case, the

T.C. Silva, L. Zhao, *Machine Learning in Complex Networks*,
DOI 10.1007/978-3-319-17290-3_3

resemblance or similarity of different data items is judged according to an adopted similarity function [44]. In outlier detection, the goal is to find data items that differ, to a large extent, from the majority of the other data items, i.e., from the original data distribution [40]. In dimensionality reduction, the objective is to dispose the data items over a lower dimensional space in relation to its original distribution, so as to simplify the relationships among the data items [39]. In association, one seeks to generate rules that relate subsets of predictive attributes [53].

The second type of learning is referred to as *supervised learning*, whose objective is to deduce concepts regarding the data. This inference is performed using the presented labeled instances, which are commonly denoted as the training set. In this regard, the learning process tries to construct a mapping function conditioned to the provided training set [1, 26, 31, 36]. Often, the learners are tested using unseen data items that compose the so-called test set. When the labels comprise discrete values, then the problem is denominated *classification*, whereas when the values are continuous, *regression* [9].

The main difference between supervised and unsupervised learning paradigms is as follows. In the first, the learner explores external information of the training set, which is available at the training stage, in order to induce the hypothesis of the classifier. In a classification task, for example, this external information is brought to the learning process in the form of classes or labels. The goal of classification, in this case, is to create a predictive function that can generalize from this training set, when applied to unknown data (test set). The performance of the classifier in this test is often termed as the classifier's generalization power. In contrast, the unsupervised learning paradigm seeks behaviors or trends in the data, trying to group them in a way that similar data tend to be agglomerated together while dissimilar data are segregated into distinct groups. Note that, in this case, no external information or labels are employed during the learning process.

Besides these two well-defined areas, a new machine learning paradigm has received attention from the related community, which is denominated *semi-supervised learning*. This paradigm has been proposed in order to combine the strengths of the supervised and unsupervised learning paradigms [10, 64, 65]. In a typical semi-supervised classification, few data are labeled, while most of them are unlabeled. Observe that this corresponds to the typical scenario nowadays, as thousands of data are generated very quickly, while only a few of them can effectively be processed and labeled. This fact is true because, in general, the labeling task is expensive, time-consuming, and prone to errors. In the semi-supervised learning, the goal is to propagate these few labels from the labeled examples to the large amount of unlabeled examples. A semi-supervised task, therefore, utilizes both information from the training and the test sets to make predictions in a simultaneous manner.

For didactic purposes, Fig. 3.1a shows a clustering task in unsupervised learning. Note that no external information is available *a priori*, and the clustering is performed by using similarity or topological information of the data. In Fig. 3.1b, it is illustrated a semi-supervised classification in semi-supervised learning. Note now that some data already possess labels (external information) beforehand, while most

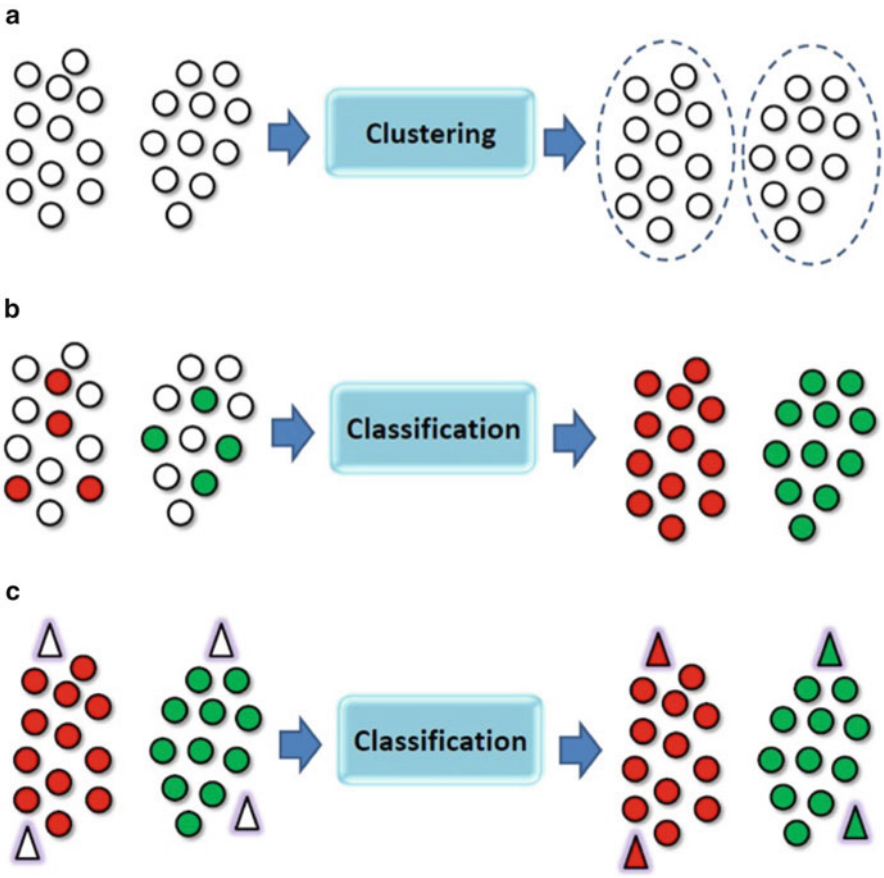

Fig. 3.1 Schematic of the three paradigms of the machine learning area. (**a**) Unsupervised learning; (**b**) Semi-supervised learning; (**c**) Supervised learning

of them are unlabeled. The learning process propagates these labels to the remaining unlabeled data. Finally, in Fig. 3.1c, a classification process in supervised learning is displayed. Initially, the classifier is trained by solely using information from the training set, which is always fully labeled. In the next phase, called classification phase, the classifier is used to predict class labels of unseen data items.

For completeness, it is worth mentioning a novel machine learning area that is not subject of study in this book. Deep learning is a parallel branch of machine learning that relies on sets of algorithms that attempt to model high-level abstractions in data by using model architectures, with complex structures composed of multiple non-linear transformations [18, 57]. Deep learning is part of a broader family of machine learning methods based on learning representations of data. An observation (e.g., an image) can be represented in many ways such as a vector of intensity values per pixel, or in a more abstract way as a set of edges, regions of particular shape, etc.

Some representations make it easier to learn tasks (e.g., face recognition or facial expression recognition) from examples. One of the promises of deep learning is replacing handcrafted features with efficient algorithms for unsupervised or semi-supervised feature learning and hierarchical feature extraction.

