Investigating Data Mining in MATLAB®

Submitted in partial fulfilment
of the requirements of the degree
Bachelor of Science (Honours)
of Rhodes University

Douglas Trewartha
Supervised by: John Ebden
November 2006

RHODES UNIVERSITY
Where leaders learn

Abstract

Data mining is a rapidly expanding field in many disciplines. It is becoming increasingly necessary to find data mining packages appropriate for a given analysis. The reasons as to why MATLAB (version 7.0) is the correct kind of package to use and its particular advantages with regards data mining are discussed in this work. MATLAB already supports various implementations of different stages of the data mining process, including various toolboxes created by experts in the field. An initial conclusion of this study is that MATLAB is a powerful and versatile package for fulfilling the requirements of the data mining process. It is clear, however, that there is a need for the extension and synthesis of the existing tools. Three such tools have been investigated fully; analysis of each tool is provided, with recommendations for further extensions.

The synthesis of data mining tools outlined and demonstrated in this thesis allows for a far more holistic approach to data mining in MATLAB than has been available previously. This work ensures that data mining becomes an increasingly straightforward task, as the appropriate tools for a given analysis become apparent. As a logical extension of the synthesis provided, a brief discussion is given with regard the creation of a data mining toolbox for MATLAB.

The open-endedness of this study provides many areas for further investigation and further synthesis, both within MATLAB and in the field of data mining as a whole.

Acknowledgements

The greatest thanks and that which overarches everything, goes to Jesus Christ, without whom, neither I, nor anything of this work, nor the universe in which we live, would exist. Thank you for inspiring me to do everything with all my heart; for You and not for men. Thank you for always being present in every aspect of this past year and for allowing me to come to You with all my worries and concerns. This work was only possible because of Your love.

Secondly I must thank John who has been the greatest supervisor I could have asked for. He knows that I like structure and has provided that in the most positive way. I hope this work brings much credit to you as supervisor; it wouldn't have been possible without your invaluable advice.

To my girlfriend, Kallah, thank you for putting up with me through the longer hours of the writing of this thesis. You are always there for me and your constant support has made this work possible, thank you!

Thank you to my family who made it possible for me to be studying at Rhodes this year. I hope that this is a work that you can be proud of in years to come and that all I have achieved over the past four years will continue to bring you joy.

To my many other friends, particularly Gondayi, Jeremy and Megan, thank you for your support and friendship which has made this year a happy and fruitful one.

Lastly I would like to acknowledge the financial support of Telkom SA, Business Connexion, Comverse SA, Tellabs, Verso Technologies, THRIP and Stortech, through the Telkom Centre of Excellence. Thank you for your assistance in the maintenance and provision of the excellent services offered by the Rhodes Computer Science Department.

Contents

List of Figures

List of Tables

Chapter 1: Introduction and Background

1.1. Problem Statement

As data repositories grow, there is an increasing need for data mining tools, which are able to glean information from data sets that are not easily understood by traditional observation or experiment.

Data mining is the means used in extracting hidden knowledge from a data set; this would be knowledge that is not readily obtained by traditional means such as queries or statistical analysis [Roiger and Geatz 2003]. Hidden knowledge can be used for classification and estimation of new instances and for prediction of future events [Roiger and Geatz 2003].

MATLAB has been used in the development of data mining tools but it is required to know to what extent the requirements of the data mining process are met by the tools currently available for MATLAB and hence by the package as a whole. In addition, it is required to know the necessity and feasibility of creating a toolbox dedicated to data mining.

1.2. Project Aim

Hence, we aim to provide in this thesis, not only an analysis of selected data mining tools available within MATLAB and a synthesis of these tools, but more importantly, a means to analyse and synthesise further data mining tools, thus providing an increasingly holistic view of the data mining capabilities of MATLAB.

Essentially then, we wish to discover the extent to which each of a number of MATLAB data mining tools is capable of carrying out the different stages of the data mining process. We wish to synthesise these tools in order to bring greater clarity to the potential of MATLAB in the data mining arena and to provide recommendations for further extension to these tools in light of this analysis and synthesis. And, as we do this,

to clearly define the methodology used in carrying out this work, in order that it might be used in future work in this area.

In summary, our aim is to create a means for obtaining a holistic view of the data mining capabilities of MATLAB. We will accomplish this by setting forth the methodology of this process and by demonstrating this methodology by investigating and synthesising several data mining tools available for MATLAB.

1.3. Project Motivation

MATLAB is a powerful and versatile tool, more than capable (as we shall show) of performing data mining. It is clear that MATLAB has not been given due attention in this area. Figure 1.1 illustrates that while a relatively popular data mining tool, MATLAB is not yet in the league of packages such as Clementine, Weka and even Excel. In addition, though MATLAB is chosen more frequently than Oracle, it is generally used in conjunction with other tools. Whereas Oracle is implemented as a stand alone tool over 50% of the time, MATLAB is used on its own just over 12% of the time.

Table 1.2 summarises the position of MATLAB over the past 7 years. Despite the fact that MATLAB is currently capable of performing some of the most popular data mining techniques available, such as those being analysed in this project, it has not yet become one of the packages of choice in this field. The popularity of these techniques is detailed in Table 1.1, which is based on a sampling of 16 different data mining techniques over the 4 year period from 2002 to 2005 [KDnuggets 2006].

One reason for MATLAB's limited use may be the fact that it is a proprietary package. However, the basic MATLAB package is easily enhanced, particularly by using open source toolboxes and script bundles, such as those examined in this study. The fact that MATLAB's data mining potential has certainly not been fully exploited (as demonstrated in Figure 1.1 and Table 1.2), together with the recent demand for data mining tools, is the central motivation for carrying out this study.

The synthesis of data mining tools provided in this thesis allows for a far more holistic approach to data mining in MATLAB than has been available previously and in addition, ensures that MATLAB can be used as a stand alone tool, rather than in

conjunction with other packages. This work ensures that data mining in MATLAB becomes an increasingly straightforward task, as the appropriate tools for a given analysis become apparent. As a logical extension of the synthesis provided, recommendations are given with regard the creation of a data mining toolbox for MATLAB.

The opportunities for extension to this work are numerous, not only in terms of extending the tools themselves but also of data mining in MATLAB as a whole.

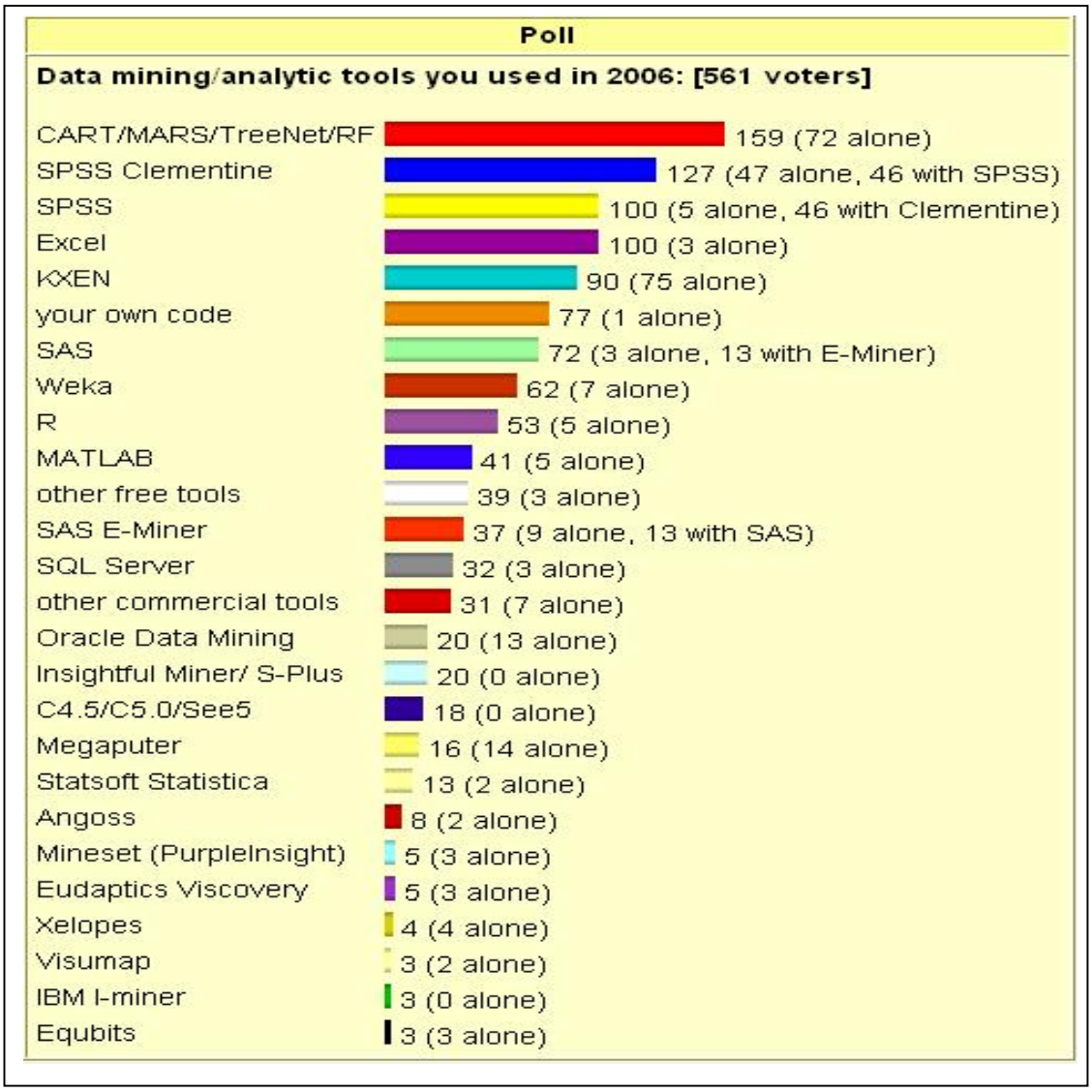


Figure 1.1: *2006 Data Mining Tools Poll 1138 Votes*
MATLAB Ranks 10th with 5% of the votes
[KDnuggets 2006]

Technique	***2002***	***2003***	***2004***	***2005***
Decision Tree	Rank:1 (16%)	Rank:1 (16%)	Rank:1 (17%)	Rank:1 (14%)
Clustering	Rank:2 (12%)	Rank:2 (12%)	Rank:3 (11%)	Rank:2 (13%)
Neural Nets	Rank:5 (9%)	Rank:4 (9%)	Rank:5 (9%)	Rank:6 (8%)
Association Rules	Rank:6 (8%)	Rank:7 (5%)	Rank:4 (9%)	Rank:7 (7%)

Table 1.1: *Polls of Popular Data Mining Techniques 2002-2005 [KDnuggets 2006]*

MATLAB	***2000***	***2001***	***2002***	***2003***	***2004***	***2005***	***2006***
Rank	∞	7	7	14	9	15	10
%	n/a	5	5	3	2	2	5

Table 1.2: *Popularity of MATLAB in Data Mining 2000-2006 [KDnuggets 2006]*

1.4. Project Overview

Due to the broad and open-ended nature of this study it is vital that we focus on a number of specific tools and case studies. The data mining tools around which this thesis will revolve are: the Neural Network Toolbox, a proprietary tool available from The MathWorks, distributors of MATLAB. The Fuzzy Clustering and Data Analysis Toolbox [Balasko et al. 2005] and the Association Rule Miner and Deduction Analysis tool [Malone 2003], which are both open source; and lastly an implementation of the C4.5 decision tree algorithm [Woolf, 2005]. The fact that these are some of the most popular tools currently being used, as demonstrated in Table 1.1 above, strengthens this study, which can easily be extended to other tools available within MATLAB.

We will be looking at a number of specific implementations of our process of synthesis. This entails the use of different case studies on separate data sets using the same process. The crucial difference between these data sets is that the dependant data attribute (that property which we are attempting to estimate or predict) is continuous in nature for one of the data sets and categorical for the others.

1.5. Overview of MATLAB

"MATLAB has excellent built-in support for many data analysis and visualization routines," [Murphy 2005] in particular, one of its most useful facilities is that of efficient exploratory data analysis [Vlachos 2005], which is a natural fit in the context of data mining [Hand et al. 2001]. The paper by Dwinnell [1998] discusses MATLAB in the context of fourth generation languages and highlights some important advantages of MATLAB over previous generations. It is important to understand the reasons for choosing MATLAB as our data mining tool [Paola et al. 2006] as this will allow us to raise questions of the software and have ready answers for these questions.

1.5.1. Fourth Generation Languages

As programming methodologies have progressed, the notion of 'generations' has become important in distinguishing the level of abstraction provided and the extent to which certain constraints of previous generations have been removed [Dwinnell 1998]. This progression has advantages in that it allows for easier implementation as the details are abstracted away [Dwinnell 1998]. Thus, when one wishes to perform specific as well as specialized tasks, such as data mining, it makes sense to implement these in a fourth generation language (4GL), such as MATLAB.

1.5.2. Advantages of MATLAB®

Those who have written open source toolboxes, such as Murphy [2005] and the developers of the Fuzzy Clustering and Data Analysis Toolbox [Balasko et al. 2005] share Dwinnell's [1998] views with regard the usefulness of MATLAB.

Two major advantages are that of portability and domain specific representations [Dwinnell 1998]. MATLAB's portability comes from the fact that all MATLAB users will have the same range of basic functions at their disposal [Dwinnell 1998]. The representation which MATLAB implements, is dealing with all data in the form of matrices [Dwinnell 1998]. This allows for many varied algorithmic implementations [Dwinnell 1998], which, as we shall see, is crucial for any data mining package.

Other advantages of MATLAB include its interactive interface, debugging facilities, object oriented nature and in particular its high quality graphics and visualisation

facilities [Burton 2006]. Lastly and perhaps most importantly, MATLAB's add on feature, in the form of toolboxes, makes it possible to extend the existing capabilities of the language with ease [Burton 2006]. Data mining, which, for the most part, consists of numerical methodologies [Woolf 2005], is thus a natural fit for implementation with the MATLAB package.

1.5.3. Disadvantages of MATLAB®

The main drawback of MATLAB is the fact that it is an interpreted language [Vlachos 2005], which leads to performance cuts, as compared with third generation languages such as C, upon which MATLAB is built. In the context of data mining, this can be a very serious issue, particularly when one is dealing with enormous quantities of data [Roiger and Geatz 2003]. Dwinnell [1998] points out that 4GL's with their own compilers are able to largely overcome this disadvantage. MATLAB does in fact possess its own compiler but it is distributed as a toolbox by The MathWorks (distributors of MATLAB). Lastly, although we do not have access to the compiler toolbox, which is proprietary software, our intention is not to deal with massive data sets but rather to illustrate the data mining capabilities of MATLAB by synthesising and extending existing works.

1.5.4. MATLAB® Summary

The need to investigate the data mining capabilities of MATLAB further than has been done previously is certainly evident. The most recent work in this area was that by Woolf [2005], whose work we intend to investigate further here. The focus of Woolf [2005] was the creation of a decision tree algorithm, yet another addition to the data mining facilities available for MATLAB.

The urgent need once again seems to be for a holistic overview of the data mining capabilities of MATLAB. It is the aim of this work to bring together a number of the available tools and to give direction to further such advances in order that such an overview might be obtained and the full potential of MATLAB in this vital field realised.

1.6. Overview of Project Chapters

Chapter 2: Design Considerations – Lays out the details of the work done in this thesis. This chapter is of great importance in that it presents the methods used in both investigating and synthesising the tools.

Chapter 3: Tool Investigation – Begins by introducing the case studies upon which the experiments carried out are to be built. Proceeds with the investigation of each of the toolboxes, outlining the investigations carried out and any problems encountered in this area. Essentially contains preliminary findings of this work, which are necessary for the implementation of our synthesis of tools.

Chapter 4: Implementation and Results – Brings together the investigation of the tools as the results of synthesis are presented and discussed.

Chapter 5: Findings and Evaluation – A brief evaluation of the results presented in Chapter 4 based on other similar case studies which were carried out as part of the investigative process of this work. The results of this evaluation are then summarised by providing suggestions with regard the creation of a data mining toolbox for MATLAB.

Chapter 6: Conclusion and Possible Extensions – Concludes the project, presenting both the findings of this work and the many possibilities for further research in this area.

1.7. Chapter Summary

In this chapter we have discussed the direction and aims of this study. We have also gained an overview of MATLAB and what is required for us to achieve with respect to data mining within this package. It is extremely exciting to embark on something as new as this, particularly since the work done here could not only enhance the usefulness of MATLAB in performing data mining, but also bring greater clarity to its place in the field as a whole.

We now embark on the development of the methodology required to attain the objectives which have been laid out.

Chapter 2: Design Considerations

Our aim, essentially, is to discover to what extent our chosen tools are capable of carrying out the different stages of the data mining process and to synthesise and recommend extensions to these tools. This will bring greater clarity to the potential of MATLAB in this field. Clearly for our purposes it is important that we understand, first data mining in a broad context and second, each stage of the data mining process. The methodology which we have chosen to use is modular in that it focuses on each stage of the data mining process separately. The reason for this choice is that it enables us to focus on the capabilities of each tool at the separate stages of the process and thus to obtain a comprehensive analysis of the tool. In addition, developing the process in this manner will aid similar work in the future.

2.1. Introduction: Data Mining

To lead us into our analysis of the data mining process, let us look briefly at the broad means used in mining data, namely, supervised and unsupervised learning.

2.1.1. Supervised Learning

Induction based supervised concept learning is the most widely used technique in the field of data mining [Roiger and Geatz 2003]. It involves induction based model creation and testing based on a set of training data and the subsequent application of the model in deductive classification of new data [Roiger and Geatz 2003].

In general, supervised learning can be classified into one of three categories:

1. classification, dealing with categorical data and current behaviour;
2. estimation, dealing with current behaviour but continuous numerical data;
3. or prediction, which deals with future behaviour and data of either type

[Roiger and Geatz 2003]. Supervised learning models include decision trees, neural networks and association rules, which are assessed in this study.

In this thesis we will be investigating a number of supervised learning techniques and applying them in different case studies. Five case studies were carried out during the course of this research with the main difference in each case being the data set analysed. The major difference between the case studies carried out is that one involves estimation, as the dependant variable is continuous, whilst the others require classification, with the dependant variable being categorical.

2.1.2. Unsupervised Clustering

Unlike the supervised learning method, no dependant variable exists when performing clustering [Roiger and Geatz 2003]. The idea behind unsupervised clustering is the discovery of concept structures within data and the formulation of questions regarding the data set, which we would never have thought to ask with only the data before us [Roiger and Geatz 2003]. Clustering demonstrates whether useful information does in fact exist in the data set and can help determine a best set of inputs for a supervised learning model [Roiger and Geatz 2003]. It can also help in the preparation of data, for example, in the detection of outliers.

A fuzzy clustering and data analysis toolbox for MATLAB has already been developed by Balasko et al. [2005] and will be investigated in this thesis using the case studies mentioned.

2.1.3. Hybrid Learning

The real potential of data mining is only fully realised when a hybrid of the above two techniques is implemented. This is one of the central aims of this thesis; to sythesise a number of different data mining tools in order to discover the potential of MATLAB in this field.

Not all algorithms do equally well in all areas [Adriaans and Zantige 1997] and it thus makes sense to implement a variety of algorithms, particularly when tackling a large project. Unsupervised clustering techniques best serve as an evaluation tool for supervised learning [Roiger and Geatz 2003]. Learner attributes must be carefully

selected when carrying out supervised learning. For example, in the construction of decision trees, nodes higher up the tree need to be those attributes with a higher information gain [Woolf 2005]. By repeatedly applying unsupervised clustering with different attribute choices the most general attributes will become evident. These attributes are those most appropriate for serving as nodes at higher levels in the tree (the root node ought, necessarily, to be the node with the highest information gain over the data set) [Roiger and Geatz 2003].

