Combining rough and fuzzy sets for feature selection

Richard Jensen

Doctor of Philosophy

School of Informatics

University of Edinburgh

2005

Abstract

Feature selection (FS) refers to the problem of selecting those input attributes that are most predictive of a given outcome; a problem encountered in many areas such as machine learning, pattern recognition and signal processing. Unlike other dimensionality reduction methods, feature selectors preserve the original meaning of the features after reduction. This has found application in tasks that involve datasets containing huge numbers of features (in the order of tens of thousands), which would be impossible to process further. Recent examples include text processing and web content classification. FS techniques have also been applied to small and medium-sized datasets in order to locate the most informative features for later use. Many feature selection methods have been developed and are reviewed critically in this thesis, with particular emphasis on their current limitations. The leading methods in this field are presented in a consistent algorithmic framework.

One of the many successful applications of rough set theory has been to this area. The rough set ideology of using only the supplied data and no other information has many benefits in FS, where most other methods require supplementary knowledge. However, the main limitation of rough set-based feature selection in the literature is the restrictive requirement that all data is discrete. In classical rough set theory, it is not possible to consider real-valued or noisy data. This thesis proposes and develops an approach based on fuzzy-rough sets, fuzzy rough feature selection (FRFS), that addresses these problems and retains dataset semantics. Complexity analysis of the underlying algorithms is included.

FRFS is applied to two domains where a feature reducing step is important; namely, web content classification and complex systems monitoring. The utility of this approach is demonstrated and is compared empirically with several dimensionality reducers. In the experimental studies, FRFS is shown to equal or improve classification accuracy when compared to the results from unreduced data. Classifiers that use a lower dimensional set of attributes which are retained by fuzzy-rough reduction outperform those that employ more attributes returned by the existing crisp rough reduction method. In addition, it is shown that FRFS is more powerful than the other FS techniques in the comparative study.

Based on the new fuzzy-rough measure of feature significance, a further development of the FRFS technique is presented in this thesis. This is developed from the new area of feature grouping that considers the selection of groups of attributes in the search for the best subset. A novel framework is also given for the application of ant-based search mechanisms within feature selection in general, with particular emphasis on its employment in FRFS. Both of these developments are employed and evaluated within the complex systems monitoring application.

Acknowledgements

Many thanks to my supervisor, Prof. Qiang Shen, for his guidance, encouragement and effort throughout this project. Without him, I'm sure this Ph.D. and thesis would never have happened. Thanks must also go to Dr. Dave Robertson, my second supervisor, particularly for his input and involvement during the initial stages of this research, and for the support of Advanced Knowledge Technologies (AKT). This work is partly funded by the UK EPSRC grant 00317404.

Additional thanks go to:

- All those in the Approximate and Qualitative Reasoning Group, for the many useful discussions over the years. Particularly, thanks to Michelle Galea for helping with the final stages of the submission of this thesis.
- Dr. Sarah Sholl for lending me her ears and her thesis.
- Dr. Mike Gordon for giving me his thesis and inviting me to set fire to it.
- Iain Langlands for his help, ideas, and the use of his financial dataset.
- Everyone involved in Carrubbers and their invaluable input over the past 4 years. Particularly those at the café.

Finally, thanks to my family and friends who patiently pretended to understand what I was talking about as I attempted to explain fuzzy-rough feature selection.

Most importantly, thanks be to God, who always leads us in triumphal procession in Christ and through us spreads everywhere the fragrance of the knowledge of Him.
(2 CORINTHIANS 2:14)

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Richard Jensen*)

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

It is estimated that every 20 months or so the amount of information in the world doubles. In the same way, tools for use in the various knowledge fields (acquisition, storage, retrieval, maintenance, etc) must develop to combat this growth. Knowledge is only valuable when it can be used efficiently and effectively; therefore knowledge management is increasingly being recognised as a key element in extracting its value.

Central to this issue is the knowledge discovery process, particularly knowledge discovery in databases (KDD) [4, 47, 49, 155]. Traditionally, data was turned into knowledge by means of manual analysis and interpretation. For many applications, this form of manual probing of data is slow, costly, and highly subjective. Indeed, as data volumes grow dramatically, this type of manual data analysis is becoming completely impractical in many domains. This motivates the need for efficient, automated knowledge discovery. The KDD process can be decomposed into the following steps, as illustrated in figure 1.1:

- Data Selection:
  A target dataset is selected or created. Several existing datasets may be joined together to obtain an appropriate example set.

- Data Cleaning/Preprocessing:
  This phase includes, among other tasks, noise removal/reduction, missing value imputation, and attribute discretization. The goal of this is to improve the overall quality of any information that may be discovered.

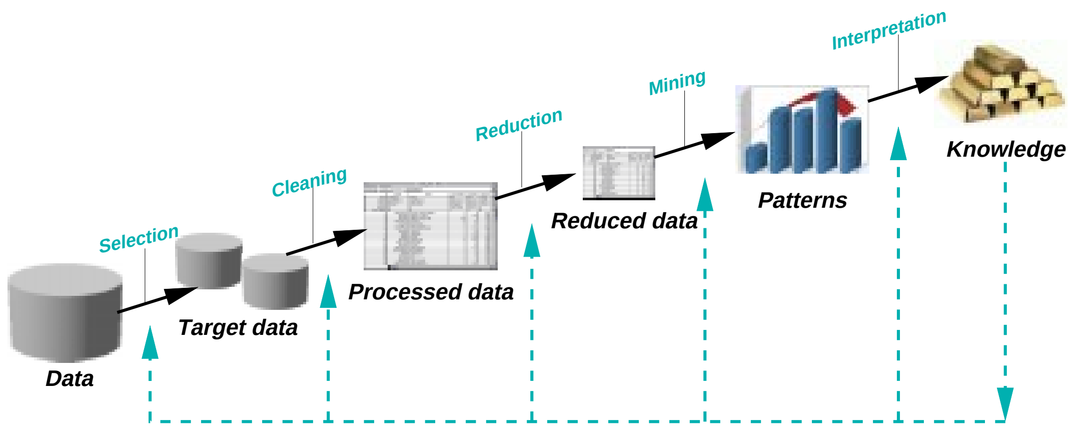


Figure 1.1: The knowledge discovery process

- Data Reduction:
  Most datasets will contain a certain amount of redundancy that will not aid knowledge discovery and may in fact mislead the process. The aim of this step is to find useful features to represent the data and remove non-relevant ones. Time is also saved during the data mining step as a result of this.

- Data Mining:
  A data mining method (the extraction of hidden predictive information from large databases) is selected depending on the goals of the knowledge discovery task. The choice of algorithm used may be dependent on many factors, including the source of the dataset and the values it contains.

- Interpretation/Evaluation:
  Once knowledge has been discovered, it is evaluated with respect to validity, usefulness, novelty and simplicity. This may require repeating some of the previous steps.

The third step in the knowledge discovery process, namely data reduction, is the point of interest for this investigation as this is often a source of significant data loss. The high dimensionality of databases can be reduced using suitable techniques, depending on the requirements of the future KDD processes. These techniques fall in to one of two categories: those that transform the underlying meaning of the data features and those that are semantics-preserving. Feature selection (FS) methods belong to the latter category, where a smaller set of the original features is chosen based on a subset evaluation function. In knowledge discovery, feature selection methods are particularly desirable as these facilitate the interpretability of the resulting knowledge.

1.1 Feature Selection

There are often many features in KDD, and combinatorially large numbers of feature combinations, to select from. Note that the number of feature subset combinations with m features from a collection of N total features is $N!/[m!(N-m)!]$. It might be expected that the inclusion of an increasing number of features would increase the likelihood of including enough information to distinguish between classes. Unfortunately, this is not true if the size of the training dataset does not also increase rapidly with each additional feature included. This is the so-called curse of dimensionality [13]. A high-dimensional dataset increases the chances that a data-mining algorithm will find spurious patterns that are not valid in general. Most techniques employ some degree of reduction in order to cope with large amounts of data, so an efficient and effective reduction method is required.

The use of rough set theory (RST) [125] to achieve data reduction is one approach that has proved successful. Over the past twenty years, rough set theory has become a topic of great interest to researchers and has been applied to many domains (e.g. classification [29, 42, 71], systems monitoring [158], clustering [62], expert systems [176]). This success is due in part to the following aspects of the theory:

- Only the facts hidden in data are analysed,
- No additional information about the data is required such as thresholds or expert knowledge on a particular domain,
- It finds a minimal knowledge representation.

Given a dataset with discretized attribute values, it is possible to find a subset of the original attributes using RST that are the most informative (termed a *reduct*); all other attributes can be removed from the dataset with minimal information loss. This method tends to be a pre-processing step to reduce dataset dimensionality before some other action is performed (for example, induction of rules [157]).

1.2 Applications of Feature Selection

As many systems in a variety of fields deal with datasets of large dimensionality, feature selection has found wide applicability. Some of the main areas of application are shown in figure 1.2.

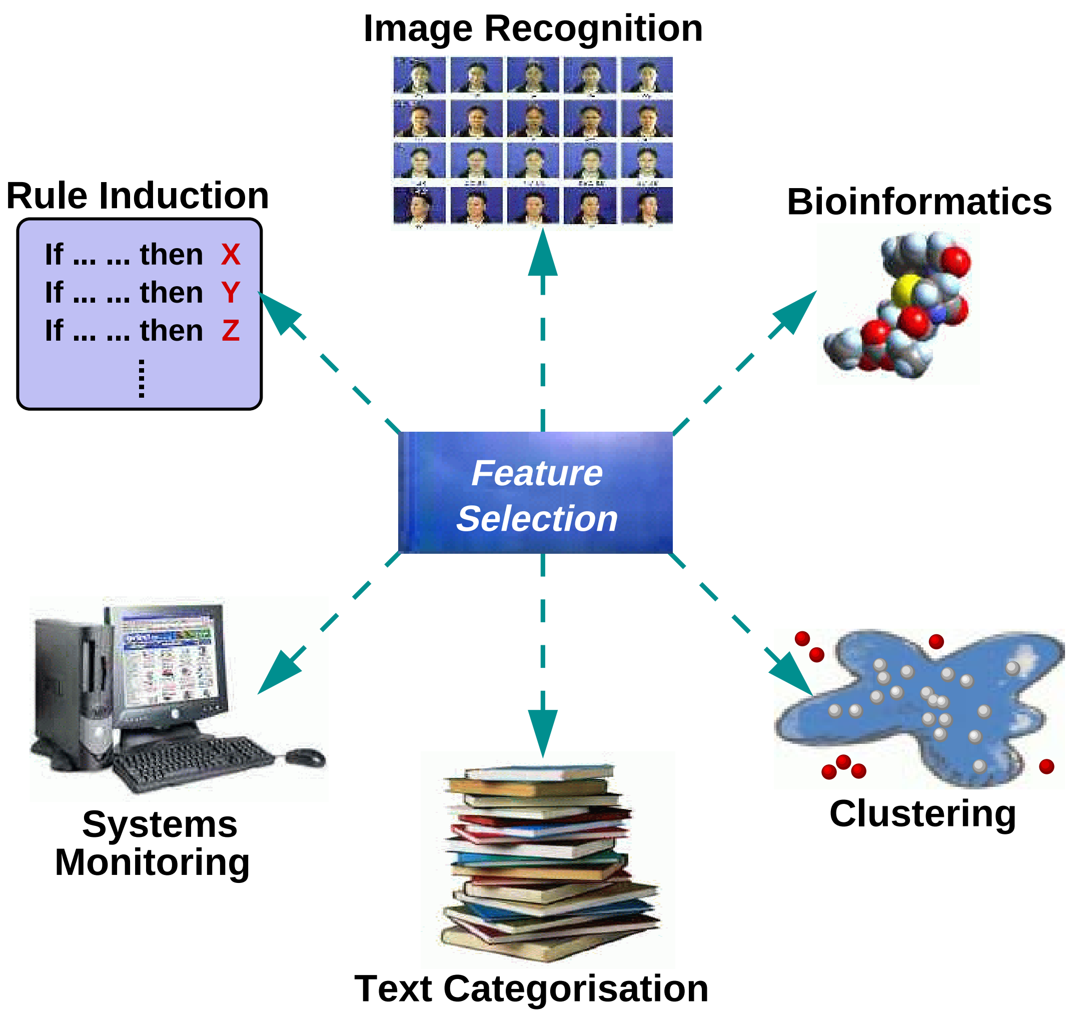


Figure 1.2: Feature selection application areas

Feature selection algorithms are often applied to optimize the classification performance of image recognition systems [70, 163]. This is motivated by a peaking phenomenon commonly observed when classifiers are trained with a limited set of training samples. If the number of features is increased, the classification rate of the classifier decreases after a peak. In melanoma diagnosis, for instance, the clinical accuracy of dermatologists in identifying malignant melanomas is only between 65% and 85% [59]. With the application of FS algorithms, automated skin tumour recognition systems can produce classification accuracies above 95%.

Structural and functional data from analysis of the human genome have increased many fold in recent years, presenting enormous opportunities and challenges for AI tasks. In particular, gene expression microarrays are a rapidly maturing technology that provide the opportunity to analyse the expression levels of thousands or tens of thousands of genes in a single experiment. A typical classification task is to distinguish between healthy and cancer patients based on their gene expression profile. Feature selectors are used (along with some initial filtering) to drastically reduce the size of these datasets which would otherwise have been unsuitable for further processing [189, 190]. Other applications within bioinformatics include QSAR [25], where the goal is to form hypotheses relating chemical features of molecules to their molecular activity, and splice site prediction [143], where junctions between coding and non-coding regions of DNA are detected.

The most common approach to developing expressive and human readable representations of knowledge is the use of if-then production rules. Yet, real-life problem domains usually lack generic and systematic expert rules for mapping feature patterns onto their underlying classes. In order to speed up the rule induction process and reduce rule complexity, a selection step is required. This reduces the dimensionality of potentially very large feature sets while minimising the loss of information needed for rule induction. It has an advantageous side-effect in that it removes redundancy from the historical data. This also helps simplify the design and implementation of the actual pattern classifier itself, by determining what features should be made available to the system. In addition, the reduced input dimensionality increases the processing speed of the classifier, leading to better response times [6, 28, 74].

Many inferential measurement systems are developed using data-based methodologies; the models used to infer the value of target features are developed using real-time plant data. This implies that inferential systems are heavily influenced by the quality of the data used to develop their internal models. Complex application problems, such as reliable monitoring and diagnosis of industrial plants, are likely to present large numbers of features, many of which will be redundant for the task at hand. Additionally, there is an associated cost with the measurement of these features. In these situations it is very useful to have an intelligent system capable of selecting the most relevant features needed to build an accurate and reliable model for the process [73, 80, 158, 136].

The task of text clustering is to group similar documents together, represented as a bag of words. This representation raises one severe problem: the high dimensionality of the feature space and the inherent data sparsity. This can significantly affect the performance of clustering algorithms, therefore it is highly desirable to reduce this feature space size. Dimensionality reduction techniques have been successfully applied to this area - both those that destroy data semantics and those that preserve them (feature selectors) [34, 97].

Similar to clustering, text categorization views documents as a collection of words. Documents are examined, with their constituent keywords extracted and rated according to criteria such as their frequency of occurrence. As the number of keywords extracted is usually in the order of tens of thousands, dimensionality reduction must be performed. This can take the form of simplistic filtering methods such as word stemming or the use of stop-word lists. However, these do not provide enough reduction for use in automated categorisers, so a further feature selection process must take place. Recent applications of FS in this area include web page and bookmark categorisation [51, 77].

1.3 Limitations of Current Methods

The use of user-supplied information is essential to many existing algorithms for feature selection in the literature. This is a significant drawback. Some feature selectors require noise levels to be specified by the user beforehand, some simply rank features leaving the user to choose their own subset. There are those that require the user to state how many features are to be chosen, or they must supply a threshold that determines when the algorithm should terminate. All of these require the user to make a decision based on their own (possibly faulty) judgement.

It is most often the case that the values of attributes may be both crisp and *real-valued*, and this is where many feature selectors, particularly those based on traditional rough set theory, encounter a problem. It is not possible to say whether two attribute values are similar and to what extent they are the same; for example, two close values may only differ as a result of noise, but in RST they are considered to be as different as two values of a different order of magnitude. According to RST, the values -0.1 and -0.11 are as different as -0.1 and 300.

One answer to this problem has been to discretise the dataset beforehand, producing a new dataset with crisp values. This is often still inadequate, however, as the degrees of membership of values to discretised values are not considered at all. For example, two values may both be mapped to the same label “Negative”, but one may be much more negative than the other. The values -0.1 and -2000 could both be mapped to this class, though they are significantly different. This is a source of information loss, which is against the rough set ideology of retaining information content. Further discussion of this issue can be found in [16].

It is clear that there is a need for some method that will provide the means of data reduction for crisp and real-value attributed datasets which utilises the extent to which values are similar. Fuzzy sets [200] and the process of fuzzification provide a mechanism by which real-valued features can be effectively managed. By allowing values to belong to more than one label, with various degrees of membership, the vagueness present in data can be modelled. This information may then be exploited by further fuzzy methods to enable reasoning under uncertainty.

For feature selection, this could be achieved through the use of *fuzzy-rough* sets [43]. Fuzzy-rough set theory is an extension of crisp rough set theory, allowing all memberships to take values in the range [0,1]. This permits a higher degree of flexibility compared to the strict requirements of crisp rough sets that only deal with full or zero set membership.

Also, there are few feature selection methods that can handle data with continuous decision attributes. This type of data is often encountered in regression/function approximation problems. For example, in QSAR [25] such datasets are encountered where the measurements (including the decision feature, molecular activity) are all continuous. The use of FS can discover the key variables that influence the decision quantity, without transforming the feature space.

1.4 Fuzzy-Rough Feature Selection and Extensions

Fuzzy-rough sets encapsulate the related but distinct concepts of vagueness (for fuzzy sets [200]) and indiscernibility (for rough sets), both of which occur as a result of uncertainty in knowledge [43]. A fuzzy-rough set is defined by two fuzzy sets, fuzzy lower and upper approximations, obtained by extending the corresponding crisp rough set notions. In the crisp case, elements that belong to the lower approximation (i.e. have a membership of 1) are said to belong to the approximated set with absolute certainty. In the fuzzy-rough case, elements may have a membership in the range [0,1], allowing greater flexibility in handling uncertainty.

Fuzzy-Rough Feature Selection (FRFS) provides a means by which discrete or real-valued noisy data (or a mixture of both) can be effectively reduced without the need for user-supplied information. Additionally, this technique can be applied to data with continuous or nominal decision attributes, and as such can be applied to regression as well as classification datasets. The only additional information required is in the form of fuzzy partitions for each feature which can be automatically derived from the data. In the work presented here, excluding the simple examples, *all* fuzzy sets are derived solely from the data. This avoids the need for domain experts to provide information on the data involved and ties in with the advantage of rough sets in that it requires no information other than the data itself. However, if such experts are readily available, it is beneficial to capture their knowledge in the form of fuzzy data partitions to improve the transparency of the selection process and any other future processes (e.g. rule induction).

FRFS forms the main contribution of this thesis, employing fuzzy-rough sets to address several of the current limitations of feature selectors. The algorithm for finding optimal or close-to-optimal feature subsets is given. This can be replaced by other search strategies. In this thesis, two alternative search mechanisms are presented, both of which may be applied to the feature selection task in general as well as FRFS in particular. The first development builds on new ideas in the feature selection field concerning feature grouping. At various stages in the selection task, features may be grouped and many selected simultaneously. The second development draws on work in swarm intelligence for solving combinatorial optimization problems [21]. By re-modelling the selection task to conform to the ant colony optimization framework [40], a new swarm intelligence-based feature subset search mechanism can be implemented.

1.5 Applications of FRFS

FRFS can be applied to any of the domains highlighted previously where feature selection has been employed. For the purposes of this thesis, two challenging domains of interest were chosen to illustrate its potential utility; web content categorisation and complex systems monitoring.

The problem of web content categorization is a significant one due to the explosive increase of information available on the web and its increasing popularity. Techniques for automated categorization of web documents help in the building of catalogues and facilitate the retrieval of material. In order to deal with the large number of features involved in such classification, feature selectors are typically used [151]. The dimensionality of the problem datasets can be sufficiently reduced to enable more sophisticated learning algorithms to perform their tasks. The work presented here looks specifically at addressing the issues of bookmark/favorite classification and web page classification. FRFS reduces the size of the datasets involved by several orders of magnitude, retaining most of the information present in the datasets and making the classification task manageable.

In systems monitoring, it is important to reduce the number of features involved for several reasons. Firstly, there is an associated cost with the measurement of a feature.

It is desirable simply from an expense-saving point of view to reduce the number of monitored variables. Secondly, the resultant transparency of the monitoring process can be improved if fewer variables are involved. Thirdly, it is often observed that the accuracy of the monitoring system can be significantly improved using fewer variables [158]. FRFS is here applied to the water treatment plant dataset [20] as an example of how this fuzzy-rough method can be used within the systems monitoring domain. Additionally, the new feature grouping and ant colony optimization-based methods are applied to this domain to show their potential utility.

1.6 Thesis Structure

The rest of this thesis is structured as follows (with an indication of the publications produced as a result of this research):

- **Chapter 2**: *Background*. A systematic overview of current techniques for dimensionality reduction with a particular emphasis on feature selection is given in this chapter. It begins with a discussion of those reduction methods that irreversibly transform data semantics. This is followed by a more detailed description and evaluation of the leading feature selectors presented in a unified algorithmic framework. A simple example illustrates their operation.

- **Chapter 3**: *Rough Set-based Approaches to Feature Selection*. This chapter presents the current state of research regarding the application of rough set theory to feature selection. It is an extended version of the work appearing in [78]. Rough Set Attribute Reduction (RSAR), the precursor to the developments in this thesis, is described in detail. However, these methods are unsuited to the problems discussed in section 1.3. In particular, they are unable to handle noisy or real-valued data effectively - a significant problem if they are to be employed within real-world applications.

- **Chapter 4**: *Fuzzy-Rough Feature Selection*. In this chapter, the theoretical developments behind this new feature selection method are presented together with a proof of generalisation. This novel approach uses fuzzy-rough sets to handle many of the problems facing feature selectors outlined previously. A complexity analysis of the main selection algorithm is given. The operation of the approach and its benefits are shown through the use of two simple examples. To evaluate this new fuzzy-rough measure of feature significance, comparative investigations are carried out with the current leading significance measures. The contents of this chapter have been published in [72, 74, 77]

- **Chapter 5**: *Developments of FRFS*. Based on FRFS, this chapter introduces two promising areas in feature selection. The first, feature grouping, is developed from recent work in the literature where groups of features are selected simultaneously. By reasoning with fuzzy labels, the search process can be made more intelligent allowing various search strategies to be employed [75]. The second, ant-based feature selection, seeks to address the non-trivial issue of finding the smallest optimal feature subsets. This new approach to feature selection uses artificial ants and pheromone trails in the search for the best subsets [76, 79]. Both of these developments can be applied within feature selection in general, but are applied to the specific problem of subset search within FRFS in this thesis.

- **Chapter 6**: *Application to Web Content Categorisation*. With the explosive growth of information on the web, there is an abundance of information that must be dealt with effectively and efficiently. This area in particular deserves the attention of feature selection due to the increasing demand for high-performance intelligent internet applications. This motivates the application of FRFS to the automatic categorization of user bookmarks/favorites and web pages [71, 77]. The results show that FRFS significantly reduces data dimensionality by several orders of magnitude with little resulting loss in classification accuracy.

- **Chapter 7**: *Application to Complex Systems Monitoring*. Complex application problems, such as reliable monitoring and diagnosis of industrial plants, are likely to present large numbers of features, many of which will be redundant for the task at hand. With the use of FRFS, these extraneous features can be removed. This not only makes resultant rulesets generated from such data much more concise and readable, but can reduce the expense due to the monitoring of redundant features. The monitoring system is applied to water treatment plant data, producing better classification accuracies than those resulting from the full feature set and several other reduction methods (including both semantics-preserving and transformation-based approaches) [73, 158].

- **Chapter 8**: *Supplementary Developments and Investigations*. This chapter presents initial investigations and ideas for further work, which were developed concurrently with the ideas presented in the previous chapters. Firstly, the utility of using the problem formulation and solution techniques from propositional satisfiability for finding rough set reducts is considered. This is presented with an initial experimental evaluation of such an approach, comparing the results with a standard rough set-based algorithm, RSAR. Secondly, the possibility of universal reducts is proposed as a way of generating more useful feature subsets. Finally, fuzzy decision tree induction based on the fuzzy-rough metric developed in this thesis is proposed.

- **Chapter 9**: *Conclusion*. The thesis is concluded in this chapter, with a summary of the key findings from the research conducted here. There is also a discussion of future work to be carried out in the area of feature selection in general, as well as fuzzy-rough feature selection in particular.

Chapter 2

Background

There are many factors that motivate the inclusion of a dimensionality reduction (DR) step in a variety of problem-solving systems [24]. Many application problems process data in the form of a collection of real-valued vectors (for example, text classification [193], bookmark categorization [71]). If these vectors exhibit a high dimensionality, then processing becomes infeasible. Therefore, it is often useful, and sometimes necessary, to reduce the data dimensionality to a more manageable size with as little information loss as possible. The process is summarised in figure 2.1, where the dimensionality reduction step is a preprocessing stage in the whole system.

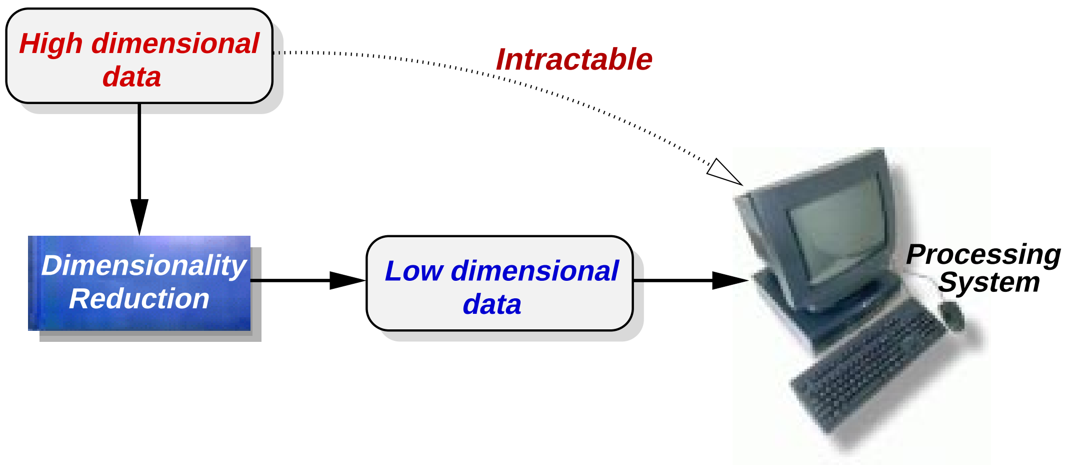


Figure 2.1: The dimension reduction problem

Sometimes, high-dimensional complex phenomena can be governed by significantly fewer, simple variables [48]. The process of dimensionality reduction here will act as a tool for modelling these phenomena, improving their clarity. There is often a significant amount of redundant or misleading information present; this will need to be removed before any further processing can be carried out. For example, the problem of deriving classification rules from large datasets often benefits from a data reduction preprocessing step [157]. Not only does this reduce the time required to perform induction, but it makes the resulting rules more comprehensible and can increase the resulting classification accuracy.

Dimension reduction problems tend to be classified into one of three categories (unless the time variable is included which may present two further categories: *static* and *dynamic*):

- *Hard* dimension reduction problems, where the data may have dimensions ranging from hundreds to hundreds of thousands of components. Usually, a severe reduction is required. Typical methods include principal component analysis (PCA) [38] and rough set analysis [29].

- *Soft* dimension reduction problems, where the data is fairly low-dimensional, usually less than a few tens of components. A typical method for this purpose is factor analysis [108].

- *Visualisation* problems, where the task is to extract and visually represent relationships within a dataset. Given a set of data with high dimensionality, these methods effectively summarize the content into four or fewer dimensions, enabling graphing or other means of visualising the results. A number of methods are used for this, such as projection pursuit [52] and multidimensional scaling [180].

This chapter deals primarily with hard dimension reduction problems as this is the area of most interest. However, many hard dimensionality reduction techniques destroy the underlying meaning behind the features present in a dataset (the semantics) - an undesirable property for many applications. This is particularly the case where the understanding of the data processing method and that of the resulting processed data is as important as the accuracy of the resultant lower dimensional dataset in use. The primary focus of this chapter, therefore, is on those techniques that perform dimensionality reduction whilst preserving the meaning of the original dataset. In particular, it has a focus on the recent development on the use of rough set theory [125] for feature selection as this forms the basis for the new developments presented in this thesis.

A taxonomy of dimensionality reduction techniques is presented in figure 2.2. The key distinction made within the taxonomy is whether a DR technique transforms or preserves the dataset semantics in the process of reduction. The choice of DR technique is often guided by the particular application it will be a part of. For example, if an application needs to use the meaning of the original feature set, then a DR technique must be chosen that will ensure this preservation. If, on the other hand, an application requires a visualisation of relationships within the dataset, then a DR approach that transforms the data into two or three dimensions whilst emphasizing those relationships may be more beneficial.

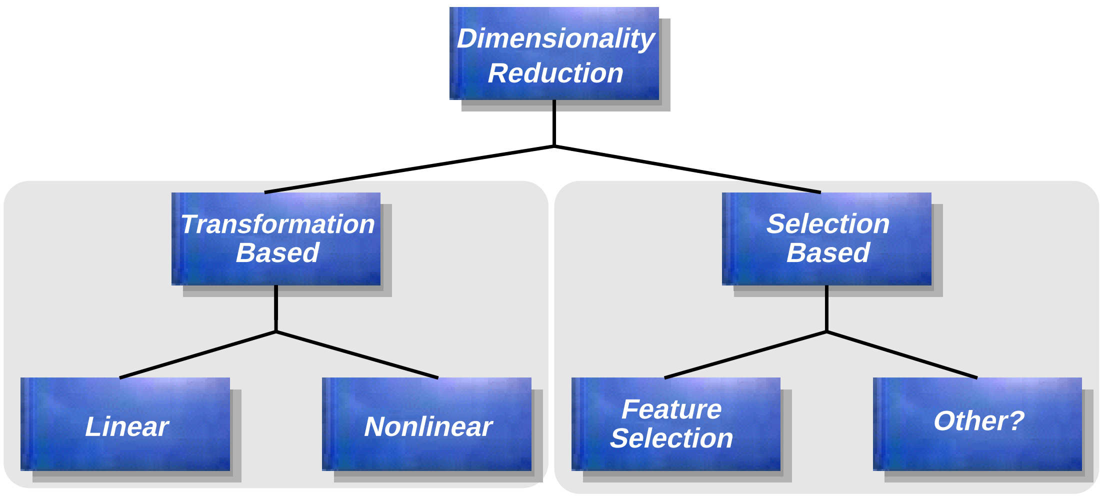


Figure 2.2: Taxonomy of dimension reduction approaches

Included in this taxonomy is the possibility of a semantics-preserving dimensionality reduction technique other than feature selection. Perhaps those techniques that perform a task where semantics-preserving dimensionality reduction is a side-effect may be classified here. For example, the machine learning algorithm C4.5 [135] constructs decision trees from data, selecting features and partitioning the dataset in the process. The resulting decision trees often involve fewer features than the original training data, so a degree of dimensionality reduction has been performed.

This chapter will first examine those techniques that irreversibly destroy feature semantics in the process of dataset dimensionality reduction, separating these into linear and nonlinear techniques. Next, semantics-preserving (feature selection) techniques are examined and their advantages and disadvantages discussed. For those methods that require it, an algorithmic outline is given.

2.1 Transformation-based Reduction

Common throughout the DR literature are approaches that reduce dimensionality but in the process irreversibly transform the descriptive dataset features. These methods are employed in situations where the semantics of the original dataset are not needed by any future process. This section briefly discusses several such popular techniques, which are separated into two categories: those methods that are linear and those that are nonlinear.

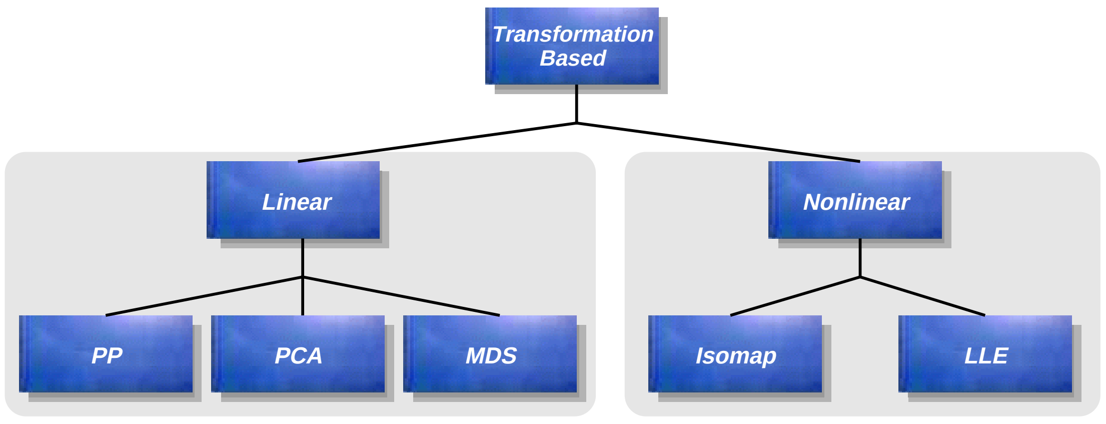


Figure 2.3: Classification of representative semantics-destroying DR techniques

2.1.1 Linear Methods

Linear methods of dimensionality reduction have been well developed over the years and include techniques such as Principal Component Analysis [82] and Multidimensional Scaling [180]. These techniques are used to determine the Euclidean structure of a dataset's internal relationships. However, when such relationships are of a higher dimensionality, these methods generally fail to detect this. This problem is not too restrictive as for many applications linear DR is all that is needed.

2.1.1.1 Principal Component Analysis

Principal Component Analysis (PCA) is a dimensionality reduction tool in common use, perhaps due to its conceptual simplicity and the existence of relatively efficient algorithms for its computation. PCA transforms the original features of a dataset with a (typically) reduced number of uncorrelated ones, termed principal components.

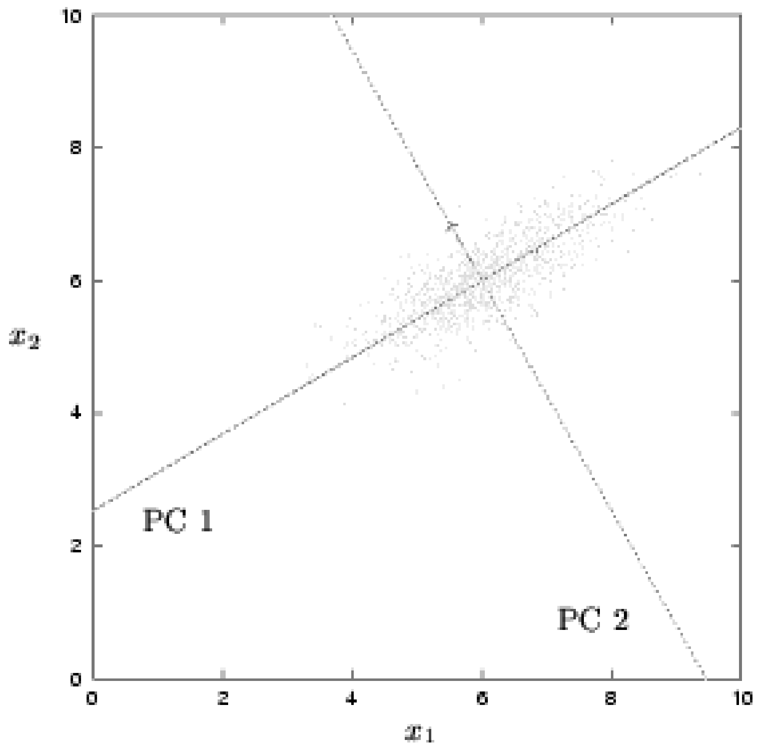


Figure 2.4: 2-dimensional normal point cloud with corresponding principal components

PCA works on the assumption that a large feature variance corresponds to useful information, with small variance equating to information that is less useful. Figure 2.4 shows an example of this, where the principal components of a two-dimensional normal point cloud are given. The first principle component indicates the direction of maximum data variance; the data is seen to be the most dispersed along this new axis in the example. PCA is employed in section 7.3.3 to perform dimensionality reduction within a monitoring system.

Data is transformed in such a way as to allow the removal of those transformed features with small variance. This is achieved by finding the eigenvectors of the covariance matrix of data points (objects), constructing a transformation matrix from the ordered eigenvectors, and transforming the original data by matrix multiplication. Consider a sample $\{\mathbf{x}_i\}_{i=1}^{n}$ of dimensionality D, with mean $\overline{x} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$ and covariance matrix $\Sigma = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \overline{x})(\mathbf{x}_i - \overline{x})^T$. The covariance matrix is symmetric with spectral decomposition $\Sigma = U\Lambda U^T$ $(U = (u_1,\ldots,u_D)$ and $\Lambda = diag(\lambda_1,\ldots,\lambda_D)$). Here, u_t is the normalised eigenvector of Σ with corresponding eigenvalue λ_t. The principal component transformation $y = U^T(\mathbf{x} - \overline{x})$ produces a reference system in which the sample has a mean of zero, and a diagonal covariance matrix containing the eigenvalues of Σ. The variables are now uncorrelated: those with small variance can be discarded, reducing the dimensionality of the dataset.

However, it is not known beforehand the number of variables to discard, introducing a potential source of error. An informed guess has to be made as to how many variables should be kept. An interesting recent approach to tackling this problem can be found in [177]. In this work, variable selection based on rough sets is applied to the results of the PCA data transformation. In order to achieve this, the transformed data has to be discretized.

PCA is not suitable for datasets with nominal attributes either, as matrix calculations are not applicable. It is not as effective with data correlated in a non-linear manner. For example, given 2-D data lying on the edge of a circle it is intuitive that this could be mapped onto 1-D using the angle that the data forms with the x-axis. PCA, however, would create two axes, each accounting for an equal amount of variance and hence no reduction would take place.

2.1.1.2 Projection Pursuit

Projection Pursuit (PP) [52, 53] attempts to optimize a quality metric in the search for a lower-dimensional projection of data. Potentially interesting projections are selected by the local optimization over projection directions of a certain index of "interestingness". This notion is motivated by the observation that for high dimensional data, most low dimensional projections are approximately normal. Hence, those projections that produce single dimensional projected distributions far from the normal distribution are determined to be interesting. Different projection indices arise from alternative definitions of normality deviation.

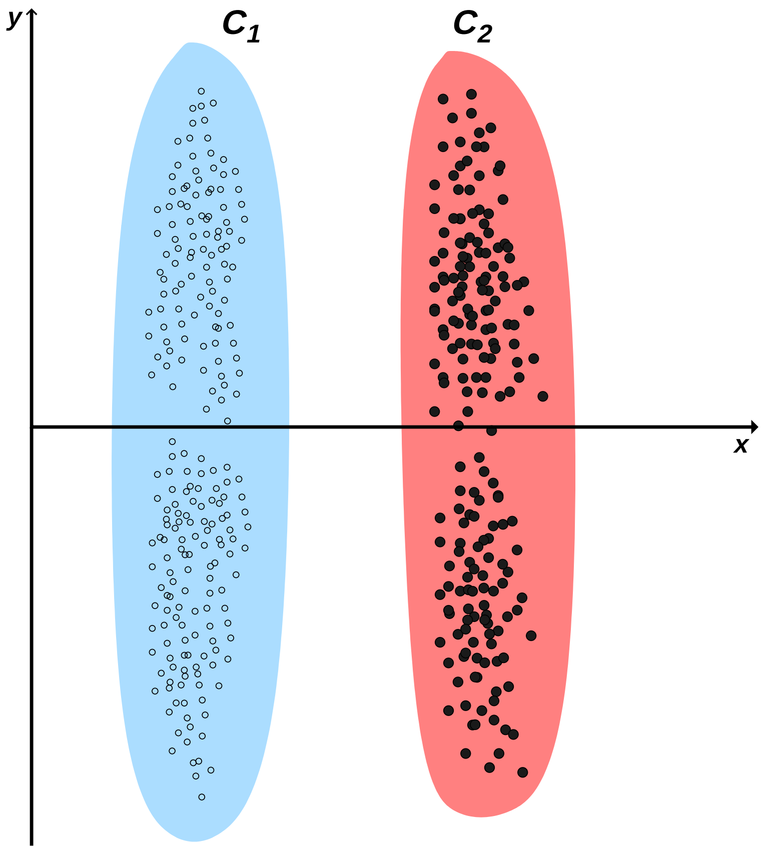


Figure 2.5: Two data clusters

How projections may discover structure in data can be illustrated by considering the data in figure 2.5, where there are two clusters in the data, C_1 and C_2. Although the data has largest variance in the *y*-axis, the clusters cannot be separated by projecting to this axis. In this case, the data variance is not a good indicator of useful information in the PCA sense. By projecting to the *x*-axis, these clusters may be successfully separated.

Typically, linear projections are used due to their simplicity and interpretability. Often there will be several projections determined to be interesting which correspond to projection index local optima. Each such projection may highlight a different (but equally interesting) aspect of the structure of the high dimensional data. This can be a problem for some projection indices as the existence of many local maxima makes locating global maxima difficult.

This suffers from many of the disadvantages of PCA. In fact, it can be shown that PCA is a special case of PP [24]. As PP works with linear projections, it is not suited to deal with highly non-linear structure. In addition, PP methods are computationally intensive. For instance, the projection index and derivatives must be rapidly computable due to repeated evaluations.

2.1.1.3 Multidimensional Scaling

Multidimensional scaling (MDS) refers to a class of techniques that use proximities of objects as input and display its structure as a geometrical picture. Proximities are a measure of the similarities (or dissimilarities) of data points. The resulting transformation to lower dimensions attempts to preserve these original proximities as far as possible. Classical MDS has its origins in psychometrics, where it was proposed to help understand people's judgments of the similarity of members of a set of objects [138].

As an example, table 2.1 presents the distances between ten cities (in the U.S.) placed in a matrix. The distances are the proximities and can be thought of as creating a dissimilarity matrix. When the data is run through an MDS algorithm using two dimensions the algorithm constructs a map based on the proximities as shown in figure 2.6. Since there is no error in the data, it reconstructs a map that shows the relative locations of the cities.

One way of modelling distance is to use the Euclidean metric. That is, the distance

	Atl	Chi	Den	Hou	L.A.	Mia	N.Y.	San F.	Sea	Was DC
Atl	0	587	1212	701	1936	604	748	2139	2182	543
Chi	587	0	920	940	1745	1188	713	1858	1737	597
Den	1212	920	0	879	831	1726	1631	949	1021	1494
Hou	701	940	879	0	1374	968	1420	1645	1891	1220
L.A.	1936	1745	831	1374	0	2339	2451	347	959	2300
Mia	604	1188	1726	968	2339	0	1092	2594	2734	923
N.Y.	748	713	1631	1420	2451	1092	0	2571	2408	205
San F.	2139	1858	949	1645	347	2594	2571	0	678	2442
Sea	2182	1737	1021	1891	959	2734	2408	678	0	2329
Was DC	543	597	1494	1220	2300	923	205	2442	2329	0

Table 2.1: Mileages between ten U.S. cities

$dist(\mathbf{x}_i, \mathbf{x}_j)$ between points $\mathbf{x}_i$ and $\mathbf{x}_j$ of dimensionality A is defined as

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{a=1}^{A} (\mathbf{x}_{ia} - \mathbf{x}_{ja})^2} \tag{2.1}$$

The first developments of MDS were metric [180, 110]; the values used had to be quantitative, complete and symmetric (as in the example given above). MDS requires as input the n data points (objects) in the dataset $\mathbf{X}$ and also a value r, where the value of r is an estimate of the true dimensionality of the dataset, or the required number of dimensions for visualization, typically two or three. If the input data are not distances, these can be computed using a suitable distance metric, resulting in an $n \times n$ matrix $\mathbf{S}$. The coordinates to be computed, $\mathbf{x}_{ia}$, are contained in the $n \times r$ matrix $\mathbf{D}$. A suitable objective function (called the stress) must be defined which is employed to minimize the discrepancies between the current and original data distances:

$$stress(\mathbf{D}) = \sqrt{\sum_{i,j=1}^{n} (dist(\mathbf{x}_i, \mathbf{x}_j) - dist(\mathbf{d}_i, \mathbf{d}_j))^2} \tag{2.2}$$

The distances contained in the matrix $\mathbf{D}$ are calculated in such a way as to closely resemble the dissimilarities $\mathbf{S}$, often by means of least-squares. $\mathbf{E}$ is a matrix of errors that are, in the least-squares optimization situation, to be minimized. As the distances $\mathbf{S}$ are a function of the coordinates $\mathbf{X}$, the goal of classical MDS is to calculate the

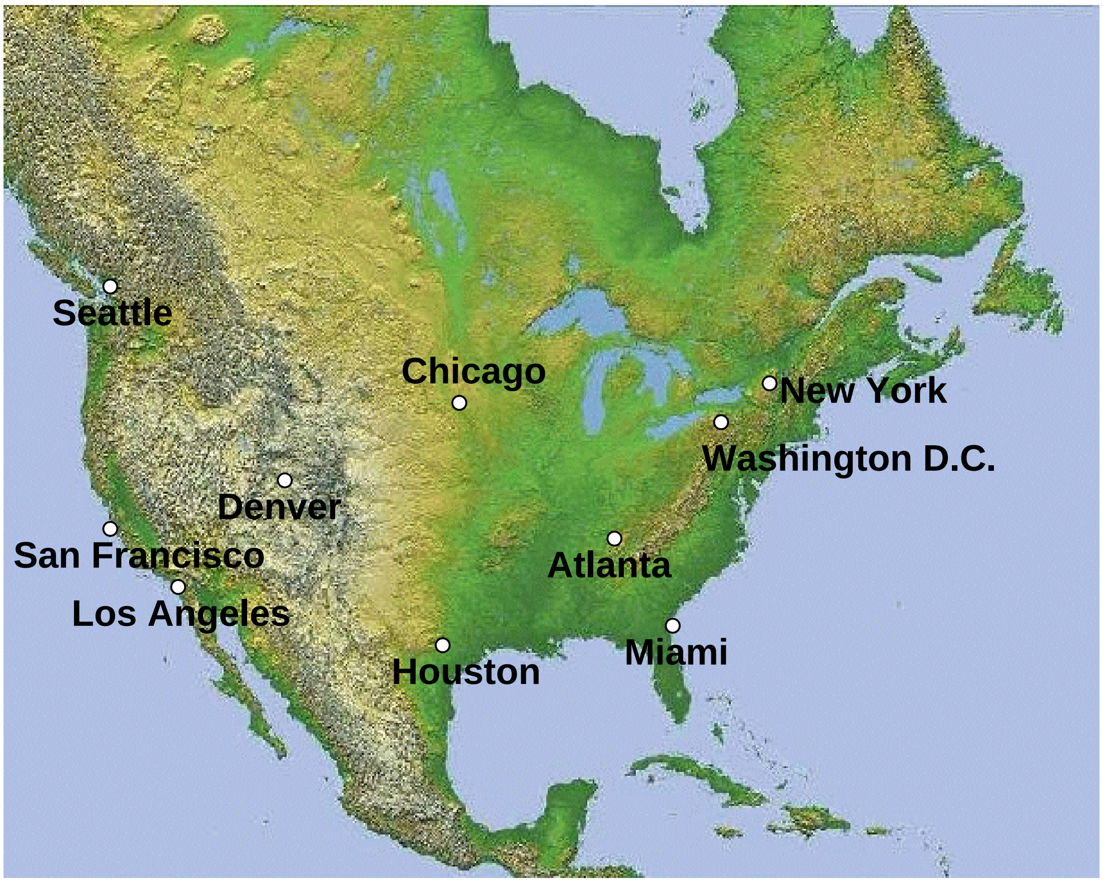


Figure 2.6: Resulting 2-dimensional map

coordinates **D** so that the sum of squares of **E** is minimized, subject to suitable normalization of **X**.

The inability of this class of MDS to handle asymmetric and incomplete data proves too restrictive for most real datasets. This led to the development of non-metric MDS to allow for these possibilities and additionally enabling the use of ordinal data [90, 159]. Other extensions to the approach include replicated MDS (RMDS) [197] where the simultaneous analysis of several matrices of similarity data is allowed, and weighted MDS (WMDS) [197].

WMDS generalises the distance model so that several similarity matrices could be assumed to differ from each other in systematically nonlinear or nonmonotonic ways. Whereas RMDS only accounts for individual differences in the ways subjects use the response scale (in psychological terms), WMDS incorporates a model to account for individual differences in the fundamental perceptual or cognitive processes that generate the responses. For this reason, WMDS is often called individual differences scaling.

2.1.2 Nonlinear Methods

As useful as the previous methods are for reducing dimensionality, their utility fails for nonlinear data. Given a dataset containing nonlinear relationships, these methods detect only the Euclidean structure. This brought about the need for methods that can effectively handle nonlinearity. The first attempts were extensions to the original PCA process, either by clustering data initially and performing PCA within clusters [23], or by greedy optimization processes [89]. Both suffer from problems brought on as a result of simply attempting to extend linear PCA [57]. This motivates the development of techniques designed to suitably and successfully handle nonlinearity.

2.1.2.1 Isomap

Isomap [178] is an extension of MDS in which embeddings are optimized to preserve geodesic distances between pairs of data points, estimated by calculating the shortest paths through large sublattices of data. The algorithm can discover nonlinear degrees of freedom as these geodesic distances represent the true low-dimensional geometry of the manifold. The “Swiss roll” dataset in figure 2.7 is an example of a set of data points that exhibit a clear nonlinear relationship. Isomap successfully determines this nonlinear structure: nearby points in the 2D embedding are also nearby points in the original 3D manifold.

The Isomap algorithm requires as input the distance $d_X(i, j)$ between all object pairs i, j from N objects (data points) in the original input space X. This can be measured by a suitable metric such as the standard Euclidean metric. The algorithm takes this input and constructs the neighbourhood graph over all objects subject to proximity constraints. Points i and j are connected if they are closer than ε (this is called ε-Isomap) or if i is one of the K nearest neighbours of j (K-Isomap). The lengths of the edges become the corresponding distance between them, $d_X(i, j)$.