In the next sections, we give an overview of the supervised, unsupervised, and semi-supervised learning paradigms.

3.2 Supervised Learning

Algorithms that exclusively utilize external information to induce or to train their hypotheses are considered supervised learning methods. In this chapter, we supply definitions and reviews on traditional supervised learning methods presented in the literature. In Chap. 5, we revisit the supervised learning paradigm with a focus on network-based methods.

3.2.1 Mathematical Formalization and Fundamental Assumptions

The mathematical formulation of a supervised learning task is defined as follows [9, 38, 44, 62]. Let $\mathscr{X}_{\text{training}} = \{(x_1, y_1), \ldots, (x_L, y_L)\}$ denote the training set, where the first component of the i-th tuple $x_i = (x_{i1}, \ldots, x_{iP})$ denotes the attributes of the P-dimensional i-th training instance. The second component $y_i \in \mathscr{Y}$ characterizes the class label or target associated to that training instance. The training set is composed of $L = |\mathscr{X}_{\text{training}}|$ data items. The goal here is to learn a mapping $x \mapsto y$ using only $\mathscr{X}_{\text{training}}$, i.e., the training data distribution and the associated labels. To check the generalization performance of the trained model, the constructed classifier is checked against a test set $\mathscr{X}_{\text{test}} = \{x_{L+1}, \ldots, x_{L+U}\}$, for which labels are not provided. The test set is composed of U data items. Each data item in that set is termed as test instance. For an unbiased learning, the training and test sets must be disjoint, i.e., $\mathscr{X}_{\text{training}} \cap \mathscr{X}_{\text{test}} = \emptyset$. Usually, $N = L + U$ denotes the total number of data items in the learning process.

In the supervised learning scheme, there are two learning phases: the *training phase* and the *classification phase* [9, 26, 31, 38]. In the training phase, the classifier is induced or trained by using the training instances (labeled data) in $\mathscr{X}_{\text{training}}$ that are always fully labeled. In the classification phase, the labels of the test instances in $\mathscr{X}_{\text{test}}$ are predicted using the induced classifier.

Once trained, one must verify how well the trained supervised learner can generalize. Without any additional assumptions over the data properties, this problem cannot be solved exactly since unseen situations might have arbitrary output values. The necessary assumptions about the nature of the target function constitute the

inductive bias of the learner. The following is a list of common inductive biases in supervised learning algorithms:

- *Maximum conditional independence*: if the hypothesis can be cast in a Bayesian framework, usually an objective function that aims at maximizing the conditional independence is employed. This is the bias used, for example, in the Naïve Bayes classifier [45, 47].
- *Maximum margin*: when drawing a boundary between two classes, attempt to maximize the width of the boundary, i.e., the distance from the data items. This is the bias used, for instance, in support vector machines. The assumption is that distinct classes tend to be separated by wide low-density boundaries.
- *Minimum description length*: when forming a hypothesis, attempt to minimize the length of the description of the hypothesis. The assumption is that simpler hypotheses are more likely to be true. The rationale here is that complex hypotheses are likely to incorporate noise coming from the training data. Hence, the model becomes overfit to the training data, in a way that its generalization power is jeopardized. A classical principle of this type of inductive bias is the Occam's Razor, which conveys the idea that the simplest consistent hypothesis about the target function is actually the best. Here, consistent means that the hypothesis of the learner yields correct outputs for all of the training data examples that have been given to the algorithm.
- *Minimum number of features*: unless there is good evidence that a feature is useful, it should be deleted. This is the assumption behind feature selection algorithms. Note that if correlation exists between different features, it correspondingly increases the variance of the overall model. In this way, it is only useful to add new features if they explain orthogonal parcels of the output or target variable that have not been explained by other features.
- *Nearest neighbors*: relying on the smoothness or continuity assumption, this inductive bias assumes that, in most of the cases, data items in small neighborhoods tend to be quite similar. Given a test instance, for which the class is unknown, we infer that its class is the same as of the majority of data items in the local neighborhood. This is the bias used, for example, in the k-nearest neighbor algorithm. The assumption is that data items that are near each other tend to belong to the same class.

In machine learning, we can form different models or hypotheses for the same data set, depending on the nature and inductive bias of the selected algorithm. A natural problem that arises is of how well the classifier can perform on unseen data. For that, we need to employ error estimation techniques. Instead of employing the entire training set to train the supervised learner, the basic idea consists in further partitioning the training set into two sub-sets, where the first one is used to train the classifier and the second one is purposed to verify its performance. The performance can be estimated because we now can compare the output value of the algorithm with the ground truth, as we are dealing with "in-sample" or training data items, for which labels are known. The method of k-fold cross-validation is the

most employed error estimation procedure in the literature.[1] When trying to choose among different models, we select the hypothesis with the lowest cross-validation error. Although cross-validation may seem to be free of bias, the "no free lunch" theorem states that cross-validation must be biased. In cross-validation, we partition a data set into k equally sized non-overlapping subsets $\mathscr{F}$. For each fold $\mathscr{F}_i$, a model is trained on $\mathscr{F} \setminus \mathscr{F}_i$ (reduced training set), which is then evaluated on $\mathscr{F}_i$ ("in-sample" test instances). Another error estimation procedure is the leave-one-out, which is a special case of the k-fold cross-validation when $k = L - 1$, where L is the number of labeled data items in the training set. The cross-validation estimator of the prediction error is defined as the average of the prediction errors obtained on each fold. While there is no overlap between the test sets on which the models are evaluated, there is overlap between the training sets whenever $k > 2$. The overlap is largest for leave-one-out cross-validation. This means that the learned models are correlated, i.e., dependent, and the variance of the sum of correlated variables increases with the amount of covariance. Therefore, leave-one-out cross-validation has large variance in comparison to smaller k. However, remark that while twofold cross validation does not have the problem of overlapping training sets, it often also has large variance because the training sets are only half the size of the original sample. A good compromise often accepted by the literature is to use ten-fold cross-validation. For a thorough review on this topic, c.f. [3, 7, 31].