Further implementations within MATLAB include ARMADA, Association Rule Miner and Deduction Analysis [Malone 2003], which we will investigate further; and toolboxes which we will not investigate, such as the Bayes Net Toolbox [Murphy 2005] and the Self Organising Map Toolbox [Vesanto et al. 2000]. Additional techniques which are beyond the scope of this study but which can also be combined (or used individually) to perform data mining are genetic algorithms and statistical techniques.

The above discussion again outlines the need for the creation of some methodology which can be used in the synthesis of tools, so as to achieve hybrid learning in MATLAB, and it is the development of this methodology to which we now progress.

2.2. The Data Mining Process

The data mining process can be broken down into four distinct phases.

1. Decision, whether or not to carry out data mining on a given data set.
2. Data preparation, readying the data for analysis.
3. Model building, where the work of building a prediction model is carried out.
4. Interpretation, which is largely carried out by individuals but which can be greatly assisted using automated means, such as graphical representations of the results.

This process is illustrated using a flow chart in Figure 2.1. For each stage of the process discussed below we will introduce the means which we feel most suited to carrying out that stage, whether supervised, unsupervised or a combination. This will serve as preparation for our tool assessment, carried out in Chapter 3 of this thesis.

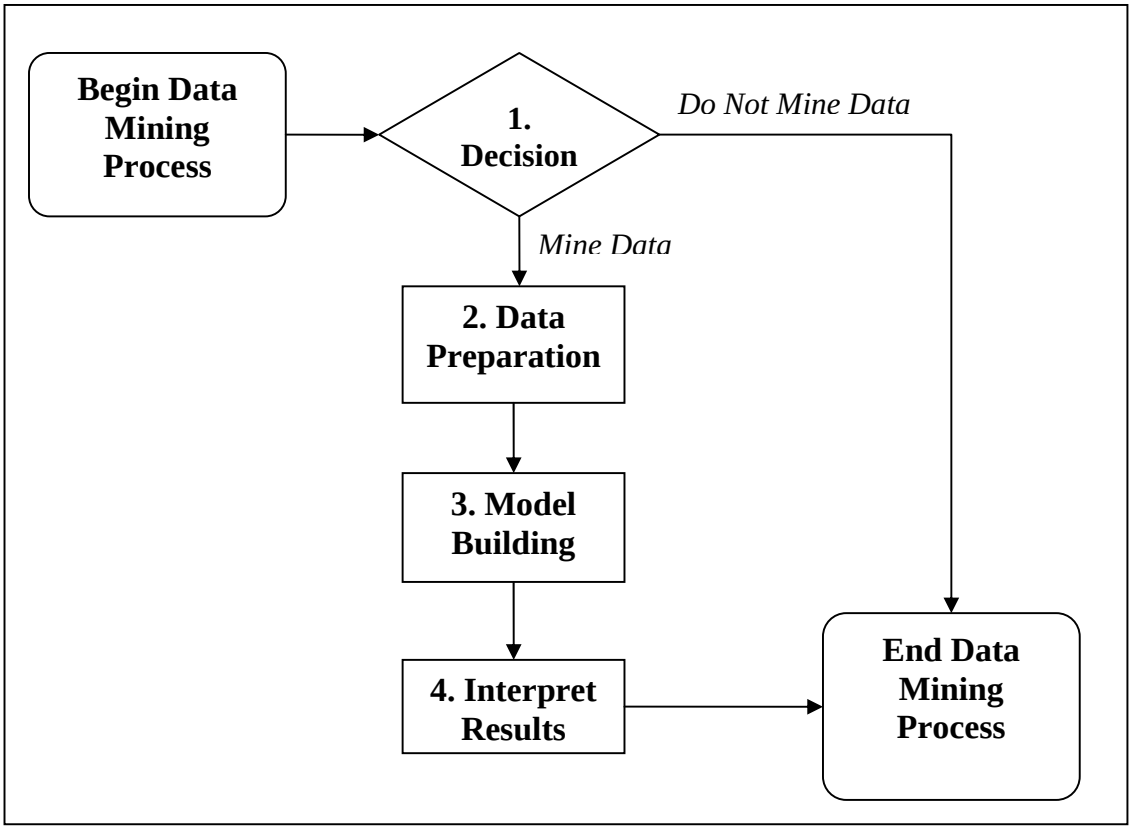


Figure 2.1: *The Four Phases of the Data Mining Process*

2.2.1. Decision Phase

The first stage in the data mining process is that of deciding whether or not to go ahead with a given analysis. This is one of the most difficult and probably the most crucial of all the stages, as it is here that we decide whether or not we are going to spend our time and other resources investigating a given data set [Roiger and Geatz 2003]. Although this stage is often given to humans, who are able to ask and answer questions pertaining to the identified problem domain, it is also useful to have automated means to help in the taking of this first step.

Unsupervised clustering is one means of determining whether relationships, in the form of concepts, exist in the data. If a clustering finds such concepts, then it can be deduced that a supervised model is likely to perform well, leading us to continue in the data mining process [Roiger and Geatz 2003].

2.2.2. Data Preparation Phase

Once we have decided to go ahead with our investigation, it is vital that the data be in a format that can be easily interpreted by the model building tool [Pyle 1999]. Data preparation is a vast topic. The scope of this study requires that we understand the importance of data preparation to the process as a whole and the need for such facilities in the tools which we investigate.

Examples of data preparation include:

- the search for outliers,
- the discretisation of continuous data
- and normalisation.

Certain tools are given almost entirely to this stage of data mining and unsupervised clustering is also widely used in this stage of the process, particularly in the discovery of outliers [Roiger and Geatz 2003].

The more pre-processing applied to a given data set, the better the results from creating a data mining model of that data set are likely to be. The more pre-processing a tool offers, whether supervised or unsupervised, the better is likely to be the performance of models created by that tool or subsequently applied tools.

2.2.3. Model Building Phase

Model building is the core of the data mining process. This is where verifiable results are obtained. The scope of this study is limited to supervised learner models. What this essentially means is that the models we create will have been trained using examples of known cases (from the data set) and then verified using further information from the data which has not yet been presented to the model. This stage is known as the training and testing or validation stage and once completed the model produced can be used to predict future outcomes, instances which have neither been seen by the model nor by individuals.

For example, in creating a neural network, a proportion of the instances in a given data set will be used to “train” the network. The neural network is then tested in order to

discover what it "knows" using the remaining data and once verified as an effective model is used in classification or prediction of unseen cases.

In this study, The Mathworks Neural Networks Toolbox will be the tool used most for the purpose of model creation.

2.2.4. Interpretation Phase

The final stage of the data mining process is that of interpretation. This stage is vital to the process as, "it is the analysis of results provided by the human element that ultimately dictates the success or failure of a data mining project" [Roiger and Geatz 2003]. As with the decision phase, however, our interpretation of the results can be assisted using automated means.

One such means is visualisation tools, which illustrate what is known by the model. MATLAB is renowned for its graphical presentation of results and all of the tools examined in this paper will illustrate the power of MATLAB in assisting the user with the interpretation phase of the data mining process.

Other automated means of interpretation are mathematical validity measures of the algorithms implemented. As we might expect, a mathematical package such as MATLAB is easily able to carry out these measures of validity. These features will be explored both for the unsupervised clustering toolbox [Balasko et al. 2005] and for the neural network models created using supervised learning techniques.

2.3. Methodology

The approach of this study can be broken down into two distinct phases. The first phase is that of analysis and assessment of the package, where we aim to validate the tool's claims and to suggest possible extensions to the tools. The second phase is that of synthesis, where we use the tools in combination and then provide a final analysis of the results of the processes implementation.

2.3.1. Tool Assessment

This phase is essentially an extended documentation of the tool. It should be clear that some experimentation with the tools is necessary in order to obtain these results. The case studies carried out are the best examples of such work and are detailed in Chapters 4 and 5. Provision of the details of every experiment is not possible within this work, although some of these details are dealt with in the appendices.

The first stage of the assessment process is a critical evaluation of the claims of the tool. These claims are evaluated in terms of that stage of the data mining process to which they pertain. Once the claims have been validated, or otherwise, suggestions are made as to how the tool could best be improved so as to fulfil its purpose more completely.

Second, the applications of the tool are briefly explored, that is, the stage or stages of the data mining process to which the tool is best suited are highlighted. The case studies will serve as specific examples of these applications and will be investigated in greater depth so as to attain examples of implementations of the tools, which obtain meaningful results (Chapter 4).

The final step in the assessment process is to further clarify ideas pertaining to the synthesis of the tool. Naturally each tool is suited to a particular domain and is constrained to fulfil certain stages of the data mining process more completely than others. This fact highlights both the need for synthesis and that which makes synthesis possible. Essentially, this stage of our methodology is aimed at focussing us on the area of greatest potential for the given tool so as to streamline the synthesis process.

2.3.2. Synthesis

Synthesis has been defined as the process of designing or building a new concept for a specific purpose, by putting parts together in a logical way. This comes closest to what we are doing in this study and the methodology for synthesising the chosen tools is where the true potential of this work to impact the way in which data mining is carried out by MATLAB is evidenced. At present no clear means exists for the synthesis of the data mining tools in MATLAB, which is a central reason for MATLAB's limited popularity, particularly as a stand alone tool. The tools currently available were either designed, with

a specific application area in mind, to solve a specific problem, or merely out of a desire to extend the capabilities of MATLAB, with little or no thought to the impact on the data mining capabilities of MATLAB as a whole. Data mining is an extremely broad field and for MATLAB to become a tool of choice in this field, a means must exist for the synthesis of the available tools.

The first stage in this process is to decide which tools are to be synthesised. As discussed, this thesis will focus on the synthesis of The MathWorks Neural Network Toolbox, Fuzzy Clustering and Data Analysis Toolbox [Balasko et al. 2005] and ARMADA (Association Rule Miner and Deduction Analysis tool) [Malone 2003].

The second stage in the process is to determine where and how these tools complement each other. This stage is, once again, firmly rooted in the data mining process and will thus already have been discussed during the course of the individual tool assessments. For example, where the Neural Network Toolbox may be deficient with regards the first two stages of the process, the Fuzzy Clustering Toolbox plays a crucial role and although the clustering tool on its own does not produce a useful model, it is a necessary precursor to obtaining the best possible results from the potential neural network (see Appendix D for the project poster which illustrates this).

Essentially, the ways in which the tools complement one another are highlighted and the suggested synthesis is then implemented.

2.3.3. Final Analysis

Lastly, the process as a whole needs to be evaluated. This is crucial in providing a holistic view of what has been achieved by the synthesis for data mining in MATLAB as a whole. The results of this last stage are evaluated in order to determine what has been gained and the new process can then, in many ways, be viewed as a "tool" in its own right.

The process is thus a circular one, as illustrated in Figure 2.2 below. This fact makes it all the more clear how a complete overview of MATLAB can be attained; once all of the currently available data mining tools have been assessed and synthesised. MATLAB will gain from this in many important ways. Most crucially perhaps, it will be clear to what extent MATLAB is capable of carrying out the data mining process as a whole and if it lacks in any area, this too will be evidenced through the extension of this work.

2.4. Chapter Summary

In this chapter, we have introduced the means by which data mining is carried out as well as the data mining process. This was necessary in order to construct our methodology, which centres on the data mining process.

Figure 2.2 is the best summary of the methodology which we have constructed in this chapter; it illustrates the broad methodology and the details of each of the broad components. In the diagram below the arrows represent a sequential flow through the process and the dotted arrows indicate a path which may or may not be taken in the process.

The broad methodology is to first evaluate the individual tools, which facilitates the synthesis of these tools and then finally to analyse the end result. Tool assessment involves determining whether or not the claims of the tool are met, what the best uses of this tool are and in particular, the phase of the data mining process to which the tool is best suited. Importantly, we must keep in mind the possibility of extending the capabilities of the existing tools, so that either they meet all of their initial claims adequately or are able to carry out their function more completely. It should be clear that tools will not always need extending and so this step is optional. When synthesising, we first decide on the tools which we will use in combination, we then designate each tool to a particular phase (or phases) of the data mining process and finally carry out the implementation of the synthesis. The final analysis is similar to a fresh assessment of the newly created "tool" and this final stage of the methodology builds up the holistic view of the data mining capabilities of MATLAB, which we are aiming for.

We now progress to Chapters 3 and 4, where we carry out our investigation and synthesis of the chosen tools. Chapter 5 serves as the final analysis of this methodology, particularly with regard to the worked examples of Chapters 3 and 4.

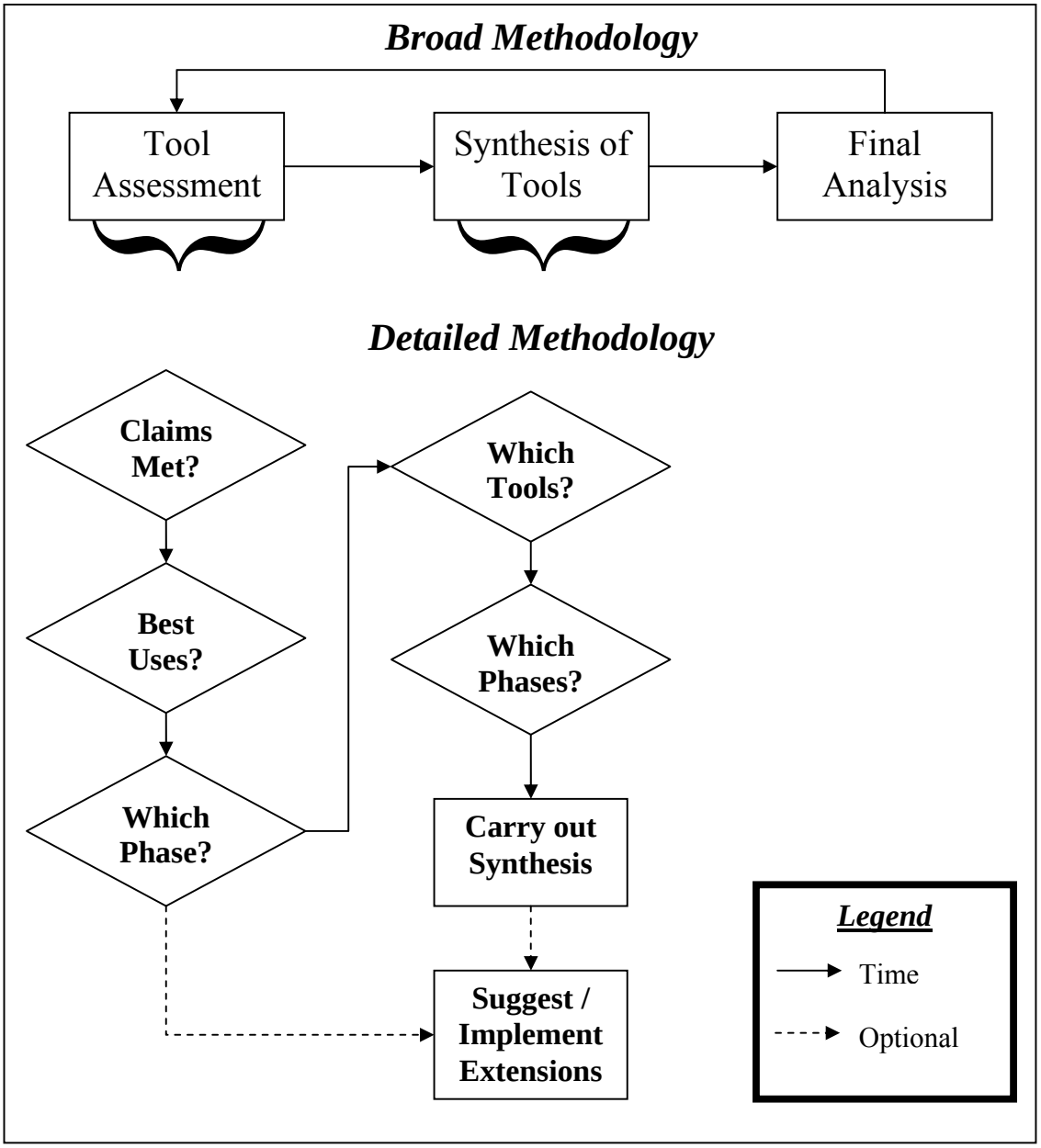


Figure 2.2: *Broad and Detailed Methodology for the Synthesis of Data Mining Tools in MATLAB*

Chapter 3: Tool Investigation

In this chapter we intend to carry out the assessment phase of the methodology and to introduce the synthesis of the tools analysed, which will largely be carried out in the Chapter 4. The final analysis phase is largely carried out in Chapter 5.

During the analysis of our chosen tools it will be necessary to have some idea as to the application of these tools. We will thus introduce the two case studies which are central to this work. It will be necessary to refer to these case studies at various stages during our investigation but most of the work done on the data will be presented in Chapters 4 and 5. After looking briefly at these case studies we will analyse each of our tools by first defining our scope, then by analysing the claims and lastly by suggesting extensions and the best uses of each of our tools, as discussed in the previous chapter.

3.1. Case Studies

As mentioned previously, the major difference between the two data sets is that the dependant variable is categorical for the first and continuous for the second.

3.1.1. Case Study 1: Japanese Business Solvency

This database contains records of various Japanese Businesses and shows whether these businesses are either solvent or bankrupt. The attributes in the table represent financial indicators, which will be defined. These indicators are relevant in determining whether a business is likely to be solvent or bankrupt and rather than explaining each statistic in detail this will be demonstrated in the course of the data mining process. The most important thing for us to note regarding this database is that a solvent business is indicated by a '*1*' and a bankrupt business by a '*0*' in the last column of the database, headed, "*Solvent*". This is the dependant variable which is clearly categorical in nature. The aim of mining this data will be to classify each of the businesses as either solvent or bankrupt. In addition to this it should be clear that it would be difficult, if not impossible,

to determine the cause of a given business being either bankrupt or solvent using traditional statistical means (such as regression), using queries or by observation.

A small portion of this database is shown below in Figure 3.1 and the attributes defined. The full database can be found both on the website (http://research.ict.ru.ac.za/g03t2052) and the accompanying CD.

