Studies in Big Data 13

Han Liu
Alexander Gegov
Mihaela Cocea

Rule Based Systems for Big Data

A Machine Learning Approach

Springer

Studies in Big Data

Volume 13

Series editor

Janusz Kacprzyk, Polish Academy of Sciences, Warsaw, Poland
e-mail: kacprzyk@ibspan.waw.pl

About this Series

The series "Studies in Big Data" (SBD) publishes new developments and advances in the various areas of Big Data- quickly and with a high quality. The intent is to cover the theory, research, development, and applications of Big Data, as embedded in the fields of engineering, computer science, physics, economics and life sciences. The books of the series refer to the analysis and understanding of large, complex, and/or distributed data sets generated from recent digital sources coming from sensors or other physical instruments as well as simulations, crowd sourcing, social networks or other internet transactions, such as emails or video click streams and other. The series contains monographs, lecture notes and edited volumes in Big Data spanning the areas of computational intelligence incl. neural networks, evolutionary computation, soft computing, fuzzy systems, as well as artificial intelligence, data mining, modern statistics and Operations research, as well as self-organizing systems. Of particular value to both the contributors and the readership are the short publication timeframe and the world-wide distribution, which enable both wide and rapid dissemination of research output.

More information about this series at http://www.springer.com/series/11970

Han Liu · Alexander Gegov
Mihaela Cocea

Rule Based Systems for Big Data

A Machine Learning Approach

Springer

Han Liu
School of Computing
University of Portsmouth
Portsmouth
UK

Mihaela Cocea
School of Computing
University of Portsmouth
Portsmouth
UK

Alexander Gegov
School of Computing
University of Portsmouth
Portsmouth
UK

ISSN 2197-6503 ISSN 2197-6511 (electronic)
Studies in Big Data
ISBN 978-3-319-23695-7 ISBN 978-3-319-23696-4 (eBook)
DOI 10.1007/978-3-319-23696-4

Library of Congress Control Number: 2015948735

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

Preface

Just as water retains no constant shape, so in warfare there are no constant conditions
—Lionel Giles, The Art of War by Sun Tzu

The ideas introduced in this book explore the relationships among rule-based systems, machine learning and big data. Rule-based systems are seen as a special type of expert systems, which can be built by using expert knowledge or learning from real data. From this point of view, the design of rule-based systems can be divided into expert-based design and data-based design. In the present big data era, the latter approach of design, which typically follows machine learning, has been increasingly popular for building rule-based systems. In the context of machine learning, a special type of learning approach is referred to as inductive learning, which typically involves the generation of rules in the form of either a decision tree or a set of if-then rules. The rules generated through the adoption of the inductive learning approach compose a rule-based system.

The focus of this book is on the development and evaluation of rule-based systems in terms of accuracy, efficiency and interpretability. In particular, a unified framework for building rule-based systems, which consists of the operations of rule generation, rule simplification and rule representation, is presented. Each of these operations is detailed using specific methods or techniques. In addition, this book also presents some ensemble learning frameworks for building ensemble rule-based systems. Each of these frameworks involves a specific way of collaborations between different learning algorithms. All theories mentioned above are designed to address the issues relating to overfitting of training data, which arise with most learning algorithms and make predictive models perform well on training data but poorly on test data.

Machine learning does not only have a scientific perspective but also a philosophical one. This implies that machine learning is philosophically similar to human learning. In fact, machine learning is inspired by human learning in order to

simulate the process of learning in computer software. In other words, the name of machine learning indicates that machines are capable of learning. However, people in other fields have criticized the capability of machine learning by saying that machines are neither able to learn nor outperform people intellectually. The argument is that machines are invented by people and their performance is totally dependent on the design and implementation by engineers and programmers. It is true that machines are controlled by programs in executing instructions. However, if a program is an implementation of a learning method, then the machine will execute the program to learn something. On the other hand, if a machine is thought to be never superior to people, this will imply that in human learning students would never be superior to their teachers. This is not really true, especially if a student has the strong capability to learn independently without being taught. Therefore, this should also be valid in machine learning if a good learning method is embedded in the machine.

In recent years, data mining and machine learning have been used as alternative terms in the same research area. However, the authors consider this as a misconception. According to them, data mining and machine learning are different in both philosophical and practical aspects.

In terms of philosophical aspects, data mining is similar to human research tasks and machine learning is similar to human learning tasks. From this point of view, the difference between data mining and machine leaning is similar to the difference between human research and learning. In particular, data mining, which acts as a researcher, aims to discover something new from unknown properties, whereas machine learning, which acts as a learner, aims to learn something new from known properties.

In terms of practical aspects, although both data mining and machine learning involve data processing, the data processed by the former needs to be primary, whereas the data processed by the latter needs to be secondary. In particular, in data mining tasks, the data has some patterns which are previously unknown and the aim is to discover the new patterns from the data. In contrast, in machine learning tasks, the data has some patterns which are known in general but are not known to the machine and the aim is to make the machine learn the patterns from the data. On the other hand, data mining is aimed at knowledge discovery, which means that the model built is used in a white box manner to extract the knowledge which is discovered from the data and is communicated to people. In contrast, machine learning is aimed at predictive modelling, which means that the model built is used in a black box manner to make predictions on unseen instances.

The scientific development of the theories introduced in this book is philosophically inspired by three main theories—namely information theory, system theory and control theory. In the context of machine learning, information theory generally relates to transformation from data to information/knowledge. In the context of system theory, a machine learning framework can be seen as a learning

system which consists of different modules including data collection, data pre-processing, training, testing and deployment. In addition, single rule-based systems are seen as systems, each of which typically consists of a set of rules and could also be a subsystem of an ensemble rule-based system by means of a system of systems. In the context of control theory, learning tasks need to be controlled effectively and efficiently, especially due to the presence of big data.

Han Liu
Alexander Gegov
Mihaela Cocea

Acknowledgments

The first author would like to thank the University of Portsmouth for awarding him the funding to conduct the research activities that produced the results disseminated in this book. Special thanks must go to his parents Wenle Liu and Chunlan Xie as well as his brother Zhu Liu for the financial support during his academic studies in the past as well as the spiritual support and encouragement for his embarking on a research career in recent years. In addition, the first author would also like to thank his best friend Yuqian Zou for the continuous support and encouragement during his recent research career that have facilitated significantly his involvement in the writing process for this book.

The authors would like to thank the academic editor for the Springer Series in Studies in Big Data Prof. Janusz Kacprzyk and the executive editor for this series Dr. Thomas Ditzinger for the useful comments provided during the review process. These comments have been very helpful for improving the quality of the book.

Contents

Chapter 1
Introduction

1.1 Background of Rule Based Systems

Expert systems have been increasingly popular for commercial applications. A rule based system is a special type of expert system. The development of rule based systems began in the 1960s but became popular in the 1970s and 1980s [1]. A rule based system typically consists of a set of if-then rules, which can serve many purposes such as decision support or predictive decision making in real applications. One of the main challenges in this area is the design of such systems which could be based on both expert knowledge and data. Thus the design techniques can be divided into two categories: expert based construction and data based construction. The former follows a traditional engineering approach, while the later follows a machine learning approach. For both approaches, the design of rule based systems could be used for practical tasks such as classification, regression and association.

This book recommends the use of the data based approach instead of the expert based approach. This is because the expert based approach has some limitations which can usually be overcome by using the data based approach. For example, expert knowledge may be incomplete or inaccurate; some of experts' points of view may be biased; engineers may misunderstand requirements or have technical designs with defects. When problems with high complexity are dealt with, it is difficult for both domain experts and engineers to have all possible cases considered or to have perfect technical designs. Once a failure arises with an expert system, experts or engineers may have to find the problem and fix it by reanalyzing or redesigning. However, the real world has been filled with big data. Some previously unknown information or knowledge could be discovered from data. Data could potentially be used as supporting evidence to reflect some useful and important pattern by using modelling techniques. More importantly, the model could be revised automatically as a database is updated in real time when data based modelling technique is used. Therefore, the data based approach would be more suitable

H. Liu et al., *Rule Based Systems for Big Data*,
Studies in Big Data 13, DOI 10.1007/978-3-319-23696-4_1

than the expert based approach for construction of complex rule based systems. This book mainly focuses on theoretical and empirical studies of rule based systems for classification in the context of machine learning.

Machine learning is a branch of artificial intelligence and involves two stages: training and testing. Training aims to learn something from known properties by using learning algorithms and testing aims to make predictions on unknown properties by using the knowledge learned in the training stage. From this point of view, training and testing are also known as learning and prediction respectively. In practice, a machine learning task aims to build a model that is further used to make predictions by adopting learning algorithms. This task is usually referred to as predictive modelling. Machine learning could be divided into two types: supervised learning and unsupervised learning, in accordance with the form of learning. Supervised learning means learning with a teacher because all instances from a training set are labelled. The aim of this type of learning is to build a model by learning from labelled data and then to make predictions on other unlabelled instances with regard to the value of a predicted attribute. The predicted value of an attribute could be either discrete or continuous. Therefore, supervised learning could be involved in both classification and regression tasks for categorical prediction and numerical prediction, respectively. In contrast, unsupervised learning means learning without a teacher. This is because all instances from a training set are unlabelled. The aim of this type of learning is to find previously unknown patterns from data sets. It includes association, which aims to identify correlations between attributes, and clustering, which aims to group objects based on similarity measures.

On the other hand, machine learning algorithms are popularly used in data mining tasks to discover some previously unknown pattern. This task is usually referred to as knowledge discovery. From this point of view, data mining tasks also involve classification, regression, association and clustering. Both classification and regression can be used to reflect the correlation between multiple independent variables and a single dependent variable. The difference between classification and regression is that the former typically reflects the correlation in qualitative aspects, whereas the latter reflects in quantitative aspects. Association is used to reflect the correlation between multiple independent variables and multiple dependent variables in both qualitative and quantitative aspects. Clustering can be used to reflect patterns in relation to grouping of objects.

In data mining and machine learning, automatic induction of classification rules has become increasingly popular in commercial applications such as predictive decision making systems. In this context, the methods for generating classification rules can be divided into two categories: 'divide and conquer' and 'separate and conquer'. The former is also known as Top-Down Induction of Decision Trees (TDIDT), which generates classification rules in the intermediate form of a decision tree such as ID3, C4.5 and C5.0 [2]. The latter is also known as covering approach [3], which generates if-then rules directly from training instances such as Prism [4]. The ID3 and Prism algorithms are described in detail in Chap. 3.

Most rule learning methods suffer from overfitting of training data, which is termed as overfitting avoidance bias in [3, 5, 6]. In practice, overfitting may results

in the generation of a large number of complex rules. This not only increases the computational cost, but also lowers the accuracy in predicting further unseen instances. This has motivated the development of pruning algorithms with respect to the reduction of overfitting. Pruning methods could be subdivided into two categories: pre-pruning and post-pruning [3]. For divide and conquer rule learning, the former pruning strategy aims to stop the growth of decision trees in the middle of the training process, whereas the latter pruning strategy aims to simplify a set of rules, which is converted from the generated decision tree, after the completion of the training process. For separate and conquer rule learning, the former pruning strategy aims to stop the specialization of each single rule prior to its normal completion whereas the latter pruning strategy aims to simplify each single rule after the completion of the rule generation. Some theoretic pruning methods, which are based on J-measure [7], are described in detail in Chap. 4.

The main objective in prediction stage is to find the first firing rule by searching through a rule set. As efficiency is important, a suitable structure is required to effectively represent a rule set. The existing rule representations include decision trees and linear lists. Decision Tree representation is mainly used to represent rule sets generated by the 'divide and conquer' approach. A decision tree has a root and several internal nodes representing attributes and leaf nodes representing classifications as well as branches representing attribute values. On the other hand, linear list representation is commonly used to represent rules generated by 'separate and conquer' approach in the form of 'if-then' rules. These two representations are described in detail in Chap. 5.

Each machine learning algorithm may have its own advantages and disadvantages, which results in the possibility that a particular algorithm may perform well on some datasets but poorly on others, due to its suitability to particular datasets. In order to overcome the above problem and thus improve the overall accuracy of classification, the development of ensemble learning approaches has been motivated. Ensemble learning concepts are introduced in Sect. 1.3 and popular approaches are described in details in Chap. 6.

As mentioned above, most rule learning methods suffer from overfitting of training data, which is due to bias and variance. As introduced in [8], bias means errors originating from learning algorithm whereas variance means errors originating from data. Therefore, it is necessary to reduce both bias and variance in order to reduce overfitting comprehensively. In other words, reduction of overfitting can be achieved through scaling up algorithms or scaling down data. The former way is to reduce the bias on algorithms side whereas the latter way is to reduce the variance on data side. In addition, both ways usually also improve computational efficiency in both training and testing stages.

In the context of scaling up algorithms, if a machine learning task involves the use of a single algorithm, it is necessary to identify the suitability of a particular algorithm to the chosen data. For example, some algorithms are unable to directly deal with continuous attributes such as ID3. For this kind of algorithms, it is required to discretize continuous attributes prior to training stage. A popular method of discretization of continuous attributes is Chi-Merge [9]. The discretization of

continuous attributes usually helps speed up the process of training greatly. This is because the attribute complexity is reduced through discretizing the continuous attributes [8]. However, it is also likely to lead to loss of accuracy. This is because information usually gets lost in some extent after a continuous attribute is discretized as mentioned in [8]. In addition, some algorithms prefer to deal with continuous attributes such as K Nearest Neighbor (KNN) [10] and Support Vector Machine (SVM) [11, 12].

In the context of scaling down data, if the training data is massively large, it would usually result in huge computational costs. In addition, it may also make learning algorithms learn noise or coincidental patterns. In this case, a generated rule set that overfits training data usually performs poorly in terms of accuracy on test data. In contrast, if the size of a sample is too small, it is likely to learn bias from training data as the sample could only have a small coverage for the scientific pattern. Therefore, it is necessary to effectively choose representative samples for training data. With regard to dimensionality, it is scientifically possible that not all of the attributes are relevant to making classifications. In this case, some attributes need to be removed from the training set by feature selection techniques if the attributes are irrelevant. Therefore, it is necessary to examine the relevance of attributes in order to effectively reduce data dimensionality. The above descriptions mostly explain why an algorithm may perform better on some data sets but worse on others. All of these issues mentioned above often arise in machine learning tasks, so the issues also need to be taken into account by rule based classification algorithms in order to improve classification performance. On the basis of above descriptions, it is necessary to pre-process data prior to training stage, which involves dimensionality reduction and data sampling. For dimensionality reduction, some popular existing methods include Principle Component Analysis (PCA) [13], Linear Discriminant Analysis (LDA) [14] and Information Gain based methods [15]. Some popular sampling methods include simple random sampling [16], probabilistic sampling [17] and cluster sampling [18].

In addition to predictive accuracy and computational efficiency, interpretability is also a significant aspect if the machine learning approaches are adopted in data mining tasks for the purpose of knowledge discovery. As mentioned above, machine learning methods can be used for two main purposes. One is to build a predictive model that is used to make predictions. The other one is to discover some meaningful and useful knowledge from data. For the latter purpose, the knowledge discovered is later used to provide insights for a knowledge domain. For example, a decision support system is built in order to provide recommendations to people with regard to a decision. People may not trust the recommendations made by the system unless they can understand the reasons behind the decision making process. From this point of view, it is required to have an expert system which works in a white box manner. This is in order to make the expert system transparent so that people can understand the reasons why the output is derived from the system.

As mentioned above, a rule based system is a special type of expert systems. This type of expert systems works in a white box manner. Higgins justified in [19] that interpretable expert systems need to be able to provide the explanation with

regard to the reason of an output and that rule based knowledge representation makes expert systems more interpretable with the arguments described in the following paragraphs:

A network was conceived in [20], which needs a number of nodes exponential in the number of attributes in order to restore the information on conditional probabilities of any combination of inputs. It is argued in [19] that the network restores a large amount of information that is mostly less valuable.

Another type of network known as Bayesian Network introduced in [21] needs a number of nodes which is the same as the number of attributes. However, the network only restores the information on joint probabilities based on the assumption that each of the input attributes is totally independent of the others. Therefore, it is argued in [19] that this network is unlikely to predict more complex relationships between attributes due to the lack of information on correlational probabilities between attributes.

There are some other methods that fill the gaps that exist in Bayesian Networks by deciding to only choose some higher-order conjunctive probabilities, such as the first neural networks [22] and a method based on correlation/dependency measure [23]. However, it is argued in [19] that these methods still need to be based on the assumption that all attributes are independent of each other.

1.2 Categorization of Rule Based Systems

Rule based systems can be categorized based on the following aspects: *number of inputs and outputs, type of input and output values, type of structure, type of logic, type of rule bases, number of machine learners* and *type of computing environment* [24].

For rule based systems, both inputs and outputs could be single or multiple. From this point of view, rule based systems can be divided into four types [25]: *single-input-single-output, multiple-input-single-output, single-input-multiple-output,* and *multiple-input-multiple-output*. All the four types above can fit the characteristics of association rules. This is because association rules reflect relationships between attributes. An association rule may have a single or multiple rule terms in both antecedent (left hand side) and consequent (right hand side) of the rule. Thus the categorization based on number of inputs and outputs is very necessary in order to make the distinction of association rules.

However, association rules include two special types: classification rules and regression rules, depending on the type of output values. Both classification rules and regression rules may have a single term or multiple rule terms in the antecedent, but can only have a single term in the consequent. The difference between classification rules and regression rules is that the output values of classification rules must be discrete while those of regression rules must be continuous. Thus both classification rules and regression rules fit the characteristics of 'single-input-single-output' or 'multiple-input-single-output' and are seen as a special type of association rules. On the basis of the above description, rule based systems can also be

categorized into three types with respects to both number of inputs and outputs and type of input and output values: *rule based classification systems, rule based regression systems* and *rule based association systems*.

In machine learning, as mentioned in Sect. 1.1, classification rules can be generated in two approaches: divide and conquer, and separate and conquer. The former method is generating rules directly in the form of a decision tree, whereas the latter method produces a list of 'if-then' rules. An alternative structure called rule based networks represents rules in the form of networks, which will be introduced in Chap. 5 in more detail. With respect to structure, rule based systems can thus be divided into three types: *treed rule based systems, listed rule based systems* and *networked rule based systems*.

The construction of rule based systems is based on special types of logic such as deterministic logic, probabilistic logic and fuzzy logic. From this point of view, rule based systems can also be divided into the following types: *deterministic rule based systems, probabilistic rule based systems* and *fuzzy rule based systems*.

As rule based systems can also be in the context of rule bases including single rule bases, chained rule bases and modular rule bases [25]. From this point of view, rule based systems can also be divided into the three types: *standard rule based systems, hierarchical rule based systems* and *networked rule based systems*.

In machine learning context, a single algorithm could be applied to a single data set for training a single learner. It can also be applied to multiple samples of a data set by ensemble learning techniques for construction of an ensemble learner which consists of a group of single learners. In addition, there could also be a combination of multiple algorithms involved in machine learning tasks. From this point of view, rule based systems can be divided into two types according to the number of machine learners constructed: *single rule based systems* and *ensemble rule based systems*.

In practice, an ensemble learning task could be done in a parallel, distributed way or a mobile platform according to the specific computing environments. Therefore, rule based systems can also be divided into the following three types: *parallel rule based systems, distributed rule based systems* and *mobile rule based systems*.

The categorizations described above aim to specify the types of rule based systems as well as to give particular terminologies for different application areas in practice. In this way, it is easy for people to distinguish different types of rule based systems when they are based on different theoretical concepts and practical techniques or they are used for different purposes in practice.

1.3 Ensemble Learning

As mentioned in Sect. 1.1, ensemble learning is usually adopted to improve overall accuracy. In detail, this purpose can be achieved through scaling up algorithms or scaling down data. Ensemble learning can be done both in parallel and sequentially.

In the former way, there are no collaborations among different learning algorithms and only their predictions are combined together for the final prediction making [26]. In this context, the final prediction is typically made by voting in classification and by averaging in regression. In the latter way of ensemble learning, the first algorithm learns a model from data and then the second algorithm learns to correct the former one, and so on [26]. In other words, the model built by the first algorithm is further corrected by the following algorithms sequentially.

The parallel ensemble learning approach can be achieved by combining different learning algorithms, each of which generates a model independently on the same training set. In this way, the predictions of the models generated by these algorithms are combined to predict unseen instances. This approach belongs to scaling up algorithms because different algorithms are combined in order to generate a stronger hypothesis. In addition, the parallel ensemble learning approach can also be achieved by using a single base learning algorithm to generate models independently on different sample sets of training instances. In this context, the sample set of training instances can be provided by horizontally selecting the instances with replacement or vertically selecting the attributes without replacement. This approach belongs to scaling down data because the training data is preprocessed to reduce the variance that exists on the basis of the attribute-values.

In sequential ensemble learning approach, accuracy can also be improved through scaling up algorithms or scaling down data. In the former way, different algorithms are combined in the way that the first algorithm learns to generate a model and then the second algorithm learns to correct the model, and so on. In this way, the training of the different algorithms takes place on the same data. In the latter way, in contrast, the same algorithm is used iteratively on different versions of the training data. In each iteration, a model is generated and evaluated using the validation data. According to the estimated quality of the model, the training instances are weighted to different extents and then used for the next iteration. In the testing stage, these models generated at different iterations make predictions independently and their predictions are then combined to predict unseen instances.

For both parallel and sequential ensemble learning approaches, voting is involved in the testing stage when the independent predictions are combined to make the final prediction on an unseen instance. Some popular methods of voting include equal voting, weighted voting and naïve Bayesian voting [26]. Some popular approaches of ensemble learning for generation of classification rules are described in Chap. 6 in more depth.

1.4 Chapters Overview

This book consists of nine main chapters namely, introduction, preliminary of rule based systems, generation of classification rules, simplification of classification rules, representation of classification rules, ensemble learning approaches, interpretability analysis, case studies and conclusion. The rest of this book is organized as follows:

Chapter 2 describes some fundamental concepts that strongly relate to rule based systems and machine learning such as discrete mathematics, statistics, if-then rules, algorithms, logic and statistical measures of rule quality. In addition, this chapter also describes a unified framework for construction of single rule based classification systems, as well as the way to construct an ensemble rule based classification systems by means of a system of systems.

Chapter 3 introduces two approaches of rule generation namely, 'divide and conquer' and 'separate and conquer'. In particular, some existing rule learning algorithms are illustrated in detail. These algorithms are also discussed comparatively with respects to their advantages and disadvantages.

Chapter 4 introduces two approaches of rule simplification namely, information theoretic pre-pruning and information theoretic post-pruning. In particular, some existing rule pruning algorithms are illustrated. These algorithms are also discussed comparatively with respects to their advantages and disadvantages.

Chapter 5 introduces three techniques for representation of classification rules namely, decision trees, linear lists and rule based networks. In particular, these representations are illustrated using examples in terms of searching for firing rules. These techniques are also discussed comparatively in terms of computational complexity and interpretability.

Chapter 6 introduces three approaches of ensemble learning namely, parallel learning, sequential learning and hybrid learning. In particular, some popular methods for ensemble learning are illustrated in detail. These methods are also discussed comparatively with respects to their advantages and disadvantages.

Chapter 7 introduces theoretical aspects of interpretability on rule based systems. In particular, some impact factors are identified and how these factors have an impact on interpretability is also analyzed. In addition, some criteria for evaluation on interpretability are also listed.

Chapter 8 introduces case studies on big data. In particular, the methods and techniques introduced in Chaps. 3, 4, 5 and 6 are evaluated through theoretical analysis and empirical validation using large data sets in terms of variety, veracity and volume.

Chapter 9 summaries the contributions of this book in terms of theoretical significance, practical importance, methodological impact and philosophical aspects. Further directions of this research area are also identified and highlighted.

References

1. Partridge, D., Hussain, K.M.: Knowledge Based Information Systems. Mc-Graw Hill, London (1994)
2. Quinlan, J.R.: C 4.5: Programs for Machine Learning. Morgan Kaufman, San Mateo (1993)
3. Furnkranz, J.: Separate-and-conquer rule learning. Artif. Intell. Rev. **13**, 3–54 (1999)
4. Cendrowska, J.: PRISM: an algorithm for inducing modular rules. Int. J. Man Mach. Stud. **27**, 349–370 (1987)
5. Schaffer, C.: Overfitting avoidance as bias. Mach. Learn. **10**, 153–178 (1993)

6. Wolpert, D.H.: On Overfitting Avoidance as Bias. Santa Fe, NM (1993)
7. Smyth, P., Rodney, G.M.: An information theoretic approach to rule induction from databases. IEEE Trans. Knowl. Data Eng. **4**(4), 301–316 (1992)
8. Brain, D.: Learning From Large Data: Bias, Variance, Sampling, and Learning Curves. Deakin University, Victoria (2003)
9. Kerber, R.: ChiMerge: discretization of numeric attribute. In Proceeding of the 10th National Conference on Artificial Intelligence (1992)
10. Altman, N.S.: An introduction to kernel and nearest-neighbour nonparametric regression. Am. Stat. **46**(3), 175–185 (1992)
11. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Pearson Education Inc, New Jersey (2006)
12. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Section 16.5. support vector machines. In Numerical Recipes: The Art of Scientific Computing, 3rd edn. Cambridge University Press, New York (2007)
13. Jolliffe, I.T.: Principal component analysis. Springer, New York (2002)
14. Yu, H., Yang, J.: A direct LDA algorithm for high diomensional data- with application to face recognition. Pattern Recogn. **34**(10), 2067–2069 (2001)
15. Azhagusundari, B., Thanamani, A.S.: Feature selection based on information gain. Int. J. Innovative Technol Exploring Eng. **2**(2), 18–21 (2013)
16. Yates, D.S., David, S.M., Daren, S.S.: The Practice of Statistics, 3rd edn. Freeman, New York (2008)
17. Deming, W.E.: On probability as a basis for action. Am. Stat. **29**(4), 146–152 (1975)
18. Kerry and Bland: Statistics notes: the intracluster correlation coefficient in cluster randomisation. Br. Med. J. **316**, 1455–1460 (1998)
19. Higgins, C.M.: Classification and Approximation with Rule-Based Networks. California Institute of Technology, California (1993)
20. Uttley, A.M.: The design of conditional probability computers. Inf. Control **2**, 1–24 (1959)
21. Kononenko, I.: Baysain neural networks. Biol. Cybern. **61**, 361–370 (1989)
22. Rosenblatt, F.: Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. Spartan Books, Washington, DC (1962)
23. Ekeberg, O., Lansner, A.: Automatic generation of internal representations in a probabilistic artificial neural network. In Proceedings of First European Conference on Neural Networks (1988)
24. Liu, H., Gegov, A., Stahl, F.: Categorization and construction of rule based systems. In 15th International Conference on Engineering Applications of Neural Networks, Sofia (2014)
25. Gegov, A.: Fuzzy Networks for Complex Systems: A Modular Rule Base Approach. Springer, Berlin (2010)
26. Kononenko, I., Kukar, M.: Machine Learning and Data Mining: Introduction to Principles and Algorithms. Horwood Publishing Limited, Chichester, West Sussex (2007)

Chapter 2
Theoretical Preliminaries

As mentioned in Chap. 1, some fundamental concepts strongly relate to rule based systems and machine learning, including discrete mathematics, statistics, if-then rules, algorithms, logic and statistical measures of rule quality. This chapter illustrates these concepts in detail. In addition, this chapter also describes a unified framework for construction of single rule based classification systems, as well as the way to construct an ensemble rule based classification systems by means of a system of systems.

2.1 Discrete Mathematics

Discrete mathematics is a branch of mathematical theory, which includes three main topics, namely mathematical logic, set theory and graph theory. In this book, rule learning methods introduced in Chap. 3 are strongly based on Boolean logic, which is a theoretical application of mathematical logic in computer science. As mentioned in Sect. 1.1, a rule based system consists of a set of rules. In other words, rules are basically stored in a set, which is referred to as rule set. In addition, the data used in machine learning tasks is usually referred to as a dataset. Therefore, set theory is also strongly related to the materials in this book. The development of rule based networks, which is introduced in Chap. 5, is fundamentally based on graph theory. On the basis of above description, this subsection introduces in more detail the three topics as part of discrete mathematics with respects to their concepts and connections to the context of this book.

Mathematical logic includes the propositional connectives namely conjunction, disjunction, negation, implication and equivalence. Conjunction is also referred to as AND logic in computer science and denoted by $F = a \wedge b$. The conjunction could be illustrated by the truth table (Table 2.1).

Table 2.1 essentially implies that the output is positive if and only if all inputs are positive in AND logic. In other words, if any one of the inputs is negative, it would result in a negative output. In practice, the conjunction is widely used to make judgments especially on safety critical judgment. For example, it can be used for security check systems and the security status is positive if and only if all

H. Liu et al., *Rule Based Systems for Big Data*,
Studies in Big Data 13, DOI 10.1007/978-3-319-23696-4_2

Table 2.1 Conjunction truth table

a	b	F
0	0	0
0	1	0
1	0	0
1	1	1

Table 2.2 Disjunction truth table

a	b	F
0	0	0
0	1	1
1	0	1
1	1	1

parameters relating to the security are positive. In this book, the conjunction is typically used to judge if a rule is firing and more details about it are introduced in Sect. 1.4.3.

Disjunction is also referred to as OR logic in computer science and denoted by $F = a \vee b$. The disjunction is illustrated by the truth table (Table 2.2).

Table 2.2 essentially implies that the output would be negative if and only if all of the inputs are negative in OR logic. In other words, if any one of the inputs is positive, then it would result in a positive output. In practice, it is widely used to make judgments on alarm system. For example, an alarm system would be activated if any one of the parameters appears to be negative.

Implication is popularly used to make deductions, and is denoted by $F = a \rightarrow b$. The implication is illustrated by the truth table (Table 2.3).

Table 2.3 essentially implies that 'a' is defined as an antecedent and 'b' as a consequent. In this context, it supposes that the consequent would be deterministic if the antecedent is satisfied. In other words, 'a' is seen as the adequate but not necessary condition of 'b', which means if 'a' is true then 'b' will definitely be true, but b may be either true or false otherwise. In contrast, if 'b' is true, it is not necessarily due to that 'a' is true. This can also be proved as follows:

$$F = a \bullet b \Leftrightarrow \neg a \vee b$$

The notation $\neg a \vee b$ is illustrated by the truth table (Table 2.4). In particular, it can be seen from the table that the output is negative if and only if 'a' provides a positive input but 'b' provides a negative one.

Table 2.4 essentially implies that the necessity condition that the output is negative is to have 'a' provide a positive input. This is because the output will definitely be positive when 'a' provides a negative input, which makes '$\neg$a' provide a positive input. In contrast, if 'a' provides a positive input and 'b' provides a negative input, then the output will be negative.

It can be seen from Tables 2.3 and 2.4 that the outputs from the two tables are exactly same. Therefore, Table 2.3 indicates that if an antecedent is satisfied then

Table 2.3 Implication truth table

a	b	F
0	0	1
0	1	1
1	0	0
1	1	1

Table 2.4 Negation truth table

a	b	¬a	F
0	0	1	1
0	1	1	1
1	0	0	0
1	1	0	1

the consequent can be determined. Otherwise, the consequent would be non-deterministic. In this book, the concept of implication is typically used in the form of if-then rules for predicting classes. The concept of if-then rules is introduced in Sect. 2.3.

Besides, negation and equivalence are actually not applied to the research methodology in this book. Therefore, they are not introduced in detail here, but the interested reader can find these two concepts in [1].

Set theory is another part of discrete mathematics as mentioned earlier. A set is defined as a collection of elements. The elements maybe numbers, points and names etc., which are not ordered nor repetitive, i.e. the elements can be stored in any order and are distinct from each other. As introduced in [2, 3], an element 'e' has a membership in a set 'S', which is denoted by 'e ∈ S' and it is said that element 'e' belongs to set 'S'. The fact that the element 'e' is not a member of set 'S' is denoted by 'e ∈ S' and it is said that element 'e' does not belong to set 'S'. In this book, set theory is used in the management of data and rules, which are referred to as data set and rule set respectively. A data set is used to store data and each element represents a data point. In this book, a data point is usually referred to as an instance. A rule set is used to store rules and each element represents a rule. In addition, a set can have a number of subsets depending on the number of elements. The maximum number of subsets for a set would be 2^n, where n is the number of elements in the set. There are also some operations between sets such as union, intersection and difference, which are not relevant to the materials in this book. Therefore, the concepts relating to these operations are not introduced here—more details are available in [1, 3].

On the other hand, relations can be defined between sets. A binary relation exists when two sets are related. For example, there are two sets denoted as 'Student' and 'Course' respectively. In this context, there would be a mapping from students and courses, and each mapping is known as an ordered pair. For example, each student can register on one course only, but a course could have many students or no students, which means that each element in the set 'Student' is only mapped to one element in the set 'Course', but an element in the latter set may be mapped to many

elements in the former set. Therefore, this is a many-to-one relation. This type of relation is also known as a function. In contrast, if the university regulations allow that a student may register on more than one course, the relation would become many-to-many and is not a function any more. Therefore, a function is generally defined as a many-to-one relation. In the above example, the set 'Student' is regarded as the domain and the set 'Course' as range. In this book, each rule in a rule set actually acts as a particular function to reflect the mapping from input space (domain) to output space (range).