With regard to the training and test set characteristics, supervised learning algorithms often assume that [9, 44]:

- For a valid estimation process, the test set must not be biased toward the training set. It must, however, be sampled from the same data distribution process that generated the training set data. This assumption makes clear that, since the classifier has been trained in accordance with the training set distribution, it is fair enough that it is only capable of efficiently inferring unseen examples of that same data distribution. In practice, however, this assumption is often violated to a certain degree. Strong violations will clearly result in poor classification rates.
- The training set must be a representative sample of the distribution or population that generated the analyzed data. Since the hypothesis that the classifier induces is based upon the training set, if the available data is not a representative sample of the true underlying data distribution process, the classifier has a great chance of being mistrained. As such, it predicts in accordance with another distribution, that of the non-representative training set.

[1] If model selection is embedded within the error estimation procedure, then the nested k-fold cross-validation is frequently indicated. The nested cross-validation works as follows:

- An inner loop is used as part of model fitting procedure. Typically, we conduct a grid search over the parameters of the technique. For instance, in the k-nearest neighbors technique, we may run through several values of the parameter k.
- An outer loop is employed to measure the performance of the model that had the best performance on the inner loop on a separate external fold (but still in the training set).

3.2.2 Overview of the Techniques

Supervised learning techniques are divided into the following groups:

- *Decision trees*: A decision tree consists of vertices and branches that serve the purpose of breaking a set of samples into a set of covering decision rules. In each vertex, in its original inception, a single test or decision is made to obtain a partition. The starting vertex is usually referred to as the root vertex. In the terminal vertices or leaves, a decision is made on the class assignment. In each vertex, the main task is to select an attribute that makes the best partition between the classes of the samples in the training set [55].
- *Rule-based induction*: One of the most expressive and human readable representations for learned hypotheses are sets of IF-THEN rules. In these kinds of rules, the IF part contains conjunctions and disjunctions of conditions composed by the predictive attributes of the learning task, and the THEN part contains the predicted class for the samples that satisfy the IF clause [54].
- *Artificial neural networks*: Neural networks are interconnected groups of neurons that use mathematical or computational models for information processing based on a connectionist approach. In most cases, an artificial neural network is an adaptive system that changes its structure, usually represented by the connection weights between pairs of neurons, based on external or internal information that flows through the network. In more practical terms, neural networks are nonlinear statistical data modeling or decision making tools. They can be used to model complex relationships between inputs and outputs or to find patterns in data. The neural network, ignorant at the start, through a repetitive "learning" process, becomes a model of the dependencies between the descriptive variables and the target behavior. The key part in developing neural networks is to choose a suitable architecture (how many layers, thresholds utilized by the neurons, etc.) and a corrective learning algorithm (back-propagation, etc.) [27].
- *Bayesian networks*: Bayesian networks constitute a probabilistic framework for reasoning under uncertainty. From an informal perspective, Bayesian networks are directed acyclic graphs, where the vertices are random variables and the edges specify the dependence assumptions that must be held between different random variables. Bayesian networks are based upon the concept of conditional independence among variables. Once the network is constructed, it is used as an efficient device to perform probabilistic inference. This probabilistic reasoning inside the network can be carried out by exact methods, as well as by approximate methods [37]. A special case of Bayesian networks is when no dependencies on the predictive variables exist. In this case, the classifier is known as Naïve Bayes [47].
- *Statistical learning theory*: Maybe the most well-known technique of this type of learning is the support vector machines (SVM), which is based on the principle of structural risk minimization. Originally, it was worked out for linear two-class classification with margins, where margin means the minimal distance from the separating hyperplane to the closest data points. SVM seeks for an optimal

separating hyperplane, where the margin is maximal. An important and unique feature of this approach is that the solution is based only on those data points that are at the margin. These points are called support vectors. The linear SVM can be extended to a nonlinear one when first the problem is transformed into a feature space using a set of nonlinear basis functions. In the feature space—which can be very high dimensional—the data points can be separated linearly. An important advantage of the SVM is that it is not necessary to implement this transformation and to determine the separating hyperplane in the possibly very-high dimensional feature space. Instead, a kernel representation can be used, where the solution is written as a weighted sum of the values of a certain kernel function evaluated at the support vectors [59].

- *Instance-based learning*: Instance-based learning has its root in the study of the nearest neighbor algorithm. The simplest form of nearest neighbor or, more generally, k-nearest neighbors (k-NN) algorithms, simply stores the training instances and classifies a new instance by predicting that it has the same class as its nearest stored instance or the majority class of its k nearest stored instances, according to some similarity measure. The essence of this learning method resides in the form of the similarity function that computes the distances from the new test instance to the training instances [15].
- *Network-based methods*: The inference is done by means of a network constructed from the training set. Up to now, there are still few network-based supervised learning techniques [25]. In Chap. 5, we work on some representative network-based supervised learning techniques.

3.3 Unsupervised Learning

Unsupervised learning methods are guided exclusively by the intrinsic structure of the data items throughout the learning process, i.e., without any sort of external knowledge. In this chapter, we provide definitions and reviews on traditional methods presented in the literature. In Chap. 6, we revisit the unsupervised learning paradigm with a focus on network-based methods.

3.3.1 Mathematical Formalization and Fundamental Assumptions

The unsupervised learning can be defined as follows [2, 9, 21, 30, 44, 45]. Let $\mathscr{X} = \{x_1, x_2, \ldots, x_N\}$ be a data set, where N is the total number of data items involved in the learning process. Then, x_i is a P-dimensional vector, where each of the entries is called a feature or descriptor, which has the role to qualitatively or quantitatively describe the data item. Typically, it is assumed that the points are

independently and identically distributed in accordance with a common distribution. In the unsupervised learning case, no professors or external sources are used, i.e., no labels are provided throughout the learning process. Therefore, it is valid to say that no training phase is involved in this type of learning. As a consequence, the algorithm must be guided by the data distribution itself to infer useful knowledge or trends. For example, in data clustering or community detection tasks, the objective is to find subgroups of data items, such that the constituents or members of the same subgroup are more similar than those of different groups. The grouping is complete over the data, i.e., one seeks subgroups $\{\mathscr{X}_1, \ldots, \mathscr{X}_k\}$, in such a way that $\cup_{i=1}^{k} \mathscr{X}_i = \mathscr{X}$.

Clustering is a discovery process in data mining. It groups a set of data in a way that maximizes the similarity within clusters and minimizes the similarity between two different clusters. These discovered clusters can help in explaining the characteristics of the underlying data distribution and serve as the foundation for other data mining and analysis techniques. Clustering is useful in characterizing customer groups based on purchasing patterns, categorizing Web documents, grouping genes and proteins that have similar functionality, grouping spatial locations prone to earthquakes based on seismological data, and so on.

