Data Mining of Range-Based Classification Rules for Data Characterization

Achilleas Tziatzios

A thesis submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy
in Computer Science

School of Computer Science & Informatics
Cardiff University

March 2014

DECLARATION

This work has not been submitted in substance for any other degree or award at this or any other university or place of learning, nor is being submitted concurrently in candidature for any degree or other award.

Signed _____ (candidate) Date _____

STATEMENT 1

This thesis is being submitted in partial fulfillment of the requirements for the degree of PhD.

Signed _____ (candidate) Date _____

STATEMENT 2

This thesis is the result of my own independent work/investigation, except where otherwise stated. Other sources are acknowledged by explicit references. The views expressed are my own.

Signed _____ (candidate) Date _____

STATEMENT 3

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed _____ (candidate) Date _____

STATEMENT 4: PREVIOUSLY APPROVED BAR ON ACCESS

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loans **after expiry of a bar on access previously approved by the Academic Standards & Quality Committee.**

Signed _____ (candidate) Date _____

"To my family and close friends
for never quite losing their patience with me on this."

Abstract

Advances in data gathering have led to the creation of very large collections across different fields like industrial site sensor measurements or the account statuses of a financial institution's clients. The ability to learn classification rules, rules that associate specific attribute values with a specific class label, from this data is important and useful in a range of applications.

While many methods to facilitate this task have been proposed, existing work has focused on categorical datasets and very few solutions that can derive classification rules of associated continuous ranges (numerical intervals) have been developed. Furthermore, these solutions have solely relied in classification performance as a means of evaluation and therefore focus on the mining of mutually exclusive classification rules and the correct prediction of the most dominant class values. As a result existing solutions demonstrate only limited utility when applied for data characterization tasks.

This thesis proposes a method that derives range-based classification rules from numerical data inspired by classification association rule mining. The presented method searches for associated numerical ranges that have a class value as their consequent and meet a set of user defined criteria. A new interestingness measure is proposed for evaluating the density of range-based rules and four heuristic based approaches are presented for targeting different sets of rules. Extensive experiments demonstrate the effectiveness of the new algorithm for classification tasks when compared to existing solutions and its utility as a solution for data characterization.

Acknowledgements

This work would not have been possible without the support of Dow Corning Corporation and the people involved in the PR.O.B.E. project. I owe deep debt of gratitude to my supervisor Dr. Jianhua Shao for his guidance and mentorship during my years of study.

I would like to thank my friends and colleagues at the School of Computer Science and Informatics for their help and encouragement. More importantly I am very grateful to Dr. Gregorios Loukides for our long standing friendship and collaboration.

Finally, I thank my family for their continuous support and the entire EWOK team at Companies House for their understanding and encouragement during my writing up stage.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recent advances in data generation and collection have led to the production of massive data sets in commercial as well as scientific areas. Examples of such data collections vary from data warehouses storing business information, biological databases storing increasing quantities of DNA information for all known organisms and the use of telescopes to collect high-resolution images of space [10, 57]. The speed at which data is gathered has far exceeded the rate at which it is being analysed.

Data mining is a field that grew for the purpose of using the information contained in these data collections and out of the limitations of existing techniques to handle the ever growing size as well as the evolving types of data. Data Mining, often referred to as Knowledge Discovery in Databases (KDD), refers to the nontrivial extraction of implicit, previously unknown and potentially useful information from data in databases. While data mining and knowledge discovery in databases (or KDD) are frequently treated as synonyms, data mining is actually part of the knowledge discovery process. Business applications range from predictive models for management decision making, to pattern extraction for customer services personalisation as well as optimisation of profit margins.

Since the goal was to meet the new challenges, data mining methodologies are strongly connected, and often build upon existing areas of data analysis. Just like with any rapid growing research field, data mining methods have evolved as different research challenges and applications, stemming from different areas, have emerged. A significant portion of research work has actually focused on defining the field and/or its relationship to existing fields [19, 36, 47, 53, 69, 93].

1.1 Classification Rule Mining

The problem of discovering knowledge in data is complex to define. The three main areas of data mining tasks consist of *classification*, *association analysis* and *clustering* [36, 52, 106]. Classification is the task of constructing a model from data for a target variable, referred to as the *class label*. Association analysis is the discovery of patterns of strongly associated features in the given data whereas clustering seeks to find groups of closely related data values so that data that are clustered together are more similar than with the remaining data.

In certain cases, however, the desired result of the data mining process may not be as discrete as described and the designed solution is a combination of more than one of the above tasks. An example of this is the area that this thesis is focused on, associative classification rule mining [74], where the goal is mining associations of data variables that are also associated with a specific class label. The following examples show the difference between a typical classification algorithm like *Naive Bayes* [61] in Figure 1.1 and an association rule mining algorithm like *RIPPER* [22] when used to mine associations only for a specific target variable in Figure 1.2. The data used are that of the well known iris dataset which is described in Table 1.1.

Attribute	sepal length	sepal width	petal length	petal width	class
Type	numeric	numeric	numeric	numeric	categorical

Table 1.1: Description of the iris dataset. The possible class values are $\langle iris - setosa, iris - versicolor, iris - virginica \rangle$.

Note in Figure 1.2 that the RIPPER algorithm generates a complete model and not independent rules. In order for each rule to be a complete classification rule its predicate needs to include the negation of any preceding rules in the model. A typical classification solution, however as in Figure 1.1, presents its results in a single model.

Naive Bayes Classifier

Attribute	Class Iris-setosa (0.33)	Iris-versicolor (0.33)	Iris-virginica (0.33)
sepallength			
mean	4.9913	5.9379	6.5795
std. dev.	0.355	0.5042	0.6353
weight sum	50	50	50
precision	0.1059	0.1059	0.1059
sepalwidth			
mean	3.4015	2.7687	2.9629
std. dev.	0.3925	0.3038	0.3088
weight sum	50	50	50
precision	0.1091	0.1091	0.1091
petallength			
mean	1.4694	4.2452	5.5516
std. dev.	0.1782	0.4712	0.5529
weight sum	50	50	50
precision	0.1405	0.1405	0.1405
petalwidth			
mean	0.2743	1.3097	2.0343
std. dev.	0.1096	0.1915	0.2646
weight sum	50	50	50
precision	0.1143	0.1143	0.1143

Figure 1.1: Classification model built by Naive Bayes for the iris dataset.

```
JRIP rules:
===========

(petallength >= 3.3) and (petalwidth <= 1.6) and (petallength <= 4.9) => class=Iris-versicolor (46.0/0.0)
(petallength <= 1.9) => class=Iris-setosa (50.0/0.0)
 => class=Iris-virginica (54.0/4.0)

Number of Rules : 3
```

Figure 1.2: Classification rules model built by RIPPER for the iris dataset.

1.1.1 Mining of Continuous Data Ranges

Another important distinction between data mining tasks is based on the type of data mined. The type of each attribute is indicative of its underlying properties and therefore an important aspect of a data mining method is the type of data it is designed to mine. The iris data used in the example consists of numerical, more specifically continuous, data attributes. In real world applications this is expected to be the case in the majority of tasks since real world data collections often contain real numbers. However existing work in the area of classification rule mining has focused primarily on mining data of a categorical nature. Applying these solutions on continuous values requires discretization of the given data.

For example, a possible discretization of the petal length attribute of the iris dataset would transform the continuous attribute into a categorical one with three possible values $\langle(-\infty, 2.45], (2.45, 4.75], (4.75, +\infty)\rangle$ so that a method designed for a categorical attribute can be applied. These solutions may be applicable on continuous data that have been transformed to categorical but determining a good way of transforming real values to categorical ones constitutes another research problem in itself.

Not all existing solutions require continuous data to be discretized. Algorithms like RIPPER, in the given example, choose the best value at which to split a continuous attribute so that one of the resulting ranges maximizes a target measure. In some cases a rule may actually contain a range of continuous values like in Figure 1.2 the rule $(petallength \geq 3.3)$ *and* $(petalwidth \leq 1.6)$ *and* $(petallength \leq 4.9) \Rightarrow class = Iris\text{-}versicolor$ includes the range $petallength \in [3.3, 4.9]$ but this is the result of two binary splits that first selected the petal-length values that were ≥ 3.3 and amongst the values that met the requirements $(petallength \geq 3.3)$ *and* $(petalwidth \leq 1.6)$ another binary split was performed at petal length 4.9. Furthermore, any relation of the form $attribute \leq v$ may be interpreted as a range $attribute \in (-\infty, v]$ where $-\infty$ may also be replaced with the v_{min} of *attribute*. Regardless of representation, however these ranges are the result of binary splits. This thesis presents a solution that for all continuous attributes attempts to mine multiple continuous ranges directly from the real($\mathbb{R}$) values of an attribute.

1.2 Research Challenges

The data mining area has been developed in order to address limitations of traditional data analysis. However, the fast evolution of modern data collections continues to pose a plethora of important research challenges in the development of data mining solutions. This section presents an overview of these challenges and discusses how they relate to the solution presented in this thesis.

- **Scalability:** Advances in data generation and collection have led to datasets of a very large size becoming more and more common. Therefore, modern data mining solutions need to be scalable in order to handle these datasets. The development of scalable algorithms may require the implementation of novel, efficient data structures, an efficient reduction of the problem space via sampling techniques or the development of a solution that may be executed in parallel threads. The area of parallel executed solutions is evolving fast and is expected to address many of the limitations of current data mining solutions [26, 62]. Typically, scalability refers to the number of records(rows) in a dataset but modern collections may include thousands of attributes for each record, presenting researchers with the added challenge of *high dimensional* data for algorithms whose complexity increases with the number of attributes.

- **Heterogeneous and Complex Data:** Traditional data collections consist of homogeneous data therefore simplifying the analysis process. Due to the increasing number of areas where data mining is applied, however, new cases arise where heterogeneous attributes need to be mined. More importantly, the inclusion of new types of complex data objects, in the forms of text data, genomes and even structured text(code) in the case of XML documents requires the development of data mining solutions that incorporate the relations between the mined data values.

- **Distributed data sources:** In some cases the data are not located in a single central storage and possibly owned by many different legal entities. Therefore, mining the data as a single dataset incorporating all the information from individual sources poses an important challenge. Furthermore, the issue of *data privacy* is another challenge when the aforementioned data sources include sensitive information that may be used for the purpose of mining the data but cannot, under any circumstance, be related to individ-

uals in the resulting model.

- **Non traditional analysis:** The traditional statistical approach focuses mainly on the testing of hypotheses, rather than generating them from the data as modern data mining tasks attempt to achieve. In order to effectively discover knowledge in the data a data mining method needs to generate and test a large number of hypotheses/models. The main areas of data mining, as mentioned in Section 1.1, cover the traditional research problems but modern analysis has diversified itself due to the incorporation of non traditional data types as well as non traditional expectations from the results. Section 1.2.1 describes such a task that this thesis attempts to address.

1.2.1 The Data Characterization Challenge

In the example of Section 1.1 the expectation is the ability to effectively classify any future iris specimen into one of the given classes. However, as explained the nature of data mining has evolved and there are tasks that go beyond the scope of predictive modelling. Large historical data collections cannot always be modelled with sufficient accuracy, or due to the volume of the data it is possible that constructing a complete model of the data is not realistic. In these cases the desired data mining output is a set of hypotheses about specific data areas that can be tested and verified by domain experts in order to gain knowledge on the process.

Improving a process relies on a good understanding of it and the data used to monitor it. *Causality* is the relation between an event and a phenomenon, referred to as the effect [34], and in order to effectively comprehend a phenomenon and either replicate or avoid it is necessary to comprehend its underlying mechanisms.

Consider the example of a high performance race car. Testing as well as racing provides a team of engineers with a large volume of data regarding the car's performance. Due to time limitations as well as the lack of established knowledge for the novel, cutting edge technologies employed it is not realistic to develop a complete model of the car's behavior and consequently the optimal performance. The only realistic option is to identify positive scenarios, as data areas, and study them in order to gain knowledge on the conditions that can potentially improve overall performance. Alternatively, improvement may be achieved by identifying negative, or undesirable scenarios and avoiding the conditions that constitute the

underlying causes. Note how we refer to improved, not optimal, performance as there is no such guarantee in this case. Figure 1.3 graphically represents the extraction of such data areas.

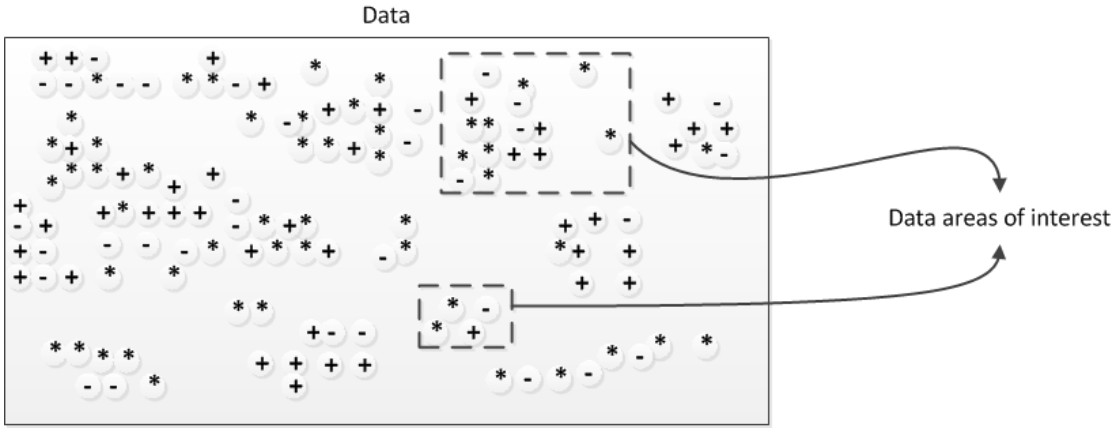


Figure 1.3: Mining data areas of interest.

In the given figure the data samples have been classified in three different classes $\langle +, -, * \rangle$ for $\langle positive, negative, average \rangle$ performance respectively. In the case of mining continuous attributes these areas are represented as classification rules of associated numerical ranges.

The distinctive difference in data characterization is that the value of the result is not based on its effectiveness of predicting new, previously unseen, data but improving the understanding of the existing dataset. In the classification context the user asks for a data model that can be applied to a new, unlabeled dataset, in characterization the desired output needs to be applicable to the training data and provide information about the original dataset. This is because in characterization we refer to users that aim to improve an underlying process and not only predict future output.

1.3 Research Contributions

Mining a single optimal numerical range with regards to an interest measure can be mapped to the *max-sum* problem. For a single attribute a solution using dynamic programming can be computed in $O(N)$ [9]. When the desired optimal range is over two numerical attributes then the problem is *NP-hard* [41, 42]. The problem of mining multiple continuous ranges that optimize a given criterion can

be mapped to the *subset sum* problem that is *NP-complete* [23]. This thesis' contribution is a scalable, effective solution that employs a heuristics-based approach. The presented methodology addresses datasets of continuous attributes and mines continuous ranges without any form of pre-processing of the data.

The primary contribution, however, of the developed algorithm is to the non traditional analysis challenge, focusing on addressing the data characterization problem. The solution described in the following chapters mines multiple rules of associated numerical ranges from continuous attributes that can be used in a data characterization scenario. Due to the nature of the problem the algorithm developed is using user-set thresholds, for the interest measures employed, as input. The presented solution is flexible, allowing for users to tune the input thresholds to different values that describe the targeted knowledge.

Extensive experiments demonstrate the effectiveness of the presented solution as a classification algorithm as well as in the area of data characterization. Furthermore, the distinctive differences in evaluating prediction accuracy and characterization effectiveness are described and employed in a comparison that demonstrates the advantages of the developed solution.

1.4 Thesis Organization

The remaining chapters of this thesis are structured as follows.Chapter 2 reviews work in the related associative classification literature. Research on mining continuous datasets as well as related rule association methods are analysed. Furthermore, an overview of existing interestingness measures for mining classification rules is presented as well as a review of existing solutions that have or could potentially address the data characterization problem.

Chapter 3 introduces important definitions and key concepts for the presented methodology. Furthermore, the concept of bounded ranges is presented as a method for reducing search space. The general algorithm is described for identifying ranges from the continuous attributes and incrementally building them into range-based rule candidates. Chapter 3 includes the concept and formal definition of a novel interestingness measure designed specifically for continuous numerical ranges. Also, the usage of a custom data structure for storing the candidates is given as well as a presentation of the concept of splitting candidates into range-based rules.

Following the general description, Chapter 4 presents four(4) different criteria for splitting rule candidate rules into the resulting range-based classification rules. The significance and exact methodology for each heuristic are explained along with the strengths and weaknesses of each method.

Chapter 5 presents an evaluation of the developed solution compared to existing algorithms. A comparative study is given for how the newly introduced interest measure affects classification outcome on a series of datasets. The experimental results for predicting unlabeled data using the described methods are provided and compared against established rule mining solutions implemented in *Weka* [51]. Chapter 5 also includes a series of experiments based on key aspects of the characterization problem that demonstrate how the presented solution addresses these tasks.

Finally, Chapter 6 concludes the thesis and discusses future directions for extending the presented work.

Chapter 2

Background Work

In this chapter, we present an overview of the literature on range-based classification rule mining and discuss how existing techniques on related problems compare to ours.

2.1 Range-Based Classification Rule Mining: Concepts and Algorithms

In order to classify related work in the area of associative classification we use the three main aspects of each approach as described below.

- The nature of the input data.
- The approach for mining the rules.
- The measures used for evaluating the rules.

The nature of the input data refers to the different types of datasets and their characteristics. Two different types of data are *categorical* and *numerical/continuous* whereas an example of data with special characteristics are time series data due to the sequential relationship between data values. This chapter discusses methods designed to handle different data types but focuses primarily on methods capable of mining continuous data. The mining approach refers to the method used for generating the rules whereas the evaluation measures concern the different criteria employed in each case for determining each rule's importance which we refer to as *interestingness*.

The problem of deriving range-based classification rules can be viewed as a problem of *supervised discretization* which we analyze in Section 2.2. In Section 2.3 we present existing work on mining range-based association rules whereas in section 2.4 we examine work in the area of *associative classification* which covers methods that incorporate characteristics of both *association rule mining* and *classification*.

A more detailed overview of the research space of associative classification can be seen in Figure 2.1.

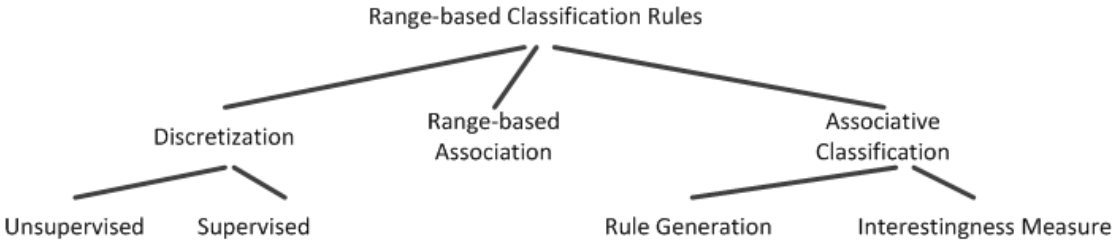


Figure 2.1: The research space of association mining.

Other approaches related to association mining that do not address the associative classification problem, and are therefore not so closely related to our work are reviewed in Section 2.6.

2.2 Discretization

One way to address the problem of mining continuous data is by *discretizing* the continuous attributes. *Discretization* is the process of transforming a continuous attribute into a categorical one [67, 77]. A special form of *discretization* is *binarization* when continuous and categorical attributes are transformed into one or more binary attributes.

Consider the database in Table 2.1 which represents a bank's clients, their accounts' balance and other useful information. The table attributes *Checking Account* and *Savings Account* represent continuous features and could all be discretized. In Table 2.2 you can see the database after the attributes *Checking Account* and *Savings Account* have been discretized.

In the aforementioned example a simple *manual* method of discretization is shown in order to demonstrate the differences with Table 2.1. The hypothesis, in this case, is that the organization holding the data chooses to group clients in pre-specified ranges because, for example, it is believed that all clients with a checking account

ClientID	Checking Account	Savings Account	Loan
C_1	2003.15	2000.0	long term
C_2	0.0	100.3	short term
C_3	56087.5	125000.0	-
C_4	127.3	0.45	long term
C_5	-345.2	5250.5	-
C_6	11023.04	0.0	short term
C_7	19873.6	22467.4	long term
C_8	4187.1	0.0	-
C_9	4850.36	445.2	short term
C_{10}	8220.4	3250.12	long term

Table 2.1: Bank clients database.

ClientID	Checking Account	Savings Account	Loan
C_1	$(2000, 5000]$	$(1000, 5000]$	long term
C_2	$[0, 1000]$	$[0, 1000]$	short term
C_3	$(50000, 100000]$	$(100000, 250000]$	-
C_4	$[0, 1000]$	$[0, 1000]$	long term
C_5	$[-1000, 0)$	$(5000, 20000]$	-
C_6	$(5000, 20000]$	$[0, 1000]$	short term
C_7	$(5000, 20000]$	$(20000, 50000]$	long term
C_8	$(2000, 5000]$	$[0, 1000]$	-
C_9	$(2000, 5000]$	$[0, 1000]$	short term
C_{10}	$(5000, 20000]$	$(1000, 5000]$	long term

Table 2.2: Bank clients database with account balance discretized.

balance in the range $(50000, 100000]$ have the same characteristics. Curved brackets are used when the corresponding value is not included in the range whereas squared brackets are used for values that are included in the range. For example, a value of 50000 is *not* included in the range $(50000, 100000]$ but 100000 is. In certain cases, discretization is a potential inefficient bottleneck, since the number of possible discretizations is exponential to the number of interval threshold candidates within the data domain [32]. There have been attempts to address the difficulty of choosing an appropriate discretization method given a specific data set [17, 29].

Data can be *supervised* or *unsupervised* depending on whether it contains class information. Consequently, *supervised discretization* uses class information while *unsupervised discretization* does not. Unsupervised discretization can be seen in early methods that discretize data in bins of either *equal-width* or *equal-frequency*. This approach does not produce good results in cases where the distribution of

the continuous attribute is not uniform and in the presence of outliers that affect the resulting ranges [16]. Supervised discretization methods were introduced where a class attribute is present to find the proper intervals caused by cut-points. Different methods use the class information for finding meaningful intervals in continuous attributes. Supervised and unsupervised discretization have their different uses although the application of supervised discretization requires the presence of a class attribute. Furthermore, the application of an unsupervised discretization method requires a separate step of mining rules from the discretized data which is why we focus, primarily, on supervised classification. Discretization methods can also be classified based on whether they employ a *top-down* approach or *bottom-up*. Top-down methods start with the full range of a continuous attribute and attempt to gradually *split* it in smaller ones as they progress. Bottom-up methods start with single values and attempt to gradually build larger numerical ranges by gradually *merging* them. Because of this we also refer to top-down methods as *split-based* whereas we refer to *bottom-up* methods as *merge-based*. A representation of the classification space for a discretization method can be seen in 2.2.

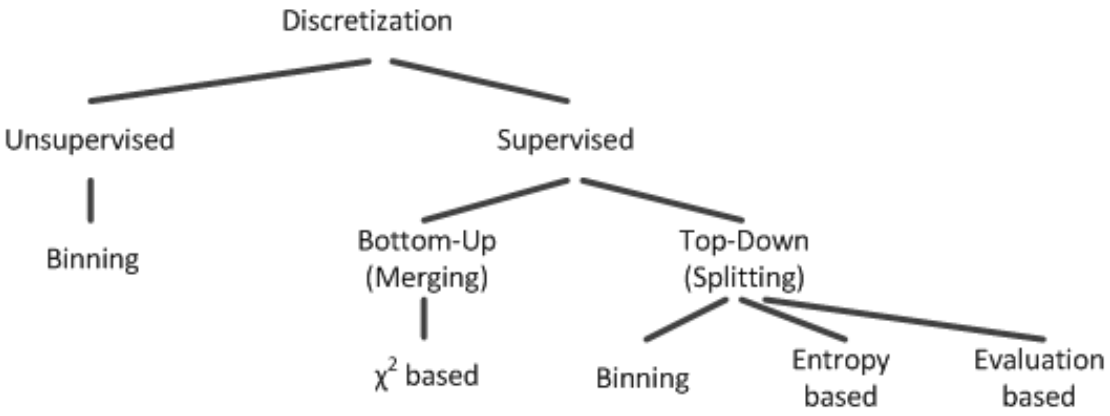


Figure 2.2: The research space of discretization.

2.2.1 Unsupervised Discretization

Binning is the method of discretizing continuous attributes into a specified number of bins of *equal width* or *equal frequency*. Regardless of which method is followed, the number of bins k needs to be given as input. Each bin represents a distinct discrete value, a numerical range. In equal-width, the continuous range of a feature is divided into intervals of equal-width, each interval constitutes a bin.

In equal-frequency, the same range is divided into intervals with an equal number of values placed in each bin.

One of the major advantages of either method is their simplicity but their effectiveness relies heavily on the selection of an appropriate k value. For example, when using equal-frequency binning, many occurrences of the same continuous value could cause the same value to be assigned into different bins. A solution would be a post-processing step that merges bins that contain the same value but that also disturbs the equal-frequency property. Another problem is data that contain outliers with extreme values that require the removal of these values prior to discretization. In most cases, equal-width binning and equal-frequency binning will not result in the same discretization [77].

2.2.2 Supervised Discretization

Unsupervised binning is meant to be applied on data sets with no class information available. However, there are supervised binning methods that address this issue. $1R$ [56] is a supervised discretization method that uses binning to divide a range of continuous values into a number of disjoint intervals and then uses the class labels to adjust the boundaries of each bin. The width of each bin is originally the same and must be specified before execution. Then each bin is assigned a class label based on which class label is associated with the majority of values in the given bin, let that class label be C_m. Finally, the boundaries of each adjacent bin are checked and if there are any values that are also associated with C_m they are merged with the given bin. $1R$ is comparatively simple to unsupervised binning but does not require the number of bins to be pre-specified, it does however, require the definition of an initial width for each bin. Another way has been developed for improving equal-frequency by incorporating class information when merging adjacent bins, by using *maximum marginal entropy* [29]. In spite of their ability to include the additional information neither approach has been shown to have better results than unsupervised binning when used for the mining of classification rules [77].

A statistical measure often employed in supervised discretization is **chi-square** χ^2 by using the $\chi^2 - test$ between an attribute and the classifier. Methods that use this measure try to improve discretization accuracy by splitting intervals that do not meet this *significance level* and merging adjacent intervals of similar class frequency [65]. There is a top-down approach, *ChiSplit*, based on χ^2 that searches

for the best split of an interval, by maximizing the chi-square criterion applied to the two sub-intervals adjacent to the splitting point and splits the interval if both sub-intervals differ statistically. Contrary to *ChiSplit*, *ChiMerge* [65] performs a local search and merges two intervals that are statistically similar. Other approaches continue to merge intervals while the resulting interval is consistent [78] or have attempted to apply the same criterion on multiple attributes concurrently [110]. Instead of locally optimizing χ^2 other approaches apply the $\chi^2 - test$ on the entire domain of the continuous attributes and continuously merge intervals while the confidence level decreases [11].

Another measure used to evaluate ranges is **entropy**. An entropy-based method detects discretization ranges based on the classifier's entropy.Class information entropy is a measure of purity and it measures the amount of information which would be needed to specify to which class an instance belongs. It is a top-down method that recursively splits an attribute in ranges while a stopping criterion is satisfied (e.g. a total number of intervals). Recursive splits result in smaller ranges, with a smaller entropy so the stopping criterion is usually defined to guarantee a minimum number of supported instances. An approach proposed by Fayyad et al. examines the class entropy of the two ranges that would result by splitting at each midpoint between two values and selects the split point which minimizes entropy [35]. When the size of the resulting ranges does not meet the *minimum description length (MDL)* the process stops. It has been shown that optimal cut points must lie between tuples of different class values [35, 54], a property that we also use in our approach. Other approaches attempt to maximize mutual dependence between the ranges and the class label and can detect the best number of intervals to be used for the discretization [20]. In [50] the authors merge the individual continuous values in ranges of maximum *goodness* while trying to maintain the highest average-goodness. This results in an efficient discretization but does not guarantee any of the desired classification properties for the resulting ranges. Finally, *CAIM* [68] uses class-attribute interdependence in order to heuristically minimize the number of discretized ranges. The authors demonstrate that this approach can result in a very small number of intervals, however unlike discretization, in range-based classification a small number of ranges is not desired when there are class values of low support.

Unlike discretization algorithms that are developed for the pre-processing step of a data mining algorithm there are methods that are designed to alter the numerical intervals based on the performance of an induction algorithm [15, 108].

In some cases algorithms are developed to adapt the ranges so as to minimize *false-positives* and *false-negatives* on a training data set [18, 81]. One important limitation, however, is the need for a predetermined number of intervals. In addition to techniques that consider error-rate minimization there are methods that integrate a cost function to adapt the cost of each prediction error to specific problems [60]. It has been shown that in order to determine the optimal discretization that maximizes class information one only has to check a small set of candidate data points, referred to as *alternation points*. However, failure to check one alternation point may lead to suboptimal ranges [31]. This idea, has been extended to develop very efficient top-down discretization solutions given an evaluation function [32] by removing any potential split-points that are proven suboptimal, from the algorithms search space.

More recently, research in discretization has focused on **adaptive discretization** approaches. *Adaptive Discretization Intervals (ADI)* is a bottom-up approach that can use several discretization algorithms at the same time which are evaluated to select the best one for the given problem and a given data set [6]. ADI has also been extended to use heuristic non-uniform discretization methods within the context of a *genetic algorithm* during the evolution process when a different discretization approach can be selected for each rule and attribute [7]. The concept of adaptive classification during the evolution process of a genetic algorithm has been researched extensively [27, 46]. A comparison of the most well-known approaches can be found in [5].

2.3 Range-based Associations

This section examines existing solutions that address the problem of mining range-based rules from continuous data.

One of the first approaches at mining association rules from a data set that includes both continuous and categorical attributes was in [102]. The aforementioned solution mines a set of association rules but requires an *equi-depth* partitioning(discretization) of the continuous attributes. Therefore, the desired ranges can only result from merging these partitions and the original problem is mapped to a boolean association rules problem. Table 2.3 demonstrates an example of equi-depth partitioning of Table 2.1, note that the number of bins used for the discretization do not have to be the same for every attribute.

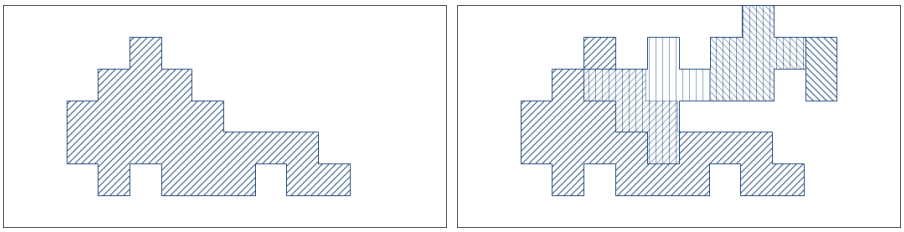
ClientID	Checking Account	Savings Account	Loan
C_1	$[-1000, 4708.75]$	$[0, 12500]$	long term
C_2	$[-1000, 4708.75]$	$[0, 12500]$	short term
C_3	$(50378.75, 56087.5]$	$(112500, 125000]$	-
C_4	$[-1000, 4708.75]$	$[0, 12500]$	long term
C_5	$[-1000, 4708.75]$	$[0, 12500]$	-
C_6	$(10417.5, 16126.25]$	$[0, 12500]$	short term
C_7	$(16126.25, 21835]$	$(12500, 25000]$	long term
C_8	$[-1000, 4708.75]$	$[0, 12500]$	-
C_9	$(4708.75, 10417.5]$	$[0, 12500]$	short term
C_{10}	$(4708.75, 10417.5]$	$[0, 12500]$	long term

Table 2.3: An equi-depth discretization of **Checking Account balance** and **Savings Account balance** into 10 bins.

Given Table 2.3 and two associations $CheckingAc \in [-1000, 4708.75] \wedge SavingsAc \in [0, 12500]$ and $CheckingAc \in (4708.75, 10417.5] \wedge SavingsAc \in [0, 12500]$ we can merge them into one rule that the authors refer to as *super-rule* $CheckingAc \in [-1000, 10417.5] \wedge SavingsAc \in [0, 12500]$. As you can see from this example, this approach results in association rule mining where the produced ranges depend on the discretization criteria, therefore, experimentation is required for producing appropriate bins that will give good results introducing an important limitation to this methodology. The approach of discretizing continuous data into ranges as a preprocessing step is a very popular one. In [64] a method is presented for mapping pairs of attributes to a graph based on their *mutual information (MI)* score. This method is based on the assumption that all interesting associations must have a high *MI* score and therefore must belong in the same clique in the graph. In some cases, however, researchers have presented the problem of mining associations from data that is already stored in the form of numerical ranges [30] which is essentially directly comparable to the aforementioned methods after the discretization phase.

Different approaches have been described for directly generating numerical ranges from the data. Given a numerical attribute and a categorical class label in [39] the authors present an approach to solving two different problems. Mining the range of the numerical attribute with the maximum support, given a confidence threshold and mining the corresponding range with the maximum confidence given a support threshold. Unlike previously described solutions, this approach includes user defined thresholds for support and confidence, although for separate problems. Furthermore, it is only applicable to one attribute, or data sets of more

attributes that only contain a single continuous attribute. An extension of this was described in [41, 42] where the solutions presented mine optimal ranges from two continuous attributes with a boolean attribute as the consequent. By using dynamic programming the aforementioned solution is able to mine the numerical range of maximum *gain*, an interest measure described in more detail in 2.5.1, but the result is a single region of optimal gain. Therefore, the aforementioned approach does not address the problem of mining all range-based classification rules even in the problem space of two continuous attributes. The difference between the desired output in a two-dimensional problem space is represented in Figure 2.3.



(a) Single region of optimal gain. (b) Multiple admissible ranges.

Figure 2.3: Single optimal region compared to multiple admissible regions.

