Chapter 1
Machine Learning and Data Mining

Machine learning and data mining are research areas of computer science whose quick development is due to the advances in data analysis research, growth in the database industry and the resulting market needs for methods that are capable of extracting valuable knowledge from large data stores. This chapter gives an informal introduction to machine learning and data mining, and describes selected machine learning and data mining methods illustrated by examples. After a brief general introduction, Sect. 1.2 briefly sketches the historical background of the research area, followed by an outline of the knowledge discovery process and the emerging standards in Sect. 1.3. Section 1.4 establishes the basic terminology and provides a categorization of different learning tasks. Predictive and descriptive data mining techniques are illustrated by means of simplified examples of data mining tasks in Sects. 1.5 and 1.6, respectively. In Sect. 1.7, we highlight the importance of relational data mining techniques. The chapter concludes with some speculations about future developments in data mining.

1.1 Introduction

Machine learning (Mitchell, 1997) is a mature and well-recognized research area of computer science, mainly concerned with the discovery of models, patterns, and other regularities in data. Machine learning approaches can be roughly categorized into two different groups:

Symbolic approaches. Inductive learning of symbolic descriptions, such as rules (Clark & Niblett, 1989; Michalski, Mozetič, Hong, & Lavrač, 1986) decision

[†]This chapter is partly based on Lavrač & Grobelnik (2003).

J. Fürnkranz et al., *Foundations of Rule Learning*, Cognitive Technologies,
DOI 10.1007/978-3-540-75197-7_1, © Springer-Verlag Berlin Heidelberg 2012

trees (Quinlan, 1986) or logical representations (De Raedt, 2008; Lavrač & Džeroski, 1994a; Muggleton, 1992). Textbooks that focus on this line of research include (Langley, 1996; Mitchell, 1997; Witten & Frank, 2005).

Statistical approaches. Statistical or pattern-recognition methods, including k-nearest neighbor or instance-based learning (Aha, Kibler, & Albert, 1991; Dasarathy, 1991), Bayesian classifiers (Pearl, 1988), neural network learning (Rumelhart & McClelland, 1986), and support vector machines (Schölkopf & Smola, 2001; Vapnik, 1995). Textbooks in this area include (Bishop, 1995; Duda, Hart, & Stork, 2000; Hastie, Tibshirani, & Friedman, 2001; Ripley, 1996).

Although the approaches taken in these fields are often quite different, their effectiveness in learning is often comparable (Michie, Spiegelhalter, & Taylor, 1994). Also, there are many approaches that cross the boundaries between the two approaches. For example, there are decision tree (Breiman, Friedman, Olshen, & Stone, 1984) and rule learning (Friedman & Fisher, 1999) algorithms that are firmly based in statistics. Similarly, ensemble techniques such as boosting (Freund & Schapire, 1997), bagging (Breiman, 1996) or random forests (Breiman, 2001a) may combine the predictions of multiple logical models on a sound statistical basis (Bennett et al., 2008; Mease & Wyner, 2008; Schapire, Freund, Bartlett, & Lee, 1998). This book is concerned only with the first group of methods, which result in symbolic, human-understandable patterns and models.

Due to the growth in the database industry and the resulting market needs for methods that are capable of extracting valuable knowledge from large data stores, *data mining* (DM) and *knowledge discovery in databases* (KDD) (Fayyad, Piatetsky-Shapiro, Smyth, & Uthurusamy, 1995; Han & Kamber, 2001; Piatetsky-Shapiro & Frawley, 1991) have recently emerged as a new scientific and engineering discipline, with separate workshops, conferences and journals. According to Witten and Frank (2005), data mining means "solving problems by analyzing data that already exists in databases". In addition to the mining of structured data stored in data warehouses—e.g., in the form of relational data tables—there has recently also been increased interest in the mining of unstructured data such as text and web.

Research areas related to machine learning and data mining include database technology and data warehouses, pattern recognition and soft computing, text and web mining, visualization, and statistics.

- *Database technology and data warehouses* are concerned with the efficient storage, access and manipulation of data.
- *Pattern recognition and soft computing* typically provide techniques for classifying data items.
- *Text and web mining* are used for web page analysis, text categorization, as well as filtering and structuring of text documents; natural language processing can provide useful tools for improving the quality of text mining results.
- *Visualization* concerns the visualization of data as well as the visualization of data mining results.
- *Statistics* is a classical data analysis discipline, mainly concerned with the analysis of large collections of numerical data.

As statistics already provides numerous data analysis tools (Breiman, 2001b; Friedman, 1998), a relevant question is whether machine learning and data mining are needed at all. There are several possible answers. First, as industry needs solutions for real-life problems, one of the most important issues is the problem solving speed: many data mining methods are able to deal with very large datasets in a very efficient way, while the algorithmic complexity of statistical methods may turn out to be prohibitive for their use on very large databases. Next, the results of the analysis need to be represented in an appropriate, usually human understandable way; apart from the analytical languages used in statistics, data mining methods also use other forms of formalisms, the most popular being decision trees and rule sets. The next important issue in a real-life setting concerns the assumptions about the data. In general one may claim that data mining deals with all sorts of structured tabular data (e.g., non-numeric, highly unbalanced, unclean data) as well as with non-structured data (e.g., text documents, images, multimedia), and does not make assumptions about the distribution of the data. Finally, while one of the main goals of statistics is hypothesis testing, one of the main goals of data mining is the construction of hypotheses.