Most existing clustering algorithms find clusters that fit some static model. Although effective in some cases, these algorithms can break down if the analyst does not choose appropriate static-model parameters. Or, sometimes, the model simply cannot adequately capture the clusters' characteristics. Most of these algorithms break down when the data contains clusters of diverse shapes, densities, and sizes. For example, Fig. 3.2 shows some illustrative cluster shapes.

Clustering results differ according to the assumptions made about the data. Some of these forms include [9, 21]:

- *The underlying process that generated the observed data has some analytical form*: the methods are biased towards finding clusters with pre-defined forms (similar with estimation). The well-known algorithm *K*-Means, for instance, seeks circular-shaped clusters [42]. This bias has a strong impact on the final results of the unsupervised learner and thus limits its power of detection. These data assumptions, on one hand, may enhance the detection power of the learner if they really reflect the data properties. On the other hand, they severely hamper the learner's detection capabilities if they are invalid.
- *Neighborhood shares the same data characteristics*: the algorithms rely on local information to infer their decisions. For a given data item, for example, the *k*-NN clustering decides in accordance with some function on the neighborhood of that data item [26, 31].
- *Data are produced in some "hierarchical" organization*: the techniques that fall into this category are the hierarchical techniques, including the divisive and agglomerative ones. For practical methods of the former, the community detection based on continuously removing edges with the maximum betweenness [50] and the bisecting *K*-Means [34] are representative examples. As good examples

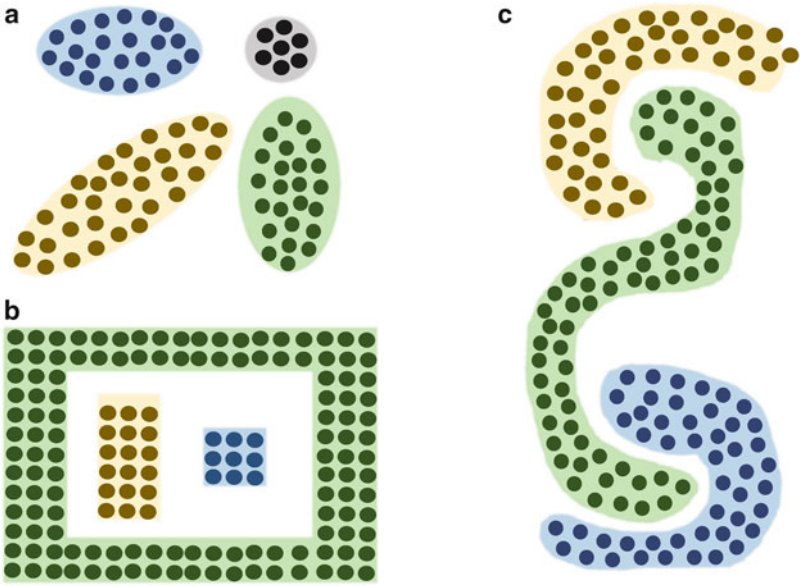

Fig. 3.2 Examples of clusters with different shapes. (**a**) Ellipsoidal clusters. (**b**) Quadrangular (hollow and opaque) clusters. (**c**) Concave (banana-shaped) clusters

of the latter, one can cite the simple-link [30, 44], complete-link [2, 30], and modularity greedy optimization [13].

- *Data fall into clusters according to some preferred directions*: the methods of this type deal with the transformation of the original data into a modified space, usually with a different number of dimensions. Examples include the principal component analysis (PCA) [32], independent component analysis (ICA) [32], and spectral graph algorithms [11].

3.3.2 Overview of the Techniques

One of the most common tasks of unsupervised learning is data clustering. Formally, data clustering aims at discovering natural groups. The groups may be defined as sets of patterns, points, or objects, all of which characterized by suitable similarity measures [9, 21, 29, 30, 63]. Each cluster is a collection of data items that are similar between them and are dissimilar to the objects belonging to other clusters. Data clustering is vital in several exploratory pattern-analysis, grouping, decision-making, and machine-learning situations. Some of them include data mining, document retrieval, image segmentation, bioinformatics, and pattern classification [12, 20, 28–30]. Unfortunately, in the majority of such tasks, only a little prior information is available about the data. In this way, advances in the methodology to automatically

understand, process, and summarize the data are required. Nowadays, this becomes even more critical by virtue of the exponential increase in both the volume and the variety of data [29, 63]. In this scenario, the decision-maker must perform as few assumptions about the data as possible. It is under these practical restrictions that the clustering procedure is specially appropriate for the exploration of inter-relationships among the data points to make assessments (perhaps preliminary) of their structure [29, 30]. Data clustering algorithms are generally divided into two types: hierarchical or partitional [8, 20, 29]. The former finds successive clusters using previously established clusters, whereas the latter determines all clusters at once. Hierarchical algorithms can be agglomerative ("bottom-up") or divisive ("top-down"). Agglomerative algorithms begin with each element as a separate cluster and merge them into successively larger clusters. Divisive algorithms begin with the whole set and proceed to divide it into successively smaller clusters. Two-way clustering, co-clustering, or bi-clustering are the names for clusterings where not only the objects are clustered but also the features of the objects, i.e., if the data is represented in a data matrix, the row and columns are clustered simultaneously [9, 21, 63]. In both approaches, the algorithms may be further categorized in network-based algorithms or non network-based algorithms.

In relation to partitional methods, several techniques have been studied in the literature [1, 26, 35, 63]. The most well-known one and the pioneer in the field is the K-Means method [42]. Even though it suffers from several caveats, such as the strong dependence on the system's initial conditions and the inherent bias to find only circular-shaped clusters, it has been further enhanced and studied by the community until today. Some methods, which have improved on the basic idea of K-Means, have been proposed, such as the K-Medoids and fuzzy C-Means [2, 21, 30]. Using a similar strategy, Clarans [51] also attempts to break a data set into K clusters such that the partition optimizes a given criterion, but it also assumes that clusters are hyper-ellipsoidal and of similar sizes. Hence, it also cannot find clusters that vary in size.

DBScan (Density-Based Spatial Clustering of Applications with Noise) [56], a well-known spatial and partitional clustering algorithm, can find clusters of arbitrary shapes. DBScan defines a cluster to be a maximum set of density-connected points, which means that every core point in a cluster must have at least a minimum number of points within a given radius. DBScan assumes that all points within genuine clusters can be reached from one another by traversing a path of density-connected points and points across different clusters cannot. DBScan can find arbitrarily shaped clusters if the cluster density can be determined beforehand and the cluster density is uniform [33].