Graph theory is also a part of discrete mathematics as mentioned earlier in this subsection. It is popularly used in data structures such as binary search trees and directed or undirected graphs. A tree typically consists of a root node and some internal nodes as well as some leaf nodes as illustrated in Fig. 2.1. In this figure, node A is the root node of the tree; node B and C are two internal nodes; and node D, E, F and G are four leaf nodes. A tree could be seen as a top-down directed graph. This is because the search strategy applied to trees is in top-down approach from the root node to the leaf nodes. The search strategy could be divided into two categories: depth first search and breadth first search. In the former strategy, the search is going through in the following order: A → B → D → E → C → F → G. In contrast, in the latter strategy, the search would be in a different order: A → B → C → D → E → F → G. In this book, the tree structure is applied to the concept of decision tree to graphically represent a set of if-then rules. More details about this are introduced in Chap. 5.

In contrast to trees, there is also a type of horizontally directed graphs in one/two way(s) as illustrated in Figs. 2.2 and 2.3. For example, a feed-forward neural network is seen as a one way directed graph and a feedback neural network as a two way directed graph.

In a directed graph, what could be judged is on the reachability between nodes depending on the existence of connections. For example, looking at Fig. 2.2, it can only be judged that it is reachable from node A to node C but unreachable in the

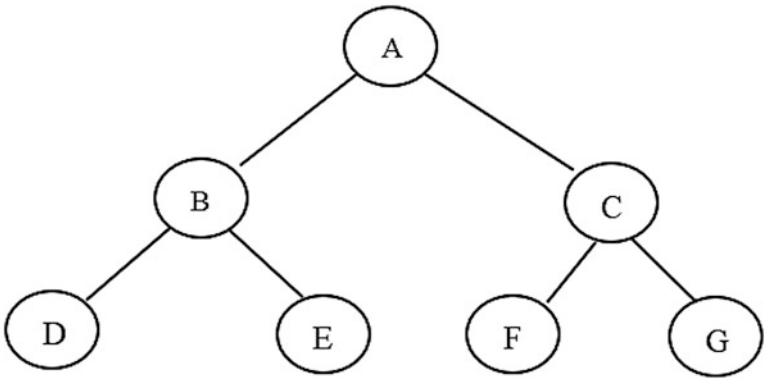

Fig. 2.1 Example of tree structure

opposite way. This is because there is only a one way connection from node A to node C. In contrast, there is a two way connection between node A and node C through looking at Fig. 2.3. Therefore, it can be judged that it is reachable between the two nodes, i.e. it is reachable in both ways (A → C and C → A). In this book, the concept of directed graphs is applied to a special type of rule representation known as rule based network for the purpose of predictive modelling. Related details are introduced in Chap. 5.

In addition, a graph could also be undirected, which means that in a graphical representation the connections between nodes would become undirected. This concept is also applied to network based rule representation but the difference to application of directed graphs is that the purpose is for knowledge representation. More details about this are introduced in Chap. 5.

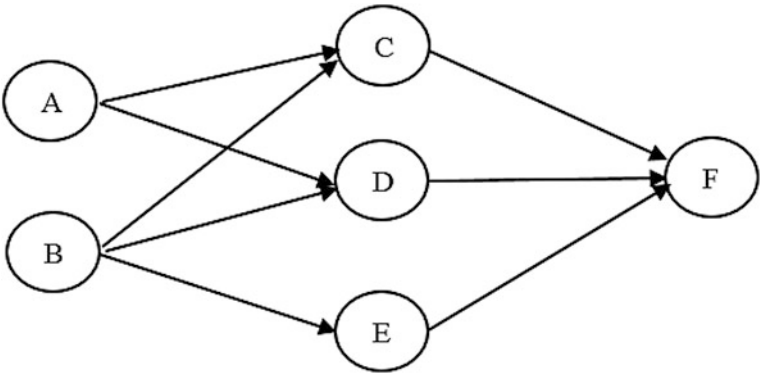

Fig. 2.2 Example of one way directed graph

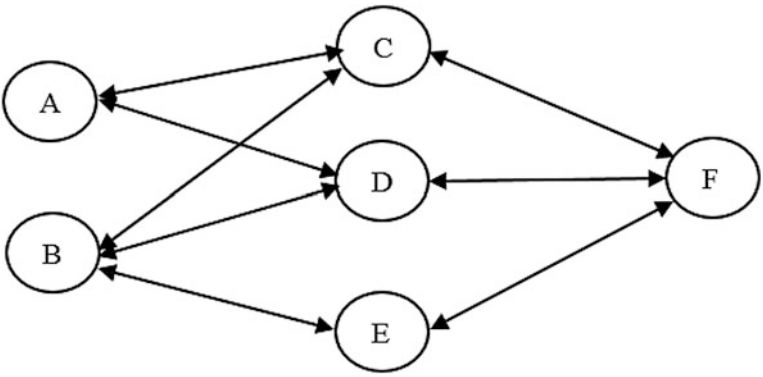

Fig. 2.3 Example of two way directed graph

2.2 Probability Theory

Probability theory is another branch of mathematics, which is a concept involved in all type of activities [4]. Probability is seen as a measure of uncertainty for a particular event. In general, there are two extreme cases. The first one is that if an event A is exact, then the probability of the event, denoted by P (A), is equal to 1. The other case is that if the event is impossible, then the corresponding probability would be equal to 0. In reality, most events have a random behavior and their corresponding probabilities would be ranged between 0 and 1. These events typically include independent events and mutually exclusive events.

Independent events generally mean that for two or more events the occurrence of one does not affect that of the other(s). However, the events will be mutually exclusive if the occurrence of one event results in the non-occurrence of the other (s). In addition, there are also some events that are neither independent nor mutually exclusive. In other words, the occurrence of one event may result in the occurrence of the other(s) with a probability. The corresponding probability is referred to as conditional probability, which is denoted by P(A|B). The P(A|B) is pronounced as 'the probability of A given B as a condition'. According to Bayes theorem [5], P(A) is seen as a prior probability, which indicates the pre-degree of certainty for event A, and P(A|B) as a posterior probability, which indicates the post-degree of certainty for event A after taking into consideration event B. In this book, the concept of probability theory introduced above is related to the essence of the methods for rule generation introduced in Chap. 3. In addition, the concept is also related to an information theoretic measure called J-measure, which is discussed in Sect. 2.6.

Probability theory is typically jointly used with statistics. For example, it can well contribute to the theory of distribution [4] with respect to probability distribution. As mentioned in [4], a probability distribution is often transformed from frequency distribution. When different events have the same probability, the probability distribution is in the case of normal distribution. In the context of statistics, normal distribution occurs while all possible outcomes have the same frequency resulting from a sampling based investigation. Probability distributions also help predict the expected outcome out of all possible outcomes in a random event. This could be achieved by weighted majority voting, while the random event is discrete, or by weighted averaging, while the event is continuous. In the above context, probability is actually used as the weight and the expected outcome is referred to as mathematical expectation. In addition, the probability distribution also helps measure the approximate distance between expected outcome and actual outcome, while the distance among different outcomes is precise such as rating from 1 to 5. This could be achieved by calculating the variance or standard deviation to reflect the volatility with regard to the possible outcome. In this book, the probability distribution is related to a technique of information theory, which is known as entropy and used as a measure of uncertainty in classification. In addition, the concept on mathematical expectation is used to measure the expected accuracy

by random guess in classification and variance/standard deviation can be used to measure the randomness of an algorithm of ensemble learning.

2.3 If-then Rules

As mentioned in Sect. 1.1, rule based system typically consists of a set of if-then rules. Ross stated in [1] that there are many different ways for knowledge representation in the area of artificial intelligence but the most popular one would perhaps be in the form of if-then rules denoted by the expression: IF cause (antecedent) THEN effect (consequent).

The expression above typically indicates an inference that if a condition (cause, antecedent) is known then the outcome (effect, consequent) can be derived [1]. Gegov introduced in [6] that both the antecedent and the consequent of a rule could be made up of multiple terms (inputs/outputs). In this context, an antecedent with multiple inputs that are linked by 'and' connectives is called a conjunctive antecedent, whereas the inputs that are linked by 'or' connectives would make up a disjunctive antecedent. The same concept is also applied to rule consequent. In addition, it is also introduced in [6] that rules may be conjunctive, if all of the rules are connected by logical conjunction, or disjunctive, if the rules are connected by logical disjunction. On the other hand, a rule may be inconsistent, which indicates that the antecedent of a rule may be mapped to different consequents. In this case, the rule could be expressed with a conjunctive antecedent and a disjunctive consequent.

In this book, if-then rules are used to make prediction in classification tasks. In this context, each of the rules is referred to as a classification rule, which can have multiple inputs, but only a single output. In a classification rule, the consequent with a single output represents the class predicted and the antecedent with a single/multiple input(s) represents the adequate condition to have this class predicted. A rule set that is used to predict classes consists of disjunctive rules which may be overlapped. This means that different rules may have the same instances covered. However, if the overlapped rules have different consequents (classification), it would raise a problem referred to as conflict of classification. In this case, conflict resolution is required to solve the problem according to some criteria such as weighted voting or fuzzy inference [1]. When a rule is inconsistent, it would result in uncertainty in classification. This is because the prediction of class becomes non-deterministic when this problem arises. More details about conflict resolution and dealing with inconsistent rules are introduced in Chap. 3.

Another concept relating to if-then rules is known as a rule base. In general, a rule base consists of a number of rules which have common input and output variables. For example, a rule base has two inputs: x_1 and x_2 and one output y as illustrated by Fig. 2.4.

If x_1, x_2 and y all belong to {0, 1}, the rule base can have up to four rules as listed below:

Fig. 2.4 Rule base with inputs x_1 and x_2 and output y

If $x_1 = 0$ and $x_2 = 0$ then $y \in \{0, 1\}$
If $x_1 = 0$ and $x_2 = 1$ then $y \in \{0, 1\}$
If $x_1 = 1$ and $x_2 = 0$ then $y \in \{0, 1\}$
If $x_1 = 1$ and $x_2 = 1$ then $y \in \{0, 1\}$

In practice, rule bases can be used to effectively and efficiently manage rules with respects to their storage and retrieval. For example, if a particular rule is searched for, it could be efficiently retrieved by locating at the rule base in which the rule is found. This is a significant difference to rule set for retrieval purpose. As mentioned earlier in this section, set is used to store a collection of elements which are not ordered nor grouped properly. From this point of view, it is not efficient to look for a particular rule in a rule set. The only way to deal with that is to linearly go through the rules one by one in the rule set until the target rule is found. In the worst case, it may be required to go through the whole set due to that the target rule is restored as the last element of the rule set. Therefore, the use of rule base would improve the efficiency in predicting classes on unseen instances in testing stage. More details about the use of rule bases are introduced in Chap. 8.

2.4 Algorithms

Aho et al. defined in [3] that "algorithm is a finite sequence of instructions, each of which has a clear meaning and can be performed with a finite amount of effort in a finite length of time". In general, an algorithm acts as a step by step procedure for problem solving. An algorithm may have no inputs but must have at least one output with regard to solving a particular problem. In practice, a problem can usually be solved by more than one algorithm. In this sense, it is necessary to make comparison between algorithms to find the one which is more suitable to a particular problem domain. An algorithm could be evaluated against the following aspects:

Accuracy, which refers to the correctness in terms of correlation between inputs and outputs.
Efficiency, which refers to the computational cost required.
Robustness, which refers to the tolerance to incorrect inputs.
Readability, which refers to the interpretability to people.

Accuracy would usually be the most important factor in determining whether an algorithm is chosen to solve a particular problem. It can be measured by providing the inputs and checking the outputs.

Efficiency is another important factor to measure if the algorithm is feasible in practice. This is because if an algorithm is computationally expensive then the implementation of the algorithm may be crashed on a hardware device. Efficiency of an algorithm can usually be measured by checking the time complexity of the algorithm in theoretical analysis. In practice, it is usually measured by checking the actual runtime on a machine.

Robustness can usually be measured by providing a number of incorrect inputs and checking to what extent the accuracy with regard to outputs is affected.

Readability is also important especially when an algorithm is theoretically analyzed by experts or read by practitioners for application purpose. This problem can usually be solved by choosing a suitable representation for the algorithm to make it easier to read. Some existing representations include flow chart, UML activity diagram, pseudo code, text and programming language.

This book addresses these four aspects in Chaps. 3, 4, 5, 6 and 7 in the way of theoretical analysis, as well as algorithm representation with regard to algorithm analysis.

2.5 Logic

Ross stated in [1] that logic is a small part of the capability of human reasoning, which is used to assist people in making decisions or judgments. Section 2.1 introduced mathematical logic which is also referred to as Boolean logic in computer science. As mentioned in Sect. 2.1, in the context of Boolean logic, each variable is only assigned a binary truth value: 0 (false) or 1 (true). It indicates that reasoning and judgment are made under certainty resulting in deterministic outcomes. From this point of view, this type of logic is also referred to as deterministic logic. However, in reality, people usually can only make decisions, and apply judgment and reasoning under uncertainty. Therefore, the other two types of logic, namely probabilistic logic and fuzzy logic, are used more popularly, both of which can be seen as an extension of deterministic logic. The main difference is that the truth value is not binary but continuous between 0 and 1. The truth value implies a probability of truth between true and false in probabilistic logic and a degree of that in fuzzy logic. The rest of the subsection introduces the essence of the three types of logic and the difference between them as well as how they are linked to the concept of rule based systems.

Deterministic logic deals with any events under certainty. For example, when applying deterministic logic for the outcome of an exam, it could be thought that a student will exactly pass or fail a unit. In this context, it means the event is certain to happen.

Probabilistic logic deals with any events under probabilistic uncertainty. For the same example about exams, it could be thought that a student has an 80 % chance to pass, i.e. 20 % chance to fail, for a unit. In this context, it means the event is highly probable to happen.

Fuzzy logic deals with any events under non-probabilistic uncertainty. For the same example about exams, it could be thought that a student has 80 % factors of passing, i.e. 20 % factors of failing, for a unit with regard to all factors in relation to the exam. In this context, it means the event is highly likely to happen.

A scenario is used to illustrate the above description as follows: students need to attempt the questions on four topics in a Math test. They can pass if and only if they pass all of the four topics. For each of the topics, they have to get all answers correct to pass. The exam questions do not cover all aspects that students are taught, but should not be outside the domain nor be known to students. Table 2.5 reflects the depth of understanding of a student in each of the topics.

In this scenario, deterministic logic is not applicable because it is never deterministic with regard to the outcome of the test. In other words, deterministic logic is not applicable in this situation to infer the outcome (pass/fail).

In probabilistic logic, the depth of understanding is supposed to be the probability of the student passing. This is because of the assumption that the student would exactly gain full marks for which questions the student is able to work out. Therefore, the probability of passing would be: $p = 0.8 \times 0.6 \times 0.7 \times 0.2 = 0.0672$.

In fuzzy logic, the depth of understanding is supposed to be the weight of the factors for passing. For example, for topic 1, the student has 80 % factors for passing but it does not imply that the student would have 80 % chance to pass. This is because in reality the student may feel unwell mentally, physically and psychologically. All of these issues may make it possible that the student will make mistakes as a result of that the student may fail to gain marks for which questions that normally he/she would be able to work out. The fuzzy truth value of passing is $0.2 = \min(0.8, 0.6, 0.7, 0.2)$. In this context, the most likely outcome for failing would be that the student only fails one topic resulting in a failure of Math. The topic 4 would be obviously the one which is most likely to fail with the fuzzy truth value 0.8. In all other cases, the fuzzy truth value would be less than 0.8. Therefore, the fuzzy truth value for passing is $0.2 = 1 - 0.8$.

In the context of set theory, deterministic logic implies that a crisp set that has all its elements fully belong to it. In other word, each element has a full membership to the set. Probabilistic logic implies that an element may be randomly allocated to one of a finite number of sets with normal distribution of probability. Once the element has been allocated to a particular set, then it has a full membership to the set. In other words, the element is eventually allocated to one set only. Fuzzy logic implies that a set is referred to as fuzzy set because each element may not have a full

Table 2.5 Depth of understanding for each topic

Topic 1	Topic 2	Topic 3	Topic 4
80 (%)	60 (%)	70 (%)	20 (%)

membership to the set. In other words, the element belongs to the fuzzy set to a certain degree.

In the context of rule base systems, a deterministic rule based system would have a rule either fire or not. If it fires, the consequence would be deterministic. A probabilistic rule based system would have a firing probability for a rule. The consequence would be probabilistic depending on posterior probability of it given specific antecedents. A fuzzy rule based system would have a firing strength for a rule. The consequence would be weighted depending on the fuzzy truth value of the most likely outcome. In addition, fuzzy rule based systems deal with continuous attributes by mapping the values to a number of linguistic terms according to the fuzzy membership functions defined. More details about the concepts on rule based systems outlined above are introduced in Chap. 5.

2.6 Statistical Measures

In this book, some statistical measures are used as heuristics for development of rule learning algorithms and evaluation of rule quality. This subsection introduces some of these measures, namely entropy, J-measure, confidence, lift and leverage.

Entropy is introduced by Shannon in [7], which is an information theoretic measure of uncertainty. Entropy E can be calculated as illustrated in Eq. (2.1):

$$E = -\sum_{i=0}^{n} p_i \cdot \log_2 p_i \tag{2.1}$$

where p is read as probability that an event occurs and i is the index of the corresponding event.

J-measure is introduced by Smyth and Goodman in [8], which is an information theoretic measure of average information content of a single rule. J-measure is essentially the product of two terms as illustrated in Eq. (2.2):

$$J(Y, X = x) = P(x) \cdot j(Y, X = x) \tag{2.2}$$

where the first term $P(x)$ is read as the probability that the rule antecedent (left hand side) occurs and considered as a measure of simplicity [8]. In addition, the second term is read as j-measure, which is first introduced in [9] but later modified in [8] and considered as a measure of goodness of fit of a single rule [8]. The j-measure is calculated as illustrated in Eq. (2.3):

$$j(Y, X = x) = P(y|x) \cdot \left(\frac{P(y|x)}{P(y)}\right) + \left(1 - P(y|x) \cdot \left(\frac{1 - P(y|x)}{1 - P(y)}\right)\right) \tag{2.3}$$

where $P(y)$ is read as prior probability that the rule consequent (right hand side) occurs and $P(y|x)$ is read as posterior probability that the rule consequent occurs given the rule antecedent as the condition.

In addition, j-measure has an upper bound referred to as jmax as indicated in [8] and illustrated in Eq. (2.4):

$$j(Y, X = x) \leq \max(P(y|x) \cdot \left(\frac{1}{P(y)}\right), \left(1 - P(y|x) \cdot \left(\frac{1}{1 - P(y)}\right)\right) \tag{2.4}$$

However, if it is unknown to which class the rule is assigned as its consequent, then the j-measure needs to be calculated by taking into account all possible classes as illustrated in Eq. (2.5):

$$j(Y, X = x) = \sum_{i=0}^{n} P(y_i|x) \cdot \left(\frac{P(y_i|x)}{P(y_i)}\right) \tag{2.5}$$

In this case, the corresponding jmax is calculated in the way illustrated in Eq. (2.6):

$$j(Y, X = x) \leq \max_i \left(P(y_i|x) \cdot \left(\frac{1}{P(y_i)}\right)\right) \tag{2.6}$$

Confidence is introduced in [10], which is considered as predictive accuracy of a single rule, i.e. to what extent the rule consequent is accurate while the rule antecedent is met. The confidence is calculated as illustrated in Eq. (2.7):

$$Conf = \frac{P(x, y)}{P(x)} \tag{2.7}$$

where $P(x, y)$ is read as the joint probability that the antecedent and consequent of a rule both occur and $P(x)$ is read as prior probability as same as used in J-measure above.

Lift is introduced in [11], which measures to what extent the actual frequency of joint occurrence for the two events X and Y is higher than expected if X and Y are statistically independent [12]. The lift is calculated as illustrated in Eq. (2.8):

$$Lift = \frac{P(x, y)}{P(x) \cdot P(y)} \tag{2.8}$$

where $P(x, y)$ is read as the joint probability of x and y as same as mentioned above and $P(x)$ and $P(y)$ are read as the coverage of rule antecedent and consequent respectively.

Leverage is introduced in [13], which measures the difference between the actual joint probability of x and y and the expected one [12]. The leverage is calculated as illustrated in Eq. (2.9):

$$Leverage = P(x, y) - P(x) \cdot P(y) \tag{2.9}$$

where $P(x, y)$, $P(x)$ and $P(y)$ are read as same as in Eq. (2.9) above.

A more detailed overview of these statistical measures can be found in [14, 15]. In this book, entropy is used as a heuristic for rule generation and J-measure is used for both rule simplification and evaluation. In addition, confidence, lift and leverage are all used for evaluation of rule quality. More details on this will be given in Chaps. 3, 4, 5 and 6.

2.7 Single Rule Based Classification Systems

As mentioned in Chap. 1, single rule based systems mean that a particular rule based system consists of one rule set only. If such rule based systems are used for classification, they can be referred to as single rule based classification systems.

In machine learning context, single rule based classification systems can be constructed by using standard rule learning algorithms such as ID3, C4.5 and C5.0. This kind of systems can also be constructed using ensemble rule based learning approaches.

In both ways mentioned above, rule based classification systems can be constructed by adopting a unified framework that was recently developed in [16]. This framework consists of rule generation, rule simplification and rule representation.

Rule generation means to generate a set of rules by using one or more rule learning algorithm(s). The generated rule set is eventually used as a rule based system for prediction making. However, as mentioned in Chap. 1, most of rule learning algorithms suffer from overfitting of training data, which means the generated rule set may perform a high level of accuracy on training instances but a low level of accuracy on testing instances. In this case, the set of rules generated need to be simplified by means of rule simplification and using pruning algorithms, which generally tends to reduce the consistency but improve the accuracy of each single rule. As introduced in Chap. 1, rule generation algorithms can be divided into two categories: divide and conquer and separate and conquer. In addition, pruning algorithms can be subdivided into two categories: pre-pruning and post-pruning. In this context, if the divide and conquer approach is adopted, rule simplification is taken to stop the growth of a branch in a decision tree if pre-pruning is adopted for the simplification of rules. Otherwise, the rule simplification would be taken to simplify each branch after a complete decision tree has been generated. On the other hand, if the separate and conquer approach is adopted, rule simplification is taken to stop the specialization of a single rule if pre-pruning is adopted for the simplification. Otherwise, the rule simplification would be taken to post-prune each single rule after the completion of its generation.

Rule representation means to represent a set of rules in a particular structure such as decision trees, linear lists and rule based networks. In general, appropriate

representation of rules enables the improvement of both interpretability and computational efficiency.

In terms of interpretability, a rule based system is required to make knowledge extracted from the system easier for people to read and understand. In other words, it facilitates the communication of the knowledge extracted from the system. On the other hand, the rules should allow to interpret knowledge in a great depth, in an explicit way. For example, when a system provides an output based on a given input, the reason why this output is derived should be explained in a straightforward way.

In terms of computational efficiency, a rule based system is required to make a quick decision in practice due to time critical aspects. In particular, rule representation is just like data structures which are used to manage data in different ways. In software engineering, different data structures usually lead to different levels of computational efficiency in some operations relating to data management such as insertion, update, deletion and search. As mentioned in Chap. 1, it is required to find the first firing rule as quickly as possible in order to make a quick prediction. Therefore, this could be seen as a search problem. As mentioned above, different data structures may provide different levels of search efficiency. For example, a collection of items stored in a linear list can only be searched linearly if these items are not given indexes. However, if the same collection of items is stored in a tree, then it is achievable to have a divide and conquer search. The former way of search would be in linear time whereas the latter way in logarithmic time. In this sense, efficiency in search of firing rules would also be affected by the structure of the rule set. It is also defined in [17] that one of the biases for rule based systems is 'search bias', which refers to the strategy used for the hypothesis search. In general, what is expected is to make it unnecessary to examine a whole rule set, but as few rule terms as possible.

Overall, the unified framework for construction of single rule based classification systems consists of three operations namely, rule generation, rule simplification and rule representation. In practice, the first two operations may be executed in parallel or sequentially depending on the approaches adopted. Rule representation is usually executed after the finalization of a rule set used as a rule based system. More details about these operations are presented in Chaps. 3, 4 and 5.

2.8 Ensemble Rule Based Classification Systems

As mentioned in Chap. 1, ensemble rule based classification systems mean that a particular rule based system consists of multiple rule sets, each of which could be seen as a single rule based systems. This defines a novel way of understanding ensemble rule based systems in the context of system theory [18]. In particular, each of the single rule based systems could be seen as a subsystem of the ensemble rule based system by means of a system of systems.

Ensemble rule based classification systems could be constructed by using ensemble learning approaches. Each of the single rule based classification systems can be constructed still based on the unified framework introduced in Sect. 2.1. In particular, for rule learning algorithms, ensemble learning can be adopted in the way that a base algorithm is used to learn from a number of samples, each of which results from the original training data through random sampling with replacement. In this case, there are n rule sets generated while n is the number of samples. In the testing stage, each of the n rule sets make an independent prediction on an unseen instance and their predictions are then combined to make the final prediction. The n rule sets mentioned above make up an ensemble rule based system for prediction purpose as mentioned in the literature [19]. A typical example of such systems is Random Forest which consists of a number of decision trees [20] and is usually helpful for decision tree learning algorithms to generate more accurate rule sets [21]. On the other hand, in order to construct ensemble rule based classification systems, ensemble learning can also be adopted in another way that multiple rule learning algorithms work together so that each of the algorithms generates a single rule set on the same training data. These generated rule sets, each of which is used as a single rule based classification systems, are combined to make up an ensemble rule based classification system.

The two ways to construct ensemble rule based systems mentioned above follow parallel learning approaches as introduced in Chap. 1. Ensemble rule based systems can also be constructed following sequential learning approaches. In particular, the same rule learning algorithm is applied to different versions of training data on an iterative basis. In other words, at each iteration, the chosen algorithm is used to generate a single rule set on the training data. The rule set is then evaluated on its quality by using validation data and each of the training instances is weighted to a certain degree based on its contribution to generating the rule set. The updated version of the training data is used at the next iteration. As the end, there is a number of rule sets generated, each of which is used as a single rule based classification system. These single systems make up the ensemble rule based classification systems.

Ensemble rule based systems can be advanced via different computing environments such as parallel, distributed and mobile computing environments.

Parallel computing can significantly improve the computational efficiency for ensemble learning tasks. In particular, as mentioned above, parallel learning can be achieved through the way that a number of samples are drawn on the basis of the training data, each of the sample is used to build a single rule based system by using the same rule learning algorithm. In this context, each of the drawn samples can be loaded into a core of a parallel computer and the same rule learning algorithm is used to build a single rule based systems on each core on the basis of the training sample loaded into the core. This is a popular approach known as parallel data mining [22, 23].

Distributed computing can make ensemble rule based systems more powerful in practical applications. For example, for some large companies or organizations, the web architecture is usually developed in the form of distributed database systems,

which means that data relating to the information of the companies or organizations is stored into their databases distributed in different sites. In this context, distributed data mining is highly required to process the data. In addition, ensemble rule based systems also motivate collaborations between companies through collaborations involved in ensemble learning tasks. In particular, the companies that collaborate with each other can share access to their databases. Each of the databases can provide a training sample for the employed rule learning algorithm to build a single rule based system. Each of the companies can have the newly unseen instances predicted using the ensemble rule based system that consists of a number of single rule based systems, each of which is from a particular database for one of the companies. Some popular distributed data mining techniques can be found in [24, 25]. Similar benefits also apply to mobile data mining such as Pocket Data Mining [26, 27].

More details about these ensemble learning approaches for construction of rule based systems is given in Chap. 6 in more depth.

References

1. Ross, T.J.: Fuzzy logic with engineering applications, 2nd edn. Wiley, West Sussex (2004)
2. Schneider, S.: The B-Method: an introduction. Palgrave Macmillian, Basingstoke, New York (2001)
3. Aho, A.V., Hopcroft, J.E., Ullman, J.D.: Data structures and algorithms. Addison-Wesley, Boston (1983)
4. Murdoch, J., Barnes, J.A.: Statistics: problems and solutions. The Macmillan Press Ltd, London and Basingstoke (1973)
5. Hazewinkel, M. (ed.): Bayes formula. Springer, Berlin (2001)
6. Gegov, A.: Advanced computation models for rule based networks, Portsmouth (2013)
7. Shannon, C.: A mathematical theory of communication. Bell Syst. Tech. J. **27**(3), 379–423 (1948)
8. Smyth, P., Rodney, G.M.: An information theoretic approach to rule induction from databases. IEEE Trans. Knowl. Data Eng. **4**(4), 301–316 (1992)
9. Blachman, N.M.: The amount of information that y gives about X. IEEE Transcations. Inf. Theory. **14**(1), 27–31 (1968)
10. Agrawal, R., Imilielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proceedings of ACM SIGMOD International Conference on Management of Data, Washington D.C (1993)
11. Brin, S., Motwani, R., Ullman, J.D., Tsur, S.: Dynamic itemset counting and implication rules for market basket data. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, Tucson, Arizona (1997)
12. Hahsler, M.: A proabalistic comparison of commonly used interest measures for association rules. Available: http://michael.hahsler.net/research/association_rules/measures.html (2015) (Online)
13. Piatetsky-Shapiro, G.: Discovery, analysis, and presentation of strong rules. In: Piatetsky-Shapiro, G., Frawley, W.J. (eds.) Knowledge Discovery in Databases. MA, AAAI/MIT Press, Cambridge (1991)
14. Tan, P.-N., Kumar, V., Srivastava, J.: Selecting the right objective measure for association analysis. Inf. Syst. **29**(4), 293–313 (2004)

15. Geng, L., Hamilton, H.J.: Interestingness measures for data mining: a survey. ACM Computing Surveys, vol. 38, no. 3 (2006)
16. Liu, H., Gegov, A., Stahl, F.: Unified framework for construction of rule based classification systems. In: Pedrycz, W., Chen, S. (eds.) Inforamtion Granularity, Big Data and Computational Intelligence, vol. 8, pp. 209–230. Springer, Berlin (2015)
17. Furnkranz, J.: Separate-and-conquer rule learning. Artif. Intell. Rev. **13**, 3–54 (1999)
18. Stichweh, R.: Systems theory. In: Badie, B.E.A. (ed.) International Encyclopaedia of Political Science, New York, Sage (2011)
19. Liu, H., Gegov, A.: Collaborative decision making by ensemble rule based classification systems. In: Pedrycz, W., Chen, S. (eds.) Granular Computing and Decision-Making: Interactive and Iterative Approaches, vol. 10, pp. 245–264. Springer, Berlin (2015)
20. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
21. Kononenko, I., Kukar, M.: Machine learning and data mining: introduction to principles and algorithms. Horwood Publishing Limited, Chichester, West Sussex (2007)
22. Li, J., Liu, Y., Liao, W.-K., Choudhary, A.: Parallel data mining algorithms for association rules and clustering. CRC Press, Boca Raton (2006)
23. Parthasarathy, S., Zaki, M.J., Ogihara, M., Li, W.: Parallel data mining for association rules on shared-memory systems. Knowl. Inf. Syst. **3**, 1–29 (2001)
24. Giannella, C., Bhargava, R., Kargupta, H.: Multi-agent systems and distributed data mining. In: Cooperative Information Agents VIII, Berlin (2004)
25. Datta, S., Bhaduri, K., Giannella, C., Wolff, R., Kargupta, H.: Distributed data mining in peer-to-peer networks. IEEE. Internet. Comput. **10**(4), 18–26 (2006)
26. Gaber, M.M., Stahl, F., Gomes, J.B.: Pocket data mining: big data on small devices, vol. 2. Springer, Switzerland (2014)
27. Gaber, M.: Pocket data mining: the next generation in predictive analytics. In: Predictive Analytics Innovation Summit, London (2012)

Chapter 3
Generation of Classification Rules

As mentioned in Chap. 1, rule generation can be done through the use of the two approaches: divide and conquer and separate and conquer. This chapter describes the two approaches of rule generation. In particular, the existing rule learning algorithms, namely ID3, Prism and Information Entropy Based Rule Generation (IEBRG), are illustrated in detail. These algorithms are also discussed comparatively with respects to their advantages and disadvantages.

3.1 Divide and Conquer

As mentioned in Chap. 1, divide and conquer approach is also known as Top Down Induction of Decision Trees (TDIDT) due to the fact that classification rules generated through the use of this approach are in the form of decision trees. The basic procedures of TDIDT are illustrated in Fig. 3.1.

For the TDIDT approach, the most important procedure is the attribute selection for partition of a training subset on a node of a decision tree. In particular, the attribution selection can be done through either random selection or employment of statistical measures such as information gain, gain ratio and Gini index [2].