	A	B	C	D	E	F	G	H	I	J
1	Firm	EBIT/TA	NI/TC	Sales/TA	EBIT/Sales	NI/Sales	WC/TA	Equity/TL	Equity/TA	Solvent
2	Chuo Seikakusho	0.096	0.089	1.505	0.064	0.027	0.35	0.708	0.414	1
3	Dai-Nippon Sugar Mfg	-0.0701	0.1246	3.7913	-0.0185	-0.0185	-1.0029	-0.4496	0.076	0
4	Fuji Seito	-0.337	0.779	2.948	-0.114	-0.135	-1.042	-0.4	-0.667	1
5	Gisen	0.034	0.03	1.676	0.02	0.012	0.089	0.668	0.4	1
6	Hikari Bussiness For	0.067	0.079	1.449	0.046	0.015	-0.08	0.186	0.157	1
7	Meiji Sugar Mfg.	-0.71	0.9775	3.8498	-0.1844	-0.1844	-1.1348	-0.4641	-0.413	0
8	Monde Distilleries	0.0101	0.0605	0.906	0.0112	0.0112	-0.2936	0.1852	-0.333	0
9	Morio Denki	0.108	0.075	1.143	0.094	0.041	0.401	1.387	0.581	1
10	Nagoya Lumber	0.019	0.039	2.177	0.009	0.003	0.064	0.201	0.617	1
11	Nagoya Seito	-0.1307	-0.3354	2.4262	-0.0539	-0.0539	-0.2872	-0.1046	0.079	0
12	Nankai Worsted Spin	0.052	0.116	1.922	0.027	0.019	0.112	0.393	0.282	1
13	SR Suntour	0.085	0.087	1.317	0.064	0.035	0.197	0.516	0.34	1
14	Takeni Senka	-0.2674	-1.5182	1.64	-0.1631	-0.1631	-0.2326	-0.2925	-0.653	0
15	Toyo Sugar	-0.002	-0.302	2.176	-0.001	-0.01	-0.543	0.007	0.006	1
16	Yamato Woolen	0.0265	0.1076	0.8962	0.0296	0.0296	-0.1839	0.2039	0.169	0
17	Yamato Woolen Textil	0.0265	0.1076	0.8962	0.0296	0.0296	-0.1839	0.2039	0.169	0

Figure 3.1: *Japanese Business Solvency Database Extract*
(WC=Working Capital; TA=Total Assets; EBIT=Earnings Before Interest and Tax; NI=Net Income; TC=Total Cost; TL=Total Liabilities)
[Simonoff 2003]

3.1.2. Case Study 2: Isomerisation of n-Pentane

Our second database contains details of the rate of a chemical reaction known as isomerisation. Once again, we need not know the details of this process and it will suffice

for us to know that an isomer (isopentane) is being produced from n-pentane in a reactor. The three inputs to the reactor are partial pressures and the output is the reaction rate per hour [Burton 2006].

The dependant variable in this case is the reaction rate, which is once again found in the last column of the table. This variable is clearly continuous in nature. The aim of mining this data will be the estimation of the reaction rate for the process of the Isomerisation of n-Pentane.

A fragment of this database is shown below in Figure 3.2. The full database can be found both on the website (http://research.ict.ru.ac.za/g03t2052) and on the accompanying CD.

	A	B	C	D
1	**isomerisation**			
2	**partial pressures**			**rates in h^{-1}**
3	**hydrogen**	**n-pentane**	**isopentane**	**rate**
4	100	100	10	9.641873
5	100	300	115	8.820432
6	150	100	10	7.726269
7	150	100	25	6.218905
8	150	300	70	10.66218
9	150	300	85	9.702286
10	200	100	10	6.445672
11	200	100	25	5.274262
12	200	250	40	10.18856
13	250	100	10	5.529226
14	250	100	100	1.344086
15	250	300	115	6.867569
16	300	100	10	4.840941
17	300	300	115	6.395571
18	350	100	10	4.305043
19	350	300	100	6.481481
20	350	300	115	5.98428
21	400	100	10	3.875969
22	400	300	115	5.622692
23	450	100	10	3.524673
24	450	300	115	5.302311
25	500	100	10	3.231764
26	500	200	70	4.104211
27	500	200	85	3.705619

Figure 3.2: *Isomerisation of n-Pentane Database Extract [Burton 2006]*

3.2. Tool Assessment

3.2.1. The MathWorks Neural Network Toolbox

A neural network is an interconnected group of nodes similar to the vast network of neurons in the human brain. In most cases a neural network is an adaptive system that changes based on the information that flows through it.

An example of a neural network can be seen in Figure 3.3 below. Every input to every node has a weight attached to it and this weight can essentially be seen as what the network "knows" or has "learnt". These weights are adjusted in the training process in order to obtain the desired output. Each of the nodes in the hidden and output layers contains a transfer function, which sums over all inputs and their respective weights in order to produce an output. An example of a transfer function is *tansig*, MATLAB's equivalent of the hyperbolic tangent function. The network's final output is compared with a given desired output. The difference between these values is known as the error and if this error is greater than a certain tolerance the network will be "trained" using the backpropagation learning algorithm. This process continues until either the desired output is attained or a certain number of iterations (epochs) is exceeded. If the desired output was attained the networks training is complete and the network can be tested on unseen data. A desired output is generally only obtained after substantial testing, using different network structures and transfer functions [Adriaans and Zantige 1997].

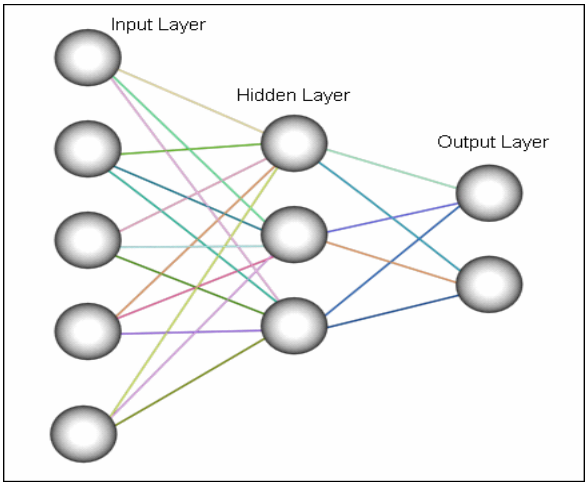


Figure 3.3: *Neural Network Model*

3.2.1.1. Scope

For the purpose of this paper we need not know the details of how the neural network learns. It suffices for us to know that once trained, it is possible to verify what the network knows by experiment.

We are interested in designing, implementing and testing neural networks using the MATLAB Neural Network Toolbox. Our attention will be focused on supervised feedforward networks, such as the one explained above.

3.2.1.2. Claims

Documentation of this toolbox is extensive and includes an extremely useful help facility which is available both as a graphical user interface and directly from the command prompt. Examples of the toolboxes uses are extensive and invaluable in making the most of all aspects of the Neural Network Toolbox as a data mining tool.

With regard to deciding whether or not to perform a given mine, the Neural Network Toolbox does not provide, nor does it claim to provide, any tools which might assist in this regard.

The toolbox does provide both pre- and post-processing functions for improving network training and assessing network performance. These functions, which were implemented in our second case study, greatly assist in data preparation. The reason for these functions being implemented in the second case study and not the first is that the range of the Isomerisation data is large compared with that of the Japanese businesses. Normalising data using the function *premnmx* brings the data into the range *[-1 1]*. It should be clear as to why this is not necessary for the first case study, as the data in Figure 3.1 already lies within a limited range.

A skeleton file for data preparation has been developed as a component of this study, as a means of verifying these claims. This file will assist greatly in the development of further neural network models and can be found in Appendix A.1.

In addition to normalising of data, the toolbox provides similar post-processing functionality (*postmnmx*), an alternative tool to normalise the mean and standard deviation of the data (as opposed to the data range) and a means for reducing the dimension of input vectors using a principal component analysis.

The focus of Neural Networks is the model building phase of the data mining process. As with all neural networks the aim is to create a trained network which is able to accept unseen input and provide an accurate prediction based on that input. In the case of the Japanese business data, the inputs to the network are the financial statistics and the output from the network is a prediction of whether a given business is either solvent or bankrupt. A number of additional case studies (using different databases), including the Isomerisation case study, have been carried out in order to determine the effectiveness of the neural network training functions provided by MATLAB. A skeleton script is provided for the creation of neural networks and is available in Appendix A.2. These skeleton files are intended for use together and have been verified based on separate case studies which we carried out in the course of this work. All case studies are available, both on the CD and the website (http://research.ict.ru.ac.za/g03t2052).

Lastly, the facilities available for the interpretation of the data mine, though not largely provided within the neural network toolbox itself, are provided within the basic MATLAB package. These facilities are largely graphics based (see results, section 4.1.2) and although these graphical representations themselves require interpretation, it is certainly an invaluable aid in the process. In addition to this, various numerical and statistical techniques can be used to get a better idea of how well a given neural network performs. In our cases we have relied heavily on the r-squared statistic to give us an indication of the goodness of fit provided by our neural network models. These options are provided in a skeleton script for simulation (interpretation), found in Appendix A.3.

3.2.1.3. Design of Implementation

It appears that the Neural Network Toolbox is relatively bullet proof in terms of the functionality which it claims to offer and this is to be expected of a piece of tried and tested proprietary software.

There are, however, a number of fundamental drawbacks to neural networks in general. The two most serious drawbacks are pertinent to this study. The first is that neural networks lack critically in terms of the decision and data preparation phases of the data mining process. There is absolutely no way of determining the effectiveness of developing a neural network prior to embarking on the process. In addition to this, the

pre-processing functions of the toolbox are limited; for example, there is no tool provided for the detection of outliers. Although it would be possible to implement algorithms to meet these needs, both of these areas are catered for more than adequately by the Fuzzy Clustering Toolbox. This tool will be discussed in detail shortly.

The second major draw back of neural networks is seen in their "black box" approach to problem solving. It is true that, whilst we may be able to verify that a given network does or does not do what it is supposed to do, we are never sure of why this is so or what the network actually "knows" [Adriaans and Zantige 1997]. This is largely because the knowledge of neural networks is represented by the weight given to every input, and even though we may be able to find out what these weights are, we cannot glean any meaning from these numbers.

Based on these draw backs, this toolbox has been implemented in conjunction with both the Fuzzy Clustering Toolbox and the Association Rule Miner (ARMADA). The results of this implementation are dealt with in Chapter 4.

3.2.2. Fuzzy Clustering and Data Analysis Toolbox

Unlike supervised learning methods, no dependant variable exists when performing clustering [Roiger and Geatz 2003]. The idea behind unsupervised clustering is the discovery of concept structures within data and the formulation of questions regarding the data which we would never have thought to ask with only the data set before us [Roiger and Geatz 2003]. In addition, clustering can be used:

1. in determining if useful information exists in the data set;
2. evaluating the likely performance of a supervised learner model;
3. determining a best set of inputs for a supervised learning model;
4. for detecting outliers and other data preparation related tasks.

[Roiger and Geatz 2003]

The creators of our clustering tool state that, "one may accept the view that a cluster is a group of objects that are more similar to one another than to members of other

clusters. The term ‘similarity’ should be understood as mathematical similarity, measured in some well defined sense” [Balasko et al. 2005]. This will become important shortly.

3.2.2.1. Scope

Our central focus on clustering will be as a pre-processing tool. We will be evaluating the Fuzzy Clustering and Data Analysis Toolbox based on how well it meets the four criteria mentioned above. That is, can this tool help us in determining if meaningful relationships, in the form of concepts, exist in the data? How comprehensive is the evaluation of the likely performance of a supervised learner model? And lastly, does our model enable us to detect outliers and perform other data preparation related tasks?

This is not a study of the algorithms used. It will be assumed that the means used for clustering and for verification of cluster strength are accurate.

3.2.2.2. Claims

The Fuzzy Clustering and Data Analysis tool uses both hard and fuzzy partitioning algorithms. A noticeable strength in the documenting of this toolbox is the fact that workable examples were made available for the testing of every facility offered. The major weakness of the documentation is that, despite its being extensive, the English is very poor and at times becomes almost impossible to understand. That said, the algorithms implemented were verified both by the examples provided and by being tested within our case studies. Hard partitioning using either the *kmeans* or *kmedoid* functions worked best when performing clustering on our data sets; this will be demonstrated in Chapter 4.

For the purposes of deciding whether a mine of the given data set is appropriate, the Fuzzy Clustering Toolbox is excellent. If clusters are found in the data and these clusters are interpreted as being meaningful, we can be fairly sure that concept structures do in fact exist in the data set being analysed. This find might lead us to the conclusion that a supervised learner model is likely to perform well and we would thus go ahead with construction of a neural network or decision tree. In addition, this tool gives insight into the need for further pre-processing, for example the spread of the cluster could indicate a need to normalise the data and instances that do not group naturally could be classified as outliers.

This tool is not a model building tool and so we would not expect it to provide any such facilities. However, despite the fact that no model is built there is still a need for the interpretation of the results provided. Once again, it must be emphasised that it is the interpretation provided by the human element that will determine the ultimate success or failure of the data mining process [Roiger and Geatz 2003]. The graphical presentation of the clusters and the validity measures provided are great assets in answering those questions posed regarding the likely performance of a supervised learner model, but they are only a guide. What we deduce from the clustering is what ultimately determines whether or not a supervised model is created. Lastly, many validation algorithms are available for use in checking the strength of a given clustering [Balasko et al. 2005]. This is yet another useful feature in the interpretation of results from any analysis.

It is apparent that the creators of the Fuzzy Clustering and Data Analysis Toolbox have a great deal of confidence in the tool which they have created. This is particularly evident in the provision of worked examples, which demonstrate the effectiveness of all the algorithms implemented. In conclusion, this tool meets the claims that it makes in most areas, and though there is room for improvement, particularly with regards documentation, it is likely that it will be used extensively in the decision making and data preparation phases of future data mining projects within MATLAB.

3.2.2.3. Design of Implementation

There is room to extend the Fuzzy Clustering and Data Analysis Toolbox, but we would first recommend that the documentation be updated in order to make it easier to understand the existing tool. The creators of this tool claim that it is easily extendable. A good place to start would be the extension of the validity measures offered, first in terms of the documentation of existing measures and secondly with regards the implementing of additional measures, such as Davies-Bouldin and silhouette indices. Implementing these extensions would certainly improve our ability to interpret the results of clustering and thus provide even more extensive capabilities for fulfilling the initial stages of the data mining process.

In conclusion this clustering tool is certainly the best available for MATLAB and provides the functionality necessary for a preliminary analysis of the data. Naturally, to

be considered an effective data mining tool, the results of clustering need to be extended to the implementation of a supervised learning model, such as a neural network or decision tree, where this is deemed appropriate. We have implemented this toolbox in synthesis with both the Association Rule Miner and the Neural Networks Toolbox, using the case studies previously laid out. The results of this process are detailed in Chapter 4.

3.2.3. Association Rule Miner and Deduction Analysis Tool

Rule mining is a powerful technique used to discover interesting associations between attributes contained in a database [Roiger and Geatz 2003]. Association rules can have one or several output attributes and an output attribute from one rule can be used as the input of another rule [Roiger and Geatz 2003]. Association rules are thus useful, both for obtaining an idea of what concept structures exist in the data (as with unsupervised clustering) and for model creation. In the second instance, the rules generated provide the underlying concepts used in the construction of decision trees and even neural networks (although this is carried out by the automated learning process).

ARMADA, or the Association Rule Miner and Deduction Analysis, is a tool created by James Malone [2003] and made available by The MathWorks on their central file exchange. ARMADA has been designed so as to allow for both straightforward mining in the form of rule extraction and as a means of analysing the knowledge extracted during a separate data mining session [Malone 2003].

As with our previous two tools, the Association Rule Miner (ARMADA) has been evaluated in detail. However, unlike the Fuzzy Clustering and Neural Network toolboxes, the results produced by the Association Rule Miner were not as useful as had initially been hoped. This is both due to the fact that it is difficult to obtain useful association rules from continuous numerical data and the fact that despite the claims made regarding the tool, it performed very poorly in many cases.

3.2.3.1. Scope

In the course of our study we were able to implement and test most of the aspects of the association rule mining tool, ARMADA. For the purposes of this thesis, our use of this tool has focussed on supervised data mining. As a consequence, we have needed to construct rules, rather than allowing the toolbox to do this arbitrarily, in an unsupervised

fashion. So, although ARMADA provides an option for mining all possible association rules we have only required those rules which have included the dependant variables from our case studies as the consequent, or output. This will be demonstrated in the implementation of this tool in Chapter 4 (section 4.1.3.)

Once again, this is not a study of the algorithms used. It will be assumed that the means used for generating association rules are accurate.

3.2.3.2. Claims

In the course of assessing this tool it became apparent that the documentation of the tool was lacking in a number of important ways. Not only were the worked examples provided in the documentation limited but the limitations of the tool with regard the mining of any given data set were never clearly stated. In addition to this, the claims regarding the usefulness of the tool were, on occasion, grossly exaggerated.

With regards the decision phase of the data mining process, the documentation acknowledges the necessity of this phase but does not claim that the tool provides direct answers to the question of whether or not to go ahead with a given mine. The documentation states that, "the value of discovering specific and accurate knowledge from…data mining, is in the unknown quantities of the data set being mined" [Malone 2003]. The advice given seems to be that for an initial exploration of the data we ought to mine the data as broadly as possible (using the minimum criteria for confidence and support of rules) and, if useful association rules are found to then increase the constraints on the criteria in order to obtain the results required. Once again, the common trend is to begin the data mining process in an unsupervised fashion and then, if this provides useful results, to concentrate our efforts on creating a supervised learning model.

The data preparation facilities offered by this tool are extremely limited. Few claims are made with regard this phase of the data mining process but neither are the limitations of the tool in this area highlighted. The main restriction is that all data has to be contained within a text (*.txt*) file, for which the formatting capabilities of MATLAB are very limited. In addition, very little error handling is provided with regard the format of the data and if a given mine fails to produce any rules, it is unclear whether the problem lies within the data being mined or elsewhere. There are many ways in which the restrictions

of ARMADA with regards simple data preparation facilities could be overcome and these will be discussed shortly (section 3.2.3.3.). In conclusion, even if a tool does not claim to provide a given facility, that does not excuse it from at the very least noting these limitations so as to make the use of the tool less problematic.

Many problems were also encountered with the claims which this tool makes regarding the model building phase of the data mining process. The tool claims to allow for the efficient extraction of association rules and for the thoroughness of the mine to be specified at the users discretion [Malone 2003]. Whilst the second of these claims is met by the tools offering the capability to mine in both supervised (using built goals) and unsupervised modes, as well as the capabilities to mine only a certain proportion of the data set, the first of these claims is rarely true.

The inefficiency of the tool in extracting rules is partly owing to the lack of data preparation facilities offered, as discussed, but is aggravated by the fact that the tool does not seem able to handle even moderately large data sets. This is yet another restriction not mentioned within the documentation.

One of the broadest claims of this tool is that, "the numerical data which ARMADA could be used to mine are virtually endless" [Malone 2003]. This is a gross exaggeration! Association rule mining is traditionally best carried out using categorical data, a fact that is agreed to by Roiger and Geatz [2003], Hand et al. [2001] and Adriaans and Zantinge [1997]. Whilst ARMADA was able to deal with the Japanese Business database it did not handle the Isomerisation database, which is significantly larger, at all well. The reason for this should be relatively clear, in that the dependant variable from our first case study is categorical but is continuous in the second. However, even with rounding of the dependant variable in order to make it categorical (this was done using the script *round2cat*, which is available in Appendix B) ARMADA performed poorly. These results will be presented in Chapter 5.

In the tools defence, it is true that, except where the criteria for mining are extremely narrow, the number of rules generated is likely to be large. Once again, however, it would have been most useful for this information to have been included in the tools documentation.

In conclusion of the capabilities of ARMADA with regards model building; it is fair to say that ARMADA is able to usefully mine small numerical data sets whose dependant variables are categorical in nature, such as our Japanese Business database, but that the tool is limited in dealing with large databases and has no capabilities whatsoever to usefully mine entirely numerical databases. This is in stark contrast to the claims of the tool.

The final phase of data mining, and the phase on which our use of the tool is largely focussed, is that of interpretation. The documentation agrees that this is, "arguably the most important phase of Data Mining," [Malone 2003]. The tool makes use of MATLAB's graphical capabilities in order to summarise the results of any given mine, a fact which is helpful in the interpretation of the results produced. These graphs summarise the support and confidence given to the rules mined. Support and confidence are variables associated with each rule mined, and they are an additional asset in the interpretation of results. Their importance becomes clear when one realises that a vast number of rules can be generated by a given mine and only a limited number of these will be informative as to the associations present within the data [Roiger and Geatz 2003]. These variables also indicate which criteria work best in the mining of the data, which is extremely useful when multiple mines are being carried out on a single data set, as is usually the case. Interpretation of results is one area in which ARMADA is relatively strong and the implementation hassle free.