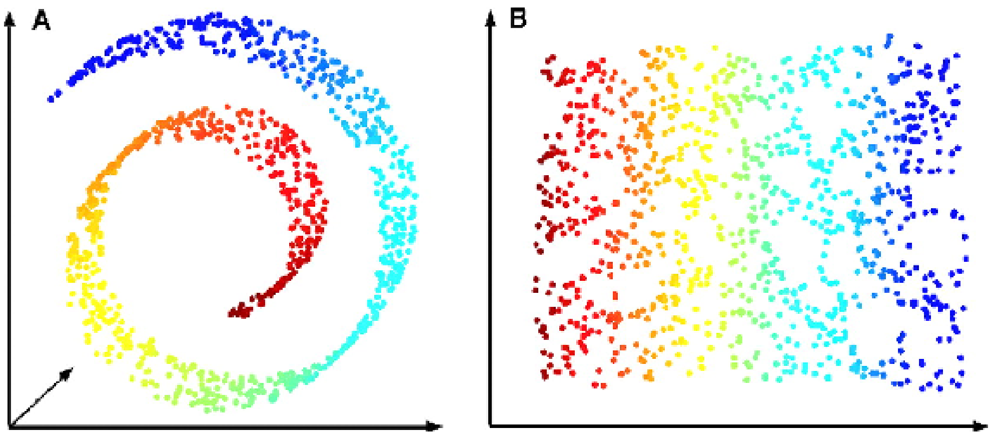


Figure 2.7: The effect of using Isomap on the “Swiss roll” dataset ($\varepsilon = 5$) [8]

Once the previous step has been completed, the shortest paths between neighbours can be calculated. If points i and j are linked then $d_G(i,j) = d_X(i,j)$, otherwise $d_G(i,j) = \infty$. Replace all entries in $d_G(i,j)$ by $min\{d_G(i,j), d_G(i,k) + d_G(k,j)\}$ for every $k = 1,\ldots,N$. The resulting matrix $D_G = \{d_G(i,j)\}$ contains the shortest path distances between all pairs of points in G. Finally, the d-dimensional embedding is constructed. This is achieved by applying classical MDS to the matrix of graph distances, D_G, such that the intrinsic geometry of the manifold is best preserved. The algorithm terminates and outputs the coordinate vectors in a d-dimensional Euclidean space, with $d << D$, the dimensionality of the original dataset.

The success of Isomap depends on being able to choose a neighborhood size (either ε or K) that is neither so large that it introduces “short-circuit” edges into the neighborhood graph, nor so small that the graph becomes too sparse to approximate geodesic paths accurately. Short-circuit edges occur where there are links between data points that are not near each other geodesically and can lead to low-dimensional embeddings that do not preserve a manifold’s true topology. This can be seen in figure 2.8, where the neighbourhood value, ε is the same but with zero-mean normally distributed noise added to the coordinates of each point. The resulting embedding contains obvious folds - the topology of the original solution has not been preserved. However, if a slightly smaller neighborhood size is chosen ($\varepsilon = 3.5$ to 4.6) then the original topology is obtained. The choice of neighbourhood size is an obvious limitation of this approach, but

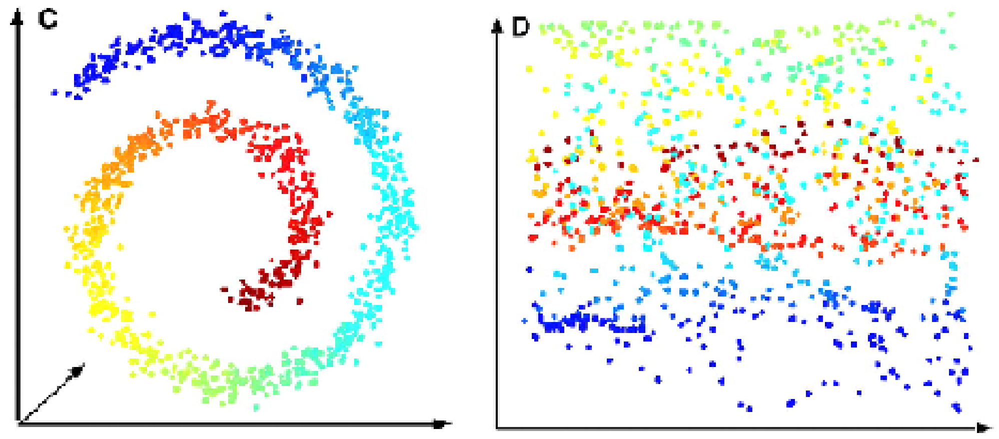


Figure 2.8: The “Swiss roll” dataset with zero-mean normally distributed noise and the resulting Isomap embedding ($\varepsilon = 5$) [8]

may be determined experimentally.

2.1.2.2 Locally Linear Embedding

Locally Linear Embedding (LLE) [141] is an eigenvector method for the problem of nonlinear DR. It calculates low dimensional neighbourhood-preserving reconstructions (embeddings) of data of high dimensionality. LLE achieves this by exploiting the local symmetries of linear reconstructions. To conceptualise this, consider the following informal analogy. The initial data is 3-dimensional and forms the topology of a 2-dimensional rectangular manifold bent into a 3-dimensional S-curve. Scissors are then used to cut this manifold into small squares representing locally linear patches of the nonlinear surface. These squares can then be arranged onto a flat tabletop whilst angular relationships between neighbouring squares are maintained. This is a linear mapping due to the fact that all transformations involve translation, scaling or rotation only. In this way, the algorithm has identified nonlinear structure through a series of linear steps.

The first step of LLE is to select neighbours for each data point. Provided there is sufficient data, it is expected that each data point and its neighbours will lie on or be close to a locally linear patch of the manifold. This selection can be achieved

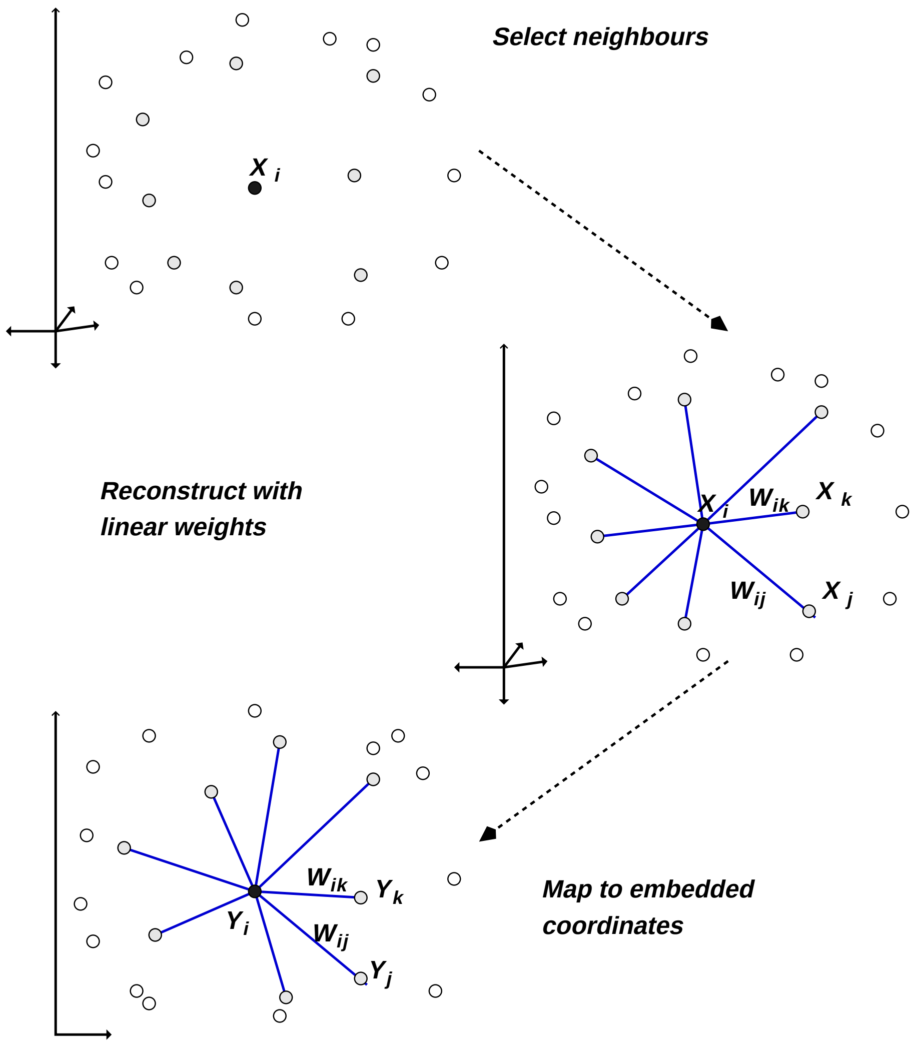


Figure 2.9: Overview of the steps involved in locally linear embedding

using the K nearest neighbours (using the Euclidean distance metric) or by choosing all points within a ball of fixed radius [57]. In figure 2.9, neighbours of the point X_i are highlighted. The second step is to compute the weights that best linearly reconstruct each data point from its neighbours by solving a least squares problem. The following cost function is minimized:

$$E_1(W) = \sum_{i=1}^{N} |(X_i - \sum_{j=1}^{N} W_{ij}X_j)|^2 \tag{2.3}$$

In the diagram, the weights of two of X_i's neighbours, X_j and X_k, are shown; namely W_{ij} and W_{ik}. Finally, the low dimensional embedding vectors are computed, reconstructed from the weights by minimizing the embedded cost function:

$$E_2(Y) = \sum_{i=1}^{N} |(Y_i - \sum_{j=1}^{N} W_{ij} Y_j)|^2 \tag{2.4}$$

The diagram shows the new lower dimensional data points Y_i, Y_j and Y_k.

LLE avoids the need to solve large dynamic programming problems. It also tends to accumulate very sparse matrices whose structure can be exploited to save time and space. However, in [141] there is no indication as to how a test data point may be mapped from the input space to the manifold space, or how a data point may be reconstructed from its low-dimensional representation. Additionally, LLE suffers from the problem of short-circuit edges as described previously for Isomap.

2.1.3 Function Approximation

Multivariate Adaptive Regression Splines (MARS) [54] is an implementation of techniques for solving regression-type problems, with the main purpose of predicting the values of a continuous decision feature from a set of conditional features. It is a non-parametric regression procedure that makes no assumptions about the underlying functional relationships. Instead, MARS constructs this relation from a set of coefficients and basis functions, defined by control points, that are selected based on the data only. The basis functions take the form:

$$(x-t)_+ = \begin{cases} x-t & x > t \\ 0 & otherwise \end{cases} \tag{2.5}$$

Parameter t is the control point of a basis function, determined from the data. Through the determination of control points, MARS attempts to approximate the shape of the underlying data hyperplane, illustrated in figure 2.10.

The general model equation of MARS is:

$$y = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(x) \tag{2.6}$$

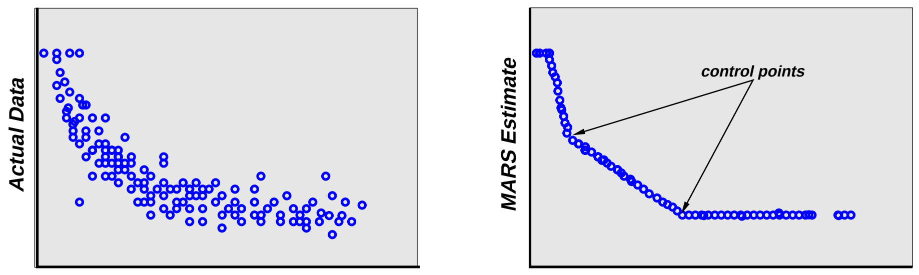


Figure 2.10: MARS data estimation using control points

The summation is over the M terms in the model, y is predicted as a function of the predictor variables x and their interactions. This function consists of an intercept parameter β_0 and the weighted (by β_m) sum of functions $h_m(X)$:

$$h_m(X) = \prod_{l=1}^{z_m} (x_{d_{ml}} - t_{ml})_{q_{ml}} \tag{2.7}$$

where z_m is the interaction level (or order) of the mth spline, d_{ml} indicates which of the p predictor variables enters into the lth interaction of the mth spline, $d_{ml} \in \{1,\ldots,p\}$, t_{ml} is a spline control point, and q_{ml} determines the orientation of the spline interactions, $q_{ml} \in \{+,-\}$.

The variables, interactions and locations of the control points are all found by a brute force approach and the regression coefficients are determined by a least squares procedure. The general MARS algorithm is as follows:

1. Begin with the simplest model involving only the constant basis function.

2. Search the space of basis functions, for each variable and for all possible control points, and add those which maximize a certain measure of goodness of fit (e.g. minimisation of the prediction error).

3. Step 2 is recursively applied until a model of pre-determined maximum complexity is derived.

4. Finally, in the last stage, a pruning procedure is applied where those basis functions are removed that contribute least to the overall (least squares) goodness of fit.

However, this is a relatively complex process, and suffers from the curse of dimensionality. Each dimension of the hyperplane requires one dimension for the approximation model, and an increase in the time and space required to compute and store the splines. The time required to perform predictions increases exponentially with the number of dimensions. Noise may also distort the model by causing MARS to generate a much more complex model as it tries to incorporate the noisy data into its approximation.

Also worth mentioning are methods based on artificial neural networks (ANNs). ANNs are mathematical models that are inspired by the connections and the functioning of neurons in biological systems, based on the topology of nodes and connections between them, and transfer functions which relate the input and output of each node. ANNs are often used as a way of optimizing a classification (or pattern recognition) procedure. They also usually have more input than output nodes; they may thus also be viewed as performing a dimensionality reduction on input data, in a way more general than principal component analysis and multidimensional scaling [18, 89, 93].

2.2 Selection-based Reduction

Whereas semantics-destroying dimensionality reduction techniques irreversibly transform data, semantics-preserving DR techniques (referred to as feature selection) attempt to retain the meaning of the original feature set. The main aim of feature selection (FS) is to determine a minimal feature subset from a problem domain while retaining a suitably high accuracy in representing the original features. In many real world problems FS is a must due to the abundance of noisy, irrelevant or misleading features. For instance, by removing these factors, learning from data techniques can benefit greatly. A detailed review of feature selection techniques devised for classification tasks can be found in [33].

The usefulness of a feature or feature subset is determined by both its *relevancy* and *redundancy*. A feature is said to be relevant if it is predictive of the decision feature(s), otherwise it is irrelevant. A feature is considered to be redundant if it is highly correlated with other features. Hence, the search for a good feature subset involves finding those features that are highly correlated with the decision feature(s), but are uncorrelated with each other.

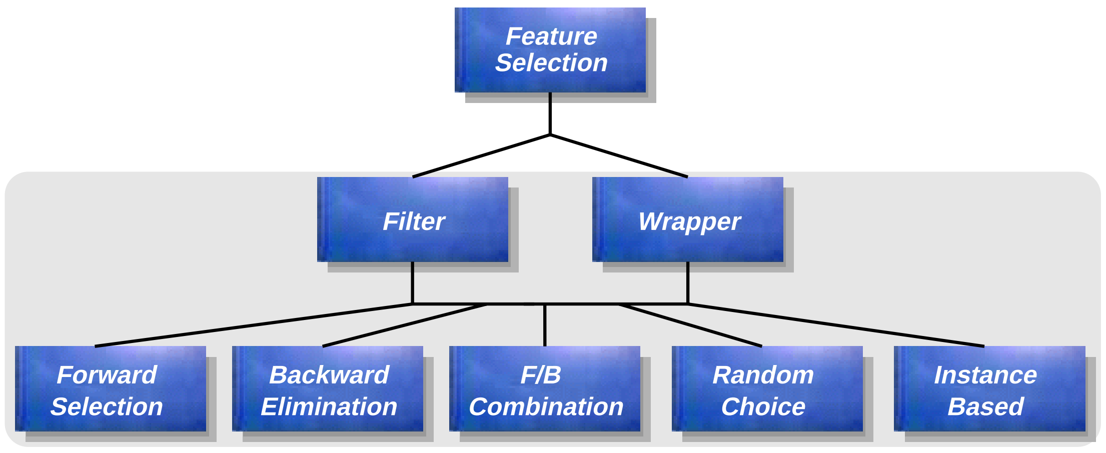


Figure 2.11: Aspects of feature selection

A taxonomy of feature selection approaches can be seen in figure 2.11. Given a feature set size *n*, the task of FS can be seen as a search for an "optimal" feature subset through the competing 2^n candidate subsets. The definition of what an optimal subset is may vary depending on the problem to be solved. Although an exhaustive method may be used for this purpose in theory, this is quite impractical for most datasets. Usually FS algorithms involve heuristic or random search strategies in an attempt to avoid this prohibitive complexity. However, the degree of optimality of the final feature subset is often reduced. The overall procedure for any feature selection method is given in figure 2.12.

The generation procedure implements a search method [95, 161] that generates subsets of features for evaluation. It may start with no features, all features, a selected feature set or some random feature subset. Those methods that start with an initial subset usually select these features heuristically beforehand. Features are added (*forward selection*) or removed (*backward elimination*) iteratively in the first two cases [33]. In the last case, features are either iteratively added or removed or produced randomly thereafter. An alternative selection strategy is to select instances and examine differences in their features. The evaluation function calculates the suitability of a feature

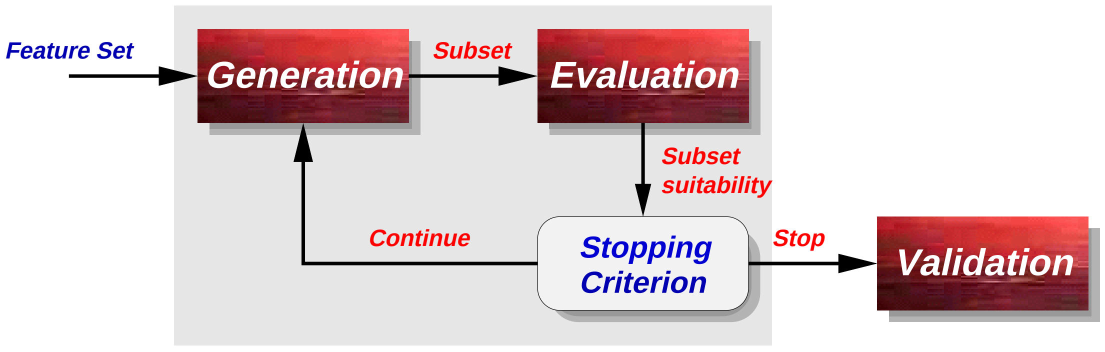


Figure 2.12: Feature Selection

subset produced by the generation procedure and compares this with the previous best candidate, replacing it if found to be better.

A stopping criterion is tested every iteration to determine whether the FS process should continue or not. For example, such a criterion may be to halt the FS process when a certain number of features have been selected if based on the generation process. A typical stopping criterion centred on the evaluation procedure is to halt the process when an optimal subset is reached. Once the stopping criterion has been satisfied, the loop terminates. For use, the resulting subset of features may be validated.

Determining subset optimality is a challenging problem. There is always a trade-off in non-exhaustive techniques between subset minimality and subset suitability - the task is to decide which of these must suffer in order to benefit the other. For some domains (particularly where it is costly or impractical to monitor many features), it is much more desirable to have a smaller, less accurate feature subset. In other areas it may be the case that the modelling accuracy (e.g. the classification rate) using the selected features must be extremely high, at the expense of a non-minimal set of features.

Feature selection algorithms may be classified into two categories based on their evaluation procedure. If an algorithm performs FS independently of any learning algorithm (i.e. it is a completely separate preprocessor), then it is a *filter* approach. In effect, irrelevant attributes are filtered out before induction. Filters tend to be applicable to most domains as they are not tied to any particular induction algorithm.

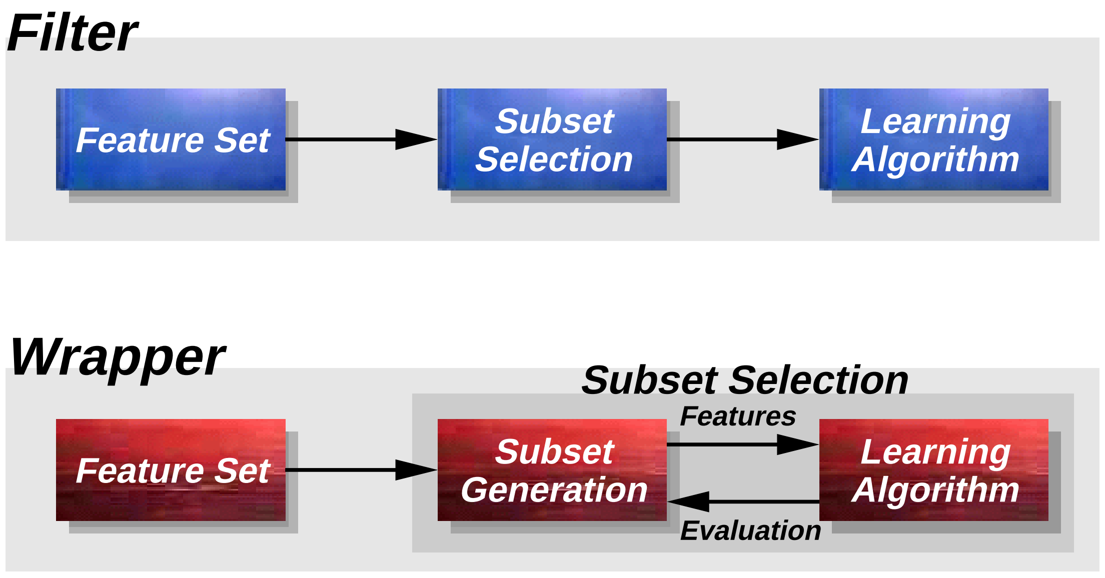


Figure 2.13: Filter and wrapper approaches to Feature Selection

If the evaluation procedure is tied to the task (e.g. classification) of the learning algorithm, the FS algorithm employs the *wrapper* approach. This method searches through the feature subset space using the estimated accuracy from an induction algorithm as a measure of subset suitability. Although wrappers may produce better results, they are expensive to run and can break down with very large numbers of features. This is due to the use of learning algorithms in the evaluation of subsets, some of which can encounter problems when dealing with large datasets.

2.2.1 Filter Methods

To illustrate the operation of the algorithms outlined in this section, an example dataset as given in table 2.2 will be used. The dataset is restricted to containing only binary values due to the different requirements of the algorithms. Throughout the text, $\mathbb{C}$ indicates the set of conditional features, whilst $\mathbb{D}$ denotes the set of decision features.

Object	a	b	c	d	e	f	$\Rightarrow$	g
0	0	1	1	1	0	0		1
1	1	0	1	1	0	0		1
2	1	1	0	0	0	0		1
3	1	1	0	1	0	1		1
4	1	1	1	0	1	0		1
5	0	0	1	1	0	0		1
6	0	0	0	1	1	1		0
7	0	0	1	0	0	1		0
8	1	0	0	0	1	1		0
9	1	0	1	0	0	1		0
10	0	1	0	0	0	1		0
11	0	1	0	1	1	1		0
12	0	1	1	0	1	0		0

Table 2.2: Example 2-class dataset

2.2.1.1 RELIEF

The first feature selection algorithms were based on the filter approach. In RELIEF [83] each feature is given a relevance weighting that reflects its ability to discern between decision class labels. An overview of this algorithm can be found in figure 2.14. The first threshold, *its* specifies the number of sampled objects used for constructing the weights. For each sampling, an object *x* is randomly chosen, and its nearHit and nearMiss are calculated. These are *x*'s nearest objects with the same class label and different class label respectively. The distance between object *o* and *x* is defined here as the sum of the number of features that differ in value between them:

$$dist(o,x) = \sum_{i=1}^{|\mathbb{C}|} diff(o_i, x_i) \tag{2.8}$$

where

$$diff(o_i, x_i) = \begin{cases} 1, & o_i \neq x_i \\ 0, & o_i = x_i \end{cases} \tag{2.9}$$

The user must supply a threshold which determines the level of relevance that features must surpass in order to be finally chosen. This method is ineffective at removing

RELIEF(O, c, its, ε).
O, the set of all objects; c, the number of conditional features;
its, the number of iterations; ε, weight threshold value.

```
(1)  $R \leftarrow \{\}$
(2)  $\forall W_a, W_a \leftarrow 0$
(3)  **for** $i = 1...its$
(4)     choose an object $x$ in $O$ randomly
(5)     calculate $x$'s nearHit and nearMiss
(6)        **for** $j = 1...c$
(7)           $W_j \leftarrow W_j -$ diff$(x_j, nearHit_j)/its$ + diff$(x_j, nearMiss_j)/its$
(8)  **for** $j = 1...c$
(9)     **if** $W_j \geq \varepsilon$; $R \leftarrow R \cup \{j\}$
(10) **return** $R$
```

Figure 2.14: The RELIEF Algorithm

redundant features as two predictive but highly correlated features are both likely to be given high relevance weightings. The method has been extended to enable it to handle inconsistency, noise and multi-class datasets [87].

RELIEF is applied to the example dataset in the following way. An object is chosen randomly, say object 0, and its nearest neighbours are found, nearHit and nearMiss. In this case, object 5 is the nearHit and object 12 is the nearMiss. For each feature, the weight is updated according to the difference of the feature and that of x's nearHit and nearMiss. This process continues until the desired number of iterations have elapsed. Features are then added to the final subset if their weights surpass the desired level, ε. Running RELIEF on the dataset with its = 100 and ε = 0 produces the final subset $\{a, d, e, f\}$. Removing from the dataset all but these attributes will result in a smaller yet still consistent set of data. However, upon examination it can be seen that a further reduction can take place; no inconsistencies are introduced by eliminating feature e from this newly reduced dataset.

2.2.1.2 FOCUS

FOCUS [1], another filter method, conducts a breadth-first search of all feature subsets to determine the minimal set of features that can provide a consistent labelling of the

```
FOCUS($O$, $c$).
$O$, the set of all objects;
$c$, the number of conditional features;
(1)  $R \leftarrow \{\}$
(2)  **for** $num = 1...c$
(3)      **for** each subset $L$ of size $num$
(4)          $cons$ = determineConsistency($L, O$)
(5)          **if** $cons$ == $true$
(6)              $R \leftarrow L$
(7)              **return** $R$
(8)          **else** continue
```

Figure 2.15: The FOCUS Algorithm

training data. The FOCUS algorithm, as summarised in figure 2.15, generates all subsets of the current size (initially one) and checks each for at least one inconsistency. If an inconsistency is found, then that particular subset is removed. This continues until a consistent subset is found or all possible subsets have been evaluated. The consistency criterion makes FOCUS very sensitive to noise or inconsistencies in the training data. Moreover, the exponential growth of the size of the power set of features makes this impractical for domains with medium to large dimensionality. By introducing a threshold value to line (5) in the algorithm, the sensitivity can be reduced, allowing a certain amount of inconsistency within the dataset.

Given the example dataset, FOCUS first evaluates the consistency of all subsets of size 1, namely $\{a\},\{b\},\{c\},\{d\},\{e\},\{f\}$. It determines that none of these subsets produces a reduced dataset with zero inconsistency. For example, selecting the subset $\{f\}$ results in objects 0 and 12 conflicting. Again, all subsets of size 2 are examined (e.g. $\{a,b\},\{a,c\}$, etc.) and no suitable subset is found. The algorithm continues until the subset $\{a,d,f\}$ is chosen - this will result in no inconsistencies. Hence, the dataset can now be reduced to one only involving these attributes. This subset of features is the minimal subset for this dataset in terms of the consistency criterion.

LVF(O, $\mathbb{C}$, att, ε).
O, the set of all objects; $\mathbb{C}$, the set of conditional features;
att, the number of iterations of the algorithm; ε, consistency threshold.

```
(1) $R \leftarrow \mathbb{C}$
(2) **for** $num = 1...att$
(3)    $S \leftarrow$ randomFeatureSubset()
(4)    **if** $|S| \leq |R|$
(5)       **if** inconsistency$(S, O) \leq \varepsilon$
(6)          **if** $|S| < |R|$
(7)             $R \leftarrow S$; **output** $R$
(8)          **else** $R \leftarrow R \cup S$
(9) **return** $R$
```

Figure 2.16: The LVF Algorithm

2.2.1.3 LVF

LVF employs an alternative generation procedure - that of choosing random feature subsets, accomplished by the use of a Las Vegas algorithm [102]. An outline of LVF is given in figure 2.16. Initially, the best feature subset is considered to be the entire conditional feature set. A feature subset is randomly chosen; if the subset has a smaller cardinality than the current best and its inconsistency rate is less than or equal to a threshold, ε, it is considered to be the new best subset. Here, the inconsistency count is defined as the sum of the number of inconsistent objects minus the number of inconsistent objects with the most frequent class label. For example, say there are n inconsistent objects for a 2-class dataset, c_1 such objects belong to class 1, c_2 to class 2, so $c_1 + c_2 = n$. If the largest of these is, for example, c_1 then the inconsistency count will be $n - c_1$. The inconsistency rate is simply the sum of all inconsistency counts divided by the number of objects. Every time a better subset is encountered, it is outputted. A problem with this approach is that it will tend to take longer to locate an optimal subset than algorithms that employ heuristic generation procedures. Additionally, when datasets are huge, checking the consistency of the dataset takes a long time.

Returning to the example, LVF randomly chooses feature subsets from among the six features present, e.g. the subset $\{a, b, c\}$. For class 1 there are 3 inconsistent objects, for class 2 there are also 3 so the inconsistency count will be $(6-3)/12 = \frac{1}{4}$. If no inconsistency is allowed ($\varepsilon = 0$), then this subset will not be kept. Given sufficient time, the algorithm should eventually reach the subset $\{a, d, f\}$ which is the smallest subset that produces the required inconsistency rate.

2.2.1.4 SCRAP

Selection Construction Ranking using Attribute Pattern (SCRAP) [137] is an instance-based filter, which determines feature relevance by performing a sequential search within the instance space. SCRAP considers objects (instances) one at a time, instead of typical forward or backward search. The central idea is to identify those features that change at decision boundaries in the data table - these are assumed to be the most informative. An algorithmic overview may be seen in figure 2.17. A sequential search is conducted starting from a random object, which becomes the first point of class change (PoC). The nearest object to this with a different class label becomes the next PoC. These two PoCs define a neighbourhood; features that change between them define the dimensionality of decision boundary between the two classes. If there is only one such feature that changes, this is determined to be absolutely relevant and is included in the feature subset. If more than one feature changes, their associated relevance weights, initially zero, are incremented. If objects of the same class label are closer than this new PoC and differ in only one feature, then that feature's weight is decremented. Objects determined to belong to neighbourhoods are then removed from processing. The process stops when all objects have been assigned to a neighbourhood. Features that have a positive relevance weight and those that have been determined to be absolutely relevant are chosen as the final feature subset.

From the example dataset, SCRAP first chooses a random object, say object 1, and proceeds to find its nearest neighbour with a different class label. In this case, object 12 is the PoC with two features that differ, d and e. These are said to be weakly relevant and their weights (initially zero) are incremented. Those objects of a lesser distance away than object 12 with the same label as object 1 are assigned to this neighbourhood. Only object 5 is closer as it differs in one feature, b. This results in b's weight being decremented. If more than one feature differed here, the weights would not have been

```
SCRAP($O$).
$O$, the set of all objects;
(1)  $A \leftarrow \{\}; \forall W_i, W_i = 0;$
(2)  $T \leftarrow$ randomObject(); $PoC \leftarrow T$
(3)  while $O \neq \{\}$
(4)      $O \leftarrow O - PoC$; $PoC_{new} \leftarrow$ NewPoC($PoC$)
(5)      $n =$ dist($PoC$,$PoC_{new}$)
(6)      if $n == 1$
(7)          $i$ = diffFeature($PoC$,$X$); $A \leftarrow A \cup \{i\}$
(8)      $N \leftarrow$ getClosestNeighbours($PoC$,$n$)
(9)      $\forall X \in N$
(10)         if classLabel($X$) == classLabel($N$)
(11)             $O \leftarrow O - X$
(12)             if dist($PoC$,$X$)==1
(13)                 $i$ = diffFeature($PoC$,$X$); $W_i = W_i - 1$
(14)         else if dist($PoC$,$X$) $> 1$
(15)             incrementDifferingFeatures($X$,$W$)
(16)  $R \leftarrow A$
(17)  $\forall W_i$, if $W_i > 0$ then $R \leftarrow R \cup \{ i\}$
```

Figure 2.17: The SCRAP Algorithm

affected - only those cases where one feature differs result in weights being reduced. Object 12 now becomes the new PoC and the algorithm continues. Object 4 is the nearest neighbour with a different class label, with only one feature differing in value, feature a. This feature is determined to be absolutely relevant and is added to the final subset, irrespective of its final relevance weight. When the algorithm eventually terminates, the subset $\{a,b,c,d,e,f\}$ is returned: a is absolutely relevant, the rest have a positive final weight. No reduction of this dataset is achieved.

The above example serves to illustrate one of the main weaknesses of this approach - it regularly chooses too many features. This is due, in part, to the situation where weights are decremented. If more than one feature changes between a PoC and an object of the same class label then the corresponding feature weights remain unaffected. This drawback could be tackled by reducing the weights of these features when this occurs. With this modification in place and running the algorithm on the dataset, a smaller subset, $\{a,b,e,f\}$ is obtained. Another alteration may be to decrement each weight proportionally, e.g. for three irrelevant features, their weights will be reduced by one third each. This combined with a similar modification for incrementing weights may produce a more accurate reflection of a feature's importance within a dataset. Currently, SCRAP will only handle nominal values, although it is relatively straightforward to extend this to continuously valued features.

2.2.1.5 EBR

EBR($\mathbb{C}$).
$\mathbb{C}$, the set of all conditional features;

```
(1) $R \leftarrow \{\}$
(2) **do**
(3)     $T \leftarrow R$
(4)     $\forall x \in (\mathbb{C} - R)$
(5)         **if** $H(R \cup \{x\}) < H(T)$
(6)             $T \leftarrow R \cup \{x\}$
(7)     $R \leftarrow T$
(8) **until** $H(R) == H(\mathbb{C})$
(9) **return** $R$
```

Figure 2.18: The Entropy-based Algorithm

A further technique for filter-based feature selection is entropy-based reduction (EBR), developed from work carried out in [71]. This approach is based on the entropy heuristic employed by machine learning techniques such as C4.5 [135]. A similar approach has been adopted in [33] where an entropy measure is used for ranking features. EBR is concerned with examining a dataset and determining those attributes that provide the most gain in information. The entropy of attribute A (which can take values $a_1 ... a_m$) with respect to the conclusion C (of possible values $c_1 ... c_n$) is defined as:

$$H(A) = -\sum_{j=1}^{m} p(a_j) \sum_{i=1}^{n} p(c_i|a_j)\, log_2\, p(c_i|a_j) \tag{2.10}$$

This can be extended to dealing with *subsets* of attributes instead of individual attributes only. Using this entropy measure, the algorithm used in rough set-based attribute reduction [29] can be modified to that shown in figure 2.18. This algorithm requires no thresholds in order to function - the search for the best feature subset is stopped when the resulting subset entropy is equal to that of the entire feature set. For consistent data, the final entropy of the subset will be zero. It is interesting to note that any subset with an entropy of 0 will also have a corresponding rough set dependency of 1.

Returning to the example dataset, EBR first evaluates the entropy of each individual attribute:

Subset	Entropy
$\{a\}$	0.8885861
$\{b\}$	0.9543363
$\{c\}$	0.9543363
$\{d\}$	0.8885860
$\{e\}$	0.8650087
$\{f\}$	0.6186034

The subset with the lowest entropy here is $\{f\}$ so this is added to the current feature subset. The next step is to calculate the entropy of all subsets containing f and one other attribute:

Subset	Entropy
$\{a, f\}$	0.42382884
$\{b, f\}$	0.46153846
$\{c, f\}$	0.55532930
$\{d, f\}$	0.42382884
$\{e, f\}$	0.40347020

Here, the subset $\{e, f\}$ is chosen. This process continues until the lowest entropy for the dataset is achieved (for a consistent dataset this is zero). The algorithm eventually reaches this lowest value when it encounters the feature subset $\{a,b,e,f\}$ ($H(\{a,b,e,f\})$ $= 0$). The dataset can now be reduced to these features only. As has been shown previously, this is close to the best feature subset for this dataset. The optimal subset was discovered by the FOCUS algorithm which works well for small datasets such as this, but cannot be applied to datasets of medium to large dimensionality. EBR does not suffer from this problem as its complexity is $O((n^2 + n)/2)$. It also does not require any user-defined thresholds in order to function, a drawback of RELIEF.

2.2.1.6 FDR

Fractal Dimension Reduction (FDR) [181] is a novel approach to feature selection based on the concept of fractals - the self-similarity exhibited by data on different scales. For example, the Sierpinski triangle [58] is constructed from an equilateral triangle, eliminating its middle triangle and repeating this process on the resulting smaller triangles. This continues infinitely, removing the middle triangles from the newly generated smaller ones. The resulting shape is not one-dimensional, however it is not two-dimensional either as its area is zero so its intrinsic dimension lies between 1 and 2. This issue is resolved by considering fractional dimensionalities. For the Sierpinski triangle its *fractal* dimension is approximately 1.58 [150].

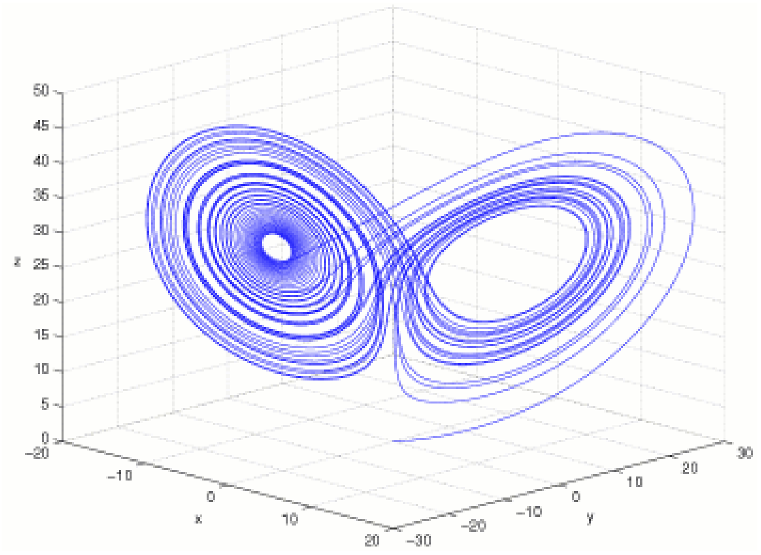


Figure 2.19: The Lorenz attractor

This concept may be applied to feature selection by considering the change in fractal dimension when certain features are removed from the original dataset under consideration. For the lorenz dataset in figure 2.19, the fractal dimension is approximately 2.05 [150]. It is clear from the figure that most of the data lie in a two-dimensional plane, the third dimension is effectively redundant. Removing this feature from the dataset will result in a slight, but acceptable reduction in the fractal dimension. This forms the basis of the work carried out in [181]. The feature selection algorithm based on this can be seen in figure 2.20. Given a dataset, the correlation fractal dimension is calculated and attributes are removed via backward elimination until the removal of any further attribute reduces the fractal dimension too much.

```
FDR($\mathbb{C}$,$\varepsilon$).
$\mathbb{C}$, the set of all conditional features;
$\varepsilon$, the allowed reduction in fractal dimension;
(1)  $R \leftarrow \mathbb{C}$; $bestDim = 0$; $d_c$ = calculateFractalDim($\mathbb{C}$)
(2)  do
(3)      $T \leftarrow R$,
(4)      $\forall x \in R$
(5)          $S \leftarrow R - \{x\}$
(6)          $d_s$ = calculateFractalDim($S$)
(7)          if $d_s > bestDim$
(8)              $bestDim = d_s$; $w \leftarrow x$
(9)      $R \leftarrow R - \{w\}$
(10) until $d_c - bestDim > \varepsilon$
(11) return $R$
```

Figure 2.20: Fractal Dimension Reduction

One problem with this approach is that the calculation of the correlation fractal dimension requires datasets containing a large number of objects. If a dataset has too few instances, the procedure for calculating the fractal dimension will not be able to estimate the dimension of the dataset effectively. Another weakness of FDR is how to estimate the extent of the allowed reduction in fractal dimensionality, ε. It is not clear how to determine this value beforehand. The fractal dimension of the full set of features is often used as an additional stopping criterion if the number of remaining features falls below this value (the intrinsic dimensionality of the dataset).

2.2.1.7 Feature Grouping

Typically in feature selection, the generation procedure incrementally adds or removes individual features. Recently, there have been a couple of investigations into the potential utility of *grouping* features at each stage. This strategy can decrease the time taken in finding optimal subsets by selecting several features at once.

An automatic feature grouping technique is proposed in [195] that uses k-means clustering [60] beforehand in order to generate groups. From these groups, one or two features are pre-selected for a typical forward search FS method. No feature grouping takes place during the search itself however. As yet, no results are available for this approach.

In Group-wise Feature Selection (GFS) [122], feature groups are again calculated beforehand. These groups are then used throughout the subset search instead of considering individual features. An overview of the algorithm can be seen in figure 2.21. Here, the effect of adding groups of features to the currently considered subset is evaluated at each stage. The group that produces the largest increase in performance is selected and all features present within the group are added to the current subset. The process continues until the performance is of an appropriate quality.

```
GFS($G$,$\varepsilon$).
$G$, the set of feature groups;
$\varepsilon$, required level of subset performance

(1)  $R \leftarrow \{\}$; $A \leftarrow \{\}$; $best = 0$
(2)    while evaluate($R$) $< \varepsilon$
(3)      for each group $G_i$
(4)        $T \leftarrow$ all features from $G_i$
(5)        $\varepsilon_t =$ evaluate($R \cup T$)
(6)        if ($\varepsilon > best$)
(7)          $A \leftarrow T$
(8)          $best = \varepsilon_t$
(9)      $R \leftarrow R \cup A$
(10) output $R$
```

Figure 2.21: The Group-wise FS Algorithm

In the evaluation, a measurement cost is also considered though this can be omitted if no measurement information is available or required. In addition to the GFS algorithm presented in figure 2.21, an extension to it, GNFS, that performs a nested forward search within groups has also been proposed. Instead of selecting the best group, GNFS searches for the best subset within a group and adds this to the currently selected feature subset. Both algorithms perform comparably with their standard individual feature selection method, but with the benefit of reduced computation time.

All feature grouping approaches so far have relied on groups being defined beforehand. Group membership is not changed in the selection process leading to a dependence on a suitably accurate grouping mechanism for good results. In addition to this, the *extent* of group membership is not considered at all; features either belong or do not belong to single groups. Fuzzy grouping can be used to handle this problem so that features can belong to more than one group with varying degrees of membership. This additional membership information can then be used in the selection process itself. These ideas motivated the development of the new rough and fuzzy-rough set-based grouping FS technique, detailed in section 5.1.

2.2.1.8 Other Approaches

In addition to those approaches outlined above, a filter method based on ideas from probabilistic reasoning and information theory is proposed in [85]. The central motivation behind this development is the observation that the goal of an induction algorithm is to estimate the probability distributions over the class values. In the same way, feature subset selection should attempt to remain as close as possible to these original distributions. The algorithm performs a backward elimination search, at each stage removing the feature that causes the least change between the distributions. The search stops when the desired number of features remain (specified by the user). An important problem with this method is that it requires the features in a dataset to be binary-valued. This constraint is added to avoid the bias toward many-valued features present in entropy-based measures.

Also worth mentioning is the Chi2 algorithm [101]. This is in effect a heuristic feature selector that discretizes continuous features and in the process removes irrelevant ones based on the χ^2 statistic [127]. In fact, feature selection takes place only as a side-effect of the discretization process; if a feature ends up with all its values mapped to a single discrete value, then it can be removed from the dataset without introduction of inconsistency.

2.2.2 Wrapper Methods

The LVW algorithm [103] is a wrapper method based on the earlier LVF algorithm [102] (described in section 2.2.1.3) and can be outlined as shown in figure 2.22. This again uses a Las Vegas style of random subset creation which guarantees that given enough time, the optimal solution will be found. As with LVF, LVW produces intermediate solutions while working toward better ones that result in a lower classification error. The algorithm requires two threshold values to be supplied; ε, the classification error threshold and the value K, used to determine when to exit the algorithm due to there being no recent updates to the best subset encountered so far.

LVW($\mathbb{C}$, K, ε).
$\mathbb{C}$, the set of conditional features;
K, update threshold; ε, error threshold.

```
(1) $R \leftarrow \mathbb{C}$; $k = 0$
(2) **while** $\varepsilon$ not updated for $K$ times
(3)     $T \leftarrow$ randomFeatureSubset()
(4)     $\varepsilon_t = \text{learn}(T)$
(5)     **if** ($\varepsilon_t < \varepsilon$) **or** ($\varepsilon_t == \varepsilon$ **and** $|T| < |R|$)
(6)         **output** $T$
(7)         $k = 0$; $\varepsilon = \varepsilon_t$; $R \leftarrow T$
(8)     $k = k + 1$
(9) $\varepsilon = \text{learn}(R)$
```

Figure 2.22: The LVW Algorithm

Initially, the full set of conditional features are considered to be the best subset. The algorithm continues to generate random subsets and evaluates them using an inductive learning algorithm until no better subsets are encountered for a given number of attempts (the K criterion). Finally, training and testing is carried out on the resulting best feature subset. In the reported experimentation, C4.5 [135] is used as the learning algorithm due to its relatively fast induction time (a critical factor in designing wrapper-based systems).

```
NEURALNET($\mathbb{C}$,$\varepsilon_{max}$).
$\mathbb{C}$, the set of all conditional features;
$\varepsilon_{max}$, network classification error threshold
(1)  $R \leftarrow \mathbb{C}$; $\varepsilon_w$=0
(2)  do
(3)      $T \leftarrow R$,
(4)      $\forall x \in R$
(5)          $S \leftarrow R - \{x\}$
(6)          $\varepsilon_S$ = trainNet($S$)
(7)          if $\varepsilon_S > \varepsilon_w$
(8)              $\varepsilon_w = \varepsilon_S$; $w \leftarrow x$
(9)      $R \leftarrow R - \{w\}$
(10) until $\varepsilon_R > \varepsilon_{max}$
(11) trainNet($T$)
```

Figure 2.23: Neural network feature selection

In [154], a neural network-based wrapper feature selector is proposed that employs backward elimination in the search for optimal subsets. This algorithm is summarised in figure 2.23. Initially, a 3-layer feedforward network is trained using all features in the dataset. This is then evaluated using an error function which involves both the classification error and a measure of the network complexity. The attribute that gives the largest decrease in network accuracy is removed. This process repeats until no more attributes can be eliminated without exceeding the maximum error threshold.

A wrapper method was proposed in [81] where feature subsets are explored using heuristic search and evaluated using *n*-fold cross validation. The training data is split into *n* partitions and the induction algorithm run *n* times; the standard C4.5 package is used for decision tree induction. For each partition, the algorithm uses $n-1$ partitions for training and the remaining partition for testing. The average of the classification accuracy over all *n* runs is used as the estimated accuracy. The results show that although there is not much difference between the compared filter and wrapper approaches in terms of classification accuracy, the induced decision trees were smaller for the wrapper method in general.

2.2.3 Genetic Approaches

Genetic Algorithms (GAs) [64] are generally quite effective for rapid search of large, nonlinear and poorly understood spaces. Unlike classical feature selection strategies where one solution is optimized, a population of solutions can be modified at the same time [162, 92]. This can result in several optimal (or close-to-optimal) feature subsets as output. Section 8.2 discusses the potential usefulness of this aspect of GAs in feature selection.

A feature subset is typically represented by a binary string with length equal to the number of features present in the dataset. A zero or one in the jth position in the chromosome denotes the absence or presence of the jth feature in this particular subset. The general process for feature selection using GAs can be seen in figure 2.24.

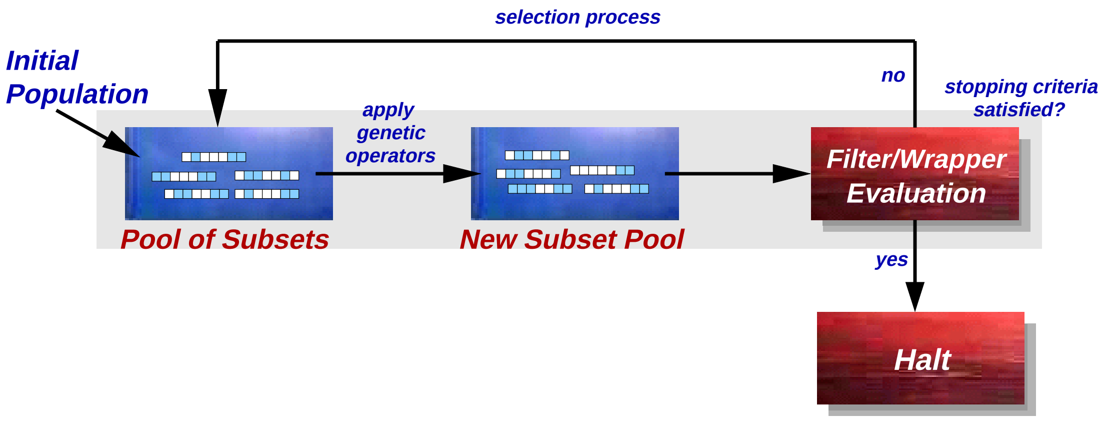


Figure 2.24: Feature Selection with Genetic Algorithms

An initial population of chromosomes is created; the size of the population and how they are created are important issues. From this pool of feature subsets, the typical genetic operators (crossover and mutation) are applied. Again, the choice of which types of crossover and mutation used must be carefully considered, as well as their probabilities of application. This generates a new feature subset pool which may be evaluated in two different ways. If a filter approach is adopted, the fitness of individuals is calculated using a suitable criterion function $J(X)$. This function evaluates the “goodness” of feature subset X; a larger value of J indicates a better feature subset. Such a criterion function could be Shannon’s entropy measure [135] or the dependency function from rough set theory [125].

For the wrapper approach, chromosomes are evaluated by inducing a classifier based on the feature subset, and obtaining the classification accuracy (or an estimate of it) on the data [173]. To guide the search toward minimal feature subsets, the subset size is also incorporated into the fitness function of both filter and wrapper methods. Indeed, other factors may be included that are of interest, such as the cost of measurement for each feature etc. GAs may also learn rules directly, and in the process perform feature selection [31, 80, 191].