A popular method that follows the divide and conquer approach is ID3, which is based on information gain for attribute selection and was developed by Quinlan in [3]. A successor of ID3, which is known as C4.5 and extended to involve direct processing of continuous attributes, was later presented by Quinlan. Another method of TDIDT is known as CART, which stands for Classification and Regression Trees and aims to generate binary decision trees [4].

3.2 Separate and Conquer

As mentioned in Chap. 1, the separate and conquer approach is also known as covering approach due to the fact that this approach involves sequential generation of if-then rules. In particular, this approach aims to generate a rule that covers the

H. Liu et al., *Rule Based Systems for Big Data*,
Studies in Big Data 13, DOI 10.1007/978-3-319-23696-4_3

```
Input: A set of training instances, attribute A_i, where i is the index of the attribute A, value V_j, where j is the index of the value V
Output: A decision tree.

  if the stopping criterion is satisfied then
    create a leaf that corresponds to all remaining training instances
  else
      choose the best (according to some heuristics) attribute A_i
      label the current node with A_i
      for each value V_j of the attribute A_i do
       label an outgoing edge with value V_j
       recursively build a subtree by using a corresponding subset of training instances
      end for
  end if
```

Fig. 3.1 Decision tree learning algorithm [1]

instances that belong to the same class and then starts to generate the next rule on the basis of the rest of training instances that are not covered by the previously generated rules. The basic procedures of the separate and conquer approach are illustrated in Fig. 3.2.

As introduced in [1], the majority rule is included as the last rule in a rule set, which is supposed to cover all the remaining instances that are not covered by any other generated rules. This rule is also referred to as default rule, which is used to assign a default class (usually majority class) to any unseen instances that cannot be classified by using any other generated rules.

On the other hand, in contrast to the divide and conquer approach, the most important procedure for the separate and conquer approach is at the selection of attribute-value pair. More details on this is introduced later in this section using specific methods that follow this rule learning approach.

```
Input: A set of training instances
Output: An ordered set of rules

  while training set is not empty do
    generate a single rule from the training set
    delete all instances covered by this rule
    if the generated rule is not good then
      generate the majority rule and empty the training set
    end if
  end while
```

Fig. 3.2 Rule covering approach [1]

A popular method that follows the separate and conquer approach is Prism, which was developed by Cendrowska in [5]. The basic procedure of the method is illustrated in Fig. 3.3.

The Prism algorithm can be seen as an example of the backward rule generation approach that was introduced in Chap. 2. This is because the algorithm aims to first give a target class as the consequent of the rule, and then to search for causes as the condition that can derive this target class until the adequacy condition is found. In other words, the Prism algorithm involves the specialization of rule antecedent in order to have all the instances covered by this rule belong to the target class assigned as the consequent of this rule.

As can be seen from Fig. 3.3, the Prism algorithm employs probability as the heuristic for the selection of attribute-value pairs to specialize the rule antecedent. In particular, the rule generation is normally completed when the posterior probability of the target class given all the chosen attribute-value pairs is equal to 1.

On the other hand, rule learning by Prism aims to extract knowledge on a single classification. In other words, the aim is to explore under what conditions an instance can be classified to a particular category. In addition, the knowledge discovery for one classification is totally independent of the others. This is because

Input: a training set T, a subset $T' \subseteq T$, an instance $t \in T$, dimensionality d, an attribute a_x (x is the index of a), class C_i (i is the index of C), number of classes n
Output: a rule set RS, a result set of instances T'' covered by a rule $R \in RS$

Initialize:
$T' = T$, $T'' = \emptyset$, $i=0$;
for $i<n$ **do**
  do generate rules for class C_i
    while $\exists$ t: t $\in T' \wedge$ t $\notin C_i$ **do**
      $x=0$;
      while $x < d$ **do**
        for each value v of a_x **do**
          Calculate P(C_i| $a_x = v$);
        end for
        x++;
      end while
      assign $a_x = v$ to R as a rule term, while P(C_i| $a_x = v$) is max;
      $\forall$t: $T'' \cap \{t\}$, if t comprise $a_x = v$;
    end while
    $RS = RS \cap \{R\}$;
    $T' = T' - T''$;
  while $\forall$t: t $\in T' \wedge$ t $\notin C_i$
  i++;
end for

Fig. 3.3 Prism algorithm

the Prism algorithm involves continuous generation of rules, all of which have the same target class assigned as the rule consequent, until all instances that belong to this class have been covered by at least one of the generated rules. Then the same process is repeated for the next class to generate any possible rules on the basis of the original training set.

Another method that follows separate and conquer approach is referred to as IEBRG, which stands for Information Entropy Based Rule Generation and has been recently developed in [6]. The basic procedure of the method is illustrated in Fig. 3.4.

The IEBRG algorithm can be seen as an example of the forward rule generation approach that was introduced in Chap. 2. This is because the IEBRG algorithm aims to specialize the antecedent of a rule until all instances covered by the rule belong to the same class. In other words, the algorithm involves specialization of rule antecedent in order to have the rule cover all instances that belong to the same class no matter which particular class it is.

As can be seen from Fig. 3.4, the IEBRG algorithm employs entropy as the heuristic for the selection of attribute-value pairs to specialize the rule antecedent. In particular, the rule generation is normally completed when the conditional entropy given all the chosen attribute-value pairs is equal to 0.

Input: a training set T, a subset $T' \subseteq T$, an instance $t \in T$, dimensionality d, an attribute a_x (x is the index of a), entropy E, number of classes n, class C_i (I is the index of C)
Output: a rule set RS, a result set of instances T'' covered by a rule $R \in RS$

Initialize:
$T' = T$, $T'' = \emptyset$, $E = -\sum P(C_i) \log_2 P(C_i)$, $i = 0,1,2....n-1$
While $T' \neq \emptyset$ **Do**
  While $E \neq 0$ **Do**
    $x = 0$;
    While $x < d$ **Do**
      For each value v of a_x **Do**
        Calculate $E(C|\ a_x = v)$;
      End For
      x++;
    End While
    assign $a_x = v$ to R as a rule term, while $E(C|\ a_x = v)$ is min;
    $\forall t$: $T'' \cap \{t\}$, if t comprise $a_x = v$;
  End While
  $RS = RS \cap \{R\}$;
  $T' = T' - T''$;
  update E;
End While

Fig. 3.4 IEBRG algorithm

On the other hand, rule learning by IEBRG aims to reduce the uncertainty in discriminating different classes. In other words, the aim is to explore under what conditions an unseen instance can be classified under certainty. In addition, IEBRG can generate a set of ordered rules in contrast to Prism. This is because IEBRG algorithm generates each of the rules on the basis of gradually reduced training subset. In other words, the training subset is reduced per each rule generated due to the removal of the instances covered by previously generated rules and is never reset to its original size. As the first rule is generated on the basis of the whole training set, this rule is the most important. The second rule is generated on the basis of the reduced training subset that excludes the instances covered by the first rule. Therefore, the second rule is less important than the first one but more important than all of the others and so on.

3.3 Illustrative Example

This section illustrates ID3, Prism and IEBRG using a data set that is named contact-lenses [5] and retrieved from UCI repository [7]. The details of this data set are illustrated in Table 3.1.

Table 3.1 Contact lenses data

Age	Prescription	Astigmatic	Tear production rate	Class
Young	Myope	No	Reduced	No lenses
Young	Myope	No	Normal	Soft lenses
Young	Myope	Yes	Reduced	No lenses
Young	Myope	Yes	Normal	Hard lenses
Young	Hypermetrope	No	Reduced	No lenses
Young	Hypermetrope	No	Normal	Soft lenses
Young	Hypermetrope	Yes	Reduced	No lenses
Young	Hypermetrope	Yes	Normal	Hard lenses
Pre-presbyopic	Myope	No	Reduced	No lenses
Pre-presbyopic	Myope	No	Normal	Soft lenses
Pre-presbyopic	Myope	Yes	Reduced	No lenses
Pre-presbyopic	Myope	Yes	Normal	Hard lenses
Pre-presbyopic	Hypermetrope	No	Reduced	No lenses
Pre-presbyopic	Hypermetrope	No	Normal	Soft lenses
Pre-presbyopic	Hypermetrope	Yes	Reduced	No lenses
Pre-presbyopic	Hypermetrope	Yes	Normal	Hard lenses
Presbyopic	Myope	No	Reduced	No lenses
Presbyopic	Myope	No	Normal	Soft lenses
Presbyopic	Myope	Yes	Reduced	No lenses

(continued)

Table 3.1 (continued)

Age	Prescription	Astigmatic	Tear production rate	Class
Presbyopic	Myope	Yes	Normal	Hard lenses
Presbyopic	Hypermetrope	No	Reduced	No lenses
Presbyopic	Hypermetrope	No	Normal	Soft lenses
Presbyopic	Hypermetrope	Yes	Reduced	No lenses
Presbyopic	Hypermetrope	Yes	Normal	Hard lenses

As mentioned in Sect. 3.1, ID3 makes attribute selection based on entropy. In addition, Prism and IEBRG make selection of attribute-value pairs based on posterior probability and conditional entropy respectively. For each of them, it is necessary to create a frequency table for each attribute (Tables 3.2, 3.3, 3.4 and 3.5).

For ID3, the average entropy for each of the attributes is in the following:

$$\begin{aligned}
\mathrm{E}(\text{age}) &= 1/3 \times (-(1/2) \times \log_2(1/2) - (1/4) \times \log_2(1/4) - (1/4) \times \log_2(1/4)) \\
&\quad + 1/3 \times (-(1/2) \times \log_2(1/2) - (1/4) \times \log_2(1/2) - (1/4) \times \log_2(1/2)) \\
&\quad + 1/3 \times (-(1/2) \times \log_2(1/2) - (1/4) \times \log_2(1/2) - (1/4) \times \log_2(1/2)) = 3/2 \\
\mathrm{E}(\text{prescription}) &= 1/2 \times (-(1/2) \times \log_2(1/2) - (1/4) \times \log_2(1/4) - (1/4) \times \log_2(1/4)) \\
&\quad + 1/2 \times (-(1/2) \times \log_2(1/2) - (1/4) \times \log_2(1/4) - (1/4) \times \log_2(1/4)) = 3/2 \\
\mathrm{E}(\text{astigmatic}) &= 1/2 \times (-(1/2) \times \log_2(1/2) - (1/2) \times \log_2(1/2)) \\
&\quad + 1/2 \times (-(1/2) \times \log_2(1/2) - (1/2) \times \log_2(1/2)) = 1 \\
\mathrm{E}(\text{tear production rate}) &= 1/2 \times (-(1/2) \times \log_2(1/2) - (1/2) \times \log_2(1/2)) \\
&\quad + 1/2 \times (-1 \times \log_2(1)) = 1/2
\end{aligned}$$

As E (tear production rate) is the minimum, the data set illustrated in Table 3.1 is split on the attribute *tear production rate*. This results in two subsets as below.

It can be seen from Table 3.6 that all instances belong to the class *no lenses*, which indicates there is no uncertainty remaining in the subset. Therefore, it results in an incomplete decision tree as illustrated in Fig. 3.5.

Table 3.2 Frequency table for age

Class label	Age = young	Age = pre-presbyopic	Age = presbyopic
No lenses	4	4	4
Soft lenses	2	2	2
Hard lenses	2	2	2
Total	8	8	8

Table 3.3 Frequency table for spectacle prescription

Class label	Prescription = myope	Prescription = hypermetrope
No lenses	6	6
Soft lenses	3	3
Hard lenses	3	3
Total	12	12

Table 3.4 Frequency table for astigmatic

Class label	Astigmatic = yes	Astigmatic = no
No lenses	6	6
Soft lenses	6	0
Hard lenses	0	6
Total	12	12

Table 3.5 Frequency table for tear production rate

Class label	Tear production rate = normal	Tear production rate = reduced
No lenses	0	12
Soft lenses	6	0
Hard lenses	6	0
Total	12	12

Table 3.6 Subset 1 for contact lenses data

Age	Prescription	Astigmatic	Tear production rate	Class
Young	Myope	No	Reduced	No lenses
Young	Myope	Yes	Reduced	No lenses
Young	Hypermetrope	No	Reduced	No lenses
Young	Hypermetrope	Yes	Reduced	No lenses
Pre-presbyopic	Myope	No	Reduced	No lenses
Pre-presbyopic	Myope	Yes	Reduced	No lenses
Pre-presbyopic	Hypermetrope	No	Reduced	No lenses
Pre-presbyopic	Hypermetrope	Yes	Reduced	No lenses
Presbyopic	Myope	No	Reduced	No lenses
Presbyopic	Myope	Yes	Reduced	No lenses
Presbyopic	Hypermetrope	No	Reduced	No lenses
Presbyopic	Hypermetrope	Yes	Reduced	No lenses

The left branch comprising *tear production rate* = *reduced* is terminated by giving a leaf node labeled *no lenses*. The right branch comprising *tear production rate* = *normal* is still not terminated, which means it is required to select another attribute other than *tear production rate* to be split at the node which is labeled with a question mark. For this, it is needed to create a frequency table for each of the rest of the attributes namely age, prescription and astigmatic from the subset 2 for the data set illustrated in Table 3.7 as follows.

According to Table 3.8, 3.9, 3.10, the average entropy for each of the attributes is shown below.

Table 3.7 Subset 2 for contact lenses data

Age	Prescription	Astigmatic	Tear production rate	Class
Young	Myope	No	Normal	Soft lenses
Young	Myope	Yes	Normal	Hard lenses
Young	Hypermetrope	No	Normal	Soft lenses
Young	Hypermetrope	Yes	Normal	Hard lenses
Pre-presbyopic	Myope	No	Normal	Soft lenses
Pre-presbyopic	Myope	Yes	Normal	Hard lenses
Pre-presbyopic	Hypermetrope	No	Normal	Soft lenses
Pre-presbyopic	Hypermetrope	Yes	Normal	Hard lenses
Presbyopic	Myope	No	Normal	Soft lenses
Presbyopic	Myope	Yes	Normal	Hard lenses
Presbyopic	Hypermetrope	No	Normal	Soft lenses
Presbyopic	Hypermetrope	Yes	Normal	Hard lenses

Table 3.8 Frequency table for age at the second iteration

Class label	Age = young	Age = pre-presbyopic	Age = presbyopic
No lenses	0	0	0
Soft lenses	2	2	2
Hard lenses	2	2	2
Total	4	4	4

Table 3.9 Frequency table for spectacle prescription at the second iteration

Class label	Prescription = myope	Prescription = hypermetrope
No lenses	0	0
Soft lenses	3	3
Hard lenses	3	3
Total	6	6

Table 3.10 Frequency table for astigmatic at the second iteration

Class label	Astigmatic = yes	Astigmatic = no
No lenses	0	0
Soft lenses	0	6
Hard lenses	6	0
Total	6	6

$$\begin{aligned}
\mathrm{E}(\text{age}) &= 1/2 \times (-(1/2) \times \log_2(1/2) - (1/2) \times \log_2(1/2)) \\
&\quad + 1/2 \times (-(1/2) \times \log_2(1/2) - (1/2) \times \log_2(1/2)) = 1 \\
\mathrm{E}(\text{prescription}) &= 1/2 \times (-(1/2) \times \log_2(1/2) - (1/2) \times \log_2(1/2)) \\
&\quad + 1/2 \times (-(1/2) \times \log_2(1/2) - (1/2) \times \log_2(1/2)) = 1 \\
\mathrm{E}(\text{astigmatic}) &= 1/2 \times (-1 \times \log_2(1)) + 1/2 \times (-1 \times \log_2(1)) = 0
\end{aligned}$$

Therefore, the attribute *astigmatic* is selected to be split at the node that is labeled with a question mark in Fig. 3.5, which means that the subset 2 for the data set illustrated in Table 3.7 is split on the *astigmatic* attribute resulting in two further subsets (Tables 3.11 and 3.12).

It is clear that both subsets illustrated above have all instances belong to the same class and thus remains no uncertainty. The complete decision tree is generated as illustrated in Fig. 3.6.

For Prism, if the class *no lenses* is chosen as the target class, the first rule generated is represented in the form: *if tear production rate = reduced then class = no lenses*; this is because the posterior probability *P (tear production rate = reduced|class = no lenses)* = 1, which is the highest one and leaves no uncertainty. It can also be seen from Table 3.5 that there are 12 instances comprising the attribute-value pair *tear production rate = reduced*, all of which belong to the class *no lenses*. All the subsequent rules are generated in the same way as the first rule by appending rule terms on the left hand side by iteratively selecting the attribute-value pair with the highest posterior probability of the target class.

For IEBRG, the first rule generated is also the same as the one represented above:

if tear production rate = reduced then class = no lenses; this is because the conditional entropy E(*tear production rate = reduced)* = 0 is the minimum and indicates that there is no uncertainty any more for classifying instances covered by this rule. The same can also be seen from Table 3.5 as mentioned above. All the subsequent rules are generated in the same way as the first rule by appending rule terms on the left hand side of the rule by iteratively selecting the attribute-value pair with the lowest conditional entropy for discriminating different classes.

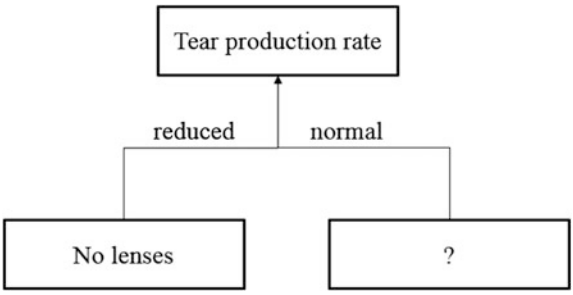

Fig. 3.5 Incomplete decision tree comprising attribute *Tear production rate*

Table 3.11 Subset 2.1 for contact lenses data

Age	Prescription	Astigmatic	Tear production rate	Class
Young	Myope	No	Normal	Soft lenses
Young	Hypermetrope	No	Normal	Soft lenses
Pre-presbyopic	Myope	No	Normal	Soft lenses
Pre-presbyopic	Hypermetrope	No	Normal	Soft lenses
Presbyopic	Myope	No	Normal	Soft lenses
Presbyopic	Hypermetrope	No	Normal	Soft lenses

Table 3.12 Subset 2.2 for contact lenses data

Age	Prescription	Astigmatic	Tear production rate	Class
Young	Myope	Yes	Normal	Hard lenses
Young	Hypermetrope	Yes	Normal	Hard lenses
Pre-presbyopic	Myope	Yes	Normal	Hard lenses
Pre-presbyopic	Hypermetrope	Yes	Normal	Hard lenses
Presbyopic	Myope	Yes	Normal	Hard lenses
Presbyopic	Hypermetrope	Yes	Normal	Hard lenses

3.4 Discussion

As summarized by Han and Kamber in [2], decision tree learning is so popular due to the following reasons:

Firstly, the generation of decision trees does not need any prior knowledge in a domain nor parameters setting. Therefore, decision tree learning is seen as an appropriate approach for knowledge discovery.

Secondly, the decision tree learning algorithms are able to effectively deal with training data in high dimensionality.

Thirdly, the decision tree representation is interpretable. In other words, knowledge extracted from a decision tree is easily communicated to people.

Fourthly, the training by the induction algorithm is not expensive and the prediction by a decision tree classifier is straightforward and efficient.

Fifthly, the decision tree learning algorithms can generate accurate classifiers in general.

Finally, the learning algorithm is not domain dependent but data dependent. In this context, the decision tree learning algorithm can be used broadly in many different application areas such as medicine, finance and biology. However, the performance in a particular task is highly dependent on the suitability to the data used.

Although the TDIDT has some advantages listed in [2] and described above, Cendrowska has criticized the principles of rule generation by the decision tree learning approach. In particular, Cendrowska pointed out in [5] that one of the disadvantages of decision tree learning approach is that there are some rules that

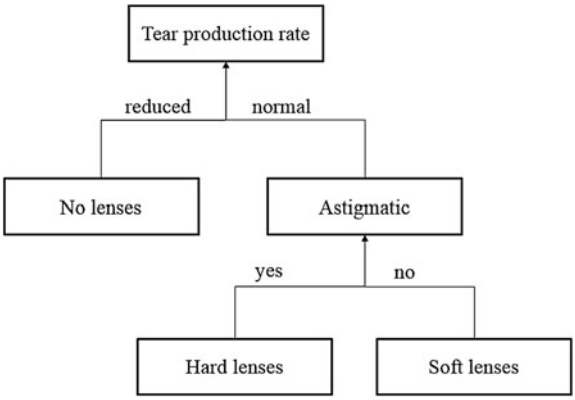

Fig. 3.6 Complete decision tree

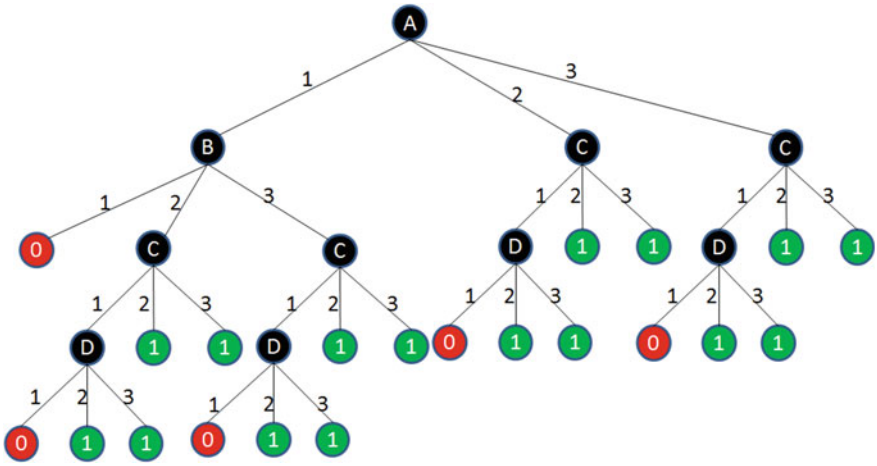

Fig. 3.7 Cendrowska's replicated subtree example [9]

cannot necessarily be represented in a tree structure. In this context, if the rules are forced to fit in a tree structure, the decision tree will encounter the replicated subtree problem as illustrated in Fig. 3.7. It can be seen from the Fig. 3.7 that the four sub-trees which have node C as their roots are identical.

In a medical diagnosis system, the replicated subtree problem may lead to unnecessary surgery [5]. There are some reasons identified in [5, 8] through analysis of ID3 as follows:

- The decision tree learning approach is attribute oriented.
- Each iteration in the generation process chooses the attribute on which to be split aiming at minimizing the average entropy, i.e. measuring the average

uncertainty. However, this doesn't necessarily mean that the uncertainty for each single rule is reduced.

- An attribute might be highly relevant to one particular classification and irrelevant to the others. Sometimes only one value of an attribute is relevant to making classifications.

Some comparisons and contrasts between ID3 and Prism are made and described in [8] as follows:

Firstly, both ID3 and Prism are greedy algorithms as they both involves greedy search for generation of rules.

Secondly, ID3 measures the reduction of uncertainty on average in making a classification on the basis of an attribute selected. In contrast, Prism measures the certainty factor in the determination of a particular classification on the basis of an attribute-value pair selected. Therefore, ID3 is attribute oriented whereas Prism is attribute-value oriented.

Thirdly, rules generated by ID3 may contain redundant attributes. In contrast, Prism usually generates more general and fewer rules than ID3.

On the basis of the above description, Prism has some obvious advantages in comparison with ID3 and other decision tree learning algorithms. However, Prism also has some disadvantages in terms of classification conflict, inconsistent rules and computational efficiency [6, 10–12].

Prism may generate a rule set which results in a classification conflict in predicting unseen instances. This can be illustrated by the example below:

What should the classification be for an instance with x = 1, y = 1 and z = 1? One rule gives *class a*, the other one gives *class b*. In the context of deterministic logic, the instance should only belong to one class. This problem needs to be solved by adopting conflict resolution strategies. A simple approach is to assign the unseen instance the class which is the consequent of the first firing rule in the rule set. However, the above strategy is effective in general if and only if the set of rules are ordered according to their importance. In other words, the first firing rule is usually considered most confident.

As mentioned in Sect. 3.2, Prism involves that rules on one classification are generated totally independent of that on the other classifications. In other words, the rules on different classifications are generated in parallel. Therefore, there is no way to rank the rules on different classifications by Prism and thus the conflict resolution strategy mentioned above is not applicable to the algorithm.

On the other hand, Prism is likely to encounter generation of inconsistent rules due to inappropriate selection of target class pre-assigned to the rule being generated. As introduced in Sect. 3.2, Prism involves continuous selection of the same class as the target class for rule generation until all possible rules have been generated. This could be seen as a bias originating from the Prism algorithm since it is highly possible at any iterations that one of the other classes is a more appropriate candidate for being the target class. If the selection of the target class is inappropriate, then it is likely to result in the generation of inconsistent rules. In other

words, the rule convers instances that belong to different classes, but there is no way to specialize the rule on its left hand side any more.

Besides, Prism is also computationally expensive. One of the reasons is due to the inappropriate selection of the target class. As mentioned above, if the selection is not appropriate, the rule generation cannot be completed until all attributes have been examined. As a result, the rule comprises the same number of rule terms as data dimensionality, i.e. the number of attributes. If the training data is large, it would result in the generation of complex rules, which usually requires high computational cost. Another reason is due to the nature of the learning strategy involved in Prism. In particular, as mentioned above, Prism involves the generation of rules on the basis of large training subsets at most iterations. This is because the reduced training subset needs to be restored to its initial version once all instances of a single class have been covered by the previously generated rules. In this context, most rules are generated on the basis of a large training set.

Due to the above limitations of Prism, IEBRG has been recently developed in [6] in order to complement Prism.

As mentioned in Sect. 3.2, IEBRG involves the generation of ordered rules. Therefore, the conflict resolution strategy is effectively applicable to the algorithm when conflicts of classification occur.

The empirical results reported in [6] indicate IEBRG is less likely to generate inconsistent rules. This is because the above algorithm pays special attention to minimizing the uncertainty for each rule being generated no matter what the target class is. This removes the bias on selection of target classes that happens from Prism.

On the other hand, the theoretical analysis made in [10] and the empirical results reported in [6] indicate that IEBRG is likely to generate a smaller number of more general rules than Prism. In addition, the empirical results also indicate that the IEBRG algorithm usually generates more accurate rule sets than the Prism.

Chapter 8 will introduce a case study on big data, which reports more detailed theoretical analysis and empirical study for the two rule generation methods mentioned above.

References

1. Kononenko, I., Kukar, M.: Machine Learning and Data Mining: Introduction to Principles and Algorithms. Horwood Publishing Limited, Chichester, West Sussex (2007)
2. Han, J., Kamber, M.: Data Mining: Concept and Techniques, 2nd edn. Morgan Kaufmann Publishers, San Francisco (2006)
3. Quinlan, R.: Induction of decision trees. Mach. Learn. **1**, 81–106 (1986)
4. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth International Group, Belmont (1984)
5. Cendrowska, J.: PRISM: an algorithm for inducing modular rules. Int. J. Man Mach. Stud. **27**, 349–370 (1987)

6. Liu, H., Gegov, A.: Induction of modular classification rules by information entropy based rule generation. In: Sgurev, V., Yager, R., Kacprzyk, J., Jotsov, V. (eds.) Innovative Issues in Intelligent Systems. Springer, Berlin (in press)
7. Lichman, M.: UCI machine learning repository, University Of California, School of Information and Computer Science. http://archive.ics.uci.edu/ml 2013. Accessed 12 May 2015 (online)
8. Deng, X.: A covering-based algorithm for classification: PRISM, SK (2012)
9. Liu, H., Gegov, A., Cocea, M.: Network based rule representation for knowledge discovery and predictive modelling. In: IEEE International Conference on Fuzzy Systems, Istanbul (2015)
10. Liu, H., Gegov, A., Stahl, F.: Unified framework for construction of rule based classification systems. In: Pedrycz, W., Chen, S. (eds.) Information Granularity, Big Data and Computational Intelligence, vol. 8, pp. 209–230. Springer, Berlin (2015)
11. Liu, H., Gegov, A.: Collaborative decision making by ensemble rule based classification systems. In: Pedrycz, W., Chen, S. (eds.) Granular Computing and Decision-Making: Interactive and Iterative Approaches, vol. 10, pp. 245–264. Springer, Berlin (2015)
12. Liu, H., Gegov, A., Stahl, F.: Categorization and construction of rule based systems. In: 15th International Conference on Engineering Applications of Neural Networks, Sofia (2014)

Chapter 4
Simplification of Classification Rules

As mentioned in Chap. 1, pruning methods are increasingly required for rule simplification due to the overfitting problem. This chapter introduces two approaches of rule simplification namely, pre-pruning and post-pruning. In particular, some existing rule pruning algorithms are described in detail. These algorithms are also discussed comparatively with respects to their advantages and disadvantages.

4.1 Pruning of Decision Trees

As mentioned in Chap. 2, decision trees can be simplified in two ways namely, pre-pruning and post-pruning. In the former way, the pruning action aims to stop a particular branch in a tree growing further. In the latter way, the pruning action aims to simplify each of the branches in a tree after the whole tree has been generated. In particular, the tree needs to be converted into a set of if-then rules before the pruning action is taken. The rest of this section introduces a particular type of tree pruning based on J-measure in the manner of both pre-pruning and post-pruning.

For pre-pruning of decision trees, it is unknown which class is eventually labelled on the leaf node of a particular branch. Therefore, the calculation for the value of J-measure needs to follow the Eqs. (2.3) and (2.6) shown in Sect. 2.6. The corresponding upper bound of J-measure, which is referred to as Jmax, is calculated following the Eqs. (2.3) and (2.7) shown in Sect. 2.6.

In the above context, J-measure based tree pre-pruning works in the procedures as illustrated in Fig. 4.1.

H. Liu et al., *Rule Based Systems for Big Data*,
Studies in Big Data 13, DOI 10.1007/978-3-319-23696-4_4

Input: a tree node *n*, J-measure of a node *J(n)*, Jmax of a node *Jmax(n)*, Maximum of J-measure *max(J),* class *C*
Output: a decision tree T
Initialize: *max(J)*= 0;
for each branch **do**
  calculate *J(n)* and *Jmax(n)*
  if *J(n)*> *max(J)* **then**
    max(J)= *J(n)*
  end if
  if stopping criteria is met **Then**
    stop this branch growing, label the current node *n* with the most common class *C*
  end if
end for

Fig. 4.1 J-measure based tree pre-pruning

The stopping criteria mentioned in Fig. 4.1 can be in accordance with value of J-measure and Jmax. In particular, the stopping criteria may be met when the highest value of J-measure observed so far is already higher than the Jmax value achieved at the current node. In addition, the stopping criteria may also be met when the value of J-measure is equal to the Jmax value at the current node. However, the above J-measure based pre-pruning method may not be able to make an optimal tree so that each of the branches in the tree achieves the highest value of J-measure. In other words, the highest value of J-measure may be achieved in a node which is a precursor of the node at which the corresponding branch is stopped growing. Due to the possible presence of this issue, the above pre-pruning method needs to be evolved to a hybrid pruning method by means of combination of pre-pruning and post-pruning strategies. In particular, each branch is forced to stop growing when it is necessary. After the completion of the whole tree generation, each of the branches needs to be converted into a single if-then rule for post-pruning to have this rule achieve the highest value of J-measure. Section 4.3 will illustrate this in detail using a specific example.

On the other hand, decision trees can also be simplified purely by post-pruning each of its branches. The J-measure based post-pruning is illustrated in Fig. 4.2.

In particular, there are two specific methods of J-measure based post pruning. The first method is inspired by SQUEEZE [1] which is illustrated in Fig. 4.3.

The second method is inspired by ITRULE [2] and is illustrated in Fig. 4.4.

The above pruning strategy involves a recursive process for searching for the global maximum of J-measure in order to get the rule with the optimized antecedent. Section 4.3 will illustrate the two methods introduced above in detail using specific examples.

Input: a set of rules *rule[i]*, where i is the index of rule, values of J-measure *J[i][j][k]*, where j is the order of a particular child rule and k is the index of a child rule in the order of j, highest value of J-measure (for *rule[i]*) *max[i]*, order for the optimal rule *m,* index for the optimal rule *n[i]*
Output: a set of simplified rules

```
for each rule rule[i] do
    max[i]=0, m=0, n[i]=0
    for each order j
        for each child rule k
            calculate J[i][j][k]
            if J[i][j][k]> max[i] then
              max[i]= J[i][j][k]
              m=j
              n[i]=k
            end if
            else if J[i][j][k]= max[i]
                take the rule with a higher order
            end else if
        end for
    end for
end for
```

Fig. 4.2 J-measure based tree post-pruning

For each single rule

1. Calculate the J-measure for the rule which is referred to as the parent rule.
2. For each of the child rules generated by removing a single rule term (attribute-value pair) from the parent rule, calculate the J-measure (if the parent rule is order K, each of the child rules is order K-1)
3. Choose the rule among the parent rule and the set of its child rules with highest J-measure. Special cases:
    a) If two rules have the same J-measure, choose the one with the lower order.
    b) If two rules of the same order have the same J-measure, randomly choose one of the two.
4. If the chosen rule is not the parent rule, the chosen rule becomes a new parent rule; repeat steps 2-4. If the chosen rule is the parent rule, terminate the process.