1.2 Historical Background

Machine learning is a well-established research area of computer science. Early machine learning algorithms were perceptrons (later called neural networks, Rumelhart & McClelland, 1986), decision tree learners like ID3 (Quinlan, 1979, 1986) and CART (Breiman et al., 1984), and rule learners like AQ (Michalski, 1969; Michalski et al., 1986) and INDUCE (Michalski, 1980). These early algorithms were typically used to induce classifiers from a relatively small set of training examples (up to a thousand) described by a small set of attributes (up to a 100). An overview of early work in machine learning can be found in (Michalski, Carbonell, & Mitchell, 1983, 1986).

Data mining and knowledge discovery in databases appeared as a recognizable research discipline in the early 1990s (Piatetsky-Shapiro & Frawley, 1991), with the advent of a series of data mining workshops. The birth of this area was triggered by a need in the database industry to deliver solutions enhancing the traditional database management systems and technologies. At that time, these systems were able to solve the basic data management issues like how to deal with the data in transactional processing systems. In online transactional processing (OLTP) most of the processing scenarios were predefined. The main emphasis was on the stability and safety of solutions.

As the business emphasis changed from automation to decision support, limitations of OLTP systems in business support led to the development of the next-generation data management technology known as data warehousing. The motivation for data warehousing was to provide tools for supporting analytical operations for decision support that were not easily provided by the existing

database query languages. Online analytical processing (OLAP) was introduced to enable inexpensive data access and insights which did not need to be defined in advance. However, the typical operations on data warehouses were similar to the ones from the traditional OLTP databases in that the user issued a query and received a data table as a result. The major difference between OLTP and OLAP is the average number of records accessed per typical operation. While a typical operation in OLTP affects only up to tens or hundreds of records in predefined scenarios, a typical operation in OLAP affects up to millions of records (sometimes all records) in the database in a non-predefined way.

The role of data mining in the above framework can be explained as follows. While typical questions in OLTP and OLAP are of the form: '*What is the answer to the given query?*', data mining—in a somewhat simplified and provocative formulation—addresses the question '*What is the right question to ask about this data?*'. The following explanation can be given. Data warehousing/OLAP provides analytical tools enabling only user-guided analysis of the data, where the user needs to have enough advance knowledge about the data to be able to raise the right questions in order to get the appropriate answers. The problem arises in situations when the data is too complex to be appropriately understood and analyzed by a human. In such cases data mining can be used, providing completely different types of operations for handling the data, aimed at hypothesis construction, and providing answers to questions which—in most cases—cannot be formulated precisely.

1.3 Knowledge Discovery Process and Standardization

Data mining is the core stage of the knowledge discovery process that is aimed at the extraction of interesting—nontrivial, implicit, previously unknown and potentially useful—information from data in large databases (Fayyad, Piatetsky-Shapiro, & Smyth, 1996). Data mining projects were initially carried out in many different ways with each data analyst finding their own way of approaching the problem, often through trial-and-error. As the data mining techniques and businesses evolved, there was a need for data analysts to better understand and standardize the knowledge discovery process, which would—as a side effect—demonstrate to prospective customers that data mining was sufficiently mature to be adopted as a key element of their business. This led to the development of the *cross-industry standard process for data mining* (CRISP-DM; Chapman et al., 2000), which is intended to be independent of the choice of data mining tools, industry segment, and the application/problem to be solved.

The CRISP-DM methodology defines the crucial steps of the knowledge discovery process. Although in most data mining projects, several iterations of individual steps or step sequences need to be performed, these basic guidelines are very useful both for the data analyst and the client. Below is a list of CRISP-DM steps.

1. *Business understanding:* understanding and defining of business goals and the actual goals of data mining.
2. *Data understanding:* familiarization with the data and the application domain, by exploring and defining the relevant prior knowledge.
3. *Data preparation through data cleaning and preprocessing*: creating the relevant data subset through data selection, as well as finding of useful properties/attributes, generating new attributes, defining appropriate attribute values and/or value discretization.
4. *Data mining*: the most important step of this process, which is concerned with choosing the most appropriate data mining tools—from the available tools for summarization, classification, regression, association, clustering—and searching for patterns or models of interest.
5. *Evaluation and interpretation of results*: aided by pattern/model visualization, transformation, and removal of redundant patterns.
6. *Deployment:* the use of the discovered knowledge.

A terminological note needs to be made at this point. While data mining is considered to be the core step of the knowledge discovery process, in this book—as with most industrial applications—we use the term data mining interchangeably with knowledge discovery.