A suitable stopping criterion must be chosen. This is typically achieved by limiting the number of generations that take place or by setting some threshold which must be exceeded by the fitness function. If the stopping criterion is not satisfied, then individuals are selected from the current subset pool and the process described above repeats. Among the selection strategies that have been applied are roulette wheel selection [121] and rank-based selection [194, 198]. In roulette wheel selection, the probability of a chromosome being selected is proportional to its fitness. Rank selection sorts all the individuals by fitness and the probability that an individual will be selected is proportional to its rank in this sorted list.

As with all feature selection approaches, GAs can get caught in local minima, missing a dataset's true minimal feature subset. Also, the fitness evaluation can be very costly as there are many generations of many feature subsets that must be evaluated. This is particularly a problem for wrapper approaches where classifiers are induced and evaluated for each chromosome.

2.2.4 Simulated Annealing-based Feature Selection

Annealing is the process by which a substance is heated (usually melted) and cooled slowly in order to toughen and reduce brittleness. For example, this process is used for a metal to reach a configuration of minimum energy (a perfect, regular crystal). If the metal is annealed too quickly, this perfect organisation is unable to be achieved throughout the substance. Parts of the material will be regular, but these will be separated by boundaries where fractures are most likely to occur.

Simulated Annealing (SA) [84] is a stochastic optimization technique that is based on the computational imitation of this process of annealing. It is concerned with the change of energy (cost) of a system. In each algorithmic step, an “atom” (a feature subset in FS) is given a small random displacement and the resulting change of energy, ΔE, is calculated. If $\Delta E \leq 0$, this new state is allowed and the process continues. However if $\Delta E > 0$, the probability that this new state is accepted is:

$$P(\Delta E) = e^{-(\frac{\Delta E}{T})} \tag{2.11}$$

As the temperature, T, is lowered, the probability of accepting a state with a positive change in energy reduces. In other words, the willingness to accept a bad move decreases. The conversion of a combinatorial optimization problem into the SA framework involves the following:

- *Concise configuration description*. The representation of the problem to be solved should be defined in a way that allows solutions to be constructed easily and evaluated quickly.

- *Random move generator*. A suitable random transformation of the current state must be defined. Typically the changes allowed are small to limit the extent of search to the vicinity of the currently considered best solution. If this is not limited, the search degenerates to a random unguided exploration of the search space.

- *Cost function definition*. The cost function (i.e. the calculation of the state's energy) should effectively combine the various criteria that are to be optimized for the problem. This function should be defined in such a way that smaller function values indicate better solutions.

- *Suitable annealing schedule*. As with the real-world annealing process, problems are encountered if the initial temperature is too low, or if annealing takes place too quickly. Hence, an annealing schedule must be defined that avoids these pitfalls. The schedule is usually determined experimentally.

To convert the feature selection task into this framework, a suitable representation must be used. Here, the states will be feature subsets. The random moves can be produced by randomly mutating the current state with a low probability. This may also remove features from a given feature subset, allowing the search to progress both forwards and backwards. The cost function must take into account both the evaluated subset "goodness" (by a filter evaluation function or a wrapper classifier accuracy) and also the subset size. The annealing schedule can be determined by experiment, although a good estimate may be $T(0) = |C|$ and $T(t+1) = \alpha * T(t)$, with $\alpha \geq 0.85$. Here t is the number of iterations and α determines the rate of cooling.

```
SAFS($T_0$, $T_{min}$, $\alpha$, $L_k$).
$T_0$, the initial temperature;
$T_{min}$, the minimum allowed temperature;
$\alpha$, the extent of temperature decrease;
$L_k$, extent of local search

(1)  $R \leftarrow$ genInitSol()
(2)  while $T(t) > T_{min}$
(4)     for i=1...$L_k$
(5)        $S \leftarrow$ genSol(R)
(6)        $\Delta E$ = cost(S)
(7)        if $\Delta E \leq 0$
(8)           $M \leftarrow S$
(9)        else if $P(\Delta E) >$ randNumber()
(10)          $M \leftarrow S$
(11)    $R \leftarrow M$
(12)    $T(t+1) = \alpha * T(t)$
(13) output $R$
```

Figure 2.25: The SAFS Algorithm

The SA-based feature selection algorithm can be seen in figure 2.25. This differs slightly from the general SA algorithm in that there is a measure of local search employed at each iteration, governed by the parameter L_k. An initial solution is created, from which the next states are derived by random mutations and evaluated. The best state is remembered and used for processing in the next cycle. The chosen state may not actually be the best state encountered in this loop, due to the probability $P(\Delta E)$ that a state is chosen randomly (which will decrease over time). The temperature is decreased according to the annealing schedule and the algorithm continues until the lowest allowed temperature has been exceeded.

Problems with this approach include how to define the annealing schedule correctly. If α is too high, the temperature will decrease slowly, allowing more frequent jumps to higher energy states, slowing convergence. However, if α is too low, the temperature decreases too quickly and the system will converge to local minima (equivalent to brittleness in the case of metal annealing). Also, the cost function definition is critical - there must be a balancing of the importance assigned to the different evaluation criteria involved. Biasing one over another will have the effect of directing search toward solutions that optimize that criterion only.

2.3 Summary

This chapter has reviewed the important problem of dimensionality reduction for datasets, with a focus on semantics-preserving reduction or feature selection. This has become a vital step in many areas such as machine learning, pattern recognition and signal processing due to their inability to handle high dimensional descriptions of input features. Additionally, feature selection can provide a better understanding of the underlying concepts within data. It is estimated that every 20 months or so the amount of information in the world doubles, similarly applications for dealing with this vast amount of information must develop. Part of the solution to this must be drastic and efficient dimensionality reduction techniques. The extent of reduction needs to be severe to allow more detailed analysis of the data to take place by future processes. Although less of an issue for dimensionality reduction which usually takes place off-line, efficiency is still an important consideration. As one of the main goals of reduction is to enable further, more complex data processing which would otherwise be intractable, the reduction methods must not themselves be subject to the curse of dimensionality if possible.

Dimensionality reduction may be split into the two disjoint areas: transformation-based and selection-based reduction. Transformation-based approaches reduce dimensional data (which may exhibit linear or nonlinear relationships) by irreversibly transforming the data points, and as a result destroy the original dataset semantics. Feature selection seeks to retain this information by selecting attributes as opposed to transforming them. This aspect is particularly useful when feature selection precedes other processes that require the original feature meanings to be intact, for example rule induction where rules may need to be human-readable. Two general methods for this, filter and wrapper approaches, were shown and examples given.

This chapter proposed an area for research based on an alternative generation procedure, namely simulated annealing-based FS. Hill-climbing techniques often fail to find minimal feature subsets as they can be misled by initially promising features. Later on in the search, these features turn out to require the presence of many other less informative attributes, leading to oversized final subsets. By using stochastic techniques, this problem may be countered by allowing a degree of randomness in the search.

Chapter 3

Rough Set-based Approaches to Feature Selection

Many problems in machine learning involve high dimensional descriptions of input features. It is therefore not surprising that much research has been carried out on dimensionality reduction [33, 83, 95, 104, 111]. However, existing work tends to destroy the underlying semantics of the features after reduction (e.g. transformation-based approaches [38]) or require additional information about the given data set for thresholding (e.g. entropy-based approaches [112]). A technique that can reduce dimensionality using information contained within the dataset and that preserves the meaning of the features (i.e. semantics-preserving) is clearly desirable. Rough set theory (RST) can be used as such a tool to discover data dependencies and to reduce the number of attributes contained in a dataset using the data alone, requiring no additional information [125, 133].

Over the past ten years, RST has indeed become a topic of great interest to researchers and has been applied to many domains. Given a dataset with discretized attribute values, it is possible to find a subset (termed a *reduct*) of the original attributes using RST that are the most informative; all other attributes can be removed from the dataset with minimal information loss. From the dimensionality reduction perspective, informative features are those that are most predictive of the class attribute.

However, it is most often the case that the values of attributes may be both crisp and *real-valued*, and this is where traditional rough set theory encounters a problem. It is not possible in the original theory to say whether two attribute values are similar and to what extent they are the same; for example, two close values may only differ as a result of noise, but in RST they are considered to be as different as two values of a different order of magnitude. As a result of this, extensions to the original theory have been proposed, for example those based on similarity or tolerance relations [166, 172, 175].

It is, therefore, desirable to develop techniques to provide the means of data reduction for crisp and real-value attributed datasets which utilises the extent to which values are similar. This can be achieved through the use of *fuzzy-rough* sets. Fuzzy-rough sets encapsulate the related but distinct concepts of vagueness (for fuzzy sets [200]) and indiscernibility (for rough sets), both of which occur as a result of uncertainty in knowledge [43]. Vagueness arises due to a lack of sharp distinctions or boundaries in the data itself. This is typical of human communication and reasoning. Rough sets can be said to model ambiguity resulting from a lack of information through set approximations.

This chapter focuses on those recent techniques for feature selection that employ a rough-set based methodology for this purpose, highlighting current trends in this promising area. Rough set fundamentals are introduced with a simple example to illustrate its operation. Several extensions to this theory are also presented, which enable alternative approaches to feature selection. Many of these are evaluated experimentally and compared.

3.1 Rough Selection

Rough set theory [46, 124, 155, 167, 168] is an extension of conventional set theory that supports approximations in decision making. It possesses many features in common (to a certain extent) with the Dempster-Shafer theory of evidence [165] and fuzzy set theory [126, 188]. The rough set itself is the approximation of a vague concept (set) by a pair of precise concepts, called lower and upper approximations, which are a classification of the domain of interest into disjoint categories. The lower approximation is a description of the domain objects which are known with certainty to belong to the subset of interest, whereas the upper approximation is a description of the objects which possibly belong to the subset. This section focuses on several rough set-based techniques for feature selection. Some of the techniques described here can be found in rough set systems available online [139, 140].

To illustrate the operation of these, an example dataset (table 3.1) will be used. Here, the table consists of four conditional features (a, b, c, d), one decision feature (e) and eight objects. The task of feature selection here is to choose the smallest subset of these conditional features so that the resulting reduced dataset remains consistent with respect to the decision feature. A dataset is consistent if for every set of objects whose attribute values are the same, the corresponding decision attributes are identical. Throughout this thesis, the terms attribute, feature and variable are used interchangeably.

$x \in \mathbb{U}$	a	b	c	d	$\Rightarrow$	e
0	1	0	2	2		0
1	0	1	1	1		2
2	2	0	0	1		1
3	1	1	0	2		2
4	1	0	2	0		1
5	2	2	0	1		1
6	2	1	1	1		2
7	0	1	1	0		1

Table 3.1: An example dataset

3.1.1 Rough Set Attribute Reduction

Rough Set Attribute Reduction (RSAR) [29] provides a filter-based tool by which knowledge may be extracted from a domain in a concise way; retaining the information content whilst reducing the amount of knowledge involved. The main advantage that rough set analysis has is that it requires no additional parameters to operate other than the supplied data [47]. It works by making use of the granularity structure of the data only. This is a major difference when compared with Dempster-Shafer theory and fuzzy set theory which require probability assignments and membership values respectively. However, this does not mean that *no* model assumptions are made. In fact by using only the given information, the theory assumes that the data is a true and accurate reflection of the real world (which may not be the case). The numerical and other contextual aspects of the data are ignored which may seem to be a significant omission, but keeps model assumptions to a minimum.

3.1.2 Theoretical Background

Central to RSAR is the concept of indiscernibility. Let $I = (\mathbb{U}, \mathbb{A})$ be an information system, where $\mathbb{U}$ is a non-empty set of finite objects (the universe) and $\mathbb{A}$ is a non-empty finite set of attributes such that $a : \mathbb{U} \rightarrow V_a$ for every $a \in \mathbb{A}$. V_a is the set of values that attribute a may take. With any $P \subseteq \mathbb{A}$ there is an associated equivalence relation $IND(P)$:

$$IND(P) = \{(x,y) \in \mathbb{U}^2 \mid \forall a \in P,\ a(x) = a(y)\} \tag{3.1}$$

The partition of $\mathbb{U}$, generated by *IND(P)* is denoted $\mathbb{U}$ */IND(P)* (or $\mathbb{U}$ */P*) and can be calculated as follows:

$$\mathbb{U}/IND(P) = \otimes\{a \in P : \mathbb{U}/IND(\{a\})\}, \tag{3.2}$$

where

$$A \otimes B = \{X \cap Y : \forall X \in A, \forall Y \in B, X \cap Y \neq \emptyset\} \tag{3.3}$$

If $(x,y) \in IND(P)$, then x and y are indiscernible by attributes from P. The equivalence classes of the P-indiscernibility relation are denoted $[x]_P$. For the illustrative example, if $P = \{b,c\}$, then objects 1, 6 and 7 are indiscernible; as are objects 0 and 4. *IND(P)* creates the following partition of $\mathbb{U}$:

$$\begin{aligned} \mathbb{U}/IND(P) &= \mathbb{U}/IND(b) \otimes \mathbb{U}/IND(c) \\ &= \{\{0,2,4\},\{1,3,6,7\},\{5\}\} \otimes \{\{2,3,5\},\{1,6,7\},\{0,4\}\} \\ &= \{\{2\},\{0,4\},\{3\},\{1,6,7\},\{5\}\} \end{aligned}$$

Let $X \subseteq \mathbb{U}$. X can be approximated using only the information contained within P by constructing the P-*lower* and P-*upper* approximations of X:

$$\underline{P}X = \{x \mid [x]_P \subseteq X\} \tag{3.4}$$

$$\overline{P}X = \{x \mid [x]_P \cap X \neq \emptyset\} \tag{3.5}$$

Let P and Q be equivalence relations over $\mathbb{U}$, then the positive, negative and boundary regions can be defined as:

$$\begin{array}{rcl} POS_P(Q) & = & \bigcup_{X\in\mathbb{U}/Q}\underline{P}X \\ NEG_P(Q) & = & \mathbb{U} - \bigcup_{X\in\mathbb{U}/Q}\overline{P}X \\ BND_P(Q) & = & \bigcup_{X\in\mathbb{U}/Q}\overline{P}X - \bigcup_{X\in\mathbb{U}/Q}\underline{P}X \end{array}$$

The positive region contains all objects of $\mathbb{U}$ that can be classified to classes of $\mathbb{U}/Q$ using the information in attributes P. The boundary region, $BND_P(Q)$, is the set of objects that can possibly, but not certainly, be classified in this way. The negative region, $NEG_P(Q)$, is the set of objects that cannot be classified to classes of $\mathbb{U}/Q$. For example, let $P = \{b,c\}$ and $Q = \{e\}$, then

$$\begin{array}{rcl} POS_P(Q) & = & \bigcup\{\emptyset,\{2,5\},\{3\}\} = \{2,3,5\} \\ NEG_P(Q) & = & \mathbb{U} - \bigcup\{\{0,4\},\{2,0,4,1,6,7,5\},\{3,1,6,7\}\} = \emptyset \\ BND_P(Q) & = & \bigcup\{\{0,4\},\{2,0,4,1,6,7,5\},\{3,1,6,7\}\} - \{2,3,5\} = \{0,1,4,6,7\} \end{array}$$

This means that objects 2, 3 and 5 can certainly be classified as belonging to a class in attribute e, when considering attributes b and c. The rest of the objects cannot be classified as the information that would make them discernible is absent.

An important issue in data analysis is discovering dependencies between attributes. Intuitively, a set of attributes Q depends totally on a set of attributes P, denoted $P \Rightarrow Q$, if all attribute values from Q are uniquely determined by values of attributes from P. If there exists a functional dependency between values of Q and P, then Q depends totally on P. In rough set theory, dependency is defined in the following way:

For P, $Q \subset \mathbb{A}$, it is said that Q depends on P in a degree k ($0 \leq k \leq 1$), denoted $P \Rightarrow_k Q$, if

$$k = \gamma_P(Q) = \frac{|POS_P(Q)|}{|\mathbb{U}|} \tag{3.6}$$

If $k = 1$, Q depends totally on P, if $0 < k < 1$, Q depends partially (in a degree k) on P, and if $k = 0$ then Q does not depend on P. In the example, the degree of dependency of attribute $\{e\}$ from the attributes $\{b,c\}$ is:

$$\begin{aligned}\gamma_{\{b,c\}}(\{e\}) &= \frac{|POS_{\{b,c\}}(\{e\})|}{|\mathbb{U}|}\\ &= \frac{|\{2,3,5\}|}{|\{0,1,2,3,4,5,6,7\}|} = \frac{3}{8}\end{aligned}$$

By calculating the change in dependency when an attribute is removed from the set of considered conditional attributes, a measure of the significance of the attribute can be obtained. The higher the change in dependency, the more significant the attribute is. If the significance is 0, then the attribute is dispensable. More formally, given P,Q and an attribute $a \in P$,

$$\sigma_P(Q,a) = \gamma_P(Q) - \gamma_{P-\{a\}}(Q) \tag{3.7}$$

For example, if $P = \{a,b,c\}$ and $Q = e$ then

$$\begin{aligned}\gamma_{\{a,b,c\}}(\{e\}) &= |\{2,3,5,6\}|/8 &= 4/8\\ \gamma_{\{a,b\}}(\{e\}) &= |\{2,3,5,6\}|/8 &= 4/8\\ \gamma_{\{b,c\}}(\{e\}) &= |\{2,3,5\}|/8 &= 3/8\\ \gamma_{\{a,c\}}(\{e\}) &= |\{2,3,5,6\}|/8 &= 4/8\end{aligned}$$

And calculating the significance of the three attributes gives:

$$\begin{aligned}\sigma_P(Q,a) &= \gamma_{\{a,b,c\}}(\{e\}) - \gamma_{\{b,c\}}(\{e\}) &= 1/8\\ \sigma_P(Q,b) &= \gamma_{\{a,b,c\}}(\{e\}) - \gamma_{\{a,c\}}(\{e\}) &= 0\\ \sigma_P(Q,c) &= \gamma_{\{a,b,c\}}(\{e\}) - \gamma_{\{a,b\}}(\{e\}) &= 0\end{aligned}$$

From this it follows that attribute a is indispensable, but attributes b and c can be dispensed with when considering the dependency between the decision attribute and the given individual conditional attributes.

3.1.3 Reduction Method

The reduction of attributes is achieved by comparing equivalence relations generated by sets of attributes. Attributes are removed so that the reduced set provides the same predictive capability of the decision feature as the original. A *reduct* is defined as a subset of minimal cardinality R_{min} of the conditional attribute set $\mathbb{C}$ such that $\gamma_R(\mathbb{D}) = \gamma_{\mathbb{C}}(\mathbb{D})$.

$$R = \{X : X \subseteq \mathbb{C}, \gamma_X(\mathbb{D}) = \gamma_{\mathbb{C}}(\mathbb{D})\} \tag{3.8}$$

$$R_{min} = \{X : X \in R, \forall Y \in R, |X| \leq |Y|\} \tag{3.9}$$

The intersection of all the sets in R_{min} is called the *core*, the elements of which are those attributes that cannot be eliminated without introducing more contradictions to the dataset. In RSAR, a subset with minimum cardinality is searched for.

Using the example, the dependencies for all possible subsets of $\mathbb{C}$ can be calculated:

$$\begin{array}{rr}
\gamma_{\{a,b,c,d\}}(\{e\}) = 8/8 & \gamma_{\{b,c\}}(\{e\}) = 3/8 \\
\gamma_{\{a,b,c\}}(\{e\}) = 4/8 & \gamma_{\{b,d\}}(\{e\}) = 8/8 \\
\gamma_{\{a,b,d\}}(\{e\}) = 8/8 & \gamma_{\{c,d\}}(\{e\}) = 8/8 \\
\gamma_{\{a,c,d\}}(\{e\}) = 8/8 & \gamma_{\{a\}}(\{e\}) = 0/8 \\
\gamma_{\{b,c,d\}}(\{e\}) = 8/8 & \gamma_{\{b\}}(\{e\}) = 1/8 \\
\gamma_{\{a,b\}}(\{e\}) = 4/8 & \gamma_{\{c\}}(\{e\}) = 0/8 \\
\gamma_{\{a,c\}}(\{e\}) = 4/8 & \gamma_{\{d\}}(\{e\}) = 2/8 \\
\gamma_{\{a,d\}}(\{e\}) = 3/8 &
\end{array}$$

Note that the given dataset is consistent since $\gamma_{\{a,b,c,d\}}(\{e\}) = 1$. The minimal reduct set for this example is:

$$R_{min} = \{\{b,d\},\{c,d\}\}$$

If $\{b,d\}$ is chosen, then the dataset can be reduced as in table 3.2. Clearly each object can be uniquely classified according to the attribute values remaining.

The problem of finding a reduct of an information system has been the subject of much research [2, 177]. The most basic solution to locating such a subset is to simply generate *all* possible subsets and retrieve those with a maximum rough set dependency degree. Obviously, this is an expensive solution to the problem and is only practical for very simple datasets. Most of the time only one reduct is required as, typically, only one subset of features is used to reduce a dataset, so all the calculations involved in discovering the rest are pointless.

$x \in \mathbb{U}$	b	d	$\Rightarrow$	e
0	0	2		0
1	1	1		2
2	0	1		1
3	1	2		2
4	0	0		1
5	2	1		1
6	1	1		2
7	1	0		1

Table 3.2: Reduced dataset

To improve the performance of the above method, an element of pruning can be introduced. By noting the cardinality of any pre-discovered reducts, the current possible subset can be ignored if it contains more elements. However, a better approach is needed - one that will avoid wasted computational effort.

```
QUICKREDUCT($\mathbb{C}$,$\mathbb{D}$).
$\mathbb{C}$, the set of all conditional features;
$\mathbb{D}$, the set of decision features.

(1) $R \leftarrow \{\}$
(2) do
(3)     $T \leftarrow R$
(4)     $\forall x \in (\mathbb{C} - R)$
(5)         if $\gamma_{R\cup\{x\}}(\mathbb{D}) > \gamma_T(\mathbb{D})$
(6)             $T \leftarrow R \cup \{x\}$
(7)     $R \leftarrow T$
(8) until $\gamma_R(\mathbb{D}) == \gamma_{\mathbb{C}}(\mathbb{D})$
(9) return $R$
```

Figure 3.1: The QUICKREDUCT Algorithm

The QUICKREDUCT algorithm given in figure 3.1 (adapted from [29]), attempts to calculate a reduct without exhaustively generating all possible subsets. It starts off with an empty set and adds in turn, one at a time, those attributes that result in the greatest increase in the rough set dependency metric, until this produces its maximum possible value for the dataset. Other such techniques may be found in [132].

According to the QUICKREDUCT algorithm, the dependency of each attribute is calculated, and the best candidate chosen. In figure 3.2, this stage is illustrated using the example dataset. As attribute d generates the highest dependency degree, then that attribute is chosen and the sets $\{a,d\}$, $\{b,d\}$ and $\{c,d\}$ are evaluated. This process continues until the dependency of the reduct equals the consistency of the dataset (1 if the dataset is consistent). The generated reduct shows the way of reducing the dimensionality of the original dataset by eliminating those conditional attributes that do not appear in the set.

Determining the consistency of the entire dataset is reasonable for most datasets. However, it may be infeasible for very large data, so alternative stopping criteria may have to be used. One such criterion could be to terminate the search when there is no further increase in the dependency measure. This will produce exactly the same path to a reduct due to the monotonicity of the measure [29], without the computational overhead of calculating the dataset consistency.

Other developments include REVERSEREDUCT where the strategy is backward elimination of attributes as opposed to the current forward selection process. Initially, all attributes appear in the reduct candidate; the least informative ones are incrementally removed until no further attribute can be eliminated without introducing inconsistencies. This is not often used for large datasets, as the algorithm must evaluate large feature subsets (starting with the set containing *all* features) which is too costly, although the computational complexity is, in theory, the same as that of forward-looking QUICKREDUCT. As both forward and backward methods perform well, it is thought that a combination of these within one algorithm would be effective. For instance, search could continue in a forward direction initially, then resort to backward steps intermittently to remove less important features before continuing onwards.

This, however, is not guaranteed to find a *minimal* subset as has been shown in [30]. Using the dependency function to discriminate between candidates may lead the search down a non-minimal path. It is impossible to predict which combinations of attributes will lead to an optimal reduct based on changes in dependency with the addition or deletion of single attributes. It does result in a close-to-minimal subset, though, which

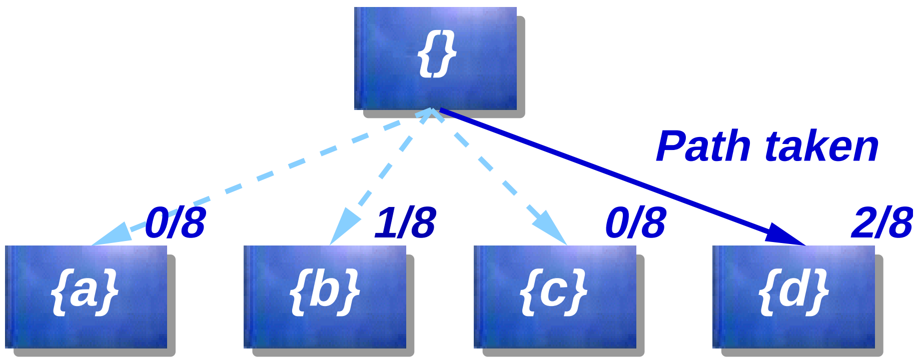


Figure 3.2: Branches of the search space

is still useful in greatly reducing dataset dimensionality.

In [30], a potential solution to this problem has been proposed whereby the QUICKREDUCT algorithm is altered, making it into an *n*-lookahead approach. However, even this cannot guarantee a reduct unless *n* is equal to the original number of attributes, but this reverts back to generate-and-test. It still suffers from the same problem as the original QUICKREDUCT, i.e. it is impossible to tell at any stage whether the current path will be the shortest to a reduct.

It is interesting to note that the rough set degree of dependency measure is very similar to the consistency criterion used by the FOCUS algorithm and others [1, 148]. In FOCUS, a breadth-first search is employed such that any subset is rejected if this produces at least one inconsistency. If this is converted into a guided search using the consistency measure as a heuristic, it should behave exactly as QUICKREDUCT. Consistency is defined as the number of discernible objects out of the entire object set - exactly that of the dependency measure.

3.2 Discernibility Matrix Approach

Many applications of rough sets to feature selection make use of discernibility matrices for finding reducts. A discernibility matrix [86, 164] of a decision table $D = (\mathbb{U}, \mathbb{C} \cup \mathbb{D})$ is a symmetric $|\mathbb{U}| \times |\mathbb{U}|$ matrix with entries defined:

$$d_{ij} = \{a \in \mathbb{C} | a(x_i) \neq a(x_j)\} \;\; i,j = 1,...,|\mathbb{U}| \tag{3.10}$$

Each d_{ij} contains those attributes that differ between objects i and j. For finding reducts, the decision-relative discernibility matrix is of more interest. This only considers those object discernibilities that occur when the corresponding decision attributes differ. Returning to the example dataset, the decision-relative discernibility matrix found in table 3.3 is produced. For example, it can be seen from the table that objects 0 and 1 differ in each attribute. Although some attributes in objects 1 and 3 differ, their corresponding decisions are the same so no entry appears in the decision-relative matrix. Grouping all entries containing single attributes forms the core of the dataset (those attributes appearing in *every* reduct). Here, the core of the dataset is $\{d\}$.

$x \in \mathbb{U}$	0	1	2	3	4	5	6	7
0								
1	a,b,c,d							
2	a,c,d	a,b,c						
3	b,c		a,b,d					
4	d	a,b,c,d		b,c,d				
5	a,b,c,d	a,b,c		a,b,d				
6	a,b,c,d		b,c		a,b,c,d	b,c		
7	a,b,c,d	d		a,c,d			a,d	

Table 3.3: The decision-relative discernibility matrix

From this, the discernibility function can be defined. This is a concise notation of how each object within the dataset may be distinguished from the others. A discernibility function f_D is a boolean function of m boolean variables $a_1^*,...,a_m^*$ (corresponding to the attributes $a_1,...,a_m$) defined as below:

$$f_D(a_1^*,...,a_m^*) = \wedge\{\vee c_{ij}^* | 1 \leq j \leq i \leq |\mathbb{U}|, c_{ij} \neq \emptyset\} \tag{3.11}$$

where $c_{ij}^* = \{a^* | a \in c_{ij}\}$. By finding the set of all prime implicants of the discernibility function, all the minimal reducts of a system may be determined. From table 3.3, the decision-relative discernibility function is (with duplicates removed):

$$\begin{aligned} f_D(a,b,c,d) = \quad & \{a \vee b \vee c \vee d\} \wedge \{a \vee c \vee d\} \wedge \{b \vee c\} \\ & \wedge \{d\} \wedge \{a \vee b \vee c\} \wedge \{a \vee b \vee d\} \\ & \wedge \{b \vee c \vee d\} \wedge \{a \vee d\} \end{aligned}$$

Further simplification can be performed by removing those sets (clauses) that are supersets of others:

$$f_D(a,b,c,d) = \quad \{b \vee c\} \wedge \{d\}$$

The reducts of the dataset may be obtained by converting the above expression from conjunctive normal form to disjunctive normal form (without negations). Hence, the minimal reducts are $\{b,d\}$ and $\{c,d\}$. Although this is guaranteed to discover all minimal subsets, it is a costly operation rendering the method impractical for even medium-sized datasets.

For most applications, a single minimal subset is required for data reduction. This has led to approaches that consider finding individual shortest prime implicants from the discernibility function. A common method is to incrementally add those attributes that occur with the most frequency in the function, removing any clauses containing the attributes, until all clauses are eliminated [120, 185]. However, even this does not ensure that a minimal subset is found - the search can proceed down non-minimal paths.

3.3 Reduction with Variable Precision Rough Sets

Variable precision rough sets (VPRS) [204] extends rough set theory by the relaxation of the subset operator. It was proposed to analyse and identify data patterns which represent statistical trends rather than functional. The main idea of VPRS is to allow objects to be classified with an error smaller than a certain predefined level. This introduced threshold relaxes the rough set notion of requiring no information outside the dataset itself. Let $X, Y \subseteq \mathbb{U}$, the relative classification error is defined by:

$$c(X,Y) = 1 - \frac{|X \cap Y|}{|X|}$$

Observe that $c(X,Y) = 0$ if and only if $X \subseteq Y$. A degree of inclusion can be achieved by allowing a certain level of error, β, in classification:

$$X \subseteq_{\beta} Y \text{ iff } c(X,Y) \leq \beta, \quad 0 \leq \beta < 0.5$$

Using $\subseteq_{\beta}$ instead of $\subseteq$, the β-upper and β-lower approximations of a set X can be defined as:

$$\underline{R}_{\beta}X = \bigcup\{[x]_R \in \mathbb{U}/R \mid [x]_R \subseteq_{\beta} X\}$$
$$\overline{R}_{\beta}X = \bigcup\{[x]_R \in \mathbb{U}/R \mid c([x]_R, X) < 1 - \beta\}$$

Note that $\underline{R}_{\beta}X = \underline{R}X$ for $\beta = 0$. The positive, negative and boundary regions in the original rough set theory can now be extended to:

$$POS_{R,\beta}(X) = \underline{R}_{\beta}X \tag{3.12}$$

$$NEG_{R,\beta}(X) = \mathbb{U} - \overline{R}_{\beta}X \tag{3.13}$$

$$BND_{R,\beta}(X) = \overline{R}_{\beta}X - \underline{R}_{\beta}X \tag{3.14}$$

Returning to the example dataset in table 3.1, equation 3.12 can be used to calculate the β-positive region for $R = \{b,c\}$, $X = \{e\}$ and $\beta = 0.4$. Setting β to this value means that a set is considered to be a subset of another if at least 60% of its elements exist in the other. The partitions of the universe of objects for R and X are:

$\mathbb{U}/R = \{\{2\},\{0,4\},\{3\},\{1,6,7\},\{5\}\}$
$\mathbb{U}/X = \{\{0\},\{1,3,6\},\{2,4,5,7\}\}$

For each set $A \in \mathbb{U}/R$ and $B \in \mathbb{U}/X$, the value of $c(A,B)$ must be less than β if the equivalence class A is to be included in the β-positive region. Considering $A = \{2\}$

gives

$$c(\{2\},\{0\}) = 1 > \beta$$
$$c(\{2\},\{1,3,6\}) = 1 > \beta$$
$$c(\{2\},\{2,4,5,7\}) = 0 < \beta$$

So object 2 is added to the β-positive region as it is a β-subset of $\{2,4,5,7\}$ (and is in fact a traditional subset of the equivalence class). Taking $A = \{1,6,7\}$, a more interesting case is encountered:

$$c(\{1,6,7\},\{0\}) = 1 > \beta$$
$$c(\{1,6,7\},\{1,3,6\}) = 0.3333 < \beta$$
$$c(\{1,6,7\},\{2,4,5,7\}) = 0.6667 > \beta$$

Here the objects 1, 6 and 7 are included in the β-positive region as the set $\{1,6,7\}$ is a β-subset of $\{1,3,6\}$. Calculating the subsets in this way leads to the following β-positive region:

$POS_{R,\beta}(X) = \{1,2,3,5,6,7\}$

Compare this with the positive region generated previously: $\{2,3,5\}$. Objects 1, 6 and 7 are now included due to the relaxation of the subset operator. Consider a decision table $A = (\mathbb{U}, \mathbb{C} \cup \mathbb{D})$, where $\mathbb{C}$ is the set of conditional attributes and $\mathbb{D}$ the set of decision attributes. The β-positive region of an equivalence relation Q on $\mathbb{U}$ may be determined by

$$POS_{R,\beta}(Q) = \bigcup_{X \in \mathbb{U}/Q} \underline{R}_{\beta} X$$

where R is also an equivalence relation on $\mathbb{U}$. This can then be used to calculate dependencies and thus determine β-reducts. The dependency function becomes:

$$\gamma_{R,\beta}(Q) = \frac{|POS_{R,\beta}(Q)|}{|\mathbb{U}|}$$

It can be seen that the QUICKREDUCT algorithm outlined previously can be adapted to incorporate the reduction method built upon the VPRS theory. By supplying a suitable β value to the algorithm, the β-lower approximation, β-positive region, and β-dependency can replace the traditional calculations. This will result in a more approximate final reduct, which may be a better generalization when encountering unseen data. Additionally, setting β to 0 forces such a method to behave exactly like RSAR.

Extended classification of reducts in the VPRS approach may be found in [14, 15, 91]. As yet, there have been no comparative experimental studies between rough set methods and the VPRS method. However, the variable precision approach requires the additional parameter β which has to be specified from the start. By repeated experimentation, this parameter can be suitably approximated. However, problems arise when searching for true reducts as VPRS incorporates an element of inaccuracy in determining the number of classifiable objects.

3.4 Dynamic Reducts

Reducts generated from an information system are sensitive to changes in the system. This can be seen by removing a randomly chosen set of objects from the original object set. Those reducts frequently occurring in random subtables can be considered to be stable; it is these reducts that are encompassed by *dynamic reducts* [10]. Let $A = (\mathbb{U}, \mathbb{C} \cup d)$ be a decision table, then any system $B = (\mathbb{U}', \mathbb{C} \cup d)$ ($\mathbb{U}' \subseteq \mathbb{U}$) is called a subtable of A. If F is a family of subtables of A, then

$$DR(A, F) = Red(A, d) \cap \{\bigcap_{B \in F} Red(B, d)\}$$

defines the set of F-dynamic reducts of A. From this definition, it follows that a relative reduct of A is dynamic if it is also a reduct of all subtables in F. In most cases this is too restrictive, so a more general notion of dynamic reducts is required.

By introducing a threshold, $0 \leq \varepsilon \leq 1$, the concept of (F, ε)-dynamic reducts can here be defined:

$$DR_{\varepsilon}(A, F) = \{C \in Red(A, d) : s_F(C) \geq \varepsilon\}$$

where

$$s_F(C) = \frac{|\{B \in F : C \in Red(B,d)\}|}{|F|}$$

is the F-stability coefficient of C. This lessens the previous restriction that a dynamic reduct must appear in *every* generated subtable. Now, a reduct is considered to be dynamic if it appears in a certain proportion of subtables, determined by the value ε. For example, by setting ε to 0.5 a reduct is considered to be dynamic if it appears in at least half of the subtables. Note that if $F = \{A\}$ then $DR(A,F) = Red(A,d)$. Dynamic reducts may then be calculated according to the algorithm given in figure 3.3. Firstly, all reducts are calculated for the given information system, A. Then, the new subsystems A_j are generated by randomly deleting one or more rows from A. All reducts are found for each subsystem, and the dynamic reducts are computed using $s_F(C,R)$ which denotes the significance factor of reduct C within all reducts found, R.

DynamicRed(A,ε,its).
A, the original decision table;
ε, the dynamic reduct threshold;
its, the number of iterations.

```
(1) $R \leftarrow \{\}$
(2) $A \leftarrow$ calculateAllReducts($A$)
(3) for $j$=1...$its$
(4)     $A_j \leftarrow$ deleteRandomRows($A$)
(5)     $R \leftarrow R \cup$ calculateAllReducts($A_j$)
(6) $\forall C \in A$
(7)     if $s_F(C,R) \geq \varepsilon$
(8)         output $C$
```

Figure 3.3: Dynamic Reduct algorithm

Returning to the example decision table (call this A), the first step is to calculate all its reducts. This produces the set of all reducts $A = \{\{b,d\}, \{c,d\}, \{a,b,d\}, \{a,c,d\}, \{b,c,d\}\}$. The reduct $\{a,b,c,d\}$ is not included as this will always be a reduct of any generated subtable (it is the full set of conditional attributes). The next step randomly deletes a number of rows from the original table A. From this, all reducts are again calculated. For one subtable this might be $R = \{\{b,d\},\{b,c,d\},\{a,b,d\}\}$. In this case, the subset $\{c,d\}$ is not a reduct (though it was for the original dataset). If the number of iterations is set to just one, and if ε is set to a value less than 0.5 (implying that a reduct should appear in half of the total number of discovered reducts), then the reduct $\{c,d\}$ is deemed not to be a dynamic reduct.

Intuitively, this is based on the hope that by finding stable reducts they will be more representative of the real world, i.e. it is more likely that they will be reducts for unseen data. A comparison of dynamic and non-dynamic approaches can be found in [11], where various methods were tested on extracting laws from decision tables. In the experiments, the dynamic method and the conventional RS method both performed well. In fact, it appears that the RS method has on average a lower error rate of classification than the dynamic RS method.

A disadvantage of this dynamic approach is that several subjective choices have to be made before the dynamic reducts can be found (for instance the choice of the value of ε); these values are not contained in the data. Also, the huge complexity of finding all reducts within subtables forces the use of heuristic techniques such as genetic algorithms to perform the search. For large datasets, this step may well be too costly.

3.5 Alternative Approaches

Other approaches to generating reducts from information systems have been developed and can be found in [19, 169, 187]. Among the first rough set-based approaches is the PRESET algorithm [115] which is another feature selector that uses rough set theory to rank heuristically the features, assuming a noise free binary domain. Since PRESET does not try to explore all combinations of the features, it is certain that it will fail on problems whose attributes are highly correlated. There have also been investigations into the use of different reduct quality measures (see [132] for details).

In [203], a heuristic filter-based approach is presented based on rough set theory. The algorithm proposed, as reformalised in figure 3.4, begins with the core of the dataset (those attributes that cannot be removed without introducing inconsistencies) and incrementally adds attributes based on a heuristic measure. Additionally, a threshold value is required as a stopping criterion to determine when a reduct candidate is “near enough" to being a reduct. On each iteration, those objects that are consistent with the current reduct candidate are removed (an optimization that can be used with RSAR). As the process starts with the core of the dataset, this has to be calculated beforehand. Using the discernibility matrix for this purpose can be quite impractical for datasets of large dimensionality. However, there are other methods that can calculate the core in an efficient manner [125]. For example, this can be done by calculating the degree of dependency of the full feature set and the corresponding dependencies of the feature set minus each attribute. Those features that result in a dependency decrease are core attributes. There are also alternative methods available that allow the calculation of necessary information about the discernibility matrix without the need to perform operations directly on it [119].

select($\mathbb{C}$,$\mathbb{D}$,O,ε).
$\mathbb{C}$, the set of all conditional features;
$\mathbb{D}$, the set of decision features;
O, the set of objects (instances);
ε, reduct threshold.

```
(1)  $R \leftarrow$ calculateCore()
(2)  **while** $(\gamma_R(\mathbb{D}) < \varepsilon)$
(3)      $O \leftarrow O - POS_R(\mathbb{D})$ //optimization
(4)      $\forall a \in \mathbb{C} - R$
(5)          $v_a = |POS_{R\cup\{a\}}(D)|$
(6)          $m_a = |$largestEquivClass$(POS_{R\cup\{a\}(D)})|$
(7)      Choose $a$ with largest $v_a * m_a$
(8)      $R \leftarrow R \cup \{a\}$
(9)  **return** $R$
```

Figure 3.4: Heuristic filter-based algorithm

Also worth mentioning are the approaches reported in [19, 187] which use genetic algorithms to discover optimal or close-to-optimal reducts. Reduct candidates are encoded as bit strings, with the value in position i set if the ith attribute is present. The fitness function depends on two parameters. The first is the number of bits set. The function penalises those strings which have larger numbers of bits set, driving the process to find smaller reducts. The second is the number of classifiable objects given this candidate. The reduct should discern between as many objects as possible (ideally all of them).

Although this approach is not guaranteed to find minimal subsets, it may find many subsets for any given dataset. It is also useful for situations where new objects are added to or old objects are removed from a dataset - the reducts generated previously can be used as the initial population for the new reduct-determining process. The main drawback is the time taken to compute each bit string's fitness, which is $O(a * o^2)$, where a is the number of attributes and o the number of objects in the dataset. The extent to which this hampers performance depends mainly on the population size.

3.6 Comparison of Crisp Approaches

In order to evaluate several of the mainstream approaches to rough set-based feature selection, an investigation into how these methods perform in terms of resulting subset optimality has been carried out here. Several real and artificial datasets are used for this purpose. In particular, it is interesting to compare those methods that employ an incremental-based search strategy with those that adopt a more complex stochastic/probabilistic mechanism.

3.6.1 Dependency Degree-based Approaches

Five techniques for finding crisp rough set reducts are tested here on 13 datasets. These techniques are: RSAR (using QUICKREDUCT), EBR (using the same search mechanism as QUICKREDUCT), GenRSAR (genetic algorithm-based), AntRSAR (ant-based) and SimRSAR (simulated annealing-based).

3.6.1.1 Experimental Setup

Before the experiments are described, a few points must be made about the later three approaches, GenRSAR, AntRSAR and SimRSAR.

GenRSAR employs a genetic search strategy in order to determine rough set reducts. The initial population consists of 100 randomly generated feature subsets, the probabilities of mutation and crossover are set to 0.4 and 0.6 respectively, and the number of generations is set to 100. The fitness function considers both the size of subset and its evaluated suitability, and is defined as follows:

$$fitness(R) = \gamma_R(\mathbb{D}) * \frac{|\mathbb{C}| - |R|}{|\mathbb{C}|} \tag{3.15}$$

AntRSAR follows the mechanism described in [76] and section 5.2. This demonstrates that the general ant-based FS framework presented in this thesis can be applied to crisp rough set-based selection. Here, the precomputed heuristic desirability of edge traversal is the entropy measure, with the subset evaluation performed using the rough set dependency heuristic (to guarantee that true rough set reducts are found). The number of ants used is set to the number of features, with each ant starting on a different feature. Ants construct possible solutions until they reach a rough set reduct. To avoid fruitless searches, the size of the current best reduct is used to reject those subsets whose cardinality exceed this value. Pheromone levels are set at 0.5 with a small random variation added. Levels are increased by only those ants who have found true reducts. The global search is terminated after 250 iterations, α is set to 1 and β is set to 0.1.

SimRSAR employs a simulated annealing-based feature selection mechanism [76]. The states are feature subsets, with random state mutations set to changing three features (either adding or removing them). The cost function attempts to maximize the rough set dependency (γ) whilst minimizing the subset cardinality. For these experiments, the cost of subset R is defined as:

$$cost(R) = \left[\frac{\gamma_{\mathbb{C}}(\mathbb{D}) - \gamma_R(\mathbb{D})}{\gamma_{\mathbb{C}}(\mathbb{D})}\right]^a + \left[\frac{|R|}{|\mathbb{C}|}\right]^b \tag{3.16}$$

where a and b are defined in order to weight the contributions of dependency and subset size to the overall cost measure. In the experiments here, $a = 1$ and $b = 3$. The initial temperature of the system is estimated as $2 * |\mathbb{C}|$ and the cooling schedule is $T(t+1) = 0.93 * T(t)$.

The experiments were carried out on 3 datasets from [137], namely *m-of-n*, *exactly* and *exactly2*. The remaining datasets are from the machine learning repository

[20]. Those datasets containing real-valued attributes have been discretized to allow all methods to be compared fairly.

3.6.1.2 Experimental Results

Table 3.4 presents the results of the five methods on the 13 datasets. It shows the size of reduct found for each method, as well as the size of the optimal (minimal) reduct. RSAR and EBR produced the same subset every time, unlike AntRSAR and SimRSAR that often found different subsets and sometimes different subset cardinalities. On the whole, it appears to be the case that AntRSAR and SimRSAR outperform the other three methods. This is at the expense of the time taken to discover these reducts as can be seen in Fig. 3.5 (results for RSAR and EBR do not appear as they are consistently faster than the other methods). In all experiments the rough ordering of techniques with respect to time is: RSAR $<$ EBR $\leq$ SimRSAR $\leq$ AntRSAR $\leq$ GenRSAR. AntRSAR and SimRSAR perform similarly throughout - for some datasets, AntRSAR is better (e.g. Vote) and for others SimRSAR is best (e.g. LED). The performance of these two methods may well be improved by fine-tuning the parameters to each individual dataset.

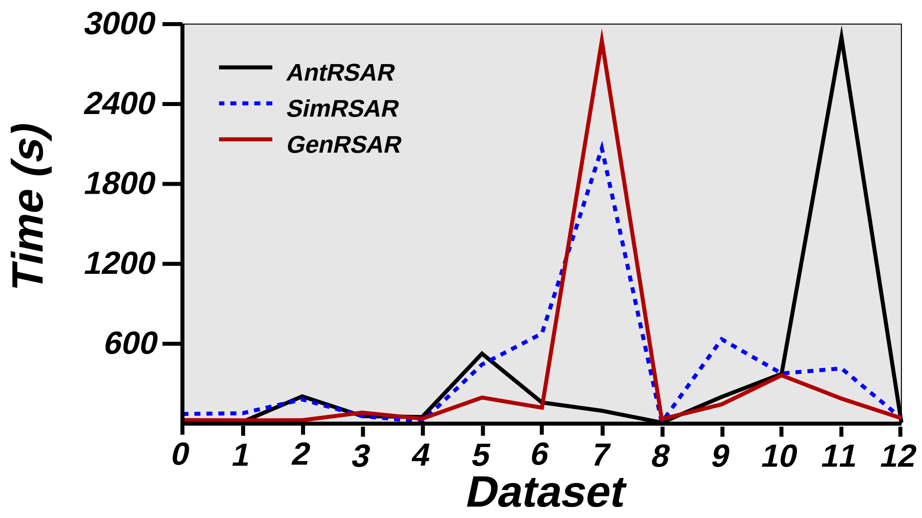


Figure 3.5: Average runtimes for AntRSAR, SimRSAR and GenRSAR

From these results it can be seen that even for small and medium-sized datasets,

Index	Dataset	Features	Optimal	RSAR	EBR	AntRSAR	SimRSAR	GenRSAR
0	M-of-N	13	6	8	6	6	6	6(6) 7(12)
1	Exactly	13	6	9	8	6	6	6(10) 7(10)
2	Exactly2	13	10	13	11	10	10	10(9) 11(11)
3	Heart	13	6	7	7	6(18) 7(2)	6(29) 7(1)	6(18) 7(2)
4	Vote	16	8	9	9	8	8(15) 9(15)	8(2) 9(18)
5	Credit	20	8	9	10	8(12) 9(4) 10(4)	8(18) 9(1) 11(1)	10(6) 11(14)
6	Mushroom	22	4	5	4	4	4	5(1) 6(5) 7(14)
7	LED	24	5	12	5	5(12) 6(4) 7(3)	5	6(1) 7(3) 8(16)
8	Letters	25	8	9	9	8	8	8(8) 9(12)
9	Derm	34	6	7	6	6(17) 7(3)	6(12) 7(8)	10(6) 11(14)
10	Derm2	34	8	10	10	8(3) 9(17)	8(3) 9(7)	10(2) 11(8)
11	WQ	38	12	14	14	12(2) 13(7) 14(11)	13(16) 14(4)	16
12	Lung	56	4	4	4	4	4(7) 5(12) 6(1)	6(8) 7(12)

Table 3.4: Subset sizes found for five techniques

incremental hill-climbing techniques often fail to find minimal subsets. For example, RSAR is misled early in the search for the LED dataset, resulting in it choosing 7 extraneous features. Although this fault is due to the non-optimality of the guiding heuristic, a perfect heuristic does not exist rendering these approaches unsuited to problems where a minimal subset is essential. However, for most real world applications, the extent of reduction achieved via such methods is acceptable. For systems where the minimal subset is required (perhaps due to the cost of feature measurement), stochastic feature selection should be used.

3.6.2 Discernibility Matrix-based Approaches

Three techniques that use the discernibility matrix to locate reducts are evaluated here on the same datasets used previously. HC is a simple hill climber that selects the next attribute based on its frequency in the clauses appearing in the discernibility matrix, following a similar strategy to that of the reduction method based on Johnson's algorithm given in the Rough Set Exploration System (RSES) [140]. NS follows a similar strategy to HC, but also uses information about the size of the clauses in the guiding heuristic.

Clause-based Search (CS), introduced here, performs search in a breadth-first manner. The process starts with an empty list, *Subsets*, which keeps a record of all current feature subsets. Clauses from the discernibility matrix are considered one at a time in order of their size, with those of the smallest cardinality chosen first. When a clause is selected, the features appearing within the clause are added to every set in *Subsets*. For example, if $Subsets$ contains $\{a,b\}$ and $\{c,d\}$, and the next considered clause is $\{d \vee e\}$ then each appearing attribute is added. The $Subsets$ list will now contain $\{a,b,d\}$, $\{a,b,e\}$, $\{c,d\}$ and $\{c,d,e\}$. This guarantees that each set in $Subsets$ satisfies all the clauses that have been encountered so far. If one of these subsets satisfies all clauses the algorithm terminates as a reduct has been found. If not, then the process continues by selecting the next clause and adding these features. This process will result in a minimal subset, but has an exponential time and space complexity.