Fig. 4.3 Post-pruning inspired by SQUEEZE [1]

For each single rule
Calculate the J-measure for the rule which is referred to as the parent rule.
Compare the current J-measure with the highest one recorded so far
Update the highest value of the J-measure
For each of its child rules
Repeat the process recursively for calculating the J-measure
Choose the child rule with the highest J-measure. Special cases:
a) If two rules have the same J-measure, the one with the lower order is selected.
b) If two rules have the same J-measure as well as the order, randomly choose one of the two.

Fig. 4.4 Post-pruning inspired by ITRULE

4.2 Pruning of If-Then Rules

As mentioned in Chap. 2, if-then rules can be simplified in two ways namely, pre-pruning and post-pruning. In contrast to tree pruning, the pruning of if-then rules is taken per single rule generated. In other words, each single rule is pruned prior to the generation of the next rule rather than posterior to the completion of the generation of a whole rule set. In this context, pre-pruning aims to stop the specialization of the consequent of a rule. Post-pruning aims to simplify the consequent of a rule after its generation has been completed. The rest of this section introduces a particular type of rule pruning based on J-measure in the manner of both pre-pruning and post-pruning.

In the above context, a J-measure based rule pre-pruning works in the procedures illustrated in Fig. 4.5.

However, the above J-measure based pre-pruning method may not be able to make an optimal rule by means of that the rule achieves the highest value of J-measure. In other words, the highest value of J-measure may be achieved at one term which is a precursor of the term at which the rule is stopped being specialized. Due to the possible presence of this issue, the above pre-pruning method needs to be evoluted to a hybrid pruning method by means of combination of pre-pruning and post-pruning strategies. In particular, a single rule is forced to stop being specialized when it is necessary. Then the rule is simplified by removing some rule terms in order to have the rule achieve the highest value of J-measure. A hybrid J-measure based pruning method is referred to as Jmid-pruning which is introduced in [3]. Section 4.3 will illustrate this method in detail using a specific example.

J-measure based post-pruning for if-then rules is similar to that applied to decision trees. The only difference is that the pruning is taken as soon as a single rule is completely generated prior to the generation of the next rule rather than is taken after the generation of the whole rule set. The two specific methods, which are

```
Input: a set of rules rule[i], a set of rule terms term [i][j], values of J-measure J[i][j],
values of Jmax Jmax[i][j], highest value of J-measure max[i], index for optimal rule index
Output: a set of simplified rules
Initialization: max[i]=0, index=-1
  for each rule rule [i] do
    for each rule term term [i][j]
      calculate J[i][j] and Jmax[i][j]
      if J[i][j]> max[i] then
        max[i]= J[i][j]
        index= j
      end if
      if max[i]> Jmax[i][j]
        stop specialization of rule consequent
      end if
    end for
  end for
```

Fig. 4.5 J-measure based rule pre-pruning

inspired by SQUEEZE and ITRULE respectively, can also be successfully applied to rule post-pruning in the same manner. Section 4.3 will illustrate these two methods mentioned above in detail using specific examples.

4.3 Illustrative Examples

This section illustrates Jmid-pruning, SQUEEZE and ITRULE using examples. For Jmid-pruning, let's see the rule: if a = 2 and b = 1 and c = 3 and d = 1 then class = 3; after adding the four terms subsequently, the corresponding J-measure and Jmax values change in the trend:

If a = 2 then class = 3; (J-measure = 0.110, Jmax = 0.530)
If a = 2 and b = 1 then class = 3; (J-measure = 0.168, Jmax = 0.293)
If a = 2 and b = 1 and c = 3 then class = 3; (J-measure = 0.004, Jmax = 0.055)
Finally: the complete rule as above: (J-measure = 0.048, Jmax = 0.048)

In the above example, Jmid-pruning would stop the rule being specialized after the third rule term c = 3 is added to the left hand side. This is because at that stage the highest J-measure recorded so far (0.168) is already higher than the current Jmax (0.055). The simplified rule is in the form: If a = 2 and b = 1 then class = 3;

For SQUEEZE, the generated rule: if a = 2 and b = 1 and c = 3 and d = 1 then class = 3; needs to be simplified by attempting to remove any one of the rule terms to get a set of its child rules as follows:

Rule 1: if b = 1 and c = 3 and d = 1 then class = 3; (J-measure = 0.029)
Rule 2: if a = 2 and c = 3 and d = 1 then class = 3; (J-measure = 0.122)
Rule 3: if a = 2 and b = 1 and d = 1 then class = 3; (J-measure = 0.159)
Rule 4: if a = 2 and b = 1 and c = 3 then class = 3; (J-measure = 0.221)

On the basis of the above four child rules, the parent rule needs to be simplified to the form of the rule 4 as this rule has the highest J-measure. The new parent rule has three child rules as follows:

Rule 4.1: if b = 1 and c = 3 then class = 3; (J-measure = 0.168)
Rule 4.2: if a = 2 and c = 3 then class = 3; (J-measure = 0.153)
Rule 4.3: if a = 2 and b = 1 then class = 3; (J-measure = 0.127)

On the basis of the above three child rules, the parent rule does not need to be simplified any more as the J-measure is higher than that of any of its child rules. Therefore, the finalized rule is in the form: if a = 2 and b = 1 and c = 3 then class = 3.

However, for ITRULE, in addition to checking child rules of Rule 4, it is also required to check all child rules for each of the other three rules namely, Rule 1, Rule 2 and Rule 3, with regard to their values of J-measure. In other words, a single rule has $2^n - 1$ child rules where n is the number of rule terms. For the above example: if a = 2 and b = 1 and c = 3 and d = 1 then class = 3; there would be 15 child rules (including the above rule itself) in total. In this case, it is necessary to check all the 15 child rules with regard to their values of J-measure and to take the one with the highest J-measure as the finalized version.

4.4 Discussion

This chapter introduces methods for pruning decision trees and if-then rules. In general, pre-pruning is computationally more efficient than post-pruning for both decision trees and if-then rules. This is because the former can usually stop rule generation in advance and thus reduce the computational costs. However, pre-pruning can usually only result in a locally optimal tree or rule set whereas post-pruning can usually get a tree or rule set optimal globally and thus perform a higher level of accuracy.

For decision tree pruning, stopping one branch growing does not affect the growth of any other branches when pre-pruning is adopted. Post-pruning each of braches, which has been converted into a single rule, does not affect the other branches either. However, post-pruning the rules, each of which is converted from one branch of a tree, may make the set of rules unable to fit in a tree structure any more. In addition, these rules may also be overlapping with each other.

For pruning of if-then rules, stopping one rule being specialized on its left hand side will affect the generation of any subsequent rules when pre-pruning is adopted. Post-pruning per rule generated will also encounter the same issue for generation of any subsequent rules.

J-measure is the most appropriate choice for estimation of rule quality as mentioned in [4] since it takes into account both simplicity and measure of fitness of a single rule [2]. In the context of rule pruning, the performance is strongly dependent on the search strategy, which is referred to as search bias [5]. In other words, it is expected to find the global maximum for J-measure among a particular rule and all its child rules with any orders. If the global maximum is successfully found, then the decision tree or rule set would be in the highest quality in theory. In order to achieve this goal, it is necessary to adopt post-pruning strategies that are based on J-measure.

Section 4.1 introduced two specific methods of post-pruning, which are inspired by SQUEEZE and ITRULE respectively. As mentioned earlier, these two methods can be successfully applied to both decision trees and if-then rules for simplification. SQUEEZE is more efficient than ITRULE. This is because the former involves a divide and conquer search and the time complexity is O (log (n)), where n is the number of child rules. For example, there is a rule referred to as parent rule and is represented in the form: if a = 1 and b = 1 and c = 1 then class = 1; (J-measure = 0.28)

This rule has three child rules by removing any one of the three terms as follows:

Rule 1: if b = 1 and c = 1 then class = 1; (J-measure = 0.35)
Rule 2: if a = 1 and c = 1 then class = 1; (J-measure = 0.23)
Rule 3: if a = 1 and b = 1 then class = 1; (J-measure = 0.30)

Rule 1 has two child rules as follows:

Rule 1.1: if b = 1 then class = 1;
Rule 1.2: if c = 1 then class = 1;

Rule 2 and 3 also have two child rules respectively. In the above example, Rule 1 has the highest J-measure so rule 1 becomes the new parent rule in the second iteration of search phase. Therefore, rules 1.1 and 1.2 need to be checked with regard to their values of J-measure but the other child rules for rule 2 and 3 are skipped as specified in Fig. 4.3. On the basis of the above illustration, SQUEEZE has been proven to be able to achieve a divide and conquer search, which makes it unnecessary to examine all of the child rules to find the optimal point and thus reduces the computational cost. However, this method is only able to get a decision tree or rule set locally optimal in most cases. In the above example, the child rules for rule 2 and rule 3 are skipped since the J-measure for the parent rule is higher than that of rule 2 and 3. However, it is possible that at least one of the child rules for rule 2 and rule 3 gets higher J-measure than the parent rule. In other words, the highest J-measure may be found from one of the skipped rules. On the basis of the above description, ITRULE inspired pruning method is most effective in searching for the global maximum of J-measure as all of the child rules need to be examined.

However, as illustrated in Fig. 4.4, this method involves a recursive search for the globally optimal point and thus is computationally more expensive. Therefore, it is necessary to find some appropriate data structures to manage the parent rule and all its child rules by means of storing and searching their corresponding values of J-measure efficiently.

On the other hand, Jmid-pruning mentioned in Sect. 4.2 shows a bias that any one of the rule terms in a rule can be removed if and only if all its subsequent rule terms in the rule are already removed. This is based on the assumption that any one of the rule terms is more important than all its subsequent rule terms in the same rule. If the assumption is not set up in practice, it usually results in bias by means of errors originating from learning algorithms.

References

1. Higgins, C.M.: Classification and Approximation with Rule-Based Networks. California (1993)
2. Smyth, P., Rodney, G.M.: An information theoretic approach to rule induction from databases. IEEE Trans. Knowl. Data Eng. **4**(4), 301–316 (1992)
3. Liu, H., Gegov, A., Stahl, F.: J-measure based hybrid pruning for complexity reduction in classification rules. WSEAS Trans. Syst. **12**(9), 433–446 (2013)
4. Kononenko, I., Kukar, M.: Machine Learning and Data Mining: Introduction to Principles and Algorithms, Chichester. Horwood Publishing Limited, West Sussex (2007)
5. Furnkranz, J.: Separate-and-conquer rule learning. Artif. Intell. Rev. **13**, 3–54 (1999)

Chapter 5
Representation of Classification Rules

As mentioned in Chap. 1, appropriate rule representation is necessary in order to improve model efficiency and interpretability. This chapter introduces three techniques for representation of classification rules namely, decision trees, linear lists and rule based networks. In particular, these representations are illustrated using examples in terms of searching for firing rules. These techniques are also discussed comparatively in terms of computational complexity and interpretability.

5.1 Decision Trees

As mentioned in Chap. 1, decision tree is an automatic representation for classification rules that are generated by a 'divide and conquer' approach. This indicates that if a rule based method that follows the above-named approach is adopted to generate rules, then the rules are automatically represented in a tree structure. As introduced in Chap. 2, appropriate representation of rules is significant for the purpose of both knowledge discovery and predictive modelling.

Decision tree representation is criticized by Cendrowska and identified as a major cause of overfitting in [1] due to the replicated sub-tree problem as illustrated in Fig. 3.5 (See Sect. 3.1). It can be seen from the Fig. 3.5 that the four sub-trees which have node C as their roots are identical. Cendrowska justified in [1] that those rules which have no common attributes are not able to fit in a tree structure and that replicated sub-tree problem would arise if such rules are forced to fit in a tree structure. It is also argued in [1, 2] that it is required to examine the entire tree in order to extract rules about a single classification in the worst case. This drawback on representation makes it difficult to manipulate it for expert systems and thus seriously lowers the computational efficiency in predicting unseen instances. Computational efficiency in testing stage is significant especially when the expert systems to be constructed are time critical [3] if the purpose of using such systems is for predictive modelling.

H. Liu et al., *Rule Based Systems for Big Data*,
Studies in Big Data 13, DOI 10.1007/978-3-319-23696-4_5

On the other hand, decision trees are often quite complex and difficult to understand [4]. Even if decision trees are simplified by using pruning algorithms, it is still difficult to avoid that the decision trees become too cumbersome, complex and inscrutable to provide insight into a domain for knowledge usage [4, 5]. This undoubtedly lowers interpretability of decision trees and is thus a serious drawback for the purpose of knowledge discovery.

The direct use of if-then rules that are represented by a linear list structure has been motivated due to the limitations mentioned above. More details about linear lists are presented in Sect. 5.2.

5.2 Linear Lists

As mentioned in Sect. 5.1, decision tree representation has serious limitations for knowledge discovery and predictive modelling. Therefore, the direct use of if-then rules is motivated. In contrast to decision trees, linear lists do not need to have the constraint that all rules must have at least one common attribute and thus the presence of redundant terms is reduced in a rule set.

However, as if-then rules are represented in a linear list structure, predicting unseen instances in this representation is demonstrated in linear search with the time complexity of O(n), where the total number of rule terms is used as the input size (n). This is because linear list representation follows a linear search by going through the whole set rule by rule in an outer loop; and by going through term by term in an inner loop. The process of linear search can be illustrated by using the example rule set below:

Rule 1: if x1 = 0 and x2 = 0 then y = 0;
Rule 2: if x1 = 0 and x2 = 1 then y = 0;
Rule 3: if x1 = 1 and x2 = 0 then y = 0;
Rule 4: if x1 = 1 and x2 = 1 then y = 1;

On the basis of above rule set, if an instance with two inputs (x1 = 1 and x2 = 1), then it needs to first go through Rule 1 checking the values of x1 and x2 and then move onto Rule 2 taking the same check again until Rule 4 is checked and found firing.

The above description implies that it may have to go through the whole rule set to find the first rule firing in the worst case. This would lead to huge computational costs when the representation is used to represent a rule set generated by learning from large training data. As mentioned in Chap. 2, a rule representation technique is expected to identify the firing rules without the need to examine the whole rule set. Therefore, for the purpose of predictive modelling, linear lists still cannot fulfil the goal with regard to efficient search of firing rules. In this sense, it is necessary to develop another technique of rule representation which demonstrates a level of computational efficiency higher than linear time.

In addition, a large training data would usually result in the generation of a large number of complex rules. In this case, the set of rules represented in a linear list structure would become very cumbersome and difficult to interpret for knowledge usage. In other words, a large number of complex rules represented in a linear list is like a large number of long paragraphs in an article, which would be very difficult for people to read and understand. Instead, people would prefer to look at diagrams to gain information. In this sense, graphical representation of rules would be expected to improve the interpretability of knowledge discovered from data.

Section 5.3 will introduce network based rule representation in terms of both knowledge discovery and predictive modelling purposes.

5.3 Rule Based Networks

Both decision trees and linear lists have limitations in terms of knowledge discovery and predictive modelling purposes. Therefore, it is necessary to develop new techniques for representation of rules. This is in order to achieve a more efficient search of firing rules than linear search as well as to deliver a more interpretable representation of knowledge than decision trees and linear lists do. The rest of this subsection introduces network based rule representations.

In order to improve the computational efficiency in comparison with decision trees and linear lists for the same rule set in the prediction stage, the attribute-value oriented rule based network topology is developed as illustrated in Fig. 5.1.

The above rule based network representation is based on the relationship between attribute values and class labels and is thus referred to as attribute-value oriented rule based network.

In general, this is an $n + 2$ layer network for classification problems. The first n layers represent n input attributes. In each of the layers, each node represents a value of the corresponding attribute. Therefore, the number of nodes in each of the n layers is dependent on the number of values for the corresponding attribute. The last second layer represents the rule index which is equal to the order of this rule minus one. For example, if the rule index is 0, it indicates that this is the first rule in the rule set. The number of nodes in this layer is dependent on the number of rules. The last layer in the network represents the class output and the number of nodes is dependent on the number of classes. There are also connections between different layers, which are to be explained further using specific examples. However, in general, the connections could be between two layers which are not adjacent to each other. For example, the nodes in the first layer could have connections with other nodes in the third layer. This is very like a travel route which includes a number of cities. In this context, each city is like a rule term and each route is like a rule. It is possible that there are cities which are not adjacent to each other but included in the same travel route. In addition, any two nodes may have more than one connection. This is because the same part of conjunction of rule terms may be in two or more rules as illustrated by the rules below:

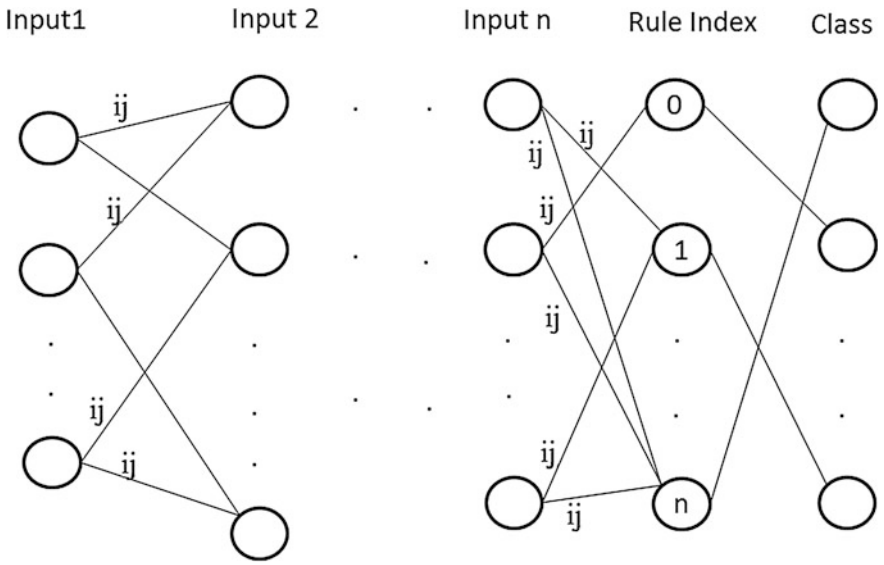

Fig. 5.1 Rule based network (version 1) [8]

If a = 0 and b = 0 and c = 0 then class = 0;
If a = 0 and b = 0 then class = 1;

In the context of travel route as mentioned above, this is like that there could be common cities included in different routes. In other words, there could be more than one path to reach between two cities.

The rule based network representation that is illustrated in Fig. 5.1 would demonstrate a divide and conquer search for firing rules. It could be justified by the example rule set used in Sect. 5.2 for analysis of time complexity for linear lists. The rule set represented by the network representation is illustrated in Fig. 5.2 and the values of the two inputs (x1 and x2) are both 1. Therefore, the two input layers (x1 and x2) both have the nodes labelled 1 and become green as illustrated in Fig. 5.2.

For Rule Based Networks, the prediction on an unseen instance is undertaken by going through rule terms in divide and conquer search (i.e. only going through those terms that fire). The total number of terms is used as the input size of data(n), which is the same as in linear list representation and thus the efficiency is O(log (n)). As it can be seen from Fig. 5.2, it only takes three steps (going through connections '31', '41' and '42') to find the first rule firing (the rule index is 3). This is because the input value of x1 is 1 and thus the connections '11' and '21' can be ignored. In the second layer, only connection '42' is checked since the input value of x2 is 1 and thus the connections '12' and '32' can be ignored. In addition, the connection '22' is ignored as well because the connection '21' is already discarded and thus it is not worth to go through the connection '22' any more.

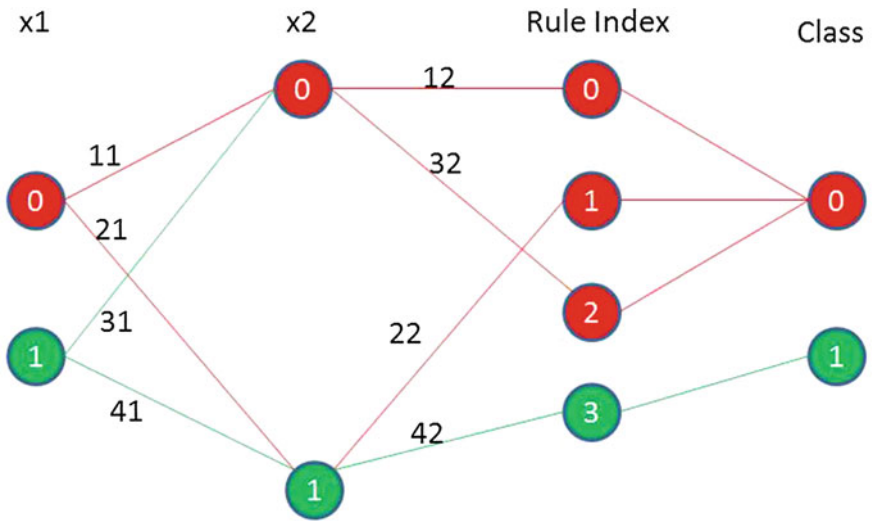

Fig. 5.2 Rule based network example (version 1) [9]

On the other hand, Higgins has developed a representation called rule based network as illustrated in Fig. 5.3. This type of network based representation is based on the relationship between input attributes and class labels and is thus referred to as attribute oriented rule based network, which is fundamentally different from the network topology illustrated in Fig. 5.1.

The network topology illustrated in Fig. 5.3 could be seen as a special type of rule based network representation based on the relationship between input attributes and class labels. This is because of the possibility that there are two or more rules that fire with different classifications as rule consequences. This issue needs to be resolved by conflict resolution strategies as introduced in [6]. Higgins's network topology actually takes into account this conflict and deals with it by the 'Winner-Take-All strategy' [7]. Therefore, the network topology could be seen as a type of non-deterministic rule based network with certain inputs but uncertain outputs. However, the conflict mentioned above would never arise with the rule sets that are generated by adopting the divide and conquer approach. In this context, if the rule generation is based on deterministic logic, both inputs and outputs would be deterministic. As it is, the networked topology is modified to become a deterministic rule based network that is illustrated by Fig. 5.4.

In general, this is a three layer network. In the first layer, each node represents an input attribute and this layer is referred to as input layer. The number of nodes in this layer is dependent on the number of attributes in a data set. In the middle layer, each node represents a rule to make the conjunction among inputs and provide outputs for the node in the last layer and thus the middle layer is referred to as conjunction layer. The number of nodes in this layer is dependent on the number of rules generated. The only node in the last layer represents the class output and thus

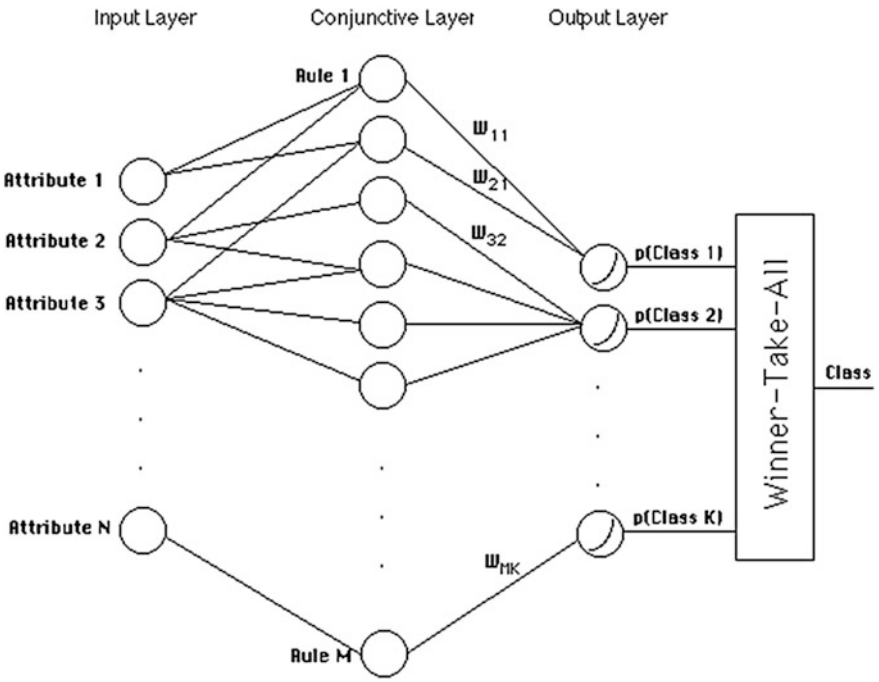

Fig. 5.3 Higgins's non-deterministic rule based network for classification [7]

this layer is referred to as output layer. In addition, the nodes in input layer usually have connections to other nodes in the conjunction layer. Each of the connections represents a condition judgment which is explained further using specific examples. However, a node in the input layer may not necessarily have connections to other nodes in the conjunction layer. This is due to a special case that an attribute may be totally irrelevant to making a classification, which is described in more depth in Sect. 5.4.

The example rule set that is used in Sect. 5.2 and represented by this network topology is illustrated in Fig. 5.5. In this diagram, both input values are supposed to be 1 (shown as green) and each node in input layer represents an input attribute; each node in the middle layer represents a rule and the layer is referred to as conjunction layer due to the fact that each rule actually reflects the mapping between inputs and outputs and that the output values strongly depends on the conjunction of input values; finally, the node in the output layer represents the class attribute. On the other hand, each of the connections between input layer and conjunction layer represents a condition judgment. If the condition is met, then the connection is coloured by green. Otherwise, it is coloured by red. In addition, each of the connections between the conjunction layer and the output layer represents an output value from the corresponding rule. In other words, if all of the conditions in a rule are met, then the corresponding node in the conjunction layer becomes green.

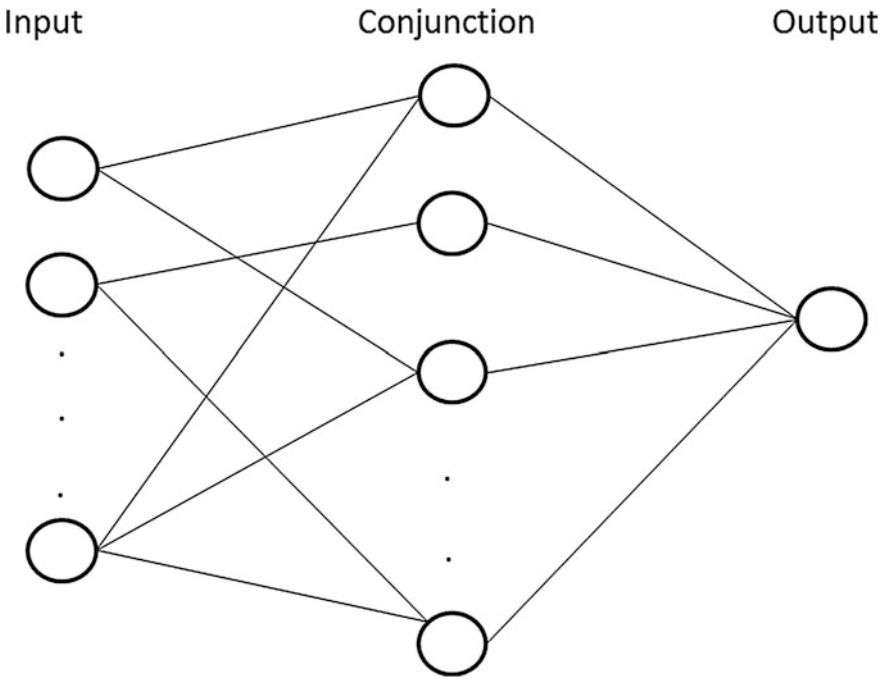

Fig. 5.4 Deterministic rule based network (version 2) [8]

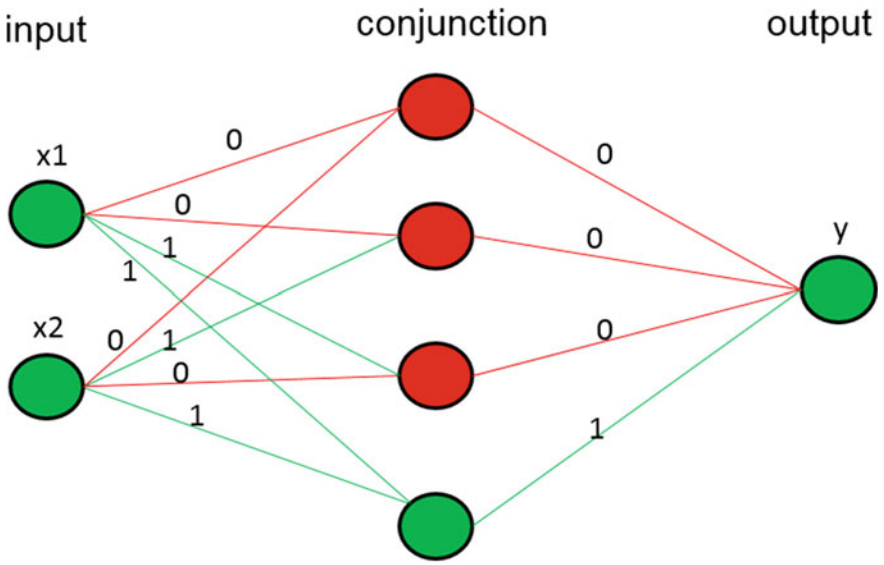

Fig. 5.5 Deterministic rule based network example (version 2) [8]

Otherwise, the corresponding node becomes red. The former case would result in that a node representing a rule becomes green and that the output value from the rule is assigned to the class attribute in the output layer. In the meantime, the connection between the node representing the rule and another node representing the class attribute becomes green, which means the class attribute would be assigned the output value from the rule. In contrast, the latter case would result in that the node in conjunction layer becomes red and that the output value from the corresponding rule cannot be assigned to the class attribute.

As mentioned earlier in this subsection, a rule set may have some or all rules non-deterministic in terms of relationships between rule antecedents and consequents due to the presence of uncertainty in datasets. In this context, the rule set would be used to predict classes based on probabilistic or fuzzy logic. Therefore, a unified topology for rule based networks, which could fulfil being based on different logics such as deterministic, probabilistic and fuzzy logic, is developed and illustrated in Fig. 5.6.

In this network topology, the modifications are made to the one illustrated in Fig. 5.6 by adding a new layer called disjunction and assigning a weight to each of the connections between nodes. The disjunction layer is similar to the output layer in Higgins's network topology illustrated in Fig. 5.3. In this layer, each node represents a class label and the number of nodes is dependent on the number of classes. However, the final prediction is not necessarily made by choosing the most common class which has the highest posteriori probability in total. In contrast to Figs. 5.4 and 5.5, the topology also allows representing inconsistent rules, which means that the same rule antecedent could be mapped to different classes (consequents). For example, the first node in the conjunction layer is mapped to both the first and the second node in the disjunction layer as illustrated in Fig. 5.6. With

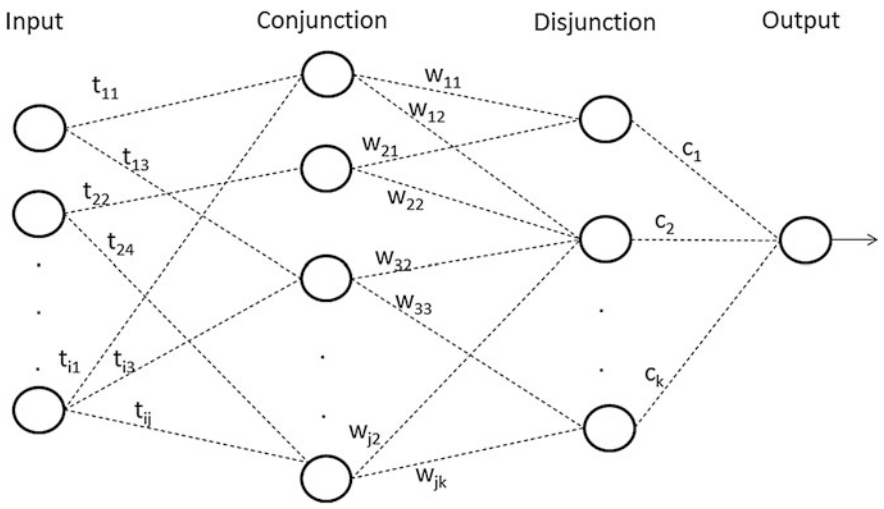

Fig. 5.6 Unified rule based network [8]

regard to the weights assigned to the connections between nodes, they would represent the truth values if the computation is based on deterministic or fuzzy logic. The truth value would be crisp (0 or 1) for deterministic logic whereas it would be continuous (between 0 and 1) for fuzzy logic. If the computation is based on probabilistic logic, the weights would represent the probabilities of the corresponding cases.