3.2.3.3. Design of Implementation

It should by now be apparent that there is room for much improvement to this tool. The fact that the tool does not deal well with large data sets is a problem that would best be addressed by additional documentation pointing out these difficulties and providing advice as to how best to deal with such cases. One option would be to keep the criteria for mining high, whilst examining only a portion of the data set, capabilities for doing this already exist within the tool.

Other useful extensions would be the inclusion of a 'tab-delimited' option for mining text files as well as allowing for the reading and formatting of data from Excel spreadsheets. These simple pre-processing facilities are available within the standard

MATLAB package. Some simple extensions which were implemented in this study included the upgrading of this tool to work with MATLAB 7.0 (and higher versions), which involved the replacing of *break* with *return* statements. An additional script was also included for the categorising of numerical data by rounding. This script (*round2cat*) is found in Appendix B and the upgraded tool is available both on the website (http://research.ict.ru.ac.za/g03t2052) and on the accompanying CD.

With regards the synthesis of data mining tools ARMADA is useful in the decision making, model building and interpretation phases of the data mining process. We will not focus on the decision making or model building phases, since unsupervised clustering and neural networks are more powerful tools in these areas. We will use this tool largely as insight into the "knowledge" gained by the neural networks implemented in our case studies. This will be an excellent aid in the interpretation of results from the data mining process and in breaking down the "black box" nature of the neural networks knowledge.

3.2.4. Decision Tree

The last tool which has been examined in this study is the decision tree algorithm implemented by Rodney Woolf [2005] of the University of Southern Queensland. The algorithm implemented is equivalent to the C4.5 decision tree algorithm, which is already very popular in data mining.

Decision trees are extremely useful supervised learning tools in the field of data mining. Non-leaf nodes within the tree represent tests on one or more attributes, similar to the association rules previously discussed, and leaf nodes reflect decision outcomes [Roiger and Geatz 2003]. The training and testing process is very similar to that of neural networks, as detailed in section 3.2.1.

Woolf [2005] showed that models created using the C4.5 implementation in MATLAB were, in some cases, more powerful than those created using The Mathworks Neural Network Toolbox. Importantly, it was noted that the decision tree made up for the deficiencies of neural networks in the interpretation phase of the data mining process [Woolf 2005]. This, once again, indicates the importance of evaluating and synthesising the data mining tools available within MATLAB.

3.2.4.1. Scope

Unfortunately, when attempting to validate the work carried out by implementing the tool with our case studies, it was discovered that not all of the source code required to implement the decision tree had been provided. Due to this fact our handling of this tool will be brief and we will not be able to include this tool in our implemented synthesis, but only recommend the data mining phase in which it could be best used.

From the results provided by Woolf [2005], this appears to be an extremely useful tool, which is the central motivation for including a discussion of it in this study.

3.2.4.2. Implementation Recommendations

Many recommendations for the extension of this tool are provided in the work by Woolf [2005] and we will not therefore go into these details here. It is valuable to note the contrast between the documentation of the decision tree and that of the association rule miner. Despite not being able to implement the decision tree, the limitations of this work were made very clear. This is an additional reason to believe that implementing this tool would be a simple matter, provided the necessary script (*create_ref_point*) was made available.

Using a decision tree in conjunction with the other tools discussed so far would give us an almost complete implementation of the data mining process. An example of a possible synthesis of these tools, based on the data mining flowchart illustrated in Figure 2.1 is shown in Figure 3.4 below.

Naturally, the first phase of the process would be carried out by unsupervised clustering, in order to determine whether concept structures exist within the data set. This could be complimented by an initial, unsupervised, mine of association rules. If we decide to go ahead with a mine of the data, the results of the decision phase would be used in the carrying out of further data preparation, such as the removal of outliers and the assessment of the entropy (information gain) of the attributes. Once completed a neural network model would be constructed and tested; the reason for carrying this out before building a decision tree is because the effort required in constructing a neural network is minimal, particularly using the skeleton scripts provided in Appendix A of this study. A supervised mine of association rules could then be carried out, both as an aid to

the interpretation of the neural network and as a precursor to the building of the decision tree. Once the decision tree has been implemented, two models will exist for the classification of results from the database being mined. These results can be compared and contrasted in order to obtain the most from our interpretation phase of the data mining process, as was done by Woolf [2005]. This will complete our mine of the given database using tools implemented entirely in MATLAB.

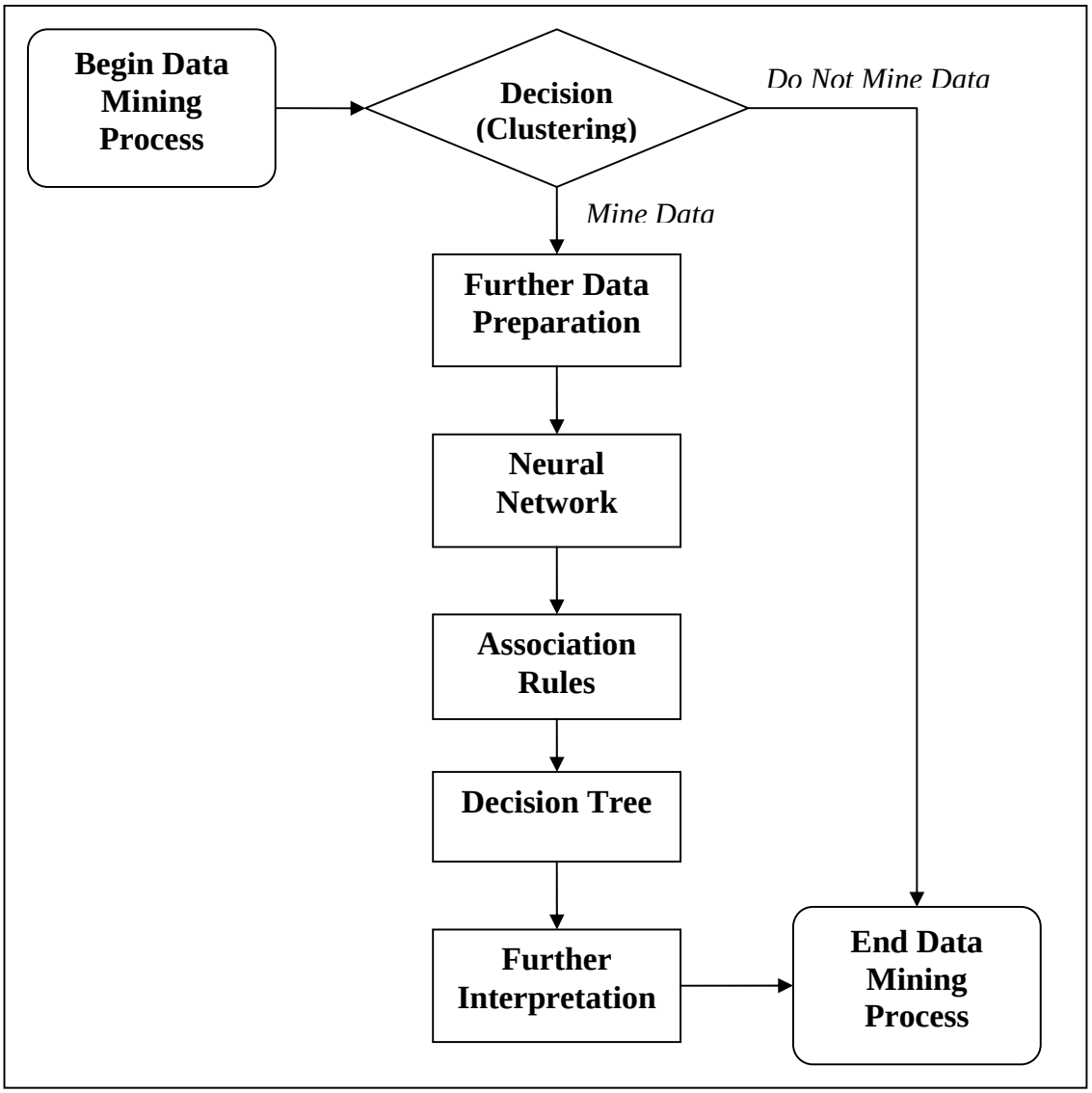


Figure 3.4: *Example of Complete Synthesis (based on Figure 2.1)*

3.3. Chapter Summary

In this chapter we have introduced our case studies and carried out the tool assessment phase of our methodology on four separate tools. In addition we have suggested the means by which we intend to synthesise our chosen data mining tools, thus using MATLAB as a stand alone tool for data mining. An example of the design of such a synthesis is given in Figure 3.4.

We now move on to the implementation of our process of synthesis. We will use the unsupervised Fuzzy Clustering Toolbox, Neural Network Toolbox and Association Rule Miner in this synthesis. Whereas the first two chapters of this thesis consisted largely of background information, Chapter 3 has played an extremely important part in the implementation process. It has provided us with a better understanding of the tools available within MATLAB and their potential uses, and has prepared us for the synthesis demonstrated in Chapter 4. Once we have obtained all the results from this process we will analyse them further in Chapter 5 and then finally conclude our work in Chapter 6.

Chapter 4: Implementation and Results

4.1. Synthesis

From the work carried out in Chapter 3 it is evident that the best possible combination of our three tools is an initial analysis using the Fuzzy Clustering and Data Analysis Toolbox [Balasko et al. 2005], followed by the creation of a neural network using The Mathworks Neural Network Toolbox, if this is deemed appropriate, and finally an analysis of results using ARMADA [Malone 2003]. Essentially then, the decision phase of our data mining process will be carried out entirely using unsupervised clustering. The data preparation phase will be shared by the fuzzy clustering tool and the data preparation script of our neural network, which we will use to normalise the data if necessary. The model building phase will be given entirely to the Neural Network Toolbox and the interpretation of results will be handled by the Association Rule Miner, complimented by the simulation script of our neural network. This process is depicted in Figure 4.1.

We will carry out this process on the Japanese Business database (Figure 3.1) and then compare these results with those obtained from the Isomerisation database (Figure 3.2) and an additional case study in Chapter 5.

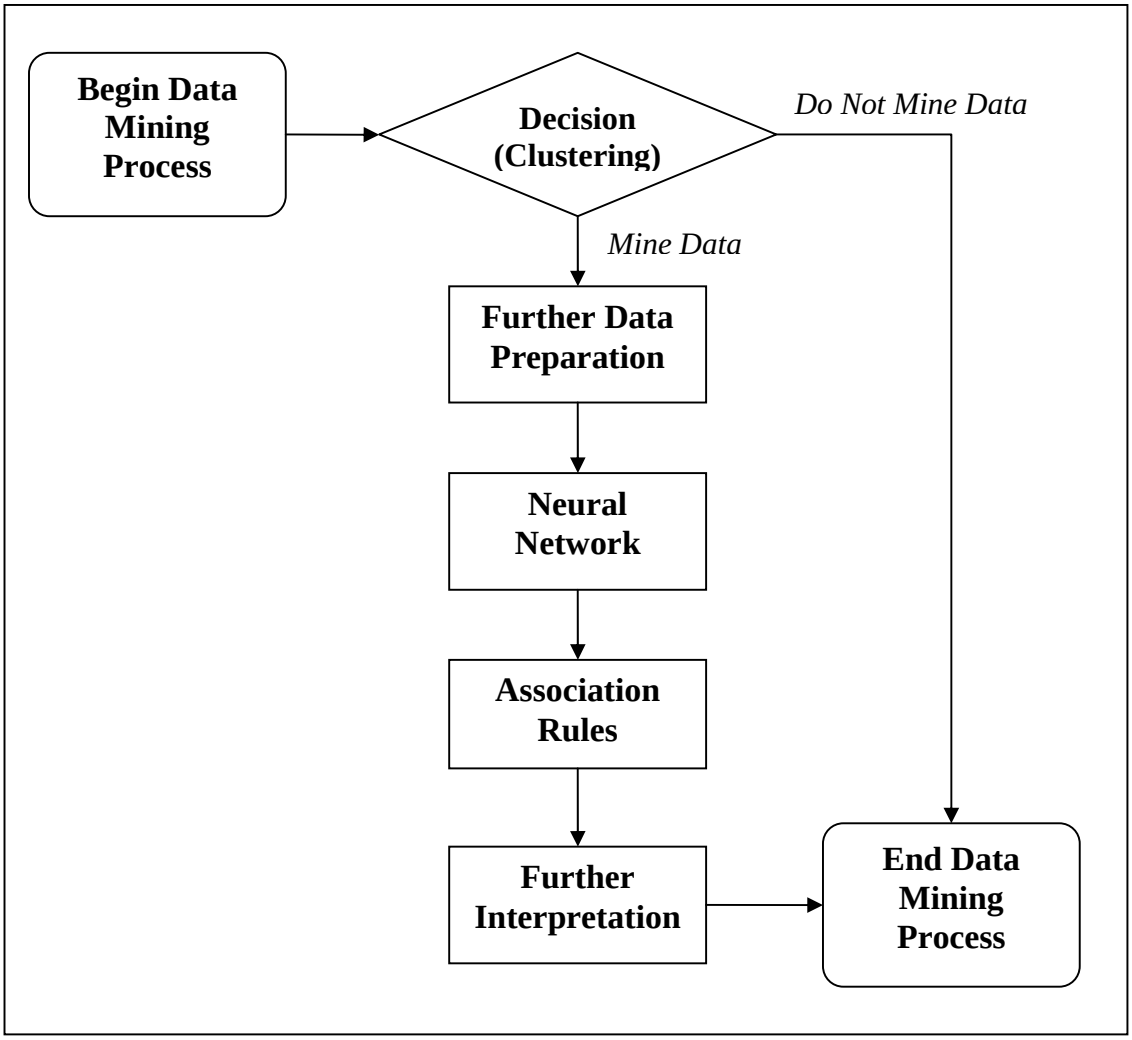


Figure 4.1: *Synthesis Implemented*
(based of Figure 2.1)

4.1.1. Fuzzy Clustering and Data Analysis Toolbox

The clustering tool was applied to the entire Japanese Business database and the results of that clustering are shown in Figure 4.2 below. The script used to obtain these results (*Kmeanscall*) from the toolbox is included in Appendix B. We will not discuss the validity measures given to this clustering (these were acceptable) due to the fact that these results can be concluded to be more than adequate by observation, as we will discuss.

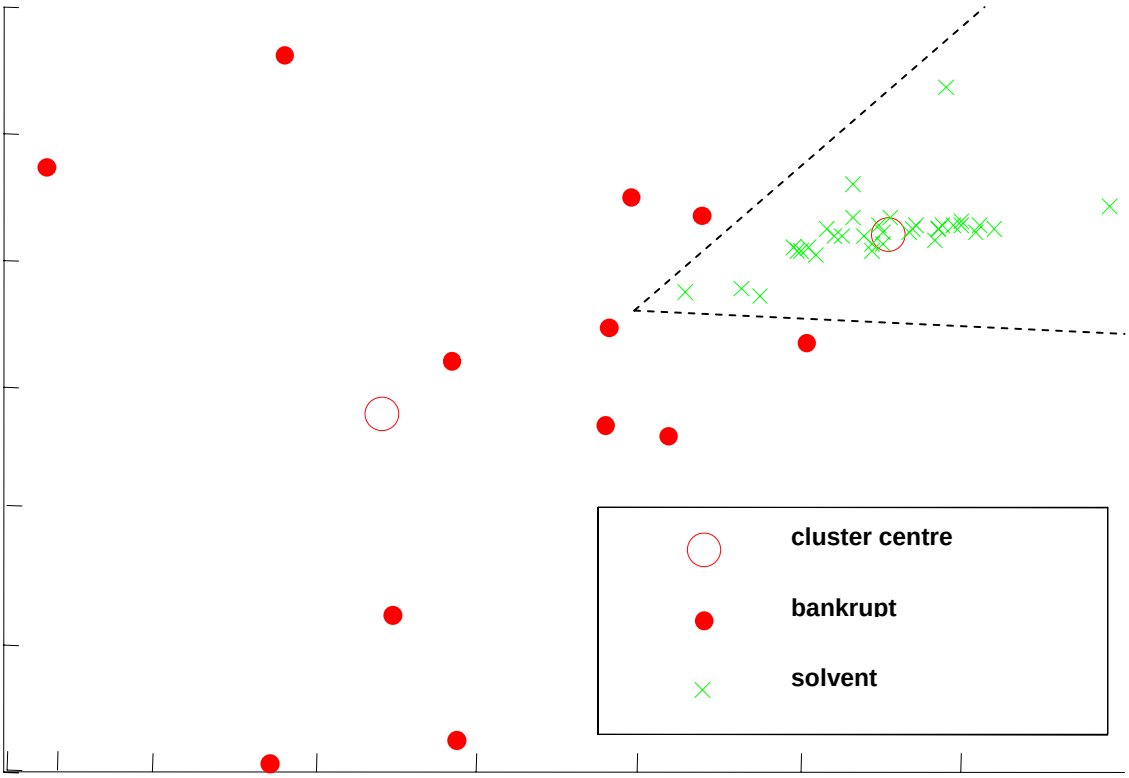


Figure 4.2: *The Results of KMeans Clustering on the Japanese Business Database with Additional Boundary Lines (---) Included*

The first thing to observe from the above result is the choice to partition the data into two clusters. This choice was a natural one, as we might expect the data set to split roughly into clusters of bankrupt and solvent businesses. As we can see, this is what has happened and whilst the above clustering undoubtedly overlaps in terms of its classification it is clear that there exist two well defined clusters, which can be separated by the boundary lines illustrated. These boundary lines were drawn in by hand and represent more or less what is looked for by a neural network when creating a model which can classify a given data set.

The boundary condition present thus indicates that concept structures are very much present in the data set and that a supervised learning model is likely to perform very well on this data set. The first phase of the data mining process can thus be completed as a result of the above clustering. Our decision is to go ahead with the creation of a neural network.

A further observation would be to take note of the spread of the given clusters. In this case, despite the spread of the cluster which we have defined as representing "*bankrupt*" businesses, the presence of the well defined boundary indicates that this is not likely to be a problem. In fact, those cases which might possibly be classified as outliers are likely to be those cases most easily classified by the network as they lie furthest from the decision boundary. The outliers we are looking to remove are those that might cause the network trouble in its learning process. For example, if there was a classification of a bankrupt business that fell within the decision boundary given, it would be advisable to remove that instance.

A last observation is that the spread of the clusters is limited, particularly in the case of those businesses which have been classified as "*solvent*". The reason for this is clear when we take into account the fact that the range of the data attributes is limited, confirming once again, that there is no need to normalise this data before constructing our neural network. In the case of the Japanese Business data we can therefore conclude, both from an examination of the data set and the given clusters, that data pre-processing is not necessary prior to a mine of this data.

4.1.2. Neural Network Toolbox

The next phase of our process is to create the neural network which will be used to classify different businesses as being either "*bankrupt*" or "*solvent*". The neural network was created using the skeleton scripts mentioned previously and available in Appendix A. Both the scripts pertaining to this specific case study and the neural network used are available on the website (http://research.ict.ru.ac.za/g03t2052) and the accompanying CD.

The network structure chosen used 8 neurons in the input layer (corresponding to the 8 input attributes), two hidden layers containing 14 and 12 neurons respectively and a single neuron in the output layer. The number of neurons used in the hidden layers was obtained by trial and error. A general rule with regard the choice of network architecture is that the complexity of the database (that is the number of attributes being fed into the network, which is 8 in this case) determines the complexity (number of neurons used) in the neural network being constructed. The output neuron gives the classification of either

"*bankrupt*" or "*solvent*". Lastly, the transfer function used in the hidden layers is the MATLAB *tansig* function, which is equivalent to the hyperbolic tangent, *tanh*. The reason for this choice is that the attribute values are both negative and positive (as seen in Figure 3.1); *tansig* is able to deal with both cases whereas a logarithmic sigmoid function, such as *logsig*, would have missed all negative values. For more details on the training of neural networks see section 3.2.1.