The results of the application of these three methods to the 13 datasets can be found in Table 3.5. HC and NS perform similarly throughout, differing only in their results

Dataset	Features	HC	NS	CS
M-of-N	13	6	6	6
Exactly	13	6	6	6
Exactly2	13	10	10	10
Heart	13	6	6	6
Vote	16	8	8	8
Credit	20	10	10	8
Mushroom	22	4	4	4
LED	24	5	5	5
Letters	25	9	10	8
Derm	34	6	6	6
Derm2	34	9	9	8
WQ	38	14	13	12
Lung	56	4	4	4

Table 3.5: Subset sizes found for discernibility matrix-based techniques

for the Letters and WQ datasets. CS will always find the smallest valid feature subset, though is too costly to apply to larger datasets in its present form. On the whole, all three methods perform as well as or better than the dependency-based methods. However, HC, NS and CS all require the calculation of the discernibility matrix beforehand.

3.7 Summary

Feature selection seeks to reduce data while retaining semantics by selecting attributes as opposed to transforming them. This aspect is particularly useful when feature selection precedes other processes that require the original feature meanings to be intact, for example rule induction where rules may need to be human-comprehensible. This chapter focussed on some of the recent developments in rough set theory for the purpose of feature selection.

Several approaches to discovering rough set reducts were experimentally evaluated and compared. The results highlighted the shortcomings of conventional hill-climbing approaches to feature selection. These techniques often fail to find minimal data reductions. Some guiding heuristics are better than others for this, but as no perfect heuristic exists there can be no guarantee of optimality. From the experimentation, it appears that the entropy-based measure is a more useful hill-climbing heuristic than the rough set-based one. However, the entropy measure is a more costly operation than that of dependency evaluation which may be an important factor when processing large datasets. Due to the failure of hill-climbing methods and the fact that exhaustive searches are not feasible for even medium-sized datasets, stochastic approaches provide a promising feature selection mechanism.

Ultimately, conventional rough set methods are all unable to deal with real-valued attributes effectively. This prompted research into the use of fuzzy-rough sets for feature selection.

Chapter 4

Fuzzy-Rough Feature Selection

The RSAR process described previously can only operate effectively with datasets containing discrete values. Additionally, there is no way of handling noisy data. As most datasets contain real-valued features, it is necessary to perform a discretization step beforehand. This is typically implemented by standard fuzzification techniques [157], enabling linguistic labels to be associated with attribute values. It also aids uncertainty modelling by allowing the possibility of the membership of a value to more than one fuzzy label. However, membership degrees of feature values to fuzzy sets are not exploited in the process of dimensionality reduction. By using *fuzzy-rough* sets [43, 123], it is possible to use this information to better guide feature selection.

4.1 Fuzzy Equivalence Classes

In the same way that crisp equivalence classes are central to rough sets, *fuzzy* equivalence classes are central to the fuzzy-rough set approach [43, 179, 196]. An introduction to fuzzy set theory can be found in appendix A. For typical RSAR applications, this means that the decision values and the conditional values may all be fuzzy. The concept of crisp equivalence classes can be extended by the inclusion of a fuzzy similarity relation S on the universe, which determines the extent to which two elements are similar in S. For example, if $\mu_S(x,y) = 0.9$, then objects x and y are considered to be quite similar. The usual properties of reflexivity ($\mu_S(x,x) = 1$), symmetry ($\mu_S(x,y)$

= $\mu_S(y,x)$) and transitivity ($\mu_S(x,z) \geq \mu_S(x,y) \wedge \mu_S(y,z)$) hold.

Using the fuzzy similarity relation, the fuzzy equivalence class $[x]_S$ for objects close to x can be defined:

$$\mu_{[x]_S}(y) = \mu_S(x,y) \tag{4.1}$$

The following axioms should hold for a fuzzy equivalence class F [63]:

- $\exists x$, $\mu_F(x)$ = 1 (μ_F is normalised)
- $\mu_F(x) \wedge \mu_S(x,y) \leq \mu_F(y)$
- $\mu_F(x) \wedge \mu_F(y) \leq \mu_S(x,y)$

The first axiom corresponds to the requirement that an equivalence class is non-empty. The second axiom states that elements in y's neighbourhood are in the equivalence class of y. The final axiom states that any two elements in F are related via S. Obviously, this definition degenerates to the normal definition of equivalence classes when S is non-fuzzy.

The family of normal fuzzy sets produced by a fuzzy partitioning of the universe of discourse can play the role of fuzzy equivalence classes [43]. Consider the crisp partitioning of a universe of discourse, $\mathbb{U}$, by the attributes in Q: $\mathbb{U}/Q$ = $\{\{1,3,6\},\{2,4,5\}\}$. This contains two equivalence classes ($\{1,3,6\}$ and $\{2,4,5\}$) that can be thought of as degenerated fuzzy sets, with those elements belonging to the class possessing a membership of one, zero otherwise. For the first class, for instance, the objects 2, 4 and 5 have a membership of zero. Extending this to the case of fuzzy equivalence classes is straightforward: objects can be allowed to assume membership values, with respect to any given class, in the interval [0,1]. $\mathbb{U}/Q$ is not restricted to crisp partitions only; fuzzy partitions are equally acceptable [109].

4.1.1 Fuzzy-Rough Sets

From the literature, the fuzzy P-lower and P-upper approximations are defined as [43]:

$$\mu_{\underline{P}X}(F_i) = inf_x max\{1 - \mu_{F_i}(x), \mu_X(x)\} \quad \forall i \tag{4.2}$$

$$\mu_{\overline{P}X}(F_i) = sup_x min\{\mu_{F_i}(x), \mu_X(x)\} \quad \forall i \tag{4.3}$$

where F_i denotes a fuzzy equivalence class belonging to $\mathbb{U}/P$. Note that although the universe of discourse in feature selection is finite, this is not the case in general, hence the use of sup and inf. These definitions diverge a little from the crisp upper and lower approximations, as the memberships of individual objects to the approximations are not explicitly available. As a result of this, the fuzzy lower and upper approximations are herein redefined as:

$$\mu_{\underline{P}X}(x) = \sup_{F \in \mathbb{U}/P} min(\mu_F(x), \inf_{y \in \mathbb{U}} max\{1 - \mu_F(y), \mu_X(y)\}) \tag{4.4}$$

$$\mu_{\overline{P}X}(x) = \sup_{F \in \mathbb{U}/P} min(\mu_F(x), \sup_{y \in \mathbb{U}} min\{\mu_F(y), \mu_X(y)\}) \tag{4.5}$$

In implementation, not all $y \in \mathbb{U}$ need to be considered - only those where $\mu_F(y)$ is non-zero, i.e. where object y is a fuzzy member of (fuzzy) equivalence class F. The tuple $< \underline{P}X, \overline{P}X >$ is called a *fuzzy-rough set*. For this particular feature selection method, the upper approximation is not used, though this may be useful for other methods.

It can be seen that these definitions degenerate to traditional rough sets when all equivalence classes are crisp. It is useful to think of the crisp lower approximation as characterized by the following membership function:

$$\mu_{\underline{P}X}(x) = \begin{cases} 1, & x \in F, \; F \subseteq X \\ 0, & otherwise \end{cases} \tag{4.6}$$

This states that an object x belongs to the P-lower approximation of X if it belongs to an equivalence class that is a subset of X. Obviously, the behaviour of the fuzzy lower approximation must be exactly that of the crisp definition for crisp situations. This is indeed the case as the fuzzy lower approximation may be rewritten as

$$\mu_{\underline{P}X}(x) = \sup_{F \in \mathbb{U}/P} min(\mu_F(x), \inf_{y \in \mathbb{U}} \{\mu_F(y) \rightarrow \mu_X(y)\}) \tag{4.7}$$

where "$\rightarrow$" stands for fuzzy implication (using the conventional min-max interpretation). In the crisp case, $\mu_F(x)$ and $\mu_X(x)$ will take values from $\{0, 1\}$. Hence, it is clear that the only time $\mu_{\underline{P}X}(x)$ will be zero is when at least one object in its equivalence class F fully belongs to F but not to X. This is exactly the same as the definition for the crisp lower approximation. Similarly, the definition for the P-upper approximation can be established.

4.1.2 Rough-Fuzzy Sets

Also defined in the literature are rough-fuzzy sets [174], which can be seen to be a particular case of fuzzy-rough sets. A rough-fuzzy set is a generalisation of a rough set, derived from the approximation of a fuzzy set in a crisp approximation space. This corresponds to the case where only the decision attribute values are fuzzy; the conditional values are crisp. The lower and upper approximations incorporate the extent to which objects belong to these sets, and are defined as:

$$\mu_{\underline{R}X}([x]_R) = inf\{\mu_X(x)|x \in [x]_R\} \tag{4.8}$$

$$\mu_{\overline{R}X}([x]_R) = sup\{\mu_X(x)|x \in [x]_R\} \tag{4.9}$$

where $\mu_X(x)$ is the degree to which x belongs to fuzzy equivalence class X, and each $[x]_R$ is crisp. The tuple $< \underline{R}X, \overline{R}X >$ is called a rough-fuzzy set. It can be seen that in the crisp case (where $\mu_X(x)$ is 1 or 0), the above definitions become identical to that of the traditional lower and upper approximations.

Rough-fuzzy sets can be generalised to fuzzy-rough sets [43], where *all* equivalence classes may be fuzzy. When applied to dataset analysis, this means that both the decision values and the conditional values may be fuzzy or crisp.

4.1.3 Fuzzy-Rough Hybrids

In addition to the fuzzy-rough definitions given previously, other generalizations are possible [129]. In [12], the concepts of information theoretic measures are related to rough sets, comparing these to established rough set models of uncertainty. This work has been applied to the rough and fuzzy-rough relational database models, where an alternative definition of fuzzy-rough sets which originates from the rough membership function is chosen [125].

Rough sets may be expressed by a fuzzy membership function to represent the negative, boundary and positive regions [188]. All objects in the positive region have a membership of one and those belonging to the boundary region have a membership of 0.5. Those that are contained in the negative region (and therefore do not belong to the rough set) have zero membership. In so doing, a rough set can be expressed as a fuzzy set, with suitable modifications to the rough union and intersection operators.

The reason for integrating fuzziness into rough sets is to quantify the levels of roughness in the boundary region by using fuzzy membership values. It is necessary, therefore, to allow elements in the boundary region to have membership values in the range of 0 to 1, not just the value 0.5. Hence, a fuzzy rough set Y is defined (by this approach) as a membership function $\mu_Y(x)$ that associates a grade of membership from the interval [0,1] with every element of $\mathbb{U}$. For a rough set X and a crisp equivalence relation R:

$$\begin{aligned}
&\mu_Y(\underline{R}X) = 1, \\
&\mu_Y(\mathbb{U} - \overline{R}X) = 0, \\
&0 < \mu_Y(\overline{R}X - \underline{R}X) < 1
\end{aligned}$$

However, this is not a true hybridization of the two approaches, it merely assigns a degree of membership to the elements depending on the crisp positive, boundary or negative region they belong to. Fuzzy equivalence classes are not used and so this does not offer a particularly useful approach for fuzzy-rough attribute reduction.

Another approach that blurs the distinction between rough and fuzzy sets has been proposed in [129]. The research was fuelled by the concern that a purely numeric fuzzy set representation may be too precise; a concept is described exactly once its membership function has been defined. This seems as though excessive precision is required in order to describe imprecise concepts.

The solution proposed is termed a *shadowed set*, which itself does not use exact membership values but instead employs basic truth values and a zone of uncertainty (the unit interval). A shadowed set could be thought of as an approximation of a fuzzy set or family of fuzzy sets where elements may belong to the set with certainty (membership of 1), possibility (unit interval) or not at all (membership of 0). This can be seen to be analogous to the definitions of the rough set regions: the positive region (certainty), the boundary region (possibility) and the negative region (no membership).

Given a fuzzy set, a shadowed set can be induced by elevating those membership values around 1 and reducing membership values around 0 until a certain threshold level is achieved. Any elements that do not belong to the set with a membership of 1 or 0 are assigned a unit interval, [0,1], considered to be a non-numeric model of membership grade. These regions of uncertainty are referred to as "shadows" (see figure 4.1). In fuzzy set theory, vagueness is distributed across the entire universe of discourse, but in shadowed sets this vagueness is localised in the shadow regions. As with fuzzy sets, the basic set operations (union, intersection and complement) can be defined for shadowed sets, as well as shadowed relations.

Shadowed sets have been applied to domains such as fuzzy clustering and image processing with some success [129]. They are particularly useful in situations where there is a trade-off between numerical precision and computational effort as they reduce the amount of processing involved compared to fuzzy sets. However, there is still a need for a method that uses object membership values when dealing with equivalence classes.

4.1.4 Fuzzy-Rough Reduction Process

Fuzzy-rough set-based Feature Selection (abbreviated FRFS hereafter) builds on the notion of fuzzy lower approximation to enable reduction of datasets containing real-valued features. As will be shown, the process becomes identical to the crisp approach when dealing with nominal well-defined features.

The crisp positive region in traditional rough set theory is defined as the union of the lower approximations. By the extension principle [201], the membership of an object $x \in \mathbb{U}$, belonging to the fuzzy positive region can be defined by

$$\mu_{POS_P(Q)}(x) = \sup_{X \in \mathbb{U}/Q} \mu_{\underline{P}X}(x) \tag{4.10}$$

Object x will not belong to the positive region only if the equivalence class it belongs

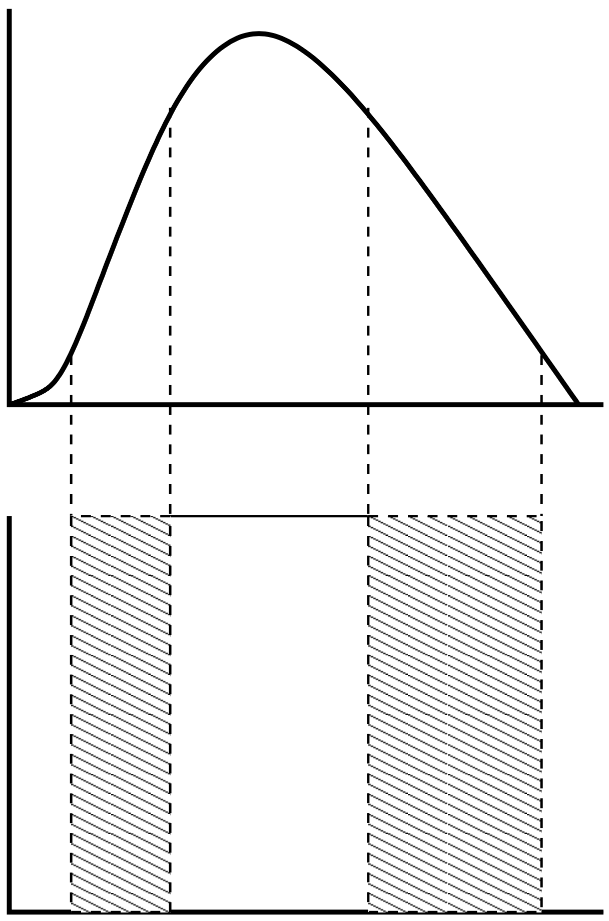


Figure 4.1: A fuzzy set and corresponding shadowed set

to is not a constituent of the positive region. This is equivalent to the crisp version where objects belong to the positive region only if their underlying equivalence class does so. Similarly, the negative and boundary regions can be defined.

Using the definition of the fuzzy positive region, the new dependency function can be defined as follows:

$$\gamma'_P(Q) = \frac{|\mu_{POS_P(Q)}(x)|}{|\mathbb{U}|} = \frac{\sum_{x\in\mathbb{U}} \mu_{POS_P(Q)}(x)}{|\mathbb{U}|} \tag{4.11}$$

As with crisp rough sets, the dependency of Q on P is the proportion of objects that are discernible out of the entire dataset. In the present approach, this corresponds to determining the fuzzy cardinality of $\mu_{POS_P(Q)}(x)$ divided by the total number of objects in the universe.

The definition of dependency degree covers the crisp case as its specific instance. This can be easily shown by recalling the definition of the crisp dependency degree given in (3.6). If a function $\mu_{POS_P(Q)}(x)$ is defined which returns 1 if the object x belongs to the positive region, 0 otherwise, then the above definition may be rewritten as:

$$\gamma_P(Q) = \frac{\sum_{x\in\mathbb{U}} \mu_{POS_P(Q)}(x)}{|\mathbb{U}|} \tag{4.12}$$

which is identical to (4.11).

If the fuzzy-rough reduction process is to be useful, it must be able to deal with multiple features, finding the dependency between various subsets of the original feature set. For example, it may be necessary to be able to determine the degree of dependency of the decision feature(s) with respect to $P = \{a,b\}$. In the crisp case, $\mathbb{U}/P$ contains sets of objects grouped together that are indiscernible according to both features a and b. In the fuzzy case, objects may belong to many equivalence classes, so the cartesian product of $\mathbb{U}/IND(\{a\})$ and $\mathbb{U}/IND(\{b\})$ must be considered in determining $\mathbb{U}/P$. In general,

$$\mathbb{U}/P = \otimes\{a \in P : \mathbb{U}/IND(\{a\})\} \tag{4.13}$$

where

$$A \otimes B = \{X \cap Y : \forall X \in A, \forall Y \in B, X \cap Y \neq \emptyset\} \tag{4.14}$$

Each set in $\mathbb{U}/P$ denotes an equivalence class. For example, if $P = \{a,b\}$, $\mathbb{U}/IND(\{a\}) = \{N_a, Z_a\}$ and $\mathbb{U}/IND(\{b\}) = \{N_b, Z_b\}$, then

$$\mathbb{U}/P = \{N_a \cap N_b, N_a \cap Z_b, Z_a \cap N_b, Z_a \cap Z_b\}$$

The extent to which an object belongs to such an equivalence class is therefore calculated by using the conjunction of constituent fuzzy equivalence classes, say F_i, $i = 1,2,\ldots,n$:

$$\mu_{F_1\cap\ldots\cap F_n}(x) = min(\mu_{F_1}(x), \mu_{F_2}(x), \ldots, \mu_{F_n}(x)) \tag{4.15}$$

4.2 Fuzzy-Rough QUICKREDUCT

A problem may arise when this approach is compared to the crisp approach. In conventional RSAR, a reduct is defined as a subset R of the features which have the same information content as the full feature set A. In terms of the dependency function this means that the values $\gamma(R)$ and $\gamma(A)$ are identical and equal to 1 if the dataset is consistent. However, in the fuzzy-rough approach this is not necessarily the case as the uncertainty encountered when objects belong to many fuzzy equivalence classes results in a reduced total dependency.

FRQUICKREDUCT($\mathbb{C}$,$\mathbb{D}$).
$\mathbb{C}$, the set of all conditional features;
$\mathbb{D}$, the set of decision features.

```
(1)  $R \leftarrow \{\}; \gamma'_{best} = 0; \gamma'_{prev} = 0$
(2)  do
(3)      $T \leftarrow R$
(4)      $\gamma'_{prev} = \gamma'_{best}$
(5)      $\forall x \in (\mathbb{C} - R)$
(6)          if $\gamma'_{R\cup\{x\}}(\mathbb{D}) > \gamma'_T(\mathbb{D})$
(7)              $T \leftarrow R \cup \{x\}$
(8)              $\gamma'_{best} = \gamma'_T(\mathbb{D})$
(9)      $R \leftarrow T$
(10) until $\gamma'_{best} == \gamma'_{prev}$
(11) return $R$
```

Figure 4.2: The fuzzy-rough QUICKREDUCT algorithm

A possible way of combatting this would be to determine the degree of dependency of a set of decision features $\mathbb{D}$ upon the full feature set and use this as the denominator rather than $|\mathbb{U}|$ (for normalization), allowing γ' to reach 1. With these issues in mind, a new QUICKREDUCT algorithm has been developed as given in figure 4.2. It employs the new dependency function γ' to choose which features to add to the current reduct candidate in the same way as the original QUICKREDUCT process (see figure 3.1). The algorithm terminates when the addition of any remaining feature does not increase the dependency (such a criterion could be used with the original QUICKREDUCT algorithm).

As the new degree of dependency measure is non-monotonic, it is possible that the QUICKREDUCT-style search terminates having reached only a local optimum. The global optimum may lie elsewhere in the search space. This motivates the adoption of an alternative search mechanism, presented in section 5.2. However, the algorithm as presented in figure 4.2 is still highly useful in locating good subsets quickly.

It is also possible to reverse the search process in a manner identical to that of REVERSEREDUCT; that is, start with the full set of features and incrementally remove the least informative features. This process continues until no more features can be removed without reducing the total number of discernible objects in the dataset. Again, this tends not to be applied to larger datasets as the cost of evaluating these larger feature subsets is too great.

4.3 Complexity Analysis

Note that an intuitive understanding of QUICKREDUCT implies that, for a dimensionality of n, $(n^2+n)/2$ evaluations of the dependency function may be performed for the worst-case dataset. However, as FRFS is used for dimensionality reduction prior to any involvement of the system which will employ those features belonging to the resultant reduct, this operation has no negative impact upon the run-time efficiency of the system.

In fact, as feature selection can only take place when $n \geq 2$, the base case is n=2. Suppose that the set of conditional features in this case is $\{a_1, a_2\}$, the QUICKREDUCT algorithm makes two initial dependency evaluations (for a_1 and a_2) and a final evaluation for $\{a_1, a_2\}$ (in the worst case). Hence, the order of complexity of the algorithm is 3 (or $(n^2+n)/2$) for n=2.
Suppose that for $n = k$ the order of complexity of the algorithm is

$$\frac{(k^2+k)}{2} \tag{4.16}$$

For $k+1$ features, $\{a_1, ..., a_k, a_{k+1}\}$, QUICKREDUCT makes $k+1$ initial evaluations of the dependency function to determine the best feature (call this a_i). Once a_i is chosen, for the remaining features there are $(k^2+k)/2$ more evaluations in the worst case according to (4.16). Hence, the total number of evaluations for $n = k+1$ is:

$$\frac{k^2+k}{2} + (k+1) \quad = \frac{k^2+3k+2}{2} \quad = \frac{(k+1)^2+(k+1)}{2}$$

The complexity of the algorithm is therefore $O((n^2+n)/2)$ in the worst case. In the best case, the first feature considered results in the maximum degree of dependency for the data, and hence only one heuristic evaluation is performed. In crisp RSAR, the average complexity is seen to be close to linear with the inclusion of several optimizations [30].

4.4 Worked Examples

To illustrate the operation of FRFS, two small example datasets are considered. The first contains real-valued conditional attributes with nominal decisions. In crisp RSAR, the dataset would be discretized using the non-fuzzy sets. However, in the fuzzy-rough approach membership degrees are used in calculating the fuzzy lower approximations and fuzzy positive regions. The second dataset is defined by fuzzy membership values only, with corresponding fuzzy decisions.

In addition to demonstrating the operation of FRFS, a further purpose of presenting these two datasets is to show the applicability of the development to different types of dataset. FRFS is equally applicable to datasets where the conditional values are crisp and decisions are fuzzy, and also to the situation where both conditional and decision attributes are crisp. In the latter case, the operation of FRFS is exactly that of the original crisp RSAR.

4.4.1 Crisp Decisions

Table 4.1 contains three real-values conditional attributes and a crisp-valued decision attribute. To begin with, the fuzzy-rough QUICKREDUCT algorithm initializes the potential reduct (i.e. the current best set of attributes) to the empty set.

Object	a	b	c	q
1	-0.4	-0.3	-0.5	no
2	-0.4	0.2	-0.1	yes
3	-0.3	-0.4	-0.3	no
4	0.3	-0.3	0	yes
5	0.2	-0.3	0	yes
6	0.2	0	0	no

Table 4.1: Example dataset: crisp decisions

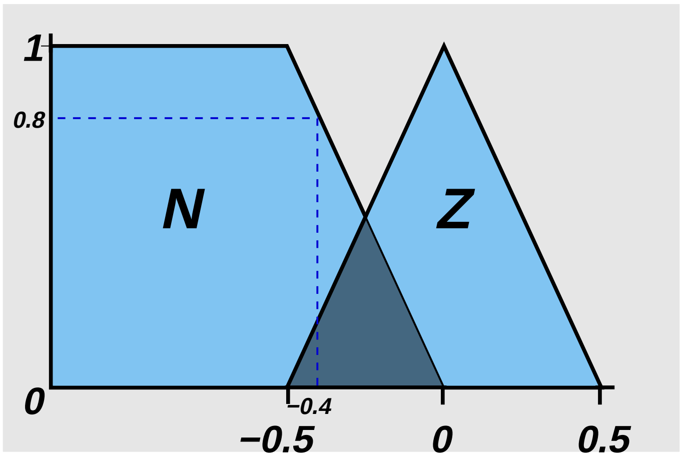


Figure 4.3: Fuzzifications for conditional features

Using the fuzzy sets defined in figure 4.3 (for all conditional attributes), and setting $A = \{a\}$, $B = \{b\}$, $C = \{c\}$ and $Q = \{q\}$, the following equivalence classes are obtained:

$$\begin{aligned} \mathbb{U}/A &= \{N_a, Z_a\} \\ \mathbb{U}/B &= \{N_b, Z_b\} \\ \mathbb{U}/C &= \{N_c, Z_c\} \\ \mathbb{U}/Q &= \{\{1,3,6\},\{2,4,5\}\} \end{aligned}$$

The first step is to calculate the lower approximations of the sets A, B and C, using equation (4.4) in section 4.1.1. To clarify the calculations involved, table 4.2 contains the membership degrees of objects to fuzzy equivalence classes. For simplicity, only A will be considered here; that is, using A to approximate Q. For the first decision

Object	a		b		c		q	
	N_a	Z_a	N_b	Z_b	N_c	Z_c	$\{1,3,6\}$	$\{2,4,5\}$
1	0.8	0.2	0.6	0.4	1.0	0.0	1.0	0.0
2	0.8	0.2	0.0	0.6	0.2	0.8	0.0	1.0
3	0.6	0.4	0.8	0.2	0.6	0.4	1.0	0.0
4	0.0	0.4	0.6	0.4	0.0	1.0	0.0	1.0
5	0.0	0.6	0.6	0.4	0.0	1.0	0.0	1.0
6	0.0	0.6	0.0	1.0	0.0	1.0	1.0	0.0

Table 4.2: Membership values of objects to corresponding fuzzy sets

equivalence class X = {1,3,6}, $\mu_{\underline{A}\{1,3,6\}}(x)$ needs to be calculated:

$$\mu_{\underline{A}\{1,3,6\}}(x) = \sup_{F\in\mathbb{U}/A} min(\mu_F(x), \inf_{y\in\mathbb{U}} max\{1-\mu_F(y), \mu_{\{1,3,6\}}(y)\})$$

Considering the first fuzzy equivalence class of A, N_a:

$$min(\mu_{N_a}(x), \inf_{y\in\mathbb{U}} max\{1-\mu_{N_a}(y), \mu_{\{1,3,6\}}(y)\})$$

For object 2 this can be calculated as follows. From table 4.2 it can be seen that the membership of object 2 to the fuzzy equivalence class N_a, $\mu_{N_a}(2)$, is 0.8. The remainder of the calculation involves finding the smallest of the following values:

$$\begin{array}{lclcl}
\max(1\text{-}\mu_{N_a}(1), \mu_{\{1,3,6\}}(1)) & = & \max(0.2,1.0) & = & 1.0 \\
\max(1\text{-}\mu_{N_a}(2), \mu_{\{1,3,6\}}(2)) & = & \max(0.2,0.0) & = & 0.2 \\
\max(1\text{-}\mu_{N_a}(3), \mu_{\{1,3,6\}}(3)) & = & \max(0.4,1.0) & = & 1.0 \\
\max(1\text{-}\mu_{N_a}(4), \mu_{\{1,3,6\}}(4)) & = & \max(1.0,0.0) & = & 1.0 \\
\max(1\text{-}\mu_{N_a}(5), \mu_{\{1,3,6\}}(5)) & = & \max(1.0,0.0) & = & 1.0 \\
\max(1\text{-}\mu_{N_a}(6), \mu_{\{1,3,6\}}(6)) & = & \max(1.0,1.0) & = & 1.0
\end{array}$$

From the calculations above, the smallest value is 0.2, hence:

$$\begin{array}{lcl}
min(\mu_{N_a}(x), \inf_{y\in\mathbb{U}} max\{1-\mu_{N_a}(y), \mu_{\{1,3,6\}}(y)\}) & = & min(0.8, \inf\{1, 0.2, 1, 1, 1, 1\}) \\
& = & 0.2
\end{array}$$

Similarly for Z_a

$$\begin{aligned} min(\mu_{Z_a}(x), \inf_{y \in \mathbb{U}} max\{1-\mu_{Z_a}(y), \mu_{\{1,3,6\}}(y)\}) &= min(0.2, \inf\{1, 0.8, 1, 0.6, 0.4, 1\} \\ &= 0.2 \end{aligned}$$

Thus,

$$\mu_{\underline{A}\{1,3,6\}}(2) = 0.2$$

Calculating the A-lower approximation of $X = \{1,3,6\}$ for every object gives

$$\begin{array}{ll} \mu_{\underline{A}\{1,3,6\}}(1) = 0.2 & \mu_{\underline{A}\{1,3,6\}}(2) = 0.2 \\ \mu_{\underline{A}\{1,3,6\}}(3) = 0.4 & \mu_{\underline{A}\{1,3,6\}}(4) = 0.4 \\ \mu_{\underline{A}\{1,3,6\}}(5) = 0.4 & \mu_{\underline{A}\{1,3,6\}}(6) = 0.4 \end{array}$$

The corresponding values for $X = \{2,4,5\}$ can also be determined:

$$\begin{array}{ll} \mu_{\underline{A}\{2,4,5\}}(1) = 0.2 & \mu_{\underline{A}\{2,4,5\}}(2) = 0.2 \\ \mu_{\underline{A}\{2,4,5\}}(3) = 0.4 & \mu_{\underline{A}\{2,4,5\}}(4) = 0.4 \\ \mu_{\underline{A}\{2,4,5\}}(5) = 0.4 & \mu_{\underline{A}\{2,4,5\}}(6) = 0.4 \end{array}$$

It is a coincidence here that $\mu_{\underline{A}\{2,4,5\}}(x) = \mu_{\underline{A}\{1,3,6\}}(x)$ for this example. Using these values, the fuzzy positive region for each object can be calculated via using

$$\mu_{POS_A(Q)}(x) = \sup_{X \in \mathbb{U}/Q} \mu_{\underline{A}X}(x)$$

This results in:

$$\begin{array}{ll} \mu_{POS_A(Q)}(1) = 0.2 & \mu_{POS_A(Q)}(2) = 0.2 \\ \mu_{POS_A(Q)}(3) = 0.4 & \mu_{POS_A(Q)}(4) = 0.4 \\ \mu_{POS_A(Q)}(5) = 0.4 & \mu_{POS_A(Q)}(6) = 0.4 \end{array}$$

The next step is to determine the degree of dependency of Q on A:

$$\gamma'_A(Q) = \frac{\sum_{x \in U} \mu_{POS_A(Q)}(x)}{|U|} = 2/6$$

Calculating for B and C gives:

$$\gamma'_B(Q) = \frac{2.4}{6}, \ \gamma'_C(Q) = \frac{1.6}{6}$$

From this it can be seen that attribute b will cause the greatest increase in dependency degree. This attribute is chosen and added to the potential reduct. The process iterates and the two dependency degrees calculated are

$$\gamma'_{\{a,b\}}(Q) = \frac{3.4}{6}, \ \gamma'_{\{b,c\}}(Q) = \frac{3.2}{6}$$

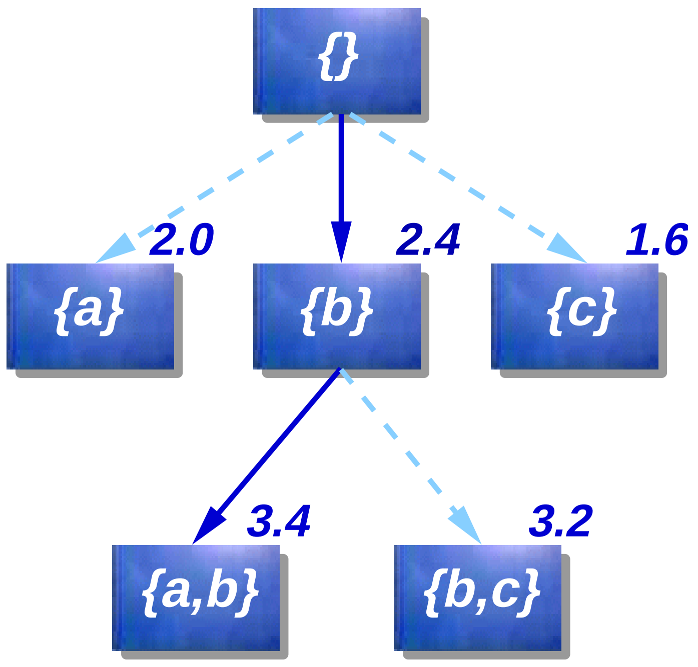


Figure 4.4: Path taken by the fuzzy-rough QUICKREDUCT algorithm

Adding attribute *a* to the reduct candidate causes the larger increase of dependency, so the new candidate becomes $\{a,b\}$. Lastly, attribute *c* is added to the potential reduct:

$$\gamma_{\{a,b,c\}}(Q) = \frac{3.4}{6}$$

As this causes no increase in dependency, the algorithm stops and outputs the reduct $\{a,b\}$. The steps taken by the fuzzy-rough QUICKREDUCT algorithm in reaching this decision can be seen in figure 4.4. The dataset can now be reduced to only those attributes appearing in the reduct. When crisp RSAR is performed on this dataset (after using the same fuzzy sets to discretize the real-valued attributes), the reduct generated is $\{a,b,c\}$, i.e. the full conditional attribute set [73]. Unlike crisp RSAR, the true minimal reduct was found using the information on degrees of membership. It is clear from this example alone that the information lost by using crisp RSAR can be important when trying to discover the smallest reduct from a dataset.

Object	a			b			c		$Plan$		
	$A1$	$A2$	$A3$	$B1$	$B2$	$B3$	$C1$	$C2$	X	Y	Z
1	0.3	0.7	0.0	0.2	0.7	0.1	0.3	0.7	0.1	0.9	0.0
2	1.0	0.0	0.0	1.0	0.0	0.0	0.7	0.3	0.8	0.2	0.0
3	0.0	0.3	0.7	0.0	0.7	0.3	0.6	0.4	0.0	0.2	0.8
4	0.8	0.2	0.0	0.0	0.7	0.3	0.2	0.8	0.6	0.3	0.1
5	0.5	0.5	0.0	1.0	0.0	0.0	0.0	1.0	0.6	0.8	0.0
6	0.0	0.2	0.8	0.0	1.0	0.0	0.0	1.0	0.0	0.7	0.3
7	1.0	0.0	0.0	0.7	0.3	0.0	0.2	0.8	0.7	0.4	0.0
8	0.1	0.8	0.1	0.0	0.9	0.1	0.7	0.3	0.0	0.0	1.0
9	0.3	0.7	0.0	0.9	0.1	0.0	1.0	0.0	0.0	0.0	1.0

Table 4.3: Example dataset containing fuzzy values

4.4.2 Fuzzy Decisions

Using the fuzzy-rough QUICKREDUCT algorithm, table 4.3 can be reduced in size. First of all the lower approximations need to be determined. Consider the first feature in the dataset; setting P = $\{a\}$ produces the fuzzy partitioning $\mathbb{U}/P = \{A1, A2, A3\}$. Additionally, setting Q = $\{Plan\}$ produces the fuzzy partitioning $\mathbb{U}/Q = \{X, Y, Z\}$. To determine the fuzzy P-lower approximation of Plan X ($\mu_{\underline{P}X}(x)$), each $F \in \mathbb{U}/P$ must be considered. For $F = A1$:

$$min(\mu_{A1}(x), \inf_{y \in \mathbb{U}} max\{1 - \mu_{A1}(y), \mu_X(y)\}) = min(\mu_{A1}(x), 0.6)$$

Similarly, for $F = A2$, $min(\mu_{A2}(x), 0.3)$ and $F = A3$, $min(\mu_{A3}(x), 0.0)$. To calculate the extent to which an object x in the dataset belongs to the fuzzy P-lower approximation of X, the union of these values is calculated. For example, object 1 belongs to $\underline{P}X$ with a membership of:

$$sup\{min(\mu_{A1}(1), 0.6), min(\mu_{A2}(1), 0.3), min(\mu_{A3}(1), 0.0)\} = 0.3.$$

Likewise, for Y and Z:

$$\mu_{\underline{P}Y}(1) = 0.2 \quad \mu_{\underline{P}Z}(1) = 0.3$$

The extent to which object 1 belongs to the fuzzy positive region can be determined by considering the union of fuzzy P-lower approximations:

$\mu_{POS_P(Q)}(1) = \sup_{S \in \mathbb{U}/Q} \mu_{\underline{P}S}(1) = 0.3$

Similarly, for the remaining objects,

$$\begin{array}{ll} \mu_{POS_P(Q)}(2) = 0.6 & \mu_{POS_P(Q)}(3) = 0.3 \\ \mu_{POS_P(Q)}(4) = 0.6 & \mu_{POS_P(Q)}(5) = 0.5 \\ \mu_{POS_P(Q)}(6) = 0.3 & \mu_{POS_P(Q)}(7) = 0.6 \\ \mu_{POS_P(Q)}(8) = 0.3 & \mu_{POS_P(Q)}(9) = 0.3 \end{array}$$

Using these values, the new degree of dependency of Q on $P = \{a\}$ can be calculated:

$$\gamma'_P(Q) = \frac{\sum_{x \in \mathbb{U}} \mu_{POS_P(Q)}(x)}{|\{1,2,3,4,5,6,7,8,9\}|} = 3.8/9$$

The fuzzy-rough QUICKREDUCT algorithm uses this process to evaluate subsets of features in an incremental fashion. The algorithm starts with an empty set and considers the addition of each individual feature:

$\gamma'_{\{a\}}(Q) = 3.8/9$
$\gamma'_{\{b\}}(Q) = 2.1/9$
$\gamma'_{\{c\}}(Q) = 2.7/9$

As feature a causes the greatest increase in dependency degree, it is added to the reduct candidate and the search progresses:

$\gamma'_{\{a,b\}}(Q) = 4.0/9$,
$\gamma'_{\{a,c\}}(Q) = 5.7/9$

Here, c is added to the reduct candidate as the dependency is increased. There is only one feature addition to be checked at the next stage, namely

$\gamma'_{\{a,b,c\}}(Q) = 5.7/9$

This causes no dependency increase, resulting in the algorithm terminating and outputting the reduct $\{a,c\}$. Hence, the original dataset can be reduced to these features with minimal information loss (according to the algorithm).

4.5 Optimizations

There are several optimizations that can be implemented to speed up the FRFS process. The original definition of the fuzzy positive region, given in equation (4.10), can be more explicitly defined as:

$$\mu_{POS_P(Q)}(x) = \sup_{X \in \mathbb{U}/Q} \left\{ \sup_{F \in \mathbb{U}/P} min(\mu_F(x), \inf_{y \in \mathbb{U}} max\{1 - \mu_F(y), \mu_X(y)\}) \right\} \tag{4.17}$$

where P is a subset of the conditional attributes, Q the decision attribute(s). In order to speed up computation time, equation (4.17) can be rewritten as:

$$\mu_{POS_P(Q)}(x) = \sup_{F \in \mathbb{U}/P} \left\{ min(\mu_F(x), \sup_{X \in \mathbb{U}/Q} \{\inf_{y \in \mathbb{U}} max(1 - \mu_F(y), \mu_X(y))\}) \right\} \tag{4.18}$$

This reformulation helps to speed up the calculation of the fuzzy positive region by considering each fuzzy equivalence class F in U/P first. If the object x is found not to belong to F, the remainder of the calculations for this class need not be evaluated, due to the use of the *min* operator. This can save substantial time, as demonstrated in table 4.4, where the two definitions of the positive region are used to determine reducts from several small to large datasets. The times here are the times taken for each version of FRFS to find a reduct. Each version of FRFS will follow exactly the same route and will locate identical reducts, hence the results are comparable. All the datasets are from the Machine Learning Repository [20] and contain real-valued conditional features with nominal classifications.

Additionally in table 4.4, average runtimes are given for the optimized implementation of the fuzzy-rough feature selector (labelled Opt. in the table). This includes the use of the algorithm presented in figure 4.5, which is designed to result in the faster

Dataset	No. of Features	Eq. (4.17) (s)	Eq. (4.18) (s)	Opt. (s)
Glass	10	29.5	26.7	7.18
Wine	14	5.41	3.05	2.20
Olitos	26	47.6	21.9	13.0
JobSat	27	19.2	5.75	2.72
Ionosphere	35	204.5	107.8	76.9
Selwood	54	57.5	15.9	5.64
Isolet	618	368.4	131.9	47.2
Phenetyl	629	740.7	145.0	70.3
Caco	714	2709.3	213.5	114.1

Table 4.4: Experimental comparison of the two formulations for the calculation of the positive region

computation of the fuzzy-rough metric for small feature subsets. Excess computation is avoided at lines (4) and (6) which exploit the nature of t-norms and s-norms.

```
CALCULATEGAMMA'($\mathbb{P}$,$\mathbb{Q}$).
$\mathbb{P}$, the current feature subset;
$\mathbb{Q}$, the set of decision features.

    (1)  $mems[], \gamma' \leftarrow 0$
    (2)  $\forall F \in \mathbb{U}/P$
    (3)      $deg = \sup_{X \in \mathbb{U}/Q} (\{\inf_{y \in \mathbb{U}} max\{1 - \mu_F(y), \mu_X(y)\}\}$
    (4)      if $deg$ != 0
    (5)          $\forall o \in \mathbb{U}$
    (6)              if $mems[o]$ != 1 && $deg > mems[o]$
    (7)              then $mems[o] = max(min(\mu_F(o), deg), mems[o])$
    (8)  $\forall o \in \mathbb{U}, \gamma'$ += $mems[o]$
    (9)  return $\gamma'$
```

Figure 4.5: Optimized γ' calculation for small subsets

4.6 Evaluating the Fuzzy-Rough Metric

In order to evaluate the utility of the new fuzzy-rough measure of feature significance, a series of artificial datasets were generated and used for comparison with 5 other leading feature ranking measures. The datasets were created by generating around 30 random feature values for 400 objects. Two or three features (referred to as x, y, or z) are chosen to contribute to the final boolean classification by means of an inequality. For example, in table 4.6, if the inequality $(x+y)^2 > 0.25$ holds for an object then it is classified as 1, with a classification of 0 otherwise. The task for the feature rankers was to discover those features that are involved in the inequalities, ideally rating the other irrelevant features poorly in contrast.

The tables presented in the metric comparison section show the ranking given to the features that are involved in the inequality that determines the classification. The final row indicates whether all the other features are given a ranking of zero. The full results can be seen in appendix B. For the data presented in table 4.5, the first feature, x, is used to determine the classification. The values of features y and z are derived from x: $y = \sqrt{x}$, $z = x^2$.

4.6.1 Compared Metrics

The metrics compared are: the fuzzy-rough measure (FR), Relief-F (Re), Information Gain (IG), Gain Ratio (GR), OneR (1R) and the statistical measure χ^2. Metrics other than the fuzzy-rough measure were obtained from [186]. A brief description of each is presented next.

4.6.1.1 Information Gain

The Information Gain (IG) [67] is the expected reduction in entropy resulting from partitioning the dataset objects according to a particular feature. The entropy of a labelled collection of objects S is defined as:

$$Entropy(S) = \sum_{i=1}^{c} -p_i log_2 p_i \tag{4.19}$$

where p_i is the proportion of S belonging to class i. Based on this, the Information Gain metric is:

$$IG(S,A) = Entropy(S) - \sum_{v \in values(A)} \frac{|S_v|}{|S|} Entropy(S_v) \tag{4.20}$$

where $values(A)$ is the set of values for feature A, S the set of training examples, S_v the set of training objects where A has the value v. This metric is the one used in ID3 [134] for selecting the best feature to partition the data.

4.6.1.2 Gain Ratio

One limitation of the IG measure is that it favours features with many values. The Gain Ratio (GR) seeks to avoid this bias by incorporating another term, split information, that is sensitive to how broadly and uniformly the attribute splits the considered data:

$$Split(S,A) = -\sum_{i=1}^{c} \frac{|S_i|}{|S|} log_2 \frac{|S_i|}{|S|} \tag{4.21}$$

where each S_i is a subset of objects generated by partitioning S with the c-valued attribute A. The Gain Ratio is then defined as follows:

$$GR(S,A) = \frac{IG(S,A)}{Split(S,A)} \tag{4.22}$$

4.6.1.3 χ^2 Measure

In the χ^2 method [101], features are individually evaluated according to their χ^2 statistic with respect to the classes. For a numeric attribute, the method first requires its range to be discretized into several intervals. The χ^2 value of an attribute is defined as:

$$\chi^2 = \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{(A_{ij} - E_{ij})^2}{E_{ij}} \tag{4.23}$$

where m is the number of intervals, k the number of classes, A_{ij} the number of samples in the ith interval, jth class, R_i the number of objects in the ith interval, C_j the number of objects in the jth class, N the total number of objects, and E_{ij} the expected frequency of $A_i j$ (E_{ij} = R_i * C_j/N). The larger the χ^2 value, the more important the feature.

4.6.1.4 Relief-F

This is the Relief-F measure, based on the original Relief measure described in section 2.2.1.1. Relief evaluates the worth of an attribute by repeatedly sampling an instance and considering the value of the given attribute for the nearest instance of the same and different class. Relief-F extends this idea to dealing with multi-class problems as well as handling noisy and incomplete data.

4.6.1.5 OneR

The OneR classifier [65] learns a one-level decision tree, i.e. it generates a set of rules that test one particular attribute. One branch is assigned for every value of a feature; each branch is assigned the most frequent class. The error rate is then defined as the proportion of instances that do not belong to the majority class of their corresponding branch. Features with the higher classification rates are considered to be more significant than those resulting in lower accuracies.

4.6.2 Metric Comparison

The tables presented here are summaries of those given in appendix B. From the results in table 4.5, it can be observed that all metrics successfully rank the influential features highest. IG, GR, 1R and χ^2 rank these features equally, whereas Re and FR rank feature z higher. Only FR, IG, GR and χ^2 rate all the other features as zero.

Feature	FR	Re	IG	GR	1R	χ^2
x	0.5257	0.31758	0.997	1.0	99.5	200
y	0.5296	0.24586	0.997	1.0	99.5	200
z	0.5809	0.32121	0.997	1.0	99.5	200
others	$= 0$	$\neq 0$	$= 0$	$= 0$	$\neq 0$	$= 0$

Table 4.5: Feature evaluation for $x > 0.5$, $y = \sqrt{x}$, $z = x^2$

As can be seen from these results, feature rankers can discover the influential features but on their own are incapable of determining multiple feature interactions. Table 4.5 could be reduced to one feature only (either x, y, or z) without any loss of information as only these contribute to the classification. However, the rankers all rate these features highly and would only provide enough information to reduce the data to at least these three attributes. Here, the rankers have found the predictive (or relevant) features but have been unable to determine which of these are redundant.

Feature	FR	Re	IG	GR	1R	χ^2
x	0.2330	0.1862	0.2328	0.1579	86.75	128.466
y	0.2597	0.1537	0.1687	0.1690	87.75	71.971
others	$\neq 0$	$\neq 0$	$\neq 0$	$\neq 0$	$\neq 0$	$\neq 0$

Table 4.6: Feature evaluation for $(x+y)^2 > 0.25$

Table 4.6 shows the results for the inequality $(x+y)^2 > 0.25$. Both features x and y are required for deciding the classification. All feature rankers evaluated detect this. FR, IG, GR, 1R and χ^2 also rank the tenth feature highly - probably due to a chance correlation with the decision. The results in table 4.7 are for a similar inequality, with all the feature rankers correctly rating the important features. FR, IG, GR and χ^2 evaluate the remaining features as having zero significance.

Feature	FR	Re	IG	GR	1R	χ^2
x	0.2090	0.140067	0.241	0.156	79.0	119.562
y	0.2456	0.151114	0.248	0.165	78.25	122.336
others	= 0	$\neq 0$	= 0	= 0	$\neq 0$	= 0

Table 4.7: Feature evaluation for $(x+y)^2 > 0.5$

In table 4.8, all metrics apart from 1R locate the relevant features. For this dataset, 1R chooses 22 features as being the most significant, whilst ranking features x and y last. This may be due to the discretization process that must precede the application of 1R. If the discretization is poor, then the resulting feature evaluations will be affected.

Feature	FR	Re	IG	GR	1R	χ^2
x	0.2445	0.1486	0.134	0.134	87.75	57.455
y	0.2441	0.1659	0.159	0.164	87.25	73.390
others	= 0	$\neq 0$	= 0	= 0	$\neq 0$	= 0

Table 4.8: Feature evaluation for $(x+y)^3 < 0.125$

Tables 4.9 shows the results for data classified by $x*y*z > 0.125$. All feature rankers correctly detect these variables. However, in table 4.10 the results can be seen for the same inequality but with the impact of variable z increased. All metrics determine that *z* has the most influence on the decision, and almost all choose *x* and *y* next. Again, the 1R measure fails and chooses features 15, 19 and 24 instead.

Feature	FR	Re	IG	GR	1R	χ^2
x	0.1057	0.0750547	0.169	0.123	64.25	73.653
y	0.0591	0.1079423	0.202	0.226	66.75	88.040
z	0.1062	0.0955878	0.202	0.160	67.50	84.283
others	= 0	$\neq 0$	= 0	= 0	$\neq 0$	= 0

Table 4.9: Feature evaluation for $x * y * z > 0.125$

Feature	FR	Re	IG	GR	1R	χ^2
x	0.1511	0.0980	0.1451	0.0947	76.5	65.425
y	0.1101	0.0557	0.0909	0.1080	78.0	35.357
z	0.2445	0.1474	0.2266	0.2271	79.75	93.812
others	= 0	$\neq 0$	= 0	= 0	$\neq 0$	= 0

Table 4.10: Feature evaluation for $x * y * z^2 > 0.125$

Only the FR and Re metrics are applicable to datasets where the decision feature is continuous. Results from the application of these two measures to such data can be found in appendix C. Both methods find the features that are involved in generating the decision values.