In the context of deterministic logic, each of the connections between the nodes in the input layer and the nodes in the conjunction layer would be labelled 1 for its weight, i.e. $t_{ij} = 1$ where i is the index of the attribute and j is the index of the rule, if the corresponding condition as part of the rule antecedent is met. A rule would have its antecedent satisfied if and only if all of the conditions are met. In this case, the rule is firing to indicate its consequent (as the class predicted) which is represented by a node in the disjunction layer. If the rule is consistent, the corresponding node should have a single connection to another node in the disjunction layer. The connection would be labelled 1 as its weight denoted by w_{jk}, where k is the index of the class. In this case, if there is only one rule firing or more rules firing without conflict of classification, then the output would be deterministic. This is because there is only one node in the disjunction layer providing a weight greater than or equal to 1 for its connection to the node in the output layer. For all other nodes, the weight provided for the corresponding connection would be equal to 0.

However, as mentioned earlier, a rule may be inconsistent, which means that the same rule antecedent may be mapped to different classes as its consequent. In this case, the corresponding node would have multiple connections to different nodes in the disjunction layer. For each of the connections, the weight would be equal to a value between 0 and 1. Nevertheless, the sum of weights for the connections would be equal to 1. With regard to each of classes, it may be mapped from different rule antecedents. Therefore, each class would have a summative weight denoted by c_k, which is equal to the sum of the weights for the rule antecedents mapped to the class. Finally, the node in the output layer makes the weighted majority voting for the final prediction.

In the context of probabilistic logic, the t_{ij} would be equal to a value between 0 and 1 as a conditional probability. Similar to deterministic logic, a rule is firing if and only if all of the conditions are met. However, the rule antecedent would be assigned a firing probability computed in the corresponding node in the conjunction layer. The firing probability is simply equal to the product of the conditional probabilities for the rule terms (if corresponding attributes are independent) and also to the posterior probability of the rule consequent given the rule antecedent. If the rule is inconsistent, the sum of posterior probabilities for the possible classes (w_{jk}) would also be equal to the firing probability above. This is because the rule consequent is the disjunction of the output terms, each of which has a different class as the output value. In the disjunction layer, each class is assigned a weight which is equal to the sum of its posterior probabilities given different rule antecedents. The final prediction is made by weighted voting in the same way as based on deterministic logic.

In the context of fuzzy logic, in contrast to probabilistic logic, in the conjunction layer, the t_{ij} would be equal to a value between 0 and 1 as a fuzzy truth value for each corresponding condition. Similar to the other two types of logic, a rule is firing if and only if all of the conditions are met. However, the rule antecedent would be assigned a firing strength computed in the corresponding node in the conjunction layer. The firing strength is simply computed by choosing the minimum among the fuzzy truth values of the conditions (that are assumed independent). The fuzzy truth value for the rule consequent is equal to the firing strength. If the rule is inconsistent, the fuzzy truth value (w_{jk}) for having each possible class as the consequent would be derived by getting the minimum between the firing strength and the original fuzzy truth value assigned to this class for this rule. In the disjunction layer, the weight for each class is computed by getting the maximum among the fuzzy truth values (w_{jk}) of the rules having the class as the consequents. The final prediction is made by weighted majority voting in the same way as the above two types of logic.

5.4 Discussion

The descriptions on attribute-value oriented rule based network in Sect. 5.3 indicates that finding the rules that fire by using this network representation does not involve examining the whole network. This makes the efficiency of the rule based network higher than that of linear lists, the latter of which is O (n) as mentioned in Sect. 5.3 and Table 5.1.

In practice, the advantage of rule based networks mentioned above would significantly speed up the process of prediction making when the corresponding rule set is generated through learning from large training data.

As mentioned in Chap. 2, rule representation is also significant in fulfilling the requirement of interpretable representation of knowledge. In this context, the attribute-value oriented rule based network topology, which is illustrated in Fig. 5.1, would also demonstrate a good interpretability with respect to correlation between inputs and outputs. A correlation could be interpreted by connections nodes in different layers. For example, it can be seen from Fig. 5.2 that node 1 in the first layer (x1) is firstly connected to node 1 in the second layer (x2) and finally to

Table 5.1 Comparison in efficiency [8]

Decision tree	Linear list	Rule based network
O(log(n)), which indicates it is not required to examine a whole tree but the value of n is likely to be higher than that in the other two representations due to the presence of redundant terms	O(n), which indicates it is required to examine a whole list in the worst case	O(log(n)), which indicates it is not required to examine a whole network

node 1 in the last layer (Class) via node 3 in the third layer (Rule Index). These connections make up a path that interprets a correlation between the two input attributes (x1 and x2) and the class attribute. In comparison with linear list, the network representation only needs each attribute and its corresponding values to appear once as a layer and a node in the layer respectively. This also reduces the structure complexity and thus improves the interpretability of knowledge.

Section 5.3 also introduced another type of network topology, which is referred to as attribute oriented rule based network. As mentioned in Sect. 5.3, a node in the input layer of the network topology illustrated in Fig. 5.4 may not necessarily have connections to the nodes in the conjunction layer. This is due to the possibility that an input attribute may be totally irrelevant to making a classification. In other words, this attribute is not involved in any rules in the form of rule terms. From this point of view, this topology of rule based network can be used to identify the relevance of attributes for feature selection tasks which is listed in Table 5.2.

On the other hand, this type of network representation is based on the relationship between input attributes and class labels as mentioned in Sect. 5.3. Therefore, this representation can be used to reflect correlations between input attributes and class labels, i.e. it enables the identification of the input attributes that have the highest impact on determining each of the class labels.

As illustrated in Table 5.2, this type of networked rule representation also shows the relationship between attributes and rules explicitly as shown connections between nodes in input layer and nodes in conjunction layer. In addition, the network topology also introduces a ranking for both input attributes and rules based on their importance. The importance of an input attribute is measured by the weighted average of ranks for those rules that relate to the input attribute. For example, an attribute A relates to two rules namely rule 1 and rule 2. If the ranks for rule 1 and rule 2 are 4 and 8 respectively, then the average of ranks would be 6 = ((4 + 8)/2). In real applications, this characteristic about ranking of attributes may significantly contribute to both knowledge discovery and feature selection with respect to feature importance. Besides, the strength of the representation also lies in the strong interpretability on mapping relationship between inputs and outputs, which is significantly useful for knowledge discovery. On the basis of the above descriptions, the rule based network representation illustrated in Fig. 5.4 is thus a practically significant technique in data mining tasks.

Table 5.2 Comparison in interpretability [8]

Criteria	DT	LL	RBN
Correlation between attributes and classes	Poor	Implicit	Explicit
Relationship between attributes and rules	Implicit	Implicit	Explicit
Ranking of attributes	Poor	Poor	Explicit
Ranking of rules	Poor	Implicit	Explicit
Attribute relevance	Poor	Poor	Explicit
Overall	Low	Medium	High

NB DT = Decision tree, LL = Linear list and RBN = Rule based network

As introduced in Sect. 5.3, a unified network topology, which is illustrated in Fig. 5.6, is developed in order to fulfil the requirement that the representation can be used based any types of logic, such as deterministic logic, probabilistic logic and fuzzy logic.

The unified network topology has some advantages. In particular, the representation does not only show which input attributes are most significant for each class label in terms of determining the class label of an unseen instance, but also measure the corresponding degree of likelihood. It is important especially for fuzzy rule based systems, which is required to assign a weight to each of the connections between nodes. This is because each of the connections is only involved in one rule in this representation. In contrast, decision tree representation may have the same connection shared by different rules with the need that different weights are assigned to the same connection for different rules, which results in confusions. On the other hand, if a linear list representation has each single rule term assigned a weight, it is likely to make the rules less readable. The similar problems also arise when decision trees and linear lists are used to represent probabilistic rule based systems that contain inconsistency in terms of mapping between rule antecedents and rule consequents. However, the above limitations can be effectively overcome by using the unified rule based network topology. Overall, the above description demonstrates a significant advantage of using the unified rule based network topology for knowledge discovery in real applications due to the presence of uncertainty.

References

1. Cendrowska, J.: PRISM: an algorithm for inducing modular rules. Int. J. Man Mach. Stud. **27**, 349–370 (1987)
2. Deng, X.: A Covering-based Algorithm for Classification: PRISM, SK (2012)
3. Gegov, A.: Complexity Management in Fuzzy Systems. Springer, Berlin (2007)
4. Furnkranz, J.: Separate-and-conquer rule learning. Artif. Intell. Rev. **13**, 3–54 (1999)
5. Quinlan, J.R.: C 4.5: Programs for Machine Learning. Morgan Kaufman, San Mateo (1993)
6. Holland, A.: Lecture 2: Rules Based Systems (2010)
7. Higgins, C.M.: Classification and Approximation with Rule-Based Networks. California (1993)
8. Liu, H., Gegov, A., Cocea, M.: Network based rule representation for knowledge discovery and predictive modelling. In: IEEE International Conference on Fuzzy Systems, Istanbul (2015)
9. Liu, H., Gegov, A., Stahl, F.: Unified framework for construction of rule based classification systems. In: Pedrycz, W., Chen, S. (eds.) Information Granularity, Big Data and Computational Intelligence, vol. 8, pp. 209–230. Springer (2015)

Chapter 6
Ensemble Learning Approaches

As mentioned in Chap. 1, ensemble learning is helpful to improve overall accuracy of classification. This chapter introduces three approaches of ensemble learning namely, parallel learning, sequential learning and hybrid learning. In particular, some popular methods for ensemble learning, such as Bagging and Boosting, are illustrated in detail. These methods are also discussed comparatively with respects to their advantages and disadvantages.

6.1 Parallel Learning

As mentioned in Chap. 1, parallel ensemble learning approach can be achieved through a combination of different learning algorithms, each of which generates a model independently on the same training set. In this way, the predictions of the models generated by these algorithms are combined to predict unseen instances. This way belongs to scaling up algorithms because different algorithms are combined in order to generate a stronger hypothesis. In addition, the parallel ensemble learning approach can also be achieved by using a single base learning algorithm to generate models independently on different sample sets of training instances. In this context, the sample set of training instances can be provided through horizontal selection of the instances with replacement or vertical selection of the attributes without replacement. This way belongs to scaling down data because the training data is preprocessed to reduce the variance that exists on the basis of the attribute-values. Some popular approaches of parallel learning include Bagging and Random Forests.

The term Bagging stands for bootstrap aggregating. It is a popular method developed by Breiman in [1] and the basic procedures of this method are illustrated in Fig. 6.1. This method follows the parallel ensemble learning approach as mentioned above. Bagging involves sampling of data with replacement. In detail, the Bagging method is to draw a sample with the size n, where n is the size of the training set, and to randomly select instances from the training set to be put into the sample set. This indicates that some instances in the training set may appear more than once in the sample set and some other instances may never appear in the

H. Liu et al., *Rule Based Systems for Big Data*,
Studies in Big Data 13, DOI 10.1007/978-3-319-23696-4_6

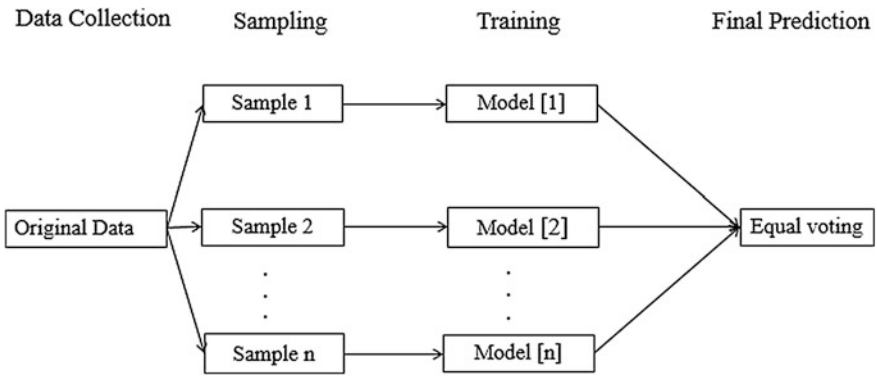

Fig. 6.1 Bagging method

sample set. On average, a sample is expected to contain 63.2 % of the training instances [1–3]. In the training stage, the generation of classifiers, each of which results from a particular sample set mentioned above, are parallel to each other. In testing stage, the combination of their independent predictions is made to predict the final classification based on equal voting. As concluded in the literatures [2, 3], Bagging is robust and does not lead to overfitting due to the increase of the number of generated models, and thus is useful especially for those non-stable learning methods such as neural networks, decision trees and rule based learning methods.

Random forests is another popular method introduced in [4], which can be seen as a special case of bagging as illustrated in Fig. 6.2. In particular, decision trees must be the base classifiers generated on each sample of training data. In addition, the attribute selection at each node of a decision tree is random to some extents. Otherwise, this ensemble learning method only belongs to Bagging. In this sense, at each node, there is a subset of attributes randomly chosen from the training set and the one which can provide the best split for the node is finally chosen [5]. In the

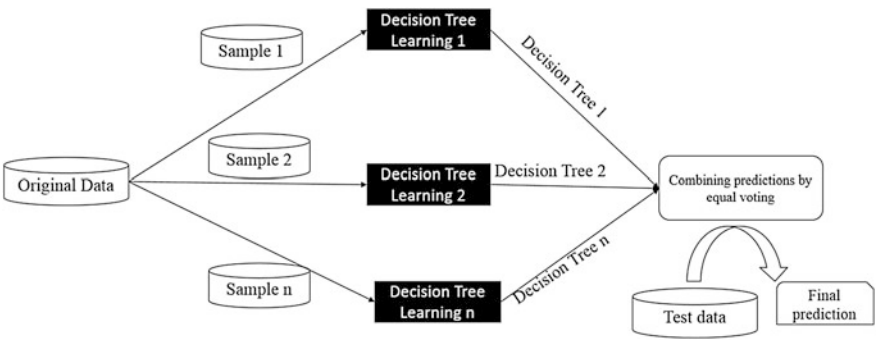

Fig. 6.2 Random forests

training stage, the chosen algorithm of decision tree learning is used to generate classifiers independently on the samples of the training data. In the testing stage, the classifiers make the independent predictions that are combined to make the final prediction based on equal voting. As concluded in the literature [2], the random forests algorithm is robust because of the reduction of the variance for decision tree learning algorithms.

The CCRDR approach of ensemble learning has been developed by Liu and Gegov in [6], in order to fill the gap that exists in Random Forests and other similar methods. The basic idea of this approach is illustrated in Fig. 6.3.

CCRDR stands for Collaborative and Competitive Random Decision Rules, which indicates that the ensemble learning framework includes both cooperative learning and competitive learning. Therefore, the above approach is designed partially to overcome the limitation that there is only a single base learning algorithm involved in the training stage, which cannot always generate robust rule sets due to the absence of competition in this stage. In order to overcome the above limitation, the CCRDR approach is designed in a way that includes multiple base algorithms for training. On the basis of the design, there is thus competition among the rule sets generated on a same sample of training data. In other words, there are multiple learning algorithms applied to each sample of the training data, which results in the generation of multiple rule sets on each sample. In this context, it becomes achievable to find better rule sets to be involved in testing stage and worse ones to be absent through competition among these rule sets. The competition is based upon the weight (confidence) of each of the rule sets by means of overall accuracy estimated using validation data. In the CCRDR framework, only the rule set with the highest weight (confidence) is eligible to be involved in the testing stage. The development of the CCRDR aims to enable that on each sample of data the hypothesis constructed becomes much stronger.

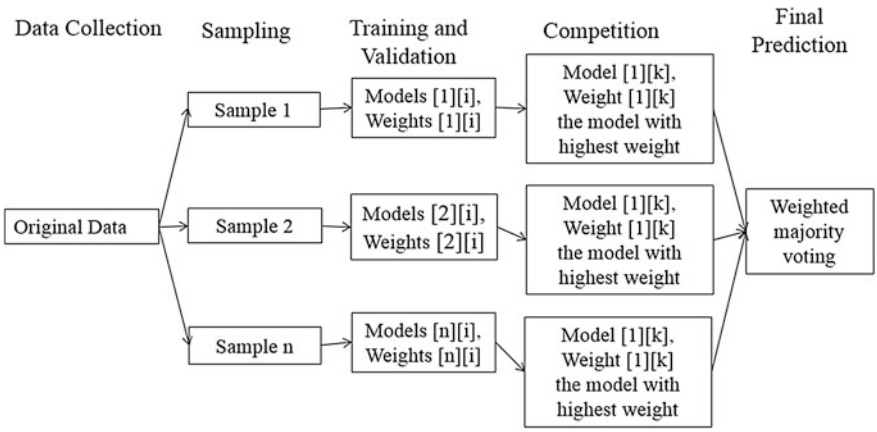

Fig. 6.3 CCRDR ensemble learning [6]

On the other hand, the CCRDR framework is also designed to address the issues relating to voting. As mentioned in Chap. 2, voting can be based on different criteria such as equal voting and weighted voting. For the two popular methods introduced above, Bagging and Random Forests are based on equal voting. In addition, Boosting, which is introduced in Sect. 6.2, is based on weighted voting. In classification tasks, the weighted voting is preferred to the equal voting. This is due to the possibility that some classifiers are highly reliable whereas the others are less reliable. For example, there are three base classifiers: A, B and C. A predicts the classification X with the weight 0.8, and B and C predict the classification Y with the weights 0.55 and 0.2 respectively so the final classification is X if using weighted voting (weight for X: $0.8 > 0.55 + 0.2 = 0.75$) but is Y if using equal voting (frequency for Y: $2 > 1$). Liu and Gegov also discussed in [6] several possible ways for determining the weight for weighted voting. These ways are based on overall accuracy, precision and recall respectively, all of which are popularly used for evaluating the quality of a classifier. These ways are also compared experimentally in terms of their effectiveness for evaluating the reliability of a classifier in ensemble learning tasks. The experimental results indicate that precision is more effective than the other two. In addition, the justifications described in the following paragraphs also strongly support the above indication from the experimental results.

The strategy based on overall accuracy is not reliable enough in determining the weight, especially for unbalanced data sets. This is due to the high possibility that a classifier performs better on predicting positive instances but worse on negative instances if it is a two class classification task. The similar cases can also happen in multi-class classification tasks. Therefore, it is necessary to adopt the individual accuracy (recall) for a single classification as the weight [6].

Precision would be more reliable than recall in determining the weight of a classifier. For example, there are 5 positive instances out of 20 in a test set and a classifier correctly predicts the 5 instances as positive but incorrectly predicts other 5 instances as positive as well. In this case, the recall/true positive rate is 100 % as all of the five positive instances are correctly classified. However, the precision on positive class is only 50 %. This is because the classifier predicts 10 instances as positive and only five of them are correct. This case indicates the possibility that high recall could result from coincidence due to low frequency of a particular classification. Therefore, precision is more reliable in determining the weight of a classifier on a particular prediction from this point of view [6].

The CCRDR framework still has limitations. One of them is due to the absence of collaboration among different learning algorithms in the training stage. Therefore, this framework only follows the parallel ensemble learning approach and has no connection to sequential ensemble learning. In particular, these learning algorithms do not collaborate with each other. For separate and conquer rule learning, collaboration can be achieved in the way that different rule learning algorithms are combined to generate a single rule set on each sample of training data. More details on this are justified in Sect. 6.3.

6.2 Sequential Learning

In sequential ensemble learning approach, accuracy can also be improved through scaling up algorithms or scaling down data. In the former way, different algorithms are combined in the way that the first algorithm learns to generate a model and then the second algorithm learns to correct the model and so on. In this way, there is nothing changed to training data. In the latter way, in contrast, the same algorithm is used iteratively on different versions of the training data. In each iteration, there is a model generated and the model is then evaluated using the validation data. According to the estimated quality of the model, the training instances are weighted to different extents and then used for the next iteration. In the testing stage, these models generated at different iterations make predictions independently and their predictions are then combined to predict unseen instances.

The term Boosting stands for Adaboost. It is introduced in [7–9] and follows sequential ensemble learning approach as illustrated in Fig. 6.4. In other words, the generation of a single classifier depends on the experience gained from its former classifier [5]. Each single classifier is assigned a weight depending on its accuracy estimated by using validation data. The stopping criteria are satisfied while the error is equal to 0 or greater than 0.5 [5]. In the testing stage, each single classifier makes an independent prediction as similar to Bagging but the final prediction is made

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X, y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : X \to \{-1, +1\}$ with error

$$\epsilon_t = \Pr_{i \sim D_t}\left[h_t(x_i) \neq y_i\right].$$

- Choose $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$.
- Update:

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} \\ &= \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t} \end{aligned}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \text{sign}\left(\sum^{T} \alpha_t h_t(x)\right).$$

Fig. 6.4 The boosting algorithm AdaBoost [7]

based on weighted voting among these independent predictions. As concluded in the literature [2], Boosting frequently outperforms Bagging, and can also be applied with those stable learning algorithms with low variance in addition to unstable ones, in contrast to Bagging. However, the Boosting may generate an ensemble learner that overfits training data. In this case, the performance of the ensemble learner is worse than that of a single learner.

On the other hand, sequential learning can be achieved through the combination of methods in rule generation and rule simplification. In other words, rule generation methods aim to learn a set of rules, each of which is corrected by rule simplification methods. As mentioned in Chap. 1, rule simplification can be achieved through pre-pruning and post-pruning. In addition, rule generation may be in the form of decision trees or if-then rules. In the former way, each of the single rules is corrected by pruning methods when it is being learned. In particular, if a decision tree is being generated, the correction by pruning methods is to stop growing a branch of the decision tree. Otherwise, the correction aims to stop specializing the left hand side of a single rule. In the latter way, each of the rules is corrected after the rule learning has been completed. In particular, if a decision tree is being generated, the tree should be converted to a set of if-then rules, each of which is then simplified on its left hand side by pruning methods, after the complete tree has been generated. Otherwise, each single rule need to be simplified immediately after the completion of generation for the current rule and before the start of generation for the next rule.

On the basis of above description, the approach described above can be seen as a new way to achieve sequential ensemble learning that involves collaborations among different algorithms. However, this approach is only applied to the construction of rule based systems due to the nature of rule learning.

6.3 Hybrid Learning

As mentioned in Sect. 6.1, separate and conquer rule learning can be enhanced through the way that different algorithms collaborate to generate a single rule set on a given data set. Therefore, the CCRDR framework still has gaps to be filled. The modified framework is illustrated in Fig. 6.5. The new framework is referred to as hybrid ensemble rule based classification [10], due to the involvement of data sampling and algorithms collaborations for reduction of bias and variance respectively.

The modified framework for ensemble rule based classification differs from the CCRDR illustrated in Fig. 6.3 in two aspects.

The first aspect is on the number of generated rule based classifiers learning from each sample of training data, i.e. there is only one classifier generated from each sample data.

The second aspect is on the removal of the competition stage illustrated in Fig. 6.3. In other words, after the completion of training and validation, all the

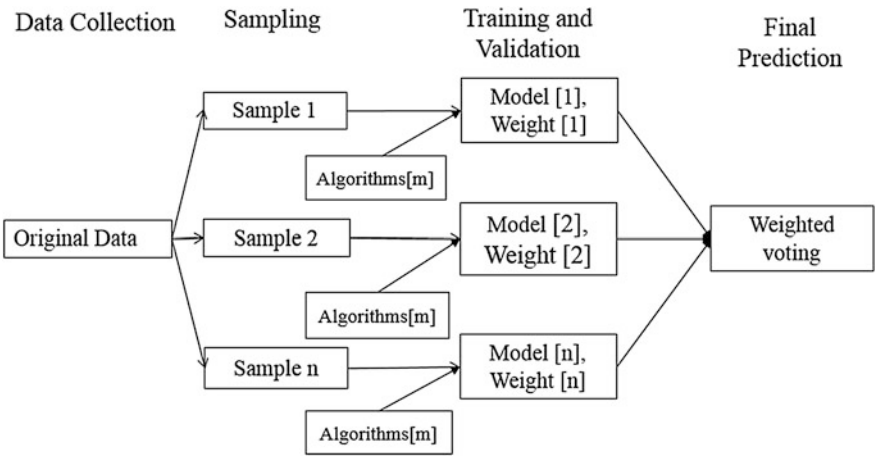

Fig. 6.5 Hybrid ensemble rule based classification framework [10]

classifiers are not involved in competition with regard to their reliabilities but are combined straightaway to predict unseen instances.

The two aspects for modification made to the CCRDR framework are due to the change in the strategy for combination of different learning algorithms. In the modified framework, the learning algorithms are combined to generate only a single classifier on each sample of training data in the way as illustrated in Fig. 6.6. Therefore, there is no competition necessary.

The combination of learning algorithms involves collaborations in a Macro vision but competitions in a Micro vision. This is because these algorithms are combined to generate a rule set (a set of rules), but each of these rules is actually generated through competitions among rules generated by different algorithms at each iteration of rule learning. In other words, at each iteration, each of the chosen algorithms generates a rule that is compared with the other rules generated by other algorithms in terms of rule quality. The competitions mentioned above aim to have each single rule with a quality as high as possible for each rule set generated from a sample of training data. The quality of a single rule can be estimated by some statistical measures, such as confidence, J-measure, lift and leverage, as introduced in Chap. 1.

The hybrid ensemble rule based classification framework can fill the gaps that exist in Random Forests and CCRDR to a large extent.

In comparison with Random Forests, the new framework illustrated in Fig. 6.5 employs algorithms that follow separate and conquer rule learning. This rule learning approach has a higher flexibility than divide and conquer rule learning.

Divide and conquer approach involves generating a set of rules, all of which are strongly connected with each other to fit in a tree structure. In this context, changing even one rule is likely to destroy the whole tree as a result of that the rules cannot fit in a tree any more. This is because each rule is represented as a branch of the tree

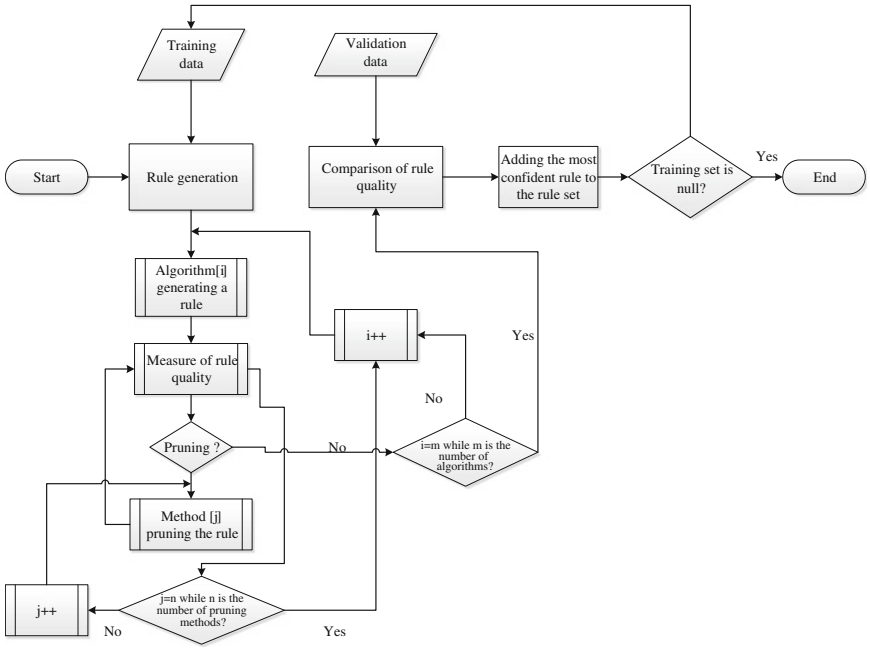

Fig. 6.6 Collaborative rule generation framework

that has some common parts with other branches. On the other hand, each of the branches is grown parallel to all the others.

In contrast, separate and conquer approach involves generating a set of modular rules. These rules are not connected each other in terms of rule representation. On the other hand, the rules are generated sequentially, i.e. the completion of the generation for one rule is followed by the start of the generation for another rule.

On the basis of above comparison, separate and conquer approach enables collaborations involved in the training stage of ensemble rule learning whereas divide and conquer approach cannot. This is because the former approach enables the evaluation of each single rule once the rule is generated due to the sequential rule generation, whereas the latter approach cannot achieve it due to parallel rule generation. Therefore, the hybrid ensemble rule based classification framework can fill the gap that exists in Random Forest.

In comparison with the CCRDR framework, the improvement is on the quality of the classifiers, each of which is generated from a particular sample of training data. This is because of the possibility originating from CCRDR that some of the rules are of higher quality but the others are of lower quality, while there is only one rule learning algorithm involved in the training stage. CCRDR involves a competition between algorithms per rule set. In other words, the competition is made after each of the algorithms has generated a rule set, in order to compare the quality of a whole rule set. In contrast, the hybrid ensemble approach involves such a

competition per rule generated. In other words, the competition is made once each of the algorithms has generated a rule in order to compare the quality of each single rule generated by a particular algorithm.

Overall, in the context of ensemble rule based classification, the new framework illustrated in Fig. 6.5 shows a greater robustness and flexibility than Random Forests and CCRDR.

6.4 Discussion

As mentioned in [2], Bagging, Boosting and Random Forests all have the disadvantage of incomprehensibility of the predictions made by different models. The same disadvantage also arises with the CCRDR approach. This is a serious drawback that arises with most existing ensemble learning approaches for data mining tasks. As mentioned in Chap. 1, data mining is aimed at knowledge discovery. Therefore, it is necessary for the models to allow explicit interpretation of the way that the prediction is made. The approach by combining methods in rule generation and rule simplification respectively, which is introduced in Sect. 6.2, would be able to fill the gap to some extent as it only generates a single rule set that is used for prediction. In addition, rule based models are highly interpretable as mentioned in Sect. 1; consequently, the above approach would fit well the purpose of knowledge discovery especially on interpretability.

With regard to accuracy, Bagging, Boosting and Random Forests all aim to improve it on the data side (scaling down data) for reduction of variance. However, there is nothing done on the algorithms side (scaling up algorithms) for reduction of bias. As mentioned in Chap. 1, it is necessary to deal with issues through reduction of both bias and variance in order to comprehensively improve the accuracy. The CCRDR can fulfil the need to a large extent. As justified in [6], each algorithm may have a different level of suitability to different data sets. On the same data set, different algorithms may also demonstrate different levels of performance. From this point of view, the CCRDR approach is designed in the way that after the training data is scaled down by drawing different samples, a group of learning algorithms are combined to generate a model on each of the samples. In this context, the CCRDR approach does not only scale down the data but also scale up the algorithms. However, as mentioned in Sect. 6.3, this approach does not involve any collaborations among different algorithms in the training stage. For rule based learning algorithms, it is very likely to generate a rule set that has some rules with a high quality but also others with a low quality. In other words, it is difficult to guarantee that each single rule generated by a particular algorithm is of high quality. In this sense, the CCRDR approach is only able to select the rule sets, each of which is generated on a particular sample set and has the highest quality on average compared with the others generated on the same sample set. In the testing stage, a rule set usually makes a prediction using a single rule that fires. If the single rule has a low quality, it is very likely to make an incorrect prediction although most of the

other rules have high quality. On the other hand, for data mining tasks, each of the rules is used to provide knowledge insight for domain experts. Therefore, the reliability of each single rule is particularly significant.

On the basis of the above description, the approach that involves combination of rule learning algorithms and pruning algorithms would be useful and effective to help the CCRDR fill the gap relating to the quality of each single rule as well as the incomprehensibility of predictions, and it thus also complements the other three popular methods mentioned earlier. Therefore, the above approach would be strongly motivated for further investigation to advance rule learning algorithms.

On the other hand, a hybrid rule based classification framework, which is introduced in Sect. 6.3, involves reduction of bias and variance through scaling up algorithms and scaling down data respectively. The results reported in [10] indicate that it is necessary to take both scaling up algorithms and scaling down data in order to comprehensively improve classification accuracy like the hybrid ensemble rule based classification framework. In this way, overall classification accuracy can be improved more comprehensively. In contrast, random forests only involve scaling down data and nothing on scaling up algorithms. Therefore, random forests only enable the reduction of variance on data side, but are biased on the decision tree learning algorithm employed. CCRDR enables the reduction of both bias and variance. However, on algorithms side, the chosen algorithms do not collaborate with each other and thus the reduction of bias is not sufficient. This could be explained by the assumption that each algorithm may generate a rule set that has some rules of high quality but the others of low quality. In other words, it cannot ensure that each single rule is generated to have a high quality and thus may result in incorrect classifications by low quality rules.

On the basis of above discussion, the hybrid ensemble rule based classification framework is strongly motivated due to its flexibility in employing rule learning algorithms and rule quality measures, as well as its involvement that different rule learning algorithms collaborate to complement each other. In particular, this framework will be investigated further with respect to the employment of rule learning algorithms and measures of rule quality. In this way, any bias originating from the employed learning algorithms or statistical measures will be avoided to a larger extent and thus accuracy will be improved further.