The problem of mining a single optimal gain region has been extended to mining an approximation of the k optimal gain regions, as the problem of mining the k regions of optimal gain is *NP-hard*, from a data set that contains both continuous and categorical attributes [14]. The proposed method still relied in a pre-processing step that places contiguous values with the same class label in the same bucket and was still limited to a total of two attributes excluding the class label. In [72] the authors also limit the problem space to two dimensions but take a different approach, they develop a method for mining the most dense ranges, that is the ranges with the most data points within the range, from the data. The concept of data density is, actually, of particular interest when dealing with continuous data ranges as we explain in Chapter 3. However, there is no evidence to support that density by itself is a sufficient evaluation measure.

Other approaches have considered mining range-based rules as an optimization problem and proposed solutions using a genetic algorithm. The mined rules are associations between numerical ranges of attributes and not classification rules. In [83] a solution is presented that uses an evolutionary algorithm based on a fitness function that improves the generated ranges between generations. This solution is able to mine overlapping ranges of high support but offers poor results

in term of confidence. *Quantminer* [97] is a genetic-based algorithm that delivers better results in terms of confidence and a reduced number of rules. The proposed solution is efficient but offers no guarantees that its solutions will meet certain thresholds.

2.4 Associative Classification

Classification rule mining and association rule mining are two popular data mining tasks with distinct differences. Classification rule mining aims to discover a small set of rules in the data to form an accurate classifier [92] whereas association rule mining aims to find all the rules that satisfy some predetermined constraints [4]. For association rule mining, the target of mining is not predetermined, while for classification rule mining the rule target can only be a specific attribute, the class. Both classification rule mining and association rule mining are useful to practical applications, therefore integrating the two tasks can be of great benefit. The integration is achieved by focusing on a special subset of association rules whose consequent is restricted to the class attribute. We refer to these rules as *class association rules* (CARs) [74]. There are researchers, however, that have expressed the opinion that due to the fact that associative classification techniques are often evaluated based on prediction effectiveness that it is, essentially, another form of classification [37]. However, even though prediction is often used to evaluate associative classification methods it is not the only method. Moreover, we view associative classification as a powerful method that is capable of addressing more problems than just classification, especially when applied in real world problems.

An approach for mining associative classification rules from categorical data is presented in [107] that seems to perform well compared to existing solutions. Early approaches have also adopted a simple technique which relied on a two-step process of mining associations from the data and then ranking the resulting rules based on a selected evaluation measure. The top ranking rules were considered to be the resulting associative classification rules as demonstrated in Figure 2.4.

Figure 2.4: Associative classification mining by filtering of association rules.

This category of methods relies heavily on the selected evaluation measure and produces variable results depending on the data set [63, 86]. In some cases heuristics are applied to actively reduce the number of generated rules making the mining algorithm more efficient [115]. Alternatively, other researchers have addressed the problem of associative classification as a problem of modeling the data using *R-Trees* based on the different class labels and then trimming the resulting models in order to avoid overfitting and reduce the number of rules [71]. These solutions prioritise classification performance and aim to increase data coverage with the fewest rules possible. Besides being applied on categorical data only, these algorithms avoid the mining of less general but more specific rules that may represent important data characteristics as discussed in Section 1.2.1.

One popular method that can be employed for mining range based classification rules is the *C4.5* algorithm [92]. This is a partitioning based technique that only looks for the best halving of the domain with regards to a specific class. It has a quadratic time complexity for non numeric data that increases by a logarithmic factor when numerical attributes are processed. *C4.5* is inefficient when dealing with strongly correlated numerical attributes. In order to improve this an approach has been proposed that given a numeric attribute and a boolean class attribute it can produce a more efficient branching for the given attribute [40]. Unlike the original *C4.5* algorithm , in the latter solution the attribute values are pre-discretized in a user set number of ranges and the discrete ranges are merged in order to produce an optimal branching that minimizes total entropy. Although the described approach has proven to improve the size of the tree and can be extended to work with class labels of more than two discrete values (non boolean), its outcome relies on the user selecting an appropriate number of buckets for the discretization whereas its classification accuracy is not evaluated. Furthermore, the direct output of these algorithms is a *Decision Tree (DT)* that requires further processing in order to transform the results into range-based classification rules. Even with the additional post-processing, however, there is no guarantee that the resulting rules meet any specific requirements (e.g. user specified thresholds).

2.5 Rule Evaluation

In this section we present measures that have been used by the research community for the evaluation of classification rules, which we refer to as *interestingness*

measures. We include measures that have originally been developed for *association rule mining* but can also be used to evaluate classification rules. We focus on *interestingness measures* that are applicable in the evaluation of *range-based rules* and split them in two main categories: *objective measures* which we describe in Section 2.5.1 and *subjective measures* which we describe in Section 2.5.2.

2.5.1 Objective Measures

An objective measure is based only on the raw data without considering any existing user knowledge or requiring application knowledge. Therefore, objective measures are based on probability theory, statistics, or information theory. In this section we examine several objective measures for an association rule $X \Rightarrow Y$, where X is the rule antecedent and Y the rule consequent. We denote the number of data records covered by the rule as $cardinality(XY)$ whereas N denotes the total number of data records/tuples.

Support and *confidence* are the most popularly accepted interestingness measures used for discovering relevant association rules and are also commonly used for evaluating associative classifiers. Although, in many cases, they are appropriate measures for building a strong model they also have several limitations that make the use of alternative interestingness measures necessary. Researchers have examined the utility of retaining support and confidence as evaluation measures while adding new ones [43].

$$Support(X \Rightarrow Y) = P(XY)$$
$$Confidence(X \Rightarrow Y) = P(X|Y)$$

Support is used to evaluate the *generality* of a rule, how many data records it covers whereas confidence is used to evaluate a rule's *reliability*. In literature, *coverage* has also been used to evaluate rule generality whereas *lift* and *conviction* [13] have been proposed as alternatives to confidence for evaluating rule reliability.

$$Coverage(X \Rightarrow Y) = P(X)$$
$$Conviction(X \Rightarrow Y) = \frac{P(X)P(\neg Y)}{P(X\neg Y)}$$
$$Lift(X \Rightarrow Y) = P(X|Y)P(Y)$$

Generality and reliability are both desired properties for a rule but using two different measures to evaluate them often leads to contradiction. As a result researchers have proposed measures that evaluate both resulting in a single ranking. Such a measure is the *IS measure* [85] which is also referred to as *cosine measure* since it represents the cosine angle between X and Y. In [90] the authors propose *leverage* that measures the difference of X and Y appearing together in the data set compared to what would be expected if X and Y were statistically dependent. Other measures bases around these criteria include *Jaccard* [105], *Klosgen's measure* [66] and *two-way support* [114].

$$Jaccard(X \Rightarrow Y) = \frac{P(XY)}{P(X) + P(Y) - P(XY)}$$
$$Klosgen(X \Rightarrow Y) = \sqrt{P(XY)} \times (P(Y|X) - P(Y))$$
$$Leverage(X \Rightarrow Y) = P(Y|X) - P(X)P(Y)$$
$$Two - Way\ Support(X \Rightarrow Y) = P(XY)\log_2 \frac{P(XY)}{P(X)P(Y)}$$

More specifically, in the area of association mining of transactional data researchers have argued for the use of traditional statistical measures that mine correlation rules instead of associations [75, 101]. Other researchers have attempted to mine strong correlations by proposing new measures, like *collective strength* which can indicate positive, as well as negative correlation [2, 3]. One drawback of collective strength is that for items of low probability the expected values are primarily influenced from transactions that do not contain any items in the evaluated itemset/rule and gives values close to one(1) which falsely indicate low correlation. A different approach, as demonstrated in [98], is to calculate the *predictive accuracy* of a rule while mining the data by using a *Bayesian frequency correction* based on the training data distribution. The authors propose the elimination of a minimum threshold for support and instead only require the user to specify the n preferred

number of rules to be returned.

$$Collective\ Strength(X \Rightarrow Y) = \frac{P(XY) + P(\neg Y|\neg X)}{P(X)P(Y) + P(\neg X)P(\neg Y)} * \frac{1 - P(X)P(Y) - P(\neg X)P(\neg Y)}{1 - P(XY) + P(\neg Y|\neg X)}$$

An interestingness measure that was proposed specifically for evaluating range-based classification rules is *gain* [41, 42], not to be confused with *information gain*. Gain is used to evaluate the benefit from including additional data records to a rule after the given confidence threshold θ has been met.

$$Gain = cardinality(XY) - \theta * cardinality(X)$$

Some researchers have proposed studying the relationship between support and confidence by defining a partially ordered relation based on them [8, 113]. Based on that relation any rule r for which we cannot find another rule r' with $support(r') \geq support(r)$ and $confidence(r') \geq confidence(r)$ is an *optimal* rule [8]. This property, however, can only be applied to an interestingness measure that is both monotone in support and confidence and results in the single best rule according to that measure.

In the area of classification rule mining the role of interestingness measures is to choose the attribute-value pairs that should be included, a process defined as *feature selection* [84]. The concepts of generality and reliability are also applicable in classification rule mining. A classification rule should be as accurate as possible on the training data and as general as possible so as to avoid *overfitting* the data. Different measures have been proposed to optimize both these criteria. *Precision* [89] is the equivalent of confidence whereas popular measures for feature selection include *entropy* [91], *gini* [12] and *Laplace* [21].

$$Gini(X \Rightarrow Y) = P(X)(P(Y|X)^2 + P(\neg Y|X)^2) + P(\neg X)(P(Y|\neg X)^2 + P(\neg Y|\neg X)^2) - P(Y)^2 - P(\neg Y)^2$$

$$Laplace(X \Rightarrow Y) = \frac{cardinality(XY) + 1}{cardinality(X) + 2}$$

In [44] the authors have shown that the gini and entropy measures are equivalent to precision since they produce equivalent (or reverse) rankings for any set of rules.

All the objective interestingness measures proposed for association rules can also be applied directly to classification rule evaluation, since they only involve the probabilities of the antecedent of a rule, the consequent of a rule, or both, and they represent the generality, correlation, and reliability between the antecedent and consequent. However, when these measures assess the interestingness of the mined rules with respect to the training data set and do not guarantee equivalent results when used for the classification of previously unseen data. Furthermore, with the exception of gain, the aforementioned interestingness measures have all been developed for the purpose of mining rules of categorical attributes and need to be appropriately redefined for range-based rules when that is possible.

Due to the large number of different measures used in association mining, researchers have also focused on methodology for selecting the appropriate interestingness measures for a given research problem [87, 105]. Specifically in the area of classification rules researchers have proved that there are measures that can significantly reduce the number of resulting rules without reducing the model's accuracy but their comparative performance is dependent on the different data sets and there cannot be a clear recommendation of a single measure [59]. Finally, researchers have also defined objective measures that evaluate rules based on their format [28, 38]. These measures, however, are not applicable when mining continuous attributes.

2.5.2 Subjective Measures

A subjective measure takes into account both the data and the user. The definition of a subjective measure is based on the user's domain or background knowledge about the data. Therefore, subjective interestingness measures obtained this knowledge by interacting with the user during the data mining process or by explicitly representing the user's knowledge or expectations. In the latter case, the key issue is the representation of the user's knowledge, which has been addressed by various frameworks and procedures in the literature [73, 76, 95].

The purpose of subjective interestingness measures is to find unexpected or novel rules in the data. This is either achieved by a formal representation of the user's existing knowledge and the measures are used to select which rules to present to

the user [73, 76, 99, 100], through interaction with the user [95] or by applying the formalized knowledge on the data and reducing the search space to only the interesting rules [88]. Even though useful, subjective interestingness measures depend heavily on the user's knowledge representation and have no proven value in the area of classification rule mining.

2.6 Other Related Methods

As described in 2.1 there are techniques that do not address the problem of range-based classification rule mining, but closely related areas like that of association rule mining of categorical data and therefore present many concepts and methods that are of interest.

In [70] the authors present an alternative to association rule mining that mines only a subset of rules called *optimal rules* with regards to an interestingness measure that is interchangeable. An optimal rule r is a rule for which there is no other rule in the resulting set that covers a superset of the tuples covered by r and is of higher interest. Even though, this approach improves efficiency significantly by reducing the number of generated associations, optimality is only defined with regards to a single evaluation measure which, as we have seen in Section 2.5 is not always the case. In our work we have modified this principle to apply to more than one evaluation measures.

The use of continuous attributes and the problems they present has also been examined in a different context. Researchers have considered the use of continuous attributes when actually mining categorical data in order to represent specific data properties. One such example is the use of an additional continuous attribute to represent the *misclassification cost* of each record or each different class. In [79] the authors extend this concept to misclassification costs that are actually numerical ranges. Another related area is *probabilistic databases* [103] where each record is associated with a probability of occurrence that is, obviously, a continuous attribute. In [109] the authors attempt to mine association rules from a transactional data base of market basket data but by also evaluating the numerical ranges of quantities purchased for each item.

2.6.1 Performance Optimization Methods

In recent years, the improvement of existing algorithmic solutions through parallel executions has generated great interest due to the significant progress in the related field. Therefore, some of the most noticeable work that could have a major impact on associative classification rule mining is in the area of performance improvements through parallel execution. One such framework, shown to significantly speed up data mining algorithms is *NIMBLE* [45], which is a java framework implemented on top of the already established tool *MapReduce*.

2.7 The Data Characterization Problem

Some of the issues we are trying to address with our approach go beyond what traditional associative classification rule mining deals with, that is either classification accuracy and rule interestingness. In the process of studying the problem of associative classification we have identified several key points regarding the desired output:

- Readability: The resulting rule set should be easy to understand without requiring knowledge of the underlying methodology that produced the results.
- Interpretability: People who are considered experts in the field that the data set belongs to, should be able to recognize the value of the presented rule and be able to evaluate them based on their experience, if necessary through the use of their own process-based measures.
- Causality: The goal of knowledge extraction from a given data set is of reduced value if the resulting rules cannot be traced back to a source that created the pattern, therefore identifying the underlying reasons for the rule to exist.

These issues have been identified by other researchers in the areas of association and classification rule mining and constitute areas of interest that are not addressed by state-of-the-art classification systems that only aim to generate a function that maps any data point to a given label with high accuracy [112]. Without denying the importance of highly efficient classifiers, a complete associative classification approach serves the purpose of knowledge discovery. The authors in [94] have attempted to address the issue by defining properties and then incorporat-

ing these properties in the reported results through a newly defined rule format. Another concept that researchers have found is not being addressed by existing solutions in the area of data mining is *causality* [111]. Some researchers have considered the problem of mining rules from ranked data records [96]. This method relies on ranking the data tuples according to a given criterion and then proceeds to mine *characterization* association rules that are more often present in the top ranking tuples than in the bottom ones. In a class labeled data set the presence of a ranking criterion is not necessary as the class labels themselves play the role of the desired property. Furthermore, the aforementioned method does not deal with numerical ranges but, nevertheless, presents an excellent argument for the importance of generating meaningful rules where the investigation of causality is possible.

The method presented in this thesis addresses the points of readability and interpretability by mining rules that are presented in a clear, readable format that only requires basic knowledge of the meaning of the data attributes and the corresponding values. Furthermore causality investigations are possible since the mined rules are independent, making it possible for domain experts to select specific rules to use for their investigation in addition to the users being able to tune the number and quality of mined rules by providing different minimum thresholds as parameters. The aforementioned criteria, however, are not the equivalent of interestingness measures as they are not expressed as measurable quantities. This is because any measure presented would rely on domain specific knowledge.

2.8 Summary

This chapter presented techniques in the existing data mining work that are related to the area of mining range based classification rules. The literature surveyed is examined based on whether real values of continuous attributes are mined directly, the form of the rules resulting from the mining method and finally the evaluation measures employed in each method. Other work that relates to the associative classification problem is also presented as well as literature on data characterization tasks or a related problem.

Supervised discretization methods have been surveyed that result in a clustering of the continuous data space but do not include the concept of a classifier and subsequently do not result in the creation of rules from the data collection. The

review of discretization is important since it is a necessary step for applying most methods developed for associative classification that focus on categorical data. Because of this, these methods rely on a good discretization of the dataset as a preprocessing step and cannot mine ranges from the data directly.

Furthermore an overview of existing interest measures used in rule induction along with a comparative analysis demonstrating that the most efficient measures have focused on addressing the inadequacies of the traditional support/confidence paradigm and evaluating the tradeoff between the two. Reviewing of the work on interest measures reveals the lack of interest measures developed for evaluating rules on continuous attributes and the identifying properties of real values in the data.

Finally, a review is given of work on mining solutions that generate rules but cannot be considered purely association or associative classification solutions. The concepts of data characterization and causal discovery are presented.

Chapter 3

Range-Based Classification Rule Mining

A classification rule mining method is evaluated against specific criteria. The aim is the mining of a set of rules from the data that can achieve a high classification accuracy when predicting new datasets but can also be used to effectively *characterize* a given dataset.

Consider the example in Table 3.1 where the problem in question is to use a dataset of existing bank customers, their accounts' balance, the outstanding debt in their loans and whether or not they have defaulted in their loan payments to mine range-based classification rules like

$$Check.Acc. \in [19873.6, 56087.5] \wedge Sav.Acc. \in [22467.4, 125000] \Rightarrow LoanDef : N$$

Because in a classification context the class attribute and its domain are normally known it is usually not included in the rule description, so the aforementioned rule description changes to

$$Check.Acc. \in [19873.6, 56087.5] \wedge Sav.Acc. \in [22467.4, 125000] \Rightarrow N$$

.

The importance of this rule is determined by two separate things. The first thing is the rule's ability to classify/predict unlabeled data instances, that is clients for who we are trying to determine whether or not they are likely to default on their loan. This is referred to as evaluation of the rule as a classifier [52, 73, 74, 106].

ClientID	Checking Account	Savings Account	Outstanding Loans	Default
C_1	2003.15	2000.0	2800.0	Y
C_2	0.0	100.3	75.8	Y
C_3	56087.5	125000.0	0.0	N
C_4	127.3	0.45	3250.6	Y
C_5	-345.2	5250.5	8725.5	N
C_6	11023.04	0.0	8725.5	N
C_7	19873.6	22467.4	2420.25	N
C_8	4187.1	0.0	575.4	Y
C_9	4850.36	445.2	7230.2	Y
C_{10}	8220.4	3250.12	25225	N

Table 3.1: Bank clients and issued loans data.

The second factor that affects a rule's importance is the ability to use the rule to identify the characteristics of clients who, in the given example, do not default on their loan. The latter allows the investigation of the causes that make specific clients default on their payments.

Section 3.1 contains necessary definitions for terms and concepts used in this chapter. Section 3.2 presents the different measures employed in this thesis to evaluate how each rule addresses the goals described. Finally the proposed methodology is described in Section 3.3.

3.1 Preliminaries

Without loss of generality, we assume that data is contained within a single table $T(A_1, A_2, \ldots, A_m, C)$, where each $A_j, 1 \leq j \leq m$, is a numerical attribute and C is a categorical class attribute. We denote the k-th tuple of T by $t_k = \langle v_{k,1}, v_{k,2}, \ldots, v_{k,m}, c_k \rangle$, where $v_{k,j} \in A_j$, $1 \leq j \leq m$. We may drop c_k from t_k when it is not needed in the discussion.

In the rule example in Section 3, $[19873.6, 56087.5]$ and $[22467.4, 125000]$ are referred to as *ranges*. A formal definition is given below.

Definition 3.1.1 (Range) *Let a and b be two values in the domain of attribute A and $a \leq b$. A* range *over A, denoted by $[a, b]_A$, is a set of values that fall between a and b. That is, $[a, b]_A = \{v | v \in A, a \leq v \leq b\}$.*

Each range covers a certain number of data tuples. In our example these are all the clients whose data values for the corresponding attribute fall within the numerical

range.

Definition 3.1.2 (Cover) *Let $r = [a, b]_{A_j}$ be a range over attribute A_j. r is said to* cover *tuple $t_k = \langle v_{k,1}, v_{k,2}, \ldots, v_{k,m} \rangle$ if $a \leq v_{k,j} \leq b$. We denote the set of tuples covered by r by $\tau(r)$.*

As can be seen in Table 3.1 the ability of a client to pay back their loan is expected to depend on several factors instead of a single one. In theory, it is possible that a single attribute of the table constitutes a unique decisive factor in the repayment of a loan and the fact that a single numerical range would be sufficient to "describe" this phenomenon. This is the case of a direct correlation which is, however, of little interest due to the simplicity of the solution and that direct correlations tend to describe existing expert knowledge. Therefore, it is evident that in order to create accurate rules it is necessary to associate ranges over more than one attribute and create conjunctions that accurately describe the knowledge hidden in the mined data. For the remainder of this thesis the following formal definitions of *associated ranges* and *range-based rules* are used.

Definition 3.1.3 (Associated ranges) *Let $r_1 = [a_1, b_1]_{A_1}, r_2 = [a_2, b_2]_{A_2}, \cdots, r_h = [a_h, b_h]_{A_h}$ be a set of ranges over attributes $A_1, A_2, \ldots, A_h$ respectively. $r_1, r_2, \ldots, r_h$ are* associated ranges *if we have $\tau(r_1) \cap \tau(r_2) \cap \cdots \cap \tau(r_h) \neq \emptyset$.*

Definition 3.1.4 (Range-based rule) *Let $c \in C$ be a class value and $r_1, r_2, \ldots, r_h$ be a set of associated ranges. $r_1 \wedge r_2 \wedge \ldots \wedge r_h \Rightarrow c$ (or simply $r_1, r_2, \ldots, r_h \Rightarrow c$) is a* range-based rule. *We call $r_1 \wedge r_2 \wedge \ldots \wedge r_h$ the rule's* antecedent *and c the rule's* consequent.

3.2 Interestingness Measures

This section presents the formal definitions of the measures employed in the presented solution for the evaluation of the extracted rules. Section 3.2.1 presents the definitions for the support/confidence framework in the context of range-based rules whereas Section 3.2.2 describes a previously undefined measure designed to capture properties that are only relevant when mining continuous data ranges.

3.2.1 Support-Confidence

The support and confidence measures are traditionally used in association rule mining. They are indicative of a rule's *strength* and *reliability* respectively. When support is high it is an indication that the rule does not occur by chance. Furthermore high confidence measures the reliability of the inference made by a rule, given the rule antecedent how likely is the consequent.

Definition 3.2.1 (Support) *Let T be a table and $\lambda : r_1, r_2, \ldots, r_h \Rightarrow c$ be a range-based rule derived from T. The* support *for λ in T is*

$$\sigma(\lambda) = \frac{|\tau(r_1) \cap \tau(r_2) \cap \cdots \cap \tau(r_h)|}{|T|}$$

where $|\cdot|$ denotes the size of a set.

Definition 3.2.2 (Confidence) *Let T be a table and $\lambda : r_1, r_2, \ldots, r_h \Rightarrow c$ be a range-based rule derived from T. The* confidence *for λ in T is*

$$\gamma(\lambda) = \frac{|\tau(r_1) \cap \tau(r_2) \cap \cdots \cap \tau(r_h) \cap \tau(c)|}{|\tau(r_1) \cap \tau(r_2) \cap \cdots \cap \tau(r_h)|}$$

where $\tau(c)$ denotes the set of tuples that have class value c in T.

Example 3.2.1 *Suppose we have the data in Table 3.1 and the rule $\lambda : [0, 4850.36]_{Check.Acc.} \wedge [0, 2000]_{Sav.Acc.} \wedge [75.8, 3250.6]_{Loan\_Out.} \Rightarrow Y$, then we have*

$$\begin{aligned}
\sigma(\lambda) &= \frac{|\tau([0, 4850.36]_{Check.Acc.}) \cap \tau([0, 2000]_{Sav.Acc.}) \cap \tau([75.8, 3250.6]_{Loan_Out.})|}{|T|} \\
&= \frac{|\{t_1, t_2, t_4, t_8, t_9\} \cap \{t_1, t_2, t_4, t_8, t_9\} \cap \{t_1, t_2, t_4, t_7, t_8\}|}{10} \\
&= \frac{4}{10} = 0.4
\end{aligned}$$

$$\begin{aligned}
\gamma(\lambda) &= \frac{|\tau([0, 4850.36]_{Check.Acc.}) \cap \tau([0, 2000]_{Sav.Acc.}) \cap \tau([75.8, 3250.6]_{Loan_Out.}) \cap \tau Y|}{|\tau([0, 4850.36]_{Check.Acc.}) \cap \tau([0, 2000]_{Sav.Acc.}) \cap \tau([75.8, 3250.6]_{Loan_Out.})|} \\
&= \frac{|\{t_1, t_2, t_4, t_8, t_9\} \cap \{t_1, t_2, t_4, t_8, t_9\} \cap \{t_1, t_2, t_4, t_7, t_8\} \cap \{t_1, t_2, t_4, t_8, t_9\}|}{|\{t_1, t_2, t_4, t_8, t_9\} \cap \{t_1, t_2, t_4, t_8, t_9\} \cap \{t_1, t_2, t_4, t_7, t_8\}|} \\
&= \frac{4}{4} = 1
\end{aligned}$$

3.2.2 Density

In example 3.2.1 rule λ achieves a very high confidence. However, continuous data values have certain properties that are not present in categorical data. When mining categorical data there are *itemsets* such as $Red, Green, Blue$ that clearly state that only records with the values $Red, Green$ or $Blue$ support it but when describing a numerical range like $[0.5, 8.2]$ any data value x with $0.5 \leq x \leq 8.2$ is covered by that range. This is because numerical values are by definition ordered which is not true for categorical values, except in cases when a specific order is explicitly defined.

Rule λ has optimal confidence (equal to 1) but looking at customer C_7 it can be seen that the value for outstanding loan debt is actually covered by the range $[75.8, 3250.6]_{Loan\_Out.}$ even though C_7 is not actually covered by rule λ. It is evident therefore that there is another property that needs to be considered when evaluating a range-based rule, its *concentration*, that is how many of the tuples that actually support the ranges support the rule. To address this issue a new interestingness measure is defined, *density*.

Definition 3.2.3 (Density) *Let T be a table and $\lambda : r_1, r_2, \ldots, r_h \Rightarrow c$ be a range-based rule derived from T. The* density *for λ in T is*

$$\delta(\lambda) = \frac{|\tau(r_1) \cap \tau(r_2) \cap \cdots \cap \tau(r_h)|}{|\tau(r_1) \cup \tau(r_2) \cup \cdots \cup \tau(r_h)|}$$

Example 3.2.2 *Suppose we have the data in Table 3.1 and the rule λ as in 3.2.1, then we have*

$$\begin{aligned}
\delta(\lambda) &= \frac{|\tau([0, 4850.36]_{Check.Acc.}) \cap \tau([0, 2000]_{Sav.Acc.}) \cap \tau([75.8, 3250.6]_{Loan_Out.})|}{|\tau([0, 4850.36]_{Check.Acc.}) \cup \tau([0, 2000]_{Sav.Acc.}) \cup \tau([75.8, 3250.6]_{Loan_Out.})|} \\
&= \frac{|\{t_1, t_2, t_4, t_8, t_9\} \cap \{t_1, t_2, t_4, t_8, t_9\} \cap \{t_1, t_2, t_4, t_7, t_8\}|}{|\{t_1, t_2, t_4, t_8, t_9\} \cup \{t_1, t_2, t_4, t_8, t_9\} \cup \{t_1, t_2, t_4, t_7, t_8\}|} \\
&= \frac{4}{6} = \frac{2}{3}
\end{aligned}$$

With these basic concepts and measures defined, new methods for deriving range-based rules that satisfy various properties are introduced in the following sections. For brevity and when the context is clear, range-based rules will simply be referred to as rules in the discussion.

3.3 Methodology

Let T be an $n \times m$ table as defined in Section 3.1 and $r_1, r_2, \ldots, r_k$ are *range-based association rules* derived from T. We call a set of such rules RS a *ruleset*.

The number of associated ranges in T can be very high whereas the majority of these ranges may not be of importance to the user. Therefore, the presented solution allows the user to predefine a minimum threshold for all the measures defined in Section 3.2 that the resulting associated ranges must meet. We denote these thresholds as $\sigma_{min}, \gamma_{min}, \delta_{min}$. Given the table T we aim to extract the ruleset RS that consists of all the range-based association rules that satisfy the given constraints $\sigma_{min}, \gamma_{min}, \delta_{min}$.

This could be described as a *skyline problem*, clearly some rules are inferior to others as but the goal is to find the ones that achieve the best balance between the three measures described in Section 3.2.

3.3.1 Minimum Requirements

One obvious class of rules of interest are those whose support, confidence and density measures are above a certain *minimum threshold*. That is, we wish to find the rules that have some "minimum credibility" from a given data set. Because the actual thresholds for these measures depend on the problem and the particular dataset, the challenge is to find a solution that allows a user to specify these values and be able to adjust them accordingly.

Definition 3.3.1 (Min-$\sigma\gamma\delta$ rule) *A range-based rule λ is said to be a* min-$\sigma\gamma\delta$ rule *if it satisfies the following properties:*

1. $\sigma(\lambda) \geq \sigma_{min}$,
2. $\gamma(\lambda) \geq \gamma_{min}$, *and*
3. $\delta(\lambda) \geq \delta_{min}$

where σ_{min}, γ_{min} and δ_{min} are user specified thresholds.

A naive solution to find all min-$\sigma\gamma\delta$ rules from a given table T is to examine all possible combinations of ranges across all attributes to see if they have sufficient support, confidence and density. This is shown in Algorithm 3.3.1.

Algorithm 3.3.1: *Naive method for finding min-$\sigma\gamma\delta$ rules*

Input: data set $T(A_1, A_2, \ldots, A_m, C)$ and parameters σ_{min}, γ_{min} and δ_{min}

Output: a set of min-$\sigma\gamma\delta$ rules R

1 $R \leftarrow \emptyset$
2 **for** *each* $c \in C$ **do**
3 **for** *each distinct* $\lambda_{ij} : [a_1, b_1]_{A_{i_1}}, [a_2, b_2]_{A_{i_2}}, \ldots, [a_j, b_j]_{A_{i_j}} \Rightarrow c$ **do**
4 **if** $\sigma(\lambda_{ij}) \geq \sigma_{min}$ *and* $\gamma(\lambda_{ij}) \geq \gamma_{min}$ *and* $\delta(\lambda_{ij}) \geq \delta_{min}$ **then**
5 $R \leftarrow R \cup \lambda_{ij}$
6 **return** R

In step 2, each class label c is examined separately. Step 3 generates all possible, but distinct rules

$$\lambda_{ij} : [a_1, b_1]_{A_{i_1}}, [a_2, b_2]_{A_{i_2}}, \ldots, [a_j, b_j]_{A_{i_j}} \Rightarrow c$$

where each A_{i_s}, $s = 1 \ldots j$, is a different attribute in $\{A_1, A_2, \ldots, A_m\}$, each $[a_s, b_s]_{A_{i_s}}$ is a range on A_{i_s}. Then each mined rule λ_{ij} is checked to satisfy the required minimum conditions in steps 3 and 4. Finally, a set of min-$\sigma\gamma\delta$ rules R is returned in step 6.

Algorithm 3.3.1 is exhaustive in nature, i.e. it attempts to find all potentially useful rules. It is, however, not efficient. Assuming that there are m attributes, each attribute has p distinct values on average, and the class attribute has h distinct values. Then Algorithm 3.3.1 will examine $O(h \times (\frac{p(p+1)}{2})^m) \approx O(h \times p^{2m})$ number of rules, which is far too expensive to compute for a non-trivial m.

To improve the performance of Algorithm 3.3.1, we introduce some "more restricted" classes of rules. That is, instead of simply finding all the rules that satisfy some minimal conditions, we will also establish some criteria that make certain rules more desirable than others. This will help prune the search space in computation. The following sections discuss different types of rules and different heuristics that are designed for deriving them.

3.3.2 Consequent bounded rules

The first consideration, are rules whose ranges are "bounded" by the class label in the consequent. We refer to these rules as *consequent bounded rules*. Figure 3.1

below is used to illustrate these rules graphically.

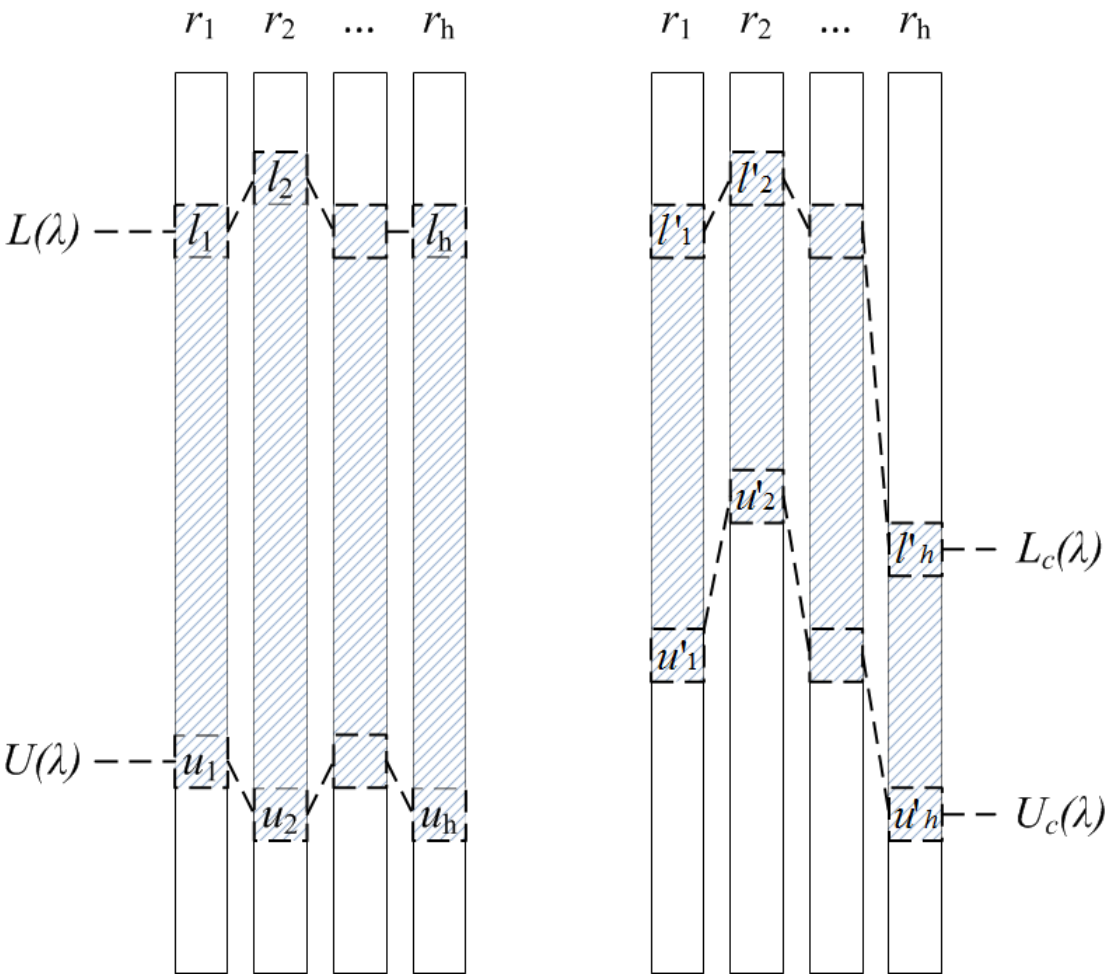


Figure 3.1: Consequent bounded rules

Each rectangular box in Figure 3.1 represents the set of tuples covered by a range. When these ranges jointly form a rule, $\lambda : r_1, r_2, \ldots, r_h \Rightarrow c$, two sets of values in Figure 3.1 are of interest. The $L(\lambda)$ set of values represents the lowest value in each range whose corresponding tuples support the rule, and the $U(\lambda)$ set of values represents the highest such value. That is, tuples that are covered by $r_i, i = 1 \ldots h$, but have values outside $[l_i, u_i]$, cannot support the rule. Thus, these two sets of values effectively form "boundaries"on the ranges within which λ may be supported. More formally,

Definition 3.3.2 (s-boundaries) *Given a rule $\lambda : r_1, r_2, \ldots, r_h \Rightarrow c$, its lower and upper* s-boundaries*, denoted by $L(\lambda)$ and $U(\lambda)$, are:*

- $L(\lambda) = \{l_1, l_2, \ldots, l_h\}$*, and* $l_i = \arg\min_{\forall t \in \mathcal{S}} \phi(t, r_i)$

- $U(\lambda) = \{u_1, u_2, \ldots, u_h\}$, *and* $u_i = \arg\max_{\forall t \in \mathcal{S}} \phi(t, r_i)$

where $\mathcal{S} = \tau(r_1) \cap \tau(r_2) \cap \cdots \cap \tau(r_h)$, $\phi(t, r_i)$ *is a function that returns the value of* t *in* r_i, *and* $i = 1 \ldots h$.