This short investigation into the utility of the new fuzzy-rough measure has shown that it is comparable with the leading measures of feature importance. Indeed, its behaviour is quite similar to the information gain and gain ratio metrics. This is interesting as both of these measures are entropy-based. As mentioned in section 2.2.1.5, a feature subset with a maximum (crisp) rough set dependency has a corresponding entropy of 0. Unlike these metrics, the fuzzy-rough measure may also be applied to datasets containing real-valued decision features.

4.7 Application to Financial Data

The comparisons carried out in section 4.6 concerning the fuzzy-rough metric used artificial data, where the relevancy of features was known beforehand. To demonstrate the utility of the new measure in determining relevant features from real world data, a financial dataset (the Trust Crisis Dataset) was chosen for analysis. An expert was asked to define a ranking of feature importance for the data in order to compare this with the automated ranking produced by the metrics.

4.7.1 The Trust Crisis Dataset

An investigation was carried out in [94] in an attempt to analyse the split capital investment trust crisis of 2001/2002. This analysis sought to examine what factors were behind the failure of a number of these trusts. Many retail investors, thinking that they were investing in low risk financial products, were left facing substantial losses in their investments. This ultimately led to a hearing by the Treasury Select Committee. Although trust managers apportioned blame to falling equity markets, leading analysts contended that this was an accident waiting to happen.

In addition to other analyses, the study investigated the effects of 18 different financial-related aspects (features) of the trusts to determine how these may have contributed to trust failure. It is therefore interesting to compare an expert-defined ordering of feature influence on failure with the automated ordering of feature ranking. The features involved in the analysis are given in table 4.11, with their corresponding ordering of importance. According to the expert, the most influential features are Split & High Yielding Trusts, though Equities, Fixed Interest & Convertible Stock, and Cash are correlated with this. Bank Debt is the next most influential feature. Although an ordering is given to Management Fees, Charge to Capital and Admin Fees, it is difficult to differentiate between them.

4.7.2 Results

The results of the application of the feature rankers as discussed in section 4.6 can be seen in table 4.12. All metrics rate features 8 (Equities) and 9 (Split & High Yielding Trusts) highly. This is in agreement with the expert-supplied rank. All measures apart from FR rank 9 first, with feature 8 second. However, only the FR metric correctly rates features 10 (Fixed Interest & Convertible Stock) and 11 (Cash) highly. After

Feature Number	Feature Name	Expert-defined Ordering
0	Bank Debt	2
1	Bank Interest Rate	6
2	Zero Dividend Preference Shares (ZDPs)	6
3	Income Shares	6
4	Capital Shares	6
5	Ordinary Income & Residual Capital Shares	6
6	Other Share Classes	6
7	Gross Assets	6
8	Equities	1
9	Split & High Yielding Trusts	1
10	Fixed Interest & Convertible Stock	1
11	Cash (estimate)	1
12	Debtors less Creditors	6
13	Total Dividends	6
14	Management Fees	4
15	Charge to Capital	3
16	Admin Fees	5
17	SORP Compliance	6

Table 4.11: Features in the trust crisis dataset

these attributes, FR ranks Bank Debt next. In fact, this is the only measure to detect the importance of this feature.

Spearman's rank correlation coefficient, denoted ρ, is often used to determine the nature of a relationship between two variables. This is useful here to examine the correlation of each metric ranking with that of the expert's ranking. The value ρ is calculated in the following way:

$$\rho = 1 - \frac{6\sum d^2}{n(n^2-1)} \tag{4.24}$$

where d is the difference between the ranks of the corresponding values of the variables, and n is the number of pairs of values. The values of ρ for the rankings produced by each metric can be found in table 4.13. From this it can be seen that all metrics

Feature	Expert	FR	Re	IG	GR	1R	χ^2
8	1	0.205	0.130	0.146	0.150	78.5	18.2
9	1	0.169	0.183	0.309	0.363	83.9	51.0
10	1	0.158	0.036	0.0	0.0	77.6	0.0
11	1	0.109	0.040	0.0	0.0	73.2	0.0
0	2	0.102	0.013	0.0	0.0	76.7	0.0
15	3	0.0	0.036	0.0	0.0	77.6	0.0
14	4	0.0	0.020	0.0	0.0	77.6	0.0
16	5	0.0	0.003	0.0	0.0	77.6	0.0
1	6	0.062	0.062	0.0	0.0	75.0	0.0
2	6	0.099	0.012	0.0	0.0	75.8	0.0
3	6	0.0	0.009	0.0	0.0	75.8	0.0
4	6	0.0	0.008	0.0	0.0	76.7	0.0
5	6	0.023	0.014	0.0	0.0	74.1	0.0
6	6	0.0	0.004	0.0	0.0	77.6	0.0
7	6	0.0	0.006	0.0	0.0	71.4	0.0
12	6	0.007	0.007	0.0	0.0	75.8	0.0
13	6	0.088	0.005	0.0	0.0	75.0	0.0
17	6	0.031	0.085	0.001	0.001	77.6	0.126

Table 4.12: Feature ranker results for the trust crisis dataset

exhibit a positive correlation, with the fuzzy-rough metric resulting in the strongest correlation.

	FR	Re	IG	GR	1R	χ^2
ρ	0.61249	0.58566	0.58772	0.58772	0.55624	0.58772

Table 4.13: Spearman's rank correlation results

To determine the significance of these values, they must be compared with the critical values of ρ with $(n-2)$ degrees of freedom. At the 0.01, 0.02 and 0.05 levels of significance the critical values are 0.625, 0.564 and 0.475 respectively for this data. To illustrate what this means, consider the following example. If the value of ρ for a set of data containing the same number of samples is 0.75, then it can be observed that this is larger than the critical value of ρ at the 0.01 level of significance (0.625).

It could then be concluded that the obtained value of ρ is likely to occur by chance less than one time in a hundred (i.e. it is highly significant). All the metrics are above the 0.05 level, but only the 1R metric falls below the 0.02 significance level. There is less confidence that the correlation has not occurred by chance for 1R. The remainder of the metrics fall between the 0.01 and 0.02 significance levels, with the FR metric closest to the 0.01 level.

These results show that the FR metric is a useful gauger of information, producing results very much in line with the expert ranking. Out of all the methods, FR appears to produce an ordering of the features closest to the expert's.

4.8 Summary

In this chapter, a method for feature selection based on fuzzy-rough sets has been presented. An algorithm for finding feature subsets, based on the new fuzzy-rough dependency measure, was introduced and illustrated by means of two simple examples. The examples were chosen to demonstrate the fact that FRFS can be applied to datasets containing crisp or real-valued features or a mixture of both. Implemented optimizations were briefly discussed that can significantly improve the runtime of FRFS. This was demonstrated by comparing the execution times of the different implementations on real-world data. One real-world and several artificial datasets were also used to evaluate the utility of the fuzzy-rough measure and provide comparisons with other leading feature importance measures. The results show that the new metric presented here is slightly better than the leading measures at locating the relevant features.

Chapter 5

Developments of FRFS

This chapter discusses several FRFS-based developments. Firstly, a new area in feature selection, fuzzy set-based feature grouping, is presented. Although the method given here is used within fuzzy-rough feature selection, it can be used with any feature selector that employs a suitable evaluation function. Indeed, it may also be applied to the standard crisp rough set-based method, RSAR. Also in this chapter, a novel framework for selecting features via Ant Colony Optimization [21, 40] is detailed. This is applied to finding optimal fuzzy-rough subsets, but can replace the search mechanism for most feature selection methods.

5.1 Feature Grouping

By its definition, the degree of dependency measure (whether using crisp or fuzzy-rough sets) always lies in the range [0,1], with 0 indicating no dependency and 1 indicating total dependency. For example, two subsets of the conditional attributes in a dataset may have the following dependency degrees:

$$\gamma'_{\{a,b,c\}}(\mathbb{D}) = 0.54, \;\; \gamma'_{\{a,c,d\}}(\mathbb{D}) = 0.52$$

In traditional rough sets, it would be said that the attribute set $\{a,b,c\}$ has a higher dependency value than $\{a,c,d\}$ and so would make the better candidate to produce a minimal reduct. This may not be the case when considering real datasets that contain noise and other discrepancies. In fact, it is possible that $\{a, c, d\}$ is the best candidate for this and other unseen related datasets. By fuzzifying the output values of the dependency function this problem may be successfully tackled. In addition to this, attributes may be *grouped* at stages in the selection process depending on their dependency label, speeding up the reduct search.

5.1.1 Fuzzy Dependency

In order to achieve this, several fuzzy sets must be defined over the dependency range (for example, Fig. 5.1). This leads to the next problem area: how are these sets to be defined? There is also the problem of how many fuzzy sets should be used to produce the most useful results. Initially, these may be defined beforehand by an expert and refined through experimentation. However, to fit in with the rough set ideology, it would be interesting to investigate how to automatically generate these sets purely from the dataset itself. This could be achieved through fuzzy clustering techniques such as fuzzy c-means [17, 45], for example. For the time being, it is assumed that these fuzzy sets have already been defined.

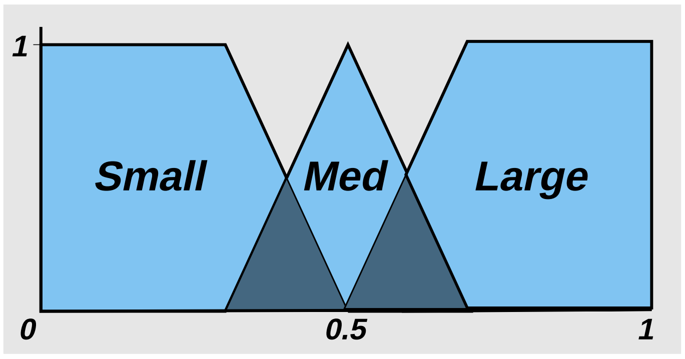


Figure 5.1: Possible fuzzification of dependency

The goal of FRFS and RSAR is to find a (possibly minimal) subset of the conditional attributes for which the dependency metric is at a maximum (ideally the value 1). In the case of fuzzy equivalence classes, where an element of uncertainty is introduced, the maximum degree of dependency may be substantially less than this. In fact, the maximum dependency for different datasets may be quite different due to differing levels of uncertainty. The maximum for dataset A may be 0.9 whereas for dataset B the maximum may be only 0.2. Given a degree of dependency of 0.19, for dataset A this is quite a small value but for dataset B this is quite large, so some way of scaling the dependency value depending on the dataset is required.

5.1.2 Scaled Dependency

The following is one potential way of achieving this for a subset P of all conditional attributes $\mathbb{C}$:

$$\gamma''_P(\mathbb{D}) = \frac{\gamma'_P(\mathbb{D})}{\gamma'_{\mathbb{C}}(\mathbb{D})}$$

In the example above, the scaled dependency degree for dataset A is now 0.21 (which fuzzifies to *Small*) and for dataset B is 0.95 (which fuzzifies to *Large*). However, a further problem is encountered as the search for a reduct nears its conclusion. In this situation, almost all of the dependency values are mapped to *Large* due to their underlying closeness in value. This means that too large a group of attributes will be selected every time. Additionally, if the data is noisy it may be the case that $\gamma''_P(\mathbb{D}) > 1$ as the dependency degree of the full set of conditional attributes may be greater than that of a particular attribute subset. An alternative scaling approach to combat both of these problems is to use the extreme values at each level of search. As soon as the reduct candidates have been evaluated, the highest and lowest dependencies ($\gamma'_{high}(\mathbb{D})$ and $\gamma'_{low}(\mathbb{D})$) are used as follows to scale the dependency degree of subset P:

$$\gamma''_P(\mathbb{D}) = \frac{\gamma'_P(\mathbb{D}) - \gamma'_{low}(\mathbb{D})}{\gamma'_{high}(\mathbb{D}) - \gamma'_{low}(\mathbb{D})}$$

This type of expression is also called an *index* [16]. By this method, the attribute subset with the highest dependency value will have a scaled dependency ($\gamma''_P(\mathbb{D})$) of 1. The subset with the lowest will have a scaled dependency of 0. In so doing, the definition of the fuzzy sets need not be changed for different datasets; one definition should be applicable to all.

The next question to address is how to handle those scaled dependencies that fall at the boundaries. For example, a value may partially belong to both *Small* and *Medium*. A simple strategy is to choose the single fuzzy label with the highest membership value. However, this loses the potentially useful information of dual fuzzy set membership. Another strategy is to take both labels as valid, considering both possibilities within the feature selection process. If, for example, a dependency value lies within the labels *Small* and *Medium* then it is considered to belong to both groups.

5.1.3 The Feature Grouping Algorithm

The new fuzzy-rough QuickReduct algorithm (FQR) which employs scaling and fuzzy dependencies can be seen in Fig. 5.2. In this algorithm, *Cands* contains sets of attributes and their corresponding dependency degrees when added to the current reduct candidate. Once each remaining attribute has been evaluated, the dependencies are scaled according to γ'_{high} and γ'_{low}. Next, the decision is made on which feature(s) to add to the current reduct. In the previous fuzzy-rough QuickReduct algorithm, this would amount to selecting the feature providing the highest gain in dependency degree. Here, other strategies may be employed; for example, attributes may be selected individually or in groups. This is discussed in more detail next.

Note that, in addition to applying this method to fuzzy-rough feature selection, it may also be applied to crisp RSAR. Given a dataset containing crisp values, the dependency values may be fuzzified similarly (with scaling) so that groups of attributes may be selected at one time. The algorithm for this is exactly the same as the one given in figure 5.2, except the dependency function used is now based on crisp rough sets. Additionally, this may be applied to any feature selector where the result of subset evaluation is a value in the range $[0,1]$.

5.1.4 Selection Strategies

When using fuzzy degrees of dependency, it is possible to change strategy at any stage of the attribute selection process. The main distinction to make in the set of possible strategies is whether features are chosen individually or in groups.

FUZZYQUICKREDUCT($\mathbb{C}$,$\mathbb{D}$).
$\mathbb{C}$, the set of all conditional attributes;
$\mathbb{D}$, the set of decision attributes.

```
(1)  $R \leftarrow \{\}; \gamma'_{best} = 0; \gamma'_{prev} = 0$
(2)  **do**
(3)    $Cands \leftarrow \{\}$
(4)    $\gamma'_{prev} = \gamma'_{best}$
(5)    $\gamma'_{high} = 0; \gamma'_{low} = 1$
(5)    $\forall x \in (\mathbb{C} - R)$
(6)      $T \leftarrow R \cup \{x\}$
(7)      $Cands \leftarrow Cands \cup (x, \gamma'_T(\mathbb{D}))$
(8)      **if** $\gamma'_T(\mathbb{D}) > \gamma'_{high}$
(9)        $\gamma'_{high} = \gamma'_T(\mathbb{D})$
(10)     **else if** $\gamma'_T(\mathbb{D}) < \gamma'_{low}$
(11)       $\gamma'_{low} = \gamma'_T(\mathbb{D})$
(12)   $Cands \leftarrow$ scale($Cands, \gamma'_{high}, \gamma'_{low}$)
(13)   $R \leftarrow R \cup$ selectFeatures($Cands$)
(14)   $\gamma'_{best} = \gamma'_R(\mathbb{D})$
(15) **until** $\gamma'_{best} == \gamma'_{prev}$
(16) **return** $R$
```

Figure 5.2: The new fuzzy-rough QUICKREDUCT algorithm with fuzzy dependencies

5.1.4.1 Individuals

In this subset of strategies, attributes are chosen one at a time in a similar fashion to that of FRFS. However, the choice of attribute depends on its corresponding linguistic label(s) and membership obtained from the dependency degree. In the example fuzzification of dependency given in Fig. 5.1, attributes are grouped into the categories *Small*, *Medium* and *Large*. A representative attribute of the required label can be chosen randomly or based on the extent to which the attribute belongs to the fuzzy set itself. Those individual attributes lying on set boundaries are assigned both fuzzy labels. Other issues include which particular group of attributes to consider. Intuitively, it would seem most appropriate to consider those belonging to the *Large* group only, however it may be worthwhile investigating *Small* and *Medium*-grouped attributes at different stages of the search process.

5.1.4.2 Grouping

To speed up the reduct search process, many attributes may be added to a reduct candidate at once, according to their label. For instance, selecting only those attributes considered to be *Large* would appear to be a suitable strategy. It may also be beneficial to add different groups of attributes at various stages of the search. To include diversity, cross-group selection is a method that picks representative attributes from each fuzzy label and adds them to the reduct candidate. Again, strategies may be changed during search; for example, it might be worthwhile using the cross-group strategy first, followed by selecting *Large*-grouped attributes later.

One problem encountered in grouping attributes in this way is that in later stages, there are sometimes too many attributes in the required label. Therefore, it is usually best to revert to individual selection when this becomes a problem, making the search more accurate. Alternatively, taking suitable α-cuts will limit the total number selected.

5.1.5 Algorithmic Complexity

The complexity of this algorithm is the same as that of QUICKREDUCT, $O((n^2+n)/2)$. This is because, in the worst case, the algorithm will add individual features (and not groups), following the same route as the standard algorithm. However, when groups of features are selected, certain areas of the QUICKREDUCT search space are avoided, decreasing the time taken to find a reduct.

5.2 FRFS with Ant Colony Optimization

Swarm Intelligence (SI) is the property of a system whereby the collective behaviours of simple agents interacting locally with their environment cause coherent functional global patterns to emerge [21]. SI provides a basis with which it is possible to explore collective (or distributed) problem solving without centralized control or the provision of a global model. For example, ants are capable of finding the shortest route between a food source and their nest without the use of visual information and hence possess no global world model, adapting to changes in the environment. Those SI techniques based on the behaviour of real ant colonies used to solve discrete optimization problems are classed as Ant Colony Optimization (ACO) techniques [21]. These have been successfully applied to a large number of difficult combinatorial problems like quadratic assignment [107] and the travelling salesman [40] problem, to routing in telecommunications networks, scheduling, and others.

ACO is particularly attractive for feature selection as there seems to be no heuristic that can guide search to the optimal minimal subset every time. Additionally, it can be the case that ants discover the best feature combinations as they proceed throughout the search space. This section discusses how ant colony optimization may be applied to the difficult problem of finding optimal feature subsets.

5.2.1 Ant Colony Optimization

The ability of real ants to find shortest routes is mainly due to their depositing of pheromone as they travel; each ant probabilistically prefers to follow a direction rich in this chemical. The pheromone decays over time, resulting in much less pheromone on less popular paths. Given that over time the shortest route will have the highest rate of ant traversal, this path will be reinforced and the others diminished until all ants follow the same, shortest path (the “system” has converged to a single solution). It is also possible that there are many equally short paths - this situation can be handled by ACO as well. In this situation, the rates of ant traversal over the short paths will be roughly the same, resulting in these paths being maintained while others are ignored. Additionally, if a sudden change to the environment occurs (e.g. a large obstacle appears on the shortest path), the system responds to this and will eventually converge to a new solution. Further details on ACO algorithms and their evaluation can be found in [21, 40]. In general, an ACO algorithm can be applied to any combinatorial problem as far as it is possible to define:

- *Appropriate problem representation.* A description of the problem as a graph with a set of nodes and edges between nodes.

- *Heuristic desirability* (η) *of edges*. A suitable heuristic measure of the "goodness" of paths from one node to every other connected node in the graph.

- *Construction of feasible solutions*. A mechanism to efficiently create possible solutions.

- *Pheromone updating rule*. A suitable method of updating the pheromone levels on edges with a corresponding evaporation rule. Typical methods involve selecting the *n* best ants and updating the paths they chose.

- *Probabilistic transition rule*. A mechanism for the determination of the probability of an ant traversing from one node in the graph to the next.

Each ant in the artificial colony maintains a memory of its history - remembering the path it has chosen so far in constructing a solution. This history can be used in the evaluation of the resulting created solution and may also contribute to the decision process at each stage of solution construction.

Two types of information are available to ants during their graph traversal, local and global, controlled by the parameters β and α respectively. Local information is obtained through a problem-specific heuristic measure. The extent to which this influences an ant's decision to traverse an edge is controlled by the parameter β. This will guide ants towards paths that are likely to result in good solutions. Global knowledge is also available to ants through the deposition of artificial pheromone on the graph edges by their predecessors over time. The impact of this knowledge on an ant's traversal decision is determined by the parameter α. Good paths discovered by past ants will have a higher amount of associated pheromone. How much pheromone is deposited, and when, is dependent on the characteristics of the problem. No other local or global knowledge is available to the ants in the standard ACO model. However, the inclusion of such information by extending the ACO framework has been investigated [21] with some success.

5.2.2 Travelling Salesman Problem - An Example

To illustrate how this may be applied to artificial systems, the application of ACO to the travelling salesman problem (TSP) [98] is presented here. The TSP is a combinatorial optimization problem where, given N cities and a distance function d between cities, a minimal tour that goes through every city exactly once is to be found.

The TSP is represented as a graph, with nodes representing cities and edges representing journies between cities. The heuristic desirability of edge (i, j) is the inverse of the distance between those cities $(1/d(i, j))$, where $i \neq j$. Pheromone is increased proportional to the inverse of the tour length. These two measures can be thought of as providing different information about the problem: the heuristic measure provides local information and pheromone global information. These are combined to form the so-called probabilistic transition rule, denoting the probability of an ant in city i choosing to travel to city j at time t:

$$p^k_{ij}(t) = \frac{[\tau_{ij}(t)]^{\alpha}.[\eta_{ij}]^{\beta}}{\sum_{l \in J^k_i} [\tau_{il}(t)]^{\alpha}.[\eta_{il}]^{\beta}} \tag{5.1}$$

where k is the number of ants, J^k_i the set of k's possible next cities, η_{ij} is the heuristic desirability of choosing city j when at city i and $\tau_{ij}(t)$ is the amount of virtual pheromone on edge (i, j). The choice of α and β is determined experimentally. Typically, several experiments are performed, varying each parameter and choosing the values that produce the best results.

With this representation, a number of ants may be placed at nodes (cities) in the graph and traverse edges to form a tour. The pheromone of the edges corresponding to the best tours are reinforced, and a new set of ants make their way through the graph. This process continues until an optimum has been discovered or a certain number of generations of ants has been tried.

5.2.3 Ant-based Feature Selection

By following similar principles, the feature selection task may be reformulated into an ACO-suitable problem. ACO requires a problem to be represented as a graph - here nodes represent features, with the edges between them denoting the choice of the next feature. The search for the optimal feature subset is then an ant traversal through the graph where a minimum number of nodes are visited that satisfies the traversal stopping criterion. Figure 5.3 illustrates this setup - the ant is currently at node a and has a choice of which feature to add next to its path (dotted lines). It chooses feature b next based on the transition rule, then c and then d. Upon arrival at d, the current subset $\{a,b,c,d\}$ is determined to satisfy the traversal stopping criterion (e.g. a suitably high classification accuracy has been achieved with this subset). The ant terminates its traversal and outputs this feature subset as a candidate for data reduction.

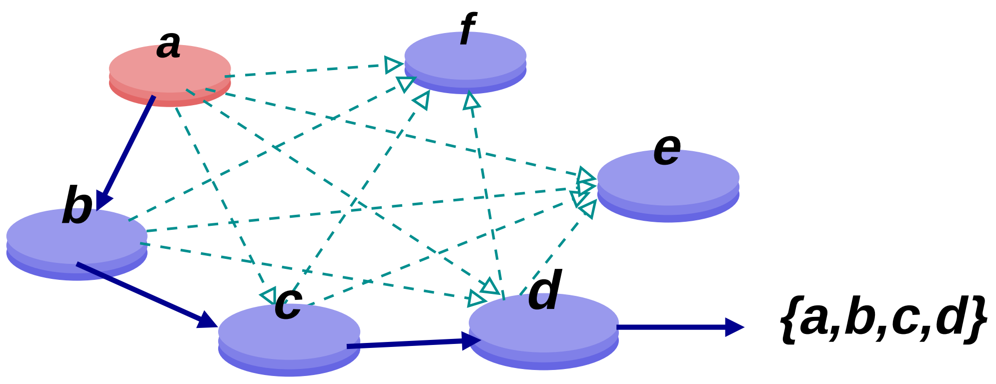


Figure 5.3: ACO problem representation for FS

A suitable heuristic desirability of traversing between features could be any subset evaluation function - for example, an entropy-based measure [135] or the fuzzy-rough set dependency measure [79]. Depending on how optimality is defined for the particular application, the pheromone may be updated accordingly. For instance, subset minimality and "goodness" are two key factors so the pheromone update should be proportional to "goodness" and inversely proportional to size. The transition rule is the same as that given in equation (5.1), however J_i^k in this case is the set of ant k's unvisited features. There is also the possibility of allowing the removal of features here. If feature h has been selected already, an alternative transition rule may be applied to determine the probability of removing this attribute. However, this is an extension of the approach and is not necessary to perform feature selection.

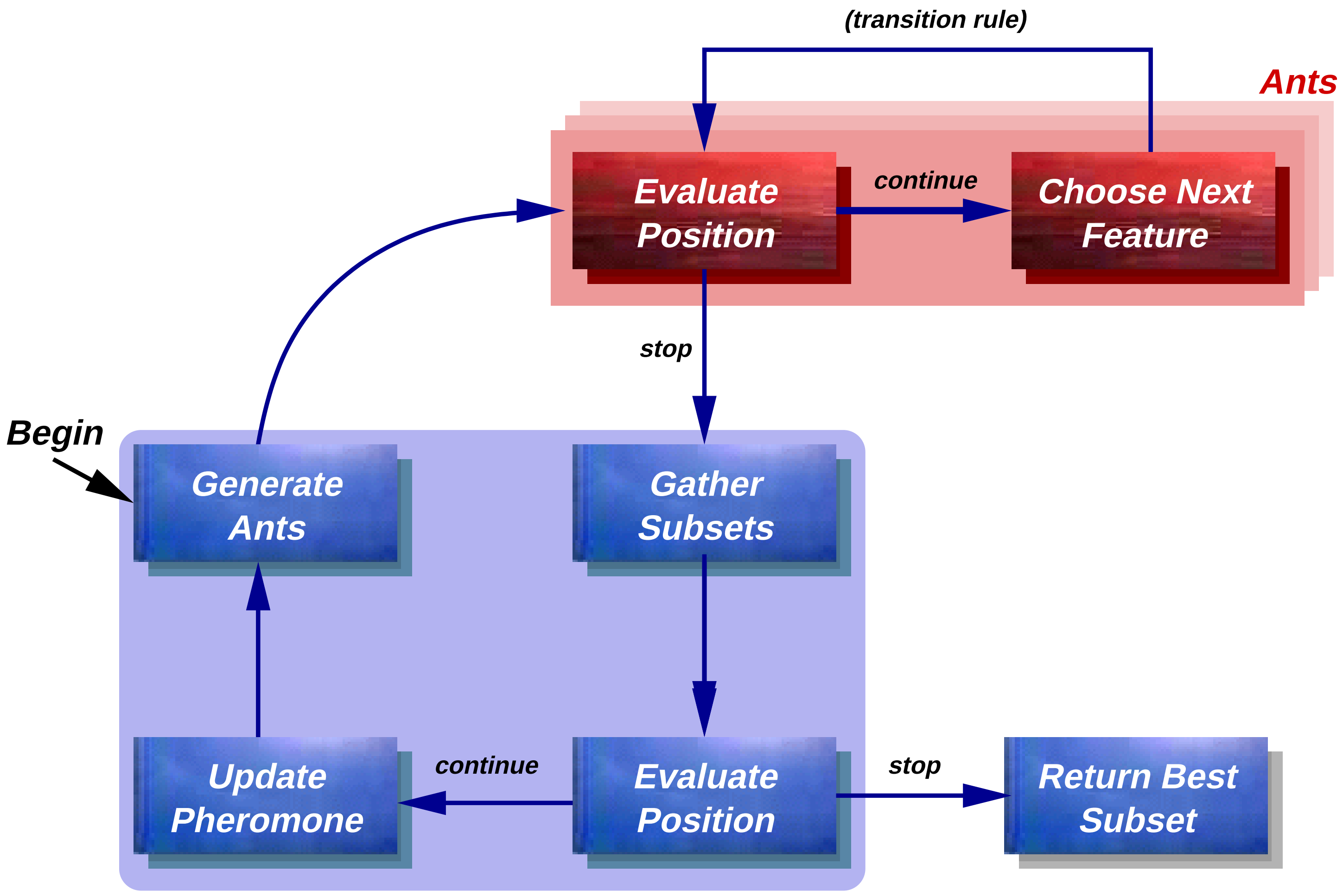


Figure 5.4: Overview of a general ACO-based FS system

5.2.3.1 Selection Process

The overall process of ACO feature selection can be seen in figure 5.4. The process begins by generating a number of ants, k, which are then placed randomly on the graph (i.e. each ant starts with one random feature). Alternatively, the number of ants to place on the graph may be set equal to the number of features within the data; each ant starts path construction at a different feature. From these initial positions, they traverse edges probabilistically until a traversal stopping criterion is satisfied. The resulting subsets are gathered and then evaluated. If an optimal subset has been found or the algorithm has executed a certain number of times, then the process halts and outputs the best feature subset encountered. If neither condition holds, then the pheromone is updated, a new set of ants are created and the process iterates once more.

5.2.3.2 Complexity Analysis

The time complexity of the ant-based approach to feature selection is $O(IAk)$, where I is the number of iterations, A the number of original features, and k the number of ants. This can be seen from figure 5.4. In the worst case, each ant selects all the features. As the heuristic is evaluated after each feature is added to the reduct candidate, this will result in A evaluations per ant. After one iteration in this scenario, Ak evaluations will have been performed. After I iterations, the heuristic will be evaluated IAk times.

This method is attractive for feature selection as there seems to be no heuristic that can guide search to the optimal minimal subset every time. Additionally, it should be the case that ants will discover best feature combinations as they traverse the graph. This information will be reflected in the pheromone matrix, used by other ants to guide their own local search.

5.2.3.3 Pheromone Update

Depending on how optimality is defined for the particular application, the pheromone may be updated accordingly. For instance, subset minimality and “goodness” are two key factors so the pheromone update must be proportional to “goodness” and inversely proportional to size.

To tailor this mechanism to find fuzzy-rough set reducts, it is necessary to use the dependency measure given in equation (4.11) as the stopping criterion. This means that an ant will stop building its feature subset when the dependency of the subset reaches the maximum for the dataset (the value 1 for consistent datasets). The dependency function may also be chosen as the heuristic desirability measure, but this is not necessary. In fact, it may be of more use to employ a non-rough set related heuristic for this purpose to avoid the pitfalls of a QUICKREDUCT style search. By using an alternative measure such as an entropy-based heuristic [135], the method may avoid feature combinations that may mislead the fuzzy-rough set-based heuristic. Again, the time complexity of this fuzzy-rough ant-based method will be the same as that mentioned earlier, $O(IAk)$.

The pheromone on each edge is updated according to the following formula:

$$\tau_{ij}(t+1) = (1-\rho).\tau_{ij}(t) + \Delta\tau_{ij}(t) \tag{5.2}$$

where

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{n} (\gamma'(S^k)/|S^k|) \tag{5.3}$$

This is the case if the edge (i, j) has been traversed; $\Delta\tau_{ij}(t)$ is 0 otherwise. The value ρ is a decay constant used to simulate the evaporation of the pheromone, S^k is the feature subset found by ant k. The pheromone is updated according to both the fuzzy-rough measure of the “goodness” of the ant’s feature subset (γ') and the size of the subset itself. By this definition, all ants update the pheromone. Alternative strategies may be used for this, such as allowing only the ants with the best feature subsets to proportionally increase the pheromone.

5.2.3.4 Approach Comparison

Upon inspection, a number of similarities and differences can be observed between the approaches described previously. In the genetic and ant-based FS, a population of potential solutions are considered on each algorithmic cycle. In the genetic case, this population is evaluated and then used to produce the next population. However in the ant-based case, each population is erased after generation; the only lasting effect the population has is in the updated pheromone levels which are used in the next generation cycle. Simulated annealing-based FS (discussed in section 2.2.4) considers an individual solution candidate only. New candidates are produced by mutating the current solution, in a similar manner to genetic FS. The extent of mutation decreases with time, allowing convergence to a single, hopefully optimal, solution. All of these approaches employ a certain degree of randomness in their search for optimal subsets.

5.3 Summary

This chapter has presented a new direction in feature selection, namely feature grouping. It was shown how fuzzifying particular evaluation functions, the fuzzy-rough or crisp rough set dependency degree, can lead to group and individual selection based on linguistic labels - more closely resembling human reasoning. In fact, this can be applied to most FS algorithms that use an evaluation function that returns values in $[0,1]$. Choosing grouped features instead of individuals also decreases the time taken to reach potentially optimal subsets.

Conventional hill-climbing approaches to feature selection often fail to find minimal data reductions. Some guiding heuristics are better than others for this, but as no perfect heuristic exists there can be no guarantee of optimality. When minimal data reductions are required, other search mechanisms must be employed. Although these methods also cannot ensure optimality, they provide a means by which the best feature subsets might be found. This chapter has presented a new method for feature selection based on ant colony optimization for this purpose. Unlike semantics-destroying approaches, clearly this approach maintains the underlying semantics of the feature set, thereby ensuring that the resulting models are interpretable and the inference explainable. This is evident from the application of FRFS to the problem domains. In particular, for complex systems monitoring (chapter 7) where fuzzy rule induction is performed, the resulting ruleset is highly comprehensible.

Chapter 6

Application to Web Content Categorization

Due to the explosive growth of electronically stored information, automatic methods must be developed to aid users in maintaining and using this abundance of information effectively. Sorting through even a fraction of the available data by hand can be very difficult. In particular, the sheer volume of redundancy present must be dealt with, leaving only the information-rich data to be processed. Information filtering and retrieval systems are therefore acquiring increasing prominence as automated aids in quickly sifting through web-based information. This chapter focuses on the application of FRFS to this task of automatic web data classification. Two tasks are considered: bookmark categorization and web page categorization.

6.1 Text Categorisation

Text categorisation (TC) has often been defined as the content-based assignment of one or more predefined categories to text. As stated in [117], it has become important from two points of view. Firstly, it is important from the Information Retrieval (IR) viewpoint because of the rapid growth of textual information sources, which requires a greater amount of information processing. Text categorisation can be used as a means to efficiently classify this textual data, or to provide support to further IR processes by performing such tasks as document filtering or information extraction. Secondly, it is important from the Machine Learning (ML) viewpoint as text categorisation provides ML with an application field. The approach that ML takes in automatic categorisation is to generate a means of classification by the use of induction over examples that have been categorised previously (a form of supervised learning).

The categorisation of textual documents involves two main phases: training and classification. The training phase involves examining sample documents and retrieving those keywords deemed important. These sets of keywords are large, rendering most text classifiers intractable, so a feature selection step is performed. Induction is carried out, and a means of classifying future data is the output. The classification phase uses the classification means from the training process to classify new documents. Several methods exist for this purpose. An outline of these is given below; details can be found in [77, 183].

6.1.1 Rule-based

For rule-based approaches to classification, a set of rules and an appropriate classifier are required. Each individual rule has a set of preconditions and an associated decision. If a document matches the preconditions, then it is classified according to the decision value.

A simple form of this is the Boolean Exact Model [145] which employs exact boolean existential rules. The problem with this is that it is inflexible - only documents for which the rule returns true match the rule. The Boolean Inexact Model (BIM) [145, 146] bypasses this problem by providing a scoring mechanism so that the rule with the highest score classifies the document. If a term exists in a document and is also present in the corresponding rule, then the score for that rule is increased. If there is more than one rule that fires then all rules must agree on the classification. If there is a conflict, then the classification is undecidable.

The main advantage of BIM is that it is fast (the computations involved are simple). It can also be quite accurate. A drawback is that words can have many meanings - something that the BIM cannot differentiate.

Fuzzy classifiers are another rule-based technique for classification. These follow the usual approach for the construction of fuzzy rule-based systems [128]. All precondition memberships are evaluated, and the necessary logical conjunctions integrated using the conventional minimum operator to derive classifications of new data.

6.1.2 Vector-based

The Vector Space Model (VSM) [183, 144] considers document representatives as binary vectors embedded in an *n*-dimensional Euclidean space (*n* is the total number of keywords). As there tends to be a large number of keywords involved, the dimensionality is also very high.

Each document and query is represented as a point in the space. To obtain the document vector, the keywords contained in the document are obtained and ranked according to their weight in the document. They are then converted to a vector. Any missing keywords (according to the universal keyword set) are marked as absent.

The procedure itself can be divided into three stages. The first stage is document indexing, where content bearing terms are extracted from the document text. The second stage is the weighting of the indexed terms to enhance retrieval of documents relevant to the user. The last stage ranks the document with respect to the query according to the similarity measure. The similarity measure used here is the cosine coefficient, which measures the angle between the rule vector **x** and the query vector **q**, and is defined as:

$$Sim(\mathbf{x},\mathbf{q}) = \frac{\sum_{i=1}^{|\mathbb{C}|} \mathbf{x}_i \cdot \mathbf{q}_i}{\sqrt{\sum_{i=1}^{|\mathbb{C}|} (\mathbf{x}_i)^2 \cdot \sum_{i=1}^{|\mathbb{C}|} (\mathbf{q}_i)^2}} \tag{6.1}$$

There are a number of disadvantages with the VSM. There is the lack of justification for some of the vector operations (for example, the choice of similarity function and the choice of term weights). It is barely a retrieval model as it does not explicitly model relevance. There is also the assumption that a query and a document can be treated the same. However, the simplicity of the model is attractive - probably why it is the most popular retrieval model today.

6.1.3 Latent Semantic Indexing

Latent Semantic Indexing is a variant of the vector retrieval model outlined above which takes into account the dependencies between terms [44, 37]. Unlike other models, LSI treats words as if they are not independent of each other; it attempts to automatically derive and model inter-relationships between them.

The term-document matrix can be considered to be a "bag of documents" and is split into a set of *k* orthogonal factors. Similarity is computed in the following way:

1. Choose *k* (not greater than the number of terms or documents),
2. Add the weighted vectors for each term - multiply each vector by term weight - sum each element separately,
3. Repeat for query or second document, and
4. Compute inner product - multiply corresponding elements and add.

An advantage that LSI has is that it reduces the original number of dimensions. Vectors that are similar are assigned to similar terms. This composite term is then mapped to a single dimension. However, as with most retrieval models, words with similar meanings confound LSI. Also, the computations required are expensive but a lot of this is carried out in advance.

6.1.4 Probabilistic

Another classic retrieval and classification method is the probabilistic retrieval technique [55], where the probability that a specific document will be judged relevant to a specific query, is based on the assumption that the terms are distributed differently in relevant and non relevant documents. The probability formula is usually derived from Bayes' theorem. Given a particular document *x* and a set of categories, the probability of *x* belonging to each category in the category set is calculated. The document is classified into the category that produced the highest probability.

6.1.5 Term Reduction

In text categorisation, the high dimensionality of the term space can be problematic, so some form of DR is employed. This can also be beneficial as it tends to reduce *overfitting*, where a classifier is tuned also to the contingent, rather than just the necessary characteristics of the training data [151]. It is quite often the case that the training data undergoes *several* stages of feature reduction, the final one being feature subset selection. The following measures and techniques are the main DR pre-processors to subset selection:

- *Stop Word Removal*. This is a simple technique for removing very information-poor terms. Stop words are connectives such as articles ('the','a','an', etc.) and contribute very little (if anything) to the classification of documents. Care must be taken when adopting this approach as it is feasible that some information-rich words might be mistakenly viewed as stop words.

- *Document Frequency*. Again, this is a simple and effective reduction strategy [3]. It has been shown that the most informative terms are those with low to medium document frequency. By removing those attributes that occur the most (and to some extent those that occur rarely), the dimensionality of the document is reduced with no or little loss in information. To make this effective stop words should be removed beforehand, otherwise only topic-neutral words may remain after reduction.

- *Word Stemming*. Word suffixes are removed, leaving only the root of the word. This is an attempt to reduce words with similar meanings to the same root, for example $retailing$ and $retailer$ both contain the same root, namely $retail$. This is not guaranteed to work, however, as many words with different meanings share a common root.

- *Other Functions*. There has been much research into using sophisticated information-theoretic term selection functions, such as chi-square [160] and correlation coefficient [118]. These functions have been shown to produce better results than document frequency.

Most text classification systems use one or more of the above DR techniques. For example, in the studies presented here, document frequency and stop word removal were implemented in performing a degree of initial data reduction. However, the use of this alone is insufficient to provide the extent of reduction required for efficient keyword search. Further reduction must take place by the use of feature selectors in order to remove the many remaining information-poor features.

6.2 System Overview

The World Wide Web (WWW) is an information resource, whose full potential may not be realised unless its content is adequately organised and described. This not only applies to the vast network of web pages, but also to users' personal repositories of web page bookmarks. However, due to the immense size and dynamicity of the web, manual categorization is not a practical solution to this problem. There is a clear need for automated classification of web content.

To demonstrate the applicability of the described fuzzy-rough methods, two relevant domains of interest regarding the web were selected, namely bookmark classification and website classification. However, these domains are quite distinct, possessing features and problems that must be independently addressed. This section will initially present those features that are common to both before focusing on domain-dependent issues.

Both applications employ a similar general system architecture in order to reduce dimensionality and perform categorization. A key issue in the design of the system was that of modularity; it should be modelled in such a way as to enable the straightforward replacement of existing techniques with new methods. The current implementations allow this flexibility by dividing the overall process into several independent sub-modules (see figure 6.1):

- *Splitting of Training and Testing Datasets*. Datasets were generated from large textual corpora and separated randomly into training and testing sets. Each dataset is a collection of documents, either bookmarks or web pages depending on the application.

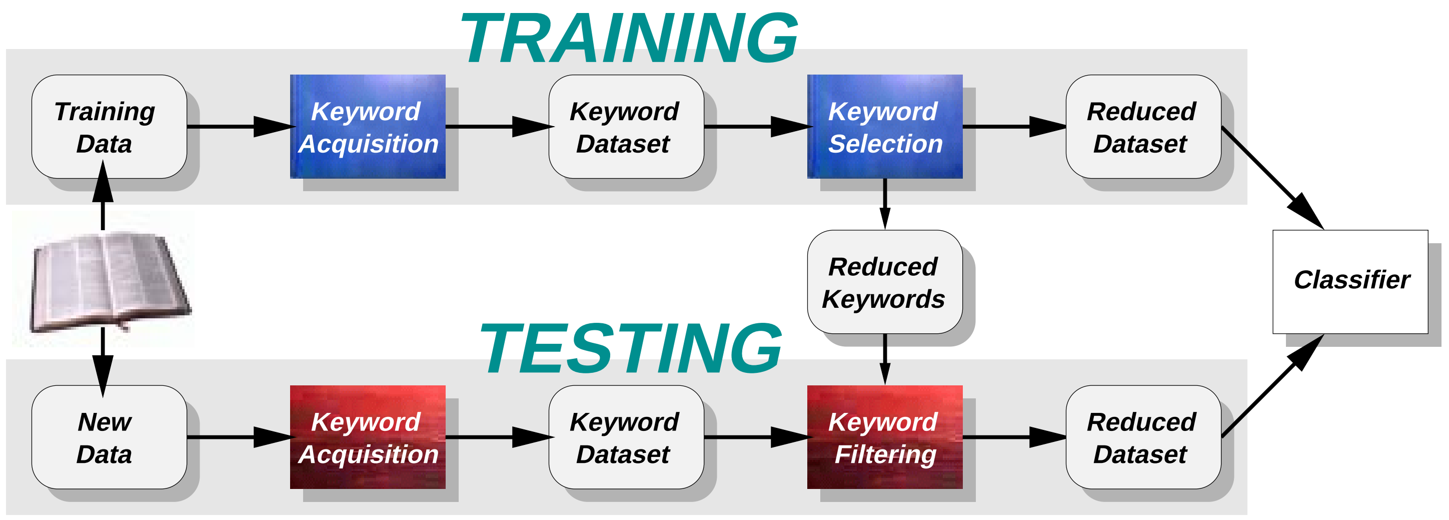


Figure 6.1: Modular decomposition of the classification system

- *Keyword Acquisition*. Given the output from the previous module, keywords/terms are extracted and weighted according to their perceived importance in the document, resulting in a new dataset of weight-term pairs. Note that in this work, no sophisticated keyword acquisition techniques are used as the current focus of attention is on the evaluation of attribute reduction. However, the use of more effective keyword acquisition techniques recently built in the area of information retrieval would help improve the system's overall classification performance further.

- *Keyword Selection*. As the newly generated datasets can be too large, mainly due to keyword redundancy, to perform classification at this stage, a dimensionality reduction step is carried out using FRFS. If this step is preceded by a discretization phase, RSAR may also be applied to the data.

- *Keyword Filtering*. Used only in testing, this simple module filters the keywords obtained during acquisition, using the reduct generated in the keyword selection module.

- *Classification*. This final module uses the reduced dataset to perform the actual categorization of the test data. More efficient and effective classifiers can be employed for this, but for simplicity only conventional classifiers are adopted here to show the power of attribute reduction. Better classifiers are expected to produce more accurate results, though not necessarily enhance the comparisons between classifiers that use reduced or unreduced datasets.

6.3 Bookmark Classification

As the use of the World Wide Web becomes more prevalent and the size of personal repositories grows, adequately organising and managing bookmarks becomes crucial.

Several years ago, in recognition of this problem, web browsers included support for tree-like folder structures for organising bookmarks. These enable the user to browse through their repository to find the necessary information. However manual Uniform Resource Locater (URL) classification and organisation can be difficult and tedious when there are more than a few dozen bookmarks to classify - something that goes against the whole grain of the bookmarking concept.

Many usability studies (e.g. [96]) indicate that a deep hierarchy results in less efficient information retrieval as many traversal steps are required, so users are more likely to make mistakes. Users also do not have the time and patience to arrange their collection into a well-ordered hierarchy. The present work can help extract information from this relatively information-poor domain in order to classify bookmarks automatically.

6.3.1 Existing Systems

As this area of research is relatively new, there has been very little research carried out. Most bookmark utilities provide no means of automatic classification; the systems outlined here are the only ones known with such functionality.

6.3.1.1 Bookmark Organiser

Bookmark Organiser (BO) [106] is an application that attempts to provide automatic assistance in organising bookmark repositories by their conceptual categories. It can operate in two modes:

1. **Fully Automatic**: If the user does not know which category the bookmark belongs to, they can request BO to insert it in the relevant folder by applying an automatic clustering technique to the document to which it refers.

2. **Semi Automatic**: In this case, the user specifies the node into which to insert the new bookmark. They can now request the bookmarks to be re-organised automatically.

Bookmarks are organised by applying the Hierarchical Agglomerative Clustering (HAC) technique to the text contained within the document the bookmark refers to. This is outlined below:

- **Start** with a set of singleton clusters, each contains one object
- **Repeat** the following steps iteratively **until** there is only one cluster
  - **Identify** the two clusters that are the most similar
  - **Merge** them together into a single cluster

This system constructs folders based on the text in the document, which should give a clear indication as to the category to which it belongs, and also enables automatic and semi-automatic classification (a useful feature). However, the automatically-generated folder titles are quite unclear and often meaningless to the user.

6.3.1.2 PowerBookmarks

This semi-structured database application (as reported in [99]) aims to provide personalised organisation and management of bookmarks in a multi-user environment.

POWERBOOKMARKS automatically classifies documents by content; it parses metadata from bookmarked URLs to index and classify them. The metadata in this case is automatically generated from the document that the bookmark refers to, essentially a summary of the information the document contains. It avoids the verbose labelling problem encountered in BO by using existing classifiers with manually selected labels (for example “Sports/Football”). Significant keywords are extracted from documents according to the word frequency analysis. Queries are then issued to an external classifier, the results of which are processed to ensure consistency with existing classifications.

6.3.2 The Implemented System

A large set of bookmarks was used as the training dataset. This database was generated by collating various online bookmark lists into one uniform collection. Each bookmark is pre-classified into a relevant category (for example, “Sports” or “Computing/Java”). An additional testing dataset of “unseen” bookmarks was also compiled from online resources. To clarify the operation of the system, an example is included. The following bookmark is one of many contained in the database of bookmarks under the category *Programming*/*Java*:

```
<A HREF="http://java.sun.com/Performance/">
Ways to Increase Java Performance</A>
```

The two previously outlined methods require information in addition to that stored in the bookmark database. Both require further processing of the documents referred to by the bookmarks. The sorting system developed here relies solely on the information contained within the bookmark database itself - no other information is used to determine the feature reductions or classifications. The system is modular in structure, allowing various sub-components to be replaced with alternative implementations if the need arises. The main modules are *Keyword Acquisition*, *Feature Selection* and *Classification*.

6.3.2.1 Keyword Acquisition

To retain as much information as possible, all fields residing within the bookmark database are considered. For every bookmark, the URL is divided into its slash-separated parts, with each part regarded as a keyword. In a similar fashion, the bookmark title is split into terms with stop words removed.

In order to compare the similarity of bookmarks, a suitable representation must be chosen. Each bookmark is considered to be a vector where the *i*th element is the weight of term *i* according to some weighting method (a metric). The size of the vector is equal to the total number of keywords determined from the training documents.

This module produces weight-term pairs given a dataset. Each encountered word in a URL or title field is assigned a weight according to the metric used. Several metrics were implemented for this purpose:

- *Boolean Existential Metric*. All keywords that exist in the document are given a weight of 1, those that are absent are assigned 0 [145].

- *Frequency Count Metric*. The normalized frequency of the keywords in the document is used as the weight [146].

- *TF-IDF*. The Term Frequency-Inverse Document Frequency Metric [147] assigns higher weights to those keywords that occur frequently in the current document but not in most others. It is calculated using the formula: $w(t,i) = F_i(t) \times log\frac{N}{N_t}$ where $F_i(t)$ is the frequency of term t in document i, N is the number of documents in the collection, and N_t is the total number of documents that contain t.

For the example bookmark, the keywords $\{$*java,sun,com,performance*$\}$ are obtained from the URL, and the keywords $\{$*ways,increase,java,performance*$\}$ from the title field. Using the simple boolean existential metric, the vector elements relating to these keywords will each contain the value 1, the remainder 0.