In addition to the CRISP-DM standardized methodology for building data mining applications, standards covering specific phases of the process are also emerging. These standards include:

– The XML-based Predictive Modeling Markup Language (PMML) (Pechter, 2009) standard for storing and sharing data mining results,
– A standard extending the Microsoft analysis server with new data mining functionality (OLE DB for data mining, using a customized SQL language),
– Part of the ISO effort to define multimedia and application-specific SQL types and their methods, including support for data mining functionality (SQL/MM), and
– The emerging Java API for data mining (JDM).

The standardization efforts and numerous tools available (IBM Intelligent Miner, SAS Enterprise Miner, SPSS Clementine, and many others), including the publicly available academic data mining platforms WEKA (Hall et al., 2009; Witten & Frank, 2005), RAPID-I (formerly YALE; Mierswa, Wurst, Klinkenberg, Scholz, & Euler, 2006), the Konstanz Information Miner KNIME (Berthold et al., 2009), ORANGE (Demšar, Zupan, & Leban, 2004), and the statistical data analysis package R (Everitt & Hothorn, 2006; Torgo, 2010) demonstrate that data mining has made progress towards becoming a mature and widely used technology for analytical practices.

Most of the available tools are capable of mining data in tabular format, describing a dataset in terms of a fixed collection of attributes (properties), as is the case with transactional databases. More sophisticated tools are available for data mining from relational databases, data warehouses and stores of text documents.

Methods and tools for the mining of advanced database systems and information repositories (object-oriented and object-relational databases, spatial databases, time-series data and temporal data, multimedia data, heterogeneous and legacy data, World-Wide Web) still lack commercial deployment.

1.4 Terminology and Categorization of Learning Tasks

In the simplest case, data mining techniques operate on a single data table. Rows in the data table correspond to objects (*training examples*) to be analyzed in terms of their properties (*attributes*) and the concept (*class*) to which they belong. There are two main approaches:

Supervised learning. Supervised learning assumes that training examples are classified (labeled by class labels)

Unsupervised learning. Unsupervised learning concerns the analysis of unclassified examples.

In both cases, the goal is to induce a *model* for the entire dataset, or to discover one or more *patterns* that hold for some part of the dataset.

In supervised learning, data is usually formed from examples (records of given attribute values) which are labeled by the class to which they belong (Kotsiantis, Zaharakis, & Pintelas, 2006). The task is to find a model (a classifier) that will enable a newly encountered instance to be classified. Examples of discrete classification tasks are classification of countries based on climate, classification of cars based on gas consumption, or prediction of a diagnosis based on patient's medical condition.

Let us formulate a classification/prediction task, and illustrate it by a simplified example. As described above, we are given a database of observations described with a fixed number of attributes $\mathbf{A}_i$, and a designated class attribute $\mathbf{C}$. The learning task is to find a mapping f that is able to compute the class value $\mathbf{C} = f(\mathbf{A}_1, \ldots, \mathbf{A}_n)$ from the attribute values of new, previously unseen observations.

Table 1.1 shows a very small, artificial sample database.[1] The database contains the results of a survey on 14 individuals, concerning the approval or disapproval of a certain issue. Each individual is characterized by four attributes—Education (with possible values primary school, secondary school, or university), MaritalStatus (with possible values single, married, or divorced), Sex (male or female), and HasChildren (yes or no)—that encode rudimentary information about their sociodemographic background. The last column, Approved, is the class attribute, encoding whether the individual approved or disapproved of the issue.

The task is to use the information in this *training set* to derive a model that is able to predict whether a person is likely to approve or disapprove the issue, based on the four demographic characteristics. While there are statistical techniques that are able

[1] The dataset is adapted from the well-known dataset Quinlan (1986).

Table 1.1 A sample database

No.	Education	Marital status	Sex	Has children	Approved
1	Primary	Single	Male	No	No
2	Primary	Single	Male	Yes	No
3	Primary	Married	Male	No	Yes
4	University	Divorced	Female	No	Yes
5	University	Married	Female	Yes	Yes
6	Secondary	Single	Male	No	No
7	University	Single	Female	No	Yes
8	Secondary	Divorced	Female	No	Yes
9	Secondary	Single	Female	Yes	Yes
10	Secondary	Married	Male	Yes	Yes
11	Primary	Married	Female	No	Yes
12	Secondary	Divorced	Male	Yes	No
13	University	Divorced	Female	Yes	No
14	Secondary	Divorced	Male	No	Yes

to solve particular instances of this problem, mainly focusing on the analysis of numeric data, machine learning and data mining techniques focus on the analysis of categorical, non-numeric data, and on the interpretability of the result.

Typical data mining approaches find patterns or models in a single data table, while some, like most of the relational data mining approaches, (Džeroski & Lavrač, 2001; Lavrač & Džeroski, 1994a) find patterns/models from data stored in multiple tables, e.g., in a given relational database.