The results of the trained model are shown in Figure 4.3 below and the testing of this model is shown in Figure 4.4.

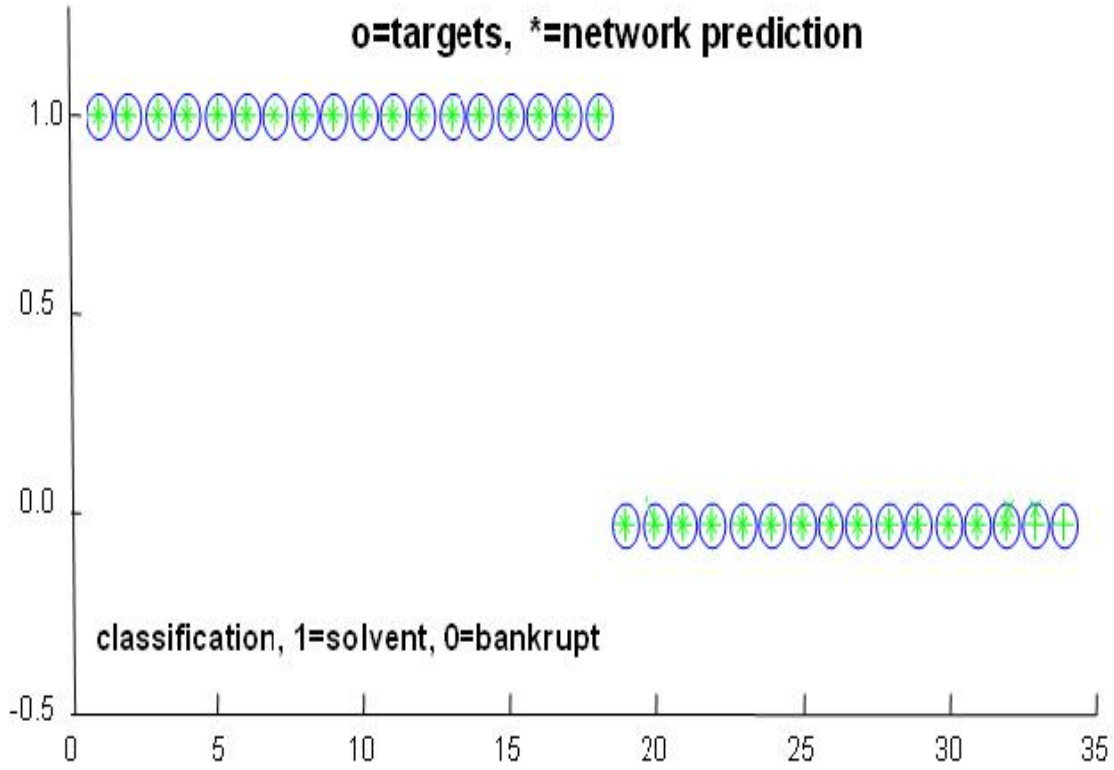


Figure 4.3: *Neural Network Results for the Japanese Business Database - Training (r^2 = 1.000000)*

Figure 4.3 above shows both network targets and their respective classifications. Each Japanese business in the training set (which consists 2/3 of the entire database) is represented by both a blue circle and a corresponding green star. The blue circles

represent the actual state of the given businesses, either the business was "*solvent*" (*1*) or "*bankrupt*" (*0*) as represented on the vertical axis; these are the target values which the network is attempting to predict. The green stars represent the networks predictions. It is clear that the minimum error of the network was reduced significantly, as all of the businesses in the training set were predicted correctly. For the sake of completeness it should be noted that the r-squared value for the training set was exactly *1.0* indicating a perfect fit.

These results are not particularly unusual; the real test comes in examining the networks ability to infer from what it "knows" in order to predict unseen cases. This is represented by the results of the test set, depicted in Figure 4.4.

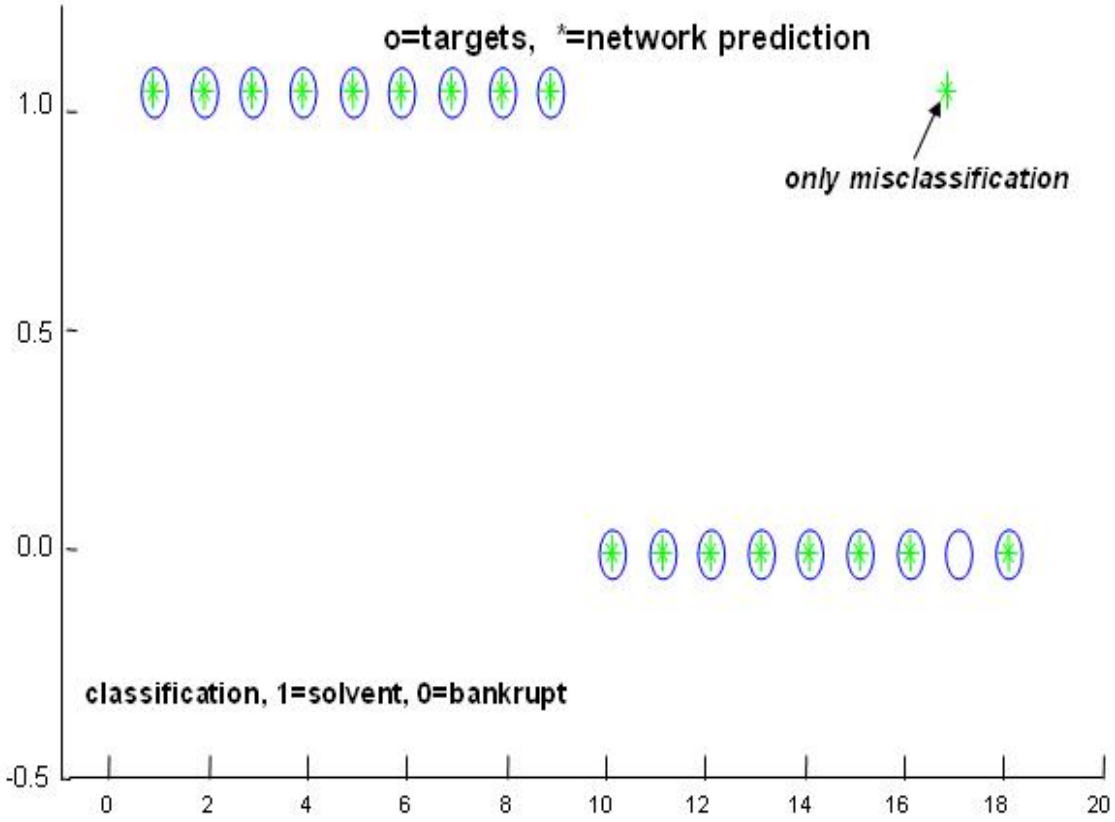


Figure 4.4: *Neural Network Results for the Japanese Business Database - Testing ($r^2 = 0.777778$)*

The results of testing are of greatest interest to us as they show the potential of the given model to accurately predict the financial position of a business based on the given

financial statistics. The remaining 1/3 of the database was used for the purposes of testing. No validation set was used as the database was too small to warrant reducing the training set, which needs to be sufficiently large in order for the model to be able to generalise well.

Figure 4.4 shows that one of the businesses that did in fact go bankrupt was predicted as being solvent. The r-squared value for the training data is *0.78*. This is an acceptable rate of error for such a model and we can thus conclude that we have been successful in creating a neural network that has mined the given data set and is able to classify unseen instances within the given domain with a high degree of confidence.

4.1.3. Association Rule Miner

The final phase of our process is to use ARMADA in the construction of association rules, in order to give us a better idea of what the neural network which we have constructed has "learnt". The central motivation for this is to aid us in the interpretation of the results which have been attained by our neural network.

As discussed in Chapter 3, this was the most frustrating part of the mining process, largely due to the tool's poor documentation. It took many separate mining attempts to obtain the following results, with many erroneous and unhelpful error messages having to be circumvented by trial and error. One example of such a difficulty was the discovery that ARMADA does not recognise *0* as a consequent or even as a numeric value in the mining of association rules. We were thus forced to change all the consequents for the bankrupt businesses to *-1* and then perform our mine on this data in order to obtain results which dealt with both bankrupt and solvent cases.

A screenshot of the criteria used in mining the Japanese Business database is provided in Figure 4.5. Figure 4.6 shows the goals which were built in order to perform the required supervised data mining session. The rules generated are summarised in Figure 4.7 and listed in Table 4.1.

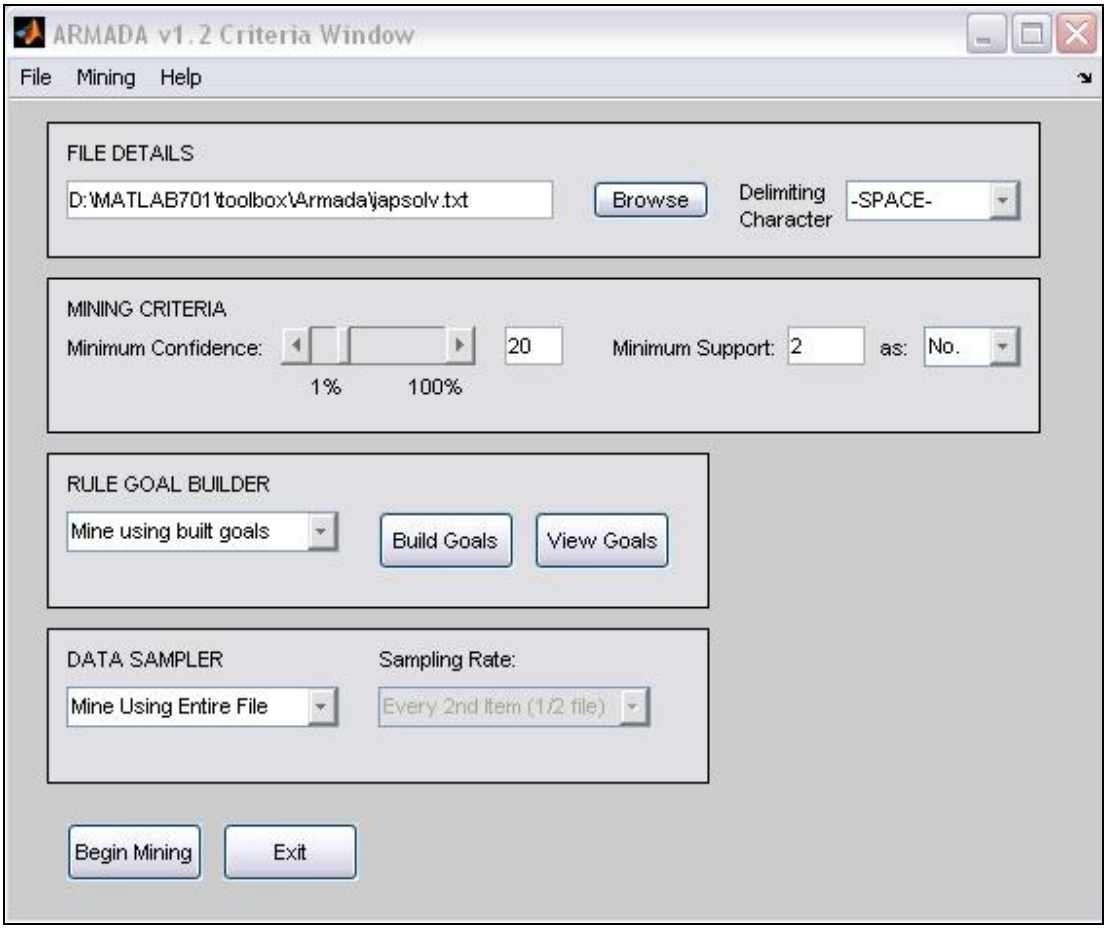


Figure 4.5: *Broad Rule Mining Criteria for the Japanese Business Database*

The broad criteria used in mining the Japanese Business database were as follows:

- Minimum Confidence = *20%* (% of times that LHS=>RHS is true)
- Minimum Support = *2* (no. of times the given rule appears)
- Mined Using Built Goals (required for supervised rule mining see Figure 4.6)
- Mined Using Entire File (database is relatively small)

Important to note is the 'minimum support' required as this has a greater effect on the number of rules generated than 'minimum confidence'. If 'minimum support' is set to *1* ARMADA cannot mine the entire file. If only *25%* of the file is mined with a support of *1*, the number of rules generated is well over *2000*. The last thing to note is that these

criteria are relatively low and will allow for the extraction of a reasonable number of rules.

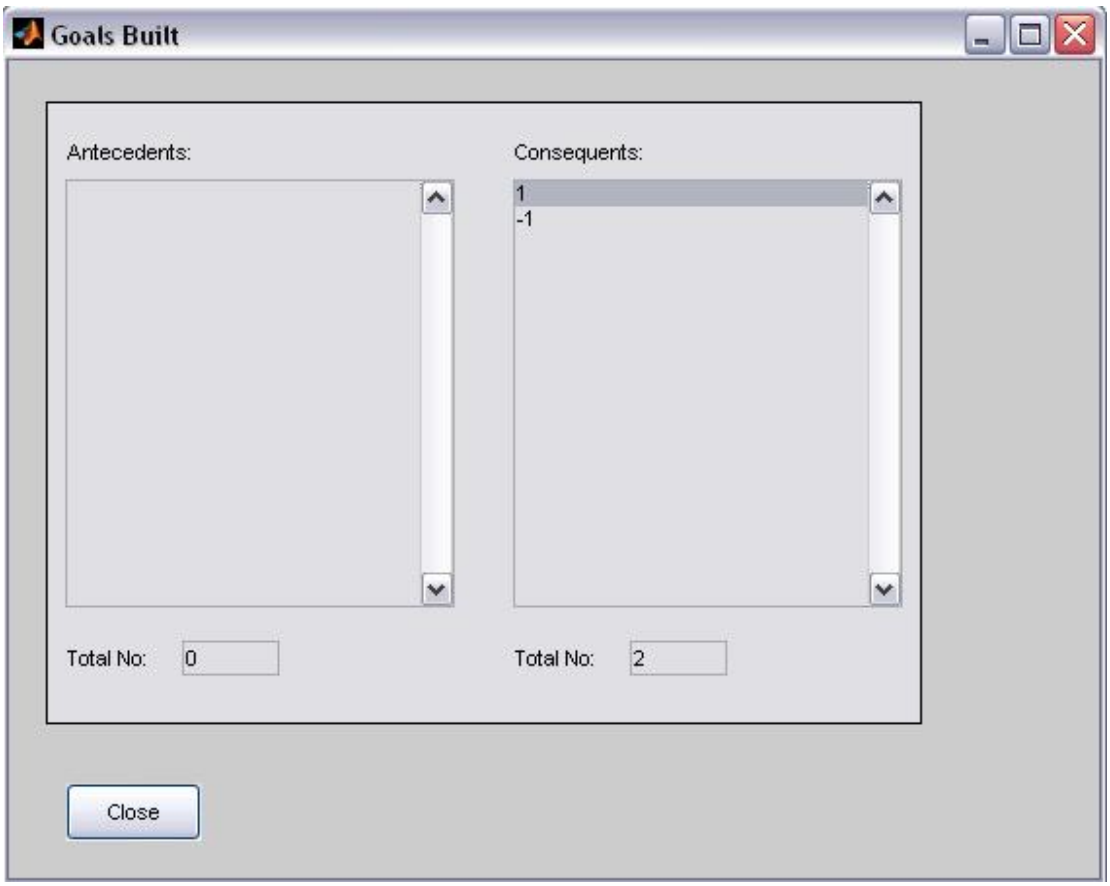


Figure 4.6: *Rule Mining Criteria for the Japanese Business Database - Goals Built (1=Solvent, -1=Bankrupt)*

It is important to note from the above goals that any entry within the database, other than those which have been specified as consequents, can be used as antecedents of any given rule. It should also be noted that *1* and *-1* are the only consequents which will be generated by the rule miner. This is seen most clearly in Table 4.1 below.

The results of the given mine are summarised in Figure 4.7 and listed in Table 4.1 using the *"Dump To Cmd Win"* button, seen in Figure 4.7.

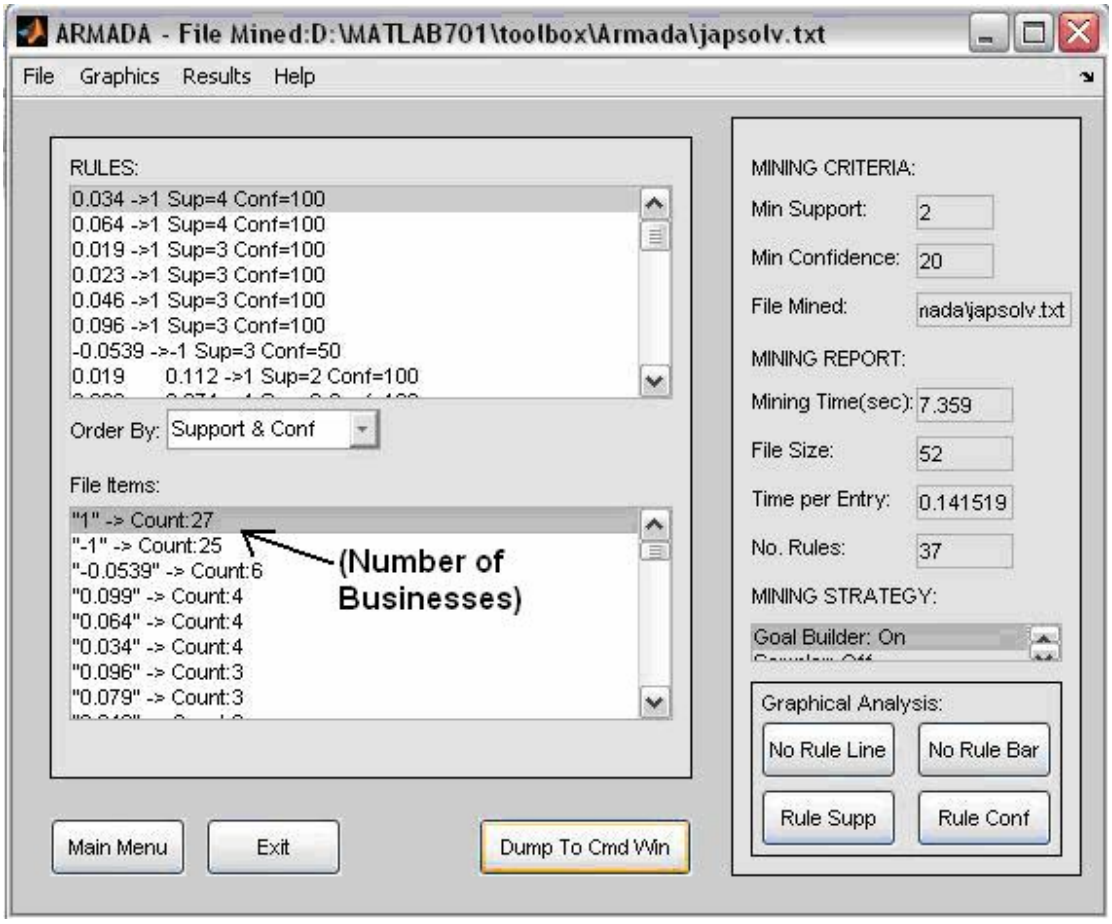


Figure 4.7: *Results of Mining the Japanese Business Database*

Rule	***Support***	***Confidence***
0.064 ->1	Sup=4	Conf=100
0.034 ->1	Sup=4	Conf=100
0.096 ->1	Sup=3	Conf=100
0.046 ->1	Sup=3	Conf=100
0.023 ->1	Sup=3	Conf=100
0.019 ->1	Sup=3	Conf=100
0.374 ->1	Sup=2	Conf=100
0.273 ->1	Sup=2	Conf=100
0.212 ->1	Sup=2	Conf=100
0.197 ->1	Sup=2	Conf=100
0.112 ->1	Sup=2	Conf=100

0.106	->1	Sup=2	Conf=100
0.089	->1	Sup=2	Conf=100
0.083	->1	Sup=2	Conf=100
0.07	->1	Sup=2	Conf=100
0.065	->1	Sup=2	Conf=100
0.049	->1	Sup=2	Conf=100
0.047	->1	Sup=2	Conf=100
0.045	->1	Sup=2	Conf=100
0.043	->1	Sup=2	Conf=100
0.041	->1	Sup=2	Conf=100
0.038	->1	Sup=2	Conf=100
0.027	->1	Sup=2	Conf=100
0.02	->1	Sup=2	Conf=100
0.013	->1	Sup=2	Conf=100
0.007	->1	Sup=2	Conf=100
0.003	->1	Sup=2	Conf=100
0.001	->1	Sup=2	Conf=100
-0.01	->1	Sup=2	Conf=100
0.079	->1	Sup=2	Conf=66.67
0.099	->1	Sup=2	Conf=50
0.065	0.07	->1 Sup=2	Conf=100
0.046	0.049	->1 Sup=2	Conf=100
0.023	0.374	->1 Sup=2	Conf=100
0.019	0.112	->1 Sup=2	Conf=100
-0.0539	-> -1	Sup=3	Conf=50
0.0785	-> -1	Sup=2	Conf=100

Table 4.1: *Association Rules for the Japanese Business Database (1=Solvent, -1=Bankrupt)*

The first rule in this rule set (*0.064 ->1 Sup=4 Conf=100*), can be interpreted as follows. The numerical attribute value, *0.064*, will imply that a given business is

solvent (rule output of *1*). This rule is supported by *4* instances and contradicted by none, leading to *100%* confidence in the accuracy of this rule. Stronger rules are those with two attributes producing a given outcome; for example, *0.065 0.07 ->1 Sup=2 Conf=100*.