The resulting sets of weight-term pairs, no matter which keyword acquisition metric is adopted, are large in size and need to be greatly reduced to be of any practical use for classification. Hence, the next step: dimensionality reduction.

6.3.2.2 Dimensionality Reduction

Given the weight-term sets, this module aims to significantly reduce their size whilst retaining their information content and preserving the semantics of those remaining keywords. FRFS is applied here to achieve this goal. Once a reduct has been calculated, the dataset can then be reduced by deleting those attributes that are absent from the reduct. The reduced dataset is now in a form that can be used by the classification module.

Returning to the example, it may be decided by this module that the term “com” provides little or no useful information. The column relating to this term is removed from the main dataset. This process is repeated for all keywords deemed by FRFS to be information-poor.

6.3.2.3 Classification

This module attempts to classify a given bookmark or bookmarks using the reduced keyword datasets obtained by the feature selection stage. Each bookmark has been transformed into a weight-term vector by the keyword acquisition process. For investigation purposes, two different inference techniques were implemented to perform classification: the Boolean Inexact Model and the Vector Space Model.

To illustrate the operation of the classification methods, consider the training and testing vectors presented in figure 6.2. The training data consists of two objects, one classified to the "Sport" category, the other classified to "News". Values in the vector represent the frequency of occurrence of terms in the training item.

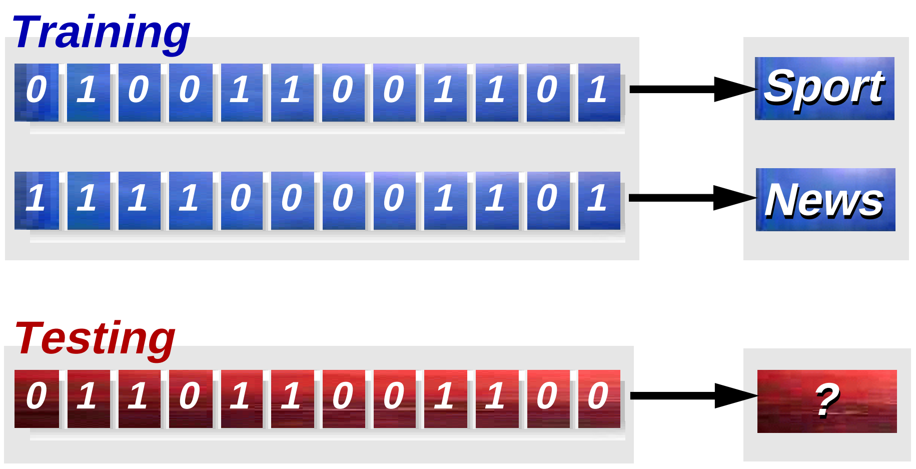


Figure 6.2: Training and testing vectors

For BIM classification, the training data (viewed as rules) is used to classify the new object by comparing term values in the vectors and incrementing a score if they are the same. Classifying using the first object results in a score of 10/12, as 10 of the term values match this training object. With the second object, a score of 7/12 is produced. Hence, for this example, the test object is classified to the "Sport" category.

In VSM, the similarity measure defined in equation (6.1) is used to determine the closeness of the test object to the training examples. For the first training item, the computed similarity is $(5/\sqrt{6\cdot 6}) = 0.83$. For the second item, the similarity is calculated to be $(4/\sqrt{7\cdot 6}) = 0.62$. Again, the test object is determined to belong to the "Sport" category.

6.3.3 Results

The experiments presented here attempt to test whether FRFS is a useful tool for reducing data whilst retaining the information content. For this domain, the FRFS approach produces exactly the same results as the crisp RSAR approach as all equivalence classes are crisp. Random-reduct (RR) generation (i.e. generating reducts randomly) was used to compare the results. This method deletes random attributes from the dataset, but is constrained to leave the same number of attributes present as the FRFS method. The results of these approaches can be found in table 6.2.

Dataset	Attributes (URL)	Attributes (Title)
Unreduced	1397	1283
FRFS-reduced	514	424

Table 6.1: Comparison of the original number of features (Unreduced) with those resulting from FRFS reduction

It can be seen from table 6.1 that using the present work, the amount of attributes was reduced to around 35% of the original. This demonstrates the large amount of redundancy present in the original datasets. In light of the fact that bookmarks contain very little useful information, the results are a little better than anticipated.

Dataset	VSM	BIM
Unreduced	55.6%	49.7%
FRFS-reduced	49.1%	47.3%
RR-reduced	37.3%	34.9%

Table 6.2: Comparison of reduction strategies with the unreduced dataset

A comparison of the performance of the dimensionality reduction techniques is presented in table 6.2. The table shows that the overall accuracy is poor (obviously, the random reduction gives worst results). The main point to make here is that the ability of the system to classify new data depends entirely on the quality (and to a certain extent the quantity) of the training data. It cannot, in general, be expected that the

FRFS-reduced experiments should perform much better than the original unreduced dataset, which itself only allows a rather low classification rate.

In light of the fact that bookmarks contain very little useful information, the results are unsurprising and perhaps a little better than anticipated. As stated earlier, the goal is to investigate how useful fuzzy-rough feature selection is in reducing the training dataset.

It is interesting to compare how the use of reduced datasets may affect the classification accuracy as compared to that of the unreduced dataset. Importantly, the performance of the reduced dataset is almost as good as the original. Although a very small amount of important information may have been lost in the attribute reduction, this information loss is not significant enough to reduce classification accuracy significantly, while the reduction of dimensionality is substantial.

Results clearly show that FRFS (and RSAR) can be used to significantly reduce the dimensionality of the training dataset without much loss in information content. The measured drop in classification accuracy was 2.4% (using BIM) and 6.5% (using VSM) for the training dataset, which is within acceptable bounds.

6.4 Website Classification

There are an estimated 1 billion web pages available on the WWW with around 1.5 million web pages being added every day [68]. The task to find a particular web page, which satisfies a user's requirements by traversing hyper-links, is very difficult. To aid this process, many web directories have been developed - some rely on manual categorization whilst others make decisions automatically. However, as web page content is vast and dynamic, manual categorization is becoming increasingly impractical. Automatic web site categorization is therefore required to deal with these problems.

6.4.1 Existing Systems

Very recently, research has been carried out into this complex text classification area. In fact, it has given rise to specific techniques for problems such as indexing, term selection etc as web pages can be considered to be a special kind of document. Web pages contain text much like any other document, but the fact that they contain pointers to *other* documents makes them an interesting area for research.

An indexing technique specific to these documents has been proposed by [5]. A web page tends to have many other pages pointing towards it. The authors reason that these documents can be combined forming an artificial document which can be considered to be a compilation of “short reviews” of the Web page. It is this compilation that is used in classification, not the original document.

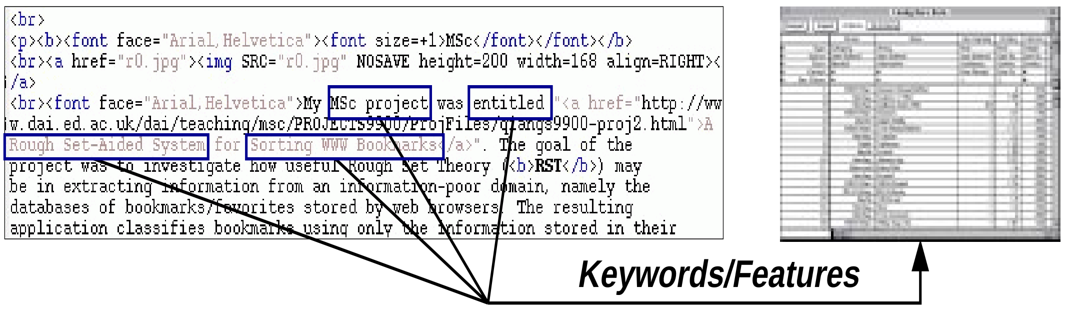
Another indexing technique has been put forward [56], where a document representation is obtained by the combined indexing of both the original document and its children (the documents that it points to). This is of interest for Web site classification, as often the children of a document contain relevant information for the site as a whole.

The topic of indexing is not the only one to have been investigated - different methods of classifier induction have been proposed. In [26], a web page categorisation approach was developed based on the hypothesis that for each document-classification pair (d,c), two values can be associated; the authority $a(d,c)$ and the hub value $h(d,c)$. The authority value is a measure of the “authoritativeness” of d on c in terms of the number of c-related pages pointing to it. The hub value measures the “informativeness” of d on c in terms of how many c-related documents it points *to*. Therefore, the purpose of this system is to identify the n most authoritative and the n most informative documents that are associated with the document, given a classification c. By issuing a query c to ALTAVISTA, an initial set of relevant documents is obtained, giving a starting point for the induction algorithm.

A method for the induction of classifiers for hierarchical category sets was developed in [142], using neural networks to achieve this. The system is composed of two networks: *gating* and *expert*. A gating network for a category c_i is a neural network that decides if a document d_j might be a plausible candidate for categorisation under any child categories of c_i. Document d_j is propagated to all child nodes of c_i if the decision is positive, otherwise no propagation takes place. An expert network for c_i simply decides whether document d_j should be classified to c_j. A classification tree consisting of gating or expert networks is used for document classification; leaf nodes are expert networks, internal nodes may be gating or expert. This allows a document to be classified under internal nodes and leaf nodes simultaneously.

6.4.2 The Implemented System

There is usually much more information contained in a web document than a bookmark. Additionally, information can be structured within a web page that may indicate a relatively higher or lower importance of the contained text. For example, terms appearing within a <TITLE> tag would be expected to be more informative than the majority of those appearing within the document body at large. Because of this, keywords are weighted not only according to their statistical occurrence but also to their location within the document itself. These weights are almost always real-valued, which can be a problem for most feature selectors unless data discretisation takes place (a source of information loss). This motivates the application of FRFS to this domain.



```
<br>
<p><b><font face="Arial,Helvetica"><font size=+1>MSc</font></font></b>
<br><a href="r0.jpg"><img SRC="r0.jpg" NOSAVE height=200 width=168 align=RIGHT><
/a>
<br><font face="Arial,Helvetica">My MSc project was entitled "<a href="http://ww
w.dai.ed.ac.uk/dai/teaching/msc/PROJECTS9900/ProjFiles/qiangs9900-proj2.html">A
Rough Set-Aided System for Sorting WWW Bookmarks</a>". The goal of the
project was to investigate how useful Rough Set Theory (<b>RST</b>) may
be in extracting information from an information-poor domain, namely the
databases of bookmarks/favorites stored by web browsers. The resulting
application classifies bookmarks using only the information stored in their
```

Figure 6.3: Keyword extraction

The training and testing datasets were generated using Yahoo [192]. Five classification categories were used, namely Art & Humanity, Entertainment, Computers & Internet, Health, Business & Economy. A total of 280 web sites were collected from Yahoo categories and classified into these categories, 56 sites per category resulting in a balanced dataset. From this collection of data, the keywords, weights and corresponding classifications were collated into a single dataset (see figure 6.3).

6.4.3 Results

For this set of experiments, FRFS is compared with the crisp RSAR approach. As the unreduced dataset exhibits high dimensionality (2557 attributes), it is too large to evaluate (hence the need for keyword selection). Using crisp RSAR the original attribute set was reduced to 29 (1.13% of the full set of attributes). However, using FRFS the number of selected attributes was only 23 (0.90% of the full attribute set). It is interesting to note that the FRFS reduct and crisp RSAR reduct share four attributes in common. With such a large reduction in attributes, it must be shown that classification accuracy does not suffer in the FRFS-reduced system.

In addition to the classification accuracy, the precision of the system is also presented. Precision is defined here to be the ratio of the number of correctly classified documents to the total number of correctly and incorrectly classified documents (expressed as a percentage). This differs from the classification accuracy, in that the accuracy also considers documents that have not been able to be classified by the system.

To see the effect of dimensionality reduction, the system was tested on the original training data first and the results are summarised in table 6.3. The results are averaged over all the classification categories. Clearly, the fuzzy method exhibits better precision and accuracy rates. This performance was achieved using fewer attributes than the crisp approach.

Method	Attributes	Average Precision	Average Accuracy
Original	2557	-	-
Crisp	29	73.7%	76.2%
Fuzzy	23	78.1%	83.3%

Table 6.3: Performance: training data (using VSM)

Table 6.4 contains the results for experimentation on 140 previously unseen web sites. For the crisp case, the average precision and accuracy are both rather low. With FRFS there is a significant improvement in both the precision and classification accuracy. Again, this more accurate performance is achieved while using fewer attributes.

It must be pointed out here that although the testing accuracy is rather low, this is

Method	% Original Attributes	Classifier	Average Precision	Average Accuracy
Crisp	1.13	BIM	23.3%	11.7%
		VSM	45.2%	50.3%
Fuzzy	0.90	BIM	16.7%	20.0%
		VSM	74.9%	64.1%

Table 6.4: Performance: unseen data

largely to do with the poor performance of the simple classifiers used. The fact that VSM-based results are much better than those using BIM-based classifiers shows that when a more accurate classification system is employed, the accuracy can be considerably improved with the involvement of the same attributes. Nevertheless, the purpose of the present experimental studies is to compare the performance of the two attribute reduction techniques, based on the common use of any given classifier. Thus, only the relative accuracies are important. Also, the FRFS approach requires a reasonable fuzzification of the input data, whilst the fuzzy sets are herein generated by simple statistical analysis of the dataset with no attempt made at optimizing these sets. A fine-tuned fuzzification will certainly improve the performance of FRFS-based systems [109].

Finally, it is worth noting that the classifications were checked automatically. Many websites can be classified to more than one category, however only the designated category is considered to be correct here.

6.5 Summary

This chapter has presented a fuzzy-rough method to aid the classification of web content, with promising results. In many text categorisation problems, feature weights within datasets are real-valued, posing a problem for many feature selection methods. FRFS can handle this type of data without the need for a discretising step beforehand. In particular, whilst retaining less attributes than the conventional crisp rough set-based technique, the work resulted in an overall higher classification accuracy.

Chapter 7

Application to Complex Systems Monitoring

The ever-increasing demand for dependable, trustworthy intelligent diagnostic and monitoring systems, as well as knowledge-based systems in general, has focused much of the attention of researchers on the knowledge-acquisition bottleneck. The task of gathering information and extracting general knowledge from it is known to be the most difficult part of creating a knowledge-based system. Complex application problems, such as reliable monitoring and diagnosis of industrial plants, are likely to present large numbers of features, many of which will be redundant for the task at hand [130, 152]. Additionally, inaccurate and/or uncertain values cannot be ruled out. Such applications typically require convincing explanations about the inference performed, therefore a method to allow automated generation of knowledge models of clear semantics is highly desirable.

The most common approach to developing expressive and human readable representations of knowledge is the use of if-then production rules [93]. Yet, real-life problem domains usually lack generic and systematic expert rules for mapping feature patterns onto their underlying classes. The present work aims to induce low-dimensionality rulesets from historical descriptions of domain features which are often of high dimensionality. In particular, a recent fuzzy rule induction algorithm (RIA), as first reported in [28], is taken to act as the starting point for this. The premise attributes of the induced rules are represented by fuzzy variables, facilitating the modelling of the inherent uncertainty of the knowledge domain. It should be noted, however, that the flexibility of the system discussed here allows the incorporation of almost any rule induction algorithm that uses descriptive set representation of features. The choice of the current RIA is largely due to its recency and the simplicity in implementation. Provided with sets of continuous feature values, the RIA is able to induce classification rules to partition the feature patterns into underlying categories.

There exists a number of approaches relevant to the rule induction task at hand, both from the point of view of applications and that of computational methods. For example, the FAPACS (*Fuzzy Automatic Pattern Analysis and Classification System*) algorithm documented in [6, 27] is able to discover fuzzy association rules in relational databases. It works by locating pairs of features that satisfy an 'interestingness' measure that is defined in terms of an adjusted difference between the observed and expected values of relations. This algorithm is capable of expressing linguistically both the regularities and the exceptions discovered within the data. Modifications to the Fuzzy ID3 (itself an augmentation of Quinlan's original ID3 [135]) rule induction algorithm have been documented [61] to better support fuzzy learning. In a similar attempt, [69] has proposed modifications to decision trees to combine traditional symbolic decision trees with approximate reasoning, offered by fuzzy representation. This approach redefines the methodology for knowledge inference, resulting in a method best suited to relatively stationary problems.

A common disadvantage of these techniques is their sensitivity to high dimensionality. This may be remedied using conventional work such as Principal Components Analysis (PCA) [38, 50]. As indicated previously, although efficient, PCA irreversibly destroys the underlying semantics of the feature set. Further reasoning about the derivation from transformed principal features is almost always humanly impossible. Most semantics-preserving dimensionality reduction (or feature selection) approaches tend to be domain specific, however, relying on the use of well-known features of the particular application domains.

In order to speed up the RIA and reduce rule complexity, a preprocessing step is required. This is particularly important for tasks where learned rulesets need regular updating to reflect the changes in the description of domain features. This step reduces the dimensionality of potentially very large feature sets while minimising the loss of information needed for rule induction. It has an advantageous side-effect in that it removes redundancy from the historical data. This also helps simplify the design and implementation of the actual pattern classifier itself, by determining what features should be made available to the system. In addition, the reduced input dimensionality increases the processing speed of the classifier, leading to better response times. Most significant, however, is the fact that fuzzy-rough feature selection (FRFS) preserves the semantics of the surviving features after removing any redundant ones. This is essential in satisfying the requirement of user readability of the generated knowledge model, as well as ensuring the understandability of the pattern classification process.

This chapter is structured as follows. The first section describes the operation of the fuzzy RIA adopted here, with a simple example of how FRFS can aid this process. Next, the problem domain is described, with the key factors that motivate the use of feature selection detailed. Additionally, key areas that need to be addressed within the monitoring system are discussed. Experimental analysis, including comparisons with other feature selection methods and RIAs, is presented next. The application of feature grouping and ACO-based fuzzy-rough feature selection to this domain is also investigated.

7.1 Fuzzy Rule Induction and Feature Selection

For self-containedness, a brief overview of the RIA given in [28] is provided here. For simplicity in outlining this induction procedure the dataset given in section 4.4.2 is used. There are three features each with corresponding linguistic terms, e.g. a has terms $A1$, $A2$ and $A3$. The decision feature $Plan$ is also fuzzy, separated into three linguistic decisions X, Y and Z.

The algorithm begins by organising the dataset objects into subgroups according to their highest decision value. Within each subgroup, the fuzzy subsethood [88, 199] is calculated between the decisions of the subgroup and each feature term. Fuzzy subsethood is defined as follows:

$$S(A,B) = \frac{M(A \cap B)}{M(A)} = \frac{\sum_{u \in \mathbb{U}} min(\mu_A(u), \mu_B(u))}{\sum_{u \in \mathbb{U}} \mu_A(u)} \tag{7.1}$$

From this, the subsethood values listed in table 7.1 can be obtained.

Plan	Linguistic term							
	A1	A2	A3	B1	B2	B3	C1	C2
X	1	0.1	0	0.71	0.43	0.14	0.52	0.76
Y	0.33	0.58	0.29	0.42	0.58	0.04	0.13	0.92
Z	0.14	0.64	0.29	0.32	0.61	0.14	0.82	0.25

Table 7.1: Subsethood values between conditional feature terms and the decision terms

For instance, $S(X,A1) = 1$ is obtained by examining the subgroup of objects that belong to the decision X in table 4.3. The objects concerned are 2, 4 and 7:

Object	$A1$	X
2	1.0	0.8
4	0.8	0.6
7	1.0	0.7

Using these values, the necessary components of equation 7.1 can be calculated as follows:

$$\begin{aligned} \mathrm{M}(X) = & \quad 0.8 + 0.6 + 0.7 = 2.1 \\ \mathrm{M}(X \cap A1) = & \quad min(0.8,1) + min(0.6,0.8) + min(0.7,1) \\ = & \quad 0.8 + 0.6 + 0.7 = 2.1 \end{aligned}$$

These subsethood values are an indication of the relatedness of the individual terms of the conditional features (or values of the features) to the decisions. This measure is central to the fuzzy RIA [28]. A suitable level threshold, $\alpha \in [0,1]$, must be chosen beforehand in order to determine whether terms are close enough or not. At most, one term is selected per feature. For example, setting $\alpha = 0.9$ means that the term with the highest fuzzy subsethood value (or its negation) above this threshold will be chosen. Applying this process to the first two decision values X and Y generates the rules:

Rule 1: IF a is $A1$ THEN $Plan$ is X

Rule 2: IF b is NOT $B3$ AND c is $C2$ THEN $Plan$ is Y

A problem is encountered here when there are no suitably representative terms for a decision (as is the case for decision Z). In this situation, a rule is produced that classifies cases to the decision value if the other rules do not produce reasonable classifications, in order to entail full coverage of the learned rules over the entire problem domain. This requires another threshold value, $\beta \in [0,1]$, which determines whether a classification is reasonable or not. For decision Z, the following rule is produced:

Rule 3: IF $MF(Rule1) < \beta$ AND $MF(Rule2) < \beta$ THEN $Plan$ is Z

where MF(Rule i) = MF(condition part of Rule i) and MF means the membership function value. The use of thresholds α and β pose a problem for the RIA, as it is not clear how to acquire such information. Typically, these are estimated from repeated experimentation over the threshold range, but this procedure is not guaranteed to find the optimum values.

Object	Classified			Actual		
	X	Y	Z	X	Y	Z
1	0.3	0.7	0.0	0.1	0.9	0.0
2	1.0	0.3	0.0	0.8	0.2	0.0
3	0.0	0.4	1.0	0.0	0.2	0.8
4	0.8	0.7	0.0	0.6	0.3	0.1
5	0.5	1.0	0.0	0.6	0.8	0.0
6	0.0	1.0	0.0	0.0	0.7	0.3
7	1.0	0.8	0.0	0.7	0.4	0.0
8	0.1	0.3	1.0	0.0	0.0	1.0
9	0.3	0.0	1.0	0.0	0.0	1.0

Table 7.2: Classified plan with all features and the actual plan

The classification results when using these rules on the example dataset can be found in table 7.2. It shows the membership degrees of the cases to each classification for the classified plan and the underlying plan present in the training dataset. Clearly, the resulting classifications are the same when the *min* t-norm is used.

This technique has been shown to produce highly competitive results [28] in terms of both classification accuracy and number of rules generated. However, as is the case for most rule induction algorithms, the resultant rules may be unnecessarily complex due to the presence of redundant or misleading features. Fuzzy-Rough Feature Selection may be used to significantly reduce dataset dimensionality, removing redundant features that would otherwise increase rule complexity and reducing the time for the induction process itself.

Rule 1: IF a is $A1$ THEN $Plan$ is X
Rule 2: IF c is $C2$ THEN $Plan$ is Y
Rule 3: IF $MF(Rule1) < \beta$ AND $MF(Rule2) < \beta$ THEN $Plan$ is Z

Figure 7.1: Generated rules using the reduced dataset

As has been demonstrated previously, the example dataset may be reduced by the removal of feature b with little reduction in classification accuracy (according to FRFS). Using this reduced dataset, the RIA generates the rules given in figure 7.1. From this, it can be seen that rule 2 has been simplified due to the redundancy of feature b. Although the extent of simplification is small in this case, with larger datasets the effect can be expected to be greater.

Object	Classified			Actual		
	X	Y	Z	X	Y	Z
1	0.3	0.7	0.0	0.1	0.9	0.0
2	1.0	0.3	0.0	0.8	0.2	0.0
3	0.0	0.4	1.0	0.0	0.2	0.8
4	0.8	0.8	0.0	0.6	0.3	0.1
5	0.5	1.0	0.0	0.6	0.8	0.0
6	0.0	1.0	0.0	0.0	0.7	0.3
7	1.0	0.3	0.0	0.7	0.4	0.0
8	0.1	0.3	1.0	0.0	0.0	1.0
9	0.3	0.0	1.0	0.0	0.0	1.0

Table 7.3: Classified plan with reduced features and the actual plan

The results using the FRFS-reduced dataset are provided in table 7.3. The differences between the classifications of the reduced and unreduced approaches have been highlighted (objects 4 and 7). In object 4, only the membership degree for Y has changed. This value has increased from 0.7 to 0.8, resulting in an ambiguous classification. Again, for case 7, the membership degree for Y is the only value to have changed; this time it more closely resembles the classification present in the training dataset.

7.2 The Application

In order to evaluate further the utility of the FRFS approach and to illustrate its domain-independence, a challenging test dataset was chosen, namely the Water Treatment Plant Database [20] (in addition to the experimental evaluation carried out in the last chapter).

7.2.1 Problem Case

The dataset itself is a set of historical data charted over 521 days, with 38 different input features measured daily. Each day is classified into one of thirteen categories depending on the operational status of the plant. However, these can be collapsed into just two or three categories (i.e. *Normal* and *Faulty*, or *OK*, *Good* and *Faulty*) for plant monitoring purposes as many classifications reflect similar performance. This is also performed to balance the class distributions present in the data. Because of the efficiency of the actual plant the measurements were taken from, all faults appear for short periods (usually single days) and are dealt with immediately. This does not allow for a lot of training examples of faults, which is a clear drawback if a monitoring system is to be produced. Note that this dataset has been utilised in many previous studies, including that reported in [157] (to illustrate the effectiveness of applying crisp RSAR as a pre-processing step to rule induction, where a different RIA is adopted from here).

The thirty eight conditional features account for the following five aspects of the water treatment plant's operation (see figure 7.2):

1. Input to plant (9 features)

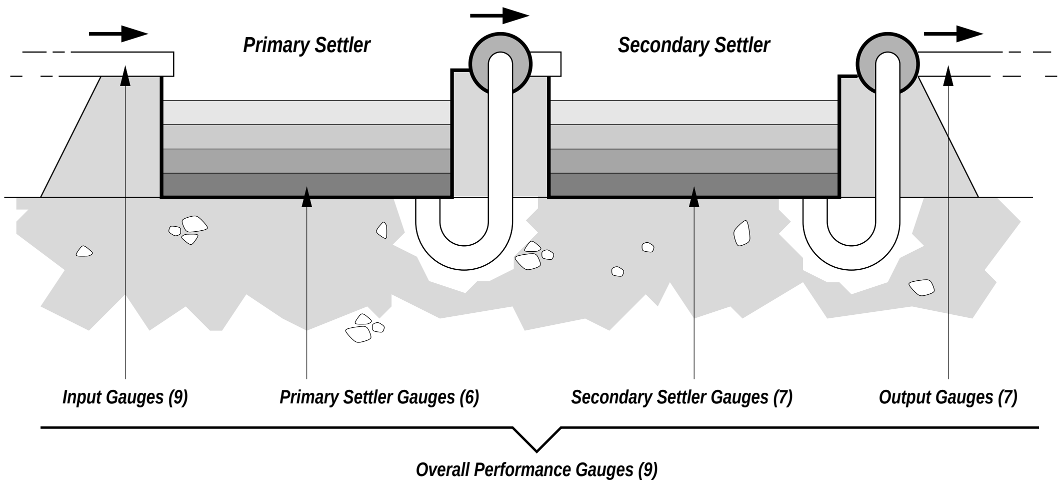

Primary Settler
Secondary Settler
Input Gauges (9)
Primary Settler Gauges (6)
Secondary Settler Gauges (7)
Output Gauges (7)
Overall Performance Gauges (9)

Figure 7.2: Water treatment plant, with number of measurements shown at different points in the system

2. Input to primary settler (6 features)

3. Input to secondary settler (7 features)

4. Output from plant (7 features)

5. Overall plant performance (9 features)

The original dataset was split into 75% training and 25% testing data, maintaining the proportion of classifications present. It is likely that not all of the 38 input features are required to determine the status of the plant, hence the dimensionality reduction step. However, choosing the most informative features is a difficult task as there will be many dependencies between subsets of features. There is also a monetary cost involved in monitoring these inputs, so it is desirable to reduce this number.

7.2.2 Monitoring System

This work follows the original approach for complex systems monitoring developed in [157]. The original monitoring system consisted of several modules as shown in figure 7.3. It is this modular structure that allows the new FRFS technique to replace the existing crisp method.

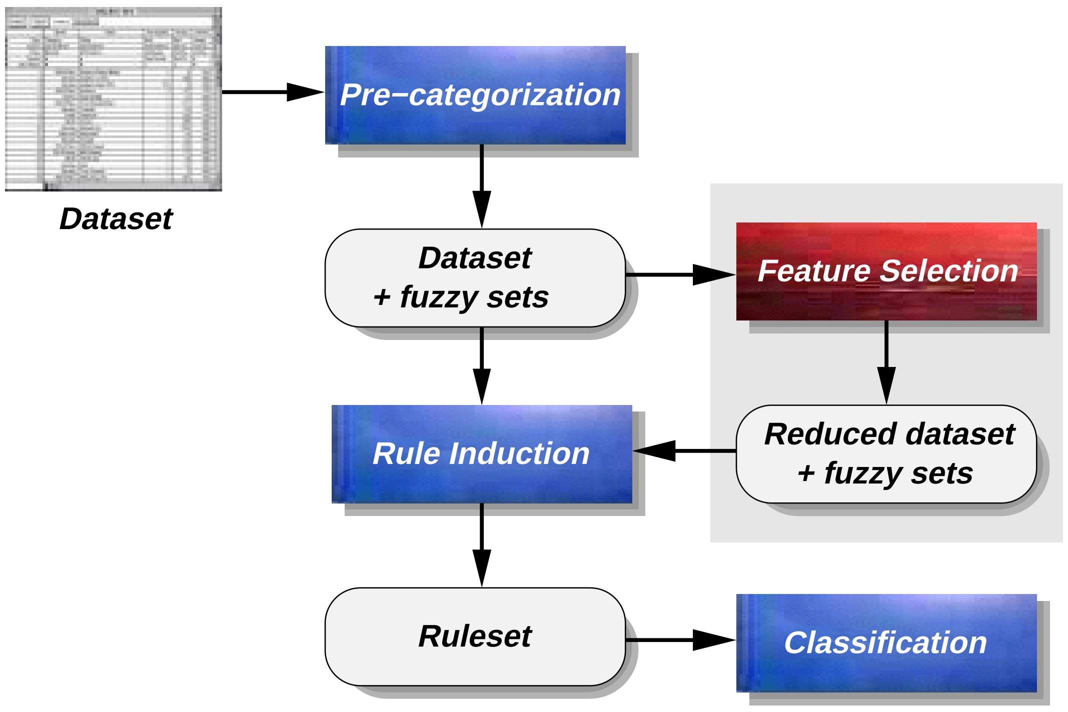


Figure 7.3: Modular decomposition of the implemented system

Originally, a precategorization step preceded feature selection where feature values were quantized. To reduce potential loss of information, the original use of just the dominant symbolic labels of the discretized fuzzy terms is now replaced by a fuzzification procedure. This leaves the underlying feature values unchanged but generates a series of fuzzy sets for each feature. These sets are generated entirely from the data while exploiting the statistical data attached to the dataset (in keeping with the rough set ideology in that the dependence of learning upon information provided outside of the training dataset is minimized). This module may be replaced by alternative fuzzifiers, or expert-defined fuzzification if available. Based on these fuzzy sets and the original real-valued dataset, FRFS calculates a reduct and reduces the dataset accordingly. Finally, fuzzy rule induction is performed on the reduced dataset using the modelling algorithm given in section 7.1. Note that this algorithm is not optimal, nor is the fuzzification. Yet the comparisons given below are fair due to their common background. Alternative fuzzy modelling techniques can be employed for this if available.

7.3 Experimental Results

This section first provides the results for the FRFS-based approach compared with the unreduced approach. Next, a comparative experimental study is carried out between various dimensionality reduction methods; namely FRFS, entropy-based feature selection, PCA and a random reduction technique.

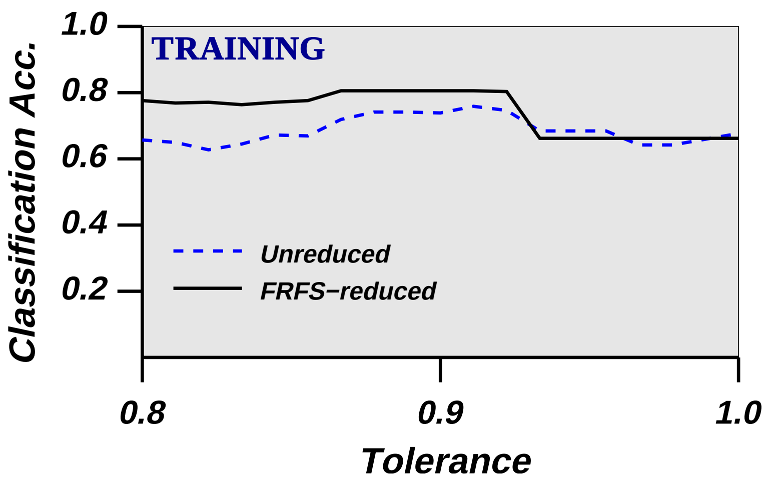
The experiments were carried out over a tolerance range (with regard to the employment of the RIA), in steps of 0.01. As mentioned earlier, a suitable value for the threshold α must be chosen before rule induction can take place. However, the selection of α tends to be an application-specific task; a good choice for this threshold that provides a balance between a resultant ruleset's complexity and accuracy can be found by experiment. It should be noted here that due to the fuzzy rule induction method chosen, all approaches generate exactly the same number of rules (as the number of classes of interest), but the arities in different rulesets differ. This helps avoid a potential complexity factor in the comparative studies due to the need otherwise of considering the sizes of learned rulesets. Only the complexity in each learned rule needs to be examined,

7.3.1 Comparison with Unreduced Features

First of all, it is important to show that, at least, the use of features selected does not significantly reduce the classification accuracy as compared to the use of the full set of original features. For the 2-class problem, the fuzzy-rough set-based feature selector returns 10 features out of the original 38.

Figure 7.4 compares the classification accuracies of the reduced and unreduced datasets on both the training and testing data. As can be seen, the FRFS results are almost always better than the unreduced accuracies over the tolerance range. The best results for FRFS were obtained when α is in the range 0.86 to 0.90, producing a classification accuracy of 83.3% on the training set and 83.9% for the test data. Compare this with the optimum for the unreduced approach, which gave an accuracy of 78.5% for the training data and 83.9% for the test data.

By using the FRFS-based approach, rule complexity is greatly reduced. Figure 7.5



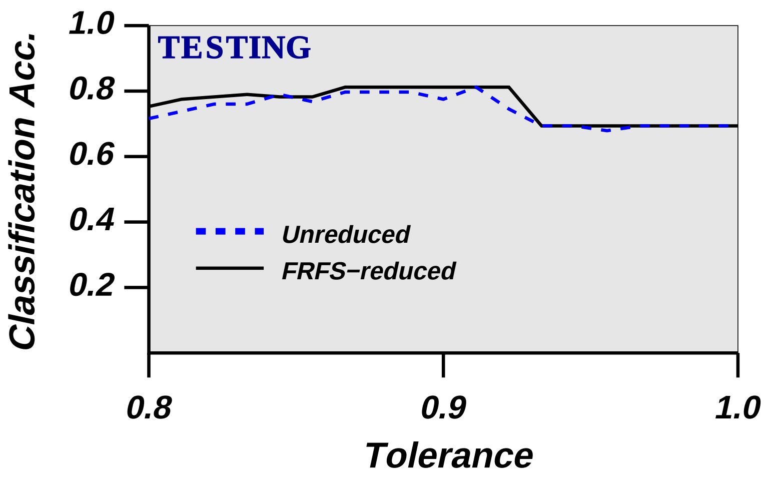


Figure 7.4: Training and testing accuracies for the 2-class dataset over the tolerance range

charts the average rule complexity over the tolerance range for the two approaches. Over the range of α values, FRFS produces significantly less complex rules while having a higher resultant classification accuracy. The average rule arity of the FRFS optimum is 1.5 ($\alpha \in (0.86, 0.9)$) which is less than that of the unreduced optimum, 6.0.

The 3-class dataset is a more challenging problem, reflected in the overall lower classification accuracies produced. The fuzzy-rough method chooses 11 out of the original 38 features. The results of both approaches are presented in figure 7.6. Again, it can be seen that FRFS outperforms the unreduced approach on the whole. The best classification accuracy obtained for FRFS was 70.0% using the training data, 71.8%

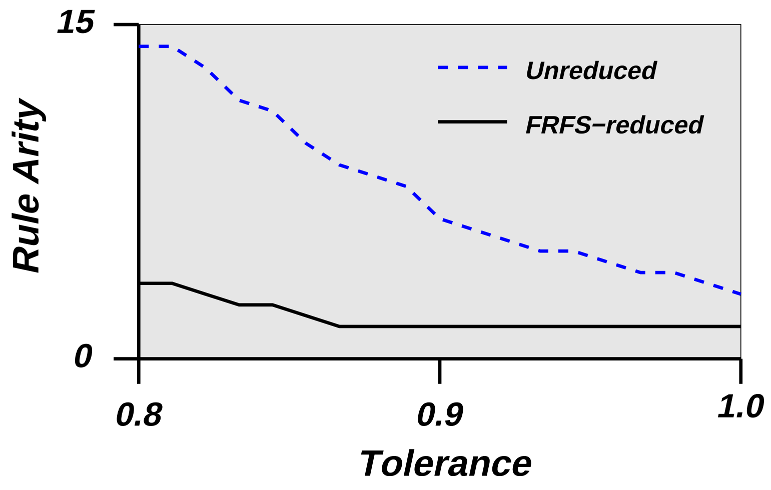


Figure 7.5: Average rule arities for the 2-class dataset

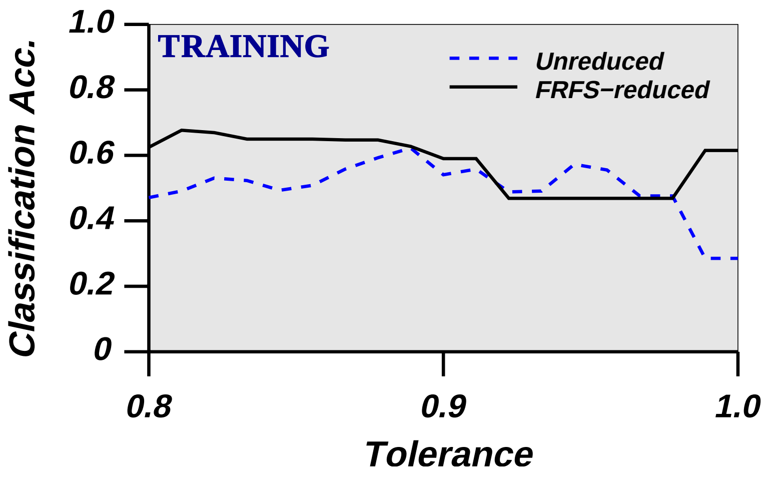
for the test data ($\alpha = 0.81$). For the unreduced approach, the best accuracy obtained was 64.4% using the training data, 64.1% for the test data ($\alpha = 0.88$).

Figure 7.7 compares the resulting rule complexity of the two approaches. It is evident that rules induced using FRFS as a preprocessor are simpler, with little loss in classification accuracy. In fact, the simple rules produced regularly outperform the more complex ones generated by the unreduced approach. The average rule arity of the FRFS optimum is 4.0 which is less than that of the unreduced optimum, 8.33.

These results show that FRFS is useful not only in removing redundant feature measures but also in dealing with the noise associated with such measurements. To demonstrate that the resulting rules are comprehensible, two sets of rules produced by the induction mechanism are given in figure 7.8. The rules produced are reasonably short and understandable. However, when semantics-destroying dimensionality reduction techniques are applied, such readability is lost.

7.3.2 Comparison with Entropy-based Feature Selection

To support the study of the performance of FRFS for use as a pre-processor to rule induction, a conventional entropy-based technique is herein used for comparison. This approach utilizes the entropy heuristic employed by machine learning techniques such



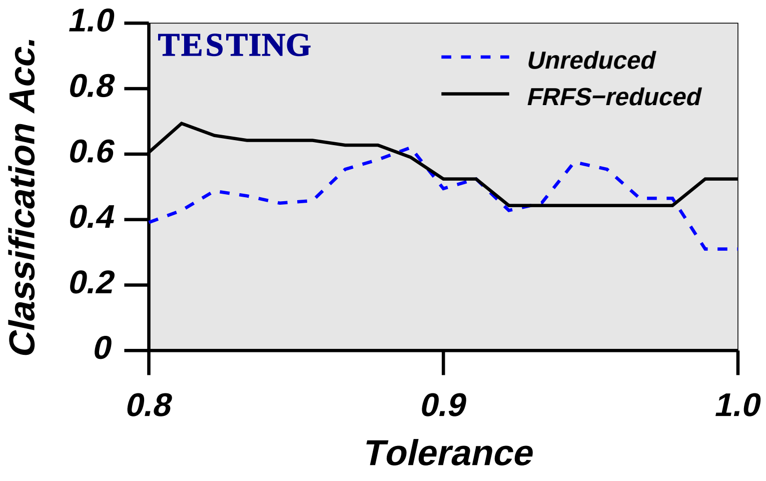


Figure 7.6: Training and testing accuracies for the 3-class dataset over the tolerance range

as C4.5 [135] and EBR (described in section 2.2.1.5). Those features that provide the most gain in information are selected. A summary of the results of this comparison can be seen in table 7.4.

For both the 2-class and 3-class datasets, FRFS selects three fewer features than the entropy-based method. FRFS has a higher training accuracy and the same testing accuracy for the 2-class data using less features. However, for the 3-class data, the entropy-based method produces a very slightly higher testing accuracy. Again, it should be noted that this is obtained with three additional features over the FRFS approach.

Approach	No. of Classes	Selected Features	No. of Features	Training Accuracy	Testing Accuracy
FRFS	2	{0,2,6,10,12,15,22,24,26,37}	10	83.3%	83.9%
Entropy	2	{1,5,6,7,9,12,15,16,20,22,29,30,33}	13	80.7%	83.9%
FRFS	3	{2,3,6,10,12,15,17,22,27,29,37}	11	70.0%	71.8%
Entropy	3	{6,8,10,12,17,21,23,25,26,27,29,30,34,36}	14	70.0%	72.5%

Table 7.4: Comparison of FRFS and entropy-based feature selection

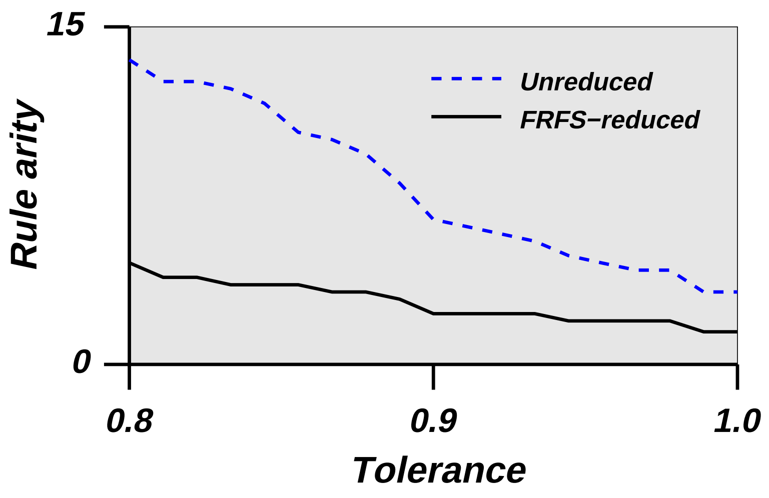


Figure 7.7: Average rule arities for the 3-class dataset

7.3.3 Comparison with PCA and Random Reduction

The previous comparisons ensured that little information loss is incurred due to FRFS. The question now is whether any other feature sets of a dimensionality 10 (for the 2-class dataset) and 11 (for the 3-class dataset) would perform similarly. To avoid a biased answer to this, without resorting to exhaustive computation, 70 sets of random reducts were chosen of size 10 for the 2-class dataset, and a further 70 of size 11 for the 3-class dataset to see what classification results might be achieved. The classification accuracies for each tolerance value are averaged.

The effect of using a different dimensionality reduction technique, namely PCA, is also investigated. To ensure that the comparisons are fair, only the first 10 principal components are chosen for the 2-class dataset (likewise, the first 11 for the 3-class dataset). As PCA irreversibly destroys the underlying dataset semantics, the resulting rules are not human-comprehensible but may still provide useful automatic classifications of new data.

The results of FRFS, PCA and random approaches can be seen in figure 7.9 for the 2-class dataset. On the whole, FRFS produces a higher classification accuracy than both PCA-based and random-based methods over the tolerance range. In fact, FRFS results in the highest individual classification accuracy for training and testing data

Rules from FRFS-reduced data

IF SED-S IS Medium THEN Situation IS Normal

IF PH-E IS NOT High AND SSV-E IS Low AND SSV-P IS NOT Medium AND PH-D IS NOT High AND DQO-D IS NOT Medium AND PH-S IS NOT High THEN Situation IS Good

IF PH-E IS NOT High AND SSV-E IS Low AND SSV-P IS Low AND DQO-D IS NOT High AND SED-S IS Medium THEN Situation IS Faulty

Rules from unreduced data

IF ZN-E IS NOT High AND SS-E IS NOT High AND SED-E IS NOT High AND SSV-D IS NOT High AND DBO-S IS Low AND SS-S IS NOT High AND SED-S IS Low THEN Situation IS Normal

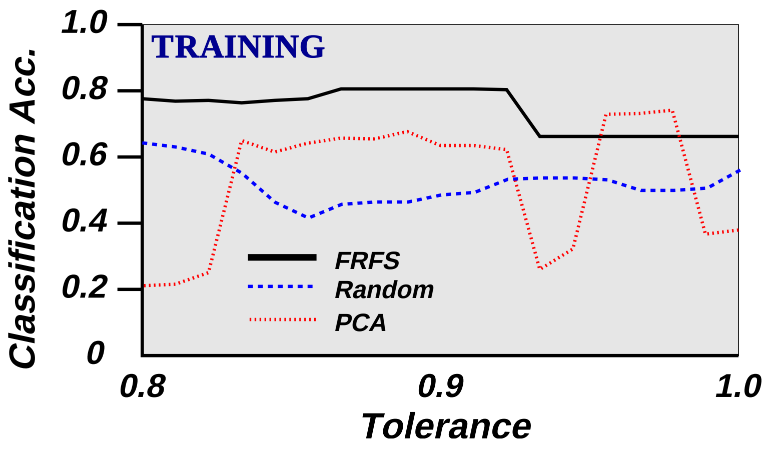
IF ZN-E IS Low AND PH-E IS NOT High AND SSV-E IS NOT High AND PH-P IS NOT High AND SSV-P IS NOT High AND PH-D IS NOT High AND DBO-D IS NOT Medium AND SSV-D IS NOT High AND SS-S IS NOT High THEN Situation IS Good

IF SSV-E IS NOT High AND SSV-P IS Low AND DQO-D IS NOT High AND SSV-D IS NOT High AND SED-D IS NOT High AND DBO-S IS Low AND SS-S IS NOT High AND SSV-S IS NOT High AND SED-S IS Low THEN Situation IS Faulty

Figure 7.8: A selection of generated rulesets

(see table 7.5).

For the 3-class dataset, the results of FRFS, PCA and random selection are shown in figure 7.10. The individual best accuracies can be seen in table 7.6. Again, FRFS produces the highest classification accuracy (71.8%), and is almost always the best over the tolerance range. Although PCA produces a comparatively reasonable accuracy of 70.2%, this is at the expense of incomprehensible rules.



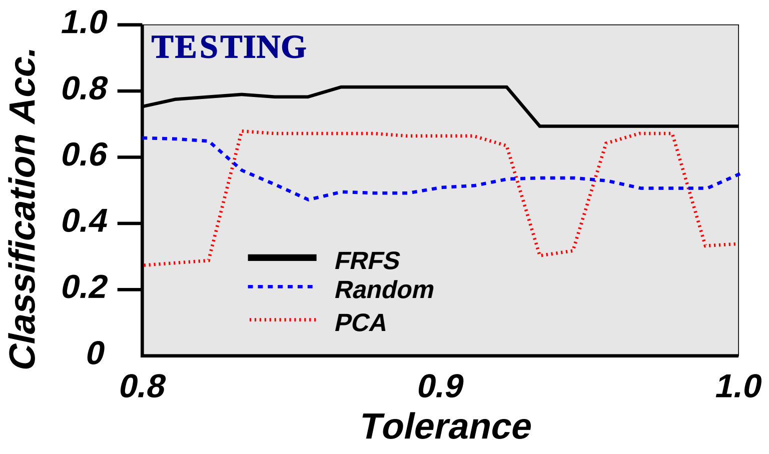


Figure 7.9: Training and testing accuracies for the 2-class dataset: comparison with PCA and random-reduction methods

7.3.4 Alternative Fuzzy Rule Inducer

As stated previously, many fuzzy rule induction algorithms exist and can be used to replace the RIA adopted in the present monitoring system. Here, an example is given using Lozowski's algorithm as presented in [105]. This method extracts linguistically expressed fuzzy rules from real-valued features as with the subsethood-based RIA. Provided with training data, it induces approximate relationships between the characteristics of the conditional features and their underlying classes. However, as with many RIAs, this algorithm exhibits high computational complexity due to its generate-and-test nature. The effects of this become evident where high dimensional data needs to be processed. Indeed, for this particular domain, feature selection is essential as

Approach	Training Accuracy	Testing Accuracy
FRFS	83.3%	83.9%
Random	66.4%	68.1%
PCA	76.7%	70.3%

Table 7.5: Best individual classification accuracies (2-class dataset) for FRFS, PCA and random approaches

Approach	Training Accuracy	Testing Accuracy
FRFS	70.0%	71.8%
Random	55.7%	54.3%
PCA	67.7%	70.2%

Table 7.6: Best resultant classification accuracies (3-class dataset) for FRFS, PCA and random approaches

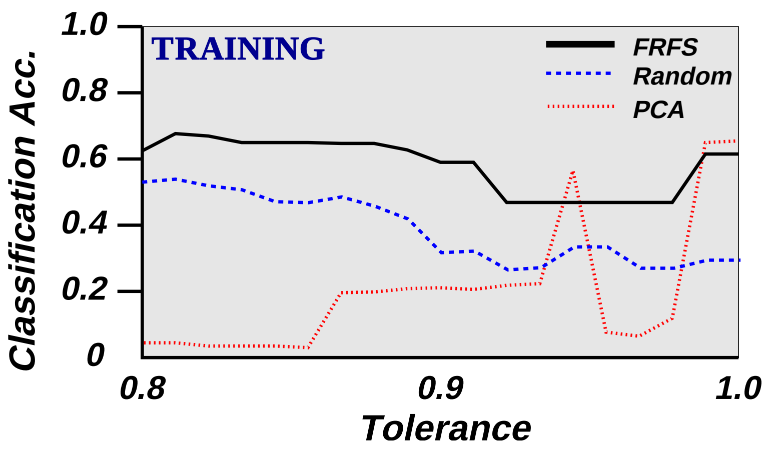
running the RIA on all conditional features would be computationally prohibitive.