Propositional learning. Data mining approaches that find patterns/models in a given single table are referred to as *attribute-value* or *propositional* learning approaches, as the patterns/models they find can be expressed in propositional logic.

Relational learning. *First-order learning* approaches are also referred to as *relational data mining* (RDM) (Džeroski & Lavrač, 2001), *relational learning* (RL) (Quinlan, 1990) or *inductive logic programming* (ILP) (Lavrač & Džeroski, 1994a; Muggleton, 1992), as the patterns/models they find are expressed in relational formalisms of first-order logic.

We further distinguish between predictive and descriptive data mining. In the example above, a predictive data mining approach will aim at building a predictive classification model for classifying new instances into one of the two class values (yes or no). On the other hand, in descriptive data mining the input data table will typically not contain a designated class attribute and will aim at finding patterns describing the relationships between other attribute values.

Predictive data mining. Predictive data mining methods are supervised. They are used to induce models or theories (such as decision trees or rule sets) from class-labeled data. The induced models can be used for prediction and classification.

Descriptive data mining. Descriptive data mining methods are typically unsupervised. They are used to induce interesting patterns (such as association rules) from unlabeled data. The induced patterns are useful in exploratory data analysis.

While there is no clear distinction in the literature, we will generally use the term *pattern* for results of a descriptive data mining process, whereas we will use the terms *model*, *theory*, or *hypothesis* for results of a predictive data mining task.

The next two sections briefly introduce the two main learning approaches, predictive and descriptive induction.

1.5 Predictive Data Mining: Induction of Models

This data analysis task is concerned with the induction of models for classification and prediction purposes, and is referred to as *predictive induction*. Two symbolic data mining methods that result in classification/prediction models are outlined in this section: decision tree induction and rule set induction.

1.5.1 Decision Tree Induction

A *decision tree* is a classification model whose structure consists of a number of *nodes* and *arcs*. In general, a node is labeled by an attribute name, and an arc by a valid value of the attribute associated with the node from which the arc originates. The top-most node is called the *root* of the tree, and the bottom nodes are called the *leaves*. Each leaf is labeled by a class (value of the class attribute). When used for classification, a decision tree is traversed in a top-down manner, following the arcs with attribute values satisfying the instance that is to be classified. The traversal of the tree leads to a leaf node and the instance is assigned the class label of the leaf. Figure 1.1 shows a decision tree induced from the training set shown in Table 1.1.

A decision tree is constructed in a top-down manner, starting with the most general tree consisting of only the root node, and then refining it to a more specific tree structure. A small tree consisting only of the root node is *too general*, while the most specific tree which would construct a leaf node for every single data instance would be *too specific*, as it would *overfit* the data. The art of decision tree construction is to construct a tree at the right 'generality level' which will adequately generalize the training data to enable high predictive accuracy on new instances.

The crucial step in decision tree induction is the choice of an attribute to be selected as a node in a decision tree. Typical attribute selection criteria use a function that measures the *purity* of a node, i.e., the degree to which the node contains only examples of a single class. This purity measure is computed for a node and

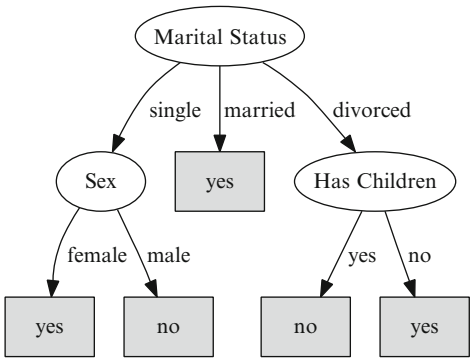

Fig. 1.1 A decision tree describing the dataset shown in Table 1.1

all successor nodes that result from using an attribute for splitting the data. In the well-known C4.5 decision tree algorithm, which uses information-theoretic entropy as a purity measure (Quinlan, 1986), the difference between the original purity value and the sum of the purity values of the successor nodes weighted by the relative sizes of these nodes, is used to estimate the utility of this attribute, and the attribute with the largest utility is selected for expanding the tree.

To see the importance of this choice, consider a procedure that constructs decision trees simply by picking the next available attribute instead of the most informative attribute. The result is a much more complex and less comprehensible tree (Fig. 1.2). Most leaves cover only a single training example, which means that this tree is overfitting the data. Consequently, the labels that are attached to the leaves are not very reliable. Although the trees in Figs. 1.1 and 1.2 both classify the training data in Table 1.1 correctly, the former appears to be more trustworthy, and it has a higher chance of correctly predicting the class values of new data.[2]

Note that some of the attributes may not occur at all in the tree; for example, the tree in Fig. 1.1 does not contain a test on Education. Apparently, the data can be classified without making a reference to this variable. In addition, the attributes in the upper parts of the tree (near the root) have a stronger influence on the value of the target variable than the nodes in the lower parts of the tree, in the sense that they participate in the classification of a larger number of instances.