An initial observation from these results is the fact that there are many more rules generated for solvent businesses (*1*) than for those that went bankrupt (*-1*). This would be a likely reason for the misclassification of one of the bankrupt businesses by our neural network as there are fewer rules from which it can "learn" the characteristics of a bankrupt business. Another reason for this misclassification may also be the fact that the network had more solvent businesses to "learn" from than bankrupt ones. Figure 4.7 summarises this information by giving the count of solvent and bankrupt businesses as being *27* and *25* respectively. A solution to this problem would be to obtain more data from Japanese businesses and to train a new network on this data. If this were possible it is likely that more association rules would be generated for classifying bankrupt businesses and not only that, but the neural network used for the purposes of classification would likely perform even better than it currently does.

Though the interpretation of these rules is difficult because of their numerical nature, it is clear that this information is likely to be exactly what a neural network is looking for in its process of learning. The strong relationships within the data are again seen in the fact that most of the rules have been generated with a confidence of *100%*, leading to less likelihood of confusion in the classification of our neural network. The strength of relationships in the data is made even more significant when we remember that our 'minimum confidence' was set at *20%*, but that the lowest confidence among the association rules generated is *50%*.

In conclusion, these rules represent generalisations which are what a neural network needs to "learn" in order to be able to solve a classification problem and particularly to classify new, unseen instances. If few such rules existed the best we could hope for would be that the neural network "memorise" the training set. As it is, this has not happened and we are able to conclude that we have been successful in creating a model which is able to generalise over new cases and which is likely to predict whether a business is either bankrupt or solvent with a very high level of accuracy.

4.2. Chapter Summary

In this chapter we have been able to show that it is possible to use MATLAB as a stand alone tool in the field of data mining, something which has rarely been done in the past. Not only this but we have also put confidence in the methodology previously constructed and can now extend this work both to further case studies of a similar nature and to entirely new syntheses, using different tools.

In Chapter 5 we will evaluate these results using two similar case studies and discuss briefly the potential for creating a data mining toolbox for MATLAB.

Chapter 5: Findings and Evaluation

Five case studies were carried out in the course of this project, including the Japanese Business and Isomerisation examples already introduced. All of these case studies were implemented similar to that detailed in Chapter 4 (see Figure 4.1). The results from these case studies, though not exactly the same as those for the Japanese Business' example, were in support of the work done here and it is felt that the best possible evaluation of the process which we have presented and of the potential of MATLAB in this field, is to compare and contrast these case studies. For the sake of space we will only deal with three of these case studies in this work; the additional case studies are available both on the website (http://research.ict.ru.ac.za/g03t2052) and on the accompanying CD.

This chapter can be viewed as the final analysis phase of the methodology outlined in Chapter 2 (see Figure 2.2) and before dealing with these results it is important to take into account the possibility of exaggerating the capabilities and success of our chosen tools. Hand et al. [2001] issues an extremely valuable warning: we must be careful not to exaggerate the likely outcomes of the data mining process. The way in which we approach the final analysis of MATLAB in this context, ought to be continually tempered by good sense and a realisation that we may not always be led to valuable results [Hand et al. 2001]. Further to this, the statement by Roiger and Geatz [2003] that, "it is the analysis of results provided by the human element that ultimately dictates the success or failure of a data mining project" is of vital importance.

If we keep these statements in mind, it will ensure that our analysis is scientific and that the results obtained from these syntheses are valuable and have meaning for those interested in the databases investigated.

We will conclude this chapter with a brief discussion on the feasibility of creating a data mining toolbox for MATLAB, and will then conclude our work in Chapter 6.

5.1. Comparison of Case Studies

We have already introduced two of the three case studies with which we will be dealing. The first is that for Japanese Businesses, the second is the Isomerisation database and the third is a much larger database, dealing with Breast Cancer [Mangasarian and Wolberg 1990]. This database is categorical in nature and has as its dependant variable a prediction of whether a given tumour is malignant (represented in the data by the value *4*) or benign (represented by *2*).

We aim to deal with these case studies briefly but nonetheless effectively; we will therefore not go into as much detail as in Chapter 4 but will highlight interesting results discovered in the mining of these data sets.

5.1.1. First Case Study

In the case of the Japanese Business database, we have a set of data, the attributes of which are continuous, and the dependant variable, categorical in nature. This database was moderately sized, not so small that no patterns could be found in the data but neither so large that the given tools had difficulty dealing with the data. These are the main reasons for having taken this database as the central case study of this work and as we have seen, data mining using MATLAB as a stand alone package proved to be extremely successful with this database.

The Fuzzy Clustering Toolbox had no problem in finding the necessary clustering of this data and this was, perhaps, the easiest stage of the synthesis process. The results of this proved to be positive, as the two clusters which we had expected to emerge were well defined and we were able to conclude that to go ahead with a supervised mine using neural networks was feasible.

Applying the Neural Network Toolbox, followed by ARMADA, once again proved to be very successful and it was concluded that the model produced, though imperfect, is certainly sufficient for future classifications of Japanese (or any other countries) businesses, given the necessary statistics.

These results show, despite the case study being very specific, that if provided with the right tools, a data mining project in MATLAB is extremely feasible and not at all difficult. It can also be concluded that both the broad methodology outlined and this

specific implementation (which was decided based on the tool evaluation of Chapter 3) are effective for the synthesis of data mining tools and provide a holistic approach to carrying out data mining in MATLAB. This will be validated and extended by the next two case studies.

5.1.2. Second Case Study

Our second database, which contains data pertaining to the Isomerisation of n-Pentane, is entirely continuous, including the dependant variable. This database consists of significantly fewer attributes than does our Japanese Business database but is also significantly larger, which had important implications for the results.

After a fair amount of experimentation with clustering on this data set, it was decided to use three clusters. This choice was supported by the numerical validation algorithms provided with this toolbox, as well as by the excellent cluster separation, depicted in Figure 5.1.

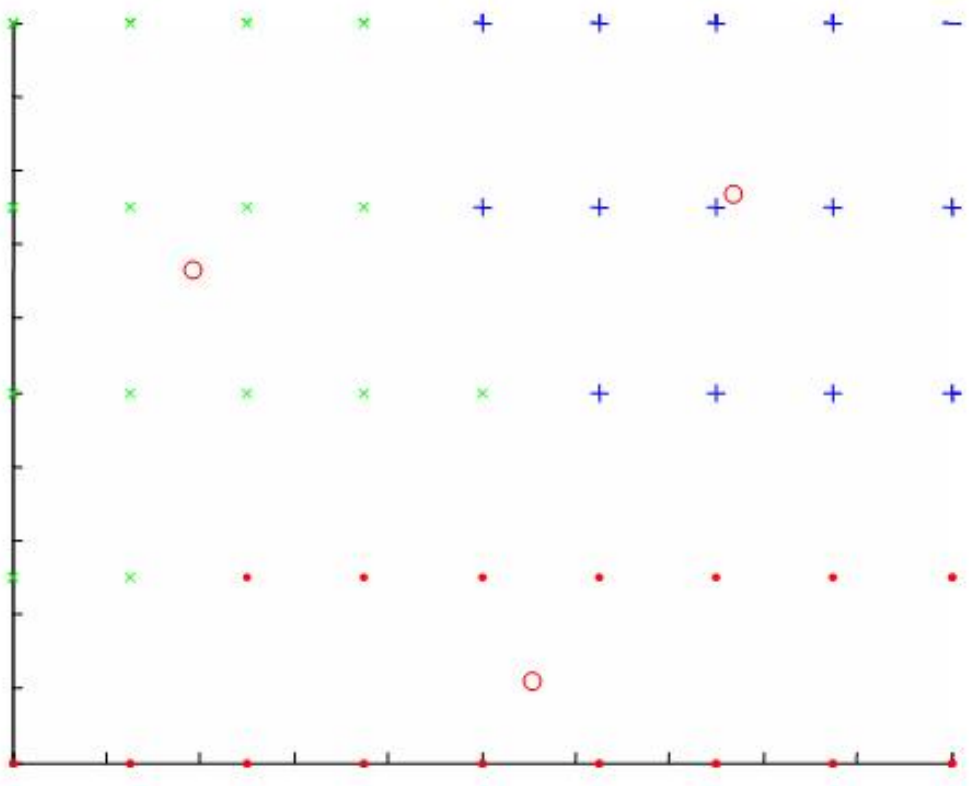


Figure 5.1: *The Results of KMeans Clustering on the Isomerisation Database*

This result once again led us to the implementation of a neural network and given the size of the database we were able, not only to test our model but also to validate it. The results of this were, once again, excellent and are depicted in Figures 5.2, 5.3 and 5.4, the training, test and validation data sets respectively. Interpretation follows.

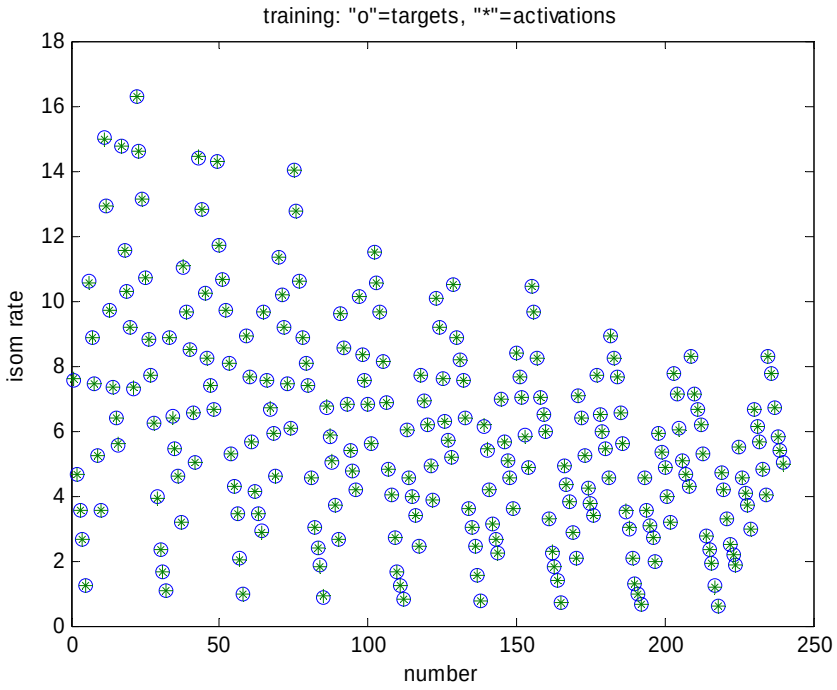


Figure 5.2: *Neural Network Results for the Isomerisation Database - Training (r^2 = 0.999997)*

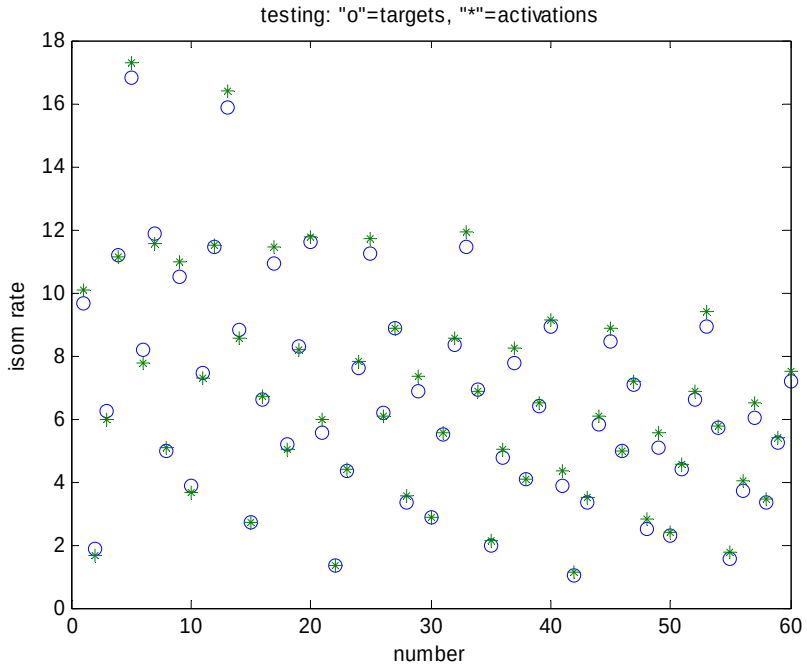


Figure 5.3: *Neural Network Results for the Isomerisation Database - Testing (r^2 = 0.993305)*

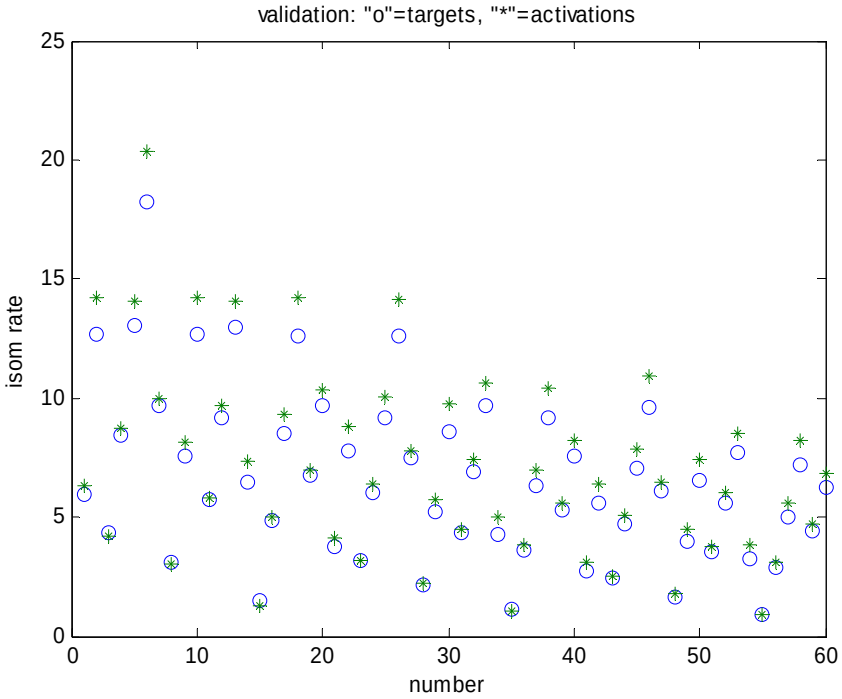


Figure 5.4: *Neural Network Results for the Isomerisation Database - Validation (r^2 = 0.962717)*

We will not deal with the basic interpretation of these results here (see Section 4.1.2) and details of the neural network used to obtain these results can be found in the scripts themselves, provided both on the website (http://research.ict.ru.ac.za) and on the accompanying CD. Our focus will be on highlighting the differences in this case study as well as on comparing the results obtained to those of the Japanese Business database.

The benefits of using a validation set are many; the central benefit is that it provides an entirely separate and often slightly differently formatted data set on which to validate both the results of training and of testing. It is common for the r-squared value to be slightly lower than those for the training and test sets and though this is seen in our case, the difference is minimal. The results of prediction for this numerical data are excellent and it is likely that, were new (unseen) data provided, this would be an excellent neural network model to use in the prediction of the rate of Isomerisation of n-Pentane.

Our methodology is thus further validated based on this slightly different case study. It is interesting to note that fewer attributes did not affect the performance of the neural network, or even the clustering. The fact that there were patterns in the data was enough to ensure a successful mine using MATLAB.

As a last note, as mentioned in Chapter 3, ARMADA is not able to mine fully numerical data sets, despite its claim to the contrary. We were able to extract some rules from this data by firstly rounding the value of the dependant variable (the simplest means of providing some categorisation) and then using a very small portion of the data. Even this, however, did not produce very useful results and it was thus decided to leave this phase out of the process.

Although this does impact significantly on the interpretation of the results, we were nevertheless able to discover certain limitations to ARMADA and thus reduce the amount of time needed to carry out data mining on similar case studies in future.

5.1.3. Third Case Study

In this final case study, the database contains attributes relevant to predicting whether a given tumour is either malignant or benign (details can be found in Appendix E). The distinguishing feature of this database is the fact that all attributes are categorical. The attribute values (of which there are *9*, excluding the ID) are integer values ranging from

1-10 and hold information relevant to the classification of the tumours into the two classes; benign (represented by a value of *2*) or malignant (represented by a value of *4*). It is clear that the dependant variable is also categorical in nature.

With this final database, we have covered most cases, having studied data sets where the databases are entirely numerical (Isomerisation), entirely categorical (Breast Cancer) and in the case of the Japanese Business database, having numerical attributes and a categorical dependant variable.

The process outlined (see Figure 4.1) worked excellently for this database. Firstly, in terms of clustering, we could once again tackle this phase of the process with ease, since we know to expect two potential clusters, that is, either a tumour is benign or malignant. Figure 5.5 shows the results of this clustering and it is once again clear that there exists a distinct separation between the two different clusters.