The results presented here compare the use of fuzzy-rough set based feature selection with the crisp rough set-based method. For RSAR, the data is discretised using the supplied fuzzy sets and reduction performed on the resulting dataset. The experiments were carried out over a tolerance range, required by the fuzzy RIA. This is a different threshold from those required in the subsethood-based approach. The tolerance here indicates the minimal confidence gap in the decision between a candidate rule and other competing contradictory rules. Again, the threshold is incremented in steps of 0.01.

Method	Attributes 2-class	Attributes 3-class
FRFS	10	11
RSAR	11	11

Table 7.7: Extent of dimensionality reduction

As can be seen from table 7.7, FRFS selects fewer attributes than the crisp method for the 2-class dataset and results in a higher classification accuracy over the entire



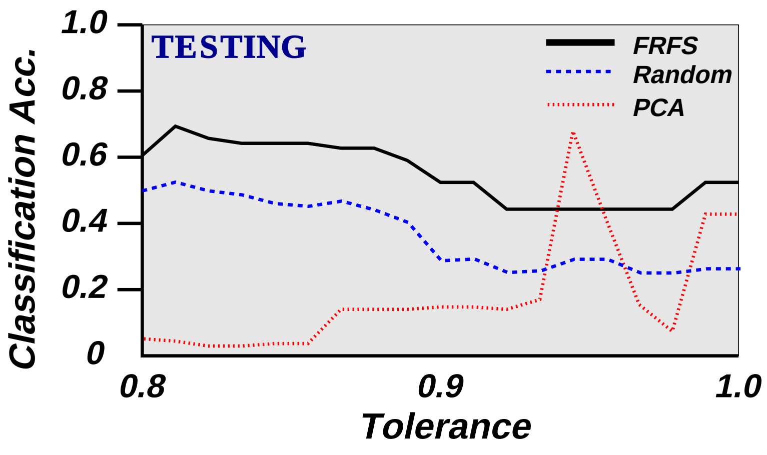


Figure 7.10: Training and testing accuracies for the 3-class dataset: comparison with PCA and random-reduction methods

tolerance range (figure 7.11). Both results show that there is a lot of redundancy in the dataset which may be removed with little loss in classification accuracy. For the 3-class dataset the approaches perform similarly, with the FRFS method generally outperforming the other two, using the same number of attributes (but not *identical* attributes). The classification results can be seen in figure 7.12.

7.3.5 Results with Feature Grouping

The experiments were carried out over a tolerance range (with regard to the employment of the RIA). It is worth reiterating here that due to the fuzzy rule induction method chosen (see section 7.1), all approaches generate exactly the same number of rules (as

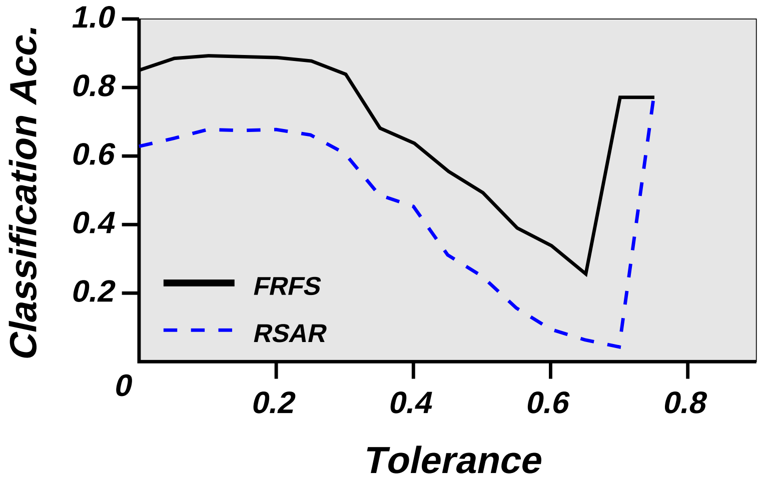


Figure 7.11: Classification accuracies for the 2-class dataset

the number of classes of interest), but the arities in different rulesets will differ. In all experiments here, FQR employs the strategy where all attributes belonging to *Large* (as defined in section 5.1) are selected at each stage.

For the 2-class problem, the FRFS feature selector returns 10 features out of the original 38, whereas FQR returns 12. Figure 7.13 compares the classification accuracies of the reduced and unreduced datasets on both the training and testing data. As can be seen, both the FRFS and FQR results are almost always better than the unreduced accuracies over the tolerance range. The best results for FQR were obtained in the range 0.85 to 0.88, producing a classification accuracy of 82.6% on the training set and 83.9% for the test data. For FRFS, the best accuracies were 83.3% (training) and 83.9% (testing). Compare this with the optimum for the unreduced approach, which gave an accuracy of 78.5% for the training data and 83.9% for the test data.

The 3-class dataset is a more challenging problem, reflected in the overall lower classification accuracies produced. Both fuzzy-rough methods, FQR and FRFS, choose 11 out of the original 38 features (but not the *same* features). The results of these approaches can be seen in figure 7.14. Again, the results show that both FQR and FRFS outperform the unreduced approach on the whole. The best classification accuracy obtained for FQR was 72.1% using the training data, 74.8% for the test data. For FRFS the best results were 70.0% (training) and 71.8% (testing). In this case, FQR has found a better reduction of the data. For the unreduced approach, the best accuracy obtained

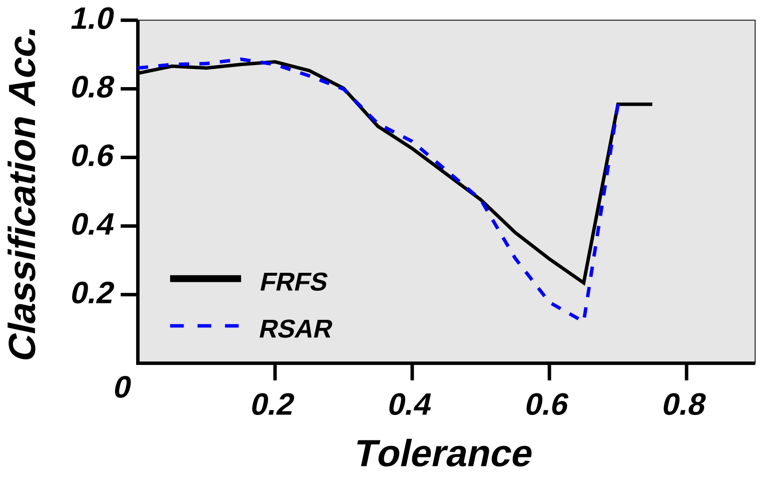


Figure 7.12: Classification accuracies for the 3-class dataset

was 64.4% using the training data, 64.1% for the test data.

7.3.6 Results with Ant-based FRFS

The results from various comparative studies regarding ant-based FRFS are presented here. The first set of experiments compares the hill-climbing and ant-based fuzzy-rough methods. An investigation into another feature selector based on the entropy measure is then presented. This is followed by comparisons with PCA and support vector classifiers [131].

7.3.6.1 Comparison of Fuzzy-Rough Methods

Three sets of experiments were carried out on both the (collapsed) 2-class and 3-class datasets. The first bypasses the feature selection part of the system, using the original water treatment dataset as input to C4.5, with all 38 conditional attributes. The second method employs FRFS to perform the feature selection before induction is carried out. The third uses the ant-based method described in section 5.2.3 (antFRFS) to perform feature selection over a number of runs, and the results averaged.

The results for the 2-class dataset can be seen in table 7.8. Both FRFS and antFRFS significantly reduce the number of original attributes with antFRFS producing the greatest data reduction on average. As well as generating smaller reducts, antFRFS

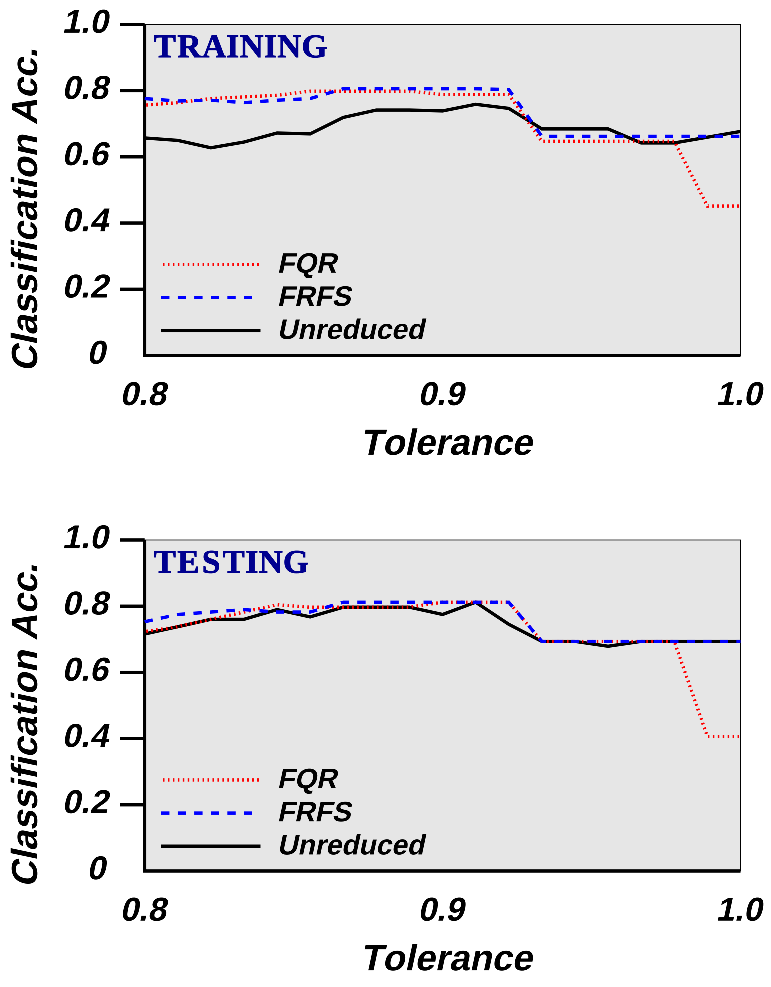
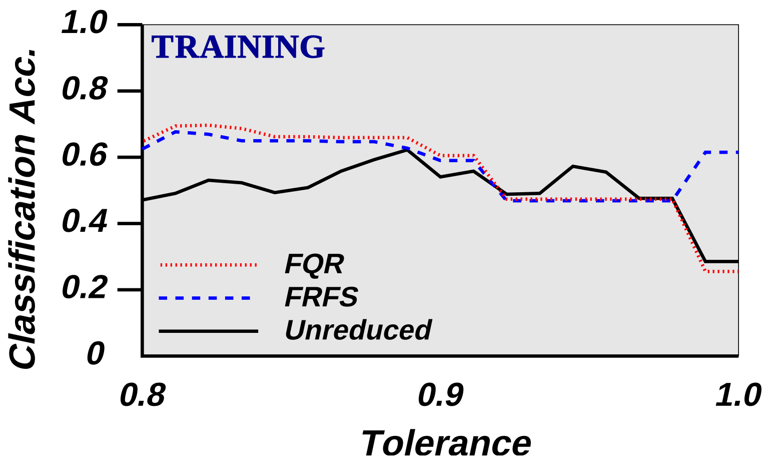


Figure 7.13: Classification accuracies for the 2-class dataset

finds reducts of a higher quality according to the fuzzy-rough dependency measure. This higher quality is reflected in the resulting classification accuracies for both the training and testing datasets, with antFRFS outperforming FRFS.

Table 7.9 shows the results for the 3-class dataset experimentation. The hill-climbing fuzzy-rough method chooses 11 out of the original 38 features. The ant-based method chooses fewer attributes on average, however this is at the cost of a lower dependency measure for the generated reducts. Again the effect of this can be seen in the classification accuracies, with FRFS performing slightly better than antFRFS. For both fuzzy methods, the small drop in accuracy as a result of feature selection is acceptable.

By selecting a good feature subset from data it is usually expected that the applied



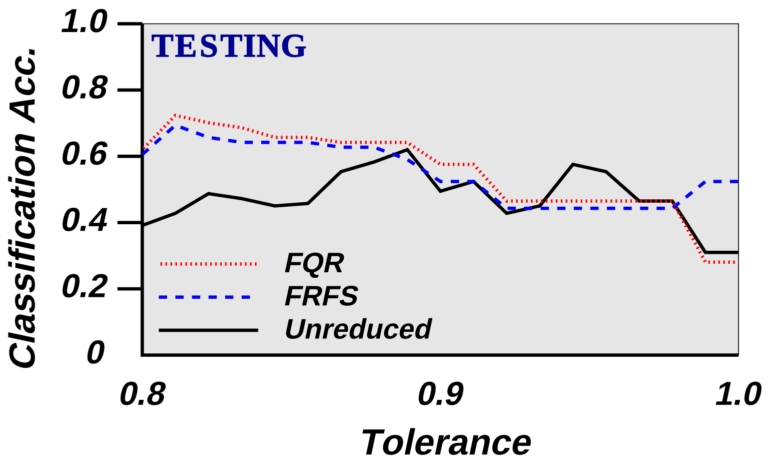


Figure 7.14: Classification accuracies for the 3-class dataset

learning method should benefit, producing an improvement in results. However, when the original training (and test) data is very noisy, selected features may not necessarily be able to reflect all the information contained within the original entire feature set. As a result of removing less informative features, partial useful information may be lost. The goal of selection methods in this situation is to minimise this loss, while reducing the number of features to the greatest extent. Therefore, it is not surprising that the classification performance for this challenging dataset can decrease upon data reduction, as shown in table 7.9. However, the impact of feature selection can have different effects on different classifiers. With the use of an alternative classifier in section 7.3.6.4, performance can be seen to improve for the test data.

Method	Attributes	γ’ value	Training accuracy	Testing accuracy
Unreduced	38	-	98.5%	80.9%
FRFS	10	0.58783	89.2%	74.8%
antFRFS	9.55	0.58899	93.5%	77.9%

Table 7.8: Results for the 2-class dataset

Method	Attributes	γ’ value	Training accuracy	Testing accuracy
Unreduced	38	-	97.9%	83.2%
FRFS	11	0.59479	97.2%	80.9%
antFRFS	9.09	0.58931	94.8%	80.2%

Table 7.9: Results for the 3-class dataset

The results here also show a marked drop in classification accuracy for the test data. This could be due to the problems encountered when dealing with datasets of small sample size. Overfitting can occur, where a learning algorithm adapts so well to a training set, that the random disturbances present are included in the model as being meaningful. Consequently, as these disturbances do not reflect the underlying distribution, the performance on the test data will suffer. Although such techniques as cross-validation and bootstrapping have been proposed as a way of countering this, these still often exhibit high variance in error estimation [22].

7.3.6.2 Comparison with Entropy-based Feature Selection

As with the experimental studies carried out in chapter 6, and in section 7.3.2, to support the investigation of the performance of the fuzzy-rough methods, a conventional entropy-based technique is herein used again for comparison. A summary of the results of this comparison can be seen in table 7.10.

For both the 2-class and 3-class datasets, FRFS and antFRFS select at least three fewer features than the entropy-based method. However, the entropy-based method outperforms the other two feature selectors with the resulting C4.5 classification accuracies. This is probably due to the fact that C4.5 uses exactly the same entropy measure in generating decision trees. In this case, the entropy-based measure will favour those attributes that will be the most influential in the decision tree generation

Approach	No. of Classes	No. of Features	Training Accuracy	Testing Accuracy
FRFS	2	10	89.2%	74.8%
antFRFS	2	9.55	93.5%	77.9%
Entropy	2	13	97.7%	80.2%
FRFS	3	11	97.2%	80.9%
antFRFS	3	9.09	94.8%	80.2%
Entropy	3	14	98.2%	80.9%

Table 7.10: Results for the three selection methods

process.

7.3.6.3 Comparison with the use of PCA

The effect of using PCA [38] is also investigated. Again, PCA is applied to the dataset and the first *n* principal components are used. A range of values is chosen for *n* to investigate how the performance varies with dimensionality. The results are summarised in table 7.11.

		No. of Features								
Accuracy	Class	5	6	7	8	9	**10**	11	12	13
Training (%)	2	80.0	80.0	80.0	80.0	80.3	**80.3**	80.3	80.8	82.1
Testing (%)	2	72.5	72.5	72.5	72.5	73.3	**73.3**	73.3	35.1	34.4
Training (%)	3	73.6	73.6	73.6	73.6	73.6	**75.9**	75.9	75.9	76.4
Testing (%)	3	80.9	80.9	80.9	80.9	80.9	**80.9**	80.9	80.9	80.2

Table 7.11: Results for the 2-class and 3-class datasets using PCA

Both antFRFS and FRFS significantly outperform PCA on the 2-class dataset. Of particular interest is when 10 principal components are used as this is roughly the same number chosen by antFRFS and FRFS. The resulting accuracy for PCA is 80.3% for the training data and 73.3% for the test data. For antFRFS the accuracies were 93.5% (training) and 77.9% (testing), and for FRFS 89.2% (training) and 74.8% (testing). In the 3-class dataset experimentation, both fuzzy-rough methods produce much higher classification accuracies than PCA for the training data. For the test data, the performance is about the same, with PCA producing a slightly higher accuracy than antFRFS on the whole.

7.3.6.4 Comparison with the use of a support vector classifier

A possible limitation of employing C4.5 in this context is that it performs a degree of feature selection itself during the induction process. The resulting decision trees do not necessarily contain all the features present in the original training data. As a result of this, it is beneficial to evaluate the use of an alternative classifier that uses all the given features. For this purpose, a support vector classifier [149] is used, trained by the sequential minimal optimization (SMO) algorithm [131]. The results of the application of this classifier can be found in table 7.12.

Approach	No. of Classes	No. of Features	Training Accuracy	Testing Accuracy
Unreduced	2	38	80.0%	71.8%
FRFS	2	10	80.0%	72.5%
antFRFS	2	9.55	80.0%	72.5%
Unreduced	3	38	74.6%	80.9%
FRFS	3	11	73.6%	80.2%
antFRFS	3	9.09	73.6%	80.9%

Table 7.12: Results for the 2-class and 3-class datasets using SMO

For the 2-class dataset, the training accuracy for both FRFS and antFRFS is the same as that of the unreduced approach. However, this is with significantly fewer attributes. Additionally, the resulting testing accuracy *is* increased with these feature selection methods. With the more challenging 3-class problem, the training accuracies are slightly worse (as seen with the C4.5 analysis). The antFRFS method performs better than FRFS for the test data and is equal to the unreduced method, again using fewer features.

7.4 Summary

Automated generation of feature pattern-based if-then rules is essential to the success of many intelligent pattern classifiers, especially when their inference results are expected to be directly human-comprehensible. This chapter has evaluated such an approach which integrates a recent fuzzy rule induction algorithm with a fuzzy-rough method for feature selection. Unlike semantics-destroying approaches such as PCA, this approach maintains the underlying semantics of the feature set, thereby ensuring that the resulting models are interpretable and the inference explainable. The method alleviates important problems encountered by traditional RSAR such as dealing with noise and real-valued features.

Overall, the experimental results presented in this chapter have shown the utility of employing FRFS, ant-based FRFS and FQR as pre-processors to fuzzy rule induction. The work has demonstrated the potential benefits of using fuzzy dependencies and attribute grouping in the search for reducts. Not only are the runtimes of the induction and classification processes improved by this step (which for some systems are important factors), but the resulting rules are less complex.

In all experimental studies there has been no attempt to optimize the fuzzifications or the classifiers employed. It can be expected that the results obtained with optimization would be even better than those already observed. The generality of this approach should enable it to be applied to other domains. The ruleset generated by the RIA was not processed by any post-processing tools so as to allow its behaviour and capabilities to be revealed fully. By enhancing the induced ruleset through post-processing, performance should improve.

Chapter 8

Supplementary Developments and Investigations

This chapter presents other developments that have been initiated and future work in the areas of feature selection and rule induction, arising from some of the issues discussed in this thesis. For feature selection, RSAR-SAT is proposed based on the discernibility matrix approach to finding reducts and employing techniques from propositional satisfiability. Additionally, the concept of fuzzy universal reducts is proposed as a way of enabling more accurate data reductions. Finally, a new decision tree induction method based on the fuzzy-rough metric is suggested.

8.1 RSAR-SAT

The Propositional Satisfiability (SAT) problem [35] is one of the most studied NP-complete problems because of its significance in both theoretical research and practical applications. Given a boolean formula (typically in conjunctive normal form (CNF)), the SAT problem requires an assignment of variables/features so that the formula evaluates to true, or a determination that no such assignment exists. In recent years search algorithms based on the well-known Davis-Logemann-Loveland algorithm (DPLL) [36] are emerging as some of the most efficient methods for complete SAT solvers. Such solvers can either find a solution or prove that no solution exists.

Stochastic techniques have also been developed in order to reach a solution quickly. These pick random locations in the space of possible assignments and perform limited local searches from them. However, as these techniques do not examine the entire search space, they are unable to prove unsatisfiability.

A CNF formula on n binary variables $x_1,...,x_n$ is the conjunction of m clauses $C_1,...,C_m$ each of which is the disjunction of one or more literals. A literal is the occurrence of a variable or its negation. A formula denotes a unique n-variable boolean function $f(x_1,...,x_n)$. Clearly, a function f can be represented by many equivalent CNF formulas. The satisfiability problem is concerned with finding an assignment to the arguments of $f(x_1,...,x_n)$ that makes the function equal to 1, signalling that it is satisfiable, or proving that the function is equal to 0 and hence unsatisfiable [202].

The problem of finding the smallest feature subsets using rough set theory can be formulated as a SAT problem. Rough sets allows the generation from datasets of clauses of features in conjunctive normal form. If after assigning truth values to all features appearing in the clauses the formula is satisfied, then those features set to true constitute a valid subset for the data. The task is to find the smallest number of such features so that the CNF formula is satisfied. In other words, the problem here concerns finding a minimal assignment to the arguments of $f(x_1,...,x_n)$ that makes the function equal to 1. There will be at least one solution to the problem (i.e. all x_is set to 1) for consistent datasets. Preliminary work has been carried out in this area [7], though this does not adopt a DPLL-style approach to finding solutions.

The DPLL algorithm for finding minimal subsets can be found in figure 8.1, where a search is conducted in a depth-first manner. The key operation in this procedure is the unit propagation step in lines (6) and (7). Clauses in the formula that contain a single literal will only be satisfied if that literal is assigned the value 1 (for positive literals). These are called unit clauses. Unit propagation examines the current formula for unit clauses and automatically assigns the appropriate value to the literal they contain. The elimination of a literal can create new unit clauses, and thus unit propagation eliminates variables by repeated passes until there is no unit clause in the formula.

Branching occurs at lines (9) to (12). Here, the next variable is chosen heuristically from the current formula, assigned the value 1, and the search continues. If this branch eventually results in unsatisfiability, the procedure will assign the value 0 to this variable instead and continue the search. A degree of pruning can take place in the search by remembering the size of the currently considered subset and the smallest optimal subset encountered so far. If the number of variables currently assigned 1 equals the number of those in the presently optimal subset, and the satisfiability of F is still not known, then any further search down this branch will not result in a smaller optimal subset.

DPLL(F).
F, the formula containing the current set of clauses.

```
(1)  if ($F$ contains an empty clause)
(2)      return unsatisfiable
(3)  if ($F$ is empty)
(4)      output current assignment
(5)      return satisfiable
(6)  if ($F$ contains a unit clause $\{l\}$)
(7)      $F' \leftarrow$ unitPropagate($F$)
(8)      return DPLL($F'$)
(9)  $x \leftarrow$ selectLiteral($F$)
(10) if ( DPLL($F \cup \{x\}$) is satisfiable)
(11)     return satisfiable
(12) else return DPLL($F \cup \{-x\}$)
```

Figure 8.1: The definition of the DPLL algorithm

Although stochastic methods [153, 66] have been applied to SAT problems, these are not applicable here as they provide no guarantee of solution minimality. The DPLL-based algorithm will always find the minimal optimal subset. However, this will come at the expense of time taken to find it. Initial experimentation has been carried out using the algorithm outlined previously; the average times are presented in table 8.1. The time taken for RSAR-SAT is split into two columns. The first indicates the average length of time taken to find the minimal subset, the second how long it takes to verify that this is indeed minimal. For RSAR, an asterisk next to the time indicates that it found a non-minimal reduct.

The results show that RSAR-SAT is comparable to RSAR in the time taken to

Dataset	No. of Features	RSAR (s)	SAT: Minimal (s)	SAT: Full (s)
M-of-N	13	0.171*	0.001	0.007
Exactly	13	0.304*	0.001	0.008
Exactly2	13	0.823*	0.001	0.008
Heart	13	0.207*	0.002	0.009
Vote	16	0.170*	0.004	0.009
Credit	20	1.988*	0.077	0.094
Mushroom	22	0.744*	0.006	0.022
LED	24	0.097*	0.041	0.051
Letters	25	0.067*	0.024	0.116
Derm	34	0.758*	0.094	0.456
Derm2	34	0.897*	0.104	0.878
WQ	38	9.590*	0.205	116.1
Lung	56	0.059	0.023	0.786
DNA	58	1.644*	0.227	53.81

Table 8.1: Runtimes for RSAR and RSAR-SAT

find reducts. However, RSAR regularly fails to find the smallest optimal subset, being misled in the search process. For larger datasets, the time taken for RSAR-SAT verification exceeds that of RSAR. There are ways in which the DPLL-based search can be made more effective and less time-consuming.

DPLL resorts to chronological backtracking if the current assignment of variables results in the unsatisfiability of F. Much research has been carried out in developing solution techniques for SAT that draws on related work in solvers for constraint satisfaction problems (CSPs) [9, 184]. Indeed the SAT problem can be translated to a CSP by retaining the set of boolean variables and their $\{0,1\}$ domains, and to translate the clauses into constraints. Each clause becomes a constraint over the variables in the constraint. Unit propagation can be seen to be a form of forward checking.

In CSPs, more intelligent ways of backtracking have been proposed such as backjumping, conflict-directed backjumping and dynamic backtracking. Many aspects of these have been adapted to the SAT problem solvers. In these solvers, whenever a conflict (dead-end) is reached, a new clause is recorded to prevent the occurrence of the same conflict again during the subsequent search. Non-chronological backtracking backs up the search tree to one of the identified causes of failure, skipping over irrelevant variable assignments.

With the addition of intelligent backtracking, RSAR-SAT should be able to handle datasets containing large numbers of features. As seen in the preliminary results, the bottleneck in the process is the verification stage - the time taken to confirm that the subset is indeed minimal. This requires an exhaustive search of all subtrees containing fewer variables than the current best solution. Much of this search could be avoided through the use of more intelligent backtracking. This would result in a selection method that can cope with many thousands of features, whilst guaranteeing resultant subset minimality - something that is particularly sought after in feature selection.

8.2 Fuzzy Universal Reducts

A single given dataset may contain more than one optimal feature subset. For the simple example given in section 3.1, there are two minimal subsets. For larger problems, there may be many more. Typically, only one of these subsets is chosen to perform the necessary dataset reduction. The problem here is how to discern between reducts in order to choose the best one to reduce the data.

A solution to this problem may be *fuzzy universal reducts*. Instead of committing to a single reduct, a fuzzy universal reduct captures the frequency information of features within all reducts for a dataset. Those features that appear often in subsets are determined to be more significant than those that appear less frequently. It is thought that by capturing this information, a better subset of features can be selected for reducing data. The techniques suggested here can be applied to the fuzzy-rough as well as the crisp rough set problem.

The set of all reducts, T, for a dataset can be calculated in the following way:

$$T = \{X : X \subseteq \mathbb{C}, \gamma_X(\mathbb{D}) = \gamma_{\mathbb{C}}(\mathbb{D})\} \tag{8.1}$$

A fuzzy universal reduct R for a dataset is defined $(\forall x \in \mathbb{C})$ as:

$$\mu_R(x) = \frac{|\{x | x \in S, \forall S \in T\}|}{|T|} \tag{8.2}$$

An attribute a belongs fully to the reduct ($\mu_R(a) = 1$) if it appears in all reducts (a member of the *core*). By applying α-cuts, subsets of the attributes may be selected for dataset reduction. However, to calculate all reducts for non-trivial datasets is not feasible. Therefore, some way is needed to approximate these memberships. It is thought that the use of pheromone in ant colony optimization-based feature selection might act as such an approximater.

The basic idea is that during the course of their search, ants traverse the graph of nodes (features) depositing artificial pheromone. The amount of pheromone on an edge is an indicator of the utility of that edge in previous searches for optimal subsets. A useful node will have high amounts of pheromone on edges connected to it. By combining the values of the pheromone levels on edges connected to a node, a measure of that node's significance can be obtained.

Another means of estimating memberships to a fuzzy universal reduct could be obtained through a genetic algorithm-based feature selection process. As the algorithm searches for optimal reducts, it encounters and evaluates many subsets along the way. By recording the information concerning the frequency of occurrence of features (and how 'good' the subset is), an estimate of feature importance can be determined.

8.3 Fuzzy-Rough Decision Trees

A decision tree can be viewed as a partitioning of the instance space. Each partition, represented by a leaf, contains the objects that are similar in relevant respects and thus are expected to belong to the same class. The partitioning is carried out in a data-driven manner, with the final output representing the partitions as a tree. An important property of decision tree induction algorithms is that they attempt to minimize the size of the tree at the same time as they optimize a certain quality measure.

The general decision tree induction algorithm is as follows. The significance of features is computed using a suitable measure (in C4.5 this is the information gain metric [135]). Next, choose the most significant feature according to this measure and partition the dataset into sub-tables according to the values this feature may take. The chosen feature is represented as a node in the currently constructed tree. For each sub-table, repeat the above procedure, i.e. determine the most significant feature and split the data into further sub-tables according to its values.

This is a similar process in the fuzzy case. However, a measure capable of handling fuzzy terms (instead of crisp values) must be used. Data is partitioned according to the selected feature's set of fuzzy terms. There must also be a way of calculating the number of examples that belong to a node. In the crisp case, this is clear; objects either contain a specific attribute value or they do not. In the fuzzy case this distinction can no longer be made, as objects may belong to several fuzzy terms.

Clearly, one important aspect of this procedure is the choice of feature significance measure. It would therefore be interesting to see how an induction algorithm based on the fuzzy-rough measure would compare with standard algorithms such as fuzzy ID3 [69, 182], both in terms of the complexity of the trees constructed and the resulting accuracy. A suitable stopping condition must also be chosen that will limit the number of nodes expanded.

$x \in \mathbb{U}$	Inc	Emp	$\Rightarrow$	$Cred$
0	0.20	0.15		0.0
1	0.35	0.25		0.0
2	0.90	0.20		0.0
3	0.60	0.50		0.0
4	0.90	0.50		1.0
5	0.10	0.85		1.0
6	0.40	0.90		1.0
7	0.85	0.85		1.0

Table 8.2: A simple dataset (taken from [69])

For example. consider the data given in table 8.2. This contains two conditional features, Income (Inc) and Employment (Emp), and one binary decision feature, Credit ($Cred$). The fuzzy sets used for both conditional features can be seen in figure 8.2.

Figure 8.3 depicts the data in a more visual manner. The data objects (numbered

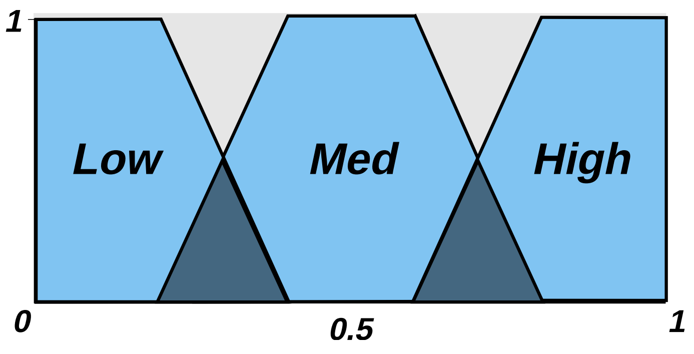


Figure 8.2: Corresponding feature fuzzifications

1 to 8) are plotted according to their values for the two attributes, Income and Employment. Objects classified to "0.0" are coloured black, objects classified to "1.0" are white. The extent to which objects belong to the fuzzy sets is indicated by the degree of blue: light blue indicates full set membership, darker blue indicates partial membership.

From this, it can be seen that using the feature "Employment" to partition the data via its fuzzy terms provides better class separability than "Income". In other words (from the visualization perspective) horizontal splits are more useful than vertical. Using "Employment", objects $\{0,1,2\}$ and $\{5,6,7\}$ can be distinguished. Using "Income", no such distinctions can be made.

Applying the fuzzy-rough metric to this dataset results in:

$$\gamma_{\{Inc\}}(\{Cred\}) = 0.0 \tag{8.3}$$

$$\gamma_{\{Emp\}}(\{Cred\}) = 0.71875 \tag{8.4}$$

The fuzzy-rough metric again finds the most useful feature and determines that the "Income" feature provides no class separability at this stage. Incidentally, the dependency of the entire dataset is 0.96875. This is less than 1 due to the extra uncertainty caused by object 1, which can be seen in the data visualization. From this initial examination, it appears to be the case that the fuzzy-rough metric could indeed be used in a new fuzzy decision tree induction method.

Fuzzy-rough sets might also be used in fuzzy rule induction. Crisp rule induction

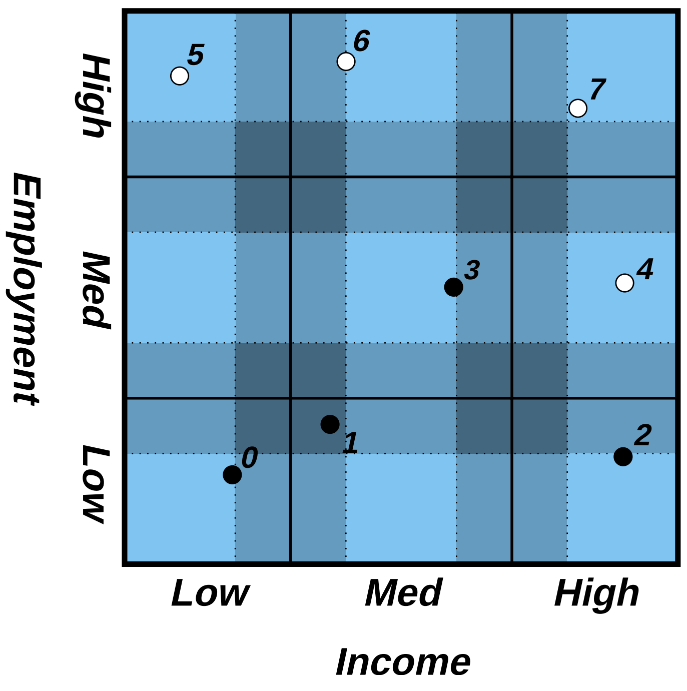


Figure 8.3: Data visualization

based on crisp rough sets has been an area of much interest and success within the rough set community [139, 140, 175]. By extending these approaches, or developing an entirely new mechanism suited to fuzzy-rough sets, it should be possible to develop fuzzy-rough rule induction methods.

8.4 Summary

This chapter has presented several potential directions for crisp and fuzzy-rough set based approaches to feature selection. One particular area of interest is that of reformulating the search for a reduct as a propositional satisfiability problem. Solution techniques from this domain can then be applied to locate the smallest reducts. In addition to this and other areas of further research in feature selection, the possibility of inducing decision trees using the fuzzy-rough metric was discussed.

Chapter 9

Conclusion

This chapter concludes the thesis. A summary of the research presented in this dissertation is given, with a focus on the main contribution, fuzzy-rough feature selection, and its applications. Based on a critical survey of the existing literature, it was seen to be the case that many feature selection methods rely on a preliminary discretization procedure in an attempt to deal with noisy and real-valued data. This is particularly the case with rough set-based approaches which depend entirely on crisp datasets. Although this provides a makeshift solution to the problem, it is reliant upon a good discretization that incorporates noise-elimination to produce a useful resulting data reduction. In fact, there may be situations where a dataset contains both nominal and real-valued conditional features.

9.1 Fuzzy-Rough Feature Selection

This thesis has been chiefly concerned with the utility of fuzzy-rough feature selection as a way of combatting the problems of noisy and real-valued data, as well as handling mixtures of nominal and continuous valued features. FRFS achieves this by the use of fuzzy-rough sets, and the new measure of feature significance: the fuzzy-rough degree of dependency. The properties of fuzzy-rough sets allow greater flexibility when handling noisy and real-valued data. A particular issue for feature selectors is the problem of real-valued decision attributes. FRFS can deal with this whereas many

FS techniques cannot.

The new fuzzy-rough metric was experimentally evaluated against other leading metrics for use in feature ranking. The results confirmed that the fuzzy-rough measure performs comparably to these metrics, and better than them in several cases. When applied to a real world dataset, the new metric was able to identify the significant features as defined by a human expert. All of this required no threshold information or additional input from a user, which is not the case for most conventional approaches.

Two new areas for research within feature selection were also presented. The first, feature grouping, introduced the concept of fuzzifying the evaluation function in feature selectors for use in a more transparent selection process. Also, by grouping features according to their linguistic label and selecting them simultaneously, the time taken for the search process could be reduced. The second development proposed an ant colony optimization-based framework to perform feature selection. This was introduced in an attempt to tackle the problem of the non-minimal subsets often generated by the greedy hill-climbing search strategy. Both of these developments can be used for feature selectors in general and not just for crisp and fuzzy-rough selection.

To demonstrate the applicability of the developments presented in this thesis, two very different problem domains were chosen: web content categorisation and complex systems monitoring.

9.2 Web Content Categorisation

With the explosive growth of available information on the web, it is essential that applications devoted to its organization and management can effectively handle this abundance of data. For machine learning tasks in particular, the sheer size of the datasets involved can be prohibitive. In the area of text categorisation, datasets are of the order of thousands, and sometimes tens of thousands, of features. This makes effective categorisation an extremely difficult problem unless the task is simplified. This is typically achieved through the use of feature selectors. This thesis presented an approach that incorporated a fuzzy-rough feature selection stage for the automated classification of web content. Two specific areas were targetted in the web domain, namely bookmark categorisation and web page categorisation.

Although techniques exist for organising bookmark databases, these always use the content of the referred-to document to generate classifications. The system developed here seeks to categorise bookmarks using only the information stored locally in the bookmark database. Given that there is very little information stored in a bookmark, this task is a particularly challenging one. However, this still results in the generated training datasets containing over a thousand features. FRFS was shown to be effective at greatly reducing this dimensionality with little resulting impact on the overall classification ability of the system.

Another important area is that of web page categorisation. The need for automated classification of web pages is demonstrated by the popularity of web directories such as Yahoo [192]. Many of these are developed and maintained manually which is becoming increasingly impractical due to the dynamicity and evolution of the web. Therefore, an automated means of deriving correct categories from text contained within web pages is required. Again, the dimensionality of the datasets involved are of the order of thousands to tens of thousands. FRFS was used to tackle this restrictive amount of data successfully within a web page categorisation system. In fact, the extent of data reduction was several orders of magnitude, making the categorisation task manageable.

9.3 Complex Systems Monitoring

The knowledge acquisition bottleneck is a significant problem that hinders the building of intelligent monitoring systems. The generation of good knowledge bases for this task is notoriously difficult. This is particularly the case where experts are not readily available. Machine learning techniques (especially rule induction methods) can be of great benefit to this area by providing strategies to automatically extract useful knowledge, given enough historical data.

For many of these techniques, the high dimensionality of the domain attributes can be too restrictive. In addition, when applying rule inducers that can cope with this size of data, the resulting knowledge in the form of rules can be extremely difficult to interpret. The control system itself then has to operate using complex rules, degrading the overall performance. Dimensionality reduction is required to alleviate this problem. In particular, a semantics-preserving approach must be used in order to ensure that the resulting ruleset is readable.

FRFS was applied to this domain to show how not only rule clarity can be significantly improved with feature selection, but also that the reduced knowledge base can achieve competitive results in terms of monitoring accuracy. The fuzzy-rough method was shown to perform very well against other feature selector methods for this task. Additionally, the new feature grouping and ant-based FRFS techniques were applied with promising results.

9.4 Future Work

Much consideration has been given to future work in the area of feature selection (and also decision tree induction) as outlined in chapter 8. These initial investigations demonstrate several interesting possibilities for future research. Particularly the use of a DPLL-style search, in a propositional satisfiability setting, for finding minimal feature subsets (section 8.1) appears to be a promising area. With the use of more intelligent backtracking techniques, the method should be able to cope with datasets containing many thousands of features, locating the smallest feature subset rapidly. This technique also guarantees that the final subset found is indeed the smallest - something that most rough set-based methods are unable to demonstrate.

There are many other areas that benefit from a data reduction step (as discussed in chapter 1). It would be highly beneficial to investigate how FRFS may be applied to other domains such as image recognition, gene expression analysis, and fuzzy clustering; particularly where the learning algorithms themselves involve fuzzy sets. For clustering itself, it would be interesting to see how a fuzzy-rough set-based approach compares with other techniques in this area. There has been some research in applying crisp rough sets to the task of clustering with promising results [62, 100]. A method based on fuzzy-rough sets should be better equipped to deal with the uncertainty and vagueness present in real world data.

In conclusion, it is worth reiterating that the topic of feature selection is highly important, particularly given the explosive growth of available information. Through this series of investigations and experiments, the potential utility of the fuzzy-rough method for feature selection has been demonstrated. Important issues affecting feature selection in general requiring further research have been highlighted and discussed. It is hoped that these developments presented in the thesis are of benefit to those working in feature selection and related areas.

Appendix A

A Brief Introduction to Fuzzy Sets

The use of fuzzy set theory is one way of capturing the vagueness present in the real world, which would otherwise be difficult using conventional set theory. This appendix starts with a quick introduction to classical set theory, using a simple example to illustrate the concept. Then, an introduction to fuzzy sets is given, covering the essentials required for a basic understanding of their use. There are many useful introductory resources regarding fuzzy set theory, for example [32, 128].

A.1 Classical Sets

In classical set theory, elements can belong to a set or not at all. For example, in the set of old people, defined here as $\{Rod, Jane, Freddy\}$, the element *Rod* belongs to this set whereas the element *George* does not. No distinction is made within a set between elements; all set members belong fully. This may be considered to be a source of information loss for certain applications. Returning to the example, *Rod* may be older than *Freddy* but by this formulation both are considered to be equally old.

More formally, let $\mathbb{U}$ be a space of objects, referred to as the universe of discourse, and x an element of $\mathbb{U}$. A classical (crisp) set A, $A \subseteq \mathbb{U}$, is defined as a collection of elements $x \in \mathbb{U}$, such that each element x can belong to the set or not belong. A classical set A can be represented by a set of ordered pairs $(x, 0)$ or $(x, 1)$ for each element, indicating $x \notin A$ or $x \in A$ respectively.

A.2 Fuzzy Sets

This concept is extended by fuzzy set theory, which allows degrees of membership of elements to sets. Previously, elements could belong fully (i.e. have a membership of 1) or not at all (a membership of 0). Fuzzy set theory relaxes this restriction by allowing memberships to take values anywhere in the range [0,1]. A fuzzy set can be defined as a set of ordered pairs $A = \{x, \mu_A(x) | x \in \mathbb{U}\}$. The function $\mu_A(x)$ is called the membership function for A, mapping each element of the universe $\mathbb{U}$ to a membership degree in the range [0.1]. The universe may be discrete or continuous. Any fuzzy set containing at least one element with a membership degree of 1 is called *normal*.

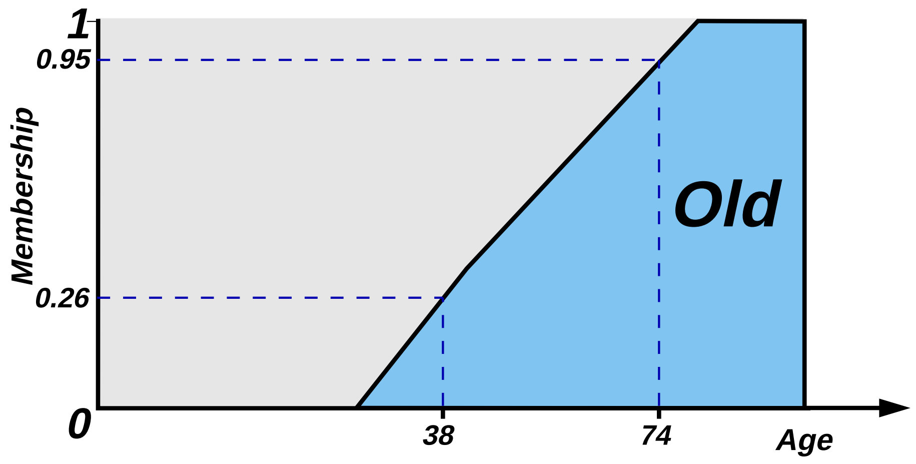


Figure A.1: Fuzzy Set representing the concept *Old*

Returning to the example, it may be better to represent the set of old people as a fuzzy set, *Old*. The membership function for this set is given in figure A.1, defined over a range of ages (the universe). Given that the age of *Rod* is 74, it can be determined that this element belongs to the set *Old* with a membership degree of 0.95. Similarly, if the age of *Freddy* is 38 the resulting degree of membership is 0.26. Here, both *Rod* and *Freddy* belong to the (fuzzy) set of old people, but *Rod* has a higher degree of membership to this set.

The specification of membership functions is typically subjective, as can be seen in this example. There are many justifiable definitions of the concept *Old*. Indeed, people of different ages may define this concept quite differently.

A.3 Fuzzy Set Operators

The most basic operators on fuzzy sets are the union, intersection and complement. These are fuzzy extensions of their crisp counterparts, ensuring that if they are applied to crisp sets, the results of their application will be identical to crisp union, intersection and complement.

A.3.1 Fuzzy Intersection

The intersection (t-norm) of two fuzzy sets, *A* and *B*, is specified by a binary operation on the unit interval; that is, a function of the form:

$$t : [0,1] \times [0,1] \rightarrow [0,1] \tag{A.1}$$

For each element x in the universe, this function takes as its arguments the memberships of x in the fuzzy sets A and B, and yields the membership grade of the element in the set constituting the intersection of A and B:

$$\mu_{A \cap B}(x) = t[\mu_A(x), \mu_B(x)] \tag{A.2}$$

The following axioms must hold for the operator t to be considered a t-norm, for all x, y and z in the range [0,1]:

- $t(x,1) = x$ (boundary condition)
- $y \leq z \rightarrow t(x,y) \leq t(x,z)$ (monotonicity)
- $t(x,y) = t(y,x)$ (commutativity)
- $t(x,t(y,z)) = t(t(x,y),z)$ (associativity)

The following are examples of t-norms that are often used as fuzzy intersections:

$$\begin{array}{rlr} t(x,y) = & min(x,y) & \text{(standard intersection)} \\ t(x,y) = & x.y & \text{(algebraic product)} \\ t(x,y) = & max(0, x+y-1) & \text{(bounded difference)} \end{array}$$

A.3.2 Fuzzy Union

The fuzzy union (t-conorm or s-norm) of two fuzzy sets A and B is specified by a function:

$$s : [0,1] \times [0,1] \rightarrow [0,1] \tag{A.3}$$

$$\mu_{A \cup B}(x) = s[\mu_A(x), \mu_B(x)] \tag{A.4}$$

A fuzzy union s is a binary operation that satisfies at least the following axioms for all x, y and z in [0,1]:

- $s(x,0) = x$ (boundary condition)
- $y \leq z \rightarrow s(x,y) \leq s(x,z)$ (monotonicity)
- $s(x,y) = s(y,x)$ (commutativity)
- $s(x,s(y,z)) = s(s(x,y),z)$ (associativity)

The following are examples of t-conorms that are often used as fuzzy unions:

$$\begin{aligned} s(x,y) = &\quad max(x,y) && \text{(standard union)} \\ s(x,y) = &\quad x + y - x.y && \text{(algebraic sum)} \\ s(x,y) = &\quad min(1, x+y) && \text{(bounded sum)} \end{aligned}$$

The most popular interpretation of fuzzy union and intersection is the max/min interpretation, primarily due to its ease of computation. This particular interpretation is used in the thesis.

Appendix B

Metric Comparison Results: Classification Datasets

This section contains the full results for the tables in section 4.6.2. The datasets were created by generating around 30 random feature values for 400 objects. Two or three features (referred to as x, y, or z) are chosen to contribute to the final boolean classification by means of an inequality. The tables show the rating given to the features from the corresponding metric. For the data presented in the first table, the first feature, x, is used to determine the classification. The values of features y and z are derived from x: $y = \sqrt{x}$, $z = x^2$. Table cells are shaded to highlight the top ranked features determined by the feature metrics. Darker shading indicates a higher ranking.