As a result of the recursive partitioning of the data at each step of the top-down tree construction process, the number of examples that end up in each node decreases steadily. Consequently, the reliability of the chosen attributes decreases with increasing depths of the tree. As a result, overly complex models are generated, which explain the training data but do not generalize well to unseen data. This is known as *overfitting*. This is the main reason why the state-of-the-art decision tree learners employ a post-processing phase in which the generated tree is simplified

[2]The preference for simpler models is a heuristic criterion known as *Occam's razor*, which appears to work well in practice. It is often addressed in the literature on model selection, but its utility has been the subject of discussion (Domingos, 1999; Webb, 1996).

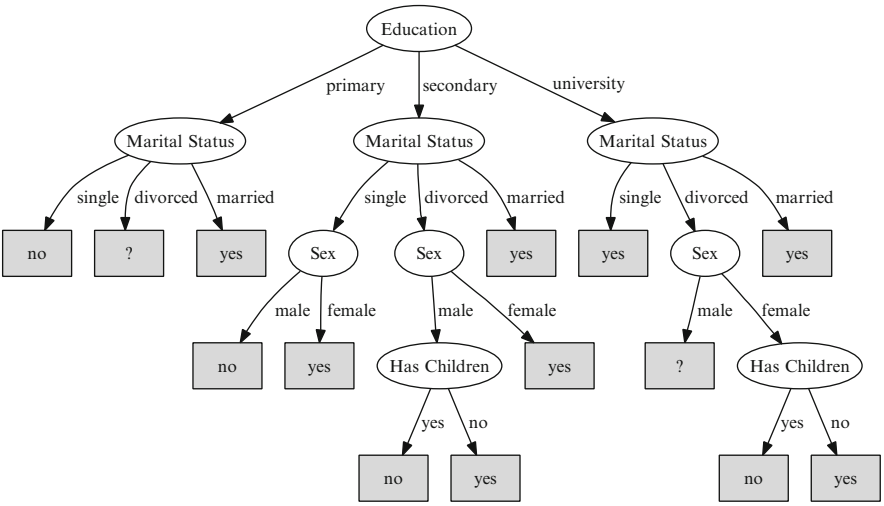

Fig. 1.2 A bad decision tree describing the dataset shown in Table 1.1

by *pruning* branches and nodes near the leaves, which results in replacing some of the interior nodes of the tree with a new leaf, thereby removing the subtree that was rooted at this node. It is important to note that the leaf nodes of the new tree are no longer pure nodes, containing only training examples of the same class labeling the leaf; instead the leaf will bear the label of the most frequent class at the leaf.

Many decision tree induction algorithms exist, the most popular being C4.5 and its variants: a commercial product SEE5, and J48, which is available in the WEKA workbench (Witten & Frank, 2005), as open source.

1.5.2 Rule Set Induction

Another important machine learning technique is the induction of rule sets. The learning of rule-based models has been a main research goal in the field of machine learning since its beginning in the early 1960s. Recently, rule-based techniques have also received increased attention in the statistical community (Friedman & Fisher, 1999).

A rule-based classification model consists of a set of if–then rules. Each *rule* has a conjunction of attribute values (which will in the following be called *features*) in the conditional part of the rule, and a class label in the rule consequent. As an alternative to such logical rules, *probabilistic rules* can be induced; in addition to the predicted class label, the consequent of these rules consists also of a list of probabilities or numbers of covered training instances for each possible class label (Clark & Boswell, 1991).

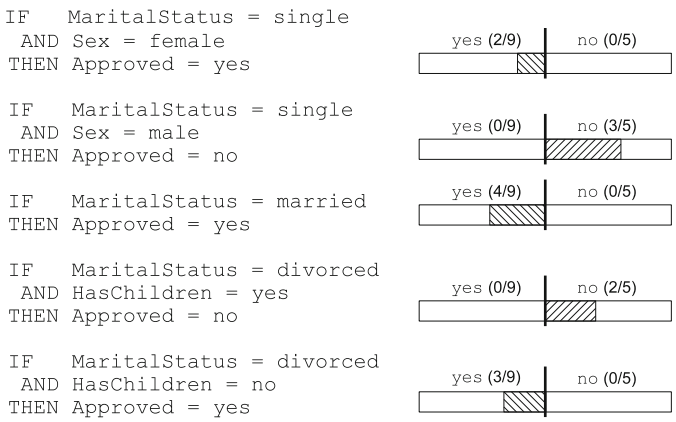

Fig. 1.3 A rule set describing the dataset shown in Table 1.1

Rule sets are typically simpler and more comprehensible than decision trees. To see why, note that a decision tree can also be interpreted as a set of if—then rules. Each leaf in the tree corresponds to one rule, where the conditions encode the path that is taken from the root to this particular leaf, and the conclusion of the rule is the label of that leaf. Figure 1.3 shows the set of rules that corresponds to the tree in Fig. 1.1. Note the rigid structure of these rules. For example, the first condition always uses the same attribute, namely, the one used at the root of the tree. Next to each rule, we show the proportion of covered examples for each class value.