References

1. Breiman, L.: Bagging predictors. Mach. Learn. **24**(2), 123–140 (1996)
2. Kononenko, I., Kukar, M.: Machine Learning and Data Mining: Introduction to Principles and Algorithms, Chichester. Horwood Publishing Limited, West Sussex (2007)
3. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Pearson Education Inc, New Jersey (2006)
4. Breiman, L.: Random Forests. Mach. Learn. **45**(1), 5–32 (2001)

5. Li, J., Wong, L.: Rule based data mining methods for classification problems in biomedical domains. In: 15th European Conference on Machine Learning and 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, Pisa (2004)
6. Liu, H., Gegov, A.: Collaborative decision making by ensemble rule based classification systems. In: Pedrycz, W., Chen, S. (eds.) Granular Computing and Decision-Making: Interactive and Iterative Approaches, vol. 10, pp. 245–264. Springer, (2015)
7. Freund, Y., Schapire, R.E.: A short introduction to boosting. J. Japan. Soc. Artif. Intell. **14**(5), 771–780 (1999)
8. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. **55**(1), 119–139 (1997)
9. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Machine Learning: Proceedings of the Thirteenth International Conference, Bari (1996)
10. Liu, H., Gegov, A., Cocea, M.: Hybrid ensemble learning approach for generation of classification rules. In: International Conference on Machine Learning and Cybernetics, Guangzhou (2015)

Chapter 7
Interpretability Analysis

Chapter 1 stressed the significance of interpretability for the purpose of knowledge discovery. This chapter introduces theoretical aspects of interpretability on rule based systems. In particular, some impact factors are identified and how these factors have an impact on interpretability is also analyzed. In addition, some criteria for evaluation on interpretability are also listed.

7.1 Learning Strategy

As mentioned earlier, different machine learning algorithms usually involve different strategies of learning. This would usually result in differences in two aspects namely, transparency and model complexity.

In terms of transparency, a rule learning method aims to generate a rule set typically in the form of either a decision tree or if-then rules. As mentioned in Chap. 1, rule based knowledge representation is able to explain the reason explicitly with regard to providing an output and, thus, it is well transparent. This is a significant advantage compared with some other popular machine learning methods such as neural networks and k nearest neighbor. Neural network learning aims to construct a network topology that consists of a number of layers and that has a number of nodes, each of which represents a perceptron. As a neural network is working in a black box manner with regard to providing an output, the transparency is poor, i.e. people cannot see in an explicit way the reason why the output is given. On the basis of the above description, neural networks have been judged poorly interpretable in [1]. K nearest neighbor involves lazy learning. In other words, the learning algorithm does not aim to learn in depth to gain some pattern from data but just to make as many correct predictions as possible. In the training stage, there is no actual learning but just some data loaded into computer memory. In this sense, there is no model built in the training stage so there is nothing to be visible for people to gain some useful patterns.

In terms of model complexity, as mentioned in Chap. 1, rule learning methods can be divided into two categories namely, 'divide and conquer' and 'separate and conquer', due to the difference in their strategy of rule generation. As mentioned in

H. Liu et al., *Rule Based Systems for Big Data*,
Studies in Big Data 13, DOI 10.1007/978-3-319-23696-4_7

[2], the latter approach usually generates fewer and more general rules than the former approach. The above phenomenon is due mainly to the strategy of rule learning. As mentioned in Chap. 3, the rule set generated by TDIDT needs to have at least one common attribute to be in the form of decision trees. The same also applies to each of the subtrees of a decision tree, which requires to have at least one common attribute represented as the root of the subtree. Due to this requirement, TDIDT is likely to generate a large number of complex rules with many redundant terms similarly to the replicated subtree problem illustrated in Chap. 3 and thus results in a model with high complexity. On the other hand, as mentioned above, k nearest neighbor does not build a model in the training stage. From this point of view, the model complexity is 0 as there is no model built.

On the basis of the above description relating to transparency and complexity, the strategies of learning involved in learning algorithms are an impact factor that affects interpretability.

7.2 Data Size

As mentioned in Sect. 7.1, different learning algorithms involve different strategies of learning and thus generate models with different level of complexity. In this sense, when the same data set is used, different learning algorithms would usually lead to different model complexity. However, for the same algorithm, data in different sizes would also usually result in the generation of models with different levels of complexity. The rest of this subsection justifies the potential correlation between data size and model complexity using rule based methods as examples.

As mentioned earlier, rule learning methods involve the generation of rule sets. The complexity of a rule set is determined by the total number of rule terms, which is dependent upon the number of rules and the average number of terms per rule. However, the total number of rule terms is also affected by the data size in terms of both dimensionality (number of attributes) and sample size (number of instances). For example, a data set has n attributes, each of which has t values, and its sample contains m instances and covers all possible values for each of the attributes. In this example, the model complexity would be equal to $\Sigma\, t^i$, while $i = 0, 1, 2, \ldots, n$, but no greater than $m \times n$ in the worst case. This indicates that a rule set consists of a default rule, which is also referred to as 'else' rule, and t^i rules, each of which has i terms, for $i = 0, 1, 2, \ldots, n$ respectively. However, each rule usually covers more than one instance and the rule set is expected to cover all instances. Therefore, the number of rules from a rule set is usually less than the number of instances from a data set. As also justified above, each rule would have up to n (the number of attributes) terms due to the requirement that each attribute can only appear once comprising one of its possible values in any of the rules.

On the basis of above description, the complexity of a rule set is up to the product of dimensionality and sample size of a data set.

7.3 Model Representation

As mentioned in Chap. 5, different types of machine learning algorithms may generate models represented in different forms. For example, as mentioned in Chap. 1, the 'divide and conquer' approach generates a rule set in the form of a decision tree whereas the 'separate and conquer' approach would generate if-then rules represented in a linear list. In addition, neural network learning algorithm would generate a multi-layer network with a number of interconnected nodes, each of which represents a perceptron. As described in Sect. 7.1, models generated by rule based learning methods are in white box and thus well transparent whereas models constructed by neural network learning methods are in black box and thus poorly transparent. As justified in Sect. 7.1, the level of transparency can affect the level of interpretability. However, models that demonstrate the same level of transparency may also have different level of interpretability due to their differences in terms of representation. The rest of this subsection justifies why and how the nature of model representation can affect the level of interpretability of rule based systems.

As argued in Chap. 5, decision trees suffer from the replicated subtree problem and thus are often difficult for people to read and understand to gain knowledge. In contrast to decision trees, linear lists do not have the constraint that all rules must have common attributes and thus reduces the presence of redundant terms in a rule set. However, redundancy may still arise with this representation, as the same attribute may repetitively appear in different rules as illustrated by the example below:

Rule 1: If x = 0 and y = 0 Then class = 0;
Rule 2: If x = 0 and y = 1 Then class = 1;
Rule 3: If x = 1 and y = 0 Then class = 1;
Rule 2: If x = 1 and y = 1 Then class = 0;

When a training set is large, there would be a large number of complex rules generated. In this case, the presence of redundancy would make the rule set (represented in a linear list) become very cumbersome and difficult to interpret for knowledge usage. In other words, a large number of complex rules represented in this way is quite like a large number of long paragraphs in an article that would be very difficult for people to read and understand. Instead, people prefer to look at diagrams to gain information. In this sense, graphical representation of rules would be expected to improve the interpretability of a model. More details about the improvement will be introduced in Chap. 8.

7.4 Human Characteristics

As mentioned in Chap. 1, different people may have different level of expertise and preferences and thus different level of cognitive capability to understand the knowledge extracted from a particular rule based system. The rest of this subsection justifies why and how human expertise and characteristics may affect the interpretability of rule based systems.

In terms of expertise, due to the fact that an expert system is used to act as a domain expert to extract knowledge or make predictions, people need to have the relevant expertise in order to be able to understand the context. From this point of view, the exactly same model may demonstrate different level of interpretability for different people due to their different level of expertise in this domain.

In terms of preferences, due to the fact that different people may have different preferences with respect to the way of reading, the exactly same model may also demonstrate different level of interpretability for different people due to their different preferences. From this point of view, human characteristics are also an impact factor that may affect the interpretability of model.

As mentioned in Sect. 7.3, model representation can affect the interpretability with respect to level of redundancy. In other words, the same model can have different level of interpretability depending on its representation. However, due to the difference in expertise and preferences, a particular representation may be understandable to some people but not to others. For example, some people in nature science/engineering would prefer to read diagrams/mathematical formulas whereas others may dislike them. From this point of view, model representation, human expertise and characteristics may jointly determine the cognitive capability for people to read and understand the knowledge extracted from a model.

7.5 Discussion

On the basis of above description in this section, the list of identified impact factors have the causal relationship to the interpretability as illustrated in Fig. 7.1.

Figure 7.1 indicates that the evaluation on interpretability could be based some criteria, namely model transparency, model complexity, model redundancy and cognitive capability, due to their direct relationships to interpretability.

In terms of model transparency, as mentioned in Sect. 7.1, the evaluation is based on information visualization. In other words, in what percentage the information is visible or hidden to people. For example, a neural network learning method generates a model in black box, which means that the information in the model is mostly hidden to people and thus poorly transparent. In contrast, a rule based learning method generates a model in white box, which means the information in the model is totally visible to people and thus well transparent.

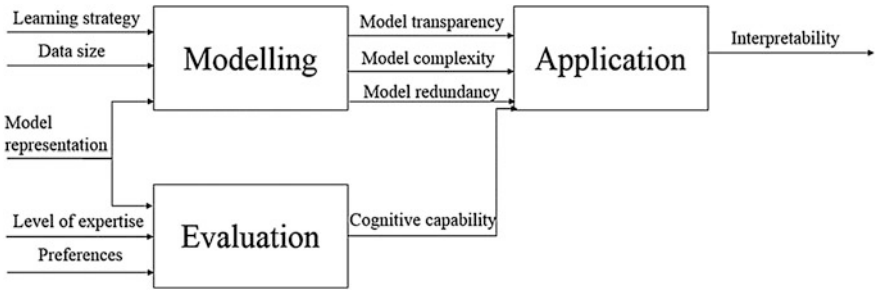

Fig. 7.1 Causal relationship between impact factors and interpretability

In terms of model complexity, the evaluation is subject to the type of learning algorithms to some extent. In particular, with regard to rule based methods, the model complexity could be measured by checking the total number of rule terms in a rule set, which is referred to as rule set complexity. For the rule set given below, the complexity would be 8.

Rule 1: If x = 0 and y = 0 Then class = 1;
Rule 2: If x = 0 and y = 1 Then class = 0;
Rule 3: If x = 1 and y = 0 Then class = 0;
Rule 2: If x = 1 and y = 1 Then class = 1;

In terms of model redundancy, the evaluation could be based on the extent of information duplication. In particular, a rule set may be represented in different forms namely, decision tree, linear list and rule based network. As mentioned in Sect. 7.3, the first two representations both may include duplicated information. For decision tree, the replicated subtree problem is a typical example to indicate that redundancy is a principal problem that arises with the representation. As can be seen from Fig. 3.2 in Chap. 3, there are four identical subtrees. For a linear list, as can be seen from the rule set given earlier in this section, all of the four rules have two common attributes, namely 'x' and 'y', which are repeated. The authors have developed two types of network topologies in order to reduce the redundancy as introduced in Chap. 5. In particular, one is attribute-value-oriented and the other one is attribute oriented. More details on the improvements have been described in Chap. 5.

In terms of cognitive capability, the evaluation would be based on empirical analysis following machine learning approaches. This is in order to analyze to what extent the model representation is understandable to particular people. In detail, this could be designed as a classification task to predict the cognitive capability in qualitative aspects or as a regression task to predict in quantitative aspects. Briefly speaking, the analysis could be done by collecting the data records on expertise and preferences from previous people who have high similarities to the current people and then taking the majority voting (if designed as a classification task) or

averaging (if designed as a regression task) with respect to the cognitive capability. The above task can be done effectively using k nearest neighbor algorithm.

References

1. Stahl, F., Jordanov, I.: An overview on the use of neural networks for data mining. WIREs Data Min. Knowl. Disc. **3**(2), 193–208 (2012)
2. Furnkranz, J.: Separate-and-conquer rule learning. Artif. Intell. Rev. **13**, 3–54 (1999)

Chapter 8
Case Studies

This chapter introduces three case studies of big data. In particular, the methods and techniques introduced in Chaps. 3, 4, 5 and 6 are evaluated through theoretical analysis and empirical validation using large data sets in terms of accuracy, efficiency and interpretability.

8.1 Overview of Big Data

As stated in [1], "Big Data is a popular term used to describe the exponential growth and availability of data, both structured and unstructured. And Big Data may be as important to business—and society—as the Internet has become." The above statement is due to the reasons identified in [1] as follows:

- More data may lead to more accurate analysis.
- More accurate analysis may lead to more confident decision making.
- Better decisions can mean greater operational efficiencies, cost reductions and reduced risk.

IBM defines in [2] that Big Data is characterized by four Vs:

- **Volume**—terabytes, petabytes, or more
- **Velocity**—data in motion or streaming data
- **Variety**—structured and unstructured data of all types—text, sensor data, audio, video, click streams, log files and more
- **Veracity**—the degree to which data can be trusted

As introduced in [3], big data and machine learning have been treated as connected activities. In particular, the relationship between big data and machine learning is very similar to that between life experience and human learning. In this context, people learn from their past experiences to deal with unfamiliar matters. Similarly, machines learn from big data to resolve newly challenging issues.

In the context of machine learning, learning algorithms are typically evaluated in terms of accuracy, efficiency and interpretability. These three dimensions can be strongly related to veracity, volume and variety respectively.

H. Liu et al., *Rule Based Systems for Big Data*,
Studies in Big Data 13, DOI 10.1007/978-3-319-23696-4_8

Veracity reflects the degree to which data can be trusted as mentioned above. In practice, the degree of trust is strongly dependent on the information/knowledge discovered from the data. This is because data usually needs to be transformed to information/knowledge prior to its actual use. Therefore, evaluation of the degree of trust for particular data can be done through estimation of the accuracy of the models built on the data.

Volume reflects the size of data. In machine learning and similar areas, the data size can be measured by the product of data dimensionality and sample size, i.e. the higher the data dimensionality/sample size the larger the data. Therefore, evaluation of the volume for particular data can be done through checking the data dimensionality and its sample size.

Variety reflects the format of data. In machine learning, the data format can be related to data types and presentation. In particular, data types include nominal, ordinary, string, Boolean, integer and real etc. More details on data types can be seen in [4]. In practice, data types are simply divided into two categories: discrete and continuous, in machine learning tasks. On the other hand, data can be represented in different forms such as text, graph and tables. This strongly relates to model representation since model is a form of information/knowledge that is transformed from data and can be represented in different forms.

Both data types and representation can impact on interpretability due to human characteristics. In particular, with regard to data types, people in some areas such as engineering and mathematics mostly deal with quantitative data so they normally prefer to see data with continuous values that reflect quantitative aspects. With regard to data representation, these people would prefer to see data in the form of diagrams or mathematical notations. In contrast, people in other areas such as humanities and social science mostly deal with qualitative data so they normally prefer to see data with discrete values that reflect qualitative aspects. With regard to data representation, these people would prefer to see data in the form of text.

8.2 Impact on Machine Learning

As mentioned in Sect. 8.3, machine learning algorithms are usually evaluated against accuracy, efficiency and interpretability. The presence of big data has deeply affected machine learning tasks in the three aspects mentioned above.

In terms of accuracy, overfitting of training data can be significantly reduced in general as the size of data is greatly increased. There is an evidence reported in [5] that learning from large training sets can significantly improve the performance in predictive modelling. The evidence is illustrated in Fig. 8.1, which was provided with an illustration by Banko and Brill in [6] that the complex problem of learning on automated word disambiguation would keep improving after the size of training data is beyond billion in words. The improvement in learning performance is due to the fact that the increase in data size can usually improve the completeness of the pattern covered. In other words, small data may be supposed to cover only a small

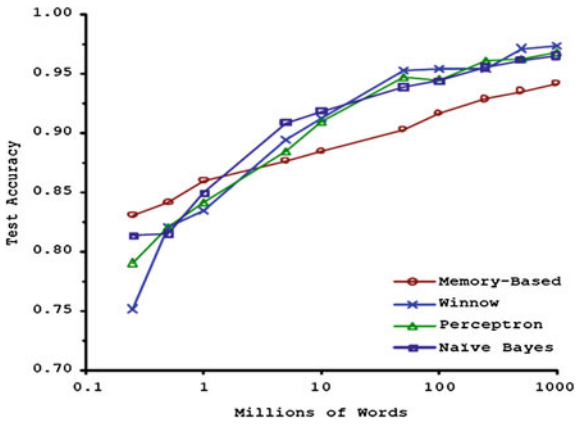

Fig. 8.1 Improvement of words disambiguation by learning from big data [6]

part of pattern in a hypothesis space. Therefore, overfitting of training data is likely to result in the case that a learning algorithm may build a model that performs greatly on training data but poorly on test data. This case occurs especially while the training data covers the pattern that is very different from that in test data. When the size of data is increased, the training data is likely to cover the pattern that is more similar to that exists in test data.

On the other hand, the increase in data size may also increase the chance to have noise and coincidental pattern present in the data. This is due to the fact that the biased improvement in the quantity of data may result in the loss of quality. In addition, large training data may cover some patterns which occur in very low frequencies. This could mean that the pattern covered by the training data is purely coincidental rather than scientifically confident.

The above issues regarding accuracy can be solved through scaling up algorithms or scaling down data. As introduced in Chap. 1, the former way is to reduce the bias on algorithms side. In particular, the algorithms can be designed to be more robust to noise and avoid being confused by coincidental patterns. In the context of rule learning, the reduction of bias can be achieved through direct advancement of rule generation methods or employment of rule simplification algorithms. The latter way is to reduce the variance on data side. In particular, data can be pre-processed through removal of irrelevant attributes by feature selection techniques or merge of redundant attributes by feature extraction techniques. In addition, data can also be resampled by selecting only those instances that are more representative.

In terms of efficiency, the increase in the size of data usually increases the computational costs in both training and testing stages. In the training stage, it may take much longer to build a predictive model by learning from big data. In the testing stage, the model is likely to be built to have a high level of complexity, which significantly slows down the process of predicting on unseen instances. In particular to rule based systems, big data may result in the generation of large number of complex rules.

As pointed out in [5], processing of big data needs decomposition, parallelism, modularity and recurrence. In this case, some machine learning algorithms, which are inflexible and work in black box manners, would be failed in dealing with big data. This case would immediately happen to those algorithms that are quadratically complex (O (n^2)), when encountering data with millions of points (instances).

The above issues regarding efficiency can also be solved through scaling up algorithms or scaling down data. In the former way, the algorithms can be designed to have a low computational complexity in training stage and thus much less affected by the increase in the size of training data. In addition, the improvement of efficiency can also be achieved through the employment of rule simplification methods as some of such methods can stop the process of rule learning earlier. In the latter way, the data size can be reduced through dimensionality reduction and data sampling. This not only reduces the computational costs in the training stage but also results in the generation of simpler models and thus speeds up the process of predicting on unseen instances in the testing stage.

In terms of interpretability, the increase in the size of data usually decreases the interpretability. As introduced in Chap. 7, interpretability can be affected by the size of training data in terms of model complexity. In the context of rule based systems, as mentioned above, big data may result in the generation of large number of complex rules, which would make it difficult for people to read and understand.

The above issues regarding interpretability can also be solved through scaling up algorithms or scaling down data. In the former way, the algorithms can be designed to be robust to noise and irrelevant/redundant attributes. In particular, the presence of noise and irrelevant/redundant attributes cannot make these algorithms learn irrelevant patterns. In the context of rule learning, rule generation algorithms may decide to skip some attributes/attribute-value pairs for generation of decision trees or if-then rules due to irrelevance. In addition, the employment of rule simplification methods also helps improve the interpretability since the employment usually results in the generation of smaller number of simpler rules. In the latter way, the data size is reduced through dimensionality reduction and data sampling as mentioned above. In particular, as already discussed in Chap. 7, the reduction of dimensionality decreases the maximum length (the maximum number of rule terms) of each single rule. The data sampling also reduces the maximum number of rules. In this way, the interpretability can be improved if the dimensionality reduction and data sampling are effectively done.

The rest of this chapter introduces three case studies using some relatively large data sets to show the significance of the approaches introduced in Chaps. 3, 4 and 6 in the processing and analysis of big data. In particular, the case studies will address the three Vs: veracity, volume and variety, through the analysis of accuracy, efficiency and interpretability.

8.3 Case Study I-Rule Generation

This case study is designed to test the significance of rule generation methods in big data processing. In particular, 15 data sets, which are retrieved from the medical repository [7], are used for the experimental study. The characteristics of these data sets are described in Table 8.1.

It can be seen from Table 8.1 that all the data sets chosen have high dimensionality. In particular, 14 out of 15 data sets have many thousands of attributes, including 9 data sets with the dimensionality higher than 12,000. In addition, all of the data sets contain continuous attributes only and involve two-class classification problems except for *MLL_Leukemia* data set.

In this experimental study, Prism and IEBRG are used as the rule generation methods. In general, cross-validation [4] is used to test the above data sets in terms of accuracy. However, some of the data sets are provided with additional test sets. In this case, the training data is used to build the rule sets that are then tested using the additional test data. In particular, the following data sets are provided with additional test data: ALL-AML, LungCancer-Harvard2, MLL_Leukemia, prostatetumorVSNormal, BCR-ABL, E2A-PBX1, Hyperdip50, MLL, T-ALL and TEL-AML1 (Table 8.2).

In addition, the accuracies achieved by the above two algorithms are also compared with that achieved by the random classifier, which predicts classification by random guess. The corresponding accuracy depends on the number of classifications and distribution of these classifications. For example, if the objective function is a two class classification problem and the distribution is 50:50, then the accuracy performed by random guess would be 50 %. Otherwise, the accuracy must

Table 8.1 Medical data sets

Name	Attribute types	#Attributes	#Instances	#Classes
ALL-AML	Continuous	7130	72	2
colonTumor	Continuous	2001	62	2
DLBCLTumor	Continuous	7130	77	2
DLBCL-Stanford	Continuous	4027	47	2
LungCancer-Harvard2	Continuous	12,534	32	2
lung-Michigan	Continuous	7130	96	2
MLL_Leukemia	Continuous	12,583	72	3
prostatetumorVSNormal	Continuous	12,601	136	2
BCR-ABL	Continuous	12,559	327	2
E2A-PBX1	Continuous	12,559	327	2
Hyperdip50	Continuous	12,559	327	2
MLL	Continuous	12,559	327	2
T-ALL	Continuous	12,559	327	2
TEL-AML1	Continuous	12,559	327	2
pos_neg_100	Continuous	928	13,375	2

Table 8.2 Accuracy (case study I)

Dataset	Prism (%)	IEBRG (%)	Random classifier (%)
ALL-AML	**91**	**91**	47
colonTumor	**82**	73	52
DLBCLTumor	76	**79**	61
DLBCL-Stanford	80	**92**	49
LungCancer-Harvard2	80	**99**	50
lung-Michigan	**94**	**94**	80
MLL_Leukemia	**66**	60	33
prostate_tumorVSNormal	73	**74**	59
BCR-ABL	96	**98**	89
E2A-PBX1	**100**	98	84
Hyperdip50	93	**96**	68
MLL	95	**99**	89
T-ALL	**100**	**100**	77
TEL-AML1	**95**	88	63
pos_neg_100	61	**73**	50

be higher than 50 % in all other cases. This setup of experimental study is in order to indicate the lower bound of accuracy to judge if an algorithm really works on a particular data set.

On the other hand, efficiency in the training stage for the two algorithms mentioned above is tested using the criteria that include summative time complexity and runtime as illustrated in Table 8.3.

Besides, efficiency in test stage and interpretability of a rule set are tested by checking the number of rules and average number of terms per rule as illustrated in Table 8.4. With regard to the experimental setup on the efficiency testing, the whole data set is used to train the rule set and then the same data set is used for testing, where there is no additional test data supplied. In particular, the following data sets are actually tested using the supplied training data in terms of efficiency: ALL-AML, LungCancer-Harvard2, MLL_Leukemia, prostatetumorVSNormal, BCR-ABL, E2A-PBX1, Hyperdip50, MLL, T-ALL and TEL-AML1. For the rest of the data sets, the testing is done on the whole data set since there is no additional test data provided.

The results shown in Table 8.2 indicate that both Prism and IEBRG outperform the random classifier in terms of classification accuracy, which means that both of the two methods really work on the chosen data sets. In the comparison between Prism and IEBRG, the results show that IEBRG outperforms Prism in 8 out of 15 cases. In addition, there are three cases that IEBRG performs the same as Prism. For the rest of the cases, IEBRG still performs a similar level of accuracy to Prism.

The results shown in Table 8.3 indicate that IEBRG is computationally more efficient than Prism in the training stage in all of the above cases. In particular, the

Table 8.3 Summative time complexity and runtime in milliseconds (Case Study I)

Dataset	Prism		IEBRG	
	Complexity	Runtime	Complexity	Runtime
ALL-AML	56,025,578	64,233	**30,814,476**	**2765**
colonTumor	49,158,150	43,156	**25,302,352**	**1469**
DLBCLTumor	243,886,464	181,326	**121,819,458**	**5203**
DLBCL-Stanford	49,369,213	55,024	**25,828,555**	**1750**
LungCancerHavard2	72,102,050	97,247	**784,113,515**	**19,829**
Lung-Michigan	349,954,962	123,101	**190,617,697**	**5406**
MLL_Leukemia	339,788,577	248,095	**128,112,977**	**9672**
prostatetumorVSNormal	486,178,307	323,069	**245,281,839**	**12,688**
BCR-ABL	1,617,079,658	196,057	**852,781,399**	**14,797**
E2A-PBX1	1,515,159,180	128,348	**885,092,190**	**11,970**
Hyperdip50	1,606,220,569	297,173	**808,234,024**	**18,064**
MLL	1,597,425,152	256,229	**252,699,246**	**6688**
T-ALL	1,515,159,180	158,918	**873,995,424**	**12,891**
TEL-AML1	807,152,929	43,643	**817,671,423**	**15,532**
pos_neg_100	827,367,228	21,936	**44,855,292**	**8422**

Table 8.4 Number of rules and average number of terms (case study I)

Dataset	Prism		IEBRG	
	Count(rules)	Ave(terms)	Count(rules)	Ave(terms)
ALL-AML	**2**	1.0	**2**	1.0
colonTumor	5	1.0	**4**	1.0
DLBCLTumor	4	1.0	**3**	1.0
DLBCL-Stanford	4	1.0	**3**	1.0
LungCancerHavard2	**2**	1.0	**2**	1.0
Lung-Michigan	**2**	1.0	**2**	1.0
MLL_Leukemia	4	1.0	**3**	1.0
ProatetmorVSNomal	6	1.0	**4**	1.0
BCR-ABL	4	1.0	**3**	1.0
E2A-PBX1	2	1.0	**2**	1.0
Hyperdip50	6	1.0	**4**	1.0
MLL	5	1.0	**2**	1.0
T-ALL	2	1.0	**2**	1.0
TEL-AML1	4	1.0	**3**	1.0
pos_neg_100	22	1.0	**12**	1.0

comparison between the two algorithms is in terms of both complexity analysis and runtime. The time complexity analysis for the two algorithms is taken based on illustration as follows:

Time complexity analysis for IEBRG regarding the generation of each rule term:
Suppose a data set has i instances and a attributes so the size is $i \times a$ as well as v attribute-value pairs and c classifications
Step 1: create a frequency table
Time complexity: $i \times v + i \times a \times c$
Step 2: calculate conditional entropy for each attribute-value pair
Time complexity: $v \times c$
Step 3: rank the conditional entropy for all attribute values
Time complexity: $v + a$
Step 4: split the dataset by deleting the instances that don't comprise the attribute-value pair
Time complexity: i
Therefore, the time complexity is: $O\ (i \times v + i \times a \times c + v \times c + v + a + i)$, for the generation of each rule term while the input size (n) is the total number of rule terms.

Time complexity analysis for Prism regarding the generation of each rule term:
Suppose a data set has i instances and a attributes so the size is $m \times n$ as well as v attribute-value pairs and c classifications
Step 1: create a frequency table
Time complexity: $i \times v + i \times a$
Step 2: calculate posterior probability of a target class given an attribute-value pair as the condition
Time complexity: v
Step 3: rank the posterior probability for all attribute values
Time complexity: $v + a$
Step 4: split the dataset by deleting the instances that don't comprise the attribute-value pair
Time complexity: i
Therefore, the time complexity is: $\mathrm{O}\ (i \times v + i \times a + v + v + a + i)$, for the generation of each rule term while the input size (n) is the total number of rule terms.

The results shown in Table 8.5 indicate that both Prism and IEBRG generate simple rule sets on all of the large data sets. In particular, on each of the data sets, both algorithms generate a small number of first order rules, which are not just efficient in predicting unseen instances, but also interpretable to people for knowledge representation. In addition, IEBRG generates a fewer number of rules than Prism in 12 out of 15 cases and the same number of rules as Prism in the other three cases.

Table 8.5 Data sets

Name	Attribute type	#Attributes	#Instances	#Classes	References
cmc	Discrete, continuous	10	1473	3	[10]
Vote	Discrete	16	435	2	[11]
kr-vs-kp	Discrete	36	3196	2	[12–14]
Ecoli	Continuous	23	336	2	[15]
Anneal.ORIG	Discrete, continuous	39	898	6	
Audiology	Discrete	69	226	24	[16]
Car	Discrete	6	1728	4	[17, 18]
Optdigits	Continuous	64	5620	10	[19, 20]
Glass	Continuous	10	214	7	[21]
Lympth	Discrete, continuous	19	148	4	[22–24]
Yeast	Discrete	8	1484	2	[25–27]
Shuttle	Continuous	9	58,000	7	
Asbestos	Discrete, continuous	4	83	3	[28]
Irish	Discrete, continuous	5	500	2	[29–31]
Breast-cancer	Discrete	9	286	2	[22–24]

8.4 Case Study II-Rule Simplification

This case study is designed to test the significance of rule simplification methods with respect to the impact on accuracy, efficiency and interpretability in the context of big data processing. In particular, 15 data sets, which are retrieved from the UCI repository [8] and statLib repository [9]. The characteristics of these data sets are described in Table 8.5.

It can be seen from Table 8.5 that all the data sets chosen are relatively small in comparison with those ones used in Case Study I. In particular, all of the data sets except for *shuttle* contain hundreds or thousands of instances. However, as can be seen from Table 8.6, Prism is chosen as the rule generation method for testing the effectiveness of rule simplification. Due to the nature of the Prism algorithm, the actual sample size of the data being processed in training stage is n times the size of the original data, where n is the number of classes in the data set. For example, the actual sample size of the *optdigits* data processed by Prism is 56,200 (5620×10) since this data set has 10 classes. On the basis of above discussion, the actual processed sample would be considered as big data.

In this experimental study, Prism is used as the rule generation method as mentioned above and Jmid-pruning is employed as the rule simplification method. Therefore, the two methods involved in the testing are referred to as Prism without pruning and Prism with Jmid-pruning. In general, cross-validation [4] is used to test the above data sets in terms of accuracy. However, some of the data sets are provided with additional test sets. In this case, the training data is used to build the rule sets that are then tested using the additional test data. In particular, the two data sets, namely *yeast* and *shuttle*, are provided with additional test data.

Table 8.6 Accuracy (Case study II)

Dataset	Prism without pruning (%)	Prism with Jmid-pruning (%)	Random classifier (%)
cmc	38	**40**	35
Vote	91	**95**	52
kr-vs-kp	**64**	54	50
Ecoli	60	**62**	27
Anneal.ORIG	**80**	78	60
Audiology	46	**47**	13
Car	70	**75**	33
Optdigits	39	**51**	10
Glass	**54**	**54**	24
Lymph	**76**	**76**	47
Yeast	**37**	36	21
Shuttle	84	**85**	65
Analcatdata_asbestos	58	**60**	43
Irish	**99**	**99**	50
Breast-cancer	68	**69**	58

In addition, the accuracies achieved by the above two methods are also compared with that achieved by the random classifier in order to indicate the lower bound of accuracy to judge if an algorithm really works on a particular data set as already mentioned in Case Study I.

On the other hand, efficiency in the training stage for the two methods as mentioned above is tested using the runtime in milliseconds as the criteria as illustrated in Table 8.3.

Besides, efficiency in test stage and interpretability of a rule set are tested by checking the number of rules and average number of terms per rule as illustrated in Table 8.4. With regard to the experimental setup on the testing of efficiency and interpretability, the whole data set is used to train the rule set and then the same data set is used for testing, while there is no additional test data supplied. In particular, the two data sets, namely *yeast* and *shuttle*, are actually tested using the supplied training data in terms of efficiency. For the rest of the data sets, the testing is done on the whole data set since there is no additional test data provided.