Feature	FR	Re	IG	GR	1R	χ^2
x	0.5257	0.31758	0.997	1	99.5	200
y	0.5296	0.24586	0.997	1	99.5	200
z	0.5809	0.32121	0.997	1	99.5	200
3	0.0	−0.00276	0	0	55.5	0
4	0.0	0.00148	0	0	47.5	0
5	0.0	−0.00268	0	0	44.5	0
6	0.0	−0.00221	0	0	58.5	0
7	0.0	−0.01002	0	0	52.5	0
8	0.0	−0.00649	0	0	57.5	0
9	0.0	0.00889	0	0	49.0	0
10	0.0	−0.00222	0	0	53.0	0
11	0.0	0.00182	0	0	59.5	0
12	0.0	0.00144	0	0	42.5	0
13	0.0	0.00475	0	0	50.0	0
14	0.0	0.01006	0	0	65.5	0
15	0.0	0.00613	0	0	55.5	0
16	0.0	0.00488	0	0	47.0	0
17	0.0	0.00563	0	0	56.5	0
18	0.0	0.01427	0	0	50.0	0
19	0.0	−0.00467	0	0	53.5	0
20	0.0	−0.01785	0	0	54.5	0
21	0.0	0.00327	0	0	50.0	0
22	0.0	0.00350	0	0	48.5	0
23	0.0	0.01339	0	0	51.5	0
24	0.0	0.00464	0	0	49.5	0
25	0.0	0.01334	0	0	59.0	0
26	0.0	0.01715	0	0	48.5	0
27	0.0	−0.01742	0	0	49.0	0
28	0.0	0.00685	0	0	60.5	0
29	0.0	−0.00206	0	0	53.5	0
30	0.0	0.00164	0	0	51.5	0
31	0.0	0.00171	0	0	49.0	0
32	0.0	−0.00325	0	0	51.0	0

Table B.1: Feature evaluation for $x > 0.5$, $y = \sqrt{x}$, $z = x^2$

Feature	FR	Re	IG	GR	1R	χ^2
x	0.2330	0.1862	0.2328	0.1579	86.75	128.47
y	0.2597	0.1537	0.1687	0.169	87.75	71.97
2	0.0	0.0132	0	0	84.5	0
3	0.0	0.0307	0	0	85.25	0
4	0.0	0.0320	0	0	86.0	0
5	0.0	0.0112	0	0	85.5	0
6	0.0	0.0127	0	0	86.0	0
7	0.0	0.0248	0	0	86.0	0
8	0.0	0.0219	0	0	86.0	0
9	0.0	0.0364	0	0	85.5	0
10	0.012	0.0345	0.0344	0.0576	86.5	23.638
11	0.0	0.0180	0	0	85.5	0
12	0.0	0.0246	0	0	85.75	0
13	0.0	0.0312	0	0	86.0	0
14	0.0	0.0182	0	0	86.0	0
15	0.0	0.0216	0	0	86.0	0
16	0.0	0.0245	0	0	86.0	0
17	0.0	0.0188	0	0	85.25	0
18	0.0	0.0145	0	0	85.5	0
19	0.0	0.0292	0	0	86.0	0
20	0.0	0.0132	0	0	86.0	0
21	0.0	0.0148	0	0	84.5	0
22	0.0	0.0116	0	0	86.0	0
23	0.0	0.0248	0	0	86.0	0
24	0.0	0.0190	0	0	86.0	0
25	0.0	0.0290	0	0	85.25	0
26	0.0	0.0222	0	0	86.0	0
27	0.0	0.0292	0	0	85.75	0
28	0.0	0.0307	0	0	84.75	0
29	0.0	0.0255	0	0	86.0	0
30	0.0	0.0163	0	0	86.0	0
31	0.0	0.0221	0	0	84.5	0

Table B.2: Feature evaluation for $(x+y)^2 > 0.25$

Feature	FR	Re	IG	GR	1R	χ^2
x	0.209	0.140067	0.241	0.156	79.0	119.56
y	0.2456	0.151114	0.248	0.165	78.25	122.34
2	0.0	0.008450	0	0	76.0	0
3	0.0	0.009063	0	0	73.75	0
4	0.0	0.005004	0	0	70.25	0
5	0.0	0.013202	0	0	74.75	0
6	0.0	0.011766	0	0	72.25	0
7	0.0	0.029141	0	0	73.5	0
8	0.0	0.007746	0	0	74.25	0
9	0.0	0.007245	0	0	73.5	0
10	0.0	0.018969	0	0	76.25	0
11	0.0	0.008741	0	0	75.5	0
12	0.0	0.012712	0	0	72.5	0
13	0.0	0.009962	0	0	72.25	0
14	0.0	−0.000115	0	0	75.0	0
15	0.0	0.003541	0	0	73.5	0
16	0.0	0.012629	0	0	75.0	0
17	0.0	0.019661	0	0	73.75	0
18	0.0	0.013886	0	0	76.0	0
19	0.0	0.011437	0	0	73.25	0
20	0.0	0.008366	0	0	74.25	0
21	0.0	0.017771	0	0	72.25	0
22	0.0	0.003630	0	0	74.5	0
23	0.0	0.013811	0	0	75.5	0
24	0.0	0.017560	0	0	74.5	0
25	0.0	0.003648	0	0	73.5	0
26	0.0	0.013574	0	0	72.75	0
27	0.0	0.009583	0	0	73.75	0
28	0.0	−0.000367	0	0	75.25	0
29	0.0	−0.000397	0	0	75.25	0
30	0.0	0.011544	0	0	76.25	0
31	0.0	0.007605	0	0	74.75	0

Table B.3: Feature evaluation for $(x+y)^2 > 0.5$

Feature	FR	Re	IG	GR	1R	χ^2
x	0.2445	0.1486	0.134	0.134	87.75	57.46
y	0.2441	0.1659	0.159	0.164	87.25	73.39
2	0.0	0.0229	0	0	88.5	0
3	0.0	0.0232	0	0	89.0	0
4	0.0	0.0322	0	0	88.25	0
5	0.0	0.0301	0	0	89.0	0
6	0.0	0.0252	0	0	89.0	0
7	0.0	0.0203	0	0	89.0	0
8	0.0	0.0341	0	0	89.0	0
9	0.0	0.0289	0	0	89.0	0
10	0.0	0.0339	0	0	88.5	0
11	0.0	0.0313	0	0	89.0	0
12	0.0	0.0287	0	0	89.0	0
13	0.0	0.0545	0	0	89.0	0
14	0.0	0.0458	0	0	89.0	0
15	0.0	0.0378	0	0	89.0	0
16	0.0	0.0289	0	0	89.0	0
17	0.0	0.0332	0	0	89.0	0
18	0.0	0.0306	0	0	89.0	0
19	0.0	0.0397	0	0	88.25	0
20	0.0	0.0247	0	0	89.0	0
21	0.0	0.0163	0	0	89.0	0
22	0.0	0.0330	0	0	89.0	0
23	0.0	0.0276	0	0	89.0	0
24	0.0	0.0189	0	0	88.75	0
25	0.0	0.0279	0	0	88.75	0
26	0.0	0.0252	0	0	88.75	0
27	0.0	0.0157	0	0	89.0	0
28	0.0	0.0304	0	0	89.0	0
29	0.0	0.0285	0	0	89.0	0
30	0.0	0.0315	0	0	88.75	0
31	0.0	0.0290	0	0	89.0	0

Table B.4: Feature evaluation for $(x+y)^3 < 0.125$

Feature	FR	Re	IG	GR	1R	χ^2
x	0.1057	0.0750547	0.169	0.123	64.25	73.65
y	0.0591	0.1079423	0.202	0.226	66.75	88.04
z	0.1062	0.0955878	0.202	0.160	67.5	84.28
3	0.0	0.0031390	0	0	56.75	0
4	0.0	−0.0156922	0	0	60.75	0
5	0.0	0.0088234	0	0	58.5	0
6	0.0	−0.0076636	0	0	53.25	0
7	0.0	0.0050098	0	0	57.5	0
8	0.0	0.0006841	0	0	55.75	0
9	0.0	−0.0015287	0	0	54	0
10	0.0	0.0031223	0	0	53	0
11	0.0	0.0021915	0	0	57.75	0
12	0.0	0.0027260	0	0	61.75	0
13	0.0	0.0108794	0	0	57.75	0
14	0.0	0.0008456	0	0	59.25	0
15	0.0	−0.0002930	0	0	60	0
16	0.0	−0.0018220	0	0	57.5	0
17	0.0	0.0019899	0	0	61.75	0
18	0.0	0.0090028	0	0	57.5	0
19	0.0	0.0043929	0	0	60.25	0
20	0.0	0.0006062	0	0	53.75	0
21	0.0	−0.0075626	0	0	53.75	0
22	0.0	0.0185202	0	0	57.0	0
23	0.0	−0.0056034	0	0	59.25	0
24	0.0	0.0116144	0	0	57.75	0
25	0.0	0.0001139	0	0	55.75	0
26	0.0	−0.0010561	0	0	56.25	0
27	0.0	0.0002921	0	0	54.5	0
28	0.0	0.0062014	0	0	51.75	0
29	0.0	−0.0092218	0	0	59.25	0
30	0.0	0.0000525	0	0	61.75	0
31	0.0	−0.0011460	0	0	57.0	0
32	0.0	−0.0059597	0	0	57.0	0

Table B.5: Feature evaluation for $x * y * z > 0.125$

Feature	FR	Re	IG	GR	1R	χ^2
x	0.1511	0.09800	0.1451	0.0947	76.5	65.43
y	0.1101	0.05571	0.0909	0.1080	78.0	35.36
z	0.2445	0.14736	0.2266	0.2271	79.75	93.81
3	0.0	0.00725	0	0	77.5	0
4	0.0	0.00652	0	0	78.5	0
5	0.0	0.01793	0	0	77.75	0
6	0.0	0.00716	0	0	78.0	0
7	0.0	0.02053	0	0	76.5	0
8	0.0	0.00339	0	0	78.25	0
9	0.0	0.01114	0	0	77.0	0
10	0.0	0.00409	0	0	77.75	0
11	0.0	0.01595	0	0	77.75	0
12	0.0	0.01640	0	0	77.75	0
13	0.0	0.01224	0	0	78.5	0
14	0.0	0.00170	0	0	75.5	0
15	0.0	0.00735	0	0	78.75	0
16	0.0	0.00575	0	0	78.0	0
17	0.0	0.01831	0	0	78.25	0
18	0.0	0.00508	0	0	76.0	0
19	0.0	0.01943	0	0	79.0	0
20	0.0	0.00929	0	0	78.5	0
21	0.0	0.00493	0	0	77.75	0
22	0.0	0.00579	0	0	75.75	0
23	0.0	0.01252	0	0	76.25	0
24	0.0	0.01957	0	0	79.0	0
25	0.0	0.01700	0	0	78.25	0
26	0.0	0.01175	0	0	76.5	0
27	0.0	0.01055	0	0	76.5	0
28	0.0	0.01405	0	0	78.0	0
29	0.0	0.02123	0	0	77.75	0
30	0.0	0.00884	0	0	77.5	0
31	0.0	0.01270	0	0	77.75	0
32	0.0	0.00806	0	0	77.5	0

Table B.6: Feature evaluation for $x * y * z^2 > 0.125$

Appendix C

Metric Comparison Results: Regression Datasets

This section contains the full results for 3 regression datasets. The datasets were created by generating around 30 random feature values for 400 objects. The decision value is created by means of a function involving 2 or more variables denoted x, y and, optionally, z if there is a third variable in the function. The tables show the rating given to the features from the corresponding metric.

Feature	FR	Re
x	0.156714	0.075208
y	0.183391	0.113038
2	0	−0.009917
3	0	−0.005152
4	0	0.002461
5	0	−0.000395
6	0	0.002699
7	0	0.003785
8	0	0.001662
9	0	−0.004740
10	0	0.001569
11	0	0.001495
12	0	−0.011035
13	0	−0.002583
14	0	−0.001381
15	0	−0.008143
16	0	−0.006520
17	0	−0.006878
18	0	−0.007850
19	0	−0.010232
20	0	−0.012889
21	0	−0.002477
22	0	−0.006587
23	0	−0.013506
24	0	−0.002338
25	0	−0.005290
26	0	−0.010280
27	0	−0.008658
28	0	−0.013362
29	0	0.000171
30	0	−0.006342
31	0	−0.012110
32	0	−0.006963

Table C.1: Feature evaluation for $f(x,y) = x * y$

Feature	FR	Re
x	0.227428	0.076988
y	0.165914	0.036674
z	0.203991	0.046234
3	0	−0.017056
4	0	0.004989
5	0.027450	−0.006416
6	0	−0.006045
7	0	0.003952
8	0	−0.001892
9	0	−0.010275
10	0	0.002356
11	0	−0.002328
12	0	−0.008287
13	0	−0.002969
14	0	−0.000050
15	0	0.002246
16	0	−0.006986
17	0	−0.001557
18	0	−0.007045
19	0	−0.002284
20	0	−0.006091
21	0	−0.015512
22	0.014446	−0.002249
23	0	−0.003250
24	0	0.000463
25	0	−0.010833
26	0	−0.002515
27	0	0.003845
28	0	−0.006644
29	0	−0.004005
30	0	−0.004674
31	0	−0.002399
32	0	0.001246

Table C.2: Feature evaluation for $f(x,y,z) = x*y*z$

Feature	FR	Re
x	0.128874	0.042855
y	0.123908	0.046120
z	0.170640	0.087663
3	0	−0.003307
4	0	0.000906
5	0	−0.005428
6	0	−0.009834
7	0	−0.014792
8	0	−0.006332
9	0	−0.008469
10	0	−0.000288
11	0	−0.005352
12	0	0.002245
13	0	−0.009150
14	0	−0.011524
15	0	−0.010116
16	0	−0.003124
17	0.005709	0.009794
18	0	−0.002233
19	0.005229	−0.011230
20	0	−0.005987
21	0	0.003280
22	0.011279	−0.010511
23	0	−0.004181
24	0.003855	−0.005324
25	0	0.000940
26	0	0.008235
27	0.019171	−0.013884
28	0	−0.006893
29	0	−0.011087
30	0	−0.011827
31	0	−0.007989
32	0	−0.018503

Table C.3: Feature evaluation for $f(x,y,z) = x * y * z^2$

Bibliography

[1] H. Almuallim and T.G. Dietterich. Learning with many irrelevant features. In the 9th National Conference on Artificial Intelligence. MIT Press, pp. 547–552. 1991.

[2] J.J. Alpigini, J.F. Peters, J. Skowronek, N. Zhong (Eds.): Rough Sets and Current Trends in Computing, Third International Conference, RSCTC 2002, Malvern, PA, USA, October 14-16, 2002, Proceedings. Lecture Notes in Computer Science 2475, Springer. ISBN 3-540-44274-X. 2002.

[3] C. Apté, F. Damerau and S.M. Weiss. Automated learning of decision rules for text categorization. ACM Transactions on Information Systems, Vol. 12, No. 3, pp. 233–251, 1994.

[4] J. Atkinson-Abutridy, C. Mellish and S. Aitken. Combining information extraction with genetic algorithms for text mining. IEEE Intelligent Systems, Vol. 19, No. 3, pp. 22–30. 2004.

[5] G. Attardi, A. Gullí and F. Sebastiani. Automatic Web Page Categorization by Link and Context Analysis. In Proceedings of THAI-99, 1st European Symposium on Telematics, Hypermedia and Artificial Intelligence, pp. 105–119. 1999.

[6] W.H. Au and K.C.C. Chan. An Effective Algorithm for Discovering Fuzzy Rules in Relational Databases. In Proceedings of the 7th IEEE International Conference on Fuzzy Systems, pp. 1314–1319. 1998.

[7] A.A. Bakar, M.N. Sulaiman, M. Othman, M.H. Selamat. Propositional Satisfiability Algorithm to Find Minimal Reducts for Data Mining. International Journal of Computer Mathematics, Vol. 79, No. 4, pp. 379–389. 2002.

[8] M. Balasubramanian, E.L. Schwartz, J.B. Tenenbaum, V. de Silva, and J.C. Langford. The Isomap Algorithm and Topological Stability. Science, Vol. 295, No. 5552, p. 7. 2002.

[9] R. Bayardo Jr. and R. Schrag. Using CSP look-back techniques to solve exceptionally hard SAT instances. In Proceedings of the International Conference on Principles and Practice of Constraint Programming, pp. 46–60. 1996.

[10] J. Bazan, A. Skowron, P. Synak. Dynamic reducts as a tool for extracting laws from decision tables. In Z. W. Ras, M. Zemankova (Eds.), Proceedings of the 8th Symposium on Methodologies for Intelligent Systems, Lecture Notes in Artificial Intelligence 869, Springer-Verlag, pp. 346–355. 1994.

[11] J. Bazan. A Comparison of Dynamic and non-Dynamic Rough Set Methods for Extracting Laws from Decision Tables. In Rough Sets in Knowledge Discovery, Physica-Verlag, Heidelberg, pp. 321–365. 1998.

[12] T. Beaubouef, F.E. Petry and G. Arora. Information Measures for Rough and Fuzzy Sets and Application to Uncertainty in Relational Databases. In [123]. 1999.

[13] R. Bellman. Adaptive Control Processes: A Guided Tour. Princeton University Press. 1961.

[14] M.J. Beynon. An investigation of β-reduct selection within the variable precision rough sets model. In Proceedings of the Second International Conference on Rough Sets and Current Trends in Computing (RSCTC 2000), pp 114–122. 2000.

[15] M.J. Beynon. Reducts within the Variable Precision Rough Sets Model: A Further Investigation. European Journal of Operational Research, Vol. 134, No. 3, pp. 592–605. 2001.

[16] M.J. Beynon. Stability of continuous value discretisation: an application within rough set theory. International Journal of Approximate Reasoning, Vol. 35, pp. 29–53. 2004.

[17] J.C. Bezdek, Fuzzy Mathematics in Pattern Classification. Ph.D. thesis, Center for Applied Mathematics, Cornell University. 1973.

[18] C.M. Bishop. Neural networks for pattern recognition. Oxford University Press Oxford, UK. ISBN:0-19-853849-9. 1996.

[19] A.T. Bjorvand and J. Komorowski. Practical applications of genetic algorithms for efficient reduct computation. In Proceedings of the 15th IMACS World Congress on Scientific Computation, Modelling and Applied Mathematics, Vol. 4, pp. 601–606. 1997.

[20] C.L. Blake and C.J. Merz. UCI Repository of machine learning databases. Irvine, University of California. 1998. `http://www.ics.uci.edu/˜mlearn/`.

[21] E. Bonabeau, M. Dorigo, and G. Theraulez. Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press Inc., New York, NY, USA. 1999.

[22] U.M. Braga-Neto and E.R. Dougherty. Is cross-validation valid for small-sample microarray classification? Bioinformatics, Vol. 20, No. 3, pp. 374–380. 2004.

[23] C. Bregler and S.M. Omoundro. Nonlinear image interpolation using manifold learning. In G. Tesauro, D.S. Touretzky and T.K. Leen (eds.), Advances in Neural Information Processing Systems 7, pp. 973–980. 1995.

[24] M.A. Carreira-Perpinñán. Continuous latent variable models for dimensionality reduction and sequential data reconstruction. PhD thesis, University of Sheffield, UK. 2001.

[25] W. Cedeño and D. K. Agrafiotis. Using particle swarms for the development of QSAR models based on k-nearest neighbor and kernel regression. Journal of Computer-Aided Molecular Design, Vol. 17, pp. 255–263. 2003.

[26] S. Chakrabarti, B. Dom, P. Raghaven, S. Rajagopalan, D. Gibson and J. Kleinberg. Automatic resource compilation by analyzing hyperlink structure and associated text. Proceedings of the 7th International World Wide Web Conference, pp. 65–74. 1998.

[27] K. Chan and A. Wong. APACS: A System for Automatic Analysis and Classification of Conceptual Patterns. Computational Intelligence, Vol. 6, No. 3, pp. 119–131. 1990.

[28] S. Chen, S.L. Lee and C. Lee. A new method for generating fuzzy rules from numerical data for handling classification problems. Applied Artificial Intelligence, Vol. 15, No. 7, pp. 645–664. 2001.

[29] A. Chouchoulas and Q. Shen. Rough set-aided keyword reduction for text categorisation. Applied Artificial Intelligence, Vol. 15, No. 9, pp. 843–873. 2001.

[30] A. Chouchoulas, J. Halliwell and Q. Shen. On the Implementation of Rough Set Attribute Reduction. Proceedings of the 2002 UK Workshop on Computational Intelligence, pp. 18–23. 2002.

[31] O. Cordón, M.J. del Jesus and F. Herrera. Evolutionary approaches to the learning of fuzzy rule-based classification systems. In Evolution of Enginnering and Information Systems and Their Applications. L.C. Jain (Ed.), CRC Press, pp. 107–160. 1999.

[32] E. Cox. The Fuzzy Systems Handbook: a Practitioner's Guide to Building, Using and Maintaining Fuzzy Systems. Academic Press, Inc., ISBN 0-121-94270-8. 1999.

[33] M. Dash and H. Liu. Feature Selection for Classification. Intelligent Data Analysis, Vol. 1, No. 3, pp. 131–156. 1997.

[34] M. Dash, K. Choi, P. Scheuermann and H. Liu. Feature Selection for Clustering – A Filter Solution. In Proceedings of IEEE International Conference on Data Mining (ICDM), pp. 115–122. 2002.

[35] M. Davis and H. Putnam. A Computing Procedure for Quantification Theory. Journal of the ACM, Vol. 7, No. 3, pp. 201–215. 1960.

[36] M. Davis, G. Logemann and D. Loveland. A machine program for theorem proving. Communications of the ACM, Vol. 5, pp. 394–397, 1962.

[37] S. Deerwester, S.T. Dumais, T.K. Landauer, G.W. Furnas and R.A. Harshman. Indexing by latent semantic analysis. Journal of the Society for Information Science, Vol. 41, No. 6, pp. 391–407. 1990.

[38] P. Devijver and J. Kittler. Pattern Recognition: A Statistical Approach. Prentice Hall. 1982.

[39] J. Dong, N. Zhong and S. Ohsuga. Using Rough Sets with Heuristics for Feature Selection. In New Directions in Rough Sets, Data Mining, and Granular-Soft Computing, 7th International Workshop (RSFDGrC 99), pp. 178–187. 1999.

[40] M. Dorigo, V. Maniezzo and A. Colorni. The Ant System: Optimization by a Colony of Cooperating Agents. IEEE Transactions on Systems, Man, and Cybernetics-Part B, Vol. 26, No. 1, pp. 29–41. 1996.

[41] G. Drwal and A. Mrózek. System RClass - software implementation of the rough classifier. In Proceedings of the 7th International Symposium on Intelligent Information Systems, pp. 392–395. 1998.

[42] G. Drwal. Rough and fuzzy-rough classification methods implemented in RClass system. In Proceedings of the 2nd International Conference on Rough Sets and Current Trends in Computing (RSCTC 2000), pp 152–159. 2000.

[43] D. Dubois and H. Prade. Putting rough sets and fuzzy sets together. In [171], pp. 203–232. 1992.

[44] S.T. Dumais. Combining evidence for effective information filtering. AAAI Spring Symposium on Machine Learning in Information Access Technical Papers, 1996.

[45] J. C. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. J. Cybernetics, Vol. 3, No. 3, pp. 32–57. 1973.

[46] I. Düntsch and G. Gediga. Rough Set Data Analysis. In: A. Kent & J. G. Williams (Eds.) Encyclopedia of Computer Science and Technology, Vol. 43, No. 28, pp. 281–301. 2000.

[47] I. Düntsch and G. Gediga. Rough Set Data Analysis: A road to non-invasive knowledge discovery. Bangor: Methodos. 2000.

[48] B.S. Everitt. An Introduction to Latent Variable Models, Monographs on Statistics and Applied Probability, Chapman & Hall, London. 1984.

[49] U. Fayyad, G. Piatetsky-Shapiro and P. Smyth. From data mining to knowledge discovery in databases. Artificial Intelligence Magazine, Vol. 17, No. 3, pp. 37–54. 1996.

[50] B. Flury and H. Riedwyl. Multivariate Statistics: A Practical Approach. Prentice Hall. 1988.

[51] G. Forman. An Extensive Empirical Study of Feature Selection Metrics for Text Classification. Journal of Machine Learning Research, Vol. 3, pp. 1289–1305. 2003.

[52] J.H. Friedman and J.W. Tukey. A projection pursuit algorithm for exploratory data analysis. IEEE Transactions on Computers, Vol. C-23, No. 9, pp. 881–890. 1974.

[53] J.H. Friedman and W. Stuetzle. Projection pursuit regression. Journal of the American Statistics Association, Vol. 76, pp 817–823. 1981.

[54] J.H. Friedman. Multivariate Adaptive Regression Splines. Annals of Statistics, Vol. 19, No. 1, pp. 1–141. 1991.

[55] N. Fuhr. Probabilistic Models in Information Retrieval. Computer Journal, Vol. 35, No. 3, pp. 243–55. 1992.

[56] N. Fuhr, N. Gövert, M. Lalmas, and F. Sebastiani. Categorisation tool: Final prototype. Deliverable 4.3, Project LE4-8303 “EUROSEARCH”, Commission of the European Communities. 1999.

[57] D. Gering. Linear and nonlinear data dimensionality reduction. Technical report, Massachusetts Institute of Technology. 2002.

[58] J. Gleick. Chaos: Making New Science. Penguin Books. ISBN: 0140092501. 1988.

[59] H. Handels, T. Roß, J. Kreusch, H.H. Wolff and S. Pöpple. Feature Selection for Optimized Skin Tumor Recognition using Genetic Algorithms. Artificial Intelligence in Medicine, Vol. 16, No. 3, pp. 283–297. 1999.

[60] J.A. Hartigan and M.A. Wong. A K-Means Clustering Algorithm. Applied Statistics, Vol. 28, No. 1, pp. 100–108. 1979.

[61] I. Hayashi, T. Maeda, A. Bastian and L.C. Jain. Generation of Fuzzy Decision Trees by Fuzzy ID3 with Adjusting Mechanism of AND/OR Operators. In Proceedings of the 7th IEEE International Conference on Fuzzy Systems, pp. 681–685. 1998.

[62] T.B. Ho, S. Kawasaki and N.B. Nguyen. Documents clustering using tolerance rough set model and its application to information retrieval. Studies In Fuzziness And Soft Computing, Intelligent Exploration of the Web, pp. 181–196. 2003.

[63] U. Höhle. Quotients with respect to similarity relations. Fuzzy Sets and Systems, Vol. 27, No. 1, pp. 31–44. 1988.

[64] J. Holland. Adaptation In Natural and Artificial Systems. The University of Michigan Press, Ann Arbour. 1975.

[65] R.C. Holte. Very simple classification rules perform well on most commonly used datasets. Machine Learning, Vol. 11, No. 1, pp. 63–90. 1993.

[66] H.H. Hoos and T. Stützle. Towards a Characterisation of the Behaviour of Stochastic Local Search Algorithms for SAT. Artificial Intelligence, Vol. 112, pp. 213–232. 1999.

[67] E. Hunt, J. Martin and P. Stone. Experiments in Induction. New York Academic Press. 1966.

[68] Internet News. '1.5 Million Pages Added To Web Each Day, Says Research Company', September 1st, 1998. `http://www.internetnews.com/bus-news/article.php/37891`.

[69] C.Z. Janikow. Fuzzy Decision Trees: Issues and Methods. IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics, Vol. 28, No. 1, pp. 1–14. 1998.

[70] J. Jelonek and J. Stefanowski. Feature subset selection for classification of histological images. Artificial Intelligence in Medicine, Vol. 9, No. 3, pp. 227–239. 1997.

[71] R. Jensen and Q. Shen. A Rough Set-Aided System for Sorting WWW Bookmarks. In N. Zhong et al. (Eds.), Web Intelligence: Research and Development, pp. 95–105. 2001.

[72] R. Jensen and Q. Shen. Rough and fuzzy sets for dimensionality reduction. In Proceedings of the 2001 UK Workshop on Computational Intelligence, pp. 69–74. 2001.

[73] R. Jensen and Q. Shen. Fuzzy-Rough Sets for Descriptive Dimensionality Reduction. In Proceedings of the 11th International Conference on Fuzzy Systems, pp. 29–34. 2002.

[74] R. Jensen and Q. Shen. Aiding Fuzzy Rule Induction with Fuzzy-Rough Attribute Reduction. In Proceedings of the 2002 UK Workshop on Computational Intelligence, pp. 81–88. 2002.

[75] R. Jensen and Q. Shen. Using Fuzzy Dependency-Guided Attribute Grouping in Feature Selection. In Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, 9th International Conference, RSFDGrC 2003, LNCS 2639, Springer, pp. 250–255. 2003.

[76] R. Jensen and Q. Shen. Finding Rough Set Reducts with Ant Colony Optimization. In Proceedings of the 2003 UK Workshop on Computational Intelligence, pp 15–22. 2003.

[77] R. Jensen and Q. Shen. Fuzzy-Rough Attribute Reduction with Application to Web Categorization. Fuzzy Sets and Systems, Vol. 141, No. 3, pp. 469–485. 2004.

[78] R. Jensen and Q. Shen. Semantics-Preserving Dimensionality Reduction: Rough and Fuzzy-Rough Based Approaches. IEEE Transactions on Knowledge and Data Engineering, Vol. 16, No. 12, pp. 1457–1471. 2004.

[79] R. Jensen and Q. Shen. Fuzzy-Rough Data Reduction with Ant Colony Optimization. Fuzzy Sets and Systems, Vol. 149, No. 1, pp. 5–20. 2005.

[80] Y. Jin. Fuzzy modeling of high-dimensional systems: complexity reduction and interpretability improvement. IEEE Transactions on Fuzzy Systems, Vol. 8, No. 2, pp. 212–221. 2000.

[81] G.H. John, R. Kohavi and K. Pfleger. Irrelevant Features and the Subset Selection Problem. Proceedings of the 11th International Conference on Machine Learning ICML94, pp. 121–129. 1994.

[82] L.T. Jolliffe. Principal Component Analysis. Springer Series in Statistics, Springer-Verlag, Berlin. 1986.

[83] K. Kira and L.A. Rendell. The feature selection problem: Traditional methods and a new algorithm. In Proceedings of Ninth National Conference on Artificial Intelligence, pp. 129–134. 1992.

[84] S. Kirkpatrick, C. Gelatt and M. Vecchi. Optimization by simulated annealing. Science, Vol. 220, No. 4598, pp. 671–680. 1983.

[85] D. Koller and M. Sahami. Toward optimal feature selection. In Proceedings of International Conference on Machine Learning, pp. 284–292. 1996.

[86] J. Komorowski, Z. Pawlak, L. Polkowski and A. Skowron. Rough Sets: A Tutorial. In [123], pp. 3–98. 1999.

[87] I. Kononenko. Estimating attributes: Analysis and Extensions of RELIEF. Proceedings of the European Conference on Machine Learning, pp. 171–182. 1994.

[88] B. Kosko. Fuzzy entropy and conditioning. Information Sciences, Vol. 40, No. 2, pp. 165–174. 1986.

[89] M.A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. AIChE Journal, Vol. 37, No. 2, pp. 233–243. 1991.

[90] J.B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. Psychometrika, Vol. 29, No. 1, pp. 1–27. 1964.

[91] M. Kryszkiewicz. Maintenance of reducts in the variable precision rough sets model. ICS Research Report 31/94, Warsaw University of Technology. 1994.

[92] M. Kudo and J. Skalansky. Comparison of algorithms that select features for pattern classifiers. Pattern Recognition, Vol. 33, No. 1, pp. 25–41. 2000.

[93] N. Kwak and C.H. Choi. Input feature selection for classification problems. IEEE Transactions on Neural Networks, Vol. 13, No. 1, pp. 143–159. 2002.

[94] I.M.F. Langlands. Accounting Issues Surrounding the Split Capital Investment Trust Crisis of 2001 and 2002. M.A. Thesis, The University of Edinburgh. 2004.

[95] P. Langley. Selection of relevant features in machine learning. In Proceedings of the AAAI Fall Symposium on Relevance, pp. 1–5. 1994.

[96] K. Larson and M. Czerwinski. Web page design: implications of memory, structure and scent for information retrieval. In Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems, pp. 25–32. 1998.

[97] M. H. Law, M. A. T. Figueiredo and A. K. Jain. Feature selection in mixture-based clustering. In Advances in Neural Information Processing Systems 15, pp. 609–616. 2002.

[98] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan and D.B. Shmoys. The Travelling Salesman Problem. John Wiley & Sons, Chichester, UK. 1985.

[99] W.S. Li, Q. Vu, D. Agrawal, Y. Hara and H. Takano. PowerBookmarks: a system for personalizable Web information organization, sharing, and management. Computer Networks: The International Journal of Computer and Telecommunications Networking, Vol. 31, No. 11-16, pp. 1375–1389. 1999.

[100] P. Lingras, R. Yan and C. West. Comparison of Conventional and Rough K-Means Clustering. In Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, 9th International Conference, RSFDGrC 2003, LNCS 2639, Springer, pp. 130–137. 2003.

[101] H. Liu and R. Setiono. Chi2: Feature selection and discretization of numeric attributes. In Proceedings of the 7th IEEE International Conference on Tools with Artificial Intelligence, pp. 336–391. 1995.

[102] H. Liu and R. Setiono. A probabilistic approach to feature selection - a filter solution. In Proceedings of the 9th International Conference on Industrial and Engineering Applications of AI and ES, pp. 284–292. 1996.

[103] H. Liu and R. Setiono. Feature selection and classification - a probabilistic wrapper approach. In Proceedings of the 9th International Conference on Industrial and Engineering Applications of AI and ES, pp. 419–424. 1996.

[104] H. Liu, H. Motoda (Eds). Feature Extraction, Construction and Selection: A Data Mining Perspective (Kluwer International Series in Engineering & Computer Science), Kluwer Academic Publishers. 1998.

[105] A. Lozowski, T.J. Cholewo and J.M. Zurada. Crisp rule extraction from perceptron network classifiers. Proceedings of the IEEE International Conference on Neural Networks, volume of Plenary, Panel and Special Sessions, pp. 94–99. 1996.

[106] Y.S. Maarek, I.Z. Ben Shaul. Automatically Organizing Bookmarks per Contents. Computer Networks and ISDN Systems, Vol. 28, No. 7-11, pp. 1321–1333. 1996.

[107] V. Maniezzo and A. Colorni. The Ant System Applied to the Quadratic Assignment Problem. Knowledge and Data Engineering, Vol. 11, No. 5, pp. 769–778. 1999.

[108] K.V. Mardia, J.T. Kent, and J.M. Bibby. Multivariate Analysis, Probability and Mathematical Statistics Series, Academic Press, New York. 1979.

[109] J. G. Marin-Blázquez and Q. Shen. From approximative to descriptive fuzzy classifiers. IEEE Transactions on Fuzzy Systems, Vol. 10, No. 4, pp. 484–497. 2002.

[110] S.J. Messick and R.P. Abelson. The additive constant problem in multidimensional scaling. Psychometrika, Vol. 21, pp. 1–17. 1956.

[111] A.J. Miller. Subset selection in regression. Chapman and Hall. 1990.

[112] T. Mitchell. Machine Learning. McGraw-Hill. 1997.

[113] D. Mladenic. Text-learning and related intelligent agents: a survey. IEEE Intelligent Systems, Vol. 14, No. 4, pp. 44–54. 1999.

[114] D. Mladenic, M. Grobelnik. Feature selection for unbalanced class distribution and Naive Bayes. In Proceedings of the 16th International Conference on Machine Learning, ICML 99, pp. 258–267. 1999.

[115] M. Modrzejewski. Feature selection using rough sets theory. In Proceedings of the 11th International Conference on Machine Learning, pp. 213–226. 1993.

[116] A. Moukas and P. Maes. Amalthaea: An evolving Multi-Agent Information Filtering and Discovery System for the WWW. Journal of Autonomous Agents and Multi-Agent Systems, Vol. 1, No. 1, pp. 59–88. 1998.

[117] I. Moulinier. A Framework for Comparing Text Categorization Approaches. In Proceedings of the 1996 AAAI Spring Symposium on Machine Learning in Information Access. 1996.

[118] H.T. Ng , W.B. Goh and K.L. Low. Feature selection, perceptron learning, and a usability case study for text categorisation. In Proceedings of SIGIR-97, the 20th ACM International Conference on Research and Development in Information Retrieval, pp. 67–73. 1997.

[119] S.H. Nguyen and H.S. Nguyen. Some efficient algorithms for rough set methods. In Proceedings of the Conference of Information Processing and Management of Uncertainty in Knowledge-Based Systems, pp. 1451–1456. 1996.

[120] H.S. Nguyen and A. Skowron. Boolean Reasoning for Feature Extraction Problems. In Proceedings of the 10th International Symposium on Methodologies for Intelligent Systems, pp. 117–126. 1997.

[121] L.S. Oliveira, N. Benahmed, R. Sabourin, F. Bortolozzi and C.Y. Suen. Feature Subset Selection Using Genetic Algorithms for Handwritten Digit Recognition. In Proceedings of the 14th Brazilian Symposium on Computer Graphics and Image Processing, pp. 362–369. 2001.

[122] P. Paclik, R.P.W. Duin, G.M.P. van Kempen and R. Kohlus. On feature selection with measurement cost and grouped features. Proceedings of the 4th International Workshop on Statistical Techniques in Pattern Recognition (SPR2002), pp. 461–469. 2002.

[123] S.K. Pal and A. Skowron (Eds.). Rough-Fuzzy Hybridization: A New Trend in Decision Making. Springer Verlag, Singapore. 1999.

[124] Z. Pawlak. Rough Sets. International Journal of Computer and Information Sciences, Vol. 11, No. 5, pp. 341–356. 1982.

[125] Z. Pawlak. Rough Sets: Theoretical Aspects of Reasoning About Data. Kluwer Academic Publishing, Dordrecht. 1991.

[126] Z. Pawlak and A. Skowron. Rough membership functions. In R. Yager, M. Fedrizzi, J. Kacprzyk (eds.), Advances in the Dempster-Shafer Theory of Evidence, Wiley, New York, pp. 251–271. 1994.

[127] K. Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can reasonably supposed to have arisen from random sampling. Philosophical Magazine, Series 5, pp. 157–175. 1900.

[128] W. Pedrycz, and F. Gomide. An Introduction to Fuzzy Sets: Analysis and Design. The MIT Press. 1998.

[129] W. Pedrycz. Shadowed sets: bridging fuzzy and rough sets. In [123], pp. 179–199. 1999.

[130] W. Pedrycz and G. Vukovich. Feature analysis through information granulation. Pattern Recognition, Vol. 35, No. 4, pp. 825–834. 2002.

[131] J. Platt. Fast Training of Support Vector Machines using Sequential Minimal Optimization. Advances in Kernel Methods: Support Vector Learning (B. Schlkopf, C. Burges, and A. Smola, eds.), pp. 185–208. The MIT Press. 1998.

[132] L. Polkowski, T.Y. Lin, S. Tsumoto (eds.), Rough Set Methods and Applications: New Developments in Knowledge Discovery in Information Systems, Vol. 56 Studies in Fuzziness and Soft Computing, Physica-Verlag, Heidelberg, Germany. 2000.

[133] L. Polkowski. Rough Sets: Mathematical Foundations. Advances in Soft Computing. Physica Verlag, Heidelberg, Germany. 2002.

[134] J.R. Quinlan. Induction of decision trees. Machine Learning Vol. 1, pp. 81–106. 1986.

[135] J.R. Quinlan. C4.5: Programs for Machine Learning. The Morgan Kaufmann Series in Machine Learning. Morgan Kaufmann Publishers, San Mateo, CA. 1993.

[136] R. Rallo, J. Ferré-Giné and F. Giralt. Best Feature Selection and Data Completion for the Design of Soft Neural Sensors. Proceedings of AIChE 2003, 2nd Topical Conference on Sensors, San Francisco. 2003.

[137] B. Raman and T.R. Ioerger. Instance-based filter for feature selection. Journal of Machine Learning Research, Vol. 1, pp. 1–23. 2002.

[138] M.W. Richardson. Multidimensional psychophysics. Psychological Bulletin, Vol. 35, pp. 659–660. 1938.

[139] The ROSETTA homepage. `http://rosetta.lcb.uu.se/general/`

[140] RSES: Rough Set Exploration System.
`http://logic.mimuw.edu.pl/˜rses/`

[141] S.T. Roweis and L.K. Saul. Nonlinear Dimensionality Reduction by Locally Linear Embedding. Science, Vol. 290, No. 5500, pp. 2323–2326. 2000.

[142] M.E. Ruiz and P. Srinivasan. Hierarchical neural networks for text categorization. In Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval, pp 281–282. 1999.

[143] Y. Saeys, S. Degroeve, D. Aeyels, P. Rouze and Y. Van De Peer. Feature selection for splice site prediction: A new method using EDA-based feature ranking. BMC Bioinformatics, Vol. 5, doi: 10.1186/1471-2105-5-64. 2004.

[144] G. Salton, A. Wong and C.S. Yang. A vector space model for automatic indexing. Communications of the ACM, Vol. 18, No. 11, pp. 613–620. 1975.

[145] G. Salton, E.A. Fox and H. Wu. Extended Boolean Information Retrieval. Communications of the ACM, Vol. 26, No. 12, pp. 1022–1036. 1983.

[146] G. Salton, Introduction to Modern Information Retrieval. McGraw-Hill. 1983.

[147] G. Salton, and C. Buckley. Term Weighting Approaches in Automatic Text Retrieval. Technical Report TR87-881, Department of Computer Science, Cornell University. 1987.

[148] J.C. Schlimmer. Efficiently inducing determinations - A complete and systematic search algorithm that uses optimal pruning. International Conference on Machine Learning (ICML 93), pp. 284–290. 1993.

[149] B. Schölkopf. Support Vector Learning. R. Oldenbourg Verlag, Munich. 1997.

[150] M. Schroeder. Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise. W.H. Freeman and Company, New York. 1991.

[151] F. Sebastiani. Machine learning in Automated Text Categorisation. ACM Computing Surveys, Vol. 34, No. 1, pp. 1–47. 2002.

[152] M. Sebban and R. Nock. A hybrid filter/wrapper approach of feature selection using information theory. Pattern Recognition, Vol. 35, No. 4, pp. 835–846. 2002.

[153] B. Selman and H.A. Kautz. Domain-independant extensions to GSAT : Solving large structured variables. In Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI'93), pp. 290–295. 1993.

[154] R. Setiono and H. Liu. Neural network feature selector. IEEE Transactions on Neural Networks, Vol.8, No. 3, pp. 645–662. 1997.

[155] H. Sever. The status of research on rough sets for knowledge discovery in databases. In Proceedings of the Second International Conference on Nonlinear Problems in Aviation and Aerospace (ICNPAA98), Vol. 2, pp. 673–680. 1998.

[156] C. Shang and Q. Shen. Rough Feature Selection for Neural Network Based Image Classification. International Journal of Image and Graphics, Vol. 2, No. 4, pp. 541–556. 2002.

[157] Q. Shen and A. Chouchoulas. A fuzzy-rough approach for generating classification rules. Pattern Recognition, Vol. 35, No. 11, pp. 341–354. 2002.

[158] Q. Shen and R. Jensen. Selecting Informative Features with Fuzzy-Rough Sets and its Application for Complex Systems Monitoring. Pattern Recognition, Vol. 37, No. 7, pp. 1351–1363. 2004.

[159] R. N. Shepard. The Analysis of Proximities: Multidimensional Scaling with an Unknown Distance Function, I, II. Psychometrika, Vol. 27, pp. 125–140, 219–246. 1962.

[160] H. Shütze, D.A. Hull and J.O. Pederson. A comparison of classifiers and document representations for the routing problem. In Proceedings of SIGIR-95, 18th ACM International Conference on Research and Development in Information Retrieval, pp. 229–237. 1995.

[161] W. Siedlecki and J. Sklansky. On automatic feature selection. International Journal of Pattern Recognition and Artificial Intelligence, Vol. 2, No. 2, pp. 197–220. 1988.

[162] W. Siedlecki and J. Sklansky. A note on genetic algorithms for large-scale feature selection. Pattern Recognition Letters, Vol. 10, No. 5, pp. 335–347. 1989.

[163] S. Singh, M. Singh and M. Markou. Feature Selection for Face Recognition based on Data Partitioning. In Proceedings of the 15th International Conference on Pattern Recognition (ICPR 02), pp. 680–683. 2002.

[164] A. Skowron and C. Rauszer. The discernibility matrices and functions in Information Systems. In [171], pp. 331–362. 1992.

[165] A. Skowron and J.W. Grzymala-Busse. From rough set theory to evidence theory. In Advances in the Dempster-Shafer Theory of Evidence, (R. Yager, M. Fedrizzi, and J. Kasprzyk eds.), John Wiley & Sons, Inc. 1994.

[166] A. Skowron and J. Stepaniuk. Tolerance Approximation Spaces. Fundamenta Informaticae, Vol. 27, No. 2, pp. 245–253. 1996.

[167] A. Skowron, Z. Pawlak, J. Komorowski and L. Polkowski. A rough set perspective on data and knowledge. Handbook of data mining and knowledge discovery, pp. 134–149, Oxford University Press. 2002.

[168] A. Skowron, S.K. Pal. Special issue: Rough sets, pattern recognition and data mining. Pattern Recognition Letters, Vol. 24, No. 6, pp. 829–933. 2003.

[169] D. Ślęzak. Approximate reducts in decision tables. In Proceedings of the 6th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU ’96), pp. 1159–1164. 1996.

[170] D. Ślęzak. Normalized decision functions and measures for inconsistent decision tables analysis. Fundamenta Informaticae, Vol. 44, No. 3, pp. 291–319. 2000.

[171] R. Slowinski, editor. Intelligent Decision Support. Kluwer Academic Publishers, Dordrecht. 1992.

[172] R. Slowinski and D. Vanderpooten. Similarity relation as a basis for rough approximations. In Advances in Machine Intelligence and Soft Computing, Vol. 4, pp. 17–33. 1997.

[173] M.G. Smith and L. Bull. Feature construction and selection using genetic programming and a genetic algorithm. Proceedings of the 6th European Conference on Genetic Programming (EuroGP 2003), pp. 229–237. 2003.

[174] P. Srinivasan, M.E. Ruiz, D.H. Kraft and J. Chen. Vocabulary mining for information retrieval: rough sets and fuzzy sets. Information Processing & Management, Vol. 37, No. 1, pp. 15–38. 1998.

[175] J. Stefanowski and A. Tsoukiàs. Valued Tolerance and Decision Rules. Rough Sets and Current Trends in Computing, pp. 212–219. 2000.

[176] R.W. Swiniarski. Rough set expert system for online prediction of volleyball game progress for US olympic team. In Proceedings of the 3rd Biennial European Joint Conference on Engineering Systems Design Analysis, pp. 15–20. 1996.

[177] R.W. Swiniarski and A. Skowron. Rough set methods in feature selection and recognition. Pattern Recognition Letters, Vol. 24, No. 6, pp. 833–849. 2003.

[178] J.B. Tenenbaum, V. de Silva, J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. Science, Vol. 290, No. 5500, pp. 2319–2323. 2000.

[179] H. Thiele. Fuzzy rough sets versus rough fuzzy sets - an interpretation and a comparative study using concepts of modal logics. Technical report no. CI-30/98, University of Dortmund. 1998.

[180] W.S. Torgerson. Psychometrika, Vol. 17, pp. 401–419. 1952.

[181] C. Traina Jr, A. Traina, L. Wu and C. Faloutsos. Fast feature selection using the fractal dimension. In Proceedings of the 15th Brazilian Symposium on Databases (SBBD), pp. 158–171. 2000.

[182] M. Umano, H. Okamota, I. Hatono, H. Tamura, F. Kawachi, S. Umedzu and J. Kinoshita. Generation of fuzzy decision trees by fuzzy ID3 algorithm and its application to diagnosis by gas in oil. In Proceedings of the 1994 Japan-USA Symposium on Flexible Automation, pp. 1445–1448. 1994.

[183] C.J. van Rijsbergen. Information Retrieval. Butterworths, London, United Kingdom. 1979. `http://www.dcs.gla.ac.uk/Keith/Preface.html`.

[184] T. Walsh. SAT v CSP. In Proceedings of the International Conference on Principles and Practice of Constraint Programming, pp. 441–456. 2000.

[185] J. Wang and J. Wang. Reduction Algorithms Based on Discernibility Matrix: The Ordered Attributes Method. Journal of Computer Science & Technology, Vol. 16, No. 6, pp. 489–504. 2001.

[186] I.H. Witten and E. Frank. Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann, San Francisco. 2000.

[187] J. Wróblewski. Finding minimal reducts using genetic algorithms. In Proceedings of the 2nd Annual Joint Conference on Information Sciences, pp. 186–189. 1995.

[188] M. Wygralak. Rough sets and fuzzy sets - some remarks on interrelations. Fuzzy Sets and Systems, Vol. 29, No. 2, pp. 241–243. 1989.

[189] E.P. Xing. Feature Selection in Microarray Analysis. A Practical Approach to Microarray Data Analysis, Kluwer Academic Publishers. 2003.

[190] M. Xiong, W. Li, J. Zhao, L. Jin and E. Boerwinkle. Feature (Gene) Selection in Gene Expression-Based Tumor Classification. Molecular Genetics and Metabolism, Vol. 73, No. 3, pp. 239–247. 2001.

[191] N. Xiong and L. Litz. Reduction of fuzzy control rules by means of premise learning - method and case study. Fuzzy Sets and Systems, Vol. 132, No. 2, pp. 217–231. 2002.

[192] Yahoo. `www.yahoo.com`

[193] Y. Yang and J.O. Pedersen. A Comparative Study on Feature Selection in Text Categorization. In Proceedings of the 14th International Conference on Machine Learning (ICML 97), pp. 412–420. 1997.

[194] J. Yang and V. Honavar. Feature subset selection using a genetic algorithm. IEEE Intelligent Systems, Vol. 13, No. 1, pp. 44–49. 1998.

[195] J. Yao. Feature selection for fluoresence image classification. KDD Lab Proposal, CALD, SCS, CMU. 2001.

[196] Y.Y. Yao. A Comparative Study of Fuzzy Sets and Rough Sets. Information Sciences, Vol. 109, No. 1-4, pp. 21–47. 1998.

[197] F.W. Young and R.M. Hamer. Theory and Applications of Multidimensional Scaling. Eribaum Associates. Hillsdale, NJ. 1994.

[198] S. Yu, S. De Backer and P. Scheunders. Genetic feature selection combined with composite fuzzy nearest neighbor classifiers for hyperspectral satellite imagery. Pattern Recognition Letters, Vol. 23, No. 1-3, pp. 183–190. 2002.

[199] Y. Yuan and M. J. Shaw. Induction of fuzzy decision trees. Fuzzy Sets and Systems, Vol. 69, No. 2, pp. 125–139. 1995.

[200] L.A. Zadeh. Fuzzy sets. Information and Control, Vol. 8, pp. 338–353. 1965.

[201] L.A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning. Information Sciences, Vol. 8, pp. 199–249, 301–357; Vol 9, pp. 43–80. 1975.

[202] L. Zhang and S. Malik. The Quest for Efficient Boolean Satisfiability Solvers. In Proceedings of the 18th International Conference on Automated Deduction, pp. 295–313. 2002.

[203] N. Zhong, J. Dong and S. Ohsuga. Using Rough Sets with Heuristics for Feature Selection. Journal of Intelligent Information Systems, Vol. 16, No. 3, pp. 199–214. 2001.

[204] W. Ziarko. Variable Precision Rough Set Model. Journal of Computer and System Sciences, Vol. 46, No. 1, pp. 39–59. 1993.