With regard to hierarchical methods, the most well-known techniques are the single-link and the complete-link [1, 2]. Among other traditional techniques, we can also include average-link and the Ward methods [44]. These algorithms differ in the way the groups similarity is evaluated. In the single-link, for instance, the similarity is established by finding the minimum distance between pairwise data items of any two given groups. The complete-link technique, in contrast, finds the

similarity based on the maximum distance of two examples of different clusters, rather than the minimum [21].

CURE (Clustering Using Representatives) [24] represents a cluster by selecting a constant number of well-scattered points and shrinking them toward the cluster's centroid, according to a shrinking factor. CURE measures the similarity between two clusters by the similarity of the closest pair of points belonging to different clusters. Unlike centroid/medoid-based methods, CURE can find clusters of arbitrary shapes and sizes, as it represents each cluster via multiple representative points. Shrinking the representative points toward the centroid allows CURE to avoid some of the problems associated with noise and outliers. However, this technique fails to account for special characteristics of individual clusters. As such, it can make incorrect merge decisions when the underlying data does not follow the assumed model or when noise is present [33].

In general, hierarchical algorithms usually provide more information than partitional algorithms [30]. For instance, though susceptible to outliers, the single-link method is able to find groups very apart from each other, concentric groups, and chain-like groups in which the K-Means technique would have trouble with. However, hierarchical algorithms often take more time to process data items, which may invalidate their employment in large-scale data sets.

3.4 Semi-Supervised Learning

Algorithms that are able to learn using only a few labeled examples have aroused the interest of the artificial intelligence community. Among other features, semi-supervised learning aims at reducing the work of human experts in the labeling process. This feature is quite interesting especially when the labeling process is expensive and time consuming. This is often the case, for example, in video indexing, classification of audio signals, text categorization, medical diagnostics, genome data, among others [10, 65]. In this chapter, we introduce definitions and reviews on traditional methods presented in the literature. In Chap. 7, we revisit the semi-supervised learning paradigm with a focus on network-based methods.

3.4.1 Motivations

Semi-supervised learning is a new paradigm in relation to unsupervised and supervised learning. In this way, we first try to understand the reason behind its creation. From an engineering standpoint, it is clear that the collection of labeled data is much more intensive and costly in relation to that of unlabeled data. The main purpose of semi-supervised learning, nonetheless, goes way beyond a purely utilitarian tool. Most natural learning (human and animal) occurs in a semi-supervised regime basis. In the world in which we live, living beings are in a constant exposure to a flow of

natural stimuli. Such stimuli include unlabeled data that are easily noticeable. In the context of recognition and phonological acquisition, for example, a child is exposed to many perceptible acoustic sounds. Many of these sounds are not familiar to the child. A positive feedback by part of another person is the main source of labeled data. In many cases, a small amount of feedback is sufficient to allow the child to master the acoustic-phonetic mapping of any languages [5, 6].

Humans have ability to learn unsupervised concepts (for example, clusters and categories of objects), suggesting that unlabeled data could be satisfactorily used for learning natural invariance to form categories and to construct classifiers. In many pattern recognition tasks, humans only have access to small amounts of labeled patterns. Hence, the success of human learning in environments with little knowledge indubitably happens with effective use and manipulation of large sets of unlabeled data to extract information that is useful for generalization purposes. Consequently, if the goal is to know how natural learning is processed, there is a need to think in terms of semi-supervised learning [4, 6].

Another motivation for studying semi-supervised learning is intrinsically linked to improving the performance of computational models. It has been shown that, by means of a finite sample analysis, if the complexity of the underlying data distribution is too high to be learned by L labeled data, but it is small enough to be learned by $U \gg L$ unlabeled data, then the semi-supervised learning is really able to improve the performance of a typical fully supervised learning task [58]. As an example, consider Fig. 3.3, where the numbered circles denote labeled data, while unnumbered circles represent unlabeled data. Applying a fully supervised algorithm to this problem, the decision boundary would most likely be established in the surroundings of the dotted vertical line.[2] Applying a semi-supervised learning algorithm, in contrast, the decision boundary would probably be fixed around the continuous line, because it is a low-density region that is away both from unlabeled and labeled data. In this example, supervised algorithms would not be able to efficiently classify the unlabeled examples. In contrast, semi-supervised methods, with the aid of the unlabeled data used in the training process, would perform better.

3.4.2 *Mathematical Formalization and Fundamental Assumptions*

Semi-supervised learning can be defined as follows [10, 65]. Let $\mathscr{X} = \{x_1, x_2, \ldots, x_{L+U}\}$ be a data set, divided into two parts: $\mathscr{X}_L = \{x_1, x_2, \ldots, x_L\}$ and $\mathscr{X}_U = \{x_{L+1}, \ldots, x_{L+U}\}$. There are L and U labeled and unlabeled data items, respectively, in a total of $N = L + U$ data items. Consider that $\mathscr{Y}$ is the label or class

[2]This is the case for algorithms with a maximum margin inductive bias. As one labeled item is positioned at -1 and the other at 1, the decision boundary that maximizes the margin between these two data items must cross the zero mark, as Fig. 3.3 shows.

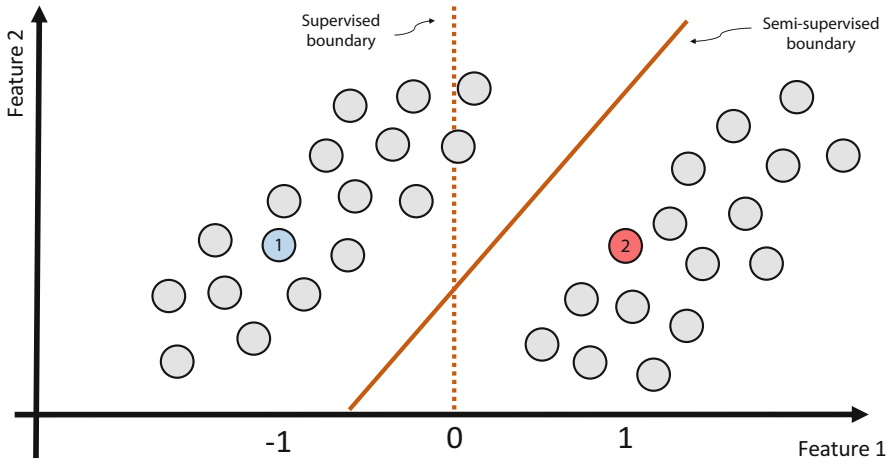

Fig. 3.3 An example of data set where semi-supervised learning techniques would lead to more robust results than supervised learning techniques. Data items are described by two numerical attributes, whose values are plotted in the horizontal and vertical axis. The numbered *circles* represent labeled data instances, while the *circles* without a number represent unlabeled ones. The *dotted line* denotes the decision boundary that would be probably output by a supervised learning method. The *continuous line* displays the same information for a semi-supervised learning algorithm