The results shown in Table 8.6 indicate that Prism outperforms the random classifier in terms of classification accuracy no matter if pruning is taken. This means that the Prism algorithm really works on the chosen data sets in both cases (with and without pruning). On the other hand, the results show that Jmid-pruning successfully helps improve the accuracy in 9 out of 15 cases. In addition, in five out of the other 6 cases, Jmid-pruning either makes no difference or decreases the accuracy slightly in comparison with Prism without pruning. On the basis of above discussion, Jmid-pruning is able to help improve accuracy for Prism when pruning

Table 8.7 Runtime in milliseconds

Dataset	Prism without pruning	Prism with Jmid-pruning
cmc	**3611**	4042
Vote	638	**632**
kr-vs-kp	**4130**	5721
Ecoli	**794**	1174
Anneal.ORIG	**9615**	13,991
Audiology	36,509	**35,056**
Car	**886**	1009
Optdigits	**165,525**	294,394
Glass	**1509**	1531
Lymph	**62**	63
Yeast	3508	**1465**
Shuttle	270,547	**260,799**
Analcatdata_asbestos	**79**	91
Irish	76	**65**
Breast-cancer	**16**	31

is required. Even when Prism does not need help from pruning methods at all, the use of Jmid-pruning only has a slight effect on the classification accuracy.

The results shown in Table 8.7 indicate that the use of Jmid-pruning has only a slight impact on computational costs and may also even help improve the efficiency in training in some cases. In general, for the generation of each single rule term, the computational complexity is normally increased since J-measure and Jmax need to be calculated in addition to the posterior probability of a target class given a particular attribute-value pair. In this context, if the Jmid-pruning can stop rule generation very early, then the number of iterations for generation of single rule terms is obviously reduced, which results in the reduction of runtime. However, if the Jmid-pruning stop the rule generation very late, then the number of iterations is not reduced a lot. In this case, the runtime may be even longer than when no pruning is taken.

The results shown in Table 8.8 indicate that Jmid-pruning helps reduce the complexity of the rule set generated by Prism in most cases. In particular, the complexity of a rule set can be measured by checking the total number of rule terms generated for the rule set, which is approximately equal to the product of the number of rules and average number of rule terms per rule. In this context, Jmid-pruning successfully help reduces the complexity in 11 out of 15 cases. The reduction of complexity for a rule set not only improves the computational efficiency in predicting unseen instances for predictive modelling, but also makes the rule set more interpretable to people for knowledge discovery and representation.

On the basis of the above discussion, Jmid-pruning is effective in improving the quality of a rule set generated by rule learning algorithms in terms of accuracy, efficiency and interpretability.

Table 8.8 Number of rules and average number of terms (case study II)

Dataset	Prism without pruning		Prism with Jmid-pruning	
	Count(rules)	Ave(terms)	Count(rules)	Ave(terms)
cmc	36	4.67	**25**	**4.48**
Vote	25	6.28	**15**	**5.13**
kr-vs-kp	63	5.84	**21**	**5.52**
Ecoli	24	**1.88**	**17**	1.94
Anneal.ORIG	16	**1.56**	**12**	3.67
Audiology	48	3.60	**38**	**2.79**
Car	**2**	**1.0**	3	2.0
Optdigits	431	7.46	**197**	**6.53**
Glass	26	**2.85**	**24**	3.29
Lymph	**10**	1.3	**10**	**1.11**
Yeast	37	1.68	**20**	**1.5**
Shuttle	30	3.87	**12**	**1.0**
Analcatdataasbestos	**5**	1.6	**5**	**1.4**
Irish	**10**	1.5	11	**1.27**
Breast-cancer	**11**	**1.09**	**11**	1.0

8.5 Case Study III-Ensemble Learning

This case study is designed to test the significance of ensemble rule learning methods in improving the overall accuracy of classification in the context of big data processing. In particular, 10 data sets, which are retrieved from the UCI repository, are used for the experimental study. The characteristics of these data sets are described in Table 8.9.

It can be seen from Table 8.9 that all of the data sets chosen are not large. In particular, all of them have no more than 1000 instances. However, as introduced in Chap. 6, ensemble learning approaches are usually computationally more expensive in comparison with standard learning approaches, due to the nature of this kind of approaches. In this experimental study, the hybrid ensemble rule based classification framework introduced in Sect. 6.3 is adopted for processing the above data sets. This approach involves resampling of data, which means that the actual sample size of the data being processed in the training stage is n times the size of the original data, where n is the number of samples. In this study, the n is set to be 10 as this is the commonly used setting. In addition, as introduced in Chap. 6, the hybrid ensemble rule based classification framework also involves a combination of different rule learning algorithms for the generation of a rule set on each sample of training data. In this context, each sample is approximately processed m times, where m is the number of rule learning algorithms. In this study, Prism and IEBRG are employed as the rule learning algorithms for collaborative rule generation on each sample. On the basis of above discussion, the actual processed sample would be considered as big data.

Table 8.9 UCI data sets

Name	Attribute types	#Attributes	#Instances	#Classes	References
Credit-a	Discrete, continuous	15	690	2	[31, 32]
Credit-g	Discrete, continuous	20	1000	2	
Vote	Discrete	16	435	2	[11]
Hepatitis	Discrete, continuous	20	155	2	[22, 33]
Lung-cancer	Discrete	32	57	3	[34]
Lymph	Discrete, continuous	19	148	4	[22–24]
Breast-cancer	Discrete	9	286	2	[22–24]
Breast-w	Continuous	10	699	2	[35, 36]
Labor	Discrete, continuous	17	57	2	[37, 38]
Heart-h	Discrete, continuous	76	920	4	[39]

The experiments are conducted by splitting a data set into a training set and a test set in the ratio of 70:30. For each data set, the experiment is done 10 times and the average of the accuracies is taken for quantitative analysis. As mentioned earlier, ensemble learning approaches are usually computationally more expensive. Therefore, cross validation is not adopted in this study. In addition, the ensemble learning approaches are also compared with the random classifier in terms of classification accuracy. This is in order to indicate the lower bound of accuracy to judge if an approach really works on a particular data set as already mentioned in Case Study I. The results are presented in Table 8.10.

The results shown in Table 8.10 show that the three employed approaches outperform the random classifier in terms of classification accuracy in all the cases. This indicates that all of the approaches properly work on the chosen data set. On the other hand, the hybrid ensemble rule based classification framework outperforms random forests and CCRDR in 8 out of 10 cases. On 'breast-w' and 'labor' data sets, the newly developed framework performs a bit worse than random forests

Table 8.10 Accuracy (Case Study III) [40]

Data set	Random forests (%)	CCRDR (%)	Hybrid (%)	Random classifier (%)
Credit-a	85	70	**87**	50
Credit-g	72	71	**74**	58
Vote	97	93	**98**	52
Hepatitis	85	84	**92**	66
Lung-cancer	70	86	**93**	56
Lymph	86	70	**90**	47
Breast-cancer	65	78	**81**	58
Breast-w	**97**	85	91	55
Labor	88	**90**	88	54
Heart-h	83	79	**85**	54

and CCRDR. This indicates the necessity pointed out in Chap. 6 to take both scaling up algorithms and scaling down data in order to achieve comprehensive improvement of overall classification accuracy.

On the basis of the above discussion, the hybrid ensemble rule based classification approach is effective in reduction of both bias and variance towards the improvement of overall accuracy of classification. In particular, the reduction of bias prevents that noise and coincidental patterns are learned from big data whereas the reduction of variance decreases the chance that noise and coincidental patterns are present in the training sample.

References

1. What is Big Data.: SAS Institute Inc, (Online). Available: http://www.sas.com/big-data/. (Accessed 17 May 2015)
2. Master Data Management for Big Data.: IBM, (Online). Available: http://www-01.ibm.com/software/data/infosphere/mdm-big-data/. (Accessed 17 May 2015)
3. Levine, P.: Machine Learning + Big Data. WorldPress, (Online). Available: http://a16z.com/2015/01/22/machine-learning-big-data/. (Accessed 15 May 2015)
4. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Pearson Education Inc, New Jersey (2006)
5. Machine Learning on Big Data.: EBTIC, 19 August 2014. (Online). Available: http://www.ebtic.org/pages/ebtic-view/ebtic-view-details/machine-learning-on-big-data-d/687. (Accessed 15 May 2015)
6. Banko, M., Brill, E.: Scaling to very very large corpora for natural language disambiguation. In: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics (2001)
7. Li, J., Liu, H.: Kent Ridge Bio-medical Dataset, I2R Data Mining Department, 2003. (Online). Available: http://datam.i2r.a-star.edu.sg/datasets/krbd/. (Accessed 18 May 2015)
8. Lichman, M.: UCI Machine Learning Repository. University of California, School of Information and Computer Science, 2013. (Online). Available: http://archive.ics.uci.edu/ml. (Accessed 12 May 2015)
9. Vlachos, P.: StatLib—Datasets Archive. Carnegie Mellon University, 19 July 2005 . (Online). Available: http://lib.stat.cmu.edu/datasets/. (Accessed 18 May 2015)
10. Lim, T.-S.: Loh, W.-Y., Shih, Y.-S.: A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. Mach. Learn. **40**:203–229 (2000)
11. Schlimmer, J.: Concept Acquisition Through Representational Adjustment. Irvine, CA (1987)
12. Shapiro, A.D.: Structured Induction in Expert Systems. Turing Institute Press in association with, Addison-Wesley (1987)
13. Muggleton, S.: Structuring knowledge by asking questions. In: Bratko, I., Lavrac, N. (eds.) Progress in Machine Learning, pp. 218–229. Sigma Press, Wilmslow (1987)
14. Holte, R.C., Acker, L., Porter, B.W.: Concept learning and the problem of small disjuncts. In: International Joint Conference on Artificial Intelligence, Detroit (1989)
15. Horto, P., Nakai, K.: A Probablistic classification system for predicting the cellular localization sites of proteins. Intell. Syst. Mol. Biol. St. Louis. **4**:109–115 (1996)
16. Bareiss, R.E., Porter, B.: Protos: an exemplar-based learning apprentice. In: The 4th International Workshop on Machine Learning, Irvine, CA (1987)

17. Bohanec, M., Rajkovic, V.: Knowledge acquisition and explanation for multi-attribute decision making. In: 8th International Workshop on Expert Systems and their Applications, Avignon (1988)
18. Zupan, B., Bohanec, M., Bratko, I., Demsar, J.: Machine learning by function decomposition. In: International Conference on Machine Learning, Nashville (1997)
19. Kaynak, C.: Methods of Combining Multiple Classifiers and Their Applications to Handwritten Digit Recognition (1995)
20. Alpaydin, E., Kaynak, C.: Cascaded classifiers. Kybernetika **34**, 369–374 (1998)
21. Evett, I.W., Spiehler, E.J.: Rule Induction in Forensic Science. Reading (1987)
22. Cestnik, G., Konenenko, I., Bratko, I.: Assistant-86: A knowledge-elicitation tool for sophisticated users. In: Bratko, I., Lavrac, N. (eds.) Progress in Machine Learning, pp. 31–45. Sigma Press, New York (1987)
23. Clark, P., Niblett, T.: Induction in noisy domains. In: Progress in Machine Learning, pp. 11–30. Sigma Press, UK (1987)
24. Michalski, R., Mozetic, I., Hong, J., Lavrac, N.: The multi-purpose incremental learning system AQ15 and its testing applications to three medical domains. In The Fifth National Conference on Artificial Intelligence, Philadelphia, PA (1986)
25. Nakai, K., Kanehisa, M.: Expert system for predicting protein localization sites in gram-negative bacteria. PROTEINS Struct. Funct. Genet. **11**, 95–110 (1991)
26. Nakai, K., Kanehisa, M.: A knowledge base for predicting protein localization sites in eukaryotic cells. Genomics **14**, 897–911 (1992)
27. Simonoff, J.S.: Analyzing Categorical Data. Springer, New York (2003)
28. Greaney, V., Kelleghan, T.: Equality of Opportunity in Irish. Educational Company, Dublin (1984)
29. Raftery, A.E., Hout, M.: Does Irish education approach the meritocratic ideal? A logistic analysis. Econ. Soc. Rev **16**, 115–140 (1985)
30. Raftery, A.E., Hout, M.: Maximally maintained inequality: expansion, reform and opportunity in Irish schools. Sociol. Educ. **66**, 41–62 (1993)
31. Quinlan, J.R.: C 4.5: Programs for Machine Learning, Morgan Kaufman, San Mateo (1993)
32. Ross, Q.: Simplifying decision trees. Int. J. Man-Mach. Stud. **51**:221–234 (27 December 1987)
33. Diaconis, P., Efron, B.: Computer-Intensive Methods in Statistics, vol. 248. Scientific American, Pasadena, CA (1983)
34. Hong, Z.Q., Yang, J.Y.: Optimal discriminant plane for a small number of samples and design method of classifier on the plane. Pattern Recogn. **24**(4), 317–324 (1991)
35. Wolberg, W.H., Mangasarian, O.L.: Multisurface method of pattern separation for medical diagnosis applied to breast cytology. In: The National Academy of Sciences, USA (1990)
36. Zhang, J.: Selecting typical instances in instance-based learning. In: The 9th International Conference on Machine Learning, Aberdeen, Scotland (1992)
37. Bergadano, F., Matwin, S., Michalski, R., Zhang, J.: Measuring quality of concept descriptions. In: The 3rd European Working Sessions on Learning, Glasgow (1988)
38. Bergadano, F., Matwin, S., Michalski, R., Zhang, J.: Representing and acquiring imprecise and context-dependent concepts in knowledge-based systems. In: International Symposium on Methodologies for Intelligent Systems, North Holland (1988)
39. Detrano, R., Janosi, A., Steinbrunn, W., Pfisterer, M., Schmid, J., Sandhu, S., Guppy, K., Lee, S., Froelicher, V.: International application of a new probability algorithm for the diagnosis of coronary artery disease. Am. J. Cardiol. **64**, 304–310 (1989)
40. Liu, H., Gegov, A., Cocea, M.: Hybrid ensemble learning approach for generation of classification rules. In: Internation Conference on Machine Learning and Cybernetics, Guangzhou (2015)

Chapter 9
Conclusion

This chapter summaries the contributions of this book in terms of theoretical significance, practical importance, methodological impact and philosophical aspects. This chapter also identifies and highlights further directions of this research area towards improvement of the research methodologies presented in this book.

9.1 Theoretical Significance

The focus of the book is to introduce a theoretical unified framework for construction of rule based classification systems. The framework includes three main operations namely, rule generation, rule simplification and rule representation as mentioned in Chap. 2. The experimental results shown in Chap. 8 prove that the incorporation of rule simplification is relevant and can generally make rule generation more effective and efficient. In addition, the theoretical results on rule representation has shown that different structure of rule set can lead to different level of efficiency in prediction stage. This indicates it is important to represent a rule set in a suitable structure.

This book also introduced an advanced framework of ensemble learning for construction of ensemble rule based classification systems in Chap. 6. In this framework, competitive learning is incorporated in the training stage in order to find better classifiers and ignore worse classifiers for the prediction stage. From theoretical point of view, this incorporation can improve the flexibility of the construction framework. This is because different learning algorithms may have different level of fitness to a particular data set due to its characteristics. In addition, a new strategy in determining the weight for weighted voting, which is based on precision for each individual classification, has been introduced in the book to increase the reliability of a classifier for making a particular classification. This is because of the fact that the overall accuracy cannot well represent the capability of a classifier in making predictions on instances of a particular classification, especially

H. Liu et al., *Rule Based Systems for Big Data*,
Studies in Big Data 13, DOI 10.1007/978-3-319-23696-4_9

on extremely unbalanced data sets. The empirical results shown in Chap. 8 indicate that the two above modifications to the ensemble learning framework can improve the overall classification accuracy.

The book also brings in a novel application of graph theory [1, 2] in Chap. 5 for the development of networked rule representation as well as a novel application of BigO notation [3] in Chap. 5 for the theoretical validation of the representation with respect to computational efficiency in the prediction stage. Both concepts are widely used in discrete mathematics [4] and complexity theory [5].

The theoretical framework for constructing rule based classification systems introduced in Chap. 2 can also be illustrated in the context of system theory in addition to machine learning. In other words, in the past research, rule based system is conceptually referred to as a special type of expert system but the construction of such systems mainly follows traditional engineering approach rather than machine learning approach. Although there is a special type of classification referred to as rule based classification in machine learning, it is usually in the context of rules or rule sets rather than system theory. In the book, the conceptual connection is made between rule based systems and machine learning as follows. In machine learning context, the objective of rule based classification is the induction of rule sets. Therefore, a rule based classifier is usually referred to as a rule set. In system theory context, the classifier can also be regarded as a rule based classification system. On the other hand, if the induction of classification rules is done through the ensemble learning approach, there would be multiple classifiers constructed as an ensemble learner. In system theory context, this can be seen as construction of an ensemble rule based classification system which has each single classifier as a subsystem of the ensemble rule based classification system. Therefore, the basis of above descriptions brings new concepts to system engineering [6, 7] with respect to methodological differences in comparison with existing methodologies [8, 9].

9.2 Practical Importance

The theoretical framework introduced in the book is generally domain independent in real applications because almost all domains usually follow similar approaches in problem solving in the machine learning context. This is a significant difference to knowledge based construction, which is generally domain dependent and needs to have expert knowledge acquired as the main task in the process of constructing a rule based system [10–13]. The framework can contribute to the development of expert systems for the purpose of knowledge discovery and predictive modelling such as medical applications as reported in [14–16].

In the aspect of knowledge discovery, rule based systems can be used by domain experts to find interesting, useful and previously unknown patterns such as causal relationships. This can help the experts further identify new research directions as

well as make necessary validations on their hypothesis by using real data. In classification, as mentioned in Chap. 2, one purpose of rule representation is to present the knowledge in different ways in accordance with specific commercial requirements. From this point of view, the networked representation can effectively reflect the importance of input attributes and provide a ranking of the attributes according to their importance. In practice, each input attribute can be seen as an impact factor to which a decision outcome is subject. And the ranking of attributes can help domain experts identify which ones are more important, less important or irrelevant to the decision outcome.

In the aspect of predictive modelling, rule based systems can be used to help with prediction problems such as recommendations, decision making and stock price prediction. In classification, it can help make categorical prediction in qualitative aspects on a single variable such as weather prediction, degree classification and faults classification. In this context, each classifier constructed in training stage actually plays the role in a decision maker to make a decision/prediction. If the ensemble learning approach is adopted, it means that multiple classifiers constructed in training stage play the role in a group of decision makers to achieve collaborative decision making. In addition, as part of construction framework for rule based classification systems, rule simplification can help speed up the process of modelling and rule representation can help make quicker decisions/predictions.

Besides, as mentioned in Chap. 2, ensemble learning could be done in parallel, which means that each single machine learner is constructed independently and that only their predictions are combined for final decision making. This indicates that the ensemble learning could be done by a parallel computer to improve the computational efficiency in both training and testing stages. In addition, each company or organization may have branches in different cities or countries so the databases for the companies or organizations are actually distributed over the world. As the existence of high performance cloud and mobile computing technologies, the ensemble learning framework can be easily transplanted into distributed or mobile computing environments such as multi-agent systems [17].

The rule based systems described in the book are based on deterministic logic. In other words, this kind of systems is usually referred to as deterministic rule based systems which make decisions under certainty in practice. However, rule based systems can also be constructed based on probabilistic logic or fuzzy logic in real applications for decision making under probabilistic or non-probabilistic uncertainty. In this context, the theoretical framework introduced in the book can also be extended for construction of probabilistic and fuzzy rule based systems when the two kinds of rule based systems are required in practice. In addition, the book also focuses on rule based systems for classification such as qualitative diagnostics. However, this kind of systems can also be used for regression and association in machine learning context so the theoretical framework can also be extended for construction of rule based regression/association systems for other practical purposes.

9.3 Methodological Impact

This book introduces some novel methods and techniques in rule based classification and ensemble learning in Chaps. 3, 4, 5 and 6 namely information entropy based rule generation (IEBRG), Jmid-pruning, rule based networks, collaborative and competitive decision rules and hybrid ensemble rule based classification.

IEBRG is developed by making modifications to the Prism algorithm in order to overcome the limitations of Prism as mentioned in Chap. 3. IEBRG represents a so-called 'from causes to effects approach' for rule generation which has been proven empirically more effective and efficient than the so-called 'from effects to causes approach' represented by Prism in Chap. 3. For example, in comparison with the latter approach, the former approach can generate more general and fewer rules than the latter approach and make the machine learning tasks more effective and efficient. In most cases, IEBRG outperforms Prism in both classification accuracy and computational efficiency on large data sets as shown by the experimental results in Chap. 8, Case study I.

Rule based network is a networked representation of rules or rule sets, which means a rule based system is represented in a networked structure. In this context, a special type of rule based systems can be referred to as rule based networks if the rule based systems are in the form of networks. In addition, in the context of complexity theory, it is proven theoretically in Chap. 5 that the networked rule representation is computationally more efficient than decision tree and linear list representations in term of time complexity for predicting unseen instances. In the context of knowledge representation, rule based network can also provide a better interpretability to reflect the relationships between inputs and outputs as well as the importance of input attributes as justified in Chap. 5.

Collaborative and competitive random decision rules (CCRDR) is an advanced framework of ensemble learning based on the modifications to Bagging based methods such as Random Forests as mentioned in Chap. 6. The incorporation of competitive learning can bring flexibilities into training stage in order to find as better fitness as possible between chosen learning algorithms and data samples. In other words, each algorithm may have different levels of fitness to different samples of data. Also, different algorithms may perform to a different level of accuracy on the same sample. In this context, competitive learning is useful to find the best potential candidate on each particular sample. In addition, appropriate employment on the criteria in weighted voting is relevant to measure more accurately the reliability of a classifier in making a particular decision/prediction. The empirical results shown in Chap. 8 prove that the appropriate employment mentioned above usually improves the overall classification accuracy. In addition, the development of another framework of ensemble learning called hybrid ensemble rule based classification also helps improve the quality of each single rule generated by learning from training set.

In comparison with other popular machine learning methods such as neural networks, support vector machine and k nearest neighbor, rule learning methods have significant advantages in terms of model transparency and depth of learning.

In terms of model transparency, neural network is in black box and thus poorly transparent. This would usually make it difficult for a general audience to understand the principles of making predictions. In data mining tasks, a general audience would only know the mapping between problems (inputs) and solutions (outputs), but would not be aware of the reason for the mapping. In data mining tasks, it is more important to find the reason between problems and solutions for knowledge discovery.

Support vector machine is not in black box, but it is still less transparent to a general audience. This is because the model built by using this algorithm is function lines as decision boundaries in geometric form or a piecewise function in algebraic form. This type of model representation would usually be less interpretable to a non-technical audience who does not know mathematics well.

K nearest neighbor does not aim to build a model but just to memorize all data instances in the training stage. Therefore, it is less transparent about what it has learned due to the absence of transformation from data to information/knowledge. In other words, the audience would usually be not interested in pure data but the pattern that is hidden inside the data. In contrast to the three methods above, rule based methods are in white box as the models built by using this type of methods are a set of if-then rules. Most audiences would easily understand the logical relationship between causes (inputs) and effects (outputs). Therefore, such models are highly interpretable so that it could be clearly known by a general audience in what way the predictions are made. This advantage would usually make rule based methods more popular than other machine learning methods for data mining tasks to extract the knowledge discovered from data.

In terms of depth of learning, a neural network does not involve a sound theory in learning strategy. In contrast, it practically starts from random construction of the network with regards to the neurons, connections and weights for inputs and then makes corrections to the network topology through experience. It indicates that a neural network does not have a clear learning outcome. A support vector machine involves a sound theory in learning strategy but the learning by this method is not in depth. This is because this method does not go through all data instances but just takes a look at a few instances that are selected as support vectors. The k nearest neighbor just involves a simple memorized learning in training stage as mentioned earlier. Therefore, the depth of learning using this method is quite insufficient. On the basis of above descriptions on support vector machine and k nearest neighbor, both methods are seen as lazy learning. The types of learning would just aim to identify ways to solve a specific problem rather than to pay attention to the essence of the solution in depth.

In contrast to the three above methods, rule learning methods aim to discover the causal relationship between problems and solutions by going through the whole data sets and to represent the causal relationship in the form of decision trees or if-then rules. In this way, the method does not only find the way to solve a problem but also brings out the essence of the solution in depth.

In addition, rule based learning methods are capable of dealing with both discrete and continuous attributes. In contrast, neural networks and support vector machines are less effective in dealing with discrete attributes. The k nearest neighbor is capable of dealing with ordinary attributes according to the rank of values such as 'very large', 'large', 'medium', 'small' and 'very small'. However, it is still less effective to deal with other types of discrete attributes for this method. Overall, the above description indicates that rule based learning methods would be useful and popular in both data mining and machine learning tasks due to their good transparency and depth of learning.

9.4 Philosophical Aspects

This book mainly focuses on scientific concepts. However, the concepts also have some philosophical aspects, which are discussed below.

One of the aspects is about the understanding of data mining and machine learning as well as the difference between them from conceptual point of view.

A criticism that has been put forward and that is still currently used is that a machine is neither able to learn nor to get beyond people scientifically. The argument is that a machine is developed by people and the performance and actions of the machine is totally dependent on the design and implementation by engineers and programmers. It is true that a machine is controlled by a program in executing anything. However, what if the program is about a learning method? The answer would be obviously that the machine executes the program to learn something. On the other hand, if a machine is thought to be never beyond people, it would be equivalent to imply in human learning that a student would never get beyond his/her teacher. It is not really true especially if the student has the strong capability to learn independently without being taught by teachers. Therefore, this should also be valid in machine learning if the machine holds a good learning method.

On the other hand, machine learning needs to be given an agreed definition, as currently there is still no unified definition of machine learning.

Langley defined in [18] that "*Machine learning is a science of the artificial. The field's main objects of study are artifacts, specifically algorithms that improve their performance with experience*."

Mitchell defined in [19] that "*Machine Learning is the study of computer algorithms that improve automatically through experience*."

Alpaydin defined in [20] that "*Machine learning is programming computers to optimize a performance criterion using example data or past experience*."

All of the three definitions would not be sufficient. The first one points out the field and objects of the study. The second one mentions the expected outcome of the research in the context of computer algorithms. The third one specifies the way to improve the performance of computer programs. However, none of them makes a strong relationship to 'learning theory'. Generally speaking, machine learning would be inspired by human learning in order to simulate the process of learning in

computer software. In other words, in the name of machine learning, it would tell people that machine is capable of learning. Therefore, the definition of machine learning would be in relation to learning methods, learning outcome and depth of learning.

In connection to data mining, there are some subtle misconceptions. Firstly, people think that data mining is an application of machine learning, which is not really true. Machine learning methods are usually just involved in the key stage for knowledge discovery in data mining tasks. In other words, it is required to do data collection and pre-processing prior to knowledge discovery in data mining tasks. For knowledge discovery, the methods adopted do not have to be machine learning methods. In principle, it could be done by experts manually walking through data or other statistical methods without actual learning. However, data mining is defined as a branch of artificial intelligence. Therefore, machine learning would be obviously one of the most popular approaches. On the other hand, data mining also involves some other tasks that are not done by machine learning techniques. For example, data mining needs to pre-process data such as feature selection/extraction and sampling by using statistical methods. In the past research, these types of methods are misclassified as machine learning methods. Strictly speaking, it is not really true because there is no learning done when these methods are used. For example, Principle Component Analysis (PCA) is just a statistical method to calculate the eigenvalues for a feature set and to make a judgment on principle components and their rankings according to their corresponding eigenvalues. These methods can be defined as tools to support learning methods in machine learning tasks. In other words, machine learning also needs methods relating to data pre-processing to improve the performance of learning algorithms. On the basis of above description, machine learning strongly overlaps with data mining in scientific research.

In scientific aspects, data mining and machine learning incorporate almost the same theoretical concepts and most methods, such as decision trees, Naive Bayes and k nearest neighbor, belong to both areas. Therefore, it seems that there is no obvious difference between data mining and machine learning from this point of view. However, the two subjects actually have very different practical purposes. Data mining is aimed at knowledge discovery, which means it is working in white box so that the pattern discovered can be visible to people and will be further used as knowledge or information. In contrast, machine learning is aimed at predictive modelling, which means it is working in black box and the model actually represents the knowledge learned from data but is only used further to help make predictions. From this point of view, people are not interested in the model contents but only in the model outputs. This is very similar to that students do not pay attention to principles of knowledge in depth but just want to know how to apply the knowledge they learned. On the basis of above descriptions, data mining and machine learning thus have different significance in validation. In detail, data mining is generally processing large volume of data in order to discover as correct a pattern as possible. On this point, the efficiency in training stage is critical as it can reflect if the chosen algorithm is computationally feasible in practice. The efficiency

in testing stage is not critical as the testing aims to check the reliability of the model for further use as knowledge or information. However, machine learning may generally process relatively small data in real applications. Therefore, the efficiency is not critical in training stage but is in testing stage. This is because training could be done offline and aims to learn knowledge from data but testing aims not only to check how accurately a prediction is made by the model but also how quickly the prediction is made due to the fact that prediction needs to be not only right but also quick. On the other hand, with regard to accuracy, data mining aims to measure to what extent the model can be trusted if it is further used as knowledge. In contrast, machine learning aims to measure how accurately the model can make predictions.

From another point of view, the difference between data mining and machine learning is also like the difference between human research and learning. As mentioned above, the purpose of data mining is for knowledge discovery. In other words, data mining acts as a researcher in order to discover something new which is previously unknown. Therefore, it is similar to research tasks and thus significant for researchers to guarantee the correctness of the pattern discovered from data. Machine learning is like human learning tasks. In other words, machine learning acts as a learner/student to learn something new which is previously known. To what extent the learning outcomes are achieved is typically measured by assessments such as examination or coursework. From this point of view, it is more important to achieve a high level of predictive accuracy on unseen instances in comparison with the correctness of the model built. This is because predictive accuracy is like marks awarded from assessments whereas model correctness is like the correctness of knowledge learned. In fact, it is possible that students do not understand the principles of knowledge in depth but can correctly answer exam questions to gain marks. Similarly, a model built by a machine learning algorithm may have bias and defects but can correctly make predictions. For example, some strategies in machine learning, such as conflict resolutions and assigning a default classification mentioned in Chap. 3, are like some examination skills in human learning. From this point of view, it indicates that not all of machine learning methods could be well used in data mining tasks. This is just like that not all of the learning methods could be evolved to a research method. In human learning, learning methods could be classified according to education level such as fundamental education, higher education and training education. Generally speaking, probably only the learning methods applied in higher education are more likely to be evolved to research methods. This is because this type of methods could usually better help learners develop the understanding of principles in depth. For example, a learning method may help students gain skills for practical applications but not develop understanding of principles in depth. This would usually result in a case that student can well apply what they learned in depth but cannot make other people understand what they learned. It is equivalent to that a model built by using a machine learning method has a good predictive accuracy but is poorly interpretable. Therefore, some machine learning methods that do not aim to learn in depth would not become good data mining methods.

The second philosophical aspect is on the understanding of ensemble learning in the context of learning theory. As mentioned in Chap. 2, ensemble learning can be done in parallel or sequentially. In the former way, there is no collaboration among different learning algorithms in the training stage and only their predictions are combined. In academic learning theory, this is like team working, which means students learn knowledge independently and only work on group works together using their knowledge. Their ways of making collaborations in the group work are just like the strategies in making final predictions in ensemble learning. In another way of ensemble learning, there are collaborations involved in the training stage in the way that the first algorithm aims to learn models and then the latter one learns to correct the models etc. In academic learning theory, this is like group learning with interactions among students in order to improve the learning skills and to gain knowledge more effectively.

The third philosophical aspect is on the understanding of ensemble rule based systems in the context of system theory [21]. As mentioned earlier, an ensemble rule based system consists of a group of single rule based systems in general, each of which is a subsystem of the ensemble system. In other words, it is a system of systems like a set of sets in set theory [22]. In addition, an ensemble rule based system can also be a subsystem of another ensemble system in theory. In other words, a super ensemble rule based system contains a number of clusters, each of which represents a subsystem that consists of a group of single rule based systems.

The fourth philosophical aspect is on the understanding of rule based systems in the context of discrete mathematics such as mathematical logic and relations. With respect to mathematical logic, rule based systems theory has connections to conjunction, disjunction and implication. In machine learning, each rule in a rule set has disjunctive connections to the others. In addition, each rule consists of a number of rule terms, each of which typically has conjunctive connections to the others. In rule based systems, each rule is typically in the form of if-then rules. In this context, it can represent an implication from the left hand side (if part) to the right hand side (then part) for logical reasoning. With respect to relations, rule based systems can reflect a functional mapping relationship between input space and output space. In other words, the if-then rules in the system must not reflect any one-to-many relationship, which means the same inputs must not be mapped to different outputs, which is a restriction similar to functions. In rule based systems, this is usually referred to as consistency.

The fifth philosophical aspect is on the novel application of mathematical theory and object oriented programming concepts. As introduced in Chap. 2, a rule base is used to manage rules that have common attributes for both inputs and outputs. A rule base can be seen as a component of a rule set.