Intuitively, a rule formed by ranges with lower and upper s-boundaries is preferred to a min-$\sigma\gamma\delta$ rule whose ranges are defined on the same set of attributes and its cover is a superset of the cover of the rule formed by s-boundaries. That is because such a rule offers the same support and confidence as the min-$\sigma\gamma\delta$ rule does, but has higher density.

Given a rule $\lambda : r_1, r_2, \ldots, r_h \Rightarrow c$, s-boundaries suggest that the ranges in λ may be reduced without affecting its support and confidence. We can observe that tuples covered by these boundary values (i.e the values in $L(\lambda)$ and $U(\lambda)$), may not have c as a class value, this suggests that we can reduce the ranges in λ without affecting its support and confidence. Intuitively, it would be more meaningful that the rules start and end with a tuple whose class value is c, which is a special case of the proof in [35, 54] that optimal cut points for discretization must lie between tuples of different class labels. This effectively requires us to move the s-boundaries further inwards to the first tuple having a class c. These two new boundaries are called c-boundaries and they are represented by the two revised sets $L_c(\lambda)$ *and* $U_c(\lambda)$ in Figure 3.1. Formally,

Definition 3.3.3 (c-Boundaries) *Given a rule* $\lambda : r_1, r_2, \ldots, r_h \Rightarrow c$, *its lower and upper* c-boundaries, *denoted by* $L_c(\lambda)$ *and* $U_c(\lambda)$, *are:*

- $L_c(\lambda) = \{a_1, a_2, \ldots, a_h\}$, *and* $a_i = \arg\min_{\forall t \in \mathcal{S} \wedge C(t) = c} \phi(t, r_i)$
- $U_c(\lambda) = \{b_1, b_2, \ldots, b_h\}$, *and* $b_i = \arg\max_{\forall t \in \mathcal{S} \wedge C(t) = c} \phi(t, r_i)$

where $\mathcal{S} = \tau(r_1) \cap \tau(r_2) \cap \cdots \cap \tau(r_h)$, $C(t)$ *is a function that returns the class value of* t, $\phi(t, r_i)$ *is a function that returns the value of* t *in* r_i, *and* $i = 1 \ldots h$.

Note that a rule formed by ranges with c-boundaries is not necessarily "better than" those formed by ranges with s-boundaries, since in moving s-boundaries to c-boundaries, support is sacrificed for confidence. However, c-boundaries are intuitively preferred to supporting boundaries, as the amount of support that is lost in the process is associated with the tuples that do not support c, the class of the rule. Thus, in this work, the goal is to find a set of rules from a given table that are min-$\sigma\gamma\delta$, and are bounded by c-boundaries.

Definition 3.3.4 (Consequent bounded rules) *A rule* $\lambda : r_1, r_2, \ldots, r_h \Rightarrow c$

is a consequent bounded rule *(or c-bounded rule for short) if it is min-$\sigma\gamma\delta$, and for each of its ranges $r_i = [a_i, b_i], i = 1 \ldots h$, we have $a_i \in L_c(\lambda)$ and $b_i \in U_c(\lambda)$, where $L_c(\lambda)$ and $U_c(\lambda)$ are the lower and upper c-boundaries of λ.*

The following example illustrates all the concepts introduced in this section using the example data in Table 3.1.

Example 3.3.1 *Suppose that we have the following rule derived from the data in Table 3.1:*

$$\lambda : [4187.1, 56087.5]_{Check.Acc.} \wedge [445.2, 125000]_{Sav.Acc.} \Rightarrow N$$

The range $\lambda : [4187.1, 56087.5]_{Check.Acc.}$ is supported by $\{C_3, C_6, C_7, C_8, C_9, C_{10}\}$ whereas the range $[445.2, 125000]_{Sav.Acc.}$ is supported by $\{C_1, C_3, C_5, C_7, C_9, C_{10}\}$ and their conjunction is supported by $\{C_3, C_7, C_9, C_{10}\}$. Therefore the rule's lower and upper s-boundaries are $L(\lambda) = \{4850.36, 445.2\}$ and $U(\lambda) = \{56087.5, 125000\}$, respectively. This is more clear when using Table 3.2.

But for the clients with Check.Acc. value 4850.36 *and Sav.Acc. value* 445.2 *- note that even though in this case these belong to the same client, C_9 it does not have to be so - the class label for $LoanDef$ is Y, therefore $L(\lambda) \neq L_N(\lambda) = \{8220.4, 3250.12\}$. However since the clients with Check.Acc. value* 56087.5 *and Sav.Acc. value* 125000 *do have a $LoanDef$ value N, $U(\lambda) = U_N(\lambda)$. Therefore, according to the definition, and assuming the min-$\sigma\gamma\delta$ conditions are met the following is a consequent bounded rule:*

$$\lambda' : [8220.4, 56087.5]_{Check.Acc.} \wedge [3250.12, 125000]_{Sav.Acc.} \Rightarrow N$$

3.3.3 Finding Consequent Bounded Rules

In this section, the problem of mining consequent bounded rules from a given set of data is examined. An overview of the proposed method is given in Algorithm 3.3.2 while the details of our solution are discussed in the following sections.. For convenience, an association of i ranges will be referred to as an i-range in the following discussion.

Algorithm 3.3.2: *Finding consequent bounded rules*

Input: dataset $T(A_1, A_2, \ldots, A_m, C)$ and parameters σ_{min}, γ_{min} and δ_{min}

Output: a set of consequent bounded rules R

1 $R \leftarrow \emptyset$, $i \leftarrow 1$
2 **for** *each* c_k *in* C **do**
3 $\quad LR_1 \leftarrow$ generate-largest-1-ranges(T, σ_{min}, c_k)
4 $\quad$ **while** $LR_i \neq \emptyset$ **do**
5 $\quad\quad$ **for** *each* $\hat{r} \in LR_i$ **do**
6 $\quad\quad\quad R \leftarrow R \cup$ getRules$(\hat{r}, c_k, \sigma_{min}, \delta_{min}, \gamma_{min})$
7 $\quad\quad LR_{i+1} \leftarrow$ generate-largest-$(i+1)$-ranges$(LR_i, LR_1, \sigma_{min})$
8 $\quad\quad i \leftarrow i + 1$
9 **return** R

Algorithm 3.3.2 works as follows. Each distinct class value c_k is considered in turn (step 2). For each c_k, a set of *largest 1-ranges* is generated at first, LR_1, i.e. the largest range in each attribute from which c-bounded rules may be derived (step 3). As long as LR_1 (and LR_i in the subsequent iterations) is not empty,each i-range $\hat{r}$ in it is analyzed to see if c-bounded rules with i ranges in the antecedent may be derived from it and any resulting rules are added to the set of current results R (step 6). Then, $(i+1)$-ranges are generated from i-ranges iteratively (step 7), until no larger associated ranges can be produced (step 4). Finally, all the mined consequent bounded rules R will be returned (step 9).

3.3.4 Generating Largest 1-ranges

To explain how largest 1-ranges are generated from T (step 3 in Algorithm 3.3.2), a description is given of how T is represented in our method first. T is stored as a set of columns, and each column A_i is represented by a triple $\langle t_{id}, val, c\rangle_{A_i}$, where *val* records the values of A_i sorted in an ascending order, t_{id} records their corresponding tuple identifiers, and c records their class values. The original tuple identifier t_{id} is stored because the sorting process changes the relevant position of each value and the new tuple order differs for each A_i depending on the actual values. For example after sorting the columns of Table 3.1 in ascending order (the result can be seen in Table 3.2) there is no connection between individual values and the corresponding client id (tuple in the original Table). t_{id} serves as that reference, that is values with the same t_{id} were originally in the same tuple and

refer to the same client id.

In the case of Table 3.1 the resulting structure after sorting the values per attribute is shown in Table 3.2. Note that even though the largest 1-range differs for each distinct class label c_k, the sorted order of the values remains the same therefore the sorting process is only performed once.

Checking Account		
t_{id}	val	c
t_5	-345.2	N
t_2	0.0	Y
t_4	127.3	Y
t_1	2003.15	Y
t_8	4187.1	Y
t_9	4850.36	Y
t_{10}	8220.4	N
t_6	11023.04	N
t_7	19873.6	N
t_3	56087.5	N

Savings Account		
t_{id}	val	c
t_6	0.0	N
t_8	0.0	Y
t_4	0.45	Y
t_2	100.3	Y
t_9	445.2	Y
t_1	2000	Y
t_{10}	3250.12	N
t_5	5250.5	N
t_7	22467.4	N
t_3	125000	N

Outstanding Loans		
t_{id}	val	c
t_3	0.0	N
t_2	75.8	Y
t_8	575.4	Y
t_7	2420.25	N
t_1	2800	Y
t_4	3250.6	Y
t_9	7230.2	Y
t_5	8725.5	N
t_6	8725.5	N
t_{10}	25225	N

Table 3.2: Table 3.1 after sorting values per attribute.

Sorting the values allows to examine each individual value as a possible starting/ending point for the largest 1-range of an attribute for a given class label, starting from the smallest and largest values and continuously reducing the range. Algorithm 3.3.3 describes this process in more detail.

Algorithm 3.3.3: *generate-largest-1-ranges*

Input: $T(A_1, A_2, \ldots, A_m, C)$, σ_{min} and c_k

Output: LR_1

```
1 LR_1 ← ∅
2 for each A_i in T do
3     ⟨tid, val, c⟩_{A_i} ← sort(A_i, c_k)
4     ⟨tid, val, c⟩_{A_i} ← revise-range(⟨tid, val, c⟩_{A_i})
5     if |⟨tid, val, c⟩_{A_i}| ≥ σ_min then
6         LR_1 ← LR_1 ∪ {⟨tid, val, c⟩_{A_i}}
7 return LR_1
```

Algorithm 3.3.3 works as follows. First, each attribute A_i is converted (sorted) into its triple structure $\langle tid, val, c\rangle_{A_i}$ (steps 2 and 3). After sorting, the *revise-range* function is used to remove the tuples whose class values are not c_k at the two ends (step 4). Once this is done and if it has enough support ($|\cdot|$ denotes cardinality), the range is added to LR_1 (steps 5 and 6). Note that strictly speaking, $\sigma_{min} \times |T|$ should be used, that is the number of supporting tuples, instead of σ_{min} which represents the support threshold itself, in step 5. Finally, LR_1 is returned as the result (step 7). The following example demonstrates this process for the data in Table 3.2.

Example 3.3.2 *Given the sorted values in Table 3.2 and for class Y, the* generate-largest-1-ranges *functions returns the following three triples as LR_1 for the corresponding attributes:*

Checking Account w.r.t. Y

t_{id}	val	c
t_2	0.0	1
t_4	127.3	1
t_1	2003.15	1
t_8	4187.1	1
t_9	4850.36	1

Savings Account w.r.t. Y

t_{id}	val	c
t_8	0.0	1
t_4	0.45	1
t_2	100.3	1
t_9	445.2	1
t_1	2000	1

Outstanding Loans w.r.t. Y

t_{id}	val	c
t_2	75.8	1
t_8	575.4	1
t_7	2420.25	0
t_1	2800	1
t_4	3250.6	1
t_9	7230.2	1

Table 3.3: LR_1 ranges for the attributes of Table 3.1

Observe that these are the largest possible ranges from which Y-bounded rules may be derived for each attribute.

Note that LR_1 is generated with regards to a specific target class label therefore the values of c change to a single bit which is 1 when the class label is the target class label, the tuple containing this attribute value supports the target class, and 0 when it is any of the other class labels. The reasons for this representation of class values will become more clear later.

3.3.5 Generating Largest $(i+1)$-ranges

This section describes how largest-$(i+1)$-ranges may be generated from i-ranges by attempting to merge i-ranges with each largest 1-range (step 7 in Algorithm 3.3.2).

Algorithm 3.3.4: *generate-largest-*$(i+1)$*-ranges*

Input: dataset LR_i, LR_1 and σ_{min}

Output: LR_{i+1}

1 $LR_{i+1} \leftarrow \emptyset$
2 **for** *each* l *in* LR_i **do**
3 **for** *each* k *in* LR_1 **do**
4 $L_{cand} \leftarrow$ revise-range$(l \wedge k)$
5 **if** $\sigma(L_{cand}) \geq \sigma_{min}$ **then**
6 $LR_{i+1} \leftarrow LR_{i+1} \cup L_{cand}$
7 **return** LR_{i+1}

This step is straightforward: for each largest i-range each largest 1-range is examined. The implementation of our algorithm uses a data structure that allows us to avoid redundant pairings between an LR_1 and an LR_i that refer to the same attribute. Section 3.3.6 offers a detailed description of this data structure. If the *candidate* largest range, formed by the conjunction of the two ranges in step 4, meets the given support threshold in step 5 then it is added as a largest i-range in step 6. Finally, the new LR_{i+1} ranges are returned in step 7.A graphical representation of the process of generating larger 3-ranges is given in Figure 3.2.

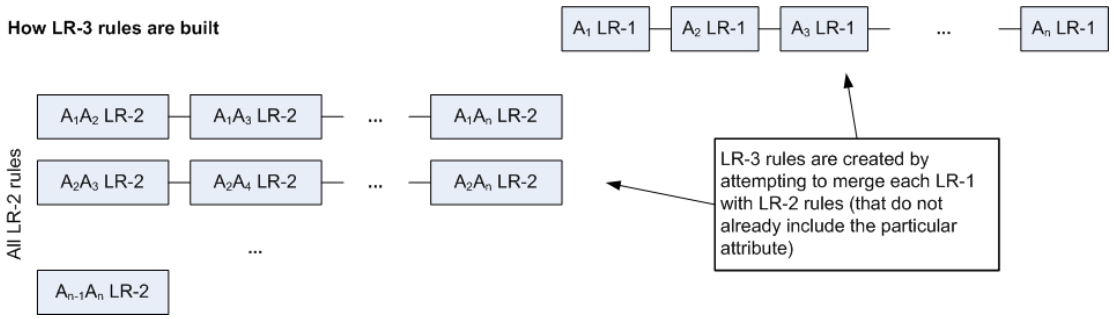


Figure 3.2: LR structure for storing candidate rules.

3.3.6 LR Structure

Algorithm 3.3.4 employs a data structure specifically designed to store the largest i-ranges. This section describes the aforementioned data structure in depth.

As demonstrated in Figure 3.3 the structure resembles a *hash table* where the role of a *key* is played by an attribute index n and the *hash function* maps all the associated ranges that include a range in A_n and attributes with index $> n$. Any attributes with index $< n$ are mapped to the attribute with the smallest index.

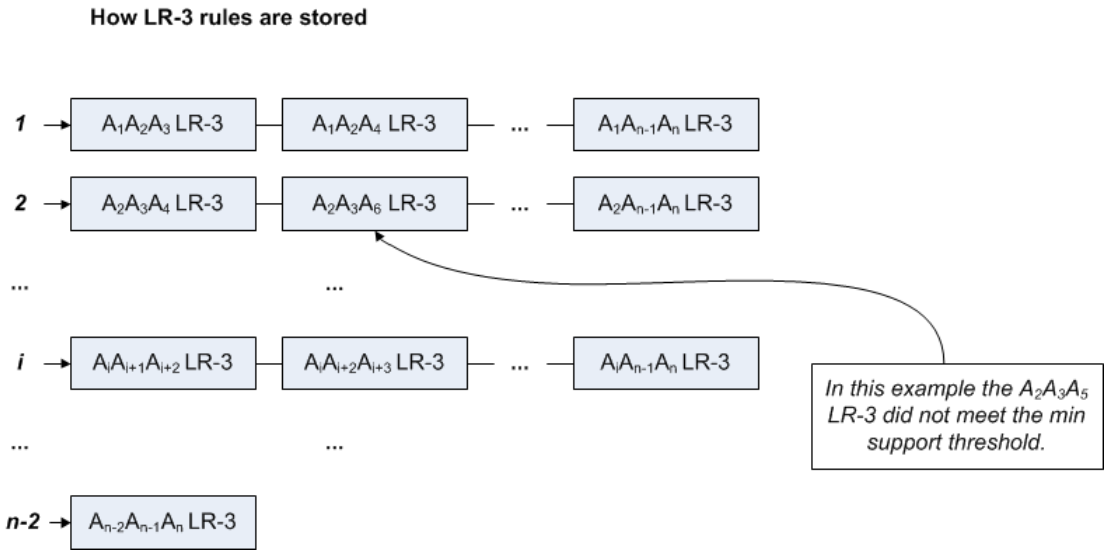


Figure 3.3: LR structure for storing candidate rules.

When generating largest $i+1$-ranges by merging 1-ranges with the largest i-ranges in the existing structure, a largest 1-range for attribute A_k can only be merged with ranges that have an index key $> k$ so as to avoid duplicate checks while merging.

The described structure is employed for the generation of each $LR_i, 1 \leq i \leq n$ for each class label c_k in the given data. Based on the description of Algorithm 3.3.4, however, each LR_i structure is only needed for the generation of LR_{i+1} and the corresponding resources may be freed afterwards.

3.3.7 Splitting Largest Ranges Into Consequent Bounded Rules

This section explains how consequent bounded rules are generated in LR_i (step 6 in Algorithm 3.3.2). Unfortunately, finding all valid consequent bounded rules according to Definition 3.3.4 is still too expensive. This is because consequent bounded rules may overlap. For example,

$$[0.12, 1.25]_{A_1} \wedge [1.22, 3.55]_{A_2} \Rightarrow c_1$$

and

$$[0.94, 1.55]_{A_1} \wedge [2.12, 4.05]_{A_2} \Rightarrow c_1$$

can both be valid rules, i.e. they both satisfy minimum support, confidence and density requirements, and yet one does not subsume the other. This significantly increases the search space for finding the rules.

To reduce complexity and allow for more realistic performance, we consider, as a heuristic, the splitting of largest ranges into "non-overlapping" consequent bounded rules. However, outside the context of splitting a specific largest range, the designed solution still mines overlapping rules due to the overlap between largest-ranges.

Definition 3.3.5 (Overlapping rules) *Let λ and λ' be two structurally identical c-bounded rules (i.e. involving the same set of attributes and the same class value):*

$$\lambda : [a_1, b_1]_{A_1} \wedge [a_2, b_2]_{A_2} \wedge \cdots \wedge [a_h, b_h]_{A_h} \Rightarrow c$$

$$\lambda' : [a'_1, b'_1]_{A_1} \wedge [a'_2, b'_2]_{A_2} \wedge \cdots \wedge [a'_h, b'_h]_{A_h} \Rightarrow c$$

λ and λ' are overlapping *if there exists at least one pair of ranges $[a_j, b_j]_{A_j}$ and $[a'_j, b'_j]_{A_j}$ such that they overlap.*

Thus, to find non-overlapping rules is equivalent to saying that once a c-bounded rule is found, the associated ranges in this rule will not be allowed to be extended or reduced to form another rule. This turns the initial search problem into a partition problem, i.e. if a large range does not satisfy minimum confidence and density requirements, it may be split into smaller ones to check if they do. This is more efficient. Chapter 4 will introduce several partitioning heuristics that use different criteria to achieve the best possible split.

The methods described in Sections 4.1.1, 4.1.2, 4.1.3 and 4.1.4 describe heuristics that the presented solution employs in step 6 of Algorithm 3.3.2 for the generation of consequent bounded rules from the generated LR_i.

As explained in [48, 49] due to the monotonic nature of support while splitting the largest ranges the resulting rules' support may only decrease. This, however, is not necessarily true for the other interestingness measures used in this approach. The following definitions describe the relationship between the new ranges that result from splitting and the range-based association rules that consist of these ranges.

Definition 3.3.6 (Subrange) *Let $r_1 = [a_1, b_1]_A$ and $r_2 = [a_2, b_2]_A$ be two ranges on an attribute A. r_1 is a* subrange *of r_2, denoted by $r_1 \sqsubseteq r_2$, if $a_1 \geq a_2$ and $b_1 \leq b_2$. If either $a_1 > a_2$ or $b_1 < b_2$, we denote the subrange as $r_1 \sqsubset r_2$.*

Definition 3.3.7 (Subrule) *Let $ar_1 : r_1, r_2, \ldots r_h \Rightarrow c$ and $ar_2 : r'_1, r'_2, \ldots r'_k \Rightarrow c$ be two range-based association rules derived from table T. We say that ar_1 is a* subrule *of ar_2, $ar_1 \prec ar_2$ if for each r_i in the antecedent of ar_1 there exists an r'_j in the antecedent of ar_2 such that $r_i \sqsubseteq r'_j$.*

The concept behind splitting is a sacrifice of support in order to increase confidence and/or gain. There are, however, no guarantees that after splitting a range the resulting rules will have increased confidence or gain in which case the resulting rules do not provide any improved knowledge on the dataset. Definition 3.3.8 formally describes this case in detail.

Definition 3.3.8 (Rule redundancy) *Consider two range-based association rules r and r' with the same consequent c. If $r \prec r'$ with $\gamma(r) \leq \gamma(r')$ and $\delta(r) \leq \delta(r')$ then we call r* redundant.

3.4 Summary

This chapter introduced our novel framework for generating range-based classification rules. The key terms and concepts used in this work have been defined, including the definition of a new measure covering a data property specific only to rules on continuous data ranges.

A general to specific solution has been defined along with the challenges of implementing such a method. Furthermore, the concept of consequent bounded rules

has been defined as a solution to reducing the complexity of the original problem. The aforementioned rules are mined from largest-ranges that have also been defined in this chapter along with an algorithmic solution describing the incremental construction of these ranges.

Finally the criteria used for splitting largest-ranges into rules that meet given criteria in the form of threshold values have been defined. The following chapter demonstrates different heuristics designed to address the splitting of largest ranges into range-based classification rules.

Chapter 4

CARM Algorithm

A largest-range constitutes the search space where consequent bounded rules may be mined from and may even itself constitute a consequent bounded rule since, by definition, it meets the given criteria of $\sigma_{min}, \gamma_{min}, \delta_{min}$.

This chapter presents the *Characterization Associative Rule Mining* (CARM) algorithm for mining range-based rules. Section 4.1 presents methods developed for the purpose of splitting a largest range in consequent bounded rules.

4.1 Heuristics

Each heuristic is driven by a specific goal. Modifying the criterion for splitting, results in a separate heuristic based method for mining consequent bounded rules. The splitting process as referred in Section 3.3.7 is achieved by removing a specific range from a given largest range as shown in Figure 4.1.

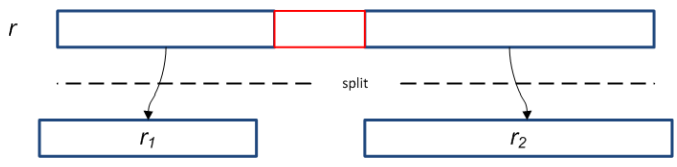


Figure 4.1: Graphic representation of a range split.

Note that the range selected for removal is not considered as one of the resulting ranges. Furthermore, it is possible that one of the resulting ranges r_1, r_2 will be

empty and the actual split will generate a single range. The methods described in this section utilise a different heuristic each to mine range-based rules.

4.1.1 Maximum Support Heuristic

Support is a measure of strength for a rule. The larger the support of a rule, potentially the more significant the rule is. Thus, it makes sense to consider a class of rules whose support is somewhat "maximized". In this section, a method for finding maximally supported rules is described.

Definition 4.1.1 (Max-σ rule) *A range-based rule $\lambda : r_1, r_2, \ldots, r_d \Rightarrow c$ is said to be a* max-σ rule *if it satisfies the following properties:*

1. *λ is a min-$\sigma\gamma\delta$ rule,*
2. *For each range r_i, $i = 1 \ldots d$ in λ, $L(r_i)$ and $U(r_i)$ each contain at least one tuple whose class is c,*
3. *There does not exist another rule $\lambda' : r'_1, r'_2, \ldots, r'_i, \ldots, r'_d \Rightarrow c$ such that*
   (a) *$\sigma(\lambda') > \sigma(\lambda)$ and $r_j \sqsubseteq r'_j$, $j = 1 \ldots d$, or*
   (b) *$\sigma(\lambda') = \sigma(\lambda)$ and there exists at least one r'_j such that $r'_j \sqsubset r_j$.*

Definition 4.1.1 requires some explanation. Conditions 1, 2 are included so that a max-σ rule will still have some minimum credibility. Condition 3 is to ensure its "uniqueness". To illustrate this, consider the following example:

Example 4.1.1 *Suppose that we have the following three rules and they all have the required minimum support, confidence and density in Table 3.1:*

$$\lambda_1 : [2003.15, 4850.36]_{Check.Acc.} \wedge [75.8, 7230.2]_{Loans\_Out.} \Rightarrow Y$$
$$\lambda_2 : [0, 4850.36]_{Check.Acc.} \wedge [75.8, 7230.2]_{Loans\_Out.} \Rightarrow Y$$
$$\lambda_3 : [0, 4850.36]_{Check.Acc.} \wedge [0, 8725.5]_{Loans\_Out.} \Rightarrow Y$$

λ_1 is not a max-σ rule since its antecedent includes the associated range $[2003.15, 4850.36]_{Check.Acc.} \sqsubset [0, 4850.36]_{Check.Acc.}$ of λ_2's antecedent and $(\sigma(\lambda_2) = 0.5) > (\sigma(\lambda_1) = 0.3)$, violating Condition 3(a) in Definition 4.1.1. λ_2 is not a max-σ rule either since $[75.8, 7230.2]_{Loans\_Out.} \sqsubset [0, 8725.5]_{Loans\_Out.}$ and $\sigma(\lambda_2) = 0.5 = \sigma(\lambda_3)$, violating Condition 3(b) in Definition 4.1.1.

In order to generate rules that meet the criteria described in Definition 4.1.1 a solution was developed and implemented employing the heuristic described in the

following section.

Maximum Support Split

In order to maximize the support of the rules resulting after the split this heuristic looks for the smallest range (i.e the range of minimum support) with a class label different than the one in the rule's consequent, in a given largest i-range. The process is described in detail in Algorithm 4.1.1.

Algorithm 4.1.1: *Maximum support rule generation*

Input: A largest i-range l, parameters σ_{min}, γ_{min} and δ_{min} and the target class label c_t

Output: A ruleset R

1 $R \leftarrow \emptyset$
2 $Q \leftarrow \varnothing$
3 $Q \leftarrow Q.enqueue(l)$
4 **while** $Q \neq varnothing$ **do**
5 $q \leftarrow Q.dequeue()$
6 **if** $\sigma(q) \geq \sigma_{min}$ **then**
7 **if** $\gamma(q) \geq \gamma_{min} \wedge \delta(q) \geq \delta_{min}$ **then**
8 $R \leftarrow R \cup q$
9 **else**
10 $< r_1, r_2 > \leftarrow$ maxSupportSplit(q)
11 $Q \leftarrow Q.enqueue(r_1)$
12 $Q \leftarrow Q.enqueue(r_2)$
13 **return** R

The algorithm describes how each largest range is checked as a possible result, notice how in step 3 the given largest range will be the first to be examined, if it meets the given thresholds and if it does will be added to the resultset R in step 8. If the candidate rule does not meet the confidence or density requirement but does meet the minimum support requirement it is split using the method described in Algorithm 4.1.2 in two new rules in step 10 that are then added to the existing queue in steps 11, 12. Therefore, the presented method is effectively performing a *breadth-first search* in the tree resulting from a recursive split of the given largest i-range using the presented heuristic. Any node of the tree that does

meet all requirements is added as a result since further splitting can only decrease its support. For the same reason any candidate that does not meet the minimum support threshold is omitted (a consequence of step 5 in the algorithm).

Algorithm 4.1.2: *maxSupportSplit*

Input: A rule $r : g_1 \wedge g_2 \wedge \cdots \wedge g_i \Rightarrow c_t$

Output: Two rules r_1 and r_2

1 $g_{min} \leftarrow$ findRangeWithMinNegSeq(r, c_t)

2 $< g'_{min}, g''_{min} > \leftarrow$ removeMinNegSeq(g_{min})

3 $r_1 \leftarrow g_1 \wedge g_2 \wedge g'_{min} \wedge \cdots \wedge g_i$

4 $r_2 \leftarrow g_1 \wedge g_2 \wedge g''_{min} \wedge \cdots \wedge g_i$

5 **return** $< r_1, r_2 >$

Algorithm 4.1.2 is rather straightforward. The range with the lowest support and a class label different than the target class is detected in step 1. By removing the aforementioned range the original range g_{min} is split in two new ranges g'_{min} and g''_{min} in step 2. These ranges are then used in conjunction with the other unmodified ranges and form the two resulting rules in steps 3 and 4. There are specific (but infrequent) cases when one of the two ranges may be empty.

The process of splitting a rule $r : [v_6, v_9]_{A_1} \wedge [u_4, u_7]_{A_2} \Rightarrow c_k$ into two new rules by removing the smallest range in length where the class value is not c_k is shown in Table 4.1 and Table 4.2. In the provided example v_i represents the sorted values of attribute A_1 for tuple i in the original table T whereas u_i represents the sorted values of attribute A_2 for the same tuple.

Table 4.2 is generated by removing a single value in this case. Note that neither of the resulting rules has support $\geq \sigma(r)$ which is why this method stops splitting as soon as a rule that meets the given thresholds is generated. In this case, however, it is presumed that r did not meet one of the $\gamma_{min}, \delta_{min}$ thresholds and it was split so that r'_1 could be generated with $\gamma(r'_1) > \gamma(r)$.

A_1	
val	c_k
v_6	1
v_{10}	1
v_1	**0**
v_{14}	1
v_7	1
v_{11}	1
v_2	1
v_{12}	1
v_5	0
v_{13}	0
v_3	0
v_4	1
v_{16}	1
v_{15}	0
v_8	0
v_9	1

A_2	
val	c_k
u_4	1
u_2	1
u_{10}	1
u_{16}	1
u_3	0
u_5	0
u_8	0
u_{15}	0
u_{11}	1
u_7	1

Table 4.1: The original rule r.

A_1	
val	c_k
v_6	1
v_{10}	1

A_2	
val	c_k
u_4	1
u_2	1
u_{10}	1
u_{16}	1
u_3	0
u_5	0
u_8	0
u_{15}	0
u_{11}	1
u_7	1

A_1	
val	c_k
v_{14}	1
v_7	1
v_{11}	1
v_2	1
v_{12}	1
v_5	0
v_{13}	0
v_3	0
v_4	1
v_{16}	1
v_{15}	0
v_8	0
v_9	1

A_2	
val	c_k
u_4	1
u_2	1
u_{10}	1
u_{16}	1
u_3	0
u_5	0
u_8	0
u_{15}	0
u_{11}	1
u_7	1

Table 4.2: The resulting rules $r'_1 : [v_6, v_{10}]_{A_1} \wedge [u_4, u_7]_{A_2} \Rightarrow c_k$ and $r'_2 : [v_{14}, v_9]_{A_1} \wedge [u_4, u_7]_{A_2} \Rightarrow c_k$.

A graphic representation of the complete method is given in Figure 4.2.

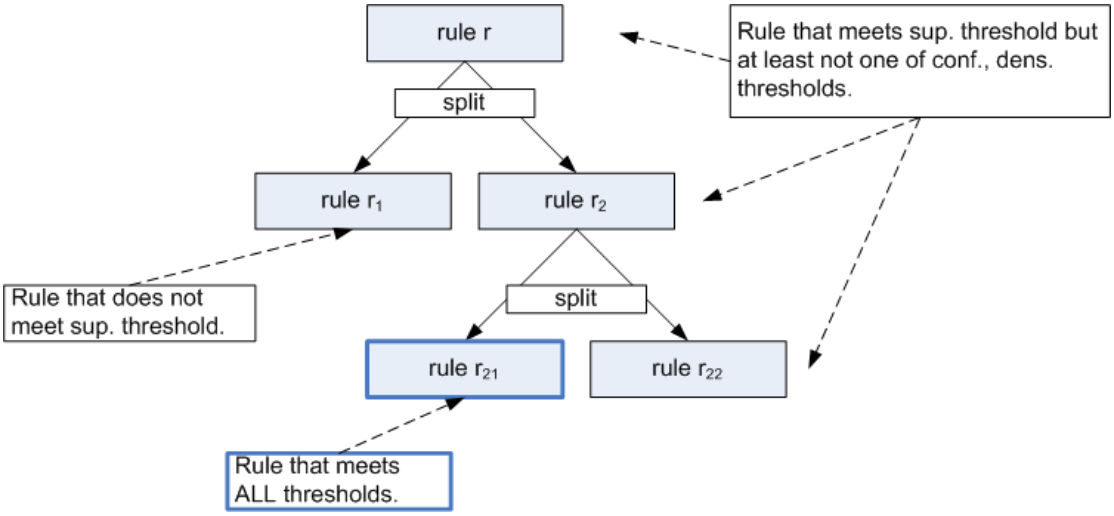


Figure 4.2: Tree generated by maximum support splits.