set. Supposing that the label set is discrete, this task is referred to as semi-supervised classification. (The same reasoning can be applied to regression.) The labels of the subset $\mathscr{X}_U$ are not known *a priori*. Normally, $L \ll U$, i.e., the great majority of data items does not possess labels. As we have already stressed, this often happens because the task of manual labeling is cumbersome and often is performed by human experts. The goal is to propagate labels from labeled instances to unlabeled instances in accordance to some diffusion rule. Based on these definitions, semi-supervised learning can be used in both data classification and clustering tasks. In the former case, the labeled examples are used in the process of labeling unlabeled examples. In the latter case, the labeled samples are responsible for imposing restrictions on the formation process of clusters [10].

It is worth mentioning that, for the proper functioning of semi-supervised learning techniques, some assumptions about the data consistency are essential [61]. Typically, a semi-supervised learning method relies on one or more of the following assumptions [10]:

- *Cluster assumption*: data points that belong to the same high-density region, i.e., are located in the same group, are plausible candidates for belonging to the same class.
- *Smoothness assumption*: data points that are near in the attribute space are probable candidates of being members of the same class. This assumption forces the decision function yielded by the classifier to be smoother in high-density

regions than in locations with low density. This analysis is in line with the cluster assumption and hence they complement each other.
- *Manifold assumption*: this idea is based upon the premise that a set of data points in a high dimensional space may be, approximately, reduced to a smaller space (manifold data) via a nonlinear mapping function. This hypothesis is usually employed to soften the curse of dimensionality problem, which is related to the fact that the volume of the space increases exponentially with the number of dimensions, and an exponentially larger number of examples would be needed for constructing induction classifiers with the same accuracy power.

The way that the semi-supervised learning algorithms treat these assumptions represents one of the fundamental differences among them.

Semi-supervised learning may refer to either *transductive learning* or *inductive learning*. The goal of transductive learning is to only infer the correct labels of the unlabeled data set $\mathscr{X}_U$. In contrast, the goal of inductive learning is to estimate a mapping from the available data to the output variable. Therefore, while in transductive learning the model is only used to predict the labels of $\mathscr{X}_U$, inductive learning goes beyond by also allowing other unlabeled data out of $\mathscr{X}_U$ to be estimated.

3.4.3 Overview of the Techniques

Traditionally, the semi-supervised learning methods are divided into the following categories:

- *Generative models*: The inference via generative models involves the estimation of a conditional density. The Expectation Maximization technique is the most known technique pertaining to this approach [52]. Besides that, a myriad of techniques proposed so far in the literature can be encountered in [2, 10, 22, 65].
- *Cluster-and-label models*: The inference is done based on the results obtained by a clustering task that is subjected to restrictions on the labeled data set. Some representative methods are given in [16, 17].
- *Low-density region separation models*: The inference is based on the development of decision boundary functions, such that each decision boundary is created as far as possible from high-density regions of labeled and unlabeled data items. The most well-known method of this category is the Transductive SVM [14, 60]. More related techniques can be found in [2, 10, 14, 65].
- *Network-based models*: The inference and label propagation are performed in a networked environment. In Chap. 7, we explore these techniques in detail.

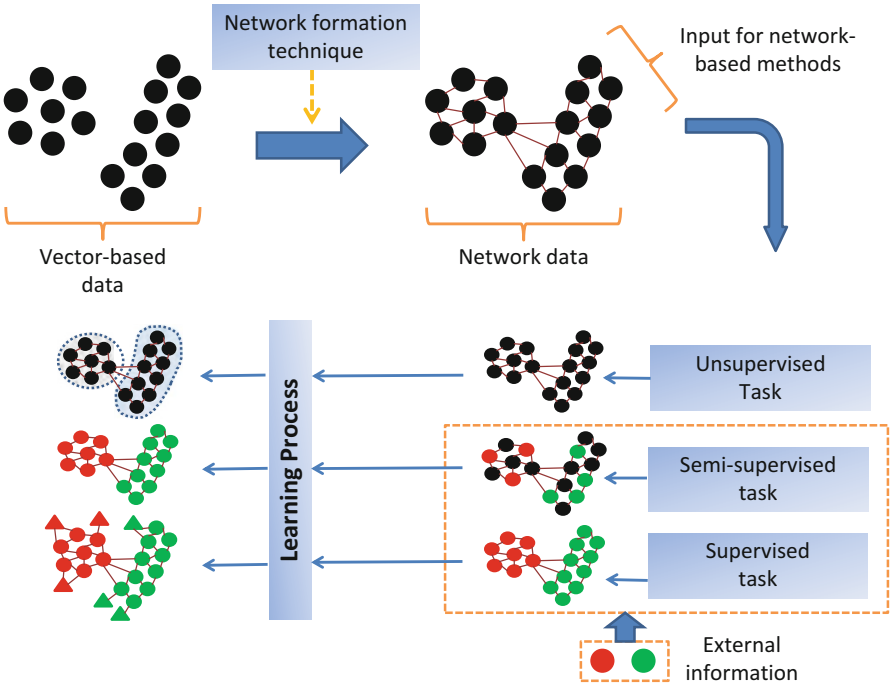

Fig. 3.4 Steps for applying machine learning methods in vector-based data. First, we transform data into a representative network using some network formation technique. That network is employed as input for network-based machine learning tasks, which in turn output the desired output

3.5 Overview of Network-Based Machine Learning

In the upcoming chapters of this book, we provide a comprehensive review on machine learning methods based on complex networks, including detailed case studies on the three learning paradigms.