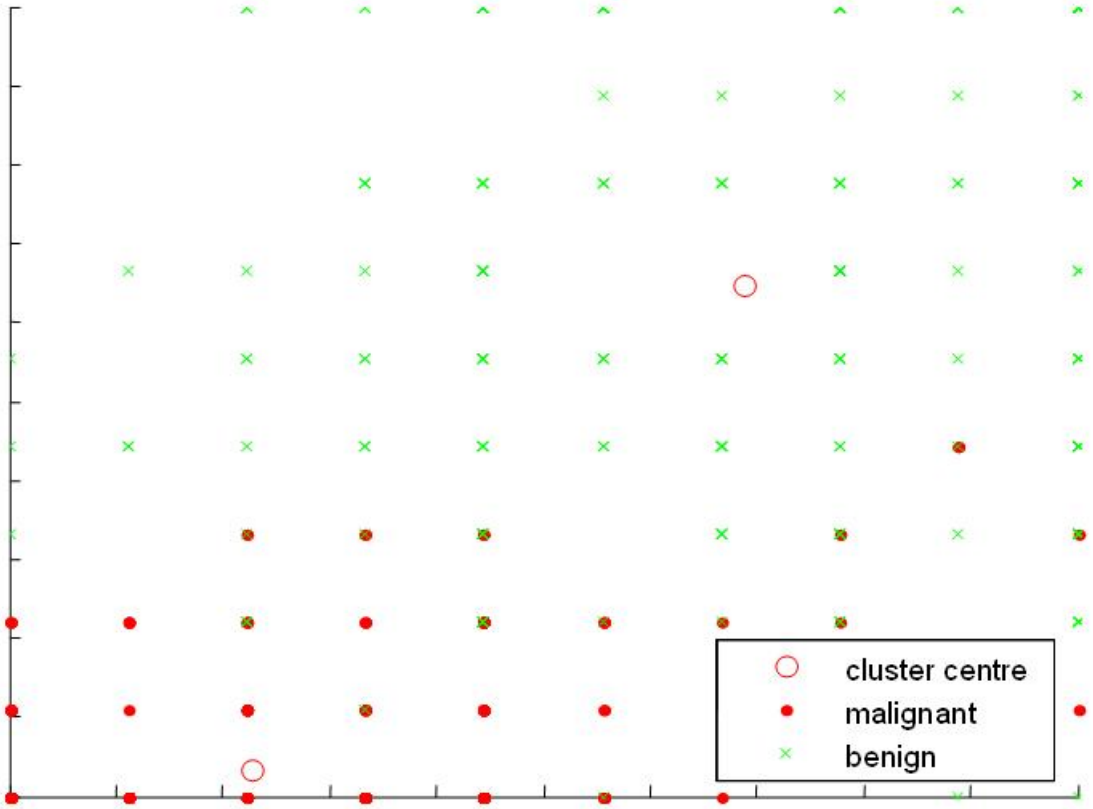


Figure 5.5: *The Results of KMeans Clustering on the Breast Cancer Database*

The little overlap which there is, is not a major concern, especially since the statistical validity measures given for this clustering are the best seen in any of the case studies so far examined.

Based on these results it was decided to mine the data. Because the database is so large (699 cases), we will not show these results here but simply give summary statistics for them; the results were similar to those for the Japanese Business neural network (Figure 4.3 and 4.4) and the reason for not showing them is that they were unavoidably squashed and thus difficult to read. In summary then:

- ¾ of the database was used for training
- r-squared statistic for the training set = 1.000000 (perfect, no wrong predictions)
- ¼ (that is 175 cases) of the data set was used for testing
- number of wrong predictions = 8 out of 175 (4.57%)
- r-squared statistic for the test set = 0.804633

It is true that for an area as important as classifying the malignancy of tumours, even a *5%* error rate is unacceptable; however, this result is more than satisfactory for our purposes. In terms of further interpretation, this result could be further improved by removing a number of the outliers (those cases which overlapped in the clustering). Detailed results are available both on the CD and the website (http://research.ict.ru.ac.za).

Additional interpretation can be gleaned from the rules mined from this data set using ARMADA. The results of this mine are shown in Figure 5.6 below.

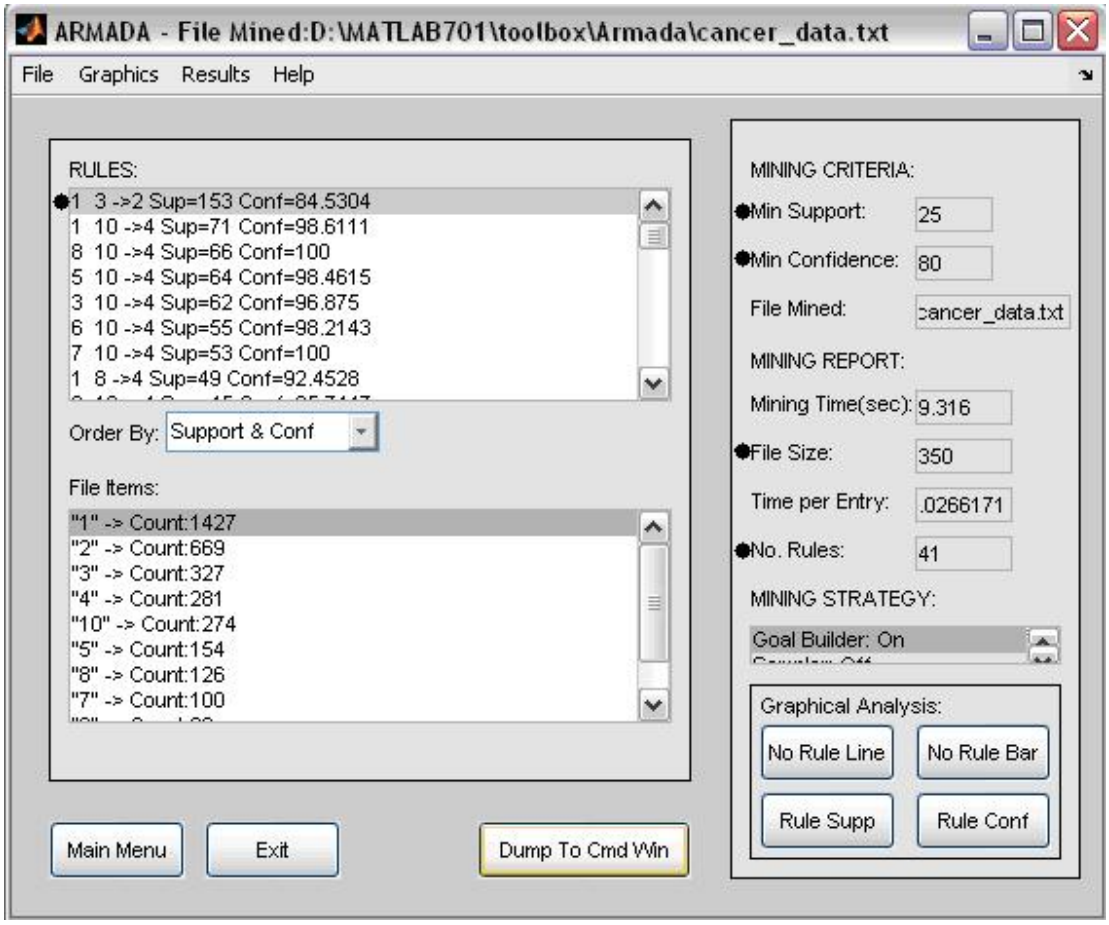


Figure 5.6: *Results of Mining the Breast Cancer Database*

We will give a brief summary of the interpretation gleaned from these results. Those details most important to this summary have been bulleted in the above figure.

Despite the fact that this mine was only carried out on half the database (this can be seen under *"File Size:" 350* in the above figure) and the criteria were relatively high (*Minimum Confidence of 80%* and *Support of 25*), a total of *41* rules were extracted. This indicates that there are many relationships within the data from which our neural network has been able to learn. Once again, these details are difficult to interpret without some idea of what the various categories indicate. For example, the first rule above can be interpreted as: attributes with the value of *1* and *3* lead to the classification of a tumour as benign; this is supported by *153* separate cases with a confidence of *84.5%* (that is *15.5%* of cases were contradictory). However, without some idea of what the values of *1* and *3* represent, it is difficult to know exactly how useful this rule is. It is clear from the neural

network model created that there must exist many meaningful relationships in the data and were the results found here to be used by a cancer expert, it is likely that greater meaning would be gleaned from the above result. The same holds true for all of the case studies used.

As a last note it is interesting that despite not having all of the knowledge of the given data, it is nevertheless possible to create a model useful for classifying the instances within the data. This fact demonstrates the power of the data mining process and particularly of this study. As the means for performing data mining in any given domain become more apparent, it is easier to divide research tasks. That is, whilst the expert in the particular domain cannot give the computer scientist all of his domain knowledge, the computer scientist, using tools such as MATLAB, is nevertheless able to carry out an extremely important part of the field expert's research, as we have demonstrated in the above case studies.

5.2. Toolbox Feasibility

From the above work it is clear that MATLAB is a powerful and versatile tool, able to carry out all stages of the data mining process. With the syntheses carried out here and the potential for further work in this area (as discussed in section 6.2. below) the creation of a data mining toolbox for MATLAB is a project with great potential.

The creation of such a toolbox is the logical result of a complete synthesis of data mining tools in MATLAB and whilst this may still be a way off, this work has shown both the necessity for such a toolbox and the feasibility of creating it. Even if this toolbox was only created on a small scale, such as a tool incorporating the use of clustering, neural networks and rule mining, the use of MATLAB as a stand alone tool in this field would increase dramatically.

In conclusion, it is imperative that such a project be approached with care following the steps outlined here in both synthesising and extending the existing tools so as to create a tool that will further improve the potential of MATLAB in this field.

5.3. Chapter Summary

In this chapter we have all but completed our investigation of data mining in MATLAB. The process of synthesis as a whole has been validated and certain limitations of its application seen with regard specific data sets. Note once again that all the work carried out in the course of this project, that is case studies and the open source data mining tools, are available on the CD and the project website (http://research.ict.ru.ac.za/g03t2052). A user manual is provided in Appendix C with detailed instructions for obtaining the results obtained for the Japanese Business case study (the other case studies will be very similar).

This chapter was concluded by emphasising the potential of MATLAB as a stand alone data mining tool and the recommendation that a data mining toolbox for MATLAB be created as an extension of this work and in order to obtain a completely holistic view of the data mining potential of MATLAB.

We now proceed to the conclusion of our study.

Chapter 6: Conclusion and Possible Extensions

6.1. Findings and Conclusion

Our central aim in this work was to not only provide an analysis and synthesis of data mining tools in MATLAB, but also a methodology which can be used in continuing this work into the future and possibly extending it to the creation of a data mining toolbox. We have been successful in creating such a methodology (see Figure 6.1 below for a summary) and we have validated these findings by evaluating and synthesising three MATLAB data mining tools. A number of important findings and achievements of this study are summarised below:

- Outlined and refined the phases of the data mining process (Figure 6.1 below)
- Established a methodology for synthesising MATLAB data mining tools (Figure 2.2) and confirmed its usefulness by experiment
- Provided neural network skeleton scripts, simplifying neural network creation (Appendix A)
- Confirmed neural network limitations in terms of:
    - Pre- and post-processing facilities
    - "Black Box" nature of results
- Verified the claims of the Fuzzy Clustering and Data Analysis Tool using:
    - Examples provided
    - Separate case studies
- Highlighted the poor English used in this (Fuzzy Clustering) tool's documentation
- Highlighted the many limitations of ARMADA:
    - Very difficult to obtain results from entirely numerical data sets
    - Handling of large data sets is poor
    - Data preparation and data handling facilities very limited
    - Need for the documentation to be explicit about tools limitations

- Wrote a script (*round2cat* available in Appendix B) for categorising numerical data to enable easier rule extraction
- Upgraded the tool to work with MATLAB 7.0 and above (mostly involved replacing *break* with *return* statements)
- Confirmed that multiple mines, using minimal restrictions at first and increasing these to obtain meaningful results, is effective within ARMADA
- Noted that the documentation of the decision tree was incomplete (missing script *create_ref_point*) and the tool cannot be implemented
- Demonstrated the use of MATLAB as a stand alone data mining tool using a number of case studies
- The creation of a data mining toolbox for MATLAB is a project with great potential

In conclusion we have been able to see the data mining capabilities of MATLAB in a far more holistic light than has been available previously. The process of synthesis outlined will enable MATLAB to be used far more extensively in this field in future, particularly as this process is extended to other tools and case studies. Possibilities for extension to this work are outlined below.

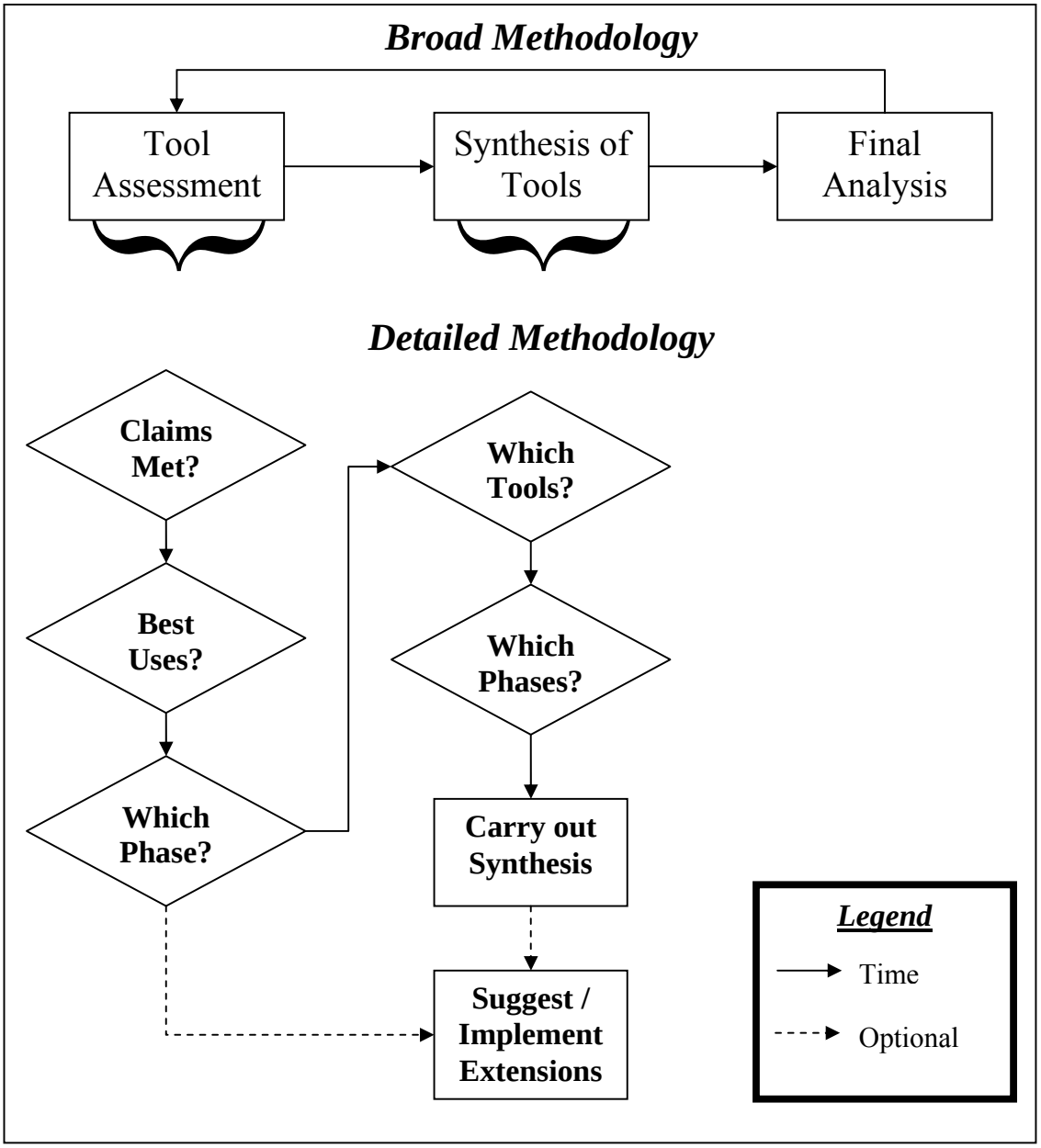


Figure 6.1: *Broad and Detailed Methodology for the Synthesis of Data Mining Tools in MATLAB*

6.2. Possible Extensions

The possibility for extension to this work ranges from the extension of individual tools to the creation of a data mining toolbox. These possibilities are summarised below:

- More generic data preparation facilities are needed for MATLAB
- Rewrite the documentation for the Fuzzy Clustering and Data Analysis Toolbox, correcting the English and extending it where necessary
- Extend ARMADA:
    - Provide a tab-delimited mining option
    - Fix the bug with loading of saved results
    - Provide more meaningful error messages (will require a lot of debugging of code)
    - Allow for mining of data held in Excel spreadsheets
    - Update the documentation, including the limitations of the tool
- Extend this work to the synthesis of different data mining tools:
    - Self Organising Map (SOM) Toolbox [Vesanto et al. 2000]
    - Bayesian Network Toolbox [Murphy 2005]
    - Further MATLAB Options
        - The Mathworks Statistical Toolbox
- Creation of a data mining toolbox

The ultimate goal of the synthesis of data mining tools in MATLAB would be the creation of a toolbox dedicated to the data mining process.

References

Adriaans, P. and Zantige, D., *Data Mining*, Addison Wesley, Harlow England, 1997, pp 39—42,69—81,91,117,127.

Balasko, B., Abonyi, J. and Feil, B., *Fuzzy Clustering and Data Analysis Toolbox for Use with Matlab*, University of Veszprem, Veszprem Hungary, 2005, Accessed: 11 May 2006, <http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=7473& objectType=file>.

Burton, M., *AM 2.2 Mathematical Programming*, Rhodes University, Grahamstown South Africa, 2006, pp 1—2.

Burton, M., *M 4.4 Neural Networks*, Rhodes University, South Africa, 2006, pp 102—111.

Dwinnell, W., *Modeling Methodology 5: Mathematical Programming Languages*, 1998, Accessed: 11 May 2006, <http://will.dwinnell.com/will/willTechnicalPublications.html>.

Hand, D., Mannila, H. and Smyth, P., *Principles of Data Mining*, MIT Press, Cambridge, Massachusetts, 2001, pp 1—24.

KDnuggets, *KDnuggets Past Polls*, 2006, Accessed: 28 September 2006, <http://www.kdnuggets.com/polls/>.

Malone, J., *ARMADA Association Rule Miner and Deduction Analysis*, 2003, Accessed: 11 May 2006, <http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=3016& objectType=file>.

Mangasarian, O.L., and Wolberg, W.H., *Cancer diagnosis via linear programming*, Siam News, Volume 23, Number 5, September 1990, pp 1,18

Murphy, K., *Bayes Net Toolbox for MATLAB*, 2005, Accessed: 10 May 2006, <http://bnt.sourceforge.net>.

Paola, B. et al., *Tool Selection Methodology in Data Mining*, 2006, Accessed: 11 May 2006, <http://www.itba.edu.ar/capis/webcapis/RGMITBA/comunicacionesrgm/JIISIC_-2006-Tool-Selection-Methodology-in-Data-Mining.pdf>.

Pyle, D., *Data Preparation for Data Mining,* Morgan Kaufman, San Francisco, California, 1999, pp 118.

Roiger, R.J. and Geatz, M. W., *Data Mining: A Tutorial Based Primer*, Addison Wesley, USA, 2003, pp 7—27,34—41.

Simonoff, J. S., *Analysing Categorical Data*, Springer-Verlag, New York, 2003, Accessed: 10 October 2006 <http://lib.stat.cmu.edu/datasets>.

Vesanto, J., Himberg, J., Alhoniemi, E. and Parhankangas, J., *SOM Toolbox for MATLAB 5*, Helsinki University of Technology, Helsinki Finland, 2000, Accessed: 13 September 2006, <http://www.cis.hut.fi/projects/somtoolbox/package/papers/techrep.pdf>.