4.1.2 Maximum Confidence Heuristic

Confidence is a measure of validity for a rule. A highly confident rule is considered a valid knowledge on the dataset. Thus, it makes sense to consider a class of rules whose confidence is somewhat "maximized" based on a different splitting criterion than in Section 4.1.1. This section describes how maximally confident rules may be mined.

Definition 4.1.2 (Max-γ rule) *A range-based rule* $\lambda : r_1, r_2, \ldots, r_d \Rightarrow c$ *is said to be a* max-γ rule *if it satisfies the following properties:*

1. λ *is a min-*$\sigma\delta\gamma$ *rule,*
2. *For each range* r_i, $i = 1 \ldots d$ *in* λ, $L(r_i)$ *and* $U(r_i)$ *each contain at least one tuple whose class is* c,
3. *There does not exist another rule* $\lambda' : r'_1, r'_2, \ldots, r'_i, \ldots, r'_d \Rightarrow c$ *such that*
   (a) $\gamma(\lambda') \geq \gamma(\lambda)$ *and* $r_j \sqsubseteq r'_j$, $j = 1 \ldots d$, *or*
   (b) $\gamma(\lambda') = \gamma(\lambda)$ *and there exists at least one* r'_j *such that* $r'_j \sqsubset r_j$.

Definition 4.1.2 is similar to Definition 4.1.1. Conditions 1 and 2 are included so that a max-γ rule will still have some minimum credibility whereas Condition 3 ensures its "uniqueness". Example 4.1.2 demonstrates this:

Example 4.1.2 *Suppose that we have the following three rules and they all have the required minimum support, confidence and density in Table 3.1:*

$$\lambda_1 : [0, 4850.36]_{Check.Acc.} \wedge [0, 100.3]_{Sav.Acc.} \Rightarrow Y$$
$$\lambda_2 : [0, 4850.36]_{Check.Acc.} \wedge [0, 2000]_{Sav.Acc.} \Rightarrow Y$$
$$\lambda_3 : [0, 4850.36]_{Check.Acc.} \wedge [0, 5250.5]_{Sav.Acc.} \Rightarrow Y$$

λ_1 is not a max-γ rule because its antecedent includes the associated range $[0, 100.3]_{Sav.Acc.} \sqsubset [0, 2000]_{Sav.Acc.}$ of λ_2's antecedent and $\gamma(\lambda_2) = 1 = \gamma(\lambda_1)$, violating Condition 3(a) in Definition 4.1.2. λ_3 is also not a max-γ rule since $[0, 2000]_{Sav.Acc.} \sqsubset [0, 5250.5]_{Sav.Acc.}$ and $(\gamma(\lambda_2) = 1) > (\gamma(\lambda_3) = \frac{5}{7})$, violating Condition 3(b) in Definition 4.1.2.

Section 4.1.2 describes a method that uses a maximum confidence heuristic in order to get rules that meet Definition 4.1.2 by splitting the given largest range.

Maximum Confidence Split

Unlike the method described in 4.1.1 this approach aims to maximize confidence, therefore the split method in this case splits the rule after removing the largest range with a class label different than the rule's consequent. The motivation is simple, whereas in 4.1.4 the removal of the minimum range with a class label different than the rule's consequent aims to increase confidence and density, in order to meet the given thresholds by removing the smallest number of tuples possible, so as to keep support as high as possible, this approach prioritizes confidence maximization which is achieved by removing negative instances from a rule's supporting tuples. Algorithm 4.1.3 describes this approach.

As with Algorithm 4.1.2 a breadth-first search method is employed and each largest range, starting with l in step 3 is being split. The criteria for stopping the expansion of the search tree is again the minimum support threshold in step 6 because support is the only monotonic interest measure used. The confidence and density of the new rules resulting from splitting a range may increase, actually the very purpose of this heuristic is to increase confidence but when a split occurs the resulting rules' support can only decrease. In step 8 the rule found with maximum confidence is recorded while every new rule that results from splitting in step 9, is added to the queue in steps 11 and 12. Note that unlike Algorithm 4.1.1 the rule that is currently the one with the maximum confidence is itself added to the queue for further splitting based on the aforementioned principal of increasing confidence with every split. Algorithm 4.1.3, as explained, uses a different method to split the rules which is described in detail in Algorithm 4.1.4.

Algorithm 4.1.3: *Maximum confidence rule generation*

Input: A largest i-range l, parameters σ_{min}, γ_{min} and δ_{min} and the target class label c_t

Output: The rule with the maximum confidence r_{max}

1 $r_{max} \leftarrow \varnothing$
2 $Q \leftarrow \varnothing$
3 $Q \leftarrow Q.enqueue(l)$
4 **while** $Q \neq \varnothing$ **do**
5 $q \leftarrow Q.dequeue()$
6 **if** $\sigma(q) \geq \sigma_{min}$ **then**
7 **if** $\gamma(q) \geq \gamma_{min} \wedge \delta(q) \geq \delta_{min} \wedge \gamma(q) \geq \gamma(r_{max})$ **then**
8 $r_{max} \leftarrow q$
9 $< r_1, r_2 > \leftarrow$ maxConfidenceSplit(q)
10 $Q \leftarrow Q.enqueue(r_1)$
11 $Q \leftarrow Q.enqueue(r_2)$
12 **return** r_{max}

Algorithm 4.1.4: *maxConfidenceSplit*

Input: A rule $r : g_1 \wedge g_2 \wedge \cdots \wedge g_i \Rightarrow c_t$

Output: Two rules r_1 and r_2

1 $g_{max} \leftarrow$ findRangeWithMaxNegSeq(r, c_t)
2 $< g'_{max}, g''_{max} > \leftarrow$ removeMaxNegSeq(g_{max})
3 $r_1 \leftarrow g_1 \wedge g_2 \wedge g'_{max} \wedge \cdots \wedge g_i$
4 $r_2 \leftarrow g_1 \wedge g_2 \wedge g''_{max} \wedge \cdots \wedge g_i$
5 **return** $< r_1, r_2 >$

The process is very similar to that in Algorithm 4.1.2. The range with the highest support and a class label different than the target class is detected in step 1. By removing the aforementioned range the original range g_{max} is split in two new ranges g'_{max} and g''_{max} in step 2. These ranges are then used in conjunction with the other unmodified ranges and form the two resulting rules in steps 3 and 4.

The example provided in Table 4.3 and Table 4.4 uses the same rule r but splits it using the method described in this section.

A_1	
val	c_k
v_6	1
v_{10}	1
v_1	0
v_{14}	1
v_7	1
v_{11}	1
v_2	1
v_{12}	1
v_5	0
v_{13}	0
v_3	0
v_4	1
v_{16}	1
v_{15}	0
v_8	0
v_9	1

A_2	
val	c_k
u_4	1
u_2	1
u_{10}	1
u_{16}	1
u_3	**0**
u_5	**0**
u_8	**0**
u_{15}	**0**
u_{11}	1
u_7	1

Table 4.3: The original rule r.

A_1	
val	c_k
v_6	1
v_{10}	1
v_1	0
v_{14}	1
v_7	1
v_{11}	1
v_2	1
v_{12}	1
v_5	0
v_{13}	0
v_3	0
v_4	1
v_{16}	1
v_{15}	0
v_8	0
v_9	1

A_2	
val	c_k
u_4	1
u_2	1
u_{10}	1
u_{16}	1

A_1	
val	c_k
v_6	1
v_{10}	1
v_1	0
v_{14}	1
v_7	1
v_{11}	1
v_2	1
v_{12}	1
v_5	0
v_{13}	0
v_3	0
v_4	1
v_{16}	1
v_{15}	0
v_8	0
v_9	1

A_2	
val	c_k
u_{11}	1
u_7	1

Table 4.4: The resulting rules $r'_1 : [v_6, v_9]_{A_1} \wedge [u_4, u_{16}]_{A_2} \Rightarrow c_k$ and $r'_2 : [v_6, v_9]_{A_1} \wedge [u_{11}, u_7]_{A_2} \Rightarrow c_k$.

Figure 4.3 is a graphic representation of the search tree generated from splitting the largest range using the described heuristic. It should be noted that unlike Section 4.1.1 the returned rule in this case is not necessarily a leaf in the search tree.

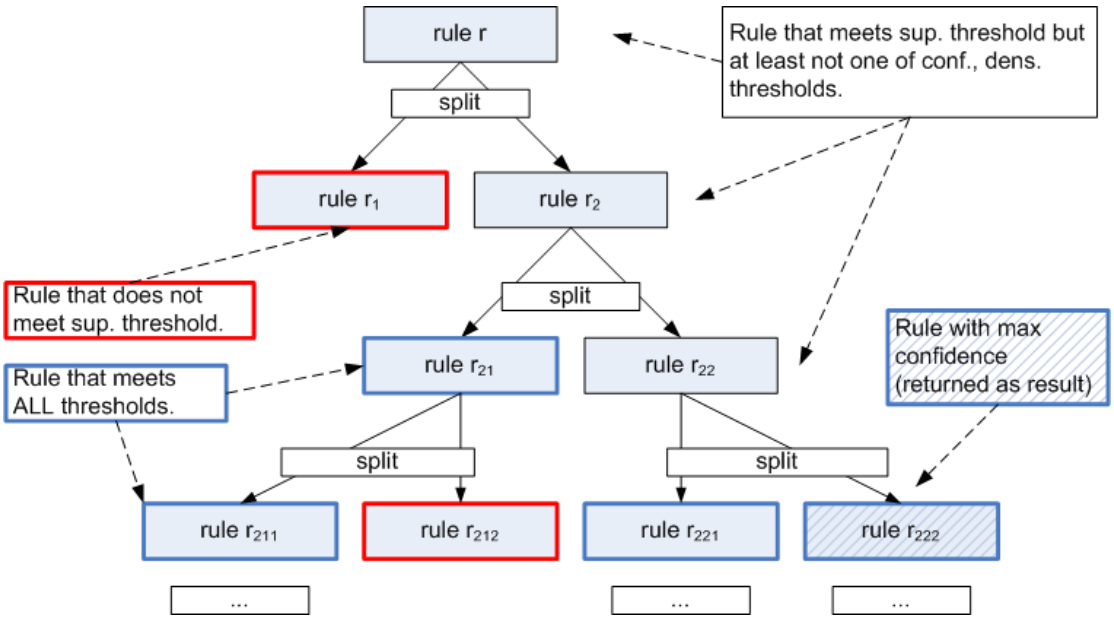


Figure 4.3: Tree generated by maximum confidence splits.

4.1.3 Maximum Gain Heuristic

It is important to mine association rules whose confidence is high, and whose support is sufficiently large. This problem is more complicated than just mining the rules with maximum confidence given a support minimum threshold or mining the rules with maximum support given a minimum threshold for confidence as in both cases the results often maximize the target measure at the expense of the second one which, in most results, is just above the minimum threshold. Intuitively the best results have a high confidence without sacrificing too much support.

One way of achieving this is by using the method described in 4.1.2 with a very high support threshold but this would be problematic for several reasons. First, since the user does not have any data knowledge setting an appropriately high σ_{min} will require experimentation and even then it is possible that a standard σ_{min} creates problems since for different class labels a very high value may reduce the number of results significantly and potentially remove useful results and therefore knowledge from the final resultset. Because the goal is to only increase an interestingness

measure if the increase in that measure justifies the potential cost in the others, a different interestingness measure called *gain*, that has already been mentioned in Section 2.5, and was originally used in [42] is employed. Below is a definition of *gain* as it is used in this work.

Definition 4.1.3 (Gain) *Let T be a table and $\lambda : r_1, r_2, \ldots, r_h \Rightarrow c$ be a range-based rule derived from T. The* gain *for λ in T is*

$$gain(\lambda) = |\tau(r_1) \cap \tau(r_2) \cap \cdots \cap \tau(r_h) \cap \tau(c)| - \gamma_{min} \times |\tau(r_1) \cap \tau(r_2) \cap \cdots \cap \tau(r_h)|$$

where $\tau(c)$ denotes the set of tuples that have class value c in T.

The following example describes the evaluation of gain for a rule.

Example 4.1.3 *Suppose we have the data in Table 3.1 and the rule $\lambda : [0, 4850.36]_{Check.Acc.} \wedge [0, 2000]_{Sav.Acc.} \wedge [75.8, 3250.6]_{Loans\_Out.} \Rightarrow Y$, and a given threshold of $\gamma_{min} = 0.5$ then we have*

$$\begin{aligned}
\gamma(\lambda) &= |\tau([0, 4850.36]_{Check.Acc.}) \cap \tau([0, 2000]_{Sav.Acc.}) \cap \tau([75.8, 3250.6]_{Loans_Out.}) \cap \tau(Y)| \\
&\quad - \gamma_{min} \times |\tau([0, 4850.36]_{Check.Acc.}) \cap \tau([0, 2000]_{Sav.Acc.}) \\
&\quad \cap \tau([75.8, 3250.6]_{Loans_Out.})| \\
&= |\{t_1, t_2, t_4, t_8, t_9\} \cap \{t_1, t_2, t_4, t_8, t_9\} \cap \{t_1, t_2, t_4, t_7, t_8\} \cap \{t_1, t_2, t_4, t_8, t_9\}| \\
&\quad - \gamma_{min} \times |\{t_1, t_2, t_4, t_8, t_9\} \cap \{t_1, t_2, t_4, t_8, t_9\} \cap \{t_1, t_2, t_4, t_7, t_8\}| \\
&= 4 - 0.5 \times 4 = 2
\end{aligned}$$

Since gain is used to evaluate the trade-off between support and confidence, and because it is not a commonly used/known measure, there is no corresponding threshold of $gain_{min}$. Definition 4.1.4 describes a *max-gain rule* in accordance to Definitions 4.1.1, 4.1.2.

Definition 4.1.4 (Max-gain rule) *A range-based rule $\lambda : r_1, r_2, \ldots, r_d \Rightarrow c$ is said to be a* max-gain rule *if it satisfies the following properties:*

1. *λ is a min-$\sigma\gamma\delta$ rule,*
2. *For each range r_i, $i = 1 \ldots d$ in λ, $L(r_i)$ and $U(r_i)$ each contain at least one tuple whose class is c,*
3. *There does not exist another rule $\lambda' : r'_1, r'_2, \ldots, r'_i, \ldots, r'_d \Rightarrow c$ such that*

(a) $gain(\lambda') \geq gain(\lambda)$ *and* $r_j \sqsubseteq r'_j$, $j = 1 \ldots d$, *or*

(b) $gain(\lambda') = gain(\lambda)$ *and there exists at least one* r'_j *such that* $r'_j \sqsubset r_j$.

Algorithm 4.1.5 describes a method that mines range-based association rules by mining the rules of maximum gain from the largest ranges by following the described process of splitting.

Algorithm 4.1.5: *Maximum gain rule generation*

Input: A largest i-range l, parameters σ_{min}, γ_{min} and δ_{min} and the target class label c_t

Output: The rule with the maximum confidence r_{max}

```
1  r_max ← ∅
2  Q ← ∅
3  Q ← Q.enqueue(l)
4  while Q ≠ ∅ do
5      q ← Q.dequeue()
6      if σ(q) ≥ σ_min then
7          if γ(q) ≥ γ_min ∧ δ(q) ≥ δ_min ∧ gain(q) ≥ gain(r_max) then
8              r_max ← q
9          < r_1, r_2 >← maxConfidenceSplit(q)
10         Q ← Q.enqueue(r_1)
11         Q ← Q.enqueue(r_2)
12 return r_max
```

Same as before a breadth-first search method is employed and each largest range, starting with l in step 3 is being split. Gain, like confidence and density is not monotonic and therefore the expansion of the search tree only stops if the minimum support threshold in step 6, is not met. In step 8 the rule found to have maximum gain is recorded while every new rule that results from splitting in step 9, is added to the queue in steps 11, 12. As in Algorithm 4.1.3 the split method employed is that of Algorithm 4.1.4 which splits a range by removing the smallest range of tuples that do not have the target class label. The reason why a new heuristic is not developed for this case is because gain is used as a comparative measure as to which rules in the search tree is the best, not as a splitting criterion.

Figure 4.4 is a graphic representation of the search tree generated from splitting the largest range using the described heuristic. Same as with Section 4.1.2 the

returned rule may be generated in any place in the search tree rather than being a leaf.

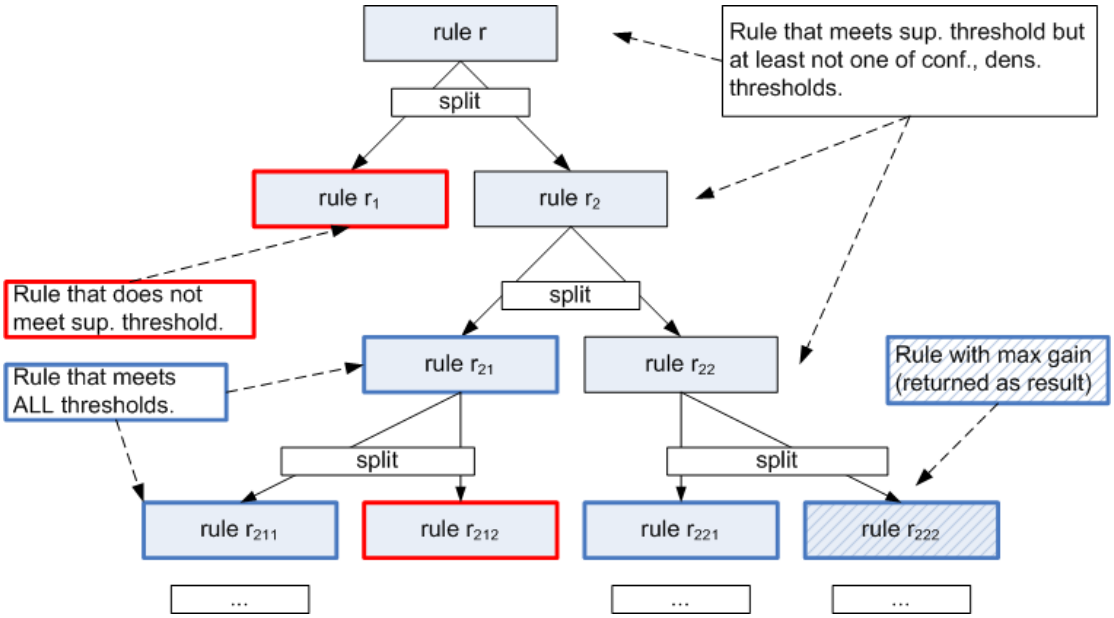


Figure 4.4: Tree generated by maximum gain splits.

4.1.4 All Confident Heuristic

Sections 4.1.1, 4.1.2 and 4.1.3 presented three different approaches to mining range based association rules driven by a different splitting method for generating a search tree, and an interestingness measure for defining what the result will be in each case. In this section, the described method takes a different approach, by using the same methodology for splitting the largest range and generating the search tree but mining a set of rules from each tree. Based on the description of the research problem these rules will have to meet the $\sigma_{min}, \gamma_{min}, \delta_{min}$, furthermore we define an additional criterion that is required in this method.

The motivation for this method is to mine as many useful rules as possible from the search tree instead of only the best based on a specific interestingness measure. This is a point that becomes more clear in the *voting method* employed for prediction experiments in Chapter 5. The generation, of a set of rules, that result from the splitting of a single largest range presents the issue of rule redundancy according to Definition 3.3.8. Because in any given subtree of the splitting tree the root is the rule with largest support and any rule represented by a node in the same subtree can only cover a subset of tuples of those covered by the root, then

if the child node is a rule of decreased confidence and density compared to the root adding it to the resultset adds no useful information to the results. Therefore any rule that meets the described criteria is only added in the results *if and only if* it is not redundant. Based on the nature of the splitting methodology any rule node has to be checked against rules represented by nodes in the same branch of the tree as the rest will be covering different tuples. An example of such case is represented in Figure 4.5.

Algorithm 4.1.6 presents the aforementioned method that mines all range-based association rules that meet the minimum thresholds and are not redundant, the process of splitting is using Algorithm 4.1.4.

Algorithm 4.1.6: *All-confident rule generation*

Input: A largest i-range l, parameters σ_{min}, γ_{min} and δ_{min} and the target class label c_t

Output: The ruleset R

1 $R \leftarrow \emptyset$
2 $Q \leftarrow \varnothing$
3 $Q \leftarrow Q.enqueue(l)$
4 **while** $Q \neq \varnothing$ **do**
5 $q \leftarrow Q.dequeue()$
6 **if** $\sigma(q) \geq \sigma_{min}$ **then**
7 **if** $\gamma(q) \geq \gamma_{min} \wedge \delta(q) \geq \delta_{min}$ **then**
8 **if** q *not redundant in* R **then**
9 $R \leftarrow R \cup q$
10 $< r_1, r_2 > \leftarrow$ maxConfidenceSplit(q)
11 $Q \leftarrow Q.enqueue(r_1)$
12 $Q \leftarrow Q.enqueue(r_2)$
13 **return** R

The differences of Algorithm 4.1.6 compared to previously described methods are in the returned result which is a ruleset R instead of a single rule and the additional redundancy check in step 8. In more detail, the algorithm starts by assigning R an empty set in step 1 and adding the largest range l to the queue in step 3. The algorithm, as in the previous methods, stops expanding the tree for any rule that does not meet the support threshold in step 6. In step 7 the rule being currently considered is checked to meet the given thresholds for confidence, density and in

step 8 for redundancy. Note that according to the description above only parent rules may render their children redundant and therefore there is never the need of removing a rule that was already added to R. Regardless if the rule was added to R in step 9, provided that minimum support is met it is split in steps $10, 11$, and 12 into two new rules, therefore expanding the search tree.

Figure 4.5 is a graphic representation of the search tree generated from splitting the largest range using the described heuristic. Same as with Section 4.1.2 a rule included in the resulting set may be generated in any place in the search tree rather than being a leaf.

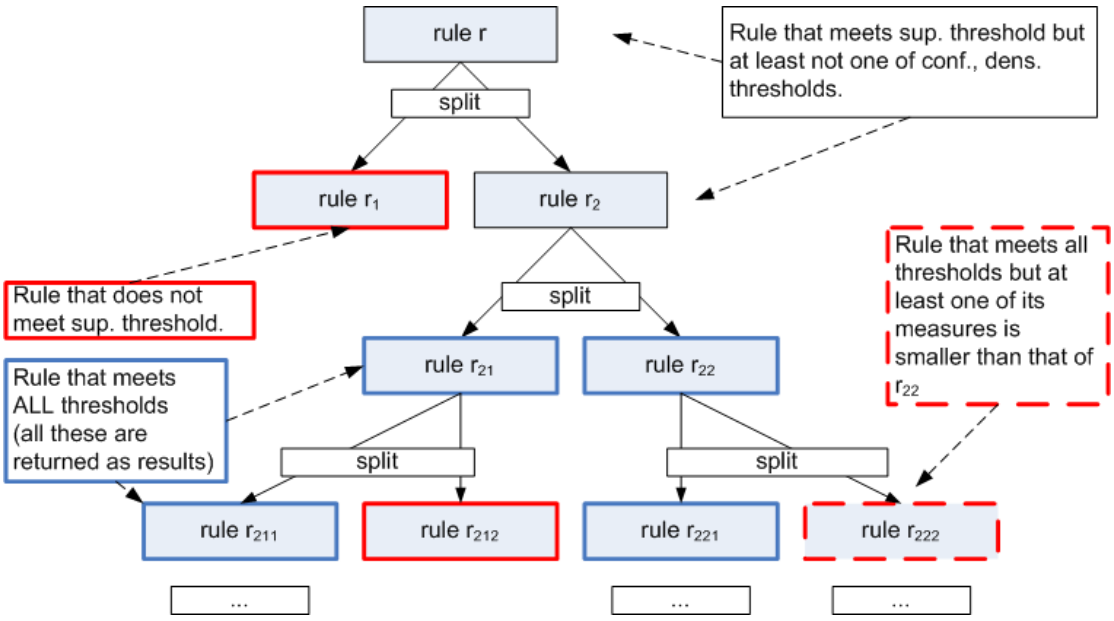


Figure 4.5: Tree generated by maximum gain splits.

The method presented in this section returns a set of non-redundant rules from the tree generated by splitting each largest range. This is a distinctive difference when comparing it with the other three methods. Due to the rule redundancy property each of the resulting rules represents a different branch in the tree, a different rule mined from the same original largest range. This property is important in the data characterization problem because of the utility of having more than one rules to describe the - possibly - more than one characteristics of a given data range.

4.2 Summary

This chapter has presented four(4) different heuristics used for splitting the largest ranges generated using the algorithms originally described in Chapter 3. Each heuristic method is driven by a different interestingness measure that it attempts to optimise while meeting the thresholds provided. The splitting method also varies depending on whether a split is performed for optimal support or for optimal confidence.

Each of the four methods creates a solution space of the rules generated by splitting the original largest range and while most methods select a single rule from that space one of the methods described returns a set of all the non-redundant rules that meet the given criteria. The following chapter evaluates the impact of the aforementioned differences by comparing experimental results of the different methods described here.

Chapter 5

Evaluation of Range-Based Classification Rule Mining

In this chapter a series of experiments are performed in order to evaluate how range-based classification rule mining addresses the original problem described in this thesis. The experiments described examine the effect on the resulting rules of different thresholds for the newly introduced density measure. The different proposed methods in Chapter 3 are compared in terms of prediction accuracy, the comparison includes results for modern solutions *C4.5* and *RIPPER* that can also generate range-based rules. Results are presented for other established rule measures besides pure prediction accuracy. Finally, the ability to mine characterization rules using the presented methods is evaluated.

The following sections describe these experiments and the corresponding results.

5.1 Data Description

A number of publicly available datasets are used for the method evaluation [24, 33, 58, 80, 82, 104]. All datasets were selected from the UCI repository, [1]. The selected datasets are amongst the most popular datasets in the research community and vary in tuple and attribute size as well as the nature of their numerical attributes and the number of different class labels. Table 5.1 contains a summary of the different characteristics of each of the aforementioned datasets.

Dataset	Tuples	Attributes	Attribute Types	Classes
Breast Cancer (Diagnostic)	569	30	*Real*	2
Ecoli	336	7	*Real*	8
Glass	214	9	*Real*	7
Image Segmentation	2310	19	*Real*	7
Iris	150	4	*Real*	3
Page Blocks	5473	10	*Integer, Real*	5
Waveform	5000	21	*Real*	3
Wine	178	13	*Integer, Real*	3
Winequality-Red	1599	11	*Real*	11
Winequality-White	4899	11	*Real*	11
Yeast	1484	8	*Real*	10

Table 5.1: Datasets

5.2 Classification Experiments

This Section presents the results of a series of experiments to evaluate the performance of the developed methods in a classification task. The datasets presented in Section 5.1 are used for prediction and the results are compared against established association rule mining algorithms. Section 5.2.1 presents how the newly introduced measure of density affects the prediction results whereas in Section 5.2.2 the designed solution is compared to the prediction accuracy of the *RIPPER* algorithm [22] and *C4.5* [92].

5.2.1 Density Effect

Description

The method presented in Chapter 3 relies on using three interest measures for the extracted association rules. The two traditional measures of *support, confidence* and a new one defined as *density*. The user defines threshold values for each one of these measures which must be met by all rules. The experiments presented in this section study the effect of the density threshold value

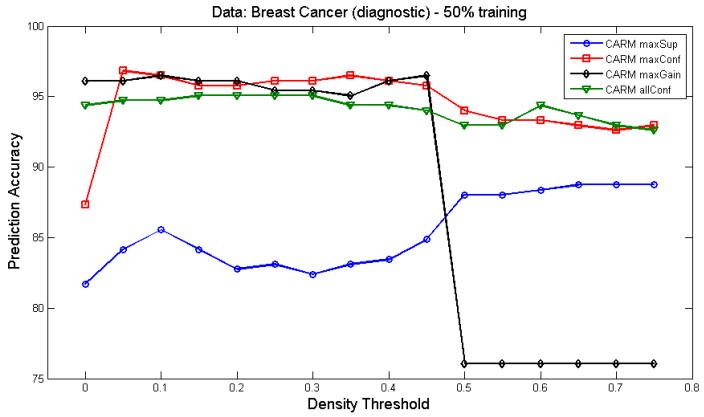
for given $< support, confidence >$ values. The best pair of values for the $< support, confidence >$ thresholds have been experimentally determined.

In order to examine how the different values of δ_{min} affect the prediction accuracy of the generated rules a series of prediction experiments are performed for each dataset described in Section 5.1. For a given percentage of the data to be used for training, ranging from 50% to 90% using 10% increments, an optimal set of values, when not considering density, for $\sigma_{min}, \gamma_{min}$ is determined experimentally. Using the aforementioned setting the δ_{min} threshold is assigned different values $\in [0, 0.75]$ and the resulting prediction accuracy is plotted for each case. These experiments are performed using all four(4) methods as described in Chapter 4.

The results presented in Section 5.2.1 determine the best values to use for the density threshold (δ_{min}) when performing experiments on prediction accuracy. As explained, δ_{min} is one of the defined thresholds that all the rules must meet, therefore the following experiments explore the effect that the newly introduced measure can have to the resulting rules when this minimum requirement increases gradually. Furthermore, rules that achieve higher density are more suitable for the data characterization problem therefore the following experiments demonstrate the loss in prediction accuracy for each method when attempting to mine range-based association rules for characterization purposes.

Results

The graphics in this Section present the results for the experiments described above, in Section 5.2.1 for all the datasets used in this chapter. Each graphic represents all the five cases for different training data percentages for comparison.



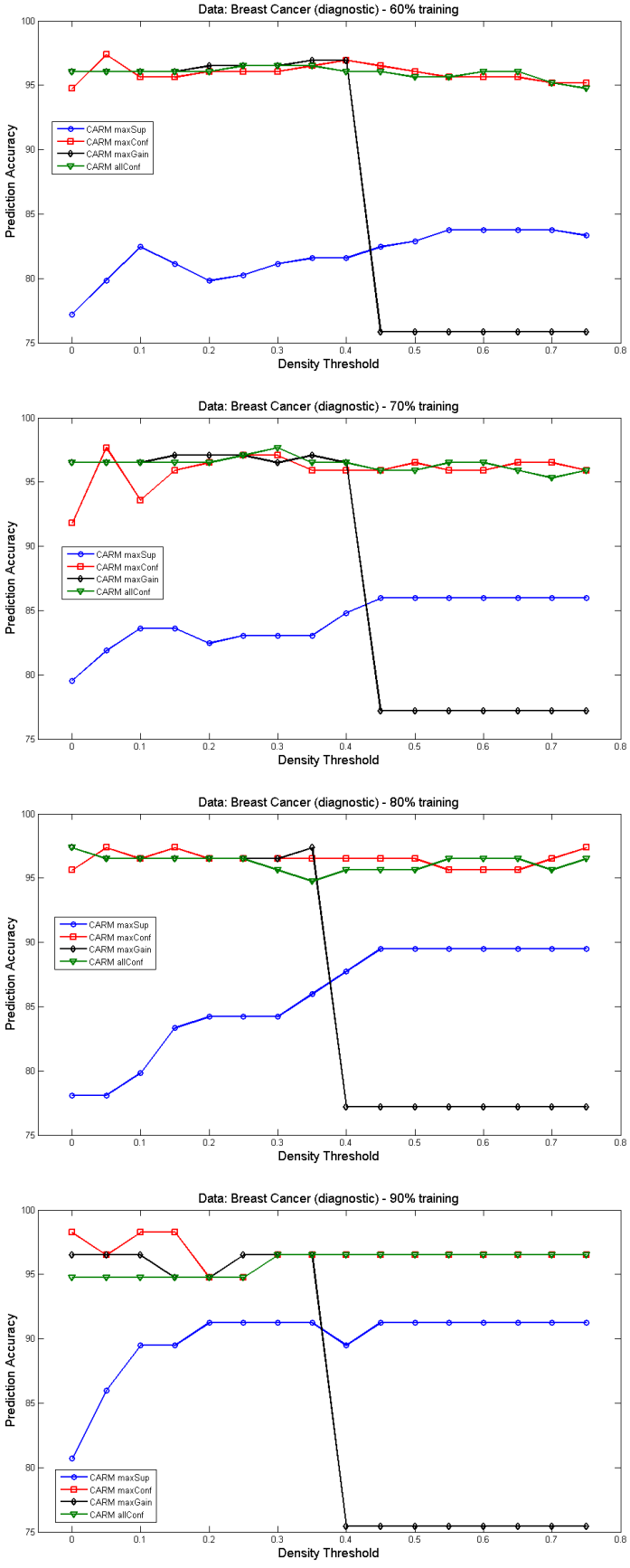
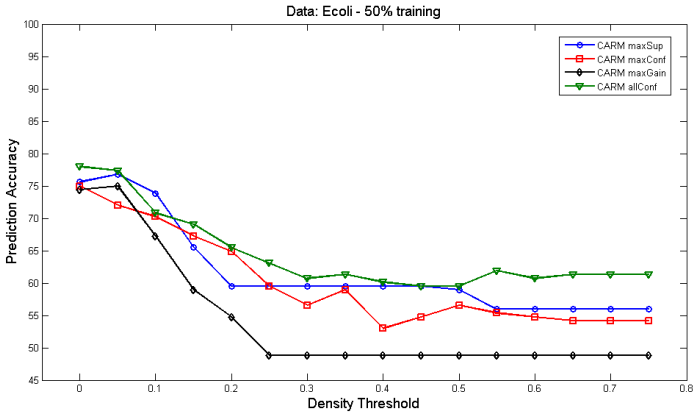
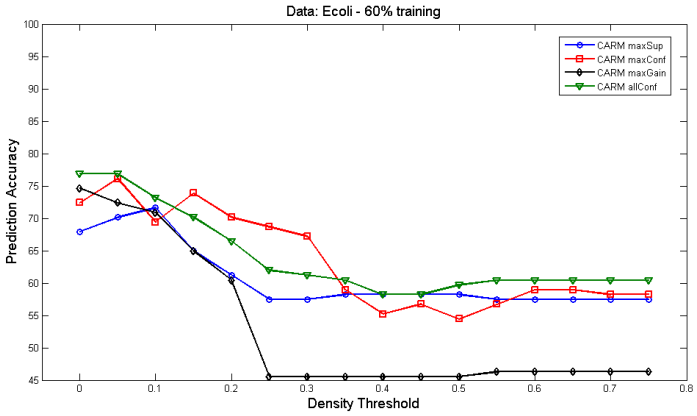


Figure 5.1: Density effect on prediction accuracy of breast cancer data.