When dealing with network-based methodologies, we need to follow a set of steps to complete machine learning tasks. Figure 3.4 illustrates the main steps involved to apply network-based methods in vector-based data, which we detail as follows:

1. *Gather the vector-based data set and preprocess it accordingly*. The preprocess may involve attribute transformation (scaling, normalization, standardization, demeaning, combination, decomposition, aggregation etc) or deletion, cleaning (removal of outliers, imputation of missing attributes etc), sampling, among many other preprocessing operations. The machine learning tools that we use influence the type of preprocessing we need to perform. For instance, if we have a large data set and an algorithm with high time complexity order, we need to

perform a sampling process on that data set so as to become viable the learning process.
2. *Transform the vector-based into network-based data*. This step is performed using a network formation technique. This is a crucial problem for network-based problems, because the resulting network must reliably represent the data relationships. Chapter 4 deals with this problem in a comprehensive manner.
3. *Apply the network-based machine learning task in the network constructed from the vector-based data*. The generated network in the previous step serves as input for network-based machine learning tasks. Three types of tasks can be performed when we are at the machine learning domain:

   - *Network-based unsupervised learning*: only the generated network is employed in the learning process. Chapter 6 reviews the state-of-the-art unsupervised learning techniques presented in the literature, while Chap. 9 presents a case study of unsupervised learning in data clustering and in community detection that is based on competition of multiple interacting particles.
   - *Network-based supervised learning*: besides the generated network, we are also given external information in the form of labels. Chapter 5 surveys representative supervised learning algorithms that learn in a networked environment. In addition, Chap. 8 explores a case study of supervised learning in data classification and in handwritten digits recognition that is based on a high-level classification framework.
   - *Network-based semi-supervised learning*: besides the generated network, we are also given external information for some of the training items in the form of labels. Chapter 7 compiles a set of semi-supervised techniques that relies on networks to learn. Moreover, Chap. 10 examines a case study of semi-supervised learning in data classification and in imperfect learning that is based on a competitive-cooperative scheme of multiple interacting particles.

3.6 Chapter Remarks

In this section, we have introduced the machine learning area and its three main learning paradigms: supervised, unsupervised, and semi-supervised. Machine learning is a field of study that gives computers the ability to learn without being explicitly programmed.

We have seen that supervised algorithms utilize external information in the form of labels or classes to induce or to train their hypotheses. In the supervised learning scheme, there are two learning phases: the training and the classification phases. In the training phase, the classifier is induced or trained by using the training instances (labeled data) that are always fully labeled. In the classification phase, the labels of the test instances are predicted using the induced classifier. We have seen that, once trained, one must verify how well the trained supervised learner can generalize.

Without any additional assumptions over the data properties, this problem cannot be solved exactly since unseen situations might have an arbitrary output values. The necessary assumptions about the nature of the target function constitute the inductive bias of the learner.

Unsupervised learning methods are guided exclusively by the intrinsic structure of the data items throughout the learning process, i.e., without any sort of external knowledge. One of the most common tasks of unsupervised learning is in data clustering. The clusters may be defined as sets of patterns, points, or objects, all of which are characterized by suitable similarity measures. Each cluster is a collection of data items that are similar between them and are dissimilar to objects belonging to other clusters. Likewise supervised learning, we have seen that unsupervised algorithms also make assumptions about the underlying data generation process and that fact leads to shortcomings and potentialities of each available algorithm.

Between supervised and unsupervised learning lies semi-supervised learning. In this paradigm, algorithms are able to learn using only a few labeled examples. Among other features, semi-supervised learning aims at reducing the work of human experts in the labeling process. The perceptive difference between semi-supervised and supervised learning is that the latter only uses labeled data in the training phase, while the former employs both the labeled and unlabeled data. Under some conditions, we have seen that the introduction of unlabeled data in the training phase can really improve the overall classification performance of the classifier.

The goal in this chapter is to provide an overview on the usual hypotheses and limitations that we encounter in machine learning tasks. In the next four chapters, we first review network formation techniques and then revisit each of the learning paradigms with a focus on network-based approaches.

References

1. Aggarwal, C.C., Reddy, C.K.: Data Clustering: Algorithms and Applications. CRC, Boca Raton (2014)
2. Alpaydin, E.: Introduction to Machine Learning (Adaptive Computation and Machine Learning). MIT, Cambridge (2004)
3. Arlot, S., Celisse, A.: A survey of cross-validation procedures for model selection. Stat. Surv. **4**, 40–79 (2010)
4. Belkin, M., Matveeva, I., Niyogi, P.: Regularization and semi-supervised learning on large graphs. In: Shawe-Taylor, J., Singer, Y. (eds.) Learning Theory. Lecture Notes in Computer Science, vol. 3120, pp. 624–638. Springer, Berlin/Heidelberg (2004)
5. Belkin, M., Niyogi, P., Sindhwani, V.: On manifold regularization. In: Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics (AISTAT 2005), pp. 17–24. Society for Artificial Intelligence and Statistics, New Jersey (2005)
6. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. J. Mach. Learn. Res. **7**, 2399–2434 (2006)
7. Bengio, Y., Grandvalet, Y.: No unbiased estimator of the variance of k-fold cross-validation. J. Mach. Learn. Res. **5**, 1089–1105 (2004)
8. Berkhin, P.: Survey of clustering data mining techniques. Technical Report, Accrue Software (2002)