Vlachos, M., *A Practical Time-Series Tutorial with MATLAB*, Hawthorne, NY, 2005, pp 7—12, Accessed: 10 May 2006 <http://www.cs.ucr.edu/~mvlachos/PKDD05/PKDD05_Handout.pdf>.

Woolf, R. J., *Data Mining Using MATLAB*, University of Southern Queensland, Queensland Australia, 2005, pp 1—3,10—12,59, Accessed: 10 May 2006 <http://eprints.usq.edu.au/archive/ 00000058/01/dissert.pdf>.

Appendix A: Neural Network Skeleton Scripts

The below skeleton scripts are designed both to give an overview of the means used in creating the neural networks implemented in this project and to allow for the efficient development of further neural networks for anyone wishing to extend this work. The major difference between these skeleton scripts and the ones used to produce and test the networks for the case studies carried out is that for each case study, *[project]* would have been replaced with the project name. If we wished to create the network for Japanese Business Solvency for example, *[project]* could be replaced with *japsolv*, so that *[project]_data* becomes *japsolv_data* and similarly for the variables within the scripts themselves.

A.1. Data Preparation Script

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%replace [project] with the name of the
%%current project
%[project]_data
%prepares [project] data for simulation
%%replace [author] with your name
%[author]
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear

%load data
%%use ‘xlsread’ for Excel spreadsheets and ‘load’ for other formats
%%such as '.mat' and '.txt'
%%help on xlsread can be obtained by typing "help xlsread" at the
%%command prompt
P=xlsread('[project].xls',1,'{data range}')';
T=xlsread('[project].xls',1,'{data range}')';

%%further pre-processing of patterns and targets could be done here

%partition the data into the training, test {and validation} sets
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
lp=length(P);

%test
%%sample frequency {eg. 6 would take 1/6 of the data set}
sf=6;

%test index
ti=[1:sf:lp];
%extract columns
```

```
ptest=P(:,ti);
ttest=T(:,ti);

%validation
%%OPTIONAL - delete up to 'train' if you don't want a validation
%%set
%%sample frequency (can differ)
sf=6;

%validation index
%%the start indices of the test and validation sets must differ
vi=[3:sf:lp];
%extract columns
pval=P(:,vi);
tval=T(:,vi);

%train
%training index (the rest of the data set)
%%if there is not validation set remove 'vi' below
tri=setdiff([1:lp],[ti vi]);
%extract columns
ptrain=P(:,tri);
ttrain=T(:,tri);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%this step is optional and depends on the nature
%%of the data; if the range of the data is large
%%normalising is a good idea
%preprocess the training data to normalise it
%pn=normalised P, tn=normalised T, -1<=pn,tn<=+1
%mp=minp, Mp=maxp, mt=mint, Mt=maxt
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[pn mp Mp tn mt Mt]=premnmx(ptrain,ttrain);

save [project]data.mat
```

A.2. Neural Network Model Script

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%replace [project] with the name of the
%%current project
%[project]_net
%trains a network to predict/simulate…
%%replace [author] with your name
%[author]
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear

load [project]data.mat

%si = number of neurons in layer i (m layers)
%%m should be changed to the number of the mth layer eg. s3
%%it has been mathematically proven that any classification problem
```

```
%%can be solved using at most 3 layers
%%a lot of experimentation here is advisable
s1=9;
s2=7;

sm=1;       %sm = number of outputs from the network

%transfer functions
%%(eg. hardlim, hardlims, purelin, satlin, satlins, logsig, tansig,
%%poslin)
%%if negative data is being analysed then ‘tansig’ or a symmetric
%%linear transfer function should be used in at least one layer
%%remember that data normalised using 'premnmx' lies between -1 and 1
%%a lot of experimentation here is advisable
transfer1='tansig';
transfer2='logsig';

transferm='purelin';    %%always use ‘purelin’ in the last layer

transfer={transfer1,transfer2,...,transferm};

%training method
%%(eg. traingd, traingdm, traingdx, trainrp, traincgf, traincgp,
%%traincgb, trainscg, trainbfg, trainoss, trainlm, trainbr)
%%’trainlm’ is default
%%generally ‘trainlm’ works best, the only drawback is that it uses a
%%lot of memory.
%%
%%Trainscg is another good option.
%%
%%’trainbr’ will use ‘trainlm’ but with the added advantage of Bayesian
%%Regularisation, which prevents overfitting of data. This leads to an
%%improved ability to generalise by the network.
%%
%%experimentation here is advisable
tm='trainscg';

%create the net
%%for non-normalised data used ‘minmax(P)’ in place of ‘[mp MP]’
%%if you wish to use ‘trainlm’ as the training method 'tm' can be
%%removed
[project]net=newff([mp Mp], [s1 s2 ... sm], transfer, tm);

%set the training parameters
%%experiment a bit with these parameters
[project]net.TrainParam.epochs=1000;
[project]net.TrainParam.goal=.0000005;

%train the net
%%for non-normalised data use ptrain,ttrain in place of pn,tn
[project]net =init([project]net);
[[project]net,tr]=train([project]net,pn,tn);

%persistance of trained network
save [project]net [project]net
```

A.3. Verification by Simulation Script

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%replace [project] with the name
%%of the current project
%[project]_sim
%simulates and tests the network
%%replace [author] with your name
%[author]
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear

load [project]data
load [project]net

%simulate on the training data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%simulate on the normalised training data
%%if data was not normalised, instead use:
%%atrain=sim([project]net,ptrain)
an=sim([project]net,pn);

%postprocess the targets from the training data
%%this isn't necessary if the data was not normalised
a=postmnmx(an,mt,Mt);
atrain=a;

%simulate in the test data
%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%this step is optional and depends on the nature
%%of the data if the range is large normalising
%%is a good idea
%preprocess the test data to normalise it
%pn=normalised p, tn=normalised t
%mp=minp, Mp=maxp, mt=mint, Mt=maxt
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[pn mp Mp tn mt Mt]=premnmx(ptest,ttest);

%simulate with the normalised test data
%%if data was not normalised, instead use:
%%atest=sim([project]net,ptest)
an=sim([project]net,pn);

%postprocess the test data
%%this isn't necessary if the data was not normalised
a=postmnmx(an,mt,Mt);
atest=a;

%simulate in the validation data
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%this step is optional and depends on the nature
%%of the data; if the range is large normalising
%%is a good idea
%preprocess the validation data to normalise it
%pn=normalised p, tn=normalised t
%mp=minp, Mp=maxp, mt=mint, Mt=maxt
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[pn mp Mp tn mt Mt]=premnmx(pval,tval);

%simulate with the normalised validation data
%%NB there may not always be a validation set
%%if data was not normalised, instead use:
%%aval=sim([project]net,pval)
an=sim([project]net,pn);

%postprocess the validation data
%%this isn't necessary if the data was not normalised
a=postmnmx(an,mt,Mt);
aval=a;

%%further post-processing could be done here

%plotting and analysis, 'o' denotes targets and '*' activations
close all

lPtrain=length(ptrain);
lPtest=length(ptest);
lPval=length(pval);

%training
plot([1:lPtrain],ttrain,'o',[1:lPtrain],atrain,'*')
title('training: "o"=targets, "*"=activations')
xlabel('...')
ylabel('...')

%testing
figure
plot([1:lPtest],ttest,'o',[1:lPtest],atest,'*')
title('testing: "o"=targets, "*"=activations')
xlabel('...')
ylabel('...')

%validation
%%there may not always be a validation set
figure
plot([1:lPval],tval,'o',[1:lPval],aval,'*')
title('validation: "o"=targets, "*"=activations')
xlabel('...')
ylabel('...')

%numerical analysis
%R2 statistic for test data
```

```
ttestbar=sum(ttest)/length(ttest);
ss=sum((atest-ttest).^2);
ssybar=sum((atest-ttestbar).^2);
r2=1-ss/ssybar;
fprintf('r2testval = %1.6f\n',r2)
%R2 statistic for training data
ttrainbar=sum(ttrain)/length(ttrain);
ss=sum((atrain-ttrain).^2);
ssybar=sum((atrain-ttrainbar).^2);
r2=1-ss/ssybar;
fprintf('r2trainval = %1.6f\n',r2)
%R2 statistic for validation data
%%NB there may not always be a validation set
tvalbar=sum(tval)/length(tval);
ss=sum((aval-tval).^2);
ssybar=sum((aval-tvalbar).^2);
r2=1-ss/ssybar;
fprintf('r2valval = %1.6f\n',r2)
```

Appendix B: Additional Scripts

B.1. Categorise Data by Rounding

```
function out = round2cat(data,col)
%function round2cat(data,col)
%round dependant variable in order to categorise it for rule mining
%%Variables:
%data=data set on which operation is to be carried out
%col=column in which dependant variable is situated
%NOTE: assumes that dependant variable is in the last column and keeps
all
%other data in tact.
%NB: file generated is called ‘data’ and should be edited in order to
%maintain file name and extension (.txt) as well as to remove any
%extraneous output within the file itself (generally only the first two
%lines).
%
%Douglas Trewartha 2006

    round(data(:,col));
    data=data(:,[1:(col-1)]);
    data=[data,ans];

    diary data
    data
    diary off;

    display('remember to edit the output file and add the .txt
extension')

end
```

B.2. Perform Kmeans Clustering on Japanese Business Data

```
%Kmeanscall
%created by Balasko et al. [2003]
%edited by Douglas Trewartha 2006
%new version works with MATLAB 7.0.1 and performs clustering on
Japanese
%business database [Simonoff 2003]

close all
clear all
```

```
%path(path,'..\..\..\FUZZCLUST')
%previous line did not work with MATLAB 7.0.1 and we thus included the
%following line:
addpath D:\MATLAB701\toolbox\FUZZCLUST

%the database
load japsolv.txt
%the columns containing attribute information
data.X = japsolv(:,[1 2 3 4 5 6 7 8]);

[N,n]=size(data.X);

%data normalization
data = clust_normalize(data,'range');
plot(data.X(:,1),data.X(:,2),'.')
hold on

%parameters
param.c=2;      %number of clusters
param.vis=1;    %visualisation option
param.val=2;    %validity option

%clustering
result=kmeans2(data,param);

%legend
legend('cluster centre','bankrupt','solvent','Location','SouthEast')

%validation
result = validity(result,data,param);
result.validity
```

Appendix C: User Manual

These are instructions for obtaining the results for the Japanese Business database, presented in Chapter 4. This process will be similar for all other case studies.

Requirements:

1. MATLAB 7.0 or higher
2. The Mathworks Neural Network Toolbox for MATLAB
3. ARMADA and The Fuzzy Clustering Toolbox directories copied to the MATLAB\toolbox directory (See "Further Instructions" Below)

Instructions:

Copy the "CD" directory from the CD onto the root directory (this must be the root of the same drive on which MATLAB is installed).

From the "CD" directory:

Clustering:

1. Follow the path: Case Studies→JapaneseBusinessSolvency
2. Extract Clustering.zip (will create folder "Clustering")
3. Open MALTAB and change the working directory to "Clustering" (top left panel)
4. Run "Kmeanscall" from the command prompt (Note: the path may need to be changed, see "Further Instructions" below)
5. Run the script a number of times to get different clusterings

Neural Net:

1. Follow the path: Case Studies→JapaneseBusinessSolvency
2. Extract JapSolvNet.zip (will create folder "JapSolvNet")
3. Open MALTAB and change the working directory to "JapSolvNet"
4. Run "japsolv_sim" from the command prompt
5. Observe results and close

ARMADA:

1. Open MATLAB and change the working directory to the ARMADA toolbox

2. Run “Armada” from the command prompt
3. Load the text file (“japsolv.txt”) using “Browse” and set the delimiting character to “-SPACE-“
4. Set the confidence to 20% and the support to 2 (no.)
5. Select “Mine Using Built Goals” (Under “RULE GOAL BUILDER”)
6. Select “Build Goals”
7. Select “Consequent” from drop down menu
8. Enter “1” and click “New Rule” (do the same for “-1”)
9. Select “Save Rules”
10. “Begin Mining” and wait a few seconds for the results

Further Instructions:

Toolbox Installation:

From the CD:

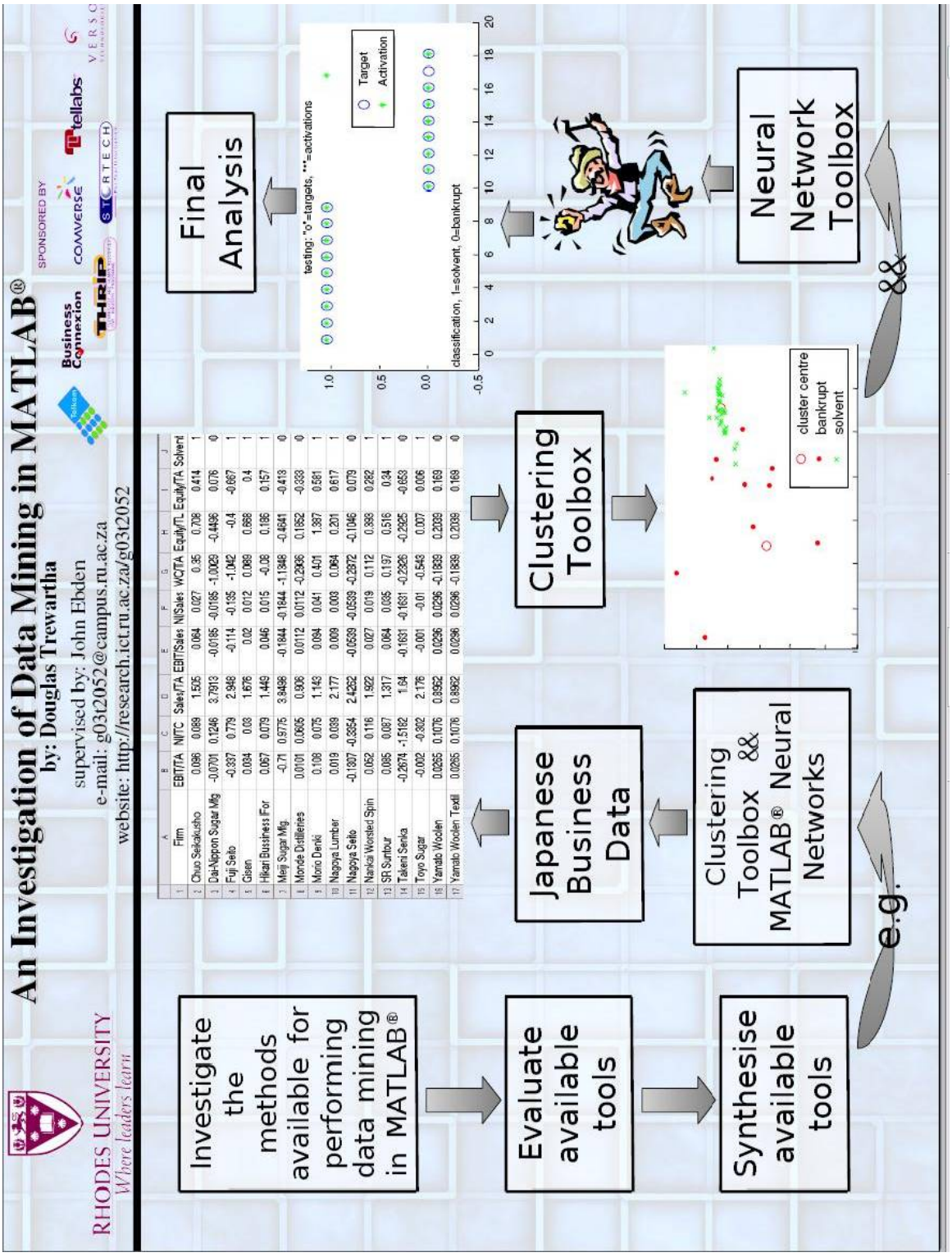
1. Open “Data Mining Tools” Directory
2. Extract ARMADA and Clustering
3. Copy the directories “ARMADA” and “FUZZCLUST” to the MATLAB\toolbox directory

Changing the Path for Clustering:

From “Clustering” (on the CD):

1. Open Kmeanscall
2. On line 13, change the path to the working directory of the installed fuzzy clustering tool.
3. Save and Exit the file

Appendix D: Project Poster



Appendix E: Further Acknowledgements

This appendix consists of details regarding the breast cancer database used in this study and includes acknowledgements required by the databases distributors.

Title: Wisconsin Breast Cancer Database (January 8, 1991)

Sources:

-- Dr. William H. Wolberg (physician)
University of Wisconsin Hospitals
Madison, Wisconsin
USA
-- Donor: Olvie Mangasarian (Mangasarian@cs.wisc.edu)
Received by David W. Aha (aha@ca.jhu,edu)
-- Date: 15 July 1992

Past Usage:

Attributes 2 through 10 have been used to represent instances.
Each instance has one of 2 possible classes: benign or malignant.

1. Wolberg, W.H. & Mangasarian, O.L. (1990). Multisurface method of pattern separation for medical diagnosis applied to breast cytology.
   -- Size of data set: only 369 instances (at that point in time)
   -- Collected classification results: 1 trial only
   -- Two pairs of parallel hyperplanes were found to be consistent with 50% of data
      -- Accuracy of remaining 33% of dataset: 93.5%
   -- Three pairs of parallel hyperplanes were found to be consistent with 67% of data
      -- Accuracy of remaining 33% of dataset: 95.9%

2. Zhang, J. (1992). Selecting typical instances in instance-based learning. Aberdeen, Scotland: Morgan Kaufmann.
   -- Size of data set: only 369 instances (at that point in time)

- -- Applied 4 instance-based learning algorithms
- -- Collected classification results averaged over 10 trials
- -- Best accuracy result:
  - -- 1-nearest neighbour: 93.7%
  - -- Trained on 200 instances, tested on the other 169
- -- Also of interest:
  - -- Using only typical instances: 92.2% (storing only 23.1 instances)
  - -- Trained on 200 instances, tested on the other 169

Relevant Information:

Samples arrive periodically as Dr. Wolberg reports his clinical cases. The database therefore reflects this chronological grouping of the data. This grouping information appears immediately below, having been removed from the data itself:

```
Group 1: 367 instances (January 1989)
Group 2:  70 instances (October 1989)
Group 3:  31 instances (February 1990)
Group 4:  17 instances (April 1990)
Group 5:  48 instances (August 1990)
Group 6:  49 instances (Updated January 1991)
Group 7:  31 instances (June 1991)
Group 8:  86 instances (November 1991)
-----------------------------------------------------
Total: 699 points (as of donated database on 15 July
1992)
```

Note that the results summarised above in Past Usage refer to a dataset of size 369, while Group 1 has only 367 instances. This because it originally contained 369 instances; 2 were removed.

Number of Instances: 699 (as of 15 July 1992)

Number of Attributes: 10 plus the class attribute

Attribute Information: (class attribute has been moved to last column)

#	Attribute	Domain
1.	Sample code number	id number
2.	Clump Thickness	1 – 10
3.	Uniformity of Cell size	1 – 10
4.	Uniformity of Cell shape	1 – 10
5.	Marginal Adhesion	1 – 10
6.	Single Epithelial Cell Size	1 – 10
7.	Bare Nuclei	1 – 10
8.	Bland Chromatin	1 – 10
9.	Normal Nucleoli	1 – 10
10.	Mitoses	1 – 10
11.	Class	(2 for benign, 4 for malignant)

Missing attribute values: 16

There are 16 instances in Groups 1 to 6 that contain a single missing (i.e., unavailable) attribute value.

Class Distribution:

Benign: 458 (65.5%)
Malignant: 241 (34.5%)