Increasing the density threshold reduces the number of rules that can be generated. For some data sets this means that the prediction accuracy is reduced due to the reduced number of generated rules that can be used for prediction. In the case of the breast cancer dataset, however, the prediction results for methods 2 and 4 are quite resilient to density threshold increases. Method 3 demonstrates different behaviour where its accuracy drops significantly past a certain density threshold. Method 1 has the lowest accuracy of all methods but is the only one that benefits from density threshold increases. This is because method 1 is mining rules of increased support but does not attempt to increase confidence beyond the specified γ_{min} therefore in this case the increased density threshold improves the confidence amongst the resulting rules and the overall prediction accuracy.





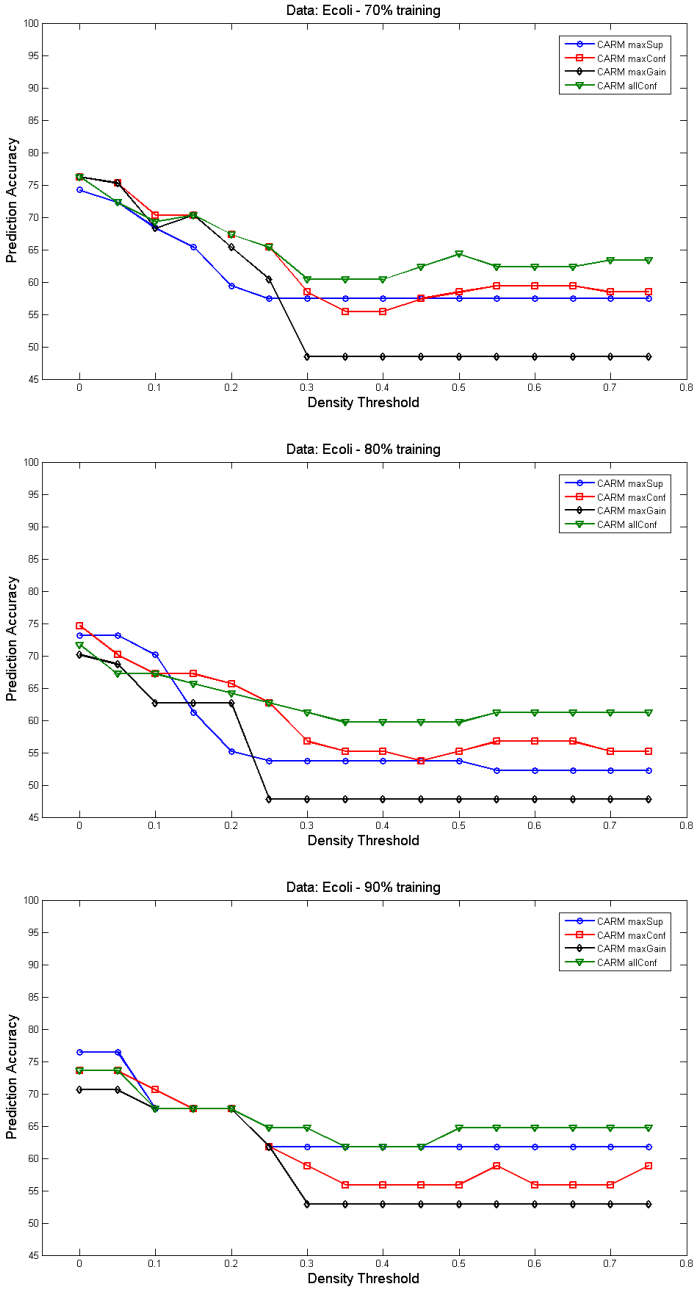
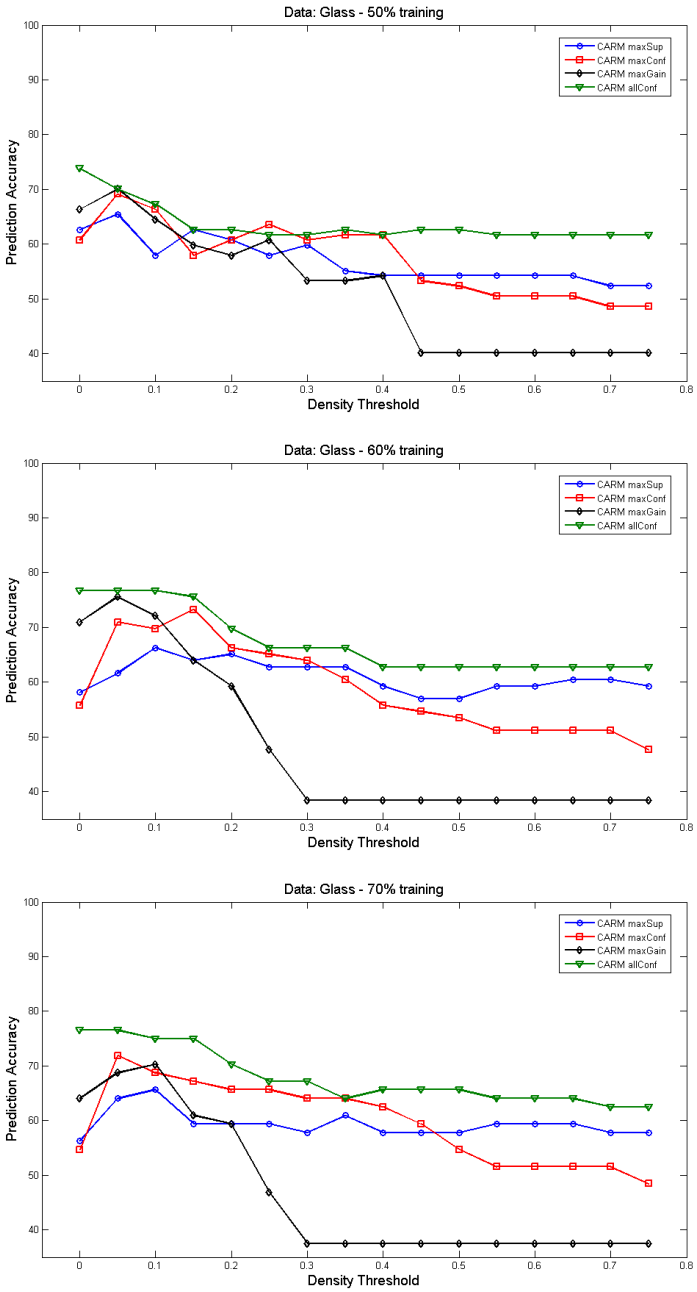


Figure 5.2: Density effect on prediction accuracy of ecoli data.

The effect of increasing δ_{min} is clear for all methods in the case of the ecoli data. Prediction accuracy decreases as the threshold is increased with the only difference being the rate of the decrease which seems to be higher for method 3. The conclusion for the ecoli dataset is, however, clear, higher precision accuracy is achieved by using a small threshold for density.

This dataset is a characteristic example where mining dense rules proves difficult.

Therefore setting a high threshold for density effectively removes from the dataset certain rules that were good classifiers (had a good effect on prediction accuracy). Although this effects the prediction results for the proposed method as shown in Figure 5.14, it is not necessarily an undesired effect since the aforementioned rules are useful predictors but not so useful when the desired output is a set of characterization rules that describe a dense data area.



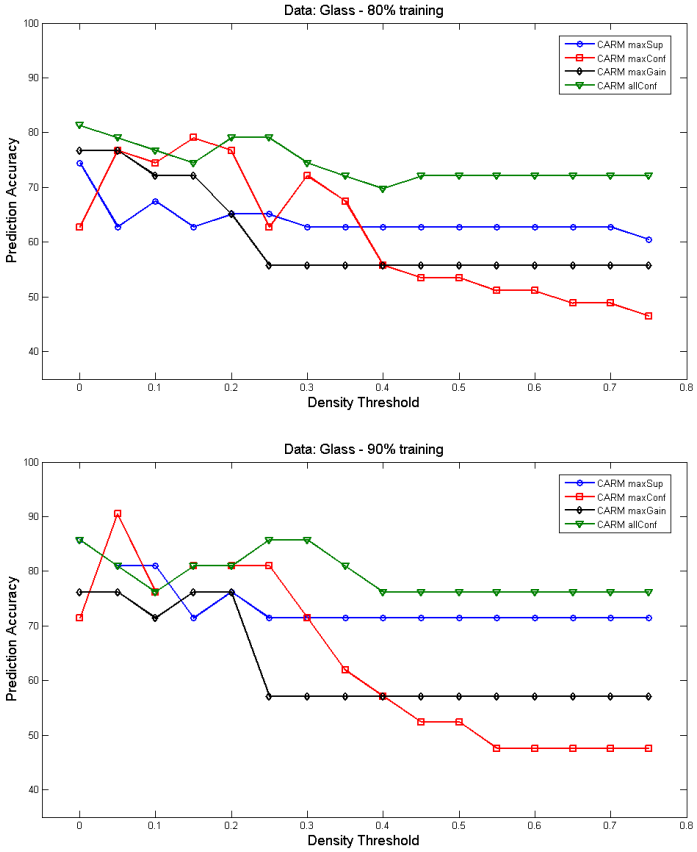
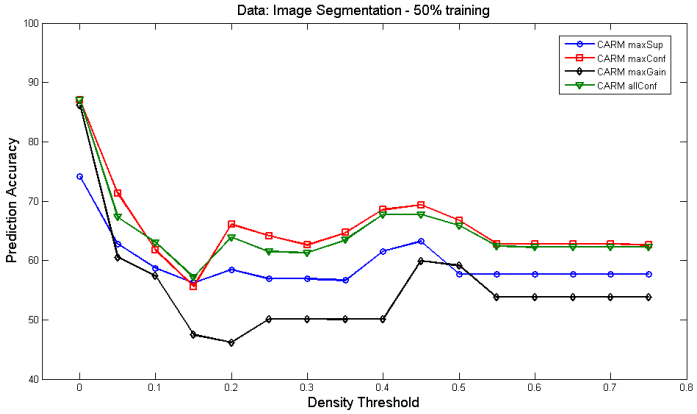
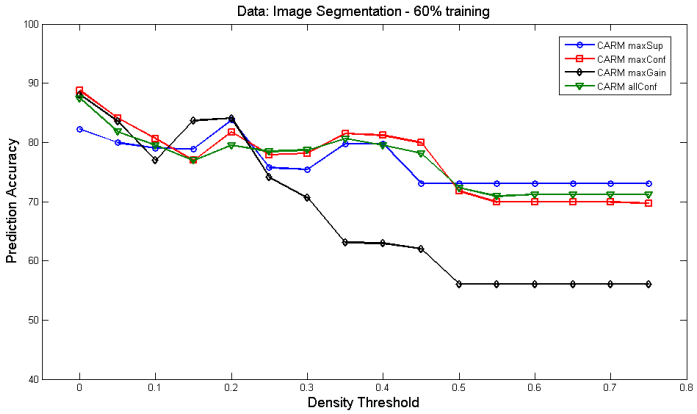
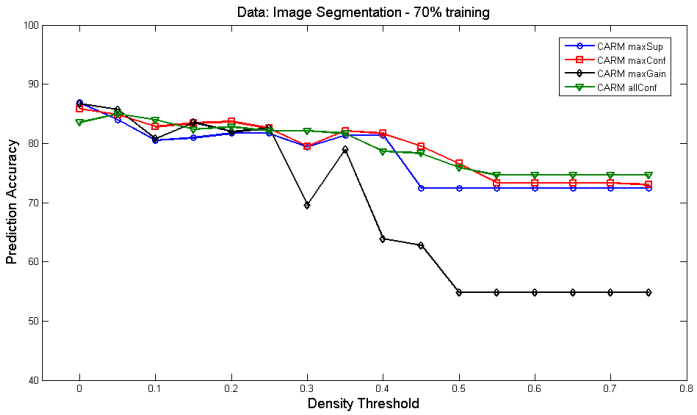


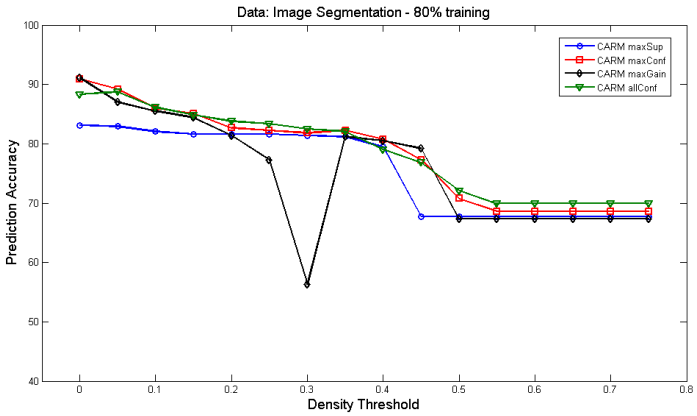
Figure 5.3: Density effect on prediction accuracy of glass data.

The density effect experiments for the glass dataset lead to similar results as for the ecoli data. Lower values of δ_{min} achieve the best prediction accuracy for all methods. Method 3, specifically, is affected more drastically as prediction accuracy decreases faster when the threshold increases to ≥ 0.2. It is worth noting, however, that unlike the ecoli data results the method 2 demonstrates a loss in prediction accuracy similar to that of method 3. The cause of this is that the most confident rule changes as the density threshold increases (if it did not the prediction results would be the same) whereas method 4 that mines all the rules that meet the desired thresholds manages to achieve a relatively constant prediction accuracy. These results are a product of the utility of mining overlapping rules that allow for accurate prediction even when the most confident rules are removed from the results.


Data: Image Segmentation - 50% training
CARM maxSup
CARM maxConf
CARM maxGain
CARM allConf
Prediction Accuracy
Density Threshold


Data: Image Segmentation - 60% training
CARM maxSup
CARM maxConf
CARM maxGain
CARM allConf
Prediction Accuracy
Density Threshold


Data: Image Segmentation - 70% training
CARM maxSup
CARM maxConf
CARM maxGain
CARM allConf
Prediction Accuracy
Density Threshold


Data: Image Segmentation - 80% training
CARM maxSup
CARM maxConf
CARM maxGain
CARM allConf
Prediction Accuracy
Density Threshold

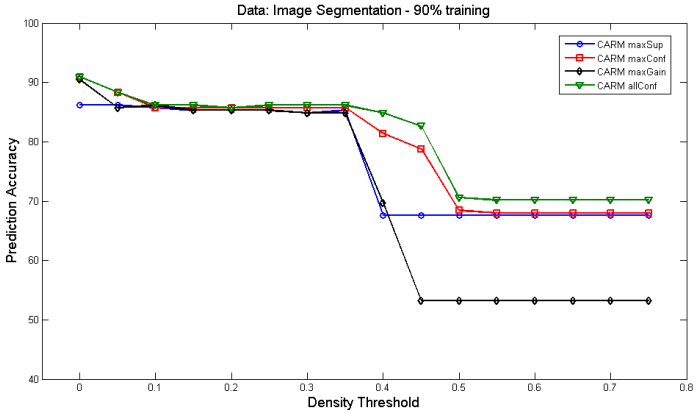
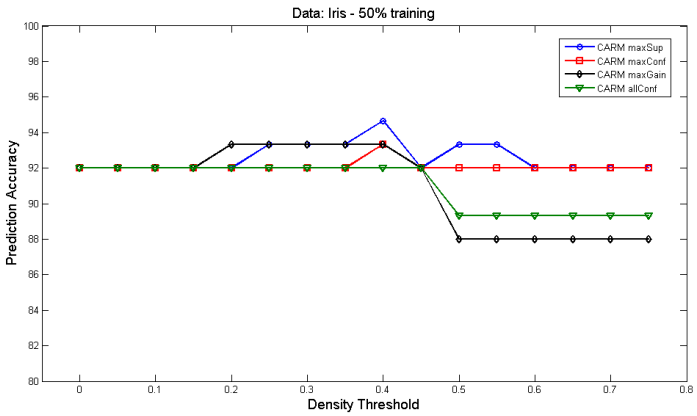


Figure 5.4: Density effect on prediction accuracy of image segmentation data.

A very low density threshold results in best accuracy prediction in the case of image segmentation data. Interestingly, in the case of using 50% of the data for training the prediction results for $\delta_{min} \geq 0.25$ are better than for $\delta_{min} \in [0.1, 0.15]$. These results mean that rules r with density $\delta(r) \in [0.1, 0.15]$ have very low accuracy which is why the prediction results improve when these rules are excluded by increasing the threshold. This case is different than the previous datasets shown above. It is the first example where the most confident rules and the rules with highest information gain are not the best predictors but it only occurs when using $\leq 70\%$ of the data for training.



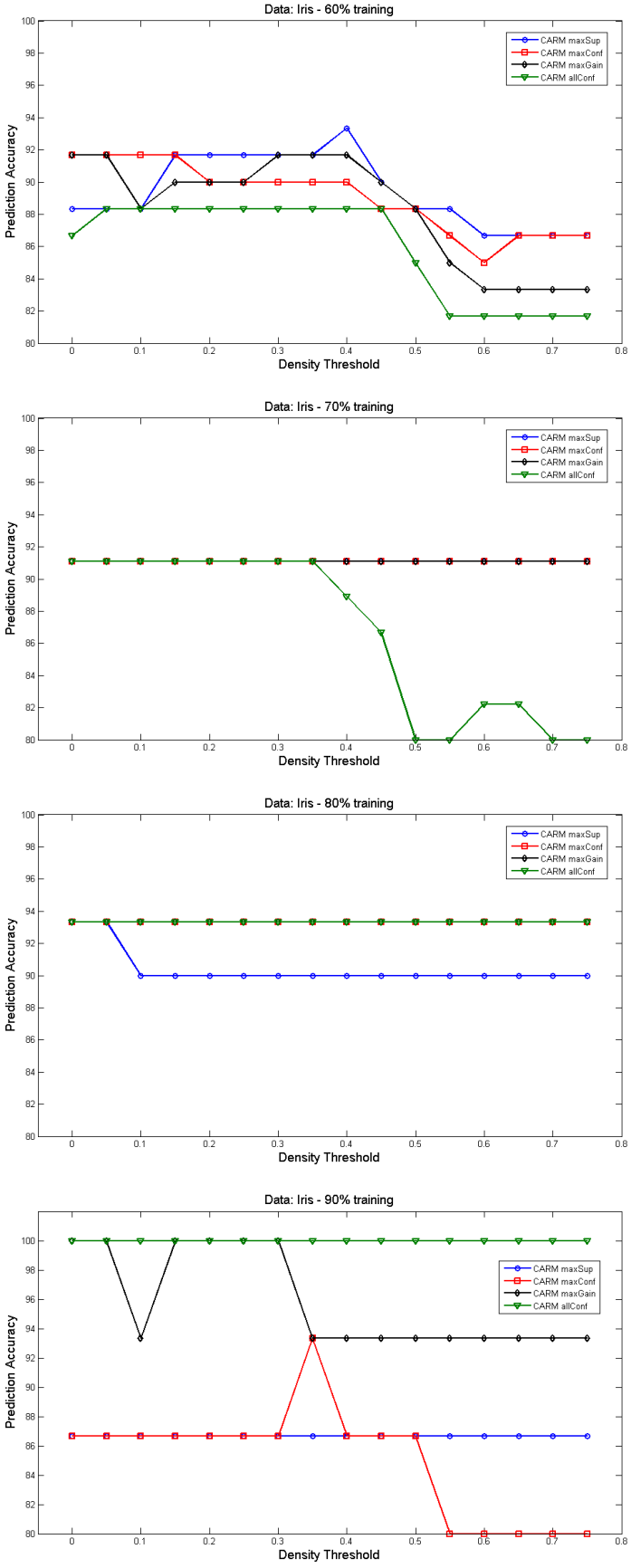
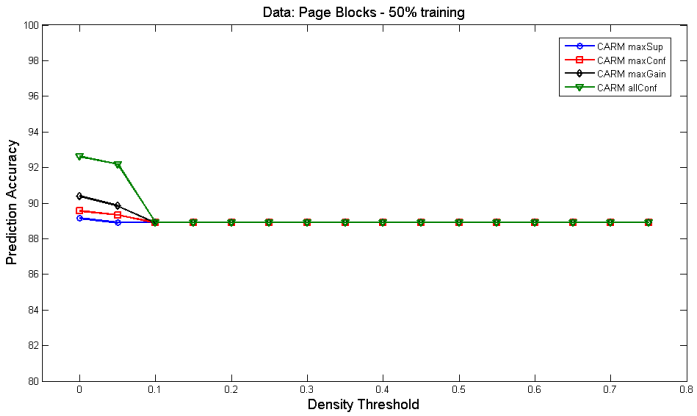
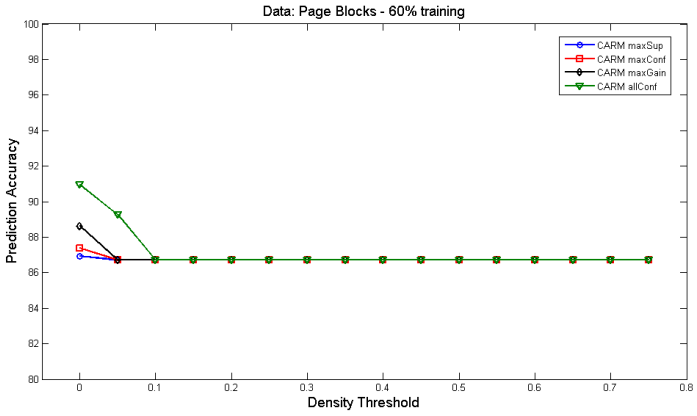


Figure 5.5: Density effect on prediction accuracy of iris data.

The iris dataset is a case where the designed solution performs very well in terms of accuracy as can be seen in Figure 5.17. Figure 5.5 shows all methods maintaining a high level of prediction accuracy for $\delta_{min} \leq 0.35$. These results indicate that the rules mined from the iris data have a high density even when density is not used as a criterion, therefore separation between the different classes is not difficult. The above is verified by the high accuracy achieved by the designed methods as well as other mining solutions in Section 5.2.2.

Density effect in this case is limited in most cases and only affects method 4 when using 70% of the data for training. The iris dataset has historical value but does not seem to be representative of modern data collections. Furthermore it is a clear case of a dataset that was originally constructed as a classification problem.





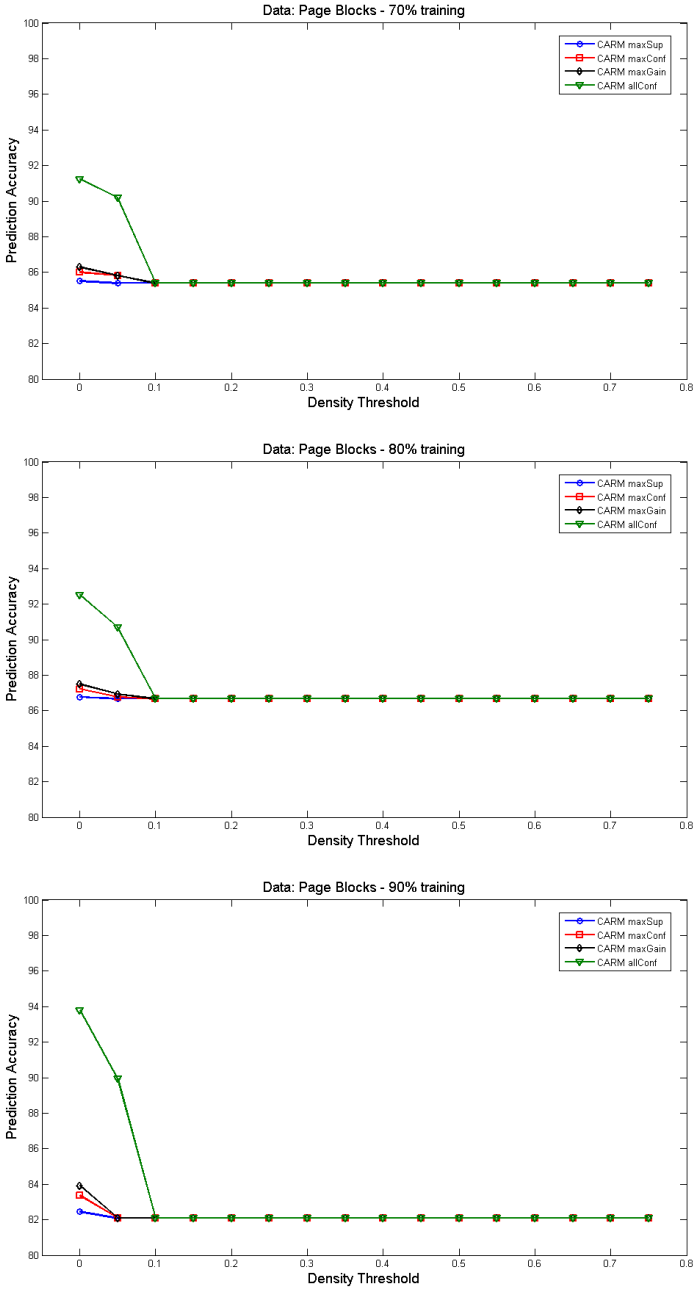
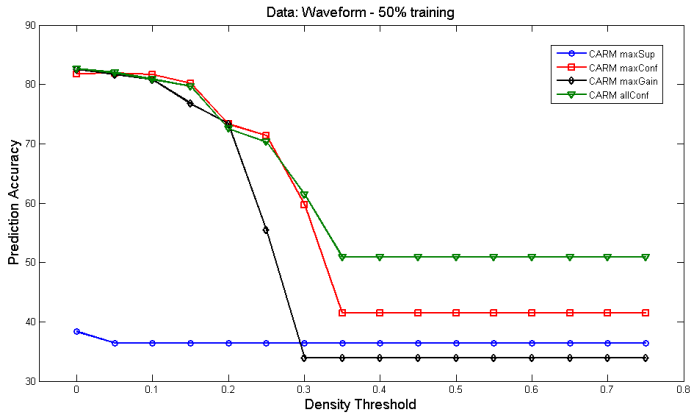
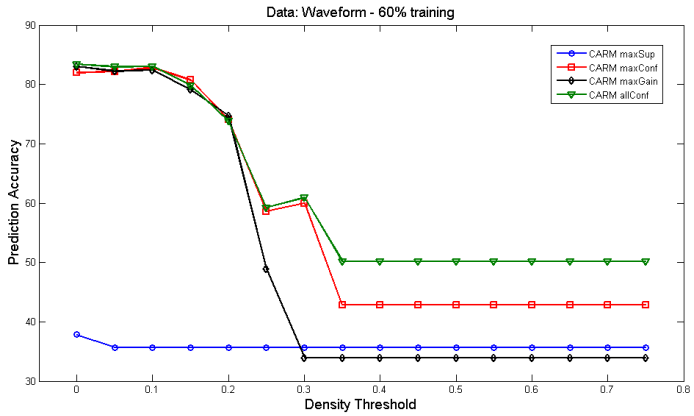


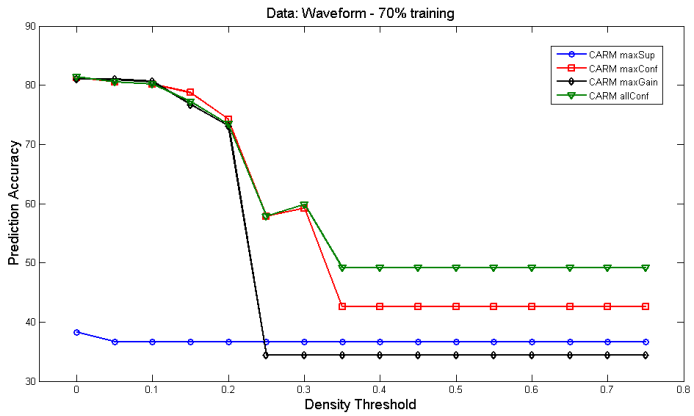
Figure 5.6: Density effect on prediction accuracy of page blocks data.

Increasing the density threshold has an interesting effect on prediction accuracy when using the page blocks dataset. As in other cases the best results are achieved with a very low threshold because in that case the algorithm has more options to construct range-based rules. However, when $\delta_{min} \geq 0.1$ all methods achieve the exact same accuracy which remains steady as the threshold increases, the actual value only depends on the percentage of data used for training. This is due to the nature of the dataset which consists of one dominant class label with a frequency

percentage of 89.8%. The results shown in Figure 5.6 demonstrate that the rules for every class label except the dominant one fail to meet the density criterion when the threshold increases. With a low δ_{min} method 4 achieves accuracy that is greater than the frequency of the dominant class indicating that the all-confident method is able to mine rules for the low frequency class labels as well as the dominant one.







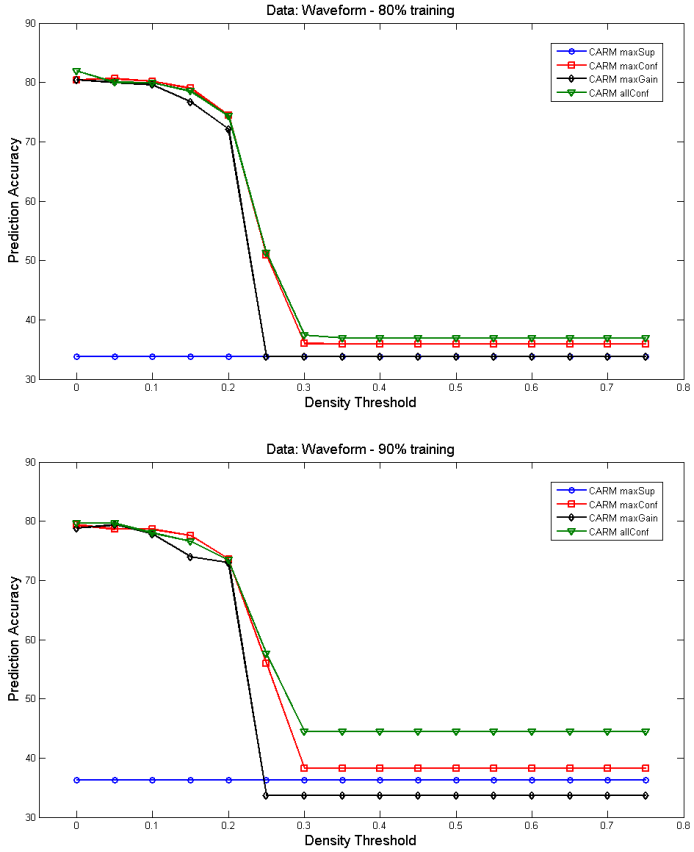
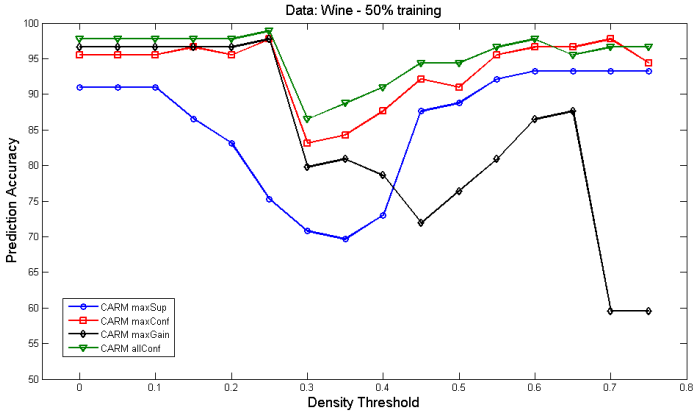
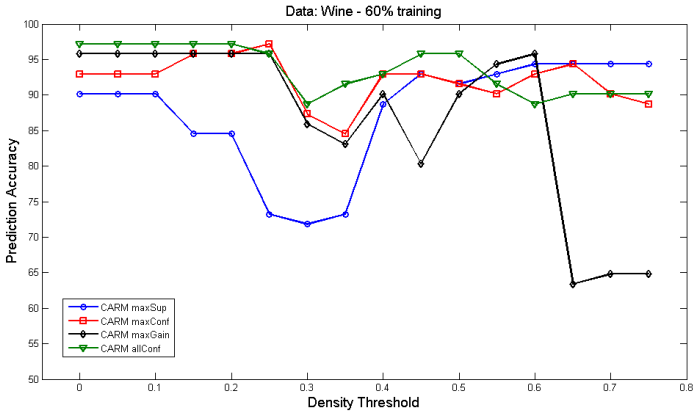
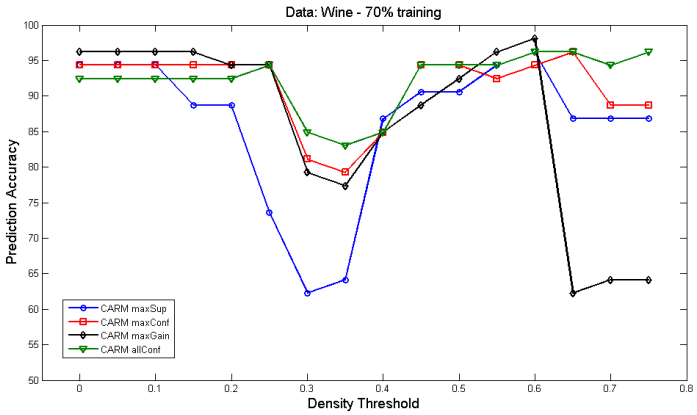


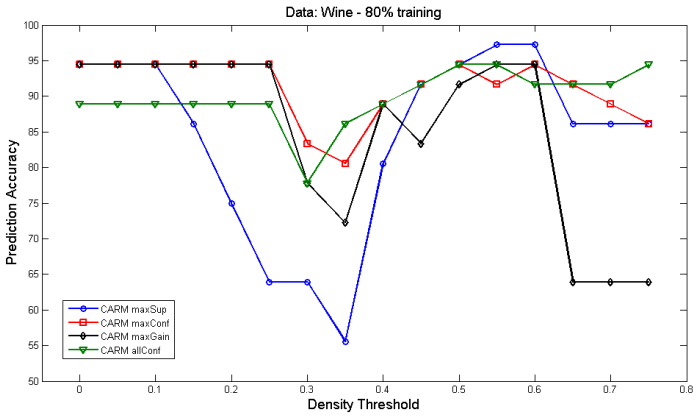
Figure 5.7: Density effect on prediction accuracy of waveform data.