In connection to functions as part of mathematical theory, a rule base can be seen as an abstract function, denoted as $f(x_1, x_2, \ldots, x_n)$, without a specific expression. In this context, a rule can be seen as a specific function with a specific expression and domain constrains for its inputs such as the notation below:

$$f(x_1, x_2) = \begin{cases} 1, & \textit{if } \mathrm{x1} = 1 \wedge \mathrm{x2} = 1 \\ 0, & \textit{if } x1 = 0 \vee x2 = 0 \end{cases}$$

In the notation above, there are two rules: if $x_1 = 1$ and $x_2 = 1$ then $f(x_1, x_2) = 1$; if $x_1 = 0$ and $x_2 = 0$ then $f(x_1, x_2) = 0$. In other words, each of rules in a rule base is corresponding to a branch of a function that is corresponded from the rule base.

In connection to object oriented programming concepts, a rule set can be seen as a subclass of abstract rule based systems. This is because a rule based system consists of a set of rules as mentioned in Chap. 1. In this context, a rule based system can be defined as a class and a rule set as an object of the system in the concept of object oriented programming. As it is unknown with respect to what rules a rule set consists of, the class that is defined to represent a rule based system would be abstract, which relates to abstraction as a part of object oriented techniques. As mentioned above, a rule base can be seen as an abstract function, which is actually corresponding to an abstract method in object oriented programming. A rule set consists of a number of rule bases which would have different input and output attributes, which is corresponding to another object oriented technique known as polymorphism. This is because it is achievable that different functions (methods) have the same name but different input parameters and type of return values. Therefore, rule bases in a rule set could be seen as abstract (functions) methods, which have the same name but different input parameters and type of return values, in a class defined for a type of rule based systems to which the rule set belongs. In addition, each of the rules in a rule base is corresponding to a branch in an if-then-else statement.

In practice, when a training set is given, an abstract class is defined for rule based systems with a number of rule bases. This is because all of possible rule bases could be derived from attribute information of the dataset. Each of the rule bases is defined by an abstract function (method). For each abstract function, its input parameters and type of return values are specified according to the input and output attributes related to the corresponding rule base. Once a particular rule based learning algorithm is chosen, a subclass of the abstract class, which is corresponding to a rule set generated using this algorithm, is created. All abstract functions, each of which represent a rule base, are overridden and overloaded in the subclass. This indicates that each of rule bases is filled by rules if the rules belong to the rule base or defined as null if none of the generated rules fits the rule base. In programming, it is equivalent to implement a function which is originally abstract by providing a specific program statement or leaving the body of the function blank. Once a test instance is given, an object of the subclass is specified to call the functions, each of which is corresponding to a rule base, in order to make predictions.

The last philosophical aspect is on the relationship of the research methodology to three main theories namely information theory, system theory and control theory. From philosophical point of view, the three main theories mentioned above could be understood by the following context:

Information theory generally means passing information from one property to another one. In the process of information passing, it actually happens to have interactions between the two properties. This could be seen as a relationship to system theory. In other words, the two properties are supposed to be two components of a system. However, it is necessary to ensure that the information passing is effective and efficient with a high quality. This is because in the process of information passing there may be noise that is present and interferes the transmission. In addition, there may be some information that is confidential to third parties. In this case, the information usually needs to be encrypted on senders' side and then decrypted on receivers' side. The context described above would belong to control theory.

In many other subject areas, the three main theories are also highly significant. A typical example would be in humanities and social science. This world consists of humans, animals, plants and all other non-biological individuals/systems. From this point of view, no one is living alone in the world. Therefore, everyone needs to have interactions with others. This indicates the involvement of system theory to identify the way to interact among individuals/groups. However, the way to achieve interactions would typically be through information passing. The way of passing information could be in many different forms such as oral, written and body languages and some other actions. This brings in control theory in order to effectively control the way of information passing. This is because inappropriate ways may result in serious accidents due to misunderstanding of information or unaccepted actions on receivers' side. Therefore, the three main theories would composite an organized entirety in real applications for most types of problem solving.

In this book, the research methodology is developed along all of the three main theories. In detail, the research methodology includes a unified framework for construction of single rule based systems. As introduced in Chap. 2, this framework consists of three modules namely rule generation, rule simplification and rule representation. This could be seen as an application of system theory. In rule generation, a newly developed method referred to as IEBRG is based on entropy which is a technique of information theory. In rule simplification, a newly developed pruning algorithm called Jmid-pruning is based on J-measure which is also an information theoretical technique. On the other hand, both rule simplification and rule representation are incorporated into the framework in order to control machine learning tasks in training stage and testing stage respectively. In detail, rule simplification aims to effectively control the generation of rules towards reduction of overfitting and rule complexity, as well as efficiency in training stage. Rule representation aims to effectively control the process of prediction towards improvement of efficiency in testing stage.

This book also has two frameworks of ensemble learning introduced. As introduced in Chap. 6, ensemble learning generally aims to combine different models that are generated by a single or multiple algorithm(s) in order to achieve collaborative predictions. In the two frameworks, there are both collaborations and competitions involved. Multiple algorithms make up an ensemble learning systems and multiple generated rule sets compose an ensemble rule based classifier.

Therefore, the development of the two frameworks involves the application of system theory. However, competitions among classifiers aim to choose the ones with higher quality. The way to measure the quality of each classifier is significant and critical and thus control theory needs to be involved. In addition, in prediction stage, each individual classifier would provide a prediction with its confidence to the final prediction maker. It indicates there is information passing between individuals and thus the application of information theory is also involved in this environment.

9.5 Further Directions

As mentioned in Chap. 2, two theoretical frameworks are introduced in the book for construction of rule based classification systems and ensemble learning. The two frameworks can be combined for construction of ensemble rule based systems. In this context, the combined framework will further be transformed into another framework referred to as networked rule bases [23–25]. A networked rule base consists of a number of single rule bases as illustrated in Fig. 9.1.

In this network, each node represents a rule base. The nodes can be connected sequentially or in parallel. In detail, each of variables labelled x_{m-1}, while m represents the number of layer in which the node locates, represents an input and y represents the output. In addition, each of these labels labelled z^{m-2} represents an intermediate variable, which means this kind of variable is used as output for a former rule base and then again as inputs for a latter rule base as illustrated in Fig. 9.1. On the other hand, there are two kinds of nodes representing rule bases as illustrated in Fig. 9.1, one of which is a type of standard rule bases and labelled RB_{m-1}. This kind of nodes is used to transform the input(s) to output(s). The other type of nodes, in addition to the standard type, represents identities. It can be seen from the Fig. 9.1 that this type of nodes does not make changes between inputs and outputs. This indicates the functionality of an identity is just like an email transmission, which means the inputs are exactly the same as outputs.

In practice, a complex problem could be subdivided into a number of smaller sub-problems. The sub-problems may need to be solved sequentially in some cases.

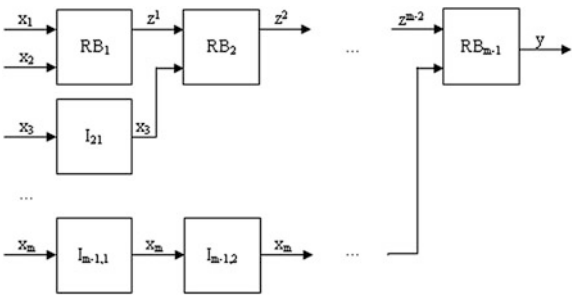

Fig. 9.1 Rule based network (modular rule bases) from [23–25]

They can also be solved in parallel in other cases. In connection to machine learning context, each sub-problem could be solved by using a machine learning approach. In other words, the solver to each particular sub-problem could be a single machine learner or an ensemble learner consisting of a single rule base.

On the other hand, a unified rule based network topology is introduced in Chap. 5. However, this topology can be generalized to fit any type of networks which are used to do computation such as neural networks, Bayesian networks and digital circuits. The topology is illustrated in Fig. 9.2.

In this network, the middle layers represent computation layers, which means that each node in this kind of layers represents a special type of computation such as conjunction, disjunction, weighted majority voting, weighted averaging and logical AND, OR and NOT. These operations can also be used in a same network representing a hybrid computational network topology. In such a type of network, there can be either a single computation layer or multiple computation layers as illustrated in Fig. 9.2. This is very similar to neural network topology which could be of either single layer perception or multi-layer perception. Similar to the rule based network topology introduced in Chap. 5 as well as neural networks, each input is assigned a weight when its corresponding value is used for computation. An output from a node in a computation layer is used again as an input with a weight to another node in a latter computation layer if applicable. In practice, this network topology can potentially fulfil the requirement that multiple types of computation must be combined to solve a particular problem.

So far, ensemble learning concepts introduced in machine learning literature mostly lie in single learning tasks. In other words, all algorithms involved in ensemble learning need to achieve the same learning outcomes in different strategies. This is defined as local learning by the authors of this book. In this context, the

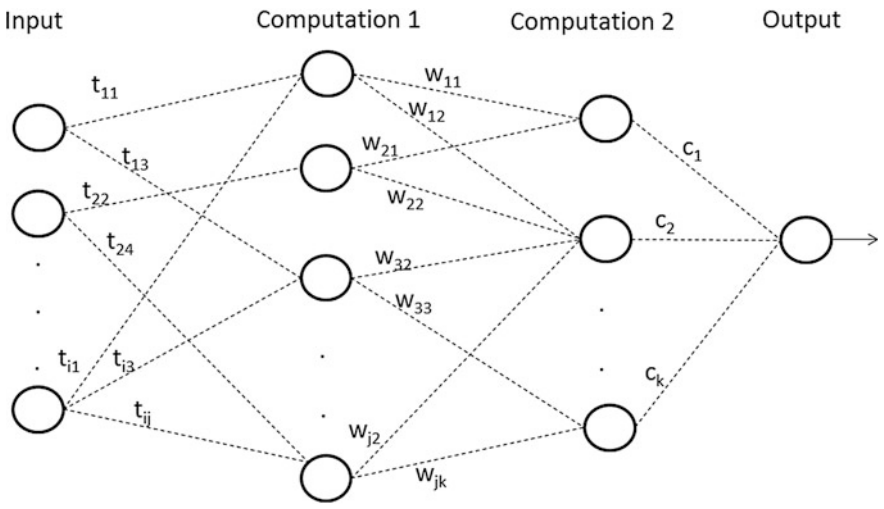

Fig. 9.2 Generic computational network

further direction would be to extend the ensemble learning framework to achieve global learning by means of different learning outcomes. The different learning outcomes are actually not independent of each other but have interconnections. For example, the first learning outcome is a prerequisite for achieving the second learning outcome such as deep learning [26]. This direction of extension is towards evolving machine learning approach in a universal vision. To fulfil this objective, the networked rule bases can actually provide this kind of environment for discovering and solving problems in a global way. In military process modelling and simulation, each networked rule base can be seen as a chain of commands (chained rule bases) with radio transmissions (identities). In a large scale raid, there may be more than one chain of commands. From this point of view, the networked topology should have more than one networked rule bases parallel to each other. All these networked rule bases should finally connect to a single rule base which represents the Centre of command.

As mentioned in Chap. 1, the main focus of the book is on rule based systems for classification. However, rule based systems can also be used for regression [27, 28] and association [29, 30]. Therefore, all of the completed and future work mentioned in the book can also be extended to regression and association subject areas for construction of rule based systems. On the other hand, the research methodology introduced in Chap. 3 is mainly based on deterministic logic. In the future, the methodology can also be extended to be based on probabilistic and fuzzy logic in practical applications.

Chapter 7 lists some impact factors for interpretability of rule based systems as well as some criteria for evaluation of the interpretability. In general, it applies to any types of expert systems. Therefore, in order to improve the interpretability of expert systems, it is necessary to address the four aspects namely, scaling up algorithms, scaling down data, selection of rule representation and assessment of cognitive capability, in accordance with the criteria for evaluation of the interpretability.

Scaling up algorithms can improve the transparency in terms of depth of learning. For example, rule based methods usually generate models with good transparency because this type of learning is in a great depth and on an inductive basis. On the other hand, the performance of a learning algorithm would also affect the model complexity as mentioned in Chap. 7. In this case, the model complexity could be reduced by scaling up algorithms. In the context of rule based models, complexity could be reduced through proper selection of rule generation approaches. As mentioned in Chap. 3, the separate and conquer approach is usually likely to generate less complex rule sets than the divide and conquer approach. In addition, it is also helpful to employ pruning algorithms to simplify rule sets as introduced in Chap. 5. In this way, some redundant or irrelevant information is removed and thus the interpretability is improved.

Scaling down data usually results in the reduction of model complexity. This is because model complexity is usually affected by the size of data. In other words, if a data set has a large number of attributes with various values and instances, the generated model is very likely to be complex. As introduced in Chap. 1, the

dimensionality issue can be resolved by using feature selection techniques, such as entropy [31] and information gain [32], both of which are based on information theory pre-measuring uncertainty present in the data. In other words, the aim is to remove those irrelevant attributes and thus make a model simpler. In addition, the issue can also be resolved through feature extraction methods, such as Principal Component Analysis (PCA) [33] and Linear Discriminant Analysis (LDA) [34]. On the other hand, when a dataset contains a large number of instances, it is usually required to take advantage of sampling methods to choose the most representative instances. Some popular methods comprise simple random sampling [35], probabilistic sampling [36] and cluster sampling [37]. Besides, it is also necessary to remove attribute values due to the presence of irrelevant attribute values. For example, in a rule based method, an attribute-value pair may be never involved in any rules as a rule term. In this case, the value of this attribute can be judged irrelevant and thus removed. In some cases, it is also necessary to merge some values for an attribute in order to reduce the attribute complexity especially when the attribute is continuous with a large interval. There are some ways to deal with continuous attributes such as kerber [38] and use of fuzzy linguistic terms [39].

As introduced in Chap. 5, a change of model representation would usually result in the change of model interpretability. As also introduced, rule based models could be represented in different forms such as decision tree and linear list. These two representations usually have redundancy present. For example, a decision tree may have the replicated subtree problem and a linear list may have the attribute appear in different rules on a repetitive basis. This kind of problem could be resolved by converting to a rule based network representation as argued in Chap. 5.

However, due to the difference in level of expertise and personal preferences from different people, the same model representation may demonstrate different level of comprehensibility for different people. For example, people who do not have a good background in mathematics may not like to read information in mathematical notations. In addition, people in social sciences may not understand technical diagrams that are usually used in engineering fields. On the basis of above description, cognitive capability needs to be assessed to make the knowledge extracted from rule based systems more interpretable to people in different domains. This can be resolved by using expert knowledge in cognitive psychology and human-machine engineering, or by following machine learning approaches to predict the capability as mentioned in Chap. 7.

The above discussion recommends that the four ways, namely, scaling up algorithms, scaling down data, selection of model representation and assessment of cognitive capability, can be adopted towards potential improvement of interpretability of rule based systems in the future.

Finally, a unified framework for control of machine learning tasks is proposed as illustrated in Fig. 9.3. This is in order to effectively control the pre-processing of data and to empirically employ learning algorithms and models generated. As mentioned in Chap. 1, it is also relevant to scale down data in addition to scaling up algorithms for improvement of classification performance. In fact, a database is daily updated in real applications, which results in the gradual increase of data size

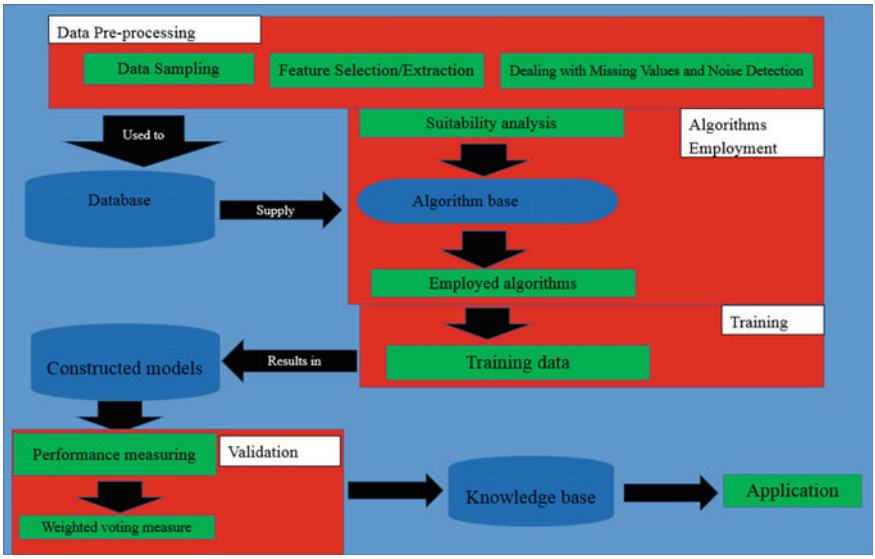

Fig. 9.3 Unified framework for control of machine learning tasks

and in changes to patterns existing in the database. In order to avoid the decrease of computational efficiency, the size of sample needs to be determined in an optimal way. In addition, it is also required to avoid the loss of accuracy. From this point of view, the sampling is critical not only in the size of sample but also in the representativeness of the sample. Feature selection/extraction is another critical task with regard to pre-processing of data. As mentioned in Chap. 1, high dimensional data would usually results in high computational costs. In addition, it is also very likely to contain irrelevant attributes which result in noise and coincidental patterns. In some cases, it is also necessary to effectively detect noise if the noise is introduced artificially. For example, noise may be introduced in a dataset due to mistakes in typing or illegal modifications from hackers. A potential solution would be using association rules to detect that the value of an attribute is incorrect on the basis of the other attribute-value pairs in the data instance. Appropriate employment of learning algorithms and models are highly required because of the fact that there are many machine learning algorithms existing but no effective ways to determine which of them are suitable to work on a particular data set. Traditionally, the decision is made by experts based on their knowledge and experience. However, it is very difficult to judge the correctness of the decision prior to empirical validation. In real applications, it is not realistic to frequently change decisions after confirming that the chosen algorithms are not suitable.

The above description motivates the development of the framework for control of machine learning tasks. In other words, this framework aims to use machine learning techniques to control machine learning tasks. In this framework, the employment of both algorithms and models follows machine learning approach.

The suitability of an algorithm and the reliability of a model are measured by statistical analysis on the basis of historical records. In detail, each algorithm in the algorithms base, as illustrated in Fig. 9.3, is assigned a weight which is based on its performance in previous machine learning tasks. The weight of an algorithm is very similar to the impact factor of a journal which is based on its overall citation rate. In addition, each model generated is also assigned a weight which is based on its performance on latest version of validation data in a database. After the two iterations of employment, a knowledge base is finalized and deployed for real applications as illustrated in Fig. 9.3.

This unified framework actually includes the three main theories involved namely, information theory, system theory and control theory as introduced in Sect. 9.4. In this framework, there are four modules namely, data pre-processing, algorithms employment, training and validation, and four bases namely, database, algorithm base, model base and knowledge base. The four bases are used to store and manage information in different forms which is in relation to information theory. The four modules are established to control machine learning tasks with respect to decision making in data sampling, use of algorithms and build and validation of models, which relates to control theory. There are also interactions between modules such as passing of chosen data, algorithms and models. What is passing between modules would be a special form of information, which could be seen as a kind of communication and thus relates to information theory. In addition, the interactions between modules would be seen as behavior of coordination between systems, which relates to system theory.

The unified framework illustrated in Fig. 9.3 provides a Macro vision for research in data mining and machine learning. This would fit the situations in real applications of machine learning. This is because in reality machine learning tasks are usually undertaken in complex environments unlike in laboratories. In the latter environment, research is usually undertaken in a Micro vision and in a pre-processed environment which ignores or eliminates all other impact factors with regard to performance of machine learning tasks. In the future, the approaches introduced in Chaps. 3, 4, 5 and 6 together with other existing approaches will be integrated into the framework for simulation of the control process.

References

1. Biggs, N., Lloyd, E., Wilson, R.: Graph Theory. Oxford University Press, Oxford (1986)
2. Bondy, J.A., Murty, U.S.R.: Graph Theory. Springer, Berlin (2008)
3. Aho, A.V., Hopcroft, J.E., Ullman, J.D.: Data Structures and Algorithms. Addison-Wesley, Boston (1983)
4. Johnsonbaugh, R.: Discrete Mathematics. Prentice Hall, USA (2008)
5. Battram, A.: Navigating Complexity: The Essential Guide to Complexity Theory in Business and Management. Spiro Press, London (2002)
6. Schlager, J.: Systems engineering: key to modern development. IRE Trans. EM. **3**(3), 64–66 (1956)
7. Sage, A.P.: Systems Engineering, San Francisco. Wiley IEEE, CA (1992)

8. Hall, A.D.: A Methodology for Systems Engineering. Van Nostrand Reinhold, New York (1962)
9. Goode, H.H., Machol, R.E.: Systems Engineering: An Introduction to the Design of Large-scale Systems. McGraw-Hill, New York (1957)
10. Aksoy, M.S.: A review of rules family of algorithms. Math. Compu. Appl. **1**(13), 51–60 (2008)
11. Liu, W.Z., White, A. P.: A review of inductive learning. In: Research and Development in Expert Systems VIII, Cambridge (1991)
12. Mrozek, A.: A new method for discovering rules from examples in expert systems. Int. J. Man Mach. Stud. **36**, 127–143 (1992)
13. Hart, A.: Knowledge Acquisition for Expert systems. Chapman and Hall, London (1989)
14. Quinlan, R.: Induction, knowledge and expert systems. In: Artificial Intelligence Developments and Applications, Amsterdam (1988)
15. Quinlan, R.: Inductive knowledge acquisition: a case study. In: Quinlan, R. (ed.) Applications of Expert Systems, pp. 157–173. Turing Institute Press, UK (1987)
16. Michalski, R., Mozetic, I., Hong, J., Lavrac, N.: The multi-purpose incremental learning system AQ15 and its testing applications to three medical domains. In: The Fifth National Conference on Artificial Intelligence, Philadelphia, PA 1986
17. Wooldridge, M.: An Introduction to Multi-Agent Systems. Wiley, New Jersey (2002)
18. Langley, P.: Elements of Machine Learning, San Francisco. Morgan Kaufmann Publishers Inc, CA (1995)
19. Mitchell, T.: Machine Learning. McGraw Hill, New York (1997)
20. Alpaydin, E.: Introduction to Machine Learning (Adaptive Computation and Machine Learning). MIT Press, Massachusetts (2004)
21. Stichweh, R.: Systems Theory. In: Badie, B.E.A. (ed.) International Encyclopaedia of Political Science. Sage, New York (2011)
22. Jech, T.: Set Theory, Thrid Millennium edn. Berlin, New York: Springer (2003)
23. Gegov, A.: Fuzzy Networks for Complex Systems: A Modular Rule Base Approach. Springer, Berlin (2010)
24. Gegov, N. Petrov, N., Vatchova, B,: Advance modelling of complex processed by rule based networks. In: 5th IEEE International Conference on Intelligent Systems, London (2010)
25. Gegov, A., Petrov, N., Vatchova, B., Sanders, D.: Advanced modelling of complex processes by fuzzy networks. WSEAS Trans. Circ. Syst. **10**(10), 319–330 (2011)
26. Bengio, Y.: Learning deep architectures for AI. Found. Trends. Mach. Learn. **2**(1), 1–127 (2009)
27. Freedman, D.A.: Statistical Models: Theory and Practice. Cambridge University Press, Cambridge (2005)
28. Armstrong, J.S.: Illusions in regression analysis. Int. J. Forecast. **28**(3), 689 (2012)
29. Okafor, A.: Entropy Based Techniques with Applications in Data Mining. Florida (2005)
30. Aitken, A.C.: Statistical Mathematics, 8th edn. Oliver & Boyd (1957)
31. Shannon, C.: A mathematical theory of communication. Bell Syst. Tech. J. **27**(3), 379–423 (1948)
32. Azhagusundari, B., Thanamani, A.S.: Feature selection based on information gain. Int. J. Inno. Tech. Exploring. Eng. **2**(2), 18–21 (2013)
33. Jolliffe, I.T.: Principal Component Analysis. Springer, New York (2002)
34. Yu, H., Yang, J.: A direct LDA algorithm for high diomensional data—with application to face recognition. Pattern Recogn. **34**(10), 2067–2069 (2001)
35. Yates, D. S., David, S. M., Daren, S. S.: The Practice of Statistics, 3rd edn. Freeman (2008)
36. Deming, W.E.: On probability as a basis for action. Am. Stat. **29**(4), 146–152 (1975)
37. Kerry, Bland, : Statistics notes: the intracluster correlation coefficient in cluster randomisation. Br. Med. J. **316**, 1455–1460 (1998)
38. Kerber, R.: ChiMerge: discretization of numeric attribute. In: Proceeding of the 10th National Conference on Artificial Intelligence (1992)
39. Ross, T.J.: Fuzzy Logic with Engineering Applications, 2nd edn. Wiley, West Sussex (2004)

Appendix 1
List of Acronyms

Bagging	Bootstrap aggregating
Boosting	Adaboost
CCRDR	Collaborative and competitive decision rules
CART	Classification and regression trees
DT	Decision trees
KNN	K nearest neighbors
IEBRG	Information entropy based rule generation
LDA	Linear discriminant analysis
LL	Linear lists
NN	Neural networks
PCA	Principal component analysis
RBN	Rule based networks
RF	Random forests
SVM	Support vector machines
TDIDT	Top-down induction of decision trees
UCI	University of California, Irvine

H. Liu et al., *Rule Based Systems for Big Data*,
Studies in Big Data 13, DOI 10.1007/978-3-319-23696-4

Appendix 2
Glossary

Terms in machine Learning	Terms in other related areas
Attribute, feature	Variable, field, column
Instance	Record, data point, tuple, row
Training, learning	Modelling, building, construction
Testing, prediction	Verification, validation, checking
Classifier, learner	Model, expert system, hypothesis
Inconsistency	Overlapping
Missing value	Unknown value
Dimensionality	Number of attributes/variables
Data size	Number of instances/data points
Classification	Categorical prediction, decision
Regression	Numerical prediction
Association	Correlation
Clustering	Grouping
Noise	Incorrect record
Classification/regression/association rules	If-then rules, decision rules
Classification/regression trees	Decision trees
Efficiency in training stage	Modelling speed
Efficiency in testing stage	Prediction speed
Computational complexity	Time complexity
Rule based classifier	Rule set
Rule based learner	Rule based model, rule based system
Rule based ensemble learner	Ensemble rule based system
Class, label	Output
Attribute value	Input/output

H. Liu et al., *Rule Based Systems for Big Data*,
Studies in Big Data 13, DOI 10.1007/978-3-319-23696-4

Appendix 3
UML Diagrams

See Figs. A.1, A.2, A.3, A.4

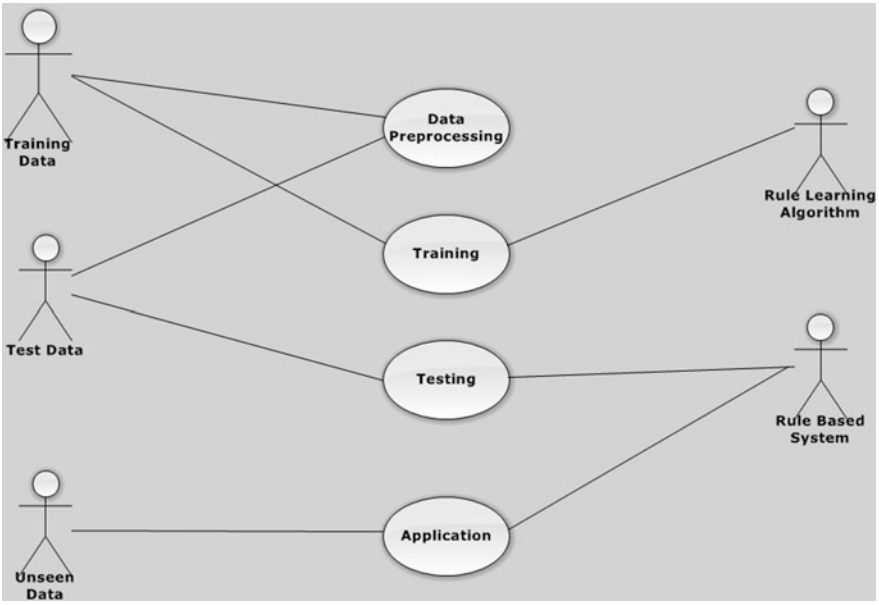

Fig. A.1 UML use case diagram for machine learning scenarios

H. Liu et al., *Rule Based Systems for Big Data*,
Studies in Big Data 13, DOI 10.1007/978-3-319-23696-4

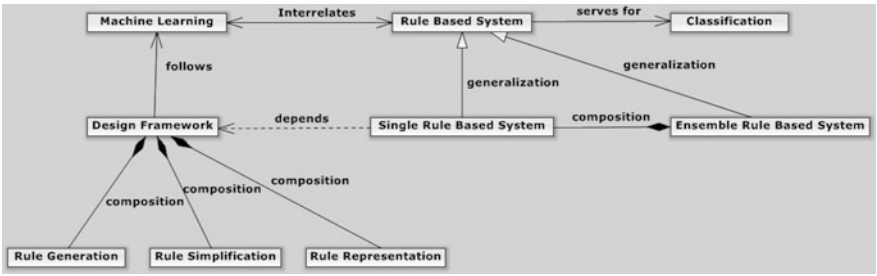

Fig. A.2 UML class diagram for research framework of rule based systems

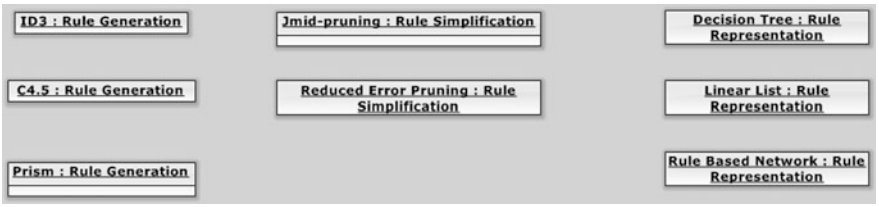

Fig. A.3 UML instance diagram for generation, simplification and representation of rules

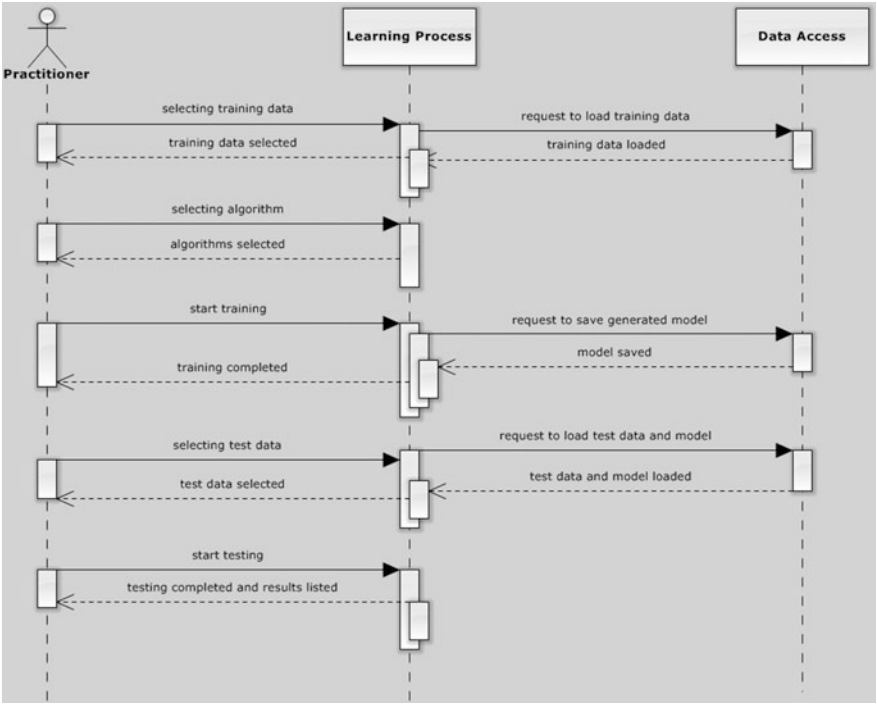

Fig. A.4 UML sequence diagram for machine learning systems

Appendix 4
Data Flow Diagram

See Fig. A.5

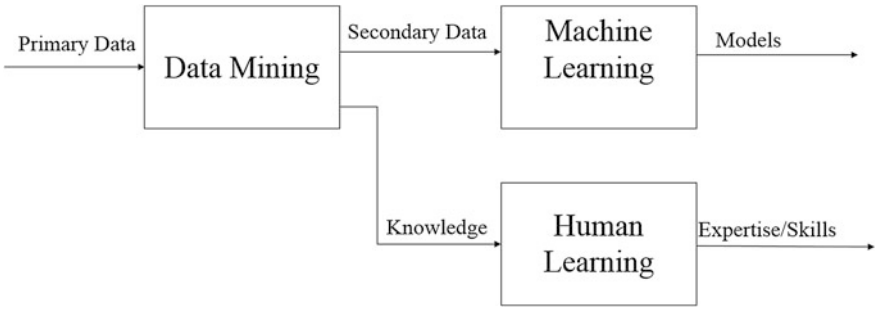

Fig. A.5 Chained relationship between data mining and machine learning

H. Liu et al., *Rule Based Systems for Big Data*,
Studies in Big Data 13, DOI 10.1007/978-3-319-23696-4