The main difference between the rules generated by a decision tree and the rules generated by a rule learning algorithm is that the former rule set consists of nonoverlapping rules that span the entire instance space (i.e., each possible combination of attribute values will be covered by exactly one rule). Relaxing this constraint—by allowing for potentially overlapping rules that need not span the entire instance space—may often result in smaller rule sets. However, in this case, we need mechanisms for tie breaking (which rule to choose when more than one covers the example to be classified) and default classifications (what classification to choose when no rule covers the given example). Typically, one prefers rules with a higher ratio of correctly classified examples from the training set.

Figure 1.4 shows a particularly simple rule set which uses two different attributes in its first two rules. Note that these two rules are overlapping: several examples will be covered by more than one rule. For instance, examples 3 and 10 are covered by both the first and the third rule. These conflicts are typically resolved by using the more accurate rule, i.e., the rule that covers a higher proportion of examples that support its prediction (the first one in our case). Also note that this rule set makes two mistakes (the last two examples). These might be resolved by resorting to a

```
IF   MaritalStatus = married
THEN Approved = yes

IF   Sex = female
THEN Approved = yes

IF   Sex = male
THEN Approved = no

DEFAULT  Approved = yes
```

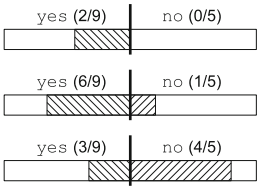

Fig. 1.4 A smaller rule set describing the dataset shown in Table 1.1

more complex rule set (like the one in Fig. 1.3) but, as stated above, it is often more advisable to sacrifice accuracy on the training set for model simplicity in order to avoid overfitting the training data. Finally, note the *default rule* at the end of the rule set. This is added for the case when certain regions of the data space are not represented in the training set.

The key ideas for learning such rule sets are quite similar to the ideas used in decision tree induction. However, instead of recursively partitioning the dataset by optimizing the purity measure over all successor nodes (in the literature, this strategy is also known as *divide-and-conquer* learning), rule learning algorithms only expand a single successor node at a time, thereby learning a complete rule that covers part of the training data. After a complete rule has been learned, all examples that are covered by this rule are removed from the training set, and the procedure is repeated with the remaining examples (this strategy is also known as *separate-and-conquer* learning). Again, pruning is a good idea for rule learning, which means that the rules only need to cover examples that are *mostly* from the same class. It turns out to be advantageous to prune rules immediately after they have been learned, that is before successive rules are learned (Fürnkranz, 1997).

1.5.3 Rule Sets Versus Decision Trees

There are several aspects which make rule learning attractive. First of all, decision trees are often quite complex and hard to understand. Quinlan (1993) has noted that even pruned decision trees may be too cumbersome, complex, and inscrutable to provide insight into the domain at hand and has consequently devised procedures for simplifying decision trees into pruned production rule sets (Quinlan, 1987a, 1993). Additional evidence for this comes from Rivest (1987), showing that decision lists (ordered rule sets) with at most k conditions per rule are strictly more expressive than decision trees of depth k. A similar result has been proved by Boström (1995).

Moreover, the restriction of decision tree learning algorithms to nonoverlapping rules imposes strong constraints on learnable rules. One problem resulting from this

constraint is the *replicated subtree problem* (Pagallo & Haussler, 1990); it often happens that identical subtrees have to be learned at various places in a decision tree, because of the fragmentation of the example space imposed by the restriction to nonoverlapping rules. Rule learners do not make such a restriction and are thus less susceptible to this problem. An extreme example for this problem has been provided by Cendrowska (1987), who showed that the minimal decision tree for the concept `x` defined as

```
IF A = 3 AND B = 3 THEN Class = x
IF C = 3 AND D = 3 THEN Class = x
```

has 10 interior nodes and 21 leafs assuming that each attribute `A ...D` can be instantiated with three different values.

Finally, propositional rule learning algorithms extend naturally to the framework of *inductive logic programming* framework, where the goal is basically the induction of a rule set in first-order logic, e.g., in the form of a Prolog program.[3] First-order background knowledge can also be used for decision tree induction (Blockeel & De Raedt, 1998; Kramer, 1996; Lavrač, Džeroski, & Grobelnik, 1991; Watanabe & Rendell, 1991), but once more, Watanabe and Rendell (1991) have noted that first-order decision trees are usually more complex than first-order rules.

1.6 Descriptive Data Mining: Induction of Patterns

While a decision tree and a set of rules represent a model (a theory) that can be used for classification and/or prediction, the goal of data analysis may be different. Instead of model construction, the goal may be the discovery of individual patterns/rules describing regularities in the data. This form of data analysis is referred to as *descriptive induction* and is frequently used in exploratory data analysis.