Figure 5.7 shows the effects of increasing the density threshold to be similar to the majority of the experiments in this Section. Low values of $delta_{min}$ result in better accuracy. However, as can be seen in Figure 5.7 the loss in accuracy is low while $\delta_{min} \leq 0.2$ but significant when $\delta_{min} \in [0.25, 0.3]$. Even though the impact in this case is not identical for all methods, the results indicate the same behaviour as with the page blocks dataset where there is a characteristic maximum value for δ_{min}. Because the waveform data contain class labels of equal frequencies the loss of accuracy affects all classes and is higher compared to the page blocks data.


Data: Wine - 50% training
Prediction Accuracy
Density Threshold
CARM maxSup
CARM maxConf
CARM maxGain
CARM allConf


Data: Wine - 60% training
Prediction Accuracy
Density Threshold
CARM maxSup
CARM maxConf
CARM maxGain
CARM allConf


Data: Wine - 70% training
Prediction Accuracy
Density Threshold
CARM maxSup
CARM maxConf
CARM maxGain
CARM allConf


Data: Wine - 80% training
Prediction Accuracy
Density Threshold
CARM maxSup
CARM maxConf
CARM maxGain
CARM allConf

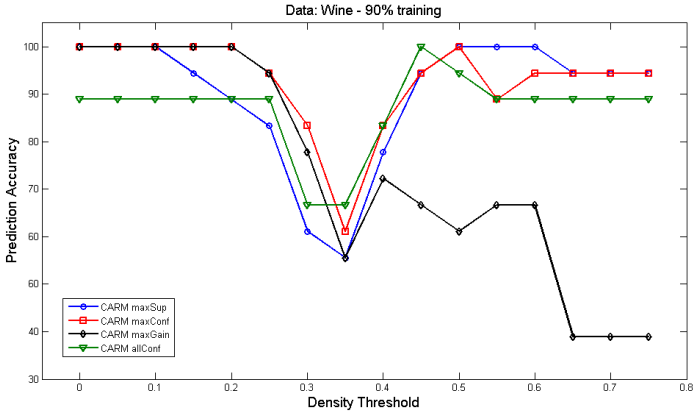
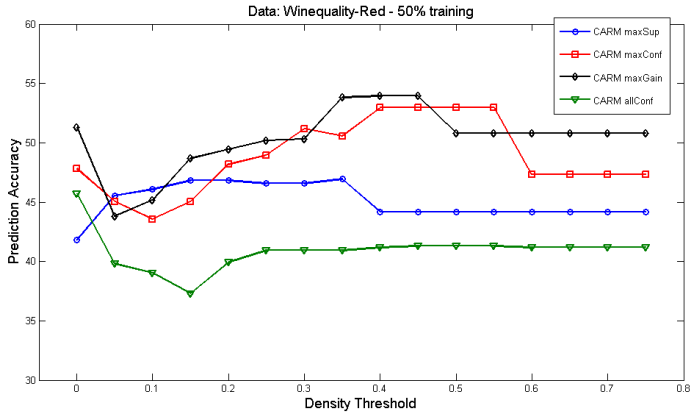


Figure 5.9: Density effect on prediction accuracy of wine data.

Prediction accuracy for the wine data is the highest overall of all the datasets. Figure 5.9 shows how the resulting accuracy is not affected as significantly as it was in other cases. More importantly, for methods $2, 3, 4$ increasing the δ_{min} value up to 0.25 has little to no effect to the measured accuracy . Furthermore, all methods increase in prediction accuracy for $\delta_{min} \geq 0.35$ and unlike other datasets the measured accuracy consistently increases for larger threshold values (method 3 demonstrates that behaviour only for $\delta_{min} \in [0.35, 0.6]$). These results are consistent with what can be seen in Figure 5.20 where the average results are compared to other algorithms, the wine data contain confident rules with high support and density that achieve high prediction accuracy.

These results also indicate that the classification results for the presented solution can improve by specifying a different threshold value in each case of different training data percentage. For consistency, however, the comparison in Section 5.2.2 the same set of $< \gamma_{min}, \delta_{min}, \sigma_{min} >$ values is used for all experiments on a specific dataset.



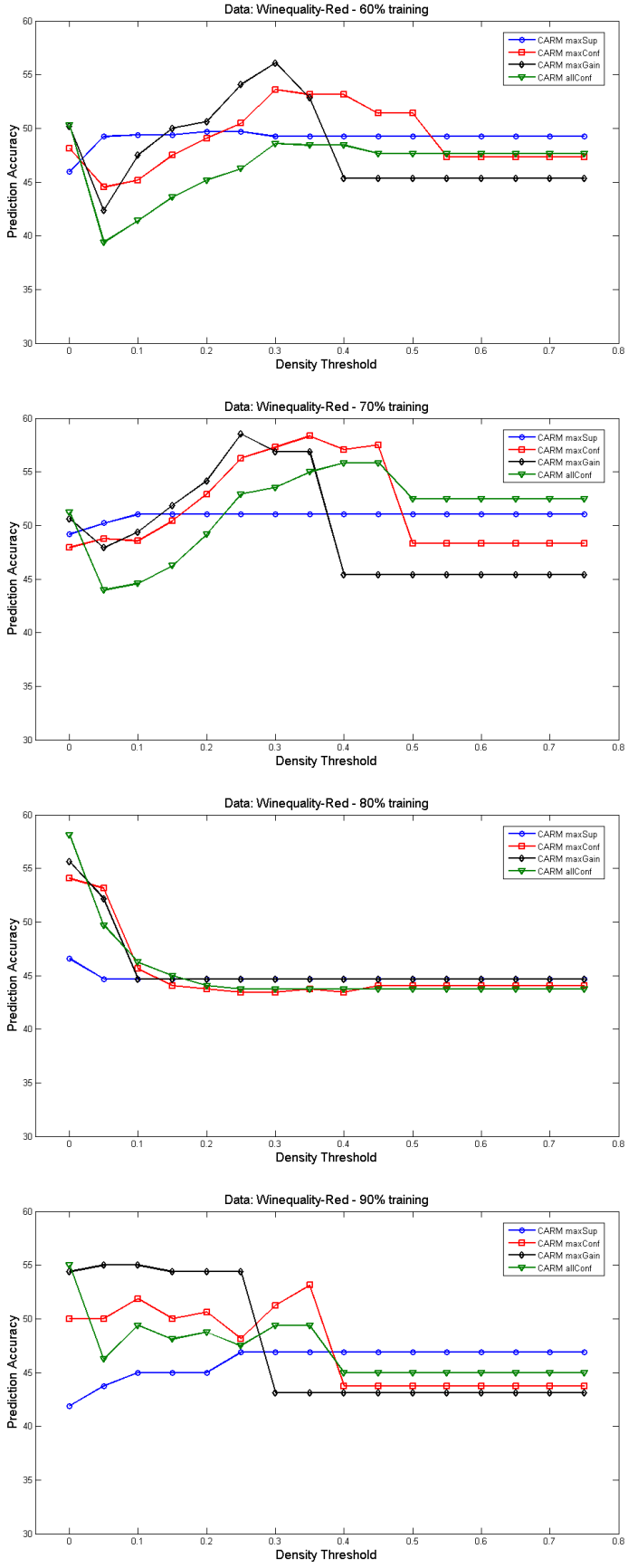
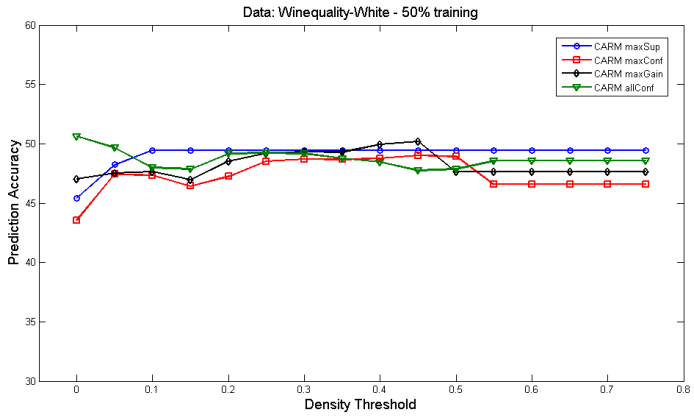
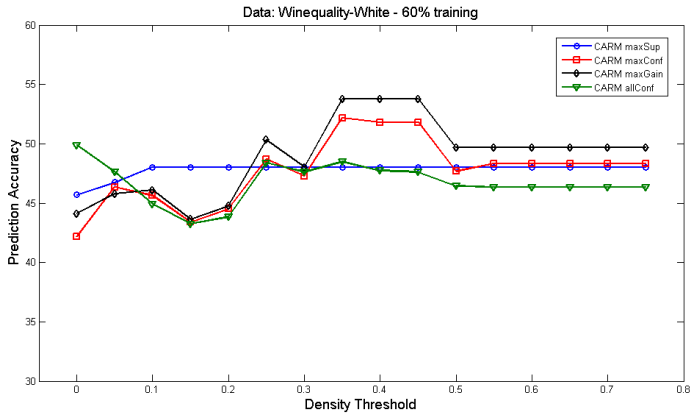


Figure 5.10: Density effect on prediction accuracy of red wine quality data.

The wine quality datasets both for red and white data have a unique property, not all the class labels are present in the dataset and the existing class labels are unbalanced. This is an important property since it increases the difficulty of prediction when the training data percentage is low. This is visible in Figure 5.10 where the accuracy in the cases of using $50\%, 60\%$ and 70% of the data for training increases as the δ_{min} threshold increases to mid-range values. When 80% of the data is used for training the best accuracy is achieved for very low δ_{min} values which is also true for methods $3, 4$ when using 90% of the data for training.

Method 1 is the exception to the above achieving relatively constant accuracy in these experiments. The unbalanced class labels produce an interesting result since method 2 that mines a single most confident range-based classification rule form every consequent bounded rule is overall more accurate than method 4, that uses the all-confident method for splitting.





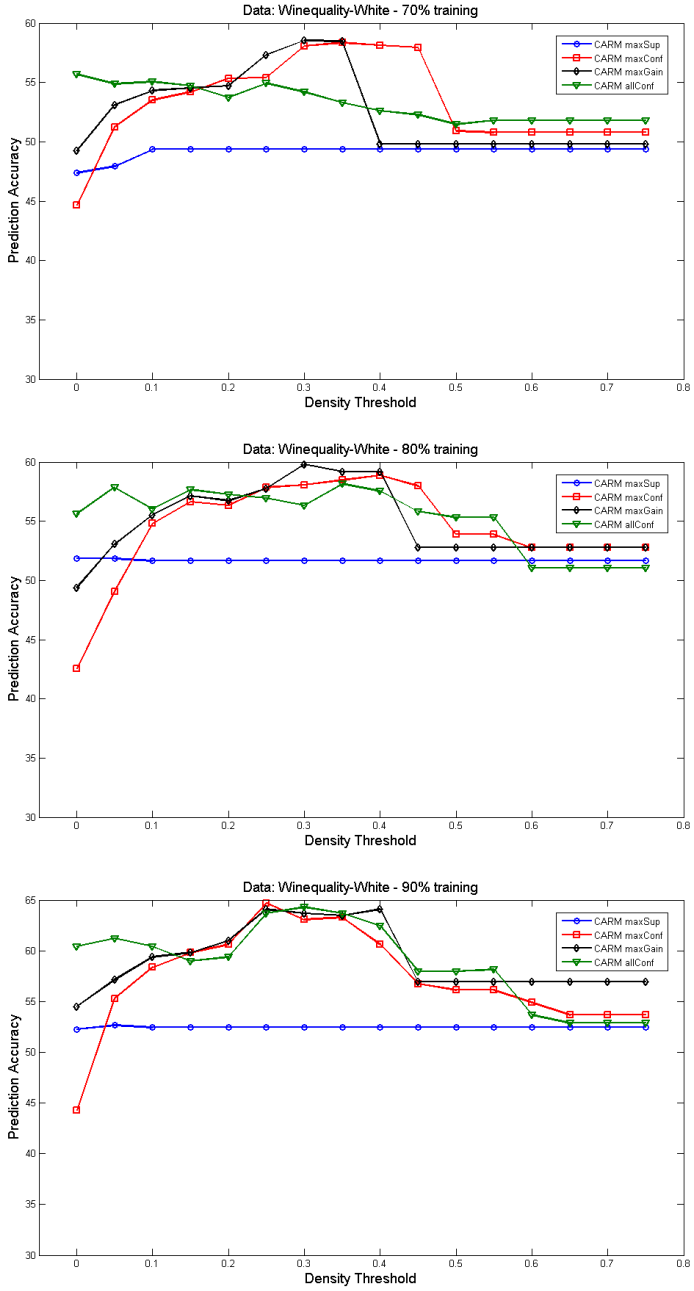
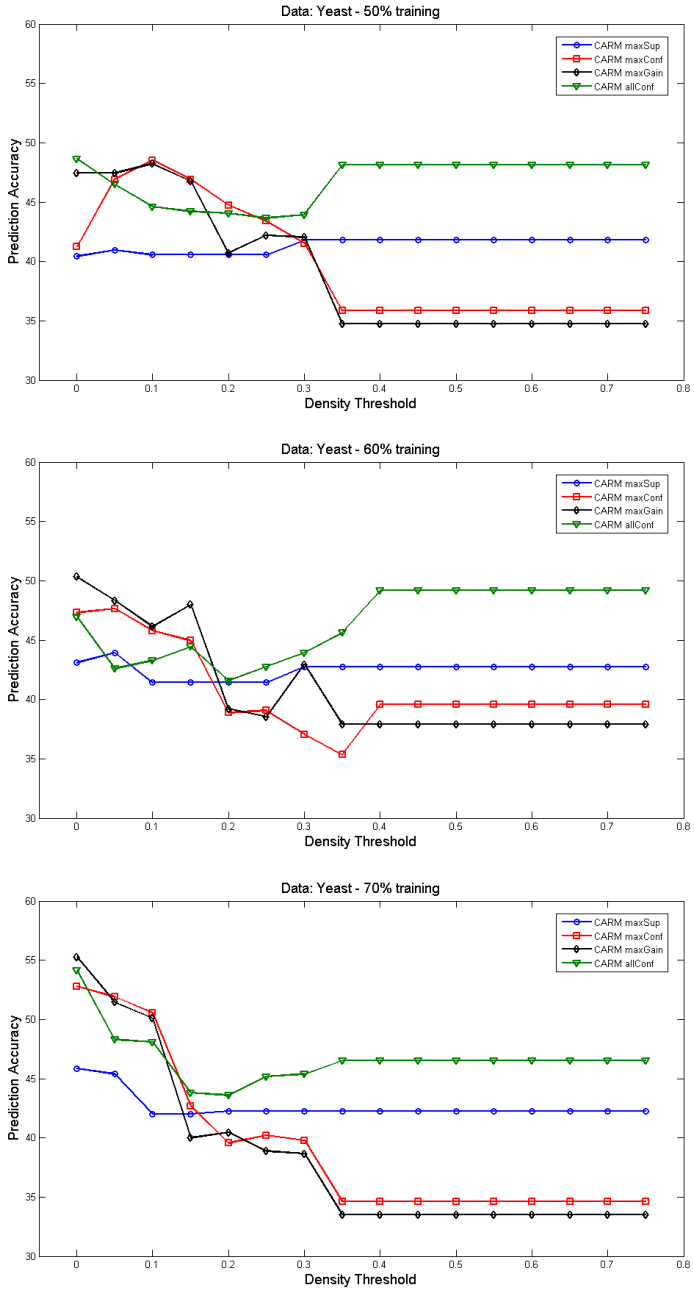


Figure 5.11: Density effect on prediction accuracy of white wine quality data.

The white wine quality dataset contains the same attributes as the red wine quality dataset but contains three times as many tuples. In this case the phenomenon of increased accuracy for mid-range values of δ_{min} is more evident as the training data percentage increases. In addition to the class imbalance that is present in this dataset as well, the reason why mid-range δ_{min} values achieve higher accuracy is related to the average accuracy achieved overall. In our results, the designed algorithm generates a large number of rules but the prediction results

shown in Figure 5.11 show that the generated rules are not suitable for prediction and that increasing the density threshold improves the prediction results. Note that the rules generated for $\delta_{min} = 0.35$ are a subset of the rules generated for $\delta_{min} = 0.05$ but filtering the lower density rules improves the accuracy achieved by the remaining rules. Based on the above, density can be a useful measure not only for characterization purposes but for prediction tasks as well when the support/confidence measures are not sufficient.



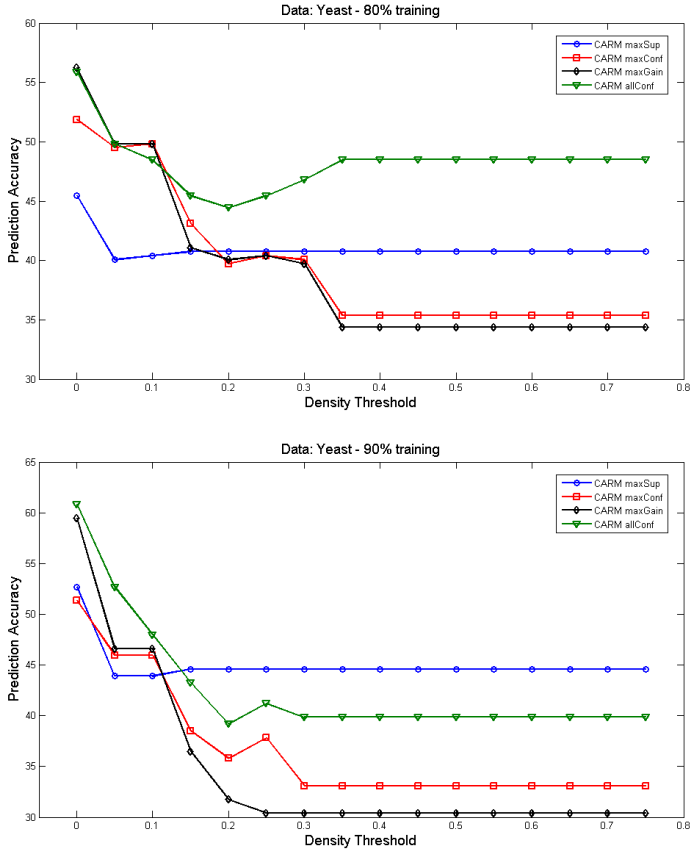


Figure 5.12: Density effect on prediction accuracy of yeast data.

Prediction accuracy, in the case of the yeast dataset is again highest when the δ_{min} threshold is set low. Figure 5.12 shows a difference in the effect of density to method 4 when compared to the other methods. The all-confident method is the only one that demonstrates an increase in prediction accuracy for $\delta_{min} \in [0.2, 0.4]$ whereas, like the other methods, it remains relatively steady for $\delta_{min} > 0.4$. Method 4 is the only designed method that from each split mines all the rules meeting the given criteria and not a single rule, therefore the improved accuracy for higher density threshold can be attributed to the positive impact on prediction, of the additional rules mined.

5.2.2 Prediction Accuracy

Description

The experiments presented in this section aim to evaluate the prediction accuracy of the developed approach. As shown in Section 5.2.1 different thresholds affect

the resulting rules significantly, therefore, before performing a comparison of the developed methods, an optimal set of values for each $< method, dataset >$ are used to compare the results for each method.

Furthermore, in order to assess, the efficiency of the algorithm as a classification method, the prediction accuracy of each method is compared against that of established rule induction solutions. The algorithms used for comparison are *Weka*'s [51] implementation of the *RIPPER* algorithm [22] which in Weka is referred as *JRip* and the implementation of an improved *C4.5* [92] referred to as *J48*.

For each dataset, experiments are performed by using 50%, 60%, 70%, 80% and 90% of the data for training and the resulting rules are used for predicting the remaining unlabeled data. As the percentage of training data increases it is expected that the prediction accuracy improves because the mining algorithms have more data to use for generating the rules. This is especially important in datasets where certain class labels are infrequent in the training data resulting in a small number of rules for these classes and consequently bad prediction results in the unlabeled data with the same class labels. However, as the training data increase in size it is also possible that the resulting rules *over-fit* the data, meaning the rules are not general enough to cover the unlabeled cases and prediction performance decreases.The training data are always selected by preserving the tuples' order (e.g. when 50% of the *Ecoli* data is used for training, this means the first 168 tuples). This way, all algorithms use the exact same tuples as the training data and are not allowed to select those tuples that provide the best results, so that the comparison is consistent. The prediction decision is using a voting table for each unlabeled tuple. For a dataset of n class labels the voting table v is an $1 \times n$ table where the v_i position of the table is the number of rules with consequent c_i that the given tuple meets. The prediction made by the algorithm is the class label with the most votes.

Table 5.2 shows an example of a voting table. In this example we attempt to classify an unlabeled tuple of a dataset with four possible class labels $\{c_1, c_2, c_3, c_4\}$. The resulting table shows that the tuple was covered by a total of 13 rules that were mined from the training data and that the majority of the rules (9) have c_3 as the rule consequent, therefore the prediction made by CARM is c_3. In the case that more than one class labels have the largest number of votes, selection between these class labels is based on their corresponding support in the training data.

c_1	c_2	c_3	c_4
0	3	9	1

Table 5.2: A possible voting table

Results

This section presents the results of the prediction experiments described in Section 5.2.2. A graphic is presented for each dataset with the results from all competing algorithms followed by a short summary of the results in each case.

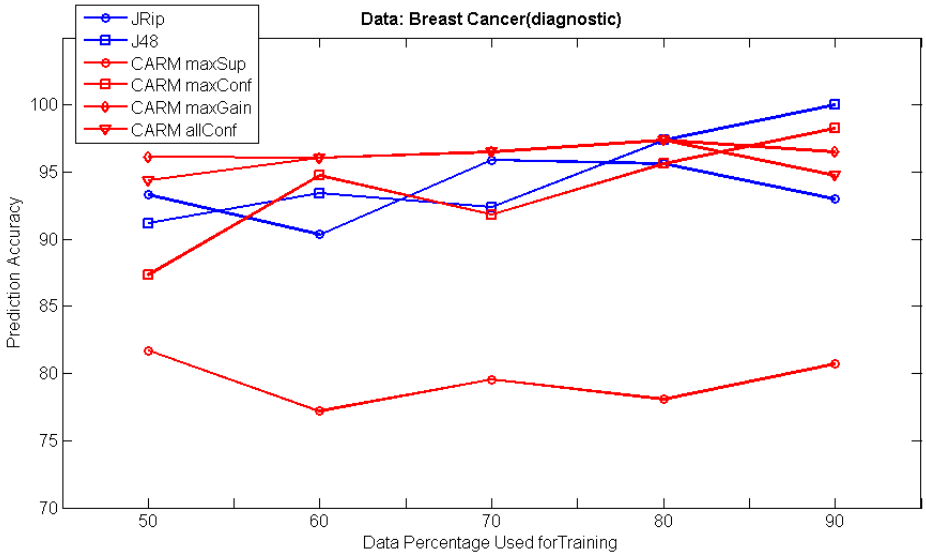


Figure 5.13: Prediction accuracy comparison for the breast cancer dataset

The maximum-support method scores lower than any other algorithm. This is expected since the mined rules are not optimised for confidence, only γ_{min} is met. The maximum-confident method performs well compared to JRip and J48 but the maximum-gain and all-confident methods clearly outperform competing solutions except for J48 when using 90% of the data for training.

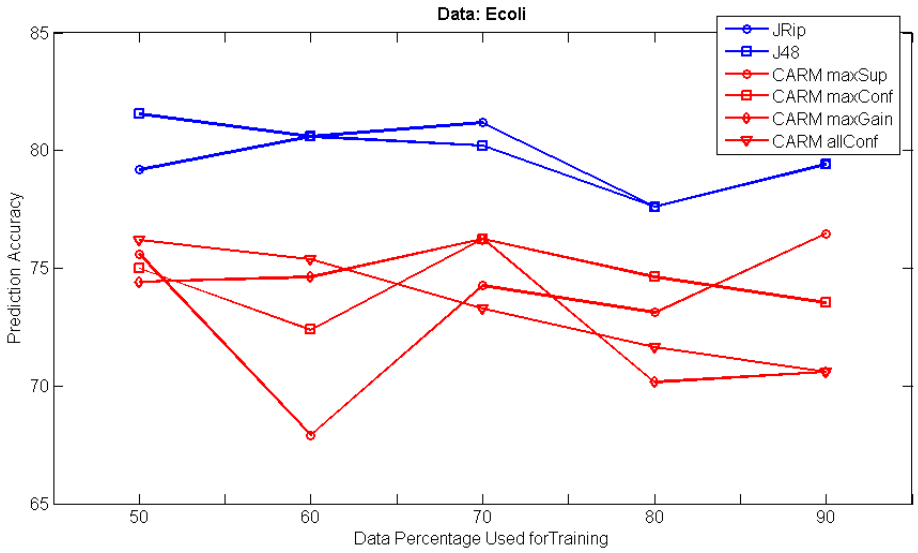


Figure 5.14: Prediction accuracy comparison for the ecoli dataset

Both J48 and JRipper outperform the designed solution for the experiment on ecoli data. This is a characteristic example of a dataset with many different, very unbalanced, class labels. The competing algorithms that are designed as classifiers ignore several of the infrequent class labels in the resulting model which helps with classification results in this case because the frequency of these classes is particularly low ([2, 20]). The designed algorithm, however, ignores class frequency and if the class label is one that good rules may be derived from it all these rules are included in the resulting rule set. This phenomenon affects the voting array however, thus reducing prediction accuracy.

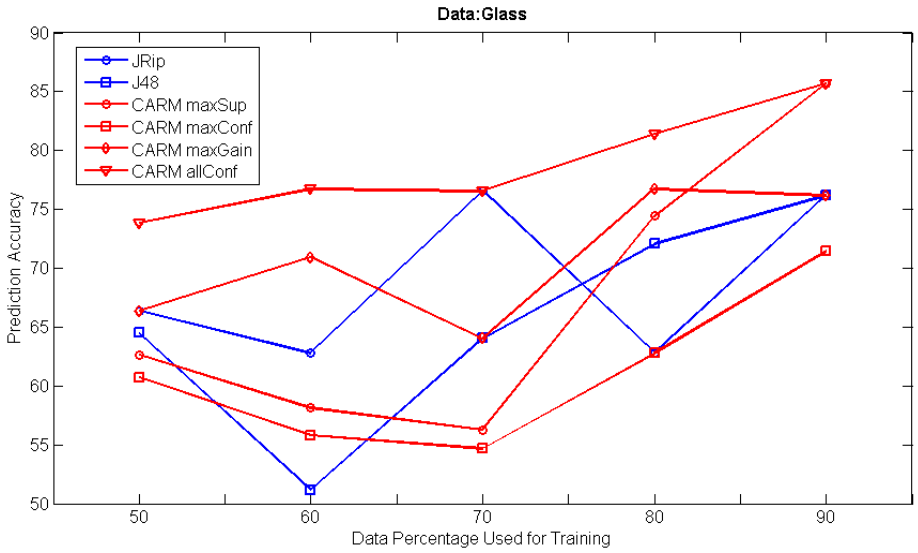


Figure 5.15: Prediction accuracy comparison for the glass dataset

The glass dataset is another dataset with a large number of different, unbalanced class labels. Because of its criminological origin the attributes are carefully selected to sufficiently describe a trace of glass material. Methods 3 and 4 outperform the competing solutions overall, more specifically, the all-confident method is consistently more accurate than 70% regardless of the data percentage used for training.

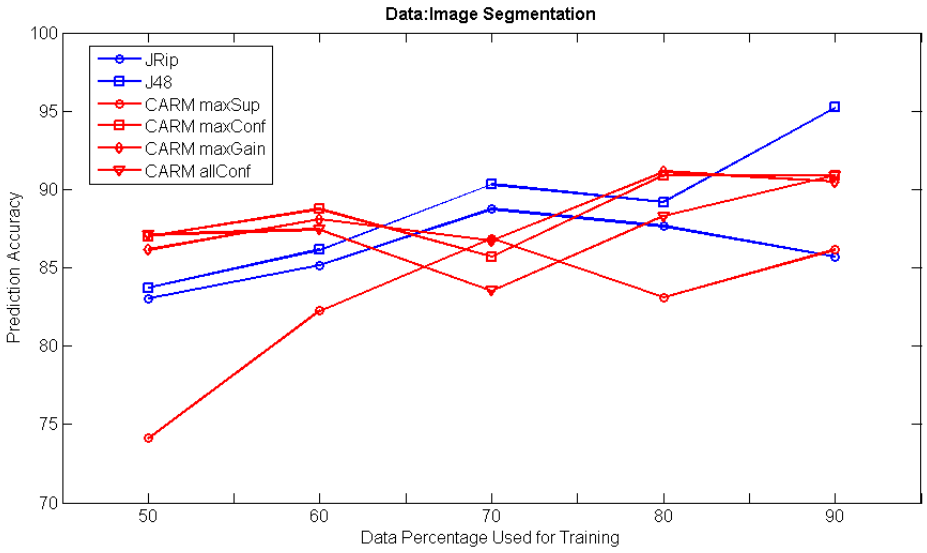


Figure 5.16: Prediction accuracy comparison for the image segmentation dataset

Image segmentation results show similar performance from methods 2, 3 and J48 whereas JRip and method 4 seem to be less accurate. The differences between the solutions, however, are small. One important note in these results is the large improvement of the maximum-support method when the training data percentage increases to 70% or more. This is an indication that rules of high support serve as good classifiers in these cases.

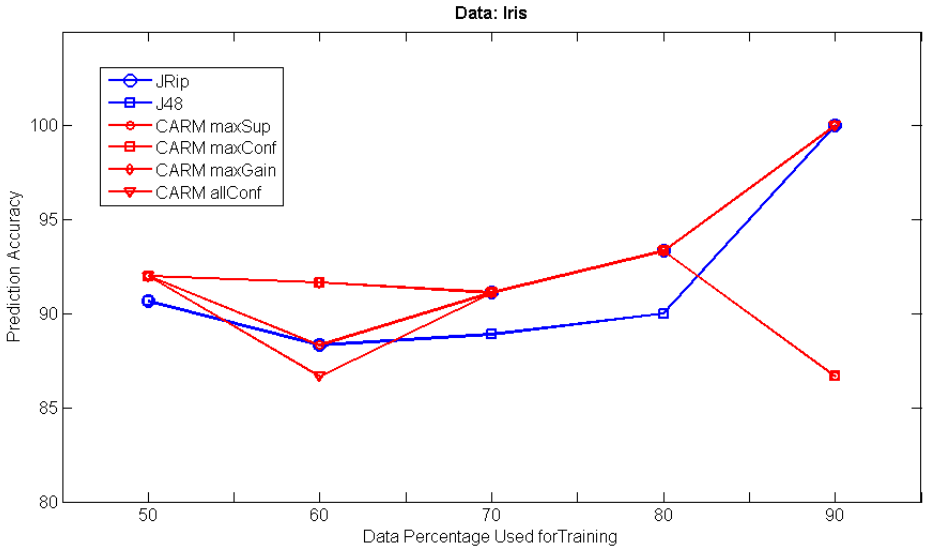


Figure 5.17: Prediction accuracy comparison for the iris dataset

The iris dataset is the most popular dataset in classification research, although it is not as challenging for modern solutions. This is evident in the results where all methods, including method 1, perform very well with accuracy $> 85\%$.

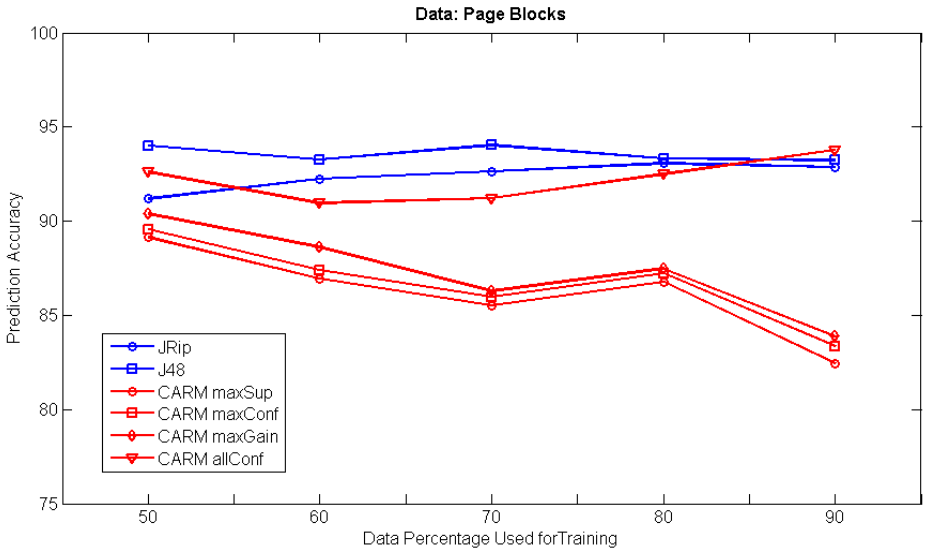


Figure 5.18: Prediction accuracy comparison for the page blocks dataset

The page blocks dataset includes integer as well as real values. Results demonstrate that with the exception of the all-confident method, the developed algorithms perform worse than the competing solutions. As is evident in Figure 5.18 the achieved accuracy is relatively high for all methods, including methods $1-3$.