9. Bishop, C.M.: Pattern Recognition and Machine Learning (Information Science and Statistics). Springer, New York (2007)
10. Chapelle, O., Schölkopf, B., Zien, A. (eds.): Semi-supervised learning. Adaptive Computation and Machine Learning. MIT, Cambridge (2006)
11. Chung, F.R.K.: Spectral graph theory. CBMS Regional Conference Series in Mathematics, vol. 92. American Mathematical Society, Providence (1997)
12. Cinque, L., Foresti, G.L., Lombardi, L.: A clustering fuzzy approach for image segmentation. Pattern Recogn. **37**, 1797–1807 (2004)
13. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. Phys. Rev. E **70**(6), 066111+ (2004)
14. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**, 273–297 (1995)
15. Cover, T.M., Hart, P.: Nearest neighbor pattern classification. IEEE Trans. Inf. Theory **13**, 21–27 (1967)
16. Dara, R., Kremer, S.C., Stacey, D.A.: Clustering unlabeled data with SOMs improves classification of labeled real-world data. In: Proceedings of the World Congress on Computational Intelligence (WCCI), pp. 2237–2242 (2002)
17. Demiriz, A., Bennett, K.P., Embrechts, M.J.: Semi-supervised clustering using genetic algorithms. In: Proceedings of Artificial Neural Networks in Engineering (ANNIE-99), pp. 809–814. ASME (1999)
18. Deng, L., Yu, D.: Deep learning: Methods and applications. Founda. Trends Signal Process. **7**(3), 197–387 (2014)
19. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. Wiley-Interscience, Chichester (2000)
20. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. Wiley, New York (2001)
21. Gan, G.: Data Clustering: Theory, Algorithms, and Applications. Society for Industrial and Applied Mathematics, Philadelphia (2007)
22. Gärtner, T.: Kernels for Structured Data, vol. 72. World Scientific Publishing, Singapore (2008)
23. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. Proc. Natl. Acad. Sci. U.S.A. **99**(12), 7821–7826 (2002)
24. Guha, S., Rastogi, R., Shim, K.: CURE: an efficient clustering algorithm for large databases. Inf. Syst. **26**(1), 35–58 (2001)
25. Hasan, M.A., Chaoji, V., Salem, S., Zaki, M.: Link prediction using supervised learning. In: Proceedings of SDM 06 workshop on Link Analysis, Counterterrorism and Security (2006)
26. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, New York (2011)
27. Haykin, S.S.: Neural Networks and Learning Machines. Prentice Hall, Englewood Cliffs (2008)
28. Husek, D., Pokorny, J., Rezanková, H., Snášel, V.: Data clustering: from documents to the web. In: Web Data Management Practices: Emerging Techniques and Technologies, pp. 1–33. IGI Global, Hershey, PA (2006)
29. Jain, A.K.: Data clustering: 50 years beyond K-means. Pattern Recogn. Lett. **31**, 651–666 (2010)
30. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: A review. ACM Comput. Surv. **31**(3), 264–323 (1999)
31. James, G., Witten, D., Hastie, T., Tibshirani, R.: An Introduction to Statistical Learning: with Applications in R. Springer, New York (2013)
32. Jolliffe, I.T.: Principal Component Analysis. Springer Series in Statistics. Springer, New York (2002)
33. Karypis, G., Han, E.H., Kumar, V.: Chameleon: hierarchical clustering using dynamic modeling. Computer **32**(8), 68–75 (1999)
34. Kashef, R., Kamel, M.S.: Enhanced bisecting K-Means clustering using intermediate cooperation. Pattern Recogn. **42**(11), 2557–2569 (2009)
35. Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data: An Introduction to Cluster Analysis. Wiley, New York (2005)

36. Kodratoff, Y., Michalski, R.S.: Machine Learning: An Artificial Intelligence Approach, vol. 3. Morgan Kaufmann, San Mateo (2014)
37. Korb, K.B., Nicholson, A.E.: Bayesian Artificial Intelligence. Chapman and Hall, Boca Raton (2010)
38. Kuhn, M., Johnson, K.: Applied Predictive Modeling. Springer, New York (2013)
39. Lim, G., Park, C.H.: Semi-supervised dimension reduction using graph-based discriminant analysis. In: Computer and Information Technology (CIT), vol. 1, pp. 9–13. IEEE Computer Society, Xiamen (2009)
40. Liu, H., Shah, S., Jiang, W.: On-line outlier detection and data cleaning. Comput. Chem. Eng. **28**, 1635–1647 (2004)
41. Lu, C.T., Chen, D., Kou, Y.: Algorithms for spatial outlier detection. In: Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM 2003). IEEE Computer Society (2003)
42. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297. University of California Press (1967)
43. Marsland, S.: Machine Learning: An Algorithmic Perspective. CRC, Boca Raton (2014)
44. Mitchell, T.M.: Machine Learning. McGraw-Hill Science/Engineering/Math, New York, NY (1997)
45. Müller, P., Quintana, F.A., Jara, A., Hanson, T.: Bayesian Nonparametric Data Analysis. Springer, New York (2015)
46. Murphy, K.P.: Machine Learning: A Probabilistic Perspective. MIT, Cambridge (2012)
47. Neapolitan, R.E.: Learning Bayesian Networks. Prentice Hall, Upper Saddle River (2003)
48. Newman, M.E.J.: Finding community structure in networks using the eigenvectors of matrices. Phys. Rev. E **74**(3), 036104 (2006)
49. Newman, M.E.J.: Modularity and community structure in networks. Proc. Natl. Acad. Sci. **103**(23), 8577–8582 (2006)
50. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. Lett. **69**, 026113 (2004)
51. Ng, R.T., Han, J.: CLARANS: A method for clustering objects for spatial data mining. IEEE Trans. Knowl. Data Eng. **14**(5), 1003–1016 (2002)
52. Nigam, K., McCallum, A.K., Thrun, S., Mitchell, T.: Text classification from labeled and unlabeled documents using EM. Mach. Learn. **39**(2–3), 103–134 (2000)
53. Piatetsky-Shapiro, G.: Discovery, Analysis, and Presentation of Strong Rules, chap. 12. AAAI/MIT, Cambridge (1991)
54. Quinlan, J.R.: Generating production rules from decision trees. In: Proceedings of the 10th International Joint Conference on Artificial Intelligence (IJCAI'87), vol. 1, pp. 304–307. Morgan Kaufmann, San Mateo (1987)
55. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Series in Machine Learning. Morgan Kaufmann, San Mateo (1992)
56. Sander, J., Ester, M., Kriegel, H.P., Xu, X.: Density-based clustering in spatial databases: the algorithm GDBSCAN and its applications. Data Min. Knowl. Disc. **2**(2), 169–194 (1998)
57. Schmidhuber, J.: Deep learning in neural networks: an overview. Neural Netw. **61**, 85–117 (2015)
58. Singh, A., Nowak, R.D., Zhu, X.: Unlabeled data: Now it helps, now it doesn't. In: The Conference on Neural Information Processing Systems NIPS, pp. 1513–1520 (2008)
59. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer, New York (1995)
60. Vapnik, V.N.: Statistical Learning Theory. Wiley-Interscience, New York (1998)
61. Wang, F., Li, T., Wang, G., Zhang, C.: Semi-supervised classification using local and global regularization. In: AAAI'08: Proceedings of the 23rd National Conference on Artificial Intelligence, pp. 726–731. AAAI (2008)
62. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kauffman, San Mateo (2005)
63. Xu, R., II, D.W.: Survey of clustering algorithms. IEEE Trans. Neural Netw. **16**(3), 645–678 (2005)

64. Zhu, X.: Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison (2005)
65. Zhu, X., Goldberg, A.B.: Introduction to Semi-Supervised Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan and Claypool Publishers, San Rafael (2009)