As opposed to decision tree and rule set induction, which result in classification models, *association rule learning* is an unsupervised learning method, with no class labels assigned to the examples. Another method for unsupervised learning is *clustering*, while *subgroup discovery*—aimed at finding descriptions of interesting population subgroups—is a descriptive induction method for pattern learning, but is at the same time a form of supervised learning due to a defined property of interest acting as a class.

[3] Prolog is a programming language, enabling knowledge representation in first-order logic (Lloyd, 1987; Sterling & Shapiro, 1994). We will briefly return to learning in first-order logic in Sect. 1.7; a systematic treatment of relational rule learning can be found in Chap. 5.

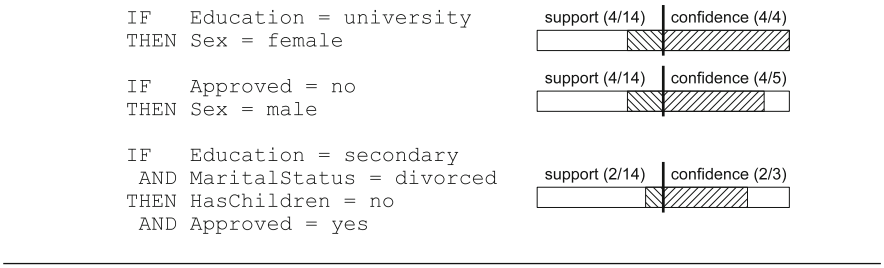

Fig. 1.5 Three rules induced by an association rule learning algorithm

1.6.1 Association Rule Learning

The problem of inducing *association rules* (Agrawal, Mannila, Srikant, Toivonen, & Verkamo, 1995) has received much attention in the data mining community. It is defined as follows: given a set of transactions (examples), where each transaction is a set of items, an association rule is an expression of the form B $\rightarrow$ H, where B and H are sets of items, and B $\rightarrow$ H is interpreted as IF B THEN H, meaning that the transactions in a database which contain B tend to contain H as well.

Figure 1.5 shows three examples for association rules that could be discovered in the dataset of Table 1.1. The first rule states that in this dataset, all people with a university education were female. This rule is based on four observations in the dataset. The fraction of entries in the database that satisfy all conditions (both in body and head) is known as the *support* of the rule. Thus, the support of the rule is the ratio of the number of records having true values for all items in B and H to the number of all records in the database. As 4 of a total of 14 persons are both female and have university education, the support of the first rule is $4/14 \approx 0.286$.

The second rule also has a support of $4/14$, because four people in the database do not approve and are male. However, in this case, the strength of the rule is not as strong as in the previous case, because only $4/5 = 0.8$ of all persons that do not approve were actually male. This value is called the *confidence* of the rule. It is calculated as the ratio of the number of records having true values for all items in B and H to the number of records having true values for all items in B.

Unlike with classification rules, the head of an association rule may also contain a conjunction of conditions. This is illustrated by the third rule, which states that divorced people with secondary education typically have no children and approve.

In all rules there is no distinction between the class attribute and all other attributes: the class attribute may appear on any side of the rule or not at all. In fact, typically association rules are learned from databases with binary features (called *items*) without any dedicated class attribute. Thus association rule discovery is an unsupervised learning task. Most algorithms, such as the well-known APRIORI

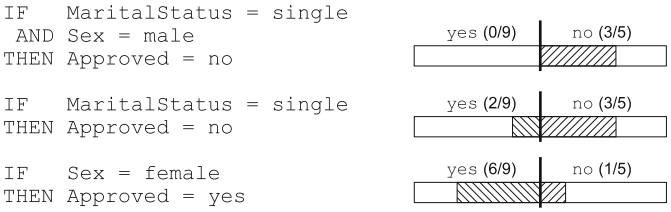

Fig. 1.6 Three subgroup descriptions induced by a subgroup discovery algorithm

algorithm (Agrawal et al., 1995), find all association rules that satisfy minimum support and minimum confidence constraints.

An in-depth survey of association rule discovery is beyond the scope of this book, and, indeed, the subject has already been covered in other monographs (Adamo, 2000; Zhang & Zhang, 2002). We will occasionally touch upon the topic when it seems appropriate (e.g., the level-wise search algorithm, which forms the basis of APRIORI and related techniques, is briefly explained in Sect. 6.3.2), but for a systematic treatment of the subject we refer the reader to the literature.

1.6.2 Subgroup Discovery

In subgroup discovery the task is to find sufficiently large population subgroups that have a significantly different class distribution than the entire population (the entire dataset). Subgroup discovery results in individual rules, where the rule conclusion is a class (the property of interest). The main difference between learning of classification rules and subgroup discovery is that the latter induces single rules (subgroups) of interest, which aim at revealing interesting properties of groups of instances, not necessarily at forming a rule set used for classification.