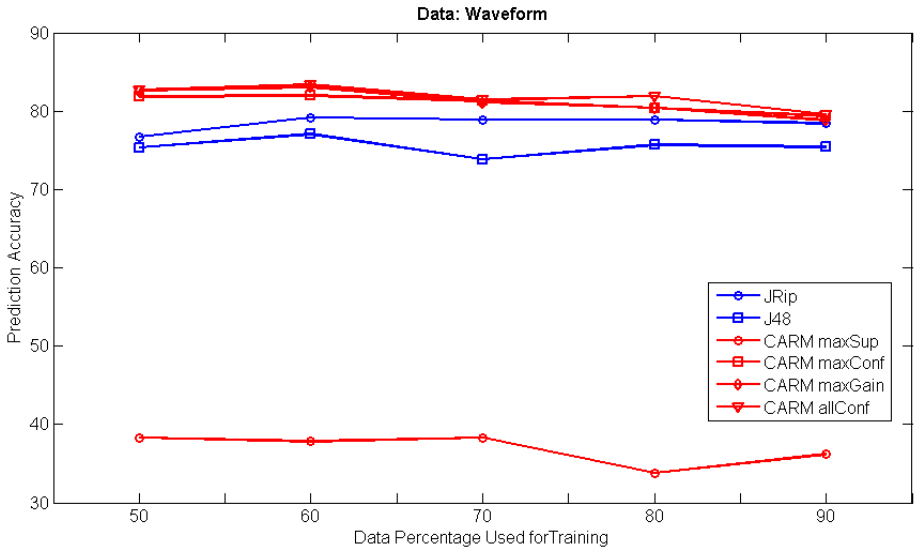


Figure 5.19: Prediction accuracy comparison for the waveform dataset

The waveform dataset is the only dataset consisting of data generated from a data-generator, the code for which is publicly available [104]. The generated data includes noise which seems to affect the prediction accuracy of the developed

algorithms less than for competing solutions. More specifically methods $2-4$ predict more accurately than both JRip and J48 in every experiment, the difference accuracy gradually decreases as the training data percentage increases. The only exception is method 1 that has very low prediction accuracy.

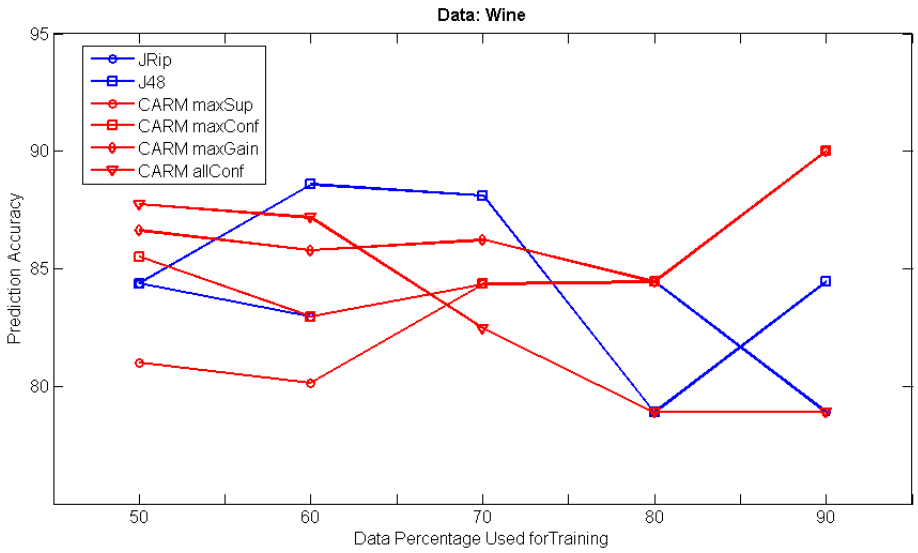


Figure 5.20: Prediction accuracy comparison for the wine dataset

The wine dataset experiment is another case where all methods score high in prediction accuracy overall. The developed method for maximum gain rules, achieves the highest accuracy on average whereas unlike in most cases, the method mining maximum confident rules performs better than the all-confident method.

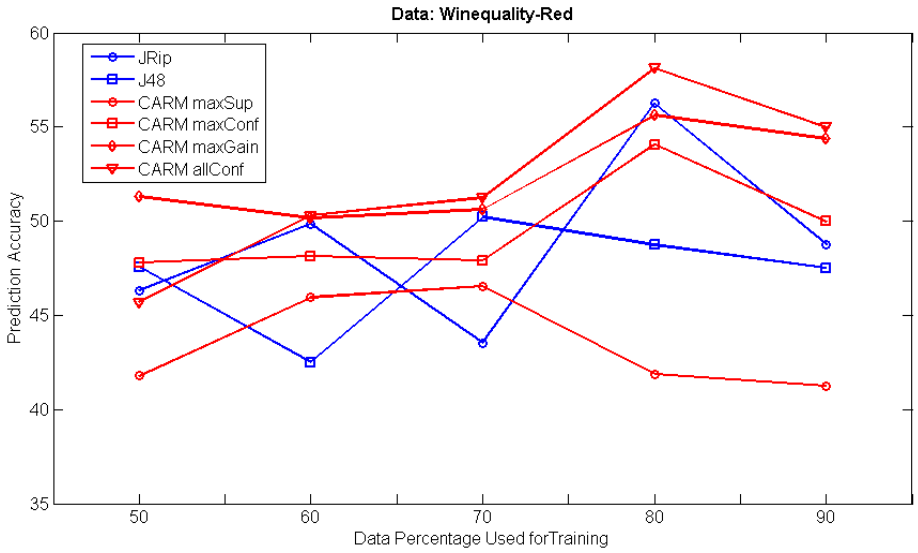


Figure 5.21: Prediction accuracy comparison for the wine quality red dataset

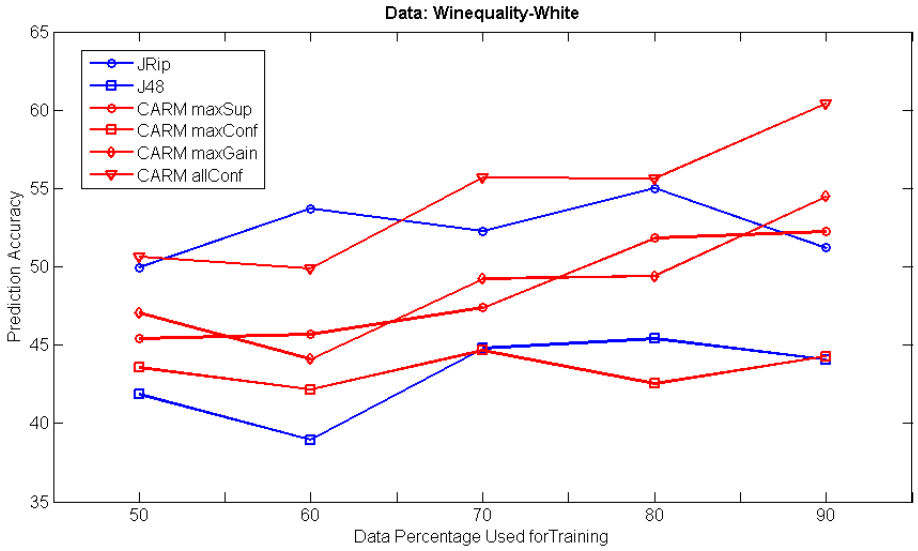


Figure 5.22: Prediction accuracy comparison for the wine quality white dataset

The wine quality datasets consist of data gathered in real world. The attribute values are indexes commonly used as wine quality indicators. The presented experiments include two datasets, one containing red wine data and one containing white wine data. These datasets are of particular interest since they concern a very realistic scenario. All methods achieve accuracy that is significantly less than in the other experiments, indicating the difficulty of separating the different data classes. Furthermore, not all of the predetermined classes are represented in the dataset which is the expected scenario in a data characterization problem.

The J48 algorithm achieves low accuracy in this experiment indicating that the use of rule based solutions is better. The developed algorithms for all methods perform well, indicating that the support based method is useful when mining data of this nature. The all-confident method is the one achieving the best accuracy on average for these data.

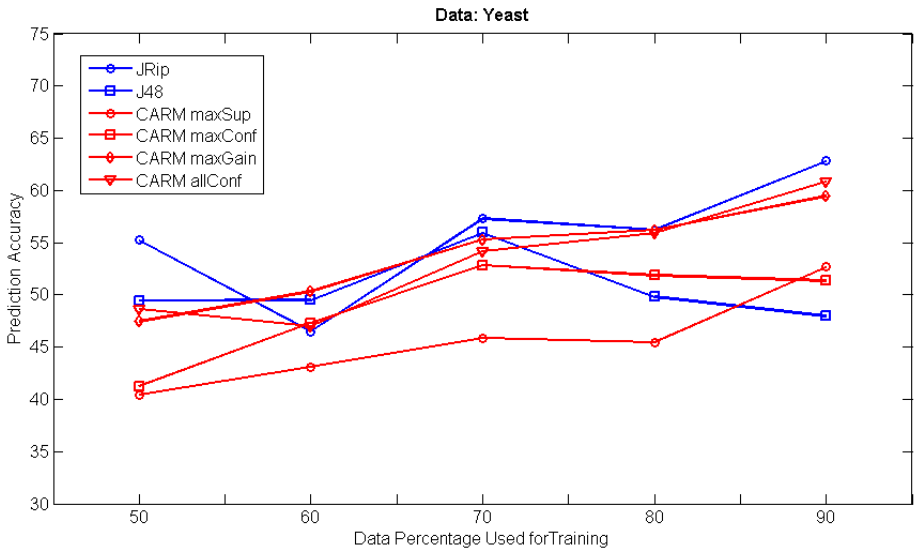


Figure 5.23: Prediction accuracy comparison for the yeast dataset

The yeast dataset consists of real data measurements of protein attributes. The developed algorithms are used for predicting the localisation site of proteins based on the aforementioned measurements. The class label frequencies are very unbalanced with half the class labels only consisting less than 3% of the data each. Methods $3, 4$ outperform J48 but the overall best average accuracy is achieved by JRip. This difference in performance is attributed to the relatively high accuracy achieved by JRip when using only 50% of the data for training. Accuracy for methods $3, 4$ improves consistently as the number of training tuples increases.

Section 5.2.2 presents the results of the aforementioned experiments collectively for a direct comparison of the average results achieved by each algorithm.

Prediction Accuracy Results Summary

Table 5.3 shows a summary of the average prediction accuracy results achieved in the series of experiments presented here.

Dataset	**Algorithm**				
	RIPPER (JRip)	**C4.5 (J48)**	**CARM M2**	**CARM M3**	**CARM M4**
Breast Cancer (Diagnostic)	93.63	94.88	93.55	**96.51**	95.8
Ecoli	79.6	**79.87**	74.36	73.2	75.25
Glass	68.94	65.6	61.09	70.86	**78.85**
Image Segmentation	86.06	**88.92**	88.69	88.52	87.46
Iris	92.69	91.58	90.96	**93.62**	92.62
Page Blocks	92.4	**93.58**	86.7	87.34	92.22
Waveform	78.4	75.47	80.97	81.18	**81.79**
Wine	93	94.88	95.45	**96.62**	93.03
Winequality-Red	48.94	47.3	49.69	**52.68**	52.07
Winequality-White	52.43	43.02	43.45	48.85	**54.44**
Yeast	**55.62**	50.54	48.91	53.75	53.3
Total Average	76.52	75.06	73.98	76.65	**77.89**

Table 5.3: Average Prediction Accuracy (%)

Table 5.3 summarises what was described in Section 5.2.2. The maximum support method is omitted, since, as mentioned in Section 5.2.2, its prediction accuracy was not comparable to the other methods. It is important to note that out of the 4 times that the highest accuracy was not achieved by one of the methods described in this thesis, 3 times the J48 implementation of C4.5 was the solution with the best overall result. Overall, however, J48 performance was lower than JRip which achieved the best accuracy only in one case.

The designed algorithms are created to address the problem of data characterization. The differences between characterization and classification are described in detail in Section 5.3.1. The aforementioned, experiments, however, prove that the classification performance of the new methods is comparable, and marginally superior, compared to that of two of the most popular rule mining solutions in the research community. In order to analyse the statistical significance of the results presented in Table 5.3 *one way ANOVA (analysis of variable)* is employed [55]. The high p-value shown in Table 5.4 verifies that CARM results are comparable to both C4.5 and RIPPER. Figure 5.24 provides a *box plot* of the results of each method shown in Table 5.3.

Source	Sum of squares	Degrees of freedom	Mean squares	F-statistic	p-value
Columns	101.5	4	25.372	0.07	0.9898
Error	17180.7	50	343.614		
Total	17282.2	54			

Table 5.4: ANOVA table for prediction summary results.

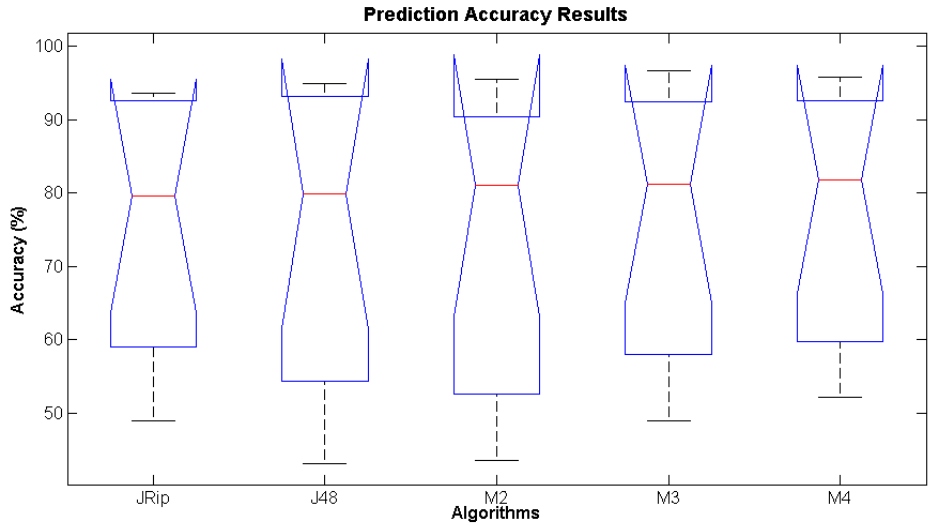


Figure 5.24: Box plot for prediction summary results.

5.3 Characterization

Section 5.2.2 presented a series of experiments that demonstrate the competitiveness of the designed solution in a classification context. The purpose of this thesis, however, is to describe a solution that generates rules that are useful for characterizing continuous data. This section presents specific aspects of range-based classification rule mining that address the purpose of data characterization better than the existing solutions.

5.3.1 Differences Between Characterization And Classification

The problem of mining characteristic rules from data is closely related to that of classification. There are, however, distinctive differences. When mining classification rules the importance lies with having rules that given a set of unlabeled data tuples can accurately classify them in a pre-defined set of classes. From a data characterization perspective, however, it is important to identify rules that represent areas in the data of potential interest and can be used for generating hypothesis that can be evaluated on the given problem space.

In classification the result is a model capable of classifying unlabeled data. Even when this model consists of a set of association rules, like in the case of RIPPER, these rules should not be interpreted or evaluated individually. Evaluating rules individually is possible after post-processing but in this case there are no guarantees that the processed rules will be of interest.

Generally speaking classification focuses on mining knowledge that will be able to accurately classify new, unseen data whereas characterization focuses on best describing the given data and their multiple characteristics. Because of the above in the case where the unlabeled dataset have the same underlying characteristics as the mined data a good characterization solution will have good classification results, but accurately predicting the unlabeled data does not mean that all the characteristic and possibly useful rules are included in the mined ruleset. This is why the features used for evaluating the resulting rules in Section 5.3.2 do not refer to unlabeled data only but properties of the mined rules themselves with regards to the mined dataset.

5.3.2 Characterization Evaluation

This section presents a comparison between range-based classification rule mining and competing solutions with regards to specific features of the developed solution that serve the purpose of data characterization and are often overlooked by methods designed specifically for classification.

The Benefit Of Ranges And Overlapping Rules

All methods applied on continuous data that do not require pre-discretization, perform a type of split of the data, thus generating numerical ranges. In existing solutions, this is a binary split, in a form of a relation of *greater-or-equal* ($\geq$) or *less-or-equal* ($\leq$). The developed method detects a numerical range, with an upper and lower limit, that effectively splits the attribute values in three ranges.

A more important differentiation is rule overlapping. The developed method mines overlapping rules which allows for alternative splits of the same problem space. Both RIPPER and C4.5 attempt to split the space in disjunct areas of high confidence, what differs is the strategy. In every step, however, the data space is split and any tuple that is included in a resulting rule is excluded from the data points that are used for further mining. Figures 5.25, 5.26 represent the two different methods when mining a dataset of two different class labels. More specifically, Figure 5.25 demonstrates the split of a given data space using non overlapping rules. Changing the rules requires mining the data again using different thresholds.

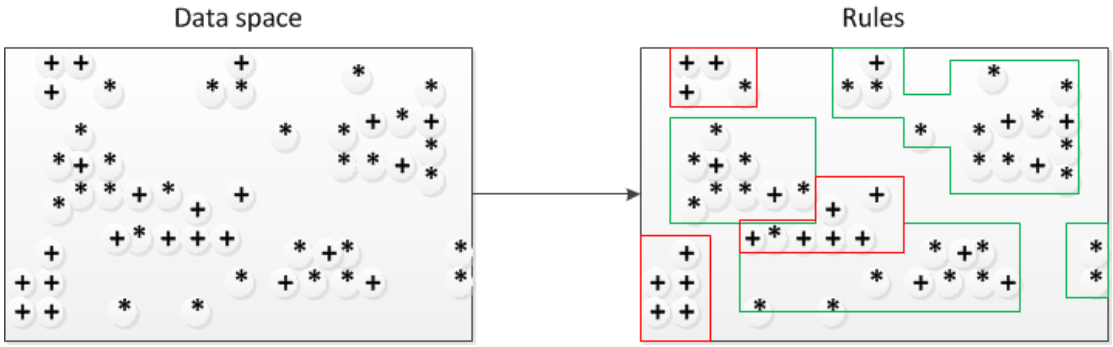


Figure 5.25: A model consisting of non overlapping rules

Mining overlapping rules is a greater challenge but allows the same tuple to be mined as more than one different rules and by extension the complete data set to be represented by more than one rule sets. In Figure 5.26 the different representations consist of a single rule-set resulting from the described method. Note that not only rules for the same class may overlap.

In the context of data characterization multiple representations of the given data space allow for more options when evaluating the knowledge represented by the rules.

Furthermore, the voting methodology during prediction gives a more complete

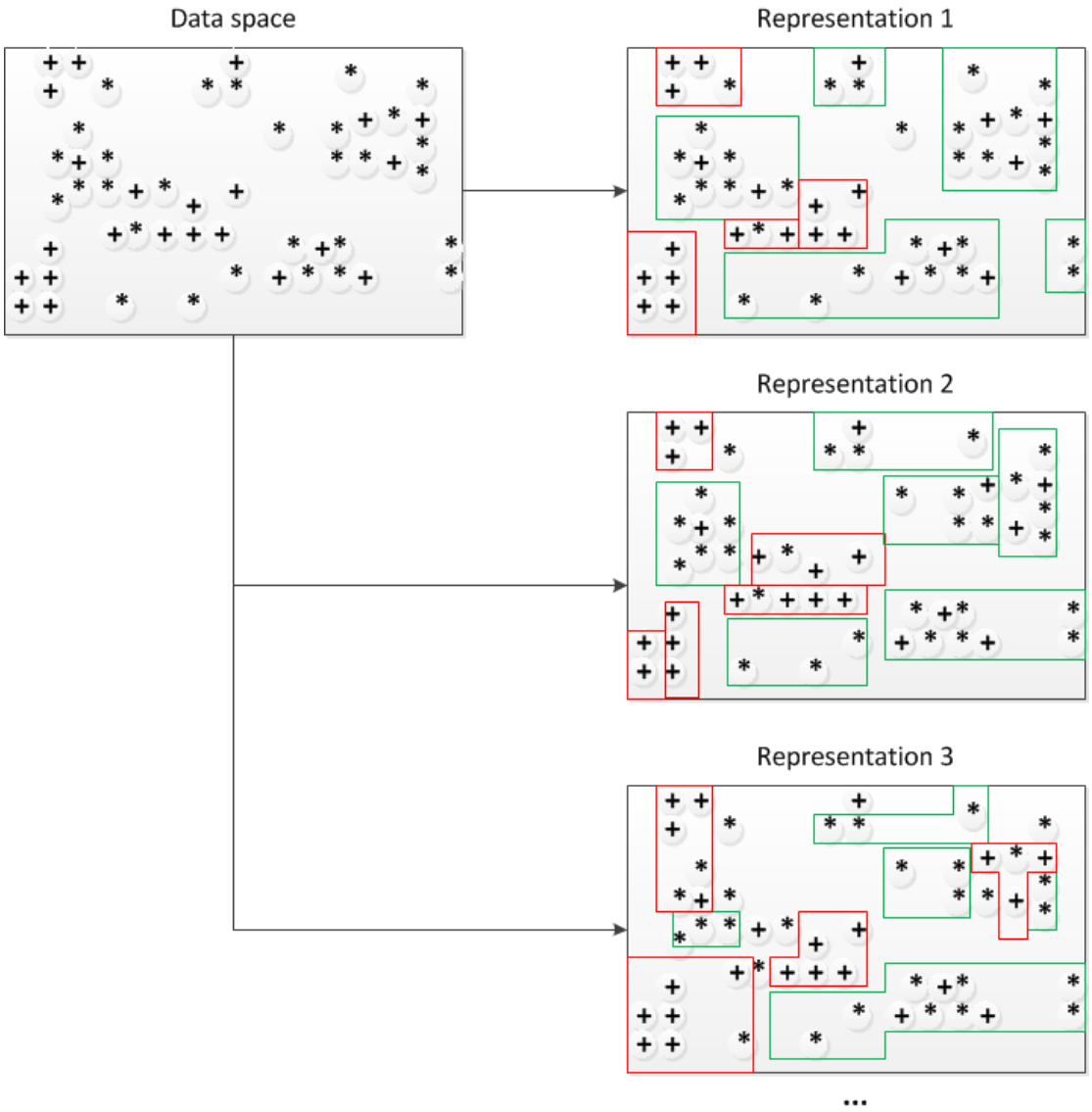


Figure 5.26: A model consisting of several representations using overlapping rules

image for the unlabeled data when the number of different classes is higher. For example, consider an unlabeled tuple where the voting table is Table 5.5.

$class_1$	$class_2$	$class_3$	$class_4$	$class_5$
4	3	0	1	0

Table 5.5: An example of a voting table

When predicting using non-overlapping rules the unlabeled tuple may only be covered by a single rule in the model which may be a default rule in the case of RIPPER (a rule for any tuple not covered by the mined rules). However, there is

no information about the relation of $class_1$ and $class_2$. If the two classes represent conditions that a domain expert considers similar then this is actually a highly confident rule, otherwise, if the two classes represent very different conditions it is not. In any case the voting table provides important information for interpreting the rule.

Result Consistency

One feature of data characterization is consistency. Classification relies heavily on the data used for training the classifier(s) and adding to the training data changes the results significantly. However, from a characterization perspective, adding 10% of tuples to the training data should not differentiate the resulting rules significantly since the data characteristics have not changed significantly.

In the graphics below there is a comparison of the absolute difference in prediction accuracy when changing the percentage of data used for training, between JRip, J48 and the max-gain and all-confident methods of the described algorithm.

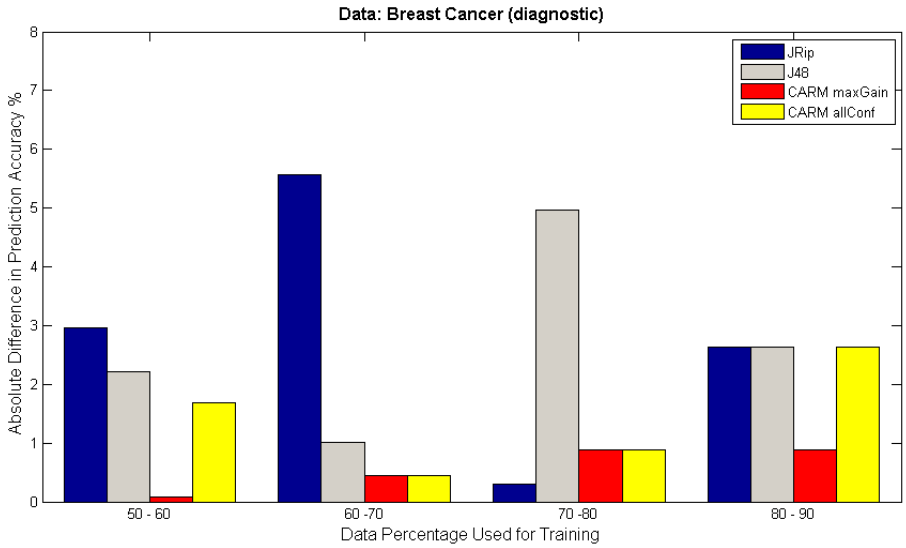


Figure 5.27: Prediction accuracy consistency for breast cancer data.

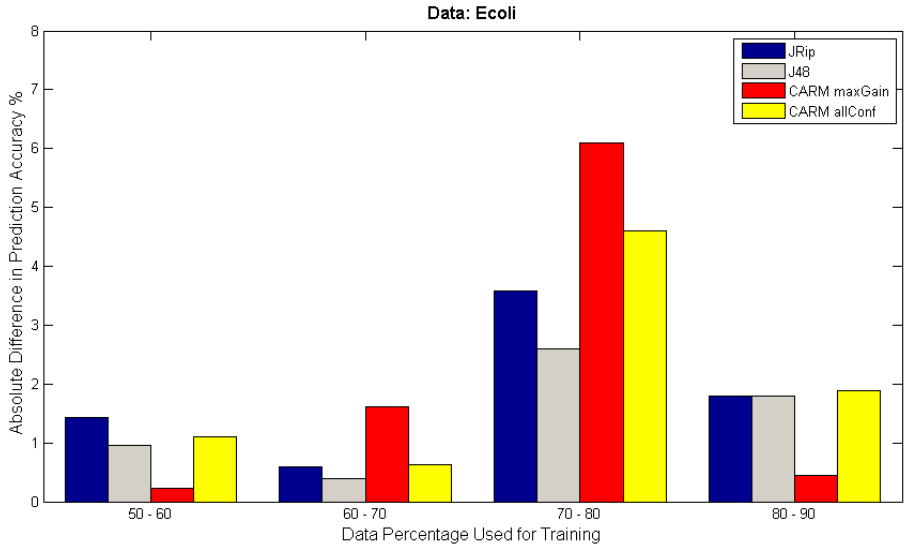


Figure 5.28: Prediction accuracy consistency for ecoli data.

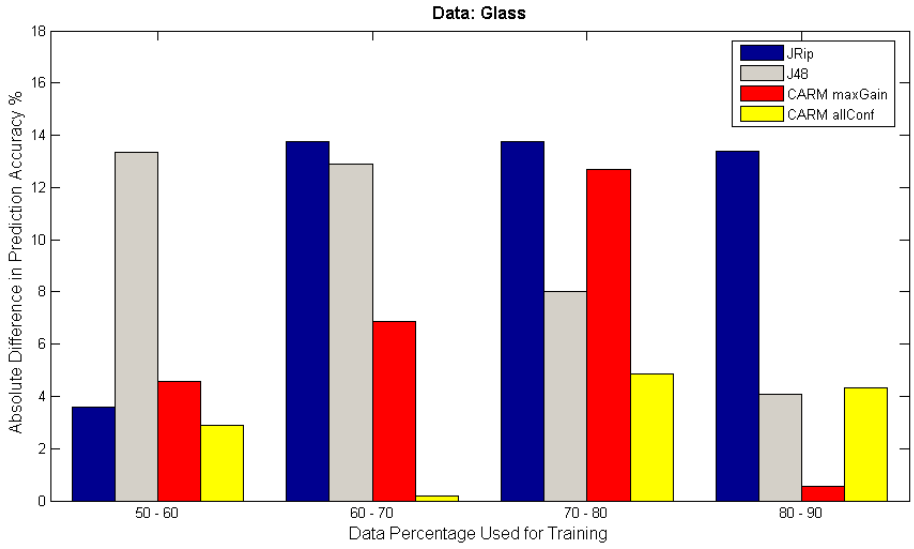


Figure 5.29: Prediction accuracy consistency for glass data.

Three datasets were selected for these experiments. The *Ecoli* data represent the most characteristic case where prediction using range-based rules is less accurate than the competition, whereas the *Glass* dataset favors it. Finally, the *breast cancer* dataset is used as a representative case of the average scenario, where the presented method outperforms both competitive solutions.

The figures demonstrate that the all-confident method performs more consistently than both JRip and J48 in most cases and with significant difference. The max-

imum gain method performs well but not as well as the all-confident approach. These results are an important point for the robustness of CARM as a classification method, which is a positive feature when characterizing data.

Top k Rule Cohesion

Data characterization is an important, realistic problem with applications in many data mining scenarios. Unlike classification, however, measuring the characterization effectiveness of a data mining methodology, and more importantly comparing the characterization effectiveness of different solutions is challenging since there are no established methods of doing so. Result consistency, as presented in Section 5.3.2 is an aspect of the presented algorithm that reinforces its value as a data characterization solution. In this section another set of experiments is presented, measuring the median density of the best k rules.

The motivation behind these experiments is to determine the characterization value of the generated rules by measuring how dense are the *best* rules generated from the data. Selecting the *best* rules is closely related to the characterization problem, when trying to identify the characteristic knowledge derived from a mined model a user is most likely to select a subset of k rules. For example, consider a racing team that is using practice data to determine the best settings for a vehicle. The team's experts will be most interested in the knowledge derived by those laps when performance is best rather than any other case. The criteria that determine exactly which rules are *best* vary for different scenarios.

Selecting the set S of the k best rules, however, is not enough since there still remains the challenge of determining how independent these rules are from the remaining data, the tuples not covered by the rules in S. A rule is independent when there is minimal overlap between the associated ranges and the data not covered by it, therefore when its density is high. The following experiments use this hypothesis to determine the characterization strength by comparing the median value of density, δ of the k best rules for each dataset, where $k = 10$. In the results presented in Table 5.6 the rules are selected using confidence whereas in Table 5.8 the rules are selected using support. In both cases the complete datasets were used to mine the rules.

Dataset	Algorithm			
	RIPPER (JRip)	**C4.5 (J48)**	**CARM M3**	**CARM M4**
Breast Cancer (Diagnostic)	0.398	0.015	**0.41**	**0.41**
Ecoli	0.126	0.013	0.058	**0.229**
Glass	0.089	0.026	**0.122**	**0.122**
Image Segmentation	0.147	0.021	**0.313**	**0.313**
Iris	**1.0**	0.32	0.625	0.625
Page Blocks	0.035	0.001	0.03	**0.076**
Waveform	**0.075**	0.011	0.019	0.035
Wine	**1.0**	0.325	0.337	0.337
Winequality-Red	0.018	0.009	0.02	**0.03**
Winequality-White	0.002	0.004	**0.016**	0.012
Yeast	**0.110**	0.007	0.026	0.015

Table 5.6: Median density cohesion for the 10 most confident rules.

Table 5.6 shows a summary of the results when the 10 most confident rules are selected for comparison. One point clearly shown in above table is that *C4.5* is clearly outperformed by the other solutions in every single case examined. The *C4.5* algorithm mines rules by performing binary splits on numerical ranges, in every experiment performed it results in more rules than *RIPPER* that achieve higher confidence but the density of the resulting rules is reduced. *RIPPER* is a special case, as seen in Table 5.6, it has the highest median density in 4 cases but in the case of the iris and wine datasets this is because of the very small number of rules actually generated by this method (3 rules for each dataset). This can be easily identified in the results by comparing Tables 5.6 and 5.8 where the median values for *RIPPER* are the same for 6 out of the 11 datasets since the selected rules are exactly the same.

The methods presented in this thesis, however, outperform overall, both competing solutions mining a high number of rules in every case and achieving comparatively high density for the selected rules. More specifically method 4 performs better than method 3 since the split methodology results in a set of all the rules meeting the given thresholds instead of only the rule of highest gain.

Table 5.7 and Figure 5.30 show ANOVA results and the corresponding box plot

for each algorithm. It is important to note that in the box plot the best results for RIPPER are marked as outliers which is consistent with the experiments that show the reason for the high values is the actual inability of the algorithm to mine a large enough number of rules in these cases.

Source	Sum of squares	Degrees of freedom	Mean squares	F-statistic	p-value
Columns	0.23623	3	0.07874	1.31	0.2844
Error	2.4037	40	0.06009		
Total	2.63993	43			

Table 5.7: ANOVA table for density cohesion of 10 most confident rules results.

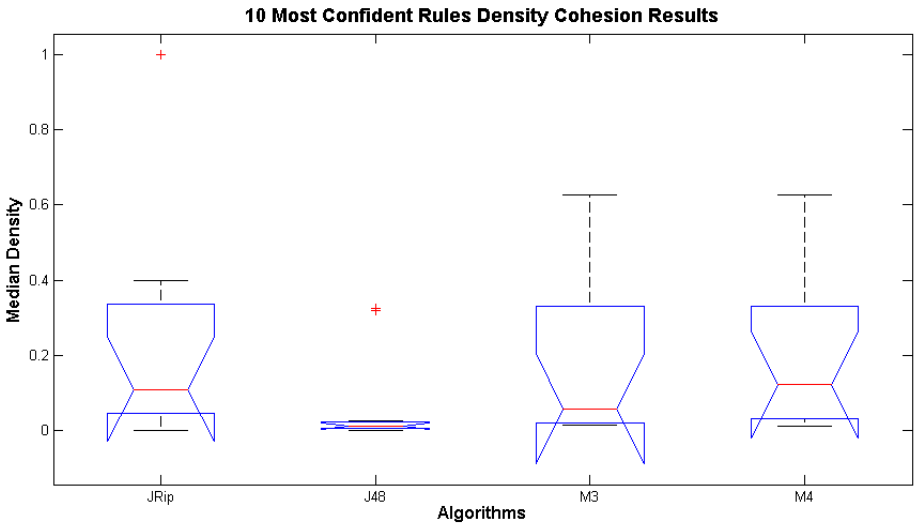


Figure 5.30: Box plot for density cohesion of 10 most confident rules results.

Table 5.8 shows a summary of the same results when the top 10 rules are selected using support as the criterion for the selection. The comparative results for *C4.5* do not change, the tree-based mining algorithm achieves low density results compared to the other methods. *RIPPER* results, excluding the iris and wine datasets, are higher than *C4.5* but not as high as the results of the presented methods. An important remark is that for all methods selecting the rules with best support results in better density. This is an indication that in most cases, in the datasets used in this chapter, the algorithms that are trying to improve rule confidence suffer a decrease in rule density.

Dataset	Algorithm			
	RIPPER (JRip)	**C4.5 (J48)**	**CARM M3**	**CARM M4**
Breast Cancer (Diagnostic)	0.398	0.019	0.596	**0.645**
Ecoli	0.126	0.042	0.515	**0.518**
Glass	0.089	0.056	0.227	**0.237**
Image Segmentation	**0.238**	0.094	0.161	0.161
Iris	**1.0**	0.32	0.625	0.749
Page Blocks	0.035	0.021	**0.999**	**0.999**
Waveform	0.163	0.025	0.191	**0.243**
Wine	**1.0**	0.325	0.376	0.404
Winequality-Red	0.064	0.024	0.174	**0.313**
Winequality-White	0.056	0.009	0.482	**0.684**
Yeast	0.132	0.038	0.16	**0.17**

Table 5.8: Median density cohesion for the 10 most supported rules.

ANOVA results shown in Table 5.9 and Figure 5.31 demonstrate how the results for C4.5 and RIPPER do not differ significantly from the selection of the 10 most confident rules. These results indicate that unlike CARM these solutions are not as flexible at focusing on a different measure and are primarily designed to target highly confident rules which is the primary concern for classification solutions. The two perfect density scores of *RIPPER* are again marked as outliers.

Source	**Sum of squares**	**Degrees of freedom**	**Mean squares**	**F-statistic**	**p-value**
Columns	0.91538	3	0.30513	4.2	0.0112
Error	2.90318	40	0.07258		
Total	3.81856	43			

Table 5.9: ANOVA table for density cohesion of 10 most supported rules results.

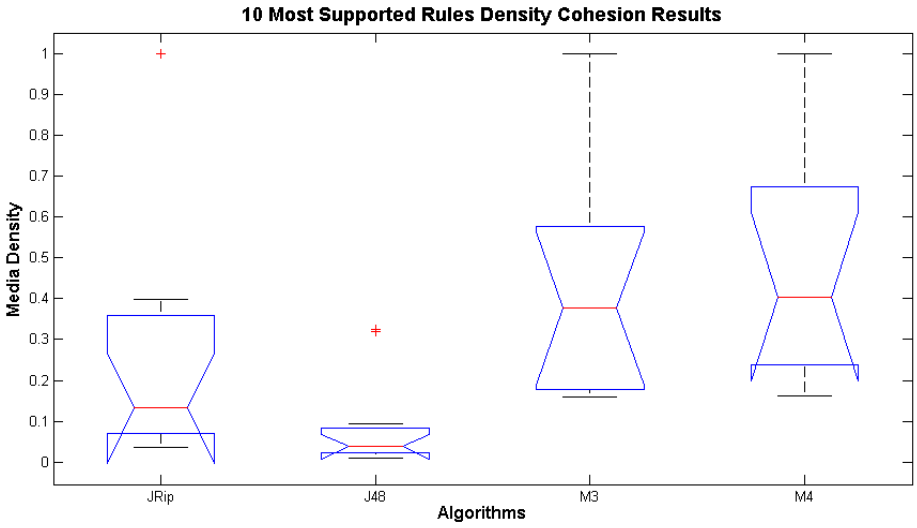


Figure 5.31: Box plot for density cohesion of 10 most supported rules results.

5.4 Summary

The extensive experiments presented in this chapter evaluate the performance of the designed algorithm when applied to either a classification or data characterization task, using a collection of popular established datasets of variable properties covering a large number of different cases.

In terms of classification, the presented algorithm achieved results comparable to those of *RIPPER* and *C4.5* and methods 3 and 4 proved better, by a small margin, in terms of prediction in the series of experiments.

Furthermore, CARM was shown to be able to address challenges related to data characterization. The evaluation in terms of results consistency and top-k rule coverage show how the properties of mining overlapping rules and generating multiple alternatives to covering the given data set make CARM a better solution for data characterization.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

Due to the increasing number of data collections that are continuously being developed across different areas the need for data mining solutions that can translate these datasets into useful domain knowledge has become increasingly relevant. This thesis has examined the problem of a dataset containing real values as attributes where the mining process needs to maintain the original properties of the recorded values as real numbers. The desired result is a set of rules that can be evaluated by domain experts and used for generating hypotheses on the given data that can lead to previously hidden domain knowledge. The contributions of the presented work can be summarised as follows:

- **Range-based rules:** This thesis proposed a novel solution that mines continuous ranges, including ranges that can overlap, that meet a set of given thresholds for confidence, support and density. The resulting rules, therefore, meet the pre-determined requirements and optimise a heuristic-based criterion.

  The fact that no discretization is necessary on the data makes the mining process simpler and also applicable in a wider range of datasets. Discretizing the continuous attributes essentially changes the nature of the data from continuous to categorical, this is evident in the fact that the efficiency of solutions that employ discretization is highly dependable on the discretization itself. Therefore, as explained in Section 2.2, discretizing the data does not resolve the problem of mining continuous attributes but transforms the

problem into finding a proper discretization method. Furthermore, a good discretization relies on the underlying data, therefore if the mined dataset changes it is very likely that a new discretization will have to be found before the mining application.

This is an important factor since many real world datasets contain real values (e.g. mining of historical data collections of industrial production sites). Using the designed solution alleviates the need for discretizing the data and allows for direct mining of the data allowing for the mining of overlapping ranges. Overlapping ranges allow for a specific data value to be included in more than one of the resulting rules which is an important feature in the exploratory analysis attempted in this thesis.

- **Density measure:** The solution described uses the traditional confidence and support paradigm for evaluating rules. However, one of the key aspects of this work is the nature of continuous data and the aforementioned measures were originally designed for mining solutions focusing on categorical data. In order to evaluate how dense is a range-based rule, that is how many of the data values are covered by the individual ranges but not by the rule, the described solution proposes a new measure of *density*. Employing the new measure during the mining process as a user set threshold allows for the mining of dense rules.

  The new measure can also be used in existing solutions by modifying the mining algorithm to include it as a threshold or simply to evaluate the resulting rules which is useful in the context of data characterization. When evaluating rules for generating hypotheses dense rules will be preferable since the described data are more independent of the remaining dataset.

  A series of experiments in Section 5.2.1 also demonstrates how different values for the density threshold affect the prediction results of the mined rules.

- **Heuristic based approaches:** Chapter 4 describes four different heuristic based algorithms for generating range-based classification rules. Each method is based on optimising a different criterion therefore allowing for a different approach to generating the resulting set of rules. The different criteria applied result in a different split of the consequent bound rules. More importantly, two methods, one based on splitting for the best gain rule and the other based on splitting for best confidence and selecting all the rules

that meet the user-defined thresholds, are shown to perform very well on traditional prediction tasks as well as data characterization.

Specifically, an evaluation over a large number of publicly available datasets shows the new methods outperforming overall the two established solutions of *Ripper* and *C4.5* in prediction in spite of not having been developed for traditional classification tasks. The new methods are shown to handle well the used datasets and performing particularly well in difficult data sets where it is hard to identify good ranges in the data. Overall the designed methods have been shown to be flexible due to the users ability of defining minimum thresholds for the mined rules, effective in prediction tasks as well as in cases where the goal is to find important data areas that can be used to discover new data knowledge.

- **Characterization:** In the described scenario the requested solution needs to go further than traditional data analysis. This thesis describes the defining properties of the data characterization problem and how it differs from traditional analysis methods. A detailed explanation is provided for cases when it is important to discover specific rules of characterization importance rather than rules that simply achieve higher classification accuracy.

  The limits of existing solutions have been previously identified in existing work [94, 96, 111, 112] but the concept of mining classification rules for the purpose of data characterization has not been addressed in the existing literature. More importantly this thesis presents specific aspects of data characterization that a rule mining solution needs to address in order to be considered an efficient data characterization solution.

  The max-gain and all-confident methods presented in this thesis are evaluated against the aforementioned criteria and shown to address the newly defined aspects that are not covered by existing solutions.

6.2 Applications

There is a range of applications for CARM. Examples provided in this thesis, represent areas where CARM can help extract important knowledge from the data. A characteristic example, and one that served as an inspiration for the development of CARM, is the analysis of a dataset recorded in a production facility where the

data consists of real values that record at regular intervals the conditions of the production process. The target classes are the classification of the process output which can be in terms of efficiency (Good, Average, etc.) profitability (Very profitable, Marginally profitable, etc.), it is also possible to repeat the analysis using different classes as the target. Consider the example of a chemical process that follows a production method. Historically this process will have been run within certain parameters as long as profitability, safety and legislation compliance are achieved. A classification model assumes that everything is known about the process and may successfully predict the outcome within the same parameters but CARM can help domain experts use the mined rules to discover possible areas of improvement and under certain conditions to better understand the underlying mechanisms.

This applies to other processes that have not been fully modelled like biochemical reactions, financial data like stock values where the monitored data are numerical and there is often a requirement of understanding (characterizing) what is in the data. CARM would be ideal for these applications.

Any of the four methods described in Chapter 4 could be applied although experimental results suggest focusing on the maximum gain and the all-confident method to mine a set of candidate rules. The resulting rules can then be ranked and/or evaluated so as to select data ranges to focus on for further exploration (hypothesis generation). This point may be considered a weakness of CARM compared to classification solutions that are more "automated" and do not require domain knowledge to be applied on the resulting model, however large real-world production processes are not fully mapped and the aforementioned models rely on assumptions that may contradict domain knowledge.

6.3 Future work

This section examines possible ways of extending the work presented in this thesis. First, Section 6.3.1 presents ways for improving the described solution by exploring options that go beyond this thesis whereas Section 6.3.2 examines possibilities of applying the presented work in different areas.

6.3.1 Modifications

The problem of generating consequent bounded rules is covered in Chapter 4 and the heuristics presented in Chapter 3 explore the options of splitting for generating the rules quite conclusively. However, there is room for exploration in creating a parallel executed algorithm, including categorical attributes in the mined data and in the area of evaluating the resulting rule particularly in the context of data characterization.

- **Parallel execution:** The most costly process in the presented algorithm is the splitting of each consequent bounded rule in the *LR structure*. However, each entry in the structure is independent of the others and the splitting process can be mapped to run on a separate processor. Therefore, it is possible to modify the existing implementation to use a modern data processing tool like [25] for executing the split processes in parallel.

  Developing such a solution will improve the scalability of the existing algorithm, especially in the case of datasets with a very high number of attributes where the number of possible associations increases exponentially. Provided that the resulting solution is efficient enough it is also possible to automate the process of identifying the optimal values for the given thresholds. More specifically when a specific evaluation process is defined, like in the exploratory analysis in Section 5.2.1, it is possible to automatically define the optimal thresholds for $\sigma_{min}, \gamma_{min}, \delta_{min}$. This could prove useful when developing a software tool that performs a complete mining process using the solution in this thesis.

- **Categorical data inclusion:** CARM has originally been designed to mine data consisting of real values only. The mining of ranges does not apply in categorical data but CARM can be modified to handle categorical attributes as well. A naive approach would be to split the dataset in two subsets one containing the categorical attributes, where a standard a-priori like solution can be applied, and one containing the continuous ones, where CARM is applied, and attempting to intersect the resulting sets. A more efficient solution, however, is to mine the frequent itemsets for each class label and then apply CARM on the tuples supporting each itemset to mine the associated numerical ranges. Note that even though it is possible to map categorical attributes to integers and just apply CARM that would be incorrect because sorting integer values in the first phase of the algorithm is meaningful for in-

tegers but not for categorical values (we should instead consider all possible combinations/sortings).

- **Characterization measures:** The lack of a clear definition of a data characterization task is evident in the existing research literature. One of the most challenging aspects of evaluating the described method was to define characterization in ways that could be used to measure the effectiveness of the resulting rules. Although a challenging task, it is possible to develop measures like the ones developed for classification that would make the characterization evaluation more straightforward. The development of such measures can be based on real world data analysis of different domains.

  Developing such measures would allow the creation of new heuristics based on them or simply be used for the evaluation of the present and other existing solutions. Alternatively, ranking mechanisms can be employed for ordering of the resulting rules in terms of domain knowledge importance.

6.3.2 New Research Directions

Other aspects of future research work can be based on the possibility of applying the work of this thesis in different areas and gathering feedback from real-world applications.

- **Data streams:** An interesting application of the presented work would be in the ever growing area of dynamic data streams (e.g. sensor data). By removing the assumption that the original dataset is a complete data collection that is available prior to data mining the nature of the research problem changes drastically. Since the data is not is not readily available a solution is to mine the existing data for the rules and then attempt to adapt the mined rules to include data in the continuously recorded data.

  This is a very challenging task but of high value in the context of mining data from data streams. For example, in an industrial production facility that records its process constantly mining a historical data collection the opportunity would be created to adapt existing process knowledge, in the form of previously mined rules, by incorporating newly created data.

- **Readability, interpretability, causality evaluation:** As described in Section 2.7 the format of the resulting rules and the mining methodology in

CARM are designed to address these issues. They are, however, not quantifiable which creates opportunities for additional research. It is possible to design a questionnaire for domain experts to compare the rules mined by CARM with those mined by competing solutions in terms of the aforementioned qualities. Such a questionnaire could also allow experts to suggest areas of improvement that can be investigated in future work and incorporated into CARM.

Bibliography

[1] D.J. Newman A. Asuncion. UCI machine learning repository, 2007.

[2] C. C. Aggarwal and P. S. Yu. Mining associations with the collective strength approach. *IEEE Trans. on Knowl. and Data Eng.*, 13(6):863–873, November 2001.

[3] Charu C. Aggarwal and Philip S. Yu. A new framework for itemset generation. In *Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, PODS '98, pages 18–24, New York, NY, USA, 1998. ACM.

[4] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[5] Jesús S. Aguilar-Ruiz, Jaume Bacardit, and Federico Divina. Experimental evaluation of discretization schemes for rule induction. In *GECCO (1)*, pages 828–839, 2004.

[6] Jaume Bacardit and Josep Maria Garrell i Guiu. Evolving multiple discretizations with adaptive intervals for a pittsburgh rule-based learning classifier system. In *GECCO*, pages 1818–1831, 2003.

[7] Jaume Bacardit and Josep Maria Garrell i Guiu. Analysis and improvements of the adaptive discretization intervals knowledge representation. In *GECCO (2)*, pages 726–738, 2004.

[8] Roberto J. Bayardo, Jr. and Rakesh Agrawal. Mining the most interesting rules. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '99, pages 145–154, New York, NY, USA, 1999. ACM.

[9] Jon Louis Bentley. Algorithm design techniques. *Commun. ACM*, 27(9):865–871, 1984.

[10] R. Booth. The beginnings of the EVN, JIVE and early space VLBI in Europe. In *Resolving The Sky - Radio Interferometry: Past, Present and Future*, 2012.

[11] Marc Boulle. Khiops: A statistical discretization method of continuous attributes. *Machine Learning*, 55:53–69, 2004.

[12] Leo Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

[13] Sergey Brin, Rajeev Motwani, Jeffrey D. Ullman, and Shalom Tsur. Dynamic itemset counting and implication rules for market basket data. In *Proceedings of the 1997 ACM SIGMOD international conference on Management of data*, SIGMOD '97, pages 255–264, New York, NY, USA, 1997. ACM.

[14] Sergey Brin, Rajeev Rastogi, and Kyuseok Shim. Mining optimized gain rules for numeric attributes. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '99, pages 135–144, New York, NY, USA, 1999. ACM.

[15] Ben Burdsall and Christophe Giraud-Carrier. Evolving fuzzy prototypes for efficient data clustering. In *IN PROC. OF THE 2ND INTERNATIONAL ICSC SYMPOSIUM ON FUZZY LOGIC AND APPLICATIONS*, pages 217–223. ICSC Academic Press, 1997.

[16] J. Catlett. On changing continuous attributes into ordered discrete attributes. In *Proceedings of the European working session on learning on Machine learning*, EWSL-91, pages 164–178, New York, NY, USA, 1991. Springer-Verlag New York, Inc.

[17] Jess Cerquides and Ramon Lpez de Mntaras. Proposal and empirical comparison of a parallelizable distance-based discretization method, 1997.

[18] Chien-Chung Chan, C. Batur, and Arvind Srinivasan. Determination of quantization intervals in rule based model for dynamic systems. In *Systems, Man, and Cybernetics, 1991. 'Decision Aiding for Complex Systems, Conference Proceedings., 1991 IEEE International Conference on*, pages 1719–1723 vol.3, Oct.

[19] Ming-Syan Chen, Jiawei Han, and Philip S. Yu. Data mining: An overview from a database perspective. *IEEE Trans. on Knowl. and Data Eng.*, 8(6):866–883, December 1996.

[20] J.Y. Ching, Andrew K C Wong, and K. C C Chan. Class-dependent discretization for inductive learning from continuous and mixed-mode data. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 17(7(1995)):641–651, Jul.

[21] Peter Clark and Robin Boswell. Rule induction with cn2: Some recent improvements. In *Proceedings of the European Working Session on Machine Learning*, EWSL '91, pages 151–163, London, UK, UK, 1991. Springer-Verlag.

[22] William W. Cohen. Fast effective rule induction. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123. Morgan Kaufmann, 1995.

[23] Thomas H. Cormen, Clifford Stein, Ronald L. Rivest, and Charles E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition, 2001.

[24] Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decis. Support Syst.*, 47(4):547–553, November 2009.

[25] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: A flexible data processing tool. *Commun. ACM*, 53(1):72–77, January 2010.

[26] David DeWitt and Jim Gray. Parallel database systems: The future of high performance database systems. *Commun. ACM*, 35(6):85–98, June 1992.

[27] Federico Divina, Maarten Keijzer, and Elena Marchiori. A method for handling numerical attributes in ga-based inductive concept learners. In *GECCO*, pages 898–908, 2003.

[28] Guozhu Dong and Jinyan Li. Interestingness of discovered association rules in terms of neighborhood-based unexpectedness. In *Proceedings of the Second Pacific-Asia Conference on Research and Development in Knowledge Discovery and Data Mining*, PAKDD '98, pages 72–86, London, UK, UK, 1998. Springer-Verlag.

[29] James Dougherty, Ron Kohavi, and Mehran Sahami. Supervised and un-

supervised discretization of continuous features. In *MACHINE LEARNING: PROCEEDINGS OF THE TWELFTH INTERNATIONAL CONFERENCE*, pages 194–202. Morgan Kaufmann, 1995.

[30] Xiaoyong Du, Zhibin Liu, and Naohiro Ishii. Mining association rules on related numeric attributes. In *Proceedings of the Third Pacific-Asia Conference on Methodologies for Knowledge Discovery and Data Mining*, PAKDD '99, pages 44–53, London, UK, UK, 1999. Springer-Verlag.

[31] Tapio Elomaa and Juho Rousu. Fast minimum training error discretization. In *ICML*, pages 131–138, 2002.

[32] Tapio Elomaa and Juho Rousu. Efficient multisplitting revisited: Optima-preserving elimination of partition candidates. *Data Min. Knowl. Discov.*, 8(2):97–126, March 2004.

[33] I. W. Evett and E. J. Spiehler. Knowledge based systems. chapter Rule induction in forensic science, pages 152–160. Halsted Press, New York, NY, USA, 1988.

[34] Andrea Falcon. Aristotle on causality. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Spring 2014 edition, 2014.

[35] Fayyad and Irani. Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. pages 1022–1027, 1993.

[36] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. Advances in knowledge discovery and data mining. chapter From Data Mining to Knowledge Discovery: An Overview, pages 1–34. American Association for Artificial Intelligence, Menlo Park, CA, USA, 1996.

[37] Alex A. Freitas. Understanding the crucial differences between classification and discovery of association rules: a position paper. *SIGKDD Explor. Newsl.*, 2(1):65–69, June 2000.

[38] Alex Alves Freitas. On objective measures of rule surprisingness. In *Proceedings of the Second European Symposium on Principles of Data Mining and Knowledge Discovery*, PKDD '98, pages 1–9, London, UK, UK, 1998. Springer-Verlag.

[39] Takeshi Fukuda, Yasuhido Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Mining optimized association rules for numeric attributes. In *Proceedings of the fifteenth ACM SIGACT-SIGMOD-SIGART symposium*

on Principles of database systems, PODS '96, pages 182–191, New York, NY, USA, 1996. ACM.

[40] Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Constructing efficient decision trees by using optimized numeric association rules. In *Proceedings of the 22th International Conference on Very Large Data Bases*, VLDB '96, pages 146–155, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers Inc.

[41] Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Data mining with optimized two-dimensional association rules. *ACM Trans. Database Syst.*, 26(2):179–213, June 2001.

[42] Takeshi Fukuda, Yasukiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Data mining using two-dimensional optimized association rules: scheme, algorithms, and visualization. In *Proceedings of the 1996 ACM SIGMOD international conference on Management of data*, SIGMOD '96, pages 13–23, New York, NY, USA, 1996. ACM.

[43] Johannes Fürnkranz and Peter A. Flach. An analysis of rule evaluation metrics. In *ICML*, pages 202–209, 2003.

[44] Johannes Fürnkranz and Peter A. Flach. Roc 'n' rule learning towards a better understanding of covering algorithms. *Mach. Learn.*, 58(1):39–77, January 2005.

[45] Amol Ghoting, Prabhanjan Kambadur, Edwin Pednault, and Ramakrishnan Kannan. Nimble: a toolkit for the implementation of parallel data mining and machine learning algorithms on mapreduce. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 334–342, New York, NY, USA, 2011. ACM.

[46] Raúl Giráldez, Jesús S. Aguilar-Ruiz, and José Cristóbal Riquelme Santos. Natural coding: A more efficient representation for evolutionary learning. In *GECCO*, pages 979–990, 2003.

[47] Clark Glymour, David Madigan, Daryl Pregibon, and Padhraic Smyth. Statistical themes and lessons for data mining. *Data Min. Knowl. Discov.*, 1(1):11–28, January 1997.

[48] Salvatore Greco, Zdzislaw Pawlak, and Roman Slowinski. Can bayesian confirmation measures be useful for rough set decision rules? *Engineer-*

ing Applications of Artificial Intelligence, 17(4):345 – 361, 2004. Selected Problems of Knowledge Representation.

[49] Salvatore Greco, R. Slowinski, and I. Szczech. Analysis of monotonicity properties of some rule interestingness measures. *Control and Cybernetics*, 38(1):9–25, 2009.

[50] Jerzy W. Grzymala-Busse. Three strategies to rule induction from data with numerical attributes. pages 54–62, 2004.

[51] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November 2009.

[52] Jiawei Han. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[53] David J. Hand. Statistics and data mining: Intersecting disciplines. *SIGKDD Explor. Newsl.*, 1(1):16–19, June 1999.

[54] K. M. Ho and Paul D. Scott. Zeta: A global method for discretization of continuous variables. In *KDD*, pages 191–194, 1997.

[55] Robert V Hogg and Johannes Ledolter. *Engineering statistics*, volume 358. MacMillan New York, 1987.

[56] Robert C. Holte. Very simple classification rules perform well on most commonly used datasets. *Mach. Learn.*, 11(1):63–90, April 1993.

[57] X. Hong, S. Ye, T. Wan, D. Jiang, Z. Qian, R. Nan, N. Wang, Z. Shen, H. Zhang, and M. Wang. The developement of VLBI in China and its related to EVN. In *Resolving The Sky - Radio Interferometry: Past, Present and Future*, 2012.

[58] Paul Horton and Kenta Nakai. A probabilistic classification system for predicting the cellular localization sites of proteins. In *In Proceeding of the Fourth International Conference on Intelligent Systems for Molecular Biology*, pages 109–115, 1996.

[59] Mojdeh Jalali-Heravi and Osmar R. Zaïane. A study on interestingness measures for associative classifiers. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, SAC '10, pages 1039–1046, New York, NY, USA, 2010. ACM.

[60] Davy Janssens, Tom Brijs, Koen Vanhoof, and Geert Wets. Evaluating the performance of cost-based discretization versus entropy- and error-based discretization. *Computers & OR*, 33(11):3107–3123, 2006.

[61] George H. John and Pat Langley. Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, UAI'95, pages 338–345, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[62] Michael D. Kane and John A. Springer. Integrating bioinformatics, distributed data management, and distributed computing for applied training in high performance computing. In *Proceedings of the 8th ACM SIGITE Conference on Information Technology Education*, SIGITE '07, pages 33–36, New York, NY, USA, 2007. ACM.

[63] S. Kannan and R. Bhaskaran. Role of interestingness measures in car rule ordering for associative classifier: An empirical approach. *CoRR*, abs/1001.3478, 2010.

[64] Yiping Ke, James Cheng, and Wilfred Ng. An information-theoretic approach to quantitative association rule mining. *Knowl. Inf. Syst.*, 16(2):213–244, July 2008.

[65] Randy Kerber. Chimerge: discretization of numeric attributes. In *Proceedings of the tenth national conference on Artificial intelligence*, AAAI'92, pages 123–128. AAAI Press, 1992.

[66] Willi Klösgen. Advances in knowledge discovery and data mining. chapter Explora: a multipattern and multistrategy discovery assistant, pages 249–271. American Association for Artificial Intelligence, Menlo Park, CA, USA, 1996.

[67] Sotiris Kotsiantis and Dimitris Kanellopoulos. Discretization techniques: A recent survey. *GESTS International Transactions on Computer Science and Engineering*, 32:47–58, 2006.

[68] L.A. Kurgan and K.J. Cios. Caim discretization algorithm. *Knowledge and Data Engineering, IEEE Transactions on*, 16(2(2004)):145–153, Feb.

[69] Diane Lambert. What use is statistics for massive data? *Lecture Notes-Monograph Series*, 43:pp. 217–228, 2003.

[70] Jiuyong Li. On optimal rule discovery. *IEEE Trans. on Knowl. and Data Eng.*, 18(4):460–471, April 2006.

[71] Wenmin Li, Jiawei Han, and Jian Pei. Cmar: Accurate and efficient classification based on multiple class-association rules. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, ICDM '01, pages 369–376, Washington, DC, USA, 2001. IEEE Computer Society.

[72] Wang Lian, David W. Cheung, and S. M. Yiu. An efficient algorithm for finding dense regions for mining quantitative association rules. *Comput. Math. Appl.*, 50(3-4):471–490, August 2005.

[73] Bing Liu, Wynne Hsu, and Shu Chen. Using general impressions to analyze discovered classification rules. In *KDD*, pages 31–36, 1997.

[74] Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. pages 80–86, 1998.

[75] Bing Liu, Wynne Hsu, and Yiming Ma. Pruning and summarizing the discovered associations. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '99, pages 125–134, New York, NY, USA, 1999. ACM.

[76] Bing Liu, Wynne Hsu, Lai-Fun Mun, and Hing-Yan Lee. Finding interesting patterns using user expectations. *IEEE Trans. Knowl. Data Eng.*, 11(6):817–832, 1999.

[77] Huan Liu, Farhad Hussain, Chew Lim Tan, and Manoranjan Dash. Discretization: An enabling technique. *Data Min. Knowl. Discov.*, 6(4):393–423, October 2002.

[78] Huan Liu and Rudy Setiono. Chi2: Feature selection and discretization of numeric attributes. In *In Proceedings of the Seventh International Conference on Tools with Artificial Intelligence*, pages 388–391, 1995.

[79] Xu-Ying Liu and Zhi-Hua Zhou. Learning with cost intervals. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 403–412, New York, NY, USA, 2010. ACM.

[80] C. Armanino M. Forina, R. Leardi and S. Lanteri. Parvus: An extendable package of programs for data exploration, classification and correlation. *Journal of Chemometrics*, 4(2):191–193, 1988.

[81] Wolfgang Maass. Efficient agnostic pac-learning with simple hypothesis. In *COLT*, pages 67–75, 1994.

[82] Donato Malerba, Floriana Esposito, and Giovanni Semeraro. A further comparison of simplification methods for decision-tree induction. In *In D. Fisher & H. Lenz (Eds.), Learning*, pages 365–374. Springer-Verlag, 1996.

[83] J. Mata, J. L. Alvarez, and J. C. Riquelme. An evolutionary algorithm to discover numeric association rules. In *Proceedings of the 2002 ACM symposium on Applied computing*, SAC '02, pages 590–594, New York, NY, USA, 2002. ACM.

[84] Sreerama K. Murthy. Automatic construction of decision trees from data: A multi-disciplinary survey. *Data Min. Knowl. Discov.*, 2(4):345–389, December 1998.

[85] Pang ning Tan and Vipin Kumar. Interestingness measures for association patterns: A perspective. In *Tschinical Report 00-036*. Department of Computer Science, University of Minnesota, 2000.

[86] Qiang Niu, Shi-Xiong Xia, and Lei Zhang. Association classification based on compactness of rules. In *Proceedings of the 2009 Second International Workshop on Knowledge Discovery and Data Mining*, WKDD '09, pages 245–247, Washington, DC, USA, 2009. IEEE Computer Society.

[87] Miho Ohsaki, Shinya Kitaguchi, Kazuya Okamoto, Hideto Yokoi, and Takahira Yamaguchi. Evaluation of rule interestingness measures with a clinical dataset on hepatitis. In *Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, PKDD '04, pages 362–373, New York, NY, USA, 2004. Springer-Verlag New York, Inc.

[88] Balaji Padmanabhan and Alexander Tuzhilin. A belief-driven method for discovering unexpected patterns. In *KDD*, pages 94–100, 1998.

[89] Giulia Pagallo and David Haussler. Boolean feature discovery in empirical learning. *Mach. Learn.*, 5(1):71–99, May 1990.

[90] Gregory Piatetsky-Shapiro. Discovery, analysis, and presentation of strong rules. In *Knowledge Discovery in Databases*, pages 229–248. AAAI/MIT Press, 1991.

[91] J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, March 1986.

[92] J. Ross Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[93] Naren Ramakrishnan and Ananth Grama. Data mining: From serendipity to science - guest editors' introduction. *IEEE Computer*, 32(8):34–37, 1999.

[94] Salvatore Ruggieri. Frequent regular itemset mining. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 263–272, New York, NY, USA, 2010. ACM.

[95] Sigal Sahar. Interestingness via what is not interesting. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '99, pages 332–336, New York, NY, USA, 1999. ACM.

[96] Ansaf Salleb-Aouissi, Bert C. Huang, and David L. Waltz. Discovering characterization rules from rankings. In *ICMLA*, pages 154–161, 2009.

[97] Ansaf Salleb-Aouissi, Christel Vrain, and Cyril Nortet. Quantminer: a genetic algorithm for mining quantitative association rules. In *Proceedings of the 20th international joint conference on Artifical intelligence*, IJCAI'07, pages 1035–1040, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[98] Tobias Scheffer. Finding association rules that trade support optimally against confidence. *Intell. Data Anal.*, 9(4):381–395, July 2005.

[99] Abraham Silberschatz and Alexander Tuzhilin. On subjective measures of interestingness in knowledge discovery. In *KDD*, pages 275–281, 1995.

[100] Abraham Silberschatz and Alexander Tuzhilin. What makes patterns interesting in knowledge discovery systems. *IEEE Trans. Knowl. Data Eng.*, 8(6):970–974, 1996.

[101] Craig Silverstein, Sergey Brin, and Rajeev Motwani. Beyond market baskets: Generalizing association rules to dependence rules. *Data Min. Knowl. Discov.*, 2(1):39–68, January 1998.

[102] Ramakrishnan Srikant and Rakesh Agrawal. Mining quantitative association rules in large relational tables. *SIGMOD Rec.*, 25(2):1–12, June 1996.

[103] Liwen Sun, Reynold Cheng, David W. Cheung, and Jiefeng Cheng. Mining uncertain data with probabilistic guarantees. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 273–282, New York, NY, USA, 2010. ACM.

[104] Ming Tan and Larry J Eshelman. Using weighted networks to represent classification knowledge in noisy domains. In *ML*, page 121, 1988.

[105] Pang-Ning Tan, Vipin Kumar, and Jaideep Srivastava. Selecting the right interestingness measure for association patterns. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 32–41, New York, NY, USA, 2002. ACM.

[106] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[107] F. A. Thabtah and P. I. Cowling. A greedy classification algorithm based on association rule. *Appl. Soft Comput.*, 7(3):1102–1111, June 2007.

[108] Rick Bertelsen Tony and Tony R. Martinez. Extending id3 through discretization of continuous inputs. In *In Proceedings of the Seventh Florida AI Research Symposium (FLAIRS'94*, pages 122–125, 1994.

[109] Pauray S. M. Tsai and Chien ming Chen. Mining quantitative association rules in a large database of sales transactions. *Journal of Information Science and Engineering*, 17:667–681, 2000.

[110] Ke Wang and Bing Liu. Concurrent discretization of multiple attributes. In *In Proceedings of the Pacific Rim International Conference on Artificial Intelligence*, pages 250–259. Springer-Verlag, 1998.

[111] Zhenxing Wang and Laiwan Chan. An efficient causal discovery algorithm for linear models. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 1109–1118, New York, NY, USA, 2010. ACM.

[112] Leland Wilkinson, Anushka Anand, and Dang Nhon Tuan. Chirp: a new classifier based on composite hypercubes on iterated random projections. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 6–14, New York, NY, USA, 2011. ACM.

[113] Hong Yao and Howard J. Hamilton. Mining itemset utilities from transaction databases. *Data Knowl. Eng.*, 59(3):603–626, December 2006.

[114] Y. Y. Yao and Ning Zhong. An analysis of quantitative measures associated with rules. In *Proceedings of the Third Pacific-Asia Conference on Methodologies for Knowledge Discovery and Data Mining*, PAKDD '99, pages 479–488, London, UK, UK, 1999. Springer-Verlag.

[115] Xiaoxin Yin and Jiawei Han. Cpar: Classification based on predictive association rules. In *SDM*, 2003.