Figure 1.6 shows three subgroup descriptions that have been induced with the MAGNUM OPUS descriptive rule learning system (Webb, 1995).[4] While the first and third rules could also be found by classification rule algorithms (cf. Fig. 1.4), the second rule would certainly not be found because it has a comparably low predictive quality. There are almost as many single persons that approve than there are singles that do not approve. Nevertheless, this rule can be considered to be an interesting subgroup because the class distribution of covered instances (2 yes and 3 no) is significantly different than the distribution in the entire dataset (9 yes and 5 no). Conversely, a classification rule algorithm would not find the first rule because if we accept the second rule for classification, adding the first one does not improve

[4]The rules are taken from Kralj Novak, Lavrač, and Webb (2009).

classification performance, i.e., it is *redundant* with respect to the second rule. Finally, note that these three rules do not cover all the examples. While it is typically considered important that each rule covers a significant number of examples, it is not necessary that each example be covered by some rule, because the rules will not be used for prediction.

Subgroup discovery and related techniques are covered in depth in Chap. 11 of this book.

1.7 Relational Data Mining

Both predictive and descriptive data mining are usually performed on a single database relation, consisting of examples represented with values for a fixed number of attributes. However, in practice, the data miner often has to face more complex scenarios. Suppose that data is stored in several tables, e.g., it has a *relational database* form. In this case the data has to be transformed into a single table in order to be able to use standard data mining techniques. The most common data transformation approach is to select one table as the main table to be used for learning, and try to incorporate the contents of other tables by aggregating the information contained in the tables into summary attributes, which are added to the main table. The problem with such transformations is that some information may be lost while the aggregation may also introduce artifacts, possibly leading to inappropriate data mining results. What one would like to do is to leave data conceptually unchanged and rather use data mining tools that can deal with multirelational data.

Integrating data from multiple tables through joins or aggregation can cause loss of meaning or information. Suppose we are given two relations: `customer(CID,Name,Age,SpendALot)` encodes the ID, name, and age of a customer, and the information whether this customer spends a lot, and `purchase(CID,ProdID,Date,Value,PaymentMode)` encodes a single purchase by a customer with a given ID. Each customer can make multiple purchases, and we are interested in characterizing customers that spend a lot. Integrating the two relations via a natural join will result in a relation `purchase1(CID,Name,Age,SpendALot,ProdID,Date,Value,PaymentMode)`. However, this is problematic because now each row corresponds to a purchase and not to a customer, and we intend to analyze our information with respect to customers. An alternative would be to aggregate the information contained in the `purchase` relation. One possible aggregation could be the relation `customer1(CID,Name,Age,NofPurchases,TotalValue,SpendALot)`, which aggregates the number of purchases and their total value into new attributes. Naturally, some information has been lost during the aggregation process.

The following pattern can be discovered by a relational rule learning system if the relations `customer` and `purchase` are considered together.

```
customer(CID,Name,Age,yes):-
      Age > 30,
      purchase(CID,PID,D,Value,PM),
      PM = creditcard,
      Value > 100.
```

This pattern, written in a Prolog-like syntax, says: *'a customer spends a lot if she is older than 30, has purchased a product of value more than 100 and paid for it by credit card.'* It would not be possible to induce such a pattern from either of the relations `purchase1` and `customer1` considered on their own.

We will return to relational learning in Chap. 5, where we take a feature-based view on the problem.

1.8 Conclusion

This chapter briefly described several aspects of machine learning and data mining, aiming to provide the background and basic understanding of the topics presented in this book. To conclude, let us make some speculations about future developments in data mining.

With regard to data mining research, every year the research community addresses new open problems and new problem areas, for many of which data mining is able to provide value-added answers and results. Because of the interdisciplinary nature of data mining, there is a big inflow of new knowledge, widening the spectrum of problems that can be solved by the use of this technology. Another reason why data mining has a scientific and commercial future was given by Friedman (1998): "Every time the amount of data increases by a factor of 10, we should totally rethink how we analyze it."

To achieve its full commercial exploitation, data mining is still lacking the standardization to the degree of, for example, the standardization available for database systems. There are initiatives in this direction, which will diminish the monopoly of expensive closed-architecture systems. For data mining to be truly successful it is important that data mining tools become available in major database products as well as in standard desktop applications (e.g., spreadsheets). Other important recent developments are open source data mining services, tools for online construction of data mining workflows, as well as the terminology and ingredients of data mining through the development of a data mining ontology (Lavrač, Kok, de Bruin, & Podpečan, 2008; Lavrač, Podpečan, Kok, & de Bruin, 2009).

In the future, we envisage intensive development and increased usage of data mining in specific domain areas, such as bioinformatics, multimedia, text and web data analysis. On the other hand, as data mining can be used for building surveillance systems, recent research also concentrates on developing algorithms for mining databases without compromising sensitive information (Agrawal & Srikant, 2000). A shift towards automated use of data mining in practical systems is also expected to become very common.

http://www.springer.com/978-3-540-75196-0

Foundations of Rule Learning

Fürnkranz, J.; Gamberger, D.; Lavrac, N.

2012, XVIII, 334 p., Hardcover
ISBN: 978-3-540-75196-0